Pamukkale Univ Muh Bilim Derg, 25(5), 586-595, 2019

Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi

Pamukkale University Journal of Engineering Sciences

İlişkisel veri tabanlarında XML veriler için Homoglif dönüşüm tabanlı damgalama

Homoglyph transformation based watermarking for XML data in relational databases

Mustafa Bilgehan İMAMOĞLU[1*], *Mustafa ULUTAŞ*[2]

[1,2]Bilgisayar Mühendisliği Bölümü, Mühendislik Fakültesi, Karadeniz Teknik Üniversitesi, Trabzon, Türkiye.
bilgehan@ktu.edu.tr, ulutas@ktu.edu.tr

Geliş Tarihi/Received: 26.06.2018, Kabul Tarihi/Accepted: 02.10.2018 doi: 10.5505/pajes.2018.65288
* Yazışılan yazar/Corresponding author Araştırma Makalesi/Research Article

Öz

Bilgi Teknolojilerindeki hızlı gelişim son zamanlarda İnternet üzerinden veri paylaşımını kolaylaştırmıştır. Dijital verilerin izinsiz olarak kopyalanması veya değiştirilmesi telif hakkı koruma sorununu beraberinde getirmiştir. Kriptografi hassas bilgileri güvenli bir şekilde paylaşmak için kullanılsa bile şifrelenmiş bilgiler orijinal veri ile ilişkilendirilemediğinden telif hakkı koruma sürecinde kullanılamazlar. Dijital haklar yönetiminde son zamanlarda sayısal damgalama yöntemleri kullanılmaktadır. Özel hazırlanmış gizli bilgileri veya damgayı sayısal ortama gizleme yöntemleri verilerin korunmasında ve sahiplik bilgilerinin belirlenmesinde ek güvenlik mekanizması sağlamaktadırlar. Çoklu ortam dosyalarındaki hak yönetimi için önerilen algoritmalar, veri tabanları üzerinde doğrudan uygulanamamaktadır. Veri tabanı damgalama için kullanılan yöntemlerden bazıları, ilişkide yer alan XML veri alanlarından faydalanmaktadır. Bu çalışmada XML alanlar üzerinde Homoglif Dönüşüm tabanlı yeni bir algoritma önerilmektedir. Birincil anahtar ve özüt fonksiyonu veri tabanında damgalanacak satırların seçiminde kullanılmıştır. Sonuçlar, önerilen yöntemin literatürdeki diğer yöntemlerle karşılaştırıldığında yüksek damgalama kapasitesinde çeşitli ataklara karşı iyileştirilmiş sağlamlık elde ettiğini göstermektedir.

Anahtar kelimeler: Sayısal damgalama, Veri tabanı damgalama, Homoglif, XML

Abstract

Rapid developments in Information Technologies have eased data sharing through the Internet recently. Copying or modification of digital data without permission and sharing by unauthorized users have brought copyright protection problem. Even though cryptography can be used to share sensitive information securely, it cannot be used to protect copyright since encrypted information cannot be associated with the original. Digital watermarking methods are used lately in digital rights management. Methods to hide specially crafted secret information or watermark into digital media provide additional security mechanism in protecting the data and determining the ownership information. Algorithms recommended for right management in multimedia files cannot be applied directly on databases. Some methods used for watermarking databases take advantage of Extensible Markup Language (XML) data fields attached. A new method based on Homoglyph Transformation in XML fields is proposed in this manuscript. Primary key and hash algorithm are used together in the selection of lines to be watermarked in the database. Results indicate improved robustness against various attacks in high watermarking capacity by the proposed method compared to other methods reported in the literature.

Keywords: Digital watermarking, Database watermarking, Homoglyph, XML

1 Giriş

Dijital damgalama, veriye özel olarak hazırlanan damganın taşıyıcı sinyale (resim, metin, ses) sinyal kalitesini bozmadan eklenmesini sağlayan tekniktir. İlk çalışmalarda dijital damgalama multimedya nesneleri üzerinde telif haklarının korunması, sahiplik kanıtı ve bütünlük kontrolü için kullanılmaktaydı. Sonrasında çalışma alanları ilişkisel veri tabanlarının telif haklarının korunması, saldırı algılama ve veri bütünlüğünün kanıtı için kullanımı şeklinde genişletilmiştir.

Veri tabanı damgalama şemaları, amaçlarına bağlı olarak sağlam veya kırılgan şemalar olarak kategorize edilebilirler. Sağlam damgalama şemaları [1]-[12] telif hakları korumayı sağlarken, kırılgan damgalama şemaları müdahale tespiti ve bütünlüğün kanıtı için kullanılırlar.

Sağlam damgalama şemalarında, veri tabanının kime ait olduğunu (kişi, kurum, organizasyon) kanıtlayabilmek için damga bilgisi çoğunlukla veri sahibinin bilgilerini taşımaktadır. Telif hakkı koruması sağlayan sağlam şemaların birçoğu [1]-[12] veri bütünlüğünü ve kullanılabilirliğini etkileyen orijinal içeriğin bozulmasına sebep olmaktadır. Bu şemalar damgalarını yerleştirmek için ilişkisel veri tabanlarının numerik [1]-[6] veya kategorik alanlarını [7],[12] kullanmaktadırlar. Kullanılan damga bilgisi anlamsız bir şekilde oluşmuş bit dizisi olabileceği gibi resim, konuşma veya sahip bilgisi gibi anlamlı bit dizisi de olabilir.

Sağlam damgalama şemalarının, damganın tamamen yok edilmesi veya algılanamaz hale getirilmesi amaçlı yapılan saldırılara dayanıklı olması gerekir. Bu sebeple algoritma tasarımının amacı damgalanmış kayıtlara yapılan silme, ekleme, değiştirme gibi saldırılara karşı şemanın dayanıklı olmasıdır. Sağlam damgalama şemalarının temel varsayımı, ilişkisel veri tabanının orijinal verileri üzerinde gerçekleştirilecek küçük değişikliklerin müsamaha edilebilir olduğudur. Bununla birlikte, tıbbi veriler gibi hassas veri tabanlarındaki değişiklikler, her ne kadar küçük de olsa, veri tabanının kullanılabilirliğini etkilemekte, hatta veri tabanının bütünlüğünün ihlaline sebep olabilmektedir. Bu sebeple, sağlam damgalama şemaları, hassas veri içeren ilişkisel veri tabanları için uygun değildir.

Kırılgan damgalama alanında yapılan çalışmaların büyük bir kısmı görüntü, ses ve videolar üzerinde yapılmıştır. Yakın

geçmişte diğer veri alanlarının önem kazanması ile birlikte metinler ve ilişkisel veri tabanları üzerinde de kırılgan damgalama şemaları kullanılmaya başlanmıştır.

2004 yılında Li ve diğ. [13] gerçekleştirmiş oldukları kırılgan damgalama şemasında, veri tabanını gruplara ayırarak damganın her gruba bağımsız olarak gömülmesini gerçekleştirmiştir. Doğrulama sürecinde, her bir grubun kendi içinde bağımsız olarak damgayı doğrulamasına çalışmıştır. Bu yöntem orijinal veri üzerinde tahribata sebep olmamakla birlikte sadece kategorik veriler üzerinde uygulanabilir olması, uygulama alanını kısıtlı kılmaktadır.

2009 yılında Kamel [14]'in önermiş olduğu kırılgan damgalama şeması da Li ve diğ. [13] şemasına benzer şekilde veri tabanı ilişkilerini gruplara ayırarak her bir gurubun bağımsız olarak damgalanmasını gerçekleştirmektedir.

2010 yılında Khataeimaragheh ve Rashidi [15] tarafından önerilen çalışma da Guo ve diğ. çalışmasında olduğu gibi nümerik alanların en anlamsız 2 bitinde değişikliğe giderek kırılgan damgalama yöntemi ile veri tabanı bütünlüğünü sağlamaya çalışmaktadır.

2011 yılında Hamadou ve diğ. [16] önermiş olduğu kırılgan damgalama şeması, veri tabanı alanlarının özüt değerleri dikkate alınarak yeniden sıralanması üzerine kuruludur. Bu sayede orijinal veri tabanı herhangi bir tahribata uğramamaktadır. Veri tabanı kaydının her bir alanının en anlamsız bitleri kullanılarak damga verisi oluşturulur. Oluşturulan damga verisi sertifika yetkilisine kayıt ettirilir. Şema alan adlarına göre sanal bir sıralamaya dayandığından, saldırgan tarafından alan adlarının değiştirilmesi sonrasında saldırının belirlenmesi mümkün olmayacaktır.

