Avrupa Bilim ve Teknoloji Dergisi
Özel Sayı, S. 251-259, Nisan 2020
© Telif hakkı EJOSAT'a aittir
Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
Special Issue, pp. 251-259, April 2020
Copyright © 2020 EJOSAT
Research Article

Hatalar ile Öğrenme Tabanlı Anahtar Kapsülleme Protokolleri İçin Uygulama Atakları ve Savunma Yöntemleri*

Bilge Kağan Yazar[1**], Erdem Alkım[2]

[1] Ondokuz Mayıs Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Samsun, Türkiye (ORCID: 0000-0003-2149-142X)
[2] Ondokuz Mayıs Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Samsun, Türkiye (ORCID: 0000-0003-4638-2422)

(Konferans Tarihi: 5-7 Mart 2020)

(**DOI:** 10.31590/ejosat.araconf32)

ATIF/REFERENCE: Yazar, B. K. & Alkım, E. (2020). Hatalar ile Öğrenme Tabanlı Anahtar Kapsülleme Protokolleri İçin Uygulama Atakları ve Savunma Yöntemleri. *Avrupa Bilim ve Teknoloji Dergisi*, (Özel Sayı), 251-259.

Öz

Şifreleme sistemleri gizli değerlerin korunmasını sağlamakta olan sistemlerdir. Bu sistemlere karşı son yıllarda kuantum bilgisayarlar üzerinde çok sayıda araştırmalar yapılmaktadır. Yeterli işlem kapasitesine sahip kuantum bilgisayarlar üretildiğinde günümüzde kullanılmakta olan açık anahtarlı şifreleme sistemlerinin güvensiz duruma geleceği düşünülmektedir. Bu durum internet üzerindeki haberleşmenin güvenliğini tehlikeye atmaktadır. Bu durumdan dolayı, kuantum bilgisayarlara karşı güvenli sistemlerin oluşturulması için kuantum bilgisayarların günlük hayata girmesinin beklenmemesi gerektiği düşünülmektedir. NIST bu doğrultuda hem günümüz hem de kuantum bilgisayarlardan gelecek ataklara karşı dayanıklı kriptosistemlerin üretilmesi için bir standartlaştırma proje başlatmıştır. Bu süreçte birçok sistem önerilmiştir. Kafes tabanlı sistemler en çok gelecek vadeden sistemler olarak ön plana çıkmıştır. Kafes tabanlı kriptosistemler kafes yapısı üzerinde tanımlı olan LWE gibi problemlerin çözümlerinin zorluğuna dayanarak önerilmiş olan sistemlerdir. Fakat yan kanal saldırıları ile kriptografik sistemler çalışırken çıkan bilgiler toplanarak gizli anahtar hakkında bilgi edinilebilmektedir. Bu durumdan dolayı NIST kuantum bilgisayarlara karşı dayanıklı şifreleme sistemleri için yaptığı çağrının yanı sıra, yapılacak sistemlerin minimum maliyetle yan kanal saldırılarına karşı dayanıklı olmasını gerektiğini söylemiştir. 2016 yılında Bindel ve arkadaşlarının yaptıkları bir çalışmada uygulama atağı olarak sınıflandırılan ve bir yan kanal saldırı çeşidi olan hata ataklarını önermişlerdir. 2018 yılında Han ve arkadaşları yaptıkları bir çalışmada, 2016 yılında önerilen yöntemleri kullanarak Lizard protokolünde gizli anahtarı elde edebildiklerini söylemişlerdir ve bu ataklara karşı alınabilecek önlemlerden bahsetmişlerdir. Bu çalışmada; önceden yapılan çalışmalarda önerilenler ve kriptosistemlerin uygulamalarında olması gereken özellikler doğrultusunda Lizard protokolünün uygulaması üzerinde yapılan değişikliklerden ve bu değişikliklerin uygulama üzerindeki etkilerinden bahsedilmiştir. Protokolün uygulamasının güvenliği artırılmış ve gerçekleştirilen bazı hata ataklarına karşı önlem alınmıştır. Yapılan değişikliklerden sonra protokolün çalışma süresinde hızlanma olduğu gözlemlenmiştir.

Anahtar Kelimeler: Kuantum Sonrası Kriptografi, Kafes Tabanlı Kriptografi, Hatalar ile Öğrenme, Yan Kanal Saldırıları, Hata Atakları.

Implementation Attacks and Their Countermeasures For Learning With Errors Based Key Encapsulation Mechanishms

Abstract

Encryption systems are built to protect shared secrets. Recently, several research projects conducted on the usability of quantum computers on the cryptanalysis of those systems. The general understanding is that the current public-key encryption systems can be broken when quantum computers with sufficient processing capacity produced. Because this endangers the security of the communication on the internet, NIST has started a standardization project to produce cryptosystems resistant to attacks from both classical and quantum computers. Many systems have been proposed in this project. Lattice-based systems, which are based on the

* Bu makale ***International Conference on Access to Recent Advances in Engineering and Digitalization (ARACONF 2020)*** de sunulmuştur.

** Sorumlu Yazar: Ondokuz Mayıs Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Samsun, Türkiye, ORCID: 0000-0003-2149-142X, bilgekaganyazar@gmail.com

difficulty of solving problems such as LWE defined on the lattices, have developed great attention as the most promising systems. However, information about the secret key can be obtained by collecting the side-channel information obtained during cryptographic systems are working. For this reason, NIST commented that proposals to its call for encryption systems that are resistant to quantum computers should require a small cost to make them resistant to side-channel attacks if they are not resistant by design. In 2016, Bindel et al. have proposed fault attacks targeting lattice-based schemes, which are classified as implementation attacks, and a type of side-channel attacks. In 2018, Han et al. showed that they were able to obtain the secret key in the Lizard protocol using the methods proposed in 2016 and proposed some countermeasures to prevent such attacks. In this study; the implementation of the Lizard protocol changed and the effects of these changes on the implementation are discussed with regards to the features suggested in the previous studies and the features that the cryptosystems should have. Not only the implementation security of the protocol has been increased and some countermeasures proposed to prevent fault attacks but also it was observed that the protocol's performance accelerated.

