NEU CY 5770 Software Vulnerabilities and Security

Instructor: Dr. Ziming Zhao

First off, Logistics!

Classes are recorded and released publicly on YouTube
But you have to attend the class in-person

Have a notebook in front of you
Bring your own laptop

https://zzm7000.github.io/teaching/2024fallcy5770/index.html

We have an online CTF platform for this class.

Feel free to interrupt me and ask questions.

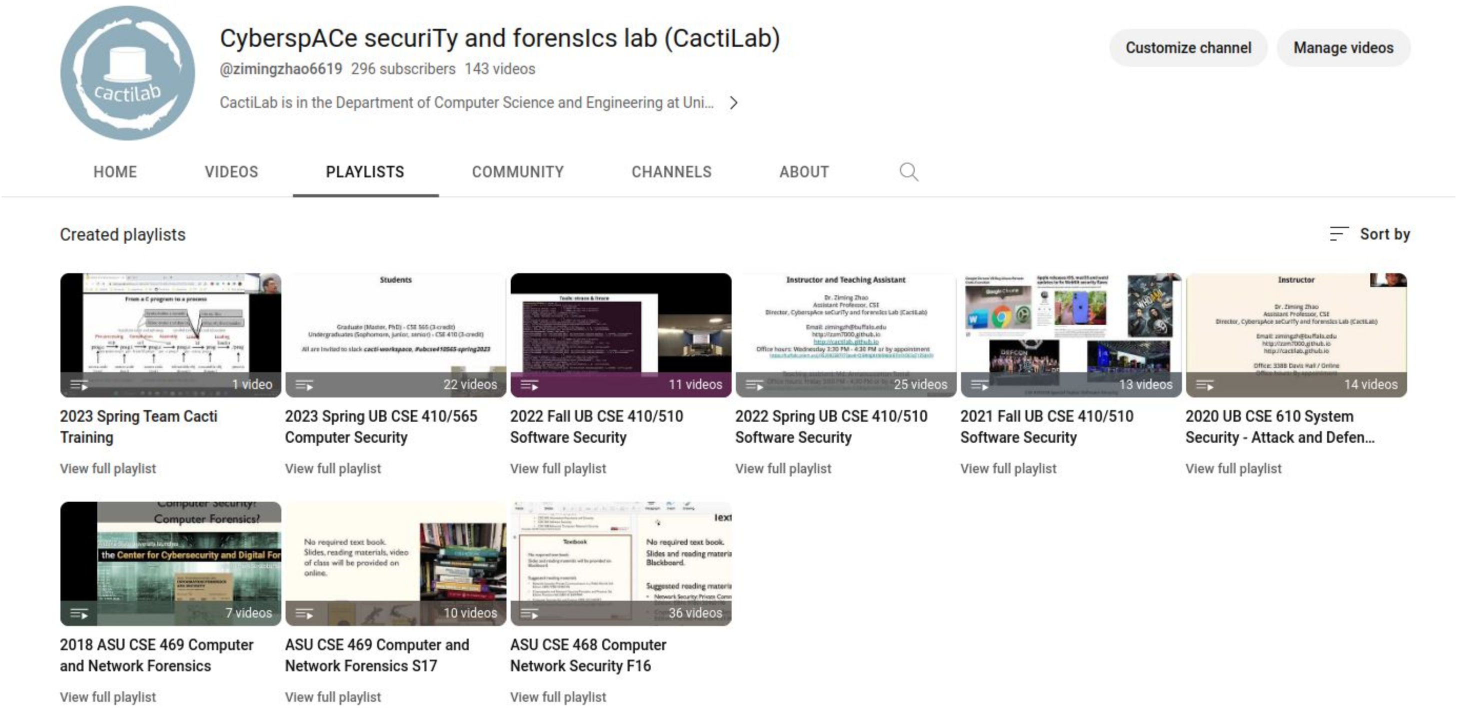
Instructor and Teaching Assistant

Dr. Ziming Zhao
Associate Professor, Khoury College of Computer Sciences
Director, CyberspAce seCuriTy and forensIcs Lab (CactiLab)

Email: z.zhao@northeastern.edu
http://zzm7000.github.io
http://cactilab.github.io
T 11:45 am - 1:25 pm, R 2:50 pm - 4:30 pm
Office hours will be T 1:45 pm - 2:45 pm or by appointment

https://northeastern.zoom.us/j/99475115019?pwd=4dMw5mmuLNHHh0LS9CCIl93xapoB4o.1

YouTube Channel



CyberspACe securiTy and forensIcs lab (CactiLab)

@zimingzhao6619 296 subscribers 143 videos

CactiLab is in the Department of Computer Science and Engineering at Uni...

