**PROJE ADI**

**Yapay Zeka Destekli**

**Gerçek Zamanlı Oltalama Önleme Sistemi**

İçindekiler

[1. ÖZET 2](#_Toc30114572)

[2. GİRİŞ 2](#_Toc30114573)

[2.1 Projenin Amacı 2](#_Toc30114574)

[2.2 Saldırı Yöntemleri 3](#_Toc30114575)

[2.3 Oltalama Saldırısı (Phishing) 3](#_Toc30114576)

[2.3.1 URL’ler Yerine IP Adresinin Kullanılması 3](#_Toc30114577)

[2.3.2 URL Taklidi 3](#_Toc30114578)

[2.3.3 URL Yönlendirilmesi 3](#_Toc30114579)

[2.4 Karşı Site Programlaması (Cross-Site Scripting, XSS) 3](#_Toc30114580)

[2.4.1 Yansıtılmış XSS (Reflected XSS) 4](#_Toc30114581)

[2.4.2 DOM (Document Object Model) XSS 4](#_Toc30114582)

[2.4.3 Depolanmış XSS (Stored XSS) 4](#_Toc30114583)

[2.5 XSS Oltalama 4](#_Toc30114584)

[2.6 Piyasa Örnekleri 4](#_Toc30114585)

[3 YÖNTEM 4](#_Toc30114586)

[3.1 Çözüm Yöntemi 4](#_Toc30114587)

[3.2 Python 5](#_Toc30114588)

[3.3 Java 5](#_Toc30114589)

[3.4 Random Forest Algoritması 5](#_Toc30114590)

[3.5 MySQL 5](#_Toc30114591)

[3.6 Kötü Amaçlı E-Posta Algılama 5](#_Toc30114592)

[3.7 Kötü Amaçlı JavaScript Kodunun Algılanması 7](#_Toc30114593)

[3.8 Yapay Zeka 8](#_Toc30114594)

[3.8.1 Veri Kümesinin Oluşturulması 9](#_Toc30114595)

[3.8.2 JavaScript Özelliklerinin Belirlenmesi 9](#_Toc30114596)

[3.8.3 Sınıflandırma Algoritmasının Seçilmesi 9](#_Toc30114597)

[3.8.4 Yapay Zeka Modelinin Eğitilmesi 10](#_Toc30114598)

[3.8.5 Yapay Zeka Modelinin Test Edilmesi 10](#_Toc30114599)

[3.9 Kullanıcı Kontrol Paneli 10](#_Toc30114600)

[4 İŞ-ZAMAN ÇİZELGESİ 11](#_Toc30114601)

[5 BULGULAR 11](#_Toc30114602)

[6 SONUÇ VE TARTIŞMA 12](#_Toc30114603)

[7 ÖNERİLER 12](#_Toc30114604)

[8 KAYNAKLAR 13](#_Toc30114605)

[9 EKLER 14](#_Toc30114606)

# ÖZET

Günümüzde yazılım ve sistemlerin daha güvenli hale gelmesiyle birlikte sosyal mühendislik saldırıları hız kazanmaktadır. Bir sosyal mühendislik türü olarak tanımlanan oltalama saldırısıyla (phishing) kendilerine güvenilir kurum görüntüsü veren dolandırıcılar, ya rastgele ya da ele geçirilen bilgilerinden kullanıcı hesaplarına elektronik posta göndererek veya zararlı internet siteleri aracılığıyla kişisel bilgileri çalmaktadır. E-postalar, bilinen web sitelerinden veya kullanıcının bankasından, kredi kartı şirketinden, e-posta veya internet hizmeti sağlayıcısından gönderilmiş gibi gösterilmektedir. Bu saldırı türünde kurban, kendisine gönderilmiş e-posta içerisindeki URL’ye tıklayarak kötü amaçlı bir internet sitesine yönlendirilmektedir. Ardından kulllanıcının tarayıcısında zararlı JavaScript kodu çalıştırılarak veriler saldırganlarca ele geçirilmektedir. Hatta bu saldırı türü güvenli sitelere kötü niyetli betiklerin enjekte edilmesini sağlayan Karşı Site Programlaması (Cross Site Scripting, XSS) saldırısıyla birleştirilerek özellikle deneyimsiz kullanıcılar tarafından yakalanması daha zor hale gelebilmektedir. Bu projede bu kapsamdaki bütün oltalama saldırılarının önlenmesi amaçlanmaktadır. Bu amaçla geliştirilen yazılım, kullanıcının elektronik posta kutusunu takip ederek yeni gelen bütün e-postaları gönderen bilgisi ve sunucusuna bakarak kara liste veritabanından kontrol etmekte, ayrıca e-postanın içeriğindeki bağlantıları ve yönlendirdiği internet sayfalarını inceleyerek de kötü amaçlı JavaScript kodlarını yapay zeka yöntemiyle tespit etmektedir. Kötü amaçlı olduğu tespit edilen e-postalar otomatik olarak çöp kutusuna taşınmaktadır. Eş zamanlı olarakda kullanıcı uyarılmaktadır.

