SALDIRI TESPİT SİSTEMLERİNE MAKİNE ÖĞRENME ETKİSİ

***

THE EFFECT OF MACHINE LEARNING ON INTRUSION DETECTION SYSTEMS

Çağdaş ÖZER*

DOI: 10.33461/uybisbbd.558192

Mustafa TAKAOĞLU**

Öz

Teknoloji ilerledikçe ve insanlar ile makineler arasındaki bağlantı arttıkça, sistem ve veri güvenliği daha önemli hale gelmektedir. Saldırganlar, sistemleri inceleyerek açıklarını bulmaya çalışmakta ve kimi zaman da başarıya ulaşmaktadırlar. Başarıya ulaşan saldırılar maddi manevi zararlara yol açmaktadır. Bunların önüne geçebilmek için anti virüs veya güvenlik duvarları kullanılmaktadır. Anti virüs ve güvenlik duvarları uzman saldırganlara karşı her zaman etkin bir savunma sağlayamayabilirler. Bu ve benzer sorunlardan yola çıkılarak saldırı tespit sistemleri geliştirilmeye çalışılmıştır. Bunu, çeşitli sistemlerden ve ağ kaynaklarından bilgi toplayarak ve sonra olası güvenlik sorunları için bilgileri analiz ederek gerçekleştirirler. Çalışmamızda bu sorunlara odaklanılmış ve makine öğrenmesi tekniklerini, bilinen saldırı çeşitlerini ve sunucu tabanlı saldırı yöntemlerinin verilerini kullanarak saldırı tespit sistemi eğitmek amaçlanmıştır. Bu doğrultuda çalışmamızda, CesarFTP, WebDAV, Icecast, Tomcat, OS SMB, OS Print Spool, PMWiki, Wireless Karma, PDF N, Backdoored Executable, Browser Attack, Infectious Media saldırı verileri birleştirilerek veri seti oluşturulmuştur. Ortaya çıkan bu veri seti ise Destek Vektör Makinesi (DVM) ve Naive Bayes (NB) kullanılarak sınıflandırılmış ve eğitilmiştir ve elde edilen sonuçlar paylaşılmıştır. DVM ile sistemin eğitilmesi ve test edilmesinden sonra 0,7129 başarı oranına, ardından tekrar uygulanan boyut azaltma ve Temel Bileşen Analizi sonrasında Naive Bayes ile birlikte 0,7914 başarı seviyesine ulaşılmıştır. Bu da bahsi geçen saldırı verileri kullanılarak eğitilen saldırı tespit sistemi aktif ve çalışıyor konumda iken, gelen saldırıları %79 oranında doğru tespit edebildiğini göstermiştir.

***Anahtar Kelimeler**: Saldırı Tespit Sistemleri, Makine Öğrenmesi, Destek Vektör Makinesi, Naif Bayes.*

Abstract

As technology advances and the link between people and machines grows, system and data security become more important. Attackers try to find gaps by examining systems and sometimes succeed. Successful attacks lead to material and moral damages. Anti-virus or firewalls are used to prevent them. Anti-virus and firewalls may not always provide an effective defense against expert attackers. Based on these and similar problems, intrusion detection systems have been developed. They do this by collecting information from various systems and network resources and then analyzing the data for possible security issues. This study focuses on these problems and aims to train an intrusion detection system using machine learning techniques, known attack types, and data from server-based attack methods. In this direction, the data set was created by combining CesarFTP, WebDAV, Icecast, Tomcat, OS SMB, OS Print Spool, PMWiki, Wireless Karma, PDF N, Backdoored Executable, Browser Attack, Infectious Media attack data. The resulting data set was classified and trained using the Support Vector Machine (DVM) and Naive Bayes (NB), and the results were shared. Following the training and testing of the system with DVM, the success rate of 0.7129 was achieved, followed by the re-applied size reduction and Principal Component Analysis with Naive Bayes and the success level of 0.7914. This showed that the intrusion detection system, which was trained using the aforementioned intrusion data, was able to detect 79 percent of incoming attacks accurately while it was active and operational.

***Keywords:** Intrusion Detection Systems, Machine Learning, Support Vector Machine, Naive Bayes.*

* Arş. Gör. İstanbul Aydın Üniversitesi, Mühendislik Fakültesi, Yazılım Mühendisliği Bölümü, cagdasozer@aydin.edu.tr, ORCID: 0000-0002-0581-7955

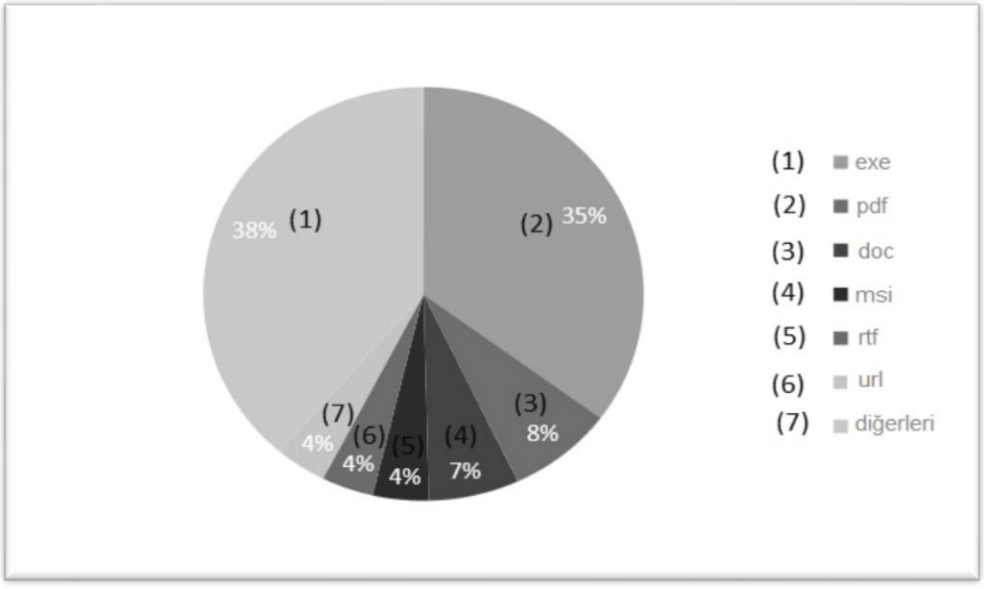
** Arş. Gör. İstanbul Aydın Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, mustafatakaoglu@aydin.edu.tr, ORCID: 0000-0002-1634-2705