Created playlists

Playlist	Videos
2023 Spring Team Cacti Training	1 video
2023 Spring UB CSE 410/565 Computer Security	22 videos
2022 Fall UB CSE 410/510 Software Security	11 videos
2022 Spring UB CSE 410/510 Software Security	25 videos
2021 Fall UB CSE 410/510 Software Security	13 videos
2020 UB CSE 610 System Security - Attack and Defen...	14 videos
2018 ASU CSE 469 Computer and Network Forensics	7 videos
ASU CSE 469 Computer and Network Forensics S17	10 videos
ASU CSE 468 Computer Network Security F16	36 videos

View full playlist

https://www.youtube.com/channel/UCkSeVUu-AxytXqalx66j7Eg/playlist

About CactiLab

Research areas:

- Systems and software security (Arm Cortex-M, Cortex-A, RISC-V, FPGA, GPU, etc.)
- Security in/with ML/DL/LLM
- Autonomous driving security
- Formally verify the security properties of crypto protocols and system code
- Hacking/CTF platforms

We need students at all levels for funded research, volunteer work, independent study, undergraduate research experience, etc.

Students

Graduate (Master, PhD) - CY 5770 (4-credit)
Undergraduates (junior, senior) - CY 5770 (4-credit)

All are invited to slack ***cacti-workspace***, ***#neucy5770-fall2024***

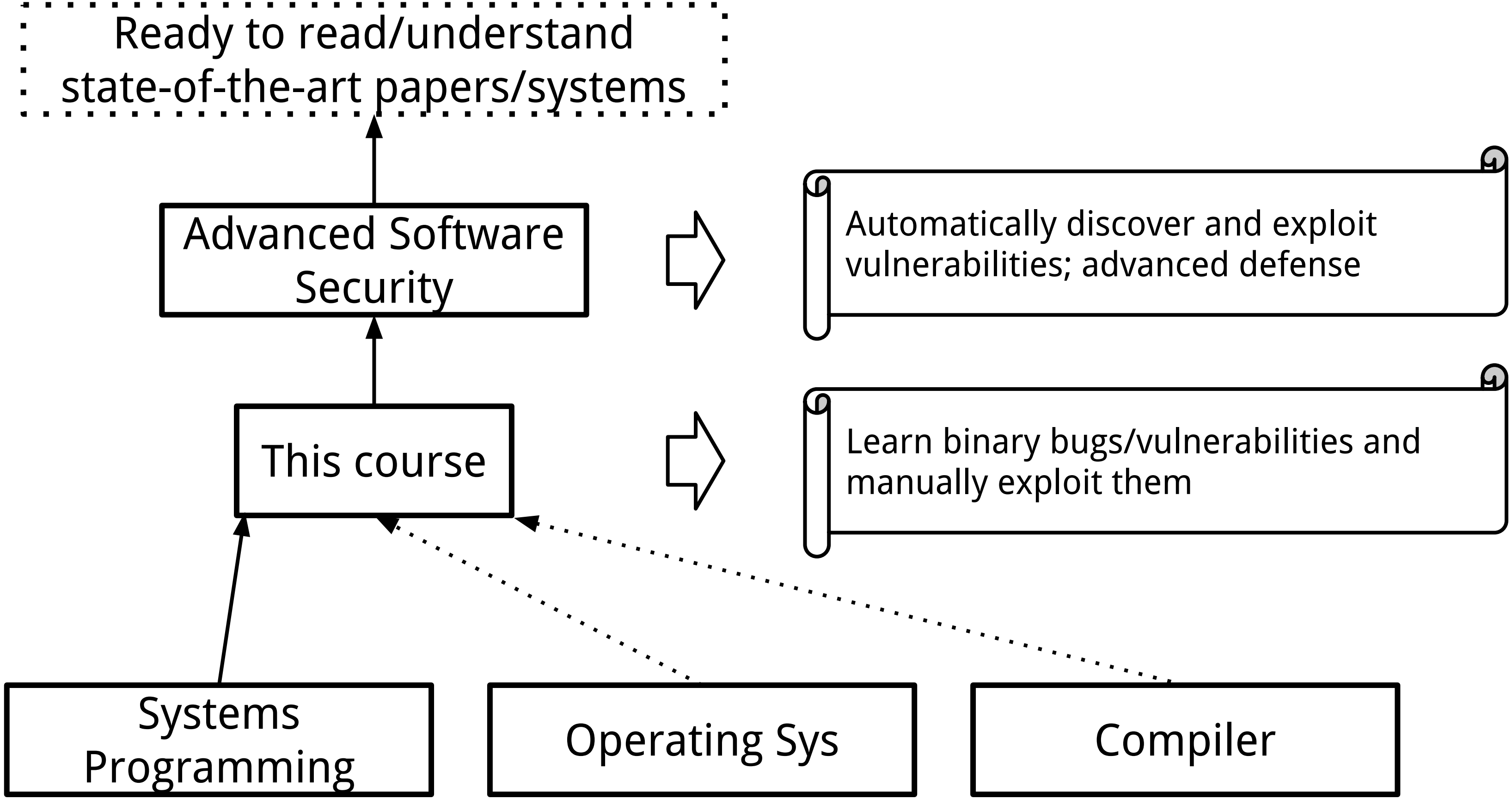
Course Goals

To provide you with good understanding of the **theories**, **principles**, **techniques** and **tools** used for binary software and system hacking and defense.

By software and system, I mean native software, binary, most likely developed in C/C++. The security of web software, Java, Python are out of the scope.

