T.C.
İSTANBUL ÜNİVERSİTESİ-CERRAHPAŞA
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

TUZAK SİSTEMLER ÜZERİNDE TOPLANAN ZARARLI DOSYALARIN ANALİZİ

Melike BAŞER

DANIŞMAN
Doç. Dr. Muhammed Ali AYDIN

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Aralık, 2022

Bu çalışmayı her daim bana destek olan aileme ithaf ediyorum…

BÜTÇE DESTEKLERİ

[TUZAK SİSTEMLER ÜZERİNDE TOPLANAN ZARARLI DOSYALARIN ANALİZİ]

Bu tez çalışması için herhangi bir kurumdan bütçe desteği alınmamıştır.

TEŞEKKÜR

Eğitim hayatım süresince beni destekleyen ve her daim yanımda olan aileme, süreçte danışmanlığımı yapan Doç. Dr. Muhammed Ali AYDIN' a, projenin aşamalarında çeşitli tavsiyeler veren Arş. Gör. Ebu Yusuf GÜVEN' e en içten dileklerimle teşekkürlerimi sunarım.

Aralık 2022 Melike BAŞER

İÇİNDEKİLER

Sayfa No

TEZ KABUL VE ONAYI ..... Hata! Yer işareti tanımlanmamış.
BEYAN ..... Hata! Yer işareti tanımlanmamış.
BÜTÇE DESTEKLERİ ..... **iii**
TEŞEKKÜR ..... **iv**
İÇİNDEKİLER ..... **v**
ŞEKİL LİSTESİ ..... **vii**
TABLO LİSTESİ ..... **viii**
SİMGE VE KISALTMA LİSTESİ ..... **ix**
ÖZET ..... **x**
ABSTRACT ..... **xii**
1. GİRİŞ **1**
1.1. Genel Bilgi 1
1.2. Problem Tanımı 3
1.3. Tezin Amacı ve Önemi 3
1.4. Araştırma Yöntemi 4
1.5. Tezin Katkısı 4
1.6. Tezin Yapısı 5
2. KAVRAMSAL ÇERÇEVE **6**
2.1. Tuzak Sistemler (Honeypots) 7
2.1.1. Tarihçe 7
2.1.2. Tuzak Sistemlerin Sınıflandırılması 8
2.1.3. Genel Uygulamalı Tuzak Sistemler 12
2.1.4. IoT-IIoT-CPS Tuzak Sistemler 14
2.2. Zararlı Yazılım 16
2.2.1. Zararlı Yazılım Türleri 17
2.2.2. Zararlı Yazılım Tespit Teknikleri 21
2.2.3. Zararlı Yazılım Analiz Yöntemleri 23
2.2.4. Zararlı Yazılımların Tespit ve Analizi için Kullanılan Ortamlar 26

2.2.5. Zararlı Yazılımların İsimlendirilmesi 29
2.3. Bilgi Entropisi 30
2.4. İlgili Çalışmalar 30
3. YÖNTEM 36
3.1. Cowrie: SSH ve Telnet Tuzak Sistemi 38
3.2. Deneysel Ortamın Hazırlanması 40
3.3. Kötü Amaçlı Yazılımların Toplanması 41
3.4. Kötü Amaçlı Yazılımların Değerlendirilmesi 42
3.4.1. Kötü Amaçlı Yazılımlarda İmza Tabanlı Tespit Tekniğinin Kullanılması 44
3.4.2. Tespit Edilemeyen Kötü Amaçlı Yazılımların Binwalk ile Entropi Analizi 45
4. BULGULAR 46
4.1. VirusTotal Sonuçlarının Genel Değerlendirmesi 46
4.2. Anti-virüs Uygulamalarının Karşılaştırılması 49
4.3. Tespit Edilemeyen Dosyaların Binwalk ile Entropi Analizi Sonucu 51
5. TARTIŞMA 56
6. SONUÇ VE ÖNERİLER 60
KAYNAKLAR 64
EKLER 72
İNTİHAL RAPORU İLK SAYFASI 86
ETİK KURUL İZİN YAZISI ..... Hata! Yer işareti tanımlanmamış.
KURUM İZNİ YAZILARI 87
PATENT HAKKI İZNİ ..... Hata! Yer işareti tanımlanmamış.
ÖZGEÇMİŞ ..... Hata! Yer işareti tanımlanmamış.

ŞEKİL LİSTESİ

Sayfa No

Şekil 1 Pyramid of Pain (Emek/Acı Piramidi) 2
Şekil 2 Bal Küplerinin Sınıflandırılması 9
Şekil 3 Yıllara Göre Toplam Zararlı Yazılım ve PUA Dağılımı 17
Şekil 4 Zararlı ve İstenmeyen Yazılımların Sınıflandırılması 18
Şekil 5 Zararlı Yazılımların İsimlendirilmesi 29
Şekil 6 Yapılan Çalışmanın Akış Diyagramı 37
Şekil 7 Cowrie Çalışma Mantığı 38
Şekil 8 Cowrie Dosya Yapısı ve Şüpheli Dosyaların Toplanması 41
Şekil 9 Zararlı yazılımların sınıflandırılması 43
Şekil 10 Cowrie Bal küpü için geliştirilen Python uygulaması akış diyagramı 44
Şekil 11 Kötü amaçlı yazılım tespitinde anti-virüs uygulamalarının değerlendirme sonucu yüzdeleri 46
Şekil 12 Şüpheli dosyaların etiketlendiği zararlı yazılım sınıflandırma oranları 47
Şekil 13 Oylama Sonucu Değerlendirilmesi 48
Şekil 14 Binwalk uygulaması ile dosyanın entropi analizinin yapılması 51
Şekil 15 Zararlı olduğu tespit edilen dosyalar ile şüpheli dosyaların entropi analizlerinin offset bazında analizi 52
Şekil 16 Bal küpüne yüklenen dosyaların offset bazında entopi analizi 53
Şekil 17 diff komutu ile iki dosyanın karşılaştırılması 54
Şekil 18 81cc93ba0d6d32e cat komutu ile okunması 54
Şekil 19 baed3518e5758ec cat comutu ile okunması 55
Şekil 20 f4817f0ee7a893a cat komutu ile okunması 55

TABLO LİSTESİ

Sayfa No

Tablo 1 Saldırgan ile olan etkileşimlerine göre tuzak sistemlerin karşılaştırılması [20] 10
Tablo 2 Statik analizde kullanılan bazı araçlar 24
Tablo 3 Dinamik analizde kullanılan bazı araçlar 25
Tablo 4 Cowrie Kullanıcı Adı ve Şifre Değişimi 39
Tablo 5 Anti-virüs uygulamalarının şüpheli dosyalara verdiği cevapların anlamları 43
Tablo 6 Anti-virüs Uygulamalarının Zararlı Olarak Yanıt Verdiği Şüpheli Dosya Sayısı 49
Tablo 7 Şüpheli Dosyaları Bulamayan veya Cevap Dönmeyen Anti-virüs Uygulamaları 50
Tablo 8 Entropi analizi yapılan şüpheli dosyaların entropi değerleri 53

SİMGE VE KISALTMA LİSTESİ

Kısaltmalar	Açıklama
IoT	: Internet of Things
IIoT	: Industrial Internet of Things
CPS	: Cyber Physical Systems
SSH	: Secure Shell
HTML	: Hypertext Markup Language
DTK	: Deception Toolkit
FTP	: File Transfer Protocol
MQTT	: Message Queuing Telemetry Transport
SIP	: Session Initiation Protocol
SMB	: Server Message Block
TFTP	: Trivial File Transfer Protocol
UPnP	: Universal Plug&Play
TCP	: Transmission Control Protocol
IP	: Internet Protocol
DNS	: Domain Name System
NTP	: Network Time Protocol
SFTP	: Secure File Transfer Protocol
SNMP	: Simple Network Management Protocol
PLC	: Programmable Logic Controller
PUA	: Potentially Unwanted Application
DL	: Deep Learning
URL	: Uniform Resource Locator
CARO	: Computer Antivirus Research Organization
MIPS	: Microprocessor without Interlocked Pipeline Stages
ARM	: Acorn RISC Machine
PE	: Portable Executaion
KNN	: K Neearest Neighbour
LR	: Logistic Regression
SVM	: Support Vector Machine
DT	: Decision Tree

ÖZET

YÜKSEK LİSANS TEZİ

TUZAK SİSTEMLER ÜZERİNDE TOPLANAN ZARARLI DOSYALARIN ANALİZİ

Melike BAŞER

İstanbul Üniversitesi-Cerrahpaşa

Lisansüstü Eğitim Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Danışman: Doç. Dr. Muhammed Ali AYDIN

Dijital teknolojiler hayatın her alanında yaygınlaşırken siber tehditler de artarak devam etmektedir. Özellikle zararlı yazılımlar veri çalmak, sistem kaynaklarını (güç, ram, hafıza veya network) sömürmek, cihazı uzaktan kontrol edilen bir robota dönüştürmek veya sistem verilerini şifreleyerek fidye istemek gibi amaçlar için geliştirilmektedir. Son kullanıcılar cihazlarını korumak için antivirüs yazılımları kullanmaktadır. Son kullanıcı cihazlarına bulaşan zararlı yazılımların anti-virüsler ile tespit edilmesi ve uzmanların incelemesi için toplanması oldukça uzun ve riskli bir süreçtir. Güvenlik araştırmacıları saldırgan davranışlarını öğrenmek, yeni saldırı yöntemlerini keşfetmek ve zararlı yazılımları elde etmek amacıyla bal küpü tuzak sistemlerinden faydalanmaktadır. Çalışmada Google Cloud üzerine kurulmuş olan tuzak sistem aracılığıyla toplanmış saldırganlar tarafından yüklenen şüpheli dosyaların için imza tabanlı tespit yöntemine dayalı antivirüsler ve statik analiz yöntemleri ile değerlendirmesi yapılmıştır. Tez kapsamında kurulan bal küpüne gelen saldırılar geliştirilen Python betiği ile otomatik olarak VirusTotal API üzerinde bulunan 64 farklı antivirüs uygulamasına sorulmuştur. Çalışmada gelen saldırılardan güncel tehdit trendleri elde edilmiş ve VirüsTotal üzerinde

bulunan antivirüs yazılımlarının şüpheli dosyalar için oluşturduğu yanıtlar değerlendirilmiştir. Ayrıca, imza tabanlı tespit yöntemini kullanan antivirüs güvenlik sistemlerinin kendi içinde karşılaştırması da yapılmıştır. Çalışmada değerlendirme içerisine katılan 64 antivirüsten yalnızca 3 tanesi %80 üstü başarı göstermiştir. Başarı sıralamasında ilk 10 antivirüs uygulamasının başarı oranı %70'lere düşerken ilk 20 antivirüs değerlendirildiğinde başarı oranının %54' e düştüğü görülmüştür. Bu sonuçlara göre imza tabanlı tespit yöntemine dayalı güvenlik yazılımlarının güncel saldırılara karşı tek başına yetersiz kaldığı görülmektedir. İmza tabanlı tespit tekniğinin farklı tespit teknikleriyle birlikte tercih edilmesinin güvenlik için önemi ortaya koyulmuştur. Çalışma kapsamında imza tabanlı yöntemler tarafından tespit edilemeyen şüpheli dosyalara entropi analizi, kaynak kod incelemesi gibi statik analiz yöntemleri uygulanmıştır. Tez çalışması kapsamında toplanan saldırı kayıtları ve 50 şüpheli dosya diğer araştırmacıların araştırmalarında ve çalışmalarında kullanması için Github üzerinde paylaşılmıştır.

Aralık 2022 , [105] sayfa.

Anahtar kelimeler: [Zararlı yazılım tespit yöntemleri, İmza tabanlı tespit yöntemi, Tuzak sistemler, Zararlı yazılım örnekleri, Anti-virüs, Entropi]

ABSTRACT

M.Sc. THESIS

ANALYSIS OF MALICIOUS FILES GATHERED ON HONEYPOT SYSTEMS

Melike BAŞER

Istanbul University-Cerrahpasa

Institute of Graduate Studies

Department of Computer Engineering

Computer Engineering

Supervisor : Assoc. Prof. Dr. Muhammed Ali AYDIN

Cyber threats are rising as digital technologies permeate all aspects of society. Malicious software is primarily created to steal data, abuse system resources (such as power, RAM, or memory), transform a device into a remote-controlled robot, or demand ransom by encrypting system data. End users use antivirus software to protect their devices. Finding malicious software that has infected end-user devices with antiviruses and gathering it for expert assessment is an inefficient, time-consuming, and dangerous operation. Systems with honeypot traps help security researchers gather malware and learn more about the tactics used by attackers. In the study, attackers uploaded suspicious files obtained through the trap system set up on Google Cloud and were examined with antivirus software using static analysis methods and signature-based detection. The built Python script automatically requested 64 different antivirus software on the Virus Total API to perform assaults on the honeypot deployed as part of the thesis. The study assessed the reactions of antivirus software on Virus Total for suspicious files. Cyber Security researchers learned current threat patterns from incoming attacks. Additionally, internal comparisons of antivirus protection programs that employ signature-based detection were made. Only 3 of the 64 antivirus programs evaluated in the research had 15 efficiencies of greater than 80%. When the first 20 antiviruses were considered, the success

ratio decreased to 54%, and the success rate for the first ten antivirus software declined to 70% in the success order. These findings show that security software relying only on signature-based detection methods is ineffective against current attacks. It has been demonstrated how crucial it is to choose a signature-based detection method and other detection methods for security. In the context of the investigation, suspicious files that were not identified by signature-based techniques were subjected to static analysis techniques, including entropy analysis and source code analysis. The attack records and 50 suspicious files collected within the scope of the thesis were shared on Github for other researchers to use in their research and studies.

December 2022, 105 pages.

Keywords: Malware detection techniques, Signature-based detection technique, Honeypot, Malware samples, Antivirus, Entropy

1. GİRİŞ

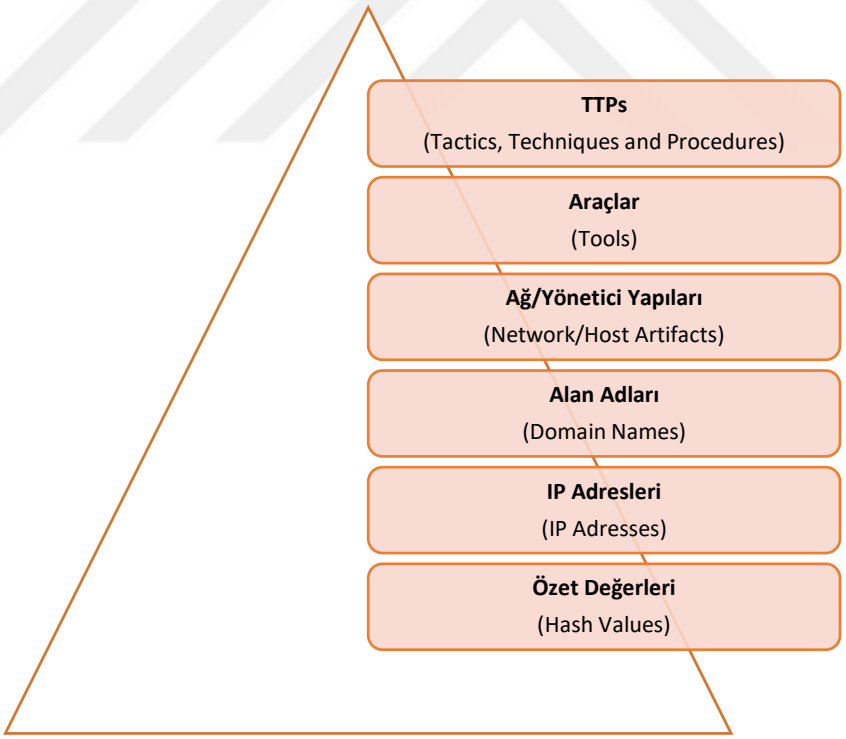
1.1. Genel Bilgi

İnternet ve bilişim sistemlerinin gelişme hızı artarken siber saldırı riskleri de artmaktadır. Pandemi dönemi ve sonrasında yaygınlaşan uzaktan çalışma kültürü akıllı cihazlara yönelik saldırılarda artışa neden oldu [1]. Siber saldırganların geliştirdikleri kötü amaçlı yazılımlar sistemlerin güvenliğini tehlikeye atan ciddi saldırı vektörleri arasında sayılabilir. Zararlı yazılım analizinin bilimsel ve teknik zorluklarının yanında güvenli bir yoldan zararlı yazılımları elde etmek de olukça zordur. Şüpheli dosyaları değerlendirmek için dosyaların özet fonksiyonlarından geçirilerek imza tabanlı, sistem üzerindeki hareketlerinden davranış tabanlı, kaynak kodu üzerinden statik analiz işlemleri yapılabilir. Zararlı yazılım elde etmek için bir sisteme bulaşan zararlı yazılımların toplanması gerekmekte veya açık kaynaklar üzerinden araştırılması gerekmektedir. Güncel saldırılarda kullanılan zararlı yazılımları bulmak için saldırganları çekerek onları gerçek bir sisteme erişim kazandıklarını düşündüren ve bal küpü (honeypot) [2] olarak isimlendirilen tuzak sistemlerin de kullanımı araştırmacılar tarafından tercih edilmektedir.

2022 Zararlı Yazılım İstatistikleri ve Gerçekleri raporunda[1], zararlı yazılımların hala gelişmekte olduğu ve her zamankine göre daha karmaşık yapıda olduğu görülmektedir. Zararlı yazılım (Malware) [3], önemli bilgileri çalmak ve uzaktaki hedef sisteme erişim sağlamak ve/veya sistem kaynaklarına zarar vermek gibi hedefleri olan kötü amaçlı bir yazılımdır. Zararlı yazılımların, Virüs (Virus), Truva atı (Trojan Horse), Solucan (Worm), Rootkit, Botnet, Arka Kapı (Backdoor) gibi türleri mevcutken Casus Yazılım (Spyware), Reklam Yazılımı (Adware) gibi potansiyel olarak istenmeyen uygulama (Potentially Unwanted Application - Grayware) türleri de mevcuttur [4]. Zararlı yazılım, günümüzde son kullanıcıların karşılaştığı büyük güvenlik tehditlerinden biri olarak görülebilir. Mevcut koşullar altında, bilgisayar ağları, taşınabilir bellek sürücüler, korsan yazılımlar ve daha fazlasını kullanarak yayılabilen zararlı yazılımları önlemek neredeyse imkansızdır. İstatistikler, her bilgisayara en az bir virüs bulaştığını göstermektedir [5].

[1] https://www.comparitech.com/antivirus/malware-statistics-facts/

Zararlı yazılımların tespitinde yaygın olarak kullanılan yöntemlerden biri antivirüs uygulamalarıdır. Antivirüs uygulamaları, dosyaları sürekli tarayarak ve bu dosyaların imzalarını bilinen zararlı yazılım imzalarıyla karşılaştırarak zararlı yazılımları algılar ve sınıflandırır. Zararlı yazılım imzaları genellikle toplanan zararlı yazılım örneklerini inceleyen antivirüs uzmanları tarafından oluşturulur. Yazılım veya dosya imzaları; dosya adı, dosya içeriğinin tamamı veya belli bölümlerinin ve bayt kodunun ifadeleri olabilir. İmza tabanlı tespit yöntemleri yalnızca veri tabanlarında imzası bulunan zararlı yazılımları tespit edebilir. Siber Tehdit İstihbaratının tehdit algılama operasyonlarında etkili bir şekilde kullanılması için kavramsal bir model olan ve **Şekil 1'** de gösterilen Emek/Acı Piramidi' ne (Pyramid of Pain) [6] göre özet (hash) bilgisinin değişikliği saldırganların en az çaba ve zaman harcadığı ayak izi (foot print) olarak değerlendirilmektedir [7]. Buna göre, zararlı yazılımlar, kodun orijinal sürümünden oldukça farklı görünmesini sağlayan kod gizleme (code obfuscation) tekniklerini kullanarak kötü amaçlı yazılımın imzasını gizleyebilmesi imza tabanlı sistemlerin güvenilirliğini tartışmalı hale getirmektedir.



Şekil 1 Pyramid of Pain (Emek/Acı Piramidi)

1.2. Problem Tanımı

Teknolojinin hızlı bir şekilde gelişmesi ve özellikle pandemi döneminde uzaktan çalışma kültürü ile dijital dünyanın genişlemesi beraberinde teknolojik ve dijital sistemlerin risk altında olduğu gerçeğini meydana getirmiştir [8]. Siber dünyada sistemlerin güvenliğinin sağlanması giderek önem kazanmaktadır [9]. Sistemlerin bozulmasına, devre dışı bırakılmasına, sistemlerden bilgi çalınmasına sebep olan önemli saldırı çeşitlerinden biri de zararlı yazılımlardır. Zararlı yazılımların tespitinde ve önlenmesinde birçok teknik ve yöntem kullanılmaktadır. Son kullanıcıların sıklıkla tercih ettiği anti-virüs uygulamaları da imza tabanlı tespit tekniklerini kullanmaktadır. Zararlı yazılımların tespitinde kullanılan "İmza Tabanlı Tespit" tekniği genellikle antivirüs uygulamalarında kullanılmakta olup bilinen zararlı yazılımların imzalarını (hash) eşleştirerek zararlı yazılım tespitini gerçekleştirir. Zararlı yazılımların tespitini ve analizini gerçekleştirmek için bu teknikte zararlı yazılımlardan oluşan geniş bir veri tabanına ihtiyaç duyulmaktadır. Kurum/kuruluşların ve son kullanıcıların sistemlerine bulaşan zararlı yazılım saldırılarından elde edilen örnekler antivirüs uygulamaları için önemli verilerdir. Yeni geliştirilen zararlı yazılım türlerini bir sisteme bulaşmadan önce imza tabanlı tekniklerle tespit etmek olanaksızdır. Yeni geliştirilen veya aynı zararlı yazılım ailesinden gelip farklı bir varyant olarak kabul edilen türler imza tabanlı tespit tekniği ile tespit edilememektedir. Tez çalışmasında bir bal küpü sistemi kurularak toplanan güncel zararlı yazılımlar ile imza tabanlı tespit tekniği kullanan antivirüs uygulamalarının başarısı araştırılmıştır. Tespit edilemeyen şüpheli dosyalar içinse statik analiz kapsamında kod okuma ve entropi analizi yöntemleri uygulanmıştır.

1.3. Tezin Amacı ve Önemi

Bu çalışma kapsamında bir tuzak sistemin (honeypot) kullanılmasıyla bu tuzak sisteme gelen gerçek saldırılardan elde edilen şüpheli dosyaların zararlı olarak ayrımının yapılması, bu zararlı dosyaların antivirüs uygulamaları ile imza tabanlı tespit tekniği kullanılarak tespiti ve sınıflandırılması amaçlanmıştır. Bu sayede imza tabanlı tespit tekniğinin gerçek saldırılardan elde edilen zararlı dosyaları tespit etmedeki başarısının ve kullanılan farklı antivirüs uygulamalarının kendi aralarında tespit oranlarının karşılaştırılmasının yapılması hedeflenmektedir. İmza tabanlı tespit yöntemlerinin tespit edemediği veya yeterli olmadığı şüpheli dosyaların statik analiz yöntemleriyle analiz edilmesi amaçlanmıştır. Böylece

kullanılan şüpheli dosya kümesinde tüm dosyaların zararlı/zararsız olarak etiketlenmesi hedeflenmiştir.

1.4. Araştırma Yöntemi

Bu çalışmada Ssh ve Telnet protokollerini simüle eden Cowrie tuzak sistemi kullanılarak gerçek saldırılardan elde edilen şüpheli dosyaların imza tabanlı tespit tekniği ve statik analiz yöntemleri ile tespiti gerçekleştirilmiştir. Şüpheli dosyalar VirusTotal aracı üzerindeki farklı antivirüs uygulamaları tespit edilmiş ve sınıflandırılmıştır. Tuzak sistem Cowrie üzerine saldırganlar tarafından yüklenen şüpheli dosyalar, geliştirilen Python betiği ile VirüsTotal API aracında bulunan çeşitli antivirüs uygulamalarının veri tabanlarında imza (hash) kontrolü yapılmıştır. Şüpheli dosyaların imzaları 64 farklı antivirüs uygulaması değerlendirme sonucuna göre oylama yapılarak zararlı yazılım ailesi sınıflandırması yapılmıştır. Ayrıca antivirüs uygulamalarının şüpheli dosyalara verdiği cevaplar da değerlendirilerek antivirüs uygulamaları kendi aralarında karşılaştırılmıştır. Tespit edilemeyen şüpheli dosyalar statik analiz yöntemlerinden kod okuma ve Binwalk aracı ile entropi analizi yapılarak incelenmiştir.

1.5. Tezin Katkısı

Bu çalışmanın başlıca katkıları şunlardır:

- Tuzak Sistemler (Honeypot), Zararlı Yazılım (Malware), Zararlı Yazılım Tespit ve Analiz yöntemleri literatürünün özetlenmesi,
- Zararlı dosyalardan oluşan yeni bir veri seti tanıtılması,
- Cowrie tuzak sistemine yüklenen dosyaların imzalarının VirusTotal kullanılarak tespitinin yapılması,
- Zararlı yazılımların İmza Tabanlı Tespit yöntemi kullanılarak tespitinin gerçekleştirilmesi, sınıflandırılması ve istatistiklerinin analiz edilmesi,
- İmza Tabanlı Tespit tekniğinin güncel saldırılara karşı başarısının incelenmesi,
- Antivirüs uygulamalarının değerlendirilmesi ve karşılaştırılmasının yapılması,

- Statik analiz yöntemlerinden kod okuma ve entropi analizi ile zararlı yazılım analizi yapılması,
- Antivirüs uygulamalarının çalışma mekanizmasının değerlendirilmesi ve imza tabanlı tespit yönteminin yeterliliğinin tartışılması.

1.6. Tezin Yapısı

Bu tez kapsamında Giriş bölümünde çalışma ile ilgili genel bilgiler verilmiş olup yapılan çalışmanın amacından ve katkılarından bahsedilmiştir. Tuzak Sistemler, Zararlı Yazılımlar Kavramsal Çerçeve başlığı altında detaylı bir şekilde anlatılmıştır. Ayrıca yapılan çalışmaya benzer literatürde bulunan ilgili diğer çalışmalar İlgili Çalışmalar alt başlığında yer almaktadır. Yapılan çalışmanın metodolojisi, kullanılan araçlar, sistem özellikleri ve deneysel ortam Yöntem bölümünde anlatılırken çalışmada elde edilen sonuçlara, tablolara, grafiklere ise Bulgular bölümünde yer verilmiştir. Elde edilen sonuçların incelenmesi ve yorumlanması Tartışma bölümünde; son olarak, bulguların tartışılması ve çözüm önerilerinin sunulması Sonuç ve Öneriler bölümünde yapılmıştır.

2. KAVRAMSAL ÇERÇEVE

Bilişim sistemlerinin ve özellikle İnternete bağlı teknolojilerin gelişmesiyle, bu sistemlerdeki güvenlik açıklarından kaynaklanan tehditlerin de artmasına sebep olmuştur. Güncel açıklıklar üzerinde güvenliğin sağlanabilmesi adına farklı türdeki saldırılara karşı, kimlik doğrulama, yetkilendirme, antivirüs uygulamaları gibi güvenlik çözümleri geliştirilmeye çalışılmıştır. Her gün hızla büyüyen saldırı tekniklerine karşı aynı şeyi savunma teknikleri özelinde söylemek zordur. Gerçekleşen bu saldırlar hakkında yeni teknikler keşfetmek ve önlemler almaya çalışmak oldukça önemlidir. Bilinen saldırı yöntemleri dışında keşfedilmemiş ve hakkında bilgi sahibi olunmayan sıfırıncı gün ataklarını (zero-day attacks) alınan önlemlerin birçoğu engelleyememektedir. Çoğu zaman saldırıya maruz kaldığı da anlaşılamamaktadır.

Günümüz bilişim dünyasında saldırıları tespit etmek ve yeni saldırı türlerini tanıyabilmek adına kullanılan en önemli yöntemlerden biri bal küpü (tuzak sistemler) sistemlerdir. Diğer yöntemlerin aksine saldırganı durdurmak için aktif rol oynamazlar. Amacı, zararlı aktivitenin kullanmış olduğu yöntem ve araçlar hakkında bilgi toplamaktır. Saldırgan ve saldırganın uyguladığı yöntem hakkında bilgi sahibi olmak mevcuttaki savunma sistemlerini geliştirmek için izlenilen önemli bir yol olmuştur. Bal küpleri, araştırmacıların bir sistemdeki kötü niyetli kullanıcı davranışlarını analiz etmelerini sağlar. Bal küpü sistemleriyle ilgili yapılan çalışmalar yıllardan beri güvenlik alanındaki önemini korumaktadır.

Bilinen ve yaygın olarak kullanılan saldırı çeşitlerinden biri de zararlı yazılımlardır. Zararlı yazılımlar bir sistemi bozabilir, servis dışı bırakabilir, sistemden bilgi toplayabilir. Zararlı yazılımların tespiti ve analizi için günümüze kadar geliştirilmiş birçok farklı yöntem ve araç bulunmaktadır. Fakat yeni saldırı yöntemlerinin geliştirildiği göz önünde bulundurulduğunda mevcut yöntemler etkisini yitirmektedir. Yaygın olarak kullanılan imza tabanlı tespit tekniği antivirüs uygulamalarının çalışma sisteminin temelini oluşturmaktadır. Geliştiriciden son kullanıcıya kadar genellikle zararlı yazılımlara karşı kullanılan en yaygın savunma aracı olan antivirüs uygulamalarının bu teknik ile tespitinin başarısı ve verimliliği bu çalışmada incelenmiş olup elde edilen sonuçlar tartışılmıştır.

2.1. Tuzak Sistemler (Honeypots)

Tuzak sistem terimsel olarak ifade edildiğinde Türkçe' ye 'bal küpü' olarak geçmiştir. Bal küpü kavramı, ayılar için kurulmuş tuzaklara yerleştirilen içi bal dolu kapları ifade eder [10]. Bilişim dünyasına uyarlandığında saldırganların tuzağa çekilmesi için tasarlanan ve kullanılan sistemlerdir. Bal küpü sistemleri, kötü amaçlı faaliyetleri veya saldırganları engellemede aktif bir rol oynamaz. Bal küpleri üzerinde herhangi bir ağ trafiği görülmez. Bal küpü üzerinde herhangi bir etkileşim gözlemlendiğinde bu etkileşimin bir saldırgan tarafından gerçekleştiriliyor olması yüksek bir ihtimaldir. Bal küpü sistemlerde kasıtlı olarak bırakılan açıklıklar saldırganların dikkatini çeker. Bal küpü sistemine saldırganın farklı saldırı teknikleri deneyebileceği açıklıklar bırakılır ve bu sayede saldırganın kullandığı yöntem ve araçlar hakkında bilgi toplanması, saldırı tekniklerinin izlenmesi amaçlanır. Bal küpü içerisinde saldırganın gerçekleştirdiği bütün aktiviteler kaydedilir ve kayıt altına alınan bu aktiviteler istihbarat analizinde kullanılabilir [11].

