BGD Siber Güvenlik ve Savunma Kitap Serisi 5

Siber Güvenlik ve Savunma

BLOKZİNCİR VE KRİPTOLOJİ

Editörler
Prof. Dr. Şeref Sağıroğlu
Doç. Dr. Sedat Akleylek

nobel

SİBER GÜVENLİK VE SAVUNMA
Blokzincir ve Kriptoloji
Editörler: Prof. Dr. Şeref Sağıroğlu - Doç. Dr. Sedat Akleylek

Yayın No.: 3758
Mühendislik/Teknik No.: 348
ISBN: 978-625-417-110-9
E-ISBN: 978-625-417-111- 6
Basım Sayısı: 1. Basım, Kasım 2021

Genel Yayın Yönetmeni: Nevzat Argun -nargun@nobelyayin.com-
Genel Yayın Koordinatörü: Gülfem Dursun -gulfem@nobelyayin.com-

Sayfa Tasarım: Tarkan Kara -erdal@nobelyayin.com-
Redaksiyon: Buse Gamze Çeliktaş -buse@nobelyayin.com-
Kapak Tasarım: Sezai Özden -sezai@nobelyayin.com-
Görsel Tasarım Uzmanı: Mehtap Yürümez -mehtap@nobelyayin.com-
Baskı Sorumlusu: Yavuz Şahin -yavuz@nobelyayin.com-
Baskı ve Cilt: Sarıyıldız Ofset Amb. Kağ. Paz. San. ve Tic. Ltd Sertifika No.: 23593
İvedik Ağaç İşleri San. Sit. 1354. Cad. 1358. Sok. No.: 31 Ostim / ANKARA

Kütüphane Bilgi Kartı

Sağıroğlu, Şeref., Akleylek, Sedat.
SİBER GÜVENLİK VE SAVUNMA - Blokzincir ve Kriptoloji / Şeref Sağıroğlu, Sedat Akleylek
1. Basım, XX + 596 s., 16x23,5 cm. Kaynakça ve dizin var.
ISBN: 978-625-417-110-9
E-ISBN: 978-625-417-111- 6
1. Blokzincirde Güvenli ve Güvenilir Uygulama Geliştirme Temelleri 2. Blokzincirinde Uzlaşı Mekanizmaları 3. Kimlik Sistemlerinde Blokzincir Kullanımı 4. Blokzinciri ve Yazılım Tanımlı Ağlar 5. Blokzincirinin Askeri Lojistik Takip Sistemlerinde Kullanılması 6. Kuantum Bilgisayar Çağında Kriptosistemlere Bir Bakış 7. Açık Anahtarlı Kriptosistemler İçin Verimli Sıkıştırma Uygulamaları 8. Kriptografide Rastgelelik 9. Simetrik Sistemlerde Kriptoanaliz Yöntemleri 10. Sır Paylaşım Şemaları ve Blokzincir 11. Kafes Tabanlı Grup İmzalama Şemalarının Özellikleri ve Değerlendirilmesi 12. Yayın Şifreleme Sistemleri

Genel Dağıtım

ATLAS AKADEMİK BASIM YAYIN DAĞITIM TİC. LTD. ŞTİ.
Adres: Bahçekapı Mh. 2465 Sk. Oto Sanayi Sitesi No:7 Bodrum Kat, Şaşmaz/ANKARA
Telefon: +90 312 278 50 77 - **Faks:** 0 312 278 21 65 - **Sipariş:** siparis@nobelyayin.com-
E-Satış: www.nobelkitap.com - esatis@nobelkitap.com / www.atlaskitap.com - info@atlaskitap.com

Dağıtım ve Satış Noktaları: Alfa Basım Dağıtım, Arasta, Arkadaş Kitabevi, D&R Mağazaları, Dost Dağıtım, Ekip Dağıtım, Kida Dağıtım, Kitapsan, Nezih Kitabevleri, Pandora, Prefix, Remzi Kitabevleri

BÖLÜM YAZARLARI

Bölüm 1
BLOKZİNCİRDE GÜVENLİ VE GÜVENİLİR UYGULAMA GELİŞTİRME TEMELLERİ
Enis KARAARSLAN - Melih BİRİM

Bölüm 2
BLOKZİNCİRİNDE UZLAŞI MEKANİZMALARI
Murat OSMANOĞLU

Bölüm 3
KİMLİK SİSTEMLERİNDE BLOKZİNCİR KULLANIMI
Serkan AYVAZ - Salih Cemil ÇETİN - Mehmet AYDAR

Bölüm 4
BLOKZİNCİRİ VE YAZILIM TANIMLI AĞLAR
Murat KARAKUŞ

Bölüm 5
BLOKZİNCİRİNİN ASKERİ LOJİSTİK TAKİP SİSTEMLERİNDE KULLANILMASI
Enis KONACAKLI - Enis KARAARSLAN

Bölüm 6
KUANTUM BİLGİSAYAR ÇAĞINDA KRİPTOSİSTEMLERE BİR BAKIŞ
Sedat AKLEYLEK - Kübra SEYHAN

Bölüm 7
AÇIK ANAHTARLI KRİPTOSİSTEMLER İÇİN VERİMLİ SIKIŞTIRMA UYGULAMALARI
Melek ÇİL - Barış Bülent KIRLAR

Bölüm 8
KRİPTOGRAFİDE RASTGELELİK
Muhiddin UĞUZ

Bölüm 9
SİMETRİK SİSTEMLERDE KRİPTOANALİZ YÖNTEMLERİ
Mehmet Emin GÖNEN - Orhun KARA - Ferhat KARAKOÇ

BİLGİ GÜVENLİĞİ DERNEĞİ'NDEN

Bilgi Güvenliği Derneği (BGD); Bilgi Güvenliği ve Siber Güvenlik ve Savunma alanında toplumun her kesimini bilgilendirmek, farkındalığını artırmak, teknolojik gelişmeleri izlemek, yerli ve milli teknolojilerin geliştirilmesine katkı sağlamak; bireysel, kurumsal ve ulusal düzeydeki riskler konusunda farkındalık oluşturmak ve kamu-sektör-üniversite işbirliklerini geliştirmek ve en önemlisi ise ulusal strateji ve politikalara katkılar sağlamak amacı ile 22.07.2007 tarihinde kurulmuştur.

BGD'nin vizyonu; "bilgi güvenliği alanında ulusal ve uluslararası düzeyde tarafsız, güvenilir ve etkin bir ulusal sivil toplum kuruluşu olmaktır." BGD amacı doğrultusunda; tüm paydaşlarla işbirliği yaparak mevzuatın oluşturulmasında ve geliştirilmesinde aktif rol almakta, gerçekleştirdiği konferans, sempozyum, çalıştay ve eğitimler, yayımladığı rapor ve yazılar ile farkındalığın oluşmasına ve bunun davranışa dönüştürülmesine katkılar sağlamaktadır.

Derneğimiz bu kapsamda; "Ulusal Siber Güvenlik Strateji Belgesi" ve "Ulusal Siber Güvenlik Eylem Planı" hazırlanmasına öncülük etmiş, hazırladığı taslak metinler kabul görmüş ve sonuçta ülkemizin siber güvenlik stratejisi ve eylem planlarının gecikmeden yayımlanasına katkı sağlamıştır. Aynı zamanda; bu alanda nitelikli insan kaynağı yetiştirilmesi, mesleki yeterliliklerin belirlenmesi, kamu-endüstri-üniversite işbirliklerinin geliştirilmesi, kümelenme çalışmaları gibi önemli politika ve stratejilerin oluşturulmasında etkin rol üstlenmiş, bu görevini de sürdürmeye devam etmektedir.

BGD amacı doğrultusunda; tüm paydaşlarla iş birliği yaparak mevzuatın oluşturulmasında ve geliştirilmesinde aktif rol almakta, gerçekleştirdiği konferans, sempozyum, çalıştay ve eğitimler ile yayımladığı rapor ve yazılar ile farkındalığın oluşmasına katkı sağlamaktadır.

BGD; "Uluslararası Bilgi Güvenliği ve Kriptoloji Konferansı", "Ulusal Siber Güvenlik Stratejisi Çalıştayı", "Veri Merkezleri ve Siber Güvenlik Çalıştayı", "Siber Güvenlik Hukuku Çalıştayı", "Mobil Dünyada Çocuk ve Gençlerin Güvenliği Sempozyumu", "IPv6 Konferansı", "Kritik Enerji Altyapılarının

Korunması Sempozyumu", "Ulusal Siber Terör Konferansı", "Siber Güvenlik Yaz Kampı" gibi etkinlikleri düzenleyerek ve destekleyerek siber güvenliğe ihtiyaç duyulan her alanda çalışmalar yürütmektedir. Cumhurbaşkanlığı Dijital Dönüşüm Ofisi, Bilgi Teknolojileri ve İletişim Kurumu, Ulaştırma ve Altyapı Bakanlığı, Sağlık Bakanlığı, Milli Eğitim Bakanlığı, Kişisel Verileri Koruma Kurumu, Üniversiteler gibi farklı paydaşlar ile çalışmalar yürütmektedir.

BGD, CyberMag Dergisi ile de toplumun tüm kesimlerine ulaşmaya, bu alandaki gelişmeleri sürekli olarak paylaşmaya ve değerli uzmanların görüşlerini aktarmaya çalışmaktadır. Ayrıca, bilgi güvenliği ve siber güvenlik alanında ulusal ve uluslararası düzeyde tarafsız, güvenilir ve etkin bir ulusal sivil toplum kuruluşu olan Bilgi Güvenliği Derneği, bünyesinde oluşturulan BGD Genç ile; bireysel, kurumsal, ulusal ve evrensel boyutlarda bilgi ve iletişim güvenliği alanında teknik, bilimsel, sosyal ve kültürel faaliyetler yürütmek, orta ve yüksek öğrenim gören genç üyelerimizin mesleki gelişimini artırmak, siber güvenlik alanında farkındalık oluşturmak, ülkemizin siber güvenlik uzman kaynağını oluşturmak için gençlerimizin bu alana ilgisini artırmak için faaliyet göstermektedir.

Bu yıl **"Sağlık Sektöründe Siber Güvenlik"** teması ile Cumhurbaşkanlığı Millet Kütüphanesi'nde 14. sünü düzenleyeceğimiz "**Uluslararası Bilgi Güvenliği ve Kriptoloji Konferansı**" kısaca **ISCTurkey Konferansı** olarak bilinen uluslararası etkinlik ile kurulduğu günden bu yana kamu kurumları, özel sektör, STK ve üniversitelerde görev yapan uzmanları bir araya getirmeyi başarmıştır.

ISCTurkey etkinlikleri, Gazi Üniversitesi, İstanbul Teknik Üniversitesi, ODTÜ ve TOBB ETÜ işbirliği ile düzenlenmekte; Ulaştırma ve Altyapı Bakanlığı, Cumhurbaşkanlığı Dijital Dönüşüm Ofisi, Bilgi Teknolojileri ve İletişim Kurumu, Kişisel Verileri Koruma Kurumu tarafından desteklenmektedir. Bu etkinlikler; ülkemizde açık kaynak olarak yapılan, IEEE tarafından desteklenen, katılımı ücretsiz alanındaki ilk ve tek etkinliktir. Ayrıca, düzenlendiği ilk yıldan beri ülkemizin siber güvenlik alanındaki bilimsel ve sektörel çalışmaların paylaşıldığı, üniversite-kamu-endüstri işbirliğinin geliştirildiği, kamunun bilgilendirildiği, paydaşların eğitildiği, tüm bilim insanları, araştırmacılar ve sektörel uygulayıcılar arasında bilgi alışverişinin sağlandığı, bu alandaki en önemli etkinliktir.

BGD, Siber Güvenlik ve Savunma Kitap Serisi ile de ülkemizde bu alandaki en kapsamlı çalışmayı yapmaktadır. Bugüne kadar 4 cildi yayımlanan kitap serisinde; "Problemler ve Çözümler", "Farkındalık ve Caydırıcılık", "Standartlar ve Uygulamalar" ve "Biyometrik ve Kriptografik Uygulamalar" başlıkları altında farklı konular ele alınmıştır. Bu kitap serisinin 5. cildi olan "**Blokzincir ve Kriptoloji**" kitabı da daha önceki serideki kitaplar gibi açık kaynak olarak yayımlanmaktadır.

Bu kitap serisinin 5. cildinin hazırlanmasında büyük emek veren, katkı sağlayan ve editörü olan; 2 dönem de BGD YK Başkanlığı yapan değerli hocamız Prof. Dr. Şeref SAĞIROĞLU'na ve bu kitaba katkı veren diğer editörümüz Doç. Dr. Sedat AKLEYLEK'e ve kamuoyu ile ücretsiz ve açık kaynak olarak paylaşılması konusunda destek veren saygıdeğer yazarlarımıza, maddi ve manevi destek sağlayan Cumhurbaşkanlığı Dijital Dönüşüm Ofisi Başkanlığı'na ve bugüne kadar ülkemiz bilgi güvenliği ve siber güvenliğinin gelişimine katkı sağlayan BGD Yöneticilerimize ve Üyelerimize bu vesile ile şükranlarımı sunarım.

Bu kitap serisinin, ülkemiz siber güvenlik ve savunma çalışmalarına katkılar sağlaması dileğiyle.

Taha YÜCEL

Bilgi Güvenliği Derneği
Yönetim Kurulu Başkanı

EDİTÖRLERDEN

Bilgi Güvenliği Derneği (BGD), kuruluşundan bugüne kadar ülkemizin **bilgi ve siber güvenliği ile savunmasının** gelişimine katkı sağlamakta, birikimini çevreye aktarmakta, bilgi güvenliği alanında açık kaynak yaklaşımını benimseyen ve bu kapsamda içerik üretilmesine ve geliştirilmesine destek vermekte, bunları yaymakta, paylaşmakta ve kamuoyunun kullanımına sunmaktadır. Düzenlediği ulusal ve uluslararası etkinliklere ait bildiri kitapları serisi, hazırladığı raporlar, taslak strateji dokümanları ve eylem planları vb. bunların başında gelmektedir. **Siber Güvenlik ve Savunma Kitapları Serisi** ise BGD'nin ülkemizin siber güvenliğine önemli bir katkısıdır.

Tehditlerin, saldırıların veya açıklıkların artması, boyut ve yön değiştirmesi, farklılaşması, siber tehdit ekosisteminin gittikçe güçlenmesi, kritik altyapıların daha çok hedef haline gelmesi, bilgi ve kaynak hırsızlıklarının çoğalması, yeraltı yapıların etkinleşmesi, siber tehditlerin artık savaşa dönüşmesi, siber suç ve suçlularının çoğalması, siber terörün yaygınlaşması vb. olumsuzlukların hızla artması, yapılacak mücadele, alınacak önlem ve karşı koyulacak yaklaşımlara duyulan ihtiyacı artırmıştır. Kapsamlı bir mücadele için; ulusal strateji ve eylem planlarına, araştırma merkezlerine, gelişmiş altyapı ve araçlara, lisans ve lisansüstü programlara, nitelikli insan kaynağına, yerli ve milli ürünlerin geliştirilmesine, siber güvenlik ve savunma ekosisteminin oluşturulmasına, ulusal siber olaylara müdahale ekiplerinin sayısının ve niteliğinin arttırılmasına, savunma sanayinin gelişmesine katkı sağlayacak yeni çalışma ve projelerin hayata geçirilmesine mevcut sistem, yapı ve organizasyonların kapsamının büyütülmesine, yeni yapıların kurulmasına ihtiyaç vardır. Duyulan bu ihtiyacı bir nebze de olsa karşılamak için bu kitap serisi hazırlanmıştır. Bu kitap serisinde, 100'e yakın farklı konu irdelenmektedir. Her bölümde, farklı bir konu siber güvenlik ve savunma kapsamında ele alınmakta, değerlendirilmekte ve alınması gereken önlemlere yer verilmektedir.

Bu kitap serisinde sunulan konu başlıkları, ülkemizde bu alanda çalışan akademisyenler, uzmanlar ve çalışanlar ile paylaşılmış ve bu kitap serisine katkı sağlamaları istenilmiştir. Zamanı uygun olan, katkı vermek isteyen uzman veya akademisyenler belirlenen bir konuda bölüm yazarı olmaları için davet

edilmişlerdir. Belirlenen süre içerisinde bölümlerini tamamlayan yazarlarımızın eserleri ise uygun olan ciltlerde basılmaktadır. Siber güvenlik ve savunmaya çok kapsamlı bir bakış sunmayı amaçlayan ve farklı başlıkları bir araya toplayan bu kapsamlı eserin, ülke siber güvenliğimiz ve savunmasına katkı sağlaması beklenmektedir.

Bu kitap serimizin **beşinci cildinde**; "***Blokzincir ve Kriptoloji***" konuları ele alınmış ve 12 farklı bölüm altında siber güvenlik ve savunma konuları sunulmuştur. Her bir bölüm; ülkemizde bu alana katkı sağlayan, bu alanda eğitim almış, tez hazırlamış, çalışmalar yapmış değerli akademisyen, kamu çalışanı ve üst düzey yöneticiler tarafından hazırlanmıştır. Editörler olarak bu kitap serisinin özenle hazırlanmasına dikkat edilmiştir. Ayrıca; kitapta konu bütünlüğü ve devamlılığının sağlanmasına özen gösterilmiş, yazarlara konu içeriği ve başlıklarla ilgili olarak bazı önerilerde bulunulmuş, düzeltmeler yapılması istenilmiş, bölümler intihal taramasından geçirilmiş ve sonuçta yapılan değişiklikler dikkate alınarak bu kitap hazırlanmıştır.

Bu kitabın; siber güvenlik ve savunma konusunda yapılacak olan çalışmalara ışık tutması, yeni çalışmaların yapılmasına katkılar sağlaması, bu alanda ihtiyaç duyulan Türkçe akademik kaynağın bir nebze de olsa karşılanması ve en önemlisi ise açık kaynak felsefesinin yaygınlaşması ile bu alanda ihtiyaç duyulan güncel kaynaklara erişimi kolaylaştırıcı **bir başvuru kitap serisi** olması beklenmektedir. Bu eser serisi **açık kaynak** olarak, Bilgi Güvenliği Derneği web sayfasında (www.bilgiguvenligi.org.tr) yayımlanmaktadır.

Bu kitapta yazarlarımız; alan uzmanlıklarına göre bölümleri hazırlamışlar, kişisel ve kurumsal bilgi birikimlerini hazırladıkları bölümlerde sunmuşlar, hazırladıkları bölümlerin açık kaynak olarak yayımlanmasını kabul etmişler ve bu kitabın basımı ve dağıtımı ile ilgili olarak herhangi bir telif hakkı talep etmemişlerdir. Yazarlarımıza, bu kitap serisinin editörleri olarak çok özel teşekkürlerimizi ve şükranlarımızı sunarız.

Kitabın titizlikle hazırlanmasında, kontrolünde ve basılmasında başta yazarlarımız olmak üzere emeği geçen tüm paydaşlarımıza, bu fikrimizin hayata geçirilmesine katkı sağlayan Bilgi Güvenliği Derneği YK üyelerimize ve özellikle de basılmasına maddi destek veren Cumhurbaşkanlığı Dijital Dönüşüm Ofisi Başkanlığımıza teşekkürlerimizi sunarız.

Prof. Dr. Şeref SAĞIROĞLU

Gazi Üniversitesi Yapay Zeka ve Büyük Veri Analitiği Güvenliği Uygulama ve Araştırma Merkezi Müdürü

- BGD Kurucu Üyesi, 2008-2012 Dönem BGD YK Başkanı
- IPv6 Forum Turkey Başkanı
- Bilgi Güvenliği Derneği Ulusal Bilim Kurulu Başkanı
- ISCTurkey Konferans Eş Başkanı
- Gazi Üniversitesi MF Bilgisayar Mühendisliği Bölümü Öğretim Üyesi
- FutureTech Genel Müdürü

Doç. Dr. Sedat AKLEYLEK

BGD Ulusal Bilim Kurulu Üyesi,
Ondokuz Mayıs Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü Öğretim Üyesi
S DataM Bilişim Teknolojileri ve Güvenliği Ltd. Şti. Kurucu Ortağı

- BGD Ulusal Bilim Kurulu Üyesi,
- ISCTurkey Akademik Program Başkanı
- Ondokuz Mayıs Üniversitesi MF Bilgisayar Mühendisliği Bölümü Öğretim Üyesi
- Turkish Journal of Electrical Engineering and Computer Sciences Editörü
- International Journal of Information Security Science (IJISS) Editörü

İÇİNDEKİLER

Bölüm 1

BLOKZİNCİRDE GÜVENLİ VE GÜVENİLİR UYGULAMA GELİŞTİRME TEMELLERİ

Enis Karaarslan - Melih Birim

Merkezi olmayan çözümlerin öneminin anlaşıldığı ve hayata geçirilmeye başlandığı bir çağdayız. Blokzincir teknolojisi ve akıllı sözleşme tabanlı merkezi olmayan uygulamalar birçok alanda iş yapma şeklimizi değiştirmeyi vadetmektedir. Merkezi olmayan uygulamalar birçok sistem için devrimsel değişiklikler yapma potansiyelini taşımaktadır ama bildiğimiz yazılım geliştirme süreçleri bu ortamlarda geliştirme yapmak için yeterli değildir. Blokzincirinde değiştirilemez kayıtlar kullanıldığı ve kodlar otonom çalıştığı için olası bir hatada değer kaybı yüksek olmaktadır. Bu kodların yazıldığı dil ve ortamlar henüz yeterince olgun değildir. Özellikle değer transferlerinde sorun yaşanmaması için güvenli ve güvenilir uygulamalar geliştirilmelidir. Bu tür kodların nasıl geliştirileceği ve test edileceği konusunda daha fazla örnek uygulama ve kılavuza ihtiyaç duyulmaktadır. Bu bölüm ile bu konularda temel altyapının verilmesi hedeflenmektedir. Bölüm blokzincir teknolojisinin ne olduğuna dair bir ön bilgi ile başlamaktadır. Ethereum, Quorum, Hyperledger, Corda, Avalanche ve Polygon gibi blokzincir platformlarına dair ön bilgi verilecektir. Blokzincir çözümlerinin nasıl geliştirileceği ele alınacaktır. DS4H blokzincir araştırma ağı hakkında bilgi verilecektir. Güvenli ve güvenilir akıllı sözleşme geliştirmeye dair uygulama örnekleri Ethereum Solidity ortamında verilecektir. Akıllı sözleşme geliştirilmesi, akıllı sözleşmelerin blokzincir sistemine yüklenmesi ve test edilmesi konusunda yapılmakta olan çalışmalardan ve geliştirilmekte olan araçlardan (ChainEx, GoHammer) söz edilecektir. Akıllı sözleşmelere olan belli başlı saldırılar için çözüm önerileri verilecektir. Akıllı sözleşmeler için güvenlik denetim listesi ve öneriler sunulacaktır. Blokzincir ağ başarım, akıllı sözleşme testleri ve güvenlik kontrolleri için kullanılabilecek yöntemler tanıtılacaktır. Blokzincir ortamında yazılım geliştirme süreçlerinde yaşanan sıkıntılara ve fırsatlara değinilecektir. Merkezi olmayan sistemlerin güvenilir ve sürdürülebilir olması için yapılabilecek çalışmalara dair öngörüler paylaşılacaktır.

1.1. GİRİŞ

İş süreçlerimizi güven (trust) içinde yapmak için güvenilir aracı kişiye veya kuruma (trusted third party) ihtiyaç duyarız. Bu süreçlere aracı olmaları ve işlemlerin kaydını güvenilir bir şekilde saklamaları için banka ve noter gibi birçok aracı (intermediary) kurumdan yararlanırız. Bu aracı kurumlar genellikle süreci yavaşlatır, işlemler ve komisyonlar çoğunlukla maliyetlidir ve işlemler sadece çalışma saatlerinde yapılabilir. Güvenilir işlemler için aracılar kullanılsa da, yine de işlemler çeşitli şekillerde manipüle edilebilir. Aracıları ortadan kaldıran otonom ve güvenilir (trusted) sistemler kurmak için bilgi teknolojilerinin kullanılması üzerine günümüzde denemeler yapılmaktadır. Merkezi olmayan P2P (peer to peer) ağlar üzerinden çalışan dosya sistemi BitTorrent bu konudaki ilk başarılı denemelerden biridir. Bu tür sistemler öncelikle dağıtıktır. Merkezi bir otoritenin/sunucunun kontrol etmediği ve aracıların olmadığı çözümlere ise merkezi olmayan (decentralized) çözümler denir.

Satoshi 2009'daki yayında [1], aracılar olmadan taraflar arasında güvenilir bir şekilde değer transferinin gerçekleşmesini sağlayacak blokzincir sistemini ve bu süreçte kullanılabilecek Bitcoin kripto parasını (cryptocurrency) tanıtmıştır. Blokzincir sistemi, 2009'dan beri aktif olan Bitcoin ile çalışabilirliğini kanıtlamıştır. Bu teknoloji, devrimsel değişiklikler vaat etmektedir. Blokzinciri üzerinde akıllı sözleşme (smart contract) adı verilen kodlarla merkezi olmayan çözümler geliştirilmesi mümkün olmaktadır. Akıllı sözleşmeler ile süreçler otonom hale gelmekte veya daha az insan katılımı ile gerçekleştirilebilmektedir. Bu da sözleşme işleme maliyetini önemli ölçüde azaltmaktadır. QYResearch'in raporuna [2] göre, küresel akıllı sözleşmeler pazar büyüklüğünün; 2019'da 106,7 Milyon ABD Dolarından 2026 yılına kadar 345,4 Milyon ABD Dolarına ulaşacağı tahmin edilmektedir. Tedarik zinciri, bankacılık, devlet, sigorta ve emlak sektörlerinde yaygın kullanım beklenmektedir. Blokzincir teknolojisinin artan popülaritesi de bu pazara olan talebi arttırmaktadır.

Akıllı sözleşmeler otonom çalıştıklarından ve blokzincirinde değiştirilmez kayıtlar kullanıldığından; olası bir hatada veya güvenlik zafiyetinde

değer kaybı yüksek olacaktır. Akıllı sözleşmelerdeki hatalar sonucunda DAO (Decentralized organization - merkezi olmayan organizasyon) saldırıları, Ether Tahtının Kralı (King of the Ether Throne) ve çok ortaklı oyunlar (Multi-player Games) gibi çeşitli saldırılar yaşanmaktadır [3]. Bu kapsamda en büyük saldırılardan olan ve 2016 senesinde yapılan DAO saldırısında; o zaman için 50 milyon dolar değerinde olan 6 milyon ETH (Ethereum) çalınmıştır [4].

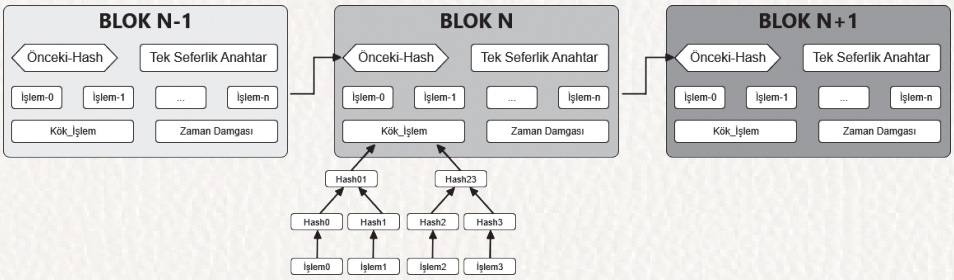
Bu saldırılar; blokzincirinin kullandığı kriptografik modelden ziyade, uygulama geliştirme süreçlerindeki sıkıntılardan kaynaklanmaktadır. Akıllı sözleşmelerin geliştirildiği dil ve ortamlar gelişmektedir ve henüz yeterince olgun değildir. Bu ortamlarda çözüm geliştirilirken nelere dikkat edilmesi gerektiği bu bölümde ele alınacaktır. Güvenli ve güvenilir yazılım geliştirmek için; blokzincir platformları, konsensüs protokolleri ve akıllı sözleşmelerin test edilmesi gereklidir. Yazılımcılar, blokzincir sistemleri üzerinde uygulama geliştirebilecekleri ve test süreçlerinde kolay kullanabilecekleri araçlara ihtiyaç duymaktadır. Bu kapsamda geliştirilmekte olan platform ve araçlar [5, 6] da bu bölümde sunulacaktır.

Bölüm 1.2'de blokzincir teknolojisinin ne olduğuna dair bir temel bilgi verilecektir. Bölüm 1.3'te merkezi olmayan çözümler anlatılacaktır. Blokzincir platformlarına, merkezi olmayan çözümlere ve yönetilebilir ağ servislerine dair temel bilgiler verilecektir. Ethereum Solidity ortamında akıllı sözleşmelerde güvenli kod geliştirme Bölüm 1.4'te verilecektir. Bölüm 1.5'te akıllı sözleşmelere olan belli başlı saldırılar ele alınacaktır. Blokzinciri ağ başarım (performance) testleri ve akıllı sözleşme testleri Bölüm 1.6'da ele alınacaktır. Bölüm 1.7'de güvenlik kontrol listesi verilecektir. Blokzinciri için yazılım geliştirmede sıkıntılar ve fırsatlar Bölüm 1.8'de tartışılacaktır. Bölüm 1.9'da sonuç ve değerlendirmeler verilecektir.

1.2. BLOKZİNCİR TEKNOLOJİSİ

Blokzinciri, merkezi olmayan bir şekilde işlemleri doğrulayan ve işlemlere dair kayıtların kaydını tutmak için kullanılan bir teknolojidir ve ilk kez Satoshi'nin Bitcoin'in nasıl çalıştığını anlattığı yayında [1] tanımlanmıştır. Blokzinciri,

dağıtık veya merkezi olmayan defter teknolojisi (DLT) olarak da adlandırılır. Blokzincirindeki blok yapısı Şekil 1.1'de verilmiştir.



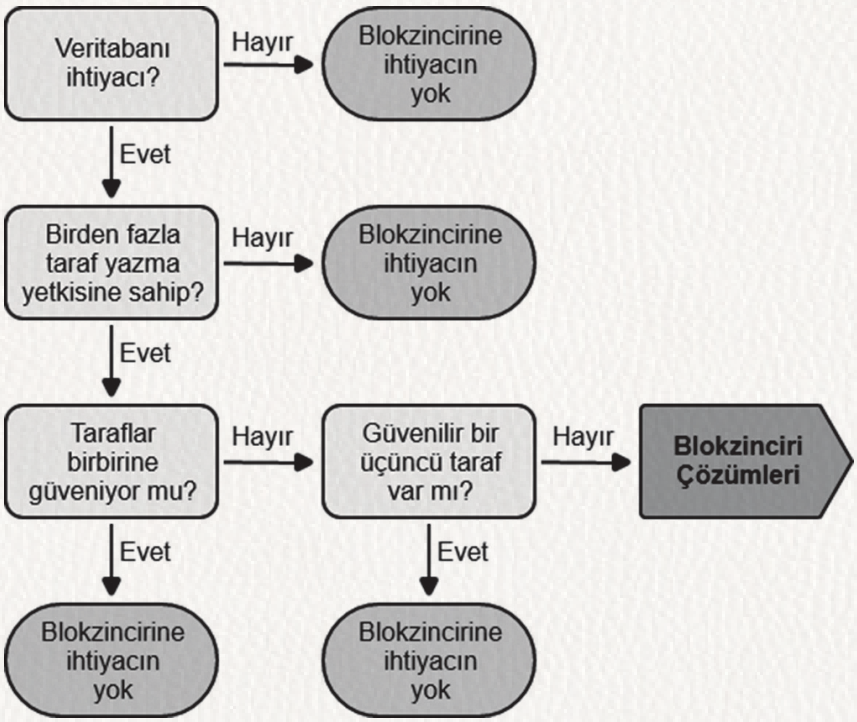
Şekil 1.1. Blok Yapısı [7]

Blokzincirinde kayıtlar, bir sistemin otonom olarak çalışmasına izin veren bir işlem (transaction) veya bir program kodu (akıllı sözleşme) hakkında bilgi içerir. İşlem bilgisine dair kaydın değişmediğini ve bütünlüğünü (integrity) göstermek için Keccak-256, SHA-3 gibi kriptografik hash fonksiyonları kullanılır [8]. İşlemlerin hash değerleri bir Merkle ağacı oluşturacak şekilde hesaplanır ve "Kök İşlem" (Tx_Root) değeri oluşturulur. Her blokta tek seferlik anahtar (nonce) değeri ve zaman damgası tutulur. Kayıtlar blok diye adlandırılan veri yapılarında toplanır. Bu bloklar bir önceki bloğun hash değerleriyle birbirine bağlanır ve bu yapıya blokzinciri denir. Bu kayıtların tümüne kayıt defteri (ledger) da denir [1]. Kayıt defteri, düğüm (node) adı verilen cihazlarda tutulur. Bu cihazlar farklı bilgisayar ağlarında çalışır ve P2P protokolleri ile birbirleriyle iletişim kurarlar. Bunlar aynı anda sunucu ve istemci görevi gerçekleştirir ve merkezi olmayan bir sistem oluştururlar. Düğümler farklı donanıma ve fonksiyonlara sahip olabilirler.

Sistemin tutarlı çalışabilmesi için bu cihazların fikir birliği/konsensüs (consensus) içinde çalışabilmesi gerekir. Bu cihazlar; sistemdeki işlemlerin geçerliliği, bir sonraki bloğu hangi cihazın yazacağı gibi ortak kararlar için konsensüs protokollerini kullanır [9]. Kullanılan konsensüs protokolüne göre blokzincir teknolojisinin çalışması için gereken minimum düğüm sayısı değişkenlik gösterir. Örneğin Raft protokolüne göre gereken minimum düğüm sayısı üç iken, IBFT protokolüne göre bu sayı dörttür.

Merkezi olmayan çözümler sürekli gelişmektedir. Blokzincirinde en önemli milatlardan birisi, Ethereum platformu ile akıllı sözleşme/kontrat (smart contract) kavramının, yani bir adreste tanımlı otonom kodların kullanılabilir olmasıdır [10]. Ethereum'dan sonra Corda, Hyperledger gibi farklı blokzincir platformları da benzer otonom kod geliştirme ortamları sunmuşlardır. Ethereum günümüzde akıllı sözleşmeleri kodlamak ve işlemek için kullanılan en gelişmiş teknolojidir. Ethereum'un 2020-2026 tahmin süresi boyunca, akıllı sözlesme pazarının en büyük payına sahip olmaya devam etmesi beklenmektedir [2].

Blokzinciri, tüm yaşanan problemler için uygun bir çözüm değildir. Bu tür bir çözüme ne zaman ihtiyaç duyulabileceği, Wüst ve Gervais'in çalışmasında [11] anlatılmıştır ve Şekil 1.2'de özetlenmiştir.



Şekil 1.2. Blokzincirine İhtiyaç Duyulma Durumları [11]

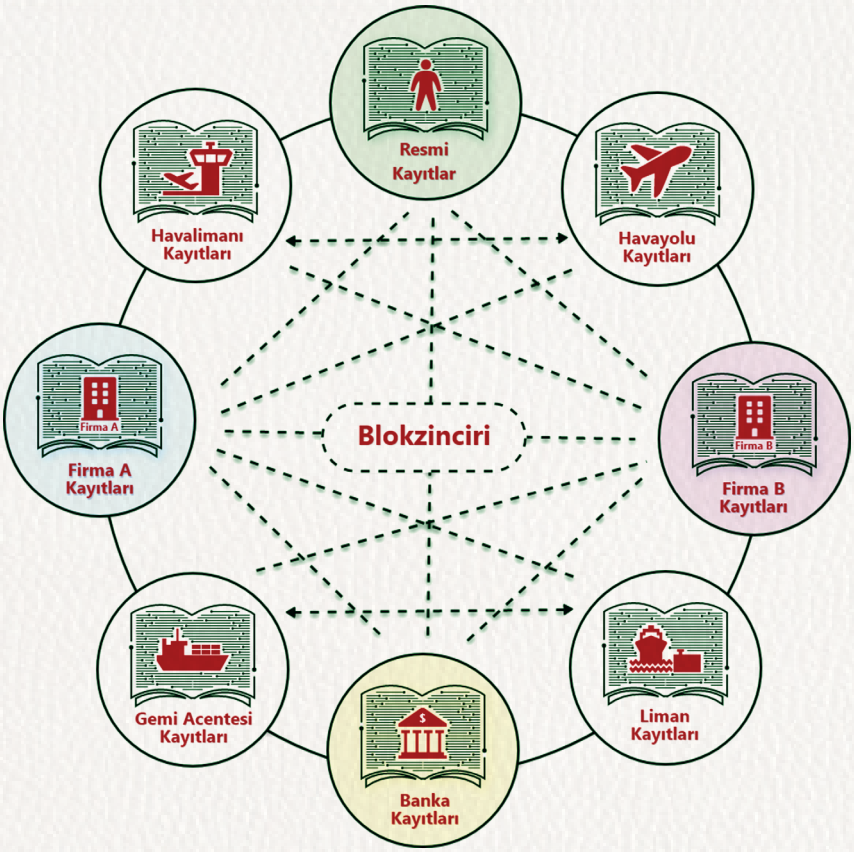
Bir alanda blokzinciri kullanımının anlamlı olabilmesi için öncelikle kayıtların saklanacağı bir veritabanı çözümüne ihtiyaç duyuluyor olmalıdır. Bu veritabanına farklı tarafların (kurum veya kişilerin) yazma ihtiyacının olması gerekir. Bu süreci gerçekleştirecek taraflar arasında güvenin düşük olması ve bu güveni sağlayacak güvenilir herhangi bir aracı kurumun (trusted third party) var olmaması gerekir. Tarafların bu şartlar altında da güvenilir işlem yapmak istemesi ve bu işlemlerin sonradan denetim için kayıt edilmesi ihtiyacının var olması durumunda blokzinciri kullanımı anlamlı olmaktadır [12]. Blokzincirin kullanımının potansiyeli olduğu başlıca alanlar olarak; finans ve değer aktarımı, tedarik zinciri (supply chain), dolandırıcılık tespiti (fraud detection), sağlık veri değişimi (health data exchange) [7], müşterini tanı (know your customer), merkezi olmayan kimlik (decentralized identity), akıllı yönetim (smart governance) [13], ulusal ve siber güvenlikten söz edilebilir.

Blokzincir süreçlerinin uygulanabileceği alanlardan birisi tedarik zinciri uygulamalarıdır. Kurum tedarik zincirindeki tüm süreçleri izlemek ve kullanıcılarına bu süreçleri şeffaf hale getirmek isteyebilir. Örneğin Muğla'daki ES firmasından Osaka'daki Burokkuchēn firmasına bir lavanta ihracatında, Şekil 1.3'de gösterildiği gibi taşıma şirketleri, limanlar, resmi kurumlar ve bankalar gibi çok farklı kurumlar bu sürece dahil olacaktır. Tüm bu kurumların aralarında veri paylaşması ve işlem oluşturması gerekecektir. Blokzinciri, birbirine güvenmek zorunda olan birçok tarafın olduğu böyle bir veri paylaşımında anlamlı bir çözümdür[1]. TradeLens [14] blokzincir çözümü bu alana örnek verilebilir. 300'den fazla organizasyon, 10 transatlantik nakliye şirketi, 600'den fazla liman ve terminal bu sistemi kullanmaktadır[2]. Blokzinciri tabanlı sistemler akıllı şehir (smart city) uygulamalarında kullanılabilir. Merkezi olmayan kimlik yapıları aracılığı ile vatandaşların bilgilerine ulaşılabilir. Vatandaşlar kendi kişisel verilerinin mahremiyetini sıfır bilgi kanıt (zero knowledge proof) [15] protokolleri

1 Mohan C. (2019). State of Permissionless and Permissioned Blockchains: Myths and Reality, BlueTalks @ Rio BNDES

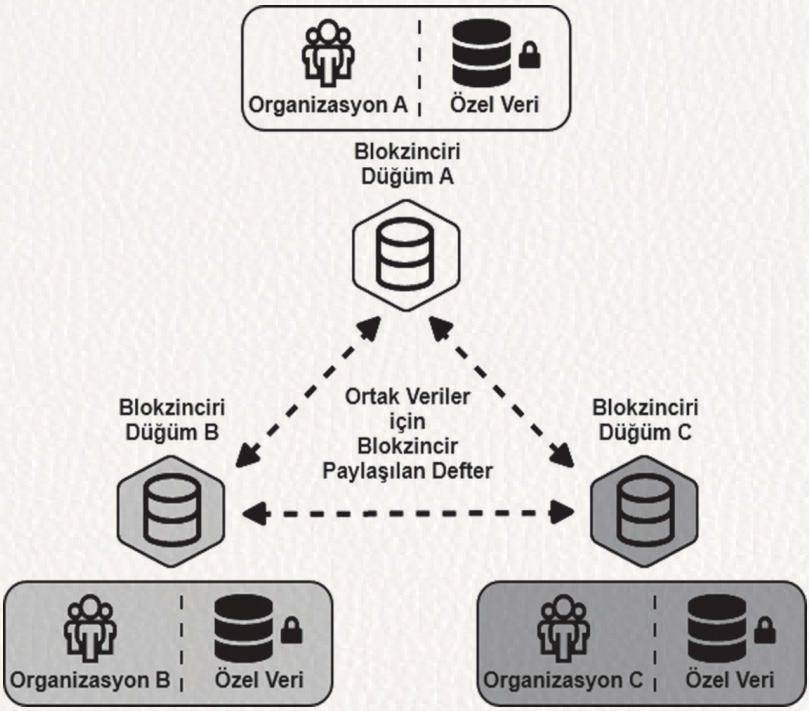
2 Hapag-lloyd and Ocean Network Express Complete Tradelens Integration, https://www.tradelens.com/post/hapag-lloyd-and-ocean-network-express-complete-tradelens-integration

ile koruyabilir. Böyle bir sistem; pasaporttan sağlık ve tapu uygulamalarına kadar birçok alanda kullanılabilir [13].



Şekil 1.3. Çok Farklı Kurum Veri Erişimi Senaryosu [7]

Blokzinciri tabanlı sistemlerde verinin kaydedilmesi ve sonradan erişilmesinde başarım ölçütleri dikkate alınmalıdır. Sistemin sürdürülebilir olması için öncelikle verinin ölçeklendirilmesi ve veriye hızlı bir şekilde ulaşılabilmesi gerekir. Bu durumda karma/hibrit çözümlerin kullanımı da söz konusudur. Böyle bir kullanım modeli Şekil 1.4'te gösterilmiştir [7].



Şekil 1.4. Paylaşılan Veri Senaryosu [7]

Blokzincir sistemi bu kullanım modelinde bir alternatif çözüm olmaktan daha çok, bütünleyici olarak yer alacaktır. Bu yapıda, kurumlar kendi özel verilerini kendi sistemlerinde tutmaya devam edeceklerdir. Ortak paylaşılan veriler ise blokzincirinde tutulacaktır [7]. Blokzincirinin veri bilimi açısından farklı uygulamaları Karaarslan ve Konacaklı'nın bir başka çalışmasında [12] ayrıntılı olarak ele alınmıştır.

1.3. MERKEZİ OLMAYAN ÇÖZÜMLER

Merkezi olmayan üç farklı çözüm türünden söz edilebilir. Bunlar; blokzincir platformları, merkezi olmayan uygulamalar ve yönetilebilir ağ servisleridir. Alt başlıklarda öncelikle blokzincir platformlarından, sonrasında da merkezi olmayan uygulamalardan söz edilecektir. Burada akıllı sözleşmelerin nasıl

çalıştığı anlatılacak ve Ethereum ortamında kullanılan belli başlı uygulama geliştirme araçları kısaca tanıtılacaktır. En son olarak da uygulama geliştirme ve test sürecini kolaylaştırmak için yönetilebilir ağ servisleri anlatılacaktır.

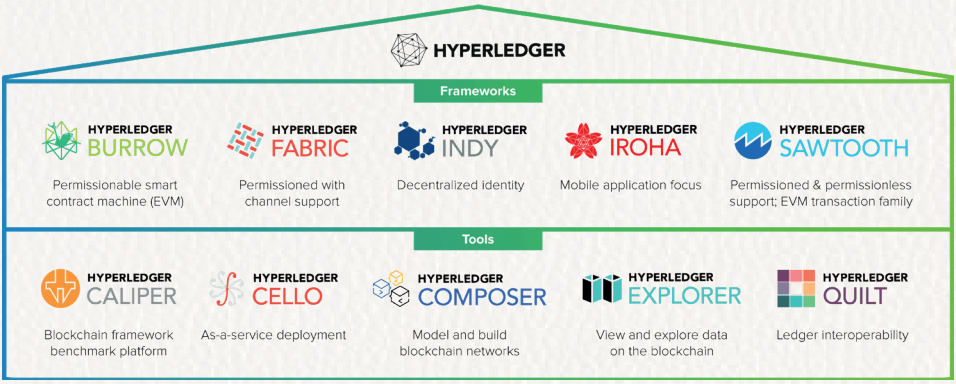
1.3.1. Blokzincir Platformları

Blokzincir platformları (blockchain framework); blokzincir sistemine dair temel altyapıyı sunarlar. Aynı zamanda blokzincir tabanlı çözümlerin geliştirilebileceği ortamlar da sağlarlar. Bu platformlar blokzincir düğümleri üzerinde bir servis olarak çalıştırılır. Ethereum, Quorum, Hyperledger ve Corda günümüzdeki belli başlı blokzincir platformlarıdır.

Merkezi olmayan uygulamalar; Ethereum gibi genel (public) ağlar üzerinde çalıştırıldığında işlem ücreti ödenmesini gerektirecektir. Kurumsal uygulamalar için ayrıca platform kurulabilir. Bu platform, bir bulut servisi olarak alınabilir veya kurum altyapısına kurulur. Platformların lisansı izin veriyorsa; platform klonlanıp yeni bir sistem oluşturulur. Kurum isterse kendi platformunu sıfırdan kendisi geliştirir.

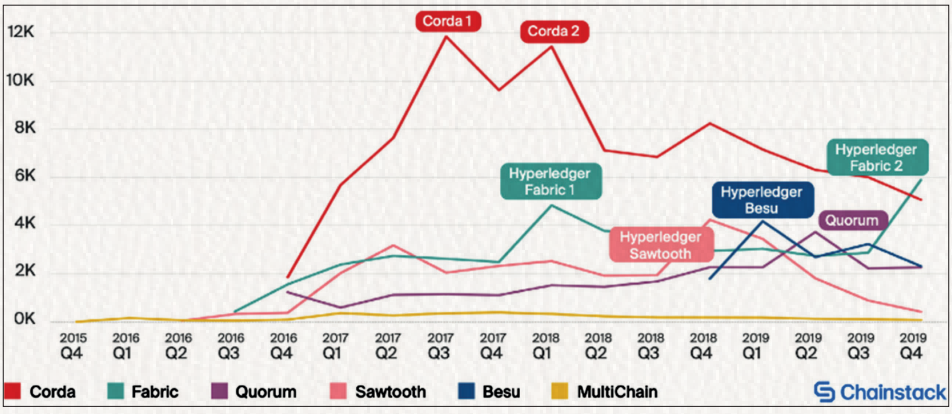
Blokzincir platformları farklı konularda uzmanlaşmaktadır. Örneğin, Corda ve Quorum finansal çözümler için geliştirilmiştir. Linux Foundation altında farklı gruplar tarafından geliştirilen Hyperledger, farklı platformları ve farklı araçları bünyesinde bulunduran bir çatıdır. Başlıca ürünler Şekil 1.5'te gösterilmiştir. Örneğin, Hyperledger Indy merkezi olmayan dijital kimlik uygulamalarında, Hyperledger Caliper başarım (performance) ölçümünde kullanılmaktadır. Bunun yanı sıra farklı platformlardan söz etmek mümkündür. Örneğin, Prof. Dr. Emin Gün Sirer'in kurduğu Avalanche Platformu[3] da gelişmekte olan [16] ve geliştiricilerine çeşitli fırsatlar sunan sayılı ortamlardan biridir. Polygon[4] ise Ethereum uyumlu blokzincir ağlarını oluşturmak ve birbirine bağlamak için bir protokol ve platformdur. Bu ortamda çoklu zincirli (multi-chain) bir ekosistem için ölçeklenebilir (scalable) çözümler geliştirilebilir.

3 Avalanche platformu, https://www.avalabs.org/
4 Polygon protokol ve platformu, https://polygon.technology/



Şekil 1.5. Belli başlı Hyperledger Çözümleri[5]

Blokzincir platformu seçerken geliştirici topluluklarının etkinliğine, topluluğun geliştirici sorunlarına cevap verme hızlarına ve projenin sürdürülebilirliğine dikkat edilmesi gereklidir. Platformlar üzerinde yapılan teknolojik geliştirmeler ve geliştirme topluluklarının etkinliği açısından Ethereum ve Hyperledger platformları sürdürülebilir projeler olarak gözükmektedir. Blokzincir platformlarının popülerliği zamana göre değişebilmektedir. Ethereum bu alanda en fazla kod yazılan ortam olarak liderliğini korumakla birlikte; Hyperledger, Quorum ve Corda'nın 2017-2019 seneleri arasında Github faaliyetlerinin değişimi Şekil 1.6'da gösterilmiştir.



Şekil 1.6. 2017-2019 Arasında Hyperledger - Quorum - Corda Platform Github Faaliyeti [17]

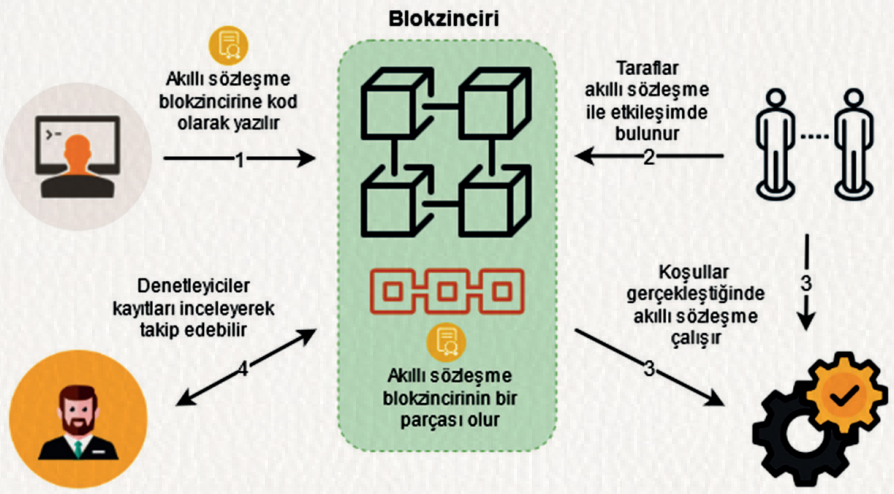
5 Hyperledger Solutions, https://events19.linuxfoundation.org/events/hyperledger-global-forum-2018/

Sekil 1.6'daki zaman diliminde; Corda'nın bir dönem ne kadar popüler olduğu ve sonra popülerliğini kaybettiği gözükmektedir. Bu zaman aralığında Hyperdleger Fabric'in popülerliği artmıştır. Zmudzinski'nin raporuna [17] göre; bu değişim Hyperledger Fabric'in 2019 yılının Kasım ayında kod yönetim aracı olarak Github'a geçmesinden sonra yaşanmıştır.

1.3.2. Merkezi Olmayan Uygulamalar ve Akıllı Sözleşmeler

Merkezi olmayan uygulamalar (decentralized applications - Dapp); blokzinciri üzerinden merkezi olmayan servisler verilmesini sağlayan uygulamalardır. Son kullanıcının bir mobil uygulama veya web tarayıcısı üzerinden blokzinciri ile iletişimini sağlayan ortamlardır. Bu uygulamalar, çoğunlukla bir API (Application Programming Interface - Uygulama Programlama Arayüzü) üzerinden blokzinciri ile iletişim kurarlar ve blokzincirindeki akıllı sözleşme olarak bilinen otonom kodu çalıştırırlar. Merkezi olmayan uygulamalar, üzerinde çalıştıkları blokzincir platformuna göre farklılıklar içermektedir. Ethereum ve türevlerinde otonom kodlara akıllı sözleşme/kontrat (smart contact) denir. Hyperledger dünyasında bu tür kodlara zincir kodu (chain code) denmektedir. Literatürde akıllı sözleşme kavramı daha sıklıkla kullanılmaktadır ve bu bölümde de bu şekilde kullanılacaktır.

Akıllı sözleşme yeni bir kavram değildir. Bu kavramı ilk olarak Nick Szabo 1997'de yayımladığı çalışmasında [18] ortaya atmıştır. Akıllı sözleşmeler, genel ağ (public network) üzerinden iletişim kuran taraflar arasındaki ilişkileri resmiyete döken ve güvence altına alan (secure) programlardır. Akıllı sözleşmelerin uygulanabilir olması için determinizm gereklidir. Bu da blokzincir ağındaki tüm düğümlerin bu kodları çalıştırdığında ortaya çıkan son durumun (resulting state) ve işlemlerin (transaction) aynı olmasını gerektirir. Ethereum [10] ve benzeri merkezi olmayan platformlar rassallık (randomness) gibi deterministik olmayan durumlara (non-determinism) izin vermeyerek bunu sağlarlar [19]. Bir akıllı sözleşmenin (kontrat) çalışma aşamaları Şekil 1.7'de gösterilmiştir.

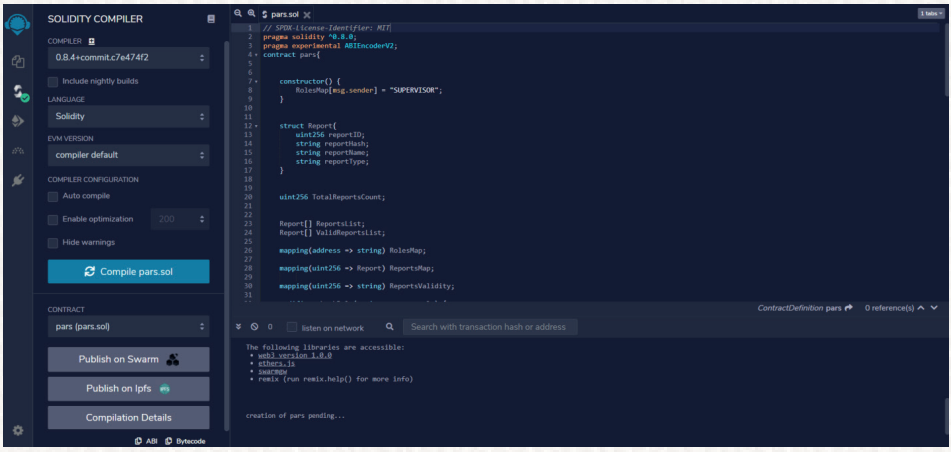


Şekil 1.7. Akıllı Sözleşmenin Çalışma Şekli

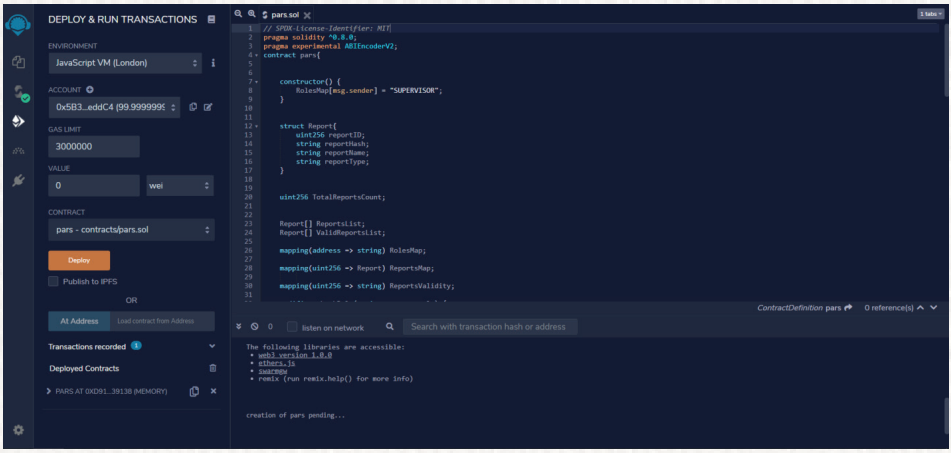
Geliştirici akıllı sözleşmeyi kodladıktan sonra blokzincirine kayıt olarak ekler. Bu kod Ethereum örneğinde genel (public) bir zincirde tutulabildiği gibi, farklı platformlarda özel (private) zincirlerde sadece belirli kurumlar arasında görünür bir şekilde de tutulabilir. Taraflar blokzincirindeki hesapları (account) üzerinden akıllı sözleşme ile etkileşimde bulunur. Akıllı sözleşmede belirlenmiş koşulların gerçekleşmesi durumunda kod kendini çalışır. İşlemin sonuçlarına dair kayıtlar blokzincirine yazılır. Böyle bir sistemde denetim süreçleri işlem kayıtlarının sonradan incelenmesi ile mümkün olur. Denetleyiciler (regulators), uygulamanın çalıştığı alana özel denetleme mekanizmalarıdır. Denetleyiciler blokzincir kayıtlarını inceleyerek kontratların çalışma durumunu takip edebilirler.

Ethereum'da bu kodlar (kontratlar) Solidity dilinde yazılır. Ethereum, şu ana kadar en fazla otonom kod geliştirilen ortamdır. Ethereum ortamında geliştirme sürecinde kullanılan belli başlı araçlar olarak Geth, Clef, Remix, Metamask, Truffle, Ganache ve web3.js örnek verilebilir. Geth, Ethereum'un Golang dilinde geliştirilmiş bir istemcisidir. Parity ve Besu gibi farklı özelliklere sahip Ethereum istemcileri de bulunmaktadır. Clef, Ethereum'da hesap yönetimi için kullanılan yeni bir sistemdir. Tek başına bir işlem olarak çalışır ve tüm hesap yönetimlerini Ethereum Geth istemcisi yerine kendisi sağlar.

Remix IDE (integrated development environment - tümleşik geliştirme ortamı), Ethereum akıllı sözleşmeleri geliştirip blokzincirine yüklemeye (deploy) imkan tanıyan, açık kaynaklı bir web ve masaüstü uygulamasıdır. Remix geliştirme ortamı Şekil 1.8'de gösterilmiştir. Bu arayüz üzerinden derleme (compile) ve blokzincirine yükleme süreçleri gerçekleştirilebilmektedir. Zengin bir eklenti setine sahiptir. Remix, tüm sözleşme geliştirme sürecinde kullanılabileceği gibi, Ethereum'u öğrenmek ve öğretmek için de kullanılabilir.



a) Remix Geliştirme Ortamı - Derleme



b) Remix Geliştirme Ortamı - Blokzincirine Yükleme

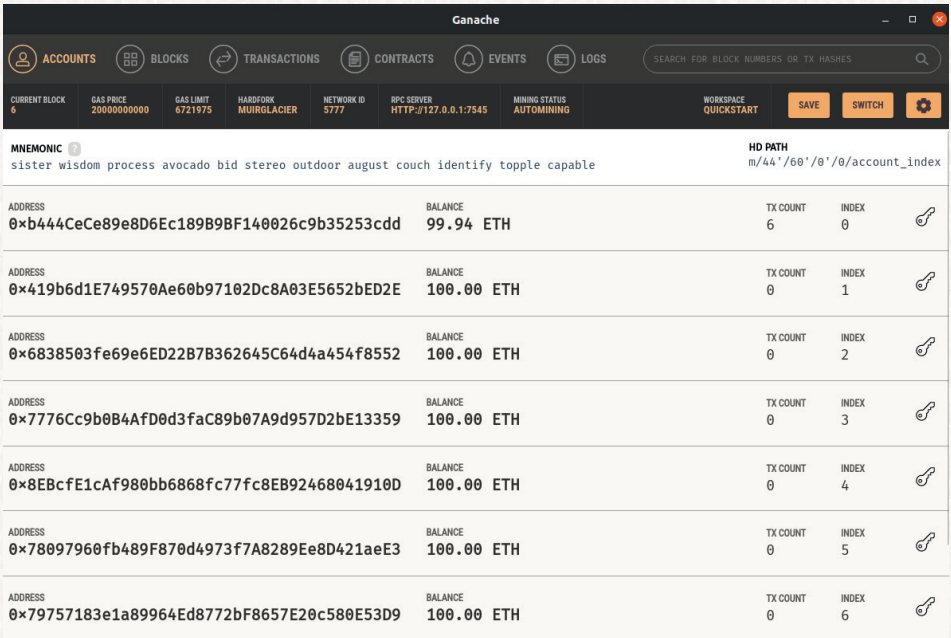
Şekil 1.8. Remix Geliştirme Ortamı Ekranları

Truffle Suite; Truffle, Ganache, Drizzle gibi bir dizi programlama aracından oluşan bir geliştirme paketidir. Ortam arayüzü Şekil 1.9'da gösterilmiştir. Truffle popüler bir geliştirme ve test ortamı, aynı zamanda da bir varlık hattıdır (asset pipeline). Bu programlar geliştirme sürecini kolaylaştırır. Akıllı sözleşme yaşam döngüsü yönetimi, otomatik akıllı sözleşme testi, betiklerle (script) dağıtım ve taşımalar (mitigation), basit ağ yönetimi ve interaktif konsol gibi akıllı sözleşme geliştirme sürecini kolaylaştıran özelliklere sahiptir.

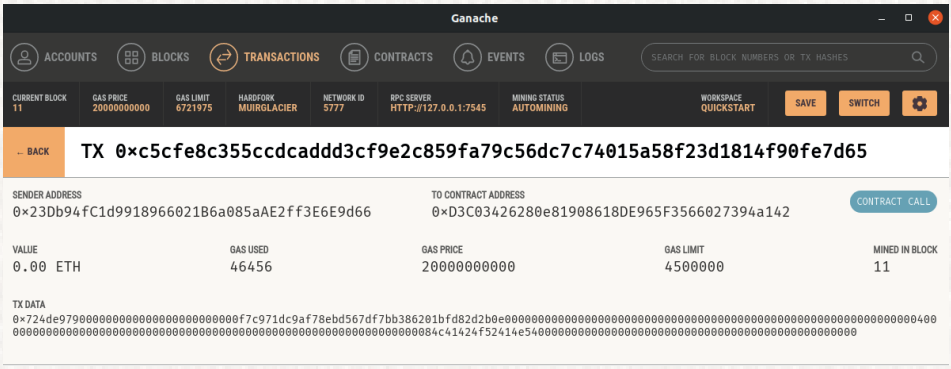
```
truffle(development)> accounts
[
  '0x23Db94fC1d9918966021B6a085aAE2ff3E6E9d66',
  '0x2Ba0463674C0D7DcE9B9A5B7e6794d6E3aA2e969',
  '0xB6F81934A6f6529D4f9dBC8E46b9aEB3c9d9916b',
  '0xF7c971DC9Af78ebD567df7Bb386201BfD82d2b0e',
  '0x43dE2AbF554Aa7Fd8833cA4E97EE611363a6bc50',
  '0x43AA5621F2eEb6aB66E3b5438A314BCf877AB9Bf',
  '0xdE5afA02efd9E8F7C7aAB122CA953B71153C1499',
  '0x0D8d2eeC60230D248e2f022bFC3d325C5D70be28',
  '0x40c731A5b1fe8A7376ad1C2181e1DEb482aE3f6A',
  '0xCdAb910144785D3b5367d16D78bC4CC328cE9E2f'
]
truffle(development)> contract.setUserRole(accounts[3], "LABORANT")
{
  tx: '0xc5cfe8c355ccdcaddd3cf9e2c859fa79c56dc7c74015a58f23d1814f90fe7d65',
  receipt: {
    transactionHash: '0xc5cfe8c355ccdcaddd3cf9e2c859fa79c56dc7c74015a58f23d1814f90fe7d65',
    transactionIndex: 0,
    blockHash: '0x5d3223b06a712356b7f9ffbdf0143d7e767fcc247a3b687741f9c2a0f5a00eb1',
    blockNumber: 11,
    from: '0x23db94fc1d9918966021b6a085aae2ff3e6e9d66',
    to: '0xd3c03426280e81908618de965f3566027394a142',
    gasUsed: 46456,
    cumulativeGasUsed: 46456,
    contractAddress: null,
    logs: [],
    status: true,
    logsBloom: '0x0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
    rawLogs: []
  },
  logs: []
}
truffle(development)> contract.getUserRole(accounts[3])
'LABORANT'
truffle(development)>
```

Şekil 1.9. Truffle Arayüzü

Truffle ile birlikte gelen Ganache ile yerel makineye bir Ethereum blokzinciri ağ simülatörü kurulur ve Solidity sözleşmeleri bu ortam üzerinde test edilir. Blokzinciri sistemi üzerindeki işlemlerin (transaction) ayrıntılarının takip edilebildiği Genanche arayüzü Şekil 1.10'da gösterilmiştir.



a) Ganache Arayüzünde İşlemlerin Takibi

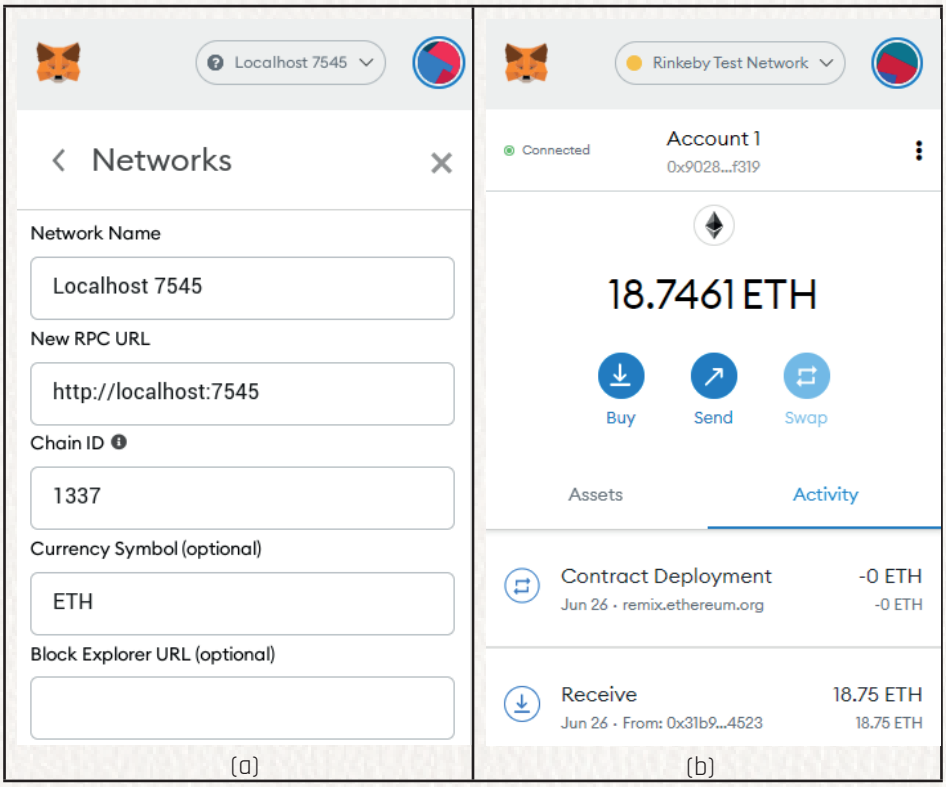


b) Ganache ile Bir İşlemin Detaylı Gösterimi

Şekil 1.10. Genanche Arayüzü Ekranları

Metamask, blokzincir uygulamaları için bir kripto cüzdanıdır. Blokzinciri tabanlı uygulamalara güvenli bir bağlantı sağlamak için bir ağ geçidi olarak da çalışır. Uygulamalar geliştirilirken Truffle ile birlikte kullanılır. Arayüzü Şekil 1.11'de gösterilmiştir. Web arayüzü geliştirmek için kullanılabilecek web3.js,

bir HTTP veya IPC bağlantısı kullanarak yerel veya uzak bir Ethereum ağı ile etkileşim kurulmasına izin veren bir kitaplık koleksiyonudur.

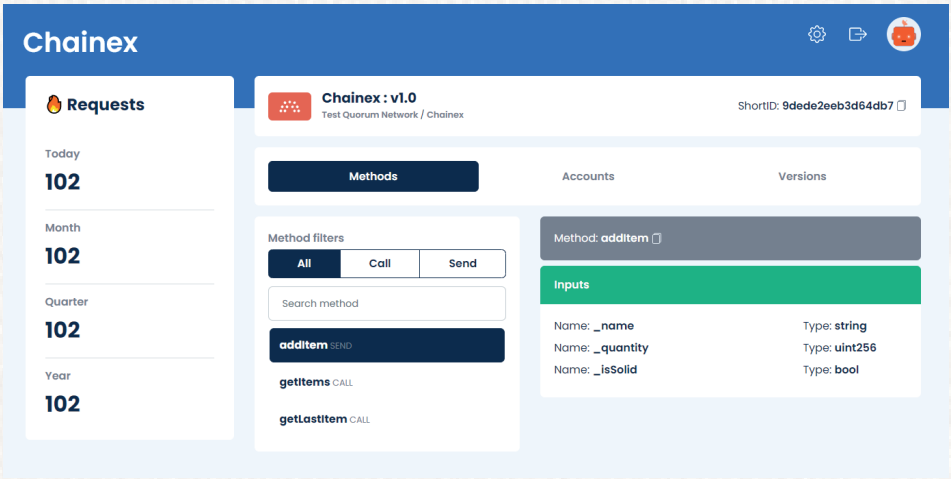


Şekil 1.11 (a) Metamask Network Ayarları
(b) Metamask ile Rinkeby Test Ağına Bağlı Bir Hesabın Takibi

1.3.3. Yönetilebilir Ağ Servisleri

Geliştiricilerin canlı ortam testlerini gerçekleştirilmesi kolay süreçler değildir. Blokzincir teknolojileri her geçen gün gelişmektedir ve tüm gelişmelere hakim olmak ciddi zaman ve efor gerektirmektedir. Yönetilebilir ağ servisleri (managed network services); test ve ölçeklendirme süreçlerini kolaylaştırıp hızlandıran ağ servisleridir. Geliştiricilerin blokzincirinde kod çalıştırdıklarında sonuçları takip etmesi de kolay değildir. Bu tür servisler; geliştiricilerin özellikle state

(durum) takibi yapmalarını kolaylaştıracaktır. Kod çalıştığında sistemde oluşturacağı değişikliğin takibi sağlanacaktır. Bu servisler; geliştirilen kodun değişik sürümlerinin (version) takibini de sağlayacaktır. Farklı konsensüs protokolleri seçiminde sistem performansın değişiminin gözlenmesi için de kullanılabilir. Örnek olarak; Simba (https://simbachain.com), Chainstack (https://chainstack.com) ve ChainEx https://github.com/chain-ex/service verilebilir. Yönetilebilir ağ servisleri; geliştiricilerin bu süreçlerde sonuçları takip edebilecekleri, kullanımı kolay arayüzler de sağlamalıdır. Örneğin, ChainEx blokzincir uygulama ortamı (workbench), geliştiricilerin akıllı sözleşmeleri blokzincir ağlarına yüklemeleri ve bu sözleşmelerle kolayca etkileşim kurmaları için geliştirilmiştir. ChainEx web arayüzü Şekil 1.12'de gösterilmiştir.



Şekil 1.12. ChainEx Web Arayüzü [5]

Geliştiriciler ChainEx web arayüzü üzerinden blokzincir ağları ile etkileşimlerini gerçekleştirebilmektedir. Bu uygulama geliştirme ortamı, merkezi olmayan uygulama programlamaya yeni başlayacak geliştiriciler için eğitim amaçlı da kullanılabilir. Muğla Sıtkı Koçman Üniversitesi'nde verilmekte olan CENG 3550 "Decentralized Systems and Applications" dersinde aktif olarak kullanılmaktadır. Bu ortamın kullanımının; merkezi olmayan uygulama projeleri geliştirme süresini ve maliyetlerini azaltmada etkisi olacaktır. Detaylı bilgi için [5] nolu kaynağa bakınız.

1.4. AKILLI SÖZLEŞMELERDE GÜVENLİ KOD GELİŞTİRME

Akıllı sözleşme programlama, alışık olduğunuzdan daha farklı bir mühendislik zihniyeti gerektirir. Uygulamada başarısızlığın maliyeti yüksektir ve sonrasında değişim zordur. Bu da Ethereum akıllı sözleşme geliştirmesini bu yönlerden web veya mobil geliştirmeye kıyasla donanım programlamaya yakınlaştırmaktadır. Bu nedenle, bilinen güvenlik açıklarına karşı savunma yapmak yeterli değildir. Bunun yerine, yeni bir geliştirme felsefesi öğrenmemiz ve bu açıkları önceden incelememize olanak sağlayacak yeni araçlar geliştirmemiz gerekecektir.

Akıllı sözleşmelerin üzerinde çalışacağı blokzinciri ve özellikle Ethereum çok yeni bir teknoloji ve yeni plaformlardır. Bu teknoloji her geçen gün gelişmeye ve iyileşmeye devam etmektedir. Alışılagelmiş programlama dillerinin kaynak kodları yıllarca incelenmiş ve geliştirilmiştir. Solidity dilinin bu şekilde bir süreçten geçmemiş olmaması; merkezi olmayan sistemler geliştirirken hatalara ve diğer olası güvenlik açıklarına hazır olmamızı gerektirmektedir. Bu nedenle, bu bölümde sunulan güvenlik uygulamalarını takip etmek, bir akıllı sözleşme geliştiricisi olarak yapmanız gereken güvenlik çalışmasının yalnızca başlangıcıdır. Ethereum tabanlı akıllı sözleşme geliştirirken unutmamanız gereken altı temel kural Şekil 1.13'te verilmiştir.

Şekil 1.13. Akıllı Sözleşme Geliştirmede Altı Temel Kural

Şekil 1.13'tende görüleceği üzere; blokzincir temellerini ve ayrıntılarını öğrenmeden, üzerinde geliştirme yapacağınız sistemi anlamanız ve ona göre kod yazmanız pek olası değildir. Blokzincir teknolojisinin güncel ge-

lişmelerine de hakim olmak gerekir. Örneğin, Ethereum 2.0 ve yan zincir (side chain) gibi yapıları öğrenmek gereklidir. Bu sistemlerde kriptoloji çok önemlidir. Kriptoloji temelleri öğrenilmeli, anlaşılmalı ve sonra da unutulmamalıdır. Geliştirici bu kriptoloji esaslarının ne için kullanıldığının farkında olmalıdır. Sonraki aşamalar yazılım geliştirme süreçlerinde gerçekleşir. Hatalar her zaman olur, önemli olan hataya önceden hazırlanmaktır. Kod hataya zarif bir şekilde cevap vermelidir. İstisnalar (exception), hata yakalama blokları kullanılmalıdır. Dördüncü adım olarak; dikkatli kod yazılmalıdır. Dikkat, kod yazarken en önemli unsurdur. Dikkatli kod yazma stili geliştirmeye önem verilmelidir. Bu aşamada yapılması gerekenler güvenlik kontrol listesi bölümünde özetlenmiştir. Beşinci adım olarak; sade/basit (simple) tasarım oluşturmalıdır. Akıllı sözleşmeler basit tutulmalıdır. Akıllı sözleşmelerin, gelişmiş yazılım dilleri ve sistemleri gibi gelişmiş yapıları olmadığı için, yazılan sistemi basit tutmak oldukça önemlidir. Altıncı adım olarak; her zaman en son çıkan sürümler kullanılmalıdır. Temel programlama mantığına aykırı olsa da gelişmekte olan bu sistemlerin en son sürümlerini ve gelişmelerini takip etmek, güvenlik açıklarını daha önceden öğrenmek ve ona göre tedbir almak açısından önemlidir.

1.4.1. Harici Çağrılarda Dikkat Edilmesi Gerekenler

Ethereum blokzinciri kayıtlarında hali hazırda var olan, önceden blokzincirine yüklenmiş kontratlar; harici çağrı (external call) ile akıllı kontrat içerisinden kullanılabilir. Bu genelde adresi bilinen başka bir akıllı sözleşme (kontrat) üzerindeki bir fonksiyonu çağırma şeklinde olmaktadır. Harici çağrıların her zaman o sözleşmede veya bağlı olduğu herhangi bir başka sözleşmede kötü amaçlı bir kod olarak yürütülebileceğini unutmamanız gerekir. Ethereum ana ağında (Mainnet) uygulanacaksa, bu şekilde kullanılması çok yerinde olacaktır.

İlk olarak güvensiz kullanıma dair bir örnekle başlayalım. B kontratından A kontratındaki fonksiyonun çağrıldığı örnekte; IA bir interface olarak tanımlanmış ve B kontratı ile adresi verildikten sonra setNumber fonksiyonu içerisinden A kontratının fonksiyonu çağrılmıştır.

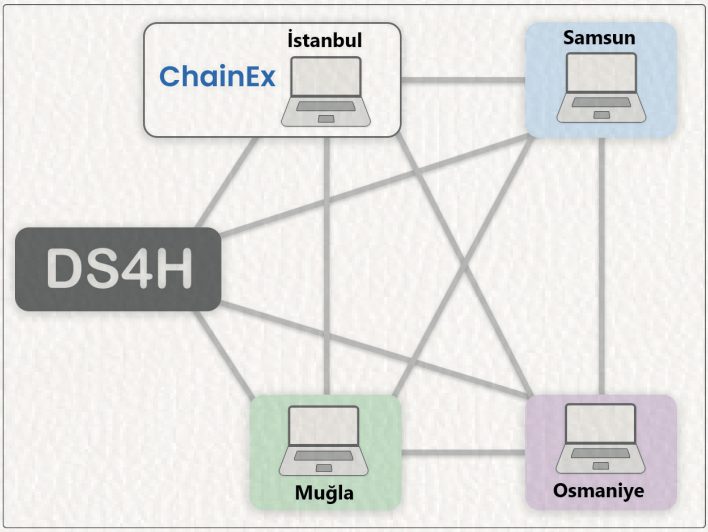
```
interface IA {
       function setNumber(uint n) external;
       function getNumber() external view returns (uint);
}
contract B {
       IA public a;
       function setContract(IA _a) public { a = _a; }
       function setNumber() public { a.setNumber(10); }
       function getNumber() public view returns (uint) {return
a.getNumber();}
}
```

Burada ilk aşamada unutulmamalıdır ki; A kontratı blokzincirine yüklendiğinde, uygulama ayrıntısı bilinmediği için B kontratı içerisindeki fonksiyonun ne yapacağı bilinmemektedir. Harici çağrılar ile iletişim halinde olduğunda; diğer kontratı yazılan akıllı sözleşme içerisinde güvensiz olarak işaretlemek önemlidir. Harici çağrılar ile yapılacak etkileşim mümkün olduğunca kodun sonlarına alınmalıdır. Bu etkileşimin kod içerisindeki en son işlem olması, kontratı olası hatalara karşı daha kapalı hale getirecektir. Aşağıdaki kontratta daha güvenli bir kodlama örneği verilmiştir. "untrusted_a" tanımı her ne kadar doğru olsa da, kontrol mekanizmasının doğru tanımlanması gerekir. Bu da "setContract" fonksiyonunda durum (state) değişikliği yapmadan önce "require" veya "assert" kontrollerinin yapılması demektir.

```
contract BB{
    IA public a;
    IA public untrusted_a;

    function setContract(IA _a) public {
        IA trusted_a =
IA(0xd9145CCE52D386f254917e481eB44e9943F39138);
        require(_a == trusted_a, "Address mismatch");
        a = _a;
    }
    function setUntrustedContract(IA _a) public { untrusted_a =
_a; }
    function setNumber() public { a.setNumber(10); }
    function getNumber() public view returns (uint) {return
a.getNumber();}
}
```

Harici çağrılarda meydana gelebilecek olası hataların; kendi yazdığınız kontratlarda kontrol edilmesi gerekir. Harici çağrılar, eğer bir adres üzerinde gidiyorsa, Solidity'nin alt düzey (low-level) çağrı özelliği ile "call", "callcode", "deletecall", "send" fonksiyonları üzerinden kullanılmalıdır. Bu alt düzey fonksiyonlar eğer bir hata var ise "false" döndürürler ve istisna fırlatmazlar. Bu yüzden "try-catch" blokları ile yakalamak yerine "false" kontrolü yapmak yeterli olacaktır. Şekil 1.14'te belirtilen şekilde yapılmaması gerekmektedir.



Şekil 1.14. DS4H Blokzincir Test Ağ Yapısı [5]

```
bad_address.send(33);
bad_address.call.value(33)(“”); // doğrudan diğer kontrat çağrıldı
```

Aşağıdaki şekilde yapın;

```
(bool success, ) = good_address.call.value(33)(“”);
if(!success) {
    // burada hata ile işlem yapabilirsiniz
}
```

Başka bir akıllı kontrat içerisinden, kontrat adresi ile çağrım yapılabilmektedir. "Address call" yapmak yerine, kontrat "import" ile kullanmak daha doğru olacaktır. Burada adresi bir interface ile kapatarak sadece belirli bir fonksiyonun çağrımı yapılabilecek hale getirilmiştir.

```
IA(trusted_address).setNumber(100)();
```

1.4.2. Kod Optimizasyonu

Ethereum'da akıllı sözleşmeler sistemde işlem (transaction) oluşturduklarında "gas" adı verilen ücreti ödemek durumundadır. Geliştiriciler; kodun sistemde harcayacağı masrafı (gas tüketim seviyelerini) kabul edilebilir bir seviyede tutmak için kodlarını iyileştirilmelidir. Çalışma (gas) maliyetlerini en aza indirirken, aynı zamanda kodun çalışma zamanını azaltmak hedeflenmelidir [20, 21].

Bu bölümde Solidity dilinde kod optimizasyonu için yapılabileceklere örnekler verilecektir. Öncelikle fonksiyon ve durum değişkenlerinin görünürlüklerinin (function visibility) düzgün ayarlanması gereklidir. Solidity dilinde fonksiyon ve durum değişkenlerinin (function visibility) "external", "public", "internal" ve "private" olmak üzere dört farklı görünürlüğü vardır. Fonksiyonlara ve değişkenlere herhangi bir görünürlük etiketi tanımlanmamışsa, varsayılan olarak "public" olarak kabul edilir. Değişkenler için otomatik olarak "set" ve "get" fonksiyonları oluşturulur. "external" fonksiyonların tanımlanması "public" olarak kullanılmasından daha doğrudur. Bu tip fonksiyonları "this" ile çağırmak mümkündür. "Public" olanlar ise doğrudan fonksiyon adı ile çağrılabilmektedir. "Internal" fonksiyonlar ve durum değişkenleri sadece içeride (mevcut ya da türetilmiş sözleşmeden) çağrılabilen fonksiyonlardır, bunlara dışarıdan erişim mümkün değildir. "Private" fonksiyonlar ve durum değişkenleri ise sadece bulunduğu kontrata özel olmaktadır. Eğer bu kontrattan türetilen başka kontrat varsa bile "private" fonksiyonlara erişim mümkün değildir. Fonksiyon tanımlarının aşağıdaki şekilde yapılmaması gerekir.

```
uint x; // public otomatik olarak get/set tanımlanır
function auto_public() { // public
}
```

Kod optimizasyonu ve okunabilirlik açısından fonksiyonların ve durum değişkenlerinin kullanıma göre isim ve görünürlük ayarlanması yapılmalıdır. Saldırılara karşı fonksiyonların varsayılan görünürlüğü[6] ve durum değişkeni varsayılan görünürlüğü[7] ayarlanmalı ve fonksiyon isimleri ile bunlar belirtilmelidir.

```
uint private x;
function external_function() external {
    // this.external_function() ile çağrılır
}
function internal_function() internal {
    // doğrudan internal_function() olarak this kullanmadan
çağrılır.
}
```

"abstract" ve "interface" kontrat tipleri bir seviyede yeniden kullanım sağlasa da, "interface" kontratların durum değişkenlerine erişemiyor olması, kod yazımı ve yönetimi bakımından önemlidir. "abstract" kontratlar, hem fonksiyon tanımı ve hem de o uygulama kodlarını içerdiğinden dolayı dikkat ve özel ilgi gerektirir. Yazılım mühendisliği uygulamaları gereği yeniden kullanım ve davranışsal kod geliştirmeye yönelik çalışmalar iyidir. Yine de bu tip kontratları kullanırken kod gözden geçirme stilleri tatbik edilmeli ve bilinmeyen/güvenilmeyen kaynaklardan kontratlar kullanılmamalıdır. Kodlanmış olan "abstract" kontratlar eğer adres tabanlı kullanım içeriyorsa "interface" üzerinden ulaşım sağlanacak şekilde değiştirilmelidir.

"fallback" (şalter) fonksiyonlar; her kontrat içerisinde sadece bir kere tanımlanabilen ve kontrat içerisinde eğer çağrılan isimde bir kontrat yoksa yerine çağrılan fonksiyonlardır. Bu fonksiyon, kontrata gönderilen tüm mesajlarda çalışır. Ayrıca kontrata doğrudan Ether gönderildiğinde de çalışırlar. Bu fonksiyona Ether göndermek hata atar çünkü ödenebilir (payable) bir fonksiyon değildir.

6 SWC-100, Function Default Visibility, https://swcregistry.io/docs/SWC-100
7 SWC-108, State Variable Default Visibility, https://swcregistry.io/docs/SWC-108

```
contract Test {
    uint x;
    fallback() external { x = 1; }
}
```

Fonksiyonlar "payable" olarak yazıldığında, kontrat artık Ether alabilir.

```
contract Test2 {
    uint x;
    fallback() external payable { x = 1; }
}
```

Fallback fonksiyonlarında yapılabilecek en doğru şey olay (event) tetiklemektir.

```
contract Test2 {
    uint x;
    event  LogFallbackFunction(address sender);
    fallback() external payable {
        emit LogFallbackFunction(msg.sender);
    }
}
```

Kontrat günlüğe alma (logging) için olay tetiklemesi kullanılmalıdır. Olaylar kontrat içerisinde kullanılan ve işlemler (transaction) sırasında olan değişimleri ve anlık durumları haber verebilecek günlük yapılarıdır. Bir kontrata atılan isteklerin durumları, o kontratın tüm işlem bilgileri o olay içerisinden bulunabilir.

Sayısal işlemlere de dikkat edilmelidir. Solidity, tüm tam sayı bölmelerini en yakın tam sayıya ve aşağıya doğru yuvarlar. Kesinliğe ihtiyacınız varsa, bir çarpan kullanmak veya hem payı hem de paydayı saklamak gerekir.

```
uint rounded_uint = 5 / 2 ;  // rounded_uint 2 olacaktır.
```

Solidity dilinde “float” ve ya “double” sayılar için ileri sürümlerde “fixed” anahtar kelimesi kullanılacaktır. Henüz en aktif versiyonda desteklenmemektedir. “fixed” in “float” ve “double” değişken tipleriyle farkı; tutulacak küsüratların, determinizim gereği belli olması gerekliliğidir. Bu tip bölme işlemlerinde, işlemlerin zincir dışında (off-chain) yapılması önerilmektedir. Bunun gerçekleşmemesi durumunda; ondalık hassasiyet/duyarlılık çarpan değeri ile belirlenmelidir.

```
uint carpan = 10;
uint bolum = (5*carpan) / 2; // bolum = 25 olacaktır.
```

Transfer fonksiyonu çağıran bir cüzdanın, bakiye kontrolünü ele alalım. Değer transferleri (asset transfer) ciddiye alınmalı ve assert(), require() doğru kullanılmalıdır. assert() ve require() fonksiyonları birbirlerine benzer olsalar da derlenmiş kod (opcode) bakımından kullanım alanları farklıdır. Her iki fonksiyon da kontrol için kullanılmaktadır ve eğer doğruluk sağlanmıyorsa hata fırlatacak yapıda tasarlanmışlardır. assert() fonksiyonu iç hatalar (internal errors) ve değişmeyen veriler için kullanılmalıdır.

```
assert(address(this).balance >= balanceBeforeTransfer);
```

Yukarıdaki kod parçacığı şu anda içerisinde yer aldığımız kontrat adresine gönderilen Ether miktarı ile transfer öncesindeki miktarı karşılaştırmaktadır. “require” ise sadece değerlerin (girdi değişkenleri, harici çağrıdan dönen değerler, akıllı sözleşme durum değişkenlerinin değerleri gibi) doğru olup olmadığını kontrol etmek için kullanılmalıdır.

```
require(balanceBeforeTransfer >= 0, “Balance değeri 0 dan büyük
veya eşit olmalı”);
```

Kontrol değeri olan "balanceBeforeTransfer" ile burada her zaman 0'dan büyük veya 0'a eşit olma durumu denetlenecektir. Eğer doğruluk sağlanmıyorsa, bir sonraki parametredeki mesaj ile hata atılacaktır.

```
function transfer(uint _amount, address to) public {
   require(_amount > 0,"Transfer tutarı 0'dan büyük olmalıdır");
   require(balanceOf[msg.sender] >= _amount,
        "Transfer tutarı cari hesap bakiyesinden büyük");
   // Temel toplama ve çıkarma işlemleri güvenli değildir
   balanceOf [to] += _amount;
   balanceOf [msg.sender] -= _amount;
   // bakiyeler için iç kontrol, hesapların güncellenmesi
   assert(balanceOf [msg.sender] + _amount >= balanceOf[to]);
}
```

Yukarıda görüldüğü gibi, transfer fonksiyonu önce değer kontrollerini ve bakiye (balance) kontrollerini yapmaktadır. Bu "require" fonksiyonu ile yapıldığında olası hataların önüne geçilmiş olacaktır. Sonrasında yapılacak işlemler, bir toplama çıkarma fonksiyonu olabileceği gibi, bir harici çağrı da olabilir. Burada olası iç hataların önüne geçilmesi için; fonksiyon bitiminde "assert" ile tekrar doğrulama yapılarak işlemlerin sağlaması yapılmalıdır.

```
modifier hasBalance(uint _amount) {
require(_amount > 0,"Transfer tutarı 0'dan büyük olmalıdır");"
require(balanceOf[msg.sender] >= _amount,
"Transfer tutarı cari hesap bakiyesinden büyük");
_; }
// hasBalance fnks ihtiyaç olan her yerde kullanılabilecek
function transfer_with_modifier(uint _amount, address to)
hasBalance(_amount) public {
balanceOf[to] += _amount;
balanceOf[msg.sender] -= _amount;
// bakiyeler için iç kontrol, hesapların güncellenmesi
assert(balanceOf[msg.sender] + _amount >= balanceOf[to]);
}
```

"Modifier" lar sadece kontrol amaçlı kullanılmalıdır. Solidity "modifier" anahtar kelimesi ile fonksiyonlar birer ön çağrı mekanizması haline dönüştürülmektedir. Buradaki amaç, "modifier" kullanan fonksiyon öncesinde başka bir kontrol fonksiyonunun çağrılması ve bu sonuca etki edecek olası değişikliklerin kontrol edilmesidir. "Modifier"lar, bir fonksiyonun iç yapısında durum değişkenlerini değiştirilebildiği için dikkatli olunmalıdır ve bu yapının önce kullanılması önemlidir. Başka akıllı sözleşmelerin içerisindeki modifier fonksiyonlardan varolan koda iç aktarım (import) yapılacaksa; kullanılmadan önce kod güvenliği açısından gözden geçirilmelidir. Modifier fonksiyonların bir diğer amacı da kod okunurluğunu arttırmaktır. Modifer kullanımı yerine "require" ve "assert" fonksiyonlarını kullanabileceği de unutulmamalıdır.

1.5. AKILLI SÖZLEŞMELERE SALDIRILAR

Akıllı sözleşmeler, günümüzde finansal değer transferinde yaygın olarak kullanılmaktadır. Bu sözleşmelerdeki olası zafiyetlerin kötüye kullanımı ve elde edilebilecek finansal getiri bilgisayar korsanlarının ilgisini çekmektedir. Bu bölümde bu konular hakkında temel bilgiler verilecektir. Akıllı sözleşmelerin zayıflıklarının sınıflandırılması ve testi için "SWC Registry" iyi bir kaynaktır[8]. Bu konuda bilinen saldırılar Consensys'in Github sayfasındaki en iyi uygulamalar dökümanında[9] anlatılmıştır. Akıllı sözleşme bazlı saldırılar ve buna karşı yapılabilecek korumalar literatürde [20-23] nolu çalışmalarda ele alınmıştır. Marchesi ve arkadaşlarının çalışmasında [20]; akıllı sözleşmelere olan saldırı süreci ayrıntılı olarak ele alınmış ve önerilerde bulunulmuştur. Akıllı sözleşmelerin güvenliği hakkında yapılan akademik çalışmalar ve olası araştırma konuları güncel bir yayında [21] incelenmiştir.

Akıllı sözleşmelere saldırılardan başlıcaları Tablo 1.1'de özetlenmiştir. Ethereum ağına özel akıllı sözleşme saldırıları Tablo 1.2'de verilmiştir.

8 SWC Registry - Smart Contract Weakness Classification and Test Cases, https://swcregistry.io/
9 Ethereum Smart Contract Best Practices, Known Attacks, https://consensys.github.io/smart-contract-best-practices/known_attacks/

Tablo 1.1. Akıllı Sözleşmelere olan temel saldırılar [22]

Saldırı Adı	Tür	Çalışma Şekli	Etkisi	Çözüm Önerisi
Yeniden Giriş	Özelliğin kötüye kullanılabilirliği	Ana işlev bir döngü ile özyinelemeli (recursive) bir geri aramasını yürütür	Sözleşmeyi tamamen yok edebilir veya değerli bilgileri çalabilir.	Güvenli kod geliştirme prensiplerinin uygulanması
Akıllı Sözleşme Taşması (overflow) ve Azalması (underflow)	Yetkisiz (unauthorized) girdilerin kabulü	Saldırganın maksimum değerden daha fazla değer göndermesi veya minimum değerden daha az değer göndermesi	Elinde bulunandan çok daha fazla jetonu (token) sistemden çekebilir.	Güvenli kod geliştirme prensiplerinin uygulanması
Delegatecall	Özelliğin kötüye kullanılabilirliği	Bir sözleşmeyi çağırmak için kullanıldığında; yürütülen kod dışında, msg.sender ve msg.value değişmez. Yeniden kullanılabilir kod oluşturularak ani kod yürütme şansını artırır.	Özel kitaplıklar oluştururken kusurlar ortaya çıkabilir, yeni güvenlik açıklarına yol açabilir	- Kitaplık ve çağrı sözleşmelerinde bir gecikme gözlemlendiğinde akıllı sözleşmenin çağrılması - Mümkün olduğunca durumsuz kitaplıkların geliştirilmesi
Varsayılan Görünürlükler	Varsayılan Ayarların kötüye kullanılması	Görünürlük belirtecinin, kullanıcıların türetilmiş sözleşmelerle harici işlevleri aramasına izin verirken kontrolü ele alması.	Saldırgan kendi amacı için kodu kötüye kullanabilir.	Kod yazarken görünürlük belirteci özel olarak ayarlanmalı

Tablo 1.2. Ethereum Ağına Özel Akıllı Sözleşme Saldırıları [22]

Saldırı Adı	**Tür**	**Çalışma Şekli**	**Etkisi**	**Çözüm Önerisi**
Kısa Adres Saldırısı	Girdi onaylama hatası	Ethereum Sanal Makinesi (EVM) zafiyetinde; saldırgan kasıtlı olarak adresin sonundaki "0" değerini göndermez.	Ciddi sorunlara yol açabilir	Girdi kontrolü yapılması
İşlem Sırasına Bağlılık (TOD)	Protokolün kötüye kullanılması	Bir işlemdeki akıllı sözleşme, bir diğer işleme bağlı çalışıyorsa; bloğu oluşturan madenci, bloktaki işlemlerin sırasını etkileyebilir.	Ethereum ağında bazı işlemler kasıtlı olarak çalıştırılmayabilir.	Otonom kodlarla kötüye kullanımın kontrolü
Zaman Damgası Bağımlılığı	Protokolün kötüye kullanılması	Akıllı sözleşme, madencilerin blok doğrulamasından sonraki 30 saniye içinde bir zaman damgası koymasına izin verdiğinden, bu değer fayda sağlamak için değiştirilebilir.	Ethereum ağı ve benzer çalışan ağlarda; madencilerin haksız kazanç elde etmesi.	Zaman damgası ile ilgili protokolün iyileştirilmesi

1.6. BLOKZİNCİR TESTLERİ

Blokzincir ağlarının ve merkezi olmayan uygulamaların başarım ölçütleri yeterince test edilmemektedir. Bu kısımda öncelikle blokzincir test ağlarından söz edilecektir. Yapılabilecek testler; blokzincir ağının performans testleri ve akıllı sözleşme testleri olarak ele alınacaktır. Geliştiricilerin kendi akıllı sözleşme test araçlarını oluşturmaları için örnek verilecektir. Güvenlik analiz araçlarından söz edilecektir.

1.6.1. Blokzincir Test Ağları

Yazılımları gerçek blokzincir ağında denemeden önce, var olan test ağları (testnet) üzerinde denenmesi gerekecektir. Her platform için çeşitli bulut tabanlı test ağları bulunmaktadır. Ethereum dışındaki platformlara ait test ağlarında özel bir öğrenme süreci ve kullanım ücreti gerekmektedir. Blokzincir test ağları genel kayıt defterleri kullanır ve bu kayıtların görünür olması yüzünden birçok proje süreçlerinde tercih edilmez. Geliştirici sürecin gizli kalmasını istediğinde kendisine özel bir ağ kurması gerekecektir. Günümüzde bu tür ağlar docker gibi konteyner (container) teknolojileriyle çok kısa bir sürede kurulabilmektedir. Geliştirici kendi makinesinde veya bulut ortamında belirli sayıda düğümü çalıştırarak test işlemlerini gerçekleştirir.

Test ağları, ölçeklenebilirlik ve performans açısından esnek değildir ve başarım testlerinde gerçekçi sonuçlar vermezler. Gerçekçi bir test ortamı oluşturmak için fiziksel bir test ağına ihtiyaç duyulmaktadır. Merkezi olmayan teknolojilerin araştırılıp geliştirilebileceği sürdürülebilir ve ölçeklenebilir bir araştırma ortamı olarak DS4H (Decentralized solutions for humanity - İnsanlık için Merkezi Olmayan Çözümler) kurulmaktadır. Şekil 1.14'te ağın kurulan ilk düğümleri gösterilmiştir. ISOC (Internet Society)'dan alınan hibe desteği ile DS4H'ın ilk deneme düğümleri Türkiye'nin dört farklı şehrinde (İstanbul, Samsun, Muğla, Osmaniye) konumlandırılmıştır.

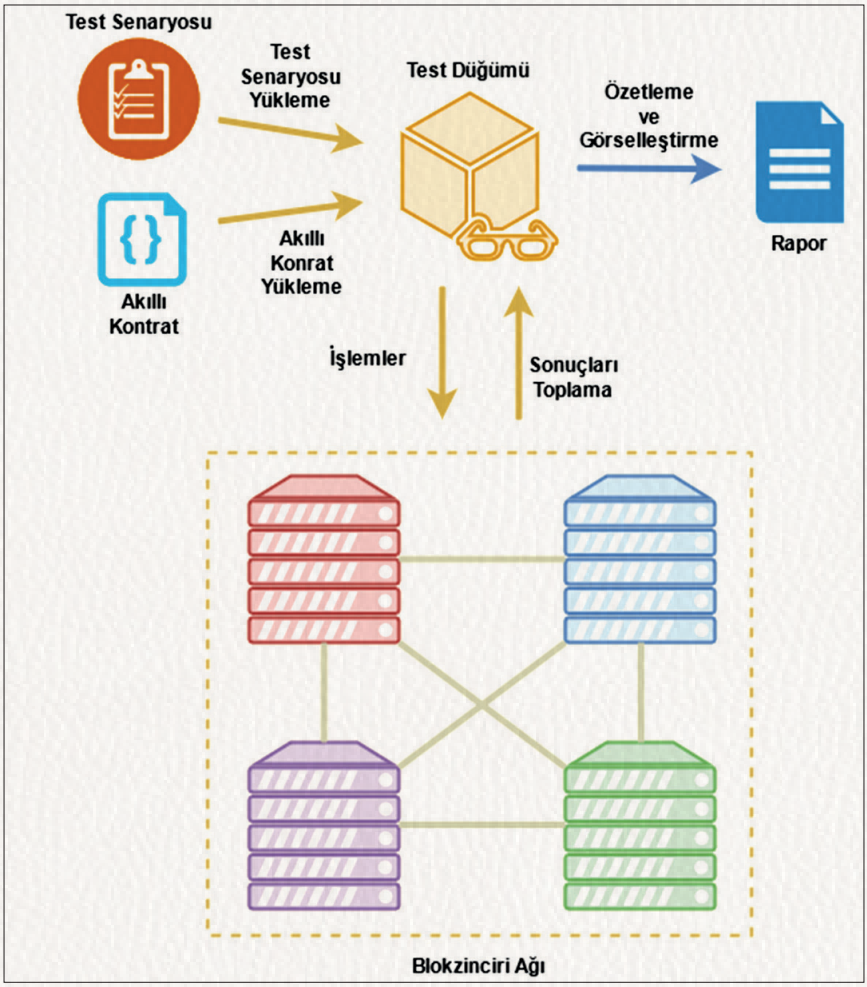
DS4H blokzincir araştırma ağının testleri ve yayın süreçleri sürmektedir [24]. Ağın otonom kodlarla yönetimi için çalışmalar devam etmektedir. Projenin bir sonraki aşamasında, araştırma kurumlarının ağa dahil olarak araştırmacılarını bu ağdan yararlandırması mümkün olacaktır. Bu araştırma ağına ait gelişmeler için DS4H proje sayfası[10] ve sosyal medyada ds4h etiketi (hashtag) takip edilebilir.

1.6.2. Blokzincir Ağının Başarım Testleri

Blokzincir ağının başarımına dair bir fikir elde edebilmek için; gecikme (latency), birim zamanda üretilen iş (throughput), kaynak kullanımı ve başarısız/gecikmiş işlem değerleri ölçülmelidir. Gecikme değerleri olarak blokzincirinde bir işlemin gerçekleşmesindeki gecikme (transaction latency) ve kayıtlardan okumadaki gecikme (read latency) ölçülür. Sistemde birim zamanda üretilen iş olarak da benzer süreçlerin hesaplamaları yapılır. Sistemdeki düğümlerin bu süreçler boyunca kaynak (işlemci, bellek, ağ, vb.) kullanımı da ölçülür [6]. Zaman aşımları nedeniyle oluşan başarısız/gecikmiş işlemlerin sayısı da takip edilmesi gereken önemli bir parametredir [25].

Başarım değerlendirmesi için kullanılabilecek örnek test süreci Şekil 1.15'te gösterilmiştir. Test edilecek blokzincir ağı ile iletişim kuracak en az bir test düğümü olmalıdır. Test senaryosu; test işlemlerinin kapsamını içeren bir konfigürasyon dosyasıdır. Sistem uygulamaya dair özel bir akıllı sözleşme ile test edilebileceği gibi bu süreçte varsayılan (default) bir akıllı sözleşme de kullanılabilir. Test düğümü; senaryoya göre sistemde yükü oluşturur ve ardından sonuçları gözlemler. Test sonuçları özetlendikten sonra görselleştirilerek rapor oluşturulur [5]. Düğümlerin kaynak kullanımı düğümlerin kendilerinde kurulacak sistemlerle veya SNMP (Simple Network Management Protocol) gibi protokoller aracılığıyla da takip edilebilir.

10 Decentralized solutions for humanity (DS4H) blokzinciri araştırma ağı proje sayfası http://wiki.netseclab.mu.edu.tr/index.php?title=DS4H



Şekil 1.15. Blokzincir Ağ Başarım Test Süreci [6]

Blokzincir ağlarında kullanılabilecek çeşitli blokzincir ağ başarım test araçları bulunmaktadır, ancak araçların çoğu belirli blokzincir platformları için oluşturulmuştur ve karmaşık konfigürasyon gerektirir. Günümüzde en kapsamlı başarım kıyaslama ortamı Hyperledger Kaliper[11]'dir. Quorum dışında birçok platformu desteklemektedir. Blokzinciri ağ başarım testlerine dair ör-

11 Hyperledger Kaliper, https://hyperledger.github.io/caliper/

nek çalışmalar [25-28]'de verilmiştir. Quorum ortamında Chainhammer[12] ve Quorum Profiling[13] gibi başarım test araçları bulunmaktadır. Bu araçların kullanımları kolay değildir, karmaşık kurulum adımlarına ihtiyaç duyulmaktadır. Chainhammer'ın bazı gerekli bağımlılıkları güncel değildir [6]. GoHammer aracı, Ethereum/Quorum blokzincir platformları için kullanımı kolay, esnek bir test aracı sağlamak üzere geliştirilmiştir. Saniyede işlem (TPS) değerleri ve çeşitli başarım ölçümlerinin test edilmesi için kullanılabilmektedir. GoHammer üzerindeki geliştirmeler devam etmektedir. Detaylı bilgi için [6] nolu kaynağa bakınız.

1.6.3. Akıllı Sözleşme Test Araçları

Aktif olarak hem geliştirmesi hem de desteği süren çeşitli akıllı sözleşme test servisi ve araçları da mevcuttur. Bunlardan başlıcaları olarak OpenZeppelin[14], hardhat[15], waffle[16] projelerinden söz edilebilir. OpenZeppelin geliştiricilere akıllı kontratlar için bir çatı ve test ortamları için bir kontrol kütüphanesi sunmaktadır. OpenZeppelin kütüphanesinde yer alan temel kontratlar yardımıyla daha önce test edilmiş ve testleri geçmiş bilinen akıllı kontratları kullanmak, test kodlarını azaltacağı gibi basit tasarım yaklaşımımızla da örtüşmektedir. Hardhat; derleme, konuşlandırma, test ve hata ayıklama kısımlarında işe yarayan çok yetenekli bir javascript kütüphanesidir. Hardhat kütüphanesi içerisinde bir Ethereum ağıyla gelmekte ve yapılan tüm işlemler için onu kullanmaktadır. Bu kütüphanenin temel işlevselliği; yığın izleri, günlüğe alma (logging) ve işlemler başarısız olduğunda açık hata mesajları içeren Solidity hata ayıklamasıdır. Waffle ise daha çok javascript kütüphanesi olarak karşımıza çıkmaktadır. Hardhat ile uyumlu çalışması sayesinde, bir test aracı olarak akıllı kontratlar haricinde yapılabilecek testler için temel oluşturmaktadır.

12 Chainhammer, https://github.com/drandreaskrueger/chainhammer
13 Quorum Profiling, https://docs.goquorum.consensys.net/en/stable/Concepts/Profiling/
14 OpenZeppelin, https://github.com/OpenZeppelin
15 Hardhat, https://hardhat.org
16 Waffle, https://getwaffle.io/

1.6.4. Kendi Akıllı Sözleşme Test Kontratınızı Oluşturmak

Geliştiriciler kendi test kodlarını da yazabilirler. Böyle bir kontrat her türlü genel ve özel ağlarda test amaçlı kullanılabilmelidir. Bunun için ek bir araç ya da özel bir yazılıma ihtiyaç duyulmamalıdır. Bu bölümde buna bir örnek verilecektir. Aşağıdaki kod parçacığı; tüm Solidity örneklerinde kullanılan "Simple Storage" kontratıdır. Örneklerde bizlere gösterilen sadece kontratın kendisidir, Solidity dilini kullanarak bu tip basit bir kontratın testini yapmak ise oldukça karmaşık olabilir. Truffle gibi araçlarla farklı ortamlarda farklı şekillerde testler yapılabilse de, Solidity dilinde yazılmış kodların (en basit anlamıyla) testi için sadece "assert" yeterli olabilmektedir.

```
contract SimpleStorage {
  uint public storedData;
  constructor() public {
    storedData = 100;
  }
  function set(uint x) public {
    storedData = x;
  }
  function get() public view returns (uint retVal) {
    return storedData;
  }
}
```

Yukarıdaki kontrat, oluşturulduğunda içerisindeki bir değişkene 100 değeri atamaktadır. Bu değer daha sonra "set" fonksiyonu ile de değişebilmektedir. Ayrıca kayıt edilen bu değişkenin değerini bize gösteren fonksiyon olarak da "get" kullanılmaktadır. Bu tip bir kontratı test edebilmek için başka bir test kontratı yazacağız. Bunun için o kontratı yazacağımız test kontratı içerisinde oluşturmamız gerekiyor. Böylelikle deterministik bir test kontratımız olacaktır. Aşağıda Remix IDE ile de kullanabileceğiniz "SimpleStorageTest" isimli test kontratı verilmiştir.

```
import "SimpleStorage.sol";
contract SimpleStorageTest {

  // Test edilecek kontrat değişken olarak tanımlanır
  SimpleStorage storage_test;
  // basit bir toggle değeri ile test fonksiyonlarının fail etmesi sağlanır
  uint i = 0;

  function beforeEach() public {
    // test edilecek kontrat burada oluşturulur
    storage_test = new SimpleStorage();
    if (i == 1) {
        storage_test.set(200);
    }
    i += 1;
  }

  function initialValueShouldBe100() public {  // ilk değerin 100 olması testi
        beforeEach();
        // değeri 100 olarak bekliyoruz, ilk seferde geçecektir.
        // 2. sefer de fail
        assert(storage_test.get() == 100);
  }
  // set fonksiyonun çağrılması ve değeri 200 yapma testi
  function valueShouldBe200() public {
      beforeEach();
      assert(storage_test.get() == 200);
  }
}
```

"beforeEach" fonksiyonu tüm alt test fonksiyonlara eklenmiş ve fonksiyon isimleri yapılacak testin içeriğini belirtecek şekilde ayarlanmıştır. Böylelikle hangi fonksiyonda neyi test ettiğimizi görme imkanı elde edilmektedir. Farklı bir yöntem olarak "constructor" oluşturularak diğer test fonksiyonları da oradan çağırılabilir. Gelişmiş programlama dillerinde de benzer "main" fonksiyonlarla test etme yöntemleri kullanılmaktadır. Hali hazırda yazılan akıllı sözleşmeleri çok kompleks tutmadan ilerleme tavsiyemiz ise, araçlara gerek kalmadan da bu testlerin yazılabileceğini göstermek amacıyla verilmiştir.

```
import "SimpleStorage.sol";
contract SimpleStorageTest {

  SimpleStorage storage_test;
  uint i = 0;

  constructor() public{
      storage_test = new SimpleStorage();
      initialValueShouldBe100();
      storage_test.set(200);
      valueShouldBe200();
  }
  function initialValueShouldBe100() private {
        assert(storage_test.get() == 100);
  }
  function valueShouldBe200() private {
      assert(storage_test.get() == 200);
  }
}
```

1.6.5. Güvenlik Analiz Araçları

Güvenlik analiz araçları üzerine ayrıntılı bir inceleme olarak; Durieux ve arkadaşlarının çalışması [32] bulunmaktadır. Bu çalışmada; belli başlı dokuz otomatik analiz aracının geçerliliği güvenlik açığı olan 69 akıllı sözleşme üzerinde denenmiştir. Sonrasında Etherscan'de bulunan 47.587 akıllı sözleşme üzerinde denetlenme gerçekleştirilmiştir. Bu deneylere göre; bu akıllı sözleşmelerinin %97'sinde zafiyet bulunmuştur. Denenen araçların dört veya daha fazlasının aynı anda bulduğu zafiyetler ise azdır. Özetle, ancak birden fazla aracı çalıştırdığımızda anlamlı bir sonuç elde edilebilmektedir.

Ethereum'da akıllı sözleşme güvenlik analizi süreçlerinde kullanılabilecek belli başlı güvenlik aracı olarak Slither[17], Echidna[18] ve Manticore[19] mevcuttur. Slither, Solidity statik analiz aracıdır, sözleşme ayrıntıları hakkında görsel bilgiler yazdırır ve özel analizleri kolayca gerçekleştirmek için bir API sağlar. Slither, geliştiricilerin güvenlik açıklarını bulmasını ve özel analizleri hızlı bir şekilde oluşturabilmesini sağlar [29]. Slither, CI/CD (continuous integration

17 Slither, https://github.com/crytic/slither
18 Echidna, https://github.com/crytic/echidna
19 Manticore, https://github.com/crytic/building-secure-contracts/tree/master/program-analysis/manticore

- sürekli entegrasyon / continuous development - sürekli geliştirme) sisteminize, örneğin Github Actions'a entegre edilerek kullanılabilir.

Echidna, Ethereum akıllı sözleşmelerin bulandırma (fuzzing) ve özellik tabanlı testi (property-based testing) için tasarlanmış bir Haskell programıdır. Kullanıcı tanımlı tahminleri (predicates) veya Solidity iddialarını (assertions) yanıltmak (falsify) için bir ABI (Application Binary Interface - Uygulama İkili Arayüzü) sözleşmesine dayalı karmaşık dil bilgisi tabanlı bulandırma yöntemleri kullanır. Kodu sözde rasgele (pseudo random) olarak üretilen işlemlerle çalıştırır. Bulandırma aracı, belirli bir özelliği ihlal etmek için bir dizi işlem bulmaya çalışacaktır.

Manticore [31], akıllı sözleşmelerdeki hataları otomatik olarak bulmak için kullanılır. Sembolik çalıştırma (symbolic execution) ortamı sunmaktadır. Her çalıştırma yolunu (execution path) matematiksel bir formüle çeviren ve üzerinde üst (top) kısıtlamaların denetlenebileceği formal bir doğrulama tekniği kullanılabilmektedir. API aracılığıyla akıllı sözleşmeler manipüle edilebilir, kısıtlamalar eklenebilir.

Bu araçların ve kullandıkları yöntemlerin kıyaslaması Ethereum'un resmi sayfasında[20] verilmiştir. En yaygın kullanılan üç ana güvenlik test ve analiz tekniğinin kıyaslanması Tablo 1.3'te verilmiştir. Statik analiz yönteminde test süreci çok kısa (saniyeler) sürmektedir, kaçırılan hatalar (bug) ortalama miktardadır, yanlış alarm oranı düşüktür. Bulandırma yönteminde test süreci kısadır (dakikalar), kaçırılan hatalar (bug) düşük miktardadır, yanlış alarm oranı yoktur. Sembolik çalışma yönteminde test süreci saatler sürmektedir, kaçırılan hata ve yanlış alarm oranı yoktur.

Tablo 1.3. Üç Ana Güvenlik Test ve Analiz Tekniğinin Kıyaslanması

Teknik	**Araç**	**Kullanımı**	**Çalışma Süresi**	**Kaçırılan hatalar**	**Yanlış Alarm Oranı**
Statik Analiz	Slither	Komut satırı, betik	Saniyeler	Ortalama	Düşük
Bulandırma	Echidna	Solidity özellikleri	Dakikalar	Düşük	Yok
Sembolik Çalıştırma	Manticore	Solidity özellikleri, betik	Saatler	Yok	Yok

20 A guide to Smart Contract Security Tools, https://ethereum.org/hr/developers/tutorials/guide-to-smart-contract-security-tools/

1.7. GÜVENLİK KONTROL LİSTESİ

Güncel bir çalışmada [20]; güvenlik riskleri ele alınmış ve bir güvenlik denetim listesi önerilmiştir. Ethereum'un resmi sayfasında bulunan güvenlik listesinde[21] de pratiğe yönelik öneriler bulunmaktadır. Bu alt başlıkta aksi belirtilmedikçe bu kaynaklardaki içeriklerden yararlanılmıştır. Bu kaynaklardan ve deneyimlerimizden yararlanılarak kategoriler oluşturulmuş ve geliştiriciler için güncel bir güvenlik kontrol listesi Tablo 1.4'te sunulmuştur. Ana kontroller, kısıtlamalar, ek destekler, yazılım mühendisliği pratikleri, testler ve belgeleme olarak kategoriler tanımlanmıştır. Bu tablonun en güncel hali proje github sayfasından[22] erişilebilir olacaktır.

Güvenlik için öncelikle kontratta mantık kontrolü yapılmalıdır. Taşmalardan (overflow), yetersizliklerden (underflow) veya sonlu aritmetiğin diğer istenmeyen özelliklerinden korunmak için bir mantık oluşturulmalıdır. Koruma kontrolü (guard check) de gerçekleştirilmelidir. Akıllı sözleşmenin durumu ve fonksiyon girdilerine dair tüm gereksinimlerin karşılandığından emin olunulmalıdır.

Fonksiyon etkileşimleri kontrol edilmelidir. Bir sözleşmede bir fonksiyonu gerçekleştirirken öncelikle tüm ön koşullar kontrol edilmelidir. Sonrasında sözleşmenin durumuna etkiler (effect) uygulanmalıdır. En son aşamada diğer sözleşmelerle etkileşime geçilmelidir.

Mahremiyetin sağlanıp sağlanmadığı kontrol edilmelidir. Karaarslan ve Konacaklı'nın çalışmasında [7] belirtildiği üzere, blokzincirinde kişisel veri tutulması tavsiye edilmemektedir. Sıfır bilgi kanıt (zero knowledge proof) [15] protokollerinin kullanımı yerinde olacaktır. Yine de kişisel verilerin tutulması gerektiğinde; bu tür veriler şifrelenerek kayıt defterinde saklanmalıdır. KVKK [33], GDPR (general data protection regulation - genel veri koruma yönetmeliği) [34] gibi yasal gereksinimler karşılanmalıdır. Bu gereksinimleri karşılarken coğrafi konumdan ziyade, hangi ülkelerin vatandaşının kişisel verilerinin tuttulduğu önemlidir. Örneğin, Avrupa Birliği bir ülkenin vatandaşına ait kişisel veriler tutuluyorsa GDPR gereksinimleri karşılanmalıdır.

21 Smart Contract Development Checklist, https://ethereum.org/hr/developers/tutorials/secure-development-workflow/

22 MSKÜ Smart Contract Security Testing, https://github.com/MSKU-BcRG/SC-SecTesting

Yetkilendirmelerin gerçekleştirilmesi kontrol edilmelidir. Kritik metodlar ancak belirli kullanıcılar tarafından yürütülmelidir. Bu da adreslerin eşlenmesi kullanılarak gerçekleştirilir ve "modifier" kullanılarak kontrol edilir. Bunun için sahiplik de tanımlanmalıdır. Sözleşme yönetiminden sorumlu olan ve özel izinlere sahip olan sözleşme sahibi belirtilmelidir. Bu da muhtemelen kritik metodları çağırmaya yetkilendirilen o tek adres olacaktır.

Akıllı sözleşmenin sonlandırması denetlenmelidir. Bir akıllı sözleşmenin kullanım ömrü sona erdiğinde sonlandırılmalıdır. Bir sözleşmeyi feshetme yetkisi genellikle sözleşme sahibindedir. Sözleşmeye önceden tasarlanmış (ad-hoc) kod ekleyerek veya "selfdestruct" (kendi kendini yok etme) işlevini çağırarak bu gerçekleştirilir.

Sözleşmelerin özel durumları da göz önünde bulundurulmalıdır. Sözleşmelerin yükseltilebilir olup olmadığı kontrol edilmelidir. Sözleşmelerin ERC'lere (Ethereum Request for Comments - Ethereum yorumlar için rica) uygunluğu denetlenmelidir. ERC'ler Ethereum kullanımında belirli bir standardı tanımlamak için yapılan EIP (Ethereum Improvement Proposal - Ethereum iyileştirme önerisi)'lere verilen etiketlerdir. ERC-20 jeton (token) standardı buna örnek verilebilir. Üçüncü taraf jetonlarıyla entegrasyon söz konusu ise; ilgili denetim listesine[23] göre davranmak yerinde olacaktır.

Kısıtlamaların gerçekleştirimi kontrol edilmelidir. Bakiye limiti kullanarak bir akıllı sözleşme içinde tutulan maksimum fon (fund) miktarı sınırlanmalıdır. Oran sınırlaması (rate limit) gerçekleştirilmelidir. Bir akıllı sözleşmeye gönderilen mesaj sayısını ve dolayısıyla hesaplama yükünü sınırlamak için; bir görevin belirli bir süre içinde ne sıklıkta yürütülebileceği düzenlenmelidir. Hassas görevlerde hız kısıtlaması (speed bump) uygulayarak süreç yavaşlatılır. Kötü niyetli eylemlerin gerçekleşmesi durumunda hasar sınırlı olacak ve buna karşı önlem almak için daha fazla zamanınız olacaktır. Zaman kısıtlaması yaparak eyleme ne zaman izin verileceği de belirtilebilir. Bu da işlemi tutan bloktaki kayıtlı zamana dayanacaktır. Zaman kısıtlaması; hız kısıtlaması ve oran sınırlaması yapılarıyla da birlikte kullanılabilir.

Vekil (proxy) ve kahin (oracle) gibi ek desteklerin kullanımı da mümkündür. Vekil temsilcileri veya vekil kalıpları; akıllı sözleşmelerin sürüm yükseltil-

23 Token integration checklist. https://ethereum.org/hr/developers/tutorials/token-integration-checklist/

mesini kolaylaştırmak için kullanılabilecek bir dizi akıllı sözleşmedir. Vekil, aslında adresi değiştirilebilen bir başka akıllı sözleşmedir. Vekil kullanımında yeni akıllı sözleşmeyi gösterecektir. Bu da blokzincir kaynaklarının idareli kullanılmasını sağlayarak gaz tasarrufu sağlamaktadır. Kahin, blokzinciri dışından veri sağlayan bir akıllı sözleşmedir. Kahin güvenilir bir kaynak tarafından veri ile güncellenmektedir. Ters kahin olarak kullanımında; akıllı sözleşmenin zincir dışı bileşenlere veri sağlaması söz konusudur.

Yazılım mühendisliği pratikleri kontrol edilmelidir. CI/CD (sürekli entegrasyon ve sürekli geliştirme) süreçlere dahil edilmelidir. Geliştirilen uygulama ya da yazılım eğer genel ağlarda çalışacaksa, anahtar bilgileri paylaşımı doğru olmayacağı için CI/CD süreçleri için çevrimiçi servis kullanımı doğru olmaz. Bu süreçleri kontrol etmek için olay yapısının takip edilmesi için bir yapı kurulması tavsiye edilir. Bunun yanı sıra; yeniden kullanılabilirlik yapılarından ve mutex'den yararlanılması iyi bir pratiktir. Çoklu oluşumlar (multiple instance) için sözleşme kitaplıkları ve şablonları kullanılır. Mutex, paylaşılan bir kaynağa eşzamanlı erişimi kısıtlamak için kullanılan bir mekanizmadır. Harici bir çağrının; onu çağıran fonksiyona yeniden girmesini engellemek için kullanılmalıdır.

Bir önceki bölümde ele alınan güvenlik analiz araçları ile güvenlik testleri gerçekleştirilmelidir. Bilinen güvenlik sorunlarını kontrol etmek için Slither, Echidna, Manticore araçları ve benzerleri ile sözleşmeler sürekli olarak gözden geçirilmelidir. Otomatik araçların her türlü sorunu bulamayacağının farkında olmak gerekir. Özellikle mahremiyet, önden giden (front running) işlemler, kriptografik işlemler ve harici DeFi (decentralized finance - merkezi olmayan finans) bileşenleriyle riskli etkileşimlere dikkat edilmelidir. Yazılımı oluşturan parçaların gerektiği gibi çalıştığından emin olmak için birim testleri de gerçekleştirilmelidir.

Belgeleme en önemli aşamalardandır. Kodun kritik güvenlik özellikleri öncelikle görsel olarak incelenmelidir. Bu süreçte Slither'ın "inheritance-graph" (miras grafik) oluşturucusu kullanılabilir. Fonksiyon görünürlüğünü ve erişim kontrollerini raporlamak için Slither'ın işlev özeti oluşturucusundan yararlanılabilir. Durum değişkenleri üzerindeki erişim kontrollerini raporlamak için Slither'ın değişken ve yetkilendirmeli (vars-and-auth) oluşturucusu kullanılabilir. Kodun kritik güvenlik özellikleri belgelenmelidir ve bunları değerlendirmek için otomatik test oluşturucularından yararlanılabilir. Öncelikle kod

için güvenlik özelliklerinin belgelenmesi öğrenilmelidir. Farklı araçlar kullanılarak; sonlu otomata, erişim kontrolleri, aritmetik işlemler, harici etkileşimler, standartlara uygunluk, kalıtım, değişken bağımlılıkları, erişim kontrolleri ve diğer yapısal konulara odaklanılmalıdır.

Tablo 1.4. Akıllı Sözleşme Güvenlik Kontrol Listesi

Güvenlik Kontrol Listesi	**Yapılma Durumu**	**Notlar**
Mantık Kontrolü	□	
Koruma Kontrolü	□	
Fonksiyon Etkileşimleri Kontrolü	□	
Mahremiyet	□	
Yetkilendirme ve Sahiplik	□	
Sonlandırma	□	
Yükseltilebilirlik ve Özel Durumlar	□	
Kısıtlamalar		
Bakiye Limiti Kullanımı	□	
Oran Sınırlandırması	□	
Süreçlerin Yavaşlatılması	□	
Zaman Kısıtlaması	□	
Ek Destek		
Vekil Temsilcisi Kullanılması	□	
Kahin Kullanımı	□	
Yazılım Mühendisliği Pratikleri		
Sürekli Entegrasyon (CI)	□	
Sürekli Geliştirme (CD)	□	
Yeniden Kullanılabilirlik	□	
Mutex Kullanımı	□	
Testler		
Araçlarla Güvenlik Testleri	□	
Birim Test	□	
Kritik Güvenlik Özelliklerinin Görselleştirmesi ve Belgelenmesi		
Görsel Olarak İnceleme	□	
Belgeleme	□	

1.8. BLOKZİNCİR ORTAMINDA YAZILIM GELİŞTİRMEDE SIKINTILAR VE FIRSATLAR

Yeni gelişen bu teknoloji için yazılım geliştirme süreçlerinde sıkıntılar yaşandığının farkında olmak gerekir. Blokzinciri ortamında yazılım geliştirmede karşılaştıkları ve gözlemledikleri sıkıntıları MSKÜ Blokzincir Araştırma Grubu (MSKÜ BcRG) üyelerine sorarak, bu konuda gördükleri fırsatları da iletmelerini istedik. Bu kısımda; sekiz üyeden aldığımız yanıtları özetleyerek çıkarımlarda bulunacağız.

Çıkarımlar aşağıdaki gibidir:

- Öncelikle blokzinciri konusunda kaynaklar yetersizdir.
- Internet üzerindeki farklı bilgilerin hangisinin doğru ve güncel olduğu da çok net değildir.
- Dökümanlardaki örnek kodların da oldukça temel düzeyde olduğunu gözlemlenmiştir.
- En iyi uygulama (best practice) örnekleri yok denecek kadar az, örnek projeler de yetersiz, özellikle bitmiş proje bulmak zordur.
- Bazı kaynaklar da oldukça pahalı olabilmektedir. Ucunda finansal getiri olma olasılığı, bu işin ticaretine yol açabilmektedir.
- Blokzincir teknolojisine dair danışılabilecek bilgili kişiler azdır.
- Geliştirici topluluğu yeterince gelişmiş değildir.
- Gelişmiş bir blokzincir ekosisteminden söz etmek ise şu anda mümkün değildir.
- Blokzincir teknolojisinde standartlar oturmadığı için, sürekli tekrardan bir şeyleri öğrenmek gerekmektedir. Örneğin web teknolojilerinde API geliştirirken sadece o yeni platformu öğrenmek yeterli olurken, blokzincir ortamlarında ise yeniden öğrenmek gerekmektedir.
- Test araçları yetersiz; kurulum ve kullanımları da çoğunlukla kolay değildir.

Bu gözlemlediğimiz problemleri çözmek için araştırma, geliştirme ve billinçlendirmeye dayalı çalışmalara devam ediyoruz. Özellikle blokzincir araştırma ağlarının daha kolay kullanılabilir olmasının sağlanması gerektiğini düşünüyoruz. Sürdürülebilir araştırma ağları için birlikte çalışmaların yapılması gerek-

mektedir. Bu konularda daha çok akademik çalışmaya ve fon desteğine ihtiyaç duyulmaktadır. Bir blokzinciri ekosisteminin gelişmesi için; bu konuda çalışacak yazılım geliştirme ekiplerine yeterli ücret ve imkan sağlanması gerekmektedir. Fırsatlar ise birçok alanda devrim yaratacak olup değişiklikler ve getirileri olacaktır. Bazı kişilere ait görüşler de aşağıda verilmiştir.

- Katılımcı#1 şu anda blokzinciri sistemlerinde mahremiyet sağlanması üzerine yüksek lisans yapıyor ve kendisinin fırsatlara dahil yorumları: "Birçok farklı sektörde blokzinciri uygulamalarını görmeye başladık. Özel sektör uygulamalarının çoğu kapalı ve izinli blokzinciri yapıları üzerinde geliştirilmektedir. Ethereum gibi açık ve izinsiz yapılarda uygulama geliştirmenin maliyetinden ötürü, yakın zamanda kapalı ve izinli blokzinciri yapılarına olan ihtiyacın daha da artacağını düşünüyorum. Kripto paraya ihtiyaç duymayan, daha az enerji tüketen, kapalı ve izinli blokzinciri yapılarının önemi artacaktır. Bu şartlar altında akıllı sözleşme güvenliği daha fazla göz önüne gelecektir. Güvenli akıllı sözleşme geliştirmek uzmanlık gerektiren bir meziyettir. Bunun için akıllı sözleşmelerin çalışma mantığını, dağıtık sistemler ve determinizm kavramlarını iyi kavramış geliştiricilere ihtiyaç duyulacaktır. Akıllı sözleşme geliştiricisi olma konusunda önümüzde büyük fırsatların olduğunu rahatlıkla söyleyebilirim."
- Katılımcı#2 şu anda finans yazılımı geliştiren bir şirkette çalışıyor. Onun fırsatlara dair yorumları: "Kripto paraların ve kripto borsaların yükselişi, popülerleşmesi ile birlikte finans yazılımlarında daha çok kripto para, kripto borsa desteği görmeye başlayacağımızı düşünüyorum. Finans yazılımı geliştiren bazı şirketler; kripto borsalarda yapılan usulsüzlükleri bulmak için zincir incelemesi vb yazılımları geliştiriyorlar. Bunun gibi, blokzincir geliştiricisi olmaktan ziyade, işin finans tarafında, usulsüzlüklerin incelenmesi bulunması tarafında, daha fazla geliştiriciye ihtiyaç olabileceğini düşünüyorum."

1.9. SONUÇ VE DEĞERLENDİRMELER

Bu bölümle merkezi olmayan çözümlerinin nasıl güvenli ve güvenilir şekilde geliştirileceği konusunda bir ön bilgi verilmiştir. Sürekli gelişmekte olan blokzincir dünyasına dair güncel bilgiler sunulmuştur. Akıllı sözleşmelerde güvenlik süreçlerine ve gas optimizasyonuna dikkat edilmelidir. Blokzincir

tabanlı sistemlerin yazılım geliştirme süreçlerinde yaşanan ve aşılması gereken sorunlar konusunda da çözüm önerileri paylaşılmıştır. Bölümde verilen örneklerin en güncel haline SC-SecTesting Github reposundan (https://github.com/MSKU-BcRG/SC-SecTesting), çizimlere de BlockchainDiagrams Github reposundan (https://github.com/MSKU-BcRG/BlockchainDiagrams) ulaşılabilir.

Yönetilebilir ağ servisleri kullanımı; akıllı sözleşme geliştirilmesi, blokzincir sistemine yüklenmesi ve test edilmesi konusunda hem süreçleri kolaylaştıracak hem de hızlandıracaktır. Geliştirmekte olduğumuz ChainEx ve GoHammer yazılımlarının bu anlamda bir fark yaratacağını düşünüyoruz. Bu yazılımlar halen birçok ArGe projesinde ve MSKÜ Bilgisayar Mühendisliği Bölümündeki eğitim ve akademik araştırmalarda kullanılmaktadır. Akıllı sözleşmelerin güvenlik testleri konusunda çalışmalara devam edilmektedir.

Blokzinciri alanında yetişmiş uzman ihtiyacı bulunmaktadır. Birçok ülke bu konuda çalışmalarda bulunmaktadır. Çin devleti kalkınma planlarına bu konuda eğitim ve geliştirmeleri eklemektedir. Kobilerin düşük maliyetlerle geliştirme yapabilmeleri için BSN (Blockchain Service Network - https://bsnbase.io/) blokzincir ağının kurulmasına destek verilmektedir[24]. Ülkemizde ise Tübitak Bağ ve DS4H blokzincir ağları gelişmekte olan yapılardır. Blokzincir ağ test ortamlarının güvenilir ve sürdürülebilir olması sağlanmalıdır. Bu kapsamda DS4H Blokzincir ağı geliştirilmeye devam edilmektedir.

İnsanoğlunun kararları önyargılıdır; herhangi bir süreç, dahil olan insana, o insanın bulunduğu duygu durumuna ve zamana göre farklı işleyebilir. Kurallara göre işleyecek deterministik bir dünya için akıllı sözleşmelerin daha yaygın kullanımı gerekecektir. Yapay zeka ile akıllı sözleşmelerin entegrasyonu, blokzincir yapılarının güvenliği alanında birçok potansiyel çalışma konusu bulunmaktadır. Blokzincir tabanlı sistemlerde ölçeklenebilirlik (scalability) ve başarım açısından çözülmesi gereken birçok problem ve fırsat (challenge) bulunmaktadır. Veri bilimi açısından bu sorunları ve çözüm önerileri [12] nolu çalışmada sunulmuştur. Blokzincir sistemlerini bir alternatif çözümden daha çok tamamlayıcı bir çözüm olarak görmek daha yerinde olacaktır. Blokzincir sistemlerinin var olan çözümlere, bulut yapılarına ve IPFS (Interplanetary File System) gibi dağıtık dosya sistemlerine entegrasyonu ile karma

24 Inside China's Effort to Create a Blockchain It Can Control, https://www.coindesk.com/china-to-create-it-can-control

sistemler kurulabilir. Bu süreçlerde; ölçeklenebilirlik, birlikte çalışabilirlik ve mahremiyet için önerdiğimiz MPISA (Multi Platform Interoperable Scalable Architecture - Çoklu Platform Birlikte Çalışabilir Ölçeklenebilir Mimari) [12] modeline benzer yapıların kullanılması faydalı olacaktır. Blokzincir sistemlerinde mahremiyetin sağlanması, kuantum sonrası kriptografi (post-quantum cryptography) kullanan blokzincir sistemleri [35] üzerinde de çalışmalar yapılması gereklidir.

Teşekkür

Yorumlarıyla bölümün şekillendirilmesine yardımcı olan Dr. Senem Yazıcı Yılmaz'a teşekkürler. MSKÜ Blokzincir Araştırma Grubu'ndan Cemal Dak'a çizimlerdeki katkıları, Ayça Öksüztepe'ye yazım kontrolündeki katkıları; Emre Ertürk, Murat Doğan ve Ümit Kadiroğlu'na ekran çıktılarındaki katkıları için teşekkür ederiz.

KAYNAKLAR

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," Whitepaper. 2008.

[2] QYResearch, "Smart Contracts Market", Valuates Reports, 2020, [Online]. Available: https://reports.valuates.com/market-reports/QYRE-Auto-31L1599/global-smart-contracts

[3] N. Atzei, M. Bartoletti and T. Cimoli, "A survey of attacks on ethereum smart contracts (sok)," In: International conference on principles of security and trust. Springer, Berlin, Heidelberg, 2017. p. 164-186.

[4] V. Dhillon, D. Metcalf, M. Hooper, "The DAO hacked," In: Blockchain Enabled Applications. Apress, Berkeley, CA, 2017. p. 67-78.

[5] E. Işık, M. Birim and E. Karaarslan, "*Merkezi Olmayan Uygulamalar için Geliştirme ve Test Ortamı: Chainex,*" UYMS 2021, 2021.

[6] M. Birim, H. E. Ari and E. Karaarslan, "GoHammer Blockchain Performance Test Tool," *Journal of Emerging Computer Technologies (JECT)*, 2021, 1.2: pp. 31-33

[7] E. Karaarslan and E. Konacaklı, "Decentralized solutions for data collection and privacy in healthcare". In: *Artificial Intelligence for Data-Driven Medical Diagnosis*. De Gruyter, 2021. p. 167-190.

[8] A. M. Antonopoulos and G. Wood, "Mastering ethereum: building smart contracts and dapps," O'reilly Media, 2018.

[9] C. Cachin and M. Vukolić, "Blockchain consensus protocols in the wild," arXiv preprint arXiv:1707.01873, 2017.

[10] F. Vogelsteller and V. Buterin, "Ethereum whitepaper," Ethereum Foundation, 2014.

[11] K. Wüst and A. Gervais, "Do you need a blockchain?". Presented at 2018 Crypto Valley Conference on Blockchain Technology (CVCBT). IEEE, 2018. p. 45-54.

[12] E. Karaarslan and E. Konacaklı, "Data Storage in the Decentralized World: Blockchain and Derivatives" in "Who Run The World:DATA", 1st ed. Istanbul, Turkey, Istanbul University Press, 2020, ch.3, pp. 37-69. [Online]. Available: https://iupress.istanbul.edu.tr/tr/book/who-runs-the-world-data/chapter/data-storage-in-the-decentralized-world-blockchain-and-derivatives

[13] Z. Durğay and E. Karaarslan, "Blokzinciri Teknolojisinin E-Devlet Uygulamalarında Kullanımı: Ön İnceleme," *Akademik Bilişim Konferansı, Karabük*, 2018.

[14] T. Jensen, J. Hedman and S. Henningsson, "How TradeLens Delivers Business Value With Blockchain Technology," MIS Quarterly Executive, 2019, 18.4.

[15] O. Goldreich and Y. Oren, "Definitions and properties of zero-knowledge proof systems," *Journal of Cryptology*, 1994, 7.1: 1-32.

[16] C. Li, P. Li, D. Zhou, Z. Yang, M. Wu, G. Yang, ... and A .C. C. Yao, "A decentralized blockchain with high throughput and fast confirmation," Presented at *2020 {USENIX} Annual Technical Conference ({USENIX}{ATC} 20)*. 2020. pp. 515-528.

[17] A. Zmudzinski, " Hyperledger Fabric Sees More Dev Activity Than Corda in Q3 2019: Report," CoinTelegraph, 2020.

[18] N. Szabo, "Formalizing and securing relationships on public networks," *First Monday*, 1997.

[19] K. Chatterjee, A. K. Goharshady and A. Pourdamghani "Probabilistic smart contracts: Secure randomness on the blockchain," Presented at 2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). IEEE, 2019. p. 403-412.

[20] L. Marchesi, M. Marchesi, L. Pompianu and R. Tonelli, "Security checklists for ethereum smart contract development: patterns and best practices," arXiv preprint arXiv:2008.04761, 2020.

[21] Z. Wang, H. Jin, W. Dai, K. K. R. Choo, and D. Zou, "Ethereum smart contract security research: survey and future research opportunities," *Frontiers of Computer Science*, 2021, 15.2: pp. 1-18.

[22] S. Sayeed, H. Marco-Gisbert and T. Caira, "Smart contract: Attacks and protections," IEEE Access, 2020, 8: pp. 24416-24427.

[23] B. Prasad, "Vulnerabilities and Attacks on Smart Contracts over BlockChain," Turkish Journal of Computer and Mathematics Education (TURCOMAT), 2021, 12.11: pp 5436-5449.

[24] E. Karaarslan, M. Birim and H. E. Ari, "*Forming a Decentralized Research Network: DS4H*", Turkish Journal of Electrical Engineering & Computer Sciences, hakem değerlendirmesinde, 2021

[25] C. Wickboldt, "*Benchmarking a blockchain-based certification storage system*," (No. 2019/5). Diskussionsbeiträge, 2019.

[26] M. Kuzlu, M. Pipattanasomporn, L. Gurses, and S. Rahman, "Performance analysis of a hyperledger fabric blockchain framework: throughput, latency and scalability," Presented at *2019 IEEE international conference on blockchain (Blockchain)*. IEEE, 2019. po. 536-540.

[27] S. Pongnumkul, C. Siripanpornchana and S. Thajchayapong, "Performance analysis of private blockchain platforms in varying workloads," Presented at *2017 26th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2017. pp. 1-6.

[28] Q. Nasir, I. A. Qasse, M. Abu Talib and A. B. Nassif, "Performance analysis of hyperledger fabric platforms," *Security and Communication Networks, vol. 2018, Article ID 3976093, 14 pages, 2018. https://doi.org/10.1155/2018/3976093*

[29] J. Feist, G. Grieco and A. Groce, "Slither: a static analysis framework for smart contracts," Presented at *2019 IEEE/ACM 2nd International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB)*. IEEE, 2019. pp. 8-15.

[30] G. Grieco, W. Song, A. Cygan, J. Feist and A. Groce, "Echidna: effective, usable, and fastW. fuzzing for smart contracts," In: *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*. 2020. pp. 557-560.

[31] M. Mossberg, F. Manzano, E. Hennenfent, A. Groce, G. Grieco, J. Feist, ... and A. Dinaburg, "Manticore: A user-friendly symbolic execution framework for binaries and smart contracts," Presented at *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 2019. pp. 1186-1189.

[32] T. Durieux, J. F. Ferreira, R. Abreu and P. Cruz, "Empirical review of automated analysis tools on 47,587 ethereum smart contracts," In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. 2020. pp. 530-541, https://doi.org/10.1145/3377811.3380364.

[33] "Kişisel Verileri Koruma Kanunu, Kanun numarası: 6698", *Resmi Gazete Sayı*, pp. 29677, 2016.

[34] "GDPR", Official Journal of the European Union, vol. L119, pp. 1-88, April 2016.

[35] T. M. Fernández-Carames and P. Fraga-Lamas, "Towards post-quantum blockchain: A review on blockchain cryptography resistant to quantum computing attacks," IEEE access, 2020, 8: 21091-21116.

Bölüm 2

BLOKZİNCİRİNDE UZLAŞI MEKANİZMALARI

Murat Osmanoğlu

Blokzinciri belirli bir ağ üzerinden birbiriyle iletişim kuran birden fazla kullanıcının koordineli bir şekilde idame ettirdiği bir dağıtık defter sistemidir. Blokzincirinde her kullanıcı ilgili defterin bir kopyasını tutmakta ve ağ üzerinden paylaşılan işlemleri üzerinde mutabık kalınan dağıtık bir protokol üzerinden bloklar halinde deftere eklemektedir. Burada dikkat edilmesi gereken husus kullanıcıların protokolün herhangi bir anında ağla irtibatlarının kopabileceği ve paylaşılan mesajları iletmek veya yeni mesajlar oluşturmak gibi protokol için gerekli bir takım aktiviteleri gerçekleştiremeyebileceği gerçeğidir. Dahası kötü niyetli bazı kullanıcılar protokolün doğru şekilde çalışmasını engellemek adına bilinçli olarak protokolden sapabilir ve tahrif edilmiş mesajlar paylaşabilirler. Bu tarz kullanıcıların varlığı, özellikle kullanıcıların takma adlarla gerçek kimliklerini gizleyebildikleri izinsiz blokzinciri gibi güvenin zor tesis edilebileceği platformlarda etkili ve güvenli protokol geliştirmeyi iyice zorlaştıracaktır. Bu kitap bölümünde, zikredilen protokol 'uzlaşı mekanizması' ya da 'uzlaşı protokolü' şeklinde adlandırılıp blokzincirin, performansını ve güvenliğini doğrudan etkileyen en önemli bileşenlerden birisi olarak incelenmiştir. Blokları üretecek liderlerin seçiminde ya da liderler tarafından üretilmiş blokların onaylanmasında kullanılan yöntemler baz alınarak belirlenen çekiliş tabanlı, oylama tabanlı ve her ikisinin harmanlandığı hibrit başlıkları altında blokzincirinde kullanılan uzlaşı mekanizmaları araştırılmış, sağladığı üstünlükler ve karşılaşılabilecek riskler irdelenmiş ve ülke açısından konunun önemi sonuçta değerlendirilmiştir.

2.1. DAĞITIK SİSTEMLERDE UZLAŞI MEKANİZMALARI

Burada genel anlamda uzlaşı mekanizmalarındaki gereksinimlerin daha iyi kavranabilmesi adına, blokzincirindeki benzer protokoller için altyapı olarak değerlendirilebilecek dağıtık sistemlerdeki uzlaşı protokolleri incelenecektir.

Dağıtık sistemler iş gücünün tek bir merkez yerine iki ya da daha çok sistemin arasında paylaştırıldığı yapılardır. Bu sistemler koordineli bir biçimde hareket ederek tek bir sunucu – istemci mimarisi üzerinden yüklenilebilecek görevleri gerçekleştirmeye çalışırlar. Yukarıda da belirtildiği gibi, bu sistemlerden bazılarındaki geçici çökmeler veya iletişim ağında ortaya çıkabilecek aksaklıklar dolayısıyla sistemler arasındaki koordinasyonda aksamalar olabilir. Ya da sistemlerden bazısı bilinçli olarak koordinasyonu baltalamak isteyebilir. Bahsedilen bu gibi durumları tolere ederek sistemlerin koordineli bir biçimde fikir birliğine varabilmelerine olanak sağlayan uzlaşı mekanizmaları dağıtık sistemler alanında uzun yıllardır çalışılan bir konudur.

Temel olarak bir uzlaşı mekanizmasının aşağıda belirtilen dört şartı sağlaması beklenir [1]:

– Bütünlük (integrity); herhangi bir mesaj hatalı olmayan düğümler[1] tarafından en çok bir defa iletilir.

– Söz birliği (agreement); herhangi bir mesaj hatalı olmayan düğümlerden biri tarafından iletilmişse, nihayetinde bütün hatalı olmayan düğümler tarafından iletilecektir.

– Geçerlilik (validity); hatalı olmayan düğümler tarafından iletilen bir mesaj, nihayetinde uzlaşının bir parçası olacaktır.

– Tam sıralama (total order); m_1 ve m_2 , P ve Q hatalı olmayan düğümleri tarafından iletilen iki mesaj olsun. Ancak ve ancak P m_1 mesajını m_2'den önce iletmişse, Q m_1 mesajını m_2'den önce iletmiştir.

Kimi çalışmalarda [2], [3] tam sıralama ve bütünlük özellikleri güvenlik (safety) başlığı altında, söz birliği ve geçerlilik özellikleri de canlılık (liveness) başlığı altında değerlendirilmiştir. Burada güvenlik ve canlılık özellikleri sırasıyla, hatalı olmayan düğümlerin iletilen mesajların sıralaması üzerinde muta-

1 Bölümde uzlaşı mekanizmasına katılan her bir sistem ya da kullanıcı düğüm olarak adlandırılacaktır.

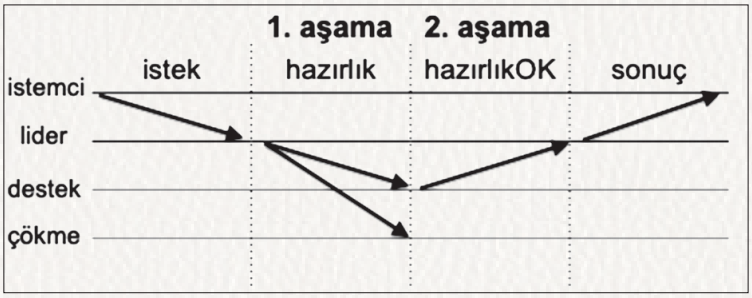
bık kalması ve hatalı olmayan düğümlerden biri tarafından iletilen bir mesajın nihayetinde uzlaşının bir parçası olması olarak özetlenebilir.

Dwork vd. [4] düğümler arası senkronizasyona bağlı olarak eşzamanlı, eşzamansız ve kısmi eşzamanlı olmak üzere 3 farklı ağ modeli önermişlerdir. Eşzamanlı ağlarda mesajın bir düğümden başka bir düğüme iletilmesi için gerekli olan süreye bir üst sınır belirlenirken, eşsamansız ağlarda böyle bir üst sınır tayin edilmemektedir. Kısmi eşzamanlı ağlar ise eşzamansız ağlar gibi mesajların gecikmesine imkan tanısa da nihayetinde bütün mesajların belirli bir süre içerisinde iletilmesini sağlamaktadırlar. Eşzamanlı ağlardan farklı olarak, kısmi eşzamanlılarda mesajların iletilmesi için gerekli süreye bir üst sınır tayin edilmiş olsa da, bu sınır uzlaşı mekanizmasının başında belirlenmemiştir. Burada incelenen uzlaşı mekanizmalarının (bir kaç istisna hariç) kısmi eşzamanlı ağ modelinde çalıştığı kabul edilecektir.

Dağıtık sistemlerdeki uzlaşı protokolleri hatalı düğümün davranışına göre çökme (crash) hata toleransı ve Bizans hata toleransı olmak üzere iki başlık altında incelenmektedir. Çökme hata toleransı modelinde hatalı düğümler donanımla ya da yazılımla ilgili arızalara, ya da ağdaki bir takım bağlantı kopukluklarına bağlı olarak protokolün gerektirdiği işleri aksatabilir ya da yapmayabilirler. Diğer yandan Bizans hata toleransı modelinde ise hatalı düğümler, yukarıda zikredilenlerin yanı sıra, bilinçli olarak ya da yazılımsal bazı hatalardan kaynaklı protokolden sapabilir, hatta protokolü baltalamak adına diğer düğümleri yanlış yönlendirebilirler. Bu kısımda, çökme hatalarını tolere edebilen Viewstamped Replication (VR) ve Raft protokolleri ile Bizans hatalarını da tolere edebilen PBHT ve FBHT uzlaşı protokolleri incelenecektir.

2.1.1. Viewstamped Replication (VR) Protokolü

VR uzlaşı protokolü Liskov ve Cowling [5] tarafından bir çoğaltılmış durum makinesi (replicated state machine) servisi olarak tasarlanmıştır. Protokole geçmeden önce, çoğaltılmış durum makinesini açıklamak daha faydalı olacaktır. Çoğaltılmış durum makinesi [6] iletişim halindeki düğümlerde kopyaları bulunan, durum değişkenlerinden ve durumu güncellemeye olanak veren fonksiyonlardan oluşan bir durum makinesidir. İstemciler belirli fonksiyonlar yoluyla güncelleme talep ettiklerinde, düğümler tek bir sunucu gibi hareket ederek deterministik bir şekilde fonksiyonları uygular ve sonucu istemciye dönerler.



Şekil 2.1. Viewstamped Protokolü

VR kısmi eşzamanlı ağ modeli için geliştirilmiş bir protokoldür. f hatalı düğüm sayısı olmak üzere, protokolün sorunsuz çalışabilmesi için toplam düğüm sayısı $2f+1$ den fazla olmalıdır. VR protokolünde lider düğüm ve destek düğüm olmak üzere iki farklı düğüm vardır. Lider, atandığı zaman aralığında istemcilerden gelen istekleri sıraya koyan ve diğer düğümleri yönlendirerek isteğin karşılanmasını sağlayan düğümken; destek ise protokol esnasında liderin belirlediği sıraya uygun istemcilerden gelen istekleri karşılayan ve protokol esnasında lidere destek olmaya çalışan düğümdür. Protokol zamanı pencere adı verilen parçalara bölmekte ve her bir parçaya IP adreslerini baz alarak küçükten büyüğe doğru sırasıyla düğümleri lider olarak atamaktadır.

İstemciler isteklerini, *op* gerçekleştirilmesini istedikleri operasyon, *c* istemcinin kimliği ve *s* istek numarası olmak üzere, *(istek,op,c,s)* şeklindeki bir mesajla ilgili pencerenin liderine gönderirler. Lider düğüm istemciden böyle bir istek geldiğinde, önce istek numarasının daha önce kendisine ulaşmış önceki isteklerin numaralarından büyük olup olmadığına bakar. Eğer *s* daha büyükse isteği log kayıtlarına ekleyerek Şekil 2.1'de görüldüğü gibi hazırlık ve hazırlıkOK olmak üzere iki aşamadan oluşan VR protokolünü başlatır.

Hazırlık aşamasında lider destek düğümlerine, *v* liderin atandığı pencere numarası, *m* istemcinin isteği, *n* ve *k* sırasıyla isteğe atanmış sıradaki işlem ve taahhüt numaraları olmak üzere *(hazırlık,v,m,n,k)* şeklinde bir mesaj gönderir. HazırlıkOK aşamasında ise destek düğümler önce kendilerine gelen hazırlık mesajının doğruluğunu kendi kütük (log) kayıtlarından kontrol ederler. Eğer kütük kayıtlarındaki en son işlem numarası $n-1$ değilse isteği reddederler. Aksi halde hazırlık mesajını kütük kayıtlarına ekleyerek *i* ilgili destek düğümünün sıra numarası olmak üzere *(hazırlık,OK,v,n,i)* şeklinde bir mesajı

lider düğüme gönderirler. Eğer lider düğüm f tane farklı destek düğümden hazırlıkOK mesajı almışsa, ilgili isteği taahhütlü kabul eder ve istenen operasyonu gerçekleştirerek sonuç değeri x'i $(yanıt,v,s,x)$ mesajıyla istemciye iletir.

VR protokolü lider düğüm hatalı davrandığında canlılığı ve güvenliği koruyabilmek adına lider değişimine izin vermektedir. Destek düğümler eğer uzun süre lider düğümden mesaj almazlarsa diğer destek düğümlerine mesajla durumu bildirip lider değişimini talep ederler. Eğer $f + 1$ destek düğümü değişimi onaylarsa, bir sonraki pencerenin lideri yeni bir mesajla diğer düğümleri bilgilendirerek yeni pencereyi başlatır.

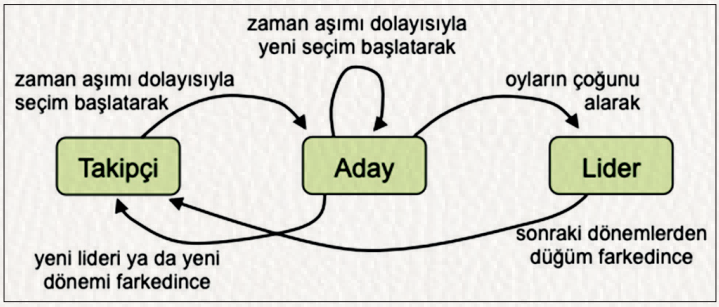
Protokolün Değerlendirilmesi. Yukarıda bahsi geçen güvenlik özelliği VR protokolü için, bütün hatalı olmayan düğümlerin isteklere atanan sıra numaraları üzerinde mutabık kalması olarak uyarlanabilir. Protokolde lider kendisine diğer düğümlerden ancak f tane $(hazırlık,OK,v,n,i)$ mesajı geldiğinde ilgili talebin n sıra numarasıyla kütüğe kaydedildiğini kabul edip, sıradaki k numarasını ilgili talebe taahhüt numarası olarak atar ve operasyonu gerçekleştirip sonucu istemciye döner. Bu esnada bazı düğümler çökseler bile, sistemde en fazla f tane hatalı düğüm bulunduğu ve oyuncuların sayısının $2f$'den büyük olduğu kabul edildiğinden talep farklı seri numarasıyla kütüğe kaydedilmeyecektir. Diğer yandan lider hatalı olduğunda protokol hatalı olmayan düğümlerin bunu fark edip lideri değiştirebilmelerine imkan verdiği için, bu şekilde sistemin canlılığı da muhafaza edilebilecektir. Sonuç olarak protokol, n toplam düğüm sayısı olmak üzere hatalı düğümlerin sayısının $\lfloor(n-1)/2\rfloor$'den fazla olmadığı kabul edildiğinde güvenlik ve canlılık özelliklerini sağlamaktadır. Ayrıca protokolün mesaj karmaşıklığı asimptotik olarak $O(n)$'dir. Bu arada protokol Bizans hatalarını tolere edememektedir.

2.1.2. Raft Protokolü

Raft protokolü, Ongaro ve Ousterhout [7] tarafından bir çoğaltılmış durum makinesi hizmeti olarak kısmi eş zamanlı ağ modeli için geliştirilmiştir. Protokol, n toplam düğüm sayısı olmak üzere, VR gibi en fazla $\lfloor(n-1)/2\rfloor$ çökme hatalı düğümü tolere edebilmektedir. Şekil 2.2'de görüldüğü gibi protokolde düğümler lider, takipçi ve aday olmak üzere üç rolden birisini üstlenirler. Lider sorumlu olduğu zaman diliminde istemcilerden gelen talepleri toplar ve taleplerin karşılanması ve sonuçlandırılması sürecini yönetir. Raft protokolünde kopya sunucular birbirleriyle 'girdi ekleme' (AppendEntries)

ve 'oylama talebi' (RequestVote) olmak üzere iki tür uzaktan yordam çağrısı (UYÇ - remote procedure call) kullanarak iletişim kurmaktadırlar.

VR protokolündekine benzer Raft'ta da zaman dönem adı verilen parçalara ayrılmıştır. Fakat VR'dan farklı olarak, her dönemin lideri dönemin başlangıcında gerçekleştirilen bir seçim yoluyla belirlenmektedir. Seçim sürecine tüm düğümler başlangıçta takipçi olarak katılırlar. Seçim sürecinde lider olmak isteyen takipçiler diğer düğümlere oylama talebi çağrısı gönderirler[2] ve aday rolüne geçerler. Bu noktada diğer düğümlerin çoğunun onayını alan adaylar, ilgili zaman döneminin lideri olur ve dönem boyuncu liderliğini sürdürebilmek için takipçilere periyodik olarak herhangi bir kütük kaydı içermeyen girdi ekleme çağrısı gönderirler. Eğer herhangi bir aday ilgili dönem içerisinde oyların çoğunu toplayamamışsa o dönem lidersiz tamamlanır ve takipçilerden birinden gelecek bir oylama talebi çağrısı ile yeni döneme girilir. Bu durum, genellikle takipçilerin başkasının liderliğini kabul etmeyip kendilerini lider olarak öne süren yeni oylama talebi çağrısı oluşturmaları ve böylece oyların birçok aday arasında bölünmesi neticesinde ortaya çıkmaktadır. Bu durumu gidermek için Raft protokolünde rastgele hale getirilmiş bir lider seçimi yürütülmektedir. Takipçiler $[150ms, 300ms]$ aralığından rastgele olarak bir mola süresi seçerler, ve ancak bu süre sonunda bir sonraki dönemin lider seçimini başlatabilirler. Rastgelelik sayesinde mola süreleri farklı olacağından, içlerinden biri diğerlerinden önce yeni dönemin lider seçim sürecini başlatabilecek ve diğerlerinin mola süresi henüz bitmeden oylama talebi çağrısını gönderebilecektir.



Şekil 2.2. Raft Protokolünde Düğümlerin Üstlendiği Roller ve Roller Arası Geçişler.

2 Oylama talebi çağrısı adayın kütük indeksini (son kütük kaydının numarası) de içermektedir. Eğer takipçinin kütük indeksi adayın kütük indeksinden daha güncelse, takipçi oylama talebi çağrısı reddedecektir.

Lider olarak seçildikten sonra düğümler, kendi dönemlerinde istemcilerden gelen talepleri karşılamak ve sonuçlandırmakla yükümlüdürler. Lider, gelen talepleri önce kütük kayıtlarındaki sırasını belirtecek bir tamsayı ve lideri olduğu dönem numarasıyla beraber kütük kayıtlarına ekler. Sonra bu talepleri, ilgili dönem numarasını ve sıra numarasını içeren girdi ekleme çağrısı yoluyla diğer düğümlerle paylaşır. Girdi ekleme çağrısını aldıktan sonra düğümler ilgili talepleri çağrıda belirtilen dönem numarası ve sıra numarası ile birlikte kendi kütük kayıtlarına ekler ve bu durumu lidere bildirirler. Düğümlerin çoğundan aynı bildiri mesajı döndüğünde (bu bildiri de girdi ekleme çağrısı yoluyla yapılmaktadır), lider bu girdiyi taahhüt edilmiş kabul eder ve bu taahhüdü kütük kayıtlarına ekler. Sonrasında lider talebi çoğaltılmış durum makinesine uygular ve sonucu istemciye döner. Bu süreç esnasında bazı düğümlerin çökebileceği ya da ağda bazı paketlerin kaybolabileceği gerçeğinden hareketle lider, istemciye sonucu ilettikten sonra bile bütün düğümlerin aynı kütük kayıtlarına sahip olabilmeleri için ilgili taleple alakalı girdi ekleme çağrısı yapmaya devam eder.

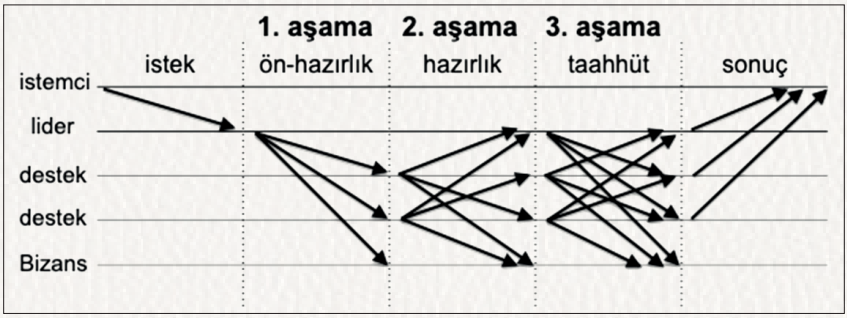
Taleplerin dışında liderler takipçilerin kütük kayıtlarında oluşabilecek yanlışlıkları fark edip düzeltebilmelerine imkan verebilmek için girdi ekleme çağrısı yapmaya devam ederler. Bu noktada liderler her bir takipçi için ayrı bir kütük indeksi tutarlar. Eğer takipçiler girdi ekleme çağrılarına olumsuz yanıt bildirirlerse, liderler takipçilere ait kütük indekslerini bir azaltırlar. Bu ayarlama takipçilerle liderin indeksleri uyuşana kadar devam eder.

Protokolün Değerlendirilmesi. Raft protokolünün seçim sürecinde, takipçilerin son kütük kaydı adayın son kütük kaydından sonraki döneme aitse, ya da son kütük kayıtları aynı döneme ait olsa bile takipçinin son kütük kayıt numarası adayınkinden daha büyükse, takipçi adayın oylama talebi çağrısını reddeder. Böylelikle çoğunluğun gerisinde kalmış olanların lider olarak seçilebilmesinin önüne geçilmektedir. Dahası bu yolla taahhüt edilmiş taleplerin, seçilen her yeni liderin kütük kayıtlarında bulunuyor olması da garantilenmektedir. Ayrıca ancak çoğunluk tarafından kütük kayıtlarına eklenmiş taleplere taahhütte bulunulduğu göz önünde bulundurulduğunda ve çökme hatalı düğümlerin azınlık olduğu kabul edildiğinde, Raft protokolü güvenlik ve canlılık özelliklerini sağlamaktadır.

2.1.3. Pratik Bizans Hata Toleransı (PBHT) Protokolü

Pratik Bizans Hata Toleransı (PBHT) protokolü, 1999 yılında Castro ve Liskov [2] tarafından VR protokolüne benzer bir çoğaltılmış durum makinesi servisi olarak kısmi eşzamanlı ağ modeli için geliştirilmiştir. Ama VR'dan farklı olarak protokol çökme hatalarının yanı sıra Bizans hatalarını da tolere edebilmektedir. Yalnız burada protokolün sorunsuz çalışabilmesi için f hatalı düğüm sayısı olmak üzere toplam düğüm sayısının $3f$+1'den fazla olması gerekmektedir. Ayrıca protokol bozulmuş mesajları tespit edebilmek, tekrarlama (replay) ve sahtecilik (spoofing) saldırılarını engelleyebilmek amacıyla dijital imza algoritması, çakışmaya dirençli özet fonksiyonu gibi bir takım kriptografik teknikler de kullanmaktadır. PBHT protokolünde her bir düğüm sırasıyla doğal sayılarla kimliklendirilir. Ayrıca VR protokolüne benzer biçimde protokol pencere adı verilen zaman parçacıkları üzerinden çalıştırılır. R protokoldeki toplam düğüm sayısı ve v pencere sırası olmak üzere, $p = v \bmod R$ şartını sağlayan p düğümü v penceresinin lider düğümü, diğerleri de destek düğümü olarak belirlenir.

PBHT protokolünde o gerçekleştirilmesi istenen operasyon, c istemcinin kimliği ve t zaman damgası olmak üzere, istemciler taleplerini imzalı $(istek,o,t,c)$ mesajı yoluyla ilgili pencerenin lider düğümüne gönderirler. Lider gelen isteğin otantisitesini dijital imzanın doğruluğunu kontrol ederek tetkik ettikten sonra; Şekil 2.3'te görüldüğü gibi ön-hazırlık, hazırlık ve taahhüt şeklinde üç aşamalı PBHT protokolünü başlatır. Ön-hazırlık aşamasında lider düğüm istemcinin talebine sıradaki sıra numarası n'yi atar ve talebi kütük kayıtlarına ekler. Sonrasında v o anki pencere numarası, m istemcinin talebi ve d talebin özet değeri olmak üzere imzalı $\left\langle \langle \text{ön} - hazırlık, v, n, d \rangle_{\sigma_p}, m \right\rangle$ mesajını tüm destek düğümlerine gönderir. Burada dijital imzanın boyutunu küçük tutmak için talebin yerine özet değerinin imzası alınmıştır. Ön-hazırlık mesajını aldıktan sonra destek düğümü, önce d'nin m talebinin gerçekten özet değeri olup olmadığını ve dijital imzanın doğru olup olmadığını kontrol eder. Sonrasında kütük kayıtlarından sıra numarası n'nin daha önce başka bir talep için kullanılıp kullanılmadığına bakar. Eğer mesaj bütün bu testleri geçerse destek düğümü hazırlık aşamasına geçerek, i destek düğümünün kimliği olmak üzere diğer düğümlere imzalı $\langle hazırlık, v, n, d, i \rangle_{\sigma_i}$ mesajını gönderir. Burada ilgili destek düğümi ayrıca iki mesajı da kütük kayıtlarına ekler.



Şekil 2.3. Pratik Bizans Hata Toleransı Protokolü

Taahhüt aşamasında destek düğümleri lider de dahil $2f$ farklı düğümden, ön-hazırlık mesajındaki n sıra numarasıyla ve birbirleriyle uyuşan hazırlık mesajı alırlarsa diğer düğümlere $\langle taahhüt, v, n, d, i\rangle_{\sigma_i}$ mesajını gönderirler. Bu noktada düğümlerin ellerinde hazırlık mesajındaki n sıra sumarasıyla uyuşan $2f$ farklı düğümden gelen taahhüt mesajı varsa, düğümler ilgili sıra numarasının gelen talebe taahhüt edildiğini kabul edip istemcinin talebini yerine getirirler ve sonucu t talebin zaman damgası ve r sonuç olmak üzere $\langle sonuç, v, t, c, i, r\rangle_{\sigma_i}$ mesajını doğrudan c istemcisine iletirler. Ayrıca düğümler bu iki mesajı da kütük kayıtlarına eklerler. Dijital imzaları doğrulanmış aynı t ve r değerlerine sahip f+1 sonuç mesajı istemcinin sonucu doğru kabul etmesi için yeterli olacaktır. Burada istemciler belli süre içerisinde eğer yeterli sonuç mesajı almazlarsa talebi bütün düğümlere göndererek aksiyon alınmasını sağlayabilirler.

PBHT protokolü, lider düğüm hatalı davrandığında canlılığı koruyabilmek ve sistemi sürdürülebilir kılmak için lider değişimine izin vermektedir. Bu doğrultuda destek düğümler bir kronometre yardımıyla lider düğümün hangi sıklıkla mesaj gönderdiğini kontrol ederler. Eğer uzun süre lider düğümden mesaj almazlarsa diğer destek düğümlerine mesajla durumu bildirip lider değişimini talep ederler. Eğer $2f$ + 1 destek düğümü bu durumu onaylarsa bir sonraki pencerenin lideri yeni bir mesajla diğer düğümleri bilgilendirerek yeni pencereyi başlatır.

Protokolün Değerlendirilmesi. Protokolde düğümler, kendilerine $2f$ + 1 tane $\langle hazırlık, v, n, d, i\rangle_{\sigma_i}$ hazırlık mesajı geldiğinde ancak d özet değerine sahip ilgili talebe taahhütte bulunur ve n seri numarasıyla onu kütük kayıtlarına

eklerler. İçlerinden bazı kötü niyetli düğümler protokol esnasında hatalı olmayan düğümleri yanıltmak adına aynı talep için farklı sıra numaralı mesajlar paylaşsalar bile, sistemde en fazla f tane hatalı düğüm bulunduğu kabul edildiğinden ve oyuncuların sayısı $3f$'den büyük olduğundan, bu mesajlar hatalı olmayan düğümler tarafından kabul edilmeyecek ve farklı seri numaralı taleplere taahhütte bulunulmayacaktır.

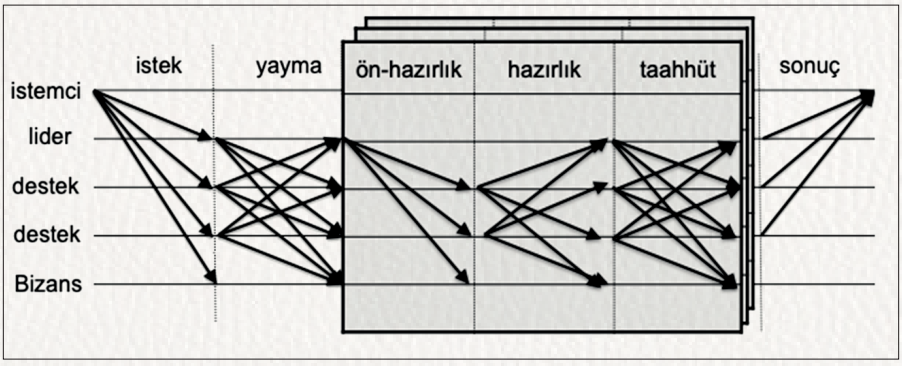
Diğer yandan protokol hatalı liderlerin sistemin canlılığını engellemesi adına hatalı olmayan düğümlere protokolü bir sonraki pencereye taşıma fırsatı da sunmaktadır. Yalnız hatalı olmayan düğümlerin lideri değiştirerek bir sonraki pencereye ilerleyebilmeleri için, bu talebin $2f + 1$ düğüm tarafından onaylanması gerekmektedir. Burda amaç kötücül düğümlerin sürekli lider değişimi talep ederek sistemi çalışılamaz hale getirmesinin önüne geçmektir. Ayrıca sistemde kötü niyetli düğümlerin sayısı en çok f olduğundan, bu tür düğümler koordineli hareket etseler bile sistemin sağlıklı bir şekilde çalışmasını en fazla f pencere için engelleyebileceklerdir. Sonuç olarak protokol, n toplam düğüm sayısı olmak üzere hatalı düğümlerin sayısının $\lfloor (n-1)/3 \rfloor$'den fazla olmadığı kabul edildiğinde güvenlik ve canlılık özelliklerini sağlamaktadır. Ayrıca protokolün mesaj karmaşıklığı asimptotik olarak $O(n^2)$'dir.

2.1.4. Fazlalık (Redundant) Bizans Hata Toleransı (FBHT) Protokolü

PBHT protokolü pratik bir Bizans hata toleransı sunuyor olsa da, kötücül düğümlerin koordineli bir şekilde hareket ederek belli bir zaman dilimi için bile olsa sistemi çalışamaz hale getirmesine ya da performansını düşürmesine imkan vermektedir. Bu zafiyetin en önemli sebebi, protokolün her bir pencere süresince istemciden gelen talepleri kabul etme, sıraya koyma ve yerine getirme noktasında ilgili pencerenin liderine bel bağlıyor olmasıdır. Bu bağımlılığı gidermek adına, Aublin vd. [8] Fazlalık Bizans Hata Toleransı (FBHT – Redundant Byzantine Fault Tolerance) adı verilen ve her bir pencere için hatalı lideri etkili bir şekilde fark edip değiştirebilen daha dirençli bir protokol önermişlerdir. FBHT protokolünde her bir pencere için önceden belirlenmiş liderlerin yürüttüğü protokolün tek bir icrası yerine, farklı düğümlerin lider olarak yönettiği çoklu icralar paralel olarak yürütülür. Ve protokol sonunda bu çoklu icralardan baş (master) icra diye adlandırılan en etkili olanı ağdaki düğümlerce geçerli icra olarak kabul edilir ve baş icrada taahhüt edilen talep

sıralaması kalıcı olarak kütük kayıtlarına eklenir. Protokol başında ilgili pencerenin liderinin yönettiği icra baş icra olarak kabul edilmektedir. Protokolde diğer icralar, verimlilik üzerinden baş icranın en etkili icra olup olmadığını tespit etmeye yarayan destek icraları olarak adlandırılır.

FBHT'de, PBHT protokolüne benzer olarak zaman pencere adı verilen parçalara bölünür ve her bir pencere için istemciden gelen talepleri kabul eden ve taleplerin karşılanması sürecini koordine eden bir lider belirlenir. Protokolde, PBHT protokolünden farklı olarak istemciler talepleri tüm düğümlere gönderirler. Düğümler gelen taleplerin geçerliliğini dijital imza algoritması ve mesaj doğrulama kodu (MDK) üzerinden tetkik ettikten sonra, yayma, ön-hazırlık, hazırlık ve taahhüt şeklinde dört aşamalı FBHT protokolünü başlatırlar. Yayma aşamasında düğümler istemcilerden gelen talepleri mesaj doğrulama kodu ekleyerek diğer düğümlere gönderirler. Düğümler herhangi bir talep için $f + 1$ yayma mesajı elde etmişlerse, ilgili talebi protokolün bundan sonraki aşamalarına taşınabilecek geçerli bir talep olarak değerlendirirler. Bu noktada alternatif lider olmak isteyen düğümler pencerenin lideri ile paralel olarak geçerli kabul ettikleri talepler için Şekil 2.4'deki gibi PBHT protokolüne benzer ön-hazırlık, hazırlık ve taahhüt aşamalarını yürütürler. FBHT protokolünde PBHT'a benzer olarak protokol esnasında talebin kendisi değil de özet değeri dolaşıma sokulmaktadır. Ama PBHT'dan farklı olarak dijital imza algoritması yerine doğrulama için mesaj doğrulama kodu kullanılmaktadır. Protokolün sonunda düğümler baş icra protokolü üzerinden taahhütte bulundukları talepleri yerine getirip sonucu ilgili istemcilere dönerler. PBHT protokolünde olduğu gibi farklı düğümlerden gönderilmiş birbiriyle uyumlu $f + 1$geçerli sonuç, istemcilerin ilgili sonucu kabul etmesi için yeterli olacaktır.



Şekil 2.4. Fazlalık Bizans Hata Toleransı Protokolü

Yukarıda da belirtildiği gibi FBHT protokolünde farklı düğümlerin yönettiği çoklu icralar paralel olarak yürütülmektedir. Herhangi bir pencere için lider yeterli performans göstermemişse, diğer bir ifadeyle başta kabul edilmiş baş icra en etkili icra değilse, düğümler icra değişim protokolü üzerinden baş icrayı değiştirirler. Biraz detaylandıracak olursak; protokolde her düğüm, paralel olarak yürüttüğü her bir icranın verimliliğini ölçmek için, her bir icrada taahhüt edilen toplam talep miktarını hesaplar. Eğer baş icranın verimliliğinin destek icralarının verimliliklerinin ortalamasına oranı belirli bir eşik değerinden küçükse, düğümler baş icra olmasını istedikleri icranın verimliliğini içeren bir icra değişim mesajı üzerinden icra değişim protokolünü başlatırlar. Eğer düğümler birbiriyle uyumlu ve geçerli $2f + 1$icra değişim mesajı elde ederlerse, mesaj üzerinden mutabık kaldıkları icrayı baş icra olarak kabul eder ve ilgili icradaki talepleri taahhüt edilmiş sayıp kütük kayıtlarına eklerler.

Protokolün Değerlendirilmesi. Protokolde baş icra PBHT protokolündekine benzer biçimde yürütüldüğü için, protokol esnasında bazı kötü niyetli düğümler hatalı olmayan düğümleri yanıltmak adına aynı talep için farklı sıra numaralı mesajlar paylaşsalar bile, sistemde f tane hatalı düğüm bulunduğundan ve oyuncuların sayısı $3f$'den büyük olduğundan bu mesajlar hatalı olmayan düğümler tarafından kabul edilmeyecek ve farklı seri numaralı taleplere taahhütte bulunulmayacaktır. Ayrıca FBHT protokolü birbirine paralel $f + 1$ farklı protokol icrası yürüttüğünden, liderler kötü niyetli olduklarında veya görece düşük performans sergilediklerinde diğer düğümlere hızlı ve etkin biçimde daha verimli olan icraya geçerek canlılığı muhafaza etme fırsatı da sunmaktadır. Sonuç olarak protokol, n toplam düğüm sayısı olmak üzere, hatalı düğümlerin sayısının $\lfloor(n-1)/3\rfloor$'den fazla olmadığı kabul edildiğinde güvenlik ve canlılık özelliklerini sağlamaktadır. Ayrıca protokolün mesaj karmaşıklığı asimptotik olarak $O(n^2)$'dir.

2.2. BLOKZİNCİRİNDE UZLAŞI MEKANİZMALARI

Bu başlık altında blokzinciri için önerilmiş ve literatürce kabul görmüş uzlaşı protokolleri incelenecektir. Protokoller, blokları üretecek liderlerin seçiminde ya da liderler tarafından üretilmiş blokların onaylanması esnasında kullanılan yöntemler referans alınarak oluşturulan üç farklı başlık - çekiliş tabanlı, oylama tabanlı ve hibrit uzlaşı protokolleri - altında değerlendirilecektir.

2.2.1. Çekiliş Tabanlı Uzlaşı Mekanizmaları

Çekiliş tabanlı uzlaşı mekanizmalarında liderler, blok üretiminde gösterdikleri çaba ya da blokzincir ağında sahip oldukları akçe miktarı gibi kriterler referans alınarak olasılıksal bir yolla seçilmektedir. Bunun yanı sıra, lider olarak seçilmiş düğümlerin ürettiği bloklar, geleneksel uzlaşı protokollerinden farklı olarak ekstra bir onay mekanizmasına ihtiyaç duymaksızın diğer düğümlerce geçerli kabul edilip zincire eklenmektedir. Bu başlık altında, lider seçiminde referans alınan kriter baz alınarak isimlendirilmiş üç farklı – emeğin kanıtı, hisse kanıtı ve alan kanıtı – uzlaşı mekanizması irdelenecektir.

2.2.1.1. Emeğin Kanıtı Tabanlı Uzlaşı Mekanizmaları

'Emeğin kanıtı' kavramı ilk olarak 1993 yılında Dwork ve Naor [9] tarafından istenmeyen (spam) epostaları engelleyebilme amacıyla geliştirilmiş bir sistem üzerinden literatüre kazandırılmıştır. Bahsi geçen sistemde kullanıcıların epostaları başarılı bir şekilde gönderebilmek için, belirli oranda çaba göstermelerini gerektiren bir maliyet fonksiyonu hesaplamaları ve sonucu sergiledikleri çabanın bir kanıtı olarak epostalara eklemeleri gerekmektedir. Diğer taraftan alıcılar ise sadece doğrulanabilir sonuca sahip epostaları kabul etmektedir.

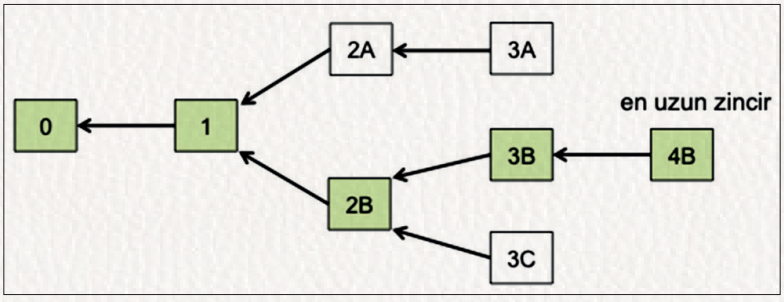
Kısaca bahsetmek gerekirse, emeğin kanıtı algoritması ispatlayıcı (prover) ve onaylayıcı (verifier) olarak adlandırılan iki kullanıcı tarafından gerçekleştirilmektedir. Algoritmada ispatlayıcı belirli bir amaca yönelik, emek gerektiren bir iş icra eder ve bu emeğini daha az maliyetli bir yolla onaylayıcıya ispatlar. Emeğin kanıtı algoritmasındaki ana nokta, ispatlayıcının ve onaylayıcının yüklendiği rollerdeki asimetridir. İspatlayıcı belli oranda elektrik tüketimi ya da özel donanım cihazları temini gibi ciddi kaynak gerektiren bir efor sergilerken, onaylayıcı görece çok daha az maliyetli bir şekilde ispatın doğruluğunu kontrol edebilmektedir. Emeğin kanıtı algoritmasının bu asimetrik yapısı kullanıldığı sistemleri yukarıdaki gibi istenmeyen epostalara ya da blokzincirinde olduğu gibi Sybil saldırılarına karşı dirençli yapmaktadır.

Blokzinciri dünyasında emeğin kanıtı mekanizması ilk olarak Satoshi Nakamoto [10] tarafından 2008 yılında 'bitcoin' olarak adlandırılan eşler arası elektronik para ödeme sisteminde kullanılmıştır. Burada emeğin kanıtı mekanizması, madenciler[3] arasından rastgele bir biçimde sıradaki bloğu üretecek

3 Bitcoinde ağ üzerinden paylaşılan işlemleri onaylayıp bloklara yazarak dağıtık defter sisteminin canlılığına katkı sağlayan kullanıcılar madenci olarak adlandırılmaktadır.

lideri belirlemek için kullanılmaktadır. Dwork ve Naor un çalışmasına [9] benzer biçimde burada da emeğin kanıtı mekanizması bir maliyet fonksiyonu üzerinden gerçekleştirilir; ama farklı olarak maliyet fonksiyonu için kriptografik özet fonksiyonu kullanılır. Biraz detaylandırmak gerekirse, madenciler ürettikleri blokta yer almasını istedikleri işlemlerle beraber 'nonce' olarak adlandırılan farklı x değerleri üzerinde ilgili özet fonksiyonunu çalıştırırlar. Bu süreci matematiksel bir bulmaca çözmek gibi düşünürsek; aslında madencilerin yapmaya çalıştığı, t bulmacanın zorluk derecesi olmak üzere farklı denemeler yaparak $H(işlemler|x) \leq t$ eşitsizliğini sağlayacak x sayısını bulmak olacaktır. İlgili x sayısını bulan madenci sıradaki bloğu üretecek lider olarak belirlenmiş olur. Sonrasında lider madenci eşitsizliği sağlayan işlemleri ve x değerini imzalayıp ağın diğer kullanıcılarıyla paylaşır. Diğer kullanıcılar liderin emeğini doğruladıktan ve işlemleri onayladıktan sonra paylaşılan yeni bloğu zincirlerine eklerler.

Bitcoinde emeğin kanıtı mekanizmasında kriptografik özet fonksiyonu kullanıldığından; eşitsizliği sağlayan x değerini bulmak ciddi kaynak sarfı gerektirirken, bulunan x değerini doğrulamak görece çok daha kolay olacaktır. Ayrıca farklı x değerleri için üretilen sonuçlar her zaman için tahmin edilemez olacağından; madenciler, ancak deneme yanılma yoluyla eşitsizliği sağlayan x değerini bulabileceklerdir. Dolayısıyla birim zamanda daha çok deneme yapabilenlerin lider seçilebilme şansı daha fazla olacaktır. Diğer bir ifadeyle, hash gücünü madencilerin birim zamanda farklı x sayısı deneyebilme kapasitesi ve w_i'yi i madencisinin hash gücü olarak tanımlarsak, i madencisinin lider olma olasılığı $w_i/\sum w_j$ olacaktır. Bu durum bitcoindeki lider seçimini hash gücü yüksek olanların daha çok bilet sahibi olduğu bir piyango çekilişine dönüştürecektir.



Şekil 2.5. Emeğin Kanıtı Algoritmasında Uygulanan En Uzun Kuralı

Bitcoinde bloklar ortalama 10 dakikada bir üretilir ve bu süre her 2016 blokta bir (yaklaşık iki hafta) bulmacanın zorluk derecesi ayarlanarak sabit tutulmaya çalışılır. Eğer bu iki haftalık sürede blok üretim aralığının ortalaması 10 dakikadan azsa zorluk derecesi artırılır, çoksa zorluk derecesi azaltılır. Ayrıca üretilen blokların yeni başka bir blok üretilmeden ağ üzerinden aynı süre içerisinde bütün kullanıcılara pratik bir biçimde ulaştırılabilmesi için blokların boyutu 1MB ile sınırlandırılmıştır. İdealde bu ayarlamalar iki farklı bloğun aynı anda üretilip bitcoin ağında dolaşıma sokulmasını engellemek için yeterli görünse de; emeğin kanıtı algoritmasının olasılıksal yapısı, bloğun iletimi esnasında ağda meydana gelebilecek gecikmeler ve kötü niyetli oyuncuların varlığı, farklı kullanıcıların aynı yüksekliğe (blokların zincir içerisindeki sırası) sahip iki farklı bloğu onaylanıp kendi zincirlerine eklemesine ve dolayısıyla ağda iki alternatif zincir oluşmasına sebep olabilmektedir. Bu durum blokzinciri literatüründe 'çatallanma problemi' olarak adlandırılmaktadır. Çatallanma problemini çözmek, diğer bir ifadeyle bitcoindeki kullanıcıların tek bir zincir üzerinde mutabık kalabilmelerini sağlamak için bitcoin en uzun zincir kuralını uygulamaktadır. Bu kurala göre bitcoin ağında alternatif zincirler olması durumunda düğümler bunlardan en çok emeğin harcandığı, diğer bir ifadeyle Şekil 2.5'te görüldüğü gibi en çok hash gücünün biriktirildiği zinciri seçeceklerdir. Nakamoto'ya [10] göre ağdaki hash gücünün çoğunluğu dürüst düğümlerin kontrolündeyse ve düğümlerin rasyonel hareket ettikleri kabul edilirse, düğümler kazanımlarını korumak adına en uzun zincire yatırım yapmaya devam edeceklerdir. Böylelikle bu zincir alternatiflerine kıyasla daha hızlı büyüyecek ve uzun vadede diğerlerini devre dışı bırakacaktır.

Bitcoin, uzlaşı mekanizmasına katılarak sistemin doğru ve etkili bir biçimde çalışabilmesine katkı sağlamaya teşvik için blok üreticilerini para ile ödüllendirmektedir. Kısaca, yeni bir blok üretildiğinde belli miktarda yeni bitcoin yaratılarak bloğa eklenen işlemlerden elde edilen kesintilerle beraber 'coinbase' adı verilen özel bir işlem yoluyla bloğun üreticisine verilmektedir.

i. Emeğin Kanıtı Tabanlı Uzlaşı Mekanizmalarının Değerlendirilmesi

Güvenlik. Garay vd. [11] dağıtık sistemler için uzlaşı mekanizmalarında tanımlanan güvenlik ve canlılık özelliklerini blokzinciri için yeniden tanımlamışlardır. Tanımlanan güvenlik özelliğine göre, herhangi bir işlem dürüst düğümlerden birinin zincirinin k (güvenlik parametresi) blok derinliğinde tu-

tuluyorsa, çok büyük bir olasılıkla diğer bütün düğümlerin zincirlerinde kalıcı bir şekilde tutuluyor demektir. Canlılık özelliğine göre ise, dürüst düğümlerin oluşturduğu işlemler nihayetinde diğer dürüst düğümlerin zincirlerinin *k* blok derinliğinde yer alacaktır. Garay vd. [11], dürüst düğümlerin ağdaki hash gücünün çoğunluğunu kontrol ettiği ve düğümlerin emeği kanıtı bulmacasının çözümüne kıyasla daha hızlı eş zamanlanabildikleri (senkronize oldukları) kabul edilirse, Nakamoto uzlaşı mekanizmasının tanımlanan güvenlik ve canlılık özelliklerini sağladığını göstermişlerdir. Düğümler yeterince hızlı eş zamanlanamadıklarında uzlaşı protokolünün dayanabileceği kötücül güç (kötü niyetli oyuncuların ağda sahip olduğu hash gücü) miktarı azalacaktır. Benzer bir çalışmada Decker ve Wattenhoffer [12] bitcoin ağında işlemlerin iletimiyle ilgili problemler olduğunda bitcoinin tolere edebileceği kötücül gücün %50' nin altına düşeceğini göstermişler.

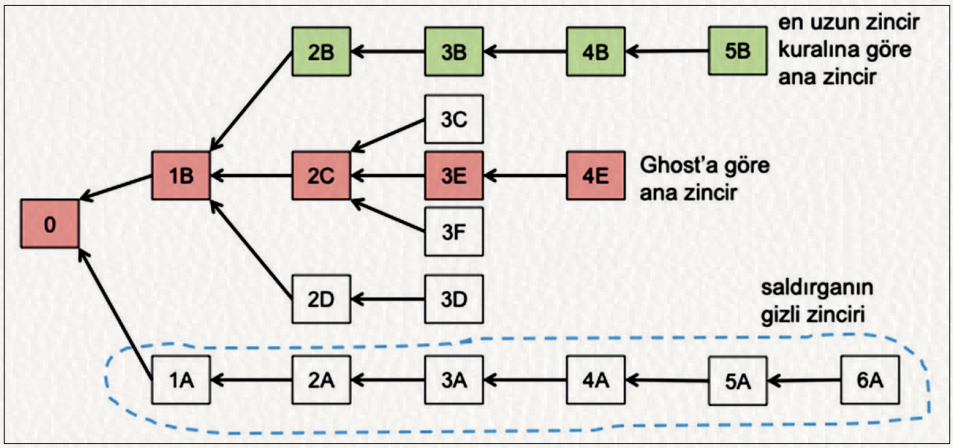
Eğer ağdaki kötü niyetli (Bizans) düğümlerin kontrol ettiği hash gücü %51'i geçerse, bu düğümler %51 saldırısı gerçekleştirerek geçici veya kalıcı olarak sistemin kontrolünü ele geçirebilirler. Bu yolla istemedikleri işlemlerin onaylanarak zincire eklenmesini engelleyebilir ya da dürüst düğümlerce onaylanmayacak işlemlerin onaylanarak zincire eklenmesini sağlayabilirler. Ya da en önemlisi 'çifte harcama' atağı gerçekleştirebilirler. Çifte harcama atağında saldırgan, daha önce bazı düğümlerce onaylanıp zincire eklenmiş bir işlemi iptal etmeye çalışır. Bunun için saldırganın o işlemi geçersiz kılacak yeni bir işlemi onaylayan, ilk zincire alternatif bir zinciri ağdaki düğümlere kabul ettirmesi gerekmektedir. Ağdaki hash gücünün %51'den fazlasını kontrol edebilen bir saldırgan böyle bir saldırı gerçekleştirebilir. Ağdaki düğümlerin rasyonel oyuncular olduğu düşünüldüğünde, Nakamoto'ya [10] göre ağdaki toplam hash gücünün yarıdan fazlasını elinde tutan bir grup bu gücü daha çok blok üreterek kazancını artırmaya yönelik kullanacak ve kazancını tehlikeye sokabilecek sistemi baltalamaya yönelik saldırılardan da uzak duracaktır. Bununla birlikte Bonneau [13] herhangi bir saldırganın rüşvet yoluyla geçici bir süre için hash gücünün yarıdan fazlasını ele geçirerek %51 saldırısı gerçekleştirebileceğini göstermiştir. Saldırı kısa süreli gerçekleştirileceği için yukarıdaki gibi saldırganın uzun vadede kazanımlarını kaybetmek gibi bir endişesi olmayacaktır. Burada rüşvetten kasıt saldırganın ağdaki diğer düğümlerin hash gücünü belli bir süre için kiralamasıdır. Örneğin nicehash.com [11] web sitesi kullanıcılarına emeğin kanıtı tabanlı uzlaşı mekanizmaları için hash gücü kiralama imkanı sunmaktadır. Cryp-

to51.app [12] sitesi de, nicehash.com web sitesindeki fiyatlar üzerinden ilgili platformlarda belli bir süre için %51 saldırısı gerçekleştirebilmenin maliyetini paylaşmaktadır.

Verimlilik. Bitcoinde blokların 10 dakikada bir üretildiği ve blok boyutunun 1MB olduğu dikkate alındığında, bitcoin ağında saniyede onaylanıp zincire eklenebilecek işlem sayısının 7-8 civarı olduğu ve diğer alternatiflerine nazaran verimliliğinin oldukça düşük kaldığı görülecektir (Visa'da bu sayı 24000'den çoktur [16]). Blok üretim aralığı ve blok boyutu bitcoinde verimliliği doğrudan etkileyen iki faktördür. Bu noktada verimliliği artırmak adına blok üretim aralığı daraltılabilir ya da blok boyutu artırılabilir. Yukarıda da belirtildiği gibi 10 dakikalık süre üretilen yeni bir bloğun başka yeni bir blok yaratılmadan önce ağ üzerinden herkese ulaştırılabilmesi için yeterlidir. Ama blok üretim aralığı daraltılırsa, henüz üretilen bloklar ağda tamamen yayılmadan yeni bloklar yaratılıp ağda dolaşıma sokulabilecektir. Diğer bir ifadeyle blok üretim aralığı daraltıldığında ağda çatallanma daha sık gözlemlenecektir. Diğer yandan blok boyutu artırılırsa üretilen blokların ağdaki dolaşım hızı azalacak ve yine bu durum çatallanma ihtimalini artıracaktır. Bu tespitleri destekleyen bir çalışmaya göre [17] blok üretim aralığı, blok boyutu ve yeni üretilen blokların ağda ulaştığı (çatallanmaya maruz kalmaksızın) düğümlerin yüzdesi arasında doğrudan bir ilişki vardır. Örneğin blok üretim aralığı 10 dakika ve blok boyutu 4MB olarak belirlenirse ağdaki düğümlerin %10'u üretilen yeni bloktan haberdar olmayacak bu durum ağın blok üretim kapasitesinde azalmaya sebep olacaktır. Eğer blok boyutu 38MB''lara çıkartılırsa bu oran %50'leri bulacaktır.

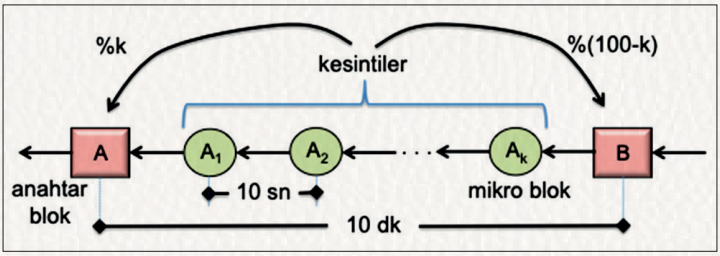
Yukarıdaki açıklamalardan da anlaşılacağı üzere verimliliği artırmak için blok boyutunu artırmak ya da blok üretim aralığını daraltmak ağda çatallanma oluşmasını riskini artırmaktadır. Her ne kadar bitcoin gibi emeğin kanıtı tabanlı uzlaşı mekanizmaları kullanan platformlar en uzun zincir kuralıyla bu çatallanmaları gidermeye çalışsalar da, zincirde çatallanmaların sayısının artması geçici süreyle de olsa ağdaki dürüst oyuncuların alternatif zincirlere dağılmasına ve kötü niyetli oyuncuların daha az hash gücüyle çifte harcama gibi saldırılar gerçekleştirebilmelerine imkan sağlayacaktır. Bu bilgiler ışığında, Sompolinsky ve Zohar [18] verimliliği artırmak adına Ghost (greedy heaviest-observed sub-tree –açgözlü gözlemlenen en ağır alt ağaç) adını verdikleri, blokzincirdeki çatallanma probleminin çözümü için en uzun zincir kuralına alternatif bir yöntem önermişlerdir. Ghost algoritmasında düğümler,

başlangıç (genesis) bloğundan başlayarak her çatallanmada en ağır alt ağacı tercih ederek yollarına devam edecekleri ana zinciri belirlerler.



Şekil 2.6. Ghost Protokolüne Göre Ana Zincirin Belirlenmesi

Blokzincir ağının, örneğin Şekil 2.6'daki gibi bir çatallanmayla karşı karşıya kaldığını ve bir saldırganın saklı tuttuğu lokal zincirini diğer düğümlerle paylaşarak ağı bölmeye çalıştığını farz edelim. Ghost protokolünü takip ederek ana zinciri belirleyecek olursak, 1A-köklü alt ağaç 6 blok ve 1B-köklü alt ağaç 12 blok içerdiği için düğümler 1B-köklü ağacı seçecektir. Sonrasında 2B-köklü alt ağaç 4 blok, 2C-köklü ağaç 5 blok ve 2D-köklü ağaç 2 blok içerdiği için, düğümler 2C-köklü ağacı seçeceklerdir. Bu şekilde devam ettiklerinde düğümler Şekil 2.6'daki gibi nihayetinde son bloğu 4E olan zinciri ana zincir olarak belirleyeceklerdir. Diğer yandan eğer düğümler ana zinciri alternatifler arasından en uzun olanı şeklinde belirliyor olsaydı, dürüst düğümlerin paylaştıkları zinciri değil saldırganın zincirini tercih ediyor olacaklardı. Burada şunu belirtmekte fayda var ki; en uzun zincir kuralının uygulandığı blokzincirlerde ana zincirde yer almayan öksüz bloklara yapılan yatırım boşa gitmektedir. Aksine Ghost protokolü, öksüz blokları çatallanmanın giderilmesi sürecine dahil ederek bu bloklara yapılan yatırımın blokzincirin güvenliğini artırma noktasında değerlendirilmesini sağlamaktadır. Öksüz bloklar dahil edildiğinde saldırganın rekabet edeceği ana zincirin ağırlığı artacak ve saldırganın işi zorlaşacaktır. Böylece blokzincirin güvenliğini riske atmadan blokzincirin işlem kapasitesini artırmak mümkün olacaktır.

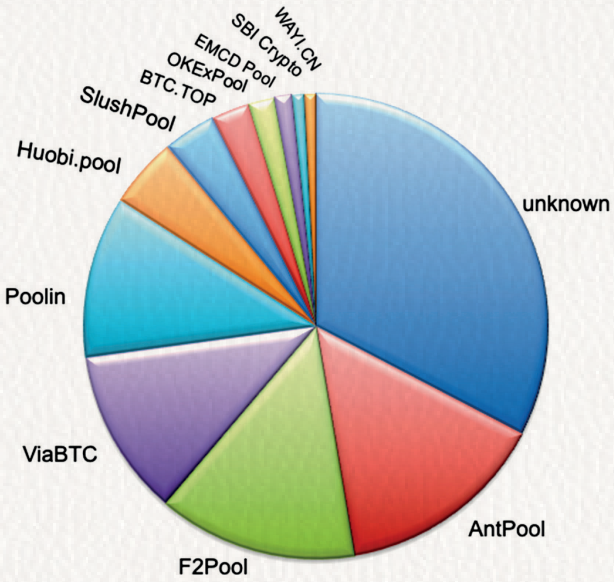


Şekil 2.7. Bitcoin-NG Protokolü

Blokzincirin verimliliği artırmak amaçlı bir diğer çalışma Eyal vd. [19] tarafından önerilmiştir. Bitcoin-NG (next generation – gelecek nesil) adı verilen çalışmada blok üretimi, lider seçimi ve işlem serileştirmesi (transaction serialization) olarak iki aşamada ele alınmıştır. Lider seçimi aşamasında bitcoinde olduğu gibi emeğin kanıtı algoritması üzerinden bir lider belirlenir ve lider anahtar blok adı verilen bloğu oluşturarak ağla paylaşır. Şekil 2.7'de görüleceği gibi, bitcoinden farklı olarak aradaki 10 dakikalık zamanı değerlendirme adına Bitcoin-NG, seçilmiş lidere mikro blok adı verilen yeni bloklar üretme imkanı sunmaktadır. Mikro blokların doğru şekilde onaylanabilmeleri için liderin açık anahtarının anahtar bloklara eklenmesi gerekmektedir. Bitcon-NG'de mikro bloklardan toplanan kesintiler, doğrudan mikro blokları oluşturan lidere verilmek yerine oluşturulan mikro blokların ağdaki diğer düğümlerce onaylanmasını teşvik etmek adına sonraki liderle paylaştırılmaktadır. Paylaşımın ne oranda olacağı seçilen liderlerin hash gücüne bağlı olarak değişebilecektir.

Merkezileşme Problemi. Bitcoin kripto para borsalarında itibari para (fiat currency) ile alınıp satılabilen bir yatırım aracı olarak değer görmektedir. Bu bölümün yazılması esnasında bitcoin en çok işlem hacmine sahip borsalardan bir olan Paribu'daki karşılığı 310.000TL'dir. TL karşılığı olarak bir blok üreticisinin 2 milyon TL civarı bir para kazanabildiği düşünüldüğünde bitcoin ağındaki düğümler için zincire eklenebilecek bloklar üretmek ciddi cazibesi olan bir uğraş olacaktır. Bununla birlikte düğümlerin diğer düğümlerce onaylanabilecek bir blok üretebilmesi için emeğin kanıtı algoritması yoluyla lider olarak seçilmesi gerekmektedir. Yukarıda da belirtildiği gibi herhangi bir düğümün lider seçilebilme olasılığı elindeki hash gücüyle doğru orantılıdır. Dolayısıyla görece çok hash gücüne sahip olanların pastadan

daha çok pay alma ihtimali yüksek olacaktır. Bu gerçekten hareketle blok üretim yarışında avantajlı konuma geçmek isteyenler ellerindeki hash gücünü artırmak adına özel donanım cihazlarına yatırım yapmaktadır. Özellikle ASIC gibi sadece spesifik bir amaç için üretilen cihazların bu sürece dahil edilmesi bu alanda yatırım yapabilecek kaynağı olmayan düğümleri yarışın dışına itmektedir.



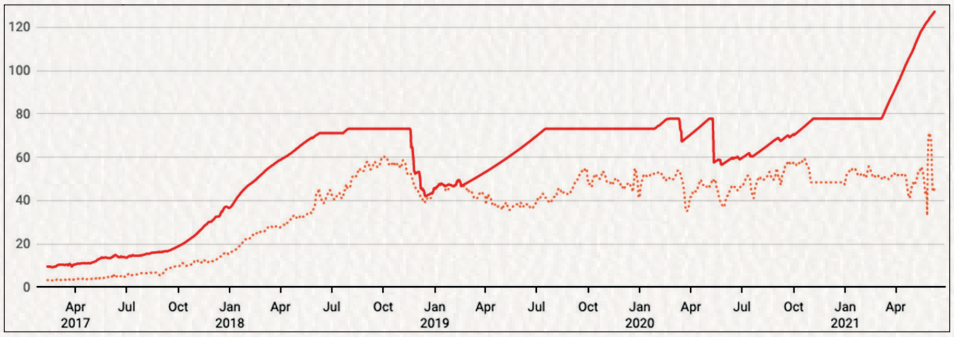
Şekil 2.8. Bitcoin Ağında Büyük Madencilik Havuzları Arasında Hash Gücünün Dağılımı

Bu tür küçük oyuncular için ikinci bir seçenek madencilik havuzları (mining pools) yoluyla güçlerini birleştirmek ve blok üretim yarışını kazanma ihtimallerini artırmak olacaktır. Madencilik havuzlarında kazanılan ödül, katılımı teşvik amacıyla havuzdaki düğümler arasında blok üretimine katkılarıyla orantılı bir biçimde paylaştırılmaktadır. İlk bakışta bu fikir küçük oyuncuları da blok üretim sürecine dahil ederek daha adil bir yarışa katkı sunuyor gibi görünse de, zamanla blok üretimini belli sayıda madencilik havuzunun tekeline bırakan bir süreci doğurmuştur. Gencer vd. [20] 2016 yılında 10 aylık bir süre zarfında bitcon ağındaki blok üretim süreçlerini gözlemlemişler ve bu süre zarfında blokların %90'ının 16 madencilik havuzu tarafından üretildiğini tespit etmişlerdir. Yine blockchain.com web

sitesinin periyodik olarak yayınladığı, belirli bir zaman aralığında üretilen bloklar baz alınarak hesaplanan hash gücü dağılım grafiği üzerinden benzer sonuçlara ulaşılabilir [21]. Şekil 2.8'de görüldüğü gibi son 4 gün içerisinde zincire eklenen blokların %80'den fazlası 5 madencilik havuzu tarafından üretilmiştir. Ağdaki hash gücünün bir kaç grubun elinde toplanması, bu grupların organize olarak %51 saldırısı yoluyla bitcoin ağının güvenliğini riske atma ihtimallerini ortaya çıkarmaktadır.

