Web Uygulamaları için Derin Öğrenme Yöntemiyle Güvenlik Duvarı Uygulaması

Firewall Application with the Deep Learning Method for the Web Applications

Şengül Bayrak[1], Behzat Ardanuç[2]

[1]Yazılım Mühendisliği Bölümü
İstanbul Sabahattin Zaim Üniversitesi
bayraksengul@ieee.org

[2]Bilgisayar Mühendisliği Bölümü
İstanbul Sabahattin Zaim Üniversitesi
ardanuc.behzat@std.izu.edu.tr

Özet

Günümüzde artan internet kullanımı ile beraber web uygulamalarına karşı yapılan saldırılar da aynı şekilde artmaktadır ve modern Web Uygulama Güvenlik Duvarları (Web Application Firewall-WAF) bu saldırılara karşı yetersiz kalmaktadır. Bu çalışmada WAF sistemlerine alternatif olarak yapay zeka yöntemleri ile web sitelerine gelen saldırılar algılanıp önlenmeye çalışılması hedeflenmiştir. Çalışmada HTTP DATASET CSIC 2010 veri seti için 2-gram yöntemi uygulanarak özellik çıkarımı yapılmıştır. N-gram yöntemi ile elde edilen veri seti derin öğrenme yöntemlerinden Uzun Kısa Süreli Bellek (Long Short-Term Memory-LSTM) yöntemi ile modellenmiştir. İki farklı model geliştirilmiştir ve en başarılı modelleme başarımı %86 olarak elde edilmiştir. Böylece web uygulamalarında geçerli ve geçersiz saldırı tespiti yapılabilmiştir.

Abstract

Today, with the increasing internet usage, attacks against web applications are also increasing and modern Web Application Firewalls (WAF) are insufficient against these attacks. This study, it is aimed to detect and prevent attacks on websites with artificial intelligence methods as an alternative to WAF systems. In the study, feature extraction has been done by applying the 2-gram method for HTTP DATASET CSIC 2010 dataset. The data set obtained by the n-gram method is modeled with the Long Short-Term Memory (LSTM) method, which is one of the deep learning methods. Two different models have been developed, and the most successful modeling success was 86%. Thus, valid, and invalid attack detection can be made in web applications.

1. Giriş

Modern güvenlik duvarları önceden verilmiş bilgiler dışında bir saldırıya maruz kaldığı zaman reaksiyon alamamakta ve üzerine kurulu olduğu sistemi yeni saldırı çeşitlerinden koruyamamaktadır. Bu çalışmada geliştirilen sistem, sunucuya gelen http isteklerini analiz edip, bu isteklerin içeriğini, Tekdüzen Kaynak Tanımlayıcı (Uniform Resource Locator – URL) bilgilerini ve gövde bilgilerini inleceyip gelen isteklerin güvenli veya güvensiz olduğunun çıkarımını yapmaktadır.

Bu güvenliğin sağlanması için tasarlanan sistem gelen isteklerdeki Hiper-Metin Transfer Protokolü (Hyper-Text Transfer Protocol-HTTP) paketlerinin URL bilgilerini inceleyip daha önce n-gramlar yardımı ile elde edilmiş önemli anahtar kelimeleri URL içerisinde aramaktadır, dahası GÖNDER (POST) ve AL (PUT) isteklerinin gövde içeriğini inceleyerek şüpheli bir trafik olup olmadığını incelemektedir. Bu hedeflere ulaşırken HTTP DATASET CSIC 2010 veri seti kullanılarak n-gram analizi yapılmıştır. Daha sonra elde edilen n-gramlar ilgili yöntemler kelime vektörlerine dönüştürülmüş ve asıl özellik veri seti çıkarılmıştır. Elde edilen veri seti LSTM yöntemi ile modellenmiştir.

