Project Group - 8

Detecting Asymmetric Encryption based Ransomware Attack

Hema Srinadh Koganti
Department of Conmputer Science and Engineering
University of North Texas
Denton, TX, USA
hemasrinadhkoganti@my.unt.edu

Santhoshi Kareddy
Department of Computer Science and Engineering
University of North Texas,
Denton, TX, USA
santhoshikareddyviii@my.unt.edu

Chandana Chamala
Department of Computer Science and Engineering
University of North Texas,
Denton, TX, USA
chandanachamala@my.unt.edu

Anupallavi Balaboina
Department of Computer Science and Engineering
University of North Texas,
Denton, TX, USA
anupallavibalaboina@my.unt.edu

Sahasra Sai Tarun Mandiga
Department of Computer Science and Engineering
University of North Texas,
Denton, TX, USA
sahasrasaitarunmandiga@my.unt.edu.

Abstract— **In this digital world, interaction with internet is crucial, but this comes with a risk of cyber-attacks, to get safeguard from these attacks we need to develop robust Intrusion detection system. For this project we have developed an asymmetric encryption-based ransomware attack and Intrusion Detection System to detect the ransomware attack. This project proposes an innovative solution to counter ransomware attacks. We have developed an IDS alerting system, if the system observes any modifications in the directory, it will alert the user and it will ask user to authorize the process, if user refuse to authorize the process it will kill the process. By killing unauthorized process, we can stop programs encrypting the files.**

Keywords— Ransomware Attack, Asymmetric encryption, Intrusion Detection System.

I. INTRODUCTION

Cybersecurity is crucial for organizations due to increasing data breaches, ransomware attacks and other malicious activities. Traditional security measures often failed to detect the attacks in early stages, more robust approaches and measures should be developed.

Securing sensitive data has become a crucial task in the digital age. The growing frequency of cyberattacks demands the robust and innovative solutions to strengthen data security and integrity. An integrated proactive intrusion detection system (IDS) is introduced as part of this project to fulfill this urgent demand. This system will offer a dynamic barrier against unwanted file modifications, guaranteeing that files in a specified directory are well-protected.

Intrusion detection system will act as safeguard to the directory, continuously monitoring the directory to counter unauthorized process to modify the file in the directory. This system not only maintain integrity of data, also provide timely alerts enabling proactive response to mitigate attack.

II. RELATED WORK

Ransomware attack implantation and analysis:

Ransomware attacks have been extensively researched. Work such as "RaaS: Demystifying the Business Model of Ransomware-as-a-Service" aids in our comprehension of the operation of ransomware and the methods used by attackers. The other one, "Anatomy of a Ransomware Attack: A Case Study of REvil," delves into actual ransomware assaults and demonstrates the instruments and techniques that attackers employ.

IDS for Ransomware defense:

Depending on the internet is great in the modern digital age, but it also exposes us to possible cyberattacks. We're concentrating on building a robust system called an Intrusion Detection System (IDS) to thwart ransomware assaults to combat this. With the use of asymmetric encryption and a unique technology that recognizes ransomware threats before they cause harm, this initiative offers a novel approach to combating ransomware.

Ransomware detection methods in IDS:

Intrusion Detection Systems are receiving awareness from those looking to protect themselves from ransomware. Research papers such as "A Survey on Ransomware Detection Techniques" and "Machine Learning for Ransomware Detection: A Review" discuss several approaches to ransomware detection, such as anomalous behavior detection or the use of specialized algorithms.

User Focused intrusion detection:

In this study, they are developing an intrusion detection system (IDS) that monitors directories and, in the event of any suspicious changes, requests user permission before taking any action. System intervenes and ends the procedure if something is wrong, and the user doesn't consent. Want to prevent ransomware from encrypting files before it's too late.

III. Implementation

In this project, we tried to demonstrate how Intrusion Detection system works, we developed two modules for this project: encryption module and detection module.

System – Mac book air.
Operating system – Ubuntu (Linux).
Programming Language – python.
Target – Directory.

Modules:

1. Encryption module acts as attack on target, tries to encrypt all the files in target directory.

   Libraries – pyCrypto.

2. Detection module acts as safeguard to the target directory, it would monitor file activities in directory, if any file modified by process, detection system will gather process details from auditd logs, and it will send alert to user to authorize the modification. If user deny the activity, Detection system will kill the process immediately to mitigate the attack.

