ECE30021/ITP30002 Operating System

Programming Assignment 1

LKM Rootkit

Overview

- You are asked to create a Linux kernel module (LKM) that works as an agent in kernel space for your commands
  1. Log the names of files that a user has accessed
  2. Prevent a kill to a specified process
  3. Hide the dogdoor module from the module list

- Three examples are given to build up backgrounds

- You have your own Ubuntu 16.04 machine as it requires you to have root permission

- PA1 should be done as a team work with your partner (2 persons team)
  - tentative team assignments will be announced
  - you need to contact with your team member within next 24 hours for initiating collaboration
  - you can reclaim your team assignment if you cannot have a meeting with your tentative partner within next 24 hours

Schedules

- 19 Mar (Tue)
  First announcement
  - Initial team setup
- 20 Mar (Wed), 9:00 PM
  Team assignment reclaim
- 20 Mar (Wed), 11:59 PM
  Final team assignment
- 21-29 Mar (Fri)
  TA help sessions (by appointment)
- 1 Apr (Mon), 11:59 PM
  Submission deadline
  - late submission is accepted only within the next 24 hr w/ 30% penalty

System Requirement

- Use Ubuntu 16.04 with Kernel 4.15.0 or higher
  - high chance that your LKM is incompatible and does not work correctly if you developed it under a lower version
  - recommend to use virtual instances
    - VMware: Ubuntu 16.04.6 LTS Desktop image http://releases.ubuntu.com/16.04/
    - Amazon EC2: Ubuntu Server 16.04 LTS (HVM), SSD Volume Type
      - the update instruction is given at `PA1/EC2.sh`

- Use GCC 5.4.0 or higher

Background: Linux Kernel Module

- A Linux kernel module (LKM) is a suite of functions in a file (i.e., module) that can be loaded to kernel space in runtime upon a superuser's request
  - usually compiled as a ko file
  - load by `insmod` ; unload by `rmmod` ; list up loaded modules by `lsmod`
  - e.g., device driver

- Example 1. `PA1/bareminimum`

- c.f., Writing a Linux Kernel Module – Part 1. Introduction
  http://derekmolloy.ie/writing-a-linux-kernel-module-part-1-introduction/

Backgrond: Proc as LKM Interface

- Proc is a virtual file system where files act as agents for a kernel data structure or kernel module to interact with user-level programs
  - usually placed at `/proc`
- A LKM can create a proc file with customized file operations to communicate with a user-level program in text
  - `write()` for receiving inputs
  - `read()` for sending out messages
- Example 2. `PA1/hellokernelworld`

Background: Intercept System Call

- A LKM can access to system data structures by a symbol name via the kernel symbol table
  - `(void *) kallsyms_lookup_name(char * name)`
- You can intercept a system call by replacing the handler routine with a function of your own
  - a list of syscall handler types can be found at `include/linux/syscalls.h`
- Example 3. `PA1/openhook`
  - count how many times a specified file get accessed

Useful Links

- Linux kernel 4.15 source code
  https://elixir.bootlin.com/linux/v4.15/source

- Kernelnewbies.org
  https://kernelnewbies.org/Documents

- Linux kernel programming tutorial
  https://linux-kernel-labs.github.io/master/index.html

Your Assignment

- Create a toy rootkit `dogdoor.ko`
  - three main functionalities
    - log the names of files that a user has accessed
    - prevent a kill to a specified process
    - hide the dogdoor module from the module list
  - create a text interface `/proc/dogdoor`
- Create a user-level program `bingo.c`, a CLI with the dogdoor module
  - communicate via `/proc/dogdoor`
  - you need to devise a small protocol for the communication

Main Functionalities (1/2)

1. Log the names of files that a user has recently accessed
   - the user specifies a user by its username (e.g., `guest`) to bingo
   - for the given user, dogdoor records the names of files (up to 10) that the user recently opens
   - when the user requests, bingo retrieves the lists and prints it to the user
   - Hint: `current->cred->uid` of `<linux/cred.h>`

2. Prevent a kill to a specified process
   - the user specifies a process ID number to bingo
   - then, dogdoor makes no other process kill the specified process, until the user commands to release this immortality
   - Hint: `sys_kill()`

Main Functionalities (2/2)

3. Hide the `dogdoor` module from the module list

- once the user gives a command, `dogdoor` makes itself disappear from the `lsmod` result
- once the user gives a command again (i.e., toggle), `dogdoor` makes itself appear again
- Hint
  - the list of loaded modules is maintained as kernel list
  - a list data structure of a module itself can be accessed by `THIS_MODULE->list`

Write Up & Demo

- Write up
  - Up to 5 pages (single- or double-column)
  - Describe how you accomplish implementing functionalities
  - Discuss issues or/and ideas as you had for the assignment
  - Submit a PDF file

- Demo
  - Create a scenario of demo to show that you complete the assignment
  - Videorecord program execution with narration, upto 10 min
  - Upload the video to a streaming service (e.g., YouTube) and submit the URL

Little Help from TA

- TA's
  - Mr. Jeewoong Kim jeewoong@handong.edu
  - Ms. Juyoung Jeon 21931009@handong.edu
- Services
  - Help equip Ubuntu systems for experiments
  - Repeat what's explained in this assignment description
- How to contact
  - ask a question on Piazza
  - make an offline meeting appointment via Piazza (as a public post) or email
    - less than 30 minutes, open to every one

Submission

- Your submission must include the followings
  - write-up: up to 5 pages (either in single- or double-columns)
    - your write-up will be open for peer evaluation
  - URL of your video demo (e.g., YouTube)
    - put the URL in your write-up
  - all related source code files

- How to submit
  - upload your files to a homework repository in Hisnet
  - by only one of the team member

Evaluation

- Points
  - Fulfillment of requirements 25%
  - Clarity in technical description 25%
  - Novelty in discussion 20%
  - Soundness of demonstration 20%
  - Peer evaluation 10%
  - Best peer review award up to extra 10%

- Notes
  - Evaluation will be primary based on your write-up and video demo
  - TAs will rehearse the demo with your submitted files on Ubuntu 16.04 Kernel 4.15.0