ESTUDAM Bilişim Derg, 2024; 5(1): 1-11.

J ESTUDAM Information, 2024; 5(1): 1-11.

DOI: 10.53608/estudambilisim.1402251

(Geliş Tarihi / Received Date: 08.12.2023, Kabul Tarihi/ Accepted Date: 03.01.2024)

(**Araştırma Makalesi**)

Web Uygulamalarında Enjeksiyon Saldırılarının Tespiti

Mehmet Serhan Erçin*[1], Esra Nergis Yolaçan[2]

[1]Eskişehir Osmangazi Üniversitesi, Mühendislik Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümü, 26040, Eskişehir, ORCID No : https://orcid.org/0000-0002-0921-3913

[2] Eskişehir Osmangazi Üniversitesi, Mühendislik Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümü, 26040, Eskişehir, ORCID No : https://orcid.org/0000-0002-0008-1037

Anahtar Kelimeler:
Enjeksiyon,
Makine Öğrenmesi,
SQLi,
XSS,
CMDi,
CVE

Özet: Enjeksiyon üst başlığında toplayabileceğimiz saldırılar, yıkıcı etkilerinden ve kolay uygulanabilirliklerinden dolayı saldırganlar tarafından daha çok tercih edilmekte, rastlanma sıklıkları her geçen gün artmaktadır. Günümüzde, web uygulamaları ve bağlantılı çerçeve yapıları, sıklıkla kullandığımız ve hayatımıza pek çok noktada dokunan, büyük hizmetlerdir. Bu yüzden siber saldırganların ilgisini sürekli canlı tutmakta ve yeni yöntemler keşfetmeye motive etmektedir. Sızma tespiti ve önlenmesi üzerine literatürde pek çok çalışma bulunmaktadır. Genel başlıklarda değerlendirilen bu çözümlerin, değişen ve gelişen uygulamalardan dolayı, alt başlıklarda ve ayrıntılı değerlendirilmesi ve buna uygun yeni çözümlerin bulunması gerekmektedir. Enjeksiyon tipi saldırılarda, girdilerin içerisindeki hedef sistem rezerve kelimeleri hariç tutulursa, kullanılan diğer harf ve rakamsal kombinasyonların sayısı sınırsızdır. Bu nedenle imza tabanlı sistemler yerine makine öğrenmesi yöntemlerinin genelleştirme performansı enjeksiyonların tespitinde önemli avantajlar sağlayacaktır. Bu çalışmada özellikle web enjeksiyon saldırılarına ilişkin saldırının doğru tespit edilmesinin yanı sıra, zamansal performans ve çıktıların sınıflandırılması da esas alınmaktadır. Rassal Orman ve Karar Ağacı sınıflandırıcılarında %94,54 ve %94,61 isabet oranları elde edilmiş, 15 ve 12 sn. öğrenme süreleri performansı ölçülmüştür.

(**Research Article**)

Detection of Injection Attacks in Web Applications

Keywords:
Injection,
Machine Learning,
SQLi,
XSS,
CMDi,
CVE

Abstract: The attacks that we can bring together on Injection subject whose frequency of occurence is increasing day by day are more preferable for attackers due to their destructive effects and easy applicability. Today, web applications and linked framework structures are great services that we use frequently and that touch our lives at many points. Therefore, it keeps the interest of cyber attackers alive and motivates them to discover new methods. Due to the changing and developing services and applications via emerging technologies that general topics often needs reconsidering as sub topics and new solutions should be found accordingly. Generalization is an intense need in injection type attacks unlike some other intrusion methods. Hereby, the generalization ability of machine learning methods despite signature-based systems will provide significant advantages in the detection of injections. Therefore, machine learning methods which have much more ability of making generalization rather than signature-based systems will provide significant advantages in the detection of injections. This study is focused on input-driven web attacks, differentiating benign and malicious traffic, time performance on processes and accurate classification of outputs. With Random Forest and Decision Tree classifiers, 94.54% and 94.61% hit rates were obtained, and 15 and 12 sec. learning time performance was measured.

*Sorumlu Yazar/Corresponding Author: msercin@gmail.com

1. GİRİŞ

Web sayfalarının çalışmasını sağlayan kodlamalar, uygun girdi ile istenilen sonucu verirken, diğer yandan siber saldırganlara aynı kodlamalara farklı girdiler vererek kendi amaçlarına uygun çıktılar almayı sağlayabilmektedirler. Bu amaçla yapılan farklı girdilerin tamamına enjeksiyon denilmektedir. Enjeksiyonlar için girdi olarak "yük" (payload) kavramının tercih edilmesinin sebebi, bağlantı için gerekli olan iletişim bilgileri (sunucu ve varsa istemci tarafı kimlik bilgileri) girildikten sonra "istek" (request) belirten kısma vurgu yapmaktır. Enjeksiyonları sınıflandırırken pek çok öne çıkan özellik seçilebilir; hedef aldığı yapı, yorumlayıcı, veri tabanı yönetim sistemi, işletim sistemi, girdinin yapısı bunlardan bazılarıdır. ABD merkezli Mitre kuruluşunun Ortak Saldırı Deseni Numaralandırma ve Sınıflandırma (Common Attack Pattern Enumeration and Classification – CAPEC) [1] listesine göre Kod Enjeksiyonu, Kod Ekleme, Komut Enjeksiyonu, Girdi Verisi Manipülasyonu, Parametre Enjeksiyonu, Altyapı Manipülasyonu genel başlıklarında enjeksiyon saldırılarının pek çok alt türü bulunmaktadır. Buna göre Siteler Arası Betik (XSS), LDAP, XML, Yapılandırılmış Sorgu Dili (SQL), NoSQL, Email (SMTP/IMAP başlığı), Biçim Dizesi (Format String), Yansıtma (Reflection), Argüman, Kısayol Değiştirme (Path Traversal), Siteler Arası Kaynak Hırsızlığı (CSRF), Olay Kaydı, Sunucu Tarafı Kapsama (SSI), WebView, İşletim Sistemi Komut Satırı türü enjeksiyonlar da alt başlıklar olarak listelenmiştir. Bunlardan XSS, SQL, XML enjeksiyonlarının da kullanılan alt yönteme göre pek çok başlığı listelenmiştir [2–8]. Enjeksiyon saldırısı ile saldırı yapılan uygulamanın çalışması belirli komutları yürütmek yönünde manipüle edilmiş olur. Neticesinde ise veriler çalınabilir, bozulabilir, değiştirilebilir ya da verilen hizmet tamamen durabilir.

Önlenmesi basit, ancak değiştirilip tekrar denenmesi de basit olduğundan dolayı on yıllardır yeni teknikler geliştirilmesine rağmen girdi problemi tam olarak çözülememektedir. Çözüm yaklaşımları açısından doğrulama (validation), zararsızlaştırma (sanitization), düzenleme (regulation) gibi yazılımsal tekniklerin geliştirilmesine rağmen çözümsüzlük devam etmektedir. Bu çözümsüzlük, web uygulamalarının da yaygınlaşmasıyla birlikte "Uygunsuz (improper) Girdiler" [9] olarak literatürdeki yerini korumaktadır.

Uygunsuz girdiler problemini doğru tanımlayabilmek için Ray ve Ligatti, 2012'de yazdıkları makalede [10] SQL enjeksiyonu ile ilgili bazı örnekleri vermişlerdir. Saldırganların mantığını izah edebilmek için devam eden örnekler oluşturulmuştur.

SELECT hesap FROM müşteriler WHERE şifre='99';

Üstteki SQL cümlesinin WHERE koşulundaki "şifre" sütunu dışarıdan parametre almaktadır. Eğer dışarıdan gelen parametre düzgün bir kontrolden geçirilemezse saldırgan 99 ifadesi yerine ' or 1=1 -- ifadesini girdi olarak gönderebilir ve neticede saldırgan aşağıdaki gibi bir SQL cümlesi elde etmiş olur.

SELECT hesap FROM müşteriler WHERE şifre='' or 1=1 --;

Saldırgan buradan bütün müşterilerin hesap bilgilerine ulaşmış olur. İmza tabanlı güvenlik sistemlerinde bunu saldırı olarak işaretleyip imza veritabanına bu saldırıyı kaydederseniz saldırgan bu kez 1=1 yerine 2=2 veya daha farklı sayısız "totoloji" oluşturan ifade uydurabilir. Bu durumda saldırganların deneyebileceği bütün kombinasyonları istatistiksel öğrenme modellerinden birisi ile yakalamak avantaj sağlayabilecektir.

