ISTANBUL TECHNICAL UNIVERSITY ★ INFORMATICS INSTITUTE

THE RANSOMWARE DETECTION AND PREVENTION TOOL DESIGN BY USING SIGNATURE AND ANOMALY BASED DETECTION METHODS

M.Sc. THESIS

Barış ÇELİKTAŞ

Department of Applied Informatics

Applied Informatics Programme

MAY 2018

ISTANBUL TECHNICAL UNIVERSITY ★ INFORMATICS INSTITUTE

THE RANSOMWARE DETECTION AND PREVENTION TOOL DESIGN BY USING SIGNATURE AND ANOMALY BASED DETECTION METHODS

M.Sc. THESIS

Barış ÇELİKTAŞ
(708161004)

Department of Applied Informatics

Applied Informatics Programme

Thesis Advisor: Prof. Dr. Ertuğrul KARAÇUHA
Thesis Co-advisor: Asst. Prof. Dr. Nafiz ÜNLÜ

MAY 2018

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

DAVRANIŞSAL VE STATİK ANALİZ YÖNTEMLERİNİ KULLANARAK FİDYE YAZILIMI TESPİT VE ÖNLEME ARACI TASARIMI

YÜKSEK LİSANS TEZİ

Barış ÇELİKTAŞ
(708161004)

Bilişim Uygulamaları Anabilim Dalı

Bilişim Uygulamaları Programı

Tez Danışmanı: Prof. Dr. Ertuğrul KARAÇUHA
Eş Danışman: Dr.Öğr.Ü. Nafiz ÜNLÜ

MAYIS 2018

Baris-CELIKTAS, a M.Sc. student of ITU Informatics Institute student ID 708161004, successfully defended the thesis/dissertation entitled "THE RANSOMWARE DETECTION AND PREVENTION TOOL DESIGN BY USING SIGNATURE AND ANOMALY BASED DETECTION METHODS", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Ertuğrul KARAÇUHA** İstanbul Technical University	....
Co-advisor :	**Asst. Prof. Dr. Nafiz ÜNLÜ** İstanbul Technical University	....
Jury Members :	**Prof. Dr. İbrahim SOĞUKPINAR** Gebze Technical University	....
	Assoc. Prof. Dr. M. Oğuzhan KÜLEKÇİ İstanbul Technical University	....
	Assoc. Prof. Dr. Enver ÖZDEMİR İstanbul Technical University	....

Date of Submission : 04 May 2018
Date of Defense : 29 May 2018

To all martyrs, veterans and their families,

FOREWORD

My master education at the Istanbul Technical University comes to an end with this thesis. I feel obliged to thank those whom I can credit for my success. First, I would like to thank the Istanbul Technical University staff, the Institute of Informatics and the Department of Computer and Informatics Engineering faculty staff. It was a great experience to spend my time with these remarkable people. Second, I would like to thank to my advisors, Prof.Dr.Ertuğrul KARAÇUHA and Asst.Prof.Dr.Nafiz ÜNLÜ, for their guidance and feedback on my thesis, and valuable help on academic issues. I devote this thesis to our martyrs who sacrifice their lives for their lands.

June 2018

Barış ÇELİKTAŞ
(IT Network & Security Administrator)

TABLE OF CONTENTS

Page

ABBREVIATIONS

AES	: Advanced Encryption Standard
APT	: Advanced Persistence Threat
BIOS	: Basic Input-Output System
C&C	: Command and Control
CAD	: Computer Aided Design
DGA	: Domain Generation Algorithm
DNS	: Domain Name Server
ECDH	: Elliptic-Curve Diffie-Hellman
EFS	: Encrypted File System
GUI	: Graphical User Interface
HKCU	: Hkey_Current_User
HKLM	: Hkey_Local_Machine
HTML	: HyperText Markup Language
HTTPs	: HyperText Transfer Protocol Secure
I2P	: Invisible Internet Project
IDS	: Intrusion Detection System
IPS	: Intrusion Prevention System
ISP	: Internet Service Provider
IT	: Information Technology
MBR	: Master Boot Record
MFT	: Managed File Transfer
NSA	: National Security Agency
OS	: Operating System
PID	: Process Identification
RAM	: Random Access Memory
RC4	: Rivest Cipher 4
RSA	: Rivest-Shamir-Adleman
SMB	: Server Message Block
TCP/IP	: Transmission Control Protocol / Internet Protocol
TTL	: Time To Live
UML	: Unified Modeling Language
VSC	: Volume Shadow Copy
VSS	: Volume Shadow Copy Service

LIST OF TABLES

LIST OF FIGURES

THE RANSOMWARE DETECTION AND PREVENTION TOOL DESIGN BY USING SIGNATURE AND ANOMALY BASED DETECTION METHODS

SUMMARY

Ransomware, which constantly improves by updating itself and transferring to the network and computing environment, is the most common type of malware used by the attackers recently. Ransomware demands ransom from the user for decrypting the encrypted files. As a result of the payment of the desired amount of ransom, the files can be opened with the decryption key delivered to the user.

Various antivirus software using signature-based detection method fail to detect the malware because they perform analysis via hash signature samples in databases. Because hash signature samples of zero-day attacks are not recorded in antivirus software databases, detecting ransomware by using anomaly-based detection method is more effective.

The most important factor for anomaly-based detection method is to be able to analyze the data sufficiently to determine the ability and the behavior of the ransomware. This method consists of two phases. These are “training/learning” and “monitoring/detecting” phases. In training phase, many academic publications, international cyber security vendors’ reports on ransomware, and interview with cyber security experts have been examined and utilized during bringing out our study, and consequently, the attack vectors of the ransomware, the core features, the identification methods and the movements based on the Windows OSs have been found. That is, machine learning techniques have not been used, and so all materials have been evaluated as inputs to the design of our tool.

In our study, after presenting the behavior of the ransomware in detail, in the monitoring/detecting phase, how the ransomware detection and prevention tool should be created to detect and prevent ransomware on Windows OSs will be explained. It is also evident that the tool in the hybrid structure using signature-based detection method, along with anomaly-based detection method, will be even more successful in detecting and preventing ransomware with minimum false-positive rate and minimum file loss.

We consider that we can give a better perspective to the users, software developers, and security administrators about the key features of the ransomware detection and prevention tool that can be used as a solution that does not require high fees for commercial software to protect against ransomware, and we can more easily take measures against ransomware.

In our study, the code written in the Python programming language of the ransomware detection and prevention tool proposed in the Appendices of our study is also presented. This code, which is open to development, is thought to be an example of the design of a commercial ransomware detection and prevention tool and maybe a guide to future academic studies on ransomware and other malicious software.

İMZA VE ANOMALİ TABANLI TESPİT YÖNTEMLERİNİ KULLANARAK FİDYE YAZILIMI TESPİT VE ÖNLEME ARACI TASARIMI

ÖZET

Bilgisayarınızın ekranında “tüm dosyalarınız şifreleme algoritmaları kullanılarak şifrelenmiştir, şifreli dosyalarınıza erişebilmek için belirtilen süre içerisinde fidye ödemelisiniz, ödemezseniz dosyalarınıza bir daha erişemeyeceksiniz” benzeri bir mesaj çıkıyor ise, dosyalarınız büyük olasılıkla güçlü şifreleme algoritmaları kullanılarak saldırgan tarafından fidye yazılımlarından biri kullanılarak şifrelenmiştir.

Sürekli kendini geliştirip güncelleyerek ağ ve bilgisayar ortamına aktarılan fidye yazılımları, saldırganlar tarafından en çok kullanılan zararlı yazılım çeşididir. Fidye yazılımları kullanıcıların dosyalarını güçlü kriptografik algoritmalar kullanarak şifrelemekte ve dosyaların açılabilmesi için de kullanıcıdan fidye istemektedir. İstenilen fidye miktarının ödenmesi sonucunda da dosyalar sadece kullanıcıya teslim edilen özel anahtar ile açılabilmektedir.

Günümüzde birçok bilgisayarları hedef alan fidye yazılımları, saldırganlar tarafından yaygın olarak kullanılmakta ve her geçen gün de ailelerine yenileri eklenmektedir. Bu tip saldırılarda en çok kullanılan atak vektörleri, sosyal mühendislik tekniklerinin de kullanıldığı spam veya oltalama e-postalarıdır. Fidye yazılımları genellikle insan müdahalesi olmadan ağ üzerinde yayıldığından, bu tehdide maruz kalan mağdur sayısı da çok fazladır. Fidye yazılımlarına hedef olan dosya tipleri ise sürekli olarak değişmekte ve çeşitlilik arz etmektedir. Şu ana kadar fidye yazılımları saldırılarının hedef aldığı 70’den fazla dosya tipi bulunmaktadır.

Zaman geçtikçe saldırganların kullandığı yöntemlerde birçok değişiklik meydana gelmiş, tehdidin boyutu ve etkisi de buna paralel bir şekilde artmıştır. Nihayetinde günümüzdeki yeni nesil fidye yazılımları halini almışlardır. Örneğin, yeni nesil fidye yazılımları hedef bilgisayardaki dosyaları şifrelerken hem simetrik hem de asimetrik şifreleme yöntemlerinin birlikte kullanıldığı hibrit şifreleme mekanizmasını tercih etmektedir.

Saldırganın atlama noktaları üzerinden hedefine ulaşırken kendi konumunu gizlemesi sonucunda tespitinin çok zor hale gelmesi, istenen fidyenin ödenmesi için benzersiz ödeme mekanizması sağlayan Bitcoin kripto parasının kullanılması, şifrelenen dosyaların en çok kullanılan dosyalardan oluşması ve karşılığında kullanıcı veya kurumun fidye ödemek zorunda bırakılması fidye yazılımlarının diğer zararlı yazılımlara tercih edilmesine sebep olmuştur.

Günümüze kadar fidye yazılımları ile şifrelenen dosyaların geri açılabilmesi için istenen fidye çeşitli yöntemlerle saldırgana ödenmiştir. Günümüzde ise genellikle saldırgan açısından kolaylık ve güvenlik sağlayan de facto standart haline gelen Bitcoin kripto parası kullanılmaktadır.

Zararlı yazılımlar ile ilgili istatistiksel verilere baktığımızda fidye yazılımı tehdidinin hem kullanıcı bilgisayarlarında hem de organizasyonlarda ne kadar büyük maddi hasara sebep olduğu görülmektedir. 2017 yılı verilerine göre, zararlı yazılımlar içerisinde fidye yazılımlarının oranı yaklaşık yüzde 60 iken, meydana getirdiği zarar ise yaklaşık 5 milyar dolardır.

Bilgisayarlarda kullanılan birçok antivirüs ve güvenlik yazılımları, veritabanlarındaki hash imza örnekleri ile statik analiz işlemi yapmaktadır. Bu sebeple sadece fidye yazılımları değil diğer zararlı yazılımların tespitinde de başarısız olmaktadırlar.

Fidye yazılımları genellikle imza tabanlı tespit yöntemini kullanan antivirüs ve güvenlik yazılımlarını atlatabilmek için geliştiricileri tarafından sürekli yenilenmektedir. Dolayısıyla her geçen gün fidye yazılımları çok daha karmaşık ve tehlikeli bir hal almaktadır.

Özellikle sıfırıncı gün ataklarının imza hash örnekleri antivirüs veritabanlarına kaydedilmemiş olduğundan, fidye yazılımının anomali tabanlı tespit yöntemini de kullanılarak tespit edilmesi büyük önem arz etmektedir. Anomali tabanlı tespit yapılabilmesi için en önemli zaruri faktör ise zararlının yeteneğini ve davranışlarını tespit edebilecek kadar veriyi analiz edebilmektir. Anomali tabanlı tespit yöntemi iki aşamadan oluşmaktadır. Bunlar “eğitim/öğrenme” ve “monitör/tespit” aşamalarıdır. Eğitim aşamasında birçok akademik yayınlardan ve önde gelen uluslararası siber güvenlik firmalarının raporlarından faydalanılmıştır. Ayrıntılı literatür taraması ve Kali Linux, Windows 7 ve Windows 10 makineler üzerinde yapılan testler sonucu en temel fidye yazılımlarının davranışları öğrenilmiştir. Başka bir ifadeyle, makine öğrenmesi teknikleri kullanılmamış, aracımızın tasarlanması için fidye yazılımlarının olağan ve anormal davranışları bir çok kaynaktan elde edilmiş ve monitör aşamasında girdi olarak kullanılmıştır.

Fidye yazılımlarının Windows işletim sistemlerine sahip makinelerde nasıl tespit edilebileceği ve engellenebileceği çalışmamızda yer almaktadır. Önerdiğimiz fidye yazılımı tespit ve önleme aracımız Windows işletim sistemindeki servislerin, işlemlerin davranışları ile kütük kayıtlarını kontrol edecek ve fidye yazılımının dosya sistemi üzerindeki başlatmaya çalıştığı şifreleme işlemlerini sürekli olarak takip edecektir. Ayrıca yeni nesil fidye yazılımlarının kullandığı yöntem olan asimetrik şifreleme anahtarlarını sakladığı C&C sunucusu ile bağlantı kurup kurmadığı, ağ trafiği analizi ve bilgisayara indirilen dosyaların statik olarak imza tabanlı kontrolünü de yapacaktır.

Çalışmamızda fidye yazılımlarının olağan davranışları tespit edilerek bu davranışların meydana geldiği zaman şifreleme işlemi başlamadan önce engelleyen fidye yazılımı tespit ve önleme aracının oluşturulması amaçlanmıştır. Bu aracı oluştururken hem anomali tabanlı hem de imza tabanlı tespit yöntemleri kullanılmıştır. Tespit yöntemleri altında da statik, dinamik ve hibrit olmak üzere üç farklı teknik kullanılmaktadır. Statik tespit tekniği zararlı çalıştırılmadan önce engellenmesini, dinamik tespit tekniği zararlının çalışıyorken veya sonrasında engellenmesini, hibrit tespit tekniği ise zararlının sürekli olarak kontrolünü kapsamaktadır. Aracımızın hibrit tespit tekniğini kullanarak, hem imza tabanlı hem de anomali tabanlı tespit metotlarını kullanıyor olması, fidye yazılımı tespit ve önleme amaçlı kullanılan diğer araçlara olan üstünlükleridir.

İmza tabanlı tespit yöntemi yanı sıra anomali tabanlı tespit yöntemini de kullanan hibrit bir yapıdaki fidye yazılımı tespit ve önleme aracının fidye yazılımlarını, minimum yanlış alarm oranı ile en az dosya kaybı yaşanacak şekilde tespit etme ve önleme konusunda çok daha başarılı olacağı da aşikârdır.

Fidye yazılımı tespit ve önleme aracımız, imza tabanlı ve anomali tabanlı tespit aşamalarında bir anormallik tespit ettiğinde kullanıcıya erken bir uyarı sağlayacak ve fidye yazılımının çalışmasını engelleyecektir. Ayrıca aracımız anomali tabanlı tespit aşaması sayesinde sadece bilinen fidye yazılımlarına karşı değil aynı zamanda daha önce hiç var olmamış sıfırıncı gün fidye yazılımlarına karşı da koruma sağlayacaktır.

Çalışmamızın temel amacı, fidye yazılımlarına karşı korunmak için ticari yazılımlara yüksek ücretler ödemeye gerek duymayan bir çözüm olarak kullanılabilecek bir fidye yazılımı tespit ve önleme aracının temel özellikleri ile ilgili kullanıcılara, yazılım geliştiricilere ve güvenlik yöneticilerine bakış açısı kazandırmak ve öneriler sunabilmektir. Bu sebeple de çalışmamızın Ek'inde önerdiğimiz fidye yazılımı tespit ve önleme aracının python programlama dilinde yazılmış kodu da sunulmaktadır. Geliştirmeye açık olan bu kodun, ticari bir fidye yazılımı tespit ve önleme aracının tasarlanmasında örnek teşkil edeceği ve fidye yazılımları ile ilgili ileride hazırlanacak akademik çalışmalara da yol gösterici olacağı düşünülmektedir.

1. INTRODUCTION

Ransomware families occupy a very important place in malware families. Many users, institutions, and organizations have been exposed to ransomware threat, resulting in major financial and reputation loss. So, it is important to design a tool that detects and prevents ransomware efficiently with the minimum false positive and lower costs.

1.1 Subject and Importance

Your files are likely to have been encrypted by the attacker using strong encryption algorithms, if the message like "all your files have been encrypted by using encryption algorithms, you have to pay the ransom within the specified period to access your encrypted files, otherwise you will not be able to access your files again" is seen on your computer’s screen.

Ransomware has been widely used to target the computers recently, and new types are being constantly added to the family. Ransomware reaches the target computer using social engineering techniques and activates after the victim downloads the e-mail attachments or clicks on insecure links. Since ransomware spreads over the network easily, the number of victims exposed to this threat is very high.

Since ransomware hides its location once it reaches to the target through jump points, detecting an attack becomes extremely difficult. The ransomware encrypts the most commonly used files in the target computer; therefore, individual users or even institutions are forced to pay a ransom for decryption. Attackers nowadays prefer Bitcoin, an anonymous payment mechanism, as a means for payment, thereby concealing their identity and location. These peculiarities encourage attackers to choose ransomware, compared to other malware, in their malicious attacks [1, 2].

It will be seen how important to protect against ransomware considering that in the first half of 2017 the average ransom value per infected machine is $544, 42% of organizations have exposed to ransomware and the total amount of new ransomware is about 700,000 [1, 3].

In our study, we present the basic characteristics of the anti-ransomware tool, which detects and prevents ransomware on machines with Windows OSs. Our study utilized many academic publications, the reports of leading international cybersecurity companies, and interview with many cybersecurity experts. Based on the findings from the comprehensive review of related literature and tests performed on Kali Linux machine, Windows 7, and Windows 10, behavioral characteristics of the most basic ransomware are analyzed and then how ransomware can be detected and prevented is described.

1.2 Purpose and Content

The primary purpose of the thesis is to give a point of view and provide suggestions for users, software developers, and security administrators about the basic features of the ransomware detection and prevention tool that can be used as a solution that does not require high fees for commercial software to protect against ransomware.

The basic contributions of this thesis to the literature are as follows:

- We will present the ransomware detection and prevention tool that tries to be effective in Windows OSs with minimum false positive alarms with a hybrid mechanism, which uses both static and behavioral analysis techniques together.
- We expect it to be the ransomware detection and prevention solution that detects and prevents ransomware and an example in the design of a commercial ransomware detection and prevention tool.
- We also consider that this thesis will be a guide for the academic studies on ransomware.

The rest of the study is as follows. Section II defines and provides an overview about ransomware, its statistics, and families, Section III explains the anatomy of ransomware on target machines, Section IV examines the modeling of ransomware detection techniques and methods, Section V describes the modeling of our tool and its implementation, and Section VI concludes with future directions and recommendations.

2 THEORY

2.1 Ransomware

Ransomware is a type of malware that encrypts files on the target computer using strong cryptographic algorithms. The ransomware, after completing the encryption process, requests ransom from the victim with the information note opened on the target computer. The victim cannot open their files without paying the requested ransom, since the private key to open the encrypted files is usually stored in the attacker's C&C server. So, the actual owners of the files become unable to use his/her files, and even if he or she has already paid the desired ransom, unfortunately, the users are left at the mercy of the attacker [1, 4].

2.2 Locker-Ransomware and Crypto-Ransomware

There are two basic types of ransomware. The first is the Locker Ransomware. This type prevents the computer's OS and applications from running and even opens the computer to prevent the user from performing normal operations. The second is the Crypto Ransomware. This ransomware encrypts the files in the computer's disk through the functions of the computer's OS are running smoothly [5, 6, 7, 8].

In summary, Crypto Ransomware demands a ransom to decrypt encrypted files on the disk, whereas Locker Ransomware demands ransom by restricting or blocking access to computer resources [7, 8, 9].

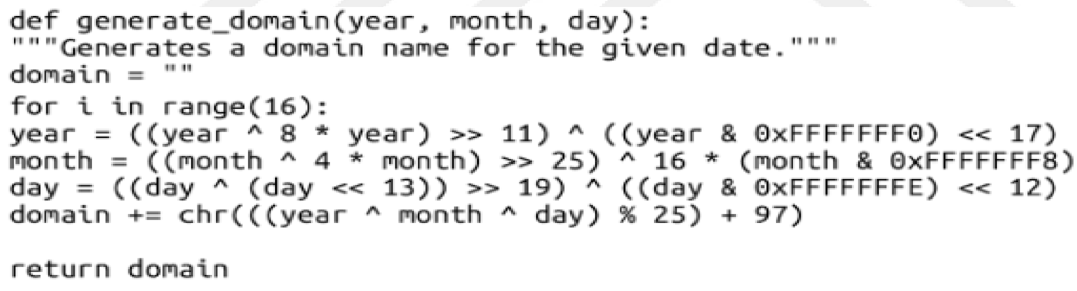
2.3 Attack Vectors

The most common attack vectors being used in ransomware attacks are spam and phishing e-mails using social engineering techniques. A ransomware is generally activated by opening the e-mail containing the malicious macro and loading to the computer an executable file of the script running with the activation of macro which is in the file [1, 10, 11]. In addition, the methods of opening the add-ons sent by spam emails, running fake software downloaded with the impression that it comes from a so-called trusted source, obtaining administrator authority and providing permanent access are among the other attack vectors [12, 13].

2.4 C&C Servers

It is very important for the attacker to ensure confidentiality and keep the location of the C&C servers undetectable [11, 14]. Thus, the new generation ransomware attacks are made via encrypted HTTPs protocol or the TOR network. The use of fully encrypted channels such as TOR and HTTPs makes the creation of the signature of the attack in advance impossible [15]. It should also be noted that the TOR network is not used for attacks of large scale because it can usually be detected by IPS and IDS devices. [12].

Additionally, C&C servers use the Dynamic DNS service, which allows a different IP address assignment by the ISP to the domain after a certain TTL value. The main reason for the Dynamic DNS service to be effective is the use of the DGA of which an example shown in Figure 2.1 [11, 15].



```
def generate_domain(year, month, day):
"""Generates a domain name for the given date."""
domain = ""
for i in range(16):
year = ((year ^ 8 * year) >> 11) ^ ((year & 0xFFFFFFF0) << 17)
month = ((month ^ 4 * month) >> 25) ^ 16 * (month & 0xFFFFFFF8)
day = ((day ^ (day << 13)) >> 19) ^ ((day & 0xFFFFFFFE) << 12)
domain += chr(((year ^ month ^ day) % 25) + 97)

return domain
```

Figure 2.1: An example of CryptoLocker's original DGA [11, 15].

2.5 Target File Types

File types targeted by ransomware constantly change and vary. The attacker determines target file types before the attack. For instance, a ransomware aimed at a large corporation targets the databases, financial and CAD files. In addition, more than 70 file types have become standard targets in ransomware attacks [1]. Although the files encrypted are the music, video and source codes in general, documents and images are actually used most [12].

2.6 Payment Analysis

Up till recently, victims paid the required ransom for the decryption of encrypted files with various methods such as via mail checks, wire transfers, and payment vouchers. However, attackers have recently preferred Bitcoin cryptocurrency, which is the de facto standard and more convenient and secure for attackers [12].

Bitcoin payment transactions cannot be recovered, one out of five people who pay the requested ransom to the attacker could not get their files back, as a result, the situation is at mercy of the attacker [16]. Amount of ransom demanded have changed over the years, and in the first half of 2017, the average ransom amount per infected machine was $544 [1]. The average ransom amounts according to years shown in Figure 2.2.

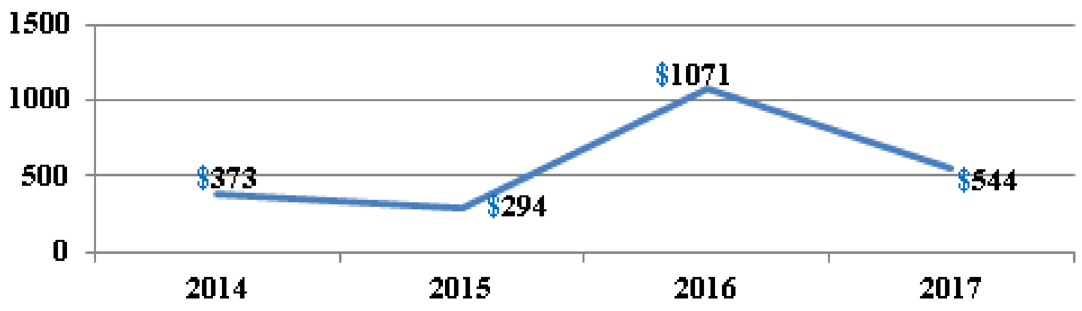
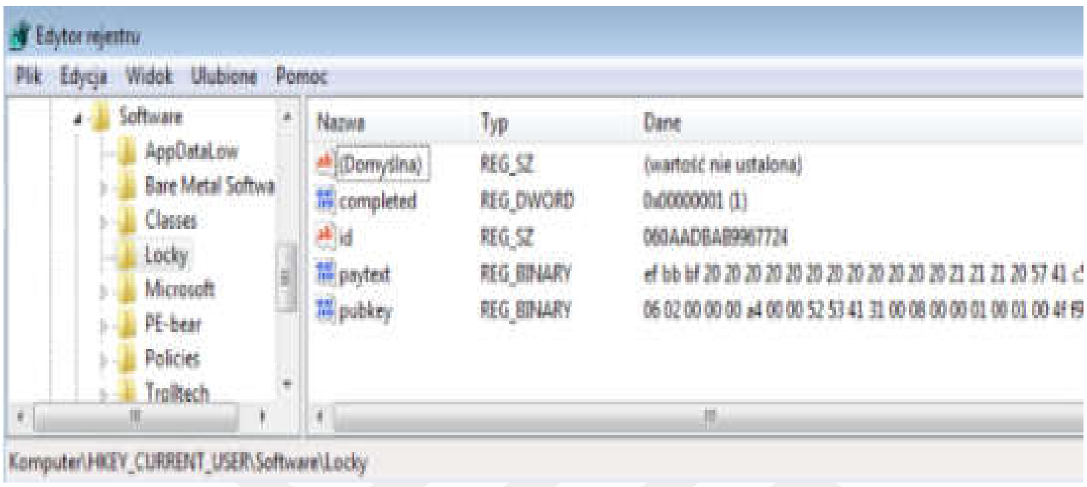


Figure 2.2: An average ransom amount per year (USD).

In early 2017, about 3400 web pages belonging to the customers of a South Korean web hosting company, which was exposed to ransomware, became inoperable and the attacker demanded a ransom worth about $1.62 million. Within a few days, it was agreed on a ransom payment of about $1 million and this amount has been recorded as the highest reported ransom payment so far [1]. Although the amount of ransom required for small organizations or a single machine is not large, the amount of ransom required for large-scale organizations, as seen in our example, can reach huge amounts due to the excess amount of machines affected.