ACIS gibi bitcoin madenciliği için üretilen özel cihazların blok üretimi yarışında küçük oyuncular için dezavantaj oluşturduğundan yukarıda bahsetmiştik. Oluşan bu dezavantajı gidermek adına bitcoinde kullanılan emeğin kanıtı algoritmasına alternatif olarak, belli büyüklükte belleğe erişimi gerektiren algoritmalar literatüre sunulmuştur [22]. Algoritmaların belleğe bağımlı bu yapısı, performanslarını da belleğe erişimdeki gecikme ya da kullanılan belleğin boyutu gibi parametrelere bağlamaktadır. Bu durum da ACIS gibi donanım parçalarının alternatiflerine (CPU/GPU) karşı üstünlüklerini ellerinden almaktadır. Ethereum (Ethash algoritması [23]) ve Monero (CryptoNight algoritması [24]) gibi bazı kripto paralar bu tarz belleğe bağımlı uzlaşı mekanizmaları kullanmaktadırlar.

Elektrik Tüketimi. Emeğin kanıtı algoritmasında madenciler SHA256 özet fonksiyonunu kullanarak istenilen *x* sayısını buluncaya kadar denemeler yapmaktadır. Ve bu denemelerin her biri belli miktarda elektrik tüketimi gerektirmektedir. Bu miktar bitcoin ağında saniyede yapılan toplam deneme miktarı (hashrate) ile doğru orantılı olarak artmaktadır. Özellikle ASIC gibi donanım cihazlarının madencilik süreçlerine dahil edilmesi ile beraber emeğin kanıtı algoritmasındaki zorluk derecesi iyice artmış ve bunun sonucu olarak bitcoin ağının hashrate'i ciddi boyutlara ulaşmıştır. Buna paralel olarak da blok üretiminde tüketilen elektrik miktarı ülkelerin elektrik tüketim oranlarıyla yarışır hale gelmiştir. Emeğin kanıtı tabanlı uzlaşı mekanizmalarının kullanıldığı Bitcoin ve Ethereum gibi kriptoparalarda enerji sarfiyatıyla ilgili verilerin paylaşıldığı diigiconomist.com web sitesi incelendiğinde, bitcoin ağında son bir yılda tüketilen elektriğin Hollanda ve Birleşik Arap Emirlikleri ölçeğindeki ülkelerden fazla olduğu görülecektir. Ayrıca Şekil 2.9 incelendiğinde bitcoin ağında blok üretimi için tüketilen elektriğin emeğin kanıtı algoritmasının zorluk derecesiyle paralel son iki yıllık zaman zarfında iki katına çıktığı gözlemlenecektir [25].



Şekil 2.9. Bitcoin Ağındaki Elektrik Tüketimi (Terawatt Saat Cinsinden)

2.2.1.2. Hisse Kanıtı Tabanlı Uzlaşı Mekanizmaları

Hisse kanıtı kavramı ilk olarak bitcoin forumlarında QuantumMechanic mahlaslı bir kullanıcı tarafından Nakamoto uzlaşı mekanizması için emeğin kanıtına alternatif olarak önerilmiştir. Yukarıda da belirtildiği gibi emeğin kanıtı algoritmasının kaba kuvvet hesaplamalarına bağlı yapısı sebebiyle, zincire eklenecek yeni blokları belirleyecek liderlerin seçimi ciddi elektrik tüketimi gerektirmekteydi. Bu gereksinimi gidermek adına QuantumMechanic'in önerdiği; basitçe ifade edecek olursak, emeğin kanıtı algoritmasının lider seçiminde referans alınan kriter adayların uygulayabildiği hash gücünün, adayların blokzincir ağında sahip olduğu hisselerle değiştirilmesidir. Yani hisse kanıtı algoritmasında sıradaki bloğu oluşturacak lider, blokzincir ağındaki hisse sahibi adaylar arasından hisse miktarıyla orantılı olarak olasılıksal bir yolla belirlenecektir. Diğer bir ifadeyle hisse kanıtı algoritmasında lider, hissesi daha fazla olanların daha çok bilet sahibi olduğu bir piyango çekilişi yoluyla seçilecektir. Emeği kanıtı tabanlı uzlaşı mekanizmalarıyla kıyaslandığında; hisse kanıtı tabanlı olanlar çok daha az elektrik tüketimi gerektirdiğinden çevreye daha duyarlı, birim zamanda daha çok işlemin onaylanmasına imkan verdiğinden daha verimli mekanizmalardır.

Laf kalabalığını engellemek adına bundan sonra hisse kanıtı tabanlı uzlaşı mekanizması kısaca hisse kanıtı olarak adlandırılacaktır. Bu bölümde uzlaşı mekanizmaları, hem hisse kanıtı hem de emeğin kanıtı algoritmalarını kullanan hibrit hisse kanıtı, liderlerin komite tarafından belirlendiği komite tabanlı hisse kanıtı ve ağdaki düğümlere sahip oldukları hisseleri lider seçiminde kullanılabilmesi amacıyla bir başkasına devretmeye olanak sağlayan yetkilendirilmiş hisse kanıtı olmak üzere üç başlık altında incelenecektir.

i. Hibrit Hisse Kanıtı

Burada hibrit hisse kanıtını mekanizması bu yapıyı kullanan Peercoin [26] ve Nxt [27] sistemleri üzerinden anlatılacaktır. Peercoin, hisse kanıtının ilk uygulaması olarak 2012 yılında King ve Nadal tarafından literatüre sunulmuştur. Bitcoinde olduğu gibi sıradaki bloğu oluşturacak lider emeğin kanıtı algoritması yoluyla seçilmektedir. Ancak bitcoinden farklı olarak algoritmanın zorluk derecesi madencinin elinde tuttuğu 'akçe[4] yaşı' miktarına göre belirlenmektedir. Peercoin'de akçe yaşı düğümlerin madencilik için ellerinde tuttukları akçeleri değerlendirmeye yarayan bir kavramdır. Bu kavrama göre, örneğin düğümler kendilerine gelen 10 akçeyi 100 blok boyunca harcamazlarsa o 10 akçenin akçe yaşı 1000 olarak hesaplanacaktır. Buradan hareketle Peercoin'de akçe yaşı miktarı çok olan düğümler için emeğin kanıtı algoritması görece daha kolay olacak ve buna bağlı olarak lider seçilme ihtimalleri artacaktır denilebilir. Akçe yaşı kavramı ayrıca; madencilik yapmak isteyen küçük oyunculara ellerindeki sınırlı miktarda akçeyi bekleterek değerlendirme ve böylelikle blok üretim yarışına katılabilme imkanı sunsa da, yine bu özelliğiyle %51 atağı gibi saldırıların maliyetini azaltmaktadır.

Sistemi saldırıya daha açık hale getirdiği için Peercoin'in aksine Nxt'de akçe yaşı kavramı kullanılmamaktadır. Yani Nxt'de akçeler harcanmadıkça değer kazanmazlar. Yine Peercoin'dekine benzer olarak Nxt'de sıradaki bloğu üretecek liderler zorluk derecesi düğümlerin ellerindeki akçe miktarına göre belirlenen emeğin kanıtı algoritması yoluyla seçilmektedir. Dolayısıyla Nxt'de akçesi çok olanların lider seçilme ihtimalleri daha çok olacaktır. Bununla birlikte düğümlerin sahip oldukları akçeleri kontrol ettikleri hesaplar arasında dolaştırarak lider seçilme ihtimallerini artırmak isteyebilirler. Nxt bu durumu engellemek için düğümlere blok üretim sürecine dahil etmek istedikleri akçeleri belli bir süre tek bir adreste tutma şartı koşmaktadır.

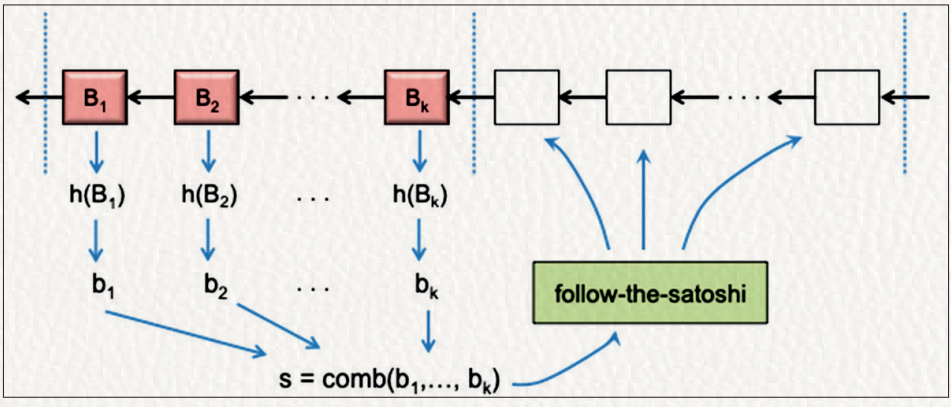
ii. Komite Tabanlı Hisse Kanıtı

Komite tabanlı hisse kanıtı algoritmalarında zamanın küçük zaman dilimlerinden oluşan turlara (epoch) bölündüğü varsayılır. Her tur için turdaki blokların üretiminden sorumlu bir komite oluşturulur. İster emeğin kanıtı algoritmasında ya da ister hibrit hisse kanıtı algoritmalarında olsun aynı anda iki fark-

4 Akçe kelimesi düğümlerin blokzinciri ağında sahip oldukları hisseyi belirtiyor olup zaman zaman hisse kelimesi yerine kullanılacaktır.

lı düğüm bulmacayı çözüp lider olarak öne çıkabilmektedir. Komite tabanlı hisse kanıtı algoritmaları, tur içerisindeki her bir zaman dilimine yalnızca tek bir lider atayarak aynı anda iki farklı düğümün lider olmasının önüne geçmektedir. Kurulum aşamasında ilk turdan sorumlu komite üyeleri başlangıç bloğu üzerinden ağa ilan edilir. Sonrasında sorumlu oldukları tur süresince komite üyeleri, aralarında güvenli çok taraflı hesaplama (secure multiparty computation) gibi araçlar çalıştırmak suretiyle her bir zaman dilimine bir lider atayarak bir sonraki turun komitesini oluştururlar.

Burada komite tabanlı hisse kanıtı algoritması kullanan Chain of Activity (CoA) [28] ve Ouroboros [29] protokolleri incelenecektir. İki protokolde de ilgili zaman dilimleri için liderler, follow-the-satoshi (FTS) adı verilen bir algoritma üzerinden rastgele bir yolla belirlenmektedir. Kısaca FTS algoritması, rastgele bir sayıyı ve lideri belirlenecek olan zaman diliminin turdaki sırasını girdi olarak alır ve deterministik bir şekilde dolaşımdaki akçelerin en küçükleri (bitcoinde örneğin Satoshi ve Ethereumda Wei) arasından birini seçer. Sonrasında algoritma seçilen bu akçenin başlangıç bloğundan itibaren izini sürerek şu anki sahibini tespit eder ve o düğümü ilgili zaman diliminin lideri olarak belirler. Akçesi fazla olan düğümlerin FTS algoritması tarafından seçilme ve dolayısıyla lider olarak belirlenme ihtimali daha fazla olacaktır. Kötü niyetli düğümler, FTS algoritmasında kullanılacak rastgele sayının oluşturulması sürecine müdahale ederek kendi kontrolündeki düğümlerin lider olarak seçilmesini sağlayabilirler. Dolayısıyla komite tabanlı hisse kanıtı algoritmalarında kritik olan konu, FTS algoritmasına girdi olarak verilecek olan rastgele sayının güvenli bir şekilde oluşturulmasıdır.



Şekil 2.10. CoA Protokolü

Şekil 2.10'da görüldüğü gibi; CoA protokolünde belirli bir turdaki her bir i lideri kendi zaman dilimi için B_i'bloğunu üretip ağla paylaştıktan sonra, h kriptografik özet fonksiyonu olmak üzere $h(B_i)$ değerini hesaplar ve $h(B_i)$'nin ilk biti b_i'yi FTS algoritmasına girdi olacak rastgele sayının oluşturulmasında kullanılmak üzere diğer liderlere gönderir. Sonrasında toplanan b_i değerleri üzerinde deterministik bir fonksiyon olan $comb$ çalıştırılarak FTS algoritmasına girdi olacak $comb(b_1, \dots, b_k)$ rastgele sayısı oluşturulur. CoA protokolünde liderlerin rastgele sayının oluşumuna katkı olarak, ürettikleri bloğun özet değerinin ilk bitini sunuyor olmaları sistem için zafiyet doğurmaktadır. Bunu gidermek adına Ouroboros, FTS algoritmasının kullanacağı rastgele sayıyı liderler arasında çalıştırılan güvenli birçok taraflı hesaplama yöntemi yoluyla oluşturmaktadır. Liderler bu yöntemi iki aşamalı olarak uygulamaktadırlar. İlk aşamada liderler önce bir taahhüt şeması yardımıyla çok taraflı hesaplama protokolünde kullanılacak lokal girdileri için bir taahhüt oluştururlar ve bu taahhüdü sır paylaşımı şeması (secret sharing scheme) yoluyla diğer liderlerle paylaşırlar. İkinci aşamada taahhüdünü paylaştıkları girdiyi yine aynı sır paylaşımı yoluyla diğer liderlerle paylaşarak doğru değere taahhütte bulunduklarını doğrularlar. Sonrasında liderler bu girdiler üzerinden FTS algoritmasında kullanılacak rastgele sayıyı hesaplarlar.

iii. Yetkilendirilmiş Hisse Kanıtı

Komite tabanlı hisse kanıtı algoritması kullanılan sistemlerde görece daha az hissesi olan düğümler, lider seçilme olasılıkları daha az olduğundan daha az çevrimiçi olmayı tercih edecekler ve belki de lider seçilseler bile farkında olmayacaklardır. Diğer yandan sistemin sağlıklı çalışabilmesi için, belirli bir zaman dilimi için lider olarak atanmış düğümlerin ilgili zaman diliminde çevrimiçi olması, kendinden beklendiği üzere bloklar üretip paylaşması ve sonraki turun liderlerinin belirlenmesine katkı sağlaması gerekmektedir. Yetkilendirilmiş hisse kanıtı algoritması, yukarıda bahsedildiği gibi sahibi çevrimiçi olmadığı için sistemin sağlıklı bir şekilde yürütülmesine katkı sağlamayan hisselerin daha aktif düğümlere devredilerek değerlendirilmesine imkan tanıyan, komite tabanlı hisse kanıtı algoritmasının daha demokratik bir formudur denilebilir.

Yetkilendirilmiş hisse kanıtı algoritmalarında düğümler özel işlemler vasıtasıyla ellerindeki hisseleri lider seçimi mekanizmasında kullanılabilmesi ama-

cıyla başka düğümlere devrederler. Komite tabanlı hisse kanıtında olduğu gibi bu algoritmalarda da zaman tur adı verilen belli sayıda (bu sayının *k* olduğunu kabul edelim) küçük zaman dilimlerinden oluşan parçalara bölünmüş olarak ele alınır. Her bir turun başında devredilmiş tüm hisseler hesaba katılarak en çok hisse sahibi ilk *k* vekil (delegatee) sırasıyla ilgili turdaki zaman dilimlerine blokları üretecek liderler olarak atanırlar. Bu *k* sayısı yetkilendirilmiş hisse kanıtı algoritması kullanan sistemlerde farklılık gösterebilmektedir. Örneğin bu sayı Tron'da [30] 27 olarak belirlenmiştir. Ayrıca yetkilendirilmiş hisse kanıtı algoritmalarında pasif düğümlere, ellerindeki hisseleri devretme noktasında teşvik amacıyla blok üretiminden toplanan ödüllerden pay verilmektedir. Bu algoritmalar aynı zamanda hissesini devretmiş olanlara, vekillerinin performansına göre hisseleri geri alma veya başka düğümlere devretme imkanı da sunmaktadırlar.

iv. Hisse Kanıtı Tabanlı Uzlaşı Mekanizmalarının Değerlendirilmesi

Güvenlik. Hisse kanıtı tabanlı uzlaşı mekanizmaları ağdaki akçelerin yarısından azını kontrol edebilen kötü niyetli düğümleri tolere edebilmektedir. Yalnız şunu belirtmemiz gerekir ki, bu başlık altında incelenen protokollerden sadece Ouroboros yukarıdaki kabulden hareketle protokolün güvenlik ve canlılık özelliklerini sağladığını gösteren formel bir kanıt sunmuştur. Diğer protokollerle ilgili literatüre yeterli güvenlik analizi sunulmamıştır.

Emeğin kanıtı tabanlı uzlaşı mekanizmalarına benzer olarak, burada da kötü niyetli düğümler ağdaki hisselerin çoğunluğuna sahip olabilirlerse geçici veya kalıcı olarak sistemi kontrol edebilir hatta çifte harcama atağı gerçekleştirebilirler. Burada bir parantez açacak olursak; hisse kanıtı tabanlı uzlaşı mekanizmasını cazip kılan en önemli faktörlerden biri, blokzincir ağında hissesi bulunanların sistemin sağlıklı bir şekilde çalışmasını sağlayacak en uygun aktörler olması gerçeğidir. Çünkü sistem zarar gördüğünde bundan en çok etkilenecek olanlar sistemde en çok hissesi bulunanlar olacaktır. Dolayısıyla ağdaki düğümlerin rasyonel olduğu düşünülürse, ağdaki hisselerin yarısından fazlasını elinde tutan bir grup bunu sistemi zarara uğratmak yerine kazancını artırabilmek adına sistemin sürekliliğini sağlamak için kullanacaktır.

Blokzincir ağında madenciler farklı alternatif zincirlere yatırım yaparak blok üretiminden elde ettikleri karı artırmak isteyebilirler. Emeğin kanıtı tabanlı uzlaşı mekanizmalarında bunu gerçekleştirebilmek için madenciler her bir zincir için farklı blok üretmelidirler. Alternatif zincirler için farklı emeğin kanıtı algoritması çalıştırmak madenci için hash gücünü alternatif zincirler arasında paylaştırmak demek olacağından, rasyonel bir madenci böyle bir strateji tercih etmeyecektir. Hisse kanıtı tabanlı uzlaşı mekanizmalarında ise durum farklıdır. Düğümler belli zaman dilimlerine blok üretecek liderler olarak atandıklarından ve uzun vadede hangi zincirin ana zincir olarak kabul edileceği kesin olarak belli olmadığından, rasyonel madenciler alternatif zincirlere bloklar ekleyerek karlarını artırmayı tercih edeceklerdir. Literatürde 'sıfır hisse' (nothing at stake) olarak adlandırılan bu saldırıyı engelleyebilmek için Daian vd. [31] aynı zaman dilimi için iki farklı blok paylaşan madencinin belli miktarda hisse kaybettiği bir ceza mekanizması önermiştir.

Yukarıda da belirtildiği gibi komite tabanlı hisse kanıtı algoritmalarında her bir turda komite üyeleri bir sonraki turun komitesini seçmekle yükümlüdür. Örneğin Chain of Activity protokolünde [28] komite üyeleri ürettikleri blokları kullanarak oluşturdukları rastgele bir sayı üzerinden bir sonraki turun komitesini seçmektedirler. Komite seçiminin tur içerisinde üretilen bloklar üzerinden yapılması sistemi 'grinding' adı verilen saldırılara açık hale getirecektir. Bu saldırıda komite üyelerinden bir kısmını kontrol edebilen bir düşman, kontrolündeki liderlerden bazılarına blok ürettirmeyerek veya bazılarının ürettiği blokların içeriğini değiştirerek kendi kontrolündeki düğümlerin sonraki turun komitesinde yer almasını sağlamaya çalışacaktır. Bu saldırının engellenebilmesi için komite üyelerinin seçiminde kullanılacak rastgele sayının Ouroboros protokolünde [29] olduğu gibi tekdüzeliğe yakın bir rastlantısallıkla üretilmesi gerekmektedir.

Verimlilik. Blokzincir ağında onaylanıp deftere yazılabilen işlem kapasitesi açısından hisse kanıtı tabanlı uzlaşı mekanizmalarının emeğin kanıtı tabanlı olanlarına nazaran çok daha iyi olduğu söylenebilir. Bitcoin ağında saniyede işlenebilen işlem sayısı 7-8 civarıyken Ouroboros protokolünde örneğin bu rakam 250'lere çıkmaktadır. Yetkilendirilmiş hisse kanıtı algoritmalarını kullanan sistemlerde bu sayı daha da artmaktadır. Tron platformu saniye-

de 2000 civarı işlem onaylayıp zincire ekleyebilmektedir. Komite tabanlı hisse kanıtı algoritmalarının performansını etkileyen bir diğer konu da komite boyutudur. Komite üyeleri karşılıklı mesaj alışverişiyle sonraki turun liderlerini seçtikleri için, komite boyutu çok yüksek tutulduğunda ağdaki mesaj yükü artacaktır. Ayrıca turun süresi uzadıkça sistemin kontrolünü ele geçirmek isteyen kötü niyetli oyunculara daha fazla alan açılmış olacaktır. Dolayısıyla komite boyutunun bu kısıtlar altında protokolün verimliliği için optimum değerde tutulması gerekmektedir (Chain of Activity protokolünde k=459 olarak belirlenmiştir). Ouroboros protokolü, komite boyutunu sınırlandırabilmek adına komite üyesi seçilmek için belirli miktarda hisse sahibi olma şartı koşmaktadır.

Teşvik. Yukarıda da belirttiğimiz gibi emeğin kanıtı tabanlı uzlaşı mekanizmalarında madenciler, blok üreterek sistemin sürekliliğine katkı sağladıkları için yeni üretilen belli bir miktar akçe ile ödüllendirilirler. Bu yöntemle madenciler blok üretimi noktasında teşvik edilirken, üretimi zamana yayılarak akçelerin ağ içerisinde görece daha adil bir dağılımı sağlanır. Hibrit hisse kanıtı kullanan sistemler veya yetkilendirilmiş hisse kanıtı kullanan bazı bazı sistemler hariç tutulursa hisse kanıtı tabanlı uzlaşı mekanizmalarında emeğin kanıtında olduğu gibi bir akçe üretimi yoktur. Sistemin yaşam süresi boyunca dolaşımda olacak akçeler tek seferde üretilip başlangıç bloğu üzerinden düğümler arasında paylaştırılmaktadır. Bu durum hisse kanıtı algoritmaları için eleştiri konusu olmaktadır. Yeni akçe üretimi olmadığı için madencileri teşvik noktasında hisse kanıtı algoritmaları blok üretimini işlemlerden elde edilen kesintilerle ödüllendirmektedir. Emeğin kanıtı algoritmalarında olduğu gibi blok üretimi ciddi hash gücü ve elektrik sarfiyatı gerektirmediği için, bu ödül sistemin sürekliliğini sağlamak için yeterli olacaktır. Tezos [32] gibi bazı sistemler, blok üreticilerini işlemlerden elde edilen kesintilerin yanı sıra yıllık bazda enflasyon oranı diye tabir edilen bir yüzdelik üzerinden üretilmiş yeni akçeler yoluyla da ödüllendirmektedir.

Merkezileşme Problemi. Bitcoin üzerinden emeğin kanıtı algoritması kullanan sistemlerin hash gücüne dayalı yapısı sebebiyle merkezileşme problemine maruz kaldığından bahsetmiştik. Benzer problem hisse kanıtı algoritması kullanan sistemler için de mevcuttur. Algoritma büyük olasılıkla blokzincir ağında daha çok hissesi olanları blok üreticisi olarak seçeceği için, sistem

içerisinde zengin diye tabir edebileceğimiz zümre blok üretiminden elde edilen ödüller üzerinden servetini daha da artıracaktır. Dolayısıyla blokzinciri ağında akçelerin çoğu uzun vadede emeğin kanıtı algoritmasını kullanan sistemlerde olduğu gibi belli bir zümrenin elinde toplanacaktır.

Mahremiyet Problemi. Emeğin kanıtı algoritmasında sıradaki bloğu üretecek liderin kimliği ancak lider bloğu üretip paylaştıktan sonra blokzincir ağınca bilinmektedir. Bunun aksine hisse kanıtı algoritmalarında ise (hibrit hisse kanıtı hariç) belirli bir tur için blokları üretecek olan liderlerin kimlikleri, tur başlamadan önce belirlenip ağdaki düğümlerle paylaşılmaktadır. Blok üreticilerinin kimliklerinin bu şekilde önceden paylaşılması, sistemi baltalamak isteyenler saldırganlara bir nevi açık adres göstermek olacaktır. Bu tür saldırganlar, DoS türü saldırılar yoluyla seçilmiş liderlerin blok üretmesini veya ürettikleri blokları diğer düğümlerle sağlıklı bir biçimde paylaşmasını engelleyebilirler. Bu tür saldırıları engelleyebilmek amacıyla Kerber vd. [33] Ourobors Crypsinous isimli, belirli bir tur için seçilmiş liderlere blokları üretip blokzincir ağıyla paylaşıncaya kadar kimliklerini gizleme imkanı veren hisse kanıtı tabanlı bir protokol önermişlerdir.

2.2.1.3. Alan Kanıtı Tabanlı Uzlaşı Mekanizmaları

Bölüm 2.2.1.1'de belirtildiği gibi emeğin kanıtı algoritmasında blok üretimi ciddi miktarda elektrik sarfiyatı gerektirmektedir. Emeğin kanıtı algoritmasını kullanan sistemlerden biri olan bitcoinde blokların üretimi için yıllık bazda tüketilen elektrik Hollanda ölçeğinde ülkelerle eşit hale gelmiştir. Bu gibi sistemlerde emeğin kanıtı algoritmasının zorluk derecesi blok üretim yarışına çokça madencinin katılması sebebiyle haylice artmış ve bu durum madencileri yarışta daha avantajlı konuma geçmek için ciddi yatırım gerektiren ASIC gibi özel cihazlar almaya sevk etmiştir. Bu cihazlar sadece blok üretimi için kullanılmakta olup üretim durduğunda atıl durumda kalacaklardır. Ayrıca bu tarz cihazlar blok üretim yarışını küçük oyuncuların aleyhine çevirmektedirler.

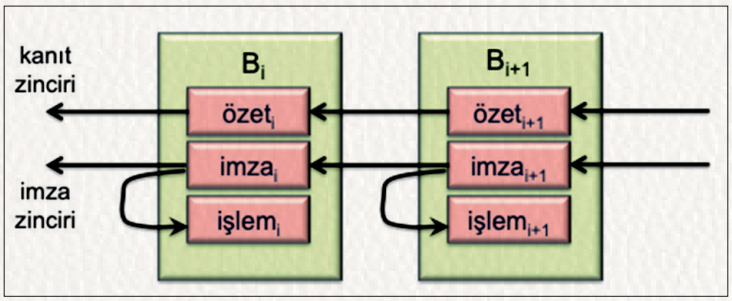
Alan kanıtı algoritmaları; hisse kanıtı algoritmaları gibi, yukarıda bahsedilen problemleri gidermek adına, lider seçiminde baz alınan referans noktasını, lider adaylarının uygulayabildiği hash gücünden lokal belleklerinde madencilik

için ayırabildikleri alana kaydıran bir alternatif olarak öne sürülmüştür. Alan kanıtı algoritmasında sıradaki bloğu oluşturacak lider, lokal belleklerinde madencilik için alan ayırmış adaylar arasından ayırdıkları alan hacmiyle orantılı olarak diğer alternatiflerinde olduğu gibi olasılıksal bir yolla belirlenir. Diğer bir ifadeyle alan kanıtı algoritmasında lider, madencilik için daha fazla alan ayıranların daha çok bilet sahibi olduğu bir piyango çekilişi yoluyla seçilmektedir. Alan kanıtı algoritması adaylara, ASIC gibi özel donanım cihazları satın almaksızın sabit belleklerindeki kullanmadıkları alanları madencilik için ayırarak blok üretim yarışına katılma imkanı sunmaktadır. Blok üretim yarışındaki şansını artırmak adına adaylar ekstra bellek satın alsalar bile, ASIC gibi cihazların aksine bu belleği farklı amaçlar için de kullanabileceklerdir.

Uzlaşı mekanizmasına geçmeden, önce mekanizmanın ana eksenini oluşturan alan kanıtı algoritmasını irdeleyelim. Alan kanıtı, emeğin kanıtı algoritması gibi ispatlayıcı (prover) ve onaylayıcı (verifier) isimli iki kullanıcı arasında iki aşamalı olarak gerçekleştirilen bir algoritmadır. Başlangıç aşamasında ispatlayıcı belli büyüklükteki bir F dosyasını lokal belleğine kaydeder ve ikinci aşamada kullanılmak üzere bu dosya ile ilgili bir S taahhüdünü onaylayıcı ile paylaşır. İkinci aşamada ispatlayıcının dosyayı orijinal haliyle tutup tutmadığını sorgulamak için onaylayıcı ispatlayıcıya bir C sınaması gönderir. İspatlayıcı da bu sınamanın karşılığı olarak bir A kanıtını oluşturup bunu doğrulayıcı ile paylaşır. Algoritmanın sonunda doğrulayıcı, doğruluğuna bağlı olarak kanıtı ya kabul eder ya da reddeder. Etkin ve güvenli bir alan kanıtı algoritmasında S taahhüdünün dosyaya nazaran çok daha küçük boyutta olması ve onaylayıcının A kanıtını etkili ve hızlı bir şekilde doğrulaması beklenir. Algoritmada aynı zamanda hilekar bir onaylayıcının S taahhüdünü oluşturduktan sonra dosyayı silebileceği göz önünde bulundurulmalıdır. Dziembowski vd. [34] Merkle ağaç yapısı kullanan etkili ve güvenli bir alan kanıtı algoritması sunmuşlardır. h kriptografik özet fonksiyonu olmak üzere, algoritmada N bitlik bir F dosyası için onaylayıcının tek bir özet değeri $h(.)$'ni taahhüt olarak saklaması yeterliyken, sınamalara karşı ispatlayıcının oluşturması gereken kanıt boyutu yalnızca $O(logN)$'dir.

Yukarıdaki açıklamalardan da anlaşılacağı gibi alan kanıtı algoritmasını, taraflar arasında belli oranda etkileşim gerektirdiği için blokzinciri teknolojisine doğrudan uygulamak zor olacaktır. Ayrıca emeğin kanıtı algoritmasına nazaran adaylar için blok üretim yarışını kazanmak daha kolay olacağından,

yarışı kazanma ihtimali olan birçok adaydan kazananı belirlemek daha zorlayıcı olacaktır. Park vd. [35], Dziembowski vd. [34]'nin önerdiği alan kanıtı algoritmasını kullanan ve az önce zikredilen endişeleri gideren SpaceMint adı verilen bir protokol önermişlerdir. SpaceMint protokolünde ispatlayıcı rolünü blok üretim yarışına katılmak isteyen adaylar ve onaylayıcı rolünü blokzincir ağındaki diğer bütün düğümler üstlenmektedir. Madenciler blokzincir ağına katıldıklarında, madencilik için lokal belleklerinde ayırdıkları alana dair taahhüdü alan kanıtı algoritmasında kullanılmak üzere özel bir işlem vasıtasıyla ağdaki diğer düğümlerle paylaşırlar. Yukarıda da belirtildiği gibi madencilerle ağdaki diğer düğümler arasında karşılıklı etkileşim pek mümkün gözükmediğinden, alan kanıtı algoritmasında kullanılan sınama değeri direk onaylayıcıdan değil de blokzincirin kendisinden sağlanacaktır. Burada madenciler zincire eklenmiş son bloğun özet değerini sınama değeri olarak kabul edip, onun karşılığı olarak oluşturdukları kanıtı ve paylaşılan kanıtlar arasından liderin seçilmesinde kullanılacak kanıtın kalite değerini ağla paylaşacaklardır. Sonrasında protokol kalite değerleri üzerinden lideri belirleyecektir. Protokolde kalite değerleri daha çok alan ayırmış olanların daha yüksek ihtimalle lider olarak seçilmesine olanak sağlayacak şekilde belirlenmektedir.



Şekil 2.11. SpaceMint Protokolünün Blok Yapısı

SpaceMint protokolünde bloklar özet alt-bloğu, işlem alt-bloğu ve imza alt-bloğu olmak şeklinde üç parçadan oluşmaktadır. Özet alt-bloğunda blok üretiminde kullanılan alan kanıtı ve bir önceki özet bloğunun özeti tutulmaktayken, işlem alt-bloğunda bloğa eklenen işlemler ve imza alt-bloğunda madencinin işlem alt-bloğu ve bir önceki imza alt-bloğu üzerinde oluşturduğu imzalar bulunmaktadır. Şekil 2.11'de görüldüğü gibi özet alt-blokları uç uca

eklenerek kanıt zincirini, imza ve işlem alt-blokları birbirine eklenerek imza zincirini oluşturmaktadır. Protokolde alan kanıtı algoritmasında kullanılacak olan sınama değeri kanıt zinciri üzerinden elde edilmektedir. Protokolde iki ayrı zincir tutulması yoluyla da bloğa eklenen işlemlerin, alan kanıt algoritmasında kullanılacak sınama değerinin oluşturulması sürecinin dışında tutulması sağlanmıştır. Böylelikle 'grinding' atağı gerçekleştirmek isteyen bir saldırganın bloğa eklenecek işlemleri değiştirmek suretiyle alan kanıt algoritmasına müdahale etmesi engellenmektedir.

Bu başlık altında değerlendirilebilecek olan bir diğer protokol, SpaceMint'e benzer şekilde bitcoindeki yüksek hesaplama gücü gerektiren ve ciddi elektrik sarfiyatına sebep olan lider seçim mekanizmasını depolama kapasitesine dayalı şekilde değiştirmeyi öneren Filecoin [36] uzlaşı protokolüdür. Filecoin akranların depolama alanlarını filecoin adı verilen bir akçe karşılığı birbirlerine açtıkları, merkezi olmayan bir depolama ağıdır. Bu depolama ağında bloklar alan-zaman kanıtı (proof of spacetime) algoritması üzerinden seçilen liderler tarafından üretilmektedir. Alan-zaman kanıtı, sıradaki bloğu üretecek liderleri Filecoin ağında depolama hizmeti için aktif olarak kullandıkları alan miktarına bağlı olasılıksal bir yolla belirleyen alan kanıtı algoritmasının bir formudur. Filecoin protokolünde sıradaki bloğu üretecek lider aşağıda verilen eşitsizlik üzerinden belirlenmektedir:

$$\frac{h\left(\langle t \| rast(t) \rangle_{M_i}\right)}{2^L} \leq \frac{p_i^t}{\sum_j p_j^t}$$

Eşitsizlikte t bloğun üretileceği zamanı, p_i^t de M_i kimlikli madencinin t zamanında blokzincir ağında aktif olarak kullandırdığı depolama alanı miktarını ve $\sum_j p_j^t$ blokzincir ağında aktif olarak kullanılan tüm depolama alanlarını belirtmektedir. h bir kriptografik özet fonksiyonu ve L bu özet fonksiyonunun çıktı boyutu olmak üzere, sıradaki bloğu üretmeye aday madenci M_i önce blokzincir ağından üretilebilecek herkese açık rastgele sayı $rast(t)$'yi bloğun üretileceği zaman dilimi t ile birleştirip imzalar ve sonra imzanın özet değerini alır. Sonra bulduğu değerin eşitsizliği sağlayıp sağlamadığını test eder. Eğer bulduğu değerler eşitsizliği sağlarsa kendini lider olarak ilan eder. p_i^t, yani madencinin ilgili zaman diliminde blokzincir ağında aktif olarak kullandırdığı depolama alanı miktarı büyükse, eşitsizliğin sağ tarafı da büyük olacağından, madencinin lider seçilme ihtimali artacaktır. Filecoin protokolünde Sap-

ceMint'ten farklı olarak, depolama hizmeti için aktif olarak kullanılan bellek alanları madencilik için de kullanılmaktadır.

i. Alan Kanıtı Tabanlı Uzlaşı Mekanizmalarının Değerlendirilmesi

Güvenlik. İddialarını ispatlama noktasında formel bir kanıt sunmasalar da iki protokol de, (SpaceMint ve Filecoin) blokzincir ağında madencilik için ayrılmış alanın çoğunun rasyonel olarak hareket eden dürüst düğümlerin elinde olduğu kabul edildiğinde, dürüst düğümlerin ellerindeki zincirlerin farklı olma ihtimalinin çok düşük olacağını iddia etmektedir. Blokzincirinde çifte harcama gibi saldırılar gerçekleştirmek isteyen bir saldırgan, bencil madencilik (selfish mining) adı verilen bir saldırı üzerinden ürettiği blokları gizli tutarak blok üretim yarışını kendi lehine çevirebilir ve uzun vadede ağdaki en uzun zinciri geçebilir. Alan kanıtı algoritmasında blok üretim yarışını kazanmak emeğin kanıtına nazaran daha kolay olduğundan ve en önemlisi alan kanıtı algoritmasındaki sınama değerleri zincirdeki son blok üzerinden oluşturulduğundan, saldırganlar bloğa eklenecek işlemleri değiştirerek işlerine yarayan sınama değerleri oluşturabilir ve kendilerini lider yapacak kanıtlar üretebilirler. SpaceMint protokolü bu tür saldırıları engellemek adına, işlemleri ayrı bir zincire ekleyerek sınama değeri oluşturulması sürecinin dışında tutmuştur. Ayrıca bencil madencilik saldırısını engelleyebilmek adına, sınama değerlerinin oluşumunda zincirdeki son bloğun değil, güvenlik değeri olarak kabul edilen bir K sayısı kadar önceki bloğun kullanılması tavsiye edilmiştir.

Yine blok üretmek emeğin kanıtı algoritmasına göre daha kolay olduğu için madenciler kârını artırmak adına ürettikleri bloklar yoluyla alternatif zincirlere yatırım yapmak isteyebilirler. SpaceMint protokolü böyle durumları ceza işlemleri yoluyla engellemektedir. Blokzincir ağında düğümler, aynı madenci tarafından oluşturulmuş aynı yüksekliğe sahip iki farklı blok fark ettiklerinde bunu ceza işlemi yoluyla diğer düğümlere duyururlar. Bu durumda protokol üretilen bu bloğun ödülünün yarısını bu durumu fark edip bildiren düğüme vermektedir. Ödülün diğer yarısı düğümler kendilerini ihbar etmesinler diye yakılmaktadır.

Teşvik. SpaceMint ve Filecoin protokollerinde de blok üreticileri teşvik için belli miktar akçe ve bloklara eklenen işlemlerden elde edilen kesintilerle ödüllendirilmektedirler. Dolayısıyla protokolde akçe üretimi blokların oluşmasıyla paralel olacaktır.

Tablo 2.1. Çekiliş Tabanlı Uzlaşı Mekanizmalarının Karşılaştırılması

protokol	lider seçimi	teşvik	hata toleransı	verimlilik	dezavantaj
Bitcoin	emeğin kanıtı	belli miktar akçe + kesintiler	toplam hash gücünün yarısından azı	7 işlem/sn	yüksek elektrik tüketimi
Ghost	emeğin kanıtı	belli miktar akçe + kesintiler	toplam hash gücünün yarısından azı	15 işlem/sn	yüksek elektrik tüketimi
Bitcoin-NG	emeğin kanıtı	belli miktar akçe + kesintiler	toplam hash gücünün yarısından azı	-	yüksek elektrik tüketimi
Peercoin	emeğin kanıtı ve hisse kanıtı	belli miktar akçe + kesintiler	toplam akçenin yarısından azı	7 işlem/sn	akçe yaşı
Nxt	hisse kanıtı	kesintiler	toplam akçenin yarısından azı	100 işlem/sn	başlangıç akçe dağılımı
CoA	hisse kanıtı	kesintiler	toplam akçenin yarısından azı	-	lider seçiminin manipülasyona açık olması
Ouroboros	hisse kanıtı	kesintiler	toplam akçenin yarısından azı	257 işlem/sn	başlangıç akçe dağılımı
Tron	yetkilendirilmiş hisse kanıtı	kesintiler	toplam akçenin yarısından azı	2000 işlem/sn	başlangıç akçe dağılımı
Tezos	yetkilendirilmiş hisse kanıtı	kesintiler + enflasyondan pay	toplam akçenin yarısından azı	40 işlem/sn	başlangıç akçe dağılımı
SpaceMint	alan kanıtı	belli miktar akçe + kesintiler	toplam depolama alanının yarısından azı	-	sıfır hisse problemi
Filecoin	alan kanıtı	belli miktar akçe + kesintiler	toplam depolama alanının yarısından azı	-	emeğin kanıtı algoritmasına bağımlı olması

2.2.1.4. Çekili Tabanlı Uzlaşı Mekanizmalarının Değerlendirilmesi

Bu bölümde incelenen uzlaşı mekanizmaları, lider seçimini belirleyen algoritma (lider seçimi), protokolde kullanılan teşvik mekanizması (teşvik), protokolün tolere edebildiği kötü niyetli düğümlerin oranı (hata toleransı), protokolün saniyede onaylayıp zincire ekleyebildiği işlem sayısı (verimlilik) ve protokolün eksik veya dezavantajlı olduğu yönü (dezavantaj) olmak üzere 5 kriter üzerinden değerlendirilecek ve kıyaslanacaktır. Burada incelenen uzla-

şı mekanizmaları için (Nakamoto uzlaşı protokolü ve Ouroboros protokolü hariç), bu mekanizmaları öneren yazarlar, protokollerin belli oranda kötücül düğümün varlığında güvenlik ve canlılık özelliklerini sağlayıp sağlamadığına dair formel bir kanıt sunmamışlardır. Yazarlar ya protokollerin belli saldırılara karşı dirençli olduğunu göstermişler ya da güvenlik analizi konusunu göz ardı etmişlerdir. Diğer yandan buradaki protokollerin hepsi için bloğun neticelendirmesi (finalization) diye adlandırılan, onaylanmış blokların üzerinde mutabık kalınması olasılıksal bir yolla yapıldığı için bu kriter tabloya eklenmemiştir. Ayrıca, Bitcoin-NG, CoA, SpaceMint ve Filecoin protokollerinin verimliliği için literatürde güvenilir bir kaynak bulunamamıştır. Çekiliş tabanlı uzlaşı mekanizmalarının yukarıda zikredilen kriterler yoluyla değerlendirilmesi Tablo 2.1'de verilmiştir.

2.2.2. Oylama Tabanlı Uzlaşı Mekanizmaları

Çekiliş tabanlı uzlaşı mekanizmalarına kıyasla oylama tabanlı uzlaşı mekanizmalarında lider seçim süreci Round-Rabin gibi görece çok daha kolay yöntemlerle belirlenmektedir. Bu yüzden, oylama tabanlı uzlaşı mekanizmaları çoğunlukla güvenin daha kolay tesis edilebildiği izinli blokzinciri platformlarında kullanılmaktadırlar. Diğer yandan lider olarak seçilmiş düğümlerin ürettiği bloklar, geleneksel uzlaşı protokollerindekine benzer bir onay mekanizmasıyla geçerli kabul edilip zincire eklenmektedir. Dolayısıyla çekiliş tabanlı olanların aksine oylama tabanlı uzlaşı mekanizmalarında blok neticelendirilmesi deterministik bir yolla yapılıyor olacaktır. Bu başlık altında, İntel ve JP Morgan gibi çok uluslu şirketlerin desteğiyle geliştirilmiş açık kaynak kodlu Hyperledger ve Quorum projelerinde uygulanan Kafka, FBHT, Raft, İBHT uzlaşı protokolleri ile SMaRtChain ve Ripple protokolleri incelenecektir.

2.2.2.1. Hyperledger

Hyperledger, Linux Foundation'ın öncülüğünde ve IBM, Intel gibi firmaların desteğiyle geliştirilmiş, blokzincir teknolojisinin günlük pratiğe adaptasyonunu amaçlayan açık kaynak kodlu blokzincir tabanlı bir projedir. Günlük pratikte, farklı iş kolları farklı gereksinimleri öne çıkardığından, Hyperledger

farklı uzlaşı mekanizmalarının uygulandığı alternatif çerçeveler sunarak bu gereksinimleri karşılama noktasında daha modüler bir yapı ortaya koymaktadır. Burada Hyperledger projesinde yaygın olarak kullanılan iki uzlaşı mekanizması incelenecektir.

Kafka. Protokol ilk olarak Kreps [37] tarafından, verileri mobil cihazlar, sensörler, veri tabanları, bulut yapıları gibi kaynaklardan eş zamanlı olarak toplayabilmek ve dağıtabilmek amacıyla dağıtık bir mesaj sistemi olarak literatüre sunulmuştur. Protokol, Hyperledger Fabric çerçevesinde üretilen blokların ağdaki bütün düğümlerce aynı sıra gözetilerek onaylanıp zincire eklenmesini sağlamak amacıyla bir uzlaşı protokolü olarak kullanılmaktadır. Hyperledger Fabric uyarlamasında [38], Raft protokolünde olduğu gibi zaman parça (partititon) adı verilen dilimlere ayrılmış olarak ele alınmaktadır. Her bir zaman parçasında, deterministik bir yolla belirlenen lider, kendisine ulaşan talepler üzerinden yeni bir blok oluşturmaktan ve bu bloğu onaylayıp zincire eklemek için kullanılacak protokolü yürütmekten sorumludur. Protokol iki aşamadan oluşmaktadır. İlk aşamada lider kendisine ulaşmış talepleri belli bir sıra gözeterek yeni bir bloğa yazar ve bu yeni bloğu diğer düğümlerle paylaşır. İkinci aşamada yeni bloğu alan her düğüm önce bloğun içerisine yazılan taleplerin geçerliliğini kontrol eder. Bu noktada geçersiz olan talepler bloktan çıkarılmaz, sadece geçersiz diye işaretlenir. Sonrasında düğümler ilgili zaman parçasının liderine, gönderilen yeni bloğu onayladıklarına dair bir mesaj gönderirler. Lider aktif olan düğümlerin yarıdan fazlasından benzer mesaj almışsa, ilgili bloğa taahhütte bulunduğuna dair diğer düğümlere bir mesaj gönderir ve bloğu zincire ekler. Protokolde liderler, diğer düğümlerin aktif olup olmadıklarını kontrol ederek, aktif olmayanları protokol dışında tutmaktadırlar. Kafka protokolü protokole katılan düğümlerin yarıdan fazlasının çevrim içi olduğu varsayıldığında güvenlik ve canlılık özelliklerini sağlamaktadır.

FBHT. Kullanıcılara kendi dijital kimliklerini oluşturup yönetebilme imkanı sunan bir blokzincir yapısı olan Hyperledger Indy [39], uzlaşı mekanizması olarak Bölüm 2.1.4.'te incelenen FBFT protokolünün bir varyantını kullanmaktadır. FBHT protokolünün bu varyantında, her bir pencere için önceden belirlenmiş liderler kendilerine ulaşmış taleplerden belli bir kısmını gruplayarak ön-hazırlık mesajını oluşturur ve bu mesajı blokzincir ağıyla paylaşırlar.

Diğer düğümler ön-hazırlık mesajındaki talepleri kabul ederlerse bunu hazırlık mesajı ile blokzincirdeki diğer düğümlere duyururlar. Bu noktada düğümler, f protokolün tolere edebileceği Bizans düğümü sayısı olmak üzere, en az $2f$ hazırlık mesajı almışlarsa mesaj üzerinden ilgili taleplere taahhüt ederler ve bunu taahhüt mesajı üzerinden blokzincir ağıyla paylaşırlar. Son aşamada düğümler en az $2f$ taahhüt mesajı almışlarsa, ilgili talepleri geçerli sayıp blok haline getirerek zincire eklerler.

FBHT protokolünde liderlerin yönettiği tek bir icra yerine, farklı düğümlerin lider olarak yönettiği çoklu icralar paralel olarak yürütülmektedir. Protokolde diğer icralar, verimlilik üzerinden liderin yönettiği baş icranın en etkili icra olup olmadığını tespit etmeye yarayan destek icraları olarak yürütülmektedirler. Benzer şekilde bu varyantta da, herhangi bir pencere için baş icra yeterince iyi değilse, düğümler icra değişim protokolü üzerinden baş icrayı daha verimli olanıyla değiştirirler. Hyperledger Indy projenin diğer çerçevelerinde olduğu gibi çoğaltılmış durum makinesi işlevi görmektedir. Kısaca istemciler durumlarda değişiklik yapılması için düğümlerden taleplerde bulunur; düğümler de bu talepler ile amaçlanan değişiklikleri durumlara yansıtırlar. Protokolün sonunda düğümler, taahhüt ettikleri taleplerle beraber ilgili değişiklikleri bloğa yazıp zincire eklerler. n toplam düğüm sayısı olmak üzere, hatalı düğümlerin sayısının $\lfloor(n-1)/3\rfloor$ 'den fazla olmadığı kabul edildiğinde protokol güvenlik ve canlılık özelliklerini sağlamaktadır.

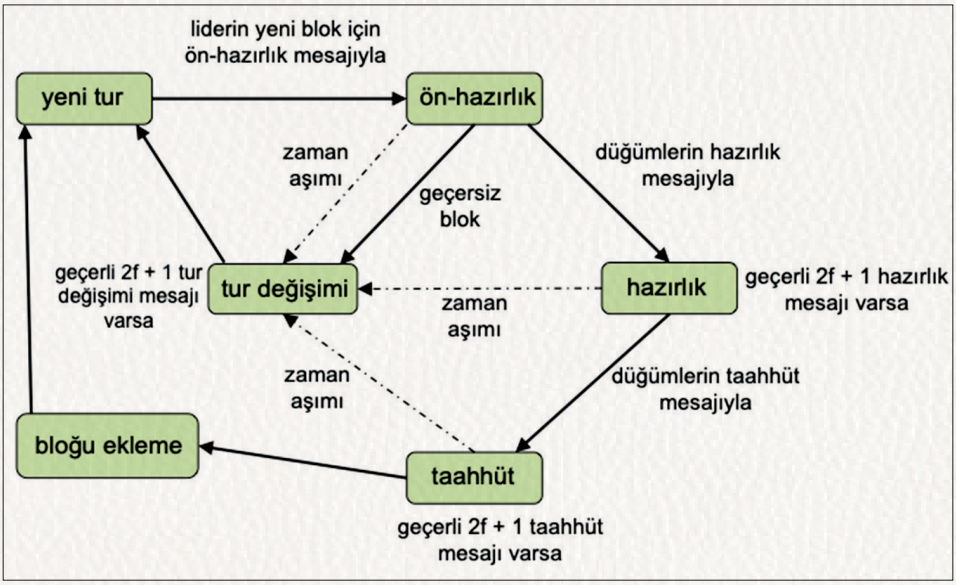
Hyperledger'ın Değerlendirilmesi. Kafka protokolü, düğümlerin yarısından azı çökme hatalı düğüm olmak koşuluyla güvenlik ve canlılık özelliklerini sağlamaktadır. Ancak protokol Bizans hatalı düğümleri tolere edememektedir. Diğer yandan FBHT protokolü, n protokoldeki toplam düğüm sayısı olmak üzere $\lfloor(n-1)/3\rfloor$'e kadar Bizans hatalı düğümleri tolere edebilmekte, güvenlik ve canlılık özelliklerini sağlayabilmektedir. Bölüm 2.1'de açıklanan FBHT ve VR gibi protokoller düğümler arası direk iletişim kanalları gerektirmektedir. Dolayısıyla bu tür protokollerin mesaj karmaşıklığı, sırasıyla asimptotik olarak $O(n^2)$ ve $O(n)$'dir. Blokzincirde ise mesajların otantisitesi dijital imza algoritmaları üzerinden doğrulanabildiğinden ve düğümler dedikodu (gossip) protokolü üzerinden iletişim kurduklarından, Kafka protokolü ve FBHT'nin mesaj karmaşıklığı $O(n)$ olacaktır.

Hyperledger Fabric üzerinde gerçekleştirilen bir simülasyonda 15 düğüm tarafında yürütülen Kafka protokolünün verimliliği 3186 işlem/sn olarak ölçülmüştür [40]. Ayrıca blokzincir ağında oluşan taleplerin, karşılanarak zincire eklenmesine kadar geçen süreyi gecikme olarak tanımlarsak; yine aynı simülasyonda Kafka protokolünde gecikme 36 sn civarı gerçekleşmektedir. Diğer yandan Hyperledger Indy üzerinde gerçekleştirilen bir simülasyonda 50 düğüm tarafında yürütülen FBHT protokolünün verimliliği 10 işlem/sn olarak belirlenmiştir [41].

2.2.2.2. Quorum

Quorum [42], Ethereum platformunun izinli blokzincir yapısına bir uygulaması olarak JPMorgan Chase şirketi tarafından geliştirilmiş bir platformdur. İzinli blokzincir olması sebebiyle, zincirin güvenli bir şekilde idame ettirilmesi noktasında kullanıcılar, Ethereum ve bitcoin gibi izinsiz emsallerine nazaran verimliliği daha yüksek ve görece daha az karmaşık uzlaşı mekanizmaları üzerinden mutabakata varabilmektedirler. Platform periyodik olarak belirli liderler yönetiminde uzlaşı protokolleri üzerinden güncellenen çoğaltılmış durum makinesi olarak çalışmaktadır. Quorum'da yalnızca çökme hatalı düğümleri tolere edebilen Raft uzlaşı protokolü ve Bizans hatalı düğümleri de tolere edebilen İstanbul Bizans hata toleransı protokolü olmak üzere yaygın olarak iki uzlaşı protokolü kullanılmaktadır.

Raft. Bölüm 2.1.2'de etraflıca incelendiği için burada sadece protokolün blokzincir dünyasına nasıl adapte edildiğinden bahsedilecektir. Kısaca protokol lider, takipçi ve aday olmak üzere üç farklı rol tarafından yürütülür. Lider sorumlu olduğu zaman diliminde kendisine gelen mesajlar üzerinden oluşturduğu bloğu takipçi ve aday düğümlerle paylaşır. Bu aşamada takipçi ve adaylar bloğu aldıktan sonra eğer geçerliyse bloğu onayladıklarına dair mesajı lidere gönderirler. Eğer düğümlerin çoğundan onay mesajı gelirse lider bloğun geçerli olduğunu blokzincir ağındaki düğümlere bildirir ve bloğu zincire ekler. Diğer düğümler de benzer şekilde bloğu zincire eklerler. Raft protokolünde liderler, zaman diliminin başında gerçekleştirilen bir seçim yoluyla belirlenir. Klasik Raft protokolünden farklı olarak, liderler çevrim dışı adayları veya takipçileri protokolün uygulayıcısı komiteden düşürebilir veya komiteye yeni adaylar ekleyebilirler.



Şekil 2.12. İstanbul Bizans Hata Toleransı Protokolü

İstanbul Bizans Hata Toleransı (İBHT). Protokol [43], Bölüm 2.1.3'de tetkik edilen PBHT protokolünün blokzinciri teknolojisine uygulanan bir varyantıdır. PBHT'ye benzer bir biçimde normal bir İBHT protokolü ön-hazırlık, hazırlık ve taahhüt isimli 3 aşamadan oluşmaktadır. Ön-hazırlık aşamasında lider ağ üzerinden kendisine ulaşmış mesajları kullanarak oluşturduğu bloğu diğer düğümlerle paylaşır. Hazırlık aşamasında düğümler geçerliliğini kontrol ettikten sonra bloğu diğer düğümlerle paylaşırlar. Taahhüt aşamasında her düğüm, n protokoldeki toplam düğüm sayısı olmak üzere kendisine ulaşan $\lfloor 2(n-1)/3 \rfloor$ geçerli hazırlık mesajı varsa ilgili bloğa taahhütte bulunur ve bloğu içeren bir taahhüt mesajını diğer düğümlerle paylaşır. Son olarak düğümler ilgili blok için $\lfloor 2(n-1)/3 \rfloor + 1$ geçerli taahhüt mesajı elde ederlerse bloğu onaylayıp zincire eklerler. İBHT protokolünde liderler her zaman dilimi için deterministik bir yolla belirlenmektedir. Şekil 2.12'de görüldüğü gibi İBHT protokolünde, her aşama için belirlenen sürede gereken mesaj sayısına ulaşılmadığında ya da lider tarafından geçerli olmayan bir blok gönderildiğinde düğümler tur değişimi yoluyla mevcut lideri değiştirebilmektedirler. Burada belirlenen sürenin aşılıp aşılmadığını tespit etmek için düğümler her aşamanın başında bir kronometre çalıştırırlar. Belirlenen süre aşılmışsa düğümler

blokzincir ağıyla tur değişimini talep eden bir mesaj paylaşırlar. Eğer düğümler $\lfloor 2(n-1)/3 \rfloor + 1$ geçerli tur değişimi mesajı alırlarsa yeni tur aşaması üzerinden yeni lideri belirleyip bir sonraki tura geçerler.

Quorum'un Değerlendirilmesi. Raft protokolü düğümlerin yarısından azı çökme hatalı düğüm olmak koşuluyla güvenlik ve canlılık özelliklerini sağlamaktadır. Ancak protokol sadece çökme hatalı düğümleri tolere edebilmektedir. Diğer yandan İBHT protokolü, n protokoldeki toplam düğüm sayısı olmak üzere $\lfloor (n-1)/3 \rfloor$'e kadar Bizans hatalı düğümleri tolere edebilmekte, güvenlik ve canlılık özelliklerini sağlayabilmektedir. Ancak iki protokol de güvenlik özelliğini kısmi eş zamanlı ağ modelinde sağlayabiliyorken, canlılık özelliğini ancak eş zamanlı ağ modelinde karşılayabilmektedir. Ayrıca iki protokol için de mesaj karmaşıklığı asimptotoik olarak $O(n)$ olacaktır.

Baliga vd. [44] geliştirdikleri bir simülasyon üzerinden İBHT ve Raft protokollerinin performanslarını analiz etmişlerdir. Simülasyonda İBHT ve Raft protokolleri sırasıyla 4 ve 3 düğüm tarafından yürütülmektedir. İki protokolde de blok üretim aralığı 1 saniye olarak belirlendiğinde, Raft protokolü saniyede 750 işlemi onaylayarak bloklara ekleyebilmekteyken, İBFT protokolünde bu sayı 600 civarı olarak ölçülmektedir. Blokzincir ağındaki işlem yükü arttıkça görece İBFT protokolünün performansı azalmaktadır. Ayrıca blokzincir ağında oluşan taleplerin, karşılanarak zincire eklenmesine kadar geçen süreyi gecikme olarak tanımlarsak; Raft protokolünde gecikme 1.5 sn civarı gerçekleşirken, İBFT'de gecikme 2.8 sn civarı gözlemlenmiştir. Yine benzer bir çalışmada [45] verimilik protokoldeki düğüm sayısı 20 olarak belirlendiğinde 600 işlem/sn olarak ölçülmüştür. Ayrıca gecikme de aynı düğüm sayısı baz alındığında 5 saniyeye kadar çıkmaktadır.

2.2.2.3. SMaRtChain

Bizans hata toleransı protokolleri, tutarlı bir zincir elde edilmesine imkan sağladığından blokzincir tabanlı platformlarda uzlaşı mekanizmaları olarak değerlendirilmektedirler. Bununla birlikte, katılımcıların sayısı ile protokollerin performansı ters orantılı olduğundan, geleneksel BHT protokolleri için ölçeklenebilme problemi ciddi sorun teşkil etmektedir. Bunun yanı sıra protokollerin uygulanmasını zorlaştıran blokzincirin yapısından kaynaklı bir takım

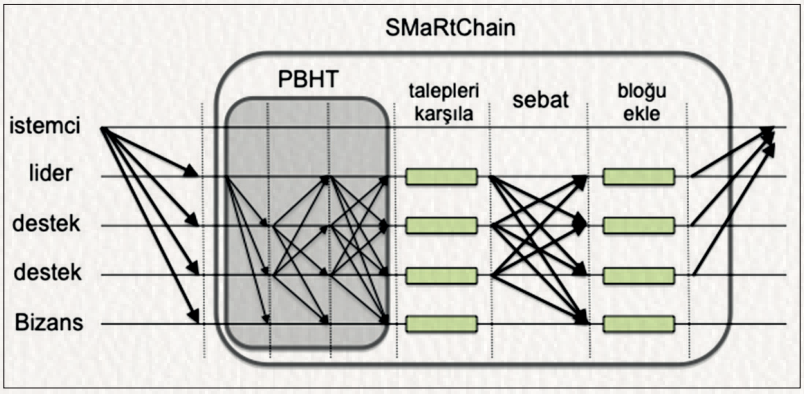
kısıtlamalar da mevcuttur. Örneğin blokzincirde, zincire eklenmiş işlemlerin denetlenmesi ya da üçüncü şahıslarca doğrulanması gerekebilir. Ya da kullanıcıların protokolü yürütecek komiteye dinamik bir biçimde dahil olması ve komiteden çıkarılması gerekebilir. SmartChain [46], yukarıda altı çizilen problemlerin giderilmesi amacıyla BFT-SMaRt uzlaşı protokolü üzerinde geliştirilmiş bir blokzincir platformudur. Burada Smartchain platformunun incelemesine geçmeden önce, altyapı oluşturması açısından kısaca BFT-SMaRt uzlaşı protokolü açıklanacaktır.

BFT-SMaRt. 2014 yılında çoğaltılmış durum makineleri için Bessani vd. [47] tarafından geliştirilmiş bir protokoldür. Protokolün normal işleyişi PBHT protokolüne benzer 3 aşamalı olarak gerçekleşmektedir. Teklif aşamasında lider kendisine ulaşmış olan bir grup talebi, ilgili pencerede karşılanıp kütük kayıtlarına eklenmesi gereken işlemler olarak diğer kopyalara gönderir. Yazım adı verilen ikinci aşamada diğer kopyalar taleplerin geçerliliğini kontrol ettikten sonra, taleplerin kriptografik özetlerini içeren mesajı diğer düğümlere gönderirler. Kabul adı verilen son aşamada ise, kopyalar tüm kopyaların üçte ikisinin fazlasından aynı özet değerlerini içeren yazım mesajı almışlarsa, diğer kopyalara ilgili özet değerlerini içeren kabul mesajı gönderirler. Protokolün sonunda kopyalar tüm kopyaların üçte ikisinin çoğundan aynı özet değerini içeren kabul mesajı almışlarsa, ilgili talepleri onaylayıp kütük kayıtlarına eklerler.

BFT-SMaRt protokolünde, lider kopya hatalı davrandığında ya da iletişim ağından kaynaklı senkronizasyon problemleri yaşandığında diğer kopyalar, yeni bir lider belirleyerek protokolü bir sonraki zaman dilimine taşımaktadırlar. Ayrıca, çöken kopyalar tekrar çevrim içi olduklarında diğer kopyalardan kütük kayıtlarının son halini talep ederek kendi kütük kayıtlarını güncelleyebilmektedirler. Protokol yeni kopyaların sisteme dahil olarak protokole katılmalarına da imkan vermektedir. Kopyaların oluşturduğu kümedeki güncellemeler ‘pencere yöneticisi’ (view manager) adı verilen güvenilir bir istemci tarafından gerçekleştirilmektedir. Burada pencere yöneticisinin yeni bir kopya için oluşturduğu katılım talebi diğer taleplere benzer bir biçimde değerlendirilir. İlgili talep onaylanıp kütük kayıtlarına eklendiğinde yeni kopya sistemin bir parçası olur ve protokole katılmaya hak kazanır.

SMaRtChain. Platformda klasik bir çağıltılmış durum makinesinde olduğu gibi istemci ve kopya olmak üzere iki tür kullanıcı vardır. İstemciler durum

makinesine yansıtılmasını istedikleri güncelleme taleplerini sistemin yürütülmesinden sorumlu kopyalara gönderirler. Platformda iletişim kullanıcılar arasındaki ikili iletişim kanalları yoluyla yapılmaktadır. Platformun kurulumunda ilk komiteyi oluşturacak kopyaların kimlikleri başlangıç bloğuna yerleştirilir. Sonrasında yeni bir kopya platforma katılmak isterse, halihazırda platformda yer alan kopyalara bir talep gönderir. Eğer o an platformda yer alan kopyaların üçte ikisinin fazlasından talebinin onaylandığına dair imzalı bir mesaj alırsa, yeni kopya bir istemci gibi hareket ederek lider kopyadan bu imzalarla beraber bir talepte bulunur. Bu taleple beraber lider, komite güncellemesi için yeni bir blok oluşturur ve BFTSmart protokolü üzerinden ilgili blokzincire eklenir. Ayrıca hatalı davrandığı tespit edilen bir kopya yine benzer bir yolla komiteden çıkarılır.



Şekil 2.13. SmaRtChain Protokolü

Smartchain'de lider kopyalar kendi zaman diliminde topladıkları talepleri yeni bir bloğa yazarak diğer kopyalara gönderirler. Sonrasında kopyalar BFTSmart protokolü üzerinden aldıkları blokları onaylar ve bloğun içerdiği talepleri yerine getirirler. Şekil 2.13'te görüldüğü gibi Smartchain'de, BFTSmart protokolünden farklı olarak bir aşama daha gerçekleştirilir. Sebat (persist) adı verilen bu aşamada çatallanmayı engellemek adına kopyalar, o an platformda yer alan kopyaların üçte ikisinin fazlasından blok için alınmış imzaları bloğun sertifikası olarak bloğa dahil eder ve bloğu zincire eklerler. Smartchain'de bloklar, başlık, gövde ve sertifika şeklinde üç bölümden oluşmaktadır. Başlık, bloğun indeksini, en sonki komite güncellenmesinin yer aldığı bloğun indek-

sini, en son kontrol noktasının oluşturulduğu bloğun indeksini, blokta yer alan talepler ve bu taleplere karşılık gelen sonuçlar için oluşturulmuş Merkle ağaçlarının köklerini ve bir önceki bloğun özetini içermektedir. Bunun yanı sıra gövde kısmı bloğa eklenen talepleri ve bu taleplerin sonuçlarını içeriken, sertifika kısmı aynı indekse sahip başka bir bloğun zincire eklenmemesini garantilemek adına, ilgili bloğun başlığı için o anda platformda yer alan kopyaların üçte ikisinin fazlasından alınmış imzayı içermektedir.

SMaRtChain'in Değerlendirmesi. Smartchain platformu, uzlaşı protokolünün yürütüldüğü zaman dilimindeki komite üyelerinin üçte ikisinden fazlası dürüst ise güvenlik ve canlılık özelliklerini sağlamaktadır. Yalnız hatalı düğümler, komiteden çıkarılmış eski kopyalarla birlikte hareket ederek (bu noktada protokolün tolere ettiği eşik değerini aşacaklardır), eski kopyaların çıkarıldığı komite güncellenmesinin öncesinden başlayan bir çatallanma oluşturabilirler. Smartchain bu durumu engellemek adına, dijital imza anahtarında kullanılacak anahtar çiftlerini, düğümlerin sisteme dahil olurken oluşturdukları anahtar çiftinden her zaman dilimi için yeniden üretmektedir. Smartchain platformu düğümler arası direk iletişim kanalları gerektirmektedir. Dolayısıyla BFTSmart protokolünün mesaj karmaşıklığı asimptotik olarak $O(n^2)$'dir. Diğer yandan, SMaRtChain üzerinde gerçekleştirilen bir simülasyonda 14 düğüm tarafında yürütülen BFTSmart protokolünün verimliliği yaklaşık olarak 14500 işlem/sn olarak ölçülmüştür [46]. Ayrıca yine aynı simülasyonda BFTSmart protokolünde gecikme 0.2 sn civarı gerçekleşmektedir. Protokole sebat aşaması eklendiğinde verimlilik 12500 işlem/sn olarak ölçülürken, gecikme de 0.21 sn olarak gerçekleşmektedir.

2.2.2.4. Ripple Uzlaşı Protokolü

Ripple, Schwartz vd. [48] tarafından çoğaltılmış durum makinesi olarak geliştirilmiş bir protokoldür. Burada çoğaltılmış durum, eşten-eşe (peer-to-peer) ağ yoluyla iletişim kuran düğümler tarafından idame ettirilen bir dağıtık defter sistemidir. Klasik çoğaltılmış durum makinelerinde olduğu gibi, istemciler dağıtık deftere yansıtılmasını istedikleri durum değişikliklerini işlemler yoluyla düğümlere bildirir. Düğümler de uzlaşı protokolu üzerinden ilgili değişiklikleri dağıtık deftere uygulayarak defteri günceller. Protokolde her i düğümü, güvendiği ve protokol esnasında kulak kabarttığı düğümlerden oluşan

sadece ona has DL_i düğüm listesi tutmaktadır. Düğümler bu listeye istedikleri düğümleri ekleyebilirler. Ayrıca herhangi iki düğüm arasındaki güven karşılıklı olmak zorunda değildir. Diğer bir ifadeyle, i ve j düğümleri için, $i \in DL_j$ iken $j \notin DL_i$ olabilir.

Dağıtık defter sisteminde her düğüm ilgili defterin bir kopyasını tutmaktadır. Protokolde zaman önceki protokollerde olduğu gibi zaman dilimi adı verilen parçalara bölünmüştür. Ancak öncekilerden farklı olarak zaman dilimleri için belirlenmiş protokolün yürütülmesinden sorumlu liderler yoktur. Protokolde her düğüm istemcilerden ya da kendi listesindeki düğümlerden kendisine ulaşmış olan bir grup işlemi yeterli sayıda düğümün onayından sonra kendi kopyasına eklemektedir. Ripple protokolünde amaç, dağıtık defterin bütün kopyalarının sistemin kurulumundan itibaren aynı sıralamaya sahip işlemler üzerinden güncellenmesini sağlamaktır.

Protokolde deftere eklenecek (ya da defterin güncellemesinde kullanılacak) işlemler müzakere yoluyla (deliberation) teklif edilmektedir. Müzakerede her i düğümü istemcilerden veya diğer düğümlerden elde ettikleri işlemlerin bir kümesini protokoldeki diğer düğümlerle imzalı bir teklif mesajı üzerinden paylaşırlar. Protokolde teklif; T teklif edilen işlemlerin kümesi, r ilgili zaman diliminin indeksi, L işlemlerin uygulanacağı defterin son güncellemesi ve i teklifi yapan düğümün kimliği olmak üzere, $\langle T, r, L, i \rangle_{\sigma_i}$ formunda yapılmaktadır. Ayrıca i düğümü, DL_i listesindeki düğümlerden gelen teklif mesajları üzerinden sürekli kendi teklifini yeniler. Bu noktada i düğümü listesindeki düğümlerin belli bir eşik değerinden fazlasından aynı T işlem kümesi ve aynı L defter güncellemesi için teklif mesajı alırsa, T işlem kümesine taahhüt eder. Sonrasında taahhüt ettiği işlemler üzerinden kendi kopyasını günceller ve ilgili işlemleri diğer düğümlerle paylaşır. Protokolde her i düğümü, kendi listelerindeki düğümlerin belli bir eşik değerinden fazlasından aynı T işlem kümesi ve aynı L defter güncellemesi için taahhüt mesajı almışsa, ilgili işlemleri dağıtık deftere yansıtarak kendi kopyasını günceller. Burada paylaşılan her işlem kümesi için, kümenin ihtiva ettiği işlemlerin dürüst düğümlerce aynı sıra gözetilerek dağıtık deftere eklendiği varsayılacaktır.

Ripple protokolünde; ağda yaşanan gecikmeler ve bazı düğümlerin Bizans hatalı düğümler olması gibi nedenlerden dolayı, aynı zaman dilimi için farklı

güncellemelerin yapıldığı dağıtık defterin birbiriyle çelişen kopyaları oluşabilmektedir. Protokolde düğümler daha tutarlı bir dağıtık defter sistemi yürütebilmek adına belirli bir kural üzerinden alternatif defter kopyalarından birini tercih ederler. Bahsi geçen kuralı vermeden önce kuralla ilintili bir kaç kavramı açıklamak yerinde olacaktır. Protokolde dağıtık defterin bir güncellemesinin destek sayısı, ilgili güncellemeyi kendi kopyalarındaki son onaylanmış güncelleme kabul eden düğümlerin sayısı olarak tanımlanmaktadır. Ayrıca herhangi bir güncellemenin kol (branch) destek sayısı, ilgili güncellemeyi ya da bu güncellemeden sonraki güncellemeleri kendi kopyalarındaki son onaylanmış güncelleme kabul eden düğümlerin sayısıdır. Son olarak herhangi bir s zaman dilimi için kararsız (uncommitted) destek sayısı, s'den ya da ağda paylaşılmış en son güncellemenin yapıldığı zaman diliminden öncesinde yapılmış güncellemeleri kendi kopyalarındaki son onaylanmış güncelleme kabul eden düğümlerin sayısıdır. Bu bilgiler ışığında protokolde düğümler; birbiriyle çelişen alternatif güncellemeler olduğunda, alternatif güncellemeler arasından birini sıradaki kurala göre tercih edeceklerdir. Düğümler son onaylanmış güncellemeler arasından ortak olandan başlayarak, bir sonraki zaman dilimindeki alternatifler arasından kol destek sayısı daha yüksek olan güncellemeyi tercih ederler. Eğer kol destek sayısı daha yüksek olan bir alternatif yoksa tercih edilen son güncellemede karar kılarlar. n_i ilgili düğümün listesindeki toplam düğüm sayısı olmak üzere, i düğümünün herhangi bir işlem kümesine taahhütte bulunması için listesindeki düğümlerden en az $q_i = \lceil 0.8n_i \rceil$ kadarıyla fikir birliğine varması gerekmektedir. Diğer bir ifadeyle protokol her DL_i düğüm listesi için en fazla $n_i - q_i$ kadar Bizans hatalı düğümü tolere edebilmektedir.

Ripple Protokolünün Değerlendirmesi. $q_i = \lceil 0.8n_i \rceil$ olarak belirlendiğinde protokolün güvenlik özelliğini sağlayabilmesi için, diğer bir ifadeyle birbiriyle çelişen alternatif güncellemelerin oluşmaması için, protokole katılan her i,j düğüm çifti için $|DL_i - DL_j| \geq 2.max\{n_i - q_i, n_j - q_j\}$ şartının sağlanması gerekmektedir [49]. Ripple protokolü halihazırda Ripple Labs tarafından geliştirilmiş dağıtık bir ödeme sistemi olarak faaliyet yürüten bir platformda uzlaşı protokolü olarak kullanılmaktadır [50]. Platformda protokolü yürüten 150 civarı düğüm varken protokolün verimliliği 1500 işlem/sn olarak ölçülmektedir. k protokoldeki düğüm listelerinin boyutu ve n toplam düğüm sayısı olmak üzere, Ripple protokolünün mesaj karmaşıklığı $O(k.n)$'dir [51].

2.2.2.5. Oylama Tabanlı Uzlaşı Mekanizmalarının Değerlendirilmesi

Bu bölümde incelenen uzlaşı mekanizmaları; liderin nasıl belirlendiği (lider seçimi), protokolün tolere edebildiği kötü niyetli düğümlerin oranı (hata toleransı), protokolün hangi tür hataları tolere edebildiği (hata türü), protokolün mesaj karmaşıklığı (karmaşıklık), protokolün saniyede onaylayıp zincire ekleyebildiği işlem sayısı (verimlilik), blokzincir ağında oluşan taleplerin karşılanarak zincire eklenmesine kadar geçen süre (gecikme) ve protokolü yürütecek komitenin dinamik mi yoksa statik mi olduğu (komite formasyonu) olmak üzere 7 kriter üzerinden değerlendirilecek ve kıyaslanacaktır. Buradaki protokollerin hepsi için bloğun neticelendirmesi (finalization) diye adlandırılan, onaylanmış blokların üzerinde mutabık kalınması deterministik bir yolla yapıldığından bu kriter tabloya eklenmemiştir. Ayrıca oylama tabanlı uzlaşı mekanizmalarında protokole katılan düğüm sayısı verimliliği doğrudan etkilediği için, verimlilik başlığı altında saniyede kaç işlemin onaylandığının yanı sıra, kaç düğüm üzerinden ilgili verimliliğin elde edildiği de tabloya eklenmiştir. Diğer yandan, FBHT ve Ripple protokollerinin verimliliği için literatürde güvenilir bir kaynak bulunamamıştır. Oylama tabanlı uzlaşı mekanizmalarının yukarıda zikredilen kriterler yoluyla değerlendirilmesi Tablo 2.2'de verilmiştir.

Tablo 2.2. Oylama Tabanlı Uzlaşı Mekanizmalarının Karşılaştırılması

protokol	lider seçimi	hata toleransı	hata türü	karmaşıklık	verimlilik	gecikme	komite formasyonu
Kafka	oylama	toplam düğümlerin yarısından azı	çökme	O(n)	15 düğüm 3186 işlem/sn	36 sn	dinamik
FBHT	Round-Rabin	toplam düğümlerin üçte birinden azı	Bizans	O(n)	50 düğüm 10 işlem/sn	-	statik
Raft	oylama	toplam düğümlerin yarısından azı	çökme	O(n)	3 düğüm 750 işlem/sn	1.5 sn	statik
İBHT	Round-Rabin	toplam düğümlerin üçte birinden azı	Bizans	O(n)	20 düğüm 600 işlem/sn	5 sn	statik
BFTSMaRt	Round-Rabin	toplam düğümlerin üçte birinden azı	Bizans	$O(n^2)$	14 düğüm 12500 işlem/sn	0.21 sn	dinamik
Ripple	lider yok	toplam düğümlerin beşte birinden azı	Bizans	O(k.n)	150 düğüm 1500 işlem/sn	-	dinamik

2.2.3. Hibrit Uzlaşı Mekanizmaları

Çekiliş tabanlı uzlaşı mekanizmalarında lider olarak seçilen düğümlerin ürettiği yeni bloklar, diğer düğümlerin onayına başvurulmaksızın geçerli kabul edilip zincire eklenmektedir. Bu durum, örneğin bitcoinde, emeğin kanıtı algoritması yoluyla aynı anda lider olabilen iki farklı düğümün ürettiği aynı yüksekliğe sahip iki farklı bloğun, blokzinciri ağındaki farklı düğümlerce geçerli kabul edilmesine ve dolayısıyla çatallanma diye adlandırılan birbirine alternatif iki farklı zincirin blokzinciri ağında var olabilmesine sebebiyet vermektedir. Ya da örneğin Ouroboros protokolünde, hisse kanıtı algoritması yoluyla lider olarak seçilmiş düğümün, blokzincir ağındaki alternatif zincirlere eklenmek üzere farklı bloklar üretip paylaşabilmesine imkan vermektedir. Oylama tabanlı uzlaşı mekanizmalarında ise; liderlerin ürettiği bloklar, Bizans hata toleransı algoritmaları üzerinden onaylanıp zincire eklenmektedir. Ağdaki bütün düğümlerin Bizans hata toleransı algoritmaları yoluyla blokların onaylanması sürecince dahil edilmesi, blokzincir ağında alternatif zincirlerin oluşmasını engellemektedir. Bununla birlikte, Bizans hata toleransı algoritmalarının mesaj karmaşıklığı algoritmayı yürüten düğümlerin sayısı ile doğru orantılıdır. Bu durum bu tür algoritmaların büyük ölçekli platformlarda uygulanmasını güçleştirmektedir.

Hibrit uzlaşı mekanizmaları, yukarıda ifade edilen problemleri gidermek adına her iki türün olumlu yönlerini harmanlamaktadır. Bu tür uzlaşı mekanizmalarında üretilen bloklar, oylama tabanlı uzlaşı mekanizmalarında olduğu gibi Bizans hata toleransı algoritmaları yoluyla onaylanıp zincire eklenmektedir. Ancak, Bizans hata toleransı algoritmaları, ağdaki bütün düğümlerin katılımıyla değil de emeğin kanıtı ya da hisse kanıtı gibi mekanizmalar yoluyla belirlenmiş komiteler tarafından yürütülmektedir. Böylece, hem blokzincir ağında alternatif zincirlerin oluşmasının önüne geçilmektedir, hem de oylama tabanlı uzlaşı mekanizmalarının sıkıntısını çektiği ölçeklenebilme problemi kısmen giderilmektedir.

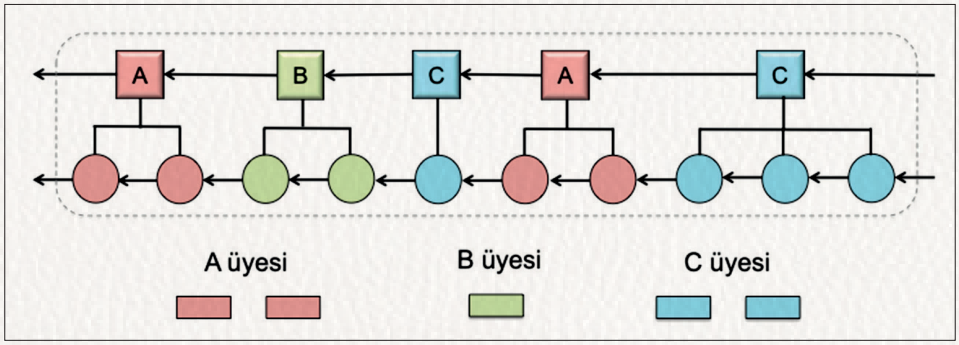
2.2.3.1. Emeğin Kanıtı ve Bizans Hata Toleransı Hibrit Yaklaşımı

Peercensus: Yukarıda da ifade edildiği gibi; bitcoinde, blokzincir ağında yaşanabilecek gecikmeler, emeğin kanıtı algoritmasında kullanılan matematiksel bulmacanın aynı anda iki farklı kişi tarafından çözülmesi veya kötü niyetli

bir düğümün çifte harcama saldırısı gerçekleştirmeye teşebbüs etmesi gibi nedenlerden dolayı blokzincir ağının farklı bölgelerinde aynı indekse sahip farklı bloklar paylaşılabilmektedir. Ağdaki düğümlerin farklı zincirlerle yola devam etmesine sebep olabilecek bitcoinin bu yapısı çatallanma problemi diye adlandırılmaktadır. Decker vd. [52], bitcoindeki bu çatallanma problemini gidermek ve düğümlerin daha tutarlı bir blokzinciri yürütmesini sağlamak amacıyla Peercensus adı verilen bir protokol önermişlerdir. Peercensus protokolünde bloklar, bitcoinde olduğu blokzincir ağında uygulayabileceği hash gücü yüksek olanların seçilme şansının daha çok olduğu emeğin kanıtı algoritması yoluyla belirlenen liderler tarafından üretilmektedir. Bitcoinden farklı olarak, bloklar zincir mutabakatı (chain agreement) adı verilen ve Bölüm 2.1.3'de açıklanan PBHT'ye benzer bir protokol yoluyla doğrulanıp zincire eklenmektedir. Zincir mutabakatı protokolünde; yine PBHT'ye benzer bir biçimde zaman, pencere adı verilen parçalara bölünmüştür. Her bir pencere için protokole katılan onaylayıcılardan biri baş düğüm rolünü üstlenerek protokolü yönetir ve destek düğüm adını alan diğer onaylayıcılar ise teklif edilen bloğun onaylanıp zincire eklenmesi sürecinde baş düğüme destek olurlar.

Emeğin kanıtı algoritması yoluyla blok üreterek bunu blokzincir ağıyla paylaşan düğümler, zincir mutabakatı protokolünü gerçekleştirecek onaylayıcılara katılmaya hak kazanırlar. Ayrıca blok üreterek onaylayıcılar arasına katılan ilk düğüm bir sonraki pencerede baş düğüm rolünü üstlenir. Zincir mutabakatı protokolü teklif, ön-hazırlık, hazırlık ve taahhüt adlı 4 aşamadan oluşmaktadır. Teklif aşamasında, emeğin kanıtı algoritması yoluyla sıradaki bloğu üretmeye hak kazanmış düğüm, ürettiği bloğu imzalı bir şekilde ilgili pencerenin baş düğümüne gönderir. Ön-hazırlık aşamasında baş düğüm, geçerliliğini kontrol ettikten sonra bloğu ve bloğun zincirdeki indeksini belirleyecek sıra numarasını imzalayarak diğer düğümlerle paylaşır. Hazırlık aşamasında ise destek düğümler önce bloğun ve bloğa atanan sıra numarasının geçerliliğini kontrol ederler. Sonrasında ön-hazırlık mesajı yoluyla aldıkları bloğu ve ilgili sıra numarasını içeren imzalı bir hazırlık mesajını blokzincir ağıyla paylaşırlar. Taahhüt aşamasında ise düğümler eğer onaylayıcıların üçte ikisinin fazlasından geçerli bir hazırlık mesajı almışlarsa, mesajın içerdiği bloğa ve sıra numarasına taahhüt ettiklerine dair imzalı bir mesajı blokzincir ağıyla paylaşırlar. Protokolün sonunda blokzincir ağındaki düğümler eğer onaylayıcıların üçte ikisinde fazlasından geçerli bir taahhüt mesajı almışlarsa, mesajda içerilen bloğu ilgili sıra numarası ve aldığı taahhüt mesajlarıyla beraber zincire eklerler.

Protokolde onaylayıcılar, sistemin daha sağlıklı çalışmasını sağlamak amacıyla diğer onaylayıcılara periyodik yoklama mesajı göndermektedir. Bu mesajlara uzun süre dönmeyen onaylayıcılar aktif onaylayıcıların katıldığı bir oylama sonucu onaylayıcı kümesinden atılmaktadır. Böylelikle bu tarz onaylayıcılar zincir mutabakatı protokolünün dışında tutularak protokolün canlılığı ve güvenliği artırılmaktadır. Peercensus, zincir mutabakatı protokolüne katılan onaylayıcıların üçte ikisinden fazlası dürüst olduğu sürece güvenlik ve canlılık özelliklerini sağlamaktadır.

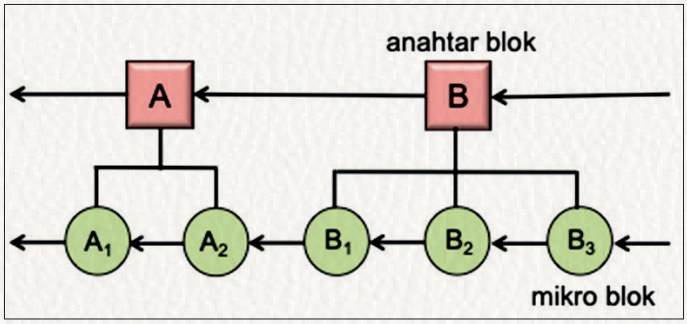


Şekil 2.14. Byzcoin Protokolünde Oy Hakkının Periyodik Dağılımı

Byzcoin: Emeğin kanıtı algoritmasıyla Bizans hata toleransı protokolünü harmanlayarak, bitcoindeki bloğun neticelendirilmesinin olasılıksal durumunu iyileştirmeyi amaçlayan bir diğer protokol Byzcoin [53] protokolüdür. Byzcoin protokolünde bloklar, emeğin kanıtı algoritması yoluyla belirlenen liderler tarafından üretilir. Üretilen bloklar Peercensus'e benzer biçimde, PBHT benzeri bir Bizans hata toleransı (BHT) protokolü yoluyla doğrulanıp zincire eklenirler. Yine Peercensus'e benzer olarak blok üreticileri BHT protokolünü gerçekleştirecek komiteye yeni üye olarak katılırlar. Fakat Peercensus'ten farklı olarak Şekil 2.14'de görüldüğü gibi komite üyeleri BHT protokolünde ürettikleri blok sayısı ile orantılı oy hakkına sahiptir. Diğer yandan Byzcoin, hash gücü yüksek blok üreticilerinin BHT protokolündeki oylamayı domine etmelerini engellemek ve düğümleri çevrim içi kalmaya teşvik etmek amacıyla komite üyeliklerini belirli peryotlarla sınırlandırır.

PBHT protokolünde olduğu gibi, BHT protokolü de belirli bir baş üye tarafından yürütülür ve diğer komite üyeleri baş üyeye bu süreçte destek olurlar.

Yine benzer şekilde protokol ön-hazırlık, hazırlık ve taahhüt aşamalarından oluşur. Fakat PBHT protokolünden farklı olarak, blokzincir ağının yüklendiği mesaj karışıklığını düşürmek adına, BHT protokülünde paylaşılan imzalı mesajlar Syta vd. [54] tarafından literatüre sunulan kollektif imza algoritması uygulanarak kompakt hale dönüştürülür. Kollektif imza algoritmasının yardımıyla Byzcoin protokolünün mesaj karışıklığı asimptotik olarak $O(1)$'e düşürülmektedir. Böylelikle BHT protokolünü yürütecek komitenin eleman sayısı açısından daha ölçeklenebilir bir yapı elde edilmiş olunur.



Şekil 2.15. Byzcoin Protokolünde Zincir Yapısı

Byzcoin ayrıca verimliliği artırmak adına Bitcoin-NG protokolünde olduğu gibi emeğin kanıtı algoritması yoluyla lider seçilen düğümlere bir sonraki lider seçilinceye kadar mikro bloklar üretme imkanı vermektedir. Üretilen mikro bloklar yine kollektif imza algoritması ile iyileştirilmiş BHT protokolü üzerinden doğrulanarak zincire eklenmektedir. Böylelikle orijinal Bitcoin-NG protokolünden farklı olarak liderin bir sonraki lider seçilinceye kadar sistemi manipüle etmesinin de önüne geçilmektedir. Ayrıca orijinal Bitcoin-NG protokolünde, mikro bloklar ve mikro blokları üretecek liderleri belirleyen anahtar bloklar aynı zincire eklenmekteydi. Byzcoin'de ise Şekil 2.15'te görüldüğü gibi mikro bloklar ve anahtar bloklar iki ayrı zincirde tutulmaktadır. Burada mikro blokların oluşturduğu zincir blokzincir ağında oluşan işlemleri saklamak için, anahtar blokların oluşturduğu zincir ise emeğin kanıtı algoritması yoluyla BHT protokolünü yürütecek üyelerin belirlenmesi için kullanılmaktadır. Byzcoin protokolü, $3f+2$ toplam düğüm sayısı olmak üzere hatalı düğümlerin sayısının f'den fazla olmadığı kabul edildiğinde güvenlik ve canlılık özelliklerini sağlamaktadır.

i. Emeğin Kanıtı ve Bizans Hata Toleransı Hibrit Yaklaşımının Değerlendirilmesi

Yukarıdada ifade edildiği gibi Peercensus, blokların onaylanması esnasında yürütülen BHT protokolüne katılan onaylayıcıların üçte ikisinden fazlası dürüst olduğu sürece güvenlik ve canlılık özelliklerini sağlamaktadır. Byzcoin ise $3f$+2 toplam düğüm sayısı olmak üzere hatalı düğümlerin sayısının f'den fazla olmadığı kabul edildiğinde güvenlik ve canlılık özelliklerini sağlamaktadır. Buradaki farklılık mikro bloklardan kaynaklanmaktadır. Durumu bir örnekle açıklamak gerekirse; farzedelim ki, protokolde f Bizans hatalı düğüm, f zincirin güncel haline sahip dürüst düğüm ve f zincirin son haline vakıf olmayan görece yavaş düğüm olsun. Yine farzedelim ki, protokolde bir sonraki bloğun üretileceği zaman dilimine geçilmiş olsun ama ilgili zaman diliminin lideri de zincirin son haline vakıf olmasın. Bu durumda lider bir önceki zaman diliminde üretilen mikroblokları göz ardı eden yeni mikro bloklar üretebilir ve bu mikro bloklar için Bizans hatalı düğümler de dahil toplamda $2f$+1 düğümden onay alabilir. Bu durumu gidermek adına, PBHT protokolünde onay için en az $2f$+2 oy toplanması gerekmektedir. BHT protokolleri genel olarak Bizans hatalı düğümlerin gerçekleştirebileceği DoS saldırılarına karşı zayıftırlar. Bununla birlikte, komite üyelerinin değişkenlik göstermesi, iki protokolü de DoS saldırılarına karşı biraz dirençli yapmaktadır.

Bölüm 2.1.3'de açıklanan PBHT protokolü düğümler arası direk iletişim kanalları gerektirmekteydi. Dolayısıyla protokolün mesaj karışıklığı asimptotik olarak $O(n^2)$ olacaktır. Peercensus'te ise mesajların otantisitesi dijital imza algoritmaları üzerinden doğrulanabildiğinden ve düğümler dedikodu (gossip) protokolü üzerinden iletişim kurduklarından, uygulanan böyle bir PBHT protokolünün mesaj karmaşıklığı $O(n)$ olacaktır. Ayrıca Byzcoin'de kollektif imza algoritmasının yardımıyla mesaj karışıklığı asimptotik olarak $O(1)$'e düşürülmektedir. Byzcoin ve Peercensus protokollerinde üretilen bloklar BHT algoritmaları yoluyla onaylanıp zincire eklendiği için, bitcoinde olduğu gibi işlemlerin geçerli kabul edilmesi için bir kaç blok beklenmesine gerek yoktur. Bu da bu protokolleri daha verimli yapmaktadır. Örneğin Byzcoin için 144 düğüm üzerinden gerçekleştirilen bir simülasyonda verimlilik 1000 işlem/sn olarak ölçülmüştür [53].

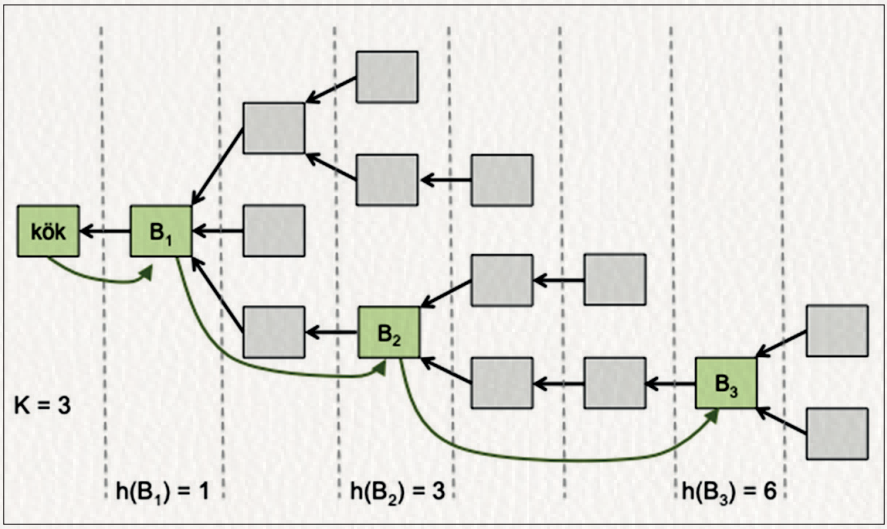
Byzcoinde yeni üretilen akçe ve kesintiler üzerinden toplanan ödül ilgili bloğu üreten düğüme verilmek yerine komite üyeleri arasında oy haklarıyla orantılı bir şekilde pay edilir. Ayrıca orijinal Bitcoin-NG protokolünde mikro

bloklardan toplanan kesintilerden pay verilmek suretiyle bir sonraki anahtar bloğunu oluşturacak liderin, bir önceki liderin ürettiği blokları görmemezlikten gelmesi engellenmektedir. Byzcoin prokolünde ise bloklar komite üyelerince onaylanıp zincire eklendiği için böyle bir teşvik mekanizmasına ihtiyaç duyulmamaktadır.

2.2.3.2. Hisse Kanıtı ve Bizans Hata Toleransı Hibrit Yaklaşımı

Casper: Bölüm 2.2.1.2'de de ifade edildiği gibi hisse kanıtı algoritmalarında lider ağda akçesi bulunan düğümlerin arasından sahip oldukları akçe miktarıyla orantılı olasılıksal bir yolla seçilmekteydi. Seçilen liderler ürettikleri blokları başlangıç bloğuna ekleyerek bir zincir oluşturmaktaydılar. Emeğin kanıtı tabanlı uzlaşı mekanizmalarındakine benzer, ağda oluşabilecek gecikmeler veya kötü niyetli liderlerin saldırıları nedeniyle, ağda birbirinin alternatifi farklı zincirler oluşabilmekteydi. Casper [55] hisse kanıtı tabanlı uzlaşı mekanizmalarındaki zikredilen bu çatallanma problemini çözmek adına tasarlanmış bir protokoldür.

Protokolde blok üretimine katılmak ve çatallanma probleminin çözümüne katkı için oy kullanmak isteyen düğümlerin belli bir miktar akçeyi depozito olarak yatırmaları gerekmektedir. Depozitoyu yatırdıktan sonra protokolde artık onaylayıcı (validator) diye adlandıracağımız düğümler Round-Rabin usulüyle lider olup sıradaki blokları üretirler. Ayrıca protokolde onaylayıcılar belli periyotlarla Bizans hata toleransı (BHT) algoritması çalıştırarak, ağ üzerindeki alternatif zincirlerden birinin üzerinde mutabık kalıp onu geçerli zincir olarak belirlerler. BHT'nda onaylayıcılar geçerli zinciri belirlemek için her K blokta bir bir oylama gerçekleştirirler. Oylamada onaylayıcılar, Şekil 2.16'da görüleceği gibi geçerli zincirde yer alması gerektiğini düşündükleri kontrol noktalarına karar vermektedirler. Onaylayıcılar oylarını, v onaylayıcının ağdaki adresi, s daha önceden onaylanmış herhangi bir kontrol noktası, t geçerli zincirde yer almasını istedikleri yeni kontrol noktası ve $h(s)$, $h(t)$ de ilgili kontrol noktalarının blokzincir ağacındaki yükseklikleri olmak üzere, $\langle v, s, t, h(s), h(t) \rangle$ şeklinde bir mesajla diğer düğümlerle paylaşırlar. Oylamada oylar sayılırken onaylayıcıların yatırdıkları depozito miktarı dikkate alınır. Örneğin A ve B onaylayıcıları sırasıyla 5 ve 3 akçe yatırmışlarsa, A'nın oyladığı kontrol noktası 5 oy ve B'nin oyladığı kontrol noktası 3 oy almış sayılır. Oyların üçte ikisinden fazlasını alan kontrol noktası onaylanmış yeni kontrol noktası olarak belirlenir.



Şekil 2.16. Casper Protokolü

Oylamada onaylayıcıların; aynı yüksekliğe sahip farklı iki kontrol noktası için oy kullandığı, ya da herhangi bir kontrol noktası için oy kullandıktan sonra yüksekliği ondan daha küçük başka bir kontrol noktası için de oy kullandığı tespit edilirse yatırdıkları depozito yanacaktır. Protokol, blokzincirin yaşam süresi boyunca sabit bir onaylayıcı kümesi yoktur. Yeni düğümler belli bir miktar depozito yatırarak onaylayıcılar kümesine dahil olabildikleri gibi, mevcut onaylayıcılar yatırdıkları depozitoyu çekerek bu kümeden ayrılabilirler. Protokol, yatırılan toplam depozitoların üçte ikisinden fazlasının dürüst doğrulayıcıların kontrolünde olduğu kabul edildiğinde, birbiriyle çelişen (alternatif zincirlerde yer alan) kontrol noktalarının her ikisinin de doğrulanmayacağını garantilemektedir (güvenlik özelliği). Ayrıca yatırılan toplam depozitoların üçte ikisinden fazlasını kontrol eden düğümler protokolü gerektiği gibi takip ederlerse, protokol zamanla yeni kontrol noktalarının doğrulanacağını, yani canlılık özelliğinin sağlanacağını da garantilemektedir.

Algorand: 2017 yılında Gilad vd. [56] tarafından geliştirilmiş, sıradaki blokları üretecek liderleri hisse kanıtı tabanlı bir algoritma kullanarak seçen ve üretilen blokların değiştirelemez bir biçimde zincire eklenmesini Bizans hata toleransı algoritması üzerinden garantileyen bir kripto paradır. Algorand'da blok üretimine hak kazanan lider ve üretilen her bir bloğun neticelendiril-

mesi sürecinde çalıştırılan Bizans hata toleransı algoritmasına dahil olacak komite üyeleri, kriptografik tabanlı bir kura seçimi ile belirlenmektedir. Bahsi geçen kura seçimi doğrulanabilir bir yalancı rastgele fonksiyon yardımıyla gerçekleştirilmektedir. Ayrıca kura seçimine dahil olabilmek için düğümlerin doğrulanabilir rastgele fonksiyonu çalıştırabilen gizli ve açık anahtar bulundurmaları gerekmektedir. Kura seçiminde her düğüm, önce gizli anahtarını kullanarak, blok üretim sürecinde üstlenmek istediği rol ve tüm düğümlerin erişebileceği bir kaynaktan elde edilmiş rastgele bir *s* değerine doğrulanabilir rastgele fonksiyonu uygular. Burada fonksiyon çıktı olarak bir *hash* özet değeri ve bir π kanıt değeri üretmektedir. Sonrasında ilgili düğüm *hash* değerini belli işlemlerden geçirerek lider ya da komite üyesi olup olmadığını belirleyecek, düğümün blokzincir ağında sahip olduğu akçelerle orantılı olarak olasılıksal bir yolla bir *j* parametresi üretir. Dolayısıyla daha çok akçe sahibi düğümlerin daha büyük parametre üretme olasılığı daha fazladır. Bunun yanı sıra düğümlerin kura seçiminde ürettikleri değerler, daha önceden ağla paylaştıkları açık anahtarları kullanılarak doğrulanabilir.

Algorand'da her bir rol için bir eşik değeri belirlenmiştir. Düğümlerin blok üretim sürecine istedikleri rol üzerinden dahil olabilmesi için kura seçiminde ürettikleri *j* parametresinin belirlenen eşik değerlerinden yüksek olması gerekmektedir. Belirlenen eşik değerinden yüksek parametre üretmiş birden fazla düğüm olabileceği için farklı düğümler lider olarak öne çıkıp ürettikleri blokları ağla paylaşabileceklerdir. Bu da protokolün performansını olumsuz yönde etkileyecektir. Bu durumu gidermek adına Algorand, düğümlerin kura çekiminde ürettikleri *hash* özet değeri ve *j* parametresinin özetini (hash) alarak bir öncelik değeri üretmektedir. Blok üretim sürecinde belirli bir eşik değerinden daha yüksek *j* parametresi üretmiş lider adayları, öncelikli olarak bütün bir blok yerine protokolün performansını artırmak adına yukarıda bahsi geçen öncelik değerini ve π kanıt değerini diğer düğümlerle paylaşırlar. Sonrasında en yüksek öncelik değerine sahip lider, ürettiği bloğu diğer düğümlerle paylaşır.

Bu aşamada komite üyeleri, iki aşamalı Bizans hata toleransı algoritması yoluyla paylaşılan bloğu oylarlar. Oylamada yine performansı artırmak adına bloğun kendisi değil de özeti oylanacaktır. İlk aşamada her bir komite üyesi diğer komite üyelerine, kura çekiminde kullanılan rastgele *s* değerini, zincire eklenmiş geçerli en son bloğun özet değerini, kura çekimi yoluyla üret-

tiği *hash* özet değerini ve π kanıt değerini, en yüksek öncelik değerine sahip onaylanacak bloğun özet değerini ve zincirdeki sırasını içeren bir mesajı imzalayıp gönderirler. Mesajı alan komite üyeleri önce imzanın doğruluğun tetkik ederler. Sonrasında mesajda yer alan zincire eklenmiş geçerli en son bloğun özet değerini kendi zincirlerindeki son bloğun özet değeri ile karşılaştırırlar. Bu kontrollerden sonra mesajı gönderen komite üyesinin açık anahtarını ve mesajdaki ilgili değerleri kullanarak *hash* özet değerini ve π kanıt değerinini doğrular ve ilgili komite üyesinin *j* parametresini hesaplarlar. Bu aşamada sadece *j* parametresi belirli bir eşik değerinden yüksek olan komite üyelerinin oyları geçerli sayılmaktadır. Ayrıca oylanan blok için toplam oy miktarı, oy kullanan komite üyelerinin blokzincir ağında sahip olduğu akçelerin toplamı olarak belirlenmektedir. İkinci aşamada komite üyeleri, eğer oylanan blok için ilk aşamada bütün komite üyelerinin sahip olduğu akçelerin üçte ikisinden fazla oy kullanılmışsa, diğer komite üyelerine ilgili bloğu geçerli kabul ettiklerini bildiren bir mesaj gönderirler. Eğer oylanan blok yeterli oy almamışsa, blokzincir ağınca kabul edilmiş boş bir bloğun özetini içeren bir mesaj gönderirler. Algorand protokolü blokzincir ağındaki akçelerin kontrolünün üçte ikisinden fazlasının dürüst düğümlerde olduğu kabul edildiğinde eşzamanlı ağ modeli için canlılık ve kısmi eşzamanlı ağ modeli için güvenlik özelliklerini sağlamaktadır.

Tendermint: Hisse kanıtı ve Bizans hata toleransı algoritmalarını hibrit bir yaklaşım haline getirerek kullanan protokollerden bir diğeri de Tendermint [57] protokolüdür. Casper protokolüne benzer biçimde Tendermint'de de, blok üretimi ve sonrasında üretilen blokların Bizans hata toleransı algoritması üzerinden onaylanması süreçlerine katılmak isteyen düğümlerin, belirli bir miktar akçeyi 'senet' adı verilen bir işlem üzerinden depozito olarak yatırmaları gerekmektedir. Depozitoyu yatırdıktan sonra protokolde artık onaylayıcı (validator) diye adlandıracağımız düğümler, yatırdıkları depozito miktarıyla orantılı bir sıklıkla Round-Rabin usulü lider olup sıradaki blokları üretirler.

Tendermint protokolü teklif, ön-oylama, ön-taahhüt, taahhüt ve yeni-yükselik adı verilen 5 aşamadan oluşmaktadır. Teklif aşamasında; sıradaki zaman dilimine atanmış lider, varsa daha önceden kabul ettiği ama zincire eklenmemiş bir bloğu, yoksa kendisine ağ üzerinden ulaşmış yeni işlemler üzerinden oluşturduğu yeni bloğu diğer düğümlerle paylaşır. Ön-oylama aşamasında, her bir onaylayıcı eğer kendisine önceki dönemlerden ulaşmış ama zincire

eklenmemiş bir blok varsa onu oyladığını belirten imzalı bir mesajı blokzincir ağıyla paylaşır. Böyle bir blok yoksa, onaylayıcı lider tarafından bu dönem için oluşturulup paylaşılmış yeni bloğu, o da yoksa boş bir bloğu oyladığını belirten imzalı bir mesajı diğer düğümlerle paylaşır. Ön-taahhüt aşamasında; eğer onaylayıcılar, herhangi bir blok için tüm onaylayıcıların çoğunluğundan[5] ön-oylama mesajı almışlarsa, aynı blok için blokzincir ağıyla imzalı bir ön-taahhüt mesajı paylaşırlar.

Taahhüt aşaması öncesi onaylayıcılar, eğer herhangi bir blok için tüm onaylayıcıların çoğunluğundan ön-taahhüt mesajı almamışlarsa bir sonraki dönemin teklif aşamasına geçerler. Eğer onaylayıcılar, herhangi bir blok için tüm onaylayıcıların çoğunluğundan ön-oylama mesajı almışlarsa ilgili bloğa taahhüt ettiklerine dair imzalı bir mesajı diğer düğümlerle paylaşırlar. Bu aşamada onaylayıcılar eğer tüm onaylayıcıların çoğunluğundan aynı bloğa taahhütte bulunduklarına dair mesaj almışlarsa, ilgili bloğu geçerli kabul edip zincire ekler ve yeni-yükseklik aşamasına geçerler. Burada yeni-yükseklik aşaması bir sonraki bloğun üretimine geçmeden önce, görece daha yavaş onaylayıcılara taahhüt aşamasını tamamlamak ve bu şekilde bloğun geçerli olmasını sağlayabilecek gerekli çoğunluğu tamamlayabilmek için protokole eklenmiş bir aşamadır.

EOSIO: Bu bölümde inceleyeceğimiz son protokol yetkilendirmiş hisse kanıtı ve Bizans hata toleransı algoritmalarını bir arada kullanan EOSIO uzlaşı protokolüdür [58]. Protokol hisse sahibi düğümlere, yetkilendirilmiş hisse kanıtı algoritmasında kullanılmak üzere hisselerini bir başkasına devretme hakkı sunmaktadır. Protokol, yetkilendirilmiş hisse kanıtı algoritması üzerinden, her 126 saniyede bir (devredilmiş hisseleri de hesaba katarak) en çok hisse sahibi ilk 21 düğümü komite olarak belirler. Komite üyeleri, ilgili turdaki blokları üretmekten ve üretilen blokları yürüttükleri Bizans hata toleransı protokolü üzerinden onaylayıp zincire eklemekten sorumludur. Her bir komite üyesi 6 saniyelik zaman diliminden sorumludur. EOSIO'da blok üretim aralığı yarım saniye olarak ayarlandığından her bir üye en fazla 12 blok üretebilmektedir.

Kendi zaman diliminde komite üyeleri, ürettikleri blokların lider olarak yürüttükleri Bizans hata toleransı algoritması üzerinden onaylatıp zincire eklenmesini sağlarlar. Bizans hata toleransı algoritması iki aşamadan oluşmak-

5 Burada çoğunluktan kastedilen Casper protokolünde olduğu gibi ön-oylama mesajı paylaşanların yatırılan tüm depozito miktarının üçte ikisinden fazlasına sahip olmasıdır.

tadır. İlk aşamada lider ürettiği bloğu diğer üyelerle paylaşır. Üyeler bloğun geçerliliğini kontrol ettikten sonra, ilgili bloğu imzalı bir onay mesajını ağla paylaşırlar. Eğer ilgili bloğu geçerli bir mesaj göndererek onayların toplam hissesi komite üyelerinin toplam hissenin üçte ikisinden fazla ise, komite üyeleri bloğa taahhütte bulunurlar ve bu taahhütlerini imzalı bir mesaj üzerinden blokzincir ağıyla paylaşırlar. Finalde eğer ilgili bloğa geçerli bir mesaj göndererek taahhütte bulunanların toplam hissesi komite üyelerinin toplam hissenin üçte ikisinden fazla ise, komite üyeleri bloğu onaylanmış kabul edip zincire eklerler.

Protokol karlarını artırmak adına düğümlere hisselerini birden fazla kişiye devredebilme hakkı vermektedir. Ayrıca düğümlerin aynı işlemleri farklı zincirlere eklemelerinin önüne geçmek ve blokzincir ağını düğümlerin hangi zincirde hisse sahibi oldukları noktasında bilgilendirmek amacıyla, protokolde işlemler oluşturulurken en son bloğun başlığının özeti de işleme eklenmektedir. EOSIO protokolü blok üreticilerini yıllık bazda %1'lik bir enflasyon oranı üzerinden üretilen yeni akçelerle ödüllendirmektedir. Ayrıca protokolde işlemler için kesinti alınmamaktadır.

i. Hisse Kanıtı ve Bizans Hata Payı Hibrit Yaklaşımının Değerlendirilmesi

Casper ve Tendermint protokolleri yatırılan toplam depozitoların üçte ikisinden fazlasının dürüst doğrulayıcıların kontrolünde olduğu kabul edildiğinde, güvenlik ve canlılık özelliklerini sağlamaktadır. Diğer yandan Algorand, blokzincir ağındaki akçelerin kontrolünün üçte ikisinden fazlasının dürüst düğümlerde olduğu kabul edildiğinde, eşzamanlı ağ modeli için canlılık ve kısmi eşzamanlı ağ modeli için güvenlik özelliklerini garantilemektedir. EOSIO'da ise oluşturulan komite üyelerinin toplam hissesinin üçte ikisinden fazlası dürüst üyelerin elindeyse güvenlik özelliğini sağlamaktadır. Bunların yanı sıra Casper ve Tendermint protokolleri, düğümlerin sıfır hisse (nothing-at-stake) saldırısı gerçekleştirerek alternatif zincirler için taahhütte bulundukları, ya da alternatif zincirler için blok ürettikleri tespit edilirse yatırdıkları depozitoyu yakmaktadırlar.

Casper ve Tendermint protokollerinde Bizans hata toleransı algoritmalarını yürütecek düğüm sayısı yüksek olacaktır. Dolayısıyla protokollerin mesaj karmaşıklığı görece daha yüksek olacak ve bu durum da ilgili protokolleri ölçeklenebilirlik (scalibility) problemi ile karşı karşıya bırakacaktır. EOSIO için

BHT algoritmasını yürütecek sadece 21 düğüm olduğundan, protokolün mesaj karmaşıklığı oldukça düşük olacaktır. Bununla birlikte, sıradaki tur için blokları üretecek liderlerin kimlikleri blokzincir ağıyla paylaşıldığından, EOSIO protokolünde düşmanın DoS türü saldırılar yoluyla ilgili komite üyelerinin işini zorlaştırması, daha kötüsü ilgili düğümleri bozarak sistemi baltalaması mümkün olabilecektir. Algorand'da ise tam tersi olarak, blok üretim sürecine lider ya da komite üyesi olarak dahil olacak herhangi bir düğümün kimliği, bu düğüm kura çekiminde ürettiği değerleri blokzincir ağıyla paylaşıncaya kadar diğer düğümlerce bilinemez. Burada kötü niyetli düğümlerin yapabileceği en fazla paylaşılan bütün açık anahtarlar için kura seçiminde kullanılan rastgele fonksiyonun doğrulamasını yaparak deneme yanılma yoluyla lideri ve komite üyelerini tespit etmeye çalışmak olacaktır. Ayrıca komite boyutu görece çok daha küçük olduğundan Algorand protokolünün mesaj karmaşıklığı Casper ve Tendermint'e nazaran çok daha azdır.

2.2.3.3. Hibrit Uzlaşı Mekanizmalarının Değerlendirilmesi

Bu bölümde incelenen uzlaşı mekanizmaları, BHT algoritmasını yürütecek komitenin nasıl oluşturulduğu (komite kurulumu), liderin nasıl belirlendiği (lider seçimi), protokolün tolere edebildiği kötü niyetli düğümlerin oranı (hata toleransı), protokolün mesaj karmaşıklığı (karmaşıklık), protokolün saniyede onaylayıp zincire ekleyebildiği işlem sayısı (verimlilik) ve protokolde kullanılan teşvik mekanizması (teşvik) olmak üzere 6 kriter üzerinden değerlendirilecek ve kıyaslanacaktır. Buradaki protokollerin hepsi için bloğun neticelendirmesi (finalization) diye adlandırılan, onaylanmış blokların üzerinde mutabık kalınması deterministik bir yolla yapıldığından bu kriter tabloya eklenmemiştir.

Peercensus ve Byzcoin, blok üreticilerini bitcoine benzer yeni üretilen belli bir akçe ve bloklara eklenen işlemlerden elde edilen kesintilerle ödüllendirmektedir. Yalnız bu ödül direk blok üreticisine verilmek yerine BHT protokolüne katılan komite üyeleri arasında paylaştırılmaktadır. Peercensus ödülü komite üyeleri arasında eşit paylaştırmaktayken, Byzcoin ödülü blok üretimine yaptıkları katkı oranında bölüştürmektedir. Benzer şekilde Casper ve Tendermint'te de ödül olarak işlemlerden elde edilen kesintiler komite üyeleri arasında yatırdıkları depozito miktarı ile doğru orantılı pay edilmektedir. Algorand'ın orijinal versiyonunda blok üreticilerini teşvik edici bir ödülden bahsedilmemektedir.

Diğer yandan, BHT algoritmalarının yapısından dolayı protokollerin mesaj karmaşıklığı protokolü yürüten komitenin boyutu ile doğru orantılıdır. Peercensus'te komite, emeğin kanıtı algoritması yoluyla lider olanlardan, Casper ve Tendermint protokollerinde ise blok üretim sürecinde yer almak için depozito yatıranlardan oluşmaktadır. Ayrıca Algorand'da komite hisse kanıtı algoritmasına dayalı yalancı rastgele fonksiyon yardımıyla belirlenirken, EOSIO protokolünde de yetkilendirilmiş hisse kanıtı algoritmasına bağlı olarak en çok oy alan 21 kişi üzerinden oluşturulmaktadır. Bu bilgiler ışığında, protokollerin mesaj karmaşıklığını blokzincir ağındaki toplam düğüm sayısına bağlı bir değer üzerinden belirleyip kıyaslamak zor olacağından, asimptotik analiz yerine birbirlerine göre performanslarını kıyaslayabilmek açısından yüksek ve düşük gibi niteleyiciler kullanılmıştır. Son olarak literatürde Peercensus, Casper ve Tendermint protokollerinin verimliliği için güvenilir bir kaynak bulunamamıştır. Oylama tabanlı uzlaşı mekanizmalarının yukarıda zikredilen kriterler yoluyla değerlendirilmesi Tablo 2.3'te verilmiştir.

Tablo 2.3. Oylama Tabanlı Uzlaşı Mekanizmalarının Karşılaştırılması

protokol	komite kurulumu	lider seçimi	hata toleransı	mesaj karmaşıklığı	teşvik	verimlilik
Peercensus	emeğin kanıtı	emeğin kanıtı	komite üyelerinin üçte birinden azı	yüksek	belli miktar akçe + kesintiler	-
Byzcoin	emeğin kanıtı	emeğin kanıtı	komite üyelerinin üçte birinden azı	düşük	belli miktar akçe + kesintiler	1000 işlem/sn
Casper	hisse kanıtı	Round Rabin	yatırılan toplam depozitonun üçte birinden azı	yüksek	kesintiler	-
Algorand	hisse kanıtı	hisse kanıtı	toplam akçenin üçte birinden azı	orta	-	1148 işlem/sn [59]
Tendermint	hisse kanıtı	Round Rabin + hisse kanıtı	yatırılan toplam depozitonun üçte birinden azı	yüksek	kesintiler	-
EOSIO	yetkilendirilmiş hisse kanıtı	Round Rabin	komite üyelerinin toplam akçesinin üçte birinden azı	düşük	belli miktar akçe	9656 işlem/sn [60]

2.3. SONUÇ VE DEĞERLENDİRMELER

Oylama tabanlı uzlaşı mekanizmaları Raft ve PBHT benzeri protokoller üzerine kurgulandığı için, dağıtık sistemlerle ilintili literatürden uzlaşı mekanizmalarının analizi ile alakalı birikimi blokzinciri dünyasına taşımak ve ilgili uzlaşı mekanizmalarının güvenlik ve canlılık gibi özellikleri sağladığını ispatlamak çok zor olmayacaktır. Diğer yandan, çekiliş tabanlı uzlaşı mekanizmalarındaki yukarıda da ifade edilen blok neticelendirilmesinin olasılıksal yapısı dikkate alındığında, geleneksel uzlaşı mekanizmalarındaki güvenlik ve canlılık gibi özellikleri bu tür uzlaşı mekanizmaları için formel olarak yeniden tanımlamak ve hash gücünün çoğunluğunun dürüst oyuncuların elinde olması gibi kabuller altında ilgili uzlaşı mekanizmalarının tanımlanan özellikleri sağladığını göstermek zorlayıcı bir uğraş olacaktır.

Bu konuda ilk çalışma Garay vd. [11] tarafından emeğin kanıtı tabanlı uzlaşı mekanizması kullanan bitcoin için yapılmıştır. Garay vd. [11] geleneksel uzlaşı mekanizmalarındaki güvenlik ve canlılık özelliklerini blokzincir için yeniden yorumlamışlar ve blokzincir ağındaki düğümler eş zamanlı çalıştığında bitcoinin, toplam hash gücünün yarısından azına sahip kötü niyetli bir düğümü tolere ederek bu özellikleri sağladığını formel olarak ispatlamışlardır. Benzer şekilde Ouroboros [29], Snow White [31] ve Algorand [56] gibi protokoller kullanılan uzlaşı mekanizmalarının güvenlik ve canlılık özelliklerini sağladıklarını formel biçimde ispat etmişlerse de; blokzincirindeki çoğu uzlaşı mekanizması için, ya mekanizmaların sadece belli saldırılara karşı dirençli olduğu ispatlanmış ya da güvenlik analizi konusunu göz ardı edilmiştir. Diğer bir ifadeyle blokzincirinde uzlaşı mekanizmalarının güvenlik analizi ile alakalı yeterli çalışma bulunmamaktadır. Halbuki; uzlaşı mekanizmalarının bir güvenlik protokolü olduğu düşünüldüğünde, uzlaşı mekanizmalarının yukarıda zikredilen canlılık ve güvenlik özelliklerini ne ölçüde sağladığının gösterilmesi gerekmektedir. Özellikle yetkilendirilmiş hisse kanıtı tabanlı EOSIO [58] ya da Tron [30] gibi yüksek verimlilik vadeden protokolleri daha sağlıklı değerlendirebilmek adına güvenlik analizi gereklilik arz etmektedir.

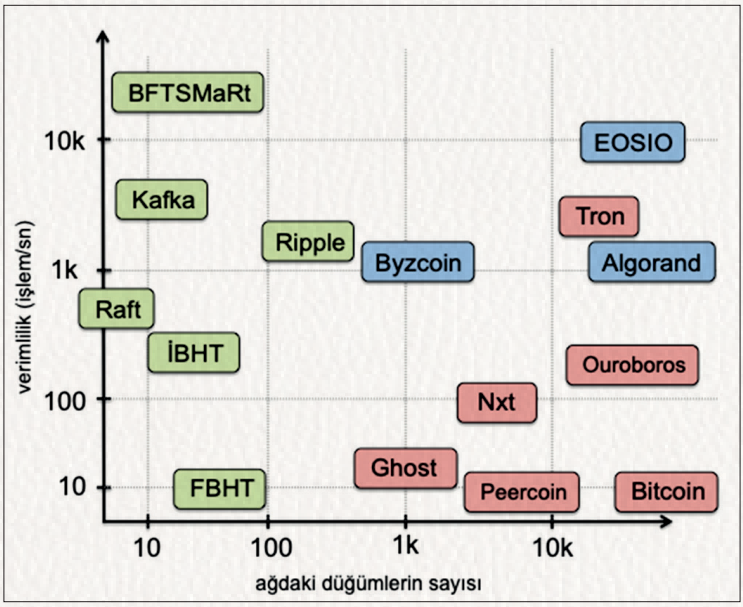
Çekiliş tabanlı uzlaşı protokollerinde bloğun neticelendirilmesi diye adlandırılan onaylanmış blokların üzerinde mutabık kalınması en uzun zincir ya

da Ghost gibi metotlar üzerinden olasılıksal bir yolla yapılmaktadır. Bloğun neticelendirilmesinin bu olasılıksal yapısı yüzünden üretilen bloklar onaylanıp zincire eklenseler bile ağda oluşacak çatallanmalar yoluyla düğümler bu bloklar yerine alternatiflerinin kabul edildiği zincirlerle yola devam edebilmektedirler. Bu durum çekiliş tabanlı uzlaşı mekanizmalarını çifte harcama atağı gibi saldırılara açık hale getirmektedir. Oylama tabanlı uzlaşı mekanizmalarında ise bloğun neticelendirilmesi ağdaki tüm düğümlerin katkılarıyla deterministik bir yolla yapılmaktadır. Bu durum blokzincir ağında alternatif zincirlerin oluşumunu engellemekte ve düğümlerin daha tutarlı bir zincir idame ettirmesine katkı sağlamaktadır.

Blokzincirinde, üretilen blokların onaylanarak zincire eklenmesi sürecine tüm düğümlerin dahil edilmesi uzlaşı mekanizmalarının mesaj yükünü ciddi oranda artırmaktadır. Altı çizilen bu ölçeklenebilme problemi sebebiyle oylama tabanlı uzlaşı mekanizmaları, daha çok katılımın izne bağlı olduğu ve dolayısıyla görece çok daha az sayıda kullanıcının yer aldığı izinli (permissioned) blokzincir platformlarında tercih edilmektedirler. Katılımın izne bağlı olmadığı ve çok sayıda kullanıcının yer aldığı izinsiz (permissionless) blokzincir platformlarında bu tür uzlaşı mekanizmalarının kullanımı önemli ölçüde performans düşüklüğüne sebep olacaktır. Diğer yandan çekiliş tabanlı uzlaşı mekanizmalarındaki mesaj yükü görece çok daha az olduğundan; oylama tabanlı uzlaşı mekanizmalarının aksine bu tür mekanizmalar, izinsiz blokzincir platformlarındaki yüksek sayıda oyuncuyu tolere edebilmektedirler.

Çekiliş tabanlı uzlaşı protokollerinde blok neticelendirilmesinin olasılıksal bir yolla yapılıyor olması, protokollerin verimliliğini (protokollerin bir saniyede onaylayıp zincire ekleyebildikleri işlem miktarı) düşürmektedir. Oylama tabanlı uzlaşı protokollerinde ise, tam tersi, blok neticelendirilmesinin deterministik olması protokollerin verimliliğini artırmaktadır. Ancak bu tür protokoller, yukarıda da ifade edildiği gibi, mesaj yükü yüksek olduğundan katılımcıların sayısı düşük tutulduğunda istenilen performansı gösterebilmektedir. Bu noktada her iki türün olumlu yanları harmanlanarak oluşturulmuş hibrit uzlaşı protokolleri, hem ölçeklenebilme problemini giderebilmekte hem de görece daha yüksek verimliliği başarabilmektedirler. Şekil 2.17'de bu bölümde incelenen uzlaşı protokolleri verimlilik ve ölçeklenebilirlik kriterleri üzerinden kıyaslanmıştır. Bu iki kriterle alakalı literatürde güvenilir bir kay-

nak bulunamayan protokoller kıyaslamanın dışında tutulmuş olup şekilden de anlaşılacağı gibi EOSIO ve Algorand gibi hibrit uzlaşı protokolleri hem ölçeklenebilirlik hem de verimlilik açısından en iyi performansı gösteren protokollerdir.



Şekil 2.17. Uzlaşı Protokollerinin Verimlilik ve Ölçeklenebilirlik Üzerinden Kıyaslaması.

Bizans hatalı düğümleri tolere edebilen geleneksel uzlaşı protokolleri; protokole katılan toplam düğüm sayısı j olmak üzere, $2\lfloor(n-1)/3\rfloor+1$ düğüm protokoldeki görevini gerektiği gibi yerine getirdiği sürece güvenlik ve canlılık özelliklerini sağlamaktadır. Diğer bir ifadeyle uzlaşı protokolleri $\lfloor(n-1)/3\rfloor$ Bizans hatalı düğümü tolere edebilmektedir. Protokollerin hata toleransının protokole katılan düğümlerin sayısı baz alınarak belirlenmesi, katılımın izne bağlı olduğu izinli blokzinciri platformları için problem teşkil etmeyecektir. İzinsiz blokzinciri platformlarında ise; kötü niyetli bir düğüm, 'Sybil' atağı adı verilen bir saldırı yoluyla çok sayıda sahte hesap oluşturarak protokolde gereken çoğunluğu elde edebilir ve düğümlerin onun talepleri üzerine fikir birliğine varmasını sağlayabilir. Çekiliş tabanlı uzlaşı protokolleri bu tür saldırıları engellemek için; protokollerin tolere edebildiği hatalı düğümlerin oranını, protokole katılan düğümlerin sayısı yerine lider seçiminde baz alınan

kriter üzerinden belirlemektedir. Örneğin bitcoin, uzlaşı protokolüne katılan bütün düğümlerin toplam hash gücünün yarısından azına sahip Bizans hatalı düğümleri tolere edebilmektedir.

Diğer yandan, uzlaşı protokollerinin hata toleransı kapasitesi ile protokolün güvenliği arasında doğrudan bir ilişki de vardır. Protokolün hata toleransı arttığında sağlayabildiği güvenlik azalmaktadır. Örneğin hisse tabanlı uzlaşı protokolü Ouroboros ağdaki toplam hisselerin yarısından azını kontrol edebilen kötü niyetli düğümleri tolere edebiliyorken, protokolde blok neticelendirmesi olasılıksal bir yolla yapıldığından ağda çatallanmalar oluşabilecektir. Diğer yandan blok üreticilerinin ağda sahip oldukları hisse baz alınarak seçildiği Algorand protokolünde, blok neticelendirilmesi BHT algoritması üzerinden deterministik bir yolla yapıldığından çatallanma riski görece çok daha azdır. Bununla birlikte, BHT algoritması sebebiyle protokol ancak ağdaki toplam hisselerin üçte birinden azını kontrol edebilen kötü niyetli düğümleri tolere edebilmektedir.

Blokzincir teknolojisi; şeffaflığı, veri bütünlüğünü garantilemesi, yolsuzluğu ve manipülasyonu engellemesi, kullanıcılara mahremiyet ve güven sağlaması gibi özellikleri dolayısıyla hem kamu hem de özel sektördeki güncel problemlerin çözümüne yönelik etkin ve güvenilir bir araç olarak öne çıkmaktadır. Bölüm içerisinde de belirtildiği gibi, blokzincirin performansını ve güvenliğini doğrudan etkilediği için uzlaşı protokolleri bu teknolojinin en önemli bileşenlerinden biridir. Kamudaki ve özel sektördeki blokzincir uygulamalarında tercih edilecek olan uzlaşı protokolü uygulamanın daha sağlıklı ve daha güvenli çalışmasına imkan sağlayabileceği gibi, sistemin performansını olumsuz yönde etkileyebilir ve sistem için ciddi güvenlik zafiyetleri oluşturabilir. Bu noktada, karar vericilerin yukarıda bahsi geçen riskleri de göz önünde bulundurarak tasarlanan sistemin gereksinimlerini iyi analiz etmeleri, sistemin güvenliğini pekiştirecek ve performansını artıracak en uygun uzlaşı protokolünü tercih etmeleri gerekmektedir.

KAYNAKLAR

[1] V. Hadzilacos and S. Toueg “A modular approach to fault-tolerant broadcasts and related problems”, Department of Computer Science, Cornell University, Ithaca, NY, USA, Tech. Rep. TR94-1425, 1994.

[2] M. Castro and B. Liskov, “Practical Byzantine fault tolerance,” *in Proc. USENIX Symp. Oper. Syst. Design Implement. (OSDI)*, Feb. 1999, pp. 173–186.

[3] S. Bano, A. Sonnino, M. Al-Bassam, S. Azouvi, P. McCorry, S. Meiklejohn and G. Danezis, "SoK: Consensus in the age of blockchains," *in Proc. ACM Conf. Adv. Financial Technol. (AFT)*, New York, NY, USA, 2019, pp. 183–198.

[4] C. Dwork, N. Lynch, and L. Stockmeyer, "Consensus in the presence of partial synchrony", *Journal of the ACM (JACM),* 35(2):288–323, 1988.

[5] B. Liskov and J. Cowling, "Viewstamped replication revisited," MITCSAIL: Computer Science and Artificial Intelligence Laboratory, Boston, MA, USA, Tech. Rep. TR2012-021, 2012.

[6] F. B. Schneider, "Implementing fault-tolerant services using the state machine approach: A tutorial", *ACM Computing Surveys*, 22(4), 1990.

[7] D. Ongaro and J. K. Ousterhout, "In Search of an Understandable Consensus Algorithm", *in Proc. USENIX Annual Technical Conference*, Philadelphia, PA, USA, 2014, pp. 305–320.

[8] P.L. Aublin, S. B. Mokhtar, and V. Quema, "Redundant Byzantine Fault Tolerance", *in Proc. 33rd International Conference on Distributed Computing Systems (ICDCS)*, Philadelphia, PA, USA, 2013, pp. 297–306.

[9] C. Dwork and M. Naor, "Pricing via processing or combatting junk mail", *in Proc. 12th Annual International Cryptology Conference (CRYPTO),* Santa Barbara, CA, USA, 1992, pp. 139–147.

[10] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System", 2008. Accessed on: June 10, 2021. [Online]. Available: http://bitcoin.org/bitcoin.pdf

[11] J. A. Garay, A. Kiayias, and N. Leonardos. "The bitcoin backbone protocol: Analysis and applications", *in Proc. 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Sofia, Bulgaria, 2015, pp. 281–330.

[12] C. Decker and R. Wattenhofer, "Information propagation in the Bitcoin network", *in Proc. 13th IEEE International Conference on P2P Computing*, Trento, Italy, 2013, pp. 1–10.

[13] J. Bonneau, "Why buy when you can rent? bribery attacks on bitcoin consensus", *in Proc. 20th Financial Cryptography and Data Security*, Barbados, 2016, pp. 19–26.

[14] "Nicehash". Accessed on: June 12, 2021. [Online]. Available: https://www.nicehash.com

[15] "Crypto51". Accessed on: June 12, 2021. [Online]. Available: https://www.crypto51.app

[16] "Visanet: The technology behind visa". Accessed on: June 13, 2021. [Online]. Available: https://usa.visa.com/dam/VCOM/download/corporate/media/visanet-technology/visa-net-booklet.pdf

[17] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, and E. G. Sirer, "On scaling decentralized blockchains," *in Proc. 20th Financial Cryptography and Data Security*, Barbados, 2016, pp. 106–125.

[18] Y. Sompolinsky and A. Zohar, "Secure high-rate transaction processing in Bitcoin", *in Proc. 19th Financial Cryptography and Data Security*, San Juan, Puerto Rico, 2015, pp. 507–527.

[19] I. Eyal, A. E. Gencer, E. G. Sirer, and R. van Renesse, "BitcoinNG: A Scalable Blockchain Protocol", *in Proc. 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, Santa Clara, CA, USA, 2016, pp. 45–59.

[20] A. E. Gencer, S. Basu, I. Eyal, R. Van Renesse and E. G. Sirer, "Decentralization in Bitcoin and Ethereum Networks," *in Proc. 22nd Financial Cryptography and Data Security*, Nieuwpoort, Curacao, 2018, pp. 439–457.

[21] "Hash Gücü Dağılımı". Accessed on: June 13, 2021. [Online]. Available: https://www.blockchain.com/charts/pools

[22] M. Abadi, M. Burrows, and T. Wobber, "Moderately hard, memory-bound functions", *in Proc. 10th Network and Distributed System Security Symposium (NDSS 2003),* San Diego, CA, USA, 2003.

[23] "Ethash". Accessed on: June 13, 2021. [Online]. Available: https://eth.wiki/en/concepts/ethash/ethash

[24] "Cryptonight Algortihm". Accessed on: June 13, 2021. [Online]. Available: https://www.programmersought.com/article/2857821139/

[25] "Digiconomist". Accessed on: June 13, 2021. [Online]. Available: https://digiconomist.net/bitcoin-energy-consumption

[26] S. King and S. Nadal, "PPCoin: peer-to-peer crypto-currency with proof-of-stake", 2012. Accessed on: June 13, 2021. [Online]. Available: https://decred.org/research/king2012.pdf

[27] Nxt Community, "Nxt Whitepaper", 2016. Accessed on: June 13 2021. [Online]. Available: https://nxtdocs.jelurida.com/Nxt_Whitepaper

[28] I. Bentov, A. Gabizon, and A. Mizrahi, "Cryptocurrencies without proof of work", *in Proc. 20th Financial Cryptography and Data Security*, Barbados, 2016, pp. 142–157.

[29] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol", *in Proc. 37th Annual International Cryptology Conference (CRYPTO 2017),* Santa Barbara, CA, USA, 2017, pp. 357–388.

[30] "Tron: Decentralize the Web". Accessed on: June 14, 2021. [Online]. Available: https://tron.network/

[31] P. Daian, R. Pass, and E. Shi, "Snow white: Robustly reconfigurable consensus and applications to provably secure proofs of stake", *in Proc. 23rd Financial Cryptography and Data Security*, St. Kitts and Nevis, 2016, pp. 23–41.

[32] "Tezos Developer Resources". Accessed on: June 14, 2021. [Online]. Available: https://tezos.gitlab.io/

[33] T. Kerber, A. Kiayias, M. Kohlweiss, and V. Zikas, "Ouroboros crypsinous: Privacy-preserving proof-of-stake", *in Proc. IEEE Symposium on Security and Privacy*, San Francisco, CA, USA, 2019, pp. 157–174.

[34] S. Dziembowski, S. Faust, V. Kolmogorov, and K. Pietrzak, "Proofs of space", *in Proc. 35th Annual International Cryptology Conference (CRYPTO 2015),* Santa Barbara, CA, USA, 2015, pp. 585–605.

[35] S. Park, A. Kwon, G. Fuchsbauer, P. Gaži, J. Alwen, and K. Pietrzak, "SpaceMint: A cryptocurrency based on proofs of space", *in Proc. 22nd Financial Cryptography and Data Security*, Nieuwpoort, Curacao, 2018, pp. 480–499.

[36] Protocol Labs, "Filecoin: A Decentralized Storage Network", 2017. Accessed on: June 14, 2021. [Online]. Available: https://filecoin.io/filecoin.pdf

[37] J. Kreps, N. Narkhede, and J. Rao, "Kafka: a Distributed Messaging System for Log Processing", *in Proc. 6th International Workshop on Networking Meets Databases*, Athens, Greece, 2011, pp. 1–7.

[38] E. Androulaki, A. Barger, and V. Bortnikov et al., "Hyperledger fabric: a distributed operating system for permissioned blockchains", *in Proc. 13th EuroSys Conference*, Porto, Portugal, 2018, pp. 1–15.

[39] "Hyperledger Indy". Accessed on: June 15, 2021. [Online]. Available: https://hyperledger-indy.readthedocs.io/projects/plenum/en/latest/index.html

[40] C. Gorenflo, L. Golab, and S. Keshav, "FastFabric: Scaling Hyperledger Fabric to 20000 transactions per second", *in Proc. IEEE International Conference on Blockchain and Cryptocurrency*, Seoul, Korea, 2019, pp. 455–463.

[41] "Throughput for Hyperledger Indy". Accessed on: June 15, 2021. [Online]. Available: https://jira.hyperledger.org/browse/INDY-1004

[42] "Consensys Quorum". Accessed on: June 15, 2021. [Online]. Available: https://consensys.net/quorum/

[43] H. Moniz, "The Istanbul BFT Consensus Algorithm", 2020. Accessed on: June 15, 2021. [Online]. Available: https://arxiv.org/pdf/2002.03613.pdf

[44] A. Baliga, I. Subhod, P. Kamat, and S. Chatterjee, "Performance Evaluation of the Quorum Blockchain Platform", 2018. Accessed on: June 15, 2021. [Online]. Available: https://arxiv.org/pdf/1809.03421.pdf

[45] G. Shapiro, C. Natoli, and V. Gromoli, "The Performance of Byzantine Fault Tolerant Blockchains", *in Proc. 19th IEEE International Symposium on Network Computing and Applications*, Cambridge, MA, USA, 2020, pp. 1–8.

[46] A. Bessani, J. Sousa, A. Oliveira, and F. Pedone, "From Byzantine Replication to Blockchain: Consensus is Only the Beginning", *in Proc. 50th IEEE International Conference on Dependable Systems and Networks*, Valencia, Spain, 2020, pp. 424–436.

[47] A. Bessani, J. Sousa, and E. A. P. Alchieri, "State Machine Replication for Masses with BFT-SMART", *in Proc. 44th IEEE International Conference on Dependable Systems and Networks*, Valencia, Spain, 2014, pp. 355–362.

[48] D. Schwartz, N. Youngs, and A. Britto, "The Ripple Protocol Consensus Algorithm", 2018. Accessed on: June 16, 2021. [Online]. Available: https://cryptoguide.ch/cryptocurrency/ ripple/whitepaper.pdf

[49] B. Chase and E. MacBrough, "Analysis of the XRP ledger consensus protocol", 2018. Accessed on: June 16, 2021. [Online]. Available: https://arxiv.org/pdf/1802.07242.pdf

[50] "XRP Validator Registry". Accessed on: June 16, 2021. [Online]. Available: https://xrpcharts.ripple.com/#/validators

[51] W. Yao, J. Ye, R. Murimi, and G. Wang, "A Survey on Consortium Blockchain Consensus Mechanisms", 2021. Accessed on: June 16, 2021. [Online]. Available: https://arxiv.org/pdf/2102.12058.pdf

[52] C. Decker, J. Seidel, and R. Wattenhofer, "Bitcoin Meets Strong Consistency", *in Proc. the 17th International Conference on Distributing Computing and Networking*, Singapore, 2016, pp. 1–10.

[53] E. Kogias, P. Jovanovic, N. Gailly, I. Khoffi, L. Gasser, and B. Ford, "Enhancing Bitcoin Security and Performance with Strong Consistency via Collective Signing", *in Proc. 25th USENIX Symposium*, Austin, TX, USA, 2016, pp. 279–296.

[54] E. Syta, I. Tamas, D. Visher, D. I. Wolinsky, P. Jovanovic, L. Gasser, N. Gailly, I. Khoffi, and B. Ford, "Keeping Authorities 'Honest or Bust' with Decentralized Witness Cosigning", *in Proc. IEEE Symposium on Security and Privacy*, San Jose, CA, USA, 2016, pp. 526–545.

[55] V. Buterin and V. Griffith, "Casper the Friendly Finality Gadget", 2017. Accessed on: June 18, 2021. [Online]. Available: https://arxiv.org/pdf/1710.09437.pdf

[56] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling Byzantine Agreements for Cryptocurrencies", *in Proc. the 26th Symposium on Operating Systems Principles,* Shangai, China, 2017, pp. 51–68.

[57] E. Buchman, "Tendermint: Byzantine fault tolerance in the age of blockchains", M.S. thesis, University of Guelph, Canada, 2016.

[58] "EOSIO Consensus Protocol". Accessed on: June 18, 2021. [Online]. Available: https://developers.eos.io/welcome/v2.0/protocol/consensus_protocol

[59] "Algorand's Throughput". Accessed on: June 18, 2021. [Online]. Available: https://metrics.algorand.org/

[60] "EOSIO's Throughput". Accessed on: June 18, 2021. [Online]. Available: https://www.eosgo.io/news/eosio-reaches-new-transaction-per-second-record

Bölüm 3

KİMLİK SİSTEMLERİNDE BLOKZİNCİR KULLANIMI

Serkan Ayvaz - Salih Cemil Çetin - Mehmet Aydar

Bu bölümde, blokzincir teknolojisi üzerinde gerçekleştirilebilecek dijital kimlik projelerinin incelenmesini içermektedir. Çalışmamızda Geleneksel kimlik sistemlerinde karşılaşılan sorunlar ve zayıflıklar incelenmiş ve daha verimli bir kimlik sistemine ulaşmak için blokzincir teknolojisinin potansiyel faydaları ortaya konulmuştur. Ayrıca, blokzincir tabanlı kimlik sistemlerinde biyometrik güvenlik mekanizmaları, özel anahtar şifreleme ve kurtarma yöntemleri irdelenmiştir. Bölümümüz, 5 başlık altında organize edilmiştir, 1. başlıkta konuya genel giriş yapılmış; 2. Başlıkta geleneksel kimlik sistemlerindeki gereksinimler ve zorluklar ele alınmış; sonrasında, blokzincir teknolojisi ile ilgili temel kavramlar ve tanımlara değinilmiş, 4. Başlıkta blokzincir temelli kimlik sistemleri detaylıca irdelenmiş; bunu takip eden bölümlerde ise bu sistemlerin kullanım senaryoları, özel anahtar şifrelenmesi ve anahtar kurtarma konuları işlenmiş; Türkiye Cumhuriyeti kimlik kartı ve blokzincir entegrasyonu bölümü detaylıca açıklanmış, son olarak da bölüm ile ilgili sonuç ve değerlendirmelerde bulunulmuştur.

3.1. GİRİŞ

Bireyler, isim, soyisim, ulusal kimlik numarası, pasaport numarası gibi farklı yöntemlerle kendilerini tanımlamakta ve tanıtmaktadırlar. Kimlik bilgileri, merkezi otoriteler tarafından fiziksel belgeler ile kayıt altına alınmaktadır.

Internet dünyasında ise kişiler kendilerini ancak merkezi yönetimlerin sayısal kayıtlarındaki kaynakları ile tanıtabilmek ve ispat edebilmektedirler. Geleneksel sistemlerde, kurumlar hassas kimlik bilgilerini kendi merkezi veritabanlarında saklarlar. Bu yöntemin, veri kopyalanması, kimlik hırsızlığı ve dolandırıcılık gibi kötüye kullanım konularına karşı zayıflıkları bulunmaktadır. Ayrıca geleneksel sistemler, güvenliğin yanı sıra yönetim ve kullanım zorluğu, gizlilik ve küreselleşme konularında oldukça verimsizdir. Tüm bunların sonucu olarak, tekil merkezli yönetilen kimlik sisteminden uzaklaşılarak kendi kimlik yönetimine izin veren ve daha güvenli bir sistemin gerekliliği ortaya çıkmıştır.

Blokzincir teknolojisinin geçmişten günümüze olan gelişimi bize kişiler ve kurumlar arasındaki ilişkiyi yeniden yapılandırma şansı sunmuştur. Bu teknoloji, merkezinde yatan güven mekanizması sayesinde işlemleri güvenli bir şekilde ilgili ağ üzerinde yapmamıza imkan tanımaktadır. Sayısal kimlik yönetim sistemleri olarak blokzincir teknolojisi; kimlik sahiplerine güven, paylaşım özgürlüğü ve kontrollü veri paylaşım imkanı sağlayarak, kimlik sahiplerinin kendi kimliklerini tamamen kendilerinin yönetmesine imkan sağlamaktadır.

3.2. GELENEKSEL KİMLİK SİSTEMLERİNDEKİ GEREKSİNİMLER VE ZORLUKLAR

Geleneksel kimlik yönetim sistemlerinde, değişime ve gelişime açık alanları dört farklı başlık altında toplayabiliriz; kullanılabilirlik, mahremiyet, güvenlik ve küreselleşebilirlik. Bu hususlar aşağıda alt başlıklarda kısaca açıklanmıştır.

3.2.1. Kullanılabilirlik

Kullanıcılar internet servislerini kullanabilmek için yaygın olarak hesap oluşturmak, bunun için de kullanıcı adı/e-posta ve şifre birleşimleri kullanmaktadırlar. Bu kullanıcı adı ve şifre tanımlama işlemi, bilgilerin merkezi bir veri tabanında saklanabilmesi ve ihtiyaç halinde erişilebilmesi içindir. Bu yöntem, farklı sistemler için farklı kullanıcı adı ve şifre ihtiyacı gibi karmaşalar doğurmaktadır. Bu durumun doğal sonuçlarından birisi de güvenilirlik amaçlanırken unutulan veya kaybolan şifreler ile özel bilgiye erişimin riske girmesidir.

Uzaktan erişim sağlanan sistemlerde, genellikle kullanıcıların kendi kişisel bilgilerine erişebilmek için, bu ve benzeri gizli soru ve cevap yöntemleriyle kendilerinin o bilginin sahibi kişi olduğunu ispatlaması gerekmektedir. Sonuç olarak, kullanıcılar kendi bilgilerini güvenli şekilde saklamak ve kendilerinin bu bilginin sahibi olduğunu ispatlamak için oldukça zaman, çaba ve para harcamaktadırlar. Centrify tarafından hazırlanan raporda, 2014 yılında 500 çalışanlı küçük şirketlerin dahi şifre işlemleri için harcadığı yıllık zaman, çaba ve ücretler toplamının 200.000 $'ı bulduğu ifade edilmektedir.

3.2.2. Mahremiyet

Kimlik bilgileri bireyleri temsil ederler. Buna rağmen geleneksel yöntemler, kişilerin hassas kimlik bilgilerini 3. parti kuruluşlarda saklamak üzerine kuruludur. 3. parti kuruluşlar, bu bir web sitesi, bir şirket veya bir devlet olabilir, kimlik bilgilerini merkezi veri tabanlarında saklarlar. Genellikle ilgili işlem için, telefon numarası, anne kızlık soyadı, kimlik numarası gibi ihtiyaç duyulandan fazla hassas bilgi talep etmektedirler. Kimlik bilgilerine ait bu hassas detaylar saldırılara açık tekil veri tabanlarında, denetimsiz, pazarlama amaçlı kullanılmaya müsait ve izinsiz şekilde birçok kurum ve kuruluş tarafından saklanmaktadır.

3.2.3. Güvenlik

Geleneksel yöntemlerde, her servis sağlayıcı kimlik doğrulaması için kişilerin hassas kimlik bilgilerinin bir kısmını kendi veri tabanlarında saklarlar. Günümüzde gerçekleşen siber saldırıların bazıları, bu bilgilerin ele geçirilmesi üzerine kurgulanmaktadır. Herhangi bir olası kimlik bilgisi ihlali, hem kimlik sahipleri hem de ilgili işletmeler açısından büyük sorunlar yaratacaktır. Javelin'in yayınladığı Kimlik Dolandırıcılığı çalışmasına göre [1], sadece 2017 yılında ABD'de de 16 milyon kimlik bilgisi kimlik dolandırıcılığından etkilenmiş ve sonuçta toplamda 16 milyar $'lık bir zarar oluşmuştur. Bu değer, önceki 8 yılda edilen zararlar toplamından daha fazladır.

Tablo 3.1'de bazı kimlik bilgisi dolandırıcılıkları verilmiştir. Yıllara göre incelendiğinde, ekonomik zararda her yıl bir artış söz konusudur. Herjavec Group şirketinin raporuna göre dünya genelindeki kimlik hırsızlığından kaynaklanacak zararın 2021 yılında 6 milyar $ olacağı tahmin edilmektedir [2].

Tablo 3.1. Birleşik Devletler Bazlı Kimlik Dolandırıcılığı ve Zarar Tablosu. ([3]'ten Alınmış ve Güncellenmiştir)

Tarih	Olay
2020	ABD kaynaklarına göre kimlik hırsızlığı sayısı bir önceki yıla oranla %45 artarak 4.8 milyona ulaşmıştır. Bu yıla ait kimlik dolandırıcılıklarının toplamda 4.8 milyar dolarlık bir zarara sebep olduğu belirtilmiştir.
2019	Sadece ABD'de, resmi kayıtlar geçen 3.3 milyon kimlik hırsızlığı ve dolandırıcılık olayı yaşanmıştır.
2017	Tahmini olarak milyonlarca ,çalışanın kimlik bilgisinin bulunduğu OneLogin firmasının veritabanına erişilmiştir.
2016	ABD kaynaklı finansal sektörde gerçekleşen kimlik dolandırıcılığında, 2015'te 13.1 milyon $, 2016 yılında 15.4 milyon $ zarar edilmiştir.
2016	600'den fazla veri hırsızlığında 21 milyondan fazla kimlik bilgisi sızdırılmıştır.
2016	Yahoo! firmasının, kullanıcı kayıtlarında 2013'ten 22016'ya kadar devam eden sızmada 1 milyardan fazla insanın kimlik bilgilerine ulaşılmıştır.
2016	Yahoo! firmasındaki bir diğer ihlalde, 500 milyondan fazla kullanıcının kayıtları 2014 yılından itibaren 2 yıl boyunca sızdırılmıştır.
2015	180 ayrı olayda, 175 milyondan fazla kullanıcı kaydı ihlal edilmiştir.
2015	Ashley Madison veri ihlali olayında, 33 milyon kullanıcının bilgileri, çalınmış ve yaklaşık 850 milyon $'a mal olmuştur.
2015	80 milyon sigorta kaydının çalındığı Anthem Sağlık Sigortası Şirketi olayında, tahminen 100 milyon $ ile 8 milyar $ arasında bir zarar ortaya çıkmıştır.
2014	eBay'de 145 müşteri hesabı ele geçirilmiş ve 200 milyon $ zarar açığa çıkmıştır.
2014	Home Depot veri ihlali olayında 50 milyondan fazla kayıt yasadışı olarak ele geçirilmiştir.
2014	Target veri ihlalinde 40 milyondan fazla kayıt çalınmıştır.
2014	JP Morgan şirketinin veritabanındaki veri sızıntısı sebebiyle 76 milyon kişi ve 7 milyon kurum verisi çalınmış, 1 milyar $'ın üzerinde bir maddi zarara sebep olmuştur.

3.2.4. Küresellik

Küresel açıdan bakıldığında, kimlik doğrulama ve kayıt ispatı kurumsal bir konudur ve sınırları uluslararası boyutlardadır. Kişilerin uluslararası seyahatlerinde kimlik ispatı genellikle pasaport ve benzeri dokümanların fiziksel ispatı gibi geleneksel yöntemlere dönmektedir. Buna ek olarak, üniversite diploması veya eğitim sertifikaları gibi bazı kaynak belgelerin ispatı, aracı kuruluş ihtiyacı ve süre gibi birçok zorluklar getirmektedir. Örnek olarak Hindistan'daki bir üniversiteden mezun olan bir kişi, ABD'deki kurum ve kuruluşlara bu mezuniyetini ispatlayabilmek için farklı 3. parti kuruluşlara ihtiyaç duymakta ve bu doğrultuda zaman, çaba ve ekonomik açıdan kayıplara uğramaktadır.

3.3. KONSEPTLER VE TANIMLAR

3.3.1. Blokzincirine Genel Bakış

Blokzincir, bloklar içerisine gömülen işlemler vasıtasıyla, kıymetli bilginin saklanması ve sahipliği kavramlarını dağıtık kayıt defterlerinde tutabilmenin yegane yoludur. Blokzincir ağı, birbirine bağlı durumda bulunan eş makineler ağından (peer-to-peer) oluşur ve her eş tüm kayıtları aynı şekilde tutmakla yükümlüdür. Dolayısıyla aynı kayıt defteri, tüm ağdaki cihazlarda aynı şekilde saklanmaktadır. Geleneksel P2P ağlarının aksine blokzincir ağındaki bilgisayarlar birbirinden bağımsız hareket etmek yerine tam eşzamanlı halde ve her işlem için, demokratik bir yöntemle mutabakat mekanizmaları çalıştıracak bir uzlaşma oluştururlar. Blokzincir, ilk olarak Bitcoin ürününün altyapısında kullanılmıştır. Bitcoin, sayısal bir değer yaratmak ve bunu aracı kurum ve kuruluşlar olmadan transfer etmeyi amaçlayan ilk kripto para ürünüdür [4].

Bir blokzincir ağı, kayıt defterinde işlemler transfer edilen sayısal değerlerin sahipliğini ve bu değerlerin transferini barındırır. Bitcoin sisteminde bu işlemler "Ödeme" veya "Transfer" olarak geçer ve tüm işlemler sayısal değerler dahil kullanıcılara asimetrik şifreleme ile atanmıştır. Blokzincir ağlarında, işlemlerin gerçeklenebilir olduğunu kontrol edip bu işlemleri ilgili bloklarda kayıt altına almaktan sorumlu olan ve genelde "Madenci" olarak bilinen bilgisayarlar mevcuttur. Bir işlem, işlem sahibi tarafından blokzincir ağında

yayınlandığında üzerinde sahibine, alıcısına, değerine, işlem zamanına benzer karakteristik özellikleri barındırır ve sayısal imzalar ile imzalanarak bu özellikler değiştirilemez hale getirilir. Yeni yayınlanmış işlemler, işlem havuzuna alınır ve burada doğrulanarak bir blok içerisine yerleştirilmeyi, böylelikle hayata geçmiş olmayı beklerler [5].

3.3.2. Uçtan Uca Ağlar

Uçtan uca ağlar, blokzincir protokollerinin temelindeki dağıtık defter kavramını oluşturan en temel yapıdır. Bu ağ yapısı, geleneksel sunucu-istemci mimarisinin aksine her cihazın bir sunucu ve aynı zamanda bir istemci olarak çalışabilmesi prensibine dayanır. Blokzincir protokollerinde uçtan uca ağların tercih edilmesinin sebebi, merkezi yapıdan uzaklaşılmaya çalışılması ve tek bir noktaya bağlı olarak yaşanacak sorunların sistem akışını sekteye uğratmamasıdır. Böylelikle ağa dahil olan herhangi bir kaynak erişilemez olması durumunda, diğer kaynaklar üzerinden sistem sağlıklı bir şekilde devam ettirilebilecektir.

3.3.3. Açık Anahtarlı Kriptografi

Açık anahtar kriptografisi (asimetrik şifreleme) [6], açık ve gizli anahtar eşlerini temel alan ve onyıllardır başarıyla kullanılmakta olan bir şifreleme yöntemidir. Asimetrik şifrelemede genel olarak mesajları iki ayrı yöntemle şifreleme ve açma metodu kullanılabilir. Bir mesaj, açık anahtarla herkes tarafından şifrelenip sadece gizli anahtar ile açılabilir, bu işlemin tam tersi de mümkündür. Gizli anahtar, sadece açık-gizli anahtar çiftinin sahibinde bulunması gereken anahtardır. Açık anahtar ise açıkça istenildiği şekilde paylaşılabilir.

3.3.4. Mutabakat Mekanizmaları

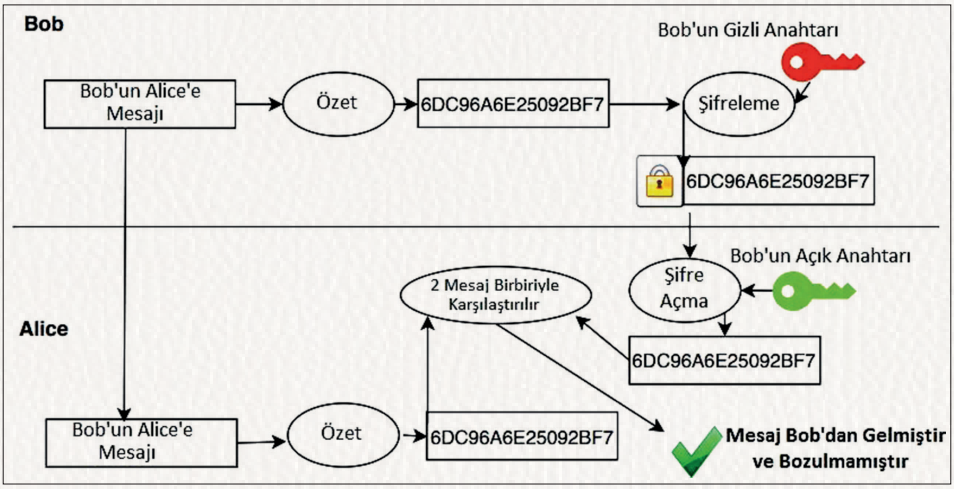
Mutabakat mekanizmaları temelinde, blokzincir protokolü üzerinde yeni bloğun güvenli bir şekilde zincire eklenmesi ve güvenliğin sağlanması amacıyla kullanılır. Daha önceden bahsedilen madenci makineler, ağ üzerinde yayınlanan işlemleri doğrular ve çeşitli özetleme ve kriptografi metodlarıyla içerisinde zaman damgası, merkle kökü ve bir önceki bloğa ait özet bilgisi olacak şekilde yeni bloğa dahil edilirler [5]. Ağ üzerindeki konsensüs mekanizması, bloğu onayladıktan sonra artık zincirin son bloğu yeni eklenen blok olacaktır. Örnek olarak işin ispatı (proof-of-work) mekanizması bitcoin ağında mutaba-

kat mekanizması olarak çalışmaktadır. Bu yöntem, madenci makinelerin yeni bloğu zincire ekleyebilmek için çok ağır bir olasılık problemini kriptografik yöntemler kullanmak suretiyle ilk önce çözebilmek için birbiriyle yarışması üzerinedir. Bir kez blok sağlandığı zaman, tüm ağa yayılır ve sağlamasının doğru olduğunu gören tüm bilgisayarlar tarafından kendi veritabanlarına eklenir. Artık son blok yeni yayınlanmış blok olacaktır. Kendine has zincir ve dağıtıklık mekanizması, blokzinciri teknolojisini geçmişi değiştirilemez ve bozulmaya dayanıklı hale getirmektedir.

Geçmiş kayıtlar üzerindeki herhangi bir işlemin en küçük bir değeri değiştirilse dahi, işleme ait özet bilgisi bozulacaktır. Bu bozulma bloğun merkle kökünün bozulmasına, bunun sonucunda bloğun özet bilgisinin bozulmasına ve dolayısıyla ilgili bloktan sonra gelen tüm blokların bozulmasına sebep olacaktır. Bu bozulma, tüm zincirin tekrar doğru hale getirilebilmesi için, bozulmanın olduğu blok dahil sonraki tüm bloktaki işlemlerin tekrar doğrulanmasını ve her blok için tekrar zorluk seviyesi yüksek matematik probleminin çözülmesi ihtiyacını doğurmaktadır. Ayrıca ağ üzerindeki bilgisayarların %50'den fazlasının bu işlemleri ve zincirin yeni halini onaylaması gerekmektedir. Bu da teorik olarak mümkün görünse de pratikte oldukça zor bir işlemdir. Buradan yola çıkarak, blokzincir teknolojisinin geçmişe yönelik değişikliklere çok sıkı bir şekilde kapalı olması, bu teknolojiyi aynı zamanda oldukça güvenli kılan bir yapıdır.

3.3.5. Sayısal İmzalar

Açık dijital imzalar; P2P ağlarda mesajın sahibi kullanıcıyı kesinleştirmek ve mesajın bütünlüğünü garanti altına almak için kullanılır, açık anahtar şifrelemesi ve özetleme fonksiyonlarından faydalanır. Örnek olarak, Şekil 3.1'de Bob Alice'e dijital imzalardan faydalanarak bir mesaj gönderir. Bob, mesajın orijinal halinin özet çıktısını gizli anahtarıyla şifreler. Bu işleme dijital imzalama denir. Bob, orijinal mesajı dijital imza ile beraber Alice'e gönderir. Alice aynı özet fonksiyonuna orijinal mesajı girdi olarak sokar ve elde ettiği çıktı ile Bob'un gönderdiği imzayı kendi gizli anahtarıyla çözerek şifrelenmiş özet çıktısını karşılaştırır. Eğer bu iki özet fonksiyon çıktısı birbirine eşit ise Alice, mesajın kendisine iletildiği esnada 3. kişiler tarafından değiştirilmediğine emin olur. Blokzincir teknolojisinde bu metot, sahipleri tarafından ağa yayınlanan işlemlerin başka kimse tarafından değiştirilmediğine ve orijinal olduğuna emin olunmak için kullanılır.



Şekil 3.1. Dijital İmzalar ile Veri Gönderimi

3.3.6. Dağıtık Tanımlayıcılar

Dağıtık tanımlayıcılar (DID); merkezi otoriteden bağımsız ve tamamen sahibi tarafından yönetilen, kişi kurum veya kuruluşlara verilmiş, belirli standartları olan, kriptografik olarak ispatlanabilir, uluslararası boyutta tekil ve kalıcı tanımlayıcılardır. Dağıtık tanımlayıcılarda, açık-gizli anahtar çiftlerinin kullanıldığı açık anahtar şifrelemesi uygulanır. Her bir dağıtık tanımlayıcı, sadece tanımlayıcının sahibi tarafından kullanılabilen gizli anahtar ile kontrol edilir. Dağıtık tanımlayıcılar kimlik sahibine, diğer kullanıcılar ile istediği kimlik bilgilerini paylaşabileceği özel bir kanal açmayı da sağlamaktadır. Bu tanımlayıcılar, aynı zamanda kullanıcıların kendisini tanıtması amacıyla kullanılır. Her bir dağıtık tanımlayıcılar, dağıtık tanımlayıcıya ait anahtarları barındıran bir dağıtık tanımlayıcı nesnesine dönüştürülebilir. Bu nesne açık olarak bir meta veri barındırır ve doğrudan bir başka dağıtık tanımlayıcı sahibiyle iletişime geçmek ve bilgi paylaşmak için kullanılır.

3.3.7. Doğrulanabilir Kaynaklar

Doğrulanabilir kaynaklar, kimlik sahiplerinin belli konularda sahip olduğu lisanslar ve yetiler için birer kanıt olarak düşünülebilir. Günlük bireysel kullanımda oldukça yaygındırlar. Kaynaklar için sürücü lisansları, üniversite dip-

lomaları, pasaportlar en basit örneklerdir. Doğrulanabilir kaynaklar, kimlik sahiplerine ait, makineler tarafından okunabilir, gizlilik dostu, kriptografik olarak güvenli kaynaklardır. Doğrulanabilir kaynaklar tam bağımsız kimlik yapısını destekler, böylelikle kullanıcılar istediği belgeyi istediği şekilde kanıt olarak sunabilir. Genellikle üçüncü bir kurum tarafından tastiklenmeye muhtaçtırlar ama aynı zamanda oldukça güvenilirdir. Onaylanma işleminde, dijital imzalardan faydalanılır. Bir dağıtık tanımlayıcı sahibi onaylayıcı, bir doğrulanabilir kaynak oluşturur ve kendi dijital imzası ile imzalar. Bu imza, onaylayıcının varlığını her zaman garanti edecektir. Onaylayıcılar, belge düzenleyicileri için bir güven objesidir.

3.3.8. Akıllı Sözleşmeler

2008 yılında Satoshi Nakamoto tarafından ortaya atılan bitcoin blokzinciri, günümüz blokzincirlerinin ilk sürümü olarak düşünülebilir. Bu protokol, sadece üzerinde tanımlı olan bitcoin adlı sayısal değer biriminin transferine müsaade ederken, daha sonra özellikle Ethereum blokzincir protokolüyle birlikte, internetin çok büyük ve tek bir bilgisayara dönüştürmüştür. Bu ve benzeri protokoller sayesinde; kullanıcılar, lojik işlemlerin çalışmasına müsaade etmek üzere özelleşmiş akıllı sözleşmelerini (kontratlarını) ağ üzerinde çalıştırılabilir veya yapabilir hale gelmiştir.

Üzerinde akıllı sözleşmelerin çalışabilmesinin yaygınlaşmasıyla birlikte artık blokzincir protokolleri sadece bir değer transfer aracı olarak değil, bunun çok daha ötesinde programlanabilir bir yapıya kavuşmuştur. Böylelikle çağımızın en hızlı gelişim sağlayan dApp adı verdiğimiz dağıtık uygulamaların önü açılmıştır.

3.4. BLOKZİNCİR TEMELLİ KİMLİK SİSTEMLERİ

Bu bölümde, veri paylaşımına izin veren, tam bağımsız, dağıtık tanımlayıcılar ve doğrulanabilir belgelerden faydalanan ve kimlik sahiplerinin güven içinde kurum ve kuruluşlarla bağlantıya geçebileceği blokzincir tabanlı sayısal kimlik çözümleri incelenmektedir.

Günümüzde, sayısal kimlikler üzerine yoğunlaşan birtakım çalışmalar mevcuttur. Dahası, devletler tarafından yapılan bazı araştırmalarda rapor edildi-

ğine göre [7]–[9], blokzincir, sayısal kimlik alanında devrim yapma potansiyeli olan bir teknolojidir. Hyperledger ürünü, Linux Foundation tarafından ortak ve ücretsiz kullanım amaçlı, açık kaynak kodlu bir blokzincir çerçeve ve araç geliştirme girişimidir. Girişim üyeleri, bankacılık, teknoloji, danışmanlık, perakende, devletler ve akademi dünyası gibi farklı endüstrilere ait aktörlerdir.

Hyperledger Fabric [10], [11]; modüler tasarlanmış, özel ve izinli blokzincir uygulamaları oluşturmak üzerine kurulu bir girişim olup, bu temelli blokzincir sistemlerinde, her biri x.509 dijital sertifika standartları [12] dahilinde sayısal kimlik sahibi olan son kullanıcı uygulamalar, değer sahipleri, sıralayıcılar ve eşler gibi aktörlere sahiptir. Kimlikler, değer sahipliği dışında, kaynak ve erişim izinlerinin yönetilmesi de Hyperledger Fabric üzerinden yönetilir. Hyperledger Fabric'te aynı zamanda, aktörlere ait kimlik sahiplerinin birimleri, kuruluşları ve izinleri gibi ek özellikleri de içeren kavramlar vardır.

Hyperledger Fabric'te bulunan doğrulanabilir sayısal kimlikler, ağ dahilindeki sertifika otoriteleri (CAs) tarafından sağlanır. Sistem üzerinde aynı zamanda, hangi sayısal kimliğe ait sertifikaların iptal edildiğinin tutulduğu bir Sertifika İptal Listesi (CRL) bulunmaktadır. Hyperledger Fabric aynı zamanda üzerinde gömülü olarak, blokzincir ağının yasal sayısal kimliklerini tanımlayan ve sayısal kimliklerin ait olduğu kuruluşları temsil eden Üyelik Servis Sağlayıcısı (MSP) bulundurur. Ürünü sıra dışı kılan bir diğer özellik ise izne bağlı olarak gizli bilgi paylaşımının mümkün olduğu özel kanallara elverişli yapısıdır.

Bir diğer Hyperledger projesi, temel olarak bağımsız dağıtık kimlik odaklı, güvenliği ön planda tutan, açık ve izinli bir blokzincir platformu sunan Hyperledger Indy [10] ürünüdür. Hyperledger Indy;

- ürün kimlik özellikleri, kütüphaneler, araçlar ve blokzincir protokolü üzerinde dağıtık kimlik oluşturabilmek için yeniden kullanılabilir bileşenleri bulunan bir setten oluşmaktadır.
- blokzinciri üzerinde veri tutulmasını desteklemektedir.
- kimlik sahiplerinin, kendi kimlik bazlı bilgilerini kendilerinin saklamasını mümkün kılar.
- bireylerin kişisel bilgilerini tutmak yerine kimliklerine bir bağlantı saklar.

- kimlik sahipleri kendi kişisel verilerine erişimi kontrol edebilirler.
- kimlik modeli, dağıtık tanımlayıcıları ve doğrulanabilir belgeleri desteklemektedir.

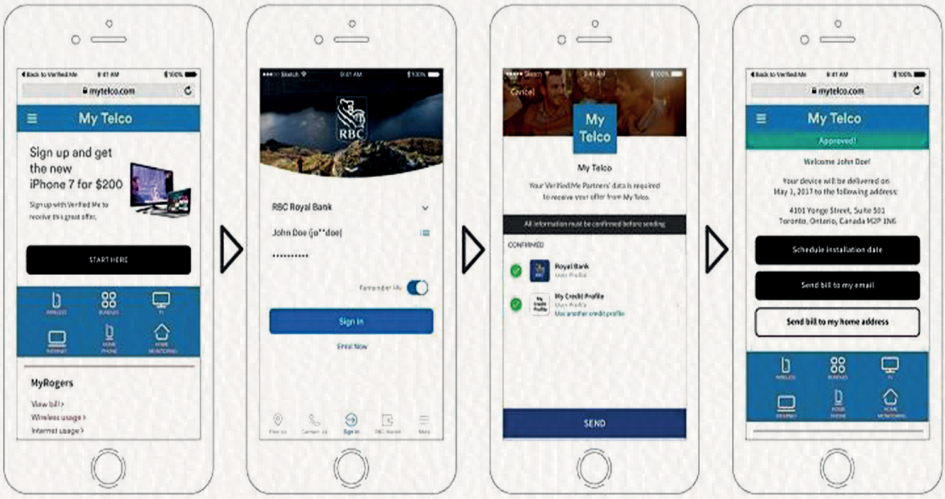
3.4.1. İzine Dayalı Veri Paylaşımı

Hali hazırda dağıtık defterler üzerine bir diğer sayısal kimlik ürünü de, Hyperledger Indy platformu üzerine kurulu olan Sovrin [13]'dir. Sovrin ağı, herkes için bir sayısal kimlik projesi oluşturulması amacıyla yola çıkıldığı için herkese açıktır. Yönetim modeli izinlidir. Bu yüzden blokzincir protokolü üzerindeki düğümler, "steward" adı verilen özel yapılar tarafından yönetilir. Sovrin dağıtık tanımlayıcılar ve doğrulanabilir belgeleri kullanır. Ölçeklenebilirliği arttırmak için Sovrin 2 tip blokzincir düğümü kullanır. Bunlar;

- defterlere kayıt ekleme yetkisi olan bir doğrulayıcılar ağı ve
- dağıtık defterlerin yazmaya müsait olmayan bir kopyasını elinde tutarak tüm işlemleri izleyebilen daha büyük bir kullanıcı ağıdır.

Sovrin herkese açık bir blokzincir protokolü kullandığı için, ağ üzerinde hangi bilginin tutulduğunun bilinmesi oldukça önemlidir. Sovrin'e göre, dağıtık tanımlayıcılar ve doğrulama anahtarları ve bitiş noktaları, şemalar ve doğrulanabilir belgeler, veri paylaşımının rıza ispatları, halka açık belgeler, iptal kayıtları ile ilintili DID dökümanları ağ üzerindeki paylaşımlı defterlerde tutulur, fakat herhangi bir tipteki gizli veri ve gizli bilgilerin ispat kayıtları tutulmaz [14].

SecureKey [15], [16] Müşterini Tanı (KYC) işlemlerinde kişilerin kimliğini devlet, telekomünikasyon veya banka gibi çevrimiçi ve güvenilir kuruluşlara zaten tanıtmış olduğu bilgilerini, bu kurum ve kuruluşlar arasında paylaşımını yönetebildiği bir sayısal kimlik yönetimi uygulama ağıdır. SecureKey, kişisel bilgilerin gizlilik çerçevesinde ve tarafların açık rızasıyla paylaşıldığını garanti eder. Şekil 3.2 bu paylaşıma dair bir senaryo görselleştirmektedir. Bu senaryoda, kullanıcının rızası dahilinde kimlik bilgileri paylaşılmaktadır. Kullanıcı bir telekomünikasyon firmasının web sayfasını ziyaret eder, istediği cihazı seçer, banka tarafından kredi puanlama sonucunun telekomünikasyon şirketiyle paylaşmasına razı olduğunu belirtir ve telekomünikasyon şirketi bankanın paylaştığı bilgiyi kullanabilir.

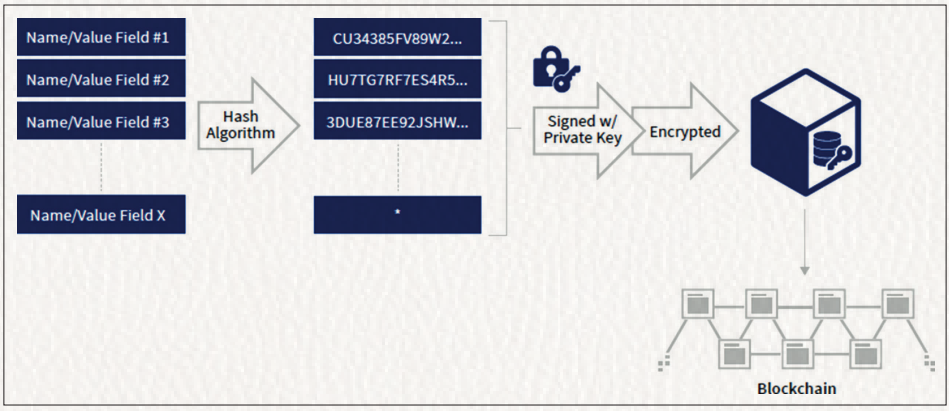


Şekil 3.2. SecureKey Kullanıcı Tecrübesi [15].

SecureKey, kurum ve kuruluşların iştirakiyle oluşturulan ortak uzlaşma tarafından yönetilen Hyperledger Fabric tabanlı izinli bir blokzincir protokolü kullanmaktadır. Blokzincir protokolü üzerindeki dağıtık defterlerde, ilgili belgelerin ispatları, kaynakları ve izinler saklanır. Kullanıcıların kimlik bilgileri, yönetici ve güvenilir kuruluşlar tarafından saklanır. SecureKey aynı zamanda, bilgi paylaşımını talep eden tarafın, bilgi paylaşımı yapacak olan tarafa belirli ücretler ödemesine de imkan sunarak, bilgi paylaşımını teşvik edici bir özelliğe de sahiptir. Uygulama mimarisi, veri sağlayıcının hiçbir şekilde kullanıcısının eriştiği servisi bilmediği ve veri talep eden tarafın veriyi tam olarak hangi veri sağlayıcıdan aldığını bilmediği fakat sadece verinin güvenilir bir veri sağlayıcı tarafından geldiğini bildiği bir üç taraflı kör paylaşım sistemine imkan kılmaktadır.

Bu konuyla ilgili, blokzincir üzerinde geliştirilmekte olan bir diğer uygulama da Shocard [3]'tır. Kimlik detayları, "ShoCard" isminde sayısal bir dosyadadır ve bu dosya kimlik sahipleri tarafından genellikle kimlik sahibinin mobil cihazında saklanır. Kimlik sahiplerinin, kendi kimlik bilgilerini yönetebilmesi için bir açık-gizli anahtar çiftinden faydalanılır. ShoCard sistemi öznitelik paylaşımını mümkün kılmaktadır. Bireylere ait kimlik detayları, farklı öznite-

likler altına kırınım gösterir. Her bir öznitelik özetleme fonksiyonundan geçirilir, gizli anahtarla imzalanır ve saklanması için blokzincir ağına gönderilir. Şekil 3.3'de, ShoCard'ın öznitelik bazlı kendinden sertifikasyon sistemini görselleştirmektedir.



Şekil 3.3. ShoCard'ın Öznitelik Temelli Kendinden Sertifikasyon Şeması [3]

Shocard sistem mimarisi, çoklu açık ve özel blokzincir ağların katmanlarından oluşmaktadır ve ShoCard çekirdek servisleri blokzincir ağının üzerindeki katmanda inşa edilmiştir. Bu mimari aynı zamanda Sidechain'ler ve sunucu önbelleği, Shocard servisleri üzerinde çalışan uygulama katmanı, SDK ve mobil uygulamaları içermektedir.

Konuya ilişkin bir diğer çalışma da, Walmart'ın bireylerin sayısal sağlık verilerini koruma ve yönetme üzerine aldığı patenttir [17]. Bu sistemde kişisel sağlık verileri blokzincir protokolü üzerinde yönetilir. Bireyler, kendilerine ait asimetrik anahtar çiftlerini kullanarak kendi verilerine erişebilir. Sistem özellikle hastanın bilinçsiz veya yetersiz kaldığı, alerji gibi doktorun bilgilendirilmesi durumunda tedavinin seyrini değiştirebilecektir ve acil durumlarda oldukça kullanışlıdır. İlgili sistemde, açık anahtar ve özel anahtarın şifrelenmiş bir haliyle (hastanın biyolojik bir özelliğiyle şifrelenir) birlikte giyilebilir bir cihazda taşınır. Hastanın açık anahtarı ve gizli anahtarın şifrelenmiş formu RFID aracılığıyla bir okuyucu sayesinde okunabilir. Ayrı bir biyometrik tarayıcı cihaz, şifrelemede kullanılan parmak izi, retina vb.

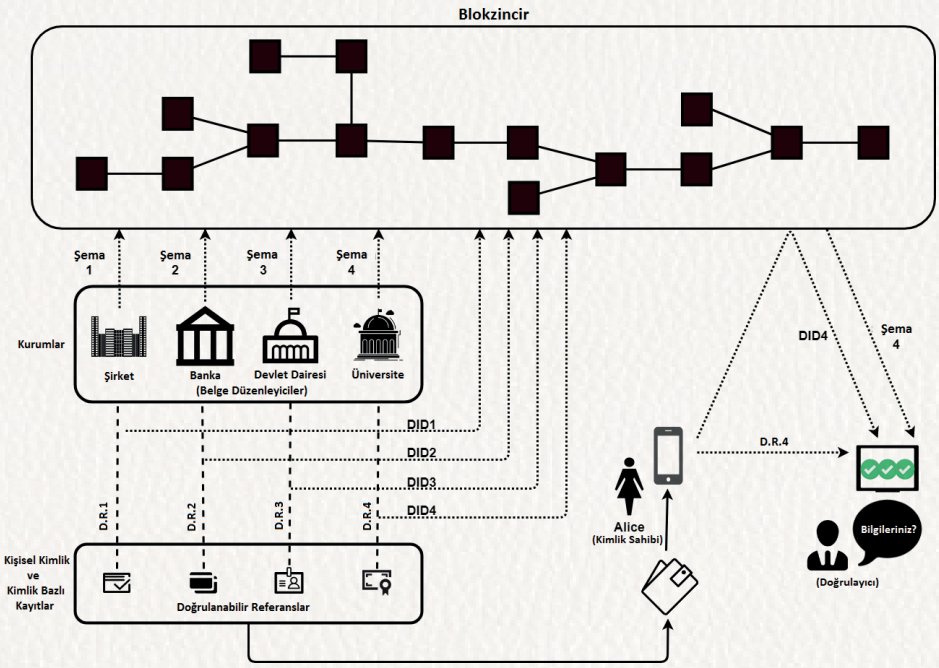
biyolojik veriyi okur ve böylelikle şifrelenmiş gizli anahtar hastanın biyolojik imzasıyla deşifre edilir (örneğin parmak izi, iris, retina, yüz tanıma vb). Böylelikle asimetrik anahtar çifti, blokzincir üzerinde bulunan hastanın medikal verisine ulaşılmasını sağlayacaktır.

Bir diğer ürün olan Bitnation [18], blokzincir temellerine oturtulmuş coğrafyadan ve devletlerden bağımsız olarak bir küresel dünya vatandaşı kimliği kavramına odaklanmıştır. Bitnation, dünya vatandaşlığı, blokzincir pasaportu, evlilik sertifikaları ve mülteciler için acil durum belirleyicileri gibi hizmetler sağlar. Bitnation kimliği, bireyin belirli bir zaman diliminde ve bir yerde var olduğunu kanıtlamayı gerektirir ve bunun için bu bilgi diğer bir kullanıcı grubu tarafından kriptografik olarak imzalanarak varlığı tescillenir. Bitnation, Ethereum [19] ağını, özetleme fonksiyonlarını, dijital imzaları ve akıllı sözleşmeleri kullanır.

Bitnation çalışmalarını, devletin yürüttüğü e-ikamet projesi dahilinde Estonya ile beraber yürütmektedir [20]. Proje, Estonya vatandaşı olmayan veya Estonya'da ikamet etmeyen fakat iş sahipliği, dijital sözleşme (kontrat) imzalama, bankacılık, vergilendirme, ödeme süreçleri veya noterlik gibi hizmetlerden faydalanmak isteyenlere yardımcı olmayı amaçlamaktadır.

Uport [21], bir diğer halka açık ve Ethereum tabanlı bir blokzincir üzerinde bulunan, kendinden bağımsız kimlik projesidir. Sovrin gibi, Uport da veri paylaşımını, dağıtık kimlikleri ve doğrulanabilir belgeleri destekler ve gizli bilgileri halka açık defter kayıtlarında saklamaz. Buna rağmen, Sovrin'in tersine halka açık bir blokzincir protokolünü kullanmaktadır ve ağ üzerindeki her işlem bir ücrete tabidir.

Türkiye'de de blokzincir temelli bütünleşik bir kimlik yönetim sistemi projesi geliştirilmiştir [22], [23]. Bu sistem, kimlik sahiplerine, sahip oldukları belgeleri istediği kurum veya kuruluşa güven içinde kanıt olarak sunma yetisi (kimlik doğrulama, oturum açma vb.) vermektedir. Böylece, üçüncü parti sertifikaların dahliyle iddiaların tasdiklenmesi daha güvenli hale gelmektedir (tasdikleme, ihtisas durumu ispatı vb.). Bu sistemin amacı, ölçeklenebilir, küresel kullanıma uygun, güvenlik ve gizliliği aynı anda sunabilen bir altyapı hazırlamaktır.



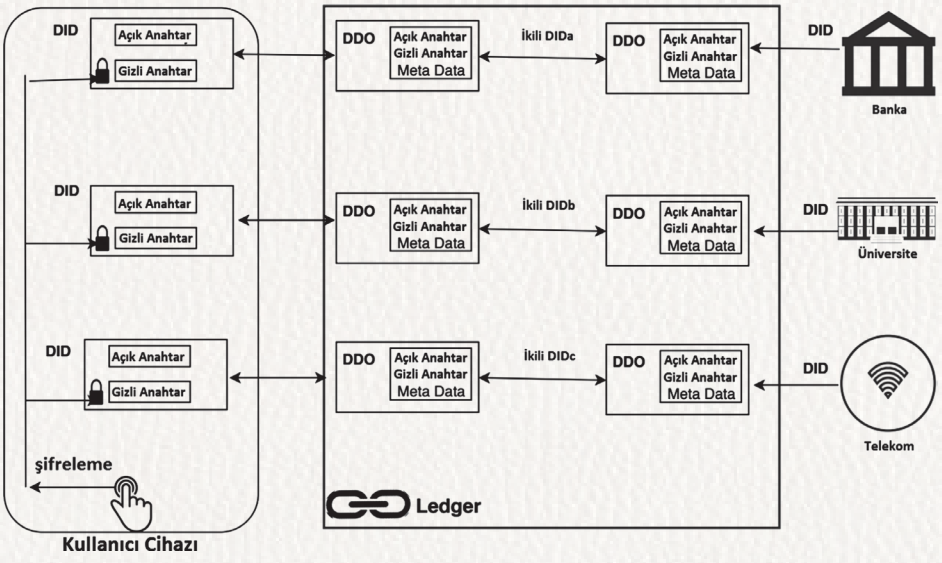
Şekil 3.4. Önerilen Sistemin Genel Çalışma Şeması

Şekil 3.4'de, bu sistemin genel işleyişini göstermektedir. Aydar ve Ayvaz'ın önerdiği sistemde, bireyler ve kurumlar blokzincir ağı tarafından yönetilen dağıtık bir platforma sayısal kimlikleri ile bağlıdırlar [22]. Sistem herhangi bir gizli anahtar veya kişisel veriyi, şifrelenmiş halde olsa bile herkese açık defterlerde saklamaz. Sistem doğrulama amaçlı olarak sadece işlemlerin bir sayısal kanıtını saklamaktadır. Kimlik sahipleri, kimlik bilgilerini tamamen kendileri kontrol ederler ve bu bilgiyi kimle paylaşacağına sadece kendileri karar verirler. Kimlik sahipleri belgelerini, şifreleme yöntemiyle sayısal olarak imzalayarak alabilir, bu belgeleri doğrulanabilir kaynaklar ile ilgili kurumlardan kriptografik olarak doğrulatabilir ve bunları kendi sayısal cüzdanında saklayabilir. Sonuç olarak sistem, merkezi otoriteler üzerinden doğrulanması gereken dokümanların daha hızlı ve güvenli bir şekilde doğrulanmasının önünü açarak, geleneksel yöntemlere göre zaman ve maddi kayıpları düşürerek ve kimlik bilgilerinin bireylerin kendi kontrolünde olmasına yardımcı olmaktadır.

3.4.2. Blokzincirde Kimlik Yönetimi

Blokzincir temelli kimlik yönetim sisteminde, bireyler ve organizasyonlar kendilerini, kendi kontrollerinde olan ve herhangi bir merkezi otoriteye bağlı olmayan, kendinden bağımsız kimliklerle temsil ederler. Sayısal kimlikler aynı zamanda, nesnelerin interneti ağında kimlik doğrulama amacıyla kullanılmak suretiyle mobil cihazlara da yaygınlaştırılabilir. Kendinden bağımsız kimlikler, her biri kimlik sahibi ve diğer kimlik sahibiyle olan bağlantıyı temsil etmek üzere çoklu dağıtık tanımlayıcılardan oluşur.

Farklı dağıtık tanımlayıcılar kullanmanın avantajı, herhangi bir anahtar sahtekârlığı karşısında, sadece ilgili dağıtık tanımlayıcının zarar görmesi diğer kimlik sahipleriyle olan ilişkilerin güvende olmasıdır. Kimlik sahibinin kontrolü altında, her bir dağıtık tanımlayıcı tekildir ve kriptografik olarak doğrulanabilir bir açık-gizli anahtar altyapısına sahiptir. Her bir dağıtık tanımlayıcı, blokzincirde saklanan bir dağıtık tanımlayıcı açıklama belgesine dönüşebilir (DDO). Her bir dağıtık tanımlayıcı açıklama belgesi üzerinde, diğer kimlik sahipleri ile güvenli bağlantı kurulabilmesi amaçlı, ilgili dağıtık tanımlayıcı ile bağlantılı bir açık anahtarı bulunur.



Şekil 3.5. Kendinden Bağımsız Kimliklerin Kullanıcı Cihazlarında Saklanması ve Eşlenik Dağıtık Tanımlayıcıların Görsel Anlatımı

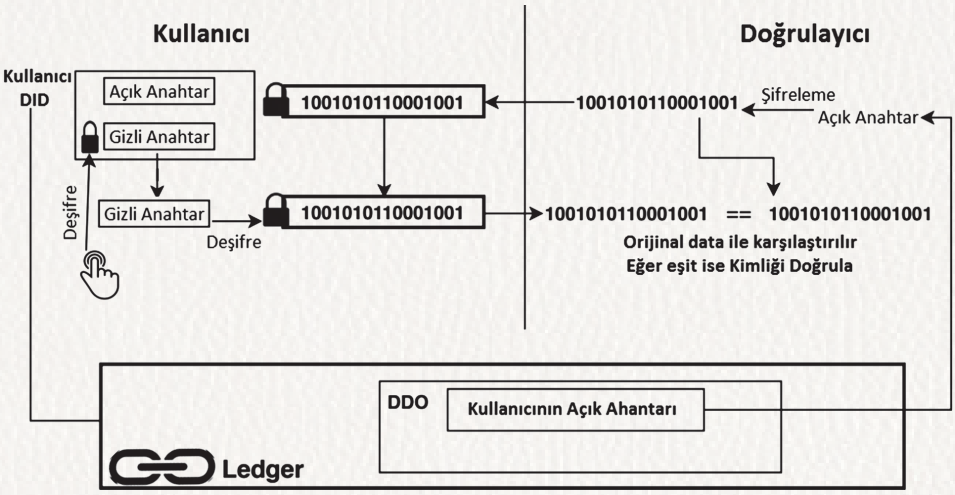
Bağımsız kimliklerdeki dağıtık belirleyicilere ait kriptografik anahtar çiftlerinin detayları, kimlik sahiplerinin cihazlarında tutulur. Açık anahtar şifrelenmemiş halde saklanırken, bu anahtarla ilişkili gizli anahtar şifrelenmiş halde saklanır. Bir gizli anahtar, sadece kimlik sahibinin bir biyometrik anahtarıyla deşifre edilecek şekilde şifrelenebilir. Bu biyometrik anahtar, parmak izi, yüz tanıma, göz-iris-retina desenleri olabilir. Gizli anahtarın şifrelenmesi, kişisel bilgilere erişim için çok faktörlü ve kimlik sahibine özel kimlik doğrulamasını sağlıyor olmalıdır. Şekil 3.5 bağımsız kimliklerin kullanıcı cihazlarında ve ikili dağıtık tanımlayıcılarda nasıl saklandığını göstermektedir.

3.4.3. Kimlik Doğrulama

Kimlik doğrulama, kimlik sahiplerinin kendisinin ilgili kişi olduğunu ispatlama işlemidir. Bu işlem genellikle bireylerin günlük hayatlarında, güvenlik kontrolü amaçlı ve özel hizmetlere erişim için uygulanmaktadır. Bu sistem açık anahtar kriptografisi temelli kimlik doğrulama yöntemini kullanmaktadır.

Kimlik sahipleri, açık anahtarla ilintili olan gizli anahtarın sahibi olduğunu bir şekilde ispatlamak zorundadırlar. Doğrulayıcı taraf (örneğin bir web sayfası), kimliğini doğrulatmak isteyen kullanıcının açık anahtarıyla bir metni şifreler ve kullanıcıya gönderir. Kimlik sahibi olduğunu iddia eden kullanıcı, eğer gerçekten o kimliğin sahibi ise, acık anahtara ait gizli anahtara sahip olacaktır. Bu anahtarla şifrelenmiş metni açabilecektir. Şifrelenmiş metni deşifre ederek doğrulayıcı tarafa geri gönderir. Doğrulayıcı taraf, orijinal metni kontrol eder ve eğer doğruysa kimlik bilgisinin kimliğini doğrular. Şekil 3.6'da, açık anahtar kriptografisi ile kimlik doğrulama sisteminin nasıl işlediğini görselleştirilmiştir.

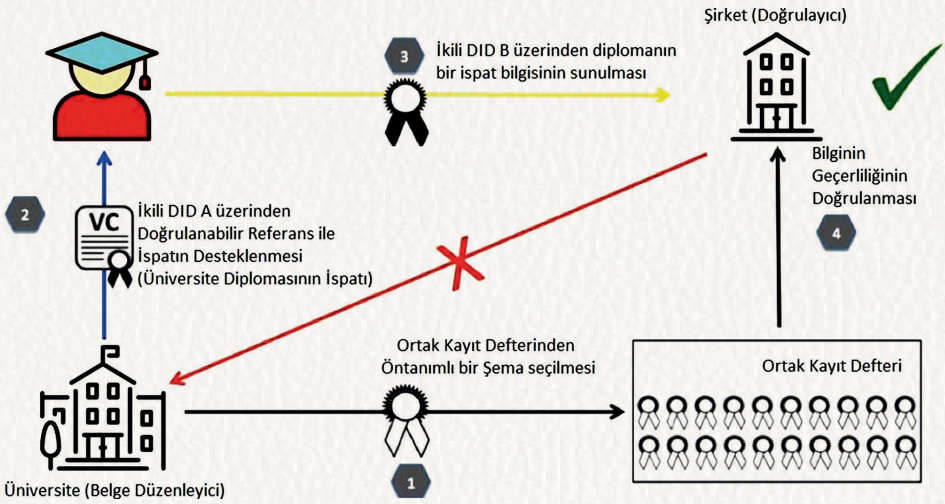
Belgelere ait veriler formatlanmamış metin, grafik veya ön tanımlı belge (şema) formatında olabilir. Sistem, kimlik sahipleri ve üçüncü parti belge düzenleyicileri tarafından kullanılabilir olması için cihazlar tarafından okunabilir hale getirilmesini teşvik etmektedir [22]. Şemalar, belgelerin tanımlanması ve cihazlar tarafından okunabilmesi için oldukça önemlidir. Sistem, şemaların ortak defterlere yayılmasına imkan sağlamaktadır. RDF ve ontolojinin kullanılması ile kişisel bilgi, sağlık bilgisi, üniversite diploması gibi tanımlı şemaların kullanımı birbirine oldukça uygundur.



Şekil 3.6. Açık Anahtar Kriptografisi ile Kimlik Tanıtma

Bir belgenin üçüncü bir parti tarafından düzenlenmesi için, belge düzenleyici öncelikle adına belge düzenlenecek kişinin kimliğini tanımalıdır. Bir kez kimlik tanımlaması yapıldıktan sonra, düzenleyici uygun bir belge tanımı seçer, belgeyi oluşturur, kendi gizli anahtarı ile belgeyi imzalar ve kimlik sahibine bu belgeyi teslim eder. Örnek olarak, sistem kullanılarak, sürücü belgeleri sayısal olarak düzenlenebilir ve doğrulanabilir kaynak olarak kullanılabilir. Bu sebepten dolayı, bir devlet kurumunun sürücü belgesi için ağ üzerinde bir belge tanımı yayınlaması yeterlidir. Belge tanımı, ilgili kişinin ismi, ehliyet numarası, ehliyet tipi, geçerlilik süresi gibi alanları barındıran bir şema barındırmalıdır. Bir belge otoritesi, lisans şemasını ağ üzerindeki ortak defterden alır, bu şema ile oluşturacağı bir belgeyi ilgili sürücü bilgileriyle doldurur, kriptografik olarak belgeyi imzalar ve sayısal bir sürücü belgesini doğrulanabilir kaynak olarak oluşturmuş olur.

Lisans sahipleri, sayısal lisanslarını kendi cihazlarında saklarlar. Otoriteler, ilgili kişinin bu lisansa sahip olduğunu, meşru lisans otoritesi tarafından verildiğini ve geçerli olduğunu doğrulayabilir. Belge tanımları ağ üzerindeki kayıt defterlerinde, indeksli bir şekilde ve aranabilir formatta saklanır. Böylelikle, kullanıcılar ihtiyaç duydukları belgeyi hangi otoriteden edinebileceklerini kolaylıkla bulabilir.



Şekil 3.7. Doğrulanabilir Hak Talepleri Arası Etkileşim Örneği

Doğrulanabilir belgeler, sayısal olarak kimlik sahipleri arasında diğer bireyler ve kurumlar dahil edilerek paylaşılabilir. Şekil 3.7'de, üniversite derecesi ile ilgili bir belge paylaşımı örneği ele alınmaktadır. Örnekte, bir üniversite kendi öğrencisinin başarı seviyesini bir kanıt ile ağ üzerinde paylaşımlı bir şema kullanarak, "A" dağıtık tanımlayıcısı üzerinden doğrulanabilir belge haline getirmektedir. Üniversite mezunu kişi, sayısal belgesini ilgili kurum ile "B" dağıtık tanımlayıcısını kullanarak şirketine sunabilir. Şirket de çalışanının eğitim durumunun geçerli olup olmadığını doğrulanabilir belgelerin elverişli yapısı sayesinde kolaylıkla doğrulayabilir.

Bir doğrulanabilir belge, aynı zamanda bir başka üçüncü parti kuruluşun talebiyle de oluşturulabilir. Bu durumda, kimlik sahipleri kimliklerini belirli kuruluşlar tarafından (belge sağlayıcısı) zaten doğrulatmış durumdadır. Bir diğer kuruluş tarafından (belge talepçisi) kimlik onayı talebi olduğu için, iki kurum arasındaki güvenilir bağlantı kullanılarak belge sağlanması mümkündür.

Bunun yapılabilmesi için, belge talep eden taraf kimlik sahibinin kimliğini doğrular ve kimlik sahibini kimlik doğrulama sistemi üzerinden belge sağlayıcıya yönlendirir. Talep edilen bilgiye göre, belge sağlayıcı belge talepçisine doğrulanabilir belge sağlar. Kimlik sahibi, doğrulanabilir belgesini kendi gizli

anahtarıyla imzalar ve kimlik doğrulama sistemi üzerinden belge talepçisine iletir. Belge talepçisi, kimlik sahibinin ve belge sağlayıcısının dağıtık belge tanımlayıcısını (DID) çağıracak ve ilgili açık anahtarlar aracılığıyla dijital imzalarını doğrulayacaktır.

Tüm bu süreç, çevrimiçi ortamda kimlik sahiplerinin sayısal ortamda kimliklerini kullanması sağlanarak saniyeler içerisinde gerçekleşecektir. Süreç, aynı zamanda Müşterini Tanı (KYC) prensibi gereksinimlerinin sağlanmasına da destek olmakta ve organizasyon ve kuruluşları doğrulanabilir belgelerle birbirine bağlayarak, bilginin sadece sahibi tarafından izin verilen kısmı kadarının paylaşılmasına imkan sunmaktadır. Bu işlemlerde verilen iznin bir kanıtı, katılımcıların hassas bilgileri ifşa edilmeksizin ağ üzerindeki ortak defterde saklanır. Örnek olarak, kimlik sahipleri kimliklerini bankalarına tanıtırsa, finansal bilgilerini çalıştığı banka üzerinden bir başka telekomünikasyon şirketi ile paylaşma imkanı bulmaktadır. Bu işlem Şekil 3.8'de görülmektedir.

Şekil 3.8. Üçüncü Parti Kuruluşlar İçin Doğrulanabilir Hak Talebi Kullanımı

Doğrulanabilir belgeler kimlik sahiplerine verilir ve yine bu kişiler tarafından saklanır. Bir belgeyi doğrulamak, belgenin doğru kimlik sahibine ait olduğunun, doğru kurum tarafından verilmiş olduğunun, belge geçerlilik süresinin dolmamış olduğunun, belgeyi veren kurum tarafından iptal etme durumunun olmadığının teşhisi gibi konuları kapsar. Bu nedenle, sistem üzerinde gecikmesiz (asenkron), kimlik sahiplerinin bilgi gizliliğini koruyacak ve merkezi sistemler tarafından yönetilmeyen (dağıtık), verimli ve etkin bir iptal etme mekanizması gerekmektedir.

Sistem aynı zamanda, dağıtık defterler üzerinde bulunan bir iptal kaydı mekanizması da bulundurmaktadır [22]. Belge vericiler, belgenin iptal edilmesinden sorumludur. İptal kaydı, ilgili belge tanımlarının veya spesifik özniteliklerin iptal olduğunu bildiren kriptografik bir akümülatördür. İptal edilen belge ile ilgili kriptografik bir özet değeri hesaplanır ve iptal kayıtları arasına atılır. Kimlik doğrulayıcılar, her doğrulama esnasında iptal kayıtlarına bu belgeye ait bir özet sonucu atılıp atılmadığını kontrol eder.

Kimlik sahibi, doğrulayıcı kurum ile bir doğrulanabilir belge paylaştığında, bu paylaşım sözleşmesine (rıza bildirisi) ait bir ispat üretilir ve dağıtık defterlerde saklanır. Bir kavram bildiri, kimlik sahibi ve doğrulayıcı tarafından imzalanır, aynı zamanda taraflara ait gizli bilgiler olmaksızın dağıtık tanımlayıcılar ve paylaşılan öznitelik isimleri ve bilgi türlerini barındırır. Konsept bildirisinin ispatı, bildirinin dağıtık defter kayıtlarında tutulan bir kriptografik özetidir. Bu mekanizma sayesinde ilerleyen zamanlarda ihtiyaç duyulması durumunda, doğrulanabilir belgelere ait değişikliğe dirençli ispat kayıtlarının korunması sağlanır.

3.4.4. Blokzincirde Saklanan Veriler

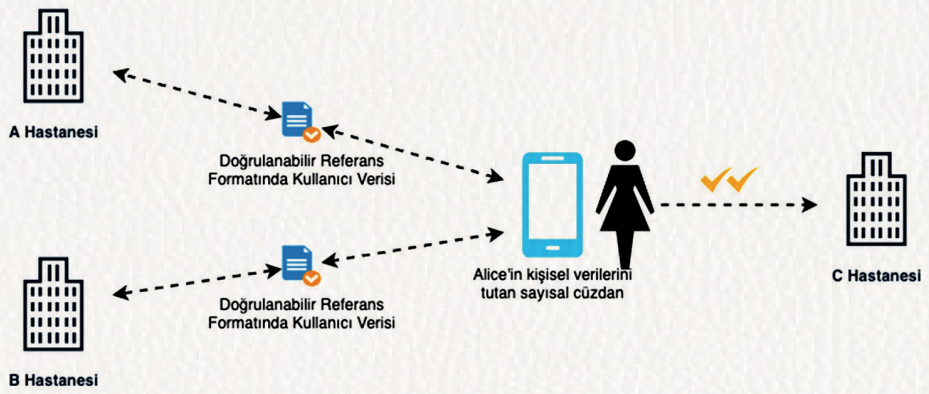
Blokzincir ağı için depolama birimleri, kimlik yönetimi, ölçeklenebilirlik, güvenlik ve gizlilik açısından ele alındığında hayati bir konudur. Olası güvenlik ve gizlilik problemlerinden uzak durulması için, blokzincir ağında özel bilgi tutulmamalıdır. İlerleyen teknolojiyle yüksek kapasiteli makineler ve kuantum bilgisayarı senaryoları dahilinde, şifrelenmiş ve özet fonksiyonundan geçirilmiş olsa dahi özel bilgi bu ağ üzerinde tutulmamaktadır [24]. Hassas

önemde olan gizli bilginin ağ üzerinde tutulması, gizli anahtarların çeşitli dolandırıcılık sonucu ele geçirilmesiyle, hassas bilgilerin ifşa edilmesi ve dolayısıyla riske edilmesi anlamına gelmektedir. Bu nedenle, sistemde veriler sadece blokzincir temel olarak dağıtık tanımlayıcıları ve kimlik sahiplerini arama amacıyla kullanılmaktadır. Ağ üzerinde, yalnızca bilgi paylaşımında kimlik sahibi ve iptal kayıtları arasındaki karşılıklı rızanın bir ispatı saklanır. Blokzincir üzerinde, bilginin kendisi yerine sadece varlığının bir ispatı tutulduğu için, ölçeklenebilirlik mücadele edilmesi gereken büyük bir sorun teşkil etmemektedir.