XML farklı türdeki verileri ortak bir dil ile tanımlamayı sağlamaktadır. Ağ teknolojilerinin gelişmesiyle, daha fazla veri XML formatında tanımlanmakta ve yer değiştirmektedir. Günümüzde verinin farklı platformlar arasında paylaşımı için XML sıklıkla kullanılmaktadır. Metin olarak saklandıkları için yasadışı izinsiz kopyalanma tehdidi altındadır. Bu sebeple XML verilerinin telif hakkı problemini çözmek önem arz etmektedir.

XML'in sahip olduğu iki özellik sebebiyle Web'in temelini oluşturduğu söylenebilir. Bu özellikler, hiyerarşik veriyi düz metin kullanarak kodlayabilmesi ve kullandığı ayrıntılı etiketler sayesinde özel bir okuyucu veya tercümana ihtiyaç duyulmadan belgenin anlaşılabilir oluşudur. XML bilgi alışverişinde, web servislerinde, şirket-tüketici veya şirketler arasındaki iletişimde ve hatta medikal uygulamalarda sıklıkla kullanılmaktadır.

Literatürde yapılmış örnek çalışmalar ve XML alanların özellikleri dikkate alındığında, bu çalışmada XML alanlar üzerinde uygulanabilir, bozulmaya sebep olmayan, etiketlerin içerdiği verinin türüne bakmaksızın damga yerleştirebilen, damga verisini sertifikalandırıp dış veri olarak farklı bir konumda saklanmasına gerek kalmayan, verinin boyutunu değiştirmeyen, damga çıkarım sonrası orijinal veriye geri dönüşüm sağlayan, algılanması güç yeni bir damgalama şeması önermekteyiz.

Önerilen yöntem veri tabanlarında XML veri alanlarına damga yerleştirme sürecinde damganın fark edilebilirliğini azaltmak, damgalama kapasitesini artırmak ve veri türünden bağımsız olarak damgalama işlemini gerçekleştirmek amacıyla [17] çalışmasında da kullanılan Homoglif Dönüşüm yönteminden faydalanmaktadır. Literatürdeki çalışmaların birçoğu XML verilerin içeriğinde yer alan metin türlerine damgalama yapabilmektedir. Metinler ile beraber nümerik veriler üzerine de damga yerleştiren az sayıda çalışma ise nümerik verinin bozulmasına ve damga çıkarım sonrası orijinal verinin elde edilememesine sebep olmaktadır. Önerilen yöntem XML alanın içerdiği nümerik ve metin verileri ayırt etmeksizin orijinal veriyi bozmadan damga yerleştirebilmektedir. Yöntemin geri dönüşüm desteği sayesinde, damgalanmış XML verisi içerisinden, kullanıcının belirlediği özel anahtar dışında hiçbir ek bilgiye gerek kalmadan yerleştirilmiş damga verisi ve XML verinin orijinal hali başarıyla elde edilebilmektedir. Literatürdeki çalışmalar XML alanların metin verileri üzerinde damga gerçekleştirirken metnin içerdiği kelimeleri sinonim havuzundan seçilen karşılıkları ile değiştirmekte, karakterleri büyültüp/küçülmekte veya boşluk eklemektedir. Bu işlemler damga verisinin saldırganlar tarafından belirlenebilmesini kolaylaştırmaktadır. Önerilen yöntem metin veriler üzerinde anlamsal bozulmaya sebep olmadığı için saldırganlar tarafından fark edilebilirliği düşüktür. Bu özelliği sayesinde sosyal ağ, forum, mesajlaşma gibi genel kullanıma açık veri tabanlarının yanı sıra tıbbi, askeri ve güvenlik alanındaki veri tabanları gibi hassas veriler üzerinde de uygulanabilirdir.

Yayının ikinci bölümünde bu alanda gerçekleştirilen çalışmalardan bahsedilecektir. Önerilen yöntemin detaylarına üçüncü bölümde değinilirken, elde edilen sonuçlar ve literatürdeki benzer çalışmalar ile gerçekleştirilen kıyaslamalar dördüncü bölümde yer almaktadır. Son olarak ise genel bir değerlendirme yapılacaktır.

2 İlgili çalışmalar

Sayısal damgalama 1990 yılından beri üzerinde çalışılan bir konu olmasına rağmen, ilişkisel veri tabanları üzerinde damgalama çalışmaları 2003 yılında ilk kez Agrawal ve diğ. tarafından yapılmıştır [1]. XML dokümanlara özel veri saklama yaklaşımı ise 2001 senesinde ilk kez Inoue ve diğ. tarafından ele alınmıştır [20]. XML etiketlerinin açıklık/kapalılık durumu, düğüm kapama etiketine fazladan boşluk ekleme, alan değerlerinin sıralanması gibi yöntemler ile verinin XML doküman içerisine saklanması sağlanmıştır. Fakat çalışma telif hakkı için damgalamadan daha çok veri saklama amacı gütmektedir. Ayrıca XML veriye eklenen boşluklar sebebiyle verinin büyümesine de sebep olmuştur.

2005 yılında Wilfred ve diğ. XML veri alanlarına veri türünü dikkate alarak damga yerleştirme çalışması yapmışlardır [21]. Damgalama yapılacak veri nümerik olması durumunda Agrawal'ın yöntemi ile LSB üzerine veri saklama gerçekleştirilirken, metin alana veri saklarken WordNet [22] sinonim havuzunu kullanmışlardır. Çalışmanın sakladığı veri miktarının düşüklüğünün yanı sıra, gömülen damganın çıkarılabilmesi için damgalama aşamasında kullanılan parametrelere ihtiyaç duyulması sebebiyle bu parametrelerin harici bir kaynakta saklanması zorunluluğu bulunmaktadır. Metinlere damga yerleştirme aşamasında seçilen sinonim karşılığın farklı uzunlukta olmasından dolayı da veri boyutunun değişimi söz konusudur.

2006 yılında Gu ve diğ. XML verilere parmak izi eklemeyi sağlayan çalışmaları ile, metin ve nümerik verilerin farklı türdeki parmak izi verisi ile damgalanmasını sağlamışlardır [19]. Çalışmalarında damga verisi ikilik düzene dönüştürülerek, birincil anahtar ve özüt fonksiyonu yardımı ile damgalanacak satırlar seçilmektedir. Veri türünün nümerik olması durumunda LSB değişimi, metin olması durumunda ise sinonim kullanarak damgalama gerçekleştirilmiştir. Damganın geri elde edilmesi için damga verisinin uzunluğunun saklanma

gereksinimi, damgalama öncesi orijinal verinin elde edilememesi, orijinal veri üzerinde bozulmaya sebep olması algoritmanın zayıf yönleridir.

XML verinin bütünlüğünün sağlanması amacıyla 2006 yılında Yao ve diğ. yapmış olduğu çalışmada nümerik alanlar dikkate alınmadan, XML etiketleri ULC (Upper-Lower Coding) yöntemi ile damgalanmıştır [26]. Yöntem veri boyutunu değiştirmeden XML verisinin bütünlüğünü sağlamayı amaçlamaktadır. Etiketler üzerinde yapmış olduğu değişiklikten dolayı damga verisi saldırgan tarafından kolaylıkla belirlenebilmektedir.

2010 yılında Saad tarafından yapılan çalışmada Guo ve diğ. çalışmasına benzer bir şekilde parmak izi verisi ile damgalama gerçekleştirilmiştir [24]. Saad'ın çalışmasında gruplama kullanımı ve parmak izinin oluşturulma aşamasında geliştirmeler bulunmaktadır. Verinin bozulmasını engellemek için nümerik alanlar üzerinde değil, metin alanlar üzerinde damgalama yapılmıştır. Metinlerin büyültülüp/küçültülmesi yöntemi ile 1/0 verisinin yerleştirilmesi sağlanmıştır.

2012 yılında Saad, 2010 yılında önermiş olduğu algoritmayı geliştirerek damgalama aşamasında veri metninin karakterlerinin büyültülmesi yerine XML verisinin kapatma etiketinin sonuna "boşluk" eklemeyi tercih etmiştir [25]. Damgalama sonrası veri boyutunun değişimi ve damganın algılanabilir olma problemi bu çalışmada da söz konusudur.

2012 yılında Tchokpon ve diğ. yapmış oldukları çalışmada rastgele üretilen damga bitlerini veri içerisinde gömmek için "boşluk ekleme" yöntemini kullanmışlardır [27]. Veri üzerinde gerçekleştirilen sorgular dikkate alınarak en önemli etiket belirlenmeye çalışılmaktadır. Belirlenen bu etiketin içerdiği veriye damga bit değeri "0" olduğu durumda boşluk eklenmiş, "1" olduğu durumda veri değiştirilmemiştir.