**Anahtar Kelimeler:** sosyal mühendislik, siber güvenlik, kötü amaçlı yazılım, yapay zeka

## GİRİŞ

## Projenin Amacı

Projede hedeflenen, e-posta sistemlerini kullanan kullanıcıları oltalama saldırılarına karşı korumaktır. Böylelikle kredi ve kimlik kartı bilgileri, şifreler gibi kritik bilgilerin saldırganların eline geçmesi ya da kullanıcın cihazının güvenliğinin ihlal edilmesinin önüne geçilmesi hedeflenmektedir.

## Saldırı Yöntemleri

Sosyal mühendislik; etkileme, zorlama, aldatıcı ilişkiler geliştirme, sorumluluğu, etik değerleri, dürüstlüğü ya da bağlılığı azaltma amacını güden yöntemler kullanarak kişileri gizli bilgi vermeleri veya erişim sağlamaları için aldatma sürecidir (Bağcı, 2016). Günümüzde yazılımların ve kullanılan donanımların daha güvenli hale gelmesiyle birlikte sosyal mühendislik saldırıları hız kazanmaktadır.

## Oltalama Saldırısı (Phishing)

Sosyal mühendisliğin bir alt türü (2009, CISA) olan oltalama saldırısı, dolandırıcıların rastgele kullanıcı hesaplarına e-mail gönderdikleri bir çevrimiçi saldırı türüdür. E-postalar, bilinen web sitelerinden veya kullanıcının bankasından, kredi kartı şirketinden, e-posta veya internet hizmeti sağlayıcısından gönderilmiş gibi gösterilmektedir. Bu e-postalar kullanıcıları bir başka web sitesine yönlendirmekte, yönlendirilen bu sitelerde genellikle hesapları güncelleyebilmek için kredi kartı numarası veya şifre gibi kişisel bilgiler sorulmaktadır. Bu site aslında ya sahte ya da değiştirilmiş bir web sitesidir. Kullanıcılardan da bu siteye gittiklerinde oltalama saldırısını yapan kişiye iletilmek üzere kişisel bilgilerini girmeleri istenmektedir (2019, Afyon Kocatepe Üniversitesi).

Oltalama saldırısı yöntemleri aşağıda sıralanmıştır:

### URL’ler Yerine IP Adresinin Kullanılması

Oltalama saldırısını kullanan saldırganlar bir bağlantının nereye gittiğini kullanıcıdan gizlemek için URL ler yerine IP adreslerini kullanmaktadır. Örneğin saldırgan http://zararlısite.com yerine http://178.14.255.0 yazdığında URL’nin e-posta sunucusunda halihazırda bulunan spam filtresini atlatma şansı artmaktadır.

### URL Taklidi

URL taklidinde saldırgan güvenilir bir URL’ye çok benzer bir URL ile kullanıcıyı kandırmaya çalışmaktadır. Örneğin saldırgan, “http://facebook.com” yerine “http://faceboook.com” adresini kullanarak kullanıcıya facebook’a girdiğini düşündürüp şifresini ele geçirebilir. Bu durumun önüne geçilebilmesi için tasarlanan projede bu çeşit alan adları kara listeye alınmaktadır ve algılandığında kullanıcı uyarılıp, e-posta çöp kutusuna atılmaktadır.

### URL Yönlendirilmesi

Bu filtreleme atlatma yönteminde ise saldırgan güvenilir bir web sitesinin yönlendirme bölümünün URL’ini vermektedir. Bu yöntem ile kullanıcı güvenli bir siteye gittiğini zannetmektedir ve filtreleme araçlarıda sadece URL’yi kontrol ederek tespit edilememektedir. Projemizde bunun engellenmesi için programımız bu URL leri kontrol etmektedir.

## Karşı Site Programlaması (Cross-Site Scripting, XSS)

Saldırganlar web sitelerindeki güvenliği, kullanıcılara farkettirmeden zararlı kodlar çalıştırarak da aşabilmektedir. Bu HTML kodlarının arasına istemci tabanlı kodların çalıştırılmasıyla gerçekleşmektedir. Bu kodlar genellikle zararlı bir siteye yönlendirilmek için kullanılmaktadır. Ama aynı zamanda kullanıcın tarayıcısındaki açıklar kullanılarak çerezlerin (cookie) çalınması vb. gibi olaylarla sonuçlanan saldırırlar gerçekleşebilmektedir.

XSS yöntemleri aşağıda sıralanmıştır:

### Yansıtılmış XSS (Reflected XSS)

Doğrudan URL satırına zararlı kod yetiştirilmesiyle yapılan saldırı türüdür. URL’nin kontrol edilmesiyle rahatlıkla anlaşılmaktadır. Bu saldırı arka uçta (backend) bulunan açıktan faydalanmaktadır.

### DOM (Document Object Model) XSS

XSS saldırısına benzerlik göstererek zararlı kodların URL satırına yerleştirilmesiyle gerçekleştirilmektedir. URL kontrol yöntemi ile algılanabilmektedir. Saldırgan, web sitesinin DOM’ u (Document Object Model) üzerinde değişiklikler yapmaktadır.

### Depolanmış XSS (Stored XSS)

Depolanmış XSS saldırısı web sitesindeki girdi girilebilecek bir bölümde meydana gelen bir açıklıktır. Bu saldırı doğrudan URL den aldılanamadığı için algılanması diğerlerine kıyasla daha zordur.