You will study, in-depth, binary reverse engineering, vulnerability classes, vulnerability analysis, exploit/shellcode development, defensive solutions, etc., to understand how to crack and protect **native** software. You will get your hands dirty.

If you want to be a systems/software security guy …



Ready to read/understand state-of-the-art papers/systems

Advanced Software Security

Automatically discover and exploit vulnerabilities; advanced defense

This course

Learn binary bugs/vulnerabilities and manually exploit them

Systems Programming

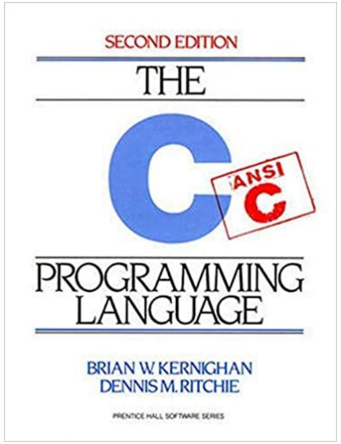
Operating Sys

Compiler

First week’s Agenda

1. Class overview and logistics
2. Background knowledge
   a. Compiler, linker, loader
   b. x86 and x86-64 architectures and ISA
   c. Linux file permissions
   d. Set-UID programs
   e. Memory map of a Linux process
   f. System calls
   g. Environment and Shell variables
   h. Basic reverse engineering

Prerequisites



The real prerequisite:
The C Programming Language

Classes that will help you understand this class:
Systems Programming

Operating Systems

Other skills:

Reverse engineering (Using objdump, IDA Pro, Ghidra, etc.)

Debugging (GDB, pwngdb)

Google, reading, self-learning, getting hands dirty

Topics

Binary attack and defense using x86 and x86-64 as examples. Discover **vulnerabilities**. Develop **exploits**. Memory corruption attacks.

1. Stack-based buffer overflow
2. Defenses against stack-based buffer overflow
3. Shellcode development
4. Format string vulnerabilities
5. Heap-based buffer overflow
6. Integer overflow
7. Return-oriented programming
8. …