2016 yılında Wen ve diğ. XML dokümanların damgalanmasını sağlayan iki farklı yaklaşım sunmuşlardır [23]. İlk yaklaşımda XML verinin görünüm şeklini veya formatını belirleyen XSLT verisi kullanılarak damgalama gerçekleştirilmiştir. İkinci yaklaşımda ise XML verinin işlevsel bağımlılığı dikkate alınmıştır. XSLT verisi olmayan XML veri tabanlarında yaklaşımın uygulanabilirliğinin olmaması, damgalama sonrası oluşturulan dosyanın harici bir ortamda saklanma gereksinimi damgalama algoritmasının zayıf yönlerini oluşturmaktadır.

3 Önerilen yöntem

Önerilen damgalama yöntemi XML veri alanlarının etiket bilgisini, veri içeriğini ve boyutunu değiştirmeksizin, veri üzerinde Homoglif Dönüşüm yöntemini kullanarak damgalama gerçekleştirmektedir. Bu yöntemde, karakterler kendilerine benzer görünümlü, farklı Unicode karakterler ile değiştirilmektedir. Önerilen algoritma, orijinal karakteri değiştirmesi durumunda "1" sayısal verisini, aynı bırakması durumunda ise "0" sayısal verisini damga olarak yerleştirmektedir. Algoritmanın bu özelliği sayesinde, damga verisinin tamamı veya büyük bir kısmı algılanamaz durumdadır. Orijinal verinin boyutu değişmemekle birlikte, verinin içeriğinde veya anlamsal bütünlüğünde de herhangi bir değişiklik yapılmamaktadır. Tüm bu özellikler sayesinde verinin bütünlüğü bozulmadan yüksek miktarda damga verisi gömülebilmektedir.

2017 yılı Haziran ayında duyurulan Unicode 10.0 versiyonu ile birlikte 1.200.000'e yaklaşan Unicode karakterler arasında birbiri ile benzerliği yüksek olan karakterler bulunmaktadır. Unicode Konsorsiyum görünüm olarak benzer karakterlerin listesini belirli aralıklarla yayınlamaktadır [18]. Bu karakterler arasında orijinal karaktere benzerliği sayesinde gözle ayırt edilemeyecekler olduğu gibi genel görünüm itibariyle benzer, fakat damgalama algoritmamızda kullanamayacağımız şekilde fark edilebilir olanlar da yer almaktadır. Değiştirilen Unicode karakterin fark edilebilirlik durumu, metnin sunulduğu ara yüzde kullanılan font ve karakter kodlama bilgisi ile de ilişkilidir. Bazı font ailelerinde birbirine benzediği düşünülen ve gözle bakıldığında farklı oldukları belirlenemeyen 2 farklı Unicode karakter, başka bir font ailesinde gözle algılanabilir olabilmektedir. Homoglif dönüşümde kullanılacak karakter ile damga verisi yerleştirildiğinde yerine kullanılacak olan Unicode karakterler seçilirken yapılan testlerde, web ortamında sıklıkla kullanılan Unicode ve Western kodlama türleri ile farklı font aileleri kullanılmıştır. Testler sonucunda, insan gözü ile fark edilebilir karakter dönüşümleri kullanım dışı bırakılmıştır. Şekil 1'de homoglif dönüşüm için kullanılan karakterler ve homoglif dönüşüm sonrası yerine kullanılacak karakterlerin listesi verilmiştir.

Orjinal Karakterler	!$%2345679ABCDEFGHJKLMNPQRSTUVXYZabcdeghijoprsuvxyz
Homoglif Karakterler	! $ % 2 3456 7 9 ABCDEFGHJKLMNPQRSTUVXYZabcdegнijoprsuvxyz

Şekil 1: Önerilen algoritmanın Homoglif Dönüşümde kullandığı karakterler ve Unicode karşılıkları

Önerilen yöntemde damga verisi ikilik tabana dönüştürülmektedir. Bu sayede damga verisinin türü algoritma açısından önemli değildir. Damga verisinin bit değerine göre homoglif dönüşümde karakter değişimi gerçekleştirilmektedir. Sadece damga bit değerinin "1" olduğu durumlarda homoglif karakter dönüşümü yapılmaktadır. Örneğin damga verisi olarak "0" değeri yerleştirilmek istendiğinde, veri içerisinde Şekil 1'de listelenen karakterlerden ilk karşılaşılan karakter değiştirilmeden bırakılacak ve "0" değerinin damgalanması sağlanacaktır. Damga verisi olarak "1" değeri yerleştirilmek istendiğinde ise, Şekil 1'de listelenen karakterlerden ilk karşılaşılan karakter homoglif eşi ile değiştirilerek "1" damga verisinin metne gömülmesi sağlanacaktır. Şekil 2'de "Bilgehan" örnek metni ve metnin içerisine "101001" damga verisinin gömülmesi sonucu metnin damgalanmış hali görülmektedir. "101001" damga verisinin ilk bit değeri "1" olduğundan, metnin ilk karakterinden başlamak üzere Şekil 1'de verilen karakterler aranır. Bulunan ilk karakter "B" karakteridir ve homoglif karşılığı ile değiştirilir. Damga verisinin ikinci bit değeri "0" olduğundan "i" karakteri olduğu gibi bırakılır. "l" karakteri Şekil 1'de verilen dönüşümde kullanılacak karakterlerden biri olmadığı için olduğu gibi bırakılır. Damga verisinin üçüncü bit değeri "1" olduğu için "g" karakteri homoglif karşılığı ile değiştirilir. Benzer şekilde "e" ve "h" karakterleri olduğu gibi bırakılırken "a" karakteri homoglif karşılığı ile değiştirilerek "1" değerindeki damga biti gömülür. 6 bitlik verinin gömülmesi sonucu metnin sadece 3 karakteri homoglif dönüşüme uğramıştır. Damga verisi içerisindeki "1" bitlerinin sayısı kadar homoglif dönüşümün gerçekleştirilecek olması, yüksek miktarda damga verisinin gömüldüğü durumlarda bile gözle fark edilme olasılığını düşürecektir.

Orjinal Metin	Bilgehan
Homoglif Metin	Bilgehan

Şekil 2: Damga verisi olarak "101001" gömülen orijinal ve damgalanmış metin.

Şekil 2'ye dikkatle bakıldığında damga verisi olarak "1" gömülen "B", "g" ve "a" karakterlerinden "g" karakterinin homoglif karşılığı, aynı metinde orijinal "g" karakteri olduğu

durumda gözle fark edilebilecek derecede farklılık göstermektedir. Damgalama kapasitesinden ödün vermeden, önerilen yöntemin sağlamlığının artırılması için homoglif dönüşümde kullanılacak karakterler "algılanabilir" ve "algılanamaz" olarak iki gruba ayrılır. Algılanamaz gruba dâhil edilen karakterler, orijinal karakter ile arasında minimum noktasal fark olan karakterlerdir. Algılanabilir gruba dâhil edilen karakterler ise orijinal karaktere benzemesinin yanında, aynı metin içerisinde orijinal karakter olması durumunda insan gözü tarafından ayırt edilebilme olasılığı olanlardır. Homoglif dönüşümde kullanılacak karakterlerin fark edilebilirlik durumuna göre gruplanması sonucu ortaya çıkan karakter grupları Şekil 3'te verilmiştir. Gruplama sonucu 31 karakter algılanamaz, 20 karakter ise algılanabilir olarak belirlenmiştir. Toplamda homoglif dönüşüm için kullanılabilecek 51 karakter ve dönüşümde yerlerine geçecek olan Unicode karşılıkları Tablo 1'de verilmiştir.

Karakter	Algılanabilir	Algılanamaz
Orjinal	$%2345679DPQRUVbghru	!ABCEFGHJKLMNSTXYZacdeijopsvxyz
Homoglif	$ % 2 3456 7 9 DPQRUVbgհru	!ABCEFGHJKLMNSTXYZacdeijopsvxyz

Şekil 3: Homoglif dönüşümde kullanılan karakterlerin algılanabilirlik durumuna göre gruplanması.

Tablo 1 incelendiğinde, alfa nümerik karakterler dışında özel karakterlerin de yer aldığı görülecektir. Bu sayede metin ve nümerik alanlar üzerinde damgalama yapılabildiği gibi, metin alanın içerdiği özel karakterler de damgalama aşamasında kullanılabilmektedir.