## XSS Oltalama

URL tabanlı XSS zayıflıkları, geleneksel oltalama saldırılarına daha kötü ve tespiti daha zor yeni bir yöntem kazandırmıştır (Checkmarx, 2016). Daha basit oltalama saldırıları sahte bağlantılarla, örneğin metin kısmında Facebook göstererek URL’de bir IP vermek veya orijinal adrese bir harf eklemek, yapılırken XSS oltalama saldırılarındaki bağlantılar gerçek olabilmektedir. Normalde URL sonuna ek yaptığından tespiti kolay olmakla birlikte, bağlantı güvenilir bir internet sitesine olduğundan deneyimsiz kullanıcının gözünden rahatlıkla kaçabilmektedir.

## Piyasa Örnekleri

Bu sorunu çözebilmek için piyasayada bir çok ürün bulunmaktadır. Ancak bu ürünlerin tamamı, maillerin güvenliğini kontrol ederken sadece mailin içerisinde hali hazırda bulunan örnekleri incelemekte, bu durum sorunun tamamen çözülmesini sağlayamamaktadır. Tasarlanan projede, hem postanın içeriği, hem de postanın kullanabileceği bütün kandırma tekniklerinin önüne geçilerek hem oltalama, hem de spear phishing saldırılarından piyasada bulunan ürünlere kıyasla çok daha üstün bir korunma sağlanmıştır.

# YÖNTEM

## Çözüm Yöntemi

Geliştirilen yazılımda oltalama tespitinde kara liste yöntemi ile çok bilinen oltalama postaları engellenmektedir. Sonrasında ise postadaki her şüpheli olay için postaya 0 dan başlayan bir risk puanı verilmektedir. Postanın risk puanı kullanıcının arayüzden seçtiği bir limiti geçerse şüpheli olarak işaretlenerek çöp kutusuna taşınmaktadır. Postanın risk puanını belirleyen faktörler ise mesajın içerisindeki URL’ler ve yönlendirdikleri internet sayfalarındaki JavaScript kodlarıdır. Program gönderilen internet sayfalarının kötü amaçlı olup olmadığını algılamakta ve risk puanını internet sitelerine göre arttırmaktadır. Eğer internet sitesinde depolanmış XSS algılanırsa URL’ni alan adı kara listeye eklenmektedir.

Yazılımın geliştirilmesinde Python ve Java programlama dilleri kullanılmaktadır. Kara liste ve kullanıcı bilgilerinin kaydedilebilmesi için MySQL kullanılmaktadır.

## Python

Python programlama dili 1991 yılında geliştirilmiş, ve geçen süre zarfında ilk halinden önemli ölçüde farklılaşmıştır. Geniş kütüphane desteği, entegrasyon olanakları ve yüksek üretkenlik özellikleriyle son yıllarda C, C++ ve Java gibi önde gelen programlama dillerini de geride bırakarak 2017 yılında IEEE tarafından en çok kullanılan programla dili olarak belirlenmiştir (Hruska, 2017). Python, popülerliği, sağladığı kütüphane zenginliği, çoğu siber güvenlik uzmanının tercihi olması ve olası problemlere çözüm bulma kolaylıkları gibi avantajlarından dolayı proje kapsamında kullanılmasına karar verilmiştir.

## Java

Java ilk olarak Sun Microsystems tarafından 1995 yılında piyasaya sürülen bir programlama dili ve bilgi işlem platformudur. Java neredeyse her türdeki ağ uygulamalarının temelini oluşturarak gömülü ve mobil uygulamalar, oyunlar, Web tabanlı içerik ve kurumsal yazılım geliştirme ve dağıtımı için küresel standarttır. Dünya genelinde 9 milyonu aşan geliştiriciyle Java, heyecan verici uygulamaları ve hizmetleri verimli bir şekilde geliştirmenize ve dağıtmanıza olanak tanır (Java, 2019).

## Random Forest Algoritması

Random Forest birkaç “tree” tahmincisinin kombinasyonuyla oluşur. Ve her bir “tree” rastgele bir vektörün değerini alır. Buradaki asıl prensip ise çok sayıdaki zayıf öğrenicinin birleşip güçlü bir öğrenici oluşturmasıdır. Tahminler yapmak için güzel bir araçtır. Aynı zamanda bu model büyük sayılar kanunundan dolayı aşırı uyum sağlamaz (Abu Nimeh, Nappa, Wang, Nair 2007).

## MySQL

MySQL bir ilişkisel veritabanı yönetim sistemidir, çoklu iş parçacıklı (multi-threaded), çok kullanıcılı (multi-user), hızlı ve sağlam (robust) bir veritabanı yönetim sistemidir (MySql, 2019).

## Kötü Amaçlı E-Posta Algılama

Geliştirilen yazılım, kullanıcının sisteme eklediği her posta hesabı için bir MailBox[[1]](#footnote-1) sınıfı oluşturmaktadır. Her bir MailBox sınıfı farklı bir iş parçasında(Thread) oluşturulmakta, kullanıcının her posta hesabı için eş zamanlı ve birbirinden bağımsız bir şekilde takibi sağlanmaktadır. Posta kutularının birbirinden bağımsız olmasının bir başka avantajı da bir posta sunucusunda yaşanan bir sıkıntının diğer mailbox objelerini etkilememesidir. Bu sayede sunucu tarafında yaşanan sıkıntıların sistemin geri kalanını etkilemesinin önüne geçilmiş olur.