Geliş Tarihi / Received: 26.04.2019
Kabul Tarihi / Accepted: 18.06.2019

1. GİRİŞ

Bilgi sistemleri ve ağlar elektronik saldırılara maruz kalabilirler. Bilgi güvenliğini ihlal etme girişimleri her gün, internette yaygın olarak bulunan güvenlik açığı değerlendirme araçlarının yanı sıra ticari olarak da kullanılabilen bu araçların kullanılabilirliği ile birlikte artmaktadır(Urmila ve Balasubramanian, 2019). SubSeven, Nmap, L0ftCrack, BackOrifce gibi araçların tümü sistemleri taramak, tanımlamak, araştırmak ve delmek için kullanılabilir. Ağları korumak için güvenlik duvarları ve çeşitli anti virüsler kullanılır. Peki, bunlar ne kadar yeterlidir? Güvenlik duvarları sizleri dışarıdan gelen saldırılara karşı çok iyi koruyabilir ama eğer sisteminizin içinde bir açık var ise uyarma olasılığı bulunmaz.”Script Kiddie” diye tabir edilen kişiler, interneti sürekli olarak alt ağlar tarafından yapılan taramalar dâhil, bilinen hatalara karşı tarar. Bazı zamanlarda ise, tamamen yasal yollarla rakip bir firma, rekabet avantajı elde etmek için personeller istihdam eder ve oluşturdukları bu ekiple sisteminize sızmaya çalışırlar. Saldırı tespit sistemlerinin önemi belirttiğimiz bu sebepten ötürü daha iyi anlaşılmaya başlamıştır. Söz konusu sistemlerin kullanımının bilgi teknolojilerinde %60’dan fazla olduğu 2012 yılında yapılan bir ankette belirtilmiştir (Richardson, 2011). Aktif bir çalışma konusu olmasına ve üzerinde çok sayıda insanın araştırma ve geliştirme yapmasına rağmen, bu tarz saldırıları etkin bir şekilde tespit edebilecek, standartları belirlenmiş bir sistem henüz geliştirilememiştir (Breitenbacher vd., 2019).

Araştırma toplulukları, sistem saldırılarına karşı istatistik tabanlı, imza tabanlı, davranış tabanlı ve karma tabanlı teknikler kullanarak savunmaya çalışmışlardır (Kraur ve Singh, 2014). Bu tekniklerin her birinin öncelikli hedefi, sistem istismarının gerçek zamanlı veya mümkün olduğunca gerçek zamana yakın olarak tanımlamak ve saldırının neden olduğu hasarı ortadan kaldırmak veya en aza indirmek için belli başlı saldırıları karantina altına almaktır (Kraur ve Singh, 2014). Şekil 1’de, 2018 yılında siber saldırıların daha çok hangi dosya biçimlerini kullandığı gösterilmiştir (cp<r>, 2018).

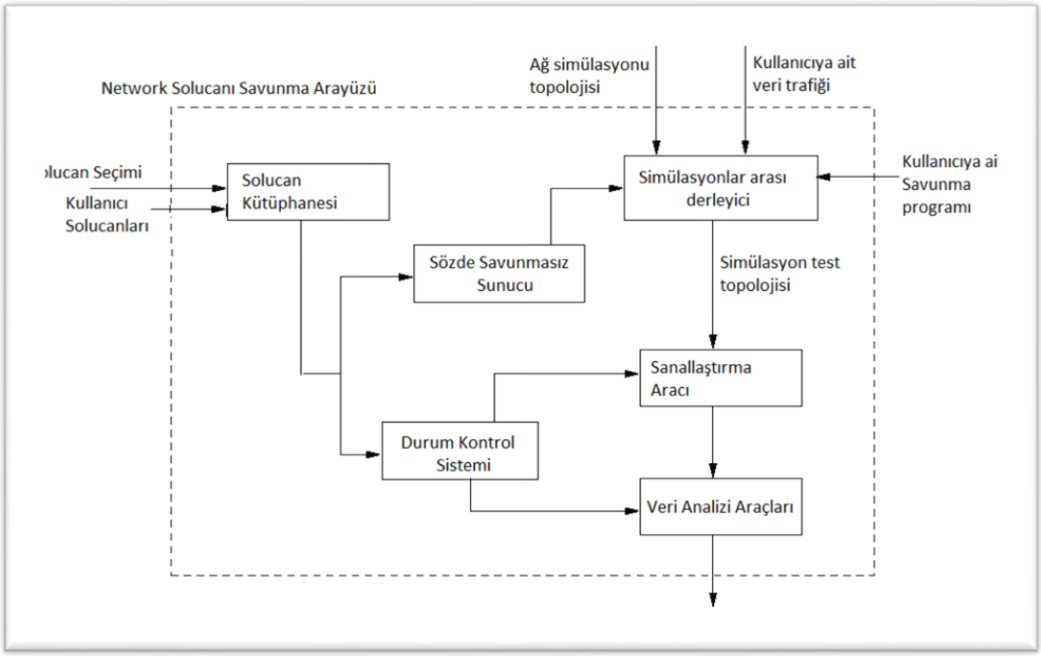


Şekil 1. Kötü Amaçlı Dosya Türleri Dağılımı (cp<r>, 2018).

Çalışmamızın amacı, var olan sunucu tabanlı saldırı çeşitleri ve bunların verilerini kullanarak saldırı tespit sistemini eğitmek, bu şekilde sistemi yeni gelebilecek saldırılara karşı korumaya çalışmaktır. Sistemin her an her çeşit saldırıya karşı koyması hem bellek yönetimi hem de güç tüketimi açısından akıllıca olmayacaktır dolayısıyla gelen saldırının doğru ve etkin bir şekilde sınıflandırılması gerekmektedir. Bu fikirden yola çıkılarak sınıflandırma algoritmaları kullanılmıştır. Saldırının doğru tahmininden sonra bu saldırıya yönelik bir savunma sisteminin devreye girmesi, sistemi amacına ulaştıracak ve minimum hasar ile saldırıyı atlatmayı sağlayacaktır.