Homoglif dönüşümde kullanılacak karakterler fark edilebilirlik durumuna göre gruplandığı için, damgalama aşamasında kullanılacak karakterlerin algılanabilirlik oranı da değişim gösterebilmektedir. Kullanıcıdan alınacak [0-1] aralığındaki "Algılanabilirlik Oranı" ile homoglif dönüşümde kullanılacak karakterler seçilir. Örneğin, kullanıcı Algılanabilirlik Oranı olarak "0" girmişse, homoglif dönüşüm aşamasında sadece Şekil 3'te verilen algılanamaz karakter grubu kullanılacaktır. Algılanabilirlik oranı 1 değerine yakınsadıkça, Şekil 3'te verilen algılanabilir karakter listesindeki karakterler de homoglif dönüşüm sürecine dâhil edilir. Kullanıcı algılanabilirlik oranı olarak 0.5 değeri seçmişse, homoglif dönüşüm için algılanamaz karakterler ile algılanabilir karakterlerin ilk 10 tanesi kullanılacak demektir. Algılanabilirlik Oranın 1'e yakınsaması değişimde kullanılacak karakter kümesini büyültmekte, dolayısıyla değişim olasılığını ve damgalama kapasitesini artırmaktadır. Orijinal karakterler ile yerlerine kullanılan Homoglif karşılıkları olan Unicode karakterlerin saklanmasında kullanılan bit sayıları aynı olacağından, damgalama sonrası orijinal verinin boyutu değişmeyecektir. Kullanıcının seçeceği algılanabilirlik oranına göre dönüşümde kullanılacak karakterleri belirleyen karakter kümesinin seçimi (3) ifadesinde verilmiştir.

$$P = \{\%, \$, D, P, Q, R, U, V, b, g, h, r, u, 2,3,4,5,6,7,9\} \quad (1)$$

$$IP = \left\{ \begin{matrix} !, A, B, C, E, F, G, H, J, K, L, M, N, S, T, X, Y, Z, \\ a, c, d, e, i, j, o, p, s, v, x, y, z \end{matrix} \right\} \quad (2)$$

$$HG = IP \parallel \text{First}\left(P, \lceil size(P) * P_p \rceil\right) \quad (3)$$

P ifadesi algılanabilir karakter kümesi, IP ifadesi algılanamaz karakter kümesi, P_p ifadesi ise kullanıcı tarafından belirlenen Algılanabilirlik Oranı ve $\parallel$ sembolü birleştirme işlemidir. $First(X,Y)$ metodu X kümesinden ilk Y adet elemanı bulmaktadır. Homoglif dönüşümde kullanılacak karakter kümesini simgeleyen HG, algılanamayan karakter kümesi IP ile kullanıcının belirlediği P_p algılanabilirlik oranında algılanabilir karakter kümesi P'den alınan karakterlerden oluşturulmaktadır.

Tablo 1: Homoglif dönüşüm sürecinde kullanılan benzer Unicode karakterler

Sembol	*Bit 0*	*Bit 1*	*Sembol*	*Bit 0*	*Bit 1*
	Orj. Kod	*Hmg. Kod*		*Orj. Kod*	*Hmg. Kod*
$	0x24	0xff04	a	0x61	0x430
%	0x25	0xff05	b	0x62	0x15af
!	0x21	0x1c3	c	0x63	0x3f2
A	0x41	0x391	d	0x64	0x501
B	0x42	0x392	e	0x65	0x435
C	0x43	0x421	g	0x67	0x261
D	0x44	0x13a0	h	0x68	0x4bb
E	0x45	0x395	i	0x69	0x456
F	0x46	0x3dc	j	0x6a	0x3f3
G	0x47	0x13c0	o	0x6f	0x3bf
H	0x48	0x397	p	0x70	0x440
J	0x4a	0x408	r	0x72	0x433
K	0x4b	0x39a	s	0x73	0x455
L	0x4c	0x216c	u	0x75	0x1d1c
M	0x4d	0x39c	v	0x76	0x1d20
N	0x4e	0x39d	x	0x78	0x445
P	0x50	0x3a1	y	0x79	0x443
Q	0x51	0xff31	z	0x7a	0x1d22
R	0x52	0x13d2	2	0x32	0xff12
S	0x53	0x405	3	0x33	0x417
T	0x54	0x3a4	4	0x34	0x13ce
U	0x55	0x144c	5	0x35	0x1bc
V	0x56	0x474	6	0x36	0x431
X	0x58	0x2169	7	0x37	0xff17
Y	0x59	0x3a5	9	0x39	0xff19
Z	0x5a	0x396			

Damga verisinin çıkarımı aşamasında yetkili kullanıcı tarafından damga yerleştirme aşamasında da kullanılan gizli anahtar ve algılanabilirlik oranı kullanıldığından, damga verisine yetkisiz kullanıcıların erişimi söz konusu değildir. Aynı zamanda damga verisinin çıkarımı sonrası, orijinal veri de kayıpsız olarak elde edilebilmektedir.

Damga yerleştirilecek satırların seçiminde (4) ifadesinden faydalanılmaktadır. Özüt fonksiyonun H, gizli anahtarın κ ile temsil edildiği eşitlik ifadesinde birleştirme işlemi ise $\parallel$ sembolü ile gösterilmiştir. H özüt fonksiyonu kullanımı sayesinde kullanıcı tarafından seçilen gizli anahtarın türünden (sayısal, alfa nümerik, kriptografik anahtar) bağımsız olarak damga yerleştirilecek satır seçimi güvenli bir şekilde yapılmaktadır.

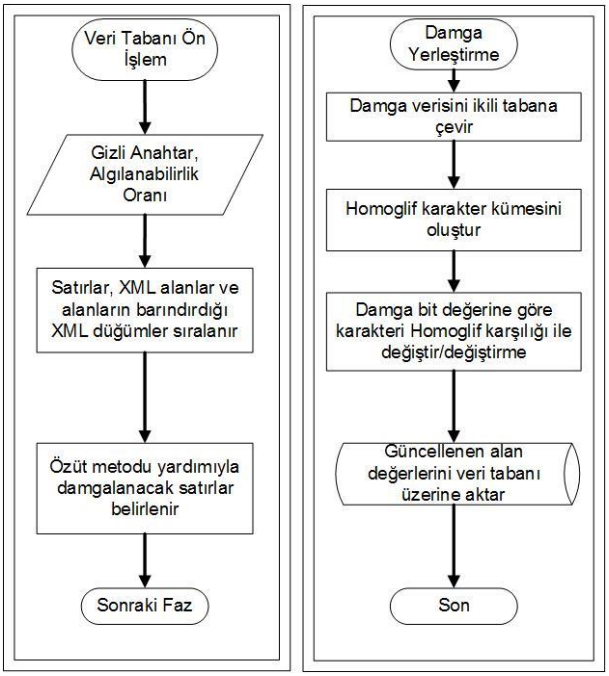
$$F(r.P) = \mathcal{H}\left(\kappa \parallel \mathcal{H}(\kappa \parallel (r.P))\right) \quad (4)$$

(4) ifadesinde $F(r.P)$ fonksiyonundan elde edilecek değerin r ile temsil edilen kaydın birincil anahtar değeri P ve gizli anahtar değeri κ ile doğrudan ilişkili olduğu görülmektedir.

3.1.1 Damga ekleme algoritması

Damgalama sürecine ait akış şeması Şekil 4'te verilmiştir. Kullanıcıdan gizli anahtar ve algılanabilirlik oranı alındıktan sonra veri tabanı sahip olduğu birincil anahtar değeri dikkate alınarak artan sırada sıralanır. Veri tabanlarında birincil anahtar değerleri aynı zamanda indeksli alanlar olduğundan bu sıralama işlemi zaman kaybı olmadan gerçek zamanlı olarak yapılabilmektedir. Ek olarak XML alanlar ve her bir XML alanın düğümleri de alfabetik sıralı olacak şekilde yer değiştirilmiştir. Bu sayede veri setinin XML alanlarına veya XML alanların içerdiği düğümlere uygulanacak yer değiştirme saldırılarına karşı algoritmanın dayanıklılığı sağlanacaktır. XML alanların sıralama işlemi veri tabanı üzerinde fiziksel bir değişime sebep olmadan algoritma tarafından kendi içinde sıralanması yöntemiyle yapılmaktadır.

Veri tabanından damgalama için kullanılacak kayıtların seçiminde $F(r.P)\ mod\ \kappa = 0$ ifadesinden yararlanılacaktır. Kriteri sağlayan kayıt damga yerleştirmek üzere seçilecektir. Veri tabanında birden çok XML alan olması durumunda, damgalanmak üzere seçilen kaydın $F(r.P)\ mod\ v$ ifadesi ile damga yerleştirilecek XML alan indisi belirlenecektir. v simgesi veri tabanının barındırdığı XML alanların sayısını belirtmektedir.



Şekil 4. Damga yerleştirme algoritmasına ait akış şeması.

