TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

AÇIK KAYNAK SALDIRI TESPİT SİSTEMLERİNİN AÇIK KURAL SETLERİYLE ANALİZİ

YÜKSEK LİSANS TEZİ

İLAYDA GÜNDOĞDU

Bilgisayar Mühendisliği Anabilim Dalı

BİLGİ GÜVENLİĞİ

Tez Danışmanı: Prof. Dr. Ali Aydın SELÇUK

NİSAN, 2022

ÖZET

Yüksek Lisans Tezi

AÇIK KAYNAK SALDIRI TESPİT SİSTEMLERİNİN
AÇIK KURAL SETLERİYLE ANALİZİ

İlayda GÜNDOĞDU

TOBB Ekonomi ve Teknoloji Üniveritesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Ali Aydın SELÇUK

Tarih: Nisan 2022

İnternet teknolojilerinin gelişmesiyle birlikte siber saldırılarda da artış gözlenmektedir. Bunun sonucunda siber saldırıları tespit etmek ve önlemek amacıyla çeşitli sistemler geliştirilmiştir. Saldırıları tespit etmek amacıyla en çok kullanılan açık kaynak ve ağ tabanlı saldırı tespit sistemleri Snort ve Suricata iken, en çok kullanılan açık kaynak ve host tabanlı saldırı tespit sistemleri ise OSSEC ve Wazuh'tur.

Bu çalışmada, Snort ve Suricata'nın en son versiyonlarının varsayılan konfigürasyonlarla ve açık kural setleri kullanılarak saldırıları tespit etmedeki etkinliği karşılaştırmalı olarak analiz edilirken, OSSEC'in ve Wazuh'un ise host tabanlı saldırı tespit sistemleri olarak tespit etkinliği incelenmiştir. Tespit etkinliğini ölçmek için alarm üretilen atak sayılarının toplam atak sayısına oranı hesaplanmıştır. Çalışmada açık Kural Seti olarak Community Kural Seti, Registered Kural Seti, Talos Kural Seti, Emerging Threats Kural Seti ve Broadcom Kuralları kullanılmıştır. Deneylerde 600 farklı atak trafiği ve 38 farklı zararsız trafik kullanılarak saldırı tespit sistemlerinin tespit etkinliği ve yanlış pozitif alarm oluşturma miktarları analiz edilmiştir. Atak trafiği olarak web uygulama atakları, zararlı yazılım trafikleri ve CVE referanslı zafiyet sömürü trafikleri olmak üzere üç farklı kategoride trafiklere yer verilmiştir. Deneyler sırasında kural setleri tek başına kullanılarak, tüm kural setleri birlikte

kullanılarak, zararlı yazılım trafikleri tehdit aktörlerine göre analiz edilerek ve CVE referanslı zafiyet sömürü atakları platformlara ve işletim sistemlerine göre analiz edilerek Snort ve Suricata saldırı tespit sistemlerinin tespit etkinliklerinin ölçülmesi amaçlanmıştır. OSSEC ve Wazuh saldırı tespit sistemlerine ise web uygulama atak trafikleri ve CVE referanslı zafiyet sömürü trafikleri gönderilerek alarm oluşturma durumları analiz edilmiş, ardından başarılı bir atak sonrasında oluşacak alarmlar incelenmiştir.

Deneyler sonucunda atak trafiklerinde kural setleri ayrı olarak kullanıldığında en etkin sonuçların Talos Kural Seti ile alındığı, tüm kural setleri birlikte kullanıldığında ise Snort saldırı tespit sisteminin her üç atak türünde de atakları tespit etmede Suricata'dan daha etkin olduğu görülmüştür. Zararsız trafiklerde ise tüm kural setleri tek başına kullanıldığında Talos Kural Seti ve Emerging Threats Kural Seti'nin yanlış pozitif alarm oluşturmadığı ve zararsız trafiklerde tüm kural setleri birlikte kullanıldığında Suricata'nın daha az yanlış pozitif alarm oluşturduğu analiz edilmiştir. OSSEC ve Wazuh Saldırı Tespit Sistemleri'nin ise sızmayla sonuçlanmayan ataklarda alarm vermemekle birlikte, kullanıcıların zararsız komutlarından çok fazla yanlış pozitif alarm oluşturduğu gözlenmiştir.

Anahtar Kelimeler: Saldırı tespit sistemi, Snort, Suricata, OSSEC, Wazuh

ABSTRACT

Master of Science

ANALYSIS OF OPEN SOURCE INTRUSION DETECTION SYSTEMS WITH OPEN RULESETS

Ilayda GUNDOGDU

TOBB University of Economics and Technology
Institute of Natural and Applied Sciences
Department of Computer Engineering
Information Security

Supervisor: Prof. Dr. Ali Aydın Selçuk

Date: April 2022

With the development of internet technologies, there is an increase in cyber attacks. As a result, various systems have been developed to detect and prevent cyber attacks. The most widely used open source and network-based intrusion detection systems are Snort and Suricata, while the most widely used open source and host-based intrusion detection systems are OSSEC and Wazuh.

In this study, the efficiency of the latest versions of Snort and Suricata in detecting attacks with default configurations and using open rule sets is comparatively analyzed, while the detection efficiency of OSSEC and Wazuh as host-based intrusion detection systems are examined. In order to measure the detection efficiency, the ratio of the number of alarms generated to the total number of attacks was calculated. Community Rule Set, Registered Rule Set, Talos Rule Set, Emerging Threats Rule Set and Broadcom Rules were used as open rule set in the study. In the experiments, detection efficiency and false positive alarm generation amounts of intrusion detection systems were analyzed by using 600 different attack traffic and 38 different benign traffic. As attack traffic, there are traffic in three different categories: web application attacks, malware traffic and CVE referenced vulnerability exploitation traffic. During the experiments, it was aimed to measure the detection efficiency of Snort and Suricata

intrusion detection systems by using rule sets alone, using all rule sets together, analyzing malware traffic according to threat actors, and analyzing CVE-referenced vulnerability exploitation attacks according to platforms and operating systems. On the other hand, web application attack traffics and CVE-referenced vulnerability exploitation traffics were sent to the OSSEC and Wazuh intrusion detection systems, and alarm generation situations were analyzed, and then the alarms that would occur after a successful attack were examined.

As a result of the experiments, it has been seen that the most effective results are obtained with the Talos Rule Set when the rule sets are used separately in attack traffic, and when all rule sets are used together, the Snort intrusion detection system is more effective detecting attacks than Suricata in all three attack types. It has been analyzed that Talos Rule Set and Emerging Threats Rule Set do not create false positive alarms when all rule sets are used alone in benign traffic, and Suricata generates fewer false positive alarms when all rule sets are used together in benign traffic. It has been observed that the OSSEC and Wazuh Intrusion Detection Systems didn’t create alarms in unsuccessful attacks, but it generates too many false positive alarms from the usual commands of the users.

Keywords: Intrusion detection system, Snort, Suricata, OSSEC, Wazuh

İÇİNDEKİLER

Sayfa

ÖZET ..... **iii**
ABSTRACT ..... **v**
İÇİNDEKİLER ..... **viii**
ŞEKİL LİSTESİ ..... **xi**
ÇİZELGE LİSTESİ ..... **xiii**
KISALTMALAR ..... **xv**
1. GİRİŞ **1**
1.1. Tezin Amacı 1
1.2. Literatür Taraması 2
1.3. Tezin İçeriği 4
2. SALDIRI TESPİT SİSTEMLERİ **5**
2.1. Saldırı Tespit Sistemleri'ne Genel Bakış 5
2.1.1. Ortama göre saldırı tespit sistemleri 5
2.1.1.1. Ağ tabanlı saldırı tespit sistemi (NIDS) 5
2.1.1.2. Host tabanlı saldırı tespit sistemi (HIDS) 5
2.1.2. Tespit yöntemine göre saldırı tespit sistemleri 6
2.1.2.1. İmza tabanlı saldırı tespit sistemi 6
2.1.2.2. Anomali tabanlı saldırı tespit sistemi 6
2.2. Snort'a Genel Bakış 6
2.3. Suricata'ya Genel Bakış 6
2.4. OSSEC'e Genel Bakış 7
2.5. Wazuh'a Genel Bakış 7
2.6. Açık Kaynak Kural Setleri ve Kurallar 7
2.6.1. Snort Community kural seti 7
2.6.2. Snort Registered kural seti 7
2.6.3. Broadcom (web) kuralları 8
2.6.4. Talos kural seti 8
2.6.5. Emerging Threats kural seti 8
3. TEST ORTAMI **9**
3.1. Topoloji 9
3.2. Kullanılan Uygulamalar ve Araçlar 9
3.2.1. Metasploit framework 9
3.2.2. Wireshark 10
3.2.3. Dvwa (Damn Vulnerable Web Application) 10
3.3. Konfigürasyon 10
3.3.1. Snort3 – Suricata konfigürasyonu 10
3.3.2. Ossec konfigürasyonu 10
3.3.3. Wazuh konfigürasyonu 10
3.4. Snort3'te Trafiklerin Analizi 11
3.5. Suricata'da Trafiklerin Analizi 11
3.6. OSSEC ve Wazuh'ta Trafiklerin Analizi 11
4. TESTLER **13**
4.1. Metodoloji 13
4.1.1. Donanım özellikleri 13

4.1.2. Saldırı tespit sistemi versiyonları 14
4.1.3. Açık kaynak kural seti versiyonları 14
4.1.4. Kullanılan yöntem ve metrikler 14
4.2. Test 1: Zararlı Trafik 16
4.2.1. Snort 16
4.2.1.1. Test 1.1: Tüm kural setlerinin tek başına ve birlikte kullanılması 16
4.2.1.2. Test 1.2: Zararlı yazılımların tehdit aktörlerine göre analizi 18
4.2.1.3. Test 1.3: CVE referanslı atakların işletim sistemlerine göre analizi 19
4.2.1.4. Test 1.4: CVE referanslı atakların platformlara göre analizi 21
4.1.2. Suricata 22
4.2.2.1. Test 1.5: Tüm kural setlerinin tek başına ve birlikte kullanılması 22
4.2.2.2. Test 1.6: Zararlı yazılımların tehdit aktörlerine göre analizi 23
4.2.2.3. Test 1.7: CVE referanslı atakların işletim sistemlerine göre analizi 24
4.2.2.4. Test 1.8: CVE referanslı atakların platformlara göre analizi 26
4.2.3. Snort ve Suricata saldırı tespit sistemlerinin karşılaştırılması 27
4.2.3.1. Test 1.9: Tüm kural setlerinin tek başına ve birlikte kullanılması 27
4.2.3.2. Test 1.10: Zararlı yazılımların tehdit aktörlerine göre analizi 28
4.2.3.3. Test 1.11:CVE referanslı atakların işletim sistemlerine göre analizi 29
4.2.3.4. Test 1.12: CVE referanslı atakların platformlara göre analizi 31
4.2.4. Test 2: OSSEC ve Wazuh 32
4.3. Test 3: Zararsız Trafik 33
4.3.1. Snort ve Suricata saldırı tespit sistemlerinin karşılaştırılması 34
4.3.1.1. Tüm kural setlerinin tek başına ve birlikte kullanılması 34
4.4. Ağ Tabanlı Saldırı Tespit Sistemlerinde Hata Matrisi Analizi 35
4.4.1. Community kural seti 35
4.4.1.1. Snort IDS 35
4.4.1.2. Suricata IDS 35
4.4.2. Broadcom kuralları 35
4.4.2.1. Snort IDS 35
4.4.2.2. Suricata IDS 36
4.4.3. Registered kural seti 36
4.4.3.1. Snort IDS 36
4.4.3.2. Suricata IDS 37
4.4.4. Talos / Emerging Threats kural setleri 37
4.4.4.1. Snort IDS 37
4.4.4.2. Suricata IDS 37
4.4.5. Tüm kural setleri birlikte 38
4.4.5.1. Snort IDS 38
4.4.5.2. Suricata IDS 38
4.5. Kesinlik, Duyarlılık ve F-Skor Analizi 38
5. SONUÇ 43
5.1. Gelecek Çalışmalar için Öneriler 44
KAYNAKLAR 45
EKLER 49
Ek 1 : Snort3-Suricata Konfigürasyon Betiği 49
Ek 2 : OSSEC Konfigürasyonu 49
Ek 3 : Wazuh Konfigürasyonu 49

Ek 4 : Snort Otomasyon Betiği 49
Ek 5 : Suricata Otomasyon Betiği 52
Ek 6 : Broadcom Kuralları 54
Ek 7 : Testlerde Kullanılan Zararlı Trafikler 55
Ek 8 : Testlerde Kullanılan Zararsız Trafikler 68

ŞEKİL LİSTESİ

Sayfa

Şekil 3.1 : Topoloji 9
Şekil 4.1 : Snort - Kural setlerinin etkinliği 16
Şekil 4.2 : Snort - Tüm atak türlerinin birlikte analizi 17
Şekil 4.3 : Snort - Tehdit aktörlerine göre etkinlik analizi 19
Şekil 4.4 : Snort - İşletim sistemlerinde kural seti bazında analiz 19
Şekil 4.5 : Snort – Platformlarda kural seti bazında analiz 21
Şekil 4.6 : Suricata - Kural setlerinin etkinliği 22
Şekil 4.7 : Suricata - Tehdit aktörlerine göre etkinlik analizi 24
Şekil 4.8 : Suricata - İşletim sistemlerinde kural seti bazında analiz 25
Şekil 4.9 : Suricata - Platformlarda kural seti bazında analiz 26
Şekil 4.10 : Snort ve Suricata – Karşılaştırmalı tespit etkinliği 27
Şekil 4.11 : Snort ve Suricata - Tehdit aktörlerinin tespit etkinliği 29
Şekil 4.12 : Snort ve Suricata - İşletim sistemlerine göre tespit etkinliği 30
Şekil 4.13 : Snort ve Suricata - Platformlara göre tespit etkinliği 31
Şekil 4.14 : Kural setlerinin yanlış pozitif alarm oluşturma yüzdesi 34
Şekil 4.15 : Kesinlik, Duyarlılık ve F-Skor Karşılaştırmalı Grafiği 42
Şekil Ek.1 : Snort3-Suricata Konfigürasyon Betiği 50
Şekil Ek.2 : OSSEC Konfigürasyonu 50
Şekil Ek.3 : Wazuh Konfigürasyonu 50

ÇİZELGE LİSTESİ

Sayfa

Çizelge 2.1 : Kullanılan IDS türleri. 5
Çizelge 4.1 : Donanım Özellikleri. 13
Çizelge 4.2 : Saldırı Tespit Sistemi Versiyonları. 14
Çizelge 4.3 : Açık Kaynak Kural Seti Versiyonları. 14
Çizelge 4.4 : Hata Matrisi. 15
Çizelge 4.5 : Atak türleri ve atak sayıları. 16
Çizelge 4.6 : Snort - Tehdit aktörleri. 18
Çizelge 4.7 : Snort-İşletim sistemleri 20
Çizelge 4.8 : Snort - Platformlar 21
Çizelge 4.9 : Suricata-Tehdit aktörleri 23
Çizelge 4.10 : Suricata - İşletim sistemleri 25
Çizelge 4.11 : Suricata - Platformlar 26
Çizelge 4.12 : OSSEC ve Wazuh Atak Sayısı 32
Çizelge 4.13 : OSSEC ve Wazuh - Komut ve alarmlar 33
Çizelge 4.14 : Community - Snort Hata Matrisi 35
Çizelge 4.15 : Community - Suricata Hata Matrisi 35
Çizelge 4.16 : Broadcom - Snort Hata Matrisi 36
Çizelge 4.17 : Broadcom - Suricata Hata Matrisi 36
Çizelge 4.18 : Registered - Snort Hata Matrisi 36
Çizelge 4.19 : Registered - Suricata Hata Matrisi 37
Çizelge 4.20 : Talos - Snort Hata Matrisi 37
Çizelge 4.21 : Emerging Threats - Suricata Hata Matrisi 37
Çizelge 4.22 : Tüm kural setleri birlikte - Snort Hata Matrisi 38
Çizelge 4.23 : Tüm kural setleri birlikte - Suricata Hata Matrisi 38
Çizelge 4.24 : Saldırı Tespit Sistemlerinin Kesinlik,Duyarlılık ve F-Skor Değerleri 38
Çizelge 4.25 : Kural Setlerinin Kesinlik, Duyarlılık ve F-Skor Değerleri 39
Çizelge Ek.7 : Zararlı Trafiklerin Listesi 56
Çizelge Ek.8 : Zararsız Trafiklerin Listesi 69

KISALTMALAR

ABD : Amerika Birleşik Devletleri

CVE : Ortak Güvenlik Açıklıkları ve Etkilenmeler (Common Vulnerabilities and Exposures)

CPU : Merkezi İşlem Birimi (Central Process Unit)

D : Duyarlılık

DVWA : Damn Vulnerable Web Application

F : F-Skor

GPL : GNU Genel Kamu Lisansı

IDS : Saldırı Tespit Sistemi

K : Kesinlik

NAT : Ağ Adresi Çeviricisi (Network Address Translation)

NIDS : Ağ Tabanlı Saldırı Tespit Sistemi

HIDS : Host Tabanlı Saldırı Tespit Sistemi

MAC : Ortam Erişim Kontrolü (Media Access Control)

NIC : Ağ Arabirim Kartı (Network Interface Card)

NIDPS : Ağ Tabanlı Saldırı Tespit ve Engelleme Sistemi

NSM : Ağ Güvenliği İzleme

NIC : Ağ Arabirim Kartı

OISF : Open Information Security Foundation

OVA : Açık Sanal Cihaz (Open Virtual Appliance)

RAM : Rastgele Erişimli Bellek (Random Access Memory)

SQL : Yapılandırılmış Sorgu Dili (Structured Query Language)

WebApp : Web Uygulaması (Web Application)

XSS : Siteler Arası Betik Çalıştırma (Cross Site Scripting)

1. GİRİŞ

Veriye ve internet servislerine erişim, ağ uygulamaları ve iletişim için kullanılan ağlar kötü niyetli aktiviteler için hedef haline gelmiştir. Oltalama saldırısı ile kişilerin kandırılıp kişisel verilerinin ele geçirilmesi, zararlı yazılım içeren dosyaların paylaşılması, sistemlere yetkisiz erişim sağlanması vb. aktiviteler ağ yoluyla bilişim sistemlerini kötüye kullanarak kişilere ve sistemlere zarar vermektedir. Bu kötü niyetli aktiviteleri tespit etmek, önlemek ve sistemlerin sürekliliğini sağlamak amacıyla çeşitli ağ güvenlik teknolojileri geliştirilmiş olup bu yöntemler içinde zararlı davranışları tespit eden ve ataklara karşı uyarı oluşturan Saldırı Tespit Sistemleri (IDS) büyük önem kazanmıştır. Saldırı Tespit Sistemleri yer aldığı ortama göre ağ tabanlı ve host tabanlı olarak ikiye ayrılmıştır. Ağ tabanlı Saldırı Tespit Sistemleri bütün ağı izlerken host tabanlı Saldırı Tespit Sistemleri ise bulunduğu host bilgisayarda meydana gelen güvenlik olaylarını izlemektedir [1] [2] [3]. Belirli örüntüler (imza) arayarak veritabanında yer alan örüntüyle uyumlu atakları tespit eden imza tabanlı Saldırı Tespit Sistemleri, bu amaçla önceden belirlenmiş kurallar kullanmaktadır. Kurallar kullanıcı tarafından atak özelinde yazılabileceği gibi çeşitli ataklara ait kuralların yer aldığı kural setleri de kullanılabilmektedir. Birden fazla Saldırı Tespit Sistemi ve açık kaynak kural setinin olması ise en etkin Saldırı Tespit Sistemi ve kural seti kombinasyonunun seçilmesi sürecini karmaşıklaştırmaktadır.

1.1. Tezin Amacı

Bu çalışmanın amacı açık kaynak Saldırı Tespit Sistemleri'nin varsayılan konfigürasyonlarla ve açık kural setleri ile tespit etkinlikleri ölçülerek en etkin açık kaynak Saldırı Tespit Sistemi ve açık kaynak kural seti kombinasyonunun belirlenmesidir. Testlerde elde edilen sonuçlar doğrultusunda ücretsiz ve açık kaynak kural seti kullanan organizasyonlar için hangi Saldırı Tespit Sistemi ve kural seti ile atakların tespiti konusunda daha etkin sonuçlar elde edileceği belirlenecektir. Bu amaçla en çok kullanılan açık kaynak Ağ Tabanlı Saldırı Tespit Sistemleri olan Snort ve Suricata IDS'lere yaygın kullanılan atak türlerinin yer aldığı üç farklı kategoride (web uygulama atakları, zararlı yazılım trafikleri, CVE (Common Vulnerabilities and Exposures) [4] referanslı zafiyet sömürü atakları) atak trafikleri gönderilerek tespit etkinlikleri ölçülecektir. OSSEC ve Wazuh açık kaynak Host Tabanlı Saldırı Tespit

Sistemi'nin (Host based Intrusion Detection System - HIDS) ise varsayılan konfigürasyonlar ile alarm oluşturma durumlarının analiz edilmesi amacıyla OSSEC ve Wazuh kurulu test bilgisayarlarına CVE referanslı zafiyet sömürü atakları ve web uygulama atakları yapılacaktır. Saldırı Tespit Sistemleri zararsız trafikleri de tehdit olarak algılayarak yanlış pozitif alarmlar üretebilmektedir. Yanlış pozitif alarm oluşturma durumlarının test edilme amacıyla ise Snort ve Suricata IDS'lere zararsız trafikler gönderilecektir.

1.2. Literatür Taraması

Şimdiye kadar yapılan çalışmalarda açık kaynak Saldırı Tespit Sistemleri'nin etkinliğini test etmek amacıyla ölçeklenebilirlik, performans, alarm üretme sayısı vb. kriterler baz alınarak testler yapılmıştır. Ancak farklı açık kaynak kural setleri kullanılarak Saldırı Tespit Sistemi'nin temel özelliği olan atak tespit etkinliği üzerine kapsamlı bir çalışma yapılmamıştır.

J. S. Whitea 2013 yılında yayınlanan çalışmasında [5] Snort ve Suricata IDS'lerini ölçeklenebilirlik ve performans kriterleri üzerinden karşılaştırmıştır. Kullanılan çekirdek sayısını (1 ila 24 çekirdek), imza karşılaştırması için kullanılan kural setlerini, iş yükünü (saniyede işlenen paket sayısı) ve her iki IDS'in konfigürasyonunu değiştirerek toplam 8600 test gerçekleştirmiştir. Testlerde kullanılan metrik ise kullanılan bellek miktarı ve CPU (Merkezi İşlem Birimi) etkinliği olmuştur. Sonuçlar, hem Snort hem de Suricata'nın ölçeklenebilir olduğunu ancak Suricata'nın neredeyse tüm test senaryolarında iş yükü metriği doğrultusunda Snort'tan daha iyi performans sergilediğini göstermiştir. Suricata ayrıca daha düşük ortalama bellek kullanımı ve daha düşük ortalama CPU kullanımı sergilemiştir. Ancak J. S. Whitea'nın çalışmasında tespit etkinliğini ölçmeye yönelik bir test yapılmamıştır.