Honeynet kavramı Türkçe' de "Bal Ağı" olarak ifade edilmektedir. Bal ağları, yüksek etkileşimli bal küpleridir. Yüksek düzeyde etkileşim sunabilecek bir bal küpü çözümü tasarlamak oldukça zordur. Bir bal ağı, sıkı kontrollerden geçen ve yüksek oranda izlenen bir ağ üzerindeki yüksek etkileşimli bal küplerinin bir koleksiyonudur. Bu ağ sistemleri, bir tür erişim kontrol cihazının (güvenlik duvarı gibi) arkasına yerleştirilir ve ağın izini sürer. Saldırganlar, Honeynet içindeki herhangi bir sistemi inceleyebilir, saldırabilir ve istismar edebilir, bu da onlara etkileşimde bulunabilecekleri gerçek işletim sistemleri ve uygulamalar sağlar. Bal ağlarında hiçbir hizmet benzetilmez ve simüle ortamlar oluşturulmaz. Kısacası, bir Honeynet içindeki sistemler gerçek üretim sistemleridir [12].

2.1.1. Tarihçe

Bal küpü kavramı 1990 öncesinde ortaya çıkmasına rağmen, kullanım alanı yaygın değildi. 1991'de, günümüzün bal küpü gelişiminin temeli olacak kavramlar üzerine bir dizi yayın açıklandı. Özellikle iki yayın göze çarpıyordu: Bunlar Clifford Stoll' un "The Cuckoos Egg" [13] ve Bill Cheswick' in "An Evening with Berferd" [14] idi. 1997'de Fred Cohen, Deception Toolkit'i piyasaya sürdü. Deception Toolkit (DTK) [15], orijinal ve dönüm noktası niteliğindeki Honeypot' lardan biridir. Genellikle, bilinen çeşitli güvenlik açıklarını taklit eden UNIX sistemleri için tasarlanmış bir PERL betikleri koleksiyonudur.

CyberCop Sting 1998' de ticari olarak geliştirilen ilk bal küpü olmuştur. CyberCrop Sting, NT üzerinde çalışan CyberCrop saldırı koruma yazılımı ailesinin bir bileşenidir. Windows işletim sistemlerini simüle eden bir sisteme sahiptir. Ayrıca Telnet protokolünü de simüle edebilir. Aynı yıl piyasaya çıkan bir diğer ticari Honeypot uygulaması da NetFacade4 oldu. CyberCrop Sting'de olduğu gibi, görünüşte savunmasız hizmetleri çalıştıran, ancak çok daha büyük ölçekte simüle edilmiş IP adresleriyle simüle edilmiş bir ana bilgisayar ağı oluşturur. 1998, Honeypot gelişimi için özellikle yoğun bir yıldı çünkü CyberCrop ve NetFacade'den sonra Windows Honeypot "Back Officer Friendly" çıktı [16].

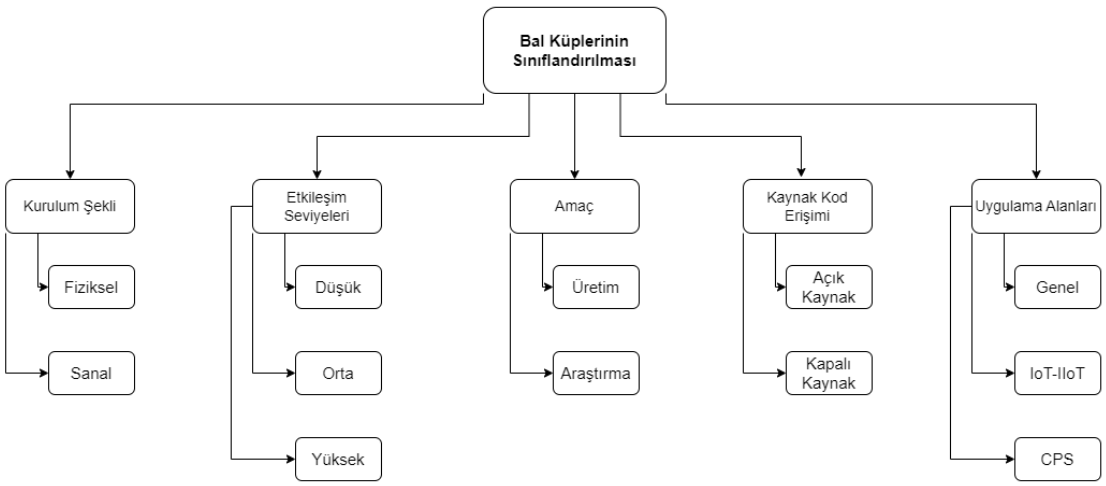
1999' da Lance Spitzner liderliğindeki bir grup insan Honeynet Projesi'ni [17] oluşturmaya karar verdi. Honeynet Projesi'nin kurulmasıyla birlikte yeni gelişmeler oluşmaya başlamıştır. Projenin hedefi, internette bulunan mevcut tehditlerle ilgili farkındalık oluşturarak gerekli araç ve yöntemleri sağlamaktı. Honeynet Projesi [2], siyah şapka (blackhat) topluluğunu araştırmaya ve çalışmalarını başkalarıyla paylaşma hedefi taşıyan ve kar amacı olmayan bir gruptur. 2002'de, honeypot ile ilgili araştırmaları ilerletmek için tüm güvenlik topluluğunu içeren Honeypot Research Alliance olarak adlandırılan bir girişim doğdu. Research Alliance, dünya çapında Honeypot araştırmasıyla ilgilenen gruplardan oluşur. Bu kurulum, Honeypot' un dünya çapında konuşlandırılmasına ve Honeypot araştırmasında kullanılan araç ve tekniklerin paylaşılması için imkan sağlamıştır.

2003 yılında, bu kuruluşlar tarafından Snort [18], Sebek [19] ve gelişmiş sanal bal ağları (Advanced Virtual Honeynets) gibi birkaç önemli Bal Küpü aracı tanıtıldı. O zamandan beri kablolu ve kablosuz ortamlar için bal küplerinde daha yeni projeler geliştirilmektedir. Honeynet Projesinde yer alan araçların ve bal küplerinin yanı sıra çeşitli kurumların ve araştırmacıların açık kaynak kodlu veya kapalı kaynak olarak geliştirdiği bal küpleri bulunmaktadır. Açık kaynak kodlu bal küplerinden ilerleyen başlıklarda bahsedilmiştir.

2.1.2. Tuzak Sistemlerin Sınıflandırılması

Bal küpü sistemleri temelde etkileşimlerine ve kurulum amaçlarına göre sınıflara ayrılırlar. Bu sınıflandırmanın genişletilmesi mümkündür. Kullanılan veya tasarlanana bal küpünün amacına göre sınıflandırma yapılabilir. Bu tez kapsamında bal küpleri kurulum şekillerine, etkileşim seviyelerine, amaçlarına, kaynak kod erişimlerine ve uygulama alanlarına göre beş farklı başlıkta sınıflandırılmaktadır. Bu sınıflandırma **Şekil 2'** de gösterilmiştir.

[2] https://www.honeynet.org/



Şekil 2 Bal Küplerinin Sınıflandırılması

2.1.2.1. Fiziksel ve Sanal Tuzak Sistemler

Bal küpleri kurulum şekli olarak ikiye ayrılmaktadır. İşlemci, bellek, sabit disk gibi fiziksel kaynaklara ihtiyaç duyulan ve gerçek işletim sistemlerine gerçek makineler olarak yüklenen bal küpleri Fiziksel Tuzak Sistemler veya Donanım Tabanlı Tuzak Sistemler olarak bilinmektedir. Bu bal küpleri kurulum ve bakım yönüyle maliyetli olabilmektedir.

Ana makine üzerinde çalışan sanal makineler (VMware, VirtualBox vb.) kullanılarak kurulan bal küpleri ise Sanal Tuzak Sistemler veya Yazılım Tabanlı Tuzak Sistemler olarak ifade edilmektedir. Fiziksel bal küplerine göre izinsiz girişler daha kolay gerçekleştirlirken, aktivite izlenimi ve sistemin korunması da kolaydır [20].

2.1.2.2. Etkileşim Tabanlı Sınıflandırma

Günümüz bal küpleri Düşük etkileşimli (Low-Interaction), Orta Etkileşimli (Medium-Interaction) ve Yüksek Etkileşimli (High-Interaction) olarak etkileşim seviyelerine göre üçe ayrılır. Buradaki etkileşim ile anlatılmak istenilen, saldırganın sistem ile gerçekleştirdiği aktivitenin seviyesi veya aktivite alanıdır.

Düşük Etkileşimli Bal küpü [21], sınırlı etkileşime sahiptir. İşletim sistemlerini ve üzerlerinde çalışan hizmetleri taklit ederek çalışır. Yalnızca sınırlı bilgileri kaydeder ve önceden bilinen saldırıları yakalamak için tasarlanmıştır.

Orta Etkileşimli Bal küpü [22], düşük etkileşimli tuzak sistemler gibi gerçek bir işletim sistemine sahip değildir. Ancak saldırgan ile daha çok etkileşime geçebilmesi ve daha karmaşık saldırıları üzerine çekebilmesi yönüyle düşük etkileşimli tuzak sistemlerden farklıdır.

Yüksek Etkileşimli Bal küpü [21], gerçek işletim sistemlerini ve uygulamaları içerir. Temel avantajı, saldırı hakkında elde edilen bilgilerin tüm faaliyetleri yakaladığı için çok detaylı olmasıdır. Sıfırıncı gün ataklarının (zero-day attacks) tespitinin yapılmasında veya saldırganın kullanmış olduğu teknikleri incelemek için idealdirler. Yüksek etkileşimli bal küpü sistemleri genellikle Honeynet örneği olarak birçok kaynakta anlatılmaktadır. Honeynet, iki veya ikiden fazla bal küpü sistemleri ile oluşturulmuş bal küpü ağına verilen addır.

Saldırganın tuzak sistem üzerinde gerçekleştirdiği aktiviteler sistemle olan etkileşimidir. Çeşitli faktörlere bağlı olarak saldırgan hangi tuzak sistemi ne zaman kullanacağını belirleyebilir. Etmenler ve tuzak sistem türlerinin bunlarla ilişkisi **Tablo 1**'de verilmiştir.

Tablo 1 Saldırgan ile olan etkileşimlerine göre tuzak sistemlerin karşılaştırılması [23]

Etmenler	**Düşük Etkileşimli**	**Orta Etkileşimli**	**Yüksek Etkileşimli**
Bulaşma derecesi	Düşük	Orta	Yüksek
Gerçek işletim sistemi	Yok	Yok	Var
Kurulum	Kolay	Zor	Çok zor
Bakım	Kolay	Kolay	Zaman alıcı
Risk	Düşük	Orta	Yüksek
Ele geçirilme beklentisi	Yok	Yok	Var
Kontrol gereksinimi	Yok	Yok	Var
Çalıştırmak için gerekli bilgi	Düşük	Düşük	Yüksek
Geliştirmek için gerekli bilgi	Düşük	Yüksek	Orta-Yüksek
Veri toplama	Kısıtlı	Orta	Kapsamlı
Etkileşim	Servis benzetimi	İsteklere göre	Tam kontrol

2.1.2.3. Kurulum Tabanlı Sınıflandırma

Bir bal küpü Üretim Bal küpü (Production Honeypot) ve Araştırma Bal küpü (Research Honeypot) gibi iki tipe ayrılabilir. Üretim bal küpleri genellikle diğer sunucular ile ağa yerleştirilir. Üretim bal küplerinin kurulumu ve kullanımı kolay olmakla beraber bu bal küpleri düşük etkileşimlidir. Araştırma bal küplerine oranla saldırılar ve saldırganlar ile ilgili daha az bilgi verirler.

Saldırganların amaçları ve saldırı teknikleri ile ilgili bilgilerin toplanması araştırma bal küplerinin en önemli amaçlarındandır. Kuruluşların karşılaştığı çeşitli tehditlere karşı bilgi sahibi olmak ve bu tehditler için savunma yöntemleri belirlemek için kullanılan araştırma bal küplerinin kurulumu ve bakımı kolay değildir. Araştırma bal küpleri, üretim bal küplerine göre daha detaylı bilgi topladıklarından genellikle araştırmacılar, devlet ve askeri kurumlar tarafından tercih edilmektedir [23].

2.1.2.4. Kaynak Kodunun Erişilebilirliğine Göre Sınıflandırma

Açık kaynak, bir yazılımın kaynak kodunun herkesin erişebileceği, değiştirebileceği ve/veya dağıtabileceği şekilde yayınlanması anlamına gelir. Açık kaynaklı yazılım, işbirlikçi geliştirmeye izin verir. Bal küpü ve bal ağı yazarlarının tümü, tuzak sistemlerinin kaynak kodunu sağlamaz. Kaynak kodunun kullanıma sunulması, diğer araştırmacıların ve geliştiricilerin mevcut bal küplerini ve bal ağlarını anlamasına ve iyileştirmesine olanak tanır.

2.1.2.5. Uygulama Alanına Göre Sınıflandırma

Uygulama, bal küpü sisteminin oluşturulduğu amacı ve verdiği hizmeti ifade eder. Bu tez kapsamında genel amaçlı kullanılan bal küpleri, IoT-IIoT ve CPS olarak uygulama sınıflandırması yapılmıştır. Genel kullanım bal küpleri, başlangıçta IoT için tasarlanmamış bal küpleridir. Ancak bunlar, daha sonra IoT bal küpleri ile yapılan araştırmalarda da kullanılmışlardır. IoT bal küpleri, genel IoT uygulamalarını ifade eden bal küpleridir. CPS ve IIoT bal küpü sistemlerine ICS, Akıllı Şebeke, Su Sistemi, Gaz Sistemi, Bina Otomasyon Sistemi örnek verilebilir.

2.1.3. Genel Uygulamalı Tuzak Sistemler

Bal küpleri simüle ettikleri sistemlere göre sınıflara ayrılabilirler. Simüle edilen sistem özel bir nitelik taşımıyorsa, çoğu sistemde kullanılabilir bir uygulama veya protokol ise bu bal küpleri genel uygulamalı tuzak sistemler olarak ifade edilebilir. Özel bir sistem, uygulama veya protokol simüle ediliyorsa tuzak sistemler de uygulama alanına göre isimlendirilebilir. Genel uygulamalı çoğu tuzak sistem özelleşmiş bal küplerine alternatif olarak da kullanılabilme özelliğine sahiptir. Örneğin, bal küpleri ve bal ağları IoT için özel olarak oluşturulmasa da IoT bal küpleri ve bal ağlarına yönelik araştırmalarda da kullanılmaktadır. Açık kaynak olarak geliştirilmiş ve yaygın olarak kullanılan genel uygulamalı bal küpleri başlıklar halinde verilmiştir.

2.1.3.1. Honeyd

HoneyD [3], düşük etkileşimli bal küplerinin oluşturulması için açık kaynaklı bir yazılımdır. Honeyd, sanal bal küpleri oluşturur ve fiziksel makine entegrasyonuna da izin verir. UDP, TCP, FTP, SMTP, Telnet, IIS ve POP protokollerini simüle edebilir. HoneyD' nin saldırganları çekmek için etkili bal küpleri oluşturup oluşturamayacağını inceleyen Stafira [24], IoT cihazlarını simüle eden bal küplerini gerçek cihazlarla karşılaştırmıştır. Sonuçlar hem bal küpleri hem de gerçek cihazlar tarafından sunulan içeriğin benzer olmasına rağmen, sorgu yanıtları ve Nmap taramaları için ortalama süreler arasında önemli farklılıklar olduğunu göstermiştir.

2.1.3.2. Dionaea

Dionaea [4], FTP, HTTP, MongoDB, MQTT, MySQL, SIP, SMB, TFTP, UPnP gibi çeşitli hizmetleri simüle edebilen orta etkileşimli ve açık kaynaklı bir bal küpü yazılımıdır [5]. Dionaea, saldırgana güvenlik açıkları sunarak kötü amaçlı yazılımları yakalamak için kullanılır. Dionaea kötü amaçlı yazılımın bir kopyasını elde etmeyi ve araştırmacıların bunu analiz etmesine yardımcı olmayı amaçlar. Dionaea statik bir konfigürasyona sahiptir, bu da konfigürasyonun gerektiği gibi uyarlanmasını zorlaştırır. Metognon ve Sadre [25], IoT bal küpü araştırmalarında Dionaea' yı kullanmışlardır. Kaur ve Pateriya [26], Raspberry Pi üzerinde Dionaea kullanarak IoT için uygun maliyetli bir bal küpü kurulmasını önermişlerdir.

[3] https://github.com/DataSoft/Honeyd
[4] https://github.com/DinoTools/dionaea
[5] https://dionaea.readthedocs.io/en/latest/introduction.html

2.1.3.3. Kippo

Kippo[6] açık kaynaklı, orta etkileşimli bir bal küpüdür. SSH protokolünün benzetilmesine odaklanır ve kaba kuvvet saldırılarının yanı sıra otomatik veya bireysel saldırılardan gelen etkileşimleri kaydeder. Dowling ve arkadaşları [27], bir ZigBee IoT bal küpü uygulaması için Kippo' yu kullanmışlardır. Günümüzde Kippo güncelleme almadığı için Cowrie bal küpü çoğunlukla tercih edilmektedir.

2.1.3.4. Cowrie

Cowrie[7], orta veya yüksek etkileşimli, açık kaynaklı bir sanal bal küpü yazılımıdır. Orta etkileşimli bir bal küpü olarak, birkaç komutu taklit eder ve bir saldırganın kabuk etkileşimini simüle edilmiş bir UNIX sisteminde kaydeder. Yüksek etkileşimli bir bal küpü olarak, bir saldırganın başka bir sistemdeki etkileşimini gözlemlemek için SSH ve Telnet için bir vekil sunucudur (Proxy). Cowrie, Kippo bal küpünden esinlenerek geliştirilmiştir ve SSH, Telnet, SFTP, SCP ve TCP/IP hizmetlerini simüle eder. Log kaydı, depolama ve görselleştirme için Elastic Search, LogStash ve Kibana entegrasyonunu destekler. Cowrie bal küpü detaylı olarak Yöntem bölümünde anlatılmaktadır.

2.1.3.5. HoneyPy

HoneyPy[8], simüle edilen hizmetlere bağlı olarak düşük ila orta etkileşimli bir bal küpü yazılımıdır. DNS, NTP, SIP, SMTP, Web vb. hizmetleri simüle etmek için kullanılabilecek çok çeşitli eklentilerle birlikte gelir. Ayrıca, gerektiğinde özelleştirilerek çalışacak şekilde yapılandırılabilir. HoneyPy, araştırmacılara, ElasticSearch, Logstash, RabbitMQ, Slack, Splunk, Twitter dahil çeşitli kayıt seçenekleri sunar. Böylece, HoneyPy kayıtlarının analizi için harici hizmetler kullanılabilir.

2.1.3.6. Glastopf

Web uygulaması, veri tabanı ve siteler arası betik çalıştırma, diğerlerinin yanı sıra web sitesini tahrif etmek, istenmeyen e-posta göndermek, web sitesini botlara dönüştürmek ve sürücü indirme saldırılarına hizmet etmek için kullanılabilecek geniş bir saldırı yüzeyine sahiptir. Glastopf [28], binlerce güvenlik açığına sahip birçok web sayfasını ve web uygulamasını barındıran savunmasız bir web sunucusunu taklit eden düşük etkileşimli bir bal küpüdür.

[6] https://github.com/desaster/kippo
[7] https://www.cowrie.org/
[8] https://github.com/foospidy/HoneyPy

Glastopf, 2009 Google of Summer Code'un bir parçası olarak öğrenci Lukas Rist tarafından geliştirildi.

2.1.4. IoT-IIoT-CPS Tuzak Sistemler

Nesnelerin İnterneti (IoT), sensörler, aktüatörler ve veri toplayabilen ve iletişim kurabilen diğer gömülü cihazlar gibi İnternet'e bağlı birçok cihazdan oluşan geniş bir ağdır. Endüstriyel IoT (IIoT), IoT'nin endüstriyel iletişim teknolojilerini kullanan otomasyon uygulamalarına uygulanmasıdır. Siber Fiziksel Sistemler (CPS) ise Sensörler, Aktüatörler, Programlanabilir Lojik Kontrolörler (PLC'ler), Uzak Terminal Birimleri (RTU'lar), Akıllı Elektronik Cihazlar (IED'ler), izleyen ve kontrol eden diğer gömülü cihazlar gibi cihazların ağlarıdır. CPS, Endüstriyel Kontrol Sistemleri (ICS), Akıllı Şebeke ve diğer akıllı altyapıları (ör. su, gaz, bina otomasyonu), tıbbi cihazları ve akıllı arabaları içerir ancak bunlarla sınırlı değildir.

IoT Honeypot, IoT sistemleri taklit eden tuzak sistemlerdir. Özellikle IoT alanındaki gelişmelerin artması ile IoT sistemler için oluşturulmuş bal küpü sistemlerine ihtiyaç duyulmuştur. Son yıllarda IoT sistemlere özel bal küpü sistemler geliştirilmiş ve araştırmalar da bu doğrultuda artmaya başlamıştır. En fazla bilinen ve birçok çalışmada kullanılmış IoT cihazlarına yönelik geliştirilmiş bal küpü sistemler alt başlıklar halinde verilmiştir.

2.1.4.1. ThingPot

Wang ve arkadaşları [29], tüm IoT platformunu ve desteklenen tüm uygulama katmanı protokollerini simüle eden orta etkileşimli, sanal, açık kaynaklı bir bal küpü olan ThingPot' u[9] önermişlerdir. ThingPot, Extensible Messaging and Presence Protocol (XMPP) ve REST API ile 45 gün boyunca test edilmiştir. Yazarlar, saldırganların Philips Hue, Belkin, Wemo ve TPlink gibi belirli cihazları aradıklarını, cihazlar hakkında bilgi almak için tarama yaptıklarını ve ardından onları kontrol etmek için kaba kuvvet (brute-force) veya fuzzing gibi daha hedefli saldırılar kullandıklarını belirtmişlerdir. Ayrıca saldırganların anonim kalmak için The Onion Router (TOR) ağını kullandığını da kaydetmişlerdir.

2.1.4.2. IoTCandyJar

Luo ve arkadaşları [30], akıllı etkileşim olarak tanımladıkları düşük/yüksek etkileşimli bal küplerinin özelliklerine sahip yeni bir tür bal küpü önermişlerdir. Bal küpü, İnternette herkese açık olan IoT cihazlarının davranışlarını otomatik olarak öğrenmek ve saldırganlarla oturumu uzatmak için hangisinin en iyi yanıtı verdiğini öğrenmek için Markov Karar Süreci ile Makine

[9] https://github.com/Mengmengada/ThingPot

Öğrenimini kullanır. IoTCandyJar, çalışma sırasında yaklaşık 1 milyon IoT ile ilgili istek dahil olmak üzere 18 milyon ham istek yakalamıştır. 80, 7547, 8443, 81, 8080 ve 88 en çok taranan numaralı bağlantı noktaları ve isteklerin çoğu HTTP olmuştur.

2.1.4.3. MTPot

MTPot[10], Mirai saldırıları için özel olarak tasarlanmış, düşük etkileşimli, sanal bir IoT bal küpüdür. MTPot Cymmetria Araştırma tarafından açık kaynak bal küpü olarak geliştirilmiştir. Mirai zararlı yazılımı ile etkileşimde olan makineleri tespit etmek ve aynı zamanda mümkünse Mirai zararlı yazılımı örnekleri toplamak için kullanılabilir. Diğer Telnet bal küpleri gibi bir Telnet sunucusunu simüle eder ve ayarlar bal küpünü enfekte etmeye çalışan Mirai' nin sürümüne göre değiştirilir [31].

2.1.4.4. HoneyThing

HoneyThing' in temel amacı modem ve yönlendiriciler için son yıllarda çıkmış popüler bazı açıklıklara karşı savunmasız, TR-069 protokolünü destekleyen bir sistem sunmak ve uygulama ile olan tüm etkileşimlerin detaylı bir şekilde kaydını tutmaktır. Sistem hem web ara yüzünü hem de CWMP protokolünü simüle edebilmektedir [22].

2.1.4.5. IoTPOT

Üniversitede bulunan araştırmacıların birlikte yaptığı IoT bal küpü çalışması olan IoTPOT, Telnet protokolüne gelen saldırılar için simüle edilmiş bir IoT bal küpü ve sandbox ortamından oluşur. Birçok IoT cihazının Telnet servisini simüle edebilen IoTPOT, iki bölümden oluşur: düşük etkileşimli bir arayüz ve IoTBOX adında yüksek etkileşimli bir sanal servis. Buna ek olarak IoTBOX, MIPS ve ARM mimari de dahil olmak üzere sekiz ayrı işlemci mimarisini destekler. Arayüz saldırgan tarafından yapılan bağlantı taleplerini ve komutları arkada çalışan servislere iletir, servislerden gelen yanıtlar da saldırgana iletir [32].

2.1.4.6. Conpot

Conpot, düşük etkileşimli bir SCADA bal küpüdür ve uygulanması son derece kolaydır. SCADA, su arıtma ve dağıtımı, elektrik enerjisi iletimi ve dağıtımı, petrol ve gaz boru hatları, ulaşım hizmetleri vb. gibi altyapıları denetlemek ve kontrol etmek için endüstriyel süreçlerde yaygın olarak kullanılan "Denetleyici Kontrol ve Veri Toplama (Supervisory Control and Data Acquisition)" anlamına gelir [11]. SCADA sistemleri, simüle edilmiş ağın bir parçası olarak Conpot' ta emüle edilir. PLC, FTP, Telnet, HTTP, SNMP ve Modbus gibi beş servis çalıştırır.

[10] https://github.com/Cymmetria/MTPot
[11] http://en.wikipedia.org/wiki/SCADA

Çoğu saldırı girişimi, saldırganlar tarafından HTTP, Modbus protokolleri üzerinde gerçekleştirilir [33].

2.1.4.7. XPOT

Lau ve arkadaşları tarafından ICS için orta etkileşimli bal küpü olarak tasarlanan XPOT [34], Siemens S7-300 serisindeki PLC' leri simüle eder ve saldırganın PLC programını derlemesine, yorumlamasına ve XPOT' a yüklemesine olanak tanır. S7comm ve SNMP protokollerini destekler.

2.1.4.8. U-PoT

U-PoT [35], UPnP protokolünü kullanan IoT cihazlarına yönelik saldırıları yakalamak için geliştirilmiş bir bal küpü sistemidir. Akıllı anahtarlar, akıllı ampuller, güvenlik kameraları vb. birçok akıllı ev cihazı UPnP protokolünü kullanır. Shodan IoT[12] arama motorunda yapılan aramada UPnP protokolünü kullanan 3,706,206 cihaz listelenir. UPnP protokolü kullanan IoT cihazının yaygınlaşması tam teşekküllü bir IoT bal küpü sistemi gerektirir. Geliştirilen bal küpü, UPnP tabanlı IoT cihazları için esnek ve yapılandırılabilir bir sistem sunar. Araştırmalara yön vermesi için U-PoT açık kaynak olarak yayınlanmış ve UPnP cihaz açıklamalarından oluşan bir veri tabanı oluşturulmuştur.

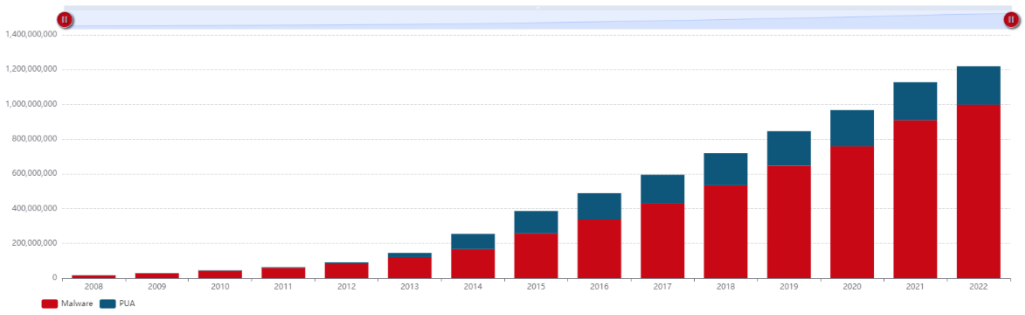
2.2. Zararlı Yazılım

Bilgi sisteminin gizliliği, bütünlüğü veya erişilebilirliği üzerinde olumsuz etkisi olacak yetkisiz bir işlem gerçekleştirmeyi amaçlayan yazılımlara zararlı yazılım denir [36]. Zararlı yazılımlar, siber saldırganlar tarafından hedefledikleri aktiviteleri gerçekleştirmeye olanak sağlayan özel olarak tasarlanmış yazılımlardır. Doğrudan bir hedef olabileceği gibi, hedefi desteklemek için bir araç olarak da geliştirilebilirler [37]. Farklı motivasyonlarla hazırlanmış zararlı yazılımlar tüm dijital sistemlerin mücadele etmesi gereken bir saldırı vektörleridir.

Zararlı yazılımlar bilgisayar sistemlerini bozmak veya bunlara zarar vermek için tasarlanır. Potansiyel olarak istenmeyen uygulamalar (Potentially Unwanted Application - PUA), doğrudan sisteme zarar vermeyen reklam yazılımı, casus yazılım veya varsayılan arama motoru değiştirme gibi özellikler içerebilir. Zararlı yazılımdan farklı olarak, PUA' lar genellikle faydalı bir özelliği olmayan, reklamlar yayınlayan, kullanıcıların davranışlarını izleyen veya verilerini

[12] https://www.shodan.io/

toplayan özellikleri bulundurur [38]. PUA' ların zararlı yazılımlara göre daha az yaygın olduğu **Şekil 3**'te yıllara göre görülmektedir.



Şekil 3 Yıllara Göre Toplam Zararlı Yazılım ve PUA Dağılımı [13]

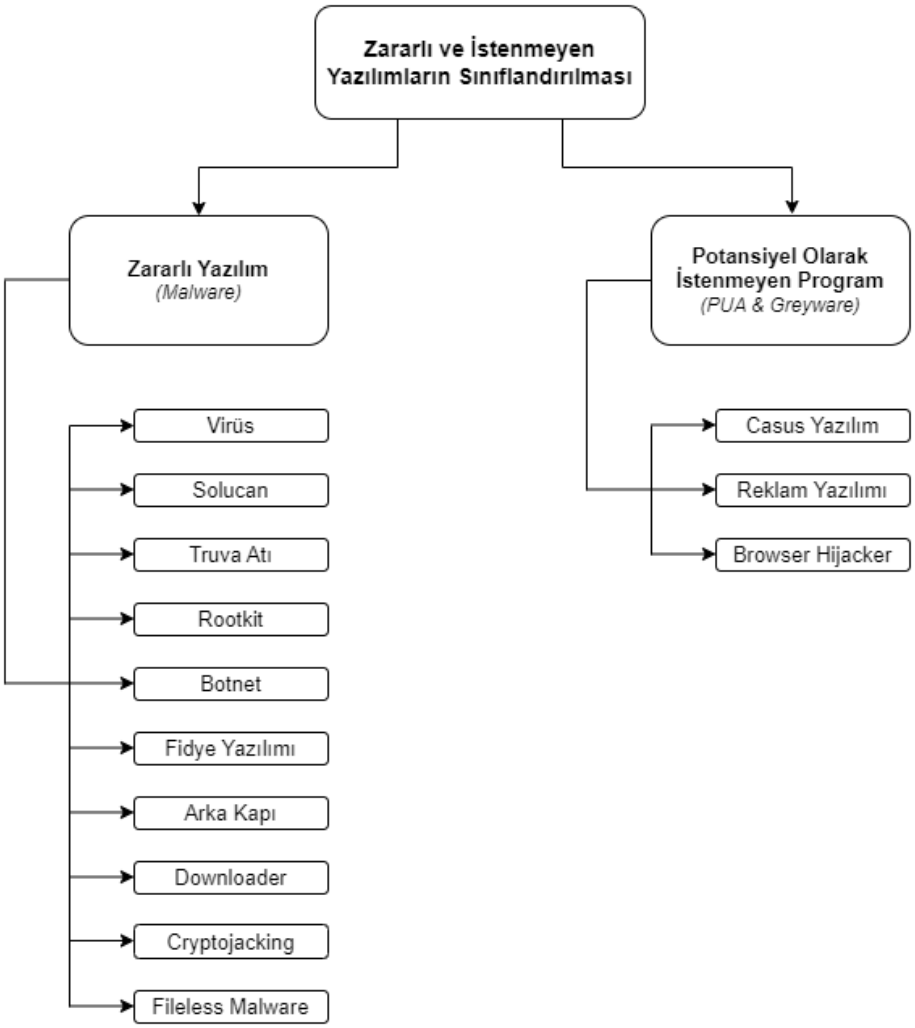
Zararlı yazılımlar gibi, PUA' lar da bellek, işlemler ve ağlar gibi bilgi işlem kaynaklarını kullanır ve örneğin sayfa etkileşimleri ve arama sorguları gibi bilgileri toplayarak kullanıcı gizliliği üzerinde olumsuz bir etkiye sahip olabilir. Zararlı yazılımlar PUA' lara göre daha kötü niyetli olarak görülmektedir. PUA' ların doğrudan sonuçları genellikle daha zararsız ve (yasal veya etik açıdan) belirsiz olarak algılanır. Bu nedenle, PUA' lara bazen "Greyware" adı verilir. Zararlı yazılımlar genellikle otomatik olarak kaldırılırken, PUA' ların kaldırılması genellikle bir kullanıcının seçimine bağlıdır [39].