Damga yerleştirilecek XML alan seçildikten sonra damga verisi ikilik tabana dönüştürülür ve P_p dikkate alınarak HG homoglif karakter kümesi elde edilir. Ardından damga yerleştirilecek XML alan verisinin düğümleri içerisinde gezilerek etiketler arasındaki metinler içerisinde Homoglif Dönüşüm için kullanılabilecek karakterler aranmaktadır. Damgala metodu, kendisine parametre olarak verilen metin içerisinde HG homoglif karakter kümesinde yer alan karakterlerden herhangi birini bulduğunda, damga verisinin sıradaki bit değerini yerleştirmekle görevlidir. İncelenen veri HG karakter kümesindeki karakterlerden herhangi birini içeriyorsa, sıradaki damga biti dikkate alınarak karakter değeri Homoglif kodu ile değiştirilmekte (damga değeri "1" ise) veya değiştirilmeden aynı bırakılmaktadır (damga değeri "0" ise). Damgalama algoritmasının yalancı kodu Şekil 5'te verilmektedir.

Giriş: R veri kümesi, $\{A_1, A_2, \dots, A_v\}$ veri alanları, W damga verisi, κ gizli anahtar
Çıkış: R_w Damgalanmış veri
for i=l to $SatırSayısı(R)$
  if $F(R_i.P)\ mod\ \kappa = 0$
    j $= F(R_i.P)\ mod\ v$
    for k=0 to $DüğümSayısı(A_i)$
      $Damgala(A_i.Düğüm_k, W_t)$
    end
  end
end

Şekil 5: Damga yerleştirme algoritmasının yalancı kodu.

3.2 Damga çıkarım algoritması

Damgalanmış veri tabanından damganın çıkarımı için damga yerleştirme sürecinde kullanıcının belirlediği gizli anahtar ve algılanabilirlik oranı gerekmektedir. Damga çıkarım süreci verinin sahipliği konusunda ihtilaf olması durumunda gerçekleştirildiğinden damga yerleştirme ve çıkarım aşamasında kullanılan gizli anahtar ve algılanabilirlik oranı bilgisinin sadece veri sahibi tarafından bilinmesi ve paylaşılmaması gerekir. Aksi takdirde saldırganlar damgalanmış veriden orijinal damga verisini çıkartarak, kendi damga verisini saklayabilmekte ve veri üzerinde sahiplik iddiasında bulunabilmektedirler. Damganın çıkarılacağı satırların belirlenmesinde damga yerleştirirken kullanılan satır seçme ifadesinden yararlanılacaktır. Satırın belirlenmesinden sonra damga verisinin saklandığı XML alan, yine damga yerleştirme aşamasında kullanılan alan seçme kıstası vasıtasıyla belirlenmektedir. Algılanabilirlik oranı dikkate alınarak Homoglif dönüşümde kullanılacak karakter kümesi oluşturulur. İlgili XML alanın tüm düğümlerindeki verinin içerdiği karakterler, oluşturulan Homoglif dönüşüm karakter kümesi ile karşılaştırılır. Homoglif dönüşüm karakter kümesinde yer alan karakterlerden herhangi birinin orijinal karakter değeri bulunduğunda çıkarılan damga biti "0", Homoglif karşılığında yer alan karakter değeri bulunduğunda ise çıkarılan damga biti "1" olarak değerlendirilmektedir. $DamgaÇıkar$ metodu çıkarılan damga bitlerini birleştirerek yerleştirilen damga verisini oluşturmaktadır. Damga çıkarım algoritmasına ait yalancı kod ifadesi Şekil 6'da verilmektedir.

Giriş: Damgalanmış R_w veri kümesi, gizli anahtar κ, veri alanları $\{A_1, A_2, \dots, A_v\}$, Damgalamada kullanılacak veri W
Çıkış: Damgalanacak veri W
for i=l to $SatırSayısı(R)$
  if $F(R_i.P)\ mod\ \kappa = 0$
    j $= F(R_i.P)\ mod\ v$
    for k=0 to $DüğümSayısı(A_i)$
      $W.Add(DamgaÇıkar(A_i.Düğüm_k))$
    end
end

Şekil 6: Damga çıkarım algoritmasının yalancı kodu

4 Deneysel sonuçlar

Deneylerin üzerinde yapıldığı veri tabanı, "1998statics.xml" isimli benzer çalışmalarda da kullanılan istatistiksel verileri içeren XML dosyasından oluşturulmuştur. 1226 düğümden oluşan XML dosyanın her bir düğümü veri setinde birer XML veri alanı haline getirilmiştir. Örnek XML düğüm yapısı Şekil 7'de verilmiştir.

```
<PLAYER>
 <NUMBER>2989</NUMBER>
 <SURNAME> John</SURNAME>
 <GIVEN_NAME>Wills </GIVEN_NAME>
 <THROWS />
 <POSITION>Relief</POSITION>
 <WINS>41</WINS>
 <LOSSES>16</LOSSES>
 <SAVES>32</SAVES>
 <GAMES>83</GAMES>
 <GAMES_STARTED>15</GAMES_STARTED>
 <COMPLETE_GAMES>11</COMPLETE_GAMES>
 <SHUT_OUTS>21</SHUT_OUTS>
 <ERA>4.45</ERA>
 <INNINGS>81</INNINGS>
 <HOME_RUNS>91</HOME_RUNS>
 <RUNS>8</RUNS>
 <EARNED_RUNS>56</EARNED_RUNS>
 <HIT_BATTER>4</HIT_BATTER>
 <WILD_PITCHES>3</WILD_PITCHES>
 <BALK>3</BALK>
 <WALKED_BATTER>0</WALKED_BATTER>
 <STRUCK_OUT_BATTER>19</STRUCK_OUT_BATTER>
</PLAYER>
```

Şekil 7: Örnek XML veri yapısı.

Test ortamı MicroSoft SQL Server 2012 veri tabanı yönetim sistemi üzerinde MicroSoft Visual Studio IDE ortamında C# programlama dili ile kodlanmıştır. Bu bölümde önerilen yöntem ve literatürdeki benzer çalışmalar farklı kriterler dikkate alınarak karşılaştırılmıştır. Damgalama algoritmalarının en önemli özelliklerinden kapasite ve algılanabilirlik analizi gerçekleştirilmiştir. XML alanlar üzerinde damgalama gerçekleştiren algoritmalara yapılan saldırılar, metin ve nümerik alanlar üzerinde damgalama gerçekleştiren algoritmalara yapılan saldırılara göre farklılık göstermektedir. Bu sebeple XML verinin bütünlüğünü ve kullanılabilirliğini bozmadan damgayı yok etmeye yönelik literatürde gerçekleştirilen saldırılar ile önerilen algoritmanın sağlamlığı test edilmiştir.

4.1 Kapasite ve algılanabilirlik analizi

Saldırganlar damga verisini bozmak veya tamamen ortadan kaldırmak için saldırılar gerçekleştirmektedirler. Bu sebeple damga verisinin fark edilebilirliğinin düşük olması önemlidir. Önerilen çalışmada Homoglif Dönüşüm yöntemi kullanımı sayesinde, damga verisi metinler üzerine eklenmesine rağmen fark edilebilirliği minimum düzeydedir. Homoglif Dönüşümde kullanılacak karakterlerin algılanabilirlik oranının kullanıcı tarafından belirlenebilmesi, kullanıcının damgalama kapasitesi ile algılanabilirlik arasındaki dengeyi gereksinimine göre belirleyebilmesini sağlamaktadır. Yüksek damgalama kapasitesi yerine damganın gizliliğinin önemli olduğu tıbbi, güvenlik ve askeri veri tabanlarındaki uygulamalarda kullanıcı saldırganlar tarafından fark edilebilirliği azaltmak için algoritmanın algılanabilirlik oranının 0'a yakın seçebilecektir. Web ortamında bilginin sunulduğu veri tabanları gibi algılanabilirlikten daha çok kapasitenin önem kazandığı durumlarda algılanabilirlik oranının 1'e yakın olması ile homoglif dönüşümde kullanılacak karakter kümesinin büyümesi sağlanarak damgalama kapasitesi maksimize edilebilmektedir. Şekil 8'de örnek bir XML kaydın orijinal ve farklı algılanabilirlik oranları ile damgalanmış halinin HTML çıktıları sunulmaktadır.