Her hesap için oluşturulan MailBox objesi, eğer o hesap daha önce sisteme eklenmediyse son postalardan kullanıcı tarafından belirlenen miktarını işlemden geçirerek önbelleğine kaydeder, böylece programın sonraki çalışmalarında sadece son postalardan ön belleğe alınmayanlar işlenerek işlem gücünden ve zamandan tasarruf sağlanır. MailBox objesi oluşturulduğunda get\_new\_mails fonksiyonunu çalıştırarak önbelleğe alınmayan postaları tarar ve her yeni posta için bir Mail objesi oluştururur. Ardından, kullanıcının belirlediği aralıklarla refresh\_new\_mails fonskiyonu çağrılarak yeni mail alınıp alınmadığı kontrol edilir. Bu fonskiyon, mail listesini sunucudan çekerek önceki listeyle karşılaştırarak çalışması ve sadece metinleri çekilmesi sayesinde hem işlem gücü hem de internet kullanımı bakımından fazlasıyla tasarruflu çalışmaktadır. Bu işlemi gerçekleştiren kod *Kaynak Kod 1*’ de verilmiştir

def refresh\_new\_mails(self):

old\_ids = self.ids

self.get\_ids()

diff\_ids = set(self.ids) - set(old\_ids)

for id in diff\_ids:

self.log.info("Found new mail with id: " + str(id))

typ, messageRaw = self.imap.uid('fetch',id,'(RFC822)')

email = messageRaw[0][1]

self.mails[id] = Mail(email, self.mysql\_creds,self.threshold,self.sensitivity,self.FROM\_EMAIL, id)

self.mails[id].check\_spam()

Kaynak Kod 1: Yeni e-postaları bulan fonksiyon

E-postaların sunucudan çekilmesinin ve önbelleğe alınmasının ardından program, e-postanın içerisinden gönderenin e-posta adresi, kullandığı sunucu ve e-postanın içerdiği mesajı çıkarmaktadır. Bu işlemden sonra, gönderenin e-posta adresi ve sunucusu MySQL sunucusunda bir sorgu çalıştırılarak e-posta adresinin, sunucunun IP’sinin ya da alan adının kara listede olup olmadığı kontrol edilmektedir.

Mysql veritabanında yapılan sorgulardan herhangi bir sonuç dönmez ise, postanın içeriğindeki URL'ler aranmakta ve bir diziye (array) kaydedilmektedir. Bundan sonra, bu URL'lerin her biri ziyaret edilerek internet sitelerinden döndürülen yanıtlar incelenmektedir, eğer başka bir websitesine yönlendiriyorsa takip edilmektedir. Bu internet sitelerinin içerisindeki her bir URL de kaydedilir ve aynı işlemden geçirilir. Bu süreç, e-postanın kullanıcıyı mantıklı sınırlar içinde yönelendirebileceği bütün internet siteleri bulunana kadar devam eder. URL'lerin bulunmasının ardından, bu URL'ler de kara liste veritabanında sorgulanmaktadır. Eğer sorguların herhangi birinden pozitif sonuç döndürülürse, posta çöp kutusuna taşınmaktadır. Ardından çöp olarak işaretlendiği tarih ve gönderen bilgileri ile beraber loglanmaktadır. Eğer pozitif sonuç döndürülmezse, bu urllerin içerisinde XSS saldırıları için kullanılabilecek karakterlerin düz metin, tek kodlanmış ve çift kodlanmış formatları aranmaktadır. Böylece sık kullanılan bir filtre atlatma tekniği olan "Double Encoding" tekniğinin önüne geçilmektedir (Owasp, 2019). Bu aramalardan pozitif sonuç dönmesi durumunda, risk puanı arttırılmaktadır. Bundan sonra, URL'lerin içerisinden JavaScript dosyası olanlar alınmaktadır ve her internet sitesinin HTML'inden JavaScript kodları çıkarılarak Random Forest Modeli ile çalışan bir yapay zeka aracılığıyla bu JavaScript kodlarının iyi ya da kötü amaçlı olduğu ayırt edilmektedir. Kötü amaçlı javascript kodunun algılandığı durumda, risk puanı yapay zekanın çıktısında kötü amaçlı olup olmamasına göre arttırılmaktadır.

## Kötü Amaçlı JavaScript Kodunun Algılanması

JavaScript, web tasarımcıları için çok yararlı bir araç olmakla beraber günümüzün en çok kulanılan dili haline gelmiştir. Dolayısıyla da bütün modern internet tarayıcıları javascript dilini desteklemektedir. Bu özelliği, JavaScript'i saldıraganlar arasında da sık kullanılan bir araç haline getirmiştir. Dolayısıyla da kötü amaçlı JavaScript kodlarının algılanıp işaretlenmesi kullanıcı güvenliği için kritik önem taşımaktadır.

