The Ransomware Report*

Anonymous
Removed

ABSTRACT

TBD

KEYWORDS

Malware, Ransomware, Cryptography

ACM Reference format:
Anonymous. 2020. The Ransomware Report. In *Proceedings of NA, NA, 2020,* 3 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

As technology continues to become more integral to the day-to-day operations of municipalities and businesses across the world, it becomes increasingly important to analyze the inherent risks that come with conducting business and storing important files in a digital world. One considerable threat to almost all organizations is ransomware: a type of malware that renders users' files inaccessible by encrypting them with a key known only to the attacker. In most cases, a ransom in the form of cryptocurrency is demanded by the malicious actor - hence the term *ransomware* [1]. This means the affected user may permanently lose access to important documents and data unless a large sum of money is paid or some other condition set by the malicious party is met.

In the last quarter of 2019, the average payment requested to release files from ransomware was $84,116, but that number has been steadily increasing and culminated in a staggering average of $190,946 in December of 2019. Over 200,000 organizations reported a ransomware infection in 2019, and experts project that this number will only increase in the coming years [7].

In one high profile incident, the city of New Orleans was forced to declare a state of emergency in response to a December 2019 ransomware attack that crippled many of the city's computer systems and left citizens unable to access basic services like online payment systems and district court appointment scheduling [8]. New Orleans is just one major example in a startling global trend; ransomware has also plagued large schools, businesses, and even hospitals across the world. The notable "WannaCry" ransomware affected businesses and users in over 150 countries and cost an estimated $4 billion in financial losses [5]. The consequences of being infected with this malware are dire, and researchers and cybersecurity experts have yet to propose a general solution to effectively combat ransomware [4]. While efforts like "No More Ransom!", a joint project between the European Cybercrime Center and cybersecurity companies like McAfee and Kaspersky, have successfully created decryption tools for many varieties of ransomware, many varieties exist without a concrete and consistent solution available.

The remainder of this paper takes into consideration the practical approaches that will be taken with testing ransomware in a controlled environment. Section 2 details specifically what design elements were crucial to the development of this malware, and what other options were available in order to simulate an attack, but were ultimately either flawed or were outside the scope of this paper. Section 3 discusses the structure and format of the project, including the considerations that were chosen to move forward with including operating system and language choice for the ransomware scripts. In detail, Section 4 gives a step-by-step account of the process in executing this artificial malware, including how the payload containing the code is created, delivered, and what sort of files are taken over when the code is executed. After considering what the ransomware does, Section 5 details the reflection process and what worked and did not work according to the original design plan, as well as what techniques helped to achieve the desired outcome for the project. Section 6 discusses the ethical and legal consequences of releasing this malware out to the public without explicit and written consent outside of a test environment, and what it could mean for the future of cybersecurity. Lastly, 7 discusses where the project currently stands in comparison to the project plan, and what work could be done to improve the project in the future.

2 DESIGN CONSIDERATIONS

The purpose of this project is to recreate what happens in a real-life ransomware attack by infecting a dummy computer that will act as a "target" machine. In order to pull off a ransomware attack of this scope, it is imperative that the intended target of the malware give explicit consent for the malware to run, since this is an accurate account of how these attacks work in practicality. Hence, the malware is designed with intentions to deliberately deceive the user with messages that are intended to act as an innocent permissions request in order to enable macros, which will aid in the delivery of the malicious script.

In addition, another design aspect that needed to be considered was the type of operating system that this target would have. Since the Windows operating system is one of the most commonly used operating systems for commercial and home use, it would be important to design the malware around the Windows file systems for the purpose of the encryption portion of the application. For reference, the "WannaCry" ransomware abuses the same tactics of rewriting the Windows file system by modifying its permissions to allow the malware to read/write user data, and from there implement its encryption attack.