2. LİTERATÜR TARAMASI

Saldırı profillerinin statik doğasından ötürü, algılama teknikleri ortamdaki zamana bağlı değişikliklere uyum sağlayamamaktadır (Aygün, 2017). Saldırı ve saldırı araçları, büyük hacimli ağ trafiği verileri, dinamik ve karmaşık davranışlar ve yeni tür saldırılarla daha karmaşık hale gelmiştir (Idhammada vd., 2018). Veri modelindeki herhangi bir değişiklik için sistem sürekli eğitim ile güncellenmiş bir profile ihtiyaç duymaktadır (Aygün, 2017). Yeni gözlemleri (yeni saldırılar) değerlendirmek için limiti (veya saptama parametrelerini) ayarlamak, tespitin kalitesi üzerinde önemli bir etkiye sahip olduğu için istatistiksel bir algılama yaklaşımı tasarlamada kritik bir adımdır (Yıldırım, 2014). Eğer eşik değeri çok dar ise, sık sık yanlış pozitif alarmlarla sonuçlanacak şekilde aşılır ve çok geniş olursa da limit asla aşılmayacak ve birçok yanlış negatif alarm ortaya çıkacaktır (Yıldırım, 2014). Zaman zaman, algılama parametreleri yeni saldırıları tespit etmek için elle çıkarılır veya ayarlanır. Tüm bu faktörler, çevrimdışı modunda çalışmak için istatistiksel algılama yaklaşımlarını sınırlar. Ve böylece, gerçek zamanda anlık algılama ve koruma için kullanılamazlar.



Şekil 2. Polimorfik Solucanlara Karşı Örnek Savunma (Cheetancheri, 2007).

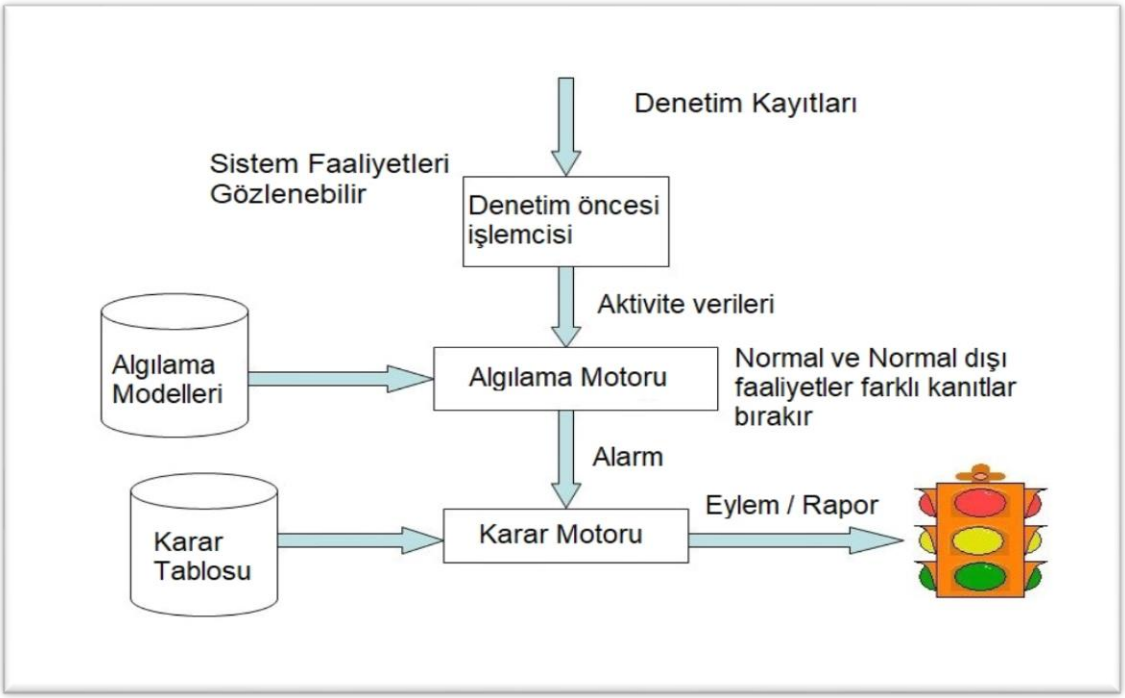
İmza temelli tespit teknikleri, esas olarak polimorfik solucanlara odaklanmaktadır. Şekil 2'de bilinen solucan türlerine karşı sistemi test etmeye dair örnek bir şekil gösterilmiştir. Bu uygulamanın sebebi ise saldırı tespit sistemlerinin tasarlanmasında, genellikle yaşanmış saldırıların felaket senaryolarının temel alınıyor olmasıdır (cp<r>, 2018). Bu saldırıların karakteristik özelliklerini incelemeden, yeni gelebilecek saldırılara karşı bir sistem tasarlanması oldukça zor olacaktır. Günlük hayatta kullandığımız anti virüs programları bile, hangi tür olduğundan bağımsız, yüklü olduğu cihazı savunmak için imzalara ihtiyaç duyar (Rozenblum, 2001). Üç tür imza vardır. Bunlar; içerik tabanlı, semantik tabanlı ve güvenlik açığı tabanlı imzalardır. İçerik tabanlı imzalar, bir solucan uygulamasına özgü özellikleri yakalar bu nedenle yeterince genel olmayabilir ve diğer istismarlar tarafından kaçırılabilir. Semantik tabanlı imzalar, alt dizelere dayalı yaklaşımlarla karşılaştırıldığında, hesaplama açısından pahalıdır. Dahası, Snort gibi mevcut STS'lerde (Saldırı Tespit Sistemi) uygulanamazlar. Güvenlik açığı tabanlı imzalar, solucanın yararlandığı ve üretilmesi zor olan güvenlik açığının özelliklerini yakalar.

Ağ anormallik tespitinde uygun bir çözüm sağlamak için normallik kavramına ihtiyaç duyulur. Normal düşüncesi, genellikle sistem dinamiklerinde yer alan temel değişkenler arasındaki

ilişkileri ifade eden resmi bir model tarafından ortaya konulur. Normallik modeli tarafından belirtilen sistemin profiline veya davranışına göre sapma derecesi yeterince yüksekse, bir olay veya nesne anormal olarak tespit edilir (Yıldırım, 2014). Örneğin, denetimli bir yaklaşım kullanan bir anormallik tespit sistemi S alalım. Bir S = (M, D) çifti olarak düşünülebilir; burada M, sistemin normal davranış modelidir ve D, bir etkinlik kaydı verildiğinde, bir sapma derecesine sahip olan bir yaklaşım ölçüsüdür (Yıldırım, 2014). Faaliyetler M modeline göre yapılmıştır. Böylece, her sistem esas olarak iki modüle sahiptir:

- Modelleme Modülü
- Algılama Modülü.