Tablo 1'de bazı yük girdisi örnekleri, bu problemi daha ayrıntılı tanımlamak ve birlikte ele alınabileceğini göstermek için verilmiştir. Çalışmada kullanılan saldırı vektörü kavramı, ilgili web adresinin isminden sonra gelen kısma işaret eder. Bu yüzden "https://guvensiz-site.com.tr" olarak maskelenmiştir.

Tablo 1 : Zararlı yük girdisi örnekleri ve türleri

Payload	Türü
https://guvensiz-site.com.tr/ara?metin=ben-geldim	Normal
guvensiz-site.com.tr/ara?metin=<script>/*+saldirgan+bu+alanda+*/</script>	Reflected XSS
https://guvensiz-site.com.tr/ara?metin="+OR+1=1--	Totoloji SQLiA
site.com.tr/ara?metin=' UNION SELECT kullanici,sifre FROM kullanicilar--	Union SQLiA
https://guvensiz-site.com.tr/evanterDurum.php?urunID=381&stokID=29	Normal E-Ticaret
https://guvensiz-site.com.tr/evanterDurum.php?dir=%3Bcat%20/etc/passwd	Command Injection
/?search=<xss+id%3dx+onfocus%3dalert(document.cookie)+tabindex%3d1>#x	Dom-Based XSS
http://domaincontroller.local??action=dir&search=admin*)(password=_@{}/()!\"$%-=^[]:;	LDAP injection
%0aZARARLI%0aiçerik%0a%2e%0aMAIL+FROM:+<mailAdresim>%0aRCPT+TO:+<mailAdresim>%0aZARARLIiçerik%0aFrom:+<mailAdresim>%0aTo:+<mailAdresim>%0aSubject:+deneme%0apostasi%0a%2e%0a	SMTP injection *(Tamamı tek satır)*
http://guvensiz-site.com.tr/envanter/resim.php?file=../../../../etc/passwd	Path Traversal

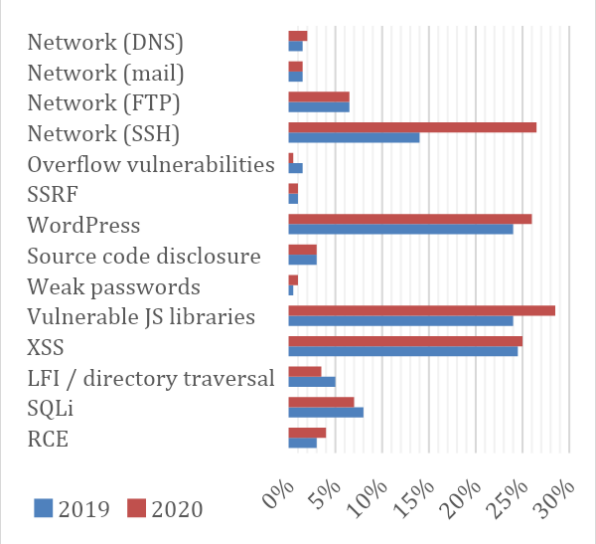
Mantıksal seviyede zararlı kod ile zararsız kodun birbirinden farkı yoktur, bu durum, zararlı kodun, makine öğrenmesi harici geleneksel tespit sistemleriyle tespitini zorlaştırmaktadır. Yorumlayıcıya (interpreter) göre, girdi alanına gelen her şey uygundur ve hatalı veya hatasız bir şekilde işlenebilir [11]. Sızma Tespit/Engelleme Sistemi (IDS/IPS) veya birlikte çalışmak üzere Web Uygulama Güvenlik Duvarı (WAF) adı verilen özelleşmiş yazılımlar, diğer pek çok bileşeni ile, enjeksiyonlara karşı güvenliği de sağlamaktadır. Sızma Tespit veya Güvenlik Duvarı yazılımları genel eğilim olarak her bir saldırıya ait yük girdilerini imza veritabanında tutar [12] ve bu saldırılara karşın tanımlanmış kurallar üzerinden tam eşleşme sağlandığında tespit ve engelleme işlemini gerçekleştirir [13]. Saldırganlar ise saldırı girdisinde saldırının özünü etkilemeyecek küçük bir değişiklikle bu imzaları atlatabilmektedir. Saldırı tespit sistemlerinde

makine öğrenmesi/derin öğrenme yaklaşımlarının kullanımı 1990 yılında bilgisayar virüslerini tespit etme ile başlarken [14] 1998'den itibaren çok çeşitli saldırı türleri için birçok modelin çalışıldığı görülmektedir [15]. Yapay öğrenme uygulamalarında, saldırı yükleri doğru bir biçimde girdi olarak verilir ve doğru modelleme yapılırsa, test aşamasında saldırı olup olmadığını ve hatta saldırının türünü dahi tespit edecek şekilde çalışabilir.

Tablo 2 : CWE numarası ile bağlantılı CVE adedi

No	CWE Tanımı	Zafiyet A.
79	Failure to Preserve Web Page Structure ('Cross-site Scripting')	17106
119	Failure to Constrain Operations within the Bounds of a Memory Buffer	11994
20	Improper Input Validation	8875
200	Information Exposure	7520
89	Improper Sanitization of Special Elements used in an SQL Command ('SQL Injection')	6849
787	Out-of-bounds Write	4283
22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	4013
125	Out-of-bounds Read	3625
94	Failure to Control Generation of Code ('Code Injection')	2736
287	Improper Authentication	2508
416	Use After Free	2322
269	Improper Privilege Management	2124
78	Improper Sanitization of Special Elements used in an OS Command ('OS Command Injection')	1729
476	NULL Pointer Dereference	1552
190	Integer Overflow or Wraparound	1514
400	Uncontrolled Resource Consumption ('Resource Exhaustion')	1017
120	Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')	982
284	Access Control (Authorization) Issues	980
434	Unrestricted Upload of File with Dangerous Type	947
362	Race Condition	895
732	Incorrect Permission Assignment for Critical Resource	795
59	Improper Link Resolution Before File Access ('Link Following')	734
77	Improper Sanitization of Special Elements used in a Command ('Command Injection')	728
74	Failure to Sanitize Data into a Different Plane ('Injection')	713
798	Use of Hard-coded Credentials	680

ABD merkezli Mitre organizasyonunun Siber Güvenlik ARGE merkezi tarafından belirlenen Ortak Zayıflık Numaralandırması (Common Weakness Enumeration – CWE) listesinde [2] pek çok başlıkta kendine yer bulan enjeksiyon tipi saldırılar, Ortak Güvenlik Açıkları ve Zafiyetler (Common Vulnerabilities and Exposures – CVE) listesinde sıklık bakımından da en üstte yer almaktadır. Bu durumu göstermek amacıyla Mitre Organizasyonu CWE çalışma grubunun En Üst-25 listesi Tablo 2'de verilmiştir. Tablonun ilk sütunu CWE numarasını, son sütunu ise barındırdığı CVE adedini gösterir. CWE-79, CWE-20, CWE-89, CWE-94, CWE-78, CWE-77, CWE-74 zayıflıkları ise OWASP vakfının "En üst 10 Web Uygulama Güvenlik Riski 2021" listesinin 3. sırasındaki A03-Injection [3] başlığı içerisinde toplanmıştır. CAPEC listesinde "ID 152: Inject Unexpected Items" kategorisinde 9 ana başlıkla ele alınmıştır. Web Uygulama Güvenlik Konsorsiyumu (The Web Application Security Consortium – WASC) Tehdit Sınıflandırması 2.0 raporunda da listenin yarıya yakını enjeksiyonlardan oluşmaktadır. Etki bakımından da pek çok veri sızıntısında başrolü oynayan bu saldırılar, her geçen gün siber saldırganların ilgisini daha çok çekmekte ve saldırılar üstel olarak artmaktadır. Şekil 1'de bir güvenlik firmasının ilgili CWE'lerin kritiklik derecesi en yüksek zafiyetlerine göre yapmış olduğu 2019 ve 2020 yıllarına dair karşılaştırmalı bir analiz bulunmaktadır.



Şekil 1. Acunetix 2021 Web Uygulamaları Güvenlik Rap. [16]

Tablo 2'de bağlantılı CVE görülme adedine göre sıralı CWE adları ve numaraları görülebilmektedir. Buna göre CWE-79 numaralı XSS zayıflığının temsil ettiği zafiyet sayısı, CWE-89 numaralı SQL injection zayıflığının temsil ettiği zafiyet sayısının neredeyse 3 katıdır. Tabloda 25 adet CWE listelenmiştir.

