Rootkit Programming

Premeeting

Manuel Andreas & Fabian Franzen

Chair of IT-Security (I20)
Prof. Dr. Claudia Eckert
Technische Universität München

January 30, 2024

What is a Rootkit?

> A rootkit is a collection of computer software, typically malicious, designed to enable access to a computer or an area of its software that is not otherwise allowed (for example, to an unauthorized user) and often masks its existence or the existence of other software.
> — Wikipedia

Course Contents

In this course you will create your **own rootkit** (aka your own piece of malware) with the following features:

- Escalate privileges to root
- Hide files on disk
- Hide processes
- Hide network connections
- ...

Your rootkit will take the form of:

- Linux Kernel Module (LKM)
- Userspace Rootkit
- Hypervisor/UEFI (planned)
- eBPF (planned)

Even more, we will focus on the **detection of rootkits** using

- Virtual Machine Introspection (VMI) / Memory Forensics

Teaching Goals

- Linux kernel hacking
  - How to create your **own kernel module**
  - How the Linux kernel tracing system works
  - Getting familiar with **fundamental linux subsystems**
- Details about the Linux kernel boot process (e.g. initramfs)
- How the kernel, the loader and the libc interact with each other to execute a program
- How a hypervisor can **interact** with and **inspect** its running VMs

Prerequisites

We **do not** have formal requirements for students who want to join the course.

However, we **strongly recommend** being familiar with the following:

- How to write **a C program** and how **pointers** work
- What a **Syscall** is
- Basic knowledge about IT Security (IN0042) and how an operating system works in general (as taught in IN0009)

Having seen or worked with **assembly** is a plus!

Organizational Matters

- The course has **20 slots**
- We will meet once a week
- You will get **weekly** exercises, which are discussed and presented in the upcoming week (there are exceptions for large tasks!)
  - You therefore have to be present in class!
- You will work with a partner in teams of **two**
- We may finish with a **project** phase, depending on your interests
  - E.g implementing novel hiding / detection techniques

Registration

Awesome! How can I join?

We want to make sure that motivated students get places!

- **No** letter of motivation
- Instead, solve a **small qualification task**
  - Create a driver for a USB device[1] (in the form of a Linux Kernel Module), that reads out a secret value (flag).
  - Due at **14.02.2024 23:59** (end of matching period)
  - Download the challenge & submit your flag at `https://courses.sec.in.tum.de/rootkit`
- Nonetheless, do not forget to **register** yourself in the **matching system**!

[1] The USB device is completely made-up and only emulated by QEMU

Qualification Challenge Hints

- The challenge runs on the Linux kernel **v6.1.74**
  - Make sure to build your kernel module against this version
  - We provide you with the kernel configuration (**config-6.1.74**). Make sure to rename it to **.config** in the Linux kernel source tree before building the kernel
- We include the source code for our USB device as a **patch**
- Our challenge runs on an x86-64 CPU with **KVM** enabled
- We recommend you to fetch the appropriate kernel sources[2], place our config and develop your kernel module there. For testing you can insert it into our remote VM and debug via **printk**. For a more sophisticated setup, you may build the patched QEMU and build your own VM.
- However, if you want to reproduce our remote setup, we ship an appropriate **Dockerfile**

[2] `https://cdn.kernel.org/pub/linux/kernel/v6.x/linux-6.1.74.tar.xz`

Q & A

We are now happy to answer your questions :)