Modelleme Modülü; normallik modelini (M) elde etmek için sistemleri eğitir. Elde edilen model daha sonra yeni olay veya nesneleri veya anormal veya aykırı olarak trafiği değerlendirmek için algılama modülü tarafından kullanılır (Yıldırım, 2014). Olayların veya nesnelerin anormal veya aykırı olarak sınıflandırılmasına izin veren sapmanın ölçümüdür. Özellikle, modelleme modülü, dinamik senaryolarla başa çıkmak için uyarlamalı olmalıdır (Yıldırım, 2014).



Şekil 3. Saldırı Tespit Sistemi Bileşenleri (Intrusion Detection/Prevention Systems, 2016).

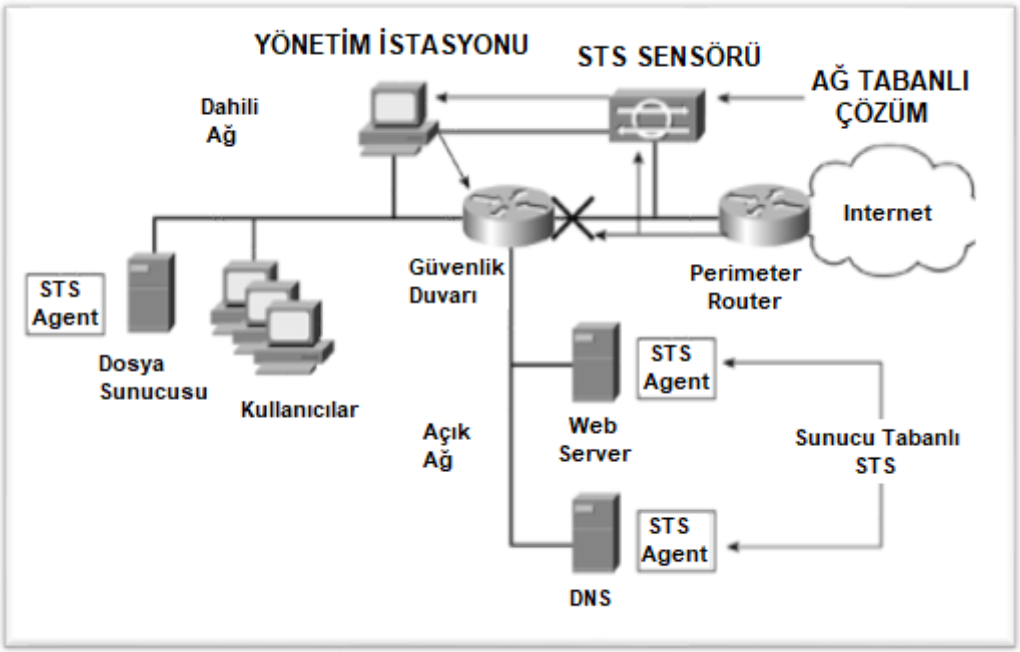
Saldırı tespit sistemleri insan yaşamının önemli bir parçası haline gelmektedir(Jabbar, Aluvalub, Reddy, (2017). İzinsiz giriş, bilgisayar ve ağ bileşenlerinin güvenliğini gizlilik, bütünlük ve kullanılabilirlik açısından tehlikeye atmayı amaçlayan bir dizi eylemdir (Aygün, 2017). Güvenlik mekanizmasının yetkisiz bir şekilde girişini ve kontrolünü sağlamak için içeriden veya dışarıdan bir saldırgan tarafından yapılabilir. Ağ sistemlerinin altyapısını korumak için, saldırı tespit sistemleri (STS'ler), olası bir güvenlik ihlalini tespit etmek için bir ev sahibi veya bir ağdaki çeşitli alanlardaki bilgileri toplayan ve analiz eden sağlam mekanizmalar sağlar (Staniford-Chen, Cheung, Crawford, Dilger, Frank, Hoagland, Levitt, Wee, Yip, Zerkle, 1999). Bir saldırı tespit sistemi iki temel fonksiyondan oluşur. Bunlar: şüphelilere ilişkin denetim verilerini toplamak ve denetim verilerini analiz etmektir (Wu, Song, Lin, Aurelle, Liu, Ding, Song, Moon (2018). Şekil 3'te, herhangi bir saldırı tespit sisteminin varsayılan olarak sahip olması gereken bileşenler gösterilmiştir.

İzinsiz giriş algılama işlevleri:

- Kullanıcı, sistem ve ağ etkinliklerini izleme ve analiz etme

- Olası güvenlik açıkları raporlarının oluşturulması için sistemleri yapılandırma
- Sistemi ve dosya bütünlüğünü değerlendirme
- Tipik saldırıların kalıplarını tanıma
- Anormal aktiviteyi analiz etme
- Kullanıcı politikası ihlallerini izleme

STS, ana makinenin veya ağın güvenliğini değerlendirmek için güvenlik açığı değerlendirmesini kullanır (cp<r>, 2018). Saldırı tespiti, hırsızlık faaliyetlerinin normal sistem aktivitelerinden belirgin şekilde farklı olduğu ve dolayısıyla saptanabilir olduğu varsayımı üzerine çalışır (Staniford-Chen, vd, 1999). STS'ler, gerçek zamanlı çalışmasına yakınlığına göre iki sınıfa ayrılabilir. Bunlardan ilki "Ana Bilgisayar Tabanlı Saldırı Tespit Sistemi", ABTSTS'dir. İkinci sınıf ise Ağ Tabanlı Saldırı Tespit Sistemi, ATSTS olarak adlandırılır (Kraur ve Singh, 2014).



Şekil 4. Sunucu Tabanlı STS ("IDS Introduction",2008).

Sunucu tabanlı STS, harici ara yüzler yerine bilgisayar sisteminin içyapısını izler ve analiz eder. Sunucu tabanlı STS hangi programın hangi kaynaklara eriştiği ve gayri meşru erişim girişiminde bulunduğu gibi iç faaliyetleri tespit edebilir (Bilge ve Dumitras, 2012). Benzer şekilde, bir sistemin durumunu ve depolanan bilgilerini, RAM veya dosya sistemindeki günlük dosyalarında veya başka bir yerde olup olmadığına bakabilir (Bilge ve Dumitras, 2012). Şekil 4'te sunucu tabanlı STS mekanizmasına dair örnek gösterilmiştir. Sunucu tabanlı STS, iç veya dış herhangi bir şeyin, işletim sisteminin yürürlüğe koyduğu güvenlik politikasını engelleyip engellemediğini izleyen bir aracı olarak düşünülebilir (Kraur ve Singh, 2014).