Kittikhun Thongkanchorn [6], üç popüler açık kaynaklı Saldırı Tespit Sistemi olan Snort, Suricata ve Bro-IDS'in tespit etkinliğini karşılaştırmış ve analiz etmiştir. Aktifleştirilmiş kural sayısının, farklı trafik yoğunluklarının ve sekiz saldırı türünün Saldırı Tespit Sistemleri'nin tespit etkinliği üzerindeki etkisinin araştırıldığı çalışmada, TCP trafiği için bütün IDS'lerin düşük paket kaybı ve düşük CPU kullanımına sahip olduğu, trafik yoğunluğu arttığında CPU kullanımının belirgin şekilde arttığı tespit edilmiştir. Herbir IDS farklı atak türleri için farklı sayıda paket

kaybı yaşamış ve farklı sayıda alarm oluşturmuştur. Yapılan çalışmada tespit etkinliği parametrelerden biri olarak kullanılmasına rağmen kural seti olarak sadece Emerging Threats kural seti kullanılmış, diğer açık kaynak kural setleriyle ilgili bir çalışma yapılmamıştır.

M. F. Ridho'nun yaptığı çalışmada [7] Snort, Bro ve Suricata, port tarama ve penetrasyon olmak üzere iki test aşamasına sokularak herbir Saldırı Tespit Sistemi'nin CPU ve RAM (Rastgele Erişimli Bellek) kullanımı ve alarm üretme miktarı metrikleri üzerinden avantaj ve dezavantajları tespit edilmiştir. Snort ve Suricata'nın kurulumunun ve kurallarının güncellenmesinin kolay olduğu, Bro'nun ise en az miktarda kaynak gerektirdiği sonucuna varılmıştır. Yapılan çalışmada kural setlerine hiç değinilmemiştir.

H. Alnabusi'nin yaptığı çalışmada [8] Snort IDS kullanarak SQL (Yapılandırılmış Sorgu Dili) Enjeksiyon atağını tespit etmek için beş farklı Snort kuralından oluşan bir çözüm önerilmektedir. Elde ettikleri sonuçlar, SQL Enjeksiyon ataklarının tespiti konusunda önerilen yöntemin aynı veri setini kullanan diğer benzer tekniklerden daha iyi performans sergilediğini göstermektedir. Yapılan çalışma SQL Enjeksiyon ataklarının Snort ile tespitinde etkili bir çözüm önerse de diğer atak türlerine dair bir çalışma yapılmamıştır.

M. Hänninen yayınladığı tez çalışmasında [9] Snort, Suricata ve Bro IDS'i konfigürasyon, tespit etkinliği ve sürdürülebilirlik kriterlerine göre karşılaştırmıştır ve en etkin sonuçları Suricata ile almıştır. Tespit etkinliği üzerine 15 farklı atak trafiği analiz edilerek skor tutulmasına rağmen tespit etkinliğini hedef alan kapsamlı bir çalışma yapılmamıştır.

H. Alyami'nin yayınladığı çalışmada [10] Zeek, Suricata, Security Onion, OSSEC ve Snort'un etkinlikleri hibrid bulanık mantık tabanlı bir yaklaşım kullanılarak tespit edilmek istenmiştir. Yapılan testler sonucunda Suricata'nın yoğun olan trafiğin analizinde Snort'tan daha etkin olduğu gözlenmiştir. Ancak kural setleriyle ilgili bir çalışma yapılmamıştır.

Bu çalışmalarda Snort ve Suricata'nın etkinliğini test etmek amacıyla farklı metrikler kullanılmıştır. Ancak açık kural setlerinin tespit etkinliğinin test edilip karşılaştırıldığı herhangi bir çalışma yapılmamıştır. OSSEC'in ve Wazuh'un alarm oluşturma etkinliği üzerine de bir çalışma yapılmamıştır.

1.3. Tezin İçeriği

1. Bölüm'de Saldırı Tespit Sistemleri'nin önemi vurgulanarak bu tezin amacı ve imza tabanlı Saldırı Tespit Sistemleri ile ilgili yapılan çalışmalar anlatılmıştır. 2. Bölüm'de Saldırı Tespit Sistemleri ortamına ve tespit yöntemine göre sınıflandırılmış ve kullanılan açık kaynak kural setleri tanıtılmıştır. 3. Bölüm'de testler sırasında kullanılan topoloji, uygulama ve araçlar ile Saldırı Tespit Sistemleri'nin konfigürasyonlarına yer verilmiştir. 4. Bölüm'de testler sırasında kullanılan metodoloji açıklanmış, zararlı ve zararsız trafikler için yapılan testler anlatılmıştır. 5. Bölüm'de ise yapılan testlerden elde edilen sonuçlar paylaşılmış ve gelecekte yapılabilecek çalışmalar için önerilerde bulunulmuştur.

2. SALDIRI TESPİT SİSTEMLERİ

2.1. Saldırı Tespit Sistemleri'ne Genel Bakış

Saldırı tespiti, bir bilgisayar sisteminde veya ağda meydana gelen olayları izleyerek bu olayları, güvenlik politikalarının ihlali veya olası ihlallerin işaretleri için analiz etme sürecidir. Saldırı Tespit Sistemleri ise saldırı tespit sürecini otomatikleştiren yazılımlardır [11].

Saldırı Tespit Sistemleri; saldırıları tespit ettikleri ortama ve tespit yöntemlerine göre sınıflandırılabilir.

Çizelge 2.1'de deneylerde kullanılan IDS'lerin türleri gösterilmiştir.

Çizelge 2.1 : Kullanılan IDS türleri.

IDS	TÜR
SNORT	Ağ Tabanlı-İmza Tabanlı
SURİCATA	Ağ Tabanlı-İmza Tabanlı
OSSEC	Host Tabanlı-İmza Tabanlı
WAZUH	Host Tabanlı-İmza Tabanlı

2.1.1. Ortama göre saldırı tespit sistemleri

2.1.1.1. Ağ tabanlı saldırı tespit sistemi (NIDS)

Ağ Tabanlı Saldırı Tespit Sistemleri (NIDS), bir sistemi ağ tabanlı tehditlerden korumak amacıyla ağ trafiğini izlemek ve analiz etmek için kullanılır. Bir NIDS gelen tüm paketleri okur ve şüpheli kalıplar arar [12]. Normal olarak bir ağ arabirim kartı (NIC) seçici (nonpromiscuous) modda çalışır. Bu çalışma modunda yalnızca kendi MAC (Ortam Erişim Kontrolü) adresine ait paketler analiz edilir. NIDS kendi MAC adresine ait olmayan trafiği dinlemek için seçici omayan (promiscous) modda çalışmalıdır. Bu modda ağ içerisinde yer alan tüm trafiği dinleyebilir [3].

2.1.1.2. Host tabanlı saldırı tespit sistemi (HIDS)

Host Tabanlı Saldırı Tespit Sistemleri IDS'in ilk türüdür ve ana işlevi bir bilgisayar veya makine içinde dahili izleme yapmaktır, ancak günümüzde ağı izlemek için kullanılabilecek birçok HIDS varyantı geliştirilmiştir. HIDS, bir sistemin güvenliğinin ihlal edilip edilmediğini belirler ve buna göre alarm oluşturur [12] [13]. Host Tabanlı

Saldırı Tespit Sistemi'nin her korunan makineye (sunucu veya son kullanıcı) kurulması gerekir. Bir HIDS sistem günlük dosyaları vedosya sistemi değişiklikleri gibi o makineye özgü verileri analiz eder [14].

2.1.2. Tespit yöntemine göre saldırı tespit sistemleri

2.1.2.1. İmza tabanlı saldırı tespit sistemi

İmza tabanlı Saldırı Tespit Sistemi'nde atak trafiklerine ait ağ paketlerinde yer alan imza olarak adlandırılan kalıplar incelenir. Daha sonra incelenen bu kalıplar Saldırı Tespit Sistemi'nin imza veritabanı ile karşılaştırılarak atak tespit edilir [15].

2.1.2.2. Anomali tabanlı saldırı tespit sistemi

Anomali tabanlı Saldırı Tespit Sistemi'nin çalışma prensibi öncelikle normal ağ davranışının tespit edilmesine dayanmaktadır. Bu normal davranışa uymayan trafikler ise anomali olarak kabul edilmektedir [16]. Bu tür Saldırı Tespit Sistemleri'nin daha önce raporlanmamış atakların tespitinde kullanılabileceği değerlendirilmektedir.

2.2. Snort'a Genel Bakış

Snort açık kaynaklı ağ tabanlı ve imza tabanlı bir Saldırı Tespit ve Engelleme Sistemi'dir (Network based Intrusion Detection/Prevention System - NIDPS). Snort IDS, kötü niyetli ağ etkinliğini tanımlamaya yardımcı olan bir dizi kural kullanır ve bu kuralları, bu kurallarda yer alan kalıplarla eşleşen paketleri bulmak ve kullanıcılar için alarmlar üretmek için kullanır [17]. Snort, neredeyse her bilgisayar mimarisi ve işletim sistemi platformu üzerine kurulabilir. Gerçek zamanlı olarak da alarm üreten Snort IDS'in kuralları tek satır şeklindedir, okunması ve anlaşılması kolaydır ve değiştirilebilir [18]. Şimdiye kadar yapılan analizler sonucunda elde edilen bulgulara göre de Snort, en popüler ve yaygın olarak kullanılan IDS'dir [19] [20].

2.3. Suricata'ya Genel Bakış

Suricata, Snort'a bir alternatif oluşturmak için ABD (Amerika Birleşik Devletleri) İç Güvenlik Bakanlığı tarafından finanse edilen bir proje kapsamında Open Information Security Foundation (OISF) tarafından geliştirilmiştir. Suricata, gerçek zamanlı saldırı tespiti (IDS), saldırı önleme (IPS), ağ güvenliği izleme (NSM) ve çevrimdışı pcap işleme yeteneğine sahip açık kaynaklı, ücretsiz ve hızlı bir tespit motorudur. Snort'tan

farklı olarak çok iş parçacıklı (multi-threaded) bir uygulamadır ve çok çeşitli ağ tehditlerini tespit etmek için kapsamlı ve güçlü bir Kural Seti ve imza veritabanına sahiptir. Linux, FreeBSD, Open BSD, Mac ve Windows üzerinde çalışır [21] [22].

2.4. OSSEC'e Genel Bakış

OSSEC ölçeklenebilir, açık kaynaklı host tabanlı Saldırı Tespit Sistemi'dir. Güçlü bir korelasyon ve analiz motoruna sahiptir. Entegre günlük analizi, dosya bütünlüğü kontrolü, merkezi politika uygulanması, rootkit algılama, gerçek zamanlı uyarı ve aktif yanıt özellikleri mevcuttur [23] [24].

2.5. Wazuh'a Genel Bakış

Wazuh, tehdit önleme, algılama ve müdahale için kullanılan ücretsiz ve açık kaynaklı bir host tabanlı Saldırı Tespit Sistemi'dir. OSSEC'in fonksiyonelliğini artırmak için geliştirilen Wazuh, güvenlik verilerini toplamak, indekslemek ve analiz etmek için kullanılır ve kuruluşların izinsiz girişleri, tehditleri ve davranışsal anormallikleri tespit etmesine yardımcı olur [25].

2.6. Açık Kaynak Kural Setleri ve Kurallar

2.6.1. Snort Community kural seti

Topluluk kuralları, açık kaynak topluluğu üyeleri veya Snort Entegratörleri tarafından gönderilen tüm kurallardan oluşur. Bu kurallar tüm Snort kullanıcılarına ücretsiz olarak sunulur ve GPLv2 tarafından yönetilir. Topluluk Kural Seti kayıt olmadan indirilebilir ve Suricata'da da kullanılabilir [26].

2.6.2. Snort Registered kural seti

Bu Kural Seti, Snort Subscriber Kural Seti'nin 30 gün gerisindedir ve sıfırıncı gün tehditleri içermez. Suricata'da da kullanılabilir [27].

2.6.3. Broadcom (web) kuralları

Broadcom'un blog sayfasında ağlara yönelik SQL Enjeksiyonu ve XSS (Siteler Arası Betik Çalıştırma - Cross Site Scripting) saldırılarını tespit atmak amacıyla Regular Expression tabanlı kurallar oluşturulmuştur. Bu kurallar Snort ve Suricata'nın her ikisinde de kullanılabilir [28].

2.6.4. Talos kural seti

Talos (eski adıyla VRT), bilgisayar korsanlığı faaliyetleri, izinsiz giriş denemeleri, kötü amaçlı yazılımlar ve güvenlik açıklarındaki en son eğilimleri proaktif olarak keşfetmek, değerlendirmek ve bunlara yanıt vermek için çalışan ağ güvenliği uzmanlarından oluşan bir gruptur. Bu ekip, Snort, ClamAV ve Spamcop.net topluluklarınınkaynakları tarafından desteklenmektedir ve bu da onu ağ güvenliği endüstrisindeki gelişmelere adanmış en büyük grup haline getirmektedir [29]. Talos Kural Seti sadece Snort3 için mevcuttur.

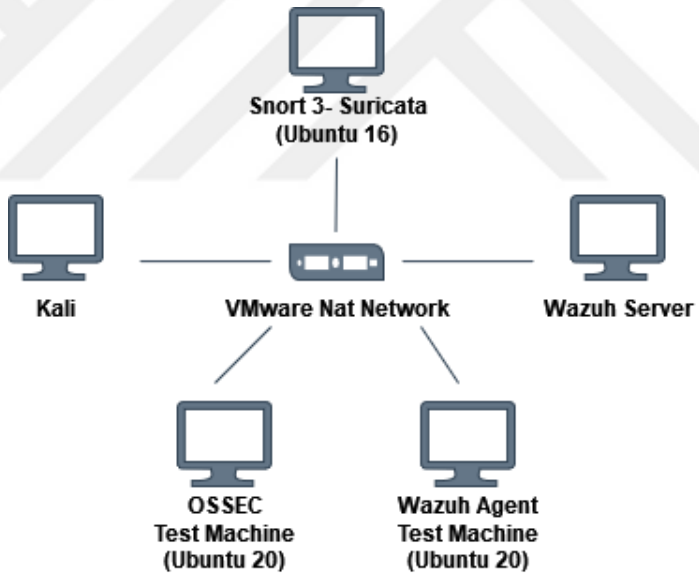
2.6.5. Emerging Threats kural seti

Eskiden Bleeding Edge Threats olarak bilinen Emerging Threats yeni ve en güncel tehditler ve araştırmalarla ilgili veriler üretmektedir [30]. Emerging Threats Kural Seti Snort 2.9.x ve Suricata'da kullanılmaktadır. Snort3 için henüz bir Kural Seti bulunmamaktadır.

3. TEST ORTAMI

3.1. Topoloji

Testler için bir Ubuntu 16 sanal makine üzerine Snort3 ve Suricata IDS'leri kurulmuştur. Wazuh HIDS için gerekli olan Wazuh Server sanal makinesi Wazuh Dokümantasyon sayfasından [31] OVA (Open Virtual Appliance) imaj dosya formatında indirilerek kurulmuştur. Birbirinin aynısı olan iki Ubuntu 20 sanal makinesi test makinesi olarak kullanılmış, sanal makineler üzerine DVWA uygulaması, lokal OSSEC HIDS ve Wazuh agent kurulmuştur. Atak trafikleri ise Kali makine üzerinden gönderilmiştir. Bu beş sanal makinenin birbiriyle NAT (Ağ Adresi Çeviricisi – Network Address Translation) ağı üzerinden haberleşmesi sağlanmıştır. Kullanılan topoloji Şekil 3.1'de gösterilmiştir.



Şekil 3.1 : Topoloji.

3.2. Kullanılan Uygulamalar ve Araçlar

3.2.1. Metasploit framework

Metasploit Framework, istismar kodunun yazılmasına, test edilmesine ve çalıştırılmasına olanak tanıyan Ruby tabanlı, modüler bir sızma testi platformudur. Metasploit Framework, güvenlik açıklarını test etmek, kullanılabilecek atak vektörlerini tespit etmek ve saldırı yürütmek için kullanılabilecek araçlar içerir.

Özünde, Metasploit Framework, penetrasyon testi için yaygın olarak kullanılan araçlardan oluşan bir koleksiyondur [32] [33].
Testler sırasında Metasploit ile oluşturulan CVE referanslı zafiyet sömürü ataklarından ve web uygulama ataklarından yararlanılmıştır.

3.2.2. Wireshark

Wireshark, yaygın olarak kullanılan ağ protokolü analiz aracıdır [34] [35].
Testler sırasında oluşturulan atak trafikleri Wireshark ile kaydedilmiş ve ardından bu trafikler Snort3 ve Suricata ile analiz edilmiştir.

3.2.3. Dvwa (Damn Vulnerable Web Application)

DVWA, bilinçli olarak üzerine çok sayıda zafiyet eklenmiş bir PHP/MySQL web uygulamasıdır. Temel hedefleri, güvenlik profesyonellerinin becerilerini ve araçlarını yasal bir ortamda test etmelerine yardımcı olmak ve web geliştiricilerinin web uygulamalarını güvence altına alma süreçlerini daha iyi anlamalarına yardımcı olmaktır [36].
Testler sırasında oluşturulan WebApp (Web uygulama) atak trafiklerinde DVWA uygulamasındaki zafiyetlere yönelik ataklar yer almaktadır.

3.3. Konfigürasyon

3.3.1. Snort3 – Suricata konfigürasyonu

Snort3 ve Suricata için varsayılan konfigürasyon kullanılmakla birlikte bütün kuralların açık olması için Ek 1'de yer alan betik kullanılmıştır.

Bu betik rules klasörünün içindeki tüm dosyalarda "# alert" olan kısımları "alert" olarak değiştirmektedir. Böylece tüm yorum satırı içindeki kurallar açılmaktadır.

3.3.2. Ossec konfigürasyonu

Testler sırasında IP bloklanması yaşanmaması için kurulum aşamasında IP bloklama seçeneği Ek 2'de gösterildiği gibi "hayır" olarak seçilmiştir.

3.3.3. Wazuh konfigürasyonu

Wazuh agent kurulumu sırasında IP bloklanması yaşanmaması için konfigürasyon dosyasında "active response" kısmı Ek 3'te gösterildiği gibi düzenlenmiştir.

3.4. Snort3'te Trafiklerin Analizi

Trafiklerin Snort3'te analiz edilmesi için Snort'un paket okuma özelliğinden yararlanılmıştır. Bu özellik ile canlı bir arayüzü dinlemek yerine daha önceden kaydedilmiş bir trafiği tekrar çalıştırarak analiz etmek mümkündür [37]. Snort3 için Community, Registered ve Talos Kural Setleri ile Broadcom kuralları kullanılmıştır. Çok sayıda paketin tek seferde analiz edilmesi için Ek 4'te yer alan Python betiğinden yararlanılmıştır.

Her trafik için oluşan alarmlar bir txt dosyasına kaydedilmiştir, pcap dosyalarının ismi de bir csv dosyasına kaydedilerek eğer trafik alarm oluşturmuşsa karşılarına 1, oluşturmamışsa 0 yazılarak kaydedilmiştir.

3.5. Suricata'da Trafiklerin Analizi

Trafiklerin Suricata'da analiz edilmesi için Suricata'nın paket okuma özelliğinden yararlanılmıştır. Bu özellik ile canlı bir arayüzü dinlemek yerine daha önceden kaydedilmiş bir trafiği tekrar çalıştırarak analiz etmek mümkündür [38]. Suricata için Community, Registered, Emerging Threats Kural Setleri ile Broadcom kuralları kullanılmıştır. Çok sayıda paketin tek seferde analiz edilmesi için Ek 5'te yer alan Python betiğinden yararlanılmıştır.

Her trafik için oluşan alarmlar bir txt dosyasına kaydedilmiştir, pcap dosyalarının ismi de bir csv dosyasına kaydedilerek eğer trafik alarm oluşturmuşsa karşılarına 1, oluşturmamışsa 0 yazılarak kaydedilmiştir.

3.6. OSSEC ve Wazuh'ta Trafiklerin Analizi

Atak trafikleri test makinelerine gönderildikten sonra OSSEC'in ve Wazuh'un web tabanlı arayüzlerinden alarm oluşup oluşmadığı kontrol edilmiştir.

4. TESTLER

Snort'un ve Suricata'nın varsayılan konfigürasyonlar ve açık kural setleri ile tespit etkinliğini ölçmek amacıyla 5 farklı Kural Seti kullanılmıştır. OSSEC ve Wazuh için ise varsayılan konfigürasyon ile üretilen alarmlar incelenmiştir.

Yaklaşım olarak Microsoft Exchange Server, Apache James, Cisco DCNM gibi kurumsal ağlarda kullanılan ürünlere ait zafiyetlerin sömürüldüğü ataklara yer verilmiştir [19].

4.1. Metodoloji

4.1.1. Donanım özellikleri

Testlerde 3.1. Topoloji bölümünde anlatıldığı şekilde VMWare sanallaştırma yazılımı üzerine beş farklı sanal makine kurularak NAT ağı içinde bu makineler konumlandırılmıştır. Sanal makinelerin donanım özellikleri aşağıdaki çizelgede verilmiştir.

Çizelge 4.1 : Donanım Özellikleri.

Suricata – Snort 3 Test Makinesi	
RAM	17 GB
Hard Disk	350 GB
İşlemci Sayısı	2
İşletim Sistemi	Ubuntu 16.04
Kali	
RAM	5.5 GB
Hard Disk	60 GB
İşlemci Sayısı	2
İşletim Sistemi	Kali 2020_4
OSSEC Test Makinesi	
RAM	4 GB
Hard Disk	20 GB
İşlemci Sayısı	2
İşletim Sistemi	Ubuntu 20
Wazuh Agent Test Makinesi	
RAM	4 GB
Hard Disk	20 GB
İşlemci Sayısı	2
İşletim Sistemi	Ubuntu 20
Wazuh Sunucusu	
RAM	4 GB
Hard Disk	40 GB
İşlemci Sayısı	4
İşletim Sistemi	wazuh-4.2.5_1.13.2

Yapılan testlerde kullanılan zararsız trafiklerin boyutunun büyük olması nedeniyle Snort 3 – Suricata test makinesinin Hard Disk boyutu 350 GB olarak tanımlanmıştır.

4.1.2. Saldırı tespit sistemi versiyonları

Testlerde kullanılan Saldırı Tespit Sistemleri'nin versiyonları aşağıdaki çizelgede verilmiştir.

Çizelge 4.2 : Saldırı Tespit Sistemi Versiyonları.

Saldırı Tespit Sistemi	**Versiyon**
Snort	3.0.3
Suricata	5.0.5
OSSEC	0.8
Wazuh	4.2.5

4.1.3. Açık kaynak kural seti versiyonları

Testlerde kullanılan açık kaynak kural setlerinin versiyonları aşağıdaki çizelgede verilmiştir.

Çizelge 4.3 : Açık Kaynak Kural Seti Versiyonları.

Açık Kaynak Kural Seti	**Versiyonu**
Snort 3 için Snort Community Kural Seti	snort3-community-rules
Suricata için Snort Community Kural Seti	community-rules
Snort 3 için Snort Registered Kural Seti	snortrules-snapshot-3000
Suricata için Snort Registered Kural Seti	snortrules-snapshot-2983
Broadcom Kuralları	[28]
Talos Kural Seti	Talos_LightSPD-2021-02-03-001
Emerging Threats Kural Seti	Snort-2.9.7.0

Testlerde kullanılan Broadcom Kuralları Ek 6'da verilmiştir.

