



WEB TABANLI SALDIRILARI ÖNLEME SİSTEMİ

 Feyza Benal¹  Ece Jilta²

¹Corresponding Author; Dept. of Computer Engineering, Bursa Uludağ University; 032090082@ogr.uludag.edu.tr

²Corresponding Author; Dept. of Computer Engineering, Bursa Uludağ University; 032090105@ogr.uludag.edu.tr

Abstract

Dünyanın daha fazla dijitalleşmesiyle birlikte web üzerindeki bilgilere erişim günden güne artmaktadır. Farkında olmadan çocuklardan yetişkinlere kadar dünyanın her yerinden kolayca ihtiyaç duyulan bilgiye web siteleri aracılığıyla ulaşılmaktadır. Bu durum web saldırılarının artmasına ve güvenlik sorunlarının ortaya çıkmasına neden olmaktadır. Web saldırıları oldukça geniş bir konu olup günümüzde yaygın bir sorundur. 2021 yılında OWASP tarafından yayınlanan En Büyük 10 Web Uygulaması Güvenlik Riski raporunda Injection Attack ilk sırada yer almaktadır. Bu saldırı türü içerisinde SQL enjeksiyonu, dijital kanıtların olduğu veritabanlarında fark edilmeden var olmuş ve 20 yılı aşkın bir süredir devam etmektedir. Ne yazık ki, web geliştiricilerinin büyük bir çoğunluğu bu güvenlik açıklarının ciddiyetinin farkında değildir. Güvenlik açıklarını gidermek için sürekli olarak araştırmalar ve geliştirmeler devam etmektedir ancak mevcut araçlar henüz tam anlamıyla etkili değildir. Bu araştırmanın amacı da Aho-Corasick desen eşleştirme algoritmasının SQL Enjeksiyon Saldırısını engellemek üzerindeki etkisini değerlendirmektir.

Keywords: Web güvenliği, SQL enjeksiyonu, veritabanları, Yapılandırılmış sorgu dili, SQLIA, Güvenlik açıkları Saldırı Tespiti, Savunma Yöntemleri

1. Introduction

2023 yılı itibarıyla <https://www.internetworldstats.com/stats.htm> tarafından yayınlanan verilere göre dünya çapındaki internet kullanıcı sayısı **5,385,798,406** nüfusuna ulaşmıştır. [1] Artan kullanıcı sayısı ile birlikte saklanan ve aktarılan veriler hızla genişlemektedir. Bu nedenle büyük hacimli uygulamaların güvenliği ve gizliliği büyük bir önem taşımaktadır. Ancak internet ortamında uygulamaların güvenliğini sağlamak her zaman kolay değildir. Web saldırıları, kullanıcıların verilerini hedef alarak ciddi riskler oluşturan bir tehdittir. Yapılandırılmış Sorgu Dili Enjeksiyon Saldırısı (SQLIA) ise çok sık görülen web saldırılarından biridir.

Web tabanlı uygulamaların büyük bir kısmı, müşterilerin hassas verileri doğrudan eklemelerine olanak tanıyan SQL Sorgu Enjeksiyon Saldırılarına karşı savunmasızdır. Saldırgan, verilere yetkisiz erişim elde etmek veya veriler üzerinde izinsiz değişiklikler yapmak amacıyla, bir web formunun giriş alanına kötü niyetli bir SQL kodu enjekte eder. Saldırgan başarılı bir şekilde işlemini gerçekleştirdikten sonra genellikle veritabanında yüksek yetkilendirme seviyesi gerektirebilecek görüntüleme, ekleme, silme gibi işlemleri gerçekleştirme yeteneğine sahip olabilir.

Bu izinsiz deęişiklikler veya silmeler, uygulamanın içeriğinde veya davranışında kalıcı deęişikliklere yol açabilmektedir.[2] Saldırganlar, güvenlik açığını sömürerek otorite, bütünlük ve gizlilik gibi unsurları tehdit edebilir. Bu nedenle finansal kurumlar, şirketler ve devlet kurumları gibi hassas bilgilere sahip olan kuruluşlar SQL enjeksiyon saldırılarının öncelikli hedefleri arasında yer almaktadır.