2.7 Encryption Methods

The ransomware use encryption algorithms that vary from RC4 to RSA+AES and ECDH+AES. While the attackers used only symmetric encryption methods with the advent of the ransomware threat, they now prefer the hybrid encryption mechanism, which utilizes both symmetric and asymmetric encryption methods [12, 17]. In the new generation ransomware attacks, only the public key is sent to the infected machine, as shown in Figure 2.3, from the cryptographic key pair created on the C&C server, and the private key is never let out from the server. Thus, after encryption on machines exposed to today's ransomware attacks is completely ended, the private key stored in the C&C server is needed to open the files [17].



Nazwa	Typ	Dane
(Domyślna)	REG_SZ	(wartość nie ustalona)
completed	REG_DWORD	0x00000001 (1)
id	REG_SZ	060AADBAB9967724
paytext	REG_BINARY	ef bb bf 20 20 20 20 20 20 20 20 20 20 20 20 20 21 21 21 20 57 41 c
pubkey	REG_BINARY	06 02 00 00 00 a4 00 00 52 53 41 31 00 08 00 00 01 00 01 00 4f f9

Komputer\HKEY_CURRENT_USER\Software\Locky

Figure 2.3: The image of Locky ransomware in registry [28].

The files on the infected machine are usually encrypted with AES, which is one of the symmetric encryption methods. The symmetric key used for encryption is re-encrypted usually with the RSA public key which is one of the asymmetric encryption methods so that encrypted files cannot be easily decrypted later [11, 18].

As seen, the power of ransomware is directly related to the encryption algorithm used, and so in new generation ransomware, the hybrid encryption mechanism is used which is carried out in three following basic phases [12, 18].

1st Phase - From Attacker to Victim: The attacker produces an asymmetric pair of keys and places these keys into the ransomware in advance. Ransomware is also activated or opened when loaded to victim's computer by using one of the attack vectors.

2nd Phase - From Victim to Attacker: With the symmetric key in the ransomware, the files on the target computer begin to be encrypted. Then the symmetric key is also encrypted with the public key generated by the C&C server. As a result, a small asymmetric ciphertext is created beside the symmetric ciphertext of the data. The symmetric key is deleted so that the original data on the computer cannot be restored. The information of target computer to be encrypted and the key are kept on the C&C

server. After the files are encrypted, the user has a pop-up message with asymmetric cipher-text and an information note about how to make the ransom payment. The victim sends the cryptocurrency he has purchased to the attacker with the asymmetric ciphertext to be able to decrypt his files.

3rd Phase - From Attacker to Victim: Once the attacker has received payment confirmation, it decrypts the asymmetric ciphertext with the private key and sends the symmetric key to the victim. Consequently, the victim decrypts the encrypted data with the private key and accesses his encrypted files on the computer.

This process happens so fast that the encryption of the files in the target machine usually takes less than three minutes [19].

2.8 Ransomware Statistics

Ransomware causes millions of dollars loss for users, organizations, and institutions. Gradually, ransomware attacks are becoming increasingly threatening, and the number of ransomware is increasing as well [9]. In other words, ransomware families are quickly improving, and they are renewed and used by more experienced hacker groups in their attacks [15].

When we examine the statistics of the year 2016 on ransomware, one company per 40 seconds and one user per 10 seconds is exposed to ransomware attacks and one out of five people who pay the ransom attacker cannot get their files back [16].

When we examine the statistics of the year 2017 on ransomware, it is seen that about 42% of organizations in the first half of the year are exposed to ransomware, a four-fold increase in the range of ransomware according to the first quarter of the previous year and ransomware has been detected in about one in five malicious emails [16, 20]. The number of newly released ransomware is about 700,000 [3]. 71% of companies exposed to ransomware have been vulnerable to this threat and in the same year, ransomware has caused a loss of about $5 billion [1]. In addition, as shown in Figure 2.4, ransomware accounts for about 64% of all malicious software [20].

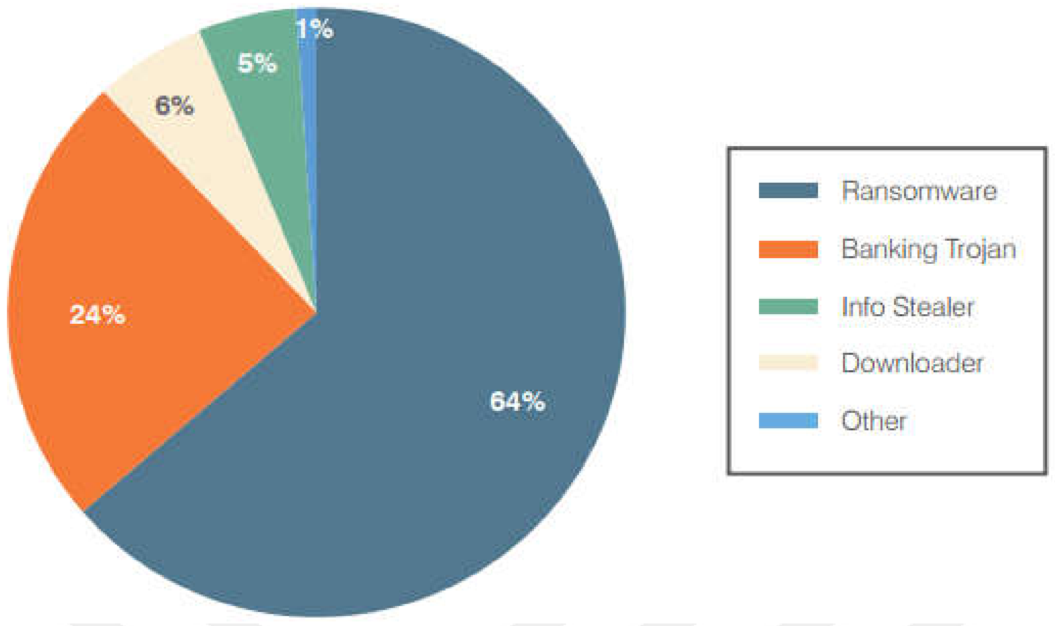


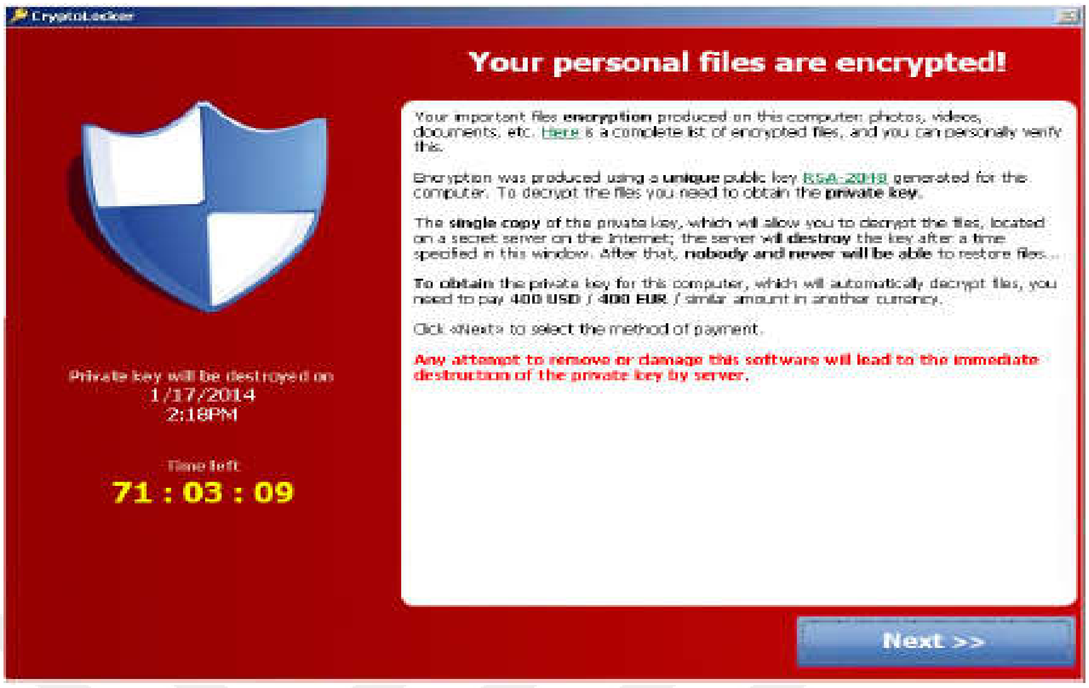
Figure 2.4: Malware by category, Q3 2017 [20].

2.9 Ransomware Families

Modern ransomware attacks are thought to have begun with the Gpcoder that emerged in 2005. In this attack, certain files were encrypted and the ransom was demanded from the victim [14, 22]. Then, many changes occurred in the methods used, the dimensions and effects of the threat increased. Ultimately, those turned into today's new generation of ransomware. In this part of our study, we will discuss the features of seven most commonly known ransomware in the previous five years, e.g. the Cryptolocker, CryptoWall, Cerber, Locky, WannaCry, Petya and Bad Rabbit.

2.9.1 Cryptolocker

Cryptolocker launched in September 2013 with a widespread attack, which started the history of modern ransomware [9, 23]. It managed to influence over 500,000 machines [12]. In just one month, the attack generated over $34,000 in revenue [24]. It often affected systems via spam e-mails. After users clicked on the link, the ransomware started scanning the network devices, modifying the names of the files and folders, and encrypting them using the RSA asymmetric algorithm [8, 23]. A pop-up message sent by Cryptolocker is shown in Figure 2.5.



Your personal files are encrypted!

Your important files **encryption** produced on this computer: photos, videos, documents, etc. Here is a complete list of encrypted files, and you can personally verify this.

Encryption was produced using a **unique** public key RSA-2048 generated for this computer. To decrypt the files you need to obtain the **private key**.

The **single copy** of the private key, which will allow you to decrypt the files, located on a secret server on the Internet; the server will **destroy** the key after a time specified in this window. After that, **nobody and never will be able** to restore files...

To obtain the private key for this computer, which will automatically decrypt files, you need to pay **400 USD / 400 EUR** / similar amount in another currency.

Click «Next» to select the method of payment.

Any attempt to remove or damage this software will lead to the immediate destruction of the private key by server.

Figure 2.5: A pop-up message sent by Cryptolocker [5].

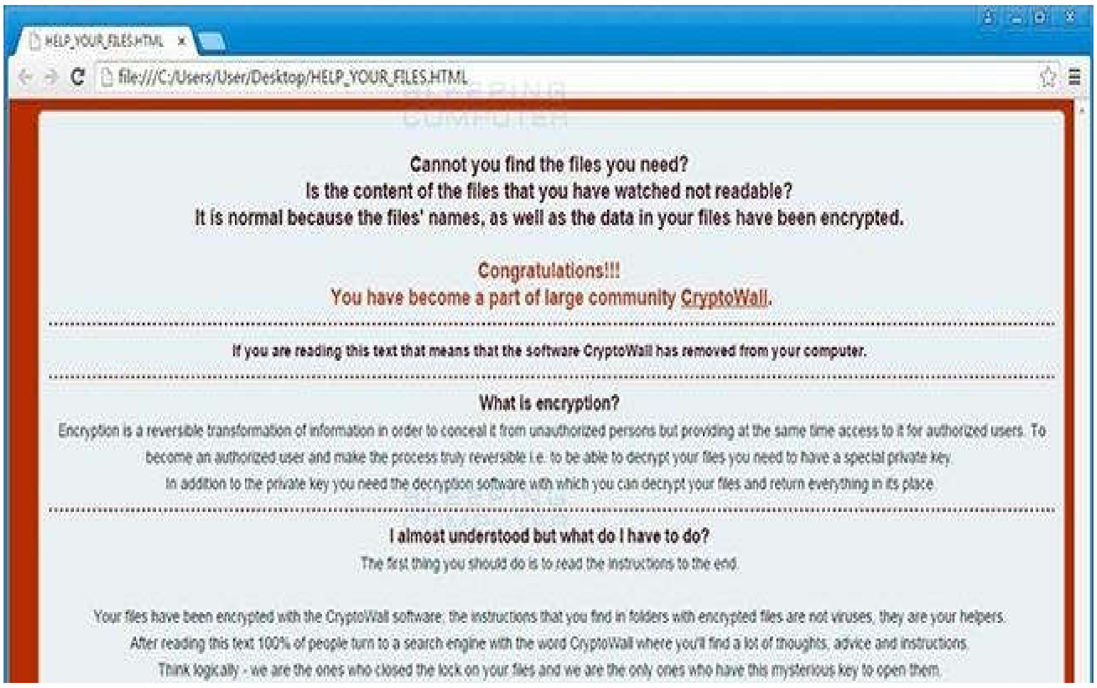
CryptoLocker 2.0, which is the newer and improved version and written with C# programming language appeared in December 2013. It has generally used Tor network for anonymity, 2048-bit encryption mechanism for extortion and Bitcoin for payment methods [24]. It has not been detected by any security software so far [8].

Although it could not sustain its presence because of the symmetric keys which are placed on C&C servers, it opened the way for new generation ransomware with the help of its source code. The most popular one is CryptoWall, which accounts for more than half of all ransomware by 2015 [23].

2.9.2 CryptoWall

First appeared in November 2013, CryptoWall is still one of the most widely used ransomware [25]. In just six months, it infected more than 600,000 machines and encrypted about 5.25 billion files. Initially, it demands about $500 ransom and after seven days about $1,000. Payments are accepted in Bitcoins [26]. 0.27% of the victims paid a total of $1,101,900 as ransom. While two-thirds of the victims paid $500, the remainder paid from $200 to $10,000 as a ransom [8, 16]. It generally uses TOR network for the payment of ransom [9]. Since the end of 2014, several versions have come into the market. It is a dangerous ransomware with its persistence and property of process injection. By making each file name encrypted unique, it is difficult to identify the danger [12]. A pop-up message sent by CryptoWall is shown in Figure 2.6.

In January 2015, CryptoWall 2.0 came into the market. This version uses a unique bitcoin address for each victim and safely deletes unencrypted original files [26]. To hide the location of C&C server, it uses TOR network as well[8].



Cannot you find the files you need?
Is the content of the files that you have watched not readable?
It is normal because the files' names, as well as the data in your files have been encrypted.

Congratulations!!!
You have become a part of large community CryptoWall.

If you are reading this text that means that the software CryptoWall has removed from your computer.

What is encryption?
Encryption is a reversible transformation of information in order to conceal it from unauthorized persons but providing at the same time access to it for authorized users. To become an authorized user and make the process truly reversible i.e. to be able to decrypt your files you need to have a special private key.
In addition to the private key you need the decryption software with which you can decrypt your files and return everything in its place.

I almost understood but what do I have to do?
The first thing you should do is to read the instructions to the end.

Your files have been encrypted with the CryptoWall software; the instructions that you find in folders with encrypted files are not viruses, they are your helpers.
After reading this text 100% of people turn to a search engine with the word CryptoWall where you'll find a lot of thoughts, advice and instructions.
Think logically - we are the ones who closed the lock on your files and we are the only ones who have this mysterious key to open them.

Figure 2.6: A pop-up message sent by CryptoWall [26].

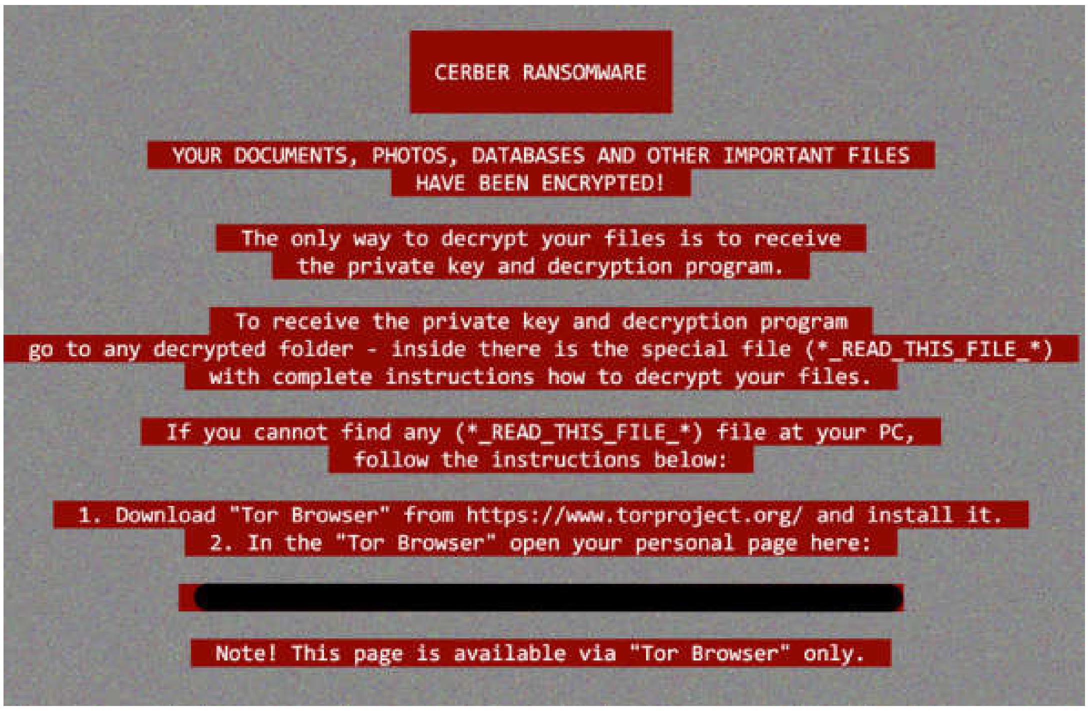
In March 2015, ransomware developers released a new version of named CryptoWall 3.0 [26]. It uses I2P for anonymity [8].

In September 2015, CryptoWall 4.0 came into the market. Compared to the previous versions, there are some significant changes in this version such as encrypted file names, a more powerful VSC deletion method, a new type of ransom note, new payment mechanisms, and redesigned HTML ransom note [8, 26]. Unlike other versions, this version is still unbroken [11].

2.9.3 Cerber

Cerber appeared in the first half of 2015. Files are encrypted with the ".cerber" extension and pop-ups are shown in HTML format [9]. It creates new unique samples every fifteen seconds using code obfuscation techniques [27]. It leaves not only the ransom note but also an audio record. So, it is the first ransomware that is interacted with the victim [11].

It is thought to be of Russian origin [11]. It creates a fake Windows system warning on infected machines and reboots the system before it starts encrypting the files [25]. Cerber's C&C servers can work even if it is offline [18]. A pop-up message sent by Cerber is shown in Figure 2.7.



CERBER RANSOMWARE

YOUR DOCUMENTS, PHOTOS, DATABASES AND OTHER IMPORTANT FILES
HAVE BEEN ENCRYPTED!

The only way to decrypt your files is to receive
the private key and decryption program.

To receive the private key and decryption program
go to any decrypted folder - inside there is the special file (*_READ_THIS_FILE_*)
with complete instructions how to decrypt your files.

If you cannot find any (*_READ_THIS_FILE_*) file at your PC,
follow the instructions below:

1. Download "Tor Browser" from https://www.torproject.org/ and install it.
2. In the "Tor Browser" open your personal page here:

[redacted]

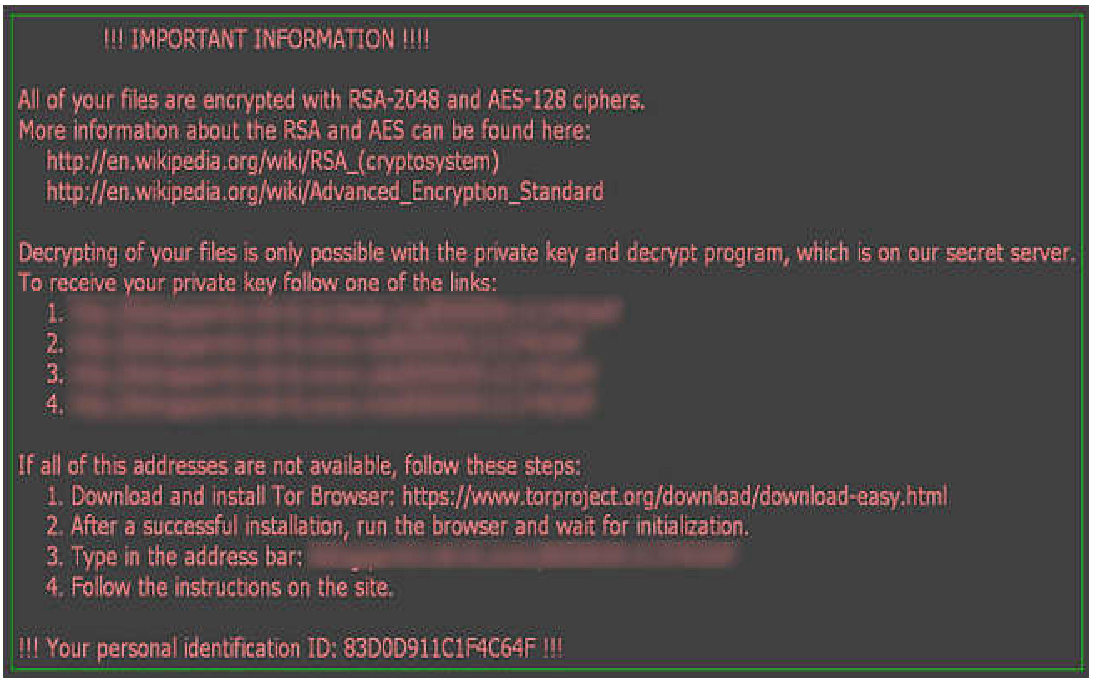
Note! This page is available via "Tor Browser" only.

Figure 2.7: A pop-up message sent by Cerber [21].

2.9.4 Locky

Locky, first seen in February 2016, is a complicated and sophisticated ransomware that uses malicious macros in a Word document and reaches the target computer via spam e-mails [11]. After the victim activates macros, the macro will start downloading executable file of Locky. The oldest versions of Locky ransomware used a phishing method as an attack vector [9]. Only in the first days of its emergence, Locky infected about 100,000 computers [7]. Locky encrypts not only the files on the computer but also the VSCs using Volume Shadow Copy Service (VSS) to prevent the user's files being restored. It encrypts VSCs and all files it can find, including system files [25]. Locky also encrypts external hard drives, database files, all network resources, and Bitcoin wallet to force the victim to pay the ransom [7].

It tries to escalate its privileges by stealing user credentials. It targets also virtual machines. Encrypted files are renamed as follows: File names are preceded by the victim's unique ID, followed by the file's ID and the ".locky" extension [28, 29]. So, Locky has a good engineering design, and it is very clever and ruthless. A pop-up message sent by Locky is shown in Figure 2.8.



```
        !!! IMPORTANT INFORMATION !!!!

All of your files are encrypted with RSA-2048 and AES-128 ciphers.
More information about the RSA and AES can be found here:
    http://en.wikipedia.org/wiki/RSA_(cryptosystem)
    http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

Decrypting of your files is only possible with the private key and decrypt program, which is on our secret server.
To receive your private key follow one of the links:
    1. [redacted]
    2. [redacted]
    3. [redacted]
    4. [redacted]

If all of this addresses are not available, follow these steps:
    1. Download and install Tor Browser: https://www.torproject.org/download/download-easy.html
    2. After a successful installation, run the browser and wait for initialization.
    3. Type in the address bar: [redacted]
    4. Follow the instructions on the site.

!!! Your personal identification ID: 83D0D911C1F4C64F !!!
```

Figure 2.8: A pop-up message sent by Locky [1].

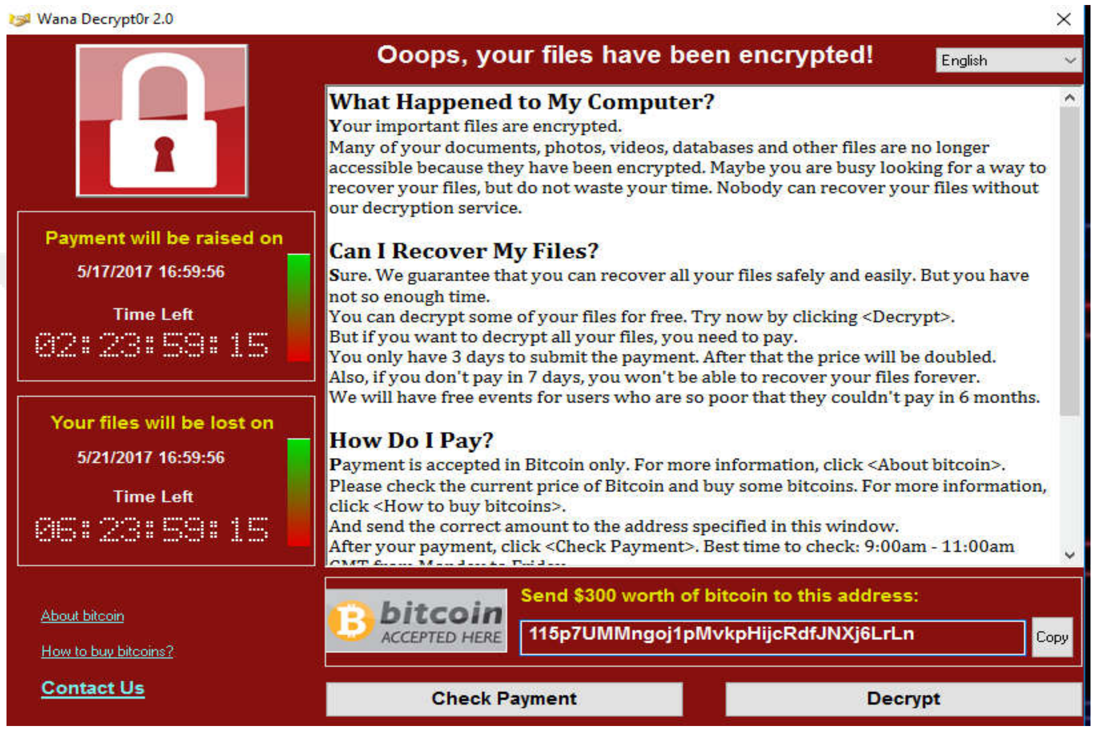
2.9.5 WannaCry/Wanna decrypt0r

WannaCry or Wanna Decryptor began to spread on May 12, 2017, by means of exploit vector named EternalBlue which was developed by the US National Security Agency (NSA) by exploiting the gap in the Microsoft's Server Message Block (SMB) protocol [25, 30]. It has spread to about 150 countries and 300,000 computers, which has been the largest ransomware attack in history. The demanded ransom, written in 28 different languages, is about $300 [12].

The widespread use of Windows XP OSs has resulted in the greatest impact on Russia and India. The update, published by Microsoft as a critical patch on March 14, 2017, has not been used by many organizations. So it has allowed WannaCry to spread very quickly over the internet [22, 25].

There are three basic attack vectors for WannaCry. These are accessing computers directly, executing unchecked e-mail attachments, and malware downloaded by victims from web pages [31].

WannaCry uses RSA-2048 as an encryption method. Like some other ransomware, it uses VSS to delete VSCs of files and TOR network to hide the C&C server [18]. A pop-up message sent by WannaCry/Wanna Decrypt0r is shown in Figure 2.9.


