**Abstract:**

**Ransomware:**

The concept behind ransomware, a type of malicious software, is straightforward: lock and encrypt data on a victim's computer or device, then demand a ransom to restore access.

In many circumstances, the victim is required to pay the cybercriminal within a certain length of time or risk losing access permanently. Furthermore, because virus attacks are frequently used by cybercriminals, paying the ransom does not guarantee access would be restored.

Ransomware encrypts your personal files, preventing you from accessing your documents, images, and financial information. Those files remain on your machine, but the infection has encrypted it, rendering the data on your computer or mobile device unreadable.

We are using AES encryption for the encryption and decryption.

Malicious link via email to perform the infection.

Ossec for the monitoring and detection purposes.

A python script to restrict the access to the critical directory.

**Introduction:**

There are total five steps to execute the project. Below is the brief description of each step of our project:

**1)Action:** To perform the encryption and decryption action, we used python programming language and written code for encryption and decryption. We used a script written in python which performed AES encryption.

**2) Infection:** We are sending a malicious link to the user which leads victim to download and run an executable that has malicious code.

**3 & 4) Monitoring and Detection:** We are going to use OSSEC HIDS for monitoring purpose. It is a scalable, multi-platform, open source Host-based Intrusion Detection System (HIDS) OSSEC has a powerful correlation and analysis engine, integrating log analysis, file integrity monitoring, Windows registry monitoring, centralized policy enforcement, rootkit detection, real-time alerting and active response. OSSEC can detect and alert when any changes are made to a monitoring folder.

**5) Mitigation:**

We are using a python code which sets only read permissions to the critical folder. So that when a program tries to access the critical folder, the permission is denied and the attack will be unsuccessful**.**

**Related works:**

**The encryption techniques we can use in ransomeware are:**

**Only symmetric encryption ransomware:**

The files can be encrypted quickly using symmetric encryption algorithms like AES. The ransomware will only employ this

encryption method using this strategy. All user files will be encrypted using the AES technique, and the encryption keys

will be stored on disk. Therefore, the decryptor will open this file with the keys and begin decrypting the files when the

infected pays the ransom. The researchers will be able to locate this file using this simple method, and since it isn't

encrypted, they can create a tool to use the keys to decrypt the files.

**Client asymmetric encryption:**

This strategy entails the ransomware creating an RSA key pair, encrypting all files with the public key, and sending the

secret key to the server to be kept. The ransomware needs to transfer the private key to a server, thus both the infected

computer and the server need to be online. This technique of encryption is quite slow; RSA encryption will take a long time

with huge files. There is an issue if either of the two parties is unconnected. The ransomware must either stop running or

it will encrypt all files with the public key and remove the private key without allowing for decryption, or it must temporarily

keep the private key on disk for later decryption.

**Server Asymmetric encryption:**

According to this plan, the server will create a key pair, the ransomware will hardcode the public key, and each file will be

encrypted using the server public key; only the server's private key will allow for file recovery. A logical issue arises when

determining whether the server will provide the client with the private key and enable file decryption. By using this method,

the researchers can obtain the secret key and propagate it among all infected individuals. As a result, with just one ransom

payment, all infections will have their files decrypted. A slower and less practical alternative to transferring huge encrypted

files over the internet is for the infected machine to send all encrypted files to the server for decryption.

**Server and Client asymmetric encryption + symmetric encryption:**

The majority of ransomware in use today uses this hybrid encryption method, which uses both symmetric and asymmetric encryption

and requires an internet connection solely for decryption.

With this plan, the server and the ransomware will each produce a pair of RSA keys. The Client public key is Cpub.key, the

Client private key is Cpriv.key, the Server public key is Spub.key, and the Server private key is Spriv.key. Here's how it will operate:

The ransomware will automatically produce Cpub.key, Cpriv.key, and Spub.key for each infection. Spub.key will also be hardcoded

into the ransomware. It will use the Spub.key to encrypt the Cpriv.key. When the file encryption method is complete, all AES

keys will be encrypted using Cpub.key. Files will then be encrypted with AES.

**Approach:**

**Step-by-step instructions to duplicate the environment:**

We used Ubuntu VM for implementing the ransomware project.

1. **Action:**

To perform the action, we will have to write the encryption and decryption codes.

Please download the encryption.py and decryption.py files on to your computer. As python is used for the encryption and decryption processes, we must run these files.

**As we are using pyAesCrypt, run the below command to install it:**

pip install pyAesCrypt

**Use the below command for encryption:**

Python3 encryption.py

After running the encryption.py you can see that all the files in it and as well in the sub directories gets encrypted as shown in the below image.

**Text

Description automatically generated**

**Use the following command for decryption:**

Python3 decryption.py

By running the decryption.py files you can see that all the encrypted files are decrypted.