Pp	Data
Original	Trabzon Milletvekili Mustafa Reşit Tarakçıoğlu ve 28 arkadaşının verdiği teklifin TBMM'de 20 Mayıs 1955 tarih ve 6594 sayılı kanunla kabul edilmesi ile kurulmuş olan Karadeniz Teknik Üniversitesi İstanbul ve Ankara illeri dışında kurulan ilk üniversitedir.
0	Trabzon Milletvekili Mustafa Reşit Tarakçıoğlu ve 28 arkadaşının verdiği teklifin TBMM'de 20 Mayıs 1955 tarih ve 6594 sayılı kanunla kabul edilmesi ile kurulmuş olan Karadeniz Teknik Üniversitesi İstanbul ve Ankara illeri dışında kurulan ilk üniversitedir.
0.5	Trabzon Milletvekili Mustafa Reşit Tarakçıoğlu ve 28 arkadaşının verdiği teklifin TBMM'de 20 Mayıs 1955 tarih ve 6594 sayılı kanunla kabul edilmesi ile kurulmuş olan Karadeniz Teknik Üniversitesi İstanbul ve Ankara illeri dışında kurulan ilk üniversitedir.
1	Trabzon Milletvekili Mustafa Reşit Tarakçıoğlu ve 28 arkadaşının verdiği teklifin TBMM'de 20 Mayıs 1955 tarih ve 6594 sayılı kanunla kabul edilmesi ile kurulmuş olan Karadeniz Teknik Üniversitesi İstanbul ve Ankara illeri dışında kurulan ilk üniversitedir.

Şekil 8: Örnek XML verisine farklı P_p değerleri ile damgalama.

P_p simgesinin algılanabilirlik oranını temsil ettiği şekilde, ilk satırdaki veri orijinal XML verisinin HTML çıktısını göstermektedir. P_p değerinin 0 alınarak algılanabilirliğin minimum olduğu durumda damgalama sonrası elde edilen metin ikinci satırda verilmiştir. 106 karakter üzerinde homoglif dönüşüm yapılmasına rağmen damgalanan karakterlerin göz ile algılanabilmesi çok zordur. P_p değerinin 0.5 alınarak gerçekleştirilen damgalama sonrası elde edilen metin üçüncü satırda verilmiştir. Algılanabilirlik oranının artması, homoglif dönüşümde kullanılacak karakter kümesinin büyümesini sağladığından 110 karakterlik damgalama kapasitesine ulaşılmıştır. Algılanabilirlikten çok damgalama kapasitesinin önemli olduğu son satırda, P_p parametresi için 1 değeri seçilmiştir. Bu durumda damga olarak yerleştirilmiş bit sayısı 143'e yükselmiştir. Önerilen algoritmada damga verisinin 0 bit değerleri için homoglif dönüşüm yapılmayacağından, damga verisi içerisindeki 0 bit sayısının artması damgalanan verinin gizliliğini artıracaktır. Şekil 8'de örnek XML verisine damgalanacak veri bitlerinin tamamı 1 seçilerek damga yerleştirilerek her bir karakter homoglif karşılığı ile değiştirilmiştir. P_p değerinin 0, 0.5 ve 1 alındığı durumlarda tamamı 1 olan 106, 110 ve 143 bit veri saklanmıştır. Bu senaryo damga verisinin güvenliğinin en düşük olduğu durumdur. Buna rağmen P_p değerinin 0 seçildiği durumda damgalanan karakterlerin algılanabilmesi mümkün değildir.

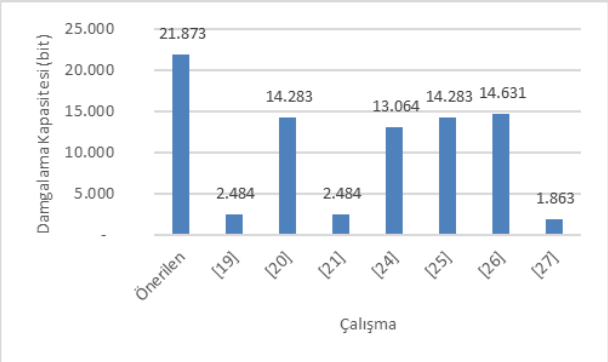
Önerilen yöntem ve literatürde XML verilere damgalama gerçekleştiren çalışmaların "<city>Los Angeles</city>" verisine damga yerleştirme sonrası saldırılara karşı fark edilemezlik durum karşılaştırmaları Tablo 2'de verilmiştir. Tabloda algoritmaların fark edilemezlik durumları ve örnek veriye damga verisi yerleştirmelerinden sonra elde edilen damgalanmış veri sunulmuştur. Guo ve diğ. [19] yaptıkları çalışmaya benzer şekilde, Wilfred ve diğ. [21] yaptığı çalışmada da damgalama sırasında kelimelerin sinonimleri ile değiştirilmesinden dolayı orijinal dosya boyutu değişecek,

metinlerde oluşabilecek anlamsal problemler sonucu damga verisinin algılanması kolaylaşacaktır. Inoue ve diğ. [20], Saad ve diğ. [25], Tchokpon ve diğ. [27] yaptıkları çalışmalarda boşluk ekleme yönteminin yanı sıra, Guo ve diğ. [19] ile Wilfred ve diğ. [21] çalışmalarında olduğu gibi sinonim kullandıklarından dolayı oluşacak ekstra boşluklar saldırganların dikkatini çekecektir. Saad ve diğ. bir diğer çalışmasında damga verisinin eklenmesi aşamasında büyültülen karakterler sebebiyle aynı kelimede yazım kurallarına uymayan büyük-küçük harf olması damganın fark edilebilmesine sebep olacaktır [24]. Ronghua ve diğ. çalışmalarında etiket karakterleri üzerinde büyültme işlemi gerçekleştirdiklerinden, damga verisinin fark edilebilme ihtimali yüksek olacaktır [26]. Literatürdeki XML verilere damgalama gerçekleştiren diğer çalışmalar ile karşılaştırıldığında, önerilen algoritmanın damga verisi fark edilemezdir.

Tablo 2: Literatürde XML damgalama yapan algoritmaların fark edilebilirlik durumları.

Çalışma	Fark Edilemezlik	Damgalanmış Veri
Önerilen	✓	<city>Los Angeles</city>
[19]	✕	<city>L.A.</city>
[20]	✕	<city >Los Angeles</city >
[21]	✕	<city>L.A.</city>
[24]	✕	<city>LoS AnGelEs</city>
[25]	✕	<city>Los Angeles</city >
[26]	✕	<CiTy>Los Angeles</CiTy>
[27]	✕	<city>Los An gele s</city>

Kapasite testlerinde veri tabanında bulunan 1226 satırdan kullanıcının seçmiş olduğu gizli anahtar vasıtasıyla belirlenen 621 aday satırlara damga verisi gömülmüştür. Önerilen algoritmanın ve literatürde XML verilerin damgalanmasını sağlayan algoritmaların damgalama kapasitelerinin karşılaştırılması Şekil 9'da verilmiştir. Önerilen algoritma kullandığı Homoglif Dönüşüm sayesinde metinler üzerinde yüksek damgalama kapasitesine sahiptir. Örnek veri seti için 621 aday satır üzerinde 21.873 bit veriyi saklayabilmiştir. Bu damgalama kapasitesine en yakın çalışma Upper-Lower Coding yöntemini kullanarak 14.631 bit veri gömebilen [26] çalışmasıdır.



Şekil 9: Literatürde XML damgalama yapan bazı algoritmaların damgalama kapasitelerinin örnek veri setinde karşılaştırılması.

4.2 Sağlamlık analizi

Sağlamlık analizi damgalama yapılmış veri tabanına saldırı yapıldıktan sonra damga çıkarım algoritması ile elde edilebilen damga verisinin doğruluk oranı dikkate alınarak yapılmaktadır. XML içeren veri tabanlarına yapılan saldırılar, sadece nümerik veya metin değerlerden oluşan veri tabanlarına yapılan saldırılara göre farklılık göstermektedir. Bu saldırılar Element Sıra Karıştırma, Etiket Ad Karıştırma, İçerik Silme ve Alan Silme/Karıştırma saldırılardır.

XML verinin yapısal özelliğinden dolayı, bir etiketin içerdiği alt etiketlerin sıralarının değişimi verinin bozulmasına sebep olmaz. *Element Sıra Karıştırma* saldırısı XML'in bu özelliği kullanılarak yapılmaktadır. Saldırganlar XML verinin yapısı üzerinde değişiklik yaparak veriyi bozmadan damgayı ortadan kaldırmaya çalışmaktadırlar.

Veriyi barındıran XML açma ve kapama etiketlerinin büyük/küçük harf duyarlı bir şekilde birbirinin aynı olması gerekir. Fakat etiket karakterlerinin büyük veya küçük olması veriyi etkilememektedir. *Etiket Ad Karıştırma* saldırı türünde, XML etiket karakterlerinin büyültülüp/küçültülmesi ile etikete yerleştirilen damganın zarar görmesi sağlanabilir.