Wana Decrypt0r 2.0

Ooops, your files have been encrypted!

English

What Happened to My Computer?
Your important files are encrypted.
Many of your documents, photos, videos, databases and other files are no longer accessible because they have been encrypted. Maybe you are busy looking for a way to recover your files, but do not waste your time. Nobody can recover your files without our decryption service.

Can I Recover My Files?
Sure. We guarantee that you can recover all your files safely and easily. But you have not so enough time.
You can decrypt some of your files for free. Try now by clicking <Decrypt>.
But if you want to decrypt all your files, you need to pay.
You only have 3 days to submit the payment. After that the price will be doubled.
Also, if you don't pay in 7 days, you won't be able to recover your files forever.
We will have free events for users who are so poor that they couldn't pay in 6 months.

How Do I Pay?
Payment is accepted in Bitcoin only. For more information, click <About bitcoin>.
Please check the current price of Bitcoin and buy some bitcoins. For more information, click <How to buy bitcoins>.
And send the correct amount to the address specified in this window.
After your payment, click <Check Payment>. Best time to check: 9:00am - 11:00am

Payment will be raised on
5/17/2017 16:59:56
Time Left
02:23:59:15

Your files will be lost on
5/21/2017 16:59:56
Time Left
06:23:59:15

Send $300 worth of bitcoin to this address:
115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn

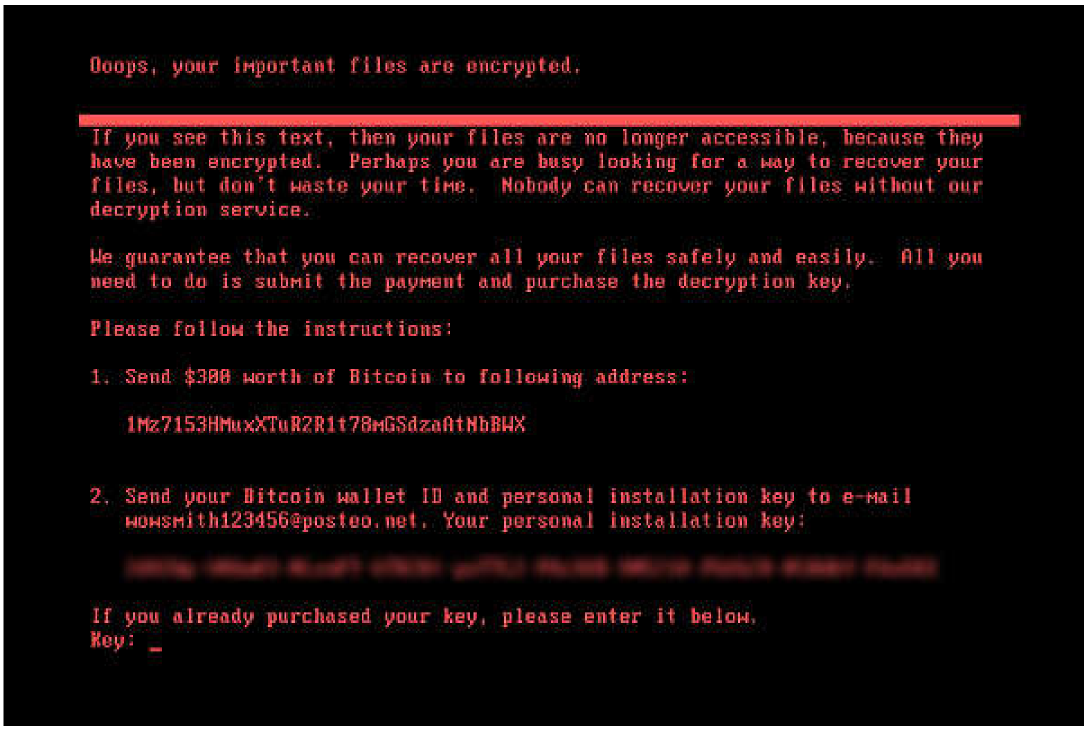
Figure 2.9: A pop-up message sent by WannaCry/Wanna Decrypt0r [22].

2.9.6 Petya

When the computer turns on, the BIOS perform some initial tests and leaves control to the MBR in the first sector of the hard disk where the boot loader is located. Petya, which first emerged in April 2016, causes a full blue screen of death crash by overwriting the Master Boot Record (MBR). Petya is also coded to delete the randomly generated unique key used to encrypt the Master File Table (MFT) [9, 25].

When the victim opens the computer, a ransom note is appeared on the computer screen. Petya benefits from "EternalBlue" exploit, which targets SMB v1 [32]. A pop-up message sent by Petya is shown in Figure 2.10.

Ransomware has gained a new approach with Petya. Petya is the first example of such ransomware with encrypting the discs, files, and preventing the computer from turns on [16]. In addition, Petya can work without contacting the C&C server, that is, independently from the internet [18].



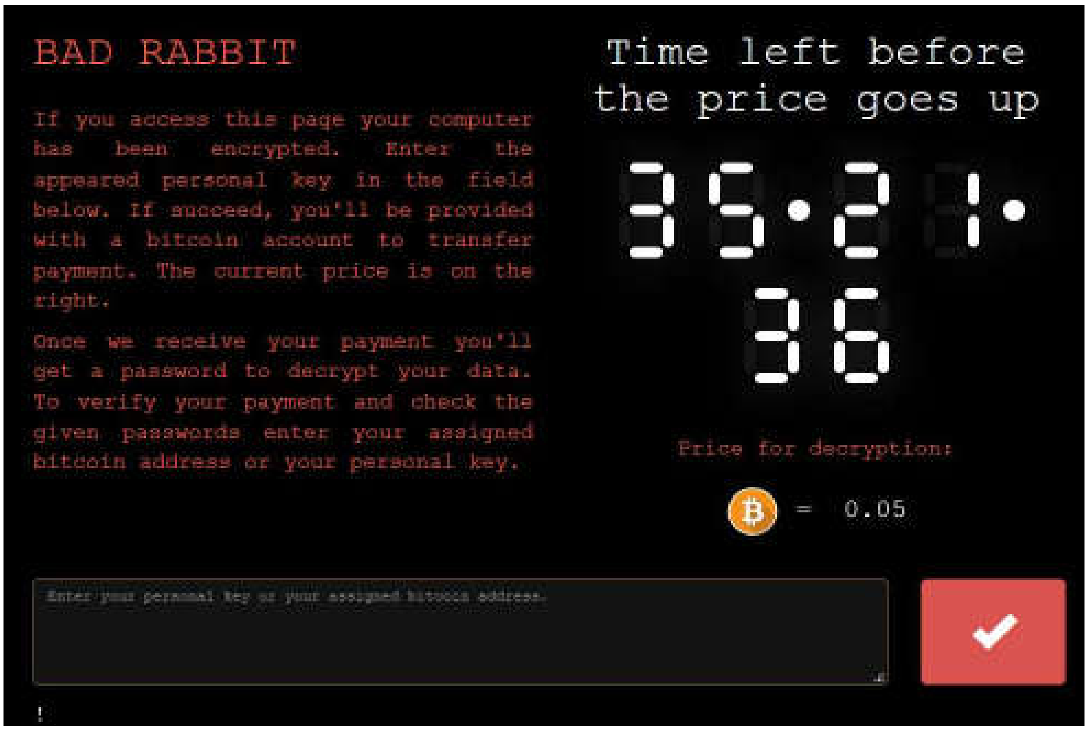
```
Ooops, your important files are encrypted.

If you see this text, then your files are no longer accessible, because they
have been encrypted.  Perhaps you are busy looking for a way to recover your
files, but don't waste your time.  Nobody can recover your files without our
decryption service.

We guarantee that you can recover all your files safely and easily.  All you
need to do is submit the payment and purchase the decryption key.

Please follow the instructions:

1. Send $300 worth of Bitcoin to following address:

   1Mz7153HMuxXTuR2R1t78mGSdzaAtNbBWX


2. Send your Bitcoin wallet ID and personal installation key to e-mail
   wowsmith123456@posteo.net. Your personal installation key:

   [redacted]

If you already purchased your key, please enter it below.
Key: _
```

Figure 2.10: A pop-up message sent by Petya [32].

2.9.7 Bad rabbit

In September 2017, Bad Rabbit infected computers worldwide by showing itself as an Adobe Flash Player update patch. Bad Rabbit has spread over internal and external networks via SMB service as in WannaCry and Petya, by using the information obtained from the infected computer and username and password information defined in itself. It adds the ".encrypted" extension to the end of the files it encrypts [33]. The demanded ransom is about $280 [25].

Considering the fact that BadRabbit has spread very quickly in the network and usernames and passwords used are not complicated, we will also see how weak usernames and passwords used in our systems [34]. A pop-up message sent by Bad Rabbit is shown in Figure 2.11.



BAD RABBIT

If you access this page your computer has been encrypted. Enter the appeared personal key in the field below. If succeed, you'll be provided with a bitcoin account to transfer payment. The current price is on the right.

Once we receive your payment you'll get a password to decrypt your data. To verify your payment and check the given passwords enter your assigned bitcoin address or your personal key.

Time left before the price goes up
35:21:36
Price for decryption:
= 0.05
Enter your personal key or your assigned bitcoin address.

Figure 2.11: A pop-up message sent by Bad Rabbit [33].

3 RANSOMWARE'S ANATOMY ON TARGET MACHINES

3.1 An Anatomy of Ransomware Attacks

Ransomware attacks take place in five phases. These phases are Deployment, Installation, C&C server, Destruction and Extortion [11]. They are shown in Figure 3.1.

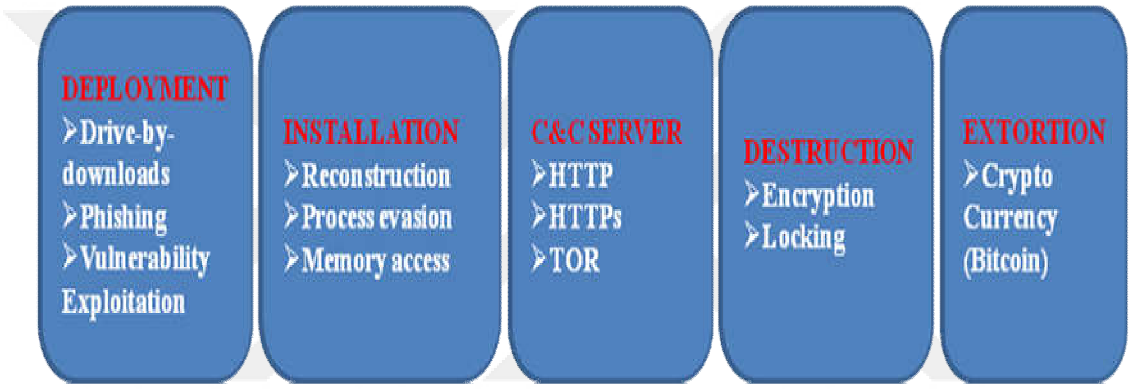


Figure 3.1: An anatomy of ransomware attacks [8, 15].

3.1.1 Deployment

In this phase, ransomware installs the components used to infect, encrypt, or lock the victim's machine. There are a few different methods such as drive-by download, phishing e-mails and exploiting vulnerabilities in internet-accessible systems [11].

3.1.2 Installation

When a malicious payload is delivered to the victim system, the infection process begins. A small piece of malicious code evades from detection and communicates with C&C servers. Ransomware downloads itself from C&C server for infection on the compromised system. Then it installs itself on the victim's machine. It will set keys in the Windows registry that ensure the malicious code starts up every time victim's machine is turned on. In addition, ransomware wants to spread throughout all network. It often disguised itself as a standard Windows process such as "explorer.exe" or "svchost.exe". It will assign itself unique computer name to ensure the C&C server knows which machine compromised. In this phase, default Windows protections can be disabled, which could include turning off shadow copy features on files and volumes, turning off system recovery features using "bcdedit.exe" etc. [8, 11].

3.1.3 C&C Server

When the malicious code is deployed and installed, it will begin to reach out C&C servers for instructions. These instructions include everything from which types of files will be encrypted, how long they should wait to begin the process and so on. Some ransomware also reports system information such as IP address, domain name, OS, anti-malware products to C&C communication servers. C&C channels can be unencrypted HTTP, encrypted HTTPs or anonymous TOR. TOR makes it even more complex and difficult to trace the exact location of the attackers. Also in his phase, the key exchange occurs, and the private key is held by the C&C servers whereas the public key is delivered to the victim system [8, 11].

3.1.4 Destruction

In this phase, all designated files will begin encrypted by the malicious code. These file types include all forms of Microsoft Office documents to JPGs, GIFs, CAD etc. [8, 11].

3.1.5 Extortion

After the files have been encrypted, the victims are shown a pop-up message that describes how they have been infected, how their files encrypted, which algorithms are used for encryption, how they can pay the ransom, what will happen to their files if they do not pay the ransom [8, 11].

3.2 Ransomware's Behavior Analysis on Target Machine

Most of the ransomware is written in the C/C++ programming language and it is hard to mention that they have code similarities. In addition, the ransomware is constantly being updated by developers to overcome static detection methods [2, 35]. So, the ransomware is gradually becoming more complex and dangerous [9, 12].

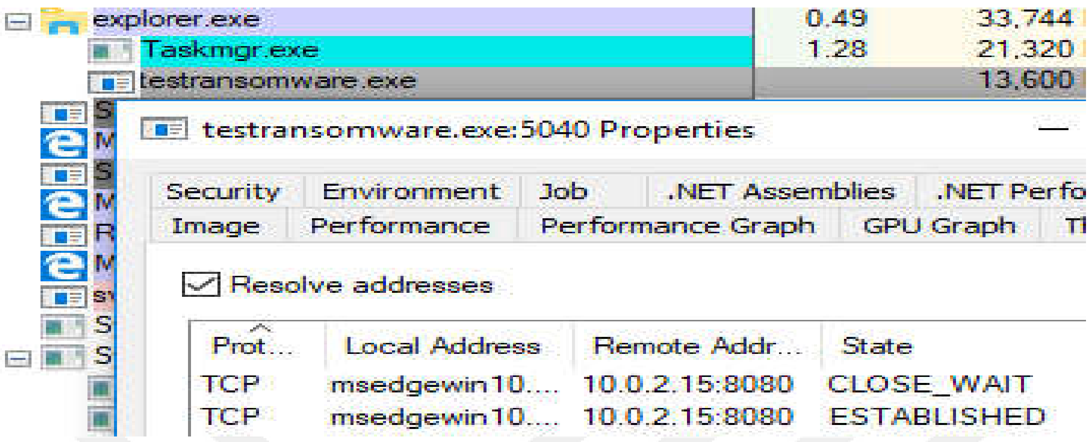
Dirty Decrypt, one of the first examples of the ransomware, has been completely detected by Checkpoint and has been prevented from encrypting files. Although the better-designed CryptoLocker ransomware used stronger encryption algorithms, the encrypted files could be partially restored using VSCs. Thus, following CryptoWall developers have deleted the VSCs on the computer and made the performance of the recovery operations impossible. However, because a pair of key generated by the ransomware running on the target computer is left undeleted, the files could be opened with the private key without paying the ransom [9, 12].

Following ransomware has begun using asymmetric cryptography method and C&C servers to store keys [9, 12]. The new generation ransomware has also made it almost impossible for the victim to access their files without paying the ransom by using both symmetric and asymmetric encryption techniques together in the encryption process.

The anatomy of the ransomware, as shown in Figure 3.1, is described in detail below.

- First, click on the link to download and run the ransomware,
- Then, the ransomware is executed [7],
- Before encryption starts, information about the target machine, such as processor, hostname, RAM, etc. information in the hash state, is gathered to identify the device. The ransomware creates a registry key in the following paths that it will use to store configuration information.
    - HKCU\Software\<uniquecomputer id>\<random id>
    - HKCU\Software\[random] [29].
- The ransomware checks whether there is any ransomware that has been injected into the system before. If there is no ransomware injected into the system;
    - The ransomware creates an instance of “explorer.exe” and injects itself into it [7]. This process is shown in Figure 3.2 and Figure 3.3.

- The most recently created instance of "explorer.exe" establishes a TCP connection with the C&C server, and the ransomware is hidden in order not to be detected. This process is shown in Figure 3.4.



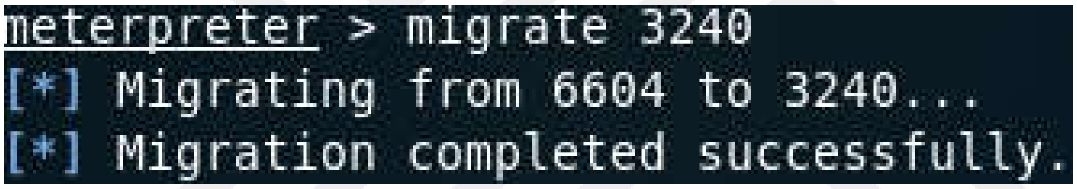
explorer.exe 0.49 33,744
Taskmgr.exe 1.28 21,320
testransomware.exe 13,600

testransomware.exe:5040 Properties

Resolve addresses

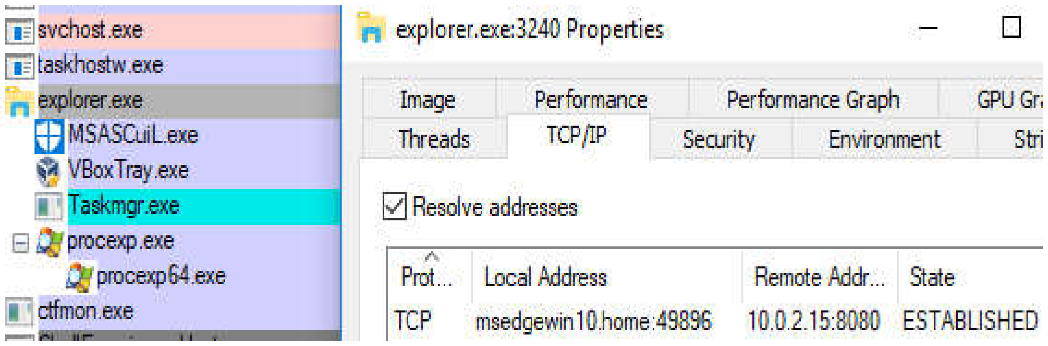
Prot...	Local Address	Remote Addr...	State
TCP	msedgewin10....	10.0.2.15:8080	CLOSE_WAIT
TCP	msedgewin10....	10.0.2.15:8080	ESTABLISHED

Figure 3.2: A connection image of "testransomware.exe" installed by C&C server.



```
meterpreter > migrate 3240
[*] Migrating from 6604 to 3240...
[*] Migration completed successfully.
```

Figure 3.3: A migration of "testransomware.exe" service to "explorer.exe" service.



svchost.exe
taskhostw.exe
explorer.exe
MSASCuiL.exe
VBoxTray.exe
Taskmgr.exe
procexp.exe
procexp64.exe
ctfmon.exe

explorer.exe:3240 Properties

Resolve addresses

Prot...	Local Address	Remote Addr...	State
TCP	msedgewin10.home:49896	10.0.2.15:8080	ESTABLISHED

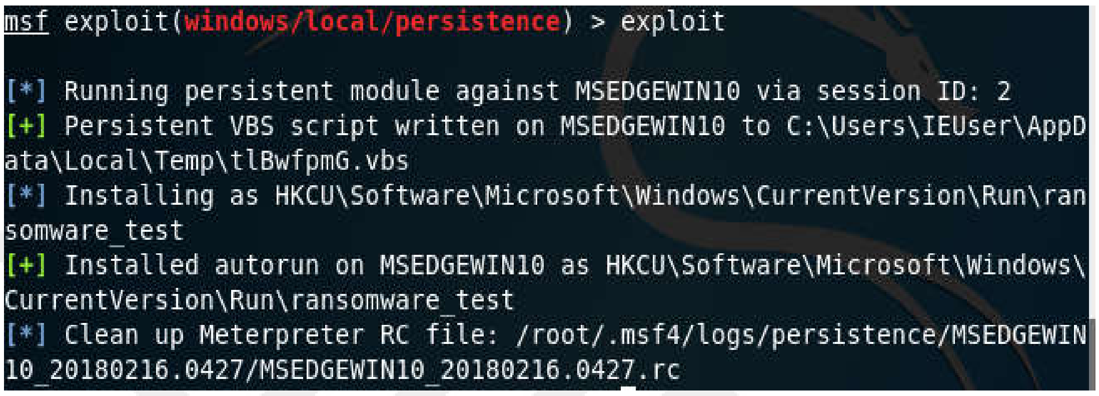
Figure 3.4: The image of "testransomware.exe" service not to be seen as a service.

- The ransomware generally runs in the following paths [36, 37].
  - % Localappdata%
  - % ProgramData%
  - % UserProfile%

    - % Temp%

- The ransomware also creates the "svchost.exe" instance [7]. This file is located in either “C:\Windows\System32” or “C:\Winnt\System32” path.

- As seen in Figure 3.5, the alteration is made at the registry run key in the following path, to become persistent and executable after restarting [7]. This path is “HKCU\Software\Microsoft\Windows\Currentversion\Run”.



```
msf exploit(windows/local/persistence) > exploit

[*] Running persistent module against MSEDGEWIN10 via session ID: 2
[+] Persistent VBS script written on MSEDGEWIN10 to C:\Users\IEUser\AppD
ata\Local\Temp\tlBwfpmG.vbs
[*] Installing as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\ran
somware_test
[+] Installed autorun on MSEDGEWIN10 as HKCU\Software\Microsoft\Windows\
CurrentVersion\Run\ransomware_test
[*] Clean up Meterpreter RC file: /root/.msf4/logs/persistence/MSEDGEWIN
10_20180216.0427/MSEDGEWIN10_20180216.0427.rc
```

Figure 3.5: A registry record creation transaction for persistent contact of "ransomware_test" payload with the C&C server.

- The ransomware uses “vssadmin.exe” service to delete or encrypt VSCs in the system [7, 11, 12, 18].

- In some ransomware attacks, “wmic.exe” service is used to delete all files in the VSCs [11, 38].

- The ransomware may prevent altering Windows boot options, and also certain backup applications. So, the ransomware wants to disable recovery and Windows error recovery on startup by using “bcdedit.exe” service [11, 12, 39].

- The ransomware may disable the Windows System Restore feature before starting the encryption process [11, 40].

- The ransomware can use “syskey.exe” service to encrypt user files (not supported by Windows 10 and Windows Servers), and “cipher.exe” service to manage encrypted data [35, 41, 42].

- The ransomware may stop special services such as “wscsvc”, “WinDefend”, “wuauserv”, “BITS”, “WerSvc”, etc. [43].

- The C&C Server is contacted and encryption keys with the victim's ID and password are transmitted [7, 11].

- All data traffic between the C&C Server and the client generally goes to and from an encrypted protocol such as HTTPs or TOR [6, 33].
- Files are encrypted and extortion messages about the payment of a ransom are displayed [7].
- Encrypted files are converted into specific file extensions. List of common ransomware encrypted file extensions are ".ccc", ".cerber", ".cerber2", ".cerber3", ".crypt", ".cryptolocker", ".cryptowall", ".ecc", ".ezz", ".locky", ".micro", ".zepto", and ".encrypted" [29, 37, 44].

In summary, the ransomware will migrate itself into the "explorer.exe" instance and creates a new instance of this process. And it copies its code in the paths like "%Appdata%", "%Programdata%" and creates registry value in "HKCU\Software\ Microsoft\Windows\Currentversion\Run". Thus, even if the system is rebooted, ransomware can run continuously and the connection between the C&C Server and the victim computer is made permanent. The attacker deletes VSCs from the system and ensures VSCs not be created later. The attacker can do this by using "vssadmin.exe" and "bdcedit.exe" services. The purpose of deleting the VSCs is to compel the victim to pay the ransom by preventing the system from getting its files back to an earlier point. After making these preliminary preparations, it encrypts the files with the example of "explorer.exe" or "svchost.exe" in which it injects itself and then gets in contact with the C&C server. After all the files are encrypted with a symetric encryption key, they have encrypted once again with the public key using one of the asymmetric encryption algorithms. Finally, an information note about how the ransom payment process is to be done is sent with a pop-up message in ".txt", ".png", ".html" etc. formats [7].

4. MODELING OF RANSOMWARE DETECTION TECHNIQUES AND METHODS

There are two different methods to detect ransomware in the system. These are "signature-based detection", and "anomaly-based detection" methods [45]. There are also three different techniques for both detection methods. These are "static", "dynamic" and "hybrid" techniques [45]. So, these ransomware detection methods and techniques are shown in Figure 4.1.

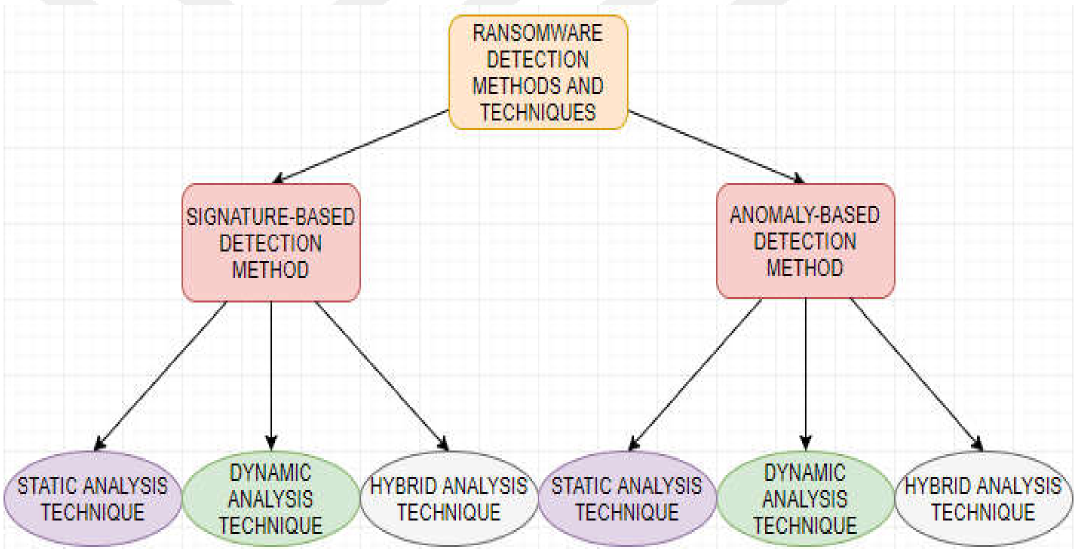

RANSOMWARE DETECTION METHODS AND TECHNIQUES
SIGNATURE-BASED DETECTION METHOD
ANOMALY-BASED DETECTION METHOD
STATIC ANALYSIS TECHNIQUE
DYNAMIC ANALYSIS TECHNIQUE
HYBRID ANALYSIS TECHNIQUE
STATIC ANALYSIS TECHNIQUE
DYNAMIC ANALYSIS TECHNIQUE
HYBRID ANALYSIS TECHNIQUE

Figure 4.1: Ransomware Detection Methods and Techniques [45].