2.2.1. Zararlı Yazılım Türleri

Zararlı yazılımlar; hedefleri, teknikleri ve yöntemleri hızla değişmekte ve gelişmekte olan siber tehdit vektörleridir. Dijital sistemlere yazılımsal ve/veya donanımsal zarar vermek; bilgi çalmak, kaynaklarını sömürmek veya bilgileri şifreleyerek fidye istemek gibi farklı amaçlarla tasarlanan yazılımlar zararlı yazılım (malware) olarak isimlendirilmektedir. Zararlı yazılımlarla mücadele için kullanılan yazılımlara anti-virüs yazılımları denilmektedir. Zararlı yazılımlar tespit için dosya imzaları, kaynak kod ve davranış analizi gibi farklı yöntemleri kullanan antivirüs uygulamaları ortaya çıkmıştır. Bu antivirüs uygulamalarının atlatılması için kötücül yazılımlar birçok yöntem kullanarak değişimler geçirmiştir [40]. Artan ve değişen tehdit vektörleri karşısında, siber güvenlik araştırmacılarının zararlı yazılımları tespit etme

[13] https://www.av-test.org/en/statistics/malware/

yöntemlerini de geliştirmesi gerekmiştir [41]. Araştırmacılar zararlı yazılımları değişimleri ve gelişimlerine, amaç ve hedeflerine, sömürdükleri zafiyetlere göre sınıflandırırlar. Geliştirilen ilk zararlı yazılım türünün eğlence amaçlı yazılım olduğu bilinirken, cihaz kaynaklarını sömürerek kripto para (Crypto Currency) üreten Cryptojacking zararlı yazılımı ortaya çıkmıştır [42], [43]. Zararlı yazılımlar için virüs, solucan, truva atı, fidye yazılımı gibi temel sınıflandırma yapılırken kök kullanıcı takımı, botnet, fileless malware gibi daha detaylı olarak yapılan sınıflandırmalar da bazı çalışmalarda mevcuttur [44], [45], [46]. Modern zararlı yazılımlar birden fazla saldırı vektörünü kullanarak saldırı gerçekleştirdikleri için bu zararlıları tanımlamak ve belirli bir sınıfa atamak zordur. Temel olarak zararlı yazılımlar **Şekil 4**' te alt başlıklar halinde verilmiştir.



Şekil 4 Zararlı ve İstenmeyen Yazılımların Sınıflandırılması

2.2.1.1. Virüs (Virus)

Bir uygulama dosyası üzerinden aktifleştirilmesiyle kullanıcının izni ve bilgisi dahilinde olmadan bilgisayara bulaşan ve kendini kopyalayabilen zararlı yazılımdır. Bir virüs sistemdeki verileri bozabilir veya silebilir, kendini diğer bilgisayarlara yaymak için e-posta programlarını kullanabilir ve sabit diskteki her şeyi silebilir [36].

2.2.1.2. Solucan (Worm)

Kendini yaymak için ağ mekanizmalarını kullanan, kendi kendine çoğalan ve yayılan bir programdır. Virüsler ve solucanlar arasındaki en önemli fark, bir virüsün çoğalmaya başlaması için virüslü dosyayı açmak gibi insan müdahalesi gerektirirken solucanlar müdahale olmadan çoğalabilir [47].

2.2.1.3. Truva Atı (Trojan Horse)

Yararlı bir işleve sahip gibi görünen, ancak bir sistem varlığının meşru yetkilerinden yararlanarak güvenlik mekanizmalarından kaçan gizli ve kötü amaçlı bir işleve sahip bilgisayar programıdır [36].

2.2.1.4. Casus Yazılım (Spyware)

Kullanıcının bilgisi ve rızası olmadan sistem üzerinden veri toplayan/aktaran yazılımlardır. Bu türdeki yazılımlar zararlı olarak düşünülse de sınıflandırma açısından PUA (greyware) başlığı altında bulunur. İnternet üzerinde yapılan aramaların kaydı ve izlenmesi, klavye verilerinin kaydedilmesi, görüntülenen web sayfalarının bilgilerini tutmak gibi birçok işlem yapabilirler. Virüsler gibi kendilerini kopyalama özellikleri yoktur [14].

2.2.1.5. Reklam Yazılımı (Adware)

Bilgisayarda veya cihazda istenmeyen reklamlar görüntüleyen PUA başlığı altonda sınıflandırılan yazılım anlamına gelir. Reklam yazılımları bazen güvenli olsa da bazı açılır pencereler yalnızca reklamları göstermeyi değil, aynı zamanda özelleştirilmiş reklamlarla veri ve bilgi toplamayı da amaçlar. Çeşitli reklam bağlantıları aracılığıyla kötü amaçlı web sitelerine ve virüslü sayfalara yönlendirebilir [15].

[14] https://faq.cc.metu.edu.tr/tr/sss/spyware-nedir
[15] https://www.eset.com/uk/types-of-cyber-threats/adware/

2.2.1.6. Browser Hijacker

Bir web tarayıcısının ayarlarını kullanıcının izni olmadan değiştiren, istenmeyen bir yazılım biçimi olarak tanımlanır. İstenmeyen reklamların tarayıcıya yerleştirilmesi ve mevcut bir ana sayfanın veya arama sayfasının korsan sayfayla değiştirilmesidir. Saldırganın daha yüksek reklam geliri elde etmesi için kullanıcıların isteseler de istemeseler de belirli web sitelerini ziyaret etmelerini sağlamaktır. Ayrıca, banka bilgileri gibi hassas verilere erişim için casus yazılım da içerebilirler [16].

2.2.1.7. Kök Kullanıcı Takımı (Rootkit)

Kötü niyetli kişilerin cihazlara ayrıcalıklı erişim sağlaması için tasarlanmış zararlı bir programdır. Rootkit kendi kendine çoğalamaz, sistemin kendisi için bir tehlike oluşturmazlar. Ancak genellikle kötü amaçlı yazılım yüklemesi gibi daha karmaşık tehdit ve saldırıların bir parçası olabilirler [48].

2.2.1.8. Botnet

Bilgisayar üzerinde bulunan ve uzaktan kontrolü sağlanan yazılım veya uygulamaya bot denir. Bu zararlının bulaştığı bilgisayar ağına ise botnet denir. “Robot” ve “Network” kelimelerinden türetilmiş olan botnet, bilgisayarların denetimini ele geçirmek ve virüs bulaşmış bilgisayarları suçlular tarafından uzaktan yönetilebilen bir "botlar" ağına dahil edebilmek için Truva atı zararlı yazılımını kullanır [17].

2.2.1.9. Fidye Yazılımı (Ransomware)

Hedef alınan kişinin veya kurumun dosyalarını şifreleyen zararlı yazılım türüdür. Siber tehdit aktörü bu saldırı ile, kurban fidyeyi ödeyene kadar kurbanın verilerine erişmesini engellemektedir. İki tür fidye yazılımı vardır. Bunlardan ilki kilitleyici fidye yazılımı (Locker ransomware) ve ikincisi şifreli fidye yazılımıdır (Crypto ransomware). Kilitleyici fidye yazılımı temel bilgisayar işlevlerini etkilerken Şifreli fidye yazılımı belirli dosyaları şifreler [49].

2.2.1.10. Arka Kapı (Backdoor)

Siber suçluların veya saldırganların bilgisayarlara uzaktan erişmesine izin veren uygulamalardır. Arka kapılar, mobil cihazlarda ve akıllı cihazlarda kötü amaçlı uygulamalar aracılığıyla da yayılabilir [18].

---

[16] https://us.norton.com/blog/malware/what-are-browser-hijackers
[17] https://www.kaspersky.com.tr/resource-center/threats/botnet-attacks
[18] https://www.trendmicro.com/vinfo/us/security/definition/backdoor

2.2.1.11. (Downloader)

Bir bilgisayara başka kötü amaçlı yazılımları indiren bir Truva atı türüdür. İndiricinin dosyaları indirebilmesi için internete bağlanması gerekir [36]. Tehdit algılama motorlarını tetiklemeyen basit bir araç olarak tasarlanmıştır ve farklı zararlı yazılımlar indirmek ve çalıştırmak için kullanılabilir.

2.2.1.12. Cryptojacking

Kripto madenciliği (Crypto mining), kripto madencilerinin kripto para birimi işlemlerini doğrulamak ve çalışmalarının karşılığı olarak kripto para kazanmak için bilgisayarları, verileri, kodları ve hesaplamaları kullandığı süreçtir. Cryptojacking, bilgisayar korsanlarının kendi çıkarları için gizlice kripto varlık madenciliği yapmak üzere kurbanın bilgi işlem kaynakları sömürmesidir. Kullanıcının cihazları (bilgisayarlar, akıllı telefonlar, tabletler ve hatta sunucular) üzerinde gizlice kripto para madenciliğinin yapıldığı zararlı yazılım türüdür. Cryptojacking diğer zararlı yazılımların aksine kullanıcıdan kendini gizlemek amacını taşır [42]. Coin Miner ise kripto para madenciliği yapmak için kurbanın bilgisayarındaki donanım öğelerini kullanan bir zararlı yazılım türüdür.

2.2.1.13. Fileless Malware

Bir cihaza bulaşmak için hedef sistemde dosya indirmek veya kurmak yerine cihazdaki mevcut yazılımları (WMI, Powershell etc.) kullanarak sisteme bulaşan yazılımlardır. İmza algılama, donanım doğrulama, model analizi, zaman damgası gibi güvenlik önlemlerini atlatabilir ve genelde arkasında herhangi bir iz bırakmaz. İz bırakmadığından dolayı "İmza Tabanlı Antivirüs" sistemlerinden kaçabilir [42].

2.2.2. Zararlı Yazılım Tespit Teknikleri

Zararlı yazılım tespit teknikleri, zararlı yazılımı belirlemek, tespit etmek, bilgisayar sistemine zararlı yazılım bulaşmasını önlemek ve sistemi korumak için kullanılır. Zararlı yazılım tespit teknikleri imza tabanlı, davranış tabanlı ve sezgisel tabanlı olmak üzere 3 ayrı kategoride sınıflandırılabilir. Bu teknikler dışında zararlı dosyaların türlerine veya yapılacak çalışmaların amacına göre geliştirilen teknikler de kullanılmaktadır. Tibra Alsmadi ve arkadaşı çalışmasında imza tabanlı, sezgisel tabanlı ve kum havuzu (sandbox) tabanlı tespit teknikleri olarak sınıflandırma yapmışlardır [50]. Bilgisayar sistemlerine kurulan antivirüs uygulamaları genellikle imza tabanlı tespit tekniği kullanır ve güvenlik yazılımlarının temelini oluşturur [51]. Ayrıca davranış tabanlı ve sezgisel tabanlı tespit tekniklerini kullanan yazılımlar da mevcuttur.

2.2.2.1. İmza Tabanlı Tespit Tekniği (Signature-based Detection)

İmza (signature), belirli zararlı yazılımları tanımlamak için kullanılan bir bayt dizisidir. Zararlı yazılım geliştirilirken bu zararlı yazılımın hangi aileye ait olduğunu belirlemek için kullanılan ve imza olarak bilinen bir bit dizisi zararlı yazılımın koduna gömülür. İmza tabanlı tespit tekniği, antivirüs uygulamalarının çoğu tarafından kullanılır. Antivirüs uygulaması, virüslü dosyanın kodunu parçalara ayırır ve bir zararlı yazılım ailesine ait imzayı arar. İmza tabanlı tespit tekniğini kullanan antivirüs yazılımları, bilinen zararlı yazılımların imzalarından oluşan bir depo bulundurur ve yeni tehditler keşfedildikçe bu depo güncellenir.

İmza tabanlı tespit tekniği basit ve hızlı olmasının yanında yalnızca yaygın kullanılan zararlı yazılım türlerine karşı etkilidir. Önceden bilinmeyen ve imzası bulunmayan zararlı yazılımları tespit etmede yetersiz kalmaktadır [52]. Antivirüs yazılımları güncel bir imza veri tabanına sahip olmadığında veri tabanında bulunmayan zararlı yazılımlar algılanamaz ve tespit edilemez. Ayrıca, imza tespitinden kaçınmak için de çeşitli şaşırtma teknikleri kullanılmaktadır. Bu tekniklerin kullanılması halinde zararlı yazılım algılanamaz ve aynı aileye ait zararlı yazılımlar küçük değişikliklerle imza tabanlı tespit tekniğinden kaçarlar [53]. İmza tabanlı tespit tekniği en yaygın kullanılan tekniklerden biri olsa da son yıllarda zararlı yazılımların artan karmaşıklığıyla birlikte, zararlı yazılımları yakalamak için artık yeterli olmamaktadır. Bu nedenle, zamanla araştırmacılar davranış tabanlı ve sezgisel tabanlı tespit yaklaşımları geliştirdiler. Ek olarak, son yıllarda DL, bulut, mobil ve IoT tabanlı algılama yaklaşımları da kullanılmaktadır [52]. Zararlı yazılım tespitinde kullanılan tekniklere genel bir bakış alt başlıklarda verilmektedir.

2.2.2.2. Davranış Tabanlı Tespit Tekniği (Behaviour-based Detection)

Davranış tabanlı zararlı yazılım tespit tekniği, izleme araçlarıyla program davranışlarını gözlemler ve programın kötü amaçlı mı yoksa iyi huylu mu olduğunu belirler. Program kodları değiştirilse de programın davranışı benzer olacağından yeni çıkan zararlı yazılımların çoğu bu yöntemle tespit edilebilir [54]. Bu teknikte sanal makine, sandbox ve/veya zararlı yazılımın davranışlarını incelemek için geliştirilmiş özel araçlar kullanılabilir. Bu araçların kullanımının getirdiği olumsuz yönlerden biri de zararlı yazılımın bütün davranışlarını göstermeme ihtimalinin olmasıdır [55].

2.2.2.3. Sezgisel Tabanlı Tespit Tekniği (Heuristic-based Detection)

Bir yazılım dosyasından ve/veya sistem üzerindeki davranışlarından (memory, işlemci, disk vb.) çıkarılan özelliklerin yapay zeka yöntemleriyle değerlendirildiği tespit yöntemidir [56]. Yapay zeka modelleri imza tabanlı ve davranış tabanlı analiz yöntemlerinin çıktılarını özellik olarak kullanabilmektedir. Uygulanan yapay zeka algoritmasına göre geliştirilen model daha önce görülmeyen öğrenilmemiş zararlı yazılımları tespit edebilir.

2.2.3. Zararlı Yazılım Analiz Yöntemleri

Zararlı yazılım analizi, zararlı yazılım tespit tekniği geliştirmek için gerekli bir adımdır. Siber güvenlik analistleri tarafından zararlı yazılımların tespitinin ve bu yazılımlardan kaçınmayı kolaylaştırmak için zararlı yazılımın işini nasıl gerçekleştirdiğinin analiz edilmesi sürecidir. Zararlı yazılım analizi iki ana tekniğe ayrılır: statik analiz ve dinamik analiz. Bunların yanı sıra hibrit analiz ve bellek (memory) analiz teknikleri de kullanılmaktadır [57], [58].

2.2.3.1. Statik Analiz (Static Analysis)

Statik analiz, programı fiilen çalıştırmadan gerçekleştirilir. Temel analiz, bilinen tehditlerde veri tabanını aramak için imza veya özet (hash) kullanır, ancak bu yöntem yeni kötü amaçlı yazılımları tespit edemez. Şüpheli dosyanın nasıl sınıflandırılacağını veya analiz edileceğini ve sonraki analiz çalışmalarının hangi yöne odaklanması gerektiği konusunda doğru karar verilmesini sağlayan ve genellikle ilk aşamada kullanılan analiz yöntemidir [59]. Gelişmiş analiz, yürütülebilir dosyayı derleme koduna dönüştürmek ve programın yapısını ortaya çıkarmak için ayrıştırıcılar ve hata ayıklayıcılarla tersine mühendislik uygular. Ardından analist, analiz edilen kod ile diğer kötü amaçlı programlar arasındaki benzerlikleri bulmak için araştırma yapar. Bu teknik, bilinmeyen zararlı yazılımları bulmak için kullanılır, bu nedenle ileri düzeyde bilgi gerektirir. Gizleme nedeniyle bu yöntem zor ve güvenilmezdir. Yaygın olarak kullanılan statik analiz teknikleri şunları içerir: API çağrısı, metin analizi, kontrol akış grafiği ve operasyon kodu (opcode). **Tablo 2**'de statik analiz için kullanılan bazı araçlar verilmiştir.

Tablo 2 Statik analizde kullanılan bazı araçlar

Statik Analiz Araçları	**Açıklama**	**Link**
YARA	Zararlı yazılım örneklerinin tanımlanmasına ve sınıflandırılmasına yardımcı olmayı amaçlayan açık kaynak kodlu bir araçtır.	virustotal.github.io/yara/
EXEinfo	.exe dosyalarını doğrulamayı ve tüm özelliklerini kontrol etmeyi sağlayan bir programdır.	exeinfo-pe.en.uptodown.com/windows
Compute hash	Seçilen dosyanın MD5, SHA1, SHA256, SHA384 ve SHA512 özetlerini (hash) hesaplayan bir uygulamadır.	subisoft.net/ComputeHash.aspx
PEstudio	Zararlı yazılımın ilk değerlendirmesini hızlandırmak için yürütülebilir dosyalar içindeki şüpheli yapıları bulmaya yönelik bir araçtır ar içindeki şüpheli yapıları bulmaya yönelik bir araçtır	winitor.com/features

2.2.3.2. Dinamik Analiz (Dynamic Analysis)

Dinamik analiz, bir kodu veya komut dosyasını çalıştırarak sistemdeki ağ, memory, işlemci, RAM veya uygulamalar üzerindeki değişiklikleri çeşitli araçlarla gözlemleyerek analiz etme sürecini ifade eder. Dinamik analizin amacı, yürütülebilir dosya tarafından gerçekleştirilen kötü amaçlı etkinliği, analiz platformunun güvenliğinden ödün vermeden çalışırken ortaya çıkarmaktır [60]. Böylece, ikili dosyaları güvenli bir ortamda çalıştırarak çalışma zamanı özellikleri hızlı bir şekilde gözlemlenir. Dinamik analiz sırasında ağ ortamını taklit etmek ve zararlı yazılımı kandırarak İnternet' e bağlı olduğunu düşünmesini sağlamak önemli görevler arasındadır. **Tablo 3**'te dinamik analiz için kullanılan bazı araçlar verilmiştir.

Tablo 3 Dinamik analizde kullanılan bazı araçlar

Dinamik Analiz Araçları	**Açıklama**	**Link**
FakeNet	Dinamik ağ analiz aracıdır. Ağ hizmetlerini simüle ederken tüm veya belirli ağ trafiğini kesmeye ve yeniden yönlendirmeye olanak tanır.	fireeye.com/services/freeware/fakenet-ng.html
RegShot	Windows Kayıt Defteri' nin (Windows Registry) anlık görüntüler ile öncesi ve sonrası karşılaştırmasına olanak tanıyan araçtır.	sourceforge.net/projects/regshot/
ProcMon	Windows sistemlerde gerçek zamanlı etkinlikleri gösteren bir izleme aracıdır.	docs.microsoft.com/en-us/sysinternals/downloads/procmon
ProcDot	Sistem aktivitelerini görselleştiren bir araçtır. Aynı zamanda ProcMon aracının çıktılarını da görselleştirir.	cert.at/en/downloads/software/software-procdot

Autoruns	Microsoft'un bir Windows makinesinde otomatik olarak başlayan tüm programları sıralayan aracıdır.	docs.microsoft.com/enus/sysinternals/downloads/autorun
Wireshark	Ağ trafiğini gerçek zamanlı olarak Windows, Mac, Unix ve Linux sistemleri için analiz eden bir araçtır.	wireshark.org/

2.2.3.3. Hibrit Analiz (Hybrid Analysis)

Hibrit analiz tekniği, herhangi bir zararlı yazılım imzasının analiz edilmesi ve kötü amaçlı yazılım analizini geliştirmek için sistem performansını etkileyen veya çalışmasını sekteye uğratan işlem gücünü tüketme, hafıza ve uygulamalar üzerindeki değişiklikler gibi davranışsal parametrelere bakılır. Statik ve dinamik analiz tekniklerinin eksikliklerinin üstesinden gelmek için kullanılan bu yöntemle, zararlı yazılımları doğru bir şekilde tespit etme yeteneği artmış olur [57].

2.2.3.4. Hafıza Analizi (Memory Analysis)

Bellek analizi, zararlı yazılım analizinde etkili bir teknik haline geldi. Bellek analizi çalışan programlar, işletim sistemi ve bilgisayarın genel durumu hakkındaki bilgileri analiz etmek için bellek görüntüsünü kullanır. Araştırmalar bellek ele geçirme (acquisition) ve bellek analizi olarak iki adımdan oluşur. Bellek toplamada, Memoryze, FastDump ve DumpIt gibi araçlar kullanılarak hedef makinenin belleği boşaltılır. Bellek analizinde, Volatility ve Rekall gibi araçlar kullanılarak bellek görüntüleri analiz edilir [58].

2.2.4. Zararlı Yazılımların Tespit ve Analizi için Kullanılan Ortamlar

Zararlı yazılımlar, kullanıcının bilgisi dışında makineye zarar verecek şekilde tasarlanmıştır. Farklı ailelere ait zararlı yazılımlar, sisteme bulaşarak geri dönüşü olmayan hasarlara neden olur ve bu nedenle zararlı yazılımların tespitinin yapılması ve analiz edilmesi gerekir. Tüm zararlı yazılım türlerinin manuel analizi, büyük çaba harcamayı gerektirdiğinden pratik bir yaklaşım değildir. Araştırmacılar üzerindeki yükün azaltılabilmesi ve sürecin iyileştirilmesi için Otomatik Zararlı Yazılım Analizi ‘ne (Automated Malware Analysis) başvurulur. Hem

çevrimdışı hem de çevrimiçi olarak birçok Otomatik Zararlı Yazılım Analizi aracı mevcuttur, ancak şüpheli bir ikili (binary) dosyayı analiz ederken hangi aracın seçileceği sorunu ortaya çıkar. Her yöntem için çeşitli araçlar mevcuttur, fakat bu tez kapsamında özelleşmiş araçların genel başlıkları verilmiştir. Yaygın olarak kullanılan zararlı yazılım tespit ve analiz ortamları başlıklar halinde verilmiştir.

2.2.4.1. Anti-virüs

İmza tabanlı tespit tekniğine dayalı antivirüsler, zararlı yazılımlara karşı ana savunma hatlarından biri olmuştur [61]. Başlangıçta virüs olarak adlandırılan, literatürde antivirüs olarak görünen yazılım, yerleşik terminolojiye göre antivirüs olarak kullanılmaya devam ediyor, ancak virüs kelimesi günümüzde alt başlık olarak kalmaktadır. Antivirüs yazılımları, daha önce tespit edilmiş zararlı yazılım imzalarını (MD5 vb.) bir veri tabanında depolar ve şüpheli dosyaları bu veri tabanındaki bilgilerle karşılaştırır. Antivirüsler, bilinmeyen veya veri tabanında bulunmayan imzalara sahip zararlı yazılımları tespit edemez. Antivirüsler zararlı yazılımın yapısını veya sistem içinde nasıl çalıştığını kontrol etmezler. Günümüzde geliştirilen zararlı yazılımlar antivirüs yazılımları tarafından algılanmamak için görünümünü (imzasını ve/veya kodunu) değiştirir. Antivirüslerin tüm zararlı yazılımlar üzerinde etkili olmamasının nedenlerinden biri, zararlı yazılımların yalnızca imzayı değil, çalışan kodun kendisini de değiştiren bir yapıya sahip olmasıdır.

2.2.4.2. VirusTotal

VirusTotal[19], şüpheli dosyaları, özetleri (hash) veya uzantıları (URL) analiz etmek için masaüstüne indirilebilen veya çevrimiçi API bulunduran ücretsiz bir araçtır. Çeşitli kötü amaçlı yazılımların algılanmasını kolaylaştırmak için bir dizi Antivirüs motoru kullanır [62]. VirusTotal, uygulamaları kötü niyetli ve zararsız olarak etiketlemez. Bir uygulamanın veya yürütülebilir dosyasının özetleri verildiğinde, içerik ve çalışma zamanı hakkında yaklaşık 60 antiviral tarayıcı tarafından uygulamaya bilgi döndürülür. Bunun sonucunda dosyanın kötü amaçlı veya zararsız olarak etiketlemek için bu tür bilgileri yorumlama stratejilerine karar vermek platformun kullanıcısına kalmıştır. Uygulamaları doğrudan etiketlemek için VirusTotal' dan alınan tarama sonuçlarını yorumlamaya yarayan standart prosedürler yoktur [63].

[19] https://www.virustotal.com

2.2.4.3. Sandbox

Sandbox teknolojisi, şüpheli dosyaları güvenli bir şekilde yürütmek ve davranışını bilmek için son kullanıcı işletim ortamlarına benzeyen izole edilmiş bir test ortamıdır. Kullanıcıların üzerinde çalıştığı uygulama ve sisteme etki etmeden şüpheli veya zararlı dosyaları çalıştırmasına/açmasına olanak tanır. Araştırmalar için kontrollü bir ortam sağlayarak zararlı yazılım analizinde kapsamlı bir çözüm sağlar. Kapalı bir sistem olduğu düşünülürse sandbox ortamında giriş ve çıkış süreçleri yer almaktadır. Dışarıdan görülen ise zararlı yazılım örneğini sandbox sistemine vermek ve çıktı olarak sonuç raporunu almaktır. Sandbox sanal ortam ile iletişimi sağlayarak içerisinde bulunan çeşitli uygulama, betik (script) ve araçlar sayesinde sonuç raporu elde etmeye olanak sağlar. Ayrıca dosya sistemini, kayıtları, süreci, servisleri ve belleği izleme uygulamaları bulunmaktadır. Sandbox ortamları temelde üç bölümden oluşur:

1. Analiz süreci yönetimi,
2. Analizi yapılmak istenen zararlı/şüpheli yazılımın çalıştırıldığı sanal ortam (VMware VirtualBox vb.),
3. Sonuç raporunun elde edilerek analiz yorumunun yapılması.

Zararlı yazılım analizinde kullanılan çeşitli sandbox sistemleri bulunmaktadır. Güvenlik araştırmacıları tarafından kullanılan bazı popüler sandbox sistemleri Comodo Valkyrie, Joe Sandbox, Cuckoo Sandbox, Minibis BitBlaze, Malwr, ThreatExpert, Trend Micro olarak verilebilir [52].

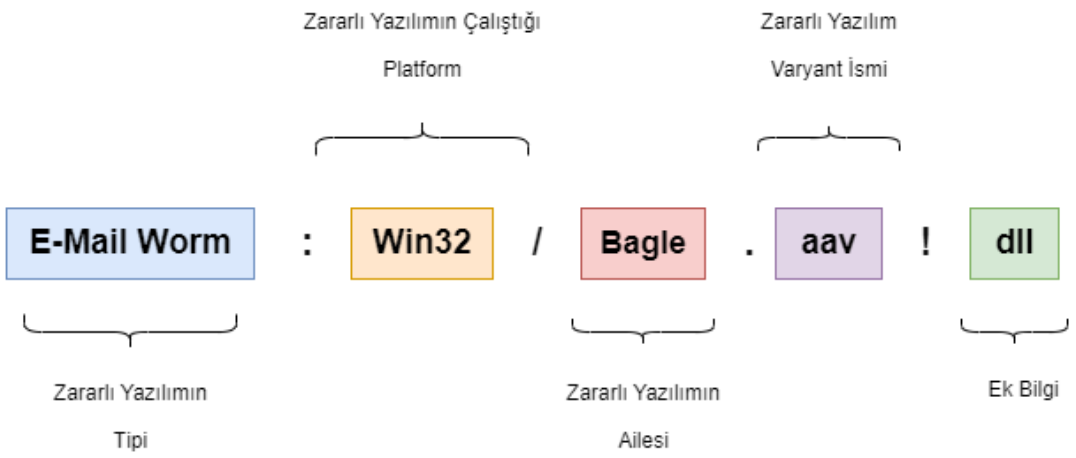
2.2.4.4. Remnux

REMnux tam adıyla Reverse Engineering Malware, zararlı yazılım analizleri için kullanılan bir Linux-Ubuntu dağıtımıdır. İçerisinde kurulumu zor olabilecek zararlı yazılım analizi araçlarını barındırmaktadır. Windows ve Linux sistemlerdeki zararlı yazılımların analizi, şüpheli PDF, DOC, XLS gibi dosyaların analizi, şüpheli dosyaların ayrılması/ayıklanması gibi işlemler gerçekleştiren birçok araç içermektedir. Şüpheli ağ trafiği için izole bir ortam sağlayan Remnux, Davranışsal Zararlı Yazılım Analizi alanında kullanılan önemli ortamlardan biridir. Ayrıca REMnux' un içerdiği tüm araçların kullanımını, ağ tabanlı zararlı yazılımların analizinin nasıl yapılacağını, şüpheli dosyaların ve belgelerin incelenmesine ilişkin açıklamalar bulunduran bir referans dokümanı bulunmaktadır [20].