Literatürde web uygulamalarına yapılan saldırıların tespiti için birçok çalışma yapıldığı görülmektedir. SQL ve XSS'i bir arada değerlendirerek tespit etmeyi hedefleyen CODDLE [17] isimli çalışmada Abaimov ve Bianchi, girdilerdeki sembolleri tip/değer çiftlerine dönüştürmüşler ve derin öğrenme modeline girerek %95 oranında isabet elde etmişlerdir. Benzer bir çalışmayı SQLiGoT [36] isimli çalışmada Kar ve diğerleri de yalnızca SQL enjeksiyonları için yapmış ve elde ettikleri Token'ları SVM'e girdi olarak vermişlerdir.

Kavitha ve diğerleri [18] K-Ortalama Kümeleme Algoritmasını kullanan denetimsiz makine öğrenimi tekniğini benimsemişlerdir. Önerilen sistemin akışına bakılacak olursa: Son kullanıcının web uygulamasında sorgulaması ve sorgu değerlerinin çıkarılıp iki katmanlı güvenlik sağlayan Sql Injection Detector'a gönderilmesi olarak özetlenebilir. Güvenliğin ilk katmanında, düşük seviyeli saldırılar için bağlamdan bağımsız dilbilgisi (Cfg) kullanılarak kalıplar oluşturulur. Üst düzey saldırılar için ikinci güvenlik katmanı kullanılarak eğitilir.

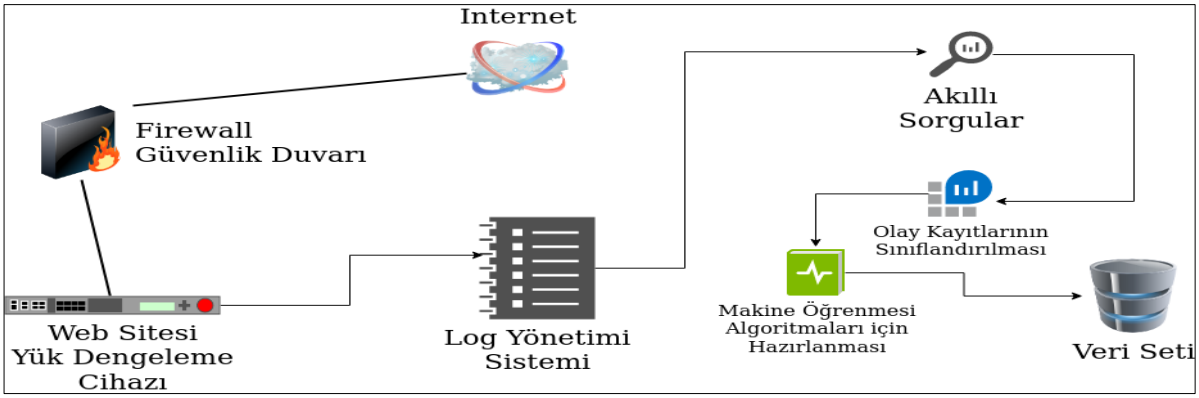
Tablo 3 : Literatürdeki yayınların ön işleme, yöntem ve amaçları

Yayın Adı	*Veri Seti*	*Ön İşleme*	*Yöntem*	*Amaç*
CODDLE [17]	Github (xss, sql payl.)	Tip / Değer Çiftleri	Derin Öğrenme	SQLi ve XSS Tespiti
Kavitha v.d. [18]	Özel: Web Server	Dilbilgisi Kalıpları (Cfg)	Denetimsiz KNN ve Statik Analiz	SQLi Tespiti
Hoang v.d.[19]	HTTP Params D.	Web Günlüğü	DT, SVM	Web Saldırıları
Yao Pan v.d. [20]	1000 Safe, 500 Attack	RSMT	Yarı Denetimli ML	Web Saldırıları
ZEROWALL [21]	1,4 Milyar Web İsteği	Tokenize (Encode/Decode)	Denetimsiz RNN	Sıfırıncı Gün Saldırıları
Gniewkowski, Mateusz v.d. [22]	CSIC2010, CIDS2018, UMP	Doğal Dil İşl. RoBERTa	Logistic Regression	HTTP isteği Anomali Tespiti
Alma, Das [23]	ECML-KDD, VULNBANK	Auto Encoder	Özel LSTM	Web Saldırıları
TbD-NNbR [24]	1204 Norm. 451 Zararlı	Tokenize	Sinir Ağı	SQLi
Betarte v.d. [25]	DARPA'99, CSIC2010, PKDD2007	Dil İşleme	Tek-Sınıf Sınıfl. ve N-Gram	Modsecurity Geliştirme
RPAD [26]	NIST:2783 Attack,512 N	Dil İşleme	Kural Tabanlı Desen Tanıma	Web Saldırıları
Ibarra-Fiallos, Santiago [27]	36000 Norm. 25000 Zararlı	RegEx, Dil İşleme	Statik Filtreleme	Girdi Doğrul., XML, JSON vb
Gogoi v.d. [28]	XSStrike, XSSer	Tokenizasyon, TF-IDF,Sampl.	Çeşitli ML algoritmaları	XSS
Nagarjun vd[29]	XSStrike,XSSer	Vektörize	Ensemble Learning	XSS
Mereani,Howe [30]	XSSed	JS Terimleri	SVM, KNN, RF	XSS
Vishnu,Jevitha [31]	XSSed	JS Terimleri	NB,SVM,J48DT	XSS
Hasan v.d. [32]	616 SQL st.	SQL Terimleri	23 ML Classif.	SQLi
Mishra [33]	Xristica, libinjection	RegEx, G-test Skoru, Entropi	NB, Gradient Boosting	SQLi
Ross [34]	Özel: Web App Host	Korelasyon için Tokenize	Decision Tree,	SQLi
TCSVM [35]	SMOTE ile	Hash fonk. ile Özellik çık.	Two-Class SVM	SQLi
Sheykhkanloo [36]	12500 Nrm. 12500 Zar.	32 tane elle seçilmiş Özellik	Yapay Sinir Ağı (%70,%15,%10)	SQLi
SQLiGoT [37]	Çok Çeşitli İnternet Ky.	Sorgu Tokenizasyon	LibSVM (tek, 2 ve 3 Kanallı)	SQLi
OFDPI [38]	SDN üzerinden	TF-IDF	Logistic Regression	Host/Net. based IDS

Yao Pan v.d. makalelerinde [20], otonom yapıdaki saldırı tespit sistemleri konseptine önemli katkılar sağlamaktadır. Web uygulamalarının çalışma zamanı davranışını otonom olarak izleyen ve karakterize eden Robust Software Modeling Tool'a (RSMT) dayalı web saldırısı tespiti için denetimsiz/yarı denetimli bir yaklaşımın fizibilitesi üzerinde çalışılmıştır. Önerilen yaklaşıma göre çıkan sonuçlarda, SQL enjeksiyonu, siteler arası komut dosyası çalıştırma ve seri kaldırma dahil olmak üzere saldırıları, minimum etki alanı bilgisi ve çok az etiketlenmiş eğitim verisi ile verimli ve doğru bir şekilde tespit edilebildiği gösterilmektedir.

Ruming Tang ve diğerleri, Zerowall ismini verdikleri çalışmalarında, 1,4 milyar Web isteği üzerinde uyguladıkları yöntemle, mevcut WAF'lar tarafından kaçırılan gerçek sıfırıncı gün saldırılarını başarılı bir şekilde saptadıklarını ve 0,98'in üzerinde yüksek F1 puanları elde ettiklerini gösteriyorlar [21].

Gniewkowski, Mateusz ve diğerleri, HTTP isteklerini doğal dil işleme tekniklerinden RoBERTa ile vektörize ederek Logistic Regression ile denetimsiz öğrenmeye tabi tutmuşlardır [22]. Veri seti olarak CSIC2010, CSE-CIC-IDS2018 ve kendi hazırladıkları girdileri kullanmışlardır. F1 skorları ortalama veri setleri için %95 civarındadır. Tikam Alma ve Lal Das, 2020'de yine HTTP isteklerini girdi olarak kullanarak Auto-Encoder'la girdilerdeki kelimelerin ağırlıklarını öğrenen ve bunu LSTM hücrelerinden oluşturdukları bir modele girdi olarak veren bir çözüm [23] geliştirerek %99,79 başarım elde etmişlerdir. Teresa George ve diğerleri, geliştirdikleri



Şekil 2. Veri seti elde etme mimarisi

TbD-NNbR [24] çerçeve (Framework) vekil (Proxy) sunucusunda gelen istekleri Token'lara çevirip test ettikten sonra zararlı ise bloklama, zararsız ise yeniden inşa ederek veritabanı uygulamasına yönlendirmektedirler. Her ne kadar bu yayında sadece SQL injection ile çalışılmışsa da yaklaşım bütün enjeksiyonlara uyarlanabilecek niteliktedir. 1204 normal, 451 zararlı veri ile çalışılmış, tespit oranı %99.75 olarak gerçekleşmiştir. Gustavo Betarte ve diğerleri, Modsecurity WAF'ının (Web Application Firewall) performansını arttırmak amacıyla yaptıkları çalışmada [25] makine öğrenmesi olarak Tek-Sınıf Sınıflandırma ve N-Gram metodunu ayrı ayrı uygulamışlar ve Modsecurity'nin performansını %20 civarında arttırmışlardır.