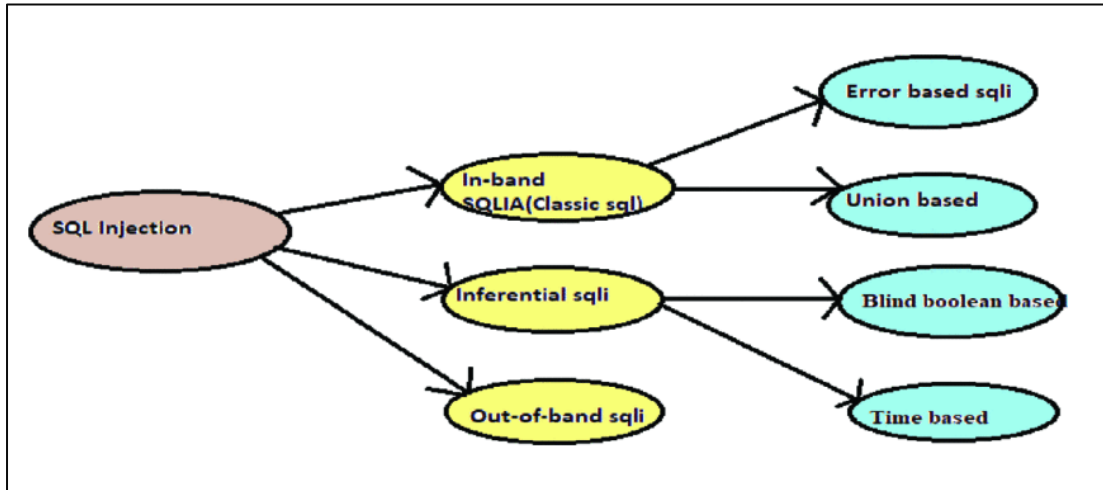
2.İlgili Çalışmalar

Web uygulamalarının birçoęu veritabanı işlemlerini gerçekleştirmek için veritabanıyla etkileşime geçmek zorundadır. Bu etkileşim sırasında, uygulamalar veri işleme bilgilerini veritabanına ileterek ilgili SQL ifadelerini çalıştırır. Bu işlem, saldırganlar için bir fırsat yaratır ve SQL enjeksiyon saldırılarına olanak tanır. Saldırganlar genellikle orijinal SQL ifadelerini deęiştirir ve web sayfasının içeriğine kötü niyetli SQL ifadelerini ekleyerek sunucuya erişim istekleri gönderir.[3] Ayrıca, saldırı amaçları doğrultusunda arka planda çalışan veritabanını yönlendirebilirler.

SQL enjeksiyon saldırılarının temelinde farklı türler ve prensipler bulunmaktadır. Bu saldırılar genellikle sayısal, karakter veya arama odaklı türlerle gerçekleştirilir ve bu türlerin saldırı prensipleri büyük ölçüde birbirine benzerlik gösterir. Bu nedenle, SQL enjeksiyon saldırılarını engellemek için birçok araştırma yapılmış ve çeşitli modeller önerilmiştir. Ancak, günümüzde SQL enjeksiyon saldırılarını tamamen önleyen kesin bir çözüm bulunmamaktadır. Bu durumda, doğru güvenlik önlemlerini ve iyi kodlama uygulamalarını benimsemek önemlidir. Web uygulamalarının güvenliğini sağlamak için, giriş verilerinin doğrulanması, parametrelerin düzgün bir şekilde kullanılması, hazır veritabanı sorguları yerine parametrelili sorguların kullanılması gibi önlemler alınmalıdır. Ayrıca, veri girişlerinin doğru bir şekilde filtrelenmesi ve sınırlandırılması, güvenlik duvarlarının etkin bir şekilde kullanılması da önemlidir.[4] Bu şekilde, riskler minimize edilebilir ve SQL enjeksiyon saldırılarının başarısızlıkla sonuçlanması sağlanabilir.

2.1Sql Enjeksiyon (sqli) Türleri

Saldırganlar, SQL enjeksiyonlarını gerçekleştirmek için üç temel yöntem kullanırlar. In-band SQLi, Inferential SQLi ve Out-of-band SQLi SQL enjeksiyonunun üç farklı türüdür. SQL enjeksiyonları, veritabanlarına erişim sağlamak ve olası hasarları değerlendirmek amacıyla kullanılır.



Şekil 1
SQL Enjeksiyon Türleri

a)Classic in-Band Sql

Temel bir SQL enjeksiyon türü olan "Bant içi" SQLIA, bir saldırganın aynı etkileşim yolunu kullanarak mantıksal bir SQL komutunu başlatması ve sonuçlarını elde etmesidir.

Bant içi SQL enjeksiyonunun en sık karşılaşılan iki yöntemi, hata tabanlı ve birleştirme tabanlı SQL enjeksiyonlarıdır.[5] Bu türün iki alt varyasyonu şunlardır:

1)Hata tabanlı SQLi

Bu yöntem veritabanı sunucusundan gelen hata mesajlarına dayanmaktadır. Bir saldırgan, kullanıcı arayüzü üzerinden geçersiz veya beklenmeyen bir giriş yaparak veritabanından hata mesajları almayı hedefler.Saldırgan, bu bilgileri kullanarak hedefe karşı saldırılarını kolayca gerçekleştirebilir.

2)Union tabanlı SQLi

Saldırganlar, bir SQL enjeksiyon saldırısı sırasında, UNION operatörünü kullanarak tek bir sorguda birden çok SQL ifadesi çalıştırma imkânı bulurlar. UNION operatörü, farklı tablolardan veya sorgulardan gelen sonuçları birleştirmek için kullanılan bir SQL operatörüdür.

b) Inferential Blind Sqli

SQL enjeksiyon saldırısı "çıkarımsal" olduğunda ne veri ne de gönderi web hizmeti üzerinden gönderilmez ve saldırgan sonucu doğrudan gözlemleyemez. Saldırgan, Tümdengelimli Mantık SQL enjeksiyonu kullanarak yükleri enjekte eder ve web uygulaması ile veritabanı sunucusunun yanıtlarını izleyerek veritabanı şemasını hızla değiştirebilir. Boole mantığı, sorunları çözmek için Boole ifadelerini kullanarak işlem yapar.[5]Bu türün iki alt varyasyonu şunlardır:

1)Boole tabanlı SQLi

Saldırganlar, SQL enjeksiyon saldırısı sırasında, veritabanı sunucusunun sonucunu (doğru veya yanlış) zorlamak için bir SQL sorgusu gönderirler. Bu saldırı yöntemi, sunucudan herhangi bir veri dönmese bile saldırganın, veri yükünün sonucunu tahmin etmesine olanak tanır.