Buna karşın, JavaScript'in çok esnek bir dil olması sebebiyle kullanıcıya gönderilen kodun kötü amaçlı olup olmadığı, modern antivirüs yazılımlarındaki gibi bir virüs veritabanı kullanarak algılanamamaktadır. Ayrıca, JavaScript dilinin sürekli olarak gelişmesi sebebiyle durmaksızın yeni virüsler ve saldırı tipleri ortaya çıktığından en mantıklı çözüm, kötü amaçlı JavaScript kodlarının sahip olduğu özelliklerin bir listesini çıkararak iyi ve kötü amaçlı JavaScript kodlarından bu özelliklerin sayısını kaydedip sonrasında bu veri ile bir yapay zeka eğitmektir.

JavaScript dili ne kadar engin bir dil olsa da, tarayıcıların zafiyetlerini kullanarak kullanıcının bilgisayarına erişim elde etmek için sıklıkla kullanılan bir dizi fonksiyon vardır. Bu fonksiyonlar ve kötü amaçlı JavaScript kodunu belirlemek için kullanılan özelliklerin listesi *Tablo 1*'de verilmiştir. Ancak saldırganlar, filtreleri atlatmak için kod karartma[[2]](#footnote-2) (obfuscation) teknikleri kullanmakta ve JavaScript filtrelerini atlatmaktadır. Bunun önüne geçmek için de JavaScript kodunu işlemeden önce tersine karartma (deobfuscation ) yaparak kodun olabildiğince büyük bir kısmını işlemden geçirmek hız-performans bakımından en ideal yöntem olarak kabul edilmektedir.

|  |  |
| --- | --- |
| **JavaScript Özelliği** | **Tanımı** |
| eval() | eval() fonksiyonu sayısı |
| setTimeout() | setTimeout() fonksiyonu sayısı |
| iframe | “iframe” içeren metin sayısı |
| unescape() | unescape() fonksiyonu sayısı |
| escape() | escape() fonksiyonu sayısı |
| classid | “classid” sayısı |
| parseInt() | parseInt() fonksiyonu sayısı |
| fromCharCode() | fromCharCode() fonksiyonu sayısı |
| ActiveXObject() | ActiveXObject() fonksiyonu sayısı |
| Doğrudan metin atama | Doğrudan metin atama sayısı |
| concat() | concat() fonksiyonu sayısı |
| indexO() | indexOf() fonksiyonu sayısı |
| substring() | substring() fonksiyonu sayısı |
| replace() | replace() fonksiyonu sayısı |
| document.addEventListener() | document.addEventListener() fonksiyonu sayısı |
| attachEvent() | attachEvent() fonksiyonu sayısı |
| createElement() | createElement() fonksiyonu sayısı |
| getElementById() | getElementbyId() fonksiyonu sayısı |
| document.write() | document.write() fonksiyonu sayısı |
| JavaScript kelimeleri | JavaScript kelime sayısı |
| JavaScript anahtar kelimeleri | JavaScript anahtar kelime sayısı |
| JavaScript karakterleri | JavaScript karakter sayısı |
| Anahtar kelime / kelime | Anahtar kelime / kelime oranı |
| Entropy | Bütün scriptin entropisi |
| En uzun JavaScript kelimesi | En uzun JavaScript kelimesinin uzunluğu |
| 200 den uzun metinler | 200 karakterden uzun metin sayısı |
| En kısa JavaScript kelimesi | En kısa JavaScript kelimesinin uzunluğu |
| En uzun JavaScript kelimesinin entropisi | En uzun JavaScript kelimesinin entropisi |
| Boşluk sayısı | JavaScripdeki boşluk sayısı |
| Kelimelerin ortalam uzunluğu | JavaScripteki kelimelerin ortalama uzunluğu |
| Hex değer sayısı | JavaScriptte kullanılan Hex değer sayısı |
| Boşluk karakterlerinin payı | Boşluk karakterlerinin payı |

Tablo 1: Yapay zeka aloritmasında kullanılan JavaScript özellikleri

## Yapay Zeka

Kötü niyetli internet sitelerinin belirlenmesi için yapay zeka kullanılması oldukça yaygın bir yöntemdir ve bu konuda çok sayıda makale bulunmaktadır. Bu çalışmalar oltalama, web spam ve malware tabanlı web saldırılarını tespit etmek için yapılmış ve kullandıkları özelliklere göre aşağıda gruplanmıştır (Šulak, 2018 ):

* **Anlamsal özellikler:** URL metninden çıkarılan özelliklerden oluşur. Kötü niyetli URL’in zararsız bir URL’e göre şüpheli olabileceği varsayımına dayanır.
* **Sunucu tabanlı özellikler:** WHOIS ve DNS bilgisine dayanır. IP adresi, coğrafi yer, bağlantı hızı ve diğer bazı özelliklikler kullanılır.
* **İçerik tabanlı özellikler:** HTML, JavaScript kodu veya ActiveX işlevlerine odaklanabileceği gibi internet sitesinin göresellerini karşılaştırarak da kötü niyetli internet siteleri belirlenmeye çalışılabilir. Bu proje kapsamında bu özelliğin bir alt grubu olan koda dayalı özellikler kullanılmıştır.
* **Kapsam tabanlı özellikler:** URL’in paylaşıldığı yer gibi geri plandaki özellikleri kullanır.
* **Popülerlik tabanlı özellikler:** Farklı arama motorlarından elde edilen bilgiler kullanılır.