Ağ tabanlı STS; Ağ verilerindeki istenmeyen ve kötü niyetli izinsiz girişleri tespit etmeye yarar (Shenfield, Day ve Ayesh, 2018). Müdahaleler tipik olarak anormal şablonlar olarak ortaya çıkar ancak belirli teknikler verileri sıralı olarak modellemekte ve anormal alt dizileri tespit etmektedir (cp<r>, 2018). Bu anormalliklerin başlıca nedeni, bilgi çalmak ya da ağı bozmak için ağa yetkisiz erişim sağlamak isteyen dış saldırganların başlattığı saldırılardır. Tipik bir ortamda, bir ağ dünyanın geri kalanına internet üzerinden bağlanır (Kraur ve Singh, 2014). Network tabanlı STS, tüm gelen paketleri veya akışları okur, şüpheli kalıpları bulmaya çalışır. Örneğin, çok sayıda farklı bağlantı noktasında çok sayıda "Geçiş kontrol protokolü" bağlantı isteği çok kısa bir sürede gözlenirse, ağdaki bazı bilgisayarlarda bir kişinin bir "bağlantı noktası taraması" yaptığını

varsayabiliriz. Port taramaları çoğunlukla, gelen shell kodlarını, sıradan bir saldırı tespit sisteminin yaptığı şekilde algılamaya çalışır (cp<r>, 2018). Gelen trafiği teftiş etmenin yanı sıra, bir Network tabanlı STS de giden veya yerel trafikten izinsiz giriş hakkında değerli bilgiler sağlar (Staniford-Chen, vd, 1999). Algılama mekanizmasına göre üç tip olarak sınıflandırılabilir. Bunlar suiistimal tabanlı, anomali tabanlı ve bunların birleşimi olan karışık ataklardır (Kraur ve Singh, 2014).

Sistemler ve uygulamalar, güvenliğin asla önemli bir sorun olmadığı bir ortamda çalışmayı hedeflemiştir (Bilge ve Dumitras, 2012). Bununla birlikte, mevcut ağ senaryosunda kullanıldığında aynı sistemler ve uygulamalar, büyük güvenlik sorunlarından biri haline gelir. Örneğin, bir sistem izole edildiğinde mükemmel bir şekilde güvenli olabilir, ancak Internet'e bağlı olduğunda savunmasız hale gelir (Kraur ve Singh, 2014). Saldırı tespiti, bu sistemlere karşı saldırıların tanımlanmasını ve böylece saldırılara yanıt vermesini sağlar (Staniford-Chen, vd, 1999). İkincisi, bilgi güvenliği ve yazılım mühendisliği uygulamalarının kısıtlamaları nedeniyle, bilgisayar sistemleri ve uygulamalarında, bir saldırgan tarafından sistemlere veya uygulamalara saldırmak için kullanılabilecek tasarım kusurları veya hataları olabilir (Bilge ve Dumitras, 2012).

İzinsiz giriş tespiti konusunda birçok çalışma yapılmıştır ve sınıflandırma ve kümeleme gibi çoklu makine öğrenme algoritmalarından veya bunları bir özellik seçimi yaklaşımı ile birlikte farklı fikirlerle birleştirerek kullanılmıştır (Mazini, 2018). Sistemlerde belirli önlemler alındığı zaman riskler en aza indirilebilir. Sistem veya ağ üzerindeki şüpheli girişler, hesaplara gereğinden fazla başarısız yanlış girişler, neden açıldığı belli olmayan kullanıcı hesapları, açıklanamayan yeni dosyalar, bilinmeyen dosya isimleri, özellikle sistem dosyalarında dosya adlarına ve/veya tarihlerinde değişiklikler, ağ üzerindeki beklenmeyen trafik artışları, kullanıcı hesaplarındaki anormal hareketler (yetkinin olmadığı yerlere ısrarlı girmeye çalışmalar), sistem üzerindeki yetki noktalarına ve önemli noktalarına erişim istekleri gözlemlenerek bu tür saldırılar gerçek zamanlı veya gerçek zamanlıya oldukça yakın bir zaman diliminde bulunabilir ve engellenebilir.

Son olarak bilgi sistemleri kaynakları üzerindeki saldırgan etki setinin, heterojen bir parametrik ortamdaki değerler arasında birçok anomaliye yol açtığı not edilmelidir (Zhumangaliyeva vd., 2019).

3. YÖNTEM VE UYGULAMA

3.1. Veri

Çalışmamızda kullanılacak veri seti; CesarFTP, WebDAV, Icecast, Tomcat, OS SMB, OS Print Spool, PMWiki, Wireless Karma, PDF N, Backdoored Executable, Browser Attack, Infectious Media saldırı verileri birleştirilerek oluşturulmuştur. Ortaya çıkan bu veri seti ise Destek Vektör Makinesi (DVM) ve Naive Bayes (NB) kullanılarak sınıflandırılmış ve eğitilmiştir.

3.2. Yöntem

Öncelikle kullanılacak veri setlerimiz Düşük Varyans Filtresi, Yüksek Korelasyon Filtresi ve Temel Bileşen Analizi yöntemlerine tabi tutulduktan sonra eğitime hazır bir hale getirilmiştir. Kullanılan yöntemler aşağıda açıklanmıştır.

3.2.1. Düşük Varyans Filtresi: Veri setinde küçük değişiklikler olan veri sütunları çok az bilgi taşır. Böylece, belirli bir eşikten daha düşük varyansa sahip tüm veri sütunları kaldırılır. Dikkatli olunması gereken nokta ise varyans aralığa bağlıdır; bu nedenle, bu tekniği uygulamadan önce normalizasyon uygulanması gereklidir.

3.2.2. Yüksek Korelasyon Filtresi: Çok benzer eğilimlere sahip veri sütunlarının da çok benzer bilgiler taşıması olasıdır. Bu durumda, yalnızca biri makine öğrenme modelini beslemek için yeterli olacaktır. Burada nümerik sütunlar arasında ve sırasıyla Pearson'un Ürün Moment Katsayısı ve

Pearson chi kare değeri olarak nominal sütunlar arasındaki korelasyon katsayısı hesaplanır. Korelasyon katsayısı eşikten daha yüksek olan sütun çiftleri yalnızca bir taneye düşürülür. Korelasyon ölçeğe duyarlı olduğundan, anlamlı bir korelasyon karşılaştırması için sütun normalizasyonu gereklidir.