2) Zaman tabanlı SQLi

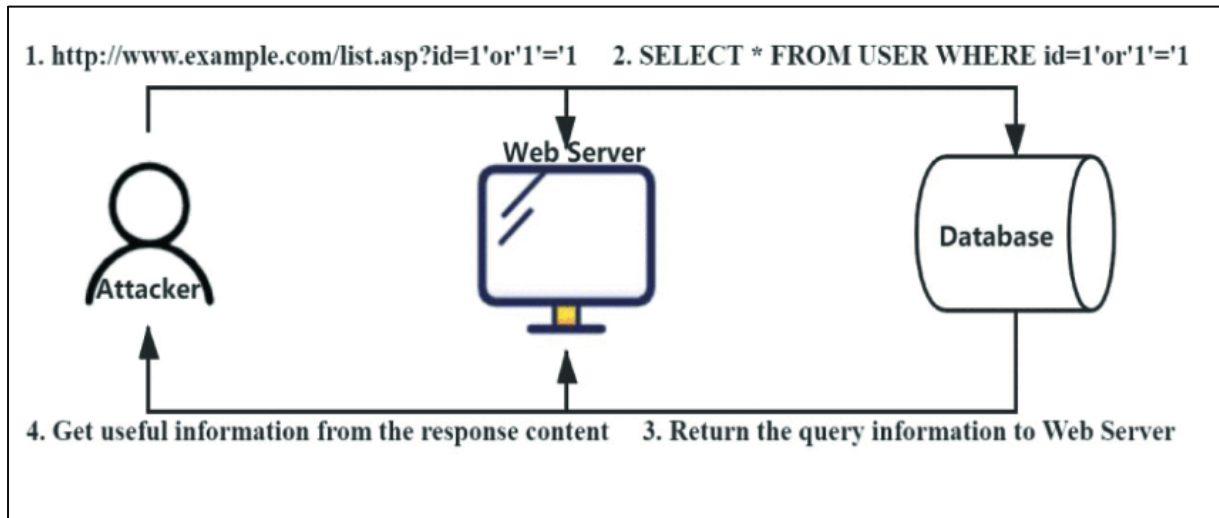
Saldırganlar, SQL enjeksiyon saldırısı sırasında, veritabanı sunucusunun yanıtını geciktirmek için bir SQL ifadesi gönderirler. Bu yöntemde, saldırgan, yanıt süresini analiz ederek, gönderilen sorgunun sonucunun doğru veya yanlış olduğunu anlayabilir.

c) Out-of Band Sqli

SQL enjeksiyonu, web uygulamasının arka uç veritabanıyla etkileşim kurarak mantığını bozmayı hedefleyen bir saldırı yöntemidir. Bu saldırı türü nadiren kullanılır çünkü SQL topluluğu tarafından dağılmıştır. Bir istenmeyen ziyaretçi, eski yöntemlerle saldırı gerçekleştiremediğinde, enjeksiyon gerçekleşir ve bu da olumsuz sonuçlara sebep olur.[5]

2.2 SQL Enjeksiyon Saldırılarının Temel Prensipleri

SQL enjeksiyon güvenlik açıkları, genellikle web uygulamalarının kullanıcı giriş verilerini yeterince doğrulamaması ve özel karakterleri etkili bir şekilde filtrelememesi gibi kod düzeyindeki kusurlardan kaynaklanır [6].Saldırganlar, genellikle web sayfası URL'leri, form giriş parametreleri veya veri paketi alanları gibi veri iletişim yolları aracılığıyla enjeksiyon noktalarını belirler ve bu noktalara değiştirilmiş SQL ifadelerini ekler.



Şekil 2

SQL enjeksiyon saldırılarının genel süreci

Sonrasında, bu değiştirilmiş SQL ifadeleriyle birlikte HTTP isteklerini Web sunucusuna göndererek ve alınan yanıt mesajlarını analiz ederek önemli verilere erişilmeye çalışılır. Başarılı bir SQL enjeksiyon saldırısı, saldırganlara hedef veritabanında saklanan verilere erişme, mevcut kayıtları manipüle etme ve hatta bazen veritabanı üzerinde tam bir kontrol sağlama olanağı sunabilir. Bu şekilde, saldırganlar hassas verilere erişebilir, veritabanını bozabilir veya yetkisiz eylemler gerçekleştirebilir. Örneğin, kullanıcı bilgilerini çalabilir, parolaları değiştirebilir, finansal verileri manipüle edebilir veya veritabanında hatalar oluşturarak sistemi çökertebilir. Bu nedenle, SQL enjeksiyon saldırıları web uygulamaları ve veritabanı güvenliği açısından ciddi bir tehdit oluşturur ve önlemek için dikkatli önlemler alınması gerekmektedir.