Keywords: Post-quantum Cryptography, Lattice-based Cryptography, Learning With Errors, Side-channel Attacks, Fault Attcaks

1. Giriş

Günümüzde internet ortamında güvenli ve düzgün veri iletişimi için açık anahtarlı şifreleme sistemleri kullanılmaktadır. Kullanılan bu sistemlerin güvenliklerinin birçoğu çarpanlara ayırma ve ayrık logaritma problemlerinin zorluklarına dayanmaktadır. En çok bilinen açık anahtarlı şifreleme sistemlerinden birisi olan RSA sisteminde anahtar boyutu 2048-bit seçildiğinde, günümüz bilgisayarlarında bu boyutta bir şifreyi çözebilecek işlem gücü bulunmadığı düşünülmektedir. Buradan anlaşılacağı üzere; günümüz bilgisayarlarına karşı şu anda kullanılan açık anahtarlı şifreleme sistemleri güvenliği sağlayabilmektedir.

Son yıllarda klasik bilgisayarlara göre çözülmesi çok zor olan bu problemleri çözmek için kuantum fiziğini temel alan bilgisayarlar üzerinde araştırmalar yapılmaktadır. Yeterli işlem kapasitesine sahip kuantum bilgisayarlar üretildiğinde günümüzde kullanılmakta olan açık anahtarlı şifreleme sistemlerinin temelinde bulunan birçok zor problemin kolayca çözülebileceği düşünülmektedir (Bernstein, 2009; Chen vd, 2016). 1994 yılında Peter Shor yaptığı bir çalışmada; çarpanlara ayrıma ve ayrık logaritma problemlerini kuantum bilgisayarlar ile birlikte polinom zamanda çözebilen bir algoritma önermiştir. Bu durum internet üzerindeki veri iletişiminin gizliliğini tehlikeye atan bir durumdur. Bundan dolayı NIST (Amerika Standartları ve Teknoloji Enstitüsü) hem günümüz bilgisayarlarından hem de kuantum bilgisayarlardan gelecek ataklara karşı dayanıklı açık anahtarlı şifreleme şemaları oluşturulması ve standart oluşturulabilmesi adına bir proje başlatmıştır. Önerilen sistemler doğrultusunda kuantum bilgisayarlar günlük hayata tamamen girmeden önlem alınması amaçlanmaktadır.

Kuantum sonrası kriptografi standartlaştırma (PQCStandardization) projesi toplamda 5 yıllık bir süreç olarak belirlenmiştir ve şuanda ikinci tur incelemeleri devam etmektedir. İlk turda 82 tane sistem önerilmiştir ve bunlardan 26 tanesi ikinci tura geçmiştir (Computer Securtiy Division, 2017). Bu süreçte kafes tabanlı sistemler ön plana çıkmıştır ve önerilen sistemlerin çoğu kafes tabanlı sistemlerdir. Kafes tabanlı sistemler; kafes yapısı üzerinde tanımlanmış olan en kısa vektör problemi (Shortest Vector Problem – SVP), en yakın vektör problemi (Closest Vector Problem – CVP), hatalar ile öğrenme (Learning With Errors – LWE), gibi problemlerin çözümlerinin zorluklarını temel almakta olan sistemlerdir.

2005 yılında Oded Regev LWE problemini ve bu problemi temel alan bir açık anahtarlı şifreleme sistemi önermiştir ve bu problemin en kötü durumdaki kafes problemleri kadar zor olduğunu göstermiştir (Regev, 2005). Devamında LWE problemi birçok kafes tabanlı kriptosistemin oluşturulmasında temel olmuştur. 2011 yılında Lindner ve Peikert tarafından yapılan bir çalışmada, LWE problemi içerisinde dikdörtgen şeklinde matrisler yerine kare matris kullanmak gibi bazı değişiklikler yapılarak daha verimli bir şifreleme sistemi önerilmiştir (Lindner ve Peikert, 2011).

2011 yılında Banerjee ve arkadaşları tarafından LWE problemine farklı bir bakış açısı olarak yuvarlayarak öğrenme (Learning With Rounding – LWR) problemi önerilmiştir. Bu problem LWE probleminin rastgele olmayan bir versiyonudur (Banerjee vd, 2011). Düzgün parametreler ile kullanıldığında en az LWE problemi kadar zor bir problem olduğu düşünülmektedir (Banerjee vd, 2011. LWR problemi ile LWE problemine göre daha hızlı şifreleme yapılabilmektedir.

Çalışmanın ana konusunu oluşturan, LWE ve LWR problemlerini temel alan ve standart kafesleri kullanan açık kaynaklı Lizard protokolü Cheon ve arkadaşları tarafından 2018 yılında önerilmiştir. Bunlara ek olarak standart kafesleri ve LWE problemini kullanan FrodoKEM (Naehrig vd, 2017), Emblem (Seo vd, 2017), Lotus (Le Trieu Phong vd, 2017) protokolleri ve LWE tabanlı bir imzalama şeması olan BLISS (Ducas vd, 2013) gibi sistemler literatürde yer almaktadır.

Şifreleme sistemleri gizli anahtarı, düz metnin veya şifre metnin saldırgan tarafından bilindiği durumlarda, saldırgandan korumakta olan sistemlerdir. Fakat yan kanal saldırıları (uygulama saldırıları) ile, kriptografik sistemler çalışırken sızan bilgiler toplanarak gizli anahtar hakkında bilgi edinilebilir (Taha ve Eisenbarth, 2015). Bu durumdan dolayı yan kanal saldırılarına dayanıklı şifreleme sistemlerine ihtiyaç duyulmaktadır. NIST standartlaştırma projesi için belirttiği kriterlerde; oluşturulacak sistemlerin minimum maliyetle yan kanal saldırılarına karşı dayanıklı olmaları gerektiğini belirtmiştir.