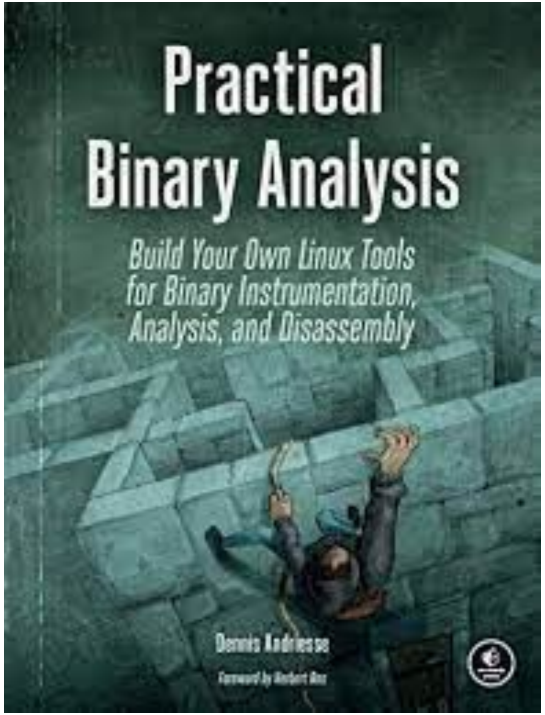
Related Books and Papers

SoK: Eternal War in Memory. IEEE S&P 2013

SoK: (State of) The Art of War: Offensive Techniques in Binary Analysis. IEEE S&P 2016

SoK: Shining Light on Shadow Stacks. IEEE S&P 2019

Practical Binary Analysis: Build Your Own Linux Tools for Binary Instrumentation, Analysis, and Disassembly


Practical
Binary Analysis
Build Your Own Linux Tools
for Binary Instrumentation,
Analysis, and Disassembly
Dennis Andriesse
Foreword by Herbert Bos