Proje kapsamında kullanılan yapay zeka çözümünde aşağıdaki adımlar izlenmiştir:

### Veri Kümesinin Oluşturulması

Yapay zekadan beklenen başarıyı seçilen algoritmadan çok veri kümesinin kalitesi belirlemektedir. Başarı oranını yükseltmek için doğru verinin bulunması ve uygun formatta hazırlanması gerekmektedir (Pahmind, 2019).

Kötü amaçlı Javascriptlerin paylaşıldığı standart bir internet adresi bulunmamaktadır. Bu konuda çalışma yapanlar veri kümelerini kendi yazdıkları betiklerlerle bazı değerlendirme (Benchmark) sitelerini kullanarak internetten indirmektedirler. Bu çalışma kapsamında kullanılan kötü niyetli JavaScript kodları daha önce başka bir araştırmacı tarafından derlenerek HynekPetrak-Github (Github, 2019) adresinde paylaşılan dosyalardan oluşmaktadır. Zararsız örnekler ise “Quancast”‘in en çok ziyaret edilen web siteleri listesindeki (Quancast, 2019) ilk 1000 siteden, python kullanılarak yazılan betikler vasıtasıyla toplanmıştır. Web sitesinde referans verilen javascript kodları *Kaynak Kod 2*’deki betik ile algılanır.

regex = re.compile(r'src? \*=? \*"(((?<=\\)"|([^"]))\*..js)"')

js\_files = [x[0] for x in regex.findall(html)]

Kaynak Kod 2: Html kodunun içinde referans verilen javascript kodlarını algılayan betik

Toplanan örneklerin dağılımı *Tablo 2*’de gösterilmektedir.

|  |  |  |  |
| --- | --- | --- | --- |
| **Görev** | **Eğitim** | **Test** | **Toplam** |
| **Zararlı** | 37461 | 1241 | 38702 |
| **Zararsız** | 3297 | 117 | 3414 |

Tablo 2: Javascript veri kümesi dağılımı

### JavaScript Özelliklerinin Belirlenmesi

Toplanan verilerin yapay zeka algoritmasında kullanılmak üzere anlamlı hale getirilmesi gerekmektedir. Toplanan her bir JavaScript için Şekil-3 de verilen 32 özellik belirlenmiştir.Bu özellikler farklı araştırmacıların çaışmalarında kullandıkları özeliklerden oluşmaktadır (Seshagiri, Vazhayil, Sriram, 2016; Cova, Kruegel, Vigna, 2010). Bunun için python kullanılarak geliştirilen özellik belirleme betiği kullanılmıştır. Bu işlem sırasında kötü niyetli JavaScriptler 0, zararsız olanlar ise 1 olarak etiketlenmiştir.

### Sınıflandırma Algoritmasının Seçilmesi

Kötü niyetli JavaSciptlerin sınıflandırılması ikili bir sınıflandırmadır; sadece kötü niyetli ve zararsız olarak sınıflandırılırlar. Yapay zeka teknikleri bu konuda oldukça etkilidir. Bu çalışma kapsamında denetimli öğrenme (supervised learning) algoritmaları icelenmiş ve kötü niyetli internet sitelerinin belirlenmesinde kullanılan SVM (Support Vector Machine), Logistic Regression, J48 Karar Ağacı, Naive Bayes, Random Forest vb. algoritmadan, bunların performans karşılaştırmasında oldukça iyi sonuçlar elde eden (Eshete, Villafiorita, Weldemarim, 2012; Patil, Patil, 2017) Random Forests algoritması kullanılmıştır.

### Yapay Zeka Modelinin Eğitilmesi

Python programlam dili ve başlıca tensorflow, keras, pandas ve sklearn kütüphaneleri kullanılarak yazılan programla yapay zeka modeli oluşturulmuş ve hazırlanan verilerle model eğitilmiştir. Model test aşamasında kullanılmak üzere kaydedilmiştir. Yapay zeka eğitimi ve kaydedilmesi işelemini yapan kod Kaynak Kod 3’tedir.

### Yapay Zeka Modelinin Test Edilmesi

parameters = {'bootstrap': True,

'min\_samples\_leaf': 3,

'n\_estimators': 100,

'min\_samples\_split': 10,

'max\_features': 'sqrt',

'max\_depth': 10,

'max\_leaf\_nodes': None}

model = RandomForestClassifier(\*\*parameters)

model.fit(train, train\_excepted\_results)

with open("AIName","wb") as ai:

pickle.dump(model, ai)

Kaynak Kod 3: Yapay zeka eğitim kodu

Bir önceki aşamada kaydedilen yapay zeka modeli daha önce eğitim aşamasında kullanılmayan kötü niyetli ve zararsız JavaScriptlerden olşan test veri kümesiyle test edilmiştir. Test sonuçlarına bulgular bölümünde yer verilmektedir.

## Kullanıcı Kontrol Paneli

Proje, son kullanıcı için tasarlandığından dolayı, kullanıcının aldığı postaların ne kadarının iyi amaçlı, ne kadarının kötü amaçlı olduğunu görüntüleyebileceği ve ayarları kolaylıkla değiştirebileceği bir kontrol paneli kritik önem taşımaktadır.