2016 yılında Bindel ve arkadaşları kafes tabanlı imzalama protokollerinin bazılarının uygulamaları üzerinde yaptıkları bir çalışmada; bir uygulama saldırısı çeşidi olan hata atakları (Fault Attack) yöntemlerinden bahsetmişlerdir. Bindel ve arkadaşları bu çalışmada kafes tabanlı imzalama protokolleri olan BLISS, ring-TESLA, GLP ve bu protokollerin uygulamalarındaki zayıflıkları incelemişlerdir. Bu incelemeler doğrultusunda bu saldırılara karşı alınabilecek önlemlerden bahsetmişlerdir. Devamında 2018 yılında Han ve arkadaşları yaptıkları bir çalışmada; Lizard protokolüne bu hata ataklarını uygulamışlardır ve gizli anahtarı elde ettiklerini söylemişlerdir. Bu doğrultuda protokolün uygulaması üzerinde yazılımsal değişiklikler ile birlikte alınabilecek bazı önlemler önermişlerdir.

(Bindel vd, 2016), (Han vd, 2018) çalışmalarından yararlanılarak ve kriptografik protokollerin uygulamalarında olması gereken özellikler doğrultusunda, bu çalışmada Lizard protokolünün güvenliğinin ve verimliliğinin artırılabilmesi için uygulamadaki bazı eksiklikler giderilmeye çalışılmıştır. Bu eksiklikler giderilirken LWE problemini kullanan farklı protokollerde kullanılan yöntemler incelenmiştir (çoğunlukla FrodoKEM). Lizard protokolünün uygulaması üzerinde hata ataklarına karşı olan güvenliği ve uygulamanın genel verimliliğini artırmak için yapılan değişikliklerden bahsedilmiştir.

2. Materyal ve Metot

2.1. Hatalar İle Öğrenme (LWE)

LWE problemi kafes yapısı üzerinde tanımlanmış olan kuantum bilgisayarlarla bile çözülemeyeceği düşünülen problemlerden bir tanesidir. 2005 yılında Oded Regev tarafından önerilmiştir ve devamında bu problemi temel alan birçok kriptografik sistem önerilmiştir. LWE problemini kuantum sonrası algoritmalar dâhil olmak üzere şuan için polinom zamanda çözebilen bir algoritmanın olmadığı düşünülmektedir.

Gizli bir $s \in \mathbb{Z}_q^n$, $A_{s,\chi} \in \mathbb{Z}_q^n \times \mathbb{Z}_q^n$; rastgele olarak seçilen bir $a \in \mathbb{Z}_q^n$ vektörü ile elde edilen olasılık dağılımı, χ değerine göre elde edilen hata değeri $e \in \mathbb{Z}_q^n$, çıktı olarak $(a, < a, s > + e) \bmod q$ olsun.

$\text{Arama} - \text{LWE}$ problemi; elimizdeki $A_{s,\chi}$ dağılımından elde edilen m tane $a_i, b_i \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ ve rastgele olarak üretilmiş $s \in \mathbb{Z}_q^n$ değerleri (s her örnek için sabit) üzerinden s değerini bulma problemidir (Peikert, 2016).

$\text{Karar Verme} - \text{LWE}$ problemi; elimizde $A_{s,\chi}$ dağılımından elde edilen m tane a_i, $b_i \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ ve rastgele olarak üretilmiş $s \in \mathbb{Z}_q^n$ değerleri (bütün örnekler için sabit) olsun. Üretilen LWE örnekleri ile rastgele olan değerleri birbirinden ayırma problemidir (Peikert, 2016).

Yeterince büyük bir q değeri ile Arama – LWE problemi en az CVP problemi kadar zor bir problemdir.

2.2. Yuvarlayarak Öğrenme (LWR)

LWE probleminin rastgele olmayan şekli olan LWR problemi Banerjee ve arkadaşları tarafından 2011 yılında önerilmiştir. Burada rastgele olmayan ile anlatılmak istenen; problem içerisindeki hata değerinin rastgele olmayışıdır.

Gizli bir $s \in \mathbb{Z}_q^n$, $A_{s,\chi} \in \mathbb{Z}_q^n \times \mathbb{Z}_q^n$; rastgele olarak seçilen bir $a \in \mathbb{Z}_q^n$ vektörü ile elde edilen olasılık dağılımı, çıktı olarak $(a, \left\lfloor \frac{p}{q} \cdot (\langle a, s \rangle \bmod q) \right\rceil)$ olsun. $[\cdot]$ işlemi en yakın tam sayıya yuvarlama işlemini temsil etmektedir.

LWR probleminin LWE problemi gibi iki çeşidi bulunmaktadır. Bunlar; $\text{Arama} - \text{LWR}$ ve $\text{Karar Verme} - \text{LWR}$ problemleridir. Tanımları LWE probleminin çeşitlerinin tanımları ile çok benzer olduğundan tekrar yapılmamıştır. Tek farklılık LWR örneklerinin üretimindedir.

Uygun parametreler ile kullanıldığı zaman ve örnek sayısı sınırlı olduğunda LWR problemi en az LWE problemi kadar zor bir problem olarak görülmektedir (Banerjee vd, 2011; Alwen vd, 2013).