Makine öğrenmesi dışında bütün HTTP isteklerinden zararlıları tespit ve engelleme niyetiyle RPAD'i [26] yayınlayan Venkatramulu, kural tabanlı desen tanıma yaklaşımında %92,8 başarım elde etmiştir. Kendisinden önce yapılan IPAAS çalışmasının %80 isabet oranıyla da performans kıyaslaması yapmıştır. Ibarra-Fiallos, Santiago ve diğerleri, web sayfalarına girdi yapılan alanlara bir ön filtre tasarlamayı düşünmüşlerdir. Bu bağlamda filtre her bir alan için, harf, rakam, nokta, soru işareti ve ünlem işareti gibi işaretleri kontrol ederken JSON ve XML dosyalarını da doğrulatmaktadır [27]. %98,4 isabet, 50 ms civarı bir performans yakalanmıştır. XSS tipi saldırıları tanımak maksadıyla Gogoi ve diğerleri [28], Nagarjun ve Ahamad [29], Mereani ve Howe [30], Vishnu ve Jevitha [31]; SQL enjeksiyonlarını tanımak maksadıyla Hasan ve diğerleri [32], Sonali Mishra [33], Kevin Ross [34], Uwagbole ve Buchanan [35] ve Sheykhkanloo [36] makine öğrenmesi sınıflandırıcıları ve derin öğrenme sinir ağlarını kullanmışlardır.

Cheng, Qiumei ve diğerleri, OFDPI ismini verdikleri çalışmalarında [38] network seviyesinde çalışan girdi makine öğrenmesi temelli girdi tanımlama sistemi geliştirmişlerdir. Buna göre paketler network seviyesinde aynalama (mirror) yöntemi ile OFDPI modelinin çalıştığı sisteme gelir. Burada şifresiz paketlerdeki girdiler okunarak TF-IDF'e önceliklendirilir ve dilsel özellikleri çıkartılır. Bu çıktılar da Logistic Regression ikili sınıflandırıcısına girdi olarak verilir. Şifreli paketlerin ise zararlı ve zararsız olanlarına göre özellikleri çıkartılarak Karar Ağacına girdi olarak verilir. İşlemci, ram, throughput, işlem zamanı gibi performansların da verildiği çalışmada %98 ve %99 başarım elde edilmiştir.

İncelenen yayınlar Tablo 3'te amaçlarına ve ikincil ölçüt olarak kullandıkları yöntemin özgünlüğüne göre sıralı olacak şekilde verilmiştir. Buna göre yalnızca SQL veya yalnızca XSS inceleyen yayınlar bulunduğu gibi bütün web saldırılarını önlemeyi amaçlayan, sıfırıncı gün saldırılarını hedef edinen, WAF performansı arttırmak isteyen yayınların bulunduğu görülmüştür. Bu çalışmada amaçlanan, girdi kalıplarının bir bütün olarak değerlendirilip hepsinin tespitinin çoklu sınıflandırma çıktısıyla verilmesi daha önce yapılmamıştır.

Çeşitli kâr amacı gütmeyen kuruluşlar, vakıflar ve güvenlik şirketlerinin raporlarında tasnif edilmiş saldırı tipleri ve etiketlenmiş girdi parçalarından (payload) hareketle makine öğrenmesi sınıflandırıcı veya derin öğrenme modeli oluşturulmuştur, tespit etme ve sınıflandırma etme performansına ek olarak ne kadar vakit harcadığı da ölçülerek not edilmiştir. Literatürde bazı çalışmaların yalnızca Saldırı/Saldırı değil olarak sınıflandırma yaptığı, birçoğunun yalnızca bir ana başlık üzerinde girdileri modellemeye çalıştığı (SQL, XSS veya Komut Enjeksiyonu gibi), pek çoğunda harcanan zamana ilişkin performans metriklerinin hepsinin ölçülmediği görülmüştür. Bu çalışmada, literatürdeki eksikliklerin giderilmesi amacıyla belirlenen konularda iyileştirmeler yapılarak katkı sunulmaktadır. Buna göre;

- Enjeksiyon saldırısı tiplerinden bilinen ve uygulaması mümkün olanların modellemeye girdi olarak verilerek geniş bir veri seti kullanılması,
- Eğitim ve test aşamaları için zamana ilişkin performansın ölçülmesi,
- Girdi olan yük parçasının hangi saldırı tipine ait olduğuna dair tahminde bulunulması,

katkıları sağlanmaktadır.

Makalenin bundan sonraki bölümlerde devam eden yapısı şu şekildedir: Bölüm 2'de çalışmanın gerçekleştirilmesi için gerekli materyal ve kullanılan metot açıklanmıştır. Buna göre veri setleri, ön işlem ve özellik çıkartma teknikleri, hazır makine öğrenmesi sınıflandırıcıları ve kurgulanan veya kullanılan yapay sinir ağı modelleri, kullanılan platform ve yazılım kütüphaneleri ve analiz araçları ayrıntıları verilmiştir. Bölüm 3'te elde edilen bulgular ve bulgulara ilişkin değerlendirmeler açıklanmaktadır. Bölüm 4'te ise çalışmanın öne çıkan yönleri, zayıflıkları ve gelecek çalışmalara ilişkin tartışmalar ve sonuçta ulaşılan yorumlara yer verilmiştir.

2. MATERYAL VE METOT

Bu çalışma kapsamında kullanılan/oluşturulan veri seti, gerçekleştirilen ön işlemler, uygulanan algoritmalar ve değerlendirme metrikleri bu bölümde detaylandırılmıştır.

2.1. Veri Setleri

Enjeksiyon başlığıyla zararlı girdiler etiketlenmiş olması gerektiğinden dolayı hazır olarak internet kaynaklarından elde edilen veri setleri, yerel bir alan adına yapılan girdilerle karşılaştırılmış ve etiketleme fonksiyonu hazırlanarak etiketleme yapılmıştır. Bu amaçla, bir alan adına (Domain) ve alt alan adlarına gelen HTTP istekleri kullanılmıştır. Veri seti güvenlik gerekçeleriyle alan adı ve alan adına işaret eden alt isimler maskelenmiş ve anonimleştirilmiştir. Buna göre 2307 zararlı, 2149 normal istek kullanılmıştır. Zararlı istekler 16 sınıfa bölünmüştür. Bunlar Piggy Backed, Blind SQL, XSS Keyword, Stored Procedure, Alternate Encoding, Union, Totoloji, SQL Error, SQLi, Enjeksiyon, SSTI, Open Redirect, XXE, Path Traversal, CMDi, CRLFi isimlerine karşılık gelecek şekilde numaralandırılmıştır. Aynı zamanda, OWASP tarafından sağlanmış GitHub'da bulunan swisskyrepo, PayloadsAllTheThings girdileri [7] ve GitHub'da bulunan XSS ve SQL injection veri setleri [39] karşılaştırma ve etiketleme için kullanılmıştır.

Anonimleştirilmiş veri setinin elde edilme adımları Şekil 2'de verilmiştir. Buna göre Internet bulutundan gelen istekler öncelikle Güvenlik Duvarı (Firewall) yazılımında karşılanır. Buradan "Zararsız" olarak geçen istekler Web Sitesi Yük Dengeleme Cihazı'na gelir. Burada çalışan Syslog yazılımı verileri hem dosyaya depolar, hem de Log Yönetimi Sistemi'ne gönderir. Log Yönetim Sistemi'nde veriler Elasticsearch veritabanında tutulur, gösterimi Kibana aracılığıyla yapılır.

Bu çalışmada veriler hem Yük Dengeleme Cihazı'nın üretmiş olduğu Syslog çıktıları, hem de Log Yönetimi Sistemi'nin verileri alınmıştır. Akıllı Sorgular Syslog çıktıları ve Elasticsearch Veritabanı'nda yapılmış, sonuçlar harmalanmıştır.