For the encryption, it was decided that the best scheme to use for this malware would be the Advanced Encryption Standard (AES)

*Produces the permission block, and copyright information of a 128-bit key length, since it has a very low chance of being cracked without utilizing extensive resources and is relatively easy to implement. There was at one point a consideration for the RSA encryption scheme, but it was determined that the encryption run time for the malware would be too extensive, especially if it is executed on a system with a vast quantity of user data.

The last important design aspect that was taken into consideration was what action the user should perform in order for the malware to go through with the process of decrypting the now-encrypted files after being run. At first it was considered to have the malware ask for a ransom in Bitcoin to decrypt, much like the WannaCry or Bad Rabbit ransomware, however this was deemed lacking in terms of originality and creativity. Instead, the choice was to implement some sort of user interactivity for the sake of this project being simply a demonstration. The two ideas for a decryption activity were two equally-challenging games: Frets on Fire (a Guitar-Hero alternative for the PC), or the popular game Snake. Due to the possibility of needing to acquire certain licenses and permissions, and the limited time frame for the project, this project will be using the latter for the decryption function of the malware. For the entire piece of malware, it was decided to have a Python code base, as it is exceptionally versatile as a scripting language and its multiple libraries provide for a larger level of creativity in implementation.

3 ARCHITECTURE

The particular technologies used to develop this malware were chosen to reflect the current state of personal computer (desktops, laptops, etc.) use by businesses, schools, and other large organizations in 2020. The target machine reflects an ordinary and freshly imaged enterprise copy of Windows 10, which is the predominant non-mobile operating system with over 80% of the market share [6]. The languages and patterns chosen to develop the ransomware with were modelled off real-world examples of ransomware that have been released under an open source license. The language used to develop the ransomware is Python 3, but because standard Windows installations do not come with a Python interpreter pre-installed, the code needs to be transformed to actually run on the target machine. Before distribution, it is converted to a binary executable file using PyInstaller that contains all necessary dependencies and a portable Python interpreter so that the executable is completely independent and able to be run.

The ransomware in this paper relies on some social engineering to convince the user to make sub-optimal decisions about their security. While this may seem unreasonable, macro-based attacks and ransomware have both been very successful since their inception and continue to be prominent threats to users [2]. Ransomware can also be delivered effectively through vectors like undisclosed "drive-by" downloads served by a compromised website or by methods as simple as distributing infected USB drives to potential victims.

4 IMPLEMENTATION

This section provides an in-depth look at the execution process and control flow of the developed ransomware from start to finish. First, the subject on the target machine will proceed to access a phishing email that contains an innocuous-seeming document stored in a common file format like .docx or .pdf. This file will seek to conduct a macro-based attack using the Microsoft Office suite's Visual Basic for Applications built-in interpreter to download and execute this paper's proposed Python ransomware code. This is a common vector for distributing malware and relies on social engineering to trick a user into downloading a file and enabling macro support in their office software [2, 3]. This file will contain "help" text describing why it is be beneficial to enable macros to trick the user into compromising their system. If successfully enabled, the macro will use the `UrlDownloadToFile()` system function to download the malware executable using an obfuscated source URL string. Once downloaded, the combined ransomware code and Python interpreter executable will be run and the malware will begin to attack the user's system. From there, the core script will search for any files that may indicate user data by the common file extensions contained in a dictionary (i.e. .gif, .png, .docx, .zip, etc.) and encrypt them using the Advanced Encryption Standard (AES) with a key length of 128 bits, and an indication will display informing the user that their access to their files has been blocked. Lastly, in lieu of digital currency to preserve the integrity of the project, the user will need to participate in an open-source licensed version of the classic Snake game written in Python to unlock their files.