3.5. KULLANIM SENARYOLARI

Gerek blokzincir teknolojisinin karmaşıklığı ve gerekse de önerilen sistemin anlaşılabilirliği açısından, sistemin kullanım senaryoları üzerinden irdelenmesi yerinde olacaktır. Burada görselleştirilen kullanım senaryoları, geleneksel yöntemlerle halledilen işlemlerin, önerilen sistemle nasıl bir süreç takip edilerek çözüldüğü ve işlerin ne şekilde kolaylaştığının anlaşılması açısından oldukça faydalıdır.

Blokzincir temelli bir kimlik sisteminde, kuruluşların dijital olarak imzaladığı belgeler kimlik sahiplerine aittir. İmzalanan belgeler doğrulanabilir kimlik bilgileri biçimindedir. Doğrulayan taraflar, belgelerin orijinal olduğunu, değiştirilmediğini ve düzenleyenler tarafından dijital imzalar yardımıyla imzalandığını doğrulayabilirler. Örneğin; Şekil 3.9, bir hastanın birden çok tıbbi kurumu içeren tıbbi kayıtlarına ilişkin örnek bir kullanım durumunu göstermektedir. Kullanım durumunda, Alice adlı bir hasta daha önce A ve B hastanelerinde tedavi görmüş ve bu kurumlardan alınan önceki tıbbi kayıtlarını C hastanesinde sunmaktadır. Hastaneler A ve B, Alice'in tıbbi kayıtlarını doğrulanabilir kimlik bilgileri biçiminde ona verir. Alice bu belgeleri güvenli dijital cüzdanında toplar ve C hastanesindeki doktoruna sunar. Hastane A ve B'nin açık anahtarlarını bilen C hastanesi, Alice'in tıbbi kayıtlarının aslında hastane A ve hastane B tarafından verildiğini ve herhangi bir değişikliğe uğramadığını doğrulayabilir.

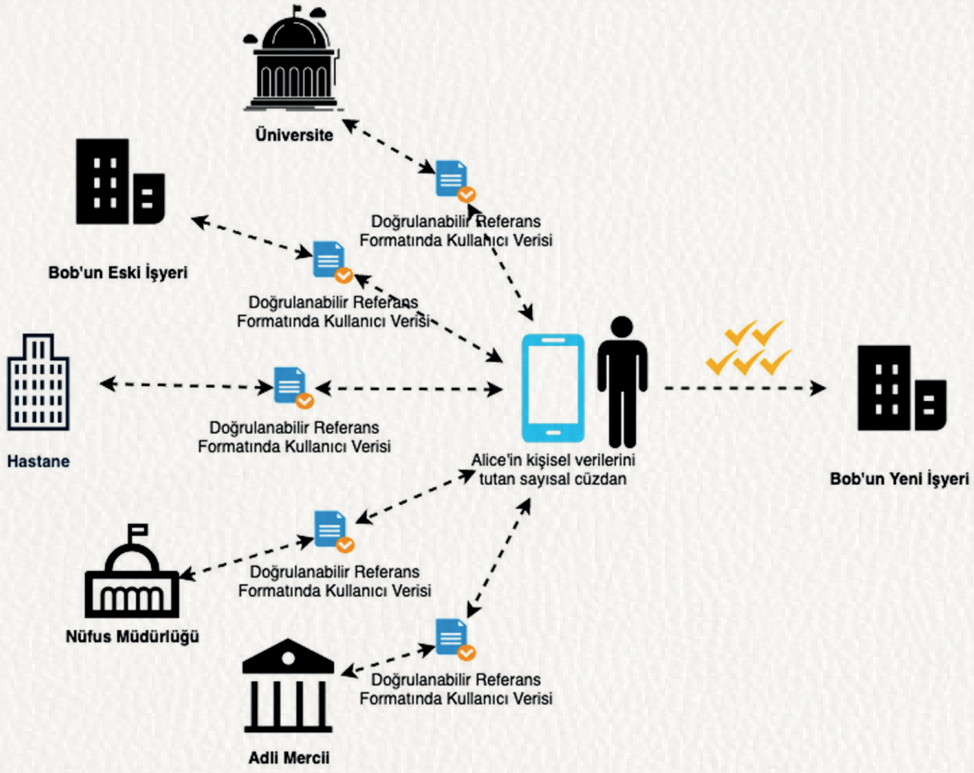


Şekil 3.9. Hastaların Tıbbi Kayıtları İçin Örnek Bir Kullanım Durumu.

Diğer bir kullanım örneği olarak işe giriş süreçlerini ele alalım. Şekil 3.10, yeni bir işe başlamak için belgeleri ve kimlik bilgilerini yeni bir işverene sunmak için örnek bir kullanım durumunu göstermektedir. Örnekte, Bob'un yeni şirketi, Bob'un eğitim derecesi kanıtı, eski istihdam kanıtı, hastaneden alınan sağlık raporu sonuçları, ikametgâh adres bilgileri ve sabıka kaydına ilişkin belgeler sunmasını talep ediyor. Bob, ilgili kurumlardan gelen belgeleri doğrulanabilir kimlik bilgileri biçiminde çevrimiçi olarak toplar ve bunları dijital cüzdanında saklar. Bob'un yeni şirketi, sunulan bu belgelerin gerçekliğini doğrulayabilir. Böylece, süreç zaman kazandırır, belgelerin çevrimiçi olarak düzenlenmesini ve doğrulanmasını sağlar ve kayıtların sahteciliğini önler.

Başka bir kullanım örneği olarak bankaya kredi başvurusu durumu verilebilir. Şekil 3.11, bir kredi başvurusu ile ilgili örnek bir kullanım durumunu göstermektedir. Kullanım durumunda, Alice, Banka B'ye kredi başvurusunda bulunur. Alice, Banka B'nin güvendiği Bank A'nın mevcut müşterisidir. Ayrıca, Alice bir araziye sahiptir ve ilgili bilgiler devlet daireleri tarafından tutulur. Ancak, Alice daha önce Banka B ile hiç çalışmamıştır. Bu nedenle Bank B'nin Alice ile ilgili herhangi bir kaydı bulunmamaktadır. KYC düzenlemelerinin bir parçası olarak, Banka B'nin ona hizmet edebilmesi için Alice'i tanıması gerekmektedir. Alice, önerilen çerçeveyi kullanarak kendisini Banka B ile doğrular. Daha sonra Banka B, Alice'in bilgilerini çevrimiçi olarak ilgili kuruluşlardan ister ve taleple ilgili bir bildirim alır. Alice, bilgilerinin doğru-

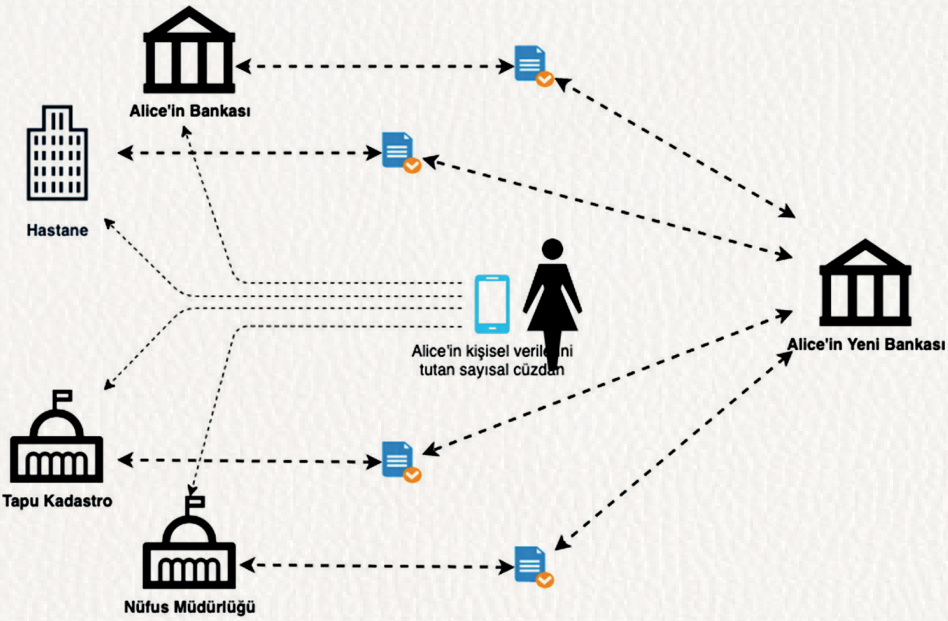
lanabilir kimlik bilgileri şeklinde Banka B ile paylaşılmasına izin verir. Banka B, toplanan bilgilere dayanarak Alice'in kredi başvurusunu işler. Tüm süreç, Alice fiziksel olarak bankaya gitmeden birkaç dakika içinde çevrimiçi olarak gerçekleşir.



Şekil 3.10. Yeni Bir İşverene Sunulması Gereken Belgeler İçin Örnek Bir Kullanım Durumu

Diğer önemli bir örnek kullanım senaryosu da, doğrudan dünyanın mevcut durumuyla ilgili COVID-19 salgınıyla mücadele çerçevesinde gösterilebilir. Küresel salgın (pandemi) ile seyahatleri kısıtlamak ve daha güvenli hale getirmek kritik önemde bir konu haline geldi. Virüsün uluslararası sınırlardan veya şehirlerarasında taşınmasını önlemek için, gerekli test sonuçlarının kişisel kimlik bilgileriyle gizli ve güvenilir bir şekilde eşleştirilmesi çok önemli-

dir. Blokzincir protokollerinin sunduğu değiştirme çabalarına dayanıklılık ve yüksek güvenlik yapısı bu konudaki çalışmaları desteklemektedir. Bu amaçla, Sirius uygulamasının geliştiricileri, COVID-19 test sonuçlarının doğrulanabilir kaynak yoluyla seyahat biletlerine veya dijital kimliklere bütünleştirilebileceğini gösterdi [25].



Şekil 3.11. Kredi Başvurusu İçin Örnek Bir Kullanım Örneği.

3.6. ÖZEL ANAHTAR ŞİFRELENMESİ

Veri gerçekliğini veya bütünlüğünü doğrulamak için, bir kullanıcının yalnızca dijital olarak imzalanmış veriye ve imzalayanın açık anahtarına ihtiyacı vardır. Bu bakış açısından, bir işlem sahibi sadece kendi özel anahtarını kullanarak blokzincirindeki bir işlemi imzalayabilir. Ardından, ağdaki diğer düğümler, yalnızca işlemi gönderenin genel anahtarını kullanarak işlemin sahibini ve bütünlüğünü doğrulayabilir. Bu nedenle, genel anahtarlar kullanıcılar arasında açıkça paylaşılabilir, fakat özel anahtarlar gizli ve güvenli tutulmalıdır [4].

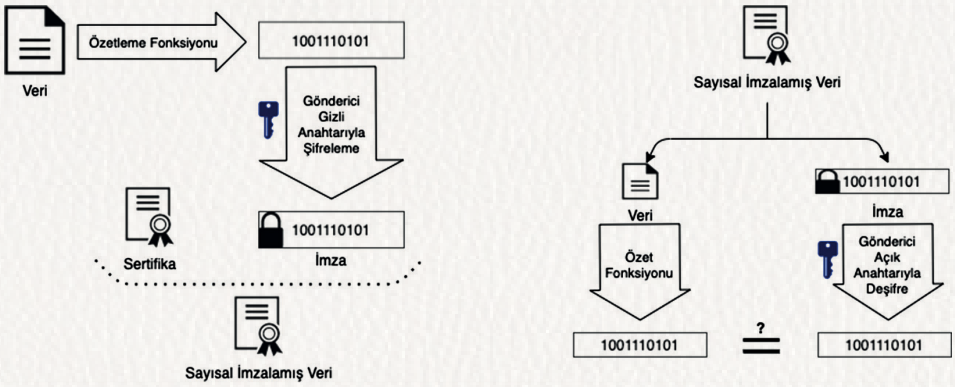
Genel olarak, kullanıcıların özel anahtarlarının güvenli saklanması için üç farklı yaklaşım mevcuttur [26]–[31]. İlk yaklaşımda, cihazda depolanan özel anahtarlara ulaşmak için ek bir güvenlik katmanı eklenir ve biyometrik kimlik doğrulama, özel bir anahtara ulaşmak için kullanılır. İkinci yaklaşımda, depolanan özel anahtarlar da biyometrik verilerle şifrelenir. Böylece özel anahtarları tutan cihazı şifrelemek yerine, özel anahtarların şifrelenmesi gerçekleştirilir. Son yaklaşımda, özel anahtarlar, biyometrik verilerin DES, RSA dahil olmak üzere bilinen önemli kriptografi algoritmalarına uygulanarak üretilir. Aydar ve arkadaşları, özel anahtar güvenli şifrelemesi için ikinci yaklaşım tercih edilmiş ve özel anahtarların kurtarılma işlemi için dağıtılmış bir anahtar kurtarma mekanizması kullanmıştır [23]. Kimlik doğrulama, popüler bir kimlik doğrulama tekniği olan parmak izi özelliklerine dayalı biyometrik veriler kullanılarak yapılmıştır.

3.6.1. Parmak İzi Kullanılarak Özel Anahtarların Şifrelenmesi ve Şifresinin Çözülmesi

Blokzincir protokollerinde, varlıklar, Dağıtık Defter Teknolojisi (DLT) ortamında dijital ve anonim olarak sahibini temsil eden genel anahtarlar ile ilişkilendirilmelidir. İşlemler ağda yayınlanmadan önce dijital olarak imzalanmalıdır. Şekil 3.12'de gösterildiği gibi, bir kullanıcının bir belgeyi dijital olarak imzalaması için özel bir anahtar yeterlidir.

Simetrik şifreleme ve şifre çözme, şifreleme ve şifre çözmenin aynı anahtar kullanılarak yapıldığı Veri Şifreleme Standardı (DES) gibi geleneksel simetrik şifreleme yöntemleri kullanılarak gerçekleştirilebilir [30]. Örnek çalışmada ise [23], özel anahtarları şifrelemek ve çözmek için simetrik şifreleme kullanmakla birlikte, özel anahtarın sahibinin parmak izi kullanarak simetrik şifrelemede kullanılan anahtarı otomatik olarak oluşturulmaktadır.

Biyometrik bir özellik olarak parmak izleri geleneksel olarak seçilen şifre kodlarına göre kullanılabilirlik avantajları sunar. Ancak, parmak izlerini kullanırken gizlilik, güvenlik ve uygulanabilirlik ile ilgili bazı endişelerin ele alınması gerekir.



Şekil 3.12. Dijital İmzalar, İmzalama ve Doğrulama

Parmak izi tanıma sistemlerinde iki ana aşama vardır: kayıt ve eşleştirme. Kayıt adımı; orijinal parmak izi görüntüsünün kaydedilmesini içerirken, eşleştirme adımı; aday parmak izi görüntüsünün kayıtlı görüntü ile eşleştirilmesini içerir [26]. Her iki aşamada da; parmak izi görüntüsü önceden işlenir, dönüştürülür ve özet alma (hashing) tekniği ile özetlenir. Farklı zamanlarda alınan iki parmak izinin aynı özet (hash) değerine sahip olması (eşleşseler bile) olasılıksal olarak zor olduğundan, verimli bir hata düzeltme mekanizması kullanılmalıdır.

Aşağıdaki adımlar da, özel anahtarların simetrik şifrelenmesin de kullanılmak üzere parmak izi verisinden elde edilecek sabit bir özet (hash) değerinin üretilmesi ve eşleştirilme mekanizması açıklanmıştır.

Ön İşleme

Ön işleme, görüntü zenginleştirme (filtreleme, ikilileştirme ve inceltme), parmak izi ayrıntı noktalarını çıkarma (minutiae), çekirdek noktası algılama ve çekirdek noktalara göre ayrıntıların hizalanmasını içerir. İyileştirme adımının amacı çizikleri ve kirlilikleri telafi ederek, parmak izi yapısını doğru bir şekilde algılamak için ikili bir parmak izi görüntüsü oluşturmaktır. Aydar ve arkadaşları [23], her pikselin tahmini sırt (ridge) frekansı ve sırt yönüne göre filtrelendiği Gabor Filtresi [32] yöntemini uygulamaktadır.

Geliştirme adımı, bir eşik değişkeni kullanılarak ikilileştirme ile devam eder ve sırt çizgilerinin genişliğini bir piksele sabitleyen inceltilir. Görüntü zenginleştirme işleminden sonra, ayrıntı algılama algoritması ile bir pikselin bir ayrıntıyı temsil edip etmediğini, kendisini çevreleyen 8-komşu pikselleri kontrol ederek tespit edilmektedir.

Eğer ilgili piksel bir sırt üzerindeyse ve 1 komşu sırt pikseline sahipse, sırt bitişi ayrıntısını temsil eder. Öte yandan, eğer piksel bir sırt üzerindeyse ve 3 komşu sırt pikseline sahipse, çatallanma tipi ayrıntıyı temsil eder.

Ayrıca, ayrıntı noktalarını kaynak olarak bu noktalara göre güvenilir bir şekilde hizalamak için çekirdek noktaların konumu ve yönelimini saptamak gerekir. Bir parmak izinin çekirdek noktaları (nokta indeksi), parmak izininin merkezlerini temsil eden özel piksellerdir. Çekirdek noktaları, döngü, delta ve sarmal çekirdek noktaları olmak üzere üç çeşittir. Bu çalışmada, Kawagoe ve arkadaşları [33] tarafından önerilen parmak izi çekirdek algılama yöntemini kullanılmaktadır. İlgili yöntem, görüntüyü alt bölgelere bölüp, yön modellerini elde edip ve kapalı bir eğri üzerinden çekirdek noktalarını hesaplar ve verilen bir (x, y) pikseli için, 8-komşuluğundaki bitişik yerel sırt yönlendirme açıları arasındaki farkı toplar. Hesaplamanın sonucuna göre:

- Sonuç sıfır ise (x, y) herhangi bir çekirdek noktası değildir,
- Sonuç 2PI ise (x, y) bir sarmal çekirdek noktasıdır,
- Sonuç PI ise (x, y) bir döngü çekirdek noktasıdır ve
- Sonuç -PI ise (x, y) bir delta çekirdek noktasıdır.

Ayrıntı hizalama adımında, her bir ayrıntı noktası, iki boyutta eksenlerin dönüşü kullanılarak döndürülür. Formül 3.1'de gösterildiği gibi bir ayrıntı noktası (x, y), aşağıdaki gibi matris çarpımı kullanılarak bir yönlendirme açısı θ ile bir çekirdek noktası (cx, cy)'ye göre saat yönünün tersi yönünde döndürülür:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{pmatrix} \begin{pmatrix} x - cx \\ y - cy \end{pmatrix} \quad (3.1)$$

Kartezyen Dönüşümü

Geri alınabilirlik, gizlilik ve güvenliği sağlamak için elzem olduğundan, biyometrik sistemin iptal edilebilir olması gerekir. Biyometrik imza olarak parmak izleri, sahibiyle kalıcı olarak ilişkilendirilir ve çalınması durumunda, daha önce ilgili parmak izi imzasıyla kullanılan tüm sistemler tehlikeye girer. Bu nedenle, kartezyen dönüştürme kullanarak parmak izi ayrıntı noktalarını tek yönlü dönüştürülmelidir. Orijinal parmak izi ayrıntı noktalarını sistemde saklamak yerine, dönüştürme parametreleri ile dönüştürülmüş ayrıntı noktalarını sistemde saklanmalıdır [23].

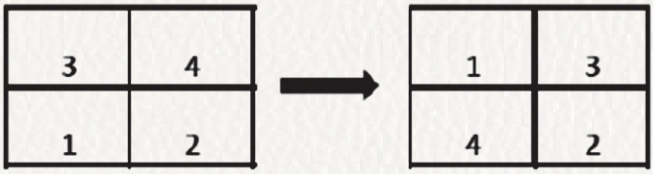
Kartezyen blok dönüşümünde, ayrıntı noktalarının temsil edildiği 2D koordinat sistemi eşit boyutlu bloklara bölünür. Başlangıçta, ayrıntı noktaları bloklara konumlarına göre yerleştirilir, böylece birbirlerine yakın olan ayrıntı noktaları aynı veya komşu bloklara yerleştirilir. Daha sonra, matris çarpımı kullanılarak bloklar karıştırılır ve ayrıntı noktaları yeni bloklarına göre konumlandırılır.

Uygulamada, 2D koordinat sistemi H x W boyutunda bloklara bölünmüştür [23]. İlk kartezyen bloklar 1 ile |H x W| arasında numaralandırılır. Bu matris boyutu 1 x |HxW| olan C matrisi olarak adlandırılır. Boyutları |HxW| x |HxW| olan ve içerik değerleri rastgele seçilmiş 0 ya da 1 olan bir M dönüşüm matrisi oluşturulur.

Örnek olarak; H = 2 ve W = 2 olduğunda, C = [1, 2, 3, 4] olur ve rastgele oluşturulmuş olan bir M Matrisi ile matris çarpımı Formül 3.2'deki gibi gösterilir:

$$C' = \begin{pmatrix}1 & 2 & 3 & 4\end{pmatrix} \begin{pmatrix}0 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 1 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{pmatrix} = \begin{pmatrix}3 & 2 & 4 & 3\end{pmatrix} \qquad (3.2)$$

İlgili Matris çarpımı sonucu, daha önce kartezyen blok 1 içine yerleştirilen ayrıntı noktalarının sonrasında blok 3 ile eşleştirildiği ve 2 içerisinde bulunan ayrıntı noktalarının blok 2 ile eşlendiği anlamına gelir. Benzer şekilde blok 3'teki ayrıntı noktaları blok 4 ile ve blok 4'tekilerinde blok 3 ile eşlendiğini gösterir. İlgili eşleşme sonuçları Şekil 3.13'de ifade edilmiştir:



Şekil 3.13. Kartezyen Dönüşüm Sonuçları.

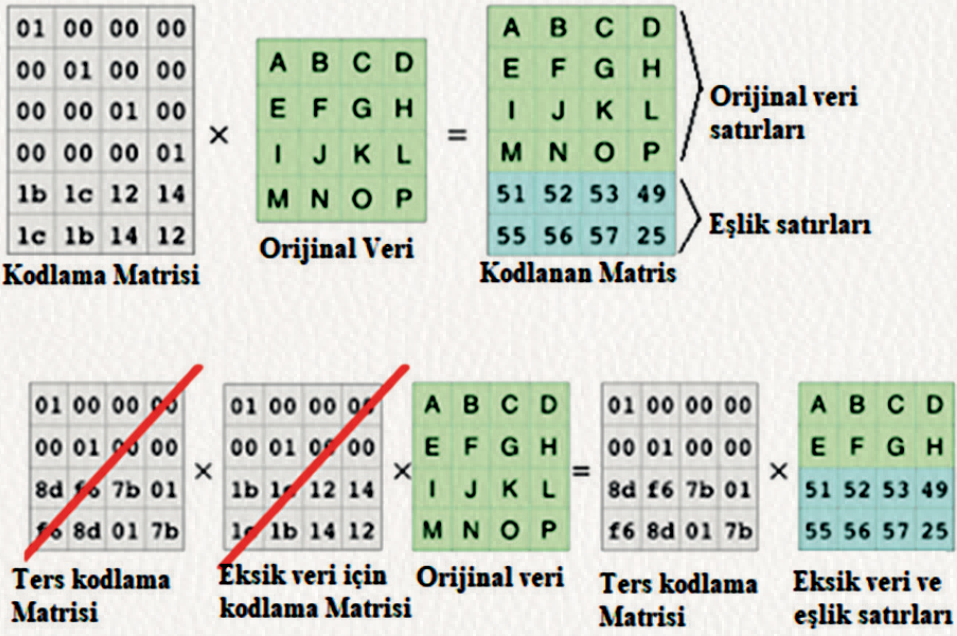
Birden fazla kartezyen bloğun, dönüştürülmüş uzayda aynı kartezyen bloğa eşlenmesi de mümkündür. Kartezyen bloklar, 2D koordinat sistemindeki konumlarına göre numaralandırılır. Kayıt aşamasında, ayrıntı noktalarının orijinal konumlarını kaydetmek yerine, dönüştürülen konumları dönüştürme parametreleri ile birlikte kaydedilir. Dönüştürme parametreleri, orijinal parmak izi görüntüsünün sınırlarını ve dönüştürme matrisini içerir. Belirli bir ayrıntı noktası için orijinal kartezyen bloğunun kayıt aşamasında kaydedilmediğine dikkat etmek önemlidir. Bununla birlikte, eşleştirme sırasında, aday parmak izi şablonu için, ayrıntı noktalarının orijinal kartezyen blokları, Reed-Solomon hata düzeltme kodlamasının kurtarma işleminde kullanılmak üzere tutulur.

Reed-Solomon Hata Düzeltme

Reed-Solomon [34] bir hata düzeltme (düzeltme kodlaması) mekanizmasıdır. Bu mekanizma, belirli bir girdi için, bazı parçalar eksik olsa bile orijinal girdiyi yeniden üretebilecek şekilde eşlik verileri (parity veri) üretir. Linux RAID ve Facebook'un soğuk deposu gibi birçok modern depolama sistemi Reed-Solomon kullanmaktadır. Reed-Solomon, verilen bir mesajı n eşit parçaya böler ve bir girdi matrisi oluşturur; burada n, matrisin yüksekliğidir. Daha sonra, n+k boyutunda bir kodlama matrisi oluşturur, burada k, eşlik satırlarının sayısıdır. Kodlama matrisinin ilk n satırında köşegende 1 değeri ve diğer matris hücreleri için 0 değeri mevcuttur. Kodlanan veriler, kodlama matrisinin orijinal matris ile çarpılmasıyla oluşturulur.

Kodlama matrisindeki köşegen 1'ler nedeniyle, kodlanan verinin ilk n satırı orijinal mesajla aynıdır ve son k satırı eşlik satırlarıdır. Böylece, kodlama matrisinin her bir satırı orijinal matristeki ilgili satıra karşılık gelir. Bu nedenle, orijinal mesajdaki bazı satırlar eksik olduğunda, kodlama matrisindeki ve

kodlanmış matristeki karşılık gelen satırlar kaldırılır ve sol taraftaki orijinal verilerle matris çarpım denklemi hala geçerli kalır. Daha sonra, yeni kodlama matrisinin ters matrisi oluşturulur ve yeni denklemin her iki tarafı ile çarpılır. Sonunda, orijinal veri matrisi denklemin sol tarafında üretilmiş olur. Şekil 3.14'de, değeri "ABCDEFGHIJKLMNOP" olan bir giriş verisinin Reed-Solomon kodlamasının bir örneğini ve bu mesajda "IJKLMNOP" eksik olduğunda Reed-Solomon hata düzeltme mekanizması ile kurtarılmasını gösterilmiştir.



Şekil 3.14. Reed-Solomon Hata Düzeltme Uygulaması

Örnek uygulamada [23], ayrıntı noktalarının özetleri (hash değerleri) girdi verileridir. Kartezyen blokları dönüştürülmeden önce, her bir blok için, ayrıntı noktalarının özet (hash) değerlerini girdi olarak kullanarak Reed-Solomon hata düzeltme mekanizması çalıştırılmaktadır. Ayrıntı noktalarının özet değerlerini kullanmak, orijinal ayrıntı noktalarının ortaya çıkmasını engelleyerek parmak izi sahibinin mahremiyetini korumaktadır. Ayrıca, her kartezyen bloğunu temsil eden toplam özet değerlerini girdi olarak kulla-

narak, tüm dikdörtgen için ayrı bir Reed-Solomon algoritması çalıştırılır. Bu şekilde, ayrıntı noktalarının eksik özetleri her bir kartezyen bloğu için kurtarılabilir.

Sonuç olarak, eşleştirme sürecinde kullanılan parmak izi sisteminin genel bir özet değerini hesaplayabilir. Genel özet değeri, sahibin özel anahtarının simetrik şifrelemesinde kullanılacak olan simetrik anahtardır. Burada nihai amaç, eşleştirme aşamasında aynı parmak izi sahibi için her zaman aynı simetrik anahtarı oluşturmaktır.

Parmak İzi Eşleştirme

Eşleştirme aşamasında, belirli bir aday parmak izi görüntüsünün orijinal parmak izi görüntüsü ile aynı genel özet değeri üretip üretmediğini belirlenir. Aday parmak izi görüntüsü, orijinal parmak izi görüntüsü ile aynı ön işleme ve dönüştürme adımlarından geçer. Kartezyen dönüşümünde, orijinal parmak izi görüntüsünün de olduğu gibi aynı dönüşüm parametreleri (sınırlar ve dönüştürme matrisi) kullanılır. Ayrıca, kayıt aşamasının aksine, orijinal kartezyen blok numaraları aday parmak izi görüntüsü için saklanır.

Eşleştirme algoritması, aday parmak izi şablonunun dönüştürülmüş ayrıntı noktalarını, orijinal parmak izi şablonunun dönüştürülmüş ayrıntı noktaları ile karşılaştırır. Bu karşılaştırma, kartezyen blokların her biri için ayrı ayrı yapılır. Geometrik olarak daha yakın ayrıntı noktaları, küçük bir hata payı ile hem orijinal hem de aday parmak izi şablonunda aynı kartezyen bloğa dönüştürülecektir. Bu nedenle, dönüştürülmüş aday parmak izi şablonundaki kartezyen blok numarası x içindeki ayrıntı noktaları, dönüştürülmüş orijinal parmak izi şablonundaki kartezyen blok numarası x içindeki ayrıntı noktaları ile karşılaştırılır. Karşılaştırma, ayrıntı nokta türlerinin eşitlik kontrolü ve makul bir eşikle Öklid mesafesi kullanılarak yapılır. Bir eşleşme bulunursa, aday parmak izi şablonuna ait olan ayrıntı noktasının dönüştürülmeden önceki kartezyen blok numarası, orijinal parmak izi şablonuna ait olan ayrıntı noktasının dönüşümünü tersine çevirmek için kullanılır. Bu şekilde, eşleşen ayrıntı noktaları için orijinal ayrıntı koordinatları kurtarılır [23].

Kartezyen blokların her biri için geri dönüştürülen ayrıntı noktaları, Reed-Solomon hata düzeltme işlemine girer ve sonuçta toplam özet değeri

oluşturulur. Oluşturulan özet değeri, kayıt aşamasında oluşturulan özet değeri ile aynıysa, parmak izi görüntüleri eşleşir. Bu özet değeri ve şifrelemede kullanılan simetrik algoritma kullanılarak, şifrelenmiş özel anahtarın şifresi çözülür. Metodolojimizde açıklanan uygulama kodu, araştırma amacıyla kullanılabilir[1].

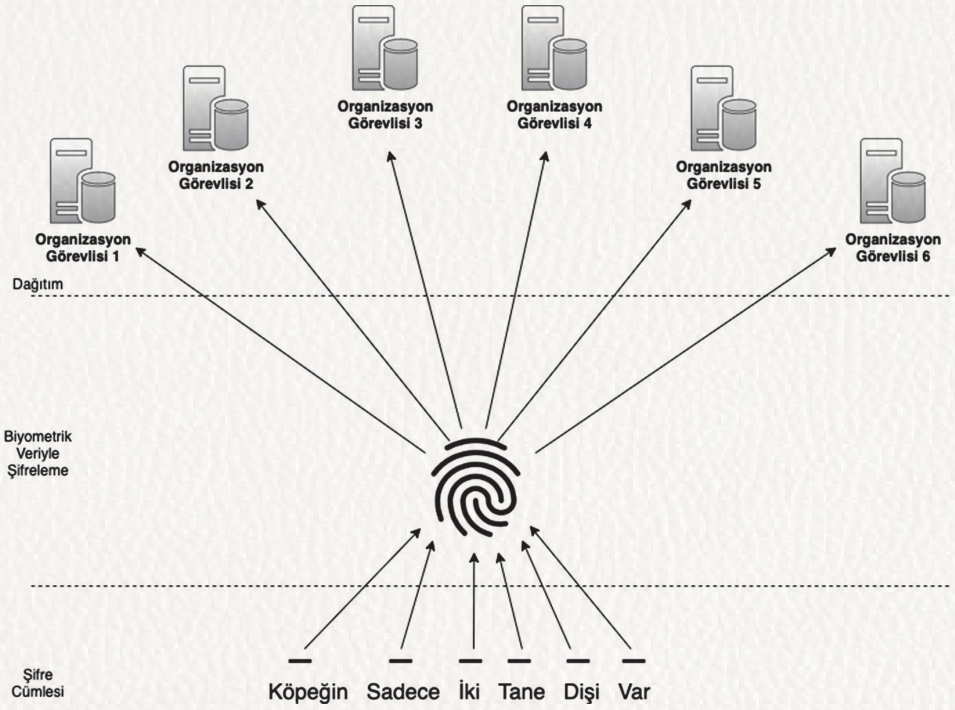
3.7. ANAHTAR KURTARMA

Blokzincir protokollerindeki kriptografi, büyük ölçüde açık ve özel anahtar çiftlerinin kullanımına dayanır. Açık anahtar herkese açık olduğundan, dağıtılmış defter teknolojisindeki anahtar kurtarma temel olarak özel anahtar kurtarmayı ifade eder. Özel anahtar hırsızlığı ve kayıpları, blokzinciri sistemlerinde büyük güvenlik sorunlarıdır. Başka bir deyişle, özel anahtarın kaybedilmesi, özel anahtarla ilişkili tüm varlıkların sahipliğini kaybetmeye yol açar. Bu nedenle, özel anahtarların kurtarılması, blokzinciri tabanlı sistemlerde son derece önemlidir. Blokzincirinde verimli, güvenli ve ölçeklenebilir bir anahtar kurtarma sağlamak için, Shamir'in Gizli Paylaşım (Shamir's Secret Sharing) şemasına dayalı olarak şifrelenmiş özel anahtarlar için dağıtılmış bir anahtar kurtarma mekanizması kullanılabilir [35].

Bu yaklaşımda uygulanan dağıtık anahtar kurtarma mekanizması, Şekil 3.15'de olduğu gibi bir örnek kullanılarak gösterilebilir [23]. Şekilde, bir parola, biyometri kullanılarak şifrelenmiştir. Şifrelenmiş metin 6 parçaya bölünerek görevliler (Stewards) adı verilen 6 partiye dağıtılır. Bu nedenle, orijinal parolayı kurtarmak için, parçalar görevlilerden toplanmalı ve şifre sahibinin biyometrik özellikleri kullanılarak şifresi çözülmelidir.

Bu çözümde, biyometri ekstra bir güvenlik katmanı sağlar. Ancak, orijinal parolayı almak için tüm görevlilerin hazır olmasını gerektirir. Bu sorunu çözmek için, kurtarma için daha fazla esneklik ekleyen Shamir'in Gizli Paylaşım (SGP) şemasını kullanılmaktadır.

1 http://bit.ly/cancellable-fingerprint-encryption -



Şekil 3.15. Dağıtık Anahtar Kurtarma Örneği

3.7.1. Shamir'in Gizli Paylaşım Şemasını Kullanarak Kurtarma

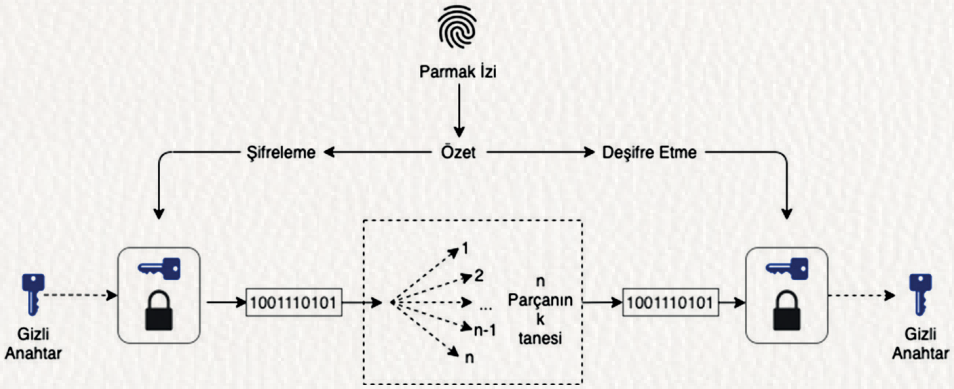
SGP şemasına göre, D verisi n parçasına bölünse bile D'nin k adet parçası D'yi yeniden oluşturabilir, ancak k-1 adet parçaları bile D hakkında hiçbir bilgi vermez [35]. Bu yöntemi kullanarak anahtar sahipleri şifrelerini n parçaya bölerek n farklı yere dağıtabilirler. Bazı parçalar kaybolsa bile, herhangi bir k parça sırrı kurtarmak için yeterli olacaktır. SGP şeması, özel bir anahtarı kurtarmak için uygulanabilir. Dağıtılmış dijital kimlik sistemlerinde, ağda güvenilen düğümler olan ve Steward olarak adlandırılan görevli aktörler vardır. Görevlilerin hizmetleri, dağıtık parçaların yerleri olarak kullanılabilir.

Bu yaklaşımda biyometrik bilgiler, özel bir anahtarı şifreleyerek korumak için kullanılır. Önce sahibin parmak izi verilerini kullanarak simetrik bir anahtar

oluşturulur. Sonrasında özel anahtarı, anahtar, şifreleme bölümünde ayrıntıları verildiği gibi simetrik anahtarla şifrelenir.

Şifreleme işleminden sonra, şifrelenmiş veri n parçalara bölünür ve parçaların her biri, görevli hizmetleri lokasyonları gibi farklı ve güvenli konumlara dağıtılır. Kurtarma aşamasında, özel anahtarı kurtarmak için n şifrelenmiş parçaların k kısımları yeterlidir. Şifrelenmiş özel anahtar kurtarıldıktan sonra, aynı simetrik anahtar parmak izi kullanılarak yeniden oluşturulur ve şifrelenmiş özel anahtarın şifresini çözmek için kullanılır. Şekil 3.16'da, önerilen özel anahtar kurtarma mekanizmasının genel görünümü ayrıntılı olarak gösterilmektedir.

Anahtar kurtarma iki ana ilkeye dayanmaktadır: biyometri ile şifreleme ve dağıtık mimaride veri tutulması. Şifrelenmiş her biyometrik veri ve özel anahtar çifti, güvenli depolama için bir dizi görevliye dağıtıldığından, blokzinciri ağındaki bir görevli düğüme bir saldırı ile izinsiz girildiğinde bile, düğümde depolanan veriler özel anahtarın şifresini çözmek için yeterli olmayacaktır. Çünkü saldırganlar, özet anahtarının tamamını tamamlamak için diğer görevlilerin yerlerini bilemezler. Ayrıca, birleşik özet değerini içeren tüm görevli düğümlerine erişilmesi durumunda bile özel anahtarın şifresi, kullanıcının yerel cihazında tutulan kullanıcının biyometrik bilgileri olmadan çözülemez. Diğer bir deyişle, iki ayrı güvenlik faktörü vardır.



Şekil 3.16. Anahtar Kurtarma Mekanizması

3.8. TÜRKİYE CUMHURİYETİ KİMLİK KARTI VE BLOKZİNCİR ENTEGRASYONU

Teknolojinin gelişmesi sadece özel şirketlerin ürettiği ürünler üzerinde değil aynı zamanda kamu kurumlarının işleyişi üzerinde de önemli değişiklikler getirilmesini zorunlu kılıyor. Teknoloji ile kişilerin isteklerine kolayca ulaşmaları, kamu kurumları ile olan işlerindeki zaman kayıplarının daha çok fark edilmesine sebep olmaktadır. Kişilerin günlük işlerinin büyük çoğunluğunun hala kamu kurumları tarafından yürütüldüğünü düşünürsek, Türkiye'deki vatandaşların işlerini kolay halledebilmeleri için yapılacak işlerin dijital ortama taşınması mutlak zorunluluk olmuştur.

Şu ana kadar blokzincir yaklaşımının getirdiği avantajlar ve temelde böyle bir yapıda olması gereken ilkelerden bahsedildi. Bu ilkeler ve avantajlar düşünüldüğünde, kişilerin kimlik bilgileri gibi güvenlik gereksinimi fazla olan bilgilerin böyle bir yapı ile desteklenmesi oldukça olası ve uygulanabilir gözükmektedir. Bu bölümde, Türkiye Cumhuriyeti Kimlik Kartı (TCKK) sisteminin blokzincir yapısına entegrasyonunun getirdiği avantajlarla birlikte karşılaşılabilecek sorunlar ele alınmıştır. Kimlik yönetim sistemlerinde kişilerin güvenli olarak sisteme ilk defa kendilerini tanıtması çok önemli ve gerekli bir iş olup, bu kimlik üzerinden veri paylaşımı olacağı için mutlak güvenilir olması gerekmektedir.

Her ne kadar farklı yaklaşımlar mevcut olsa da teknolojinin ve güvenliğin bu aşamasında ülkeler bazında tamamen merkezi olmayan (merkeziyetsizleştirme) bir yapı önermek ve bu önerilen yapının uygulanabilirliği çok gerçekçi durmuyor. Böyle bir uygulamanın ilk başta devlet tarafından desteklenerek başlaması ideal bir yaklaşım olmakla birlikte, hükümetlerin tutucu ve temkinli yaklaşımlarından dolayı sistemin uygulanabilirlik potansiyeli düşmektedir. Ancak bu çalışmada önerilen blokzincir tabanlı kimlik sisteminin devlet tarafından vatandaşlara atanan kimlik sistemleriyle entegrasyonu sistemi oldukça kullanılabilir hale getirmektedir.

Bu sebepten, ilk etapta devletin blokzincir ağına dahil olması yerine, zaten var olan akıllı kimlik kartlarındaki biyometrik özelliklerden faydalanılarak sistemde kişinin kendisini ispat etmesi, öteki kimlik sahipleri ve sisteme dahil olan kurumlarla olan ilişkilerinde güven mekanizmasının erken tesis edilmesine olanak sağlaması sebebiyle daha avantajlı ve uygulanabilir bir yöntem-

dir. Bu amaçla, Türkiye Cumhuriyeti'nde kullanılan yeni nesil akıllı kimlik kartları, sistemimizde doğrulanabilir kaynak olarak tanımlanabilir. TCKK sahipliğinin ispatı TCKK içerisindeki biyometrik veri ile kişinin mevcut biyometrik verisinin eşleşmesi yöntemi ile doğrulamak mümkündür. Bu yaklaşım ile kullanıcıların kendilerini tanıtan kaynaklarının merkezi bir yapı ile desteklenip, daha sonrasında bu bilgi üzerinden diğer bilgilerinin akışının ademi merkeziyetçi bir yapı ile sürdürülmesi sağlanır. Kişilerin kimlik bilgilerini içeren doğrulanabilir kaynak oluştururken kişinin kendini ispat etmesi için TCKK Elektronik Kimlik Denetim Sistemi (EKDS) ile entegrasyonu sağlanabilir.

3.8.1. EKDS'nin Blokzincir Tabanlı Kimlik Yönetim Sistemine Entegrasyonu

Önerilen yapıda, devletlerin ve kişilerin birbirlerine güvenip bilgi paylaşımı yapmalarını sağlamak için en azından başlangıçta merkeziyetçi bir yapının desteği şarttır. Bu destek kimlik bilgisinin onayından sorumlu devlet kurumunun blokzincir yapısında aktif bir paydaş olarak var olmasını gerektirmektedir. Fakat böyle bir kurumun özellikle tutucu ve korumacı ülkelerde sisteme dahil ettirilmesi oldukça zordur. Bunun yerine T.C. kimlik kartlarındaki biyometrik verilerden faydalanarak, devlet kurumunun iş yükünü artırmadan sisteme dahil edilmesi hedeflenmektedir.

Öncelikle, kişiler nasıl günlük hayatta kimlik kartlarını göstererek varlıklarını ispat ediyorlarsa, aynı şekilde kişilerin kimlik kartlarını kart okuyucu yardımı ile okutarak dijital ortamda varlıklarını ispat edebilirler. T.C. Kimlik kartlarının (TCKK) içerisindeki bilgiler ISO/IEC 7816 standardına uygun herhangi bir kart okuyucu ile okunabilmektedir [36]. Bu bilgilere ulaşabilmek için birden farklı kimlik doğrulama yöntemi önerilmektedir. Bunlar kısaca, sadece kimlik kartı; kimlik kartı ile sahibi tarafından belirlenmiş bir şifre; ya da kimlik kartı, şifre ve sahibine ait herhangi bir parmak izi gibi biyometrik veridir. Birden farklı kimlik doğrulama yöntemleri bulunmasına rağmen güvenlik çözümünün en yüksek olduğu düşünülen biyometrik veri ile kart sahibini doğrulama yöntemi en uygun yaklaşımdır.

TCKK kimlik kartları içerisinde özel kriptografik anahtarlar ve bu anahtarlarla ilişkili olan sayısal sertifikalar mevcuttur. Biyometrik veriye ulaşabilmek

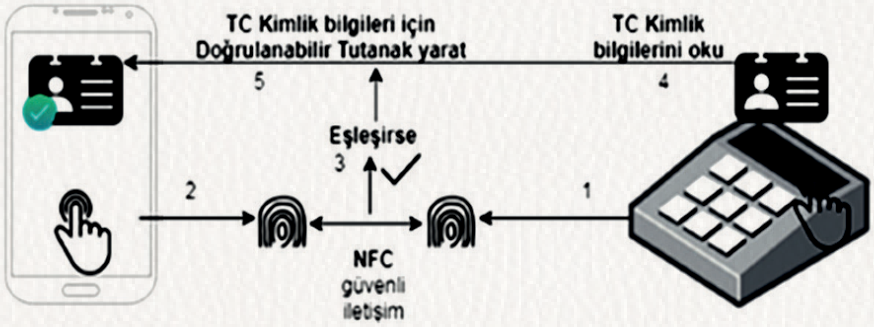
için KEC ile KEC cihazının simetrik anahtarının olduğu GEM (Güvenli Erişim Modülü) kartı gerekir. T.C. kimlik kartı proje kapsamında milli olarak gerçekleştirilen açık anahtar altyapısı (Public Key Infrastructure - PKI 2) kullanılmıştır [37]. Fakat bu durum sertifika dağıtımı konusunda yetkili kişilerle güven esasına dayanmaktadır.

Biyometrik sensör tarafından kart kullanıcısına ait girilen gerçek zamanlı biyometrik verinin, TCKK içerisinde güvenli bir şekilde saklanmış biyometrik veri ile karşılaştırılarak doğrulanabilir. Fakat TCKK içerisindeki biyometrik verinin okunması, güvenlik ve kişisel mahremiyeti koruma gerekçeleri nedeniyle, standart bir kart okuyucu ile mümkün olmamaktadır. Bu amaç için özel üretilmiş Kart Erişim Cihazlarının (KEC) kullanılması gerekmektedir. TCKK sisteminde biyometrik veriyi kapsamayan diğer doğrulama yöntemleri için herhangi bir standart kart okuyucu kullanılabilir. TÜBİTAK tarafından tasarlanan KEC kart okuyucuları için Kiosk KEC cihazları kamuya açık ortamlarda vatandaşın kullanımına sunulmuştur. Kiosk KEC'lerde yapılabilecek işlemler aşağıda listelenmiştir [38].

- PIN doğrulama
- En üst seviye kimlik doğrulama (kart test)
- Mevcut PIN ile yeni PIN tanımlama
- Biyometrik doğrulama ile yeni PIN tanımlama
- PUK girilerek PIN blokesi kaldırma
- Biyometrik doğrulama ile PIN blokesi kaldırma
- Kimlik bilgilerini ekranda gösterme (PIN doğrulaması ile)
- Kişisel mesaj değiştirme
- Temassız arayüz bilgilerini ekranda gösterme

Önerilen çalışmada, Yakın Saha İletişimi (Near Field Communication (NFC)) teknolojisi yardımıyla, kişinin mobil cihazından alınan gerçek zamanlı biyometrik verisi, kart okuyucudan alınan TCKK kartında bulunan kişinin biyometrik verisi karşılaştırılır. Bilgiler eşleştiği takdirde, kimlik sahibinin kendi kimliğini devlet tarafından kendisine atanan TCKK yardımı ile doğruladığını gösteren özel bir doğrulanabilir kaynak yaratılır ve ilgili mobil cihazda kay-

dedilir. Şekil 3.17'de bu senaryonun işleyiş mekanizması detaylandırılmıştır. Bahsedilen senaryonun ilgili mobil cihaz için bir sefer gerçekleşmesi ve tekrar eden kimlik doğrulama işlemlerinin yaratılan ilgili doğrulanabilir kaynaktan faydalanılması öngörülmüştür.



Şekil 3.17. Blokzincir Tabanlı Kimlik Yönetim Sisteminin EKDS Sistemine Entegrasyonu

Böyle bir blokzincir temelli kimlik sisteminde, kişiler kimlik kartlarını okuttuktan sonra oluşan doğrulanabilir kaynakın ispatı blokzincirde kişisel bilgilerin kendisi de mobil telefonlarda ya da kişisel bilgisayarlarda oluşturulan cüzdanlarda doğrulanabilir kaynak olarak tutulur. Bu özel bilgilerin özellikle güvenlik yönünden aşırı zaafları olan mobil telefonlarda tutulması farklı güvenlik çözümlerini ele almayı gerektirmektedir.

3.8.2. Blokzincir Tabanlı Kimlik Sisteminde Doğrulanabilir Şemalar

Blokzincirde tutulan veriler değiştirilemez olduğundan ve veriler ağdaki paydaşlara açık olduğundan blokzincirde tutulan veriler önem arz etmektedir. Önerilen kimlik sisteminde, kimlik bilgileri ve kimliğe dayalı veriler yine kimlik sahibine ait olan kişisel cihazlarda tutulur. Blokzincirden ise, dağıtık tanımlayıcıların tanımlanmasında, doğrulanabilir kaynakların doğrulanmasında, doğrulanabilir kaynakların taslak yapılarının (doğrulanabilir şemalar) kaydedilmesinde ve sağlanan doğrulanabilir kaynakların iptal kayıtlarının tutulması sebebiyle yararlanılır.

Doğrulanabilir şemalar (credential schema) için, kimlik bilgileri ile ilgili şema bilgisi mevcuttur. Fakat kimliğe dayalı bilgi paylaşımı arttıkça diploma, sağlık raporları gibi her bilgi ile alakalı yeni şemalar yaratılacaktır. Doğrulanabilir şemalar için "Verifiable Credentials Implementation Guidelines 1.0" dokümanında bulunan standartlar temel alınarak oluşturulmuştur [39]. Aşağıda kimlik bilgisi için doğrulanabilir şema örneği verilmiştir.

```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "id": "http://example.edu/credentials/3732",
  "type": ["VerifiableCredential", "IdentityCredential"],
  "issuer": "https://example.edu/issuers/14",
  "issuanceDate": "2010-01-01T19:23:24Z",
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "name":"Cenk"
    "surname":"Aydın"
    "parentnames": {
      "type": "Mother",
      "name": "Ayse"
      "type": "Father",
      "name": "Cemil"
    }
    "seriId":"124-543"
  },
  "credentialSchema": {
    "id": "https://example.org/examples/degree.json",
    "type": "JsonSchemaValidator2018"
  },
  "proof": { ... }
}
```

Özetle, TCKK'da bulunan pasaport bilgileri cep telefonları ya da harici olarak bulunan NFC yardımı ile okunup son kullanıcı uygulama katmanı olarak web tabanlı cüzdana doğrulanabilir kaynak olarak girişi yapılmaktadır. Bu doğrulanabilir kaynak blokzincirde tutulan ve yukarıda da bir örneği verilen doğrulanabilir şeması kullanılarak yaratılmaktadır.

3.8.3. KVKK ve Türk Ceza Kanunu ile Uyumluluk

Önerilen sistem yardımıyla kişilerin bilgilerini kendi cihazlarında tutup, bu cihazlardaki bilgileri kullanarak işlem yaptığında, blokzincir yapısı ile kendisini ispatlayan bir sistem meydana gelmektedir. Uygulama kapsamında, kişilerin hayatına oldukça kolaylık sağlamasına rağmen, hukuki yönden bazı potansiyel sorunlar ile karşılaşılmaktadır. Hukuki açıdan blokzincir temelli kimlik yönetimi yaklaşımı, mevcut kurallarla uyumluluk göstermektedir.

Türkiye Bilişim Vakfı tarafından Şubat 2019'da yayınlanan "Dünyada blokzincir regülasyonları ve uygulama örnekleri" raporu ile var olan örnekler hukuksal olarak ele alınmıştır. Bu çalışmada asıl önerilen kimlik uygulaması olduğu için "veri" kavramı ve bununla alakalı hukuksal düzenlemeler dikkate alınmıştır. Yapılan işlemler için blokzincirde farklı dijital kimlikler yaratıldığı için bu kapsamda Avrupa Birliği'nin 2014 yılında imzalanmış olduğu Elektronik Kimlik Belirleme ve Güven Hizmetleri Düzenlemesi (eIDAS) karşımıza çıkmaktadır [40]. Bu yönetmelik ile kimlik doğrulama işlemleri düzenlenmektedir. Dijital kimlik ile gelen önemli diğer olgu ise dijital imzalardır. Halen dijital imzalar mevcut devlet kurumlarında kişiler tarafından oldukça aktif olarak kullanılmaktadır. Bu sebepten dolayı, böyle bir olgunun uyumlanmasında problem yaşanılacağı beklenmemektedir.

Kimlik yönetimi sisteminde kişisel veriler sistemde tutulduğundan 2016 yılında kanun haline gelen kişisel verileri koruma kanununun bu sisteme entegre edilmesi gerekmektedir. Kişisel veriler kimlik sahibine ait halka açık olmayan bütün veriler olarak tanımlanabilir. Türk Ceza Kanunu'nda (TCK), kişisel verilerin korunması ve ilgili cezai hükümler madde 135 ve 140 arasında açıkça belirtilmiştir. TCK'nın madde 135 hükmü ve birinci fıkrasına göre, hukuka aykırı olarak kişisel verileri kaydeden kimseye bir yıldan üç yıla kadar hapis cezası verilir. Aynı maddenin ikinci fıkrasında ise, hukuka aykırı olarak kaydedilen kişisel verinin, kişilerin siyasi, felsefi veya dini görüşlerine, ırki kökenlerine, hukuka aykırı olarak ahlaki eğilimlerine, cinsel yaşamlarına, sağlık durumlarına veya sendikal bağlantılarına ilişkin olması durumunda öngörülen ceza yarı oranında artırılır. Ek olarak, madde 136 kişisel verilerin yayılması, madde 138 ise hukuki sürenin geçmiş olmasına rağmen kişisel verilerin yok edilememesi ile ilgili cezai sorumlulukları düzenlemektedir [41].

Aynı zamanda, 2016 yılında 6698 sayılı Kisisel Verileri koruma kanunu (KVKK) resmi gazetede yayımlanarak yürürlüğe girmiştir. KVKK'nın Avrupada yürürlüğe giren daha kapsamlı yasal düzenleme ise General Data Protection Regulation (GDPR)'dır [42]. GDPR'ın 6 ana ilkesi mevcuttur [43]:

- Kanuna uygunluk, adalet ve şeffaflık
- Amacın açıklığı
- Sınırlandırılmış veri paylaşımı
- Doğruluk
- Veriyi tutma zamanının kısıtlanması
- Bütünlük ve mahremiyet

Kimlik sahipleri, ilgili kurumlardan verilerinin işlenip işlenmediği öğrenme, eğer işlendiyse ilgili bilgileri talep etme, amacının ne olduğunu ve bu amaca uygun kullanılıp kullanılmadığını öğrenme hakkına sahiptir. Ayrıca, kişiler, kişisel verilerin üçüncü kişilere aktarılıp aktarılmadığını, verilerin doğru işlenip işlenmediği, eğer yanlış işlenmiş ise silinmesini veya yok edilmesini talep etme haklarına da sahiptirler.

Bu çalışma kapsamında önerilen blokzincir yapısında, kimlik bilgisi tutulan her kişinin blokzincirde var olan ispat, şemalar ve iptal verilerini okumaya hakkı olup, bu verileri sadece belirli güvenilir paydaşların blokzincire kaydetme yetkisi vardır. Eğer önerilen blokzincir yapısı yukarıda bahsedilen 6 temel ilke uyumluluğu sağlanırsa, KVKK ve TCK kanunları ile uyumlu hale gelecektir. Böylece, kurumların ve kişilerin bu sistemi benimsemesi kolaylaşacaktır.

Blokzincirde tutulan bilgiler şeffaf olduğundan, ilk ilkeyi blokzincir yapısı için uygulamak zor olmayacaktır. Blokzincirde tutulan veriler silinemez ve mevcut kayıtlar değiştirilemez. Fakat önerdiğimiz sistemde kişisel veriler blokzincirde tutulmayacağı için, kimlik sahibinin izni dahilinde verisinin kaldırılması işlemi sorun teşkil etmeyecektir. Kimlik sahiplerine verilen doğrulanabilir kaynakların iptal edilebilmesi, kullanım sürelerinin sınırlandırılması ve geçersiz kılınması da yine blokzincirde muhafaza edilecek iptal kayıtları sayesinde gerçekleştirilecektir.

3.8.4. EKDS Sisteminin Blokzincir Tabanlı Kimlik Yönetimi Sistemine Entegrasyonun Avantajları ve Dezavantajları

EKDS sisteminin blokzincir tabanlı bir kimlik yönetimi sistemi ile entegrasyonun farklı açılardan çeşitli potansiyel faydaları mevcuttur. Başlıca katkılarını aşağıdaki gibi listelenebilir:

- Devletin güvenirliğini güçlendirme: Böyle bir yapı ile devletlerin kişiler üzerindeki yetkisi tamamen kalkmadığı için daha güçlü hale gelmektedir.
- Kurumlardan bilgi edinme kolaylığı: Böyle bir sistem ile kişiler istediği bilgiyi kolayca dijital ortamdan edinip, bu bilgiyi daha sonra başka kurumlarla istedikleri kadarını paylaşabilir.
- Sadece yeteri kadar bilginin paylaşılması (selective disclosure): Kişiler herhangi bir yere kendilerini tanıtırken bütün bilgilerini ifşa etmek yerine sadece orası için gerekli olan bilgiyi vererek kişisel güvenliklerini artırmaktadır.
- Zaman ve maliyet tasarrufu: Bilgi edinme kolaylıkları kişiler için özellikle zamandan ve maliyetten tasarruf etmelerine olanak sağlamaktadır.
- Farklı verilerin birbirleri ile uyumlu hale gelmesi: Tek bir yapı ile verilerin farklı kurumlar arasında paylaşılması, verilerin tek ve aynı olmasını sağlamaktadır.

Sistemin avantajları yanında, muhtemel zorluklar ve dezavantajları gözden geçirilmelidir. Böyle büyük bir yapının kurulması ve büyük kurumlar tarafından desteklenmesi projeyi gerçekleştirme bakımından mutlaka olması gereken bir durumdur. Fakat kurumları böyle bir ihtiyaca ikna etmek oldukça zor bir iştir. Kurumlar sistemi kullanmaya ikna olsalar bile kullanıcıların da sistemi benimseyebilmeleri gerekir. Orta ve genç yaştaki kullanıcıların dijital sistemleri benimseme oranı yüksek olarak beklense de yaşlı kesimin böyle bir sisteme uyum sağlaması oldukça zaman alabilir. Yine yöntemden bahsederken, değinilen güvenlik ve kişisel bilgilerin tutulduğu mobil cihazların yazılım ve donanım olarak güvenliğinin sağlanması dikkatle ele alınması gereken bir diğer dezavantajdır. Yeterli güvenlik önlemleri ile kişilerin ve kurumların desteği alınınca böyle bir sistemin dezavantajları olabildiğince azalmaktadır.

EKDS sisteminin entegrasyonu ile kimlik onayı için gerekli olan devlet kurumunun bir fiil sisteme dahil edilmesine gerek kalmayıp, önerilen sistemin daha kolay kullanılabilir hale getirilmesi hedeflenmiştir. Özetle, ulusal bir kimlik yönetim sistemi için, TCKK kartlarının blokzincire entegrasyonu başlangıç için oldukça motive edici ve uygulanabilir olup ardışık (iteratif) şekilde sistemin gelecekte büyüyeceği ve kullanışlılığının artacağı değerlendirilmektedir.

3.9. SONUÇ VE DEĞERLENDİRMELER

Bu çalışmada, kullanıcıların kendi verilerine tamamen hâkim olmalarını sağlayacak, modern kriptografi ve doğrulanabilir dijital kimlik bilgileriyle desteklenen blokzincir temelli sayısal kimlik yönetim sistemleri açıklanmıştır. Geleneksel kimlik yönetimi yöntemlerinde var olan sorunlar ve zorluklar güvenlik, mahremiyet, kullanılabilirlik ve küreselleşme açısından tanımlanmış; literatürdeki mevcut çözümleri karşılaştırılıp kimlik sahiplerinin taşınabilir kimliklerine ve kimlik tabanlı kayıtlarına bağımlı olmadan kontrol ettikleri bir dijital kimlik çözümünün çerçevesi ve bileşenleri değerlendirilmiş; blokzincir tabanlı kimlik sistemlerinde biyometrik güvenlik mekanizmaları, özel anahtar şifreleme ve kurtarma yöntemleri detaylıca işlenmiş ve blokzincir tabanlı sayısal kimliklerin yeni nesil Türkiye Cumhuriyeti Kimlik kartlarına entegrasyonu detaylıca ele alınmıştır.

Blokzincir teknolojisi yıkıcı bir teknoloji olarak kabul görse de, blokzincir tabanlı teknolojilerin hayata geçmesinde kullanıcıların teknolojiye yabancı olması, geliştirme zorlukları ve yasal düzenleme eksiklikleri gibi önemli engeller mevcuttur. Aynı zamanda, blokzincir projelerinin kripto paraların popüler olduğu dönemlerde oldukça ilgi gördüğü gözlemlenmekle birlikte, kripto paraların düşüş eğilimine girdiği dönemlerde popülerliğinde azalma görülmektedir. Ülke olarak blokzincir teknolojisinden en büyük faydayı elde edebilmek için öncelikle teknolojiyi doğru anlamamız ve doğru ele almamız gerekmektedir. Blokzincir teknolojisinin yükseköğretim programlarına girmesi, teknolojinin anlaşılmasını sağlayacak ve bu konuda yetişmiş insan kaynağını artıracaktır.

Günümüzde birbirleriyle rekabet içerisinde olan firmaların bile işbirlikleri gerçekleştirmesi neredeyse bir zorunluluk haline gelmektedir. Güven probleminin giderek arttığı dünyamızda bireyler ve kuruluşları demokratik ve güvenli bir şekilde bir araya getirebilecek yapının blokzincir teknolojisi ile

mümkün olabileceği anlaşılmaktadır. Teknolojiyi doğru şekilde takip edebilmek ve kullanımını yaygınlaştırmak adına, kural ve yönetmeliklerin tüm dünya ile eşgüdümlü olması önem arz etmektedir. Blokzincir teknolojilerinin kullanımında karşılaşılan hukuki zorlukların hukukçular, kanun yapıcılar, sivil toplum örgütleri ve diğer tüm paydaşlar ile birlikte ortak akıl düzleminde değerlendirilerek uygulamada ihtiyaçları karşılayacak yönetmeliklere dönüştürülmesi ve mevcut bankacılık sistemine bütünleşmesi ile büyük kazanımlar elde edileceğini düşünmekteyiz. Finans merkezi olma hedefi olan ülkemizde kripto paraların yükselişi göz ardı edilemeyecek boyuttadır. Bu yapının yasal dayanaklarının en kısa sürede devreye alınmasının hem yatırımcılara, hem de devletimize büyük kazanç getireceği düşüncesindeyiz.

Öte yandan, büyük kurumların yürütmekte oldukları blockzincir projelerinin, yasal altyapının istenilen seviyede olmamasından dolayı iç projeler seviyesinde kaldığı ve yaygınlaşmasının gerçekleşemediği de göz ardı edilmemelidir. Blokzincir projelerinin başarısız olmasında en önemli sebeplerden biri yetişmiş insan kaynakları eksikliğidir. Bunun başlıca sebepleri arasında blokzincir konusunun yeni bir alan olması, teknik altyapısının karmaşıklığı ve öğrenim zorluğu, blokzincir konusunun örgün eğitim müfredatlarında yeterince yer bulamaması, blokzincir teknolojilerini kolayca kullanıma sunan platformların henüz yeterince geliştirilmemiş olmasını sayabiliriz. İnsan kaynaklarını geliştirme konusunda atılacak adımlar ve bu alanda yapılacak yatırımlar, uzun vadede blokzincir teknolojilerin yaygınlaştırılması ve yenilikçi girişimlerin artmasına önemli katkılar sunacaktır.

Blokzincir projelerinin genele yayılmasındaki bir diğer büyük engel de son kullanıcıların sisteme dahil olabilmek için yönetmesi gereken açık ve gizli anahtar gibi görece karmaşık yapıları yönetme güçlükleridir. Bu çalışmada ele aldığımız yeni nesil TC kimlik kartlarında halihazırda mevcut olan biyometrik veriler kullanılarak, kişilerin kendilerini sisteme tanıtmaları ile blokzincir tabanlı kimlik sistemini tamamlayıcı etki sağlayacağı değerlendirilmiştir.

Konuya dair bir diğer yaklaşım da sayısal mülkiyet sahipliği ve telif, sanal oyunlar, yardım ve bağış organizasyonları gibi hızla gelişen teknoloji sayesinde hayatımıza hızla giren kavramlardır. Sürekli gelişen sayısal/çevrimiçi ekosistemin en kritik kavramlarından biri sayısal kimliğin alt yapısının oluşturmasıdır. Bunun geleceğe yatırım açısından hayati bir öneme sahip olduğu bizce göz ardı edilmemelidir.

KAYNAKLAR

[1] A. Pascual, K. Marchini, and S. Miller, *2018 identity fraud: Fraud enters a new era of complexity*. 2018.

[2] S. Morgan, "White paper: 2017 Cybercrime Report," Herjavec Group, 2017.

[3] Inc. ShoCard, "White paper: Identity Management Verified Using the Blockchain," ShoCard, Inc., 2017.

[4] S. Nakamoto and others, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[5] R. C. Merkle, "Protocols for public key cryptosystems," in *Security and Privacy, 1980 IEEE Symposium on*, 1980, pp. 122–122.

[6] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[7] N. R. Institute, "Survey on Blockchain Technologies and Related Services. FY2015 Report," Nomura Research Institute, 2016.

[8] O. Inspector General, "Blockchain Technology: Possibilities for the U.S. Postal Service", United States Postal Service, Technical report RARC-WP-16-011, 2016. Available: https://www.uspsoig.gov/sites/default/files/document-library-files/2016/RARC-WP-16-001.pdf

[9] M. Walport, "Distributed ledger technology: Beyond blockchain," *UK Gov. Off. Sci.*, 2016.

[10] H. Foundation, "An Introduction to Hyperledger," Hyperledger Foundation, 2018.

[11] C. Cachin, "Architecture of the hyperledger blockchain fabric," in *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, 2016, vol. 310.

[12] R. Housley, W. Ford, W. Polk, and D. Solo, "Internet X. 509 public key infrastructure certificate and CRL profile," 1998.

[13] S. Foundation, "Sovrin: A Protocol and Token for Self-Sovereign Identity and Decentralized Trust," Sovrin Foundation, 2018.

[14] S. Foundation, "Sovrin: What Goes on the Ledger?," Sovrin Foundation, 2017.

[15] SecureKey, "Identity Now: The Vital Role Telecommunications Companies Play and the Tremendous Opportunity in Evolving Identity Ecosystems," SecureKey, 2017.

[16] SecureKey, "Identity Now: A Whitepaper for Banks Trying to Determine the Role They Should Play in Evolving Identity Ecosystems," SecureKey, 2017.

[17] D. R. High et al., "Obtaining a medical record stored on a blockchain from a wearable device," U.S. Patent Application No. 15/840,589, 2018.

[18] O. Jacobovitz, "Blockchain for identity management," *Lynne William Frankel Cent. Comput. Sci. Dep. Comput. Sci. Ben-Gurion Univ. Beer Sheva Google Sch.*, 2016.

[19] V. Buterin and others, "A next-generation smart contract and decentralized application platform," *White Pap.*, 2014.

[20] C. Sullivan and E. Burger, "E-residency and blockchain," *Comput. Law Secur. Rev.*, vol. 33, no. 4, pp. 470–481, 2017.

[21] C. Lundkvist, R. Heck, J. Torstensson, Z. Mitton, and M. Sena, "Uport: A platform for self-sovereign identity," 2017.

[22] M. Aydar and S. Ayvaz, "Towards a Blockchain based digital identity verification, record attestation and record sharing system," *ArXiv Prepr. ArXiv190609791*, 2019.

[23] M. Aydar, S. C. Cetin, S. Ayvaz, and B. Aygun, "Private key encryption and recovery in blockchain," *ArXiv Prepr. ArXiv190704156*, 2019.

[24] L. Tessler and T. Byrnes, "Bitcoin and quantum computing," *ArXiv Prepr. ArXiv171104235*, 2017.

[25] P. Minenkov and M. Lytaev, *sirius-sdk-python*. GitHub, 2019.

[26] S. Barman, D. Samanta, and S. Chattopadhyay, "Fingerprint-based crypto-biometric system for network security," *EURASIP J. Inf. Secur.*, vol. 2015, no. 1, p. 3, 2015.

[27] N. Bansal, "Enhanced RSA Key Generation Using Fingerprint Biometric," PhD Thesis, NIT, Jamshedpur, 2018.

[28] E. Benli, I. Engin, C. Giousouf, M. A. Ulak, and S. Bahtiyar,"BioWallet: A Biometric Digital Wallet" *In Proceedings of the Twelfth International Conference on Systems*, Venice, Italy, 2017, pp. 23-27.

[29] B. Chen and V. Chandran, "Biometric based cryptographic key generation from faces," in *9th Biennial Conference of the Australian Pattern Recognition Society on Digital Image Computing Techniques and Applications (DICTA 2007)*, 2007, pp. 394–401.

[30] D. Bhattacharyya, R. Ranjan, F. Alisherov, M. Choi, and others, "Biometric authentication: A review," *Int. J. U- E-Serv. Sci. Technol.*, vol. 2, no. 3, pp. 13–28, 2009.

[31] X. Boyen, Y. Dodis, J. Katz, R. Ostrovsky, and A. Smith, "Secure remote authentication using biometric data," in *annual international conference on the theory and applications of cryptographic techniques*, 2005, pp. 147–163.

[32] L. Hong, Y. Wan, and A. Jain, "Fingerprint image enhancement: algorithm and performance evaluation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 777–789, 1998.

[33] M. Kawagoe and A. Tojo, "Fingerprint pattern classification," *Pattern Recognit.*, vol. 17, no. 3, pp. 295–303, 1984.

[34] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *J. Soc. Ind. Appl. Math.*, vol. 8, no. 2, pp. 300–304, 1960.

[35] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[36] "TS ISO/IEC 7816-2. Kimlik kartları – Tümleşik devre kartları – Bölüm 2: Kontaklı kartlar – Kontakların boyutları ve konumu.," Apr. 16, 2014. https://intweb.tse.org.tr/Standard/Standard/Standard.aspx?081118051115108051104119110104055047105102120088111043113104073085104086085118072051115077057107 (accessed Apr. 01, 2021).

[37] V. Celik and O. Adalier, "Türkiye Cumhuriyeti Kimlik Kartı (TCKK) ve Elektronik İmza," Accessed: Jan. 04, 2021. [Online]. Available: https://kamusm.bilgem.tubitak.gov.tr/dosyalar/bildiriler/Turkiye_Cumhuriyeti_Kimlik_Karti_ve_Elektronik_Imza.pdf.

[38] "Elektronik Kimlik Doğrulama Sistemi." Accessed: Jan. 05, 2019. [Online]. Available: , https://www.ekds.gov.tr/ekds/elektronik-kimlik-dogrulama-sistemi, last accessed.

[39] "Verifiable Credentials Implementation Guidelines 1.0." Accessed: Oct. 29, 2020. [Online]. Available: https://www.w3.org/TR/vc-imp-guide/.

[40] "Dünyada Blokzinciri Regülasyonları ve Uygulama Örnekleri Karşılaştırma Raporu." Hukuk Düzenlemeler ve Kamu İlişkileri Çalışma Grubu, Türkiye Bilişim vakfı, Feb. 2019, Accessed: Oct. 27, 2020. [Online]. Available: https://bctr.org/dokumanlar/Dunyada_Blokzinciri_Regulasyonlari.pdf.

[41] M. V. Dülger, "Kişisel verilerin korunması kanunu ve Türk ceza kanunu bağlamında kişisel verilerin ceza normlarıyla korunması," *İstanbul Medipol Üniversitesi Hukuk Fakültesi Derg.*, vol. 3, no. 2, pp. 101–168, 2016.

[42] P. Voigt and A. Von dem Bussche, "The eu general data protection regulation (gdpr)," *Pract. Guide 1st Ed Cham Springer Int. Publ.*, vol. 10, p. 3152676, 2017.

[43] L. Nalbantoğlu and L. Özkaya, "Kisisel verilerin korunması: Bilinmesi gerekenler," DL Avukatlık Bürosu, Istanbul, Turkey, Tech. Rep. 1, Available: https://www2.deloitte.com/content/dam/Deloitte/tr/Documents/legal/KisiselVeriler.pdf

Bölüm 4

BLOKZİNCİRİ VE YAZILIM TANIMLI AĞLAR

Murat Karakuş

Yazılım Tanımlı Ağ (YTA) mimarisi ağ yönetimini ve yapılandırmasını iyileştirse de bu mimari birçok siber saldırı türüne karşı hala tam savunma mekanizmaları geliştirememiştir. Öte yandan, blokzinciri teknolojisi eşler arası ağlarda güvenilir bir üçüncü tarafa olan ihtiyacı ortadan kaldırdığı için merkeziyetsizlik perspektifinden YTA mimarisinin tam tersi bir yapı sunmaktadır. blokzinciri ve YTA paradigmalarının füzyonu iletişim altyapılarının verimliliğinin artırılmasına, ölçeklenebilirliğinin geliştirilmesine, merkezi aracılara bağımlılığının azaltılmasına ve tek hata noktası gibi siber güvenlik sorunlarının giderilmesine yardımcı olabilir. Bu kitap bölümü, YTA ve blokzinciri teknolojilerinin mevcut kullanımlarını ve uygulamalarını siber güvenlik perspektifinde kapsamlı bir şekilde analiz etmektedir. Ayrıca, siber güvenlik altyapılarına gizlilik, bütünlük ve kullanılabilirlik sağlamak için YTA ve blokzinciri teknolojilerini entegre etmenin fizibilitesini tartışmaktadır. Bu bölümde öncelikle kısaca YTA ve blokzinciri teknolojilerinin detayları kısaca verilmektedir. Daha sonra bu teknolojilerin erişim kontrolü, doğrulama, yönlendirme ve güvenlik gibi ağ fonksiyonları çerçevesinde işbirlikleri açıklanmaktadır. Sonrasında, blokzinciri teknolojisinin YTA çözümlerinde faydalanılan değiştirilemezlik, merkeziyetsizleştirme, şeffaflık ve güvenlik/gizlilik gibi özellikleri anlatılmaktadır. Bu ikilinin işbirliklerinin yaygın olduğu Nesnelerin İnterneti, Araçsal Ağlar, Bulut Mimarileri ve 5G gibi iletişim ağı mimarilerinde yapılan çalışmalar açıklanarak konu ile ilgili sonuç ve değerlendirmeler ise sunulmuştur.

4.1. GİRİŞ

Artan bulut servisleri, sunucu sanallaştırması, hareketlilik (mobilite) ve içerik merkezli video servisleri gibi daha birçok servis, araştırmacıları günümüzün ağ mimarilerini tekrar gözden geçirmeleri için harekete geçirmiştir. Geleneksel ağ mimarilerinde, ağ cihaz ve aygıtlarının tekrar yapılandırılmaları ve kurulumları karmaşık ve zor olduğu için bu cihazların bakımı tecrübeli ve teknik bilgisi yüksek personel gerektirmektedir. Ayrıca, mevcut bir bilgisayar ağına yeni bir cihaz eklemek veya çıkarmak ekstra maliyet getirmektedir. Bu durum bilgi teknolojileri uzmanlarının birçok anahtar ve yönlendirici ile işlem yapmasını ve Erişim Kontrol Listeleri (ACLs), Sanal Yerel Ağlar (VLANs) ve diğer mekanizmalar gibi birçok fonksiyonların güncellemesini gerektirdiğinden zaman alıcıdır [1]. Diğer taraftan, ticari talepler ve kullanıcı ihtiyaçları günden güne arttığı için uygulama geliştiriciler ve servis sağlayıcılar yeni servislerin ve olanakları araştırmaya başladılar. Aygıt üreticisine bağımlılık (vendor dependency), yavaş donanım ürün döngüsü (slow equipment production cycle), uygulama testi gibi sorunlar, geliştiricileri ve sağlayıcıları kendi ağları için yeni ağ uygulamaları ve hizmetleri geliştirmekten caydıran engeller haline gelmiştir. Bu nedenle günümüzün veri merkezleri, veri taşıyıcıları ve kampüsleri daha dinamik ağ mimarilerine ihtiyaç duymaktadır.

Yazılım Tanımlı Ağ (Software-Defined Network - YTA) [2] mimarisi geleneksel ağ mimarilerinin yukarıda belirtilen sorunlarına yanıt olarak ortaya çıkmıştır. YTA kontrol düzlemini ve veri düzlemini ayırmayı hedeflemektedir. Bu ayrım ağ operatörlerine ve yöneticilerine ağ kaynaklarının verimli ve kolay bir biçimde kullanıcılara sağlanması imkânını sunmaktadır. Ayrıca, YTA bir ağdaki tüm özellikleri değiştirme noktasında programlanabilirlik kolaylığı sağlamaktadır. Bu nedenle ağ operatörleri YTA mimarisindeki dinamik, otonom ve ticari olmayan programlarla ağ kaynaklarını kolay ve hızlı bir şekilde yönetebilir, yapılandırabilir ve optimize edebilir [3]. Google'ın Geniş Alan Ağı (B4), yukarıda belirtilen amaçlarla büyük ölçekli bir ağda benimsenen YTA örneklerinden birisidir [4]. Ek olarak, YTA'da ağ mantıksal anlamda merkezileştirildiğinden denetleyiciler geleneksel ağın aksine tüm ağ küresel görünürlüğüne sahip olurlar. Bu nedenle akış yönetimini ve kaynaklarını dinamik olarak optimize edebilirler.

Blokzinciri teknolojisi 2008 yılında anonim bir kişi veya grup olarak bilinen Satoshi Nakamoto [5] tarafından yayımlanan bir makale ile tanınmış ve yayılmıştır. Fakat bu teknolojinin temelleri geçmiş yıllarda yapılan araştırmaların sonuçlarına dayanmaktadır. 1976 yılında yayımlanan bir makale de karşılıklı olarak dağıtılan kayıtlarda kullanılmak üzere tek bir şifreleme anahtarı yerine özel anahtar ve açık anahtar terimlerini öne sürerek simetrik şifrelemenin yerini alması için asimetrik şifreleme teknolojisi önerilmiştir [6]. Blokzinciri teknolojisinin ortaya çıkmasındaki temel fikir 1982 yılında yazılan bir makalede ortaya atılan "Bizans Generalleri Problemi" dir [7]. Bu problem, merkezi bir otorite olmadan karar verme, sistemdeki tüm katılımcıların sağlıklı iletişimi ve kötü niyetli katılımcıların sistemi etkilemesini önleme gibi mantıksal ikilemleri araştırmaktadır. 1991 yılında Haber ve Stornet [8], blokzinciri teknolojisinde kullanılan veri bloklarının zaman damgası, dijital imza ve kriptografik bağlantıları kavramları üzerine bir makale yayımladı. Bu makale aslında zaman damgası sunucularına odaklanmak ile birlikte, sistem katılımcılarının mutabakata varması için bir model de önermektedir. Leslie Lampard tarafından 1998 yılında yayımlanan başka bir makale [9], sistem katılımcıları arasında uzlaşma ve karar verme süreçlerini yürütmek için ayrıntılı bir algoritmayı açıklamaktadır. Bu makale ayrıca dağıtık kayıtların ve uzlaşma mekanizmalarının kullanımını tartışmaktadır. Satoshi Nakamoto, tüm bu çalışmalara ek olarak 2008 yılında kriptografik elektronik para ve [10] nolu çalışmada bahsedilen ödeme sistemleri kavramlarını bir araya getiren bir makale yayımladı. İlgili makale ilk kripto para birimi olan Bitcoin ve temeldeki blokzinciri konseptini ortaya koymaktadır.

Blokzinciri teknolojisinin getirmiş olduğu en büyük yenilik katılımcıların halka açık bir bilgisayar ağında işlem yapmak ve etkileşimde bulunmak için birbirlerini tanımalarına veya birbirlerine güvenmelerini gerek olmamasıdır. Bu teknolojide, elektronik işlemler merkezi bir otoriteye ihtiyaç duyulmadan şifreleme algoritmaları aracılığıyla insan müdahalesi olmaksızın ağdaki katılımcılar tarafından otomatik olarak gerçekleştirilir [11]. Blokzinciri birden fazla bağımsız tarafın verilerin ortak bir sürümünü paylaşmasına ve bu verileri kendi aralarında senkronize etmesine olanak tanır. Böylece farklı tarafların ortak mülkiyetindeki verilerin versiyonlarını birbirlerine doğru şekilde yansıtmak için uzlaşma, yayma ve senkronize etme ihtiyaçlarını ortadan kaldırır.

Bu kitap bölümü, YTA mimarisi ve blokzinciri teknolojilerinin mevcut kullanımlarını ve uygulamalarını siber güvenlik perspektifinde kapsamlı bir şekilde analiz etmektedir. Ayrıca, siber güvenlik altyapılarına gizlilik, bütünlük ve kul-

lanılabilirlik sağlamak için YTA ve blokzinciri teknolojilerini entegre etmenin fizibilitesini tartışmaktadır. Bu bölümün kalanında öncelikle okuyucu sıkmayacak şekilde YTA ve blokzinciri teknolojilerinin detayları anlatılmaktadır. Daha sonra YTA ve blokzinciri teknolojilerinin erişim kontrolü, doğrulama, yönlendirme ve güvenlik gibi ağ fonksiyonları çerçevesinde işbirlikleri açıklanacaktır. Sonrasında, YTA çözümlerinde faydalanılan blokzinciri teknolojisinin değiştirilemezlik, merkeziyetsizleştirme, şeffaflık ve güvenlik/gizlilik gibi özellikleri anlatılacaktır. Sonuç ve değerlendirmelerin öncesinde ise YTA ve blokzinciri teknolojilerinin işbirliklerinin yaygın olduğu Nesnelerin İnterneti (Internet of Things - IoT), Araçsal Ağlar (VANETs), Bulut Mimarileri (Cloud Architectures) ve 5G gibi iletişim ağı mimarilerinde ki çalışmalar ortaya konacaktır.

4.2. YAZILIM TANIMLI AĞ (YTA) MİMARİSİ

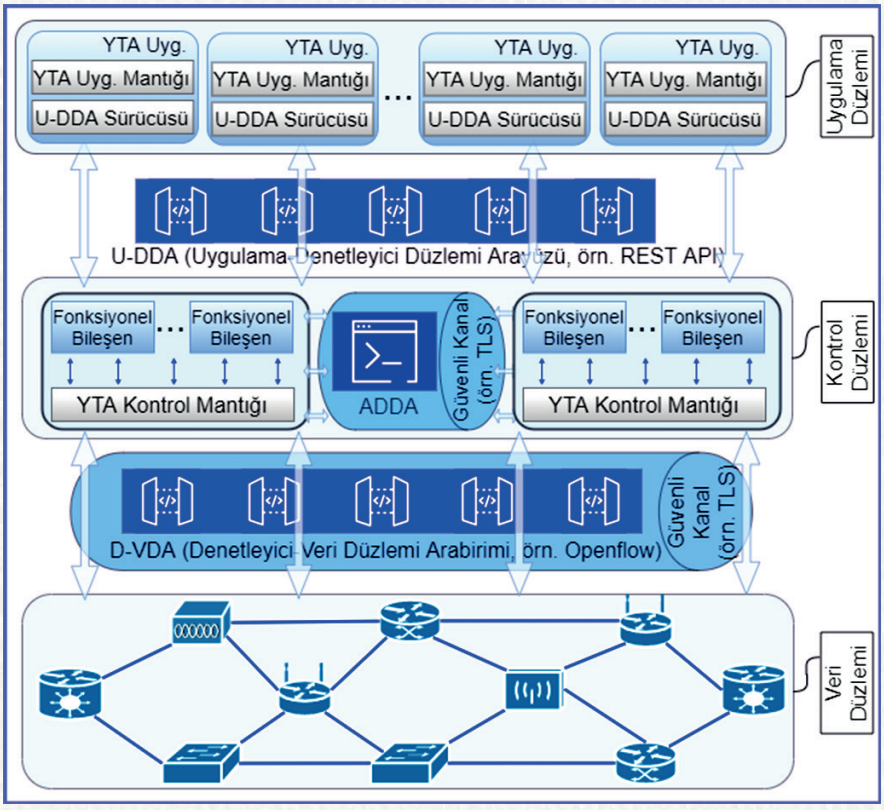
YTA mimarisi, OpenFlow [12] gibi yeni ve açık kaynak kodlu Uygulama Programlama Ara Yüzleri (Application Programming Interface - API) kullanarak ağ programlanabilirliğini sağlamaktadır. Ağ üzerinde tam bir kontrol elde etmek için bir ağ denetleyicisine ev sahipliği yapan kontrol düzlemi ile sadece anahtarlama fonksiyonunun gerçekleştirildiği veri düzleminin birbirlerinden izole edilmesini gerçek kılmaktadır. [13] nolu kaynakta verilen rapora göre YTA teknolojisin pazar büyüklüğünün 2018 de 8,8 Milyar USD'den 2023'e kadar 28,9 Milyar USD'ye çıkması beklenmektedir. Bu verilerin de yansıttığı üzere, YTA teknolojisinin kurumsal ağlardan veri merkezi ağlarına kadar çeşitli ağ dağıtımlarındaki uygulamaları artarak devam etmektedir.

OpenFlow protokolüne sahip bir YTA mimarisi, ağ operatörlerinin akışları (flows) denetleyiciler aracılığıyla geleneksel ağlara kıyasla daha hassas bir şekilde yönetmesine imkân sağlar. Geleneksel bir ağda, akışlar temel olarak en uzun hedef IP önekleri (longest IP prefixes), hedef MAC adresleri veya IP adresleri ve TCP Port numaralarının bir kombinasyonu gibi paket başlıklarının tek veya birkaç öznitelik kombinasyonuna dayalı olarak karakterize edilirler. YTA, akışları OpenFlow protokolü gibi bir Denetleyici-Veri Düzlemi Arayüzü (D-VDA) aracılığıyla paket başlıklarının daha fazla özniteliğine göre yönetmemizi mümkün kılmaktadır. Şekil 4.1'de gösterildiği üzere, YTA mimarisinin kullanımının yaygınlaştırılmasında öncü kurumlardan olan Open Networking Foundation (ONF) [14], YTA mimarisini dikey

olarak üç ana düzleme ayrılmakta [15] olup, bu düzlemler aşağıda alt başlıklarda açıklanmıştır.

4.2.1. Veri Düzlemi

Veri düzlemi alt düzlem olup yönlendiriciler, fiziksel/sanal anahtarlar, erişim noktaları gibi ağ cihazlarından oluşur. Bu cihazlar YTA denetleyicileri tarafından D-VDA'lar aracılığıyla erişilebilir ve yönetilirler. Ağ elemanları ve denetleyicileri, TLS (Transport Layer Security) gibi güvenli kanallar (secure channels) aracılığıyla iletişim kurabilirler. OpenFlow protokolü, denetleyiciler ve veri düzlemi cihazları arasındaki iletişim için kullanılan en yaygın standart D-VDA'dır.



Şekil 4.1. YTA Mimarisinin Ana Düzlemleri: Veri Düzlemi (Data Plane), Kontrol Düzlemi (Control Plane) ve Uygulama Düzlemi (Application Plane).

Şekil 4.1'den de görülebileceği gibi OpenFlow ile çalışan bir YTA anahtarında üç ana bölüm vardır: *Akış Tablosu*, *Güvenli Kanal* ve *OpenFlow Protokolü*. Bir OpenFlow anahtarı, akış girdilerinin (flow entries) listesini içeren bir dizi akış tablosu tutar. Her akış girdisi 3 bölümden oluşur: Akış girdisini kaynak/hedef adresleri gibi belirli başlık özniteliklerine göre tanımlamak için bir *Kural* alanı, Kural alanındaki değerlerle eşleşen bir pakete uygulanacak *Aksiyon (Action)* alanı ve girdiler için bazı sayaçları (counters) tutmak üzere bir *İstatistikler* alanı [12]. Güvenli kanal (ör: TLS), veri düzlemini uzak noktada bulunan bir denetleyiciye bağlayan aracı birimdir. Anahtarlar, güvenli kanal kullanılarak denetleyiciler tarafından yönetilir ve yapılandırılırlar. Ek olarak, denetleyici anahtarlardan ağ olaylarını alır ve bu kanal üzerinden anahtarlara ilgili paketleri gönderir.

4.2.2. Kontrol Düzlemi

Bir YTA kontrol düzlemi, D-VDA aracılığıyla ağ yönlendirme politikalarını denetleyerek kontrol fonksiyonları sağlamak için bir veya daha fazla YTA denetleyicisinden oluşur. Bir kontrol düzlemi üzerindeki denetleyiciler arasında iletişimi sağlayan ara denetleyici düzlem arayüzü (A-DDA) [16], denetleyiciler ve ağ aygıtları arasında iletişimi sağlayan D-VDA ve denetleyiciler ile uygulamalar arasında iletişimi sağlayan uygulama-denetleyici düzlemi arayüzü (U-DDA) gibi ara yüzlere sahiptir. Bir U-DDA, ağ uygulamaları/servisleri ile denetleyiciler arasında iletişimi ağ güvenliği ve yönetimi gibi durumlar için mümkün kılar. U-DDA, YTA topluluğu tarafından çoğunlukla "Kuzey Bağlantı Arayüzü (Northbound Interface)" olarak da adlandırılır. Bir denetleyici iki ana bileşenden oluşur: Fonksiyonel bileşenler ve kontrol mantığı. Denetleyiciler, denetleyici politikalarını yönetmek için koordinatör ve Sanallaştırıcı gibi birden fazla fonksiyonel bileşen içerebilirler. YTA kontrol mantığı (SDN Control Logic), ağ uygulamaların gereksinimlerini ağ cihazlarının kaynakları ile komutlar kullanarak eşleştirir [15].

4.2.3. Uygulama Düzlemi

YTA uygulama düzlemi, iç karar verme süreçleri için ağın özet görünümünü kullanmak amacıyla denetleyiciler ile etkileşime giren güvenlik duvarı

(firewall), yük dengeleyici (load balancer) gibi çeşitli ağ uygulamalarından oluşur. Bu uygulamalar denetleyiciler ile açık bir U-DDA (ör: REST API) aracılığıyla iletişim kurarlar. YTA uygulaması bir YTA uygulama mantığı (SDN App Logic) ve U-DDA Sürücüsünden oluşur.

4.3. YAZILIM TANIMLI AĞ MİMARİSİNİN SAĞLADIĞI AĞ FONKSİYONLARI

YTA ağ yöneticilerinin hizmet odaklı yönlendirme modelleri geliştirmesine yardımcı olabilir [17]. YTA mimarisi ile bireysel akış (flow) bazlı yönlendirme (hem ağ içi hem de transit akışlar için), geleneksel mimarilere kıyasla daha ölçeklenebilir, daha basit ve daha az zaman alan mekanizmalarla uygulanabilir hale gelmektedir. OpenFlow, ağ operatörlerine veri düzleminde hizmet kalitesi odaklı akışlar (QoS-based flows) gibi özel gereksinimli akışları yöneten yönlendirme tabloları oluşturmak için klasik en kısa yol algoritmalarından farklı ve akıllı yönlendirme algoritmalarını kullanma yeteneği sağlar. Ayrıca, ağ cihazlarının kontrol ve yönlendirme işlevlerinin ayrılması sayesinde akışların dinamik yönlendirmesi ağ denetleyiciler tarafından mümkün hale gelmektedir. Bireysel akış bazlı ve dinamik yönlendirme gibi beceriler, ağ yöneticilerinin ağları için farklı hizmet kalitesi odaklı yönlendirme mekanizmaları oluşturmalarına olanak sağlar.

Diğer taraftan, ağlar arasındaki uçtan uca hizmet kalitesi gereksinimli akışların yönlendirilmesi, ağların izlenerek her bağlantı için gecikme, bant genişliği ve paket kaybı oranı gibi güncel ağ geneli durum bilgilerinin ve ilgili istatistiklerin toplanmasını gerektirir. YTA, ağ yöneticilerinin ağ dinamiklerini izlemelerine ve paket, port, yönlendirme tablosu ve kuyruk (queue) bazlı detaylı ve farklı düzeylerde sayaçlar aracılığıyla güncel küresel ağ durumu istatistikleri toplamasına olanak tanır. Bu kabiliyetler sayesinde, ağ denetleyicileri anahtarlama kurallarını (gerekirse) bireysel akış bazlı olarak tanımlayabilir ve bunları farklı ağ aygıtlarına uygulayabilirler. YTA'nın tüm bu yetenekleri kuşkusuz hizmet odaklı yönlendirme için önemli ve etkili fırsatlar sunmaktadır.

Son olarak YTA, ağ programlanabilirliği, donanım/yazılım bağımsızlığı, sanallaştırılmış yazılım altyapısı (virtualized software infrastructure), çoklu-kiracı mimarisi (multi-tenancy) ve kaynak havuzu oluşturma (resource pooling) gibi bazı ana özellikleri sayesinde İnternet Servis Sağlayıcıların

– İSS ağlarının sermaye (CAPEX) ve operasyonel (OPEX) harcamalarını azaltmalarına yardımcı olabilir [18]–[20]. YTA, bir ağın sermaye harcamalarını farklı şekillerde etkileyebilir. Bir YTA'da şirketler tarafından tasarlanan özel yazılımların karmaşık özelliklerine ihtiyaç duyulmadığından, her veri katmanı ağ aygıtı sadece anahtarlama yapan daha ucuz cihazlar olacaktır. Ağ denetleyicileri, YTA durumunda tüm ağ bazlı kontrole sahip olabileceğinden yük dengeleme (load balancing) gibi bazı yöntemlerle daha iyi ağ kaynağı kullanımını mümkün kılabileceklerdir. Bu nedenle, sermaye harcamalarını artırabilecek ağın aşırı tahsisine (network over-provisioning) ihtiyaç kalmaz. YTA, servis sağlayıcılar için bazı ana operasyonel harcama kalemlerini de azaltmayı vaat ediyor. Bu mimari de anahtarlar ihtiyaç duyduğu toplam enerjinin çoğunu tüketen gömülü bir kontrol düzlemi bulundurmazlar. Ayrıca, YTA ağ cihazları üzerinde daha verimli trafik optimizasyonuna izin verdiğinden, ağda enerji tüketen toplam cihaz sayısını azaltır. Buna ek olarak, YTA durumunda bağımsız olarak yönetilmesi, bakımlarının yapılması ve onarılması gereken, farklı şirketlere ait cihazlar bulunmaz. Ağ senaryolarının otomatik yapılandırılabilmesi ve programlanabilmeleri sayesinde ağ görevlerine daha az personel ihtiyacı ve azaltılmış manuel yapılandırma gibi nedenlerden dolayı YTA senaryosunda servis sağlama maliyetinin daha düşük olması beklenmektedir.

4.4. BLOKZİNCİRİ TEKNOLOJİSİNİN TEMELLERİ

Blokzinciri teknolojisi, blokzinciri ağına katılan tüm katılımcıların merkezi bir otorite olmadan ağdaki işlemleri takip edebileceği demokratik bir sistem sunmaktadır. Başka bir açıdan, sistem gerçekleştirilen tüm işlemleri kaydeden dağıtık bir veri tabanıdır. Blokzinciri, belirli protokoller ve güven mekanizmaları sayesinde varsayılan bir güven ilişkisi veya merkezi güvenilir üçüncü bir taraf olmadan katılımcılar arasında dağıtık bir veri tabanı sağlayan bir yapı ortaya koymaktadır. Blokzinciri teknolojisinde, işlem adı verilen her kayıt bloklara yazılır ve genel blokzinciri yapısını oluşturmak için zincire eklenir [21].

4.4.1. İşlem Yapısı

Bir blokzinciri işlemi, genel anlamda bir blokzinciri ağında tek bir veri aktarım işlemi veya bu işlemin blokzincirin de kaydedilmesidir. Bu işlemler blokzincire kaydedilen durumları güncellerler. Kullanım durumlarına bağlı olarak

çeşitli veriler içerebilirler. Örneğin, kripto para birimi işlemleri durumunda durum bilgisi kripto para birimlerinin hesaplar (adresler) arasında transferleri ile ilgilidir. İşlem, oluşturulduktan sonra işlemi oluşturanın imzası ile imzalanır. Bu imza parayı harcama veya işlemler ile ilişkili veri parametrelerini iletme yetkisi sağlar. Dijital olarak imzalanmış bir işlemin gerçekleştirilebilmesi için gerekli tüm bilgileri içermesi gerekir.

Şekil 4.2. Bir Blokzincirindeki Genel İşlem Veri Yapısı.

Şekil 4.2'de bir blokzincirinde genel işlem yapısını göstermektedir. İşlem yapıları blokzincirin kullanıldığı alana göre birçok farklı bilgi içerebilir.

- *İşlem No*: Bu bölüm blokzinciri işleminin benzersiz kimliğini oluşturur ve genellikle bu işlemin kriptografik özetinden (hash) oluşur.
- *Dijital İmza*: Blokzinciri düğümü tarafından gizli anahtarı kullanılarak imzalanan işlemin dijital imzasıdır.
- *Açık Anahtar*: Bir işlemi oluşturan blokzinciri düğümünün kriptografik açık anahtarıdır.
- *Boyut*: Bu bölüm işlemin boyutu gösterir.
- *Veri*: Bu kısım blokzinciri uygulamasına bağlı olarak diğer gerekli bilgileri içerir. Örneğin; Bitcoin blokzinciri ağında Girdiler ve Çıktılar şeklinde veriler mevcuttur. Bu alan blokzincirinin ihtiyaçlarına göre farklılık gösterebilir ve bu sebep ile farklı renkte gösterilmiştir.

Blokzincirinde bir düğüm tarafından üretilen bir işlem, blokzinciri ağındaki eşlik (peering) ilişkisi bulunan diğer düğümlere işlemin geçerliliğini kontrol etmesi için gönderilir. Bu düğümler de işlemleri tekrar doğrular ve işlemler ağdaki her düğüme ulaşana kadar kendi eşlik ilişkisi bulunan diğer düğümlere gönderilmeye devam eder. Geçersiz işlemler göz ardı edilerek blokzinciri defterlerine eklenmezler. İşlem doğrulama kuralları, bir işlemi tanımlamak için hangi verilerin gerekli olduğunu belirlerler. Bu kurallar şunları içerir:

(i) Biçimsel doğruluk – işlemin gerekli tüm verileri içermesi ve verilerin doğru biçimde verilmesinin sağlanması,

(ii) Anlamsal doğruluk – işlem verilerinin bağlamı ve beklenen etkisi ile ilgilidir ve

(iii) Yetkilendirme – hesap sahibinin kendi adına bir işlem gerçekleştirmesi için blokzincirine talimat vermesi gerektiği anlamına gelir.

Bu kurallar, blokzincirinin uygulama amacına özgüdür ve bu amaç ile paralel olarak değişiklikler gösterebilir. Başka bir deyişle, dijital para sahipliğini yönetmeyi amaçlayan bir blokzinciri, tedarik zinciri yönetimi ele alan başka bir blokzincirinden farklı kimlik doğrulama kurallarına sahip olabilir. Bu doğrulama kuralları kullanım amacına göre yeni kurallar ekleme veya çıkarma şeklinde değişiklik yapılmasına müsait olmalıdır.

4.4.2. Blok Yapısı

Bir blok, yapılarının ve imzalarının onaylanmasından sonra onu oluşturan düğüm tarafından belirli bir veri yapısına göre bir araya getirilen bir dizi blokzinciri işleminden oluşur. Bir blok, blokzinciri defterinde ki kendisinden önceki bloğa blokzincirin güvenliğini garanti eden önlemlerden biri olan bir kriptografik özet işaretçisi (hash pointer) içerir. Şekil 4.3'te blokzincirlerinde kullanılan blokların içerdiği temel veri alanlarını göstermektedir. Bir blokzincirinde bloklar blok bilgilerini içeren blok başlığı ve blokzinciri işlemlerini içeren blok defteri olmak üzere iki bölümden oluşmaktadır. Blok başlığını oluşturan veri alanları, blokzinciri kullanım senaryolarına ve kullanılan uzlaşma protokolüne bağlı olarak değişiklik gösterebilir.

Şekil 4.3. Blokzincirindeki Bir Bloğun Veri Yapısının Temsili

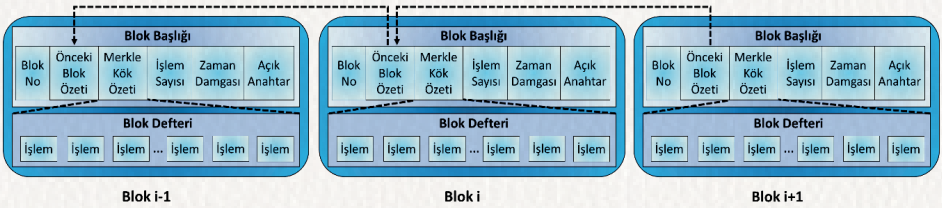
Bir bloğun yapısı temel olarak bloğu tanımlamak için çeşitli verileri içeren bir blok başlığından ve blok içindeki işlemlerin listesini içeren bir blok defterinden oluşur.

- *Blok No*: Bloğun benzersiz numarasıdır.
- *Önceki Blok Özeti*: Blokzincirinde ki önceki bloğun kriptografik özetidir. Blokzincirin tutarlılığını sağlar.
- *Merkle Kök Özeti*: Merkle ağacının kök özetidir.
- *İşlem Sayısı*: Bloğun içerdiği işlem sayısıdır.
- *Zaman damgası*: Bloğun oluşturulduğu zamanı gösterir.
- *Açık Anahtar*: Bloğu oluşturan blokzinciri düğümünün kriptografik açık anahtarıdır.

Blokzincirine yeni bir blok ağdaki doğrulama düğümleri arasında uzlaşma sağlandığında eklenir. Blok doğrulama kuralları, blok başlıklarının biçimsel ve anlamsal doğruluğuna odaklanır. Bu ilkeler işlemin veri içeriğine karşı agnostiktir. Aksine, bilginin blokzinciri veri yapısına nasıl uygulandığı ile ilgilidir. Buna göre, blok başlığının doğrulanması bir uzlaşma protokolü ile gerçekleştirilir. Bu nedenle blok doğrulaması uzlaşma protokolünün kuralları tarafından belirlenir. Örneğin Bitcoin ağı blokzincirine eklenecek bir bloğun geçerli olup olmadığını varsaymak için İş Kanıtı (Proof-of-Work – PoW) konsensüs protokolünü kullanır.

4.4.3. Blokzinciri Yapısı

Şekil 4.4'te bir blokzincirinin genel veri yapısını göstermektedir. Bir blokzincirinde her kullanıcı blokzincirinde yayımlanan işlemleri kontrol ederek geçersiz işlemler içeren blokları kabul etmezler. Önceki bloğun kriptografik özet değeri de kontrol edilir. Bu şekilde, Genesis (başlangıç) bloğu adı verilen ilk bloğa kadar oluşturulan tüm bloklar doğrulanır. Sistemdeki katılımcıların hiçbiri, verileri değiştirme suretiyle önceki blokları değiştiremez. Bu durum, blokzinciri teknolojisini verilerin değiştirilmesine karşı dayanıklı (tamper-proofness) bir yapı haline getirmektedir. Blokzincirine yeni bir blok ekleneceğinde ağdaki tüm düğümlerin blok üzerinde uzlaşmaya varmaları gerekmektedir. Literatürde blokzinciri veri bütünlüğünü sağlayan farklı özelliklere sahip çeşitli uzlaşma protokolleri mevcuttur [22].



Şekil 4.4. Blokzinciri Veri Yapısının Bir Temsili.

Blokzinciri teknolojisi bir blokzinciri operasyonlarına katılan bütün taraflar için aşağıdaki noktaları mümkün hale getirmektedir:

- *Rakipler Arasında İşbirliği.* Normalde birbirleriyle rekabet halinde olan taraflar, aralarından birinin kuralları gizlice aldatacağından korkmadan doğrudan işbirliği yapabilecekleri ortak bir platforma sahip olurlar.
- *Esneklik*. Bir blokzinciri uygulamasında katılımcıların kimliklerinin ne olduğu veya bu kimliklerin zaman içerisinde aynı kalıp kalmadıkları önemlidir. Bu nedenle, paydaş sentezinin sürekli döngü içerisinde olduğu uygulamalar tasarlanabilir.
- *Dayanıklılık*. Birçok blokzinciri uygulamasının şeffaflık özelliği, sisteme dâhil olan üyelerin sayısının artmasını sağlar. Bu artış, sistemi tehditlere ve sansüre karşı dayanıklı hale getirir.
- *Çoklu ve Dağıtık Kimlik Doğrulama*. İşlemler farklı yerlerde saklanır ve birçok taraf tarafından bağımsız olarak onaylanabilir.

4.4.4. Uzlaşma Protokolleri/Süreçleri

Bir blokzinciri ağını etkinleştiren birçok fonksiyonel bileşen olmasına rağmen, dağıtık bir uzlaşma protokolü, tüm katılımcıların üçüncü taraf bir otoritenin müdahalesi olmadan karşılıklı bir işlem defteri üzerinde anlaşmasını garanti ederek blokzincirinin âdemi merkeziyetçiliğini kolaylaştıran ve ağdaki güveni artıran bir işlevselliktir. Dağıtık bir uzlaşma protokolü, blokzincirine katılan düğümlerde mesaj alışverişi ve yerel karar verme kurallarını belirler. Bir uzlaşma mekanizmasındaki tüm tasarım stratejileri kullanıldığı blokzinciri ağının esneklik, ölçeklenebilirlik ve dayanıklılık dâhil olmak üzere bir birçok açıdan performansını etkiler.

Blokzinciri yapılarında zincire yeni blok ekleme mekanizması her biri kendi algoritma seçimine sahip iki temel adımı içerir: (i) blok önericinin seçimi ve (ii) önerilen bloğun doğruluğu üzerinde anlaşma (blok doğrulama). Birinci adım, hangi düğümün bir sonraki bloğu oluşturacağını belirlemeyi içerirken, ikinci adım blokzincirine yeni bir bloğun kabul edilip edilmeyeceğine odaklanır. Her iki adımda tamamen bağımsız olmasına rağmen, blokzinciri için eşit derecede önemlidir.

Blokzincirinde kullanım durumlarına ve ihtiyaçlara bağlı olarak farklı özelliklere sahip çeşitli uzlaşma protokolleri vardır [23]. PoW, uzlaşma için kullanılan en yaygın uzlaşma mekanizmalarından birisidir. PoW, zincire bir blok eklemek için gereken matematiksel işlem gücünü temsil eder ve madencilik (mining) adı verilen faaliyetleri içerir. Bilinen ve yaygın bir diğer mutabakat mekanizması olan Proof-of-Stake (PoS) madencilik faaliyeti içermemekte ve blok yayımlama süreci katılımcının ağdaki pay oranına göre belirlenmektedir. Uzlaşma protokolleri hakkında daha fazla ayrıntı ve kapsamlı protokol örnekleri literatürdeki ilgili derleme çalışmalarında bulunabilir [24].

4.4.5. Blokzinciri Türleri

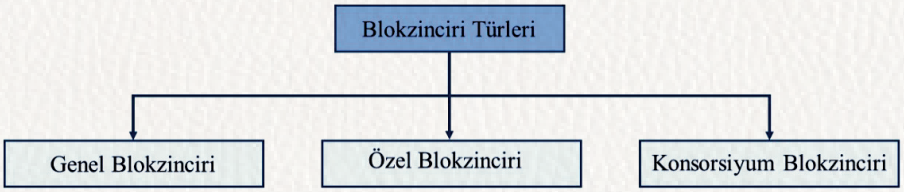
Blokzinciri sistemleri farklı erişim izin modellerine göre sınıflandırılabilirler. Bir blokzinciri ağında katılımcılar ağın türüne göre genellikle şu haklara sahip olabilirler:

(i) blokzinciri üzerindeki verilere erişme (okuma),

(ii) işlemleri blokzincirine gönderme (yazma) ve

(iii) blokzinciri durumunu yeni bloklarla güncellemedir.

Çeşitli kriterlere göre blokzinciri sistemleri Şekil 4.5'te gösterildiği gibi genel, özel veya konsorsiyum olarak sınıflandırılır.



Şekil 4.5. Blokzinciri Türleri

4.4.5.1. Genel Blokzincirleri

Genel bir blokzinciri, çeşitli kuruluşlardan, altyapıdan ve alanlardan kişilerin katılması, işlem yapması ve madencilik yapması için açık bir platform sağlar. Bu operasyonların hiçbirinde kısıtlama yoktur. Bu blokzincirini herkes indirebilir ve buradaki verileri okuyabilir. Dileyen herkes bu blokzincirinin bir parçası olabilir. Blokzincirine yeni katılan herhangi bir yeni düğüm dağıtık defterin bir kopyasını alabilir. Bu nedenle, bu tip blokzincirlerine izinsiz blokzincirleri de denir. Her katılımcıya herhangi bir zamanda işlemleri okuma ve yazma, blokzincirinde denetim gerçekleştirme veya blokzincirinin herhangi bir bölümünü gözden geçirme konusunda tam yetki verilir. Bu tür blokzincirleri açık ve şeffaftır. Belirli bir onaylayıcı düğüm yoktur. Tüm kullanıcılar işlemleri toplayabilir ve madencilik ödülleri kazanmak için madencilik sürecine başlayabilirler. Blokzincirinin tüm düğümler ile senkronize edilmiş kopyalarının var olması onu değiştirilemez kılar.

Tablo 4.1. Farklı Blokzinciri Türleri ve Özellikleri [25]

Özellik	Genel	Özel	Konsorsiyum
İzin Durumu	Evet	Hayır	Hayır
Değişmezlik	Evet	Kısmi	Kısmi
Verimlilik	Düşük	Yüksek	Yüksek
Merkeziyetçilik	Hayır	Evet	Kısmi olarak
Kimlik	Anonim	Onaylı katılımcılar	Onaylı Katılımcılar
Erişim	Açık okuma/ yazma	Kısıtlanabilir	Kısıtlanabilir
İşlem Hızı	Yavaş	Hızlı	Hızlı
Uzlaşmaya Katılım	Bütün Düğümler	Tek Kuruluş	Birden çok kuruluştan seçili düğümler

Bitcoin [5] ve Ethereum [26] genel blokzincirlerinin en iyi bilinen örnekleridir. Bu platformlarda, blokzinciri durumları dijital paralardır (kripto para birimleri) ve bir işlem paraları bir adresten diğerine taşır. Bitcoin ve Ethereum blokzincirleri dâhil çoğu genel blokzinciri platformu, uzlaşma için PoW'ın varyantlarını kullanır. Tablo 4.1'de farklı blokzincir türlerini ve özelliklerini özetlemektedir.

4.4.5.2. Özel Blokzincirleri

Özel blokzinciri sistemleri bir grup birey arasında (tek bir kuruluşta) veya birden fazla kuruluş arasında özel veri paylaşımını ve alışverişini kolaylaştırmak için kurulurlar. Ayrıca, özel bir davet almadıkları sürece bilinmeyen kullanıcılar bu tür blokzincirlerine erişim sağlayamayacağı için onlara izinli blokzincirleri de denir. Düğümlerin katılımına ya bir dizi kural ya da erişimi kontrol eden ağ tarafından karar verilir. Bu durum Nakamoto tarafından tanımlanan tam âdemi merkeziyetçilik ve açıklık gibi temel blokzinciri özellikleri ile çatışarak, ağı merkezileştirmeye daha çok yöneltir. Özel bir blokzinciri sisteminde, düğümler ağın bir parçası olduklarında, merkezi olmayan bir ağın çalıştırılmasına katkıda bulunurlar. Her düğüm defterin bir kopyasını tutar ve güncelleme için bir uzlaşmaya varmak için işbirliği yapar. Ancak, genel blokzincirinden farklı olarak veri yazmaları sınırlıdır. Hyperledger [27] ve Multichain [28] en popüler özel blokzincirlerindendir.

4.4.5.3. Konsorsiyum Blokzincirleri

Bir konsorsiyum blokzinciri kısmen özel ve izinli bir blokzinciri olarak düşünülebilir. Burada tek bir kuruluş uzlaşma ve blok doğrulamadan sorumlu değildir. Bunun yerine önceden belirlenmiş düğümler kümesi bu görevi yerine getirir. Bu düğümler kimin ağın parçası olacağına ve kimin madencilik yapabileceğine karar verirler. Blok doğrulaması için bloğun yalnızca bu düğümler tarafından imzalanmışsa geçerli kabul edildiği çoklu imza şeması kullanılır. Bu nedenle, tamamen merkezileştirilmiş özel blokzincirinin ve tamamen merkezi olmayan genel blokzincirinin aksine, bazı seçilmiş onaylayıcı düğümlerin kontrolünden ötürü kısmen merkezi bir sistemdir. Konsorsiyum, okuma veya yazma izinlerinin herkese açık veya ağ katılımcılarıyla sınırlı olup olmayacağına karar verir. Ayrıca, mutabakatın bir dizi düğümle sınırlandırılması, konsorsiyumun çoğunluk tarafından kontrolü blokzincirinde değişikliğe yol açabileceğinden, değiştirilemezlik ve geri döndürülemezlik özellikleri garanti etmez.

4.4.6. Akıllı Sözleşmeler

Akıllı sözleşme (smart contract), blokzinciri üzerine yerleştirilmek üzere tanıtılan yeni bir kavramdır. Akıllı sözleşme, güvenilmeyen taraflar arasındaki bir anlaşmanın şartlarını birlikte çalışmak için kolaylaştırmak ve uy-

gulamak için kurallara tabi olarak çalışan çalıştırılabilir bir kod bütünüdür. Bu konsept, aralarında anlaşmaya varılan hüküm ve koşullara göre ilgili tarafların tamamına veya bir kısmına dijital varlıkları serbest bırakan bir sisteme ortaya koymaktadır. Akıllı sözleşmeyi geleneksel sözleşmeyle karşılaştırdığımızda, geleneksel sözleşmelerde istediğiniz sonuçlara ulaşmak için üçüncü bir tarafa ve çok sayıda evraka ihtiyacınız vardır. Bu da daha yüksek işlem maliyetine yol açacaktır. En yaygın akıllı sözleşme geliştirme platformu Ethereum'dur [29].

Akıllı sözleşme, kendi kendini yürüten sözleşmelerdir. Bu sözleşmeler, bir kod satırı biçiminde blokzincirine kaydedilirler. Bu kodlar ve anlaşmalar, dağıtık ve merkezi olmayan bir blokzinciri ağına dâhil edilirler. Akıllı sözleşmeler, bir aracı olmadan anonim taraflar arasında gerçekleştirilecek güvenilir işlemler ve anlaşmalar sağlar. Akıllı sözleşmedeki her işlev işlemin gerçekleşmesi için yürütülür. Yürütme gerçekleştiğinde, akıllı sözleşmedeki değişkenlerin durumu, işlevin mantıksal uygulamasına göre değişecektir. Sözleşmelerin bir kopyası tüm kullanıcılar için bütün veri tabanına dağıtılacaktır. Blokzinciri ağındaki herhangi bir kullanıcı, yeni bloğun doğrulanmasının bir parçası olarak, her üye için her bir düğümde sözleşme kodu yürütülecek her türlü işlem gönderme işlevini tetikleyebilir [30].

4.5. YAZILIM TANIMLI AĞ VE BLOKZİNCİRİ TEKNOLOJİLERİNİN POTANSİYEL İŞBİRLİĞİ ALANLARI

Blokzincirinin uygulama alanları kabaca finansal ve finansal olmayan alanlar olarak ayrılabilir. Takas işlemleri ve değer transferleri finansal alanlardaki baskın uygulamalar iken doğrulama, erişim izinleri, yönetim, gözlem ve takip finans dışı alanlarda blokzincirinin yaygın uygulamalarıdır [31]. Blokzinciri teknolojisi hala gelişen bir teknoloji olsa da hızla yeni uygulama alanları bulmaktadır. Bu teknoloji önümüzdeki yıllarda araştırmacıların, geliştiricilerin ve uygulayıcıların sayısı arttıkça hızla gelişecek ve birçok kullanım alanı bulacaktır. Bu bağlamda literatürde YTA ve blokzinciri teknolojilerinin işbirliği uygulamalarını ve/veya kullanım durumlarını sunan çalışmalar da bulunmaktadır.

Tablo 4.2. YTA ve Blokzinciri Teknolojilerinden Faydalanan Çalışmaların İşbirliği Amaçları, İncelediği Sorunlar ve Çözüm Yaklaşımları

Eser	Blokzinciri ve YTA İşbirliği Amacı	Ele Alınan Problem	Çözüm Yaklaşımı
[32]	Erişim Kontrolü	Çoklu alanlar arasında verimli ve güvenli ağ kaynağı dilimleme.	Çoklu-Alan-Orkestratörü'nde kimlik yönetimi için akıllı sözleşmelerin kullanımı
[33]	Erişim Kontrolü	YTA ağ ve denetleyici bilgilerinin bütünlüğünün sağlanması	Blokzinciri tabanlı dağıtık bir kimlik doğrulama mekanizması oluşturulması
[34]	Erişim Kontrolü	Kablosuz sanallaştırmada verimliliği ve güvenliği artırma	Ağ için YTA ve kablosuz altyapılara çift harcama saldırısı için blokzinciri kullanımı
[35]	Erişim Kontrolü	Kişisel verilerin kontrolünün ve yönetimin sağlanması	Blokzinciri tabanlı merkezi olmayan kişisel veri erişim kontrol sistemi
[36], [37]	Erişim Kontrolü	Kişisel verilere erişim kontrolü	Üçlü bir model: veri yönetimi protokolü, mesajlaşma servisi ve veri depolama sistemi
[38]	Doğrulama	Ağdaki kimlikler için anonim kimlik doğrulama	Kök Sertifika Yetkilisi (KSY) ve Hiyerarşik SY kullanarak doğrulanabilir kimlikler oluşturmak
[39]	Doğrulama	IoT ağlarının botnet saldırılarına karşı güvenceye alınması	Akıllı sözleşmeler kullanarak botnet DDoS saldırılarına blokzinciri kullanılması
[40]	Doğrulama	Fiziksel, ağ ve uygulama katmanlarında IoT güvenliği	Güven sorunu olan tarafların YTA ağlarında iletişimini sağlamak için blokzinciri kullanımı
[41]	Doğrulama	Sürdürülebilir IoT tabanlı uç hesaplama ağı tasarlama	Uç hesaplama ağı için akış kuralı bölümü ve tahsis algoritması
[42]	Doğrulama	5G hücreleri arasında sık geçiş kaynaklı performans sorunu	Blokzinciri ve YTA kullanarak geçiş sırasında yeniden kimlik doğrulama ihtiyacını azaltmak
[43]	Yönlendirme	Ağlar arası gecikme ölçümlerinin bütünlüğünün sağlaması	Gecikme ölçüm verileri periyodik olarak bir blokzinciri ağında yayınlanır ve saklanır
[44]	Yönlendirme	Minimum bağlantı çakışması ile alt akışların yönetilmesi	YTA tabanlı trafik mühendisliği ve konsorsiyum blokzincirinde oluşan bir çerçeve
[45]	Yönlendirme	IoT ağlarında veri iletimi için hizmet kalitesi iyileştirmesi	YTA ve blokzincirinin hizmet kalitesi sağlamak için oyun teorik bir yaklaşımla IoT için kullanılır
[46]	Yönlendirme	Hizmet kalitesinin korunması için bant genişliği yönetimi	Blokzinciri ve hizmet kalitesi yönetimini kolaylaştırmak için YTA kullanımı
[47]	Yönlendirme	Blokzinciri uzlaşma protokollerinin sonlandırılma sürenin azaltılması	Kaynak tahsisi için daha yüksek öncelikli kuyruklara uzlaşma ile ilgili akışları atamak
[48]	Yönlendirme	Hizmet kalitesi farkındalıklı servis kompozisyonu	Servis kompozisyonu için Ethereum tabanlı akıllı sözleşme algoritması
[49]	Yönlendirme	İçerik Dağıtım Ağları'nda servis konsepti	İçerik Dağıtım Ağları için blokzinciri destekli yeni bir servis düzenleme çerçevesi
[50]	Yönlendirme	Kablosuz Sensör Ağlarında yönlendirme güvenliği	Blokzinciri ve Pekiştirmeli Öğrenme'yi kullanarak güvenilir bir yönlendirme modeli
[51]	Yönlendirme	BGP yönlendirme sisteminin güvenliği	Clique uzlaşma protokolü ile yönlendirme bilgilerinin blokzincirinde saklanması
[52]	Yönlendirme	IoT ağlarında merkeziyetsiz yönlendirme yapılması	Merkezi olmayan blokzinciri tabanlı sözleşmeli bir yönlendirme protokolünün kullanılması
[53]	Güvenlik	Blokzincirinin YTA da bazı saldırılara karşı kullanılması	Akış kurallarına dayalı işlemleri içeren blokların kullanılması ve tüm anahtarlara gönderilmesi
[54]	Güvenlik	Birden fazla denetleyicinin aynı ağ anahtar setini yönetebilmesi	YTA anahtarının denetleyicilerin yanıtlarını blokzincirine işlem olarak yayınlaması
[55]	Güvenlik	İşbirlikçi Saldırı Tespit Ağlarını saldırılara karşı koruma	YTA yardımıyla güvenilir bir Snort imza veri tabanını dağıtmak için blokzinciri uygulanması
[56]	Güvenlik	Ağ-içi ve ağlar-arası DDoS saldırılarının azaltılması	DDoS saldırılarını azaltmak için Ethereum'un akıllı sözleşmelerinin kullanılması
[57]	Güvenlik	Özel veya konsorsiyum blokzincirlerinin güvenlik sorunları	YTA işlevlerini ile ağ trafiğini filtreleyerek blokzinciri için güvenlik duvarı oluşturmak
[58]	Güvenlik	Blokzinciri düğümlerini DNS saldırılarından koruma	Entropi hesaplama şeması ile izinsiz akışları engelleyen gerçek zamanlı bir tespit şeması

Tablo 4.2, YTA ve blokzincir teknolojilerinden faydalanan çalışmaların işbirliği amaçlarını, incelediği sorunları ve çözüm yaklaşımlarını özetlemektedir. Bu gelecek vaat eden teknolojilerin işbirliği, siber güvenlik ekosistemlerinde erişim kontrolü, onaylama/kimlik doğrulama, yönlendirme, güvenlik gibi çeşitli işlevlerde faydalanılmakta olup bunlar alt başlıklarda sunulmaktadır.

4.5.1. Erişim Kontrolü

Erişim kontrolü, neredeyse tüm uygulamaların gerekli bir güvenlik parçasıdır. Erişim kontrol sistemleri, bir sistemin kaynaklarına erişimi düzenlemek için uygulanır ve bilgisayar ve ağ güvenliğinin temel parçasıdır. Erişim denetimi, genellikle sistem politikalarına dayalı bir dizi yetkilendirmeye karşı uygulanır. Değiştirilemezlik, dayanıklılık, denetlenebilirlik ve güvenilirlik gibi blokzincirine özgü özellikler bu teknolojiyi erişim kontrol sistemleri için tamamlayıcı bir çözüm olarak düşünmeye götürmektedir. Blokzinciri kendisini erişim kontrol sistemleri için güvenilir bir alternatif altyapı haline getiren özelliklere sahiptir. Blokzincirinin dağıtık yapısı tek bir hata noktası (single point of failure) sorunu ve diğer merkezi yönetim sorunlarını çözer. Ayrıca, üçüncü tarafları ortadan kaldırarak onlara bağlı gizlilik sızıntıları konularında endişeleri giderir. Dahası, akıllı sözleşmeler kullanarak karmaşık koşullar altında erişim izinlerini izleyebilir ve uygulayabilir. Tüm bu özellikler araştırmacıları blokzincirini erişim kontrol sistemleri için bir altyapı olarak görmeye motive etmektedir.

Farklı gereksinimleri olan 5G hizmetleri (artırılmış gerçeklik, araçsal iletişim, yoğun nesnelerin interneti vb.), iletişim hizmeti sağlayıcılarının müşterilerinden kaynaklanan bir zorluk olan gelişmiş çok yönetimli alan hizmeti dağıtımları gerektirmektedir. Dilimler olarak sunulan ortak altyapı kaynaklarını paylaşan çeşitli uçtan uca hizmet türleri, müşterilerin çeviklik ve şeffaflık taleplerini karşılarken uygulama gereksinimlerini sürekli olarak ortaklık anlaşmalarına dönüştürmek için taşıyıcılara otomasyon zorlukları yaratır. Bu durum karmaşık bir dağıtık Hizmet Seviyesi Anlaşması - HSA (Service Level Agreement - SLA) tabanlı ağ kaynaklarının yönetilmesi noktasında harmoni oluşturma tehlikesine katkıda bulunur. [32] nolu çalışma, uzlaşma için akıllı sözleşmelere ihtiyaç duyan merkezi olmayan ve birbirlerine güven sorunu yaşayan eşler barındırmak için çoklu hizmetlerin yaşam döngüsü yönetimini

yerine getirmek amacı ile blokzinciri tabanlı dağıtık uygulamalara (Distributed Apps - DApp) dayanan ve paylaşımlı bir defter tarafından ortaya çıkan fırsatları yönetici etki alanı senaryoları için ortaya koymaktadır. Yazarlar, açık kaynak özellikli bir prototip uygulaması aracılığıyla, blokzinciri DApp'nin, yalnızca yetkili kuruluşların uzaktan yönetim Çoklu-Alan-Organizatörü'nde (Multi-Domain Orchestrator - MdO) belirli görevleri gerçekleştirmesine izin verilen bir kimlik yönetimi şeması altında akıllı sözleşmeler uygulama yeteneğini sergiliyor.

Kablosuz Sanallaştırma (Wireless Virtualization - Wi-Vi), fiziksel kablosuz altyapının ve Radyo Frekansı (RF) dilimlerinin, HSA'lara göre birden çok mantıksal/sanal kablosuz ağ operatörüne (Wireless Network Operators - WNOs) paylaşılmasını sağlayan bir teknoloji olarak kabul edilmektedir. Bilişsel radyo ağlarına en iyi alternatif olarak kabul edilen Wi-Vi, sadece RF spektrum kullanım verimliliğini, kablosuz ağ kapasitesini ve kapsama alanını değil, aynı zamanda kablosuz güvenliği de geliştirmeyi vaat etmektedir. [34] nolu eserde, kablosuz ağ sanallaştırma için yeni gelişen üç teknolojinin (YTA, Sınır Bilişim (Edge Computing - EC) ve Blokzinciri) füzyonunu içeren bir perspektif sağlamaktadır. YTA, denetleyicilerin yardımıyla, verimli ağ yönetimi için ağ kaynaklarının dinamik yapılandırılmasını mümkün kılmaktadır. EC, yalnızca ilgili baz istasyonlarındaki kullanıcı sinyallerini ve sorgularını mümkün olan en kısa gecikmeyle işlemeye yardımcı olmakla kalmıyor, aynı zamanda bir baz istasyonu ile merkezi denetleyici arasında yüksek hızlı bir ana taşıyıcı bağlantısı ihtiyacını ortadan kaldırmaya da yardımcı oluyor. Blokzinciri teknolojisi ise, kablosuz altyapı sahiplerini aynı kablosuz kaynağı (RF dilimini) birden çok sanal kablosuz ağa tahsis eden çift harcama saldırısından (double spending attack) korumakta. Önerilen yaklaşım, iş anlaşmazlıklarını azaltmayı ve kablosuz ağ endüstrisinde güveni ve şeffaflığı artırmayı amaçlamaktadır.

Şu anki hali ile YTA akış kuralı bütünlüğünü garanti etmek için gereken tam bir güvenlik mekanizmasından yoksundur. Bu durum, saldırganların kötü niyetli müdahale yoluyla çeşitli yıkıcı ağ saldırılarını tetiklemesine katkıda bulunur. Blokzinciri teknolojisinin değiştirilemezlik ve âdemi merkeziyetçilik özelliklerine dayanarak, [33] nolu eserde YTA akış kurallarını, düğüm kimliğini, ayrıcalık bilgilerini ve denetleyici genel bilgilerini kaydederek ağ bilgilerinin bütünlüğünü sağlayabilen blokzinciri tabanlı bir YTA güvenlik modeli önermektedir. Bu çalışma, geleneksel merkezi kimlik doğrulamasının tek nokta saldırılarına karşı savunmasız olma sorununu azaltan blokzinciri

ağı aracılığıyla dağıtık bir kimlik doğrulama mekanizması kurmaktadır. Ek olarak, depolamaya ve dağıtık kimlik doğrulamaya dayalı olarak rol tabanlı hak yönetimi gerçekleştirir.

Erişim kontrolü, bilgilere kimin erişebileceğini belirtmek için izinler atamayı amaçlamaktadır. Toplumumuzda vatandaşlar, kişisel verilerinin nasıl kullanıldığı hakkında çok az bilgiye sahiptir. Fakat vatandaşlar kişisel verilerinin nerede saklandığını ve verilerine kimlerin erişebileceğini kontrol etmeye iradesini de göstermektedirler. Özellikle şehirlerin kalabalıklaşması ile birlikte, merkezi bir erişim kontrol sunucusuna dayanarak tüm vatandaşların kişisel verilerinin erişim kontrolünü yönetmek zor hale gelmektedir. Bazı araştırmacılar, blokzinciri teknolojisini kullanarak kişisel veri erişim kontrolünü geliştirmeye çalışmaktadırlar. Bu amaçla paralel olarak, [35] nolu çalışmada blokzinciri tabanlı merkezi olmayan bir kişisel veri erişim kontrol sistemi önerilmektedir. Bu sistemde üç varlık bulunmaktadır: kullanıcılar, hizmetler ve düğümler. Benzer şekilde [36] ve [37] nolu çalışmalarda, merkezi olmayan kullanıcı merkezli bir erişim kontrol modeli önerilmektedir. Model üç ana bileşenden oluşur: Veri yönetimi protokolü, mesajlaşma servisi ve veri depolama sistemi. Veri depolama sisteminde, veri talep edenlerin veri sahiplerinin kişisel verilerine erişmelerine izin verilip verilmediğini bilebilecekleri erişim kontrol verilerini depolamak için blokzinciri teknolojisi kullanılır.

4.5.2. Doğrulama

Küçük ve büyük kuruluşlar, dijital altyapılarını iş uygulamaları için yenilikçi ve dinamik ortamlar sağlayan bulut ve sınır altyapılarına taşımaktadırlar. Bu teknolojiler, çeşitli akıllı şehir uygulamaları için uygun altyapı sağlayarak bu uygulamaların bilgi işlem ve depolama talebini karşılamak için dinamik yapılandırılabilirlik sağlamaktadırlar. Örneğin, akıllı şehirlerde IoT, sensörler, akıllı telefonlar, RFID'ler gibi farklı kaynaklardan oluşturulan büyük veriler kendi kendine doğrulama ve toplanan verilere güven tesis etmek için özerk olarak doğrulanabilir kimlikler gerektirmektedir. Bu bağlamda işletmeler, telekom sektörü, kamu ve özel kuruluşlar, alana özgü hizmetleri, uygulamaları ve verileri güvenli bir şekilde yönetmek için temel iletişim altyapısında yeni çerçeveler, protokoller ve mekanizmalar tanımlamaktadırlar. [38]'de yazarlar, blokzinciri tekniğine dayalı doğrulanabilir kimlikler tasarlamak, bu kimlikleri dağıtmak ve yeni doğrulama protokolleri kullanarak bunları onaylamak için

VeidBlock adlı yeni bir yaklaşım sunmaktadırlar. Yazarlar, çözümlerini Ağ Fonksiyon Sanallaştırma konseptine dayalı dağıtık bir YTA altyapısında uygulamaktadırlar. Çözümün farklı akıllı şehirlerde, IoT çözümlerinde ve düşük gecikmeyle ilgili uygulamalarda kullanılabileceğini iddia edilmektedir.

Botnet, kötü amaçlı yazılım bulaşmış IoT cihazları veya botlar tarafından kontrol edilen virüslü cihazlardan oluşan bir bilgisayar ağıdır. Genellikle, bir botnet komut ve kontrol sunucuları aracılığıyla bir bot yöneticisi tarafından kontrol edilir. Ağlarda Dağıtılmış Hizmet Reddi (DDoS) saldırıları başlatan botnet sorunu, esas olarak bu ağlar üzerinden dağıtılan güvensiz IoT cihazlarının sayısındaki hızlı artıştan kaynaklanmaktadır. [39] nolu çalışmada IoT cihazlarının güvenliğini sağlamak için merkezi olmayan blokzinciri tabanlı bir mimari önerilmektedir. Önerilen mimari, tamamen dağıtık bir şekilde IoT için bir botnet önleme sistemini yönetmek adına blokzinciri, akıllı sözleşmeler ve YTA konseptlerini kullanmaktadır. Amaç, IoT cihazlarının diğer ağlara ve varlıklara DDoS saldırıları başlatmak için botnet ağının bir parçası olmasını sağlayacak uygulamaları çalıştırmasına izin vermeyerek otomatik ve yönetimi kolay bir önleme sistemi oluşturmaktır. Bu çalışmada YTA'nın amacı ağ işlevlerini otomatize etmek ve veri düzlemini ağlara programlanabilirlik ve otomasyon uygulayan kontrol düzleminden ayırarak iş çevikliğini sağlamaktır. YTA denetleyicisi, güvenlik politikalarını ve hizmetlerini dinamik bir şekilde özelleştirmek için etkili bir yaklaşım ortaya koyduğundan, çalışmada güvenlik politikalarını uygulamak ve botnet oluşumlarının önlenmesi adına şüpheli trafiğin izlenmesi için akıllı sözleşmelerde akış kuralları uygulanmaktadır.

IoT, iletişimden finansal işlemlere, ulusal güvenliğe kadar toplumumuzun birçok alanında yer almaya başlayan bir teknolojidir. Bu teknolojide güvenlik, karşılaştırılabilirlik, enerji tüketimi ve cihazların heterojenliği gibi uzun süredir devam eden zorluklar mevcuttur. Bir IoT ağına bağlı cihazların sınırlı enerji ve bilgi işlem kaynakları nedeniyle, IoT ve uç ağlar arasında veri aktarımında güvenlik ve enerji konuları önemli roller oynamaktadır. İster kötü amaçlı ister kazara olsun, bir IoT ağındaki verilere müdahale, potansiyel olarak ciddi sonuçlar ortaya koyabilir. [40] nolu eserde yazarlar, IoT'nin karşılaştığı bazı zorlukları hafifletmek için blokzinciri ve YTA'yı IoT bünyesinde entegre etme potansiyelini araştırıyorlar. Yazarlar, IoT ağlarındaki YTA denetleyicileri için yeni bir blokzinciri tabanlı mimari önermektedirler. Önerilen mimari, blokzinciri katmanı ile güvenilir bir aracı olmadan birbirlerine güven duymayan tarafların her bir kümede (YTA alanı) birbirleriyle doğrulanabilir

bir şekilde etkileşime girebildiği, dağıtık bir eşler arası ağa izin vermektedir. Bu durum, her YTA alanındaki IoT cihazları arasında güvenli etkileşimi kolaylaştırmaktadır. IoT cihazlarının hızlı büyümesi ve birçok yeni IoT uygulamasının ortaya çıkışı nedeniyle veri trafiği hacmi katlanarak artmaktadır. IoT cihazlarından üretilen büyük veri hacmi, sınırlı bant genişliği, yüksek gecikme süresi ve gerçek zamanlı analiz gereksinimleri nedeniyle, geleneksel merkezi ağ mimarisi kullanıcılarının gereksinimlerini karşılayamaz duruma gelmiştir. Farklı sensör türlerinin yayılımı nedeniyle yoğun gerçek zamanlı veri analizi mevcut son teknoloji merkezi mimarilerdeki en büyük zorluklardan biridir. Mevcut zorlukları ele almak ve mimari tasarım ilkelerine bağlı kalmak adına [41] nolu eserde yazarlar, sürdürülebilir bir sınır hesaplama ağı için bir blokzinciri tasarımı ile YTA tabanlı dağıtık katmanlı bir ağ mimarisi olan SoftEdgeNet modelini önermektedirler. SoftEdgeNet modeli sis katmanında, güvenlik saldırılarını azaltmak ve gerçek zamanlı analitik hizmetleri sağlamak için YTA tabanlı güvenli bir sis düğümü mimarisi sunulmaktadır. Ayrıca, yazarlar ağ sınırında bir akış kuralı bölümlenmesi ve tahsis algoritmasını da önermektedirler.

Yoğun heterojen 5G hücreleri arasında sık geçiş (handover) nedeniyle yaşanan performans düşüşünü en aza indirmek için blokzinciri ve YTA kullanılarak geliştirilmiş bir kimlik doğrulama yaklaşımı [42] nolu eserde önerilmiştir. Önerilen kimlik doğrulama mekanizması, geçiş sırasında yeniden kimlik doğrulama ihtiyacını en aza indirerek daha düşük gecikme ve hizmet kesintisi sağlamaktadır. Bunu başarmak için, blokzinciri tarafından kontrol edilen dağıtık bir YTA denetleyici mimarisi önerilmektedir. Blokzinciri, kullanıcıların ilk ekleme işlemi sırasında kimlik doğrulama için kullanılan genel ve özel anahtarlarını depolar. Kimlik doğrulama yapıldığında, blokzinciri bilgileri bitişik hücrelere dağıtır. Ayrıca, blokzinciri ağdaki her düğümün (YTA denetleyicisi, baz istasyonları) kimliğini doğrular. Bilginin güvenli bir şekilde paylaşılması kimlik doğrulama ihtiyacını en aza indirdiği için ağdaki geçiş süreleri azalmaktadır.

4.5.3. Yönlendirme

Sınır Geçit Protokolü (Border Gateway Protocol - BGP) mevcut interneti 20 yıldan fazla bir süredir yönlendirmektedir. BGP'nin yönlendirme karar algoritması, en iyi yol hesaplamaları için ana faktör olarak otonom sistemler (Autonomous Systems - ASes) arasındaki yol uzunluğunu (AS-length) kullanmaktadır. Otonom sistemler arasındaki eşleşme mimarilerinin maliyetlerinin giderek daha

uygun hale gelmesi nedeniyle, yol uzunluğu genel olarak dünya çapındaki internet ağında azalmakta ve bu da gerçek zamanlı internet protokol trafiği için daha az verimli yollara neden olmaktadır. Geleneksel BGP tabanlı internetteki en iyi yol seçme algoritmaları, gecikme gibi gerçek zamanlı tıkanıklık ölçümlerini işleyişlerinde dâhil etmezler. Fakat gecikme ölçümü ile ilgili zorluklardan birisi gecikme verilerinin birden çok otonom sistem arasında yeniden duyuru olarak verilmesinden dolayı geçerlilikleri ve güvenilirlikleridir. Bu çerçevede, [43] nolu kaynakta birden çok otonom sistemden oluşan ve YTA'da gecikmeye duyarlı yönlendirme sağlayan bir internet mimarisi önermek adına dağıtık bir blokzincirinde ağ alanları arası (inter-domain) gidiş-dönüş süreleri (Roud-Trip Time - RTT) ölçümlerinin kullanılması incelemektedir. Önerilen mimari gecikme ölçüm bütünlüğünü sağlamak için her otonom sistemin bir blokzinciri düğümünü barındırabildiği blokzincirindeki çeşitli otonom sistemler arasında ölçülen RTT'yi periyodik olarak depolamaktadır. RTT değerleri blokzincirinde depolandıktan sonra, sistem anormallikler için iki taraflı veri bütünlüğünü onaylar ve otonom sistemler arası RTT sonuçlarını gecikmeye duyarlı yönlendirme kararı için YTA denetleyicisine bildirir.

Konsorsiyum blokzincirleri birçok insanın bir araya gelerek sosyal ağ oluşturulmasında da kullanılabilmektedir. Bu tür bir yapıda kurulan sosyal ağlar da içerikler ağa katılan üyelerden veri parçaları (alt akışlar) şeklinde toplanabilmektedir. Bu ilginç mekanizma, sosyal ağ üzerinden kullanıcıya alt akışları aktarmak için büyük ölçüde çok yollu veri aktarımına dayanır. Bununla birlikte, konsorsiyum blokzincirlerini destekleyen çok yollu veri iletimi açısından bazı yeni sorunlar da ortaya çıkmaktadır. Bir yandan trafik sıkışıklığı oluşmasını azaltmak adına aynı içerikten gelen veri parçaları (alt akışlar) minimum örtüşen bağlantı yolları boyunca aktarılmalı iken diğer taraftan bu iletim yolları genellikle farklı seviyelerde yayılma gecikmesine sahiptir. Bu durumda da kullanıcı tarafından alınan düzensiz alt akışlara neden olmaktadır. Bu nedenle, minimum bağlantı örtüşmesi ve iletim yolları arasında performans farklılaşması ile alt akışları verimli bir şekilde yönetmek gerekmektedir. Bu sorunları çözmek için [44] nolu eserde yazarlar, YTA tarafından sağlanan küresel trafik mühendisliği daha esnek akış aktarımı gerçekleştirebildiği için YTA tabanlı bir çerçeve tasarlamaktadırlar. Kullanılan YTA denetleyicileri, topoloji keşfi, alt akış yöneticisi ve yol seçimi olmak üzere üç modüle sahiptir. Sosyal ağı minimum bağlantı çakışması ve performans farklılığına sahip bir dizi yolun bulunabileceği en uygun konsorsiyum blokzincirinden oluşturmaktadırlar.

Son yıllarda mevcut internet trafiği arasında ki multimedya içerikli veri trafiği oranı giderek artmaktadır. İletilen multimedya odaklı önemli miktardaki veri, Ağ Servis Sağlayıcılarının (Network Service Providers - NSPs) ve İçerik Sağlayıcıların (Content Providers - CPs) son kullanıcılara daha kaliteli video hizmeti sunmalarını zorlaştırmaktadır. Bu durum son kullanıcı tarafından talep edilen hizmetin garanti altına alınmasını önemli bir husus haline gelmiştir. Bu amaç doğrultusunda [45] nolu çalışmada, ağdaki bağlantı tıkanıklığı ile ilgilenerek son kullanıcılara daha yüksek hizmet kalitesi sunan önerilere yer verilmiştir. Çalışmada, Stackelberg oyun teorisi ile birlikte dinamik bir bağlantı fiyatlandırma modeli önerilerek ağ denetleyicisinin tek nokta arızalarına karşı güvenlik zorluklarıyla başa çıkmak için blokzinciri kullanılmaktadır.

Genel olarak, bir İSS aynı bölgedeki birden çok müşteriyle statik bir bant genişliği kaynak havuzunu paylaşır. Aynı kaynağa erişen müşterilerin sayısı arttıkça kullanıcıların internet hızı da dalgalanmaktadır. Bu durum yoğun saatlerde sorun haline gelmektedir. Diğer taraftan ise İSS'ler tarafından müşterilerin sınırlı bant genişliğini eşzamanlı olarak kullanması beklenir ki bu da genel kullanıcı deneyiminin kalitesini düşürmektedir. Böylece, müşteriler İSS'nin pazarladığından daha düşük internet hızı sağlayarak İSS'nin HSA'yı ihlal ettiğini iddia edebilirler. Bu durum İSS'ler için bir ikilem doğurmaktadır: Bant genişliği kaynak havuzunu artırmak sorunu çözebilirken, ek bant genişliği yoğun saatler sona erdiğinde işe yaramaz hale gelmektedir. Bu nedenle, aşırı tahsis (over-allocation) sorunu gündeme gelmektedir. [46] nolu kaynakta yazarlar, İSS için çevik ağ bant genişliği yönetimi sağlamak adına blokzinciri ve YTA kullanımını önermektedirler. Ayrıca, bir İSS'nin bant genişliği kullanım verimliliğini en üst düzeye çıkarırken gelir üretme yeteneğini azaltmadan müşterilerin bant genişliği tüketimlerini ihtiyaçlarına göre en iyi şeklide kontrol etmelerine izin verecek dinamik ve adil bant genişliği tahsisini mümkün kılmanın fizibilitesini sunmuşlardır. Bunun için, İSS ile müşteriler arasındaki üç kullanım durumu (Talep Üzerine Bant Genişliği, Satış Bant Genişliği ve Gerçek Zamanlı Fiyatlandırma) dikkate alınarak blokzincir ve YTA'dan yararlanmışlardır.

Oylamaya dayalı blokzinciri uzlaşma mekanizmaları, ağ düğümleri arasında iletişim kaynaklarının fazladan kullanımını gerektirir. Bir liderin seçilmesi veya ağ verilerinin durumunun güncellenmesi gibi görevleri gerçekleştirmek için sık sık mesaj alışverişi yapmak gerekir. Bu sık mesaj alışverişi ihtiyacı, bu uzlaşma mekanizmalarının altta yatan ağın durumuna duyarlılığını artırır.

Oylamaya dayalı bir uzlaşma mekanizmasının sona erdirilmesi tüm uzlaşma katılımcılarının ağ durumuna ilişkin aynı ve en son görüşü elde etmesi demektedir. Tıkanıklık veya bağlantı darboğazları gibi ağ sorunları, uzlaşma yakınsamasının bozulmasına ve sonlandırılmasına zarar verilebilir. [47] nolu raporda yazarlar, oylamaya dayalı uzlaşma mekanizmalarının güvenilir bir şekilde çalışmasını sağlamak için hizmet kalitesi tekniklerinin uygulamasını vurgulamaktadırlar. Önerilen strateji sonlandırma süresini azaltmak ve uzlaşma mekanizmalarının doğru çalışmasını sağlamak için YTA'ya bağlıdır. Farklı hizmet kalitesi seviyesinde kuyruklar oluşturarak uzlaşma için ayrılmış hizmet kalitesi kuyruğuna minimum bant genişliği rezervasyonu sağlıyorlar. YTA paradigması, dağıtık veri düzlemini mantıksal olarak merkezileştirilmiş kontrol düzleminden ayrıldığı için ağ politika tanımlamaları, kontrol düzlemi ile sınırlandırılır ve ardından kuyruklar ve bant genişliği limitleri gibi mevcut kaynaklara göre veri düzleminde kopyalanırlar.

Blokzinciri teknolojisini önerdikleri yönlendirme veya hizmet kalitesi temelli sistemlerde ele alan başka çalışmalarda literatürde mevcuttur. Blokzinciri teknolojisi yönlendirme dinamikleri göz önüne alındığında servis düzenleme (service orchestration) konseptlerine de fayda sağlayabilir. [48] nolu çalışmada yazarlar, Ethereum platformunda çalışan hizmet kalitesi farkındalıklı servis kompozisyonu için akıllı sözleşmeye dayalı bir algoritma öneriyorlar. Önerilen çalışmada, servis istemcisi istediği hizmet kalitesi kısıtlamalarını (yanıt süresi vb.) ve bütçesini karşılayan servis sağlayıcılar tarafından sağlanan servisleri, istediği sistemi oluşturabilmek için birleştirmektedir. Yine, İçerik Dağıtım Ağları (Content Delivery Networks) için blokzinciri destekli yeni bir servis düzenleme çerçevesi [49] ile Kablosuz Sensör Ağlarında (WSN) yönlendirme güvenliğini ve verimliliğini artırmak için blokzinciri ve pekiştirmeli öğrenmeyi (Reinforcement Learning) kullanarak güvenilir bir yönlendirme planı [50] önerilmektedir. Uygulanabilir olan bir yönlendirme şeması, blokzincirindeki yönlendirme düğümlerinin yönlendirme bilgisini elde etmek için verilirken Pekiştirmeli Öğrenme modeli yönlendirme düğümlerinin dinamik olarak daha güvenilir ve verimli yönlendirme bağlantıları seçmesine yardımcı olmak için kullanılmaktadır. [51] nolu kaynakta yazarlar, BGP korsan (hijacking) saldırılarını önleyerek ve internet yönlendirme yollarının uyumlu ve tutarlı bir görünümünü koruyarak BGP yönlendirme sistemini güvenceye almak için blokzinciri tabanlı bir çerçeve olan RouteChain'i sunmaktadırlar. Araştırmacılar, bir BGP duyurusunu otonom sistemler arasında değiş tokuş edilecek bir işlem

(transaction) olarak ele almışlar ancak, hizmet istekleri için hizmet kalitesi odaklı uçtan uca bir yol oluşturma problemini göz önünde bulundurmamışlardır. Otonom sistemler arasında uzlaşma oluşturmak için Clique adlı Otorite Kanıtı (Proof-of-Authority - PoA) tabanlı bir uzlaşma protokolünü kullanarak yönlendirme bilgilerini karşılıklı olarak paylaşılan ve herkesin erişebildiği ortak bir defterde güvenli bir şekilde kayıt altına almaktadırlar. Önerilen modelin etkililiği, bir saldırının başlatılmasından önce otonom sistemler arasında yönlendirme bilgileri için bir uzlaşma anlaşması olmasına dayanmaktadır. [52] nolu çalışmada, merkezi olmayan blokzinciri tabanlı sözleşmeli bir yönlendirme protokolünü (BCR) tanıtmaktadır. BCR protokolü, farklı üreticilerin IoT cihazlarının birbirlerine güvenmelerini ve veri iletişimi sırasında işbirliği yapmalarını sağlamaktadır. Bu yaklaşımı kullanarak, gecikmeye-toleranslı IoT ağındaki cihazlar, bir ağ geçidine veya hedef cihaza merkezi olmayan bir şekilde yol bulabilmektedir.

4.5.4. Güvenlik

Blokzincirlerinin gelecek vaat eden uygulamaları arasında ağ izleme ve kimlik doğrulama, gizlilik, mahremiyet, bütünlük gibi güvenlik hizmetleri bulunmaktadır. Şu anda, bu hizmetler güvenilir üçüncü taraf aracılar tarafından veya verimsiz dağıtık yaklaşımlar kullanılarak sağlanmaktadır. Sonuç olarak, güvenlik, mevcut uygulamalar için büyük bir problemdir. Öte yandan, blokzinciri teknolojisi, tamamen dağıtık ve kanıtlanabilir şekilde güvenli bir uzlaşma çözümü sağlamanın yanı sıra birçok geleneksel zorluğu çözen güvenlik garantileri de sağlayabilir. Blokzinciri teknolojisinin tam tersi ise, ağ yönetimi ve yapılandırmasının bir yazılım varlığında yani denetleyicide merkezileştirildiği YTA'dır. YTA mimarisi, yeni saldırı vektörlerini ortaya çıkarırken veri düzleminin kontrol düzleminden ayrılması yoluyla geleneksel ağlarda var olan birkaç güvenlik sorununu çözmektedir [59]. Blokzinciri teknolojisinin uygulanmasına benzer bir yöntem şeklinde bazı ağ işlevlerinin dağıtılması, YTA mimarisindeki güvenlik açıklarını artırabilir. YTA, bazı tehditlere karşı daha güvenli olmakla birlikte ölçeklenebilir ve genişletilebilir ağ izleme araçları sağlarken, geleneksel ağlarda bulunmayan yeni güvenlik açıkları da sunmaktadır. Tüm ağ yönetimi ve yapılandırmasının merkezi bir YTA denetleyicisinde toplanması, tek bir hata noktası olarak kabul edildiğinden bazı YTA güvenlik hizmetlerinin dağıtılması ve ağ varlıkları arasında blokzincirinde

olduğu gibi merkezi bir varlığa ihtiyaç duyulmadan karşılıklı güven oluşturulması, YTA mimarisinin sağlamlığını önemli ölçüde iyileştirir, güvenliğini artırır ve iletişimini büyük oranda siber olaylara karşı korur.

YTA mimarisinde içerisinde bir saldırgan ağın tam kontrolünü elde etmek adına veri düzlemindeki akış tablosu girdilerini güncelleme veya kontrol düzlemi operasyonlarını engelleme şeklinde farklı saldırı türleri (gizli dinleme, ortadaki adam saldırısı ve DoS vb.) kullanarak denetleyicinin güvenliğini ihlal edebilir. Bu nedenle, yukarıda belirtilen saldırıların üstesinden gelmek ve YTA tabanlı ağ mimarisinin güvenliğini sağlamak için yeni stratejiler ve çözümler gereklidir. [53] nolu eserde çeşitli güvenlik sorunları ve farklı saldırı vektörleri olası çözümlerle birlikte tartışılmıştır. Bu saldırıları azaltmak için, YTA'da bir hizmet çerçevesi olan blokzinciri tabanlı BlockSDN modeli önerilmiştir. Çalışmada, izinli blokzinciri mimarisi ve BlockSDN'nin uygulanabilirliğini göstermek için iki saldırı senaryosu sunulmaktadır; i) veri düzleminde yazılım güvenliği ihlal edilmiş kötü amaçlı bir anahtar ve ii) kontrol düzleminde dağıtık hizmet reddi saldırısı. Benzer şekilde [54], birden fazla denetleyicinin aynı anahtar setini yönetmesine izin vererek anahtarlar ve denetleyiciler arasında dinamik bir ilişki kurmak için blokzinciri teknolojisini kullanmaktadırlar. Dahası, blokzinciri üzerindeki veri bloklarına dayalı tutarlı ve çoklu denetleyici işbirliğini sağlayarak ve ortak paylaşılan bir veri tabanı oluşturarak, YTA denetleyicileri arasındaki tutarlılığı artırmaya yardımcı olmaktadır. Ayrıca, çalışmada blokzinciri hatalı davranan denetleyicileri tespit etmek ve önerilen sistemdeki anahtarlar ile denetleyiciler arasındaki güvenilir ilişkiyi sağlamak için kullanılmaktadır. Önerilen sistem, anahtarları ve denetleyicileri blokzincirinde ki düğümler olarak kabul etmekte ve YTA anahtarlarının paket bildirim mesajlarını (PacketIn) tüm denetleyicilere (düğümlere) yayınlamasına izin vererek blokzinciri teknolojisinden yararlanır.

Siber saldırıların hızla artması nedeniyle, saldırı tespit sistemleri (Intrusion Detection Systems - IDS) işbirlikçi yaklaşımlara doğru kaymaktadır. Büyük ağ ortamlarının tehditlere karşı güvenliğini sağlamak için giderek büyüyen bir talep oluşmaktadır. Tespit performansını optimize etmek için bir grup IDS düğümünün zorunlu bilgileri (IDS imzaları, saldırı alarmları vb.) karşılıklı olarak paylaşmasına ve değiş tokuş etmesine olanak tanıyan, pratik senaryolarda İşbirlikçi Saldırı Tespit Ağları (Collaborative Intrusion Detection Networks - CIDNs) yaklaşımları son zamanlarda benimsenmiştir. Fakat CIDN'ler doğası gereği dağıtık olduğu için bu tür ağlar hala birçok uygulama sorunuyla karşı karşıyadır-

lar. Bilhassa, saldırganlar içeriden herhangi bir güvenlik düğümüne kolayca hâkim olabilir ve tüm güvenlik sistemini savunmasız bırakabilirler. IDS düğümleri arasında güvene dayalı iletişim sağlamak için blokzinciri uygulamalarındaki son gelişmeler, CIDN ağlarında güvene dayalı iletişim oluşturmak adına uygun bir aday olarak kabul edilir. [55] nolu çalışma, CIDN ağını ve blokzincirini YTA bağlamında birleştirmektedir. Yazarlar, bir YTA denetleyicisi olan Ryu'dan en son imza güncellemesini almak ve daha sonra bu tür imza güncellemelerini test yatağındaki diğer tüm Snort düğümleriyle güvenli bir şekilde paylaşmak için ortaklaşa çalışan üç Snort IDS'i kullanmaktadırlar.

Güvenli olmayan cihazların sayısının katlanarak artmasıyla birlikte DDoS saldırılarının etkisi hızla artmaktadır. Mevcut DDoS saldırılarını azaltma planları, düşük esneklik, kaynak eksikliği ve yüksek maliyet nedeniyle sorunlar yaşamaktadır. Blokzinciri gibi yeni ortaya çıkan teknolojiler, birden çok ağ alanında DDoS saldırılarının azaltılması için düşük maliyetli, verimli ve esnek yeni fırsatlar sunmaktadır. [56] nolu eserde yazarlar, iki saldırı azaltma düzeyini (ağ-içi ve ağlar-arası DDoS azaltma) birleştiren Cochain-SC adlı blokzinciri tabanlı bir yaklaşım önermektedirler. Ağ-içi için yazarlar, etkili bir DDoS saldırısı azaltma yöntemini YTA bağlamında önermektedirler. Bu yöntem üç şemadan oluşmaktadır:

1) sFlow kullanarak ağ alanı içindeki verilerin rastgeleliğini ölçmek için Entropi tabanlı bir şema (I-ES);
2) Hatalı akışları entropi değerlerine göre sınıflandırmak için Intra-Bayes tabanlı bir şema (I-BS);
3) Ağ alanı içindeki hatalı akışları azaltmak için ağ-içi azaltma (I-DM) planıdır.

Ağlar-arası için ise blokzinciri işbirliğine dayalı bir DDoS azaltma planı önerilmektedir. Araştırmacılar çalışmada DDoS saldırılarını azaltmak ve YTA tabanlı ağlar arasındaki işbirliğini kolaylaştırmak için Ethereum tabanlı akıllı sözleşmeleri kullanmaktadırlar. Bu amaçla, yazarlar, birden fazla YTA tabanlı ağın güvenli bir biçimde işbirliği yapmasına ve merkezi olmayan bir şekilde saldırı bilgilerini aktarmasına izin veren yeni ve güvenli bir plan tasarlamaktadırlar.

YTA ağ kontrolünü veri düzleminden ayırarak ağ yapılandırması için çok fazla küresel görünürlük ve esneklik sağlayabilmektedir. Fakat YTA hala birçok sorun ile karşı karşıyadır. Örneğin, merkezi denetleyici tek bir hata noktası

haline gelebilirken, dağıtık denetleyiciler içeriden gelen saldırılara karşı savunmasız olabilirler. Blokzinciri teknolojisi bilinmeyen varlıkları güvenilir bir üçüncü taraf olmadan birbirleriyle iletişim kurma olanağı sağladığından, araştırma toplulukları YTA'yı blokzincirleri ile birleştirme olanaklarını araştırmaya çoktan başladı. Ayrıca, YTA, blokzinciri uygulamalarındaki birçok güvenlik sorununu ele almak için bir çözüm olarak incelenmektedir. Örneğin, Steichen ve arkadaşları [57], bir saldırganın belirli düğümlerde blokzinciri işlemini durdurabileceğini belirlemiş ve blokzincirleri için ağ trafiğini filtreleyebilecek bir mekanizma olan ChainGuard'ı önermiştirler. Benzer şeklide, El Houda ve arkadaşları [58] blokzinciri düğümlerini DNS amplifikasyon saldırılarından korumaya yardımcı olabilecek ChainSecure adlı YTA tabanlı bir çözüm sunmaktadırlar. Ayrıca, [44] nolu eserde yazarlar, konsorsiyum blokzincirleri için akışları daha esnek bir şekilde aktarmak maksadı ile YTA'nın nasıl kullanılabileceğini açıklamaktadırlar.

4.6. YAZILIM TANIMLI AĞ ÇÖZÜMLERİNDE KULLANILAN BLOKZİNCİRİ TEKNOLOJİSİ ÖZELLİKLERİ

Blokzinciri, prensip olarak herhangi bir veri odaklı disipline uygulanabilir. Fakat, herhangi bir disipline uygulamadan önce blokzincirinin değişmezlik, âdemi merkeziyetçilik, şeffaflık, güvenlik ve gizlilik gibi potansiyel faydalarını anlamak ve değerlendirmek o uygulamadaki başarıyı artırmak için önem arz etmektedir. Bu bölüm, siber güvenlik ve iletişim ağları ile ilgili kavramlar ve uygulamalar bağlamında bu tür temel özellikleri gözden geçirecektir.

4.6.1. Değiştirilemezlik

Blokzinciri ağında işlemler doğrulandıktan sonra bir zaman damgası alır ve bir özet algoritması tarafından kriptografik olarak korunan bir bloğa eklenir. Her blok, kendinden önceki bloğa o bloğun kriptografik özetini içererek bağlanır. Bu mekanizma birden çok bloğu bağlar ve kronolojik bir zincir oluşturur. Özellikle, yeni bir bloğun özet işlemi her zaman bir önceki bloğun özet değerinin meta verilerini içerir ki bu da zincir verilerini büyük ölçüde değiştirilemez kılar. Blokzincirinin bu değişmezlik özelliği, Yazılım Tanımlı Optik Ağlarda (YTOA) [60] korumalı spektrum paylaşımı, YTA'lar da cihazdan cihaza bağlantı ve/veya gizlilik korumalı ağ görselleştirme gibi farklı araştırma alanlarında verile-

rin güvenli bir şekilde depolanmasını ve alışverişini kolaylaştırır. Ayrıca, blokzincirleri değiştirilemez işlem defterlerini kullanarak, büyük ölçekli güvensiz ortamlarda uyuşmazlık olması durumunda, hizmet kalitesi ile ilgili verilerin bütünlüğünü sağlamak ve otonom sistemler arasındaki anlaşmazlıkları uzlaştırmak noktasında da faydalı olabilir. Ağ hizmeti sağlayıcıları Yazılım Tanımlı Kablosuz Ağlar (YTKA) ve Yazılım Tanımlı IoT Ağları (YTIoTA) gibi çeşitli ağ mimarilerindeki işlem defterlerini kullanarak, büyük ölçekli IoT işbirliğine dayalı projeleri veya güvenilir IoT ortamlarında mobil sınır/bulut bilişim gibi heterojen ağ iletişimi ve bilgi işlem uygulamaları için güvenli iletişimler de kurabilirler [45]. Blokzincirinin bu tür kullanımları farklı uygulama durumlarını gösterirken hem akademiden hem de endüstriden uygulayıcılar ve araştırmacılar için yeni araştırma alanları açabilir.

4.6.2. Merkeziyetsizleştirme

Blokzinciri dağıtık bir ortam sağlamasından dolayı işlemleri yönetmek için merkezi bir varlığa bağımlı değildir. Blokzinciri bir ağdaki kullanıcılar arasında işlemleri onaylamak için merkezi bir yapıya veya harici bir tarafa güvenmek yerine uzlaşma protokollerini kullanmaktadır. Bu özellik, merkezi güç kesintisi nedeniyle tek nokta arıza riskini azaltmak, işletim maliyetlerini düşürmek ve güvenilirliği artırmak gibi önemli avantajlar sunmaktadır.

YTA'da, Aracı (Broker) konsepti [61], Pathlets yöntemi [62] ve SDN Exchange Point (SDX) projesi [63] gibi modeller hizmet kalitesi tabanlı ağlar arası yönlendirme uygulamaları için önerilmiştir. Fakat bu öneriler merkezi yapıları ve özel bilgileri üçüncü şahıslarla paylaşma gerekliliği nedeniyle ağ yöneticilerinde güvenlik ve mahremiyet konusunda tereddütler ortaya çıkarmaktadırlar [17]. Hizmet kalitesi hedeflerini gerçekleştirmek için tamamen veya kısmen harici bir varlıktan yararlanmaktadırlar. Bu dış kaynak temelli metodolojiler ağlar için belirli düzeylere kadar çalışabilse de servislerden tam olarak yararlanmak için özel bilgileri paylaşma zorunluluğu nedeniyle de güvenlik sorunları oluştururlar. Buna ek olarak, bu yaklaşımlar ağları merkezi yapıları nedeniyle ağlar arası yönlendirme için yukarıda belirtilen dışsal birimlere bağlı hale getirir. Bu nedenle, ağ yöneticilerinin ağlar arası yönlendirme için hizmet kalitesi sağlarken bu tür merkezi yapıları ortadan kaldırma ihtiyacı ve talebi olmaktadır. Blokzinciri, ağlar arası seviyede ağ operatörlerinin hizmet kalitesi tabanlı yönlendirme deneyimlerini geliştirmek

için fırsatlar sunabilir. Uzlaşma protokollerini kullanarak işlemleri yönetmek için dağıtık bir ortam sağlar. Blokzincirinin âdemi merkeziyetçilik özelliği, ağların yukarıda belirtilen Broker kavramı, Pathlets yöntemi ve SDX projeleri gibi merkezi yapılarda özel bilgileri üçüncü şahıslarla paylaşma zorunluluğundan kaynaklanan güvenlik ve gizlilikle ilgili olarak belirtilen tereddütlerinin üstesinden gelmelerine yardımcı olabilir. Ek olarak, YTA ağlarında harici bir varlığa bağımlı olmadan dağıtık bir şekilde ağlar arası yönlendirme için hizmet kalitesi sağlanmasında yardımcı olabilir.

4.6.3. Şeffaflık

İzinsiz (permissionless) bir blokzincirinin şeffaflığı, işlemlerin doğrulama için blokzinciri ağı üzerinden yayılması neticesinde blokzinciri defterlerindeki tüm verilerin ağdaki tüm katılımcılar tarafından görülebilir olmasından kaynaklanır. Bu nedenle, blokzincirindeki tüm katılımcılar, blokzincirinin şeffaflık özelliği sayesinde ağ üzerinden işlem etkinliklerini kolayca elde edebilir, onaylayabilir ve izleyebilir. Bu şeffaflık, onaylanmamış veri değişikliği riskini azaltarak blokzinciri tabanlı sistemlerin bütünlüğünün korunmasına da katkıda bulunur. Bu tür bir işlevsellik, özellikle şeffaflık ve eşitlik gerektiren ağ oluşturma ekosistemleri için uygundur. Örneğin, blokzincirleri şeffaf ve güvenli veri dağıtımını ve ödemeyi teşvik etmek için işbirliğine dayalı ağ ortamlarında açık defter çözümleri sunabilir. Böylece kaynak sağlayıcıları ve ağ kaynak dilimleri müşterileri işlemleri izleyebilir ve görüntüleyebilirler. Ek olarak, 5G'de mobil ağ kaynağı ticareti gibi hizmet ticareti sistemleri, farklı servis sağlayıcılar ve IoT kullanıcıları arasında açık ve güvenilir veri alışverişini garanti eden akıllı sözleşmeler kullanılarak otomatik olarak blokzinciri üzerinde yürütülebilirler.

4.6.4. Güvenlik ve Gizlilik

Sunduğu güvenlik ve mahremiyet, blokzinciri teknolojisinin en güçlü özelliklerindendir. Blokzincirindeki temel güvenlik noktası, genel ve özel anahtarların kullanılmasıdır. Blokzinciri sistemleri, katılımcılar arasındaki işlemleri güvence altına almak için asimetrik kriptografiden yararlanır. Bu anahtarlar sayı dizileriyle rastgele elde edilirler. Böylece bir kullanıcı diğer kullanıcıların özel anahtarını genel anahtarlarından matematiksel olarak tahmin edemez. Bu durum, blokzinciri kayıtlarını olası tehditlere karşı korur ve veri sızıntısı ile ilgili şüpheleri azaltır. Ek olarak, blokzinciri ve akıllı sözleşmeler tarafın-

dan sunulan gizlilik hizmeti, veri kaynağına ayrıcalıklar sağlar. Başka bir deyişle, veri sahiplerinin bilgilerinin gizliliğini bir blokzincirinde sağlamasına imkân verir. Blokzinciri özellikle akıllı sözleşmeler tabanlı erişim kuralları oluşturarak kişisel verilerin korunmasını ve mülkiyetini sağlar. Kötü amaçlı erişimler, kullanıcı kimlik doğrulaması ve akıllı sözleşme yetkilendirmesi ile tespit edilebilmektedir.

4.7. YAZILIM TANIMLI AĞ VE BLOKZİNCİRİ TEKNOLOJİLERİNİN YAYGIN KULLANILDIĞI AĞ MİMARİLERİ

Blokzinciri teknolojisi ve YTA konsepti sağladıkları etkin fonksiyonlar sayesinde birçok akademik ve endüstri çalışmasında çeşitli ağ mimari çözümlerinde faydalanılmaktadır. Bu bölüm, siber güvenlik ve iletişim ağları ile ilgili çözümleri nesnelerin interneti (IoT), araçsal ağlar, bulut teknolojileri ve 5G gibi YTA ve blokzinciri işbirliklerinin yoğun olduğu ağ mimarileri perspektiflerinden gözden geçirecektir.

Tablo 4.3. YTA ve Blokzinciri Teknolojilerinin Yaygın Kullanıldığı Ağ Mimarileri, Kullanılan YTA Denetleyici Mimarisi ve Faydalanılan Blokzinciri Özellikleri

Eser	Ağ Mimarisi	Denetleyici Mimarisi	Kullanılan Blokzinciri Özellikleri
[64]	Nesnelerin İnterneti	Dağıtık	Değişmezlik, İzlenebilirlik, Şeffaflık
[45]	Nesnelerin İnterneti	Dağıtık	Değişmezlik, İzlenebilirlik, Şeffaflık, Merkeziyetsizleşme
[65]	Nesnelerin İnterneti	Merkezi	Değişmezlik, Merkeziyetsizleşme
[66]	Nesnelerin İnterneti	Merkezi	Değişmezlik, İzlenebilirlik, Şeffaflık
[67]	Araçsal Ağlar	Dağıtık	Değişmezlik, Şeffaflık, Merkeziyetsizleşme
[68]	Araçsal Ağlar	Belirtilmemiş	Değişmezlik, Merkeziyetsizleşme, Anonimlik, Güven İhtiyaçsızlık
[69]	Araçsal Ağlar	Merkezi	Değişmezlik, Şeffaflık, Anonimlik, Güven İhtiyaçsızlık
[70]	Araçsal Ağlar	Merkezi	Değişmezlik, İzlenebilirlik, Şeffaflık, Merkeziyetsizleşme
[71]	Araçsal Ağlar	Merkezi	Değişmezlik, İzlenebilirlik, Şeffaflık, Merkeziyetsizleşme
[72]	Bulut	Merkezi	Değişmezlik, İzlenebilirlik, Şeffaflık
[73]	Bulut	Merkezi	Değişmezlik, İzlenebilirlik, Şeffaflık
[74]	Bulut	Merkezi	Değişmezlik, İzlenebilirlik, Şeffaflık, Güven İhtiyaçsızlık
[75]	Bulut	Merkezi	Değişmezlik, İzlenebilirlik, Şeffaflık
[76]	Bulut	Merkezi	Değişmezlik, İzlenebilirlik, Şeffaflık, Anonimlik
[77]	5G	Merkezi	İzlenebilirlik, Anonimlik

Tablo 4.3, YTA ve blokzinciri teknolojilerinin yaygın kullanıldığı ağ mimarileri, kullanılan YTA denetleyici mimarisi ve faydalanılan blokzinciri özelliklerini özetlemektedir. Blokzinciri ve YTA teknolojileri nesnelerin interneti, bulut ve araçsal ağlar gibi mimarilerde birbirini tamamlayan özelliklerinden dolayı yaygın kullanım alanı bulmaktadırlar. Bu tür işbirliklerinde blokzincirinin değişmezlik, izlenebilirlik, şeffaflık, merkeziyetsizleşme, anonimlik gibi özellikleri araştırmacıların çalışmalarında faydalandıkları özelliklerindendir.

4.7.1. Nesnelerin İnterneti (IoT)

IoT, hesaplama kaynakları (CPU, RAM, depolama vs.) bakımından kısıtlı birbirine bağlı cihazların ağıdır. Bu tür cihazların heterojen veya homojen olarak bağlantısı bazı zorlukları da beraberinde getirmektedir. Bu zorluklar arasında şeffaflık, denetlenebilirlik ve kimlik çatışması bulunmaktadır. Bu tür cihazları yönetmenin merkezileştirilmiş, merkeziyetsiz ve dağıtık modeller gibi metotları mevcuttur. Merkezi sistemlerde mevcut tüm düğümlere ulaşmak çok zordur. Bu tür sistemlerdeki ana zorluklara sunucu tarafında veya istemci tarafında bir arızanın olması örnek verilebilir. Her iki durumda da bu tür sorunları merkezi olarak yönetmek zordur. Bununla birlikte, cihazları merkezi olmayan bir yöntem ile kontrol etmek performans sorunları getirmektedir. Blokzincirleri ve akıllı sözleşmeler bu sorunları ortadan kaldırabilir [78]. Blokzinciri kullanarak, aynı ekosistemdeki cihazları kontrol etmek ve yönetmek mümkündür. IoT cihazlarının tamamen veya kısmen bağımlı cihazlar olduğu göz önüne alındığında, saldırılara karşı hassas olmaları söz konusudur. Öte yandan, blokzincirleri, blokzinciri platformunda depolanan genel anahtar ve IoT cihazlarında depolanan özel anahtar aracılığıyla güvenliği sağlayabilirler [79].

IoT konseptinin yakın zamanda yaygınlaşması, yeni yaygın hizmetler sağlamak ve günlük yaşam görevlerimizi otomatize etmek için milyarlarca IoT cihazının birbirine bağlandığı akıllı şehirlerin ortaya çıkmasının önünü açıyor. Bununla birlikte, güvenli olmayan IoT cihazlarının sayısı hızla artması, DDoS saldırılarının etkisini de hızla artıyor. Mirai gibi IoT botnetlerinin ortaya çıkmasıyla, IoT'ye yönelik bakış açısı, akıllı şehirlerin bir etkinleştiricisi pozisyonunda olmaktan siber saldırılar için güçlü bir yükseltme aracına dönüşmektedir. Bu durum saldırıya karşı işbirliği konusunda karar verme sürecinde esneklik ve verimlilik sağlamak için YTA tabanlı yeni tekniklerin

geliştirilmesini motive etmektedir. YTA ve blokzinciri gibi yeni ortaya çıkan teknolojiler, IoT ortamı için güvenli, düşük maliyetli, esnek ve verimli DDoS anti-saldırı işbirliği için yeni fırsatlar doğuruyor. [64] nolu çalışmada, işbirlikçi DDoS saldırılarını hafifletmek için blokzinciri tabanlı bir çerçeve olan Co-IoT'yi önermektedirler. Çalışmada YTA tabanlı ağ alanları arasında saldırıya karşı işbirliğini kolaylaştırmak ve saldırı bilgilerini güvenli, verimli ve merkezi olmayan bir şekilde aktarmak için Ethereum tabanlı akıllı sözleşmeler kullanılmaktadır.

Geleneksel bir ağ mimarisinde, uygulamalar tarafından kullanılan hizmet kalitesi yaklaşımı statiktir. Ağ kaynakları, belirli bir uygulamanın konuşlandırılması sırasında sabit tutulur. IoT ağları da ağ koşullarına dinamik olarak uyum sağlayacak kadar çevik değildirler. Bu nedenle, son kullanıcıya dinamik hizmet kalitesi sağlamak için, ağ kaynaklarının talep üzerine tahsis edilebilmesi için ağın yeniden yapılandırılması gerekir. Öte yandan, YTA, tüm ağı mantıksal bir varlık olarak ele alması ve programlanabilir doğası nedeniyle ağa esneklik sağlar. YTA'da trafik yönetimi, rota yönetimi, rota hesaplama gibi uygulamalar, ağ kaynaklarının dinamik olarak izlenmesini ve kontrolünü kolaylaştırmak için merkezi bir denetleyicinin üzerine konuşlandırılır. [45] nolu çalışmada, gerekli hizmet kalitesi seviyesini elde etmek için bir YTIoTA'da blokzinciri teknolojisini önermektedirler. İki taraf arasındaki kurallar ve hizmet anlaşmaları, çalışmada Ethereum blokzinciri düğümlerine yerleştirilen akıllı sözleşmelere uyarlanmaktadır. Stackelberg oyun stratejisini kullanarak hem hizmet sağlayıcı hem de hizmet talep eden, kendi yardımcı program işlevlerini en üst düzeye çıkarmaya çalışırlar. Bu durumda, akıllı sözleşmeler, servis sağlayıcıların ve kullanıcıların verileri değiştirmesini neredeyse imkânsız hale getirmekteler.

Akıllı şehir IoT ağlarında güvenli olmayan sabit ve taşınabilir cihazların kullanımının katlanarak artması, akıllı şehirlerin güvenliğini siber saldırılara karşı hayati bir sorun haline getirmektedir. IoT ağlarında merkezileştirilmiş ve dağıtık mimarilere dayanan güvenlik saldırılarını tespit etmek için çeşitli mekanizmalar literatürde önerilmiştir. Ancak IoT cihazlarında bulunan depolama kısıtlamaları, yüksek hesaplama maliyeti, yüksek gecikme süresi ve tek bir hata noktası gibi sorunlar nedeniyle bunlar verimsiz kalma eğilimindedir. Dahası, mevcut güvenlik mekanizmaları, siber saldırılara karşı optimum güvenlik ve savunma sağlamak için tüm akıllı şehir IoT ağı boyunca geçmiş verileri izleme ve toplama sorunuyla karşı karşıyadır. Mevcut zorlukları ele almak

içinde akıllı şehir IoT ağı, IoT ağındaki saldırıları daha etkili bir şekilde tespit etmek adına YTA, blokzinciri, sis ve mobil uç bilişim teknolojilerine dayanan merkezi olmayan bir güvenlik mimarisi önerilmektedir [65]. Çalışmada YTA, önerilen mimaride optimum bir saldırı algılama modeli sağlamak için tüm IoT ağındaki trafik verilerinin sürekli izlenmesinden ve analizinden sorumludur. Blokzinciri ise mevcut mimarinin doğasında bulunan tek hata noktası sorununu azaltmak için merkezi olmayan saldırı algılaması sunmaktadır. Sis ve mobil uç bilişim ise sis düğümünde saldırı algılamayı ve ardından kenar düğümünde saldırı azaltmayı desteklemektedir. Böylece önerilen mimari daha az depolama kısıtlaması, daha ucuz hesaplama ve düşük gecikme ile erken algılama ve azaltma sağlar.

İnternete bağlı akıllı cihazların sayısındaki ve çeşitliliğindeki hızlı artış, mevcut IoT ağında esneklik, verimlilik, kullanılabilirlik, güvenlik ve ölçeklenebilirlik sorunlarını gündeme getirmiştir. Bu sorunlar, IoT ağına büyük ölçekte dağıtılan anahtar mekanizmalardan kaynaklanmaktadır. [66] nolu eserde güvenli, ölçeklenebilir ve verimli bir ağ mimarisi tasarlamak için gerekli ilkeleri izleyerek ve blokzinciri teknolojisini (DistBlockNet) kullanarak dağıtık güvenli bir YTA mimarisini IoT için önerilmektedir. Çalışmada, akış kuralı tablosunun güvenli bir şekilde doğrulamak ve IoT yönlendirme cihazları için en son akış kuralları tablosunu indirmek için bir blokzinciri tekniği kullanan yeni bir şema önerilmiştir. Önerilen mimaride, yöneticinin binlerce öneri ve görüşü manuel olarak gözden geçirmesi ve uygulaması gerekmeden, güvenlik otomatik olarak tehdit ortamına uyarlanmaktadır.

4.7.2. Araçsal Ağlar

Akıllı Ulaşım Sistemlerinin - AUS (Intelligent Transportation Systems) amacı, Araçsal Ad Hoc Ağlar'ın (Vehicular Ad Hoc Networks - VANETs) ağ performansını artırmaktır. Akıllı ulaşım sistemleri Araçların İnterneti (Internet of Vehicle - IoV) ortamına yeni fırsatlar sunsa da IoV önerilerinde birbirine bağlı eşler arasında güven tesis etme ihtiyacı da dâhil olmak üzere bazı güvenlik endişeleri vardır. Güvenilir ve düşük gecikmeli iletişim hizmetleri sağlayan 5G iletişim sistemi, VANET'lerdeki zorlukları üstlenebilecek teknoloji olarak görülüyor. YTA'ların dâhil edilmesi de etkili ağ yönetimi sağlamaktadır. Bazı özellikleri olarak âdemi merkeziyetçilik, şeffaflık ve değişmezliğe sahip olan blokzinciri, ağ platformlarında güven sağlamak için tasarlanmıştır. [67]

nolu çalışmada, 5G ve sis bilişim sistemlerinde IoV için blokzinciri ve YTA destekli bir mimari sunmaktadır. YTA, 5G ağlarında VANET sistemleri için yönetim süreçlerini idare etse de blokzincirinin eklenmesi ağın verimliliğini artırmaktadır. Ayrıca, bir araç tarafından sağlanan her bilgi çok önemli görüldüğünden, önerilen çalışmada araçlar arasında güveni artırmak için blokzinciri dâhil edilmiştir. Çalışma, ayrıca birbirine bağlı akranlar tarafından verilen kararlara göre güvenilir kabul edilen mesajlar veren araçlara itibar puanlarının verildiği bir güven sistemi önermektedir.

Araçsal IoT güvenliği günden güne kritik bir konu haline gelerek daha fazla dikkat çekmektedir. Araçlar, artan yerleşik algılama, hesaplama ve iletişim yetenekleriyle daha fazla özerkliğe sahip olmaktadırlar. Böylelikle araçsal ağlar, araçlar arasında yüksek çözünürlüklü videolar gibi büyük hacimli verileri paylaşabilir ve bunları bilgi işlem veya yönetim merkezine bildirebilirler. Bu tür mesajlar, ulaşım sistemindeki güvenlik garantisine ve trafik durumlarının farkında olunmasına yardımcı olarak yönetim verimliliğini artırmaktadır. Ancak bir VANET'te araçlar, çoğu durumda birbirlerine yabancı oldukları için genellikle tam olarak işbirliği yapma ve birbirlerine güvenme konusunda isteksizdirler. VANET yüksek hareketlilik (mobilite) ve değişkenliğe sahip olduğundan büyük ölçekli senaryolarda kötü niyetli araçlar veya yanlış davranışlar olması kaçınılmazdır. Gerçek olmayan verilerin (örneğin aslında sıkışık veya trafik kazası olan bir yolun temiz olduğunu iddia etmek gibi) kötü niyetli araç düğümleri tarafından yayılması, ulaşım sistemini büyük ölçüde tehlikeye atmaktadır. VANET'in, hizmetle ilgili verilerin doğrulanmış veya güvenilir araçlardan ve altyapılardan sağlandığı güvenli bir IoT ortamını garanti etmesi gerekmektedir. Bu veriler değişmez, güvenilir ve gerçek olmalıdır. Kişisel bilgilerinin sızmasını tolere edebilecek bir kullanıcı olmadığından, veri alma ve iletiminde kullanıcı gizliliğinin korunması bir diğer kritik konudur. Ayrıca, araçların IoT hizmetlerinde ki gelecekteki eğilim, merkezsiz güven, işbirliğine dayalı zekâ ve mekânsal-zamansal duyarlılık gerektirmektedir [80].

Xie ve arkadaşları [68], YTA özellikli 5G-VANET'ler de araçsal IoT ortamı için merkezi olmayan blokzinciri tabanlı bir güvenlik çerçevesi önermektedirler. Çalışmadaki blokzinciri, araç sistemindeki tüm aktif düğümler tarafından oluşturulan bir P2P ağı tarafından korunmaktadır. Araçlar, gerçek zamanlı yol durumu mesajlarını birbirlerine yayınlar iken blokzinciri mesaj kaynaklarının yanı sıra tüm mesajları da kaydetmektedir. Blokzinciri değiştirilemezlik özelliği kullanılarak, kaynak mesajların doğruluklarının onaylanmasında kul-

lanılmaktadır. Yazarlar, blokzinciri tabanlı çerçevenin desteğiyle, kötü niyetli düğümlerin sahte mesajlar üretmesi veya mesajların değiştirilmesi durumlarına karşı araçsal ağlar için güven yönetimi sunmaktadırlar. Önerilen çalışmada ayrıca önerilen çerçeveye gerçek zamanlı bir video rapor hizmeti de uygulamaktadır. Videolar şifrelendikten sonra bulut sunucularına yüklenir ve ilgili mesajlarla eşleşip eşleşmediklerini kontrol etmek için kullanılmaktadır. [69] nolu çalışmada yazarlar, Araçsal Sosyal Ağı (Vehicular Social Network - VSN) verimli bir şekilde yönetmek ve güvence altına almak için Yazılım Tanımlı Araç Ağları (Software-Defined Vehicular Networks - SDVN) ve blokzinciri olmak üzere iki ana konsepte dayanan yeni bir çerçeve önermektedirler. SDVN kullanımı, ağı programlanabilir, sanallaştırılmış ve bölümlenebilir hale getirirken aynı zamanda da tek hata noktası adı verilen bir güvenlik açığı oluşturur. Bu nedenle, yazarlar, işlemlerin onaylanmasını sağlayan ve blokzinciri madenci düğümlerini kullanarak dağıtık bir şekilde verilerin anonimliğini sağlayan bir blokzinciri paradigması sunmaktadırlar. Bu amaçla, çalışmada üç seviyeli denetleyici sunulmaktadır: Ana Denetleyici (AD), Yol Kenarı Birimleri (YKB) ve madenciler. AD, ağ topolojisi gibi ağa genel bir bakış sunarken YKB, AD ve madenciler arasında bir aracı olarak görev yapmaktadır.

Elektrikli Araçlar (Electrical Vehicles - EVs) potansiyel olarak karayolları üzerinde 5G gibi teknolojilerle bağlanan geniş bir araç ağı oluşturacaktırlar. Elektrikli araçlar, pillerinin/enerjilerinin daha iyi yönetimi için enerji ticareti kararlarını almak ve optimize etmek amacıyla enerji (ve diğer) bilgilerini kendi aralarında veya hizmet sağlayıcı/hizmet kuruluşu ile paylaşmak için mevcut bilgi ve iletişim teknolojilerini kullanabilirler. Bu durumda elektrikli araçlardan hizmet kuruluşuna ve tersi yönde bilgilerin açıkça güvence altına alınması gerekir. Böylece hem elektrikli araçlar hem de hizmet sağlayıcı enerji ticaretindeki karar verme sürecinde kullanılan bilgilere güvenebilirler. Bununla birlikte, ağda fiyatlandırma bilgilerini veya enerji taleplerini değiştirebilen finansal olarak motive olmuş bir saldırgan varsa, bu durumda potansiyel güvenlik ve gizlilik çıkarımlarının yanı sıra şebeke arızası ve ölümler gibi durumlar nedeniyle finansal ve yasal sonuçlar da vardır. [70]'te yazarlar, AUS de güvenli enerji ticareti için blokzinciri tabanlı bir enerji ticareti planı olan BEST'i sunmaktadırlar. Öneri, temel mimari olarak YTA'yı kullanmaktadır. Önerilen şemada, blokzinciri konsorsiyumunda enerji ticareti işlemleri için enerji paraları kullanılmaktadır. Çalışmada tüm ağ işlemlerini doğrulamaktan

sorumlu madenci düğümleri, enerji gereksinimleri, fiyatlandırma ve kalma süresi gibi çeşitli faktörlere göre seçiliyorlar. Bu, bir saldırganın düğüm seçim sürecini etkileme veya değiştirme zorluğunu artırarak sistemin genel güvenliğini artırmaktadır.

Elektrikli araçlar, akıllı şebekeye çeşitli enerji yönetimi çözümleri sunarak akıllı ulaşım sektörünü dönüştürmektedirler. Araçtan şebekeye (Vehicle-to-Grid - V2G) bir ortamda elektrikli araçlar ve şarj istasyonları (Charging Stations - CS) arasında enerji ticareti, akıllı şebekedeki popüler dikey durumlardan biridir. Ancak, enerji ticareti kararlarının uzaktan kontrol merkezlerinde işlenmesi, gecikmede ve ağ yükünde artışa neden olur. Diğer taraftan, bu sorunların dışında, böyle bir ortamda enerji ticareti yapılırken güvenlik endişeleri de devam etmektedir. Bu nedenle, [71] nolu çalışma yukarıda belirtilen sorunları ele almak için YTA destekli V2G ortamında güvenli enerji ticareti için blokzinciri tabanlı bir hizmet-olarak-kenar (edge-as-a-service) çerçevesi olan SURVIVOR'u sunmaktadır. Önerilen çerçevede, enerji ticareti kararları, uç düğümler aracılığıyla elektrikli araçların konumuna daha yakın işlenmektedir. Ayrıca, enerji ticareti işlemlerini güvence altına almak için, onaylayıcı düğümlerin bir fayda işlevi temelinde mevcut tüm düğümler arasından seçildiği ve işlemlerin doğrulanmasından sorumlu olduğu bir blokzinciri kullanılmaktadır. Bu tür düğümler seçildikten sonra, YTA özellikli V2G ortamında güvenli enerji ticareti için fikir birliğine dayalı bir blokzinciri mekanizması sunulmaktadır. Bu mekanizmada, PoW problemleri oluşturmaktan uç düğümler sorumludur. Her bir PoW problemi için benzersiz bir PoW özet değeri hesaplanır. Son olarak, tüm çerçeve, genel gecikmeyi azaltmak ve akıllı ulaşım ağının verimini artırmak için YTA mimarisi tarafından desteklenmektedir.

4.7.3. Bulut Mimarileri

Bulut bilişim, modern bilgi sistemlerimizde yaygın olarak kullanılmaktadır. Hem akademi hem de endüstri, yüksek ölçeklenebilirlik, tatmin edici kullanılabilirlik, yüksek performans, uygun maliyetli yatırım, gelişmiş hata toleransı yeteneği ve benzeri yetenekleri nedeniyle bulut bilişim teknolojisiyle ilgilenmektedir [81]. Geleneksel bulut bilişim sistemi, büyük ölçekli Bilgi Teknolojileri (BT) kaynaklarını (ağ iletişimi, önbelleğe alma ve bilgi işlem kaynakları vb.) entegre ederek aşırı hesaplama artışı sorununu çözmeyi amaçlamaktadır. Geleneksel bulut bilişim sistemi, isteğe bağlı BT kaynaklarını dinamik

olarak sağlasa da küresel bulut hizmetlerinin gereksinimlerini karşılayamaz. Bu durumda, bulut federasyonu, bağımsız bulut hizmeti sağlayıcıları arasında işbirliğini mümkün kılarak bulutlar arası hizmetleri sağlamak için yeni nesil bulut bilişim olarak önerilmektedir. Bir bulut federasyon sisteminde, bulutlar arası hizmetlerin sağlanması için verilerin farklı bulut hizmeti sağlayıcıları arasında paylaşılması gerekmektedir. Böylece, depolanan veriler hem verilere sahip olan bulut hizmeti sağlayıcısı hem de federasyon sistemindeki diğer bulut hizmeti sağlayıcıları tarafından kullanılabilir. Bulut federasyon sisteminde güvenli veri paylaşımını desteklemek için, [72]'de verilen çalışmada erişim kontrolü için blokzinciri teknolojisi uygulanmıştır. Önerilen çalışmada erişim kontrol politikaları, erişim kontrol yönetimini otomatize etmek için blokzincirinde depolanan akıllı sözleşmeler olarak programlanmaktadır. Benzer şekilde, [73] nolu çalışmada, bulut hizmeti sağlayıcılarının güvenilirliğini değerlendirmek için DC-RSF adı verilen blokzinciri tabanlı bir itibar sistemi önerilmiştir. Bu çalışmada her bir bulut hizmet sağlayıcısının kredi değeri blokzincirinde saklanmaktadır.

Bulut mimarisi, yüksek düzeyde kullanılabilirliği, ölçeklenebilirliği ve stratejik değeri nedeniyle büyük veri analitiği gibi farklı uygulamalar için değerli bir çözüm haline gelmiştir. Bununla birlikte, bulut güvenliği gibi alanlarda bulut mimarisini yönetmede hala zorluklar mevcuttur. [74] nolu çalışmada yazarlar, bütünlük ve kullanılabilirlik gibi bulut yönetiminin güvenliğini artırmak için blokzinciri ile güçlendirilmiş YTA destekli bir ağ altyapısı geliştirmektedirler. Önerilen ağ altyapısı temel olarak iki katmandan oluşur: Çok denetleyicili YTA ağ katmanı ve blokzinciri tabanlı güvenlik ve özerklik katmanı. Bu iki katmanın entegrasyonu, kontrol ve yönetim komutlarının bütünlüğünü geliştirmek için tasarlanmıştır. Önerilen çerçevede, blokzinciri tabanlı güvenlik ve özerklik yönetim katmanı, YTA denetleyicisi aracılığıyla çok denetleyicili YTA ağ katmanıyla etkileşime girerek daha geniş bir zaman ölçeğinde karar vermeye izin vermektedir. Son olarak, çalışmada önerilen ağ altyapısı, bulut mimarisinin kullanılabilirliğini iyileştirmek için otonom bant genişliği provizyonuna da olanak tanımaktadır.

YTA teknolojisi, internetin ve iletişim ağlarının gelişiminin evrimini hızlandırmaktadır. Denetleyicilerin mantıksal olarak merkezileştirilmesi ve genel ağ hâkimiyeti ile YTA, esneklik, kullanılabilirlik ve programlanabilirlik açısından bilgisayar ağlarının özelliklerini değiştirmektedir. Ancak bu gelişme, ağ iletişim güvenliği zorluklarını da artırmaktadır. [75] nolu kaynakta, YTA

güvenliğini artırmak için YTA veri katmanı cihazlarına yanlış akış kuralları enjeksiyonunu önlemek için BCFR çözümünü önermektedir. Önerilen çözümde yazarlar, denetleyici kimlik doğrulamasını ve denetleyici ile diğer ağ öğeleri arasında dolaşan trafik akışının bütünlüğünü sağlamak için blokzinciri teknolojisini kullanmaktadırlar.

YTA tabanlı enerji İnternet, yenilenebilir enerji için dağıtık bir mimari olduğundan ve giderek daha fazla ilgi gördüğü için, geleneksel merkezi elektrik enerjisi ticaret modeli artık geçerliğini yitirmektedir. Fakat aynı zamanda, sözü edilen bu sistemlerde bazı problemler ve zorluklar giderek daha belirgin hale gelmektedir. Blokzinciri, âdemi merkeziyetçilik, koordine edilmiş özerklik ve değiştirilemezlik sayesinde hızla geliştirilmekte ve çeşitli alanlarda uygulanmaktadır. [76] nolu çalışmada yazarlar, blokzinciri teknolojisi ile desteklenen, YTA özellikli, dağıtık bir İnternet enerji ticareti planı önermektedirler. Önerilen çalışma, mahremiyetin korunması öncülüğünde işlem nesnelerinin makul bir eşleşmesini sağlamaktadır. Sistemde tüm veriler bulutta saklanmaktadır. Çalışmada yalnızca iki işlem eşleştiğinde ve başarılı olduğunda, düğüm daha fazla eşleştirme için buluttan bilgi alabilmektedir. Önerilen sistem verileri bulutta depolanır ve blokzinciri düğümleri bunlara doğrudan erişemez. Böylece kullanıcıların gizliliğini korunmaktadır.

4.7.4. 5G

Blokzinciri, çeşitli sektörlerde hizmetlerin sunulma biçiminde devrim yaratma potansiyeline sahip en umut verici teknolojilerden biridir. Bu teknoloji tek bir aracıya güvenmeksizin herkesin kabul ettiği, dağıtık, değiştirilemez, tek bir doğruluk kaynağı sağlar. Blokzinciri, tüm 5G ekosisteminde uçtan uca hizmet sunumunu sağlamak için 5G ağı ile entegre olma potansiyeline sahiptir. Bu nedenle, blokzinciri, otomatik kaynak yönetimi, güvenlik ve dolandırıcılık önleme, her yerde bulunan bilgi işlem, güvenilir içerik dağıtımı ve veri yönetimini içeren 5G ağlarının tam potansiyelini gerçekleştirmede önemli bir rol oynayacaktır. Blokzinciri, kendine has özellikleri ile P2P tarzında veri işlemlerini yüksek güvenlik ve güvenilirlikle sağlayarak 5G ağlarında çeşitli hizmetleri mümkün kılacaktır. Şu anda, mevcut 5G platformları için en büyük zorluk, olağanüstü sayıda kaynak ve birkaç kötü niyetli kullanıcı olması durumunda halka açık, şeffaf ve adil bir sistemi garanti etme ihtiyacıdır. Eşsiz âdemi merkeziyetçilik, yüksek düzeyde veri gizliliği, güvenlik,

şeffaflık ve değiştirilemezlik özellikleriyle blokzinciri, bariz bir seçim haline gelmektedir. Bu nedenle, blokzincirini 5G mimarisine entegre etme ihtiyacı giderek artmaktadır. Böyle bir mimari, merkezi bir aracıya ihtiyaç duymadan işlemleri gerçekleştirebilen, otomatik, kendi kendini idame ettiren, kendi kendine hizmet veren ve kendi kendini yöneten bir ağ ortaya koyacaktır. Bir blokzinciri önerisi, heterojen erişim düğümleri ve cihazlar arasında sorunsuz provizyona izin vererek yeni nesil dağıtık kablosuz ağlara da yardımcı olacaktır. Blokzinciri ile erişim düğümleri, ağlar ve aboneler arasındaki hükümler ve anlaşmalar, dijital akıllı sözleşmeler olarak anında müzakere edilebilir. Blokzinciri, ağdaki cihazların ağ operatörü ile en iyi hizmet için müzakere etmesine izin vererek bu anlaşmaların akıllı sözleşmeler kullanılarak gerçekleştirilmesine imkân sağlayacaktır. Bu durum, 5G ağında yeni ücretlendirme ve iş modelleriyle sonuçlanan, ağdaki ayrı düğümlere özelleştirilmiş hizmet sunumuna izin verecektir.

Modern akıllı ulaşım sistemleri, yalnızca araçsal IoT hizmetleri için yeni fırsatlar değil, aynı zamanda VANET ağları için de yeni zorluklar getirmektedir. Gelişmiş ağ performansının yanı sıra, aynı zamanda kullanıcı gizliliğini korurken güven yönetimini gerçekleştirmek için pratik ve güvenilir bir güvenlik şemasına ihtiyaç vardır. Ortaya çıkan 5G mobil iletişim sistemi, ultra güvenilir, düşük gecikmeli kablosuz iletişim hizmetleri için öne çıkan bir teknoloji olarak görülmektedir. Ayrıca, YTA mimarisini 5G tabanlı VANET'e (5G-VANET) dâhil etmek, küresel bilgi toplama ve ağ kontrolüne olanak tanıyacağı düşülmektedir. Bu nedenle, ulaşım izleme ve raporlamayla ilgili gerçek zamanlı IoT hizmetleri etkili bir şekilde desteklenebilir. [68] nolu çalışma, YTA özellikli 5G-VANET'te ulaşım sistemi ve araçsal IoT ortamındaki güvenlik ve gizlilik sorununu araştırmaktadır. Blokzinciri tabanlı bir güvenlik çerçevesi, blokzincirinin merkeziyetsizlik ve değiştirilemezlik özellikleri sayesinde gerçek zamanlı bulut tabanlı video raporları ve araç mesajlarında güven yönetimi gibi araçsal IoT hizmetlerini desteklemek için tasarlanmıştır. Çalışma, YTA özellikli 5G-VANET modelini ve blokzinciri tabanlı çerçevenin zamanlama prosedürlerini açıkça göstermektedir.

IoT'nin ve yüksek bit oranlı video uygulamalarının hızlı evrimi ile birlikte 5G ve Ötesi – 5GÖ sistemleri geliştirmek, ortaya çıkan hizmetleri karşılamak için geniş bant genişliğine sahip yüzlerce terminali birbirine bağlayan kaçınılmaz bir trend haline geldi. Bu tür hizmetleri barındırmak için, Yazılım Tanımlı Veri Merkezi Optik Ağı - YTVOA (Software-Defined Data Center Optical

Network - SDCON) büyük bant genişliğine sahip içerik depolama ve bağlantı sağlamasına ek olarak farklı talepler için arzu edilen ölçülere sahip özelleştirilmiş optik spektrum atar. Kontrol perspektifinden ise, kontrol esnekliğini gerçekleştirerek küresel bir bakış açısıyla optik kaynakların işlevleri ve hizmetleri üzerinde bir kontrol sunmaktadır. Yang ve arkadaşları [77], iki tane dağıtık çok denetleyicili güvenilir yönlendirme şeması (MCR) sunarak, 5GÖ öngörülerinde YTVOA için yeni bir dağıtık blokzinciri tabanlı güvenilir denetim mimarisi (BlockTC) önermektedirler. BlockTC, her bir ağ etki alanının (domain) özel bilgilerini ifşa etmeye gerek duymayan çok ağlı (multi-domain) YTVOA senaryosunda, üçlü denetleyiciler arasında güvenilir ağlar arası yönlendirme doğrulaması gerçekleştirebilir.

4.8. SONUÇ VE DEĞERLENDİRMELER

YTA, veri ve kontrol düzlemlerinin dikey entegrasyonunu bozarak ağın kontrol mantığını YTA denetleyicisi adı verilen merkezi bir varlığa taşımaktadır. Bu uygulama, ağ yönetimini ve yapılandırmasını iyileştirse de YTA mimarisi çok sayıda siber saldırı türüne karşı hala savunmasızdır. Öte yandan, blokzinciri teknolojisi, verilerin merkezi olmadığı ve bir P2P ağında güvenilir bir üçüncü tarafa olan ihtiyacı ortadan kaldırdığı için YTA'nın tam tersidir. Blokzinciri, işlemlerin erişilebilirliğine göre genel, özel veya konsorsiyum olarak kategorize edilir. Herkese açık blokzincirinde, tüm düğümler uzlaşma sürecine katılır ve işlem ayrıntılarını görüntüleyebilirler. Bununla birlikte, özel ve konsorsiyum blokzincirinde, işlemlerin erişilebilirliği, merkezi bir kuruluş tarafından alınan bir karara dayanarak verilirken yalnızca sınırlı sayıda önceden onaylanmış düğümler, uzlaşma prosedüründe yer alırlar.