3.2.3. Temel Bileşen Analizi (TBA) : Bir veri setinin orijinal n koordinatlarını ortogonal olarak asıl bileşenler adı verilen yeni bir n koordinat grubuna dönüştüren istatistiksel bir prosedürdür. Dönüşümün bir sonucu olarak, ilk ana bileşen olası en büyük varyansa sahiptir; müteakip her bileşen, önceki bileşenlerle ortogonal olduğu (yani onlarla korelasyon olmayan) kısıtlama altında mümkün olan en yüksek varyansa sahiptir. Yalnızca ilk m <n bileşeninin tutulması, veri bilgisinin çoğunu, yani verideki değişimi korurken veri boyutsallığını azaltır. TBA dönüşümünün orijinal değişkenlerin göreceli ölçeklendirilmesine duyarlı olduğuna dikkat edilmesi gerekir. TBA uygulanmadan önce veri sütunu aralıklarının normalleştirilmesi gerekir.

3.2.4. Destek Vektör Makinesi (DVM) : Destek vektörleri, hiper düzlemine daha yakın olan ve hiper düzlemin konumunu ve yönünü etkileyen veri noktalarıdır. Bu destek vektörlerini kullanarak, sınıflandırıcının marjını maksimize ederiz. Destek vektörlerini silmek, hiper düzlemin konumunu değiştirir. Lojistik regresyonda, lineer fonksiyonun çıktısı alınmıştır ve değeri sigmoid fonksiyonunu kullanarak [0,1] aralığında yayılmıştır. Eğer yayılan değer bir eşik değerden (0.5) büyükse, ona etiket 1, yoksa etiket 0 atanmıştır. DVM'de doğrusal fonksiyonun çıktısı alınmıştır ve bu çıktı 1'den büyükse, bir sınıfla ve eğer çıktı -1 ise, başka bir sınıfta olduğu tespit edilmiştir. DVM'de eşik değerleri 1 ve -1 olarak değiştirildiğinden, marjini etkileyen bu takviye değerleri aralığı [-1,1] olarak elde edilmiştir. Gerekli özellikler çıkarılmış ve eğitim ve test verilerine ayırılmıştır. Verilerin %90'ı eğitim için, geri kalan %10'u test için kullanılmıştır. Python dili, Numpy kitaplığı kullanılarak DVM modeli oluşturulmuştur. epochs = 1 ve alpha = 0.0001 kullanılarak ilerlenmiştir. Kullanılan kütüphaneler ve DVM standart tanımlamalarından bir örnek, Şekil 5'te gösterilmiştir.

```
import numpy as np
from numpy import linalg
import cvxopt
import cvxopt.solvers

def linear_kernel(x1, x2):
    return np.dot(x1, x2)

def polynomial_kernel(x, y, p=3):
    return (1 + np.dot(x, y)) * p

def gaussian_kernel(x, y, sigma=5.0):
    return np.exp(-linalg.norm(x-y)**2 / (2 * (sigma ** 2)))
```

Şekil 5. Kullanılan Python Kütüphaneleri ve DVM tanımlamaları (Blondel, 2019).

Oluşturduğumuz veri seti, doğrusal olmayan bir yapıya sahip olduğu için bu yapıya sahip bir fonksiyon ile test edilmiştir. Kullanılan fonksiyon, Şekil 6'da gösterilmiştir.

```
def test_non_linear():
    X1, y1, X2, y2 = gen_non_lin_separable_data()
    X_train, y_train = split_train(X1, y1, X2, y2)
    X_test, y_test = split_test(X1, y1, X2, y2)

    clf = SVM(gaussian_kernel)
    clf.fit(X_train, y_train)

    y_predict = clf.predict(X_test)
    correct = np.sum(y_predict == y_test)
    print "%d out of %d predictions correct" % (correct, len(y_predict))

    plot_contour(X_train[y_train==1], X_train[y_train==-1], clf)
```

Şekil 6. Doğrusal Olmayan Veriler için Test Fonksiyonu (Blondel, 2019).

3.2.5. Naif Bayes: Naif Bayes modeli, eğitim veri setindeki verilerin bir özetinden oluşur. Bu özet daha sonra tahminlerde bulunurken kullanılır. Toplanan eğitim verilerinin özeti, her özniteliğin sınıf değerine göre ortalamasını ve standart sapmasını içerir. Örneğin, iki sınıf değeri ve 7 sayısal özellik varsa, o zaman her bir özellik (7) ve sınıf değeri (2) kombinasyonu için ortalama ve standart bir sapmaya yani 14 özellik özetine ihtiyaç vardır. Bunlar, her bir sınıf değerine ait belirli özellik değerlerinin olasılığını hesaplamak için tahminlerde bulunurken gereklidir. Bu özet verilerin hazırlanması şu alt görevlere ayırabilir: Verileri sınıflara göre ayırma, hesap ile ortalama, standart sapmayı hesaplama, veri kümesini özetleme ve sınıflara göre öznitelikleri özetleme.

Ardından eğitim verilerinden hazırlanan özetler kullanılarak tahminler yapılabilir. Öngörüler yapmak, verilen bir veri örneğinin her sınıfa ait olma olasılığını hesaplamayı ve ardından tahmin olarak en büyük olasılık olan sınıfı seçmeyi içerir. Bu şu şekilde ayrılabilir: Gauss olasılık yoğunluğu fonksiyonun hesaplama, sınıf olasılıklarını hesaplama, tahmin yapma ve doğruluk tahmini hesaplama. Son olarak, test veri setindeki her veri örneği için tahminler yaparak modelin doğruluğu tahmin edilebilir. GetPredictions() bu tahmini yapar ve her test örneği için bir tahmin listesi döndürür. Tahminler, test veri setindeki sınıf değerleri ile karşılaştırılabilir ve bir sınıflandırma doğruluğu, %0 ve %100 arasında bir doğruluk oranı olarak hesaplanabilir. GetAccuracy() bu doğruluk oranını hesaplar. Naif Bayes algoritmasının kullanılan fonksiyonlardan birkaçı, Şekil 7'de gösterilmiştir.

```
#Separate by Class
def separateByClass(dataset):
        separated = {}
        for i in range(len(dataset)):
                vector = dataset[i]
                if (vector[-1] not in separated):
                        separated[vector[-1]] = []
                separated[vector[-1]].append(vector)
        return separated

#Test separating by class
"""
dataset = [[1,20,1],[2,21,0],[3,22,1]]
separated = separateByClass(dataset)
print('Separated instances: {0}'.format(separated))
"""

#Calculate Mean
def mean(numbers):
        return sum(numbers)/float(len(numbers))

def stdev(numbers):
        avg = mean(numbers)
        variance = sum([pow(x-avg,2) for x in numbers])/float(len(numbers)-1)
        return math.sqrt(variance)

#Test stdev & mean calculation
"""
numbers = [1,2,3,4,5]
print('Summary of {0}: mean={1}, stdev={2}'.format(numbers, mean(numbers), stdev(numbers)))
"""
```

Şekil 7. Naive Bayes Örnek Fonksiyonları (Askaruly, 2019).