2.2. Ön İşleme ve Özellik Çıkartma

Veri seti hazırlandıktan sonra ön işlem fazında ilgili veri tabanı sorgu dili, programlama dili, protokol ve işletim sistemi rezerve kelimelerinin veri setindeki ilgili değerlerle eşleştirilmesi, kavramların bir metodoloji güdülerek tekrar kodlanması, saldırılar için önemsiz olan değerlerin veri setinden çıkartılarak bu boş kalan alanların işaretlenmesi işlemleri yapılmıştır.

Veri tabanı yönetim sistemleri, programlama dili yorumlayıcıları, işletim sistemleri, protokoller, servisler, sorgu dilleri farklı yöntemlerle girdi almaktadır. Genel olarak bu çalışmada XSS, SQL injection, Command Injection, SMTP, LDAP, Path Traversal, NoSQL, GraphQL, XML, gibi girdi ile çalışan yapıların özellikleri belirlenerek özellik çıkartımı buna göre yapılmıştır.

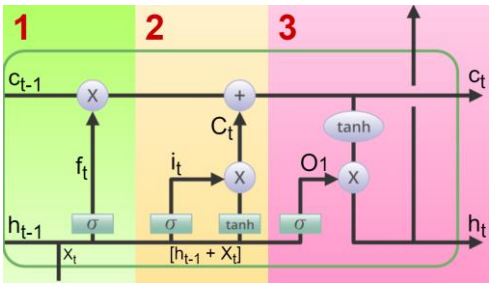
Girdilerde zararlı parçanın hedef aldığı yapının rezerve kelimeleri (MySql, JavaScript, Bash, PHP, Microsoft Transact-SQL v.d. gibi) tespit edilerek RZR ile numaralandırılmıştır. Buna göre 1846 adet rezerve kelime elde edilmiştir. Operatörler (<, ', ", >, |, &, (, [, --,),], ` v.d. gibi) RegEx kullanılarak tespit edilip OPR ile numaralandırılmıştır. Geri kalan kısımda mutasyona sebebiyet veren kısımlar değişken kabul edilip uygun bir isimlendirmeyle genelleştirilmiştir.

Mahalanobis uzaklığı hesaplanmış, kavramların birbirine yakınlığı ölçülüp sınıflandırıcılardaki performansına bakılmış, girdilerin uzunluğu belirli bir ölçekte sabitlenerek yapay sinir ağlarının girdileri tanıması sağlanmıştır.

2.3. Makine Öğrenmesi ve Yapay Sinir Ağları

Sinir ağlarının ve makine öğrenmesi algoritmalarının etkili sınıflandırma yapabilmeleri için veri setindeki kavramların sayısallaştırılması ve mahalanobis uzaklıklarının hesaplanması yapılmıştır. Hiperparametreler ağın veya sınıflandırıcının eğilimleri, problemin özü ve veri setinin yapısı dikkate alınarak ayarlanmıştır.

Bu çalışmada, LSTM (Long-Short Term Memory), Multi-Layer Perceptron, Evrişimsel Sinir Ağları ve Lineer SVC (Support Vector Classifier), K-Nearest Neighbors, Gaussian Naive Bayes, Decision Tree, Random Forest, Ridge, Stochastic Gradient Descent (SGD), Passive Aggressive, Gradient Boosting, Extra Trees algoritmaları ile tespit ve hız performansı ölçülmüştür. Girdilerin tamamı etiketli olduğundan ve özelliklerin seçilmesinden dolayı denetimli öğrenme yapılmıştır. Daha yüksek başarım için hiper parametrelerin optimizasyonu üzerine çalışmalar yapılmıştır.



Şekil 3. Uzun-Kısa Dönem Hafıza (LSTM) mimarisi

Çalışmanın ön işlem fazından sonraki aşamalarının Doğal Dil İşleme (NLP) problemine dönüşmesi sebebiyle NLP'lerde etkili olan tekniklerin burada da etkili olabileceği değerlendirilmiştir. Şekil 3'te mimarisi verilen LSTM'in literatürde [23] başka çalışmalarda da tercih edildiği görülmektedir. LSTM'in Unutma (1), Durum (2) ve Çıktı (3) katmanları bir cümlenin öğelerini en iyi şekilde algılar ve büyük bir metnin içerisindeki en önemli öğeleri en öne çıkartır.

Karar ağaçları ve genel olarak ağaç ve orman yaklaşımları da cümlenin içerisindeki kavramlar arasında ilişkisellikler kurarak büyük metnin içerisinde hangi kavramın hangisiyle daha fazla ilişkili olduğunu ortaya koyabilmektedir. Literatürde incelenen bazı çalışmalarda bu tekniklerin başarılı bir şekilde kullanıldığı görülmüştür [19, 30, 34].

Literatürde daha önceden yapılmış çalışmalar da dikkate alınarak, sinir ağlarının çok hücreli yapısının bu problemi çözmede faydalı olabileceği değerlendirilmiştir ve bu ağlar kullanılmıştır. Karar ağacı, lineer ve topluluk öğrenmesi sınıflandırıcılarının parametreler arası ilişki kurabileceği ve genelleştirme yapmadaki performansları göz önüne alınarak bu modeller kullanılmıştır. Tahmin temelli modellerin (Gaussian Naive Bayes, Lineer Regression v.d.) hem zaman, hem de tespit oranları bakımından performanslarının bu problemi çözmede uygun olmadığı görülmüştür.

2.4. Kullanılan Platform ve Araçlar

Linux Mint işletim sistemi üzerinde ve Quad core Intel Core i7-4790 (-MT-MCP-) speed/max~1225/4000 MHz Kernel~4.15.0-76-generic x86_64 15948.6MB Ram donanımıyla çalışan bilgisayarda Python 3.x derleyicisi, Keras API'leri kullanılarak gerçekleştirilmiştir.

Python 3.6.9 üzerinde Tensorflow = 2.2, Keras = 2.2, GenSim 3.2 kullanılarak yapay sinir ağları ve sınıflandırıcıların kütüphaneleri, doğal dil işleme ve uzaklık fonksiyonları kullanılmıştır. Python'da ayrıca

Pandas, Numpy, Requests gibi kütüphaneler dizilerin tutulması, matematiksel işlemlerin yapılması ve HTTP isteklerinin alınması veya gönderilmesi için kullanılmıştır.

Python'da Django ve Flask framework'leri (Web gösterimi ve canlı olarak saldırıları test edebilmek amacıyla geliştirilmiş API için), platform bağımsız masaüstü uygulaması için de Kivy framework'ü kullanılmıştır.

2.5. Analiz Yöntemi

Çalışma kapsamında ele alınan yüklerden saldırı olarak etiketlenenler "pozitif", saldırı olmayan yükler ise "negatif" olarak isimlendirilmektedir. Yük tipinin doğru tespit edilmesine "T(True)", yanlış tespit edilmesine ise "F(False)" olarak belirtilmektedir. Trafik tipi ve tespit sonuçları ile ilgili ifadeler Tablo 4'te sunulmaktadır. Bu çalışmada gerçekleştirilen testlerin değerlendirilmesi için Tablo 5'te sunulan performans metrikleri kullanılmaktadır.

Tablo 4 : Trafik tipinin sınıflandırılması mantıksal gösterimi

Trafik Tipi	Pozitif Tahmin	Negatif Tahmin
Saldırı (Pozitif)	Saldırıyı saldırı olarak tahmin (TP)	Saldırıyı normal olarak tahmin (FN)
Normal (Negatif)	Normali saldırı olarak tahmin (FP)	Normali normal olarak tahmin (TN)

Tablo 5 : Performans metrikleri

Metrik	Hesaplama
Doğruluk (Accuracy)	$\frac{TP+TN}{TP+TN+FN+FP}$
Hassasiyet (Precision)	$\frac{TP}{TP+FP}$
Özgüllük (Recall)	$\frac{TP}{TP+FN}$
F1-Skoru (F1-Score)	$2\frac{Hassasiyet+Özgüllük}{Hassasiyet*Özgüllük}$

Çalışmada kullanılan veriler saldırı tiplerine göre de etiketli olduğundan, çoklu sınıflandırma performansı da değerlendirilmektedir. Doğruluk değeri, modelin çok etiketli veri setindeki bütün girdilere dair isabetli tahmin etme oranını verir. Hassasiyet değeri, ilgili sınıfa dair isabetli değerlerin modelin o sınıfa ait olduğunu tahmin ettiği tüm örneklerine oranını verirken, özgüllük değeri modelin o sınıftaki tüm örnekler arasında doğru şekilde sınıflandırdığı örneklerin oranıdır. F1-Skoru ise aslında modelin bir sınıfa dair girdileri diğerlerinden ne oranda başarı ile ayırt edebildiğini gösteren hassasiyet ve özgüllük değerlerinin harmonik ortalamasını verir.

