Malware programming:

Hackers' stuff?

Author: Rita Alonso Casablancas

Tutors: Ramon Balcells, Ana Garciolo

Center: INS Aran

Course: 2022/23

Type : Science / Technology, ICT

Introduction

I have always been passionate about ICT security and hacking, so I chose this subject because it is a field which I had not studied before. I consider that malware plays a very important role in hacking, so I wanted to learn more, and this assignment was a great opportunity. Apart from the relation it has with what I want to study, it is a good way to learn to exploit vulnerabilities and to expand my programming knowledge.

The objectives of this paper were:

- Explain what malware is, classify the different types existing and describe their functions.
- Determine the difficulty to design and create a malware which can really cause some kind of damage, and demonstrate that it is not more difficult than programming other kinds of applications, by coding one myself.
- Learn and improve my programming skills.

I prioritized the functionality of the malware over the quality of the code or the methods used to obtain those functions. However, I always tried to optimize the code and of course, do the best and simplest possible. Also, I explained it as detailed as possible to make it more understandable.

I seeked to learn more about malicious coding and how this type of software is produced and distributed, in order to increase my understanding of this security threat.

Brief summary of the paper

1. What is Malware

Malware, short for malicious software, is any intrusive software intended to harm computers, networks and other associated devices by stealing information, corrupting files and threatening users' privacy. It spreads mostly through networks and portable devices (less common at present), and transfers to devices without the knowledge of the user.

Brief history

Between 1971 and early 2000, malware was mostly relegated to mischief and attempts by virus authors to see if something they had created would work. During the late 1980s, malwares were simple boot sectors and file infectors spread via floppy disk. In the 1990s, macro viruses, which spread via email attachment and exploited Microsoft Office products proliferated due to the increased use of email.

Examples of malware of these years are The Creeper, Elk Cloner, Brain, Morris worm, AIDS and Michelangelo

From the year 2000, the threat landscape has evolved from mischief to include profitable cybercrime and nation-state attacks. An increase in the use of exploit kits, programs used by cybercriminals to exploit system vulnerabilities, led to another increase in malware delivered online. Since 2007, when some ways to mass compromise websites eased distribution capabilities of malware, the number of attacks has grown exponentially. Socially engineered worms and spam proxies began to appear, as well as phishing and other credit card scams.

Some malwares of these years are ILOVEYOU, Blaster, MyDoom, CoolWebSearch, Stuxnet, Flame, CryptoLocker, WannaCry, GandCrab and CovidLock

Programming languages

There are two types of programming languages:

- Low-level: their instructions have direct control over the hardware, the processor can run low-level programs directly without the need of a compiler or interpreter. Low-level languages are binary, machine code and Assembly. Programs written in these tend to be relatively non-portable.
- High-level languages: they have strong abstraction from the details of the computer, so the programs are independent of a particular type of computer. They may use natural language elements. These languages need an interpreter or compiler to translate them to low-level code.

When it comes to malware, most of it is written in either C or C++ or some other compiled language, although many times it depends on what platform the attacker is willing to target. Assembly is also a consistent choice.

Propagation

The most common method to spread malware is through phishing emails, although malvertising is also a common source of malware infections.

Infection requirements

For a malware to be able to run on a defined operating system, it has to be written according to that system's API. Else, it will become innocuous. Also, if the OS's antivirus software detects the malware, it will delete it immediately so the infection will not happen either.

2. Types of malware

- Virus: fragment of code embedded in a legitimate program that can replicate and spread to other programs after a person first runs it on their system. It is able to modify or destroy essential files, which may cause system crashes, program malfunctions and data loss.
- Worm: can propagate or replicate itself from one computer to another without human interaction after having accessed a machine. It can drop other malware, delete certain files or steal data, open a backdoor…
- Trojan: any malware that seeks to mislead the user of its true intent. It is used to distribute other kinds of threats or to create backdoors.

- Ransomware: threatens to publish or block access to the victim's personal data on a computer system unless a ransom is paid. It is the top variety of malicious software.
- Fileless malware: is designed to work in volatile system areas such as the system registry. It can also hide its code inside existing benign files
- Wiperware: is the most destructive form of malware. It causes damage by erasing the hard drive of the computer it infects.
- Grayware: is not necessarily harmful but is often unpleasant or irritating. It may monitor and capture personal and sensitive data or display advertisements within a web browser.
- Rootkits: enable access to a target device and control an area of the software that is not accessible for a normal user.
- Cryptojacking: hides on a computer and uses the machine's computing power to generate cryptocurrencies.
- Rogueware: deceive users into believing their computer is afflicted with a virus.
- Crimeware: any malware designed for the express purpose of conducting criminal activities online.
- Bots: converts a computer into a zombie device and connects it to a botnet that a cybercriminal can control at will and use to perform other types of attacks.
- Unix, Mac and Android malware: There is few malware for Unix systems because it is more difficult to gain root access. In Mac, it is because it has a lot of updates which solve vulnerabilities. Android is the most susceptible to malware infection, because there are fewer layers of protection in downloads.

