BY TEAM 38

Self Replicating Malware

3RD SEM PROJECT

Abstract

A project for the planning , development and creation of a computer worm commonly known as a *self replicating multi-functional malware*.

Employing a modular approach and using c++ standard library functions will increase cross-user compatibility and readability.

The final executable will be designed to spread to different directories through self replication. The nature of the design will allow free addition and removal of payloads (of different severity) via editing of the source code.

Abstract

Any code designed to do more than spread the worm is typically referred to as the "payload"of the worm,this is the malicious part of a worm.

Typical malicious payloads might delete files on a host system (e.g., the ExploreZip worm), encrypt files in a ransomware attack, or exfiltrate data such as confidential documents or passwords.Some special worms attack industrial systems in a targeted manner most notably, Stuxnet.

Stuxnet was primarily transmitted through LANs and infected thumb-drives.

This virus can destroy the core production control computer software used by chemical, power generation and power transmission companies in various countries around the world.

Existing /Similar Works

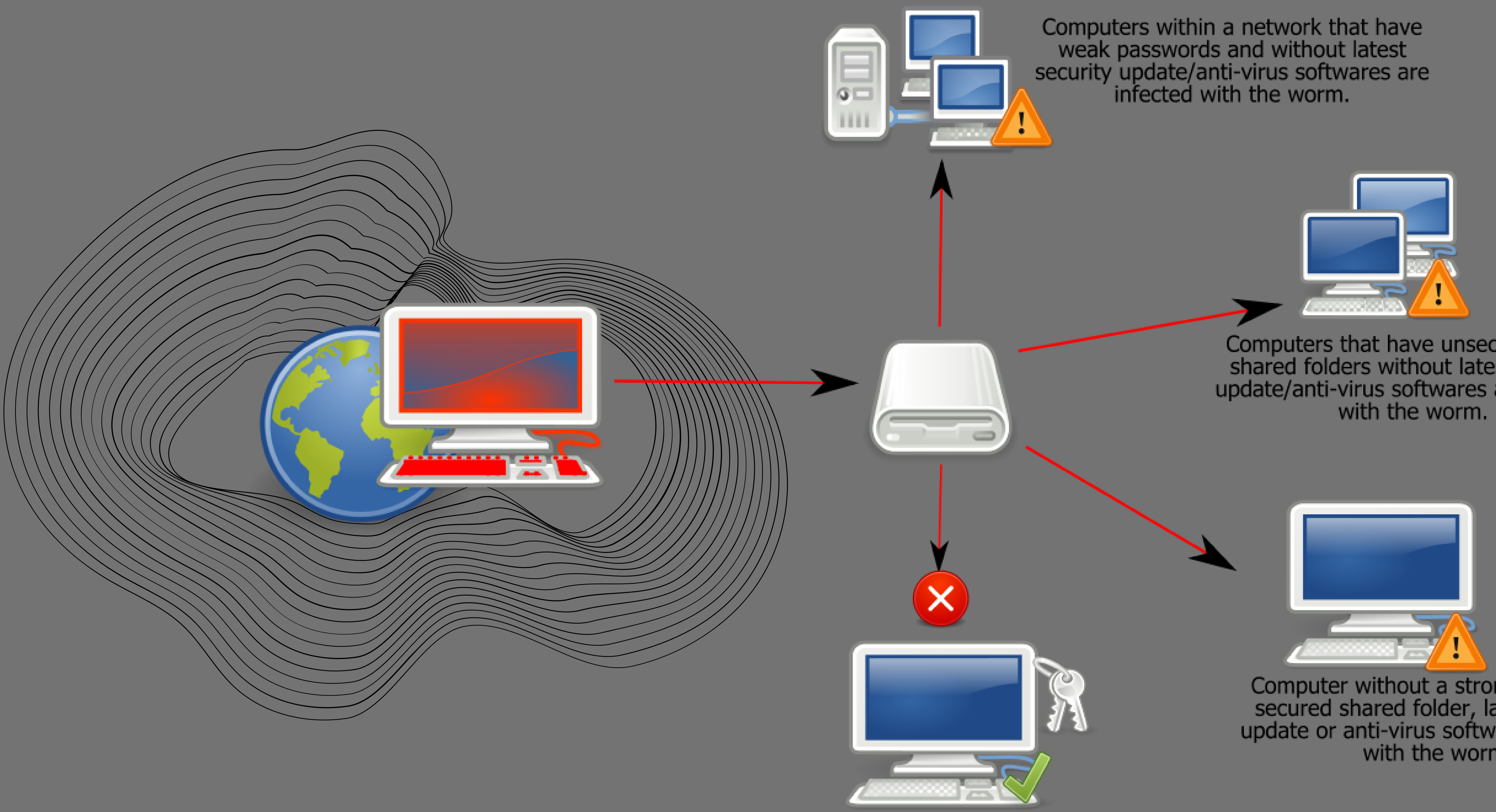
- Beginning with the very first research into worms at Xerox PARC, there have been attempts to create worms. Those worms allowed John Shoch and Jon Hupp to test the Ethernet principles on their network of Xerox Alto computers.