2001 yılında web uygulama güvenliği zafiyetlerinin artması ile beraber Open Web Application Security Project (OWASP) başlatılmıştır [1]. Bu çalışma kapsamında web uygulama güvenliğine yönelik en zararlı ve en sık görülen açıklar listelenip açık kaynak olarak her yıl paylaşılmaktadır. Bununla beraber OWASP 10 adı altında web uygulama güvenliği zafiyetlerinin en önemli 10 tanesi listelenmeye başlanmıştır [2]. Literatür incelendiğinde Karacan ve Sevri'nin yaptığı çalışmaya kadar bu alanda yapılan çalışmaların kısıtlı saldırı çeşitlerine yönelik olduğu görülmektedir [3]. Hassan vd. [4] yalnızca Yapısal Sorgulama Dili (Structured Query Language - SQL Injection) saldırıları için çalışmalar yapmışlardır. Fang vd. [5] çalışmalarında Siteler arası komut dosyası çalıştırma saldırısı (Cross-site scripting-XSS) saldırısı için tek bir saldırı tipine yoğunlaşmışlardır. Ancak Sevri ve Karacan [6]'ın ele aldığı çalışma ile OWASP 10 içerisindeki çeşitli saldırı tipleri analiz

edilmiş ve ayrı ayrı tespit edilerek saldırı tiplerine göre sınıflandırılmıştır. HTTP isteklerinin içerisindeki bilgilerin analizi için n-gram analiz yöntemi kullanan Torrano-Gimenez vd. [7], HTTP paketlerinin içerisindeki korelasyonun çıkarılabilmesi için 5 gram harf analizi yapmış ve 5'li harflerin birliktelik tekrarlarını incelemişlerdir. Vartouni vd. [8] n-gram analizinden sonra elde edilen veri seti çok büyük olduğu için Temel Bileşen Analizi (TBA) yöntemi ile öz nitelik indirgemesi yapmışlardır. Aynı zamanda bu çalışma içerisinde HTTP içerisinden önemli bilgiler ayrıştırılmaya çalışılmıştır. Liang vd. [9] Tekrarlayan Yapay Sinir Ağı (Recurrent Neural Network-RNN) ve LSTM yöntemlerini elde ettikleri veri setinde kullanarak TBA özellik seçimi işleminden özgürleşmiş ve verilerin sınıflandırmasını bu yöntemler ile sağlamıştır. Tekerek [10] derin öğrenme yöntemlerini kullanarak bir web uygulamasında anomali tabanlı bir web saldırısı algılama mimarisi önermiştir. Mimari yapı, veri ön işleme ve Evrişimli Sinir Ağı (Cellular Neural Network-CNN) adımlarından oluşur. Önerilen CNN mimarisinin uygunluğunu ve başarısını kanıtlamak için CSIC2010v2 veri kümeleri kullanılmıştır. Çalışmada CNN derin öğrenme mimarisi ile %97 oranında başarılı sonuçlar sunmuştur. Zuek vd.[11] sınıf dengesizliğini, siber güvenlik ve makine öğrenimi için önemli bir durum olarak vurgulamışlardır. Sınıf dengesizliğini göz önünde bulundurarak son CSE-CIC-IDS2018 veri setinden web saldırılarını tespit etmede sınıflandırma performansını topluluk öğrenme (ensemble learning) yöntemleri ile çalışmışlardır. Rastgele alt örneklemenin ile Alıcı Çalışma Karakteristiği Eğrisinin Altında Kalan Alan (Area Under the Receiver Operating Characteristic Curve-AUC) açısından istatistiksel olarak anlamlı bir şekilde performansı iyileştirdiğini değerlendirmişlerdir.

Bu çalışmada da çeşitli saldırı tipleri ele alınıp, istekler içerisinde algılanıp şüpheli istekler ayırt edilecektir. Bunun için, 2-gram yöntemi ile veri setindeki her bir kelimenin birbiri ile olan korelasyonu incelenip, bütün veri seti içerisindeki frekansı hesaplanmıştır. Çalışma sonunda bu proje web uygulama güvenlik duvarlarının performansının ve güvenirliğinin arttırılmasında, WAF sistemlerinin kurulu olduğu sistemlerin güvenliğini arttırmada önemli katkılar sağlaması amaçlanmaktadır. Bu çalışma kapsamında problemin çözümüne ulaşmak üzere gerçekleştirilen 3 adım vardır. (i) CSIC 2010 HTTP veri setinin istenilen formata getirilebilmesi için veri ön işleme aşaması, (ii) işlenmiş olan veri içerisinde geçen ikili kelimelerin frekans analizini yapmak üzere n-gram kullanılması (iii) modelleme işlemi ise önceki adımlar sonucunda elde edilen veri setinin yapay zeka algoritmalarından LSTM yöntemine verilip bir model oluşturulmasıdır. Daha sonra bu modelin başarımı test edilip doğruluğu ölçülmüştür.