Kontrol panelinin arayüzü Java Swing ile yazıllmıştır ve 3 farklı sayfa içermektedir. Bu sayfaların biri kullanıcıyı bilgilendirmek için, diğer ikisi ise hesapların, kara listenin ve veritabanı bağlantısının kolaylıkla ayarlanması için eklenmiştir.

İlk sayfa , kullanıcının kolaylıkla okuyup anlayabileceği bir grafik arayüzüdür. Sayfa ilk açıldığında, MySQL veritabanındaki loglar çekilerek son 24 saat içerisinde saat başına alınan e-posta sayısı bir çizgi grafiği ile kullanıcıya gösterilmektedir. Bunun yanında, son 24 saatte alınan postalarının ne kadarının kötü amaçlı, ne kadarının iyi amaçlı olduğu yüzdelik olarak kullanıcıya bir pasta grafiği aracılığıyla sunulmaktadır. Ayrıca, kullanıcının tehdit analizini daha verimli bir şekilde yapmasını sağlamak için de oltalama postası gönderen sunucuların sayısı, çoktan aza doğru sıralanarak bir tablo ile sunulmaktadır. Bilgilendirme sayfası, şekil 7’da ve bilgilendirme sayfasını oluşturan *initGraph()* metodu *Kaynak Kod 4*’te görülebilir.

[Bilgilendirme Sayfası]

Şekil 7: Kullanıcı Bilgilendirme (Dashboard) sayfası

DateTimeFormatter pattern = DateTimeFormat.forPattern("YYYY-MM-dd HH:mm:ss");

DateTime start = DateTime.parse("2020-01-06 20:10:39", pattern);

XYSeries series = new XYSeries("");

for(int i = 1; i<=24; i++) {

String query = String.format("SELECT COUNT(\*)" +

" FROM logs" +

" WHERE timestamp>='%s'" +

" AND timestamp< '%s';",

pattern.print(start.minusHours(i)),

pattern.print(start.minusHours(i - 1)));

ResultSet rs = stmt.executeQuery(query);

rs.next();

int mail\_num = rs.getInt("COUNT(\*)");

series.add(i - 1, mail\_num);

Kaynak Kod 4: initGraph() metodu

Bunun yanında, kullanıcının kara listelere kolaylıkla farklı sunucular ve mail adresleri ekleyerek kara listelerin son kullanıcının tercihlerine göre özelleştirilmesi ve kullanıcının sorunsuz bir şekilde farklı posta kutuları eklemesini sağlamak için bir *Yönetim(Management)* sayfası oluşturulmuştur.

Yönetim sayfası kullanıcının program ile etkileşime geçebileceği tek arayüz olduğundan dolayı kullanıcı dostu olması oldukça önemlidir ki geliştirilen yazılım bu göz önüne alınarak yapılmıştır.

# İŞ-ZAMAN ÇİZELGESİ

İş zaman çizelgesi Tablo 3’te verilmiştir.

| İşler | Eylül | Ekim | Kasım | Aralık | Ocak |
| --- | --- | --- | --- | --- | --- |
| Literatür Taraması | **X** | **X** | **X** |  |  |
| Arka-uç Geliştirmesi |  | **X** | **X** | **X** |  |
| Arayüz Geliştirmesi |  |  | **X** | **X** | **X** |
| Yapay Zeka için Veri Toplanması |  |  | **X** | **X** |  |
| Yapay Zekanın Eğitilmesi |  |  |  | **X** | **X** |
| Proje Raporu Hazırlama |  |  |  | **X** | **X** |

Tablo 3: İş zaman çizelgesi

# BULGULAR

Proje kapsamında geliştirilen masaüstü yazılım aşağıda verilen tablodaki test adımlarından başarıyla geçmiştir.

|  |  |
| --- | --- |
| **Test** | **E-posta çöp kutusuna atıldı** |
| Zararlı içeriğe sahip olmayan e-posta | X |
| Kara listede bulunan e-posta adresinden gönderilen e-posta | ✔ |
| Kara listede bulunan URL içerikli e-posta | ✔ |
| İçinde Zararlı JavaScript bulunan URL içerikli e-posta (XSS oltalama) | ✔ |
| Kara listede olmayan bir URL içerikli e-posta; URL’dekara listede bulunan bir URL’ye referans var | ✔ |
| Kara listede olmayan bir URL içerikli e-posta; URL’de zararlı JavaScript’e referans var | ✔ |

Tablo 4: E-posta test sonuçları

Yapay zeka modeli ayrıca modelin eğitiminde kullanılmayan test veri kümesiyle test edilmiş ve Şekil-6’daki sonuçlar elde edilmiştir. Test sonuçlarındaki başarı oranı “%99.41” olarak hesaplanmıştır.

|  |  |  |
| --- | --- | --- |
| **Çıktı\Beklenen** | **Zararlı** | **Zararsız** |
| **Zararlı** | 1241 | 8 |
| **Zararsız** | 0 | 109 |