[20] https://docs.remnux.org/

2.2.5. Zararlı Yazılımların İsimlendirilmesi

Zararlı yazılımların isimlendirilmesi konusunda "Computer Antivirus Research Organization (CARO)" [21] adında resmi olmayan bir grup tarafından isimlendirme şeması oluşturulmuştur. CARO, zararlı yazılımları araştırmak ve incelemek için 1990 yılında kurulmuş bir kuruluştur. Günümüzde antivirüs yazılımları çoğunlukla CARO tarafından oluşturulan şemaya benzer bir şema veya direkt olarak bu kuruluşun şemasını kullanır. CARO' nun oluşturduğu zararlı yazılım isimlendirme şemasına göre sırasıyla zararlı yazılımın tipi, zararlı yazılımın çalışabildiği platform, zararlı yazılım aile ismi, varyant ve ek bilgiler verilir. Örneğin **Şekil 5'** te görüldüğü gibi E-MailWorm:Win32/Bagle.aav!dll zararlı yazılımı solucan sınıfında Windows 32 bit işlemci platformu üzerinde çalışabilir. Zararlı yazılım Bagle ailesinden aav varyantıdır. Zararlı yazılımın aile ismi zararlı yazılımın yazarı, zararlı yazılımın sahip olduğu davranışlar, kod yapılarındaki benzerlik gibi ortak özellik bilgilerine göre verilir.



Şekil 5 Zararlı Yazılımların İsimlendirilmesi

[21] https://uia.org/

2.3. Bilgi Entropisi

Entropi, bilginin veya belirsizliğin matematiksel bir ölçüsü olarak düşünülebilir ve bir olasılık dağılımının bir fonksiyonu olarak hesaplanır. Bilgi entropisi kavramı, Claude Shannon tarafından 1948 tarihli "A Mathematical Theory of Communication" adlı makalesinde [64] tanıtıldı ve aynı zamanda Shannon entropisi olarak da anılır. Shannon'ın teorisi, üç öğeden oluşan bir veri iletişim sistemini tanımlar: bir veri kaynağı, bir iletişim kanalı ve bir alıcı. Shannon tarafından ifade edildiği şekliyle "iletişimin temel sorunu", alıcının kanal aracılığıyla aldığı sinyale bağlı olarak kaynak tarafından hangi verilerin üretildiğini tanımlayabilmesidir. Shannon, bir veri kaynağından gelen mesajları kodlamanın, sıkıştırmanın ve iletmenin çeşitli yollarını düşünmüş ve ünlü kaynak kodlama teoreminde, entropinin, kaynaktan gelen verilerin kayıpsız bir şekilde tamamen gürültüsüz bir kanala ne kadar iyi sıkıştırılabileceği konusunda mutlak bir matematiksel sınırı temsil ettiğini kanıtlamıştır. Shannon, gürültülü kanal kodlama teoreminde gürültülü kanallar için bu sonucu önemli ölçüde güçlendirmiştir.

2.4. İlgili Çalışmalar

Zararlı yazılımların tespitini ve analizini gerçekleştirmek için her şeyden önce zararlı yazılımlardan oluşan bir veriye ihtiyaç duyulmaktadır. Sistemlere bulaşan, kurum ve kuruluşlara gelen zararlı yazılım saldırılarından elde edilen örnekler olsa da yeni geliştirilen türleri bilme imkanı bir sisteme bulaşmadan önce olmamaktadır. Yeni geliştirilen veya aynı zararlı yazılım ailesinden gelip farklı bir varyant olarak kabul edilen türler bal küpü tuzak sistemler kullanılarak keşfedilebilir. Tuzak sistemlere yüklenen şüpheli dosyaların tespit ve analizi zararlı yazılımlara karşı alınabilecek önlemleri geliştirmeye imkan sağlayabilir. Sethia ve arkadaşı bal küpü üzerinden topladıkları zararlı yazılımları saldırı aldıkları protokollere göre sınıflandırmış ve en fazla SQL 2000XP, SMB, FTP protokollerinden saldırı geldiği görülmüştür [65]. Kurumsal yapılarda sıklıkla kullanılan protokollerin saldırı alması zararlı yazılımların kuruluşları hedef aldıklarını göstermektedir. Matin ve arkadaşı, Modern Honey Network kullanarak Portable Executable (PE) zararlısı toplamak için bir çerçeve (framework) önermiştir. Kullanılan bal küpü, Dionaea sensörlü Modern Honey Network' tür. Virus Total kullanılarak zararlı yazılım türlerini sınıflandırmışlardır [66]. Kyriakou ve arkadaşı yaygın kullanılan protokol etkileşimlerini kaydeden, Virüs Total' in analiz motoru kullanılarak zararlı yazılımları tespit eden ve dağıtık çalışan bir bal küpü sistemi tanıtmışlardır [67]. İlgili çalışma ile bir organizasyonun risk değerlendirmesine yardımcı olmanın yanı sıra bal küpleri ile saldırganlara

karşı gerçek zamanlı istatistikler ve analizler sunulabileceğini göstermişlerdir. Chris Moore tarafından Microsoft Windows ağına yönelik bir fidye yazılımı saldırısını tespit etmek ve saldırıları azaltmak için bal küpü ilkelerinin nasıl kullanılabileceği incelenmektedir [68]. Wang ve arkadaşları ise IoT Honeypot olarak önerdikleri IoTCMal tarafından yakalanan kötü amaçlı yazılım ikili (binary) dosyalarının benzerliğini kapsamlı analizini gerçekleştirerek en az 11 saldırgan grubu tarafından kontrol edilen 8 zararlı yazılım ailesi keşfetmişlerdir [69]. Yapılan araştırma çalışmaları zararlı yazılım elde etmek ve analiz etmek için bal küplerinin faydalı bir yöntem olduğunu göstermektedir.

Zararlı yazılımların elde edilmesinden sonraki en önemli aşama zararlı yazılımların zararlı olduğunun tespiti, çalışma şekilleri ve hedefleri doğrultusunda gerçekleştirdikleri davranışların analizinin yapılmasıdır. Zararlı yazılımların tespitinde ve analizinde kullanılan birçok farklı tespit tekniği ve analiz yöntemi mevcuttur. Tespit tekniklerinde yaygın olarak kullanılan imza tabanlı tespit tekniği, bilinen zararlı yazılımlar için hızlı ve etkili bir sonuç vermektedir. Ayrıca bu teknik, çoğu Anti-virüs uygulamasının temelini oluşturmaktadır. İmza tabanlı tespit tekniği sistemlerde yaygın olarak bulunan zararlı yazılımlar için hızlı ve etkili sonuçlar verse de çok biçimli ve değiştirilmiş kodlara sahip zararlı yazılımlara karşı zayıftır. Buna karşılık, davranış tabanlı tespit tekniği her türdeki zararlı yazılım için iyi sonuçlar verir. Ancak, zararlı yazılım izlenen herhangi bir yazılımın normal davranışı olarak ele alındığında yetersiz kalmaktadır. Spesifikasyona Tabanlı (Spesification-based) tekniği ise, incelenen bir programın kötü niyetli olup olmadığına karar vermek için neyin geçerli davranış olduğuna dair bazı spesifikasyonlardan veya kural setinden yararlanır. Kullanıcılara yeterli koruma ve aynı zamanda iyi bir kullanıcı deneyimi sağlayan imza tabanlı ve davranış tabanlı yaklaşımları kapsadığı için verimli bir teknik olduğu sonucuna varılabilir [70].

İnternete bağlı her cihaz siber saldırıya karşı savunmasızdır, onları herhangi bir saldırıdan korumak için Anomali tabanlı algılama, spesifikasyon tabanlı algılama ve imza tabanlı algılama gibi birden fazla teknik sunulur, ancak siber güvenlik önlemlerinde karşılaşılan tehditler özellikle zararlı yazılım alanında zamanla gelişmiştir. Tipik olarak, zararlı yazılım, antivirüs yazılımı tarafından algılanabilen dosya sistemini temel alır. Fakat son zamanlarda geliştirilmiş dosyasız zararlı yazılımlar (fileless malware) imza tabanlı herhangi bir algılamayı atlatabilir. Bilgisayar sistemini güvenli hale getirmek için zararlı yazılımın sürekli analizi gereklidir, böylece kötü amaçlı yazılım kolayca tespit edilebilir. Himanshu Anand çalışmasında, LR (Logistic Regression), KNN (K-Nearest Neighbour), DT (Decision Tree) ve SVM (Support

Vector Machine) olmak üzere 4 farklı makine öğrenimi algoritması kullanmış ve 10 farklı dosyasız zararlı yazılım içeren veri setine algoritmayı uygulamıştır [71]. Gizleme stratejilerinin son yıllarda önemli ölçüde gelişmesiyle antivirüs yazılımlarının kullandığı imza tabanlı tespiti atlatan polimorfik ve metamorfik metodolojilere dayalı zararlı yazılımlar geliştirilmiştir. Dosyasız zararlı yazılım gibi yalnızca geçici bellekte yerleşik olan zararlı yazılım çeşidi disk analizini anlamsız hale getirmiştir. H´ector D. Men´endez tarafından yapılan incelemede, zararlı yazılımları algılama veya analiz tekniklerinden korumak için tanıtılan farklı gizleme stratejilerinin tarihsel bir arka plan verilmiştir. Statik ve dinamik analizlerin yanı sıra hem saldırganlar hem de savunanlar açısından makine öğrenimine dayalı yeni stratejiler geliştirmiştir [72]. İmza tabanlı tespit tekniği, sıfırıncı gün saldırılarını tespit etmede yetersiz olduğunu savunan C.P. Patidar ve Harshita Khandelwal makine öğrenimi ve yapay zeka kullanarak sıfırıncı gün zararlı yazılımlarının tespitine dayalı bir çözüm önermişlerdir [73].

Yapay zeka teknolojilerinin gelişmesiyle zararlı yazılım tespitinde yapay zeka kullanan çalışmalar da artmaktadır. Mevcut kötü amaçlı yazılım algılama teknolojilerinin verimliliğini artırmanın yollarının ayrıntılı bir incelemesini sunan Faruk ve arkadaşları, yapay zekanın kötü amaçlı yazılımdan koruma sistemlerinin geliştirilmesi için umut verici olduğunu göstermişlerdir [74]. Karşılaştırılan imza tabanlı Antivirüslere karşı istatistiksel olarak daha üstün ve daha etkili sonuçlar verdiği gösterilmiştir. Rani ve arkadaşları da anormallik tespitinde K-NN algoritması uygulanmış ve kötü amaçlı yazılım sınıflandırıcılarının uygulanmasıyla ilgili zorlukları tartışmıştır [75]. Birden fazla Machine Learning yöntemi kullanarak geliştirilen saldırı tespit sistemlerinin kötü amaçlı yazılıma karşı daha fazla güvenlik sağladığı görülmüştür [76]. Assegie ise bir Kaggle veri kümesi üzerinde K-NN modeli ile uygulama programlama arayüzü (API) kullanarak sınıflandırma ve analizine dayalı imza tabanlı bir kötü amaçlı yazılım tespit modelini %98,17 doğrulukla göstermişlerdir. Ayrıca, daha yüksek sınıflandırma doğruluğu elde etmek adına en uygun K değerini bulmak için Grid Arama (Grid Search) yöntemi kullanılmıştır [77].

Anti-virüs yazılımları bilinen zararlı yazılım tehditlerinin engellenmesinde etkili olmasına rağmen taşınabilirlik, kaynak tüketimi ve maliyet problemleri oluşturur. Her bilgisayarda ayrı bir antivirüs uygulaması kurulması gerekir ve ayrıca antivirüs uygulamalarının güncellenmesi ve lisans yenilenmesi gibi periyodik ihtiyaçları vardır. Akshay Chavan ve arkadaşları, herhangi bir taşınabilir aygıtta bulunabilen ve bilgisayar sistemi bağlandığında imza tabanlı ve sezgisel tabanlı tespit taraması yapabilen bir anti-virüs tasarlamışlardır [51]. Taşınabilir antivirüs

uygulamasının statik veri tabanı kullanması ve her gün yeni oluşturulan binlerce virüsün ortaya çıkmasından kaynaklanan riskler içermektedir. İnternet üzerinden alınan dinamik ve sürekli güncellenen bir veri tabanına sahip olmak son derece önemli hale gelmiştir. Bir başka antivirüs tasarlayan çalışma Botacin ve arkadaşlarına aittir. Önerilen HEAVEN modeli ile Intel x86/x86-64 ve MS Windows için Anti-virüslerin performansını ve etkinliğini iyileştirmek için çerçeve (framework) tanıtmışlardır. HEAVEN modeli farklı kategorilerden 10.000 zararlı yazılım ve 1.000 iyi huylu yazılım örneğinden oluşan bir veri seti ile kötü amaçlı yazılımların nerdeyse tamamını tespit ettiklerini göstermişlerdir [61].

Anti-virüs, işletim sistemini virüslerden, casus yazılımlardan, hacker saldırılarından ve değerli kişisel verileri çalmak veya yetkisiz bilgisayar yönetimi için diğer yetkisiz erişimlerden korumak için tasarlanmış özel bir programdır. Antivirüs programları kullanılan koruma teknolojilerine, ürünün işlevselliğine ve hedef platforma göre sınıflandırılabilir. Anti-virüs dedektörleri, antivirüs bulaşmış dosyaları bulur ve tespit edilirse bir mesaj görüntülenir. Rohith ve arkadaşı antivirüs uygulamalarının şüpheli bir dosyayı tespiti, zararlı yazılımların yayılmasının engellenmesi ve şüpheli dosyaların kimliğinin algılanmasını sağlayan yöntemleri çalışmalarında paylaşmışlardır [5]. Antivirüs tarafından kullanılan zararlı yazılım tespit ve önleme teknikleri üzerine kapsamlı bir çalışma yapan Cheerala Rohith ve Gagandeep Kaur antivirüs yazılımlarının çalışma mekanizmasını ve bazı popüler antivirüs yazılımlarının (Kaspersky, Norton, McAfee, Antiviral Toolkit Pro vb.) da çalışma mantığından bahsetmiştir [78]. İncelenen ticari antivirüslere bağlı olarak zararlı yazılım tespiti %0 ile %99,11 arasında değişmekte ve uygun bir antivirüs seçimin siber saldırılara karşı korunmada önemli bir rol oynamaktadır. Sidney M. L. Lima ve arkadaşları tarafından yapılan çalışmada incelenen antivirüsler ortalama olarak, değerlendirilen zararlı yazılımların %54,82' sini tespit edebildiler. Çalışmada VirusTotal platformu kullanılmış ve değerlendirilen antivirüslerin yaklaşık %17' si, zararlı örneklerin hiçbirini tespit edememiştir. Son zamanlarda yapay zeka ticari antivirüs üreticileri için iyi bir alternatif haline gelmiştir. Yapay zeka, öğrenme aşamasında binlerce dosyayı analiz etme ve bunlardan özellik çıkarma yeteneğine sahiptir. Öğrendikten sonra, etiketlenmemiş yeni bir dosya ile karşılaşıldığında, yakalananlarla karşılaştırılarak zararlı yazılım olarak doğru bir şekilde sınıflandırılması mümkün olacaktır. Bir antivirüs, yapay zeka kullanarak binlerce şüpheli dosya özelliğinin analizini büyük ölçekte ve gerçek zamanlı olarak otomatik hale getirebilir. Sidney M. L. Lima ve arkadaşları tarafından önerilen metodolojide yürütülebilir dosyalardan çıkarılan özellikler yapay sinir ağlarının girdi nitelikleri olarak

alınmıştır. Önerilen metodolojiyi doğrulamak için toplamda 6272 yürütülebilir dosya kullanılmıştır. Önerilen antivirüs, yalnızca 0,07 saniyelik ortalama yanıt süresiyle birlikte, iyi huylu ve zararlı yürütülebilir dosyalar arasındaki ayrımda ortalama %98,32' lik bir performans elde etmiştir. Son teknoloji antivirüsler ile karşılaştırıldığında istatistiksel olarak daha üstün ve daha etkili olduğu sonucunu elde etmişlerdir [79].

Zararlı yazılımlardan oluşan büyük veri setlerini veya örnekleri statik olarak analiz ederken, paketlenmiş ve şifrelenmiş zararlı yazılım özniteliklerinin ve işlevlerinin tanımlanmasını otomatikleştirmek önemli bir zorluk teşkil eder. Entropi analizi, çalıştırılabilir zararlı yazılımlardaki istatistiksel değişkenliği inceleyerek analistlerin paketlenmiş ve şifrelenmiş örnekleri hızlı ve verimli bir şekilde tanımlamasını sağlar. Robert Lyda ve James Hamrock araştırmalarında, Bintropy aracını ve metodolojisini önermişlerdir. Önde gelen bir antivirüs satıcısından alınan 21.576 PE biçimli zararlı dosyalar Bintropy aracı kullanılarak şifreleme ve paketleme ile ilgili entropi eğilimleri araştırılmıştır [80]. Munkhbayar Bat-Erdene ve arkadaşları bilinmeyen paketleyicileri saptamak için paketlenmiş yürütülebilir dosyalarda paketten çıkarma işlemlerinin entropi değerlerini incelemiştir. Zararlı veya zararsız programlar olup olmadığına bakılmaksızın, bilinmeyen paketli yürütülebilir dosyaların paketleme algoritmalarını sınıflandırmak için bir yöntem önermişlerdir [81].

Kyungroul Lee ve arkadaşları fidye yazılımlarını tespit etmek ve önlemek için dosya ve davranış tabanlı tespit yöntemlerini araştırmışlardır. Şifrelenmiş dosyanın tekdüzelik özelliğini ölçmek için entropi tekniğini kullanarak makine öğrenimi, virüslü dosyaların dosya entropi analizine dayalı olarak sınıflandırılmasını sağlamışlardır. Gerçekleştirilen analiz sonuçları, önerilen yöntemin mevcut tespit yöntemlerine kıyasla düşük yanlış pozitif ve yanlış negatif oranları ile yüksek bir tespit oranı sağladığını doğrulamaktadır [82]. Kyoung Soo Han ve arkadaşları ikili dosyaları gri ölçekli bitmap görüntülere dönüştüren ve entropi grafikleri oluşturan bir zararlı yazılım görselleştirme yöntemi önermişlerdir. Ayrıca, görselleştirilmiş görüntüleri kullanarak zararlı yazılımların benzerliklerini hesaplamış ve zararlı yazılım ailelerini sınıflandırmışlardır [83]. Joon-Young Paik ve arkadaşları, yapısal entropi (structural entropy) kavramını içeren bir özellik mühendisliği tekniği önermektedir. Bu teknik, her bölümün her bayt değeri için 256 entropi değeri olarak temsil edilir. Ayrıca yüksek seviyeli özellikler çıkarmak için, bir evrişimli sinir ağı (CNN) mimarisi kullanmışlardır [84].

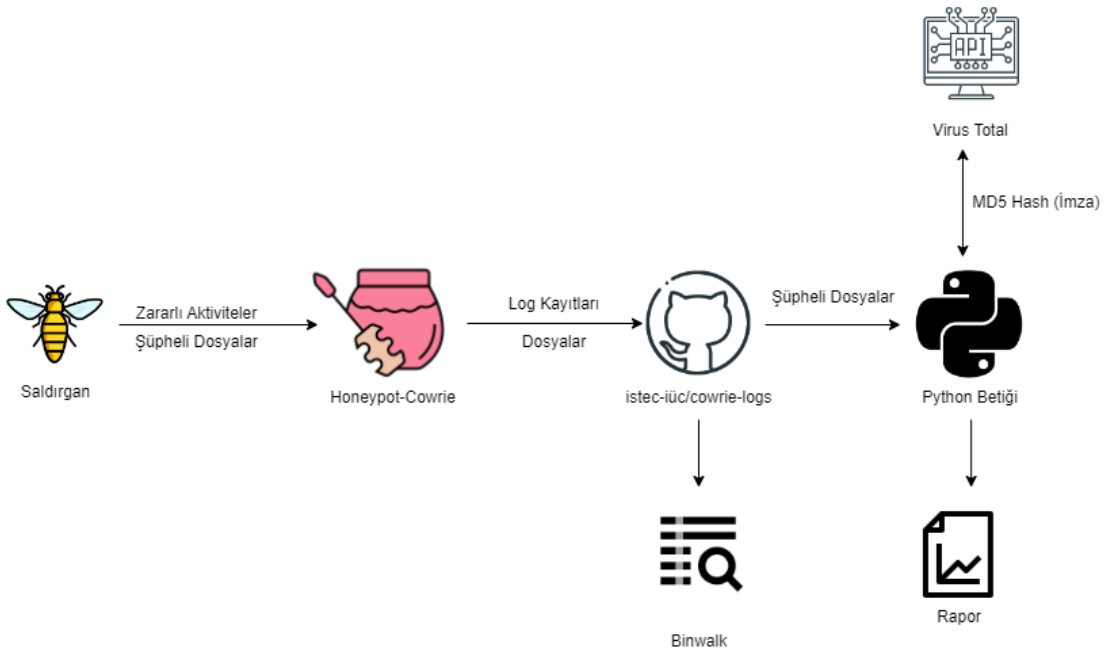
Bu çalışma kapsamında Google Cloud üzerine kurulmuş olan Cowrie bal küpü sistemi ile toplanan şüpheli dosyaların değerlendirilme çalışması yapılmıştır. Çalışma kapsamında imza tabanlı tespit tekniği kullanılmış olup Cowrie bal küpü sistemi üzerine gelen saldırıları VirusTotal aracılığıyla anlık değerlendirebilecek bir program betiği geliştirilmiştir. VirusTotal üzerinde çalışan Anti-virüs uygulamalarından gelen cevaplar ile şüpheli dosyaların zararlı veya zararsız dosya ayrımı yapılmıştır. Kötü amaçlı yazılımlar sınıflandırılarak istatistikler belirlenmiş ve antivirüs uygulamalarının performansları değerlendirilmiştir. VirusTotal üzerinde çalışan antivirüs uygulamaları tarafından cevap dönülmeyen şüpheli dosyaların statik analiz kapsamında entropi analizi ve kaynak kod okuma yöntemi gerçekleştirilmiştir.

3. YÖNTEM

Zararlı yazılımlar sistemlerin çalışmasını sekteye uğratan veya engelleyen, saldırı amaçlarına göre tasarlanan ve analiz edilen yazılımlardır. NIST (National Institute of Standards and Technology) tanımına göre ise zararlı yazılımlar kurbanın verilerinin, uygulamalarının veya işletim sisteminin gizliliğini, bütünlüğünü veya kullanılabilirliğini tehlikeye atmak amacıyla başka bir programa gizlice eklenen bir programlar şeklinde ifade edilebilir [49]. Zararlı yazılımlar amaç ve hedeflerine göre farklı türlere ayrılmaktadırlar. Her bir tür bir aile olarak sınıflandırılmakta ve her geçen gün yeni çıkan bu ailelere ait farlı varyantlar tespit edilerek sınıflandırmaya dahil edilirler. Zararlı yazılımları tespit etmek ve hangi aile sınıfına ait olduğunu belirlemek, zararlı yazılımlar için geliştirilmiş çeşitli tespit teknikleri ve analiz yöntemleri ile yapılabilmektedir. Fakat, yeni zararlı yazılım varyantları ve ailelerin oluşması ile bazı yöntemler tespit ve analiz için yeterli olmamaktadır. Tespiti yapılamayan zararlı yazılımlar sistemler ve kullanıcılar için büyük riskler taşımaktadır.

Bilinen zararlı yazılımları elde etmek ve tespitini gerçekleştirmek kolay olsa da henüz bilinmeyen bir zararlı yazılıma veya o yazılım ailesine ait farklı bir varyanta ulaşmanın yollarından biri tuzak sistemler kullanmaktır. Bilgi sistemlerine yetkisiz erişen saldırganlar ya da kullanıcılar hakkında bilgi toplamaya yarayan tuzak sistemler, daha önce görülmeyen saldırı çeşitlerini gerçek sistemin zarar görmeden elde edilmesini sağlarlar. Saldırganlar tarafından tuzak sisteme yüklenen, ağda yayılması hedeflenen veya sisteme zarar vermesi beklenen zararlı yazılımların incelenmesi, tespitinin ve analizinin yapılması zararlı yazılımlara karşı önlemler almaya imkan sağlamaktadır. Zararlı yazılımların tespitinde yaygın olarak kullanılan İmza tabanlı tespit tekniği, antivirüs uygulamalarının çalışma mekanizmasının temelidir. Bugüne kadar geliştirilmiş bilinen zararlı yazılımların imzaları sistemdeki zararlı yazılım imzaları ile karşılaştırılarak zararlı yazılımın tespiti sağlanır. Fakat imza tabanlı tespit tekniğinin ve anti-virüs uygulamasının tespitinin başarı ile sonuçlanması kötü amaçlı yazılımın önceden biliniyor olmasına bağlıdır. Yeni geliştirilen ve henüz imzası bilinmeyen kötü amaçlı yazılımların imza tabanlı tespit tekniği ile keşfedilmesi mümkün değildir.

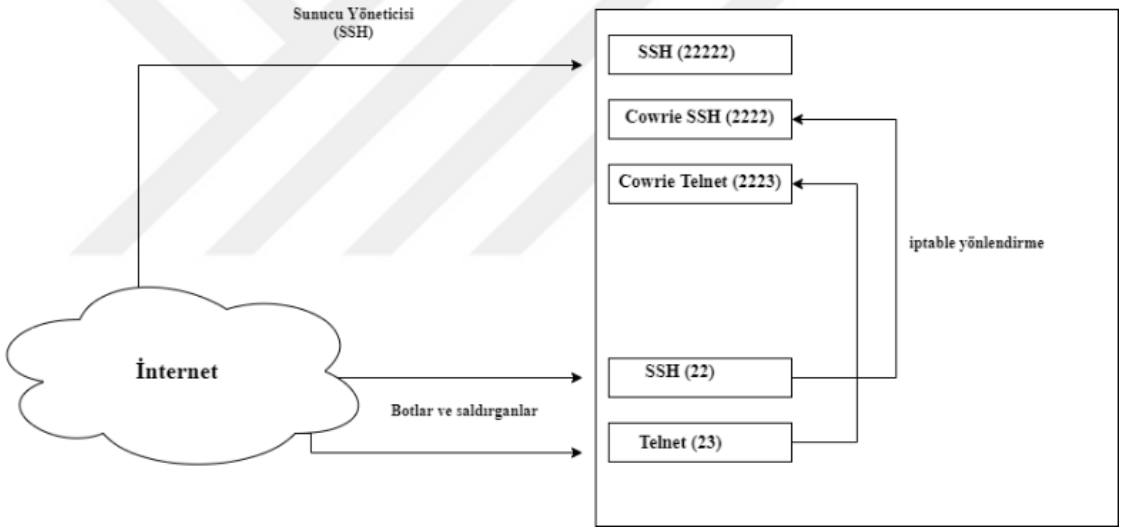
Kötü amaçlı yazılım tespiti ile ilgili yapılan akademik çalışmalar son yıllarda artış göstermiştir. Genellikle çalışmalarda imza tabanlı tespit tekniği yaygın olarak kullanılırken zamanla araştırmacılar davranış tabanlı, sezgisel tabanlı ve model denetleme tabanlı tespit teknikleri de kullanmışlardır. Ayrıca, son birkaç yıl içinde derin öğrenme ve IoT tabanlı tespit teknikleri de kullanılmaya başlanmıştır. Bu tez kapsamında SSH ve Telnet protokollerini simüle eden Cowrie tuzak sistemi kullanılarak gerçek saldırılardan elde edilen şüpheli dosyaların imza tabanlı tespit tekniği ile tespitinin yapılması ve yaygın kullanılan tekniğin yeterliliğinin tartışılması amaçlanmıştır. Google Cloud Platform üzerine kurulmuş olan tuzak sistem Cowrie üzerine gelen gerçek zamanlı saldırılarda kayıt altına alınan şüpheli dosyaların VirusTotal kullanılarak tespiti gerçekleştirilmiştir. Toplamda 47 gün bekletilen tuzak sistem, 3.626.394 adet saldırı almış ve tümü kayıtlarda tutulmuştur. Cowrie tuzak sisteminden elde edilen 52 zararlı dosyayı otomatik olarak VirusTotal üzerindeki Antivirüs uygulamaları ile değerlendirmeye yarayan Python dilinde bir program betiği geliştirilmiştir. VirusTotal üzerinde bulunan yaklaşık 64 antivirüs uygulamalarından elde edilen sonuçlar karşılaştırılmış ve değerlendirilmiştir. Ayrıca VirusTotal ile tespit edilemeyen 2 şüpheli dosyanın statik analizi gerçekleştirilmiş olup Binwalk aracı ile entropi analizi yapılmıştır. Çalışmaya ait akış diyagramı **Şekil 6**' da verilmiştir.



Şekil 6 Yapılan Çalışmanın Akış Diyagramı

3.1. Cowrie: SSH ve Telnet Tuzak Sistemi

Cowrie [85], kaba kuvvet (brute-force) saldırılarını ve saldırganın kabuk etkileşimini (shell interaction) kaydedebilen düşük-orta etkileşimli SSH ve Telnet bal küpü tuzak sistemidir. Michel Oosterhof tarafından geliştirilmiştir ve açık kaynaklı bir projedir. Cowrie bal küpünün tercih edilmesinin nedeni açık kaynaklı olup kolay bir kullanım sağlaması ve SSH, Telnet protokolleri üzerinden gelen saldırıları gözlemlememize ve değerlendirmemize yardımcı olmasıdır. Cowrie, Ssh ve Telnet protokollerini simüle eder, bu da saldırganın bu protokolleri kullanan gerçek bir servise saldırdığını düşündüğü anlamına gelir. Saldırgan sistemde üzerinde oturum açmak için doğru verilerden oluşan kullanıcı adı ve şifre girdiğinde, sahte bir sisteme giriş yapmış olurlar. **Şekil 7'**de Cowrie çalışma mantığı grafik üzerinde anlatılmıştır.



Şekil 7 Cowrie Çalışma Mantığı

Bu çalışmada kullanılan bal küpü sistemi Cowrie, Google Cloud Platformu[22] üzerine kurulmuştur. Cowrie bal küpünden elde edilen çıktıların analiz aşamasında farklı araçları kullanmasına imkan sağlaması amacıyla Json formatında log dosyalarında ve Mysql veri tabanında kayıt altına alınması düşünülmüştür. Bu kayıtların görsel analizi Json log dosyaları kullanılarak ELK Stack[23] aracı ile yapılmıştır.

[22] https://cloud.google.com/why-google-cloud
[23] https://www.elastic.co/what-is/elk-stack

Saldırganın gerçek bir SSH bağlantı noktasında olduğunu düşünmesi için 22 numaralı bağlantı noktası (varsayılan SSH bağlantı noktası) ‘22222’ ile değiştirilmiştir. Cowrie, 2222 numaralı bağlantı noktasında çalışır ve iptable kullanılarak 22 portuna gelen trafik 2222 noktasına yönlendirilir. Yani saldırganlar 22 numaralı porta bağlandığında aslında yönlendirme nedeniyle bal küpünün çalıştığı 2222 numaralı porta bağlanmış olacaktır. 23 numaralı portta çalışan Telnet iptable kullanılarak 2223 noktasına yönlendirilir ve saldırgan 23 portuna bağlanmak istediğinde 2223 numaralı porta bağlanmış olur.

Varsayılan olarak, cowrie root kullanıcının şifresi için belirli kurallara sahiptir. Bal küpünü daha gerçekçi hale getirmek için varsayılan şifreler değiştirilmiştir. **Tablo 4** değiştirilen varsayılan kullanıcı adı ve şifrelerini göstermektedir.