Bu çalışmada kullanılan malzeme ve yöntem ikinci bölümde, deneysel sonuçlar üçüncü bölümde, gelecek hedefleri ve sonuç kısmı dördüncü bölümde verilmiştir.

2. Materyal ve Yöntem

CSIC, web uygulaması sızma testi paketleri de dahil olmak üzere HTTP/1.1 paketlerinin orijinal veri kümesini oluşturmaktadır ve normal ve anormal olma üzere iki sınıflıdır [12]. Bir web sayfasında herhangi bir yere tıkanıldığı zaman karşıdaki sunucuya çeşitli parametreler ile istekler gönderilmektedir. Kullanıcının çeşitli bilgilerini saklayıp sunucuya gönderen ve sunucuya kullanıcının sunucudan beklediği bilgileri taşıyan bu paketlere HTTP paketleri, diğer bir ismiyle HTTP istekleri denilmektedir. Bu makale kapsamında HTTP DATASET CSIC 2010 veri setindeki HTTP paketleri veri seti için seçilmiştir. Veri temizleme aşamasında, veri setinin içerisindeki HTTP istekleri makale kapsamında önemli görülmeyen çeşitli parametre ve bilgiyi de taşımaktadır. Gerekli görülmeyen satırlar veri setinden silinmiştir. Son olarak elde edilen güncel veri seti elde edilmiştir. Veri temizleme ile elde edilen bu çıktının içerisindeki formatı düzenlemek için ve etkin bir şekilde modelleyebilmek için boş satırların hepsi veri setinden temizlenmiştir. Bu gruplamadan sonra veri setinin içerisindeki, n-gram analizini bozabilecek bütün işaretler boşluk ile değiştirilmiş ve elde edilen son veri seti makalede ön işlem uygulanmış veri setidir.

2.1. Modelleme

Yapay Sinir Ağı (YSA) yöntemi, ağdan hesaplanan hata değerinin gradyanlarını hesaplayarak ağırlıkları güncellemektedir. Geri yayılımlı bir YSA ağı çıkış katmanından giriş katmanına kadar basit bir sinir ağında ağırlıkların güncellenmesinde sorun yaşanmayabilir, ancak derin bir sinir ağında gradyan kaybı veya gradyan patlaması ismi verilen sorunlarla karşılaşılaşılabilmektedir. LSTM, uzun ve kısa olarak her türlü girdi bilgisini hafızasında saklayabilmektedir. Bunu yaparken de kendi içerisindeki kapı mimarisinden faydalanmaktadır. Bu mimaride 3 adet kapı bulunmaktadır. Bunlar unutma kapısı (forget gate), giriş kapısı (input gate) ve çıkış kapısı (output gate) olarak sıralı bir şekildedir [13, 14]. LSTM bir girdisini önceki girdinin çıktısından diğer bir girdisini ise o anki girdiden almaktadır. LSTM yönteminin ilk kapısı olan unutma kapısı bir önceki girdinin LSTM çıktısını alarak, bu bilginin tutulup tutulmaması gerektiğine karar vermektedir. Karar verme mekanizması aşağıdaki adımlarla çalışmaktadır [15, 16].

C.önceki×f.şimdiki=0 …if f.şimdiki=0 (her şeyi unut)

C.önceki×f.şimdili=C.önceki …if f.şimdiki=1 (hiç bir şeyi unuma)

Giriş kapısında ise o anki veriler alınıp hatırlanması için işlendiği kapıdır. Bu kısımda:

I.şimdiki=lambda×(X.şimdiki×U.i+H.önceki×Wi)

hesaplaması kullanılmaktadır. Burada X.şimdiki şimdiki girdi, U.i inputun ağırlık matrisi, H.önceki bir önceki zaman damgasındaki gizli durum ve W.i ise gizli durumun ağırlık matrisinin bilgisidir. Ve son olarak çıkış kapısı tüm bu hesaplamaların ardından önceki iki hesaplamalara benzer hesaplar yaparak tahmin çıkarımları yapmaktadır.