2.3. Lizard

Lizard protokolü LWE ve LWR problemlerini birleştirerek kullanmakta olan bir açık anahtarlı şifreleme/kapsülleme protokolüdür (Cheon vd, 2018). Açık anahtar m tane n boyutlu LWE örneğinden ve $n + \ell$ tane m boyutlu LWR örneğinden oluşmaktadır. ℓ değeri burada düz metin vektörlerinin boyutunu ifade etmektedir. Protokolün genel yapısı şu şekildedir;

İlklendirme

1. m, n, q, p, t ve ℓ pozitif tamsayıları seçilir.
2. Gizli anahtar için $D_s \in \mathbb{Z}^n$, geçici gizli değer için $D_r \in \mathbb{Z}^m$ dağılımları ve ayrık Gauss dağılımı (χ_σ) için σ değeri seçilir.
3. Çıktı olarak $params \longleftarrow (m, n, q, p, t, \ell, D_s, D_r, \sigma)$

Anahtar Üretimi (params)

1. Rastgele $A \in \mathbb{Z}_q^{m \times n}$ matrisi üretilir.
2. Reddetme örneklemesi ile gizli $S = (s_1 \parallel \cdots \parallel s_l) \in \mathbb{Z}^n$ matrisi D_s dağılımından bağımsız olarak üretilir.
3. Belirli bir dağılım tablosu üzerinden (Cumulative Distribution Table) $E = (e_1 \parallel \cdots \parallel e_l) \in \chi_q^{m \times l}$ hata matrisi üretilir.
4. $B \longleftarrow AS + E \in \mathbb{Z}_q^{m \times l}$ işlemi ile açık anahtarın bir parçası üretilir.
5. Açık anahtar $pk \longleftarrow (A \parallel B) \in \mathbb{Z}_q^{m \times (n+l)}$ ve gizli anahtar $sk \longleftarrow S \in \mathbb{Z}^{n \times l}$ çıktı olarak verilir.

Şifreleme (m)

1. $m \in \mathbb{Z}_t^l$ düz metni için, $r \in \mathbb{Z}^m$ şeklinde bir vektör D_r dağılımında seçilir.

2. $c_1' \longleftarrow A^T r$ ve $c_2' \longleftarrow B^T r$ vektörleri elde edilir.
3. $c_1 \longleftarrow [(\frac{p}{q}) \cdot c_1'] \in \mathbb{Z}_p^n$, $c_2 \longleftarrow [(\frac{p}{t}) \cdot m + (\frac{p}{q}) \cdot c_2'] \in \mathbb{Z}_p^l$ vektörleri hesaplanır.
4. Çıktı olarak $c \longleftarrow (c_1, c_2) \in \mathbb{Z}_p^{n+l}$ şifre metni verilir.

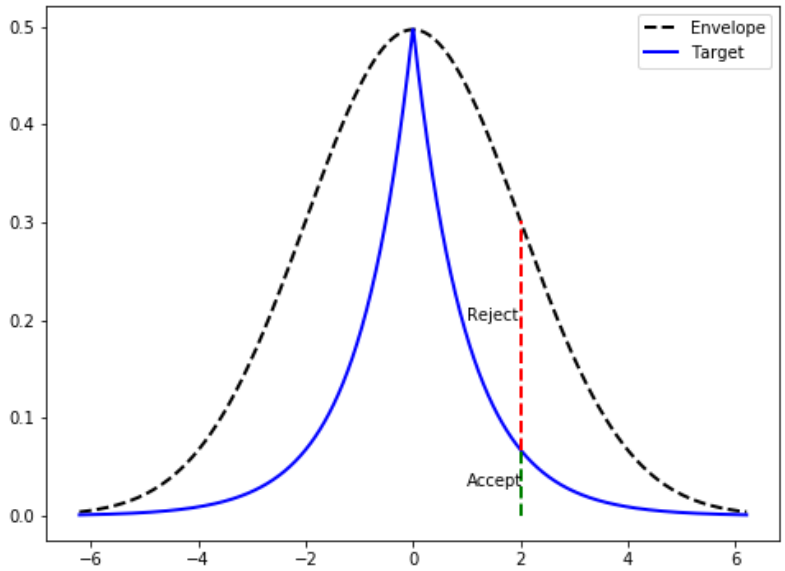
Şifre Çözme (c)

1. Şifre metinler ile $m' \longleftarrow [\frac{t}{p}(c_2 - S^T c_1)]$ işlemi yapılır.
2. m ve m' değerleri eşit mi değil mi kontrol edilir.

Lizard protokolü LWE ve LWR problemlerinin çözümlerinin zor olduğu varsayımı ile kuantum sonrası için 128-bit ve IND-CPA güvenliği sağlamaktadır (Cheon vd, 2018).

2.4. Reddetme Örneklemesi (Rejection Sampling)

Kafes tabanlı protokollerde ayrık Gauss dağılımı kullanılarak rastgele hata üretimi veya rastgele değerlerin üretilmesi işlemi yapılmaktadır. Reddetme örneklemesi kafes tabanlı protokollerde hata örneklemesi için önerilmiş olan ilk yöntemdir (Gentry vd, 2008). Bu yöntem ile bir olasılık dağılımı (f) verildiğinde isteğe bağlı bir hedef dağılım (g) üretilebilmektedir (Genrty vd, 2008; Howe vd, 2016). g dağılımından bir örnek $f(x)/(M \cdot g(x))$ $(M \in \mathbb{R}^+)$ olasılığı ile $f(x)/g(x)$ sınırı altında kabul edilebilir. Ayrık Gauss dağılımından örnek üretmek için reddetme örneklemesi kullanılırken, düzgün bir rastgele değer x, rastgele bir değer $\mu \in [0,1)$ seçilir ve $\mu < f_\sigma(x)$ olup olmadığına bakılır (Howe vd, 2016). Eğer rastgele μ değeri Gauss dağılımı eğrisinin altında ise örnek kabul (accept), eğrinin üstünde ise örnek reddelir (reject). Bu yöntem kullanılırken, bir örnek kabul edilene kadar ortalama olarak $2\tau/\sqrt{2\pi}$ deneme yapılmaktadır (τ = kuyruk kesimi). Kabul edilecek değere ulaşana kadar çok fazla reddetme işlemi gerçekleştiğinden çok maliyetli bir yöntemdir. Uygulamalarda verimsiz olduğu için çok fazla tercih edilmeyen bir yöntemdir. Lizard protokolünde S matrisi üretilirken bu yöntem kullanılmaktadır. Ek olarak S matrisini ayrık Gauss dağılımı ile veya rastgele üretmek sistemin güvenliği üzerinde bir etki yaratmamaktadır (Peikert, 2016).



