CY-4973/7790

Kernel Security: how2rootkit

$ whoami

- Kai
- Security Researcher
- Lecturer + PHD candidate @ NEU
    - I have two emails and I check neither of them
    - Please ping me on discord if you send me an email.
- Ch0nky LTD: AKA why teaching kids to write malware is actually a good idea!

Course Overview

Abstract

This advanced course explores Linux kernel security through common rootkit techniques, focusing on the ARM64 architecture. Students will gain hands-on experience by building a fully-featured rootkit targeting a bespoke Linux environment. The course emphasizes both offensive and defensive perspectives: students implement rootkit techniques to understand attack surface, then develop detection and mitigation strategies.

TLDR

- Learn about the Linux Kernel
- Focus on Armv8-a
- "But what happens when an attacker gets `root`?"

Benefits of Learning to Write Malware

- The first thing I do when learning about a new system/architecture is write a poc implant targeting it.
- usually this means writing a simple loader/beacon
- Loading code, and communicating to a remote Command and Control (C2) Server
- Trojan horsed way of teaching Systems security :)

Getting Started:

- Esoteric knowledge of the target
- Computer Systems/Operating Systems
- Computer networking
- linking/loading

Why get into Infosec

- It’s fun :D.
- There is no shortage of interesting work.
- The impact you can have is massive
- **very** generous compensation.
- Deeper understanding of the types of technology that power the modern world
- You’ll become really fun at parties.

Goals of the Course

- Learn basic Linux internals and `Armv8-a` development
- Create a safe environment to allow students to explore malware
- Create clear paths for students to enter the security community
- Help make everyone here more security literate.
- Get your hands dirty reversing and writing malware
- You read that right. In this class you will be writing malware

Goals of the Course

- Learn basic Linux internals and `Armv8-a` development
- Create a safe environment to allow students to explore malware
- Create clear paths for students to enter the security community
- Help make everyone here more security literate.
- Get your hands dirty reversing and writing malware
- You read that right. In this class you will be writing malware

MalwareCourse/Capstone

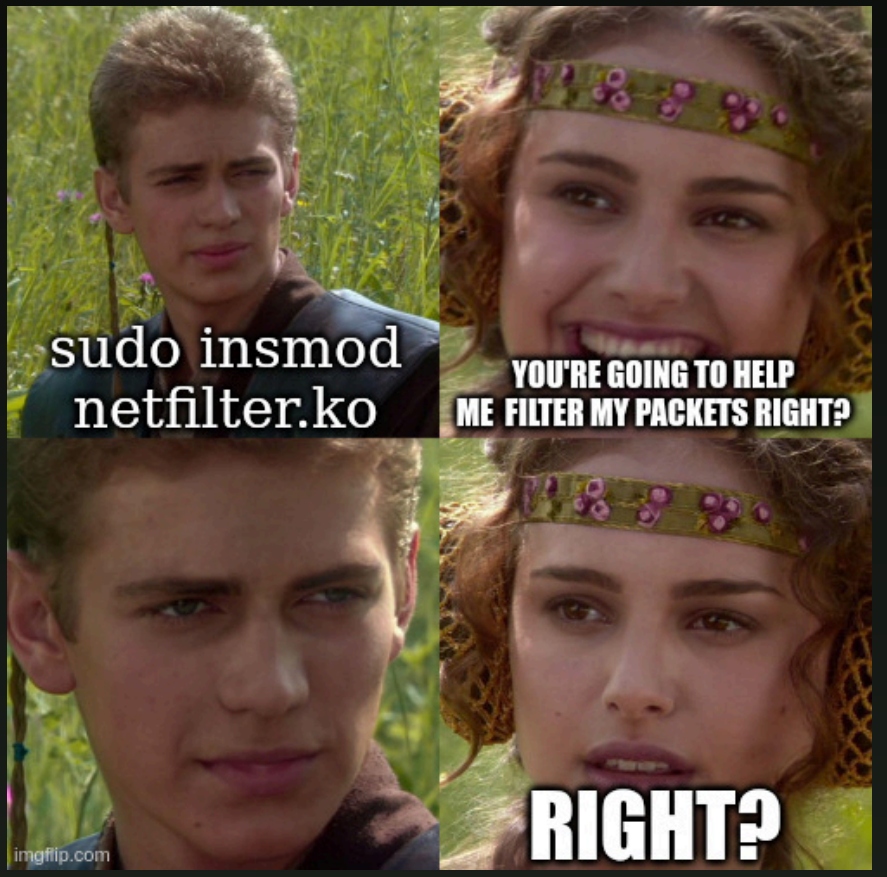
- The majority of your grade comes from the capstone
- The capstone is **really** hard and is too difficult to complete on your own
- There are two Projects that you will implement over the course of this semester:
    - LD_PRELOAD rootkit (userland)
    - LKM Kernel Mode Rootkit
        - Grad students will receive an extra challenge