3. Deneysel Sonuçlar

3.1. Veri ön işleme adımı ile, CSIC HTTP veri setinde önceden kurulmuş sunucuya gelen 60 bin adet veri bulunmaktadır. Her bir veri sunucuya gelen http isteklerini barındırmakta ve bu http isteklerinin başlık kısmında kullanılan metot, URL, Kullanıcı Temsilcisi (User Agent), Pragma, Önbellek Kontrolü (Cache

Control), Kabul (Accept), Kabul Kodlama (Accept Encoding), Kabul Karakter Seti (Accept-Charset), Kabul Dili (Accept-Language), Host, Çerez (Cookie), Bağlantı (Connection) ve İçerik Uzunluğu (Content Length) bilgileri; kullanılan gövde kısmında ise Content bulunmaktadır. Bu kısımların her biri ayrı ayrı manipüle edilerek ayrı saldırı yolları uygulanabilse de bu çalışmada ele aldığımız saldırıların tespiti için yalnızca kullanılan metot, URL bilgisi, content length ve content içeriği önem taşımaktadır. Bu sebepten ötürü öncelikle çalışma için önemsiz olan bilgiler veri seti içerisinden silinmiştir. Daha sonra temizlenen veri setinin içerisindeki boş kısımlar ve boşluk tekrarları kaldırılmıştır. Her bir http isteğinin bir arada ele alınabilmesi için her bir POST, GET ve PUT isteği ayrı ayrı satırlara alınarak ilgili bilgileri aynı satırlarda toplanmıştır. Veri setindeki bütün harfler küçültülüp, n-gram analizinin sağlıklı olabilmesi için veri setindeki bazı semboller boşluk ile değiştirilmiştir. Her sembolün veri setinden kaldırılmamasındaki amaç bazı saldırılarda kullanılan sembollerin de gözden kaçmasını engellemektir. Çizelge 1'de verilen 2-gram verileri özellik vektörünün oluşturulmasında kullanılacaktır.

Çizelge 1: Elde edilen 2-gram özellik setinin ilk 5 verisi

Id	2-grams	Tekrar sayısı
1	http localhost	61065
2	localhost 8080	61065
3	8080 tienda1	61048
4	normal get	43088
5	get http	43088

LSTM modele HTTP veri setinin tamamının uygulanması için optimum modellenmesi için saklı birim (hidden layer) sayısı 100, sınıflandırılacak etiketler geçerli - geçersiz (valid - unvalid) olmak üzere 2 sınıflı, maksimum epok sayısı 70, öğrenme kat sayısı (learning rate) 0,001, miniBatchSize değeri 8 olarak seçilmiştir. Modelleme uygulama adımları şöyledir:

3.1. LSTM Modele HTTP Veri Setinin Tamamının Uygulanması

Adım 1: $60{,}000 \times 80{,}145$ boyutlu veri setinin transpozesi hesaplanmış alınmış,

Adım 2: Transpozu hesaplanan matrisin her bir satırı tek tek hücreler haline getirilerek $1 \times 60{,}000$ boyutlu bir hücre matrisi (cell matrix) elde edilmiştir.

Adım 3: Her bir hücre içerisinde o girdiye ait 80,146 adet öznitelik bilgisi bulunmaktadır. Daha sonra elde ettiğimiz hücre matrisinin tekrar transpozu alınarak LSTM algoritmasına uygulanmıştır.

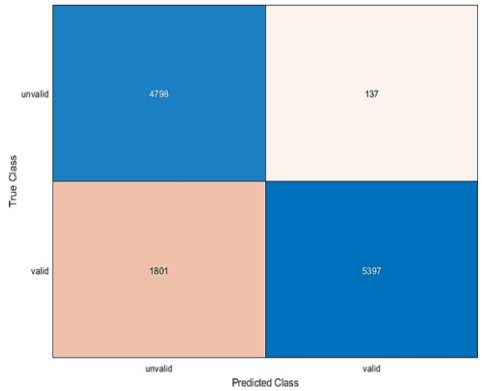
Adım 4: HTTP veri setinin %80 eğitim, %20'si test işlemleri için ayrılmıştır. Hücre haline getirilmiş her bir girdi için ayrı ayrı sınıf etiketleri başka bir kategorik matris içerisinde oluşturulmuştur.