4.1 Ransomware Detection Techniques

Three different techniques used in each of detection methods are explained below.

4.1.1 Static analysis technique

In this technique, the syntax or structural features of malware are used. This technique is used for detection before malware is run [45].

4.1.2 Dynamic analysis technique

In this technique, runtime information of malware is used. This technique is used for detection while malware is running or afterward [45].

4.1.3 Hybrid analysis technique

This technique consists of the use of the two techniques mentioned above. This technique is used for detection before, during, and after the malware is run [45]. It has superior advantages over other techniques.

4.2 Ransomware Detection Methods

Two different methods to detect ransomware in the system are explained below.

4.2.1 Signature-based detection method

In this method, ransomware is detected by signature information. Storage area such as repository is needed for storing signatures [45]. Current signature-based ransomware detection systems are largely based on syntactic signatures. These systems are equipped with a database of regular expressions that are considered malicious [46]. Static and hybrid signature-based detection methods are shown in Figure 4.2 and Figure 4.3.

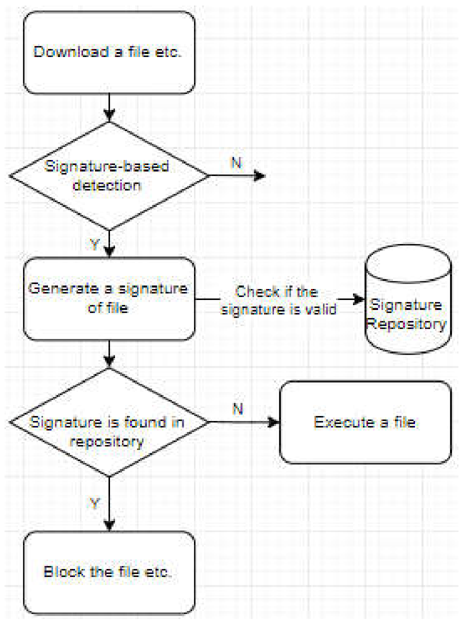

Download a file etc.
Signature-based detection
N
Y
Generate a signature of file
Check if the signature is valid
Signature Repository
Signature is found in repository
N
Execute a file
Y
Block the file etc.

Figure 4.2: Static Signature-Based Detection Method's UML Diagram.

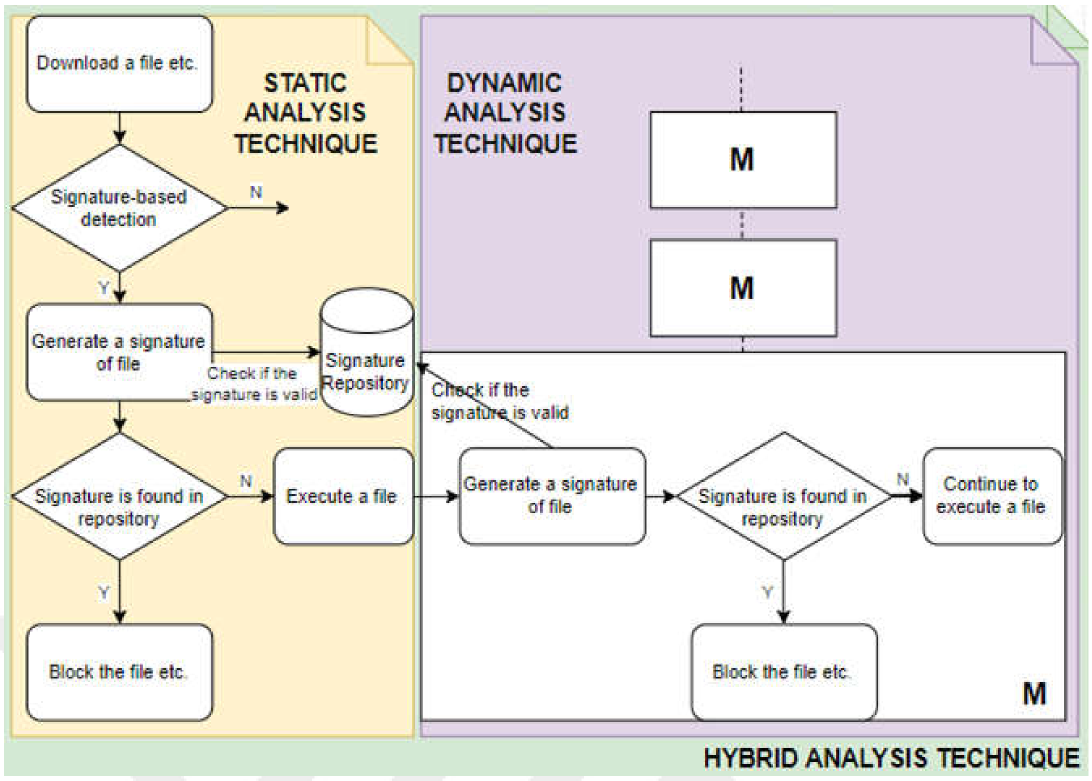

Download a file etc.
STATIC ANALYSIS TECHNIQUE
DYNAMIC ANALYSIS TECHNIQUE
M
M
Signature-based detection
N
Y
Generate a signature of file
Check if the signature is valid
Signature Repository
Check if the signature is valid
Signature is found in repository
N
Execute a file
Generate a signature of file
Signature is found in repository
N
Continue to execute a file
Y
Y
Block the file etc.
Block the file etc.
M
HYBRID ANALYSIS TECHNIQUE

Figure 4.3: Hybrid Signature-Based Detection Method's UML Diagram.

When the signature of the ransomware is created, it is recorded in the signature repository. The main reasons are to reduce the number of ransomware signatures in the repository, and speed up the subsequent detection rate [45].

Conventional signature-based detection systems only detect ransomware whose signatures are previously stored in the database [47]. These systems fail to detect new, previously unseen threats, that is, zero-day attacks [48]. This method is also less susceptible to obfuscation techniques [45, 46]. These systems can be easily deceived since signatures stored in databases are only based on regular expressions and string matching [47].

4.2.2 Anomaly-based detection method

In this method, it is essential to have the knowledge to decide whether the software is harmful. Rule sets are used to determine whether the software has established benign and valid behavior [45]. The major drawback of this method is defining its rule sets [47]. But the anomaly-based detection systems will work steadily as long as the rule sets are well defined before.

The main advantage of this method compared to signature-based detection systems is that it is possible to detect zero-day attacks that are not known before [47]. So, benign or malicious behavior can be detected, whether it is known attack or not [49].

This method consists of two phases. These are “training/learning” and “monitoring/detecting” phases. During the training phase, the benign behaviors of the system are revealed and the patterns of the system are observed [45, 49]

After the benign behaviors are defined in the training phase, this method switches to “monitoring” phase and compares the real-time system with benign behaviors learned during the previous phase [49]. The monitoring phase generates an alarm or warns when something except benign behaviors occurs in the system [45].

Disadvantages of this method are that it has high false-positive, too many alerts, and also difficulty and complexity in determining rule sets during the training phase [45].

Static and hybrid anomaly-based detection methods are shown in Figure 4.4 and Figure 4.5.

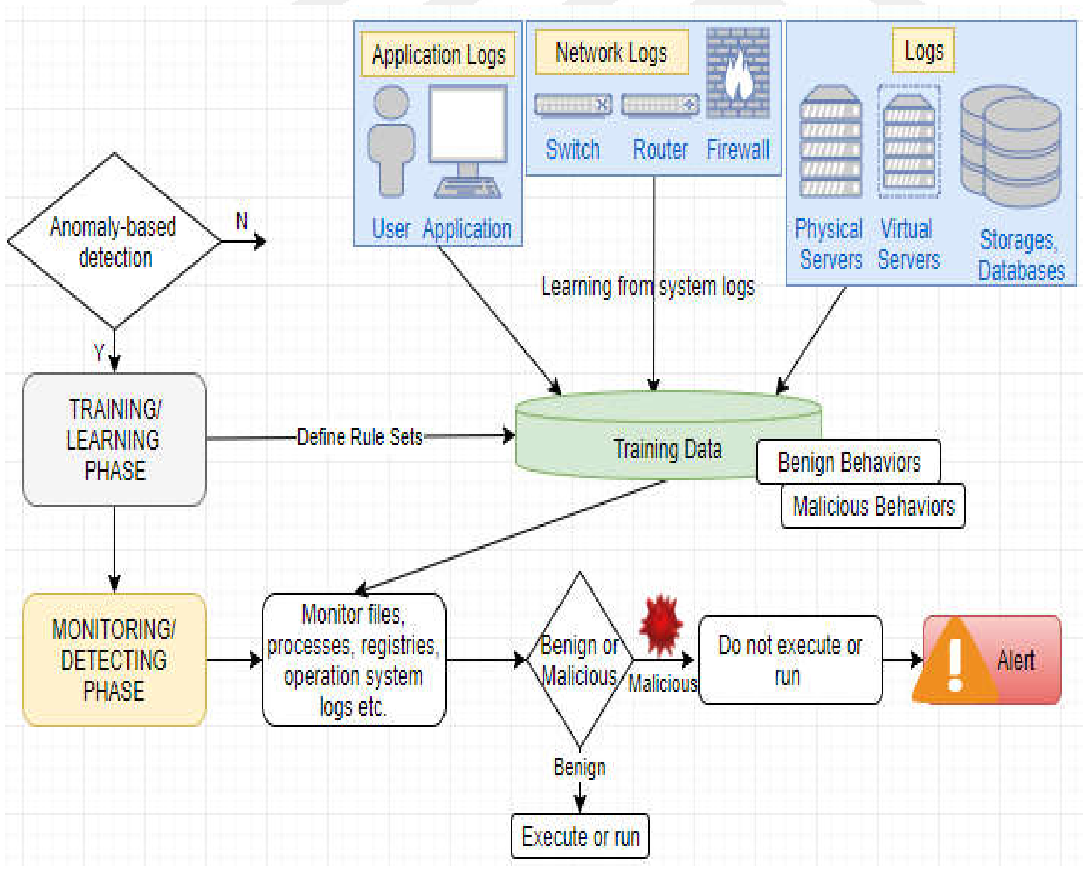


Figure 4.4: Static Anomaly-Based Detection Method’s UML Diagram.

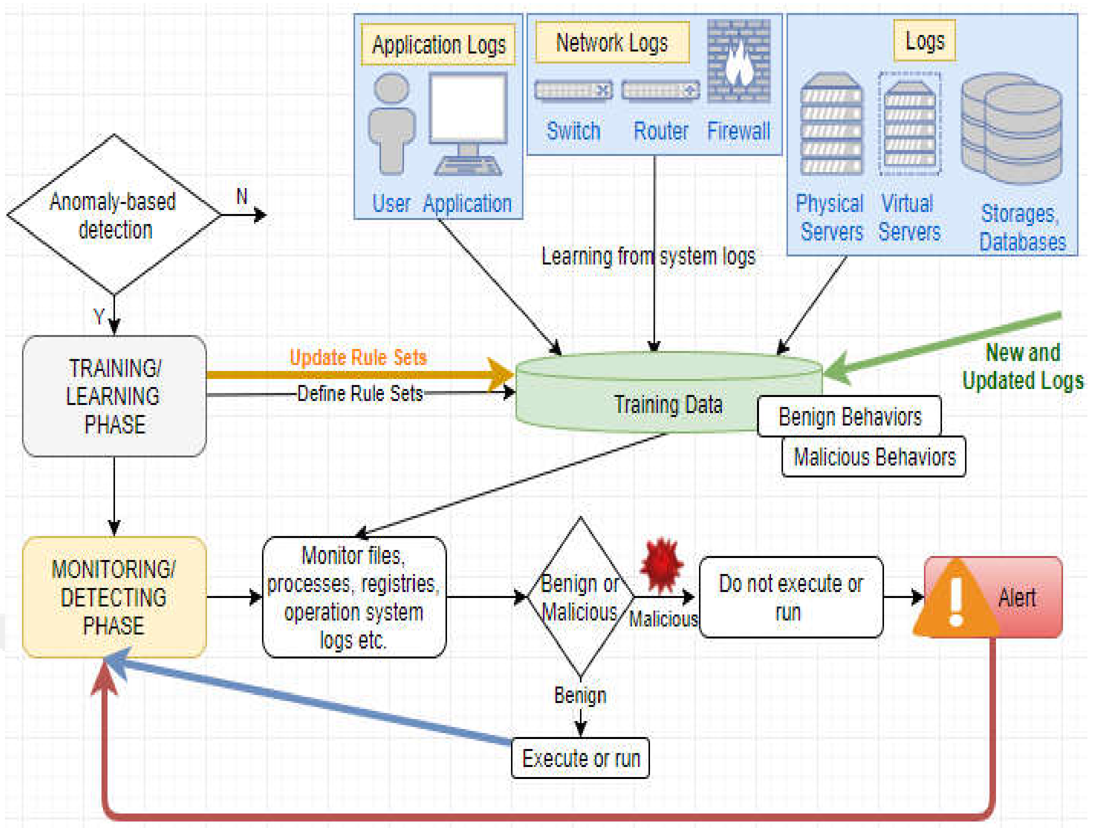


Figure 4.5: Hybrid Anomaly-Based Detection Method's UML Diagram.

Given the ransomware detection techniques and methods, the tool designed by using a hybrid analysis technique, signature-based, and anomaly-based detection methods can reach the most successful results to detect ransomware. So, the tool we designed is working in a hybrid structure because it uses both static and dynamic analysis techniques simultaneously. And also, all of the detection methods we have mentioned above have been used in the designing tool. Only the "learning" phase of the anomaly-based detection method which uses machine-learning techniques is not used. It is based on extensive literature research, reports from IT security companies, interview with cybersecurity experts and tests we have done in the virtual machines. That is, machine learning techniques have not been used, instead all materials have been evaluated as inputs to the design of our tool. So, it has been tried to obtain a result that produces much more accurate, minimum false positive, and minimum file loss.

5 MODELING OF RANSOMWARE DETECTION AND PREVENTION TOOL AND ITS IMPLEMENTATION

The ransomware uses a variety of attack vectors to access the target computer and develops itself constantly against the security measures taken. The ransomware constantly adds new features against the defense mechanisms [9]. Today, considering the position of ransomware they reached, security solutions such as firewalls and antiviruses, which are traditional detection methods using threat signature samples in databases, fail to detect and prevent ransomware [2, 12]. Therefore, using both static and behavioral methods to detect and prevent ransomware is a wiser solution.

Since the files in many different formats on the target machine are encrypted using advanced encryption algorithms, it is almost impossible for encrypted files to be decrypted without the decrypt key. As there is no guarantee that our files will be recovered after the cryptocurrency paid to receive the decrypt key, our main aim in this study is to introduce a method that detects and prevents the ransomware attacks at the beginning of the encryption. Therefore, the threat is to be prevented with minimum false-positive alarms and minimum file loss before the files encrypted.

The ransomware detection and prevention tool we designed checks registry records, the behavior of services and processes at the OS level and continuously monitors the encryption process that the ransomware tries to start on the file system. In addition, it checks whether the asymmetric encryption keys used by the new generation ransomware are in contact with the C&C server, analyzes the network traffic, and controls the downloaded files. In order to reach the aim detecting and preventing with minimum file loss and minimum false-positive alarms, the ransomware detection and prevention tool is designed to have a hybrid structure that combines the signature-based detection method with the anomaly-based detection method. In addition, it also uses the hybrid detection technique in order to monitor continuously.

Ransomware detection and prevention tool consist of two phases. These phases are defined below.

1st Phase: Hybrid Signature-Based Detection Phase

2nd Phase: Hybrid Anomaly-Based Detection Phase

Ransomware detection and prevention tool's UML diagram is shown in Figure 5.1.

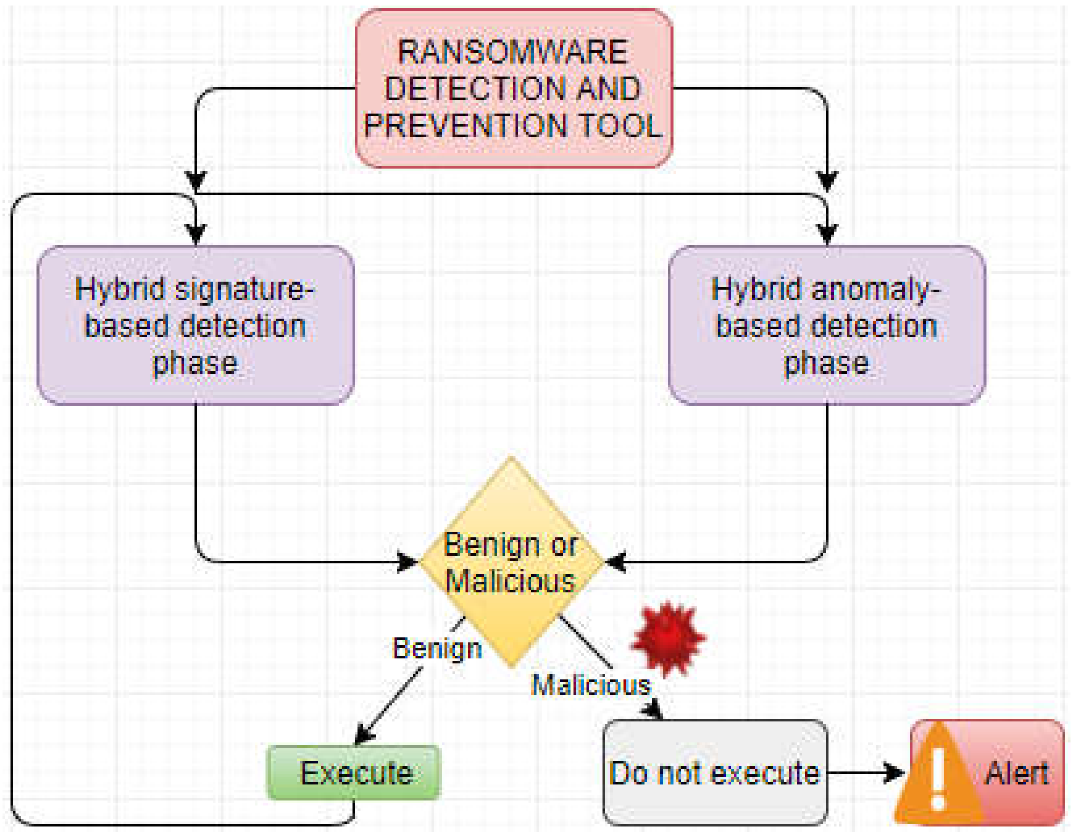
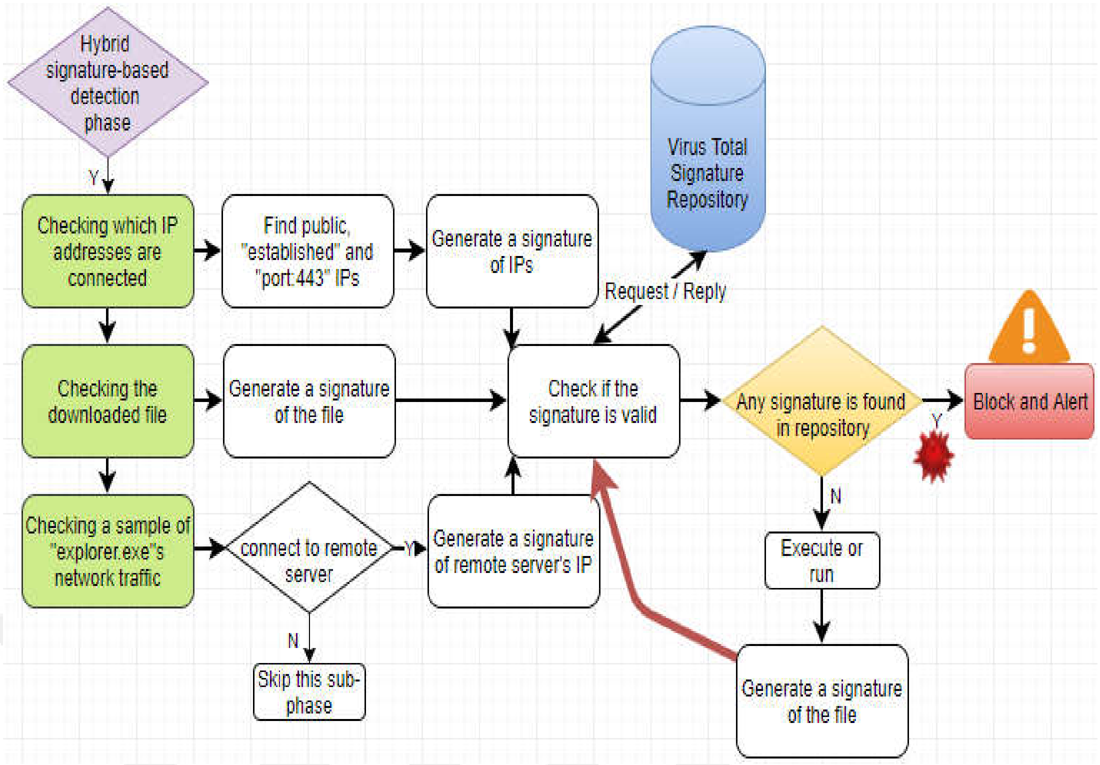

RANSOMWARE DETECTION AND PREVENTION TOOL
Hybrid signature-based detection phase
Hybrid anomaly-based detection phase
Benign or Malicious
Benign
Malicious
Execute
Do not execute
Alert

Figure 5.1: Ransomware Detection and Prevention Tool's UML Diagram.

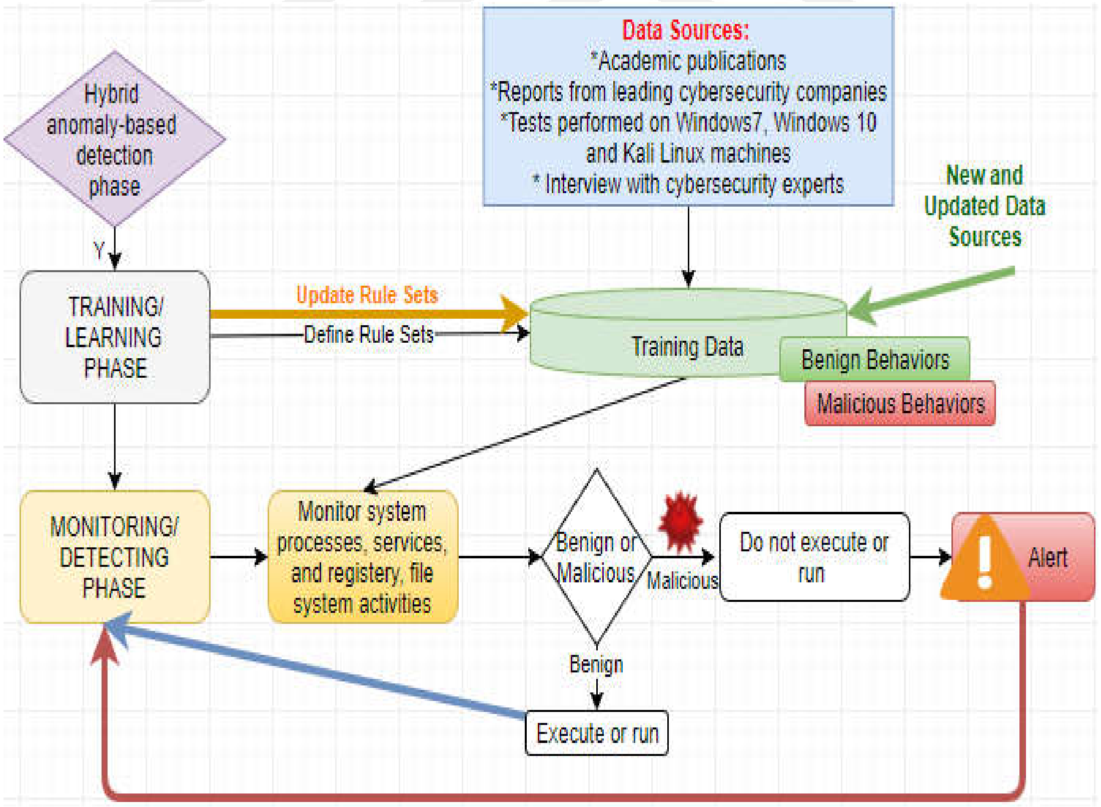
5.1 Modeling of Ransomware Detection and Prevention Tool

There are two main phases that Ransomware Detection and Prevention Tool used. These are a hybrid signature-based detection phase and a hybrid anomaly-based detection phase. Both phases work simultaneously in the detection process. The flow charts of these phases are shown as UML diagrams in Figure 5.2, Figure 5.3, Figure 5.4.



