

Malware Developing Guide: Encryption and Decryption

Jutiphat Chatsomsanga, Chawalit Benjangkprasert

Department of Computer Engineering, School of Engineering
King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand.
62601049@kmitl.ac.th, chawalit.be@kmitl.ac.th

Abstract— This research aims to understand malware from malware author's view. Starting from Ransomware, we attempt to develop ransomware written in python and encrypt all files in Windows platform without doing harm to crucial system files. In this research we use Asus K550J laptop with Windows 10 education (Build 19042) as a victim machine. We ran encryption and decryption program with different privilege and observe the results as well as time takes to encrypt and decrypt files. Results show that encryption process is much slower than decryption process and both encryption and decryption work slower in the bigger file size.

Keywords— Malware, Ransomware, Python, Encryption, Windows

I. INTRODUCTION

The Advanced Encryption Standard (AES) features and implementation are thoroughly explained by A. M. Abdullah in [1]. Able to encrypt and decrypt faster than its older brothers: data encryption standard (DES) and triple DES (3DES). Till this day, there is no evidence to support that AES can be broken. Such encryption is also used by ransomware to encrypt its victim's data.

Ransomware has become a popular form of attack in recent years growing 350% in 2018. The estimated global damage from ransomware in 2018-2019 is about \$19.5 billion [2]. Not only victims will need to pay the ransom, but also the downtime cost of its service caused by ransomware was taken into account. Creators of ransomware use various methods to spread and these cyber-criminals are keen to develop new methods to spread it [3]. Research suggests that to prevent from ransomware, training company's staff for cybersecurity awareness is recommended [4].

Ransomware is so popular that there is even a Ransomware as a service (RaaS). Ransomware developers sell access to distributors (customers) through a partnership program. The program is operated under a financial model that splits profit per infection between the developers and distributors (e.g., 60/40 split). The most used technique by ransomware to spread in 2019 was via file download/email with malicious link. It is 29% of all method used to spread ransomware from 2,538 cases worldwide [5].

There are two main types of ransomware locker ransomware and crypto ransomware. The former will lock the

victim out of their infected devices, preventing them from using it but will not encrypt files inside then tell victims to pay ransom to unlock their device. The latter, however, will encrypt files in infected devices mostly they encrypt files based on important or common file type such as databases file, pictures files and document files. Research shows that locker ransomware is more common than crypto ransomware [6].

Cryptographic algorithm is designed to protect the data transmitted between two party: sender and receiver but to do so the algorithm must be decryptable. Ransomware takes this technology to its malicious purpose by held your file encrypted as a hostage, keep the key to its distributors and make you pay ransom. This proves that in the wrong hands, technology can be dangerous.

Although is dangerous, it can be detected by many ways [7] and it has a huge weakness which is if ransomware is stopped in the middle of their action the damage done to the infected systems will stop as well.

This paper focuses on encryption part of ransomware. Tests the encryption and decryption on Windows operating system, Windows10 education then describe its efficiency.

II. RELATED WORK

Research in [8] describes efficiency analysis of 4 encryption algorithm which are Blowfish, DES, 3-DES, and AES against different file type which are videos, text, and picture and suggest that blowfish algorithm is the most suitable encryption algorithm for videos and text files. But AES is the most suitable encryption algorithm for picture files. DES and 3-DES algorithm is susceptible for brute force attack and meet in the middle attack respectively.

Ransomware can be detected by various machine learning technique as suggests in [9] each technique has different accuracy the most accurate is random forest algorithm it achieved 82.80% accuracy against 10 malware families in a total of 101 ransomware.

A. H. Mohammad [10] describes behavior of various infamous ransomware once it ran on victim's system and suggests that ransomware can be detected by monitoring system files and registry activities.

S. Sechel [11] shows the difficulty of antiviruses to detect evasive ransomware. The common evasion method used by ransomware authors involves use of packers and crypters.

Packer uses compression to obfuscate the content of ransomware while crypter uses encryption to obfuscate content of ransomware. Packed malicious programs caused difficulties for signature and heuristic based malware detection algorithm to detect malicious program.