4.1.4. Kullanılan yöntem ve metrikler

Testlerde yer alan IDS'lerin tespit etkinliği alarm üretilen atak sayısının toplama atak sayısına oranının hesaplanması aracılığıyla ölçülmüştür.

Tespit etkinliğinin ölçülmesi amacıyla deneylerde kullanılan metrik aşağıdaki gibidir.

$$Etkinlik = (Alarm\ Üretilen\ Atak\ Sayısı\ /\ Toplam\ Atak\ Sayısı)\ x\ 100 \quad (4.1)$$

Tahmin edilen sonuçlar ve gerçekleşen sonuçlar ile ilgili analizleri yapmak için ise Hata Matrisi yöntemi ve Kesinlik (Precision – K), Duyarlılık (Recall – D) ve F-Skor (F) metrikleri kullanılmıştır. Hata Matrisi aşağıda gösterilmiştir. Matrisin her sütunu tespit edilen sonuçları gösterirken her satırı ise gerçek sonuçları göstermektedir [39].

Çizelge 4.4 : Hata Matrisi.

Hata Matrisi		**Tespit Sonucu**		**Toplam**
		Zararsız	**Zararlı**	
Gerçek Sonuç	**Zararsız**	GN	YP	GN+YP
	Zararlı	YN	GP	YN+GP
Toplam		GN+YN	YP+GP	

Gerçek Pozitif (GP): Zararlı trafik olarak sayılan zararlı trafik sayısı
Gerçek Negatif (GN): Zararsız trafik olarak sayılan zararsız trafik sayısı
Yanlış Pozitif (YP): Zararlı trafik olarak sayılan zararsız trafik sayısı
Yanlış Negatif (YN): Zararsız trafik olarak sayılan zararlı trafik sayısı

Kesinlik metriği pozitif olarak tahmin edilen değerlerin gerçekte kaçının pozitif olduğunu göstermektedir ve aşağıdaki denklem ile hesaplanır.

$$K = \frac{GP}{YP+GP} \tag{4.2}$$

Duyarlılık metriği pozitif olarak tahmin edilmesi gereken değerlerin kaçının pozitif tahmin edildiğini göstermektedir. Aşağıdaki denklem ile hesaplanır.

$$D = \frac{GP}{YN+GP} \tag{4.3}$$

F-Skor metriği Kesinlik ve Duyarlılık değerlerinin harmonik ortalamasını göstermektedir. Harmonik ortalamanın alınmasının sebebi ise uç durumların göz ardı edilmemesidir. Aşağıdaki denklem ile hesaplanır.

$$F = \frac{2*K*D}{K+D} \tag{4.4}$$

4.2. Test 1: Zararlı Trafik

Deneyler için zararlı yazılım (malware) trafikleri, web uygulama atak trafikleri ve CVE referanslı zafiyet sömürü atak trafikleri olmak üzere üç farklı türde toplam 600 farklı atak trafiği kullanılmıştır. Zararlı yazılım trafiklerinin analizi için malware-traffic-analysis.net [39] sitesinde yer alan, 2013 - 2021 yılları arasına ait trafikler kullanılmıştır [19]. Oluşturulacak atak trafiklerinin tespitinde exploit-db [40] veritabanından ve rapid7 [41] veritabanından yararlanılmıştır. Atakların dağılımları aşağıdaki çizelgede yer almaktadır.

Çizelge 4.5 : Atak türleri ve atak sayıları.

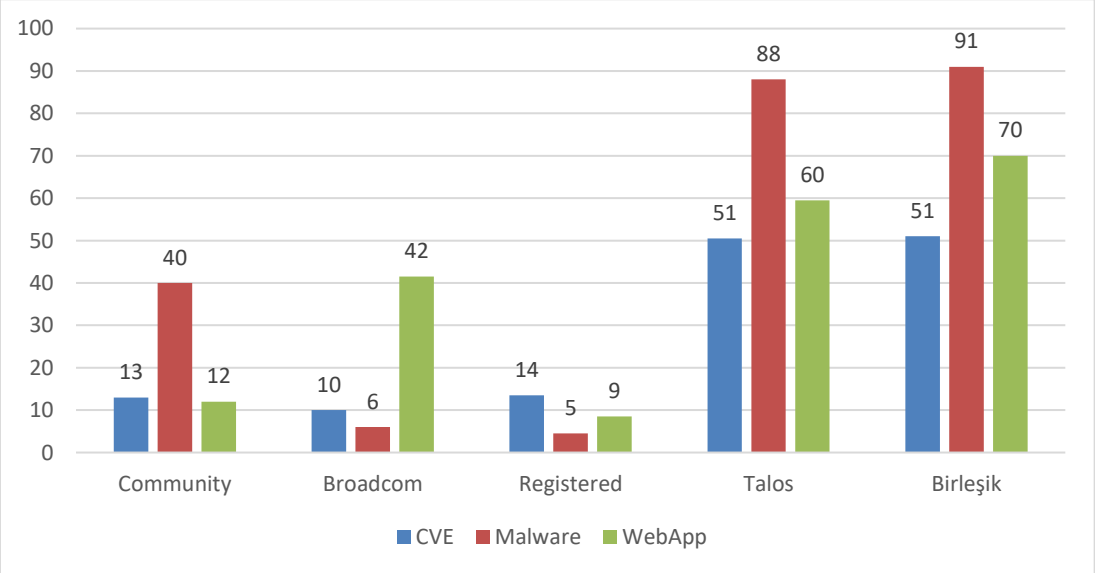
ATAK TÜRÜ	ATAK SAYISI
CVE REFERANSLI	200
ZARARLI YAZILIM (MALWARE)	200
WEB UYGULAMA (WEBAPP)	200

Testlerde kullanılan zararlı trafiklerin listesi Ek 7'de yer almaktadır.

4.2.1. Snort

4.2.1.1. Test 1.1: Tüm kural setlerinin tek başına ve birlikte kullanılması

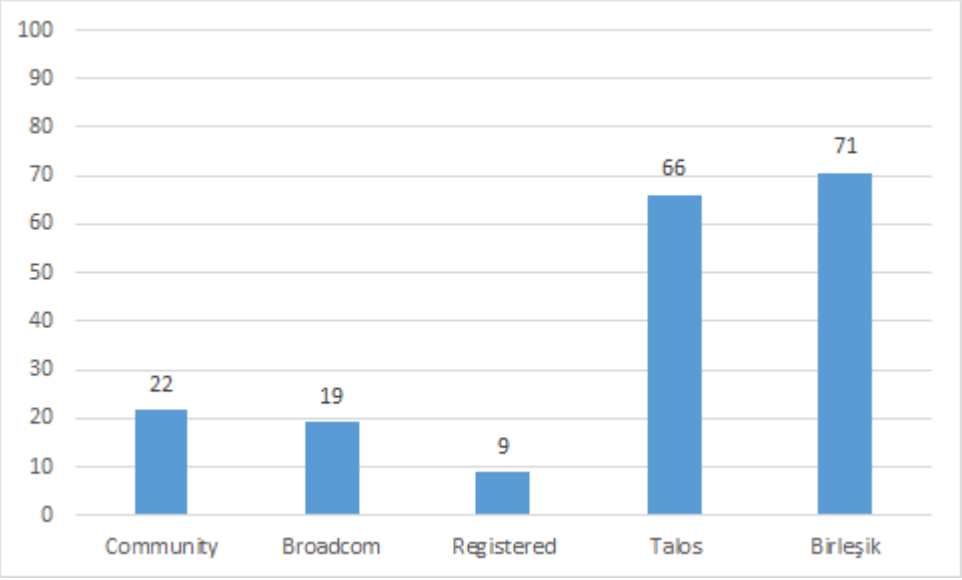
Bu deneyde tüm kural setleri ayrı ve birleşik olarak kullanıldığında Snort IDS ile tespit etkinliklerinin ölçülmesi hedeflenmiştir. Sonuçlar Şekil 4.1'de gösterilmiştir.



Şekil 4.1 : Snort - Kural setlerinin etkinliği.

Tüm kural setleri ayrı olarak analiz edildiğinde CVE referanslı zafiyet sömürü ataklarında en yüksek tespit etkinliğinin %51 ile Talos Kural Seti'nde elde edildiği, en düşük tespit etkinliğinin ise %10 ile Broadcom kuralları kullanılarak elde edildiği gözlenmiştir. Zararlı yazılım ataklarında en yüksek tespit etkinliğinin %88 ile Talos Kural Seti'nde elde edildiği, en düşük tespit etkinliğinin ise %5 ile Registered Kural Seti ile elde edildiği gözlenmiştir. Web uygulama ataklarında ise en yüksek tespit etkinliğinin %60 ile Talos Kural Seti kullanılarak elde edildiği, en düşük tespit etkinliğinin ise %9 ile Registered Kural Seti kullanılarak elde edildiği gözlenmiştir.

Tüm kural setleri birlikte kullanıldığında elde edilen sonuçlar incelendiğinde CVE referanslı zafiyet sömürü ataklarında atak tespit etkinliğinin %51 ile Talos Kural Seti'nin etkinliği ile aynı olduğu, Zararlı Yazılım ataklarının tespitinde atak tespit etkinliğinin %91'e çıktığı ve Web Uygulama ataklarında tespit etkinliğinin %70'e çıktığı gözlenmiştir.
Tüm atak türlerinin birlikte analizi Şekil 4.2'de gösterilmiştir.



Şekil 4.2 : Snort - Tüm atak türlerinin birlikte analizi.
Tüm atak türleri birlikte analiz edildiğinde (CVE referanslı, Web Uygulama, Zararlı Yazılım) ve kural seti bazlı incelendiğinde tek başına en etkin sonucun %66 ile Talos Kural Seti kullanılarak elde edildiği, en az etkinliğin ise %9 ile Registered Kural Seti kullanılarak elde edildiği gözlenmiştir. Tüm kural setleri birlikte kullanıldığında ise tespit etkinliği %71'e çıkmıştır.

4.2.1.2. Test 1.2: Zararlı yazılımların tehdit aktörlerine göre analizi

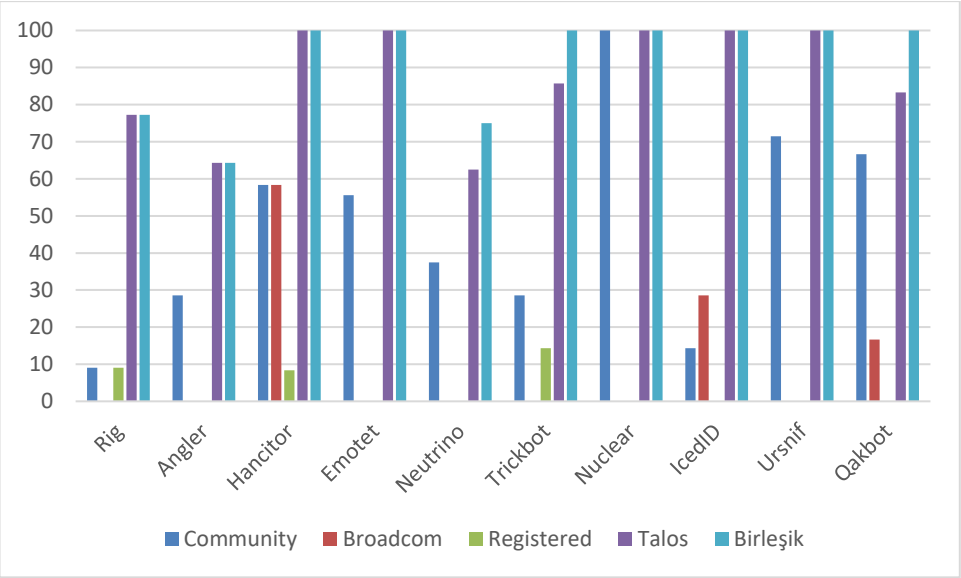
Bu deneyde Snort IDS'in zararlı yazılım trafiklerinde yer alan tehdit aktörlerini tespit etmedeki etkinliğinin ölçülmesi hedeflenmiştir. Deneyde kullanılan zararlı trafiklerde 52 farklı tehdit aktörünün yer aldığı saptanmış olsa da istatistiki hesaplamalara altı ve daha fazla sayıda trafiğin içinde yer alan tehdit aktörleri dahil edilmiştir. Aşağıdaki çizelgede tehdit aktörleri, kaç adet trafik paketinde yer aldıkları ve kural setlerinin bu trafiklerden kaç tanesine alarm oluşturduğu bilgileri yer almaktadır. En sağ sütunda ise tüm kural setleri birlikte kullanıldığında oluşan alarm sayısı gösterilmiştir.

Çizelge 4.6 : Snort - Tehdit aktörleri.

TEHDİT AKTÖRÜ	**PAKET SAYISI**	**Community**	**Broadcom**	**Registered**	**Talos**	**Birlikte**
Rig	22	2	0	2	17	17
Angler	14	4	0	0	9	9
Hancitor	12	7	7	1	12	12
Emotet	9	5	0	0	9	9
Neutrino	8	3	0	0	5	6
Nuclear	7	2	0	1	6	7
Trickbot	7	7	0	0	7	7
IcedID	7	1	2	0	7	7
Ursnif	7	5	0	0	7	7
Qakbot	6	4	1	0	5	6
Toplam:	**99**	**40**	**10**	**4**	**84**	**87**

99 atak trafiğinden Community Kural Seti 40 atak trafiğini, Broadcom kuralları 10 atak trafiğini, Registered Kural Seti 4 atak trafiğini ve Talos Kural Seti ise 84 atak trafiğini tespit etmiştir. Tüm kural setleri birlikte kullanıldığında ise 87 atak trafiği tespit edilmiştir.

Snort'un zararlı yazılım tehdit aktörlerini tespit etkinliği ise aşağıdaki grafikte gösterilmiştir (Şekil 4.3).



Şekil 4.3 : Snort - Tehdit aktörlerine göre etkinlik analizi.

Zararlı yazılım trafiklerinde kullanılan tehdit aktörlerinden en çok kullanılanlar incelendiğinde tüm kural setleri tek başına kullanıldığında Broadcom kurallarının on tehdit aktöründen sadece üçünün (Hancitor, IcedID, Qakbot) yer aldığı saldırıları tespit ederek diğer tehdit aktörlerini tespit etmede yetersiz kaldığı, Registered Kural Seti'nin on tehdit aktöründen üçünün (Rig, Hancitor, Trickbot) yer aldığı saldırıları tespit ederek diğer tehdit aktörlerini tespit etmede yetersiz kaldığı, Community Kural Seti'nin on tehdit aktöründen hiç tespit edemediği bir aktörün olmadığı ancak tespit oranlarının Talos Kural Seti'ne göre daha düşük olduğu gözlenmiştir. En yüksek tespit oranının Talos Kural Seti'yle olduğu ve tüm çeşitlerdeki tehdit aktörlerini tespit ettiği gözlenmiştir. Tüm kural setleri birlikte kullanıldığında ise on çeşit tehdit aktöründen yedisi tespit edilerek en yüksek tespit oranı elde edilmiştir. Nuclear tehdit aktörüne ait saldırıların tamamı hem Community Kural Seti hem de Talos Kural Seti tarafından tespit edilmiştir. Talos Kural Seti Emotet, Ursnif, Nuclear, Hancitor ve IcedID tehdit aktörlerine ait atakların tamamını tespit etmiştir.

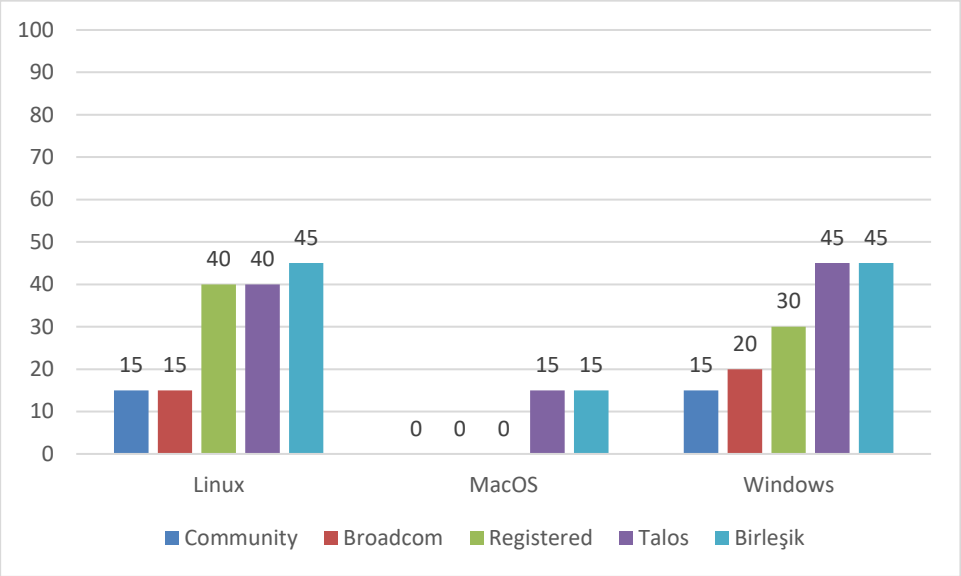
4.2.1.3. Test 1.3: CVE referanslı atakların işletim sistemlerine göre analizi

Bu deneyde Snort IDS'in açık kural setleri ile Linux, Windows ve MacOS işletim sistemlerine yönelik yapılan atakların tespit etkinliğinin ölçülmesi hedeflenmiştir. Bu amaçla öncelikle CVE referanslı zafiyet sömürü ataklarının hangi işletim sistemlerine yönelik yapıldığı tespit edilmiş ve analiz için her işletim sistemine ait trafiklerden 20

adet seçilmiştir. İşletim sistemleri ve paket sayıları Çizelge 4.7'de gösterilmiştir. Deneylerden elde edilen sonuçlar ise Şekil 4.4'te gösterilmiştir.

Çizelge 4.7 : Snort-İşletim sistemleri.

İŞLETİM SİSTEMİ	PAKET SAYISI
LINUX	20
WINDOWS	20
MACOS	20



Şekil 4.4 : Snort - İşletim sistemlerinde kural seti bazında analiz.

Deney sonuçları incelendiğinde Community Kural Seti'nin %15 ile Linux ve Windows işletim sistemlerinde tespit etkinliği gösterdiği, Broadcom kurallarının %15 ile Linux'ta ve %20 ile Windows'ta tespit etkinliği gösterdiği, Registered Kural Seti'nin %40 ile Linux'ta ve %30 ile Windows'ta tespit etkinliği gösterdiği, en yüksek tespit etkinliğinin ise Talos Kural Seti ile elde edildiği gözlenmiştir. Talos Kural Seti Windows işletim sistemine ait atakların %45'ini tespit ederken, Linux'a ait atakların %40'ını MacOS'a ait atakların ise %15'ini tespit etmiştir. MacOS işletim sisteminde Talos dışında hiçbir Kural Seti atakları tespit edememiştir. Tüm işletim sistemlerine yönelik yapılan atakların tespit etkinliğinin %50'nin altında kaldığı gözlenmiştir.

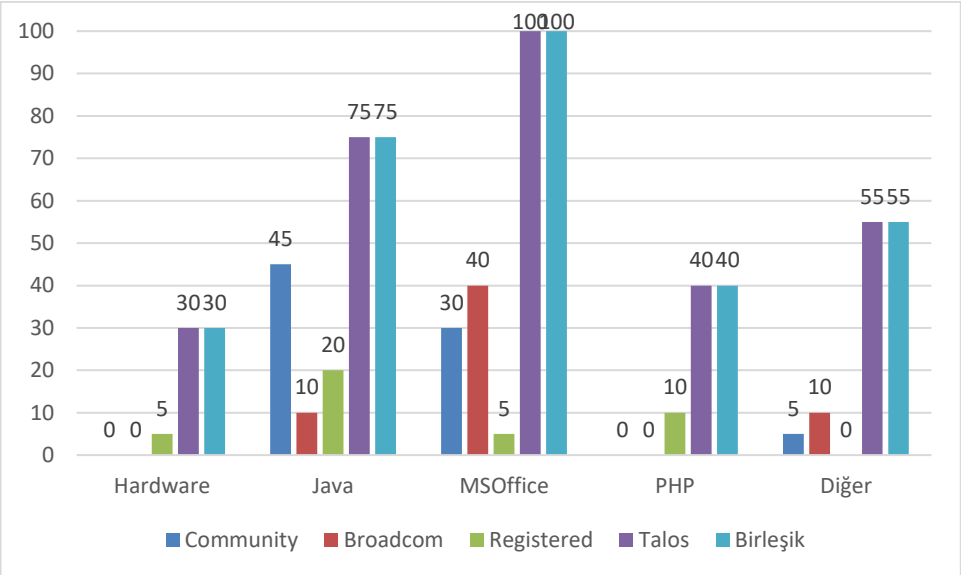
Tüm kural setleri birlikte kullanıldığında yalnızca Linux işletim sisteminin tespit etkinliği artmıştır, tespit etkinliği %40'tan %45'e yükselmiştir. MacOS ve Windows'un tespit etkinlik değerleri aynı kalmıştır.

4.2.1.4. Test 1.4: CVE referanslı atakların platformlara göre analizi

Bu deneyde Snort IDS'in açık kural setleri ile Java, PHP, MSOffice, Hardware (Donanım) platformları ve Diğer Platformlara yönelik yapılan atakların tespit etkinliğinin ölçülmesi hedeflenmiştir. Bu amaçla öncelikle CVE referanslı zafiyet sömürü ataklarının hangi platformlara yönelik yapıldığı tespit edilmiş ve analiz için her platforma ait trafiklerden 20 adet seçilmiştir. Platformlar ve paket sayıları Çizelge 4.8'de gösterilmiştir. Deneylerden elde edilen sonuçlar ise Şekil 4.5'te gösterilmiştir.

Çizelge 4.8 : Snort – Platformlar.

PLATFORM	PAKET SAYISI
JAVA	20
PHP	20
MSOFFICE	20
HARDWARE (DONANIM)	20
DİĞER	20



Şekil 4.5 : Snort – Platformlarda kural seti bazında analiz.