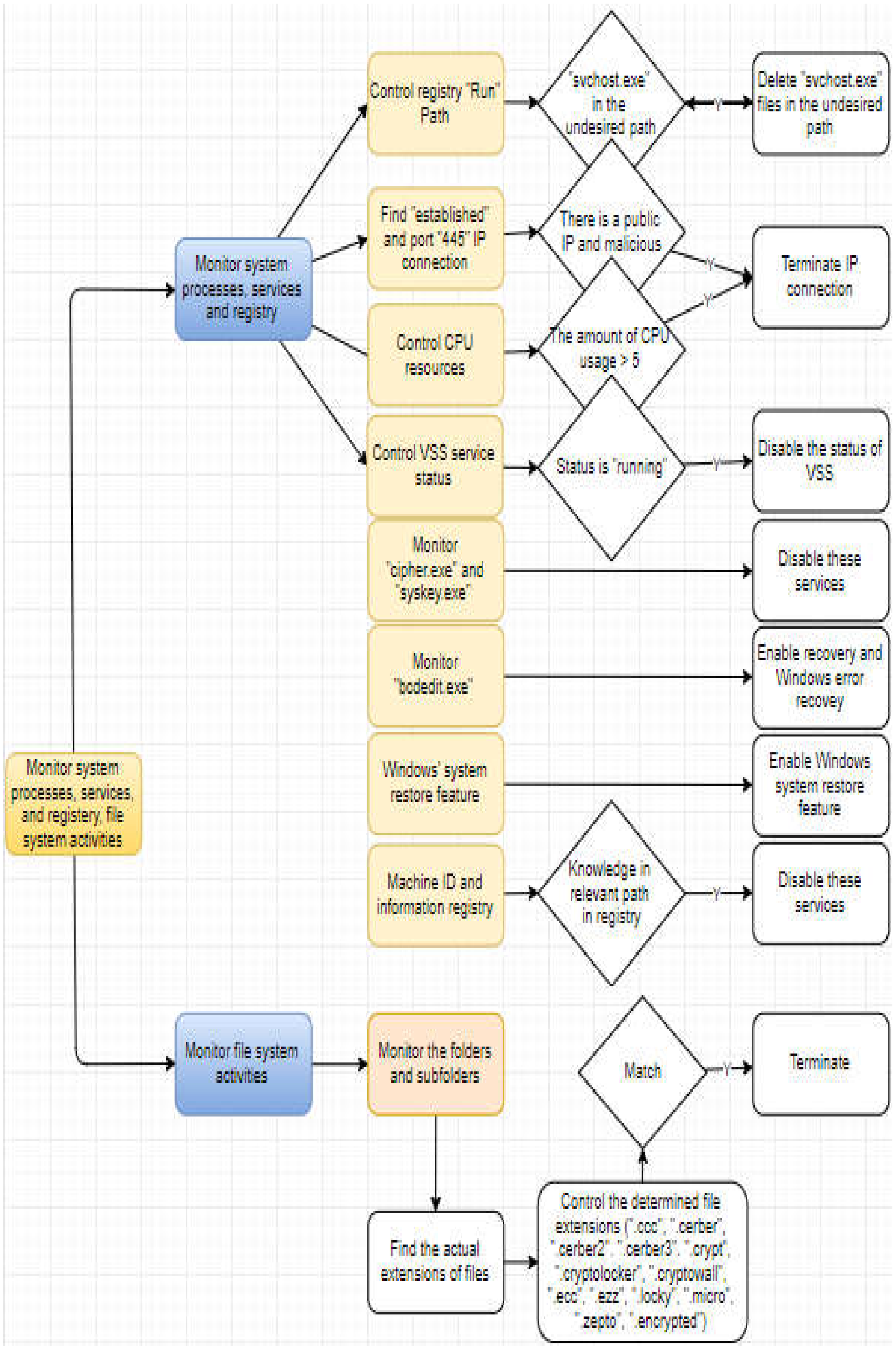
Hybrid signature-based detection phase
Y
Checking which IP addresses are connected
Find public, "established" and "port:443" IPs
Generate a signature of IPs
Virus Total Signature Repository
Request / Reply
Checking the downloaded file
Generate a signature of the file
Check if the signature is valid
Any signature is found in repository
Y
Block and Alert
N
Checking a sample of "explorer.exe"s network traffic
connect to remote server
Y
Generate a signature of remote server's IP
N
Skip this sub-phase
Execute or run
Generate a signature of the file

Figure 5.2: Ransomware Detection and Prevention Tool's Hybrid Signature-Based Detection Phase's UML Diagram.



Data Sources:
*Academic publications
*Reports from leading cybersecurity companies
*Tests performed on Windows7, Windows 10 and Kali Linux machines
* Interview with cybersecurity experts
Hybrid anomaly-based detection phase
Y
New and Updated Data Sources
TRAINING/ LEARNING PHASE
Update Rule Sets
Define Rule Sets
Training Data
Benign Behaviors
Malicious Behaviors
MONITORING/ DETECTING PHASE
Monitor system processes, services, and registery, file system activities
Benign or Malicious
Malicious
Do not execute or run
Alert
Benign
Execute or run

Figure 5.3: Ransomware Detection and Prevention Tool's Hybrid Anomaly-Based Detection Phase's UML Diagram *Monitoring Phase will be detailed shown in Figure 5.4.



Control registry "Run" Path
"svchost.exe" in the undesired path
Y
Delete "svchost.exe" files in the undesired path
Find "established" and port "445" IP connection
There is a public IP and malicious
Y
Terminate IP connection
Monitor system processes, services and registry
Control CPU resources
The amount of CPU usage > 5
Y
Control VSS service status
Status is "running"
Y
Disable the status of VSS
Monitor "cipher.exe" and "syskey.exe"
Disable these services
Monitor "bcdedit.exe"
Enable recovery and Windows error recovey
Monitor system processes, services, and registery, file system activities
Windows' system restore feature
Enable Windows system restore feature
Machine ID and information registry
Knowledge in relevant path in registry
Y
Disable these services
Monitor file system activities
Monitor the folders and subfolders
Match
Y
Terminate
Find the actual extensions of files
Control the determined file extensions (".ccc", ".cerber", ".cerber2", ".cerber3", ".crypt", ".cryptolocker", ".cryptowall", ".ecc", ".ezz", ".locky", ".micro", ".zepto", ".encrypted")

Figure 5.4: Ransomware Detection and Prevention Tool's Hybrid Anomaly-Based Detection Phase's Monitoring/Detecting Phase's UML Diagram.

5.2 Implementation of Ransomware Detection and Prevention Tool

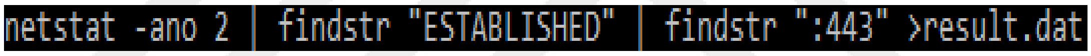
The phases of Ransomware Detection and Prevention Tool will be explained in detail below.

5.2.1 Hybrid signature-based detection phase

In this phase, the computer network connections and downloaded files are analyzed for malicious content.

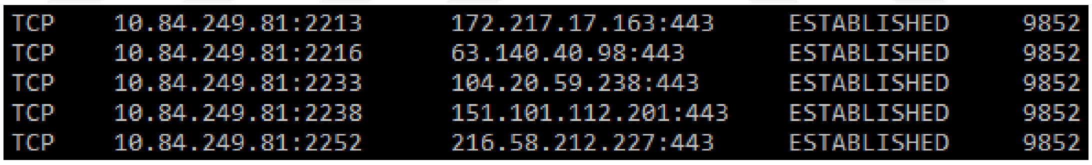
5.2.1.1 Checking which IP addresses are connected

IP addresses and PID numbers that are "Established" from port 443 will be continuously monitored as seen in Figure 5.5 and Figure 5.6. In Figure 5.5, "2" is the interval. It means to collect the statics every two seconds. "Https" is a protocol and 443 port is generally used by C&C server. And then, the results are saved as a "result.dat" file.



```
netstat -ano 2 | findstr "ESTABLISHED" | findstr ":443" >result.dat
```

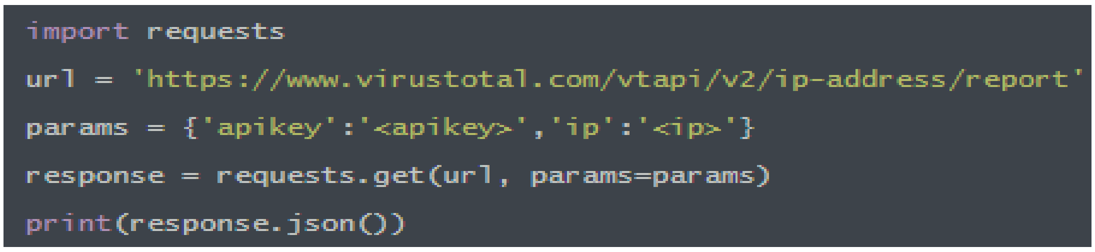
Figure 5.5: Searching network connections which are "Established" from "Port 443".



```
TCP    10.84.249.81:2213      172.217.17.163:443     ESTABLISHED     9852
TCP    10.84.249.81:2216      63.140.40.98:443       ESTABLISHED     9852
TCP    10.84.249.81:2233      104.20.59.238:443      ESTABLISHED     9852
TCP    10.84.249.81:2238      151.101.112.201:443    ESTABLISHED     9852
TCP    10.84.249.81:2252      216.58.212.227:443     ESTABLISHED     9852
```

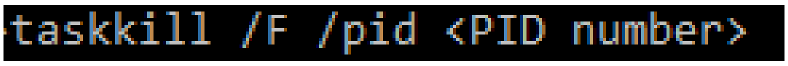
Figure 5.6: The result of the command in Figure 5.5.

The public IP addresses in Figure 5.6 are checked through the VirusTotal API as shown in Figure 5.7 and if the response comes as malicious, the connection is killed by terminating the PID number of that IP as shown in 5.8 [50].



```
import requests
url = 'https://www.virustotal.com/vtapi/v2/ip-address/report'
params = {'apikey':'<apikey>','ip':'<ip>'}
response = requests.get(url, params=params)
print(response.json())
```

Figure 5.7: An instance of a python code that checks IP address [50].

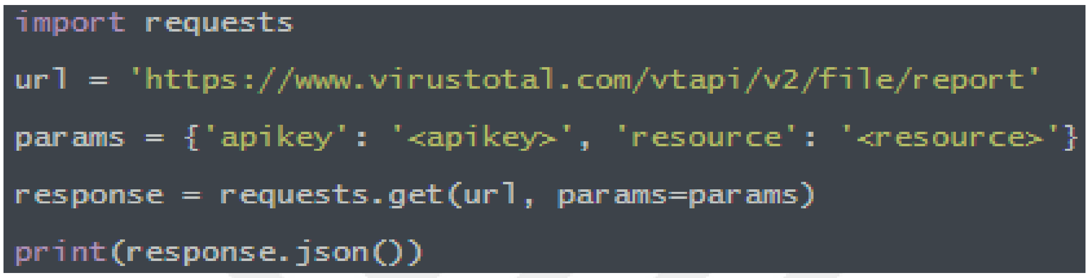


```
taskkill /F /pid <PID number>
```

Figure 5.8: Terminating a malicious IP's connection.

5.2.1.2 Checking the downloaded file

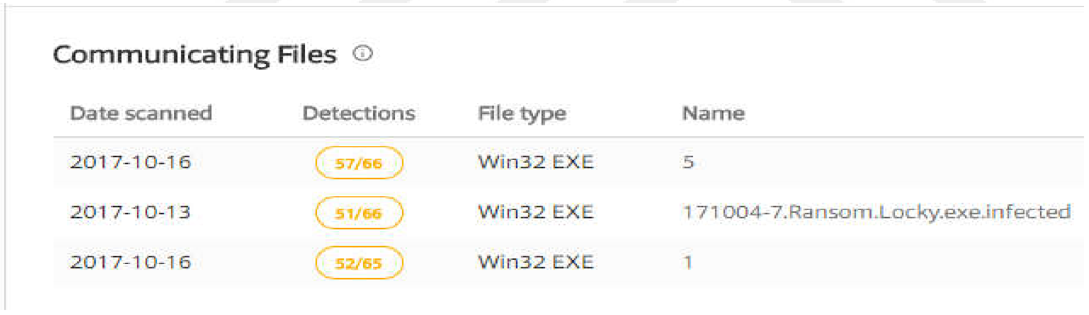
Whether the file downloaded to the computer is malicious or not is checked in almost 70 cybersecurity vendor database as in Figure 5.9. This python code sends a hash value of a downloaded file to VirusTotal via the get method in order to check whether the file is malicious. The file that comes out “clean” or “not malicious” from this control can be run. Otherwise, the file will be deleted before running and the user will be warned. Like sandbox, VirusTotal is used to detect whether a file is malicious.



```
import requests
url = 'https://www.virustotal.com/vtapi/v2/file/report'
params = {'apikey': '<apikey>', 'resource': '<resource>'}
response = requests.get(url, params=params)
print(response.json())
```

Figure 5.9: An example of a python code to check downloaded files [50].

Figure 5.10 shows a sample image of a file that is found to be malicious.



Communicating Files

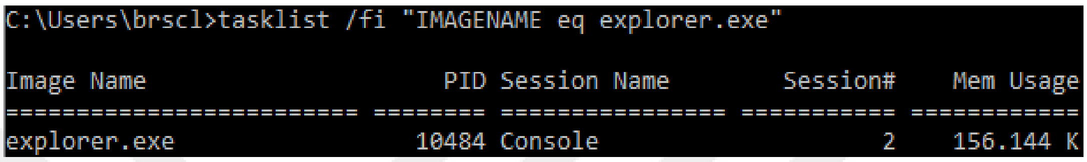
Date scanned	Detections	File type	Name
2017-10-16	57/66	Win32 EXE	5
2017-10-13	51/66	Win32 EXE	171004-7.Ransom.Locky.exe.infected
2017-10-16	52/65	Win32 EXE	1

Figure 5.10: A sample image of a file detected as a malicious.

5.2.1.3 Checking a sample of explorer.exe`s network traffic

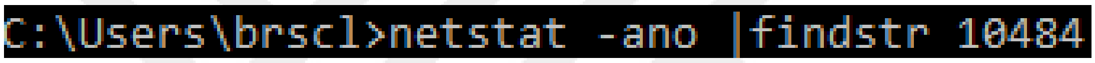
"Explorer.exe" is a component of the Windows OS that provides many users with interfaces on the monitor such as the taskbar and the desktop besides provides a GUI to access the file systems [51]. Before affecting the system, the ransomware launches a new suspended "explorer.exe" process and injects itself by allocating some memory [52]. The ransomware injected by the attackers communicate with the C&C servers through this process as shown in Figure 3.2. So, the movement of the "explorer.exe" process will also be examined.

Before encrypting the files of the victim, the C&C servers are contacted and the encryption keys are transmitted between the victim and the attacker. In this phase, the tool tries to determine whether the connection with the C&C servers is established. If any connection is detected, i.e., the attacker has control over the victim computer through the ransomware and can start the encryption process, the connection has to be terminated. To find an answer to the question "Does the last instance of "explorer.exe" create a TCP/IP connection with a remote C&C server?", the commands in Figure 5.11 and Figure 5.12 are executed.



```
C:\Users\brscl>tasklist /fi "IMAGENAME eq explorer.exe"

Image Name                     PID Session Name        Session#    Mem Usage
========================= ======== ================ =========== ============
explorer.exe                 10484 Console                    2    156.144 K
```

Figure 5.11: Learning “explorer.exe”’s PID number.



```
C:\Users\brscl>netstat -ano |findstr 10484
```

Figure 5.12: Learning C&C Server’s IP address connected via TCP protocol.

If the instance of "explorer.exe" connecting to a remote computer is detected, this connection will be terminated as in Figure 5.8.

5.2.2 Hybrid anomaly-based detection phase

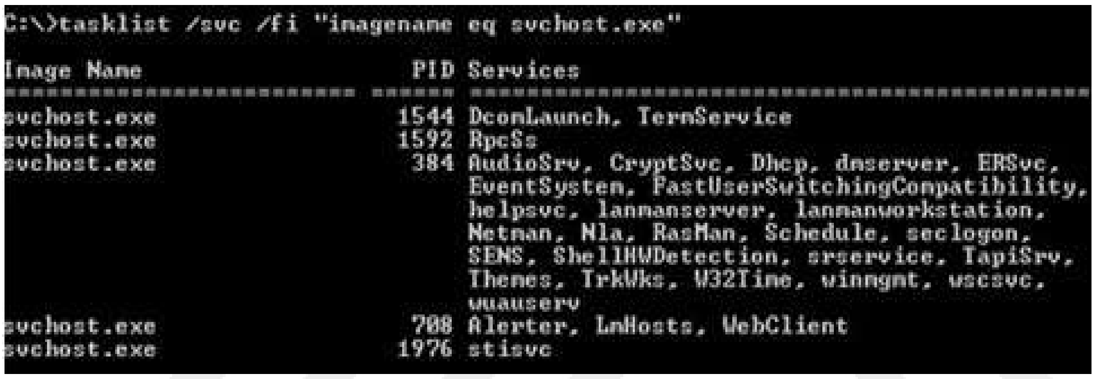
As we mentioned earlier, our tool aims to detect and prevent the ransomware at the OS level. Thus, analyzing the behavior of the Windows OS processes, services, registry records and the file system is needed. In this phase, the steps used for controlling and monitoring to achieve defined aims are explained in detail.

5.2.2.1 System processes, services, and registry

OSs cannot provide full functionality to their users without Windows processes. Some processes require special rights or resources that a normal user can not use. In some cases, an attacker may also escalate his/her privileges by exploiting the "normal" behavior of these processes.

5.2.2.1.1 Svchost.exe

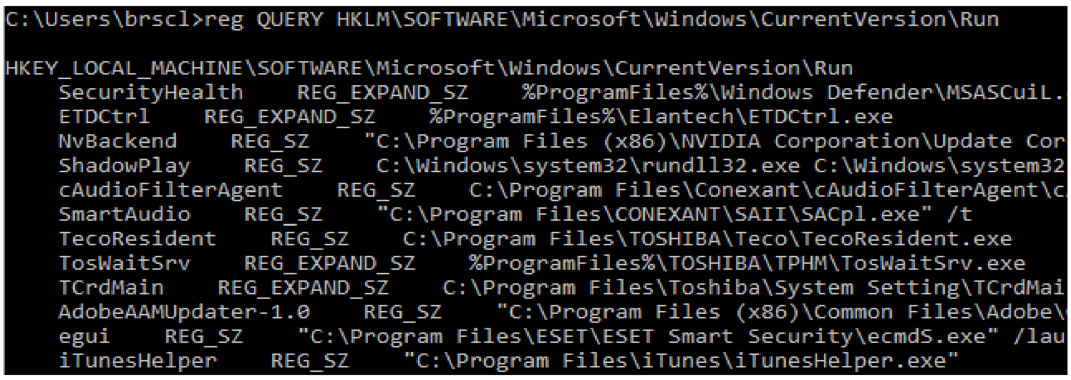
This process is required to load the ".dll" files that are needed for the Windows and Windows programs running on the computer [53]. “Svchost.exe” file is located at the "C:\\Windows\System32" or "C:\\Winnt\System32" paths. Because "svchost.exe" is the common system process like "explorer.exe", malware usually is hidden under this name [54]. To reduce resource consumption, many services share the "svchost.exe" process and migrate itself to this process [51]. As seen in Figure 5.13, a single “svchost.exe” process can load and manage multiple services. These services are in the following Windows registry key: “HKLM\Software\Microsoft\Windows NT\CurrentVersion\SvcHost”.



```
C:\>tasklist /svc /fi "imagename eq svchost.exe"

Image Name                     PID Services
========================= ======== ============================================
svchost.exe                   1544 DcomLaunch, TermService
svchost.exe                   1592 RpcSs
svchost.exe                    384 AudioSrv, CryptSvc, Dhcp, dmserver, ERSvc,
                                   EventSystem, FastUserSwitchingCompatibility,
                                   helpsvc, lanmanserver, lanmanworkstation,
                                   Netman, Nla, RasMan, Schedule, seclogon,
                                   SENS, ShellHWDetection, srservice, TapiSrv,
                                   Themes, TrkWks, W32Time, winmgmt, wscsvc,
                                   wuauserv
svchost.exe                    708 Alerter, LmHosts, WebClient
svchost.exe                   1976 stisvc
```

Figure 5.13: The image of a “svchost.exe” process in the registry.

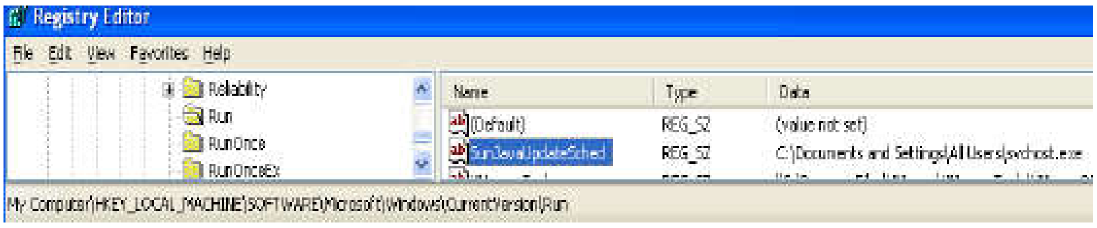
"Svchost.exe" infections are usually installed by copying the executables into the system files and making necessary modifications in the registry so that infections can be run whenever the system is rebooted [11]. To accomplish this operation, "HKLM\Software\Microsoft\Windows\CurrentVersion\Run" path is used as shown in Figure 5.14, Figure 5.15 and Figure 5.16.



```
C:\Users\brscl>reg QUERY HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
    SecurityHealth    REG_EXPAND_SZ    %ProgramFiles%\Windows Defender\MSASCuiL.
    ETDCtrl    REG_EXPAND_SZ    %ProgramFiles%\Elantech\ETDCtrl.exe
    NvBackend    REG_SZ    "C:\Program Files (x86)\NVIDIA Corporation\Update Cor
    ShadowPlay    REG_SZ    C:\Windows\system32\rundll32.exe C:\Windows\system32
    cAudioFilterAgent    REG_SZ    C:\Program Files\Conexant\cAudioFilterAgent\c
    SmartAudio    REG_SZ    "C:\Program Files\CONEXANT\SAII\SACpl.exe" /t
    TecoResident    REG_SZ    C:\Program Files\TOSHIBA\Teco\TecoResident.exe
    TosWaitSrv    REG_EXPAND_SZ    %ProgramFiles%\TOSHIBA\TPHM\TosWaitSrv.exe
    TCrdMain    REG_EXPAND_SZ    C:\Program Files\Toshiba\System Setting\TCrdMai
    AdobeAAMUpdater-1.0    REG_SZ    "C:\Program Files (x86)\Common Files\Adobe\
    egui    REG_SZ    "C:\Program Files\ESET\ESET Smart Security\ecmdS.exe" /lau
    iTunesHelper    REG_SZ    "C:\Program Files\iTunes\iTunesHelper.exe"
```

Figure 5.14: An instance of programs running in the “HKLM\Software\Microsoft\Windows\CurrentVersion\Run” path.

In Figure 5.15, malicious “svchost.exe” example at the “C:\Documents and Settings\All Users” path and alteration are made to the “Run” path, to become persistent and executive after the restarting.



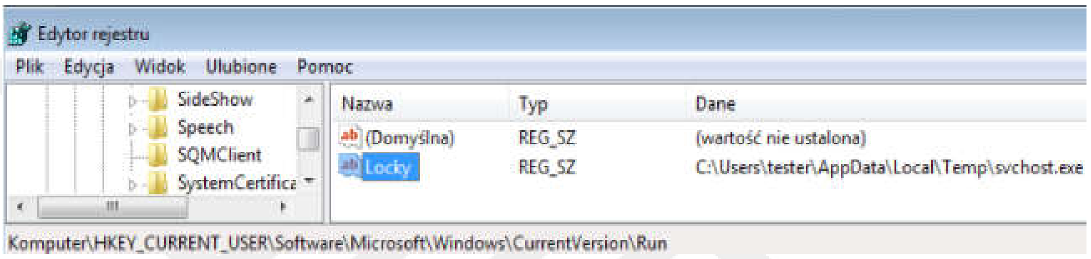
Registry Editor

File Edit View Favorites Help

Name	Type	Data
(Default)	REG_SZ	(value not set)
SunJavaUpdateSched	REG_SZ	C:\Documents and Settings\All Users\svchost.exe

My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

Figure 5.15: An alteration sample in the “HKLM\Software\Microsoft\Windows\CurrentVersion\Run” path.



Edytor rejestru

Plik Edycja Widok Ulubione Pomoc

Nazwa	Typ	Dane
(Domyślna)	REG_SZ	(wartość nie ustalona)
Locky	REG_SZ	C:\Users\tester\AppData\Local\Temp\svchost.exe

Komputer\HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run

Figure 5.16: A Locky ransomware sample in the “HKLM\Software\Microsoft\Windows\CurrentVersion\Run” path [28].

The tool should continuously monitor the “HKLM\Software\Microsoft\ Windows\CurrentVersion\Run” path and if a new instance of "svchost.exe" occurs outside the “%ProgramFiles%, %ProgramFiles(x86)% and %System Root%\ System32” paths, it is highly likely to be malware or ransomware [54]. Hence, Figure 5.17 shows how to delete the instance of “svchost.exe" in the undesired paths.

```
C:\WINDOWS\system32>del C:\Documents and Settings\All Users\svchost.exe /F /Q
```

```
C:\WINDOWS\system32>del C:\Users\tester\AppData\Local\Temp\svchost.exe /F /Q
```

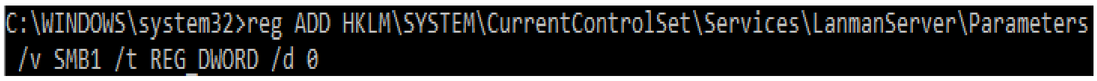
Figure 5.17: Deleting the instances of "svchost.exe" in the undesired paths.

In addition, with the command in Figure 5.18, the list of running programs each time when the system starts and the user logs on can be continuously checked. Because many ransomware is trying to make itself work constantly as we have mentioned before [38].

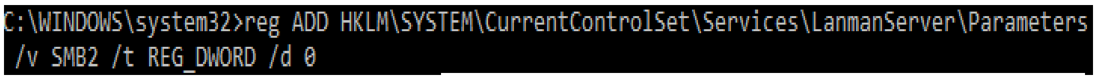
```
wmic startup list full /every:5
```

Figure 5.18: The command of showing the list of running programs.

WannaCry and Petya ransomware may infect by exploiting the gap of Windows SMB protocol. So, we can protect the systems by making SMBv1 and SMBv2 disable, as in Figure 5.19 [55].



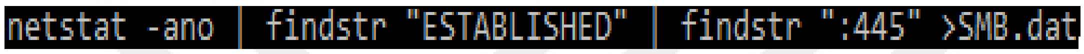
```
C:\WINDOWS\system32>reg ADD HKLM\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters
 /v SMB1 /t REG_DWORD /d 0
```



```
C:\WINDOWS\system32>reg ADD HKLM\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters
 /v SMB2 /t REG_DWORD /d 0
```

Figure 5.19: The commands of disabling SMBv1 and SMBv2 on the machine [56].

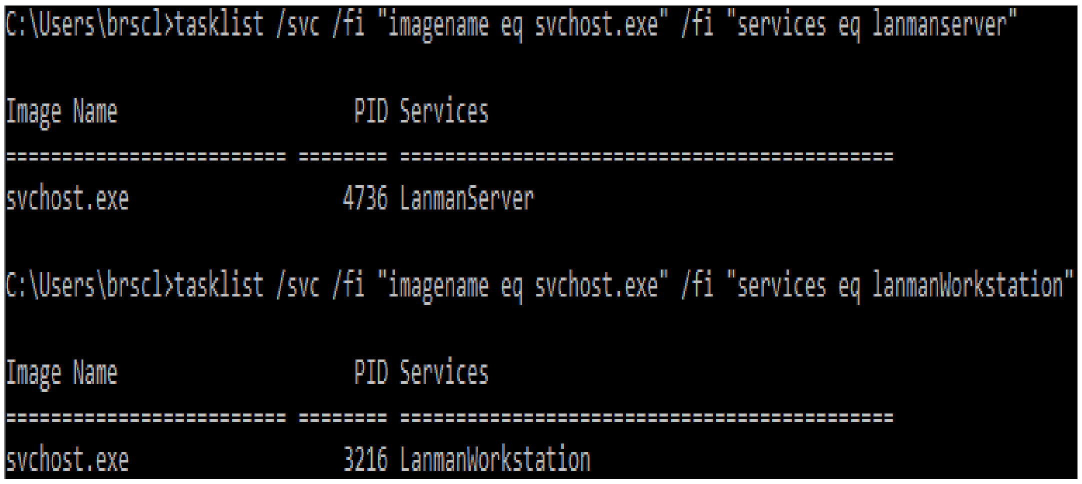
Although SMBv1 and SMBv2 cannot be disabled, it will constantly be checked whether there is a connection from port 445 as shown in Figure 5.20, and if there is such a connection, it will be terminated as in Figure 5.8[31].



```
netstat -ano | findstr "ESTABLISHED" | findstr ":445" >SMB.dat
```

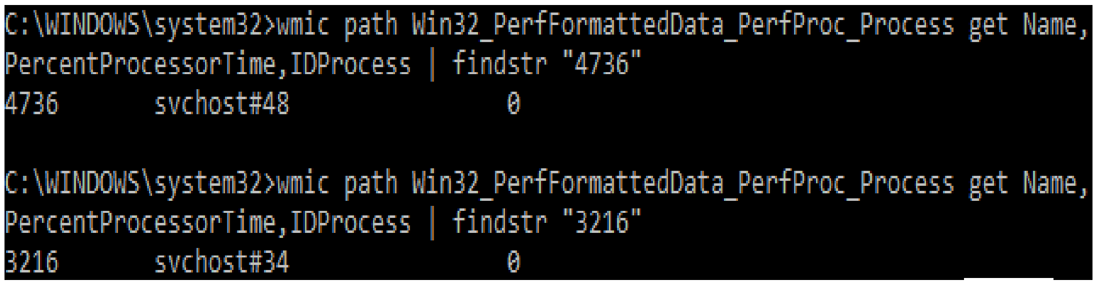
Figure 5.20: Connections established via port 445.