J. Kaur, et al. [12] analyzed ransomware in the following families Sage Ransomware, WannaCryptor, Lambda-Locker, Hydra-Crypt, CryptoWall, and SamSam. They monitored ransomsware interaction with system files, registry values and DLLs called by ransomware. They found that ransomware create persistence keys on the victim's system to bypass a reboot. Ransomware deletes shadow copies to prevent user to restore their file using different command to achieve the same goal and leverage DLLs to promote code reuse and efficient memory usage. Ransomware will deletes itself after they finished working using batch script.

III. EXPERIMENT METHOD

In this research we attempt to create ransomware from scratch. The scope here is only encrypt and decrypt part. The aim is that to know that how to encrypt/decrypt files and which files should be encrypted that will not break the OS. The programming language using in this experiment is python with cryptographic library [10] as an encrypter/decrypter. It will encrypt files on target computer with administrator privilege by clicking run as administrator. It has separate files for encryption program and decryption program. The Windows security and acronis active protection was not disable during encryption. Windows security did not detect encryption program as malicious program but somehow has detected decryption program as malicious, therefore in the decryption phrase Windows security and Acronis Active Protection need to be turned off or allowed the decryption program to be on the machine by user.

A. Encryption and decryption

Encryption and decryption are the main functions of ransomware. Ransomware will encrypt its victim's files. The algorithm used in this research to make a cypher is AES-256. AES-256 is virtually impenetrable using only brute force method but it is not the fastest algorithm to encrypt data [11]. When the encryption program is run, first, it will create a keyfile.txt contains a 32-bytes key, since AES algorithm is symmetric. Encryption and decryption key are the same. Later when decryption is needed, the decryption program will read key from the same keyfile.txt. This means that the keyfile.txt should be omitted during encryption, otherwise the original will not be able to restore by decryption program again because the key would not match. In this case keyfile.txt is stored in the victim machine to reduce complexity of the program. In real world scenario, ransomware would send its key to attacker somehow and will not store the key within the victim's machine. The encryption program then traveling through file path using os.walk() method. Next, it will read each file's data as byte then Encrypt the data and try to write encrypted data and append file with ".weep". This program will check first if the file has write permission or not to

prevent the unnecessary encryption. In the case that write permission is denied. It will log that full file path to "write denied.txt". Decryption program however, works differently it will decrypt based on file extension ".weep" only. Therefore, there will be only the file that has been modified (encrypted) to decrypt only. This makes decryption faster than encryption.

Encryption will expand original data by 33% because Base-64 encoding scheme is used in this cryptography library [12]. Base-64 encoding can only represents 64 (6 bits) per byte different value per byte whereas in normal, 1 bytes can represents up to 256 (8 bits) values therefore, every 3 bytes of information Base-64 will need 4 bytes to represents it. As show in Figure 1 and Figure 2.

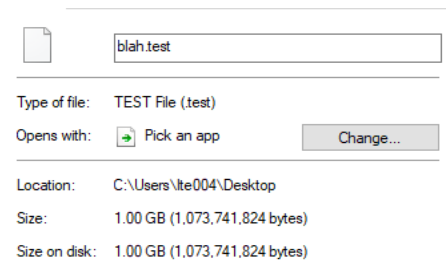


Figure 1. 1 GB file before encryption

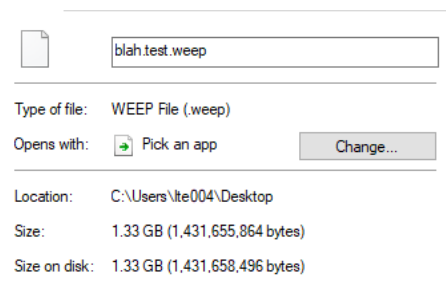


Figure 2. 1 GB file expand to 1.33 GB after encryption

Decryption will restore original (1 GB) data even if it was expanded during encryption process.

B. Pyinstaller

Pyinstaller [13] is a tool to make .exe files from python language. It provides an option to bundle all dependencies of any python program into one .exe file because Windows did not come with python installed. Anyway, Pyinstaller still left a footprint to be detected but is not considered as a malicious file on VirusTotal because some benign program written in python, if want to be portable there is a chance that author used Pyinstaller to generate .exe from those programs.