Şekil 1. Reddetme Örneklemesi (Rejection Sampling Explanined, 2018)

2.4. Hata Atakları (Fault Attack)

Hata atakları aktif uygulama saldırıları olarak sınıflandırılmaktadır (Taha ve Eisenbarth, 2015). Bir hata atağında, saldırgan taraf kriptografik sistemde kasıtlı olarak bir hata oluşturur ve bu hatanın sistemde oluşturduğu sonucu inceler (Farhady Ghalaty, 2016). Hata ataklarının amacı; sistemdeki gizli anahtarı açığa çıkarmak ve sistemin güvenliğini azaltmaktır (Farhady Ghalaty, 2016). Genel olarak hata atakları ile ilgili birçok çalışma bulunmaktadır fakat kafes tabanlı sistemler üzerinde bu atakların çok fazla uygulaması bulunmamaktadır. Bilindiği kadarıyla kafes tabanlı sistemler üzerinde yapılan hata atakları NTRU (Hoffstein vd, 2010) sistemi üzerinde geçekleştirilmiştir. Kamal ve Youssef 2011 yılında NTRUEncrypt ve 2012 yılında NTRUSign protokolleri üzerinde hata atakları yöntemleri ile saldırı gerçekleştirmişlerdir.

2016 yılında Bindel ve arkadaşları yaptıkları bir çalışmada kafes tabanlı imzalama şemalarını inceleyerek, uygulamalar üzerinde gerçekleştirilebilecek üç tane hata atağı yöntemi önermişlerdir. Bu hatalar; rastgeleleştirme (randomization), atlama (skipping) ve sıfırlama (zeroing) hatalarıdır.

- Rastgeleleştirme Hatası: Bu yöntemde, uygulama içerisinde kullanılmakta olan bir değer rastgele olarak değiştirilir. Saldırgan değişkenin değerini bilmez fakat değişkenin hangi aralıkta olduğunu bilir ve bu açıktan yararlanır. Saldırgana bağlı olarak hata tüm değişkeni veya sadece bazı bytelar ile bitleri hedef alabilir (Bindel vd, 2016).
- Atlama Hatası: Program içerisinden seçilen kod satırlarının göz ardı edilmesi ile bu yöntem gerçekleştirilir (Bindel vd, 2016).
- Sıfırlama Hatası: Saldırgan program içerisindeki bir değerin tamamını veya bir kısmını sıfır değeri ile değiştirerek saldırıyı yapmaktadır (Bindel vd, 2016).

2.5. Lizard Protokolü Üzerinde Gerçekleştirilen Hata Atakları

Han ve arkadaşları 2018 yılında yaptıkları bir çalışmada atlama ve sıfırlama hataları ile Lizard[+] protokolünde kullanılan gizli değerleri elde edebildiklerini, rastgeleleştirme hatası ile herhangi bir bilgi elde edemediklerini söylemişlerdir. Bu bölümde Lizard protokolüne Han ve arkadaşları tarafından yapılan çalışmadaki bu ataklardan bahsedilmiştir. Sadece atlama ve sıfırlama hata ataklarından sonuç alınabildiği için bu ataklardan bahsedilmiştir. Rastgeleleştirme hata atağı ile herhangi bir sonuç alınamadığından bu kısımda bu ataktan bahsedilmemiştir.

Lizard protokolüne atlama hata atakları üç parça olarak uygulanmıştır. Bunlar; rastgele sayı üretimi, toplama işlemleri ve modüler indirgeme işlemleri üzerinde yapılmıştır. Yapılan atak whitebox atak modeli olarak varsayılmıştır. Bu yöntem saldırgan tarafın birkaç gizli bilgi dışında her şeyi bildiği ve kaynak koda erişimi olduğu bir saldırı modelidir. Lizard protokolünün yapısı gereği S, E matrisleri ve r vektörü bilinmemektedir. Ancak bu elemanlar dışında bütün açık parametreler bilinmektedir.

Rastgele sayı üretimi kısmında rastgele A matrisi oluşturulması, E hata matrisi oluşturulması ve S gizli matrisinin oluşturulması aşamasında atlama hatası uygulanmıştır.

```
1  void gen_A_CPA() {
2      for (int i = 0; i < LWE_M; ++i) {
3          uint16_t* pk_Ai = pk_CPA.A + LWE_N * i;
4          for (int j = 0; j < LWE_N; ++j) {
5              pk_Ai[j] = rand() << _16_LOG_Q;
6          }
7      }
8  }
```

Şekil 2. Lizard protokolü rastgele A matrisi üretimi

Lizard protokolünde A matrisinin üretimi Şekil 2 de görülen kod bloğu ile yapılmaktadır. Bu kod bloğu içerisinde matrisin üretildiği 5. satır atlandığında, C programlama dili A matrisini otomatik olarak 0 değerine eşitlediğinden düz metin şifre metin üzerinden elde edilebilir hale gelmektedir (Han vd, 2018). Bu durum şu şekilde açıklanabilir; A matrisi 0 olduğunda $c_1' \leftarrow A^T r$ işleminden dolayı c_1' değeri 0 olmaktadır ve $c_1 \leftarrow [(p/q) \cdot c_1']$ işlemi 0 sonucunu verdiğinden düz metin direk elde edilebilmektedir.

```
1  void gen_E_CPA() {
2      for (int i = 0; i < LWE_M; ++i) {
3          uint16_t* pk_Bi = pk_CPA.B + LWE_L * i;
4          for (int j = 0; j < LWE_L; ++j) {
5              pk_Bi[j] = SAMPLE_DG() << _16_LOG_Q;
6          }
7      }
8  }
```

Şekil 3. Lizard protokolü E hata matrisi üretimi

Lizard protokolünde E matrisinin üretimi Şekil 3 deki kod bloğu ile yapılmaktadır. Bu kod bloğu içerisinden 5. satır atlandığında Gauss elemesi yöntemi kullanılarak S matrisi kabul edilebilir bir zamanda elde edilebilmektedir. E matrisi üretimi atlandığında, matrisin değeri otomatik olarak 0 olduğundan LWE probleminden E matrisi çıkarılmış olmaktadır. $S = A^{-1}B$ işlemi yapılarak gizli anahtar kolayca elde edilmektedir. Bu gizli bilgi kullanılarak düz metin şifre metin üzerinden elde edilebilmektedir (Han vd, 2018).