Alan Silme/Karıştırma ismi verilen saldırı türünde ise XML etiket özelliklerinin yerlerinin değiştirilmesi veya etiket özelliklerinin silinmesi, etiket özellikleri kullanılarak yerleştirilen damga verisinin kaybolmasına yol açabilir.

XML etiketlerinin içerdiği verinin bir kısmının veya tamamının silinmesi yöntemiyle gerçekleştirilen *İçerik Silme* saldırısı damga verisinin eklendiği içerikleri hedef aldığında damganın kaybolmasına sebep olacaktır.

Tüm bu saldırı türleri örnek veri seti üzerinde gerçekleştirilerek sonuçları Tablo 3'te sunulmuştur. Guo ve diğ. [19], Inoue ve diğ. [20], Wilfred ve diğ. [21], Ronghua ve diğ. [26] çalışmaları Element Sıra Karıştırma saldırısından etkilenecek ve yerleştirdikleri damganın tamamını çıkaramayacaklardır. Inoue ve diğ. [20] ile Saad ve diğ. [25] çalışmalarında damga yerleştirme sürecinde XML düğümün kapama etiketine boşluk eklendiğinden Etiket Ad Karıştırma saldırısından algoritmalar etkilenecek ve damganın bütünlüğü bozulacaktır. Benzer şekilde Ronghua ve diğ. çalışmalarında damgalama işleminde etiketlerin karakterleri büyük/küçük karakter karşılıkları ile değiştirildiğinden Etiket Ad Karıştırma saldırısı bu çalışmada da damga verisinin bozulmasına sebep olacaktır [26]. Alan Silme/Karıştırma saldırısı ise Inoue ve diğ. çalışmasını etkileyecek ve yerleştirilen damganın kaybolmasına sebep olacaktır [20].

Tablo 3: Literatürde XML damgalama yapan bazı algoritmaların saldırılara karşı dayanıklılıkları.

Algoritma	Saldırılar			
	Element Sıra Karıştırma	Etiket Ad Karıştırma	İçerik Silme	Alan Silme/Karıştırma
Önerilen	Etkilenmez	Etkilenmez	Etkilenir	Etkilenmez
[19]	Etkilenir	Etkilenmez	Etkilenir	Etkilenmez
[20]	Etkilenir	Etkilenir	Etkilenir	Etkilenir
[21]	Etkilenir	Etkilenmez	Etkilenir	Etkilenmez
[24]	Etkilenmez	Etkilenmez	Etkilenir	Etkilenmez
[25]	Etkilenmez	Etkilenir	Etkilenir	Etkilenmez
[26]	Etkilenir	Etkilenir	Etkilenir	Etkilenmez
[27]	Etkilenmez	Etkilenmez	Etkilenir	Etkilenmez

Önerilen çalışma damga yerleştirme ve çıkarma işlemi öncesinde etiketleri alfabetik olarak sıraladığından, Element Sıra Karıştırma saldırısından etkilenmeyecektir. Veri saklama işlemi etiketler içerisine yapılmadığından Etiket Ad Karıştırma saldırısı da önerilen algoritmanın damgalama başarını

etkilemeyecektir. Benzer şekilde Alan Silme/Karıştırma saldırısı önerilen çalışmanın damgalama sağlamlığını olumsuz yönde etkilemeyecektir. İncelenen saldırılar içerisinde önerilen çalışmanın sağlamlığını etkileyen tek saldırı literatürdeki tüm çalışmaları da etkileyen, veri kaybın kaçınılmaz olduğu İçerik Silme saldırısıdır.

Gerçekleştirilen saldırılar sonrasında önerilen algoritmanın karşılaştırılan diğer algoritmalara göre XML verinin silinmesi dışındaki tüm saldırılara karşı dayanaklı ve saldırganlar tarafından fark edilemez olduğu görülmüştür.

4.3 Benzer yöntemler ile karşılaştırılması

Literatürde XML veri damgalama üzerine yapılan çalışmalar ile önerilen algoritmanın sağladığı özellikler karşılaştırılmıştır. Çalışmaların desteklekdikleri veri türü, damga çıkarımı için ihtiyaç duyulan ekstra bilgilerin saklanma ihtiyacı, damgalama sonrası veri boyutunun değişimi, damga yerleştirme yöntemi, geri dönüşüm desteği, veri üzerinde bozulmaya sebep olup olmadığı gibi özellikler dikkate alınarak oluşturulan karşılaştırma tablosu Tablo 4'te verilmiştir.

XML veri alanı içerisinde tutulan verinin türü metin veya nümerik olabilmektedir. Önerilen algoritma XML alanın içerdiği veri türünden bağımsız olarak çalışabilmektedir. Bu sayede XML etiketlerin içerdiği metin ve nümerik veri türlerinin her ikisine de damgalama desteğine sahiptir. Guo ve diğ. [19], Wilfred ve diğ. [21] ile Wen ve diğ. [23] çalışmaları metin ve nümerik veri türlerinin her ikisini de desteklerken Inoue ve diğ. [20], Saad ve diğ. [24],[25], Ronghua ve diğ. [26] ile Tchokpon ve diğ. [27] çalışmaları sadece metin veriler üzerinde damgalama yapabilmektedirler.

Önerilen algoritmada yerleştirilen damga verisinin çıkarımı için gerekli olan bilginin farklı bir konumda saklanması gerekmemektedir. Damga çıkarım süreci için damgalama aşamasında kullanıcı tarafından belirlenen gizli anahtar ve algılanabilirlik oranı yeterlidir. Inoue ve diğ. [20] ile Ronghua ve diğ. [26] çalışmaları damga çıkarım işlemi için farklı bir konumda saklanan veriye ihtiyaç duymazken Guo ve diğ. [19], Wilfred ve diğ. [21], Wen ve diğ. [23], Saad ve diğ. [24],[25] ile Tchokpon ve diğ. [27] çalışmaları damganın geri dönüşümü gerekli veriyi farklı bir konumda saklamak zorundadırlar.

Orijinal veri tabanına yerleştirilen damga verisinin boyutu ne kadar fazla olursa olsun, önerilen algoritma XML verinin veya dosya boyutunun değişmesine sebep olmamaktadır. Literatürde incelenen benzer uygulamalardan Saad ve diğ. [24] ile Ronghua ve diğ. [26] çalışmaları bu özelliğe sahiptir. Guo ve diğ. [19], Inoue ve diğ. [20], Wilfred ve diğ. [21], Wen ve diğ. [23], Saad ve diğ. [25] ile Tchokpon ve diğ. [27] çalışmaları damgalama sonrası orijinal verinin boyutunda değişime sebep olarak saldırılara karşı zaaf göstermektedirler. Damgalama algoritmalarının yerleştirdiği damga verisini çıkardıktan sonra damgalama öncesi orijinal veriyi hatasız olarak elde edebilmesi istenir. Geri dönüşüm özelliği olarak bilinen bu özellik önerilen algoritma ve literatürde incelenen algoritmalar arasında Saad ve diğ. [25], Ronghua ve diğ. [26] ile Tchokpon ve diğ. [27] çalışmalarında sağlanmaktadır. Buna karşın Guo ve diğ. [19], Inoue ve diğ. [20], Wilfred ve diğ. [21], Wen ve diğ. [23] ile Saad ve diğ. [24] çalışmaları damga çıkarım sonrası orijinal veriyi elde edememektedirler.

İdeal damgalama algoritmalarında, damgalama işleminin orijinal veri üzerinde olabildiğince az miktarda değişime sebep olması istenmektedir. Önerilen algoritma kullanmış olduğu Homoglif Dönüşüm sayesinde damgalama gerçekleştirdiği verilerin içerdiği nümerik değerler üzerinde bir değişim, metinler üzerinde farklı bir kelime kullanımı veya fazladan boş ekleme gibi işlemler gerçekleştirmediğinden orijinal veri üzerinde tahribata neden olmamaktadır. Literatürde incelenen çalışmalardan Wen ve diğ. [23], Saad ve diğ. [25], Ronghua ve diğ. [26] ile Tchokpon ve diğ. [27] çalışmaları bu özelliği taşırken Guo ve diğ. [19], Inoue ve diğ. [20], Wilfred ve diğ. [21] ile Saad ve diğ. [24] çalışmaları damga yerleştirme sürecinde metinler üzerinde sinonim kelime kullanımı, boşluk ekleme, karakter büyültme/küçültme gibi işlemlerden dolayı orijinal verinin bozulmasına sebep olmaktadırlar. Önerilen algoritma karakterler üzerinde Homoglif dönüşüm gerçekleştirdiğinden fark edilemezliği yüksektir. Damga verisini çıkardıktan sonra damgalama öncesi orijinal veriyi elde edebilmektedir. XML verinin barındırdığı nümerik ve metin veri türlerine damga yerleştirebilmektedir. Damga çıkarım işlemi için farklı konumda saklanan veriye ihtiyacı yoktur. Damgalama sonrası orijinal verinin bozulmasına sebep olmamaktadır.