Blokzinciri ve YTA, erişim kontrolü, onaylama/kimlik doğrulama, yönlendirme, güvenlik gibi çeşitli ağ işlevlerinden yararlanarak birçok kullanım durumunda işbirliği yapabilirler. Blokzinciri; YTA tabanlı iletişim altyapılarının ağ verimliliğinin ve ölçeklenebilirliğinin arttırılmasına, dağıtık bir uzlaşmaya dayalı veri tabanı sağlayarak merkezi aracılara bağımlılığının azaltılmasına ve tek hata noktası gibi siber güvenlik sorunlarının azaltmasına yardımcı olabilir. Bununla birlikte, bu teknolojilerin, entegrasyonları ile oluşturulan yapıya taşınan kendi iç sorunları da vardır. Örneğin, blokzinciri, DDoS saldırıları, çift harcama saldırıları, Sybil saldırıları, defter güncellemeleri için

gecikme sorunları ve yoğun hesaplama yükü gibi çeşitli saldırı vektörlerinden muzdariptir. Öte yandan, YTA'da sahtecilik saldırıları (spoofing attakcs), korsan saldırıları (hijacking attacks), DoS saldırıları ve ortadaki adam saldırıları (man-in-the-middle attacks) gibi sorunlar da vardır. Bu sorunlar, blokzinciri teknolojisinin YTA altyapıları üzerinde düzgün çalışmasını tehlikeye atabilir. Örneğin, blokzincirine katılan bir YTA denetleyicisi saldırıya uğrayarak ele geçirilebilir ve bu nedenle blokzincirine gerçek dışı işlemler/bloklar eklenebilir.

Blokzinciri, son yıllarda gelişmekte olan ve gelecek vaat eden bir teknoloji olmasına rağmen, diğer tüm yeni teknolojiler gibi zorluklar da barındırmaktadır. Bu sorunlar hem akademiden hem de endüstriden araştırmacılar için olası yeni ve özgün araştırma/çalışma alanları/perspektifleri ortaya koymaktadır. Blokzinciri teknolojisini YTA gibi programlanabilir yeni ağ mimarilerinde entegre ederken ortaya çıkabilecek bazı zorluklar şu şekilde ifade edilebilir:

- *Blok Oluşturma Süresi*: Blok aralığı olarak da bilinen bu süre, blokzincirinin daha yüksek işlem hacmi elde etmesine yardımcı olabilirken blokzincirinde tutulan YTA ağlarının olay kayıtlarının (log) durumlarıyla ilgili olarak blokzinciri defterini güncel tutabilir. Öte yandan, bu aralık YTA ağ denetleyicilerinde daha fazla hesaplama yapılmasına ve ağlarda daha fazla bağlantı kaynağı kullanılmasına neden olabilir. Bu nedenle, blok oluşturma süresini ayarlamak için (olası makine öğrenmesi teknikleri aracılığı ile) hassas bir denge kurulmasına ihtiyaç vardır.
- *Uzlaşma Protokolü Ölçeklenebilirliği*: YTA mimarisi ile entegre edilmiş bir blokzinciri modelinde blokzinciri düğümleri olarak görev yapacak YTA ağ denetleyicileri, ağ ile ilgili görev ve paket işlemleri ile zaten yoğun olarak çalışır durumda olurlar. YTA ağ denetleyicilerini yeni bir blok oluşturmak için PoW uzlaşma protokolünde olduğu gibi kriptografik problemleri çözmek için yoğun hesaplama gerektiren görevlerle yüklemek, bu ağ denetleyicilerinin ağ ile ilgili performanslarını sınırlayabilir ve dolayısıyla daha fazla hesaplama gücü gerektirir. Hesaplama gereksinimi bakımından daha hafif ancak verimli bir uzlaşma protokolü ağ denetleyicilerinin yükünü hafifletecektir. Bu nedenle, bu ihtiyaçlara cevap verecek bir blokzinciri uzlaşma protokolü tasarlanması YTA ağlarında blokzinciri entegrasyonu için ihtiyaç haline gelmiştir.

- *Blokzinciri Defter Güncelleme Süresi*: YTA ve blokzinciri entegrasyonunda, ağ olay kayıtlarını (log) içeren blokzinciri işlemlerinin blokzinciri defterine yazılması gerekir. Oluşturulan blokzinciri işlemleri blokzinciri defterine yansıtılana kadar bazı adımlardan (diğer düğümlere gönderme, yeni bloğa dahil edilmeyi bekleme, yeni bloğu blokzinciri ağına yayma, vb.) geçer. Bu blokzinciri işlemlerinde belirtilen ağ kayıtlarının (log) durum değişikliklerinin, ağların dinamik yapısından dolayı blokzinciri defterine yansıtılana kadar tekrar değişmesi mümkündür. Bu nedenle, Blokzincirindeki bazı işlemler ağdaki gerçek güncel olay kayıt durumlarını yansıtmayabilir. Burada ifade edilen blokzinciri işlemlerinde tutulan ağ olay kayıtları ile gerçek ağ kayıt durumları arasında ki senkronizasyonu sağlayacak ve olası farklılıkları minimize edecek mekanizmalara ihtiyaç vardır.
- *İşlem Miktarı*: Blokzinciri teknolojisi ile güçlendirilen bir YTA mimarisinde blokzinciri işlemleri ağ olay kayıtlarını (log) yansıtarak topoloji, güvenlik ve hizmet kalitesi gibi durumlar ile ilgili verileri tutabilirler. Ağlar, boyut, topoloji, hizmetler, kullanıcılar vb. açısından karmaşık yapılara sahip olduklarından, çalışmaları sırasında sık sık güncellemeler alabilirler. Bu güncellemeler yeni işlemlere yansıtılır ve blokzincirine katılan tüm ağlara (düşümlere) yayılır. Bu blokzinciri işlemlerinin depolaması uzun vadede bir sorun haline gelebilir. Bu durum, denetleyicilerin uğraştığı ağ yükünü de etkileyebilir. Bu nedenle, ağ durum kayıtlarını kayba uğratmadan blokzinciri işlemlerini daha etkili ve verimli bir şekilde depolanmasını sağlayacak özgün mekanizmalara olan ihtiyaç artmaktadır.

Ülkemizde de YTA kullanan iletişim ağları çözümleri ve blokzinciri odaklı girişimler son zamanlarda adından söz ettirmeye başlamıştır. Türk Telekom'un SD-WAN (Software Defined-Wide Area Networks) çözümü olan TT Akıllı Ağ ile geniş alan ağları kolayca yönetebilir ve uygulama bazlı trafik yönetimini sağlayarak ağ kaynakları optimum seviye kullanılabilir. Yine bir Türk Telekom iştiraki olan Argela Networks YTA çözümleri sunmaktadır [82]. Ulak Haberleşme A.Ş YTA tabanlı Veri Merkezi çözümü olan uMAYA SD-DC ve YTA tabanlı geniş alan ağı çözümü olan uMAYA SD-WAN çözümlerini kullanıma sunmuşlardır [83]. Blokzinciri tarafında ise, BİLGEM UEKAE Matematiksel ve Hesaplamalı Bilimler Biriminin altında blokzinciri teknolojilerinin altyapısı, kurulumu, güvenlik ve mahremiyet analizi, iş modelleri, kitle

fonlama yaklaşımları ve muhtelif teknik detayları üzerine Ar-Ge faaliyetlerini icra etmek üzere Blokzincir Araştırma Laboratuvarı (BZLab) mevcuttur [84]. Yine, *Blockchain Türkiye Platformu* [85] Türkiye'de sürdürülebilir blokzinciri ekosistemi oluşturarak, bu teknoloji ile yeni dönem iş yapış biçimlerinin önündeki zorlukların giderilmesine yönelik bir paylaşım platformu oluşturma amacı ile birçok üniversite, enstitü, sivil toplum kuruluşu ve resmi kurum ile işbirlikleri yürütmektedir. Her ne kadar blokzinciri teknolojisi ve YTA mimarisinin bireysel kullanımları son yıllarda kurumsal ve ulusal boyutlarda artan bir ilgi görüyor olsa da bu iki teknolojinin meziyetlerinden aynı anda faydalanmak için onları beraber kullanan olgunlaştırılmış girişimler henüz ortaya konulmamıştır.

Sonuç olarak, blokzinciri ve YTA teknolojilerinin entegrasyonu, blokzincirinin değiştirilemezlik, âdemi merkeziyetçilik, şeffaflık ve gizlilik gibi doğal özelliklerinden yararlanarak ağ güvenliği ve erişim kontrolü gibi noktaları iyileştirmek için ağlara birçok avantaj sağlayabilir. YTA'yı blokzinciri kullanımlarına uygulayarak veya YTA uygulamalarına blokzinciri entegre ederek bu umut vaat eden iki teknoloji birçok pratik senaryoda birbirlerini tamamlayabilirler.

Teşekkür

Bu çalışma, Türkiye Bilimsel ve Teknolojik Araştırma Kurumu (TÜBİTAK) tarafından 120E448 nolu proje kapsamında desteklenmektedir. Yazar, desteklerinden ötürü TÜBİTAK'a teşekkürlerini iletmektedir.

KAYNAKLAR

[1] S. Sezer *et al.*, "Are we ready for SDN? Implementation challenges for software-defined networks," *Commun. Mag. IEEE*, vol. 51, no. 7, pp. 36–43, 2013, doi: 10.1109/MCOM.2013.6553676.

[2] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A Survey on Software-Defined Networking," *Commun. Surv. Tutorials, IEEE*, vol. PP, no. 99, p. 1, 2014, doi: 10.1109/COMST.2014.2330903.

[3] K. Bakshi, "Considerations for Software Defined Networking (SDN): Approaches and use cases," in *Aerospace Conference, 2013 IEEE*, 2013, pp. 1–9, doi: 10.1109/AERO.2013.6496914.

[4] S. Jain *et al.*, "B4: Experience with a Globally-deployed Software Defined Wan," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, 2013, pp. 3–14, doi: 10.1145/2486001.2486019.

[5] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," http://bitcoin. org/bitcoin. pdf," 2008.

[6] W. Diffie and M. E. Hellman, "New Directions in Cryptography," *IEEE Trans. Inf. Theory*, vol. 22, no. 6, pp. 644–654, 1976, doi: 10.1109/TIT.1976.1055638.

[7] L. Lamport, R. Shostak, and M. Pease, "The Byzantine Generals Problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, 1982, doi: 10.1145/357172.357176.

[8] S. Haber and W. S. Stornetta, "How to time-stamp a digital document," *J. Cryptol.*, vol. 3, no. 2, pp. 99–111, Jan. 1991, doi: 10.1007/BF00196791.

[9] L. Lamport, "The part-time parliament," *ACM Trans. Comput. Syst.*, vol. 16, no. 2, pp. 133–169, 1998, doi: 10.1145/279227.279229.

[10] D. Chaum, "Blind Signatures for Untraceable Payments," in *Advances in Cryptology*, Springer, 1983, pp. 199–203.

[11] M. Atzori, "Blockchain technology and decentralized governance: Is the state still necessary?," *Available SSRN 2709713*, vol. 6, no. 1, pp. 45–62, 2015, doi: 10.22495/jgr_v6_i1_p5.

[12] Open Networking Foundation, "OpenFlow Switch Specification (1.5.1)", Mar. 2015. https://opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf. Erişim Tarihi: 28 Haziran 2021.

[13] MarketsandMarkets™, "Software Defined Networking Market by SDN Type (Open SDN, SDN via Overlay, and SDN via API), Component (Solutions and Services), End User (Data Centers, Service Providers, and Enterprises), and Region - Global Forecast to 2023", 2019. https://www.marketresearch.com/MarketsandMarkets-v3719/Software-Defined-Networking-SDN-Type-12290928/. Erişim Tarihi: 28 Haziran 2021.

[14] Open Networking Foundation (ONF). https://www.opennetworking.org. [Online]. Available: https://www.opennetworking.org. Erişim Tarihi: 28 Haziran 2021.

[15] Open Networking Foundation (ONF), "SDN Architecture", Tech. Rep.," Jun. 2016. https://opennetworking.org/wp-content/uploads/2014/10/TR-521_SDN_Architecture_issue_1.1.pdf. Erişim Tarihi: 28 Haziran 2021.

[16] P. Lin *et al.*, "A west-east bridge based SDN inter-domain testbed," *Commun. Mag. IEEE*, vol. 53, no. 2, pp. 190–197, Feb. 2015, doi: 10.1109/MCOM.2015.7045408.

[17] M. Karakus and A. Durresi, "Quality of Service (QoS) in Software Defined Networking (SDN): A survey," *Journal of Network and Computer Applications*, vol. 80. pp. 200–218, 2017, doi: 10.1016/j.jnca.2016.12.019.

[18] M. Karakus and A. Durresi, "Economic Impact Analysis of Control Plane Architectures in Software Defined Networking (SDN)," in *2018 IEEE International Conference on Communications (ICC)*, May 2018, pp. 1–6, doi: 10.1109/ICC.2018.8422749.

[19] M. Karakus and A. Durresi, "An economic framework for analysis of network architectures: SDN and MPLS cases," *J. Netw. Comput. Appl.*, vol. 136, pp. 132–146, 2019, doi: 10.1016/j.jnca.2019.02.032.

[20] M. Karakus and A. Durresi, "Economic Analysis of Software Defined Networking (SDN) under Various Network Failure Scenarios," 2019, doi: 10.1109/ICC.2019.8761124.

[21] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "NISTIR 8202 Blockchain Technology Overview," *Natl. Inst. Stand. Technol.*

[22] W. Wang *et al.*, "A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks," *IEEE Access*, vol. 7, pp. 22328–22370, 2019, doi: 10.1109/access.2019.2896108.

[23] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A Survey of Distributed Consensus Protocols for Blockchain Networks," *IEEE Commun. Surv. Tutorials*, vol. 22, no. 2, pp. 1432–1465, 2020, doi: 10.1109/COMST.2020.2969706.

[24] U. Bodkhe, D. Mehta, S. Tanwar, P. Bhattacharya, P. K. Singh, and W. C. Hong, "A Survey on Decentralized Consensus Mechanisms for Cyber Physical Systems," *IEEE Access*, vol. 8, pp. 54371–54401, 2020, doi: 10.1109/ACCESS.2020.2981415.

[25] M. Ahmed, I. Elahi, M. Abrar, U. Aslam, I. Khalid, and M. A. Habib, "Understanding Blockchain: Platforms, Applications and Implementation Challenges," 2019, doi: 10.1145/3341325.3342033.

[26] G. Wood and others, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Proj. yellow Pap.*, vol. 151, no. 2014, pp. 1–32, 2014.

[27] E. Androulaki *et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the thirteenth EuroSys conference*, 2018, pp. 1–15.

[28] G. Greenspan, "Multichain private blockchain-white paper," *URl http//www. multichain. com/download/MultiChain-White-Paper. pdf*, 2015, [Online]. Available: https://www.multichain.com/download/MultiChain-White-Paper.pdf.

[29] S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han, and F. Wang, "Blockchain-Enabled Smart Contracts: Architecture, Applications, and Future Trends," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 49, no. 11, pp. 2266–2277, 2019, doi: 10.1109/TSMC.2019.2895123.

[30] S. Wang, Y. Yuan, X. Wang, J. Li, R. Qin, and F. Wang, "An Overview of Smart Contract: Architecture, Applications, and Future Trends," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 108–113, doi: 10.1109/IVS.2018.8500488.

[31] M. Crosby, Nachiappan, P. Pattanayak, S. Verma, and V. Kalyanaraman, "Blockchain Technology - BEYOND BITCOIN," *Berkley Eng.*, 2016, doi: 10.1515/9783110488951.

[32] R. V. Rosa and C. E. Rothenberg, "Blockchain-Based Decentralized Applications Meet Multi-Administrative Domain Networking," in *Proceedings of the ACM SIGCOMM 2018 Conference on Posters and Demos*, 2018, pp. 114–116, doi: 10.1145/3234200.3234217.

[33] S. Wang, X. Zhu, and S. Zhao, "Blockchain-Based SDN Security Guarantee Model," in *International Conference on Communication Technology Proceedings, ICCT*, 2019, pp. 1296–1300, doi: 10.1109/ICCT46805.2019.8947081.

[34] D. B. Rawat, "Fusion of Software Defined Networking, Edge Computing, and Blockchain Technology for Wireless Network Virtualization," *IEEE Commun. Mag.*, vol. 57, no. 10, pp. 50–55, 2019, doi: 10.1109/MCOM.001.1900196.

[35] G. Zyskind, O. Nathan, and A. '. Pentland, "Decentralizing Privacy: Using Blockchain to Protect Personal Data," in *2015 IEEE Security and Privacy Workshops*, 2015, pp. 180–184, doi: 10.1109/SPW.2015.27.

[36] S. H. Hashemi, F. Faghri, and R. H. Campbell, "Decentralized user-centric access control using pubsub over blockchain," *arXiv Prepr. arXiv1710.00110*, 2017.

[37] S. H. Hashemi, F. Faghri, P. Rausch, and R. H. Campbell, "World of Empowered IoT Users," in *2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI)*, 2016, pp. 13–24, doi: 10.1109/IoTDI.2015.39.

[38] A. G. Abbasi and Z. Khan, "VeidBlock: Verifiable Identity Using Blockchain and Ledger in a Software Defined Network," in *Companion Proceedings of The10th International Conference on Utility and Cloud Computing*, 2017, pp. 173–179, doi: 10.1145/3147234.3148088.

[39] H. Al-Sakran, Y. Alharbi, and I. Serguievskaia, "Framework Architecture for Securing IoT Using Blockchain, Smart Contract and Software Defined Network Technologies," in *2019 2nd International Conference on new Trends in Computing Sciences (ICTCS)*, 2019, pp. 1–6, doi: 10.1109/ICTCS.2019.8923080.

[40] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, Q. Zhang, and K.-K. R. Choo, "An Energy-Efficient SDN Controller Architecture for IoT Networks With Blockchain-Based Security," *IEEE Trans. Serv. Comput.*, vol. 13, no. 4, pp. 625–638, 2020, doi: 10.1109/TSC.2020.2966970.

[41] P. K. Sharma, S. Rathore, Y. S. Jeong, and J. H. Park, "SoftEdgeNet: SDN Based Energy-Efficient Distributed Network Architecture for Edge Computing," *IEEE Commun. Mag.*, vol. 56, no. 12, pp. 104–111, 2018, doi: 10.1109/MCOM.2018.1700822.

[42] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, and K. R. Choo, "Blockchain-enabled Authentication Handover with Efficient Privacy Protection in SDN-based 5G Networks," *IEEE Trans. Netw. Sci. Eng.*, p. 1, 2019, doi: 10.1109/TNSE.2019.2937481.

[43] A. Arins, "Blockchain based Inter-domain Latency Aware Routing Proposal in Software Defined Network," in *AIEEE*, 2018, pp. 1–2.

[44] W. Hou, Z. Ning, L. Guo, and P. Guo, "SDN-based Optimizing Solutions for Multipath Data Transmission Supporting Consortium Blockchains," in *2018 International Conference on Computer, Information and Telecommunication Systems (CITS)*, 2018, pp. 1–5.

[45] P. Kamboj and S. Pal, "QoS in Software Defined IoT Network Using Blockchain Based Smart Contract: Poster Abstract," in *SenSys'19*, 2019, pp. 430–431, doi: 10.1145/3356250.3361954.

[46] Y. E. Oktian, E. N. Witanto, S. Kumi, and S. Lee, "ISP Network Bandwidth Management: Using Blockchain and SDN," in *ICTC*, 2019, pp. 1330–1335.

[47] G. R. Carrara, L. H. A. Reis, C. V. N. Albuquerque, and D. M. F. Mattos, "A Lightweight Strategy for Reliability of Consensus Mechanisms based on Software Defined Networks," in *2019 Global Information Infrastructure and Networking Symposium (GIIS)*, 2019, pp. 1–6.

[48] P. Wang, X. Liu, J. Chen, Y. Zhan, and Z. Jin, "QoS-Aware Service Composition Using Blockchain-Based Smart Contracts," in *Proceedings of the 40th International Conference on Software Engineering (ICSE): Companion Proceeedings*, 2018, pp. 296–297, doi: 10.1145/3183440.3194978.

[49] E. Ak and B. Canberk, "BCDN: A proof of concept model for blockchain-aided CDN orchestration and routing," *Comput. Networks*, vol. 161, pp. 162–171, 2019, doi: 10.1016/j.comnet.2019.06.018.

[50] J. Yang, S. He, Y. Xu, L. Chen, and J. Ren, "A Trusted Routing Scheme Using Blockchain and Reinforcement Learning for Wireless Sensor Networks," *Sensors*, vol. 19, no. 4, p. 970, 2019, doi: 10.3390/s19040970.

[51] M. Saad, A. Anwar, A. Ahmad, H. Alasmary, M. Yuksel, and A. Mohaisen, "Routechain: Towards blockchain-based secure and efficient bgp routing," in *ICBC 2019 - IEEE International Conference on Blockchain and Cryptocurrency*, 2019, pp. 210–218, doi: 10.1109/BLOC.2019.8751229.

[52] G. Ramezan and C. Leung, "A Blockchain-Based Contractual Routing Protocol for the Internet of Things Using Smart Contracts," *Wirel. Commun. Mob. Comput.*, vol. 2018, 2018, doi: 10.1155/2018/4029591.

[53] G. S. Aujla, M. Singh, A. Bose, N. Kumar, G. Han, and R. Buyya, "BlockSDN: Blockchain-as-a-Service for Software Defined Networking in Smart City Applications," *IEEE Netw.*, vol. 34, no. 2, pp. 83–91, 2020, doi: 10.1109/mnet.001.1900151.

[54] M. Azab, R. R. Ergawy, E. M. Ghourab, A. Mokhtar, and M. Rizk, "Towards Blockchain-based Multi-controller Managed Switching for Trustworthy SDN Operation," in *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, 2019, pp. 991–998.

[55] R. M. A. Ujjan, Z. Pervez, and K. Dahal, "Snort Based Collaborative Intrusion Detection System Using Blockchain in SDN," in *2019 13th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, 2019, pp. 1–8, doi: 10.1109/SKIMA47702.2019.8982413.

[56] Z. Abou El Houda, A. S. Hafid, and L. Khoukhi, "Cochain-SC: An Intra-and Inter-Domain Ddos Mitigation Scheme Based on Blockchain Using SDN and Smart Contract," *IEEE Access*, vol. 7, pp. 98893–98907, 2019, doi: 10.1109/ACCESS.2019.2930715.

[57] M. Steichen, S. Hommes, and R. State, "ChainGuard — A firewall for blockchain applications using SDN with OpenFlow," in *2017 Principles, Systems and Applications of IP Telecommunications (IPTComm)*, 2017, pp. 1–8, doi: 10.1109/IPTCOMM.2017.8169748.

[58] Z. A. El Houda, L. Khoukhi, and A. Hafid, "ChainSecure - A Scalable and Proactive Solution for Protecting Blockchain Applications Using SDN," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6, doi: 10.1109/GLOCOM.2018.8647279.

[59] D. Kreutz, F. M. V Ramos, and P. Verissimo, "Towards Secure and Dependable Software-Defined Networks," in *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, 2013, pp. 55–60, doi: 10.1145/2491185.2491199.

[60] S. Ding, G. Shen, K. X. Pan, S. K. Bose, Q. Zhang, and B. Mukherjee, "Blockchain-Assisted Spectrum Trading between Elastic Virtual Optical Networks," *IEEE Netw.*, vol. 34, no. 6, pp. 205–211, 2020, doi: 10.1109/MNET.011.2000138.

[61] D. Marconett and S. J. B. Yoo, "FlowBroker: A Software-Defined Network Controller Architecture for Multi-Domain Brokering and Reputation," *J. Netw. Syst. Manag.*, vol. 23, no. 2, pp. 328–359, 2015, doi: 10.1007/s10922-014-9325-5.

[62] V. Kotronis *et al.*, "Stitching Inter-Domain Paths over IXPs," in *Proceedings of the Symposium on SDN Research*, 2016, pp. 17:1--17:12, doi: 10.1145/2890955.2890960.

[63] A. Gupta *et al.*, "SDX: A Software Defined Internet Exchange," in *Proceedings of the 2014 ACM Conference on SIGCOMM*, 2014, pp. 551–562, doi: 10.1145/2619239.2626300.

[64] Z. A. El Houda, A. Hafid, and L. Khoukhi, "Co-IoT: A Collaborative DDoS Mitigation Scheme in IoT Environment Based on Blockchain Using SDN," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6, doi: 10.1109/GLOBECOM38437.2019.9013542.

[65] S. Rathore, B. Wook Kwon, and J. H. Park, "BlockSecIoTNet: Blockchain-based decentralized security architecture for IoT network," *J. Netw. Comput. Appl.*, vol. 143, pp. 167–177, 2019, doi: 10.1016/j.jnca.2019.06.019.

[66] P. K. Sharma, S. Singh, Y. Jeong, and J. H. Park, "DistBlockNet: A Distributed Blockchains-Based Secure SDN Architecture for IoT Networks," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 78–85, 2017, doi: 10.1109/MCOM.2017.1700041.

[67] J. Gao *et al.*, "A Blockchain-SDN-Enabled Internet of Vehicles Environment for Fog Computing and 5G Networks," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4278–4291, 2020, doi: 10.1109/JIOT.2019.2956241.

[68] L. Xie, Y. Ding, H. Yang, and X. Wang, "Blockchain-Based Secure and Trustworthy Internet of Things in SDN-Enabled 5G-VANETs," *IEEE Access*, vol. 7, pp. 56656–56666, 2019, doi: 10.1109/ACCESS.2019.2913682.

[69] Y. Yahiatene and A. Rachedi, "Towards a Blockchain and Software-Defined Vehicular Networks Approaches to Secure Vehicular Social Network," in *2018 IEEE Conference on Standards for Communications and Networking (CSCN)*, 2018, pp. 1–7, doi: 10.1109/CSCN.2018.8581756.

[70] R. Chaudhary, A. Jindal, G. S. Aujla, S. Aggarwal, N. Kumar, and K.-K. R. Choo, "BEST: Blockchain-based Secure Energy Trading in SDN-enabled Intelligent Transportation System," *Comput. Secur.*, vol. 85, pp. 288–299, 2019, doi: https://doi.org/10.1016/j.cose.2019.05.006.

[71] A. Jindal, G. S. Aujla, and N. Kumar, "SURVIVOR: A Blockchain based Edge-as-a-Service Framework for Secure Energy Trading in SDN-enabled Vehicle-to-Grid Environment," *Comput. Networks*, vol. 153, pp. 36–48, 2019, doi: https://doi.org/10.1016/j.comnet.2019.02.002.

[72] S. Alansari, F. Paci, and V. Sassone, "A Distributed Access Control System for Cloud Federations," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, 2017, pp. 2131–2136, doi: 10.1109/ICDCS.2017.241.

[73] F. Ye, Z. Zheng, C. Chen, and Y. Zhou, "DC-RSF: A Dynamic and Customized Reputation System Framework for Joint Cloud Computing," in *2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2017, pp. 275–279, doi: 10.1109/ICDCSW.2017.21.

[74] P. Fernando and J. Wei, "Blockchain-Powered Software Defined Network-Enabled Networking Infrastructure for Cloud Management.," in *2020 IEEE 17th Annual Consumer Communications Networking Conference (CCNC)*, 2020, pp. 1–6, doi: arXiv:1909.01851v1.

[75] S. Boukria, M. Guerroumi, and I. Romdhani, "BCFR: Blockchain-based Controller Against False Flow Rule Injection in SDN," in *2019 IEEE Symposium on Computers and Communications (ISCC)*, 2019, pp. 1034–1039.

[76] X. Lu *et al.*, "Blockchain-Based Distributed Energy Trading in Energy Internet: An SDN Approach," *IEEE Access*, vol. 7, pp. 173817–173826, 2019, doi: 10.1109/access.2019.2957211.

[77] H. Yang, Y. Li, S. Guo, J. Ding, Y. Lee, and J. Zhang, "Distributed Blockchain-Based Trusted Control with Multi-Controller Collaboration for Software Defined Data Center Optical Networks in 5G and Beyond," in *2019 Optical Fiber Communications Conference and Exhibition (OFC)*, 2019, pp. 1–3.

[78] K. Christidis and M. Devetsikiotis, "Blockchains and Smart Contracts for the Internet of Things," *IEEE Access*, vol. 4. pp. 2292–2303, 2016, doi: 10.1109/ACCESS.2016.2566339.

[79] S. Huh, S. Cho, and S. Kim, "Managing IoT devices using blockchain platform," in *2017 19th International Conference on Advanced Communication Technology (ICACT)*, 2017, pp. 464–467, doi: 10.23919/ICACT.2017.7890132.

[80] C. Wu, Z. Liu, D. Zhang, T. Yoshinaga, and Y. Ji, "Spatial Intelligence toward Trustworthy Vehicular IoT," *IEEE Commun. Mag.*, vol. 56, no. 10, pp. 22–27, 2018, doi: 10.1109/MCOM.2018.1800089.

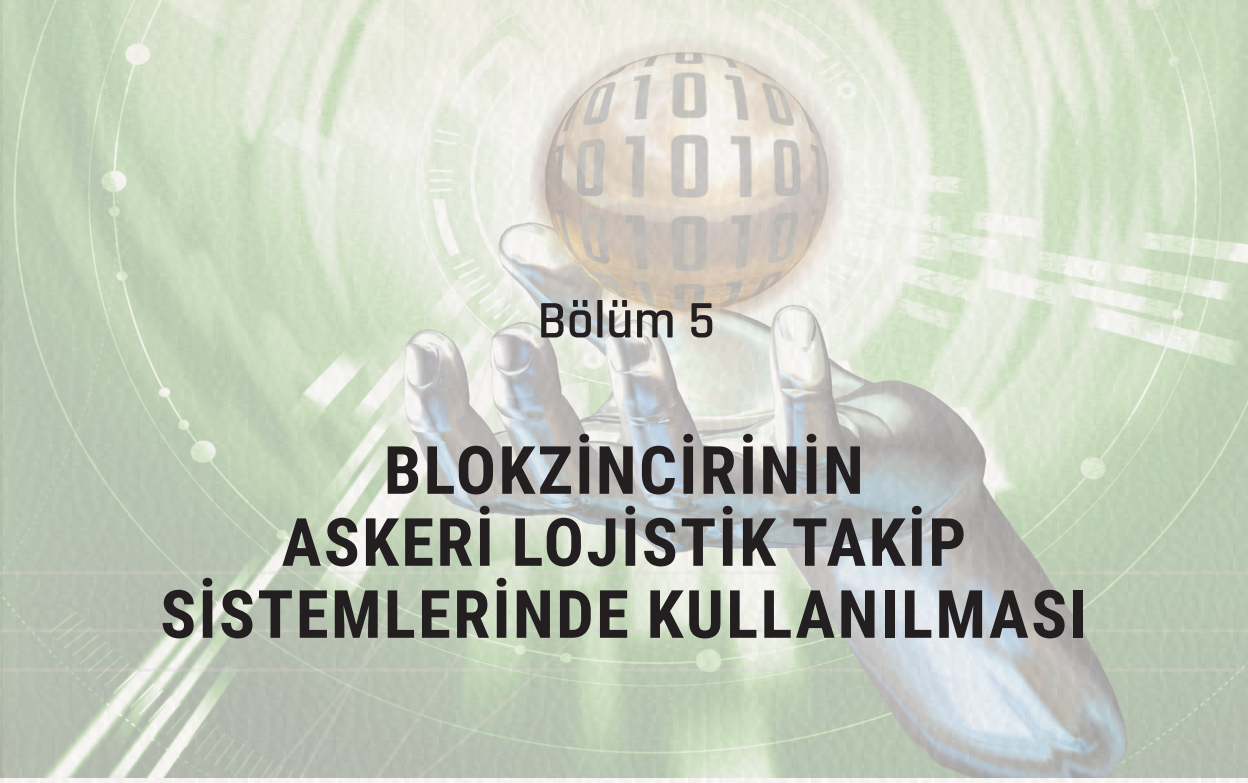
[81] S. Patidar, D. Rane, and P. Jain, "A Survey Paper on Cloud Computing," in *2012 Second International Conference on Advanced Computing Communication Technologies*, 2012, pp. 394–398, doi: 10.1109/ACCT.2012.15.

[82] Argela Networks. https://www.argela.com.tr/. Erişim Tarihi: 28 Haziran 2021.

[83] Ulak Haberleşme. https://www.ulakhaberlesme.com.tr/index.php/tr/. Erişim Tarihi: 28 Haziran 2021.

[84] Blokzincir Araştırma Laboratuvarı (BZLab). https://blokzincir.bilgem.tubitak.gov.tr/bzlab.html. Erişim Tarihi: 28 Haziran 2021.

[85] Blockchain Türkiye Platformu. https://bctr.org/. Erişim Tarihi: 28 Haziran 2021.



Bölüm 5

BLOKZİNCİRİNİN ASKERİ LOJİSTİK TAKİP SİSTEMLERİNDE KULLANILMASI

Enis Konacaklı - Enis Karaarslan

Blokzincir teknolojisi ilk olarak kripto paraların mimari altyapısı olarak tanınmıştır. Akıllı sözleşmelerin/anlaşmaların kullanılmaya başlamasıyla finans sektöründen lojistik yönetimine kadar pek çok alanda kullanım imkânı bulmuştur. Bu teknolojinin kuvvetli, güvenli, denetlenebilir ve daha sonra kanıtlanabilir kayıtlara ihtiyaç duyulan Hava Kuvvetleri bünyesinde silah ve uçak teçhizatlarının lojistik yönetiminde kullanılması mümkündür. Bu kitap bölümünde; blokzinciri teknolojisi kullanılarak silah, uçuşu destekleyen teçhizat ve uçak ana parçalarının lojistik kayıtlarının oluşturulabileceği ve takip edilebileceği bir model sunulmuş, bu modelin askeri lojistik takip ve uygulamalarında kullanılabilirliği gösterilmiş, bununla ilgili olarak değerlendirmede bulunulmuş ve gelecekte yapılabilecek çalışmalarla ilgili çıkarımlarda ve değerlendirmelerde bulunulmuştur.

5.1. GİRİŞ

Blokzinciri, kripto paralarda kazandığı ün ve sağladığı güvenlik kolaylıkları sayesinde son on yıl içerisinde üzerinde en çok araştırma yapılan teknolojilerden biri olmuştur. Nakamoto'nun ortaya koyduğu teorinin ilk uygulaması Bitcoin [1] ve sonrasından bu teknolojiyi örnek alan diğer kripto para birimleri sayesinde herkes tarafından kullanılabilir bir teknoloji haline gelmiştir. Akıllı sözleşmelerin blokzincir yapısına entegre edilmesi ile finans sektöründen lojistik yönetimine kadar pek çok alanda kullanım imkânı bulmaktadır.

Blokzinciri veriyi ve güvenliğinin sağlanmak üzere farklı bir yaklaşım sergileyen dağıtık kayıt defteri mimarisini kullanır. Bu mimari aynı zamanda, graf ve dağıtık özet tablosu gibi veri yapılarında kullanılmaktadır [2,3]. Blokzincir türevleri olarak da anılan bu tür yapılarda da dağıtık kayıt defteri süreci işletilmektedir. Blokzincirini diğer güvenli sayısal mimarilerden üstün kılan en temel özellik, sahip olduğu kayıt defterine giren bilgilerin döngüsel olarak devamlı kriptolanması ile matematiksel olarak kırılması çok zor kriptografik bir sayısal zincir oluşturmasıdır. Her blok kendinden bir önceki bloğun özet değerini kullanarak kapatılır. Bu yapısı ile blokzinciri kayıt defterinin silinmesinin ve değiştirilmesinin çok zor olduğu kabul edilir.

Lojistik sistem kayıtları terminaller tarafından oluşturulan, güvenli bir şekilde saklanan, işlenmek üzere analiz edilebilen veri kayıtlarından oluşur. Bu sistemlerin sağlıklı çalışması stok kontrol ve dağıtım süreçleri için büyük önem arz eder. Silinmez ve inkar edilemez kayıtların oluşturulmasına büyük ihtiyaç duyulur. Lojistik yönetim sistemlerinin bu nitelikleri, blokzinciri çalışma prensiple ile büyük benzerlik gösterir. Bu sebeple blokzinciri teknolojisinin lojistik yönetim süreçlerinde kullanımını ele alan pek çok çalışma bulunmaktadır. Bu çalışmalardan bazıları günümüzde projelendirilmiş, bir kısmı ise aktif olarak kullanılmaya başlanmıştır. IBM'in Farmer Connect uygulaması ile Dammam ve Rotterdam limanlarında kullanılan blokzinciri tabanlı lojistik takip sistemleri bu teknolojinin hayata geçirilmiş en başarılı örneklerinden bazıları olarak gösterilebilir.

Hava gücü elastik bir kuvvet yapısındadır. Bu özelliği sayesinde 7 gün 24 saat mekandan bağımsız olarak harekata hazır bir kuvvet olma özelliği gösterir. Otomatize edilmiş ve ihtiyaçlara hızlı cevap verebilen bir lojistik yönetim yapısının ihtiyaç duyar. Yüksek ar-ge, geliştirme, üretim ve temin maliyetlerine sahip bu sistemlerin lojistik yönetiminde yaşanabilecek herhangi bir aksama hava harekatını olumsuz etkileyebilir, operasyonun başarısızlığına sebep olabilecek zincirleme bir etki oluşturabilir. Uygun şartlarda depolanmayan ve nakli sağlanmayan lojistik malzeme kullanılamaz hale getirebilir. Bu sistemlerin depolanma ve nakil işlemlerinin ne zaman ve hangi şartlarda gerçekleştirildiği daha sonra hukuki olarak değerlendirmeye tabi tutulmak üzere çeşitli soruşturmalara konu edilebilir.

Bu kitap bölümünde; hava kuvvetleri lojistik sisteminin sahip olması gereken temel nitelikler özetlenmiş, blokzinciri teknolojisi anlatılmış, bu teknolojinin

kullanımında karşılaşılan problemler ve çözüm önerileri ele alınmış, akıllı sözleşmeler ve dağıtık kayıt defteri teknolojileri tanıtılmıştır. Ayrıca, blokzincirinin lojistik takip uygulamalarındaki kullanım alanları hakkında bir literatür taraması yapılmıştır. Hava kuvvetleri lojistik sistemlerinin güvenliğinin arttırılarak sonradan kanıtlanabilir inkar edilemez kayıtların oluşturulabileceği blokzinciri tabanlı bir model tasarımı yapılmış ve bu model üzerinde gerçekleştirilebilecek bir kayıt örneği HyperLedger Fabric platformu kullanılarak örneklenmiştir. Sonuç olarak, önerilen modelin uygulanabilirliği ve diğer kuvvet lojistik takip sistemlerinde kullanılabilirliği ile ilgili bir değerlendirme yapılmıştır.

5.2. ASKERİ LOJİSTİK VE BİLGİ SİSTEMLERİ

Askeri lojistik, askeri harekatı destekleyecek ekipman ve malzemenin planlanmasını, üretimini, tedarikini, dağıtımını, bakım idame ve yer değiştirmesini kapsayan süreçtir. Harekatı başarıya ulaştırmak üzere ihtiyaç duyulan malzemeyi istenilen yerde, istenilen zamanda, istenilen durumda hazır bulundurma sanatı olarak da tanımlanabilir [4].

Teknolojinin hızla gelişmesi ve giderek artan entegrasyon ihtiyacı lojistik bilişim sistemlerini modern orduların vazgeçilmez bir parçası haline getirmiştir. Harekat üstünlüğünü elde etmek üzere lojistik yönetimini otomatize edilerek kaynakların süratli ve verimli bir şekilde kullanılması büyük önem arz eder. Bu sebeple günümüzde silahlı kuvvetlerin lojistik yönetiminde bilişim sistemleri etkin bir şekilde kullanılmaktadır.

Hava gücü bir harekatta belirleyici ve sonuç alıcı güç konumundadır. Çevik, hızlı, elastik ve uzun menzil özellikleriyle; taktik, operasyonel ve stratejik seviyelerde harekatın sonucunu değiştirebilecek çok önemli bir etki alanına sahiptir. Yakından uzağa doğru verilen tüm görevleri icra edebilir. Tüm hedeflere süratli ve hatasız bir şekilde nüfuz edebilir. Tüm kuvvet yapılarında büyük öneme sahip olan lojistik, yüksek hızda karar verme ve 7 gün 24 saat mekandan bağımsız olarak harekata hazır bir kuvvet olma özelliği gösteren hava kuvveti açısından hatasız yönetilmesi gereken bir süreçtir. Harekatı süratli ve sürekli destekleyebilmesi için hava kuvveti yüksek hızlı ve güvenilir bir lojistik sistemine ihtiyaç duyar.

Başarılı bir lojistik yönetim bilgi sistemi ihtiyaçları mevcut sarfiyat durumunu göz önünde bulundurarak öngörebilecek kabiliyette olmalıdır. Merkezi yö-

netimi desteklemeli ani çıkan ihtiyaçları hızlı bir şekilde karşılayabilecek bir altyapı sağlayabilmelidir. Yüksek ar-ge, geliştirme, üretim ve temin maliyetlerine sahip hava sistemlerinin lojistik yönetiminde yaşanabilecek herhangi bir aksama hava harekatını olumsuz etkileyebilir. Operasyonun başarısızlığına sebep olabilecek zincirleme bir etki oluşturabilir. Uygun şartlarda depolanmayan ve nakli sağlanmayan lojistik malzeme kullanılamaz hale getirebilir. Bu sistemlerin depolanma ve nakil işlemlerinin ne zaman ve hangi şartlarda gerçekleştirildiği daha sonra hukuki olarak değerlendirmeye tabi tutulmak üzere çeşitli soruşturmalara konu edilebilir. Bu sistemlerin başarılı bir şekilde hizmet edebilmeleri için tasarımlarında yukarıda belirtilen kriterlerin göz önünde bulundurulması büyük önem arz eder.

5.3. BLOKZİNCİRİ

Blokzinciri kayıtları ardışık şekilde özet (hash) değerleri ile bağlandığından, silinmesi ve değiştirilmesi matematiksel olarak çok zor olan bir dağıtık kayıt defteri teknolojisidir. Nakamoto mahlası ile kaleme alınan "Bitcoin: A peer-to-peer electronic cash system" [1] isimli yayın ile ilk olarak bilim dünyasına tanıtılmış lider kripto para birimi Bitcoin ile ilk kez uygulamaya geçirilmiştir. Kripto para birimlerinin temel mimari yapısını oluşmak üzere kullanılırken Ethereum Blokzinciri Platformu ile birlikte ilk kez blokların içerisine önceden belirlenmiş kuralları işleyen kodlar yazılmaya başlanmıştır. Akıllı sözleşme (smart contract) adı verilen bu yenilik ile çok farklı disiplinlerde kullanım alanı bulmaya başlamıştır [2,3].

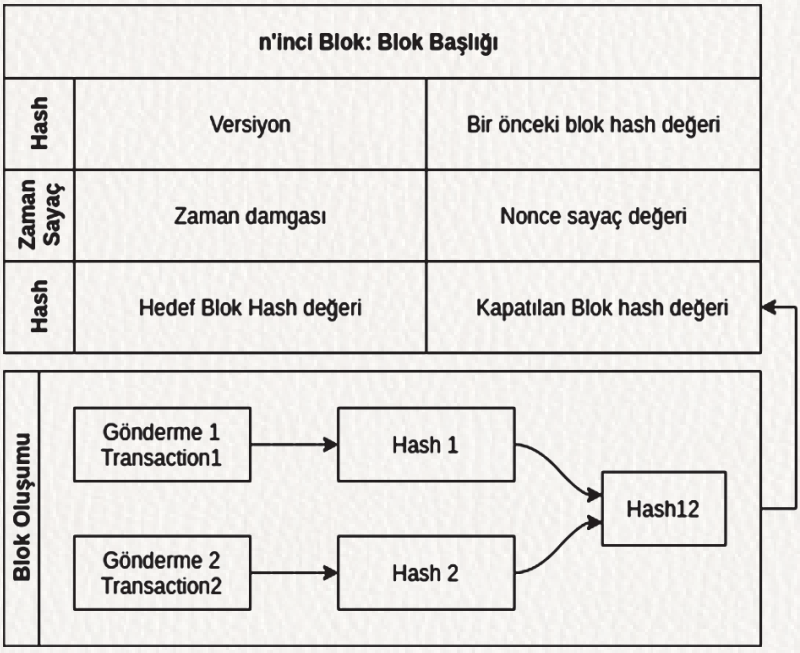
Blokların kaydının tutulduğu yapılar kayıt defteri (ledger) olarak adlandırılır. En kısa anlatımı ile gerçekleştirilen gönderme işlemleri (transaction) toplanarak bir blok oluşturacak şekilde sıralanır. Kriptografik teknikler kullanılarak bu bloklar ardışık şekilde birbirine bağlanır. Böylece birbirini devamlı olarak takip eden ardışık ve güvenilir bir zincir yapısı oluşturulur. Kriptolama ve blokları kapatma işlemi ile bu işlemlerin yönetimi süreci blokzinciri tipi ve uzlaşma (konsensüs) protokolüne göre değişiklik gösterecektir.

Blokzincirinde işlenmekte olan bir bloğun yapısı şematik olarak Şekil 5.1'de gösterilmiştir. Blok yapısı blokzincirinin tipine ve mimarisine göre değişiklik gösterebilir. Tüm yapılarda ortak olarak; versiyon bilgisi, zaman damgası, bir önceki blok özet değeri, kapatılan (bir önceki) bloğa ait özet değeri, tek sefer-

lik anahtar (nonce) değeri ve hedef özet değeri bilgilerini içerir.

Blokzincirinde kayıt defteri P2P yapıda eşler (düğümler) arasında eş zamanlı olarak tutulur. Bir eşin tuttuğu defterin aynısı diğer eşte de mevcuttur. Tutulmakta olan defterin boyutu eşlerin donanımsal özelliklerine uygun bir şekilde arasında hiyerarşik olarak dağıtılır. Eşler aynı anda hem sunucu hem de istemci olarak davranabilirler. Bu merkezi olmayan mimari dahilinde oluşturulan düğüm tipleri Şekil 5.2'de özetlenmiştir. Eşler farklı donanıma ve farklı işlevlere sahip olabilir [4]. Defterin ne kadarının hangi eşte tutulacağı sahip olduğu donanım kapasitelerine göre belirlenir. Verilen örnek şekilde de anlatıldığı üzere; kullanılacak donanımın tam düğüm, masaüstü bilgisayarlarının yarım düğüm, mobil cihazların basit düğüm olarak nitelendirildiği görülmektedir. Bu yapı eşlere kapasitesine göre yük bindirilmesini ve maliyetlerin dengeli olarak dağıtılmasını sağlar. Uygulamada, tutulmakta olan en düşük defter uzunluğu dahi eşin güvenliğini sağlamak için yeterli boyuttadır.

Yeni bloğu hangi düğümün yazacağı, madencilik uygulamalarında hangi düğümün blok ödülünü hak edeceği gibi durumlar; önceden belirlenmiş kuralları içeren uzlaşma protokolleri (PoW, PoS, RAFT, IBFT, vb.) ile belirlenir [2].



Şekil 5.1. Blok Yapısı

Şekil 5.2. Hiyerarşik Düğüm Yapısı

Gerçekleştirilmesi planlanan bir projede hangi blokzinciri tipinin kullanılacağı, tasarım aşamasından önce planlanmalıdır. Yararlanılmak istenilen kolaylıklar, kullanılacak blokzincir tipi ve mimarisini planlayabilmek için belirleyici kriterdir. İzinsiz blokzincir yapılarının en bilinen örneği Bitcoin'dir. Bu tip yapılarda kullanım kolaylığı için kayıtları okuma ve düğüm olma hakkı her katılımcıya sağlanır. Bu tip kripto para birimlerinde herhangi bir kullanıcı ağa dahil olmak için herhangi bir kısıtlamaya uğramaz. Kullanıcıların kimlikleri niteliğindeki kullanıcı numaraları kendi anahtarları olarak işlem görür. Bu yapılarda maliyetler, enerji sarfiyatı çok yüksektir. Ölçeklenebilirlik ise oldukça düşüktür. Kullanılan blokzinciri tipinden istenen nitelik kapsamında gizlilik ve mahremiyetten bazı durumlarda taviz verilebilir.

Blokzincirinin farklı amaçlarla kullanıldığı kurumsal çözümlerde daha çok özel izinli veya hibrit yapılar tercih edilir. Özel izinli yapıların oluşturulmasında amaç, ağda tanımlı eşlerin belirli bir bilgiyi tam güven içerisinde paylaşmasıdır. Eşler ancak belirlenmiş olan otoritenin izni ile ağa dahil olabilirler. HyperLedger Fabric, Hyperledger Burrow, R3, Corda bu mimariye örnek verilebilecek platformlardır. Bu yapılarda gizlilik artarken maliyetler düşer. Bu durumda taraflar demokratik olmayan bir sistem içerisinde diğer eşlere uygulanan kuralları bilemeyeceğinden gizli sözleşme endişesi ortaya çıkabilir. Verim istenen amacı karşılayacak düzeylerde sağlanamayabilmektedir. Hibrit yapılar da kullanım amacına göre izinsiz ve özel izinli yapılar birlikte çalışırlar. Esnek bir kullanıma sahiptir. Bu mimari yapılarda ise birlikte çalışabilirlik (interoperability) ile ilgili problemler ortaya çıkabilmektedir.

5.3.1. Blokzincirinin Sağladığı Güvenlik Servisleri

Blokzincirinin sağladığı güvenlik servisleri Tablo 5.1'de gizlilik (confidentiality), bütünlük (integrity) ve kullanılabilirlik (availability) maddeleri altında ele alınmıştır. Sağladığı güvenliğin derecesi blokzincir yapılarında kullanılan güvenlik protokolleri ve blokzincirinin amacına uygun olarak belirlenen uzlaşma tipine göre değişiklik gösterir [3].

Tablo 5.1. Blokzincirinde Sağlanan Güvenlik Unsurları /Servisleri

Güvenlik Unsurları/ Servisleri	Sağlanan Güvenlik Unsurları Açıklamaları
Gizlilik	Blokları kapatırken kullandığı kriptolama fonksiyonu ile sağlamaktadır
Bütünlük	Yazılı veriyi bloklar arasında özet fonksiyonları ile kapatması ile sağlanmaktadır
Kullanılabilirlik	Bir veya birden çok eş ağa dâhil olmasa dahi diğer eşler üzerinden deftere erişimin sağlanabilmesi ile sağlanmaktadır
Kimlik Denetimi	Kullanılan açık anahtarlar yapısı ile sağlanmaktadır
Doğruluğun teyidi	Kullanıcı ve bilginin doğruluğunun teyidi sertifika yönetimi ve kayıt defterinin dağıtık yapısı ile sağlanır
Kanıtlanabilirlik	Sonradan kanıtlanabilirlik kayıt defterinin değiştirilemez yapısı ile sağlanmaktadır
İnkâr edilemezlik	Kayıt defterinin değiştirilemez yapısı ile sağlanmaktadır

5.3.2. Yapısal Sorunlar

Blokzinciri teknolojisinin kullanımın kolaylıkları ve sağladığı güvenlik servislerinin yanında uygulamalardaki kısıtlamalarının da uygulama alanına göre iyi analiz edilmesi gerekir. Blokzinciri uygulamaları geliştirirken göz önünde bulundurulması gereken problemler aşağıda belirtilmiştir [5]:

- **Maliyetler (Enerji tüketimi, soğutma, donanım):** Blokzincirinin özellikle madencilik yapısını kullanan protokollerde (PoW) oldukça yüksek enerji tüketmesi ve pahalı fiziksel altyapıya ihtiyaç duymasıdır.
- **Ölçeklenebilirlik ve hız:** Büyük miktardaki verinin onaylanabilmesi ve gönderilebilmesi işleminin tatmin edici ve gerekli hızda gerçekleştirilebilmesinin zorluklarıdır.
- **Birlikte çalışabilirlik:** Diğer uygulama ve sistemlerle bütünleşik ve eşzamanlı çalışabilirlik ihtiyacıdır.
- **Kuantum Sonrası:** Kuantum hesaplama çağında kriptoloji zorlukları ve buna göre tasarım yapma ihtiyacıdır.
- **Mahremiyet:** Uygulamanın alanına göre istenen mahremiyet ve gizliliğin mimari yapı oluşturulmadan önce tanımlanmasının gerekliliğidir.

Bazı zafiyetler blokzincirinin tipine göre değişiklik göstermekle birlikte önemli güvenlik zafiyetleri aşağıda maddeler halinde özetlenmiştir:

- **Gönderme sünmesi gecikmesi:** Özellikle kontrata bağlı göndermelerde kontrat şartlarının tam karşılanmaması durumunda göndermedeki gecikme olarak ifade edilir,
- **Ağ güvenliği:** He ne kadar güvenli bir yapı sağlasa da, farklı blokzinciri platformlarının farklı güvenlik açıkları bulunmaktadır. Örneğin; ağdaki hatanın bazen bencil madencilik (selfish mining) veya çift ödemeye sebebiyet verdiği görülebilir.
- **Mahremiyet:** Açık blokzincir yapılarında her eşin izin almadan ağa dahil olması sağlanırken gizlilik ve mahremiyetten ödün verilmek zorunda kalınabilir. Bu durum kullanılan blokzincir mimarisinin tasarım amacına ve ağ tipine göre bir problem olarak değerlendirilebilir. Bu durumda blokzincirinde kişisel verilerin tutulmamasını sağlayarak ve sıfır bilgi kanıt (zero knowledge proof) protokolleri kullanarak bu zafiyetler aşılabilir.

- **Akıllı sözleşmelerdeki zafiyetler:** Akıllı sözleşmelerdeki hatalı kodlar (bug) yanlış işlemlere sebebiyet verebilirler.
- **Suç Aktiviteleri:** Bitcoin ve özellikle mahremiyet tabanlı Monero gibi diğer kripto para birimleri harcama takibinin zorlaştırdığından bunlar suç unsurlarını finanse etmek için kullanılabilir.

Blokzincir uygulanacak modelde; hız, mahremiyet ve güvenlik gereksinimleri göz önünde bulundurularak kullanılacak güvenlik ve uzlaşma protokolleri belirlenmelidir. Bu durumun dikkate alınmaması durumunda verimliliğin düşebileceği veya modelin iyi bir şekilde gerçeklenememesi gibi sorunlarla karşı karşıya kalınabileceği unutulmamalıdır.

5.3.3. Blokzincirine Gerçekleştirilebilecek Saldırılar

Blokzinciri teknolojisi sağlam bir güvenlik altyapısı sunmaktadır ama hiçbir sistem %100 güvenli olmadığı gibi, blokzincirinde de çeşitli saldırılar ve olası uygulama hataları bulunabilmektedir. Özellikle blokzinciri altyapısını kullanan kripto varlıkların (crypto currency) yüksek değerler kazanmasıyla bu altyapılar da kötü niyetli aktörlerin hedefidir. Karşılaşılan veya teorik olarak geliştirilebileceği değerlendirilen atak tiplerini öncelikle kripto varlıklara özgü ve genel saldırılar olarak iki ayrı kategoride toplamak mümkündür [5]. Kripto varlıklara ve PoW uzlaşma protokolü kullanan ağlara özgü ve genel saldırıların başlıcaları aşağıda özetlenmiştir [23]:

- **%51 Saldırısı:** Bir organizasyonun veya kişinin PoW tabanlı blokzinciri altyapısının (ağ madenciliğinin) büyük çoğunluğunu kontrol edecek grafik işlemci gücüne (GPU) sahip olmasıdır. Bu durumda bu kişi sistemi kendi çıkarlarına göre kullanabilir, işlemlerin kabul edilip edilmemesine hükmedebilir. Bu durumda sistemin güvenilirliği kalmayacaktır. Herhangi bir otoritenin böyle bir güçle sistemde olmaması sağlanmalıdır.
- **Çift Harcama Saldırısı (Double Spending):** Kişinin eş zamanlı olarak iki kripto para harcaması yaparak sistemi yanıltma girişimidir. Bu tip saldırıları önlemek üzere; blokzincir ağında yalnızca harcanmayan kripto paranın, son göndermeyi takip eden işlemde kullanıldığından emin olunması gereklidir. Bu sebeple tüm işlemlerin madenci ağı dahilinde belirlenmiş kurallar dahilinde doğrulanması ve kayıt defterine işlenmesi önem taşır.

- **Bencil Madencilik (Selfish Mining):** Bazı madencilerin diğer madencileri aldatılarak, kaynaklarını blok bulma dışında harcamaları için yönlendirdikleri saldırıdır. Bu sayede kendi hak ettiklerinden daha büyük bir ödül elde edeceklerdir. Bu saldırıya karşı önlem olarak önerilen ZeroBlock tekniği veya işlenemez zaman damgaları (unforgeable timestamps) kullanılabilmektedir. Diğer bir savunma yöntemi olarak, iki veya daha fazla madenciyi eşit uzunlukta blokları çözmek üzere yönlendirilerek çıkar çatışması yaratan böylece bencil madenciyi tespit etmeyi amaçlayan DECOR+ önerilmektedir.
- **Denge Saldırısı:** Teorik olarak dengelenmiş madencilik gücüne sahip madenci alt grupları arasındaki ağ iletişimini geciktirmeyi hedefleyen saldırıdır.
- **Canlılık (Liveness) Saldırısı:** İşlem onay süresini geciktirmek için gerçekleştirilen bir saldırıdır. Saldırgan dürüst eşlere karşı kendi özel zincirlerini oluşturmak üzere bir avantaj elde etmeye ve eşler arası güven ilişkisine karşı ağda avantaj sağlamaya çalışır. Daha sonra işlemi içeren gerçek bloğu geciktirmek üzere gelen işlemleri reddeder. Blokzinciri oluşturma sürecini ve ağın hızını yavaşlatır. Saldırı ilerledikçe ağda yaratılan etki zincirleme olarak genişler.
- **Özel Anahtar Saldırısı:** Normal şartlarda özel anahtar blokzincirine erişimini kendi hesabı üzerinden sahibine açan anahtar rolündedir. Bu anahtarın kaybedilmesi veya silinmesi verinin kaybolmasına sebebiyet verir.
- **Gönderme Mahremiyet İhlali:** Kripto para borsası gibi aracı kurumların durumunda; kullanıcı ihmali veya hatası veya aracı kurum sebebi ile mahremiyet ihlal edilebilir. Merkezi olmayan borsaların da sayısı az olmakla birlikte var olduklarını da belirtelim, buralardan sadece akıllı sözleşme destekleyen kripto paralar edinilebilmektedir.
- **Sybil Saldırısı:** Sistemi yavaşlatmak veya çökertmek için sahte düğümlerin kullanıldığı saldırıdır. Bunu da ağı meşgul etmek için yanlış bilgi göndererek gerçekleştirirler. Bu saldırının sadece teoride planlanabileceği, sanal düğümlerin yoğun hesaplama yapamayacağı öngörüsünden yola çıkılarak bugünkü teknoloji ile gerçekleştirilemeyeceği değerlendirilmektedir [23].

- **DAO Saldırısı:** Özel bir Ethereum uygulaması olan DAO ağına gerçekleştirilen bir zararlı kod içeren akıllı sözleşme atağıdır. Saldırgan 18 Ocak 2016'da ağda mevcut 3.6 milyon Ether'i oluşturduğu yeni DAO ağına aktarmıştır. Bu saldırı genel Ethereum ağını etkilememekle birlikte bir blokzincir ağına gerçekleştirilen en önemli saldırılardan biri niteliğindedir.

5.4. LİTERATÜRDEKİ ÇALIŞMALAR

Blokzincirinin lojistik ve taşımacılık alanlarında kullanımına yönelik pek çok akademik çalışma ve uygulama mevcuttur. Literatürde var olan çalışmalar son zamanlarda yayımlanmış (2020 yılı) ve en az bir kez farklı bir kaynakta referans gösterilmiş olma şartı ile on yedi çalışma seçilerek kullanılmıştır.

Batta vd. (2020) tarafından yapılan çalışmada blokzinciri teknolojisinin lojistik ve taşımacılık yönetim süreçlerine uygulanmasına dair çalışmalar incelenmiştir. Bu kapsamda blokzincirinin bu süreçlerdeki kullanım potansiyeli dahilinde henüz yeterince araştırılmadığı ve gelecekte bu sektörün blokzincirine daha fazla entegre edilebileceği şeklinde çıkarımda bulunulmuştur [6].

Johnson vd. (2020), çalışmasında blokzinciri teknolojisinin soğuk zincir ile taşımacılık hizmeti veren şirketlerin lojistik yönetim süreçlerindeki uygulama alanlarını incelemiş, bu teknolojinin lojistik endüstrisi ve iş modellerine etkisine ilişkin çıkarımlarda bulunmuştur [7].

Müssigmann vd. (2020), gelecekte yapılacak araştırmalara bir temel oluşturmak üzere 2016-2020 arasında yazılmış olan ve blokzinciri teknolojisinin lojistik ve tedarik zinciri süreçlerine uygulanması konusundaki 613 akademik çalışmadan alınan verileri içeren bir bibliyometrik analiz sunmuştur [8].

Tan vd. (2020), blokzinciri tabanlı bir yeşil lojistik (green logistic) takip sistemi modellemiştir [9]. Sensör olarak kullanılacak nesnelerin interneti (Internet of things - IoT) cihazları fiziksel katmanda modele entegre edilmiş ve bu sistemin sağlayabileceği faydalara değinilmiştir. Bu kapsamda karşılaşılabilecek kısıtlamalar ve gelecek uygulamalarda sistemin nasıl geliştirilebileceği üzerinde durulmuştur.

Kifokeris ve Koch (2020), materyal, ekonomik akış (para akışı) ve verinin birbirine entegre edildiği blokzinciri tabanlı bir sistem önermiştir. Ardından

bu modeli "swimlane" süreç akış diyagramında okuyucuya görsel olarak sunmuşlardır. Blokzincirinin IoT cihazlar ile entegrasyonu sayesinde bu alanda verimli sonuçlar elde edilebileceğini vurgulamışlardır [10].

Irannezhad (2020), limanlarda kullanılan lojistik iş yönetim süreçlerinde (port community system-PCS) işlem maliyetini düşürmek için blokzinciri tabanlı bir çözüm sunmuştur. Buna ek olarak daha önce ilgili konularda yapılmış çalışmalarla ilgili bir literatür taraması gerçekleştirilmiştir [11].

Humayun vd. (2020) tarafından yapılan çalışmada akıllı ulaşım ve lojistiğin bir ülkenin gelişimindeki önemine değinilmiştir. IoT ve blokzinciri entegrasyonu ile akıllı ulaşım ve lojistikte kullanılabilecek şeffaf, güvenli, sonradan kanıtlanabilir ve inkâr edilemezliğin sağlandığı bir model tasarlanmış ve iki kullanım alanında da bu konu örneklenmiştir. SmaTaxi uygulaması ile Dammam ve Rotterdam limanları arasındaki lojistik takibi için kullanılan blokzinciri tabanlı lojistik takip sistemine de değinilmiştir. Gelecek çalışma olarak ise akıllı ulaşım ve lojistik uygulamalarının birlikte çalıştırılabilirlik ve ölçeklendirilebilirlik açısından değerlendirileceği bir sistem tasarlanması planlanmaktadır [12].

Orji vd. (2020), blokzincirinin lojistikteki kullanım alanlarını; teknoloji, organizasyon ve çevre değişkenleri çerçevesinde analitik ağ süreci yöntemiyle önceliklendirmiştir. Blokzinciri teknolojisinin benimsenmesini etkileyen en önemli faktörün "belirli blokzinciri araçlarının kullanılabilirliği", "altyapı tesisi" ve "hükümet politikası ve desteği" olduğu tespit edilmiştir. Devlet kurumları, nakliye lojistik firmaları ve blokzinciri hizmet sağlayıcılarına, bu teknolojinin geliştirilmesi, benimsenmesi ve rekabet gücünün iyileştirilmesi için katkı sağlayacağı değerlendirmesinde bulunulmuştur [13].

Ar vd. (2020), çalışmasında analitik hiyerarşi sürecini (analytical hierarchy process-AHP) sezgisel bulanık teori altında çok kriterli bir karar destek modeliyle bir fizibilite çalışması yapmış ve bu metodu Türkiye'deki büyük ölçekli lojistik firmaları üzerinde uygulayarak çıkarımda bulunmuştur [14].

Jain vd. (2020), blokzinciri teknolojisinin lojistik endüstrisinde kullanımının müşteriler tarafından ne ölçüde anlaşılıp kabul edildiğini araştırmıştır. Bu kapsamda tavır, algılanan kullanılabilirlik ve algılanan kullanım kolaylığı değişkenlerini göz önünde bulundurarak "teknoloji kabul modeli" kapsamında blokzinciri teknolojisine hedef müşterilerin tavrını değerlendirmiştir. Blok-

zincirin lojistik endüstrisinde kabul edilebilir, kullanılabilir ve gelecek vadeden bir teknoloji olduğu vurgulanmıştır [15].

Koh vd. (2020), çalışmasında nakliye ve lojistikte blokzincirin kullanımına ilişkin en son uygulama ve çalışmaları özetlemiş, gelecekteki kullanım alanlarına yönelik çıkarımlarda bulunmuştur [16].

Hribernik vd. (2020) çalışmasında e-ticaretteki büyüme ile kurye, ekspres ve paket (courirer-express-packet-CEP) sektöründeki oluşan sorunlara çözüm için blokzinciri tabanlı bir model üzerinde durulmuştur. Çalışmada şehir içi veya çevresinde kurulmuş mikro dağıtım istasyonları ile son müşteri arasındaki lojistik takip sürecinde blokzincirinin kullanımına değinilmiştir. Yalnızca müşterilerin değil, aynı zamanda belediyelerin ve yerel yönetimlerin de kabul edeceği bir çözüm üretilmesi amaçlanmıştır. Konu ile ilgili bir literatür taraması yapılmış ve şehir içi lojistik dağıtım sürecinde bu teknolojinin kullanım alanı bazında uygulanabilirliği tabloda özetlenmiştir [17].

Pournader vd. (2020); teknoloji, güven, ticaret ve izlenebilirlik/şeffaflık ana kümeleri üzerinde tedarik zinciri, lojistik ve nakliye yönetimi hakkında yayımlanmış blokzinciri çalışmalarını incelemiş ve bu konularda çıkarımlarda bulunmuştur [18].

Issaoui vd. (2020), akıllı lojistikte blokzincirinin çeşitli uygulamalarını tanımlamıştır. Yapılmakta olan uygulamalar bilgi, ulaşım, finans ve yönetim olmak üzere dört ana küme dahilinde sınıflandırılmış ve bu uygulamaların somut örnekleri çalışmada sunulmuştur [19].

Mezquita vd. (2020), distribütör ve müşteriler arasındaki çok sayıda aracı kurumun ve üçüncü tarafın da dahil olduğu lojistik hizmetlerinin daha yönetilebilir hale getirilmesini sağlamak üzere blokzinciri tabanlı bir sistem önermiştir. Önerilen modelde, güvenliği artırarak süreçleri otomatikleştirmek için akıllı sözleşmeleri kullanmak, böylece çok temsilcili siteleri tek bir platformda birleştirip dağıtım sürelerini önemli ölçüde hızlandırmak hedeflenmiştir [20].

Tian vd. (2020) tarafından yapılan çalışmanın amacı ise kentsel lojistik zincirindeki müşteri, hizmet sağlayıcılar ve üçüncü otoriteler arasındaki şeffaflık problemini blokzinciri tabanlı bir sistemle çözmektir. Bu kapsamda müşteri memnuniyeti için blokzinciri tabanlı bir değerlendirme yaklaşımı önerilmiş ve kentsel lojistikte müşteri memnuniyetini etkileyeceği öngörülen dört kri-

ter belirlenmiştir. Çalışmada uzun kısa süreli bellek (long short-term memory-LSTM) algoritması kullanılarak gelecek dönemdeki müşteri memnuniyetinin tahmin edilmesi hedeflenmiş ve müşteri memnuniyetinin düşük olması durumunda müşterilere para iadesi gerçekleştirilmesini sağlayacak bir akıllı sözleşme tasarlanmıştır. Tasarlanan akıllı sözleşme gaz tüketimi ile ilgili müşteri memnuniyeti süreçlerinde test edilmiştir. Gelecek çalışma olarak, sürdürülebilir kalkınma için kentsel lojistikte blokzincirinin pratik uygulamalarını teşvik edecek bir çözüm bulmak planlanmış ve kentsel lojistikte sürdürülebilirliği etkileyecek daha fazla faktörü dikkate alarak modelin geliştirilmesi hedeflenmiştir [21].

Tönnissen ve Teuteberg (2020) tarafından yapılan çalışmada, blokzinciri teknolojisinin tedarik zinciri ve lojistik yönetimi süreçlerindeki uygulama alanları incelenmiş, blokzinciri teknolojisinin lojistik endüstrisi ve iş modellerine etkisine ilişkin çıkarımlarda bulunulmuş ve araştırma sorularına ilişkin cevaplar bulunulmaya çalışılmıştır [22].

Yapılan diğer çalışmalar genel olarak değerlendirildiğinde blokzincirinin lojistik, taşımacılık, malzeme nakli, depolama yönetimi konularında bazı uygulamaların hayata geçirildiği görülmekle birlikte bu alanda değerlendirilmesi gereken ciddi bir potansiyel olduğu belirtilmektedir. Yapılan tüm çalışmalarda bu konu, araştırmacılar için bir fırsat olarak gösterilmekte ve iyi değerlendirilmesi gereken bir çalışma alanı olarak tanımlanmaktadır. Bu çalışmalar arasında özellikle Müssigmann ve arkadaşları [8] tarafından yapılan çalışmadaki detaylı tarama blokzincirinin bu alanda kullanımını ve potansiyelini tarafsız bir şekilde özetler niteliktedir. Issaoui ve arkadaşları, çalışmalarında benzer bir taramayı bilgi, ulaşım, finans ve yönetim başlıkları altında dört ana küme dâhilinde ele almış ve benzer sonuçlara ulaşmıştır [19]. 2020'de yapılan çalışmalara yönelik verilen bilgileri içeren özet gösterim Tablo 5.2'de sunulmuş olup tabloda yer alan çalışmalarda, blokzincirinin bu alanda kullanımının getireceği yeniliklere değinilmiş veya bir model ile genişletilerek blokzincirinin alana özel kullanılabilirliği değerlendirilmiştir. Tabloda kullanılan kısaltmalar; Bitcoin (BTC), makine öğrenmesi (machine learning-ML), teknoloji kabul modeli (technology acceptance model - TAM) ve yapısal eşitlik modellemesi (structural equation modeling-SEM) kısaca verilmiştir.

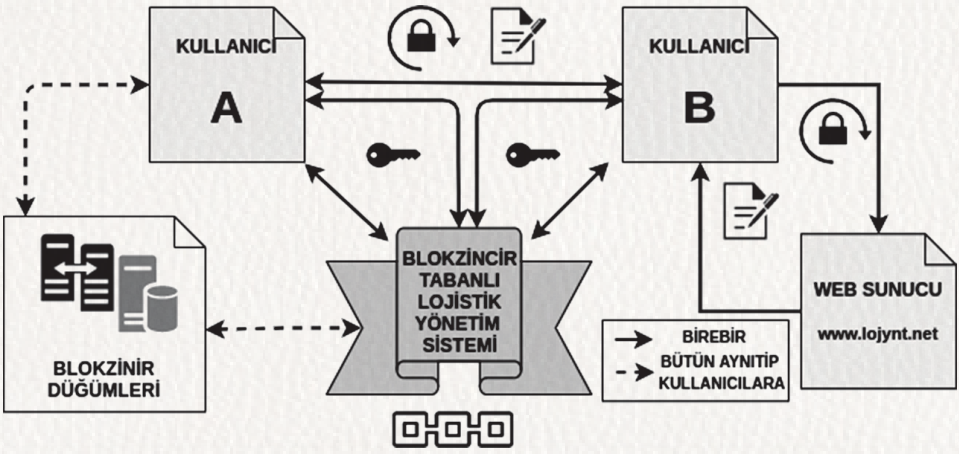
Tablo 5.2. Literatür Çalışmalarının Kıyaslanması

Yayın	Çalışmanın Kapsamı	Kullanılan Teknolojiler
[10]	Kentsel lojistik zincirindeki müşteri, hizmet sağlayıcıları ve üçüncü otoriteler arasındaki şeffaflık problemini çözmek üzere bir model tasarlanmıştır.	BTC, IoT ve ML
[12]	IoT ve blokzinciri entegrasyonu ile akıllı ulaşım ve lojistikte kullanılabilecek; şeffaflık, güvenlik sonradan kanıtlanabilirlik, inkâr edilemezliğin sağlandığı model tasarlanmıştır.	IoT ve BTC
[9]	Fiziksel katmanda sensör olarak IoT cihazların kullanıldığı blokzinciri tabanlı bir yeşil lojistik takip sistemi modellenmiştir.	IoT ve BTC
[10]	Materyal, ekonomik akış ve verinin birbirine entegre edildiği blokzinciri tabanlı bir sistem önerilmiştir. Bu model "swimlane" süreç akış diyagramında okuyucuya görsel olarak sunulmuştur.	BTC
[17]	E-ticaretteki büyüme ile CEP sektöründeki sorunlara çözüm getirmek için blokzinciri tabanlı bir model sunulmuştur.	IoT ve BTC
[14]	Çalışmada AHP'yi sezgisel bulanık teori altında VIKOR'a dahil eden çok kriterli bir karar destek modeli kullanılmıştır.	BTC, AHP, VIKOR ve Bulanık Mantık
[19]	Akıllı lojistikte blokzincirin çeşitli uygulamaları tanımlanmış ve yapılmakta olan uygulamalar bilgi, ulaşım, finans ve yönetim olmak üzere dört sınıfa ayrılmıştır.	BTC
[15]	Blokzinciri teknolojisinin lojistik endüstrisinde kullanımının müşteriler tarafından ne ölçüde anlaşıldığı ve kabul edildiği araştırılmıştır.	BTC, TAM ve SEM
[16]	Nakliye ve lojistikte blokzincirin kullanımına ilişkin en son uygulama ve çalışmaları özetlenmiş ve gelecekteki kullanım alanlarına yönelik çıkarımlarda bulunulmuştur.	BT
[11]	Limanlarda kullanılan lojistik iş yönetim süreçlerinde işlem maliyetini düşürmek için blokzinciri tabanlı bir çözüm sunulmuştur.	IoT ve BTC
[6]	Blokzinciri teknolojisinin lojistik ve taşımacılık yönetim süreçlerinde kullanımına yönelik bir inceleme yapılmıştır ve gelecek uygulama alanlarına yönelik tavsiyelerde bulunulmuştur.	BTC
[13]	Blokzinciri teknolojisinin lojistikteki kullanım alanları teknoloji-organizasyon-çevre değişkenleri çerçevesinde analitik ağ süreci yöntemi kullanılarak önceliklendirilmiştir.	BTC (tarama)
[18]	Teknoloji, güven, ticaret ve izlenebilirlik/şeffaflık ana kümeleri üzerinde tedarik zinciri, lojistik ve nakliye yönetimi üzerine yayımlanmış blokzinciri konusundaki akademik çalışmalar incelenmiş ve bu konularda çıkarımlarda bulunulmuştur.	BTC (tarama)
[8]	Gelecekteki araştırmalara bir temel oluşturmak üzere blokzinciri teknolojisinin lojistik ve tedarik zinciri süreçlerine uygulanması konusundaki 613 akademik çalışmadan alınan verileri içeren bir bibliyometrik analiz sunulmuştur.	BTC (tarama)
[22]	Blokzinciri teknolojisinin tedarik zinciri ve lojistik yönetimi süreçlerindeki uygulama alanları incelenmiş, lojistik endüstrisi ve iş modellerine etkisine ilişkin çıkarımlar verilmiştir.	BTC (tarama)

5.5. ÖNERİLEN MODEL

Model tasarımı Hava Kuvvetlerinin harekât ihtiyacını karşılayacak bir lojistik yönetim sistemi üzerinde düşünülerek planlanmıştır. Oluşturulabilecek sistemin ana elemanları ve blokzincir akış şeması Şekil 5.3'te verilmiştir. Blokzincir tabanlı askeri lojistik yönetim sistemi aşağıdaki elemanlardan oluşmaktadır:

- Blokzincir dahilinde gönderme yapılarak kayıt defterine yazılacak ana harp silah ve malzemeleri
- Bu göndermeyi gerçekleyecek kullanıcılar,
- Bu göndermelerin yapılacağı terminaller,
- Bu sitemin yetkili kullanıcılar tarafından takibini sağlayan web arayüzü
- Defterin hiyerarşik kaydının tutulacağı düğümlerdir.

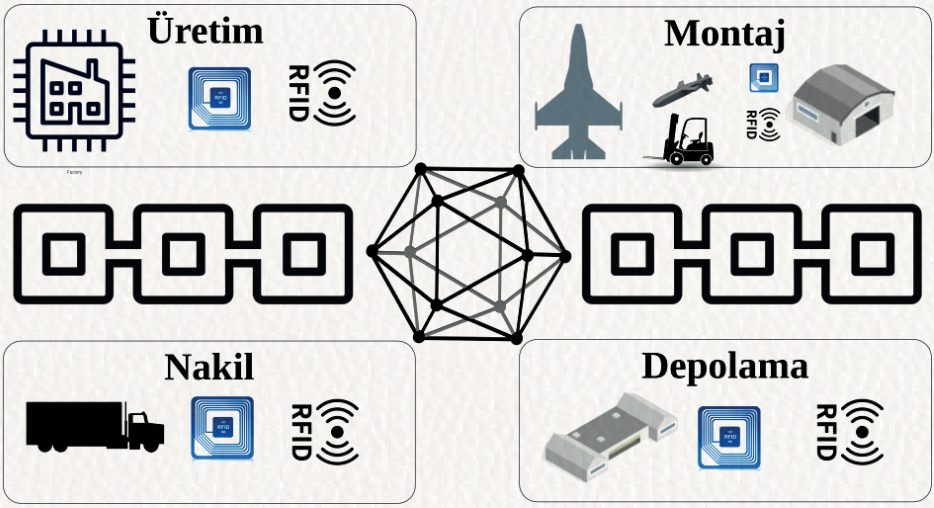


Şekil 5.3. Blokzincir Tabanlı Lojistik Yönetim Sistemi

Model; fiziksel, akıllı sözleşme ve sayısal veri olmak üzere toplam üç katmandan oluşan bir yapı içerisinde teşkil edilmiştir. Fiziksel katman; teçhizat veya teçhizatın taşıma kutusu üzerine montajlı radyo frekansı ile tanımlama (radio-frequency identification - RFID) teknolojisi kullanılarak oluşturulmuş

kimlik tespiti, belirlenmiş kapılar ve depolar üzerine montajlı algılayıcılar ile oluşturulmuştur.

Model ana elemanları Şekil 5.4'te gösterilmiştir. Bu algılayıcıların gönderme yaratırken malzemenin kimlik, zaman, ısı ve nem bilgilerini yazmaları planlanmıştır. Kapı ve depolar ile sensörlerin veri iletişimi kablolu/kablosuz ağ bağlantı cihazları ile sağlanacaktır.



Şekil 5.4. Model Ana Elemanları

Akıllı sözleşme; kullanılan uzlaşma modelini ve modeldeki değişkenlerin tiplerine göre görevlerini tanımlar. Veri ise blokzinciri içerisinde bloklara yazılarak tutulacak olan veridir. Kullanılan malzemenin taşınması ve depolanması büyük dikkat gerektirmektedir. Ayrıca gerekli depolama şartlarının sağlanabilmesi için ana malzeme ve teçhizat montajları yapılana kadar uygun ısı ve nem şartlarında muhafaza edilmelidir. Bu sebeple model kapsamında fiziksel katmanı oluşturan cihazların malzeme veya taşıma kutularına monte edilmesi planlanmıştır. Tasarlanan modelde mühimmat kutularına RFID ve diğer cihazlar IoT olarak tanımlanmış ve mühimmat kutularının saklama koşulu bilgilerini düzenli olarak kablosuz bağlantı ile merkezi sistemle paylaştığı kabul edilmiştir.

5.6. SONUÇ VE DEĞERLENDİRMELER

Blokzinciri gelecek vadeden bir teknoloji olarak popülerliğini korumaya devam etmektedir. Bu teknoloji diğer alanlarda olduğu gibi özellikle askeri lojistik yönetimi konusunda başarılı bir şekilde kullanılabilir. Blokzinciri platformu seçimi dizayn ve kodlama aşamasında bölüm içerisinde verilen açıklık ve saldırı tipleri göz önünde bulundurularak gerekli önlemlerin alınması gerekecektir. Ağ genişledikçe ortaya çıkacak problemler ile ilgili detaylı analiz yapılmalıdır. Model daha geniş bir ağ için tasarlanacaksa sistem üzerinde tesis edilecek sensörlerin ve algılayıcı cihazların topolojik bağlantılarının da göz önünde bulundurulması gerekecektir. Sistem genişledikçe bu cihazların sayısı artacak, bu durumda mimari yapının artan sayıda kablosuz algılayıcı ağ fiziksel kapasite ve ihtiyaçlarını da göz önünde bulundurularak yeniden tasarlanmasına ihtiyaç duyulacaktır. Birim zamanda üretilecek onaylama gönderme hızının da daha verimli olabilecek holo-chain, hashgraph gibi dağıtık kayıt defteri yapılarında gerçeklenmesi veya uygulamaya özel veri yapısı, uzlaşma ve ağ mimarisinin tasarlanması uygulanabilirliği arttıracaktır. Dar kapsamlı bir ağda algılayıcılar daha statik kalacağından blokzincirinin gerektirdiği matematiksel işlemler, bu cihazların bağlı oldukları daha güçlü bir makine tarafından gerçekleştirilebileceği değerlendirilmektedir.

Özellikle bu çalışma kapsamında tasarlanan ve sunulan model hakkında aşağıdaki çıkarımlardan söz etmek mümkündür:

- Hava kuvvetleri malzeme ve teçhizatları ile silah sistemlerinin lojistik takibi göz önünde bulundurularak örneklendiği bu çalışmanın başarılı bir şekilde uygulanabilecektir.
- Saniyede 3500 işlemi destekleyen Hyperledger Fabric altyapısı bu tür bir sistemi rahatlıkla kaldırabilecektir.
- Bu çalışmada önerilen çözüm gibi akıllı hava mühimmatlarının ve silah sistemlerinin lojistik kayıtlarının oluşturulabileceği ve takip edilebileceğidir.

Bu çalışmanın sonraki versiyonunda genişletilmiş sözde kod (pseudo code) eklenerek geliştirilecektir. Bu konuda çalışmak isteyenler bu çalışmanın kaynak kodlarına Github (https://github.com/MSKU-BcRG/Lojistik) adresinden erişebilecektir. Modelin gerçeklenerek bir blokzinciri test ağında denenmesi ve iyileştirilmesi hedeflenmektedir. Farklı konsorsiyum protokollerine sahip arayüzler kullanılarak bir performans değerlendirmesi yapılacaktır.

Blokzinciri teknolojisinin; askeri teknolojilerde kullanımı ile ilgili literatürde yeterli miktarda çalışmanın bulunmadığını vurgulayalım. Blokzincirinin özellikle askeri haberleşme güvenliği konularında kullanımı konusunda çalışmak isteyenler için bu değerlendirilmesi gereken bir fırsat olarak görülebilir. Bu ve buna benzer çalışmaların sayısının artması ile hem bu teknolojinin anlaşılması hem de farklı alanlarda güvenliği sağlama adına yeni çözümlerin geliştirilmesinin önünün açılması mümkündür.

KAYNAKLAR

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," Whitepaper. 2008.

[2] E. Konacaklı, "Ulusal güvenlik için blokzinciri tabanlı siber güvenlik modeli," Yüksek Lisans Tezi, Eskişehir Teknik Üniversitesi Fen Bilimleri Enstitüsü, Eskişehir, 50, 2019.

[3] E. Karaarslan, E. Konacaklı, "Data Storage in the Decentralized World: Blockchain and Derivatives" in " Who Run The World:DATA", 1st ed. Istanbul, Turkey, Istanbul University Press, 2020, ch.3, pp. 37-69. [Online]. Available: https://iupress.istanbul.edu.tr/tr/book/who-runs-the-world-data/chapter/data-storage-in-the-decentralized-world-blockchain-and-derivatives

[4] H. Basak, "Savunma Sanayinin Önündeki Gelecek, Lojistik Destek", Savunma Sanayii Gündemi Dergisi, 2013. s:12, Sayı: 20.

[5] S. Singh, A.S. Hosen and B. Yoon, "Blockchain security attacks, challenges, and solutions for the future distributed iot network," *IEEE Access*, 2021, 9: 13938-13959.

[6] A. Batta, M. Gandhi, A.K. Kar, N. Loganayagam and V. Ilavarasan, "Diffusion of blockchain in logistics and transportation industry: an analysis through the synthesis of academic and trade literature," *Journal of Science and Technology Policy Management, 2020, Vol. ahead-of-print No. ahead-of-print. https://doi.org/10.1108/JSTPM-07-2020-0105*

[7] A. Johnson, D. McCurdy, D. Schechter and K. Loch, "Hot or Cold... How Ready are Third Party Logistics Cold Storage Companies to Implement Blockchain?," Presented at HICSS, 2020, [Online]. Available: https://aisel.aisnet.org/hicss-53/os/blockchain/5/

[8] B. Müssigmann, H. von der Gracht and E. Hartmann, "Blockchain technology in logistics and supply chain management—A bibliometric literature review from 2016 to January 2020," *IEEE Transactions on Engineering Management*, 2020, 67.4: 988-1007.

[9] B. Q. Tan, F. Wang, J. Liu, K. Kang and F. Costa, "A blockchain-based framework for green logistics in supply chains," *Sustainability*, 2020, 12.11: 4656.

[10] D. Kifokeris and C. Koch, "A conceptual digital business model for construction logistics consultants, featuring a sociomaterial blockchain solution for integrated economic, material and information flows," *J. Inf. Technol. Constr.*, 2020, 25: 500-521.

[11] E. Irannezhad, "The architectural design requirements of a blockchain-based port community system," *Logistics*, 2020, 4.4: 30.

[12] M. Humayun, N.Z. Jhanjhi, B. Hamid and G. Ahmed, "Emerging smart logistics and transportation using IoT and blockchain," *IEEE Internet of Things Magazine*, 2020, 3.2: 58-62.

[13] I. J. Orji, S. Kusi-Sarpong, S. Huang and D. Vazquez-Brust, "Evaluating the factors that influence blockchain adoption in the freight logistics industry," *Transportation Research Part E: Logistics and Transportation Review*, 2020, 141: 102025.

[14] I.M. Ar, I. Erol, I. Peker, A. I. Ozdemir, T.D. Medeni and I. T. Medeni, "Evaluating the feasibility of blockchain in logistics operations: A decision framework," *Expert Systems with Applications*, 2020, 158: 113543.

[15] G. Jain, H. Singh, K.R. Chaturvedi and S. Rakesh, "Blockchain in logistics industry: in fizz customer trust or not," *Journal of Enterprise Information Management*, 2020, Vol. 33 No. 3, pp. 541-558. https://doi.org/10.1108/JEIM-06-2018-0142.

[16] L. Koh, A. Dolgui and J. Sarkis, "Blockchain in transport and logistics–paradigms and transitions," International Journal of Production Research, 2020, 58:7, 2054-2062, DOI: 10.1080/00207543.2020.1736428

[17] M. Hribernik, K. Zero, S. Kummer and D. M. Herold, "City logistics: Towards a blockchain decision framework for collaborative parcel deliveries in micro-hubs," *Transportation Research Interdisciplinary Perspectives*, 2020, 8: 100274.

[18] M. Pournader, Y. Shi, S. Seuring and S. L. Koh, "Blockchain applications in supply chains, transport and logistics: a systematic review of the literature," *International Journal of Production Research, 2020, 58:7, 2063-2081, DOI: 10.1080/00207543.2019.1650976*

[19] Y. Issaoui, A. Khiat, A. Bahnasse and H. Ouajji, "Smart Logistics: Blockchain trends and applications," *J. Ubiquitous Syst. Pervasive Networks*, 2020, 12.2: 9-15.

[20] Y. Mezquita, R. Casado-Vara, A.G. Briones, J. Prieto and J.M. Corchado, "Blockchain-based architecture for the control of logistics activities: Pharmaceutical utilities case study.", Logic Journal of IGPL.

[21] Z. Tian, R.Y. Zhong, A. Vatankhah Barenji, Y. T. Wang, Z. Li and Y. Rong, "A blockchain-based evaluation approach for customer delivery satisfaction in sustainable urban logistics," *International Journal of Production Research, 2021, 59:7, 2229-2249, DOI: 10.1080/00207543.2020.1809733.*

[22] S. Tönnissen and F. Teuteberg, "Analysing the impact of blockchain-technology for operations and supply chain management: An explanatory model drawn from multiple case studies", *International Journal of Information Management*, 2020, 52: 101953.

[23] T. A. Ş. Oğuzhan and F. Kiani, (2018). Blok zinciri teknolojisine yapılan saldırılar üzerine bir inceleme. *Bilişim Teknolojileri Dergisi*, 2018, 11.4: 369-382.

Bölüm 6

KUANTUM BİLGİSAYAR ÇAĞINDA KRİPTOSİSTEMLERE BİR BAKIŞ

Sedat Akleylek - Kübra Seyhan

Kuantum mekaniği üzerine yapılan çalışmalar sonucu farkına varılan bazı özellikler kuantum bilgisayar kavramının hayatımıza girmesine olanak sağlamıştır. Hızlı işlem yapma gücüne sahip olabilecek bu bilgisayarların üretilmesi ve pratikte kullanılabilir olması açık anahtarlı kriptosistemler özelinde güvenli haberleşme temellerini olumsuz yönde etkileyecektir. Bu etkilere erken önlem alınarak oluşabilecek olası problemlerin ortadan kaldırılması amacıyla kuantum bilgisayarlar sonrası açık anahtarlı kriptosistemlerin belirlenmesi üzerine çalışmalar yapılmaktadır. Bu bölümde kuantum bilgisayarlar sonrası kriptografi kavramının ortaya çıkış süreci, etkileri ve sonucunda gerçekleştirilen eylemler ifade edilmiştir. Ayrıca kuantum bilgisayarlar varlığında güvenli olduğu bilinen/inanılan kriptosistem aileleri, genel çalışma yaklaşımları ve özellikleri ele alınmıştır. Bu sınıflandırma özelinde NIST tarafından kuantum bilgisayarlar sonrası güvenli açık anahtarlı kriptosistem tasarımı sürecinde yer alarak son tura geçmeye hak kazanan şifreleme/anahtar paketleme ve imzalama şemaları için finalist ve alternatif aday algoritmaların özellikleri ve farklılıkları detaylandırılmıştır.

6.1. GİRİŞ

Kuantum sonrası kriptografi kavramı günümüz hesaplama sistemleri karşısında güvensiz olarak hem geleneksel hem de kuantum kriptanalize dirençli olan kriptosistemlerin oluşturulması gereksinimi ile önem kazanmıştır. Kuantum meka-

niği üzerinde çalışmalar yapan araştırmacılar farkına vardıkları yeni özelliklerin klasik bilgisayarlardan üstün sistemlerin tasarlanmasında kullanılabileceği sonucuna varmıştır. Keşfedilen özellik incelendiğinde klasik hesaplama ve kuantum hesaplama arasındaki temel farklılığın bilginin depolandığı birimin özelliklerinin farklılaşması ile elde edildiği gözlemlenmiştir. Klasik hesaplamada bilgi değeri 0 ve 1 olan bitlerde depolanırken kuantum hesaplama da ise hem 0 hem de 1 değerini aynı anda tutarak iki durumun süper pozisyonunu da içerebilen kübitlerde depolanır. Bilginin depolanmasında ortaya çıkan farklılaşma sonucu kuantum bilgisayarlarda çok daha hızlı paralel işlem yapma gücünün elde edilebileceği gözlenmiştir [1]. Bu sonuçlar kuantum özelliklerini içeren bir kuantum bilgisayarın aynı sorunu çözebilen algoritmaların koşulduğu klasik bilgisayarlardan çok daha hızlı çarpanlara ayırma ve arama işlemlerini yapabilmesine olanak sağlamıştır. Kuantum sonrası dünyada yaşanan bu gelişme ile ortaya çıkan yaklaşımda kötü niyetli kişide saldırı aracı olarak kuantum bilgisayarın kullanması durumunda olası güvenlik senaryoları incelenir. Bu saldırı modeli 90'lı yılların ortasında kuantum teknolojisi ve kuantum hesaplama kavramlarına önem kazandıran bir gelişime sebep olmuştur. Bu kapsamda kuantum bilgisayarların varlığını temel alan kuantum bilgisayar algoritmaları teorisi ortaya atılmıştır. Bu teoriye göre yeterince güçlü kuantum bilgisayarlarda Shor [2] algoritması yaygın olarak kullanılan açık anahtarlı şifreleme şemalarını kırabilecekken, Grover algoritması standartlaşmış simetrik algoritmaları zayıflatabilecektir. Bu etki, Shor algoritmasının sayıları çarpanlarına ayırması ve Grover [3] algoritmasının aramaları daha verimli gerçekleştirmesi ile ortaya çıkmıştır. Sonuç olarak bu algoritmalar güvenliklerini ilgili matematiksel problemlerin belirli koşullar altında çözülmesinin zor veya imkansız olduğu gerçeğine dayandıran, yaygın olarak kullanılan birçok kriptosistemi tehdit etmektedir. Bu değişime neden olabilecek kuantum bilgisayarların oluşturulma süreci belirsiz olsa da yeni kriptografik sistemlerin tasarlanma sürecinin uzun olması bu alana olan ilgiyi artırmıştır [4].

Kuantum sonrası kriptografik gelişmeler Gelişmiş Şifreleme Standardı (Advanced Encryption Standard-AES) ve Güvenli Özet Algoritmaları (Secure Hash Algorithms-SHA) gibi simetrik şifreleme sistemlerinin sundukları koruma gücünü azaltsa da çok büyük bir değişimi beraberinde getirmemiştir. Çünkü bu kriptosistemler anahtar boyutu iki katına çıkarılarak Grover algoritmasının etkisini ortadan kaldırabilecek özelliktedir. Örneğin, AES-128'i kırmak için 2^{128} işleme ihtiyaç duyulurken kuantum bilgisayarlarda bu değer 2^{64} kuantum işlem olur. Güvenlik karşısında daha büyük anahtar boyutuna geçme maliyeti ihmal edi-

lerek AES-256 kullanımı ile süreç yönetilebilecektir [5]. Ancak, açık anahtarlı kriptosistemlerde durum biraz daha farklıdır. Geleneksel açık anahtarlı kriptosistemler hesaplamalı olarak zor matematiksel problemlerin zorluk varsayımını temel alır. Daha açık bir ifadeyle, Rivest–Shamir–Adleman (RSA) şifreleme/anahtar paketleme kriptosistemi tamsayı çarpanlara ayırma problemi, eliptik eğri (elliptic-curve cryptography-EC)-tabanlı kriptosistemler eliptik eğri üzerindeki ayrık logaritma problemi, Diffie-Hellman (DH) anahtar değişimi ve Elektronik İmza Algoritması (Digital Signature Algorithm-DSA) imzalama şeması tamsayılar üzerinde ayrık logaritma probleminin zorluk varsayımına dayanır. Yaygın olarak kullanılan bu kriptosistemlerin bazı özellikleri ve kuantum bilgisayarlar sonrası kriptografide kullanım durumları Tablo 6.1'de özetlenmiştir.

Tablo 6.1. Geleneksel Kriptosistemlerde Bazı Temel Özellikler

	Simetrik Sistemler	**Anahtarsız Sistemler**	**Asimetrik/Gizli Anahtarlı Sistemler**		
Örnek Kriptosistemler	*AES*	*SHA-2, SHA-3*	*RSA*	*DH/ECDH*	*DSA/ECDSA*
Amaç	*Şifreleme*	*Özet İşlevi*	*Şifreleme İmzalama Anahtar Paketleme*	*Anahtar Değişim*	*İmzalama*
Zor Problem	-	*Tek Yönlülük Çığ Etkisi Kararlılık Çakışmaya Direnç Tahmin Edilememe*	*Tamsayı Çarpanlara Ayırma Problemi*	*Tamsayı/Eliptik Eğri Ayrık Logaritma Problemi*	*Tamsayı/Eliptik Eğri Ayrık Logaritma Problemi*
Kuantum Sonrası Etki	*Anahtar Uzunluğu Artırılarak Güvenli*	*Çıktı Boyutlarının Artırılması ile Güvenli*	*Güvenli Değil*	*Güvenli Değil*	*Güvenli Değil*

Günümüz hesaplama sistemlerinde çalışan kriptanaliz yöntemleri ile polinom zamanda bu zor problemler (tamsayı çarpanlara ayırma ve ayrık logaritma) çözülemediği için geleneksel açık anahtarlı kriptosistemler güvenli iletişimin sağlanmasına olanak sağlar. Bu bağlamda Shor algoritması ile güçlü kuantum bilgisayarlarda polinom zamanda ortaya çıkan etki, açık anahtarlı kriptosistemleri güvensiz hale getirmiştir [6]. Bu durum açık anahtarlı kriptosistemlerin kullanıldığı simetrik kriptosistemlerde anahtar paylaşımı, belgelerin/mesajların doğrulanması, paylaşılan anahtar kullanılarak güvenli iletişimin sağlanması gibi temel işlemlerin güvenli bir şekilde yapılabilmesini engelleyecektir. Oluşan bu senaryoyu daha da kötüleştiren durum ise geçmişe yönelik şifre çözme yaklaşımıdır. Daha açık bir ifadeyle, bugün saldırgan tarafından ele geçirilen herhangi bir şifre metin, büyük ölçekli kuantum bilgisayar

varlığında kullanılan algoritmaların etkinliğini kaybedecek olması nedeniyle çözülebilecektir. Dolayısıyla kuantum bilgisayar gücü tam olarak ortaya çıkmadan kuantum güvenli kriptosistemlerin oluşturulması hem mevcut verilerin hem de gelecekte ortaya çıkacak verilerin güvenliği açısından oldukça önemlidir. Bu kapsamda gizli kalması gereken verilerin korunması amacıyla çalışan araştırmacıların ileriye dönük ele alması gereken bileşenler Mosca tarafından şu şekilde tanımlanmıştır [5].

- *x*: *Şifrelemenin güvenli olması için kaç yıla ihtiyaç var?*
    - Bu bileşen gizli verinin kullanım özelliğine göre incelenmelidir. Kısa sürede değişme özelliği oldukça yüksek olan veri gruplarının herkese açık hale gelmesi problem oluşturmaz. Ancak kimlik bilgileri ve çok gizli askeri bilgiler gibi hassas durumlar dikkatle ele alınmalıdır. Dolayısıyla, *x*'in değerini tanımlamak makul miktarda düşünce, risk analizi ve modelleme gerektiren önemli bir konudur.
- *y*: *Mevcut sistem altyapısını kuantum bilgisayarlar açısından güvenli hale getirmek kaç yıl sürecek?*
- *z*: *Büyük ölçekli bir kuantum bilgisayar kaç yıl içinde inşa edilecek?*
    - Bilinmezliğini sürdüren bir değerdir. 2011 yılında IBM tarafından yapılan açıklamada pratikte uygulanabilir bir kuantum bilgisayarın oluşturulmasından uzak olunsa da takip edilmesi gereken durumların belirlenmesi açısından dikkatle ele alınması gereklidir.

Burada *x+y>z* durumu incelendiğinde; *y*'nin son yıllarında ya iş yapmayı bırakmamız ya da *z* yıllarında kırılacağını bildiğimiz kriptosistemleri kullanmaya devam etmemiz gerektiği anlamına gelir. Her iki durumda iş yapmak açısından uygun zamanlar olmayacağından hemen harekete geçilmesi gerekir. Bu varsayımlar temelinde önlem alınabilmesi hem geçmişe yönelik hem de gelecekte veri güvenliği için kuantum sonrası güvenli açık anahtarlı kriptosistem tasarımının en kısa sürede ele alınması gerekliliği ortaya çıkmıştır. Bu kapsamda gerçekleştirilen girişimlerden en önemlisi ABD Ulusal Standartlar ve Teknoloji Enstitüsü (National Institute of Standards and Technology-NIST) tarafından 2017 yılında başlatılan standartlaşma sürecidir [8]. Büyük ve güçlü kuantum bilgisayarlar varlığında açık anahtarlı kriptosistemlerde meydana gelebilecek olası güvenlik zafiyetlerini önleyen kuantum dirençli açık anahtarlı kriptosistem(leri) standartlaştırmak amaçlanmıştır. Herkese açık olan bu çağrıya şifreleme, anahtar paketleme

(key encapsulation mechanism-KEM) ve imzalama şemalarını içeren 82 algoritma başvuruda bulunmuştur. Referans ve optimize edilmiş C kodu uygulamaları, bilinen cevap testleri, yazılı şartname, fikri mülkiyet beyanları ve algoritmaların çeşitli yazılım ve donanım platformlarındaki uygulamaları şartlarını yerine getiren 69 aday algoritma ilk tur için seçilmiştir [9]. Bu algoritmaların tüm standartlaşma süreci boyunca maliyet ve performans, güvenlik ve algoritma-uygulama özellikleri değerlendirme kriterleri baz alınarak incelenmesine karar verilmiştir. Bu kriterler kapsamında yapılan değerlendirmeler sonucu ikinci tura 26 aday algoritma seçilmiştir. Üçüncü tur değerlendirme sonuçları ise 22 Temmuz 2020'de 7 finalist ve 8 alternatif aday algoritmanın açıklanması ile sonuçlanmıştır [10]. Tahmini 2023-2024'te tamamlanması planlanan standartlaşma süreci için kuantum-dirençli kriptosistem aileleri açısından algoritmaların sınıflandırılmasına ait sayısal veriler Tablo 6.2'de özetlenmiştir.

Tablo 6.2. NIST Standartlaşma Sürecinde Kuantum Sonrası Güvenli Kriptosistem Aileleri

	İmzalama				Şifreleme ve Anahtar Paketleme				Toplam			
	1.Tur	*2.Tur*	*3. Tur* Finalist	*3. Tur* Alternatif	*1.Tur*	*2.Tur*	*3. Tur* Finalist	*3. Tur* Alternatif	*1.Tur*	*2.Tur*	*3.Tur* Finalist	*3.Tur* Alternatif
Kod-tabanlı	~~3~~	~~-~~	-	-	~~17~~	~~7~~	1	2	~~20~~	~~7~~	1	2
Kafes-tabanlı	~~5~~	~~3~~	2	-	~~23~~	~~9~~	3	2	~~28~~	~~12~~	5	2
Özet-tabanlı	~~3~~	~~2~~	-	1	-	-	-	-	~~3~~	~~1~~	-	1
Çok Değişkenli Polinom Sistemleri	8	4	1	1	~~2~~	-	-	-	~~10~~	4	1	1
Diğer	~~2~~	~~0~~	-	1	6	~~1~~	-	1	8	~~2~~	-	2
Toplam	~~21~~	~~9~~	3	3	~~48~~	~~17~~	4	5	69	~~26~~	7	8

Büyük ölçekli kuantum bilgisayarların oluşturulması ihtimalinin etkilediği alanlardan bir de elektronik imzalarda kullanılan standartlardır. Temmuz 2013'te NIST tarafından Federal Bilgi İşleme Standartları Yayınları (FIPS PUBS) ile oluşturulan FIPS 186 standardı ile elektronik imzalar oluşturmada kullanılacak standart imzalama algoritmaları RSA ve (EC) DSA olarak belirlenmiştir [12]. Tamsayı çarpanlara ayırma ve (eliptik eğri) ayrık logaritma problemlerinin zorluk varsayımına dayandırılan bu şemaların kuantum bilgisayarlar varlığında kaybolacak etkisinin önlenmesi ve yeni standartların oluşturulması amacıyla elektronik imzalar özelinde standartlaşma süreci başlatılmıştır [13]. NIST, durum bilgisi içeren özet-tabanlı imzalar (stateful hash-based signature) ile uygulama programlama arayüzü (Application Programming Interface-API) için beklentilerin karşılanmaması nedeniyle bu standartlaşma sürecini açık anahtarlı kriptosistemleri standartlaşma sürecinden ayrı tutmuştur [14]. 2018 yılında ilan edilen çağrıda durum bilgisi içeren özet tabanlı şemalar ile kısıtlama yapılması-

nın nedeni bu şemaların güvenliğinin yalnızca temeldeki özet fonksiyona bağlı olmasıdır. Bu özellik ile ters görüntü ve ikinci ters görüntüye dayanıklılık garanti edileceğinden büyük ölçekli kuantum bilgisayarlar karşısında güvenli imzalama şemalarının oluşturulabileceği düşünülmüştür. Ekim 2020'de tamamlanan bu süreçte standart elektronik imza algoritmalarına ek durum bilgisi içeren özet-tabanlı iki elektronik imza şeması belirlenmiştir. Bunlar:

- Leighton-Micali Signature (LMS), Hierarchical Signature System (HSS)
- eXtended Merkle Signature Scheme (XMSS), multi-tree XMSS

NIST SP 800-208 kapsamında standartlaştırılan elektronik imza şemaları genellikle sağlanması zor olan dikkatli bir durum yönetimi gerektirmesi nedeniyle genel kullanım için uygun olmasa da büyük ölçekli kuantum bilgisayarlar varlığında kısıtlı cihazlar ve özel kullanım amaçlarında aktif olarak kullanılabilecek özellikte olduğu ifade edilmiştir [15].

Farklı standartlaşma süreçleri ile büyük ölçekli kuantum bilgisayarların varlığında meydana gelebilecek güvenlik problemlerinin ele alınması günümüzde ön plana çıkan araştırma konularından biridir. Bu kapsamda kuantum-dirençli kriptografinin inşa edilerek Internet ve diğer teknolojilerin güvenliğinin sürdürülebilmesine olanak sağlayan matematiksel teknikleri içeren kriptosistem aileleri vardır. Yapıları nedeniyle günümüzde standart olarak kullanılmasa da sağladıkları kuantum sonrası güvenlik garantisi ile değerlendirilen temel kriptosistem aileleri; çok değişkenli polinom sistemleri, kod, kafes ve özet-tabanlı kriptosistemlerdir. Bu kriptosistem sınıfları ve temel özellikleri Bölüm 6.2'de açıklanmıştır.

6.1.1. Kapsam ve Motivasyon

Büyük ölçekli kuantum bilgisayarların oluşturulması üzerine yapılan çalışmalar Shor algoritması ile açık anahtarlı kriptosistemlerin güvenliğini tehdit etmektedir. Kuantum bilgisayarların hesaplama gücü karşısında polinom zamanda çözülebilecek olan hesaplamalı olarak zor problemlerin yerini alabilecek problemleri içeren çeşitli kriptosistem sınıfları mevcuttur. Bu kriptosistem ailelerinin açıklanması, hangi özellikleri nedeniyle nasıl tercih edildikleri, standartlaşma süreçlerinde yerinin ifade edilmesi, NIST'in açık anahtarlı kriptosistemleri standartlaştırma sürecinin gelişimi ve son tura kalmaya hak kazanan algoritmaların özetlenmesi bu alanda yapılacak olan çalışmalar için yol gösterici olacaktır. Bu bölümün kapsamı şu şekildedir:

- Kuantum sonrası kriptografi kavramının ortaya çıkışı, etkilediği geleneksel kriptosistemler ve yapılan çalışmaların özetlenmesi.
- Kuantum sonrası güvenli çok değişkenli polinom sistemleri, kod, kafes ve özet-tabanlı kriptosistem aileleri, temel özellikleri ve genel kullanım alanlarının detaylandırılması.
- Kuantum sonrası kriptografinin gelişimi, NIST standartlaşma sürecinde kuantum sonrası güvenli kriptosistem ailelerinin yeri ve değerlendirme kriterlerinin ifade edilmesi.
- Standartlaşma sürecinde son tura geçmeye hak kazanan finalist/alternatif şifreleme/anahtar paketleme ve imzalama şemalarının değerlendirme kriterleri özelinde özetlenmesi.

6.1.2. Organizasyon

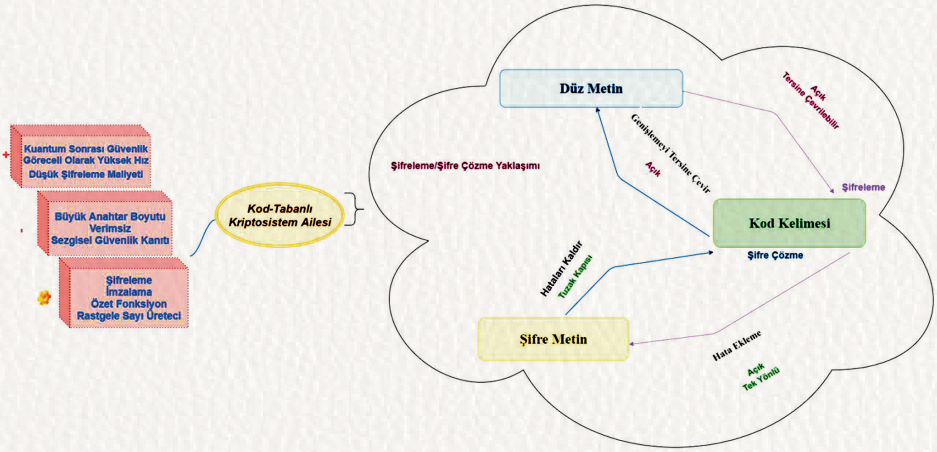
Bu çalışmada Bölüm 6.2'de kuantum bilgisayarlar karşısında güvenli olduğu bilenen kriptosistem sınıfları, temel özellikleri ve genel kullanım amaçları ifade edilmiştir. Bölüm 6.3'te NIST standartlaşma sürecinde yer alarak son tura geçmeye hak kazanan finalist ve alternatif adaylar dahil oldukları kriptosistem ailesi özelinde incelenmiştir. Ayrıca değerlendirme kriterleri baz alınarak üstün ve eksik yanları özetlenmiştir. Son olarak Bölüm 6.4'te sonuç ve öneriler verilmiştir.

6.2. KUANTUM SONRASI GÜVENLİ KRİPTOSİSTEM AİLELERİ

Kuantum mekaniğinde fark edilen yeni özelliklerin bir sonucu olarak ortaya çıkan daha güçlü ve hızlı hesaplama sistemlerinin üretilebilecek olması ihtimali en temelde güvenlik kavramının etkisini değiştirmiştir. Gelişimsel süreci Bölüm 6.1'de açıklanan kuantum sonrası kriptografi, geleneksel açık anahtarlı kriptosistemlerin temel aldığı hesaplamalı olarak zor problemlerin yerini alabilecek eski ve yeni yaklaşımların tespit edilmesi ve uyarlanması ile ilgilenir. Her ne kadar bir varsayımın etkisi üzerinde hareket edilse de gelişen teknoloji ile artan araştırmalar hem geçmişe hem de geleceğe yönelik güvenlik garantisi için kuantum bilgisayarlar sonrası kriptosistemlerin tasarlanmasını gerekli kılmıştır. Bu kapsamda bu bölümde tamsayı/eliptik eğri ayrık logaritma ve çarpanlara ayırma problemlerinin yerini alabilecek çözülmesi hesaplamalı olarak zor veya imkansız problemleri veya iyi çalışılmış işlevleri içeren kuantum sonrası güvenli kriptosistem aileleri ve özellikleri açıklanmıştır.

6.2.1. Kod-Tabanlı Kriptosistem Ailesi

Kodlama teorisi sayısal bilginin gürültülü bir kanaldan güvenli iletim esnasında ortaya çıkan hataların düzeltilmesinde hata düzeltme kodlarını kullanılabilir hale getirmiştir [16]. Her ne kadar bazı özel kodlar için hatayı düzeltmek mümkün olsa da rastgele doğrusal kodlar için bu problem oldukça zordur. Bu anlamda kullanılan özel kodlardan biri de etkili hata düzeltmeye olanak sağlayan Goppa kodlarıdır [17]. Kodlama ve kod çözme işlem adımlarını gizleyerek güvenli bir kodlama şeması oluşturulabilmesine olanak sağlayan bu kodlarda kullanılan yaklaşım NP-tam problem olan sendrom kod çözme (syndrome decoding) probleminin zorluğuna dayanır. Daha açık bir ifadeyle, bu problem hem günümüz hesaplama sistemleri hem de kuantum bilgisayarlar için çözülmesi zor veya imkansız bir problemdir [18]. Kod-tabanlı kriptosistemlerin genel özellikleri ve işlem adımları Şekil 6.1'de görselleştirilmiştir.



Şekil 6.1. Kod-Tabanlı Kriptosistem Ailesi

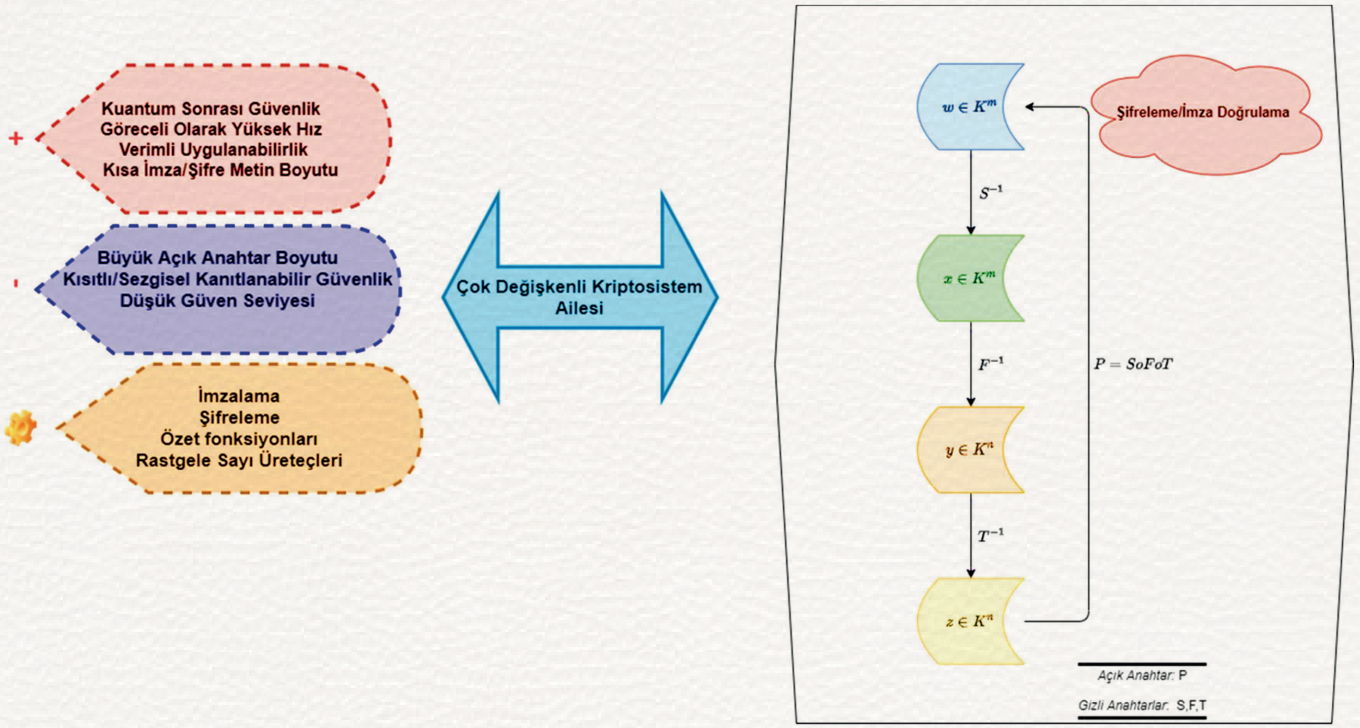
Hata düzeltme kodları teorisini kullanan kod-tabanlı kriptosistemler ilk olarak 1978'de McEliece [19] şifreleme sisteminin önerilmesi ile ortaya çıkmıştır. Güvenlik varsayımı genel bir doğrusal kodun çözümünün NP-zor bir problem olması ile açıklanan bu açık anahtarlı şifreleme sisteminde şifreleme ve şifre çözme aşamalarındaki düşük karmaşıklık göreceli olarak yüksek hız

elde edilmesine olanak sağlamıştır. Gizli anahtar rastgele seçilen ikili Goppa kodu iken açık anahtar bu kodun izin verilen bir çeşidinin rastgele bir üretici matrisidir. Bu anahtarlar kullanılarak elde edilen şifre metin ise bazı hataların eklendiği bir kod sözcüğüdür. Bu sistemin en temel özelliği bazı parametrelerde ayarlamalar gerektirse de kuantum bilgisayarlarda bile güvenliği garanti etmesidir [20]. Kod-tabanlı ilk imzalama şeması 2001'de güvenlik varsayımı sendrom kod çözme probleminin zorluğuna dayandırılan Courtois, Finiasz ve Sendrier tarafından önerilen şemadır [21]. Her ne kadar McEliece şifreleme şemasında olduğu gibi büyük açık anahtar boyutu dezavantajına sahip olsa da imza boyutları oldukça kısadır. İmza doğrulanması hızlıyken imzanın oluşturulması göreceli olarak verimsiz olan bu şemanın dezavantajlarında literatürdeki çalışmalar ile iyileştirmeler yapılmıştır.

McEliece şifreleme sisteminin önerilmesinden bu yana kırılamamış olması hata düzeltme kodlarına dayanan birçok sistemin önerilmesine olanak sağlamıştır. Güvenlik ve verimlilik arasında bir tercih yapılmasına ihtiyaç duyan kod-tabanlı kriptosistemler oldukça hızlı olmalarına rağmen anahtar boyutlarının geleneksel asimetrik sistemlere göre büyük olmasının dezavantajını yaşamaktadır [22]. Oluşturulan kod-tabanlı yeni ve alternatif kriptosistemler ile farklı yaklaşım/yöntemlerle anahtar boyutlarının küçültülmesi hedeflense de bu durum önerilen kriptosistemleri çeşitli saldırılar karşısında savunmasız bırakmıştır. Genellikle kod-tabanlı şifreleme ve imzalama şemaları tasarlansa da tanımlama şemaları [23], [24], [25], rastgele sayı üreteçleri [26], [27] ve kriptografik özet fonksiyonu tasarımında kod-tabanlı yaklaşım üzerine çalışmalar [28] yapılmıştır.

6.2.2. Çok Değişkenli Polinom Sistemleri-Tabanlı Kriptosistem Ailesi

Çok değişkenli kriptografi kavramı açık anahtarlı kriptosistemlerin tuzak kapısı tek yönlü fonksiyonunun sonlu bir cisim üzerinde çok değişkenli ikinci dereceden bir polinom haritası şeklinde tanımlanması ile ifade edilir [1]. Verimlilik nedenlerinden dolayı sistemi oluşturan çok değişkenli polinomların seçiminde belirli sınırlar mevcuttur. Genellikle $\mathbb{F}_{2^8}$, $\mathbb{F}_{31}$ gibi küçük bir sonlu cisim üzerinde tanımlanan ikinci dereceden polinomlar kullanılır [29]. Bu kriptosistem ailesine ait genel özellikler ve şifreleme/imza doğrulama yaklaşımı Şekil 6.2'de özetlenmiştir.



Şekil 6.2. Çok Değişkenli Polinom Sistemleri-Tabanlı Kriptosistem Ailesi

1980'li yıllarda tasarlanmaya başlanan çok değişkenli polinom sistemleri-tabanlı kriptosistemlerin güvenliği ikinci dereceden çok değişkenli denklemler sistemine sonlu cisimler üzerinde çözüm bulmanın (MQ probleminin çözümü) zor veya imkansız olması ile tanımlanır. Açık anahtarlı kriptosistem sınıfından olan çok değişkenli polinom sistemleri, elektronik imza şemalarının tasarımında yaygın olarak kullanılmaktadır. Bu duruma neden olan etken en güvenli çok değişkenli kriptosistemlerin düzgün rastgele katsayıları olan denklem sistemlerinden oluşturulan imza şemaları ile tasarlandığının ifade edilmesidir [5]. Bu kapsamda çok değişkenli kriptosistemlerin temel özellikleri şu şekilde özetlenebilir [20], [30]:

- Verimli uygulanabilirliği ile göreceli olarak yüksek hız garantisi.
- Küçük ve sonlu cisimler üzerinde toplama ve çarpma gibi temel aritmetik işlemleri yapısında bulundurmasından dolayı düşük hesaplama gereksinimi.
- Kısa imza boyutu.
- Kriptanaliz açısından saldırı karmaşıklıklarının teorik tahminlerinin deneysel sonuçlarla örtüşmesi ile sağlam temel.
- Ana dezavantaj; ≈10−100 KB'lık büyük açık anahtar boyutu.
- Kanıtlanabilir güvenlik açısından, çok değişkenli şemaların güvenliğini çok değişken-tabanlı zor matematik problemlere indirgeyen kesin kanıtların sınırlı olması.

Çok değişkenli polinom sistemleri-tabanlı kriptosistemlerden en bilinenleri Matsumoto ve Imai [31] kriptosistemi ve dengesiz yağ-sirke (Unbalanced Oil-Vinegar-UOV) [32] imzalama şemalarıdır [1]. Çok değişkenli polinom sistemleri kullanılarak elde edilen imzalama şemalarının şifreleme şemalarına göre daha etkin özelliklere sahip olduğu gözlemlenmiştir. Çok büyük açık anahtar boyutu ve uzun şifre çözme süreleri şifreleme şemalarında dezavantaj oluştururken çok küçük imza boyutu imzalama şemalarının üstünlüğünü oluşturmuştur [5].

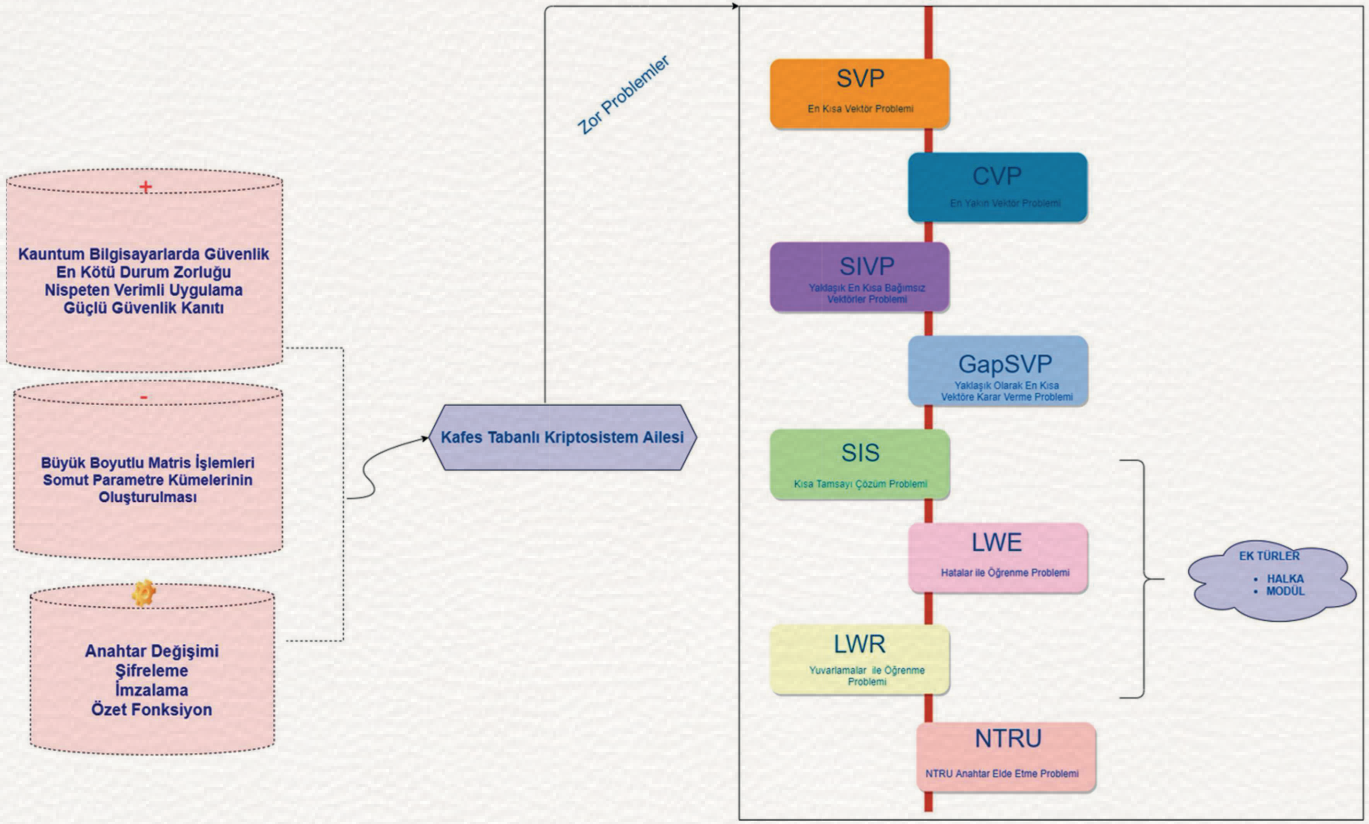
6.2.3. Kafes-Tabanlı Kriptosistem Ailesi

Matematikte kullanımı on sekizinci yüzyıla dayanan kafes yapısının kriptografik yapılarda kullanımı ilk olarak Ajtai [33] tarafından çok boyutlu kafeslerin kullanımı ile ortaya çıkmıştır [34]. 1996 yılında ortalama durum kısa tamsayı çözüm (short integer solution-SIS) probleminin zorluk varsayımını temel alan bu kriptografik yapının güvenliği en kötü durumda en kısa vektör probleminin (shortest vector problem-SVP) zorluk varsayımı ile ilişkisi açıklanarak tanımlanmıştır [35]. Ajtai fonksiyonu ile eş zamanlı yıllarda Hoffstein, Pipher ve Silverman tarafından kafes-tabanlı açık anahtarlı NTRU şifreleme şeması önerilmiştir. Polinomlar halkasını kullanan ilk kriptografik yapı olma özelliğini taşıyan NTRU, bütünsel anahtar özelliği ve pratikte verimliliği ile önem kazanmıştır [36]. Bu gelişmelerden etkilenerek 1997 yılında Goldreich, Goldwasser ve Halevi (GGH) [37] tarafından açık anahtarlı şifreleme ve imzalama şemaları oluşturulmuştur. Sezgisel güvenliği açıklanan GGH şifreleme şeması başarılı şifreleme ve asimptotik olarak direnç sağlarken GGH imzalama şeması sonraki yıllarda kırılmıştır.

Kafes-tabanlı kriptografik yapıların bilinen en temel özellikleri en kötü durum zorluğuna, nispeten verimli uygulamalara ve mükemmel basitliğe dayanan çok güçlü güvenlik kanıtlarına sahip olmalarıdır. En kötü durum zorluğuna dayanan kafes problemleri güçlü güvenlik garantisi sağlasa da bazıları pratikte kullanılabilecek kadar verimlidir. Bu kriptografik yapıları kırmak başka bir deyişle, altta yatan kafes probleminin herhangi bir örneğini çözmek için etkili bir algoritmanın bulunması en az en kötü durumda birkaç kafes problemini çözmek kadar zordur [20]. Zor kafes problemlerini temel alan kafes-tabanlı kriptosistemlerin özellikleri ve temel zor kafes problemleri Şekil 6.3'te ifade edilmiştir.

Güvenlik açısından kafes-tabanlı kriptografik yapılar ikiye ayrılabilir [38].

- Tipik olarak çok verimli olan ancak genellikle destekleyici bir güvenlik kanıtı olmayan pratik teknikleri içeren yapılar.
- Kafes problemlerinin en kötü durum zorluğuna dayanan güçlü güvenlik garantileri olan sadece birkaçı pratikte verimli olan yapılar.



Şekil 6.3. Kafes-Tabanlı Kriptosistem Ailesi

Zor kafes problemlerini içeren kriptosistemler ele alındığında en iyi bilinen klasik algoritmalardan anlamlı ölçüde daha iyi performans göstererek bu problemleri çözebilen kuantum bilgisayarlarda polinom zamanda çalışan bir algoritma henüz bilinmemektedir. Bu zor kafes problemlerini çözme girişimleri Shor algoritmasının keşfi ile etkinlik kazansa da herhangi başarılı bir sonuç elde edilememiştir. Bu durumun temel nedeni, kuantum bilgisayarlarda polinom zamanda çalışan bir algoritmada kullanılan periyodiklik bulma tekniğinin kafes problemlerinde geçerli olmamasıdır. Sonuç olarak kafes-tabanlı kriptosistem ailesi sağladığı özellikler ile kuantum bilgisayarlar sonrası kriptografi için güvenli alternatif sistemlerden biri olmuştur.

6.2.4. Özet-Tabanlı Kriptosistem Ailesi

Matematiksel olarak rastgele/keyfi uzunluktaki dizileri sabit uzunlukta dizilere eşleyen özet fonksiyonların kriptografik ilkelerde kullanılmasıyla özet-tabanlı kriptosistemler ortaya çıkmıştır. Pratikte neredeyse tüm kriptografik şemalarda kullanılan özet fonksiyonların temel özelliği hesaplanmasının kolay, tersine çevrilmesinin zor olmasıdır. Bu özelliği içeren en bilinen imzalama şeması 1979 yılında Lamport [39] tarafından önerilmiştir. Büyük mesajlar için gizli ve açık anahtar ikililerinin yanı sıra imzanın da son derece büyük olduğu bu şemada anahtarlar tek kullanımlıktır. Başka bir deyişle, her imza için yeni gizli-açık anahtar çiftlerinin her seferinde yeniden oluşturulması gerekmektedir. Ancak bu şemada gizli anahtarın işlevini artırmak ve imza boyutunu indirgemek için çeşitli yaklaşımları içeren birkaç iyileştirme literatürde yer almıştır.

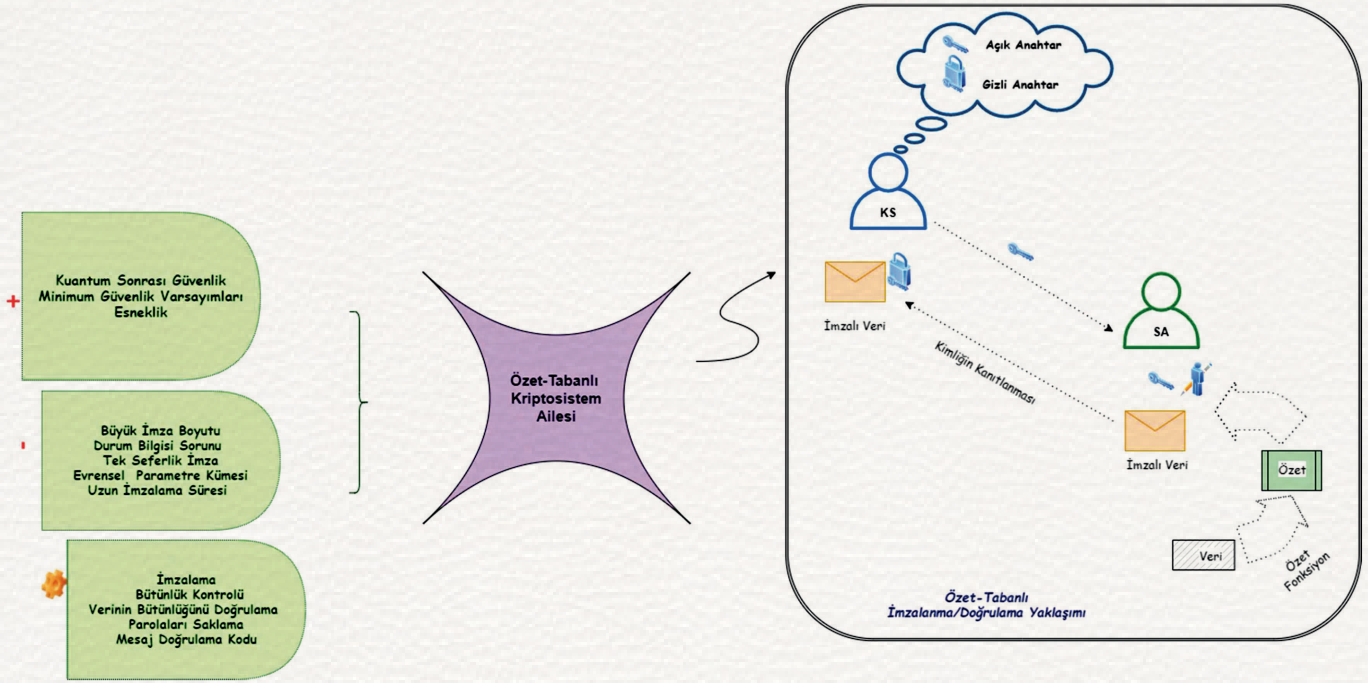
1989'da Ralph Merkle, Lamport imzalarından yola çıkarak özet-tabanlı ilk imzalama şeması önermişlerdir [40]. Güvenliğinde tek-yönlü bir fonksiyonun gerekli ve yeterli olması yaklaşımını kullanılan bu şemada temel fikir, birçok tek seferlik doğrulama anahtarının geçerliliğini bir açık anahtarın geçerliliğine indirgeyen bir özet ağaç kullanılmasıdır. Merkle ağacı olarak adlandırılan bu ikili özet ağacında yapraklar, tek seferlik imzalama yaklaşımında yer alan açık anahtarların özet değerlerinden oluşturulmuştur. Her bir

iç düğüm ise iki alt düğümünün birleştirilmesi sonucunun özetinin alınması ile hesaplanmıştır. Bu yapıda çakışmaya dirençli bir özet fonksiyonu kullanılarak kök düğümün tüm yaprak düğümleri, tek seferlik imzalama yaklaşımında yer alan açık anahtarların doğrulanmasında kullanılmıştır [41]. Merkle ağacı kullanılarak oluşturulan bu şema açık doğrulama anahtarı ve gizli imza anahtarının yalnızca tek bir belgenin imzalanması ve doğrulanmasında kullanılması dezavantajına sahiptir. Bu yaklaşımın orijinal hali RSA imza şemasına kıyasla yeterince verimli olmasa da literatürde yer alan özet-tabanlı imzalar, RSA ve eliptik eğri imza şemalarına alternatif olarak ön plana çıkmıştır [20].

Özet-tabanlı imza şemaları diğer herhangi bir elektronik imza şemasında olduğu gibi kriptografik özet fonksiyonlar kullanılarak güvenliği özet fonksiyonunun çakışma direncine dayandırılarak açıklanır. Kriptografik bir özet fonksiyonunda olması gereken temel üç özellik vardır. Bunlar:

- *Çakışmaya Dayanıklılık:* Aynı özet değeri veren iki mesajı bulmak hesaplamalı olarak imkansızdır.
- *Ters Görüntüye Dayanıklılık:* Verilmiş bir özet değerine sahip mesajı bulmak hesaplamalı olarak imkansızdır.
- *İkinci Ters Görüntüye Dayanıklılık:* Aynı özet değere sahip iki ayrı mesajın bulunması hesaplamalı olarak zordur.

Bu özellikleri ile keyfi uzunluktaki mesajları işlemek için elektronik imzalama şemalarında aktif olarak kullanılan özet fonksiyonlar açık anahtarlı kriptografinin temel yapı taşlarından birini oluşturmuştur. Her yeni kriptografik özet fonksiyondan yeni özet-tabanlı imzalama şemasının oluşturulması yaklaşımı güvenli imza şemalarının tasarımında sayılar teorisindeki zor problemlerden bağımsızlığı açıklar. Kuantum bilgisayarların hesaplama gücüne karşı direnci ile ilgili kesin kanıtı olmamasına rağmen, güvenlik gereksinimleri minimum düzeyde olan özet-tabanlı kriptosistem ailesi kuantum sonrası dünya için güvenli adaylardan biri olarak değerlendirilir. Özet-tabanlı kriptosistemlerin temel özellikleri ve genel imzalama/doğrulama yaklaşımı Şekil 6.4'te ifade edilmiştir.



Şekil 6.4. Özet-Tabanlı Kriptosistem Ailesi

İyi anlaşılmış güvenlik varsayımları ile kuantum-dirençli adaylar arasında ön plana çıkan özet-tabanlı imza şemalarının standartlaşma süreçlerinde önemli bir yeri vardır. Bu bağlamda elektronik imzaların kuantum sonrası güvenliği için NIST'in kuantum sonrası açık anahtarlı [8] ve elektronik imzalar için standart belirleme [14] süreçlerinde durum bilgili ve durum bilgisiz özet-tabanlı imza şemaları yer almıştır. Tek seferlik imza şemalarının kullanılması ile ortaya çıkan durumsallık (statefulness) kavramı durumların somut yönetiminden kaynaklanan sistem güvenliği sorunlarının ele alınması gerekliliğini ortaya çıkarmıştır. Durum bilgisi içeren imzalama şemalarının ihtiyaç duyduğu durum gereksinim yönetimi bu şemaların genel amaçlı kullanımını engellese de kısıtlı cihazların doğrulanması gibi özel durumlar için tercih edilir özelliktedir. Bu kapsamda NIST, Ekim 2020'de durum bilgisi içeren özet-tabanlı LMS, HSS, XMSS ve multi-tree XMSS imza şemalarını kuantum sonrası kriptografi için standartlaştırdığını duyurmuştur [15]. Genel kullanım durumları için ise kuantum sonrası açık anahtarlı kriptosistemlerin standartlaştırılması sürecinde durum bilgisiz özet-tabanlı SPHINCS+ imza şeması son tura kalmaya hak kazanan alternatif adaylardan biri olmuştur.

Bu bölümde genel özellikleri açıklanan kriptosistem aileleri, kuantum bilgisayarlar sonrası güvenli kriptosistemlerin oluşturulmasına temel olmuştur. Bu kapsamda NIST tarafından başlatılan kuantum dirençli açık anahtarlı kriptosistemlerin belirlenmesi sürecinde son tura geçmeye hak kazanan aday algoritmalar Bölüm 6.3'te özetlenmiştir.

6.3. NIST STANDARTLAŞMA SÜRECİNDE FİNALİST/ALTERNATİF ŞEMALAR

Bu bölümde kuantum sonrası güvenli açık anahtarlı kriptosistemlerin standartlaştırılması sürecinde son aşamaya kalan finalist ve alternatif algoritmaların değerlendirilmesinde ele alınan temel kriterler açıklanmıştır. Ayrıca aday algoritmaların genel özellikleri ve diğer algoritmalardan farklılaşma nedenleri özetlenmiştir. Detaylı bilgi için [9], [10] raporları ve [11] süreç gereksinim belgesine bakılmalıdır.

Kuantum sonrası güvenli açık anahtarlı kriptosistemlerin standartlaştırılması süreci boyunca NIST, aday algoritmaların değerlendirilmesinde üç temel kriteri baz almıştır. Bu kapsamda aday algoritmalar güvenlik, maliyet ve performans, algoritma-uygulama özelliklerini temel alarak değerlendirmeleri gerçekleştireceğini açıklamıştır.

Güvenlik:

Açık anahtarlı kriptosistemlerin TLS, SSH, IKE, IPSec gibi internet protokolleri, sertifikalar, yazılım kodu imzalama, güvenli önyükleyiciler gibi birçok uygulamada kullanılacak olması bu kriterin en önemli faktör olarak değerlendirilmesine neden olmuştur. Güvenliğin ölçülmesinde imzalama ve şifreleme/anahtar paketleme şemaları için farklı güvenlik tanımları belirlenmiştir.

- *Şifreleme/Anahtar paketleme şemaları için güvenlik tanımları:* Ayırt edilememezlik altında uyarlanabilir seçilmiş şifreli metin saldırısı (indistinguishability under adaptive chosen ciphertext attack-IND-CCA2) ve ayırt edilememezlik altında seçilen düz metin saldırısı (indistinguishability under chosen plaintext attack-IND-CPA) karşısında anlamsal olarak güvenliği (semantically secure) sağlayan güvenlik kanıtlarının sunulması istenmiştir.
- *İmzalama şemaları için güvenlik tanımları:* Uyarlanabilir seçilmiş mesaj saldırısı altında varoluşsal taklit edilemezlik (existential unforgeability under chosen message attack-EUF-CMA) için güvenlik kanıtlarının açıklanması istenmiştir.

Tablo 6.3. NIST Kuantum Sonrası Açık Anahtarlı Kriptosistemleri Standartlaştırma Sürecinde Güvenlik Seviyeleri

Güvenlik Kategorisi	Güvenlik Seviyesi	Açıklama*
I	Kırılması en az AES-128 kadar zor	*128-bitlik bir anahtara sahip bir blok şifresinde kapsamlı anahtar araması*
II	Kırılması en az SHA-256/SHA3-256 kadar zor	*256-bitlik bir özet fonksiyonda çakışma araması*
III	Kırılması en az AES-192 kadar zor	*192-bitlik bir anahtara sahip bir blok şifresinde kapsamlı anahtar araması*
IV	Kırılması en az SHA-384/SHA3-384 kadar zor	*384-bitlik bir özet fonksiyonda çakışma araması*
V	Kırılması en az AES-256 kadar zor	*256-bitlik bir anahtarla bir blok şifresinde anahtar araması*

**Örneğin, kategori-I'de ilgili güvenlik tanımını ihlal eden herhangi bir saldırı, 128 bitlik bir anahtara sahip bir blok şifresinde AES-128 kapsamlı anahtar arama için gerekli olanlarla karşılaştırılabilir veya daha büyük hesaplama kaynakları gerektirmelidir.*

Bu güvenlik tanımlarının olası ihlal durumlarında algoritmaların hesaplama karmaşıklığının sınıflandırılması amacıyla tanımlanan beş temel güvenlik kategorisi Tablo 6.3'te hatırlatılmıştır. Ayrıca, mükemmel ileri gizlilik (perfect forward secrecy), yan kanal (side-channel) ve çok anahtarlı (multi-key) saldırılara karşı direnç ve kötüye kullanıma karşı dayanıklılık gibi güvenlik özelliklerinden de bahsedilmiştir. Farklı hesaplamalı olarak zor problemlerin varsayımlarını uzun vadeli güvenlik hedefi olarak belirleyen NIST, bu varsayımları temel alan farklı kriptosistem ailelerinden pratik olarak verimli olanların standartlaştırılması beklenmektedir.

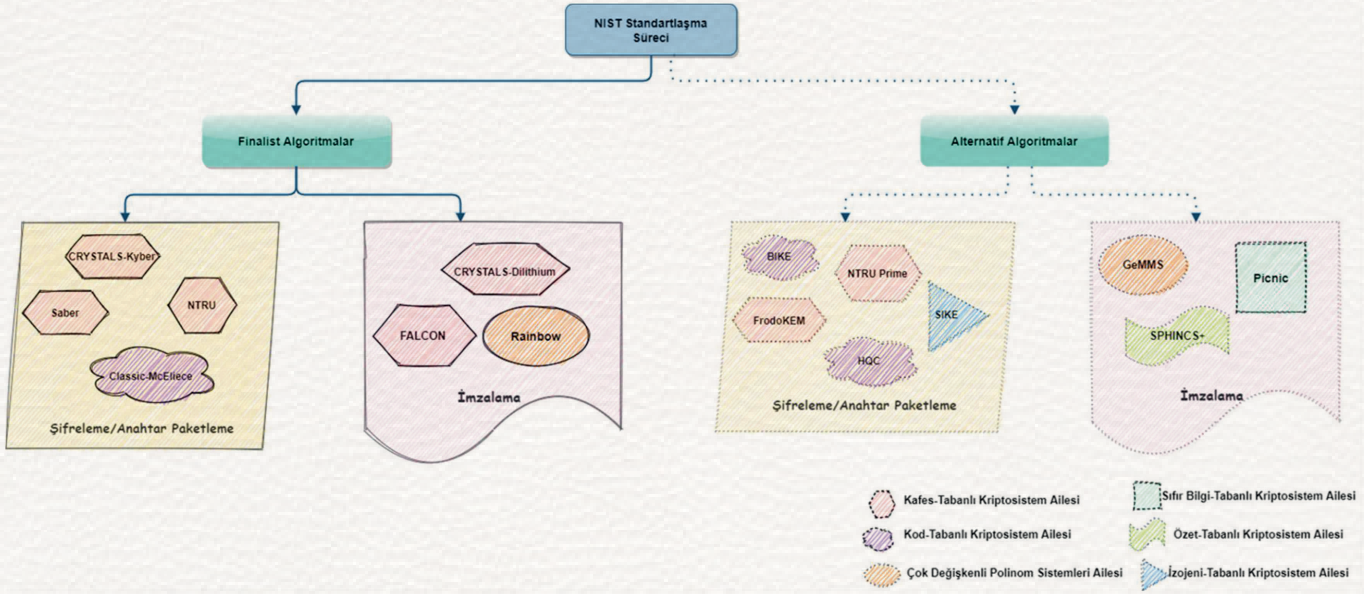
Maliyet ve Performans:

En önemli ikinci değerlendirme kriteri olarak ifade edilir. Maliyet ile anahtar üretimi, açık/gizli anahtar işlemlerinde hesaplama verimliliği, açık anahtar ve imza/şifre metni iletim maliyetleri, RAM ve kapı sayıları açısından uygulama maliyetlerinin incelenmesi istenmiştir. Maliyet ile elde edilecek performans verilerinin, zamanlama, güç izleme, hata saldırıları gibi yan kanal saldırılarına karşı koruma sağlayan uygulamaları da içermesi beklenmiştir. Genel performans karşılaştırmasında ise hem hesaplama maliyeti hem de veri aktarım maliyeti dikkate alınmıştır.

Algoritma ve Uygulama Özellikleri:

Aday algoritmalar için bu kriterde esneklik, basitlik ve algoritmanın yaygın olarak benimsenmesini engelleyebilecek faktörler ele alınmıştır. Esneklik ile çeşitli platformlarda verimli bir şekilde çalışabilen algoritmaları ve daha yüksek performans elde etmek için paralellik veya komut kümesi uzantılarını kullanan algoritmaların ön plana çıkacağı ifade edilmiştir. Basitlik ile algoritmalarda yer alan fikrin daha kolay ve iyi anlaşılmasına olanak sağlayan detaylı analiz içeren sade tasarımların tercih edileceği açıklanmıştır. Son olarak NIST, bir algoritma veya uygulamayı kapsayan fikri mülkiyet ve ilgili taraflar için kullanılabilirlik ve lisans koşulları dahil bir algoritmanın benimsenmesini engelleyebilecek veya teşvik edebilecek tüm faktörleri dikkate alacağını ifade etmiştir.

Açıklanan temel ve ek kriterler incelenerek NIST tarafından son tura geçmeye hak kazanan kuantum sonrası güvenli açık anahtarlı finalist ve aday kriptosistemler Şekil 6.5'te özetlenmiştir.



Şekil 6.5. NIST Kuantum Bilgisayarlar Sonrası Açık Anahtarlı Kriptosistem Standartlaşma Süreci Finalist ve Alternatif Algoritmalar

Şekil 6.5'te özetlenen kriptosistemler ve temel özellikleri Bölüm 6.3.1, Bölüm 6.3.2, Bölüm 6.3.3 ve Bölüm 6.3.4'te detaylandırılmıştır.

6.3.1. NIST Finalist Şifreleme ve Anahtar Paketleme Şemaları

Finalist şifreleme ve anahtar paketleme şemaları kod-tabanlı Classic-McEliece ve kafes-tabanlı CRYSTALS-Kyber, NTRU ve Saber kriptosistemleridir.

6.3.1.1. Classic-McEliece

Classic-McEliece [42], 1979 yılında önerilen kod-tabanlı McEliece şifreleme sistemine dayanan ikili Goppa kodlarını kullanan KEM şeması olarak tasarlanmıştır. Verimlilik ve CCA güvenliği sağlamak için iyileştirmeler içeren bu algoritma ikinci turda yer alarak aynı kodları kullanan NTS-KEM [43] algoritması ile birleşerek üçüncü tura geçmiştir. Yapılan iyileştirmelerin güvenliği McEliece şifreleme sisteminin seçilmiş düz metin saldırı güvenliğine karşı tek yönlü olmasına (one-wayness against chosen-plaintext-OW-CPA) indirgenerek açıklanmıştır. Buradaki zorluk rastgele ikili Goppa kodunun kod çözümünün zor olmasıdır. Classic-McEliece algoritması ikili Niederreiter [44] OW-CPA açık anahtarlı McEliece şifreleme şemasını temel almıştır ve Dent tarafından önerilen yaklaşım ile IND-CCA2 güvenli KEM'e dönüştürülmüştür. Oluşturulan şemanın kuantum rastgele kahin modelinde (quantum-accessible random oracle model-QROM) IND-CCA2 güvenlik analizi gerçekleştirilmiştir. Olumlu ve olumsuz temel özellikleri şu şekilde özetlenebilir:

+ Bu kriptosisteme duyulan güven Goppa kodun yaklaşık kırk yıldır çeşitli saldırılar karşısında iyileştirmelerle sağlam kalmasına dayanmaktadır.
+ Şifre metin boyutu oldukça küçüktür. Rakip KEM'lere göre en küçük şifre metin boyutuna sahiptir.
– Açık anahtar boyutu oldukça büyüktür. Daha açık bir ifadeyle, en yüksek güvenlik seviyesi için 1MB'den fazla olan açık anahtar boyutuna sahiptir.

6.3.1.2. CRYSTALS-Kyber

Kafes-tabanlı kriptosistem ailesinde bir KEM olarak tasarlanan CRYSTALS-Kyber'ın [45] güvenliği zor kafes problemlerinden modüller üzerinde tanımlı hatalar ile öğrenme (module learning with errors-MLWE) ve yuvarlama ile öğrenme (module learning with rounding-MLWR) problemlerinin bir kombinasyonuna dayandırılmıştır. Modül yapısının oluşturulmasında sayı teorik dönüşümü (number theoretic transform-NTT) içeren iki devirli halka yapısı temel alınmıştır. MLWE standart zor kafes problemlerine kıyasla daha yeni bir problem olsa da LWE problemine dayandırılması ve LWE'ye karşı herhangi bir saldırının bilinmemesi bu kriptosistemin güvenlik garantisini sağlamlaştırmıştır. Bu KEM'in oluşturulmasında 2004 yılında Regev'ın [46] açık anahtarlı kriptosistem tasarımı konusundaki fikri temel alınmıştır. Farklı olarak polinom vektörleri kullanılmış ve şifre metin üzerinde ek sıkıştırma yapılmıştır. Fujisaki-Okamoto dönüşümünün bir çeşidi kullanılarak IND-CCA güvenliği tanımlanan CRYSTALS-Kyber ayrıca QROM'da bir güvenlik kanıtıyla desteklenmiştir. Temel özellikleri şu şekildedir:

+ Modül yapısında polinom çarpımı için kullanılan NTT, hızlı hesaplamalar ile çoğu uygulama için çok yönlü performans ve uygulama kolaylığı sağlar.
+ Göreceli olarak küçük parametre boyutlarına ihtiyaç duyar.
+ İyi çalışılmış zor kafes problemi ile güçlü güvenlik garantisi sunar.
+ Polinom çarpımları tüm güvenlik seviyeleri için aynı halkada gerçekleştirildiği için güvenlik seviyeleri arasında ölçeklendirme kolaydır.
– Diğer kafes-tabanlı şemalara kıyasla NIST güvenlik kategorisi-I'in elde edilmesinde nispeten daha düşük düzeyde CoreSVP güvenliği sağlar.
– NTT alanında üretilen ve sıkıştırılan öğeler içermesi nedeniyle diğer çarpma algoritmalarının kullanılması pratikte zordur.

6.3.1.3. NTRU

Kafes-tabanlı bir diğer KEM şeması da yapılandırılmış kafesleri kullanan NTRU'dur [47]. İkinci turda benzerliklerinden dolayı ilk tur aday algoritmalarından NTRUEncrypt ve NTRU-HRSS-KEM birleşimi ile oluşturulan bu kriptosistem diğer kafes-tabanlı aday kriptosistemler gibi RLWE veya MLWE

problemlerinin zorluk varsayımını temel almaz. Standart NTRU yaklaşımını temel alan bu kriptosistemin güvenliğinin açıklanmasında en kötü durumdan ortalama duruma indirgeme gerçekleştirilmese de çok fazla çalışılmış ve analiz edilmiş yapısı güvenliğini açıklamaktadır. Tamamen doğru, kararlı bir açık anahtarlı şifreleme şemasından (deterministic public key encryption-dPKE) KEM oluşturmak için [48]'de önerilen OW-CPA şemadan IND-CCA2 güvenli KEM tasarımı yaklaşımı kullanılmıştır. Standart NTRU'nun bir çeşidi olarak tasarlanan NTRU mükemmel doğruluğu sağlar ve QROM'da bir güvenlik kanıtı içerir. Temel özellikleri şu şekilde özetlenebilir:

+ Güvenlik varsayımı iyi çalışılmış bir problemi temel alır.
+ Tasarımında kullanılan dPKE ile çeşitli kullanım durumları için parametrelendirilebilirliği ile esneklik sunar. Başka bir deyişle, farklı kullanım durumları için boyut-güvenlik-verimlilik ihtiyaçlarının karşılanması esnekliğine sahiptir.
+ İçerdiği parametre sayısının azlığı ile basitlik sunar.
+ NTRU'nun literatürde çok çalışılmış olması beklenmedik fikri mülkiyet iddiaları karşısında daha az riskin oluşmasına olanak sağlamıştır.
– Çok verimli olsa da tam olarak en yüksek performanslı kafes şemaları düzeyinde özellikler sunamaz. Başka bir deyişle, RLWE ve MLWE'ye dayalı kafes-tabanlı şemalardan daha yavaş anahtar üretimine sahiptir.
– CRYSTALS-Kyber ve Saber ile karşılaştırıldığında performans açığı oldukça azdır.

6.3.1.4. Saber

Saber [49], güvenliği MLWR probleminin zorluk varsayımını temel alan kafes-tabanlı IND-CPA güvenli şifreleme şeması ve IND-CCA güvenli KEM şemasını içeren kriptosistemler ailesidir. IND-CCA güvenliği Fujisaki-Okamoto dönüşümünün bir çeşidi tarafından sağlanan Saber, içerdiği kendine özgü NTT olmayan çarpma stili ile modüler indirgeme ve polinom çarpma adımlarının verimli optimizasyonuna olanak sağlar. Temel özellikleri şu şekildedir:

+ İyi çalışılmış zor kafes problemi ile güçlü güvenlik garantisi sunar.

+ Yuvarlamalara dayalı hata yapısı ek hata terimlerinin oluşmasını engelleyerek iletişim bant genişliğini azaltır.
+ Mod değerinin seçimi ile ortaya çıkan güç ve hata teriminin yokluğu maskeleme işlemlerinde kolaylık sağlar.
+ Mod değerinin seçiminde kullanılan yaklaşım, tipik olarak asal modüllere dayalı şemalarda mevcut olan açık modüler indirgeme veya reddetme örneklemesine olan ihtiyacını ortadan kaldırır. Ayrıca özet fonksiyonu çağrılarının sayısının azalmasına olanak sağlar.
+ Çarpma işlemlerinde bir bileşenin her zaman küçük katsayılara sahip olması özelliği uygulamaları optimize etmede kullanılmasına imkan tanır.
+ Tasarımı, uygulamaları belirli bir çarpma algoritmasıyla sınırlamaz.
– Mod değerinin seçiminde kullanılan yaklaşım NTT'nin doğal olarak desteklenmesine engel oluşturur. Ancak Karatsuba ve Toom-Cook gibi çarpma algoritmaları desteklenebilmektedir.

6.3.2. NIST Alternatif Şifreleme ve Anahtar Paketleme Şemaları

Alternatif olarak seçilen şifreleme ve anahtar paketleme şemaları kod-tabanlı BIKE ve HQC, kafes tabanlı FrodoKEM, NTRU Prime ve izojeni-tabanlı SIKE kriptosistemleridir.

6.3.2.1. BIKE

BIKE [50], yarı-döngüsel orta yoğunluklu eşlik kontrolü (quasi-cyclic moderate density parity-chec-QC-MDPC) kodlarına dayanan kod-tabanlı bir KEM şemasıdır. Klasik McEliece'de olduğu gibi açık anahtar, bir hata düzeltme kodunu belirtir, ancak farklı olarak kod, açık anahtarın sıkıştırılmasına izin veren yarı döngüsel bir genel yapıya sahiptir. Temel olarak TLS gibi geçici anahtarlarla eşzamanlı iletişim sağlayan protokollerde, başka bir deyişle, her anahtarın değişim oturumu için yeni bir açık/gizli anahtar çiftinin kullanıldığı durumlar için tasarlanmıştır. Bu yaklaşım belirli bir gizli anahtarla paket çözmeye yalnızca bir kere izin verildiği için ileri gizlilik

kavramının elde edilmesine olanak sağlamıştır. Bazı ince ayarlarla Niederreiter çerçevesi üzerine inşa edilen BIKE'ın KEM tasarımında açık anahtarlı şifreleme şemasından IND-CCA güvenli KEM'e dönüşümünde Fujisaki-Okamoto dönüşümünün bir versiyonunu kullanılmıştır. Genel kullanım için dengeli performans sunan yapılandırılmış kafes-tabanlı KEM'lere benzese de daha yavaş paket çözme-anahtar oluşturma ve daha fazla bant genişliği (açık anahtar boyutu + şifreli metin boyutu) sunan yapılandırılmış bir kod-tabanlı KEM olma özelliğini taşır. BIKE çeşitli güvenlik endişelerinin ortadan kaldırılmasında ihtiyaç duyacağı zaman nedeniyle alternatif sistem olarak değerlendirilmiştir.

6.3.2.2. HQC

Hamming Quasi-Cyclic (HQC) [51], parite problemli karar verme yarı döngüsel sendrom kod çözme probleminin zorluk varsayımına dayanan kod-tabanlı bir KEM şemasıdır. Birçok kafes-tabanlı şifreleme sistemi ile aynı gürültülü Diffie-Hellman yapısına sahip olan HQC şemasında Reed-Muller ve Reed-Solomon kodlarından oluşturulan hata düzeltme kodları kullanılmıştır. CCA2 güvenliği ve şifre çözme başarısızlık oranının titiz bir analizi sunulan bu kriptosistemin performans özellikleri kafes-tabanlı adaylar karşısında etkin değildir. Ancak kullanılan kodun gizli yapısını elde etmeyi amaçlayan saldırılara karşı gücü, küçük açık anahtar boyutu, şifre çözme başarısızlık oranının tahmin edilebilirliği ve klasik kod çözme algoritmalarına dayalı etkin uygulamaları HQC'nin NIST tarafından üçüncü turda alternatif bir aday olarak değerlendirilmesine olanak sağlamıştır.

6.3.2.3. FrodoKEM

FrodoKEM [52], güvenlik varsayımı LWE probleminin zorluğuna dayandırılan kafes-tabanlı KEM şemasıdır. 2011 yılında Lindner ve Peikert [53] tarafından önerilen yaklaşımın özel bir hali olan bu şemanın QROM'da güvenlik kanıtıyla desteklenen CCA güvenliği Fujisaki-Okamoto dönüşümü ile sağlanmıştır. Standartlaşma sürecinde yer alan diğer kriptosistemlere göre daha az miktarda bileşen içermesi olası cebirsel saldırılara karşı direncini artırmıştır. Ayrıca diğer şemalardan farklı olarak modül veya halka ile yapılandırılmamış bir zorluk varsayımının kullanılması performans açısından dezavantajlara

neden olmuştur. Ancak, yüksek trafikli TLS sunucularında gözle görülür ve etkili bir performansa sahip olacağı tahmin edilen FrodoKEM, yüksek gizlilik ve güvenin performanstan daha önemli olduğu kullanım durumları için tercih edilebilir bir şema olacaktır. Standartlaşma sürecinde KEM seçiminde yaygın olarak kullanılan uygulamalarda kabul edilebilir performansa sahip olacak bir şemanın seçilme önceliği FrodoKEM'in alternatif bir sistem olarak değerlendirilmesine neden olmuştur.

6.3.2.4. NTRU Prime

NTRU Prime [54], cebirsel yapı açısından farklılık gösterse de birçok tasarım öğesini paylaşan Streamlined NTRU Prime ve NTRU LPRime kafes-tabanlı KEM'lerin birleşiminden oluşmaktadır. Streamlined NTRU Prime, NTRU benzeri bir şema olduğu için klasik NTRU varsayımı altında uzun ve yerleşik bir kriptanalize karşı güvenlidir. NTRU LPRime ise Lyubashevsky, Peikert ve Regev'in RLWE benzeri KEM yapısını [55] temel alan bir tasarım modeli içermiştir. Bu özelliği nedeniyle CRYSTALS-Kyber ve Saber şemalarına benzer performans özellikleri sunmaktadır. Şifre çözme hatası olmayan NTRU Prime şifre çözme hataları nedeniyle ortaya çıkan saldırılar karşısında dirençlidir. Fujisaki-Okamoto tipi bir dönüşüm aracılığıyla CCA güvenliği oluşturulan NTRU Prime standartlaşma sürecinde alternatif şemalardan biri olarak üçüncü tura geçmeye hak kazanmıştır.

6.3.2.5. SIKE

SIKE [56], eliptik eğrilerin izojenilerine dayanan izojeni-tabanlı bir KEM şemasıdır. Güvenliği temelde süpersingular izojeni grafiklerde sözde rastgele yürüyüşlerin zorluğu problemine dayandırılır. Bu problem diğer problemlere göre kriptografik dünyada yeni bir problem olsa da süpersingular eliptik eğrilerin izojenilerini inceleme problemi oldukça eski bir matematik problemidir. Her ne kadar performans açısından diğer şemalarla yarışamasa da SIKE şemasını diğer şemalardan ayıran özellik izojeni-tabanlı şemaların yapısından kaynaklı küçük açık ve şifreli anahtar boyutu sunmasıdır. Bu özellikleri ve geliştirilebilecek olası iyileşmelere açık olması nedeniyle NIST tarafından alternatif sistemlerinden biri olarak seçilmiştir.

6.3.3. NIST Finalist İmzalama Şemaları

Finalist imzalama şemaları kafes-tabanlı CRYSTALS-Dilithium, FALCON ve çok değişkenli polinom sistemleri-tabanlı Rainbow şemalarıdır.

6.3.3.1. CRYSTALS-Dilithium

Üçüncü turdaki finalist iki kafes-tabanlı imza şemasından biridir. Güvenliği, zor kafes problemlerinden MLWE ve modül kısa tamsayı çözüm (module short integer solutions-MSIS) problemlerine dayandırılmıştır [57]. Fiat-Shamir dönüşümü kullanılarak kimlik şemasından bir imza şeması tasarımı yaklaşımı kullanılarak oluşturulan bu şemanın iç yapısında düzgün dağılım yoluyla tüm parametre kümeleri ve örnekler için aynı modül ve halka kullanılmıştır. Elektronik imzalar için standart güvenlik kavramı her ne kadar seçilen mesaj saldırıları altında güvenliğin (unforgability against chosen-message attacks-UF-CMA) kanıtlanması yeterli olsa da CRYSTALS-Dilithium seçilmiş mesaj saldırıları altında güçlü sahtecilik (strong unforgeability under chosen message attacks-SUF-CMA) açısından da güvenlik analizi değerlendirmelerini içermektedir. İkinci turda yer alan versiyonunda, belirleyici olmayan bir imza oluşturma seçeneği eklenmiştir. Ayrıca SHAKE yerine AES kullanmaya dayalı bir uygulama ile donanım talimatlarının gelecekteki faydaları gösterilmiştir. Temel özellikleri şu şekildedir:

+ Anahtar/imza boyutları ve anahtar oluşturma-imzalama-doğrulama işlemlerinin verimliği açısından güçlü ve dengeli performans sunar.
+ Pratikte iyi performans gösterir.
+ Parametre kümeleri ve örneklerin seçiminde aynı halka ve modül yapısının kullanımı FALCON'a kıyasla daha basit uygulamanın elde edilmesine olanak sağlamıştır.
+ Kararlı (başka bir deyişle, belirli bir mesajın imzası her zaman aynı olan) veya rastgele imzalar üretme seçeneği sunar.
+ NTT'ye uyarlanabilir parametreleri sayesinde çarpma işlemleri verimli bir şekilde gerçekleştirilebilir.
+ – Diğer finalist kafes-tabanlı imzalama şemaları ile karşılaştırıldığında hem hız açısından en yavaş hem de anahtar ve imza boyutu açısından en küçük değerlere sahiptir.

6.3.3.2. FALCON

FALCON [58], Gentry-Peikert-Vaikuntanathan (GPV) [59] tarafından önerilen yaklaşımı temel alan kafes-tabanlı imzalama şemasıdır. Özet ve imza (hash and sign) yaklaşımını kullanan bu şemanın güvenliği SIS probleminin NTRU kafesleri üzerindeki zorluğuna dayandırılmıştır. İmzalama süresinin iyileştirilmesi amacıyla Hızlı Fourier örnekleme kullanan bu imzalama şemasının güvenlik analizi hem rastgele kahin modeli (random oracle model-ROM) hem de QROM'da gerçekleştirilmiştir. Temel tasarım amacı tüm aritmetik işlemlerin verimli Fourier dönüşümü teknikleri kullanılarak gerçekleştirilmesi olan bu şemanın temel özellikleri şu şekildedir:

+ Standartta kullanılan şemalara ve uygulamalara uyarlanabilirlik açısından çok iyi performans ortaya koyar.
+ İkinci turda yer alan imzalama şemalarına kıyasla en küçük bant genişliğini başka bir deyişle, açık anahtar ve imza boyutu sunar.
+ Güvenliği güçlü varsayımlara dayandırılan etkili bir imza şemasıdır.
+ – Anahtar oluşturma işlemleri çok fazla zaman alsa bile imzalama ve doğrulama aşamalarında daha verimlidir.
– Diğer kafes-tabanlı imzalama şeması CRYSTALS-Dilithium ile karşılaştırıldığında içerdiği ağaç veri yapısı, kayan nokta işlemleri ve birden fazla ayrık Gauss dağılım kullanımı özellikleri ile daha karmaşıktır.
– Yapısında bulunan kayan nokta aritmetiğinin kullanımı uygulama hatalarının ortaya çıkmasında diğer şemalara göre etkisi bilinmemektedir.
– NIST tarafından önerilen güvenlik seviyelerinde kategori-I parametreleri CoreSVP açısından göreceli olarak zayıftır.

6.3.3.3. Rainbow

Rainbow [60], UOV imza şemasına dayanan katmanlı bir yapıya sahip çok değişkenli bir imzalama şemasıdır. İlk olarak her biri farklı parametrelere sahip birden çok katmana izin veren UOV'nin bir genellemesi olarak 2005 yılında önerilmiştir [61]. Rainbow'da RSA imzalarına benzer şekilde, yalnızca gizli anahtarın sahibinin ön görüntüleri hesaplayabildiği bir tuzak kapısı

fonksiyonu kullanılmıştır. Bu fonksiyon en iyi, iki veya daha fazla yağ ve sirke tuzak kapılarının bileşimi olarak tanımlanmıştır. Temel tasarım yaklaşımı OV tuzak kapısının yinelenmesi ile saldırılara karşı daha dirençli ve daha verimli parametre seçimlerine olanak sağlayan bir yapının elde edilmesidir. 2005 yılında önerilen standart Rainbow imzalama şeması, yalnızca evrensel sahtecilik sağlarken standardizasyon sürecinde yer alan versiyonu EUF-CMA güvenliğini bir dönüşüm uygulayarak elde etmiştir. Temel özellikleri şu şekilde özetlenebilir:

+ Hızlı imzalama&doğrulama ve çok kısa imza boyutu sunar.
+ Çok küçük sonlu cisimler üzerinde yalnızca doğrusal cebire dayalı işlemler içermesi kriptografik açıdan ek kaynaklara ihtiyaç duymadan düşük maliyetli cihazlarda uygulamaya uygundur.
+ Daha yavaş imzalama sürelerinin kabul edildiği durumlarda, açık anahtar boyutunu neredeyse üç kat sıkıştırmak mümkündür.
+ Anahtarları çok sık gönderme gereksinimine sahip olan uygulamalar için küçük ve hızlı imzalar sunar.
+ Rainbow şemasını anlamak ve uygulamak için gerekli olan cebir bilgisinin minimum olması ve şemaya saldırmak için kullanılabilecek çok fazla sayıda plan yapısının olması basitliğe sebep olmuştur.
+ – Yapısında bulunan ek katmanlar açık imza yapısının güvenlik gücünü artırabilirken imzalayan ve doğrulayan kuruluşların ihtiyaç duyduğu verimliliği ve kaynakları etkiler.
– Yenilemeli tuzak kapısı yaklaşımı beraberinde getirdiği ek karmaşıklıkla bazı yeni saldırı stratejilerinin önünü açmaktadır.
– Göreceli olarak çok büyük açık anahtarlara sahiptir. Bu özellik halihazırda kullanılan algoritmaların yerini alacak genel amaçlı bir imzalama algoritması olmasını engeller.

6.3.4. NIST Alternatif İmzalama Şemaları

Alternatif olarak belirlenen imzalama şemaları çok değişkenli polinom sistemleri-tabanlı GeMSS, sıfır bilgi-tabanlı Picnic ve özet-tabanlı SPHINCS+ şemalarıdır.

6.3.4.1. GeMSS

GeMSS [62], büyük alan paradigması temel alınarak oluşturulan çok değişkenli imzalama şemasıdır. 1990'lı yıllarda ortaya çıkan HFEv yapısını kullanan bu şema EUF-CMA güvenliği sağlamada Feistel-Patarin yapısını kullanmaktadır. Standartlaşma sürecinde yer alan diğer imzalama şemalarına göre en küçük imza boyutunu sunarak göreceli olarak hızlı doğrulama gerçekleştirir. Kararlı ve iyi çalışılmış matematik problemine dayandırılması ile güvenlik garantisi sağlanan bu şema oldukça büyük açık anahtar boyutu ve imzalama sürelerinden muzdariptir. Her ne kadar GeMSS düşük kaliteli cihazlarda uygulama zorluğu yaşasa da farklı bir güvenlik varsayımına dayanması alternatif aday olarak seçilmesine olanak sağlamıştır.

6.3.4.2. Picnic

Picnic [63], herhangi yapılandırılmış veya sayı-teorik zorluk varsayımını temel almayan sıfır bilgi-tabanlı imzalama şemasıdır. Yapısında bulunan sıfır bilgi kanıtı, özet fonksiyonlar ve blok şifreleri gibi simetrik temel ilkeler ile kuantum sonrası için alternatif adaylardan biri olmuştur. Bu kapsamda güvenliği özet fonksiyonu varlığı ve LowMC blok şifresine dayandırılır. Picnic imzası, gizli anahtar bilgisinin etkileşimli olmayan sıfır bilgili bir kanıtıdır. İmzanın uzunluğu, şifreleme şemasının çarpımsal karmaşıklığına ve sıfır bilgili bir bilgi kanıtı oluşturmaya yönelik özel tekniğe bağlı olarak değişir. Küçük açık anahtar boyutu avantajı varken büyük imzalara ve yavaş imzalama/doğrulama mekanizmasına sahiptir. İçerdiği Fiat-Shamir veya Unruh dönüşümü ile interaktif olmayan hale getirilen bu şema standartlaşma sürecinde alternatif imzalama şemalarından biri olarak yerini almıştır.

6.3.4.3. SPHINCS+

SPHINCS+ [64], durum bilgisi içermeyen keyfi, güvenli bir kriptografik özet fonksiyonu kullanılarak oluşturulan özet-tabanlı bir imza şemasıdır. Güvenliği diğer şemalardan farklı olarak hesaplamalı olarak zor bir probleme dayandırılmaz, temel alınan özet fonksiyonunun güvenliğine dayandırılır. ROM'da güvenlik analizi gerçekleştirilen SPHINCS+, diğer imzalama şemalarına göre çok eski bir araştırma geçmişine sahip kriptografik özet fonksiyonları içerdiği için kırılma olasılığı çok düşüktür. Ataklar açısından kolay analiz edilebilen

bu şema, küçük açık anahtar boyutunun yanı sıra yerleşik yapı taşlarının yeniden kullanımı avantajlarına sahiptir. Tasarımındaki temel fikir, çok sayıda az zamanlı imza anahtar çiftinin doğrulanması olan bu şema büyük imza boyutu ve yavaş imzalama hızı açısından dezavantajlara sahiptir. Her ne kadar daha iyi performans gösteren adaylar var olsa da NIST, imza şemalarına olan güveninin sarsılabilmesi durumlarını göz önüne alarak alternatif sistemlerden biri olarak SPHINCS+'yı belirlenmiştir.

6.4. SONUÇ VE DEĞERLENDİRMELER

Güvenli haberleşmeye ihtiyaç duyan her alan için gereksinim olan şifreleme sistemleri, anahtar değişim/paketleme ve imzalama şemaları mevcut hesaplama gücü karşısında talep edilen güvenliği karşılamaktadır. Kuantum bilgisayar kavramının ortaya çıkışı ile elde edilen/edilecek olan hesaplama gücünün kullanımı halihazırda kullanılan açık anahtarlı kriptosistemlerin güvenliğini etkisiz hale getirecektir. Her ne kadar büyük ölçekli kuantum bilgisayar oluşturulmamış olsa da yeni kriptosistemlerin tasarlanması ve pratikte kullanılabilir hale gelmesi çok fazla zaman ve çaba gerektirmektedir. Bu nedenle son yıllarda yapılan çalışmalarda kuantum-dirençli açık anahtarlı kriptosistemlerin oluşturulabilmesi için hesaplamalı olarak zor, kuantum bilgisayarlarda bile polinom zamanda çözülemeyen problem sınıflarını içeren kriptosistem ailelerine yoğunlaşılmıştır. İçerdikleri bazı özellikler nedeniyle halihazırda kullanılan standartlarda tercih edilmeyen bu kriptosistem sınıfları kuantum bilgisayarlar sonrası dünya için önemli rol üstlenmektedir. Dolayısıyla, kriptosistem aileleri özelinde önerilen algoritmaların incelenmesi ve detaylandırılması neden-sonuç ilişkisi içerisinde alternatif sistemlerin tasarlanması açısından yol gösterici olacaktır. Bu kapsamda bu bölümde NIST tarafından başlatılan kuantum sonrası güvenli açık anahtarlı kriptosistemleri standartlaştırma sürecinde yer alan temel kriptosistem sınıfları üzerine bir inceleme gerçekleştirilmiştir. Temel özellikleri açıklanan bu aileler özelinde NIST değerlendirme kriterleri açıklanarak son tura geçmeye hak kazanan finalist/alternatif şifreleme/anahtar paketleme ve imzalama şemalarının güvenlik varsayımları, üstünlükleri ve eksik yanları özetlenmiştir.

Ülkemizde kuantum sonrası güvenli iletişim için yapılması önerilenler şu şekilde özetlenmiştir.

- Kuantum kriptografi ve kuantum sonrası kriptografi için algoritma, protokol ve mimarilerin geliştirilmesine yönelik **yol haritası**nın (**eylem planı**) hazırlanmasına ihtiyaç duyulmaktadır. Bu eylem planı içerisinde, destek mekanizmalarının, laboratuvarların ve Üniversite-Kamu-Sanayi işbirliklerinin somutlaştırılma adımlarının yer alması önemlidir.
- Kuantum bilgisayarların varlığının güvenli iletişimi aktif olarak etkileme varsayımı üzerine kuantum sonrası kriptografik algoritma, protokol ve mimarilere **göç planı**nın milli/yerli sistemler için hazırlanması gerekmektedir.
- Kuantum ataklara karşı zafiyet içeren RSA, Diffie-Hellman, ECDSA gibi klasik açık anahtarlı sistemleri kullanan Kamu ürünleri için **ihtiyaç analizi**nin yapılması ve kuantum sonrası güvenlik için **kontrol listesi**nin bağımsız araştırma grupları tarafından oluşturulması önemlidir.

Gelecekte yapılması planlanan çalışmada kuantum bilgisayarlar sonrası güvenli kriptosistem aileleri kullanılarak oluşturulabilecek yeni sistem tasarımında kullanılabilecek zor problemlerin, temel tasarım yaklaşımlarının ve yöntemlerin bir araya getirilerek incelenmesi hedeflenmektedir.

Teşekkür

Bu çalışma EEEAG-121R006 proje numarası ile TÜBİTAK tarafından desteklenmiştir.

KAYNAKLAR

[1]. M. Campagna et al., "Quantum Safe Cryptography and Security: An introduction, benefits, enablers and challenges," European Telecommunications Standards Institute, Fransa, ETSI White Paper No. 8, 2015.

[2]. P. W. Shor, "Algorithms For Quantum Computation: Discrete Logarithms and Factoring," In Proceedings 35th Annual Symposium on Foundations of Computer Science, pp. 124-134, 1994.

[3]. L. K. Grover, "A Fast Quantum Mechanical Algorithm For Database Search," In Proceedings of the Twenty-Eighth Annual Symposium on the Theory of Computing, pp. 212-219, Haziran 1996.

[4]. M. Cenk, "Siber Güvenlikte Kriptografi," Siber Güvenlik ve Savunma: Farkındalık ve Caydırıcılık Cilt II, 1st ed. Ankara, Türkiye: Grafiker Yayınları, ch. 2, pp. 64-87, 2019.

[5]. W. Beullens, "Post-Quantum Cryptography: Current State and Quantum Mitigation," Enisa, 2021. [Online] Available: https://www.enisa.europa.eu/publications/post-quantum-cryptography-current-state-and-quantum-mitigation

[6]. S. Akleylek, M. Soysaldı, "Kuantum Bilgisayarlar ile Kriptoanaliz ve Kuantum Sonrası Güvenilir Kripto Sistemleri," Siber Güvenlik ve Savunma: Farkındalık ve Caydırıcılık Cilt II, 1st ed. Ankara, Türkiye: Grafiker Yayınları, ch. 4, pp. 138-171, 2019.

[7]. M. Mosca, "Setting the Scene for the ETSI Quantum-safe Cryptography Workshop," ETSI, 2013. [Online] Available: http://docbox.etsi.org/Workshop/2013/201309_CRYPTO/e-proceedings_Crypto_2013.pdf

[8]. NIST, "Post - Quantum Cryptography Standardization Project," 2016. [Online]. Available: https://csrc.nist.gov/projects/post - quantum-cryptography

[9]. G. Alagic et al., "Status Report on the First Round of the NIST Post-Quantum Cryptography Standardization Process," Washington, DC: US Department of Commerce, National Institute of Standards and Technology, 2019. [Online]. Available: https://www.nist.gov/publications/status-report-first-round-nist-post-quantum-cryptography-standardization-process

[10]. G. Alagic et al. "Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process," 2020. [Online]. Available: https://csrc.nist.gov/publications/detail/nistir/8309/final

[11]. "Submission Requirements and Evaluation Criteria for the Post-Quantum Cryptography Standardization Process," 2016. [Online]. Available: https://csrc.nist.gov/csrc/media/projects/post-quantum-cryptography/documents/call-for-proposals-final-dec-2016.pdf

[12]. FIPS PUB 186-4, "Federal Informatıon Processıng Standards Publıcatıon Digital Signature Standard (DSS)," 2013. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf

[13]. "Stateful Hash-Based Signatures," [Online]. Available: https://csrc.nist.gov/Projects/stateful-hash-based-signatures

[14]. "Stateful Hash-Based Signatures: NIST Wants Your Input," [Online]. Available: https://csrc.nist.gov/news/2018/stateful-hash-based-signatures-nist-wants-input

[15]. D. A. Cooper et al., "Recommendation for Stateful Hash-Based Signature Schemes," NIST Special Publication 800-208, 2020. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-208.pdf

[16]. R. Hill, "A First Course in Coding Theory," Oxford University Press, 1968, pp. 165-173.

[17]. N. Sendrier, "Code-Based Cryptography: State of the Art and Perspectives," In IEEE Security & Privacy, vol. 15, no. 4, pp. 44-50, 2017.

[18]. J. H. Lint, "Introduction to Coding Theory," Springer, 1992, pp. 40-44.

[19]. R. J. McEliece. "A Public-Key Cryptosystem Based on Algebraic," Coding Thv., vol. 4244, pp. 114-116, 1978.

[20]. D. J. Bernstein, J. Buchmann, and E. Dahmen, "Post-Quantum Cryptography," Springer, 2008.

[21]. N. T. Courtois, M. Finiasz, and N. Sendrier, "How to Achieve a McEliece-based Digital Signature Scheme," In International Conference on the Theory and Application of Cryptology and Information Security, vol. 2248, pp. 157-174, 2001.

[22]. L. Chen et al., "Report on Post-Quantum Cryptography," US Department of Commerce, National Institute of Standards and Technology, NISTIR 8105, 2016. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/ir/2016/NIST.IR.8105.pdf

[23]. C. Aguilar, P. Gaborit, and J. Schrek, "A New Zero-Knowledge Code Based Identification Scheme with Reduced Communication," In 2011 IEEE Information Theory Workshop, pp. 648-652, 2011.

[24]. P. L. Cayrel, P. Véron, and S. M. E. Y. Alaoui, "A Zero-Knowledge Identification Scheme Based on the q-ary Syndrome Decoding Problem," In International Workshop on Selected Areas in Cryptography, vol. 6544, pp. 171-186, 2010.

[25]. P. Gaborit, M. Girault, "Lightweight Code-Based Identification and Signature," In 2007 IEEE International Symposium on Information Theory, pp. 191-195, 2007.

[26]. J. B. Fischer, J. Stern, "An Eficient Pseudo-Random Generator Provably As Secure As Syndrome Decoding," Advances in Cryptology - EUROCRYPT '96, vol. 1070, pp. 245-255, 1996.

[27]. P. Gaborit, C. Laudaroux, and N. Sendrier, "Synd: A Very Fast Code-Based Cipher Stream with a Security Reduction," In IEEE Conference ISIT'07, vol. 7, pp. 186-190, 2007.

[28]. D. Augot, M. Finiasz, and N. Sendrier, "A Family of Fast Syndrome Based Cryptographic Hash Functions," In Proc. of Mycrypt 2005, vol. 3715, pp. 64-83, 2005.

[29]. O. Billet, J. Ding, "Overview of Cryptanalysis Techniques in Multivariate Public Key Cryptography," In Gröbner Bases, Coding, and Cryptography, 2009, pp. 263-283. Accessed on: 04,27,2021, DOI: 10.1007/978-3-540-93806-4_15, [Online].

[30]. J. Ding, A. Petzoldt, "Current State of Multivariate Cryptography," IEEE Security & Privacy, vol. 15, no. 4, pp. 28-36, 2017.

[31]. T. Matsumoto, H. Imai, "Public Quadratic Polynomial-Tuples for Efficient Signature-Verification and Message-Encryption," In Workshop on the Theory and Application of of Cryptographic Techniques, vol. 330, pp. 419-453, 1988.

[32]. J. Patarin, "Hidden Field Equations (HFE) and Isomorphisms of Polynomials (IP)," In: Proceedings of EUROCRYPT'96, vol. 1070, pp. 38-48, 1996.

[33]. M. Ajtai, "Generating Hard Instances of Lattice Problems," Quaderni di Matematica (Preliminary version in STOC 1996), vol. 13, pp. 1-32, 2004.

[34]. S. Akleylek, K. Seyhan, "Kuantum Bilgisayarlar Sonrası Güvenilir Kafes Tabanlı Kriptosistem Temellerine Giriş," Siber Güvenlik ve Savunma: Farkındalık ve Caydırıcılık Cilt II, 1st ed. Ankara, Türkiye: Grafiker Yayınları, ch. 5, pp. 172-209, 2019.

[35]. H. Satılmış, S. Akleylek, "Kafes Tabanlı Kriptografide Kullanılan Zor Problemlerin Kriptanalizi ve Yazılım Kütüphaneleri," Siber Güvenlik ve Savunma: Farkındalık ve Caydırıcılık Cilt 4, 1st ed. Ankara, Türkiye: Nobel Yayınları, ch. 7, pp. 233-256, 2020.

[36]. C. Peikert, "A Decade of Lattice Cryptography," Foundations and Trends® in Theoretical Computer Science, vol. 10, no. 4, pp. 283-424, 2016.

[37]. O. Goldreich, S. Goldwasser, and S. Halevi, "Public-Key Cryptosystems From Lattice Reduction Problems," In CRYPTO, vol. 1294, pp. 112-131, 1997.

[38]. D. Micciancio, S. Goldwasser, "Complexity of Lattice Problems: A Cryptographic Perspective," Springer Science and Business Media, vol. 671, pp. 185-187, 2012.

[39]. L. Lamport, "Constructing Digital Signatures From A One-Way Function," Technical Report CSL-98, SRI International, 1979, [Online]. Available: https://www.microsoft.com/en-us/research/uploads/prod/2016/12/Constructing-Digital-Signatures-from-a-One-Way-Function.pdf

[40]. R. C. Merkle, "A Certified Digital Signature," In Conference on the Theory and Application of Cryptology, vol. 435, pp. 218-238, 1989.

[41]. A. Hülsing et al., "Hash-Based Signatures: An Outline for a New Standard," In NIST Workshop on Cybersecurity in a Post-Quantum World, 2015, [Online]. Available: https://csrc.nist.gov/csrc/media/events/workshop-on-cybersecurity-in-a-post-quantum-world/documents/papers/session5-hulsing-paper.pdf

[42]. D. J. Bernstein et al., "Classic McEliece: Conservative Code-Based Cryptography," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[43]. M. Albrecht et al., "Nts-kem," NIST Round2 Submissions, 2019, [Online]. Available: https://csrc.nist.gov/Projects/post-quantum-cryptography/round-2-submissions

[44]. H. Niederreiter, "Knapsack-Type Cryptosystems and Algebraic Coding Theory," In Problems of Control and Information Theory, vol. 15, no. 2, pp. 159-166, 1986.

[45]. R. Avanzi et al., "CRYSTALS-Kyber," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[46]. O. Regev, "New Lattice-Based Cryptographic Constructions," Journal of the ACM, vol. 51, no.6, pp. 899-942, 2004.

[47]. C. Chen et al., "NTRU," NIST Round3 Submissions, 2021, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[48]. D. Hofheinz, K. Hövelmanns, and E. Kiltz, "A Modular Analysis of the Fujisaki-Okamoto Transformation," In TCC 2017: 15th Theory of Cryptography Conference, vol. 10677, pp. 341-371, 2017.

[49]. JP. D'Anvers et al., "Saber: Module-LWR Based Key Exchange, CPA-Secure Encryption and CCA-Secure KEM," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[50]. N. Aragon, "BIKE: Bit Flipping Key Encapsulation," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[51]. C. A. Melchor et al., "HQC (Hamming Quasi-Cyclic)," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[52]. E. Alkim et al., "FrodoKEM Learning with Errors Key Encapsulation," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[53]. R. Lindner, C. Peikert, "Better Key Sizes (and Attacks) for LWE-Based Encryption," In Topics in Cryptology – CTRSA2011, vol. 6558, pp. 319-339, 2011.

[54]. D. J. Bernstein et al. "NTRU Prime," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[55]. V. Lyubashevsky, C. Peikert, and O. Regev, "On Ideal Lattices and Learning with Errors over Rings," In annual International Conference on the Theory and Applications of Cryptographic Techniques – EUROCRYPT, vol. 6110, pp. 1-23, 2010.

[56]. D. Jao et al., "SIKE," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[57]. L. Ducas et al. "CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[58]. T. Prest et al., "FALCON," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[59]. C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for Hard Lattices and New Cryptographic Constructions," In 40th Annual ACM Symposium on Theory of Computing, pp. 197-206, Mayıs 2008.

[60]. J. Ding et al., "Rainbow," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[61]. J. Ding, D. Schmidt, "Rainbow, A New Multivariable Polynomial Signature Scheme," In International Conference on Applied Cryptography and Network Security, vol. 3531, pp. 164-175, 2005.

[62]. A. Casanova et al., "GeMSS," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

[63]. D. Kales, G. Zaverucha, "Improving the Performance of the Picnic Signature Scheme," IACR Transactions on Cryptographic Hardware and Embedded Systems, vol. 2020, no. 4, pp. 154-188, 2020.

[64]. J. P. Aumasson et al., "SPHINCS," NIST Round3 Submissions, 2020, [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

Bölüm 7

AÇIK ANAHTARLI KRİPTOSİSTEMLER İÇİN VERİMLİ SIKIŞTIRMA UYGULAMALARI

Melek Çil - Barış Bülent Kırlar

Bu bölümde, verimli sıkıştırma algoritmalarının açık anahtarlı kriptosistemlerdeki uygulamalarından bahsedilecektir. Bunu yapabilmek için ilk olarak, d/k ve $n=k/d>1$ olmak üzere $\mathbb{F}_{q^d}$ sonlu cismi üzerindeki n. dereceden lineer yineleme bağıntısından istifade edilerek verimli sıkıştırma algoritmalarının nasıl tanımlandığı gösterilecektir. Sonrasında, bu verimli algoritmaların şifreleme ve imzalama şemalarındaki etkinliği ve eşleme tabanlı kriptografide son üs alma işlemindeki kullanışlılığı detaylı bir şekilde açıklanacaktır.

7.1. GİRİŞ

Sonlu cisimler üzerinde n. dereceden lineer yineleme bağıntısı kriptografide farklı uygulama alanlarına sahiptir. Bu bağıntı simetrik kriptografide, özellikle akışkan şifreleme sistemlerinde etkin uygulama alanlarına sahipken, asimetrik kriptografide, bir başka deyişle Açık Anahtarlı Kriptosistemlerde (PKC) hem verimli hesaplama, kompakt temsil ve arttırılmış güvenlik gibi özelliklere sahip olan diziler kullanılarak şifreleme ve imzalama algoritmaları tanımlanmasını, hem de eliptik eğri eşleme tabanlı kriptografide son üs alma işleminin yine aynı diziler sayesinde etkin olarak hesaplanmasını sağ-

lar. Biz bu çalışmada, n. dereceden lineer yineleme bağıntısının açık anahtarlı kriptosistemlerdeki verimli uygulamalarından bahsedeceğiz.

Bu konular ile ilgili temel çalışmalar ve kriptosistemler; $n=2$ için LUC-PKC [40,41], $n=3$ için GH-PKC [23,24], XTR-PKC [30,42], $n=5$ için GG-PKC [17,18], $n=6$ için Shirase ve arkadaşları [39] ve Karabina [28], $n=4$ için ise Karabina [28] tarafından yapılmıştır. Bu sistemlerin hepsinde $\mathbb{F}_{q^k}$ sonlu cisminin $G_{r,q,k}$ siklotomik alt grubu, $r \nmid k$ ve $r \mid \Phi_k(q)$ olacak şekilde r asal mertebeli bir alt grup olarak tanımlanır ve bu sistemlerde $G_{r,q,k} \subset \mathbb{F}_{q^k}^*$ nın elemanlarının sıkıştırılmış formlarının verimli algoritmaları kullanılarak kriptografik sistemler tanımlanır. Bu sistemlerde, cisim genişlemesinin güvenliği sağlanırken, grup işlemleri ara veya asal alt cisimlerde gerçekleştirilir.

Smith ve Lennon [40] tarafından $\mathbb{F}_q$ üzerinde 2. dereceden lineer yineleme bağıntısına dayalı olarak oluşturulan LUC-PKC, Smith ve Skinner [41] tarafından daha iyi bir hale getirilmiştir. Onlar, $\mathbb{F}_{q^2}^*$ de, $r \mid \Phi_2(q) = q+1$ olmak üzere, mertebesi r olan $G_{r,q,2}$ alt grubunun elemanlarının, $\mathbb{F}_q$ üzerindeki izleri ile sıkıştırma faktörü 2 olacak şekilde belirlenebileceğini göstermişlerdir. Ayrıca onlar, bu yapıyı kullanarak adına LUC-PKC denilen açık anahtarlı kriptosistemi tanımlarken, sistemin $\mathbb{F}_{q^2}$ güvenliğini sağladığını da göstermişlerdir.

Gong ve Harn, $r \mid \Phi_3(q) = q^2+q+1 \in \mathbb{F}_{q^3}^*$ olmak üzere mertebesi r olan $G_{r,q,3}$ alt grubunun elemanlarının 3/2 sıkıştırma faktörü ile belirlenebileceğini göstermişlerdir [23]. Ayrıca, bu elemanların sıkıştırılmış formları için etkili bir üs alma algoritması elde ederek, $\mathbb{F}_{q^3}$ güvenliğini sağlayan GH-PKC algoritmasını tanımlamışlardır.

Brouwer, Pellikaan ve Verheul, $\mathbb{F}_{q^6}^*$ da $r \mid \Phi_6(q) = q^2-q+1$ olmak üzere, mertebesi r olan $G_{r,q,6}$ alt grubunun elemanlarının 3 sıkıştırma faktörü ile

belirlenebileceğini göstermişlerdir [11]. Ancak, $G_{r,q,6}$ nın elemanlarının sıkıştırılmış formları için bir üs alma algoritması vermemişlerdir. 2000'de, Lenstra ve Verheul [30], $r|\Phi_6(q) = q^2 - q + 1 \in \mathbb{F}^*_{q^6}$ olmak üzere, mertebesi r olan $G_{r,q,6}$ alt grubunun elemanlarının, $\mathbb{F}_{q^2}$ üzerindeki izleri ve 3 sıkıştırma faktörü ile tam olarak temsil edilebileceğini gösterip, etkili bir üs alma algoritması vermişlerdir.

2003 yılında ise Giuliani ve Gong [17], $\mathbb{F}_q$ üzerinde 5. dereceden lineer yineleme bağıntısına dayanan bir sistem tanımlamışlar ve bu sistemin güvenliğinin $\mathbb{F}_{q^5}$ üzerinde tanımlanan ayrık logaritma problemini çözmenin zorluğuna dayandığını, ancak hesaplamaların $\mathbb{F}_q$ üzerinde gerçekleştiğini göstermişlerdir. Diğer taraftan onlar, $r|\Phi_{10}(q) = q^4 - q^3 + q^2 - q + 1 \in \mathbb{F}^*_{q^{10}}$ olmak üzere, r mertebeli $G_{r,q,10}$ alt grubunun elemanlarının $5/2$ sıkıştırma faktörü ile tam olarak temsil edilebileceğini göstermişlerdir. Ayrıca, $\mathbb{F}_{q^2}$ de hesaplamalar yapan ve $\mathbb{F}_{q^{10}}$ un güvenliğini sağlayan, 5. dereceden lineer yineleme bağıntısına dayanan GG-PKC yi önermişler [18] ve bu sistemin hesaplama maliyetlerini daha sonra yaptıkları çalışmada iyileştirmişlerdir [20].

Daha sonra, Stam ve Lenstra [42] ile Shirase ve arkadaşları [39] yaptıkları çalışmalarda sistemin altında yatan fikre sadık kalarak XTR da bazı ilerlemeler kaydetmişlerdir. Shirase ve arkadaşları [39], t tek tamsayı, $q = 3^t$ ve $r|q - \sqrt{3q} + 1 \in \mathbb{F}^*_{q^6}$ olmak üzere mertebesi r olan $G_{r,q,6}$ alt grubunun elemanlarının, $\mathbb{F}_q$ üzerinde 6 sıkıştırma faktörü ile tam olarak temsil edilebileceğini gösterip, verimli bir algoritma da sunmuşlardır. 2009 yılında Karabina [28], Shirase ve arkadaşlarının yöntemini kullanarak yine bazı ilerlemeler kat etmiş ve t bir tek tamsayı $q = 3^t$ ve $q \pm \sqrt{3q} + 1 \in \mathbb{F}^*_{q^6}$ olmak üzere, mertebesi r olan elemanların 6 sıkıştırma faktörü ile tek bir şekilde temsil edilebileceğini gösterip, etkin altı ayrı algoritma yazmıştır. Diğer taraftan, t bir tek tamsayı ve $q = 2^t$ ve $q \mp \sqrt{2q} + 1 \in \mathbb{F}^*_{q^4}$ olmak üzere, mertebesi r olan

elemanların 4 sıkıştırma faktörü ile tek bir şekilde temsil edilebileceğini gösterip, etkin beş ayrı üs alma algoritması tanımlamıştır.

2014 yılında Akyıldız ve Ashraf [2], iz tabanlı açık anahtarlı kriptosistemler üzerine bir değerlendirme çalışması yaparken, Ashraf ve Kırlar [3], Gong ve Harn'ın üçüncü dereceden lineer yineleme bağıntısını kullanarak geliştirdikleri alt yapıyı kullanarak yeni bir güvenli şifreleme ve imzalama algoritması tanımlamışlardır.

2015 yılında Akleylek ve Kırlar [1], XTR altyapısını kullanarak yeni bir şifreleme sistemi tanımlamışlar ve bu sistemin hem güvenlik hem de hesaplama maliyeti olarak benzer sistemlere göre daha etkin olduğunu göstermişlerdir.

2017 yılında Kırlar ve Çil [29], $d \mid k$ ve $n = k / d > 2$ olmak üzere $\mathbb{F}_{q^d}$ sonlu cismi üzerinde n. dereceden indirgenemez polinom tarafından üretilen lineer yineleme bağıntısının değişmelilik (bir ikili işlemin öğelerinin sırasına bağlı olmaması) yasasına (commutative law) dayanan yeni bir geçici statik şifreleme şeması sunmuşlardır. Bu şifreleme şemasının güvenliğinin, n. dereceden LFSR problemini çözmenin zorluğuna dayandığını gösterirken, aynı zamanda bu şemanın semantik olarak da güvenli olduğunu ispat etmişlerdir. Onlar, $n=2$ için Lucas ve Fibonacci dizilerinin değişmelilik yasasını kullanarak, $n>2$ için önerilen şema ile aynı güvenlik seviyesine sahip başka bir şifreleme şeması da önermişlerdir.

Diğer taraftan, $\mathbb{F}_q$ sonlu cismi üzerindeki n. dereceden lineer yineleme bağıntısı eliptik eğri eşleme tabanlı kriptografide de son üs alma işlemi olarak kullanılmaktadır. Miller (1985) ve Koblitz (1987) tarafından birbirinden bağımsız olarak ortaya çıkarılan eliptik eğri kriptosistemleri (ECC), kriptografik araştırmacılar tarafından çok ilgi görmüştür. ECC'nin bu çekiciliğinin ana nedeni, uygun şekilde seçilmiş bir eliptik eğri üzerindeki ayrık logaritma problemini çözmek için bilinen bir alt-üstel algoritmanın olmamasıdır. Bu durum, ECC de, RSA ve DSA'daki güvenlik seviyesi korunurken önemli ölçüde daha küçük boyutlarda anahtar kullanılabileceği anlamına gelir. Daha hızlı hesaplamalar, işlem gücündeki azalmalar, depolama alanı ve bant ge-

nişliği, daha küçük boyutta anahtara sahip olmanın bazı avantajlarıdır. Bu yüzden ECC; çağrı cihazları, kişisel dijital asistanlar (PDA), cep telefonları ve akıllı kartlar gibi kısıtlı ortamlar için idealdir. Son zamanlarda, eliptik eğriler eşleme tabanlı kriptografik protokollerde yaygın olarak kullanılmaya başlanmıştır. Bu protokoller, $k \leq 50$ olmak üzere, k gömme derecesine sahip belirli eliptik eğrilerden türetilmiş Weil ve Tate iki lineer (bilinear) eşlemelerine dayanmaktadır. Başlangıçta, bu tür protokolleri gerçekleştirmek için, uygun bir yapı olarak Weil eşlemesi önerilse de artık genellikle Tate çifti daha etkili olduğu için tercih edilmektedir [8].

Tate eşlemesinin hem süpersingüler eliptik eğriler hem de belirli sıradan (ordinary) eğriler üzerinde etkili bir şekilde hesaplanması sebebiyle her iki eğrinin de eşleme tabanlı şemalarda kullanımının eşit derecede uygun olduğu gösterilmiştir [5,4,16]. Esasında, sıradan eğriler güvenlik parametrelerinin seçiminde daha fazla esneklik sağlar [5,34]. Tate eşlemesinin ve türevleri olan Eta, Ate, R-Ate gibi eşlemelerin hesaplanması; Miller'ın eşleme değerini hesaplayan algoritması ve son üs alma algoritması olmak üzere iki ana bileşenden oluşur. Son üs alma işlemi; elde edilen eşleme çıktısının $(q^k - 1)/r$. kuvveti hesaplanarak $\mathbb{F}_{q^k}^*$ cisim genişlemesi içerisindeki r. birim kök grubunda tek bir eleman elde etme işlemine karşılık gelir. Bu işlemin yapılabilmesi için de yukarıda bahsettiğimiz n. dereceden lineer yineleme bağıntısı kullanılarak elde edilen verimli algoritmalar kullanılmaktadır.

Bölüm 7.2'de, $d \mid k$ ve $n = k/d > 1$ olmak üzere $\mathbb{F}_{q^d}$ üzerinde tanımlı n. dereceden bir karakteristik dizi ve özellikleri detaylı bir şekilde incelenecektir. 7.3. bölümde, eliptik eğri eşleme tabanlı kriptografi hakkında temel bilgilerden bahsedilecektir. 7.4. bölümde ise n'nin bazı değerleri için verimli sıkıştırma algoritmaları verilecektir. 7.5. bölümde, n. dereceden indirgenemez polinom tarafından üretilen n. dereceden lineer yineleme bağıntısının bazı özelliklerine dayanan geçici-statik bir şifreleme şeması detaylı bir şekilde incelenecektir. 7.6. bölümde ise kısaca sıkıştırılmış eşlemelerden bahsedilip, 7.7. bölümde ise çalışmaya ait sonuç ve değerlendirmeler sunulacaktır.

7.2. SONLU CİSİMLER ÜZERİNDE n. DERECEDEN LİNEER YİNELEME BAĞINTISI

q bir asal ya da bir p asalının kuvveti ve $k \geq 2$ pozitif bir tamsayı olmak üzere $\mathbb{F}_{q^k}$ mertebesi q^k olan bir sonlu cisim tanımlar. Bu alt bölümde, $\mathbb{F}_{q^k}$ daki siklotomik alt grupların elemanlarının sıkıştırılmış formları olarak adlandırılan ve daha az bit ile temsil edilen bir yöntemi açıklayacağız. Daha fazla detay ve ispat için [19,24,10] kaynaklarına bakılabilir. $\mathbb{F}_{q^k}$ da bir $G_{r,q,k}$ siklotomik alt grubu, $r \nmid k$ ve $r \mid \Phi_k(q)$ olacak şekilde r asal mertebeli bir alt grup olarak tanımlanır. Şimdi, bir $G_{r,q,k}$ siklotomik alt grup elemanlarının minimal polinom katsayıları ile minimal polinoma karşılık gelen lineer yineleme bağıntısı arasındaki ilişkiyi göstereceğiz.