[+] https://github.com/LizardOpenSource/Lizard_c

```
void gen_sk_CPA() {
    for (int i = 0; i < LWE_L; ++i) {
        uint16_t* sk_i = sk_CPA + LWE_N * i;
        for (int j = 0; j < LWE_N; ++j) {
            sk_i[j] = (rand() & 0x01) + (rand() & 0x01) - 1;
        }
    }
}
```

Şekil 4. Lizard protokolü gizli S matrisi (gizli anahtar) üretimi

Protokol içerisinde S matrisinin üretimi Şekil 4 de görülmektedir. Bu kod bloğu içerisinde 5. satır atlandığında, A matrisinin atlandığı zaman ortaya çıkan durumla aynı durum ortaya çıkmaktadır. $m' \leftarrow [t/p(c_2 - S^T c_1)]$ işleminden dolayı düz metin şifre metin üzerinden direk elde edilebilmektedir (Han vd, 2018).

Lizard protokolünde 3 kısımda toplama işlemi yapılmaktadır. Bunlar; B matrisinin üretimi, c_2 şifre metni üretimi ve m' düz metni üretimi aşamalarında yapılmaktadır. B matrisinin üretimi aşamasındaki toplama işlemi atlandığında m' düz metni direkt olarak elde edilebilmektedir (Han vd, 2018). Han ve arkadaşları şifreleme ve şifre çözme kısımlarındaki toplama işlemleri üzerinde atlama hatası kullanılarak herhangi bir bilgi elde edilemediğini söylemişlerdir (Han vd, 2018).

Yukarıdaki kısımlarda bahsedildiği üzere; rastgele sayı üretimi kısmında atlama yapıldığında, C programlama dili oluşacak değeri 0'a eşitlediğinden bu kısımda atlama hatası ile sıfırlama hatası aynı durumda olmaktadır. Yani sıfırlama hata ataklarından elde edilen sonuçlar atlama hata ataklarından elde edilen sonuçlar ile aynıdır.

3. Araştırma Sonuçları ve Tartışma

Bu kısımda Lizard protokolünün verimliliğinin, güvenliğinin artırılması için ve hata analizi saldırılarına karşı olan güvenliği artırmak için uygulama üzerinde yapılan değişikliklerden ve elde edilen sonuçlardan bahsedilmiştir.

3.1. Rastgele Sayı Üretimi İçin Yapılan Değişiklik

Lizard protokolünde yapılan ilk değişiklik rastgele sayı üretimi üzerinde yapılmıştır. Günümüzde kullanılmakta olan neredeyse bütün kriptosistemlerin güvenliğinin temelinde yüksek kalitede rastgele sayı üretimi işlemi bulunmaktadır (Hughes ve Nordholt, 2016). Bu rastgele sayılar uzun vadeli anahtar oluşturma, geçici anahtar oluşturma ve yan kanal saldırılarını önleme amacıyla kullanılmaktadır. Gerçek rastgele veriler entropi denilen belirsizliği ölçülebilir bir özelliğe sahiptir. Eğer kriptosistemlerde kullanılmakta olan rastgele sayılar gerçekten rastgele sayı değilse (entropisi düşükse), kriptosistemin güvenliği büyük ölçüde tehlikeye girmektedir.

Lizard protokolünde rastgele sayı üretimi için C programlama dilinin rastgele sayı üretici fonksiyonu olan rand() fonksiyonu kullanılmıştır. Fakat bu fonksiyon belirli sayıda çalıştırmadan sonra sürekli aynı çıktıyı üretmektedir. Bunun sebebi bu fonksiyonda kullanılan giriş verisinin (tohum, seed) sistem saati olmasıdır ve bu girdinin entropisinin düşük olmasıdır. Rastgele sayı üretirken belirli bir sayıya göre üretim yapmak güvenliği artırmaktadır. Fakat kullanılan bu sayı her saniye değişiyor olsa bile vereceği çıktı tahmin edilebilir olmaktadır (Eastlake vd, 2005). Bu durumdan dolayı rand() fonksiyonu kriptografik sistemlerde kullanılması önerilmemekte olan bir fonksiyondur.

Han ve arkadaşları yaptıkları çalışmada gerçek rastgelelik için kernelde bulunan dev/urandom fonksiyonunun kullanılmasını önermişlerdir. dev/urandom fonksiyonunun herhangi bir limiti yoktur ve bundan dolayı istenilen kadar byte çıktı vermektedir. Bu durum entropi havuzunun tekrar yenilenmesi için yeterli zamanı vermeden daha fazla byte gerektirdiğinden oluşacak rastgele sayılar kriptografik açıdan güçlü olacaktır. Ancak bu güvenlik seviyesi kriptografik sistemler için yeterli olmayabilmektedir.

Rastgele sayılar istendiğinde entropi havuzu içerisinden SHA (Secure Hash Algorithm) özeti alınarak elde edilebilmektedir. SHA özet değeri kriptografik açıdan güçlü olduğundan entropi havuzunun içeriğini göstermemektedir ve fonksiyona verilen girdiyi elde etmek için SHA çıktısını tersine çevirmek gerekmektedir. Bu durum hesaplama açısından mümkün olmayan bir durumdur. Bundan dolayı rastgele sayı üretimi için özet fonksiyon kullanılması, kriptografik sistemler açısından güvenliği büyük ölçüde artırmaktadır.

Bu incelemeler doğrultusunda protokol içerisindeki rand() fonksiyonları cSHAKE-128 fonksiyonu ile değiştirilmiştir. Ek olarak ayrık Gauss dağılımı ile örnek üretme aşamasında kullanılan rand() fonksiyonları uygulamadan çıkarılmıştır. Bu yapılanlar doğrultusunda, uygulama rastgele sayı üretimi açısından öncesine oranla çok daha güvenli hale gelmiştir.