Mevcut "<http://www.ornek.com>" web sitesi, veri filtrelemesi yapmadığı bir senaryoda bulunduğunu varsayarsak bu durumda, bir saldırganın tarayıcı üzerinden web sayfasının URL'sine ve arka planda çalışan SQL ifadesine müdahale ederek kullanıcı bilgilerine erişim talepleri başlatması mümkün olabilir. Örneğin, <http://www.ornek.com/list.asp?id=1> gibi bir URL kullanılabilir. Saldırgan, URL'ye eklediği tek tırnak (') karakteriyle SQL ifadesini etkileyebilir ve veritabanına müdahale etmeye çalışabilir. Bu şekilde, saldırgan ekstra bir SQL ifadesi ekleyerek kullanıcı bilgilerini çalmaya veya veritabanını manipüle etmeye çalışabilir. Bu örnekte, saldırganın kasıtlı olarak SQL ifadelerini eklemesi, veritabanında istenmeyen sonuçlara neden olabilecek bir risk oluşturur. Bu yüzden web uygulamalarının, kullanıcı giriş verilerini sıkı bir şekilde doğrulaması, özel karakterleri filtrelemesi ve güvenlik önlemlerini etkin bir şekilde uygulaması son derece önemlidir.

2.3 SQL Enjeksiyon Saldırısı Sınıflandırması

Saldırganın amacına bağlı olarak, SQL enjeksiyon saldırıları çeşitli şekillerde kategorize edilebilir [7] :
Veri Hırsızlığı: Saldırganın amacı veritabanından kullanıcı adları, şifreler, kredi kartı bilgileri gibi verileri çalmaktır.

Veri Manipülasyonu: Saldırgan, veritabanındaki verileri değiştirmeye veya manipüle etmeye çalışır. Bu tür saldırılar sonucunda, veritabanında yanlış bilgilerin görüntülenmesi veya verilerin silinmesi gibi etkiler ortaya çıkabilir.

Yetki Yükseltme: Saldırganın hedefi SQL enjeksiyonu kullanarak kendi yetkilerini artırmak veya yönetici düzeyine erişmektir.

Veritabanı Parmak İzi (Database Fingerprinting): Saldırgan, farklı saldırı türlerine hazır olmak için veritabanının türünü ve sürümünü belirlemeye çalışır.

Sistem Erişimi: Bu tür saldırıların amacı, SQL enjeksiyonunu kullanarak veritabanı sunucusuna veya sistem kaynaklarına erişim elde etmektir. Saldırganlar bu şekilde kötü amaçlı kodları yürütebilir veya sistemde diğer saldırılar için zemin hazırlayabilir.

2.4 SQLIA Tarafından Oluşturulan Tehditler

TABLO I SQL enjeksiyon saldırılarının oluşturduğu çeşitli tehditler ve örnekler.

SQLİ TÜRLERİ	ÖRNEKLER
Spoofing	Başka bir kullanıcının kimlik bilgilerini kullanma ve diğer kullanıcılara ait verileri alma
Tampering	Veritabanındaki verileri değiştirme
Repudiation	İşlem kayıtlarını ve günlük dosyaları silme
Information disclosure	Kredi kartı bilgileri gibi verilere erişme
Denial of Service	Sunucuda ağır yüke sebep olan ve hizmeti çökerten SQL sorgularını çalıştırma
Elevation of privileges	Yönetici oturum açma kimlik bilgilerini taklit ederek yönetici ayrıcalıklarını kazanma

3.Literatür Taraması

Komal & S. Deswal (2018) [8] tarafından gerçekleştirilen araştırmada, SQL Injection, XSS (Cross-Site Scripting), CSRF (Cross-Site Request Forgery) ve tampon taşması gibi güvenlik açıklarını tespit etmek için bir hibrit dize eşleştirme algoritması olan KMPS (Knuth-Morris-Pratt Shift) önerilmiştir. Bu çalışmada, KMPS algoritması kötü niyetli kalıpları tespit etmek amacıyla kullanılmıştır. Algoritma, bir URL içerisindeki olası kötü niyetli kalıpları bulmak için Pazar Arama Algoritması'nın kaydırma adımını ve KMP Algoritması'nın eşleştirme mantığını birleştirir. Bu yöntem, web uygulamalarındaki güvenlik açıklarını tespit etmek ve saldırıları önlemek için bir araç olarak kullanılabilir. KMPS algoritması, verilen bir dizede belirli kalıpları hızlı bir şekilde tespit edebilen bir dize eşleştirme algoritması olduğu için bu amaçla tercih edilmiştir. Önerilen tekniğin Boyer Moore Algoritmasına karşı karşılaştırmalı çalışması, Tablo II'de gösterildiği gibi arama süresi, verim ve doğruluk açısından daha iyi sonuçlar göstermektedir. **Sunulan tekniğin performansı 120'den fazla URL uygulanarak incelenmiştir.** Araştıranlar, güvenlik açıklarının tamamen ortadan kaldırılamayacağını ancak web uygulaması geliştirmenin farklı aşamalarında güvenlik açıklarını tespit etmek için farklı yöntemlerin uygulanabileceğini vurgulayarak sonuca varıyorlar.

TABLO II Boyer Moore ve KMPS algoritmalarının arama süresi, verim ve doğruluk açısından sonuçlarını gösterir.

	Boyer Moore	KMPS
Seraching Time(ns)	264814477.6	127342594
Throughput	11.82879	20.84
Accuracy	70.9	72.8

TABLO III Boyer Moore ve KMPS algoritmalarının işleyiş mekanizması ve kaydırma adımları sonuçlarını gösterir.

