Introduction to Computer Security

Project 3: Worm Hiding/Propagation and Its Detection

Chi-Yu Li (2020 Spring)
Computer Science Department
National Chiao Tung University

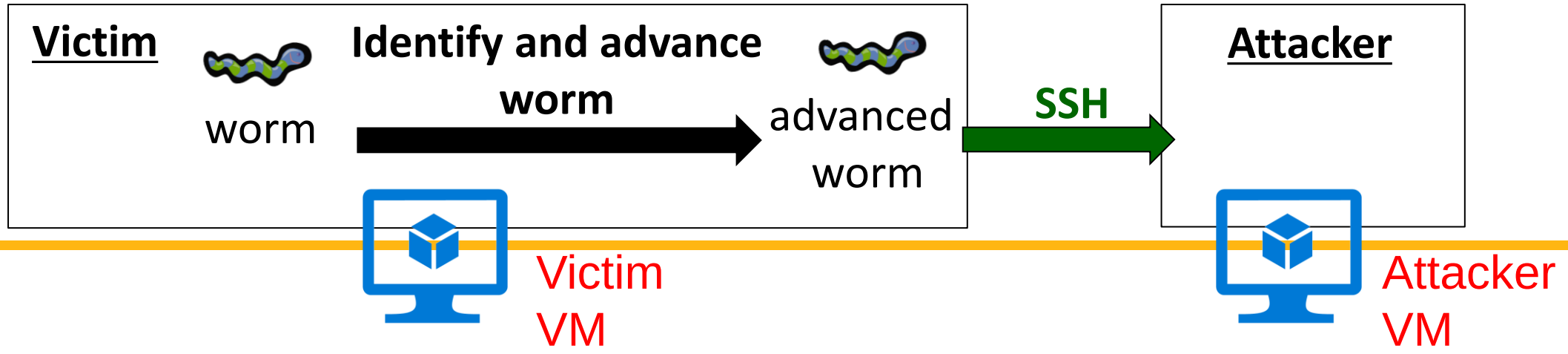
Goal

- Understand how a worm hides and propagates itself and its detection

- You will learn about
  - SSH security and command operations
  - Simple methods for ciphering and deciphering (stream and block ciphers)
  - Analysis of abnormal processes on Linux
  - Routine task scheduling on Linux

Attack Scenario

- An attacker has successfully deployed a worm in a victim system
- Assume that you are the victim by having a victim VM and you know where (IP address) the attacker machine is
  - You are asked to identify and remove the worm
  - You are also asked to take revenge on the attacker by deploying an advanced worm in the attacker system



Attack Scenario (Cont.)

- Advanced worm's capabilities
  - Propagation
    - Cracking the attacker's password by launching a dictionary attack
    - Propagating itself to the attacker system via SSH
  - Hiding itself in the attacker system
    - Putting itself into multiple hidden directories
    - Naming itself using a popular program's name
    - Supporting a simple recovery mechanism
  - Payload
    - Doing RSA encryption on all the files of the attacker's desktop directory (/home/attacker/Desktop)
    - Launching DoS based on a Ping flooding attack
  - Trigger
    - The payload is triggered automatically every 1 minute

原本在victim裡面的只會在reboot的時候被triggered 所以這裡要修改成每一分鐘都被triggered一次

Requirements

- You need to develop/run your program in a given virtual machine
  - VM image: TA will send out the link on 5/8
    - [Victim account] username/password: victim/victim
    - [Attacker account] username: attacker (the password needs to be cracked)
- You are allowed to use C/C++, Shell Script or/and Python
- You are allowed to team up. Each team has at most 2 students.
  - Teams: discussions are allowed, but no collaboration
- Please submit your source codes/scripts and report to E3

Three Tasks

- Task I: Identify and remove the worm in the victim system (20%)
- Task II: Develop a new worm with the specified capabilities (50%)
  - Imitating the given worm's hiding, payload, and trigger actions
- Task III: Report (30%)