$k \geq 2$ olmak üzere, α, $G_{r,q,k}$ siklotomik alt grubun bir elemanı olsun. $d \mid k$ ve $n = k/d > 1$ olmak üzere $\mathbb{F}_{q^d}$ üzerinde

$$f_\alpha(x) = x^n - a_1 x^{n-1} + \cdots + (-1)^{n-1} a_{n-1} x + (-1)^n a_n$$

polinomu, α nın minimal polinomu olsun. Bu durumda, $a_n = 1$ olur. Diğer taraftan, $f_\alpha(x)$ in tüm kökleri $\alpha, \alpha^q, \ldots, \alpha^{q^{n-1}}$ olur ve buradan $1 \leq i \leq n$ için a_i ler

$$a_1 = \sum_{i=0}^{n-1} \alpha^{q^i},\ a_2 = \sum_{i<j} \alpha^{q^i + q^j}, \ldots,\ a_n = \prod_{i=0}^{n-1} \alpha^{q^i} = 1$$

şeklindeki simetrik fonksiyonlara karşılık gelir.

$f_\alpha(x)$ polinomu,

$$s_t = a_1 s_{t-1} - a_2 s_{t-2} + \cdots - (-1)^n s_{t-n},\quad t \geq n$$

şeklinde Doğrusal Geri Beslemeli Kaydırmalı Yazdırgaç (Linear Feedback Shift Register) dizisi olarak bilinen n. dereceden bir lineer yineleme $\{s_i\} = \underline{s}$ dizisi tanımlar. $\underline{s}$ dizisinin başlangıç değerleri, $Tr: \mathbb{F}_{q^n} \to \mathbb{F}_q$ iz fonksiyonu olmak üzere $i = 0, 1, \cdots, n-1$ için

$$s_i = Tr(\alpha^i) = \alpha^i + \alpha^{iq} + \ldots + \alpha^{iq^{n-1}} \quad (1)$$

eşitliği ile elde edilir. $\underline{s}$ dizisi $\mathbb{F}_{q^d}$ üzerinde α tarafından üretilen n. dereceden karakteristik dizi olarak da adlandırılır.

per(f_α), f_α nın periyodu olmak üzere per(f_α) =per $\left(\underline{s}\right)$ değerinin, $\alpha \in \mathbb{F}_{q^n}$ nın mertebesine eşit olduğunu bir not olarak belirtmeliyiz [31, Chapter 8]. f_α nın sabit terimi,

$$\prod_{i=0}^{n-1} \alpha^{q^i} = \alpha^{\frac{q^n-1}{q-1}} = \alpha^{1+q+\cdots+q^{n-1}} = 1$$

olduğundan, α nın mertebesi Q, $1+q+\cdots+q^{n-1}$ yi bölmelidir.

(1) eşitliğinden, bir m tamsayısı için α^m nin minimal polinomu, $i=0,1,\cdots,n-1$ için kökleri α^{mq^i} olan

$$f_{\alpha^m}(x) = x^n - a_{1,m}x^{n-1} + a_{2,m}x^{n-2} - \cdots + (-1)^{n-1}a_{n-1,m}x + (-1)^n$$
$$= \prod_{i=0}^{k-1} (x - \alpha^{mq^i})$$

şeklinde olur. Buradan, α^m yi (ve eşleniklerini) $\{a_{1,m}, a_{2,m}, \cdots, a_{n-1,m}\}$ kümesi ile temsil edebileceğimiz anlaşılır.

Burada, $f_{\alpha^m}(x)$ in katsayıları, $1 \le i \le n-1$ için

$$a_{i,m} = \sum_{0 \le j_1 \le \cdots \le j_i \le n-1} \alpha^{m(q^{j_1} + q^{j_2} + \cdots + q^{j_i})} \quad (2)$$

şeklinde yazılabilir. gcd($per(f_\alpha), n$) $=1$ ise f_α ve $f_{\alpha^m}(x)$ polinomlarının periyotları aynıdır; bir başka deyişle $f_{\alpha^m}(x)$ polinomu $\mathbb{F}_q$ üzerinde indirgenemezdir. Denklem (2) den $a_{i,m} = a_{n-i,-m}$ eşitliği elde edilir. Newton Formülünden [31], aşağıdaki eşitlikleri kullanarak herhangi bir $i \in \{1,\cdots,n-1\}$ için $\{s_m, s_{2m}, \cdots, s_{im}\}$ den etkili bir şekilde $\{a_{1,m}, a_{2,m}, \cdots, a_{i,m}\}$ elde edilebilir:

$$s_{im} = a_{1,m} s_{(i-1)m} - a_{2,m} s_{(i-2)m} + \cdots - (-1)^i i a_{i,m}$$
$$a_{i,m} = i^{-1}((-1)^{i+1} s_{im} + \cdots + a_{i-1,m} s_m).$$

Bu bölümdeki amacımız α^m yi daha az sayıda elemanla temsil edebilmek olduğundan, aşağıdaki gerçekler bu hususta oldukça önem arz edecektir.

1. $k = 2l$ çift ise, $\alpha \in \mathbb{F}_{q^{2l}}$ nin mertebesi $q^l + 1$ i böler. Bu da, $\alpha^{mq^l} = \alpha^{-m}$ olduğu anlamına gelir. Bundan dolayı, $i = 1, \cdots, n-1$ için

$$\begin{aligned} a_{n-i,m} &= a_{i,-m} \\ &= \sum_{0 \le j_1 \le \cdots \le j_i \le n-1} \alpha^{-m(q^{j_1} + q^{j_2} + \cdots + q^{j_i})} \\ &= \sum_{0 \le j_1 \le \cdots \le j_i \le n-1} \alpha^{mq^l(q^{j_1} + q^{j_2} + \cdots + q^{j_i})} \\ &= a_{i,m}^{q^l} \end{aligned}$$

eşitliğini elde ederiz. Dolayısıyla, α^m yi (ve eşleniklerini) $\{a_{1,m}, \cdots, a_{(n-1)/2,m}\}$ kümesi ile temsil edebiliriz.

2. $d \mid l$ iken $k = 2l$ ise, $i = 1, \cdots, n-1$ için önceki sonuçtan $a_{n-i,m} = a_{i,m}^{q^l}$ eşitliğini elde ederiz. $d \mid l$ yani $n = k/d$ çift olduğundan $a_{i,m}^{q^l} = a_{i,m}$ eşitliğini elde edeceğimizden α^m yi (ve eşleniklerini) $\{a_{1,m}, \cdots, a_{n/2,m}\}$ kümesi ile temsil edebiliriz.

Yardımcı Teorem 1 [10]: $k = de$, $e > 1$ olsun. $G_{r,q,k}$ siklotomik alt grubunun herhangi bir α elemanı ve bir m tamsayısı için α^m , $\mathbb{F}_{q^d}$ de $\#R_{\alpha^m}$ sayıda eleman ile temsil edilebilir:

$$\#R_{\alpha^m} = \begin{cases} e-1, & de\ tek \\ \dfrac{e-1}{2}, & d\ çift\ ve\ e\ tek \\ \dfrac{e}{2}, & e\ çift \end{cases}$$

Not 1: $k \geq 2$ olmak üzere, α, $G_{r,q,k}$ siklotomik alt grubun bir elemanı olsun. $d \mid k$ ve $n = k/d > 1$ olmak üzere $\mathbb{F}_{q^d}$ üzerinde tanımlı $f_\alpha(x) = x^n - a_1 x^{n-1} + \cdots + (-1)^{n-1} a_{n-1} x + (-1)^n$ polinomunu kullanarak α^m yi (ve eşleniklerini) temsil edeceğimiz $\{a_{1,m}, a_{2,m}, \cdots, a_{n-1,m}\}$ kümesindeki dizilerin $m.$ terimini veren literatürde etkin algoritmalar mevcuttur. Örneğin, [44] de $n=2$ için $a_{1,m}$ dizisinin $m.$ terimini hesaplamak için, [23,24] da $n=3$ için $a_{1,m}$ ve $a_{1,-m}$ ikilisinin $m.$ terimini hesaplamak için, [30,42] da $n=3$ ve $d=2$ için $a_{1,m}$ dizisinin $m.$ terimini hesaplamak için, [14] da $n=5$ için $m.$ terim olan $a_{1,m}, a_{1,-m}, a_{2,m}$ ve $a_{2,-m}$ dizilerini hesaplamak için, [18,20] da $n=5$ ve $d=2$ için $a_{1,m}$ ve $a_{2,m}$ dizisinin $m.$ terimlerini hesaplamak için etkili algoritmalar bulunmaktadır.

Yardımcı Teorem 2 (Değişmelilik Yasası) [29]:

$f_\alpha(x) = x^n - a_1 x^{n-1} + \cdots + (-1)^{n-1} a_{n-1} x + (-1)^n$, $\mathbb{F}_{q^d}$ üzerinde tanımlı bir polinom ve $\underline{s} = \{s_i\}$ de onun karakteristik dizisi olsun. O halde, tüm $e, r \in \mathbb{Z}$ için,

$$a_{1,e}(f_{\alpha^r}) = a_{1,er}(f_\alpha) = a_{1,r}(f_{\alpha^e})$$

İspat:

$$f_{\alpha^r}(x) = \prod_{i=0}^{n-1} (x - \alpha^{rq^i})$$
$$= x^n - a_{1,r} x^{n-1} + \cdots + (-1)^{n-1} a_{n-1,r} x + (-1)^n$$

Bu nedenle,

$$a_{1,e}(f_{\alpha^r}) = \sum_{i=0}^{n-1} (\alpha^{rq^i})^e = \sum_{i=0}^{n-1} (\alpha^{q^i})^{er} = a_{1,er}(f_\alpha)$$

elde edilir.

7.3. ELİPTİK EĞRİ EŞLEME TABANLI KRİPTOGRAFİ

G_1, G_2ve G_3, n mertebeli devirli gruplar olsun. Bu durumda, iki lineer eşleme,

$$e: G_1 \times G_2 \to G_3$$
$$(P, Q) \mapsto e(P, Q)$$

şeklinde tanımlanıp aşağıdaki özelliklere sahip olan bir dönüşümdür.

- (İki Lineerlik) Her $P \in G_1$, $Q \in G_2$ ve $a, b \in \mathbb{Z}$ için $e(aP, bQ) = e(P, Q)^{ab}$ dir.
- (Dejenere Olmama) Her $P \in G_1$, $P \neq Id_{G_1}$ için $e(P, Q) \neq Id_{G_3}$ değerini sağlayacak bir $Q \in G_2$ vardır. Her $Q \in G_2$, $Q \neq Id_{G_2}$ için $e(P, Q) \neq Id_{G_3}$ değerini sağlayacak bir $P \in G_1$ vardır.

İki lineer eşleme özelliklerini kullanan birçok eşleme tabanlı protokol vardır. Biz aşağıda sadece en önemli protokollerden biri olan Boneh, Lynn ve Shacham (BLS) tarafından tanımlanan ve kısa imzalama şeması olarak adlandırılan algoritmayı vereceğiz [9]: $P \in G_1$ mertebesi n olan bir nokta olmak üzere $G_1 = <P>$ olsun.

- **Anahtar Oluşturma:** Rastgele bir $c \in \mathbb{Z}_n^*$ seçerek cP değerini hesaplayın. c gizli anahtar, cP ise açık anahtardır.
- **İmza:** $H: \{0,1\}^* \to G_1^*$ bir kriptografik özet fonksiyonu olsun. c gizli anahtarı ve $m \in \{0,1\}^*$ mesajı verildiğinde imzayı $S = cH(m) \in G_1$ hesaplayın.
- **Doğrulama:** cP açık anahtarı, m mesajı ve S imzası verildiğinde $e(cP, H(m)) = e(P, S)$ olduğunu doğrulayın.

Sonlu cisimler üzerinde tanımlı eliptik eğrilerdeki Weil ve Tate eşlemeleri, eşleme tabanlı kriptografik protokollerde kullanılan iki lineer eşlemelerin en klasik örneklerindendir. Daha sonradan, Tate eşlemesinin türevleri de bu protokollerde kullanılmaya başlamıştır.

E, karakteristiği p olan $\mathbb{F}_q$ sonlu cismi üzerinde bir eliptik eğri olsun. r, mertebesi $n = \#E(\mathbb{F}_q)$ olan eliptik eğri grubunun en büyük asal böleni ve $\gcd(r, q) = 1$ olsun. E nin gömme derecesi k, $r | q^k - 1$ olacak şekilde tanımlanan en küçük tamsayıdır. $P \in E(\mathbb{F}_{q^k})$ ve ∞, E nin birim elemanı olmak üzere $rP = \infty$ olsun. O halde, P nin mertebesi r olur ve P noktası, r-burulma (torsion) noktası olarak isimlendirilir. $E(\mathbb{F}_{q^k})$ daki r-burulma noktalarının grubu $E(\mathbb{F}_{q^k})[r]$ ile gösterilir. Ayrıca, $rE(\mathbb{F}_{q^k}) = \{rP \mid P \in E(\mathbb{F}_{q^k})\}$da $E(\mathbb{F}_{q^k})$ nin bir alt grubudur ve $E(\mathbb{F}_{q^k}) / rE(\mathbb{F}_{q^k})$ bölüm grubu mertebesi r olan bir gruptur. $Q \in E(\mathbb{F}_{q^k}) / rE(\mathbb{F}_{q^k})$ olsun. E eliptik eğrisi üzerinde bir divisor D, $n_p \in \mathbb{Z}$ ve sonlu sayıda n_p sıfır olmak üzere $D = \sum_{P \in E(\mathbb{F}_{q^k})} n_p(P)$ toplamı şeklinde tanımlanır. Şimdi, $R \in E(\mathbb{F}_{q^k})$ rastgele bir nokta olmak üzere $D = (Q + R) - (R)$ divisor değerini dikkate alalım. Burada, D ile $(P) - (\infty)$ aralarında asal olmalıdır. Yukarıdaki bilgilerin ışığında Tate eşlemesi aşağıdaki gibi tanımlanır:

$$\langle ., . \rangle_r : E(\mathbb{F}_{q^k})[r] \times E(\mathbb{F}_{q^k}) \Big/ rE(\mathbb{F}_{q^k}) \mapsto \mathbb{F}^*_{q^k} \Big/ \left(\mathbb{F}^*_{q^k}\right)^r$$
$$(P, Q) \mapsto f_{r,P}(D)$$

$\mathbb{F}^*_{q^k} / (\mathbb{F}^*_{q^k})^r$ bölüm grubunun, $a \equiv b$ ancak ve ancak $\exists\ c \in \mathbb{F}^*_{q^k}$ öyle ki $a = bc^r$ denklik bağıntısının meydana getirdiği $\mathbb{F}^*_{q^k}$ nın denklik sınıflarının kümesi olarak düşünülebileceği açıktır. Bu durum Tate eşlemesinin, $\mathbb{F}^*_{q^k}$ daki bir değerin r. kuvvetinin bir çarpanı olarak tanımlandığı anlamına gelir. Kriptografideki birçok uygulamada tek bir değer elde etmek gerekir. Bu nedenle, indirgenmiş Tate eşlemesi şu şekilde tanımlanır:

$$e_r(P,Q) = f_{r,P}(D)^{(q^k-1)/r}$$

Burada, $f_{r,P}(D)$ çıktısının $(q^k-1)/r$ kuvvetinin alınması son üs alma işlemi olarak bilinir. Bu işlem, aynı zamanda Tate eşlemesinin türevleri için de kullanılır. Tate eşlemesi, Miller algoritması kullanılarak etkili bir şekilde hesaplanabilir [33]. Daha detaylı bilgi için [6] kaynağındaki 9. bölüme bakılabilir.

7.4. SON ÜS ALMA

İndirgenmiş Tate eşlemesinin (ve türevlerinin) $e_r(P,Q) = f_{r,P}(D)^{(q^k-1)/r}$ ihtiyaç duyduğu üs alma $k=2,4,6$ gömme dereceli süpersingüler eliptik eğriler için hesaplanmıştır [4]. Daha sonra bu hesaplama, aşağıda açıklandığı şekilde Lynn tarafından yapılmıştır [32]. Tate eşlemesinin Miller algoritmasıyla elde edilen değeri a olsun. a değerinin $\frac{q^k-1}{r}$ kuvvetini hesaplamak için ilk olarak

$$q^k - 1 = \prod_{d|k} \Phi_d(q)$$

eşitliğinden faydalanılır. k gömme derecesi olduğu için r nin siklotomik polinom $\Phi_k(q)$ yu bölmesi gerekir. $c = \prod_{d|k,\, d<k} \Phi_d(q)$ olmak üzere $b = a^c$ değeri hesaplanır. $\Phi_k(q)/r$ bir tamsayı olduğu için, standart üs alma algoritması kullanılarak $b^{\Phi_k(q)/r}$ sonucu elde edilir. Bu yöntemin, [4] deki süpersingüler eliptik eğriler için verilen yaklaşımdan daha hızlı olduğunu belirtmek gerekir.

Örnek 1: $\mathbb{F}_{q^2} = \mathbb{F}(\alpha) \cong \mathbb{F}_q[x]/<x^2-\delta>$ olsun. O halde, $a \in \mathbb{F}_{q^2}$ elemanı $u, v \in \mathbb{F}_q$ ve $\alpha^2 = \delta$ olmak üzere $a = u + \alpha v$ şeklinde yazılabilir. $q^2 - 1 = \Phi_2(q)\,\Phi_1(q)$ ve $r \mid \Phi_2(q) = q+1$ olduğu açıktır. Dolayısıyla,

$$b^{(q+1)/r} = (a^{q-1})^{(q+1)/r} = \left(\frac{u-\alpha v}{u+\alpha v}\right)^{(q+1)/r}$$

elde edilir.

$$b=\frac{u-\alpha v}{u+\alpha v}=\frac{u^2-v^2}{u^2+v^2}-\alpha\frac{2uv}{u^2+v^2}$$

olduğundan dolayı cisim elemanı b üniter (unitary) olur [26,38]. Bir başka deyişle, $\bar{b}$, b nin $\mathbb{F}_{q^2}$ deki eşleniği olmak üzere $b\bar{b}=1$ olur. Daha sonra, $a^{(q^2-1)/r}$ değerini elde etmek için standart üs alma algoritması kullanılarak $b^{(q+1)/r}$ değeri hesaplanır. Sonuç olarak, bu yaklaşım kullanılarak son üs alma işleminin boyutu etkin bir şekilde yarıya indirilmiş olur.

Not 2: $k=2d$ için, son üs alma

$$\frac{q^k-1}{r}=(q^d-1)\frac{(q^d+1)}{\Phi_d(q)}\frac{\Phi_d(q)}{r}$$

şeklinde yazılabilir. (q^d-1) . kuvveti alındıktan sonra, cisim elemanı üniter hale gelir. Bu özellik aşağıda belirtilen iki önemli sonuca olanak sağlar:

i. Üniter elemanların karesini almak, üniter olmayan elemanların karesini almaktan önemli ölçüde daha az maliyetlidir.

ii. Üniter elemanlar için ileride gerçekleşebilecek herhangi bir ters alma işlemi basit bir eşleme ile uygulanabilir.

7.5. VERİMLİ SIKIŞTIRMA ALGORİTMALARI

7.5.1. Sıkıştırma Faktörünün 2 Olması

$\alpha\in\mathbb{F}_{q^2}^*, G_{r,q,2}$ nin bir elemanı ve

$$f_\alpha(x)=x^2-a_1x+1$$

polinomu da α nın $\mathbb{F}_q$ üzerindeki minimal polinomu olsun. $f_\alpha(x)$ polinomu,

$$s_{t+2}=a_1s_{t+1}-s_t$$

şeklindeki $\{s_i\}$ ikinci dereceden lineer yineleme bağıntısını tanımlar. Bir m tamsayısı için, α^m nin $\mathbb{F}_q$ üzerindeki minimal polinomu, kökleri α^m ve α^{mq} olan ve $a_{1,m} = Tr(\alpha^m) = \alpha^m + \alpha^{mq}$ eşitliğini sağlayan

$$f_{\alpha^m}(x) = x^2 - a_{1,m}x + 1$$

şeklinde olur. Lucas dizisi olarak da bilinen $\{a_{1,m}\}$ dizisi

$$a_{1,0} = 2,\ a_{1,1} = a_1,\ a_{1,u+1} = a_1 a_{1,u} - a_{1,u-1}$$

şeklindeki lineer yineleme bağıntıları ile tanımlanır.

Smith ve Skinner [41], $\mathbb{F}_{q^2}^*$ içerisindeki $G_{r,q,2}$ nin elemanlarının $\{a_{1,m}\}$ dizisi ile olağan durumun $1/2$ oranı ile belirlenebileceğini göstermişlerdir. Daha açık bir ifadeyle, $G_{r,q,2}$ nin elemanları, o elemanların $\mathbb{F}_q$ üzerindeki izleri ile tek türlü ifade edilebilir. Bu yapı, sıkıştırma faktörü 2 olan duruma karşılık gelir.

$\{a_{1,m}\}$ Lucas dizisi, her $u, v \in \mathbb{Z}$ için

$$a_{1,u+v} = a_{1,u}a_{1,v} - a_{1,u-v}$$
$$a_{1,2u} = a_{1,u}^2 - 2$$

özellikleri kullanılarak Algoritma 7.1 ile etkili bir şekilde hesaplanabilir.

Yen ve Laih [44], aynı zamanda Smith ve Lennon [40] çalışmasındaki 3.8 ve 3.9 nolu denklemlerden de elde edilebilen yukarıda verilen Lucas dizisinin özelliklerini kullanarak yeni algoritma geliştirmişlerdir. Algoritma 1, soldan sağa tarama yaklaşımına dayanmaktadır, bu çalışmada verilmeyen diğer algoritma [40] ise sağdan sola tarama yaklaşımına dayanmaktadır. $\lfloor\ \rfloor$ en büyük tamsayı fonksiyonu olmak üzere, her iki algoritmanın da hesaplama maliyeti $\mathbb{F}_q$ üzerinde $2 + 2\lfloor \log_2 m \rfloor$ çarpmadır. Ancak, sağdan sola tarama algoritmasında daha fazla geçici belleğe ihtiyaç duyulmaktadır.

Algoritma 7.1 Lucas Dizisini Hesaplama

Girdi: $a_1 \in \mathbb{F}_q$ ve $m_1 = 1$ olmak üzere $m = \sum_{j=0}^{t} m_j 2^j \in \mathbb{Z}^+$

Çıktı: $\left(a_{1,m}, a_{1,m+1}\right)$

1: $\left(a_{1,y}, a_{1,y+1}\right) \leftarrow \left(2, a_1\right)$

2: **for** $j \leftarrow t$ **to** 0 **do**

3: **if** $m_j = 1$ **then**

4: $a_{1,y} \leftarrow a_{1,y} a_{1,y+1} - a_1,\ a_{1,y+1} \leftarrow a_{1,y+1}^2 - 2$

5: **else**

6: $a_{1,y} \leftarrow a_{1,y}^2 - 2,\ a_{1,y+1} \leftarrow a_{1,y} a_{1,y+1} - a_1$

7: **end if**

8: **end for**

9: **return** $(a_{1,y}, a_{1,y+1})$

7.5.2. Sıkıştırma Faktörünün 3/2 Olması

$\alpha \in \mathbb{F}_{q^3}^*, G_{r,q,3}$ ün bir elemanı ve

$$f_\alpha(x) = x^3 - a_1 x^2 + a_2 x - 1$$

polinomu da α nın $\mathbb{F}_q$ üzerindeki minimal polinomu olsun. $f_\alpha(x)$ polinomu

$$s_{t+3} = a_1 s_{t+2} - a_2 s_{t+1} + s_t$$

şeklindeki $\{s_i\}$ üçüncü dereceden lineer yineleme bağıntısını tanımlar. Bir m tamsayısı için, α^m nin $\mathbb{F}_q$ üzerindeki minimal polinomu, kökleri α^m, α^{mq} ve α^{mq^2} olan ve $a_{1,m} = Tr(\alpha^m) = \alpha^m + \alpha^{mq} + \alpha^{mq^2}$ eşitliğini sağlayan

$$f_{\alpha^m}(x) = x^3 - a_{1,m}x^2 + a_{1,-m}x - 1$$

şeklinde olur.

Gong ve Harn [23], $\mathbb{F}_{q^3}^*$ içerisindeki $G_{r,q,3}$ ün elemanlarının 3/2 sıkıştırma faktörüyle $\{a_{1,m}, a_{1,-m}\}$ dizileri ile belirlenebileceğini göstermişlerdir. Ayrıca, bu elemanların sıkıştırılmış formları için etkili bir üs alma algoritmasını her $u, v \in \mathbb{Z}$ için

$$a_{1,u+v} = a_{1,u}a_{1,v} - a_{1,u-v}a_{1,-v} + a_{1,u-2v}$$
$$a_{1,2u} = a_{1,u}^2 - 2a_{1,-u}$$

eşitliklerini kullanarak Algoritma 7.2 de elde etmişlerdir.

Algoritma 7.2'de, $\{a_{1,-m}\}$ yi hesaplamak için T_i ve T_{i-1}, sırasıyla $-T_i$ ve $-T_{i-1}$ ile değiştirilmelidir. Algoritma 7.2'yi kullanarak $a_{1,m}$ ve $a_{1,-m}$ dizi çiftinin m. terimini $\mathbb{F}_q$ da hesaplamak için ortalama olarak $9\log_2 m$ çarpmaya ihtiyaç duyulur. Bu yöntem, polinom modülünü kullanan Fiduccia nın [15] algoritmasına göre daha etkilidir. Daha sonra, Gong, Harn ve Wu işaretli basamak (signed-digit) gösterimini kullanarak çok daha etkili bir algoritma önermişlerdir [24]. Bu gösterimi kullanarak $a_{1,m}$ ve $a_{1,-m}$ dizi çiftinin m. terimini $\mathbb{F}_q$ da hesaplamak için ortalama olarak $4\log_2 m$ çarpma işlemine ihtiyaç olduğunu göstermişlerdir. Bu yöntem, q nun bir asal veya bir p asalının kuvveti olduğu durumlarda Gong ve arkadaşları tarafından optimize edilmiştir [25].

Algoritma 7.2 $\{a_{1,m}\}$ ve $\{a_{1,-m}\}$ Dizilerini Hesaplama

Girdi: $a_1, a_2 \in \mathbb{F}_q$, $1 \le i \le t$ için $T_0 = m_0 = 1$, $T_i = m_i + 2T_{i-1}$ olmak üzere $m = \sum_{j=0}^{t} m_j 2^{t-j} \in \mathbb{Z}^+$ ve bu yüzden $T_t = m$

Çıktı: $(a_{1,m-1}, a_{1,m}, a_{1,m+1})$

1: $(a_{1,T_i-1}, a_{1,T_i}, a_{1,T_i+1}) \leftarrow (3, a_1, a_1^2 - 2a_2)$

2: **for** $j \leftarrow 0$ **to** t **do**

3: **if** $m_j = 1$ **then**

4: $a_{1,T_i-1} \leftarrow a_{1,T_{i-1}}^2 - 2a_{1,-T_{i-1}}$

5: $a_{1,T_i} \leftarrow a_{1,T_{i-1}} a_{1,T_{i-1}+1} - a_1 a_{1,-T_{i-1}} + a_{1,-(T_{i-1}-1)}$

6: $a_{1,T_i+1} \leftarrow a_{1,T_{i-1}+1}^2 - 2a_{1,-(T_{i-1}+1)}$

7: **else**

8: $a_{1,T_i-1} \leftarrow a_{1,T_{i-1}} a_{1,T_{i-1}-1} - a_2 a_{1,-T_{i-1}} + a_{1,-(T_{i-1}+1)}$

9: $a_{1,T_i} \leftarrow a_{1,T_{i-1}}^2 - 2a_{1,-T_{i-1}}$

10: $a_{1,T_i+1} \leftarrow a_{1,T_{i-1}} a_{1,T_{i-1}+1} - a_1 a_{1,-T_{i-1}} + a_{1,-T_{i-1}-1}$

11: **end if**

12: **end for**

13: **return** $(a_{1,T_i-1}, a_{1,T_i}, a_{1,T_i+1})$

7.5.3 Sıkıştırma Faktörünün 3 Olması

$\alpha \in \mathbb{F}_{q^6}^*, G_{r,q,6}$ nın bir elemanı olsun. Bu durumda, $\alpha \in \mathbb{F}_{q^6}^*$ nın $\mathbb{F}_{q^2}$ deki eşlenikleri, α, α^{q^2} ve α^{q^4} olur. Buradan, $a_1 = Tr_{\mathbb{F}_{q^6}/\mathbb{F}_{q^2}}(\alpha) = \alpha + \alpha^{q^2} + \alpha^{q^4}$ eşitliği elde edilir. $q^2 \equiv q - 1 (\bmod\, q^2 - q + 1)$ ve $q^4 \equiv -q (\bmod\, q^2 - q + 1)$ oldu-

ğu için $\alpha \in G_{r,q,6}$ nın eşlenikleri aynı zamanda α, α^{q-1} ve α^{-q} olur. Bu da 2. elementer simetrik fonksiyonun $a_2 = \alpha\alpha^{q^2} + \alpha\alpha^{q^4} + \alpha^{q^2}\alpha^{q^4} = a_1^q$ olduğu anlamına gelir. Dolayısıyla, α nın $\mathbb{F}_{q^2}$ üzerindeki minimal polinomu

$$f_\alpha(x) = x^3 - a_1 x^2 + a_1^q x - 1$$

şeklinde olur. $f_\alpha(x)$ polinomu $\mathbb{F}_{q^2}$ üzerinde

$$s_{t+3} = a_1 s_{t+2} - a_1^q s_{t+1} + s_t$$

şeklindeki $\{s_i\}$ üçüncü dereceden lineer yineleme bağıntısını tanımlar. Bir m tamsayısı için, α^m nin $\mathbb{F}_{q^2}$ üzerindeki minimal polinomu, kökleri α^m, α^{mq^2} ve α^{mq^4} olan ve

$$a_{1,m} = Tr_{\mathbb{F}_{q^6}/\mathbb{F}_{q^2}}(\alpha) = \alpha^m + \alpha^{m(q-1)} + \alpha^{-mq}$$

eşitliğini sağlayan

$$f_{\alpha^m}(x) = x^3 - a_{1,m} x^2 + a_{1,m}^q x - 1$$

şeklinde olur. Bu sebeple, f_{α^m} polinomu $\{a_{1,m}\}$ dizisi ile tamamıyla belirlenmiş olur.

Lenstra ve Verheul [30], yukarıdaki prosedürü kullanarak XTR (Etkili ve Kompakt Alt Grup İz Temsili) şifreleme sistemini tanımlamışlar ve $\mathbb{F}_{q^6}^*$ içerisindeki $G_{r,q,6}$ nın elemanlarının 3 sıkıştırma faktörüyle $\mathbb{F}_{q^2}$ üzerinde $\{a_{1,m}\}$ dizisi ile tam olarak belirlenebileceğini göstermişlerdir. Ayrıca, bu elemanların sıkıştırılmış formu $\{a_{1,m}\}$ için XTR üs alma algoritmasını her $u, v \in \mathbb{Z}$ için

$$a_{1,u+v} = a_{1,u}a_{1,v} - a_{1,v}^q a_{1,u-v} + a_{1,u-2v}$$
$$a_{1,2u} = a_{1,u}^2 - 2a_{1,u}^q$$

eşitliklerini kullanarak Algoritma 7.3 ([30], Algoritma 2.3.7) de elde etmişlerdir. Dahası, XTR üs alma algoritmasında $a_{1,m}$ dizisinin m. terimini $\mathbb{F}_q$ da hesaplamak için ortalama olarak $8\log_2 m$ çarpma işlemine ihtiyaç duyulduğunu da göstermişlerdir.

Algoritma 7.3 XTR Dizisini Hesaplama

Girdi: $a_1 \in \mathbb{F}_{q^2}$ ve $m_t = 1$ olmak üzere $m = \sum_{j=0}^{t} m_j 2^j \in \mathbb{Z}^+$

Çıktı: $(a_{1,2m}, a_{1,2m+1}, a_{1,2m+2})$

1: $(a_{1,y-1}, a_{1,y}, a_{1,y+1}) \leftarrow (3, a_1, a_1^2 - 2a_1^q)$

2: **for** $j \leftarrow t$ **to** 0 **do**

3: **if** $m_j = 1$ **then**

4: $a_{1,y-1} \leftarrow a_{1,y}^2 - 2a_{1,y}^q$

5: $a_{1,y} \leftarrow a_{1,y+1} a_{1,y} - a_{1,y}^q a_1 + a_{1,y-1}^q$

6: $a_{1,y+1} \leftarrow a_{1,y+1}^2 - 2a_{1,y+1}^q$

7: **else**

8: $a_{1,y-1} \leftarrow a_{1,y-1}^2 - 2a_{1,y-1}^q$

9: $a_{1,y} \leftarrow a_{1,y-1} a_{1,y} - a_{1,y}^q a_1^q + a_{1,y+1}^q$

10: $a_{1,y+1} \leftarrow a_{1,y}^2 - 2a_{1,y}^q$

11: **end if**

12: **end for**

13: **return** $(a_{1,y-1}, a_{1,y}, a_{1,y+1})$

7.5.4 Sıkıştırma Faktörünün 5/2 Olması

$\alpha \in \mathbb{F}_{q^{10}}^*, G_{r,q,10}$ nın bir elemanı ve

$$f_\alpha(x) = x^5 - a_1 x^4 + a_2 x^3 - a_2^p x^2 + a_1^p x - 1$$

polinomu da α nın $\mathbb{F}_{q^2}$ üzerindeki minimal polinomu olsun. $f_\alpha(x)$ polinomu

$$s_{t+5} = a_1 s_{t+4} - a_2 s_{t+3} + a_2^p s_{t+2} - a_1^p s_{t+1} + s_t$$

şeklindeki $\{s_i\}$ beşinci dereceden lineer yineleme bağıntısını tanımlar. Bir m tamsayısı için, α^m nin $\mathbb{F}_{q^2}$ üzerindeki minimal polinomu,

$$f_{\alpha^m}(x) = x^5 - a_{1,m} x^4 + a_{2,m} x^3 - a_{2,m}^p x^2 + a_{1,m}^p x - 1$$

şeklinde olur. Burada, başlangıç koşulları $a_{1,-1} = a_1^p$, $a_{1,0} = 5$, $a_{1,1} = a_1$, $a_{1,2} = a_1^2 - 2a_2$, $a_{1,3} = a_1^3 - 3a_1 a_2 + 3a_2^p$ olmak üzere

$$a_{1,m} = \mathrm{Tr}_{\mathbb{F}_{q^{10}}/\mathbb{F}_{q^2}}(\alpha^m) = \sum_{i=0}^{4} \alpha^{mq^{2i}}$$

ve

$$\begin{aligned} a_{2,m} &= \mathrm{Tr}_{\mathbb{F}_{q^{10}}/\mathbb{F}_{q^2}}(\alpha^{m(q^2+1)} + \alpha^{m(q^4+1)}) \\ &= \sum_{0 \le i < j \le 4} \alpha^{mq^{2i} + mq^{2j}} \end{aligned}$$

olarak elde edilir. Bu sebeple, f_{α^m} polinomu $\{a_{1,m}, a_{2,m}\}$ kümesi ile tamamıyla belirlenmiş olur.

Giuliani ve Gong [18], yukarıdaki prosedürü kullanarak $\mathbb{F}_{q^{10}}^*$ içerisindeki $G_{r,q,10}$ nın elemanlarının 5/2 sıkıştırma faktörüyle $\mathbb{F}_{q^2}$ üzerinde $\{a_{1,m}, a_{2,m}\}$ kümesi ile tam olarak belirlenebileceğini göstermişlerdir. Ayrıca, bu elemanların sıkıştırılmış formlarını $\{a_{1,m}, a_{2,m}\}$ hesaplamak için her $u, v \in \mathbb{Z}$ için

$$a_{1,2u} = a_{1,u}^2 - 2a_{2,u}$$

$$a_{2,2u} = a_{2,u}^2 + 2a_{1,u}^p - 2a_{1,u}a_{2,u}^p$$

$$a_{1,3u} = a_{1,u}^3 - 3a_{1,u}a_{2,u} + 3a_{2,u}^p$$

$$a_{2,3u} = a_{2,u}^3 - 3a_{1,u}^p a_{2,u} - 3a_{1,u}a_{2,u}a_{2,u}^p + 3a_{1,u}^2 a_{1,u}^p + 3a_{2,u}^{2p} - 3a_{1,u}$$

$$a_{1,u+v} = a_{1,u}a_{1,v} - a_{1,u-v}a_{2,v} + a_{1,u-2v}a_{2,v}^p - a_{1,u-3v}a_{1,v}^p + a_{1,u-4v}$$

$$3a_{2,u+v} = a_{1,u}^p a_{2,u-v} - a_{2,u}a_{2,v} + a_{1,u}a_{1,v}a_{1,u+v} - a_{1,u-2v}a_{1,u-v} + a_{1,2u-3v} - a_{1,u+2v}a_{1,u} - a_{1,2u+v}a_{1,v} + a_{1,u+v}^2$$

eşitliklerini kullanarak Algoritma 7.4 ü elde etmişlerdir. Dahası, Algoritma 7.4 te $a_{1,m}$ ve $a_{2,m}$ dizilerinin m. terimini $\mathbb{F}_q$ da hesaplamak için ortalama olarak $108.5\log_2 m$ çarpma işlemine ihtiyaç olduğunu göstermişlerdir. Daha sonra, Quoos ve Mjølsnes [36] bu dizileri hesaplamak için bir algoritma verip, bu iki diziyi hesaplama maliyetinin, $\mathbb{F}_q$ da $102\log_2 m$ çarpma işlemi olduğunu göstermişlerdir. 2006 yılında ise, Giuliani ve Gong [20], $\{a_{1,m}\}$ ve $\{a_{2,m}\}$ dizilerinin m. terimini hesaplayan ve Çapraz Çift Ekleme (Diagonal Double-Add) adı verilen yeni bir algoritma önermişlerdir. Bu algoritma, $\mathbb{F}_q$ üzerinde $74\log_2 m$ çarpma işlemi gerektirdiğinden daha önce önerilen algoritmalar içerisinde en etkili olanı olarak dikkati çekmektedir.

Algoritma 7.4 $\{a_{1,m}\}$ ve $\{a_{2,m}\}$ Dizilerini Hesaplama

---

Girdi: $a_1, a_2 \in \mathbb{F}_{q^2}$ ve $c_j = \{-1, 0, 1\}$ olmak üzere $m = \sum_{j=0}^{n} c_j 3^j$

Çıktı: $(a_{1,m}, a_{2,m})$

1: $a_{1,y} \leftarrow (a_{1,-1}, a_{1,0}, a_{1,1}, a_{1,2}, a_{1,3})$

2: $a_{2,y} \leftarrow (a_{2,-1}, a_{2,0}, a_{2,1}, a_{2,2}, a_{2,3})$

3: $l \leftarrow 1$

4: **for** $i \leftarrow n-1$ **to** 0 **do**

5: **if** $c_i = -1$ **then**

6: $a_{1,y} \leftarrow (a_{1,3l-3}, a_{1,3l-2}, a_{1,3l-1}, a_{1,3l}, a_{1,3l+1})$

7: $a_{2,y} \leftarrow (a_{2,3l-3}, a_{2,3l-2}, a_{2,3l-1}, a_{2,3l}, a_{2,3l+1})$

8: **end if**

9: **if** $c_i = 0$ **then**

10: $a_{1,y} \leftarrow (a_{1,3l-2}, a_{1,3l-1}, a_{1,3l}, a_{1,3l+1}, a_{1,3l+2})$

11: $a_{2,y} \leftarrow (a_{2,3l-2}, a_{2,3l-1}, a_{2,3l}, a_{2,3l+1}, a_{2,3l+2})\ a_{1,y-1} \leftarrow a^2_{1,y-1} - 2a^q_{1,y-1}$

12: **end if**

13: **if** $c_i = 1$ **then**

14: $a_{1,y} \leftarrow (a_{1,3l-1}, a_{1,3l}, a_{1,3l+1}, a_{1,3l+2}, a_{1,3l+3})$

15: $a_{2,y} \leftarrow (ta_{2,3l-1}, a_{2,3l}, a_{2,3l+1}, a_{2,3l+2}, a_{2,3l+3})$

16: **end if**

17: $l \leftarrow 3l + c_i$

18: **end for**

19: **return** $(a_{1,y}, a_{2,y})$

7.5.5. Sıkıştırma Faktörünün 4 ve 6 Olması

t bir tek tamsayı ($t=2l+1$) olmak üzere, $q=3^t$ olsun. O halde, $\sqrt{3q}=3^{l+1}$ şeklinde bir tamsayıdır ve

$$q^2-q+1=(q+\sqrt{3q}+1)\,(q-\sqrt{3q}+1)$$

olur. Buradan yola çıkarak Shirase ve arkadaşları, XTR algoritmasının geliştirilmiş versiyonunu takdim etmişlerdir [39]. Onlar, $\mathbb{F}_{q^6}^*$ içerisindeki $G_{r,q,6}$ nın $r \mid q-\sqrt{3q}+1$ özelliğini sağlayan elemanlarının 6 sıkıştırma faktörüyle $\mathbb{F}_q$ üzerinde $\{a_{1,m}\}$ kümesi ile tam olarak temsil edilebileceğini gösterdiler. Ayrıca, XTR algoritmasının benzerini ve aşağıdaki eşitlikleri kullanarak, bu elemanlar için bir üs alma algoritması tanımladılar.

$$a_{1,m}=Tr_{\mathbb{F}_{q^6}/\mathbb{F}_q}(\alpha^m)=\alpha^m+\alpha^{mq}+\cdots+\alpha^{mq^5},$$
$$b_{1,m}=Tr_{\mathbb{F}_{q^6}/\mathbb{F}_{q^2}}(\alpha^m)=\alpha^m+\alpha^{mq^2}+\alpha^{mq^4}.$$

Burada $a_{1,m}=b_{1,m}+b_{1,m}^q$ olduğu açıktır. Dahası, $\lfloor\ \rfloor$, en büyük tamsayı fonksiyonu ve $HW(\,.\,)$, Hamming ağırlık fonksiyonu olmak üzere, $a_{1,m}$ dizisinin m. terimini hesaplamak için $\mathbb{F}_q$ üzerinde yaklaşık $8\log_2 m+2\lfloor\log_2(t-1)\rfloor+HW(t-1)+2$ çarpma işleminin gerekli olduğunu göstermişlerdir. Açıkça görüleceği gibi, $a_{1,m}$ verilirse,

$$x^2-a_{1,m}x+a_{1,m}^{\sqrt{3q}}$$

polinomu kullanılarak $b_{1,m}$ etkili bir şekilde elde edilebilir, çünkü $b_{1,m}$ ve $b_{1,m}^q$ yukarıdaki polinomun kökleridir.

2010 yılında Karabina [28], Shirase ve arkadaşlarının yöntemini kullanarak, $\mathbb{F}_{q^6}^*$ içerisindeki $G_{r,q,6}$ nın $r \mid q\mp\sqrt{3q}+1$ özelliğini sağlayan elemanlarının 6 sıkıştırma faktörüyle $\mathbb{F}_q$ üzerinde $\{a_{1,m}\}$ kümesi ile tam olarak temsil edi-

lebileceğini göstermiştir. Ayrıca, altı farklı üs alma algoritması önererek bu algoritmaları karşılaştırmıştır. Bu algoritmalardan bir tanesinde $t = \mp 3^{l+1}$ Frobenius dönüşümünün izi olmak üzere

$$f(x) = x^6 - a_{1,m}x^5 + (a_{1,m}^t + a_{1,m})x^4 - (a_{1,m}^2 + a_{1,m}^t + 2)x^3 + (a_{1,m}^t + a_{1,m})x^2 - a_{1,m}x + 1$$

polinomunu kullanmıştır. Bu algoritma, Shirase ve arkadaşları [39] tarafından önerilen algoritmadan % 59 daha hızlıdır.

Diğer taraftan, t bir tek tamsayı ($t = 2l + 1$) olmak üzere, $q = 2^t$ olsun. Karabina $\mathbb{F}_{q^4}^*$ içerisindeki $G_{r,q,4}$ nın $r \mid q \mp \sqrt{2q} + 1$ özelliğini sağlayan elemanlarının 4 sıkıştırma faktörüyle $\mathbb{F}_q$ üzerinde $\{a_{1,m}\}$ kümesi ile tam olarak temsil edilebileceğini göstermiştir. Ayrıca, dört farklı üs alma algoritması önererek bu algoritmaları karşılaştırmıştır. Bu algoritmalardan bir tanesinde $t = \mp 2^{l+1}$ Frobenius dönüşümünün izi olmak üzere

$$f(x) = x^4 + a_{1,m}x^3 + a_{1,m}^t x^2 + a_{1,m}x + 1$$

polinomunu kullanmıştır.

7.6. n. DERECEDEN LFSR TABANLI BİR ŞİFRELEME ŞEMASI

Bu bölümde, 2017 yılında Kırlar ve Çil [29] tarafından önerilen $d \mid k$ ve $n = k/d > 2$ olmak üzere $\mathbb{F}_{q^d}$ sonlu cismi üzerinde tanımlı n. dereceden indirgenemez polinom tarafından üretilen n. dereceden lineer yineleme bağıntısı $a_{1,m}$ nin değişmelilik yasasına dayanan geçici-statik şifreleme şemasından (ephemeral-static encryption scheme) bahsedeceğiz.

Geçici statik açık anahtarlı şifreleme sisteminde, başlatıcı geçici, yanıtlayıcı ise statik bir anahtar kullanır ve eposta gibi ikisinin de çevrimiçi olmasına ihtiyaç olmadan iletişim kurabilirler. Kırlar ve Çil [29] tarafından önerilen

bu şifreleme şemasının güvenliği n. dereceden LFSR tabanlı problemleri çözmenin zorluğuna dayanır ve bu şema aynı zamanda semantik olarak da güvenlidir.

Giuliani ve Gong [19], n. dereceden indirgenemez polinom tarafından üretilen n. dereceden LFSR dizisi ile ilişkili bazı karmaşık problemler tanımlamıştır. Şimdi, aşağıdaki algoritmanın güvenliğinin bağlı olacağı bu problemleri vereceğiz.

Tanım 1: $f_1 := (a_{1,1}, a_{2,1}, \ldots, a_{n-2,1}, a_{n-1,1})$ ve $f_e := (a_{1,e}, a_{2,e}, \ldots, a_{n-2,e}, a_{n-1,e})$ verildiğinde, e yi bulma problemine n. dereceden LFSR tabanlı ayrık logaritma problemi (n-LFSR-DLP) denir.

Tanım 2: f_1, f_e ve f_r verildiğinde, f_{er} yi bulma problemine n. dereceden LFSR tabanlı Diffie-Hellman problemi (n-LFSR-DHP) denir.

Tanım 3: Düzgün rastgele bir z değeri için f_1, f_e, f_r ve f_z verildiğinde, $f_{er} = f_z$ olup olmadığını belirleme problemine n. dereceden LFSR tabanlı Diffie-Hellman karar (decisional) verme problemi (n-LFSR-DDHP) denir.

n-LFSR-DHP'nin, n-LFSR-DDHP'den net bir şekilde ayırt edilebilmesi için bazen n. dereceden LFSR tabanlı Diffie-Hellman hesaplama (computational) problemi (n-LFSR-CDHP) olarak adlandırıldığını belirtelim.

Not 3: $\mathbb{F}_{q^k}$ üzerinde n. dereceden LFSR-DLP'nin hesaplama olarak DLP ye eşit olduğu ispatlanmıştır [43]. Ayrıca Giuliani ve Gong, DHP ve DDHP'nin de benzer olduklarını ispatlamışlardır [19].

1982'de semantik anlamda güvenlilik kavramı ilk olarak Goldwasser ve Micali tarafından önerilmiştir [21]. Onlar, tanımlarını geliştirerek semantik anlamda güvenliliğin, şifreli metin ayırt edilemezliği olarak adlandırılan güvenliğe eşdeğer olduğunu göstermişlerdir. Semantik anlamda güvenliliğin

birçok farklı tanımı olmasına karşın, aşağıdaki versiyonu önerilen şemanın güvenliğinin gösterilmesinde yardımcı olacaktır.

Tanım 4 (Semantik anlamda güvenlilik): Herhangi iki açık metin m_1, m_2 ve bunlardan birinin şifrelenmesi ile elde edilen c şifreli metni verilsin. Polinom zamanlı düşmanlar, m_1, m_2, c ve açık parametrelere sahipken, $1/2$ den daha büyük bir olasılıkla, c nin m_1 ve m_2 den hangisinin şifreli metni olduğunu tahmin edemiyorlarsa, şifreleme şemasına semantik anlamda güvenlidir denir.

Algoritma 7.5. q bir asal ya da bir p asalının kuvveti olmak üzere, $k \geq 2$ pozitif bir tamsayı olsun. α, $G_{r,q,k}$ siklotomik alt grubun bir elemanı, $d \mid k$ ve $n = k/d > 1$ olmak üzere $\mathbb{F}_{q^d}$ üzerinde

$$f_\alpha(x) = x^n - a_1 x^{n-1} + \cdots + (-1)^{n-1} a_{n-1} x + (-1)^n$$

polinomu, α nın minimal polinomu olsun. Bu polinom tarafından üretilen karakteristik dizi $\underline{s} = \{s_i\}$ olmak üzere, α nın mertebesi $Q = \text{per}(f_\alpha)$, $1 + q + \cdots + q^{n-1}$ i bölsün.

$\mathcal{A}$ ve $\mathcal{B}$ iki taraf olsun. $\mathcal{B}$, $r \in \mathbb{Z}$, $0 < r < Q$ ve $\gcd(r, Q) = 1$ olmak üzere, statik gizli anahtarını rastgele seçer ve $f_r := (a_{1,r}, a_{2,r}, \ldots, a_{n-2,r}, a_{n-1,r}) \in \mathbb{F}_q^{n-1}$ statik açık anahtarını hesaplar.

- **Açık Parametreler:** f_1, f_r.
- **Gizli Parametreler:** r.
- **Şifreleme:** $\mathcal{A}$, $m \in \mathbb{F}_q$ mesajını şu şekilde şifreler ve $\mathcal{B}$ ye gönderir:

i. $\mathcal{A}$, $e \in \mathbb{Z}$, $0 < e < Q$ ve $\gcd(e, Q) = 1$ olmak üzere, geçici gizli anahtarını rastgele seçer ve $f_e := (a_{1,e}, a_{2,e}, \ldots, a_{n-2,e}, a_{n-1,e}) \in \mathbb{F}_q^{n-1}$ geçici açık anahtarını hesaplar.

ii. $\mathcal{A}$, değişmelilik yasasını ve $\mathcal{B}$ nin statik açık anahtarı f_r yi kullanarak $(a_{1,er}(f), a_{1,-er}(f)) = (a_{1,e}(f_r), a_{1,-e}(f_r)) \in \mathbb{F}_q^2$ ikilisini hesaplar.

iii. $\mathcal{A}$, $c = m + (a_{1,er} + a_{1,-er})$ değerini hesaplar ve şifreli metin $c \in \mathbb{F}_q$ ile geçici açık anahtarı f_e yi, $\mathcal{B}$ ye gönderir.

- **Şifre Çözme:** $\mathcal{B}$, $m \in \mathbb{F}_q$ mesajını şu şekilde elde eder:

  i. $\mathcal{B}$, değişmelilik yasası ve $\mathcal{A}$ nın geçici açık anahtarı f_e yi kullanarak $(a_{1,re}, a_{1,-re}) = (a_{1,r}(f_e), a_{1,-r}(f_e))$ yi hesaplar.

  ii. $\mathcal{B}$, $m = c - (a_{1,er} + a_{1,-er})$ değerini hesaplar.

7.6.1. Güvenlik Analizi

Kırlar ve Çil, önerdikleri Algoritma 7.5 in güvenliğini göstermek için Akleylek ve Kırlar'ın [1] çalışmasında belirttikleri, semantik anlamda güvenlilik (semantic security) [22], ayırt edilemezlik (indistinguishability) [22] ve değiştirilemezlik (non-malleability) [13] hedeflerine odaklanmışlar ve bu hususları detaylıca incelemişlerdir. Bu bölümde, Algoritma 7.5 in, Shannon'ın mükemmel gizlilik (perfect secrecy) kavramının hesaplama karmaşıklığı olarak da düşünülebilecek, semantik anlamda güvenliliğinden bahsedeceğiz.

Not 4: Kırlar ve Çil, Algoritma 7.5 in ayırt edilemez şifreli metin saldırısına (IND-CCA) karşı güvenli olduğunu göstermişlerdir. Bir kriptosistem IND-CCA ise, aynı zamanda seçilmiş düz metin saldırısı altında ayırt edilemezdir (IND-CPA) ve seçilmiş şifreli metin saldırısı altında da değiştirilemezdir (NM-CCA) [12].

n-LFSR-DLP, n-LFSR-DHP ve n-LFSR-DDHP problemlerinin çözülebilir olması durumunda, Kırlar ve Çil tarafından önerilen şifreleme şemasının

(Algoritma 7.5) kırılabilir olduğu açıktır. Tersi düşünülecek olursa, Algoritma 7.5 in güvenliği $a_{1,er}+a_{1,-er}\in\mathbb{F}_q$ toplamından $(a_{1,er},a_{1,-er})$ alt dizisinin belirlenmesine bağlıdır. [35] te ispatlandığı üzere, $a_{1,er}+a_{1,-er}\in\mathbb{F}_q$ toplamını ayrıştırmanın $\frac{q-1}{2}$ yolu vardır, ancak $(a_{1,er},a_{1,-er})$ alt dizisini açık bir şekilde tanımlayacak bir yöntem bilinmemektedir. $a_{1,er}+a_{1,-er}$ alt dizisinin toplanması, her ne kadar, tam dizi $f_{er}=(a_{1,er},a_{2,er},\ldots,a_{n-2,er},a_{1,-er})$ içerisinde kesin olarak yer alsa da, rastgele seçilmiş gizli anahtarlar e ve r bilinmeden, tam dizi f_{er} düşman tarafından hesaplanamaz. Bu sebeple, n-LFSR tabanlı probleme karşılık gelen alt dizi $(a_{1,er},a_{1,-er})$ 'nin de hesaplanması mümkün olmaz.

Not 5: Kırlar ve Çil, Algoritma 7.5 in tek yönlü (one-way) olduğunu ispat etmişlerdir. Daha kapsamlı bilgi için [29] a bakılabilir.

Teorem 3: Algoritma 7.5 semantik olarak güvenlidir (semantically secure).

İspat: $m_1,m_2\in\mathbb{F}_q$, düşman $\mathscr{E}$ tarafından bilinen iki mesaj olsun ve $\mathscr{E}$ bu iki mesajı şifrelemesi için $\mathscr{A}$ ya göndersin. $\mathscr{A}$, m_1 ya da m_2 mesajını $c=m+(a_{1,er}+a_{1,-er})$ şeklinde şifreler ve c şifreli metnini $\mathscr{E}$ ye gönderir. Ardından, m_1, m_2, c ve açık parametrelerin bilgisiyle, $\mathscr{E}$, c den m_1 i (sırasıyla m_2 yi), $c-m_1=d_1+(a_{1,er}+a_{1,-er})$ olacak şekilde (sırasıyla $c-m_2=d_2+(a_{1,er}+a_{1,-er})$) çıkarabilir. Burada, $d_1=m-m_1$ (sırasıyla $d_2=m-m_2$) olur. Bu bilgiden, d_1 in (sırasıyla d_2 nin) hesaplanmasının $a_{1,er}+a_{1,-er}$ den $a_{1,er}$ ve $a_{1,-er}$ alt dizisini elde etmeye eşdeğer olduğu ve bunun da mümkün olmadığı sonucuna varılır. Bu sebeple, $\mathscr{E}$ gözardı edilemeyecek şekilde $1/2$ den büyük olasılıkla c'nin m_1 ya da m_2 nin hangisinin şifreli metni olduğunu bulamaz.

7.6.2. Hesaplama Maliyeti

Kırlar ve Çil tarafından önerilen Algoritma 7.5 şifreleme şemasında, $\mathbb{F}_q$ üzerinde şifreleme ve şifre çözme prosedürlerinden gelen herhangi bir çarpma işlemi yoktur. Başka bir deyişle, toplam hesaplama maliyetleri, şifrelemede f_e tam dizisinin ve $(a_{1,er}, a_{1,-er})$ alt dizisinin, şifre çözmede de $(a_{1,er}, a_{1,-er})$ alt dizisinin hesaplama maliyetlerine karşılık gelir. Üstelik, f_r tam dizisi sabit olduğu için, f_r nin hesaplanması önerilen algoritmada bir ön hesaplama olarak da görülebilir.

Örnek olarak $n=3$ seçilirse, Ashraf ve Kırlar [3, Algoritma 2] tarafından sunulan verimli algoritma kullanılarak, belirli bir m ve başlangıç değeri $(a_{1,1}, a_{1,-1})$ için $(a_{1,m}, a_{1,-m})$ alt dizisinin $\mathbb{F}_q$ üzerinde hesaplama maliyeti $5.2\log m$ çarpma işlemi gerektirir. Bu sebeple, Algoritma 7.5 toplamda $\mathbb{F}_q$ üzerinde $15.6\log m$ çarpma işlemine gerek duyar. Benzer algoritmalarla yapılan karşılaştırma, sırasıyla $\mathbb{F}_q$ üzerinde çarpma işlemi $\mathbf{M}$ ve ters alma işlemi $\mathbf{I}$ ile gösterilmek üzere Tablo 7.1'de verilmiştir. $n=2,3,5$ için dizilerin hesaplama maliyetleri hakkında daha fazla detaylı bilgi için [3, 30, 42, 2] bakılabilir.

Tablo 7.1. k=3 için önerilen benzer şifreleme şemalarının karşılaştırılması

	XTR [30]	Improved XTR [42]	Improved GH [8]	Kırlar ve Çil [29]
Şifreleme	$16\log m\ \mathbf{M}$	$10,4\log m\ \mathbf{M}$	$(1+10,4\log m)\ \mathbf{M}$	$10,4\log m\ \mathbf{M}$
Şifre çözme	$8\log m\ \mathbf{M}$	$5,2\log m\ \mathbf{M}$	$1\ \boldsymbol{I}+(1+5,2\log m)$	$5,2\log m\ \mathbf{M}$

7.7. SIKIŞTIRILMIŞ EŞLEMELER

Sıkıştırılmış indirgenmiş Tate eşlemesi (ve türevleri) $Tr(e_r(P,Q)) = Tr(f_{r,P}(D)^{(q^k-1)/r})$ şeklinde tanımlanır [38]. Bu tanımın nasıl kullanıldığı Scott ve Barreto [38] tarafından detaylıca anlatılmıştır. Genel olarak, $\mathbb{F}_{q^k}^*$ daki $G_{r,q,k}$ siklotomik alt grubunun elemanlarının minimal polinomunun elementer simetrik fonksiyonlarının oluşturduğu dizilerin $(a_{1,m}, a_{2,m}, \ldots, a_{n-2,m}, a_{1,-m})$ kümesine karşılık gelir. Bazı eşleme tabanlı kriptografik protokoller, sıkıştırılmış eşlemelerden fayda sağlamak için kullanılır. Bunlar için verilebilecek en klasik örnek, Bölüm 3 de verilen BLS kısa imza şemasıdır [9]. Şimdi, sıkıştırılmış eşlemeler için Scott ve Barreto tarafından önerilmiş ve daha sonra biraz değiştirilmiş imza şemasını vereceğiz [38]:

$P \in E(\mathbb{F}_{q^k})$ mertebesi n olan bir nokta olmak üzere $G_1 = <P>$ olsun.

- **Anahtar Oluşturma:** Rastgele bir $c \in \mathbb{Z}_n^*$ seçerek cP değerini hesaplayın. c gizli anahtar ve ξ, cP nin x koordinatı olmak üzere açık anahtardır.

- **İmza:** $H:\{0,1\}^* \to G_1^*$ bir kriptografik özet fonksiyonu olsun. c gizli anahtarı ve $m \in \{0,1\}^*$ mesajı verildiğinde imzayı $S = cH(m) \in E(\mathbb{F}_{q^k})$ hesaplayın. İmza σ, $\mathbb{F}_{q^k}$ nın bir elemanı olan $S = cH(m)$ 'nin x koordinatı olur.

- **Doğrulama:** Bir ξ açık anahtarı, bir m mesajı ve bir σ imzası verildiğinde $e(P, \pm S) = e(\pm cP, H(m))$ veya

  $e(P, \pm S) = e(\pm cP, H(m))^{-1}$ olduğunu doğrulayın.

Herhangi bir eşleme değerinin üniter olması özelliğinden faydalanarak, BLS imza şemasını doğrulamak için $Tr(e(P, \pm S)) = Tr(e(\pm cP, H(m)))$ basitçe kontrol edilebilir.

7.8. SONUÇ VE DEĞERLENDİRMELER

Bu çalışmada, verimli sıkıştırma algoritmalarının açık anahtarlı kriptosistemlerdeki uygulamalarından bahsedilmiştir. Bunun için ilk olarak, $d \mid k$ ve $n = k/d > 1$ olmak üzere $\mathbb{F}_{q^d}$ sonlu cismi üzerindeki n. dereceden lineer yineleme bağıntısı tanımlanmış ve bu tanımdan faydalanılarak verimli sıkıştırma algoritmaları ifade edilmiştir. Sonrasında, bu verimli algoritmaların şifreleme ve imzalama şemalarındaki etkinliği ve eşleme tabanlı kriptografide son üs alma işlemindeki kullanışlılığı örneklerle gösterilmiştir.

Gelecek çalışmalarda;

- bu çalışmada bahsedilen n'nin bazı değerleri için tanımlanmış algoritmalara alternatif daha etkin algoritmaların tanımlanması,
- n. dereceden lineer yineleme bağıntısının özelliklerine dayanan daha etkin şifreleme ve imzalama algoritmalarının tanımlanması

beklenmektedir. Aynı zamanda;

- bu bölümde tanımlanan ve benzeri dizi tabanlı yapıların günümüz önemli teknolojilerinden blokzincir içerisinde kullanılıp kullanılamayacağının, kullanılabilirse bunun nasıl olacağının araştırılması da bir açık problem olarak değerlendirilmektedir, ve
- burada kullanılan sıkıştırma mantığının (sıkıştırma fonksiyonları, vb.), özellikle yakın gelecekte kuantum bilgisayarların kullanımının artması ile birlikte daha önemli bir yere sahip olacak kuantum sonrası kriptografik uygulamalar içerisinde daha fazla kullanılacağı düşünülmektedir.

KAYNAKLAR

[1]. S. Akleylek, B. B. Kırlar, New methods for public key cryptosystems based on XTR, Security and Communication Networks, Vol 8(18), pp 3682–3689, 2015.

[2]. E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, J. Comput. Appl. Math. 259-B, pp 599-621, 2014.

[3]. M. Ashraf, B. B. Kırlar, Message transmission for GH-public key cryptosystem, J. Comput. Appl. Math. 259-B, pp 578-585, 2014.

[4]. P. S. L. M. Barreto, H. Y. Kim, B. Lynn, M. Scott, Efficient algorithms for pairing-based cryptosystems, Advances in Cryptology - Crypto 2002, Lect. Notes in Comp. Sci., 2442 (2002), Springer-Verlag, 354-368.

[5]. P. S. L. M . Barreto, B. Lynn, M. Scott, On the selection of pairing-friendly groups, Selected Areas in Cryptography - SAC 2003, Lect. Notes in Comp. Sci. 3006 (2004), 17-25.

[6]. I. F. Blake, G. Seroussi, N. P. Smmart, Advances in Elliptic Curve Cryptography, London Math. Soc. Lect. Note Sci. 317, Cambridge, 2005.

[7]. D. Bleichenbacher, W. Bosma, A. K. Lenstra, Some Remarks on Lucas-Based Cryptosystems, Advances in Cryptology - Crypto'95, Lect. Notes in Comp. Sci. 963 (1995), 386-396.

[8]. D. Boneh, M. Franklin, Identity-based encryption from the Weil pairing, SIAM Journal of Computing, 32(3), pp 586-615, 2003.

[9]. [D. Boneh, B. Lynn, H. Shacham, Short signatures from the Weil pairing, Advances in Cryptology - Asiacrypt 2001, Lect. Notes in Comp. Sci. 2248, pp 514-532, 2002.

[10]. W. Bosma, J. Hutton, E. Verheul, Looking beyond XTR, Advances in Cryptology - Asiacrypt 2002, Lect. Notes in Comp. Sci. 2501, pp 46-63, 2002.

[11]. A. Brouwer, R. Pellikaan, E. Verheul, Doing more with fewer bits, Advances in Cryptology - Asiacrypt 1999, Lect. Notes in Comp. Sci., 1716, pp 321-332, 1999.

[12]. D. Catalano, R. Cramer, I. Damgard, G. D. Crescenzo, D. Pointcheval, T. Takagi, Contemporary Cryptology, Advanced Courses in Mathematics, 2005.

[13]. D. Dolev, C. Dwork, M. Naor, Non-malleable cryptography, Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, pp 542-552, 1991.

[14]. Q. Duanmu, X. Zhang, Y. Wang, K. Zhang, A New Public-Key Cryptosystem Based on Fifth-Order LFSR Sequence, The 1st International Conference on Information Science and Engineering (ICISE 2009), pp 1557-1560, 2009.

[15]. C. M. Fiduccia, An efficient formula for linear recurrences, SIAM J. Comput., 14, pp 106-112, 1985.

[16]. S. Galbraith, K. Harrison, D. Solera, Implementing the Tate pairing, Algorithmic Number Theory Symposium - ANTS V, Lect. Notes in Comp. Sci., 2369, pp 324-337, 2002.

[17]. K. Giuliani, G. Gong, Analogues to the Gong-Harn and XTR cryptosystems, Technical report, University of Waterloo, CORR 2003-34.

[18]. K. Giuliani, G. Gong, Efficient Key Agreement and Signature Schemes Using Compact Representations in , IEEE International Symposium on Information Theory - ISIT'04.

[19]. K. Giuliani, G. Gong, New LFSR-Based Cryptosystems and the Trace Discrete Log Problem (Trace-DLP), Sequences and Their Applications - SETA'04, Lect. Notes in Comp. Sci., 3486, pp 298-312, 2005.

[20]. K. Giuliani, G. Gong., A New Algorithm to Compute Remote Terms in Special Types of Characteristic Sequences, Sequences and Their Applications - SETA'06, Lect. Notes in Comp. Sci., 4086, pp 237-247, 2006.

[21]. S. Goldwasser, S. Micali, Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Information, Proceedings of 14 Annual ACM Symposium on Theory of Computing pp 365-377, 1982.

[22]. S. Goldwasser, S. Micali, Probabilistic encryption, Journal of Computer and System Sciences 28, pp 270-299, 1984.

[23]. G. Gong, L. Harn, Public-key cryptosystems based on cubic finite field extensions, IEEE Transactions on Information Theory 45 (7), pp 2601-2605, 1999.

[24]. G. Gong, L. Harn, H. Wu, The GH Public-key cryptosystem, SAC'01, Lect. Notes in Comp. Sci., 2259, pp 284-300, 2001.

[25]. G. Gong, A. Hassan, H. Wu, A. Youssef, An Efficient Algorithm for Exponentiation in DH Key Exchange and DSA in Cubic Extension Fields, Research report at Faculty of Math., University of Waterloo, 2002.

[26]. K. Hoffman, R. Kunze, Linear Algebra, Prentice Hall, New Jersey, USA, 2nd edition, 1971.

[27] M. Joye, J. J. Quisquarter, Efficient computation of full Lucas sequences, Electronics Letters, 32(6), pp 537-538, 1996.

[28] K. Karabina, Factor-4 and 6 compression of cyclotomic subgroups of $\mathbb{F}^*_{2^{4m}}$ and $\mathbb{F}^*_{3^{6m}}$, J. Math. Crypt., 4(1), pp 1-42, 2010.

[29]. B. B. Kırlar, M. Çil, On the k-th order LFSR sequence with public key cryptosystems, Mathematica Slovaca, 67(3), pp 601-610, 2017.

[30]. A. K. Lenstra, E. R. Verheul, The XTR public key system, Advances in Cryptology - Crypto'00, Lect. Notes in Comp. Sci., 1880, pp 1-19, 2000.

[31]. R. Lidl, H. Niederreiter, Finite Fields, Addison-Wesley Publishing Company, Reading, MA, 1983.

[32]. B. Lynn, On The Implementation of Pairing-Based Cryptosystems, PhD Thesis, 2007.

[33]. V. Miller, The Weil pairing, and its efficient calculation}, Journal of Cryptology, 17(4), pp 235-262, 2004.

[34]. A. Miyaji, M. Nakabayashi, S. Takano, New explicit conditions of elliptic curve traces for FR-reduction, IEICE Trans. Fund. Electron. Comm. Comput. Sci., E84-A(5), pp 1234-1243, 2001.

[35]. A. Muratovic, A. Ribic, Q. Wang, Partitions and Compositions over Finite Fields, The Electronic Journal of Combinatorics, 20(1), pp 1-14, 2013.

[36]. L. Quoos, S. F. Mjolsnes, Public Key Systems Based on Finite Field Extensions of Degree Five, Presented at Fq7 conference, 2003.

[37]. S. Samet, A. Miri, Privacy preserving ID3 using Gini Index over horizontally partitioned data, Proceedings of the 2008 IEEE/ACS International Conference on Computer Systems and Applications pp 645-651, 2008.

[38]. M. Scott, P. Barreto, Compressed pairings, Advances in Cryptology - Crypto 2004, Lect. Notes in Comp. Sci., 3152, pp 140-156, 2004.

[39]. M. Shirase, D. Han, Y. Hibin, H. Kim, T. Takagi, A more compact representation of XTR cryptosystem, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E91-A, pp 2843-2850, 2008.

[40]. P. J. Smith, M.J.J. Lennon, A New Public Key System, Proceedings of the IFIP TC11 Ninth International Conference on Information Security IFIP/Sec'93, pp 103-117, 1993.

[41]. P. Smith, C. Skinner, A public-key cryptosystem and a digital signature system based on the Lucas function analogue to discrete logarithms, Advances in Cryptology - Asiacrypt'94, Lect. Notes in Comp. Sci., 917, pp 357-364, 1995.

[42]. M. Stam, A. K. Lenstra, Speeding up XTR, Advances in Cryptology - Asiacrypt'01, Lect. Notes in Comp. Sci., 2248, pp 125-143, 2001.

[43]. C. H. Tan, X. Yi, C. K. Siew, On the n-th Order Shift Register Based Discrete Logarithm, IEICE Trans. Fundamentals., E86-A, pp 1213-1216, 2003.

[44]. S. M. Yen, C. S. Laih, Fast algorithms for LUC digital signature computation, IEE Proc. Comput. Tech., 142(2), pp 165-169, 1995.

[45]. C. H. Yu, S. S. M. Chow, K. M. Chung, F. H. Liu, Efficient Secure Two-Party Exponentiation, Topics in Cryptology--CT-RSA 2011, Lect. Notes in Comp. Sci., 6558, pp 17-32, 2011.

[46]. J. L. Yucas, Irreducible polynomials over finite fields with prescribed trace/ prescribed constant term, Finite Fields and Their Applications, 12, pp 211-221, 2006.

Bölüm 8

KRİPTOGRAFİDE RASTGELELİK

Muhiddin Uğuz

Rastgele sayılar kriptolojide oldukça büyük bir öneme sahiptir. Simetrik ve asimetrik anahtarlı tüm kripto sistemlerde kullanılan kriptografik anahtarlar rastgele sayılar olmalıdır. Kimlik doğrulama gerektiren tüm uygulamalar, aynı zamanda hücresel haberleşmede veri aktarımından önce doğrulama için baz istasyonu ve kullanıcı arasında gerçekleşen protokoller ve bunun gibi bir çok uygulama için rastgele sayılara ihtiyaç duyulmaktadır. Başka bir deyişle, tüm kriptografik parametrelerin oluşturulmasında rastgele sayılara ihtiyaç duyulmaktadır. Bu ihtiyacın karşılaması adına *Rastgele Sayı Üreteçleri* kullanılır. Bu üreteçler kriptolojinin dışında, çekilişlerde, oyun teorisinde, nümerik analizde, kuantum mekaniğinde, istatistiksel örneklemelerde, simülasyonlarda ve benzeri birçok uygulamada kullanılmaktadırlar. Rastgele sayı dizilerinin kriptografide, algoritmaların parametrelerini belirlemenin yanında, bir başka yaygın uygulama alanı daha bulunmaktadır. Blok şifreleme ve özet fonksiyonlarının rastgele fonksiyonlar gibi davranmaları beklenmektedir. Bu algoritmaların çıktılarının rastgele üretilmiş bir sayı dizisinden ayırt edilemiyor olmaları gerekir. Bu konudaki bir zafiyet algoritma için önemli güvenlik açıklarına sebep olabilmektedir. Girdi ve çıktılarda tespit edilebilecek bir ilişki, diferansiyel ve lineer saldırılara kapı açmaktadır. Bu sebeple tasarım aşamasında algoritmanın döngü (round) sayısına karar verirken, test aşamasında da blok şifre algoritmasını değerlendirirken rastgeleliği hakkında bir ölçü getirebilmek oldukça önemlidir.

8.1. GİRİŞ

Günlük hayatta birçok alanda rastgele sayı üreteçleri kullanılır. Bunlara bazı örnekler aşağıda verilmiştir. Bunlar;

- **Şans oyunları:** Belli kalıpların kolayca tespit edilmesi ilgili tarafa avantaj sağlayacaktır bu da oyunun cazibesini azaltır. Çekilişlerde kazanan numaralar da rastgele sayılar olmalıdır.
- **İstatistiksel uygulamalar:** Örneklemeler yaparken geneli temsil etmek adına genelden rastgele örnekleme alınmalıdır.
- **Sınavlar:** Yapılacak bir sınav için, adayların kimliklerini gizli tutma amacıyla aday numarası belirlenmesinde rastgele sayılar önem kazanır. Aday numaralarının rastgele olmaması durumunda, bu numaralarda adaya özgü bilgi kodlaması söz konusu olabilir, bu da sınavın adaletini ortadan kaldırabilir.
- **Şifreleme:** Anahtar veya tohum üretiminde, buna ek olarak, internet üzerinde iki cihazın güvenli bir şekilde haberleşmesine olanak sağlayan anahtar değişiminde rastgele sayılar güvenliğin en temel parçalarıdır.
- **Bilgisayar oyunları:** Her seferinde kendini tekrar eden oyunlar ilgi çekici olmaktan uzaktır. Tahmin edilemez parametrelerin de oyunun akışını etkiliyor olması gerekir. Oyuncunun aynı başlangıç durumunda oyuna başlaması durumunda bile her seferinde farklı bir oyunla karşılaşması oyunun cazibesini artırır.
- **Bilgisayar simülasyonları.**
- **Bilimde:** Rastgele sayılar kullanılarak oluşturulan rastgele örnekler üzerinde hipotezler test edilir. Örneğin yeni geliştirilen bir aşının insanlar üzerindeki etkisi rastgele sayılar kullanılarak yapılacak örneklemelerle tespit edilebilir.

Bu **örneklerin** sayısı çoğaltılabilir. İhtiyaç duyulan rastgele sayılar rastgele sayı üreteçleri kullanılarak elde edilir. Temelde iki tip rastgele sayı üreteci vardır. Bunlardan birincisi **Gerçek Rastgele Sayı Üreteçleri** (**TRNG**: True

Random Number Generator) olarak adlandırılır. Bu tip üreteçler, atmosferdeki gürültüler, radyoaktif çürüme veya yarı iletken dirençlerdeki ısı ölçümleri gibi bazı çevresel fiziki kaynakları kullanarak rastgele sayı üretirler. Bu üreteçlerin kullanılmasının pratikte yaratacağı en önemli problemlerden biri, gerektiğinde tekrarlanmasının ve dolayısıyla aynı sayı veya sayı dizisinin üretilmesinin imkânsıza yakın olması, yani aynı dizinin tekrar üretilemez olmasıdır. Bu üreteçlerin kriptografide kullanılabilmesi için, kullanıcının, yetkisiz kişilerin de aynı ölçümleri yapıp aynı sayıları üretemeyeceğinden emin olması gerekir. Bu da hem çevresel hem de matematiksel güvenliği bir arada gerektirir. Bu sebeplerden dolayı bu tür üreteçler bu bölümde ele alınacak konular kapsamı dışındadır.

Kriptografide genellikle ikinci tip sayı üreteci olan **Sözde Rastgele Sayı Üreteçleri** (**PRNG**: Pseudo Random Number Generator) kullanılır. Bu üreteçler nispeten kısa bir rastgele sayı dizisinden, yani tohum/anahtardan (seed/key), deterministtik algoritmalar kullanarak, rastgele bir diziden ayırt edilemeyecek sayı dizileri üretmektedirler. Tohumun kendisi, üretecin çıktısını belirlediği için, tamamen rastgele olmalıdır. Bu bakımdan sözde rastgele sayı üreteçlerini harekete geçiren gerçek rastgele sayı üreteçleridir. Bu diziler, saklanması veya paylaşılması daha kolay olan, aynı tohum kullanılarak, tekrar üretilebilir. Bu özellik, bu tür üreteçleri, özellikle simetrik şifreleme algoritmaları için, kullanıma elverişli hale getirir.

Burada üzerinde durulması gereken husus "*rastgele bir diziden ayırt edilemeyecek*" ifadesidir. Kararlı (Deterministtik) bir algoritma ile üretilmiş bir sayı dizisine, gerçek rastgele sayı dizilerinin gösterdiği bazı temel özellikleri sağlaması durumunda, ***sözde rastgele*** dizi adı verilir. Bu özellikler kümesindeki şartları birçok istatistiki testler yardımı ile kontrol eden algoritmaya da *Test Paketi* (Test Suite) adı verilir.

Bir dizinin rastgele olup olmadığını matematiksel olarak ispatlamanın bir yolu bulunmamaktadır. Bahsi geçen herhangi uzunluktaki her bir dizi tamamen rastgele olarak ortaya çıkmış olabileceği gibi, deterministtik bir yöntemle de üretilmiş olabilir. Nitekim belirlenen$n$ elemanlı bir $\{a_i\}_{i=1}^{n}$ sayı

dizisi için, nümerik analiz derslerinde öğretilen interpolasyon yöntemleri ile, her indis değerinde dizinin ilgili terimine eşit değer alan $(n-1)$. dereceden bir $P(x) = p_0 + p_1x + p_2x^2 \dots + p_{n-1}x^{n-1}$; polinomu oluşturmak oldukça kolay bir işlemdir. Nitekim, 1. terimi a_1, 2. terimi a_2, …, n. terimi de a_n sayısına eşit olan, başka bir deyişle her $i = 1 \dots n$ için a_i değerini alan Lagrange İnterpolasyon polinomu aşağıdaki şekilde verilir.

$$P(x) = \sum_{i=1}^{n} \frac{(x-1)(x-2)\dots\big(x-(i-1)\big)(x-(i+1))\dots(x-n)}{(x_i-1)(x_i-2)\dots(x_i-(i-1))(x_i-(i+1))\dots(x_i-n)} a_i$$

Bu sebeple, söz konusu sayı dizisinin hangi yöntemle üretildiği ya da ortaya çıktığı konusunda herhangi bir iddiada bulunmak mümkün değildir.

Öte yandan, deterministtik bir şekilde çalışan Sözde Rastgele Sayı üretecinin gerçekten "iyi" olduğunu, başka bir deyişle çıktılarının gerçekten rastgele bir şekilde elde edilmiş diziden ayırt edilemeyeceği, ya da bir blok şifreleme algoritmasının çıktılarının rastgele bir diziden ayırt edilemeyeceğini iddia edebilmek için, sayı dizileri için "gerçek rastgele dizilerden ayırt edilememe" kriterlerinin belirlenmesine, netleştirilmesine, bilimsel bir tabana oturtulmasına ihtiyaç vardır. Bu kriterler istatistik biliminin bizlere sağladığı bazı yöntemlerle tanımlanmakta ve test edilmektedir.

Yukarıda belirtildiği üzere bir sayı dizisinin rastgele olup olmadığına kesin olarak karar vermek imkânsızdır. Bunun yanında "rastgele" sıfatını da dikkatli kullanmak gerekir. Örneğin 5 sayısının bir rastgele sayı olup olmadığı, ya da $\{a_i\}_{i=1}^{4} = 5,7,3,4$ dizisinin rastgele olup olmadığı sorusu bir bakıma anlamsızdır.

Kazanan numaraların belli bir kurala bağlı olmadan belirlendiği 7 rakamlı biletlerden oluşan bir çekilişte ikramiyenin "rastgele" bir bilete çıkacağı aşikârdır. Bu durumda, örneğin 1,1,1,1,1,1,1 veya 1,2,3,4,5,6,7 veya 1,9,1,9,1,9,1 gibi, rakamlarının dizilişinde derhal fark edilebilen bir düzen bulunan numaraları taşıyan biletleri satın almaktan kaçınma eğilimi gözlenebilir. Bu eğilimi körükleyen temel güdü, söz gelimi, $\{a_i\}_{i=1}^{7} = 1,1,1,1,1,1,1$numaralı bir bilete, rastgele görünmediği için, büyük ikramiyenin çıkma olasılığının, örneğin $\{c_i\}_{i=1}^{7} =$

7,4,8,6,2,4,8 numaralı bilete göre çok daha az olduğu hissiyatıdır. Oysa ikinci biletteki numaralar da, her biri bir öncekinin $mod10$ da 2 katı olduğu için hiç de rastgele bir dizi değildir. Bu şekilde sayı dizisi üretme yöntemine literatürde *doğrusal denklik üreteci* ile sayı üretme adı verilir. Bu üreteçler önceden belirlenen a, b ve m büyük tam sayılarını kullanarak, tohum olarak kullanılacak bir X_0 tam sayısından $n = 1,2, \ldots$ için $X_{n+1} = aX_n + b(mod\ m)$ formülüne göre sayı dizileri üretirler.

Buna ek olarak, yukarıdaki $\{c_i\}_{i=1}^{7} = 7,4,8,6,2,4,8$ sayı dizisi aynı zamanda

$$P(x) = 57 - \frac{1141}{12}x + \frac{589}{10}x^2 - \frac{739}{48}x^3 + \frac{77}{48}x^4 - \frac{1}{48}x^5 - \frac{1}{240}x^6$$

polinomunun $x = 1,2, \ldots, 7$ tam sayı noktalarındaki değerleri olduğu için de rastgele bir dizi değildir denilebilir. Hatta daha da ileri gidilerek bir sonraki teriminin de (eğer tanımlıysa) $P(8)$ yani -21 olacağı da iddia edilebilir. Dikkat edilecek olursa bu -21 sayısı, dizinin bir doğrusal denklik üreteci ile üretilmiş olduğunu iddia eden bir kişinin savına göre, olması gereken 6 sayısı ile uyuşmamaktadır.

Yukarıdaki çekiliş örneğinde vurgulamak istenilen nokta şudur. Sonlu (uzunluktaki) diziler için bir rastgelelik tanımı yoktur. 1,1,1,1,1,1,1 dizisi ile 7,4,8,6,2,4,8 dizisi rastgelelik açısından ayırt edilemez. Nitekim her iki bilete de büyük ikramiyenin denk gelme olasılığı aynı, yani $\frac{1}{10^7}$ dir.

Buna göre sonlu sayı dizilerinin rastgeleliğinden anlaşılması gereken dizinin rastgele olup olmaması olamaz. Sonlu bir dizinin rastgeleliği ancak, elde edilme yöntemi için söz konusu edilebilir. Başka bir deyişle, **kısa bir diziden ziyade, o dizinin elde ediliş süreci veya onu oluşturan üretecin rastgeleliği söz konusudur.** Elde edilme yöntemine göre, her sonlu dizi rastgele kabul edilebileceği gibi, kabul edilmeyebilir de. Örneğin 7 sayıdan oluşan diziler üreten bir üreteç, eğer sözgelimi çıktı dizilerindeki terimlerin toplamı çoğunlukla 2021 sayısına eşit oluyorsa, bir rastgele dizi üreteci olarak kabul edilemez.

Rastgele sayı üreteci olarak kullanılacak olan bir üretecin tasarımında bir hata bulunması ve dolayısıyla tahmin edilebilir bazı özellikler yansıtmasının sonuçları çok vahim olabilir. Burada, "hata" bazen kasıtlı olarak da yapılmış olabilir. Kriptolu bir cep telefonun işletim sisteminde rastgele olması gereken parametrelerin gerektiği gibi olmaması sonucunda sistemin kolaylıkla kırılmaya elverişli olması ve dolayısıyla gizli kalması gereken konuşmaların açığa çıkması buna bir örnektir. Başka bir örnek ise, sınava girecek adaylara verilecek olan sınav aday numaralarının rastgele olmaktan uzak, bilakis bazı adayları ayırt edici olması rastgele sayı üreteçlerindeki zafiyetlerin sebep olabileceği sonuçların önemini vurgulama adına ikinci bir çarpıcı örnek olarak verilebilir. Bu sebeple, rastgele parametrelere ihtiyaç duyan algoritmalara kaynak sağlayacak olan üreteçlerin rastgelelik adına test edilmeleri büyük önem taşımaktadır.

8.2. RASTGELELİK TESTLERİ

Bu bölümde bir dizi veya üretecin rastgele olarak sınıflandırılmasının ne anlama geldiği, bu sınıflandırmaya temel teşkil eden postulatları, bu postulatların pratikte nasıl kullanılabileceği hakkında bilgiler verdikten sonra hipotez testi, test paketleri ve paket içeriğindeki testler arasındaki korelasyon ölçümleri hakkında bazı temel kavramlardan bahsedilecektir. Son olarak, bazı temel test örnekleri verilecektir.

8.2.1. Golomb'un Rastgelelik Postulatları

Golomb'un Rastgelelik Postulatları [1] sonlu (veya periyodik) dizilerin, rastgele bir diziden ayırt edilip edilemediğine karar vermek için hazırlanmış olan hemen hemen tüm rastgelelik testleri için önemli bir temel oluşturmaktadır.

Periyodik bir $0-1$ dizisinin rastgele olarak kabul edilebilmesi için üç önemli kavramdan bahsetmek gerekir:

- *Düzgün Dağılım* (Uniformity): Dizideki 1ve 0 terimleri düzgün olarak dağıtılmış olmalıdır.

- *Bağımsızlık* (Independence): Dizideki terimler birbirinden bağımsız olmalıdır.
- *Tahmin edilemezlik* (Unpredictability): Dizinin belli bir kısmının bilinmesi takip eden kısmı hakkında hiçbir bilgi vermemelidir.

Sözde Rastgele Dizi Üreteçleri birer *Sonlu Durum Makineleri* (Finite State Machines) olduklarından, ürettikleri diziler de periyodik, yani belli bir terimden sonra kendini tekrar eden diziler olacaktır. Yukarıda bahsedilen temel kavramları ölçme adına, literatürdeki birçok testte de temel olan, periyodik $0-1$ dizileri için üç maddeden oluşan postulatları şunlardır.

GP1: Dizinin bir periyodunda 0 ve 1 lerin sayısının farkı en fazla bir olmalıdır. Başka bir deyişle dizi *dengeli* (balanced) olmalıdır.

Dikkat edilecek olursa, rastgele bir diziden ayırt edilemeyecek bir dizinin dengeli olmasını beklemek her ne kadar doğal ve gerekli olsa da yeterli değildir. Sözgelimi uzunluğu 10 olan aşağıdaki diziler dengeli olmalarına karşın rastgele gibi kabul etmek rahatsızlık verir:

1,1,1,1,1,0,0,0,0,0

1,0,1,0,1,0,1,0,1,0

1,1,0,0,1,1,0,0,1,1

Bu gibi örnekler çoğaltılabilir. Birinci örnekte 5 uzunluğunda bloklar, 2 tane, yani "beklenenden" çok fazladır. İkinci örnekte 1 uzunluğundaki bloklar, 3. Örnekte de 2 uzunluğundaki bloklar rastgele bir dizide beklenen sayılara göre çok fazladır. Burada "beklenen" değerlerin matematiksel zeminde verilmesi gerekir. Öncelikle bir tanım verelim.

Bir dizide ardışık ve birbirinin aynısı terimlere ***öbek*** (run) adı verilir.

Sözgelimi 1,1,1,1,1,0,0,0,0,0 dizisinde 2 tane öbek bulunur ve her birinin uzunluğu 5'dir.

Benzer şekilde 1,0, 1,0, 1,0, 1,0, 1,0 dizisinde toplam 10 tane öbek vardır ve her bir öbeğin uzunluğu 1 dir. 1,1,0,0,1,1,0,0,1,1 dizisinde ise her biri uzunluğu 2 olan 5 tane öbek bulunur.

Son olarak 0,1,1,1,0,0,1,1,1,0 dizisinde uzunluğu 1 olan 2 tane, 2 olan 1 tane, 3 olan 2 tane olmak üzere toplam 5 öbek bulunmaktadır. Burada cevaplandırılması gereken soru, uzunluğu n olan bir ikili dizide toplam öbek sayısının beklenen değeri kaç olduğu ve bu sayının öbek uzunluklarına göre dağılımının nasıl olması gerektiğidir. Golomb'un ikinci postulat bu konudadır.

- **GP2:** Dizinin bir periyodundaki tüm *öbeklerin* sayısının dağılımı şu şekilde olmalıdır:
  - 1 uzunluğundaki *öbeklerin* sayısı tüm *öbeklerin* sayısının $\frac{1}{2^1}$ si,
  - 2 uzunluğundaki *öbeklerin* sayısı tüm *öbeklerin* sayısının $\frac{1}{2^2}$ si,
  - 3 uzunluğundaki *öbeklerin* sayısı tüm *öbeklerin* sayısının $\frac{1}{2^3}$ ü,
  - .
  - .
  - k uzunluğundaki *öbeklerin* sayısı tüm *öbeklerin* sayısının $\frac{1}{2^k}$ sı

kadar olmalıdır.

İkinci postulatın açıklamasını yapma adına bir tanım verelim:

Tanım: Periyodu n olan bir $\{a_i\}_{i=1}^{n} = a_1, a_2, \dots, a_n$ dizisinin ***türevi***

$$\{\triangle a_i\}_{i=1}^{n} = a_2 - a_1, a_3 - a_2, a_4 - a_3, \dots, a_n - a_{n-1}, \mathbf{1}$$

olarak tanımlanır.

Örnek olarak

$\{a_i\}_{i=1}^{n}$ =1,1,1,1,1,0,0,0,0,0 dizisinin türevi $\{\triangle a_i\}_{i=1}^{n} = 0,0,0,0,1,0,0,0,0,1$

$\{b\}_{i=1}^{n}$ =1,0,1,0,1,0,1,0,1,0 dizisinin türevi $\{\triangle b_i\}_{i=1}^{n} = 0,0,0,0,1,0,0,0,0,1$

$\{c\}_{i=1}^{n}$ =1,1,0,0,1,1,0,0,1,1 dizisinin türevi $\{\triangle c\}_{i=1}^{n} = 0,1,0,1,0,1,0,1,0,1$

dir. Dikkat edilecek olursa dizinin türevindeki son terim her zaman 1 olarak tanımlanmıştır ve türevdeki her bir 1 terimi, dizideki bir değişime karşılık gelmektedir. Buna göre dizinin türevinin ağırlığı, yani türevdeki 1 lerin sayı-

sı, son terim olarak tanımlanan 1 de hesaba katıldığında, dizideki öbeklerin sayısına eşittir. Aynı türeve sahip, birinin ilk terimi 0, diğerinin ilk terimi 1 olan tam iki tane dizi bulunur.

Bir dizi rastgele olarak kabul edilecekse, türevi de, son terimi hariç, rastgele kabul edilebilmelidir. Bu durumunda dizinin türevindeki 1 lerin beklenen sayısı yukarıda yazan GP1'e göre $\frac{n-1}{2} + 1 = \frac{n+1}{2}$dir.

Dizinin türevinde gözüken ardışık iki tane 1, orijinal dizide bir uzunluğundaki bir öbeğe işaret eder. Buna göre, rastgele görünen bir türev dizisinde $\frac{n+1}{2}$ tane beklenen 1 teriminin her birinin tekrar 1 ile devam etme olasılığı $\frac{1}{2}$ olduğundan, dizinin türevinde ardışık iki tane 1 lerin beklenen değeri $\frac{n+1}{4}$ dür. Yani dizide beklenen toplam öbek sayısının yarısı kadardır. Aynı şekilde, 2 uzunluğundaki öbeklerin sayısı, türev dizisinde 101 öbeğinin beklenen değeri olan, toplam sayının 4 de biri, 3 uzunluğundakiler 8 de biri, $k$uzunluğunda olan öbeklerin sayısının da toplam öbek sayısının 2^k da biri olarak beklenir.

- **GP3:** *Oto Korelasyon* (Autocorrelation) fonksiyonu $C(t)$ iki değerli bir fonksiyon olmalıdır: Başka bir deyişle matematiksel olarak, periyodu n olan bir $\{a_i\}_{i=1}^{n}$ dizisi için

$$C(t) = \sum_{i=1}^{n-1} (-1)^{(a_i + a_{i+t(modn)})}$$

fonksiyonunun değeri tüm $t \in \{1,2,\dots,n-1\}$ değerleri için aynı tam sayıya eşit olmalıdır ($C(0) = n$ olduğu aşikar). Bu fonksiyon, dizinin, kendisinin t kadar döngüsel olarak kaydırılmış hali ile ne kadar uyuştuğunu ölçmektedir.

Golomb'un üç maddede özetlediği, özelliklerin tam olarak sağlanıp sağlanmadığını ölçmek zor değil, ancak buradaki amaç bu özelliklerin hangi tolerans değerine kadar gevşetilebileceğidir. Örnek vermek gerekirse hilesiz bir para 100 kere atılırsa 50 keresinde tura gelmesi beklenir. Bir deney yaptığımızda 51 kere tura gelirse bu paraya hileli-dengesiz diyebilir miyiz? Demeli

miyiz? Peki, 55 kere gelirse? Ya da 40 kere gelirse? Bu durumda karar verilebilmesi için bir eşik değerinin belirlenmesi gerekir. Yani testin nasıl uygulanacağının kurgulanması gerekir. Tekrar vurgulamakta yarar var; 100 kere atılan bir para tam 50 keresinde de tura gelse, bu para kesinlikle hilesizdir diyemeyiz. Benzer şekilde, uzun veya kısa hiçbir dizi için, yapılacak olan istatistiki testler sonucunda, kesin bir karar verilemez. Ancak gözlemlenen değerler açısından kendisi gibi ya da daha uç istatistiki değer üreten dizilerin sayısının tüm dizilerin sayısına oranı belirlenebilir.

8.2.2. Dizilerin Değerlendirilmesi-Hipotez Testi

Rastgelelik kavramının, sonlu uzunluktaki diziler için iyi tanımlı bir anlamı olmadığı için, Sözde Rastgele Sayı Dizileri veya Üreteçlerinin testleri istatistiki yöntemler kullanılarak, söz konusu üreteçlerin ürettiği dizilerin incelenmesi ile gerçekleştirilir. İstatistiksel yöntemlerin güvenilir cevap verebilmesi için *örnek* sayısının büyük olması gerekir. Örnek sayısını büyük tutmanın da temelde iki yöntemi vardır.

- Uzun (terim sayısı büyük: 10^6 veya daha fazla) bir sayı dizisi almak,
- Kısa (terim sayısı küçük olan) sayı dizilerinden çok sayıda almak.

Başka bir deyişle rastgelelik testi

- Bir üretecin çıktısı olan bir uzun sayı dizisi için, ya da
- Kısa sayı dizileri üreten bir üreteçten alınan çok sayıda çıktı, başka bir deyişle, bir çıktı demeti için,

anlamlıdır. Her iki durumda da sonuç ele alınan dizinin değil; üretecin rastgele kabul edilip edilemeyeceği hakkındadır.

Öncelikle bazı temel tanımlardan bahsetmekte fayda var. Unutulmamalıdır ki bu bölümün amacı ne istatistik biliminin bir dalını ayrıntıları ile anlatmak ne de teorik bütünlük sağlamaktır. Amaç, kriptografik uygulamalardan kullanılan sayı üreteçlerinin rastgelelik testlerinin temel çalışma prensiplerini anlatmaktır. Bu sebeple, verilecek örnekler basit-çarpıcı tutulmaya çalışılacaktır.

Örnek-1:

Diyelim ki boyu çok uzun olan insanları belirlemek istiyoruz. Öncelikle bir rastgele değişken X tanımlamalıyız. Bizim örneğimizde X rastgele değişkenini, tanım kümesi insanlar, değer kümesi santimetre cinsinden $\{60,61,\ldots,260\}$ tam sayılar kümesi olan bir fonksiyondur (Kayıtlara girmiş, yürüyebilen en kısa insan 67cm, en uzun ise 251 cm boyundadır.).

X rastgele değişkeninin her bir insana atadığı değer, o kişinin santimetre cinsinden bir tam sayı olan boyudur. Bu durumda kimlere çok uzun boylu demeliyiz? (Benzer şekilde hangi dizilere rastgele değil demeliyiz?). Başka bir deyişle X değişkeni alacağı hangi değerler için, o değerleri aldığı kişileri çok uzun boylu olarak tanımlamalıyız?

Bu sorunun tam bir cevabı olamaz. Bazıları boyu 180cm den daha çok olanları çok uzun olarak tanımlarken, bazıları bu alt sınırı 195 cm olarak alabilirler. Ayrıca bu sınır değerleri ülkelere göre, cinsiyetlere göre de farklılaşır. Bu yüzden bu sorunun cevabı net bir sayı olamaz.

Bu soruya ikna edici bir cevap olarak şu tanım verile bilinir: "boyu, o toplumdaki insanlar arasında en uzun yüzde birlik dilimde olanlar çok uzun boyludur". Buradaki %1 lik *eşik değeri*, gösterim olarak α=0,01 seçimi, tamamen keyfidir. Değişik uygulama türlerinde farklı hassaslık değerleri gerekebilir.

Benzer şekilde anormal boyda olanlar kimlerdir? Normal boyda olma sınırı kaçtır? Bu soruya cevap, toplumda en uzun %0,05'lik ve en kısa %0,05'lik dilimde kalanların boyları anormal olarak tanımlanabilir.

Bir uzun sayı dizisini, ya da kısa sayı dizilerinden oluşan bir demeti istatistiksel testler kullanarak rastgeleliği ölçmek için istatistikteki en temel kavramlardan biri olan *hipotez testleri* kullanılır. Öncelikle H_0ile gösterilen ve söz konusu bir uzun dizi veya bir kısa dizi demetinin rastgele bir dizi veya demetten ayırt edilemeyeceğini kabul eden hipotez ile teste başlanır. Daha sonra bu hipotezin kabul edilip edilmeyeceğine karar verilir. Bu karar, bilimsel olması adına, dağılım fonksiyonları bilinen çeşitli rastgele değişkenler

kullanılarak verilir. Söz konusu dizi-demet için yapılan ölçüm değerleri, o rastgele değişkenin beklenen değeri ile karşılaştırılarak bir p-değeri elde edilir. Elde edilen p-değerinin 1'e eşit olması, söz konusu dizi-demetin tamamen rastgele kabul edilebileceğini, 0 olması durumunda ise rastgelelikten tamamen uzak kabul edilmesi gerektiğini belirtir.

p-değerinin önceden belirlenen bir $\alpha \epsilon [0,1]$ eşik değerini aşması durumunda H_0 hipotezi kabul edilir, yani dizi-demetin yeterince rastgele olarak kabul edilebileceği kararı verilir. Aksi durumunda, yani hesaplanan $p \in [0, \alpha)$durumunda, rastgelelik testine tabi tutulan dizi-demet rastgelelik testinden kalır. Başka bir deyişle, hesaplanan $p \in [0,1]$ değeri, söz konusu rastgele X değişkeninin, n uzunluğundaki tüm diziler söz konusu edildiğinde, rastgele seçilen bir dizi için alacağı değerin, değerlendirmeye tabii tutulan dizi için aldığı değer veya daha uç bir değer alması olasılığını verir. Tanım kümesinin sonlu veya sayılabilecek çoklukta olması durumunda bu durumu vurgulama adına X tesadüfi değişkenine *ayrık* (discrete) tesadüfi değişken adı verilir.

İstatistiki yöntemler vasıtası ile alınan kararda elbette bir kesinlik söz konusu olamaz. Hipotez testlerinde iki tip hata söz konusudur. Birincisi, **Tip-I hata** adı verilen hata, teste tabi tutulan söz konusu örnek, gerçekte H_0 hipotezini sağlamasına rağmen, hipotezin kabul edilmemesi hatasıdır. Bunun söz konusu bölümdeki karşılığı, gerçekten rastgele bir şekilde oluşmuş olan bir dizinin rastgele bir diziden ayırt edilemez olduğunu kabul etmemektir. Bu hatanın yapılmış olma olasılığı da önceden belirlenen $\alpha \epsilon [0,1]$ eşik değerine (hassaslık derecesi) eşittir. Bu değer birçok uygulamada $\alpha = 0{,}01$ olarak alınır. Bunun anlamı ortalama her 100 rastgele dizide 1 tanesinin Rastgelelik testinden kalmasıdır. Söz konusu diziye ait, X rastgele değişkeni için hesaplanan p değeri $0{,}01$ veya daha büyük ise söz konusu dizi X rastgele değişkeni bakış açısından, rastgele bir diziden ayırt edilememiş anlamına gelir.

Bir diğer hata şekli ise, **Tip-II hata** adı verilen, genelde daha vahim sonuçları olabilecek olan hatadır. Bu hatalara ait test sonuçları Tablo 8.1'de veril-

miştir. Bu gerçekte H_0 hipotezini sağlamamasına rağmen, hipotezin kabul edilmesi hatasıdır. Başka bir deyişle, rastgele olmayan bir dizinin rastgele kabul edilmesi hatasıdır. Bu tip hatanın olasılığı β'nın hesaplanması hemen hemen imkânsız olduğu için bu hata tipi ile burada ilgilenilmemektedir.

Tablo 8.1. Tip-I ve Tip-II Hata Test Sonuçları

	H_0 **hipotezi**	**Gerçek: Doğru**	**Gerçek: Yanlış**
Test Sonucu	Kabul	Kabul→Doğru Karar Olasılığı=1-α	Kabul→ Yanlış Karar (Tip-II hata) Olasılığı=β
	Ret	Ret → Yanlış Karar (**Tip-I** hata) Olasılığı=α	Ret → Doğru Karar Olasılığı=1-β

Örnek-2:

$n = 4$ durumunu ele alalım. Uzunluğu 4 olan toplam 16 tane 0-1 dizisi vardır. Bunlardan

1 tanesinin ağırlığı 0 ((0,0,0,0))

4 tanesinin ağırlığı 1 ((0,0,0,1), (0,0,1,0), (0,1,0,0), (1,0,0,0))

6 tanesinin ağırlığı 2 ((0,0,1,1), (0,1,0,1), (1,0,0,1), (0,1,1,0), (1,0,1,0), (1,1,0,0))

4 tanesinin ağırlığı 3 ((0,1,1,1), (1,0,1,1), (1,1,0,1), (1,1,1,0))

1 tanesinin ağırlığı 4 ((1,1,1,1)) dür.

Bu durumda uzunluğu 4 olan ikili dizilerde ağırlık ölçen X rastgele değişkeninin beklenen değeri

$$E(X) = \frac{1}{16}(1.0 + 4.1 + 6.2 + 4.3 + 1.4) = \frac{32}{16} = 2$$

olarak hesaplanır. Sözgelimi $\{a_i\}_{i=1}^{4} = (1,0,1,1)$ dizisinin ağırlığı 3, yani beklenen değerden 1 farklıdır. Buna göre, $\{a_i\}_{i=1}^{4}$ dizisi için p-değeri, başka bir deyişle, ağırlığı beklenen değerden 1 ya da daha fazla farklı olan dizilerin tüm dizilere oranı olan

$$p = \frac{1}{16}(1 + 4 + \cancel{6} + 4 + 1) = \frac{10}{16} = 0{,}625 \text{ tir.}$$

Benzer şekilde $\{b_i\}_{i=1}^{4} = (1,1,1,1)$ dizisinin ağırlığı 4, yani beklenen değerlen 2 farklı olduğu için, ağırlığı, bu dizi gibi, beklenen değerden en az 2 veya daha fazla farklı olan 4 uzunluğundaki dizilerin sayısının 4 uzunluğundaki tüm dizilerin sayısı olan 16 ya oranı, yani bu dizi için p değeri

$$p = \frac{1}{16}(1 + \cancel{4 + 6 + 4} + 1) = \frac{2}{16} = 0{,}125 \text{ tir.}$$

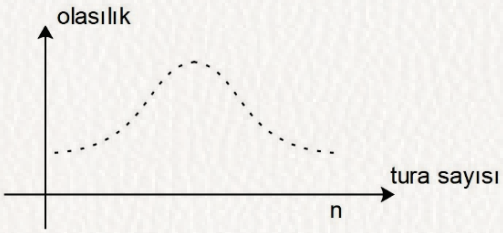
Dikkat edilecek olursa, yukarıda da bahsedildiği üzere, söz konusu dizi için hesaplanan rastgele değişken değeri, beklenen değerinden uzaklaştıkça p değeri de sıfıra yaklaşmaktadır. α eşik değeri, $\alpha = 0{,}2$ olarak belirlenirse, yukarıdaki iki dizi de rastgele olarak kabul edile bilinir. Öte yandan, $\alpha = 0{,}1$olarak belirlenmesi durumunda $\{a_i\}_{i=1}^{4}$dizisi rastgele olarak kabul edilecekken, $\{b_i\}_{i=1}^{4}$dizisi kabul edilemez. Bu durumda, eğer $\{b_i\}_{i=1}^{4}$dizisi gerçekte tamamen rastgele bir şekilde elde edilmiş ise, $\alpha = 0{,}1$ alınarak çalıştırılan testin sonucu olarak rastgele bir diziden ayır ediliyor kanaati getirilecek, başka bir deyişle, H_0 hipotezini sağlamasına rağmen hipotez kabul edilmeyecek, yani Tip-I hata yapılmış olunacaktır. Böyle bir durumla karşılaşma olasılığı da $\alpha = 0{,}1$ dir.

Yukarıda belirtildiği üzere, kullanılan istatistiki testte göre bir dizi için H_0 hipotezinin kabul edilip edilmemesi belirlenen α eşik değerine doğrudan bağlıdır.

Örnek-3:

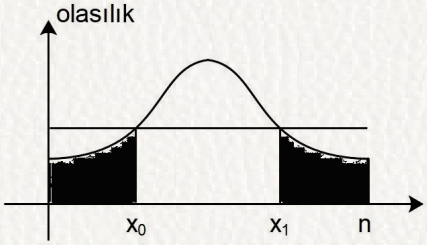
Bir madeni parayı n kere atalım ve çıkan yazı-tura sonuçlarını 0-1 olarak kodlayarak, n uzunluğunda bir dizi oluşturalım. Oluşabilecek dizilerin $\Omega_n$kümesinin toplam eleman sayısı 2^ndir. X ayrık tesadüfi değişkenini bir dizideki tura sayısı, yani elde edilen dizinin ağırlığı olarak tanımlayalım. Bu durumda X değişkeninin görüntü kümesi $T = \{0,1,2,\dots,n\}$ dir. Bir $t \in T$ için X değişkene ait olasılık dağılım fonksiyonu $f(t) \in [0,1]$ değeri, rastgele olarak seçilecek herhangi bir $\{a_i\}_{i=1}^{n} \in \Omega_n$ dizisi için,

$X(s) = t$ olma olasılığı olarak tanımlanır. Tura sayısı t olan dizilerin sayısı $\binom{n}{t}$ binom sayısı olduğundan $f(t) = \frac{\binom{n}{t}}{2^n}$ olarak hesaplanır. Her bir $t \in T$ için $f(t)$ hesaplanarak ayrık olasılık dağılım grafiği Şekil 8.1'de verilmiştir.



Şekil 8.1. Tura atışının ayrık olasılık dağılım grafiği

Bir madeni paranın hileli olup olmadığına karar vermek için bu para n kere atılsın ve gelen yazı ve tura sayıları kayıt edilsin. Hilesiz bir para n kere atıldığında $\frac{n}{2}$ tane tura gelmesi beklenir. Tura sayısının bundan çok az veya çok fazla olması durumunda kullanılan paranın hileli olduğu kanaatine varılır. Burada kullanılan 'çok az' ve 'çok fazla' terimleri göreceli kavramlardır ve somutlaştırılması gerekir. Bu amaçla x_0'dan küçük değerler ve x_1'den büyük değerlerin olasılıkları toplamı α olacak şekilde $x_0$ve x_1 değerleri belirlenir. Burada $x_0$ve x_1 değerleri 'çok az' ve 'çok fazla' kavramları yerine kullanılır. x_0 ve x_1 değerleri belirlenirken en küçük olasılıktan başlanarak sırayla ve tek tek α değerine ulaşana kadar toplanır.



Şekil 8.2. Tura atışının olasılığı

Şekil 8.2'de verilen grafikten görülebileceği gibi, değerlendirmeye tabi tutulacak dizi için hesaplanacak olan X rastgele değişkeninin değerinin x_0'dan küçük olması veya x_1'den büyük olması durumunda H_0 hipotezi kabul edilmez ve dizi Rastgelelik testinden kalır.

Örneği sayısallaştırmak gerekirse, bir madeni para 20 kere atılsın. Hilesiz bir para için beklenen tura sayısı 10 dur. X tesadüfi değişkeni gelen tura sayısını göstersin ve hassaslık seviyesi $\alpha = 0{,}04$ olarak belirlensin. Bu durumda, sınır değerleri olan $x_0$ve x_1 değerlerini belirlemek için elde edilen olasılık değerleri en küçükten başlanarak sırayla ve tek tek α değeri elde edilene kadar toplanır:

$$\frac{1}{2^{20}}\left[\binom{20}{0}+\binom{20}{1}+\binom{20}{2}+\binom{20}{3}+\binom{20}{4}+\binom{20}{5}+\binom{20}{15}+\binom{20}{16}+\binom{20}{17}\binom{20}{18}+\binom{20}{19}+\binom{20}{20}\right]$$
$$= 0{,}04139$$

Bu örnek için $\alpha = 0{,}04$ hassaslık derecesi, $x_0 = 5$ ve $x_1 = 15$ sınır değerlerini belirlemiştir. Başka bir deyişle rastgele seçilen 20 uzunluğundaki bir dizide 1'lerin sayısının 5'e eşit veya 5'ten daha az ya da 15'e eşit veya 15'ten daha fazla olma olasılığı $0{,}04139$'dur ve bu değer $\alpha = 0{,}04$ hassaslık derecesini geçen en küçük değerdir. $\alpha = 0{,}04$ için belirlenen bu $0{,}04139$ değerine *gerçeklenen hassaslık değeri* anlamında *gerçeklenen* α adı verilir.

Sonuç olarak belirlenen α hassaslık değeri için bir madeni para 20 defa atıldığında gelen tura sayısı 5'e eşit veya 5'ten daha az ya da 15'e eşit veya 15'ten fazla ise paranın hileli olduğu kanaatine varılır. Bu kanaatin hatalı olması, yani aslında hilesiz bir paranın bu sonucu vermiş olma olasılığı, belirlenen $\alpha = 0{,}04$ değerinden küçüktür. Başka bir α seçildiğinde bu sınır değerleri farklı olacaktır.

Buraya kadar verdiğimiz örneklerin hepsi uzun dizileri test etmek içindi. Şimdi ise bir kısa dizi üretecini test etme yöntemi konusundan kısaca bahsedelim. Sözgelimi, 64 bit uzunluğunda diziler üreten bir üreteci rastgelelik açısından test etmek isteyelim. Öncelikle, bir X rastgele değişkeni ve α eşik değeri belirlenmeli ve görüntü kümesi $T = \{t_1, t_2, \ldots, t_k\}$ olan X değişkeni-

ne ait olasılık dağılım fonksiyonu hesaplanır. Bu üreteçten alınacak $n$tane örnekten (kısa diziden) $f(t_i)n$ tanesi için olasılık dağılım fonksiyonu f'nin t_i değeri alması beklenir. Genelde 5 ila 8 arasındaki *kutu*, gene genellikle eşit toplam olasılıklarla belirlenir. Söz gelimi i. kutu $K_i = \{t_{i_1}, t_{i_2}, \dots, t_{i_{ki}}\}$ olarak gösterilmek üzere Tkümesinin bir parçalanışı ($\cup K_i = T$, ve i≠j için $K_i \cap K_j = \{\ \}$ -$boş\ küme$) belirlenir. Burada, her bir i değeri için $p(K_i) = f(t_{i_1}) + \cdots + f(t_{i_{ki}})$ olasılık değeri toplamları genelde birbirine yakın olacak şekilde ayarlanır. Öncelikle her bir kutunun sayacı sıfırlanır. n tane kısa diziden her biri için X değişkeninin değeri hesaplanır ve bu değer hangi kutu ile ilişkilendirilmiş ise o kutunun sayacı bir artırılır. İşlem bittiğinde her bir kutunun sayacının son değeri, başka bir deyişle gözlenen değeri O_i(observed value) beklenen değer olan $E_i = \sum np(K_i)$ (expected value) ile karşılaştırılır. Bu karşılaştırma serbestlik derecesi kutu sayısının 1 eksiği alınarak hesaplanan Ki-Kare değeri

$$X^2 = \sum \frac{(E_i - O_i)^2}{E_i}$$

kullanılarak yapılır. p-değeri

$$p = igam\left(\frac{k-1}{2}, \frac{X^2}{2}\right) = \int_{\frac{X^2}{2}}^{\infty} t^{\frac{k-1}{2}-1} e^{-t} dt$$

olarak hesaplanarak üretecin rastgeleliği hakkında görüş oluşturulur.

Burada bahsedilen kısa dizi demetlerini, başka bir değişle, kısa dizi üreteçlerini test etme yöntemi bazen uzun dizileri test ederken de kullanılmaktadır. Uzun bir diziyi test ederken diziyi olduğu gibi bir bütün olarak, yapısını koruyarak ele almak tercih edilse de, bazen bu doğal yaklaşım çeşitli hesaplama problemlerine sebep olmaktadır. Bu durumda, X rastgele değişkeninin dağılım fonksiyonu yerine çeşitli yakınsamalar, asimptotik yaklaşımlar kullanılması zorunlu hale gelmektedir. Örneğin binom dağılımı yerine normal dağılım, kullanılmakta bu da hesaplamalarda bazı hatalara sebep olmaktadır.

Bu durumun üstesinden gelmenin bir yöntemi de, test edilecek olan uzun diziyi *parçalara* ayırarak, kısa diziler demeti olarak görmektir. Burada 'parçalama' işlemi için de farklı yaklaşımlar söz konusudur. Birinci yaklaşım ki literatürde de genellikle kullanılmış olan yaklaşım budur, uzun diziyi **sabit uzunlukta** olan kısa diziler demetine ayırma şeklindedir. Söz gelimi, uzun diziyi 128 bit uzunluğunda parçalara ayırarak kısa dizi demeti oluşturmak: $\{a_i\}_{i=1}^{n}$ dizisinden $k = 0,1, \ldots.$ için her biri sabit 128 bit uzunluğunda olan, $\{a_i^k\}_{i=128k+1}^{128(k+1)}$ dizilerden oluşan kısa diziler demeti oluşturup, demeti test etmektir. Parçalama işlemi, yukarıda belirtildiği üzere, ***kesişmeyecek*** (non-overlapping) şekilde yapılabileceği gibi, ***kesişecek*** (overlapping) şekilde de yapılabilir. Bu duruma örnek olarak bir kaydırmalı kısa diziler demeti verilebilir: $\{a_i\}_{i=1}^{n}$ dizisinden $k = 0,1, \ldots.$ için her biri sabit 128 bit uzunluğunda olan, $\{a_i^k\}_{i=k+1}^{k+128}$ dizilerden oluşan kısa diziler demeti oluşturup, demet test edilir.

Sabit uzunlukta parçalama yaklaşımının bir dezavantajı, uzun dizinin, çok da doğal olmayan bir şekilde parçalanması sonucunda bazı hassas bilgilerin kaybedilme tehlikesidir. **Dinamik (değişken) parçalama** [2] adı verilen bir diğer yaklaşım, ise söz konusu X rastgele değişkenine bağlı olarak, diziyi değişken uzunluklardaki kısa dizilere ayırarak kısa dizi demeti elde etmedir. Bu yaklaşımı açıklama adına Xdeğişkenini dizinin ağırlığını veren değişken olarak düşünelim. Önceden belirlenen bir n_0 pozitif tam sayısı için, uzun dizinin ağırlığının n_0 değerine ulaştığı ilk indeks kayıt edilir ve dizinin bu ilk kısmı silinir. Dizinin kalan kısmı ile aynı işlem tekrarlanır. Bu işlem sırasında dizinin ağırlığı, önceden belirlenen makul bir uzunluk sonucunda, örneğin t terim sonucunda, n_0 değerine ulaşmaz ise t indeksi kayıt edilir ve dizinin bu t terimi silinir. Burada $t$sayısı, bu terime kadar ağırlığın n_0 değerine ulaşmamış olma olasılığı oldukça küçük olacak şekilde belirlenir. Elde edilen kısa dizi demeti Ki-Kare testi yöntemi ile değerlendirmeye alınır. Örnek olarak $n_0 = 128$ ve $t = 280$ seçimlerine göre Ki-Kare tablosu Şekil 8.2'de verilmiştir.

Tablo 8.2. Ki-Kare Test Sonuçları

i	1	2	3	4	5	6	7	8
Parça Uzunluğu	128-238	239-245	246-250	251-255	256-260	261-266	267-274	>274
Olasılık Değeri	0,13522	0,12626	0,11445	0,12404	0,12171	0,12823	0,12455	0,12549

8.2.3. Test Paketi

Bir uzun dizinin, ya da kısa diziler demetinin rastgele olarak kabul edilebilir olup olmadığına bir test, söz gelimi *ağırlık testi* ile karar vermek doğru olmaz. Zira, söz gelimi, ilk 2^{n-1} terimi 0, takip eden terimleri de 1 olan $0,0,\dots,0,1,1,\dots,1$ dizisi, ya da $1,0,1,0,1,0,\dots$ dizisi bu testten mükemmel olarak geçerler, oysa rastgele bir dizi olarak kabul etmek en başta sağduyumuz ile çelişir. Bu sebeple literatürde birçok test tanımlanmıştır. Genellikle rastgeleliğe karar vermek için birden fazla testten oluşan *test paketleri* kullanılır.

Bir test paketi, söz konusu uzun bir diziyi veya bir kısa dizi demetini, birbirinden bağımsız yönleri ile kontrol eden, "yeterli" sayıda testten oluşur.

Pakette yer alan her testin, pratikte hesaplanabilir *ağırlıklı olasılık dağılım fonksiyonunun* (cumulative probability distribution functions) hesaplanmış olması, bu testlerin birbirinden bağımsız olması, başka bir ifadeyle, *korelasyon*larının düşük olması, bir yandan daha çok yönden kontrol yapabilmek için sayılarının çok olması, diğer yandan da hem zaman hem işlemci açısından masrafının çok olmaması, yani test paketinin *ekonomik* olması ve son olarak da bıraktığı, rastgele kabul etmediği, dizi/üreteçlerin oranının makul düzeyde olması beklenir. Farklı uygulamalar için farklı testler pakete dâhil edilebilir, paketten çıkarılabilir, ya da farklı parametrelerle kullanıla bilinir. Bu sebeple hiçbir zaman bir paket bir tam paket olarak kabul edilemez.

Farklı uygulamalar veya farklı hassasiyetlere yönelik farklı test paketleri oluşturulabilinir. Bir test paketinin güvenilirliği açısından birçok testi ihtiva etmesi gerekirken, bilinçsizce kalabalıklaştırılmış bir paketinde güvenilirliği yüksek olacak diye bir şey söz konusu değildir. Bu durum ek olarak işlem sayısı-zaman-hafıza gibi faktörlerde de gereksiz şişmelere sebep olacaktır. Bu sebeple bir test paketi oluştururken güvenilirliği ve etkinliği artırma yönünde uygun testleri seçmek hayati önem taşımaktadır.

Bir test paketi oluştururken aşağıdaki maddeler göz önüne alınmalıdır:

- Test paketinde yer verilecek testlerin her biri için matematiksel altyapının geliştirilmiş, olasılık dağılım fonksiyonları pratikte hesaplanabilecek şekilde elde edilmiş olmalıdır.
- Pakette değişik bakış açılarını değerlendiren yeterli sayıda test bulunmalıdır.
- Paket amacı doğrultusunda hazırlanmalıdır. Kısa dizileri değerlendirecek bir paket ile uzun dizileri değerlendirecek bir paket ortak testler içerebileceği gibi amaçlarına uygun farklı testler de içermelidir.
- Test Paketinin *Kapsamı* (coverage), paketteki en azından bir testten kalan dizilerin sayısının örnek diziler kümesindeki dizi sayısına oranı olarak tanımlanır. Bu oranın düşük olması, paketteki test sayısının artırılması gerekliliğine ya da var olan testlerdeki parametrelerin değiştirilmesine bir işaret olarak kabul edilir.
- Paket kapsamındaki testler aralarındaki *korelasyonlar* hesaplanmış olmalıdır. Bu değerler bazı özel amaçlar için daha hafif bir paket hazırlanması gerektiğinde, hangi testlerin elenebileceği hakkında bilgi verir. Ayrıca paketin korelasyonu yüksek birçok test içermesi, hem çalışma zamanı, hem de hesaplama gereksinimleri açısında olumsuzluklar yaratabilir. Korelasyonların küçük olması bir anlamda Kapsamın (coverage) yüksek olması demektir. İki test arasındaki korelasyon genelde Pearson Korelasyon Katsayısı ile ölçülür. Bu katsayı -1 ile 1 arasında değer alır ve birbirinden bağımsız testler için sıfıra

yakın değerler üretir. Temelde iki test arasında doğrusal bağlantıları kontrol eden bu testin sonucunun 1 e veya -1 e yakın olması testlerin aralarında kuvvetli bir pozitif veya negatif doğrusal bağ olduğu anlamına gelir. Test tek yönlüdür. Başka bir deyişle korelasyon değerinin sıfıra yakın olması testlerin bağımsızlığını garanti etmez. Burada karşılaştırma, sadece iki test arasında değil, iki test gurubu arasında da yapıla bilinir.

- Pakete yeni bir test ekleneceği zaman bu yeni testin var olan pakete katkısı hesaplanmalıdır. Bu hesaplama şu şekilde yapılabilir: Pakette var olan testlerin mevcut kapsamı C_0, iken yeni test eklenince paketin kapsamı C_1olsun.$C_1 - C_0$ farkının pakette belirlenen eşik değerine yakın olması beklenir. Eğer bu değer eşik değerinden çok düşük olması, ya yeni eklenen testin kendi başına oldukça zayıf bir test olduğu, ya da pakette hâlihazırda var olan bazı testler ile yüksek derecede korelasyonu var olduğu anlamına gelir. Bu durumda yeni testi pakete eklemek için bir kez daha düşünmek gerekir.

Sonuç olarak, bir test paketi, hazırlanma amacı doğrultusunda, bir yandan kapsamı olabildiğine geniş olacak şekilde birçok test içermeli, öte yandan da paketi çalışma zamanı, işlemci gücü, hafıza gibi kalemler sebebi ile pratik olmaktan uzaklaştırmamalıdır.

Tüm bu maddelerin ışığı altında tasarlanacak ve birbirinden bağımsız k tane testten oluşan bir test paketi için, test paketinin eşik değeri, yani Tip-I hata olasılığı α olarak belirlenmesi durumunda, her bir testin, birbirinden bağımsız olduğu kabulü ile, çalıştırılması sırasında kullanılacak olan μ eşik değeri ile α arasındaki

$$1 - (1 - \mu)^k = \alpha$$

bağıntısından μ eşik değeri hesaplanabilir.

μ=0,01 eşik değeri ile çalıştırılacak olan her bir test için (Tablo 8.3), birbirinden bağımsız k adet testten oluşan bir test paketinin paket eşik değeri olan α değeri, aşağıdaki tabloda $k = 1,2,..,9$ için listelenmiştir.

Tablo 8.3. Ki-Kare Test Sonuçları

Test Sayısı	α
1	0,010000
2	0,019900
3	0,029701
4	0,039404
5	0,049010
6	0,058520
7	0,067935
8	0,077255
9	0,086483

Burada dikkat edilmesi gereken önemli bir nokta var. Birbirinden bağımsız k tane testten oluşan bir test paketi için, test paketinin eşik değeri olacak α'nın belirlediği her bir testte kullanılacak olan μ değerinden bahsetmiştik. Ancak, ayrık rastgele değişken kullanıldığında tam olarak μ eşik değerini veren test parametreleri mevcut olmayabilir. Bu durumda μ yerine ona en yakın değer alınır. Paketteki i. test için belirlenen eşik değerini μ_i ile gösterirsek, test paketinin *gerçek eşik değeri* de α değil ama ona oldukça yakın olan

$$\alpha_{gerçek} = 1 - [(1-\mu_1)(1-\mu_2)\dots(1-\mu_k)]$$

olur. Buna alternatif bir yaklaşım ise dizinin paketteki testlerden bir tanesinden kaldığı anda H_0 hipotezinin reddidir. Bu yaklaşımın daha genellemesi şu şekilde verilebilir: belirlenen bir sayıdaki testten, örneğin $r(k)$ tane testten kalması durumunda hipotezin reddidir. Başka bir değişle, k tane, birbirinden bağımsız, testten oluşan bir pakete sokulan bir dizi r veya daha az sayıda testten kalırsa H_0 hipotezi ve dolayısıyla dizinin rastgele bir diziden ayırt edilebilecek bir özelliğinin tespit edilemediği kabul edilecektir. Bu durumda aşağıdaki iki maddenin doğruluğunu varsaydığımızda:

- Pakette bulunan k tane testin her biri diğerlerinden bağımsız,
- Paketteki her bir testin p değeri sürekli, yani $\alpha_{gerçek} = \alpha$,

$$\alpha = 1 - \sum_{i=0}^{r-1} \binom{k}{i} \mu^i (1-\mu)^{k-i}$$

Eşitliğinden α değeri hesaplanabilir. Söz gelimi test paketinin $k = 47$ bağımsız testten oluşması ve bu testlerin her birinin sürekli p-değeri olması durumunda, paketin eşik değerinin $\alpha = 0{,}01$ olması için $r = 2,3$ veya 4 durumlarında her bir testi çalıştırırken alınması gereken μ eşik değerleri Tablo 8.4'te verilmiştir.

Tablo 8.4. Eşik Değerleri μ Sonuçları

$\alpha = 0{,}01$	
r	μ
2	0,00318982
3	0,00943612
4	0,0179369

Buna karşılık olarak, her bir test için eşik değerinin $\mu = 0{,}01$ olarak belirlenmesi durumunda test sayısı r nin $r = 2,3$ veya 4 değerleri için oluşacak paketin eşik değeri olan α Tablo 8.5'te verilmiştir.

Tablo 8.5. Eşik Değerleri α Sonuçları

$\mu = 0{,}01$	
r	α
2	0,08045749844
3	0,0116858474
4	0,00126590028

Literatürde birçok rastgelelik testi ve dolayısıyla birçok rastgelelik test paketi, bulunmaktadır. Bunlardan bazıları aşağıda listelenmiştir.

- Knuth'un "Programlama Sanatı" kitabı [3] (11 adet istatistiksel test tanımlanmıştır.)
- Ruhkin'in Rastgelelik Testi [4],
- DIEHARD [5] (15 adet istatistiksel test tanımlanmıştır.)
- DIEHARDER [6] (26 adet istatistiksel test tanımlanmıştır.)
- CRYPT-X [7] (8 adet istatistiksel test tanımlanmıştır.)
- TestU01 [8] (16 adet istatistiksel test tanımlanmıştır.)
- NIST [9] (16 adet istatistiksel test tanımlanmıştır.)

8.2.4. Testler Arasındaki Korelasyon Ölçümleri

Bir test paketi, birbirlerinin çok benzeri, dizilerin aynı özelliklerini değerlendiren testler içermemelidir. Testlerin aralarındaki korelasyonları belirlemenin bir yöntemi de testlerin diziler ve aynı dizilerin dönüşüme uğramış halleri üzerindeki sonuçlarını değerlendirmektir. Testlerin bir diziyi olduğu gibi değerlendirmesinin yanı sıra, diziye bazı dönüşümler uygulayıp, elde edilen yeni dizinin de testler tarafından değerlendirilmesi ve sonuçların karşılaştırılması, başka bir deyişle hangi testlerin hangi dönüşümlere benzer tepkiler gösterdiğinin belirlenmesi önemlidir. Örneğin ağırlık testi dizinin elemanlarının yer değiştirmesine duyarsız kalacaktır. Öte yandan öbek (run) testlerinin sonuçları tamamen farklılık gösterecektir. Bu bağlamda bu iki testin de bir pakette bulunması gerekir. Öte yandan hem ağırlık testi hem de blok testi dizinin ters sırada yazılmasına duyarsız kalarak bu yönde bir benzerlik gösterirler.

Burada dönüşümler derken ne gibi dönüşümleri kastettiğimizi bazı örnekler vererek açıklamakta fayda var. Değerlendirmeye alınacak bir $\{a_i\}_{i=1}^{n} = a_1, a_2, a_3, a_4, a_{5,\dots,} a_n$dizisi için

- Dizinin tümleyenini almak, yani $(a_1, a_2, a_3, \dots, a_n)$ dizisi ile sabit $(1,1, \dots,1)$ dizisinin terimlerini mod 2 de toplamak (xor larını almak):

$$(a_1 \oplus 1, a_2 \oplus 1, a_3 \oplus 1, \dots, a_n \oplus 1)$$

- Diziyi ters çevirmek:

$$(a_n, a_{n-1}, a_{n-2}, \dots , a_1)$$

- Dizinin elemanlarını ikili olarak yer değiştirmek

$$(a_2, a_1, a_4, a_3, a_{6,\dots})$$

- Dizinin baş kısmı ile son kısmının yerlerini değiştirmek:

$$(a_k, a_{k+1}, a_{k+2}, \dots, a_n, a_1, a_2, \dots, a_{k-1})$$

- Bu ve benzer dönüşümlerin birden fazlasını ardışık olarak uygulamak, örneğin:

$$(a_k, a_{k+1}, a_{k+2}, \dots, a_n, a_{k-1}, a_{k-2}, \dots a_2, a_1)$$

Bu konuda Doğanaksoy vd. tarafından yapılan bir çalışma [10] örnek olarak verilebilir.

Kitabın bu bölümünün bundan sonraki kısmında, temelde yukarıda verilen Golomb'un postulatların ışığında tasarlanmış olan Rastgelelik testlerine bazı örnekler verilecektir.

8.2.5. Bazı Test Örnekleri

1-Ağırlık Testi:

Örnek olarak, $\{a_i\}_{i=1}^{16} = \{1,1,0,1,1,0,0,1,1,1,0,1,1,1,0,0\}$ dizisinin ğı $wt(\{a_i\}_{i=1}^{16}) = 10$ dur. 2^nçeşit n uzunluğundaki her bir $\{a_i\}_{i=1}^{n}$ dizisi için, o dizideki 1lerin sayısı olarak tanımlanan ve dizinin ağırlığını veren X rastgele değişkenini ele alalım. Bu değerin k veya daha küçük olma olasılığı olarak tanımlı *Kümülatif Olasılık Dağılım Fonksiyonu* $F_n(k) = \frac{1}{2^n}\sum_{i=1}^{k}\binom{n}{i}$ dir.

Binom katsayıları n sayısı büyüdükçe ($n \geq 1000$) ve k sayısı da n'nin yarısına yakınken hesaplamalarda zorluk çıkartmaktadır. Bu durumun üstesinden gelmenin çeşitli yöntemleri vardır.

- Bir yöntem ayrık binom dağılımı yerine onu büyük örnek sayılarında yakınsayan normal dağılımı kullanmak: Burada X değişkeninin kümülatif (birikmiş) olasılık dağılım fonksiyonu yerine, hesaplamalar daha uygun olması için dizinin ağırlığının, beklenen ağırlık değeri $\mu = \frac{n}{2}$ den farkı olarak tanımlanması ile ortalaması 0, standart sapması $\sigma = \frac{\sqrt{n}}{\sqrt{2}}$ olan değişken $Z = \frac{X-\mu}{\sigma} = \frac{\sqrt{2}X}{\sqrt{n}}$ alınarak standart normal dağılım kullanılır. Bu yöntemde, dizinin bir bütün olarak yapısının korunması sağlanırken, hesaplamalarda yapılan yakınsamalar tam sonuçlara ulaşılmasını engellemektedir.
- Bir başka yöntem ise indirgeme bağlantılarından yararlanmak. Uzunluğu 68 ile 4096 arasında olan ikili diziler için, $F_n(k)$ değerlerini hesaplarken $F_1(0) = 1,\ F_1(1) = \frac{1}{2} \quad n \geq 2$ ve $k = 0$

$$F_n(k) = \frac{1}{2}[F_{n-1}(k) + F_{n-1}(k-1)]$$

  indirgene bağıntısı ile ($k \geq n$ durumlarında $F_n(k) = 1$ alınarak), $n \in \{64,65, \ldots,128\}$ için hesaplamalar yapılır.
- İndirgeme bağlantılarının da yetersiz olduğu durumlarda, uzun diziden sabit uzunluklu ya da, X değişkenine bağlı olarak, değişken uzunluklu kısa diziler demeti elde edilip, demet değerlendirmeye tabi tutulur.

  - ***Sabit Uzunlukta Parçalanış***

    Örnek olarak, $n = 64$ sabit uzunluk durumunda (64 bit uzunluğun diziler üreten bir üretecin değerlendirilmesinde ya da uzun bir dizinin 64 bitlik sabit uzunluktaki kısa dizilere par-

çalanarak elde edilen demetin değerlendirilmesinde), $F_{64}(k)$ değerleri hesaplanarak Tablo 8.6'da gösterilen 8 kutu-lu χ^2 tablosu hesaplanmıştır:

Tablo 8.6. 8 Kutulu χ^2 Sonuçları

Kutu i	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Kutu Sınır Değeri T_i	0-25	26-27	28	29	30-31	32	33-34	35-64
Beklenen Olasılık Değeri E_i	0,09966	0,12153	0,08339	0,09489	0,09489	0,09462	0,15123	0,15244

Buna göre uzunluğu $n = 64$ olan bir ikili dizide 1 değerine eşit olan terimlerin sayısının $\{0,1,\dots,25\}$ kümesinde olma olasılığı $0{,}09966$, benzer şekilde $\{26,27\}$ kümesinde olma olasılığı $0{,}12153$, … $\{35,36,\dots,64\}$ kümesinde olma olasılığı ise $0{,}15244$'tür.

Bu beklenen değerler, gerçekleşen değerlerle karşılaştırılarak Ki Kare (χ^2) testi uygulanır. Sözgelimi O_1 gözlem değeri, söz konusu test edilecek olan kısa dizi demetindeki (her biri 64 bit uzunluğunda olan N tane kısa dizi) dizide ağırlığı 1. Kutu için belirlenen sınır değerleri olan $\{0,1,\dots,25\}$ kümesindeki sayılardan birine eşit çıkan kısa dizilerin sayısı yani gözlenen değeri (observed value), …, O_8'te demette, ağırlığı 8. Kutu için belirlenen $\{35,36,\dots,64\}$ kümesinin bir elemanına eşit olan kısa dizilerin sayısını göstermek üzere,

$$\chi^2 = \sum_{i=1}^{8} \frac{(O_i - E_i)^2}{E_i}$$

değeri serbestlik derecesi 7 alınarak, verilen kısa dizi demeti rastgelelik açısından değerlendirilir.

- ***Dinamik (Değişken) Uzunlukta Parçalanış***

  Bu durumda, örnek olarak $n_0 = 128$ ve $t = 280$ seçimlerine göre kullanılabilecek Ki-Kare tablosu aşağıda verilmiştir.

Tablo 8.7. Ki-Kare Sonuçları

i	1	2	3	4	5	6	7	8
Parça uzunluğu	128-238	239-245	246-250	251-255	256-260	261-266	267-274	>274
Olasılık değeri	0,13522	0,12626	0,11445	0,12404	0,12171	0,12823	0,12455	0,12549

2-Öbek Testleri:

Öbek testi bir dizideki toplam öbek sayısını ve bu öbeklerin boylarına göre dağılımını değerlendirir. Birçok test süitinde öbek testi sadece toplam öbek sayısının değerlendirilmesi şeklinde yer almıştır. NIST test süitinde ise toplam öbek testi (toplam öbek sayısı) ve 1'lerden oluşan en uzun öbek testine yer verilmiştir. Toplam öbek sayısının, beklenen değeri olan $\frac{n+1}{2}$ ye yakınlığı değerlendirilmektedir. İkinci testte ise dizide bulunan en uzun 1 ler öbeğinin, beklenen değeri ile uyumluluğu Ki-kare testi ile kontrol edilmektedir. Burada uzunlukları 10^3den 10^7 ye kadar olan diziler ele alındığı için olasılık değerleri hesaplanırken asimptotik yaklaşımlar kullanılmıştır. Ancak bu yaklaşımlar dizi uzunluğu olan n sayısının küçük olması durumunda yanıltıcı bilgiler vermektedir. Bu durumlar için tam değerlerin kullanılması gerekmektedir; ancak bunlar da hesaplaması zor olan sayılardır.

n uzunluğundaki bir dizideki k uzunluğundaki öbeklerin sayısını hesaplamak ilginç ve zor bir kombinatorik problemdir. "New statistical randomness tests based on length of runs" [11] başlıklı makalede $n = 64$, $n = 128$ ve $n = 256$ uzunluğundaki diziler için $k = 1$, $k = 2$ ve $k = 3$ uzunluğundaki öbeklerin gözükme olasılıkları hesaplanmıştır. Buna göre, örneğin $n = 64$ bit uzunluğundaki bir diziyi test etmek için belirlenen kutu değerleri Tablo 8.8'de verilmiştir:

Tablo 8.8. Farklı k Uzunluklarında Öbeklerin Gözükme Olasılıkları

	1-uzunluğundaki öbekler		**2**-uzunluğundaki öbekler		**3**-uzunluğundaki öbekler	
$n = 64$	Aralık Değerleri	Olasılık Değeri	Aralık Değerleri	Olasılık Değeri	Aralık Değerleri	Olasılık Değeri
Kutu-1	0-13	0,190082	0-5	0,161344	0-2	0,207825
Kutu-2	14-16	0,238877	6-7	0,260964	3	0,204319
Kutu-3	17-18	0,174560	8	0,149093	4	0,216732
Kutu-4	19-21	0,211470	9-10	0,245287	5-6	0,283245
Kutu-5	22-64	0,185009	11-32	0,183309	7-21	0,087877

Ancak bu makaledeki yöntemleri daha büyük n değerleri ve daha uzun k öbekleri için genelleştirmek çok kolay değildir. Bu nedenle, "R-2 composition tests: a family of statistical randomness tests for a collection of binary sequences" [12] başlıklı makalede üreteç fonksiyonlar kullanılarak $n = 4096$ uzunluğuna kadar olan diziler için

- n uzunluğundaki bir dizinin tam olarak r tane öbek içermesi olasılığı,
- n uzunluğundaki bir dizinin k uzunluğunda tam olarak a tane öbek içermesi olasılığı,
- n uzunluğundaki bir dizinin en az k uzunluğunda tam olarak a tane öbek içermesi olasılığı,
- n uzunluğundaki bir dizinin L uzunluğundan daha büyük bir öbek içermeme olasığı

hesaplanmıştır. Bu olasılıklar kullanılarak 8 tane yeni öbek testi önerilmiştir. Bu testlerin en büyük avantajı asimptotik yaklaşımlar kullanılmadığı için kısa dizileri de öbek testlerini kullanarak test edebilmesidir. Manipüle edilerek üretilmiş bazı diziler arka arkaya konularak NIST test paketindeki öbek testleri uygulandığında bu manipülasyon gözlenmemiştir. Ancak makaledeki yeni önerilen testler bunu fark etmiştir. Bu da yeni önerilen testlerin daha verimli olduğunu göstermektedir.

3-Örtüşen Kalıplar Testleri:

Bu test ailesinde belirli uzunluk ve periyottaki kalıpların dizide görülme sıklıklarına dayalı rastgele değişkenler tanımlanır. Örneğin 010 kalıbı:

- 1101100011000110 dizisinde 0 kere
- 1000110010010001 dizisinde 2 kere
- 010101111010100 dizisinde ise 4 kere geçmektedir.

Bu testte belirlenen bir kalıbın n uzunluğundaki bir dizide k kere geçme olasılığı kullanılır ve bu olasılıklar yardımıyla bir istatistiksel rastgelelik testi tanımlanır. NIST test paketinde Örtüşen Kalıplar Testi (Overlapping Template Matching Test) ve Örtüşmeyen Kalıplar Testi (Nonoverlapping Template Matching Test) yer almaktadır.

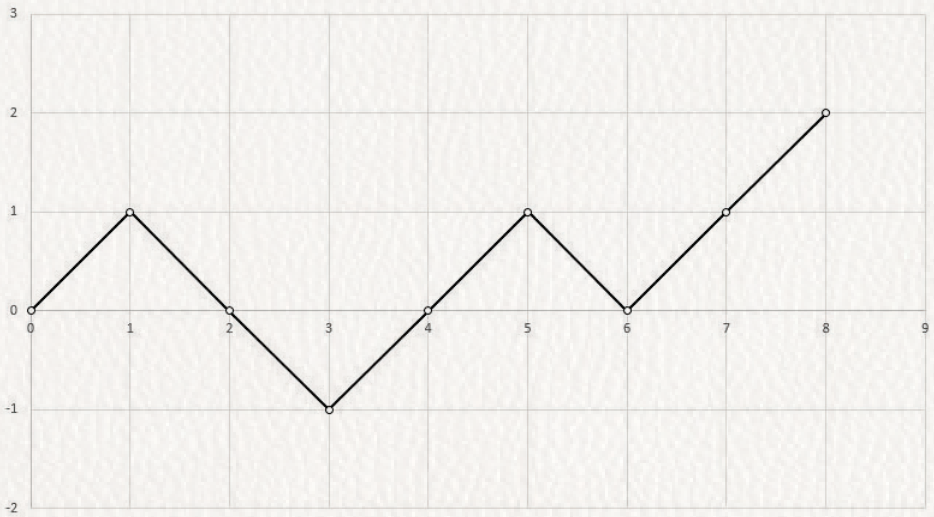
Yakın zamanda Sulak vd. "Periodic template tests: A family of statistical randomness tests for a collection of binary sequences" [13] başlıklı makalede belirtilen olasılığın kalıbın periyoduna bağlı olduğunu göstermişlerdir. NIST paketindeki Örtüşen Kalıplar Testi periyodu 1 olan 9 uzunluğundaki kalıpları (B=111111111) sayarken Örtüşmeyen Kalıplar Testinde periyodu 9 olan tüm kalıplar sayılmakta ve her birisi için ayrı bir p −değeri üretilmektedir. Farklı bir periyoda sahip bir kalıp için olasılık değerleri NIST paketinde bulunmadığı için bu kalıplarla test yapmak mümkün olmamaktadır. Ancak [13] da tüm periyotlar ve dolayısıyla tüm kalıplar için bu problem çözülmüştür.

4-Rastgele Yürüyüş Testleri:

Bu testte söz konusu ikili $\{a_i\}_{i=1}^{n}$ dizisi önce $\{(-1)^{a_i}\}_{i=1}^{n}$ olarak işaret dizisine dönüştürülür. Daha sonra, koordinat düzleminde $(0,0)$ orijin noktasından başlanarak $(1,(-1)^{a_1})$ noktası doğru parçası ile birleştirilir. Ardından bu nokta $(2,(-1)^{a_1}+(-1)^{a_2})$ noktası ile doğru parçası ile birleştirilerek devam edilip, son olarak da $(n-1,(-1)^{a_1}+\cdots+(-1)^{a_{n-1}})$ noktası $(n,(-1)^{a_1}+\cdots+(-1)^{a_{n-1}}+(-1)^{a_n})$ noktası ile doğru parçası ile birleştirilerek, $\{a_i\}_{i=1}^{n}$ dizisine karşılık gelen, kesik çizgilerden oluşan sürekli fonksiyonun grafiği elde edilir. Bu fikri daha netleştirme adına bir örnek

verelim: $\{a_i\}_{i=1}^{8} = 0,1,1,0,0,1,0,0$ dizisinden elde edilen $\{(-1)^{a_i}\}_{i=1}^{8} = 1,-1,-1,1,1,-1,1,1$ işaret dizisine karşılık elde edilen grafik aşağıda gösterilmiştir.

Tablo 8.9. Rastgele Yürüyüş Testi



Bu aşamadan sonra söz konusu dizinin rastgelelik bakımından incelenmesi adına birden çok rastgele değişken tanımlanabilir. Örneğin, $X_t(n,k)$ kümesi $n$uzunluğunda olan tüm ikili dizilerden $y = t$ doğrusunu tam olarak k kere kesenlerin kümesi olarak tanımlanır ve $p_t(n,k)$ ile n uzunluğundaki bir dizinin $X_t(n,k)$ kümesinin elemanı olma olasılığı gösterilerek, verilen n uzunluğunda bir dizinin, $y = t$ doğrusunu kaç kere kestiğini sayan X rastgele değişkeni kullanılarak, rastgele bir diziden ayırt edilip edilemeyeceği hakkında karar verilir. Benzer çalışmalar, dengeli diziler alt kümesinde de yapılabilir. Başka bir rastgele değişken, diziye karşılık çizilen grafiğin ulaştığı en yüksek değer kullanılarak da tanımlanabilir. Tekrar etmekte fayda var; burada en önemli nokta tanımlanan rastgele değişkenler için olasılık dağılım fonksiyonlarının hesaplanabilir olmasıdır. Örnek olarak, AES algoritmasının çıktılarını Rastgele Yürüyüş Testlerinden birisinin kullanarak değerlendirilmesini şu şekilde yapabiliriz. 0 dan 100.000'e kadar sayılara karşılık gelen açık metinleri, anahtarı 0 alarak, AES algoritması ile şifreleyerek elde edilen

$N = 100.000$ tane 128 bit uzunluğunda dizilerden oluşan demeti ele alalım. Demetteki her bir dizi, $y = 2$ doğrusunu kesmesi açısından test edildiğinde, indirgeme bağıntıları kullanılarak elde edilen beklenen değerler ve gözlenen değerler Tablo 8.9'da verilmiştir.

Tablo 8.10. Ki-Kare Çizelgesi

	1.Kutu	**2.Kutu**	**3.Kutu**	**4.Kutu**	**5.Kutu**	**6.Kutu**	**7.Kutu**	**8.Kutu**
	0,…,11	12,13	14	15	16	17,18	19,20	21,…,128
Beklenen	10517,1	14153,2	9444,8	10377,2	10602,3	19119,4	13467,7	12318,3
Gözlemlenen	10541	14089	9456	10282	10634	19089	13575	12334

Tablo 8.9'da verilen Ki-Kare test sonuçları kullanılarak

$$X^2 = \sum \frac{(E_i - O_i)^2}{E_i} = \sum \frac{(Np_i - O_i)^2}{Np_i} = 2{,}2682$$

ve dolayısıyla p değeri

$$p = igam\left(\frac{8-1}{2}, \frac{2{,}2682}{2}\right) = 0{,}9435$$

elde edilir. Bu değer ile, belirlenecek olan eşik değer ile karşılaştırılarak, alınacak karar AES algoritması çıktısının, söz konusu rastgele değişken açısından, bir zayıflığının tespit edilip edilemediği hakkında, başka bir deyişle H_0hipotezi kabul edilip edilemeyeceği hakkında olacaktır.

5-Bazı Diğer Testler:

Örtüşen Kalıplar Testleri: Belirli uzunluk ve periyottaki kalıpların dizide görülme sıklıklarına dayalı rastgele değişkenler tanımlanır. Kalıpların frekansı incelenirken dizinin örtüşen (overlapping) parçalanışları üzerinden frekans hesabı yapılır.

- **Ayrık Kalıp Tam Sayı Testleri:** Dizinin uzunluğuna göre belirlenmiş bir kalıp uzunluğunda örtüşmeyen (non-overlapping) kalıplara

ayrılır. İkilik tabandaki her parçanın 10'luk tabandaki tam sayı değerleri hesaplanarak dizi tam sayı dizisine dönüştürülür. Elde edilen bu diziye

- Tam sayı maksimum testi (kalıptaki en büyük sayının k veya daha küçük olması),
- Tam sayı minimum testi (kalıptaki en küçük sayının k veya daha küçük olması),
- Tam sayı fark testi (kalıptaki en büyük ve en küçük sayı arasındaki fark),
- Tam sayı kapsam testi (kalıpta kaç tane farklı tam sayı bulunduğu),
- Frekans testi (her bir tam sayının kaç kere görüldüğü),
- Tekerrür noktası testi (herhangi bir elemanın ikinci kez ve geri kalan tüm elemanların sadece bir kez görüldüğü en kısa alt dizinin uzunluğu),
- Doyma noktası testi (elemanların tamamının ilk görüldüğü indeks),

- **Doğrusal Karmaşıklık Testleri:** Bir dizinin doğrusal karmaşıklığı bu diziyi üretebilen en kısa LFSR'ın uzunluğu olarak tanımlanır. Dizinin doğrusal karmaşıklığı Berlekamp-Massey Algoritması ile hesaplanır. Uzunluğu n olan bir dizinin karmaşıklığının k veya daha küçük olma olasılığı hesaplanarak dizi değerlendirilir. Başka bir değerlendirme şekli de dizinin Doğrusal Karmaşıklık profilindeki eleman sayısını gözetir.
- **İndeks-değer çakışması testi:** indeks-değer çakışmasının rastgele bir dizide hangi olasılıkla görüleceği değerine göre dizi değerlendirilir.

gibi değerleri ölçen rastgele değişkenler kullanılarak dizi veya demetler, rastgele dizi veya demetlerden ayırt edilip edilemeyeceği açışından değerlendirilir.

8.3. SONUÇ VE DEĞERLENDİRMELER

TC Kimlik Numaralarının belirlenmesinden asal sayıların üretilmesine, şans oyunlarından, bir seçimde veya sınavda kullanılacak aday numaralarının belirlenmesine, istatistikte örneklemeler yapmadan bilgisayar oyunlarına, simülasyonlardan şifrelemeye, askeri uygulamalardan milli güvenliği gerektiren konulara birçok alanda ihtiyaç duyulan rastgele sayılar genelde sözde rastgele sayı üreteçleri tarafından üretildiğinden rastgele sayıları üretmek, kullanmak veya uygulamak en önemli konular arasındadır.

Bu kitap bölümünde;

- Rastgele olmaktan uzak, tamamen kararlı bir algoritma ile üretilen sayı dizilerinin, rastgele elde edilmiş dizilerden ayırt edilip edilemeyeceğine, değişik bakış açıları ile karar vermeye çalışmak amacıyla yapılan istatistiki testler tanıtılarak, bu konunun önemi gösterilmiştir.
- Literatürde var olan birçok testin dizileri hangi açılardan ve nasıl değerlendirdiği, bir testin tanımlanmasında karşılaşılabilecek teorik ve pratik hesaplama problemler açıklanmıştır.
- Yeni bir test tanımlarken nelere dikkat edilmesi gerektiğini, testin pakete katkısının nasıl hesaplanabileceği anlatılmıştır.
- Bir test paketinin kapsam bakımından geniş olması gerektiğini ancak bu özelliği sağlarken gereksiz yere hantal-yavaş olmasından da kaçınılması gerektiğine dikkat çekilmiştir.
- Testlerin, bazı özelliklerine göre, nasıl sınıflandırabileceği gösterilmiştir

Günümüzde de yeni testler tanımlanmaya devam etmektedir. Teorik olarak tanımlanan bir testin matematiksel bir değeri olsa da, pratikte de kullanılabilir olması, söz konusu rastgele değişkene ait olasılık dağılım fonksiyonunun hem teoride hem de pratikte hesaplanabilir olması önemlidir.

Tanımlanacak yeni bir testin kıymeti, kullanılacağı test paketine katkısıyla, başka bir değişle mevcut testlerle korelasyonuna bağlıdır. Testin tanımında kullanılan matematiksel hesapların tam olup olmadığı, gerekli hesaplamaları sonuçlandırma adına bazı yaklaşımlar kullanıldı ise bu yaklaşımlardaki hata oranının hangi parametre değerleri için (örneğin hangi uzunluktaki diziler için) kabul edilebilir sınırlar içinde olduğunun tespiti büyük önem taşır.

Yeni bir test tanımı yapabilmek için en başta özellikle sonlu matematik alanında yetkin, istatistik biliminin temellerine hâkim ve bilgisayar programlama konusunda da etkin olmak gerekmektedir. Bu özellikler bazen bir kişide toplanmayabilir. Bu durumda grup çalışması büyük önem taşımaktadır. Rastsallık testleri konusunda kendini geliştirmek isteyen bir kişinin en başta olasılık dağılım fonksiyonları, hipotez testlerinin nasıl yapıldığı konusunda geliştirmesi ve ardından NIST'in (National Institute of Standards and Technology) İstatistiksel Test Paketi gibi çalışmaları okuması, anlaması ve yorumlayabilmesi önerilir.

NIST paketi değerlendirildiğinde, temel kavramların ve testlerin verildiği fakat yeterli olmaktan uzaktır. Kaynaklar kısmında verilen makalelerde, söz konusu pakette bulunan bazı testlerdeki hesaplamalar geliştirilmiş, düzeltilmiş, olmayan bazı yeni testler eklenmiştir.

Sonuç olarak, bilimsel ve sektörel pekçok çalışmanın altlığını oluşturun rasgele sayı üretiminin, gelişmeye açık, güncel bir araştırma konusu olarak öne çıktığı, bu konuya daha çok önem verilmesi ve yeni çalışmaların yapılması gereklidir.

Teşekkür

Bu bölümün hazırlanmasında kıymetli desteklerini esirgemeyen Sn. Doç. Dr. Ali Doğanaksoy'a ve Sn. Doç. Dr. Fatih Sulak'a sonsuz teşekkürlerimi sunarım.

KAYNAKLAR

[1]. S. W. "Golomb, Shift Register Sequences", Aegean Park Press, Laguna Hills, CA,USA, 1982, ISBN 978-0894120480.

[2]. Z. Akcengiz, M. Aslan, Ö. Karabayır, A. Doğanaksoy, M. Uğuz, F. Sulak, "Statistical Randomness Tests of Long Sequences by Dynamic Partitioning", IEEE, 2020 International Conference on Information Security and Cryptology (ISCTURKEY), **DOI:** 10.1109/ISCTURKEY51113.2020.9308005

[3]. D. E. Knuth, "The Art of Computer Programming", Vol.2 (3rd Ed.): Seminumerical Algorithms, Addison-Wesley Longman Publishing Co., Inc., Boston,MA, USA, 1997, ISBN 0-201-89684-2.

[4]. A. Ruhkin, "Testing randomness": A Suitee of Statistical Procedures, Theory of Probability & Its Applications, 45(1), pp. 111–132, 2001. doi.org/10.1137/S0040585X97978087

[5]. G. Marsaglia, "The marsaglia random number" CDROM including the diehard battery of tests of randomness, http://www.stat.fsu.edu/pub/diehard/, 1995.

[6]. D. Eddelbuettel, R. G. Brown, "DIEHARDER": an r interface to the dieharder Suitee of random number generator tests, 2007.

[7]. W. Caelli, "Crypt x package documentation", Technical Report, Information Security Research, 1992.

[8]. P. L'Ecuyer and R. Simard, "Testu01: A C library for empirical testing of randomnumber generators", ACM Trans. Math. Softw., 33(4), August 2007, ISSN 0098-3500.

[9]. L. E. Bassham, III, A. L. Rukhin, J. Soto, J. R. Nechvatal, M. E. Smid, E. B. Barker, S. D. Leigh, M. Levenson, M. Vangel, D. L. Banks, N. A. Heckert, J. F.Dray, and S. Vo, Sp 800-22 rev. 1a. "a statistical test Suitee for random and pseudorandom number generators for cryptographic applications", Technical report, NIST, Gaithersburg, MD, United States, 2010.

[10]. A. Doğanaksoy, F. Sulak, M. Uğuz, O. Şeker, Z. Akcengiz, "Mutual correlation of NIST statistical randomness tests and comparison of their sensitivities on transformed sequences" ,Turkish Journal of Electrical Engineering & Computer Sciences (2017) 25: 655-665 doi:10.3906/elk-1503-214

[11]. A. Doğanaksoy, F. Sulak, M. Uğuz, O. Şeker, Z. Akcengiz, " New Statistical Randomness Tests Based on Length of Runs", Hindawi Publishing Corporation Mathematical Problems in Engineering, Volume 2015, Article ID 626408, 14 pages http://dx.doi.org/10.1155/2015/626408

[12]. M. Uğuz, A. Doğanaksoy, F. Sulak, O. Koçak, "R-2 composition tests: a family of statistical randomness tests for a collection of binary sequences", Cryptography and Communications (2019) 11:921–949 https://doi.org/10.1007/s12095-018-0334-1

[13]. F. Sulak, A. Doğanaksoy, M. Uğuz, O. Koçak, "Periodic template tests: A family of statistical randomness tests for a collection of binary sequences", Discrete Applied Mathematics, Volume 271, 1 December 2019, Pages 191-204

Bölüm 9

SİMETRİK SİSTEMLERDE KRİPTOANALİZ YÖNTEMLERİ

Mehmet Emin Gönen, Orhun Kara, Ferhat Karakoç

Bu bölümde simetrik şifreleme algoritmalarının iki kategorisinden birisi olan blok şifreleme algoritmaları ile diğeri olan dizi şifreleme algoritmala-rına yapılan belli başlı saldırılar tanıtılmakta ve bu saldırılar ile ilgili literatürdeki gelişmeler aktarılmaktadır. Blok şifreleme algoritmaları ve dizi şifreleme algoritmaları hakkında özet bilgi verildikten sonra böl ve fethet saldırıları, tahmin et ve belirle saldırıları, ödünleşim (trade-off) saldırıları gibi genel saldırı yöntemleri anlatılmıştır. Ardından doğrusal kriptoanalizden, farksal kriptoanaliz ve türevlerinden, imkansız farksal kriptoanalizden, bumerang ve dikdörtgen saldırılarından, kare ve integral kriptoanalizden, ortada buluşma saldırılarından ve cebirsel saldırılardan bahsedilmiştir. Ayrıca, kriptoanalizlerin anlatımlarından önce hem kriptoanalizlerde örnek gösterilmesi amacıyla hem de en çok bilinen şifreleme algoritmaları oldukları için DES ve AES blok şifreleme algoritmaları özet olarak anlatılmıştır. Genel olarak bu bahsi geçen saldırıların alt yapıları ve temel teorileri ile birlikte literatürdeki güncel uygulamalar aktarılmıştır.

9.1. GİRİŞ

Tek bir anahtar ile hem şifreleme ve hem de şifre çözme işlemlerinin yapıldığı kriptografik algoritmalara simetrik şifreleme algoritmaları denir. Simetrik şifreleme algoritmaları kendi içerisinde blok şifreleme ve dizi şifreleme algoritmaları olmak üzere iki bölüme ayrılırlar.

Blok şifreleme algoritmaları her bir anahtar için büyük boyutlarda (örneğin 8 Bayt ya da 16 Bayt) sözde-rastsal permütasyon (pseudorandom permutation) üreten algoritmalardır. Blok şifreleme algoritmalarını diğer kriptografik yapıtaşlarından ayıran en belirgin özellik, her bir anahtar için üretilen permütasyonun tersinin olmasıdır. Bu permütasyonlar değişik çalışma kipleriyle çok farklı amaçlarla, hatta dizi şifreleme algoritmaları olarak da kullanılabilirler.

Dizi şifreleme algoritmaları ise, kayan anahtar üreteci ile ilklendirme (initial) algoritmasından oluşur ve açık metinden bağımsız olarak, zamana bağlı hafızalı kayan anahtar dizisi üretirler. Bu anahtar dizisi, açık metnin şifrelenmesinde kullanılır. İlklendirme algoritması gizli anahtar ile BİKSA (Bir kullanımlık Sayı- Nonce: Number Used Once) özelliğini sağlayan BV (Başlangıç Vektörü - Initial Vector) değerinden, tohum (seed) adı verilen ilk içsel durumu sağlayan sayıyı üretir. İlk içsel duruma tohum adının verilmesinin nedeni, geri kalan bütün içsel durumların ve kayan anahtar dizisinin ilk içsel durumdan üretiliyor olmasıdır. Kayan anahtar üreteci, tohumdan başlayarak bir yandan güncelleme fonksiyonu ile içsel durumları güncellerken diğer taraftan çıktı fonksiyonu ile güncel içsel durumdan kayan anahtar dizisinin o anki çıktısını üretir. Zamana bağlı bir anahtar dizisi üretimi söz konusudur. Üretilen anahtar dizisi açık metnin bit uzunluğu kadardır. Şifreleme işlemi ise anahtar dizisi ile açık metnin bit bazında YaDa (XOR) işlemine tabi tutulmasıdır.

Bir simetrik şifreleme algoritmasının matematiksel yapısının analiz edilerek, herhangi bir anahtarın oluşturduğu permütasyonun rasgele bir permütasyondan (blok şifrelemelerde) ya da fonksiyonun rasgele bir fonksiyondan ayırt

edilebilmesi (dizi şifrelemelerde) faaliyetlerine bu şifreleme algoritmasının kriptoanalizi denir. Literatürde bu analiz yöntemleri genellikle algoritmada kullanılan gizli anahtarın ele geçirilmesini hedefler. Gizli anahtarın ele geçirilebilmesi için bu anahtarla üretilmiş açık-metin/şifreli metin çiftleri kullanılır. Bu çiftler kriptoanaliz algoritmasının girdilerini oluşturur ve çiftlerin sayısı veri karmaşıklığını ifade eder.

Bu bölümde literatürde sıklıkla uygulanan en yaygın kriptoanaliz yöntemlerinden örnekler anlatılacaktır. Öncesinde blok şifreleme algoritmaları ve dizi şifreleme algoritmaları hakkında genel bilgi verilecek ve en bilinen blok şifreleme algoritmaları olan DES ve AES'in genel yapısı anlatılacaktır. Blok şifreleme algoritmalarına uygulanan belli başlı saldırılardan farksal kriptoanaliz, doğrusal kriptoanaliz, imkânsız farksal kriptoanaliz, bumerang saldırısı, integral kriptoanaliz, ortada buluşma saldırıları ve cebirsel saldırıya yer verilmiştir. Bu saldırılara geçmeden önce, genel saldırılardan olan böl ve fethet saldırıları, dizi şifreleme algoritmalarına uygulanan tahmin et ve belirle saldırıları ve ödünleşim saldırıları anlatılmıştır.

Bu bölüm simetrik şifreleme algoritmalarına uygulanan kriptoanaliz tekniklerinin hepsini kapsamamaktadır. Var olan tüm saldırı yöntemlerini tek bir bölüme sığdırmak mümkün değildir. Bu nedenle literatürde en çok uygulanan belli başlı saldırılara yer verilmiştir. Bu saldırılar dışında Biham tarafından tanıtılan ilintili anahtar saldırıları (related key attacks) [17], Tezcan tarafından geliştirilen beklenmedik farksal saldırı (improbable differential attack) [108], Jacobsen ve Knudsen tarafından geliştirilen interpolasyon saldırısı (interpolation attack) [61], Wagner ve Biryukov tarafından geliştirilen sürgü saldırısı (slide attack) [124], Kara tarafından geliştirilen yansıtma saldırısı (reflection attack) [63], Khovratovich ve arkadaşları tarafından geliştirilen bi-klik saldırısı [67], Dinur ve Shamir tarafından geliştirilen küp saldırısı [48] gibi saldırılar da mevcuttur.

Bu bölümde yer alan kriptoanalitik teknikler, yer sıkıntısından dolayı kısa kısa anlatılmıştır. Ayrıca öncesinde blok şifreleme algoritmaları ve dizi şifreleme algoritmalarıyla ilgili kısa ve öz olacak şekilde temel bilgiler verilmiştir. An-

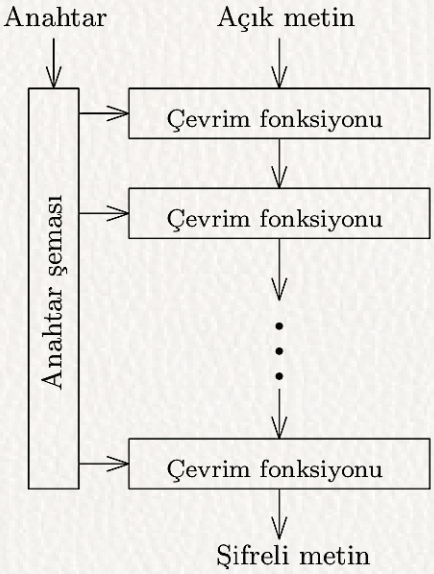
cak bu bilgiler analiz metotlarını kavramak açısından yetersiz kalabilir. Böyle bir durumda; öncesinde blok şifreleme algoritmaları, dizi şifreleme algoritmaları, bu algoritmalarının yapı taşları ve yapı taşlarının güvenlik ölçütleri ilgili daha geniş kaynaklara başvurulabilir. Örneğin, bu kitap serisinin ikincisinde yer alan ve Tolga Sakallı tarafından kaleme alınan "kriptografik test yöntemleri ve kriptoanaliz" başlıklı bölüm okunabilir [103]. İlgili bölümde, blok şifreleme algoritmalarının ve dizi şifreleme algoritmalarının ayrıntılı bir şekilde anlatılmasıyla birlikte, bu algoritmalarının yapı taşlarına da geniş yer verilmiştir. Boole fonksiyonları, bu fonksiyonların cebirsel normları, doğrusalsızlık hesapları, doğrusal saldırıya ve farksal saldırıya karşı bu yapıların sağlaması gereken ölçütler ayrıntılı bir şekilde ve örneklerle ilgili bölümde yer almaktadır. Belli başlı algoritmaların S kutuları ve bu kutuların fark tabloları, doğrusalsızlıkları gibi kriptografik özellikleri, blok şifreleme algoritmalarının katman yapıları için de [103] nolu kitap bölümüne başvurulabilir.

9.2. BLOK ŞİFRELEME ALGORİTMALARI

Sabit uzunlukta açık metin ve sabit uzunlukta şifreleme anahtarını girdi olarak alıp, anahtara bağlı tersinir bir permütasyon kullanarak açık metin ile aynı uzunlukta şifreli metin üreten simetrik şifreleme algoritmalarına blok şifreleme algoritmaları denir. Bir blok şifreleme algoritması, belirli bir giz (çevrim anahtarı) kullanılarak oluşturulmuş ve kriptografik olarak son derece zayıf olan bir permütasyonun defalarca uygulanmasıyla elde edilir. Bu zayıf permütasyona çevrim fonksiyonu adı verilir. Permütasyonu belirleyen gizeme ise çevrim anahtarı denir. Ayrıca permütasyonun uygulanma sayısına çevrim sayısı denir.

Notasyonu belirlemek amacıyla, E blok şifreleme algoritması olmak üzere; $E_K(P) = C$ eşitliği, P açık metin bloğunun K anahtarı ile şifrelenmesiyle C şifreli metin bloğunun üretilmesini ifade eder. Benzer şekilde $D_K(C) = P$ eşitliği ise şifre çözme algoritmasını temsil eder. Dolayısıyla, D_K permütasyonu E_K permütasyonunun tersidir: n blok bit uzunluğu ve k anahtar bit uzunluğu olmak üzere $D_K\big(E_K(x)\big) = E_K\big(D_K(x)\big) = x, \forall x \in GF(2)^n$ ve $\forall K \in GF(2)^k$.

Algoritmayı oluşturan çevrim fonksiyonu, çevrim anahtarını ve bir önceki çevrim fonksiyonunun çıktısını girdi olarak alıp çevrim çıktısını üretir. İlk çevrim numarası 1 olmak üzere i. çevrim fonksiyonu i. çevrim anahtarı RK_i ile, $(i-1)$. çevrim çıktısı R_{i-1} 'den bir sonraki çevrim çıktısı olan R_i değerini üretir: F çevrim fonksiyonu olmak üzere $F_{RK_i}(R_{i-1}) = R_i$. Burada, r çevrim bir blok şifreleme algoritmasında $P = R_0$ ve $C = R_r$ olacaktır. Her bir RK_i çevrim anahtarı, K ana anahtarından A anahtar şeması algoritması kullanılarak $A(K, i) = RK_i$ eşitliğiyle üretilir. Bazı algoritmalarda ilk çevrim fonksiyonu çalıştırılmadan önce ve/veya son çevrim fonksiyonu çalıştırıldıktan sonra aklama anahtarlarının (whitening key) da kullanıldığı görülmektedir. Bu anahtarlar sırasıyla RK_0 ve RK_{r+1} ile temsil edilecektir. Anahtar şeması hafifsıklet (lightweight) blok şifreleme algoritmalarında son derece basit olmakla birlikte bazı algoritmalarda tek yönlü fonksiyon özelliğine sahip olacak kadar karmaşık olabilmektedir. Bir blok şifreleme algoritmasının genel yapısı Şekil 9.1'de gösterilmiştir.



Şekil 9.1. Bir blok şifreleme algoritmasının genel yapısı

Yıllar içinde literatürdeki örnekler çevrim sayısı ile güvenlik arasında net bir korelasyon olduğunu göstermiştir. Gerçekten de birkaç istisnai örnek algoritma hariç, blok şifreleme algoritmalarının büyük çoğunluğunda çevrim sayısının artmasıyla, bilinen bütün kriptoanaliz yöntemlerine karşı güvenliğin de arttığı gözlenmektedir. Blok şifreleme algoritmalarına yapılan saldırıların hemen hepsi, genel olarak blok şifreleme permütasyonunu rastsal permütasyondan ayırt etmeye yönelik olsa da, bu saldırıların ana hedefi belirli bir ya da birkaç çevrim anahtarını, oradan da ana anahtarı ele geçirmektir.

Blok şifreleme algoritması yukarıda tanımlandığı üzere sabit uzunlukta açık metinleri şifrelemektedir. Uzun bir metin şifrelenmesi için şifrelenecek metin algoritmanın blok uzunluğundaki parçalara ayrılır ve her bir parça ayrı ayrı şifrelenir. Bu şifreleme yöntemine Elektronik Kod Defteri ("Electronic codebook") kipi adı verilir. Bu yöntem ile şifreleme yapılması durumunda aynı açık metin blokları için aynı şifreli metin blokları oluşur. Bunu önlemek için farklı blok şifreleme kipleri tanıtılmıştır. Bunlardan bazıları Şifre-bloğu zincirleme ("Cipher-block chaining"), Şifre-bloğu geribesleme ("Cipher feedback") ve Çıktı geribesleme ("Output feedback") kipleridir.

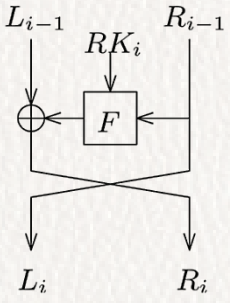
Çevrim fonksiyonunu, çevrim anahtarı ekleme, çevrim girdisi ve çevrim anahtar bitlerinin yayılımı sağlayan doğrusal fonksiyon olan yapı taşları ve çevrim girdisi ve çevrim anahtarı bitlerinin doğrusal olmayan bir yöntemle işleme girmesini sağlayan yapı taşları oluşturmaktadır. Burada kullanılan doğrusallık özelliği, bir f fonksiyonu için $f(x \oplus y) = f(x) \oplus f(y)$ özelliğinin sağlanması olarak tanımlanır. Bu işlemler, çevrim fonksiyonu ardışık olarak çalıştırılarak her bir şifreli metin biti için açık metnin her bir biti ve anahtarın her bir bitine bağlı olarak karmaşık bir denklem elde edilir. Böylece blok şifreleme algoritması sayesinde karmaşık bir denklem sistemi elde edilmiş olur.

Bir blok şifreleme algoritması tasarlanırken genellikle aşağıdaki iki yöntemden biri tercih edilebilir:

Feistel Ağı Yapısı. Bu tasarım yönteminde, çevrim girdisi iki parçaya ayrılır ve iki parçadan oluşan çevrim çıktısı elde edilir. i. çevrim için çevrim girdisi parçaları L_i ve R_i, çevrim çıktısı parçaları L_{i+1} ve R_{i+1} ve çevrimde kullanılan fonksiyon da F ile gösterildiğinde çevrim çıktısı aşağıdaki işlem ile hesaplanır:

$$(L_i, R_i) = (R_{i-1}, L_{i-1} \oplus F_{RK_i}(R_{i-1}))$$

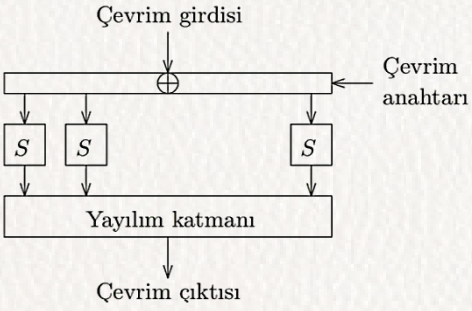
Bu yöntem Şekil 9.2'de gösterilmiştir. Feistel ağ yapısının temel özelliği çevrim çıktısındaki bazı bit gruplarının çevrim girdisindeki bazı bit grupları ile aynı olmasıdır. Feistel ağ yapısının tercih edilmesinin temel nedenlerinden bir tanesi, çevrim anahtarlarının ters sırada kullanılması durumunda yapılan işlemin tersinin kendisine eşit olmasıdır. Çevrim fonksiyon girdisinin iki yerine daha fazla parçalara ayrılmasıyla ve bu parçalardan sadece birine veya birkaçına F fonksiyonu uygulanarak ve diğer parçalar ile YaDa işlemleri uygulanarak oluşan yapıya ise genelleştirilmiş Feistel ağ yapısı adı verilir.



Şekil 9.2. Feistel Yapının i. Çevrimi

YPA (Yerini Alma – Permütasyon Ağı- Substitution Permutation Network) yapısı. Bu yapıda, Feistel ağ yapısının tersine, çevrim çıktısındaki hiçbir bit çevrim girdisindeki hiç bir bite eşit olarak tanımlanmayıp, tüm bitler güncellenir. Şekil 9.3'te bu yapının genel bir tasarımı verilmiştir. Çevrim fonksiyonunun yapı taşlarından bir tanesi çevrim anahtarının kullanımıdır. Bu kullanım genellikle YaDa işlemi ile yapılır. Diğer bir yapı taşı ise doğrusal olma-

yan ve bitler arasında karmaşık denklemlerin oluşmasını sağlayan "yerini alma kutuları"dır (substitution box, s-box). Bitlerin yayılımını sağlamak, bitler arasındaki etkileşimi artırmak için ise doğrusal özelliği sağlayan bir yapı taşı "yayılım katmanı" (diffusion layer) kullanılır. Bu yapı taşları uygulanarak çevrim çıktısı elde edilir. Feistel ağ yapısında kullanılan F fonksiyonu da aslında kendi içinde bir YPA yapısıdır, F fonksiyonu da çevrim anahtarı ekleme, "yerini alma kutuları" ve "yayılım katmanı" yapıtaşlarından oluşur.



Şekil 9.3. YPA Yapının Bir Çevrimi

9.3. DES VE AES BLOK ŞİFRELEME ALGORİTMALARI

9.3.1 DES (Data Encryption Standard)

1972 yılında, o zamanki adıyla NBS (National Bureau of Standards) şimdiki adıyla NIST(National Institute of Standards and Technology) kurumu tarafından, ABD için bir standart şifreleme algoritması tasarlanması talebinde bulunuldu. IBM tarafından 1974 yılında tasarlanan LUCIFER algoritması, NSA (National Security Agency) tarafından bazı değişiklikler yapılarak 1977'de DES adıyla ABD'nin resmi şifreleme algoritması ilan edildi. NSA tarafından yapılan değişiklikler bir dizi spekülasyonları da beraberinde getirdi. En çok tartışılan konu ise anahtar boyunun 56 bit gibi kısa bir değer olmasıydı. 1980'lerin başında algoritmanın tüm yapıtaşlarının literatüre açık hale gelmesi DES üzerinde kriptoanaliz çalışmalarını hızlandırdı. Nitekim

1990 yılında Eli Biham ve Adi Shamir tarafından [19] farksal atak kullanılarak 15 çevriminin kaba kuvvet saldırısından daha iyi bir karmaşıklıkla kırılabileceği gösterildi. Sonrasında bu atağın 16 çevrime genişletilebileceği gösterildi. Ayrıca 1993 yılında Matsui [83] tarafından bulunan doğrusal atak ile DES algoritmasının tüm çevriminin kaba kuvvet saldırısından daha iyi bir karmaşıklıkla kırılabileceği gösterildi. DES'in zayıflıklarının ortaya çıkması sonucu, yeni bir standart algoritma arayışına girildi. NIST tarafından yapılan yarışma sonucunda Rijndeal algoritması 2000 yılında AES standart şifreleme algoritması olarak seçildi.

DES'in Genel Yapısı DES, genel yapı itibari ile *Feistel Ağı* yapısı kullanan 64 bit blok boyu, 56 bit anahtar boyuna sahip bir blok şifreleme algoritmasıdır. Feistel yapısının özelliğinden dolayı açık metin 32 bitlik iki kola ayrılarak çevrim fonksiyonu uygulanmaktadır. DES algoritması, **karıştırma** (confusion) ve **yayılım** (diffusion) katmanları içerecek şekilde tasarlanmıştır. Karıştırma katmanı ile hedeflenen, anahtar ve şifreli metin arasındaki ilişkiyi karmaşıklaştırmaktır. Bunun için doğrusal olmayan S-kutuları kullanılmıştır. Yayılım katmanı ile de, açık metindeki ve anahtardaki her bir bitin şifreli metindeki tüm bitleri birkaç çevrim sonunda etkilemesi hedeflenmiştir. DES algoritmasında bunun için kullanılan yapı ise bit permütasyonudur. Birbirini tekrar eden 16 çevrimden oluşan algoritmanın genel yapısı Şekil 9.4'te gösterilmiştir.

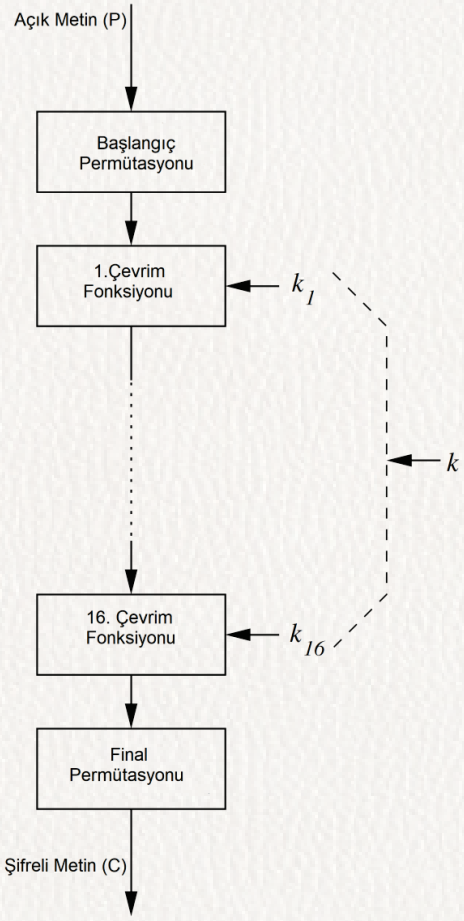
Algoritmanın her çevriminde, anahtar şeması kullanılarak ana anahtar k'dan üretilen çevrim anahtarları $k_1,\ k_2, \cdots, k_{16}$ YaDa işlemi ile eklenir.

Başlangıç permütasyonunu IP ile gösterirsek, 64 bitlik açık metin IP'den geçirildikten sonra 32 bitlik iki parçaya ayrılır. İlk parçalara L_0 ve R_0 diyelim. 16 çevrim boyunca F çevrim fonksiyonu bu parçalara tekrarlı (iterative) şekilde $i = 1, \cdots, 16$ olmak üzere aşağıdaki gibi uygulanır:

$$L_i = R_{i-1},$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, k_i)$$

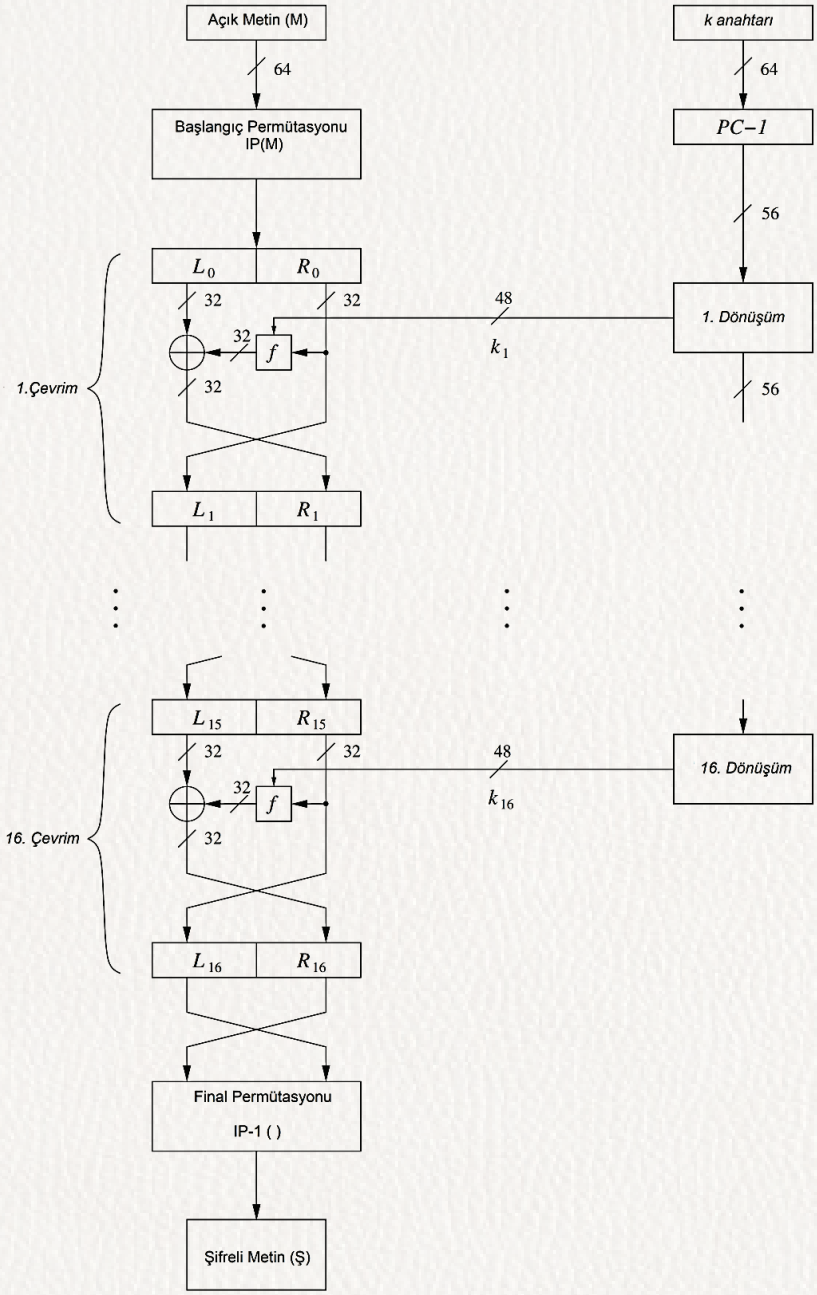
L_{16}, R_{16} parçalarının final permütasyonu olan IP^{-1}'den geçirilmesiyle şifreli metin elde edilir. Donanımda gerçekleme kolaylığı sağlayan başlangıç ve final permütasyonlarının algoritmanın güvenliğine bir katkısı bulunmamaktadır. DES şifre çözme işlemi ise Feistel yapısının özelliğinden dolayı şifrelemenin aynısıdır. Sadece çevrim anahtarlarının ters sıra ile kullanılması gerekmektedir. Algoritmanın ve anahtar şemasının ayrıntılı gösterimi Şekil 9.5'te verilmiştir.



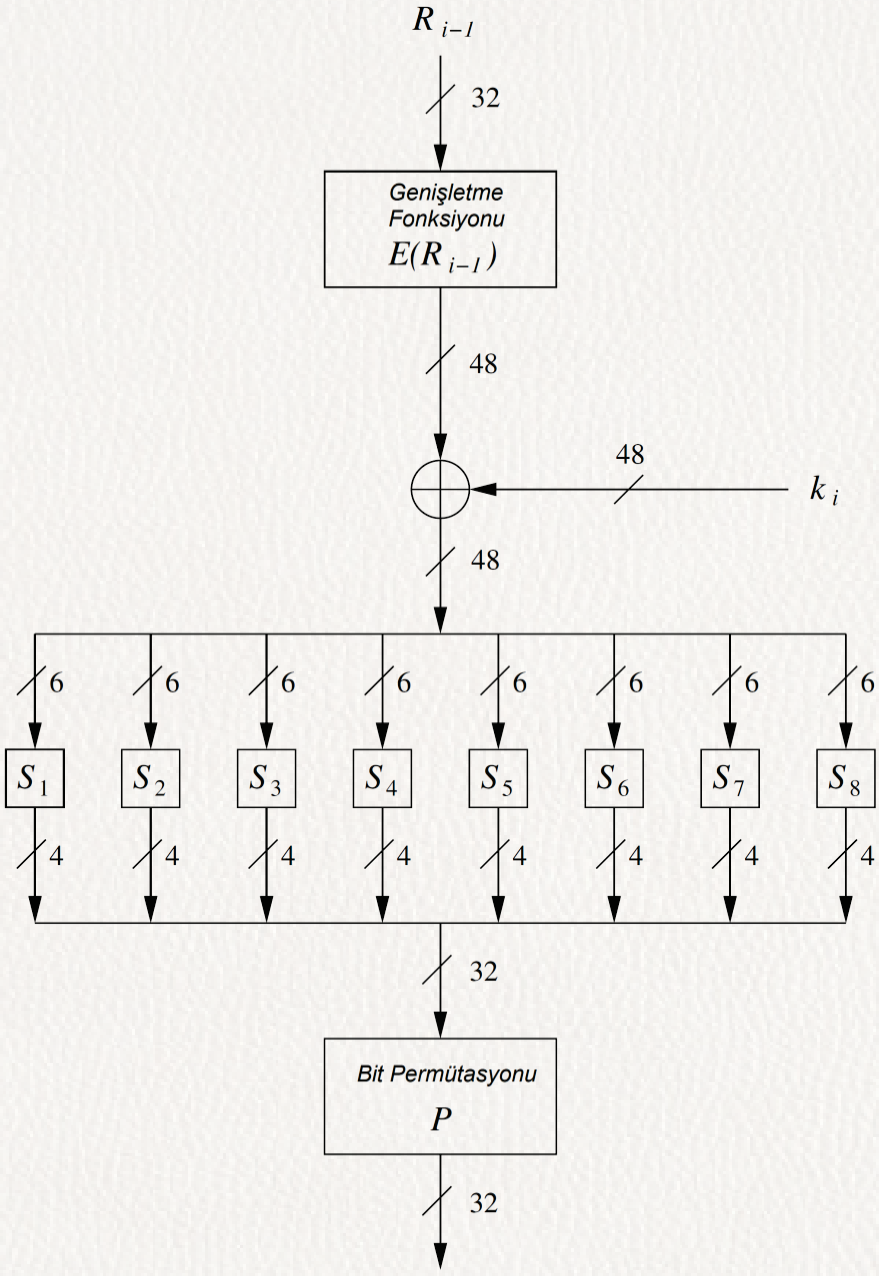
Şekil 9.4. DES Genel Yapı

DES Çevrim Fonksiyonu DES'in güvenliğinde en önemli rolü oynayan F çevrim fonksiyonu; genişletme katmanı, anahtar ekleme katmanı, S-kutusu katmanı ve bit permütasyonu katmanından oluşmaktadır. F fonksiyonu Şekil 9.6'da gösterilmiştir.

32 bitlik R_i parçası E genişletme fonksiyonu kullanılarak 48 bite genişletildikten sonra birbirinden farklı olan S kutularından geçirilir. S kutuları girdisi 6 bit çıktısı 4 bit olan doğrusal olmayan kutulardan seçilmiştir. S kutularının farksal saldırıya karşı dayanıklı olabilecek özellikler içerdiği ve en güvenli kombinasyon ile sıralandıkları görülmektedir. Bu da tasarımcıların farksal saldırı hakkında bilgi sahibi olabileceği tezini gündeme getirmiştir. Ancak, bu önlemlere rağmen DES algoritmasına kaba kuvvet saldırısından daha hızlı farksal kriptoanaliz uygulamak mümkün olmuştur [20]. Ayrıca DES'in doğrusal saldırı ile kırılabileceği de gösterilmiştir [83]. Bu saldırılar ilgili bölümlerde detaylı olarak anlatılmaktadır.



Şekil 9.5. DES Ayrıntılı Yapı



Şekil 9.6. F Fonksiyonu Ayrıntılı Yapı

9.3.2. AES (Advanced Encryption Standard)

DES'in, kısa anahtar boyuna sahip olması ve farksal, doğrusal saldırı ile kırılabilmesi farklı bir algoritma kullanılması gerekliliğini ortaya çıkardı. İlk olarak 2DES, 3DES gibi DES algoritmasının farklı türevler denense de bunların performans ve anahtar boyu olarak istenilen seviyeyi sağlayamadığı anlaşıldı. Sonuç olarak, 1997 yılında NIST tarafından yeni bir şifreleme standardı geliştirilebilmesi için bir çağrı yapıldı. Fakat bu sefer DES standartlaşma sürecinden farklı bir süreç izlendi. Yeni standart algoritma için tüm dünyanın katılımına açık yarışma başlatıldı. "Advanced Encryption Standard" adı ile başlatılan bu yarışmada tasarlanacak olan algoritmadan istenilen şartlar;

1. 128 bit blok boyuna sahip olması,
2. 128, 192 ve 256 bitlik farklı anahtar boyuna sahip kullanımı olması,
3. Yazılım ve donanımda verimli olarak çalışabilmesi,
4. Güvenlik seviyesinin başvurular arasındaki derecesinin iyi olması

olarak belirlenmişti. Yapılan başvurular bir dizi eleme sürecine tabi tutuldu ve 9 Ağustos 1999'da 5 algoritma finalist olarak açıklandı:

- MARS, (IBM Ekibi)
- RC6, (RSA Laboratuvarı)
- Rijndael, (Joan Daemen ve Vincent Rijmen)
- Serpent, (Ross Anderson, Eli Biham ve Lars Knudsen)
- Twofish, (Bruce Schneier ve arkadaşları)

2 Ekim 2000 yılında Rijdael algoritması NIST tarafından kazanan algoritma olarak seçildi ve 26 Kasım 2001 yılında standart olarak belirlendi.

AES'in Genel Yapısı AES, blok boyu 128 bit anahtar boyu ise 128, 192, 256 bit olabilen bir blok şifreleme algoritmasıdır. YPA (**SPN** -Substitution Permutation Network) yapısında tasarlanan algoritma, açık metni Feistel

yapısı gibi parçalara bölerek değil bir bütün halinde çevrim fonksiyonundan geçirmektedir. Dolayısıyla, girdi bitlerinin birbirinden etkilenmesi daha hızlı olmaktadır. 128, 192 ve 256 bit anahtar boyları için çevrim sayıları sırasıyla 10, 12 ve 14'tür. Çevrim fonksiyonu; anahtar ekleme, bayt değiştirme (S-kutusu) katmanı ve yayılım katmanından (satır kaydırma + sütun karıştırma) oluşmaktadır. AES algoritmasının genel yapısı Şekil 9.7'de verilmiştir.

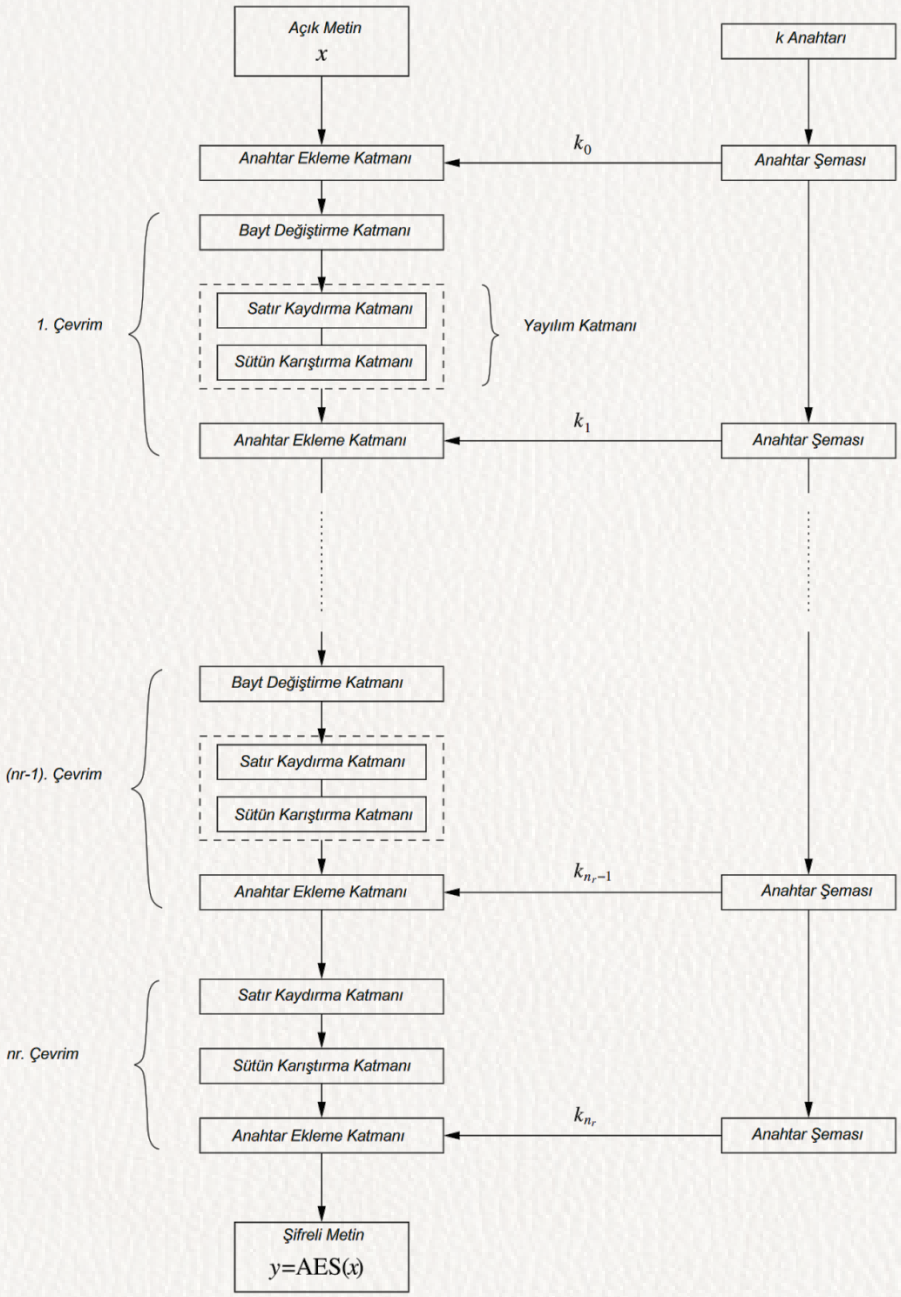
AES Çevrim Fonksiyonu AES çevrim fonksiyonuna giren 128 bitlik açık metin, elemanları $GF(2^8)$'den olan 4×4'lük matris olarak işlemlere tabi tutulur. AES çevrim fonksiyonunda bulunan yapılar aşağıdaki şekilde özetlenebilir:

Anahtar Ekleme (AE): YaDa operasyonu kullanılarak, anahtar şemasından gelen 128 bitlik değerin ilgili çevrim girdisine eklenme işlemidir.

Bayt Değiştirme (BD): 4×4'lük matrisin gözünde bulunan 16 baytın doğrusal olmayan 8 bitlik S kutularından geçirilerek yeni bir bayt değerine atanması işlemidir. Farksal ve doğrusal atağa dayanıklılık için özel olarak seçilmiştir.

Satır Kaydırma (SaK): $0 \leq i \leq 3$ için 4×4'lük matrisin her i.satırını i kadar sola kaydırma işlemidir.

Sütun Karıştırma (SuK): 4×4'lük matrisin her sütununun, yine 4×4'lük $GF(2^8)$'de tanımlı MDS matrisi ile çarpılma işlemidir. İmkansız farksal atak, integral atak vb. için dallanma sayısı önemli olduğu için matris MDS olarak seçilmiştir.



Şekil 9.7. AES Genel Yapı

9.4. DİZİ ŞİFRELEME ALGORİTMALARI

Bir dizi şifreleme algoritması k bitlik ana anahtardan ve b bitlik başlangıç vektörü olan BV'den pratikte istenildiği kadar uzun kayan anahtar dizisi üretir. Bunun için hafızalı bir kayan anahtar üreteci kullanılır. Bu kayan anahtar üretecinin hafızasında sürekli güncellenen bir içsel durum vektörü saklıdır ve o andaki kayan anahtar çıktı biti ya da kelimesi bu içsel durum vektöründen üretilir.

Üretilen kayan anahtar dizisi açık metin dizisiyle bit bazında YaDa işlemiyle toplanır ve şifreli metin dizisi elde edilir. Alıcı taraf da ürettiği aynı kayan anahtar diziyle şifreli metin dizisini YaDa işleminden geçirip açık metin dizisine ulaşır. Bunun için BV'nin de şifreli metin dizisiyle birlikte alıcı tarafa gönderilmesi gerekir. Bu şekilde yapılan şifrelemeye senkronize dizi şifrelemesi denir. Ayrıca kendi kendine senkron dizi şifreleme yöntemi de mevcuttur ancak bu algoritmalar, bölümün konusu dışında tutulmuştur.

İçsel durumlarından tohum adı verilen ilk içsel durumu üretmek için ilklendirme algoritması kullanılır. İlklendirme algoritması anahtar ve BV başlangıç vektöründen tohumu üretir. Kayan anahtar üreteci de girdi olarak tohumu alır ve her seferinde bir yandan içsel durumu günceller, bir yandan da kayan anahtar dizisinin güncel terimini üretir.

İlklendirme algoritması IA ile gösterilsin. Bu durumda S_0, s bitlik tohum olmak üzere $S_0 = IA(K, BV)$ ile elde edilir. Burada K, k bitlik ana anahtarı ve BV ise b bitlik başlangıç vektörünü göstermektedir.

Kayan anahtar üretecinde, f_G güncelleme fonksiyonunu ve f_C de çıktı fonksiyonunu temsil etsin. Bu durumda $f_G(\mathrm{S}_i) = \mathrm{S}_{i+1}$ ve $f_C(\mathrm{S}_i) = \mathrm{z}_i$ olacaktır. Kayan anahtar dizisi $\{Z_n\} = z_1, \cdots, z_i, \cdots$ şeklinde gösterilsin. Açık metin dizinin $i$nci terimi P_i ise şifreleme $C_i = P_i \oplus z_i$ ve şifre çözme $P_i = C_i \oplus z_i$ olacaktır. Burada $\oplus$ bit bazında YaDa işlemini temsil etmektedir ve eşit iki bitten 0, farklı iki bitten 1 çıktısı üretmektedir.

Bazı kayan anahtar üreteçlerinin güncelleme fonksiyonları içsel durumla birlikte ana anahtarı da kullanmaktadır: $f_G(S_i, K) = S_{i+1}$. Bu tür güncelleme fonksiyonlarına anahtarlı içsel durum güncellemesi denilmektedir. Anahtar kullanarak içsel durum güncellemesi yapan kayan anahtar üreteçleri de anahtarlı içsel durum güncellemeli kayan anahtar üreteçleri adını alırlar.

Anahtarın sabit olması ve donanıma yakılması durumunda yonganın mantık kapılarında yer kaplamayacağı varsayımı ile küçük içsel durum kullanıp ultra hafifsıklet algoritma geliştirme fikrini hayata geçirmek amacıyla son zamanlarda anahtarlı içsel durum güncellemesi yapan dizi şifreleme algoritmaları tasarlanmıştır. Sprout [5], Plantlet [87], Fruit [55] ve LILLE [12] algoritmaları bu tür algoritmalara örnek verilebilir. Ancak, şimdilik bu algoritmaların LILLE dışında hepsinde de zayıflıklar keşfedilmiştir ve literatürde anahtarlı güncelleme yapan ultra hafifsıklet güvenli algoritma tasarımı genel olarak açık bir konu olarak durmaktadır. İlk tasarlanan algoritma olan Sprout pratik olarak kırılmıştır [73,120,52]. Ardından Sprout'taki zayıflıkları gidermek amacıyla onun yeni sürümü olan Plantlet tasarlanmış ancak Plantlet'e de saldırılar yayınlanmıştır [109,114,11,38]. Bu algoritmalardan farklı olarak tasarlanan Fruit algoritmasına da saldırılar mevcuttur [114,109].

9.5. BÖL VE FETHET SALDIRILARI

Böl ve fethet saldırılarının ana hedefi, giz değerinin bir kısmını geri kalan kısmından bağımsız olarak ele geçirmektir. Bu giz değeri ana anahtar, blok şifreleme algoritmalarında çevrim anahtarı ya da kayan anahtar üreteçlerinde içsel durum olabilir.

Blok şifreleme algoritmalarına yapılan saldırılarının hemen hemen hepsi böl ve fethet türü saldırılardır. Belirli bir çevrim anahtarının belirli bitleri ile girdi/çıktı bitleri arasında genellikle istatistiksel olan bir ilişki ya da korelasyon keşfedilir. Bu korelasyon ayırt edici bir özelliktir ve hedefteki çevrim anahtarı bitlerinin doğru tespit edilmesinde kullanılır. Eğer çevrim anahtarının boyu ana anahtarın boyundan küçük ise hedefteki belirli bir çevrim

anahtarının alabileceği değerlerin her biri için ayıraç testi yapan ve veriler ile istatistiksel korelasyonu üreten adayı doğru çevrim anahtarı olarak teşhis eden saldırılar da böl ve fethet türü saldırılar sınıfına girer. Bu saldırılarda önce hedefteki çevrim anahtarına tek tek deneme yapılır ve doğru çevrim anahtarı ele geçirildikten sonra ana anahtarın geri kalan kısmı elde edilir.

