# January 2023 CSE 406
# Assignment on Malware

Last Modification: July 21, 2023

Deadline: August 4, 2023, 11:55 PM

## Overview

Malware, or malicious software, is any program or file that is intentionally harmful to a computer, network, or server. Malware can infect networks and devices and is designed to harm those devices, networks, and/or their users in some way.

Different types of malware have unique traits and characteristics. Types of malware we will explore in this assignment:

- **Virus:** A virus is the most common type of malware that can execute itself and spread by infecting other programs or files.
- **Worm:** A worm can self-replicate without a host program and typically spreads without any interaction from the malware authors.

## Setup

We will use the pre-built SEED Ubuntu 20.04 VM for this assignment.

### Docker

We will use docker for the emulation process. Note that you do not need to know the details of docker in this assignment. All the required command are given in the next section and all the required software are already installed in the SEED Ubuntu 20.04 VM.

### Commonly Used Commands

Here, some of the commonly used commands related to Docker are listed. Since you are going to use these commands very frequently, aliases have been created for them in the .bashrc file in SEED Ubuntu 20.04 VM. You can use the aliases instead of the full command.

```
$ docker-compose build  # Build the container images
$ docker-compose up     # Start the containers
$ docker-compose down   # Shut don the containers

// Aliases for the Compose commands above
$ dcbuild       # Alias for: docker-compose build
$ dcup          # Alias for: docker-compose up
$ dcdown        # Alias for: docker-compose down
```

All the containers will be running in the background. To run commands on a container, you would often need to get a shell on that container. You first need to use the ”docker ps” command to find out the ID of the container, and then use ”docker exec” to start a shell on that container. Aliases have been created for them in the .bashrc file.

```
$ dockps        // Alias for: docker ps --format "{{.ID}}  {{.Names}}"
$ docksh <id>   // Alias for: docker exec -it <id> /bin/bash

// The following example shows how to get a shell inside hostC
$ dockps
b1004832e275  hostA-10.9.0.5
0af4ea7a3e2e  hostB-10.9.0.6
9652715c8e0a  hostC-10.9.0.7

$ docksh 96
root@9652715c8e0a:/#

// Note: If a docker command requires a container ID, you do not need to
//       type the entire ID string. Typing the first few characters will
//       be sufficient, as long as they are unique among all the containers.
```

## Reading Materials

Read **FooVirus_AbraWorm.pdf** carefully and make an effort to understand the **FooVirus.py and AbraWorm.py** files completely.

## Task 1

Taking cues from the code shown for **AbraWorm.py**, turn the **FooVirus.py** virus into a worm by incorporating networking code in it. The resulting worm will still infect only the **'.foo'** files, but it will also have the ability to hop into other machines.

## Task 2

Modify the code **AbraWorm.py** code so that **no two copies of the worm are exactly the same** in all of the infected hosts at any given time.

One way to accomplish this would be by inserting worm alteration code after the comment line

# Now deposit a copy of AbraWorm.py at the target host:

that you see near the end of the main infinite loop in the script. This additional code in the worm could insert some extra newline characters between a randomly chosen set of lines, some extra randomly selected characters in the comment blocks, some extra white space between the identifiers in each statement at randomly chosen places, and so on. And if you are ambitious, you can get the worm to modify the code in more significant ways (without altering its overall logic) before depositing a copy of itself in a target host. For example, since you can use different control structures for infinite loops, you could randomly choose from amongst a given set of possibilities for each new version of the worm. The net result of all these changes on the fly will be that you will make it much harder for the worm to be recognized with simple signature-based recognition algorithms.

## Task 3

If you examine the code in the worm script **AbraWorm.py**, you'll notice that, after the worm has broken into a machine, it examines only the top-level directory of the username for the files containing the magic string **"abracadabra."** Extend the worm code so that it descends down the directory structure and examines the files at every level.

## Prerequisites

The Attacker and Victim machines need to have **paramiko and scp** installed for the attack to work correctly.

- sudo apt-get update -y
- sudo apt-get install -y python3-paramiko
- sudo apt-get install -y python3-scp

## Submission Guidelines

You have to submit the following two contents:

- A **detailed lab report**, with screenshots, to describe what you have done and what you have observed for each task. The PDF would be named in the format "**1805123_AssignmentReport.pdf**".
- A folder containing all the source code files you needed to modify. Please only place the source code of the files you had to modify to complete all tasks. **The prefix of all of your source files must have your student id and task no. (i.e. 1805123_1.py, 1805123_2.py, 1805123_3.py).** Placing unnecessary source files/executables would result in mark deduction. The folder should be named in the format "**1805123_Code**".

Now create a folder named after your 7-digit roll number. Place the pdf and the folder containing the source codes in this folder. Zip the folder and submit it in Moodle. The zip file would be named in the format **1805123.zip.**

## Deadline

**August 4, 2023, 11:55 PM (Strict)**.

## Mark Distribution

The tentative mark Distribution are given below:

| Task | Mark |
|---|---|
| Task 1 | 20 |
| Task 2 | 30 |
| Task 3 | 20 |
| Report | 25 |
| Correct Submission | 5 |
| Total | 100 |

## Plagiarism

Do not copy from any web source or friends. Students involved in such activities will be severely penalized by awarding **-100%** of the total marks.

## Acknowledgment

This assignment was collected from the Computer and Network Security course by Avinash Kak. We modified it to make it suitable for this course.