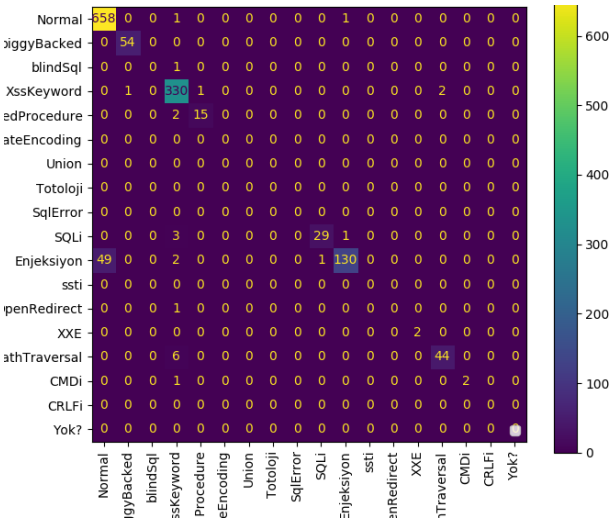
3. BULGULAR VE DEĞERLENDİRMELER

Karar Ağacı ve Rassal Orman sınıflandırıcılarının ileri düzey performans gösterdikleri görülmüştür. Sonrasında lineer sınıflandırıcılar ve hemen hemen denk olarak topluluk öğrenmesi sınıflandırıcılarının iyi performans gösterdikleri not edilmiştir. Tamamında hiper parametre optimizasyonları yapılmış, özellikle tahmine dayalı sınıflandırıcılarda doğru tespit oranının %50'lere düştüğü, işlem süresinin de 10 dakikanın üzerine çıktığı gözlemlenmiştir.

Yapay Sinir Ağları'nda parametreler dikkatle ayarlanmalı ve kaç devir boyunca eğitim sürecinin işletileceği doğru belirlenmelidir. Ayrıca veri seti yeterli miktarda olmadığı için yapay sinir ağlarında isabet oranları yakalansa da eğitim süreçlerinin uzunluğu göz önüne alınmalıdır.

	precision	recall	f1-score	support
CMDi	0.00	0.00	0.00	1
Enjeksiyon	0.81	0.75	0.78	79
Normal	0.94	1.00	0.97	302
PathTraversal	0.93	0.93	0.93	30
SQLi	0.60	0.35	0.44	17
XXE	1.00	0.50	0.67	2
XssKeyword	0.96	0.94	0.95	199
piggyBacked	1.00	1.00	1.00	26
storedProcedure	0.93	1.00	0.96	13
accuracy			0.93	669
macro avg	0.80	0.72	0.75	669
weighted avg	0.92	0.93	0.93	669

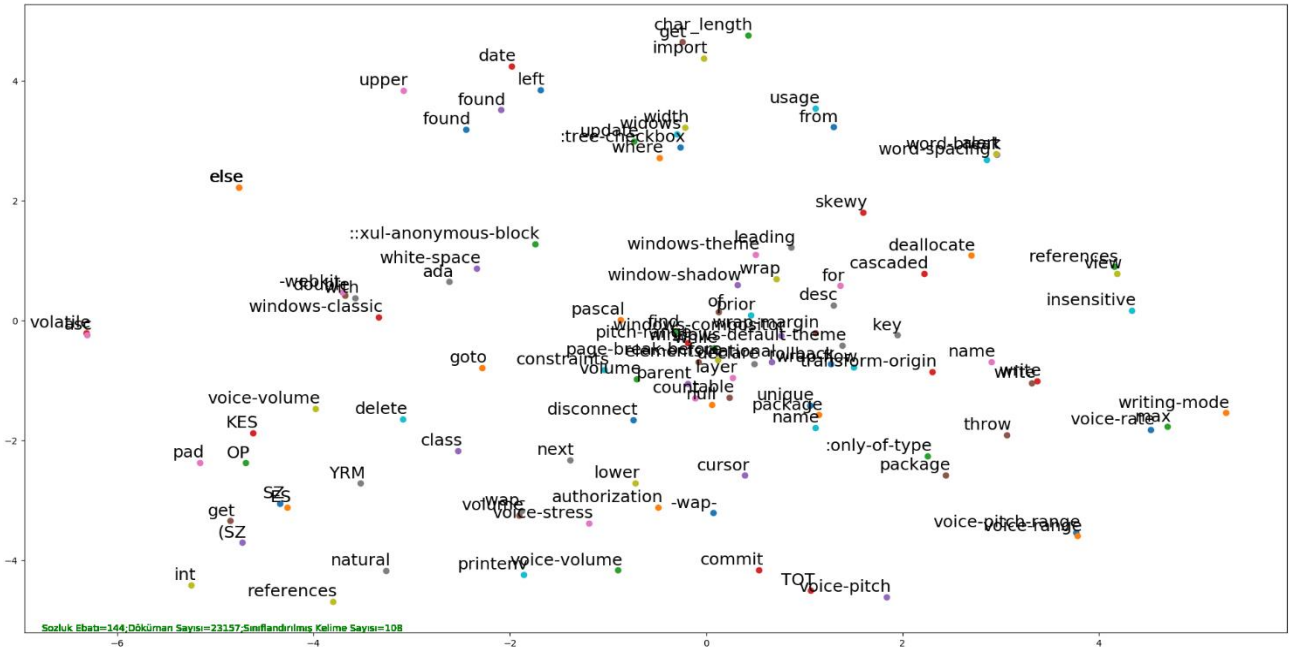
Şekil 4. LSTM Sinr Ağı çalışma performansı



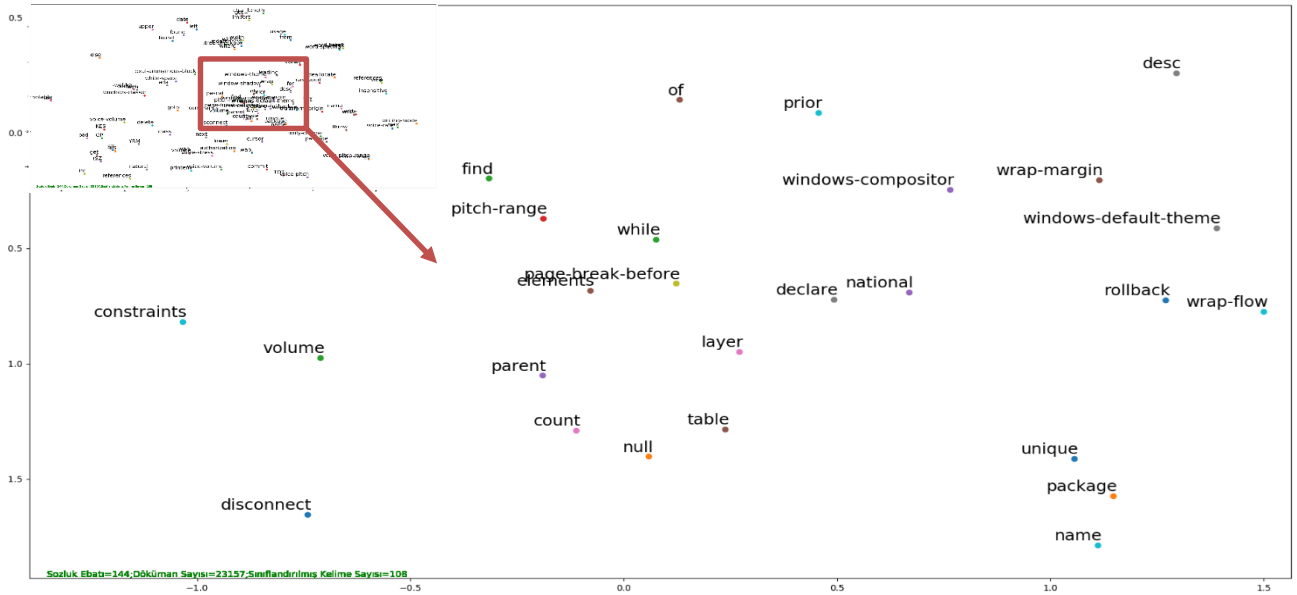
Şekil 5. Rassal Orman (Random Forest) Sınıflandırıcı modeli karışım matrisi