Once deployed and running on a test virtual machine through RIT's RLES service, the malware will be analyzed using Ghidra, an open-source reverse engineering tool created by the NSA, in an attempt to see what processes on the Windows target machine are running when the malware is executed, and what resources are being utilized at run-time. Based on what is found from the preliminary Ghidra analysis, the encryption scheme can possibly be deduced and an attempt to decrypt the key to the test files can be made based on the AES scheme. Finding success in decrypting the user's files this way is highly improbable, but it is still a theoretically-viable recovery option if all other options fail. To move towards a more comprehensive solution to ransomware attacks, this paper also focuses on exploiting issues within common ransomware implementations as well as leveraging existing solutions like file recovery to mitigate the damage the malware can cause. By analyzing the memory layout of the malware process at runtime, it is possible to determine where the decryption key is being created and stored on the system or what its value was before it was communicated to an external server. An anti-malware tool created in Python will be developed to attempt to crack this key in a non-intrusive approach, otherwise it will shut down the processes that are responsible for the encryption scheme, and attempt to recover previous versions of the dummy files prior to the malicious executable being activated.

5 LESSONS LEARNED

5.1 Response to Investigating Teams

TBD

5.2 Other Lessons

TBD

6 ETHICAL AND LEGAL ISSUES

TBD

7 CURRENT STATUS & FUTURE WORK

At Phase Two in this project timeline, the current status of the project consists of having a finished phishing email template, which contains the executable macro for the malicious Python code. Exploratory testing has also been done around macro-based attacks, which will be enabled based on user-acceptance in the Windows environment. During testing, it was found that obscuring malicious Visual Basic code behind the AutoOpen macro for Microsoft Office was able to get past all enterprise-level antivirus programs, with the exception of Windows Defender, which will act as a payload for the malware executable.

Additionally, in its current implementation, the link that contains the malware is converted into an executable file that will begin the encryption scheme once run. Additionally, the portion of the ransomware that is responsible for bypassing file permissions and the encryption/decryption methods on multiple files via multi-threading has been finished and tested on a Windows virtual machine in a controlled environment. For the time being, there are currently no deviations from the initial project proposal and what has been implemented thus far.

Going forward onto Phase Three of the project plan, it is expected to have the game of Snake in Python mostly working, where completing the game will cause the decryption of the user files to take place. From there, there will be an anti-malware tool that can exploit the developed malware in order to recover the encrypted files on the test virtual machine without the need for playing Snake. With regards to that, it is indicated that the anti-malware will also attempt to crack AES as a viable way of recovering the encrypted files, however this is now considered a low-priority task in the interest of time. In terms of the document, it is also expected to have all sections filled in with content and in an almost-ready state by Phase Three

REFERENCES

[1] U.S. Cybersecurity and Infrastructure Security Agency. 2019. Ransomware. (2019). https://www.us-cert.gov/Ransomware

[2] Paul Ducklin. 2015. Why Word malware is BASIC: SophosLabs takes apart a booby-trapped document. (Sep 2015). https://news.sophos.com/en-us/2015/09/28/why-word-malware-is-basic/

[3] Brett Williams Jarded Myers. 2017. Threat Analysis: Word Documents with Embedded Macros Leveraging Emotet Trojan. (Aug 2017). https://www.carbonblack.com/2017/08/28/threat-analysis-word-documents-embedded-macros-leveraging-emotet-trojan/

[4] Alison Grace Johansen. 2018. What is ransomware and how to help prevent ransomware attacks. (2018). https://us.norton.com/internetsecurity-malware-ransomware-5-dos-and-donts.html

[5] Kaspersky Lab. 2018. What are the different types of ransomware? (2018). https://www.kaspersky.co.uk/resource-center/threats/ransomware-examples

[6] NetApplications. 2020. Operating System Market Share. (Jan 2020). https://netmarketshare.com/operating-system-market-share.aspx

[7] Nathaniel Popper. 2020. Ransomware Attacks Grow, Crippling Cities and Businesses. (Feb 2020). https://www.nytimes.com/2020/02/09/technology/ransomware-attacks.html

[8] Davey Winder. 2019. New Orleans Declares State Of Emergency Following Cyber Attack. (Dec 2019). https://www.forbes.com/sites/daveywinder/2019/12/14/new-orleans-declares-state-of-emergency-following-cyber-attack/#6b0530b56a05