Blok şifreleme algoritmalarında genellikle ayıraç belirli bir çevrim sayısı için şifreleme fonksiyonunun kendisinde mevcuttur. Verilmiş ya da seçilmiş açık metinlere karşılık gelen şifreli metinlerde (ya da bazı örneklerde açık metinlerde) belirli istatistiksel sapmalar gözlemlenir. İntegral saldırıda olduğu gibi bazen deterministik ayıraçlar da keşfedilebilir ancak bu durum nadirdir. Fazladan bir veya birkaç çevrim daha eklendiğinde istatistiksel sapma ya da deterministik ayıraç kaybolabilir. Ancak eklenen çevrimlerde işin içine giren çevrim anahtarlarının bitlerinin doğru tahmin edilmesi ve algoritmanın geriye doğru çalıştırılması ile ayıracı gözlemlemek mümkündür. Diğer taraftan, tahminin yanlış olması durumunda ise sapmanın yok olacağı varsayılır. Bu varsayım hemen hemen bütün örneklerde doğru çalışsa da genel olarak ispatlanabilmiş değildir. İlgili çevrim anahtarı bitlerine yapılan doğru tahmin ile yanlış tahminler arasındaki bu fark değerlendirilir ve çevrim anahtarı bitleri üzerinde tarama yapılarak ayraç test edilir ve bu bitler elde edilir. Bu tipik bir böl ve fethet saldırısıdır. Ana anahtar hakkında kısmi bir bilgi elde edilmiş olur. İlgili bitlerin sayısının anahtarın uzunluğundan çok daha az olması beklendiği için bu bitlerde yapılacak tarama anahtara kaba kuvvet saldırısından çok daha hızlıdır. Ancak ayıracın karakteristiğine göre ayıracı gözlemlemek için oldukça fazla veri gerekebilir.

Blok şifreleme algoritmalarına uygulanan literatürde yayınlamış ve bu kitap bölümünde yer alan cebirsel saldırı dışındaki bütün saldırılar böl ve fethet saldırılarıdır.

Dizi şifreleme algoritmalarına da böl ve fethet saldırıları uygulamak mümkündür. Bu saldırılar daha çok algoritmaların kayan anahtar üreteçlerinde içsel durumları ele geçirmeyi hedefler. Bu saldırılar korelasyon saldırıları

sınıfında toplanırlar. İçsel durumun bir kısmı ile kayan anahtar dizisi arasında içsel durumun geri kalan kısmından bağımsız olarak bir korelasyon bulunur. Bu korelasyonu yüksek ihtimalle doğrulayabilmek için yeteri kadar veri varsa içsel durumun ilgili kısmı veya ilgili kısmın tohum değeri ele geçirilebilir.

İlk korelasyon saldırısını DGÖY (Doğrusal Geri Beslemeli Ötemeli Yazmaç - LFSR) tabanlı dizi şifreleme algoritmalarına Siegenthaler önermiştir [106]. Doğrusal olmayan bağlantılı birkaç DGÖY'den birisinin çıktısı ile kayan anahtar dizisi arasında bağlantı fonksiyonunun sebep olduğu korelasyon kullanılarak, bu ilintili DGÖY'ün ilk içsel durumu hedef alınabilir. Yeterince veri varsa, kayan anahtar dizisi ile Hedef DGÖY'ün ürettiği dizi arasında korelasyonu gözlemleyerek, DGÖY'ün tohum değerini diğer DGÖY'lerden bağımsız olarak ele geçirmek mümkündür.

Belli hata düzeltme kodlarının kod çözme algoritmalarını kullanarak Siegenthaler korelasyon atağını hızlandırmak mümkündür. Bu durumda hedef DGÖY'in tohumunun tek tek denenmesi ve beklenen korelasyonun gözlemlenmesi yerine, tanımlanmış hata düzeltme kodunun hata düzeltme algoritması ile DGÖY dizisinin belli terimlerini hiç deneme yapmadan ele geçirilebileceği ve sonrasında doğrusal denklem sistemi çözerek tohumun hesaplanabileceği Meier ve Siegenthaler tarafından gösterilmiştir [86]. Bu saldırılara hızlı korelasyon saldırıları denilmektedir. Hızlı korelasyon saldırıları ile ilgili literatürde pek çok derleme ve genelleme makalesi mevcuttur [114,34,121,2,85].

Kayan anahtar üreteçlerine yapılan korelasyon saldırıları son derece etkilidir ve günümüzde tasarlanmış modern şifreleme algoritmaları dahi bu saldırılara karşı zafiyet gösterebilmektedir. Grain algoritmasına yakın zamanda uygulanmış bir korelasyon saldırısı mevcuttur [114]. Son beş yıl içinde tasarlanmış olan Fruit ve Plantlet algoritmalarında da korelasyon saldırıları uygulanmıştır [109]. Ayrıca Fruit'e yapılmış korelasyon atağı daha da geliştirilmiştir [122].

9.6. TAHMİN ET VE BELİRLE SALDIRILARI

Tahmin et ve belirle türü ataklar anahtar, çevrim anahtarı veya içsel durum gibi giz parametresinin bir kısmının tahmin edilip geri kalanının veri kullanılarak belirlenmesi yada hesaplanması yöntemine dayanır. Belirleme safhası da tamamlandıktan sonra giz parametresinin bir kısmına yapılan tahminin doğru olup olmadığı kontrol edilmelidir. Bu safhaya kontrol etme safhası denir.

Genellikle kontrol etme safhasını çalıştırabilmek için giz değerinin bütününün ele geçirilmiş olması beklenir. Böylece belirli bir girdi/çıktı çifti kullanılarak elde edilmiş giz doğru mu diye kontrol edilebilir. Ancak tahminin doğruluğunu kontrol etmek için giz parametresinin bütününü ele geçirmek zorunlu değildir. Bu durumda gizin ele geçirilmiş bir kısmının doğru olup olmadığının kontrolünü yapmak için ayrıca bir de böl ve fethet türü bir saldırı gerçekleştirilmesi gerekmektedir. Bu şekilde yapılan saldırılar hibrit saldırılardır ve son derece karmaşık olabilmektedirler.

Tahmin et ve belirle türü saldırılarda girdi/çıktı ve hedefteki giz parametresinin bir kısmına yapılan tahmine bakarak hedef gizin geri kalanlarını belirlenmesi için oluşturulan fonksiyon, şifreleme fonksiyonundan elde edilir. Bu nedenle şifrelemede kullanılan fonksiyonun belirleme fonksiyonu üretilebilecek şekilde basit olması beklenir. Genellikle blok şifreleme algoritmalarında şifreleme fonksiyonları son derece karmaşıktır ve açık metin/kapalı metin çiftleri ile anahtarın bir kısmının kullanılması sonucu anahtarın geri kalan kısmı hakkında bilgi ortaya çıkarılması son derece zor bir problemdir. Bu problem çevrim anahtarları için de oldukça karmaşık olabilir. Bu sebeple literatürde blok şifreleme algoritmalarına uygulanmış tahmin et ve belirle türü saldırılara pek rastlanmaz. Nadir örneklerden birisi Khudra algoritmasının 10 baytlık anahtarının 8 baytını tahmin edip 2 baytını bir açık metin kapalı metin çifti ile hesaplamaya, diğer bir çift ile de doğrulamaya yönelik 14 çevrimlik ataktır [94].

Tahmin et ve belirle türü saldırılar özellikle basit güncelleme fonksiyonu ve basit çıktı fonksiyonu olan kayan anahtar üreteçlerine uygulanmaktadır. Kayan anahtar dizisi kullanılarak hedef içsel durumun bir kısmı tahmin edilmekte ve geri kalan kısmı belirlenmektedir. Tamamı ele geçirilen içsel durum ile çıktı üretilir ve gerçek kayan anahtar dizisiyle karşılaştırılarak yapılan tahminin doğru olup olmadığı kontrol edilir. Bu döngü doğru tahmin yapılıncaya kadar tekrar edilir. Bu sürecin en kritik aşaması belirleme aşamasıdır. Kayan anahtar üretecinin çıktı fonksiyonu içsel durumun geri kalan kısmını belirlemeye olanak sağlayacak şekilde basit olmalıdır. Diğer taraftan içsel durumda tahmin edilen bitlerin sayısı anahtar boyundan büyükse tahmin et ve belirle saldırısı kaba kuvvet saldırısından daha yavaş olacaktır. Bu nedenle basit çıktı fonksiyonlu kayan anahtar üreteçlerinin içsel durumları anahtar boylarından çok daha büyük seçilebilmektedir. RC4 ve Trivium iki iyi bilinen örnektir. RC4'ün 128 bit anahtar boyuna karşılık $2^8 \cdot 256!$ boyutunda içsel durum uzayı, Trivium'un ise 80 bit anahtar boyuna karşılık 288 bit içsel durum boyu mevcuttur. Her iki algoritma da çok farklı tasarım tekniklerine sahip, farklı yöntemler kullanılarak tasarlanmış algoritmalardır. RC4, bayt tabanlı ve anahtar ile karıldıktan sonra rastgele üretilmiş gibi davranan bir permütasyon tablosu kullanır. Bu tablo bir yandan basit bir yer değiştirme operasyonu ile güncellenirken bir yandan da tablonun rastgele bir değeri çıktı olarak üretilmektedir. Diğer taraftan Trivium'da ise birbirlerine geribesleme yapan üç NFSR (Nonlinear Feedback Shift Register) mevcuttur. Bu iki algoritmanın ortak özelliği anahtar boylarına göre oldukça büyük içsel durumları olması ve içsel durum güncellemelerinin son derece basit olmasıdır. Bu yapıdaki algoritmaların içsel durumlarını ele geçirmeye yönelik tahmin et ve belirle türü saldırılar mümkün olabilmektedir. Örneğin Trivium'a tahmin et ve belirle türü saldırılar yayınlanmıştır ama Trivium'un içsel durum boyunun göreceli olarak oldukça büyük olması nedeniyle bu saldırılar kaba kuvvet saldırısından daha hızlı değildir ancak Trivium için kabul edilebilecek maksimum anahtar boyunu tayin edebilmek için önem arz etmektedir [84,62].

9.7 ÖDÜNLEŞİM SALDIRILARI

Hem blok şifreleme algoritmalarına hem de dizi şifreleme algoritmalarına uygulanan zaman-hafıza-veri ödünleşim saldırıları, genel saldırılardır ve şifreleme algoritmalarının iç yapılarından bağımsız olarak çalışırlar.

Hedef giz değerinin (şifrelemede anahtar, özet fonksiyonlarında ters görüntü ya da kayan anahtar üreteçlerinde içsel durum) uzayı N, kullanılan bellek M, kullanılan veri D ve harcanan zaman karmaşıklığı T ile gösterilsin. Ödünleşim saldırıları, zaman karmaşıklığının, bazı durumlarda sadece bellek ve bazı durumlarda da veriyle birlikte bellek kullanılarak, N uzayının kaba kuvvet saldırısıyla taranmasının maliyetinin altına düşürülmesi esasına dayanmaktadır. Bunun için önceden belli tabloların hazırlanması ve M belleğinin bu tablolarla doldurulması gerekmektedir. Bu safhaya çevrim dışı safhası denir ve genellikle çevrim dışı safhanın zaman karmaşıklığı verisiz ödünleşimlerde kaba kuvvet saldırısının karmaşıklığı ile aynıdır, N'dir. Veriyi oluşturmada kullanılmış gizin ele geçirilmesi işlemine çevrim içi safhası denir. Genellikle çevrim dışı safhasını kullanılan veri miktarı oranında hızlandırmak mümkündür.

Ödünleşim gerektirmeyen en temel iki saldırıdan birisi kaba kuvvet saldırıdır [128]. Bu durumda $T = N$'dir ve M ile D ihmal edilecek kadar azdır. Diğer saldırı ise tüm hedef girdileri çıktılarıyla birlikte bir bellekte çıktılara göre sıralı bir şekilde saklamaktır. Bu durumda $M = N$ olur ve T ile D ihmal edilecek kadar azdır. Her iki saldırıda da ödünleşimden söz etmek mümkün değildir. Ödünleşim saldırılarında zamanla bellek arasında ya da veri ile bellek arasında bir ödünleşim söz konusudur. Bazen her üçü arasında ödünleşim olabilir. Bu ödünleşimde bir parametrenin artması diğer parametrenin belli oranda azalmasına sebep olur. Bu oranı ifade eden fonksiyonun grafiğine ödünleşim eğrisi denir.

Blok şifreleme algoritmalarına yapılan ilk ödünleşim saldırısı Hellman saldırısıdır [57]. Hellman DES'e uyguladığı ödünleşim saldırısında bellek kullanarak kaba kuvvet saldırısını geliştirebileceğini göstermiştir. Tablolar yar-

dımıyla, ödünleşim eğrisi $M^2\,T = N^2$ olacak şekilde blok şifreleme algoritmalarının anahtar uzaylarını taramada zaman karmaşıklarını düşürmek mümkündür. Eğri üzerindeki optimum nokta $T = M = N^{2/3}$ noktasıdır. Örneğin, tek bir seçili açık metin kapalı metin çifti ile önceden hazırlanmış $2^{50,3}$ TB bellek kullanarak çevrim içi zamanda $2^{85,3}$ deneme ile 128 bitlik AES anahtarını ele geçirmek mümkündür. Ancak toplamda $2^{50,3}$ TB boyutundaki tabloları hazırlamak için çevrim dışında 2^{128} AES şifrelemesinin yapılması gerekmektedir. Dolayısıyla, AES'e bir kere dahi kaba kuvvet saldırısı uygulanabilecek teknolojik kapasiteye sahip olunmadığı bir dünyada AES'e ödünleşim atağını gerçeklemekten de söz etmek mümkün gözükmemektedir.

Dizi şifreleme algoritmalarına uygulanan ödünleşim saldırılarında ana anahtarı ele geçirmekten daha çok, kayan anahtar üretimi sırasında güncellenen içsel durumlardan herhangi birisinin ele geçirilmesi hedeflenir. En temel ödünleşim saldırısı, bağımsız zamanlarda Golic [56] ve Babbage [9] tarafından önerilmiştir.

Hellman saldırısından farklı olarak Golic-Babbage ödünleşimi veri kullanır ve veri ile bellek arasında bir ödünleşim eğrisi kurar. Bu ödünleşimde içsel durum uzayının büyük bir çoğunluğu, ürettiği çıktılarla birlikte bellekte depolanır. Hedef kayan anahtar dizisinden yeterince veri varsa, bu dizinin üretiminin belirli bir anında bellekte yer alan içsel durumlardan birisinin kullanılmış olması ihtimali kayda değer derecede yüksek olacaktır. Bu da kullanılan içsel durumlardan birisinin bellekteki tablolar yardımıyla ele geçirilmesine olanak sağlar. Golic-Babbage saldırısının ödünleşim eğrisi doğum günü paradoksunun bir sonucu olarak $MD = N$ şeklindedir. Burada her bir verinin tabloda olup olmadığı kontrol edileceği için $T = D$'dir. Eğrideki en optimum nokta $M = D = \sqrt{N}$ noktasıdır. Dolayısıyla atağın kaba kuvvet atağından daha yavaş olabilmesi için $\sqrt{N}$'in anahtar uzayından daha büyük olması beklenir. Diğer bir ifade ile dizi şifreleme algoritmalarının kayan anahtar üreteçlerinin içsel durum boyları, anahtar boylarının en az iki katı kadar olmalıdır.

Bu şart kriptoloji camiasında kabul görmüş oldukça yaygın bir güvenlik ölçütü olarak karşımıza çıkmaktadır.

Bu güvenlik ölçütü özellikle son yirmi yılda hafif-sıklet dizi şifreleme algoritması tasarımında önemli bir engel teşkil etmiştir. Anahtar boyunun en az iki katı boyunda bir içsel durum tasarlamak ve bu içsel durumu özellikle donanımda yazmaçlarda saklamak gerekmektedir. Bunun bir sonucu olarak, özellikle son zamanlarda tasarlanmış hafifsıklet simetrik şifreleme algoritmalarının ezici bir çoğunluğunun blok şifreleme algoritmaları olduğu görülmektedir.

Anahtar boyunun en az iki katı uzunluğunda içsel durum boyu ölçütü, ödünleşim ataklarına karşı gereğinden fazla konservatif davranılarak alınmış bir önlem olarak karşımıza çıkmaktadır. Bu aşırı ihtiyatlı yaklaşımın temelinde ödünleşim ile içsel durum ele geçirme saldırılarını, kaba kuvvet saldırılarıyla karşılaştırmak yatmaktadır. Halbuki, içsel durumu ele geçirmek amacıyla dizi şifreleme algoritmalarının kayan anahtar üreteçlerine uygulanan ödünleşim saldırılarının kaba kuvvet saldırısı ile değil de anahtarı ele geçirmeye yönelik konvansiyonel ödünleşim saldırılarıyla karşılaştırılmaları gerektiği yakın zamanda yayınlanmış iki kitap bölümünde tartışılmıştır [88,65]. Bu durumda bir kayan anahtar üretecinin içsel durum boyunun anahtar uzunluğunun en azından iki kat olması yerine 4/3 katı olması, ödünleşim ataklarına karşı güvenlik için yeterlidir.

İçsel durum boyunun anahtar boyunun 4/3'ü ile sınırlandırılmasıyla ultra hafifsıklet dizi şifreleme algoritma tasarımının önü açılabilir. Ancak bu koşulun benimsenmesi için her bir anahtar ile yapılacak şifrelemelerin anahtar uzayının küp köküyle sınırlandırılması gerekmektedir. Böylece 4n/3 boyutunda içsel durumu olan bir dizi şifreleme algoritmasında, ödünleşim saldırılarının çevrim dışı zaman karmaşıklıkların kaba kuvvet saldırısından daha hızlı olmalarının önüne geçilebilir. Genellikle küçük içsel boylu dizi şifreleme algoritmaları hafifsıklet uygulamalarda göz önünde bulunduruldukları için, anahtar başına şifrelenecek veri miktarı sınırlandırılmasının pratikte

verimsizliğe neden olması beklenmez. Dolayısıyla, ölçütteki bu yeni sınırın benimsenmesinin özellikle hafifsıklet dizi şifreleme algoritma tasarımlarının gelişmesine olumlu etki etmesi beklenmektedir.

Bir $f: GF(2)^n \rightarrow GF(2)^n$ tek yönlü fonksiyonu için, ters görüntü bulma problemi, verilen bir $y \in GF(2)^n$ için $f(x) = y$ olacak şekilde bir $x \in GF(2)^n$ girdisi bulma problemidir. Güvenli bir tek yönlü fonksiyon için hiçbir hazırlığın yapılmadığı durumda x'i bulmanın en hızlı yolu tek tek değerleri deneme anlamına gelen kaba kuvvet saldırısıdır ve karmaşıklığı $N = 2^n$'dir. Ancak, Hellman tabloları ile 2^n denemeden daha kısa sürede ters görüntü bulmak mümkündür.

Ters görüntü kümesinde bir eleman bulunması amacıyla, çıktı $y \in GF(2)^n$ değeri verilmeden önce f fonksiyonu ile bazı tablolar oluşturmak ve y değerini bu tablolarda arayarak ters görüntü bulmayı hızlandırmak mümkündür [57]. Önceden seçilmiş $z_j \in GF(2)^n$ değerleri ile

$$z_j, f(z_j), f^2(z_j), \cdots, f^t(z_j)$$

zincirleri oluşturulsun. Bu zincirlerden m tanesi bir araya getirilerek başlangıç değerleri hariç mt elemanlı bir matris üretilebilir. Matrisin satırları son sütununu oluşturan $f^t(z_j)$ değerlerine göre sıraya dizilirse, verilmiş bir y değerinin matriste yer aldığı, herhangi bir i değeri için $f^i(y)$'in matrisin son sütununda olup olmadığı kontrol edilerek test edilebilir. Bu matrislere Hellman tabloları adı verilmektedir.

Eğer hiçbir $i = 0, \cdots, t-1$ değeri için $f^i(y)$ değeri matrisin son sütununda yer almıyorsa y matriste değildir. Aksi durumda, $f^i(y) = f^i(z_j)$ olacak şekilde bir (i, j) ikilisi vardır. Bu durumda, $y = f^{t-i}(z_j)$ eşitliğinin doğru olma ihtimali yüksektir. Eğer eşitlik doğru ise bu durum y'nin tabloda yer aldığı anlamına gelir. Tabloda ilk sütundakiler hariç her elemanın ters görüntüsü mevcuttur ve kolayca bulunabilir. Eğer y tabloda ise $f(f^{t-i-1}(z_j)) = y$ olacaktır. Daha açık bir ifadeyle, $f^{t-i-1}(z_j)$ bir ters görüntü olacaktır.

Verilen y değerinin Hellman tablosunda olup olmadığını araştırmak için tablounun sadece son sütunu saklamak yeterlidir. Eğer y değerinin tablonun belirli satırında ve sütununda olduğu kanaatine varılır ise bu satır ve bir önceki sütundaki elemanı bulmak için tablonun ilk sütuna ihtiyaç duyulmaktadır. Dolayısıyla mt elemanlı tablonun sadece ilk ve son sütunlarını bellekte saklamak yeterlidir.

Bir Hellman tablosu mt eleman saklar ve hepsi de farklı ise tablo mükemmel tablo adını alır. Eğer tabloda bir çakışma olursa çakışmanın olduğu satırlarda çakışma, satırlardan birinde sonuna varıncaya dek devam eder. Bu nedenle Hellman tablolarının verimli olarak saklayabilecekleri belirli kapasiteleri vardır. Eğer $mt^2 > N$ ise tabloda çakışma ihtimalleri son derece yüksektir ve çakışmalar oldukça yaygındır. Diğer taraftan $mt^2 \leq N$ olduğu durumda da büyük ihtimalle çakışmalar olacaktır ancak önemli olan tabloda çakışmaların olması değil, kaç farklı eleman olduğudur. Eğer $mt^2 \leq N$ ise, bu çakışmaların beklenen sayıları son derece azdır ve tablodaki farklı eleman sayısı neredeyse mt kadardır. Bu nedenle tablodaki satır ve sütun sayısı $mt^2 = N$ olacak şekilde sınırlandırılır. Ancak tek bir tabloda en fazla mt girdi saklanabileceği için f fonksiyonun basit varyasyonlarıyla elde edilmiş fonksiyonlar ile t tane farklı tablo kurmak gerekmektedir.

Her bir tablo m kadar yer kaplamaktadır ve sonuçta t tabloyu saklamak için $M = mt$ kadar belleğe ihtiyaç vardır. Diğer taraftan y değerinin bir tabloda olup olmadığını kontrol etmek için en kötü durumda t kere f fonksiyonunu çağırmak gerekmektedir. Dolayısıyla t tablonun hepsinde y 'yi aramanın zaman karmaşıklığı $T = t^2$'dir. Sonuç olarak $M^2T = m^2t^2 = N^2$ ifadesi elde edilir. Başka bir ifadeyle, ödünleşim eğrisi $M^2T = N^2$ olacaktır. Her bir tablonun hazırlanmasının maliyeti ise mt kere f fonksiyonunu çağırmaktır. Toplam t tablonun inşası için $mt^2 = N$ kere f fonksiyonunun çağrılması gerekmektedir. Daha açık bir ifadeyle, tabloların çevrim dışında oluşturulmalarının maliyeti kaba kuvvet saldırısınınki kadardır.

Hellman tablolarıyla herhangi bir tek yönlü fonksiyona ters görüntü bulunabilir. Fonksiyonun girdi ve çıktı boyları eşit olmak zorunda değildir. Bu

durumda hedef fonksiyonu, belli indirgemeler veya genişletmeler yaparak Hellman tablosu oluşturulabilen bir tek yönlü fonksiyona dönüştürmek mümkündür. Ayrıca bir blok şifreleme algoritması tek yönlü bir fonksiyon olarak tanımlanabilir. $E_K(P) = C$ bir blok şifreleme algoritması olsun. Bu durumda, önceden seçilmiş bir P_0 açık metni için $f(x) = E_K(P_0) = y$ tek yönlü fonksiyonu tanımlanabilir. Fonksiyon anahtarı girdi olarak alıp, P_0'ın şifreli halini çıktı olarak üretmektedir. Hellman tablosu ile bulunan ters görüntü anahtar olmayabilir. Ancak tablolar anahtar uzayının çoğunluğunu kaplıyorlarsa tablodan anahtarı bulma ihtimali de o oranda yüksek olacaktır. Bulunan ters görüntünün anahtar olmaması durumunda tabloları tarama işlemi başka bir ters görüntü bulununcaya kadar kaldığı yerden devam ettirilir.

Benzer şekilde, seçilmiş bir IV_0 ile anahtarı girdi olarak alan ve anahtar boyu kadar kayan anahtar biti üreten fonksiyon da dizi şifreleme algoritmasından üretilmiş tek yönlü bir fonksiyondur. Verilmiş kayan anahtar dizi parçası için fonksiyonun girdisi anahtar olacaktır. Diğer bir ifade ile anahtar boyu k bit olan hem blok şifreleme algoritmaları, hem de dizi şifreleme algoritmaları için çevrim içi zaman karmaşıklığı $2^{2k/3}$ olacak şekilde konvansiyonel bir ödünleşim atağı yapmak mümkündür. Dolayısıyla, diğer her türlü ödünleşim atağının başarısı kaba kuvvet saldırı ile karşılaştırılarak değil, bu konvansiyonel öndünleşim saldırısı ile karşılaştırılarak ölçülmelidir. Diğer taraftan, bu konvansiyonel saldırının çevrim dışı zaman karmaşıklığı kaba kuvvet saldırısının karmaşıklığı kadardır. Herhangi bir ödünleşim saldırısının çevrim dışı safhası da kaba kuvvet saldırısıyla karşılaştırılmalıdır.

9.7.1. Saldırının Güncel Uygulamaları ve Türevleri

Hellman tablosu hazırlanırken tek bir fonksiyon kullanılır. Bu durumda tabloda herhangi iki satırdaki çakışma ilgili satırlar boyunca yayılacaktır. Oechslin tablo oluştururken her sütunda farklı bir fonksiyon kurmayı önerdi [93]. Böylece tabloda iki satırdaki çakışma eğer aynı sütuna denk gelmiyorsa yayılmayacaktır. Böylece uzayın hemen hemen bütün elemanlarını tek bir tabloda temsil etmek mümkün olabilmektedir. Ancak verilen bir $y \in GF(2)^n$ değeri-

nin tabloda olup olmadığını kontrol etmek çok daha karmaşık bir hal almaktadır. Oechslin bu tabloya "Gökkuşağı Tablosu" adını vermiştir.

$z_j \in GF(2)^n$ değerleri ile $f_1(z_j)$ sütunu oluşturulur. Sonra $f_2(f_1(z_j))$ sütunu oluşturulur. Bu şekilde devam edilerek en son sütunda $f_t(\cdots f_1(z_j))$ değerleri sıralı halde saklanır. Burada f_i fonksiyonları f fonksiyonunun basit varyasyonlarıdır. Verilen bir y değerinin matriste olup olmadığını anlamak için son sütunda mı diye bakılır, yoksa $f_t(y)$ son sütunda mı diye kontrol edilir. O da yoksa bu durumda $f_t(\ f_{t-1}(y))$ son sütunda mı diye kontrol edilmelidir. Bu şekilde devam edildiğinde, y değerinin $(t-i)$-inci sütunda olup olmadığını test etmek için, $f_i(\cdots f_1(y))$ değeri son sütunda mı diye kontrol etmek gerekmektedir. Sonuç olarak f fonksiyonu $t(t+1)/2$ kere çağrılacaktır. Burada tek bir tablo olduğundan $M = m$ ve $T = t^2$'dir ve $M^2T = N^2$ eğrisi elde edilir. Asimptotik olarak bu eğri Hellman ödümleşiminin eğrisi ile denktir.

Hellman ya da Gökkuşağı tabloları kullanılarak kayan anahtar üreteçlerinin içsel durumlarını ele geçirmek mümkündür. Kayan anahtar dizisinin üretiminde içsel durumlar sürekli güncellendiği ve birden fazla içsel durum kullanıldığı için bu içsel durumlardan herhangi birini ele geçirmek yeterli olacaktır. Biryukov ve Shamir kayan anahtar üretecinde çoklu veri kullanarak Hellman tablolarını blok şifrelemeye nazaran çok daha etkili kullanılabileceğini göstermişlerdir [25]. Bu durumda ödünleşime veri de dahil olmuştur ve $M^2D^2T = N^2$ eğrisi elde edilmiştir. Burada en verimli durum olan tek bir tablo kullanılması dikkate alınmıştır ve bunun sonucu olarak $D \leq \sqrt{T}$ kısıtlaması söz konusudur. Eğrinin optimum noktası $T = M = \sqrt{N}$, $D = N^{1/4}$ değeridir. Ayrıca çevrim dışı safhası N/D kadardır.

Barkan ve arkadaşları Hellman tabloları ile Gökkuşağı tablolarını karşılaştırmış ve hibrit tabloların performanslarını hesaplamışlardır [14].

Hellman tablolarının bir önemli yan etkisi yanlış alarmların çok olmasıdır. Verilen bir y değeri için, $f^i(y)$ tablonun son sütununda gözükmesine rağmen y'nin kendisi tabloda değilse, Hellman saldırısında buna yanlış alarm

denir. Yanlış alarmlar asimptotik olarak zaman karmaşıklığına etki etmese de pratik uygulamalarda son derece etkilidir. Gildas ve arkadaşlarının yanlış alarmların önceden fark edilip, zaman karmaşıklığının iyileştirilmesine yönelik önerileri mevcuttur [8].

Ödünleşim atakları, özellikle Hellman ve Gökkuşağı tabloları, içsel durum boyu ya da anahtar boyu küçük olan şifreleme algoritmalarında son derece etkilidir ve pratik olarak anahtar ele geçirmede kullanılırlar. Birçok araba firmasının motor kilitlerinde (immobilizer) kullanılan Hitag2 algoritmasının ödünleşim ile çevrim içi kırıldığı gösterilmiştir [112]. Pratik ve güncel uygulamaları ile ilgili daha fazla örnek ve detaylı bilgi için [107] ve [64] numaralı kaynaklar incelenebilir.

Gökkuşağı tablolarının bir kullanım alanı da özet fonksiyonlarına ters görüntü bulma problemini çözmektir. Özet fonksiyonları parolaları güvenli saklamada kullanılır ve bir parolanın (varsa tuz değeri ile birlikte) özet değerine sahip başka her türlü girdi parola ile eşdeğerdir ve sisteme güvenli giriş için kullanılabilir. Bu nedenle parola özet değerlerine ters görüntü bulmak amacıyla Gökkuşağı tabloları kullanılır ve GPU, FPGA gibi güçlü işlemciler ile pratik olarak parolaların kırılabildiği gösterilmiştir [7,6,16,59,119].

9.8. DOĞRUSAL KRİPTOANALİZ

Modern blok şifreleme algoritmalarının atası olarak bilinen DES şifreleme algoritmasına uygulanabilen iki temel saldırı yönteminden birisi olan doğrusal kriptoanaliz aynı zamanda DES'e yapılmış en başarılı saldırıdır.

Doğrusal saldırı yöntemi Matsui tarafından geliştirilmiş ve ilk olarak DES algoritmasına uygulanmıştır [83]. Saldırı yönteminin ana felsefesinde şifreleme algoritmasının girdisi, çıktısı ve anahtar bitleri arasında yüksek ihtimallerle sağlanan doğrusal yaklaşımlar oluşturulması yatmaktadır. Şifreleme algoritmasının kendisine bu doğrusal yaklaşımları oluşturmak için algoritmanın yapıtaşlarında doğrusal olmayan fonksiyonlara doğrusal yaklaşımlar

yapılır ve bu yaklaşımların çevrimler boyunca yayılması hesaplanır. Böylece doğrusal karakteristikler oluşturulur.

Algoritmada her bir doğrusal olmayan fonksiyon için bir doğrusal yaklaşım tablosu oluşturulur. Bu tablolara kısaca LAT tabloları denmektedir. Bir fonksiyonun LAT (Linear Approximation Table) tablosunun satır ve sütunlarında, fonksiyonun her bir girdi bitleri doğrusal kombinasyonun, her bir çıktı bitleri doğrusal kombinasyonuna kaç girdide eşit olduğu yazılıdır. Fonksiyonun herhangi bir girdi doğrusal kombinasyonunun çıktı doğrusal kombinasyonuna eşitlenmesine fonksiyonun doğrusal yaklaşımı denir. Bu doğrusal yaklaşımın sağlandığı girdi değerlerinin tüm uzaya oranı da doğrusal yaklaşımın ihtimalidir.

Bir blok şifreleme algoritmasında yer alan doğrusal olmayan yapı taşlarına yapılan doğrusal yaklaşımlar çevrimler boyu ilerletilebilir. Algoritmadaki doğrusal fonksiyonlar bu doğrusal yaklaşımları başka doğrusal yaklaşımlara dönüştürürler ancak ihtimallerini değiştirmezler. Böylece çevrimler boyu ilerleyen doğrusal yaklaşımlar ile şifreleme algoritmasının bütünü için her bir anahtara özel olacak şekilde açık metin ve şifreli metin bitleri arasında bir doğrusal yaklaşım bulunur. Bu doğrusal yaklaşımın ihtimalinin bire ya da sıfıra olabildiğince yakın olmasına, diğer bir ifade ile % 50'den sapmasının olabildiğince yüksek olmasına dikkat edilir.

Bir blok şifreleme algoritmasında P açık metin, C karşılık gelen şifreli metin ve K anahtar olsun. $P[i_1, \cdots, i_k] = P[i_1] \oplus \cdots \oplus P[i_k]$ olsun. $P[i_1, \cdots, i_k]$ açık metnin $i_1, \cdots, i_k$ bitlerinin YaDa'larını (modulo ikide toplamlarını) ifade etsin. Benzer şekilde $C[i_1, \cdots, i_j] = C[i_1] \oplus \cdots \oplus C[i_j]$ ve $K[i_1, \cdots, i_r] = K[i_1] \oplus \cdots \oplus K[i_r]$ olsun. Bu durumda, belirli bir yöntem izlenerek şifreleme algoritması için elde edilmiş

$$P[i_1, \cdots, i_p] \oplus C[j_1, \cdots, j_r] = K[k_1, \cdots, k_s] \quad (1)$$

doğrusal yaklaşımı anahtardan bağımsız olarak $p = \frac{1}{2} + \epsilon$ ihtimali ile doğru olsun. Sapma değerini ifade eden ϵ sıfırdan farklıysa doğrusal saldırı yap-

mak mümkün olabilir. Özellikle $|\epsilon|$, 1/2'ye ne kadar yakın bir değer ise Denklem 1 ile ifade edilen doğrusal yaklaşım o derece güçlü olmaktadır. Doğrusal yaklaşımının sol tarafı açık metin kapalı metin çiftlerinden oluşmaktadır ve değişkendir. Eşitliğin sağ tarafındaki anahtar bitleri ise bilinmezdir ve sabittir. Dolayısıyla yeterince açık metin şifreli metin çifti alınır ve her bir çift için eşitliğin sol tarafında oluşan değerin 0 mı ya da 1 mi olduğuna bakılır. Bütün çiftler için eşitliğin solundaki hesaplanan 0'lar ve 1'ler sayılır ve 0 ya da 1'den hangisinin çoğunlukta olduğu tespit edilir. Eğer ϵ pozitif ise, anahtar bitlerinin $K[i_1, \cdots, i_r]$ doğrusal kombinasyonunun eşitliğin sol tarafında oluşan değerlerden çoğunluğa eşit olduğu hükmüne varılır. Benzer şekilde, eğer ϵ negatif ise $K[i_1, \cdots, i_r]$'in azınlığa eşit olduğu hükmüne varılır. Bu da anahtar hakkında bir bitlik bilgi içerir. Bu hükmün %90 gibi kayda değer derecede yüksek ihtimalle doğru olması için yaklaşık $1/\epsilon^2$ kadar açık metin kapalı metin çiftine ihtiyaç vardır. Matsui bu saldırıyı Algoritma 1 olarak ifade etmiştir [83].

Denklem 1, algoritmanın bütünü için değil de birkaç çevrim eksiği için de bulunabilir. Bu durumda yukarıdan ya da aşağıdan eklenen çevrimlerde Denklem 1'in sol tarafını hesaplamak için işin içine giren çevrim anahtar bitlerine deneme yapılır. Her bir anahtar denemesi için açık metin kapalı metin çiftlerinden Denklem 1'in sol tarafında oluşan kombinasyonların değerleri bulunur ve toplamda anahtar başına sol tarafın sapması hesaplanır. En yüksek sapmaya sahip anahtar bitleri doğru anahtar bitleri olarak değerlendirilir. Ayrıca sapmanın pozitif ya da negatif olmasına göre $K[i_1, \cdots, i_r]$ doğrusal kombinasyonunun değerine karar verilir. Bu şekilde çok daha fazla anahtar biti ele geçirilebilir. Matsui bu saldırıyı Algoritma 2 olarak ifade etmiştir [83]. Algoritma 2'de doğrusal yaklaşım daha az sayıda çevrime yapıldığı için sapmanın da tüm çevrime yapılmış doğrusal yaklaşıma göre daha yüksek olması beklenir. Ancak bu yöntemde gerekli veri sayısı taranacak anahtar bitlerinin sayısına bağlı olarak $1/\epsilon^2$ 'nin birkaç katı artacaktır.

9.8.1. Doğrusal Karakteristik Bulma

Bir blok şifreleme algoritmasının girdi, çıktı ve anahtar bitleri arasında bir doğrusal yaklaşım oluşturmak için her bir çevrim fonksiyonuna doğrusal yaklaşımlar bulmak gerekir. Bu doğrusal yaklaşımlar öyle olmalıdır ki ardışık çevrimlerde bir yaklaşımın çıktısı sonraki yaklaşımın girdisine eşit olmalıdır. Böylece her bir çevrimde oluşan doğrusal denklemlerin alt alta yazılıp mod 2'de toplanmasıyla elde edilen doğrusal yaklaşımda, ara değerler çift sayıda gözükmeli ve dolayısıyla elenmeli ve sadece açık metin, kapalı metin ve anahtar bitleri yer almalıdır. Aşağıda her bir çevrim için yazılmış doğrusal yaklaşımlar alt alta toplandığında çevrimlerin içsel değerleri olan X_i değerlerinin doğrusal kombinasyonları kaybolacaktır.

1. çevrim: $P[i_1,\cdots,i_s]\oplus X_1[j_1,\cdots,j_r] = K_1[k_1,\cdots,k_p]; \quad p_1 = \frac{1}{2}+\epsilon_1$

2. çevrim: $X_1[j_1,\cdots,j_r]\oplus X_2[f_1,\cdots,f_n] = K_2[d_1,\cdots,d_m]; \quad p_2 = \frac{1}{2}+\epsilon_2$

Son çevrim: $X_{R-1}[b_1,\cdots,b_t]\oplus C[l_1,\cdots,l_v] = K_R[m_1,\cdots,m_y]; \quad p_R = \frac{1}{2}+\epsilon_R$

Bu doğrusal yaklaşımlarının toplanmasıyla sadece açık metin, şifreli metin ve çevrim anahtar bitlerinin yer aldığı

$$P_i[i_1,\cdots,i_s]\oplus C[l_1,\cdots,l_v] = K_1[k_1,\cdots,k_p]\oplus \cdots \oplus K_R[m_1,\cdots,m_y] \qquad (2)$$

doğrusal yaklaşımı elde edilir. Her bir çevrimdeki doğrusal yaklaşımların birbirlerinden bağımsız olduğu, bir doğrusal yaklaşımın sağlandığı durumda diğer doğrusal yaklaşımların sağlanma ihtimallerinin değişmediği varsayımı altında, Denklem 2'nin doğru olma ihtimali

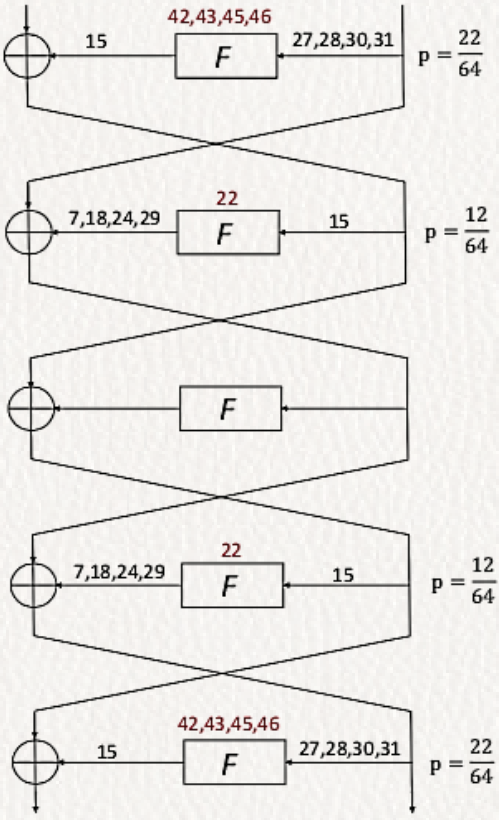
$$p = \frac{1}{2}+\epsilon;\ \epsilon = 2^{R-1}\prod_{i=1}^{R}\epsilon_i\,,$$

ifadesine eşittir. Atağın başarısı sapmanın mutlak değer büyüklüğüne, ϵ değerinin olabildiğince 1/2'ye ya da -1/2'ye yakın olmasına bağlıdır. Bunun için her bir ϵ_i değerinin olabildiğince 1/2'ye ya da -1/2'ye yakın olması gerekir. Dolayısıyla her bir çevrim için oluşturulan doğrusal yaklaşımların sap-

maları olabildiğince yüksek olmalıdır. Bu doğrusal yaklaşımların bir araya gelmesiyle blok şifreleme algoritması için doğrusal karakteristik oluşturulmuş olur.

9.8.2. DES'e Doğrusal Karakteristik

Feistel ağı yapısındaki DES'in 5 çevrimi için Matsui tarafından [83] bildirisinde oluşturulmuş doğrusal karakteristik Şekil 9.8'de gösterilmiştir. Her bir çevrim fonksiyonu için verilen doğrusal yaklaşım ve yaklaşımın ihtimali aşağıda sıralanmıştır.



Şekil 9.8. Matsui'nin 5 Çevrimlik DES'e Doğrusal Karakteristiği [83].

Şekil 9.8 verilen F fonksiyonunun sağındaki bitler doğrusal yaklaşımdaki girdileri, solundaki bitler de çıktıları göstermektedir. Üstünde yer alan kırmızı renkteki bitler ise çevrim anahtarı bitleridir.

1. çevrim: $X_1[27,28,30,31] \oplus K_1[42,43,45,46] = Y_1[15]; \quad p_1 = \frac{22}{64}$

2. çevrim: $X_2[15] \oplus K_2[22] = Y_2[7,18,24,29]; \quad p_2 = \frac{12}{64}$

4. çevrim: $X_4[15] \oplus K_4[22] = Y_4[7,18,24,29]; \quad p_4 = \frac{12}{64}$

5. çevrim: $X_5[27,28,30,31] \oplus K_5[42,43,45,46] = Y_5[15]; \quad p_5 = \frac{22}{64}$

Burada X_i, i ninci çevrimde F fonksiyonu girdisini, Y_i, $i$ninci çevrimde F fonksiyonu çıktısını ve K_i ise $i$ninci çevrim anahtarını göstermektedir. Bu denklemlerde

$$X_2[15] \oplus Y_1[15] = P_L[15]$$

$$Y_2[7,18,24,29] \oplus X_1[27,28,30,31] \oplus X_3[7,18,24,29] = P_R[7,18,24,29,27,28,30,31]$$

olacak şekilde yazılırsa ve bu iki ifade toplanırsa

$$X_1[27,28,30,31] \oplus Y_1[15] \oplus X_2[15] \oplus Y_2[7,18,24,29]$$

yaklaşımının yerine

$$P_L[15] \oplus P_R[7,18,24,29,27,28,30,31] \oplus X_3[7,18,24,29]$$

yaklaşımının kullanılabileceği görülür. Burada P_L açık metnin sol yarısı, P_R ise açık metnin sağ yarısıdır. Dikkat edilirse $X_3[7,18,24,29]$ kombinasyonu 3. çevrimde F fonksiyonunun girdisinden oluşturulmuş olsa da doğrusal karakteristikte 3. çevrim pasiftir (bkz Şekil 9.8). Benzer şekilde, $X_3[7,18,24,29]$ kombinasyonunun yer aldığı bir yaklaşımı şifreli metin için de yazmak mümkündür.

$$X_4[15] \oplus Y_5[15] = C_L[15]$$

$$Y_4[7,18,24,29] \oplus X_5[27,28,30,31] \oplus X_3[7,18,24,29] = C_R[7,18,24,29,27,28,30,31]$$

yaklaşımlarını kullanıp

$$X_5[27,28,30,31] \oplus Y_5[15] \oplus X_4[15] \oplus Y_4[7,18,24,29]$$

yaklaşımının yerine

$$C_L[15] \oplus C_R[7,18,24,29,27,28,30,31] \oplus X_3[7,18,24,29]$$

yaklaşımı yazılabilir. Sonuç olarak bütün bu dört yaklaşım alt alta toplandığında açık metin ve kapalı metin çiftindeki

$$P_L[15] \oplus P_R[7,18,24,29,27,28,30,31] \oplus C_L[15] \oplus C_R[7,18,24,29,27,28,30,31]$$

yaklaşımının

$$K_1[42,43,45,46] \oplus K_2[15] \oplus K_4[15] \oplus K_5[42,43,45,46]$$

anahtar bitleri doğrusal kombinasyonuna $p = \frac{1}{2} + 2^3 \frac{100-400}{2^{24}}$ ihtimalle eşit olması beklenir.

Benzer şekilde başka doğrusal karakteristikler oluşturmak mümkündür. En yüksek sapmalı karakteristikler Matsui'nin orjinal çalışmasında tablo halinde verilmiştir [83]. Bu karakteristikler içinde 16 çevrimlik DES'i kırmak için kullanılan 15 çevrimlik karakteristik te vardır.

9.8.3. AES'in Doğrusal Kriptoanalizi

DES'de keşfedilen zayıflıklar ve özellikle DES'in anahtar boyunun kısa olmasından dolayı DES'in yerine alacak ABD NIST FIPS 198 standardı için açılan yarışmaya katılan algoritmalar özellikle doğrusal saldırıya ve farksal saldırıya dayanıklı olacak şekilde tasarlanmıştı. AES algoritmasının kendisi de doğrusal saldırıya karşı güvenli olacak şekilde tasarlanmıştır.

AES'in sütun karıştırma operasyonu M matrisi ile çarpma işlemidir. Koordinatları $GF(256)$ sonlu cisminde tanımlı $x = (x_1, x_2, x_3, x_4) \in GF(256)^4$ ve $y = (y_1, y_2, y_3, y_4) \in GF(256)^4$ vektörleri için; M matrisi x vektörünü $M \cdot x^\top = y^\top$ ifadesi ile y vektörüne dönüştürmektedir. Burada $x^\top$, x'in transpozudur. y vektörü için verilecek bir $\alpha_1 y_1 \oplus \alpha_2 y_2 \oplus \alpha_3 y_3 \oplus \alpha_4 y_4$ doğrusal yaklaşımı x cinsinden

$$(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \cdot M \cdot x^{\top}$$

şeklinde ifade edilebilir. Burada $(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \cdot M$, girdi baytlarını oluşturan x için çıktı doğrusal yaklaşıma karşılık gelen girdi doğrusal yaklaşımını verecektir. Girdi yaklaşımı $(\beta_1 x_1 \oplus \beta_2 x_2 \oplus \beta_3 x_3 \oplus \beta_4 x_4)$ ise,

$$(\beta_1, \beta_2, \beta_3, \beta_4) = (\alpha_1, \alpha_2, \alpha_3, \alpha_4) \cdot M$$

olacaktır. AES şifrelemesinde matris çarpmaları soldan yapıldığı için girdi/çıktı yaklaşımlarının maskeleri

$$M^{\top} \cdot \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{pmatrix} = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{pmatrix}$$

şeklinde ifade edilebilir. M, bir MDS matris olduğu için M'nin transpozu olan $M^{\top}$ de bir MDS matristir. Sıfırdan farklı girdisi ve çıktısı arasında en fazla 3 koordinatı sıfır olabilir. Bu durumda M sütun karışma operasyonunun aktif olduğu bir doğrusal yaklaşımda toplam sekiz tane α_i ve β_i koordinatlarından en az beş tanesi sıfırdan farklı olacaktır. Diğer bir ifade ile, M'den önce ve M'den sonra uygulanan S kutularından en azından 5 tanesi aktif olacaktır. Benzer şekilde, 4 çevrimlik AES'in ilk iki çevriminde i tane aktif sütun karıştırma varsa diğer iki çevriminde de en azından $5 - i$ tane aktif sütun karıştırma olacaktır. Sonuç olarak, 4 çevrim AES'de en azından 5 aktif sütun karıştırma beklenir ki bu da en azından 25 aktif S-kutusu anlamına gelmektedir.

AES'in Bayt değiştirme katmanı olan S-kutusu özel bir tasarımdır ve doğrusal yaklaşımları için S-kutusunun LAT tablosunda görülen en düşük değer 112, en yüksek değer ise 144'tür. S-kutusunun herhangi girdi doğrusal kombinasyonunun herhangi çıktı doğrusal kombinasyonuna eşit olma ihtimali 1/2'den en fazla 2^{-4} gibi son derece düşük bir oranla sapmaktadır. Dolayısıyla, 4 çevrimlik herhangi bir doğrusal karakteristiğin sapması (1/2'den farkı) en fazla $2^{64} 2^{-4 \cdot 25} = 2^{-74}$ olacaktır. Bu da AES'in doğrusal saldırıya karşı ne kadar dayanıklı olduğunu göstermektedir.

9.8.4. Saldırının Güncel Uygulamaları ve Türevleri

Matsui doğrusal saldırı yöntemini yayınladıktan bir sene sonra kendisi 12 adet PC üzerinde veri hazırlamış, literatürde DES'in pratik olarak kırılmasını ilk kez başarıyla göstermiştir [89]. Deney için 15 çevrimlik bir yaklaşım kullanmış ve bu yaklaşımı hem şifreleme hem de şifre çözme yönünde uygulayarak ilk ve son çevrim anahtarlarından altışar bit elde etmiştir. Birer bit de doğrusal yaklaşımın şifreleme ve şifre çözme yönlerindeki sapmalarının işaretlerinden elde edilmiştir. Geri kalan anahtar bitleri kaba kuvvet saldırısı ile bulunmuş ve bu tarama, o dönemin PC'leri üzerinde on gün sürmüştür. Saldırıyı uygulamak için gerekli 2^{43} açık metin kapalı metin çiftini hazırlamak ise Matsui'nin 40 gününü almıştır [89].

Matsui'nin doğrusal saldırısının başarı oranı literatürde son derece yoğun bir şekilde tartışılmıştır. Selçuk başarı oranlarını hesaplama yöntemi öne sürmüş ve ayrıca başarı oranı arttırmak üzere saldırıda iyileştirmeler önermiştir [104]. Hermelin ve arkadaşları birçok doğrusal yaklaşımı bir arada değerlendirme için Selçuk'un anahtar sıralama ve avantaj kavramını baz alarak belli başlı istatistiksel dağılımlarda gerekli veri miktarını azaltmaya yönelik yöntem önermişlerdir [58]. Belirli başarı oranı için gerekli veri miktarını daha ayrıntılı hesaplama konusunda da çalışmalar mevcuttur [27,60].

Farksal kriptoanalizdeki farksal karakteristik ile diferansiyel arasındaki ilişkiye benzer ilişki doğrusal saldırı için de kurulmaya çalışılsa da kayda değer bir ilerleme sağlanamamıştır. Aynı açık metin şifreli metin doğrusal kombinasyonuna sahip olan, fakat farklı karakteristik yollar izleyerek farklı anahtar bitleri kombinasyonları elde edilmesi ile oluşturulan doğrusal kabukların analizleri ilk olarak Nyberg tarafından yapılmıştır [92]. Röck ve Nyberg, Matsui'nin 0-R saldırısı olan Algoritma 1'inde doğrusal kabuk etkisini analiz ettiler ve özellikle PRESENT gibi sıralı anahtar kullanan blok şifreleme algoritmalarında doğrusal kabuğun matematiksel yapısını oluşturdular [102]. Chen ve Wang ise doğrusal kabukların Matsui'nin Algoritma 1'ine olan etkisini incelediler [32]. Leander doğrusal karakteristiklerde kabuk etkisini tartışmış ve PUFFIN ile PRESENT algoritmalarında kabuk etkisini incelemiş-

tir [75]. Yakın zamanda yayınlanmış konferans bildirisinde SPN yapılarda doğrusal kabuğu oluşturan karakteristiklerin ihtimallerinin toplamlarının nasıl hesaplanacağı tartışılmıştır [51].

Matsui'nin tanımladığı Algoritma 2'de, ilgili her bir çevrim anahtarı için bütün girdi çıktı çiftlerinin doğrusal yaklaşımdaki değerleri hesaplanmaktadır ve denenen her bir anahtar için sapma bulunmaktadır. Collard ve arkadaşları FFT dönüşümü kullanılmasıyla bu aşamanın kayda değer derecede hızlandırılabileceğini gösterdiler [37]. Gutiérrez ve Naya-Plasencia matris tabanlı gelişmiş FFT tekniğiyle Algoritma 2'deki anahtar tarama aşamasını daha da hızlandırmayı başardılar [54].

9.9. FARKSAL KRİPTOANALİZ

9.9.1. Farksal Kriptoanalizin Genel Tanımı

Biham ve Shamir tarafından 1990 yılında DES şifreleme algoritmasına uygulanarak ilk olarak CRYPTO'90 konferansında [19] nolu bildiri ile ve daha sonra genişletilmiş hali [129] nolu makale ile literatüre kazandırılan bu kriptoanaliz tekniği, blok şifreleme algoritmalarına uygulanan en yaygın kriptoanaliz tekniklerinden biridir ve blok şifreleme algoritma tasarımlarında dikkate alınan analiz yöntemlerinin başında gelir. Bu kriptoanaliz tekniği doğrusal kriptoanaliz gibi istatistiksel bir yöntemdir, kullanılan açık/şifreli metinlerin anahtardan bağımsız belli bir olasılıkla belli bir özelliği göstermeleri keşfedilerek yapılan bir kriptoanaliz yöntemidir.

Bu kriptoanaliz yönteminin detaylarına geçmeden önce blok şifreleme algoritmalarına yapılan temel kriptoanaliz yöntemleri yaklaşımlarının birini hatırlatmak faydalı olur. Anlatılacak bu yaklaşıma ayıraç yöntemi yaklaşımı adı da verilebilir. Hatırlanacağı üzere blok şifreleme algoritmaları çevrim fonksiyonunun yinelemeli olarak uygulanmasından oluşur ve her bir çevrimde bir çevrim anahtarı kullanılır. Bu çevrim anahtarları da çevrim anahtar şeması kullanılarak şifreleme anahtarından elde edilir. Ayıraç yaklaşımına

dayalı kriptoanaliz yöntemleri şu şekilde çalışır. Şifreleme algoritması r çevrimden oluşmak üzere, öncelikle algoritmanın iç yapısı incelenerek $r-1$ çevrim için anahtardan bağımsız bir ayıraç bulunur. Bu ayıraçtaki özelliğin gerçekleşme olasılığının bu özelliğin rasgele olma olasılığından büyük olması gerekir. Aslında bu olasılık rasgele olasılıktan küçük olsa bile saldırı gerçekleştirilebilir fakat saldırının başarı olasılığı ve gerekli veri miktarı oldukça fazla olur.

Bazı blok şifreleme algoritmalarında belirli girdi farkına sahip açık metinlere karşılık gelen şifreli metinlerde oluşan fark, anahtardan bağımsız olarak beklenenden çok daha yüksek ihtimalle belirli bir değere sapabilir. Farksal kriptoanaliz yöntemi bu ayıracı esas alır. $r-1$ çevrimden oluşan algoritma parçasının çıktısı, P açık metni için Y, P' açık metni için de Y' ile temsil edilsin. Girdide sabit bir $\Delta P = P \oplus P'$ farkı varken, $r-1$ çevrim sonrasında belirli bir $\Delta Y = Y \oplus Y'$ olma olasılığı anahtardan bağımsız bir şekilde p olsun. Eğer p olasılığı çıktıda rasgele ΔY farkı görünme olasılığı olan n^{-1}'den (n algoritmanın blok boyu) büyük ise $(\Delta P, \Delta Y, r-1, p)$ bu algoritma için farksal bir ayıraç olur. Burada ayıraç bulunan çevrim sayısı $r-1$ yerine $r-2$, $r-3$, $r-i$ vb. çevrim sayıları da olabilir [127]. Ayrıca $r-i$ çevrimden oluşan algoritma parçası şifreleme algoritmasının ilk kısmı, son kısmı ya da ortada bir kısmı olabilir.

Bir blok şifreleme algoritması için $(\Delta P, \Delta Y, r-1, p)$ ayıracı bulmak başlı başına bir araştırma problemidir. Bu problemin genellikle en zor kısmı ΔP girdi farkının hangi olasılıkla ΔY çıktı farkına gittiğini hesaplamak, başka bir deyişle, p değerini bulmaktır. $(\Delta P, \Delta Y)$ diferansiyeli yerine her bir çevrimde muhtemel en yüksek olasılıkla oluşan farklar hesaplanır ve i-nci çevrim çıktısında X_{i+1} farkı oluşacak şekilde $\Delta P = X_1, X_2, \cdots X_{r-1} = \Delta Y$ farksal karakteristiği elde edilir. Çevrim bazında X_i farkının X_{i+1} farkına gitme olasılığı p_i olsun. Bu durumda p_i ihtimalini hesaplamak çok daha kolaydır. $(\Delta P, X_2, \cdots \Delta Y)$ farksal karakteristiğinin olasılığı her bir çevrimde oluşan farkların birbirlerinden bağımsız olduğu varsayımı altında $\prod_{i=1}^{r-1} p_i$

olacaktır. $\prod_{i=1}^{r-1} p_i$ olasılığının p olasılığından düşük olduğu aşikardır. Gerçekte p değeri, girdi farkı ΔP ve çıktı farkı ΔY olan bütün farksal karakteristiklerin ihtimallerinin toplamına eşittir. Diğer taraftan $(\Delta P, \Delta Y)$ diferansiyelini oluşturan milyarlarca karakteristik olabilir ve herbirinin ihtimalini hesaplamak son derece karmaşıktır. Ancak farksal kriptoanalizde bir $(\Delta P, \Delta Y)$ diferansiyelinin, olasılığına çok yakın olasılıkta bir farksal karakteristiği olduğu varsayılır ve bu farksal karakteristiğin olasılığı üzerinden hesaplamalar yapılır. Özellikle DES gibi farksal kriptoanalize karşı zayıf algoritmalarda bu varsayım son derece gerçekçidir.

Bir blok şifreleme algoritmasının ilk $r-1$ çevrimine bulunmuş yüksek ihtimalli girdi çıktı farkı r-nci çevrim anahtarını ele geçirmek için ayıraç olarak kullanılabilir. Ayıraç olasılığı ile ilişkili olarak yeterli sayıda (c saldırının başarısını etkileyen bir katsayı olmak üzere aralarında ΔP farkı olan $c \times p^{-1}$açık metin çiftleri seçilir. Bu açık metin çiftlerine karşılık gelen şifreli metin çiftleri, son çevrim için tahmin edilmiş bir çevrim anahtarı ile sadece bir çevrim çözülür. En temel saldırıda şifreleme anahtarı dolayısıyla da çevrim anahtarları bilinmediğinden son çevrim anahtarının alabileceği her bir değer için bu işlem yapılır. Saldırının gelişmiş sürümlerinde son çevrim anahtarlarını tek tek denemeye gerek yoktur. Son çevrimin geri çalıştırılması sonucu ΔY farkını sağlayan metin çiftleri sayılır ve p olasılığı ile karşılaştırılır. Doğru çevrim anahtarı için olasılığı p'ye yakın ve daha fazla çıkması beklenirken yanlış anahtar denemeleri için olasılığın, rasgele gelme olasılığına yakın olması beklenir, başka bir ifadeyle, ihtimalin n^{-1} civarında olması gerekir. Ancak yanlış anahtarlar için bu beklenen ihtimali hesaplayacak kadar veri yoktur. Diğer taraftan, bir anahtar adayının yanlış olduğuna karar vermek için beklenen ihtimali hesaplamaya da gerek yoktur. Eğer p ihtimali n^{-1}'den oldukça yüksekse ΔY farkını birkaç metin çiftinden sağlayan anahtar adayı çok yüksek olasılıkla doğru anahtar olacaktır. ΔY farkını veren metin çiftleri sayısının beklenen değeri doğru anahtar için c iken yanlış bir anahtar için sıfıra yakındır.

Farksal saldırıda hedef alınan çevrim anahtarlarını ele geçirmek için bu anahtarları tek tek denemeye gerek yoktur. Her bir şifreli metin çifti için $(r-i)$-nci çevrimde ΔY farkının oluşmasını sağlayan anahtarlar tespit edilir ve bir anahtar adayının bütün veriler içinde kaç kere ΔY farkını ürettiği sayılır. En yüksek sayıda ΔY farkını üreten anahtar büyük ihtimalle doğru anahtardır. Bu kriptoanaliz için gerekli veri miktarı sinyal-gürültü oranı ve farksal karakteristik ihtimali ile belirlenir. Gerçek veride $(r-i)$-nci çevrimde ΔY farkının oluştuğu çiftler sinyal olarak adlandırılır. Her bir sinyalde doğru anahtar kesin olarak bir kez sayılacaktır. Bunun dışında, gerçekte ΔY farkı oluşmasa da, analiz edilirken $(r-i)$-nci çevrimde ΔY farkı oluşturan anahtar adayları da gürültü olarak sayılacaktır. Bu tür farklara aldatıcı farklar denir. Bir aldatıcı fark birden fazla anahtar adayı önerebilir. Bu sayı ortalama olarak α olsun. Diğer taraftan verilmiş bir açık metin kapalı metin çiftinin hiçbir anahtar için $(r-i)$ -nci çevrimde ΔY oluşturamayacağı bazı durumlarda analiz etmeden tespit edilebilir. Bu girdiler hiç kullanılmadan çöpe atılır. Kullanılan veri oranına β densin. Bu durumda yanlış bir anahtar $\frac{c}{p}$ veri çifti için ortalama olarak $\frac{c\cdot\beta\cdot\alpha}{p\cdot 2^k}$ kere sayılacaktır. Burada k taranan anahtar bitleri sayısıdır. Her bir sinyal için doğru anahtar ise yaklaşık c kere sayılacaktır. Sonuç olarak sinyal gürültü oranı $\frac{p\cdot 2^k}{\beta\cdot\alpha}$ olarak ifade edilebilir. Atağın başarılı olması için bu oran birden büyük olmalıdır. Oran birden ne kadar büyük olursa c değeri o kadar düşük alınabilir ve sonuçta daha az veri ile saldırı gerçekleştirilebilir.

Bazı anahtar şemalarında çevrim anahtarlarından şifreleme anahtarının bazı bitleri direk olarak elde edilebilirken, bazı anahtar şemaları için bu o kadar kolay olamayabilir. Fakat zaten son çevrim anahtarının elde edilmesi demek, algoritmanın artık r çevrim yerine $r-1$ çevrime indirilmesi demek olacağından bu aşamadan sonra benzer saldırı, daha az veriyle uygulanarak sondan başa doğru tüm çevrim anahtarları teker teker elde edilebilir. $r-1$ çevrim ayıraç kullanılarak yapılan saldırılara $1R$ saldırıları denir. $r-1$ çevrim yerinde $r-2$ çevrim için ayraç bulunup son iki çevrimdeki çevrim anahtar-

ları elde edilmeye çalışılabilir. Bu saldırılara da $2R$ çevrim saldırıları denir. Benzer şekilde daha az çevrim için ayıraç bulunarak da saldırılar yapılabilir. Ayrıca $r-1$ ilk çevrimler yerine son çevrimler de olup saldırıda ilk çevrim anahtarı elde edilmek de istenebilir. Blok şifreleme algoritmasının tersinin de aslında bir blok şifreleme algoritması olduğu unutulmamalıdır.

Bu kriptoanaliz yönteminin en önemli adımlarından birisi yukarıda bahsi geçen iyi ayıracın/ayıraçların bulunmasıdır. Burada "iyi"den kastedilen, ayıracın gerçekleşme olasılığının olabildiğince yüksek olmasıdır ki gerekli veri sayısı o kadar az olabilsin ve o kadar az işlem ile saldırı yapılabilsin. Bu amaçla algoritmadaki yapı taşları tek tek incelenerek farksal özellikleri analiz edilir.

Bir blok şifreleme algoritmasının yapı taşları genelde YaDa işleminden oluşan çevrim anahtarı eklemekten, doğrusal olmayan yapıları oluşturan ve algoritmanın karıştırma katmanı olan yerini alma kutularından (AES şifreleme algoritmasında "bayt değiştirme" işlemi) ve doğrusal fonksiyonlardan oluşan yayılım katmanından ibarettir. Farksal saldırıda belirli farkların, girdiler bilinmediği durumda davranışları sadece doğrusal olmayan yapı taşlarında ihtimallere dayalıdır. Doğrusal yapı taşları ise girdilerden bağımsız olarak girdi farklarından 1 olasılıkla çıktı farkları oluştururlar.

Anahtar ekleme için genelde YaDa işlemi kullanılırken bunun yerine modülo toplama işlemlerinin tercih edildiği algoritmalar da bulunur. YaDa işleminde girdi farkı çıktıda da 1 olasılıkla gerçekleşir: $(x \oplus k) \oplus \big((x \oplus \Delta x) \oplus k\big) = \Delta x$. Burada Δx farkı x girdisinden ve k anahtarından bağımsız olarak anahtar ekleme işleminden değişmeden çıkmaktadır.

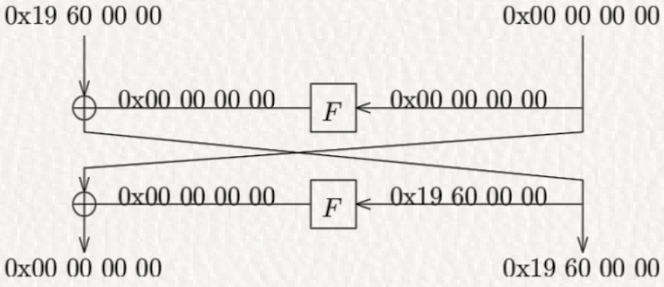
Yerini alma kutuları doğrusal olmayan yapılar oldukları için bu kutuların girdilerindeki herhangi bir fark değerini, çıktıda 1 olasılıkla belirli bir farka götürmesi beklenmez. Bu yapılar için bir fark dağılım tablosu oluşturulur. Bu tablo aslında girdi çiftleri bilinmezken ve rastgele gelirken her bir girdi farkı için çıktı farklarının hangi olasılıkla göründüğünü içeren bir tablodur.

Şifreleme algoritmasının kendisi için yüksek ihtimalli girdi çıktı farkı bulunabilmesi için bir yandan çevrimler boyunca olabildiğince az sayıda yerini

alma kutusuna girdi farkı verilmesi, bir yandan da girdisine fark verilen yerini alma kutularında olasılığı yüksek olan girdi-çıktı farkları seçilmesi yaklaşımı benimsenir. Yayılım katmanı ise doğrusal bir yapıya sahiptir ve bir farkın yayılım katmanından çıktısı girdilerden bağımsız olarak hesaplanabilir. Burada dikkat edilmesi gereken husus, girdi ve çıktıda farklara sahip bit sayılarının olabildiğince az olması veya yayılım katmanın öncesinde ve katmandan sonra uygulanan aktif s-kutularının ("yerini alma kutuları"nın) sayısının olabildiğince az olmasının sağlanmasıdır. s-kutularının aktif olması, bu kutuların girdilerinde fark olmasıdır. Bu sayının az olması demek ayıracın olasılığının yüksek olması demek olur çünkü yerini alma kutusu girdisinde fark yoksa ($\Delta = 0$ ise) çıktısında da fark 1 olasılıkla 0'dır. Bu prensipler dikkate alınarak $r-1$ çevrimlik algoritma için her bir çevrimde her bir yapı taşının girdi çıktı farkları belirlenir ve ayıraç olasılığı aktif yerini alma kutuları için girdi-çıktı fark olasılıklarının çarpımı olarak hesaplanır. Bu değer $r-1$ çevrimlik farksal karakteristiğin olasılığıdır.

9.9.2. Saldırının DES ve AES Algoritmalarına Uygulanması

Farksal kriptoanaliz yöntemi ile 16 çevrim DES algoritmasının 15-çevrim farksal karakteristik ve $1R$ saldırısı ile 2^{57} metin çifti kullanılarak kırılabileceği [19] numaralı makale ile gösterildi. Bu çalışmada Şekil 9.9 ile verilen iki çevrimlik yinelemeli farksal karakteristik tanıtıldı ve bu iki çevrimlik karakteristiğin ard arda bitiştirilmesi ile daha fazla çevrimler için farksal karakteristik elde edilebileceği gösterildi.



Şekil 9.9. 2-Çevrim Yinelemeli Farksal Karakteristik

Şekil 9.9 ile verilen karakteristiğe dikkat edilecek olursa ilk çevrimde F girdisinde bir fark bulunmadığı için bu çevrimin çıktısında farkın sıfır olma olasılığı 1 olur. İkinci çevrimde ise F girdisindeki fark olan bitler 3 farklı "yerini alma kutusu" girer. Bu kutuların fark dağılım tablosu kullanılarak girdi-çıktı farklarının gerçekleşme olasılıkları çarpıldığında F çıktısında farkın görülme olasılığının $1/234$ 'e eşit olduğu görülür. Dolayısı ile bu 2 çevrim farksal karakteristiğin olasılığı $1/234$'tür. Ayrıca dikkat edilecek olursa ikinci çevrim sonucunda kolların yer değiştirme işlemi uygulanınca 2-çevrimlik kısmın çıktısındaki fark ile girdisindeki farkın aynı olduğu görülür. Bu nedenle bu iki çevrimlik farksal karakteristik ard arda bitiştirilerek çok sayıda çevrim için karakteristik elde edilebilir. Örneğin 16-çevrim DES'e 1R saldırısı yapmak için 15-çevrimlik farksal karakteristik gerektiğinde bu iki çevrimlik karakteristik 7 kez ard arda eklenerek 14-çevrim karakteristik elde edilebilir ve 1 çevrim olarak da 2 çevrimlik karakteristiğin ilk çevrimi eklendiğinde toplamda 15-çevrimlik bir karakteristik elde edilmiş olur. Bu 15-çevrimlik karakteristikte 7 adet F fonksiyonu girdisinde fark olacağı için 15 çevrimlik karakteristiğin olasılığı çevrimlerdeki s-kutusu girdilerinin birbirinden bağımsız olduğu varsayımı altında $(1/234)^7 \approx 2^{-55.1}$ olur.

Farksal kriptoanaliz DES'in dışında birçok, hatta hemen hemen tüm blok şifreleme algoritmalarına uygulanabilir. AES algoritması da bu algoritmalardan biridir. AES ve benzeri kelime tabanlı şifreleme algoritmalarında, bu algoritmaların farksal kriptoanalizi karşı dayanıklılığını ölçmenin genel olarak, DES ve benzeri bit tabanlı algoritmaların dayanıklılığını ölçmekten daha kolay olduğu söylenebilir.

Doğrusal kriptoanalizin AES algoritmasına uygulanması önceki bölümde anlatılmıştı. Farksal kriptoanaliz için de benzer yaklaşım geçerlidir. Algoritmada kullanılan MDS yapısının dallanma sayısı 5'tir; sayısal olarak 4 adet 8-bit girdi ve 4 adet 8-bit çıktı için eğer girdi farkı sıfır değilse toplam 8 adet 8-bit değerden en az beşinin aktif olduğu kesindir. MDS matrisi öncesi ve sonrasında bayt değiştirme (yerini alma kutuları) kullanıldığı için tek bir MDS matrisinin aktif olması en az 5 adet bayt değiştirme kutusunun aktif olması anlamına gelir. AES algoritmasında kullanılan satır kaydırma işlemi

de dikkate alındığında, doğrusal kriptoanalizde yapıldığı gibi 4 çevrimde en az 5 sütun karışma ve dolayısıyla en az 25 aktif s-kutusu olduğu görülür. AES bayt değiştirme kutusunun farksal dağılım tablosuna bakıldığında bayt değiştirme kutusu aktif iken olasılığı en fazla 2^{-6} olduğu tespit edilebilir. Bu nedenle en şanslı durumda bile (bu 2^{-6} olasılığa sahip girdi çıktı farkları kullanıldığı durum) 5 çevrimlik bir farksal karakteristiğin olasılığının $(2^{-6})^{25} = 2^{-150}$ olduğu görülür.

Aslında 4 çevrimlik AES farksal karakteristiklerinin olasılıklarının, beklenen değer olan 2^{-128}'den küçük olmasının nedeni, diferansiyel olasılıkların göz ardı edilmesi ve hesaplamada s-kutusu girdilerinin birbirinden bağımsız olduğunun varsayılmasıdır; halbuki girdiler gerçekte bağımsız değildir. Aynı zamanda bu üstten sınır 4 çevrimlik bir farksal karakteristik olasılığı için verilmiştir. Ancak saldırıda asıl sinyal gürültü oranını belirleyen aynı girdi ve çıktı farkı veren tüm karakteristiklerin olasılıklarının toplamıdır. Başka bir ifadeyle, 4 çevrimlik AES diferansiyellerinin olasılığı saldırıda ya da güvenlik ispatında dikkate alınmalıdır. Bu olasılıkların en büyüğüne MBDO (Maksimum Beklenen Diferansiyel Olasılığı) denir. Canteaut ve Roué, YPA yapılarının MBDO hesabı için yöntem önermişlerdir [30]. AES'de 4 çevrimlik bir diferansiyeli oluşturan milyonlarca karakteristik vardır. Her biri en az 25 aktif s-kutusu içerse ve olasılığı 2^{-150}'den küçük olsa da bu olasılıkların toplamı 2^{-128}'in çok daha üstünde olabilir. Park ve arkadaşları AES'in 4 çevrimlik MBDO'sunun 2^{-111} ile sınırlandırıldığını göstermişlerdir [95].

DES'in farksal kriptoanalizinde MBDO hesabı yapmaya gerek yoktur. DES'in 15 çevrimi için verilen $(1/234)^7$ olasılığa sahip karakteristik, dominant bir karakteristiktir ve aynı girdi çıktı farkına sahip diğer karakteristiklerin olasılıkları çok daha küçüktür. Dolayısıyla DES için diferansiyel olasılığının da hemen hemen $(1/234)^7$ civarında olduğunu söylemek yanlış olmaz.

Biham ve Shamir'in DES'in farksal kriptoanalizi için [19] nolu bildiride kullandığı argümanlar ve aktif S-kutusu sayma yöntemi AES'in 4-5 çevrim gibi

az sayıda çevriminin farksal analizi ve diferansiyel olasılık hesabı için çok da geçerli olmayabilir. Fakat AES'in 4 çevrimde en az 25 aktif s-kutusuna sahip olması şunu göstermektedir: AES algoritması farksal saldırıya karşı özel olarak tasarlanmış yapı taşları içermektedir ve YPA yapılarında saldırıya karşı güvenliğin en önemli şartı olan aktif s-kutusu sayısını olabildiğince yüksek tutulmasına özen gösterilmiştir. Sonuç olarak saldırıda kullanılabilecek bir 5-çevrimlik farksal analiz bulunamamaktadır. AES'e yapılan farksal kriptoanaliz detayları için tasarım felsefesinin ve güvenlik analizlerinin detaylı anlatıldığı [43] nolu kaynak incelenebilir.

9.9.3. Saldırının Türevleri

Diğer kriptoanaliz ve saldırı yöntemlerinde olduğu gibi farksal kritoanaliz yöntemi de zamanla geliştirilmiştir. Ayrıca farksal kriptoanalizden türemiş birçok farklı kriptoanaliz yöntemi vardır. Bunlardan en belli başlıları doğrusal-farksal kriptoanaliz, budanmış farksal kriptoanaliz, imkansız farksal kriptoanaliz, bumerang saldırısı ve dikdörtgen saldırısıdır.

1994 yılında Langford ve Hellman tarafından geliştirilen doğrusal-farksal kriptoanaliz; farksal ve doğrusal kriptoanaliz yöntemlerinin birleştirilmesi ile elde edilmiş hibrit bir yöntemdir [74]. Bu yöntemde öncelikle algoritma iki parçaya ayrılır ve ilk parça için farksal karakteristik ve son parça için ise doğrusal karakteristik bulunur. Algoritmanın ilk i çevrimi için p olasılığı ile bir farksal karakteristik ve geri kalan $r-i$ çevrimi için $\frac{1}{2}+q$ olasılığı ile bir doğrusal karakteristik var ise, belirli bir fark verilmiş açık metinlere karşılık gelen şifreli metinlerde doğrusal karakteristikteki yaklaşımların eşit olma olasılığı $\frac{1}{2}+2pq^2$ olacaktır. Dolayısıyla yaklaşık $O(p^{-2}q^{-4})$ seçili açık metin çifti olması durumunda, karşılık gelen şifreli metin çiftlerinde doğrusal yaklaşımların eşitliği ayıraç olarak kullanılabilir. Bu saldırının başarılı olabilmesi için hem farksal karakteristiğin oldukça yüksek olasılıkla olması, hem de doğrusal karakteristiğin sapmasının son derece büyük olması beklenir. Bu nedenle, ilk çevrimleri hem farksal kriptoanalize karşı hem de doğrusal kriptoanalize karşı son derece zayıfken bir iki çevrim eklenmesiyle hızla

güçlenen algoritmalara doğrusal-farksal saldırı, doğrusal saldırıdan da, farksal saldırıdan da daha başarılı bir şekilde uygulanabilir. Biham ve arkadaşları saldırıyı geliştirdiler [23]. Razoolzadeh ve arkadaşları tam çevrim TODO'yu kırmak için doğrusal-farksal saldırıyı kullandılar [99]. Ayrıca bu saldırı yöntemi dizi şifreleme algoritmalarına da uygulanabilmektedir [117].

Lars Knudsen tarafından 1994 yılında tanıtılan budanmış farksal kriptoanaliz, farksal kriptoanalizin değişik bir versiyonudur [70]. Bu kriptoanaliz yönteminde, farksal kriptoanaliz yöntemindeki gibi kelimelerde (bit gruplarında) farkın belli bir değere eşit olmasından ziyade, girdide belirli bir fark kümesinden alınan açık metin çiftine karşılık gelen şifreli metin çifti arasındaki farkın belirli bir fark kümesinin elamanı olup olmadığına bakılır. Eğer çıktı farkının ilgili kümede olma olasılığı beklenen değerden yüksek ise saldırıda ayıraç olarak kullanılabilir. Kelime tabanlı blok şifreleme algoritmalarında genellikle en iyi budanmış karakteristikler belli kelimelerde fark olup olmamasının analiz edilmesiyle bulunmuş karakteristiklerdir. Bu nedenle budanmış farksal kriptoanalizin bazı kaynaklarda bu çerçevede değerlendirildiğini görmek mümkündür.

Örneğin, AES şifreleme algoritmasının çevrim girdisinde en soldaki 8-bitlik değerde fark varken çevrim çıktısındaki 8-bitlik değerlerde fark olup olmaması dikkate alınır. Klasik farksal kriptoanalizde ise en soldaki 8-bitlik değerde belli bir fark varken çevrim çıktısındaki 8-bitlik değerlerde beklenen farkın görünme olasılığı söz konusuydu. Bu açıdan bakılınca klasik farksal kriptoanalizde bu bir çevrim için belli bir olasılıkla farksal karakteristik sağlanırken, budanmış farksal karakteristiğin %100 olasılıkla sağlandığı görülür. Bunun nedeni, bayt değiştirme kutusu girdisinde fark varken çıktısında da kesin olarak fark olmasıdır. Satır kaydırma işlemi sadece 8-bitlik değerlerin yerini değiştirdiği için 1 olasılık ile karakteristik devam eder. Sütun karıştırma işleminde ise girdide sadece bir 8-bitlik değerde fark olduğu için MDS özelliğinden dolayı çıktıdaki tüm 8-bitlik değerlerde 1 olasılıkla fark olur. Böylece bu bir çevrimlik budanmış farksal karakteristiğin olasılığı 1 olur. Fakat sütun karıştırma işleminde bir yerine birden fazla 8-bit değerde fark olması durumunda az önce belirtilen gibi çıktıdaki tüm 8-bitlerde kesin-

likle fark olur denemez. Bu nedenle sütun karıştırma işlemi için bir tablo oluşturulur. Bu tablo girdideki aktif 8-bit durumları için çıktıdaki 8-bitlerin aktiflik durumunun gerçekleşme olasılığını içerir.

Özetle budanmış farksal karakteristikte fark değerinden ziyade fark olup olmaması önemlidir. Farksal kriptoanalizde, farksal karakteristiğin olasılığının n blok boyu olmak üzere n^{-1} 'den büyük olması gerekir ki karakteristiği sağlayan birkaç tane açık metin çifti elde edilebilecek kadar yeterli veri olsun. Fakat budanmış farksal kriptoanalizde bu şart geçerli değildir. Böylece daha düşük olasılıklı budanmış farksal karakteristikler de saldırıda kullanılabilir hale gelir. Açık metinler arasındaki fark sabitlenmediği için daha fazla açık metin çiftleri oluşturulabilir. Bu nedenle klasik farksal saldırıyla karşılaştırıldığında budanmış farksal saldırıda çok daha fazla veri oluşturulabilir. Farklı karakteristikler için veri miktarı 128 bitlik bir blok şifrelemede 2^{200}'lere kadar çıkabilir.

Örneğin 128-bit blok boyuna sahip ve 8-bit kelime tabanlı (AES) bir şifreleme algoritması ele alınsın ve budanmış farksal karakteristikte açık metinler arasında sadece en soldaki 8-bitlik değerde fark olsun ve diğer 8-bit değerler bütün açık metinlerde aynı olsun. En soldaki 8-biti 0 olan 2^{120} adet açık metin vardır. Bu her bir açık metin için şöyle bir küme oluşturulabilir: sağdaki 120 biti sabit değiştirilmeden sadece soldaki 8 biti değiştirilerek elde edilen açık metinler kümeye eklenir. Böylece 256 açık metinden oluşan 2^{120} adet küme elde edilir. Her bir küme kullanılarak ise $256 \times 255/2 \approx 2^{15}$ adet açık metin çifti elde edilir. Bu çiftlerin özelliği sadece en soldaki 8-bitlik değerde fark olması ve diğer bitlerde fark olmamasıdır. Böylece 2^{120} küme olduğu için toplamda 2^{135} adet istenilen açık metin çiftleri elde edilebilir. Bu kadar açık metin çifti olduğu için de beklenen ihtimalden 2^{-134} olasılığa kadar sapan bir budanmış farksal karakteristik saldırıda kullanılabilir. Bu olasılığa sahip bir klasik farksal karakteristik saldırıda kullanılamaz çünkü bu karakteristiği sağlayan bir açık metin çiftinin var olması çok düşük bir olasılıktır.

Budanmış farksal kriptoanalizin birçok kriptoanaliz ile korelasyonu olduğu yayınlardan anlaşılmaktadır. Bao ve arkadaşları AES'in 3 çevrimlik integral

ayıracının 5 çevrimlik budanmış farksal karakteristiğe nasıl genişletilebileceğini gösterdiler [13]. Blondeau ve Nyberg budanmış farksal saldırı ile çok boyutlu doğrusal saldırı arasındaki ilişkiyi açıkladılar ve budanmış farksal saldırının çok boyutlu doğrusal saldırıdan daha üstün olduğunu gösterdiler [26]. Bu ilişkiyi içeren benzer bir çalışmada Sun ve arkadaşları karakteristik bulan tarama yöntemi önermiştir [107]. Li ve arkadaşları ortada buluşma saldırısı tekniği ile budanmış farksal karakteristik oluşturmayı CLEFIA ve Camellia üzerinde gösterdiler [76]. Moghaddam ve Ahmadian budanmış farksal karakteristik bulmak için karma tamsayı doğrusal programlama ile tarama yöntemi önerdiler ve bu yöntemleriyle Midori, SKINNY ve CRAFT'a budanmış farksal karakteristikler buldular [90]. Wu ve Wang da kelime tabanlı blok şifreleme algoritmalarına budanmış farksal karakteristik bulan otomatik tarama yöntemi geliştirdiler [118].

Budanmış farksal kriptoanaliz blok şifreleme algoritmalarının yanı sıra özet alma algoritmalarının güvenlik seviyesini ölçmek için sıklıkla kullanılan bir kriptoanaliz yöntemidir. SHA-1 özet alma algoritmasına uygulanması ve yeni bir özet alma algoritması olan Poseidon algoritması tasarımında güvenlik analizinde kullanılması örnek olarak verilebilir [69,125].

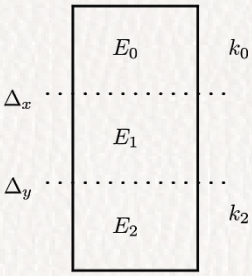
Farksal kriptoanaliz yönteminin diğer türevleri olan bumerang, imkansız farksal, dikdörtgen kriptoanaliz yöntemleri bu bölümünün ilerleyen başlıklarında ayrıntılı olarak anlatılmıştır.

9.10. İMKANSIZ FARKSAL KRİPTOANALİZ

9.10.1. Saldırının Genel Tanımı

İmkansız farksal ve benzeri olan ortada buluşamama saldırıları (Miss in the Middle) 1999 yıllarında geliştirilmiştir [21]. Bu saldırıda, saldırının yapılacağı şifreleme algoritması $E = E_2 \circ E_1 \circ E_0$ olmak üzere 3 parçaya ayrılır (bkz. Şekil 9.10). Ayıraç E_1'dedir. E_0 ve E_2'deki anahtarlar taranarak ayıraç kontrol edilir.

E_1 kısmı için bu kısımdaki anahtar ne olursa olsun, gerçekleşme olasılığı 0 olan bir girdi-çıktı çifti farkı (Δ_x, Δ_y) bulunur. Bulunan bu çifte imkansız fark çifti denir ve aralarında Δ_x farkı bulunan açık metin çiftine E_1 işlemi uygulanması sonucu oluşan açık metinler arasındaki farkın Δ_y olması imkansızdır. Verilen açık-şifreli metin çiftleri kullanılarak ve E_2 ve E_0'da kullanılan anahtar parçaları (sıra ile k_0 ve k_2) taranarak E_1'in girdi ve çıktı farkları hesaplanır.



Şekil 9.10. İmkansız Farksal Kriptoanaliz

Aday anahtar kümesinde başlangıçta (k_0, k_2) için tüm olası değerler bulunur. Bir girdi çıktı farkının tahmin edilen bir (k_0, k_2) anahar değeri ile, E_1 için bulunan imkansız fark çiftine gitmesi durumunda, tahmin edilen (k_0, k_2) değerinin yanlış olduğuna karar verilir ve bu değer aday anahtar kümesinden çıkarılır.

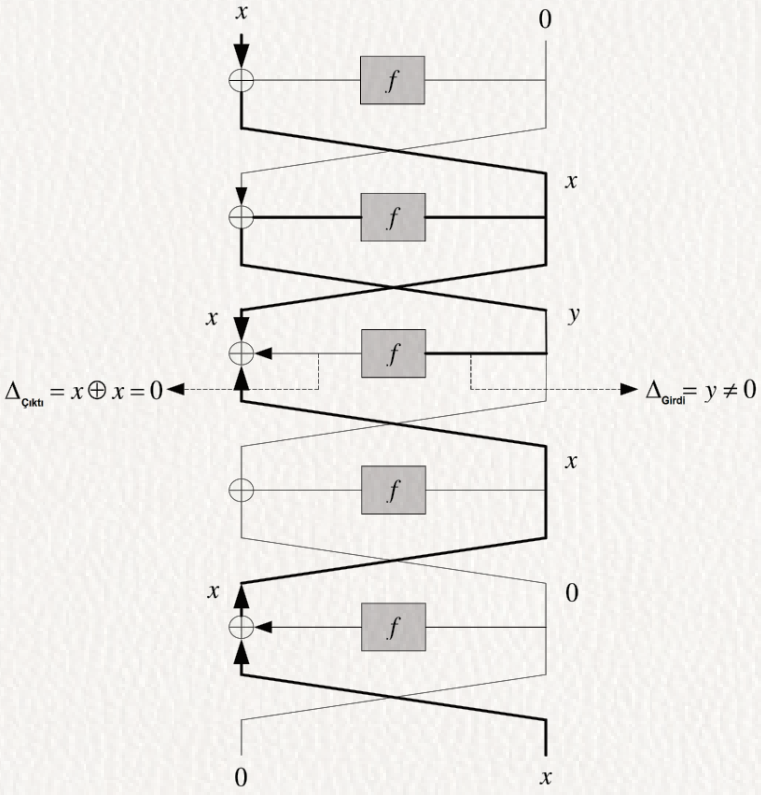
Sadece doğru anahtar kalıncaya kadar yanlış olduğu tespit edilen tüm anahtarların aday anahtar kümesinden çıkartılması için gerekli metin çiftleri şu şekilde hesaplanır: Tahmin edilen anahtar bit sayısı k rasgele bir metin çifti için E_1'in girdi ve çıktısında bulunan imkansız farkın sağlanma olasılığı p ve gerekli metin çifti sayısı m olsun. Bu durumda $(1-p)^m \times 2^k \leq 1$ olması gerekir ki aday anahtar kümesinde sadece doğru anahtar kalsın. Dolayısıyla, m için aşağıdaki koşul elde edilir:

$$m \geq \frac{k}{p} \times ln2$$

Saldırıda kullanılan bellek miktarı ise tüm aday anahtarlarının saklandığı kümenin büyüklüğüdür.

9.10.2. Saldırının Feistel Yapısına Uygulanması

Feistel ağında birebir F fonksiyonu kullanıldığında, F fonksiyonunu oluşturan S kutuları, doğrusal katman ya da diğer her türlü yapı taşından bağımsız olarak, 5 çevrimlik genel karakteristik bulunmaktadır. Bu karakteristik Şekil 9.11'de gösterilmiştir.



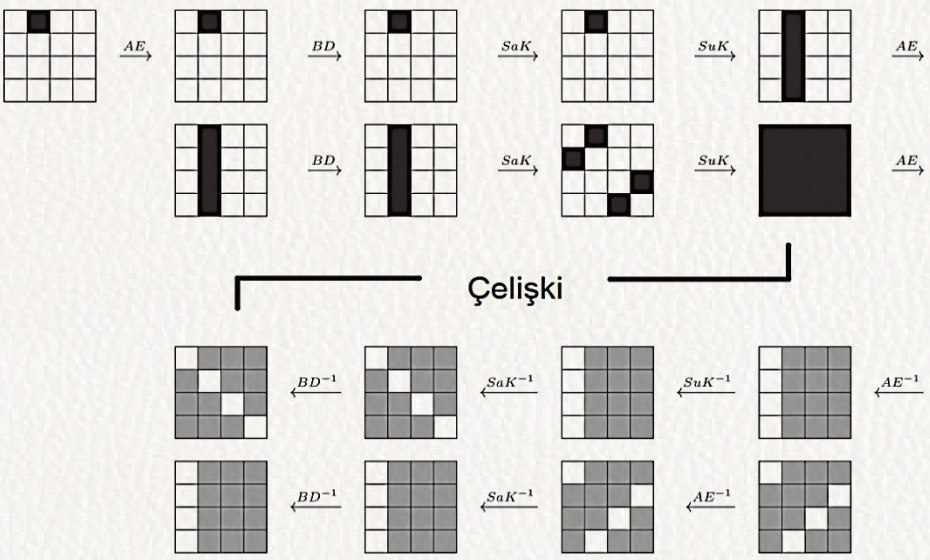
Şekil 9.11. Feistel 5 Çevrim İmkansız Farksal Karakteristik

Eğer girdide $(x, 0)$ farkı verilirse çıktıda $(0, x)$ farkı hiçbir zaman görülmemektedir. 5 çevrimlik yapının 3. çevriminde F fonksiyonu birebir olduğu için sıfırdan farklı y farkının sıfırdan farklı bir farka gitmesi gerekmektedir. Fakat şekilde x farkları eşit olduğu için F fonksiyonunun çıktısının 0 olduğu

görülmektedir. Bundan dolayı çelişki elde edilir. Birebir F fonksiyonu kullanan Feistel yapılara imkansız atak 5 çevrimlik bu ayraç kullanılarak çevrim fonksiyonundan bağımsız olarak yapılabilmektedir. Eğer F fonksiyonu birebir değil ise bu durumda 4 çevrimlik imkansız karakteristik bulunabilir.

9.10.3. Saldırının AES Algoritmasına Uygulanması

AES gibi YPA yapıdaki şifreleme algoritmaları için de Feistel yapılarda olduğu gibi genel bir imkansız karakteristik yazmak mümkündür. AES'de kullanılan sütun karıştırma matrisinin dallanma sayısı 5 olduğu için Şekil 9.12'de gösterilen 4 çevrimlik imkansız karakteristik elde edilmektedir. Girdi olarak 1 tane S-kutusuna fark verildiğinde çıktıda köşegen ya da ters köşegenlerin hepsinde fark olmama olasılığı 0'dır. Çünkü üstten 2 çevrim sonunda tüm kutularda fark olurken, alttan 2 çevrim sonunda köşegenler boş kalmaktadır. Bu karakteristik kullanılarak AES-128, AES-192 ve AES-256'ya imkansız farksal saldırılar uygulanmıştır.



Şekil 9.12. AES 4 Çevrim İmkansız Farksal Karakteristik

9.10.4. Saldırının Güncel Uygulamaları ve Türevleri

İmkansız farksal atak, ilk olarak Biham, Biryukov and Shamir tarafından [21] makalesinde önerilmiş ve Skipjack blok şifreleme algoritmasının kriptoanalizinde kullanılmıştır. Sonrasında popüler bir saldırı haline gelen imkansız farksal kriptoanaliz; MISTY, Camellia, ARIA ve CLEFIA gibi bir çok farklı blok şifreleme algoritmasına uygulanmıştır [49,80,110,115]. Bu saldırının uygulandığı algoritmalar arasında en önemlisi kuşkusuz AES şifreleme algoritmasıdır. AES için yapılan ilk imkansız farksal kriptoanaliz 2000 yılında Biham and Keller tarafından yayınlanan [18] makalesinde yer almıştır. Bu makalede 5 çevrime kadar AES'e imkansız farksal atak uygulanmıştır. 2001 yılında [35] makalesinde, atak geliştirilmiş ve 6 çevrime çıkarılmıştır. AES'in imkansız farksal atağı üzerine yazılan ve diğer ataklara ışık tutan makale Phan [96] tarafından anahtar şemasındaki zayıflık kullanılarak AES'in 7 çevrimine yapılmıştır. Phan'ın makalesindeki karakteristik kullanılarak AES'e yapılan imkansız farksal atak geliştirilmiş ve [10,79,82] gibi farklı makaleler yayınlanmıştır. Yapılan ataklar Tablo 9.1'de bulunmaktadır. İmkansız farksal saldırı hakkında algoritma özelinde yayınlanan makaleler olduğu gibi genel Feistel yapısına [29,104] ve SPN yapısına [46,77] uygulanan kriptoanalizler de bulunmaktadır. Saldırının ilintili anahtar ve bumerang ile birlikte uygulanan türevleri de bulunmaktadır.

Tablo 9.1. 7 Çevrim AES-192 ve AES-256 Veri-Zaman Karmaşıklıkları

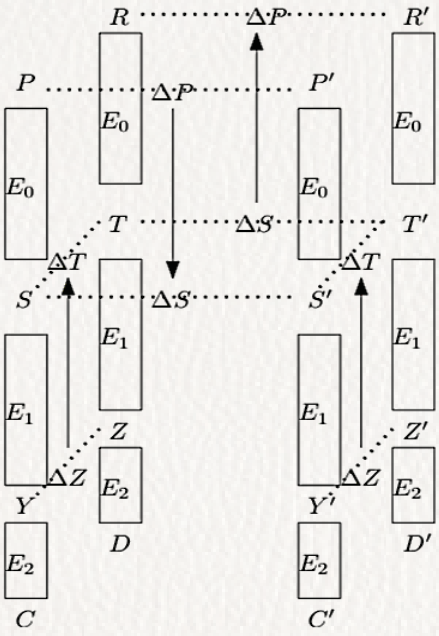
Algoritma	Data	Zaman	Makale
7-Çevrim AES-192	$2^{117.5}$	2^{121}	[10]
	$2^{113.8}$	$2^{118.8}$	[82]
	2^{92}	$2^{186.2}$	[96]
	2^{92}	2^{162}	[126]
	$2^{91.2}$	$2^{139.2}$	[79]
7-Çevrim AES-256	$2^{115.5}$	2^{119}	[126]
	$2^{113.8}$	$2^{118.8}$	[79]
	$2^{92.5}$	$2^{250.5}$	[96]
	2^{92}	2^{163}	[79]

9.11. BUMERANG VE DİKDÖRTGEN SALDIRILARI

9.11.1. Saldırının Genel Tanımı

Farksal kriptoanalizin bir türevi olan bumerang saldırı yöntemi 1999 yılında Wagner tarafından tanıtıldı [113]. Hatırlanacağı üzere, farksal kriptoanaliz yönteminde, algoritmanın bir kısmı için, örneğin $r-1$ çevrimi için, yüksek gerçekleşme olasılığına sahip bir farksal ayıraç bulunur ve bu ayıraç kullanılarak son çevrimdeki çevrim anahtarı ele geçirilmeye çalışılır. Fakat çoğu durumda algoritmanın az sayıda çevrimi için yüksek olasılıklı ("iyi") ayıraç bulunabilirken fazla çevrimlerden oluşan kısmı "iyi" bir ayıraç bulmak mümkün olmayabilir. Bunun nedeni, çevrim sayısı arttıkça anahtar bitleri ve açık metin bitlerinin yayılımının hızla artmasıdır. Bu tip algoritmalarda klasik farksal kriptoanaliz uygulanması yerine bumerang ve dikdörtgen saldırıları uygulamak mümkün olabilir. Bu bölümde öncelikle bumerang saldırı yöntemi tanıtılıp ardından dikdörtgen saldırısının nasıl yapılabileceğinden bahsedilecektir.

Bumerang saldırı yönteminde, ayıracın bulunacağı algoritma kısmı iki parça olarak ele alınır. Her bir parça için birbirinden bağımsız farksal ayıraçlar bulunur. Böylece çok çevrimden oluşan kısım için farksal ayıraç bulmak yerine daha az çevrimden oluşan iki farklı kısım için yüksek olasılığa sahip farksal ayıraçlar bulunabilir. Bu durumda Şekil 9.13'te görüldüğü gibi algoritma toplamda 3 parçaya ayrılmış olur. E_0 algoritmanın ilk kısmı için $(\Delta P, \Delta S)$ farksal ayıracı bulunur, E_1 algoritmanın ikinci kısmı için $(\Delta T, \Delta Z)$ farksal ayıracı bulunur ve E_2 kısmında kullanılan çevrim anahtarı veya çevrim anahtarının bir kısmı ele geçirilmeye çalışılır. Bu noktada şunu hatırlatmak faydalı olabilir, bu örnekte de yine son çevrim anahtarını ele geçirme senaryosu kullanılmıştır fakat son çevrim yerine ilk çevrim anahtarı ya da ilk ve son çevrim anahtarları veya ilk ve son birkaç çevrim anahtarlarını ele geçirme şeklinde de saldırı yapılabilir. Bunun için ayıraçların algoritmanın ilk kısmı yerine son kısmında veya algoritmanın ortasındaki kısım için bulunması gerekir.



Şekil 9.13. Bumerang Saldırısı

E_0 için bulunan farksal ayıracın gerçekleşme olasılığı p_0, E_1 için bulunan farksal ayıracın gerçekleşme olasılığı is p_1 olsun. Bu durumda ΔP farkına sahip (P, P') açık metne karşılık gelen (S, S') çiftinin ΔS farkına sahip olma olasılığı p_0'dır. (S, S') çiftinin E_1 çıktısı (Y, Y') çifti olsun. Y değerine ΔZ farkı eklenerek Z, Y' değerine de ΔZ farkı eklenerek Z' elde edilsin. Z ve Z' değerleri için E_1'in tersi uygulanarak T ve T' elde edilsin. Y ve Z arasındaki fark ΔZ olduğu için S ile T arasındaki farkın ΔT olma olasılığı p_1'dir. Benzer şekilde S' ile T' arasındaki farkın da ΔT olma olasılığı p_1'dir. $S \oplus S' = \Delta S$, $S \oplus T = \Delta T$ ve $S' \oplus T' = \Delta T$ olduğu durumda (bu durumun olma olasılığı $p_0 \times p_1 \times p_1$) $T \oplus T' = \Delta S$ olur. T ve T''ne karşılık gelen açık metinler R ve R' olsun. E_0'daki karakteristikten dolayı T ile T' arasında ΔS farkı varken R ve R' arasında ΔP farkı olma olasılığı p_0'dır. Toplamda bakılacak olursa aralarında ΔP farkı olan P ile P''nün E_0 ve E_1 şifreleme sonucunun ΔZ farkı uygulandıktan sonra geri çözülmesi ile elde edilen R ile R' arasındaki farkın

da ΔP olma olasılığı ${p_0}^2 \times {p_1}^2$ olur. Bu olasılık çoğu durumda E_0 ve E_1'nin bütünü için bulunabilecek bir farksal ayıracın olasılığına göre daha büyük olabilir.

Yukarıdaki paragrafta bir bumerang ayıracının nasıl oluşturulabileceğinden bahsedildi. Şimdi bu ayıraç kullanılarak saldırının nasıl yapılabileceği anlatılacaktır. Saldırı adımları şu şekildedir. c bir sabit sayı olmak üzere $c \times ({p_0}^2 \times {p_1}^2)^{-1}$ adet ΔP farkına sahip (P, P') çiftleri alınır. Son çevrim anahtarının her bir olası değeri için şu işlemler uygulanır. (P, P') çiftlerine karşılık gelen şifreli metin çiftleri (C, C') öğrenilir. Tahmin edilen son çevrim anahtarı kullanılarak E_2 kısmının tersi çalıştırılıp karşılık gelen (Y, Y') çifti öğrenilir. Y ve Y''ne ΔZ farkı eklenerek (Z, Z') hesaplanır ve tahmin edilen son çevrim anahtarı kullanılarak (Z, Z') 'ne E_2 kısmı uygulanır ve (D, D') çifti elde edilir. (D, D') şifreli metinlerine karşılık gelen (R, R') açık metinleri arasında ΔP farkı olup olmadığına bakılır. Bu fark sağlanıyorsa tahmin edilen son çevrim anahtarının sayaç değeri bir arttırılır. Böylece her bir olası son çevrim anahtarı için bu ayıracın sağlanma sayısı bulunmuş olunur. Yanlış anahtarlarda bunun sağlanma olasılığı n^{-1} iken doğru anahtar için bunun sağlanma olasılığı ${p_0}^2 \times {p_1}^2$'dir. ${p_0}^2 \times {p_1}^2$'in n^{-1}'den yeterince büyük olması durumunda sayaç değerlerinden doğru anahtar ortaya çıkacaktır.

Konvansiyonel bumerang saldırında, saldırı sırasında belirlenen şifreli metinlere karşılık gelen açık metinlerin elde edilebiliyor olması gerekir. Başka bir ifadeyle, bu saldırı adaptif seçili şifreli metin saldırısıdır. Blok şifreleme algoritması çalışma modlarından bazılarında şifre çözme yönü kullanılamaz. Bu durumlar için bumerang saldırısı, Biham ve arkadaşlarının geliştirdiği bir dikdörtgen saldırısına dönüştürülebilir [22]. Böylece şifre çözme ihtiyacına gerek kalmaz. m adet aralarında ΔP farkına sahip (P, P') çifti alınır. Bir çiftin E_0 çevrim sonrasında ΔS farkını sağlama olasılığı p_0 'dır. Alınan iki çiftin $(P_i, P_i'), (P_j, P_j')$ E_0 sonrasında $S_i \oplus S_i' = \Delta S$ ve $S_j \oplus S_j' = \Delta S$ farklarını sağlama olasılığı ${p_0}^2$ olmakla beraber $S_i \oplus S_j = \Delta T$ farkını da sağlama olasılığı rasgelelikten dolayı ${p_0}^2 \times n^{-1}$ 'dir. Bu $(S_i, {S_i}', S_j, {S_j}')$ dörtlüsüne E_1 uy-

gulandıktan sonra elde edilen (Y_i, Y_i', Y_j, Y_j') değerleri için $Y_i \oplus Y_j = \Delta Z$ ve $Y_i' \oplus Y_j' = \Delta Z$ farklarının gerçekleşme olasılığı da ${p_1}^2$'dir. Sonuç olarak $P_i \oplus P_i' = \Delta P$ ve $P_j \oplus P_j' = \Delta P$ şartını sağlayan (P_i, P_i', P_j, P_j') dörtlüsü E_0 ve E_1 kısımlarından oluşan şifreleme işleminden geçtikten sonra elde edilen (Y_i, Y_i', Y_j, Y_j') dörtlüsünün $Y_i \oplus Y_j = \Delta Z$ ve $Y_i' \oplus Y_j' = \Delta Z$ farkını sağlama olasılığı ${p_0}^2 \times n^{-1} \times {p_1}^2$'dir. Bu farkların rasgele sağlanma olasılığı ise n^{-2}'dir. ${p_0}^2 \times n^{-1} \times {p_1}^2$'nin n^{-2}'den büyük olması durumda tahmin edilen son çevrim anahtarlarından doğru anahtar kolaylıkla elde edilebilir. Saldırının veri ve zaman karmaşıklığına bakılacak olursa ilk bakışta $c \times ({p_0}^2 \times n^{-1} \times {p_1}^2)^{-1}$ açık metin dörtlüsü gerekiyor gibi görünebilir. Fakat aslında yaklaşık olarak $(c \times ({p_0}^2 \times n^{-1} \times {p_1}^2)^{-1})^{1/2}$ adet açık metin çiftlerinden ihtiyacı karşılayacak kadar açık metin dörtlüsü elde edilebilir. Saldırı sırasında $(c \times ({p_0}^2 \times n^{-1} \times {p_1}^2)^{-1})^{1/2}$ adet açık metne karşılık gelen şifreli metinler her bir son çevrim aday anahtarı ile çözülüp bu çiftlere karşılık gelen (Y_i, Y_i') çiftleri elde edilir. Bu (Y_i, Y_i') çiftleri bir tabloya sıralı olarak yazılarak ΔZ farklarını sağlayan dörtlü sayısı kolaylıkla bulunabilir. Böylece aday anahtar sayaçları hesaplanıp doğru anahtar ayırt edilir. Bu saldırının bumerang saldırısına karşı en önemli avantajı seçili şifreli metin kullanmaması ve bu nedenle özellikle anahtar şemasının tersten çalıştırılmasına gerek duyulmadan ilintili anahtarlarla daha yüksek çevrim sayısına sahip karakteristikler üretilebilmesidir. Böylece, literatürde bumerang saldırısına karşı dayanıklı blok şifreleme algoritmalarına (düşük çevrim sayılarında) ilintili anahtar dikdörtgen saldırısının uygulandığı örnekler görülmektedir [68,33,123].

9.11.3. Saldırıların Bazı Uygulamaları

Bumerang saldırı örneklerinden en popüler olanlarından biri ilintili anahtar saldırı modeli ile bumerang saldırısının AES-192 ve AES-256 algoritmalarına uygulandığı çalışmadır [24]. İlintili anahtar saldırı modeli, farklı ama ilintili anahtarlar altında şifrelenmiş açık metin ve şifreli metin çiftlerinin

kullanıldığı saldırı modelleridir. Bumerang kriptoanaliz üzerine yapılan en güncel çalışmalardan biri ise Dunkelman ve arkadaşları tarafından 2020 yılında EUROCRYPT konferansında sunuldu [50]. Bu çalışmada yazarlar bumerang kriptoanaliz yöntemin geliştirerek "retracing boomerang attack" adını verdikleri yeni bir yöntem buldular. Bu yöntem ile klasik yöntemdeki $p_0{}^2 \times p_1{}^2$ olasılığını $p_0{}^2 \times p_1$'e yükselttiler. Bu yeni yöntemin başarısını göstermek için ise bu yöntemi 5 çevrimden oluşan AES algoritmasına uyguladılar. 2018 yılına kadar 5-çevrim AES algoritmasına yapılan saldırıların karmaşıklığı 2^{32} iken bu karmaşıklık [15] çalışması ile 2^{24}'e düşürülmüştü. "Retracing boomerang attack" yöntemi ile ise 5-çevrim AES'e $2^{16.5}$ şifreleme işlemi ile saldırı yapılabileceği gösterildi.

9.12. KARE VE İNTEGRAL KRİPTOANALİZLERİ

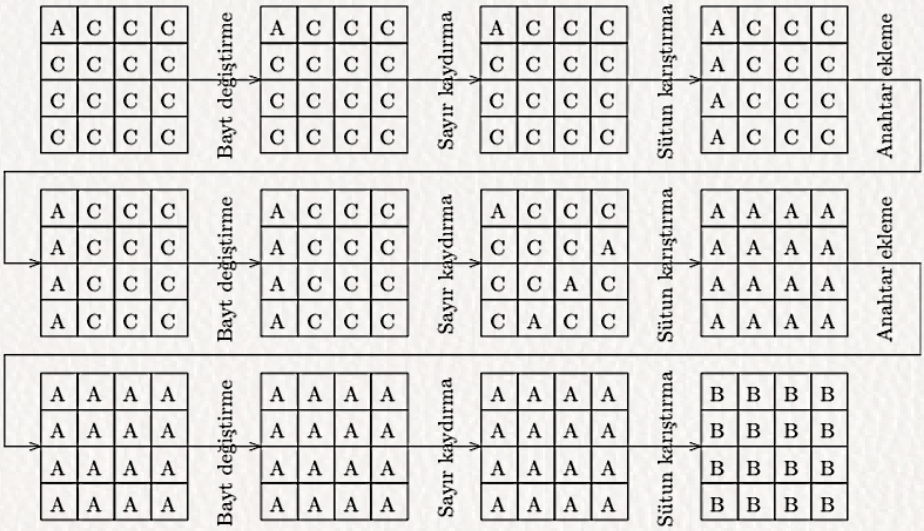
Algoritmanın bir kısmı için ayıraç bulup diğer kısmında kullanılan anahtarların bu ayıraç ile sınanarak bulunması yöntemine dayanan diğer bir kriptoanaliz yöntemi de kare kriptoanaliz yöntemi ve bu yöntemin genelleştirilmiş hali olan integral kriptoanaliz yöntemidir.

Kare kriptoanaliz yöntemi Square blok şifreleme algoritmasına 1997 yılında uygulanarak tanıtılmıştır [44]. Bu yöntemde saldırıda kullanılan ayıraç şu şekildedir. Açık metnin bazı bitleri belirlenir ve diğer bitleri sabit olan ve belirlenen bu bitlerin olası tüm değerleri alabilen bir açık metin kümesi alınır. Örneğin seçilen bitler açık metnin soldan ilk sekiz biti olsun. Bu sekiz bitin alabileceği değer sayısı 256 olduğu için 256 adet açık metin seçilir öyle ki soldan ilk sekiz bit haricindeki bitler 256 açık metin için aynı olup sadece bu sekiz bit değeri farklıdır. Bu açık metin kümesine karşılık gelen $r-1$ çevrim çıktısında bazı bitlerden oluşan bit gruplarının bazı özellikleri sağlaması beklenir. Örneğin 256 adet $r-1$ çevrim çıktısının en soldaki sekiz bit değerleri YaDa işlemine tabi tutulduğunda elde edilen sekiz bitlik değerin 0 olması beklenir.

Genel olarak kriptoanaliz yöntemlerinin basit olarak ifadesi için anlatımlarda ilk $r-1$ çevrim için ayıraç bulunur ve son çevrimdeki çevrim anahtarı sınanır şeklinde anlatılmasına rağmen aslında ayıraç ilk çevrimler yerine son çevrimler için bulunup ilk çevrim anahtarını elde etme yönünde de saldırı yapılabilir. Ayrıca ayıraç $r-1$ çevrim yerine daha az çevrim sayısından oluşan kısım için bulunup, saldırı uygulanan çevrim sayısı bir yerine daha fazla olabilir. Daha da ötesi, ayıraç bulunan kısım ne başta ne de sonda olup algoritmanın orta kısmında bulunabilir ve ilk ve son çevrimlerdeki çevrim anahtarı ele geçirilmeye çalışılabilir. Fakat anlatımın basitliği açısından ve genel bir fikir verilmesi için ayıracın ilk $r-1$ çevrim için bulunması ve son çevrimdeki çevrim anahtarının ele geçirilmesi şekilde anlatılmıştır.

Kare kriptoanaliz yöntemi genellikle Square ve AES gibi kelime tabanlı şifreleme algoritmaları üzerinde diğer algoritmalara göre daha başarılıdır. Kelime tabanlı şifreleme algoritmalarından kasıt, algoritma içinde kullanılan yapı taşlarının kelimeler (bit grupları) üzerinde çalışmasıdır. Örneğin AES şifreleme algoritmasında kullanılan yapıtaşlarına bakılırsa, bu algoritmanın 8-bit kelimeler üzerinde çalıştığı görülebilir. Yerini alma kutuları 8-bit girdi alıp 8-bit çıktı üretmekte, satır kaydırma işlemi 8-bitlik değerlerin yerini değiştirmekte ve sütun karıştırma işlemi de yine 8-bitlik 4 girdi alıp 8-bitler üzerinde doğrusal matris çarpması yaparak 8-bitlik 4 çıktı üretmektedir.

AES algoritmasında, örneğin açık metnin en soldaki 8-biti farklı olan ve diğer bitleri aynı olan 256 adet açık metin alınsın. Şekil 9.14'ten görüleceği üzere bu ilk sekiz bit "A" harfi ile diğer sabit sekiz bitler ise "C" harfi ile temsil edilsin. A aktif kelimeyi, bu kelimede bütün değerlerin eşit sayıda gözüktüğünü, C ise sabit kelimeyi; A olan kelimelerde her bir değer birer kez gözükürken C olan kelimelerde ise hep sabit değerlerin olduğunu ifade etmektedir. "B" harfi ise bu 256 adet açık metin için hesaplanan 8-bit değerlerin YaDa işlemine tabi tutulması sonucu elde edilen 8-bitlik değeri 0 olmasını temsil etsin.



Şekil 9.14. 3 Çevrimlik Kare Ayıracı

"A", "B" ve "C" özelliklerine sahip olan 8-bit kümelerinin AES'te kullanılan yapı taşlarından geçtikten sonraki özellikleri incelendiğinde şu sonuçlar ortaya çıkar.

— Anahtar ekleme işlemi: bir anahtar altında YaDa işlemi ile anahtar eklenmesi işlemi bire-bir bir fonksiyon olduğundan "A" özelliğine sahip küme anahtar eklendikten sonra da yine "A" özelliğine sahip olur. "B" özelliğine sahip küme için ise, aynı anahtar toplamda çift defa kümeye ekleneceği için anahtar eklenmesi önemsiz hale gelip küme yine "B" özelliğini sağlar. "C" özellikli küme için ise, tüm kümedeki 8-bitlere aynı anahtar uygulanacağı için küme yine "C" özelliğini sağlar.

— Bayt değiştirme kutusu: bu işlem de benzer şekilde bir bire-bir bir işlem olduğu için "A" özelliğini korur. Fakat burada "B" özelliğini koruması garanti edilemez. "C" özelliğinin korunduğu gözlemlemek ise oldukça basittir.

— Satır kaydırma işlemi: bu işlemde sadece 8-bitlerin yeri değiştiği için "A", "B" ve "C" özellikleri korunup sadece blok içerisindeki konumu değişir.

— Sütun karıştırma işleminde ise MDS matrisinden dolayı eğer girdideki dört 8-bitlik değerden sadece biri "A" ve diğerleri "C" ise çıktıdaki tüm dört 8-bitler "A" olur. Eğer birden fazla 8-bitlik değer "A" ise bu durumda çıktıda sadece 8-bit değerlerin "B" özelliğini sağladığı söylenebilir.

AES algoritmasında kullanılan yapıtaşlarının yukarıda bahsedilen özellikleri sağlamasından dolayı Şekil 14 ile gösterilen 3 çevrimlik bir ayıraç ortaya çıkar. Bu ayıracın başına bir çevrim ve sonuna da 2 çevrim eklendiğinde toplam 6 çevrim AES'e saldırı yapılabilir [43].

2002 yılında Knudsen ve Wagner tarafından tanıtılan integral kriptoanaliz [72] kare kriptoanalizin genelleştirilmiş hali ve farksal kriptoanalizin duali olarak düşünülebilir. Hatırlanacağı üzere farksal kriptoanalizde belli bir farka sahip açık metin çiftleri kullanılıyordu. Kare kriptoanalizde ise bir veya birkaç kelimesi aktif olan açık metin kümeleri kullanılıyordu, aslında bir kelimeye olası tüm farklar eklenerek bir küme oluşturuluyordu.

İntegral kriptoanalizde ise birden fazla aktif kümelerin farklı sabitlerle bileşkeleri alınır. Bu sabitler de bir araya getirildiklerinde kendi içinde aktif kümeler oluştururlar ki her bir katman bir derece olarak ifade edilir. Eğer girdi kümesinde sabit tutulan her bir kelime için sadece bir sabit varsa birinci dereceden integral küme kullanılmış demektir. Bir kelimede sabitler olası bütün değerleri alıyorsa bu ikinci derece integral kümedir. Benzer şekilde bir kelimede sabitler olası bütün değerleri alıyorsa ve bu kelimedeki sabit bir değer için başka bir kelimedeki sabitler de olası bütün değerleri alıyorsa bu üçüncü derecede integral kümedir. Knudsen ve Wagner bu tür integral kümelerle karakteristiklerdeki dengelilik ayıracının birkaç çevrim daha uzatılabileceğini göstermişlerdir [72]. Kare ve integral kriptoanalizin en bilinen uygulamalarından bazıları AES'e yapılan saldırılardır [53,111].

9.13. ORTADA BULUŞMA SALDIRILARI

9.13.1. Saldırının Genel Tanımı

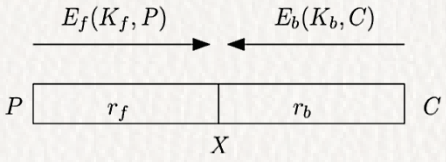
Anahtar bitlerinin çevrimler boyunca yayılımının az olması durumunda sıklıkla tercih edilen bir yöntem olan ortada buluşma saldırısı ilk olarak 1977 yılında çift anahtarlı DES (double DES) şifreleme algoritmasına uygulandı [116]. İlginçtir ki bu saldırı yöntemi onlarca yıl kullanılmadı ve 2008 yılında özet alma algoritmalarında kullanımının ortaya çıkmasıyla popülerlik kazanmaya başladı.

Bu saldırı yöntemindeki temel fikir, algoritma içerisindeki bazı değişkenlerin tüm anahtar bitleri yerine daha az anahtar biti bilinerek açık metinden hesaplanabilmesi ve aynı değişkenlerin ayrıca tüm anahtar bitleri yerine daha az anahtar biti bilinerek şifreli metinden hesaplanabilmesine dayanır. Dolayısıyla algoritma parçalara bölünmüş ve her bir parça için ayrı anahtar bitleri denenmiş ve bu anahtar bitleri de algoritma içindeki değişkenlerde aynı değere ulaşabilmesi yöntemi ile sınanmış olur. Böylece böl ve fethet yöntemi ile doğru anahtara ulaşılmaya çalışılır.

Temel ortada buluşma saldırısı Şekil 9.15 yardımı ile aşağıdaki adımlarla detaylı olarak anlatılabilir. Örneğin algoritma şu şekilde iki parçaya ayrılabilsin: E_f algoritmanın r_f çevrimden oluşan ilk parçası ve E_b'de algoritmanın r_b çevrimden oluşan ikinci parçasının tersi olsun. E_f kısmı için kullanılan anahtar bitleri K_f ile ve E_b kısmı için kullanılan anahtar bitleri de K_b ile gösterilsin ve ana anahtar $K = K_f||K_b$ olsun. K_f ve K_b anahtar bitlerinden oluşsun. Bu durumda şu şekilde bir saldırı yapılabilir:

- Bir adet açık / şifreli metin çifti verilsin ((P, C)).
- k_f anahtarının olası tüm değerleri için açık metinden $X = E_f(K_f, \mathrm{P})$ değeri hesaplansın ve bu sonuçlar bir tabloda sıralı olarak saklansın.
- k_b anahtarının olası tüm değerleri için şifreli metinden $X = E_b(K_b, \mathrm{C})$ değeri hesaplasın ve bulununan sonucun önceki adımda

oluşturulan tabloda olup olmadığına bakılsın. Eğer tabloda bir eşleşme varsa eşleşmeye neden olan anahtar çifti (K_f, K_b) doğru anahtar adayı olarak belirlensin.



Şekil 9.15. Temel Ortada Buluşma Saldırısı

Saldırının doğruluğu. Bu saldırıda aşikardır ki eğer (K_f, K_b) çifti doğru anahtar ise %100 olasılıkla tabloda bir eşleşme var olur. Bu eşleşmenin yanlış bir (K_f, K_b) anahtarı için gerçekleşme olasılığı ise rasgele olarak eşitliğin sağlanma olasılığı olur ki bu da n blok boyu, X'in bit uzunluğu olmak üzere 2^{-n} 'dir. Dolayısıyla saldırı sonucunda bir doğru anahtar ve K ana anahtarın bit uzunluğu olmak üzere 2^{k-n} adet yanlış anahtar elde edilir. Bu anahtarlardan doğru anahtarı elde etmek için ise başka bir (P, C) çifti kullanılarak sınama yapılır.

Saldırının hesaplamalı karmaşıklığı. k_f ve k_b sırası ile K_f ve K_b anahtarlarının bit uzunluğu olmak üzere, açık metinden K_f anahtarı tahmin edilerek toplam 2^{k_f} şifreleme işlemi yapılıp ve olası X değerleri bir tabloya yazılır. Şifreli metinden ise olası X değerlerini hesaplamak için 2^{k_b} şifreleme işlemi yapılır. Dolayısıyla saldırıda toplam $2^{k_f} + 2^{k_b}$ şifreleme işlemi yapılır ki bu hesaplamalı karmaşıklık kaba kuvvet karmaşıklığı olan $2^{k_f+k_b}$'den çok daha küçüktür. Ancak tablonun saklanması için 2^{k_f} boyutunda belleğe ihtiyaç vardır.

9.13.2. Saldırının Türevleri ve Güncel Uygulamaları

Zamanla ortada buluşma saldırıları yöntemi geliştirildi ve saldırının farklı versiyonları yayınlandı. Bunlara örnek olarak *üçlü küme ortada buluşma*

saldırı (3-subset meet in the middle attack) [3], *kısmi buluşturma (partial matching) [3]*, *yapıştır ve kes (splice and cut) [4]*, *Demirci-Selçuk saldırısı [45]*, *bi-klik (biclique) [28]* versiyonları verilebilir.

Çoğu durumda algoritmanın çift anahtarlı DES örneğinde olduğu gibi iki parçaya ayrılması mümkün olmayabilir. Örneğin, algoritmanın ilk kısmı ile son kısmında bazı anahtar bitleri ortak olabilir. Örneğin K_f sadece ilk kısmında kullanılan anahtar bitleri, K_b son kısmında kullanılan anahtar bitleri ve K_c de her iki kısımda da kullanılan anahtar bitleri olsun. Bu durumda ana anahtar bitleri K_f, K_b ve K_c anahtarlarından oluşur. Böyle bir algoritmada saldırı şu şekilde yapılabilir. K_c'nin her bir olası değeri için yukarıdaki saldırı uygulanır. Dolayısıyla saldırının karmaşıklığı $2^{K_c} \times (2^{K_f} + 2^{K_b})$ olur ki yine bu sonuç kaba kuvvet saldırısı karmaşıklığı olan $2^{K_f+K_b+K_c}$'ye göre çok daha düşüktür. Bu yöntem *üçlü küme ortada buluşma saldırısı* olarak adlandırılır [3].

Yine yukarıdaki anlatılan senaryoya uygun gerçek bir algoritma bulunmayabilir. Örneğin ortada buluşulmak istenen X değeri dikey kesilmiş bir ara değer olmayabilir. Bunun yerine örneğin X 'in bazı kısımları bir çevrimin yerini alma kutularının girdisi, X 'in diğer kısımları da aynı çevrimin yerini alma kutularının çıktısı olabilir. Bu durumda yukarıdaki saldırı benzer şekilde uygulanabilir.

Modern algoritmalarda bu şekilde bir senaryo da söz konusu olmayabilir. Bu durumda aşağıdaki yol izlenebilir. Algoritma E_{b0}, E_f ve E_{b1} olmak üzere Şekil 9.16'da görüldüğü gibi 3 parçaya ayrılır. K_b anahtarı E_{b0} ve E_{b1} kısımlarında kullanılan ve K_f anahtarı da E_f kısmında kullanılan anahtar olsun. Bu durumda saldırı şu şekilde gerçeklenebilir:

— Bir X değeri seçilir.

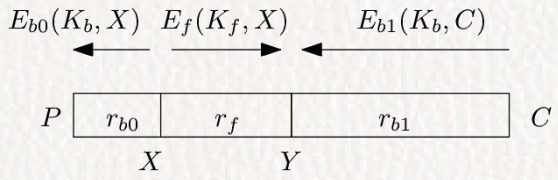
— Olası tüm K_b değerleri için

- $E_{b0}(K_b, X)$ hesaplanarak P açık metni elde edilir.

- Bu elde edilen açık metne karşılık gelen şifreli metin istenir.
- Bu şifreli metinden $Y = E_{b1}(K_b, C)$ hesaplanarak Y elde edilir ve bir tabloda saklanır.

— Olası tüm K_f değerleri için $Y = E_f(K_f, X)$ hesaplanır ve tabloda olup olmadığına bakılır. Tabloda eşleşmeye neden olan (K_f, K_b) çifti aday anahtar olarak belirlenir.

Bu saldırı da aslında yukarıdaki saldırılara benzemekle birlikte temel fark kullanılan açık / şifreli metin sayısıdır. Hesaplamalı karmaşıklık aynı olmakla beraber bu saldırıda kullanılan (P, C) çifti sayısı 2^{k_b}'dir. Bu yöntem *yapıştır ve kes saldırısı* olarak adlandırılır [4]. Yukarıdaki saldırılarda olduğu gibi diğer uyarlamalar da bu saldırı yapılabilir. Örneğin Y'nin tüm blok olması yerine kısmi bir blok olması ve X 'nin ve Y 'nin dikey çizgiler yerine merdiven vari çizgiler olması gibi kombinasyonlar uygulanabilir.



Şekil 9.16. Yapıştır ve Kes Saldırısı

Ortada buluşma saldırısının diğer bir gelişmiş versiyonu da Demirci-Selçuk saldırısıdır [45]. Tahmin edilen çevrim anahtar bit sayısı veya algoritma içerisindeki değişken sayısının toplam bit uzunluğu, buluşulan noktanın bit uzunluğu ile anahtar uzunluğunun toplamına eşit veya büyük ise yukarıdaki yöntemler olası anahtar uzayını daraltmayacağı için bir fayda sağlamaz. Örneğin 256-bit AES algoritması düşünüldüğünde toplamda 264 bit tahmin yapılıp sadece 8-bit uzunluğunda bir değişkende buluşulmaya çalışılırsa saldırı sonunda elde 2^{256} aday anahtar kalır ki bu durum kaba kuvvet saldırısına karşı bir kazanç sağlamayacaktır. Bu nedenle buluşma noktasındaki

rasgele eşitlenme olasılığının çok daha düşük olması beklenir ki yanlış tahminleri eleme gücü fazla olsun. Bu nedenle açık-şifreli metinlerin teker teker sınamada kullanılması yerine bazı özelliklere sahip açık metin şifreli metin kümeleri kullanılır. Böylece birden fazla açık metin şifreli metin için ortada buluşma kısıtının aynı anda sağlanması gerekeceği için ortada buluşma kısıtından dolayı eleme gücü çok daha fazla olacaktır. Bu da eleme sonucunda aday anahtar sayısını oldukça azaltacaktır. Yukarıdaki örnekte 8 adet metin çifti kullanılabilirse ortada buluşma kısıtının rasgele gerçekleşme olasılığı $(2^{-8})^8 = 2^{-64}$ olur. Bu durumda elde kalan aday anahtar sayısı $2^{264-64} = 2^{200}$ kalır ve bu da kaba kuvvet saldırısına göre çok daha iyi bir sonuçtur. Ortada buluşma saldırısı bir çok algoritma üzerinde uygulanmakla beraber AES üzerinde en çok uygulanan saldırı yöntemlerinden biridir. Bu yöntem ile 7-çevrim AES'e saldırı yapılabilmiştir [47].

9.14. CEBİRSEL SALDIRILAR

Bu kısımda, literatürde bulunan farklı yaklaşımlara göre şifreleme algoritmalarına yapılan cebirsel kriptoanaliz anlatılmaktadır. Atakların daha iyi anlaşılabilmesi için bazı tanımlara yer verilecektir.

Blok şifreleme algoritması, n bit girdiyi k bitlik anahtar altında n bitlik çıktıya götüren birebir fonksiyondur. Cebirsel saldırının amacı bazı özel cebirsel denklem sistemleri kurup bu sistemlere özel çözümler geliştirerek K anahtarı hakkında bilgi elde etmektir.

Blok şifreleme algoritmalarının çevrim fonksiyonu genel olarak; doğrusal anahtar ekleme işlemi, doğrusal olmayan S-kutusu işlemi ve L doğrusal katman işleminden oluşur. Bu adımların ayrıntıları şu şekilde özetlenebilir: k_i alt anahtar ekleme işlemi genellikle YaDa işlemidir. s -kutusu işlemi ise paralel olarak küçük bloklara (4-bit, 8-bit gibi) S-kutusu işlemi uygulanması şeklindedir. L ise genellikle sonlu cisimde çarpma işlemlerinden oluşan matris operasyonudur. Bu işlemlerden oluşan çevrim fonksiyonu r defa uygulanarak şifreli metin oluşturulur.

$$p \to \overset{\downarrow k_0}{\oplus} \to x_1 \to S \to y_1 \to L \to \overset{\downarrow k_1}{\oplus} \to x_2 \cdots x_r \to S \to y_r \to L \to \overset{\downarrow k_r}{\oplus} \to c.$$

Şekil 9.17. Blok Şifreleme Genel Yapı

Şekil 9.17'de, $x_1 = p \oplus k_0$ ve $i = 1, \cdots, r$ için $x_{i+1} = k_i \oplus L\big(S(x_i)\big) = k_i \oplus L(y_i)$ ve $x_{r+1} = c$ elde edilir. Blok şifrelerde $i = 1, \cdots, r$ için (x_i, y_i)) ikililerinin bitleri arasındaki düşük dereceli cebirsel denklemler cebirsel atak-lar için çok kullanışlı olabilmektedir. Şifreleme algoritması için açık metin, şifreli metin ve anahtar bilgisinden bağımsız olan bu denklemler, sadece S kutusu özellikleriyle ilgilidir. Doğrusal katmandan ise çevrim anahtarlarını, (x_i, y_i) ikililerini, açık metni ve şifreli metni birbirine bağlayan denklemler elde edilmektedir. s -kutusu katmanından ve L katmanından elde edilen denk-lemlerle, şifreleme algoritmasının eksiksiz bir cebirsel tanımı elde edilir.

9.14.1 Cebirsel XSL Atağı

Bu kısımda Courtois ve Pieprzyk'in [40] nolu bildirilerinde yayınladıkları XSL atağının kısa bir anlatımı verilmektedir. Cebirsel saldırıların amacı eldeki algoritmanın (çoğunlukla GF(2) üzerinde) girdi, çıktı ve anahtar bitle-ri arasında çok değişkenli polinomsal denklem sistemi elde etmek ve bu sistemi çözerek anahtar bitlerini ele geçirmektir. Bu tip denklem sistemleri-nin çözümünün genel olarak NP-hard olduğu bilinmektedir. Ancak denklem sistemi yapısının seyrek (sparse) olması durumunda, açık literatürde [41] nolu bildiride tanıtılan XL ve bunun daha da geliştirilmiş bir versiyonu olan XSL gibi bazı algoritmalar ile bu tür denklem sistemlerine çözümler öneril-mektedir [40,42]. Literatürde Courtois ve Pieprzyk'in önerdiği XSL algorit-masının, yazarları tarafından iddia edildiği performansta ve karmaşıklıkla çalışmayacağına ilgili eleştirel makaleler bulunmaktadır. Özellikle üretilen denklemler içinde doğrusal bağımsız denklem sayısı konusunda literatürde tartışma hakimdir. [40,42] nolu makalelerde üretilen denklemlerin belli bir sayıdan sonra çok büyük bir çoğunluğunun doğrusal bağımlı olduğu düşü-nülmektedir [71, 36, 91,78, 39].

XSL ve kompakt XSL saldırılarının çalışma prensipleri aşağıda kısaca özetlenmiştir: S-kutusundan elde edilen d. derece (d =2 ya da 3) denklemlerin sayısı r olsun ve bu denklemlerin içinde yer alan monom sayısı t olsun. Anahtar şeması denklemleri de kullanıldığında N_r çevrim sayısı olmak üzere algoritma bir kez çalıştırılır. Atağı gerçekleştirmek için bir adet açık-kapalı metin çifti kullanılır. Eğer toplamda kullanılan S-kutusu sayısı S, bir çevrimde kullanılan S-kutusu sayısı B ile gösterilecek olursa ve anahtar şemasında kullanılan toplam S-kutusu sayısı S_k ile gösterilecek olursa,

$$S = B \cdot N_r + S_k$$

olacaktır.

Her bir S-kutusunun denklemleri, diğer S-kutularının denklemlerinin $(P-1)$'li altkümelerinin terimlerinin çarpımlarıyla çarpılır. Atak karmaşıklığının hesaplanmasında önemli bir parametre olan P tamsayı değeri daha sonra belirlenecektir. Bu şekilde elde edilecek toplam denklem sayısı, içlerindeki doğrusal bağımlı olanlar elendikten sonra,

$$R \approx \sum_{i=1}^{P} \binom{S}{i} r^i \binom{S-i}{P-i} (t-r)^{P-i} \approx \binom{S}{P} (t^P - (t-r)^P)$$

ve bu denklemlerdeki toplam farklı monom sayısı,

$$T \approx t^P \cdot \binom{S}{P}$$

olacaktır.

Ardından [56] makalesinde bahsedilen T' metodu adı verilen yöntemle, yeni doğrusal bağımsız denklemler elde edilir. Bu denklemlerin sayısı yaklaşık,

$$T' \approx t' t^{P-1} \cdot \binom{S-1}{P-1}$$

kadardır. S-kutusuna göre değişiklik gösteren $t' < t$ değeri [56] 'de verilmektedir.

Elde edilecek denklemler sadece yukarıdakilerden ibaret değildir. Difüzyon katmanındaki doğrusal denklemler de diğer S-kutularının $(P-1)$ 'li altkümelerinin terimleriyle çarpılarak yeni denklemler elde edilir. s, S-kutusunun çıktı boyunu göstermek üzere bu denklemlerin sayısı,

$$R' \approx (N_r \cdot B + S_k) \cdot s \cdot (t-r)^{P-1} \cdot \binom{S}{P-1}$$

olarak hesaplanır. Eldeki doğrusal bağımsız denklem sayısının, toplam terim sayısından fazla olduğunu garantilemek için gerekli koşul,

$$T - R - R' < T'$$

şeklindedir. Bu koşulu sağlayan P değeri bulunduktan sonra her terim yeni bir değişkenmiş gibi kabul edilerek, sistem Gauss eleme yöntemiyle çözülür. Bu hesaplamalarda Gauss eleme yöntemindeki w parametresi genellikle en optimistik değer olan 2.376 olarak alınır.

Kompakt XSL algoritmasında ise kullanılan terim sayısının azaltılması hedeflenmiştir. Bunun için kullanılan S-kutusu d. derece ($d = 2$ ya da 3) denklemlerinde beliren monomlardan $t - r$ tanesi taban olarak alınır. Bu tabanın içerisinde birinci derece girdi ve çıktılar olmaz. Öte yandan "1" tabanın içerisinde yer almalıdır. Taban belirlendikten sonra geriye kalan r terim bu taban elemanların doğrusal birleşimleri olarak yazılır. Bu adımdan sonra kullanılan mantık XSL algoritmasındaki gibidir. Tek fark artık denklemlerin oluşturulmasında kullanılan monomların taban elemanlarından oluşacak olmasıdır.

Bu atakların öngörüldüğü gibi çalışıp çalışmadığı konusunda kriptoloji camiasında bir tartışma vardır. [42] nolu makalede, yazarlar kübik denklemler alındığında $t = 697, r = 471, t' = 242$ olacağını ve $P = 5$ için AES-256'nın oluşturduğu denklemlerin çözüm karmaşıklığının yaklaşık 2^{203} olacağını iddia etmişlerdir. Blok şifreleme algoritmalarına uygulanan XSL saldırısı ile AES algoritmasının kırıldığı iddia edilse de, bu ataktaki karmaşıklık hesapları literatürde oldukça yoğun bir şekilde tartışılmıştır ve hala doğrulanamamıştır.

XSL Algoritmasının Analizi. Cid ve Leurent, Courtois ve Pieprzyk tarafından 2002 Asiacrypt makalesi [40] ile önerdikleri kompakt XSL atağını daha açık bir şekilde anlatmışlardır [36]. Yazarlar, XSL atağının tanımlandığı şekliyle AES sistemini çözmede yeterli olamayacağına dair güçlü kanıtlar ortaya koyduklarını ifade etmektedirler. Courtois ve Pieprzyk'in makalesinde komşu S kutularındaki terimlerin yeni denklem oluşturmada kullanılmamasını ve farklı S kutularındaki terimlerin kullanılmasını tavsiye etmesinin neticesinde çarpımlarda oluşan terimlerde yerine koymanın yeni terimleri sisteme katacağı, yerine koyma olmadan da yeni denklemler (expression) elde edilemeyeceği ifade edilmektedir. Bunu aşmak için de P terimin farklı S kutularından gelme şartının ve komşu S kutularından terim gelmemesi şartının kaldırılmasını tavsiye etmişler ve bu yeni haline sXL algoritması ismini vererek analizlerini yapmışlardır. Sonuç olarak [40] makalesinde T' yönteminin çalışmayacağını, T yönteminin ise, denklem ve parametre sayısı belli bir eşiği geçince literatürde daha önce yayınlanan XL yöntemine [41] denk olduğunu ve bu yöntem ile denklem sistemi kurmanın ve çözmenin maliyetinin ise AES-128 için bile 2^{488} civarında olacağı iddia edilmiştir [36].

9.14.2. Knudsen ve Miolane Bakış Açısı ile Cebirsel Atak

Bu bölümde [71] referanslı makalenin içeriği kısaca özetlenmiştir. [71] makalesinde blok şifreleme algoritmalarına yapılan cebirsel saldırılardaki doğrusal bağımsız denklem sayısı hesaplanmaya çalışılmıştır. Blok şifreleme algoritmalarına yapılan cebirsel saldırılardaki temel fikir şifreleme algoritması için ilk başta oluşturulan denklemleri sistemde yer alan değişkenlerle çarpıp daha fazla denklemden oluşan yeni bir sistem oluşturup bu denklem sistemini çözmeye çalışmaktır. Bu yaklaşımdaki en temel sorun oluşan yeni denklem sistemindeki doğrusal bağımsız denklem sayısının tam olarak hesaplanamamasıdır. Bu durum, bu yaklaşımlardaki saldırıların karmaşıklığının tam olarak hesaplanamamasına yol açmaktadır. İlgili saldırı için bazı tanımlar verilecektir.

Doğrusal Bağımsız Denklem Sayısı Hesaplama $GF(2)[x_1, \cdots, x_n]$ polinom halkasındaki terimlerin (monom) sıralanması için "graded reverse lexicographic mertebe (grevlex)" tercih edilmiştir. İlgili sıralamanın tanımı, Tanım 9.1'de verilmiştir.

Tanım 9.1. *(Graded Reverse Lexicographic mertebe, [71] Tanım 1)*

1. *Eğer* $i = 1, \cdots, n$ *için* $\alpha_i = \beta_i$ *ise,*

$$x_1^{\alpha_1} \cdots x_n^{\alpha_n} = x_1^{\beta_1} \cdots x_n^{\beta_n}$$

2. *Eğer* $|\beta| = \sum_{i=1}^{n} \beta_i < |\alpha| = \sum_{i=1}^{n} \alpha_i$ *veya* $|\alpha| = |\beta|, (\alpha_1 - \beta_1, \alpha_2 - \beta_2, \cdots, \alpha_n - \beta_n)$*'nin sıfırdan farklı en sağdaki terimi negatif bir sayı ise,*

$$x_1^{\beta_1} \cdots x_n^{\beta_n} < x_1^{\alpha_1} \cdots x_n^{\alpha_n}$$

Bu bölümde farklı alt kümelerden elde edilebilecek doğrusal bağımsız denklem sayısını hesaplamak için bazı araçlar anlatılmaktadır. Bu kısımda teoremlerin ispatları verilmeyecektir.

Teorem 9.1.*([71] Teorem 2)* $\mathcal{E}_L, GF(2)$ *üzerinde* n *tane değişken içeren bir dereceli* $m \leq n$ *tane doğrusal bağımsız bir denklem sistemi olsun.* $\mathcal{E}_L$*, denklem sisteminin en azından bir tane çözümünün olduğu kabul edilsin.* $n - m \geq d - 1$ *şartını sağlayan* $d > 0$ *seçelim ve* d*-1 dereceli tüm terimlerle* $\mathcal{E}_L$ *sistemindeki tüm denklemler çarpılsın. Oluşturulan denklemler, terimlerini daha önce bahsedilen mertebeye göre sıralanarak bir matris formatında yazılsın. Gauss eleme yöntemiyle farklı* d*-dereceli en önemli terime sahip* $N = \binom{n}{d} - \binom{n-m}{d}$ *tane doğrusal bağımsız denklem elde edilir.(*$k >$ *lir.(*$k > n$ *için* $\binom{n}{k} = 0$ *olarak tanımlanmıştır.)*

Teorem 9.1 kullanılarak doğrusal katmanlardan elde edilebilecek farklı d dereceli en önemli terime sahip doğrusal bağımsız denklem sayısı hesaplanabilir.

S-kutusu Katmanındaki Denklemler Bu bölümde b bit girdiye, c bit çıktıya sahip sistemdeki S kutularından elde edilecek farklı d dereceli en önemli terime sahip doğrusal bağımsız denklem sayısının nasıl hesaplanacağı verilmiştir.

Bir blok şifreleme algoritmasında toplam s tane S kutusu olduğu kabul edilsin. Her S kutusu $GF(2)$ üzerinde çok değişkenli denklemlerle ifade edilebilir. $h(i)$, S kutusunun denklemlerinde en önemli terim olmayan $b+c$ değişkenden oluşan i dereceli terimlerin (monom) sayısı olarak tanımlanmıştır. Sabit terim hiçbir zaman bir denklemin en önemli terimi olmaz. Bu yüzden $h(0)=1$'dir. Çoğu S kutusu için $h(1)=b+c$'dir. Aksi takdirde girdi ve çıktı bitleri arasında bir olasılıklı doğrusal denklemler var demektir. Çoğu algoritmada böyle kötü bir S kutusu seçimi yapılmamaktadır. Diğer i değerleri için $h(i)$ değerleri polinomsal denklemlerdeki farklı i dereceli en önemli terimler sayılarak hesaplanır. Herhangi bir b bit girdiye, c bit çıktıya sahip S kutusu için $\sum_{i=0}^{b+c} h(i) = 2^b$ eşitliği sağlanır.

Toplam s tane S kutusunun denklemlerinin en önemli terimlerinden hiç birisine bölünmeyen s tane S kutusunun değişkenleri kullanılarak oluşturulan d dereceli monomların sayısı hesaplansın. v_i, i. S kutusunun denklemlerinin hiç birisinin en önemli terimi olmayan bir terim olsun. Bu d dereceli monomlar, $v_1 \cdot v_2 \cdot v_3 \cdots v_s$ şeklinde ifade edilebilir. d_i, v_i monomunun derecesi olsun.

$$S_d = \sum_{d=d_1+d_2+\cdots+d_s} \prod_{i=1}^{s} h(d_i)$$

değerini ile istenen özellikteki d dereceli monomların sayısı tam olarak hesaplanır.

$$P(x) = h(0) + h(1)x + h(2)x^2 + \cdots + (b+c)x^{b+c}$$

polinomu ele alınsın. S_d değeri, $P(x)^s = S_0 + S_1 x + S_2 x^2 + \cdots + S_{b+c} x^{(b+c)s}$ polinomunun katsayılarından bulunabilir. Bu notasyonlardan

sonra doğrusal olmayan katmanlardan oluşturulabilecek farklı d dereceli en önemli terime sahip doğrusal bağımsız denklem sayısını hesaplamaya yarayan aşağıdaki teorem yazılabilir.

Teorem 9.2.*([71] Teorem 3) b bit girdiye c bit çıktıya sahip s tane S kutusu kullanılan bir algoritma olsun. Her bir S kutusu $b + c$ tane girdi, çıktı bitleri cinsinden bir denklem kümesiyle ifade edilsin. $h(d_i)$ yukarıda tanımlandığı gibi olsun. Bu s tane denklem kümesinin birleşiminde $n = (b + c)s$ tane değişken kullanılır. Bu değişkenler cinsinden yazılan d dereceli farklı en önemli terime sahip elde edilebilecek doğrusal bağımsız denklem sayısı tam olarak $N = \binom{n}{d} - S_d$'dir. Bu ifadedeki $S_d = \sum_{d=d_1+d_2+\cdots+d_s} \prod_{i=1}^{s} h(d_i)$'dir.*

Örnek bir algoritma için N_d değerini hesabı aşağıda verilmiştir. 4 bit girdiye 4 bit çıktıya sahip 4 tane S kutusu kullanılsın ve her bir S kutusu 21 tane 2. dereceden denklemle ifade edilebilsin. O halde $P(x) = 1 + 8x + 7x^2$ olarak hesaplanır. Buradan $P(x)^4 = (1 + 8x + 7x^2)^4$ olur. $P(x)^4$'ün x^5 teriminin katsayısından $S_5 = 19040$ olarak hesaplanır. Teorem 142'den 5 dereceli farklı en önemli terime sahip $N_5 = \binom{(4+4)\cdot 4}{5} - S_5 = 182336$ tane denklem vardır.

Bir blok şifreleme algoritması için $GF(2)$ üzerinde ilk başta oluşturulan denklem sistemi iki ayrı alt-kümeye bölünmektedir [71] $\mathcal{L}$: Doğrusal katmanlardanelde edilen denklemleri içeren alt-küme, S: Doğrusal olmayan katmanlardan (örneğin S-kutusu) elde edilen denklemleri içeren alt-küme olsun. Daha sonra da bu iki alt-kümedeki denklemler sistemdeki değişkenlerle çarpılarak bu alt-kümelerde elde edilen önceden belirlenmiş bir dereceye sahip doğrusal bağımsız denklem sayıları tam olarak hesaplanabilmektedir. T, denklem sistemindeki bir dereceli terimlerin (değişkenler) sayısı olsun. $\mathcal{L}_d$, $\mathcal{L}$ kümesindeki denklemlerin T 'deki terimlerin çarpımıyla elde edilen farklı en önemli terime sahip d dereceli doğrusal bağımsız denklemlerin kümesi olsun. Benzer şekilde S_d, S kümesindeki denklemlerin T'deki terimlerin çarpımıyla elde edilen farklı en önemli terime sahip d dereceli doğrusal

bağımsız denklemlerin kümesi olsun. Bu makalede $\mathcal{L}_d$ ve S_d kümelerinin eleman sayısı tam olarak hesaplanabildiği gösterilmiştir. Anahtar değeri bu iki kümenin bileşiminden elde edilen kümedeki denklemlerin çözümüdür. Çözüm ayrı ayrı $\mathcal{L}_d$ ve S_d alt-kümelerinin de çözümüdür. Bu iki kümenin toplamındaki doğrusal bağımsız denklem sayısı açık problem olarak kalmıştır. Bunun için üstten sınır $|\mathcal{L}_d| + |S_d|$'dır.

AES s-kutusu 39 tane derecesi 2 olan 120 terimli, 432 tane derecesi 3 olan 560 terimli ve 1790 tane derecesi 4 olan 1820 terimli denklemler üretmektedir. Bu denklemlerden $d \geq 5$ olmak üzere $\binom{16}{d}$ kadar baş terimi farklı denklem elde edilebilir. Sonuç olarak, Teorem 9.2'de geçen h değerleri AES S-kutusu için, $h(0) = 1, h(1) = 16, h(2) = 81, h(3) = 128, h(4) = 30$ ve diğer geri kalan değerler için $h(i) = 0$ olacaktır. Tüm şifreleme algoritması için toparlandığında, bu denklem ve bilinmeyen sayıları AES-128 için hesaplandığında toplamda 200 S-kutusu ve 7800 ikinci dereceden denklem elde edilir. Anahtar şemasıyla birlikte 9400 ikinci derece denklem ve 3200 değişken mevcuttur.

İkiden büyük derecedeki terimlerle elde edilmiş denklemlerin doğrusallaştırılmaları ile elde edilen doğrusal bağımsız denklemler için Teorem 9.2 kullanılırsa 4. dereceden yaklaşık 2^{42} bilinmeyenli toplamda yaklaşık $2^{41,9}$ doğrusal katmandan ve $2^{35,2}$ S-kutularından; 5. dereceden yaklaşık $2^{51,3}$ bilinmeyenli toplamda yaklaşık $2^{51,26}$ doğrusal katmandan ve $2^{45,3}$ S-kutularından; 6. dereceden yaklaşık $2^{60,4}$ bilinmeyenli toplamda yaklaşık $2^{60,34}$ doğrusal katmandan ve $2^{54,9}$ S-kutularından doğrusal bağımsız denklem elde edilebilmektedir [71]. Derece $d > 6$ olduğunda artık $|\mathcal{L}_d| + |S_d| < |T_d|$ olmaktadır. Bilinmeyen sayısından daha fazla doğrusal bağımsız denklem üretilmektedir. Diğer taraftan en optimum durum olan 6. dereceden denklemlerin çözümü için bile yaklaşık 2^{60} bağımsız denkleme ihtiyaç vardır. Çünkü $T_6 \approx 2^{60}$ değişken mevcuttur. Bu durumda dahi denklem sistemini çözmek kaba kuvvet saldırısından çok daha yavaştır ve üstelik bir o kadar da bellek ihtiyacı vardır. Kaldı ki, AES S-kutuları bu kadar denklem üretememektedir. Doğrusal kat-

manın ürettiği denklemler ise AES'e özgü değildir ve bütün YPA yapılarının doğrusal katmanları yaklaşık aynı sayıda denklem üretirler.

9.14.3. BES Yaklaşımı

Murphy ve Robshaw AES algoritmasının işlemlerini genelleştirdiler ve bütün işlemleri $GF(256)$'da olacak şekilde tanımlayıp BES (Big Encyption System) algoritmasını sundular [91]. AES algoritması genel olarak güvenlik açısından BES'e denkti ve XLS algoritmasını BES'e uygulamak çok daha pratikti. Çünkü BES'te bütün işlemler $GF(256)$ cismi üzerinde yapılıyordu ve denklem sistemleri de $GF(2)$ yerine $GF(256)$ üzerinde oluşturulabilirdi. Bu durumda AES S-kutusu son derece basit bir cebirsel ifade haline dönüşmektedir: $S(s) = y \Rightarrow x \cdot y = 1$. Satır kaydırma zaten $GF(256)$ üzerindeki elemanlara herhangi bir işlem yapmamaktadır. Sütun karıştırma ise $GF(256)$ üzerinde 4×4 matris çarpmasıdır. Dolayısıyla BES algoritmasının bütün yapıtaşları $GF(256)$ üzerinde cebirsel işlemlerden oluşmaktadır. XSL saldırısını BES'e uygulamak ve karmaşıklığını kontrol etmek çok daha az çetrefelli bir süreçtir. Eğer XSL saldırısının karmaşıklığı ve saldırıda elde edilen bağımsız denklem sayısı doğru ise bu durumda BES'e yapılan XSL saldırısının karmaşıklığı 2^{87} olmalıydı.

Diğer taraftan Lim ve Khoo BES algoritmasına uygulanan XSL saldırısındaki denklemlerin doğrusal bağımsızlıklarını analiz ettiler [78]. Courtois ve Pieprzyk'in [40,42] iddia ettiklerinin aksine BES için oluşturan denklemler büyük oranda doğrusal bağımlı idi. Lim ve Khoo, BES-128'e yapılan XLS saldırısının gerçek karmaşıklığının 2^{401} olduğunu gösterdiler [78]. Diğer taraftan, Courtois ve Pieprzyk'in XSL saldırının IACR eprint arşivinde olan [42] nolu makalelerinde geçen sürümüyle ilgili olarak da bağımsız denklem sayısının $rSt^{P-1}\binom{S-1}{P-1}$ değil, çok daha küçük bir sayı olan $\binom{S}{P}(t^P - (t-r)^P)$ kadar olacağı gösterilmiştir [78]. Sonuç olarak XSL saldırısında elde edilen doğrusal denklemlerin taradıkları uzayın doğrusal boyutu literatürde bir tar-

tışma konusudur ve bu sistemlerin Courtois ve Pieprzyk'in iddia ettikleri kadar büyük bir uzaya sahip olmadıkları kriptoloji camiasında genel kabul görmüş bir yaklaşımdır.

9.15. SONUÇ VE DEĞERLENDİRMELER

Bu kitap bölümünde simetrik şifreleme algoritmalarına uygulanan belli başlı saldırı yöntemleri tanıtılmıştır. Simetrik şifreleme algoritmaları blok şifreleme algoritmaları ve dizi şifreleme algoritmaları olarak ikiye ayrılırlar. Günlük hayattaki bilgi güvenliği gerektiren hemen hemen bütün uygulamalarda bu algoritmalara rastlamak mümkündür. Kişisel bilgisayarlar ile bağlanılan İnternet siteleri arasındaki iletişimi korumak için kullanılan TLS protokolünde RC4, AES, IDEA, 3DES, Camellia, ChaCha20; hücresel ağ teknolojilerinde SNOW-3G dizi şifreleme ve AES blok şifreleme algoritması; Whatsapp Signal protokolünde AES; WiFi erişimi için kullanılan WEP/WPA protokollerinde RC4 ve AES algoritmaları kullanılmaktadır. Dolayısıyla simetrik şifreleme algoritmalarının güvenlik analizleri sıradan insanların dahi günlük hayatını etkileyebilecek derecede önem arz etmektedir.

Hem blok şifreleme algoritmalarına hem de dizi şifreleme algoritmalarına yapılan saldırılarda ana hedef kaba kuvvet saldırısından daha hızlı anahtarı ele geçirmektir. Bu amaçla son otuz yılda literatürde birçok kriptoanaliz yöntemi yayınlanmıştır. Bu kitap bölümünde simetrik şifreleme algoritmalarına yapılan başlıca saldırılar ele alınmış ve bu saldırılara güncel literatürden örnekler sunulmuştur.

Son zamanlarda hızla yaygınlaşan ve özellikle kısıtlı kaynak gerektiren kablosuz ağlar, RFID etiketleri, IoT (Internet of Things) gibi teknolojilerde gizliliği ve kaynak doğrulamayı sağlamak amacıyla tasarlanan modern hafifsıklet simetrik şifreleme algoritmalarının büyük bir çoğunlukla blok şifreleme algoritmaları olduğu görülmektedir. Bunun temel nedeni, dizi şifreleme algoritmalarının ödünleşim ataklarına karşı güvenlik amacıyla anahtar boy-

larının en az iki katı kadar büyük içsel duruma sahip olmaları gerekliliğidir [25,9,56]. Bu nedenle, genel olarak dizi şifreleme algoritmalarının kayan anahtar üreteçleri donanımda hafifsıklet olamayacak kadar çok yer kaplamak durumundadır. Örneğin 128 bit anahtar boyuna sahip bir dizi şifreleme algoritmasının kayan anahtar üreteci için en azından 256 bitlik bir yazmaç (register) gerekecektir. Diğer taraftan, 256 bitlik bir yazmaç değerini saklamak için en azından 1200-1500 GE'lik (Gate Equivalent) bir yonga mantık alanına ihtiyaç vardır. Bu da, donanımda alan maliyeti açısından dizi şifreleme algoritmaları için blok şifreleme algoritmalarıyla kıyaslandığında bir dezavantaj oluşturmaktadır. Son on-on beş yılda tasarlanmış ve literatürde yayınlanmış hafifsıklet simetrik şifreleme algoritmalarının büyük çoğunluğunun blok şifreleme algoritmaları olmasının temel nedeni budur.

Kısıtlı kaynak gerektiren uygulamalar için tasarlanan hafifsıklet blok şifreleme algoritmalarının güvenlik analizleri genel açık bir problemdir. Bu algoritmaların özellikle anahtar şemaları son derece basittir ve anahtar şemalarının genel kabul görmüş güvenlik ölçütlerine sahip değillerdir. Bu açıdan, anahtar şemalarının saldırılara karşı sağlamaları gereken güvenlik ölçütleri hakkında literatürde ortak bir görüş sağlanabilmesi için anahtar şeması analizlerinin belli olgunluğa ulaşmış olması gerekir.

Yakın zamanda yayınlanan bir çalışmada anahtar şemaları doğrusal fonksiyon olan blok şifreleme algoritmalarının belli çevrim sayılarında bulunmuş ayıraçlara, ilgili çevrim anahtarlarını taramak için maksimum sayıda çevrim ekleyerek olabilecek en yüksek çevrime saldırı gerçekleştirebilmek için yeni bir otomatik MILP modeli önerilmiştir [98]. Bu model SKINNY algoritmasına uygulanmış ve 30 çevrime kadar ilintili anahtar saldırısı yapılmıştır.

En yeni çalışmalardan birisi olarak, 2020 yılında anahtar şemalarının kriptografik gücünü test etmeye yönelik istatistiksel ölçütlerin öne sürüldüğü ve güçlü bir anahtar şemasının sağlaması gereken özelliklerin anlatıldığı bir

çalışma yayınlanmıştır [1]. Ancak, hafifsıklet blok şifreleme algoritmalarının hemen hepsinin de anahtar şemalarının bu ölçütleri sağlamıyor olması düşündürücüdür. İlginç olan başka bir nokta ise, bu makaleden hemen bir sene sonrasında anahtar şeması olmadan güvenli Feistel blok şifreleme tasarımının nasıl yapılabileceğine dair çalışma yayınlanmış olmasıdır [105]. Sonuç olarak anahtar şemalarının sağmaları gereken minimum güvenlik ölçütleri hala açık problemdir ve bu konuda kriptoloji camiasında ortak bir görüşoluşmamıştır.

Dizi şifreleme algoritmalarına uygulanan ödünleşim saldırılarını değerlendirilirken, özellikle kayan anahtar üreteçlerinin içsel durum vektörlerinden herhangi birisini ele geçirmeye yönelik ödünleşim ataklarının kaba kuvvet ataklarıyla kıyaslandığı görülmektedir [25,97,100,101]. İçsel durum boyunun en azından anahtar boyunun iki katı kadar olması gerekliliği bu kıyaslamanın bir neticesidir. Ancak konvansiyonel Hellman saldırısı ile dizi şifreleme algoritmaları da dahil olmak üzere, herhangi bir tek yönlü fonksiyona (one way function) ters görüntü (preimage) bulmanın çevrimiçi maliyeti fonksiyonun iç yapısından bağımsız olarak $2^{2n/3}$'tür [57]. Burada n, fonksiyonun girdi boyudur. Dolayısıyla içsel durum boyunun anahtar boyunun en azından $4/3$ katı olması yeterlidir. Bu durumda çevrimdışı (offline) saldırının kaba kuvvet saldırısından daha yavaş olabilmesi için anahtar başına şifrelenecek veri miktarının anahtar uzayının küp kökünü geçmemesi gerekecektir. Bu kabuller altında, kayan anahtar üreteçleri için uygulanan ödünleşim saldırıları kaba kuvvet saldırılarıyla değil de klasik Hellman saldırısı ile kıyaslanmalıdır. Bu yeni güvenlik eşiği son zamanlarda literatürde tartışmaya açılmıştır [52,66].

Yeni ölçütün kabul görmesiyle hafif sıklet dizi şifreleme algoritması tasarımı önemli bir aşama kaydedecektir ve küçük kayan anahtar üreteçlerinin tasarımlarının önü açılacaktır. Ancak bu durumda, içsel durumu ele geçirme saldırılarının etkili olma riski artmaktadır. Küçük içsel durumlu dizi şifreleme algoritma tasarımında literatür henüz olgunlaşmamıştır ve bu konuda

ciddi bir araştırma boşluğu mevcuttur. Önümüzdeki yıllarda küçük kayan anahtar üreteçli dizi şifreleme algoritmaları tasarımları ve kriptoanalizleri konusunda birçok çalışmanın yayınlanacağı düşünülmektedir. Bu çalışmalarla birlikte küçük dizi şifreleme algoritmalarının analizleri de hız kazanacak ve literatürde bu konuda bilgi artışı yaşanacaktır.

Yakın zamanda tasarlanmış küçük içsel durum boylu dizi şifreleme algoritmaları mevcuttur. Ancak bu algoritmalar klasik bakış açısıyla tasarlanmamış, algoritmalarda ödünleşim saldırılarına karşı güvenlik için içsel durum güncellemede ana anahtarın kullanımı da göz önünde bulundurulmuştur. Kayan anahtar üreteçlerinde ana anahtarı kullanan algoritmalara "anahtarlı güncelleme yapan kayan anahtar üreteçleri" adı verilmektedir. İçsel durum güncellemesinde anahtarı kullanan algoritmalar eğer anahtar cihazda hayat boyu hiç değiştirilmeyecekse hafifsıklet uygulamalar için verimli olabilmektedir. Ancak bu tür uygulamalar son derece kısıtlıdır.

Son on yıl içinde yayınlanmış anahtarlı güncelleme yapan algoritmalardan ilki Sprout dizi şifreleme algoritmasıdır [5]. Ancak algoritmada ciddi zayıflıklar keşfedilmiş ve birçok saldırı yayınlanmıştır [73,81,120,52]. Daha sonra Plantlet [87] ve Fruit [122] algoritmaları yayınlamıştır. Bu algoritmalara da uygulanmış son derece başarılı kriptoanalizler vardır [109,114,38]. Yakın zamanda yayınlanmış LILLE isimli dizi şifreleme algoritması da anahtarlı içsel durum güncellemesi yapmaktadır ve şu ana kadar algoritmaya başarılı bir saldırı mevcut değildir [12].

Literatürde var olan modern blok şifreleme algoritmalarının ve dizi şifreleme algoritmalarının güvenlik düzeyleri genel olarak açık sorudur ve bir algoritmaya yapılan her bir kriptoanaliz algoritmanın güvenliği hakkında literatüre yeni bir bilgi sunar. Örneğin AES algoritmasının güvenliği konusunda kriptoanaliz çalışmaları hala devam etmektedir ve literatürde yayınlar çıkmaktadır. AES'in farksal saldırıya karşı güvenli olduğu çeyrek asır önce, tasarımcıları tarafından gösterilmişti. Klasik farksal saldırıda kullanıldığında

başarısız olan karakteristiklerden bazılarının AES'in özet fonksiyonu olarak kullanımında (HCF-AES256) "rebound" saldırısının kuantum kurulumunda çakışma (collision) bulmakta kullanılabileceği 2021 başında gösterilmiştir [31]. Saldırıda kullanılan 10 çevrimlik farksal karakteristiğin ihtimali yaklaşık olarak 2^{-160}'dır ki, AES'in blok boyunun 128-bit olması nedeniyle, bu ihtimale sahip karakteristiklerle klasik farksal saldırı gerçekleştirmek mümkün değildir. Bu örnek bize AES'in farksal analiz çalışmalarının dahi günümüzde hala araştırma konusu olarak devam ettiğini göstermektedir.

Ülkemizde simetrik şifreleme algoritmalarının analizi ve tasarımı çalışmaları 1970'li yılların sonlarına dayanmaktadır. Kıbrıs Barış Harekatı'nın ardından gelen ambargo, milli savunma sanayinin gelişmesine olan ihtiyacı tetiklemiştir. Bunun bir örneği olarak özellikle Türk Silahlı Kuvvetleri'nde bilgi güvenliğinin tesisi için milli kripto cihazlarına ihtiyaç duyulmuştur. Bu ihtiyacı karşılamak için TÜBİTAK bünyesinde başlatılan projeler meyvesini vermiş ve TÜBİTAK UEKAE (Uusal Elektronik ve Kriptoloji Araştırma Enstitüsü) tarafından üretilen ilk milli kripto cihazları 80'li yılların başında TSK envanterine geçmiştir. Bu cihazlarda çalışan şifreleme algoritmalarının tasarımları ve analizleri için o dönemde UEKAE'de oluşturulan yeteneğin yarım asra yaklaşan geliştirilmesi süreci günümüzde, UEAKE'nin de bir araştırma enstitüsü olarak içinde bulunduğu TÜBİTAK BİLGEM'de (Bilişim ve Bilgi Güvenliği İleri Teknolojiler Araştırma Merkezi) yapılan Ar-Ge çalışmalarıyla hala sürdürülmektedir. Benzer Ar-Ge çalışmaları ASELSAN bünyesinde de yürütülmektedir.

Teşekkür

Dokümanın Latex formatından dönüştürülmesinde ve kaynakların bib dosyasından düzenlenmesinde katkılarından dolayı Azhar Murzaeva'ya; yapıcı geri bildirimlerde bulunan ve yardımlarını bizden esirgemeyen Sedat Akleylek'e teşekkür ederiz.

KAYNAKLAR

[1] Afzal, S., Yousaf, M., Afzal, H., Alharbe, N., & Mufti, M. R. (2020). Cryptographic Strength Evaluation of Key Schedule Algorithms. *Secur. Commun. Networks, 2020*, 3189601:1–3189601:9.

[2] Ågren, M., Löndahl, C., Hell, M., & Johansson, T. (2012). A survey on fast correlation attacks. *Cryptogr. Commun., 4*, 173–202.

[3] Bogdanov, A. &Rechberger C. (2010). A 3-Subset Meet-in-the-Middle Attack: Cryptanalysis of the Lightweight Block Cipher KTANTAN. *Selected Areas in Cryptography*, (s. 229-240).

[4] Aoki, K., & Sasaki, Y. (2009). Meet-in-the-Middle Preimage Attacks Against Reduced SHA-0 and SHA-1. S. Halevi (Dü.), *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings.* içinde *5677*, s. 70–89. Springer.

[5] Armknecht, F., & Mikhalev, V. (2015). On Lightweight Stream Ciphers with Shorter Internal States. G. Leander (Dü.), *Fast Software Encryption - 22nd International Workshop, FSE 2015, Istanbul, Turkey, March 8-11, 2015, Revised Selected Papers.* içinde *9054*, s. 451–470. Springer.

[6] Avoine, G., Bourgeois, A., & Carpent, X. (2015). Analysis of Rainbow Tables with Fingerprints. E. Foo, & D. Stebila (Dü.), *Information Security and Privacy - 20th Australasian Conference, ACISP 2015, Brisbane, QLD, Australia, June 29 - July 1, 2015, Proceedings.* içinde *9144*, s. 356–374. Springer.

[7] Avoine, G., Carpent, X., & Lauradoux, C. (2015). Interleaving Cryptanalytic Time-Memory Trade-Offs on Non-uniform Distributions. G. Pernul, P. Y. Ryan, & E. R. Weippl (Dü.), *Computer Security - ESORICS 2015 - 20th European Symposium on Research in Computer Security, Vienna, Austria, September 21-25, 2015, Proceedings, Part I.* içinde *9326*, s. 165–184. Springer.

[8] Avoine, G., Junod, P., & Oechslin, P. (2008). Characterization and Improvement of Time-Memory Trade-Off Based on Perfect Tables. *ACM Trans. Inf. Syst. Secur., 11*, 17:1–17:22.

[9] Babbage, S. (1995). Improved exhaustive search attacks on stream ciphers. Security and Detection 1995, European Convention IET, 1995.

[10] Bahrak, B., & Aref, M. (2008). Impossible differential attack on seven-round AES-128. *IET Inf. Secur., 2*, 28-32.

[11] Banik, S., Barooti, K., & Isobe, T. (2019). Cryptanalysis of Plantlet. *IACR Trans. Symmetric Cryptol., 2019*, 103–120.

[12] Banik, S., Isobe, T., & Morii, M. (2018). On Design of Robust Lightweight Stream Cipher with Short Internal State. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci., 101-A*, 99–109.

[13] Bao, Z., Guo, J., & List, E. (2020). Extended Truncated-differential Distinguishers on Round-reduced AES. *IACR Trans. Symmetric Cryptol., 2020*, 197–261.

[14] Barkan, E., Biham, E., & Shamir, A. (2006). Rigorous Bounds on Cryptanalytic Time/Memory Tradeoffs. C. Dwork (Dü.), *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006, Proceedings.* içinde *4117*, s. 1–21. Springer.

[15] Bar-On, A., Dunkelman, O., Keller, N., Ronen, E., & Shamir, A. (2018). Improved Key Recovery Attacks on Reduced-Round AES with Practical Data and Memory Complexities. H. Shacham, & A. Boldyreva (Dü.), *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II.* içinde *10992*, s. 185–212. Springer.

[16] Bieniasz, J., Skowron, K., Trzepinski, M., Rawski, M., Sapiecha, P., & Tomaszewicz, P. (2015). Hardware Implementation of Rainbow Tables Generation for Hash Function Cryptanalysis. A. Grzech, L. Borzemski, J. Swiatek, & Z. Wilimowska (Dü.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology - ISAT 2015 - Part II, Karpacz, Poland, September 20-22, 2015.* içinde *430*, s. 189–200. Springer.

[17] Biham, E. (1994). New Types of Cryptanalytic Attacks Using Related Keys. *J. Cryptol., 7*, 229–246.

[18] Biham, E., & Keller, N. (2000). Cryptanalysis of reduced variants of Rijndael. *Cryptanalysis of reduced variants of Rijndael*.

[19] Biham, E., & Shamir, A. (1990). Differential Cryptanalysis of DES-like Cryptosystems. A. Menezes, & S. A. Vanstone (Dü.), *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings.* içinde *537*, s. 2–21. Springer.

[20] Biham, E., & Shamir, A. (1992). Differential Cryptanalysis of the Full 16-Round DES. E. F. Brickell (Dü.), *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings.* içinde *740*, s. 487–496. Springer.

[21] Biham E., Biryukov A., and Shamir A. Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. In J. Stern, editor,EUROCRYPT, volume1592 ofLecture Notes in Computer Science, pages 12–23. Springer, 1999.

[22] Biham, E., Dunkelman, O., & Keller, N. (2001). The Rectangle Attack - Rectangling the Serpent. B. Pfitzmann (Dü.), *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding.* içinde *2045*, s. 340–357. Springer.

[23] Biham, E., Dunkelman, O., & Keller, N. (2002). Enhancing Differential-Linear Cryptanalysis. Y. Zheng (Dü.), *Advances in Cryptology - ASIACRYPT 2002, 8th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, New Zealand, December 1-5, 2002, Proceedings.* içinde *2501*, s. 254–266. Springer.

[24] Biryukov, A., & Khovratovich, D. (2009). Related-Key Cryptanalysis of the Full AES-192 and AES-256. M. Matsui (Dü.), *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings.* içinde *5912*, s. 1–18. Springer.

[25] Biryukov, A., & Shamir, A. (2000). Cryptanalytic Time/Memory/Data Tradeoffs for Stream Ciphers. *Advances in Cryptology - ASIACRYPT 2000*, (s. 1–13).

[26] Blondeau, C., & Nyberg, K. (2014). Links between Truncated Differential and Multidimensional Linear Properties of Block Ciphers and Underlying Attack Complexities. P. Q. Nguyen, & E. Oswald (Dü.), *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings.* içinde *8441*, s. 165–182. Springer.

[27] Blondeau, C., & Nyberg, K. (2017). Joint data and key distribution of simple, multiple, and multidimensional linear cryptanalysis test statistic and its impact to data complexity. *Des. Codes Cryptogr., 82*, 319–349.

[28] Bogdanov, A., Khovratovich, D., & Rechberger, C. (2011). Biclique Cryptanalysis of the Full AES. D. H. Lee, & X. Wang (Dü.), *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings.* içinde *7073*, s. 344–371. Springer.

[29] Boura, C., Naya-Plasencia, M., & Suder, V. (2014). Scrutinizing and Improving Impossible Differential Attacks: Applications to CLEFIA, Camellia, LBlock and Simon (Full Version). *IACR Cryptol. ePrint Arch.*

[30] Canteaut, A., & Roué, J. (2015). Differential Attacks Against SPN: A Thorough Analysis. S. E. Hajji, A. Nitaj, C. Carlet, & E. M. Souidi (Dü.), *Codes, Cryptology, and Information Security - First International Conference, C2SI 2015, Rabat, Morocco, May 26-28, 2015, Proceedings - In Honor of Thierry Berger.* içinde *9084*, s. 45–62. Springer.

[31] Chauhan, A. K., Kumar, A., & Sanadhya, S. K. (2021). Quantum Free-Start Collision Attacks on Double Block Length Hashing with Round-Reduced AES-256. *IACR Trans. Symmetric Cryptol., 2021*, 316–336.

[32] Chen, H., & Wang, X. (2016). Improved Linear Hull Attack on Round-Reduced Simon with Dynamic Key-Guessing Techniques. T. Peyrin (Dü.), *Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers.* içinde *9783*, s. 428–449. Springer.

[33] Chen, L., Wang, G., & Zhang, G. (2019). MILP-based Related-Key Rectangle Attack and Its Application to GIFT, Khudra, MIBS. *Comput. J., 62*, 1805–1821.

[34] Chen, S., & Jin, C. (2015). An Improved Way to Construct the Parity-check Equations in Fast Correlation Attacks. *J. Networks, 10*, 443–447.

[35] Cheon, J., Kim, M., Kim, K., Lee, J.-Y., & Kang, S. (2001). Improved Impossible Differential Cryptanalysis of Rijndael and CRYPTON. *ICISC*.

[36] Cid, C., & Leurent, G. (2005). An Analysis of the XSL Algorithm. B. Roy (Dü.), *Advances in Cryptology - ASIACRYPT 2005* içinde (s. 333–352). Berlin: Springer Berlin Heidelberg.

[37] Collard, B., Standaert, F.-X., & Quisquater, J.-J. (2007). Improving the Time Complexity of Matsui's Linear Cryptanalysis. K.-H. Nam, & G. Rhee (Dü.), *Information Security and Cryptology - ICISC 2007, 10th International Conference, Seoul, Korea, November 29-30, 2007, Proceedings*. içinde *4817*, s. 77–88. Springer.

[38] Copeland, J., & Simpson, L. (2020). Finding Slid Pairs for the Plantlet Stream Cipher. P. P. Jayaraman, D. Georgakopoulos, T. K. Sellis, & A. Forkan (Dü.), *Proceedings of the Australasian Computer Science Week, ACSW 2020, Melbourne, VIC, Australia, February 3-7, 2020* içinde (s. 7:1–7:7). ACM.

[39] Courtois, N. T., & Patarin, J. (2003). About the XL Algorithm over GF(2). M. Joye (Dü.), *Topics in Cryptology — CT-RSA 2003* içinde (s. 141–157). Berlin: Springer Berlin Heidelberg.

[40] Courtois, N. T., & Pieprzyk, J. (2002). Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. Y. Zheng (Dü.), *Advances in Cryptology — ASIACRYPT 2002* içinde (s. 267–287). Berlin: Springer Berlin Heidelberg.

[41] Courtois, N. T., Klimov, A., Patarin, J., & Shamir, A. (2000). Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. B. Preneel (Dü.), *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*. içinde *1807*, s. 392–407. Springer.

[42] Courtois, N., & Pieprzyk, J. (2002). Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. IACR Cryptol. ePrint Arch. 2002: 44 (2002)

[43] Daemen, J., & Rijmen, V. (2002). *The Design of Rijndael: AES - The Advanced Encryption Standard.* Information Security and Cryptography, Springer 2002, ISBN 3-540-42580-2.

[44] Daemen, J., Knudsen, L. R., & Rijmen, V. (1997). The Block Cipher Square. E. Biham (Dü.), *Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings*. içinde *1267*, s. 149–165. Springer.

[45] Demirci, H., & Selçuk, A. A. (2008). A Meet-in-the-Middle Attack on 8-Round AES. K. Nyberg (Dü.), *Fast Software Encryption, 15th International Workshop, FSE 2008, Lausanne, Switzerland, February 10-13, 2008, Revised Selected Papers*. içinde *5086*, s. 116–126. Springer.

[46] Derbez, P., & Fouque, P.-A. (2016). Automatic Search of Meet-in-the-Middle and Impossible Differential Attacks. *IACR Cryptol. ePrint Arch., 2016*, 579.

[47] Derbez, P., Fouque, P.-A., & Jean, J. (2013). Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting. T. Johansson, & P. Q. Nguyen (Dü.), *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings.* içinde *7881*, s. 371–387. Springer.

[48] Dinur, I., & Shamir, A. (2008). Cube Attacks on Tweakable Black Box Polynomials. *IACR Cryptol. ePrint Arch., 2008*, 385.

[49] Dunkelman, O., & Keller, N. (2008). An Improved Impossible Differential Attack on MISTY1. J. Pieprzyk (Dü.), *Advances in Cryptology - ASIACRYPT 2008* içinde (s. 441–454). Berlin: Springer Berlin Heidelberg.

[50] Dunkelman, O., Keller, N., Ronen, E., & Shamir, A. (2020). The Retracing Boomerang Attack. A. Canteaut, & Y. Ishai (Dü.), *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part I.* içinde *12105*, s. 280–309. Springer.

[51] Eichlseder, M., Leander, G., & Rasoolzadeh, S. (2020). Computing Expected Differential Probability of (Truncated) Differentials and Expected Linear Potential of (Multidimensional) Linear Hulls in SPN Block Ciphers. K. Bhargavan, E. Oswald, & M. Prabhakaran (Dü.), *Progress in Cryptology - INDOCRYPT 2020 - 21st International Conference on Cryptology in India, Bangalore, India, December 13-16, 2020, Proceedings.* içinde *12578*, s. 345–369. Springer.

[52] Esgin, M. F., & Kara, O. (2015). Practical cryptanalysis of Full Sprout with TMD Tradeoff Attacks. O. Dunkelman, & L. Keliher (Dü.), *Selected Areas in Cryptography - SAC 2015 - 22nd International Conference, Sackville, NB, Canada, August 12-14, 2015, Revised Selected Papers.* içinde *9566*, s. 67–85. Springer.

[53] Ferguson, N., Kelsey, J., Lucks, S., Schneier, B., Stay, M., Wagner, D. A., et al. (2000). Improved Cryptanalysis of Rijndael. B. Schneier (Dü.), *Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings.* içinde *1978*, s. 213–230. Springer.

[54] Flórez-Gutiérrez, A., & Naya-Plasencia, M. (2020). Improving Key-Recovery in Linear Attacks: Application to 28-Round PRESENT. A. Canteaut, & Y. Ishai (Dü.), *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part I.* içinde *12105*, s. 221–249. Springer.

[55] Ghafari, V. A., & Hu, H. (2018). Fruit-80: A Secure Ultra-Lightweight Stream Cipher for Constrained Environments. *Entropy, 20*, 180.

[56] Golić, J. D. (1997). Cryptanalysis of Alleged A5 Stream Cipher. *EUROCRYPT '97*, (s. 239–255).

[57] Hellman, M. E. (1980). A cryptanalytic time-memory trade-off. *IEEE Transactions on Information Theory, 26*, 401–406.

[58] Hermelin, M., Cho, J. Y., & Nyberg, K. (2019). Multidimensional Linear Cryptanalysis. *J. Cryptol., 32*, 1–34.

[59] Horalek, J., Holík, F., Horák, O., Petr, L., & Sobeslav, V. (2017). Analysis of the use of Rainbow Tables to break hash. *J. Intell. Fuzzy Syst., 32*, 1523–1534.

[60] Huang, J., Vaudenay, S., Lai, X., & Nyberg, K. (2015). Capacity and Data Complexity in Multidimensional Linear Attack. R. Gennaro, & M. Robshaw (Dü.), *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I.* içinde *9215*, s. 141–160. Springer.

[61] Jakobsen, T., & Knudsen, L. R. (1997). The Interpolation Attack on Block Ciphers. E. Biham (Dü.), *Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings.* içinde *1267*, s. 28–40. Springer.

[62] Jiao, L., Hao, Y., & Li, Y. (2019). Improved guess-and-determine attack on TRIVIUM. *IET Inf. Secur., 13*, 411–419.

[63] Kara, O. (2008). Reflection Cryptanalysis of Some Ciphers. D. R. Chowdhury, V. Rijmen, & A. Das (Dü.), *Progress in Cryptology - INDOCRYPT 2008, 9th International Conference on Cryptology in India, Kharagpur, India, December 14-17, 2008. Proceedings.* içinde *5365*, s. 294–307. Springer.

[64] Kara, O. (2021). Tradeoff attacks on symmetric ciphers. *IACR Cryptol. ePrint Arch., 2021*, 319.

[65] Kara, O. (2021). Tradeoff attacks on symmetric ciphers. R. Bernardini (Dü.), *Cryptography-Recent Advances and Future Developments.* içinde London: IntechOpen, DOI: 10.5772/intechopen.96627. Avaliable at [Online First] https://www.intechopen.com/online-first/tradeoff-attacks-on-symmetric-ciphers. To be published.

[66] Kara, O., & Esgin, M. F. (2019). On Analysis of Lightweight Stream Ciphers with Keyed Update. *IEEE Trans. Computers, 68*, 99–110.

[67] Khovratovich, D., Rechberger, C., & Savelieva, A. (2012). Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 Family. A. Canteaut (Dü.), *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers.* içinde *7549*, s. 244–263. Springer.

[68] Kim, J., Hong, S., Preneel, B., Biham, E., Dunkelman, O., & Keller, N. (2012). Related-Key Boomerang and Rectangle Attacks: Theory and Experimental Analysis. *IEEE Trans. Inf. Theory, 58*, 4948–4966.

[69] Knellwolf, S., & Khovratovich, D. (2012). New Preimage Attacks against Reduced SHA-1. R. Safavi-Naini, & R. Canetti (Dü.), *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings.* içinde *7417*, s. 367–383. Springer.

[70] Knudsen, L. R. (1994). Truncated and Higher Order Differentials. B. Preneel (Dü.), *Fast Software Encryption: Second International Workshop. Leuven, Belgium, 14-16 December 1994, Proceedings.* içinde *1008*, s. 196–211. Springer.

[71] Knudsen, L. R., & Miolane, C. V. (2010). Counting equations in algebraic attacks on block ciphers. *Int. J. Inf. Sec., 9*, 127–135.

[72] Knudsen, L. R., & Wagner, D. A. (2002). Integral Cryptanalysis. J. Daemen, & V. Rijmen (Dü.), *Fast Software Encryption, 9th International Workshop, FSE 2002, Leuven, Belgium, February 4-6, 2002, Revised Papers.* içinde *2365*, s. 112–127. Springer.

[73] Lallemand, V., & Naya-Plasencia, M. (2015). Cryptanalysis of Full Sprout. IACR Cryptol. ePrint Arch. 2015*: 232 (2015)*

[74] Langford, S. K., & Hellman, M. E. (1994). Differential-Linear Cryptanalysis. *CRYPTO*, (s. 17-25).

[75] Leander, G. (2011). On Linear Hulls, Statistical Saturation Attacks, PRESENT and a Cryptanalysis of PUFFIN. K. G. Paterson (Dü.), *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings.* içinde *6632*, s. 303–322. Springer.

[76] Li, L., Jia, K., Wang, X., & Dong, X. (2015). Meet-in-the-Middle Technique for Truncated Differential and Its Applications to CLEFIA and Camellia. G. Leander (Dü.), *Fast Software Encryption - 22nd International Workshop, FSE 2015, Istanbul, Turkey, March 8-11, 2015, Revised Selected Papers.* içinde *9054*, s. 48–70. Springer.

[77] Li, R., Sun, B., & Li, C. (2011). Impossible differential cryptanalysis of SPN ciphers. *IET Inf. Secur., 5*, 111-120.

[78] Lim, C.-W., & Khoo, K. (2007). An Analysis of XSL Applied to BES. A. Biryukov (Dü.) içinde, *Fast Software Encryption, 14th International Workshop, FSE 2007, Luxembourg, Luxembourg, March 26-28, 2007, Revised Selected Papers* (Cilt 4593, s. 242–253). Springer.

[79] Lu, J., Dunkelman, O., Keller, N., & Kim, J. (2008). New Impossible Differential Attacks on AES. D. R. Chowdhury, V. Rijmen, & A. Das (Dü.), *Progress in Cryptology - INDOCRYPT 2008, 9th International Conference on Cryptology in India, Kharagpur, India, December 14-17, 2008. Proceedings.* içinde *5365*, s. 279–293. Springer.

[80] Lu, J., Kim, J., Keller, N., & Dunkelman, O. (2008). Improving the Efficiency of Impossible Differential Cryptanalysis of Reduced Camellia and MISTY1. T. Malkin (Dü.), *Topics in Cryptology - CT-RSA 2008, The Cryptographers' Track at the RSA Conference 2008, San Francisco, CA, USA, April 8-11, 2008. Proceedings.* içinde *4964*, s. 370–386. Springer.

[81] Maitra, S., Sarkar, S., Baksi, A., & Dey, P. (2015). Key Recovery from State Information of Sprout: Application to Cryptanalysis and Fault Attack. *IACR Cryptol. ePrint Arch., 2015*, 236.

[82] Mala, H., Dakhilalian, M., Rijmen, V., & Modarres-Hashemi, M. (2010). Improved Impossible Differential Cryptanalysis of 7-Round AES-128. *INDOCRYPT.*

[83] Matsui, M. (1993). Linear Cryptanalysis Method for DES Cipher. T. Helleseth (Dü.), *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings.* içinde *765*, s. 386–397. Springer.

[84] Maximov, A., & Biryukov, A. (2007). Two Trivial Attacks on Trivium. C. M. Adams, A. Miri, & M. J. Wiener (Dü.), *Selected Areas in Cryptography, 14th International Workshop, SAC 2007, Ottawa, Canada, August 16-17, 2007, Revised Selected Papers.* içinde *4876*, s. 36–55. Springer.

[85] Meier, W. (2011). Fast Correlation Attacks: Methods and Countermeasures. A. Joux (Dü.), *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers.* içinde *6733*, s. 55–67. Springer.

[86] Meier, W., & Staffelbach, O. (1989). Fast Correlation Attacks on Certain Stream Ciphers. *J. Cryptol., 1*, 159–176.

[87] Mikhalev, V., Armknecht, F., & Müller, C. (2016). On Ciphers that Continuously Access the Non-Volatile Key. *IACR Trans. Symmetric Cryptol., 2016*, 52–79.

[88] Mileva, A., Dimitrova, V., Kara, O., & Mihaljević, M. J. (2021). Catalog and Illustrative Examples of Lightweight Cryptographic Primitives. G. Avoine, & J. Hernandez-Castro (Dü.), *Security of Ubiquitous Computing Systems: Selected Topics* içinde (s. 21–47). Cham: Springer International Publishing.

[89] Matsui, M. (1994). The First Experimental Cryptanalysis of the Data Encryption Standard. *The First Experimental Cryptanalysis of the Data Encryption Standard*, 1–11. (Y. Desmedt, Dü.) Santa Barbara, California, USA.

[90] Moghaddam, A. E., & Ahmadian, Z. (2020). New Automatic Search Method for Truncated-Differential Characteristics Application to Midori, SKINNY and CRAFT. *Comput. J., 63*, 1813–1825.

[91] Murphy, S., & Robshaw, M. J. (2002). Essential Algebraic Structure within the AES. M. Yung (Dü.) içinde, *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings* (Cilt 2442, s. 1–16). Springer.

[92] Nyberg, K. (1994). Linear Approximation of Block Ciphers. A. D. Santis (Dü.), *Advances in Cryptology - EUROCRYPT '94, Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9-12, 1994, Proceedings.* içinde *950*, s. 439–444. Springer.

[93] Oechslin, P. (2003). Making a Faster Cryptanalytic Time-Memory Trade-Off. D. Boneh (Dü.), *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings.* içinde *2729*, s. 617–630. Springer.

[94] Özen, M., Çoban, M., & Karakoç, F. (2015). A Guess-and-Determine Attack on Reduced-Round Khudra and Weak Keys of Full Cipher. *IACR Cryptol. ePrint Arch., 2015*, 1163.

[95] Park, S., Sung, S. H., Lee, S., & Lim, J. (2003). Improving the Upper Bound on the Maximum Differential and the Maximum Linear Hull Probability for SPN Structures and AES. T. Johansson (Dü.), *Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, Sweden, February 24-26, 2003, Revised Papers.* içinde *2887*, s. 247–260. Springer.

[96] Phan, R. C.-W. (2004). Impossible differential cryptanalysis of 7-round Advanced Encryption Standard (AES). *Inf. Process. Lett., 91*, 33–38.

[97] Preneel, B. (2005). NESSIE Project. H. C. van Tilborg (Dü.) içinde, *Encyclopedia of Cryptography and Security* (s. 408–413). Boston, MA: Springer US.

[98] Qin, L., Dong, X., Wang, X., Jia, K., & Liu, Y. (2021). Automated Search Oriented to Key Recovery on Ciphers with Linear Key Schedule Applications to Boomerangs in SKINNY and ForkSkinny. *IACR Trans. Symmetric Cryptol., 2021*, 249–291.

[99] Rasoolzadeh, S., Ahmadian, Z., Salmasizadeh, M., & Aref, M. R. (2014). Total break of Zorro using linear and differential attacks. *ISC Int. J. Inf. Secur., 6*, 23–34.

[100] Rijmen, V. (2010). Stream ciphers and the eSTREAM project. *ISC Int. J. Inf. Secur., 2*, 3–11.

[101] Robshaw, M. (2008). The eSTREAM Project. M. J. Robshaw, & O. Billet (Dü) içinde, *New Stream Cipher Designs - The eSTREAM Finalists* (Cilt 4986, s. 1–6). Springer.

[102] Röck, A., & Nyberg, K. (2013). Generalization of Matsui's Algorithm 1 to linear hull for key-alternating block ciphers. *Des. Codes Cryptogr., 66*, 175–193.

[103] Sakallı M. T. (2019). Kriptografik test yöntemleri ve kriptoanaliz. In Siber Güvenlik ve Savunma, Problem ve Çözümleri, BGD Siber Güvenlik ve Savunma Kitap Serisi 2, pages 87–137, Ankara, 2019. Grafiker Yayınları.

[104] Sasaki, Y., & Todo, Y. (2016). New Impossible Differential Search Tool from Design and Cryptanalysis Aspects - Revealing Structural Properties of Several Ciphers. *IACR Cryptol. ePrint Arch., 2016*, 1181.

[105] Shen Y., Yan H., Wang L., and Lai X. Secure key-alternating feistel ciphers without key schedule. Sci. China Inf. Sci., 64(1), 2021.

[106] Siegenthaler T.(1985). Decrypting a class of stream ciphers using ciphertext only. IEEE Trans. Computers, 34(1):81–85, 1985.

[107] Sun, L., Gérault, D., Wang, W., & Wang, M. (05 Oct 2020). On the Usage of Deterministic (Related-Key) Truncated Differentials and Multidimensional Linear Approximations for SPN Ciphers. *IACR Trans. Symmetric Cryptol., 2020*, 262–287.

[108] Tezcan, C. (2010). The Improbable Differential Attack: Cryptanalysis of Reduced Round CLEFIA. G. Gong, & K. C. Gupta (Dü.), *Progress in Cryptology - INDOCRYPT 2010 - 11th International Conference on Cryptology in India, Hyderabad, India, December 12-15, 2010. Proceedings.* içinde *6498*, s. 197–209. Springer.

[109] Todo Y., Meier W., and Aoki K.(2019). On the data limitation of small-state stream ciphers: Correlation attacks on fruit-80 and plantlet. In K. G. Paterson andD. Stebila, editors, Selected Areas in Cryptography - SAC 2019, pages 365–392. Springer, 2019.

[110] Tsunoo, Y., Tsujihara, E., Shigeri, M., Saito, T., Suzaki, T., & Kubo, H. (2008). Impossible Differential Cryptanalysis of CLEFIA. K. Nyberg (Dü.), *Fast Software Encryption* içinde (s. 398–411). Berlin: Springer Berlin Heidelberg.

[111] Tunstall, M. (2012). Improved "Partial Sums"-based Square Attack on AES. P. Samarati, W. Lou, & J. Zhou (Dü.), *SECRYPT 2012 - Proceedings of the International Conference on Security and Cryptography, Rome, Italy, 24-27 July, 2012, SECRYPT is part of ICETE - The International Joint Conference on e-Business and Telecommunications* içinde (s. 25–34). SciTePress.

[112] Verdult, R., Garcia, F. D., & Balasch, J. (2012). Gone in 360 Seconds: Hijacking with Hitag2. T. Kohno (Dü.), *Proceedings of the 21th USENIX Security Symposium, Bellevue, WA, USA, August 8-10, 2012* içinde (s. 237–252). USENIX Association.

[113] Wagner, D. A. (1999). The Boomerang Attack. L. R. Knudsen (Dü.), *Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings.* içinde *1636*, s. 156–170. Springer.

[114] Wang, S., Liu, M., Lin, D., & Ma, L. (2019). Fast Correlation Attacks on Grain-like Small State Stream Ciphers and Cryptanalysis of Plantlet, Fruit-v2 and Fruit-80. *IACR Cryptol. ePrint Arch., 2019*, 763.

[115] Wenling Wu, & Feng, D. (2006). Impossible Differential Cryptanalysis of ARIA and Camellia. *Impossible Differential Cryptanalysis of ARIA and Camellia*.

[116] Whitfield Diffie and Martin E. Hellman. (1977). Special feature exhaustive cryptanalysis of the NBS Data Encryption Standard. *Special feature exhaustive cryptanalysis of the NBS Data Encryption Standard*.

[117] Wu, H., & Preneel, B. (2007). Differential-Linear Attacks Against the Stream Cipher Phelix. A. Biryukov (Dü.) içinde, *Fast Software Encryption, 14th International Workshop, FSE 2007, Luxembourg, Luxembourg, March 26-28, 2007, Revised Selected Papers* (Cilt 4593, s. 87–100). Springer.

[118] Wu, S., & Wang, M. (2012). Automatic Search of Truncated Impossible Differentials for Word-Oriented Block Ciphers. S. D. Galbraith, & M. Nandi (Dü.), *Progress in Cryptology - INDOCRYPT 2012, 13th International Conference on Cryptology in India, Kolkata, India, December 9-12, 2012. Proceedings.* içinde *7668*, s. 283–302. Springer.

[119] Ying, H.-M., & Kunihiro, N. (2016). Decryption of Frequent Password Hashes in Rainbow Tables. *Fourth International Symposium on Computing and Networking, CANDAR 2016, Hiroshima, Japan, November 22-25, 2016* (s. 655–661). IEEE Computer Society.

[120] Zhang, B., & Gong, X. (2015). Another Tradeoff Attack on Sprout-Like Stream Ciphers. T. Iwata, & J. H. Cheon (Dü.), *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II.* içinde *9453*, s. 561–585. Springer.

[121] Zhang, B., Gong, X., & Meier, W. (2017). Fast Correlation Attacks on Grain-like Small State Stream Ciphers. *IACR Trans. Symmetric Cryptol., 2017*, 58–81.

[122] Zhang, C., & Dong, L. (2020). An Improved Fast Correlation Attack on Fruit-80. L. Barolli, F. Amato, F. Moscato, T. Enokido, & M. Takizawa (Dü.), *Advanced Information Networking and Applications - Proceedings of the 34th International Conference on Advanced Information Networking and Applications, AINA-2020, Caserta, Italy, 15-17 April.* içinde *1151*, s. 1426–1436. Springer.

[123] Zhao, B., Dong, X., Meier, W., Jia, K., & Wang, G. (2020). Generalized related-key rectangle attacks on block ciphers with linear key schedule: applications to SKINNY and GIFT. *Des. Codes Cryptogr., 88*, 1103–1126.

[124] Biryukov, A. & Wagner, D. (1999). Slide Attacks. Fast Software Encryption, 6th International Workshop, FSE '99, Rome Italy, March 24-26, 1999.

[125] Grassi L., Kales D., Khovratovich D., Roy A., Rechberger C., and Schofnegger M. (2019).Starkad and Poseidon: New hash functions for zero knowledge proof systems. IACR Cryptol. ePrint Arch., 2019:458, 2019.

[126] Zhang W., Wu W., and Feng D.(2007). New results on impossible differential cryptanalysis of reduced AES. In K. Nam and G. Rhee, editors, Information Security and Cryptology - ICISC 2007, 10th International Conference, Seoul, Korea, November 29-30, 2007, Proceedings, volume 4817 of Lecture Notes in Computer Science, pages 239–250. Springer, 2007.

[127] Selçuk A.A.(2008). On probability of success in linear and differential cryptanalysis. J. Cryptol., 21(1):131–147, 2008.

[128] Tardif F.(2019). Practical Considerations on Cryptanalytic Time-Memory Trade-Offs.(Considérations Pratiques sur les Compromis Temps-Mémoire Cryptanalytiques). PhD thesis, University of Rennes 1, France, 2019.

[129] Biham E. and Shamir A.(1991). Differential cryptanalysis of DES-like cryptosystems. J. Cryptol., 4(1):3–72, 1991.

Bölüm 10

SIR PAYLAŞIM ŞEMALARI VE BLOKZİNCİR

Ahmet Sınak

Bu bölümde, sır paylaşım şemalarının lineer kodlar üzerindeki tasarım yöntemleri ele alınacak ve bu şemaların blokzincir sistemlerindeki kullanım alanları verilecektir. Sır paylaşım şeması gizli bilgilerin birden çok kişi arasında güvenli bir şekilde paylaştırılmasını, saklanmasını ve istenildiği zaman yetkili kişiler tarafından tekrar elde edilmesini sağlayan kriptografik bir protokoldür. Korunması gereken önemli bir bilginin, zarar görmesi, değiştirilmesi veya kaybolması gibi olası durumlar düşünüldüğünde bu bilgiyi bir tek kişide veya hafızada tutmak yerine paylaşım şemaları ile farklı yerlere dağıtmak bilginin güvenliğini artırmaktadır. Sır paylaşım şemaları genellikle şifreleme ve imzalama anahtarlarını blokzincir ağındaki kullanıcılar arasında güvenli bir şekilde paylaştırmak için kullanılmaktadır. Bazı blokzincir sistemlerinde ise kullanıcı ve veri gizliliğini artırmak için çok taraflı hesaplama protokolleriyle birlikte kullanılmaktadır. Ayrıca, bu şemalar büyük veri depolayan blokzincir sistemlerinde ve veri tabanlarında, MDS kodlarla birlikte depolama ve iletişim maliyetlerini iyileştirmek için kullanılmaktadır. Bu kitap bölümünün temel amacı, kod tabanlı sır paylaşım şemalarının tasarım yöntemlerini tanıtmak ve bu şemaların blokzincir sistemlerindeki kullanım amaçlarını açıklamaktır. Ayrıca, verilen teorik tanımlamaları ve yöntemleri daha anlaşılır kılmak için çeşitli örnekler verilmektedir. Bu örneklerde MDS kodlardan mükemmel ve ideal sır paylaşım şemaları elde edilirken minimal kodların dual kodlarından demokratik sır paylaşım şemaları elde edilmektedir.

10.1. GİRİŞ

Bu bölümde, sır paylaşım şemalarına neden ihtiyaç duyulduğu örneklerle açıklanacak ve bu şemaların kullanım alanlarından bahsedilecektir.

Gerçek hayatta bazen gizli bilgilerin (şifreleme anahtarı, imzalama anahtarı, vb.) bir grupta güvenli bir şekilde paylaştırılması ve ihtiyaç duyulduğunda tekrar elde edilmesi gerekmektedir. Örneğin, bir kurumda şifreli bir bilginin gizli anahtarının kurum yöneticilerinden oluşan bir grupta paylaştırılması ve bu gruptan en az iki kişinin bir araya gelerek anahtarı tekrar elde etmesi istensin. Gizli anahtar grup üyeleri arasında bu şekilde nasıl paylaştırılabilir? Eğer gruptaki her üyeye anahtarın bir kopyası dağıtılırsa,

- anahtar üçüncü kişilerin eline geçebilir (herhangi bir grup üyesinin anahtarı çaldırması veya bir başkasına vermesi durumu); ve/veya
- herhangi bir grup üyesi diğer grup üyelerinden habersiz şifreyi tek başına çözebilir (bir grup üyesinin dürüst olmaması durumu).

Diğer taraftan, grup üyelerinin anahtarı küçük parçalara ayırarak aralarında paylaştıklarını düşünelim. Örneğin, 256 bitlik bir anahtar 8 kişiye paylaştırılacak şekilde her biri 32 bit olan 8 parçaya bölünsün ve her kişi kendisine verilen 32 bitlik parçayı saklasın. Anahtarın tekrar elde edilmesi için bu 8 kişinin anahtar parçalarını birleştirmesi gerekmektedir. Bu durumda, en az bir kişi kendisine verilen parçayı kaybettiği zaman veya herhangi bir nedenle vermediği zaman diğer kişiler anahtarı elde edemezler. Burada belirtmek gerekir ki bu diğer kişiler anahtar hakkında kısmi bilgiye sahiptir. Örneğin, 1 kişi hariç diğer 7 kişi anahtar parçalarını birleştirdikleri zaman anahtarın 224 bitini elde eder ve geri kalan parçasını da en kötü ihtimalle 2^{32} deneme yaparak bulabilirler. Aslında bu 8 kişinin her biri anahtar hakkında kısmi bilgiye sahiptir (bir kişi anahtarın 32 bitini bildiği için anahtarın geri kalan parçasını en kötü ihtimalle 2^{224} deneme yaparak bulabilir). Ancak, bu grupta yer almayan bir kişi anahtar hakkında hiçbir bilgiye sahip değildir. Diğer bir ifadeyle, dışardan bir kişi anahtarın hiçbir bit değerini bilmediğinden bu kişinin anahtarı bulmak için 2^{256} farklı ihtimali denemesi gerekmektedir

(anahtarın 256 bit olduğu biliniyor). Bu grup içerisindeki herhangi bir kişinin anahtarı elde etmek için yapması gereken iş grup dışından herhangi birine göre daha az olmasından dolayı bu sistem güvenli değildir. Görüldüğü gibi yukarıda sunulan çözümler yeni güvenlik sorunlarına ve pratik kullanımda istenilmeyen yeni problemlere neden olmaktadır. Bu ve benzeri problemlere yol açmadan anahtarı grup üyeleri arasında güvenli bir şekilde paylaştırabilen kriptografik bir yöntem mevcuttur. Bu yöntem, anahtarı parçalara ayırmak ya da bütün olarak dağıtmak yerine anahtarla ilişkili yeni parçalar üreterek bu parçaları grup üyelerine dağıtır. Daha sonra, parçalardan bazılarını (veya tamamını) birleştirerek kurtarma protokolü ile tekrar bu anahtarı elde eder. Bu sistemde anahtar bir bütün olarak hiçbir yerde (herhangi bir grup üyesinde veya bilgisayar hafızasında) kayıt altında tutulmamaktadır. Dolayısıyla, dağıtılan parçalardan sistemin gerekli kıldığı kadarını birleştirmeden anahtarı elde etmek mümkün değildir; hatta kısmi bilgi dahi elde edilemez. Ayrıca, grup üyeleri de kendilerine tahsis edilen parçalardan anahtar hakkında hiçbir bilgi edinemezler. Bu sistemin diğer bir avantajı ise, grup üyelerinden bazıları parçalarını kaybetmiş olsalar bile belirli sayıdaki parçalar birleştirilerek anahtar elde edilebilmektedir. Bu avantajlara sahip olan kriptografik yöntem *sır paylaşım şeması (secret sharing scheme - SSS)* olarak adlandırılır.