Tablo 4: Literatürde XML damgalama yapan bazı algoritmaların özellik karşılaştırmaları.

Algoritma	Yıl	Damga Veri Türü		Damga Çıkarım İçin Ekstra Bilgi Saklama Gerekliliği	Veri Boyutunu Değiştirme	Damgalama Metodu		Orijinal Veriye Geri Dönme	Bozulmaya Sebep Olma	Damga Boyutu (bit)
		Sayısal	Metin			Sayısal	Metin			
Önerilen	2017	✓	✓	✕	✕	Homoglif	Homoglif	✓	✕	21.873
[19]	2006	✓	✓	✓	✓	LSB	Sinonim	✕	✓	2.484
[20]	2001	✕	✓	✕	✓	✕	Sinonim, Boşluk Ekleme, Sıralama	✕	✓	14.283
[21]	2005	✓	✓	✓	✓	LSB	Sinonim	✕	✓	2.484
[23]	2016	✓	✓	✓	✓	✕	--	✕	✕	-
[24]	2010	✕	✓	✓	✕	✕	Karakter Büyültme/Küçültme	✕	✓	13.064
[25]	2012	✕	✓	✓	✓	✕	Boşluk Ekleme	✓	✕	14.283
[26]	2006	✕	✓	✕	✕	✕	ULC	✓	✕	14.631
[27]	2012	✕	✓	✓	✓	✕	Boşluk Ekleme	✓	✕	1.863

Tüm bu özellikleri bünyesinde barındıran tek çalışma olarak gerçekleştirilen saldırılara karşı dayanıklılığı üst düzeydedir. Damgalama algoritmalarının başarısını belirleyen kriterlerden bir diğeri de damgalama kapasitesidir. Literatürdeki benzer çalışmalar ile karşılaştırıldığında aynı aday veri seti üzerinde önerilen çalışma 21.873 bit veri saklama kapasitesine sahipken, bu değere en yakın çalışma olan [26] çalışması 14.631 bit saklayabilmiştir.

5 Değerlendirme

Bu çalışmada ilişkisel veri tabanlarında XML veri türündeki alanlar üzerinde Homoglif Dönüşüme dayalı damgalama yöntemi önerilmiştir. Damga yerleştirilecek kayıtların seçiminde özüt fonksiyondan yararlanılmıştır. Literatürde XML veri türleri üzerinde damgalama yapan çalışmalar göz önüne alındığında önerilen çalışma XML alanlarda tutulan nümerik ve metin veri türleri üzerinde uygulanabilir olması, damga çıkarım aşaması için ekstra bilgi saklama gereksinimi bulundurmaması, eklenen damganın verinin boyutunu değiştirmemesi, damga çıkarım sonrasında orijinal veriyi hatasız elde edebilmesi, damgalama sebebiyle veri üzerinde bozulmaya sebep olmaması ve fark edilebilirliğin düşük olması özelliklerinin tamamını üzerinde barındıran tek çalışmadır. İlerleyen çalışmalarda önerilen algoritmaya XML veri alanları ile birlikte nümerik ve metin verisi bulunduran alanlar üzerinde de damga yerleştirme özelliği katılarak, farklı veri türleri içeren veri setlerinin tüm alanları üzerinde damgalama gerçekleştirmesi sağlanması düşünülmektedir.

6 Kaynaklar

[1] Agrawal R, Haas PJ, Kiernan J. "Watermarking relational data: framework, algorithms and analysis". *The International Journal on Very Large Data Bases,* 12(2), 157-169, 2003.

[2] Gupta G, Pieprzyk J. "Database relation watermarking resilient against secondary watermarking attacks". *5th International Conference on Information Systems Security*, Kolkata, India, 14-18 December 2009.

[3] Zhang Y, Niu X, Zhao D. "A method of protecting relational databases copyright with cloud watermark". *International Journal of Information and Communication Engineering*, 1, 337-341, 2005.

[4] Guo F, Wang J, Zhang Z, Ye X, Li D. "An improved algorithm to watermark numeric relational data". *6th International Workshop on Information Security Applications*, Jeju Island, Republic of Korea, 22-24 August 2005.

[5] Huang M, Cao J, Peng Z, Fang Y. "A new watermark mechanism for relational data". *4th International Conference on Computer and Information Technology*, Wuhan, China, 16 September 2004.

[6] Hu T, Chen G, Chen K, Dong J. "Garwm: towards a generalized and adaptive watermark scheme for relational data". *6th International Conference in Advances in Web-Age Information Management*, Hangzhou, China, 11-13 October 2005.

[7] Sion R. "Proving ownership over categorical data". *20th International Conference on Data Engineering*, Mass, USA, 30 March-2 April 2004.

[8] Sion R, Atallah M, Prabhakar S. "Rights protection for categorical data". *IEEE Transactions on Knowledge and Data Engineering*, 17(7), 912-926, 2005.

[9] Wang C, Wang J, Zhou M, Chen G, Li D. "Atbam: an arnold transform based method on watermarking relational data". *International Conference on Multimedia and Ubiquitous Engineering*, Beijing, China, 24-26 April 2008.

[10] Zhou X, Huang M, Peng Z. "An additive-attack-proof watermarking mechanism for databases copyrights protection using image". *ACM Symposium on Applied Computing*, Seoul, Republic of Korea, 11-15 March 2007.

[11] Al-Haj A, Odeh A. "Robust and blind watermarking of relational database systems". *Journal of Computer Science*, 4(12), 1024-1029, 2008.

[12] Wang H, Cui X, Cao Z. "A speech based algorithm for watermarking relational databases". *International Symposiums on Information Processing*, Moscow, Russia, 23-25 May 2008.

[13] Li Y, Guo H, Jajodia S. "Tamper detection and localization for categorical data using fragile watermarks". *4th ACM Workshop on Digital Rights Management*, Washington, DC, USA, 25 October 2004.

[14] Kamel I. "A schema for protecting the integrity of databases". *Computers and Security*, 28(7), 698-709, 2009.

[15] Khataeimaragheh H, Rashidi H. "A novel watermarking scheme for detecting and recovering distortions in database tables". *International Journal of Database Management Systems*, 2(3), 1-11, 2010.

[16] Hamadou A, Sun X, Gao L, Shah SA. "A fragile zero-watermarking technique for authentication of relational databases". *International Journal of Digital Content Technology and its Applications*, 5(5), 189-200, 2011.

[17] Rizzo SG, Bertini F, Montesi D. "Content-preserving Text watermarking through unicode homoglyph substitution". *20th International Database Engineering & Applications Symposium*, Montreal, Canada, 11-13 July 2016

[18] The Unicode Consortium. "Unicode Security Mechanisms for UTR". http://www.unicode.org/Public/security/10.0.0/confusables.txt (20.06.2018).

[19] Guo F, Wang J, Zhang Z, Li D. "A new scheme to fingerprint XML data". *International Workshop on Knowledge Discovery from XML Documents*, 9 April 2006.

[20] Inoue S, Makino K, Murase I, Takizawa O, Matsumoto T, Nakagawa H. "A proposal on information hiding methods using XML". *First NLP and XML Workshop*, Tokyo, Japan, 30 November 2001.

[21] Wilfred Ng, Ho-Lam L. "Effective approaches for watermarking XML data". *International Symposium on Database Systems for Advanced Applications*, Beijing, China, 17-20 April 2005.

[22] Fellbaum C. *WordNet: An Electronic Lexical Database*. Cambridge, Massachusetts, The MIT Press, 1998.

[23] Wen Q, Wang Y, Li P. "Two zero-watermark methods for XML documents". *Journal of Real-Time Image Processing*, 14(1), 183-192, 2016.

[24] Saad M. "An optimized fingerprinting system for tamper proof and copy control of XML documents". *International Conference of Soft Computing and Pattern Recognition*, Paris, France, 7-10 December 2010.

[25] Saad M. "New system to fingerprint extensible markup language documents using winnowing theory". *The Institution of Engineering and Technology Signal Processing*, 6(4), 348-357, 2012.

[26] Ronghua Y, Qijun Z, Hongtao L. "A novel watermark algorithm for integrity protection of XML documents". *International Journal of Computer Science and Network Security*, 6(2), 202-207, 2006.

[27] Tchokpon R, Nadia B. "Robust XML watermarking using fuzzy queries". *Institute of Electrical and Electronics Engineers 36th International Conference on Computer Software and Applications Workshops*, 16-20 July 2012.