C. Target computer.

1) Hardware: The factory default Asus K550J with CPU: Intel core I7-4710HQ, up to 3.5 GHz, 4 GB ram and WDC-WDS240G2G0A-00JH30 hard drive as a Victim computer. This computer is copied from computer engineering's student computer. There is an actual software, documents (laboratory result reports, source code) for most realistic victim computer environments.

2) Operating System: Windows 10 Education (Build 19042) 64-bit.

3) Antivirus: Acronis active protection detected decryption program as “possible ransomware” but did not detect encryption as a harmful. Before when tests against windows 10 enterprise edition this same thing has happened and it needs further investigation of the cause.

D. Program structure

For the encryption program, we want to know which drive is a primary drive. we used subprocess module in python to run a windows command line: “echo %SystemRoot%” to find which drive windows system files reside then we will omit the Windows directory to reduce the chances that the program will break windows system file. After that if there is other drive, this program will try to encrypt every file in every drive.

The emergency break is the part of this program. It is developed because while developing encryption/decryption program. It is possible to accidentally encrypt important works on malware developing machine. This function will compare static MD5 hash of my computer “hostname” if the hash of computer’s “hostname” matched the program will exit before executing anything else.

Finally, it will return to “SystemRoot” drive and encrypt every file except those in Windows directory. In Windows OS, name of the drives can only be one ‘A-Z’ letter only. This program generated a ‘A-Z’ list using loop. It did not store any static ‘A-Z’ character in this program hoping to avoid detection by reverse engineering this encryption program.

IV. RESULTS OF THE EXPERIMENT

After encrypting all files that is smaller than 1 GB (1073741824 bytes). Except for Windows directory in the main drive and all files in other drives.

The reason for not encrypting file that is bigger than 1 GB is bigger files use more time to encrypt. We want to encrypt most files as possible in least time as possible.

Tables 1-4 show the results of encryption and decryption speed based on the performance of the computer stated as section III (c).

TABLE 1. ENCRYPTION AND DECRYPTION SPEED

Process	Processed file (files)	Total file size (GB)	Total time used (s)	Speed (MB/s)
Encryption	549,416	92.08	32,820	2.873
Decryption	524,659	109.43	4560	24.57
File count difference	24,757	-	-	-

TABLE 2. ENCRYPTION AND DECRYPTION SPEED WITH 1 FILE

Process	Processed file (files)	Total file size (GB)	Total time used (s)	Speed (MB/s)
Encryption	1	1	129	7.937
Decryption	1	1	35	29.257

TABLE 3. ENCRYPTION SPEED OF ALL-ZEROS FILE (TEST FILES)

Total file size (MB)	Total time used (s)	Speed (MB/s)
102.4	5.996	17.078
192.4	12.049	15.968
307.2	17.439	17.616
409.6	23.998	17.068
512	32	16
614.4	50.957	12.057
716.8	66.5237	10.775
819.2	83.4908	9.811
921.6	104.703	8.802
1024	129	7.937

TABLE 4. DECRYPTION SPEED OF ALL-ZEROS FILE (TEST FILES)

Total file size (MB)	Total time used (s)	Speed (MB/s)
102.4	3.59	28.524
192.4	6.98	25.419
307.2	10.899	28.186
409.6	15.049	27.218
512	19.129	26.766
614.4	26.0483	23.587
716.8	31.168	23
819.2	34.898	23.474
921.6	44.487	20.716
1024	53.107	19.282

The results show decryption process is faster than encryption process. As shown in Table 1 the number of encrypted file is more than decrypted files but total size of encrypted file is smaller than decrypted file because data in the files is expanded during encryption. There are 24,757 (4.56%) files that were encrypted but never decrypted. This is not a big problem because the “.weep” file is still there they can be decrypted anytime since we have a key and decryption program but need further investigation for why they were not decrypted.

However, the true encryption and decryption rate of this encryption program and decryption program against all-zeros files generated from “fsutil” command in Windows. The speed is measured by encrypt and decrypt 1 file of 1 GB size which achieved a speed of 7.937 MB/s and 29.257 MB/s respectively, as shown in Table 2.

In Table 3 and Table 4 show encryption speed and decryption speed degradation over a bigger file size. We can see that encryption speed degraded faster than decryption speed for a bigger file size.