Task I: Identify and Remove the Worm

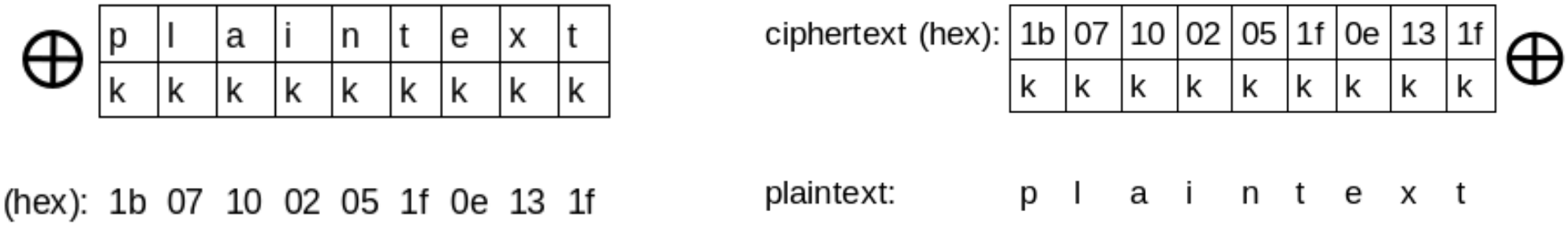
- Identify where (which directory) the worm is
  - Worm: two attack modules (encryption and flooding) 可以重複使用這兩個binary file
- Check how it can be triggered automatically after system reboot
- Hints
  - A useful management tool on Linux: **htop**
    - Used to check the condition of each process
  - A time-based job scheduler on Linux: **cron**
    - Used in any Unix-like computer OS

Victim account in the given VM
Worm behaviors - **Hiding**: Hidden in a certain directory **H** - **Trigger**: Automatically triggered after system reboot - **Payload I**: Encrypting the files in /home/victim/Desktop using XOR ciphering - **Payload II**: Ping flooding

Victim VM

Task I: Identify and Remove the Worm (Cont.)

- Achievement verification
  - Mark your student ID in a file located in the hidden directory **H** with the worm
    - Replace the value of a verification flag with your ID
    - E.g., Verification_flag: 1234567 → Verification_flag: 0756436
  - However, the file is encrypted by the XOR stream cipher
    - The key length is single byte (You should figure out the key by yourself)
    - E.g.,

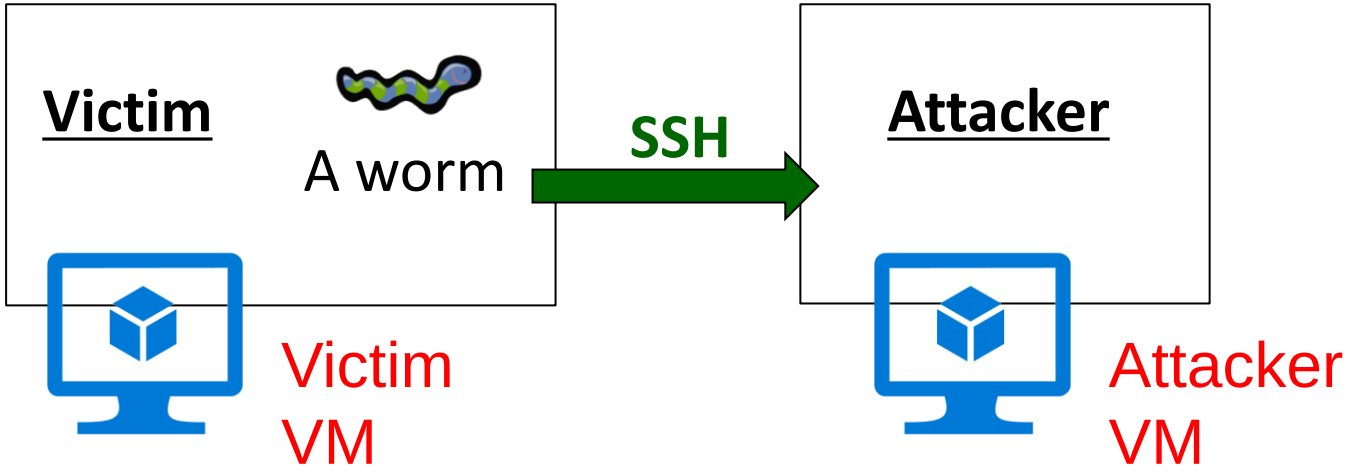


Task I: Identify and Remove the Worm (Cont.)