**** Infosec is a Team sport!****

- Go make friends with your neighbor

Capstone Progression

- The course is all about demistifying the Linux kernel
- To that end, lectures will be focused on specific features of the kernel
    - This includes the userland and kernel perspective



This is a new field. All concepts learned in this class are directly applicable to industry.

Course Success Stories

For students who demonstrate passion for the subject, I will personally leverage my network to help you find a job

Other incentives include:

- Really cool Ghost stickers
- Free trip to DefCon
- Introduction to folks from industry
- Jobs lined up for high performers in the course

Syllabus#Tentative Topic list

- TLDR: C, Armv8-A, Linux (mostly userland) Internals, Ghidra, implant dev/rev
- This is the **first** time the course is focused on root scenarios
    - There are going to be hiccups, and I am writing a large part of the course material on the fly!
- Depending on how things go, we may cover more or less than what is listed on the syllabus.
- Since this is designed to be current, topics might shift according to current events.

Why should you not take this class this year

- its new :)

Why should you not take this class this year

- This class is not easy. Far from it.
- This class is taught exclusively in c.
- c will happily let you blow your foot off
- Iteration on kernel programming is time consuming
- You will almost certainly spend hours debugging code
- The capstone project is **very hard** and is difficult to debug
- **I say this not to scare you, but to be clear: you need a certain amount of academic maturity to succeed in this course! More on difficulty later...**

Baseline Course Policies

Policies#Attendance

- Attendance is mandatory. Unless cleared with me, you may not asynchronously take the course.
- While I record lectures, they are often released 2-4 days after they are recorded.
- I do not take attendance, but I do give regular quizzes at the start of class.
- If you need to miss class for any reason, please contact me
- If I have any reason to believe you are not physically in Boston for the semester, that Is a problem

Policies#Communication

- We have a dedicated Discord server and you are expected to join. Please see the course docs for getting started.
- I will also respond to email, but will be slower there (at least 24-48 hours)
- If you have a personal question about course administration, job stuff, or anything else that you don't think your fellow classmates can benefit from, you may DM
- If someone else can benefit from the question, please ask it publicly.
- If you need help, Ask

Academic Honesty

- TLDR: Don’t cheat. Thnx
- Cheating and plagiarism of any kind will not be tolerated.
- Cheating robs you of the experience of writing your own code, and in turn is a waste of your time and money.
- I will not tolerate anything that compromises your ability to learn in this class.
- You’re all adults, and know the drill.

Use of ChatGPT/LLMs

- 😄 You are welcome to use LLMs to help explain documentation.
- 😄 You can ask LLMs to help you find bugs in your code or to diagnose compiler errors.
- 😄 You can (try) asking for inspiration for your final project.
- 😄 You may use LLMs to help write documentation, unit tests and comments
- 🚫 You may **not** copy and paste generated code and submit it as your own.
- If I catch you submitting generated code, you are going to have a bad time.
- I can't stop you from pasting assembly/decompiled code and pasting it into Chat GPT (i do this during CTFs all the time!) but am going to advise against it. If you don't learn the basics, you are going to **suffer greatly** on something LLMs can't help with

Academic Honesty (cont)