Varsayılan Kullanıcı Adı ve Şifreler	**Değiştirilmiş Kullanıcı Adı ve Şifreler**
root : x : ! root	root : x : password
root : x : ! 123456	root : x : admin
root : x : ! /honeypot / i	root : x : 123456
root : x : *	root : x : default
tomcat : x : *	tomcat : x : tomcat
oracle : x : *	oracle : x : oracle

Tablo 4 Cowrie Kullanıcı Adı ve Şifre Değişimi

Bal küpü içerisinde gerçekleşen her şey analiz için kaydedilir. Üç tür log kaydı yapılabilir. Bunlar ‘Connections’, ‘Shell Interaction’ ve ‘File Transfer’ başlıkları altında özetlenebilir. Sırasıyla, tuzak sisteme yapılan kaba kuvvet saldırılarıyla ilişkili tüm veriler kaydedilir. Başarılı bir oturum açtıktan sonra, tüm kabuk oturumunun komutları, çıktıları ve zaman damgaları kaydedilir. Bir saldırganın wget, curl, scp veya sftp kullanarak aktarmaya çalıştığı tüm dosyalar kaydedilir. Bu tezin amacı tuzak sisteme aktarılan şüpheli dosyaları imza tabanlı tespit yöntemi ile zararlı veya zararsız olarak belirlemek ve sınıflandırmaktır. Ayrıca antivirüs uygulamalarının tespit ve sınıflandırmada ne kadar iyi sonuç verdiği karşılaştırılmış ve değerlendirilmiştir.

3.2. Deneysel Ortamın Hazırlanması

Zararlı yazılımlar saldırganların sistemin erişilebilirliğini, bütünlüğünü ve gizliliğini bozmaya yönelik kullandığı yöntemlerdendir. Sistemlere zararlı yazılımlar aracılığı ile yapılan saldırıların tespiti ve analizi önlem almak ve güvenliği sağlamak adına önemlidir. Kötü amaçlı yazılımları yakalamanın ve analiz etmenin ardındaki motivasyon, zararlı yazılımın mekanizmasını ve işletim sistemiyle nasıl etkileşime girdiğini incelemektir. Ağ üzerinden ne tür faaliyetler gerçekleştiğini anlayarak insanları ve çeşitli kuruluşları bu tür zararlı yazılım saldırılarından korumak için bir güvenlik mekanizması geliştirebilir. Bu çalışmada saldırganları çekmek ve onları gerçek bir sisteme erişim kazandıklarını düşündürmek için tuzak görevi gören bir araç kullanılmış ve bu tuzak sisteme yapılan saldırılardan elde edilen dosyaların zararlı olup olmadığı değerlendirilmiştir. Yapılan değerlendirme sonucunda antivirüs uygulamalarının da karşılaştırılması yapılmıştır.

Tuzak sistemin yapılandırıldığı ortam Google Cloud Platform[24] olarak tercih edilmiştir. Google Cloud Platform, Google firmasının sunucu altyapı hizmetlerini kullanıcıya açtığı, sanal makineler oluşturmaya ve çalıştırmaya olanak tanıyan bir bulut bilgi platformudur. Google Cloud Platform üzerinde kurulan sanal makineye bir Dış (External) IP adresi vermesi ve makineye ait Firewall kurallarını kolayca değiştirmeye olanak sağladığı için tercih edilmiştir. Google Cloud Platform üzerinde 2 CPU, 4 GB RAM özelliklerine ve Ubuntu 18.04 LTS işletim sistemine sahip bir sanal makine yapılandırılmıştır. Bu sanal makine için Amerika bölgesi lokasyon olarak tercih edilmiştir. Firewall kuralları güncellenerek gelen tüm ağ trafiğine (ingress) izin verilmiştir. Oluşturulan sanal makine üzerine Cowrie tuzak sistemi kurulmuştur. Cowrie[25], kaba kuvvet (brute-force) saldırılarını ve saldırganın kabuk etkileşimini (shell interaction) kaydedebilen düşük-orta etkileşimli SSH ve Telnet tuzak sistemidir.

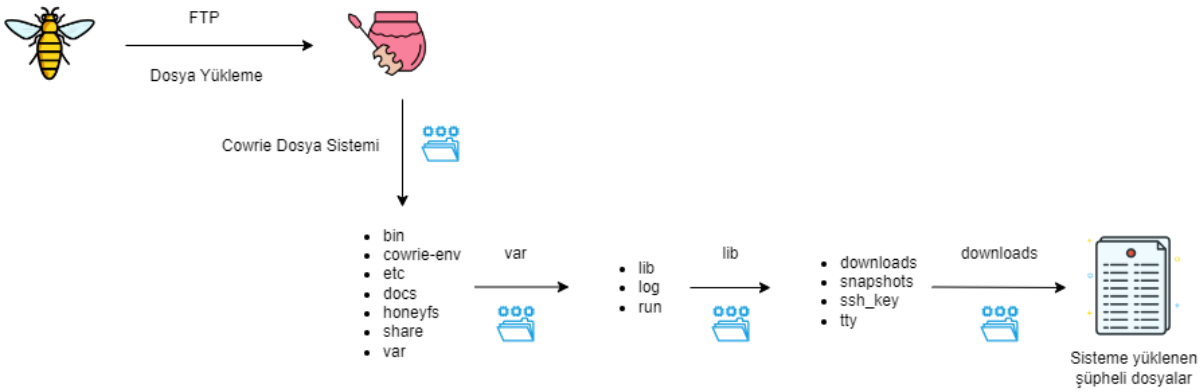
Google Cloud Platform üzerindeki sanal makineye kurulan Cowrie bal küpü tuzak sisteminin saldırıya açılmasından itibaren olan süre ile incelenmesi için seçilen son tarih arasındaki kayıtlar baz alınarak sonuçlar oluşturulmuştur. Buna göre toplamda 47 gün bekletilen bal küpü sistemi, 3.626.394 adet saldırı almış ve bu saldırıların tümü log kayıtlarında tutulmuştur. Saldırılardan elde edilen dosyalar da ayrıca bal küpü sisteminin içerisine kaydolmuştur [85].

[24] https://cloud.google.com/why-google-cloud
[25] https://cowrie.readthedocs.io/en/latest/

Saldırganların tüm aktivite kayıtları ve sisteme yükledikleri şüpheli dosyalar Github üzerinde paylaşılmıştır[26].

3.3. Kötü Amaçlı Yazılımların Toplanması

Cowrie bal küpü SSH ve Telnet protokollerini simüle ederek uzaktan bağlantı yoluyla sistem içerisinde saldırganın birçok aktivite yapmasına izin veren bir tuzak sistemdir. Sistem içerisinde saldırganın çalıştırabileceği ve sisteme dosya yüklemesine veya sistemden dosya çekmesine izin veren protokol FTP (File Transfer Protokol), saldırganlar tarafından kullanılmış olup saldırılardan 52 adet şüpheli dosya elde edilmiştir. **Şekil 8'** de tuzak sistem Cowrie' nin dosya yapısı ve saldırgan tarafından sisteme yüklenen dosyaların nerede tutulduğu gösterilmektedir.



Şekil 8 Cowrie Dosya Yapısı ve Şüpheli Dosyaların Toplanması

Saldırganların sistem içerisinde gerçekleştirdiği diğer aktiviteler **var-log** klasörü altında kaydedilirken yüklenen dosyalar için ayrı bir dizin kullanılmaktadır. Cowrie bal küpüne saldırgan tarafından FTP kullanılarak yüklenen dosyalar sırasıyla **var-lib-downloads** klasörlerini takip ederek kaydolur. Buraya kaydedilen 50 adet şüpheli dosya incelenmek üzere tuzak sistem içerisinden alınmıştır. Bu çalışma kapsamında yüklenen dosyaların zararlı olup olmadığı, imza tabanlı tespit tekniği ile kontrol sağlayan VirusTotal üzerinde bulunan anti-virüs uygulamaları ile incelenmiştir. 64 ayrı anti-virüs uygulaması tarafından incelenen dosyaların otomatik bir şekilde değerlendirilmeye alınması için Python dili ile yazılmış bir program betiği

[26] https://github.com/istec-iuc/cowrie-logs

geliştirilmiştir. Bunun sonucunda çeşitli anti-virüs uygulamalarından alınan cevaplar karşılaştırılarak değerlendirilmiş ve imza tabanlı tespit tekniğinin yeterliliği tartışılmıştır.

3.4. Kötü Amaçlı Yazılımların Değerlendirilmesi

Zararlı yazılımların tespitinde ve analizinde kullanılan farklı yöntemler bulunmaktadır. Bu yöntemlerden en çok tercih edilen ve hızlı sonuç veren teknik imza tabanlı tespit tekniğidir. imza tabanlı tespit tekniğinde ücretsiz hizmet imkanı sunan VirusTotal, Trendmicro, URLVoid gibi araçlar kullanılabilmektedir.

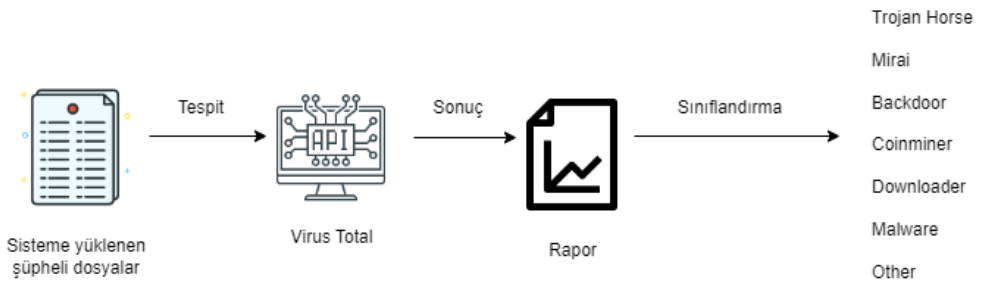
Tez çalışmamızda kullandığımız VirusTotal, şüpheli dosyaları, imzaları (hash) veya URL' leri analiz etmek için masaüstüne indirilebilen veya çevrimiçi API bulunduran ücretsiz bir araçtır. Çeşitli kötü amaçlı yazılımların algılanmasını kolaylaştırmak için bir dizi anti-virüs motoru kullanır. VirusTotal, uygulamaları zararlı ve zararsız olarak etiketlemez [62]. Bir uygulamanın veya yürütülebilir dosyasının imzası (hash) verildiğinde, içerik ve çalışma zamanı hakkında yaklaşık 60 antiviral tarayıcı tarafından uygulamaya bilgi döndürülür. Bunun sonucunda dosyayı kötü amaçlı veya zararsız olarak etiketlemek için bilgileri yorumlama stratejilerine karar vermek platformun kullanıcısına kalmıştır. Uygulamaları doğrudan etiketlemek için VirusTotal' dan alınan tarama sonuçlarını yorumlamaya yarayan standart prosedürler yoktur [63]. Bu nedenle anti-virüs uygulamalarından gelen yanıtları değerlendiren oylama bazlı (zararlı olarak işaretleyen kaç anti virüs uygulaması var) ve cevap bazlı (herhangi bir anti virüs zararlı olarak işaretlemiş mi) olarak raporlama ve değerlendirme çalışmaları yapılmıştır.

Bu çalışmada VirusTotal yardımı ile imza tabanlı tespit tekniği kullanılarak zararlı dosyalar belirlenmiş ve antivirüs uygulamalarının cevapları değerlendirilmiştir. 64 farklı antivirüs uygulamasının verdiği sonuçlar bu tez kapsamında üç ayrı kategoride sınıflandırılmıştır. **Tablo 5**' te gösterildiği gibi antivirüs uygulamalarının yüklenen şüpheli dosya için cevap döndüğü üç durum mevcuttur. Pozitif durumlar antivirüs uygulamasının veri tabanında dosyanın imzasının bulunduğu ve zararlı yazılım olduğunun tespit edildiği anlamını taşımaktadır. Negatif durumlar antivirüs uygulamasının veri tabanında dosyanın imzası bulunamadığını ve şüpheli dosyanın zararsız olduğunu tespit ettiğini göstermektedir. Son olarak antivirüs uygulamasının veri tabanında dosyanın imzası bulunamamış ve gereken sürede cevap verilememişse "Cevap Yok" dönüşü sağlanmaktadır.

Tablo 5 Anti-virüs uygulamalarının şüpheli dosyalara verdiği cevapların anlamları

Durum	Anlam
Pozitif	Antivirüs uygulamasının veri tabanında dosyanın imzası bulunmuş ve zararlı yazılım olduğu tespit edilmiştir.
Negatif	Antivirüs uygulamasının veri tabanında dosyanın imzası bulunamamış ve zararlı yazılım olmadığı tespit edilmiştir.
Cevap Yok	Antivirüs uygulamasının veri tabanında dosyanın imzası bulunamamış ve gereken sürede cevap verememiştir.

Zararlı yazılım sınıflandırması, bir kötü amaçlı yazılım örneğini belirli bir kötücül amaçlı yazılım ailesine atama işlemidir. Bir aile içindeki zararlı yazılımlar, algılama ve sınıflandırma için imzalar oluşturmak üzere kullanılabilecek benzer özellikleri paylaşır. VirusTotal üzerindeki anti-virüs uygulamalarının yaptığı sınıflandırma temel alınarak bu tez kapsamında 7 başlığa ayrılmıştır. Tez kapsamında toplanan zararlı yazılımları Mirai, Trojan Horse, Backdoor, Downloader, CoinMiner, Malware ve Other olarak sınıflandırılması **Şekil 9**' da gösterilmiştir. Other başlığı altında sınıfı tam olarak belirlenememiş zararlı yazılım varyantları mevcuttur. Ekler bölümünde verilen **Tablo Ek 1-7** arasında tüm varyantların ait olduğu sınıflar belirlenmiş ve gösterilmiştir. Örneğin Linux.Trojan.Agent.WM6SLY zararlısı isminden de anlaşılacağı üzere Trojan Horse ailesine aittir.

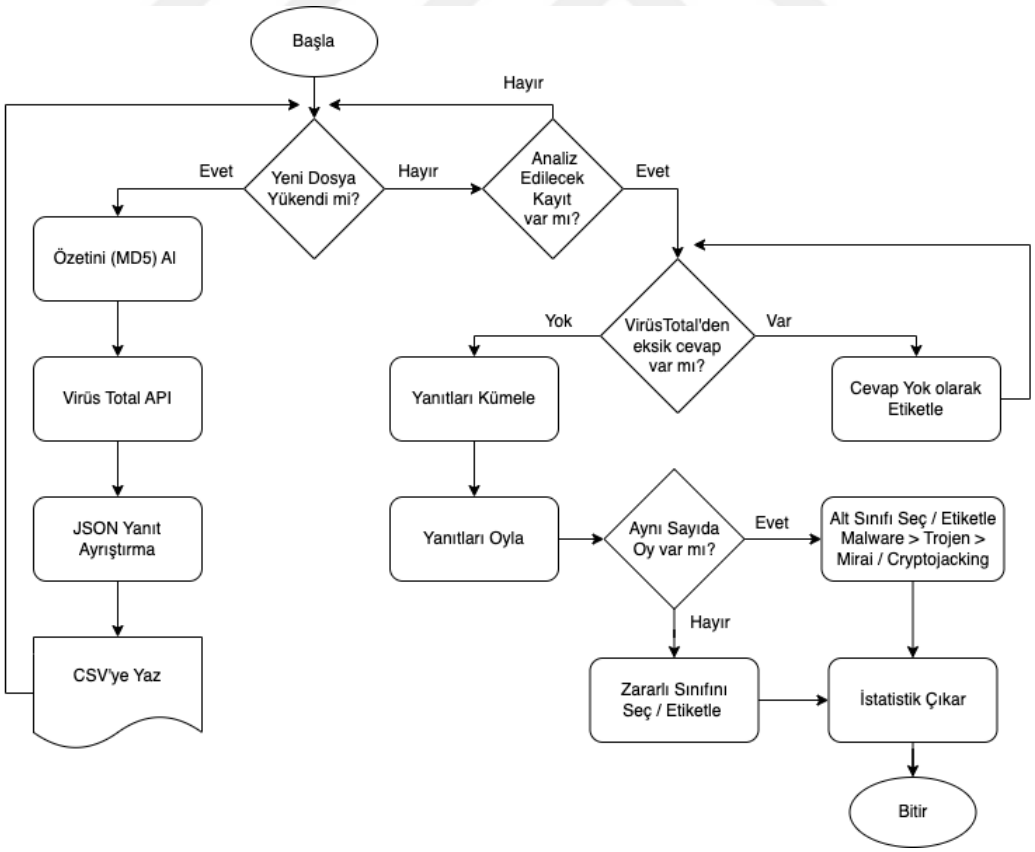


Şekil 9 Zararlı yazılımların sınıflandırılması

3.4.1. Kötü Amaçlı Yazılımlarda İmza Tabanlı Tespit Tekniğinin Kullanılması

Geliştirilen uygulama Cowrie bal küpü üzerine yüklenen şüpheli dosyaların özetlerini alarak VirüsTotal' e API üzerinden gelen yanıtları **Şekil 10'**da görüldüğü gibi CSV formatında kaydetmektedir. Ayrıca VirusTotal API' dan dönen cevap JSON nesnesinde bulunmayan antivirüs uygulamalarını "cevap yok" olarak etiketlenmiştir.

Antivirüs uygulamaları şüpheli dosyanın zararlı olup olmadığının yanında bir de zararlı yazılımın ait olduğu zararlı yazılım ailesini vermektedir. Ancak antivirüs uygulamaları arasında isimlendirme farkı olduğu için yanıtların tekilleştirilmesi gerektiği görüşmüştür. **Tablo Ek-1-7** arasında bulunan Malware, Trojan, Downloader, Backdoor, CoinMiner, Mirai ve Other olmak üzere 7 küme oluşturulmuştur. Oylama usulü ile zararlı yazılımın hangi kümeye ait olduğu antivirüslerin yanıtlarına göre değerlendirilmiştir. Aynı sayıda oy çıkması durumunda özel zararlı yazılım kümeleri genel kümelere tercih edilmiştir. En genel kümeler büyükten küçüğe sırasıyla Other, Malware ve Trojan olarak kabul edilmiştir. İkinci seviyede Downloader ve Backdoor tanımlanmıştır. En özel kümeler olarak CoinMiner ve Mirai kabul edilmiştir.



Şekil 10 Cowrie Bal küpü için geliştirilen Python uygulaması akış diyagramı

3.4.2. Tespit Edilemeyen Kötü Amaçlı Yazılımların Binwalk ile Entropi Analizi

Binwalk [27], bir ürün yazılımının görüntülerini analiz etmek, tersine mühendislik yapmak ve yazılımı çıkarmak için kullanılan açık kaynaklı bir araçtır. 2010 yılında Craig Heffner tarafından oluşturulan Binwalk uygulaması, dosya sistemi görüntülerini, yürütülebilir kodu, sıkıştırılmış arşivleri, önyükleyici ve çekirdek görüntülerini, JPEG ve PDF gibi dosya formatlarını ve daha fazlasını tanımlamak ve çıkarmak için bir aygıt yazılımı görüntüsünü tarayabilir ve dosya imzalarını arayabilir. Güvenlik açıklarını aramak için dosya sistemi görüntülerinin ikili dosyalarını (.bin) tersine çevirebilir. Görüntüden dosyaları çıkarabilir ve arka kapı şifrelerini veya dijital sertifikaları keşfedebilir. Analiz edilen ikili dosyaların hangi CPU mimarileri için işlem kodlarından oluştuğunu tanımlayabilir.

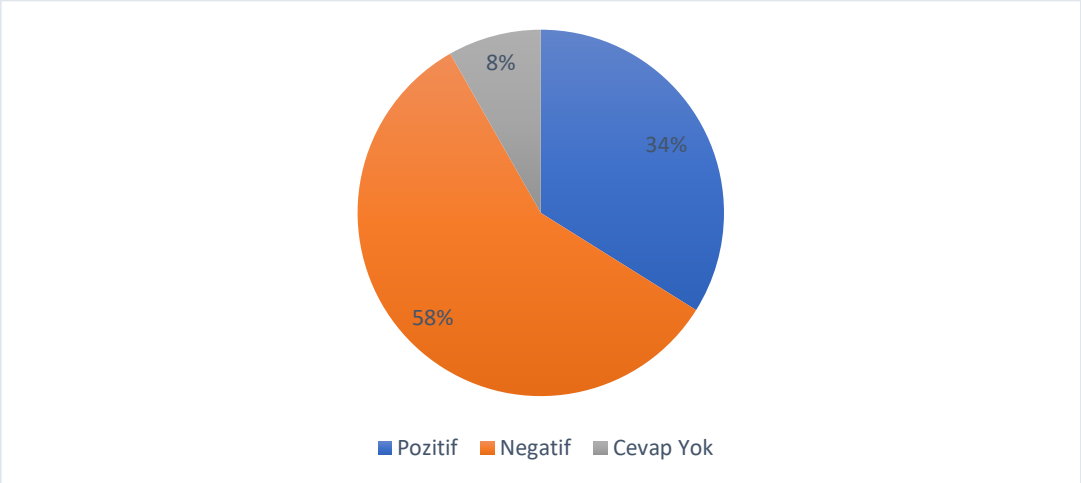
Belirli parola dosyalarını (passwd, shadow, vb.) aramak için dosya sistemi görüntülerinin sıkıştırmasını açabilir ve parola karmalarını kırmaya çalışabilir. Sıkıştırılmış verileri veya sabit kodlanmış kripto anahtarlarını aramak için veri entropi analizi yapabilir. Tüm bunlar, kaynak kodu erişimine ihtiyaç duyulmadan Binwalk aracı kullanılarak yapılabilir.

[27] https://embeddedbits.org/reverse-engineering-router-firmware-with-binwalk/

4. BULGULAR

4.1. VirusTotal Sonuçlarının Genel Değerlendirmesi

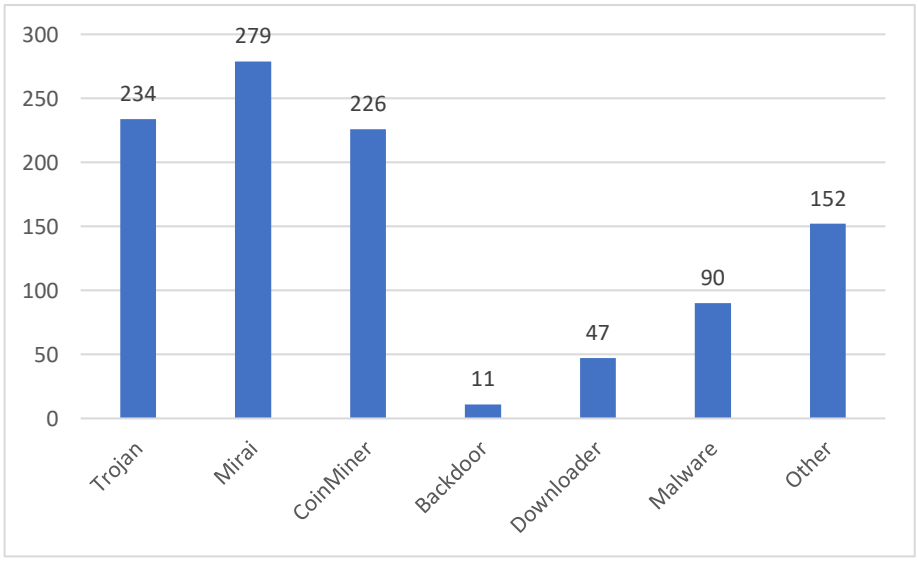
Google Cloud üzerine kurulan Cowrie Honeypot 47 gün boyunca saldırgan davranışları izlenerek kayıt altına alınmış ve saldırganlar tarafından sisteme yüklenen 50 adet şüpheli dosya ele geçirilmiştir. Geliştirilen pyhton betiği ile şüpheli dosyalar VirusTotal API (Application Programing Interface) üzerinde çalışan antivirüs uygulamaları ile imza tabanlı tespit tekniğine göre değerlendirilmiştir. Şüpheli dosyaların imzalarına VirusTotal üzerinden gelen değerlendirme sonuçları **Tablo 5'** te açıklaması verilen 'zararlı', 'zararlı değil' veya 'cevap yok' şeklinde sınıflandırılmıştır. VirusTotal üzerindeki 64 antivirüs uygulaması zararlı olarak belirlediği dosyaları, 'pozitif'; zararlı olarak belirlemediği dosyaları 'negatif'; herhangi bir sınıf belirleyemediği dosyalar için de 'cevap yok' dönüşünü vermiştir. Bunlar dışında belirli süre içerisinde hiçbir geri dönüş sağlamayan antivirüs uygulamaları da olmuştur. **Şekil 11**' de belirtilen sonuçlara göre 64 antivirüs uygulamasından yüzde otuz dört oranında zararlı dosya olduğu anlamına gelen pozitif dönüş alırken, yüzde elli sekiz oranında yüklenen dosyaların zararlı olmadığı sonucunu veren negatif dönüş alınmıştır. Bu dosyalara zararlı veya zararlı değil sınıflandırmasını yapamayan antivirüs uygulamaları da yüzde sekiz oranında gözükmektedir.



Şekil 11 Kötü amaçlı yazılım tespitinde anti-virüs uygulamalarının değerlendirme sonucu yüzdeleri

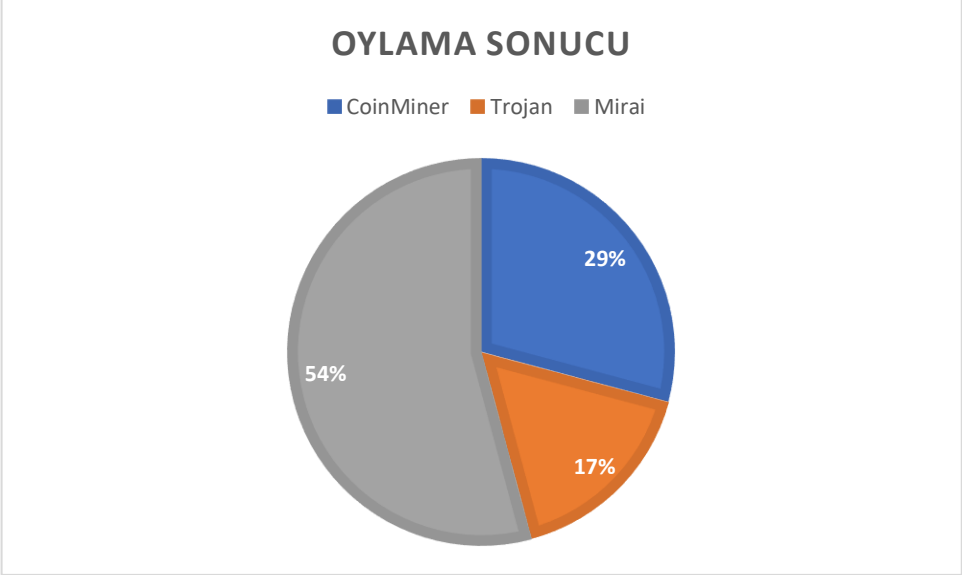
Şüpheli dosyalara dönüşü sağlanan cevaplara göre yapılan değerlendirmede herhangi bir şüpheli dosyaya yalnızca bir tane pozitif cevabının dönülmesi o şüpheli dosyanın pozitif olarak etiketlenmesi, yani zararlı olarak ele alınması için yeterli sayılmıştır. Örneğin addd0760… ile başlayan şüpheli dosyaya bir pozitif, 57 negatif, 6 cevap yok dönüşü sağlanmıştır. Bu sonuçlara göre addd0760… ile başlayan şüpheli dosyasının bir tane pozitif cevaba sahip olması zararlı olarak kabul edilmesini sağlamıştır.

Zararlı dosyalar amaçlarına, çalışma şekillerine veya farklı parametrelere göre çeşitli sınıflara ayrılmaktadırlar. Bunlardan kapsamlı olarak 'Zararlı Yazılımların Sınıflandırılması' başlığı altında bahsedilmiştir. Özet (hash) bilgileri sorgulanarak antivirüs uygulamalarından alınan cevaplara göre bu çalışma kapsamında altı ana başlık belirlenmiş ve sınıflandırma **Şekil 10**' da gösterildiği gibi yapılmıştır. Buna göre bu altı başlık 'Trojan Horse', 'Mirai', 'CoinMiner', 'Backdoor', 'Downloader', 'Malware' ve ‘Other’ olarak sıralanmıştır. En fazla tercih edilen zararlı dosya Mirai varyantları olurken bunu sırasıyla Trojan Horse ve CoinMiner takip etmiştir. Bir ya da iki kere bildirilen kümelerinin dışındaki zararlı dosyaların sınıflandırılması ‘Other’ başlığı altında yapılmıştır.



Şekil 12 Şüpheli dosyaların etiketlendiği zararlı yazılım sınıflandırma oranları

Antivirüs uygulamaları tarafından değerlendirilen 50 şüpheli dosya türünün sınıflandırma dağılımı **Şekil 12'** de verilmiştir. Bu sınıflandırma dağılımı her antivirüs uygulamasının şüpheli dosyalara verdiği yanıtlardan oluşmaktadır. 64 farklı antivirüs uygulamasının bir dosya imzasına verdiği yanıtlardan elde edilen sınıf dağılımı sayısı da **Şekil 12'** de görülmektedir. Oylama usulü ile antivirüs yazılımlarının bir dosya için en çok verdiği yanıt o şüpheli dosyanın türü kabul edilmiştir. Örneğin 46ffeb0d… ile başlayan şüpheli dosyaya 29 Pozitif cevap, 30 negatif cevap, 5 cevap yok kategorisinde dönüşler yapılmıştır. Bu cevaplar içerisinde oluşturulan 7 sınıfa (Trojan, Mirai, CoinMiner, Backdoor, Downloader, Malware, Other) ait dağılımlar incelenmiştir. 46ffeb0d… dosyası ise 9 Trojan, 0 Mirai, 14 CoinMiner, 0 Backdoor, 0 Downloader, 3 Malware, 3 Other dağılımına sahiptir. Buna göre oylama sonucu yapıldığında bu şüpheli dosya CoinMiner olarak etiketlenmiştir. Değerlendirilen tüm şüpheli dosyaların dağılımı incelendiğinde üç ayrı sınıf öne çıkmaktadır. **Şekil 13'** te belirtildiği üzere %54 oranında Mirai sınıfı ilk sırada bulunurken CoinMiner %29 oranıyla Mirai' yi ikinci sırada takip etmektedir. Trojan Horse %17 oranı ile üçüncü sırada yer almaktadır.



Şekil 13 Oylama Sonucu Değerlendirilmesi

4.2. Anti-virüs Uygulamalarının Karşılaştırılması

VirusTotal üzerinde bulunan ve bu çalışma kapsamında kullanılan 64 antivirüs uygulamasının belirli süre içerisinde yüklenen 50 zararlı dosyaya en fazla pozitif cevabı döndüğü ilk 20 anti-virüs uygulaması **Tablo 6**' da verilmiştir. Değerlendirmesi yapılan 50 şüpheli dosyadan iki tanesine hiçbir antivirüs uygulamasından pozitif cevap alınamamıştır. Bu iki dosya 'Tespit Edilemeyen Dosyaların Binwalk ile Entropi Analizi' başlığı altında ayrıca incelenmiştir. Bu sonuçlara göre en fazla pozitif cevabı dönen anti-virüs uygulaması AVG olurken bu sıralamayı Avast ve DrWeb uygulamaları takip etmektedir.