3.2. LSTM Modele HTTP Veri Setinin Gruplanarak Uygulanması

LSTM modele HTTP veri setinin gruplanarak uygulanmasında optimum değerler saklı birim (hidden layer) sayısı 100, sınıflandırılacak etiketler geçerli - geçersiz (valid - unvalid) olmak üzere 2 sınıflı, maksimum epok sayısı 70, öğrenme kat sayısı (learning rate) 0,001, miniBatchSize değeri 8 olarak seçilmiştir. Modelleme uygulama adımları şöyledir:
Adım 1: Transpozu hesaplanan matrisin her bir satırı tek tek hücreler haline getirilerek 1×60,000 boyutlu bir hücre matrisi (cell matrix) elde edilmiştir.
Adım 3: Her bir hücre içerisinde o girdiye ait veri 128'erli gruplar halinde bölünmüş öznitelik bilgisi bulunmaktadır. Daha sonra elde ettiğimiz hücre matrisinin tekrar transpozu alınarak LSTM algoritmasına uygulanmıştır.
Adım 4: Gruplanan HTTP veri setinin 2-fold çapraz doğrulama yöntemi ile doğruluk oranı hesaplanmıştır.

3.3. Deneysel Sonuçları Değerlendirilmesi

Web saldırı tespitinin başarım oranına etkisini tespit etmek için LSTM modele HTTP veri setinden elde edilen hücresel verinin tamamı ve HTTP veri setinden elde edilen hücresel veri gruplanarak uygulanmıştır. Model başarımında HTTP veri setinin tamamının LSTM modele uygulanması ile en başarılı doğruluk oranı elde edilmiştir. Bu çalışmada yapılan deneysel sonuçlara göre, web saldırı tespitinde valid ve unvalid tespit için optimum değerler; maksimum epok sayısı için 70, öğrenme kat sayısı (learning rate) için 0,001, miniBatchSize değeri için 32 olduğu sonucuna varılmıştır. mini- batch size değeri 8'den 32'ye çıkarıldığında eğitim işlemi hem daha hızlı hem de daha yüksek doğruluk oranı vermiştir. İki model karşılaştırıldığında;
- Modelin eğitim işlemi için gerekli süre farkı 68,199 saniye olmuştur
- Eğitim işleminde doğruluk oranı %81'den %86'ya yükselmiştir.
Mini-batch-size değeri = 32 için geliştirilen modelin test başarımı Şekil 1'deki karışıklık matrisi ile verilmiştir.



Şekil 1. Mini-batch-size = 32 için modelin test başarımı

Şekil 1'e göre elde edilen model test başarımı Çizelge 2'de verilmiştir. Çizelge 2'ye göre test doğruluk oranı, duyarlılık oranı, özgüllük oranı sırasıyla %84,02, %97,20, ve %75,00 elde edilmiştir.

Çizelge 2: Modelin değerlendirilmesi

Performans Ölçütü	Başarım oranı (%)
Doğruluk oranı	84,02
Duyarlılık	97,20
Özgüllük	75,00

4. Sonuçlar

Bu çalışmada, web uygulama güvenlik duvarlarının performansının ve güvenilirliğinin sağlanmasında HTTP DATASET CSIC 2010 verisi kullanılmıştır. Veri seti içerisindeki 46 bin farklı kelimeden özellik olarak alınabilecekler bulunup yapay zekâ algoritmalarından LSTM yöntemiyle iki farklı şekilde modellenmiştir. LSTM yöntemi ile modellenen veri seti için valid ve invalid olan web saldırı tespiti yapılmıştır. Mini batch size = 32 için model başarımının, mini batch size = 8 model başarımından %5'lik oranda ve modelleme için geçen zaman bakımından daha hızlı olduğu sonucuna varılmıştır. En başarılı model olan mini batch size = 32 için sınıflandırmada test başarımı sırasıyla doğruluk oranı için %84,02, duyarlılık oranı için %97,20 ve özgüllük oranı için %75,00 elde edilmiştir. Bu çalışma, web uygulama güvenlik duvarlarının performansının ve güvenirliğinin arttırılmasında, WAF sistemlerinin kurulu olduğu sistemlerin güvenliğini arttırmada önemli katkılar sağlayacaktır. Elde edilen bu model sayesinde web uygulamalarına gelen paketler ön işleme alınacak bir web uygulaması ile bütünleşmiş hale getirilecektir. Sistem üzerine kurulan yazılımın sistem üzerindeki darboğaz etkisi ölçülecek ve minimum seviyede sistemi yoracak seviyeye getirilmesi hedeflenecektir. Gelecekte bu aşama, çalışmanın geliştirilme ve ürün haline getirilmesi aşamasına dahil edilecektir.