	Boyer Moore	KMPS
İşleyiş Mekanizması	Kalıbın sağdan sola doğru eşleştirme yapmasını sağlar	Önek tablosu (prefix table) oluşturarak çalışır
Kaydırma Adımları	Kötü eşleşmelerdeki karakterlere dayalı olarak kaydırma adımları yapar	Önek tablosunu kullanarak kaydırma adımları yapar

Nency Patel ve Narendra Shekokar (2015) [9] tarafından yapılan çalışmada, SQLIA (SQL Injection Attack) tespiti ve önlenmesi için değiştirilmiş Aho-Corasik model eşleştirme algoritması, SQLMAP aracı ve AIIDA-Sql tekniği kullanarak yeni bir teknik önerilmiştir. Bu çalışmada, SQLMAP aracı, SQL Injection saldırılarını algılamak ve kullanmak için kullanılan bir araç olarak değerlendirilmiştir. Ancak SQLMAP'nin algılama süresi 15-20 dakika gibi uzun olabilmektedir. Bu dezavantajı aşmak için, yazarlar AIIDA-Sql adını verdikleri yeni bir yaklaşım geliştirmişlerdir. Bu yaklaşım, sinir ağı temelli bir sistem olan AIIDA-sql kullanmaktadır. Öncelikle, etkilenen sorgunun tespiti için sinir ağı kullanılmaktadır. Sinir ağı, SQL enjekte edilen sorguyu tespit etmek için eğitilir. Ardından, tespit edilen sorgu statik bir kalıp listesinden geçirilir ve Aho-Corasik kalıp eşleştirme algoritması kullanılarak kötü amaçlı kalıplarla karşılaştırılır. Eğer sorgu listede bulunmuyorsa, kabul edilir; aksi takdirde, reddedilir. Bu sistem, sinir ağının eğitimi sırasında hesaplama açısından ek bir yük getirdiği için bir dezavantaja sahiptir. Ayrıca, sistemin doğruluğu büyük ölçüde sinir ağının eğitime bağlı olduğundan, bu aşama oldukça önemlidir. Nency Patel ve Narendra Shekokar'ın bu çalışması, SQL Injection saldırılarını algılama ve önleme konusunda yeni bir yaklaşım sunmaktadır.

Değiştirilmiş Aho-Corasik model eşleştirme algoritması, SQLMAP aracı ve AIIDA-Sql tekniği bir araya getirilerek daha etkili ve hızlı bir tespit ve önleme süreci sağlanmaktadır. S. Ali, SK. Shahzad ve H. Javed (2009) [10] tarafından yapılan çalışmada, SQL Injection Protector for Authentication (SQLIPA) adı verilen bir yöntem sunulmuştur. Bu teknik, özellikle totoloji SQL enjeksiyon saldırılarını önlemeye odaklanmaktadır. SQLIPA, kullanıcı adı ve parola giriş alanlarının güvenliğini sağlamak için karma mekanizmasını kullanır. Her yeni kullanıcı hesabı oluşturulduğunda, kullanıcı adı ve şifre için hash değerleri üretilir. Bu hash değerleri, veritabanında kullanıcı adı ve parola sütunlarından bağımsız olarak ayrı sütunlarda saklanır. Kullanıcılar oturum açmaya çalıştıklarında, oturum açma kimlik bilgileri veritabanında saklanan hash değerlerine göre doğrulanır. Oluşturulan hash değerleri, kullanıcı adı ve parolanın saklanan hash değerleriyle karşılaştırılarak kontrol edilir. Bu sayede, bir güvenlik katmanı eklenir ve totoloji enjeksiyon saldırıları etkisiz hale getirilir. S. Ali, SK. Shahzad ve H. Javed'in bu çalışması, SQLIPA adlı yöntemi sunarak, kullanıcı kimlik doğrulama sürecinde SQL enjeksiyon saldırılarının önlenmesine odaklanmaktadır. Karma mekanizmasıyla kullanıcı bilgilerini güvenli bir şekilde saklamak ve doğrulamak, totoloji enjeksiyon saldırılarına karşı etkili bir koruma sağlar.

WGJHalfond & A.Orso (2006)[11] tarafından yapılan çalışmada, SQLIA'nın tespit edilmesi ve önlenmesi için AMNESIA adlı bir araç önerilmiştir. Bu yaklaşım, statik analiz kullanarak sıcak noktaları belirlemekte ve SQL sorgularını önceden oluşturulmuş bir modele göre kontrol etmektedir. Ayrıca, tüm meşru sorguların ve çalışma zamanı izlemenin bir SQL sorgu modelini oluşturmaktadır. AMNESIA, otomatik bir yaklaşımdır ve bir web uygulamasını girdi olarak gerektirmektedir. Ek bir çalıştırma ortamı kurmaya ihtiyaç duymadığından kullanımı kolaydır. Ampirik değerlendirmeler, AMNESIA'nın SQLIA'yı tespit etme ve önleme konusundaki etkinliğini göstermektedir. Ancak, önerilen yöntemin Java kitaplıklarına bağımlılığı dezavantaj olarak belirtilmektedir. Ayrıca, bazı durumlarda yanlış pozitif ve yanlış negatif sonuçlar üretebilme potansiyeli bulunmaktadır. WGJHalfond & A.Orso'nun bu çalışması, SQLIA tespiti ve önlenmesi için AMNESIA adlı bir araç önererek, web uygulamalarında SQL enjeksiyonu ile ilgili güvenlik açıklarının saptanması ve önlenmesine yönelik etkili bir yaklaşım sunmaktadır.