- Morris Worm created by graduate student Robert T. Morris at Cornell University back in 1988.It disrupted many computers conected to the Internet, guessed at the time to be one tenth of all those connected.
  Morris himself became the first person tried and convicted under the 1986 Computer Fraud and Abuse Act.

- Conficker, also known as Downup, Downadup and Kido, is a computer worm targeting the Microsoft Windows operating system that was first detected in November 2008.

EXISTING WORK 1

Worm:Win32 Conficker



Computers within a network that have weak passwords and without latest security update/anti-virus softwares are infected with the worm.

Computers that have unsecured/open shared folders without latest security update/anti-virus softwares are infected with the worm.

Computer without a strong password, secured shared folder, latest security update or anti-virus software is infected with the worm.

Computer with strong password, secured shared folder, latest security update and anti-virus software is protected from the worm.

EXISTING WORK 2

Proposed Work

- A malware capable of acting independently and self regulates its functions as per initial instructions inputted by the creator.

- One of the proposed modules will be capable of achieving administrator level privileges after which it shall signal it to the user.

- Since we are using Win32 API, the level of complexity will only be bounded by the functions present in the WinBase section of the API.Anyone with basic Microsoft Win32 knowledge can change it as per their wish.

- For the purpose of safety we shall test it in a virtual machine and build it such that it requires preliminary initiation when it starts up for the safety of our personal computers.

Novelty of Work

- The key difference between the proposed work and existing work will be the modularity and generally easy to understand approach to a complex malware.

- Our team will use C/C++ and Python since :
  - it is a very popular amongst software developers.
  - C++ and python both encorporate object oriented and modular functionality approach.
  - Most OS are written partially or completely in C/C++ and heavily depend on python.

- Our source code will be short and the executable will be portable enough to carry in a usb drive.

NOVELTY

Conclusion

- Worms are a very interesting variety of computer malware which is the reason for our focus.
- Our team will use generalised methods to create the malware.
- We will also try to make it as modular as possible for future users(if any).
- Our executable malware will be portable, concise and will be attempt to be independent to a safe degree.
- We will attempt to make its behaviour such that it remains undetected by host user(s) unless a thorough inspection of all currently running processes is done.

Software Resources

- C++ v17
- Python (optional)
- Notepad++ (Basic Text Viewing)
- Microsoft Visual Studio 2019 (Primary Compilation)
- Windows 10 Home Ver.2004
- MS DOS Box (backwards compatibilty debugger/testing)
- Virtual Machines (Test Emulation)

REFERENCES

Knowledge Resources

- CPP Reference -

  https://en.cppreference.com/w/cpp

- C++ Core Guidelines By Bjorn Stroustrup

- Microsoft WinBase Documentation -

  https://docs.microsoft.com/en-us/windows/win32/api/winbase

- Wikipedia Org -

  https://en.wikipedia.org/wiki/Computer_worm

- Visual Studio Editor Help Section

REFERENCES

End of Presentation

Ghanishth Goyal

Rajat Gaur

Prathamesh Auti

Sirshak Sarkar

Under Guidance of Mr.Muneeswaran V

Cyber Team **38**