Ek A

Teşekkür

Bu çalışma İstanbul Sabahattin Zaim Üniversitesi Proje Destek Programı (İZUPRO) kapsamında İZÜ-ÖGR-2022-23 numaralı proje için desteklenmiştir.

5. Kaynaklar

[1] Owasp wiki. Owasp. https://tr.wikipedia.org/wiki/Owasp. 2022. Erişim tarihi 10 Şubat 2022.

[2] Wichers, D. Owasp top-10 2013. OWASP Foundation, https://owasp.org/www-project-top-ten. February, 2013. Erişim tarihi 10 Şubat 2022.

[3] Karacan, H. ve Sevri, M., "A Novel Data Augmentation Technique and Deep Learning Model for Web Application Security", IEEE Access, 9 (150), 781–150 797, 2021.

[4] Hassan, M. M., Ahmad, R. B. ve Ghosh, T., "Sql Injection Vulnerability Detection Using Deep Learning: A Feature-Based Approach", Indonesian Journal of Electrical Engineering and Informatics (IJEEI), 9(3), 702–718, 2021.

[5] Fang, Y., Li, Y., Liu, L. ve Huang C., "Deepxss: Cross Site Scripting Detection Based on Deep Learning", in Proceedings of the 2018 International Conference on Computing and Artificial Intelligence, 47–51, 2018.

[6] Karacan, H. ve Sevri, M., "A Novel Data Augmentation Technique and Deep Learning Model For Web Application Security", IEEE Access, 9, 150781-150797, 2021.

[7] Torrano-Gimenez, C., Nguyen, H. T., Alvarez, G., Petrović, S. ve Franke, K., "Applying Feature Selection to Payload-Based Web Application Firewalls", In 2011 Third International Workshop on Security and Communication Networks (IWSCN), 75-81, 2011.

[8] Vartouni, A. M., Teshnehlab, M. ve Kashi, S. S. "Leveraging Deep Neural Networks for Anomaly ‐ Based Web Application Firewall", IET Information Security, 13(4), 352-361, 2019.

[9] Liang, J., Zhao, W. ve Ye, W., "Anomaly-based Web Attack Detection: A Deep Learning Approach", In Proceedings of the 2017 VI International Conference on Network, Communication and Computing, 80-85, 2017.

[10] Tekerek, A., "A Novel Architecture for Web-Based Attack Detection Using Convolutional Neural Network", Computers & Security, 100, 102096, 2021.

[11] Zuech, R., Hancock, J., Khoshgoftaar ve T. M., "Detecting Web Attacks Using Random Undersampling and Ensemble Learners", Journal of Big Data, 8(1), 1-20, 2021.

[12] Torrano-Gimenez M. C. ve Villegas, A. P., http dataset csic 2010, Website: https://www.tic.itefi.csic.es/dataset/. 2012.Erişim tarihi 15 Şubat 2022.

[13] Bayrak, S., Yucel, E. ve Takci, H., "Epilepsy Radiology Reports Classification Using Deep Learning Networks", Cmc-Computers Materials & Continua, 70(2), 3589-3607, 2022.

[14] Çakmak, E. ve Selvi, İ. H., "Derin Öğrenme (CNN, RNN, LSTM, GRU) Kullanarak Protein İkincil Yapı Tahmini", Acta Infologica, 6(1), 43-52, 2022.

[15] Tummalapalli, S. ve Lalita Bhanu Murthy, N., "Web Service Anti-Patterns Prediction Using Lstm with Varying Embedding Sizes", In International Conference on Advanced Information Networking and Applications, 399-410, 2022.

[16] Yu, Y., Si, X., Hu, C. ve Zhang, J., "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures", Neural computation, 31(7), 1235-1270, 2019.