**Text

Description automatically generated**

**References:** The encryption and decryption code are referred from the blog of Felipe(Author).

**2)Infection:**

For infection of the ransomware, we are creating a server on discord. Discord is a gaming server on which users can connect to send instant messages and to share game files or executable files among themselves. We are sending email using sendemail command from Kali Linux Operating System. For sending mail we are using an open-source smtp mail server sendinblue in which we are using port:587 for achieving this. We are using the discord link https://discord.gg/kDUaEajf in which we have uploaded the malicious code as an executable file which has been disguised as a game to the victim. Here the victim will think that he is downloading the latest version of pubg. After downloading the game, he will run the executable file to open and install the game. But here’s the surprise! Instead of getting the game, his files will be encrypted by the malicious code.

**Note :** If you get permission denied while executing the file, please change the permission of the exe file which allows it run as a program

**Graphical user interface, text, application

Description automatically generated**

**Graphical user interface, text, application

Description automatically generated**

**3,4) Monitoring & Detection:**

We tried various methods to implement monitoring and Detection. One of them is ossec.

**a)ossec:**

Below are the steps to implement ossec for the monitoring and detection purposes.

apt-get update

apt-get install apache2 -y

apt-get install build-essential -y

apt-get update

wget https://github.com/ossec/ossec-hids/archive/2.9.2.tar.gz

tar -zxvf 2.9.2.tar.gz

cd ossec-hids-2.9.2/

/var/ossec/bin/ossec-control start

cd ..

unzip master.zip

apt-get install unzip -y

mv ossec-wui-master /var/www/html/ossec

cd /var/www/html/ossec

systemctl restart apache2

apt-get update -y

apt-get install php -y

apt-get install build-essential gcc make apache2 libapache2-mod-php7.4 php7.4 php7.4-cli php7.4-common apache2-utils unzip wget sendmail inotify-tools -y

**we also must make the below changes in /var/ossec/etc/ossec.conf:**

<syscheck>

<!-- Frequency that syscheck is executed - default to every 22 hours -->

<frequency>79200</frequency>

<alert\_new\_files>yes</alert\_new\_files>

Added report\_changes="yes" realtime="yes" to all directory tags.(for critical dir as well)

<!-- Directories to check (perform all possible verifications) -->

<directories report\_changes="yes" realtime="yes" check\_all="yes">/etc,/usr/bin,/usr/sbin</directories>

<directories report\_changes="yes" realtime="yes" check\_all="yes">/bin,/sbin</directories>

**Added this rule in local\_rules.xml:**

<rule id="554" level="0">

<category>ossec</category>

<decoded\_as>syscheck\_new\_entry</decoded\_as>

<description>File added to the system.</description>

<group>syscheck,</group>

</rule>

**Note**: Restart the ossec after setup

**After the ossec setup you can use the interface as below:**

Run ifconfig to get the ip address of the system:

Text

Description automatically generated

**Monitoring the folder using the ossec interface:**

Open your firefox browser and type the below in the browser:

<http://10.0.2.15/ossec/>

Replace my IP address with yours so that you will the ossec interface.

Note: There is a certain frequency set for the ossec to scan the files and check the integrity of the files. As this alert pops up only after the scan , it may take sometime after running the encryption to display the alert. This is the default feature of ossec.

Then you will find an alert which says the integrity checksum is changed as below:

Graphical user interface, application

Description automatically generated

1. **Inotify:**

We can also use inotify in order to monitor this directory. Below is the command to monitor the critical directory:

inotifywait -m -r /home/sec-lab/critical

A screenshot of a computer

Description automatically generated

Graphical user interface, text

Description automatically generated

1. **Watchdog:**

We tried to implement watch dog, but if we use watchdog it doesn’t explain what action is exactly being performed on the files which we are monitoring.

**5)Mitigation:**

We are implementing a python script which modifies the file permissions inorder for stopping the encryption of files. After running the program, we can see that the encryption of files do not happen which can be used as a mitigation strategy. After restoring the file permissions we can see that the encryption can again be performed which means that our strategy works.

You can see in the below screenshot that ./pubg is not executed and files are not encrypted:

Text

Description automatically generated

**Results:**

1. **Result of action step:**

**Successfully encrypted by running encryption script:**

**Text

Description automatically generated**

**Successfully decrypted by running decryption script:**

**Text

Description automatically generated**

1. **Result of infection step:**

**Executable file downloaded successfully from the server link provided as game called pubg:**

**Graphical user interface, text, application

Description automatically generated**

**Executing the exe file thinking it as an game executable successfully encrypted the crtitcal folder:**

**Text

Description automatically generated**

**c)Result of Monitoring and Detection:**

**using ossec**

Graphical user interface, application

Description automatically generated

**Using inotify:**

Graphical user interface, text

Description automatically generated

**d)Result of Mitigation:**

**Text

Description automatically generated**

**References:**

* Felipe “Recursive File Encryption in Python 3”.CodeOnBy In Between Bytes. https://codeonby.com/2019/12/18/recursive-file-encryption-in-python-3/(accessed Nov. 30, 2022).
* MadhuSudhan “Ownership of Files”. Geeksforgeeks. https://www.geeksforgeeks.org/chown-command-in-linux-with-examples/