Deney sonuçları incelendiğinde Community Kural Seti'nin %45 ile Java, %30 ile MSOffice platformlarında ve %5 ile Diğer platformlarda tespit etkinliği gösterdiği;

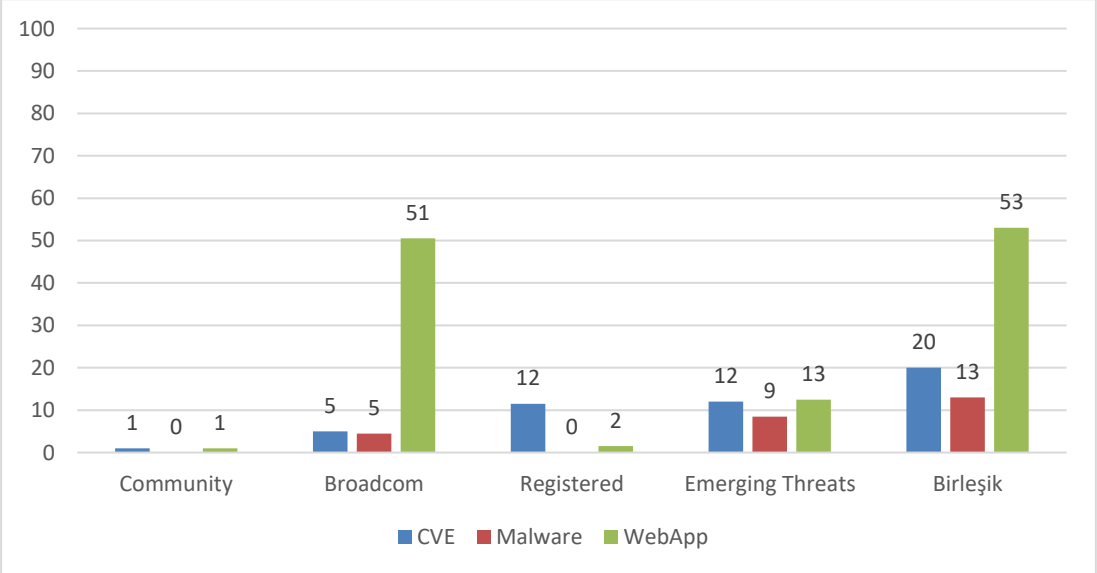
Broadcom kurallarının %10 ile Java, %40 ile MSOffice platformlarında ve %10 ile Diğer platformlarda tespit etkinliği gösterdiği tespit edilmiştir. Registered Kural Seti'nin %5 ile Donanım (Hardware), %20 ile Java, %5 ile MSOffice ve %10 ile PHP platformalarında tespit etkinliği gösterdiği, Diğer platformlarda tespit etkinliği göstermediği tespit edilmiştir. Tüm platformlarda en çok tespit etkinliğinin Talos ile elde edildiği görülmüştür. MSOffice uygulamalarına yönelik yapılan saldırılarda Talos Kural Seti %100 tespit etkinliği göstermiştir. %30 ile Donanım, %75 ile Java, %40 ile PHP platformlarında ve %55 ile Diğer platformlarda tespit etkinliği göstermiştir.

Tüm kural setleri birlikte kullanıldığında platformlardaki maksimum tespit etkinliği değişmemiş ve Talos'un tespit etkinliği ile aynı kalmıştır.

4.1.2. Suricata

4.2.2.1. Test 1.5: Tüm kural setlerinin tek başına ve birlikte kullanılması

Bu deneyde tüm kural setleri ayrı ve birleşik olarak kullanıldığında Suricata IDS ile tespit etkinliklerinin ölçülmesi hedeflenmiştir. Deney sonuçları Şekil 4.6'da gösterilmiştir.



Şekil 4.6 : Suricata - Kural setlerinin etkinliği.

Suricata'nın her bir kural seti için tespit etkinliği analiz edildiğinde Community Kural Seti'nin CVE referanslı zafiyet sömürü ataklarında ve web uygulama atakalrında %1 tespit etkinliği gösterdiği, Broadcom kurallarının ise %5 ile CVE referanslı zafiyet sömürü ataklarında ve zararlı yazılım trafiklerinde tespit etkinliği gösterdiği, web

uygulama ataklarında ise %51 tespit etkinliği gösterdiği gözlenmiştir. Registered Kural Seti %12 ile en çok CVE referanslı zafiyet sömürü ataklarında etkili olmuştur. Zararlı yazılım trafiklerinde hiç tespit etkinliği gösteremezken web uygulama atakalarında ise %2 tespit etkinliği göstermiştir. Emerging Threats Kural Seti ise CVE referanslı zafiyet sömürü ataklarında %12 ve zararlı yazılım trafiklerinde %9 tespit etkinliği gösterirken, web uygulama ataklarında %13 tespit etkinliği göstermiştir.

Tüm kural setleri birlikte kullanıldığında %53 ile en etkin şekilde web uygulama (WebApp) atakları tespit edilmiştir. CVE referanslı zafiyet sömürü ataklarının tespit yüzdesi ise %20'ye çıkmıştır. Ancak zararlı yazılım (Malware) trafiklerinin tespitinde %13 tespit etkinliği ile belirgin bir artış gözlenmemiştir.

4.2.2.2. Test 1.6: Zararlı yazılımların tehdit aktörlerine göre analizi

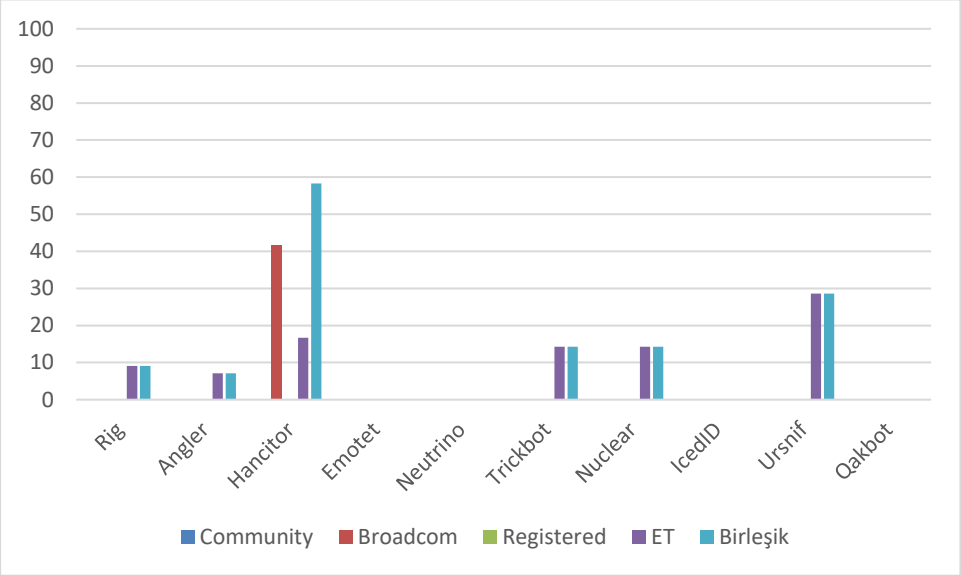
Bu deneyde Suricata IDS'in zararlı yazılım trafiklerinde yer alan tehdit aktörlerini tespit etmedeki etkinliğinin ölçülmesi hedeflenmiştir. Deneyde kullanılan zararlı trafiklerde 52 farklı tehdit aktörünün yer aldığı saptanmış olsa da istatistiki hesaplamalara altı ve daha fazla sayıda trafiğin içinde yer alan tehdit aktörleri dahil edilmiştir. Aşağıdaki çizelgede tehdit aktörleri, kaç adet trafik paketinde yer aldıkları ve kural setlerinin bu trafiklerden kaç tanesine alarm oluşturduğu bilgileri yer almaktadır. En sağ sütunda ise tüm kural setleri birlikte kullanıldığında oluşan alarm sayısı gösterilmiştir.

Çizelge 4.9 : Suricata-Tehdit aktörleri.

TEHDİT AKTÖRÜ	PAKET SAYISI	Community	Broadcom	Registered	Emerging Threats	Hepsi
Rig	22	0	0	0	2	2
Angler	14	0	0	0	1	1
Hancitor	12	0	5	0	2	7
Emotet	9	0	0	0	0	0
Neutrino	8	0	0	0	0	0
Nuclear	7	0	0	0	1	1
Trickbot	7	0	0	0	1	1
IcedID	7	0	0	0	0	0
Ursnif	7	0	0	0	2	2
Qakbot	6	0	0	0	0	0
Toplam:	99	0	5	0	9	14

99 atak trafiğinden Community ve Registered Kural Seti hiçbir atak trafiğini tespit edemezken, Broadcom kuralları 5 atak trafiğini ve Emerging Threats Kural Seti ise 9 atak trafiğini tespit etmiştir. Tüm kural setleri birlikte kullanıldığında ise 14 atak trafiği tespit edilmiştir.

Suricata'nın zararlı yazılım tehdit aktörlerini tespit etkinliği ise aşağıdaki grafikte gösterilmiştir (Şekil 4.7).



Şekil 4.7 : Suricata - Tehdit aktörlerine göre etkinlik analizi.

Zararlı yazılım trafikleri Suricata'da analiz edildiğinde Community ve Registered Kural Seti ile hiçbir atak trafiğinin tespit edilemediği, Broadcom kurallarının sadece %42 ile Hancitor tehdit aktörünü tespit ettiği gözlenmiştir. Emerging Threats Kural Seti'nde en yüksek tespit oranının ise %29 ile Ursnif tehdit aktöründe olduğu gözlenmiştir. En yüksek etkinlikte tespit edilen tehdit aktörü tüm kural setleri birlikte kullanıldığında %58 ile Hancitor olmuştur. Hiçbir tehdit aktöründe %100 tespit etkinliği sağlanamamıştır.

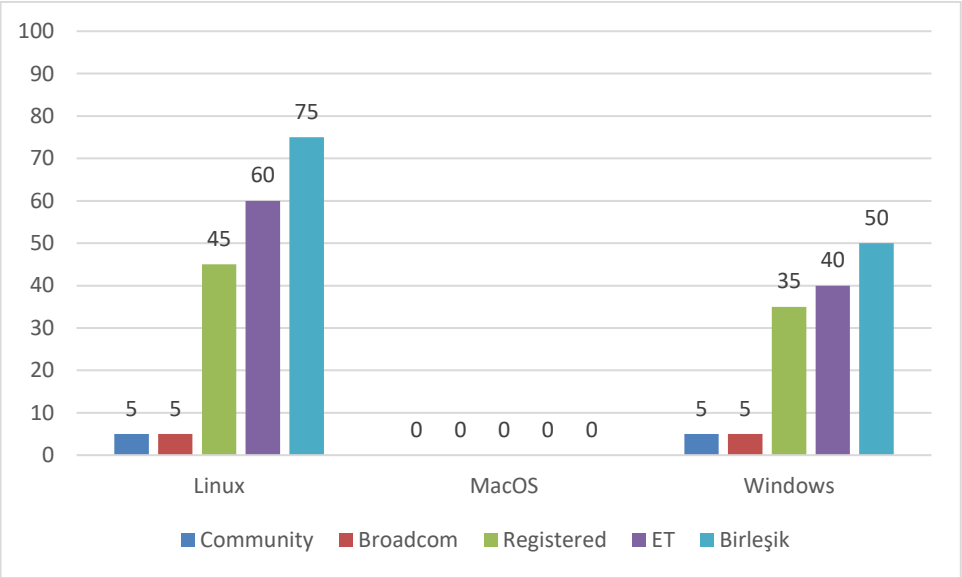
4.2.2.3. Test 1.7: CVE referanslı atakların işletim sistemlerine göre analizi

Bu deneyde Suricata IDS'in açık kural setleri ile Linux, Windows ve MacOS işletim sistemlerine yönelik yapılan atakların tespit etkinliğinin ölçülmesi hedeflenmiştir. Bu amaçla öncelikle CVE referanslı zafiyet sömürü ataklarının hangi işletim sistemlerine yönelik yapıldığı tespit edilmiş ve analiz için her işletim sistemine ait trafiklerden 20

adet seçilmiştir. İşletim sistemleri ve paket sayıları Çizelge 4.10'da gösterilmiştir. Deney sonuçları ise Şekil 4.8'de gösterilmiştir.

Çizelge 4.10 : Suricata - İşletim sistemleri.

İŞLETİM SİSTEMİ	PAKET SAYISI
LINUX	20
WINDOWS	20
MACOS	20



Şekil 4.8 : Suricata - İşletim sistemlerinde kural seti bazında analiz.

Deney sonuçları incelendiğinde Community Kural Seti'nin %5 ile Linux ve Windows işletim sistemlerinde tespit etkinliği gösterdiği, Broadcom kurallarının %5 ile Linux'ta ve Windows'ta tespit etkinliği gösterdiği, Registered Kural Seti'nin %45 ile Linux'ta ve %35 ile Windows'ta tespit etkinliği gösterdiği, en yüksek tespit etkinliğinin ise Emerging Threats Kural Seti ile elde edildiği gözlenmiştir. Emerging Threats Kural Seti Windows işletim sistemine ait atakların %60'ını tespit ederken, Linux'a ait atakların %40'ını tespit etmiştir. MacOS işletim sisteminde hiçbir kural seti atakları tespit edememiştir. Bu nedenle testler tekrarlanmış, ancak aynı sonuç elde edilmiştir.

Tüm kural setleri birlikte kullanıldığında hem Linux hem Windows işletim sisteminin tespit etkinliği artmıştır. Linuxun tespit etkinliği %60'dan %75'e, Windows'un tespit

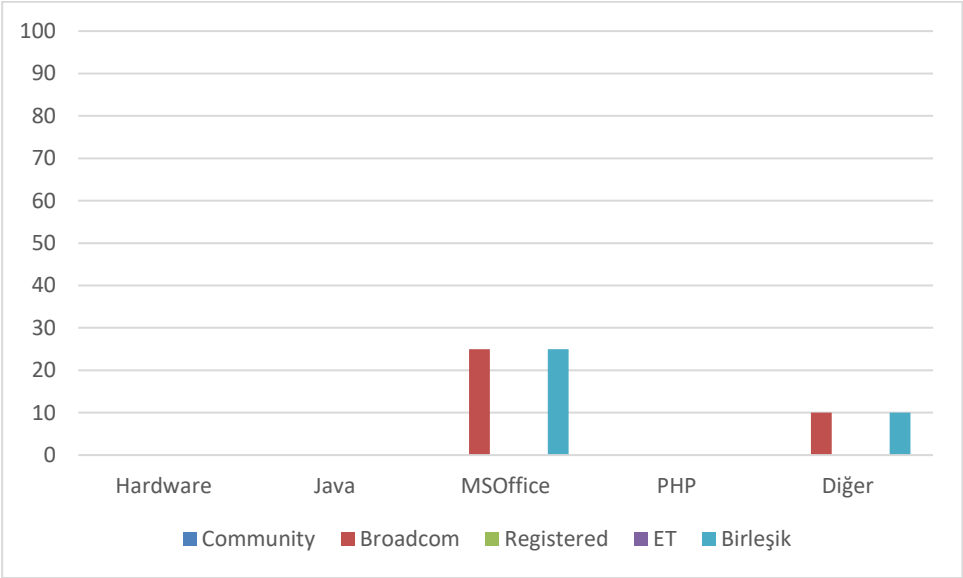
etkinliği %40'tan %50'ye yükselmiştir. MacOS'a yönelik hiçbir atak tespit edilememiştir.

4.2.2.4. Test 1.8: CVE referanslı atakların platformlara göre analizi

Bu deneyde Suricata IDS'in açık kural setleri ile Java, PHP, MSOffice, Hardware (Donanım) platformları ve Diğer Platformlara yönelik yapılan atakların tespit etkinliğinin ölçülmesi hedeflenmiştir. Bu amaçla öncelikle CVE referanslı zafiyet sömürü ataklarının hangi platformlara yönelik yapıldığı tespit edilmiş ve analiz için her platforma ait trafiklerden 20 adet seçilmiştir. Platformlar ve paket sayıları Çizelge 4.11'de gösterilmiştir. Deneylerden elde edilen sonuçlar ise Şekil 4.9'da gösterilmiştir.

Çizelge 4.11 : Suricata – Platformlar.

PLATFORM	PAKET SAYISI
JAVA	20
PHP	20
MSOFFICE	20
HARDWARE (DONANIM)	20
DİĞER	20



Şekil 4.9 : Suricata - Platformlarda kural seti bazında analiz.

Deney sonuçları incelendiğinde. Community, Registered ve Emerging Threats kural setleri hiçbir platformda atak tespit etmezken, Broadcom kuralları MSOffice

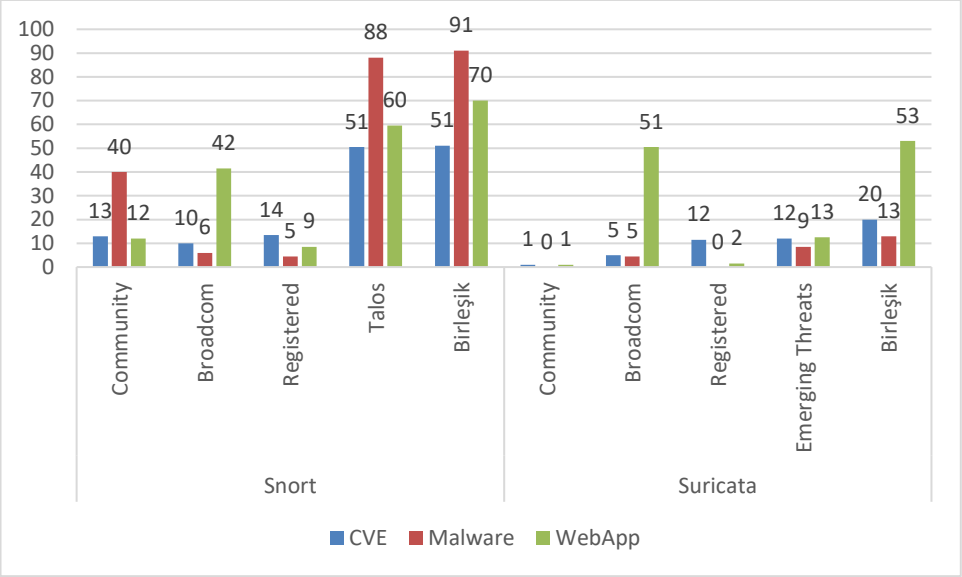
platformunda %25 tespit etkinliği göstermiş ve Diğer platformlarda ise %10 tespit etkinliği göstermiştir.

Tüm kural setleri birlikte kullanıldığında platformlardaki maksimum tespit etkinliği değişmemiş ve Broadcom kurallarının tespit etkinliği ile aynı kalmıştır.

4.2.3. Snort ve Suricata saldırı tespit sistemlerinin karşılaştırılması

4.2.3.1. Test 1.9: Tüm kural setlerinin tek başına ve birlikte kullanılması

Bu deneyde Snort ve Suricata IDS'lerinde tüm kural setleri tek başına kullanıldığında ve birlikte kullanıldığında elde edilen sonuçların analiz edilmesi amaçlanmıştır. Deneylerden elde edilen sonuçlar Şekil 4.10'da gösterilmiştir.



Şekil 4.10 : Snort ve Suricata – Karşılaştırmalı tespit etkinliği.

Tüm kural setleri tek başına kullanıldığında aşağıdaki sonuçlar elde edilmiştir:

- Community Kural Seti ile Snort'ta zararlı yazılım trafiklerinde %40 tespit etkinliği elde edilirken, CVE referanslı zafiyet sömürü ataklarında %13, web uygulama ataklarında ise %12 tespit etkinliği elde edilmiştir. Suricata'da zararlı yazılım trafiklerinde hiçbir atak tespit edilemezken, CVE referanslı zafiyet sömürü ataklarında ve web uygulama ataklarında %1 tespit etkinliği elde edilmiştir.

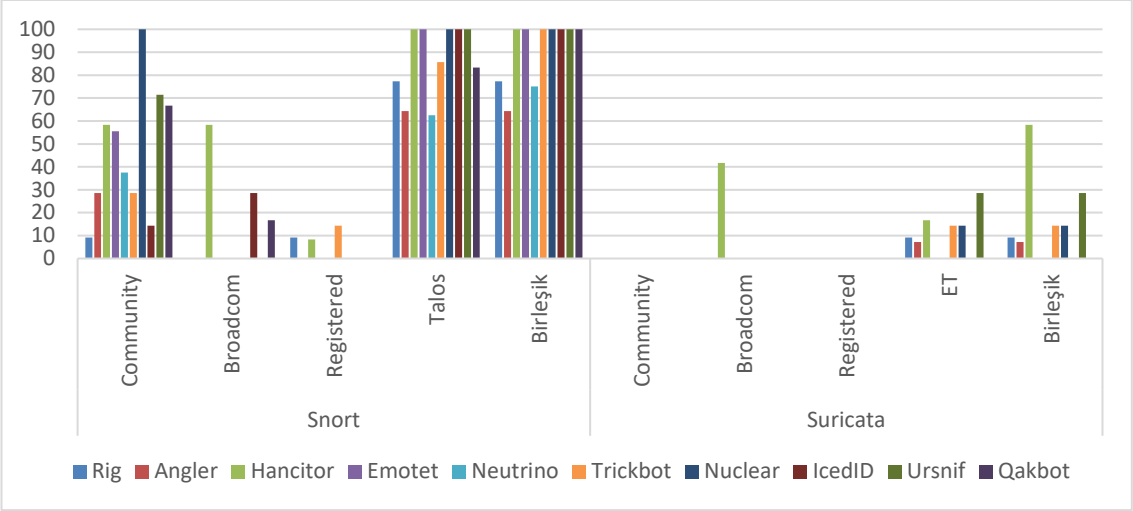
- Broadcom Kural Seti ile Snort'ta web uygulama ataklarında %42 tespit etkinliği elde edilirken, CVE referanslı zafiyet sömürü ataklarında %10 ve zararlı yazılım trafiklerinde %6 tespit etkinliği elde edilmiştir. Suricata'da ise web uygulama ataklarında %51 tespit etkinliği elde edilirken, CVE referanslı zafiyet sömürü ataklarında ve zararlı yazılım trafiklerinde %5 tespit etkinliği elde edilmiştir.
- Snort'ta kullanılan Talos Kural Seti ve Suricata'da kullanılan Emerging Threats Kural Seti'nin etkinliği karşılaştırıldığında Snort'ta Talos Kural Seti'nin zararlı yazılım trafiklerinde %88 tespit etkinliği gösterdiği, web uygulama ataklarında %60 ve CVE referanslı zafiyet sömürü ataklarında %51 tespit etkinliği gösterdiği gözlenmiştir. Suricata'da Emerging Threats Kural Seti'nin ise web uygulama ataklarında %13, CVE referanslı zafiyet sömürü ataklarında %12 ve zararlı yazılım trafiklerinde %9 tespit etkinliği gösterdiği görülmüştür.
- Registered Kural Seti ile Snort'ta CVE referanslı zafiyet sömürü ataklarında %14, web uygulama ataklarında %9 ve zararlı yazılım trafiklerinde %5 tespit etkinliği elde edildiği gözlenmiştir. Suricata'da ise CVE referanslı zafiyet sömürü ataklarında %12 tespit etkinliği elde edilirken, web uygulama ataklarında %2 tespit etkinliği elde edildiği, zararlı yazılım trafiklerinde ise hiçbir atağın tespit edilemediği görülmüştür.
- Tüm atak türlerinde en yüksek tespit etkinliğinin Snort'ta Talos Kural Seti ile elde edildiği görülmüştür.

Tüm kural setleri birlikte kullanıldığında aşağıdaki sonuçlar elde edilmiştir:

- Snort'ta zararlı yazılım trafiklerinde %91 tespit etkinliği elde edilirken, web uygulama ataklarında %70 ve CVE referanslı zafiyet sömürü ataklarında %51 tespit etkinliği elde edilmiştir. Suricata'da ise web uygulama ataklarında %53 tespit etkinliği elde edilirken, CVE referanslı zafiyet sömürü ataklarında %20 ve zararlı yazılım trafiklerinde %13 tespit etkinliği elde edildiği görülmüştür.
- Tüm kural setleri birlikte kullanıldığında tüm atak türlerinde Snort'un tespit etkinliğinin daha yüksek olduğu gözlenmiştir.

