WLKOM

Wild Linux Kernel Object Module

Jules Aubert

2025

Groups

The project is made for groups from 1 to 4 students

You can mix CYB and DEV students (and you should for reasons I explain later)

You can mix groups A/B/C students

Deadline

You have about two and a half months

No one-third supplement time for concerned students because of timing reasons, sorry :(

[REDACTED]

[REDACTED]

EpiRootkit

You have to design a rootkit

What is a rootkit?

Malware that allows an attack program to maintain access to a remote victim machine

Scenario

Pirate Captain Ching Shih wants to attack the EFREI fleet (Echoués en Formation Rongés d'Echecs et Incompétences)

She organizes a fleet of pirates leds by Captains Jacques Moineaux, Whisky Bill, Gin Jane, René Bojolais and Ronda Rhum

She calls upon the APPING class to achieve this

The target is the EFREI accountant, Gérard Delacompta

Scenario

Ching Shih coordinates the entire fleet

Jacques Moineaux is in charge of finding the place where Gérard Delacompta will be most vulnerable

Whisky Bill must find an electronic device

Gin Jane must code a script that runs automatically

René Bojolais must find an admin program with a vulnerability

Ronda Rhum must use this vulnerability to load code into the kernel

The APPING class is in charge of the rootkit

Scenario

Gérard Delacompta finds a flash drive on the port of its company

He inserts it into his professional computer

A payload executes scripts to get admin access through privilege escalation

The now admin payload inserts code into the Windows kernel to stay executed on boot with kernel access on kernel land (and not userland)

An attacking program installed on a controlled environment connects with the rootkit

The pirates **Command** and **Control** (C2) Gérard's professional computer

Don't be like Gérard

Be like the pirates

What you don’t have to do

Prepare a trapped flash drive and scripts to escalate privileges

Code for Windows Kernel

Make a pirates fleet (although I wouldn’t mind)

What you have to do

- Documentation
- Code

Environment

Your environment is the EPITA's laptop with Ubuntu 24.10

But not only

You will have to generate 2 virtual machines

Documentation

A full documentation to

- Create and configure the attacking virtual machine
- Create and configure the victim virtual machine
- Deploy and use the attacking program
- Deploy and use the rootkit

Imagine Claire Leroux has to follow your documentation to build, deploy and use everything

(Use printscreens ;))

Tech

You are free to make me install **free** and **open source** softwares

For example for the hypervisors, you can use QEMU(/KVM), Proxmox or VirtualBox

For the virtual machines, you must use a Linux distro, but you can use whathever distro you want as long as the Linux kernel is at version 4.0.0 minimum

If you're crazy enough you can build your own distros for the attacking virtual machine and even compile your very own kernel

Attacking program

The attacking program can use whatever tech you want

It can be a CLI tool, a web UI or a GUI

Rootkit

The rootkit must be a kernel module, so it's pure C

C’est Clair(e) ?

Remember, Claire Leroux can’t guess how to install, deploy and use your tools

You must document everything for her to be the best hacker in the world

The features

You will have to code some features for your rootkit

Compile

First of all… it would be great for your rootkit to compile and be loadable into the kernel

Connection

Once loaded, your rootkit must contact the attacking program to **alert** its installation

I want a visual alert about the state of the connection: connected or disconnect

Persistence

When you simply insert a kernel object into the kernel, it disappears after a reboot

Make it persistent

Executing commands

From your attacking program, be able to execute programs and display the **stdout**, **stderr**, **exit status** on the attacking program

Password

You don’t want anyone to be able to use your rootkit

Add a password to begin to interact with it

Don’t write the password hardcoded into the rootkit, cypher it

Upload and download

I want commands to upload file from the attacking machine to the victim machine and download files from the victim machine to the attacking machine

Cardboard box

You may need to add configuration into system files. It means that the sysadmin can discover them and read your added lines

Make it possible to hide data from targeted files if something from the userland tries to open and read it

Also, you may need a directory to put your stuff for the rootkit, hide it from the userland if someone lists the directory where your rootkit directory files is

And finally, can you make your module disappears from the modules list?

Crypto

Did you know that whatever you were doing on the network was in plaintext all along?

Encrypt your connection so that network admins can't read the data through the logs

Bonuses

Bonuses can help you having a better grade if you don't want to make mandatory features

You can ask me if the bonus you want to make is interesting enough to replace an mandatory featured

If you do everything, congrats you have 20/20

If you do everything with bonuses, congrats you have 20/20

Everything

You are expected to document **EVERYTHING**

The choice of the hypervisor, the choice of the distro, the choice of the version of the distro, the choice of the kernel version (why not a more recent version that the one you're proposing?)

How to install the hypervisors on Ubuntu 24.10

How to create the virtual machines inside the hypervisor, how to configure them

How to download the distros, how to install them

How to configure the distros, how install the packages you need and configure them

Where do I put the files? How to deploy your tools? How to use them?

Everywhere

I'm not in your head, I don't want to guess looking at the code. I am your client. It's ok if the tool is not super easy to use (but make it easy to use nonetheless), but I want at least a full documentation I can refer to to understand **EVERYTHING**

All at Once

I will work on an Ubuntu 24.10, the same you have on your EPITA's laptop, but you're free to use whatever tech you want, as long as everything is free to use and open source[1]

[1] open standard for PDF :)

Assemble

To create a group, it's easy

1. Write a **new** mail to jules**1**.aubert@epita.fr
2. Put your teammates in carbon copy
3. The mail name must be [SYS2][WLKOM] YourTeamName
4. Write Hello, rewrite your team name and list the **logins** of your mates
5. Kindly regards, signature, you know the drill
6. Click Send

Of course, write your team name instead of YourTeamName... unless you want YourTeamName as a team name (first arrived, first served)

You have until the end of week to make the teams, after that... randomizator.py

It's only pedagogical

This is a **pedagogical** rootkit

You earn points doing the documentation and features. If a team is using a recent kernel version, it will earn the same amount of points than another team using an older kernel version. But, I'm expecting you to document why you used the version you're asking to use, with technical details, not just "Because it doesn't work, duh". You can even put web sources into your documentation to support your statements

Of course you should consider going for the most recent kernel version possible with the code you produce

It goes the same way for every choice. You made a fully working CLI tool and another team made a very comfy web UI? Everyone earn the same amount of points

On point

I give points for good documentation and working features.

I just want you to think about the possibilities and have fun doing it

Grades

About points, here is how I am going to give you points:

I told you I **REALLY** want documentation

I will test and grade each feature **if and only if** it is accompanied by documentation

Once again

You are expected to document **EVERYTHING**

?

Questions?