Deevi Radha Rani, B.Siva Kumar, L.Taraka Rama Rao, VTSai Jagadish, M.Pradeep (2012) [12] tarafından yapılan çalışmada, SQLIA'nın tespit edilmesi ve önlenmesi için saklı yordamlarla şifreleme kullanan bir teknik önerilmiştir. Bu yöntemde, kullanıcı tarafından girilen veriler için gizli bir anahtar orta katmanda oluşturulmakta ve şifrelenmiş veriler veritabanında saklanmaktadır. AES_ENCRYPT() fonksiyonu kullanılarak dinamik olarak karma oluşturulmaktadır. Bu uygulama, saldırganın web uygulamasına bulaşmasını engellemek için gizli anahtarın bilinmediği ve dinamik olarak üretildiği bir yaklaşım sunmaktadır. Veriler şifrelendiği için saldırganın veritabanındaki hassas bilgilere erişmesi veya manipüle etmesi zorlaşmaktadır. Deevi Radha Rani ve diğer yazarların çalışması, SQLIA tespiti ve önlenmesi için saklı yordamlarla şifreleme kullanarak güvenlik önlemlerini artıran bir yaklaşım sunmaktadır. Bu yöntem, kullanıcı verilerinin güvenliğini sağlamak ve SQL enjeksiyon saldırılarına karşı web uygulamalarını korumak için etkili bir araç olabilir.

Akhtar Rasool, Amrita Tiwari, Gunjan Singla, Nilay Khare [13] tarafından yapılan çalışmada, Brute force Algoritması, Naive dizi eşleştirme Algoritması, Knuth-Morris-Pratt Algoritması, Rabin-Karp Algoritması, Boyer-Moore Algoritması, Commentz Walter Algoritması ve Aho-Corasick Algoritması gibi çeşitli dizi eşleştirme algoritmalarının karşılaştırmalı bir analizi sunulmuştur. Performans, zaman karmaşıklığı ve mekân karmaşıklığı bu çalışma için dikkate alınan metriklerdir.

Yapılan analizlerin sonuçları aşağıdaki tabloda özetlenmiştir.

TABLO IV Çeşitli dizi eşleme algoritmalarının Zaman karmaşıklıklarını gösterir.

ALGORITHMS	TIME COMPLEXITY
Brute Force	$O(n-m+1)m$
Rabin-Karp	$\theta(m), \theta(n+m)$
Boyer-Moore	$O(m + \ \sum\), O(n)$
Knuth Morris Pratt	$O(m), O(n+m)$
Aho-Corasick	$O(m), O(m+z)$

4. Önerilen Metot

Bu bölümde, SQL Injection Attack'ı tespit etmek ve önlemek için kullanılan Aho-Corasick Pattern eşleştirme algoritmasının inceledik. Aho-Corasick algoritması iki bileşene sahiptir. Bunlar Statik ve Dinamik Aşama'dır.

a) Statik Aşama[15]:

1. Kullanıcı tarafından giriş yapılan SQL Sorgusu, belirlenen Statik Model Eşleştirme Algoritmasına aktarılır.
2. Anormallik Model Listesi, içerisinde tutulan anormallik modelleri ile birlikte eşleştirme sürecinde kullanılır.
3. Her bir model, Anormallik Model Listesi'ndeki desenlerle karşılaştırılır.
4. Eğer SQL Sorgusu, Anormallik Desen Listesi'ndeki herhangi bir desenle tam bir şekilde eşleşirse, SQL Enjeksiyon Saldırısından etkilenmiş kabul edilir.

b) Dinamik Aşama[15]:

1. Statik Fazda eşleştirme kriterlerini karşılamayan SQL Sorgusu için Anomali Skoru hesaplanır.
2. Anomali Skoru, belirlenen bir Eşik değeri ile karşılaştırılır.
3. Eğer Anomali Skoru, Eşik değerinden yüksekse, bir alarm durumu oluşur ve Sorgu Yöneticisine bildirilir.
4. Sorgu Yöneticisi, Alarm aldığı anda, ilgili SQL Sorgusunu manuel olarak analiz eder.
5. Eğer SQL Sorgusu herhangi bir enjeksiyon saldırısından etkilenmişse bir model oluşturulur ve bu model Statik Model Listesi'ne eklenir.

4.1 Anomali Puanı Değeri Hesaplaması

Statik aşamada Sorgu Yapısı, Giriş Kontrolü gibi faktörlerin belirlenen bir ağırlıklandırma ve puanlama sistemiyle Anormallik Puanı hesaplanır. Eğer sorgu Statik Kalıp Listesindeki herhangi bir şekilde %100 eşleşirse Sorgu, SQL Enjeksiyon Saldırısından etkilenmiş kabul edilir. %100 eşleşme olmazsa da yüksek uyum puanı, bir sorgunun Anomalite Puanı olarak adlandırılır. Anomalite Puanı, bir Eşik değeri (Threshold) ile karşılaştırılır[16]. Eğer Anomalite Puanı, eşik değerinden büyükse (%50 olarak kabul edelim), Sorgu Yöneticiye iletilir.