4.2.3.2. Test 1.10: Zararlı yazılımların tehdit aktörlerine göre analizi

Bu deneyde Snort ve Suricata IDS'lerinin zararlı yazılım trafiklerinde yer alan tehdit aktörlerini tespit etmedeki etkinliklerinin karşılaştırılması hedeflenmiştir. Deneylerden elde edilen sonuçlar Şekil 4.11'de gösterilmiştir.



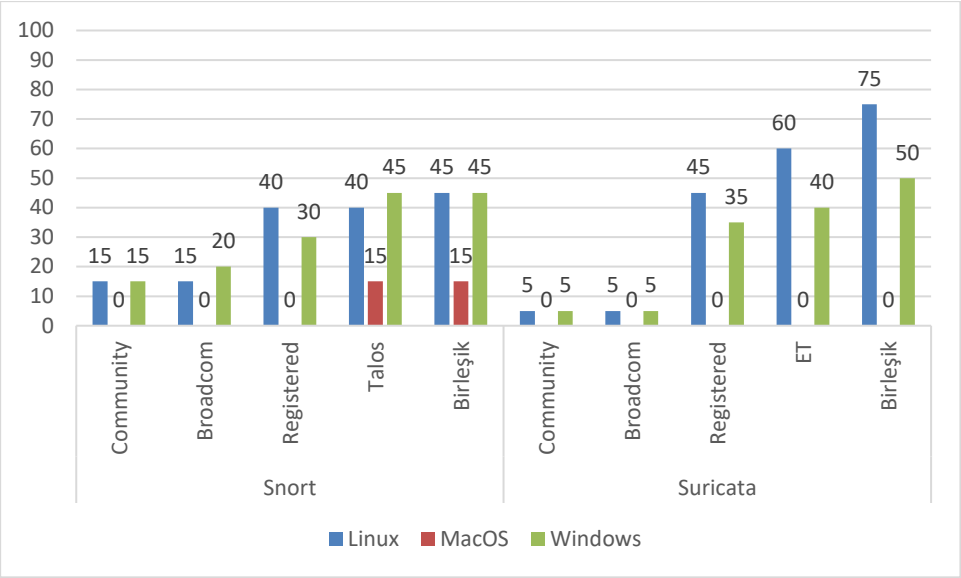
Şekil 4.11 : Snort ve Suricata - Tehdit aktörlerinin tespit etkinliği.

Atak trafikleri tehdit aktörlerine göre analiz edildiğinde Snort'ta Community Kural Seti ile Nuclear tehdit aktöründe %100 tespit etkinliği elde edildiği, Talos Kural Seti ile ise Hancitor, Emotet, Nuclear, IcedID ve Ursnif tehdit aktörlerinde %100 tespit etkinliği elde edildiği görülmüştür. Suricata'da ise hiçbir kural setinde %100 tespit etkinliği elde edilmemiştir.

Tüm kural setleri birlikte kullanıldığında ise Snort'ta on tehdit aktöründen yedisinde %100 tespit etkinliği elde edildiği (Hancitor, Emotet, Trickbot, Nuclear, IcedID, Ursnif ve Qakbot), Suricata'da ise hiçbir tehdit aktöründe %100 tespit etkinliği elde edilmediği görülmüştür.

4.2.3.3. Test 1.11: CVE referanslı atakların işletim sistemlerine göre analizi

Bu deneyde Snort ve Suricata IDS'leri kullanılarak açık kural setleri ile MacOS, Linux ve Windows işletim sistemlerine yönelik yapılan atakların tespit etkinliklerinin karşılaştırılması hedeflenmiştir. Deneylerden elde edilen sonuçlar Şekil 4.12'de gösterilmiştir.



Şekil 4.12 : Snort ve Suricata - İşletim sistemlerine göre tespit etkinliği.

Deney sonuçları incelendiğinde aşağıdaki bulgular elde edilmiştir:

- Community Kural Seti kullanılarak Snort'ta Linux ve Windows işletim sistemlerinde %15 tespit etkinliği elde edildiği, MacOS'ta hiçbir atağın tespit edilemediği görülmüştür. Suricata'da ise Linux ve Windows işletim sistemlerinde %5 tespit etkinliği elde edildiği, MacOS'ta hiçbir atağın tespit edilemediği görülmüştür.

- Broadcom kuralları kullanılarak Snort'ta Windows işletim sisteminde %20 ve Linux'ta %15 tespit etkinliği elde edildiği, MacOS'ta hiçbir atağın tespit edilemediği görülmüştür. Suricata'da ise Linux ve Windows'ta %5 tespit etkinliği elde edildiği, MacOS'ta hiçbir atağın tespit edilemediği görülmüştür.

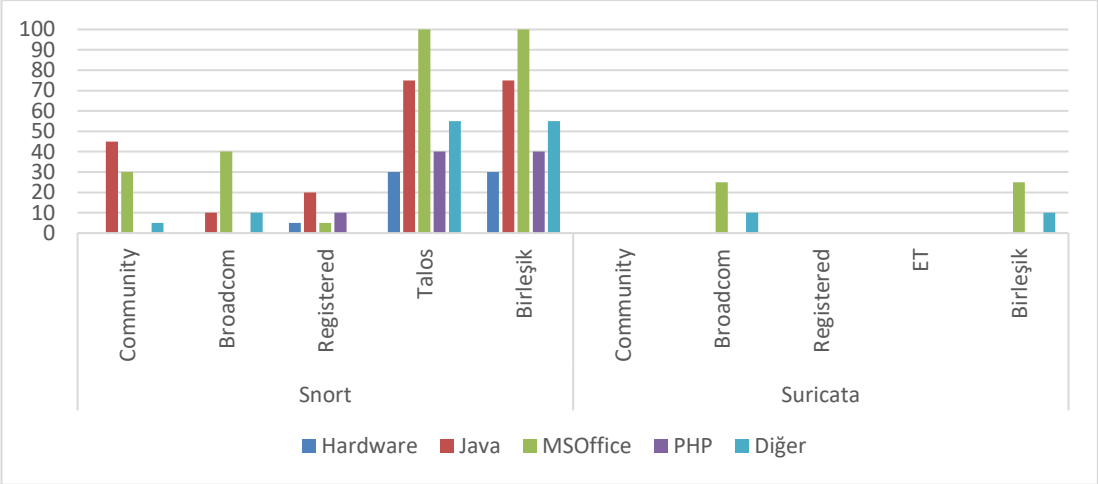
- Registered Kural Seti kullanılarak Snort'ta Linux işletim sisteminde %40 ve Windows'ta %30 tespit etkinliği elde edildiği, MacOS'ta hiçbir atağın tespit edilemediği görülmüştür. Suricata'da ise Linux işletim sisteminde %45 ve Windows'ta %35 tespit etkinliği elde edildiği görülmüştür. MacOS'ta ise hiçbir atağın tespit edilemediği görülmüştür.

- Talos Kural Seti kullanılarak Snort'ta Windows işletim sisteminde %45, Linux'ta %40 ve MacOS'ta %15 tespit etkinliği elde edildiği görülmüştür. Suricata'da ise Emerging Threats Kural Seti ile Linux'ta %60 ve Windows'da %40 tespit etkinliği elde edildiği, MacOS'ta hiçbir atağın tespit edilemediği görülmüştür.

- Tüm kural setleri birlikte kullanıldığında ise Snort'ta Linux ve Windows işletim sistemlerinde %45 tespit etkinliği elde edildiği, MacOS'ta ise %15 tespit etkinliği elde edildiği görülmüştür. Suricata'da Linux işletim sisteminde %75 tespit etkinliği ve Windows'ta %50 tespit etkinliği elde edildiği görülmüştür. MacOS'ta hiçbir atağın tespit edilemediği görülmüştür.

4.2.3.4. Test 1.12: CVE referanslı atakların platformlara göre analizi

Bu deneyde Snort ve Suricata IDS'in açık kural setleri ile Java, PHP, MSOffice, Donanım (Hardware) platformları ve Diğer Platformlara yönelik yapılan atakları tespit etkinliğinin ölçülmesi hedeflenmiştir. Deneylerden elde edilen sonuçlar Şekil 4.13'te gösterilmiştir.



Şekil 4.13 : Snort ve Suricata - Platformlara göre tespit etkinliği.

Deney sonuçları incelendiğinde aşağıdaki bulgular elde edilmiştir:

- Community Kural Seti kullanılarak Snort'ta Java'da %45, MSOffice'de %30 ve Diğer platformlarda %5 tespit etkinliği elde edilirken Donanım ve PHP platformlarına yönelik hiçbir atağın tespit edilemediği görülmüştür. Suricata'da ise hiçbir platforma yönelik atağın tespit edilemediği görülmüştür.
- Broadcom kuralları kullanılarak Snort'ta MSOffice'de %40, Java ve Diğer platformlarda %10 tespit etkinliği elde edildiği; Donanım ve PHP platformlarına yönelik hiçbir atağın tespit edilemediği görülmüştür. Suricata'da ise MSOffice'de %25 ve Diğer platformlarda %10 tespit etkinliği elde edildiği; Donanım, Java ve PHP platformlarına yönelik hiçbir atağın tespit edilemediği görülmüştür.

- Registered Kural Seti kullanılarak Snort'ta Java'da %20, PHP'de %10, Donanım ve MSOffice'de %5 tespit etkinliği elde edildiği; Diğer platformlara yönelik hiçbir atağın tespit edilemediği görülmüştür. Suricata'da ise hiçbir platforma yönelik atağın tespit edilemediği görülmüştür.
- Talos Kural Seti kullanılarak Snort'ta MSOffice uygulamalarında %100 tespit etkinliği elde edildiği; Java'da %75, Diğer platformlarda %55, PHP'de %40 ve Donanımda %30 tespit etkinliği elde edildiği görülmüştür. Emerging Threats Kural Seti ile Suricata'da ise hiçbir platforma yönelik atağın tespit edilemediği görülmüştür.
- Tüm kural setleri birlikte kullanıldığında Snort'ta MSOffice uygulamalarında %100 tespit etkinliği elde edildiği; Java'da %75, Diğer platformlarda %55, PHP'de %40 ve Donanımda %30 tespit etkinliği elde edildiği görülmüştür. Suricata'da ise MSOffice'de %25 ve Diğer platformlarda %10 tespit etkinliği elde edildiği; Donanım, Java ve PHP platformlarına yönelik hiçbir atağın tespit edilemediği görülmüştür.

4.2.4. Test 2: OSSEC ve Wazuh

Üzerine OSSEC ve Wazuh kurulmuş test makinelerine CVE referanslı zafiyet sömürü atakları ve web uygulama ataklarından oluşan 150 farklı atak gerçekleştirilmiştir. Her iki saldırı tespit sisteminde de atakların hiçbirinde alarm oluşmamıştır. Atak sayıları ve sonuçlar Çizelge 4.12'de gösterilmiştir.

Çizelge 4.12 : OSSEC ve Wazuh Atak Sayısı.

TOPLAM ATAK SAYISI:	150
TESPİT EDİLEN ATAK SAYISI:	0

Başarısız olan ataklarda alarm oluşmadığı için başarılı bir saldırı sonrasında gerçekleşecek adımlarda oluşacak OSSEC ve Wazuh alarmlarını simüle etmek amacıyla SSH bağlantısıyla test bilgisayarına bağlanılıp ardından çeşitli komutlar çalıştırılmıştır. Her iki saldırı tespit sisteminde de aynı alarmlar oluşmuştur. Komutlar ve bunlar sonucu oluşan alarmlar aşağıdaki çizelgede yer almaktadır (Çizelge 4.13):

Çizelge 4.13 : OSSEC ve Wazuh - Komut ve alarmlar.

KOMUTLAR	ÇIKTILAR
ssh testmachine@ip_address	First time user logged in. Accepted password for testmachine from 192.168.182.130 port 38590 ssh2
ssh testmachine@ip_address	Login session opened. pam_unix(sshd:session): session opened for user testmachine by (uid=0)
sudo cat /etc/shadow	Successful sudo to ROOT executed testmachine-ubuntu sudo: testmachine : TTY=pts/4 ; PWD=/home/testmachine ; USER=root ; COMMAND=/bin/cat /etc/shadow
sudo su	User successfully changed UID to root. testmachine-ubuntu su[10345]: + /dev/pts/4 root:root
sudo su	Login session opened testmachine-ubuntu su[10345]: pam_unix(su:session): session opened for user root by testmachine(uid=0)
useradd hacker	New group added to the system testmachine-ubuntu useradd[10370]: new group: name=hacker, GID=1004
useradd hacker	New user added to the system testmachine-ubuntu useradd[10370]: new user: name=hacker, UID=1004, GID=1004, home=/home/hacker, shell=
passwd hacker	User changed password. testmachine-ubuntu passwd[10542]: pam_unix(passwd:chauthtok): password changed for hacker
su hacker	First time (su) is executed by user testmachine-ubuntu su[10604]: + /dev/pts/4 root:hacker

4.3. Test 3: Zararsız Trafik

Deneylerde Snort ve Suricata IDS'lerinin zararsız trafiklerde yanlış pozitif alarm oluşturma miktarlarının karşılaştırılması hedeflenmektedir. Zararsız trafik olarak toplamda 82 GB boyutunda 38 farklı pcap dosyası kullanılmıştır. Deneylerde kullanılmak üzere aşağıdaki veri setlerinden yararlanılmıştır:

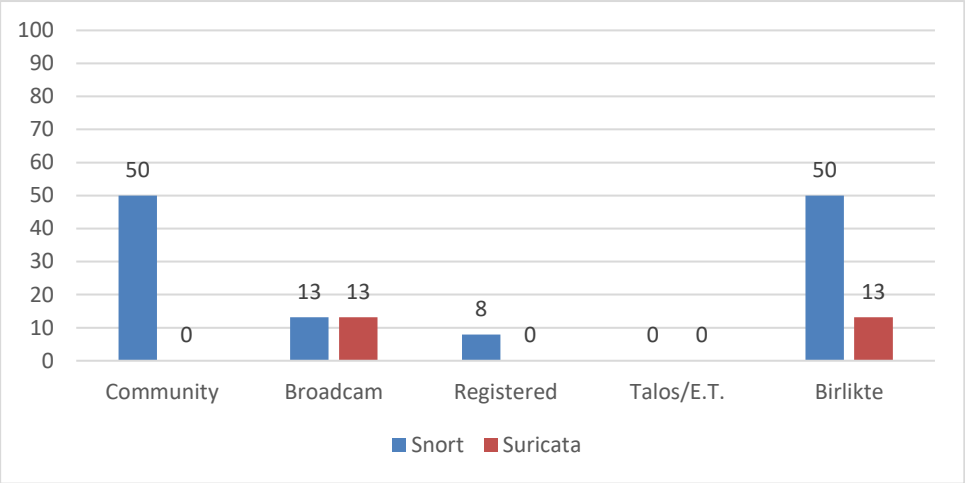
- CIRA-CIC-DoHBrw-2020 Data Set [42] [43]
- Intrusion Detection Evaluation Dataset (CIC-IDS2017) [44] [45]
- Intrusion Detection Evaluation Dataset (ISCXIDS2012) [46]

Testlerde kullanılan zararsız trafiklerin listesi Ek 8'de yer almaktadır.

4.3.1. Snort ve Suricata saldırı tespit sistemlerinin karşılaştırılması

4.3.1.1. Tüm kural setlerinin tek başına ve birlikte kullanılması

Bu deneyde Snort ve Suricata IDS'lerinin tüm kural setleri ayrı olarak ve birlikte kullanıldığında yanlış pozitif alarm oluşturma miktarlarının karşılaştırılması hedeflenmiştir. Deneylerden elde edilen sonuçlar Şekil 4.14'te gösterilmiştir.



Şekil 4.14 : Kural setlerinin yanlış pozitif alarm oluşturma yüzdesi.

Deney sonucunda aşağıdaki bulgular elde edilmiştir:

- Community Kural Seti Snort'ta %50 yanlış pozitif alarm oluştururken Suricata'da hiçbir yanlış pozitif alarm oluşturmamıştır.
- Broadcom kuralları hem Snort hem de Suricata'da %13 yanlış pozitif alarm oluşturmuştur.
- Registered Kural Seti Snort'ta %8 yanlış pozitif alarm oluştururken Suricata'da hiçbir yanlış pozitif alarm oluşturmamıştır.
- Talos ve Emerging Threats Kural Seti hiçbir yanlış pozitif alarm oluşturmamıştır.
- Tüm kural setleri birlikte kullanıldığında Snort'ta %50 yanlış pozitif alarm oluşurken Suricata'da %13 yanlış pozitif alarm oluşmuştur.

4.4. Ağ Tabanlı Saldırı Tespit Sistemlerinde Hata Matrisi Analizi

4.4.1. Community kural seti

4.4.1.1. Snort IDS

Snort Saldırı Tespit Sistemi ve Community Kural Seti ile yapılan testler sonucu elde edilen Hata Matrisi aşağıda yer almaktadır (Çizelge 4.14).

Çizelge 4.14 :Community - Snort Hata Matrisi.

Hata Matrisi		Tespit Sonucu		Toplam
		Zararsız	Zararlı	
Gerçek Sonuç	Zararsız	19	19	38
	Zararlı	470	130	600
Toplam		489	149	

4.4.1.2. Suricata IDS

Suricata Saldırı Tespit Sistemi ve Community Kural Seti ile yapılan testler sonucu elde edilen Hata Matrisi aşağıda yer almaktadır (Çizelge 4.15).

Çizelge 4.15 : Community - Suricata Hata Matrisi.

Hata Matrisi		**Tespit Sonucu**		**Toplam**
		Zararsız	**Zararlı**	
Gerçek Sonuç	**Zararsız**	38	0	38
	Zararlı	596	4	600
Toplam		634	4	

4.4.2. Broadcom kuralları

4.4.2.1. Snort IDS

Snort Saldırı Tespit Sistemi ve Broadcom Kuralları ile yapılan testler sonucu elde edilen Hata Matrisi aşağıda yer almaktadır (Çizelge 4.16).

Çizelge 4.16 : Broadcom - Snort Hata Matrisi.

Hata Matrisi		Tespit Sonucu		Toplam
		Zararsız	Zararlı	
Gerçek Sonuç	Zararsız	33	5	38
	Zararlı	485	115	600
Toplam		518	120	

4.4.2.2. Suricata IDS

Suricata Saldırı Tespit Sistemi ve Broadcom Kuralları ile yapılan testler sonucu elde edilen Hata Matrisi aşağıda yer almaktadır (Çizelge 4.17).

Çizelge 4.17 : Broadcom - Suricata Hata Matrisi.

Hata Matrisi		Tespit Sonucu		Toplam
		Zararsız	Zararlı	
Gerçek Sonuç	Zararsız	33	5	38
	Zararlı	480	120	600
Toplam		513	125	

4.4.3. Registered kural seti

4.4.3.1. Snort IDS

Snort Saldırı Tespit Sistemi ve Registered Kural Seti ile yapılan testler sonucu elde edilen Hata Matrisi aşağıda yer almaktadır (Çizelge 4.18).

Çizelge 4.18 : Registered - Snort Hata Matrisi.

Hata Matrisi		Tespit Sonucu		Toplam
		Zararsız	Zararlı	
Gerçek Sonuç	Zararsız	35	3	38
	Zararlı	547	53	600
Toplam		582	56	

4.4.3.2. Suricata IDS

Suricata Saldırı Tespit Sistemi ve Registered Kural Seti ile yapılan testler sonucu elde edilen Hata Matrisi aşağıda yer almaktadır (Çizelge 4.19).

Çizelge 4.19 : Registered - Suricata Hata Matrisi.

Hata Matrisi		**Tespit Sonucu**		**Toplam**
		Zararsız	**Zararlı**	
Gerçek Sonuç	**Zararsız**	38	0	38
	Zararlı	574	26	600
Toplam		612	26	

4.4.4. Talos / Emerging Threats kural setleri

4.4.4.1. Snort IDS

Snort Saldırı Tespit Sistemi ve Talos Kural Seti ile yapılan testler sonucu elde edilen Hata Matrisi aşağıda yer almaktadır (Çizelge 4.20).

Çizelge 4.20 : Talos - Snort Hata Matrisi.

Hata Matrisi		**Tespit Sonucu**		**Toplam**
		Zararsız	**Zararlı**	
Gerçek Sonuç	**Zararsız**	38	0	38
	Zararlı	204	396	600
Toplam		242	396	

4.4.4.2. Suricata IDS

Suricata Saldırı Tespit Sistemi ve Emerging Threats Kural Seti ile yapılan testler sonucu elde edilen Hata Matrisi aşağıda yer almaktadır (Çizelge 4.21).

Çizelge 4.21 : Emerging Threats - Suricata Hata Matrisi.

Hata Matrisi		**Tespit Sonucu**		**Toplam**
		Zararsız	**Zararlı**	
Gerçek Sonuç	**Zararsız**	38	0	38
	Zararlı	534	66	600
Toplam		572	66	

4.4.5. Tüm kural setleri birlikte

4.4.5.1. Snort IDS

Snort Saldırı Tespit Sistemi ve tüm kural setleri birlikte kullanılarak yapılan testler sonucu elde edilen Hata Matrisi aşağıda yer almaktadır (Çizelge 4.22).

Çizelge 4.22 : Tüm kural setleri birlikte - Snort Hata Matrisi.

Hata Matrisi		**Tespit Sonucu**		**Toplam**
		Zararsız	**Zararlı**	
Gerçek Sonuç	**Zararsız**	19	19	38
	Zararlı	176	424	600
Toplam		195	443	

4.4.5.2. Suricata IDS

Suricata Saldırı Tespit Sistemi ve tüm kural setleri birlikte kullanılarak yapılan testler sonucu elde edilen Hata Matrisi aşağıda yer almaktadır (Çizelge 4.23).

Çizelge 4.23 : Tüm kural setleri birlikte - Suricata Hata Matrisi.

Hata Matrisi		**Tespit Sonucu**		**Toplam**
		Zararsız	**Zararlı**	
Gerçek Sonuç	**Zararsız**	33	5	38
	Zararlı	428	172	600
Toplam		461	177	

4.5. Kesinlik, Duyarlılık ve F-Skor Analizi

Yapılan testler sonucu elde edilen Kesinlik (K), Duyarlılık (D) ve F-Skor (F) değerleri aşağıdaki çizelgede yer almaktadır (Çizelge 4.24).

Çizelge 4.24 : Saldırı Tespit Sistemlerinin Kesinlik, Duyarlılık ve F-Skor Değerleri.