Tablo 6 Anti-virüs Uygulamalarının Zararlı Olarak Yanıt Verdiği Şüpheli Dosya Sayısı

Anti-virüs Uygulamaları	Pozitif Sayısı
AVG	47
Avast	42
DrWeb	40
TrendMicro-HouseCall	40
TrendMicro	38
McAfee-GW-Edition	36
Fortinet	35
ESET-NOD32	34
Kaspersky	34
McAfee	34
GData	33
Symantec	30
BitDefender	28
Emsisoft	28
FireEye	28
Avira	27
Cynet	27
Jiangmin	26
Lionic	26
MicroWorld-eScan	26

Antivirüs uygulamalarından bazılarında yüklenen dosyalara karşı 'pozitif' veya 'negatif' şeklinde sınıflandırma yapamadığı veya belirli süre içerinde hiç cevap vermediği gözlemlenmiştir. Sınıflandırma yapamayan veya cevap dönemeyen 19 anti-virüs uygulaması **Tablo 7**' de verilmiştir.

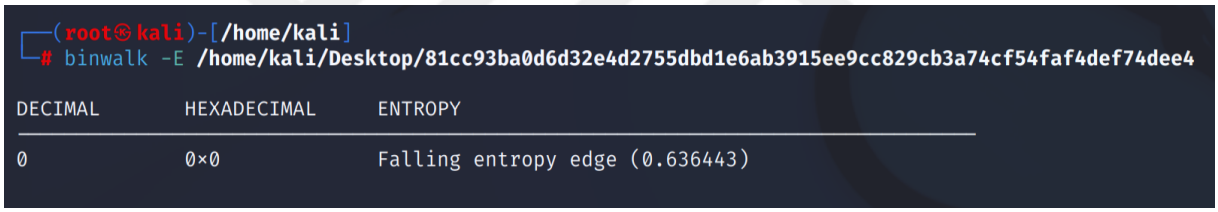
Tablo 7 Şüpheli Dosyaları Bulamayan veya Cevap Dönmeyen Antivirüs Uygulamaları

Anti-virüs Uygulamaları	Pozitif Sayısı
Acronis	0
Baidu	0
Bkav	0
CMC	0
eGambit	0
Elastic	0
F-Secure	0
K7AntiVirus	0
K7GW	0
Kingsoft	0
Malwarebytes	0
Panda	0
SUPERAntiSpyware	0
TACHYON	0
VBA32	0
VIPRE	0
ViRobot	0
Yandex	0
Zoner	0

4.3. Tespit Edilemeyen Dosyaların Binwalk ile Entropi Analizi Sonucu

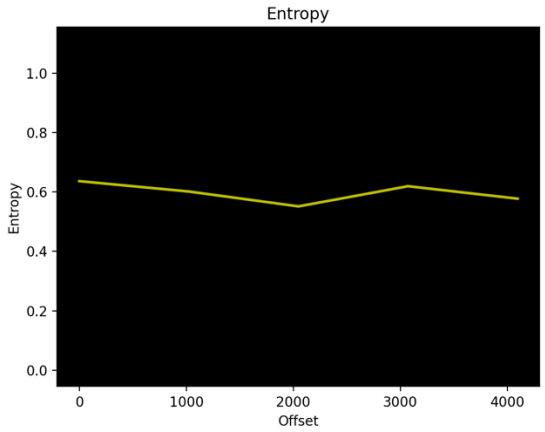
Zararları yazılımlar statik analiz yöntemlerini kullanan güvenlik uygulamalarından kaçınmak için genellikle içeriği saldırganlar tarafından doğrudan okunmayacak hale getirilir. Kaynak kod karmaşıklaştırma (code obfuscation) zararlı yazılımların anlaşılmasının önünde büyük bir engel olsa da entropi analizi ile yeni bir araştırma alanı sunulmaktadır. Bir dosya sıkıştırıldığında ve şifrelendiğinde dosyanın entropisi bire yaklaşır. Dosyanın içerisinde bilgi içeren kısımlar bulundukça dosyanın entropisi sıfıra yaklaşır.

Binwalk uygulaması ile terminal arayüzü üzerinden dosyaların entropisi 'binwalk -E' parametresi ile hesaplanır. İlk olarak VirüsTotal üzerinden 64 anti-virüs uygulaması tarafından tespit edilemeyen dosya **Şekil 14**' te görüldüğü üzere analiz edilmiştir. Entropi değeri ortalama 0.636443 olarak hesaplanmıştır.

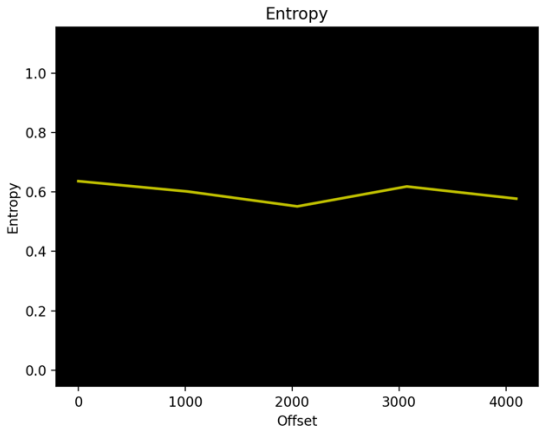


Şekil 14 Binwalk uygulaması ile dosyanın entropi analizinin yapılması

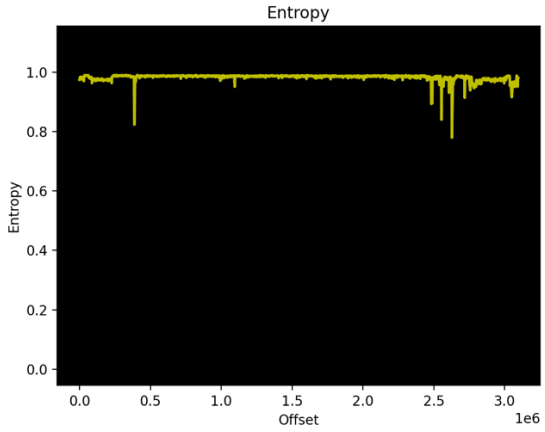
Zararlı olarak işaretlenen dosyalar ile şüpheli dosyaların offset bazında grafikleri çizdirildiğinde şüpheli iki dosyanın **Şekil 15.a** ve **Şekil 15.b** grafiklerin benzer bir görüntü oluşturduğu görülmüştür. İmza-tabanlı tespit tekniği ile zararlı dosya olduğu tespit edilen iki zararlı dosyanın **Şekil 15.c** ve **Şekil 15.d**' te benzer davranış gösterdiği görülmektedir. Bu iki tür dosyaların birbirinden görsel olarak ayrıldığı **Şekil 15**' te görülmektedir.



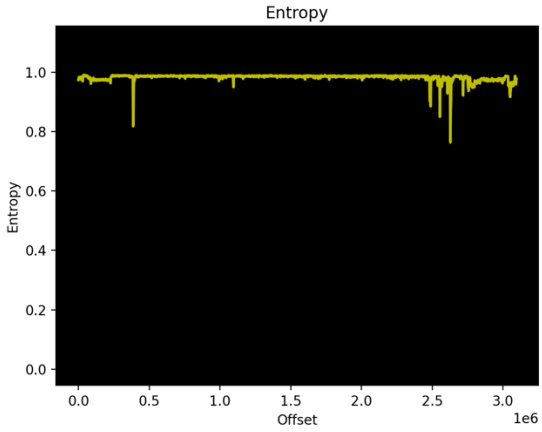
Şekil 15.a 81cc93ba0d6d32e şüpheli dosyanın entropi analizi



Şekil 15.b baed3518e5758ec şüpheli dosyanın entropi analizi



Şekil 15.c c3a1344da7beefb zararlı dosyanın entropi analizi



Şekil 15.d f4817f0ee7a893a zararlı dosyanın entropi analizi

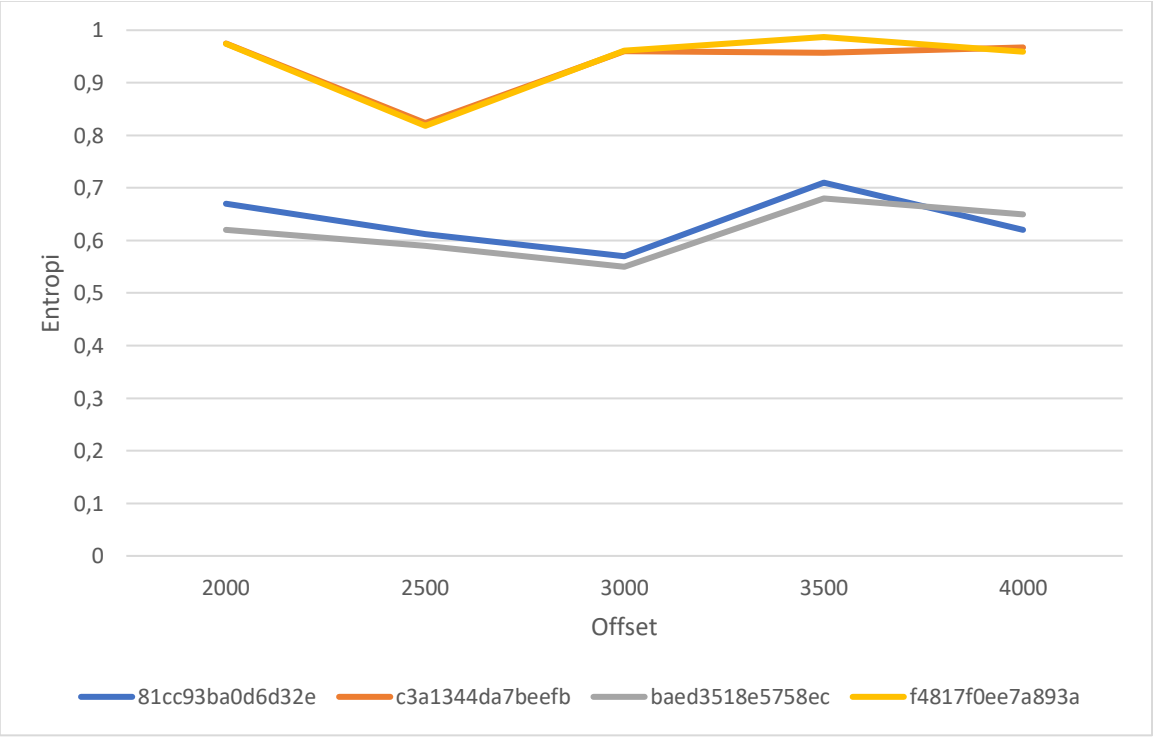
Şekil 15 Zararlı olduğu tespit edilen dosyalar ile şüpheli dosyaların entropi analizlerinin offset bazında analizi

Entropi değeri bire yakın olan ve zararlı yazılım olarak etiketlenmiş zararlı dosyalar diğerleriyle şüpheli iki dosya ile karşılaştırıldığında entropi değerlerinin **Tablo 8**'de ayrıştığı görülmektedir.

Tablo 8 Entropi analizi yapılan şüpheli dosyaların entropi değerleri

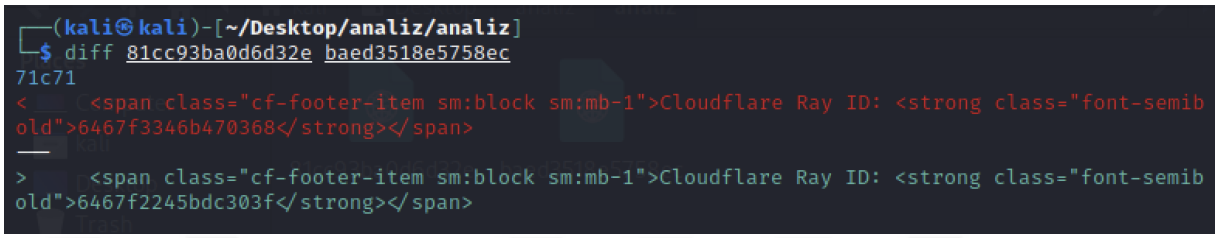
Hash	Entropi	Virüs Total Dönüşü
81cc93ba0d6d32e	0.636443	Cevap yok
baed3518e5758ec	0.636443	Cevap yok
c3a1344da7beefb	0,934245667	Pozitif
d20df25dbf2da55	0,9373115	Pozitif
e5b4786c5a4c61	0,936209706	Pozitif
f4817f0ee7a893a	0,939645722	Pozitif

Zararlı ve şüpheli dosyaların entropisinin offset bazında dağılımının karşılaştırılması **Şekil 16'** da görülmektedir. Şüpheli dosyaların entropileri paralel giderken zararlı dosyaların offset bazında entropi değeri grafikte nerdeyse üst üste binmektedir.



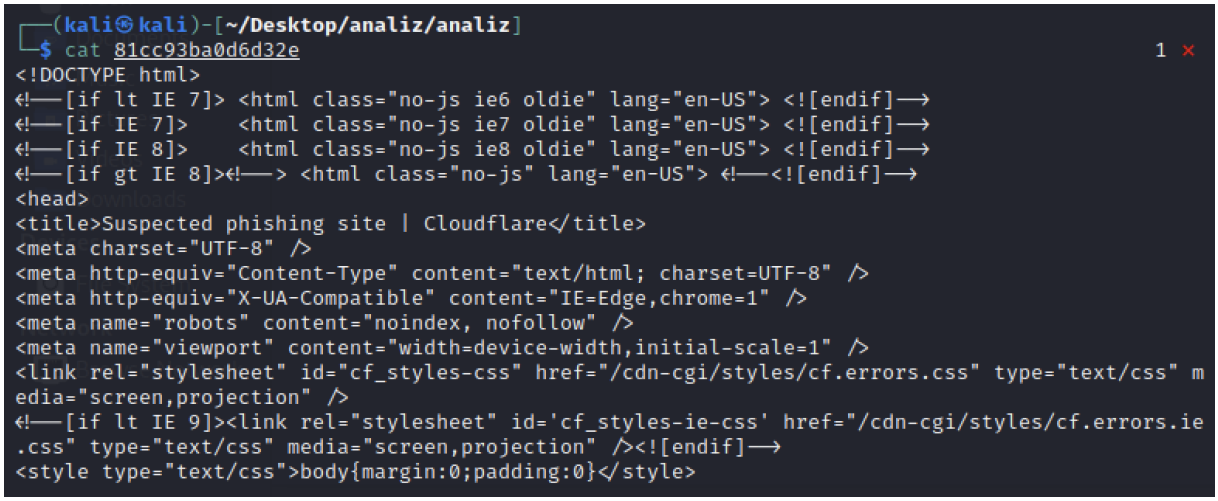
Şekil 16 Bal küpüne yüklenen dosyaların offset bazında entopi analizi

İki dosyanın farklı özet (hash) değerine sahip olmasına rağmen entropi değerlerinin aynı çıkması sebebiyle dosyaların içerikleri ikili olarak Linux işletim sistemlerinde kullanılan ve iki dosya arasındaki farkı veren ‘diff’ komutu ile **Şekil 17’** de gösterildiği şekilde karşılaştırma yapılmıştır.

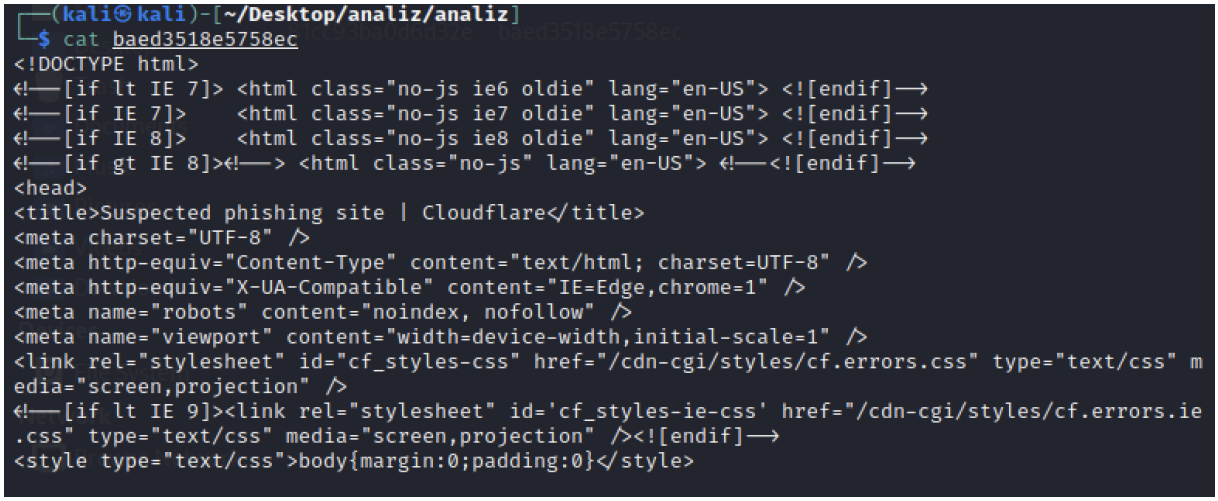


Şekil 17 diff komutu ile iki dosyanın karşılaştırılması

Linux işletim sistemlerinde kullanılan ve dosyaların içeriğini terminal üzerinden okumaya yarayan ‘cat’ komutu kullanılarak dosyalar okunduğunda html bir sayfanın kaynak kodlarına benzer olduğu görülmüştür. **Şekil 18** ve **Şekil 19’** da görüntülenen dosya içeriği muhtemel bir HTML hata sayfasının kaynak kodlarını gösterilmektedir.

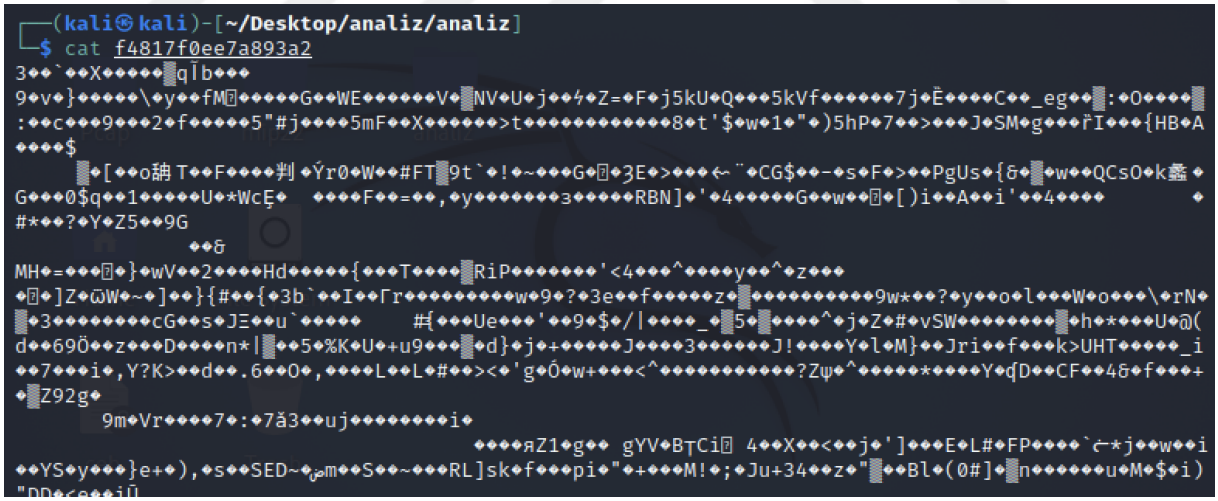


Şekil 18 81cc93ba0d6d32e cat komutu ile okunması



Şekil 19 baed3518e5758ec cat comutu ile okunması

Zararlı olarak etiketlenen dosyalarda ‘cat’ komutu kullanıldığında dosyanın içeriğinin okunamadığı **Şekil 20’** de görülmektedir.



Şekil 20 f4817f0ee7a893a cat komutu ile okunması

5. TARTIŞMA

Siber risklerin yönetilmesi için siber saldırı yöntemlerine karşı alınmış savunma tedbirleri yetersiz kalmaktadır. Klasik saldırı yöntemlerinin aksine 'zero-day attack' olarak adlandırılan keşfedilmemiş saldırı türlerinin mevcudiyeti alınan önlemleri geçersiz kılabilmektedir. Bu saldırıları ve diğer pek çok saldırı çeşidini kayıt altına alarak yeni güvenlik önlemlerinin uygulanması bal küpleri (honeypot) olarak adlandırılan sistemlerle mümkün olabilmektedir. Bu sistemlerin amacı saldırganın veya zararlı aktivitenin kullanmış olduğu yöntem ve araçlar hakkında bilgi toplamaktır. Özellikle zero-day attack saldırı türlerini keşfetmeye ve onlar için yeni savunma yöntemleri geliştirmeye de olanak sağlar. Saldırganlar cihazlara uzaktan erişim için yaygın olarak kullanılan SSH (Secure Shell) ve Telnet gibi protokolleri hedef haline getirmiştir. Bu çalışma kapsamında Google Cloud platformu üzerinde Ssh ve Telnet protokollerini simüle eden Cowrie bal küpü kurulumu yapılmıştır. Google Cloud üzerinde bulunan bal küpü sistemi güvenlik duvarındaki (firewall) konfigürasyon ayarlarıyla birlikte internetten yapılan tüm erişim isteklerine açık hale getirilmiştir. Saldırganların zafiyetli bir bulut servisi olarak gördüğü ve Shodan açık kaynak IoT cihazı arama motoruna da yansıyan Cowrie bal küpü farklı IP adreslerinden 3 milyon 626 bin 394 adet saldırı almıştır. Bal küpü sistemine erişim sağlayan saldırganlar tarafından 50 şüpheli dosya bal küpü içerisine yüklenmiş ve kaydedilmiştir. Tez çalışmasında gerçek saldırganlar tarafından sisteme yüklenen şüpheli dosyalar imza tabanlı tespit sistemleri ile değerlendirilerek, antivirüs uygulamalarının doğruluğunun ve performansının ölçülmesi amaçlanmıştır.

Çalışma kapsamında saldırı süresince güncel saldırılardan elde edilmiş 50 şüpheli dosyayı imza-tabanlı tespit yöntemi ile değerlendiren 64 antivirüs uygulamasının VirusTotal API üzerinden otomatik değerlendirilmesi amacıyla bir uygulama geliştirilmiştir. İmza tabanlı tespit tekniği ile çalışan antivirüsler, 50 şüpheli dosyadan 48'ini zararlı yazılım olarak etiketlemiştir ve bu sonuçlar eşliğinde antivirüs uygulamalarının yanıtları değerlendirilmiştir. 50 şüpheli dosya içerisinden iki şüpheli dosya antivirüs uygulamaları tarafından tespit edilememiştir ve tespit edilemeyen şüpheli iki dosya statik analiz yöntemlerinden olan kod okuma ile değerlendirilmiştir. Ayrıca statik analiz kapsamında zararlı dosyayı çalıştırmadan dosyanın entropi değerine de bakılarak zararlı olup olmadığı değerlendirilmiştir. Yapılan çalışmada aktif

olarak sahadan zararlı yazılımların toplanması ve bu zararlı yazılımların sınıflandırılarak aynı zamanda antivirüs uygulamalarının başarı oranının değerlendirilmesi hedeflenmiştir. Sonuçlara göre başarı oranı en yüksek antivirüs uygulaması 50 şüpheli dosyadan 47'sini zararlı olarak etiketleyen AVG olurken bu sıralamayı ikinci olarak Avast 42 tanesini ve üçüncü olarak DrWeb 40 tanesini zararlı yazılım olarak işaretlemiştir. Çalışma sonucunda iki şüpheli dosyayı hiçbir antivirüs uygulaması zararlı olarak etiketlememiştir. Kümülatif olarak değerlendirildiğinde anti-virüs uygulamaları Virüs Total üzerinden sorgulanan 50 şüpheli dosyandan 48' inin zararlı yazlım olduğunu tespit etmiştir. 50 şüpheli dosya için hiç geri dönüş alınamayan 19 antivirüs uygulamasının ise veri tabanlarının güncel olmadığı veya belirli süre içerisinde cevap dönemedikleri tahmin edilmektedir. VirüsTotal üzerinde bulunan 64 anti-virüs uygulamasının yaklaşık %30' unun güncel zararlı yazılımları tespit edemediği veya gereken zaman içinde (API cevap süresi) yanıt oluşturamadığı görülmüştür. Tez çalışmasında kullanılan imza tabanlı tespit yöntemi zararlı yazılımları tespit etmek için sahada yaygın olarak kullanılan bir yöntemdir. Test edilen anti-virüs uygulamalarından AVG zararlı yazılımların tamamına yakınını bilerek en yüksek başarıyı göstermiştir. AVG den sonra en yüksek %87,5 başarı ile AVAST ve % 83,3 başarı DrWeb antivirüs uygulamaları olmuştur. En fazla pozitif cevap dönen ilk 3' te %85' in altında olan 10 antivirüs uygulaması değerlendirildiğinde ise başarının %70 oranlarına düştüğü görülmektedir.

Truva atları ve diğer zararlı yazılımlar kişisel bilgisayarların gelişmesiyle ortaya çıkmışlardır. İlk Truva atları 1980'lerin başında teorik bir kavram olarak karşımıza çıkarken 1990' larda geliştirilmeye devam edilmiştir. Çalışma kapsamında anti-virüs uygulamalarının döndüğü cevaplara göre değerlendirildiğinde %17 oranı ile en fazla saldırının geldiği zararlı yazılım sınıfı üçüncü sırada Truva atı olmuştur. Truva Atı zararlı yazılımları uzun yıllardır bilişim sistemlerini tehdit eden bir zararlı yazılım olmasına rağmen hala aktif bir saldırı vektörüdür. Truva atlarının geliştirilme yılına göre değerlendirildiğinde daha yeni bir sınıf olan ve Ekim 2016 yılında DYNDNS saldırısı ile hayatımıza girerek firmaların milyarlarca dolar zarar etmesine sebep olan Mirai zararlı yazılım saldırısı hala aktif bir vektör olarak denenmektedir. Bu çalışmada anti-virüs uygulamaları zararlı dosyaların sınıfını %54 oranı ile ilk sırada yer alan Mirai olarak belirlemiştir. Bal küpü sisteminin Ssh ve Telnet protokollerini simüle ediyor olması da Mirai saldırısı için uygun bir ortam oluşturmuştur. Varsayılan (default) kullanıcı adı-şifre ilişkilerini kullanan saldırıların hala aktif olması sistemlerin öncelikli zafiyetinin varsayılan erişim bilgileri olduğunu göstermektedir. Kripto varlıkların önem kazanmasıyla

zararlı yazılımların da kullanıcıların bilgilerini çalmak yerine işlem gücünü sömüren saldırı yöntemleri geliştirilmeye başlanmıştır. European Union Agency for Cybersecurity (ENISA) yıllık raporuna göre, Cryptojacking 2021' de en yaygın üçüncü siber güvenlik tehdidi olmuştur. Aynı yıl, Google'ın Siber Güvenlik Eylem Ekibi, güvenliği ihlal edilmiş bulut platformlarının %86' sının kripto hırsızlığından kaynaklandığını tespit etmiştir. 2020' de Cisco, müşterilerinin %69' unun kripto madencilik zararlı yazılımlarından etkilendiğini bildirmiştir. Bu çalışma kapsamında ise bulut platformu üzerinde bulunan Cowrie bal küpüne gelen saldırılardan elde edilmiş zararlı dosyaların %29' u ile ikinci sırada yer alan Cryptojacking zararlı yazılımı tespit edilmiştir. Sonuç olarak eski/geleneksel tehditler günümüzde kullanılmaya devam etmekte ve bunlara ek olarak her geçen gün farklı amaçlar doğrultusunda zararlı yazılımlar geliştirilmektedir.

AVG anti-virüs uygulamasının bilemediği tek şüpheli dosyayı tespit edebilen antivirüs uygulamasının McAfee-GW-Edition olduğu görülmüştür. İlgili şüpheli dosyaya McAfee-GW-Edition antivirüs uygulamasından başka pozitif dönüş yapan anti-virüs uygulamasının olmadığı da görülmüştür. Bu şüpheli dosya McAfee-GW-Edition tarafından Trojan olarak etiketlenmiştir. Sistem güvenliğinin ihlal edilmesi için bir zararlı yazılımın sisteme bulaşması yeterli olduğu göz önüne alındığında imza tabanlı sistemlerin %99 başarıda bile ne kadar kırılgan olduğu görülmektedir. İlgili truva atının AVG anti-virüs uygulamasının çalıştığı bir sisteme bulaşması durumunda %97.91 başarısı sistem güvenliği için yeterli olmayacaktır. Şüpheli dosya kümemizde %75 başarı oranı sağlayan McAfee-GW-Edition anti-virüs uygulamasının ilgili truva atını (trojan horse) tek başına tespit etmesi olası diğer saldırıların %25' inden sistemi koruyamadığı görülmektedir. Antivirüs uygulamaları VirusTotal gibi platformlarla birlikte çalıştırılmalı veya sistem güvenliği için yeni yöntemler kullanılmalıdır.

Çalışma kapsamında toplanan dosyalardan imza tabanlı tespit tekniği ile tespit edilemeyen iki şüpheli dosya için statik analiz kapsamında kod okuma ve entropi analizi yöntemleri uygulanmıştır. Şüpheli dosyaların güvenliği sağlanmış ve internet erişimi kesilmiş Linux işletim sistemine sahip bir sanal makine ortamında çalıştırılmadan statik olarak değerlendirmesi yapılmıştır. Entropi analizinde şüpheli dosyaların entropisi 0,6364 değerini verirken zararlı yazılım olarak işaretlenen dosyalar için 0,9 un üzerinde bir değer verdiği görülmüştür. Şüpheli dosyaların entropi analizi sonucu aynı çıkarken dosya özetlerinin farklı olması, dosyaların içeriklerinin benzer olduğu düşüncesini ortaya çıkarmıştır. Yapılan benzerlik değerlendirmesinde kod içeriğin yalnızca iki satırında benzerlik görülmüştür. Benzeyen

içeriğin HTML kodu olduğu anlaşılmıştır. Dosyaların yaygın PUA' lardan (grayware) olan Browser Hijacking dosyası olup olmadığı kontrol edilmiş ve dosya içerikleri okunarak bir hata sayfası olduğu görülmüştür. Bu durum saldırganın sisteme dosya yüklemeden önce rastgele bir web adresinin içeriğini Cowrie bal küpü sistemine yüklemiş olduğunu düşündürmektedir. Temel statik analiz kapsamında uygulanan yöntemler zararlı olarak tespit edilen yazılımlarda içerik karmaşıklaştırıldığı (code obfuscation) için değerlendirmeye yetecek bir bulgu vermemiştir.

Şüpheli dosyalara yapılan toplamda 3200 değerlendirmeden anti-virüs uygulamaları tarafından yalnızca 1039 pozitif veya negatif cevap verilmiştir. Sonuç olarak yapılan API isteklerinin yaklaşık yüzde 67'sinin negatif cevap vermesi ya da hiç cevap vermemesi imza tabanlı sistemlerin başarısını tartışmalı hale getirmektedir. 19 adet antivirüs uygulamasından hiçbir şekilde geri dönüş alınamamıştır. Değerlendirmeye dahil 64 antivirüs uygulamasının 19 tanesinden geri dönüş alınamaması;

- Anti-virüs uygulamasının performansı,
- VirusTotal performansı,
- API response süresinin yeterli olmaması,
- Çalışmada kullanılan veri setinin güncel olması

gibi nedenlere bağlı olabilmektedir. Bu çalışmayla birlikte tek başına imza tabanlı tespit yönteminin yeterli olmadığı anlaşılmıştır. İmza tabanlı tespit yöntemi haricinde statik analiz ve dinamik analiz yöntemlerinin uygulanmasının sistem güvenliği açısından önemi ortaya konulmuştur. Tez çalışmasıyla ulaşılan sonuçlar imza-tabanlı yöntemleri kullanan uygulamanın başarısını uygulama bazında gösterirken, genel itibariyle yöntemin çalışma prensibi de kırılganlığını göstermektedir. Emek/Acı piramidine (Pain of Pyramid) göre en kolay değiştirilebilen izlerden olan özet bilgisinin kolaylıkla değiştirilebilir oluşu, güvenliği sağlamak için sadece bu yöntemi kullanan güvenlik uygulamaları için risk oluşturmaktadır.