Finally, encryption and decryption program was detected as malicious by 2 out of 72 antivirus engines from VirusTotal which are SecureAge APEX and Cynet. Windows security and acronis active protection, however, has detected our

decryption program as malicious and try to erase it from the drive.

V. CONCLUSIONS AND FUTURE WORK

This encryption program was not detected by most antivirus but it is slow. Encrypting all files in the computer at administrator and user privilege does not damage crucial Windows file but for ransomware, it should encrypt only important file extensions or encrypt based on user's directory for more speed.

In the future, we would like to improve encryption speed. By creating custom library for encryption/decryption and try to use other encryption algorithm.

REFERENCES

- [1] A. M. Abdullah, "Advanced encryption standard (AES) algorithm to encrypt and decrypt data", Ph.D. thesis, Eastern Mediterranean University-Cyprus, 2017.
- [2] (2020) the purplesec website. [online]. Available: <https://purplesec.us/resources/cyber-security-statistics/ransomware>.
- [3] S. Aurangzeb, M. Aleem, M. Iqbal, and A. Islam, "Ransomware: A survey and trends", *Journal of Information Assurance and Security*, vol. 12, pp. 48-58, 2017.
- [4] M. Kiru, and J. Aman, "The age of ransomware: Understanding ransomware and its countermeasures", *Artificial Intelligence and Security Challenges in Emerging Networks*, first Edition: Publisher: IGI Global, 2019.
- [5] (2020) the crowdstrike website. [online]. Available: <https://go.crowdstrike.com/rs/281-OBQ-266/images/Report2020CrowdStrikeGlobalThreatReport.pdf>
- [6] C. Simoiu, J. Bonneau, C. Gates, and S. Goel, "A study of ransomware", in *Proc. the Fifteenth Symposium on Usable Privacy and Security*, pp. 155-174, Aug. 2019.
- [7] D. Gonzalez, and T. Hayajneh, "Detection and prevention of crypto-ransomware", in *Proc. IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference*, pp. 472-478, 2017.
- [8] G. S. Vennela, N. V. Varun, N. Neelima, L. S. Priya and J. Yeswanth, "Performance analysis of cryptographic algorithms for cloud security", in *Proc. Second International Conference on Inventive Communication and Computational Technologies*, pp. 273-279, 2018.
- [9] F. Noorbehbahani, F. Rasouli and M. Saberi, "Analysis of machine learning techniques for ransomware detection", in *Proc. 16th International ISC (Iranian Society of Cryptology) Conference on Information Security and Cryptology*, pp. 128-133, 2019.
- [10] A. H. Mohammad, "Analysis of ransomware on windows platform", *International Journal of Computer Science and Network Security*, vol. 20, no. 6, pp. 21-27, 2020.
- [11] S. Sechel, "A comparative assessment of obfuscated ransomware detection methods", *Informatica Economica*, vol. 23, no. 2, pp. 45-62, 2019.
- [12] J. Kaur, F. Jaafar, P. Zavorsky, "An empirical analysis of crypto-ransomware behavior", in *Proc. International Conference on Systems*, pp. 1-6, 2018.
- [13] (2020) the github website. [online]. Available: <https://github.com/pyca/cryptography>
- [14] M. Panda, "Performance analysis of encryption algorithms for security," in *Proc. International Conference on Signal Processing, Communication, Power and Embedded System*, pp. 278-284, 2016.
- [15] I. Sumartono, A. Siahaan, P. U. Andysah, and A. Arpan, "Base64 character encoding and decoding modeling", *International Journal of Recent Trends in Engineering & Research*, vol. 2, no. 12, pp. 63-68, Dec. 2016.
- [16] (2020) the pyinstaller website. [online]. Available: <https://www.pyinstaller.org/>



Jutiphat Chatsomsanga received his B.Eng. in information engineering from King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand in 1990. He is now studying master's degree in information engineering. His current research interests are in cryptography, information security and malware.



Chawalit BENJANGKAPRASERT received his M.Eng. degree from King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand in 1990. In 2006, he received Doctor degree in engineering from Tokai University, Japan. He is currently an associate professor in Department of Computer Engineering, School of Engineering, King Mongkut's Institute of Technology Ladkrabang. His current research interests are in signal processing, wireless communication system, antenna design and machine learning.