4.2 Model Eşleştirme Algoritması

Aho-Corasick algoritması, metin içinde birden çok kalıbı eşleştirmek için kullanılan etkili bir algoritmadır. Bu algoritmanın bir parçası olan Model Eşleştirme Algoritması ise Aho-Corasick algoritmasının kalıpları eşleştirmek için kullandığı yöntemdir[17].


```

1: Procedure SPMA(Query, SPL[ ])
  INPUT: Query ← User Generated Query
         SPL[ ] ← Static Pattern List with m Anomaly
         Pattern
2: For j = 1 to m do
3: If (AC (Query, String.Length(Query), SPL[j][0]) =  $\phi$ )
  then
4:    $Anomaly_{score} = \frac{Matching_{value}(Query, SPL[j])}{StringLength(SPL[j])} \times 100$ 
5:   If ( $Anomaly_{score} \geq Threshold_{Value}$ )
6:   then
7:     Return Alarm → Administrator
8:   Else
9:     Return Query → Accepted
10:  End If
11: Else
12:   Return Query → Rejected
13: End If
14: End For
End Procedure

```

Şekil 3

Model eşleştirme algoritmasının sözde kodu

C. Aho-Corasick Algoritması

Çeşitli stratejilerle, sonlu otomata dayalı kalıpları tanıma amacıyla kullanılan birçok yaklaşım mevcuttur. Aho-Corasick algoritması [18], bu konuda klasik bir algoritmadır. Buradaki temel düşünce, algoritmanın önceden hesaplama aşamasında, anahtar kelimeler kümesi kullanılarak sonlu bir otomatın oluşturulması ve eşleştirmenin, SQL sorgusundaki her karakteri tam olarak bir kez okuyan ve her okuma için sabit bir zaman harcayan otomat tarafından gerçekleştirilmesidir.

Aho - Corasick Çoklu Anahtar Kelime Eşleme Algoritması

AC algoritması, Anormallik Anahtar Kelimeler kümesini bir model eşleştirme süreci sırasında depolamak için bir deneme ayrıntısını kullanır.

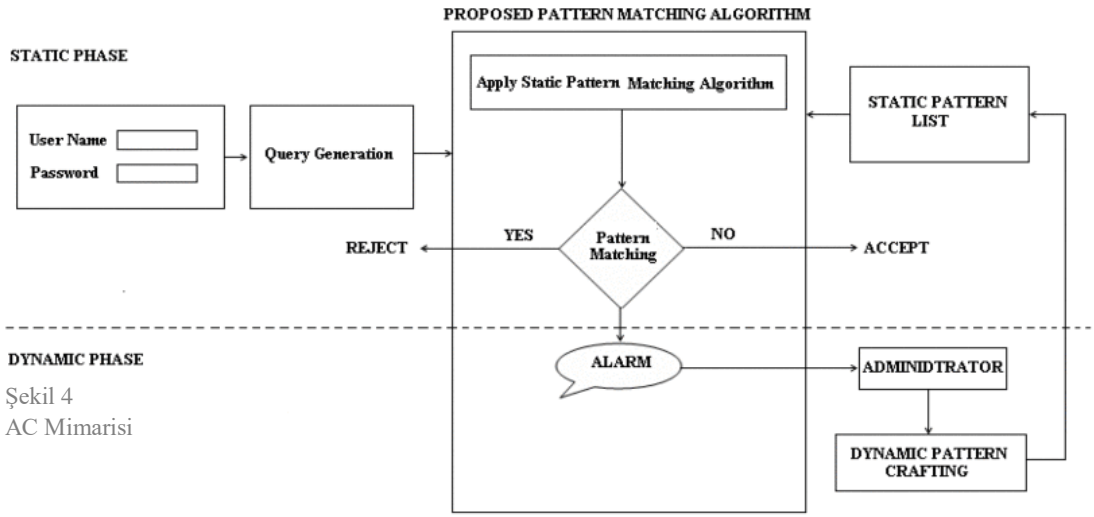
```

1: Procedure AC(y,n,q0)
  INPUT: y ← array of m bytes representing the text input
         (SQL Query Statement)
         n ← integer representing the text length
         (SQL Query Length)
         q0 ← initial state (first character in pattern)
2: State ← q0
3: For i = 1 to n do
4:   While g ( State, y[i] ) = fail do
5:     State ← f (State)
6:   End While
7:   State ← g (State, y[i])
8:   If o (State) ≠  $\phi$  then
9:     Output i
10:  Else
11:    Output  $\phi$ 
12:  End If
13: End for
14: End Procedure

```

Şekil 4

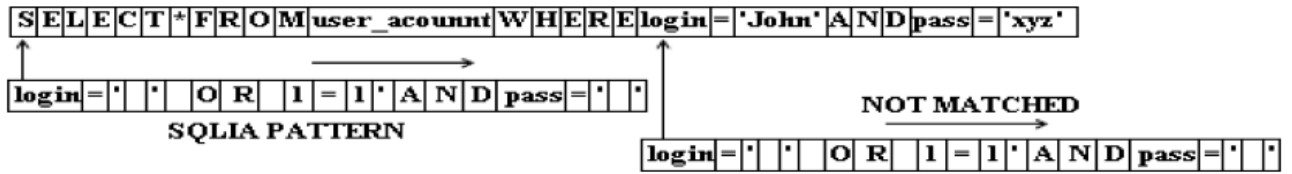
AC algoritmasının sözde kodu



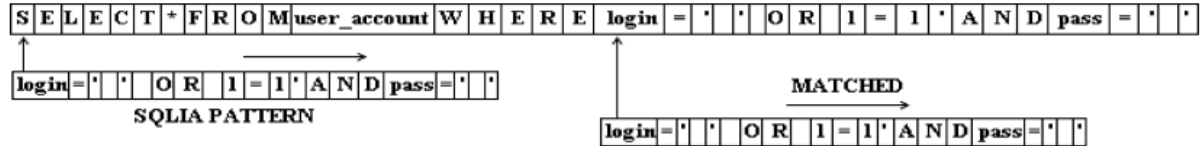
Şekil 4
AC Mimarisi

SELECT*FROMuser_acounntWHERElogin='John'ANDpass='xyz'