DVM algoritmasının seçilmesinin sebebi: DVM'nin net bir ayrılma marjı ile gerçekten iyi çalışması, yüksek boyutlu alanlarda etkili olması, boyut sayısının örnek sayısından daha büyük olduğu durumlarda etkili olması, karar işlevinde de (destek vektörleri denir) bir eğitim noktaları alt kümesi kullanması, bu nedenle de belleği etkin bir şekilde kullanmasıdır. Naif Bayes algoritmasının tercih edilme sebebi ise, destek vektör makinesi ile birlikte kullanıldığında nasıl bir sonuç vereceğini izlemek ve daha iyi bir sınıflandırma yapabiliyor mu onu test etmektir.

4. BULGULAR

Çalışmamızda oluşturduğumuz veri kümemize ilk olarak Destek Vektör Makinesi algoritması uygulanmış, 6013 tane veri içerisinden 4287 tanesi doğru tahmin edilirken 1726 tanesi yanlış tahmin edilmiştir. Başarı oranı ~= 0.71 olmuştur. Elde edilen sonuçlar ayrıca Tablo 1'de paylaşılmıştır.

Tablo 1. DVM Algoritması Başarı Oranı

```
# Başarı Oranı
accuracy = accuracy_score(y_test, result)
# Sonuç : 0,712938192283729182
print(accuracy)
```

Başarı oranını arttırmak için ikinci bir sınıflandırma algoritması kullanılmak istenmiştir. Var olan veri setimiz boyut azaltma ve temel bileşen analizine tekrar tabi tutulmuştur ve ardından eğitim süresini azaltmak amacıyla veri setimizde benzer eğilime sahip test verileri ayrılmıştır. Bu işlemler sonrasında Naive Bayes algoritması uygulanmıştır. Özetlemek gerekirse, Naive Bayes bütün koşullu olasılıkların çarpımı olarak düşünülebilir. Elde edilen sonuç Tablo 2'de paylaşılmıştır.

Tablo 2. Naive Bayes Algoritması Başarı Sonucu

```
# Başarı Oranı
predictions = getPredictions(summaries, testSet)
accuracy = getAccuracy(testSet, predictions)
# Sonuç : 0, 791428501394022834
```

Naive Bayes algoritmasının uygulanmasından sonra elde edilen sonucu incelediğimizde, başarı oranı ~= 0,79 olmuştur. Naif Bayes algoritmasının hızlı çalışan istekli bir sınıflandırma algoritması olması, gerçek zamana yaklaşabilmesi ve aynı zamanda çok sınıflı tahmin özelliği için iyi çalışması özelliği ile tercih sebebi olmuştur. Zira çalışmamızda hem savunma sistemleri hem de kullandığımız veri seti olsun, çok sınıf bulunmaktadır.

5.SONUÇ

Zarara sebebiyet verebilecek saldırılara karşı koymak için kullanılabilecek sistemler ve bu sistemi izleyen insanların gözünden kaçabilecek şeyler olduğu da düşündüğünde, risk faktörünü en aza indirgeyecek olan şey bu kontrolleri makineye bırakmaktır. Bunun olması için sistemlere felaket senaryoları düzenlenmeli, daha önceden yapılmış olan saldırılara karşı dayanıklı olup olmadığı test edilmeli, yazılan yazımların güvenlik faktörleri de düşünülerek doğru bir şekilde planlanmaları ve ona göre geliştirilmeleri gerekmektedir. Makine öğrenmesi yöntemleri kullanılarak devamlı olarak izlenmesi gereken sistemler otomatik hale getirilebilir ama bu sistem kesinlikle güvenli çalışıyor demek değildir. Zira daha önceden kullanılmamış bir yöntem bulan saldırgan(lar) bir sistem açığından faydalanıp zarara sebebiyet verebilir. Sistemler bu yüzden devamlı olarak geliştirilmeli, güncel teknoloji haberleri de takip edilip kullanılan sistemler de güncellenmelidir.

DVM sınıflandırıcılar, Naïve Bayes algoritmasına kıyasla daha iyi doğruluk ve daha hızlı tahmin oranlarıyla çalışırlar. Karar aşamasında daha az hafıza kullanılır çünkü bir alt eğitim noktası kullanılmaktadır. DVM net bir ayrılma marjı ve yüksek boyutlu boşluk ile iyi çalışır. DVM, fazla eğitim süresi nedeniyle büyük veri kümeleri için uygun değildir. Ayrıca eğitimlerde Naive Bayes ile karşılaştırıldığında daha fazla zaman almaktadır. Örtüşen sınıflarla zayıf çalışır ve kullanılan çekirdek türüne de duyarlıdırlar.

Çalışmamız için gerçekleştirdiğimiz uygulamada, DVM algoritması; düşük varyans filtresi, yüksek ilgileşim filtresi ve temel bileşen analizi yöntemleri ile temizlediğimiz veri setlerimizle kullanılmıştır. Elde edilen veriler Naif Bayes yöntemine tabi tutulmuştur. DVM ile sistemin eğitilmesi ve test edilmesinden sonra 0,712938192283729182 başarı oranına ulaşılmış, ardından

tekrar uygulanan boyut azaltma ve TBA sonrasında Naif Bayes kullanılarak 0,791428501394022834 yani %79'luk başarı seviyesine ulaşılmıştır. Elde edilen %79'luk başarı seviyesi, saldırı tespit sistemleri için yüksek bir başarı oranıdır.

Başarı oranını ilerletmek için hassasiyet ve modellerin tekrar gözden geçirilmesi gerekir. Algoritmanın iyi veya kötü çalışması, tamamıyla verinin büyüklüğü, kalitesi ve niteliğine, kullanılabilir hesaplama süresine, görevin aciliyetine ve verilerle ne yapmak istendiğine bağlıdır. Bu yüzden herhangi bir algoritma için diğerlerine göre daha "iyidir" diye kesin bir söylemde bulunmak doğru olmaz.

Ayrıca anlaşılması gereken bir durum ise, saldırı tespit sistemlerinin savunma yapmaya tek başlarına yeterli olmadıklarıdır. Bu sistemlerin trafiği devamlı olarak izlenir ve sonuçları bir yöneticiye rapor edilir. Tespit edilen bir istismarın sistemi ele geçirmesini önlemek için otomatik olarak işlem yapılamaz. Saldırganlar, ağa girdikten sonra güvenlik açıklarından çok hızlı bir şekilde yararlanabilir ve bu sistemlerin müdahalesini sınırlandırabilirler.