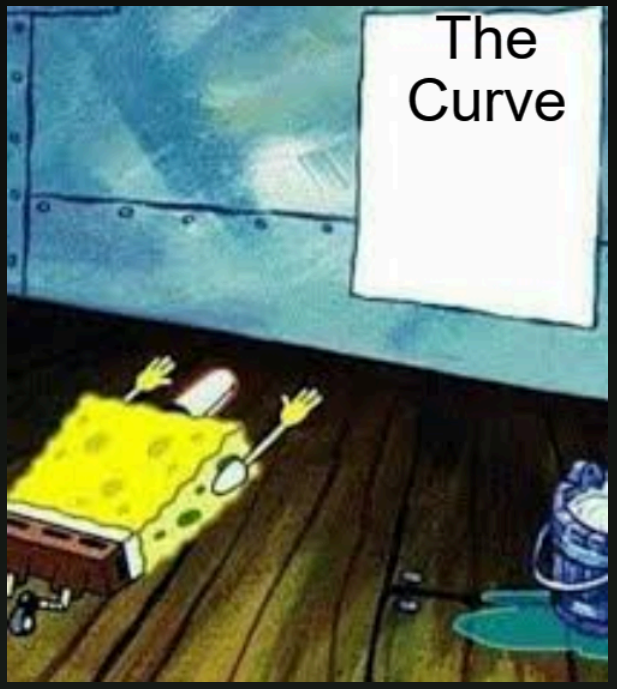
- If you submit code that is identical to another classmates, you will fail.
- If you collaborate when it is expressly forbidden, you will fail.
- If you copy and paste code you don't understand in the real world, you are incurring a lot of personal and institutional risk.
- If you would like to work in a group for homework assignments you can where permitted, but you can’t both submit the exact same code. You also need to clearly identify who you worked with and where your sources came from.

Academic Honesty (cont)

- I am not a cop.
- You are paying a lot of money to be here.
- It is my job to make sure you get your monies worth
- It is your job to give this class your all

Grading Policies

- ** This is not a weed out course. **
- You get what you put in, use the time to learn and try new things.
- There is a curve, and the average is a B+ (A- for grad)
- If you are on the border of a letter grade, I will probably bump it.
- I do not curve grades down
- If everyone in the class crushes it, then everyone gets an A.
- In general, it is common to get the average, rare to fail, and difficult to get an A.



Difficulty

- This class is not easy
- It is easy to spend hours on problems that are simple in hindsight
- The capstone will be very time consuming and will require you to work in a group
- You may not work solo unless you're a Grad student.

> **Trying hard literally pays off!**

Reminder: many students have received job offers based on the work they did for the capstone!

Difficulty (cont)

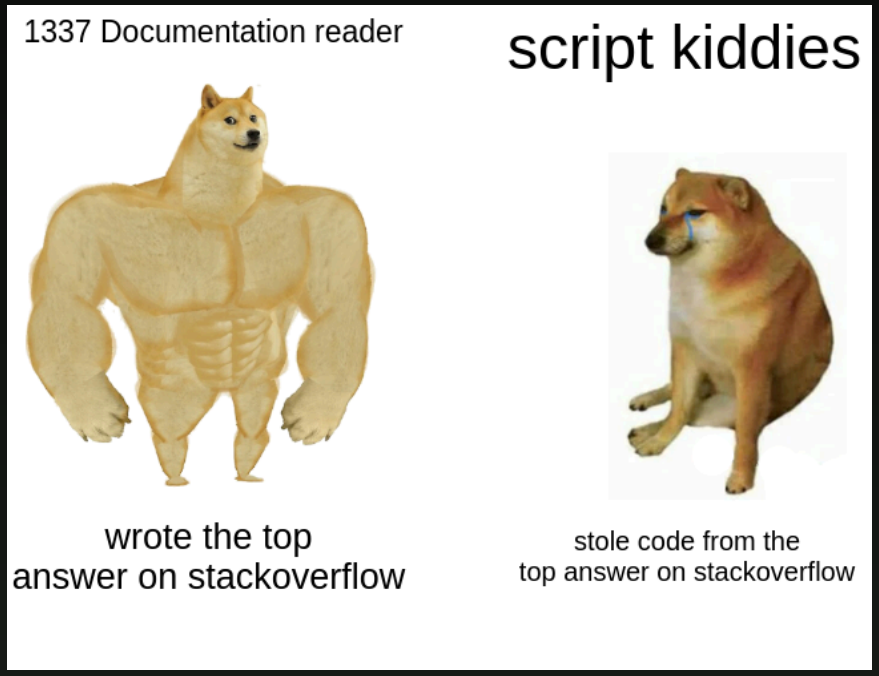
- Coding assignments will require you to read documentation
- Reading the documentation can feel tedious but is a valuable skill in and of itself.
- Being a hacker means caring about material that others overlooked/trusted implicitly
- As a hacker, you should enjoy getting into the weeds ** (Or at the very least not be afraid to dive in :)

Difficulty (cont)

<rant>

- Most of this content is simple in hindsight
- Simple != easy
- Getting comfortable being lost and working out what is happening is an essential skill for hacking