Rastgele sayı üretimi aşamasındaki yapılan değişiklikler sonrasında, A matrisinin üretimi kısmında yapılan atlama hatası uygulanamaz duruma gelmiştir. Çünkü uygulamada A matrisinin üretimi atlandığı zaman derleme aşamasında program hata üretmektedir. Bu sayede atlama ataklarına kısmen de olsa önlem alınabilmiştir. S ve E matrisleri üzerinde yapılan atlama ataklarına karşı bir önlem alınamamıştır.

```
void gen_A_CPA() {
    unsigned char seed_A[32] = { 0,1,2,3,4,5,6,7,0,1,2,3,4,5,6,7,
    0,1,2,3,4,5,6,7,0,1,2,3,4,5,6,7 };
    uint16_t* pk_Ai = pk_CPA.A;
    cshake128_simple((uint8_t*)pk_Ai, 2 * LWE_M*LWE_N, 0, seed_A, 32);
    for (int i = 0; i < LWE_M*LWE_N; ++i) {
        pk_Ai[i] = pk_Ai[i] << _16_LOG_Q;
    }
```

Şekil 5. Değiştirilmiş olan A matrisi üretimi kodları

3.2. Gizli Anahtar Üretim Fonksiyonunda Yapılan Değişiklik

Lizard protokolünde S matrisinin (gizli anahtar) üretimi Şekil 4 de gösterilen gen_sk_CPA() fonksiyonu ile yapılmaktadır. Bu üretim işleminde reddetme örneklemesi yöntemi kullanılmaktadır. S matrisi üretilirken; değeri 0 veya 1 olan rastgele iki değer üretilmektedir. Devamında bu değerler toplanıp 1 çıkarılmaktadır. Bu şekilde bir üretim yapıldığında -1 ve 1 değerleri %25 ihtimalle, 0 değeri ise %50 ihtimalle seçilmektedir. Han ve arkadaşları yaptıkları çalışmada bu durumun tutarsız bir durum olduğunu ve saldırganların gizli anahtarı kolayca tahmin edebileceklerini söylemişlerdir (Han vd, 2018). Bu durumdan dolayı -1, 0, 1 değerlerinden oluşmakta olan S matrisinde, bütün değerlerin eşit şekilde seçilmesi (reddedilmesi) gerektiğini söylemişlerdir (Han vd, 2018). Bu öneri doğrultusunda Lizard protokolünün uygulamasında S matrisinin üretilmesi aşamasında kullanılan reddetme örneklemesi yöntemi; -1, 0 ve 1 değerleri eşit (%33) ihtimaller ile seçilecek, diğer olası durumları reddedecek şekilde yeniden düzenlenmiştir. Bu sayede gizli anahtar üretimi kısmına gelebilecek ataklara önlem alınması amaçlanmıştır.

```
void gen_sk_CPA() {
    int count = 0;
    int count2 = 0;
    unsigned char seed_A[32] = { 0,1,2,3,4,5,6,7,0,1,2,3,4,5,6,7,
    0,1,2,3,4,5,6,7,0,1,2,3,4,5,6,7 };
    uint16_t* sk_i = sk_CPA;
    cshake128_simple((uint8_t*)sk_i, LWE_N*LWE_L * 2, 0, seed_A, 32);
    SAMPLE_DG(sk_i, LWE_N*LWE_L);
    for (int i = 0; i < LWE_N*LWE_L; i++) {
        count = 0;
        if ((sk_i[i] & 0x03) != 1) {
            sk_i[i] = (sk_i[i] & 0x01) + ((sk_i[i] >> 1) & 0x01) - 1;
        }
        else if ((sk_i[i] & 0x03) == 1) {
            if (count < 8) {
                sk_i[i] = (sk_i[i] >> 2);
                count++;
            }
            else {
                count = 0;
                cshake128_simple(sk_i + i, 2, count2++, seed_A, 32);
            }
        }
    }
}
```

Şekil 6. Değiştirilmiş olan S matrisi üretimi kodları

3.3. Yapılan Değişikliklerden Sonra Protokoldeki Değişimler

Protokolün uygulamasında rastgele sayı üretimi fonksiyonları ve S matrisi üretim fonksiyonu daha verimli ve güvenli hale getirildikten sonra, çalışma süresi açısından uygulama önceki hali ile karşılaştırılmıştır. Protokolün uygulaması üzerinde yapılan değişikliklerin birçoğu anahtar üretimi kısmında yapılmıştır. Bu durumdan dolayı sadece anahtar üretimi kısmı çalışma süresi açısından incelenmiştir. Yapılan değişikliklerden sonra protokolün uygulamasındaki değişiklikler Tablo 1'de verilmiştir.

Sonuçlar elde edilirken Intel Core i7-4700HQ 2.40GHz işlemci ve 16 GB RAM'a sahip bir bilgisayar ve Windows işletim sistemi üzerinde Ubuntu 16.04 LTS işletim sistemi (kabuk) kullanılmıştır. Kodların derlenmesi ve çalıştırılması için gcc 5.4.0 derleyicisi, ayrıca derleyici tarafından sağlanan en iyi optimizasyon seçeneği kullanılmıştır (-03). Bütün kodlamalar C programlama dilinde yapılmıştır. Lizard protokolünün *Recommended* parametre seti üzerinde bu çalışma yapılmıştır.

