Introduction to Computer Security

Project 2: Worms Replication through SSH and Its Detection

Chi-Yu Li (2019 Spring)
Computer Science Department
National Chiao Tung University

Goals

- Understand how a worm with the capability of hiding itself can replicate through SSH, and how to detect it

- You will learn about
  - SSH operations and login with public key authentication
  - Automatic remote login
  - Key generation for public key authentication
  - Analysis for abnormal processes on Linux
  - Routine task scheduling on Linux

Requirements

- You need to develop/run your program in a given virtual machine
  - VMware Workstation Player (or VirtualBox)
  - VM image: the same as Project I
    - Username/password: cs2019/cs2019
  - Language: any supported ones in VM
    - e.g., C/C++, shell script, etc.
- You are allowed to team up. Each team has at most two students
  - Discussions are allowed between teams, but any collaboration is prohibited
- Please submit your source codes/scripts, and use them for your demo
  - Makefile must be provided to compile your source codes
- TA: Wei-Xun Chen (chrissy81527@gmail.com)

Attack Scenario

- An attacker seeks to develop a worm that can automatically replicate through SSH and have the following capabilities
  - It relies on some techniques to crack username/password
  - Once it successfully logins with an account, it can enable its successful login even if the password is changed afterwards
  - Hiding in the victim system: (1) putting its attack module into hidden directories; (2) naming the attack module using a popular program's name
  - Payload: flooding attack against local loopback interface ('lo', 127.0.0.1)
  - Trigger: trigger the attack module automatically every 1 minute, if it is not running

Our Focus

- An attacker seeks to develop a worm that can automatically replicate through SSH and have the following capabilities
  - It relies on some techniques to crack username/password

    Assumption: a pair of username/password and a target IP are given

  - Once it successfully logins with an account, it can enable its successful login even if the password is changed afterwards
  - Hiding in the victim system: (1) putting its attack module into hidden directories; (2) naming the attack module using a popular program's name
  - Payload: flooding attack against local loopback interface ('lo', 127.0.0.1)
  - Trigger: trigger the attack module automatically every 1 minute, if it is not running

How to Proceed?

- Given materials
  - A VM for the attacker/victim development
    - Two VM copies: one copy for the attacker; the other for the victim
  - A zip file of the worm (i.e., worm_sample.zip)
    - Will be provided by 4/5
- Preparation
  - Create a victim account at the VM of the victim
    - username: victim; password: victim
  - Run the given worm binary at the VM of the victim with the superuser privilege (i.e., unzip the zip file and run with ./worm_sample)
    - An attack module is put into a hidden directory
    - Automatically run the attack module after reboot
    - Flooding attack against local loopback interface ('lo', 127.0.0.1)

Three Tasks

- Task I: Identify what the given worm does and remove it completely
- Task II: Develop a new worm by imitating the given worm and advancing it (55%)
- Task III: Enhance the new worm to replicate through SSH (30%)
- Demo Q&A (15%)

Task I

- Identify what the given worm does and remove it completely
  - Identify which hidden directory the attack module is put into
  - Check how it can be triggered automatically after reboot

Account: victim/victim

An attack module

- In a hidden directory H
- Automatically trigger after reboot
- Flooding local loopback interface

Victim VM

Hints

- Useful management tool on Linux: **htop**
  - Can be used to check the condition of each process
- Time-based job scheduler on Linux: **cron**
  - It is used in any Unix-like computer OSes

Task II

- Develop a new worm by imitating the given worm and advancing it
- It has the following capabilities
  - Hiding: put the attack module into ***two*** hidden directories including the directory H
    - The attack module name should be the same as the given one
    - The other directory H' is chosen by you
    - When one of attack modules is removed, the attack can still be launched by the other
  - Payload: flooding attack against local loopback interface
  - Trigger: trigger automatically every 1 minute, if it is not running

Account: victim/victim

Attack module

- In two hidden directories **H and H'**
- Automatically trigger every 1 min
- Flooding local loopback interface