	Community		**Broadcom**		**Registered**		**Talos/E.T.**		**Birlikte**	
	Snort	**Suricata**	**Snort**	**Suricata**	**Snort**	**Suricata**	**Snort**	**Suricata**	**Snort**	**Suricata**
K	0,87	1,00	0,96	0,96	0,95	1,00	1,00	1,00	0,96	0,97
D	0,22	0,01	0,19	0,20	0,09	0,04	0,66	0,11	0,71	0,29
F	0,35	0,01	0,32	0,33	0,16	0,08	0,80	0,20	0,81	0,44

Yapılan analizler sonucunda Community, Broadcom ve Registered Kural Seti'nde Suricata'nın Kesinlik değeri daha yüksek çıkarken; Talos ve Emerging Threats Kural Setleri için yapılan analizde her iki Saldırı Tespit Sistemi'nin Kesinlik değeri aynı çıkmıştır. Yani pozitif tahmin edilen değerlerin pozitif çıkma oranı kural setlerinin üçünde Suricata'da daha yüksek olmuştur. Community, Registered ve Talos Kural Setleri'nde Snort'un Duyarlılık değeri daha yüksekken Broadcom Kuralları'nda Suricata'nın Duyarlılık değeri daha yüksek çıkmıştır. Yani pozitif olarak tahmin edilmesi gereken değerlerin pozitif çıkma oranı kural setlerinin üçünde Snort'ta daha yüksektir. F-Skor değeri Community, Registered ve Talos Kural Setleri'nde Snort'ta daha yüksek çıkarken, Broadcom Kuralları'nda az bir farkla Suricata'da yüksek çıkmıştır. Yani Snort trafikleri sınıflandırırken kural setlerinin üçünde daha etkin performans göstermiştir. Kesinlik, Duyarlılık ve F-Skor değerleri aşağıdaki grafikte de gösterilmiştir. Tüm kural setleri birlikte kullanıldığında ise Suricata'nın kesinlik değeri çok az farkla yüksek çıkarken Snort'un Duyarlılık ve F-Skor değeri daha yüksek çıkmıştır. Yani pozitif tahmin edilen değerlerin pozitif çıkma oranı Suricata'da daha yüksek çıkarken pozitif olarak tahmin edilmesi gereken değerlerin pozitif çıkma oranı Snort'ta daha yüksektir ve Snort trafikleri sınıflandırırken daha etkin performans göstermiştir. Kural setlerinin karşılaştırmalı analizi ise Çizelge 4.25'te gösterilmiştir.

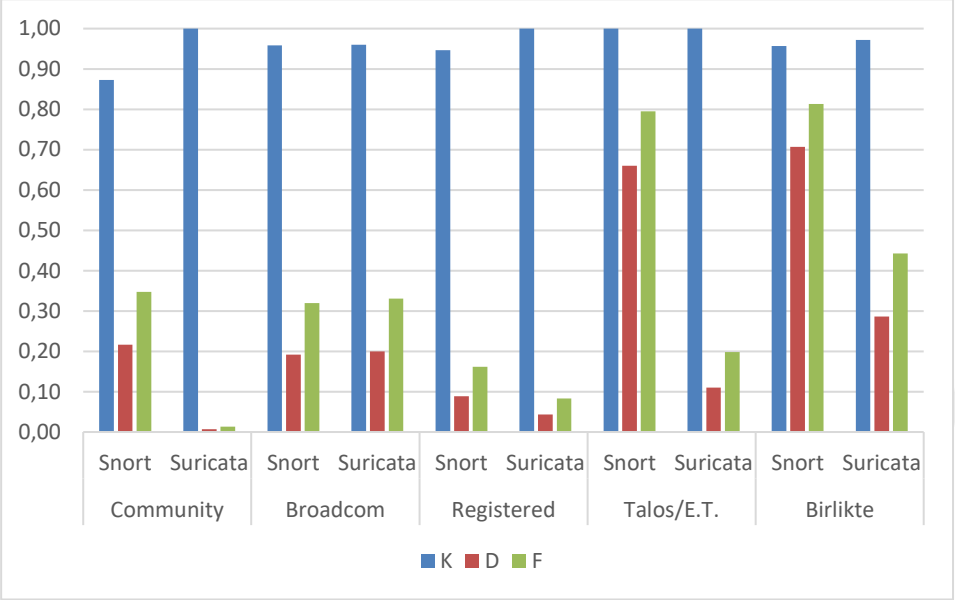
Çizelge 4.25 : Kural Setlerinin Kesinlik, Duyarlılık ve F-Skor Değerleri.

	Snort					**Suricata**				
	C	**B**	**R**	**T / E.T.**	**Birlikte**	**C**	**B**	**R**	**T / E.T.**	**Birlikte**
K	0,87	0,96	0,95	1,00	0,96	1,00	0,96	1,00	1,00	0,97
D	0,22	0,19	0,09	0,66	0,71	0,01	0,20	0,04	0,11	0,29
F	0,35	0,32	0,16	0,80	0,81	0,01	0,33	0,08	0,20	0,44

C: Community Kural Seti
B: Broadcom Kuralları
R: Registered Kural Seti
T: Talos Kural Seti
E.T.: Emerging Threats Kural Seti

Kural setleri karşılaştırmalı olarak analiz edildiğinde ise tüm kural setleri ayrı kullanıldığında Snort'ta en yüksek Kesinlik, Duyarlılık ve F-Skor değerleri Talos'ta çıkarken, Suricata'da ise Broadcom dışındaki kural setlerinin Kesinlik değerinin aynı ve daha yüksek olduğu, Broadcom kurallarının Duyarlılık ve F-Skor değerlerinin ise

daha yüksek çıktığı gözlenmiştir. Yani Snort'ta pozitif tahmin edilmesi gereken ve pozitif tahmin edilen değerlerde ve etkin sınıflandırmada Talos daha etkin sonuç gösterirken; Suricata'da ise pozitif tahmin edilen değerlerin pozitif çıkma oranı incelendiğinde en az etkinliğe sahip olan kurallar Broadcom kuralları olurken, pozitif tahmin edilmesi gereken değerlerin pozitif çıkma oranı ve etkin sınıflandırma metrikleri incelendiğinde Broadcom kuralları daha etkin sonuç göstermiştir. Elde edilen sonuçlar Şekil 4.15'te gösterilmiştir.



Şekil 4.15 : Kesinlik, Duyarlılık ve F-Skor Karşılaştırmalı Grafiği.

5. SONUÇ

Yapılan tüm deneyler sonucunda elde edilen bulgulardan yola çıkılarak aşağıdaki sonuçlar elde edilmiştir:

Kural setleri ayrı olarak kullanıldığında en etkin sonuçlar Talos Kural Seti ile Snort'ta elde edilmiştir. Tüm kural setleri birlikte kullanıldığında üç atak türünde de (Zararlı yazılım trafikleri, CVE referanslı zafiyet sömürü atakları, web uygulama atakları) Snort daha etkin sonuçlar vermiştir. Windows ve Linux işletim sistemine ait atak trafiklerinin tüm kural setleri birlikte kullanıldığında Suricata'da daha etkin tespit edildiği görülmüştür. Her iki saldırı tespit sistemi de MacOS işletim sistemine yönelik saldırıları tespit etmekte yetersiz kalmıştır. MacOS işletim sistemine yönelik atakların tespiti %15 ile sadece Talos Kural Seti ile Snort'ta elde edilmiştir. OSSEC ve Wazuh Saldırı Tespit Sistemleri ağ tabanlı saldırılarda alarm vermemekle birlikte, başarılı atak sonrası sistemde yapılan değişikliklerde alarm oluşturmakta ve kullanıcıların zararlı olmayan komutlarından da çok fazla yanlış pozitif alarm oluşturmaktadır.

Zararsız trafiklerde tüm kural setleri tek başına kullanıldığında Talos Kural Seti ve Emerging Threats Kural Seti yanlış pozitif alarm oluşturmamıştır. Broadcom kuralları az sayıda olmasına rağmen web uygulama ataklarına karşı etkili olmuştur, ancak ürettiği yanlış pozitif alarm sayısı da fazladır. Tüm kural setleri birlikte kullanıldığında ise yine Snort'un daha etkin olduğu gözlenmiştir. Ancak zararsız trafiklerde tüm kural setleri birlikte kullanıldığında Suricata daha az yanlış pozitif alarm oluşturmuştur.

Kesinlik, Duyarlılık ve F-Skor metrikleri analiz edildiğinde Community, Registered ve Talos Kural Setleri için Snort'un trafikleri Suricata'dan daha doğru sınıflandırdığı, Broadcom Kuralları'nda ise Suricata'nın az bir farkla daha doğru sınıflandırma yaptığı görülmüştür. Tüm kural setleri birlikte analiz edildiğinde ise pozitif tahmin edilen değerlerin pozitif çıkma oranları Suricata'da az bir farkla yüksek çıkarken pozitif

olarak tahmin edilmesi gereken değerlerin pozitif çıkma oranı Snort'ta daha yüksektir ve Snort trafikleri sınıflandırırken daha etkin performans göstermiştir.
Bu sonuçlar ışığında, ücretsiz ve açık kaynak kural seti kullanan organizasyonlar için Snort ve OSSEC / Wazuh Saldırı Tespit Sistemleri'nin birlikte kullanılması ve Snort içinde açık tüm kural setlerinin birlikte kullanılmasının tespit etkinliğini artıracağı görülmektedir.

5.1. Gelecek Çalışmalar için Öneriler

Bu çalışmanın ilerde geliştirilmesi amacıyla aşağıdaki uygulamalar yapılabilir:

- Daha çeşitli atak türlerinde trafikler kullanılması:
    - Zararlı yazılım ataklarının alt başlıklara ayrılarak incelenmesi (Virüs, Solucan, Truva Atı, Casus Yazılım, Fidye Yazılımı, Dosyasız Zararlı Yazılım vs.)
    - Parola Atakları
    - DoS (Servis Dışı Bırakma Saldırısı – Denial of Service) Atakları
    - Rootkit Trafikleri
    - Ağ Tarama Atakları
- Farklı saldırı tespit sistemlerinin de çalışmaya dahil edilmesi
    - Açık Kaynak olmayan Saldırı Tespit Sistemleri
    - Samhain HIDS
- Deneylerde kullanılması için otomatik bir atak üreteci geliştirilmesi (attack generator): Atak üreteci Saldırı Tespit Sistemi analizi için farklı türlerde atak trafikleri oluşturarak trafik oluşturma sürecini otomatikleştirecektir.
- MacOS işletim sistemine yönelik atakların etkin tespiti için açık kaynak saldırı tespit sistemi kuralları geliştirilmesi

KAYNAKLAR

[1] **S. Potteti ve N. Parati**, (June 2015). An Innovative Intrusion Detection System using SNORT for Cloud Environment, *International Journal of Innovative Research in Computer and Communication Engineering*, cilt 3, no. 6, pp. 5679-5687.

[2] **M. Akhlaq, F. Alserhani, I. Awan, J. Mellor, A. J. Cullen ve A. Al-Dhelaan**, (2011). Implementation and Evaluation of Network Intrusion Detection Systems, Next Generation Internet, LNCS, pp. 988-1016.

[3] **J. Beale, A. B. Baker ve J. Esler,** (2007). Snort IDS and IPS Toolkit, New York: Syngress.

[4] Url 1: https://cve.mitre.org/. alındığı tarih: 31 07 2021.

[5] **J. S. Whitea, T. T. Fitzsimmons ve J. N. Matthews**, (2013). Quantitative Analysis of Intrusion Detection Systems: Snort and Suricata, International Society for Optics and Photonics.

[6] **K. Thongkanchorn, S. Ngamsuriyaroj ve V. Visoottiviseth**, (2013). Evaluation Studies of Three Intrusion Detection Systems under Various Attacks and Rule Sets, IEEE International Conference of IEEE Region 10 (TENCON), pp. 1-4.

[7] **M. F. Ridho, F. Yasin ve Y. Sulistyo**, (2014). Analysis And Evaluation Snort, Bro, And Suricata as Intrusion Detection System Based on Linux Server, Naskah_Publikasi.

[8] **H. Alnabulsi, M. R. İslam ve Q. Mamun**, (2014). Detecting SQL injection attacks using SNORT IDS, Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE), pp. 1-7.

[9] **M. Hänninen**, (December 2019). Open source intrusion detection systems evaluation for small and medium-sized enterprise environments, Master's thesis.

[10] **H. Alyami, T. J. Ansari, A. Alharbi, W. Alosaimi, M. Alshammari, D. Pandey, A. Agrawal, R. Kumar ve R. A. Khan**, (2022). Effectiveness Evaluation of Different IDSs Using Integrated Fuzzy MCDM Model, Electronics, cilt 11, no. 6, p. 859.

[11] **K. Scarfone ve P. Mell**, (February 2007). NIST Special Publication 800-94: Guide to Intrusion Detection and Prevention Systems, National Institute of Standards and Technology (NIST).

[12] **A. P. Singh ve M. D. Singh**, (2014). Analysis of Host-Based and Network-Based Intrusion Detection System, I.J. Computer Network and Information Security, pp. 41-47.

[13] **P. d. Boer ve M. Pels**, (2005). Host-based Intrusion Detection Systems, Revision 1.10, pp. 5-7.

[14] **T. Holland**, (2020). Understanding IPS and IDS:Using IPS and IDS together for Defense in Depth, SANS Institute.

[15] **V. Kumar ve O. P. Sangwan**, (2012). Signature Based Intrusion Detection System Using SNORT, International Journal of Computer Applications & Information Technology , cilt 1, no. 3, pp. 35-41.

[16] **V. Jyothsna, V. V. R. Prasad ve K. M. Prasad**, (2011). A Review of Anomaly based Intrusion Detection Systems,» International Journal of Computer Applications, cilt 28, no. 7, pp. 26-35.

[17] Url 2: https://www.snort.org/. alındığı tarih: 31 07 2021.

[18] **N. Khamphakdee, N. Benjamas ve S. Saiyod**, (2014). Improving Intrusion Detection System Based on Snort Rules for Network Probe Attack Detection, 2nd International Conference on Information and Communication Technology (ICoICT).

[19] **İ. Gündoğdu, S. Özarslan ve A. A. Selçuk**, (2021). Snort Saldırı Tespit Sisteminde Kullanılan Açık Kural Setlerinin Etkinlik Analizi, 29th Signal Processing and Communications Applications Conference (SIU).

[20] **B. M. Beigh**, (2015). Framework for choosing best intrusion detection system, BIJIT - BVICAM's International Journal of Information Technology, cilt 7, no. 1, pp. 821-826.

[21] **A. Gupta ve L. S. Sharma**, (2020). Performance Evaluation of Snort and Suricata Intrusion Detection Systems on Ubuntu Server, Proceedings of ICRIC 2019, pp. 811-821, Springer.

[22] Url 3:: https://suricata.io/. alındığı tarih: 31 07 2021.

[23] **J. Timofte**, (2008). Intrusion Detection using Open Source Tools, Revista Informatica Economică nr.2, pp. 75-79.

[24] Url 4: https://www.ossec.net/. alındığı tarih: 31 07 2021.

[25] Url 5: https://wazuh.com/. alındığı tarih: 10 10 2021.

[26] Url 6: https://www.snort.org/faq/what-are-community-rules. alındığı tarih: 31 07 2021.

[27] Url 7: https://www.snort.org/documents/registered-vssubscriber. alındığı tarih: 23 02 2021.

[28] Url 8: https://community.broadcom.com/symantecenterprise/communities/communityhome/librarydocuments/viewdocument?DocumentKey=001f5e09-88b4-4a9a-b3104c20578eecf9&CommunityKey=1ecf5f55-9545-44d6b0f4-4e4a7f5f5e68&tab=librarydocuments. alındığı tarih: 23 02 2021.

[29] Url 9: https://www.snort.org/talos. alındığı tarih: 23 02 2021.

[30] Url 10: https://doc.emergingthreats.net/bin/view/Main/AboutEmergingThreats. alındığı tarih: 31 07 2021.

[31] Url 11: https://documentation.wazuh.com/current/virtual-machine/virtual-machine.html#virtual-machine. alındığı tarih: 10 10 2021.

[32] Url 12: https://docs.rapid7.com/metasploit/msf-overview/. alındığı tarih: 31 07 2021].

[33] **F. Holik, J. Horalek, O. Marik, S. Neradova ve S. Zitta**, (2014). Effective penetration testing with Metasploit framework and methodologies, CINTI 2014 - 15th IEEE International Symposium on Computational Intelligence and Informatics, pp. 237-242.

[34] Url 13: https://www.wireshark.org/. alındığı tarih: 31 07 2021.

[35] **S. Wang, D. Xu ve S. Yan**, (2010). Analysis and Application of Wireshark in TCP/IP Protocol Teaching,» 2010 International Conference on E-Health Networking, Digital Ecosystems and Technologies, pp. 269-272.

[36] Url 14: https://dvwa.co.uk/. alındığı tarih: 31 07 2021.

[37] Url 15: http://manual-snort-org.s3-website-us-east-1.amazonaws.com/node8.html. alındığı tarih: 31 07 2021.

[38] Url 16: https://suricata.readthedocs.io/en/suricata-6.0.0/manpages/suricata.html. alındığı tarih: 31 07 2021.

[39] **T. M. Mahmoud, A. A. Ali ve H. M. Elshafie**, (2016). A Hybrid Snort-Negative Selection Network Intrusion Detection Technique, International Journal of Computer Applications, cilt 146, no. 5, pp. 24-31.

[40] Url 17: https://www.malware-traffic-analysis.net/. alındığı tarih: 31 07 2021.

[41] Url 18: https://www.exploit-db.com/. alındığı tarih: 23 02 2021.

[42] Url 19: https://www.rapid7.com/. alındığı tarih: 10 10 2021.

[43] Url 20: https://www.unb.ca/cic/datasets/dohbrw-2020.html. alındığı tarih: 31 07 2021.

[44] **M. MontazeriShatoori, L. Davidson, G. Kaur ve A. H. Lashkari**, (2020). Detection of DoH Tunnels using Time-series Classification of Encrypted Traffic, The 5th IEEE Cyber Science and Technology Congress, Calgary, Canada.

[45] **I. Sharafaldin, A. H. Lashkari ve A. A. Ghorbani**, (2018). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization, 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal.

[46] Url 21: https://www.unb.ca/cic/datasets/ids-2017.html. alındığı tarih: 31 07 2021.

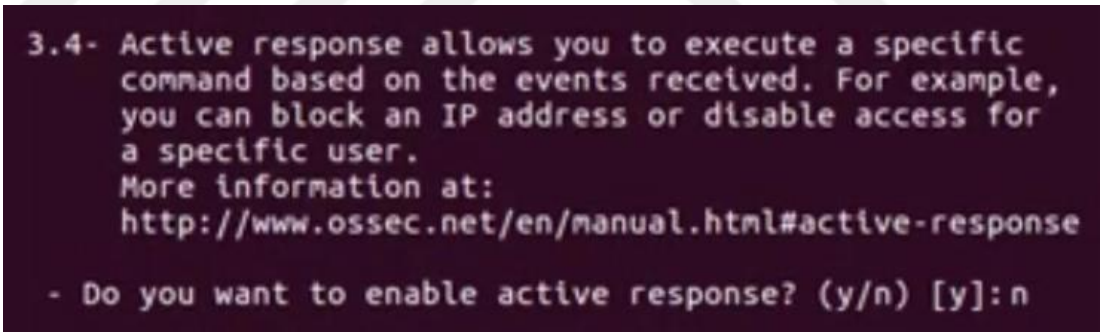
EKLER

Ek 1 : Snort3-Suricata Konfigürasyon Betiği

```
import os
folder="rules"

for root, dirs, files in os.walk(folder):
    for filename in files:


        with open(folder+"/"+filename, 'r') as file :
           filedata = file.read()

        # Replace the target string
        filedata = filedata.replace('# alert', 'alert')

        # Write the file out again
        with open(folder+"/"+filename, 'w') as file:
           file.write(filedata)
```

Şekil Ek.1 : Snort3-Suricata Konfigürasyon Betiği.

Ek 2 : OSSEC Konfigürasyonu



Şekil Ek.2 : OSSEC Konfigürasyonu.

Ek 3 : Wazuh Konfigürasyonu

```
<!-- Active response -->
<active-response>
  <disabled>yes</disabled>
</active-response>
```

Şekil Ek.3 : Wazuh Konfigürasyonu.

Ek 4 : Snort Otomasyon Betiği

```
import subprocess
import shutil
import os
import csv
folder ="Pcap Folder"


for root, dirs, files in os.walk(folder):
   for filename in files:
      with open('alert.txt', 'w') as fp:
            pass
      alert_file = open("alert.txt",'a')
      subprocess.call(["snort",                    "-q",                    "-c",
"/home/snort3/Desktop/Snort3_Rules/Community_Rules/snort/snort.lua",          "-r",
os.path.abspath(folder + "/"+filename),"-A", "full"],stdout=alert_file)
      alert_file = open("alert.txt",'r')
      output = alert_file.read()

      output_file = open("Snort_Alerts_Community_"+folder+".txt", "a")
      output_file.write("+++++++++++++++"+filename+"+++++++++++++++\n")
      for line in output.splitlines():
        if "[**]" in line:
            output_file.writelines(line + "\n")
            with open('Snort_Results_Community_'+folder+'.csv', 'a') as file:
                writer = csv.writer(file)
                writer.writerow([filename,"1"])
      if (os.stat("alert.txt").st_size == 0):
        with open('Snort_Results_Community_'+folder+'.csv', 'a') as file:
            writer = csv.writer(file)
            writer.writerow([filename,"0"])
      output_file.write("-----------------------------------------------------------------
\n\n\n\n")

      os.remove("alert.txt")


for root, dirs, files in os.walk(folder):
   for filename in files:
      with open('alert.txt', 'w') as fp:
            pass
      alert_file = open("alert.txt",'a')
      subprocess.call(["snort",                    "-q",                    "-c",
"/home/snort3/Desktop/Snort3_Rules/Web_Rules/snort/snort.lua",                "-r",
os.path.abspath(folder + "/"+filename),"-A", "full"],stdout=alert_file)
      alert_file = open("alert.txt",'r')
      output = alert_file.read()

      output_file = open("Snort_Alerts_Web_"+folder+".txt", "a")
      output_file.write("+++++++++++++++"+filename+"+++++++++++++++\n")
      for line in output.splitlines():
```

```
        if "[**]" in line:
            output_file.writelines(line + "\n")
            with open('Snort_Results_Web_'+folder+'.csv', 'a') as file:
                writer = csv.writer(file)
                writer.writerow([filename,"1"])
      if (os.stat("alert.txt").st_size == 0):
        with open('Snort_Results_Web_'+folder+'.csv', 'a') as file:
            writer = csv.writer(file)
            writer.writerow([filename,"0"])
      output_file.write("-------------------------------------------------------------------
\n\n\n\n")

      os.remove("alert.txt")

for root, dirs, files in os.walk(folder):
   for filename in files:
      with open('alert.txt', 'w') as fp:
            pass
      alert_file = open("alert.txt",'a')
      subprocess.call(["snort",                    "-q",                    "-c",
"/home/snort3/Desktop/Snort3_Rules/Registered      Rules/snort/snort.lua",      "-r",
os.path.abspath(folder + "/"+filename),"-A", "full"],stdout=alert_file)
      alert_file = open("alert.txt",'r')
      output = alert_file.read()

      output_file = open("Snort_Alerts_Registered_"+folder+".txt", "a")
      output_file.write("+++++++++++++++"+filename+"+++++++++++++++\n")
      for line in output.splitlines():
        if "[**]" in line:
            output_file.writelines(line + "\n")
            with open('Snort_Results_Registered_'+folder+'.csv', 'a') as file:
                writer = csv.writer(file)
                writer.writerow([filename,"1"])
      if (os.stat("alert.txt").st_size == 0):
        with open('Snort_Results_Registered_'+folder+'.csv', 'a') as file:
            writer = csv.writer(file)
            writer.writerow([filename,"0"])
      output_file.write("-------------------------------------------------------------------
\n\n\n\n")

      os.remove("alert.txt")


for root, dirs, files in os.walk(folder):
   for filename in files:
      with open('alert.txt', 'w') as fp:
            pass
      alert_file = open("alert.txt",'a')
```

```
        subprocess.call(["snort",                    "-q",                    "-c",
"/home/snort3/Desktop/Snort3_Rules/lightspd/snort/snort.lua",                    "-r",
os.path.abspath(folder + "/"+filename),"-A", "full"],stdout=alert_file)
        alert_file = open("alert.txt",'r')
        output = alert_file.read()

        output_file = open("Snort_Alerts_lightspd_"+folder+".txt", "a")
        output_file.write("++++++++++++++++"+filename+"++++++++++++++++\n")
        for line in output.splitlines():
          if "[**]" in line:
              output_file.writelines(line + "\n")
              with open('Snort_Results_lightspd_'+folder+'.csv', 'a') as file:
                    writer = csv.writer(file)
                    writer.writerow([filename,"1"])
        if (os.stat("alert.txt").st_size == 0):
          with open('Snort_Results_lightspd_'+folder+'.csv', 'a') as file:
              writer = csv.writer(file)
              writer.writerow([filename,"0"])
        output_file.write("------------------------------------------------------------
\n\n\n\n")

        os.remove("alert.txt")
```