Sır paylaşım şeması, gizli bir bilginin bir bütün olarak sadece bir kişi tarafından saklanması yerine birçok kişi arasında paylaştırılarak saklanmasını sağlayan bir sistemdir. Daha açık ifade edersek, sır paylaşım şemaları gizli bir bilginin grup üyeleri arasında güvenli bir şekilde paylaştırılmasına ve istenildiği zaman yetkili üyeler (grup üyelerinin tamamı veya bir kısmı) tarafından tekrar elde edilmesine olanak sağlayan bir sistemdir. Gerçek hayatta, korunması gereken gizli bir bilginin, ifşa edilmesi, zarar görmesi, değiştirilmesi veya kaybolması gibi olası durumlar düşünüldüğünde bilginin bir bütün olarak bir tek kişide veya hafızada tutulmasının çok riskli olduğu bilinmektedir. Gizli bilgiyi bir tek kişide tutmak yerine bu bilgiden elde edilen parçaları farklı kişilerde/ortamlarda tutmak bilginin güvenliğini artırmaktadır.

Bu bölümde son olarak, kriptografik yöntemlerin sağladığı güvenlik (security) ve gizlilik (privacy) kavramları verilmektedir. Kriptografide, güvenlik

terimi ile birbirini tamamlayan mahremiyet (Confidentiality), bütünlük (Integrity) ve erişilebilirlik (Availability/Accessibility) kavramları ifade edilmektedir. Bu kavramlar, kısaca CIA olarak da adlandırılmaktadır. Gizlilik terimiyle hem kullanıcı gizliliği (anonimlik) hem de veri gizliliği kastedilmektedir.

10.1.1. Sır Paylaşım Şemalarının Kullanım Alanları

Sır paylaşım şemaları; ilk olarak gizli anahtarları (şifreleme ve imzalama anahtarlarını) bir kişide veya bilgisayarda bir bütün olarak saklamak yerine birçok kişi arasında paylaştırmak amacıyla ortaya çıkmıştır. Daha sonra, bu şemalar gerçek kullanımda sağladıkları önemli avantajlardan dolayı birçok alanda kullanılmaya başlanmıştır. En önemli kullanım alanları arasında eşik kriptografisi (threshold cryptography), Bizans anlaşması (Byzantine agreement), erişim kontrolü, bankacılık sistemleri, nükleer silahların kontrolü, elektronik ticaret, veri tabanları, dağıtık depolama sistemleri ve blokzincir sistemleri sayılabilir.

Bu şemaların kriptografi alanındaki kullanımları arasında anahtar dağıtımı, öznitelik tabanlı şifreleme (attribute-based encryption), bulut bilişim, habersiz aktarım (oblivious transfer - OT), sıfır bilgi ispatı (zero-knowledge proof - ZKP), güvenli çok taraflı hesaplama (secure multiparty computation - sMPC) ve elektronik oylama protokolleri sayılabilir. Kriptografi alanındaki en yaygın kullanım amacı şifreleme ve imzalama anahtarlarını ilgili katılımcılar arasında paylaştırmaktır. Benzer şekilde, bulut bilişimde gizli anahtarları çevrimiçi sunucular arasında paylaştırmak için yine bu şemalar kullanılmaktadır. Bu sayede, gizli anahtarlar bir bütün olarak hiçbir yerde (çevrimiçi sunucularda veya kullanıcılarda) bulunmaz. Diğer taraftan, elektronik oylama protokollerinde ise bu şemalar kullanılarak her bir oy kullanıcısının gizli oyu parçalara ayrılır ve protokol içerisinde parçalar halinde tutulur. Böylece kullanıcıların oy tercihleri bir bütün olarak sistem içerisinde yer almaz ve güvenli bir şekilde sayım aşamasına gönderilir. Literatürde sır paylaşım şeması kullanan birçok elektronik oylama protokolleri tasarlanmıştır. Örneğin, Bartolucci ve ark. [1] blokzincir sistemi üzerinde tasarladıkları güvenli elektronik oylama protokolünde Shamir sır paylaşım şemasını kullanmışlardır.

Sır paylaşım şemalarının yaygın olarak kullanıldığı diğer bir alan dağıtık veri tabanlarıdır. Bazı veri tabanlarında bu şemalar veri gizliliğini iyileştirmek için sMPC protokolleri ile birlikte kullanılmaktadır. Ayrıca, güvenli veri depolayan veri tabanlarında bu şemaların yanı sıra simetrik şifreleme algoritmaları da kullanılmaktadır. Verileri güvenli ve dağıtık bir şekilde depolamak için sır paylaşım şemaları ile şifreleme algoritmalarını birlikte kullanan birçok veri tabanı önerilmiştir [2] [3]. Enigma [2] dağıtık veri depolayabilen ve güvenli hesaplama yapabilen merkeziyetsiz (decentralized) bir hesaplama platformudur. Bu veri tabanı, geleneksel blokzincir sistemlerinde ağ dışı eklenti olarak kullanılarak bu sistemlerin gizlilik ve ölçeklenebilirlik (scalability) ihtiyaçlarını karşılayabilmektedir. Enigma, veri sorgulamalarını sMPC protokolü kullanarak güvenilir üçüncü taraf olmadan gizli ve dağıtık bir şekilde yapmaktadır. Bu veri tabanında, hesaplama işlemlerinde gizliliği sağlamak için sMPC protokolü içerisinde işlenen veriler doğrulanabilir sır paylaşım şeması ile kullanıcılar arasında paylaştırılmaktadır. Kullanılan sır paylaşım şeması sayesinde veriler katılımcılar arasında güvenli bir şekilde paylaştırılmakta ve diğer kullanıcılara hiçbir bilgi sızdırılmadan hesaplama işlemleri yapılabilmektedir. Choi ve ark. [3] tarafından önerilen sunucu tabanlı dağıtık veri depolama sisteminde verilerin güvenliğini sağlamak ve depolama maliyetini düşürmek için sır paylaşım şeması ile birlikte AES şifreleme algoritması kullanılmıştır. Bu önerilen sistemde veriler AES algoritması ile şifrelendikten sonra sır paylaşım şeması kullanılarak kullanıcılar ve sunucular arasında dağıtılmaktadır. Ayrıca, AES algoritmasının şifreleme anahtarı sır paylaşım şeması ile paylaşılmaktadır. Böylece, önerilen veri tabanı üzerinde kullanıcı verileri güvenli bir şekilde dağıtık olarak depolanabilmektedir.

Sır paylaşım şemalarının en güncel kullanım alanı blokzincir sistemleridir. Bu şemaların blokzincir sistemlerindeki kullanım amacı, gizli anahtarları blokzincir ağındaki katılımcılar arasında paylaştırmaktır. Örneğin, blokzincir sisteminde elektronik imzalama algoritmasının imza anahtarı eşik sır paylaşım şeması ile blokzincir ağındaki katılımcılar arasında paylaştırılarak eşik imzalama şeması oluşturulabilmektedir. Bu sayede, sistem tarafından yetkilendirilmiş katılımcıların blokzincir ağındaki diğer katılımcılar adına imza işlemini gerçekleştirmesi mümkün olabilmektedir [4] [5]. Diğer taraftan, blokzincir ağındaki simetrik şifreleme algoritmasının şifreleme anahtarını katılımcılar

arasında paylaştırmak için de sır paylaşım şemaları kullanılmaktadır. Örneğin, bitcoin kripto cüzdanlarının şifreleme anahtarları sır paylaşım şeması ile kullanıcılar arasında paylaştırılmakta ve bu anahtar bir bütün olarak hiçbir yerde tutulmamaktadır [6]. Ayrıca, bazı blokzincir sistemlerinde ve dağıtık veri tabanlarında işlem verilerinin gizliliğini artırmak ve kullanıcıların anonimliğini iyileştirmek için sıfır bilgi ispatı protokolleri ile birlikte sır paylaşım şemaları kullanılmaktadır. Örneğin, Zerocoin ve Zerocash blokzincirlerinde kullanıcı ile kullanıcının işlem verisi arasındaki bağlantının izlenememesi için sıfır bilgi ispatı protokolleri kullanılmaktadır [4].

Sır paylaşım şemalarının blokzincir sistemlerindeki diğer bir kullanım amacı ise bu sistemlerin depolama maliyetlerini iyileştirmektir. Blokzincir sistemleri transfer verilerini dağıtık bir deftere kaydeder ve bu defter her bir katılımcının tüm verilerinin bir kopyasını depolar; bu yüzden sistemde zincir büyüdükçe depolama sorunları ortaya çıkmaktadır. Dolayısıyla, blokzincir sistemlerinde blok verilerinin depolanması büyük bir problem teşkil etmektedir. Bu sistemlerde blok verilerini ve varsa bloğun şifreleme anahtarını katılımcılar arasında paylaştırmak için MDS kodlar (özellikle Reed-Solomon kodlar) ve sır paylaşım şemaları birlikte kullanılmaktadır. Böylece, bu şemalar büyük verilerin depolandığı sistemlerde depolama maliyetlerinin iyileştirilmesine katkı sağlamaktadır. Literatürde, blokzincir sistemlerinin depolama maliyetlerini iyileştirmek için çeşitli sır paylaşım şemaları önerilmiştir [7] [8] [9] [10] (bu çalışmalarda önerilen sır paylaşım şemaları Bölüm 10.7'de verilecektir). Son olarak ifade edelim ki blokzincir sistemlerinde yaygın olarak kullanılan veri (data) kelimesi ile veri tabanındaki veri, işlem verisi, transfer verisi, blok verisi ve akıllı sözleşme verisi gibi veriler kastedilmektedir.

Bu kitap bölümünün temel amacı; sır paylaşım şemalarının lineer kodlarla ilişkilerini incelemek, ve bu şemaların dağıtık veri tabanlarındaki ve blokzincir sistemlerindeki kullanım amaçlarını açıklamaktır. Öncelikle, literatürde öne çıkan lineer kodlar üzerindeki sır paylaşım şeması tasarım yöntemleri verilecektir. Bu yöntemlerden çeşitli sır paylaşım şemaları elde edilecek ve örnekler verilecektir. Ayrıca, sır paylaşım şemalarının blokzincir sistemlerindeki uygulama alanlarına değinilecek ve kullanım amaçları açıklanacak-

tır. Özellikle, bu şemaların blokzincir sistemlerinde çok taraflı hesaplama ve sıfır bilgi ispatı gibi kriptografik protokollerle birlikte kullanımlarına vurgu yapılacaktır. Kitap bölümünün içeriği çok fazla teorik tanımlara girilmeden basit bir dilde verilecektir. İçeriğin oluşturulmasında [4] [11] [12] [13] [14] numaralı kaynaklardan sıkça faydalanılmıştır.

Bu kitap bölümünün içeriği aşağıdaki gibi organize edilmiştir. Bölüm 10.2'de blokzincir sistemlerinde kullanılan kriptografik bileşenlerden bahsedilmektedir. Bölüm 10.3 sır paylaşım şemalarının genel yapısını verdikten sonra bazı sır paylaşım şemalarını tanımlamaktadır. Bölüm 10.4 Lagrange interpolasyon metodunu ve Shamir sır paylaşım şemasını vermektedir. Bölüm 10.5'te lineer kodlarla ilgili gerekli temel bilgiler verildikten sonra bu kodların sır paylaşım şemaları ile bağlantıları ele alınmaktadır. Bölüm 10.6'da lineer kodlar vasıtasıyla sır paylaşım şemalarının tasarım yöntemleri ayrıntılı bir şekilde verilmekte ve bazı özel kodlardan sır paylaşım şemaları elde edilmektedir. Bölüm 10.7 sır paylaşım şemalarının blokzincir sistemlerindeki kullanım amaçlarını açıklamaktadır. Son olarak, Bölüm 10.8'de kitap bölümüyle ilgili değerlendirmeler ve sonuçlar verilmektedir.

10.2. BLOKZİNCİR İÇİN KRİPTOGRAFİ

Blokzincir teknolojisinin birçok alana adapte edilebilecek çok güçlü bir teknoloji olduğu düşünülmektedir. Fakat bu teknolojinin baş etmesi gereken birçok problem olduğu da bilinmektedir. Geleneksel blokzincir sistemlerinde çözülmesi gereken başlıca problemler sistem güvenliğinin iyileştirilmesi, gizlilik, ölçeklenebilirlik, anahtar yönetimi ve akıllı sözleşmelerin yönetimidir. Bu problemlerin temel kaynağı, blokzincirdeki ağ yapısı ve uzlaşma mekanizmasıdır. Bu problemlerin çözümü için özet fonksiyonları, imzalama, şifreleme, sıfır bilgi ispatı (ZKP), güvenli çok taraflı hesaplama (sMPC), habersiz aktarım (OT) ve sır paylaşım şeması (SSS) gibi kriptografik yöntemler kullanılmaktadır. Bu bölümde, blokzincir sistemlerinde kullanılan bu protokollere ve sır paylaşım şemalarının bu protokollerle birlikte kullanımlarına değinilecektir.

Kriptografide şifreleme algoritmaları, simetrik ve asimetrik (açık anahtarlı) olmak üzere iki gruba ayrılır. Simetrik kriptografide şifreleme algoritmaları

bir tek gizli anahtara sahiptir ve hem şifreleme hem de şifre çözme işlemlerinde bu gizli anahtar kullanılır. Dolayısıyla, iletişimde simetrik şifreleme kullanıldığı zaman alıcı ve gönderici tarafların her ikisinin de bu gizli anahtara sahip olması gerekmektedir. Günümüzde en yaygın kullanılan simetrik şifreleme algoritması AES blok şifre algoritmasıdır. Geleneksel blokzincir sistemlerinde blok/işlem verileri genellikle şifrelenmemektedir; fakat bazı özel blokzincir sistemlerinde bu veriler simetrik algoritmalar ile şifrelenmektedir [7] [8] [15]. Bu sistemlerde şifreleme işlemi ile veri güvenliği sağlanmaktadır, örneğin Hyperledger fabric [15] üzerinde akıllı sözleşmelerin mahremiyetini sağlamak için simetrik şifreleme kullanılmaktadır. Ayrıca, simetrik şifreleme algoritmaları kripto para cüzdanlarını (cryptocurrency wallets) şifrelemek için de yaygın olarak kullanılmaktadır. Örneğin, Ethereum blokzincir sisteminde kullanıcıların kripto cüzdanları AES algoritması ile şifrelenmektedir [16]. Simetrik şifreleme algoritmalarının kullanıldığı blokzincir sistemlerinde ve kripto para cüzdanlarında şifreleme anahtarları eşik sır paylaşım şemaları ile blokzincir ağındaki (veya ilgili gruptaki) kullanıcılar arasında güvenli bir şekilde paylaştırılmaktadır. Böylece, şifreleme anahtarları bir bütün olarak hiçbir kullanıcıda ve/veya bilgisayar hafızasında tutulmaz.

Asimetrik kriptosistemler biri gizli diğeri açık olmak üzere birbiriyle ilintili iki farklı anahtara sahiptir ve gizli anahtardan açık anahtar kolaylıkla elde edilebilirken tersi mümkün değildir. Bu tek yönlü geçişin sağlanmasının nedeni açık anahtarlı algoritmaların tek yönlü matematiksel zor problemlere dayanmasıdır. Örneğin, RSA algoritmasının güvenliği çarpanlara ayırma probleminin zorluğu ile ilişkiliyken Diffie-Hellman anahtar değişim protokolünün güvenliği sonlu cisimler üzerindeki ayrık logaritma probleminin zorluğuna dayanmaktadır. Asimetrik kriptosistem ile bir iletişimi şifrelemek için gönderici şifreleme işlemini alıcının açık anahtarı ile yapar ve alıcı da kendi gizli anahtarı ile bu şifreli iletişimi çözer. Asimetrik kriptosistemler hem şifreleme hem de imzalama işlemleri için yaygın olarak kullanılmaktadır. Örneğin, sır paylaşım şemaları içerisinde kullanıcı paylarını şifrelemek için RSA şifreleme algoritması kullanılmaktadır [7] [17]. Blokzincir sistemlerinde ise elektronik imzalama işlemi için kullanılmaktadır.

Elektronik imzalama blokzincir sistemlerinde veri bütünlüğünü sağlamak ve kullanıcı kimlik denetimini yapmak için kullanılan bir yöntemdir. Blokzincir ağında uzlaşmanın sağlanmasında da önemli rol oynayan elektronik imzalama işlemi blokzincir platformu için en önemli bileşenlerden bir tanesidir. İmzalama işlemi yapmak için gönderici kendi gizli anahtarı (imzalama anahtarı) ile veriyi imzalar ve alıcı göndericinin açık anahtarı (doğrulama anahtarı) ile kendisine gönderilen imzanın doğruluğunu kontrol eder. Gönderilen verinin gönderici tarafından imzalanması ile göndericinin gönderdiği veriyi inkâr edememesi (non-repudiation) sağlanırken gönderilen imzanın alıcı tarafından kontrol edilmesiyle de göndericinin kimlik denetimi (authentication) yapılmaktadır. Blokzincir sistemlerinde kullanıcılar blok/işlem verilerini imzalamak için elektronik imzalama algoritmalarını kullanmaktadır. Blokzincir ağında bir kullanıcının işlem verisini imzalaması sayesinde, bu kullanıcının hem yaptığı işlemi inkâr edememesi sağlanır hem de kimlik denetimi yapılır. Ayrıca işlem verisinin bozulması ve değiştirilmesi de önlenir. Bazı blokzincir sistemlerinde, kullanıcı mahremiyetini ve anonimliğini iyileştirmek için çoklu imzalama, kör (blind) imzalama, halka (ring) imzalama ve eşik imzalama gibi farklı imzalama şemaları kullanılmaktadır. Örneğin, eşik imzalama şemasında imzalama anahtarı eşik sır paylaşım şeması kullanılarak kullanıcılar arasında paylaştırılmakta ve imzalama işlemi paylaşım şemasının yetkili kullanıcıları tarafından ortak olarak gerçekleştirilmektedir [4] [5].

Bazı blokzincir sistemlerinde kullanıcı anonimliğini ve veri gizliliğini iyileştirmek için sıfır bilgi ispatı (ZKP), güvenli çok taraflı hesaplama (sMPC) ve habersiz aktarım (OT) protokolleri de kullanılmaktadır. Blokzincir sistemlerinde hali hazırda kullanılmakta olan ve gelecekte kullanılma potansiyeli olan tüm kriptografik yöntemler [4] numaralı kaynakta ayrıntılı olarak ele alınmıştır. Ayrıca, olası kuantum sonrası blokzincir sistemleri için bu kriptografik yöntemlerin büyük ölçekli kuantum bilgisayarlar karşısındaki güvenlik durumları [18] numaralı kaynakta ele alınmıştır. Kuantum sonrası kriptografi hakkında detaylı bilgi için önceki kitap serilerinden [19] numaralı kaynak kitap kullanılabilir.

10.3. SIR PAYLAŞIM ŞEMALARI

Bu bölümde sır paylaşım şemalarının genel yapısı verilecek ve literatürde yer alan bazı önemli sır paylaşım şemaları tanımlanacaktır.

Sır paylaşım şeması kavramı ilk olarak 1979 yılında Shamir [20] ve Blakley [21] tarafından birbirlerinden bağımsız olarak ortaya atılmıştır. Bu şema bir dağıtıcı D (genellikle, bir bilgisayar ya da sunucu) ve n katılımcıdan $P = \{P_1, P_2, \dots, P_n\}$ oluşur. Sistem içerisinde S sır uzayı ve $S_1, S_2, \dots, S_n$ katılımcı paylarının uzayları (genellikle, sonlu bir cisim) mevcuttur. Başlangıç aşamasında dağıtıcı D tarafından sistem için gerekli olan parametreler oluşturulur ve kullanıcıların kimlik denetimleri yapılır. Şemada temel olarak iki protokol çalışmaktadır:

1. **Sır değerini paylaştırma protokolü.** Dağıtıcı D, S uzayından gizli (sır) s değerini seçer, her katılımcı P_i için S_i kümesine ait s_i (s sır değerinin bir parçası) payını oluşturur ve P_i katılımcısına gönderir, $1 \leq i \leq n$.
2. **Sır değerini kurtarma protokolü.** $1 \leq t \leq n$ için P kümesinden t tane $P_{i_1}, P_{i_2}, \dots, P_{i_t}$ yetkili katılımcı $s_{i_1}, s_{i_2}, \dots, s_{i_t}$ paylarını dağıtıcı D'ye gönderir ve dağıtıcı s sır değerini oluşturur.

Sır paylaşım şemalarının sağlaması gereken temel güvenlik kriterleri doğruluk ve gizliliktir.

- **Doğruluk:** Eğer sistemdeki bütün katılımcılar ve dağıtıcı dürüst bir şekilde işlemlerini tamamlar ise, yetkili katılımcılar kurtarma protokolü ile sır değerini elde eder.
- **Gizlilik:** Sistem içerisindeki yeterli sayıda yetkili katılımcı dışında hiçbir grup ya da katılımcı sır değeri hakkında hiçbir bilgi edinemez.

Sır değerini kurtarma aşamasında, paylarını birleştirerek s sır değerini elde edebilen yetkili katılımcıların kümesi $\{P_{i_1}, P_{i_2}, \dots, P_{i_t}\}$ *erişim kümesi (access set)* olarak adlandırılır. Tüm erişim kümelerinden oluşan küme ailesine sır paylaşım şemasının *erişim yapısı (access structure)* denir. Bir erişim kümesinin kendisinden başka hiçbir alt kümesi erişim kümesi değilse, bu erişim

kümesine *minimal erişim kümesi* denir. Bütün minimal erişim kümelerinin kümesi de *minimal erişim yapısı* olarak adlandırılır.

Bir sır paylaşım şemasında herhangi bir erişim kümesinin herhangi bir üst kümesi de bu şemanın erişim kümesi ise, bu şema *monoton erişim yapısına* sahiptir denir. Monoton erişim yapısına sahip sır paylaşım şemasında, minimal erişim kümeleri erişim yapısını tamamen karakterize eder. Başka bir ifadeyle, bu sır paylaşım şemalarında sadece minimal erişim kümelerini belirlemek yeterlidir. Bir sır paylaşım şemasında eğer her $t \geq 1$ tane katılımcı grubu aynı sayıda minimal erişim kümesinde yer alıyorsa, bu şema *t-dereceli demokratik (t-demokratik)* sır paylaşım şeması olarak adlandırılır. Özel olarak, her bir katılımcı aynı sayıda minimal erişim kümesinde yer alıyorsa, bu şema *demokratik sır paylaşım şeması* olarak adlandırılır. Eğer bir katılımcı tüm minimal erişim kümelerinde yer alıyorsa, bu katılımcı *diktatör* olarak adlandırılır. Bir sır paylaşım şemasında, tüm katılımcıların kümesinin her bir alt kümesi paylarını birleştirerek s sır değerini elde edebiliyorsa veya s sır değeri hakkında hiçbir bilgi edinemiyorsa bu sır paylaşım şeması *mükemmel* olarak adlandırılır. Mükemmel bir sır paylaşım şemasında, katılımcıların paylarının büyüklüğü sır değerinin büyüklüğüne eşitse (sır uzayı ile katılımcı uzayları aynı ise), bu şema *ideal* olarak adlandırılır. n katılımcıdan oluşan sır paylaşım şemasında, $1 \leq t \leq n$ olmak üzere eğer

- en az t tane katılımcı paylarını birleştirerek s sır değerini elde edebiliyorsa, ve
- t'den daha az sayıdaki hiçbir katılımcı grubu s sır değeri hakkında hiçbir bilgi edinemiyorsa,

bu şema *(t, n)-eşik şeması* olarak adlandırılır. Bu eşik şeması bazen *(n, t)-eşik şeması* olarak da adlandırılır. Shamir [20] ve Blakley [21] tarafından önerilen klasik şemalar ideal (t, n)-eşik sır paylaşım şemalarıdır. Herhangi bir (t, n)-eşik sır paylaşım şeması mükemmel ve t-demokratiktir. Sır paylaşım şemaları hakkında detaylı bilgi için [22] numaralı kaynak kitaptan faydalanılabilir.

Klasik sır paylaşım şemaları pasif saldırılara karşı güvenli olmasına rağmen aktif saldırılara karşı güvenli değildir. Daha açık bir ifadeyle, bu şemalarda dürüst olmayan bir dağıtıcı ve/veya katılımcı olması durumunda sır güvenli

bir şekilde paylaştırılamaz. Bu nedenle, dağıtıcı ve katılımcıların güvenilir oldukları varsayılmaktadır, diğer bir deyişle sistem içerisinde hiç kimsenin hile yapmadığı kabul edilmektedir. Ancak bu varsayımın gerçekleştirilmesi günlük hayat uygulamalarında oldukça zordur. Bu varsayımı ortadan kaldırmak için Shamir şemasına doğrulama algoritması eklenerek ilk doğrulanabilir sır paylaşım (verifiable secret sharing - VSS) şeması Chor ve ark. tarafından 1985 yılında önerilmiştir [23]. Doğrulama algoritmaları sistem içerisindeki dağıtıcı ve aktif katılımcıların dürüstlüğünün anlık olarak kontrol edilmesine olanak sağlamaktadır. Bir VSS şemasında, dürüst olmayan dağıtıcı ve/veya katılımcı doğrulama algoritması kullanılarak diğer kullanıcılar ve/veya dağıtıcı tarafından kolayca tespit edilebilmektedir. Dürüst olmayan katılımcıların herkes tarafından da denetlenebilmesi için literatürde açıkça doğrulanabilir sır paylaşım şeması da önerilmiştir [24]. Bir VSS şemasının aşağıda verilen iki tür aktif saldırıya karşı güvenli olduğu söylenebilir:

- Dürüst olmayan bir dağıtıcı, payların dağıtılması aşamasında katılımcıların pay değerlerini dağıtmadan önce değiştirebilir ve yanlış bir pay değeri gönderebilir.
- Herhangi bir kötü niyetli katılımcı, sır değerini kurtarma aşamasında gerçek pay değerini göndermek yerine sahte pay değeri gönderebilir.

Birden çok sır değerini aynı anda paylaşabilme problemi ilk olarak 1993 yılında Jackson ve ark. [25] tarafından ele alınmış ve çoklu sır eşik şeması önerilmiştir. Daha sonra, 1995 yılında He ve Dawson [26] tarafından Shamir'in şemasına dayanan birden çok sır değerlerini aynı anda paylaşabilen çoklu sır paylaşım (multi-secret sharing - MSS) şeması önerilmiştir. Bir MSS şemasında, her bir katılımcıya yalnızca bir pay verilerek (diğer bir ifadeyle, her katılımcı yalnızca bir pay saklar) birden çok sır değeri paylaştırılır. Burada, sır uzayları ve katılımcı paylarının uzayları aynı olduğu zaman paylaşılan payların boyutunun bir sır değerinin boyutuyla aynı olduğu varsayılır. 1995 yılında Harn [27] eşzamanlı olarak birden çok sır paylaşmanın yanı sıra dürüst olmayan dağıtıcı ve katılımcıları tespit eden ilk eşik tabanlı doğrulanabilir çoklu sır paylaşım (verifiable multi secret sharing - VMSS) şemasını önermiştir. Eşik tabanlı VMSS şemaları, pratikte birçok kullanım alanına sahiptir ve bu şemalarla ilgili literatürde çeşitli teknikler geliştirilmiştir. Bu şemalar doğrulama algoritmalarına sahip oldukları için hem pasif hem de aktif saldırılara karşı güvenlidir.

Doğrulanabilir çoklu sır paylaşım şemalarında, dağıtıcı tarafından gerekli parametreler oluşturulduktan ve kullanıcıların kimlik denetimleri yapıldıktan sonra temel olarak üç protokol çalışmaktadır.

1. **Paylaştırma Protokolü:** Dağıtıcı saklanacak olan sır değerlerini seçer, karşılık gelen katılımcı paylarını ve kontrol parametrelerini oluşturur. Daha sonra, her bir katılımcı payını ve kontrol parametresini ilgili katılımcıya gönderir.
2. **Kontrol Protokolü:** Her bir katılımcı, kontrol parametrelerini kullanarak kendisine gönderilen payların doğruluğunu doğrulama algoritması ile kontrol eder. Eğer bir yanlışlık varsa dağıtıcıya geri gönderir ve sisteme uyarı verir.
3. **Kurtarma Protokolü:** Yetkili katılımcılar paylarını dağıtıcıya gönderir ve dağıtıcı kurtarma protokolü ile sır değerlerini oluşturur. Burada, her bir katılımcının gönderdiği payların doğruluğunu dağıtıcı (ve isteyen diğer katılımcılar) doğrulama algoritmasını kullanarak kontrol eder.

Bu şemalar doğrulama algoritmalarına sahip oldukları için doğruluk ve gizlilik kriterlerinin yanı sıra doğrulanabilirlik kriterini de sağlaması gerekmektedir. **Doğrulanabilirlik:** paylaştırma aşamasında yanlış pay değeri gönderen dağıtıcı kontrol aşamasında denetlenebilir ve benzer şekilde, kurtarma aşamasında sahte pay değeri gönderen katılımcılar da aynı aşamada denetlenebilir.

10.3.1. Literatür Taraması

Sır paylaşım şeması yaklaşık olarak 40 yıldır literatürde çalışılmaktadır ve günlük hayattaki kullanıma yönelik çeşitli şemalar önerilmiştir. İlk olarak, 1979 yılında Shamir [20] cebirsel sır paylaşım şemasını ve Blakley [21] geometrik bir fikir üzerinde sır paylaşım şemasını tanımlamıştır. Shamir'in sır paylaşım şeması sonlu cisimler üzerinde polinom interpolasyonuna dayanırken Blakley'in şeması sonlu geometriye dayanmaktadır. Daha açık bir ifadeyle, sır paylaşım probleminin çözümü için Shamir polinom interpolasyonunu kullanırken Blakley hiper düzlem geometrisini (hyperplane geometry) kullanmıştır. Bu ilk çalışmalar sır paylaşım şeması tasarımında litera-

türde öncü çalışmalar olmuş ve bu alandaki çalışmalara yön vermiştir. 1983 yılında Asmuth ve Bloom [28] Çin Kalan Teoremine dayanan aritmetik bir yaklaşım kullanarak bu şemalardan farklı yeni bir sır paylaşım şeması önermişlerdir. Fakat bu sır paylaşım şemaları oldukça basit erişim yapılarına sahiptir. Örneğin, bu şemalar bazı katılımcıların (şirket CEO'su, kurum yöneticisi, ülke lideri vb.) diktatör katılımcı olmalarına olanak sağlayan erişim yapısına sahip değildir. Bu şemaların erişim yapılarının günlük kullanımda bu tarz ihtiyaçları karşılamamasından dolayı daha iyi erişim yapılarına sahip yeni sır paylaşım yöntemleri araştırılmıştır. Bu amaçla literatürde birçok tasarım yöntemi önerilmiştir ve bu yöntemlerden en önemlisi kodlama teorisine dayanmaktadır.

Kodlama teorisinin en önemli kod sınıfını oluşturan lineer kodlardan elde edilen sır paylaşım şemaları çok iyi erişim yapılarına sahiptir. Literatürde MDS kodlardan birçok ideal sır paylaşım şeması elde edilmiştir [11] [29] [30] [31] [32] [33] [34]. Özellikle, McEliece ve Sarwat [9] tarafından önerilen tasarım yöntemi kullanılarak MDS kodlardan ideal sır paylaşım şeması tasarımı yaygın olarak çalışılmıştır [31] [32] [33] [34]. Diğer taraftan, lineer kodların çok özel bir alt sınıfını oluşturan minimal kodların dual kodları da çok iyi erişim yapılarına sahip sır paylaşım şemaları vermektedir. Literatürde, minimal kodların dual kodlarına dayanan birçok demokratik sır paylaşım şeması ve diktatör katılımcılara sahip sır paylaşım şeması elde edilmiştir [11] [12] [13] [34] [35].

Gerçek hayat uygulamalarında sıkça ihtiyaç duyulan doğrulanabilir sır paylaşım şemaları literatürde etraflıca çalışılmış ve bu alanda birçok yöntem önerilmiştir [7] [17] [23] [36] [37] [38] [39]. Bu şemalarda dağıtıcı ve katılımcıların dürüstlüğünü kontrol eden doğrulama algoritmaları matematiksel zor problemlere dayanmaktadır, diğer bir ifadeyle doğrulama işleminde kullanılan taahhüt fonksiyonu tek yönlü (one-way) bir fonksiyondur. Genellikle, bu fonksiyon günümüz bilgisayarları karşısında güvenli olan ayrık logaritma problemine dayanan modüler üst alma fonksiyonudur. Ayrıca, bu problem kuantum güvenli olmadığı için kuantum güvenli olduğu düşünülen kafes-tabanlı zor problemlerin kullanımı da literatürde önerilmiştir [7] [40] [41] [42].

Bir sır paylaşım şemasında, dağıtıcı ve katılımcılar arasında güvenli iletişimi sağlamak için güvenli kanallar olduğu varsayılmaktadır. Daha açık bir ifadeyle, payların dağıtıcı tarafından güvenli kanallar aracılığıyla dağıtıldığı ve toplandığı varsayılmaktadır. Fakat günlük hayattaki kullanımda güvenli kanal oluşturmak oldukça zor ve maliyetli bir işlemdir. Bu durumda, sistem içerisinde güvenli kanal olmadan güvenli iletişimi sağlamak için dağıtıcı ve katılımcılar arasındaki iletişim RSA gibi açık anahtarlı algoritmalarla şifrelenmektedir [7] [17] [39]. Payların dağıtılma aşamasında, dağıtıcı her bir katılımcının payını açıktan göndermek yerine katılımcının açık anahtarı ile şifreleyerek göndermektedir. Benzer şekilde, sır değerini oluşturma aşamasında aktif katılımcılar paylarını dağıtıcının açık anahtarı ile şifreleyerek dağıtıcıya göndermektedir. Diğer taraftan, iletişimin kuantum güvenli olabilmesi için literatürde RSA algoritmasının yanı sıra kuantum güvenli olduğu düşünülen kafes-tabanlı NTRU şifreleme algoritmasının kullanımı da önerilmiştir [7]. Böylece, sistem içerisinde güvenli kanallar olmadan güvensiz kanallar üzerinden güvenli iletişim gerçekleşmektedir.

Sır paylaşım şemaları blokzincir sistemlerinde ve veri tabanlarında OT, ZKP ve sMPC gibi kriptografik yöntemlerle birlikte gizlilik artırıcı protokol olarak da kullanılmaktadır. Örneğin, veri tabanlarında, verileri güvenli ve dağıtık olarak depolayabilmek için sMPC protokolü ve şifreleme algoritması ile birlikte kullanılmaktadır [1] [2] [3]. Bu şemalar bazı geleneksel blokzincir sistemlerinde şifreleme/imzalama anahtarlarını kullanıcılar arasında paylaştırmak için kullanılırken [4] [5] [6] dağıtık depolamalı blokzincir (distributed storage blockchain - DSB) sistemlerinde depolama ve iletişim maliyetlerini iyileştirmek için kullanılmaktadır [7] [8] [9] [10].

10.4. SHAMİR SIR PAYLAŞIM ŞEMASI

Bu bölümde, Lagrange interpolasyon metodu tanımlandıktan sonra Shamir'in eşik sır paylaşım şeması [20] verilecektir.

Verilen p asal sayısı için $Z_p = \{0,1, \dots, p-1\}$ kümesi modüler toplama " $+$ " ve modüler çarpma " $\cdot$ " işlemleriyle birlikte cebirsel bir yapı oluşturur, bu yapı $(Z_p, +)$ ile gösterilir ve *sonlu asal cisim* olarak adlandırılır.

Sonlu cisimlerle ilgili detaylı bilgi için [43] numaralı kaynak kitaptan faydalanılabilir. Z_p sonlu asal cismi üzerindeki tüm işlemler modüler aritmetik kullanılarak $\bmod p$'ye göre yapılmaktadır. Z_p sonlu cismi mühendislik alanlarında genellikle Galois cismi olarak adlandırılır ve $GF(p)$ ile gösterilir, bu kitap bölümünde ise F_p gösterimi kullanılacaktır. Ayrıca, lineer kodlar ve sır paylaşım şemaları F_p üzerinde tanımlanmaktadır.

Polinomlar üzerindeki Lagrange interpolasyon metodu aşağıdaki gibi tanımlanır. Eğer $h(x) \in F_p[x]$ derecesi en fazla l olan bir polinom ise,

$$\delta_i(x) = \prod_{j \in S, j \neq i} \frac{x-j}{i-j} \bmod p$$

olmak üzere bu polinom

$$h(x) = \sum_{i \in S} h(i)\delta_i(x) \qquad (10.1)$$

şeklinde yazılabilir öyle ki $S \subset F_p$ ve $|S| = l + 1$. Bu ifadenin doğruluğu şöyle açıklanabilir. Her bir $\delta_i(x)$ polinomu l tane tek terimli polinomun çarpımı olduğu için bu polinom derecesi en fazla l olan bir polinomdur. Dolayısıyla, (10.1) denkleminin sağ tarafı derecesi en fazla l olan bir polinomdur ve

$$h(x) - \sum_{i \in S} h(i)\delta_i(x) = 0 \qquad (10.2)$$

denklemi elde edilir. Bir cisim üzerinde derecesinden daha fazla köke sahip olabilen polinom sadece sıfır polinomu olduğu için, denklem (10.2)'deki polinom, sıfır polinomudur ve böylece denklem (10.1) sağlanır. Benzer şekilde, Lagrange interpolasyonunun neden çalıştığı aşağıdaki gibi açıklanabilir. Sır paylaşım şemasında katılımcı paylarının bir kümesi $\{s_i \in F_p \mid i \in S\}$ verilsin öyle ki $s_i = h(i) \bmod p$ ve $|S| = l + 1$ olsun. Dolayısıyla, her $i \in S$ için (i, s_i) noktalarını kullanarak derecesi en fazla l olan $h(x)$ polinomu

$$h(x) = \sum_{i \in S} s_i \delta_i(x)$$

şeklinde elde edilebilir. $i = 1,2, \dots, n$ için $r_i = \delta_i(0)$ olmak üzere Lagrange interpolasyonunun sonucu olarak derecesi en fazla $(n-1)$ olan $h(x)$ polinomundan

$$h(0) = \sum_{i=1}^{n} r_i h(i)$$

hesaplanabilir. Burada r_i değerleri $h(x)$ polinomuna bağlı değildir çünkü $\delta_i(x)$ polinomu $h(x)$ polinomundan bağımsızdır. Sonuç olarak, her sır değeri $s \in F_p$ ve her alt küme $S \subset F_p$ (öyle ki $|S| = t$ ve $0 \notin S$) için, $f(0) = s$ olacak şekilde derecesi en fazla t olan rastgele bir $f(x)$ polinomu alındığı zaman, t tane $(f(i))_{i \in S}$ paylarının dağılımı F_p cismi üzerinde s sır değerinden bağımsızdır. Dolayısıyla, hiçbir katılımcı ya da bir başkası s sır değeri hakkında bilgi edinemez.

Shamir sır paylaşım şeması yukarıda verilen polinom interpolasyonuna dayanmaktadır. D dağıtıcı, $P = \{P_1, \dots, P_n\}$ katılımcı kümesi ve t eşik değer olmak üzere, $2 \leq t \leq n$, Shamir (t, n)-eşik sır paylaşım şeması F_p sonlu cismi, $p > 2n - t + 1$, üzerinde aşağıdaki gibi çalışmaktadır. Katılımcılar arasında paylaştırılacak olan gizli bilgi (sır) $s \in F_p$ olsun (s sır değeri $1 \leq s \leq p-1$ aralığında bir sayıdır). Dağıtıcı D, s sır değerini n katılımcı arasında paylaştırır ve bu katılımcılar arasından en az t katılımcının paylarını birleştirerek bu sır değerini tekrar oluşturur.

1. **Sır değerini paylaştırma protokolü.**

   - Dağıtıcı D, n tane katılımcı arasında paylaştırmak için sır uzayından $s \in F_p$ sır değerini alır.
   - Dağıtıcı D rastgele $(t-1)$ tane $a_k \in F_p$ elemanını seçer, $k = 1, \dots, t-1$, ve $(t-1)$-inci dereceden $f(x) = s + a_1 x + \cdots + a_{t-1} x^{t-1}$ polinomunu oluşturur öyle ki $f(0) = s$.
   - Dağıtıcı D her $i = 1, 2, \dots, n$ için $y_i = f(i) \bmod p$ değerini hesaplar ve P_i katılımcısına gönderir.

2. **Sır değerini kurtarma protokolü:** Dağıtıcı D, en az t katılımcı paylarını birleştirerek yukarıda verilen Lagrange interpolasyon yöntemi ile s sır değerini ortaya çıkarır.

Örnek 10.1'de $p = 13, n = 5$ ve $t = 3$ parametreleri için $s = 8$ sır değerini paylaştıran Shamir sır paylaşım şeması verilmektedir.

Örnek 10.1:

F_{13} cismi üzerinde tanımlanan Shamir'in (3,5)-eşik sır paylaşım şemasında dağıtıcı D, P_1, P_2, P_3, P_4, P_5 katılımcıları arasında $s = 8$ sır değerini paylaştırsın ve herhangi 3 katılımcının payını birleştirerek sır değerini tekrar elde etsin.

Sır değerini paylaştırma protokolü. Bu protokolde dağıtıcı D aşağıdaki adımları izler.

- F_{13} cisminden rastgele $a_1 = 2$ ve $a_2 = 7$ sayılarını seçer ve bu cisim üzerindeki ikinci dereceden

$$h(x) = s + a_1 x + a_2 x^2 = 8 + 2x + 7x^2$$

  polinomunu oluşturur.

- Her $i \in \{1, \dots, 5\}$ için $y_i = h(i) \bmod 13$ değerlerini hesaplar:
  - $y_1 = h(1) = 8 + 2 + 7 \bmod 13 = 4,$
  - $y_2 = h(2) = 8 + 2 \cdot 2 + 7 \cdot 4 \bmod 13 = 1,$
  - $y_3 = h(3) = 8 + 2 \cdot 3 + 7 \cdot 9 \bmod 13 = 12,$
  - $y_4 = h(4) = 8 + 2 \cdot 4 + 7 \cdot 16 \bmod 13 = 11,$
  - $y_5 = h(5) = 8 + 2 \cdot 5 + 7 \cdot 25 \bmod 13 = 11.$

- P_1, P_2, P_3, P_4, P_5 katılımcılarına sırasıyla $(1,4), (2,1), (3,12), (4,11), (5,11)$ ikililerini gönderir.

Sır değerini kurtarma protokolü. Eşik değer $t = 3$ olduğu için katılımcılardan herhangi 3 tanesi bir araya gelerek s sır değerini tekrar elde edebilir. Varsayalım ki erişim kümesi $S = \{P_2, P_3, P_4\}$ olsun. Lagrange interpolasyon yöntemi ile $h(x)$ polinomunu oluşturmak için aşağıdaki adımlar gerçekleştirilir.

$$\delta_2(x) = \prod_{j=3,4} \frac{x-j}{2-j} = \frac{(x-3)(x-4)}{(2-3)(2-4)} = 2^{-1}(x-3)(x-4)$$
$$\equiv 7(x-3)(x-4) \bmod 13,$$

$$\delta_3(x) = \prod_{j=2,4} \frac{x-j}{3-j} = \frac{(x-2)(x-4)}{(3-2)(3-4)} = -(x-2)(x-4) \qquad (10.3)$$
$$\equiv 12(x-2)(x-4)\ mod\ 13,$$

$$\delta_4(x) = \prod_{j=2,3} \frac{x-j}{4-j} = \frac{(x-2)(x-3)}{(4-2)(4-3)} = 2^{-1}(x-2)(x-3)$$
$$\equiv 7(x-2)(x-3) \bmod 13.$$

Bu şemada, F_{13} sonlu cismi üzerinde çalışıldığı için $2^{-1} \equiv 7 \bmod 13$ ve $-1 \equiv 12 \bmod 13$ olur. Kolaylıkla kontrol edilebilir ki

$$\delta_2(2) = 1 \bmod 13,\ \ \delta_2(3) = 0 \bmod 13 \ \text{ ve } \ \delta_2(4) = 0 \bmod 13$$
$$\delta_3(2) = 0 \bmod 13,\ \ \delta_3(3) = 1 \bmod 13 \ \text{ ve } \ \delta_3(4) = 0 \bmod 13$$
$$\delta_4(2) = 0 \bmod 13,\ \ \delta_4(3) = 0 \bmod 13 \ \text{ ve } \ \delta_4(4) = 1 \bmod 13$$

denklemleri sağlanır ve böylece $h(2) = y_2, h(3) = y_3$ ve $h(4) = y_4$ elde edilir. Dolayısıyla, denklem (10.3) kullanılarak

$$h(x) = y_2\delta_2(x) + y_3\delta_3(x) + y_4\delta_4(x)$$
$$= y_2 \cdot 7(x-3)(x-4) + y_3 \cdot 12(x-2)(x-4) + y_4 \cdot 7(x-2)(x-3)$$
$$= (7y_2 + 12y_3 + 7y_4)x^2 + (3y_2 + 6y_3 + 4y_4)x + (6y_2 + 5y_3 + 3y_4)$$

polinomu elde edilir. Dağıtıcı D tarafından oluşturulan polinom $h(x) = s + a_1x + a_2x^2$ şeklinde olduğu için bu polinomun katsayıları

$$s \equiv 6y_2 + 5y_3 + 3y_4 \bmod 13,$$
$$a_1 \equiv 3y_2 + 6y_3 + 4y_4 \bmod 13,$$
$$a_2 \equiv 7y_2 + 12y_3 + 7y_4 \bmod 13$$

denklemlerinden hesaplanır. Bu üç denklemde $y_2 = 1, y_3 = 12$ ve $y_4 = 11$ pay değerleri yerlerine yazılarak $h(x) = 8 + 2x + 7x^2$ polinomu oluşturulur. Ayrıca, $h(x)$ polinomu hesaplanmadan y_2, y_3, y_4 değerleri kullanılarak sadece birinci denklemden de $s = 8$ sır değeri elde edilebilir. Böylece, (3,5)-eşik Shamir sır paylaşım şemasında $S = \{P_2, P_3, P_4\}$ erişim kümesi tarafından $s = 8$ sır değeri elde edilmiş olur.

10.5. LİNEER KODLAR

Verilen n ve k iki pozitif tam sayı için F_p cismi üzerindeki F_p^n vektör uzayının k boyutlu lineer alt uzayı *lineer kod* olarak adlandırılır ve bu kod genellikle C ile gösterilir. F_p cismi üzerinde tanımlı olan bu C kodu $[n, k, d]_p$ parametreleri ile ifade edilir, burada n parametresi kodun uzunluğunu, k boyutunu ve d minimum Hamming uzaklığını ifade etmektedir. C kodunun dual kodu $C^{\perp}$, F_p cismi üzerinde n uzunluklu ve $(n - k)$ boyutlu bir alt uzaydır ve

$$C^{\perp} = \{b \in F_p^n \mid b \cdot a = 0 \text{ her } a \in C \text{ için}\}$$

şeklinde tanımlanır, burada " $\cdot$ " işlemi F_p^n üzerinde iki vektörün standart iç çarpımını göstermektedir. Dolayısıyla, $C^{\perp}$ dual kodu $[n, n - k, d^{\perp}]_p$ parametreleri ile gösterilir ve burada $d^{\perp}$ dual kodun minimum Hamming uzaklığıdır.

Bir lineer kodun minimum Hamming uzaklığı o kodun en önemli parametrelerinden biridir ve kodun hata düzeltme kapasitesini belirler. Ayrıca, lineer kodun ve dual kodun minimum uzaklıkları bu koddan elde edilen sır paylaşım şemasının erişim yapısının belirlenmesinde de önemli rol oynar. Her lineer kod sır paylaşım şeması tasarımında kullanıldığı için bir lineer kodun kendisinden ve dual kodundan elde edilen bir çift sır paylaşım şeması vardır. Elde edilen bu şemalar birbirleriyle ilişkili olsalar da farklı şemalardır.

F_p sonlu cismi üzerinde tanımlı $[n, k, d]_p$ parametreli C kodunun $\mathbf{G}$ *üreteç matrisi (generator matrix),* satırları C alt uzayının bazlarından oluşan $k \times n$ boyutlu

$$\mathbf{G} = [\boldsymbol{g}_1^{\mathrm{T}}, \boldsymbol{g}_2^{\mathrm{T}}, \ldots, \boldsymbol{g}_n^{\mathrm{T}}] = \begin{bmatrix} g_{11} & \cdots & g_{1n} \\ \vdots & \ddots & \vdots \\ g_{k1} & \cdots & g_{kn} \end{bmatrix}$$

şeklinde gösterilen bir matristir, burada $\boldsymbol{g}_i^{\mathrm{T}}$ sütun vektörü $\boldsymbol{g}_i = (g_{1i}, g_{2i}, \dots, g_{ki})$ vektörünün transpozudur, $1 \leq i \leq n$. Satırları $C^{\perp}$ dual kodunun bazlarından oluşan $(n-k) \times n$ boyutlu $\mathbf{H} = [\boldsymbol{h}_1, \dots, \boldsymbol{h}_n]$ matrisi ise $C^{\perp}$ dual kodun üreteç matrisi ve C kodunun *eşlik denetim matrisi (parity-check matrix)* olarak adlandırılır. Ayrıca, herhangi bir $\mathbf{G} \in F_p^{k \times n}$ matrisi için $[n, k, d]_p$ parametreli C kodu

$$C = \{\mathbf{mG} \mid \mathbf{m} \in F_p^k\}$$

şeklinde tanımlanabiliyorsa, bu matris C kodunun üreteç matrisidir. Benzer şekilde, eğer $C = \left\{\mathbf{x} \in F_p^n \;\middle|\; \mathbf{Hx^T} = 0\right\}$ veya $C^{\perp} = \{\mathbf{yH} \mid \mathbf{y} \in F_p^{(n-k)}\}$ şeklinde tanımlanabiliyorsa, $\mathbf{H} \in F_p^{(n-k) \times n}$ matrisi $C^{\perp}$ dual kodunun üreteç matrisidir. Burada, $\mathbf{Hx^T}$ ifadesi $\mathbf{x}$ vektörünün *sendromu (syndrome)* olarak adlandırılır. Bu üreteç matrislerinin tanımlarından, $\mathbf{GH^T} = \mathbf{0}_{k \times (n-k)}$ olduğu kolayca görülebilir. Sır paylaşım şeması tasarımında kullanılacak lineer kodların üreteç matrislerinin tüm sütun vektörleri sıfırdan farklı olmalıdır.

Bu bölümde lineer kodlar standart Hamming metriği ile ifade edilmektedir. Bir $\boldsymbol{a} = (a_0, \dots, a_{n-1}) \in F_p^n$ vektörünün *Hamming ağırlığı (weight)* bu vektörün sıfırdan farklı koordinatlarının sayısıdır, diğer bir ifadeyle bu vektörün desteği olarak tanımlanan

$$\mathrm{supp}(\boldsymbol{a}) = \{0 \leq i \leq n-1 : a_i \neq 0\}$$

kümesinin eleman sayısıdır ve $wt(\boldsymbol{a}) = \#\,\mathrm{supp}(\boldsymbol{a})$ şeklinde gösterilir. C kodunun elemanları olan vektörler, bu kodun kod sözcüğü (codeword) olarak adlandırılır ve sıfırdan farklı kod sözcüklerinin en küçük Hamming ağırlığı bu kodun minimum Hamming uzaklığı olarak tanımlanır. Verilen $\boldsymbol{a}, \boldsymbol{b} \in F_p^n$ için eğer $\boldsymbol{a}$ kod sözcüğünün desteği $\boldsymbol{b}$ kod sözcüğünün desteğini içeriyorsa, daha açık bir ifadeyle $\mathrm{supp}(\boldsymbol{b}) \subseteq \mathrm{supp}(\boldsymbol{a})$ ise, $\boldsymbol{a}$ kod sözcüğü $\boldsymbol{b}$ kod sözcüğünü *kapsar* denir.

C kodu içerisinde sıfırdan farklı bir $\boldsymbol{a}$ kod sözcüğü sadece kendisinin sabit (scalar) katlarını kapsıyor ama diğer sıfırdan farklı kod sözcüklerini kapsamıyor ise, $\boldsymbol{a}$ kod sözcüğüne *minimal* kod sözcüğü denir. C kodunun sıfırdan farklı tüm kod sözcükleri minimal ise, bu kod *minimal kod* olarak adlandırılır. Minimal kodlar lineer kodların çok önemli bir alt sınıfıdır ve cebirsel

özelliklerinden dolayı pratik kullanımda birçok uygulama alanına sahiptir. Bir lineer kodun minimal kod olup olmadığını belirlemek genel olarak oldukça zor bir problemdir; ancak bazı özel yapıdaki kodlar için nispeten kolay olduğu söylenebilir. Bu özel yapıdaki lineer kodların minimal olup olmadığını belirleyebilmek için literatürde birçok yöntem önerilmiştir. Bunlardan en önemlisi 1998 yılında A. Ashikhmin ve A. Barg [35] tarafından tanımlanan ve Lemma 10.2'de verilen yeter koşuludur.

Lemma 10.2: F_p cismi üzerinde tanımlı olan C lineer kodunun en küçük ve en büyük Hamming ağırlıkları sırasıyla wt_{min} ve wt_{max} ile gösterilsin. Eğer

$$\frac{p-1}{p} < \frac{wt_{min}}{wt_{max}} \tag{10.4}$$

ise, C lineer kodu minimaldir.

Diğer taraftan, herhangi bir minimal kodun denklem (10.4)'deki yeter koşulu sağlaması gerekmez. Bu yeter koşulu sağlamayan minimal kod üretimi literatürde yaygın olarak çalışılmaktadır. Örneğin, 2018 yılında Heng ve ark. [44] bir lineer kodun minimal olması için yeni bir gerek ve yeter koşul tanımladılar (bkz. Teorem 11, [44]) ve bu yeter koşulu sağlamayan ilk minimal kod sınıfını elde ettiler.

F_p sonlu cismi üzerinde tanımlı olan C $[n,k,d]_p$ lineer kodunun parametreleri arasında

$$d \leq n-k+1$$

bağıntısı vardır ve bu bağıntı *Singleton Bound (sınırı)* olarak adlandırılır. Bu bağıntı C kodunun minimum Hamming uzaklığı için bir üst sınır vermektedir. Eğer C lineer kodu Singleton sınırını sağlar ise, yani $d = n-k+1$ ise, C kodu *maksimum uzaklıklı ayrılabilir* (*Maximum Distance Separable* - MDS) kod olarak adlandırılır ve $[n,k,n-k+1\,]_p$ şeklinde gösterilir. Bu durumda, C MDS kodunun dual kodu $C^{\perp}$ $[n,n-k,k+1\,]_p$ parametreli başka bir MDS kodudur. Ayrıca, C MDS kodunun üreteç matrisinin herhangi k tane sütun vektörleri lineer bağımsızdır. Lineer kodlar hakkında detaylı bilgi için [45] numaralı kaynak kitaptan faydalanılabilir.

Lineer kodlar ile sır paylaşım şemaları arasındaki bağlantı ilk olarak McEliece ve Sarwat [46] tarafından 1981 yılında ortaya çıkarılmıştır. McEliece ve Sarwat, bu çalışmada polinom interpolasyonu ile ifade edilen Shamir'in sır paylaşım şemasının Reed-Solomon kodlarla ifade edilebileceğini göstermiştir. Böylece, kodlama teorisinin sır paylaşım şeması tasarımında kullanılabileceği fikrinin önü açılmıştır. Kodlama teorisindeki hata-düzelten lineer kodların sır paylaşım şeması tasarımında kullanılabilmesini sağlayan bu fikir şöyle açıklanabilir. Bir s sır değerinin $\mathbf{c} = (c_1, c_2, \dots, c_n) \in F_p^n$ kod sözcüğü ile kodlandığını varsayalım. Eğer bu kod sözcüğünden yeterli sayıda $c_i \in F_p$ değerleri bilinirse, lineer kodun hata düzeltme mekanizması kullanılarak diğer $c_j \in F_p$ değerleri bulabilir ve böylece kodlanan sır değeri elde edebilir. Daha sonra, 1989 yılında Brickell [29] ve 1993 yılında Massey [47] [48] bu fikirden yola çıkarak çok iyi erişim yapısına sahip yeni kod tabanlı sır paylaşım şemaları tasarlamışlardır. Ayrıca, McEliece ve Sarwat tarafından önerilen tasarım yöntemi geliştirilerek MDS kodlardan ideal eşik sır paylaşım şemaları tasarlanmıştır [30]. Bu şemada, dürüst olmayan katılımcılar MDS kodların kod çözme algoritmaları sayesinde belirlenebilmekte ve böylece sistem içerinde hile yapmak önlenebilmektedir. Bu çalışmaların yanı sıra literatürde birçok araştırmacı lineer kodlardan yeni sır paylaşım şemaları elde etmiştir [12] [13] [31] [32] [33] [49] [50]. Aslında, lineer koddan sır paylaşım şeması tasarlama teknikleri, sır paylaşım şeması tasarlama problemini lineer kod tasarlama problemine indirgemektedir. Özellikle, iyi erişim yapılı sır paylaşım şemalarının tasarlanması problemi MDS kodların ve minimal kodların tasarlanması problemine indirgenmektedir. Böylece kodlama teorisinin en önemli problemlerinden olan MDS kod ve minimal kod üretme problemleri daha da önemli hale gelmiştir.

Bir lineer kodun minimal kod olup olmadığını belirlemek genel olarak zor bir problem olmasına rağmen özel yapıdaki bazı lineer kodlar için nispeten daha kolay bir problem olduğu söylenebilir. Özellikle, kriptografik fonksiyonlardan elde edilen lineer kodların minimal kod oldukları kolayca gösterilebilmektedir [13] [49] [50]. Minimal lineer kodların cebirsel özellikleri, yüksek demokrasiye sahip sır paylaşım şemalarının elde edilmesine olanak sağlamaktadır [47]. Literatürde, minimal lineer kodların dual kodlarından birçok sır paylaşım şeması elde edilmiştir ve bu şe-

maların erişim yapıları belirlenmiştir [12] [13] [34] [35]. Ayrıca, kodlama teorisinin en iyi kod sınıfını oluşturan MDS kodların inşası ve bu kodlardan sır paylaşım şemalarının tasarımı da literatürde yaygın olarak çalışılmaktadır [11] [29] [30] [31].

10.6. LİNEER KODLARDAN SIR PAYLAŞIM ŞEMALARININ TASARIMI

Bu bölümde, lineer kodlardan sır paylaşım şemalarının tasarım yöntemleri verilecektir. Özellikle, MDS kodlardan ve minimal kodların dual kodlarından elde edilen sır paylaşım şemalarının erişim yapıları üzerinde durulacaktır.

Sır paylaşım şemalarının lineer kodlar üzerindeki tasarımı yaygın olarak çalışılmış olsa da literatürde öne çıkan iki tasarım yöntemi vardır. Bölüm 10.6.1'de verilen birinci yöntem McEliece ve Sarwat [46] tarafından önerilirken Bölüm 10.6.2'de verilen ikinci yöntem Massey [47, 48] tarafından önerilmiştir.

10.6.1. McEliece-Sarwat Tasarım Yöntemi (Birinci Yöntem)

Sır paylaşım şemaları ve lineer kodlar arasındaki bağlantı ilk olarak McEliece ve Sarwat tarafından 1981 yılında keşfedilmiştir [46]. Bu çalışmada, Shamir tarafından önerilen sır paylaşım şemasının Reed-Solomon kodlama şemasıyla ilişkisi ortaya çıkarılmıştır. Daha açık ifade etmek gerekirse, Shamir'in önermiş olduğu sır paylaşım tekniğinin Reed-Solomon kodunun [51] kod çözme algoritmasının özel hali olduğu gözlemlenmiştir. Bu gözlemden yola çıkılarak ileri sürülen kod tabanlı sır paylaşım şeması tasarım yöntemi aşağıda açıklanmaktadır.

$C \subset F_p^n$ alt uzayı F_p sonlu cismi üzerinde $[n, k, d]_p$ parametreli bir lineer kod olsun, burada $k \geq 2$. C kodunun $k \times n$ boyutlu üreteç matrisi $\mathbf{G} = [\boldsymbol{g}_1^{\mathrm{T}}, \boldsymbol{g}_2^{\mathrm{T}}, \dots, \boldsymbol{g}_n^{\mathrm{T}}]$ olsun, burada $1 \leq i \leq n$ için $\boldsymbol{g}_i^{\mathrm{T}}$ sütun vektörleri sıfırdan farklıdır. Dağıtıcı D ve katılımcı kümesi $P = \{P_1, P_2, \dots, P_n\}$ olmak üzere $\mathbf{G}$ üreteç matrisine (dolayısıyla, C koduna) dayanan sır paylaşım şemasının tasarım yöntemi aşağıda açıklanmaktadır. C kodu F_p sonlu cismi üzerinde tanımlı olduğu için bu sır paylaşım şemasında sır uzayı ve katılımcı paylarının uzayı F_p sonlu cismidir, ve bu şemada tüm işlemler F_p üzerinde yapılmaktadır.

1. **Sır değerini paylaştırma protokolü.**
   - Dağıtıcı D, $P_1, P_2, \dots, P_n$ katılımcıları arasında paylaştırmak için gizli $s_1 \in F_p$ değerini belirler.
   - Dağıtıcı D, F_p cisminden rastgele $s_2, s_3, \dots, s_k$ değerlerini seçer.
   - Dağıtıcı D, $\boldsymbol{s} = (s_1, s_2, \dots, s_k) \in F_p^k$ vektörü için
     $$\boldsymbol{t} = (t_1, t_2, \dots, t_n) \coloneqq \boldsymbol{s}\,\mathbf{G} = (\boldsymbol{s}\cdot\boldsymbol{g}_1, \boldsymbol{s}\cdot\boldsymbol{g}_2, \dots, \boldsymbol{s}\cdot\boldsymbol{g}_n) \in C$$
     pay vektörünü (kod sözcüğünü) hesaplar. Burada " $\cdot$ " işlemi iki vektörün standart iç çarpımını ifade etmektedir.
   - Dağıtıcı D, her bir $i \in \{1, \dots, n\}$ için $t_i \in F_p$ pay değerini sırasıyla P_i katılımcısına gönderir.
2. **Sır değerini kurtarma protokolü**. $1 \le m \le n-1$ olmak üzere $\{P_{i_1}, P_{i_2}, \dots, P_{i_m}\}$ erişim kümesi olsun. İfade edelim ki, $\{P_{i_1}, P_{i_2}, \dots, P_{i_m}\}$ erişim kümesi gizli s_1 değerini elde edebilir ancak ve ancak $\boldsymbol{e}_1 = (1,0,\dots,0)$ vektörü $\boldsymbol{g}_{i_1}, \boldsymbol{g}_{i_2}, \dots, \boldsymbol{g}_{i_m}$ vektörlerinin lineer kombinasyonu olarak yazılabilir. Dolayısıyla, $1 \le j \le m$ için
   $$\boldsymbol{e_1} = x_1\,\boldsymbol{g}_{i_1} + x_2\,\boldsymbol{g}_{i_2} + \cdots + x_m\boldsymbol{g}_{i_m} \qquad (10.5)$$
   olacak şekilde $x_j \in F_p$ elemanları vardır.
   - Dağıtıcı D, denklem (10.5)'te verilen denklem sistemini çözerek $x_j \in F_p$ değerlerini hesaplar, $1 \le j \le m$.
   - Dağıtıcı D, $\{P_{i_1}, P_{i_2}, \dots, P_{i_m}\}$ erişim kümesinin $\{t_{i_1}, t_{i_2}, \dots, t_{i_m}\}$ paylarını ve bu değerleri kullanarak
     $$s_1 = \boldsymbol{s}\cdot\boldsymbol{e_1} = \sum_{j=1}^{m} \mathrm{x_j}\boldsymbol{s}\cdot\boldsymbol{g}_{i_j} = \sum_{j=1}^{m} \mathrm{x_j t_{i_j}}$$
     sır değerini elde eder.

Bu şema, ideal (aynı zamanda mükemmel) bir sır paylaşım şemasıdır. Önerme 10.3'te bu şemanın erişim yapısı tanımlanmaktadır.

Önerme 10.3: C kodu F_p cismi üzerinde $[n,k,d]_p$ parametreli bir lineer kod olsun. C kodunun $\mathbf{G} = [\boldsymbol{g}_1^{\mathrm{T}}, \boldsymbol{g}_2^{\mathrm{T}}, \dots, \boldsymbol{g}_n^{\mathrm{T}}]$ üreteç matrisine dayanan sır paylaşım şemasının erişim yapısı

$$\Gamma = \{\{P_i \mid i \in \mathrm{supp}((x_1, x_2, \dots, x_n))\} : e_1 = x_1\boldsymbol{g}_1 + x_2\boldsymbol{g}_2 + \cdots + x_n\boldsymbol{g}_n\}$$

şeklinde tanımlanır.

Örnek 10.4 Reed-Solomon kodundan elde edilen sır paylaşım şemasını vermektedir.

Örnek 10.4: C kodu F_{11} sonlu cismi üzerinde tanımlı $[5,4,2]_{11}$ parametreli bir Reed-Solomon kodu (aynı zamanda MDS kod) olsun. Bu kodun 4×5 boyutlu üreteç matrisi

$$\mathbf{G} = [\boldsymbol{g}_1^{\mathrm{T}}, \dots, \boldsymbol{g}_5^{\mathrm{T}}] = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 4 & 9 & 5 & 3 \\ 1 & 8 & 5 & 9 & 4 \end{bmatrix}$$

olsun. Dolayısıyla, $\mathbf{G} \in F_{11}^{4\times 5}$ üreteç matrisine sahip C Reed-Solomon kodundan $(4,5)$-eşik sır paylaşım şeması elde edilir. Bu şemada, dağıtıcı D, $s_1 = 3$ sır değerini 5 katılımcı arasında paylaştırsın ve 4 katılımcı ile tekrar elde etsin. Bu şemanın iki temel protokolü aşağıdaki gibi çalışmaktadır.

Sır değerini paylaştırma protokolü. Bu protokolde dağıtıcı D aşağıdaki adımları gerçekleştirir.

- P_1, P_2, P_3, P_4, P_5 katılımcıları arasında paylaştırmak için $s_1 = 3 \in F_{11}$ sır değerini belirler.
- F_{11} uzayından rastgele $s_2 = 2$, $s_3 = 5$ ve $s_4 = 8$ değerlerini seçer.
- Oluşturulan $\boldsymbol{s} = (3,2,5,8) \in F_{11}^4$ vektörü için

$$\boldsymbol{t} = (t_1, t_2, t_3, t_4, t_5) = (3,2,5,8)\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 4 & 9 & 5 & 3 \\ 1 & 8 & 5 & 9 & 4 \end{bmatrix} = (7,3,6,9,5) \in C$$

pay vektörünü hesaplar.

- P_1, P_2, P_3, P_4, P_5 katılımcılarına sırasıyla $t_1 = 7, t_2 = 3, t_3 = 6, t_4 = 9, t_5 = 5$ paylarını gönderir.

Sır değerini kurtarma protokolü. Varsayalım ki $\{P_1, P_2, P_4, \mathrm{P}_5\}$ erişim kümesi olsun. Dağıtıcı D aşağıdaki adımları gerçekleştirir.

- Verilen $\mathbf{G}$ matrisi için $e_1 = x_1\boldsymbol{g_1} + x_2\boldsymbol{g_2} + x_3\boldsymbol{g_4} + x_4\boldsymbol{g_5}$ denklem sistemini çözerek $x_1 = 7, x_2 = 4, x_3 = 9$ ve $x_4 = 3$ değerlerini elde eder.
- $\{P_1, P_2, P_4, \mathrm{P}_5\}$ erişim kümesinin $t_1 = 7, t_2 = 3, t_4 = 9, t_5 = 5$ paylarını ve elde edilen $x_j \in F_{11}$ değerlerini kullanarak $s_1 = x_1t_1 + x_2t_2 + x_3t_4 + x_4t_5 = 3 \bmod 11$ sır değerini ortaya çıkarır.

Literatürde, bu yöntem kullanılarak MDS kodlardan ideal sır paylaşım şeması tasarımı yaygın olarak çalışılmıştır [31] [32] [33] [34]. Renvall ve Ding [34] bazı MDS kodlardan elde edilen ideal sır paylaşım şemalarının erişim yapılarını tanımlamışlardır. Özetle, $\boldsymbol{e}_1 = (1,0,\ldots,0)^{\mathrm{T}}$ birim vektör olmak üzere $[n, k, n-k+1]_p$ parametreli C MDS kodunun $\mathbf{G} = [\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_n]$ üreteç matrisinden elde edilen şemanın erişim yapısı hakkında aşağıdaki sonuçları vermişlerdir [34].

- Bu şemada k katılımcıdan oluşan herhangi bir grup sır değerini elde edebilir.
- $\mathbf{G}$ üreteç matrisinin i-ninci sütunu $\boldsymbol{e}_1$ vektörünün bir sabit katı ise, bu durumda herhangi bir katılımcı kümesinin sır değerini elde edebilmesi için gerek ve yeter koşul bu kümenin yalnızca i-ninci katılımcıdan oluşması veya katılımcı sayısının en az k olmasıdır. Bu erişim yapısına sahip sır paylaşım şemaları önemli uygulama alanlarına sahiptir. Örneğin, bir şirkette gizli bir bilginin sır paylaşım şeması ile şirket çalışanlarından oluşan bir grupta paylaşıldığını varsayalım. Bu gizli bilgiye şirket yöneticisinin tek başına ulaşması veya belirli sayıdaki çalışanın birlikte ulaşması istenebilir. Bu gibi durumlarda MDS kodlardan elde edilen bu şemalar kullanılabilir. Ayrıca, her MDS kodunun bu şekilde üreteç matrisine sahip olabileceğini belirtelim.

- $\boldsymbol{e}_1$ vektörü $\mathbf{G}$ üreteç matrisinin $\boldsymbol{g}_1$ ve $\boldsymbol{g}_2$ sütun vektörlerinin lineer kombinasyonu olsun (fakat bunların herhangi birinin katı olamaz). Bu durumda aşağıdaki ifadeler doğrudur.
  - k tane katılımcıdan oluşan herhangi bir grup sır değerini elde edebilir.
  - $(k-1)$ tane katılımcıdan oluşan bir grup sır değerini elde edebilir eğer t_1 ve t_2 paylarının katılımcıları bu grupta varsa.
  - Katılımcı sayısı en fazla $(k-2)$ olan bir grubun sır değerini elde edebilmesi için gerek ve yeter koşul t_1 ve t_2 paylarının katılımcılarının bu grupta olmasıdır.
- Elde edilen şemanın (k,n)-eşik şeması olması için gerek ve yeter koşul $\boldsymbol{e}_1$ vektörünün $\mathbf{G}$ matrisinin herhangi $(k-1)$ tane sütun vektörlerinin lineer kombinasyonu olarak yazılamamasıdır.

Örnek 10.5: F_p cismi üzerinde tanımlı C kodu verilsin. $1 \le k \le n$ olmak üzere n ve k tam sayıları verilsin. $v_1, v_2, \dots, v_n \in F_p$ sıfırdan ve birbirinden farklı elemanlar olmak üzere C kodunun üreteç matrisi

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ v_1 & v_2 & \cdots & v_n \\ \vdots & \vdots & \vdots & \vdots \\ v_1^{k-1} & v_2^{k-1} & \cdots & v_n^{k-1} \end{bmatrix} \in F_p^{k \times n}$$

şeklinde tanımlansın. Bu durumda, C kodu genelleştirilmiş Reed-Solomon kodudur, aynı zamanda $[n, k, n-k+1]_p$ parametreli bir MDS koddur. Burada, $\boldsymbol{e}_1$ vektörü $\mathbf{G}$ matrisinin herhangi $(k-1)$ tane sütun vektörünün lineer kombinasyonu olarak yazılamadığı için C MDS kodundan elde edilen şema ideal (k,n)-eşik sır paylaşım şemasıdır. Bu şema aynı zamanda (k,n)-eşik Shamir şemasıdır.

10.6.2. Massey Tasarım Yöntemi (İkinci Yöntem)

Literatürde önemli bir yere sahip olan diğer kod tabanlı sır paylaşım şeması tasarım yöntemi 1993 yılında J. L. Massey [47, 48] tarafından önerilmiştir.

Bu çalışmada, lineer kod üzerinde sır paylaşım şeması tanımlanmış ve bu şemanın erişim yapısı ile dual kodun kod sözcükleri arasındaki bağlantı ortaya çıkarılmıştır.

$C \subset F_p^n$ alt uzayı F_p sonlu cismi üzerinde tanımlı $[n, k, d]_p$ parametreli bir lineer kod olsun. C kodunun $k \times n$ boyutlu üreteç matrisi $\mathbf{G} = \left[\boldsymbol{g}_0^{\mathrm{T}}, \boldsymbol{g}_1^{\mathrm{T}}, \dots, \boldsymbol{g}_{(n-1)}^{\mathrm{T}}\right]$ olsun öyle ki $0 \leq i \leq n-1$ için $\boldsymbol{g}_i^{\mathrm{T}}$ sıfırdan farklı sütun vektörleridir. Burada C kodu F_p cismi üzerinde tanımlı olduğu için sır uzayı ve katılımcı paylarının uzayları F_p sonlu cismidir. Dağıtıcı D ve katılımcı kümesi $P = \{P_1, P_2, \dots, P_{n-1}\}$ olmak üzere Massey tarafından önerilen C lineer kodundan sır paylaşım şeması tasarım yöntemi aşağıda açıklanmaktadır.

1. **Sır değerini paylaştırma protokolü.**
   - Dağıtıcı D, $P_1, P_2, \dots, P_{n-1}$ katılımcıları arasında paylaştırmak için gizli $s \in F_p$ değerini belirler.
   - Dağıtıcı D, s sır değerine karşılık gelen katılımcı paylarını hesaplamak için
     $$s = \boldsymbol{u} \cdot \boldsymbol{g}_0$$
     olacak şekilde bir $\boldsymbol{u} = (u_0, u_1, \dots, u_{k-1}) \in F_p^k$ vektörü seçer. Burada, " $\cdot$ " işlemi iki vektörün standart iç çarpımını ifade etmektedir. Bu şekilde p^{k-1} tane $\boldsymbol{u} \in F_p^k$ vektörü vardır ve bu $\boldsymbol{u}$ vektörü bilgi vektörü olarak adlandırılır.
   - Dağıtıcı D, bu vektöre karşılık gelen
     $$\boldsymbol{t} = \left(t_0, t_1, \dots, t_{(n-1)}\right) \coloneqq \boldsymbol{u}\,\mathbf{G} = (\boldsymbol{u} \cdot \boldsymbol{g}_0, \boldsymbol{u} \cdot \boldsymbol{g}_1, \dots, \boldsymbol{u} \cdot \boldsymbol{g}_{n-1}) \in C$$
     pay vektörünü (kod sözcüğünü) hesaplar.
   - Dağıtıcı D, her bir $i \in \{1, \dots, n-1\}$ için t_i pay değerini P_i katılımcısına gönderir.
2. **Sır değerini kurtarma protokolü.** $1 \leq m \leq n-1$ olmak üzere $\{P_{i_1}, P_{i_2}, \dots, P_{i_m}\}$ erişim kümesi olsun. Belirtmek gerekir ki $\{P_{i_1}, P_{i_2}, \dots, P_{i_m}\}$ erişim kümesinin gizli s değerini elde edebilmesi için gerek ve yeter koşul $\boldsymbol{g}_0$ vektörünün $\boldsymbol{g}_{i_1}, \boldsymbol{g}_{i_2}, \dots, \boldsymbol{g}_{i_m}$ vektörleri-

nin lineer kombinasyonu olarak yazılabilmesidir. Bu durumda, $1 \leq i_1 < \cdots < i_m \leq n-1$ için $\boldsymbol{g}_0$ vektörü $\boldsymbol{g}_{i_1}, \boldsymbol{g}_{i_2}, \ldots, \boldsymbol{g}_{i_m}$ vektörlerinin lineer kombinasyonu olmak üzere $\{P_{i_1}, P_{i_2}, \ldots, P_{i_m}\}$ erişim kümeleri bu şemanın erişim yapısını verir. Dolayısıyla, $1 \leq j \leq m$ için

$$\boldsymbol{g}_0 = x_1 \boldsymbol{g}_{i_1} + x_2 \boldsymbol{g}_{i_2} + \cdots + x_m \boldsymbol{g}_{i_m} \qquad (10.6)$$

olacak şekilde $x_j \in F_p$ elemanları vardır.

- Dağıtıcı D, denklem (10.6)'da verilen denklem sisteminden $x_j \in F_p$ değerlerini hesaplar.
- Dağıtıcı D, $\{P_{i_1}, P_{i_2}, \ldots, P_{i_m}\}$ erişim kümesinin $\{t_{i_1}, t_{i_2}, \ldots, t_{i_m}\}$ paylarını ve elde edilen bu değerleri kullanarak

$$s = \sum_{j=1}^{m} \mathrm{x_j t_{i_j}}$$

sır değerini elde eder.

Uyarı: Eğer $\boldsymbol{g}_0$ vektörü $\boldsymbol{g}_{i_1}, \boldsymbol{g}_{i_2}, \ldots, \boldsymbol{g}_{i_m}$ vektörlerinin lineer kombinasyonu olarak yazılamaz ise, $\{P_{i_1}, P_{i_2}, \ldots, P_{i_m}\}$ kümesi s sır değeri hakkında hiçbir bilgi edinemez. Bu durumda, bu sır paylaşım şeması mükemmel sır paylaşım şemasıdır.

Bu tasarım yönteminde katılımcıların paylarının hesaplanma aşamasında $\mathbf{G}$ üreteç matrisi kullanılmaktadır. Dolayısıyla, bu üreteç matrisinin seçimi katılımcıların pay değerlerine etki etmektedir; fakat sır paylaşım şemasının erişim yapısına herhangi bir etkisi yoktur. Bu yöntemin birinci yöntemden en önemli farkı $\mathbf{G}$ üreteç matrisinin, sır paylaşım şemasının erişim yapısını etkilememesidir.

Önerme 10.6, C kodundan elde edilen sır paylaşım şemasının erişim yapısını tanımlamaktadır.

Önerme 10.6: F_p cismi üzerinde $[n, k, d]_p$ parametreli C kodu verilsin. C kodundan elde edilen sır paylaşım şemasının erişim yapısı

$$\Gamma = \left\{ \{P_i \mid i \in \text{supp}\big((a_1, \dots, a_{n-1})\big)\} : \ (1, a_1, \dots, a_{n-1}) \in C^{\perp} \right\}$$

şeklinde tanımlanır.

Örnek 10.7: F_3 sonlu cismi üzerinde tanımlı C kodu $[8,4,4]_3$ verilsin ve bu kodun 4×8 boyutlu üreteç matrisi

$$\mathbf{G} = [\boldsymbol{g}_0^{\mathrm{T}}, \dots, \boldsymbol{g}_7^{\mathrm{T}}] = \begin{bmatrix} 1\,0\,0\,0\,1\,1\,1\,2 \\ 0\,1\,0\,0\,1\,2\,2\,0 \\ 0\,0\,1\,0\,0\,1\,2\,2 \\ 0\,0\,0\,1\,1\,1\,2\,1 \end{bmatrix}$$

olsun. Dolayısıyla, $\mathbf{G} \in F_3^{4\times 8}$ üreteç matrisine sahip olan C kodundan elde edilen sır paylaşım şeması 7 katılımcıdan oluşur. $C^{\perp}$ dual kodunun üreteç matrisi

$$\mathbf{H} = \begin{bmatrix} 1\,0\,0\,0\,2\,2\,0\,2 \\ 0\,1\,0\,0\,2\,1\,2\,2 \\ 0\,0\,1\,0\,2\,1\,1\,1 \\ 0\,0\,0\,1\,1\,0\,1\,2 \end{bmatrix}$$

kullanılarak bu şemanın minimal erişim kümeleri belirlenebilir (bkz. Önerme 10.6). Bu şemada, dağıtıcı $D, s = 2 \in F_3$ sır değerini $P_1, P_2, P_3, P_4, P_5, P_6, P_7$ katılımcıları arasında paylaştırsın ve $\{P_4, \mathrm{P}_5, \mathrm{P}_7\}$ erişim kümesi ile tekrar elde etsin. Bu şemanın iki temel protokolü aşağıdaki gibi çalışmaktadır.

Sır değerini paylaştırma protokolü. Dağıtıcı D, $2 = (u_1, u_2, u_3, u_4) \cdot (1,0,0,0)$ olacak şekilde $\boldsymbol{u} = (2,1,0,0) \in F_3^4$ vektörünü seçer. Daha sonra, seçilen $\boldsymbol{u} = (2,1,0,0) \in F_3^4$ vektörü için

$$\boldsymbol{t} = (t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7) = (2,1,0,0) \begin{bmatrix} 1\,0\,0\,0\,1\,1\,1\,2 \\ 0\,1\,0\,0\,1\,2\,2\,0 \\ 0\,0\,1\,0\,0\,1\,2\,2 \\ 0\,0\,0\,1\,1\,1\,2\,1 \end{bmatrix}$$

$$= (2,1,0,0,0,1,1,1) \in C$$

pay vektörünü hesaplar ve $P_1, P_2, \dots, P_7$ katılımcılarına sırasıyla $t_1, t_2, \dots, t_7$ pay değerlerini gönderir.

Sır değerini kurtarma protokolü. $\{P_4, P_5, P_7\}$ erişim kümesi için dağıtıcı D, $\mathbf{G}$ matrisinin ilgili sütun vektörlerini kullanarak

$$\boldsymbol{g_0} = x_1\boldsymbol{g_4} + x_2\boldsymbol{g_5} + x_3\boldsymbol{g_7}$$

$$(1,0,0,0) = x_1(1,1,0,1) + x_2(1,2,1,1) + x_3(2,0,2,1)$$

denklem sistemini çözer ve $x_1 = x_2 = x_3 = 1 \in F_3$ değerlerini hesaplar. Daha sonra, erişim kümesindeki P_4, P_5, P_7 katılımcılarının $t_4 = 0, t_5 = 1, t_7 = 1$ paylarını ve bu değerleri kullanarak $s = x_1t_4 + x_2t_5 + x_3t_7 = 2$ sır değerini elde eder.

Önerme 10.6'da görüldüğü gibi, C kodundan elde edilen sır paylaşım şemasının erişim kümeleri ile $C^{\perp}$ dual kodunun ilk koordinatı 1 olan kod sözcükleri arasında birebir eşleme vardır. Dolayısıyla, bu önermede $C^{\perp}$ dual kodunun minimal kod olduğunu varsayarsak, C kodundan elde edilen sır paylaşım şemasının minimal erişim kümeleri ile $C^{\perp}$ dual kodunun ilk koordinatı 1 olan minimal kod sözcükleri arasında birebir eşleme olur. Bu kod sözcüklerinin sıfırdan farklı koordinatları (birinci koordinat hariç) minimal erişim kümesindeki katılımcılara karşılık gelmektedir. Monoton erişim yapısına sahip sır paylaşım şemasının erişim yapısını belirlemek için sadece minimal erişim kümelerini bulmak yeterlidir. Dolayısıyla, C kodundan elde edilen sır paylaşım şemasının erişim yapısını belirlemek için $C^{\perp}$ dual kodunun ilk koordinatı 1 olan minimal kod sözcüklerini belirlemek yeterlidir. Böylece, genel olarak çok zor bir problem olan sır paylaşım şemalarının erişim yapılarını belirleme problemi nispeten daha kolay olan lineer kodun minimal kod sözcüklerini belirleme problemine indirgenmiş olur.

C lineer kodunun minimum uzaklığı d, bu koddan elde edilen sır paylaşım şemasının herhangi bir minimal erişim kümesinin eleman sayısının alt sınırını $(d-1)$ belirler. $C^{\perp}$ dual kodun minimum uzaklığı $d^{\perp}$, C kodundan elde edilen sır paylaşım şemasının erişim yapısının ne kadar demokratik olabileceğinin belirlenmesinde önemli rol oynar. Bu iki minimum Hamming uzaklıklar arasında $d + d^{\perp} \leq n + 2$ ödünleşim dengesi (trade-off) vardır. Özellikle, C MDS kod ise, $d + d^{\perp} = n + 2$ olur.

Bir C lineer kodu aslında bir çift sır paylaşım şeması vermektedir. Bu şemalardan bir tanesi C üzerinde tanımlanırken diğeri dual kod $C^{\perp}$ üzerinde tanımlanmaktadır. Önerme 10.8, C minimal kodunun dual kodundan elde edilen sır paylaşım şemasının erişim yapısını tanımlamaktadır.

Önerme 10.8: [13] [52] F_p cismi üzerinde $[n, k, d]_p$ parametreli C minimal kodu verilsin. Bu kodun üreteç matrisi $\mathbf{G} = [\boldsymbol{g}_0^{\mathrm{T}}, \boldsymbol{g}_1^{\mathrm{T}}, \dots, \boldsymbol{g}_{n-1}^{\mathrm{T}}]$ olsun öyle ki $\boldsymbol{g}_i^{\mathrm{T}}$ sütun vektörleri sıfırdan farklıdır. $C^{\perp}$ dual kod üzerindeki sır paylaşım şemasında, $(n-1)$ tane katılımcı $P = \{P_1, P_2, \dots, P_{n-1}\}$ ve p^{k-1} tane minimal erişim kümesi vardır. Bu şemanın erişim yapısı, $C^{\perp}$ dual kodunun minimum Hamming uzaklığına bağlı olarak aşağıdaki gibi tanımlanır.

- $d^{\perp} = 2$ olduğu zaman, $1 \leq i \leq n-1$ olmak üzere
  - eğer $\boldsymbol{g}_i$ vektörü $\boldsymbol{g}_0$ vektörünün bir katı ise, P_i katılımcısı bütün minimal erişim kümelerinde yer alır,
  - eğer $\boldsymbol{g}_i$ vektörü $\boldsymbol{g}_0$ vektörünün bir katı değilse, P_i katılımcısı $(p-1)p^{k-2}$ tane minimal erişim kümesinde yer alır.
- $d^{\perp} \geq 3$ olduğu zaman $1 \leq t \leq \min\{k-1, d^{\perp}-2\}$ aralığındaki sabit herhangi bir t için, t katılımcıdan oluşan her bir grup $(p-1)^t p^{k-(t+1)}$ tane minimal erişim kümesinde yer alır.

Bu önermede, $d^{\perp} = 2$ durumunda bütün minimal erişim kümelerinde yer alan P_i katılımcısı *diktatör* katılımcı olarak adlandırılır ve bu tarz şemalara günlük hayattaki kullanımlarda çok ihtiyaç duyulmaktadır. Örneğin, bir kurumda önemli bir gizli bilgi sır paylaşım şeması kullanılarak paylaşıldığı zaman kurum yöneticisinin her erişim kümesinde yer alması gerekebilir. Literatürde minimal kodlardan diktatör katılımcılara sahip birçok sır paylaşım şeması elde edilmiştir [12] [13] [49] [50] [52]. Diğer yandan, $d^{\perp} \geq 3$ durumunda her katılımcı aynı sayıda minimal erişim kümesinde yer aldığı için her biri aynı role sahiptir. Bu yapıya sahip paylaşım şemaları *demokratik sır paylaşım şemaları* olarak adlandırılır. Ayrıca, t katılımcıya sahip her grup aynı sayıda minimal erişim kümesinde ise, t-inci dereceden demokratik sır paylaşım şeması olarak adlandırılır. Demokratik sır paylaşım şeması gün-

lük hayatta çeşitli alanlarda sıkça ihtiyaç duyulan erişim yapısına sahiptir. Literatürde minimal kodların dual kodlarından birçok demokratik sır paylaşım şeması elde edilmiştir [12] [13] [49] [50] [52].

Örnek 10.9'da minimal kodun dual kodundan elde edilen diktatör katılımcıya sahip sır paylaşım şeması verilmektedir.

Örnek 10.9: F_3 sonlu cismi üzerinde tanımlı C $[13,3,7]_3$ minimal kodunun üreteç matrisi

$$\mathbf{G} = \left[\boldsymbol{g}_0^{\mathrm{T}}, \dots, \boldsymbol{g}_{12}^{\mathrm{T}}\right] = \begin{bmatrix} 2&0&0&1&1&0&2&1&2&1&0&0&0 \\ 1&0&2&2&2&1&2&2&0&0&2&1&0 \\ 2&2&2&0&2&2&1&1&2&0&1&0&1 \end{bmatrix} \in F_3^{3\times 13}$$

olsun. C kodunun dual kodu $C^{\perp}$, $[13,10,2]_3$ parametrelerine ve

$$\mathbf{H} = \left[\boldsymbol{h}_0^{\mathrm{T}}, \dots, \boldsymbol{h}_{12}^{\mathrm{T}}\right] = \begin{bmatrix} 1&0&0&0&0&0&0&0&0&1&0&2&1 \\ 0&1&0&0&0&0&0&0&0&0&0&0&1 \\ 0&0&1&0&0&0&0&0&0&0&0&1&1 \\ 0&0&0&1&0&0&0&0&0&2&0&1&0 \\ 0&0&0&0&1&0&0&0&0&2&0&1&1 \\ 0&0&0&0&0&1&0&0&0&0&0&2&1 \\ 0&0&0&0&0&0&1&0&0&1&0&1&2 \\ 0&0&0&0&0&0&0&1&0&2&0&1&2 \\ 0&0&0&0&0&0&0&0&1&1&0&0&1 \\ 0&0&0&0&0&0&0&0&0&0&1&1&2 \end{bmatrix} \in F_3^{10\times 13}$$

üreteç matrisine sahiptir. $\mathbf{H}$ üreteç matrisine sahip $C^{\perp}$ dual kodundan elde edilen sır paylaşım şemasının erişim yapısı Önerme 10.6 ve 10.8 kullanılarak belirlenebilir. Bu şema 12 katılımcıdan $\{P_1, \dots, P_{12}\}$ oluşur ve şemanın $p^{k-1} = 3^2 = 9$ tane minimal erişim kümesi vardır. Bu erişim kümeleri C kodunun $(1, a_1, \dots, a_{12}) \in C$ şeklindeki kod sözcüklerinin sıfırdan farklı a_i koordinatlarına karşılık gelen P_i katılımcılarından oluşmaktadır. Dolayısıyla, C kodunun $\mathbf{G}$ üreteç matrisi kullanılarak minimal erişim kümeleri aşağıdaki gibi belirlenebilir:

- $MEK_1 = \{P_1, P_2, P_3, P_5, P_7, P_8, P_9, P_{10}, P_{12}\}$,
- $MEK_2 = \{P_1, P_2, P_3, P_6, P_7, P_9, P_{11}, P_{12}\}$,

- $MEK_4 = \{P_1, P_4, P_5, P_7, P_9, P_{10}, P_{11}, P_{12}\}$,
- $MEK_5 = \{P_2, P_3, P_4, P_5, P_6, P_7, P_{10}, P_{11}\}$,
- $MEK_6 = \{P_1, P_2, P_4, P_6, P_7, P_8, P_9, P_{11}, P_{12}\}$,
- $MEK_7 = \{P_1, P_2, P_4, P_5, P_6, P_7, P_8, P_{10}, P_{12}\}$,
- $MEK_8 = \{P_1, P_3, P_5, P_6, P_7, P_8, P_{10}, P_{11}, P_{12}\}$,
- $MEK_9 = \{P_2, P_3, P_4, P_5, P_7, P_8, P_9, P_{10}, P_{11}\}$.

Ayrıca, $C^{\perp}$ dual kodunun minimum Hamming uzaklığının $d^{\perp} = 2$ olduğu kolayca görülebilir. Bu durumda, **G** üreteç matrisinde $\boldsymbol{g}_7 = 2\boldsymbol{g}_0$ olduğu için P_7 katılımcısı bütün minimal erişim kümelerinde bulunur ama diğer tüm katılımcılar $(p-1)p^{k-2} = 6$ tane minimal erişim kümesinde yer alır. Bu şemada dağıtıcı D, $s = 2 \in F_3$ sır değerini 12 katılımcı arasında paylaştırsın ve $MEK_3 = \{P_3, P_4, P_6, P_7, P_8, P_9\}$ erişim kümesi ile tekrar elde etsin. Bu şemanın iki temel protokolü aşağıdaki gibi çalışmaktadır.

Sır değerini paylaştırma protokolü. Dağıtıcı D, $2 = s = \boldsymbol{u} \cdot \boldsymbol{g_0} = (u_1, \dots, u_{10}) \cdot (1,0,0,0,0,0,0,0,0,0)$ olacak şekilde $\boldsymbol{u} = (2,1,1,2,0,0,0,0,1,1) \in F_3^{10}$ vektörünü seçer. Sonra, $\boldsymbol{u}$ vektörünü ve **H** matrisini kullanarak

$$\boldsymbol{t} = (t_0, \dots, t_{12}) = (2,1,1,2,0,0,0,0,1,1) \begin{bmatrix} 1&0&0&0&0&0&0&0&0&1&0&2&1 \\ 0&1&0&0&0&0&0&0&0&0&0&0&1 \\ 0&0&1&0&0&0&0&0&0&0&0&1&1 \\ 0&0&0&1&0&0&0&0&0&2&0&1&0 \\ 0&0&0&0&1&0&0&0&0&2&0&1&1 \\ 0&0&0&0&0&1&0&0&0&0&0&2&1 \\ 0&0&0&0&0&0&1&0&0&1&0&1&2 \\ 0&0&0&0&0&0&0&1&0&2&0&1&2 \\ 0&0&0&0&0&0&0&0&1&1&0&0&1 \\ 0&0&0&0&0&0&0&0&0&0&1&1&2 \end{bmatrix}$$
$$= (2,1,1,2,0,0,0,0,1,1,1,2,1) \in C$$

pay vektörünü hesaplar ve $P_1, \dots, P_{12}$ katılımcılarına sırasıyla $t_1, \dots, t_{12}$ pay değerlerini gönderir.

Sır değerlerini kurtarma protokolü. $MEK_3 = \{P_3, P_4, P_6, P_7, P_8, P_9\}$ erişim kümesi için dağıtıcı D, **H** matrisinin ilgili sütun vektörlerini kullanarak

$$\boldsymbol{h_0} = x_1\boldsymbol{h_3} + x_2\boldsymbol{h_4} + x_3\boldsymbol{h_6} + x_4\boldsymbol{h_7} + x_5\boldsymbol{h_8} + x_6\boldsymbol{h_9}$$

denklem sistemini çözer ve F_3 uzayında $x_1 = x_2 = 1,\ x_3 = 2, x_4 = 1,\ x_5 = 2$ ve $x_6 = 1$ değerlerini elde eder. Son olarak, MEK_3 erişim kümesinin paylarını ve elde edilen bu değerleri kullanarak

$$s = x_1t_3 + x_2t_4 + x_3t_6 + x_4t_7 + x_5t_8 + x_6t_9 = 5 \equiv 2 \bmod 3$$

denkleminden $s = 2$ sır değerini hesaplar.

Örnek 10.10'da minimal kodun dual kodundan demokratik sır paylaşım şeması elde edilmektedir.

Örnek 10.10: F_2 sonlu cismi üzerinde tanımlı $C\ [12, 4, 6]_2$ minimal kodunun üreteç matrisi

$$\mathbf{G} = [\boldsymbol{g}_0^{\mathrm{T}}, \dots, \boldsymbol{g}_{11}^{\mathrm{T}}] = \begin{bmatrix} 0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1 \\ 1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1 \\ 0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1 \\ 0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1 \end{bmatrix} \in F_2^{4\times 12}$$

olsun. Bu durumda, $C^{\perp}$ dual kodundan elde edilen sır paylaşım şemasının erişim yapısı Önerme 10.8 göz önüne alınarak kolayca belirlenebilir. Bu şema 11 katılımcıdan $\{P_1, \dots, P_{11}\}$ oluşur ve şemanın $p^{k-1} = 2^4 = 8$ tane minimal erişim kümesi vardır. Bu erişim kümeleri C kodunun $(1, a_1, \dots, a_{12}) \in C$ şeklindeki kod sözcüklerinin sıfırdan farklı a_i koordinatlarına karşılık gelen P_i katılımcılarından oluşmaktadır. Dolayısıyla, C kodunun **G** üreteç matrisi kullanılarak minimal erişim kümeleri belirlenebilir:

- $MEK_1 = \{P_1, P_6, P_7, P_8, P_9\}$,
- $MEK_2 = \{P_2, P_5, P_7, P_8, P_{10}\}$,
- $MEK_3 = \{P_1, P_4, P_5, P_{10}, P_{11}\}$,
- $MEK_4 = \{P_2, P_4, P_6, P_9, P_{11}\}$,
- $MEK_5 = \{P_3, P_5, P_6, P_8, P_{11}\}$,

- $MEK_6 = \{P_3, P_4, P_7, P_9, P_{10}\}$,
- $MEK_7 = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7\}$,
- $MEK_8 = \{P_1, P_2, P_3, P_8, P_9, P_{10}, P_{11}\}$.

Bu kümelerden görüldüğü gibi her bir katılımcı 4 farklı erişim kümesinde yer almaktadır. Dolayısıyla, bu sır paylaşım şeması 1-demokratiktir.

10.6.3. Çoklu Sır Paylaşım Şeması Tasarımı

Bu bölümde, Bölüm 10.6.1'de verilen McEliece-Sarwat tasarım yönteminden elde edilen çoklu sır paylaşım şeması tasarım yöntemi verilecektir.

F_p sonlu cismi üzerinde tanımlı $[n, k, d]_p$ parametreli C lineer kodunun $k \times n$ boyutlu üreteç matrisi $\mathbf{G} = [\boldsymbol{g}_1^{\mathrm{T}}, \boldsymbol{g}_2^{\mathrm{T}}, \dots, \boldsymbol{g}_n^{\mathrm{T}}]$ olsun. C kodunun $\mathbf{G} \in F_p^{k \times n}$ üreteç matrisinden elde edilen çoklu sır paylaşım şeması k tane $s_1, s_2, \dots, s_k$ sır değerini aynı anda n tane katılımcı $P = \{P_1, P_2, \dots, P_n\}$ arasında paylaştırır. $1 \leq u \leq n$ için u eşik değer olmak üzere bu şema (u, k, n)-

eşik çoklu sır paylaşım şeması olarak adlandırılır. k tane sır değerinin sır vektörü $\boldsymbol{s} = (s_1, s_2, \dots, s_k) \in F_p^k$ ve n tane katılımcının pay vektörü $\boldsymbol{t} = (t_1, t_2, \dots, t_n) \in F_p^n$ olsun. Burada $t_1, t_2, \dots, t_n$ değerleri sırasıyla $P_1, P_2, \dots, P_n$ katılımcılarının paylarıdır.

Sır değerlerini paylaştırma protokolü.

- Dağıtıcı D, $P_1, P_2, \dots, P_n$ katılımcıları arasında paylaştırmak için F_p cisminden k tane $s_1, s_2, \dots, s_k$ sır değerini seçer ve $\boldsymbol{s} = (s_1, s_2, \dots, s_k) \in F_p^k$ sır vektörünü oluşturur.
- Dağıtıcı D, oluşturulan $\boldsymbol{s} = (s_1, s_2, \dots, s_k) \in F_p^k$ sır vektöründen

  $$\boldsymbol{t} = (t_1, t_2, \dots, t_n) := \boldsymbol{s}\,\mathbf{G} = (\boldsymbol{s} \cdot \boldsymbol{g}_1, \boldsymbol{s} \cdot \boldsymbol{g}_2, \dots, \boldsymbol{s} \cdot \boldsymbol{g}_n) \in C$$

  pay vektörünü hesaplar.
- Dağıtıcı D, her bir $i \in \{1, \dots, n\}$ için $t_i \in F_p$ pay değerini P_i katılımcısına gönderir.

Sır değerlerini kurtarma protokolü. $1 \leq u \leq n$ ve $1 \leq i_1 < \cdots < i_u \leq n$ olmak üzere $\{P_{i_1}, P_{i_2}, \ldots, P_{i_u}\}$ erişim kümesi olsun. Bu durumda, $\mathbf{G}$ matrisinin $\boldsymbol{g}_{i_1}^{\mathrm{T}}, \boldsymbol{g}_{i_2}^{\mathrm{T}}, \ldots, \boldsymbol{g}_{i_u}^{\mathrm{T}}$ sütun vektörlerinden oluşan alt matris $\mathbf{G}(i_1, i_2, \ldots, i_u)$ olmak üzere dağıtıcı D, erişim kümesinin $\boldsymbol{t}$ pay vektörünü ve bu alt matrisi kullanarak

$$\boldsymbol{s}\mathbf{G}(i_1, i_2, \ldots, i_u) = \left(\boldsymbol{s} \cdot \boldsymbol{g}_{i_1}, \boldsymbol{s} \cdot \boldsymbol{g}_{i_2}, \ldots, \boldsymbol{s} \cdot \boldsymbol{g}_{i_u}\right) = (t_{i_1}, t_{i_2}, \ldots, t_{i_u}) \qquad (10.7)$$

lineer denklem sistemini çözer ve $\boldsymbol{s} = (s_1, s_2, \ldots, s_k) \in F_p^k$ sır vektörünü elde eder. Böylece, k tane sır değeri elde edilir.

Not: Bu denklem sistemi (10.7), *Gauss eleme yöntemi* ile çok hızlı bir şekilde çözülebilir. Dolayısıyla, k tane sır değerini elde etme işlemi bu lineer kodun kod çözme işleminden çok daha hızlıdır.

Bu şemada **s** sır vektöründen **t** katılımcı paylarının vektörünü hesaplama işlemi, F_p^k uzayından F_p^n uzayına giden $f(\boldsymbol{s}) = \boldsymbol{s}\,\mathbf{G} = \boldsymbol{t}$ fonksiyonu olarak tanımlanabilir. Bu fonksiyon birebir fonksiyondur ve *paylaşım fonksiyonu* olarak adlandırılır. f paylaşım fonksiyonuna sahip bir şemada, eğer her $a, a' \in F_p$ sayıları ve her $\boldsymbol{s}, \boldsymbol{s}' \in F_p^k$ vektörleri için $f(a\boldsymbol{s} + a'\boldsymbol{s}') = af(\boldsymbol{s}) + a'f(\boldsymbol{s}')$ sağlanıyorsa bu şema *lineer sır paylaşım şeması* olarak adlandırılır. Örneğin, polinom interpolasyonuna dayanan Shamir sır paylaşım şeması lineerdir. Yukarıda tanımlanan çoklu sır paylaşım şemasının da lineer olduğu kolayca görülebilir. Bu lineer çoklu sır paylaşım şeması ile üreteç matrisi $\mathbf{G}$ olan $C\ [n, k, d]_p$ lineer kodu arasında birebir eşleme vardır. Bu paylaşım fonksiyonu aynı zamanda C kodunun kodlama fonksiyonudur.

Önerme 10.11: Yukarıda verilen tasarım yönteminde, C kodundan tasarlanan lineer çoklu sır paylaşım şeması (k, k, n)-eşik şemasıdır ancak ve ancak

(i) $\mathbf{G}$ üreteç matrisine sahip olan C kodu $[n, k, n-k+1]_p$ parametreli MDS kodudur, ve

(ii) $\mathbf{G}$ matrisinin herhangi $(k-1)$ tane sütun vektörlerinin kümesi $[k, k-1, 2]_p$ parametreli başka bir MDS kod tanımlar.

Önerme 10.11 göz önüne alındığında, (k, k, n)-eşik çoklu sır paylaşım şeması tasarlamak için herhangi bir üreteç matrisine sahip $[n, k, n-k+1]_p$ parametreli MDS kod kullanmanın yeterli olmadığı görülmektedir. Dolayısıyla, kullanılan MDS kod için *(ii)* şartını sağlayan $\mathbf{G}$ üreteç matrisinin bulunması gerekmektedir (bir MDS kodun her üreteç matrisi *(ii)* şartını sağlamayabilir).

Örnek 10.12'de, *(ii)* şartını sağlayan $\mathbf{G}$ üreteç matrisine sahip MDS kodundan (k, k, n)-eşik çoklu sır paylaşım şeması tasarımı verilmektedir.

Örnek 10.12: F_7 sonlu cismi üzerinde tanımlı C $[4, 3, 2]_7$ Reed-Solomon kodu (aynı zamanda MDS kodu) verilsin. Bu kodun 3×4 boyutlu üreteç matrisi

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 1 & 4 & 2 & 2 \end{bmatrix}$$

olsun. $\mathbf{G}$ üreteç matrisinin yukarıdaki *(ii)* şartını sağladığı kolayca görülebilir. Dolayısıyla, $\mathbf{G}$ üreteç matrisine sahip C MDS kodundan $(3, 3, 4)$-eşik çoklu sır paylaşım şeması tasarlanabilir. Bu şemanın iki temel protokolü aşağıdaki gibi çalışmaktadır.

Sır değerini paylaştırma protokolü. Bu protokolde dağıtıcı D aşağıdaki adımları gerçekleştirir.

- P_1, P_2, P_3, P_4 katılımcıları arasında paylaştırmak için $s_1 = 3, s_2 = 4$ ve $s_3 = 5$ sır değerlerini belirler ve $\boldsymbol{s} = (3, 4, 5) \in F_7^3$ sır vektörünü oluşturur.
- $\boldsymbol{s} = (3, 4, 5) \in F_7^3$ sır vektörünü ve $\mathbf{G}$ üreteç matrisini kullanarak

$$\boldsymbol{t} = (3, 4, 5) \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 1 & 4 & 2 & 2 \end{bmatrix} = (5, 3, 4, 1) \in C$$

  pay vektörünü hesaplar.
- P_1, P_2, P_3, P_4 katılımcılarına sırasıyla $t_1 = 5, t_2 = 3, t_3 = 4, t_4 = 1$ paylarını gönderir.

Sır değerlerini kurtarma protokolü. Varsayalım ki $\{P_1, P_2, P_4\}$ erişim kümesi olsun. Dolayısıyla, D

$$(s_1, s_2, s_3)\begin{bmatrix}1 & 1 & 1\\ 1 & 2 & 4\\ 1 & 4 & 2\end{bmatrix} = (5, 3, 1)$$

lineer denklem sistemini çözerek $\boldsymbol{s} = (3, 4, 5) \in F_7^3$ sır vektörünü elde eder. Sonuç olarak, $s_1 = 3, s_2 = 4$ ve $s_3 = 5$ sır değerleri ortaya çıkarılır.

Not: Yukarıda tasarlanan lineer (k, k, n)-eşik çoklu sır paylaşım şemasında,

- her katılımcı sır vektörü hakkında aynı oranda bilgiye sahiptir, ve
- aynı sayıda katılımcıya sahip iki erişim kümesi sır vektörü hakkında aynı oranda bilgiye sahiptir.

Dolayısıyla, bu şemalar bilinen en demokratik şemalardır.

10.7. SIR PAYLAŞIM ŞEMALARININ BLOKZİNCİR UYGULAMALARI

Blokzincir teknolojisi, 2008 yılında Satoshi Nakamoto tarafından yazılan Bitcoin makalesi [6] ile birlikte hem endüstride hem de akademide büyük ilgi görmeye başlamıştır. Blokzincir sistemlerinde, işlem verilerinin gizliliği, kullanıcı anonimliği, kimlik denetimi, uzlaşma ve güvenli ödeme sisteminin oluşturulması gibi temel gereksinimler kriptografik yöntemlerle sağlanabilmektedir. Bu bölümde, blokzincir sistemlerinde kullanılan sır paylaşım şemalarının kullanım amaçlarından bahsedilecektir (diğer kriptografik yöntemlerin kullanım amaçları hakkında detaylı bilgi için [4] numaralı güncel kaynaktan faydalanılabilir). Sır paylaşım şemaları geleneksel blokzincir sistemlerinde şifreleme/imzalama anahtarlarını paylaştırmak için kullanılırken dağıtık depolamalı blokzincir (DSB) sistemlerinde depolama maliyetini iyileştirmek için kullanılmaktadır.

Blokzincir sistemlerinde kullanılan en önemli kriptografik yöntemlerden birisi elektronik imzalama işlemidir. Bu sistemlerde imzalama işleminin temel amacı veri bütünlüğünü sağlamak ve kimlik denetimini yapmaktır. Bunlara ek olarak kullanıcı anonimliğini sağlamak için eşik imzalama yöntemi gibi çeşitli imzalama yöntemleri kullanılmaktadır. Eşik imzalama yöntemi, imzalama algoritmasının eşik sır paylaşım şeması ile birlikte kul-

lanılmasıyla elde edilmektedir. Eşik imzalama yönteminde imzalama anahtarı blokzincir ağındaki katılımcılar arasında eşik sır paylaşım şeması ile güvenli bir şekilde paylaştırılmaktadır. Örneğin, n katılımcıdan oluşan bir blokzincir ağında herhangi t tane katılımcının birleşerek blok/işlem verisini imzalaması gerektiğini düşünelim. Bu durumda, imzalama anahtarı blokzincir ağındaki n katılımcı arasında (t, n)-eşik sır paylaşım şeması ile paylaştırılır ve imzalama işlemi yapılacağı zaman bu katılımcılar arasından en az t katılımcı (erişim kümesi) paylarını kullanarak blok verisini birlikte imzalar. İmzalama aşamasında erişim kümesindeki katılımcılar imzalama anahtarını ortaya çıkarmadan direkt olarak imzalama işlemini gerçekleştirir. Böylece, gizli kalması gereken imzalama anahtarı hiçbir yerde bir bütün olarak tutulmaz ve bu anahtara hiç kimse erişemez. Bu imzalama yöntemi (t, n)-eşik imzalama yöntemi olarak adlandırılır. Son zamanlarda, geleneksel blokzincir sistemlerinde veri gizliliğini artırmak ve kullanıcı anonimliğini güçlendirmek için çeşitli eşik imzalama şemaları önerilmiştir.

Örneğin, Ziegeldorf ve ark. [5] Bitcoin blokzincirinde kullanıcı anonimliğini iyileştirmek için sMPC protokolü kullanarak ECDSA imzalama algoritmasının eşik varyant şemasını önermişlerdir. Burada imzalama algoritmasının eşik şema olmasını sağlayan sMPC protokolü içerisinde kullanılan ve imzalama anahtarını paylaştıran eşik sır paylaşım şemasıdır. Ayrıca, homomorfik mini-blokzincir sistemi üzerinde eşik imzalama yöntemi için Shamir eşik sır paylaşım şeması kullanılmaktadır.

Blokzincir sistemlerinde, sır paylaşım şemalarının diğer bir kullanım amacı şifreleme anahtarlarının paylaştırılmasıdır. Geleneksel blokzincir sistemlerinde genel olarak blok/işlem verileri şifrelenmemektedir; fakat bazı blokzincir sistemleri üzerinde çalışan kripto paraların kullanıcı cüzdanları simetrik şifreleme algoritmaları ile şifrelenmekte ve şifreleme anahtarları sır paylaşım şemaları ile paylaşılmaktadır. Örneğin, bitcoin cüzdanları (çevrimiçi veya çevrimdışı) simetrik şifreleme algoritması ile şifrelenmekte ve kullanıcının gizli anahtarı sır paylaşım şeması ile saklanmaktadır. Diğer taraftan, dağıtık depolamalı blokzincir (DSB) sistemlerinde blok verileri simetrik şifreleme algoritması ile şifrelenmektedir. Bu sistemlerde şifreleme anahtarları sır paylaşım şemaları ile blokzincir ağındaki katılımcılar arasında güvenli bir şekilde paylaştırılmaktadır [7] [8] [9]. Aslında, DSB sistemlerinde kul-

lanılan sır paylaşım şemaları bu sistemlerin depolama maliyetlerinin iyileştirilmesine büyük katkı sağlamaktadır.

Blokzincir sistemlerinde, her katılımcı tüm blok verilerinin bir kopyasını saklamak zorundadır. Bu sistemlerde blok sayısı sürekli artacağı için zamanla blok verilerinin depolanmasının büyük bir problem olması kaçınılmazdır. Bu problemin çözümü için literatürde DSB sistemi önerilmiştir [8]. Bu sistemlerde blok verilerini blokzincir ağındaki katılımcılar arasında paylaştırmak için MDS kodlarla birlikte çeşitli sır paylaşım şemaları kullanılmaktadır. Literatürde bu amaç için Shamir sır paylaşımı, yerel sır paylaşımı (Local Secret Sharing - LSS), çoklu sır paylaşımı, doğrulanabilir eşik sır paylaşımı gibi çeşitli şemalar önerilmiştir [7] [8] [9] [10]. Bu çalışmalarda, MDS kodlar ve önerilen şemalar birlikte kullanılarak blok verileri blokzincir ağındaki kullanıcılar arasında paylaştırılmakta ve böylece her bir kullanıcının tüm veriyi depolaması yerine belirli bir kısmını depolaması sağlanmaktadır. Bu çalışmalarda önerilen DSB sistemleri ve sır paylaşım şemaları aşağıda açıklanmaktadır.

Raman ve Varshney [8] 2018 yılında geleneksel blokzincir sistemlerindeki depolama maliyetlerini blokzincir ağındaki katılımcılara paylaştırmak için dağıtık depolamalı blokzincir (DSB) sistemini önermişlerdir. Bu çalışmada önerilen DSB sisteminde blok verilerini dağıtmak için Shamir paylaşım şeması kullanılmaktadır. DSB sistemi içerisinde t-inci bloğun blok verisi $\mathbf{B}^{(t)}$ olsun. Bu sistemde, tüm katılımcıların kümesi (diyelim ki n katılımcı var), her biri m elemanlı L tane alt kümeye bölünür. $l = 1, \dots, L$ olmak üzere her A_l alt kümesi, t-inci bloğun blok verisini şifrelemek için kullanılan simetrik şifreleme algoritmasının $K_l^{(t)}$ gizli anahtarına sahip olsun (her alt kümenin kendisine özel gizli anahtarı mevcuttur). A_l alt kümesi kendisine ait gizli anahtarı kullanarak şifreleme algoritması ile $\mathbf{B}^{(t)}$ blok verisini $\mathbf{m}_l^{(t)} = E_{K_l^{(t)}}\left(\mathbf{B}^{(t)}\right)$ şeklinde şifreler. Bu şifreli blok verisi MDS $[m, m]$ kodu ile kodlanarak A_l kümesindeki katılımcılara dağıtılır. Ayrıca, $K_l^{(t)}$ gizli anahtarı ve blok verisinin özet değeri $W^{(t)}$ iki bağımsız Shamir (m, m)-eşik paylaşım şeması ile A_l kümesindeki katılımcılar arasında paylaştırılır. Önerilen bu yöntem sayesinde mev-

cut blokzincir sisteminin depolama maliyeti büyük oranda azaltılmıştır. Diğer taraftan, A_l alt kümesindeki herhangi bir katılımcının anahtar parçasını (Shamir şeması ile kendisine verilen pay değeri) kaybetmesi, bu kümenin $K_l^{(t)}$ gizli anahtarının kaybolmasına sebep olmaktadır. Eğer DSB sistemindeki her alt kümeden en az 1 katılımcı anahtar parçasını kaybederse, bu bloktaki blok verisine erişilemez. Daha açık bir ifadeyle, her bir alt kümenin gizli anahtarının kaybolması hiçbir alt kümenin şifreli blok verisini çözemeyeceği anlamına gelir, dolayısıyla $\mathbf{B}^{(t)}$ blok verisine ulaşılamaz. Bu sorun $1 < u < m$ olmak üzere Shamir (u, m)-eşik şeması kullanılarak çözülebilir; ancak bu durumda her bir kullanıcının depolama maliyeti artmaktadır.

2019 yılında Kim ve ark. [9] mevcut DSB sistemi için daha önce [8] çalışmasında elde edilen sonuçları iyileştirmek amacıyla yerel olarak kurtarılabilir kodları (locally recoverable codes - LRC) kullanarak yerel sır paylaşım şemasını (LSS) önerdiler. DSB sistemi içerisinde, önerilen $(m-1, m)$-eşik LSS şeması kullanılarak her bir alt kümenin gizli anahtarı $K_l^{(t)}$ o kümenin katılımcıları arasında paylaştırılırken blok verisinin özet değeri $W^{(t)}$ blokzincir ağındaki tüm katılımcılar arasında paylaştırılmaktadır. Böylece, önerilen LSS şeması sayesinde her bir alt kümenin gizli anahtarı ve blok verisinin özet değeri eş zamanlı olarak bir tek şema kullanılarak paylaştırılmaktadır. Ayrıca, her bir alt kümeye ait şifreli blok verisi $[m, m-1]$ MDS kodu ile kodlanarak alt kümenin katılımcıları arasında dağıtılmaktadır. Önerilen bu LSS şeması sayesinde, DSB sisteminin kurtarma iletişim (recovery communication) maliyeti iyileştirilmiş ve dayanıklılığı (robustness) artırılmıştır. Diğer yandan, bir $[n, k, m-1]$ LRC kodundaki tek bir hata $(m-1)$ tane doğru sembolle kurtarılabildiği için DSB sistemindeki bir hata $(m-1, m)$-eşik LSS şeması tarafından düzeltilebilmektedir. Dolayısıyla, her bir alt kümeden en fazla 1 katılımcının anahtar parçasını kaybetmesi DSB sisteminde hiçbir sorun yaratmamaktadır. Ama her alt kümeden en az 2 katılımcının anahtar parçasını kaybetmesi blok verisinin tamamen kaybolmasına sebep olmaktadır.

2020 yılında Mesnager ve ark. [7] mevcut DSB sistemini iyileştirmek için doğrulanabilir çoklu eşik sır paylaşım (VMSS) şeması önerdiler. Önerilen

bu şema ile DSB sistemindeki her bir alt küme, $K_l^{(t)}$ gizli anahtarını ve $W^{(t)}$ blok özet değerini aynı anda katılımcıları arasında paylaştırmaktadır. Önerilen VMSS şeması sayesinde, DSB sistemi için daha önce [8] ve [9] çalışmalarında elde edilen sonuçlar iyileştirilmiştir. Daha açık söylemek gerekirse, DSB sisteminin kurtarma iletişim maliyeti düşürülmüş ve dayanıklılığı artırılmıştır. Ayrıca, önerilen şemanın eşik değerinin esnek olması sayesinde az sayıdaki kullanıcıların anahtar parçalarını kaybetmeleri blok verilerinin kaybolmasına sebep olmamaktadır. Diğer taraftan, önerilen şemanın doğrulama algoritması sayesinde sistem içerisindeki kullanıcıların ve dağıtıcının hile yapmaları önlenebilmektedir (sistem aktif saldırılara karşı güvenlidir). Önerilen şemanın DSB sistemi için diğer bir avantajı ise bu şemada kullanıcıların paylarının açık anahtarlı kriptografi ile şifrelenerek gönderilmesidir (bu durumda sistem içerisinde dağıtıcı ve katılımcılar arasında güvenli kanallar olmadan güvenli iletişim sağlanabilmektedir). Dolayısıyla, bu çalışmada önerilen VMSS şeması mevcut DSB sistemini birçok farklı yönden iyileştirmektedir.

Diğer yandan, Chen ve ark. [10] mevcut DSB sisteminde blok verilerini şifrelemeden katılımcılar arasında paylaştıran çoklu sır paylaşım şeması tasarlamışlardır. Bu çalışmada, polinom interpolasyonuna dayanan çoklu sır paylaşım şemasına sahip düşük-depolama şeması (low-storage scheme) önerilmiştir. Önerilen şema blok verilerini birçok parçaya bölerek her bir parçayı blokzincir ağındaki katılımcılar arasında paylaştırmaktadır. Bu DSB sisteminde, $\mathbf{B}^{(t)}$ blok verisini n katılımcı arasında paylaştırmak için önce bu blok verisi $b_1, b_2, \dots, b_m$ şeklinde m eşit parçaya bölünür öyle ki $b_1||b_2||\dots||b_m = \mathbf{B}^{(t)}$ ve $m < n$. Bu şema özyinelemeye dayanmakta ve $i = 1, \dots, m$ için b_i parçasını kodlamak için F_p cismi üzerinde tanımlı olan

$$g_i(x) = b_i + g_{(i-1,1)}x + g_{(i-1,2)}x^2 + \cdots + g_{(i-1,i)}x^i$$

paylaşım polinomlarını kullanmaktadır. Burada p asal sayısı b_i parçalarından ve n sayısından büyük olmalıdır, dolayısıyla şemanın çalıştığı F_p cisminin çok büyük olması gerekmektedir. Bu şemada, $g_m(x_1), g_m(x_2), \dots, g_m(x_n)$ pay değerleri blokzincir ağındaki ilgili katılımcılara dağıtılır. Burada

$x_1, x_2, \ldots, x_n$ girdi değerleri katılımcıların açık bağlantılarıdır. Bu şema $(m+1, n)$-eşik çoklu sır paylaşım şemasıdır çünkü herhangi $m+1$ veya daha fazla katılımcı $\mathbf{B}^{(t)}$ blok verisini elde edebilirken m veya daha az katılımcıdan oluşan hiçbir grup bu veriyi elde edemez.

10.8. SONUÇ VE DEĞERLENDİRMELER

Kodlama teorisinin en önemli kod sınıfını oluşturan lineer kodlar sır paylaşım şemaları tasarlamak için kullanılabilmektedir. Özellikle, MDS kodlar ve minimal kodların dual kodları çok iyi erişim yapılarına sahip paylaşım şemaları vermektedir, ve bu şemalar günlük hayat kullanımında önemli uygulama alanlarına sahiptir. Örneğin, pratik kullanımda ihtiyaç duyulan mükemmel ve ideal şemalar MDS kodlardan elde edilirken demokratik şemalar ve diktatör katılımcıya sahip şemalar minimal kodların dual kodlarından elde edilmektedir. Bu kitap bölümünde, sır paylaşım şemalarının lineer kodlar üzerindeki tasarım yöntemleri verilmiş ve bu yöntemler kullanılarak paylaşım şemaları tasarlanmıştır. Örneklerle küçük parametreli kodlardan paylaşım şemaları elde edilmiş ve bu şemalarla örnek sır değerleri paylaştırılmıştır. Bu bölümde, sır paylaşım şemalarının dağıtık veri tabanlarındaki ve blokzincir sistemlerindeki güncel kullanımlarına da değinilmiştir. Bu şemalar, blokzincir sistemlerinde ve veri tabanlarında gizli anahtarları paylaştırmak için tek başlarına kullanılabildikleri gibi ZKP, sMPC ve OT protokolleri içerisinde gizlilik artırıcı protokol olarak da kullanılabilmektedir. Bu şemalar ayrıca kripto para cüzdanlarında gizli anahtarları saklamak için de kullanılmaktadır. Bu şemaların diğer bir kullanım amacı ise DSB sistemlerinde depolama ve iletişim maliyetlerini iyileştirmektir.

Literatürde, MDS kodlardan elde edilen ideal şemalar ayrıntılı olarak çalışılmasına rağmen minimal kodların dual kodlarından elde edilen demokratik şemalar üzerine çok fazla çalışma olmadığı görülmektedir. Demokratik şemaların lineer kodlarla ilişkisi ve pratik kullanımda sağladığı avantajlar göz önüne alındığında bu şemaların detaylı çalışılması gerektiği söylenebilir. Diğer taraftan, DSB sistemlerinde depolama ve iletişim maliyetlerini iyileştirmek için bu sistemlerde ideal ve demokratik şemaların kullanım avantajları araştırılabilir ve yeni şemalar önerilebilir. Dolayısıyla, mevcut DSB sistemlerini iyileştirmek için bu sistemlere doğrulanabilir ideal ve demokratik sır paylaşım şemalarının uyarlaması yapılabilir.

Doğrulanabilir sır paylaşım şemaları blokzincir sistemleri ve dağıtık veri tabanlarında yaygın olarak kullanılmaktadır. Bu şemaların doğrulama mekanizmalarındaki tek yönlü fonksiyonların güvenliği, ayrık logaritma probleminin zorluğuna dayanmaktadır. Son zamanlarda, küçük ölçekli kuantum bilgisayarların üretilmesiyle birlikte birçok alanda olduğu gibi bu fonksiyonlar için de kuantum güvenli olduğu düşünülen kafes tabanlı problemlerin kullanımı önerilmiştir [7] [42]. Bu fonksiyonlar için kuantum güvenli olduğu düşünülen sendrom kod çözme probleminin ve aynı sayıda noktaya sahip iki eğri arasında izojeni oluşturma probleminin kullanılması da mümkün olabileceği öngörülmektedir. Dolayısıyla, doğrulanabilir şemaların uygulanacağı platformlar dikkate alınarak bu problemler doğrulama mekanizması için analiz edilebilir ve uygun problem belirlenerek yeni kuantum güvenli doğrulanabilir şemalar tasarlanabilir. Ayrıca, bu bölümde verilen kod tabanlı şemalara kuantum güvenli doğrulama mekanizmaları eklenerek kod tabanlı doğrulanabilir şemalar tasarlanabilir. Böylece, potansiyel kuantum sonrası blokzincir sistemleri için kuantum güvenli doğrulanabilir sır paylaşım şemaları elde edilebilir.

KAYNAKLAR

[1] S. Bartolucci, P. Bernat and D. Joseph, "SHARVOT: secret SHARe-based VOTing on the blockchain," in Proc. WETSEB '18, Gothenburg, Sweden, pp. 30–34, 2018.

[2] G. Zyskind, O. Nathan and A. Pentland, "Enigma: Decentralized computation platform with guaranteed privacy," MIT, USA, 2015.

[3] S. Choi, S. Haruta, Y. An and I. Sa, "A Server-Based Distributed Storage Using Secret Sharing with AES-256 for Lightweight Safety Restoration," *IEICE TRANS. INF. & SYST.,* vol. 103, no. 7, pp. 1647-1659, 2020.

[4] M. Raikwar, D. Gligoroski and A. K. Kralevska, "SoK of Used Cryptography in Blockchain," *IEEE Access,* vol. 7, pp. 148550-148575, 2019.

[5] J. H. Ziegeldorf, F. Grossmann, M. Henze and N. Inden, "CoinParty: Secure Multi-Party Mixing of Bitcoins," in Proc. CODASPY '15, San Antonio Texas USA, pp. 75–86, 2015.

[6] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review,* vol. 21260, pp. 1-9, 2008.

[7] S. Mesnager, A. Sınak and O. Yayla, "Threshold-Based Post-Quantum Secure Verifiable Multi-Secret Sharing for Distributed Storage Blockchain," *Mathematics,* vol. 8, no. 12, pp. 1-15, 2020.

[8] R. K. Raman and L. R. Varshney, "Distributed Storage Meets Secret Sharing on the Blockchain," in Proc. ITA, San Diego, CA, USA, pp. 1–6, 2018.

[9] Y. Kim, R. K. Raman, Y.-S. Kim, L. R. Varshney and N. R. Shanbhag, "Efficient local secret sharing for distributed blockchain systems," *IEEE Communications Letters,* vol. 23, no. 2, pp. 282-285, 2018.

[10] H. Chen, H. Wu, C. Chang and L. Chen, "Light Repository Blockchain System with Multisecret Sharing for Industrial Big Data," *Security and Communication Networks,* vol. 2019, pp. 1-8, 2019.

[11] C. Ding, "Secret Sharing with Linear Codes," in *Concise Encyclopedia of Coding Theory*, USA, Chapman and Hall/CRC, 2021, pp. 785-797.

[12] J. Yuan and C. Ding, "Secret Sharing Schemes From Three Classes of Linear Codes," *IEEE Trans. Inform. Theory,* vol. 52, no. 1, pp. 206-2012, 2006.

[13] C. Carlet, C. Ding and J. Yuan, "Linear codes from perfect nonlinear mappings and their secret sharing schemes," *IEEE Trans. Inform. Theory,* vol. 51, no. 6, pp. 2089-2102, 2005.

[14] H. Özadam, F. Özbudak and Z. Saygı, "Secret sharing schemes and linear codes," in Proc. ISCTURKEY, Ankara, Turkey, pp. 101–106, 2007.

[15] E. Androulaki and et al., "Hyperledger fabric: A distributed operating system for permissioned blockchains," in Proc. EuroSys '18, Porto, Portugal, pp. 1–15, 2018.

[16] V. Buterin, "A next-generation smart contract and decentralized application platform," *white paper,* vol. 3, no. 37, pp. 1-36, 2014.

[17] Y. Liu, F. Zhang and J. Zhang, "Attacks to some verifiable multi-secret sharing schemes and two improved schemes," *Information Sciences,* vol. 329, no. 1, pp. 524-539, 2016.

[18] T. M. Fernandez Carames and P. Fraga-Lamas, "Towards Post-Quantum Blockchain: A Review on Blockchain Cryptography Resistant to Quantum Computing Attacks," *IEEE ACCESS,* vol. 8, pp. 21091-21116, 2020.

[19] S. Ayleylek and M. Soysaldı, "Kuantum Bilgisayarlar ile Kriptoanaliz ve Kuantum Sonrası Güvenilir Kripto Sistemleri," in *Siber Güvenlik ve Savunma: Problemler ve Çözümler*, Ankara, Grafiker, 2019, pp. 137-168.

[20] A. Shamir, "How to share a secret," *Communications of the ACM,* vol. 22, no. 11, pp. 612-613, 1979.

[21] G. R. Blakley, "Safeguarding cryptographic keys," in *Managing Requirements Knowledge, International Workshop on IEEE Computer Society*, New York, 1979.

[22] R. Cramer, I. B. Damgård and . J. B. Nielsen, Secure Multiparty Computation and Secret Sharing, Cambridge: Cambridge University Press, 2015.

[23] B. Chor, S. Goldwasser, S. Micali and B. Awerbuch, "Verifiable secret sharing and achieving simultaneity in the presence of faults," in Proc. SFCS, Portland, USA, pp. 383-395, 1985.

[24] B. Schoenmakers, "A simple publicly verifiable secret sharing scheme and its application to electronic voting," in Proc. CRYPTO'99, Wiener M. (ed). Lecture Notes in Computer Science, vol 1666. Springer, Berlin, Heidelberg, pp. 148-164, 1999.

[25] W. A. Jackson, K. M. Martin and C. M. O'Keefe, "Multisecret Threshold Schemes," in Proc. CRYPTO'94, Desmedt Y.G. (eds). Lecture Notes in Computer Science, vol 839. Springer, Berlin, Heidelberg, pp. 150-164, 1994.

[26] J. He and E. Dawson, "Multi-secret sharing scheme based on one-way function," *Electronics Letters,* vol. 31, no. 2, pp. 93-95, 1995.

[27] L. Harn, "Efficient sharing (broadcasting) of multiple secrets," *IEEE Proceedings - Computers and Digital Techniques,* vol. 142, no. 3, pp. 237-240, 1995.

[28] C. Asmuth and J. Bloom, "A modular approach to key safeguarding," *IEEE Trans. Inform. Theory,* vol. 29, no. 2, pp. 208-210, 1983.

[29] E. F. Brickell, "Some ideal secret sharing schemes," in Proc. EUROCRYPT '89, Quisquater JJ., Vandewalle J. (eds). Lecture Notes in Computer Science, vol 434. Springer, Berlin, Heidelberg, pp. 468-475, 1990.

[30] J. Pieprzyk and X. M. Zhang, "Ideal threshold schemes from MDS codes," *Discrete Mathematics and Theoretical Computer Science,* vol. 6, no. 2, pp. 471-482, 2004.

[31] O. Koji and K. Kaoru, "MDS secret-sharing scheme secure against cheaters," *IEEE Trans. Inform. Theory,* vol. 46, no. 3, pp. 1078-1081, 2000.

[32] C. Ding, D. R. Kohel and S. Ling, "Secret-sharing with a class of ternary codes," *Theor. Comput. Sci.,* vol. 246, no. 1-2, pp. 285-298, 2000.

[33] K. Ehud, J. Greene and M. Hellman, "On secret sharing systems," *IEEE Trans. Inform. Theory,* vol. 29, no. 1, pp. 35-41, 1983.

[34] A. Renvall and C. Ding, "The access structure of some secret-sharing schemes," in Proc. ACISP 1996, Pieprzyk J., Seberry J. (eds). Lecture Notes in Computer Science, vol 1172. Springer, Berlin, Heidelberg, pp. 67-78,1996.

[35] A. Ashikhmin and A. Barg, "Minimal vectors in linear codes," *IEEE Trans. Inform. Theory,* vol. 44, no. 5, pp. 2010-2017, 1998.

[36] D. Chen, W. Lu, W. Xing and N. Wang, "An efficient verifiable threshold multi-Secret sharing scheme with different stages," *IEEE Access,* vol. 7, no. 1, pp. 107104-107110, 2019.

[37] O. Ersoy, T. B. Pedersen, K. Kaya, A. A. Selçuk and E. Anarim, "A CRT-based verifiable secret sharing scheme secure against unbounded adversaries," *Security and Communication Networks,* vol. 9, no. 17, pp. 4416-4427, 2016.

[38] P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," in Proc. SFCS'87, Los Angeles CA USA, pp. 427-438, 1987.

[39] J. Zhao, J. Zhang and R. Zhao, "A practical verifiable multi-secret sharing scheme," *Computer Standards and Interfaces,* vol. 29, no. 1, pp. 138-141, 2007.

[40] H. Pilaram and T. Eghlidos, "An efficient lattice based multi-stage secret sharing scheme," *IEEE Transactions on Dependable and Secure Computing,* vol. 14, no. 1, pp. 2-8, 2015.

[41] A. N. Amroudi, A. Zaghain and M. Sajadieh, "A verifiable (k, n, m)-threshold multi-secret sharing scheme based on NTRU cryptosystem," *Wireless Personal Communications,* vol. 96, no. 1, pp. 1393-1405, 2017.

[42] B. Rajabi and Z. Eslami, "A verifiable threshold secret sharing scheme based on lattices," *Information Sciences,* vol. 501, no. 1, pp. 655-661, 2019.

[43] R. Lidl and H. Niederreiter, Finite Fields, Cambridge: Cambridge University Press, 1997.

[44] Z. Heng, C. Ding and Z. Zhou, "Minimal linear codes over finite fields," *Finite Fields th App,* vol. 54, pp. 176-196, 2018.

[45] F. MacWilliams and N. Sloane, The theory of error-correcting codes, Amsterdam: North-Holland, Seventh Impression, 1992.

[46] R. J. McEliece and D. V. Sarwate, "On sharing secrets and Reed-Solomon codes," *Communications of the ACM,* vol. 24, no. 9, pp. 583-584, 1981.

[47] J. L. Massey, "Minimal Codewords and Secret Sharing," in Proc. 6th Joint Swedish–Russian Workshop on Information Theory, Molle, Sweden, pp. 246–249, 1993.

[48] J. L. Massey, "Some applications of coding theory in cryptography," in Proc. the Fourth IMA Conference on Cryptography and Coding, England, pp. 33-47, 1995.

[49] S. Mesnager and A. Sınak, "Several Classes of Minimal Linear Codes With Few Weights From Weakly Regular Plateaued Functions," *IEEE Trans. Inform. Theory,* vol. 66, no. 4, pp. 2296-2310, 2020.

[50] S. Mesnager, F. Özbudak and A. Sınak, "Linear codes from weakly regular plateaued functions and their secret sharing schemes," *Des. Codes, Cryptogr. ,* vol. 87, no. 2-3, pp. 463-480, 2019.

[51] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics (SIAM),* vol. 8, no. 2, pp. 300-304, 1960.

[52] C. Ding and J. Yuan, "Covering and Secret Sharing with Linear Codes," in Proc. DMTCS'03, Calude C.S., Dinneen M.J., Vajnovszki V. (eds). Lecture Notes in Computer Science, vol 2731. Springer, Berlin, Heidelberg, pp. 11-25, 2003.

Bölüm 11

KAFES TABANLI GRUP İMZALAMA ŞEMALARININ ÖZELLİKLERİ VE DEĞERLENDİRİLMESİ

Meryem Soysaldı Şahin - Sedat Akleylek

Ayrık logaritma ve çarpanlarına ayırma problemlerinin zorluğuna dayanan klasik kriptosistemlerin yerine kullanılabilecek polinom zamanda çözülemeyen çok değişkenli polinomlar, özet, kod, kafes ve izojeni tabanlı sistemler oluşturulmaktadır. Bu bölümde grup imzalama kavramının ortaya atılmasından başlayarak literatürdeki kafes tabanlı grup imzalama şemalarına yer verilmektedir. Şemalar gruba yeni üyelerin eklenmesi veya gruptan üyelerin çıkarılmasına izin vermeleri bakımından sınıflandırılarak oluşturulan her bir grup tipinin işleyişi gösterilmektedir. Farklı grup tiplerindeki şemaların içermesi gereken algoritmalar ve güvenlik gereksinimleri tanımlanmaktadır. Literatürde önerilen şemalar incelenerek şemalar imza ve grup açık anahtarı boyutu, dayandırıldıkları zor problem, oluşturulurken kullanılan alt şemalar ve sağlamaları gereken güvenlik gereksinimleri gibi özelliklere göre karşılaştırılmaktadır.

11.1. GİRİŞ

Teknolojik gelişmeler, internetin yaygınlaşması ve mobil cihazların artması ile birlikte günümüzde her türlü işlem elektronik ortamda yapılabilir hale gelmiştir. Bu işlemleri; e-devlet işlemleri, bankacılık işlemleri, elektronik ticaret ve online eğitim şeklinde örneklendirebiliriz. Elektronik ortamda yapılan tüm işlemlerde iletişimin problemsiz bir şekilde kurulabilmesi, işlemlerin

gerçekleştirilmesi ve tarafların haklarının korunması için bilgi güvenliği kavramlarının sağlanması gerekmektedir. Bu nedenle, verilere üçüncü kişilerin erişmesinin engellenmesi için şifreleme sistemleri, veri içeriğinin değiştirilmemesi için özet fonksiyonları, doğru kişi ile haberleşmenin sağlandığından emin olmak ve yapılan işlemlerin sonradan reddedilmemesi için ise kimlik doğrulama ve imzalama sistemleri kullanılmaktadır. Örneğin; günlük hayatta kağıt üzerinde yaptığımız işlemler elektronik ortama aktarılırken ıslak imzanın yerini elektronik imza almaktadır [1].

İmzalama sistemleri tıpkı kağıt ortamında olduğu gibi belgeyi onaylayanın kimliğinin bilinmesi, belgenin sonradan değiştirilememesi, imza atılarak onaylandıktan sonra içeriğin reddedilememesi ve imza atanın elektronik ortamda gerçekleştirdiği işlemleri inkar edememesi amacıyla kullanılmaktadır. Güvenli bir şekilde elektronik imza elde etmek için imzalama şemaları oluşturulmaktadır. Günümüz imzalama şemaları ayrık logaritma, çarpanlarına ayırma problemleri gibi klasik bilgisayarlarda polinom zamanda çözülemeyen zor problemler üzerine kurulmaktadır. Bu şemaları kırmaya yönelik olarak yapılan saldırılar hesaplama gücü yetersizliğinden dolayı başarısız olmaktadır. Öte yandan, 1994 yılında önerilen bir algoritma ile çarpanlarına ayırma ve ayrık logaritma problemlerinin kuantum hesaplama gücüyle polinom zamanda çözülebileceği belirtilmiştir [2]. Bununla birlikte, büyük şirketler yüksek kübite sahip kuantum bilgisayarların oluşturulması için yürüttükleri çalışmaları açıklamışlardır [3], [4], [5]. Bu nedenle, kuantum bilgisayarların hesaplama gücü imzalama şemaları gibi açık anahtarlı şifreleme sistemleri için tehlike arz etmektedir. Kuantum bilgisayarlar ile klasik kriptosistemlerin kırılacak olması araştırmacıları kuantum saldırılarına karşı dirençli yeni sistemleri oluşturmaya yöneltmektedir. Amerikan Ulusal Standartlar ve Teknoloji Enstitüsü (NIST), kuantum bilgisayarlar ile kırılamayacak açık anahtarlı sistemlerin oluşturulması gerektiğini vurgulayarak proje başlatmıştır. Kuantum bilgisayarlar ve kuantum sonrası güvenilir kriptosistemler hakkında daha detaylı bilgi için Siber Güvenlik ve Savunma kitabının 2. serisi 4. bölümüne bakınız [6].

İmzalama sistemleri açık anahtarlı sistemler olduklarından bu sistemlerde kullanıcı, açık ve gizli anahtar olmak üzere bir çift anahtara sahiptir. İmzalama sürecinde kullanıcı göndermek istediği mesajın özetini alarak kendi gizli anahtarı ile şifreleyerek imzayı oluşturmaktadır. Karşı taraf gönderenin açık anahtarını kullanarak mesajın içerisindeki özet değerini elde etmektedir. Bu özet ile kendi hesapladığı mesaj özetini karşılaştırarak imzayı doğrulamak-

tadır. Buradan anlaşılacağı üzere klasik imzalama yapısında bir kişi mesajı imzalamakta ve mesajı alanlar imzalayanın kimliğini ve imzayı doğrulayabilmektedir. Bu temel mantık çerçevesinde oluşturulmuş farklı amaçlara yönelik imzalama şemaları mevcuttur: Grup (Group), Halka (Ring), Kör (Blind) vb.

1991 yılında Chaum ve van Heyst tarafından grup imzalama kavramı ortaya atılmıştır [7]. Bu imzalama tipinde belli sayıda üyeden oluşan bir grup oluşturulmaktadır. Grup üyelerinden herhangi biri, mesajı grup adına anonim olarak imzalarken diğer grup üyeleri atılan imzayı doğrulayabilmektedir. Öte yandan, gerekli olduğu durumlarda mesajı imzalayanın kimliği açığa çıkarılmaktadır. Chaum ve arkadaşları grup imzalamanın ilk örneklerini oluşturmuşlardır. Yaptıkları çalışmada dört farklı şema önermişlerdir. Önerilen ilk iki şemada, grup sabit tutulmaktadır. Diğer şemalarda ise önceden belirlenen üyeler ile grubun oluşturulduğu bir kurulum aşaması mevcuttur. Bu mimaride farklı üyeler seçilerek grubun oluşturulması mümkün değildir. Bunun yanında, şemaların tamamında grup oluşturulduktan sonra gruba yeni üyelerin eklenmesi de mümkün olmamaktadır. Şemaların üç tanesinde grup yöneticisinin imza atan üyenin kimliğini açığa çıkarması için geriye kalan tüm üyeler ile iletişime geçmesi gerekirken son şemada böyle bir işlemin yapılması gerekmemektedir. Bu nedenle, önerilen üç şema hesaplamalı olarak anonim olma özelliğini sağlarken son şema teorik olarak anonim olma özelliğini sağlamaktadır. Şemaların açık anahtarı, gruptaki üye sayısı ile doğru orantılı olarak artmaktadır. Ayrıca güvenlikleri büyük tam sayılar için çarpanlarına ayırma ve ayrık logaritma problemlerinin zorluğuna dayanmaktadır. Bir başka çalışmada, kurulum aşamasından sonra bile yeni üyelerin dinamik olarak eklenebildiği iki farklı grup imzalama şeması önerilmiştir [8]. Ayrık logaritma probleminin zorluğuna dayanan [7] nolu çalışmada, grup yöneticisinin mesajı imzalayan üyeyi bulmak için tüm grup üyeleriyle iletişim kurmasını gerektiren soruna çözüm önerisi getirilmiştir. Önerilen şemalarda grup yöneticisi grupta yer alan tüm üyelerin gizli anahtarını bilmektedir. Bu yaklaşımdaki temel problem, gerçekte imza atmayan bir üyenin grup yöneticisi tarafından mesajı imzalamakla suçlanmasıdır. Bu sorun [9]'da imza açma aşamasına kanıt değerinin eklenmesi ile çözülmüştür. Gruba yeni üyelerin dinamik olarak eklenmesine izin veren şema, imzanın oluşturulması için yapılan hesaplamalar bakımından [8]'de verilen şemalardan dört kat daha verimlidir. Öte yandan, açık anahtar boyutu ve imza boyutu [8] ve [9]'da verilen şemalardaki gibi gruptaki üye sayısıyla doğru orantılı olarak artmaktadır. Üye sayısı fazla

olan gruplar için istenmeyen bu durumun üstesinden gelmek için imza ve açık anahtar boyutu gruptaki üye sayısından bağımsız olan bir şema önerilmiştir [10]. Önerilen bu şemanın başka bir katkısı, grup açık anahtarında herhangi bir güncelleme ihtiyacı olmadan yeni üyelerin eklenmesine imkan vermesidir. Grup imzalama şemalarını imza boyutu, anahtar boyutları ve imzalama - imzanın açılması aşamalarında geçen süreler bakımından daha verimli hale getirebilmek için farklı varsayımlara dayanan çalışmalar yapılmıştır. Örneğin; RSA problemine dayanan [11], [12], [13] veya grupların ikili doğrusal haritalama (bilinear map) ile ilişkilendirildiği [14], [15], [16] şemalar önerilmiştir. Kitabın bu bölümünde kafes tabanlı grup imzalama şemalarına yer verileceğinden geleneksel yöntemlere dayanan şemalara değinilmeyecektir.

Grup imzalama şemaları, imza atan üyelerin gerçek kimliklerinin grup yöneticisi tarafından açığa çıkarılmadığı müddetçe gizli kalması gerektiği ve atılan imzaların gruptaki diğer üyeler tarafından doğrulanabildiği senaryolarda uygulama alanı bulmaktadır [17]. Grup imzalama şemaları yaygın olarak kurum yapılarının gizlenmesi gereken uygulamalarda kullanılmaktadır: Örneğin; şirket çalışanları şirket adına sözleşmeleri imzalayabilir, basın açıklaması yayınlayabilir, kamu ihalelerine katılabilir veya finansal işlemleri yürütebilir. Çalışanlar bu işlemleri yürütürken grup imzalarını kullanırsa çalışanların kimlikleri gizli kalacaktır. Öte yandan, eğer bir çalışan şirketin itibarına zarar verecek bir işlemde bulunursa şirket yöneticisi tarafından kimliği tespit edilerek imza hakları iptal edilmektedir. Grup imzalama şemaları çeşitli e-ticaret uygulamalarında da kullanılmaktadır. Örneğin; elektronik para ile işlem yapan kullanıcıların gizliliğini korumayı amaçlayan e-para uygulamalarında ve müzayedeye katılanların gizliliğini korumak için dijital müzayedeler veya seçmenlerin isimsiz olarak oy kullanmaları gereken dijital oylama gibi açık artırma uygulamalarında kullanılmaktadır [17], [18]. Bunların yanı sıra, anonim olarak online haberleşmede, güvenilir bilgi işlem platformları, gizlilik koruma mekanizmaları ve dijital hakların yönetiminde de kullanılmaktadır.

11.1.1. Organizasyon

Çalışmanın geri kalan kısmı şu şekilde organize edilmiştir. Bölüm 11.2'de grup imzalama şemaları incelenmiştir. Şemalar dört farklı gruba ayrılarak her bir grubun genel işleyiş yapısı verilmiştir. Grup imzalama şemalarında kul-

lanılan algoritmalar tanımlanarak şemaların güvenlik gereksinimlerine yer verilmiştir. Literatürde önerilen grup imzalama şemaları detaylı olarak ele alınmıştır. Bölüm 11.3'te grup imzalama şemalarının kullanım senaryolarına örnekler verilmiştir. Bunun yanında, Bölüm 11.2'de yer alan şemalar dayandıkları zor problem, hangi alt sistemleri içerdikleri, güvenlik gereksinimleri ile açık anahtar ve imza boyutları gibi kriterlere göre karşılaştırılmıştır. Bölüm 11.4'te ise sonuç ve değerlendirmelere yer verilmiştir.

11.2. KAFES TABANLI GRUP İMZALAMA ŞEMALARININ İNCELENMESİ

Bu bölümde, literatürdeki kafes tabanlı grup imzalama şemalarını inceleyeceğiz. Şemalar incelendiğinde farklı kafes problemlerinin zorluk varsayımları altında oluşturuldukları görülmektedir. Kitabın bu bölümünde şemalarda kullanılan zor kafes problemlerinin tanımına yer verilmeyecektir. Kafes problemleri hakkında detaylı bilgi için bu kitap serisinin 2. kitabının 5. bölümüne bakınız [19]. Bu bölümde literatürdeki kafes problemlerine dayanan grup imzalama şemaları, gruptaki üye sayısı bakımından statik, dinamik, gruba üye eklenmesine izin vermeyen ancak gruptan üyelerin çıkarılabildiği doğrulayıcı yerel iptal (Verifier-Local Revocation-VLR) mekanizmasına sahip olanlar şeklinde sınıflandırılmaktadır. Bu şemaları elde edebilmek için kullanılan algoritmalar verilerek şemaların güvenlik gereksinimleri tanımlanmıştır. Bu bölümde kullanılan kısaltma ve semboller anlamlarıyla birlikte Tablo 11.1'de verilmiştir.

Tablo 11.1. Kullanılan Kısaltmalar-Semboller ve Anlamları

Kısaltma/Sembol	Anlamı
NIST	Amerikan Ulusal Standartlar ve Teknoloji Enstitüsü (National Institute of Standards and Technology)
RLWE	Halkada hatalar ile öğrenme problemi (Ring Learning With Error)
RSIS	Halkada kısa tamsayı çözüm problemi (Ring Short Integer Solution)
SIVP	En kısa bağımsız vektör problemi (Shortest Independent Vector)
PKE	Açık anahtar şifreleme (Public Key Encryption)

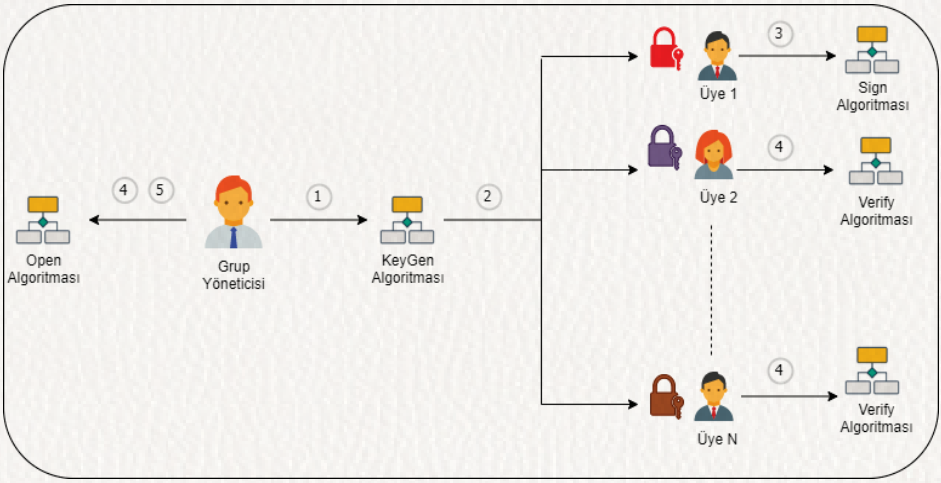
GSS	Grup imzalama şeması (Group Signature Scheme)
VLR	Doğrulayıcı yerel iptali (Verifier-Local Revocation)
NIZK	Etkileşimsiz sıfır bilgi paylaşımlı şema (Non-Interactive Zero Knowledge)
GM	Grup yöneticisi (Group Manager)
TM	İzleme yöneticisi (Tracing Manager)
Issuer	Gruba yeni kullanıcı ekleyen yönetici
Opener	Grupta imza atan üyenin kimliğini açığa çıkaran yönetici
KeyGen, GKgen	Anahtar üretim algoritmaları
GSetup	Grup kurulum algoritması
Join,Iss	Yeni grup üyesine sertifika oluşturulması ve gruba eklenmesi için etkileşimli algoritmalar
Sign	İmzalama algoritması
Verify	İmzayı doğrulama algoritması
Open, Trace	İmza atan üyeyi açığa çıkarma algoritması
Revoke	Üyelik iptal algoritması
Judge	Üye-imza doğrulama algoritması
RL	İptal listesi (Revocation List)
SM	Standart model (Standard Model)
ROM	Rastgele kahin modeli (Random Oracle Model)
N	Maksimum grup üyesi sayısı
gpk	Grup açık anahtarı
gsk	Grup üyelerinin gizli anahtarı
$gmsk$	Grup yöneticisinin gizli anahtarı
(upk, usk)	Gruba katılacak kullanıcıların açık ve gizli anahtarları
(mpk, msk)	Grup yöneticisinin açık ve gizli anahtarı
(tpk, tsk)	İmza açma yöneticisinin açık ve gizli anahtarı
(ik, ok)	Issuer ve Opener adlı yöneticilerin gizli anahtarları
grt	Grup üyelerinin iptal kimlikleri
pp	Grup açık parametreleri
Σ	İmza
M	Mesaj
S	Grup üyelerinin açık anahtarının kümesi
reg	Kayıt listesi
$info$	Güncel grup bilgileri

11.2.1. Statik Grup İmzalama Şemaları

Grup imzalama şeması (GSS), en temel olarak grup yöneticisi ve grup üyelerinden oluşmaktadır. Grup yöneticisi, grubu oluşturacak olan üyelere ait anahtarları ürettikten sonra bu anahtarları üyelere dağıtmaktadır. Bu şekilde gruba dahil olan grup üyeleri, mesajları grup adına anonim olarak imzalayabilmektedir. Oluşturulan imza gruptaki diğer üyeler tarafından doğrulanırken imzanın kim tarafından atıldığı bilinmemektedir. Gerekli olduğu durumlarda grup yöneticisi imzayı atan üyenin kimliğini açığa çıkarmaktadır. GSS'lerin ilk örneklerinde gruba üye eklenmesine veya gruptan üye çıkarılmasına izin verilmemektedir. Grubu oluşturulacak üyeler, grubun kurulum aşamasında belli ve sabit olduğundan bu tip imzalama şemaları statik olarak isimlendirilmektedir.

Şekil 11.1'de verilen eşit imzalama hakkına sahip N tane üyeden oluşan statik bir GSS şu şekilde çalışmaktadır:

- 1 numaralı adımda grup yöneticisi KeyGen algoritmasını kullanarak grup açık anahtarını ve gruptaki üyeler için bir gizli anahtar üretmektedir.
- 2 numaralı adımda her grup üyesine özgü olacak şekilde üretilen anahtarlar üyelerine dağıtılmaktadır. Grup üyeleri kendilerine ait olan anahtar çiftlerini aldıktan sonra grup adına imza atma yetkisine sahip olmaktadır.
- 3 numaralı adımda grup üyelerinden birisi olan Üye 1 imzalama algoritmasını kullanarak mesajı anonim bir şekilde imzalamaktadır.
- Bir sonraki adımda Üye 2 imzalanan mesajı alarak doğrulama algoritmasını kullanarak imzayı oluşturanın bir grup üyesi olduğunu ve imzanın geçerli olduğunu doğrulamaktadır. Öte yandan, gerekli olduğu durumda 4 numaralı adımdaki gibi grup üyeleri atılan imzayı doğrulamadan önce grup yöneticisi imza atan üyenin kimliğini açığa çıkarabilmektedir.



Şekil 11.1. Statik Grup İmzalama Şemasının İşleyişi

Şekil 11.1'de çalışma şekli verilen statik grup imzalama şeması temelde dört tane algoritmadan oluşmaktadır:

- KeyGen(1^{λ},1^{N}): $\lambda \in \mathbb{N}$ güvenlik parametresi ve $N \in \mathbb{N}$ gruptaki üye sayısını göstermek üzere 1^{λ} ve 1^{N} parametrelerini girdi olarak alan algoritma $(gpk, gmsk, gsk)$ değerlerini üretmektedir. gpk grup açık anahtarını, $gsk = \{gsk[i] \,|i \in \{0, \dots, N-1\}\}$ N üyeye sahip olan grubun i. üyesine ait gizli anahtarı ve $gmsk$ grup yöneticisinin gizli anahtarını ifade etmektedir.
- Sign($gpk, gsk[d], M$): Bu algoritma gizli anahtarı $gsk[i]$ ve mesajı M girdi olarak alıp imzayı Σ üretmektedir.
- Verify(gpk, M, Σ): İmzanın doğrulanması aşamasında grup açık anahtarı gpk, mesaj M ve bu mesaja karşılık üretilmiş olan imza değeri Σ alınarak imza doğrulanmaktadır. İmza doğrulanırsa 1, doğrulanmazsa 0 değeri döndürülür.
- Open($gmsk, M, \Sigma$): Grup yöneticisinin gizli anahtarını $gmsk$, mesajı M ve imzayı Σ alan bu algoritma sonlandığında imzayı atan üyenin indisini $i \in \{0, \dots, N-1\}$ döndürmektedir. İmzayı atan üye grupta yer almıyorsa $\perp$ değeri döndürülmektedir.

Yukarıda verilen algoritmaları içeren bir GSS, genellikle bir imzalama şeması, şifreleme şeması ve etkileşimsiz olan sıfır bilgi paylaşımlı (non-interactive zero knowledge proof system-NIZK) kimlik doğrulama şemasının birleştirilmesi ile elde edilmektedir [20].

11.2.1.1. Güvenlik Gereksinimleri

Camenisch ve Stadler, oluşturulan statik GSS'lerin güvenliğinin ispatlanması için aşağıda verilen özelliklerin sağlanması gerektiğini belirtmiştir [10]:

- **Taklit edilememezlik (Unforgeability):** Bu özelliği sağlayan bir şemada mesajları sadece grup üyeleri imzalayabilmektedir.
- **Anonimlik (Anonymity):** Mesajı imzalayan üyenin kimliği gizli olmalıdır.
- **Bağlantısızlık (Unlinkability):** Herhangi iki imzanın aynı üyeye ait olup olmadığı hakkında bilgi sahibi olunamaz.
- **Çerçeveleme saldırılarına karşı güvenlik (Security against framing attacks):** Grup üyeleri hatta grup yöneticisi bile imzanın doğrulanmasını engelleyememelidir.

Bellare ve arkadaşları ise statik GSS'lerin güvenliği için tam anonimlik ve tam izlenebilirlik kavramlarını tanımlamıştır [20].

Tam Anonimlik (Full Anonymity): Mesaj ve imza çifti verildiğinde imzayı atan üyeyi bulabilen imza açma kahinine erişimi olan bir saldırganın olduğunu farz edelim. Böyle bir saldırgan bile grup yöneticisinin gizli anahtarı olmadan sadece imzaya bakarak imzayı atan üyenin kim olduğunu anlayamıyorsa şema tam anonimlik özelliğini sağlamaktadır. Başka bir ifadeyle, saldırgan iki grup üyesinin gizli anahtarlarını gsk_{j_0}, gsk_{j_1} bilse bile bu iki farklı j_0 ve j_1 grup üyesine ait olan imzaları birbirinden ayırt edememektedir.

Tam İzlenebilirlik (Full Traceability): Kötü niyetli üyelerin hiçbiri, grup yöneticisinin gizli anahtarını bilse bile gruptaki üyeler tarafından izlenemeyen geçerli bir imza üretememelidir. Başka bir ifadeyle, herhangi bir anlaşmazlık veya olumsuzluk durumunda üyeliğini kötüye kullanan üyenin kimliğini açığa çıkaracak bir izleme mekanizmasının olması gerekmektedir.

Oluşturulan grup imzalama şemalarının düzgün bir şekilde çalıştığının gösterilebilmesi için tamlık özelliğini sağladığı kanıtlanmalıdır. Bu özellik statik veya dinamik olan tüm grup imzalama şemaları için sağlanmalıdır.

Tamlık (Correctness): Bu özellik, grup üyelerinin (üyeliği iptal edilmemiş) ürettiği her imzanın geçerli olduğunu ve anahtar üretme algoritması ile üretilen parametrelere göre doğrulanabildiğini göstermektedir. Dinamik yapıdaki şemalarda Open algoritmasına mesaj ve imza çifti verildiğinde imza atan üyenin kimliğine ulaşılmalıdır. Bunun yanında, Open algoritması tarafından döndürülen kanıt, Judge algoritması tarafından kabul edilmelidir.

11.2.1.2. Önceki Çalışmalar

Çarpanlarına ayırma ve ayrık logaritma problemlerinin zorluğuyla ilgili varsayımlara dayanan sistemlerin yerine kuantum saldırılarına dirençli olan sistemlerin oluşturulması gerektiği yönünde başlayan akım ile kafes tabanlı grup imzalama sistemleri üzerinde çalışmalar artmıştır. Kafes tabanlı sistemlerin tercih edilme nedenleri arasında kafesler üzerinde işlem yapmanın kolay olması, paralelleştirilebilmesi, asimptotik açıdan verimli ve kafes problemleri arasında en kötü durum / orta durum bağlantılarının olması gibi sebepler yer almaktadır. Bu bölümde literatürdeki kafes tabanlı statik grup imzalama şemalarına yer verilmektedir. İncelenen şemalar dayandıkları zor problem ve sağladıkları güvenlik gereksinimleri bakımından Tablo 11.3'te, şemalarda kullanılan alt bileşenler ve imza ve anahtar boyutları bakımından ise Tablo 11.4'te karşılaştırılmaktadır.

Kafes tabanlı ilk grup imzalama şeması 2010 yılında oluşturulmuştur [21]. Grup imzalama şemasının oluşturulmasında [20] nolu çalışmada verilen grup imzaların temel yapısı benimsenerek statik GSS'ler için yukarıda bahsi geçen algoritmalar oluşturulmuştur. Çalışmada, GPV imzalama şeması [22] kullanılarak imzalama algoritmasında üretilen imza Regev şifreleme sistemi [23] ile şifrelenmiştir. Oluşturulan yapı Micciancio ve arkadaşlarının sıfır bilgi paylaşımlı şemasının [24] temel mantığı altında yürütülerek kafes tabanlı grup imzalama şeması elde edilmiştir. LWE probleminin çözümünün zor olması varsayımı altında rastgele kahin modelinde (ROM) şemanın anonimlik ve izlenebilirlik özelliklerini sağladığı gösterilmiştir. Gruptaki üye sayısının sabit olduğu bu şemada grup yöneticisinin anahtarları ve imza boyutu gruptaki üye sayısı ile doğrusal olarak artmaktadır. GSS'lerde kullanılan şifreleme, kimlik

doğrulama ve imzalama şemalarının hangi algoritmalarda nasıl kullanıldığını göstermek için bu şemanın işleyişi şu şekilde açıklanabilir:

- Anahtar üretim algoritmasında GPV imzalama ve Regev şifreleme şemalarında kullanılacak parametreler ile grup açık anahtarı, imzayı açma anahtarı ile kullanıcıların imzalama anahtarları üretilmektedir.
- İmzalama algoritmasında mesajı imzalayacak olan kullanıcı kendi gizli anahtarı ile GPV imzalama şemasında olduğu gibi mesajı imzalamaktadır. Oluşturulan imza Regev şifreleme yöntemi ile şifrelendikten sonra sıfır bilgi paylaşımlı şemadan elde edilen kanıt değerleriyle birleştirilerek imza elde edilmektedir.
- İmzanın doğrulanması aşamasında GPV imzalama şemasındaki imza doğrulama işlemleri yerine getirilmektedir. İmza açılarak elde edilen değerler ile açık anahtar ve mesaj kullanılarak hesaplanan değerlerin birbirine eşit olup olmadığı kontrol edilerek imza doğrulanmaktadır.
- İmzanın açılması algoritmasında ise imza, mesaj ve imza açma anahtarı şifreleme şemasının doğrulaması için kullanılarak mesajı imzalayan üyenin kimliği açığa çıkarılmaktadır.

2012 yılında yapılan bir çalışmada grup imzalama şemalarının genelleştirilmiş bir versiyonu olarak adlandırılan anonim özellik belirteç (anonymous attribute token) şeması oluşturulmuştur [25]. Bunun yanında, [21]'de verilen kafes tabanlı ilk grup imzalama şeması tam anonimlik özelliğini sağlayacak şekilde iyileştirilmiştir. [21]'de verilen şemada mesajı imzalayan bir üyenin kimliği açığa çıkarıldıktan sonra üyeliğini kötüye kullanmamış veya imza attığından dolayı daha önce kimliği açığa çıkarılmamış bir üye olsa bile tüm üyeler ifşa olmaktadır. Bu nedenle [21]'de verilen şema imzaların hiç açılmayacağı durumlarda kullanışlıdır ve anonimlik özelliğini sağlamaktadır. [25] nolu çalışmada bu sorun tam anonimlik özelliğiyle çözülmüştür. Önerilen şema, mesaj ve imza verildiğinde kimin imza attığını bulabilen bir kahine sorgu gönderebilen bir saldırganın var olması durumunda bile dürüst kullanıcıların anonimliğini tamamen sağlamaktadır. Ayrıca [21]'de verilen şema ile karşılaştırıldığında çerçevesizlik (non-frameability) özelliğine sahiptir. Bu özellik sayesinde sahtekar bir grup yöneticisinin grup üyeleri adına imza atması engellenmektedir. Şemanın en önemli katkısı, tam anonimlik ve çerçevesizlik özelliklerine sahip kafes tabanlı ilk grup imzalama şeması olmasıdır. Bunun yanı sıra, [21]'de gruptaki üye sayısı ile doğrusal olarak artan grup

yöneticisinin anahtar çiftleri iyileştirilerek bu şemada gruptaki üye sayısından bağımsız olarak üretilmektedir. Öte yandan, şemanın imza boyutu gruptaki üye sayısı ile doğrusal olarak artmaktadır. Üye sayısı fazla olan gruplar açısından düşünüldüğünde imza boyutu ve açık anahtar boyutu oldukça büyük olmaktadır. Bu soruna çözüm olarak imza ve açık anahtar boyutu gruptaki üye sayısı ile logaritmik olarak artan bir şema önerilmiştir [26]. Bu sayede kafes tabanlı grup imzalama şemaları büyük gruplarda kullanabilir hale gelmiştir. [21]'deki şemada kullanılan GPV imzalama şeması yerine Boyen imzalama şemasını [27] kullanarak imza boyutunun $\tilde{\mathcal{O}}(N)$ ile doğrusal artması engellenmiştir. Bilindiği üzere Boyen imzalama şeması standart modelde güvenli olduğu ispatlanmış kafes tabanlı imzalama şemalarından biridir [27]. Şema tasarlanırken Boyen imzalama şeması kullanılmasının rastgele kahine erişimi olmayan bir saldırgana karşı şemanın güvenliğini ispatlamayı kolaylaştırdığı belirtilmiştir. [21]'deki şema ile karşılaştırıldığında tek fark imzalama şeması değildir. Bu şemada sıfır bilgi paylaşımlı şema olarak Lyubashevsky tarafından önerilen [28] nolu çalışmadaki kimlik doğrulama şeması kullanılırken şifreleme için Regev şifreleme şeması [22] kullanılmıştır. LWE ve SIS problemlerinin zorluk varsayımlarına dayanan şema [21]'deki şemada olduğu gibi rastgele bir kahine erişimi olmayan bir saldırgan için anonim olma özelliğini sağlamaktadır. Öte yandan tam anonimlik özelliğini sağlayan bir versiyonu da oluşturulmuştur.

Kafes tabanlı grup imzalama şemalarının oluşturulması konusunda açık problemler mevcuttur. Bunlardan bazıları;

- Bellare ve arkadaşları tarafından [20] nolu çalışmada tanımlanan grup imzalama yapısına dayanan imza boyutu $\tilde{\mathcal{O}}(\lambda\, logN)$ olan ve daha zayıf zorluk varsayımına dayanan grup imzalama şeması oluşturulabilir mi?
- SIS ve LWE problemlerinin halka versiyonlarına dayanan grup imzalama şeması oluşturulabilir mi [26]?

şeklindedir. Bu açık problemlere çözüm önerisi getirilmiştir [29]. Açık anahtar boyutu $\tilde{\mathcal{O}}(\lambda^2\, logN)$ ve imza boyutu $\tilde{\mathcal{O}}(\lambda\, logN)$ olan bir şema oluşturulmuştur. [26]'daki şema ile karşılaştırıldığında anahtar boyutları 4 kat azalmış ve imza boyutu da daha küçüktür. Bunun yanında elde edilen bu şema yukarıda verilen açık probleme çözüm olarak halka yapısına dönüştürülmüştür. Sonuç olarak $\tilde{\mathcal{O}}(\lambda\, logN)$ açık anahtar ve imza boyutuna sahip halka tabanlı bir grup imza-

lama şeması da önerilmiştir. Şemalarda [26]'da olduğu gibi Boyen imzalama şeması [27] kullanılmıştır. Öte yandan kafes tabanlı grup imzalama şemasında şifreleme şeması olarak [22]'de verilen şema kullanılırken halka tabanlı şemada ise [30]'da verilen LPR şeması kullanılmıştır. NIZK şeması olarak ise Stern tarzında [31] nolu çalışmada verilen şema kullanılarak Fiat-Shamir dönüşümü ile grup imzalama şemaları elde edilmiştir. Statik GSS'lerin güvenlik gereksinimi olan anonimlik ve izlenebilirlik özelliği sırasıyla şemanın dayandığı şifreleme ve imzalama şemalarının zorluk varsayımlarına dayandırılarak ROM'da ispatlanmıştır.

Kafes tabanlı statik grup imzalama şemalarını daha verimli hale getirmek için yapılan başka bir çalışmada [21], [26], [29] ve [32]'de verilen şemalardan açık anahtar ve imza boyutu bakımından verimli olan daha basit bir grup imzalama şeması önerilmiştir [33]. Gruptaki üyelerin kimliklerinin anonimliğini sağlayabilmek için kimlik tabanlı şifreleme (identity-based encrypriton-IBE) kullanılarak [34] nolu çalışmada verilen kodlama (encoding) tekniği temel alınmıştır. Kullanılan bu kodlama tekniğinin sağladığı avantajlar şu şekilde sıralanabilir:

1) Kısa grup açık anahtarı elde etmede kullanılan bu kodlama fonksiyonları için sadece üç matris kullanılmaktadır. Oysa [21] ve [26]'da verilen şemalarda sırasıyla bu işlem için N ve $log\ N$ tane matrise ihtiyaç vardır.

2) NIZK bir sistemin oluşturulması için daha basit bir üyelik ilişkisi sağlamaktadır. [21] ve [26]'da verilen şemalarda NIZK sistemlerin elde edilmesi için daha karmaşık üyelik ilişkilerine ve ilave şifreleme sistemlerine ihtiyaç duyulmaktadır. Bunun sonucu olarak [21] ve [26]'daki şemalar yüksek hesaplama maliyetine ve büyük imza boyutuna sahiptir. Şema tasarlanırken kullanılan kodlama tekniğinin yanı sıra [28], [35] ve [36] nolu çalışmalardaki mantık birleştirilerek LWE ve SIS problemlerine dayanan sıfır bilgi paylaşımlı etkileşimli bir sistem elde edilmiştir. İmzalama şeması olarak Lyubashevsky'nin imzalama [35] mantığı temel alınırken şifreleme şeması olarak yine Regev'in şifreleme şeması [23] kullanılmıştır. ROM'da tam anonimlik ve izlenebilirlik ispatları verilen bu şemanın açık anahtar ve imza boyutları sırasıyla $\tilde{O}(\lambda^2\ log^2 N)$ ve $\tilde{O}(\lambda +\ log^2 N)$ şeklinde verilmektedir.

Dağıtık protokollerin tasarımında dijital imzalara alternatif olan akümülatör yapısı kullanılarak [37] nolu çalışmada, grup üyeliklerini sıfır bilgi esasına göre gizleyen, kafes tabanlı bir akümülatör oluşturulmuştur. Şemada SIS problemine dayanan bir özet fonksiyon Merkle ağaç yapısında [38] kullanılarak gizli bilginin yapraklarda saklanması ve bilginin yapraklardan köke kadar özet zincir şeklinde sıfır bilgi paylaşımlı olarak tutulması sağlanmıştır. Elde edilen şema yukarıda bahsi geçen şemalardan daha zayıf zorluk varsayımına, daha küçük açık anahtar ve imza boyutuna sahiptir. İmza boyutu $\tilde{O}(\lambda\, logN)$'dir. Bunun yanında, [26], [29], [32] ve [33]'te verilen şemaların aksine herhangi bir GPV imzalama yapısının [22] kullanılmadığı ilk kafes tabanlı grup imzalama şemasıdır. Bu özellik parametrelerin daha verimli bir şekilde seçilmesini sağlamaktadır. Stern tarzında [39] bir NIZK şeması ve Regev şifreleme şeması, Naor-Yung çift şifreleme yaklaşımı [40] ile birleştirilerek grup imzalama şeması elde edilmiştir. Şemanın güvenliği LWE ve SIS problemlerinin zorluğuna dayandırılarak şemanın tam anonimlik ve izlenebilirlik özelliklerini sağladığı ispatlanmıştır.

Statik imzalama şemalarında aynı mesajın birden çok üye tarafından imzalanması durumunda daha verimli çalışacak mesaja bağlı olarak imzaların açıldığı bir grup imzalama şeması (group signature scheme with message-dependent opening / GSS-MDO) oluşturulmuştur [41]. GSS-MDO şemalarında hiçbir yetkili tek başına imza atan üyelere ulaşamamaktadır. İmza açma yetkisi imzaları açıcı (opener) ve alıcı (admitter) olmak üzere iki farklı yöneticiye verilmiştir. Bir M mesajını imzalayan grup üyesini bulabilmek için ilk olarak alıcı yetkilisinin ilgili mesaj için bir tuzak (trapdoor) t_M oluşturması gerekmektedir. Daha sonra imza açma yetkilisi t_M 'yi kullanarak kendi gizli anahtarı ile imza atan üyenin kimliğini açığa çıkarmaktadır. Bu şekilde imza atan üyelerin kimliklerinin mesaja bağlı olarak gerçekleştirildiği bu şema GSS-MDO şemalarının kafes tabanlı ilk örneğidir. GSS-MDO şemaları kimlik tabanlı şifreleme ile eşleştirilmektedir. Şema [29] nolu çalışma verilen şema üzerine inşa edilmiştir. Bu nedenle şemada grup üyelerinin gizli anahtarını üretmek için kimlik tabanlı şifrelemede kullanılabilen ve anahtar boyutu küçük olan Boyen imzası [27] kullanılmıştır. Regev'in şifreleme şeması [22] ile üyelerin gizli anahtarları şifrelenerek imzalar oluşturulmuştur. Bu yapılar imza atan üyenin gizlenmesi ve imzanın doğrulanabilmesi için Stern tarzı sıfır bilgi paylaşımlı bir

şema [42] ile birleştirilmiştir. SIS ve LWE varsayımlarının zorluğu altında imza açma ve alıcı yetkilileri açısından şemanın ROM'da anonim ve izlenebilir olduğu ispatlanmıştır.

Grup imzalama şemalarında üyeler mesajları grup adına anonim olarak imzalarken gerekli olduğu durumlarda imza atan üyenin kimliği açığa çıkarılmaktadır. Bununla birlikte, literatürde önerilen şemaların çoğu üyenin kimliği ve gizli anahtarı açığa çıkarıldıktan sonra güvenliği garanti etmemektedir. Bu nedenle, gizli anahtarlar açığa çıkarıldıktan sonra bile saldırganın geçmiş dönemlerde atılan imzaları taklit etmesini veya diğer üyelerin gizli anahtarlarını elde etmesini engelleyen gereksinim olarak tanımlanan ileri güvenliği (forward secure) sağlayan kafes tabanlı bir şema önerilmiştir [43]. İleri güvenliği sağlamak amacıyla gizli anahtar için Bonsai ağaç yapısı kullanılmıştır. Bununla birlikte, sıfır bilgi paylaşımlı bir şema elde edebilmek için [32]'deki şema temel alınmıştır. İleri güvenliği sağlayan bu şemanın ROM'da tam anonimlik özelliğini sağladığı kanıtlanmıştır. Şemaya eklenen ileri güvenlik özelliğinden dolayı şemanın ileri güvenlikli izlenebilir (forward-secure traceability) olduğu ispatlanmıştır. Bu güvenlik gereksinimi ile saldırganın, imza açma yetkilisinin veya bazı grup üyelerinin gizli anahtarlarını ele geçirmesi durumunda bile gizli anahtarlarını bildiği veya bilmediği grup üyelerine ait olacak geçerli imzaları üretebilmesi engellenmiştir.

İmza boyutu gruptaki üye sayısına bağlı olmayan ROM'da güvenlik ispatları verilen bir şema önerilmiştir [44]. Şema oluşturulurken daha basit imzalama algoritmasına ve daha küçük imza boyutuna sahip olduğundan Lyubashevsky'nin imzalama şeması kullanılmıştır. Klasik grup imzalamalarının aksine bu şemanın sıfır bilgi paylaşımlı bir alt bileşeni yoktur. RSIS ve RLWE problemlerine dayanan şemanın tam anonimlik ve tam izlenebilirlik özelliklerini sağladığı ispatlanmıştır.

ROM'da güvenli olduğu ispatlanan çalışmaların yanı sıra standart modelde (SM) güvenli olan kafes tabanlı grup imzalama şemaları da oluşturulmuştur [45]. Kafes tabanlı grup imzalama şemalarının oluşturulmasında en çok tercih edilen LWE ve SIS problemlerinin zorluk varsayımı altında SM'de güvenlik ispatları verilen grup imzalama şemalarının ilk örneklerinden olan iki farklı grup imzalama şeması önerilmiştir. Birinci şemanın

parametreleri ve imza boyutu gruptaki üye sayısı ile doğrusal olarak artarken ikinci şemanın boyutları gruptaki üye sayısından bağımsızdır. ROM'da güvenli olan statik grup imzalama şemalarından farklı olarak sıfır bilgi paylaşımlı sistem yerine özellik tabanlı imzalama şeması (attribute-based signature-ABS) [46] kullanılmıştır. Bu nedenle şemaların güvenliği ROM yerine SM'de yapılabilmiştir. SIS varsayımı altında güçlü taklit edilememezlik özelliğine sahip tek kullanımlık bir imzalama şeması [47], LWE varsayımının zorluğuna dayanan seçilmiş mesaj saldırılarına karşı dirençli ve anonim olan bir gizli anahtar şifreleme (secret key encryption-SKE) şeması [23] ile mükemmel gizliliği (perfect privacy) ve taklit edilememezliği sağlayan ABS şeması [46] olmak üzere 3 farklı yapı birleştirilerek iki farklı grup imzalama şeması elde edilmiştir. Sonuç olarak şemaları oluşturulan alt şemaların güvenliği dikkate alınarak şemaların kendi kendine anonim ve izlenebilir olduğu ispatlanmıştır.

Standart modelde sıfır bilgi paylaşımlı bir şema içermeyen bir grup imzalama şeması daha önerilmiştir [48]. [45]'te verilen sıfır bilgi paylaşımlı bir sistem kullanmadan ABS şeması ile grup imzalama şeması oluşturma fikrinden yola çıkılarak standart modelde kafes tabanlı ileri güvenliği sağlayan bir şema oluşturulmuştur. İleri güvenliği sağlayan ABS şema oluşturabilmek için [43] nolu çalışmada verilen ileri güvenliği sağlayan şema yapısı [46] nolu çalışmada önerilen ABS şemasına uyarlanmıştır. Daha sonra ileri güvenliği sağlayan ABS şeması [45] çalışmasındaki ABS şeması ile yer değiştirilerek standart modelde güvenli olduğu ispatlanabilen bir şema önerilmiştir. Kendi kendine anonimlik ve izlenebilirlik ispatları standart modelde LWE ve SIS problemlerinin zorluğuna dayandırılarak yapılan bu şemanın imza ve grup açık anahtarının boyutu gruptaki üye sayısından bağımsızdır.

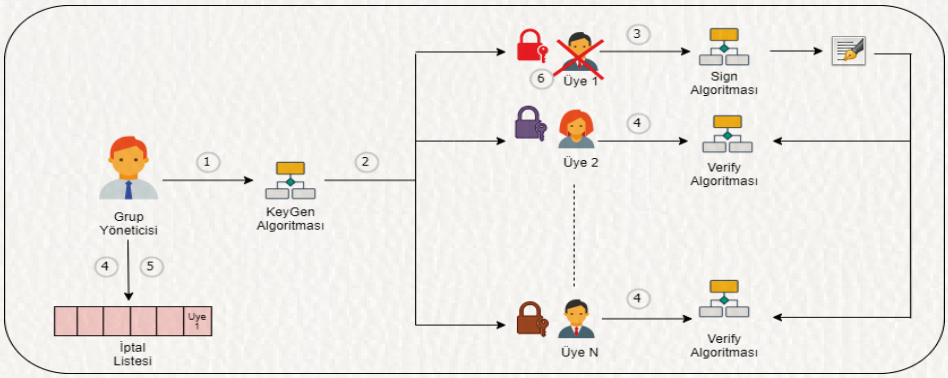
Standart modelde önerilen [45] ve [48] dikkate alındığında literatürde standart modelde SIS tabanlı NIZK bir sistemin kullanıldığı bir şemanın olmadığı görülmektedir. Bu eksiklik [49] nolu çalışmada giderilmiştir. İlk olarak [28]'de verilen ISIS probleminin zorluğuna dayanan etkileşimli sıfır bilgi paylaşımlı kimlik doğrulama şeması etkileşimsiz hale getirilmiştir. Standart modelde güvenli olan bir grup imzalama şeması oluşturmak istedikleri için yazarlar etkileşimsiz hale getirdikleri kimlik doğrulama şemasını, standart modelde

güvenli olduğu ispatlanmış [50]'de önerilen imzalama şemasını ve Regev'in şifreleme sistemini [23] birleştirerek bir grup imzalama şeması oluşturmuşlardır. Kafes tabanlı standart modelde güvenli olan ilk NIZK içeren grup imzalama şeması olan bu şema için tamamen anonim ve tamamen izlenebilirlik ispatları yapılmıştır.

Grup imzalama şemalarının maliyetini ve anahtar boyutunu düşürebilmek amacıyla yapılan başka bir çalışmada NTRU problemine dayanan grup imzalar oluşturulmuştur [51]. Bu çalışma, SIS ve LWE problemleri haricinde bir kafes probleminin zorluk varsayımı altında oluşturulduğundan önemlidir. NTRU sistemleri, açık anahtar boyutunu azalttıkları ve daha hızlı hesaplama gücüne sahip oldukları için tercih edilmiştir. NTRU tabanlı grup imzaların oluşturulması için ayrık Gauss örnekleme algoritması ile GPV imzalama şeması birleştirilmiştir. Şemanın anonimliği LWE probleminin zorluğuna dayanırken izlenebilir olması GPV imzalama şemasının güvenliği ile ispatlanmıştır.

11.2.2. VLR İptal Mekanizması

Bölüm 11.2.1'de anlatılan statik grup imzalama şemalarında bazı üyelerin grup adına imza atması istenmediğinde üyeleri gruptan çıkarmak veya imza atmalarını engellemek için grubun yeniden oluşturulması gerekmektedir. Bu durum pratikte istenmeyen bir durumdur. Grup imzalama şemalarında üyeliği iptal etmek için iki temel yaklaşım söz konusudur. Birinci yaklaşım, her üyelik iptalinden sonra grupta kalan üyelerin imzalama anahtarlarının güncellenmesini gerektirmektedir. Bunun yanı sıra, imzayı doğrulayacak tüm grup üyeleri yeni grup açık anahtarını almalıdır. Bu yaklaşım uygulamada verimli değildir. İkinci yaklaşım ise VLR yaklaşımı olarak isimlendirilmektedir [52]. Bu yaklaşımda üyelik iptal edildikten sonra iptal bilgilerini sadece imzayı doğrulayacak olan üyeler güncellemektedir. Bu nedenle imza atacak grup üyeleri iptal edilen üyelerle ilgili bir güncellemeye ihtiyaç duymadan imza atabilmektedir. Grup imzalama uygulamalarında geçerli imzayı doğrulayan üyelerin sayısının imza atabilenlerin sayısından daha az olduğu düşünüldüğünde bu yaklaşım daha verimli olmaktadır. Şekil 11.2'de VLR yaklaşımını kullanan şemalarında grubun oluşturulmasından bir üyenin iptal edilmesine kadar geçen süreç gösterilmiştir. Şemada sırasıyla aşağıda verilen işlemler gerçekleştirilmektedir:



Şekil 11.2. GSS-VLR Şemasının İşleyişi

- 1 numaralı adımda grup yöneticisi KeyGen algoritması ile gizli anahtarları ve iptal kimliklerini üretmektedir. Her bir üye için oluşturulan iptal kimliği üyenin gizli anahtarına bağlı olarak üretilmektedir.
- 2 numaralı adımda grup yöneticisi ürettiği anahtarları ve iptal kimliklerini üyelere vermektedir.
- Bir sonraki adımda gizli anahtarını ve iptal kimliğini alan Üye 1 güncel iptal listesine ihtiyaç duymadan sadece kendi gizli anahtarı ile mesajı imzalamaktadır.
- Üye 1 tarafından atılan imza 4 numaralı adımda grubun diğer üyeleri tarafından Verify algoritması kullanılarak doğrulanabilmektedir.
- 5 numaralı adımda Üye 1 hatalı davranırsa ve üyeliğinin iptal edilmesi gerekirse grup yöneticisi Üye 1'e ait iptal kimliğini iptal listesine RL'ye kaydederek en güncel listeyi doğrulayıcılara göndermektedir. Üyelik iptali için Üye 1 tarafından atılan imzanın doğrulanması şart değildir.
- Son adımda grup yöneticisinin RL'ye eklediği Üye 1 gruptan çıkarılarak imza yetkisi elinden alınmıştır. Eğer 4 numaralı adımda Üye 1'in üyeliği iptal edilmişse ve bu adımdan sonra Üye 2 atılan imzayı doğrulamak isterse RL'ye bakarak imza atan üyenin artık grup üyesi olmadığını tespit etmektedir.

GSS-VLR şemalarındaki iptal listelerinde üyelerin gizli anahtarları yerine iptal kimliklerinin kullanılması grup üyelerinin imzalama hakkını elinden

almakla kalmaz aynı zamanda o üye tarafından oluşturulan tüm imzaların geçersiz olmasına neden olmaktadır. VLR yaklaşımı ile üyeliğini kötüye kullanan üyelerin gruptan çıkarıldığı GSS-VLR şemaları aşağıda verilen üç algoritmadan oluşmaktadır:

- KeyGen(1^{λ},1^{N}): Bu algoritma ile grubun açık anahtarı gpk, her üyenin gizli anahtarı $gsk = (gsk[0], gsk[1], \dots, gsk[N-1])$ ve üyelerin iptal kimlikleri $grt = (grt[0], grt[1], \dots, grt[N-1])$ üretilmektedir. Burada grt iptal kimliklerinin gsk üyelerin gizli anahtarına bağlı olarak üretildiği unutulmamalıdır.
- Sign($gpk, gsk[d], M$): Grup açık anahtarını gpk, imzalayanın gizli anahtarını $gsk[d]$ ve imza atılacak mesajı M alan algoritma, grubun $d.$ üyesine ait olan imzayı Σ üretmektedir.
- Verify(gpk, RL, M, Σ): Grup açık anahtarını gpk, gruptan çıkarılan üyelerin kimliklerinin tutulduğu iptal listesini $RL \subseteq \{grt[0], grt[1], \dots, grt[N-1])\}$, mesajı M ve imzayı Σ alan algoritma imza atanın grup üyesi olduğunu, üyeliğinin iptal edilmediğini ve imzanın geçerli olduğunu doğrulamaktadır.

GSS-VLR şemaları, imza atan üyeleri izleyebilmek için Open algoritması yerine gizli izleme algoritması (implicit tracing algorithm) içermektedirler. Bu algoritmada izleme anahtarı olarak grt iptal kimlikleri kullanılmaktadır. $i \in \{0,1, \dots, N-1\}$ gruptaki üyeleri temsil etmek üzere mesaj ve imza çifti (M, Σ) verildiğinde bütün iptal kimliklerini $RL = grt[i]$ bilen yetkili bir kişi imzayı atan üyeyi tespit etmek için aşağıda verilen adımları uygulamaktadır.

- Verilen mesaj ve imza çifti (M, Σ) ile tüm kullanıcıların iptal kimliğini içeren iptal listesini $RL = \{grt[i] \mid i \in \{0,1, \dots, N-1\}\}$ kullanarak her üye için Verify($gpk, RL = \{grt[i]\}, \Sigma, M$) algoritmasını çalıştırmaktadır.

- Verify algoritmasının imzayı doğrulayamadığı ilk grup üyesinin indisini döndüren bu izleme algoritması, imza atan üyenin kim olduğunu açığa çıkarmaktadır. Öte yandan, Verify algoritması tüm grup üyeleri için verilen imzayı doğrularsa izleme algoritması başarısız olmaktadır.

Gizli izleme algoritması, gruptaki tüm üyeler için doğrulama algoritmasını çalıştırarak imza atan üyenin kimliğini tespit edebildiğinden büyük gruplarda verimli çalışmamaktadır. Bunun yanında, Open algoritması ile karşılaştırıldığında bu izleme algoritması daha fazla zaman almaktadır. Öte yandan, grup üyelerinin izleme anahtarları, gizli anahtarlarından türetildiğinden depolanmasına gerek yoktur.

11.2.2.1. Güvenlik Gereksinimleri

Burada GSS-VLR şemalarının güvenlik gereksinimlerinden bahsedilecektir. Hatırlanacağı üzere statik grup imzalama şemalarının güvenlik gereksinimlerini Bellare ve arkadaşları tam anonimlik ve tam izlenebilirlik olarak tanımlamışlardır. Statik şemalarda üyelerin sadece gizli anahtarları vardır. Tam anonimliği sağlayan şemalarda üyeler gizli anahtarları ile imza atarken gizli anahtarı ele geçiren bir saldırgan imzalayanın kimliğine ulaşamamaktadır. Öte yandan, GSS-VLR şemalarında üyeler, gizli anahtara ilave olarak iptal kimliklerine sahiptir. Bu iptal kimliklerinin ele geçirilmesi GSS-VLR şemalarının güvenliği açısından büyük bir tehdit oluşturmaktadır. Örneğin; saldırgan grubun i_0'ıncı üyesine ait iptal kimliğine $grt[i_0]$ ulaşırsa $grt[i_0]$'ı RL'ye ekleyerek Verify($gpk, RL = grt[i_0], \Sigma, M$) algoritmasını çalıştırarak verilen imzanın bu üyeye ait olup olmadığını anlayabilmektedir. Buradan VLR şemalarının tam anonimliği sağlayamadığı anlaşılmaktadır. Bu nedenle, GSS-VLR şemalarının güvenliği için tam anonimlik gereksinimindeki gizli anahtarlara ve iptal kimliklerine olan erişim kısıtlanarak yeni bir güvenlik gereksinimi tanımlanmıştır [52]. Kendi kendine anonimlik (selfless anonymity) olarak isimlendirilen bu güvenlik gereksinimi herhangi bir üyenin, mesajı imzaladığını unutması durumunda bile kendi gizli imzalama anahtarının belirli bir imza oluşturmak için kullanılıp kullanılmadığını tespit etmesine imkan vermektedir. Eğer imza bu üyeye ait değilse imzayı oluşturan üyenin kim olduğu yine gizli kalmaktadır. Boneh ve Shacham [52], VLR yaklaşımına dayanarak üyelik iptalini gerçekleştiren şemaların kendi kendine anonimlik özelliği haricinde tamlık ve izlenebilirlik özelliklerini de sağlamaları gerektiğini belirtmiştir.

11.2.2.2. Önceki Çalışmalar

Literatürde grup imzaların boyutlarında iyileştirmeler yapıldıkça şifreleme sistemleri kullanılmadan, daha zayıf güvenlik varsayımlarına dayanan ve üyeliklerin iptal edilebildiği daha verimli şemaların oluşturulup oluşturulamayacağı açık problemler olarak bırakılmıştır. Bu problemlere çözüm önerisi olarak gruptaki üyelerin iptal edilebildiği ilk kafes tabanlı grup imzalama şeması SIVP probleminin zorluğuna dayandırılarak oluşturulmuştur [32]. Bu şema oluşturulurken genel yapının aksine

- şemanın verimliliğini fazla arttırmadığı ve
- oluşturulacak grup imzalama şemasının güvenliğinin kullanılan şifreleme şemasının güvenliğine dayandırılmasının daha güçlü güvenlik varsayımlarına neden olduğu

gerekçeleriyle şifreleme şeması kullanılmamıştır. VLR yaklaşımı ile üyeliklerin iptal edilebildiği bu şemada etkileşimli bir yapı kullanıldığından ispatlayıcı kendisinin onaylanmış bir grup üyesi olduğunu ve üyeliğinin iptal edilmediğini doğrulayıcıya ispatlayabilmektedir. Öte yandan, şema zor rastgele kafesler için Bonsai ağaç yapısında yürütülmektedir [53]. Bonsai ağaç yapısı kullanılarak grup üyelerinin kimlikleri bir Bonsai imzası olarak onaylanmıştır. Şemada üyelik iptalini gerçekleştirmek amacıyla Stern'ün sıfır bilgi paylaşımlı 3-aşamalı kimlik doğrulama şemasına [39] yeni bir c_0 meydan okuma değeri eklenmiştir. Bunun yanında, etkileşimli olan bu kimlik doğrulama şemasının sağlamlık oranını düşürebilmek için pek çok kez çalıştırılmış ve Fiat-Shamir yaklaşımı [54] ile grup imzalama şemasına dönüştürülmüştür. SIVP probleminin en kötü durum zorluğuna dayanan şemanın açık anahtar boyutu $\tilde{O}(\lambda^2\, logN)$ ve imza boyutu ise $\tilde{O}(\lambda\, logN)$ olarak verilmiştir. Yapılan güvenlik analizinde şemanın doğruluğu verilerek kendi kendine anonimlik ve izlenebilirlik özelliklerini rastgele kahin modelinde sağladığı ispatlanmıştır. Öte yandan, yapılan incelemelerde şemanın doğrulama aşamasında imza atan üyenin iptal listesinde olup olmadığını kontrol eden yöntemde hata olduğu tespit edilmiştir. Yapılan hatadan dolayı şema gerçekte izlenebilir değildir. Yazarlar üyelerin iptal kimliklerini LWE örnekleri ile ilişkilendirerek üyelik iptalini güncellemişlerdir [55]. Bu şekilde izlenebilirlik saldırılarına karşı dirençli olan ve gruptan üyelerin çıkarılabildiği bir şema oluşturulmuştur.

İptal kimliklerini bilen herhangi biri imza doğrulama algoritmasını çalıştırarak imzanın ilgili üyeye ait olup olmadığını doğrulayabildiğinden literatürde tam anonimlik özelliğini sağlayan ve VLR yaklaşımını temel alan kafes tabanlı grup imzalama şeması önerilmemiştir. Örneğin; bir saldırgan iptal kimliklerini biliyorsa bu iptal kimliklerini ve Verify algoritmasını kullanarak imzayı doğrulamaktadır. İmza doğrulanmazsa iptal kimliğinin ait olduğu üyenin imzayı attığı anlaşılmaktadır. Bu nedenle doğrulayıcılara iptal kimlikleri verilmemelidir.

Saldırgana tüm iptal kimliklerinin verildiği durumda bile güvenli olan bir grup imzalama şeması oluşturulmuştur [56]. SIS ve LWE problemlerinin zorluğuna dayandırılarak oluşturulan şemada tek kullanımlık imzalama (one time signature-OTS) şeması ve Kawachi'nin Stern tarzı kimlik doğrulama şeması [42] kullanılmıştır. Elde edilen şemada, grup yöneticisi gruptan çıkarılacak üyeyi RL'ye eklemeden önce kendi gizli anahtarını kullanarak üyenin iptal kimliğini imzalamaktadır. Öte yandan, imzanın doğrulanması aşamasında doğrulayıcı,

- imzalayanın iptal kimliğinin RL'de olup olmadığını,
- imzanın geçerli olup olmadığını,
- RL'deki iptal kimliklerinin grup yöneticisi tarafından imzalanıp imzalanmadığını

kontrol etmelidir. İmzanın doğrulanması için grup yöneticisinin açık anahtarı kullanılmaktadır. Şemada saldırgan iptal kimliklerinin hepsini elinde bulundursa bile grup yöneticisinin gizli imzalama anahtarını bilmediğinden imzanın doğrulanması mümkün değildir. Sonuç olarak, RL'ye eklenen iptal kimliklerinin grup yöneticisi tarafından imzalanması ile şemaya tam anonimlik özelliği kazandırılmıştır. Başka bir ifadeyle, üyelerin gizli anahtarları ve iptal kimliklerini bilen bir saldırgan bile imzanın hangi üye tarafından atıldığını anlayamamaktadır. Bu nedenle, elde edilen bu şema VLR yaklaşımı ile üyelik iptalini gerçekleştiren ve tam anonim olma güvenlik gereksinimini sağlayan kafes tabanlı ilk grup imzalama şemasıdır. Bunun yanı sıra, kendi kendine anonimlik özelliğini sağlamadığından grup üyeleri kendi gizli anahtarları kullanılarak imzanın oluşturulup oluşturulmadığını kontrol edememektedir. Bu işlemin yapılabilmesi için üyeler grup yöneticisinin gizli anahtarına sahip olmalıdır. Ayrıca grup yöneticisinin iptal edilen her üyenin iptal kimliğini imzalaması gerekmektedir. Bu durum şemanın güvenliğinin grup yöneticisine

bağlı olduğunu göstermektedir. Gruptaki üyelerin bir şekilde grup yöneticisinin anahtarlarını elde ettiği bir durumda şema güvenli değildir.

GSS-VLR şemalarında imzalayanların kimliklerinin açık izlenebilirlik algoritması ile açığa çıkarılması için imzalayanların imzalama sırasında grupta kaçıncı üye olduğunu gösteren indislerini grup yöneticisinin açık anahtarını kullanarak şifrelemeleri gerekmektedir. Bu sayede sadece grup yöneticisi imzalayanların indislerinin şifresini çözmekte ve imzalayan üyenin kimliğine ulaşmaktadır. Bu amaç doğrultusunda, imza atan üyenin imzanın geçerli olduğunu, iptal kimliklerinin üyenin gizli imzalama anahtarlarından bağımsız bir şekilde oluşturulduğunu ve üyelik indisinin doğru bir şekilde şifrelendiğini ispatlamasını mümkün kılan etkileşimli sıfır bilgi paylaşımlı bir şema önerilmiştir [57]. Kawachi ve arkadaşlarının taahhüt fonksiyonu [42] kullanılarak LWE problemine dayanan Stern tarzı [39] bir şema oluşturulmuştur. Sıfır bilgi paylaşımlı bu şemanın GSS-VLR şemalarında kullanılabileceği ifade edilmiştir.

VLR yaklaşımının kullanıldığı başka bir çalışmada, ROM'da açık izlenebilirliğe (explicit tracing) sahip bir şema oluşturulmuştur [58]. Gizli izlenebilirliğe sahip imzalama şemalarında imza atan üyenin tespit edilebilmesi aşamasında tüm grup üyelerinin imzayı atıp atmadığının kontrol edilmesi gerekmektedir. Öte yandan, açık izlenebilirlikte böyle bir gereklilik olmadığından imza atan üyenin açığa çıkarılması daha az zaman almaktadır. Klasik GSS-VLR şemalarındaki üç algoritmaya Open algoritması eklenerek imza atan üyenin kimliğinin gruptaki üye sayısından bağımsız ve sabit bir zamanda açığa çıkarılması sağlanmıştır. Anahtar boyutunu küçültebilmek için üyelerin grup içerisindeki indisleri [33] nolu çalışmada verilen kimlik tabanlı kodlama tekniği (identity-encoding technique) ile kodlanarak kullanılmıştır. Kodlanan indisler [22]'de verilen LWE problemine dayanan sisteme göre şifrelenmiş ve imza atan üyelerin sertifikalı grup üyesi olduklarını ispatlayabilmek için Stern tarzı sıfır bilgi paylaşımlı bir sistem ile birleştirilerek verimli bir şema elde edilmiştir. Bunun yanında, kendi kendine anonimliği ve açık izlenebilirliği sağlayan bu şemanın üye sayısı fazla olan büyük gruplarda uygun bir şekilde kullanılabileceği belirtilmiştir.

2021 yılında tamamen anonim olan GSS-VLR oluşturabilmek için genel yapı tanımlanmıştır [59]. GSS-VLR şemasında tam anonimliği ye-

rine getirebilmek için herhangi bir açık anahtarlı şifreleme (Public Key Encryption-PKE) sistemi kullanılarak tasarlanmıştır. Her grup üyesi için kendi anahtar çiftlerine ilave olarak ilgili PKE sistemine ait şifreleme ve şifre çözme anahtarları da üretilmiştir. Tam anonimlik için üyelerin ilgili PKE şifre çözme anahtarı ve kendi açık anahtarı o üyeye ait iptal kimliği olarak tanımlanmıştır. Üye imzasını oluşturduktan sonra ilgili PKE şifreleme anahtarını kullanarak imzasını şifrelemektedir. İmzanın doğrulanması aşamasında ise iptal listesindeki iptal kimlikleri (her üyeye ait PKE şifre çözme anahtarları) kullanılarak imzanın şifresi çözülerek imzanın iptal edilen bir üye tarafından oluşturulup oluşturulmadığı kontrol edilmektedir. Oluşturulan bu yapıda üyelerdeki şifreleme anahtarından iptal kimliklerini üretmek mümkün olmadığından ilgili PKE sisteminin güvenliğine dayandırılarak şemanın tam anonimlik özelliğini sağladığı ispatlanmıştır. Bunun yanında, şemaya gruptan çıkarılan üyenin üyeliği iptal edilmeden önce attığı imzalarda bile kimliğinin gizli kalmasını sağlayan geriye doğru bağlantısızlık (backward unlinkability) özelliği kazandırılmıştır. Bu çalışmanın en önemli katkısı, imzalama şeması, PKE şeması ve sıfır bilgi paylaşımlı kimlik doğrulama şeması kullanılarak tamamen anonim ve geriye doğru bağlantı kurulamayan bir GSS-VLR şeması oluşturabilmek için genel yapıyı vermesidir. Bunun yanında, GSS-VLR şemaları oluşturulurken PKE yerine kimlik tabanlı şifreleme (Identity-based Encyrption-IBE) kullanıldığında şemadaki anahtar boyutlarının azaltılabileceği vurgulanmıştır.