Tablo 6 : Çalıştırılan modeller ve performansları

Sınıflandırıcı	Doğruluk	Öğrenme Süresi (sn)
Decision Tree	0.946	12.45
Random Forest	0.945	15.047
K-Nearest Neig.	0.905	7.88
Lineer SVC	0.933	24.97
Ridge Class.	0.888	17.167
Passive Aggressive	0.890	34.77
Gradient Boosting	0.874	486.02
Extra Trees	0.863	49.75
LSTM	0.930	~1800
MLP	0.865	300



Şekil 6. Kelimelerin veri setinde rastlanma sıklıkları ve yakınlıkları



Şekil 7. Kelimelerin veri setinde rastlanma sıklıkları ve yakınlıkları (Şekil 6 – Orta bölüm)

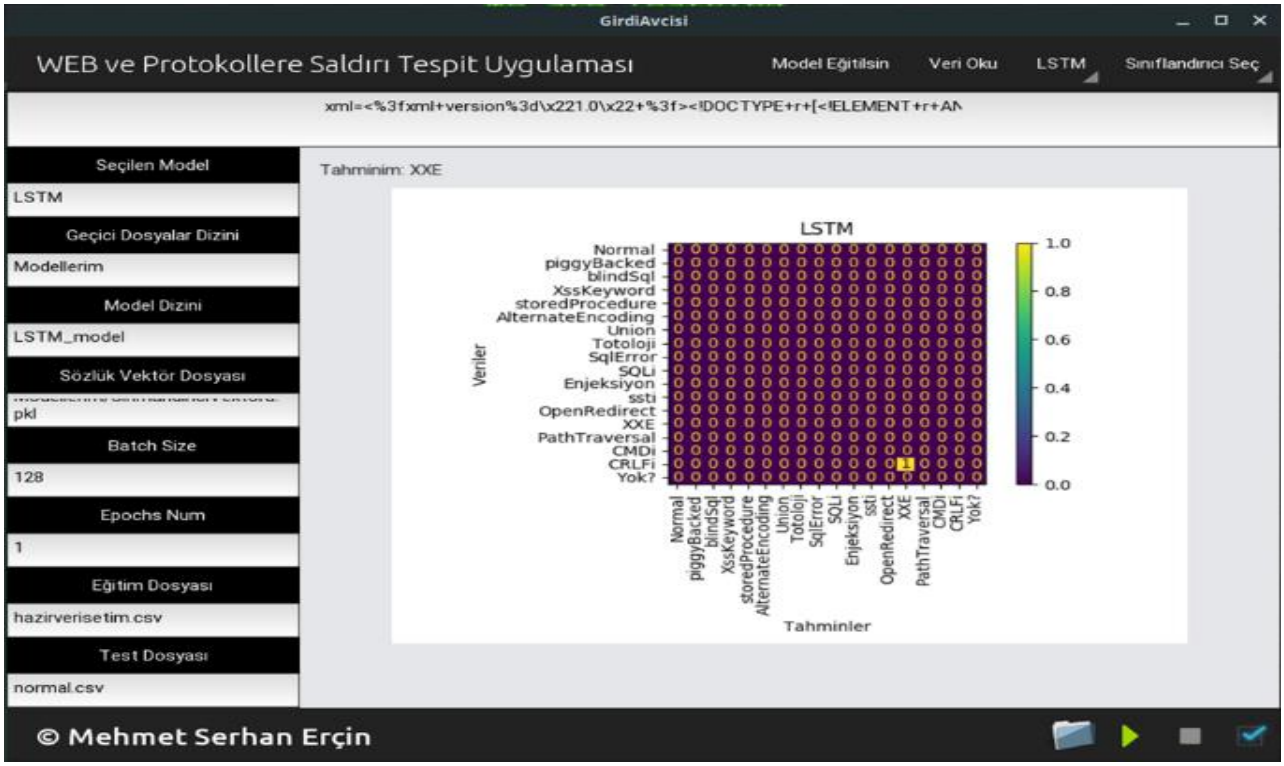
Uygulanan ön işlem adımlarının ve etiketlendirme faaliyetlerinin isabetli olduğu bazı sınıflandırıcıların eğitim süreçlerinin kısalığından çıkartılabilir. Tablo 6'dan çalıştırılan modellerin performansları izlenebilir. Şekil 4'te LSTM sinir ağının performansı karışım matrisinde gösterilmiştir. Şekil 5'de Rassal Orman Sınıflandırıcısı'nın y ekseninde verinin ne olduğu, x ekseninde tahminler olmak üzere performansı gösterilmiştir. Veri setinin dengeli olmamasının sonuçları etkilediği değerlendirilmektedir.

Şekil 6'da veri setinde görülen kavramların birbirlerine yakınlıkları görülebilmektedir. Örneğin, XSS keyword'lerinin birbirine yakın düşmesi beklenir. Modelin sınıflara dair kavramları bu şekilde birbirine yakınsatması sınıfları birbirinden ne kadar başarılı ayırdığını gösterirken gelecekte karşılaşabileceği ve daha önce hiç tanımadığı girdileri doğru sınıflandıracağına dair fikir vermektedir. Şekil 7, Şekil 6'nın ortasındaki üst üste gelmiş kelimelerin daha detaylı görülebilmesi için eklenmiştir.

Bu çalışma kapsamında geliştirilen uygulama Şekil 8'de görülmektedir. Uygulama ekranında, istenilen test verisi girilebilmekte, web tarayıcı üzerinden gelen istekler sınıflandırılabilmekte ve modeller çeşitli parametrelerle eğitilebilmektedir.

4. TARTIŞMA VE SONUÇ

Bu çalışmada, web saldırılarına ilişkin bütün saldırı tiplerinin sınıflandırmasının doğru bir şekilde yapılmasının yanı sıra zamansal performansın ölçülmesi esas alınmıştır. Bu amaçla, dış dünya verilerinden elde edilmiş özgün bir veri seti kullanılmıştır. Veri setinin üzerinde çoklu sınıflandırma etiket çalışması yapılmıştır.



Şekil 8. Masaüstü Uygulaması, örnek bir saldırı yükü sonucu.

Elde edilen bu etiketli veri seti üzerinde özgün bir ön işlem fazı gerçekleştirilmiştir. Ön işlem fazından sonra makine öğrenmesi sınıflandırıcıları ve yapay sinir ağları kullanılarak çoklu sınıf sınıflandırma işlemi gerçekleştirilmiştir. Çalışmanın pratikte kullanılabilmesi için platform bağımsız bir uygulama geliştirilmiştir

Rassal Orman ve Karar Ağacı sınıflandırıcılarında %94,54 ve %94,61 isabet oranları elde edilmiş, 15 ve 12 sn. öğrenme süreleri performansı ölçülmüştür. LSTM ağının Normal (Saldırı olmayan) veriyi diğer etiketli verilerden ayırma performansına bakılırsa %97'lik F1 Skoru görülebilir. Sistemin normal tipteki veriyi saldırı tipindeki veriden neredeyse hatasız ayırabildiği elde edilen sonuçlardan görülebilmektedir.

Yapısı gereği enjeksiyon tipindeki saldırılar bir girdi verisinin içerisinde birden fazla bulunabilmektedir. Söz gelimi bir yük girdisi, XSS saldırısı içerisinde SQL enjeksiyonu ve hatta aynı anda CMD enjeksiyonunu da birlikte içerebilir. XSS'in "Script" etiketi içerisinde POST parametreleri arasında "Select" ifadesi gönderilebilirken, Windows işletim sistemine dair komutları da görme ihtimalimiz vardır. Gerçek dünyada karşılaşılabilecek bu tip örnekler çalışmanın bir diğer zorlayıcı tarafıdır, çünkü sınıflandırma performansını olumsuz etkilemektedir.

Geliştirilen uygulamanın Web Application Firewall - WAF veya Security Information and Event Management - SIEM gibi sistemler için doğrulama ünitesi gibi çalışabileceği değerlendirilmektedir. Girdi temelli saldırıların günümüz dünyasında halen en üst düzey güvenlik tehditlerinden olduğu, veri sızıntılarının engellenemediği değerlendirilirse güvenlik bileşenlerinin çeşitliliğinin arttırılması ve yapay zeka ve makine öğrenmesiyle desteklenmesinin kritik olduğu görülebilir. İlerleyen çalışmalarda normal tipteki veriye karşılık saldırı bloğundaki bütün sınıflandırmaların birleştirilerek ikili sınıflandırma sonuçlarına bakılabileceği değerlendirilmiştir.