Ek 5 : Suricata Otomasyon Betiği

```
import subprocess
import shutil
import os
import csv
folder ="Pcap Folder"


for root, dirs, files in os.walk(folder):
    for filename in files:
        with open('fast.log', 'w') as fp:
              pass
        subprocess.call(["suricata",                                                    "-c",
"/home/snort3/Desktop/Suricata_Rules/Community_Rules/suricata.yaml",          "-r",
os.path.abspath(folder + "/"+filename), "-v"])
        alert_file = open("fast.log",'r')
        output = alert_file.read()

        output_file = open("Suricata_Alerts_Community_"+folder+".txt", "a")
        output_file.write("++++++++++++++++"+filename+"++++++++++++++++\n")
        for line in output.splitlines():
          if "[**]" in line:
              output_file.writelines(line + "\n")
              with open('Suricata_Results_Community_'+folder+'.csv', 'a') as file:
```

```
                writer = csv.writer(file)
                writer.writerow([filename,"1"])
        if (os.stat("fast.log").st_size == 0):
          with open('Suricata_Results_Community_'+folder+'.csv', 'a') as file:
                writer = csv.writer(file)
                writer.writerow([filename,"0"])
        output_file.write("-----------------------------------------------------------------
\n\n\n\n")

        os.remove("fast.log")


for root, dirs, files in os.walk(folder):
    for filename in files:
        with open('fast.log', 'w') as fp:
                pass
        subprocess.call(["suricata",                                                    "-c",
"/home/snort3/Desktop/Suricata_Rules/Web_Rules/suricata.yaml",                  "-r",
os.path.abspath(folder + "/"+filename), "-v"])
        alert_file = open("fast.log",'r')
        output = alert_file.read()

        output_file = open("Suricata_Alerts_Web_"+folder+".txt", "a")
        output_file.write("++++++++++++++++"+filename+"++++++++++++++++\n")
        for line in output.splitlines():
          if "[**]" in line:
                output_file.writelines(line + "\n")
                with open('Suricata_Results_Web_'+folder+'.csv', 'a') as file:
                        writer = csv.writer(file)
                        writer.writerow([filename,"1"])
        if (os.stat("fast.log").st_size == 0):
          with open('Suricata_Results_Web_'+folder+'.csv', 'a') as file:
                writer = csv.writer(file)
                writer.writerow([filename,"0"])
        output_file.write("-----------------------------------------------------------------
\n\n\n\n")

        os.remove("fast.log")

for root, dirs, files in os.walk(folder):
    for filename in files:
        with open('fast.log', 'w') as fp:
                pass
        subprocess.call(["suricata",                                                    "-c",
"/home/snort3/Desktop/Suricata_Rules/Registered       Rules/suricata.yaml",      "-r",
os.path.abspath(folder + "/"+filename), "-v"])
        alert_file = open("fast.log",'r')
        output = alert_file.read()

        output_file = open("Suricata_Alerts_Registered_"+folder+".txt", "a")
```

```
        output_file.write("+++++++++++++++"+filename+"+++++++++++++++\n")
        for line in output.splitlines():
          if "[**]" in line:
                output_file.writelines(line + "\n")
                with open('Suricata_Results_Registered_'+folder+'.csv', 'a') as file:
                      writer = csv.writer(file)
                      writer.writerow([filename,"1"])
        if (os.stat("fast.log").st_size == 0):
          with open('Suricata_Results_Registered_'+folder+'.csv', 'a') as file:
                writer = csv.writer(file)
                writer.writerow([filename,"0"])
        output_file.write("------------------------------------------------------------------
\n\n\n\n")


        os.remove("fast.log")
```

Ek 6 : Broadcom Kuralları

alert tcp any any -> any $HTTP_PORTS (msg:"SQL Injection - Paranoid";flow:to_server,established;pcre:"/(\%27)|(\')|(\-\-)|(%23)|(#)/i"; classtype:web-application-attack; sid:909900;rev:5;)

alert tcp any any -> any $HTTP_PORTS (msg:"Modified regex for detection of SQL meta-characters";flow:to_server,established;pcre:"/((\%3D)|(=))[^\n]*((\%27)|(\')|(\-\-)|(\%3B)|(\;))/i"; classtype:web-application-attack; sid:910000;rev:5;)

alert tcp any any -> any $HTTP_PORTS (msg:"Regex for typical SQL Injection attack";flow:to_server,established;pcre:"/\w*((\%27)|(\'))((\%6F)|o|(\%4F))((\%72)|r|(\%52))/ix"; classtype:web-application-attack; sid:910001;rev:5;)

alert tcp any any -> any $HTTP_PORTS (msg:"Regex for detecting SQL Injection with the UNION keyword";flow:to_server,established;pcre:"/((\%27)|(\'))union/ix"; classtype:web-application-attack; sid:910002;rev:5;)

alert tcp any any -> any $HTTP_PORTS (msg:"Regex for detecting SQL Injection attacks on a MS SQL Server";flow:to_server,established;pcre:"/exec(\s|\+)+(s|x)p\w+/ix"; classtype:web-application-attack; sid:910003;rev:5;)

alert tcp any any -> any $HTTP_PORTS (msg:"Regex for simple CSS attack";flow:to_server,established;pcre:"/((\%3C)|<)((\%2F)|\/)*[a-z0-9\%]+((\%3E)|>)/ix"; classtype:web-application-attack; sid:910004;rev:5;)

Ek 7 : Testlerde Kullanılan Zararlı Trafikler

Atak trafikleri “https://github.com/compleng/Pcaps-For-IDS-Analysis” web adresinde yer almaktadır.

Çizelge Ek.7 : Zararlı Trafiklerin Listesi.

CVE Referanslı Zafiyet Sömürü Atakları	
ID	**Dosya Adı**
1	2017-07-05-Japanese-malspam-traffic.pcap
2	2018-06-04-ransomware-malspam-infection-traffic.pcap
3	2018-09-06-infection-traffic-from-password-protected-Word-doc.pcap
4	2018-10-26-GlobeImposter-ransomware-from-password-protected-Word-doc-malspam.pcap
5	2018-12-10-password-protected-Word-doc-pushes-Nymaim.pcap
6	2018-12-17-IcedID-from-password-protected-Word-doc.pcap
7	2019-09-05-2nd-run-Word-doc-causes-Vidar-infection.pcap
8	2019-11-06-Ursnif-infection-with-Dridex-and-proxy-activity.pcap
9	2020-03-12-infection-traffic.pcap
10	2020-04-21-Fastloader-pushes-Trickbot-and-AnyDesk-traffic.pcap
11	2020-05-01-XLS-to-Loader-to-IcedID-infection-traffic.pcap
12	2020-05-27-IcedID-infection-from-COVID19-themed-Word-doc.pcap
13	2020-05-27-traffic-from-Valak-infection-with-IcedID.pcap
14	2020-07-07-Ursnif-infection-with-IcedID.pcap
15	2020-07-20-IcedID-infection-traffic.pcap
16	2020-10-12-Lokibot-infection-traffic.pcap
17	2020-12-03-traffic-analysis-quiz.pcap
18	2021-04-16-TA551-Ursnif-traffic.pcap
19	2021-05-24-TA551-IcedID-infection-traffic.pcap
20	2021-06-18-TA551-Gozi-ISFB-Ursnif-infection-traffic.pcap
21	9-8-3-PHP_memory_limit_vulnerability_exploit_attempt_CVE-2007-1868_IBM_POST_Request.pcap
22	9-8-5-PHP_memory_limit_vulnerability_exploit_attempt_(Attempted_User_Privilege_Gain)_-_CVE-2007-1868_IBM_POST_Request.pcap
23	9-8-6-cross-site_scripting_attempt_via_form_data_attempt_(Attempted_User_Privilege_Gain)_-_CVE-2007-3383_tomcat_sendmail_attack.pcap
24	accellion_fta_getstatus_oauth.pcap
25	adobe_flash_delete_range_tl_op.pcap
26	adobe_flash_hacking_team_uaf.pcap
27	adobe_flash_nellymoser_bof.pcap
28	airties_login_cgi_bof.pcap
29	alienvault_centerd_soap_exec.pcap
30	apache_activemq_upload_jsp.pcap
31	apache_james_exec_1.pcap
32	apache_james_exec_2.pcap
33	apache_jetspeed_file_upload.pcap

34	apache_roller_ognl_injection.pcap
35	arkeia_agent_exec.pcap
36	asus_infosvr_auth_bypass_exec.pcap
37	asuswrt_lan_rce.pcap
38	bludit_upload_images_exec.pcap
39	bmc_patrol_cmd_exec.pcap
40	C-CVE-2014-9390.pcap
41	C-CVE-2015-0072.pcap
42	C-CVE-2015-0072.pcap.pcapng
43	chrome_array_map.pcap
44	cisco_dcnm_upload.pcap
45	cisco_dcnm_upload_2019%0A.pcap
46	cisco_dcnm_upload_2019.pcap
47	cisco_rv32x_rce%0A.pcap
48	clamav_milter_blackhole%0A.pcap
49	claymore_dual_miner_remote_manager_rce.pcap
50	confluence_widget_connector.pcap
51	cve_2019_0708_bluekeep_rce.pcap
52	cve_2019_1663_cisco_rmi_rce.pcap
53	CVE-2005-3934_Symantec_PcAnywhere_Denial_of_Service1.pcap
54	CVE-2005-3934_Symantec_PcAnywhere_Denial_of_Service2.pcap
55	CVE-2006-4110_Apache_for_Windows_Source_Code_Disclosure.pcap
56	CVE-2007-0515_Word_Document_Malformed_Drawing_Object_Code_Execution-_Vulnerability.pcap
57	CVE-2007-1256_Mozilla_Firefox_onUnload_Memory_Corruption.pcap
58	CVE-2007-1868_IBM_POST_Request.pcap
59	CVE-2007-3383_tomcat_sendmail_attack.pcap
60	CVE-2007-5020_Acrobat_mailto_URI_Handler.pcap
61	CVE-2007-5405_ibm_lotusnotes_agfile_encoding_bo.pcap
62	CVE-2007-5406_ibm_lotusnotes_agfile_bo.pcap
63	CVE-2007-5646_Simple_Machines_Forum_index.php_SQL_Injection.pcap
64	CVE-2008-0352_ipv6_command-as-per-exploit.pcap
65	CVE-2008-0420_Firefox_BMP_leak_attack.pcap
66	CVE-2008-0514_Joomla_Mambo_com_glossary_SQLInjection.pcap
67	CVE-2008-1275_Mailenable_EXPN_onserverside.pcap
68	CVE-2008-1275_Mailenable_VRFY.pcap
69	CVE-2008-1933_MS_Zune_ActiveX.pcap
70	CVE-2008-1965_IBM_Expeditor_cai_URI_Handlerargument_injection.pcap
71	CVE-2012-0158_malware_CnC_beacon.pcap
72	CVE-2014-1854-xx.pcap
73	CVE-2014-8440.pcap
74	CVE-2015-0235_exim_all.pcap
75	CVE-2015-0311_variant_6_swf.pcap
76	CVE-2015-0311_variant_7_swf.pcap
77	CVE-2015-0311_variant_8_swf.pcap
78	CVE-2015-0313-prestep2-post-request.pcap
79	CVE-2015-0313-prestep2-request-flash-download.pcap

80	CVE-2015-0313-step1.pcap
81	CVE-2015-0313-step1-request-malicious-page.pcap
82	CVE-2015-0359.pcap
83	CVE-2015-0359-part1.pcap
84	CVE-2015-0359-part2.pcap
85	CVE-2015-0359-part3.pcap
86	dlink_dcs931l_upload.pcap
87	dlink_dwl_2600_command_injection.pcap
88	dlink_hnap_bof.pcap
89	drupal_restws_unserialize%0A.pcap
90	drupal_restws_unserialize.pcap
91	elfinder_php_connector_exiftran_cmd_injection.pcap
92	evocam_webserver%0A.pcap
93	evocam_webserver.pcap
94	exagrid_known_privkey%0A.pcap
95	eyesofnetwork_autodiscovery_rce.pcap
96	f5_bigip_known_privkey.pcap
97	f5_icall_cmd.pcap
98	ftpshell_cli_bof%0A.pcap
99	fusionpbx_operator_panel_exec_cmd_exec%0A.pcap
100	getsimplecms_unauth_code_exec.pcap
101	hp_vsa_exec.pcap
102	ibm_openadmin_tool_soap_welcomeserver_exec%0A.pcap
103	ibm_openadmin_tool_soap_welcomeserver_exec.pcap
104	ibm_tm1_unauth_rce.pcap
105	iis_webdav_scstoragepathfromurl%0A.pcap
106	iokit_keyboard_root.pcap
107	is_known_pipename%0A.pcap
108	itms_overflow.pcap
109	jenkins_java_deserialize.pcap
110	jenkins_ldap_deserialize%0A.pcap
111	jenkins_metaprogramming%0A.pcap
112	jenkins_xstream_deserialize.pcap
113	jira_hipchat_template.pcap
114	jquery_file_upload.pcap
115	kaltura_unserialize_cookie_rce.pcap
116	laravel_token_unserialize_exec%0A.pcap
117	lexmark_markvision_gfd_upload.pcap
118	linksys_wrt110_cmd_exec.pcap
119	linksys_wvbr0_user_agent_exec_noauth.pcap
120	loginext.pcap
121	manage_engine_opmanager_rce.pcap
122	manageengine_auth_upload.pcap
123	mantisbt_manage_proj_page_rce.pcap
124	moodle_cmd_exec%0A.pcap
125	mozilla_mchannel.pcap

126	ms17_010_psexec%0A.pcap
127	navigate_cms_rce.pcap
128	netgear_readynas_exec.pcap
129	netgear_unauth_exec.pcap
130	nexus_repo_manager_el_injection.pcap
131	nginx_chunked_size%0A.pcap
132	nostromo_code_exec.pcap
133	nuuo_nvrmini_upgrade_rce.pcap
134	october_upload_bypass_exec.pcap
135	openmrs_deserialization.pcap
136	opensmtpd_mail_from_rce.pcap
137	oracle_ats_file_upload.pcap
138	oracle_btm_writetofile.pcap
139	oracle_weblogic_wsat_deserialization_rce.pcap
140	php_fpm_rce%0A.pcap
141	php_fpm_rce.pcap
142	php_imap_open_rce%0A.pcap
143	phpcollab_upload_exec.pcap
144	phpmyadmin_lfi_rce%0A.pcap
145	phpmyadmin_lfi_rce.pcap
146	pimcore_unserialize_rce.pcap
147	playsms_template_injection.pcap
148	plesk_mylittleadmin_viewstate%0A.pcap
149	pulse_secure_cmd_exec.pcap
150	pureftpd_bash_env_exec.pcap
151	qnap_transcode_server.pcap
152	quicktime_rtsp_content_type.pcap
153	rails_dynamic_render_code_exec.pcap
154	rootpipe.pcap
155	rootpipe_entitlements.pcap
156	safari_file_policy.pcap
157	safari_proxy_object_type_confusion.pcap
158	safari_user_assisted_applescript_exec.pcap
159	script_mvel_rce.pcap
160	search_groovy_script.pcap
161	sercomm_exec.pcap
162	setuid_tunnelblick.pcap
163	setuid_viscosity.pcap
164	shiro_rememberme_v124_deserialize%0A.pcap
165	shopware_createinstancefromnamedarguments_rce.pcap
166	smb_webexec%0A.pcap
167	smt_ipmi_close_window_bof.pcap
168	software_update.pcap
169	solarwinds_store_manager_auth_filter.pcap
170	solr_velocity_rce%0A.pcap
171	sonicwall_scrutinizer_methoddetail_sqli.pcap

172	stagefright_mp4_tx3g_64bit.pcap
173	struts_dev_mode.pcap
174	struts_dmi_exec.pcap
175	sudo_password_bypass.pcap
176	synology_dsm_smart_exec_auth.pcap
177	sysaid_rdslogs_file_upload.pcap
178	sysaid_rdslogs_file_upload.pcap
179	thinkphp_rce.pcap
180	tomcat_cgi_cmdlineargs%0A.pcap
181	tomcat_jsp_upload_bypass%0A.pcap
182	tomcat_jsp_upload_bypass.pcap
183	tplink_archer_a7_c7_lan_rce.pcap
184	trans2open%0A.pcap
185	trans2open_multiple.pcap
186	tuleap_rest_unserialize_exec.pcap
187	ufo_ai.pcap
188	unraid_auth_bypass_exec.pcap
189	upnp_location.pcap
190	vbulletin_widget_template_rce%0A.pcap
191	visual_mining_netcharts_upload.pcap
192	wd_mycloud_multiupload_upload.pcap
193	weblogic_deserialize_asyncresponseservice.pcap
194	webstar_ftp_user.pcap
195	webview_addjavascriptinterface.pcap
196	wepresent_cmd_injection%0A.pcap
197	wp_crop_rce.pcap
198	xymon_useradm_cmd_exec.pcap
199	zenworks_assetmgmt_uploadservlet.pcap
200	zenworks_configuration_management_upload.pcap
Web Uygulama Atakları	
ID	**Dosya Adı**
201	AND-boolean-based-blind---WHERE-or-HAVING-clause-(MySQL-comment).pcap
202	I-3-9-1-Havij_advanced_SQL_injection_tool_user-agent_string.pcap
203	I-8-4-1-sqlmap_SQL_injection_scan_attempt.pcap
204	I-9-8-4-SQL_union_select_sql_injection_attempt_GET_parameter.pcap
205	Microsoft-SQL-Server---Sybase-boolean-based-blind---Parameter-replace-(original-value).pcap
206	MySQL-_=-5.0-boolean-based-blind---Parameter-replace-(original-value).pcap
207	MySQL-_-5.0-boolean-based-blind---Parameter-replace-(original-value).pcap
208	MySQL_Authentication_Bypass_with_ZeroLength_String2.pcap
209	MySQL-boolean-based-blind---Parameter-replace-(bool_int---original-value).pcap
210	MySQL-boolean-based-blind---Parameter-replace-(ELT---original-value).pcap
211	MySQL-boolean-based-blind---Parameter-replace-(MAKE_SET---original-value).pcap
212	MySQL-boolean-based-blind---WHERE,-HAVING,-ORDER-BY-or-GROUP-BY-clause-(RLIKE).pcap
213	S-3-9-1-Havij_advanced_SQL_injection_tool_user-agent_string.pcap
214	S-8-3-1-oversized_cast_statement_possible_sql_injection_obfuscation.pcap

215	S-8-4-1-sqlmap_SQL_injection_scan_attempt.pcap
216	S-9-8-4-SQL_union_select_sql_injection_attempt_GET_parameter.pcap
217	sqli2.pcap
218	sqli6.pcap
219	SW-apostrophemask_SQLi.pcap
220	SW-apostrophenullencode_SQLi.pcap
221	SW-appendnullbyte_SQLi.pcap
222	SW-base64encode_SQLi.pcap
223	SW-between_SQLi.pcap
224	SW-bluecoat_SQLi.pcap
225	SW-chardoubleencode_SQLi.pcap
226	SW-charencode_SQLi.pcap
227	SW-charunicodeencode_SQLi.pcap
228	SW-concat2concatws_SQLi.pcap
229	SW-equaltolike_SQLi.pcap
230	SW-greatest_SQLi.pcap
231	SW-halfversionedmorekeywords_SQLi.pcap
232	SW-ifnull2ifisnull_SQLi.pcap
233	SW-informationschemacomment_SQLi.pcap
234	SW-lowercase_SQLi.pcap
235	SW-Microsoft_SQL_Server_Sybase_inline_queries.pcap
236	SW-Microsoft_SQL_Server_Sybase_stacked_queries.pcap
237	SW-Microsoft_SQL_Server_Sybase_timebased_blind.pcap
238	SW-Microsoft_SQL_ServerSybase_AND_errorbased_WHERE_or_HAVING_clause.pcap
239	SW-modsecurityversioned_SQLi.pcap
240	SW-modsecurityzeroversioned_SQLi.pcap
241	SW-multiplespaces_SQLi.pcap
242	SW-MySQL_5.0.11_AND_timebased_blind.pcap
243	SW-MySQL_5.0.11_stacked_queries.pcap
244	SW-MySQL_5.0_AND_errorbased_WHERE_or_HAVING_clause.pcap
245	SW-MySQL_inline_queries.pcap
246	SW-nonrecursivereplacement_SQLi.pcap
247	SW-overlongutf8_SQLi.pcap
248	SW-percentage_SQLi.pcap
249	SW-PostgreSQL_8.1_AND_timebased_blind.pcap
250	SW-PostgreSQL_8.1_stacked_queries.pcap
251	SW-PostgreSQL_AND_errorbased_WHERE_or_HAVING_clause.pcap
252	SW-PostgreSQL_inline_queries.pcap
253	SW-randomcase_SQLi.pcap
254	SW-randomcomments_SQLi.pcap
255	SW-referer_header_based_AND_boolean_based_blind_SQLi.pcap
256	SW-referer_header_based_AND_error_based_SQLi.pcap
257	SW-referer_header_based_generic_union_query_SQLi.pcap
258	SW-referer_header_based_MySQL_inline_queries_SQLi.pcap
259	SW-referer_header_based_MySQL_union_query_SQLi.pcap
260	SW-referer_header_based_time_based_blind_SQLi.pcap