6. SONUÇ VE ÖNERİLER

Bilişim sistemlerinin ve özellikle İnternet teknolojilerinin gelişmesiyle birlikte, bu sistemlerdeki açıklıklardan kaynaklanan güvenlik tehditleri de artmaya başlamıştır. Güncel açıklıklar üzerinde güvenliğin sağlanabilmesi adına farklı türleri olan saldırılara karşı kimlik doğrulama, yetkilendirme, antivirüs programları gibi güvenlik çözümleri geliştirilmeye çalışılmıştır. Her geçen gün büyük bir hızla artan saldırı tekniklerine karşı savunma teknikleri için aynı şeyi söylemek zordur. Gerçekleşen bu saldırılar hakkında yeni teknikler keşfetmek ve zamanında önlemler almaya çalışmak oldukça önemlidir. Bilişim dünyasında saldırıları tespit etmek ve yeni saldırı türlerini tanıyabilmek adına kullanılan en önemli yöntemlerden biri bal küpü (honeypot) sistemleridir. Diğer yöntemlerin aksine saldırganı durdurmak için aktif rol oynamazlar. Amacı, zararlı aktivitenin kullanmış olduğu yöntem ve araçlar hakkında bilgi toplamaktır. Saldırgan ve yöntemi hakkında bilgi sahibi olmak var olan savunma sistemlerini geliştirmek için izlenilen önemli bir yol olmuştur. Bal küpü sistemleriyle ilgili yapılan çalışmalar yıllardan beri güvenlik alanındaki önemini korumaktadır. Bu çalışma kapsamında Google Cloud platformu kullanılarak bulut bilişim üzerinde güncel zararlı yazılımları toplamak amacıyla Cowrie bal küpü kurulmuştur. Bulut üzerinde 47 gün boyunca açık kalan sisteme dünyanın farklı bölgelerinde bulunan IP adreslerinden gerçek saldırganların ya da saldırganların robotlaştırdığı bilgisayarların (bot) yaptığı saldırılar ile şüpheli dosyalar toplanmıştır. Tez çalışması kapsamında Cowrie bal küpü üzerinde şüpheli dosyaları değerlendirme özelliği bulunmadığı için geliştirilen Python uygulamasıyla VirusTotal API kullanılarak antivirüs uygulamalarına imza tabanlı tespit yaptırılmıştır. Tez çalışması kapsamında imza tabanlı tespit sistemlerinin güncel zararlı yazılımlara karşı doğruluğunun ve performansının ölçülmesi amaçlanmıştır.

VirusTotal üzerindeki 64 antivirüs uygulaması zararlı olarak belirlediği dosyaları 'positif'; zararlı olarak belirlemediği dosyaları 'negatif'; belirleyemediği dosyalar için de 'cevap yok' dönüşünü vermiştir. Antivirüs yazılımlarının herhangi biri tarafından zararlı olarak bildirilen yani pozitif dönüşünü veren şüpheli dosyalar bu çalışmada zararlı yazılım (malware) olarak kabul edilmiştir. Antivirüs uygulamaları yüklenen 50 zararlı dosyadan 48 tanesine dönüş yaparken iki tanesine hiçbir pozitif cevap alınamamıştır. Sonuçlara göre en fazla pozitif cevabı dönen antivirüs uygulaması 47 pozitif cevap ile AVG olurken bu sıralamayı 42 pozitif cevap ile Avast ve 40 pozitif cevap ile DrWeb uygulamaları takip etmektedir. Antivirüs uygulamalarından bazılarında yüklenen dosyalara karşı 'zararlı' veya 'zararlı değil' şeklinde sınıflandırma yapamadığı veya belirli süre içerinde hiç cevap vermediği gözlemlenmiştir. Sınıflandırma yapamayan veya cevap dönemeyen 19 anti-virüs uygulaması **Tablo 7**' de verilmiştir. Bu bağlamda 50 dosya için hiç geri dönüş alınamayan 19 antivirüs uygulamasının veri tabanlarının güncel olmadığı veya belirli süre içerisinde cevap dönemeyecek kadar veri tabanlarının büyük olduğu görüşleri oluşmuştur. Çalışma kapsamında antivirüs yazılımlarının döndüğü negatif ve pozitif yanıtların yanında zararlı yazılımın türünü de tespit eden dönüşler mevcuttur. Her antivirüs uygulaması farklı isimlendirmeler kullanabildiğinden öncelikli olarak cevapların ana başlıklar içinde sadeleştirmesi yapılmıştır. Yapılan sadeleştirme aşamasında olası bilgi kayıpları genel istatistikler çıkarılabilmesi için göz ardı edilmiştir. Ham dosyalar Github[28] adresinde, antivirüs uygulamaları tarafından verilen cevaplar ve sınıflar ekte tablo olarak verilmiştir. Bir şüpheli dosyanın türüne karar vermek için antivirüs yazılımlarının pozitif olarak dönmüş olduğu cevapların arasından en çok tercih edilen sınıf, oylama gibi değerlendirilerek en çok oyu alan sınıf olduğu kabul edilmiştir. Eşit oy alan sınıflardan daha özelleşmiş alt sınıflar tercih edilmiştir. Örneğin Trojen Horse ile Mirai eşit sayıda dönüş aldığında şüpheli dosya Mirai zararlısı sınıfıyla etiketlenmiştir. Saldırganlar tarafından en fazla tercih edilen zararlı dosya Mirai varyantları olurken bunu sırasıyla Trojen Horse ve CoinMiner takip etmiştir.

[28] https://github.com/istec-iuc/cowrie-logs/

Geliştirilen Python uygulaması ile VirusTotal üzerinde imza sorgulamasında herhangi bir yanıt dönüşü olmayan (sonuç yok, cevap yok) veya zararlı yazılım değil (negatif) olarak cevap dönülen şüpheli dosyaların ise statik analiz kapsamında kod okuması ve entropi analizi yöntemleri kullanılmıştır. Entropi analizi doğrudan statik analiz başlığı altında kullanılan bir yöntem değildir. Fakat zararlı yazılımın sistem üzerinde çalıştırılmadan içeriği hakkında entropisine bakılarak bilgi edinilebilmesi statik analiz kapsamına girmektedir. Bu çalışmada entropi analizi statik analiz değerlendirmesi için bir yöntem olarak sunulmaktadır. Entropi analizi karmaşıklaştırılmış, sıkıştırılmış ve şifrelenmiş dosyalar için doğrudan fikir vermesine karşın bir şüpheli dosyanın doğrudan zararlı olduğu hakkında ya da zararlı yazılım türü hakkında bir bilgi vermemektedir. Temel statik analiz araçları ise entropisi bire yakın, içeriği gizlenmiş dosyaları analiz etmek için yetersiz kalabilmektedir. Çalışma kapsamında tespit edilemeyen şüpheli dosyaların analizi için kullanılmıştır ve entropi değerlerinin 0.6' ya yakın olduğu gözlemlenmiştir. Entropisi yüksek dosyalar için içeriğin gizlenmesini sağlayan yöntemin tespitinin ve tersine mühendislik yöntemleriyle analizinin yapılması gerekmektedir. Çalışma kapsamında entropi değeri 0,65' nin altında olan dosyalar için temel statik analiz yöntemleriyle şüpheli dosyaların değerlendirilmesi yapılmıştır. Antivirüslerin tespit edemediği entropisi bire yakın dosyalar sisteme yüklenmediği için tez çalışmasında bire yakın entropisi bulunan dosyaların statik analizi yapılmamıştır.

Anti-virüs uygulamalarının başarı oranı zararlı yazılımlar için yüzde 99 olsa bile, kurumsal firmalar ve son kullanıcılar için değerlendirildiğinde imzası veri tabanında bulunmayan herhangi bir zararlı yazılım bulaşması halinde sistemler zarar görebilir. Çalışma kapsamında yapıldığı gibi bir işletim sistemi üzerinde birden fazla anti-virüs uygulamasının çalıştırılması genellikle mümkün olmadığı için imza veri tabanları güncel ve geniş olan anti-virüs uygulamalarının tercih edilmesi gerekmektedir. Genellikle anti-virüs uygulamalarının sistem gereksinimleri (işlem gücü, ram, internet vb.) sistem performansını olumsuz etkilemesi ya da lisans maliyetlerinin fazla olmasından kaynaklı kullanılmaması gibi dezavantajlar mevcuttur. Bu nedenle tez çalışmasında birden çok anti-virüs uygulamasının değerlendirilmesinin yanında zararlı dosyaların çoklu antivirüs üzerinde denenme imkanı olmadığı için tespit edilememe ihtimalini artırdığı görülmüştür. Bu bağlamda sistemlerin daha güvenli hale gelmesi için antivirüs uygulamalarının birbirleriyle veri tabanı paylaşımı yapması uygun bir çözüm olarak değerlendirilebilir. VirusTotal birden fazla anti-virüs uygulaması çalıştırdığından dolayı çoklu antivirüs kullanımını kolay ve avantajlı hale getirmektedir. Çalışma kapsamında belirli bir süre

açık tutulan bal küpü üzerine yüklenen dosyalar analiz edilmiştir. İlerleyen dönemde yeterli bütçe ve ortam sağlanması durumunda sürekli tehditleri analiz eden ve zararlı yazılım toplayan canlı sistem geliştirilmesi mümkündür. Yüklenen zararlı yazılımlardan anti-virüslerin tespit edemediği dosyalara entropi ve statik analiz yanında tersine mühendislik ve dinamik analiz yöntemlerinin de uygulanması mümkündür.

KAYNAKLAR

[1] H. Wang, H. He, W. Zhang, W. Liu, P. Liu, and A. Javadpour, "Using honeypots to model botnet attacks on the internet of medical things," *Comput. Electr. Eng.*, vol. 102, no. June, 2022, doi: 10.1016/j.compeleceng.2022.108212.

[2] M. R. Amal, "Review of Cyber Attack Detection : Honeypot System," vol. 19, no. 1, pp. 5497–5514, 2022, doi: 10.14704/WEB/V19I1/WEB19370.

[3] S. S. Chakkaravarthy, D. Sangeetha, and V. Vaidehi, "A Survey on malware analysis and mitigation techniques ☆," *Comput. Sci. Rev.*, vol. 32, pp. 1–23, 2019, doi: 10.1016/j.cosrev.2019.01.002.

[4] N. Pachhala, S. Jothilakshmi, and B. P. Battula, "A Comprehensive Survey on Identification of Malware Types and Malware Classification Using Machine Learning Techniques," pp. 1207–1214, 2021.

[5] C. Rohith, "A Comprehensive Study on Malware Detection and Prevention Techniques used by Anti-Virus," 2021.

[6] H. Almohannadi, I. Awan, J. Al Hamar, A. Cullen, J. P. Disso, and L. Armitage, "Cyber threat intelligence from honeypot data using elasticsearch," *Proc. - Int. Conf. Adv. Inf. Netw. Appl. AINA*, vol. 2018-May, pp. 900–906, 2018, doi: 10.1109/AINA.2018.00132.

[7] K. Oosthoek and C. Doerr, "Cyber Threat Intelligence : A Product Without a Process ? Cyber Threat Intelligence : A Product Without a Process ?," *Int. J. Intell. CounterIntelligence*, vol. 34, no. 2, pp. 300–315, 2021, doi: 10.1080/08850607.2020.1780062.

[8] A. Choudhary, G. Choudhary, K. Pareek, C. Kunndra, J. Luthra, and N. Dragoni, "Emerging Cyber Security Challenges after COVID Pandemic: A Survey," *J. Internet Serv. Inf. Secur.*, vol. 12, no. 2, pp. 21–50, 2022, doi: 10.22667/JISIS.2022.05.31.021.

[9] T. Untawale, “Importance of Cyber Security in Digital Era,” *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 9, no. 8, pp. 963–966, 2021, doi: 10.22214/ijraset.2021.37519.

[10] R. BENZER and S. M. ARIKAN, “Bir Güvenlik Trendi: Bal Küpü,” *Acta INFOLOGICA*, vol. 2, no. 1, pp. 1–11, 2018, doi: 10.30801/acin.356815.

[11] F. Dal, “A CASE STUDY ON THE ANALYSIS OF ATTACKS AGAINST INTERNET OF THINGS SYSTEMS WITH HONEYPOT SYSTEMS (Yüksek Lisans Tezi),” *Hacettepe Üniversitesi*, 2019.

[12] J. W. Hoopes, *Virtualization for Security*. 2009.

[13] C. Stoll, *THE CUCKOO ’ S EGG*. .

[14] B. Cheswick and T. B. Laboratories, “An Evening with Berferd In Which a Cracker is Lured , Endured , and Studied.”

[15] Cohen Fred, “A Note on the Role of Deception in Information Protection,” vol. 17, pp. 483–506, 1998.

[16] E. Joshi, R. C., and Anjali Sardana, *Honeypots: a new paradigm to information security*. 2011.

[17] L. Spitzner, “The Honeynet Project: Trapping the Hackers,” pp. 15–23, 2003.

[18] M. Roesch, “Snort - Lightweight Intrusion Detection for Networks,” *Proc. LISA ’99 13th Syst. Adm. Conf.*, 1999.

[19] E. Balas, “Sebek,” *Usenix Mag.*, vol. 28, no. 6, 2003.

[20] S. Yüksel, “Analyzing the Medium-Interaction Honeypot: a Case Study,” no. February, 2018.

[21] A. N. Singh, “A Honeypot System for Efficient Capture and Analysis of Network Attack Traffic,” no. Icsccn, pp. 514–519, 2011.

[22] Ö. Erdem, M. Kara, and A. İkinci, “HoneyThing : Nesnelerin İnterneti için Tuzak Sistem,” no. October 2015, 2016.

[23] M. F. Razali, G. Muruti, M. N. Razali, N. Jamil, and F. Z. Mansor, “IoT honeypot: A review from researcher’s perspective,” *2018 IEEE Conf. Appl. Inf. Netw. Secur. AINS 2018*, pp. 93–98, 2019, doi: 10.1109/IISA.2018.8631494.

[24] L. A. Stafira, “EXAMINING EFFECTIVENESS OF WEB-BASED INTERNET OF THINGS HONEYPOTS,” 2019.

[25] L. Metongnon and R. Sadre, “Prevalence of IoT protocols in telescope and honeypot measurements,” *J. Cyber Secur. Mobil.*, vol. 8, no. 3, pp. 321–340, 2019, doi: 10.13052/jcsm2245-1439.832.

[26] B. Kaur and P. K. Pateriya, “A Survey on Security Concerns in Internet of Things,” *Proc. 2nd Int. Conf. Intell. Comput. Control Syst. ICICCS 2018*, no. Iciccs, pp. 27–34, 2019, doi: 10.1109/ICCONS.2018.8663175.

[27] S. Dowling, M. Schukat, and H. Melvin, “A ZigBee honeypot to assess IoT cyberattack behaviour,” *2017 28th Irish Signals Syst. Conf. ISSC 2017*, 2017, doi: 10.1109/ISSC.2017.7983603.

[28] L. Rist, S. Vetsch, M. Koßin, and M. Mauer, “Know Your Tools : Glastopf- A Dynamic Low Interaction Web Application Honeypot,” *Honeynet Proj.*, pp. 1–29, 2010, [Online]. Available: http://www.honeynet.org.

[29] M. Wang, J. Santillan, and F. Kuipers, “ThingPot: an interactive Internet-of-Things honeypot,” 2018, [Online]. Available: http://arxiv.org/abs/1807.04114.

[30] T. Luo, Z. Xu, X. Jin, Y. Jia, and X. Ouyang, “IoTCandyJar: Towards an Intelligent-Interaction Honeypot for IoT Devices,” *Black Hat 2017*, pp. 1–11, 2017, [Online]. Available: https://paper.seebug.org/papers/Security Conf/Blackhat/2017_us/us-17-Luo-Iotcandyjar-Towards-An-Intelligent-Interaction-Honeypot-For-IoT-Devices-wp.pdf.

[31] P. Krishnaprasad, “Capturing attacks on IoT devices with a multi-purpose IoT honeypot,” no. May, 2017, [Online]. Available: https://security.cse.iitk.ac.in/sites/default/files/15111021.pdf.

[32] Y. Minn *et al.*, “IoTPOT : Analysing the Rise of IoT Compromises Graduate School of Environment and Information Sciences / Institute of Advanced Sciences Yokohama

National University , Japan National Institute of Information and Communications Technology , Japan Institute ," *9th USENIX Work. Offensive Technol. (WOOT' 15)*, 2015.

[33] S. Gokhale, "Industrial Control Systems Honeypot : A Formal Analysis of Conpot," no. December, pp. 44–56, 2020, doi: 10.5815/ijcnis.2020.06.04.

[34] S. Lau, J. Klick, S. Arndt, and V. Roth, "POSTER: Towards highly interactive honeypots for industrial control systems," *Proc. ACM Conf. Comput. Commun. Secur.*, vol. 24-28-Octo, pp. 1823–1825, 2016, doi: 10.1145/2976749.2989063.

[35] M. A. Hakim, H. Aksu, A. S. Uluagac, and K. Akkaya, "U-PoT: A Honeypot Framework for UPnP-Based IoT Devices," *arXiv*, 2018.

[36] Committee on National Security Systems, "Committee on National Security Systems (CNSS) Glossary," 2005. doi: 10.1201/9780203888933.ch13.

[37] İ. Bulut, "Analiz sürecini atlatmaya çalışan zararlı yazılımlar ve derin öğrenme temelli zararlı yazılım tespiti," 2017.

[38] V. Stavova, L. Dedkova, V. Matyas, M. Just, D. Smahel, and M. Ukrop, "Experimental large-scale review of attractors for detection of potentially unwanted applications," *Comput. Secur.*, vol. 76, pp. 92–100, 2018, doi: 10.1016/j.cose.2018.02.017.

[39] V. Stavova, V. Matyas, and M. Just, "On the impact of warning interfaces for enabling the detection of Potentially Unwanted Applications," 2016.

[40] D. Aygör, "The limitations of signature-based and dynamic analysis methods in detecting malwares : A case study Kötücül yazılımların tespitinde imza temelli ve dinamik analiz yöntemlerinin zayıflıkları : Örnek olay çalışması The limitations of signature-based and dy," vol. 1, pp. 305–315, 2022, doi: 10.17341/gazimmfd.668290.

[41] U. Inayat, M. F. Zia, and F. Ali, "Comprehensive Review of Malware Detection Techniques," no. Icic, pp. 0–5, 2021.

[42] D. Laka, "Malware : Types , Analysis and Classification."

[43] E. Tekiner, A. Acar, A. S. Uluagac, E. Kirda, and A. A. Selcuk, "SoK : Cryptojacking

Malware," pp. 120–139, 2021, doi: 10.1109/EuroSP51992.2021.00019.

[44] S. Talukder and Z. Talukder, "A SURVEY ON MALWARE DETECTION," vol. 12, no. 2, pp. 37–57, 2020, doi: 10.5121/ijnsa.2020.12203.

[45] S. A. Roseline and S. Geetha, "A comprehensive survey of tools and techniques mitigating computer and mobile malware attacks," *Comput. Electr. Eng.*, vol. 92, no. March, p. 107143, 2021, doi: 10.1016/j.compeleceng.2021.107143.

[46] S. Varlioglu, N. Elsayed, Z. Elsayed, and M. Ozer, "The Dangerous Combo : Fileless Malware and Cryptojacking," pp. 125–132, 2022.

[47] M. N. Alenezi, H. Alabdulrazzaq, A. A. Alshaher, and M. M. Alkharang, "Evolution of Malware Threats and Techniques : A Review," no. December 2020, 2021.

[48] I. Kuzminykh, "Analysis of Security of Rootkit Detection Methods," no. December, 2019.

[49] M. Nieles, K. Dempsey, and V. Y. Pillitteri, "An Introduction to Information Security," *NIST Spec. Publ.*, pp. 1–101, 2017, [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-12r1.pdf.

[50] N. Alqudah, "A Survey on malware detection techniques," pp. 371–376, 2021.

[51] A. Chavan, "Implementation of Portable Antivirus System using Signature-based Detection and Heuristic Analysis," pp. 1481–1486, 2021.

[52] Ö. ASLAN, "Zararlı Yazılımların Göstermiş Oldukları Davranışlara Göre Analiz ve Tespit Edilmesi," 2020.

[53] A. Damodaran, F. Di, T. Corrado, A. Visaggio, T. H. Austin, and M. Stamp, "A comparison of static , dynamic , and hybrid analysis for malware detection," *J. Comput. Virol. Hacking Tech.*, vol. 13, no. 1, pp. 1–12, 2017, doi: 10.1007/s11416-015-0261-z.

[54] Ö. Aslan, "A Comprehensive Review on Malware Detection Approaches," *IEEE Access*, vol. 8, pp. 6249–6271, 2020, doi: 10.1109/ACCESS.2019.2963724.

[55] Y. Alosefer, “Analysing web-based malware behaviour through client honeypots,” no. February, 2012, [Online]. Available: http://orca.cf.ac.uk/29469/.

[56] Z. Bazrafshan, H. Hashemi, S. Mehdi, H. Fard, and A. Hamzeh, “A Survey on Heuristic Malware Detection Techniques,” pp. 113–120, 2013.

[57] A. Detection, N. T. Mohammed, M. Hasan, and A. J. Rashid, “Review of Hybrid Analysis Technique for Malware Detection Review of Hybrid Analysis Technique for Malware Detection,” 2020, doi: 10.1088/1757-899X/769/1/012075.

[58] R. Sihwail, K. Omar, K. Akram, and Z. Ariffin, “An Effective Memory Analysis for Malware Detection and Classification An Effective Memory Analysis for Malware,” no. February, 2021, doi: 10.32604/cmc.2021.014510.

[59] F. Luis and G. Moncayo, *Learning Malware Analysis*. .

[60] O. R. I. Or-meir, N. I. R. Nissim, Y. Elovici, and L. Rokach, “Dynamic Malware Analysis in the Modern Era — A State of the Art Survey,” vol. 52, no. 5, 2019.

[61] M. Botacin, M. Zanata, D. Oliveira, and A. Grégio, “HEAVEN : A Hardware-Enhanced AntiVirus ENgine to accelerate real-time , signature-based malware detection,” *Expert Syst. Appl.*, vol. 201, no. January, p. 117083, 2022, doi: 10.1016/j.eswa.2022.117083.

[62] R. Masri and M. Aldwairi, “Automated Malicious Advertisement Detection using VirusTotal, URLVoid, and TrendMicro,” pp. 6–11, 2017.

[63] A. Salem, S. Banescu, and A. Pretschner, “Maat : Automatically Analyzing VirusTotal for Accurate Labeling and Effective Malware Detection,” vol. 24, no. 4, 2021.

[64] C. E. Shannon, “A mathematical theory of communication Bell Syst,” *tech. J*, vol. 27, no. 379, p. 623, 1948.

[65] V. Sethia, “Malware Capturing and Analysis using Dionaea Honeypot,” pp. 0–3, 2019.

[66] M. Malik, “A Framework for Collecting and Analysis PE Malware Using Modern Honey Network (MHN),” 2014.

[67] A. Kyriakou and N. Sklavos, “Container-Based Honeypot Deployment for the Analysis of Malicious Activity.”

[68] C. Moore, “Detecting Ransomware with Honeypot techniques.”

[69] B. Wang, Y. Dou, Y. Sang, Y. Zhang, and J. Huang, “IoTCMal : Towards A Hybrid IoT Honeypot for Capturing and Analyzing Malware,” 2020.

[70] S. Paul, A. K. Chakravarty, A. Raj, and A. S, “A study of signature-based and behaviour-based malware detection approaches,” *Int. J. Adv. Res. Ideas Innov. Technol.*, vol. 5, no. 3, pp. 1509–1511, 2019.

[71] H. Anand, “FILE-LESS MALWARE DETECTION,” 2022.

[72] H. D. Menendez, “Malware: The Never-Ending Arm Race,” *Open J. Cybersecurity*, pp. 1–25, 2021, doi: 10.46723/ojc.1.1.3.

[73] C. P. Patidar and H. Khandelwal, “Zero Day Attack Detection Using Machine Learning,” *Int. J. Res. Anal. Rev.*, vol. 6, no. 1, pp. 1364–1367, 2019.

[74] J. H. Faruk, H. Shahriar, M. Valero, F. L. Barsha, and S. Sobhan, “Malware Detection and Prevention using Artificial Intelligence Techniques,” pp. 5369–5377, 2021.

[75] S. Rani and A. Kumar, “Analysis of Anomaly detection of Malware using,” pp. 774–779, 2022.

[76] A. Katkar, S. Shukla, D. Shaikh, and P. Dange, “Malware Intrusion Detection For System Security,” pp. 5–9, 2021.

[77] T. A. Assegie, “An Optimized KNN Model for Signature-Based Malware Detection,” no. 2, pp. 46–49, 2021.

[78] C. Rohith and G. Kaur, “A Comprehensive Study on Malware Detection and Prevention Techniques used by Anti-Virus,” *Proc. 2021 2nd Int. Conf. Intell. Eng. Manag. ICIEM 2021*, pp. 429–434, 2021, doi: 10.1109/ICIEM51511.2021.9445322.

[79] S. M. L. Lima, “Artificial intelligence-based antivirus in order to detect malware preventively,” *Prog. Artif. Intell.*, vol. 10, no. 1, pp. 1–22, 2021, doi: 10.1007/s13748-

020-00220-4.

[80] J. Lyda, Robert; Hamrock, “Using Entropy Analysis to Find Encrypted and Packed Malware,” *IEEE Secur. Priv.*, 2007.

[81] M. Bat-Erdene, H. Park, H. Li, H. Lee, and M. S. Choi, “Entropy analysis to classify unknown packing algorithms for malware detection,” *Int. J. Inf. Secur.*, vol. 16, no. 3, pp. 227–248, 2017, doi: 10.1007/s10207-016-0330-4.

[82] K. Lee, S. Y. Lee, and K. Yim, “Machine Learning Based File Entropy Analysis for Ransomware Detection in Backup Systems,” *IEEE Access*, vol. 7, pp. 110205–110215, 2019, doi: 10.1109/ACCESS.2019.2931136.

[83] K. S. Han, J. H. Lim, B. Kang, and E. G. Im, “Malware analysis using visualized images and entropy graphs,” *Int. J. Inf. Secur.*, vol. 14, no. 1, pp. 1–14, 2015, doi: 10.1007/s10207-014-0242-0.

[84] J. Y. Paik, R. Jin, and E. S. Cho, “Malware classification using a byte-granularity feature based on structural entropy,” *Comput. Intell.*, vol. 38, no. 4, pp. 1536–1558, 2022, doi: 10.1111/coin.12521.

[85] Melike Başer; Ebu Yusuf Güven; Muhammed Ali Aydın, “SSH and Telnet Protocols Attack Analysis Using Honeypot Technique:* Analysis of SSH AND TELNET Honeypot,” 2021, vol. 7, doi: 10.1109/UBMK52708.2021.9558948.