3. Prevention

Some ways to protect from malware are installing antimalware programs, limiting the use of administrative accounts, keeping software updated, installing firewalls and limiting application privileges or access to sensors.

4. Experimental part

I am willing to demonstrate that we are closer to malware than we think and that programming malware is relatively easy. To do so, I will attempt to create one completely from scratch. I will use Python to program it.

I decided to create a ransomware because I found it the most threatening kind. I used Python's library 'cryptography', specifically the Fernet module for symmetric encryption.

Version 1

Using the Fernet module's methods, I created four functions, one to generate the key in the current directory, one to load that key into a variable, one to encrypt and another to decrypt the data. This first program, consisting of two scripts (one to decrypt and the other to decrypt) allowed me to, running the first one, encrypt all the files in a specified folder, and running the second one, decrypt them. It also created a text file in the encrypted directory claiming that it had been encrypted.

This version is stable, and can be run without any problem, but the computer has to have installed Python, as well as the required libraries.

Version 2

In version two, I added to the previous version the following features: encryption of a whole user directory, filtration of the files according to their extension and appearance of a window where a password has to be entered in order to decrypt the files.

Firstly, I created a function to move the program files to a directory which I know for sure that will not be encrypted, to prevent them from being encrypted. Then another function gets the users and scans them to find all the encryptable files. Another function will display a window, with a space to enter a password. If the password is correct, it will call the decrypt function, else, it will show an error message.

This version is also stable, but each function has to be stored on separate files because elseway, Windows antimalware detected it as a potential threat. I was able to compile the program so it can run on any computer. The problem of this version is that the key file is fairly accessible for the user, so the data could be decrypted without knowing the password.

Version 3 (beta)

This version is intended to solve the problem of the previous one. It includes, apart from version 2's features, asymmetric encryption of the key through a web server: when the files are encrypted, a function makes a request to a server, also created by me, which sends a private key that is used to encrypt the symmetric key. Only when the correct password is entered, another function sends it to the server, and the server returns the private key, which is used to decrypt the symmetric key and ultimately, the data itself.

This version is in development, and has not been tested yet.

5. Conclusion

I consider that the objectives have been accomplished: I have explained and classified the types of malware, created a fully functional malware and improved my programming skills.

The theoretical part was relatively easy, it was only an exposition of theory, so I only had to look for the information and summarize it.

For the practical part, I expected to create a more complex program, but this one turned out more difficult than I thought it would be, especially in the asymmetric encryption and the web server scripts. In any case, I have accomplished my main objective and demonstrated that it is really easy to program malware with little programming knowledge.

Although I think that with a little bit more time dedicated I could have accomplished greater and more interesting results, I am in general satisfied with the ones I obtained, especially from the version 2 of the program, which took me some time to complete. This project was quite interesting from the start, and I learned a lot of things that I would like to put into practice. I had the opportunity to explore, experiment with, and ultimately create a variety of scripts that helped me to better understand how malware works. In the future, I will surely try to apply this knowledge to create other types of malware, web exploits and other kinds of applications used in hacking and security audits.

A difficulty that I encountered was that the programs coded in python could only be run in computers which had Python and all the required libraries installed. I managed to overcome this situation in the second version of mocaMW, joining all the functions in the same file and then compiling it to transform it to an executable (.exe) file. Another problem I had was that whenever Windows antimalware service scanned the computer, if I created all the functions in the same file, it would erase that file because it detected it as a grave threat. I solved this, as I said, creating multiple files, one for each function, and importing them to each other whenever necessary. However, the complete program in one file is available on GitHub.

6. Bibliography

Avizienis, S. [et. al.] (2016, January 20). *Malware propagation modeling considering software diversity and immunization*. Journal of Computational Science. Retrieved December 14, 2022, from https://www.sciencedirect.com/science/article/abs/pii/S1877750316300035

Brathwaite, S. (2021, August 23). *Top programming languages for malware analysis*. Cybrary. Retrieved December 14, 2022, from https://www.cybrary.it/blog/top-programming-languages-for-malware-analysis/

Cisco. (2022, June 6). *What is malware? - definition and examples*. Cisco. Retrieved December 14, 2022, from https://www.cisco.com/c/en/us/products/security/advanced-malware-protection/what-is-malware.html

The difference between a virus, Worm and trojan horse. DigiCert. (n.d.). Retrieved December 14, 2022, from https://www.websecurity.digicert.com/security-topics/difference-between-virus-worm-and-trojan-horse

Higher level and lower level languages. Higher level and lower level languages - Computer Science Wiki. (n.d.). Retrieved December 14, 2022, from https://computersciencewiki.org/index.php/Higher_level_and_lower_level_languages

Landesman, M. (2021, March 10). *The first 25 years of malware*. Lifewire. Retrieved December 14, 2022, from https://www.lifewire.com/brief-history-of-malware-153616

VM introspection - university of manchester. Manchester Research. (n.d.). Retrieved December 14, 2022, from https://www.research.manchester.ac.uk/portal/files/32297162/FULL_TEXT.PDF