Şekil 5
Yasal kullanıcı adı ve parola ile SQL Sorgu Oluşturma



Şekil 6
SQLIA Model Eşleştirme Süreci



Şekil 7
SQLIA Kalıbı Tam Olarak Eşleşiyor

References

- [1] Miniwatts Marketing Group, 2023. Internet World Stats. Website: <https://www.internetworldstats.com/stats.htm> access on 20 May "World Internet Users Statics and 2023 World Population Stats.
- [2] M. Junjin, "An Approach for SQL Injection Vulnerability Detection", International Conference on Information Technology: New Generations , 2009.
- [3] S Mukherjee, P Sen, S Bora ve diğerleri, "SQL Injection: A sample review", International Conference on Computing Communication and Networking Technologies (ICCCNT) , s. 1-7, 2015.
- [4] Rua Mohamed Thiyab, Musab AM Ali, Farooq Basil ve Abdulqader, "The Impact of SQL Injection Attacks on the databases security", Proceedings of the 6th International Conference on Computing and Informatics , ICOCI 2017.
- [5] J E T Akinsola, Oludele Awodele, A. Idowu and Kuyoro Shade, "SQL Injection Attacks Predictive Analytics Using Supervised Machine Learning Techniques", International Journal of Computer Applications Technology and Research, vol. 9, pp. 139-149, 2020.
- [6] L Ma, D Zhao, Y Gao ve C Zhao, "Web Tabanlı SQL Enjeksiyon Saldırısı ve Önleme Teknolojisi Araştırması", *Uluslararası Bilgisayar Ağı Elektronik ve Otomasyonu Konferansı (ICCNEA)* , s. 176-179, 2019.
- [7] H. Dehariya, P. Kumar Shukla ve M. Ahirwar, "Sql enjeksiyon saldırıları için tespit ve önleme teknikleri üzerine bir anket", *International Journal of Wireless and Microwave Technologies*, cilt. 6, hayır. 6, s. 72-79, 2016.
- [8] S.Deswal Komal, "KMPS: Web Uygulama Güvenlik Açıklarını Tespit Eden Hibrit Bir Algoritma", International Journal of Computer Sciences and Engineering, cilt. 6, hayır. 6 Haziran 2018.
- [9] H. Dehariya, P. Kumar Shukla ve M. Ahirwar, "Sql enjeksiyon saldırıları için tespit ve önleme teknikleri üzerine bir anket", *International Journal of Wireless and Microwave Technologies*, cilt. 6, hayır. 6, s. 72-79, 2016.
- [10] Nency Patel ve Narendra Shekokar, "SQLIA'yı savunmak için model eşleştirme algoritmasının uygulanması", *Uluslararası Gelişmiş Bilgi İşlem Teknolojileri ve Uygulamaları Konferansı (ICACTA-2015) Procedia Computer Science* , cilt. 45, s. 453-459, 2015.
- [11] S. Ali, SK. Shahzad ve H. Javed, "SQLIPA: SQL Enjeksiyonuna Karşı Bir Kimlik Doğrulama Mekanizması", *European Journal of Scientific Research* , cilt. 38, hayır. 4, s. 604-611, 2009.
- [12] WGJ Halfond ve A. Orso, "AMNESIA kullanarak SQL enjeksiyon saldırılarını önleme", *28. uluslararası Yazılım mühendisliği konferansı (ICSE) Bildiriler Kitabı'nda sunuldu* , s. 795-798, 2006.
- [13] Deevi Radha Rani, B. Siva Kumar, L. Taraka Rama Rao, VT Sai Jagadish ve M. Pradeep, "Stoklanmış Prosedürlerde Şifreleme Kullanarak SQL Enjeksiyonunu Önleyerek Web. 3, Güvenliği", (IJCSIT) International Journal of Computer Science and Information Technologies , cilt . 3, hayır. 2, s. 3689-3692, 2012.
- [14] Akhtar Rasool, Amrita Tiwari, Gunjan Singla ve Nilay Khare, "String Matching Methodologies: A Comparative Analysis", *International Journal of Computer Science and Information Technologies* , cilt. 3, hayır. 2, s. 3394-3397, 2012.
- [15]-[16]-[17] M. A. Prabakar, M. KarthiKeyan and K. Marimuthu, "An efficient technique for injection attack using pattern matching algorithm," 2013 IEEE International Conference ON Emerging Trends in Computing, Communication and Nanotechnology (ICECCN), Tirunelveli, India, 2013, pp. 503-506, doi: 10.1109/ICE-CCN.2013.6528551.
- [18] CJ Ezeife, J. Dong, AK Aggarwal, "SensorWebIDS: A Web Mining Intrusion Detection System", International Journal of Web Information Systems, cilt 4, s. 97-120, 2007