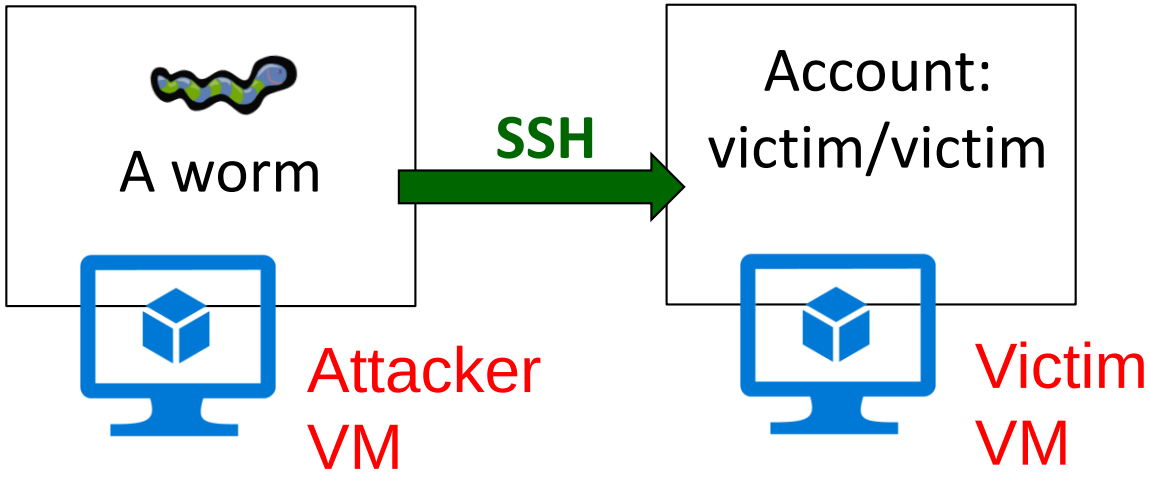
Victim VM

Task II (Cont.)

- Note
  - You are allowed to reuse any files in the given zip file (i.e., worm_sample.zip)
  - Be careful that don’t flood to the destinations outside the VM
  - The worm: may have multiple files including binary and scripts

Task III

- Enhance the new worm to replicate through SSH
- Given the victim's IP and username/password, running your binary file (or script) at the attacker can achieve the following things
  - Automatically connect to the victim through SSH
  - Send the worm (from Task II) to the victim, and execute it
  - Configure the victim so that it can successfully login to the victim even if the password is changed afterwards
    - E.g., generating keys and configuring SSH



Hints

- SSH (Secure Shell): a cryptographic network protocol for operating network services securely over an unsecure network
  - Can be used for any network services
  - Generally used to access Unix-like OSes, but also available on Windows
- Authentication: two approaches
  - Password authentication
  - Manually generated public-private key pairs

Example of Demo Procedure

- At the attacker, make and run your worm binary file (or script) with the given parameters
  - The victim's IP address and username/password
- At the victim, TA will ask you to do the following
  - Show how the victim can identify your attack module and its hidden directory
  - Terminate the ongoing attack (module), and see whether it runs again after 1 min
  - Delete the binary of the identified attack module, and see whether another binary will be triggered after 1 min
  - Remove the second binary file
  - Change the victim's password, and see whether the worm can be deployed successfully again after running the worm binary file (at the attacker)

Project Submission

- Due date: 5/1 11:55 p.m.
- Submission rules
  - Put all your files into a directory, the name of which is your student ID, and upload its zip file to New e3
    - Including only source codes, scripts, and Makefile
    - Your source codes must be able to be compiled by your own Makefile
  - If your team has two members, please create a zip file with the name as the concatenation of your IDs separated by "-"
  - Sample: 1234567.zip or 1234567-7654321.zip
    - 1234567
    - Makefile
    - worm.cpp
    - wormsh.sh
    - …

Demo

- Date: 5/2 (9:30a.m.-5p.m.) @ EC315
- TA will prepare your zip file for you to demo
- You will
  - be asked to replicate your worm and verify your worm behaviors at the victim
  - be only allowed to “make” to compile all your files, and run your worm binary (or script)
  - be not allowed to modify your codes or scripts
  - be asked some questions
    - e.g., How do you do automatic SSH Login even when the password is changed (explanation with your codes)? How to defend against this attack?
  - be responsible to show the outcome to TA and explain why you have successfully achieved the goals of Tasks II and III

Questions?