The LanmanWorkstation service creates and maintains client network connections to remote servers using the SMB protocol. The LanmanServer service supports file, printer and named channel sharing over the network. As a consequence of disabling these services, the SMB protocol becomes inoperable [57]. Therefore, first of all, the PIDs of these services are found as in Figure 5.21, and they are constantly monitored as to how much they use the system CPU resources in total, as in Figure 5.22. When the amount of CPU used increases as shown in Figure 5.23, these two services (LanmanWorkstation and LanmanServer) will be terminated as in Figure 5.8.



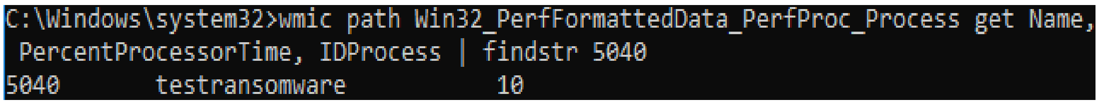
```
C:\Users\brscl>tasklist /svc /fi "imagename eq svchost.exe" /fi "services eq lanmanserver"

Image Name                     PID Services
========================= ======== ============================================
svchost.exe                   4736 LanmanServer

C:\Users\brscl>tasklist /svc /fi "imagename eq svchost.exe" /fi "services eq lanmanWorkstation"

Image Name                     PID Services
========================= ======== ============================================
svchost.exe                   3216 LanmanWorkstation
```

Figure 5.21: Getting Lanmanserver and LanmanWorkstation services' PID numbers in the command line.



```
C:\WINDOWS\system32>wmic path Win32_PerfFormattedData_PerfProc_Process get Name,
PercentProcessorTime,IDProcess | findstr "4736"
4736       svchost#48                0

C:\WINDOWS\system32>wmic path Win32_PerfFormattedData_PerfProc_Process get Name,
PercentProcessorTime,IDProcess | findstr "3216"
3216       svchost#34                0
```

Figure 5.22: Getting LanmanServer and LanmanWorkstation services' %CPU usage.



```
C:\Windows\system32>wmic path Win32_PerfFormattedData_PerfProc_Process get Name,
 PercentProcessorTime, IDProcess | findstr 5040
5040       testransomware           10
```

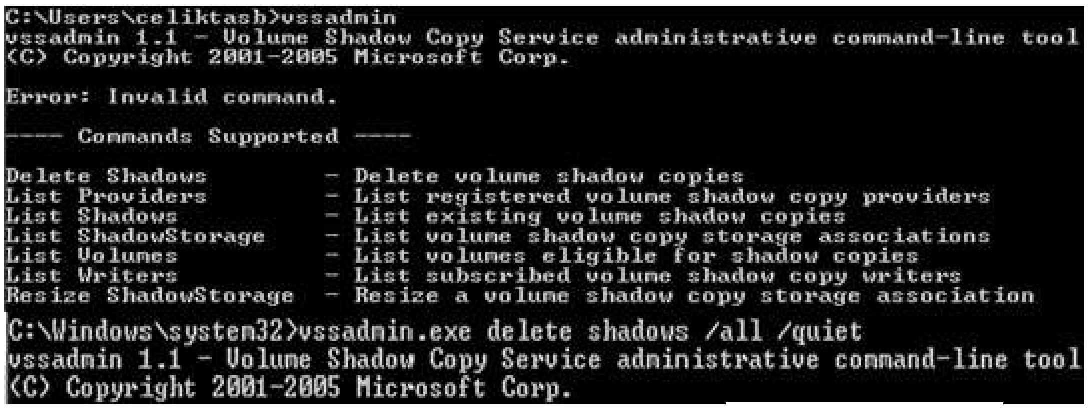
Figure 5.23: The image of the percentage of CPU used by "testransomware" service.

5.2.2.1.2 Vssadmin.exe and wmic.exe

VSC is a feature that provides snapshots and backups of your files. As long as VSS runs, these snapshots will try to be created at a certain time and allow you to restore documents to previous versions even though they are deleted or encrypted. Thus, since ransomware developers are aware of VSCs, deleting these VSCs is the first thing they do when they get involved in the system [12].

Whereas some malware and APTs cannot delete the backed up files in the system, the new generation ransomware can permanently delete backed up files in VSCs using the VSS. Thus, the victim cannot recover their files and has to pay the ransom to the attacker.

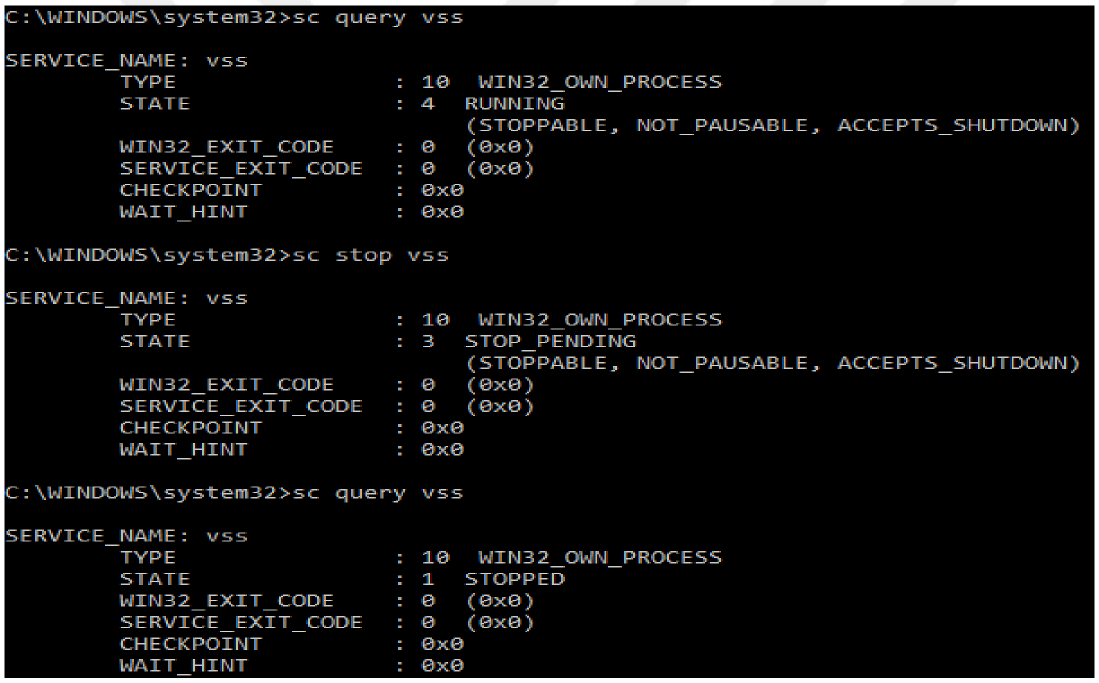
The Vssadmin tool is used to encrypt or delete VSCs. For example, Cerber, Locky, and CryptoWall completely delete the backed up folders and files on the system. And some terminate VSS so that deleted or encrypted folders and files cannot be recovered [58]. The command in Figure 5.24 execute the "vssadmin.exe" utility and quietly delete all of the Shadow Volume Copies on the computer [11, 12, 59].



```
C:\Users\celiktasb>vssadmin
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2005 Microsoft Corp.

Error: Invalid command.

---- Commands Supported ----

Delete Shadows        - Delete volume shadow copies
List Providers        - List registered volume shadow copy providers
List Shadows          - List existing volume shadow copies
List ShadowStorage    - List volume shadow copy storage associations
List Volumes          - List volumes eligible for shadow copies
List Writers          - List subscribed volume shadow copy writers
Resize ShadowStorage  - Resize a volume shadow copy storage association
C:\Windows\system32>vssadmin.exe delete shadows /all /quiet
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2005 Microsoft Corp.
```

Figure 5.24: The command of deleting the VSCs on the computer [59].

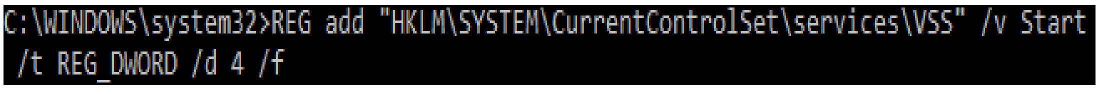
In order prevent the attackers run these commands, the VSS must be in the "Stopped" state. Therefore, the state of the VSS will be continuously monitored in our tool as shown in Figure 5.25.



```
C:\WINDOWS\system32>sc query vss

SERVICE_NAME: vss
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 4  RUNNING
                                (STOPPABLE, NOT_PAUSABLE, ACCEPTS_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0

C:\WINDOWS\system32>sc stop vss

SERVICE_NAME: vss
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 3  STOP_PENDING
                                (STOPPABLE, NOT_PAUSABLE, ACCEPTS_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0

C:\WINDOWS\system32>sc query vss

SERVICE_NAME: vss
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 1  STOPPED
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0
```

Figure 5.25: Commands for checking the status of the VSS and stopping it afterward.

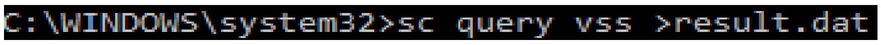
In addition, we need to disable the VSS service by the command in Figure 5.26. So nobody can start VSS if startup type is disabled.



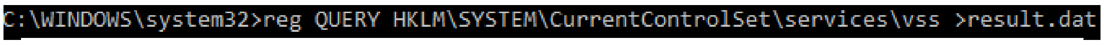
```
C:\WINDOWS\system32>REG add "HKLM\SYSTEM\CurrentControlSet\services\VSS" /v Start
 /t REG_DWORD /d 4 /f
```

Figure 5.26: The command of disabling the VSS.

The ransomware will need to reverse the operations to delete VSCs even the VSS is “Stopped” and “Disabled”. Hence, the output of Figure 5.27 will be continuously monitored, where any change will indicate an abnormality and the tool will terminate the connection as in Figure 5.8.



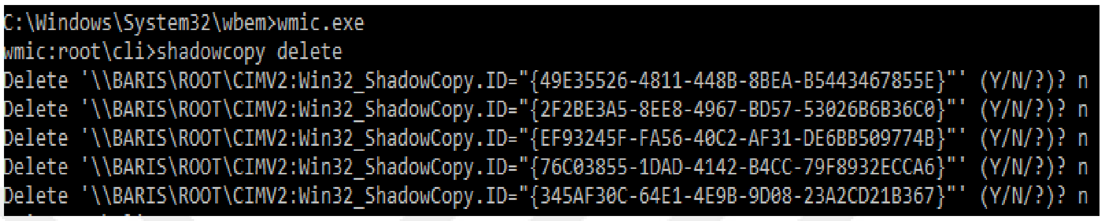
```
C:\WINDOWS\system32>sc query vss >result.dat
```



```
C:\WINDOWS\system32>reg QUERY HKLM\SYSTEM\CurrentControlSet\services\vss >result.dat
```

Figure 5.27: The commands that give the outputs of the status of registry and state of the VSS.

Some ransomware also uses "wmic.exe" as in Figure 5.28 to delete VSCs [11, 18].

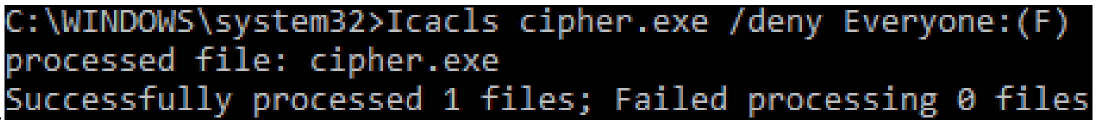


```
C:\Windows\System32\wbem>wmic.exe
wmic:root\cli>shadowcopy delete
Delete '\\BARIS\ROOT\CIMV2:Win32_ShadowCopy.ID="{49E35526-4811-448B-8BEA-B5443467855E}"' (Y/N/?)? n
Delete '\\BARIS\ROOT\CIMV2:Win32_ShadowCopy.ID="{2F2BE3A5-8EE8-4967-BD57-53026B6B36C0}"' (Y/N/?)? n
Delete '\\BARIS\ROOT\CIMV2:Win32_ShadowCopy.ID="{EF93245F-FA56-40C2-AF31-DE6BB509774B}"' (Y/N/?)? n
Delete '\\BARIS\ROOT\CIMV2:Win32_ShadowCopy.ID="{76C03855-1DAD-4142-B4CC-79F8932ECCA6}"' (Y/N/?)? n
Delete '\\BARIS\ROOT\CIMV2:Win32_ShadowCopy.ID="{345AF30C-64E1-4E9B-9D08-23A2CD21B367}"' (Y/N/?)? n
```

Figure 5.28: The command of deleting VSCs.

5.2.2.1.3 Cipher.exe and syskey.exe

"Cipher.exe" is the command-line tool for managing encrypted data using the EFS [41]. The ransomware generally uses "cipher.exe" after the encryption is finished. So, the tool disables "cipher.exe" as shown in Figure 5.29.

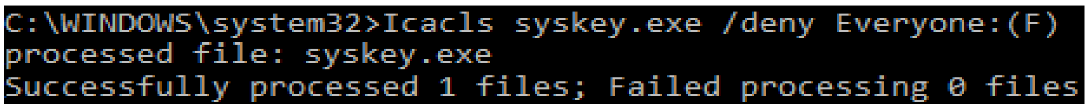


```
C:\WINDOWS\system32>Icacls cipher.exe /deny Everyone:(F)
processed file: cipher.exe
Successfully processed 1 files; Failed processing 0 files
```

.

Figure 5.29: The command of taking all authority of "cipher.exe" file back and making unusable.

"Syskey.exe" is a Windows built-in root encryption key used to encrypt sensitive Windows OS status data such as user account passwords. According to Microsoft, the use of the SysKey encryption key and "syskey.exe" is no longer considered safe and is frequently used by computer hackers [35]. Thus, we should disable "syskey.exe" like we did for "cipher.exe" as seen in Figure 5.30.

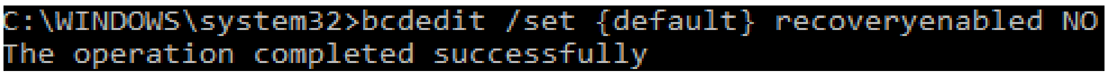


```
C:\WINDOWS\system32>Icacls syskey.exe /deny Everyone:(F)
processed file: syskey.exe
Successfully processed 1 files; Failed processing 0 files
```

Figure 5.30: The command of taking all authority of "syskey.exe" file is back and making unusable.

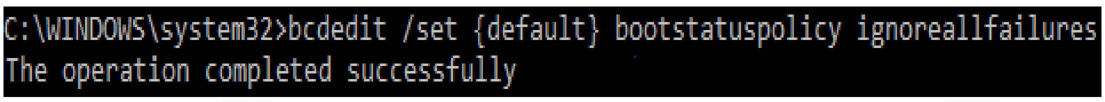
5.2.2.1.4 Bcdedit.exe

Ransomware blocks altering Windows boot options, it may also block certain backup applications. As seen in Figure 5.31 and Figure 5.32, ransomware wants to disable Recovery and Windows Error Recovery on startup [11, 12, 39]. By disabling Windows Error Recovery, the computer will attempt to boot normally after an error occurs [60].



```
C:\WINDOWS\system32>bcdedit /set {default} recoveryenabled NO
The operation completed successfully
```

Figure 5.31: The command of disabling recovery.

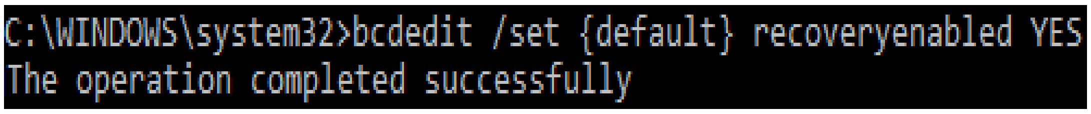


```
C:\WINDOWS\system32>bcdedit /set {default} bootstatuspolicy ignoreallfailures
The operation completed successfully
```

Figure 5.32: The command of disabling Windows Error Recovery on startup.

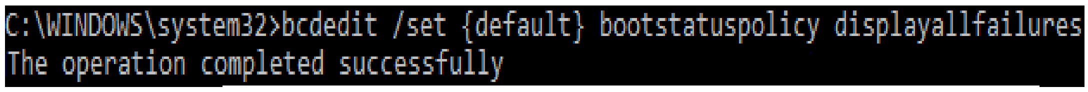
Therefore, the commands in Figure 5.33 and Figure 5.34 will run continuously and the attempt of the ransomware to prevent the system from correcting the error will be eliminated.

The command in Figure 5.34 tries to fix booting related issues automatically. If PC crashes twice consecutively or fails to boot, it will launch the automatic repair procedure and will try to fix the issue, which caused the crash [60].



```
C:\WINDOWS\system32>bcdedit /set {default} recoveryenabled YES
The operation completed successfully
```

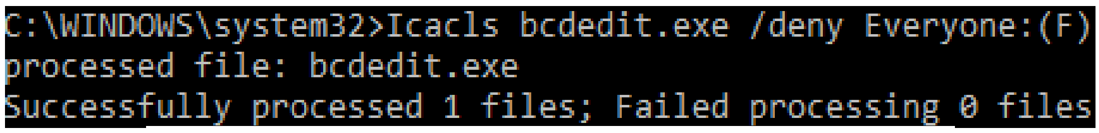
Figure 5.33: The command of enabling recovery.



```
C:\WINDOWS\system32>bcdedit /set {default} bootstatuspolicy displayallfailures
The operation completed successfully
```

Figure 5.34: The command of enabling Windows Error Recovery on startup.

In addition, the "bcdedit.exe" file is prevented from being executed by unauthorized people with the command in Figure 5.35, thus the danger to our system is prevented.

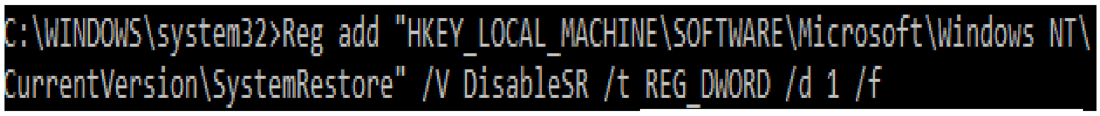


```
C:\WINDOWS\system32>Icacls bcdedit.exe /deny Everyone:(F)
processed file: bcdedit.exe
Successfully processed 1 files; Failed processing 0 files
```

Figure 5.35: The command of making "bcdedit.exe" unusable.

5.2.2.1.5 Windows system restore feature

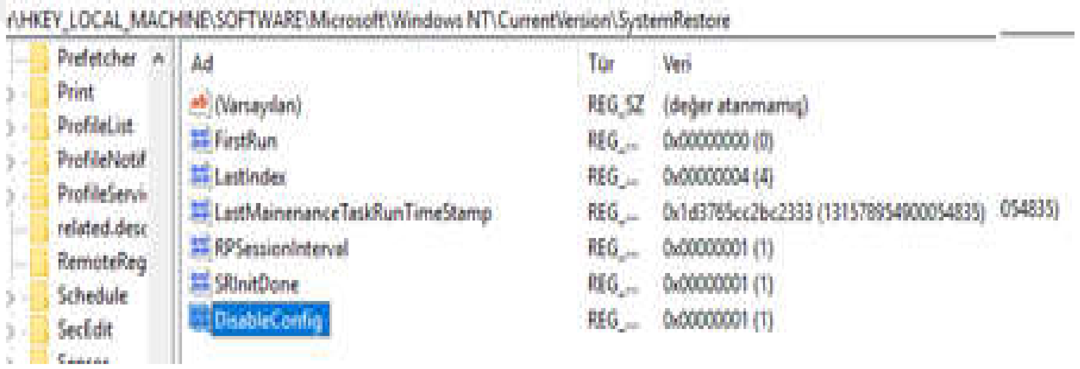
Windows System Restore Feature allows to roll back Windows to a previously working configuration in case there is a problem. In some ransomware attacks, the attacker disables the Windows System Restore Feature before starting the encryption process. To do so, the attacker runs the command in Figure 5.36.



```
C:\WINDOWS\system32>Reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\
CurrentVersion\SystemRestore" /V DisableSR /t REG_DWORD /d 1 /f
```

Figure 5.36: The command of disabling system Windows System Restore Feature.

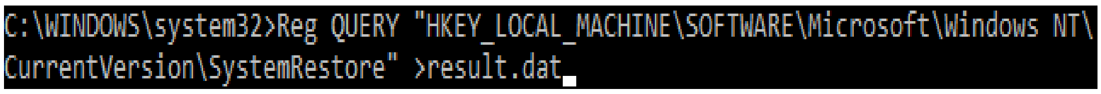
And so, the image of newly created registry sample is shown in Figure 5.37 after the command cited above is run.



```
\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SystemRestore

Prefetcher    Ad                                   Tür      Veri
Print         (Varsayılan)                         REG_SZ   (değer atanmamış)
ProfileList   FirstRun                             REG_...  0x00000000 (0)
ProfileNotif  LastIndex                            REG_...  0x00000004 (4)
ProfileServi  LastMainenanceTaskRunTimeStamp       REG_...  0x1d3765cc2bc2333 (131578954900054835)  054835)
related.desc  RPSessionInterval                    REG_...  0x00000001 (1)
RemoteReg     SRInitDone                           REG_...  0x00000001 (1)
Schedule      DisableConfig                        REG_...  0x00000001 (1)
SecEdit
```

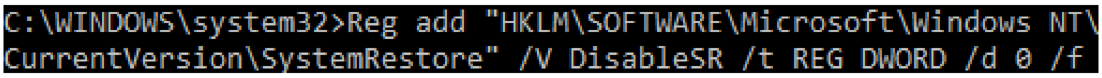
Figure 5.37: Creating a new DWORD value which disables system restore [40].

As a precaution, the output of Figure 5.38 should be monitored constantly and checked to see if the ransomware makes changes in the registry as in Figure 5.37. If any change is detected, the command in Figure 5.39 should be executed.



```
C:\WINDOWS\system32>Reg QUERY "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\
CurrentVersion\SystemRestore" >result.dat_
```

Figure 5.38: The command of permanently monitoring a registry path.



```
C:\WINDOWS\system32>Reg add "HKLM\SOFTWARE\Microsoft\Windows NT\
CurrentVersion\SystemRestore" /V DisableSR /t REG DWORD /d 0 /f
```

Figure 5.39: The command of enabling Windows System Restore Feature.

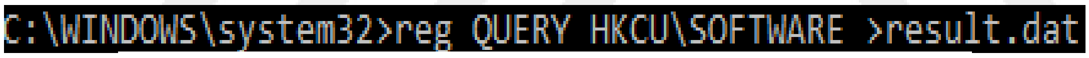
5.2.2.1.6 Machine ID and information registry

Before encryption starts, information about the machine, such as processor, hostname, RAM etc. information in the hash state, is collected to identify the device at C&C server.

When CryptoWall encrypts a file it will store the file and its path as a value in the registry. The location of the subkey is in “HKCU\Software\<unique computer id>\<random id>” format. It will then create a value for each file that it encrypts under this key [26].

The other example is Locky. It creates a registry key to store configuration information. This registry key is located at “HKCU\Software\[random]”, “HKCU\Software\Locky”, “HKCU\Software\Locky\id”, “HKCU\Software\Locky\pubkey”, “HKCU\Software\Locky\paytext”, “HKCU\Software\Locky\completed” [61]

So, the “HKCU\Software” path in the registry should be continuously monitored, as shown in Figure 5.40 and it should warn when a registry such as CryptoWall or Locky instances are created as above.



```
C:\WINDOWS\system32>reg QUERY HKCU\SOFTWARE >result.dat
```

Figure 5.40: An “HKCU\Software” path query to control registry.

5.2.2.2 File system activities

Because the main goal of the ransomware threat is encryption of files, monitoring and analyzing file system activities is very important phase. In order to monitor and analyze file system activities, “kernel32.dll” is used with hook API calls. All pre and post files and processes operations are hooked to monitor file and subprocesses created.

5.2.2.2.1 Monitoring the folders and subfolders in real time

The ransomware generally creates a copy of itself in the victim’s specific folder path under a filename generally chosen randomly and obtained from the %WINDIR%\system32 folder [39]. %Localappdata%, %ProgramData%, %User Profile%, %Temp% sections are described as encryption locations generally used by ransomware [18]. But our ransomware detection and prevention tool continuously monitor all file paths.

5.2.2.2.2 Extensions of encrypted files in system

Today, all categories of the ransomware have encrypted about 200 types of file [44]. In this phase, the most popular and widely used extensions among these file types will be checked. List of the common encrypted file extensions are ".ccc", ".cerber", ".cerber2", ".cerber3", ".crypt", ".cryptolocker", ".cryptowall", ".ecc", ".ezz", ".locky", ".micro", ".zepto", and ".encrypted" [29, 37, 44]. If the file extensions stated above are detected in the monitored file paths, they will indicate that there is a ransomware in the system.

In order to monitor file system activities; "explorer.exe" is on Windows XP machines, "svchost.exe" is on newer Windows OS distributions are monitored with all subprocesses. While monitoring these processes, if there is an attempt to create a file with one of the suspicious extensions, an alert is triggered and the process is killed immediately.

The other important point here is that the actual file extensions of the files on the computer need to be identified with tools such as the hex editor. Because the fake extension can be created by attackers for the purpose of making it difficult to be detected by security devices. By using the tool such as "ExifTool(-k)", the actual extension of the file "test.docx" appears to be ".pdf" as in Figure 5.41.