</rant>



The Linux Kernel is very complex

- The kernel is huge and layered: millions of lines of code across subsystems (memory, scheduler, VFS, networking, drivers).
- Fragmentation and configuration sprawl: countless versions, distro patches, CPU architectures, kernel configs, and out-of-tree modules mean “the kernel” is not one uniform target.
- Massive attack surface: every syscall, ioctl, netlink message, filesystem parser, and driver path can be a security boundary.
- Deep hardware knowledge: MMU/virtual memory, interrupts, DMA/IOMMU, speculation, privilege rings, firmware, and device semantics routinely shape real security outcomes. Each arch has its own hardware features that enable kernel featured

How are we going to learn Linux?

- Reading source code
- Tracing
- Reading more source code
- Debugging
- Reading more source code

Kernel Resources to start with

- https://makelinux.github.io/kernel/map/
- https://elixir.bootlin.com/linux/v6.18.4/source (pick the version)
- https://explorar.dev/torvalds/linux

Policies#No Jail

Policies#No Jail

- I AM NOT A LAWYER. (IANAL)
- The Computer Fraud and Abuse Act 1986: bans “ intentionally accessing a computer without authorization or in excess of authorization”.
- We will not protect you if you conduct operations or deploy any malware / tools outside your approved lab environment.
- We will not bail you out of jail

Policies#No Jail

- Do not piss off anybody who has more time and money than you. The laws are intentionally vague, and the
    - Judges are technically illiterate
    - You will lose and be made an example of.
- When in doubt, ask course staff but always error on the side of caution!

Seriously, Don’t Hack Without Permission

- Aside from it being illegal, immoral, creepy, and risky, you will fail the class.
- Plus, the payout most criminal hackers get is considerably less than if you do the same job but call it “adversary emulation” or run a threat intel company.
- Don't ruin it for everyone else.

Do not F*ck with the Government

- The current geopolitical climate is hot.
- Countries haven't regularly resorted to kinetic action in response to hacking, but that isn't a hard and fast rule. It is a norm that might not be respected tomorrow
- Do not, under any circumstances, hack a foreign or domestic government. They have all the time, all the money, and capabilities that you cannot hope to compete with. **You will lose** and **you will be caught**
- If you really want to do this kind of work, go get a clearance.
- ** *Kinetic*** is a euphemism for killing people.

Use caution when publishing tools

- Plenty of researchers publish offensive security tools
- Some advance the field, some are published for clout. Be careful with especially sophisticated, easy to use, and low detection tools.
- Our job is to advance the security industry, not make script kiddies look good.
- Think long and hard about the implications of submitting (for example) a pull request to metasploit or open sourcing your final project

Be Thoughtful

- Threat actors use open source tools.
- Don’t be the reason that a hospital gets hit with ransomware.
- If you do decide to publish tools, make sure to also include countermeasures that defenders can use to combat your tool.
- For questions about licenses, responsible disclosure, or advice on whether or not to publish a tool, please contact us! We are happy to provide feedback.

Policies#Grading

Policies#Late policy

- You are allocated 3 late days for the semester, and may use them at your discretion.
- You may ask for more late days for extenuating circumstances. I am pretty reasonable when it comes to providing extensions.
- Late days may only be used on reverse engineering assignments, or coding assignments.
- There are no late days for the capstone project or exercises.
- I do not accept submissions for exercises if you are remote.

Assignment Philosophy:

- The questions on homework assignments will increase in difficulty.
- Difficulty should be a spectrum
- Often times the solution is simple
- Again: Simple != easy
- You are rarely expected to complete the totality of an assignment

: Coding Assignments

- Coding assignments are auto graded.
- Unless otherwise stated, you may NOT work in groups assignments.
- You may only collaborate when it is explicitly stated.

Coding assignments are meant to be completed solo!

RE: Exercises

- Exercises are weekly, quick turn assignments (usually in person)
- They are designed to be sanity checks for topics as the course progresses
- Should take 5-10 minutes
- Questions will always be based on recent material, but topics build on each other
- Bring your laptop to class.

Grading Breakdown

Subject to change.