Etik Hususlar

Etik kurallara uyum

Bu araştırmanın planlanmasından uygulanmasına, verilerin toplanmasından verinin analizine kadar olan tüm süreçte "Yükseköğretim Kurumları Bilimsel Araştırma ve Yayın Etiği Yönergesi" kapsamında uyulması belirtilen tüm kurallara uyulmuştur. Yönergenin ikinci bölümü olan "Bilimsel Araştırma ve Yayın Etiğine Aykırı Eylemler" başlığı altında belirtilen eylemlerden hiçbiri gerçekleştirilmemiştir. Çalışmanın yazım sürecinde bilimsel etik ve alıntı kurallarına uyulmuş, toplanan veriler üzerinde herhangi bir tahrifat yapılmamış ve bu çalışma herhangi başka bir akademik yayın ortamına değerlendirme için gönderilmemiştir.

Finansman

Yazarlar kamu, ticari veya kâr amacı gütmeyen sektörlerdeki fon kuruluşlarından özel bir hibe alınmadığını beyan ederler.

Çıkar çatışması

Çalışma ile ilgili herhangi bir kişi veya kurumla çıkar çatışmasının bulunmadığını yazarlar olarak onaylıyoruz.

KAYNAKÇA

[1] Mitre, CAPEC. 2023. CAPEC VIEW: Domains of Attack. capec.mitre.org/data/definitions/3000.html (Erişim Tarihi: 04.12.2023)

[2] Mitre, CWE. 2021. CWE Top 25 Most Dangerous Software Weaknesses cwe.mitre.org/top25/archive /2021/2021_cwe_top25.html (Erişim Tarihi: 04.12.2023)

[3] OWASP. 2021. OWASP Top Ten. owasp.org/www-project-top-ten/# (Erişim Tarihi: 04.12.2023)

[4] OWASP. 2021. A03:2021 – Injection owasp.org/Top10/A03_2021-Injection/ (Erişim Tarihi: 04.12.2023)

[5] OWASP. 2021. Injection Flaws owasp.org/www-community/Injection_Flaws (Erişim Tarihi: 04.12.2023)

[6] Milzarek, R. 2020. Injection Attacks Types and How to Best Protect Your Web Apps crashtest-security.com/what-are-the-different-types-of-injection-attacks/ (Erişim Tarihi: 04.12.2023)

[7] GitHub. 2023. Payloads All The Things github.com/swisskyrepo/PayloadsAllTheThings (Erişim Tarihi: 04.12.2023)

[8] OWASP. 2023. Attacks. owasp.org/www-community/attacks/ (Erişim Tarihi: 04.12.2023)

[9] The Web Application Security Consortium. 2010. Tehdit Sınıflandırması 2.0, WASC-20: Improper Input Handling. projects.webappsec.org/w/page/13246933/Improper%20Input%20Handling (Erişim Tarihi: 04.12.2023)

[10] Ray, D., Ligatti, J. 2012.Defining code-injection attacks. Acm Sigplan Notices, 47(1), 179-190.

[11] Asif, M., Chirchi, E. M. 2021. Implementation of ML Algorithm's for Cyber Security.

[12] Valenza, A., Demetrio, L., Costa, G., Lagorio, G. 2020. WAF-A-MoLE: An adversarial tool for assessing ML-based WAFs. SoftwareX, 11, 100367.

[13] Inamdar, D. M., Gupta, S. 2020. A Survey on Web Application Security.International Journal of Scientific Research in Computer Science, Engineering and Information Technology, (6), 223-228.

[14] Fox, K., Henning, R., Reed, J., Simonian R. 1990. A neural network approach towards intrusion detection. Proceeding of 13th National Computer Security Conference, Baltimore, MD, pp. 125–134, 1990.

[15] Abaimov, S., Bianchi, G. 2021. A survey on the application of deep learning for code injection detection. Array, 11, 100077.

[16] Acunetix. 2021. Web Uygulama Zafiyet Raporu. https://www.acunetix.com/white-papers/acunetix-web-application-vulnerability-report-2021/ (Erişim Tarihi: 04.12.2023)

[17] Abaimov, S., Bianchi, G. 2019. CODDLE: Code-injection detection with deep learning. IEEE Access, 7, 128617-128627.

[18] Kavitha, M. N., Vennila, V., Padmapriya, G., Kannan, A. R. 2021. Prevention of SQL injection attack using unsupervised machine learning approach. vol, 12, 12.

[19] Hoang, X. D. 2021. Detecting common web attacks based on machine learning using web log. In Advances in Engineering Research and Application: Proceedings of the International Conference on Engineering Research and Applications, ICERA 2020 (pp. 311-318). Springer International Publishing.

[20] Pan, Y., Sun, F., Teng, Z., White, J., Schmidt, D. C., Staples, J., Krause, L. 2019. Detecting web attacks with end-to-end deep learning. Journal of Internet Services and Applications, 10(1), 1-22.

[21] Tang, R., Yang, Z., Li, Z., Meng, W., Wang, H., Li, Q., Liu, Y. 2020, July. Zerowall: Detecting zero-day web attacks through encoder-decoder recurrent neural networks. In IEEE INFOCOM 2020-IEEE Conference on Computer Communications (pp. 2479-2488). IEEE.

[22] Gniewkowski, M., Maciejewski, H., Surmacz, T. R., Walentynowicz, W. 2021. HTTP2vec: Embedding of HTTP requests for detection of anomalous traffic. arXiv preprint arXiv:2108.01763.

[23] Alma, T., Das, M. L. 2020. Web Application Attack Detection using Deep Learning. arXiv preprint arXiv:2011.03181.

[24] George, T. K., Jacob, K. P., James, R. K. 2018. Token based detection and neural network based reconstruction framework against code injection vulnerabilities. Journal of Information Security and Applications, 41, 75-91.

[25] Betarte, G., Giménez, E., Martínez, R., Pardo, Á. 2018. Machine learning-assisted virtual patching of web applications. arXiv preprint arXiv:1803.05529.

[26] Venkatramulu, S., Guru, R. 2017. RPAD: Rule based pattern discovery for input type validation vulnerabilities detection & prevention of HTTP requests. International Journal of Applied Engineering Research, 12(24), 14033-14039.

[27] Ibarra-Fiallos, S., Higuera, J. B., Intriago-Pazmiño, M., Higuera, J. R. B., Montalvo, J. A. S., Cubo, J. 2021. Effective filter for common injection attacks in online web applications. IEEE Access, 9, 10378-10391.

[28] Gogoi, B., Ahmed, T., Saikia, H. K. 2021. Detection of XSS attacks in web applications: A machine learning approach. International Journal of

Innovative Research in Computer Science & Technology (IJIRCST) ISSN, 2347-5552.

[29] Nagarjun, P. M. D., Shaik, S. A. 2020. Ensemble methods to detect XSS attacks. International Journal of Advanced Computer Science and Applications, 11(5).

[30] Mereani, F. A., Howe, J. M. 2018, January. Detecting cross-site scripting attacks using machine learning. In International conference on advanced machine learning technologies and applications (pp. 200-210). Cham: Springer International Publishing.

[31] Vishnu, B. A., Jevitha, K. P. 2014, October. Prediction of cross-site scripting attack using machine learning algorithms. In Proceedings of the 2014 International Conference on Interdisciplinary Advances in Applied Computing (pp. 1-5).

[32] Hasan, M., Balbahaith, Z., Tarique, M. 2019, November. Detection of SQL injection attacks: a machine learning approach. In 2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA) (pp. 1-6). IEEE.

[33] Mishra, S. 2019. SQL injection detection using machine learning.

[34] Ross, K. 2018. SQL injection detection using machine learning techniques and multiple data sources.

[35] Uwagbole, S. O., Buchanan, W. J., Fan, L. 2017, May. Applied machine learning predictive analytics to SQL injection attack detection and prevention. In 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM) (pp. 1087-1090). IEEE.

[36] Sheykhkanloo, N. M. 2020. A learning-based neural network model for the detection and classification of SQL injection attacks. In Deep Learning and Neural Networks: Concepts, Methodologies, Tools, and Applications (pp. 450-475). IGI Global.

[37] Kar, D., Panigrahi, S., Sundararajan, S. 2016. SQLiGoT: Detecting SQL injection attacks using graph of tokens and SVM. Computers & Security, 60, 206-225.

[38] Cheng, Q., Wu, C., Zhou, H., Kong, D., Zhang, D., Xing, J., Ruan, W. 2021. Machine learning based malicious payload identification in software-defined networking. Journal of Network and Computer Applications, 192, 103186.

[39] Taşdelen, İ. 2023. Payload Box – Attack Payloads github.com/payloadbox/
(Erişim Tarihi: 04.12.2023)