Bu bölümde anlatılan şemalardan farklı olarak VLR yaklaşımı özellik tabanlı grup imzalama şemasında da kullanılmıştır [60]. Özellik tabanlı gruplarda grup üyelerinin hepsi aynı imza haklarına sahip değildir. Üyelere verilen özelliklere göre imzalama koşulunu sağlayan üye grup adına anonim olarak imza atabilmektedir. Üyelerin iptal edilebildiği özellik tabanlı grup imzalama şemasının ilk örneği olan bu şema, Boyen ve arkadaşlarının imzalama şeması [27] ve etkileşimsiz sıfır bilgi paylaşımlı Stern tarzı [39] bir şemanın birleşimidir. Öte yandan, VLR yaklaşımına göre tasarlandığından [32] nolu çalışmada olduğu gibi şifreleme şeması kullanılmamıştır. Şemanın açık anahtar boyutu $\tilde{O}(\lambda^2 \, logN)$ iken imza boyutu $\tilde{O}(\lambda \, logN)$'dir. Güvenlik analizi ROM'da verilen şemanın anonimlik, izlenebilirlik ve çerçevesizlik özelliklerini sağladığı SIVP probleminin zorluğuna dayandırılmıştır.

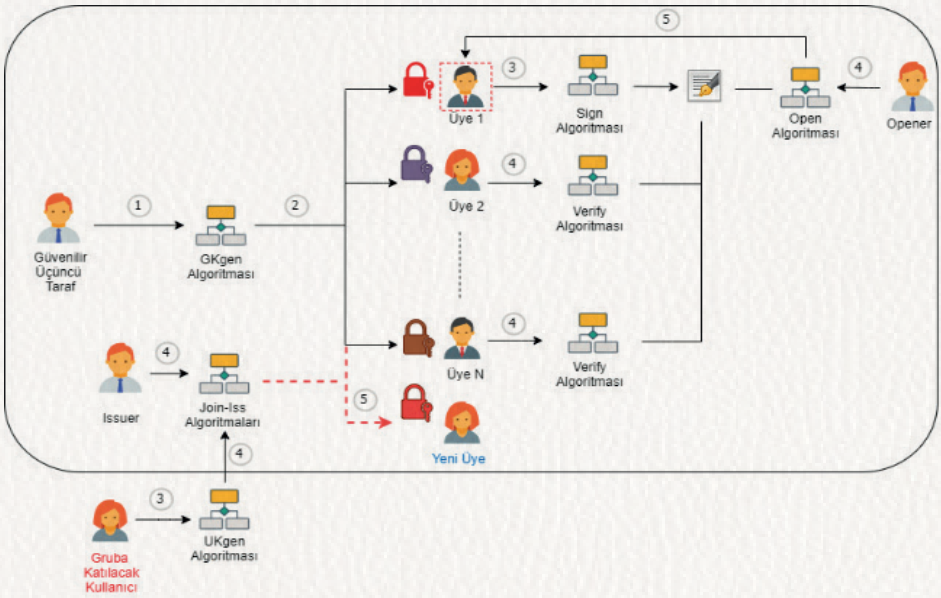
GSS-VLR ile ilgili daha önce yapılan çalışmalardan farklı olarak;

- üyenin kimlik bilgilerini kodlamak için sabit sayıda matrise ihtiyaç duyan bir kimlik kodlama tekniğini kullanan ve bu nedenle grup açık anahtarı ve üyelerin gizli anahtarları $log\ N$ ile orantılı olan,
- imza atan üyelerin sabit zamanda açığa çıkarılması için [22] nolu çalışmada verilen LWE problemine dayanan şifreleme sistemine dayanarak açık izleme yapabilen,
- iptal kimliklerinin üyelerin gizli anahtarlarından bağımsız olarak kısa Gauss vektörleri ve açık matrislerden elde edildiği

tamamen anonim GSS-VLR şemalarının oluşturulmasında kullanılabilecek yeni bir etkileşimsiz sıfır bilgi paylaşımlı Stern tarzı bir şema önerilmiştir [61]. Kawachi ve arkadaşlarının taahhüt fonksiyonunun [42] kullanıldığı şemanın güvenli olduğu hesaplama maliyeti, tamlık ve istatistiksel sıfır bilgi ispatlarıyla gösterilmiştir. Bu bölümde incelenen GSS-VLR şemaları Tablo 11.3'te şemalarda kullanılan kafes problemleri ve şemaların sağladığı güvenlik gereksinimleri açısından karşılaştırılmıştır. Tablo 11.4'te ise şemaların oluşturulmasında kullanılan alt sistemler ile imza ve anahtar boyutlarına göre karşılaştırma verilmiştir.

11.2.3. Kısmen Dinamik Grup İmzalama Şemaları

Grup imzalama şemasının daha özelleşmiş hali olarak düşünülebilir. Dinamik grup imzalama şemalarında (DGSS) kullanıcıların istedikleri zaman gruba katılarak grubun yeni üyesi olmalarını sağlayan etkileşimli bir şema bulunmaktadır. Öte yandan gruptan üyelerin çıkarılması söz konusu değildir. Bu nedenle, bu kitap bölümünde grup kurulduktan sonra yeni üyelerin eklenebildiği ancak gruptan üyelerin çıkarılamadığı bu tipteki şemaları kısmen dinamik grup imzalama şemaları (almost-dynamic group signature) olarak isimlendiriyoruz. Şekil 11.3'te başlangıçta anahtar üretimini yapan güvenilir üçüncü taraf, Issuer ve Opener adı verilen iki adet yetkili ve eşsiz tanımlayıcı numaralara sahip olacak şekilde grup üyeleri olan kısmen dinamik bir grup imzalama şeması verilmektedir.



Şekil 11.3. Kısmen Dinamik Grup İmzalama Şemasının İşleyişi

Şemanın işleyişi şu şekildedir:

- 1 numaralı adımda güvenilir üçüncü taraf Issuer ve Opener'a ait gizli anahtarları üretmektedir.
- 2 numaralı adımda grupta *N* tane grup üyesinin gruba eklenmiş olduğu düşünülmektedir. Bu adımdan sonra *N* tane üyeye sahip olan bu gruptaki imza atma, doğrulama ve gruba yeni bir üyenin eklenmesi işlemleri gerçekleştirilmektedir.
- 3 numaralı adımda grup üyelerinden herhangi biri örneğin; Üye 1 mesajı imzalamaktadır.
- 4 numaralı adımda gruptaki diğer üyeler imzayı doğrulayabilirken aynı zamanda Opener tarafından imza atanın Üye 1 olduğu da açığa çıkarılabilmektedir. İmzaların doğrulanması veya imza atan üyenin kim olduğunun tespit edilmesi işlemlerinde sıra gözetilmemektedir. Öte yandan, Üye 1 imza attığı sırada gruba yeni katılacak kullanıcı UKgen algoritması ile gizli anahtarlarını oluşturabilmektedir. Issuer ile gruba katılacak kullanıcı arasında yürütülen işlemler sonucunda 5 numaralı adımda olduğu gibi gruba yeni üye eklenmektedir.

Kısmen dinamik imzalama şemalarında sırasıyla aşağıda verilen algoritmalar çalıştırılmaktadır:

- GKgen(1^{λ}): Kurulum aşamasında güvenilir üçüncü taraf bu algoritmayı çalıştırmaktadır. λ güvenlik parametresi olmak üzere 1^{λ} değerini alan algoritma gpk grup açık anahtarını, ik ve ok Issuer ve Opener'ın gizli anahtarlarını üretmektedir.
- UKgen(1^{λ}): Gruba katılmak isteyen her kullanıcının anahtar üretmek için çalıştırdığı algoritmadır. 1^{λ} değerini alarak kullanıcı için açık ve gizli anahtar çiftini $(upk[i], usk[i])$ üretmektedir. Bütün kullanıcıların açık anahtarı upk tablosunda herkesin erişimine açıktır.
- <Join, Iss>: Kullanıcılar UKgen algoritması ile kişisel gizli ve açık anahtarlarını ürettikten sonra gruba katılmak isteyen kullanıcıların gruba katılması için <Join, Iss> etkileşimli algoritmaları çalıştırılmaktadır. Bu işlemde kullanıcı tarafındaki işlemler Join algoritmasını Issuer tarafındaki işlemler Iss algoritmasını oluşturmaktadır. Join algoritmasının sonunda kullanıcı artık bir grup üyesidir ve $gsk[i]$ grup gizli anahtarına sahiptir. Öte yandan, Iss algoritmasının sonunda kullanıcı gruba katılır ve bu kullanıcıya ait bilgiler kayıt $reg[i]$ tablosuna kaydedilmektedir.
- GSign($gpk, gsk[i], M$): $gsk[i]$ gizli anahtarına sahip olan i indisli grup üyesi $M \in \{0.1\}^*$ mesajına karşılık gelen imzayı Σ üretmek istediğinde bu algoritmayı kullanmaktadır.
- GVerify(gpk, M, Σ): Grup açık anahtarını gpk bilen her üye imzayı doğrulamak için bu algoritmayı çalıştırmaktadır. Grup açık anahtarı gpk, mesaj M ve imzaya Σ karşılık bir karar biti üretilmektedir. İmza Σ doğrulanırsa karar biti 1'e, aksi halde 0'a eşit olmaktadır.
- Open(ok, reg, M, Σ): Opener, kendi gizli anahtarını ok, Issuer'ın kayıt tablosunu reg, mesajı M ve imzayı Σ girdi olarak alarak bu algoritmayı çalıştırmaktadır. Algoritma sonlandığında grup açık anahtarı gpk ile ilişkili olacak şekilde mesaja M karşılık olarak imzayı Σ oluşturan i indisli grup üyesinin kimliği açığa çıkarılmaktadır. İmzayı atan kullanıcı grup üyesi değilse $i = 0$ değerini almaktadır. Open algoritmasının ikinci çıktısı ise Judge algoritması ile doğrulanabilecek τ kanıt değeridir.

- Judge $(gpk, j, upk[j], M, \Sigma, \tau)$: Grup açık anahtarını gpk, $j \geq 1$ indis değerini, j indisli kullanıcının açık anahtarını $upk[j]$, mesajı M, bu mesajın geçerli imzasını Σ ve kanıt değerini τ alan algoritma imzayı atan üyenin j indisli üye olduğunu gösteren kanıtın τ geçerli olup olmadığını kontrol etmektedir.

11.2.3.1. Güvenlik Gereksinimleri

Kısmen dinamik şemaları oluşturabilmek için yukarıda bahsi geçen 7 algoritmadan oluşan bu model Bellare ve arkadaşları [62] tarafından önerilmiştir. Kiayias ve Yung [63] da kısmen dinamik grup imzalar için bir model oluşturmuştur. Modeller arasındaki temel fark; birinci modelde imza açma algoritmasında bir kanıt oluşturularak bu algoritmanın düzgün çalıştığının Judge algoritması ile bir yetkiliye ispatlanmasıdır. Kiayias ve Yung tarafından önerilen kısmen dinamik şemalarda Open algoritmasında çıktı olarak kanıt değeri üretilmemektedir. Bu nedenle, Judge algoritması da bulunmamaktadır. Bunun yanında, bu iki farklı tip kısmen dinamik şemalarının güvenlik gereksinimlerinde de farklılıklar mevcuttur. Bellare ve arkadaşlarının kısmen dinamik şemalar için tanımladığı güvenlik gereksinimleri şöyledir:

- **Anonimlik:** İmza ve mesaj verildiğinde, imzayı atan üyeyi açığa çıkaran kahine erişimi olan bir saldırganın olduğunu düşünelim. Anonim olma güvenlik gereksinimini sağlayan bir şemada, saldırganın imza ve mesaj çiftinden imzayı atan üyenin kimliğine ulaşması imkansız olmalıdır. Başka bir ifadeyle, saldırgan verilen mesaja iki üyeden hangisinin imza attığını bilmemelidir.
- **İzlenebilirlik:** Grupta oluşturulan her geçerli imza bazı grup üyeleri tarafından izlenebilmelidir. Grupta imzalayanın kimliğini açığa çıkaran yetkili Judge algoritmasının kabul edeceği bir kanıt üretebilmelidir.
- **Çerçevesizlik (Non-frameability):** Dürüst bir grup üyesinin sonradan oluşturulmayan geçerli bir imza ile mesajı imzaladığını düşünelim. Bu durumda saldırgan, bu imza için Judge algoritmasına verildiğinde kabul edilecek bir kanıt oluşturamamalıdır. Başka bir ifadeyle, dürüst bir şekilde imza atan üyenin kimliğini açığa çıkaran Opener'ın doğru bir şekilde kimliği açığa çıkarması için kullanacağı imzanın saldırgan tarafından üretilemiyor olması gerekmektedir.

Kiayias ve Yung ise oluşturdukları kısmen dinamik şemaların sağlaması gereken özellikleri aşağıdaki gibi tanımlamışlardır [63].

- **Yanlış tanımlama (mis-idenitification) saldırılarına karşı güvenli olma:** Saldırgan, grubu oluşturan üyelerin kim olduğunu bilse bile açıldığında bu üyelere ait olduğu anlaşılacak imzaları üretememelidir.
- **Çerçeveleme (framing) saldırılarına karşı güvenli olma:** Bu özelliği sağlayan bir şemada, gruptaki tüm üyeler bir araya gelip dürüst bir üyeye komplo kursa bile dürüst grup üyesi imzalamadığı bir mesajı imzalamakla suçlanamaz.
- **Anonim olma:** İmza ve mesaj verildiğinde hangi grup üyesinin verilen mesaja karşılık verilen imzayı oluşturduğu bilinmemelidir.

gibi özellikleri sağlaması gerektiğini ifade etmişlerdir.

11.2.3.2. Önceki Çalışmalar

Hatırlanacağı üzere önceki bölümlerde gruba katılacak üyelerin kurulum aşamasında belirlendiği veya sabit olduğu statik grup imzalama şemalarından bahsedilmiştir. Bu bölümde ise gruptaki üye sayısının kurulum aşamasında belirlenmediği, kurulum aşamasından sonra gruba yeni üyelerin eklenebildiği kafes tabanlı şemalardan bahsedilecektir. Bu şemalar dayandıkları kafes varsayımları ve sağladıkları güvenlik gereksinimleri açısından Tablo 11.3'te, şemalarda kullanılan alt sistemler, imza ve açık anahtar boyutları açısından ise Tablo 11.4'te karşılaştırılmıştır.

2016 yılında ise herhangi bir zamanda gruba üyelerin katılabileceği ilk kafes tabanlı kısmen dinamik imzalama şeması önerilmiştir [64]. Şemada grup yöneticisi kullanıcıların gizli anahtarını imzalamak yerine yalnızca bilinen mesajları imzalamakta ve kullanıcının genel anahtarını imzalayarak bir üyelik sertifikası oluşturmaktadır. Üyelik sertifikası oluşturulan kullanıcıların yeni üye olarak gruba katılmalarını sağlayan bu yapı şemaya dinamiklik özelliği kazandırmıştır. İmzalama şeması olarak Böhl ve arkadaşları tarafından önerilen Boyen'in imzalama şemasının [27] bir versiyonu olan SIS tabanlı bir şema kullanılmıştır [65]. Stern tarzı etkileşimsiz bir sıfır bilgi paylaşımlı şema, Regev şifreleme şeması [22] ve Böhl imzalama şeması [65] birleştirilerek kısmen dinamik grup imzalama şeması elde edilmiştir. Önerilen şemanın birçok kullanıcının aynı anda grup yöneticisi ile etkileşimde bulunmak istediği ve

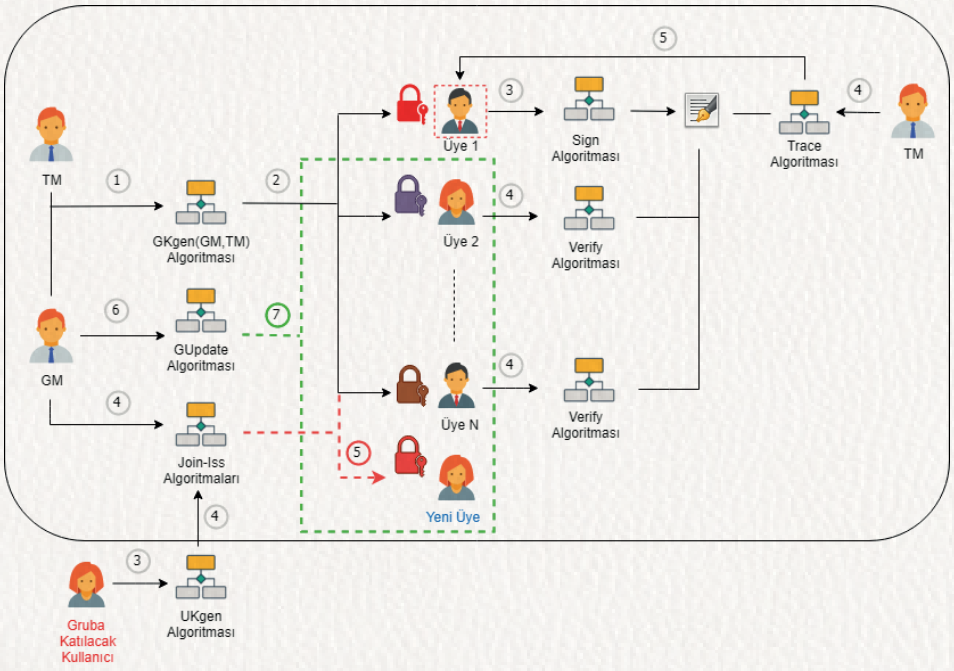
gruba katıldığı ortamda bile güvenli olduğu belirtilmiştir. Şemanın güvenlik analizinde Kiayias ve Yung'ın önerdiği yanlış tanımlama, çerçeveleme saldırılarına karşı güvenli ve anonim olma özelliklerini sağladığı ispatlanmıştır.

Bölüm 11.2.1'de verilen [21], [25], [26], [29], [32], [33] ve [37] nolu çalışmalardaki statik grup imzaları dahil olmak üzere şemaların parametreleri gruptaki maksimum üye sayısı olan N'ye bağlıdır. Bunun sonucu olarak imza boyutu gruptaki üye sayısından bağımsız olan imzalama şemalarının oluşturulması için çalışmalar yapılmıştır. RSIS ve RLWE problemlerine dayanan sabit boyutlu kafes tabanlı grup imzalama şeması önerilmiştir [66]. Şemanın sabit boyutlu olarak isimlendirilmesi imza boyutunun gruptaki maksimum üye sayısından bağımsız olduğu anlamına gelmektedir. Gruptaki açık anahtar ve imzalama anahtarının boyutları ile imza boyutu $\tilde{\mathcal{O}}(\lambda)$'dır. Bellare ve arkadaşlarının önerdiği yaklaşıma [62] göre oluşturulan şema kısmen dinamik yapıda olduğundan kurulum aşamasında grupta yer alacak maksimum üye sayısını sınırlandırmak gerekmemektedir. Bu şemayı oluşturmak için Ducas-Micciancio imzalama şeması [67], Naor-Yung çift şifreleme yaklaşımı [40] ile Regev şifreleme şeması [30] ve Stern tarzı [39] etkileşimsiz sıfır bilgi paylaşımlı bir kimlik doğrulama şeması birleştirilmiştir. Şemaya ait güvenlik analizi ROM'da tamlık, anonimlik, izlenebilirlik ve çerçevesizlik özelliklerinin sağlandığının ispatlanması ile yapılmıştır.

11.2.4. Tamamen Dinamik Grup İmzalama Şemaları

Tamamen dinamik grup imzalama şemalarında herhangi bir zamanda gruba yeni bir üyenin eklenmesi veya gruptan bir üyenin çıkarılması mümkündür. Bootle ve arkadaşlarının önerdiği dinamik modelde grup, gruba yeni üyelerin katılmasını sağlayan bir grup yöneticisi GM, imzaları açarak imzalayanların anonimliğini kaldıran bir izleme yöneticisi TM ve eşsiz kimlik bilgilerine sahip gruba katılma ihtimali olan kullanıcılardan oluşmaktadır [68]. Kullanıcılar, grup yöneticisi GM'nin onayı ile herhangi bir zamanda gruba katılabilmekte veya gruptan çıkabilmektedirler. Bu modelde, grup yöneticisi grubun güncel bilgilerini $info$ düzenli olarak gruptaki üyelere göndermektedir. Bu bilgiler grubun mevcut üyelerini veya gruptan çıkarılmış olan üyelerin bilgilerini içermektedir. Güncel grup bilgisini bir önceki grup bilgisi ile birleştiren bir üye, gruptan ayrılan üyelerin listesine ulaşmaktadır. Şekil 11.4'te bir GM ve TM yönetimindeki N üyeden oluşan tamamen dinamik yapıdaki şema şu şekilde yürütülmektedir:

- 1 ve 2 numaralı adımlarda GM ve TM kendi gizli anahtarları ile grup açık anahtarını üretmektedir. 2 numaralı adımın sonunda *N* tane üyenin gruba eklendiği kabul edilmektedir. Bu adımdan sonra grup adına imzanın atılması, imza atan üyenin açığa çıkarılması ve üyeliğinin iptal edilmesi ile gruba yeni bir üyenin eklenmesi işlemleri gerçekleştirilmektedir.
- Üye 1, 3 numaralı adımda mesajı anonim olarak imzalamaktadır.
- 4 numaralı adımda gruptaki üyeler imzayı doğrulayabilmektedir. Doğrulama işleminden bağımsız bir şekilde TM imzanın Üye 1 tarafından atıldığını tespit ederek üyenin kimliğini açığa çıkarmaktadır.
- 5 numaralı adımda kimliği açığa çıkarılan Üye 1 gruptan çıkarılmaktadır.
- 4 ve 5 numaralı adımlarda Üye 1 gruptan ayrılırken aynı zamanda yeni bir üyede gruba katılabilmektedir. 3 numaralı adımda UKgen algoritmasını çalıştırarak anahtarlarını üreten kullanıcı GM'nin onayı 5 numaralı adım ile gruba katılmaktadır.



Şekil 11.4. Tam Dinamik Grup İmzalama Şemasının İşleyişi

- 6 numaralı adımda GM, üyelik iptalini ve gruba yeni bir üyenin katıldığını belirten güncel grup bilgisini oluşturmaktadır.
- 7 numaralı adımda GM'nin güncel grup bilgisini grup üyelerine dağıtmaktadır. Bu şekilde grup üyeleri gruptan çıkarılan ve gruba yeni eklenen üyelerden haberdar olmaktadır. Üye 1'in gruptan çıkarıldığı ve yeni bir üyenin gruba eklendiği grubun son hali de bu adımda gösterilmektedir.

Bootle ve arkadaşlarının tam dinamik şemaların oluşturulması için geliştirdiği model polinom zamanda çalışan 9 algoritmadan oluşmaktadır:

- GSetup(1^{λ}): λ güvenlik parametresini alarak şemada kullanılacak açık parametreleri pp üretmektedir.
- <GK$gen_{GM}(pp)$,GK$gen_{TM}(pp)$>: Bu algoritma çifti grup yöneticisi GM ile izleme yöneticisi TM arasında etkileşimli olarak yürütülmektedir. Açık parametre olan pp değerini alan GKgen_{GM} ve GKgen_{TM} algoritmaları sırasıyla grup yöneticisine ve izleme yöneticisine ait olacak anahtar çiftlerini (mpk, msk) ve (tpk, tsk) üretmektedir. Bunun yanında GKgen_{GM} algoritması tamamlandığında grup yöneticisi GM grup bilgisini tutacak $info$ ve kayıt tablosunu ifade eden reg değerlerini başlatmaktadır. Etkileşimli olan iki algoritma sonucunda grup açık anahtarı $gpk = (pp, mpk, tpk)$ döndürülmektedir.
- UKgen(pp): Gruba katılacak kullanıcılar için açık ve gizli anahtar çiftinin üretildiği algoritmadır. Açık parametre pp girdi değerine karşılık kullanıcı için anahtar çifti (upk, usk) üretilmektedir.
- <Join($info_{\tau_{cs}}, gpk, upk, usk$), Issue($info_{\tau_{cs}}, mpk, upk$)>: Kullanıcı ve grup yöneticisi GM arasında yürütülen etkileşimli algoritma çiftidir. Join algoritması tamamlandığında kullanıcı uid indeksi ile gruba katılırken yeni grup üyesinin imzalama anahtarı $gsk[uid]$ olarak kaydedilmektedir. Issue algoritması tamamlandığında ise yeni grup üyesinin bilgileri reg kayıt tablosunun uid'inci indeksine girilmektedir.
- GUpdate($gpk, msk, info_{\tau_{cs}}, S, reg$): Grup yöneticisi GM grup bilgilerini güncellemek için bu algoritmayı çalıştırmaktadır. Grubun açık anahtarını gpk, grup yöneticisinin gizli anahtarını msk, grubun güncel

bilgisini $info_{\tau_{cs}}$, üyeliği iptal edilen kullanıcıların açık anahtarlarının kümesini S ve kayıt listesini reg girdi olarak alan algoritma grubun güncel durumdaki bilgisini $info_{\tau_{ns}}$ üretmektedir. GM, üyelik iptalinden sonra grubun güncel bilgi olan $info_{\tau_{ns}}$ değerini alıp kayıt tablosunu reg güncellemektedir. Grupta değişiklik yapılmadıysa GM işlem yapmadığını belirten $\perp$ döndürmektedir.

- Sign($gpk, gsk[uid], info_{\tau}, M$): Grubun uid'inci grup üyesi bu algoritma ile mesaja M karşılık gelen imzayı Σ üretmektedir. İmza atacak grup üyesinin τ döngüsünde aktif bir grup üyesi olmaması durumunda $\perp$ döndürülmektedir.
- Verify($gpk, info_{\tau}, M, \Sigma$): Mesaja M karşılık üretilen imzanın Σ geçerli olup olmadığını kontrol etmektedir. İmza doğrulanırsa 1, aksi halde 0 döndürülmektedir.
- Trace($gpk, tsk, info_{\tau}, \pi_{trace}, M, \Sigma$): İmzayı izleme yöneticisi olan TM tarafından çalıştırılan bu algoritma ile TM, imzayı atan üyenin kimliğini uid ve imzanın bir grup üyesine ait olduğunu gösteren kanıtı π_{trace} elde etmektedir. İmza grup üyelerinden birine ait değilse $\perp$ dönmektedir.
- Judge($gpk, uid, info_{\tau}, \pi_{trace}, M, \Sigma$): Bu algoritma Trace algoritmasının ürettiği π_{trace} değerinin geçerli olup olmadığını kontrol etmektedir.

11.2.4.1. Güvenlik Gereksinimleri

Tamamen dinamik grup imzalama şemalarını tanımlayan Bootle ve arkadaşları bu şemaların güvenlik gereksinimlerinin anonimlik, çerçevesizlik, izlenebilirlik ve sağlamlık izleme olduğunu ifade etmişlerdir [68].

- **Tam Anonimlik:** Atılan imzalardan imzayı üreten grup üyesinin kimliğine ulaşılamamalıdır. Tam anonimlik oyununda saldırgan, gruptaki tüm kullanıcıların gizli anahtarlarına ve imzalarına erişmektedir. Saldırgan herhangi bir zamana ait grup bilgisini, iki grup üyesini ve mesajı seçerek üyelerden birinin imzasını almaktadır. Bu oyunda saldırgan, imzanın seçtiği üyelerden hangisine ait olduğunu doğ-

ru bir şekilde tahmin ederse tam anonimlik oyununu kazanmaktadır. Saldırganın bu oyunu kazanması şemanın tam anonim olmadığını ifade etmektedir.

- **Çerçevesizlik:** Bu özelliği sağlayan bir şemada, tüm grup üyeleri hatta izleme ve grup yöneticileri bir grup üyesine karşı bir araya gelseler bile tek kalan grup üyesinin üretmediği bir imzayı ona ait olacak şekilde üretemezler. Saldırganın tüm üyelerin hatta yöneticilerin gizli anahtarlarına ulaşabildiği kabul edilmektedir.
- **İzlenebilirlik:** Bu özellik saldırganın gruptaki üyeler tarafından üretilmeyen bir imzayı oluşturamamasını sağlamaktadır. Saldırgan, grup yöneticisi dışında tüm üyelerin hatta izleme yöneticisinin bile imzalama anahtarına erişmektedir. Grup yöneticisinin gizli anahtarına ulaşması sahte kullanıcılar oluşturmasına neden olacağından saldırgana izin verilmemektedir. Saldırganın başarılı olması için kimliği tanımlamayan veya gruptan çıkarılmış bir üyeye ait olan bir imza üretmelidir.
- **Sağlamlık İzleme (Tracing Soundness):** Bu gereklilik, grup ve izleme yöneticisi dahil tüm grup üyeleri işbirliği yapsalar bile açıldığında iki farklı üyeye ait olacak geçerli bir imzanın üretememesini sağlamaktadır.

11.2.4.2. Önceki Çalışmalar

Hatırlanacağı üzere Bölüm 11.2.2 ve Bölüm 11.2.3'te üyeliklerin iptal edilebildiği [32] ve herhangi bir zamanda üyelerin eklenebildiği [64] kafes tabanlı grup imzalama şemalarından bahsedilmiştir. Bu şemalar herhangi bir zamanda gruba yeni üyelerin eklenmesi ve gerekli olduğu durumda üyelerin gruptan çıkarılma özelliklerinin ikisini birden sağlamamaktadır. Kullanıcıların gruba katılması ve gruptan çıkarılması işlemlerini yerine getiren kafes tabanlı şemaların oluşturulması açık problem olarak bırakılmıştır. 2017 yılında bu açık probleme çözüm önerisi getirilerek SIS ve LWE problemlerinin zorluk varsayımı altında Bootle ve arkadaşlarının önerdiği modelin güvenlik gereksinimlerini karşılayan ilk kafes tabanlı tamamen dinamik grup imzalama şeması önerilmiştir [69]. [37] nolu çalışmada verilen şemanın tamamen statik olan yapısından yola çıkılarak basit ve anlaşılır yaklaşım ile tamamen dinamik bir grup imzalama şeması elde edilmiştir. [37]'deki şemaya tam dinamiklik özelliği kazandırabilmek için aşağıda verilen değişiklikler yapılmıştır:

- [37]'de verilen şemanın kurulum aşamasında gruba katılacak her kullanıcının anahtarını grup yöneticileri rastgele seçmektedir. Tamamen dinamik şema oluşturulurken bu yaklaşım değiştirilerek çerçevesizliği yerine getirebilmek için anahtarların seçilmesi işlemini grup yöneticisi yerine kullanıcıların kendileri yapmaktadır.
- Hatırlanacağı üzere [37]'deki şema SIS tabanlı Merkle ağaç akümülatörünün temel mantığına dayanmaktadır. Statik yapıda olan bu akümülatöre dinamik bir bileşen eklenerek yeni üyeler gruba katıldığında veya üyelikler iptal edildiğinde üye bilgilerini güncellemek amacıyla kullanılmıştır. Gruba katılan üyelerin kimlikleri Merkle ağaç yapısında saklanacağından üyelik iptali söz konusu olduğunda ağacın yaprağından köküne kadar verimli bir şekilde güncelleme yapılmasını sağlayan bir algoritma eklenmiştir.
- Gruba yeni üyelerin eklenmesi ve üyeliklerin iptal edilmesi işlemlerinin basit bir şekilde gerçekleştirilmesi için grup üyelerinin her biri ağacın yapraklarıyla ilişkilendirilmiştir. Eğer kullanıcı gruba katılmışsa veya üyeliği iptal edilmişse, bu grup üyesi ile ilişkilendirilen yaprağın değeri 0 olarak atanmıştır. Gruba yeni bir üye katılmış ise ağaçta yeni üye ile ilişkilendirilen yaprağın değerine yeni grup üyesinin açık anahtarı atanmaktadır.

Hem gruba yeni üyelerin eklenmesi hem de gruptan üyelerin çıkarılmasını sağlayan bu şema statik şemalar içerisinde en verimli olarak bilinen [37]'de verilen şemadan daha kısa imza boyutuna sahiptir. Şemanın imza boyutu $\tilde{O}(\lambda\, logN)$ ve açık anahtar boyutu $\tilde{O}(\lambda^2 + \lambda\, logN)$ şeklinde verilmektedir. Şemanın güvenlik analizinde tamlık, anonimlik, çerçevesizlik, izlenebilirlik ve sağlamlık izleme özelliklerinin sağlandığını gösteren ispatlar yer almaktadır. 2019 yılında tamamen dinamik olan bu şemaya Ishida ve arkadaşları tarafından önerilen reddedilebilirlik (deniability) özelliği [70] kazandırılmıştır [71]. Belirli bir grup üyesinin verilen imzayı atmakla suçlandığı bir senaryo düşünüldüğünde bu özellik sayesinde imza açma yöneticisinin anonimliği koruyarak belirtilen kullanıcının gerçekte imzayı atan üye olmadığını gösteren bir kanıt yayınlamaktadır. Özelliğin sağlanması için şemaya Bootle ve arkadaşlarının oluşturduğu 9 algoritmadan oluşan modele iki farklı algoritma daha eklenmiştir. Eklenen algoritmalar ile belirtilen imzayı atmakla suçlanan üyenin gerçekten imzayı oluşturan kişi olup olmadığı kontrol edilmiştir.

Yapılan başka bir çalışmada ise üyelerin gruba eklenmesi ve gruptan çıkarılması için VLR yaklaşımı kullanılarak kafes tabanlı ve tamamen dinamik bir şema elde edilmiştir [72]. Grup imzalama şemasını oluşturmak için tek kullanımlık bir imzalama şeması [73], Regev'in şifreleme şeması [30] ve Stern tarzı [39] bir etkileşimsiz sıfır bilgi paylaşımlı kimlik doğrulama şeması [42] kullanılmıştır. Hatırlanacağı üzere [64]'te gerektiğinde gruba yeni üyelerin katıldığı bir şema oluşturulmuştur. [64] nolu çalışmadaki kısmen dinamik şema yapısının üzerine üyelerin gruptan çıkarılması için Revoke algoritması eklenmiştir. Şemanın tamamen dinamik yapıda olması ve üye ekleme-çıkarmaya izin vermesi için VLR yaklaşımındaki iptal listesi kullanılmıştır. Gruba yeni üyeler kaydedildiğinde kayıt tablosuna üyelerin kimlikleri kaydedilmektedir. Eklenen bu üyenin gruptan çıkarılması gerektiğinde ise kayıt tablosundaki kimlik silinerek iptal listesine kaydedilmektedir. İptal listesine üyelerin imzalama anahtarlarının kaydedilmemesi saldırgana tüm imzalama anahtarlarının verildiği durumda bile şemanın tam anonimliği yerine getirmesini sağlamıştır. Bunun yanında, grubun kurulum aşamasında veya grup oluşturulduktan sonra saldırgan gruba katılmak isterse saldırgan için de iptal kimliği oluşturulmaktadır. Saldırganın bu kimliği kullanması şemanın anonimlik özelliğini tehdit etmektedir. Bu durumda şema dinamik ancak yaklaşık olarak tam anonimlik özelliğini sağlamaktadır. Bununla birlikte, SIS ve LWE problemlerinin zorluğuna dayandırılarak şemanın izlenebilirlik ve çerçevesizlik özelliklerini de sağladığı gösterilmiştir. Bu şemanın gerçek yaşam uygulamasında kullanılabilirliğini göstermek için şemaya zaman damgası eklenmiştir [18]. Zaman damgası eklenerek kısa bir süre zarfında otelde kalan müşterilerin anonim olarak otel imkanlarından faydalanabilmeleri sağlanmıştır. Otele gelen müşterilerin geçici olarak gruba eklenmesi iptal listelerinin boyutlarının çok hızlı bir şekilde artmasına neden olacağından imzanın doğrulanması aşamasında imza atan üyenin iptal listesinde olup olmadığının kontrol edilmesi uzun sürmektedir. Eklenen zaman damgası ile gruba eklenen her üyenin gizli anahtarı için bir geçerlilik süresi oluşturulmaktadır. Bu sayede geçerlilik süresi dolan üye gruptan çıkarılmış olacağından imza yetkisi elinden alınmış olmakla beraber bu üyenin iptal listesine eklenmesine gerek kalmamaktadır. Bu da imzanın doğrulanması aşamasında uzun iptal listelerinin kontrol maliyetini düşürmektedir.

Bölüm 11.2.3'te [64] nolu çalışmada kurulum aşamasından sonra gruba üyelerin eklenebildiği kısmen dinamik bir şema önerildiğinden bahsedilmişti. Bu

çalışmaya VLR üyelik mekanizması ile üyelik iptali eklenerek tamamen dinamik bir yapı kazandırılmıştır [74]. Bu nedenle, elde edilen şemada kurulum aşamasından sonra gruba yeni üyeler eklenirken gruptan üyelerin çıkarılması da mümkündür. Şemada, grup yöneticisi gruba katılacak yeni üyelere sertifika sağlarken aynı zamanda hatalı davranan üyelerin üyeliğini iptal etmektedir. Bu üyelik iptalinin gerçekleştirilmesi için grup yöneticisi tarafından üyelerin gizli imzalama anahtarlarından üretilen iptal kimlikleri kullanılmıştır. İptal kimliklerinin grup yöneticisi tarafından oluşturulması hatalı davranan üyenin üyeliği iptal edilirken grup yöneticisine veya imzalama sırasında doğrulayıcıya sahte iptal kimlikleri vermesini engellemektedir. Üyelerin iptal edilebilmesi için [64]'te verilen şemaya Revoke algoritması eklenmiştir. Öte yandan, şemaya üyelik iptalinin kazandırılması tam anonimlik gereksinimini yerine getirememesine neden olmaktadır. Tam anonimliği sağlayan şemaların aksine üyelerin gizli anahtarlarının saldırgana verilmesine izin verilmemektedir. Bu nedenle, tam anonimliği yerine getiremeyen bu şema için güvenlik gereksinimi olarak saldırgan tarafından sorgulanmadığı sürece iptal kimliklerinin verilmediği ve iptal kimlikleri sorgulanan üyelere ait imzaların taklit edilemediği yaklaşık olarak tam anonimlik tanımı yapılmıştır. ROM'da izlenebilirlik ve çerçevesizlik gereksinimlerinin sağlandığı da ispatlanmıştır.

11.3. KAFES TABANLI GRUP İMZALAMA ŞEMALARININ KULLANIM SENARYOLARI VE KARŞILAŞTIRILMALARI

Bu bölümde kafes tabanlı grup imzalama şemalarının uygulama alanlarından bahsedilmektedir. Grup imzaların gerçek hayattaki kullanım senaryolarına örnekler verilmektedir. Bunun yanında, literatürde yer alan grup imzalama şemaları kafes varsayımları çerçevesinde ele alınarak şemalar sağladıkları özellikler, güvenlik gereksinimleri, anahtar ve imza boyutlarına gibi kriterlere göre karşılaştırılmaktadır.

11.3.1. Kullanım Senaryoları

Grup imzalama şemaları, imza atan üyelerin gerçek kimliklerinin grup yöneticisi tarafından açığa çıkarılmadığı müddetçe gizli kalması gerektiği ve atılan imzaların gruptaki diğer üyeler tarafından doğrulanabildiği senaryolarda uygulama alanı bulmaktadır [17]. Grup imzalama şemaları yaygın olarak kurum

yapılarının gizlenmesi gereken uygulamalarda kullanılmaktadır: Örneğin; şirket çalışanları şirket adına sözleşmeleri imzalayabilir, basın açıklaması yayınlayabilir, kamu ihalelerine katılabilir veya finansal işlemleri yürütebilir. Çalışanlar bu işlemleri yürütürken grup imzalarını kullanırsa çalışanların kimlikleri gizli kalacaktır. Öte yandan, eğer bir çalışan şirketin itibarına zarar verecek bir işlemde bulunursa şirket yöneticisi tarafından kimliği tespit edilerek imza hakları iptal edilmektedir. Grup imzalama şemaları çeşitli e-ticaret uygulamalarında da kullanılmaktadır. Örneğin; elektronik para ile işlem yapan kullanıcıların gizliliğini korumayı amaçlayan e-para uygulamalarında ve müzayedeye katılanların gizliliğini korumak için dijital müzayedeler veya seçmenlerin isimsiz olarak oy kullanmaları gereken dijital oylama gibi açık artırma uygulamalarında kullanılmaktadır [17], [18]. Bunların yanı sıra, anonim olarak online haberleşmede, güvenilir bilgi işlem platformları, gizlilik koruma mekanizmaları ve dijital hakların yönetiminde de kullanılmaktadır. Grup imzaların kullanılabileceği gerçek hayat senaryolarından bazıları şu şekildedir:

Senaryo 1: Bir proje değerlendirme sisteminde projeleri değerlendiren bir kurumda grup imzalar kullanılabilmektedir. Üç aşamalı bir değerlendirme sürecinin olduğu ve her bir aşamanın tamamlanması için sadece bir hakemin projeyi kabul etmesinin yeterli olduğu bir değerlendirme sürecini düşünelim. Projeleri değerlendiren bu kurumda, üç farklı kategoride hakemlerin ve bir başkanın olduğunu farz edelim. Proje değerlendirme sürecinin birinci aşamasındaki hakemlerden herhangi biri gelen projeyi kabul ederse proje bir sonraki aşamaya geçecektir. Aksi halde bu aşamadaki hakemlerden bir tanesinin daha değerlendirme yapması beklenecektir. Bu şekilde devam edilerek üç aşamayı tamamlayan proje kabul edilmektedir. Böylesi bir yapıda her bir aşamada projeyi kabul veya reddeden hakemlerin kimler olduğu diğer hakemler tarafından bilinmezken başkan değerlendirme sürecine katılan tüm hakemlerin kim olduğunu bilmektedir. Projeler bu şekilde değerlendirildiğinde proje sahibi, gönderdiği projenin sadece kabul veya reddedildiğini öğrenmektedir. Başka bir ifadeyle, kendisine gönderilen bilgilendirmeden projeyi değerlendiren kurumu ve sonuç bilgisini doğrulayabilmektedir. Projeyi kabul veya reddedenin kim veya kimler olduğunu bilmemektedir.

Senaryo 2: Otelde kalan müşterileri yönetmek için grup imzalama şemaları kullanılabilmektedir. Otel bir grup olarak düşünüldüğünde otele gelen her müşteri gruba yeni üye olarak eklenmektedir. Otel yönetimi, her müşteriye

bir otel kartı vermektedir. Müşteriler otelde kaldıkları süre boyunca bu kartı kullanarak otel imkanlarından faydalanmaktadırlar. Örneğin, müşteriler kendilerine verilen kart ile odalarına giriş/çıkış yapabilir veya spor salonunu kullanabilirler. Bu şekilde müşteri kendisine sunulan hizmetlerden faydalanırken müşterinin ne zaman hangi hizmetten yararlandığı bilgisi oteldeki kart görevlileri tarafından izlenebilmektedir. Öte yandan müşteri otelden ayrıldığında kartın geçerlilik süresi dolacağından kartın tekrar kullanılması söz konusu olmamaktadır.

Senaryo 3: Grup imzalar pandemi sürecinde ulaşım sistemlerinde kullanılabilir. Örneğin, hes kodu tanımlaması yapılmış ulaşım kartı ile bir yolcu metroya binebilir. Yolcu kartının kullanıldığı her noktada zaman damgası eklenmelidir. Belirli saatler içerisinde aynı metroda bulunan kişilerden bir tanesinin Covid-19'a yakalanması üzerine eklenen zaman damgası kullanılarak onunla temaslı olan tüm yolculara ulaşılarak flasyon çalışması gerçekleştirilebilir.

Senaryo 4: Akıllı otopark sistemlerinde de kullanılabilir. Bir araç sürücüsünün aracını otoparka park etmek istediği bir senaryoyu ele alalım. İlk olarak, bu sürücünün bulunduğu ildeki park hizmet sağlayıcısına kaydolması gerekmektedir. Bu şekilde bulunduğu ilde akıllı park sistemini kullanan otoparklara kayıt yapılmaktadır. Sürücü gideceği hedefe en yakın noktadaki otopark ile iletişime geçerek doluluk oranını öğrenebilmektedir. İlgili otoparktan park iznini aldıktan sonra park cihazını kullanarak park etme işlemini gerçekleştirmektedir. Böyle bir senaryoda araç sürücülerinin akıllı cihazlarından veya araçlardan gelen bildirimlerdeki gizliliği artırmak için grup imzalar kullanılmaktadır. Bunun yanında, sistemde gerçekleşecek olan bildirimlerde anonimliği, bütünlüğü, kimlik doğrulamayı ve inkar edilemezliği sağlayabilmek için grup imzalar kullanılmaktadır.

11.3.2. Karşılaştırma

Bu bölümde, ilk olarak statik, VLR, kısmen veya tamamen dinamik grup imzalama şemalarının sağlaması gereken güvenlik gereksinimlerine yer verilmektedir. Sonrasında Bölüm 11.2'de anlatılan şemalar, dayandıkları zor problemler, güvenlik gereksinimleri ve üye ekleme/çıkarmaya izin verme durumlarına, alt yapılarında kullanılan sistemlere, grup açık anahtar ve imza boyutları gibi kriterlere göre iki farklı tabloda incelenmektedir.

Tablo 11.2. Grup İmzalama Şemalarının Güvenlik Gereksinimleri

Şema Tipi	Güvenlik Gereksinimi
Statik [20]	Tamlık Tam Anonimlik Tam İzlenebilirlik
VLR [52]	Tamlık Kendi Kendine Anonim İzlenebilirlik
Kısmen Dinamik (Model 1) [62]	Tamlık Anonim Olma İzlenebilirlik Çerçevesizlik
Kısmen Dinamik (Model 2) [63]	Tamlık Anonim Olma Yanlış Tanımlama Çerçeveleme
Tamamen Dinamik [68]	Tamlık Anonim Olma İzlenebilirlik Çerçevesizlik Sağlamlık İzleme

Tablo 11.2'de şema tipine göre şemaların güvenlik gereksinimlerine yer verilmektedir. Tablo 11.2 dikkate alındığında şema tipinden bağımsız olarak tüm grup imzalama şemalarının tamlık ve anonim olma özelliklerini sağlaması gerekmektedir. Tanımlanan modellere göre güvenlik gereksinimlerinin isimleri değişiklik göstermesine karşın temelde hepsinde sağlanması gereken özellikler şöyledir:

- Atılan imza geçerli olmalıdır ve doğrulanabilmelidir.
- Bir grup üyesi imza attığında kimliği gizli kalmalıdır.
- İmza atan üyelerin kimliklerine ulaşılabilmelidir.
- Hiç kimse başkasının yerine imza atamamalıdır.
- Grup üyesi imza atıldıktan sonra imzayı attığını inkar edilememelidir.
- Hiç bir üye atmadığı bir imzayı atmakla suçlanamamalıdır.

Tablo 11.3. Şemaların Güvenlik Gereksinimlerine Göre Karşılaştırılması

Şema	Dayandığı Problem	Anonimlik	İzlenebilirlik	Çerçevesizlik	Üye Ekleme	Üyelik İptali	Statik/Dinamik
[21]	LWE	+	+	-	-	-	Statik
[25]	LWE	Tam A.	+	+	-	-	Statik
[26]	LWE SIS	Tam A.	+	-	-	-	Statik
[29]	LWE SIS	+	+	-	-	-	Statik
[33]	LWE SIS	Tam A.	+	-	-	-	Statik
[37]	LWE SIS	Tam A.	+	-	-	-	Statik
[41]	LWE SIS	Tam A.	+	-	-	-	Statik
[43]	LWE SIS	Tam A.	+	-	-	-	Statik
[44]	RLWE RSIS	Tam A.	+	-	-	-	Statik
[45]	LWE SIS	Kendi kendine A.	+	-	-	-	Statik - Sabit
[48]	LWE SIS	Kendi kendine A.	+	-	-	-	Statik - Sabit
[49]	LWE SIS	Tam A.	+	-	-	-	Statik
[32]	SIVP	Kendi kendine A.	-	-	-	+	VLR
[55]	SIVP	Kendi Kendine A.	+	-	-	+	VLR
[56]	LWE SIS	Tam A.	+	-	-	+	VLR
[58]	LWE SIS	Kendi kendine A.	+	-	-	+	VLR
[64]	LWE SIS	+	-	-	+	-	Kısmen Dinamik
[66]	RLWE RSIS	Tam A.	+	+	+	-	Kısmen Dinamik
[69]	LWE SIS	+	+	+	+	+	Tamamen Dinamik
[72]	LWE SIS	Yaklaşık Tam A.	+	+	+	+	Tamamen Dinamik
[74]	LWE SIS	Yaklaşık Tam A.	+	+	+	+	Tamamen Dinamik

Bölüm 11.2'de yer alan şemalar, dayandıkları zor problemler, güvenlik gereksinimleri ve gruba üye ekleme/çıkarmaya izin verme durumları bakımından Tablo 11.3'te karşılaştırılmaktadır. Kafes varsayımları altında önerilen şemaların güvenliklerinin LWE ve SIS problemlerinin zorluğuna dayandığı görülmektedir. Tabloda sırasıyla Tam A. ve Kendi Kendine A. tam ve kendi kendine anonim olan şemaları ifade ederken Statik-Sabit imza boyutu gruptaki maksimum üye sayısından bağımsız olan şemaları ifade etmektedir. Hatırlanacağı üzere grup imzalama şemasının genelde bir imzalama şeması, şifreleme şeması ve etkileşimsiz sıfır bilgi paylaşımlı bir şemanın birleşiminden elde edilmektedir.

Tablo 11.4'te Bölüm 11.2'de bahsi geçen şemaların oluşturulmasında kullanılan alt şemalar ve grup açık anahtarı ile imza boyutları verilmektedir. Bununla birlikte, şemaların Tablo 11.3'te verilen güvenlik gereksinimlerini hangi modelde (SM / ROM) sağladığı gösterilmektedir.

Tablo 11.4. Şemaların Alt Sistem ve İmza Boyutlarına Göre Karşılaştırılması

Şema	Şifreleme	NIZK	İmzalama	İmza Boyutu	Açık Anahtar Boyutu	Kuantum Güvenliği
[21]	Regev [23]	Micciancio [24]	GPV [22]	$\tilde{O}(\lambda^2 N)$	$\tilde{O}(\lambda^2 N)$	ROM
[25]	Regev [23]	Micciancio [24]	GPV [22]	$\tilde{O}(N)$	$\tilde{O}(N)$	ROM
[26]	Regev [23]	Lyubashevsky [28]	Boyen [27]	$\tilde{O}(\lambda^2 \log N)$	$\tilde{O}(\lambda^2 \log N)$	ROM
[29]	GPV-IBE [22]	Stern tarzı [31], [39], [42]	Boyen [27]	$\tilde{O}(\lambda \log N)$	$\tilde{O}(\lambda^2 \log N)$	ROM
	LPR [30]	Stern tarzı [31], [39], [42]	Boyen [27]	$\tilde{O}(\lambda \log N)$	$\tilde{O}(\lambda \log N)$	
[33]	Regev [23], [34]	Lyubashevsky [28], [35], [36]	Lyubashevsky [35]	$\tilde{O}(\lambda + \log^2 N)$	$\tilde{O}(\lambda^2 \log^2 N)$	ROM
[37]	Regev [23] Naor Yung [40]	Stern tarzı [31], [39]	Merkle ağaçları [38]	$\tilde{O}(\lambda \log N)$	$\tilde{O}(\lambda^2 \log N)$	ROM
[41]	GPV [22]	Stern tarzı [31], [39]	Boyen [27]	$\tilde{O}(\lambda \log N)$	$\tilde{O}(\lambda^2 \log N)$	ROM
[43]	GPV [22]	Stern tarzı [31], [39]	Bonsai ağaçları [53]	$\tilde{O}(\lambda \log N)$	$\tilde{O}(\lambda^2 \log N)$	ROM
[44]	LPR [30]	-	Lyubashevsky [35]	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda)$	ROM
[45]	Regev [23] OTS [47]	-	ABS [46]	$\tilde{O}(\lambda^2 N)$ $\tilde{O}(\lambda)$	$\tilde{O}(\lambda^2 N)$ $\tilde{O}(\lambda)$	SM
[48]	Regev [23] OTS [47]	-	ABS [46]	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda^2)$	SM
[49]	Regev [23]	Lyubashevsky [28], [35]	Fenghe [50]	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda^2 \log N)$	SM
[32]	-	Stern tarzı [39], [42]	Bonsai ağaçları [53]	$\tilde{O}(\lambda \log N)$	$\tilde{O}(\lambda^2 \log N)$	ROM
[55]	-	Stern tarzı [39], [42]	Bonsai ağaçları [53]	$\tilde{O}(\lambda \log N)$	$\tilde{O}(\lambda^2 \log N)$	ROM
[56]	-	Stern tarzı [39], [42]	OTS [73]	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda)$	ROM
[58]	GPV [22]	Stern tarzı [39], [42]	IBE [33], [34]	$\tilde{O}(\lambda \log N)$	$\tilde{O}(\lambda^2)$	ROM
[64]	Regev [23]	Stern tarzı [39], [42]	Böhl [65]	$\tilde{O}(\lambda \log N)$	$\tilde{O}(\lambda^2 \log N)$	ROM
[66]	LPR [30] Naor Yung [40]	Stern tarzı [39]	Ducas Micciancio [67]	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda)$	ROM
[69]	Regev [23] Naor Yung [40]	Stern tarzı [39], [42]	Merkle ağaçları [38]	$\tilde{O}(\lambda \log N)$	$\tilde{O}(\lambda^2 + \lambda \log N)$	ROM
[72]	GPV [22]	Stern tarzı [39], [42]	OTS [73]	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda)$	ROM
[74]	GPV [22]	Stern tarzı [39], [42]	Böhl [65]	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda^2 \log N)$	ROM

11.4. SONUÇ VE DEĞERLENDİRMELER

Kuantum bilgisayarlar ile hesaplama gücünün artacak olması açık anahtarlı sistemler için tehdit oluşturmaktadır. Günümüzde kullanılan sistemlerin kuantum saldırılarına dirençli sistemler ile değiştirilmesi gerektiğinden kuantum bilgisayarlarda bile çözülemeyen sistemler geliştirilmektedir. Kuantum saldırılarına dirençli olduğu bilinen aday sistemler arasında işlem basitliğinden, asimptotik açıdan verimli olmalarından ve kafes problemleri arasında en kötü durum / orta durum arasında geçiş yapmaya izin vermelerinden dolayı kafes tabanlı sistemler daha çok tercih edilmektedir.

Dijital imzalar mesajı imzalayanın kim olduğunu, mesajın bütünlüğünün bozulmadığını ve imza atanın sonradan attığı imzayı inkar edemediğini garanti etmektedir. Grup imzalarda ise grubu oluşturan üyeler grup adına anonim olarak imza atmaktadır. Dijital imzalarda imza atanın kimliği herkes tarafından bilinirken grup imzalarda imza atanın kimliğini sadece yöneticisi bilmektedir. Dijital imzaları herkes atabilirken bir grup imzasını sadece grup üyeleri atabilmektedir. İmzaların doğrulanması aşamasındaki temel farklılık, imzalayanın kimliğinin doğrulanıp doğrulanmadığıdır. Dijital imzayı alan bir kullanıcı imzalayanın kimliğini doğrularken grup imzasını doğrulayan bir kullanıcı, grup yöneticisinin imzalayanın kimliğini açığa çıkaracak tek kişi olduğundan emin olmakla birlikte imzalayanın sadece bir grup üyesi olduğunu doğrulayabilmektedir. Grup imzalar birden fazla kullanıcı ve bu kullanıcıları yöneten yöneticilerin olduğu, kullanıcılardan birisinin imza atmasının yeterli olduğu, imzaların taklit edilemediği, yönetici haricinde imzayı atan kullanıcının bilinmediği ancak gerekli olduğu durumlarda imza atan kullanıcının kimliğinin herkese açılması gerektiği tüm senaryolarda kullanım alanı bulmaktadır.

Grup imzalama şeması genel olarak bir kimlik doğrulama şeması, imzalama şeması ve bir şifreleme şemasının birleşiminden oluşmaktadır. Grup imzalama şemalarını oluşturmak için literatürde grubu oluşturan üye sayısı bakımından farklı modeller ve güvenlik gereksinimleri tanımlanmıştır. Öte yandan, temelde tüm şemalarda imzanın geçerli grup üyesi tarafından imzalandığı doğrulanırken imza atanın kimliği gizlenmektedir. Bunun yanında, imzayı atan grup üyesi sadece grup yöneticisi tarafından bilinirken herhangi bir kimsenin grup üyelerinin imzasını taklit etmesi de engellenmektedir.

Bu çalışmada, kafes tabanlı grup imzalama şemaları ele alınmıştır. Grup imzalama şemaları gruba yeni üyelerin eklenme veya gruptan üyelerin çıkarılma durumlarına göre sınıflandırılmıştır. Yapılan sınıflandırmaya göre grup imzalama şemalarının genel yapıları tanımlanarak, şemalarda kullanılan algoritmalardan bahsedilmiştir. Grup imzalama şemalarının oluşturmak için hangi bileşenlerin olması gerektiği üzerinde durulmuştur. Bunun yanında, literatürde yer alan şemalar incelenmiştir. Bu şemalar anahtar ve imza boyutları, sağladıkları güvenlik gereksinimleri, şemaları oluşturmada kullanılan alt sistemlere göre karşılaştırılmıştır. Gelecek çalışmalarda, kafes tabanlı grup imzalama şemalarında kullanılan kriptografik bileşenler ele alınacaktır. Bunun yanında, grup imzalama şemaları için farklı kullanım senaryoları ile uygulamaların geliştirilmesi planlanmaktadır.

Literatürdeki grup imzalama şemalarının ilk örneklerinde imza ve anahtar boyutları gruptaki üye sayısıyla doğrusal olarak arttığı bilinmektedir. Son önerilen şemalarda ise imza ve anahtar boyutları sabit olacak şekilde iyileştirilmiştir. Daha verimli grup imzalama şemalarının oluşturulup oluşturulmayacağı güncel bir araştırma konusudur. Şemalar incelendiğinde anahtar ve imza boyutlarının asimptotik olarak ifade edildiği görülmektedir. Anahtar ve imza boyutlarının asimptotik yerine bit olarak gerçek boyutları bakımından karşılaştırmasının verilmesi gerekmektedir. Bunun yanında var olan şemalar için SM ve ROM'da güvenlik ispatları yapılmaktadır. Önerilen şemalarda kullanılan alt imzalama sistemleri dışında farklı imzalama şemaları kullanılarak kuantum rastgele kahin modelinde (QROM) güvenli olacak şekilde grup imzalama şemalarının oluşturup oluşturulmayacağı açık problem olarak bırakılmıştır. Bunun yanında, NIST PQC projesine gönderilen imzalama şemaları kullanılarak anahtar ve imza boyutu açısından daha verimli grup imzalama şemalarının oluşturulması güncel bir araştırma konusudur. Öte yandan, kuantum sonrası süreç için grup imzalama şemalarının standartlaşması için yol haritasının çıkarılması gerekmektedir.

Teşekkür

Bu çalışma EEEAG-121R006 proje numarası ile TÜBİTAK tarafından desteklenmiştir.

KAYNAKLAR

[1]. Ş. Sağıroğlu and M. Alkan, "Her Yönüyle Elektronik İmza e-İmza, " Grafiker Yayınları, 2005.

[2]. P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring, " in Proceedings 35th annual symposium on foundations of computer science, 1994s, pp. 124–134.

[3]. "Google Quantum AI" https://quantumai.google/hardware, Son Erişim Tarihi: 1 Haziran 2021.

[4]. "D-Wave," https://www.dwavesys.com/quantum-computing, Son Erişim Tarihi: 1 Haziran 2021.

[5]. "IBM Quantum Computing" https://www.ibm.com/quantum-computing/, Son Erişim Tarihi: 1 Haziran 2021.

[6]. S. Akleylek and M. Soysaldi, "Kuantum Bilgisayarlar ile Kriptoanaliz ve Kuantum Sonrası Güvenilir Kripto Sistemleri," in *Siber Güvenlik ve Savunma: Farkındalık ve Caydırıcılık Cilt II*, M. e. Prof. Dr. Şeref Sağıroğlu, Ed. Grafiker Yayınları, 2019, pp. 137–168.

[7]. D. Chaum and E. van Heyst, "Group Signatures," in *Advances in Cryptology — EUROCRYPT '91*, D. W. Davies, Ed.Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, pp. 257–265.

[8]. L. Chen and T. P. Pedersen, "New group signature schemes," in *Advances in Cryptology — EUROCRYPT'94*, A. De Santis, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 171–181.

[9]. J. Camenisch, "Efficient and Generalized Group Signatures," in *Advances in Cryptology — EUROCRYPT '97*, W. Fumy, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 465–479.

[10]. J. Camenisch and M. Stadler, "Efficient group signature schemes for large groups," in *Advances in Cryptology — CRYPTO '97*, B. S. Kaliski, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 410–424.

[11]. J. Camenisch and A. Lysyanskaya, "Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials," in Advances in Cryptology — CRYPTO 2002, M. Yung, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 61–76.

[12]. G. Ateniese, D. Song, and G. Tsudik, "Quasi-Efficient Revocation of Group Signatures," Financial Cryptography, p. 183–197, 2003. [Online]. Available: https://link.springer.com/chapter/10.1007/3-540-36504-4_14

[13]. A. Kiayias and M. Yung, "Group Signatures with Efficient Concurrent Join," in Advances in Cryptology – EUROCRYPT 2005, R. Cramer, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 198–214.

[14]. G. Ateniese, J. Camenisch, S. Hohenberger, and B. De Medeiros, “Practical Group Signatures without Random Oracles,”IACR Cryptol. ePrint Arch., vol. 2005, p. 385, 2005.

[15]. X. Boyen and B. Waters, “Full-Domain Subgroup Hiding and Constant-Size Group Signatures,” in Public Key Cryptography – PKC 2007, T. Okamoto and X. Wang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 1–15.

[16]. J. Groth, “Fully Anonymous Group Signatures Without Random Oracles,” in Advances in Cryptology – ASIACRYPT 2007, K. Kurosawa, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 164–180.

[17]. M. Manulis, N. Fleischhacker, F. Günther, F. Kiefer, and B. Poettrering, “Group Signatures: Authentication With Privacy,” Bundesamt fur Sicherheit in der Informationstechnik, Bonn, Germany, Tech. Rep, 2012. [Online]. Available: https://nilsfleischhacker.de/publication/group-signatures-authentication-with-privacy/

[18]. M. N. S. Perera and T. Koshiba, “A Guests Managing System with Lattice-Based Verifier-Local Revocation Group Signature Scheme with Time-Bound Keys,” in Proceedings of the Fifth International Conference on Mathematics and Computing, D. Giri, A. T. S. Ho, S. Ponnusamy, and N.-W. Lo, Eds. Singapore: Springer Singapore, 2021, pp. 81–96.

[19]. S. Akleylek ve K. Seyhan, “Kuantum Bilgisayarlar Sonrası Güvenilir Kafes Tabanlı Kriptosistem Temellerine Giriş,” Siber Güvenlik ve Savunma: Farkındalık ve Caydırıcılık Cilt II, Şeref Sağıoğlu, Mustafa Şenol : Editörler. Grafiker Yayınları, 2019, pp. 171–209.

[20]. M. Bellare, D. Micciancio, and B. Warinschi, “Foundations of Group Signatures: Formal Definitions, Simplified Requirements, and a Construction Based on General Assumptions,” in Advances in Cryptology — EUROCRYPT 2003, E. Biham, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 614–629.

[21]. S. D. Gordon, J. Katz, and V. Vaikuntanathan, “A Group Signature Scheme from Lattice Assumptions,” in Advances in Cryptology - ASIACRYPT 2010, M. Abe, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 395–412.

[22]. C. Gentry, C. Peikert, and V. Vaikuntanathan, “How to Use a Short Basis: Trapdoors for Hard Lattices and New Cryptographic Constructions,” in Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, ser. STOC ’08. New York, NY, USA: Association for Computing Machinery, 2008, p. 197–206. [Online]. Available: https://doi.org/10.1145/1374376.1374407

[23]. O. Regev, “On Lattices, Learning with Errors, Random Linear Codes, and Cryptography,” J. ACM, vol. 56, no. 6, Sep. 2009. [Online]. Available: https://doi.org/10.1145/1568318.1568324

[24]. D. Micciancio and S. P. Vadhan, “Statistical Zero-Knowledge Proofs with Efficient Provers: Lattice Problems and More,” in Advances in Cryptology - CRYPTO 2003, D. Boneh, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 282–298.

[25]. J. Camenisch, G. Neven, and M. Rückert, "Fully Anonymous Attribute Tokens from Lattices," in Security and Cryptography for Networks, I. Visconti and R. De Prisco, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 57–75.

[26]. F. Laguillaumie, A. Langlois, B. Libert, and D. Stehlé, "Lattice-Based Group Signatures with Logarithmic Signature Size," in Advances in Cryptology - ASIACRYPT 2013, K. Sako and P. Sarkar, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 41–61.

[27]. X. Boyen, "Lattice Mixing and Vanishing Trapdoors: A Framework for Fully Secure Short Signatures and More," in Public Key Cryptography – PKC 2010, P. Q. Nguyen and D. Pointcheval, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 499–517.

[28]. V. Lyubashevsky, "Lattice-Based Identification Schemes Secure Under Active Attacks," in Public Key Cryptography – PKC 2008, R. Cramer, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 162–179.

[29]. S. Ling, K. Nguyen, and H. Wang, "Group Signatures from Lattices: Simpler, Tighter, Shorter, Ring-Based," in Public- Key Cryptography – PKC 2015, ser. Lecture Notes in Computer Science, J. Katz, Ed. Berlin, Heidelberg: Springer, 2015, pp. 427–449.

[30]. V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," Advances in Cryptology – EUROCRYPT 2010, p. 1–23, 2010. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3- 642-13190-5_1

[31]. S. Ling, K. Nguyen, D. Stehlé, and H. Wang, "Improved Zero-Knowledge Proofs of Knowledge for the ISIS Problem, and Applications," in Public-Key Cryptography – PKC 2013, K. Kurosawa and G. Hanaoka, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 107–124.

[32]. A. Langlois, S. Ling, K. Nguyen, and H. Wang, "Lattice-Based Group Signature Scheme with Verifier-Local Revocation," in Public-Key Cryptography – PKC 2014, ser. Lecture Notes in Computer Science, H. Krawczyk, Ed. Berlin, Heidelberg: Springer, 2014, pp. 345–361.

[33]. P. Q. Nguyen, J. Zhang, and Z. Zhang, "Simpler Efficient Group Signatures from Lattices," in Public-Key Cryptography – PKC 2015, ser. Lecture Notes in Computer Science, J. Katz, Ed. Berlin, Heidelberg: Springer, 2015, pp. 401–426.

[34]. S. Agrawal, D. Boneh, and X. Boyen, "Efficient Lattice (H)IBE in the Standard Model," in Advances in Cryptology – EUROCRYPT 2010, H. Gilbert, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 553–572.

[35]. V. Lyubashevsky, "Lattice Signatures without Trapdoors," in Advances in Cryptology – EUROCRYPT 2012, D. Pointc- heval and T. Johansson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 738–755.

[36]. D. Micciancio and P. Mol, "Pseudorandom Knapsacks and the Sample Complexity of LWE Search-to-Decision Reductions," in Advances in Cryptology – CRYPTO 2011, P. Rogaway, Ed.Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 465–484.

[37]. B. Libert, S. Ling, K. Nguyen, and H. Wang, "Zero-Knowledge Arguments for Lattice-Based Accumulators: Logarithmic- Size Ring Signatures and Group Signatures Without Trapdoors," in Advances in Cryptology – EUROCRYPT 2016, ser. Lecture Notes in Computer Science, M. Fischlin and J.-S. Coron, Eds. Berlin, Heidelberg: Springer, 2016, pp. 1–31.

[38]. R. C. Merkle, "A Certified Digital Signature," in Advances in Cryptology — CRYPTO' 89 Proceedings, G. Brassard, Ed. New York, NY: Springer New York, 1990, pp. 218–238.

[39]. J. Stern, "A new paradigm for public key identification," IEEE Trans. Inf. Theory, vol. 42, pp. 1757–1768, 1996.

[40]. M. Naor and M. Yung, "Public-Key Cryptosystems Provably Secure against Chosen Ciphertext Attacks," in Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing, ser. STOC '90. New York, NY, USA: Association for Computing Machinery, 1990, p. 427–437. [Online]. Available: https://doi.org/10.1145/100216.100273

[41]. B. Libert, F. Mouhartem, and K. Nguyen, "A Lattice-Based Group Signature Scheme with Message-Dependent Opening," in Applied Cryptography and Network Security, M. Manulis, A.-R. Sadeghi, and S. Schneider, Eds. Cham: Springer International Publishing, 2016, pp. 137–155.

[42]. A. Kawachi, K. Tanaka, and K. Xagawa, "Concurrently Secure Identification Schemes Based on the Worst-Case Hardness of Lattice Problems," in Advances in Cryptology - ASIACRYPT 2008, J. Pieprzyk, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 372–389.

[43]. S. Ling, K. Nguyen, H. Wang, and Y. Xu, "Forward-Secure Group Signatures from Lattices," in Post-Quantum Cryptography, J. Ding and R. Steinwandt, Eds. Cham: Springer International Publishing, 2019, pp. 44–64.

[44]. Q. Luo and C. Jiang, "A New Constant-Size Group Signature Scheme From Lattices," IEEE Access, vol. 8, pp. 10 198– 10 207, 2020.

[45]. S. Katsumata and S. Yamada, "Group Signatures Without NIZK: From Lattices in the Standard Model," Advances in Cryptology – EUROCRYPT 2019, pp. 312—344, 2019. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-17659-4_11

[46]. R. Tsabary, "An Equivalence Between Attribute-Based Signatures and Homomorphic Signatures, and New Constructions for Both," in Theory of Cryptography, Y. Kalai and L. Reyzin, Eds. Cham: Springer International Publishing, 2017, pp. 489–518.

[47]. P. Mohassel, "One-Time Signatures and Chameleon Hash Functions," in Selected Areas in Cryptography, A. Biryukov, G. Gong, and D. R. Stinson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 302–319.

[48]. S. Canard, A. Georgescu, G. Kaim, A. Roux-Langlois, and J. Traoré, "Constant-Size Lattice-Based Group Signature with Forward Security in the Standard Model," pp. 24–44, 2020.

[49]. T. Preethi and B. B. Amberker, "Lattice-based group signature scheme without random oracle," Information Security Journal: A Global Perspective, vol. 29, no. 6, pp. 366–381, 2020. [Online]. Available: https://doi.org/10.1080/19393555.2020.1777357

[50]. W. Fenghe and L. Zhenhua, "Short and provable secure lattice-based signature scheme in the standard model," Security and Communication Networks, vol. 9, no. 16, pp. 3627–3632, 2016. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/sec.1567

[51]. Q. Ye, X. Yang, X. Yan, and Z. Zhao, "Efficient Group Signature Scheme Over NTRU Lattice," in Cloud Computing and Security, X. Sun, Z. Pan, and E. Bertino, Eds. Cham: Springer International Publishing, 2018, pp. 554–562.

[52]. D. Boneh and H. Shacham, "Group Signatures with Verifier-Local Revocation," in Proceedings of the 11th ACM Conference on Computer and Communications Security, ser. CCS '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 168–177. [Online]. Available: https://doi.org/10.1145/1030083.1030106

[53]. D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert, "Bonsai Trees, or How to Delegate a Lattice Basis," in Advances in Cryptology – EUROCRYPT 2010, H. Gilbert, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 523–552.

[54]. A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in Odlyzko A.M. (eds) Advances in Cryptology — CRYPTO' 86. CRYPTO 1986. Lecture Notes in Computer Science, vol. 263. Springer, 1987, pp. 186–194.

[55]. S. Ling, K. Nguyen, A. Roux-Langlois, and H. Wang, "A lattice-based group signature scheme with verifier-local revocation," Theoretical Computer Science, vol. 730, p. 1–20, Jun 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0304397518302056

[56]. M. N. S. Perera and T. Koshiba, "Achieving Full Security for Lattice-Based Group Signatures with Verifier-Local Revocation," in Information and Communications Security, D. Naccache, S. Xu, S. Qing, P. Samarati, G. Blanc, R. Lu, Z. Zhang, and A. Meddahi, Eds. Cham: Springer International Publishing, 2018, pp. 287–302.

[57]. T. Koshiba and M. N. S. Perera, "Zero-Knowledge Proof for Lattice-Based Group Signature Schemes with Verifier-Local Revocation," in Advances in Network-Based Information Systems, L. Barolli, N. Kryvinska, T. Enokido, and M. Takizawa, Eds. Cham: Springer International Publishing, 2019, pp. 772–782.

[58]. Y. Zhang, X. Liu, Y. Hu, Q. Zhang, and H. Jia, "Lattice-Based Group Signatures with Verifier-Local Revocation: Achieving Shorter Key-Sizes and Explicit Traceability with Ease," in Cryptology and Network Security, Y. Mu, R. H. Deng, and X. Huang, Eds. Cham: Springer International Publishing, 2019, pp. 120–140.

[59]. A. Kitagawa, Y. Sakai, K. Emura, G. Hanaoka, and K. Tanaka, "Fully Anonymous Group Signature with Verifier-Local Revocation," Cryptology ePrint Archive, Report 2021/170, 2021, https://eprint.iacr.org/2021/170.

[60]. Y. Zhang, Y. Gan, Y. Yin, and H. Jia, "Attribute-Based VLR Group Signature Scheme from Lattices," Algorithms and Architectures for Parallel Processing, p. 600–610, 2018. [Online]. Available: https://www.springerprofessional.de/en/attribute-based-vlr-group-signature-scheme-from-lattices/16321222

[61]. Y. Zhang, X. Liu, Y. Yin, Q. Zhang, and H. Jia, "On New Zero-Knowledge Proofs for Fully Anonymous Lattice-Based Group Signature Scheme with Verifier-Local Revocation," Lecture Notes in Computer Science, pp. 381–399, 2020. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-61638-0_21

[62]. M. Bellare, H. Shi, and C. Zhang, "Foundations of Group Signatures: The Case of Dynamic Groups," in Topics in Cryptology – CT-RSA 2005, A. Menezes, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 136–153.

[63]. A. Kiayias and M. Yung, "Secure Scalable Group Signature with Dynamic Joins and Separable Authorities," Int. J. Secur. Netw., vol. 1, no. 1/2, pp. 24–45, Sep. 2006. [Online]. Available: https://doi.org/10.1504/IJSN.2006.010821

[64]. B. Libert, S. Ling, F. Mouhartem, K. Nguyen, and H. Wang, "Signature Schemes with Efficient Protocols and Dynamic Group Signatures from Lattice Assumptions," in Advances in Cryptology – ASIACRYPT 2016, J. H. Cheon and T. Takagi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 373–403.

[65]. F. Böhl, D. Hofheinz, T. Jager, J. Koch, and C. Striecks, "Confined Guessing: New Signatures From Standard Assumptions," Journal of Cryptology, vol. 28, no. 1, p. 176–208, Apr 2014. [Online]. Available: https://link.springer.com/article/10.1007/s00145-014-9183-z

[66]. S. Ling, K. Nguyen, H. Wang, and Y. Xu, "Constant-Size Group Signatures from Lattices," in Public-Key Cryptography – PKC 2018, ser. Lecture Notes in Computer Science, M. Abdalla and R. Dahab, Eds. Cham: Springer International Publishing, 2018, pp. 58–88.

[67]. L. Ducas and D. Micciancio, "Improved Short Lattice Signatures in the Standard Model," in Advances in Cryptology – CRYPTO 2014, J. A. Garay and R. Gennaro, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 335–352.

[68]. J. Bootle, A. Cerulli, P. Chaidos, E. Ghadafi, and J. Groth, "Foundations of Fully Dynamic Group Signatures," in Applied Cryptography and Network Security, M. Manulis, A.-R. Sadeghi, and S. Schneider, Eds. Cham: Springer International Publishing, 2016, pp. 117–136.

[69]. S. Ling, K. Nguyen, H. Wang, and Y. Xu, "Lattice-based group signatures: Achieving full dynamicity with ease," in International Conference on Applied Cryptography and Network Security. Springer, 2017, pp. 293–312.

[70]. A. Ishida, K. Emura, G. Hanaoka, Y. Sakai, and K. Tanaka, "Group Signature with Deniability: How to Disavow a Signature," in Cryptology and Network Security, S. Foresti and G. Persiano, Eds. Cham: Springer International Publishing, 2016, pp. 228–244.

[71]. S. Ling, K. Nguyen, H. Wang, and Y. Xu, "Lattice-based group signatures: Achieving full dynamicity (and deniability) with ease," Theoretical Computer Science, vol. 783, p. 71–94, Sep 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0304397519301884

[72]. M. N. S. Perera and T. Koshiba, "Achieving Almost-Full Security for Lattice-Based Fully Dynamic Group Signatures with Verifier-Local Revocation," in Information Security Practice and Experience, C. Su and H. Kikuchi, Eds. Cham: Springer International Publishing, 2018, pp. 229–247.

[73]. D. Naor, A. Shenhav, and A. Wool, "One-Time Signatures Revisited: Have They Become Practical?" IACR Cryptol. ePrint Arch., vol. 2005, p. 442, 2005.

[74]. M. N. S. Perera and T. Koshiba, "Achieving Strong Security and Verifier-Local Revocation for Dynamic Group Signatures from Lattice Assumptions," in Security and Trust Management, S. K. Katsikas and C. Alcaraz, Eds. Cham: Springer International Publishing, 2018, pp. 3–19.

Bölüm 12

YAYIN ŞİFRELEME SİSTEMLERİ

Hüseyin Bodur - Resul Kara

Yayın şifreleme şemaları yayın iletiminde büyük bir öneme sahiptir. Çünkü şema anahtarıyla şifrelenmiş bir mesaja, bir yayın şemasında bulunan yetkili kullanıcıların dışında yetkisi olmayan kullanıcıların erişememesi, erişebilse bile bir anlam çıkartamaması gerekir. Bu bölümde, bir kaynaktan çoklu kullanıcılara doğru yapılan yayın iletimlerinde kullanılan iletişim şemaları ele alınmıştır. Bu şemalar ön tanımlı ve dinamik olmak üzere ikiye ayrılmış, şemalar üzerinde güvenli yayın iletimi ile anahtar dağıtımının nasıl yapıldığı detaylı olarak açıklanmıştır.

12.1. GİRİŞ

Yayın veya yayın yapma, bir göndericiden bir veya daha fazla alıcıya metin, ses veya görüntülerin transfer edilmesi anlamında kullanılmaktadır. Yıllara bakıldığında;

- 1895 yılında Marconi'nin telsizi icadı ile radyo dalgalarının bir noktadan başka bir noktaya aktarılabileceği fikri üzerinde çeşitli bilimsel çalışmalar
- 1905 yılında ilk deneysel radyo yayını uygulaması
- 1920'li yılların başlarında ticari radyo yayınları denemeleri

- 1950'li yıllardan itibaren gelişmiş ülkelerde devlet eliyle işletilen veya özel kuruluşlarca işletilen radyolar
- Transistörün icadından sonra radyo ve televizyon alıcılarının maliyetlerinin düşmesi ve boyutlarının küçülmesinin ardından tüm dünyada yayıncılık alanında bir ivmelenme

görülmektedir.

Yayınların ödemeli olarak yapılması fikri;

- 1980'li yılların sonlarında gündeme gelmiş,
- Müzik ve spor müsabakalarının ödemeli sistemlerle yapılması "şifreli yayın sistemi" veya "yayın şifreleme" kavramının oluşmasını sağlamış,
- Başlangıçta analog yayınların şifreli olarak alıcılara ulaştırılmasıyla başlayan ödemeli sistem, sayısal yayın sistemlerinin geliştirilmesi ile yeni bir boyut kazanmış,
- 2000'li yılların başından itibaren ise dünyada ödemeli yayın sistemleri alanında çok sayıda örnek oluşmuştur.

Yayın şifreleme sistemleri genel itibariyle bir mesajın bir kaynaktan belirli bir kullanıcı grubuna güvenli bir şekilde iletilmesini amaçlar. Kullanıcılar yayın erişimine sahip yetkili bir gruptur. Yetkisiz kullanıcılar erişseler bile ilgili yayından anlamlı bir mesaj elde edememelidir. Yayın iletimini sadece belirli bir kullanıcı grubunun erişimine sahip olmasını sağlamak, iletim, hesaplama ve saklama maliyeti gibi çeşitli maliyetlerin ortaya çıkmasına neden olur.

Başarılı bir yayın iletimi için;

- Yayın iletimi sorunsuz olmalıdır. Özellikle giriş-çıkışların sürekli olduğu dinamik kullanıcı gruplarında gruba katılan kullanıcıların yayın erişimlerinin hemen sağlanması, ayrılan kullanıcıların yayın erişimlerinin ise hemen kısıtlanması gerekir.
- Yayın iletimi sırasında güvenlik zafiyetleri oluşmamalıdır.
- Yayın iletimi düşük maliyetli olmalıdır.

Tüm bu hususlar dikkate alındığında, başarılı bir yayın iletiminin gerçekleştirilmesi için çeşitli yayın şifreleme şemalarından yararlanılır. Yayın şifreleme şemaları kısaca, içlerinde şifreleme yöntemlerinin kullanıldığı farklı topolojik

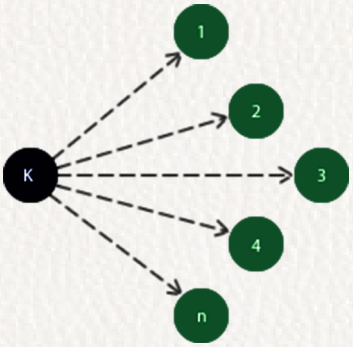
özelliklere sahip yapılardır. Yayın şifreleme şemalarının temelinde yer alan güvenli grup iletişimi günümüzde video konferans, çevrimiçi oyunlar, e-sağlık, askeri iletişim ve internet protokol TV (IPTV) gibi alanlarda sıklıkla kullanılmaktadır [1] [2] [3] [4]. Akıllı telefon ve tablet gibi cihazlar özellikle video konferans, çevrimiçi oyunlar ve e-sağlık gibi alanlara olan ilgiyi arttırmıştır. Güvenli grup iletişimi ayrıca saklama, hesaplama, iletişim ve enerji gibi açılardan sınırlı kapasiteye sahip cihazların yer aldığı nesnelerin interneti tabanlı çalışmalarda güvenli iletişimin sağlanması için kullanılmaktadır [5] [6].

12.2. YAYIN ŞİFRELEME

Yayın şifreleme; bir yayın kanalı üzerinden gönderilen verinin yalnızca yetkili bir kullanıcı kümesi için anlamlı hale gelmesini, yetkisiz kullanıcıların ilgili yayından anlamlı bir mesaj elde edememesini amaçlayan kriptografik bir konudur. 1991 yılında Shimson Berkovits tarafından başlatılan yayın şifreleme [7], 1993 yılında Amos Fiat ve Moni Naor tarafından ciddi bir araştırma konusu haline getirilmiştir [8].

Yayın şifreleme, yetkili bir kullanıcı kümesi için, güvenli bir yayın iletiminin nasıl yapılacağı problemine çözüm getirir. Kullanıcı kümesi genellikle sürekli güncellenebilen, dinamik bir yapıya sahiptir.

Şekil 12.1'de görüldüğü üzere, yayın iletimi kaynaktan hedef kullanıcılara olmak üzere tek yönlüdür. Kaynaktan hedefe tek yönlü iletilen bir veri, yalnızca yetkili kullanıcılar için anlamlı halde olmalı, yetkisi olmayan kullanıcılar yayın verisine erişseler bile anlamlı mesajı elde edememelidir.



Şekil 12.1. Kaynak (K)'dan Hedef Kullanıcılara Tek Yönlü, Eş Zamanlı Yayın İletimi

Yayın iletimi birçok farklı alan ve uygulama içerisinde kullanım oranına sahiptir. Bir radyo sinyalinin belirli bir frekansta bir kaynaktan, radyo alıcısına sahip kişilere iletilmesiyle, ilgili yayının dinlenebilmesi yayın iletimine bir örnektir. Bir diğer örnek olarak, uzay tabanlı uydu navigasyon sistemi olan ve dünya üzerinde yer tespiti yapılmasını sağlayan GPS verilebilir.

Yayın şifreleme konusuna verilebilecek en iyi örnek ise dijital yayınlardır. Bir kullanıcı ilgili yayını yalnızca belirli bir miktar para ödeyip abone olarak takip edebilmeli, abone olmayan kullanıcılar dijital yayından yararlanamamalıdır. Bu nedenle yayın güvenliğinin sağlanması ve belirli kullanıcı gruplarına kısıtlanması gerekir.

12.3. YAYIN ŞİFRELEME ŞEMALARI

Bir yayın şifreleme şemasında, içerik mesajı çoklu kullanıcılar onu çözsün diye şifrelenir. Bir mesajı çoklu olarak kullanıcılara göndermek için öncelikle kullanıcıları gruplamak gerekir. Kullanıcı gruplama bir diğer ifadeyle yetkili-yetkisiz kullanıcı kümesi oluşturma işlemi ön tanımlı olarak yapılabileceği gibi dinamik olarak da yapılabilir. Bu bölümde ön tanımlı ve dinamik şemalar açıklanacaktır.

12.3.1. Ön Tanımlı Şemalar

Ön tanımlı yayın şemalarının özelliği, yayın iletimi başlamadan önce içerisinde bir başlangıç adımının bulunmasıdır. Başlangıç adımında kullanıcının şifreli mesajı çözmek için kullanacağı anahtar değeri kullanıcı cihazına yüklenir. Bu işlemin ardından yayın mesaj iletimine geçilebilir. Kullanıcıda saklanan şifre çözme anahtarında herhangi bir güncelleme yapılmaz. Şemalar bu nedenle durumsuz olarak adlandırılır.

Ön tanımlı şemalara örnek olarak Tam Alt Ağaç ve Alt Küme Farkı şemaları verilebilir. Her iki şemada da yayın iletimi başlamadan önce yetkili ve yetkisiz kullanıcı kümeleri belirlenmiş, yetkili kullanıcılara yayın mesaj şifresini çözmek için kullanacakları anahtar bilgileri gönderilmiştir.

Her iki şemada kullanıcılarda bulunan gizli bilgi üzerinde bir değişiklik olmadan kullanıcı erişim hakları üzerinde değişikliğe izin verirler. Daha açık

bir ifadeyle, yetkili kullanıcılar kümesi içerisinde bir kişinin yetkisinin elinden alınması (örneğin; abonelikten çıkartılması) ve bunun sonucu olarak ilgili yayını takip etmemesi istenebilir. Bu durumda kullanıcıya başlangıç adımında gönderilen, kullanıcının cihazında bulunan ve yayın mesajını çözme işlemi için kullanılan anahtar bilgisi üzerinde bir değişiklik yapılmadan, yalnızca ağaç yapısı üzerinde erişim kısıtlaması yapılarak yayın engellenebilir.

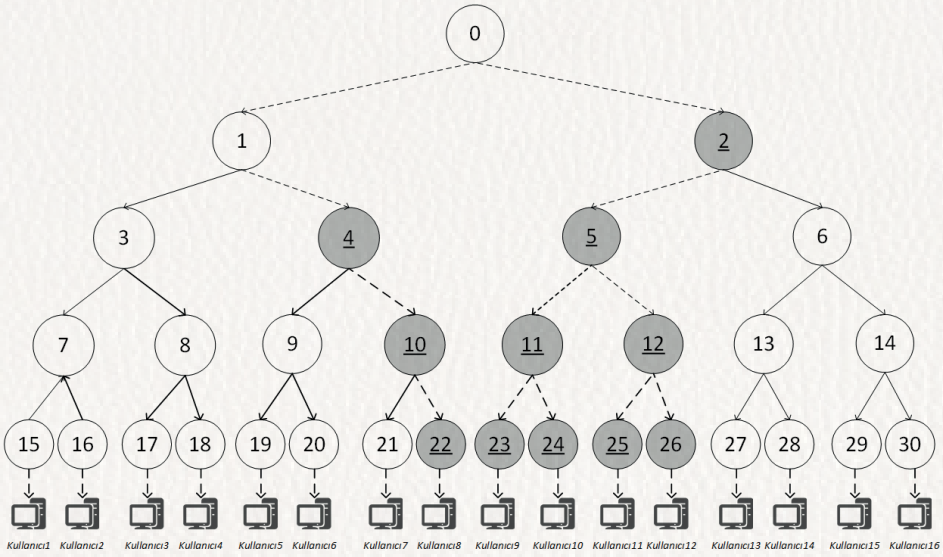
Tam Alt Ağaç

Naor ve arkadaşları tarafından önerilen Tam Alt Ağaç (TAA) şemasında kullanıcılar yapraklarda bulunur [9]. Şema üzerinde bir altküme olan S_i, i'inci düğümü ve o düğüm altındaki tüm düğümleri içerir. Şekil 12.2'de görüldüğü üzere S_3, S_9, S_{21}, S_6 altkümeleri yetkili, diğer altkümeler yetkisizdir. Yetkili altkümeler altındaki tüm kullanıcılar şifreli yayın mesajını çözebilme yetkisine sahiptir. TAA üzerindeki n sayıda düğüm için ağaçtaki altküme sayısı $2n-1$'dir. Ağaç üzerindeki tüm düğümlerin indis değerleri ise n adet yaprak için $\{0,..,2n-2\}$ aralığındadır.

Yayın iletimi başlamadan önce, ağaç üzerindeki her yetkili altküme için rastgele anahtar üretilmelidir. Ardından ağaçta bulunan her kullanıcıya özel bir kanal üzerinden gizli bilgiler gönderilmelidir. Güvenlik sorunlarına yol açmaması için bilgiler, örneğin dijital bir tv yayınında alıcının kullanacağı akıllı kartı üzerine, üretim aşamasında yüklenebilir. Bir kullanıcı, kendisine gönderilen gizli bilgileri kullanarak altkümelere ait gizli anahtar değerlerini elde edebilmelidir.

Bir şifreli mesajı yalnızca ağaç üzerinde bulunan yetkili kullanıcılar çözebilmelidir. Her kullanıcı bir altkümenin üyesidir ve o altkümenin anahtar değerine sahiptir. Şifreleme işlem adımları aşağıdaki gibidir.

1) Rastgele bir oturum anahtarı "K" üretilir ve mesaj bu anahtar değeri ile şifrelenir: $F_K(M)$.
2) Her altküme için o alt kümenin anahtarı "L_i" ile "K" değeri şifrelenir: $E_{L_i}(K)$.
3) Gönderilecek mesajın içerik kısmına şifreli mesaj, baş kısmına ise indis değerleriyle birlikte şifreli anahtar değerleri eklenir ve kullanıcılara iletilir.



Şekil 12.2. Örnek Bir TAA Şeması

Yalnızca yetkili kullanıcılar oturum anahtarını çözebileceklerdir, çünkü yetkisiz kullanıcılar bir altküme anahtarına sahip değildirler.

Şifre çözme işlem adımları ise aşağıdaki gibidir.

1) Kullanıcının bulunduğu S_i altkümesine gelen mesajın başlık kısmı aranır, "i" 'inci indis değerine sahip şifreli anahtar değeri elde edilir.
2) S_i'inci altkümenin gizli anahtar değeri "L_i" kullanılarak mesajın başlık kısmından alınan şifreli anahtar değeri çözülür ve "K" değeri elde edilir: $E_{L_i}^{-1}\left(E_{L_i}(K)\right) = K$.
3) "K" değeri kullanılarak mesaj çözülür: $F_K^{-1}\left(F_K(M)\right) = M$.

Alt Küme Farkı

Alt Küme Farkı (AKF), TAA şemasına benzerlik gösterir. TAA şeması gibi, AKF şemasını da Naor ve arkadaşları önermiştir [9]. Kullanıcılar ikili ağaç yapısı üzerinde yaprak düğümlerde bulunur. Bir kullanıcı birden fazla altkümeye ait olabilir. Şema üzerindeki altkümeler iki düğüm tarafından tanımlanır. Örneğin altküme $S_{i,j}$'de yetkili kullanıcılar kümesi $S_{i,j} = S_i \backslash S_j$'dir. Daha

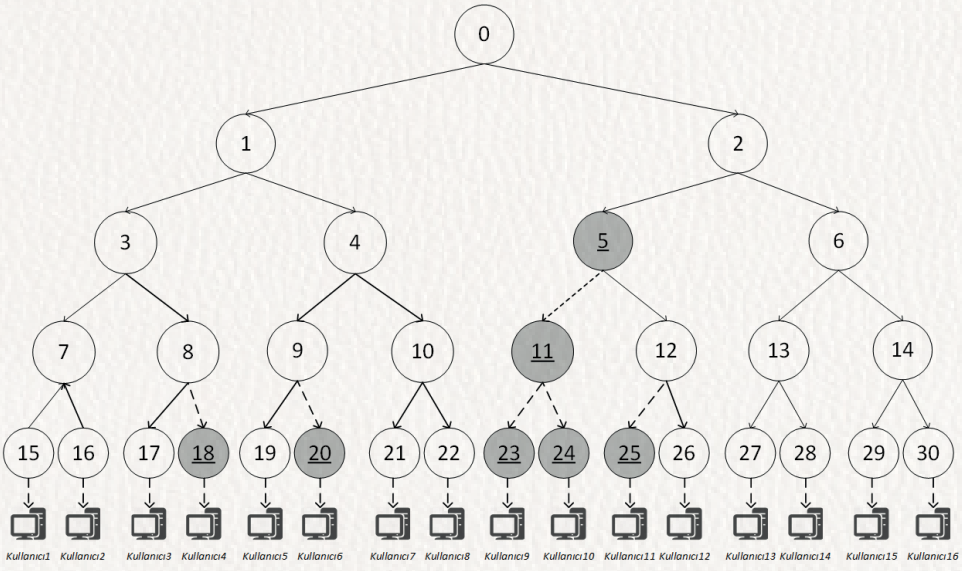
açık bir ifadeyle $S_{i,j}$ altkümesi, j düğümü ve bu düğümün altkümeleri hariç i düğümü ve i'nin tüm alt kümelerini temsil eder.

Şekil 12.3'de görüldüğü üzere, 5, 11, 18, 20, 23, 24, 25 numaralı altkümeler yetkisizdir. Yetkili kullanıcı altkümeleri ise şu şekildedir: $S_{3,18}, S_{9,20}, S_{2,5}, S_{12,25}$.

Örneğin;

$S_{3,18}$: 18. düğüm ve bu düğümün (varsa) altkümeleri hariç, 3. düğüm ve bu düğümün tüm altkümelerinin yetkili olduğu anlamına gelir.

Anahtarların doğrudan kullanıcılarda saklanması mümkündür. Ağaç üzerinde her düğüm bir etiketle ilişkilidir. Bir etiket, şemanın başlangıç aşamasında oluşturulan rastgele bir bit dizisidir. Tüm olası altküme anahtarları birkaç etiket kullanılarak türetilebilir.



Şekil 12.3. Örnek Bir AKF Şeması

Altküme anahtarlarını türetmek için, bir rastgele dizi üretici G kullanılır: $G: \{0,1\}^n \rightarrow \{0,1\}^{3n}$.

Üreteç tek yönlü güçlü bir fonksiyondur. Üç ayrı fonksiyonun kombinasyonu olarak da düşünülebilir: G_L, G_R ve G_M. Bu üç fonksiyon ağaç üzerinde uygulanarak anahtar değeri elde edilir.

G fonksiyonunun çıktısı sol, sağ ve orta olmak üzere üç parçaya bölünür. Sol ve sağ parçalar ara etiket olarak adlandırılır ve $I_{i,j}$ ile gösterilir.

Bir altküme anahtarı üretildiğinde, üreteç yinelenen bir davranışla bir etikete uygulanır. Şekil 12.4 üzerinden bir anahtar üretme prosedürü gösterilmiştir.

Şekil 12.4'de görüldüğü üzere, $S_{2,12}$ altkümesinin anahtarı $L_{2,12}$'yi üretmek için 2. düğümün etiketi I_2, kullanılarak işleme başlanır.

5.düğüm 2.düğümün sol çocuğu olduğundan G üretecinin sol fonksiyonu G_L'ye I_2 sokulur ve $I_{2,5}$ elde edilir: $G_L(I_2) = I_{2,5}$.

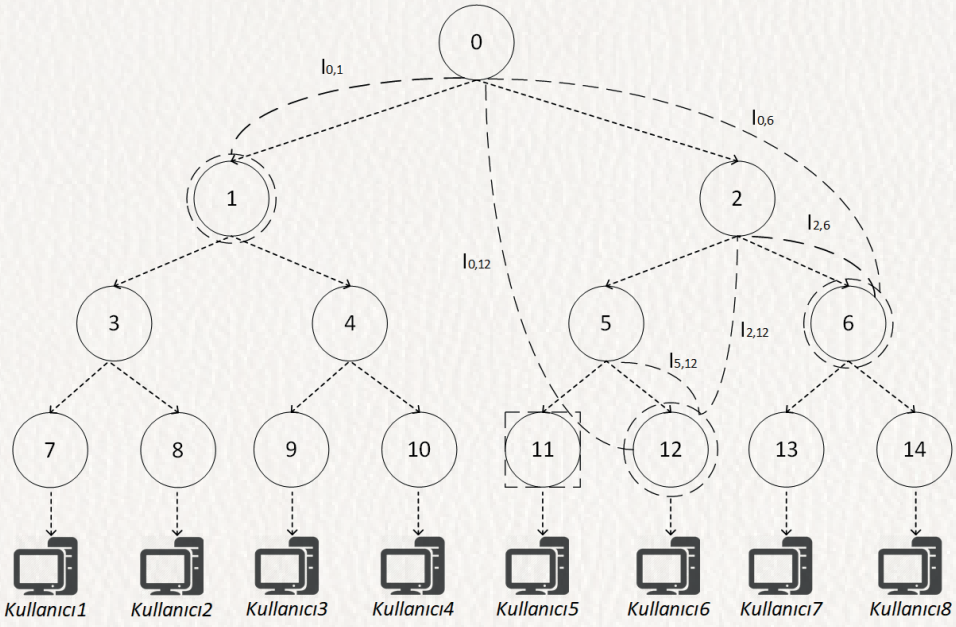
Bir sonraki adımda 12. düğüm ile 5. düğümün sağ çocuğu olduğundan G üretecinin sağ fonksiyonu G_R'ye $I_{2,5}$ sokulur ve $I_{2,12}$ elde edilir: $G_R(I_{2,5}) = I_{2,12}$.

Elde edilen değer her anahtar üretme işleminin son aşaması olan G üretecinin diğer fonksiyonu G_M'ye $I_{2,12}$ sokulur ve anahtar değeri $L_{2,12}$ elde edilir: $G_M(I_{2,12}) = L_{2,12}$.

Bir etiket tüm altküme anahtarlarını üretmek için kullanılabileceğinden, alıcıda saklanacak bilgi miktarını azalmış olur.

AKF, TAA şemasına göre daha karmaşıktır. Bir anahtarın üretilmesi için ilgili $S_{i,j}$ altkümesinin sağ ve sol durumlarına göre G_L, G_R kullanılarak işlem yapılır.

En son yaprakları olmayan düğüm üzerinde (son düğüm olan j) G_M işlemi uygulanarak altküme anahtar değeri elde edilir. TAA şemasında anahtar boyutu $O(log(N))$ iken, AKF şemasında $O(N)$'dir.



Şekil 12.4. $S_{2,12}$ Alt Kümesi

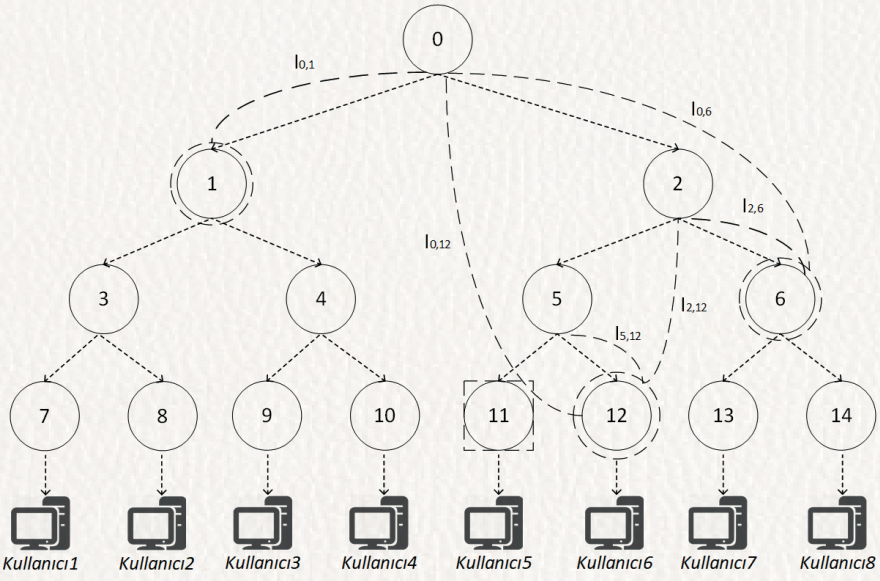
Ağaç üzerinde altküme anahtar değeri oluşturulmaması gereken kullanıcılara ilgili etiket değeri gönderilmemelidir. Yani $S_{i,j}$ altkümesinde j düğümü ve j düğümünün altkümeleri yetkisiz kullanıcılar ise bu düğüme bulunduğu altkümenin anahtarını elde etmesini sağlayacak I_i etiketi gönderilmemelidir.

Şekil 12.3'e bakacak olursak $S_{9,20}$'nin 20. düğümü yetkisizdir. Bu nedenle bu düğüme I_{20} etiketi gönderilmemelidir.

Ağaç üzerindeki tüm düğümlerin etiketleri rastgele seçmeli, ardından kullanıcılara bulundukları altkümelerin anahtarlarını türetebilmeleri için ihtiyaç duydukları etiketler gönderilmelidir. Etiketler kullanıcılara güvenli bir yolla iletilmelidir. Eğer kullanıcılara etiket değerleri gönderilmezse kullanıcılar altküme anahtarlarını türetemezler. Şekil 12.5'de 11 numaralı kullanıcı düğümün ihtiyaç duyduğu etiket değerleri gösterilmektedir.

AKF şemasında şifreleme işlemi TAA şemasına oldukça benzer. Şifreleme işlemi işlem adımları aşağıdaki gibidir:

1) Rastgele bir oturum anahtarı *"K"* üretilir ve mesaj bu anahtar değeri ile şifrelenir: $F_K(M)$.
2) Her altküme $S_{i,j}$ için o altkümenin gizli anahtarı "$L_{i,j}$" ile *"K"* değerini şifrelenir: $E_{L_{i,j}}(K)$.
3) Gönderilecek mesajın içerik kısmına şifreli mesaj, baş kısmına ise indis değerleriyle birlikte şifreli anahtar değerleri eklenir ve alıcılara iletilir.



Şekil 12.5. 11. Düğümün İhtiyaç Duyduğu Etiket Değerleri

Şifre çözme işlem adımları ise aşağıdaki gibidir:

1) Kullanıcının bulunduğu $S_{i,j}$ alt kümesine gelen mesajın başlık kısmı aranır.
2) Altküme anahtarını türetmek için, gizli bilgi içindeki ara etiket değeri bulunur.
3) Ara etiketten altküme anahtarı "$L_{i,j}$" türetilir.
4) "$L_{i,j}$" kullanılarak mesajın başlık kısmından alınan şifreli anahtar değeri çözülür ve *"K"* değeri elde edilir: $E_{L_{i,j}}^{-1}\left(E_{L_{i,j}}(K)\right) = K$.
5) *"K"* değeri kullanılarak mesaj çözülür: $F_K^{-1}\left(F_K(M)\right) = M$.

12.3.2. Dinamik Şemalar

Dinamik şemalarda, ön tanımlı şemaların aksine, yetkili kullanıcı kümesine bir kullanıcı eklendiğinde veya çıkartıldığında gizli anahtarlar değerleri güncellenir. Bu nedenle kullanıcılar, herhangi bir anahtar güncelleme mesajını kaçırmamak için yayın ağına sürekli bağlı olmalıdırlar. Dinamik bir şema iki ana bileşenden oluşur. Bu bileşenler sırasıyla Grup Anahtar Yönetimi (GAY) ve Grup Üyelik Yönetimi (GÜY)'dir.

GAY, şemada bulunan grup üyeleri arasında ortak gizli bir anahtar oluşturulmasını sağlar. Ortak gizli anahtar mesaj güvenliği açısından büyük bir öneme sahiptir. Grup iletişiminde mesajları şifrelemek için kullanılır.

Bir yayın şemasının dayanıklılığı, içerisinde kullanılan anahtar yönetim protokolüne ve ortak gizli anahtarın boyutuna bağlıdır. Anahtar yönetim protokolü bir grup anahtarının nasıl üretileceği, dağıtılacağı ve güncelleneceğini belirler. İleri ve geri gizliliğin sağlanması için ortak gizli anahtarın her üyelik işleminin ardından güncellenmesi gerekir.

GÜY bir kullanıcının gruba katılma ve ayrılma işlemlerini tanımlar. Gruba katılma işleminde ilk olarak kimlik doğrulama işleminin yapılması gerekir. Grup içerisindeki mesajlara yalnızca kimliği doğrulanmış kullanıcılar erişebilmelidir.

Dinamik bir şemanın iletişim gereksinimleri olarak adlandırılan bazı gereksinimlere sahip olması gerekir. Bu gereksinimler aşağıda listelenmiştir.

İleri Gizlilik: Yayın ortamından ayrılan bir kullanıcının gelecek yayın mesajları çözmesini engellemektir.

Geri Gizlilik: Yayın ortamına eklenen bir kullanıcının geçmiş mesajları çözmesini engellemektir.

Kimlik doğrulama: Şemada, grup kullanıcılarına erişim izni verilmeden önce kimlik doğrulaması yapılmalıdır. Aksi halde kimlik tabanlı ataklara maruz kalınabilir [10].

Grup Mesaj Bütünlüğü: Mesaj bütünlüğü korunmalı, kimliği doğrulanmamış varlıklar tarafından mesajın bir kısmı veya tamamı üzerinde ekleme, silme ve düzeltme yapılmasına izin verilmemelidir [10] [11].

Grup Mesaj Gizliliği: Sadece kimliği doğrulanmış yetkili kullanıcılar şifreli veriden anlamlı mesajı elde edebilmelidir. Gizlilik bir grup anahtarı ile şifreleme yapılarak sağlanabilir. Mesaj içeriğinin önemine göre, grup iletişimindeki mesajların bir kısmı güçlü şifreleme anahtarları ile şifrelenirken, bir kısmı daha zayıf anahtarlar ile şifrelenebilir [11].

Gizli Anlaşmaya Dayanıklılık: Bir saldırgan grupta bulunan bir üye ile anlaşma sağlayarak grup anahtarını elde edebilir ve gizli verilere erişim sağlayabilir. Bir şemada düğümler sürekli kontrol edilmeli, gizli anlaşma yapan bir düğüm tespit edildiğinde hemen gruptan çıkartılmalıdır [11].

Yeniden Anahtarlama: Her üyelik değişimi sonunda ortak gizli anahtar başta olmak üzere bazı anahtarların güncellenmesi gerekir. Güncelleme işlemi mümkün olduğunca hızlı yapılmalıdır. Aksi halde, ayrılan kullanıcılar grup anahtarı güncellenene kadar grup mesajlarını elde edebilir. Eklenen kullanıcılar ise grup iletişimine hemen dahil olamayabilir [11].

Hizmet devamlılığı: Dinamik bir şemada grup işlemleri sırasında ortaya çıkan tek bir sorun grup iletişim işlemini etkilememelidir [12] [13].

Ölçeklenebilirlik: İster küçük ister büyük bir kullanıcı grubu olsun bir şemada anahtar yönetim, güvenlik ve etkililik özellikleri sağlanmalıdır. Grup yönetim veya üyelik işlemlerinin ardından güncellenen ortak gizli anahtarın ölçülebilir bir gecikme, kabul edilebilir bir hesaplama ve iletişim maliyetiyle iletimi sağlanmalıdır [12] [13] [14].

Güvenilirlik: Dinamik bir şemada güncel anahtarların dağıtımı güvenli olmalıdır. Gruptaki tüm üyelere anahtarlar güvenli bir yoldan zamanında dağıtılmalıdır [12] [13].

Esneklik: Bir şema farklı uygulamalar içerisinde kullanılabilir olmalıdır. Kullanıcı ekleme-çıkarma işlemleri herhangi bir zamanda yapılabilmelidir [11] [15].

Bu bölümde literatürde bulunan dinamik şemalar merkezi, dağıtık ve birleşik olmak üzere üç kategoride sınıflandırılmış ve detaylı olarak açıklanmıştır. Bu şemalar arasındaki temel fark Grup Yöneticisi (GY) olarak adlandırılan bir varlığın tanımıdır. GY, GAY işlemlerini gerçekleştiren varlıktır. Merkezi ve birleşik şemalarda GY bulunurken, dağıtık şemalarda GY bulunmaz. Dağıtık şemalarda GY'nin görevini geçici bir süreliğine bir kullanıcı ya da kullanıcı grubu üstlenir.

12.3.2.1. Merkezi Güvenli Grup İletişim Şemaları

Merkezi grup anahtar yönetim şemalarında, GY olarak isimlendirilen bir merkezi güvenli varlık bulunur. Bir GY, grup içerisinde bulunan anahtarların üretiminden, dağıtımından ve güncellenmesinden sorumludur. Merkezi grup anahtar yönetim şemalarında genellikle simetrik şifreleme yöntemlerinden yararlanılır.

İkili Anahtarlar

GY, gruptaki her kullanıcıyla birebir iletişim kurar. Her kullanıcının kendisine ait bir gizli anahtarı bulunur. GY bu gizli anahtarı kullanarak kullanıcılara güncel anahtarı iletir. Gruptan bir kullanıcı ayrıldığında anahtarın güncellenmesi gerekir. Bunun için GY her kullanıcı ile birebir iletişime geçmeli ve kullanıcının gizli anahtarını kullanarak kendisine güncel anahtarı iletmelidir. Bu yöntemde geri gizlilik sağlanırken, ileri gizlilik için O(n) yeniden anahtarlama mesajı söz konusudur. Bu nedenle bu yöntem geniş ve dinamik gruplar için uygun değildir.

Mantıksal Anahtar Hiyerarşisi

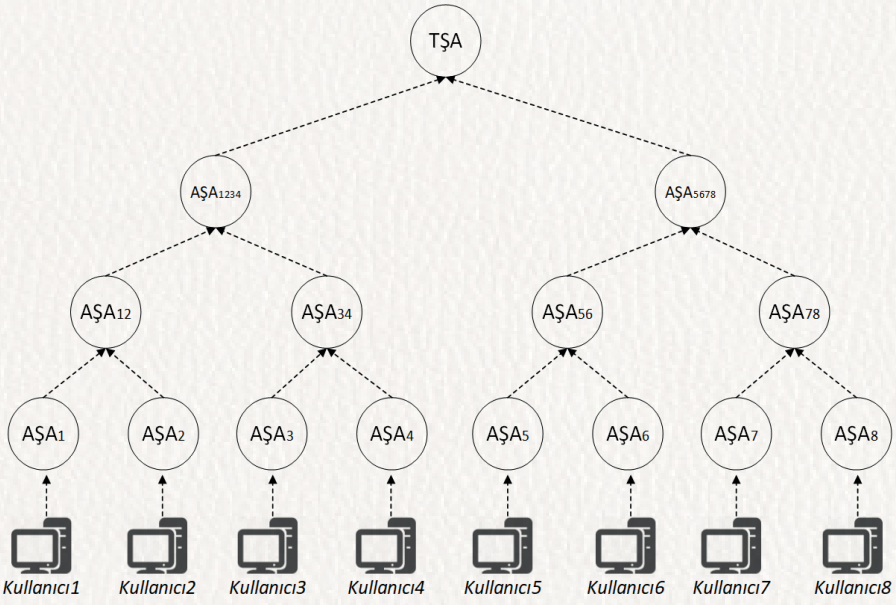
Mantıksal Anahtar Hiyerarşisi (MAH) (Logical Key Hierarchy), birbirlerinden bağımsız iki araştırma grubu olan Wong ve arkadaşları [16] ile Waller ve arkadaşları [17] tarafından yaklaşık olarak aynı zamanda önerilmiştir. Günümüzde popüler olarak kullanılan grup anahtar yönetim şemalarından bir tanesidir. MAH'ın ana fikri bir şifreleme anahtar kümesini içeren ve yetkili kullanıcıların eklendiği bir anahtar ağaç yapısı oluşturmaktır.

Bir mantıksal anahtar ağacı bir anahtar sunucu tarafından yönetilir. Bir ağaç içerisinde, anahtar düğümleri ve kullanıcı düğümleri olmak üzere iki farklı düğüm vardır. Kullanıcı düğümleri ağaçtaki en alt düğümler olan yapraklarda yer alır. Bir yaprak düğümde, yaprakta yer alan kullanıcı düğümü ile ilişkili bireysel anahtarlar bulunur.

Yayın merkezi ise kök düğümde yer alır. Yayın merkezi ile Trafik Şifreleme Anahtarı (TŞA) olarak isimlendirilen bir anahtar ilişkilendirilir. Yayın merkezinden kullanıcılara kadar olan yol üzerinde bulunan anahtar düğümler aradüğümler olarak isimlendirilir. Aradüğüm anahtarları Anahtar Şifreleme Anahtarı (AŞA) olarak isimlendirilir ve anahtar sunucu tarafından grup kullanıcılarına TŞA anahtarını güvenli bir şekilde teslim etmek için kullanılır.

Şema üzerinde bulunan her kullanıcıya, kök düğümde bulunan TŞA'yı hesaplayabilmesi için, kendisinden kök düğüme kadar olan yolda bulunan aradüğüm anahtarları güvenli bir yoldan iletilmelidir. İleri ve geri gizliliğin sağlanması için, şemaya bir kullanıcı eklendiğinde veya şemadan bir kullanıcı çıkartıldığında, kullanıcının bulunduğu noktadan kök düğüme kadar olan yol üzerinde bulunan anahtarlar güncellenmelidir. Şekil 12.6'da 8 kullanıcılı dolu bir ağaç yapısı bulunmaktadır. Örneğin $Kullanıcı_8$ ağaca katıldığında GY, AŞA8, AŞA78, AŞA5678 ve TŞA anahtarlarını güncellemeli ve $Kullanıcı_8$'e iletmelidir.

Bir kullanıcı sunucuya katılma isteği gönderdiğinde, öncelikle GY ile kullanıcı arasında doğrulama işlemi gerçekleştirilir. Eğer kullanıcının kimliği doğrulanırsa, bir kullanıcı düğümü ve onunla ilişkili bir anahtar düğümü oluşturulur. Oluşturulan anahtar değeri kullanıcıya güvenli bir yolla iletilir.



Şekil 12.6. 8 Kullanıcılı 3 Dereceli Bir Anahtar Ağacı

Bir sonraki adım kullanıcının ekleneceği düğümü bulmaktır. Eğer bir altkümedeki düğüm derecesi, ağacın derecesinden küçükse, bir diğer ifadeyle altkümedeki kullanıcı sayısı ağacın maksimum kullanıcı sayısından küçükse, yeni kullanıcı bu altküme içerisine eklenir. Eğer değilse, yeni bir altküme yaratılır ve yeni kullanıcı bu alt kümeye eklenir.

Ağaca eklenen veya çıkartılan her bir kullanıcı için o kullanıcının bireysel anahtarı oluşturulmalı veya silinmelidir. Kullanıcının bulunduğu yol üzerindeki düğüm anahtarları da değişmek zorundadır. Yeni kullanıcıya eklendiği katılım noktasından köke kadar ki tüm anahtar değerleri verilmelidir. Aynı şekilde grup anahtarı değişen düğümlere de yeni grup anahtarları dağıtılmalıdır. n kullanıcı için anahtar dağıtım karmaşıklığı $O(logn)$'dir.

Eğer ağaca yeni kullanıcılar eklenir ve ağaçta yeniden anahtarlama işlemi yapılmazsa, yeni kullanıcılar önceki mesajları çözebilme imkânına sahip olur. Bunu engellemek için ağaç üzerinde kullanıcının eklendiği ekleme noktasından köke kadar ki tüm düğümlere yeni anahtar değerleri atanmak ve o düğümlerin kullanıcılarına güncel anahtar değerleri dağıtılmak zorundadır. Bu sayede geriye erişim kontrolü garanti altına alınmış olur.

$Kullanıcı_8$'in ağaca eklendiğini varsayalım. $Kullanıcı_8$ ağaca dahil olduğunda aşağıdaki adımlar sırasıyla gerçekleştirilir.

$\{\text{AŞA}_{78}^{\text{rastgele}}\}_{\text{AŞA}_8} \rightarrow \text{AŞA}'_{78}$ anahtarı AŞA_8 anahtarıyla şifrelenerek $Kullanıcı_8$'e iletilir.

$\{\text{AŞA}'_{78}\}_{\text{AŞA}_7} \rightarrow \text{AŞA}'_{78}$ anahtarı AŞA_7 anahtarıyla şifrelenerek $Kullanıcı_7$'ye iletilir.

$\{\text{AŞA}'_{5678}\}_{\text{AŞA}_8} \rightarrow \text{AŞA}'_{5678}$ anahtarı AŞA_8 anahtarıyla şifrelenerek $Kullanıcı_8$'e iletilir.

$\{\text{AŞA}'_{5678}\}_{\text{AŞA}_{5678}} \rightarrow \text{AŞA}'_{5678}$ anahtarı AŞA_{5678} anahtarıyla şifrelenerek $Kullanıcı_5, Kullanıcı_6$ ve $Kullanıcı_7$'ye iletilir.

$\{TŞA'\}_{\text{AŞA}_8} \rightarrow TŞA'$ anahtarı AŞA_8 anahtarıyla şifrelenerek $Kullanıcı_8$'e iletilir.

$\{TŞA'\}_{TŞA} \rightarrow TŞA'$ anahtarı $TŞA$ anahtarıyla şifrelenerek $Kullanıcı_1$, $Kullanıcı_2, Kullanıcı_3, Kullanıcı_4, Kullanıcı_5, Kullanıcı_6$ ve $Kullanıcı_7$'ye iletilir.

Bir MAH şemasında bulunan kullanıcı sayısı n ise, ağacın yüksekliği

$h = log_2\, n$'dir. Ağacın derinliği onun derecesi olarak ifade edilir. Her kullanıcı, bir tanesi kendi bireysel anahtarı olmak üzere, bulunduğu düğümden kök düğüme kadar olan yolda bulunan toplam rastgele+1 adet anahtarı saklamaya ihtiyaç duyar. Diğer rastgele adet anahtar değeri o düğümden kök düğüme olan aradüğümlerin anahtarlarıdır. Bu nedenle gereken saklama alanı boyutu $O(h)$'dır.

Bir kullanıcıyı silmek, ekleme işlemine benzer yolla yapılır. Öncelikle kullanıcıyla ilişkili düğümler ağaçtan silinir. Silinen kullanıcının gelecek mesajları çözmesini engellemek için, bulunduğu noktadan kök düğüme kadar ki yol üzerindeki tüm aradüğümler için yeni anahtarlar hesaplanır. Ardından bu anahtarlar yol üzerinde bulunan kullanıcılara dağıtılır. Bu sayede ileri gizlilik sağlanmış olur.

Şekil 12.6 üzerinde $Kullanıcı_8$'in ağaçtan silindiğini varsayalım. Bu durumda $Kullanıcı_8$'in ağaç üzerinde yayınlanacak mesajları okumasını engellemek için kullanıcının bulunduğu konumdan kök düğüme kadar olan düğümler üzerinde yeniden anahtarlama işlemlerinin yapılması gerekir.

Öncelikle $Kullanıcı_8$'e ait AŞA_8 anahtarı silinir.

Ardından AŞA_{78}, AŞA_{5678} ve $TŞA$ anahtarları güncellenir.

Güncelleme işleminin ardından yeni $\text{AŞA}'_{78}$, $\text{AŞA}'_{5678}$ ve $TŞA'$ anahtarlarının diğer kullanıcılara güvenli bir şekilde iletilmesi gerekir. Bunun için aşağıdaki işlem adımları uygulanır.

$\{\text{AŞA}'_{78}\}_{\text{AŞA}_7} \rightarrow \text{AŞA}'_{78}$ anahtarı AŞA_7 anahtarıyla şifrelenerek $Kullanıcı_7$'ye iletilir.

$\{\text{AŞA}'_{5678}\}_{\text{AŞA}_{78}} \rightarrow \text{AŞA}'_{5678}$ anahtarı AŞA_{78} anahtarıyla şifrelenerek $Kullanıcı_7$'ye iletilir.

$\{\text{AŞA}'_{5678}\}_{\text{AŞA}_{56}} \rightarrow \text{AŞA}'_{5678}$ anahtarı AŞA_{56} anahtarıyla şifrelenerek $Kullanıcı_5$ ve $Kullanıcı_6$'ya iletilir.

$\{TŞA'\}_{AŞA'_{5678}} \rightarrow TŞA'$ anahtarı $AŞA'_{5678}$ anahtarıyla şifrelenerek $Kullanıcı_5$, $Kullanıcı_6$ ve $Kullanıcı_7$'ye iletilir.

$\{TŞA'\}_{AŞA_{1234}} \rightarrow TŞA'$ anahtarı $AŞA_{1234}$ anahtarıyla şifrelenerek $Kullanıcı_1$, $Kullanıcı_2, Kullanıcı_3$ ve $Kullanıcı_4$'e iletilir.

Bir mesaj tüm yetkili kullanıcılara yayınlandığında, basit bir şekilde bir kök anahtar ile şifrelenir. Çünkü yalnız yetkili kullanıcılar kök anahtarına sahiptir ve herhangi bir ek hesaplama yapmadan yayın mesajını çözebilirler. Bu nedenle bir yayın şifreleme ve çözme karmaşıklığı *O*(1)'dir.

Tek Yönlü Fonksiyon Ağacı

Tek-yönlü Fonksiyon Ağacı (TFA) (One-way Function Tree), geniş ve dinamik gruplar için etkili bir anahtar yönetim şemasıdır. Sherman ve McGrew tarafından önerilmiştir [18]. Ağaç üzerindeki kullanıcı anahtarları aşağıdan yukarıya doğru bir tek yönlü fonksiyon g(), bir birleştirme fonksiyonu f() ve bir anahtar oluşturma fonksiyonu anahtar() kullanılarak hesaplanır. Kullanıcılar yapraklarda bulunur. Her kullanıcı düğümü (x) ile ilişkilendirilmiş üç kriptografik anahtar bulunur:

n_x düğüm sırrıdır.

n_x^l ise kör düğüm sırrıdır. Bir tek yönlü fonksiyon *g()* kullanılarak hesaplanır. $n_x^l = g(n_x)$

Her düğüm anahtarı k, düğüm sırrının *anahtar()* fonksiyonuna girmesiyle elde edilir. $k_x = anahtar(n_x)$

f() fonksiyonu ise karıştırma (ör. XOR) fonksiyonudur.

MAH şemasında anahtar değerleri kökten yapraklarda bulunan kullanıcılara yukarıdan aşağıya doğru dağıtılırken, TFA şemasında yapraklarda bulunan kullanıcılardan köke doğru aşağıdan yukarıya bir yol izler.

Her düğüm sırrı n_x, sol ve sağ çocuklarının kör düğüm sırlarının birleştirilmesinden elde edilir. Aradüğüm hesaplaması aşağıdaki denklemdeki gibidir.

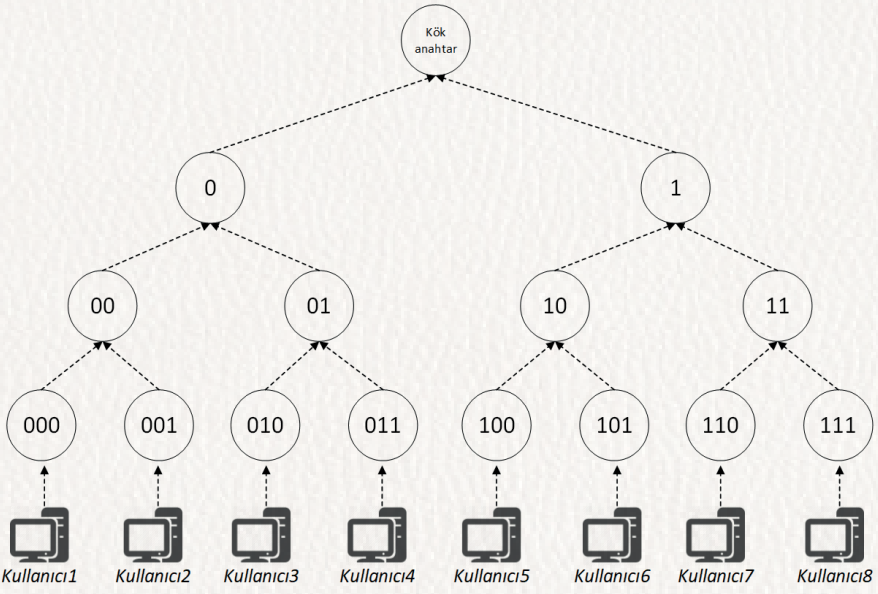
$$n_x = f(g\left(n_{x_{sol_{cocuk}}}\right), g\left(n_{x_{sağ_{cocuk}}}\right)) = f(n^l_{x_{sol_{cocuk}}}, n^l_{x_{sağ_{cocuk}}})$$

$$n_{00} = f(g(n_{000}), g(n_{001})) \qquad n_{01} = f(g(n_{010}), g(n_{011}))$$

$$n_{10} = f(g(n_{100}), g(n_{101})) \qquad n_{11} = f(g(n_{110}), g(n_{111}))$$

$$n_0 = f(g(n_{00}), g(n_{01})) \qquad n_1 = f(g(n_{10}), g(n_{11}))$$

$$n_{kok_{anahtar}} = f(g(n_0), g(n_1))$$



Şekil 12.7. Kullanıcılardan Kök Düğüme Gizli Anahtar Hesaplama

Yapraklarda bulunan her kullanıcının, düğüm sırlarını hesaplayabilmesi için kendisinden kök düğüme kadar ki yolda kardeş kör düğüm sırlarını bilmesi gerekir. Örneğin; Şekil 12.7'de $Kullanıcı_7$'ye n^l_{111}, n^l_{10} ve n^l_0 değerleri iletilmelidir. Aksi halde kullanıcı grup anahtarını hesaplayamaz. Sistem güvenliği her kullanıcının anahtar ağacının mevcut durumu hakkındaki bilgisine dayalıdır.

Ağaca bir kullanıcının eklenmesi için, öncelikle ağaç üzerinde konumunun belirlenmesi gerekir. Ardından o konumdaki yaprak düğüm x, sol(x) ve sağ(x)

olmak üzere ikiye bölünür. Mevcut düğüm sol kısımla yeni kullanıcı ise sağ kısımla ilişkilendirilir. Yeni anahtarlar oluşturulup her iki kullanıcıya da verilir. Güncellenen kör düğüm sırları ilgili kullanıcılara güvenli bir şekilde dağıtılır. Yeni eklenen kullanıcıya grup anahtarını hesaplayabilmesi için güvenli bir yoldan kendisinden kök düğüme kadar ki yolda bulunan kardeş düğüm kör anahtar sırları iletilir. Ağacın yüksekliğini mümkün olduğunca düşük tutmak için, bir kullanıcı eklenmek istendiğinde kök düğüme en yakın düğüm bölünerek ekleme işlemi yapılır.

Bir kullanıcı gruptan ayrıldığında kardeşi grubun ebeveyni konumuna getirilir ve kendisine yeni bir yaprak anahtar değeri verilir. Değişen kör düğüm anahtarlarının yeni değerleri ihtiyaç duyan kullanıcılara güvenli bir şekilde aktarılır.

Eğer ağacın yüksekliği rastgele ise, bir ekleme veya çıkarma işleminin ardından yaklaşık olarak rastgele adet yeni anahtar iletimi yapılır. MAH ağacının aksine anahtar değerleri fonksiyonel olarak birbirine bağlı olduğundan, TFA ağacında bir ekleme veya çıkarma işleminin ardından x bitlik anahtar iletimleri söz konusu iken bu değer MAH ağacında 2x boyutundadır.

Tek Yönlü Fonksiyon Zinciri

Tek Yönlü Fonksiyon Zinciri (TFZ) (One-way Function Chain), Canetti ve arkadaşları tarafından önerilmiştir [19]. TFA'nın aksine TFZ de kör düğüm sırları bulunmaz. Biri düğüm sırrı (r_x) diğeri de düğüm anahtarı (k_x) olmak üzere her düğüm ile iki anahtar ilişkilendirilir. Kullanıcılar yapraklarda bulunur. Ağaçtan ayrılan kullanıcının ebeveyni ile kök düğüm arasında bulunan düğüm sırları arasında her zaman fonksiyonel bir ilişki bulunur. Bu ilişki zincir olarak isimlendirilir.

TFZ içerisinde kullanılan f tek yönlü bir rastgele üreteçtir. İçerisine giren bir verinin iki katı boyutunda çıktı üretir. Bu fonksiyon düğüm anahtarları ve düğüm sırlarını üretmek için kullanılır.

Şekil 12.7'de görüldüğü üzere, ağaçta 8 kullanıcı olduğunu ve ağaçtan $Kullanıcı_1$ 'in çıkarıldığını varsayalım. Öncelikle $Kullanıcı_1$ ile ilişkili düğümün ağaç üzerinden silinmesi gerekir. Ardından silinen düğümün ebeveyninden kök düğüme kadarki yol üzerinde bulunan düğüm anahtar ve düğüm

sır değerlerinin güncellenmesi ve güncel anahtarların ihtiyaç duyan kullanıcılara dağıtılması gerekir.

GY yeni bir rastgele r_{00} değeri üretir. Oluşturulan r_{00}, k_{001} ile şifrelenerek $Kullanıcı_2$'ye iletilir. Ardından r_{00} bir $f()$ fonksiyonuna sokulur. Bu fonksiyonun sol yarısı ilgili düğümün düğüm anahtarı olan $k_{00}{}' = f(r_{00})|_L$ ile, sağ yarısı ise üst düğümün düğüm sırrı olan $r_0 = f(r_{00})|_R$ ile ilişkilidir.

r_0 değeri, k_{01} ile şifrelenerek $Kullanıcı_3$ ve $Kullanıcı_4$'e iletilir.

$k_0{}' = f(r_0)|_L$ işlemi ile bu düğümün düğüm anahtarı ve $r_{kök_{anahtar}} = f(r_0)|_R$ işlemi ile kök düğümün düğüm sırrı elde edilir.

$r_{kök_{anahtar}}$ değeri k_1 ile şifrelenerek $Kullanıcı_5 - Kullanıcı_8$'e iletilir.

Son olarak $k_{kök_{anahtar}}{}' = f(r_{kök_{anahtar}})|_L$ işlemiyle kök düğümün düğüm anahtarı elde edilir.

Ağaçta bulunan kullanıcılar ihtiyaç duydukları anahtarları kolaylıkla hesaplayabilirler.

Örneğin $Kullanıcı_2, r_{00}$ değerini elde etmek için k_{001} anahtarını kullanır. Ardından sırasıyla $k_{00}{}' = f(r_{00})|_L$, $k_0{}' = f(f(r_{00})|_R)|_L$ yi ve $k_{root}{}' = f(f(f(r_{00})|_R)|_R)|_L$ işlemlerini yaparak kök düğümün güncel anahtarını elde eder. İletişim maliyeti $logn$'dir. TFZ üzerinde düğüm sırları arasındaki zincir ilişki genellikle bir kullanıcı eklendiğinde değil, bir kullanıcı ayrıldığında ortaya çıkmaktadır.

Tek Yönlü Melez Anahtar Dağıtım Şeması

Tek Yönlü Melez Anahtar Dağıtım Şeması (TMAD) şeması Bodur ve Kara tarafından önerilmiştir [20]. İkili ağaç yapısı temellidir. Yayın merkezi kök düğümde, kullanıcılar yapraklarda bulunmaktadır. Anahtar hesaplaması TFA ve TFZ şemalarında olduğu gibi yapraklardan kök düğüme doğrudur. Anahtar dağıtımında hem açık anahtarlı hem de gizli anahtarlı şifreleme yöntemlerinden yararlanılır.

Kullanıcıların ortak gizli anahtar değerini elde etmelerini sağlamak için Eliptik Eğri Diffie-Hellman (EEDH) protokolünden yararlanılır. Ardından kök düğüme kadarki yol üzerinde bulunan aradüğümlerin ve kök düğüm ortak gizli anahtarının hesaplanmasında özetleme fonksiyonundan yararlanılır.

Ağaca eklenen her kullanıcı, EEDH protokolünü kullanarak açık anahtarını ($pu_{kullanıcı\_x}$) ve gizli anahtarını ($pr_{kullanıcı\_x}$) oluşturur.

Ardından simetrik anahtar değerini oluşturmak için $pr_{kullanıcı\_X}$ anahtarını bir rastgele üreteç kullanılarak oluşturulmuş, bir rastgele veri ile tuzlayarak bir $ozet()$ fonksiyonuna sokar. $ozet()$ fonksiyonu ilk olarak içerişine aldığı verinin özetini çıkartır. Ardından özet verisini belirli bir boyuta indirger.

$$n_{kullanıcı\_x} = ozet(pr_{kullanıcı\_x} + rasgele_{veri})$$

Elde edilen $n_{kullanıcı\_x}$ değeri kullanıcının simetrik değeridir.

Kullanıcı $pu_{kullanıcı\_x}$ ve $pr_{kullanıcı\_x}$ anahtarlarına ve $n_{kullanıcı\_x}$ değerine sahiptir. Bu anahtarlardan $pu_{kullanıcı\_x}$ 'i ve şema üzerindeki pozisyonuna göre $n_{kullanıcı\_x}$ değerinin sol ya da sağ yarısını önceden tanımlı Açık Anahtar Kütüphanesi (AAK)'ne yükler.

TMAD şemasında yayın merkezinde bulunan ortak gizli anahtarı hesaplamak için, TFA şemasına benzer bir şekilde, kullanıcılardan kök düğüme doğru bir anahtar hesaplama işlemi gerçekleştirilir. Her aradüğüm değeri n_x sol çocuğunun simetrik değerinin sol yarısı ile sağ çocuğunun simetrik değerinin sağ yarısının birleştirilip özetinin alınmasıyla elde edilir. İkili ağaç üzerinde yaprak düğümlerden kök düğüme doğru aradüğüm değerlerinin hesaplanması aşağıdaki gibidir.

$$n_x = ozet(n_{x_{solCocukAnahtarınınSolYarısı}} + n_{x_{sagCocukAnahtarınınSagYarısı}})$$

$$n_{00} = ozet(n_{000_{solYari}} + n_{001_{sagYari}})$$

$$n_{01} = ozet(n_{010_{solYari}} + n_{011_{sagYari}})$$

$$n_{10} = ozet(n_{100_{solYari}} + n_{101_{sagYari}})$$

$$n_{11} = ozet(n_{110_{solYari}} + n_{111_{sagYari}})$$

$$n_0 = ozet(n_{00_{solYari}} + n_{01_{sagYari}})$$

$$n_1 = ozet(n_{10_{solYari}} + n_{11_{sagYari}})$$

$$n_{kok\_deger} = ozet(n_{0_{left}} + n_{1_{sagYari}})$$

$$n_{kok\_anahtar} = anahtar(n_{kok\_deger})$$

Kök düğümde bulunan n_{kok_deger} değeri bir $anahtar()$ fonksiyonuna sokularak, ortak gizli anahtar bir diğer ifade ile simetrik anahtar elde edilir. $anahtar()$ fonksiyonu yalnızca yayın merkezinde ve kullanıcılarda bulunur. TMAD şemasında kullanıcılardan kök düğüme doğru simetrik değerlerin yalnızca yarısı iletilir. Bu durum anahtar güncelleme işlemlerinde iletilen toplam anahtar boyutunun azalmasını sağlar.

12.3.2.2. Dağıtık Güvenli Grup İletişim Şemaları

Dağıtık GGİ şemalarında merkezi bir varlık GY bulunmaz. Tüm grup kullanıcıları aralarında iş birliği yaparak, bir GY iş yükünü paylaşırlar. Dağıtık şemaların çoğu GÜY ile ilgilenmezler yalnızca GAY çözümleri sunarlar. Merkezi grup anahtar yönetimi ile karşılaştırıldığında, dağıtık grup anahtar yönetim şeması hata toleransı açısından avantaja sahip iken, hesaplama maliyeti açısından dezavantaja sahiptir.

Ağaç Tabanlı Grup Diffie Hellman

Ağaç tabanlı Grup Diffie-Hellman (AGDH) (Tree-based Group Diffie-Hellman), ikili ağaç temellidir. Kim ve arkadaşları tarafından önerilmiştir [21]. Temelinde açık anahtarlı şifreleme algoritmaları kullanılır. AGDH içerisinde kullanılan notasyonlar şunlardır:

$< l, v >$: ağacın l.seviyesindeki v.düğüm anlamına gelir.

$k_{<l,v>}$: $< l, v >$ düğümünün anahtarı (gizli anahtar) anlamına gelir.

$bk_{<l,v>}$: $< l, v >$ düğümünün kör anahtarı (açık anahtar) anlamına gelir.

$T_{<i,j>}$: $< i, j >$ düğümünün kök olduğu alt ağaç anlamına gelir.

AGDH yönteminde $< 0,0 >$ düğümü kök düğümdür. Kullanıcılar yapraklarda bulunur. Diğer düğümler aradüğümlerdir. Her aradüğümün iki çocuğu bulunur. $< l, v >$ düğümünün sol çocuğu $< l + 1,2v >$, sağ çocuğu ise $< l + 1,2v + 1 >$ konumunda bulunmaktadır.

Z_p içerisinde g primitif bir kök olmalı ve p değeri olarak büyük bir asal sayı seçilmelidir.

$< l, v >$ konumundaki düğümün kör anahtarı $bk_{<l,v>} = g^{k<l,v>}$ denklemiyle hesaplanır. Kök düğüm haricindeki her düğümün bir gizli birde açık anahtarı bulunur.

Aradüğüm anahtarları aşağıdan yukarıya anahtarların kullanılmasıyla hesaplanır. Bu nedenle AGDH şeması TFA ile benzerlik göstermesine rağmen aralarında beş önemli fark bulunur.

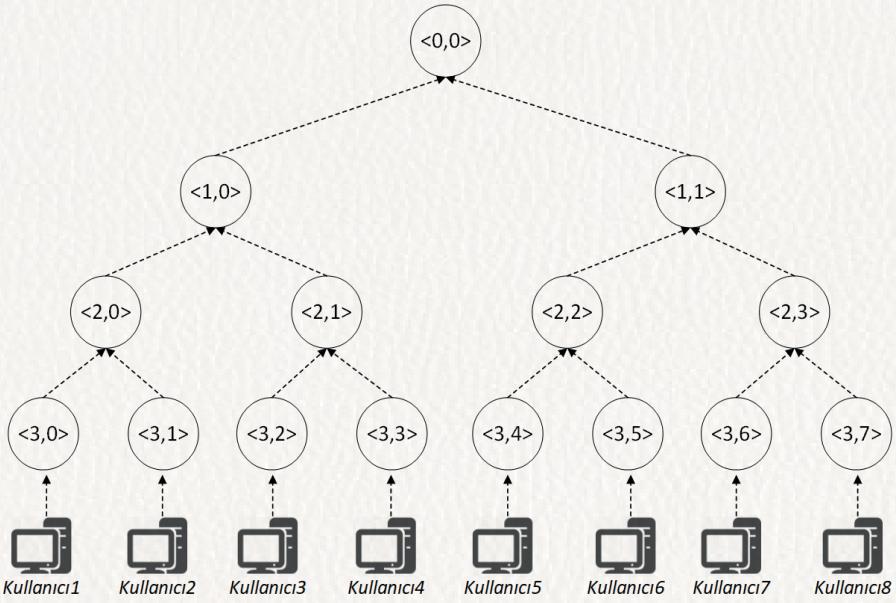
1) TFA'daki kör anahtar ortaya çıkarılmaması gereken bir anahtar değeri iken, AGDH'daki tüm kör anahtarların, açık anahtar altyapısına sahip olduklarından, gizli tutulmasına gerek yoktur.
2) TFA merkezi iken AGDH dağıtıktır. AGDH'da gizli anahtarı taşıyan hiçbir merkezi varlığa gerek yoktur.
3) TFA'dan farklı olarak AGDH'da ayrıca birleştirme ve bölme protokolleri tanımlanmıştır.
4) Şemaların güvenliği birbirinden farklıdır.
5) TFA bir k-ary yapısı olduğundan farklı k değerlerinde ağaç şeması oluşturulabilirken, AGDH'da bir aradüğümün en fazla 2 çocuğu bulunabilir.

Aradüğüm anahtarları, Diffie-Hellman anahtar değişimi kullanılarak hesaplanır. Aradüğüm gizli anahtarları, aradüğümlerin çocuklarından birinin açık anahtar değerinin, diğer çocuğun gizli anahtar değeriyle üs işlemine tabi tutulup, ağacın kurulum aşamasında belirlenen p değeriyle modu alınarak elde edilir.

$$k_{<l,v>} = (bk_{<l+1,2v>})^{k<l+1,2v+1>} \bmod p = (bk_{<l+1,2v+1>})^{k<l+1,2v>} \bmod p$$

$$= g^{(k_{<l+1,2v>} * k_{<l+1,2v+1>})} \bmod p$$

8 elemanlı bir AGDH ağacı Şekil 12.8'deki gibidir.



Şekil 12.8. 8 Elemanlı AGDH Anahtar Ağacı

$k_{<2,0>}$ düğümünün gizli anahtarının hesaplanması için aşağıdaki işlem gerçekleştirilir.

$$k_{<2,0>} = (bk_{<3,0>})^{k_{<3,1>}} \, mod \, p \; = (bk_{<3,1>})^{k_{<3,0>}} \, mod \, p$$

$$= g^{(k_{3,0>} * k_{<3,1>})} \, mod \, p$$

Her kullanıcı, kendisinden kök düğüme kadarki yol üzerinde bulunan düğümlerin gizli anahtarlarını hesaplayabilmesi için, kendisinden kök düğüme kadarki yol üzerinde bulunan kardeş düğüm açık anahtarlarını bilmelidir.

Bir yeni kullanıcı AGDH'a katılmak isterse, gruba içerisinde kendi açık anahtarının da yer aldığı bir istek mesajı göndermelidir. Gruptaki mevcut kullanıcılar, yeni kullanıcının ekleneceği konumu ve sponsor düğümü belirler. Sponsor düğüm eklenen ya da ayrılan kullanıcının konumuna göre mevcut kullanıcılardan seçilir. Sponsor düğüm kullanıcı ekleme-çıkarma işlemlerinin ardından, AGDH şeması üzerinde ileri ve geri gizliliğin sağlanması için, anahtar güncelleme ve güncel anahtarları yayınlama görevini geçici olarak üstlenir.

Bir sponsor düğüm aşağıdaki gibi tanımlanır.

1) Bir alt ağacın sponsoru, alt ağaçtaki en sağdaki yaprakta bulunan kullanıcıdır.
2) Bir yaprak düğümün sponsoru, kendisinin kullanıcısı olduğu en alt ağacın sağındaki (kendisi hariç) yaprak düğümdür.

Ekleme işleminde, ağaçta bulunan her kullanıcı, yeni eklenen kullanıcı için bir yeni yaprak düğüm ve bir yeni aradüğüm oluşturur. Aradüğümü yeni eklenen düğümün ebeveyni olacak şekilde ağacı günceller. Ayrıca her kullanıcı sponsor düğümden kök düğüme kadar tüm gizli anahtarları ve kör anahtarları silerler.

Sponsor düğüm, bulunduğu konumdan kök düğüme kadarki yol üzerinde bulunan gizli anahtarları ve kör anahtarları hesaplar. Ardından, kör anahtarları içeren yeni ağacı yayınlar. Ağaçta bulunan diğer kullanıcılar güncel kör anahtarları ile yeni grup anahtarını hesaplarlar.

Gruptan bir kullanıcı ayrıldığında, ilk olarak sponsor düğüm belirlenir. Ağaçta bulunan diğer kullanıcılar ayrılan kullanıcı düğümünü siler. Silinen kullanıcının kardeş düğümü ebeveyn düğümü konumuna getirilir. Sponsor düğüm kendisinden kök düğüme kadar tüm gizli anahtar ve kör anahtar değerlerini hesaplar ve yeni kör anahtar setini yayınlar. Ağaçta bulunan diğer kullanıcılar güncel kör anahtarları ile yeni grup anahtarını hesaplarlar.

Dağıtık Ölçeklenebilir Güvenli İletişim

Dağıtık Ölçeklenebilir Güvenli İletişim (DÖGİ) (Distributed Scalable Secure Communication), Dondeti ve arkadaşları tarafından önerilmiş dağıtık bir şemadır [22]. TFA şeması ile arasında notasyon ve terminoloji açılarından bazı farklılıklara sahiptir. TFA şemasındaki bir tek yönlü fonksiyon f(), DÖGİ şemasında E olarak ifade edilir.

TFA içindeki bir düğüm sırrı, DÖGİ içerisinde gizli anahtar olarak isimlendirilir. TFA içindeki bir kör düğüm sırrı, DÖGİ içinde kör anahtar (açık anahtar) olarak isimlendirilir. TFA içindeki düğüm anahtarları, DÖGİ içindeki düğüm sırlarına denk gelmektedir.

E fonksiyonu, basit karıştırma işlemi yapan bir tek yönlü fonksiyondur; $mix(x, y) = x \oplus y$. Her X düğümünde iki anahtar bulunur: bir gizli anahtar k_x and bir kör anahtar $E(k_x)$. Bir kullanıcı, düğümünün gizli anahtarını rast-

gele seçer ve gizli tutar. Bir aradüğümün gizli anahtarı çocuklarının kör anahtarları karıştırılarak hesaplanır. Her düğümün kör anahtarı, gizli anahtarının E fonksiyonuna sokulmasıyla elde edilir.

Her kullanıcı ikilik düzende bir kimliğe sahiptir. Kullanıcı düğümden kök düğüme kadar anahtar hesaplaması yapabilmesi için, kullanıcıların komşularının belirlenmesi gerekir. Bunun için ikilik düzendeki kimlik bilgisi kullanılarak AnahtarIsbirligiBul() algoritması ile her düğümün anahtar iş birliği grubu belirlenir. Kullanıcı bu gruptaki kullanıcılardan sırasıyla kör anahtar bilgilerini alarak kök düğüme kadar anahtar hesaplama işlemini gerçekleştirir.

Ağaca yeni bir kullanıcının (x düğümü) katılması durumunda öncelikle kullanıcının ekleneceği pozisyon belirlenir. Kendisinde, kök düğümünden bulunduğu konuma doğru ikilik düzende bir kimlik verilir. Kullanıcı düğümünden kök düğüme kadarki yol üzerinde bulunan anahtar hesaplamaları aşağıdaki gibidir:

1) AnahtarIsbirligiBul() algoritması kullanılarak x düğümün anahtar iş birliği grubu belirlenir.
2) x düğümü, gizli anahtar k_x için bir rastgele değer seçer.
3) k_x'den kör anahtar değeri $E(k_x)$ hesaplanır.
4) x düğümü, sırasıyla anahtar iş birliği grubundaki komşusundan ilgili düğümün kör anahtar bilgisini alır.
5) Bir üst düğümün gizli anahtarı hesaplanır, $k_{\text{ebeveyn}} = mix\big(E(k_x), E(k_{\dot{x}})\big) = E(k_x) \oplus E(k_{\dot{x}})$.
6) Bir üst düğümün kör anahtarı hesaplanır, $E\big(k_{\text{ebeveyn}}\big)$.

(4)-(6) arası adımlar anahtar işbirliği grubundaki komşularının sayısı kadar devam eder. Bu işlemin sonucunda yeni kullanıcı güncel kök düğüm kör anahtarı ve gizli anahtar değerlerini hesaplar.

Ağaçtaki mevcut kullanıcılar yukarıdaki adımlara benzer şekilde komşularının güncel kör anahtar değerlerini elde ederek güncel kök düğüm kör anahtar ve gizli anahtar değerlerini hesaplarlar.

Kullanıcı çıkarma işleminde ise, bir kullanıcının ayrılmasının ardından kardeşi yeni bir kimliğe sahip olarak ebeveyn pozisyonuna gelir. Yeni bir gizli anahtar üretir. Ardından AnahtarIsBirligiBul() algoritması ile önceden tespit

edilmiş anahtar işbirliği grup kullanıcılarına sırasıyla kör anahtar bilgilerini hesaplayıp göndermesi gerekir. Güncel kör anahtar bilgilerini alan kullanıcılar kendi alt gruplarındaki kullanıcılara güncel anahtar bilgisini göndermekten sorumludur.

Sıska Ağaç

Sıska Ağaç (SISA) (Skinny Tree) şeması ilk olarak Steer ve arkadaşlarının çalışmalarıyla ortaya atılmıştır [23]. Kim ve arkadaşları bu yöntemi dinamik grup işlemlerini kapsayacak şekilde genişletmişlerdir [24]. Bir SISA şemasında aşağıdaki notasyonlar kullanılır.

n, N : grup kullanıcılarının sayısı

$Kullanıcı_i$: i'inci grup kullanıcısı

p: büyük bir asal sayı

α: üs alma tabanı

r_i: $Kullanıcı_i$'nin gizli anahtarı

br_i: $Kullanıcı_i$'nin kör (açık) anahtarı ($\alpha^{r_i} mod\ p$)

k_j: $Kullanıcı_1 \ldots . Kullanıcı_j$ arasında paylaşılan gizli anahtar

bk_j: k_j'nin kör anahtarı ($\alpha^{k_i} mod\ p$)

$N_{<j>}$: j ağaç düğümü

$IN_{<l>}$: l seviyesindeki iç ağaç düğümü

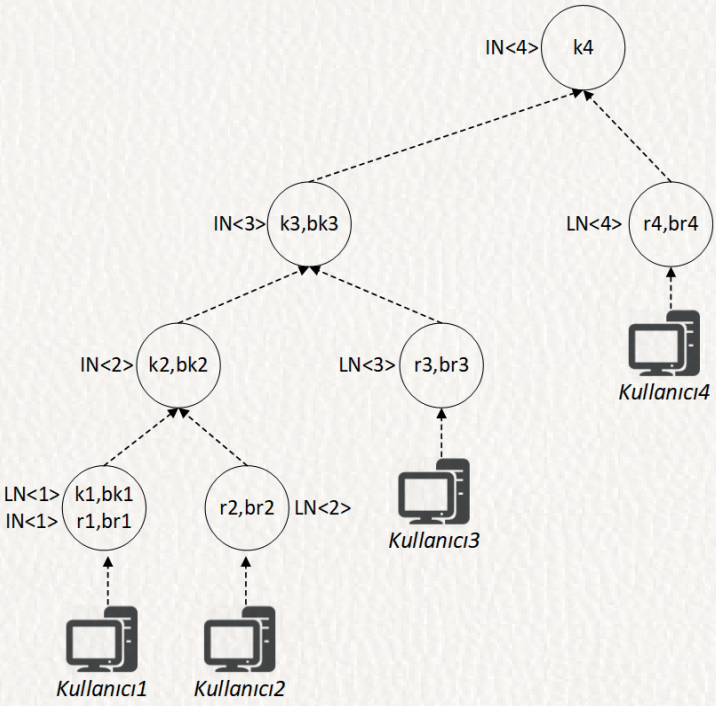
$LN_{<i>}$: $Kullanıcı_i$ ile ilişkili yaprak düğüm

$T_{<i>}$: $Kullanıcı_i$'nin ağacı

$BT_{<i>}$: $Kullanıcı_i$'nin tüm kör anahtarlarını içeren ağaç

Bir SISA ağacında, yaprak düğüm ve aradüğüm olmak üzere iki çeşit düğüm bulunur. En yüksek indisli aradüğüm kök düğüm olarak isimlendirilir. Kök düğümde bulunan gizli anahtar grup anahtarı olarak paylaşılır.

Şekil 12.9'da görüldüğü üzere bir aradüğüm $IN_{<i>}$'ın her zaman iki çocuğu bulunur: Bu çocuklardan biri aradüğüm $IN_{<i-1>}$ diğeri ise yaprak düğüm $LN_{<i+1>}$'dir. Bir istisna $IN_{<1>}$ aradüğüm olmakla birlikte yaprak düğüm $Kullanıcı_1$ ile de ilişkilidir.



Şekil 12.9. STR Ağacı

Her yaprak düğüm $LN_{<i>}$ bir gizli anahtar r_i 'e sahiptir. $LN_{<i>}$'in kör anahtarı r_i kullanılarak elde edilir: $br_i = \alpha^{r_i} mod\ p$

Her aradüğüm $IN_{<j>}$, bir gizli anahtar k_j ile ilişkilidir. $IN_{<j>}$'ın kör anahtarı, k_j kullanılarak elde edilir: $bk_j = \alpha^{k_j} mod\ p$

Kullanıcılardan kök düğüme k_i anahtarının hesaplanması için aşağıdaki işlem gerçekleştirilir (eğer i>1).

$$k_i = (bk_{i-1})^{r_i} mod\ p = (br_i)^{k_{i-1}} mod\ p = \alpha^{r_i k_{i-1}} mod\ p$$

Her kullanıcının grup anahtarını hesaplayabilmesi için kendi gizli anahtarını ve kardeş alt ağacının kör anahtarlarını bilmesi gerekir. Bunun için öncelikle iki kullanıcı $Kullanıcı_1$ ve $Kullanıcı_2$ grup anahtarını hesaplar.

$Kullanıcı_1$ 'in grup anahtarını hesaplayabilmesi için sırasıyla aşağıdaki işlemleri gerçekleştirmesi gerekir.

$k_2 = (br_2)^{r_1} \bmod p = \alpha^{r_1 r_2} \bmod p, \qquad bk_2 = \alpha^{k_2} \bmod p$

$k_3 = (br_3)^{k_2} \bmod p = \alpha^{r_1 r_2 r_3} \bmod p, \qquad bk_3 = \alpha^{k_3} \bmod p$

$k_4 = (br_4)^{k_3} \bmod p = \alpha^{r_1 r_2 r_3 r_4} \bmod p$

$k_N = (br_N)^{k_{N-1}} \bmod p$

Ardından $Kullanıcı_1$ tüm kör anahtarları $(1 \le i \le N-1;\ bk_i)$ yayınlar. Bu sayede her kullanıcı k_N'i hesaplayabilir.

Her kullanıcı $Kullanıcı_i$ (i>2) yayın mesajından kendi gizli anahtarı r_i 'yi ve bk_{i-1}'i bilir. $k_i = {bk_{i-1}}^{r_i} \bmod p$ işlemlerini gerçekleştirerek k_N'i hesaplayabilir.

Ağaca kullanıcı ekleme ya da ağaçtan kullanıcı çıkarma işlemlerinde anahtar güncelleme adımları belirlenen bir sponsor kullanıcı tarafından gerçekleştirilir. Ekleme işleminde sponsor, mevcut gruba eklenmiş son kullanıcıdır $(Kullanıc_N)$. Bir yeni kullanıcı $(Kullanıcı_{N+1})$ kendi kör anahtarını içeren bir katılma istek mesajı yayınlar. Bunun üzerine sponsor mevcut grup anahtarının kör versiyonunu (bk_N) hesaplar ve yeni kullanıcıya tüm kör anahtarları ve kör oturum anahtarlarını gönderir. Ağaçta bulunan diğer kullanıcılara yeni kullanıcının kör anahtarını gönderir. Yeni kullanıcıya mevcut grup anahtarının kör versiyonu verildiğinden ekleme işleminde geri gizlilik sağlamaktadır.

n kullanıcılı bir gruptan $Kullanıcı_d$ $(d \le n)$ ayrılsın. Eğer $d > 1$ ise sponsor düğüm, ayrılan kullanıcı düğümünün altında yer alan yaprak düğümdür, yani $Kullanıcı_{d-1}$. Ayrılma işleminin ardından, ağaçtaki kullanıcılar, $Kullanıcı_d$ ve ebeveyn düğümü $IN_{<d>}$ ile ilişkili olan $LN_{<d>}$ düğümünü silerek ağacı

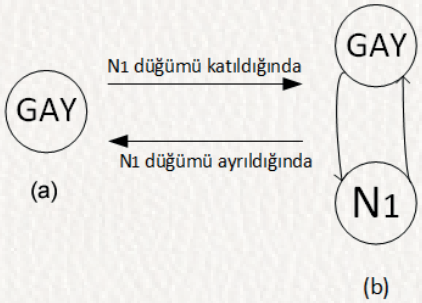
günceller. $Kullanıcı_d$'in kardeşi $IN_{<d-1>}$, çıkan düğüm $Kullanıcı_d$'nin ebeveyni konumuna yerleşir. Sponsor düğüm bir yeni gizli anahtar belirler, kök düğüme kadar ki yol üzerinde bulunan anahtarları ve kör anahtarları hesaplar, kör anahtarları gruba yayınlar. Bu sayede grupta bulunan diğer kullanıcılar yeni grup anahtarını hesaplayabilirler. Gruptan çıkartılan kullanıcı sponsor düğümün güncellediği gizli anahtara sahip olmadığından güncel grup anahtarını hesaplayamaz. Bu sayede ileri gizlilik sağlanmış olur.

12.4. BİRLEŞİK GÜVENLİ GRUP İLETİŞİM ŞEMALARI

Birleşik grup anahtar yönetiminde grup anahtar üretimi, bir merkezi grup anahtar yönetimi gibi bir GY tarafından gerçekleştirilir. Ancak grup anahtar dağıtımı bir dağıtık grup anahtar yönetiminde olduğu gibi grupta bulunan kullanıcılar tarafından gerçekleştirilir.

Mantıksal Halka Tabanlı Güvenli Grup İletişim Şeması

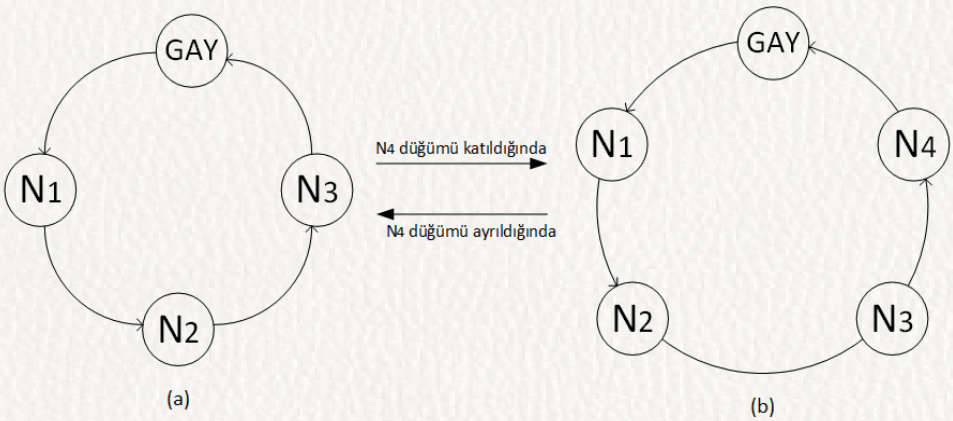
Omar Cheikhrouhou ve arkadaşları tarafından önerilen Mantıksal Halka Tabanlı Güvenli Grup İletişim Şeması (MHTGG) (Logical Ring Based Secure Group Communication Scheme) diğer şemaların aksine ikili ağaçtan farklıdır [25] [26]. Halka biçiminde sıralanmış kullanıcı düğümlerinden oluşur. Bu halka bir GY tarafından yönetilir. Mantıksal halka, bir mesajın GY'den tüm kullanıcı düğümlerine iletimi için görev dağıtımına izin verir. Mantıksal halka topolojisinde, bir mesaj gönderilen kaynağa tekrar ulaşana kadar düğümden düğüme dolaşır. Bu sayede GY, n kullanıcı için $O(n)$ mesaj göndermek yerine yalnızca $O(1)$ mesaj gönderir.



Şekil 12.10. Mantıksal Halka İçerisinde Bir Düğümü Ekleme-Silme İşlemleri - 1

Bir mantıksal halkada başlangıçta GY'yi içeren tek bir halka bulunur (Şekil 12.10a). Gruba yeni bir kullanıcı katılmak istediğinde, halkanın kuyruğuna yeni bir düğüm eklenir (Şekil 12.10b).

GY kuyruğunda yer alan düğüme bir mesaj iletir. Her düğüm kendisine gelen mesajı kuyruğuna bağlı olan bir diğer düğüme iletir. Mesaj başlangıç düğüm olan GY'e ulaştığında mesaj iletimi tamamlanmış olur. Gruba yeni bir kullanıcı eklendiğinde ya da bir kullanıcı ayrıldığında, GY yeni bir anahtar üretir ve grupta bulunan tüm kullanıcılara dağıtır. Bu sayede yeni kullanıcı önceki mesajları çözemez. Gruptan çıkan kullanıcı ise gelecek mesajlara erişemez. Bu sayede ileri ve geri gizlilik sağlanmış olur.



Şekil 12.11. Mantıksal Halka İçerisinde Bir Düğümü Ekleme-Silme İşlemleri - 2

GY, tüm grup kullanıcı düğümlerinin adreslerini yönetir. Her düğüm yalnızca bir önceki ve bir sonraki düğümün adresini bilir. Örneğin, Şekil 12.11(a)'da N_2'de bir önceki düğüm olan N_1'in adresi ve bir sonraki düğüm olan N_3'ün adresi bulunur.

Halkaya yeni bir düğüm eklendiğinde, GY yeni eklenen düğüme ağaca en son eklenmiş olan düğümün adresini verir. GY, ağaçtaki son kullanıcıya yeni düğümün bilgisini gönderir.

Şekil 12.11(b)'de görüldüğü üzere, N_4'ün halkaya eklenmesinin ardından, GY, N_4'e bir önceki düğüm olan N_3'ün adresini gönderir. N_4'den sonraki dü-

düm GY'dir. GY ayrıca N_3'e kendi bir sonraki düğüm adresini güncellemesi için bir sonraki düğüm olan N_4'ün adresini gönderir.

Şemadan bir kullanıcı ayrıldığında, ayrılan kullanıcı düğümü mantıksal halkadan silinmelidir. Bunun için GY, ayrılan düğümün üst düğümüne, ayrılan düğümün alt düğümünün adresini gönderir. Örneğin; Şekil 12.11(a)'da halkadan N_2 düğümü ayrılırsa, GY N_3'ü önceki düğüm adresinin N_1 olduğu ve N_1'i sonraki düğüm adresinin N_3 olduğu konusunda bilgilendirir.

Çoklu İletişim için Güvenli Seviye Anahtar Altyapısı

Çoklu İletişim için Güvenli Seviye Anahtar Altyapısı (ÇİGSAA) (Secure Level Key Infrastructure for Multicast) şeması çoklu iletişimde veri bütünlüğünü korumak için Huang ve arkadaşları tarafından geliştirilmiş bir güvenli anahtar altyapısıdır [27].

ÇİGSAA şemasında, bir grup yönlendirme ağacı branşlara ve seviyelere bölünür. Bu sayede şemada bulunan kullanıcılar alt gruplara bölünürler. Ağacın her branşının her seviyesindeki düğümler arasındaki iletişim bir seviye anahtarı tarafından korunur. Seviye anahtarı şifreleme ve şifre çözme işlemlerinde kullanılır. Ağaca yeni bir düğüm ekleme veya bir düğüm çıkarma işleminin ardından yalnızca ilgili seviye anahtarı güncellenir. Bu durum geleneksel anahtar yönetim metotlarıyla karşılaştırıldığında enerji tasarrufu sağlar.

Ağaç üzerinde iki çeşit düğüm bulunur. Bu düğümler kullanıcı düğüm ve kullanıcı olmayan yönlendirme düğümleridir. Kullanıcı düğümlerinde seviye anahtarı bulunurken, yönlendirme düğümlerinde bulunmaz. Yönlendirme düğümleri ağaç üzerinde gönderilen verinin içeriği ile ilgilenmezler. Veriyi yalnızca sevk ederler. Kendilerinde seviye anahtarı bulunmadığından iletilen veriyi çözemezler.

Bir seviye belirli bir branş boyunca bulunan düğüm sayısı bakımından tanımlanır. Her bir seviyede bir ana düğüm ve kendisine bağlı kullanıcı düğümler ile yönlendirme düğümleri bulunur.

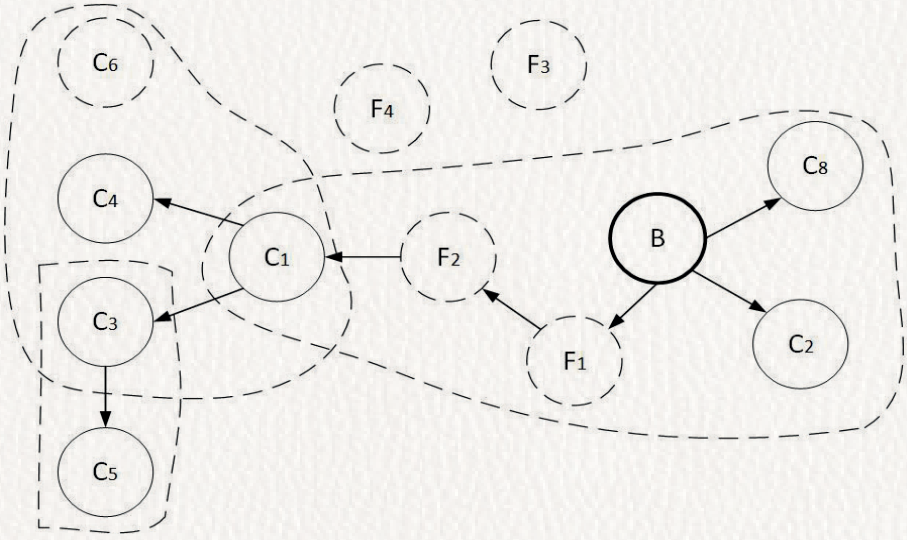
Bir ana düğüm, veriyi kendi seviye anahtarı ile şifreler ve kendisine bağlı kullanıcı düğümlere gönderir. Ana düğümün çocuğu olan her kullanıcı düğüm mesajı seviye anahtarı ile çözebilir. Eğer bir kullanıcı düğümün, ebeveyni olan ana düğümden gelen şifreli veriyi kendisinden sonraki seviyede çocuğu

pozisyonunda bulunan bir düğüme göndermesi gerekiyorsa, öncelikle şifreli veriyi ebeveyninin seviye anahtarı ile çözer. Ardından kendi seviye anahtarı ile şifreleyerek gönderir.

Her seviyenin anahtarı farklı olduğundan, her seviyede verinin şifresi çözülmesi ve gönderilmeden önce bir sonraki seviye anahtarı ile yeniden şifrelenmesi gerekir.

Şekil 12.12'de üç seviye bulunmaktadır.

Seviye 1'de B düğümü ebeveyndir. C_1, C_2 ve C_8 kullanıcı düğümleri ile F_1 ve F_2 yönlendirme düğümleri ana düğüm olan B düğümüne bağlıdırlar.



Şekil 12.12. Örnek Bir ÇİGSAA Şeması

Seviye 2'de ebeveyn düğüm C_1, kullanıcı düğümler ise C_3 ve C_4'dür.

Seviye 3'de ise ebeveyn düğüm C_3, kullanıcı düğüm ise C_5'dir. Her seviyede o seviyeye ait ve yalnızca o seviyenin ebeveyn düğümü ile kullanıcı düğümlerinin bildiği seviye anahtarları bulunur. Seviye anahtarları yalnızca o seviyeye yeni bir düğüm eklendiğinde veya ayrıldığında güncellenir. Bu durum anahtar güncellemesinin neden olduğu işlem maliyetini ve güç tüketimini azaltır.

ÇİGSAA şeması düşük iletişim maliyeti ve güç tüketimine sahiptir. Fakat gruptaki kullanıcı sayısı arttıkça güç tüketimi artmakta, performansı azalmaktadır.

12.5. SONUÇ VE DEĞERLENDİRMELER

Bu bölümde bir kaynaktan çoklu kullanıcılara doğru yapılan yayın iletimlerinde kullanılan yayın şifreleme şemalarına değinilmiştir. Yayın şifreleme şemaları ön tanımlı ve dinamik olarak ikiye ayrılmış, ön tanımlı şemalara kıyasla dinamik şemalara daha fazla ağırlık verilmiştir.

Yayın şifreleme şemalarının amacı bir mesajı bir kaynaktan çoklu kullanıcılara doğru güvenlik sorunlarına sebebiyet vermeden, düşük hesaplama, saklama ve iletişim maliyetiyle iletmektir. Bu şemalar günümüzde video konferans, çevrimiçi oyunlar, e-sağlık, askeri iletişim, internet protokol TV (IPTV), nesnelerin interneti vb. birçok alanda kullanılmakta ve güvenli çoklu yayın iletimine olanak sağlamaktadır. Çoklu veri iletiminin gerektiği bir çalışma alanında bir yayın şifreleme şeması kullanılmak istenirse hem topolojik açıdan hem de performans ve güvenlik açılarından o alana en uygun şema seçilmelidir. Bir yayın şifreleme şeması üzerinden aktarılan verilerin güvenliği, şema içerisinde kullanılan şifreleme algoritmasına bağlıdır. Şifreleme algoritmasının hesaplama gücü şema performansını doğrudan etkilemektedir. Örneğin bir askeri iletişim için şema güvenliği yüksek seviyede olacağından ortaya çıkacak maliyetler göz ardı edilebilir. Ancak sınırlı kapasiteye sahip cihazların yer aldığı nesnelerin interneti gibi bir alanda güvenlik mümkün olan minimum seviyede tutularak maliyetlerin azaltılması gerekir.

Karasal yayıncılığa ek olarak sayısal yayıncılık faaliyetleri ülkemizde de yaygın olarak kullanılmaktadır. Genişbant İnternetin yaygınlaşması ve kullanıcıların erişiminin kolaylaşmasından sonra sayısal yayıncılık için yeni bir alan olarak İnternet üzerinden radyo veya televizyon yayıncılığında da bir ivmelenme oluşmuştur.

Yayınların belirli kullanıcı kitlelerine şifreli olarak internet üzerinden ulaştırılması, yayın şifreleme şemalarının kullanımını daha önemli hale getirmiştir. Radyo ve Televizyon Üst Kurulu ile Bilgi Teknolojileri ve İletişim Kurumu tarafından 1 Ağustos 2019 tarihinde yayınlanan Radyo, Televizyon ve İste-

ğe Bağlı Yayınların İnternet Ortamından Sunumu Hakkında Yönetmelikte 4. Maddenin 1. Fıkrasının l bendinde "internet üzerinden şifreli yayın" kavramının da içerildiği tanım yer almaktadır. Aynı yönetmeliğin 16. Maddesinin 1. Fıkrasının e fıkrasında "şifreli yayınlarda görüntü ile birlikte sesin de anlaşılamayacak şekilde şifrelenmesi" konusuna yer verilmiştir.

Bu bölümde ele alınan yayın şifreleme şemaları ile veri, ses veya görüntüler şifreli olarak kullanıcılara ulaştırılabildiğinden yönetmelikte yer verilen konulara da uygun faaliyetler yapılması için imkan oluşturulabilir.

KAYNAKLAR

[1]. S. Iqbal, M.L.M. Kiah, B. Dhaghighi, M. Hussain, S. Khan, M.K. Khan and K.K.R. Choo, "On cloud security attacks: A taxonomy and intrusion detection and prevention as a service," Journal of Network and Computer Applications, vol. 74, pp. 98-120, 2016.

[2]. S. Iqbal, M.L.M. Kiah, A.A. Zaidan, B.B. Zaidan, O.S. Albahri, A.S. Albahri and M.A. Alsalem, "Real-time-based E-health systems: Design and implementation of a lightweight key management protocol for securing sensitive information of patients," Health and Technology, vol. 9, no. 2, pp. 93-111, 2019.

[3]. Yang, X. Zheng, X. Liu, S. Zhong and V. Chang, "Cross-domain dynamic anonymous authenticated group key management with symptom-matching for e-health social system," Future Generation Computer Systems, vol. 84, pp.160-176, 2018.

[4]. P. Vijayakumar, R. Naresh, S. Islam and L.J. Deborah, "An effective key distribution for secure internet pay-TV using access key hierarchies," Security and Communication Networks, vol. 9, no.18, pp. 5085-5097, 2016.

[5]. C. Esposito, M. Ficco, A. Castiglione, F. Palmieri and A. De Santis, "Distributed group key management for event notification confidentiality among sensors," IEEE Transactions on Dependable and Secure Computing, vol. 17, no. 3, pp. 566-580, 2018.

[6]. M.A. Kandi, Lakhlef, A. Bouabdallah and Challal, "A versatile Key Management protocol for secure Group and Device-to-Device Communication in the Internet of Things," Journal of Network and Computer Applications, vol. 150:102480, 2020.

[7]. S. Berkovits, "How to broadcast a secret," Workshop on the Theory and Application of Cryptographic Techniques, Springer, Heidelberg, Berlin, pp. 535-541.

[8]. A. Fiat and M. Naor, "Broadcast encryption," in Annual International Cryptology Conference, Springer, Heidelberg, Berlin, 1993, pp. 480-491.

[9]. D. Naor, M. Naor and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," in: Advances in Cryptology-CRYPTO '01, Lecture Notes in Computer Science, Heidelberg, Berlin, 2001, pp. 41-62.

[10]. P. Sakarindr and N. Ansari, "Security services in group communications over wireless infrastructure, mobile ad hoc, and wireless sensor networks," IEEE Wireless Communications, vol. 14, no. 5, pp. 8–20, 2007.

[11]. O. Cheikhrouhou, "Secure group communication in wireless sensor networks: a survey," Journal of Network and Computer Applications, vol. 61, pp. 115–132, 2016.

[12]. B. Daghighi, M. L. M. Kiah, S. Shamshirband and M.U. Rehman, "Toward secure group communication in wireless mobile environments: Issues, solutions, and challenges," Journal of Network and Computer Applications, vol. 50, pp. 1–14, 2015.

[13]. B. Daghighi, M. L. M. Kiah, S. Shamshirband and P. Asghari, "Key management paradigm for mobile secure group communications: Issues, solutions and challenge," Computer Communications, vol. 72, pp. 1–16, 2015.

[14]. X. He, M. Niedermeier and De Meer, "Dynamic key management in wireless sensor networks: A survey," Journal of Network and Computer Applications, vol. 36, no. 2, pp. 611–622, 2013.

[15]. Xiao, V. K. Rayi, B. Sun, X. Du, F. Hu and M. Galloway, "A survey of key management schemes in wireless sensor networks," Computer communications, vol. 30, no. 11-12, pp. 2314–2341, 2007.

[16]. C. K. Wong, M. Gouda and S. S. Lam, "Secure group communications using key graphs," IEEE/ACM transactions on networking, vol. 8, no. 1, pp. 16–30, 2000.

[17]. D. Wallner, E. Harder and R. Agee, "Key management for multicast: Issues and architectures," RFC 2627, 1999.

[18]. A. T. Sherman and D. A. McGrew, "Key establishment in large dynamic groups using one-way function trees," IEEE transactions on Software Engineering, vol. 29, no. 5, pp. 444–458, 2003.

[19]. R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor and B. Pinkas, "Multicast security: A taxonomy and some efficient constructions," In IEEE INFOCOM'99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No. 99CH36320, 1999, pp. 708–716.

[20]. Bodur and R. Kara, "A Comparison on Broadcast Encryption Schemes: A New Broadcast Encryption Scheme," Advances in Electrical and Computer Engineering, vol. 20, no. 4, pp. 69–80, 2020.

[21]. Kim, A. Perrig and G. Tsudik, "Tree-based group key agreement," ACM Transactions on Information and System Security (TISSEC), vol. 7, no. 1, pp. 60–96, 2004.

[22]. L. R. Dondeti, S. Mukherjee and A. Samal, "DISEC: a distributed framework for scalable secure many-to-many communication," In Proceedings ISCC 2000. Fifth IEEE Symposium on Computers and Communications. IEEE, 2000, pp. 693–698.

[23]. D. G. Steer, L. Strawczynski, Diffie and M. Wiener, "A secure audio teleconference system," In Proceedings on Advances in cryptology, Springer, New York, USA, 1990, pp.520–528.

[24]. Kim, A. Perrig and G. Tsudik, "Communication-efficient group key agreement," IFIP International Information Security Conference, Springer, Boston, MA, 2001.

[25]. O. Cheikhrouhou, A. Koubaa, O. Gaddour, G. Dini and M. Abid, "RiSeG: A logical ring based secure group communication protocol for Wireless Sensor Networks," In Communication in Wireless Environments and Ubiquitous Systems: New Challenges (ICWUS), 2010 International Conference on. IEEE, 2010, pp. 1–5.

[26]. O. Cheikhrouhou, A. Koubaa, G. Dini and M. Abid, "RiSeG: a ring based secure group communication protocol for resource-constrained wireless sensor networks," Personal and Ubiquitous Computing, vol. 15, no. 8, pp. 783–797, 2011.

[27]. J. Huang, J. Buckingham and R. Han, "A level key infrastructure for secure and efficient group communication in wireless sensor network," In Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'05), IEEE, 2005, pp. 249–260.

DİZİN

YAZARLAR

EDİTÖRLER

Prof. Dr. Şeref Sağıroğlu

Gazi Üniversitesi Yapay Zeka ve Büyük Veri Analitiği ve Güvenliği Merkez Müdürü
Bilgi Güvenliği Derneği Ulusal Bilim Kurulu Başkanı
Gazi Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü Öğretim Üyesi

Prof. Sağıroğlu, ülkemizde bilgi güvenliği, siber güvenlik ve büyük veri analitiği, güvenliği ve mahremiyeti konularında çalışmalar yapmaktadır. 20'nin üzerinde yayınlanmış kitabı bulunmaktadır. Biri amerikan patenti olmak üzere alındığı ve müracaat aşamasında olan 10'un üzerinde patenti, 100'ün üzerinde ulusal ve uluslararası indeksli dergilerde yayınlanmış makalesi ile 300'e yakın ulusal ve uluslararası yayımlanmış bildirisi ve 9000'in üzerinde atıfı bulunmaktadır. Bilgi güvenliği alanında iki akademik derginin de baş editörlüğünü yapmaktadır.

Uluslararası Bilgi Güvenliği ve Kriptoloji Konferansı (www.iscturkey.org), IEEE Uluslararası Bilgisayar Bilimleri ve Mühendisliği Konferansı (www. ubmk.org), IEEE Uluslararası Makine Öğrenmesi ve Uygulamaları Konferansı (www.icmla-conferences.org), Büyük Veri Analitiği, Güvenliği ve Mahremiyeti Ulusal Kamu Çalıştayı (bigdatacenter.gazi.edu.tr), Ulusal Siber Terör Konferansı (www.siberteror.org), Açık Veri Türkiye Konferansı (www.acikveriturkiye.org), Siber Güvenlik ve Savunma Çalıştayı (www.iscturkey.org), UluslararasıAdli Bilişim ve Güvenlik Sempozyumu (www.isdfs.org) gibi konferansların başkanlığını veya eşbaşkanlığını yürütmüş veya yürütmektedir.

Bilgi Güvenliği Derneği (BGD), Türk Bilim Araştırma Vakfı (TÜBAV), Geleceği Önemseyenler Derneği (GÖNDER) Kurucu Üyesidir. İki dönem, BGD Yönetim Kurulu Başkanlığı ve TÜBAV Genel Başkanlığı, Gazi Üniversitesi Fen Bilimleri Enstitüsü Müdürlüğü, Kısa bir süre Gazi Üniversitesi Mühendislik Fakültesi Dekanlığı görevini gibi görevleri yürütmüştür.

Gönüllü olarak pek çok sosyal projeyi de yürütmüş olan Sağıroğlu, TÜBİTAK, Cumhurbaşkanlığı DDO, Avrupa Birliği, BAP gibi Bilimsel Araştırma Projeleri tamamlamış veya yürütmeye devam etmektedir. IEEE gibi ulusal ve uluslararası konferanslarda, Bilgi Güvenliği, Büyük Veri, Siber Güvenlik ve Savunma, Akıllı Şebekeler, Yapay Zeka, Biyometrik Uygulamalar, İnovasyon Kültürü Oluşturma gibi konularda davetli konuşmacı olarak seminer ve konferanslar vermiş ve vermeye devam etmektedir.

Son dönemde ise ülkemizde yeni alanlarda Türkçe içerik oluşturma konusunda projeler yürütmekte ve özellikle de Yapay Zeka ve Büyük Veri Kitap Serisi, Siber Güvenlik ve Savunma Kitap Serisi, Dijital Okur Yazarlık Kitap Serisi gibi projelere Editör olarak katkı vermektedir.

Prof. Sağıroğlu; Gazi Üniversitesi ve Erciyes Üniversitesi Bilgisayar Mühendisliği Bölüm Başkanlığı, ISACA Ankara Chapter Akademi Koordinatörlüğü, Yükseköğretim Kurulu Siber Güvenlik Çalışma Grubu Üyeliği, Bilim Sanayi ve Teknoloji Bakanlığı Yazılım Sektörü Çalışma Grubu Üyeliği, BGD Yönetim Kurulu Üyeliği, IPV6Forum Turkey Başkanlığı, IEEE üyeliği, IEEE Biyometrik Görev Gücü Üyeliği, ACM Üyeliği, Avrupa ETSI Standartları Gözlemci Komisyon Üyeliği, AB Projeleri Danışmanlığı, Sektör Proje Danışmanlığı, Futuretech Genel Müdürlüğü gibi görevleri yürütmektedir. Havelsan, Kişisel Verileri Koruma Kurumu, Bilgi Teknolojileri ve İletişim Kurumu, gibi kurumlarda danışmanlık ta yapmış olan Sağıroğlu, şu anda Gazi Üniversitesi Yapay Zeka ve Büyük Veri Analitiği ve Güvenliği Merkez Müdürlüğü görevini yürütmektedir.

Doç. Dr. Sedat Akleylek

Ondokuz Mayıs Üniversitesi, Bilgisayar Mühendisliği Bölümü Doçent Doktor, Samsun Hesaplamalı Bilimler Doktora Programı A.B.D. Başkanı

Doç. Dr. Sedat Akleylek, İzmir doğumludur. 2004 yılında Ege Üniversitesi Matematik Bölümü'nde lisans eğitimini tamamlamıştır. Öğretim Üyesi Yetiştirme Programı kapsamında sırasıyla 2008 ve 2010 yıllarında yüksek lisans ve doktora çalışmalarını ODTÜ Uygulamalı Matematik Enstitüsü Kriptografi Programı'nda tamamlamıştır. 2012 yılında Almanya Bochum Ruhr Üniversitesi Donanım Güvenliği Grubu'nda, Almanya'da ve 2014-2015 yıllarında Almanya Darmstadt Teknik Üniversitesi Kriptografi ve Bilgisayar Cebri Grubu'nda misafir öğretim üyesi olarak görev almıştır. 2016 yılında Bilgisayar/Bilişim Bilimleri ve Mühendisliği, Bilgi Güvenliği ve Kriptoloji alt alanında doçent ünvanını almıştır. 2011 yılından bu yana Ondokuz Mayıs Üniversitesi, Bilgisayar Mühendisliği Bölümü'nde akademik hayatına devam etmektedir. Doç. Dr. Sedat Akleylek, kuantum sonrası kriptografi, verimli kriptografik hesaplamalar, Boolean fonksiyonlar ve siber güvenlik için uygulamalı kriptografi alanlarında çalışmalarını sürdürmektedir. Ulusal ve uluslararası kapsamda bilgi güvenliği ve kriptoloji alanında TÜBİTAK, Malezya UTAR, KOSGEB, Üniversite-Sanayi İş Birliği Projeleri ve Üniversiteler tarafından desteklenen Bilimsel Araştırma Projelerinde yürütücü, araştırmacı ve danışman olarak görevler almıştır. Doç. Dr. Sedat Akleylek, SCI-Expanded kapsamındaki uluslararası saygın üç dergide bilgi güvenliği ve kriptoloji alan editörlüğü görevlerini sürdürmektedir.

YAZARLAR*

Dr. Enis Karaarslan

MSKÜ Bilgisayar Mühendisliği bölümünde Siber Güvenlik Abd. Başkanıdır. MSKÜ Yapay Zeka disiplininin kurucularındandır ve öğretim üyesidir. İletişim ağları, güvenlik eğitimi ve araştırması için NetSecLab'ı kurdu. MSKÜ blokzincir araştırma grubunda (MSKÜ BcRG) 2017'den beri blokzincirinin potansiyellerini incelemektedir. DS4H (decentralized solutions for humanity) blokzincir araştırma ağının kurucularındandır. Araştırma alanları bilgisayar ağları, siber güvenlik, blokzinciri, veri bilimi, afet yönetimi ve dijital ikizdir. Akıllı sözleşmelerle ve sıfır bilgi kanıt protokolleri ile güçlendirilmiş mahremiyet, merkezi olmayan kimlik ve blokzincir teknolojisinin etkin kullanımı üzerine patent başvuruları, danışmanlık ve yayınları bulunmaktadır.

Linkedin: https://www.linkedin.com/in/enis-karaarslan-1b195617/

Google Scholar: https://scholar.google.com/citations?hl=tr&user=D3dqZ5UAAAAJ

Melih Birim

Melih Birim 2006 yılında Marmara Üniversitesi Bilgisayar Mühendisliği Bölümünden mezun olmuştur. 2016 dan beridir blokzinciri konusunda TUBU ARGE firmasında kurucu ortak olarak araştırmalar yapmaktadır. Ayrıca Consensys GoQuorum sisteminde gönüllü elçi olarak seminerler ve eğitimler düzenlemektedir. Telekom operatörlerinin ve EPİAŞ yeşil enerji sertifika sisteminin blokzinciri mimarı olarak Türkiye'deki ender projelere imza atmıştır. tubu.io, gohammer gibi açık kaynak kodlu blokzincir yazılım araçlarının geliştirilmesinde katkıda bulunmuştur.

Linkedin: https://www.linkedin.com/in/melihbirim/

Google Scholar: https://scholar.google.com/citations?hl=tr&user=0so1uMAAAAAJ

* Yazarlar bölüm sıralamasına göre verilmiştir.

Dr. Murat Osmanoğlu

-Ankara Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Ankara.

Yüksek lisans derecesini Uygulamalı Matematik alanında Yıldız Teknik Üniversitesi'nden ve doktora derecesini bilgisayar bilimleri alanında Connecticut Üniversitesi'nden almıştır. Doktorada seviye kavramının kriptografik ilkellere uygulanması üzerine çalışmalar yapmıştır. 2017'den beri Ankara Üniversitesi Bilgisayar Mühendisliği Bölümünde araştırma görevlisi doktor olarak görev yapmaktadır. Çalışma alanları arasında eşik kriptografi, giz paylaşım şemaları, blokzincir teknolojisi, aranabilir şifreleme ve bilgi güvenliği bulunmaktadır.

Dr. Öğr. Üyesi Serkan AYVAZ

sayvaz@yildiz.edu.tr

-Yıldız Teknik Üniversitesi Bilgisayar Mühendisliği, Dr. Öğr. Üyesi, İstanbul

Yıldız Teknik Üniversitesi Bilgisayar Mühendisliği bölümünde doktor öğretim üyesi olarak görev yapmaktadır. Öncesinde Bahçeşehir Üniversitesi Yazılım Mühendisliği bölümünde doktor öğretim üyesi olarak görev yapmıştır. Doktorasını 2015 yılında Amerika Birleşik Devletleri Kent State Üniversitesi'nde bilgisayar bilimi alanında tamamladı. 2008 yılında aynı üniversiteden bilgisayar teknolojisi yüksek lisans programından mezun oldu. Lisans derecesini Bahçeşehir Üniversitesi matematik ve bilgisayar bilimleri bölümünden aldı. Büyük veri analitiği, yapay zeka, blokzincir ve semantik web alanlarında araştırma yapmaktadır.

Salih Cemil Çetin

salihcemil@gmail.com

-Softtech A.Ş. - Kıdemli Blokzincir ArGe Mühendisi, İstanbul

Softtech A.Ş.'de blokzincir yazılım araştırma-geliştirme mühendisi olarak görev almaktadır. Uludağ Üniversitesi Elektronik Mühendisliği lisansının ardından Bahçeşehir Üniversitesi Bilgisayar Mühendisliği programında yüksek lisansını tamamlamıştır. Dağıtık defter teknolojileri ve uygulamaları ile ilgilenmektedir. Yüksek lisans tez projesi olarak bir blokzinciri protokolünde eşler arası dijital varlık aktarımına izin veren bir uygulama geliştirmiştir. Enerji, sermaye piyasaları, finans ve telekomünikasyon sektörlerinde olmak üzere 9 yıllık mühendislik tecrübesi bulunmaktadır.

Dr. Mehmet AYDAR

mehmetaydar@gmail.com

- Ford Otosan Arge Merkezi Kurumsal Mimar, İstanbul

Yaklaşık 15 yıldır bilişim sektöründe tecrübeli olan Mehmet Aydar, Ford Otosan Arge Merkezinde kıdemli kurumsal mimar olarak Blokzincir ve bağlı araç teknolojileri alanında görev yapmaktadır. Huawei Türkiye Araştırma Geliştirme Merkezi'nde Kıdemli Araştırma Mühendisi olarak Blokzincir ve Makine öğrenimi üzerine çalışmalar yapmıştır. 2015 yılında Kent State Universitesi Bilgisayar bilimleri bölümünden Doktora derecesini almıştır. Cleveland Clinic Kalp ve Damar Enstitüsü Medikal Araştırmalar merkezinde Sistem analisti olarak görev yapmıştır. İlgi alanları arasında Blokzincir, Yapay Zeka, IoT ve Anlamsal Ağlar bulunmaktadır.

Murat KARAKUŞ
Dr. Öğr. Üyesi
Bayburt Üniversitesi
muratkarakus@bayburt.edu.tr

Dr. Murat KARAKUŞ, 2009 yılında Süleyman Demirel Üniversitesi Matematik Bölümü'nden lisans derecesini, 2013 yılında Amerika Birleşik Devletleri'nde Michigan Üniversitesi-Flint (University of Michigan-Flint) Üniversitesi'nden Bilgisayar Bilimi ve Bilgi Sistemleri alanında Yüksek Lisans derecesini ve 2018 yılında yine Amerika Birleşik Devletleri'nde Purdue Üniversitesi'nden (Purdue University) Bilgisayar Bilimleri alanında doktora derecesi almıştır. Halen Türkiye'de Bayburt Üniversitesi'nde Dr. Öğr. Üyesi olarak görev yapmaktadır. ACM SIGITE 2011 konferansında En İyi Bildiri Ödülü'nü almıştır. Kendisini 20'den fazla hakemli yayını vardır ve saygın dergiler ve konferanslar için hakem olarak hizmet vermektedir. Şu anki araştırma ilgi alanları arasında blokzinciri teknolojisi, Yazılım Tanımlı Ağlar gibi yeni nesil ağ mimarileri, ağ mimarilerinin ve tasarımlarının ekonomik analizi, ağ hizmetlerinin fiyatlandırılması, ağ mimarilerinin ölçeklendirilmesi, ağlar arası iletişimde hizmet kalitesi (QoS), yönlendirme ve ağ güvenliği gibi alanlar bulunmaktadır.

Enis Konacaklı

Enis Konacaklı, 19 yıldır CIS ve BT ekiplerini, sistemlerini, hizmetlerini satın almadan teslimata ve yaşam döngüsü sürdürmelerine kadar yönetmektedir. 2017 yılından bu yana MSKÜ BcRG (Blockchain Research Group) üyesi olarak ulusal güvenlik ve askeri uygulamalar için blokzinciri teknolojisinin potansiyel faydalarını araştırmaktadır. Ulusal güvenlikte blokzinciri teknolojisinin kullanımı üzerine yüksek lisans tezi bulunmaktadır. Araştırma alanları bilgisayar ağları, siber güvenlik ve blokzinciridir.

Linkedin: https://www.linkedin.com/in/enis-k-b89302168/

Google Scholar: https://scholar.google.com/citations?user=WAAINIUAAAAJ&hl=tr&oi=sra

Arş. Gör. Kübra SEYHAN

Ondokuz Mayıs Üniversitesi, Bilgisayar Mühendisliği Bölümü
Araştırma Görevlisi, Samsun

Arş. Gör. Kübra Seyhan, Gümüşhane doğumludur. 2010 yılında Gebze Yüksek Teknoloji Enstitüsünde başladığı bilgisayar mühendisliği lisans eğitimini 2016 yılında Karadeniz Teknik Üniversitesi Bilgisayar Mühendisliği Bölümünde tamamladı. 2017 yılında Ondokuz Mayıs Üniversitesi Bilgisayar Mühendisliği Bölümünde başladığı yüksek lisans eğitimini 2020 yılında tamamladı. Aynı yıl Şubat ayı itibariyle Ondokuz Mayıs Üniversitesi Lisansüstü Eğitim Enstitüsü Hesaplamalı Bilimler A.B.D. doktora programında başladığı doktora eğitimi halen devam etmektedir. Kuantum sonrası kriptografi, bilgi güvenliği ve kriptoloji gibi çalışma alanlarında akademik çalışmalarını sürdürmektedir. Ulusal ve uluslararası kapsamda bilgi güvenliği ve kriptoloji alanında TÜBİTAK tarafından desteklenen Bilimsel Araştırma Projelerinde araştırmacı olarak görev almıştır.

YAZAR EPOSTA: kubra.seyhan@bil.omu.edu.tr,

Melek Çil

1988 Antalya/Alanya doğumludur. 2010 yılında Dokuz Eylül Üniversitesi Fen Fakültesi Matematik bölümünden mezun olmuştur. 2013 yılında Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Matematik Anabilimdalı'nda yüksek lisans eğitimini tamamlamıştır. 2013 yılından beri aynı üniversitede sonlu cisimler ve kriptoloji üzerine doktora çalışmalarına devam etmektedir. 2011-2018 yılları arasında Süleyman Demirel Üniversitesinde, 2020 yılından bu yana da Mehmet Akif Ersoy Üniversitesi Matematik bölümünde Araştırma Görevlisi olarak çalışmaktadır.

Doç. Dr. Barış Bülent Kırlar

Süleyman Demirel Üniversitesi, Matematik Bölümü, Cebir ve Sayılar Teorisi A.B.D. Öğretim Üyesi

Doç. Dr. Barış Bülent Kırlar, Konya doğumlu olup aslen Niğde, Ulukışlalıdır. 2001 yılında Ankara Üniversitesi Matematik Bölümü'nde lisans eğitimini, 2005 yılında Orta Doğu Teknik Üniversitesi Matematik Bölümü'nde yüksek lisans eğitimini, 2010 yılında da Orta Doğu Teknik Üniversitesi Kriptografi Programı'nda doktora eğitimini tamamlamıştır. 2018 yılında Matematik, Cebir ve Sayılar Teorisi alanında doçent unvanını almıştır. 2011 yılından bu yana Süleyman Demirel Üniversitesi, Matematik Bölümü'nde akademik çalışmalarına devam etmektedir. Doç. Dr. Barış Bülent Kırlar, eliptik eğri kriptografisi, eşleme tabanlı kriptografi, açık anahtarlı kriptosistemlerin verimli uygulamaları, sonlu cisimlerin uygulamaları ve kuantum sonrası kriptografi alanlarında çalışmalarını sürdürmektedir. Ulusal ve uluslararası kapsamda bilgi güvenliği ve kriptoloji alanında TÜBİTAK, Üniversite-Sanayi İş Birliği Projeleri ve üniversiteler tarafından desteklenen Bilimsel Araştırma Projelerinde yürütücü ve araştırmacı olarak görev almıştır.

Öğr. Gör. Dr. Muhiddin Uğuz

Orta Doğu Teknik Üniversitesi Matematik Bölümü doktor öğretim görevlisi, ODTÜ Uygulamalı Matematik Enstitüsü Kriptografi Anabilim Dalı bağlı öğretim görevlisi
muhid@metu.edu.tr

Ankara doğumlu olan Dr. Muhiddin Uğuz, 1990 yılında ODTÜ Matematik Bölümünden lisans diplomasını aldıktan sonra 1992 yılında Michigan State Üniversitesinde Master, ve ardından tekrar ODTÜ Matematik bölümünde Diferansiyel geometri ve 4 manifoldlar konusunda doktorasını yapmış ve 2002 yılından beri de ODTÜ Matematik Bölümünde Dr. Öğr.Gör. olarak çalışmaktadır. Matematik bölümde verdiği birçok lisans dersi ve koordinatörlüğünü yaptığı Kalkülüs servis dersleri dışında, ODTÜ Uygulamalı Matematik Enstitüsü Kriptoloji Anabilimdalında yüksek lisans ve doktora seviyelerinde çeşitli dersler vermektedir. Simetrik Kriptoloji alanında, özellikle de rassallık testleri konusunda akademik çalışmalarına devam etmektedir.

Mehmet Emin Gönen

TÜBİTAK-BİLGEM-Ulusal Elektronik ve Kriptoloji Araştırma Enstitüsü

Uzman Araştırmacı

Mehmet Emin Gönen, 2011 yılında Boğaziçi Üniversitesi Matematik Bölümü'nde lisans eğitimini tamamlamıştır. 2011-2013 yılları arasında Bahçeşehir Üniversitesinde araştırma görevlisi olarak çalışmış ve yüksek lisans derecesini, 2013 yılında Bahçeşehir Üniversitesi Uygulamalı Matematik Bölümünden almıştır. 2014 yılında Gebze Teknik Üniversitesi Matematik Bölümünde doktora eğitimine başlamıştır. Doktora tez çalışmasına, eliptik eğri ve izojen tabanlı kriptografik protokoller ve algoritmalar konusunda devam etmektedir. 2013 yılında, TÜBİTAK BİLGEM'de Kriptografik Mimari ve Algoritmalar Bölümü'nde araştırmacı olarak çalışmaya başlamıştır. BİLGEM bünyesindeki bilgi güvenliği ve kriptoloji alanındaki birçok projede, kriptografik algoritma ve protokol tasarımında yer almıştır. Hali hazırda Uzman Araştırmacı ve Proje Yürütücüsü olarak BİLGEM UEKAE'de görevine devam etmektedir.

Ferhat Karakoç

Ferhat Karakoç doktora, yüksek lisans ve lisans derecelerini İstanbul Teknik Üniversitesinden almıştır. 2020 yılından beri Ericsson'da bilgi güvenliği konusunda araştırmacı olarak çalışmaktadır. Ericsson'dan önce bilgi güvenliği, kriptoloji ve yazılım alanlarında çeşitli özel sektör ve kamu kurumlarında çalışmıştır. Ayrıca bilgi güvenliği ve kriptoloji konularında üniversitelerde çeşitli dersler vermiştir. Ferhat Karakoç kriptoloji ve bilgi güvenliği konusunda bir çok akademik makalede yazar olarak yer almaktadır.

Orhun Kara

Orhun Kara lisans, yüksek lisans ve doktora derecelerini İ.D. Bilkent Üniversitesi Matematik bölümünden almıştır. Uzun yıllar TÜBİTAK BİLGEM'de kriptografik algoritma ve protokollerin tasarımları ve analizleri konusunda araştırmacı olarak görev yapmıştır. 2020 Ağustosundan beri İYTE matematik bölümünde öğretim üyesidir ve kriptoloji üzerine çalışmaktadır.

Dr. Öğretim Üyesi Ahmet SINAK

Matematik ve Bilgisayar Bilimleri Bölümü

Necmettin Erbakan Üniversitesi

Ahmet Sınak, lisans derecesini Muğla Sıtkı Koçman Üniversitesi Fen Fakültesi Matematik Bölümünden 2009 yılında almıştır. Yüksek Lisans ve Doktora derecelerini Orta Doğu Teknik Üniversitesi (ODTÜ) Uygulamalı Matematik Enstitüsü (UME) Kriptografi bölümünden sırasıyla 2012 yılında ve 2017 yılında almıştır. 2018 yılında TÜBİTAK-BİDEB 2219 burs programı kapsamında Paris 8 Üniversitesi (Paris, Fransa) Matematik bölümünde doktora sonrası araştırmacı olarak akademik çalışmalar yapmıştır. Lisansüstü eğitimi süresince (2012–2018 yılları arasında) ODTÜ UME'de araştırma görevlisi olarak görev yapmıştır. 2018 – 2020 yılları arasında Necmettin Erbakan Üniversitesi Fen Fakültesi Matematik ve Bilgisayar Bilimleri Bölümü'nde doktor araştırma görevlisi olarak görev yaptıktan sonra Ocak 2020 tarihinden bu yana aynı bölümde Dr. Öğretim Üyesi olarak görev yapmaktadır.

Akademik çalışmaları arasında sonlu cisimler üzerindeki kriptografik fonksiyonlar ve doğrusal kodların yanı sıra sır paylaşım şemalarının analizi ve tasarımı da yer almaktadır.

Arş. Gör. Meryem SOYSALDI ŞAHİN

Ondokuz Mayıs Üniversitesi, Bilgisayar Mühendisliği Bölümü
Araştırma Görevlisi, Samsun

Nevşehir doğumlu olan Soysaldı, 2013 yılında Fırat Üniversitesi Bilgisayar Mühendisliği Bölümü'nde lisans eğitimini tamamlamıştır. Öğretim Üyesi Yetiştirme Programı kapsamında 2018 yılında Ondokuz Mayıs Üniversitesi Bilgisayar Mühendisliği Bölümü'nde yüksek lisans eğitimini tamamlamış ve sonrasında Ondokuz Mayıs Üniversitesi Hesaplamalı Bilimler Anabilim Dalı'nda doktora çalışmalarına başlamıştır. Kuantum sonrası kriptografi ve bilgi güvenliği için uygulamalı kriptografi alanlarında çalışmalarını sürdürmektedir.

Dr. Hüseyin BODUR

Düzce Üniversitesi

huseyinbodur@duzce.edu.tr

2012 yılında Pamukkale Üniversitesi Bilgisayar Mühendisliği bölümünden mezun oldu. Aynı yıl Düzce Üniversitesine Araştırma Görevlisi olarak atandı. 2020 yılında Düzce Üniversitesi Elektrik-Elektronik ve Bilgisayar Mühendisliği anabilim dalından doktora derecesini aldı. Bilgi Güvenliği ve Kriptoloji alanlarında çalışmalar yapmaktadır.

Prof. Dr. Resul KARA

Düzce Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü

resulkara@duzce.edu.tr

2009 yılında Sakarya Üniversitesi Elektrik Elektronik Mühendisliği anabilim dalından doktora derecesini aldı. Düzce Üniversitesi Bilgisayar Mühendisliği Bölümünde 2009-2012 yılları arasında Doktor Öğretim Üyesi, 2012-2017 yılları arasında Doçent, 2017 yılından itibaren Profesör unvanıyla görev yaptı. Bilgisayar ve İletişim Ağları, Bilgi Sistemleri ile Bilgi Güvenliği ve Kriptoloji alanlarında çalışmalar yapmaktadır.