Tablo 5: Yapay zeka test sonuçları

# SONUÇ VE TARTIŞMA

Tasarlanan projede günümüz dünasının önemli problemlerinden olan oltalama sorununa çözüm getirilmiştir. Bu sorunu çözmek için kulanılan yazılım, IMAP protokolünü destekleyen herhangi bir posta hesabına bağlanarak gelen yeni mailleri inceleyerek maillerin içeriğindeki urlleri ve gönderici bilgilerini, tasarladığımız algoritma aracılığıyla değerlendirmekte ve algoritmanın çıktısına göre, kullanıcıya uyarı verilmektedir. Piyasada bulunan örneklerin aksine, proje, sadece gelen postaların içeriğini değil, aynı zamanda bu postanın kullanıcıyı yönlendirebileceği bütün web-sitelerini inceler ve böylece diğer ürünlere kıyasla çok daha derinlemesine ve etkili bir koruma sağlamaktadır. Ayrıca, proje, IMAP protokolünü destekleyen herhangi bir posta kutusuna bağlanabildiğinden, piyasaki ürünlerin aksine sadece büyük şirketler değil, son kullanıcı tarafından da kullaışabilmektedir.

# ÖNERİLER

Proje ne kadar piyasadaki ürünlere kıyasla çok daha etkili bir koruma sağlasa da, korumanın daha kapsamlı hale getirilebilmesi için yapılabilecek geliştirmeler de bulunmaktadır. Bu geliştirmelerin başında, sık kullanılan internet sitelerinin görüntüleri kaydedilerek e-postaların içerisindeki URL’lerin herhangi birinin bir Spear Phishing saldırısı olup olmadığı çok daha etkili ve doğru bir şekilde algılanabilir. Bunun yanında, MySQL veritabanından farklı olarak SQLite ya da MongoDB gibi farklı veritabanları da desteklenerek programın kullanımı son kullanııcı için daha da kolaylaştırılabilir.

# KAYNAKLAR

* Abu-Nimeh S, Nappa D, Wang X, Nair S. (2007) A comparison of machine learning techniques for phishing detection. Erişim adresi: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.114.1242&rep=rep1&type=pdf
* Afyon Kocatepe Üniversitesi (2019) Phishing (Oltalama Nedir?, Erişim tarihi: 10/12/2019, https://bim.aku.edu.tr/phishing-oltalama-nedir/
* Bağcı, H.(2016). Sosyal Mühendislik ve Denetim. Denetişim, (1) 42-51. Erişim adresi: https://dergipark.org.tr/tr/pub/denetisim/issue/22499/240554
* Checlmarx (2016) Everyone Talks About Phishing, But No One Blames XSS, Erişim tarihi: 17/12/2019, https://www.checkmarx.com/2016/04/26/everyone-talks-phishing-no-one-blames-xss/
* Cova M, Kruegel C, Vigna G. (2010), Detection and analysis of drive-by-download attacks and malicious JavaScript code. Proceedings of the 19th International Conference on World Wide Web; 2010. p. 281-90.
* Eshete B., Villafiorita A., Weldemarim K, (2012) , BINSPECT: Holistic Analysis and Detection of Malicious Web Pages, pp 149-166, Erişim adresi: https://link.springer.com/chapter/10.1007/978-3-642-36883-7\_10
* Github, (2019), Javascript malware collection, Erişim adresi: https://github.com/HynekPetrak/javascript-malware-collection
* Hruska, J. (2017), Python Tops 2017’s Most Popular Programming Languages. Erişim adresi: https://www.extremetech.com/computing/252987-python-tops-list-2017s-popularprogramming-languages
* Java (2019) Java teknolojisi nedir ve neden kullanmam gerekir?, Erişim tarihi: 17/12/2019, https://java.com/tr/download/faq/whatis\_java.xml
* Mysql (2019) About Mysql Erişim tarihi: 18/12/2019, https://www.mysql.com/about/
* Owasp (2019) Double Encoding, Erişim tarihi:20/12/2019, https://www.owasp.org/index.php/Double\_Encoding
* Pathmind (2019), Datasets and Machine Learning, Erişim tarihi:22/12/2019, Erişim adresi: https://pathmind.com/wiki/datasets-ml
* Patil D.R., Patil J.B. (2017) Detection of Malicious JavaScript Code in Web Pages, Indian Journal of Science and Technology, Vol 10(19), DOI: 10.17485/ijst/2017/v10i19/114828, Erişim adresi: http://www.indjst.org/index.php/indjst/article/view/114828
* Quincast, (2019), Top 1000 sites, Erişim adresi: https://www.quantcast.com/top-sites/
* Seshagiri P, Vazhayil A, Sriram P. AMA (2016), Static code analysis of web page for the detection of malicious scripts. Procedia Computer Science. 2016 Dec 31; 93:768-73.
* SICA (2009). Security Tip (ST04-014): Avoiding Social Engineering and Phishing Attacks Erişim tarihi: 19/12/2019, https://www.us-cert.gov/ncas/tips/ST04-014.
* Šulak (2018) Detection of Malicious Web Sites Using Machine Learning, BRNO UNIVERSITY OF TECHNOLOGY

# EKLER

1. MailBox sınıfı, kullanıcının posta kutusuna bağlanıp izlenmesi için kullanılan bir sınıfıdır. [↑](#footnote-ref-1)
2. Deobfuscation için pythonun jsbeautifier kütüphanesi kullanılmıştır. [↑](#footnote-ref-2)