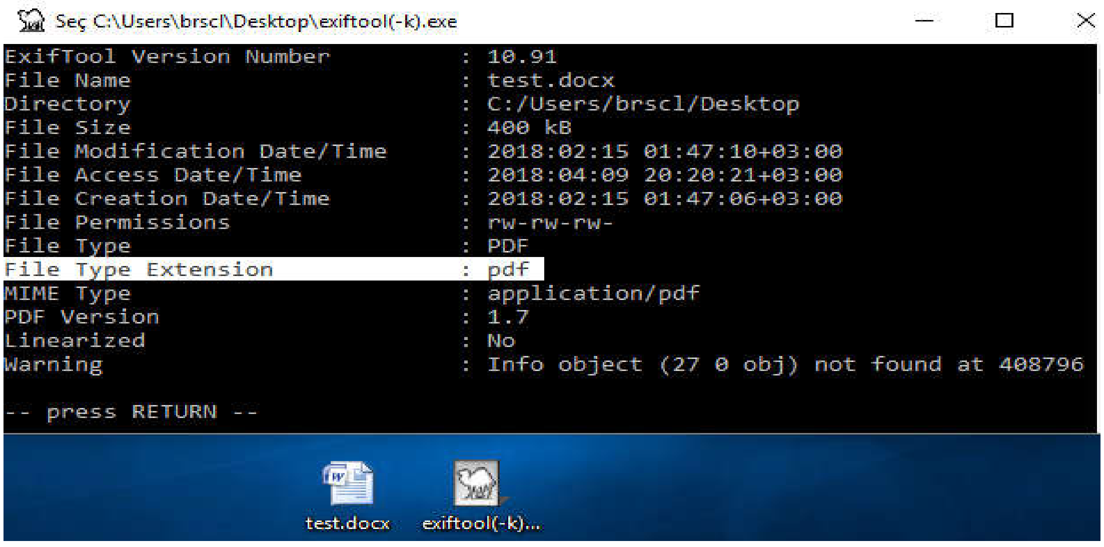
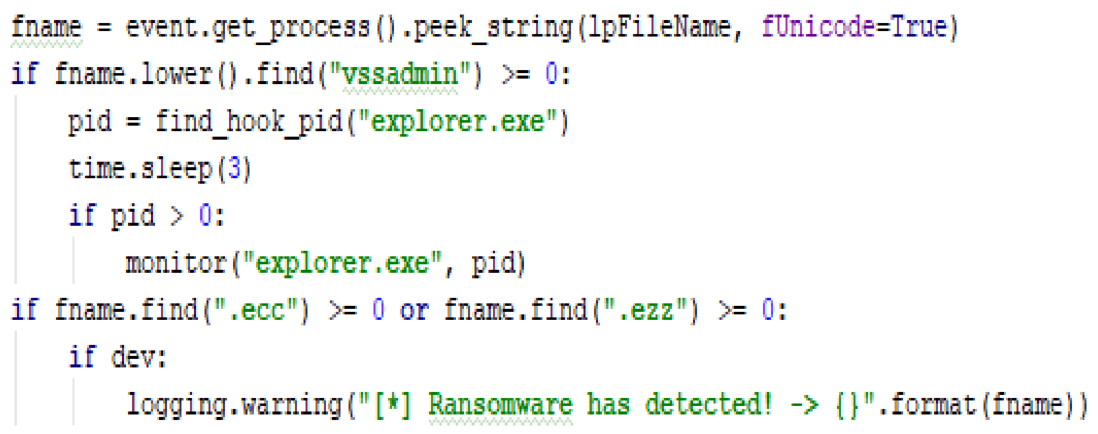

```
Seç C:\Users\brscl\Desktop\exiftool(-k).exe
ExifTool Version Number         : 10.91
File Name                       : test.docx
Directory                       : C:/Users/brscl/Desktop
File Size                       : 400 kB
File Modification Date/Time     : 2018:02:15 01:47:10+03:00
File Access Date/Time           : 2018:04:09 20:20:21+03:00
File Creation Date/Time         : 2018:02:15 01:47:06+03:00
File Permissions                : rw-rw-rw-
File Type                       : PDF
File Type Extension             : pdf
MIME Type                       : application/pdf
PDF Version                     : 1.7
Linearized                      : No
Warning                         : Info object (27 0 obj) not found at 408796

-- press RETURN --
```

Figure 5.41: An example of finding the actual extension of the file.

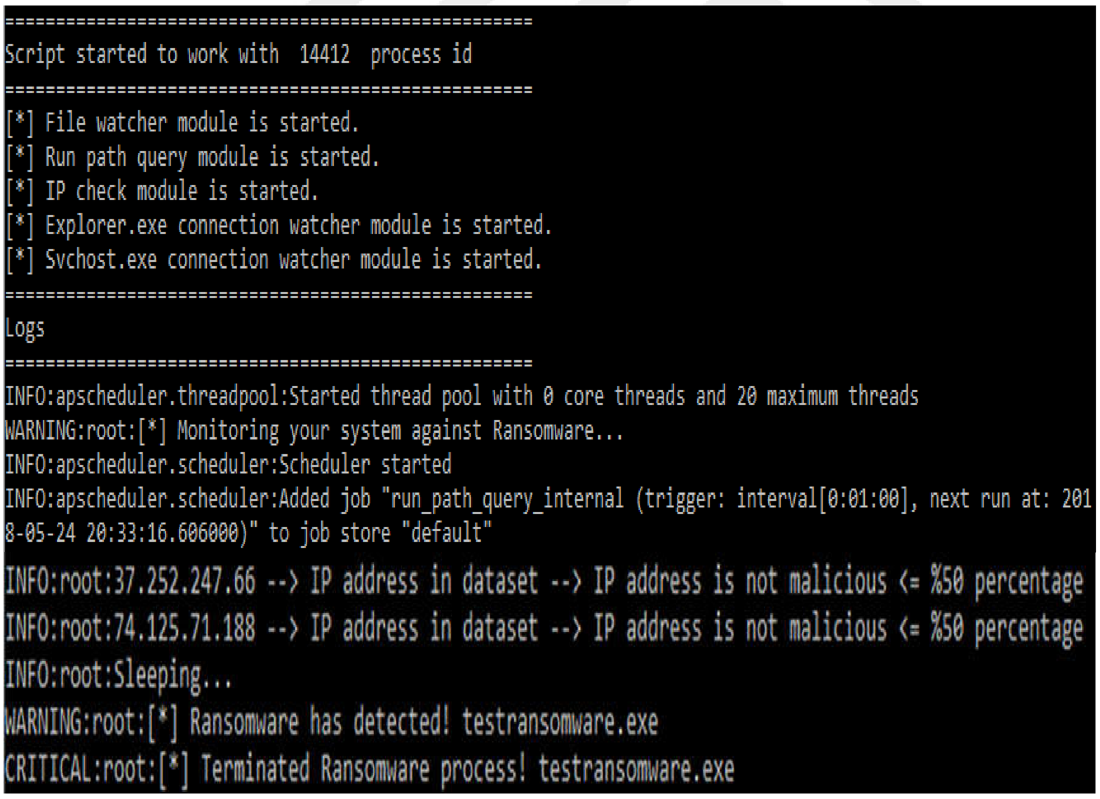
Then, within the monitored folders, the presence of the actual extensions listed above is checked as shown in Figure 5.42.



```python
fname = event.get_process().peek_string(lpFileName, fUnicode=True)
if fname.lower().find("vssadmin") >= 0:
    pid = find_hook_pid("explorer.exe")
    time.sleep(3)
    if pid > 0:
        monitor("explorer.exe", pid)
if fname.find(".ecc") >= 0 or fname.find(".ezz") >= 0:
    if dev:
        logging.warning("[*] Ransomware has detected! -> {}".format(fname))
```

Figure 5.42: An example of generating a message when a file with extension ".ecc" or ".ezz" written in Python script language is detected.

A sample output of Ransomware Detection and Prevention Tool is shown in Figure 5.43.



```
==================================================
Script started to work with  14412  process id
==================================================
[*] File watcher module is started.
[*] Run path query module is started.
[*] IP check module is started.
[*] Explorer.exe connection watcher module is started.
[*] Svchost.exe connection watcher module is started.
==================================================
Logs
==================================================
INFO:apscheduler.threadpool:Started thread pool with 0 core threads and 20 maximum threads
WARNING:root:[*] Monitoring your system against Ransomware...
INFO:apscheduler.scheduler:Scheduler started
INFO:apscheduler.scheduler:Added job "run_path_query_internal (trigger: interval[0:01:00], next run at: 201
8-05-24 20:33:16.606000)" to job store "default"
INFO:root:37.252.247.66 --> IP address in dataset --> IP address is not malicious (<= %50 percentage
INFO:root:74.125.71.188 --> IP address in dataset --> IP address is not malicious (<= %50 percentage
INFO:root:Sleeping...
WARNING:root:[*] Ransomware has detected! testransomware.exe
CRITICAL:root:[*] Terminated Ransomware process! testransomware.exe
```

Figure 5.43: A sample output of Ransomware Detection and Prevention Tool

5.3 Related Work in Literature, Comparison and Discussion

Antivirus, anti-malware or anti-ransomware are expected to have some certain features to detect or prevent malware. Especially the minimum false positive rate, protect against zero-day attacks, minimum latency and maximum performance values are the most important features. All features are mentioned below.

- Protect Against Zero-Day Attacks?
- Minimum File Loss
- Minimum False Positive Rate
- Inputs Are Enough?
- Minimum Latency and Maximum Performance
- Invulnerable to Obfuscation Techniques
- Offer Countermeasures?

In this section, developed methods and techniques against ransomware threat will be compared each other according to the features stated above. The comparison of methods and techniques used for detecting or preventing ransomware is shown in Table 5.1.

By using values such as minimum false positive, minimum latency and performance, which are quantitative and obtained from the same test data, it can be achieved with more accurate results. When examining studies about ransomware detection and prevention methods and techniques, it appears that there is hardly any quantitative data about the features. So, these values have also been expressed as qualitative.

The features which are not mentioned in related study expressed as “Unknown”. Some features taken part in related study are expressed as “YES”, and others not taken part in related study are expressed as “NO”.

Because ransomware detection and prevention tool that uses signature and anomaly-based detection methods contains the most of the features stated in Table 5.1, it is considered to provide a better solution compared to other solution methods in the literature although it may cause problems in terms of latency and performance.

Table 5.1: The comparison of methods and techniques used for detecting or preventing ransomware

Nu.	Methods and Techniques Used for Detecting or Preventing Ransomware	FEATURES						
		Protect Against Zero-Day Attacks?	Minimum File Loss	Minimum False Positive Rate	Inputs Are Enough?	Minimum Latency and Maximum Performance	Invulnerable to Obfuscation Techniques	Offer Countermeasure?
1	Our Tool (Both Signature-Based and Anomaly-Based Method)	YES	YES	YES	YES	NO	YES	YES
2	Automated Dynamic Analysis Method [13]	YES	YES	YES	YES	UNKNOWN	YES	NO
3	A Large-Scale, Automated Approach [62]	YES	YES	YES	YES	UNKNOWN	NO	NO
4	Connection-Monitor & Connection-Breaker Method [63]	YES	YES	YES	YES	NO	UNKNOWN	NO
5	Software-Defined Networking-Based Method [64]	UNKNOWN	YES	YES	YES	UNKNOWN	NO	YES
6	Machine Learned Behavioral-Based Method [2]	YES	YES	NO	YES	NO	NO	NO
7	Monitoring Process and Input / Output Events [65]	YES	NO	NO	NO	YES	NO	NO
8	Monitoring Abnormal Filesystem and Registry Activities [66]	UNKNOWN	NO	UNKNOWN	YES	NO	NO	YES
9	Cloud-Based Detection Method [6]	UNKNOWN	UNKNOWN	UNKNOWN	NO	NO	YES	NO
10	Cloud-Based Sandbox Environment Method [67]	UNKNOWN	UNKNOWN	UNKNOWN	NO	NO	YES	YES

6 CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE RESEARCH

The comprehensive review of related literature and professional reports suggests that only using the signature-based detection method fails to detect and prevent ransomware. So, we have targeted to design a ransomware detection and prevention tool which detects the normal behaviors of the ransomware and prevents these behaviors before the encryption starts. To maintain minimum false positive alarms and minimum file loss on the system, a hybrid structure by using both signature-based and anomaly-based detection methods is preferred and suggested.

In the hybrid signature-based detection phase of our Ransomware Detection and Prevention Tool which is aiming to detect and prevent ransomware at the OS level, network traffic analysis of the last created "explorer.exe" sample is performed, connected IP addresses and downloaded files are controlled through API.

In the hybrid anomaly-based detection phase, monitoring the OS processes, services, registry records, as well as behaviors on the file system are controlled.

When Ransomware Detection and Prevention Tool detects an abnormality in the signature-based and anomaly-based detection phases, it provides an early warning to the user and prevents the ransomware from running. In addition, our tool provides protection against not only the known ransomware but also zero-day ransomware that has never existed before.

The code developed in the Python programming/script language to design Ransomware Detection and Prevention Tool, in the Appendices of our study is also presented. Thus, when we encounter any ransomware trying to encrypt our files at the level of Windows OSs, we have obtained the tool that can detect and prevent with the minimum false positive alarms and minimum file loss.

This code, which is open to development, is thought to be an application of this study and also a simple part of a commercial ransomware detection and prevention tool and might be a guide to future academic studies on ransomware and other malware.

It has a modular structure and uses system features over python libraries to detect and prevent ransomware. The current version uses an open API as ransomware database for detection modules. But, by checking the hash signature through a database installed in the local machine, the ransomware detection and prevention tool will enable the system to operate in a network independent and work much more efficiently. Multi-thread structure is used so that all detection and prevention modules can run at the same time. All modules log their actions and inform the user of the script. The current version has no user interface but it can be implemented easily.

In summary, this study suggests a better perspective to the users, software developers, and security administrators about the key features of the Ransomware Detection and Prevention Tool that can be used as a solution. This tool does not require high fees for commercial software and provides the users with flexibility in taking measures against ransomware. We believe that several people involved in the software development business will be able to design ransomware detection and prevention tools like Ransomware Detection and Prevention Tool by examining the content of our study. We also think that our study will be a guide for future academic studies on ransomware and other malware.

REFERENCES

[1] **D. O'Brien** (2017). Ransomware 2017, Internet Security Threat Report, Symantec.

[2] **D. Nieuwenhuizen** (2017). A behavioral-based approach to ransomware detection, MWR Labs Whitepaper, <*https://labs.mwrinfosecurity.com/assets/resourceFiles/mwri-behavioral-ransomware-detection-2017-04-5.pdf*>, data retrieved 01.04.2018

[3] **McAfee,** (2017). McAfee Labs Threat Report.

[4] **N. Sacife, H. Carter, P. Traynor and K. R.B Butler** (2016). CryptoLock (and Drop It): Stopping Ransomware Attacks on User Data, *IEEE 36th International Conference on Distributed Computing Systems*.

[5] **K. Savage, P. Coogan, and H. Lau** (2015). The Evolution of Ransomware, Symantec, Security Response.

[6] **A. Bhardwaj, V. Avasthi, H. Sastry and G. V. B. Subrahmanyam** (2016). Ransomware Digital Extortion: A Rising New Age Threat, *Indian Journal of Science and Technology*, Vol 9(14).

[7] **M. Wecksten, J. Frick, A. Sjostrom and E. Jarpe** (2016). A Novel Method for Recovery from Crypto Ransomware Infections, *2nd IEEE International Conference on Computer and Communications*.

[8] **M. H. U. Salvi, & M. R. V. Kerkar** (2016). Ransomware: A cyber extortion, *Asian Journal of Convergence in Technology*, 2(3).

[9] **A. Zahra and A. S.Munam** (2017). IoT Based Ransomware Growth Rate Evaluation and Detection Using Command and Control Blacklisting, *Proceedings of the 23rd International Conference on Automation & Computing*, University of Huddersfield, Huddersfield, UK, 7-8.

[10] **CheckPoint** (2017). Ransomware: Attack Trends, Prevention, And Response, White Paper.

[11] **A. Liska and T. Gallo** (2016). Ransomware: Defending Against Digital Extortion, O'Reilly Media, Inc.

[12] **Vadim Kotov, Mantej Singh Rajpal** (2014). In-Depth Analysis of the Most Popular Malware Families, Bromium, Understanding Crypto-Ransomware Report.

[13] **D. Sgandurra, L. Muñoz-González, R. Mohsen, & E.C. Lupu** (2016). Automated dynamic analysis of ransomware: Benefits, limitations and use for detection.

[14] **N. Hampton and Z. A. Baig** (2015). Ransomware: Emergence of the cyber-extortion menace, *Aust. Inf. Secur. Manag. Conf.*, vol. 2015, pp. 47–56.

[15] **A. Liska,T. Gallo** (2016). Ransomware: Defending Against Digital Extortion, O'Reilly Media, Inc. First Ed.

[16] **Kaspersky Lab** (2016). Kaspersky Security Bulletin, Story of The Year: The Ransomware Revolution, Report.

[17] **C. Krzysztof and M. Wojciech** (2016). Using Software-Defined Networking for Ransomware Mitigation: The Case of CryptoWall, *Network Forensics And Surveillance For Emerging Networks*.

[18] **A. Adamov, A. Carlsson** (2017). The state of ransomware. Trends and mitigation techniques, *2017 IEEE East-West Design & Test Symposium*, vol. 00, no., pp. 1-8, Doi:10.1109/EWDTS.2017. 8110056.

[19] **B. Heater** (2016). How ransomware conquered the world, PC Magazine Digital Edition.

[20] **Proofpoint** (2017). 2017 Q3 Threat Report.

[21] **Malwarebytes** (2017). Cybercrime tactics and techniques,Report, Q1.

[22] **T. Anjana** (2017). Discussion On Ransomware, Wannacry Ransomware and Cloud Storage Services Against Ransom Malware Attacks, *The International Journal for Research Trends and Innovation*, Vol.2, Issue 6, ISSN: 2456-3315.

[23] **S. Ragan** (2017). Badrabbit ransomware attacks multiple media outlets", <*https://www.csoonline.com/article/3234691/security/badrabbit-ransomware-attacks-multiple-media-outlets.html*>, data retrieved 04.02.2018.

[24] **Symantec** (2014). CryptoDefense, the CryptoLocker Imitator, Makes Over $34,000 in One Month, Symantec Security Response.

[25] **E. Dunn John** (2017). The 12 worst types of ransomware - we name the internet's nastiest extortion malware, <*https://www.computerworlduk .com/galleries/security/worst-ransomware-attacks-we-name-internets-nastiest-extortion-malware-3641916/*>, data retrieved 04.03.2018.

[26] **BleepingComputer** (2014). CryptoWall and Help_Decrypt Ransomware Information Guide and FAQ, <*https://www.bleepingcomputer.com/ virus-removal/cryptowall-ransomware-information*> data retrieved 07.02.2018.

[27] **Calyptix** (2015). Critroni Ransomware Decryption: Not an Option, <*https://www.calyptix.com/malware/critroni-ransomware-decryption-not-an-option/*>, data retrieved 02.01.2018.

[28] **Malwarebytes** (2016). Look into Locky Ransomware <*https://blog. malwarebytes.com/threat-analysis/2016/03/look-into-locky/*>, data retrieved 21.02.2018.

[29] **BleepingComputer** (2016). Locky Ransomware Information, Help Guide, and FAQ, <*https://www.bleepingcomputer.com/virus-removal/locky-ransomware-information-help*>, data retrieved 24.01.2018.

[30] **StreamScan** (2017). WannaCry Ransomware Analysis, White Paper.

[31] **D. Darragh** (2017). How to detect the presence of WannaCry Ransomware and SMBv1 servers on your network, <*https://www.netfort.com/blog/ detect-wannacry-ransomware/*>, data retrieved 16.03.2018.

[32] **Symantec Security Response** (2017). Petya ransomware outbreak: Here's what you need to know, <*https://www.symantec.com/blogs/threat-intelligence/ petya-ransomware-wiper*>, data retrieved 27.02.2018.

[33] **Bi Zone** BadRabbit*,* <*https://www.bi.zone/upload/medialibrary/ 332/3325249 7baa074b092cb1a5400cfee77.pdf*>, data retrieved 06.02.2018.

[34] **StreamScan** (2017). Analysis of Bad Rabbit Ransomware, White Paper.

[35] **Microsoft** (2017). Syskey.exe utility is no longer supported in Windows 10 version 1709 and Windows Server version 1709, <*https://support. microsoft.com/tr-tr/help/4025993/syskey-exe-utility-is-no-longer-supported-in-windows-10-version-1709*>, data retrieved 15.03.2018.

[36] **L. Abrams** (2014). CryptoDefense and How_Decrypt Ransomware Information Guide and FAQ*,* <*https://www.bleepingcomputer.com/ virus-removal/cryptodefense-ransomware-information*>, data retrieved 02.04.2018.

[37] **Webroot** (2017). MSP Guide: Stopping Crypto Ransomware Infections in SMBs, 16 Easy Actions for MSPs, White Paper.

[38] **S. Ed** (2008). Built-in Windows commands to determine if a system has been hacked", <*http://searchsecurity.techtarget.com/tip/Built-in-Windows-commands-to-determine-if-a-system-has-been-hacked*>, data retrieved 30.01.2018.

[39] **A. Anubhav and R. Ellur** (2016). Cerber: Analyzing a Ransomware Attack Methodology To Enable Protection, Threat Research, Advanced Malware, FireEye.

[40] **Microsoft** (n.d.). How to Disable the System Restore Configuration User Interface", <*https://support.microsoft.com/en-us/help/283073/how-to-disable-the-system-restore-configuration-user-interface*>, data retrieved 15.01.2018.

[41] **Microsoft** (n.d.). "Cipher.exe Security Tool for the Encrypting File System", <*https://support.microsoft.com/en-au/help/298009/cipher-exe-security -tool-for-the-encrypting-file-system*>, data retrieved 13.02.2018.

[42] **D. Shinder** (2003). Use cipher.exe for command line encryption, TechRepublic, <*https://www.techrepublic.com/article/use-cipherexe-for-command-line-encryption/*>, data retrieved 24.02.2018.

[43] **EarthLink Symantec Page** (2017). <*https://www.symantec.com/security_ response/earthlink_writeup.jsp?docid=2017-100516-2840-99*>, data retrieved 13.02.2018.

[44] **File-Extensions** (n.d.). Common ransomware encrypted files, <*https://www.file-extensions.org/common/ransomware*>, data retrieved 12.02.2018.

[45] **N. Idika, A.P.Mathur** (2007). A Survey of Malware Detection Techniques, *Department of Computer Science, Purdue University*, 48.

[46] **Moser, C. Kruegel, and E.Kirda** (2007). Limits of Static Analysis for Malware Detection, *23rd Annual Computer Security Applications Conference*, 1063-9527/07 $25.00 © 2007 IEEE, DOI 10.1109/ACSAC.2007.21

[47] **V.Jyothsna, V.V.R. Prasad, K.M.Prasad** (2011). A Review of Anomaly based Intrusison Detection Systems, *International Journal of Computer Applications* (0975–8887), Volume 28-No.7, August 2011.

[48] **I.Firdausi, C.Lim, A.Erwin, and A.S.Nugroho** (2010). Analysis of Machine Learning Techniques Used In Behavior-Based Malware Detection, *Advances in Computing, Control and Telecommunication Technologies (ACT), 2010 Second International Conference on* (pp. 201-203). IEEE

[49] **W.Robertson, G.Vigna, C.Kruegel, and R.A.Kemmerer** (2006). Using Generalization and Characterization Techniques in the Anomaly-based Detection ofWeb Attacks, *Proceedings of the Network and Distributed System Security Symposium*, NDSS 2006, San Diego, California, USA

[50] **Virustotal,** (n.d.). <*https://developers.virustotal.com/v2.0/reference#ip-address -report*>, data retrieved 04.02.2018.

[51] **L. Pevzner** (2016) Digging Deeper: How Ransomware and Malware use Microsoft Windows' Known Binaries, <*https://blog.checkpoint.com/ 2016/04/26/how-ransomware-and-malware-use-microsoft-windows-known-binaries/*>, data retrieved 12.02.2018.

[52] **Y. Sela** (2015). Anatomy of CryptoWall 3.0 Virus – a look inside ransomware code & tactics, <*https://www.sentinelone.com/blog/ anatomy-of-cryptowall-3-0-a-look-inside-ransomwares-tactics/*>, data retrieved 09.02.2018.

[53] **L. Abrams** (2015). How to determine what services are running under a svchost.exe process", <*https://www.bleepingcomputer.com/tutorials/ list-services-running-under-svchostexe-process/#procexp*>, data retrieved 19.02.2018.

[54] **S. Pilici** (2017). How to remove SvcHost.exe Malware (Virus Removal Guide), *<https://malwaretips.com/blogs/svchost-exe-virus-removal/>,* data retrieved 17.02.2018.

[55] **S. Travis** (2017). WannaCelebrate – How to Protect Against WannaCry Ransomware Tripwire, <*www.tripwire.com/state-of-security/security-data-protection/cyber-security/wannacelebrate-protect-wannacry-ransomware/*>, data retrieved 22.02.2018.

[56] **Microsoft** (2017). How to detect, enable and disable SMBv1, SMBv2, and SMBv3 in Windows and Windows Server*,* <*https://support.microsoft. com/en-au/help/2696547/how-to-detect-enable-and-disable-smbv1-smbv2-and-smbv3-in-windows-and*>, data retrieved 24.02.2018.

[57] **The Network Encyclopedia** (n.d.). <*http://www.thenetworkencyclopedia. com/entry/server-service/*>, data retrieved 12.02.2018.

[58] **K. K. Gagneja** (2017). Knowing the ransomware and building a defense against it - specific to healthcare institutes, *2017 Third International*

Conference on Mobile and Secure Services (MobiSecServ), Miami Beach, FL, 2017, pp. 1-5.

[59] **L. Abrams** (2015). Why Everyone Should disable VSSAdmin.exe Now!, <*https://www.bleepingcomputer.com/news/security/why-everyone-should-disable-vssadmin-exe-now/*>, data retrieved 10.02.2018.

[60] **Microsoft** (n.d.). BCDEdit/set, <*https://docs.microsoft.com/tr-tr/windows-hardware/drivers/devtest/bcdedit—set*>, data retrieved 03.02.2018.

[61] **Proofpoint** (2016). Dridex Actors Get In the Ransomware Game With Locky, Proofpoint Staff

[62] **A. Kharraz, S. Arshad, C. Mulliner, W. K. Robertson, and E. Kirda** (2016). UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware. In *USENIX Security Symposium* (pp. 757-772).

[63] **M. M. Ahmadian, H. R. Shahriari, and S.M. Ghaffarian** (2015). Connection-monitor & connection-breaker: A novel approach for prevention and detection of high survivable ransomwares. In *Information Security and Cryptology (ISCISC), 12th International Iranian Society of Cryptology Conference on* (pp. 79-84). IEEE.