Single source of truth is in the Syllabus

- Homework: 50%
- Capstone Project: 50-100%

Bonus Points:

- Class Participation: 10%
- Guest lecture attendance 2.5%
- Vibes (20%+)
    - ???
    - How is this fair? It only helps you.
    - Computed after the final averages.
- I have no issue giving everyone an A.
    - (But you gotta earn it :-)

Remarks about Mental Health

- I give incompletes to students who have extenuating circumstances: usually medical emergencies.
- Mental health emergencies are medical emergencies.
- There will be some sections that discuss material that may be triggering to some.
- Warnings will be given over Discord and at the start of such lectures, and attendance is completely optional.

To Reiterate

- I give incompletes.
- If you say “I am having a medical emergency and need an incomplete” you will get one; zero questions asked.
- Note this means you get about 30-90 extra days to complete the outstanding work.

Technology

- Computer capable of running QEMU, and enough space for the images/snapshots
    - in practice, this is means about 50GB of free space, and for a good experience 16GB + of ram
- If you do not have a suitable computer, let me know
- Ideally, you have a computer capable of running a Linux VM

Course Administrative Technology

- Students are required to join the class Discord server
- As of now, all assignments are to be submitted through GradeScope
    - this may change as I explore more virtualization/autograding tech
- Course notes, slides and assignment directions will use Obsidian.md
- If you see a broken link in a github MD file, it is probably because you need to open it in Obsidian

Syllabus#PreReqs

- Comfort with using Linux. Shells, text editors, configuration ...etc
- Computing systems : familiarity with memory management, c programming, reading assembly, using debuggers and compiler toolchains.
- Operating systems
- Computer Networks: (TCP/IP/UDP/HTTP/TLS)

Syllabus#PreReqs

- GDB/experience debugging native binaries
- Experience with compiler toolchains (compilers, linkers)
- Academic maturity

Requirements: Programming/scripting

- Python 3: you need to be very proficient with python. If you don't know python, Java is also OK but you will need to learn python :-)
- c/c++/some other systems programming language: you should be able to write simple C programs, know the difference between Stack and Heap memory. I.e. how to manage memory
- intel x86 assembly: you should be able to read/write x86 assembly. (or some other assembly language)
- This class will be taught in ARMv8-a Arm64

Syllabus#Protocols

- IP: you should know the difference between a private and public IP, the basics of addressing, and know the relationship between a domain name, DNS and an IP address
- TCP: More so the basics of socket programming, less so the internals of how reliability is achieved or what the handshake process looks like.
- DNS: You don't need to understand **how** DNS works, but you need to know what it does.
- TLS: You need to understand what a Certificate Authority is, a TLS cert is, and the basics of establishing secure tunnels
- HTTP(s): HTTP verbs, request structure, and error codes

Requirements: Emulator +VM

- 16GB of RAM, a decent CPU, at least 100GB of storage
  - If your computer is low on space, you can install the VM in an external SSD connected via USB3. You can find these at microcenter for around $50-100
- Internet connection

Remarks about pre-requisite

- Is Infosec for me?
- Short answer: yes
- Long answer: also yes
- Do I have the right background for this class?
- Take a look at HW 0 (CPractice)
- If you are lost, it might be a good idea to take this course next year! You are welcome to stick it out but you might need to put in extra work.
- Due to the size of the class, it might be difficult to give individualized attention on all assignments

Course Livestream

The course will be streamed via either Discord

- As of now, it looks like I will use OBS + Discord
- Bear with me as I learn how to live stream :-)
    - In particular, expect some recordings to get corrupted or the audio quality to be sub par.

Office Hours

- Non-deterministic. By appointment only.
- Where is my office?
- Good question. I don’t know :D
    - For the time being, I will usually migrate to a coffee shop

Bonus/Guest Lectures

- Experts from industry, some of which are hiring, will periodically swing by to give a bonus lecture
- You should come if you can make it!
- There will often be Pizza
    - If you have a dietary restriction, let me know :-)

Questions?

Homework

- Join Discord
- Join Gradescope
- C practice problems
    - Released Monday, due Friday before class

Homework (Continued)

- The first assignment will be released on the course GitHub
- It will require a working Lab environment

More on the Capstone

- Two components:
  - 1. Userland Rootkit:
  - 2. Kernel Rootkit
- Grad Students:
  - Exploit via shellcode/ kernel R/W
  - More features

Capstone Flow

- Learn about Kernel API, feature, object ...etc
- Write userland code to interact with X
- Write Kernel Code to trace/inspect X
- Write Userland "capability"
- Write Kernel Capability

Discussion

- What is a Rootkit?