   Libraries – auditd, watchdog, tkinter.

Libraries:

1. **PyCrypto**: It is a popular python library, offers different set of encryption options like advanced AES and RSA algorithms for developers.
2. **Auditd**: It is a powerful tool for auditing and monitoring system events on Linux system. It records and captures security-related actions, like file access and user authentication, allowing administrators to examine and evaluate logs for incident investigation and security compliance. This improves system security and accountability in general.
3. **Watchdog**: It is a Python package makes file and directory monitoring easier. It is a flexible tool that can be used to create programs that are responsive and dynamically respond to filesystem events, increasing efficiency in system monitoring.

A. Idea

Main idea of the project is to demonstrate working model of IDS system.
Initially, anomaly file tries to encrypt files in directory using public key, and it will write encrypted data to file. In backend auditd will be recorded and stored all the logs in aduitd.logs file. When proactive IDS system detect modification in target directory, then it will fetch process details which is trying to modify files in target director from auditd.logs files. It will check if the process is authorized or not, if not, it alerts the user and it will kill the process immediately to mitigate the attack.

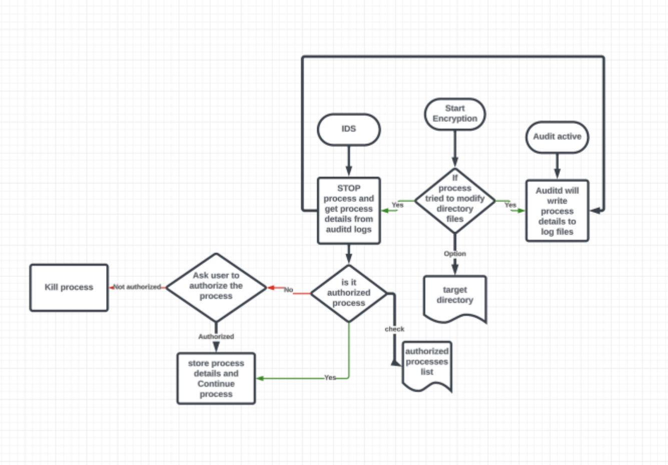

IDS
Start Encryption
Audit active
STOP process and get process details from auditd logs
If process tried to modify directory files
Yes
Yes
Auditd will write process details to log files
Option
target directory
Kill process
Not authorized
Ask user to authorize the process
No
is it authorized process
check
authorized processes list
Authorized
store process details and Continue process
Yes

Figure 1: flow chart of the project

Pseudo code for detection system:

This code will give general idea on IDS logic implementation.

```
monitoring system (watch dog and auditd) will be active
in the victim system.
{
    if file modified in target directory.
    {
        auditd record logs to auditd.log file.
        watchdog will get pid and pname from
        auditd.log.
        watchdog will pause the process.
        watchdog will check if the process is previously
        authorized or not.
        {
            if previously authorized.
            {
                continue process.
            }
            if not authorized.
            {
                Notify the user in popup window about
                the process and ask to authorize it.
                if user authorize the process
                {
                     store the process details for future
                check.
                     continue process.
                }

                if user don't authorize the process
                {
                     kill the process immediately.
                }

            }

        }

    }

}
```

B. Attack

To attack the victim, we choose phishing email technique, First, we will create an email which looks like a company offering food coupon code for a free lunch. we will send phishing email to victim with anomaly file attachment. Attachment contains two files; one is public key file and other one is python program file to encrypt files in target directory.

How encryption works?
We used both AES (symmetric) and RSA (asymmetric) algorithms to use advantages of both algorithms. Firstly, we will generate public and private keys using RSA algorithm. While encrypting target file, generate random session key and encrypt it with public key using RSA algorithm. Then, encrypt file data with encrypted session key using AES algorithm in EAX mode.
EAX mode in AES algorithm will provide authentication and encryption in single step.

When user clicks the link in email, attachments will be downloaded, and encryption file will start encrypting files in target directory.

C. Monitoring

For monitor propose, we used auditd linux library. Firstly, auditd is installed in victim system, configured rule for auditd to define what file events should to recorded in log file.

```
Rule:   -w /path/to/directory/ -p w -k target_dir
```

-w /path/to/directory/ -> this "-w" flag indicates watch the mentioned directory.
-p w -> this is a filter to auditd trigger, auditd will trigger when write operation is performed on file.

-k target_dir -> "-k" is a key flag. This string will be associated to rule event records. It will help in finding the event records recorded by rule in log file.