Çalışmamızda paylaşmış olduğumuz sonuç, makine öğrenmesi yöntemleriyle eğitilen saldırı tespit sistemlerinin hem içeriden hem de dışarıdan gelebilecek benzer saldırılara karşı yüksek başarı oranıyla yakalayabildiğini göstermektir. Bu tip sistemlerin geliştirilmesindeki en büyük sorun, paylaşılmayan verilerdir. Sıfırıncı gün saldırıları gibi şirketlerin paylaşmaya yanaşmadığı kritik veriler, eğitim aşamasında kullanılmadığı için sistem bu tarz saldırılara karşı yanıt vermeye yetersiz kalmaktadır. Eğitim ve test aşamaları için daha fazla veri bulunması, başarı sonucunu mutlak suretle arttıracaktır.

Son olarak bu alanda yapılabilecek başlıca çalışmalar; hibrit saldırı tespit sistemlerinin geliştirilmesidir. Yeni geliştirilen saldırı tekniklerinin takip edilmesi ve önerilecek hibrit sistemlerde kullanılması, ayrıca en önemli sorunlardan biri olan saldırı verilerinin temin edilmesi ve verilerin kullanılabilir hale getirilmesinde yapılabilecek çalışmalara ihtiyaç bulunmaktadır.

KAYNAKÇA

Askaruly S. (2019). Naive Bayes From Scratch in Python Erişim Tarihi: https://gist.github.com/tuttelikz/94f750ef3bf14f8a126a.Rozenblum D. (2001). Understanding Intrusion Detection Systems. The SANS Institute Information Security Reading Room.

Aygün R. C. (2017). Derin Öğrenme Yöntemleri ile Bilgisayar Ağlarında Güvenliğe Yönelik Anormallik Tespiti. İstanbul, 2016-04-01-YL01, s.27. Erişim adresi: https://docplayer.biz.tr/64409341-Derin-ogrenme-yontemleri-ile-bilgisayar-aglarinda-guvenlige-yonelik-anormallik-tespiti-r-can-aygun-danisman-doc-dr-a.html

Bilge K., Dumitras T. (2012). An Empirical Study of Zero-Day Attacks In The Real World. CCS'12, October 16–18 Raleigh, North Carolina, USA. Copyright 2012 ACM 978-1-4503-1651-4/12/10.

Blondel M. (2019) Support Vector Machines Erişim adresi: https://gist.github.com/mblondel/586753/f740949d0336484567dd422fe53445ac8821f5b2

Breitenbacher D., Homoliak I., Aung Y. L., Tippenhauer N. O., and Elovici Y. (2019). HADES-IoT: A Practical Host Based Anomaly Detection System for IoT Devices. In ACM Asia Conference on Computer and Communications Security (AsiaCCS '19), July 9–12, 2019, Auckland, New Zealand. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3321705.3329847

Cheetancheri G. S. (2007). Collaborative defense against zero-day and polymorphic worms: detection, response and an evaluation framework. Submitted in partial satisfaction of the

requirements for the degree of Doctor of Philosophy in Computer Science in the Office of Graduate Studies of the University of California Davis.

Cp<r>, (2018). Check Point Cyber Attack Trends: Mid-Year Report 2018, Check Point Research Software Technologies.

Idhammada M., Afdela K., Belouchb M. (2018). Distributed Intrusion Detection System for Cloud Environments based on Data Mining techniques. ScienceDirect. Procedia Computer Science 127 (2018) 35–41.

IDS Introduction (2008). Erişim adresi: http://etutorials.org/Networking/Router+firewall+security/Part+VII+Detecting+and+Preventing+Attacks/Chapter+16.+Intrusion-Detection+System/IDS+Introduction/

Intrusion Detection/Prevention Systems (2016). Objectives and Deliverable Understand the concept of IDS/IPS and the two major categorizations: by features/models. Erişim adresi: https://slideplayer.com/slide/4982018/

Jabbar M. A., Aluvalub R., Reddy S. S. S. (2017). RFAODE: A Novel Ensemble Intrusion Detection System. ScienceDirect. Procedia Computer Science 115 (2017) 226–234.

Kraur R., Singh M. (2014). Efficient Hybrid Technique for Detecting Zero-Day Polymorphic Worms (978-1-4799-2572-8/14/$31.00_c IEEE).

Mazini, M., et al. (2018). Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and AdaBoost algorithms. Journal of King Saud University – Computer and Information Sciences (2018), https://doi.org/10.1016/j.jksuci.2018.03.011

Richardson R. (2011). Computer Crime and Security Survey. Computer Security Institute, CSI. Tech. report, p.25, figure 15.

Shenfield A., Day D., Ayesh A. (2018). Intelligent intrusion detection systems using artificial neural networks. ScienceDirect. ICT Express Volume 4, Issue 2, June 2018, Pages 95-99.

Staniford-Chen S., Cheung S., Crawford R., Dilger M., Frank J., Hoagland J., Levitt K., Wee C., Yip R., Zerkle D. (1999). Graph Based Intrusion Detection System For Large Networks. Department of Computer Science, University of California at Davis, CA 95616.

Urmila T. S. ve Balasubramanian R. (2019). Dynamic Multi-layered Intrusion Identification and Recognition using Artificial Intelligence Framework. International Journal of Computer Science and Information Security (IJCSIS), Vol. 17, No. 2, February 2019, ISSN 1947-5500

Wu M., Song J., Lin L. W. L., Aurelle N., Liu Y., Ding B., Song Z., Moon Y. B. (2018). Establishment of intrusion detection testbed for CyberManufacturing systems. ScienceDirect. Procedia Manufacturing 26 (2018) 1053–1064.

Yıldırım M. Z. (2014). Makine Öğrenmesi Yöntemleri ile Network Üzerinde Saldırı Tespiti. Yüksek Lisans Tezi, Karabük Üniversitesi, 902-1-014, s.5.

Zhumangaliyeva N., Korchenko A., Doszhanova A., Shaikhanova A., Gulmira S., Smagulov S. ve Erzhan S. (2019). Detection Environment Formation Method for Anomaly Detection Systems, International Journal of Mechanical Engineering and Technology, 10(3), 2019, pp. 220-235. http://www.iaeme.com/IJMET/issues.asp?JType=IJMET&VType=10&IType=3