EKLER

MALWARE CLASSIFICATION OF SUSPICIOUS FILES THAT ANTIVIRUS APPLICATIONS RESPONSES

Tablo Ek 1 Trojan sınıfına giren isimlendirmeler

Trojan
Trojan.CYGU-5
Trojan.GenericKD.37014839
Trojan.GenericKD.46282322 (B)
Linux.Trojan.Agent.WM6SLY
Trojan.GenericKD.36919466 (B)
Trojan.Generic.D233049B
Trojan:Linux/Multiverze
Trojan.Gen.NPE
E32/Trojan.OJUR-2
TROJ_GEN.R049C0PF82
Trojan/Generic.ASSuf.3CFBA
Trojan.Generic-Bash.Save.81cad571
Linux.Trojan.Agent.BYCO3E
Trojan.Linux.Agent
Trojan.Generic.D2C47968
Trojan.Generic-Bash.Save.b4766376
TrojanDownloader.Linux.w
Trojan-Downloader.Linux.Sh
Trojan.Generic.D2C23652
Linux.Trojan.Agent.U1BI7F
E32/Trojan.ADHC-11
Trojan.BDNM-12
Trojan.Malware.121218.susgen
Trojan.SuspectCRC
Unix.Trojan.DarkNexus-7679166-0
Trojan.Shell.Agent
Trojan.GenericKD.37014839 (B)
Trojan.ElfArm32.Dwn.ixvyeu
Trojan.Generic
Trojan.GenericKD.36890669 (B)
TrojanDownloader.Linux.yr
Trojan.GenericKD.46111158 (B)
Trojan.Gafgyt.Linux.28527

Trojan.GenericKD.46354653
Trojan.GenericKD.37044452 (B)
Elf.Trojan.A1686007
Trojan.GenericKD.46111158
Linux.Trojan.Agent.956L7O
Trojan.Generic.D2C3EBF0
Trojan.U.Downloader.oa
Trojan.Generic-Script.Save.ba695
Trojan.Shell.Agent.a!c
E32/Trojan.KSLF-8
Trojan.GenericKD.46373334 (B)
ELF.Trojan.42987.GC
TrojanDownloader.Linux.xc
Linux.DownLoader.598
E32/Trojan.BIKP-4
Archive.Trojan.Agent.237VFG
Trojan.GenericKD.46300839
Win32.Trojan-downloader.Agent.Ljke
Trojan.GenericKD.36946419
Trojan:Script/Woreflint.A!cl
Trojan.GenericKD.36897947 (B)
TROJ_GEN.R002C0DDR21
Trojan.Generic.D233C1F3
TROJ_FRS.VSNW17D21
Linux/TrojanDownloader.SH.AUQ
HEUR:Trojan-Downloader.Shell.Agent.bc
Trojan.U.Heur.oa
Trojan.GenericKD.46394352 (B)
Trojan.GenericKD.36946419 (B)
TrojanDDoS.Linux.nk
Trojan.GenericKD.46300839 (B)
Trojan.Dwn.ixwzdx
Trojan.Generic.D2C27EA7
Trojan.ElfArm32.Dwn.iyigiv
Trojan.GenericKD.37032770
HEUR:Trojan-Downloader.Shell.Agent.p
Trojan.GenericKD.46430568
TrojanDownloader.Linux.it
ELF.Trojan.43175.GC
Trojan.NQTO-2
Trojan.VSHD-13
Trojan.Generic-Bash.Save.2cd19841
Linux/TrojanDownloader.SH.AUZ
Trojan.GenericKD.37044452
TrojanDownloader.Linux.lb
Trojan.GenericKD.36991053

Trojan.GenericKD.36919466
TrojanDownloader.Linux.ln
Trojan.GenericKD.36991053 (B)
Trojan.ZIP.Linux.4!c
Artemis!Trojan
Trojan.GenericKD.37032770 (B)
Trojan.GenericKD.46394352
BehavesLike.VBS.Trojan.dv
Trojan.Shell.Agent.V (B)
Trojan.Shell.Agent.V
Trojan.Script.Bash.4!c
Trojan.Generic.D2351342
Trojan.Generic.D23358AA
Trojan.Generic.D2C350DD
Trojan:Win32/Casdet!rfn
Trojan.SH.MALKEY.AA
Linux.Trojan.Agent.X66BXS
Trojan.Dwn.ixwpzz
Trojan.GenericKD.46354653 (B)
Trojan.GenericKD.36897947
Elf.Trojan.A1667385
Trojan.Script.Shell.4!c
Trojan.GenericKD.36890669
HEUR:Trojan-Downloader.Shell.Agent.a
Trojan.GenericKD.46282322
Trojan.GenericKD.46430568 (B)

Tablo Ek 2 Mirai sınıfına giren isimlendirmeler

Mirai
LINUX/Dldr.Mirai.gours
HEUR:Trojan-Downloader.Linux.Mirai.d
ELF/Mirai.A!tr.dldr
Backdoor.Linux.MIRAIA.USNELFD21
Gen:NN.Mirai.34058
Linux/Mirai.Gen35
Generic.Bash.MiraiB.F5F3FD03(B)
LINUX/Mirai.bykao
Backdoor.Mirai/Linux!1.BAF6(CLASSIC)
Possible_MIRAIDLOD.SMLBAT5
Trojan.Linux.Mirai.K!c
Backdoor.Mirai/Linux!1.B311(CLASSIC)
UDS:Backdoor.Linux.Mirai
Linux/Mirai.D!tr.dldr

Linux.Trojan-downloader.Mirai.Wqmn
Backdoor.Linux.MiraiDownload.ve
Linux/Mirai.B!tr.bdr
ELF:MiraiDownloader-KG[Trj]
Linux.Mirai.2924
HEUR:Trojan-Downloader.Linux.Mirai.e
ELF/Mirai.D!tr
ELF:MiraiDownloader-JD[Trj]
Lnx/Mirai-FEBO!62CF685B889F
LINUX/Mirai.apkfl
Linux.Mirai.4511
Backdoor.Linux.MIRAIA.USNELF821
Gen:NN.Mirai.34104
avariantofLinux/TrojanDownloader.Mirai.W
Possible_MIRAIDLOD.SMLBAT6
Linux.Trojan-downloader.Mirai.Phhe
Generic.Bash.MiraiB.F5F3FD03
Trojan-Downloader.Linux.Mirai
Linux.Mirai.1669
ELF:Mirai-AHC[Trj]
Linux.Trojan-downloader.Mirai.Pctb
Generic.Bash.MiraiA.FADADCE2(B)
avariantofLinux/Mirai.A
Linux.Mirai
Generic.Bash.MiraiA.8CDBB6E7(B)
Linux.Trojan-Downloader.Mirai
Trojan.Linux.Mirai
HEUR:Backdoor.Linux.Mirai.b
Generic.Bash.MiraiA.C7D1F9AE(B)
Linux.Mirai.1758
Generic.Bash.MiraiA.D6E7BB1C
ELF:MiraiDownloader-BF[Drp]
Gen:NN.Mirai.34722
ELF:Mirai-FY[Trj]
Trojan.Elf32.Mirai.iusilp
ELF:MiraiDownloader-KC[Trj]
Trojan.Linux.Mirai.GED(B)
Generic.Bash.MiraiA.8CDBB6E7
avariantofLinux/Mirai.ATO
ELF:Mirai-PB[Trj]
HEUR:Backdoor.Linux.Mirai.r
Trojan.Elf32.Mirai.ivwzwh
LINUX/Dldr.Mirai.itcql
Generic.Bash.MiraiA.FADADCE2
ELF:Mirai-FS[Trj]
ELF/Mirai.ASX!tr

ELF/Mirai.A!tr
Backdoor.Linux.MIRAI.USELVEP21
Possible_MIRAIDLOD.SMLBR5
Linux.Trojan-downloader.Mirai.Hugd
Generic.Bash.MiraiA.95CFC17A
Generic.Bash.MiraiA.95CFC17A(B)
Unix.Trojan.Mirai-7100807-0
Backdoor.Linux.MIRAI.USELVDG21
Generic.Bash.MiraiA.C7D1F9AE
Generic.Bash.MiraiA.D6E7BB1C(B)
Linux.Trojan-downloader.Mirai.Eehg
E64/Mirai.A.gen!Camelot
Backdoor.Linux.Mirai.wz
LINUX/Mirai.glfer
Backdoor.Linux.Mirai.wan
avariantofLinux/TrojanDownloader.Mirai.A
LINUX/Dldr.Mirai.nmpob
Trojan.Linux.Mirai.GED
Generic.Bash.MiraiA.EB5DB9B6(B)
Generic.Bash.MiraiA.EB5DB9B6
LINUX/Dldr.Mirai.aymds

Tablo Ek 3 CoinMiner sınıfına giren isimlendirmeler

CoinMiner
Multios.Coinminer.Miner-6781728-2
E64/CoinMiner.B.gen!Camelot
Downloader/Shell.ElfMiner.S1165
Application.Linux.Miner.LL
Trojan.U.CoinMiner.oa
Linux/CoinMiner.Gen2
ELF/BitCoinMiner.HF!tr
not-a-virus:HEUR:RiskTool.Linux.BitCoinMiner.b
LINUX/BitCoinMiner.fmbfm
Linux.Risk.Bitcoinminer.Dwtp
a variant of Linux/CoinMiner.AV potentially unwanted
Linux.Risk.Bitcoinminer.Wvau
Tool.Linux.BtcMine.3120
Linux.Risk.Bitcoinminer.Hsit
Linux.Risk.Bitcoinminer.Dxcs
Trojan:Linux/CoinMiner.K
Trojan:Linux/CoinMiner.N!MTB
Linux.Risk.Bitcoinminer.Hyju
Riskware.Linux.BitCoinMiner.1!c

Coinminer.json.g
Linux.BtcMine.271
Trojan.GenericKD.46373334
Linux.Risk.Bitcoinminer.Wuhd
JSON.Miner.42626
Coinminer.Linux.MALXMR.PUWEMO
Linux.Risk.Bitcoinminer.Eclg
Win64/CoinMiner.RO potentially unwanted
HackTool.MinerCfg/JSON!1.CABA (CLASSIC)
Riskware.Elf64.BitCoinMiner.ivgbvp
Application.Linux.Miner.LL (B)

Tablo Ek 4 Backdoor sınıfına giren isimlendirmeler

Backdoor
Backdoor.Linux.rec
Backdoor.Linux.ZYX.USDSEED21
Backdoor.Linux.bskw
Backdoor.Linux.Agent
HEUR:Backdoor.Linux.Gafgyt.df
Backdoor/Text.CryptoBot
RDN/Generic BackDoor

Tablo Ek 5 Downloader sınıfına giren isimlendirmeler

Downloader
Shell/ElfDownloader.S3
Linux.DownLoader.535
Linux.DownLoader.533
BV:Downloader-JS [Drp]
TrojanDownloader.Linux.eh
Linux/Downloader.1288
RDN/Generic Downloader.x
Linux/Downloader.w
Linux.DownLoader.532
Linux.DownLoader.459
Linux.DownLoader.531
Linux.DownLoader.1833
Linux.DownLoader.517
Linux.DownLoader.1834
Linux.DownLoader.534
Linux.DownLoader.523
Linux/ShellDLoader.RMF!tr
BV:Downloader-AEH [Drp]

Linux.DownLoader.1915
Linux/Downloader.p
BV:Downloader-TJ [Drp]
Linux.DownLoader.37

Tablo Ek 6 Malware sınıfına giren isimlendirmeler

Malware
Malware@#1hds26rs9a7iq
Malware.Generic-Script.Save.ba303
Malware@#t8qixr26txeu
Mal/ShellDl-A
Malware@#z42j21fz2ven
Malware@#11tskun56nig
Malware.Generic-Script.Save.ba104
Other:Malware-gen [Trj]
Malware.ELF-Script.Save.b606d819
Malware@#3noqscxynq9q3

Tablo Ek 7 Other sınıfına giren isimlendirmeler

Other
GenericRXPJ-JL!B7740B0E0D53
GenericRXPL-SI!3B6C0968416D
Script.Application.Agent.ESB7CE
multiple detections
LINUX.Agent
Suspicious.Linux.Save.a
a variant of Generik.JHEILIY
PUA.Generic
Linux/DDoS-BIB
a variant of Linux/Gafgyt.AXI
GenericRXHV-CO!494F5DC3C97D
GenericRXID-MY!CEAE104121FB
Script.Application.Agent.BYSNFG
ex_virus.script.bash.000002
LINUX/Agent.SH.CDWC
GenericRXIA-AZ!3DC11C062A9F
LINUX/Gafgyt.quxje
BV:Agent-BHX [Drp]
GenericRXLU-EV!62CF685B889F
LINUX/Dldr.Agent.chw
GenericRXIA-VB!2CA7C03A8DCB
Linux.Lightaidra
PUP-XNS-AN

GenericRXHZ-TQ!F42C2BC71DC8
GenericRXIA-WK!D0B5FB9EA23F
a variant of Generik.GVFDFVN
Linux/DDoS-CIA
ELF:CVE-2017-17215-A [Expl]
PossibleThreat
Linux/Gafgyt.DF!tr.bdr
GenericRXIB-LH!FC4CD46E3F7C
HackTool.XMRMiner!1.C0AC (CLASSIC)
GenericRXIB-BH!F3630D8BF607
RDN/Generic.dx
GenericRXIB-FF!4CDE0578155B
RiskTool.Linux.clc
Static AI - Malicious ELF

Tablo Ek 8 Zararlı yazılım sınıflarının ve oylama sonucunun dağılımı

Dosya adı	md5	Pozitif	Negatif	Cevap_Yok	Oylama Sonucu	Trojan	Mirai	CoinMiner	Backdoor	Downloader	Malware	Other
46ffeb0d8a02b3e	8389e0831930a093b8fd67ddf067628d	29	30	5	CoinMiner	9	0	14	0	0	3	3
cb11b02ba39dfae	3b6c0968416d4a31e316dff1345e4366	14	47	3	Mirai	1	10	0	0	1	0	2
d20df25dbf2da55	9fb64b8b2ee489e18c4b97840d68bff0	30	28	6	CoinMiner	8	0	12	0	0	3	7
2191947a5542e55	7e11ec87bc3369e9fe72b30876588935	21	28	5	Mirai	4	13	0	0	3	0	1
51807c319bc68d8	ce1935cfb27f08b36aae681dda6136a8	29	29	6	CoinMiner	1	0	18	0	0	3	7
4b7285d072dd7f4	f42c2bc71dc85a1eb1ad02898d76a5e6	20	40	4	Mirai	7	10	0	0	0	0	3
e5d86015cc227dd	fc4cd46e3f7c010ea1bd9f86d490487a	14	47	3	Mirai	1	9	0	0	1	0	3
09c83cad76e68fb2	2b63dec4082101cc1d6cc358f6256371	32	25	7	Trojan	11	0	10	0	0	4	7
5b0ef68f823deaac	7962bd6235fbdc50f1bd31388f68a278	21	37	6	Mirai	6	8	0	0	4	3	0
f4f8168e4dee5b88	494f5dc3c97d4fc007a05c845e8a4752	14	46	4	Mirai	1	10	0	0	1	0	2
241719e57efbec80	58859640cf9efb2ac1764e2268af04f3	5	54	5	Trojan	2	0	0	0	0	1	2
baa6874682dadc45	390d1c889c593ffbd00ac546cffd0afc	30	27	7	CoinMiner	9	0	12	0	0	3	6
7b11d91298d65cf2	12172ed0b6e6d45e3a311fddd5b1e154	24	34	6	CoinMiner	8	0	8	0	0	3	5
3bf6fac29f3f460e	4cde0578155b88cd4bed078ec57d1060	19	41	4	Mirai	4	12	0	0	1	0	2
bb2ab2e3191df74	f3c44f30c58a16e66c13a6b8ca9d341b	10	51	3	Mirai	0	8	0	0	1	1	0
8dbee006d9b8c7c	3dc11c062a9fbde71d81d6f07a97abd8	16	42	6	Mirai	3	9	0	0	1	0	3
1ec237da28aad15	282ca84cddf592fa4051634f0bb96215	30	29	5	CoinMiner	10	0	11	0	0	4	5
9c1e5fbea05c1365	b3f10c5438932fdbebe83e8e5d10b3fc	13	49	2	Mirai	0	10	0	0	1	0	2
165b5ae81159a62	1fac57c62f9c33737474aac40805763b	32	25	7	CoinMiner	4	0	17	0	0	4	7
5d10cfa7344b8794	d0b5fb9ea23f18b89d57b5e17fc62d38	19	41	4	Mirai	4	11	0	0	2	0	2
cb0228d746e02ec3	4834f8d67cf3cb75048ae8e18f8ef7ef	9	52	3	Mirai	0	8	0	0	1	0	0
81cc93ba0d6d32e4	0a86f8aef65ae2abda178e9944589e90	0	59	5	-	0	0	0	0	0	0	0

addd0760d5c2d53f	af3dc82d08e25e0c37222312652eebd5	1	57	6	Trojan	1	0	0	0	0	0	0
a8460f446be54041	a420f7a60a40f3ff3a806a01feb1dfda	16	42	6	Trojan	11	0	1	1	0	3	0
b688cb360f9398f1	62cf685b889f34a1e60e8ae89f576458	39	23	2	Mirai	5	25	0	3	0	2	4
0153b3ac01db6e9	2ca7c03a8dcb245fe6d66f29e6725c19	13	47	4	Mirai	1	9	0	0	2	0	2
8762ac159dde051	c7c1a50219c06f9ebd875cbd4b204732	33	28	3	Mirai	11	14	0	1	0	1	6
e5b4786c5a4c615	562ebbe43a69718ac8f04ba955cea91b	27	32	5	CoinMiner	9	0	12	0	0	3	3
9ab1d4f8c8ec7e8d	de980a884c69ca47f68c51355bd97f1f	29	28	7	CoinMiner	1	0	19	0	0	4	5
61738c39f35d5ed3	b7740b0e0d53bf2dcded3c2efe8d211c	13	47	4	Mirai	0	10	0	0	1	0	2
c3a1344da7beefb4	584782de825e7dcacf46cf2b66347406	31	26	7	CoinMiner	9	0	13	0	0	4	5
ed20688ea7f70348	155dceb8560b50bc6f7fa6d50a103524	15	45	4	CoinMiner	2	0	8	0	0	2	3
36322e950d9000bd	6d5ccacf840f0a066d7a1c118cb956c2	32	25	7	Trojan	11	0	10	0	0	4	7
7e4904caa9efd47e	0880c240dfc4e5961d6c6def6f0d6e82	13	45	6	Mirai	0	9	0	0	2	2	0
37467f83f3190d3f4	813a5db3a1d88982b2a7449021a7b43e	29	28	7	CoinMiner	4	0	16	0	0	4	5
10e10e0037791994	3079b76cee6ade49eac560190d6b4a80	32	27	5	Trojan	14	9	0	0	0	3	6
ef8f83cf318308bb	dba9a0856b52cb469d3a49f891417b40	33	25	6	CoinMiner	8	0	16	0	0	4	5
50363e111bf51e55	ceae104121fb468f2de68d712a561171	14	46	4	Mirai	1	9	0	0	1	0	3
baed3518e5758ec3	7cd8c8bbe22ddc44c381f2489ceaa65a	0	59	5	-	0	0	0	0	0	0	0
434779509e49d54c	f3630d8bf607f6017ed071f1e016dee4	13	48	3	Mirai	1	9	0	0	1	0	2
bc4bfa51738e4350	aa3fc228aab8a4d673418b5e796246a8	18	39	7	Mirai	5	8	0	0	0	2	3
b1f10c3575cfda274	eb052c0af3431a9208904f0baca7c73c	21	37	6	Mirai	6	8	0	0	4	3	0
c183ce13572bf08d5	3484b80b33ee8d53336090d91ad31a6b	25	31	8	Mirai	8	10	0	0	5	1	1
d51e8e059bfbe2299	96c783862e92140f8a98b365795a62d8	30	28	6	CoinMiner	2	0	19	0	0	4	5
697d216fbd0134d61	45b6f1b54e3bbda4563085f802659273	11	50	3	Mirai	1	10	0	0	0	0	0
5679498b959d55ce	c4f881c4e4ce8c461549c4e8360d7d43	12	48	4	Mirai	2	9	0	0	1	0	0
b1bc0ff1f6b1f17a60	ae074e3b6d54454340304e3782933c4a	27	32	5	Mirai	6	8	1	1	6	2	3
14e6ddcd00d75ae6	2c163f15af33741ed4f44932d3fc498f	27	34	3	Trojan	9	6	0	5	0	3	4
94ec28743a21beee4	c8337513071da7946def1b953a185b8c	30	26	8	Trojan	10	0	9	0	0	4	7
3342f1ff4e972c406e	553a7386a66d54148c5c2fb2f7842bf7	23	36	5	Mirai	3	8	0	0	7	3	2

Tablo Ek 9 En iyi sonuç veren ve en popüler Antivirüs cevapları

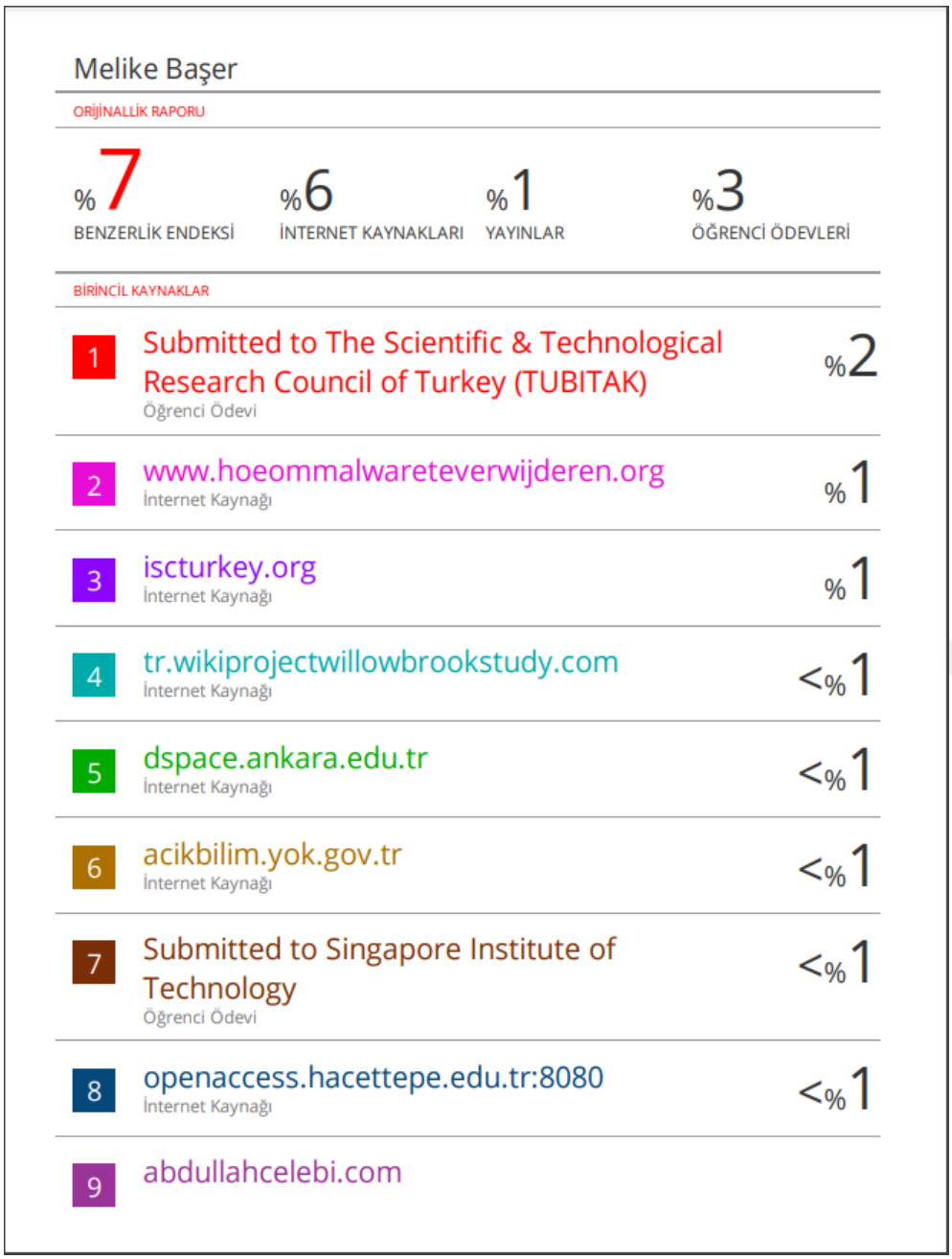
Dosya Adı	AVG	Avast	DrWeb	McAfee	Kaspersky
46ffeb0d8a02b3e	Other:Malware-gen [Trj]	negatif	Tool.Linux.BtcMine.3120	Coinminer.json.g	not-a-virus:HEUR:RiskTool.Linux.BitCoinMiner.b
cb11b02ba39dfae	ELF:Mirai-FS [Trj]	ELF:Mirai-FS [Trj]	Linux.DownLoader.523	GenericRXPL-SI!3B6C0968416D	HEUR:Trojan-Downloader.Linux.Mirai.d
d20df25dbf2da55	Other:Malware-gen [Trj]	negatif	Tool.Linux.BtcMine.3120	negatif	not-a-virus:HEUR:RiskTool.Linux.BitCoinMiner.b
2191947a5542e55	ELF:MiraiDownloader-KG [Trj]	ELF:MiraiDownloader-KG [Trj]	Linux.DownLoader.523	RDN/Generic Downloader.x	HEUR:Trojan-Downloader.Linux.Mirai.d
51807c319bc68d8	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Tool.Linux.BtcMine.3120	PUP-XNS-AN	not-a-virus:HEUR:RiskTool.Linux.BitCoinMiner.b
4b7285d072dd7f4	ELF:MiraiDownloader-JD [Trj]	ELF:MiraiDownloader-JD [Trj]	Linux.DownLoader.598	cevap_yok	HEUR:Trojan-Downloader.Linux.Mirai.d
e5d86015cc227dd	ELF:MiraiDownloader-BF [Drp]	ELF:MiraiDownloader-BF [Drp]	Linux.DownLoader.523	GenericRXIB-LH!FC4CD46E3F7C	HEUR:Trojan-Downloader.Linux.Mirai.d
09c83cad76e68fb2	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Tool.Linux.BtcMine.3120	PUP-XNS-AN	cevap_yok
5b0ef68f823deaac	BV:Downloader-AEH [Drp]	BV:Downloader-AEH [Drp]	negatif	Linux/Downloader.w	HEUR:Trojan-Downloader.Shell.Agent.bc
f4f8168e4dee5b88	ELF:MiraiDownloader-JD [Trj]	ELF:MiraiDownloader-JD [Trj]	Linux.DownLoader.534	GenericRXHV-CO!494F5DC3C97D	HEUR:Trojan-Downloader.Linux.Mirai.d
241719e57efbec80	BV:Agent-BHX [Drp]	BV:Agent-BHX [Drp]	negatif	negatif	HEUR:Trojan-Downloader.Shell.Agent.bc
baa6874682dadc45	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Tool.Linux.BtcMine.3120	PUP-XNS-AN	cevap_yok
7b11d91298d65cf2	Other:Malware-gen [Trj]	negatif	Tool.Linux.BtcMine.3120	negatif	negatif
3bf6fac29f3f460e	ELF:MiraiDownloader-KC [Trj]	ELF:MiraiDownloader-KC [Trj]	Linux.DownLoader.523	cevap_yok	HEUR:Trojan-Downloader.Linux.Mirai.d
bb2ab2e3191df74	ELF:Mirai-PB [Trj]	ELF:Mirai-PB [Trj]	Linux.DownLoader.532	negatif	HEUR:Trojan-Downloader.Linux.Mirai.d

8dbee006d9b8c7c	ELF:MiraiDownloader-JD [Trj]	ELF:MiraiDownloader-JD [Trj]	Linux.DownLoader.532	GenericRXIA-AZ!3DC11C062A9F	HEUR:Trojan-Downloader.Linux.Mirai.d
1ec237da28aad15	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Tool.Linux.BtcMine.3120	Coinminer.json.g	negatif
9c1e5fbea05c1365	ELF:MiraiDownloader-JD [Trj]	ELF:MiraiDownloader-JD [Trj]	Linux.DownLoader.531	negatif	HEUR:Trojan-Downloader.Linux.Mirai.d
165b5ae81159a62	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Tool.Linux.BtcMine.3120	PUP-XNS-AN	cevap_yok
5d10cfa7344b8794	ELF:MiraiDownloader-JD [Trj]	ELF:MiraiDownloader-JD [Trj]	Linux.DownLoader.535	cevap_yok	HEUR:Trojan-Downloader.Linux.Mirai.d
cb0228d746e02ec3	ELF:MiraiDownloader-JD [Trj]	ELF:MiraiDownloader-JD [Trj]	Linux.DownLoader.517	negatif	HEUR:Trojan-Downloader.Linux.Mirai.d
81cc93ba0d6d32e4	cevap_yok	negatif	negatif	negatif	negatif
addd0760d5c2d53f	cevap_yok	negatif	negatif	negatif	negatif
a8460f446be54041	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Linux.BtcMine.271	negatif	negatif
b688cb360f9398f1	ELF:Mirai-AHC [Trj]	ELF:Mirai-AHC [Trj]	Linux.Mirai.2924	GenericRXLU-EV!62CF685B889F	HEUR:Backdoor.Linux.Mirai.b
0153b3ac01db6e9	ELF:MiraiDownloader-JD [Trj]	ELF:MiraiDownloader-JD [Trj]	Linux.DownLoader.459	GenericRXIA-VB!2CA7C03A8DCB	HEUR:Trojan-Downloader.Linux.Mirai.d
8762ac159dde051	ELF:CVE-2017-17215-A [Expl]	ELF:CVE-2017-17215-A [Expl]	Linux.Mirai.1758	RDN/Generic.dx	HEUR:Backdoor.Linux.Mirai.b
e5b4786c5a4c615	Other:Malware-gen [Trj]	negatif	negatif	Coinminer.json.g	not-a-virus:HEUR:RiskTool.Linux.BitCoinMiner.b
9ab1d4f8c8ec7e8d	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Tool.Linux.BtcMine.3120	Coinminer.json.g	cevap_yok
61738c39f35d5ed3	ELF:MiraiDownloader-JD [Trj]	ELF:MiraiDownloader-JD [Trj]	Linux.DownLoader.533	GenericRXPJ-JL!B7740B0E0D53	HEUR:Trojan-Downloader.Linux.Mirai.d
c3a1344da7beefb4	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Tool.Linux.BtcMine.3120	Coinminer.json.g	cevap_yok
ed20688ea7f70348	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	negatif	PUP-XNS-AN	negatif

36322e950d9000bd	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Tool.Linux.BtcMine.3120	Coinminer.json.g	cevap_yok
7e4904caa9efd47e	BV:Downloader-AEH [Drp]	negatif	negatif	Linux/Downloader.w	UDS:Backdoor.Linux.Mirai
37467f83f3190d3f4	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Tool.Linux.BtcMine.3120	PUP-XNS-AN	cevap_yok
10e10e0037791994	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Linux.Mirai.4511	RDN/Generic.dx	HEUR:Backdoor.Linux.Mirai.b
ef8f83cf318308bb	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Tool.Linux.BtcMine.3120	Coinminer.json.g	not-a-virus:HEUR:RiskTool.Linux.BitCoinMiner.b
50363e111bf51e55	ELF:MiraiDownloader-JD [Trj]	ELF:MiraiDownloader-JD [Trj]	Linux.DownLoader.531	GenericRXID-MY!CEAE104121FB	HEUR:Trojan-Downloader.Linux.Mirai.d
baed3518e5758ec3	cevap_yok	negatif	negatif	negatif	negatif
434779509e49d54c	ELF:MiraiDownloader-JD [Trj]	ELF:MiraiDownloader-JD [Trj]	Linux.DownLoader.517	GenericRXIB-BH!F3630D8BF607	HEUR:Trojan-Downloader.Linux.Mirai.d
bc4bfa51738e4350	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	negatif	negatif	HEUR:Trojan-Downloader.Shell.Agent.a
b1f10c3575cfda274	BV:Downloader-AEH [Drp]	BV:Downloader-AEH [Drp]	negatif	Linux/Downloader.w	HEUR:Trojan-Downloader.Shell.Agent.bc
c183ce13572bf08d5	BV:Downloader-TJ [Drp]	BV:Downloader-TJ [Drp]	Linux.DownLoader.37	Linux/Downloader.p	HEUR:Trojan-Downloader.Shell.Agent.p
d51e8e059bfbe2299	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Tool.Linux.BtcMine.3120	PUP-XNS-AN	cevap_yok
697d216fbd0134d61	ELF:MiraiDownloader-JD [Trj]	ELF:MiraiDownloader-JD [Trj]	Linux.DownLoader.598	negatif	HEUR:Trojan-Downloader.Linux.Mirai.d
5679498b959d55ce	ELF:MiraiDownloader-JD [Trj]	ELF:MiraiDownloader-JD [Trj]	Linux.DownLoader.1915	negatif	HEUR:Trojan-Downloader.Linux.Mirai.d
b1bc0ff1f6b1f17a60	BV:Downloader-AEH [Drp]	BV:Downloader-AEH [Drp]	Linux.DownLoader.1833	Linux/Downloader.p	HEUR:Trojan-Downloader.Shell.Agent.p
14e6ddcd00d75ae6	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Linux.Mirai.1669	RDN/Generic BackDoor	HEUR:Backdoor.Linux.Gafgyt.df

94ec28743a21beee4	Other:Malware-gen [Trj]	Other:Malware-gen [Trj]	Tool.Linux.BtcMine.3120	PUP-XNS-AN	cevap_yok
3342f1ff4e972c406e	BV:Downloader-JS [Drp]	BV:Downloader-JS [Drp]	Linux.DownLoader.1834	Linux/Downloader.w	HEUR:Trojan-Downloader.Shell.Agent.p

İNTİHAL RAPORU İLK SAYFASI



Melike Başer

ORİJİNALLİK RAPORU

%7	%6	%1	%3
BENZERLİK ENDEKSİ	İNTERNET KAYNAKLARI	YAYINLAR	ÖĞRENCİ ÖDEVLERİ

BİRİNCİL KAYNAKLAR

1	Submitted to The Scientific & Technological Research Council of Turkey (TUBITAK) Öğrenci Ödevi	%2
2	www.hoeommalwareteverwijderen.org İnternet Kaynağı	%1
3	iscturkey.org İnternet Kaynağı	%1
4	tr.wikiprojectwillowbrookstudy.com İnternet Kaynağı	<%1
5	dspace.ankara.edu.tr İnternet Kaynağı	<%1
6	acikbilim.yok.gov.tr İnternet Kaynağı	<%1
7	Submitted to Singapore Institute of Technology Öğrenci Ödevi	<%1
8	openaccess.hacettepe.edu.tr:8080 İnternet Kaynağı	<%1
9	abdullahcelebi.com	

KURUM İZNİ YAZILARI

Uyarı: Canlı ve cansız deneklerle yapılan tüm çalışmalar için kurum izin belgelerinin eklenmesi zorunludur. Gizlilik ve mahremiyet içeren durumlarda kurum adı kapatılmalıdır.

☐ **Kurum izni gerekmektedir.**

☒ **Kurum izni gerekmemektedir.**

Öğrenci Adı SOYADI
(İmza)