- Achievement verification actions
  - Decipher the file and mark your student ID
  - Cipher the modified file using the same key
  - Rename the file with “task1_result.log”
  - Include the final file in the final submission package

Task II: Develop a New Advanced Worm

- Once the worm is executed in the victim system, the worm can propagate itself (two attack modules) to the attacker system
  - Only the attacker's username and IP address are given
  - Two attack modules: ping flooding and file encryption



Task II: Develop a New Advanced Worm (Cont.)

- Advanced worm's capabilities
  - Propagation
    - Cracking the attacker's password by launching a dictionary attack
    - Propagating itself to the attacker system via SSH
  - Hiding itself (or two attack modules) in the attacker system
    - Putting it into two hidden directories including the directory **H**
    - The other directory **H'** is /home/attacker/Desktop/.Backup
    - When the attack modules in one directory are removed, the payload can still be triggered from those in the other directory
  - Payload
    - Doing RSA encryption on all the files of the attacker's desktop directory (/home/attacker/Desktop)
    - Launching a Ping flooding attack
  - Trigger
    - The payload is triggered automatically every 1 minute

Task II: Worm Propagation

- Cracking the attacker's password using a dictionary attack
  - Assume that the password is created based on the attacker's personal information
    - A file including the attacker's personal information is given: /home/victim/materials/attacker.dat
    - Note: the password is composed of only some information entries
- Hints
  - A module for trying string combination in Python: **itertools**
  - Automatic SSH and SFTP operation in Python: **paramiko**
  - Passing a password to the ssh command in Shell: **sshpass**

Task II: Payload

- Doing RSA encryption on all the files of the attacker's desktop folder
  - An RSA encryption/decryption binary is given
    - /home/victim/materials/RSA/RSA_encrypt
  - A set of public/private keys is also given
    - /home/victim/materials/RSA/key.dat
  - Each trigger: check any unencrypted files in the directory and encrypt them **using the public key**
    - Note: you should avoid encrypting a file more than once
  - Hint
    - A crontab management module in Python: **crontab**

這邊獨立寫一個排程的python檔案
透過SSH傳到目標之後
再用paramiko遠端執行這個獨立的python code

- Launching a ping flooding attack
  - You can use the ping flooding binary found in Task I or write it by yourself

Task II: Payload (Cont.)

- Hint: Verification by TAs
  - Creating several files in /home/attacker/Desktop
  - After launching your worm
    - Deciphering the files in /home/attacker/Desktop with the private key
    - Checking whether they are the same as the original ones
  - After killing the ping flooding process
    - Checking whether there are many ICMP packets 1-2 min later

Task III

- Item 1 (10%) : Please describe how you finished Task I
  - Only description is sufficient and no more than 200 English words
- Item 2 (10%) : Please propose three security settings in SSH server that can prevent common dictionary attack
  - Description should be clear and no more than 200 English words
- Item 3 (10%) : Please explain why Linux differentiates crontab into three types (users, system and applications).
  - Description should be clear and no more than 200 English words
- Note the report must be written in English with font size 11 or 12 in Times New Roman. It must be submitted in one PDF file with a name “report.pdf”.

Important: How to prepare your worm and flag files

- Must provide a **Makefile** which compiles your source code into one executable file named **worm_revenge** (Missing: -20%)

- Test requirements for the program (Missing: -10% for each)
  - Must be run in the given VM without any additional tools or libraries
  - Must work for the test command: ./worm_revenge <Attacker IP>
    - E.g. ./worm_revenge 10.0.2.5
  - After being executed, the worm should propagate the attack modules through SSH, place the trigger, trigger the payloads, and then terminate

Project Submission

- Due date: 6/4 11:55 p.m.
  - Makeup submission: 6/21 11:55 p.m. (75 points at most)
- Submission rules
  - Put all your files into a directory and name it using your student ID(s)
    - If your team has two members, please concatenate your IDs separated by "-"
  - Zip the directory and upload the zip file to New e3
  - A sample of the zip file: 1234567.zip or 1234567-7654321.zip
    - 1234567 or 1234567-7654321 (Directory Name)
      - Makefile
      - Worm.cpp
      - report.pdf
      - task1_result.log
      - …

Questions?