National University of Computer and Emerging Sciences

Network and Cyber – 2

Alternate Assignment 1

Objective:

Students will understand the concepts of malware behavior by creating a simulation that demonstrates how malware can manipulate file systems. This assignment will help students gain insights into how malware operates, particularly focusing on file scanning and data corruption techniques.

Scope:

The assignment involves developing a simple malware simulation that:

1. Scans the Windows directory structure.
2. Identifies files and folders with specified names.
3. Replaces these files and folders with garbage data (placeholder files).

Requirements:

1. Programming Language: Any programming language can be used.
2. Controlled Environment: Use virtual machines with isolated environments to avoid any damage to actual systems.

Assignment Instructions:

1. Setup and Preparation:

- Create a virtual machine (VM) running Windows.
- Set up a designated folder structure within the VM for testing purposes (e.g., D:\).

2. Task Breakdown:

a. Directory Scanning:

- Write a code that scans the Windows directory structure starting from a specified root directory (D:\).
- Implement functionality to recursively scan subdirectories.

b. File and Folder Identification:

- Define a list of target file and folder names that the malware simulation should look for.
- Implement the logic to identify files and folders matching these names.

c. Replacement with Garbage Data:

- For each identified file, replace it with a placeholder file containing garbage data.
- For each identified folder, replace it with a folder containing placeholder files with garbage data.

d. Logging and Reporting:

- Implement logging to keep track of which files and folders were identified and replaced.
- Provide a summary report of the operations performed by the simulation.

3. Documentation:

- Prepare a detailed report describing how the malware simulation works.
- Explain the potential impact of such malware on a real system and discuss countermeasures to prevent similar attacks.

4. Submission:

- Submit the code, the test environment setup, and the final report.
- Provide evidence of the test results showing how the script interacted with the test environment.