From this log records we can get who modified the file, file accessed process details, timestamps when file modified etc.,

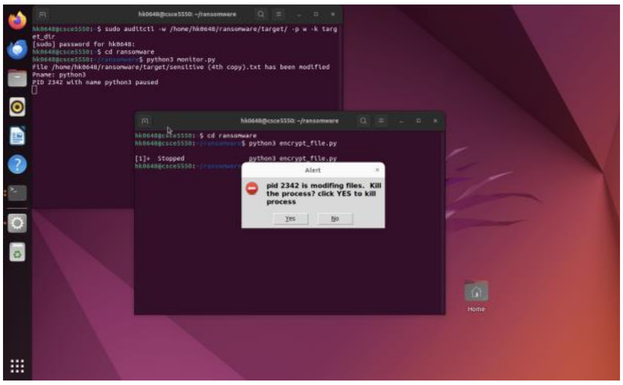
D. Detection

For Detection, we used watchdog python library, it is a python program, it will dynamically respond when file is modified in target directory.

When a file is modified in directory, it will get the process details of which is responsible for the modification. And it will check if this process is previously authorized or not from the file (authorized_process.txt).

If file doesn't contain process id, watchdog will send an alert popup to user to notify about file modification and process details like process id and process name. And it asks user to authorize this file modification.

If user authorize the action, it will store process details to authorized_process.txt file for future reference. Else, watchdog will kill the process immediately.



```
hk0648@csce5550:~$ sudo auditctl -w /home/hk0648/ransomware/target/ -p w -k targ
et_dir
[sudo] password for hk0648:
hk0648@csce5550:~$ cd ransomware
hk0648@csce5550:~/ransomware$ python3 monitor.py
File /home/hk0648/ransomware/target/sensitive (4th copy).txt has been modified
Pname: python3
PID 2342 with name python3 paused
```

```
hk0648@csce5550:~$ cd ransomware
hk0648@csce5550:~/ransomware$ python3 encrypt_file.py

[1]+  Stopped                 python3 encrypt_file.py
hk0648@csce5550:~/ransomware$
```

Alert

pid 2342 is modifing files. Kill the process? click YES to kill process

Figure 2: alert message shown by watchdog.

Pseudo code for detection:
This code will help in understanding how detection logic implemented.

```
When file_modified
{

        Get process_details from auditd_logs
        Stop process
        Check process_id in authorized_process.txt:
           {
                If pid_present in file:
                    {
                       Continue process
                    }
                  Else
                    {
                         Ask user_to_authorize:
                           {
                                  If authorized_by_user:
                                        {
                                               Store process_details_to_file
                                               Continue process
                                           }
                                   Else:
                                           {
                                           Kill process
                                            }
                             }
                       }
           }
}
```

E. Mitigation

To mitigate the attack, we follow some standard mitigation procedures:

- Disconnect the system from network.
- Alert if any other users connected to the same network.
- Restore effected files from backup.
- Update the system to latest version.

For our project we can decrypt all encrypted files using private key.

IV. Experiments

To develop this project, we tried many options like:

1. System is very much accurate because it will ask to approve the process when the attack modify first file. Every time the first file will get effected.
2. Implementing couple of techniques like symmetric and asymmetric to implement encryption logic.
3. Storing the authorized process details in different location to avoid alerting user every time file modified by approved process.
4. Recording only certain file events to log file to avoid disk usage.
5. Using data extraction techniques to extract only required process details like process id and process name from a big log file.

V. Discussion

We have observed few issues like:

- System crash while implementing detection module.
- While data extracting from log files, but we designed logic to get 100% accurate process details.
- There will be some cases where watchdog can’t extract process details from auditd logs for all the file modifications events.

VI. Conclusion

We were successfully able to demonstrate our encryption module and proactive attack detection system module on Linux system.

VII. Reference

I. https://medium.com/@Bhupi2508/understanding-symmetric-and-asymmetric-aes-algorithms-for-data-encryption-d39a14e6dd91

II. https://pypi.org/project/watchdog/

III. https://pypi.org/project/pycrypto/

IV. https://izyknows.medium.com/linux-auditd-for-threat-detection-d06c8b941505

V. https://ejournal.uin-suka.ac.id/saintek/ijid/article/view/09101

VI. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9360899

VII. https://doi.org/10.1016/j.cose.2021.102469