[64] K**. Cabaj, M. Gregorczyk, and W. Mazurczyk** (2017). Software-defined networking-based crypto ransomware detection using HTTP traffic characteristics. *Computers & Electrical Engineering*.

[65] **S. Song, B. Kim, and S. Lee** (2016). The effective ransomware prevention technique using process monitoring on android platform. *Mobile Information Systems*.

[66] **P. Zavarsky, and D. Lindskog** (2016). Experimental analysis of ransomware on windows and android platforms: Evolution and characterization. *Procedia Computer Science*, *94*, 465-472.

[67] **A. Bhardwaj** (2017). Ransomware: A rising threat of new age digital extortion. In *Online Banking Security Measures and Data Protection* (pp. 189-221). IGI Global.

APPENDICES

APPENDIX A.1 : The Main Part of Ransomware Detection and Prevention Tool

```
# Ransomware Detection and Prevention Tool
# Designed and Coded by:
# Baris CELIKTAS <celiktas16@itu.edu.tr>
# Bilal Gultekin <gultekinb@itu.edu.tr>

import os
import filewatcher as fw
import othercontrols as otc
import threading
import sys
import signal
import time
import logging

def isUserAdmin():
    if os.name == 'nt':
        import ctypes
        # WARNING: requires Windows XP SP2 or higher!
        try:
            return ctypes.windll.shell32.IsUserAnAdmin()
        except:
            traceback.print_exc()
    else:
        raise RuntimeError, "[*] Unsupported operating system for this module: %s" % (os.name,)
        sys.exit(1)

def cls():
    if sys.platform == 'linux-i386' or sys.platform == 'linux2':
        os.system("clear")
    elif sys.platform == 'win32':
        os.system("cls")
    else:
        os.system("cls")

def main():

    pid = os.getpid()

    # to see every output
    logging.basicConfig(stream=sys.stdout, level=logging.INFO)

    signal.signal(signal.SIGINT, fw.stop)
    signal.signal(signal.SIGTERM, fw.stop)

    # clear the terminal
    cls()
    print "=========================================================="
    print "Script started to work with ", pid, " process id"
    print "=========================================================="

    # check if it is admin, fw requires to administrator privileges
    if isUserAdmin() == 0:
        print "[*] You must have an administrator privilege."
        print "[*] Script is stopping"
        sys.exit(1)

    # create threads
    global thread1

    # start
    try:
        # give info
        print "[*] File watcher module is started."
        print "[*] Run path query module is started."
        print "[*] IP check module is started."
        print "[*] Explorer.exe connection watcher module is started."
        print "[*] Svchost.exe connection watcher module is started."
        print "=========================================================="
        print "Logs"
        print "=========================================================="

        # start them
        thread1 = threading.Thread(target=fw.main)
        thread1.daemon = True
        thread1.start()

        thread2 = threading.Thread(target=otc.run_path_query)
        thread2.daemon = True
        thread2.start()

        thread3 = threading.Thread(target=otc.ip_check)
        thread3.daemon = True
        thread3.start()

        thread4 = threading.Thread(target=otc.explorer_exe_TCP_IP_connection)
        thread4.daemon = True
        thread4.start()

        thread5 = threading.Thread(target=otc.svchost_exe_TCP_IP_connection)
        thread5.daemon = True
        thread5.start()

    except KeyboardInterrupt:
        thread1._Thread__stop()

    # since winappdbg has some problems with exiting while debugging
    # we should use threads in daemon mode
    # and keep caller alive by using sleep
    while True:
        time.sleep(1)

if __name__ == "__main__":
    main()
```

Figure A.1 : The Main Part

APPENDIX A.2 : The Part of Monitoring Files of the Ransomware Detection and Prevention Tool

```
# For this part of the Ransomware Detection and Prevention Tool
# Mert Sarıca's Cryptolocker Detection & Killer Utility v1.0
# has been utilized and developed.

# Designed and Coded by:
# Baris CELIKTAS <celiktas16@itu.edu.tr>
# Bilal Gultekin <gultekinb@itu.edu.tr>

import os
import binascii
import time
from winappdbg import Process, System, Debug, EventHandler
import locale
import re
import platform
import subprocess
import sys
import signal
import re
import threading
import Queue  # This is thread safe
import time
from threading import *
import datetime
import logging
from winappdbg.win32 import ERROR_ACCESS_DENIED
reload(sys)
sys.setdefaultencoding('iso-8859-9')

# Global Variables
dev = 0
turkish = 0
gpid = 0
oldpid = 0
fileLock = Semaphore(value=1)
filename = r"C:\Cryptokiller\log.txt"
filename_err = r"C:\Cryptokiller\error.txt"
folder = r"C:\Cryptokiller"

if dev:
    logging.info("Platform: {}".format(platform.release()))

if platform.release().find("XP") >= 0:
    xp = 1
else:
    xp = 0


def excepthook(*args):
    sys.exit(1)
```

```
# Reference: http://stackoverflow.com/questions/19672352/how-to-run-python-script-with-elevated-
privilege-on-windows

def isUserAdmin():
    if os.name == 'nt':
        import ctypes
        # WARNING: requires Windows XP SP2 or higher!
        try:
            return ctypes.windll.shell32.IsUserAnAdmin()
        except:
            traceback.print_exc()
    else:
        if turkish:
            raise RuntimeError, "[*] Unsupported operating system for this module: %s" % (os.name,)
        else:
            raise RuntimeError, u"[*] Desteklenmeyen işletim sistemi: %s" % (os.name,)
        sys.exit(1)

def log(txt):
    if not os.path.exists(folder):
        os.mkdir(folder)

    try:
        fileLock.acquire()
        FILE = open(filename, "a")
    except IOError:
        if console:
            logging.critical("[-] Can not open log.txt!")
        log("[-] Hata: Can not open log.txt!")
        sys.exit(1)

    start = str(datetime.datetime.now())
    txt = "(" + start + ")" + " " + txt + "\r\n"
    FILE.writelines(txt)
    FILE.close()
    fileLock.release()

def error_log(txt):
    try:
        fileLock.acquire()
        FILE = open(filename_err, "a")
    except IOError:
        if console:
            logging.critical("[-] Can not open error.txt!")
        error_log("[-] Can not open error.txt!")
        sys.exit(1)

    start = datetime.datetime.now()
    txt = "(" + str(start) + ")" + " " + txt + "\r\n"
    FILE.writelines(txt)
    FILE.close()
    fileLock.release()

def get_svchost_pid():
    cmd = 'tasklist /svc /fi "imagename eq svchost.exe" /fi "services eq LanmanServer"'
    p = subprocess.Popen(cmd,
                         stdout=subprocess.PIPE,
                         stderr=subprocess.STDOUT,
                         stdin=subprocess.PIPE)
    result = p.communicate()[0]
    re1 = '.*?'  # Non-greedy match on filler
    re2 = '(\\d+)'  # Integer Number 1

    rg = re.compile(re1 + re2, re.IGNORECASE | re.DOTALL)
    m = rg.search(result)
    if m:
        int1 = m.group(1)
        return int1

def get_explorer_pid():
    # Request debug privileges.
    System.request_debug_privileges()

    # Scan for running processes.
    system = System()
    try:
        system.scan_processes()
        # system.scan_process_filenames()
    except WindowsError:
        system.scan_processes_fast()

    # For each running process...
    for process in system.iter_processes():
        try:

            pid = process.get_pid()

            if pid in (0, 4, 8):
                continue

            if dev:
                logging.warning("* Process:", process.get_filename() + "Pid:" + pid + "Time:" +
process.get_running_time())
            if process.get_filename() == "explorer.exe":
                if process.get_running_time() < 300000:
                    return pid

        # Skip processes we don't have permission to access.
        except WindowsError, e:
            if e.winerror == ERROR_ACCESS_DENIED:
                continue
            raise

    sys.exit(1)

def cls():
    if sys.platform == 'linux-i386' or sys.platform == 'linux2':
        os.system("clear")
    elif sys.platform == 'win32':
        os.system("cls")
    else:
        os.system("cls")

def kill_cryptolocker(pname, pid):
    # Instance a Process object.
    process = Process(pid)

    # Kill the process.
    process.kill()
    proc = "(" + pname + ":" + str(gpid) + ")"

    txt = "[*] Terminated Ransomware process! " + proc
    log(txt)
    logging.critical("[*] Terminated Ransomware process! " + proc)

    sys.exit(1)


class MyEventHandler(EventHandler):
    global xp

    # Here we set which API calls we want to intercept
    apiHooks = {

        # Hooks for the kernel32 library
        'kernel32.dll': [
            #  Function            Parameters
            ('CreateFileA', 7),
            ('CreateFileW', 7),
            ('CreateProcessA', 10),
            ('CreateProcessW', 10),
        ],
    }

    # Now we can simply define a method for each hooked API.
    # Methods beginning with "pre_" are called when entering the API,
    # and methods beginning with "post_" when returning from the API.

    def pre_CreateProcessA(self, event, ra, lpApplicationName, lpCommandLine, lpProcessAttributes,
lpThreadAttributes,
                           bInheritHandles, dwCreationFlags, lpEnvironment, lpCurrentDirectory,
lpStartupInfo,
                           lpProcessInformation):
        if dev:
            self.__print_ansi(event, "CreateProcessA", lpApplicationName)

    def pre_CreateProcessW(self, event, ra, lpApplicationName, lpCommandLine, lpProcessAttributes,
lpThreadAttributes,
                           bInheritHandles, dwCreationFlags, lpEnvironment, lpCurrentDirectory,
lpStartupInfo,
                           lpProcessInformation):
        if dev:
            self.__print_unicode(event, "CreateProcessW", lpApplicationName)

    def post_CreateProcessA(self, event, retval):
        if dev:
            logging.debug("XP:" + xp)
        if xp:
            time.sleep(3)
            pid = find_hook_pid("explorer.exe")
            if pid > 0:
                monitor("explorer.exe", pid)

    def post_CreateProcessW(self, event, retval):
        if dev:
            logging.debug("XP:" + xp)
        if xp:
            time.sleep(3)
            pid = find_hook_pid()
        if pid > 0:
            monitor("explorer.exe", pid)

def pre_CreateFileA(self, event, ra, lpFileName, dwDesiredAccess,
                    dwShareMode, lpSecurityAttributes, dwCreationDisposition,
                    dwFlagsAndAttributes, hTemplateFile):

    if dev:
        self.__print_ansi(event, "CreateFileA", lpFileName)

    if int(dwCreationDisposition) == 3:

        fname = event.get_process().peek_string(lpFileName, fUnicode=False)
        if fname.lower().find("vssadmin") >= 0:
            pid = find_hook_pid("explorer.exe")
            time.sleep(3)
            if pid > 0:
                monitor("explorer.exe", pid)
        if fname.find(".sifreli") >= 0 or fname.find(".encrypted") >= 0:
            if dev:
                logging.debug("[*] Ransomware has detected! -> {}".format(fname))

            pid = event.get_pid()
            pname = event.get_process().get_filename()
            pname = pname.split("\\")[2]
            proc = "(" + pname + ":" + str(pid) + ")"

            txt = "[*] Ransomware has detected! " + proc
            log(txt)
            logging.warning("[*] Ransomware has detected! " + proc);

            kill_cryptolocker(pname, pid)

def pre_CreateFileW(self, event, ra, lpFileName, dwDesiredAccess,
                    dwShareMode, lpSecurityAttributes, dwCreationDisposition,
                    dwFlagsAndAttributes, hTemplateFile):

    if dev:
        self.__print_unicode(event, "CreateFileA", lpFileName)

    if int(dwCreationDisposition) == 3:

        fname = event.get_process().peek_string(lpFileName, fUnicode=True)
        if fname.lower().find("vssadmin") >= 0:
            pid = find_hook_pid("explorer.exe")
            time.sleep(3)
            if pid > 0:
                monitor("explorer.exe", pid)
        if fname.find(".ecc") >= 0 or fname.find(".ezz") >= 0:
            if dev:
                logging.warning("[*] Ransomware has detected! -> {}".format(fname))

            pid = event.get_pid()
            pname = event.get_process().get_filename()
            pname = pname.split("\\")[2]
            proc = "(" + pname + ":" + str(pid) + ")"

            txt = "[*] Ransomware has detected! " + proc
            log(txt)
            logging.warning("[*] Ransomware has detected! " + proc)
            kill_cryptolocker(pname, pid)

    # Some helper private methods...

    def __print_ansi(self, event, tag, pointer):
        string = event.get_process().peek_string(pointer, fUnicode=False)
        tid = event.get_tid()
        logging.warning("%d: %s: %s" % (tid, tag, string))

    def __print_unicode(self, event, tag, pointer):
        string = event.get_process().peek_string(pointer, fUnicode=True)
        tid = event.get_tid()
        logging.warning("%d: %s: %s" % (tid, tag, string))


def find_hook_pid(procname):
    global gpid
    global xp
    global oldpid

    s = System()
    s.request_debug_privileges()

    try:
        s.scan_processes()
        s.scan_process_filenames()
    except WindowsError:
        s.scan_processes_fast()

    pid_list = s.get_process_ids()
    pid_list.sort(reverse=True)

    if not pid_list:
        logging.warning("Unknown error enumerating processes!")
        # s = raw_input()
        sys.exit(1)

    for pid in pid_list:
        p = s.get_process(pid)
        fileName = p.get_filename()
        fname = str(fileName).lower()
        if dev:
            logging.warning("Process:" + fname + "Pid:" + pid)
        if fname.find(procname) >= 0:
            if int(pid) != int(gpid):
                oldpid = gpid
                gpid = pid
                if procname.find("svchost.exe") >= 0:
                    gpid = int(get_svchost_pid())
                    return gpid
                elif procname.find("explorer.exe") >= 0:
                    gpid = int(get_explorer_pid())
                    return gpid
                else:
                    return pid
    return 0

def usage():
    print "Usage: python cryptokiller.py [hidden]\n"
try:
    debug = Debug(MyEventHandler())
except:
    logging.warning("Can't interrupt the kernel process")
    logging.warning("There can be only one filewatcher running script, you should kill others")
finally:
    debug.stop()

def monitor(procname, pid):
    if dev:
        logging.warning(procname + ":" + str(pid))

    # Instance a Debug object.
    if pid > 0:
        # debug = Debug( MyEventHandler() )
        try:
            debug.stop(True)
            # Attach to a running process.
            debug.attach(pid)

            # Wait for the debugee to finish.
            debug.loop()

        # Stop the debugger.
        finally:
            debug.stop()

def stop(*args):
    logging.warning("stopping...")
    debug.stop()

signal.signal(signal.SIGINT, stop)
signal.signal(signal.SIGTERM, stop)

def main():
    import sys
    global turkish

    try:
        if dev:
            logging.warning("Locale:" + locale.getdefaultlocale()[0])
        if locale.getdefaultlocale()[0].lower() == "tr_tr":
            turkish = 0
    except:
        turkish = 0

    if isUserAdmin() == 0:
        logging.critical("[*] You must have an administrator privilege.")
        sys.exit(1)

    txt = "[*] Monitoring your system against Ransomware..."
    log(txt)
    logging.warning(txt)

    try:
        if xp:
            monitor("explorer.exe", find_hook_pid("explorer.exe"))
        else:
            monitor("svchost.exe", find_hook_pid("svchost.exe"))
    except KeyboardInterrupt:
        try:
            thread1.stop()
        except:
            pass
        stop()

    stop()


# When invoked from the command line,
# the first argument is a process ID,
# the second argument is a DLL filename.

if __name__ == "__main__":
    global thread1

    try:
        thread1 = threading.Thread(target=main)
        thread1.start()
    except KeyboardInterrupt:
        thread1._Thread__stop()
        stop()

    stop()
```

Figure A.2 : The Part of Monitoring Files

APPENDIX A.3 : Other Parts of the Detection and Prevention Tool

```
# Designed and Coded by:
# Baris CELIKTAS <celiktas16@itu.edu.tr>
# Bilal Gultekin <gultekinb@itu.edu.tr>

import numpy as np
import requests
import re
import time
import os
import platform
import subprocess
import sys
from winappdbg import Process, System, Debug, EventHandler
import hashlib
from py_essentials import hashing as hs
import ipaddress
from collections import namedtuple
from apscheduler.scheduler import Scheduler
import logging

API_KEY = "GET API KEY FROM VIRUSTOTAL"

black_hashes = []
white_hashes = []

black_ips = []
white_ips = []

def main():
    # schedule run_path_query for every 60 seconds
    run_path_query()
    netstat_connection_control()
    explorer_exe_TCP_IP_connection()
    svchost_exe_TCP_IP_connection()
    ip_check()

def ip_check():

    suspicious_connections = []

    while True:

        suspicious_connections = fetch_suspicious_connections()

        count = 1

        for connection in suspicious_connections:
            remote_ip, remote_port = connection.remote_address.split(":")

            # if ip is already queried and malicious then kill pid
            if remote_ip in white_ips:
                # do nothing
                pass

            elif remote_ip in black_ips:
                kill_pid(connection.pid)

                logging.warning(remote_ip + " --> " + response_dict["verbose_msg"] + " --> IP address ")
            else:
                try:

                    url = 'https://www.virustotal.com/vtapi/v2/ip-address/report'
                    params = {'apikey': API_KEY, 'ip': remote_ip}
                    response = requests.get(url, params=params)

                    if response.status_code != 200:
                        logging.error("There is an issue about Virustotal API")
                        continue

                    response_dict = response.json()

                    url_list = []
                    url_list = url_list + (response_dict["detected_urls"] if "detected_urls" in response_dict
else [])
                    url_list = url_list + (response_dict["detected_referrer_samples"] if
"detected_referrer_samples" in response_dict else [])
                    url_list = url_list + (response_dict["detected_downloaded_samples"] if
"detected_downloaded_samples" in response_dict else [])
                    url_list = url_list + (response_dict["detected_communicating_samples"] if
"detected_communicating_samples" in response_dict else [])

                    positives = float(0)
                    total = float(0)

                    if len(url_list) == 0:
                        percentage = 0
                    else:
                        for url in url_list:
                            positives += url["positives"]
                            total += url["total"]

                        percentage = positives / total

                    if percentage > 0.7:
                        kill_pid(connection.pid)
                        black_ips.append(remote_ip)

                        logging.warning(remote_ip + " --> " + response_dict["verbose_msg"] + " --> IP
address is absolutely malicious (minimum false-positive score > %70 percentage)")
                    elif percentage > 0.5:
                        white_ips.append(remote_ip)

                        logging.warning(remote_ip + " --> " + response_dict["verbose_msg"] + " --> IP
address is probably malicious (medium false-positive %70 percentage > score > %50 percentage)")
                    else:
                        white_ips.append(remote_ip)

                        logging.info(remote_ip + " --> " + response_dict["verbose_msg"] + " --> IP address is
not malicious <= %50 percentage")

                except (KeyError, ValueError) as err:
                    logging.error(err)
                except:
                    logging.error("Unexpected error:", sys.exc_info()[0])

                count += 1
                if (count % 3 == 0):
                    logging.info("Sleeping...")
            time.sleep(60)

        time.sleep(20)

def fetch_suspicious_connections():
    suspicious_connections = []
    suspicious_connections = suspicious_connections + explorer_exe_TCP_IP_connection()
    suspicious_connections = suspicious_connections + svchost_exe_TCP_IP_connection()
    suspicious_connections = netstat_connection_control('netstat -ano | findstr "ESTABLISHED"')

    return suspicious_connections

def kill_pid(pid):
    cmd = "taskkill /F /pid " + str(pid)
    process = subprocess.Popen(cmd, stdout=subprocess.PIPE, stderr=None, shell=True)
    output = process.communicate()
    logging.debug(output[0])
    return output[0]

def del_path(path):
    cmd = "del {0} /F /Q".format(path)
    process = subprocess.Popen(cmd, stdout=subprocess.PIPE, stderr=None, shell=True)
    output = process.communicate()
    logging.debug(output[0])
    return output[0]

def netstat_connection_control(cmd):
    process = subprocess.Popen(cmd, stdout=subprocess.PIPE, stderr=None, shell=True)
    output = process.communicate()

    def parseLine(line, namedTuple):
        # parse columns
        fields = line.split()

        # if there is no 4 columns then assume it is not valid line
        if len(fields) == len(namedTuple._fields):
            return namedTuple(fields[0], fields[1], fields[2], fields[3], fields[4])

    # read file and get content
    content = output[0]

    # split lines
    lines = content.split("\r\n")

    # fill suspicious connections into this list
    suspicious_connections = []

    # iterate lines
    for line in lines:
        # define row as namedtuple
        netstatRow = namedtuple('netstatRow', ['protocol', 'local_address', 'remote_address', 'status',
'pid'])

        # parse line
        line_tupple = parseLine(line, netstatRow)

        # if line_tupple is not None
        if line_tupple:
            # if it is like [::]:0 line, then pass it
            if line_tupple.remote_address.find("::") != -1:
```

```
            continue

        # parse remote address as ip and port
        remote_ip, remote_port = line_tupple.remote_address.split(":")

        # check if ip is not private
        if not ipaddress.ip_address(unicode(remote_ip, "utf-8")).is_private:
            suspicious_connections.append(line_tupple)

    return suspicious_connections

def explorer_exe_TCP_IP_connection():
    explorer_exe_PID = get_program_pid("explorer.exe")
    cmd = "netstat -ano | findstr " + str(explorer_exe_PID)

    return netstat_connection_control(cmd)

def get_program_pid(program_name):
    cmd = 'tasklist /svc /fi "imagename eq {0}"'.format(program_name)
    process = subprocess.Popen(cmd, stdout=subprocess.PIPE, stderr=None, shell=True)
    output = process.communicate()

    # split lines
    lines = output[0].split("\r\n")
    pid = None

    for line in lines:
        # strip spaces
        line = line.strip()

        # check if it is explorer.exe
        if line.startswith(program_name):
            line_columns = line.split()
            pid = line_columns[1]

    return pid

def svchost_exe_TCP_IP_connection():
    svchost_exe_PID = get_program_pid("svchost.exe")
    cmd = "netstat -ano | findstr " + str(svchost_exe_PID)

    return netstat_connection_control(cmd)

def run_path_query():
    # define work
    def run_path_query_internal():
        cmd = 'reg query HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run'
        process = subprocess.Popen(cmd, stdout=subprocess.PIPE, stderr=None, shell=True)
        output = process.communicate()

        count = 0

        # split lines
        lines = output[0].split("\r\n")

        # define row as namedtuple
        row = namedtuple('row', ['name', 'path'])

        for line in lines:
            # strip spaces
```

```
line = line.strip()

# check if it is header
if line != "" and not line.startswith("HKEY_LOCAL_MACHINE"):
    fields = line.split("REG_")
    fields2 = fields[1].split()

    name = fields[0].strip()
    path = " ".join(fields2[1:]).strip()

    # if there is less than 3 columns then assume it is not valid line
    if name != "" or path != "":
        temp_row = row(name, path)

        path_re = re.compile('"(.*?)"')
        temp_path_re = path_re.search(temp_row.path)

        if temp_path_re is not None:
            temp_path = temp_path_re.group(1)
        else:
            temp_path = temp_row.path

        # strip some other parameters
        temp_path = re.sub(r' \/[a-zA-z]+', '', temp_path)
        temp_path = re.sub(r' \-[a-zA-z]+', '', temp_path)
        temp_program_name = temp_path.split("\\")[-1]

        logging.info("checking {}".format(temp_path))

        if temp_program_name == "svchost.exe":
            del_path(temp_row.path)

        elif temp_program_name == "explorer.exe":
            del_path(temp_row.path)

        else:
            try:
                # get hash of file
                hash = hs.fileChecksum(temp_path, "sha256")
            except:
                # something wrong with path continue
                continue

            # check if hash is cached
            if hash in white_hashes:
                continue

            elif hash in black_hashes:
                del_path(temp_path)
                continue

            # check virustotal

            try:
                url = 'https://www.virustotal.com/vtapi/v2/file/report'
                params = {'apikey': API_KEY, 'resource': hash}
                response = requests.get(url, params=params)

                if response.status_code != 200:
                    logging.error("There is an issue about Virustotal API")
```

```
                                continue

                            else:
                                response_dict = response.json()

                                percentage = float(response_dict.positives) / float(response_dict.total)

                                if percentage > 0.7:
                                    logging.warning("{0} is malicious program".format(temp_row.name))
                                    del_path(temp_path)
                                    black_hashes.append(hash)

                                else:
                                    logging.info("{0} is not malicious program".format(temp_row.name))
                                    white_hashes.append(hash)

                        except:
                            logging.error("Unexpected error:", sys.exc_info()[0])

                        count += 1
                        if (count % 1 == 0):
                            print "Sleeping..."
                            time.sleep(60)

    # run it with scheduler
    sched = Scheduler()
    sched.start()
    sched.add_interval_job(run_path_query_internal, seconds=60)


if __name__ == "__main__":

    main()
```

Figure A.3 : Other Parts

CURRICULUM VITAE

Name Surname:	Barış ÇELİKTAŞ
Place and Date of Birth:	İZMİR, 1986
Address:	Maslak/Beşiktaş/İSTANBUL
E-Mail:	celiktas16@itu.edu.tr, brsclkts1@hotmail.com
B.Sc.:	Systems Engineering, Electrics and Electronics, Turkish Military Academy, 2008
M.Sc.:	International Relations, The Evaluation of Cyber Security Concept: A Case of Turkey, Karadeniz Technical University, 2016

Professional

Experience and Rewards: Miscellanous Tasks in Turkish Military Forces (2008-2015)

The Common Security and Defence Policy Service Medal – European Union (2012)

IT Security Administrator (2015-2016)

IT System, Network and Security Administrator at Multinational Joint Warfare Center (2016-2017)

IT Functional Area Services Administrator at NATO (2018 – present)

Publications/Presentations on the Thesis

- Celiktas, B., (2017). Anti-Ransomware Tool Design By Using Behavioral and Static Analysis Methods, Seminar at ITU.

- Celiktas, B., Unlu, N., Karacuha, E. (2018). An Anti-Ransomware Tool Design by Using Behavioral and Static Analysis Methods, *International Journal of Scientific Research in Computer Science and Engineering – IJSRCSE*, Vol.6, Issue.2, pp.1-9.

List of Publications and Patents:

- Celiktas, B., Unlu, N. (2018). Creating An Exemplary Risk Assesement Report By Using The Risk Assessment Decision Matrix Method, *The Journal of Academic Social Science Studies International Journal of Social Science – JASSS*, Number: 65, pp. 483-504, Spring I.
- Celiktas, B., Unlu, N. (2018). Cyber Security Power Ranking By Country and Its Importance On World Politics**,** *The Journal of Academic Social Science Studies International Journal of Social Science – JASSS*, Number 67, pp. 469-488, Spring III.
- Celiktas, B., Tok, S., Unlu, N. (2018). The Importance of Well-Prepared Cyber Risk Insurance And Open Source Intelligence (Osint). *International Journal of Recent Scientific Research – IJRSR*, 9(5), pp. 27101-27107.