Tablo 1. Yapılan değişikliklerden sonra Lizard protokolü anahtar üretimi kısmındaki değişiklikler

	Uygulamanın Orijinal Anahtar Üretimi	Uygulamanın Değiştirilmiş Anahtar Üretimi
Çalışma Süresi (ms)	18 - 19	12 - 14
Cycle	4 – 4.5 milyon	3 – 3.5 milyon

4. Sonuç

Bu çalışmada Lizard protokolünün uygulaması verimlilik ve güvenlik açısından incelenmiştir. Uygulamalar üzerindeki hata ataklarının incelemesi yapıldıktan sonra, Lizard protokolü üzerine yapılan hata ataklarının uygulamaları incelenmiştir. (Bindel vd, 2016) ve (Han vd, 2018) çalışmalarından yola çıkılarak Lizard protokolünün hata ataklarına veya gelebilecek farklı saldırılara karşı güçlendirilmesi ve daha verimli çalışması için uygulama üzerinde değişiklikler yapılmıştır. Bu doğrultuda gelecek çalışma olarak, NIST'e önerilmiş olan kafes tabanlı şifreleme sistemleri üzerinde bu gibi incelemeler yapılması ve güvenliğinin yetersiz olduğu düşünülen protokollerin uygulamalarının iyileştirilmesi amaçlanmaktadır.

Teşekkür

Bu çalışma EEEAG – 116E279 numaralı proje kapsamında TÜBİTAK tarafından desteklenmiştir.

Kaynakça

Alwen, J., Krenn, S., Pietrzak, K., & Wichs, D. (2013, August). Learning with rounding, revisited. In Annual Cryptology Conference (pp. 57-74). Springer, Berlin, Heidelberg.

Banerjee, A., Peikert, C., & Rosen, A. (2012, April). Pseudorandom functions and lattices. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 719-737). Springer, Berlin, Heidelberg.

Bernstein, D. J. (2009). Introduction to post-quantum cryptography. In *Post-quantum cryptography* (pp. 1-14). Springer, Berlin, Heidelberg.

Bindel, N., Buchmann, J., & Krämer, J. (2016, August). Lattice-based signature schemes and their sensitivity to fault attacks. In 2016 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC) (pp. 63-77). IEEE.

Chen, L., Jordan, S., Liu, Y. K., Moody, D., Peralta, R., Perlner, R., & Smith-Tone, D. (2016). Report on post-quantum cryptography," National Institute of Standards and Technology. US Department of Commerce.

Cheon, J. H., Kim, D., Lee, J., & Song, Y. (2018, September). Lizard: Cut off the tail! A practical post-quantum public-key encryption from LWE and LWR. In International Conference on Security and Cryptography for Networks (pp. 160-177). Springer, Cham.

Computer Security Division, I. T. L. (2017, February 3). Round 1 Submissions—Post-Quantum Cryptography | CSRC. CSRC | NIST. https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions

Ducas, L., Durmus, A., Lepoint, T., & Lyubashevsky, V. (2013, August). Lattice signatures and bimodal Gaussians. In Annual Cryptology Conference (pp. 40-56). Springer, Berlin, Heidelberg.

Eastlake, D., Schiller, J., & Crocker, S. (2005). Randomness requirements for security. RFC4086.

Farhady Ghalaty, N. (2016). Fault Attacks on Cryptosystems: Novel Threat Models, Countermeasures and Evaluation Metrics (Doctoral dissertation, Virginia Tech).

Gentry, C., Peikert, C., & Vaikuntanathan, V. (2008, May). Trapdoors for hard lattices and new cryptographic constructions. In Proceedings of the fortieth annual ACM symposium on Theory of computing (pp. 197-206).

Han, S., Choi, N., An, H., Choi, R., & Kim, K. (2018, January). Prey on Lizard: Mining Secret Key on Lattice-based Cryptosystem. In 2018 Symposium on Cryptography and Information Security (SCIS 2018). IEICE Technical Committee on Information Security.

Hoffstein, J., Howgrave-Graham, N., Pipher, J., & Whyte, W. (2009). Practical lattice-based cryptography: NTRUEncrypt and NTRUSign. In The LLL Algorithm (pp. 349-390). Springer, Berlin, Heidelberg.

Howe, J., Khalid, A., Rafferty, C., Regazzoni, F., & O'Neill, M. (2016). On practical discrete Gaussian samplers for lattice-based cryptography. IEEE Transactions on Computers, 67(3), 322-334.

Hughes, R., & Nordholt, J. (2016). Strengthening the security foundation of cryptography with Whitewood's quantum-powered entropy engine.

Kamal, A. A., & Youssef, A. (2011). Fault analysis of the NTRUEncrypt cryptosystem. IEICE transactions on fundamentals of electronics, communications and computer sciences, 94(4), 1156-1158.

Kamal, A. A., & Youssef, A. M. (2012, August). A scan-based side channel attack on the NTRUEncrypt cryptosystem. In 2012 Seventh International Conference on Availability, Reliability and Security (pp. 402-409). IEEE.

Le Trieu Phong, T. H., Aono, Y., & Moriai, S. LOTUS: Algorithm Specifications and Supporting Documentation.

Lindner, R., & Peikert, C. (2011, February). Better key sizes (and attacks) for LWE-based encryption. In Cryptographers' Track at the RSA Conference (pp. 319-339). Springer, Berlin, Heidelberg.

Naehrig, M., Alkim, E., Bos, J. W., Ducas, L., Easterbrook, K., LaMacchia, B., ... & Raghunathan, A. (2017). FrodoKEM: practical quantum-secure key encapsulation from generic lattices. NIST submissions.

Peikert, C. (2016). A decade of lattice cryptography. Foundations and Trends® in Theoretical Computer Science, 10(4), 283-424.

Regev, O. (2005). On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing, 84–93. https://doi.org/10.1145/1060590.1060603

Sampling: Rejection Sampling Explained. Erişim tarihi: 11 Şubat 2020, gönderen https://relguzman.blogspot.com/2018/04/rejection-sampling-explained.html

Seo, M., Park, J. H., Lee, D. H., Kim, S., & Lee, S. J. (2017). Emblem and r. EMBLEM. Technical Report. National Institute of Standards and Technology.

Shor, P. W. (1994, November). Algorithms for quantum computation: discrete logarithms and factoring. In Proceedings 35th annual symposium on foundations of computer science (pp. 124-134). Ieee.

Taha, M., & Eisenbarth, T. (2015). Implementation Attacks on Post-Quantum Cryptographic Schemes. IACR Cryptology ePrint Archive, 2015, 1083.