261	SW-securesphere_SQLi.pcap
262	SW-sp_password_SQLi.pcap
263	SW-space2comment_SQLi.pcap
264	SW-space2dash_SQLi.pcap
265	SW-space2hash_SQLi.pcap
266	SW-space2morehash_SQLi.pcap
267	SW-space2mssqlblank_SQLi.pcap
268	SW-space2mssqlhash_SQLi.pcap
269	SW-space2mysqlblank_SQLi.pcap
270	SW-space2mysqldash_SQLi.pcap
271	SW-space2plus_SQLi.pcap
272	SW-space2randomblank_SQLi.pcap
273	SW-SQLite_inline_queries.pcap
274	SW-unionalltounion_SQLi.pcap
275	SW-unmagicquotes_SQLi.pcap
276	SW-varnish_SQLi.pcap
277	SW-versionedkeywords_SQLi.pcap
278	SW-versionedmorekeywords_SQLi.pcap
279	SW-xforwardedfor_SQLi.pcap
280	9-1-1-Apache_XSS_using_HTTP_Request_invalid_Content_Length.pcap
281	Apache_XSS_using_HTTP_Request_invalid_Content_Length.pcap
282	mambo_mambo_xss.pcap
283	MSExchange_Server_Outlook_Web_Access_XSS.pcap
284	SW-Local_file_inclusion_using_XSS.pcap
285	SW-XSS_basic_in_javascript_tag.pcap
286	SW-XSS_basic_URL_encoded.pcap
287	SW-XSS_in_different_tags.pcap
288	SW-XSS_in_form_element.pcap
289	SW-XSS_in_iframe.pcap
290	SW-XSS_in_iframesrc.pcap
291	SW-XSS_in_img_src_onerror_property.pcap
292	SW-XSS_in_input_element.pcap
293	SW-XSS_in_marquee_onstart.pcap
294	SW-XSS_in_object_element.pcap
295	SW-XSS_in_onhashchange_property.pcap
296	SW-XSS_in_onmouseover_property.pcap
297	SW-XSS_in_prompt_command.pcap
298	SW-XSS_in_splitted_script_tag.pcap
299	SW-XSS_in_svg_tag.pcap
300	SW-XSS_in_video_src_onerror_property.pcap
301	SW-XSS_with_mixed_letters_.pcap
302	SW-XSS_with_null_characters.pcap
303	SW-XSS_with_oncut.pcap
304	SW-XSS_with_svg_onload.pcap
305	SW-bw_http_response_splitting_1.pcap
306	S-bw_insecure_VNC_configuration_and_weak_password.pcap

307	SW-bw_server_side_request_forgery_SSRF_port_scan.pcap
308	S-bw_php_cgi_remote_code_execution_CVE-2012-1823_defacement.pcap
309	SW-bw_command_execution_through_RFI.pcap
310	SW-Remote_file_inclusion_via_FTP.pcap
311	SW-bw_acquire_shell_access_through_LFI_stage1.pcap
312	SW-bw_acquire_shell_access_through_LFI_stage2.pcap
313	S-bw_phpmyadmin_errorphp_bbcode_tag_xss_cve_2010_4480.pcap
314	SW-bw_unvalidated_redirects_and_forwards_1.pcap
315	SW-bw_remote_file_inclusion_rfi_sensitive_file_disclosure.pcap
316	SW-bw_server_side_includes_SSI_injection_defacement_with_echo.pcap
317	SW-bw_php_code_injection.pcap
318	SW-bw_reflected_html_injection_with_get.pcap
319	SW-Remote_file_inclusion_with_question_character.pcap
320	SW-bw_local_file_inclusion_lfi_sensitive_file_disclosure.pcap
321	SW-Command_injection_using_error_logs.pcap
322	SW-Directory_listing_with_null_byte_injection.pcap
323	SW-Local_file_inclusion_using_data_URIs.pcap
324	SW-bw_HTML5_web_storage_sensitive_data_exposure.pcap
325	SW-Local_file_inclusion_using_PHP_stream.pcap
326	SW-Local_file_inclusion_with_dot_truncation.pcap
327	SW-bw_http_response_splitting_2.pcap
328	S-bw_php_source_code_disclosure.pcap
329	SW-Local_file_inclusion_using_XSS.pcap
330	SW-Remote_file_inclusion_with_hash_character.pcap
331	SW-bw_server_side_includes_SSI_injection_ls_command.pcap
332	SW-Local_file_inclusion_with_path_truncation.pcap
333	SW-bw_unvalidated_redirects_and_forwards_url_encoded.pcap
334	SW-bw_server_side_includes_SSI_injection_cat_etc_passwd_command.pcap
335	SW-bw_stored_html_injection_with_post.pcap
336	SW-Local_file_inclusion_with_null_byte_injection.pcap
337	Command_Inj.pcap
338	File_Upload.pcap
339	Payload1.pcapng
340	Payload10.pcapng
341	Payload11.pcapng
342	Payload12.pcapng
343	Payload13.pcapng
344	Payload14.pcapng
345	Payload15.pcapng
346	Payload16.pcapng
347	Payload17.pcapng
348	Payload18.pcapng
349	Payload19.pcapng
350	Payload2.pcapng
351	Payload20.pcapng
352	Payload3.pcapng

353	Payload4.pcapng
354	Payload5.pcapng
355	Payload6.pcapng
356	Payload7.pcapng
357	Payload8.pcapng
358	Payload9.pcapng
359	PayloadExtra.pcapng
360	Sqli_low.pcap
361	sqlmap_dvwa.pcap
362	wp_google_document_embedder_exec.pcap
363	op5_license.pcap
364	wikka_spam_exec.pcap
365	umbraco_upload_aspx.pcap
366	log1cms_ajax_create_folder.pcap
367	basilic_diff_exec.pcap
368	vbulletin_unserialize.pcap
369	php_volunteer_upload_exec.pcap
370	projectpier_upload_exec.pcap
371	wp_ninja_forms_unauthenticated_file_upload.pcap
372	sugarcrm_unserialize_exec.pcap
373	mobilecartly_upload_exec.pcap
374	qdpm_upload_exec.pcap
375	wp_infinitewp_auth_bypass.pcap
376	phpmyadmin_3522_backdoor.pcap
377	webcalendar_settings_exec.pcap
378	symantec_web_gateway_exec.pcap
379	glassfish_traversal.pcap
380	hastymail_exec.pcap
381	escan_password_exec.pcap
382	novell_mdm_lfi.pcap
383	mantisbt_php_exec.pcap
384	sysaid_auth_file_upload.pcap
385	phptax_exec.pcap
386	egallery_upload_exec.pcap
387	phpmailer_arg_injection.pcap
388	clipbucket_fileupload_exec.pcap
389	openfire_auth_bypass.pcap
390	wp_easycart_unrestricted_file_upload.pcap
391	joomla_comfields_sqli_rce.pcap
392	zimbra_lfi.pcap
393	webid_converter.pcap
394	apprain_upload_exec.pcap
395	webpagetest_upload_exec.pcap
396	xoda_file_upload.pcap
397	wp_foxypress_upload.pcap
398	auxilium_upload_exec.pcap

399	symantec_web_gateway_file_upload.pcap
400	cuteflow_upload_exec.pcap
Zararlı Yazılım Trafikleri	
ID	**Dosya Adı**
401	2013-06-18-Neutrino-EK traffic.pcap
402	2013-07-08-DotkaChef-EK-traffic.pcap
403	2013-07-14-DotkaChef-EK-traffic.pcap
404	2013-07-21-Blackhole-EK-traffic.pcap
405	2013-07-28-phishing-malware-traffic.pcap
406	2013-07-31-Cool-EK-traffic.pcap
407	2013-08-05-Styx-EK-traffic.pcap
408	2013-08-10-Blackhole-EK-traffic.pcap
409	2013-08-16-Styx-EK-traffic.pcap
410	2013-08-23-g01pack-EK-traffic.pcap
411	2013-09-07-Sweet-Orange-EK-traffic.pcap
412	2013-09-28-Blackhole-EK-traffic.pcap
413	2013-10-28-Sibhost-EK-traffic.pcap
414	2013-11-15-Gondad-EK-traffic.pcap
415	2013-11-23-Styx-EK-traffic.pcap
416	2013-12-09-Whitehole-EK-traffic.pcap
417	2013-12-19-Neutrino-EK-traffic.pcap
418	2013-12-23-Neutrino-EK-traffic.pcap
419	2013-12-27-Styx-EK-traffic.pcap
420	2014-01-09-DotkaChef-EK-traffic.pcap
421	2014-01-21-Neutrino-EK-traffic.pcap
422	2014-02-04-Sweet-Orange-EK-traffic.pcap
423	2014-02-22-Neutrino-and-Nuclear-EK-traffic.pcap
424	2014-03-04-Hello-EK-traffic.pcap
425	2014-03-17-Zuponcic-EK-traffic.pcap
426	2014-04-11-Fiesta-EK-traffic.pcap
427	2014-04-23-Goon-EK-traffic.pcap
428	2014-05-11-fake-Flash-updater-analysis-from-malwr.pcap
429	2014-05-21-Sweet-Orange-EK-traffic.pcap
430	2014-06-06-FlashPack-EK-traffic.pcap
431	2014-06-30-Infinity-EK-traffic.pcap
432	2014-07-09-Zuponcic-EK-traffic-java-6u25.pcap
433	2014-07-26-Rig-EK-traffic.pcap
434	2014-08-01-phishing-email-follow-up-malware-download.pcap
435	2014-08-21-Sweet-Orange-EK-traffic.pcap
436	2014-09-11-Asprox-phishing-malware-live-analysis.pcap
437	2014-09-27-Angler-EK-traffic.pcap
438	2014-10-01-phishing-malware-analysis-from-malwr.pcap
439	2014-10-30-FlashPack-EK-traffic.pcap
440	2014-11-06-Nuclear-EK-traffic.pcap
441	2014-11-21-fake-av-traffic.pcap
442	2014-12-03-phishing-malware-malwr-analysis.pcap
443	2014-12-12-Nuclear-EK-traffic.pcap

444	2015-01-12-Sweet-Orange-EK-traffic.pcap
445	2015-01-26-Dridex-traffic-from-infected-VM.pcap
446	2015-02-02-malspam-email-infected-VM-traffic.pcap
447	2015-02-11-Windigo-Group-Nuclear-EK-traffic-example-01.pcap
448	2015-03-01-Magnitude-EK-initial-infection.pcap
449	2015-03-30-Fiesta-EK-infection-traffic.pcap
450	2015-04-03-Nuclear-EK-traffic.pcap
451	2015-04-25-Angler-and-Magnitude-EK-traffic.pcap
452	2015-05-11-malspam-campaign-traffic.pcap
453	2015-05-21-Fiesta-EK-infection-after-visajourney.com.pcap
454	2015-06-12-Nuclear-EK-traffic.pcap
455	2015-06-15-Angler-EK-traffic.pcap
456	2015-07-16-Rig-EK-traffic.pcap
457	2015-07-27-Angler-EK-sends-CryptoWall-3.0-traffic.pcap
458	2015-08-07-Rig-EK-traffic.pcap
459	2015-08-26-Upatre-malspam-infection-traffic.pcap
460	2015-09-14-Angler-EK-sends-CryptoWall-3.0-traffic.pcap
461	2015-09-23-bartalex-word-macro-sends-pony-and-vawtrak.pcap
462	2015-10-18-BizCN-gate-actor-Nuclear-EK-traffic.pcap
463	2015-10-27-Angler-EK-sends-TeslaCrypt-2.1-traffic.pcap
464	2015-11-16-Rig-EK-traffic-first-run.pcap
465	2015-11-27-Angler-EK-traffic.pcap
466	2015-12-04-Angler-EK-sends-TeslaCrypt-traffic.pcap
467	2015-12-08-Angler-EK-traffic.pcap
468	2016-01-11-Rig-EK-sends-Qbot-traffic.pcap
469	2016-01-25-EITest-Angler-EK-traffic.pcap
470	2016-02-07-Rig-EK-traffic-first-run.pcap
471	2016-02-19-Admedia-Angler-EK-after-alcomedicalservices.com.pcap
472	2016-03-02-admedia-Angler-EK-after-dompetdhuafa.org.au.pcap
473	2016-03-14-Rig-EK-after-dtransrentcar.com.pcap
474	2016-04-01-psuedo-Darkleech-Angler-EK-after-iteliness.com.mx.pcap
475	2016-04-19-traffic-caused-by-a-TeslaCrypt-malspam-attachment.pcap
476	2016-05-05-EITest-Neutrino-EK-sends-Cerber.pcap
477	2016-05-28-KaiXin-EK-traffic-from-threatglass.pcap
478	2016-06-01-pseudoDarkleech-Angler-EK-after-hideandseek.leadconcept.net.pcap
479	2016-06-08-2036-UTC-email.pcap
480	2016-07-15-other-Neutrino-EK-sends-Gootkit.pcap
481	2016-07-25-Magnitude-EK-sends-Cerber.pcap
482	2016-08-16-pseudoDarkleech-Neutrino-EK-sends-CrypMIC-after-blenheim-lodge.com.pcap
483	2016-08-23-fake-tech-support-popup-traffic.pcap
484	2016-09-12-locky-malspam-traffic-first-example.pcap
485	2016-09-26-1804-UTC-Locky-malspam-traffic.pcap
486	2016-10-04-Afraidgate-Rig-EK-sends-Locky-ransomware.pcap
487	2016-10-26-malspam-traffic.pcap
488	2016-11-13-pseudoDarkleech-RIGv-sends-Cerber-ransomware.pcap
489	2016-11-29-Locky-malspam-infection-traffic-example-from-1st-wave.pcap

490	2016-12-08-Sundown-EK-first-run.pcap
491	2016-12-21-Afraidgate-Rig-V-sends-Locky-ransomware.pcap
492	2017-01-11-EITest-Rig-V-sends-CryptoMix-ransomware-1st-run.pcap
493	2017-01-24-pseudoDarkleech-Rig-V-sends-Cerber-ransomware.pcap
494	2017-02-06-Afraidgate-Rig-EK.pcap
495	2017-02-14-EITest-Rig-EK-sends-CryptoShield-ransomware-1st-run.pcap
496	2017-03-13-Kovter-Locky-malspam-traffic.pcap
497	2017-03-30-Terror-EK-traffic.pcap
498	2017-04-07-1st-run-EITest-campaign-HoeflerText-popup-sends-Spora-ransomware.pcap
499	2017-04-25-Good-man-campaign-Rig-EK-sends-Latentbot.pcap
500	2017-05-04-fake-Adobe-Flash-player-site.pcap
501	2017-05-23-Jaff-ransomware-malspam-traffic.pcap
502	2017-06-12-Trickbot-malspam-traffic.pcap
503	2017-06-21-Rig-EK-sends-Bunitu.pcap
504	2017-07-10-Kovter-Nemucod-malspam-traffic.pcap
505	2017-07-24-Trickbot-malspam-traffic.pcap
506	2017-08-02-Magnitude-EK-sends-Cerber-ransomware.pcap
507	2017-08-28-Fobos-campaign-Rig-EK-sends-Bunitu.pcap
508	2017-09-06-Japanese-malspam-pushing-Ursnif-traffic.pcap
509	2017-09-15-Locky-binary-from-ekolapsm.top.pcap
510	2017-10-06-Boleto-malspam-traffic.pcap
511	2017-10-17-post-infection-traffic-from-Terror-EK-payload.pcap
512	2017-11-12-Mercury-Text-pop-sends-coinminer.pcap
513	2017-11-28-payment-slip-malspam-traffic.pcap
514	2017-12-13-Hancitor-malspam-traffic.pcap
515	2017-12-27-Necurs-Botnet-malspam-traffic.pcap
516	2018-01-12-Rig-EK-sends-Smoke-Loader-sends-Monero-coinminer.pcap
517	2018-01-14-Rig-EK-sends-Monero-coinminer.pcap
518	2018-02-01-Trickbot-infection-traffic.pcap
519	2018-02-08-malspam-pushing-Quant-Loader-1st-run.pcap
520	2018-03-09-Loki-Bot-malspam-traffic.pcap
521	2018-03-22-GoDaddy-phish-traffic.pcap
522	2018-04-06-Rig-EK-traffic-1st-run.pcap
523	2018-04-20-fake-Spotify-login-page-traffic.pcap
524	2018-05-08-Brighttax-malspam-infection-traffic.pcap
525	2018-05-27-1st-run-Grandsoft-EK-and-post-infection-traffic.pcap
526	2018-06-13-Necurs-Botnet-malspam-infection-traffic-for-Flawed-Ammyy.pcap
527	2018-06-29-Trickbot-infects-client-then-moves-to-DC.pcap
528	2018-07-04-fake-update-page-sends-Chthonic.pcap
529	2018-07-05-1st-run-fake-update-page-sends-Dridex.pcap
530	2018-07-25-Rig-EK-itraffic.pcap
531	2018-08-01-traffic-from-link-in-DHL-themed-malspam.pcap
532	2018-08-16-Hancitor-malspam-infection-traffic.pcap
533	2018-09-06-Emotet-infection-with-IcedID-and-AZORult.pcap
534	2018-10-10-fake-updater-infection-traffic.pcap
535	2018-10-26-GlobeImposter-ransomware-from-password-protected-Word-doc-malspam.pcap

536	2018-11-06-Emotet-infection-with-Trickbot.pcap
537	2018-11-26-Lokibot-infection-traffic.pcap
538	2018-12-05-Hancitor-infection-with-Ursnif.pcap
539	2018-12-10-Emotet-infection-with-IcedID.pcap
540	2019-01-22-2nd-run-Emotet-infection-with-IcedID.pcap
541	2019-01-22-3rd-run-Emotet-infection-with-Gootkit.pcap
542	2019-02-22-infection-traffic-from-malspam-pushing-Vidar.pcap
543	2019-02-28-Fallout-EK-from-HookAds-campaign.pcap
544	2019-03-06-Flawed-Ammyy-traffic.pcap
545	2019-03-20-Spelevo-EK-sends-malware.pcap
546	2019-04-05-Fake-Updates-Campaign-pushes-Chthonic.pcap
547	2019-04-24-Banload-infection-traffic-from-Brazil-malspam.pcap
548	2019-05-20-Formbook-infection-traffic.pcap
549	2019-05-23-Lokibot-infection-all-traffic.pcap
550	2019-06-25-Rig-EK-and-Pitou.B-traffic.pcap
551	2019-06-28-fake-updates-campaign-sends-Chthonic.pcap
552	2019-07-02-Trickbot-infection-with-CookiesDll-module.pcap
553	2019-07-25-Hancitor-style-Amadey-with-Pony-and-Cobalt-Strike.pcap
554	2019-08-02-Lord-EK-sends-Eris-Ransomware.pcap
555	2019-08-26-SocGholish-campaign-fake-Chrome-update-pushes-NetSupport-RAT.pcap
556	2019-09-05-1st-run-Word-doc-causes-Ursnif-infection-with-Trickbot.pcap
557	2019-09-24-Quasar-RAT-infection-traffic.pcap
558	2019-10-15-Shade-ransomware-infection-traffic.pcap
559	2019-10-21-Ursnif-infection-with-IcedID.pcap
560	2019-11-11-Emotet-epoch-1-infection-with-Trickbot-gtag-mor41.pcap
561	2019-11-25-Ursnif-infection-with-Dridex.pcap
562	2019-12-20-Emotet-epoch-2-with-Trickbot-mor70.pcap
563	2019-12-27-Qakbot-infection-traffic.pcap
564	2020-01-15-RevengeRAT-infection-traffic.pcap
565	2020-01-24-Ursnif-infection-traffic.pcap
566	2020-02-04-socgholish-traffic-example.pcap
567	2020-02-25-Trickbot-gtag-red4-infection-traffic.pcap
568	2020-03-03-Ursnif-infection-traffic.pcap
569	2020-03-27-IcedID-infection-traffic.pcap
570	2020-04-15-Hancitor-infection-1st-run.pcap
571	2020-04-30-Dridex-infection-from-attachment-in-German-malspam.pcap
572	2020-05-05-Qakbot-spx113-infection-traffic.pcap
573	2020-05-27-IcedID-infection-from-COVID19-themed-Word-doc.pcap
574	2020-06-12-Qakbot-spx139-with-ZLoader-infection-traffic.pcap
575	2020-06-15-Lokibot-infection-traffic.pcap
576	2020-06-30-Valak-infection-with-IcedID.pcap
577	2020-07-13-Dridex-infection-traffic.pcap
578	2020-07-20-IcedID-infection-traffic.pcap
579	2020-08-05-Emotet-infection-traffic.pcap
580	2020-08-10-Emotet-infection-with-Qakbot.pcap
581	2021-01-19-Qakbot-infection-traffic.pcap

582	2021-02-05-Spelevo-EK-sends-SmokeLoader.pcap
583	2021-03-18-Hancitor-infection-traffic.pcap
584	2021-10-13-Dridex-infection-traffic.pcap
585	2021-09-29-Hancitor-with-Cobalt-Strike-traffic.pcap
586	2021-03-08-Spelevo-EK-sends-ZLoader.pcap
587	2021-04-09-part-1-JS-file-retrieves-installer-DLL.pcap
588	2021-09-14-Hancitor-with-Cobalt-Strike-infection-traffic.pcap
589	2021-02-04-Rig-EK-sends-possible-BuerLoader.pcap
590	2021-06-15-Hancitor-with-Ficker-Stealer-and-Cobalt-Strike.pcap
591	2021-01-05-PurpleFox-EK-and-post-infection-traffic.pcap
592	2021-08-30-traffic-from-STRRAT-infection.pcap
593	2021-05-26-Trickbot-infection-with-Cobalt-Strike.pcap
594	2021-06-17-Hancitor-infection-with-Cobalt-Strike.pcap
595	2021-04-06-Hancitor-infection-with-Ficker-Stealer-and-attempted-Cobalt-Strike.pcap
596	2021-04-23-part-2-installer-DLL-causes-IcedID-infection.pcap
597	2021-05-13-Hancitor-traffic-with-Ficker-Stealer-and-Cobalt-Strike.pcap
598	2021-02-24-Qakbot-infection-with-spambot-traffic.pcap
599	2021-10-01-TR-Qakbot-infection-wtih-spambot-acitvity.pcap
600	2021-03-19-IcedID-infection-traffic-carved.pcap

Ek 8 : Testlerde Kullanılan Zararsız Trafikler

Çizelge Ek.8 : Zararsız Trafiklerin Listesi.

ID	Dosya Adı
1	AdGuard_dump_00001_20200114102945.pcap
2	AdGuard_dump_00002_20200114114901.pcap
3	AdGuard_dump_00003_20200114130936.pcap
4	AdGuard_dump_00004_20200114141606.pcap
5	AdGuard_dump_00005_20200114153502.pcap
6	CloudFlare_dump_00001_20200113152847.pcap
7	CloudFlare_dump_00002_20200113162614.pcap
8	CloudFlare_dump_00003_20200113182754.pcap
9	CloudFlare_dump_00004_20200113193921.pcap
10	CloudFlare_dump_00005_20200113205730.pcap
11	Google_dump_00001_20200113100617.pcap
12	Google_dump_00002_20200113111300.pcap
13	Google_dump_00003_20200113120939.pcap
14	Google_dump_00004_20200113132407.pcap
15	Google_dump_00005_20200113142226.pcap
16	Quad9_dump_00001_20200111222621.pcap
17	Quad9_dump_00002_20200111232233.pcap
18	Quad9_dump_00003_20200112004539.pcap
19	Quad9_dump_00004_20200112015042.pcap
20	Quad9_dump_00005_20200112025337.pcap
21	Quad9_dump_00006_20200112040340.pcap
22	Quad9_dump_00007_20200112051950.pcap

23	Quad9_dump_00008_20200112063544.pcap
24	Quad9_dump_00009_20200113074803.pcap
25	AdGuard_dump.pcap
26	AdGuard_dump2.pcap
27	CloudFlare_dump.pcap
28	CloudFlare_dump2.pcap
29	Google_large_dump.pcap
30	Google_medium_dump.pcap
31	Google_small_dump.pcap
32	Quad9_dump.pcap
33	Quad9_dump2.pcap
34	Monday-WorkingHours.pcap
35	IG_1_mail_youtube_googlesearch_downloading.pcapng
36	IG2_alisveris_siteleri_datasetindirme.pcapng
37	testbed-11jun.pcap
38	testbed-12jun.pcap