Related Books and Papers

SoK: Eternal War in Memory. IEEE S&P 2013

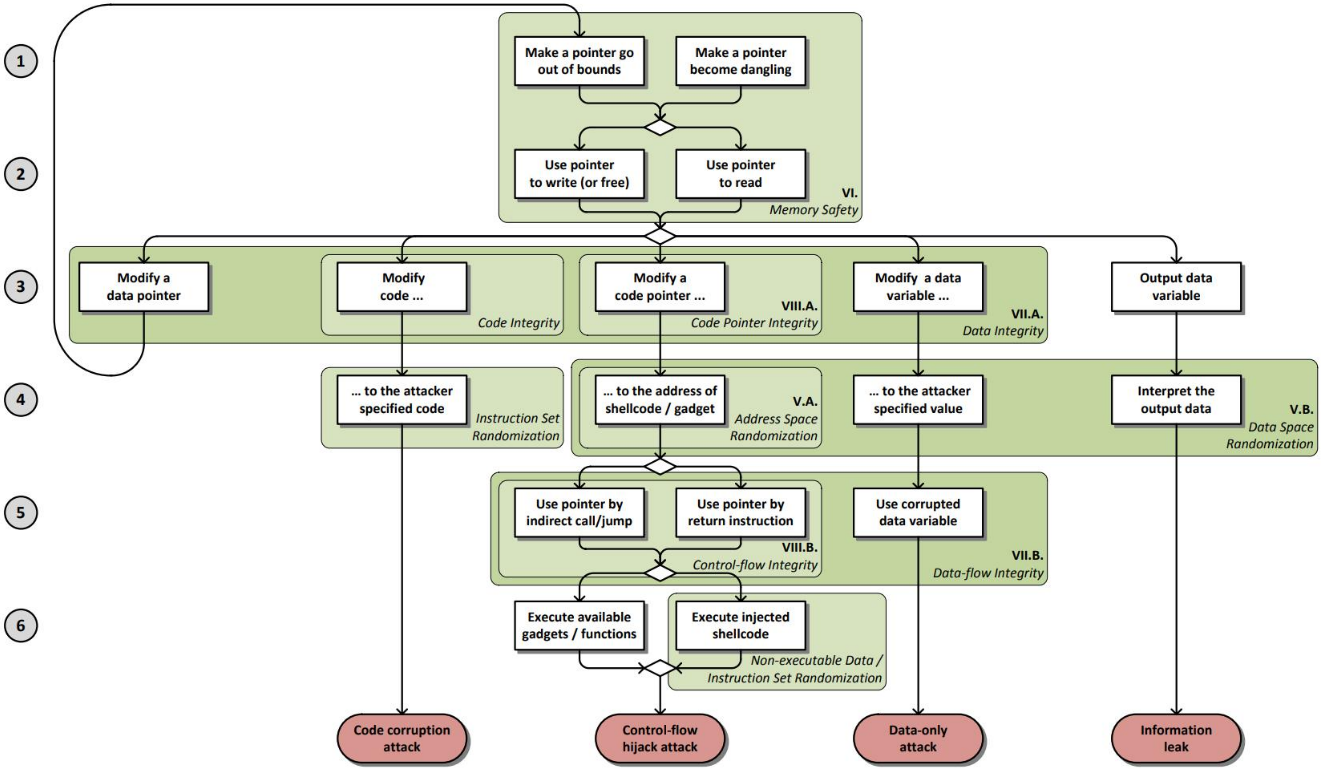

1
2
3
4
5
6
Make a pointer go out of bounds
Make a pointer become dangling
Use pointer to write (or free)
Use pointer to read
VI.
Memory Safety
Modify a data pointer
Modify code ...
Code Integrity
Modify a code pointer ...
VIII.A.
Code Pointer Integrity
Modify a data variable ...
VII.A.
Data Integrity
Output data variable
... to the attacker specified code
Instruction Set Randomization
... to the address of shellcode / gadget
V.A.
Address Space Randomization
... to the attacker specified value
Interpret the output data
V.B.
Data Space Randomization
Use pointer by indirect call/jump
Use pointer by return instruction
VIII.B.
Control-flow Integrity
Use corrupted data variable
VII.B.
Data-flow Integrity
Execute available gadgets / functions
Execute injected shellcode
Non-executable Data / Instruction Set Randomization
Code corruption attack
Control-flow hijack attack
Data-only attack
Information leak

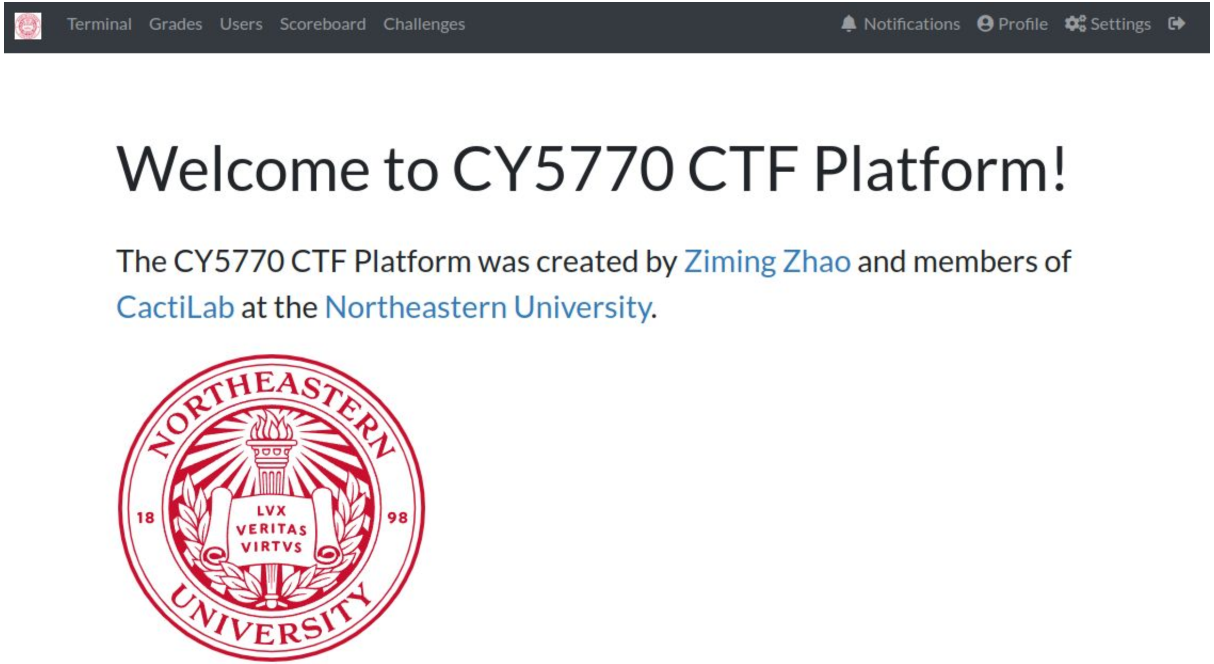
Figure 1. Attack model demonstrating four exploit types and policies mitigating the attacks in different stages

The Hacking Environment

http://cy5770-cacti.khoury.northeastern.edu/

Only NEU students can access this website. If you are off-campus, you need to VPN to connect to NEU network to access

Register an account with your NEU username and email address, so we know who you are.



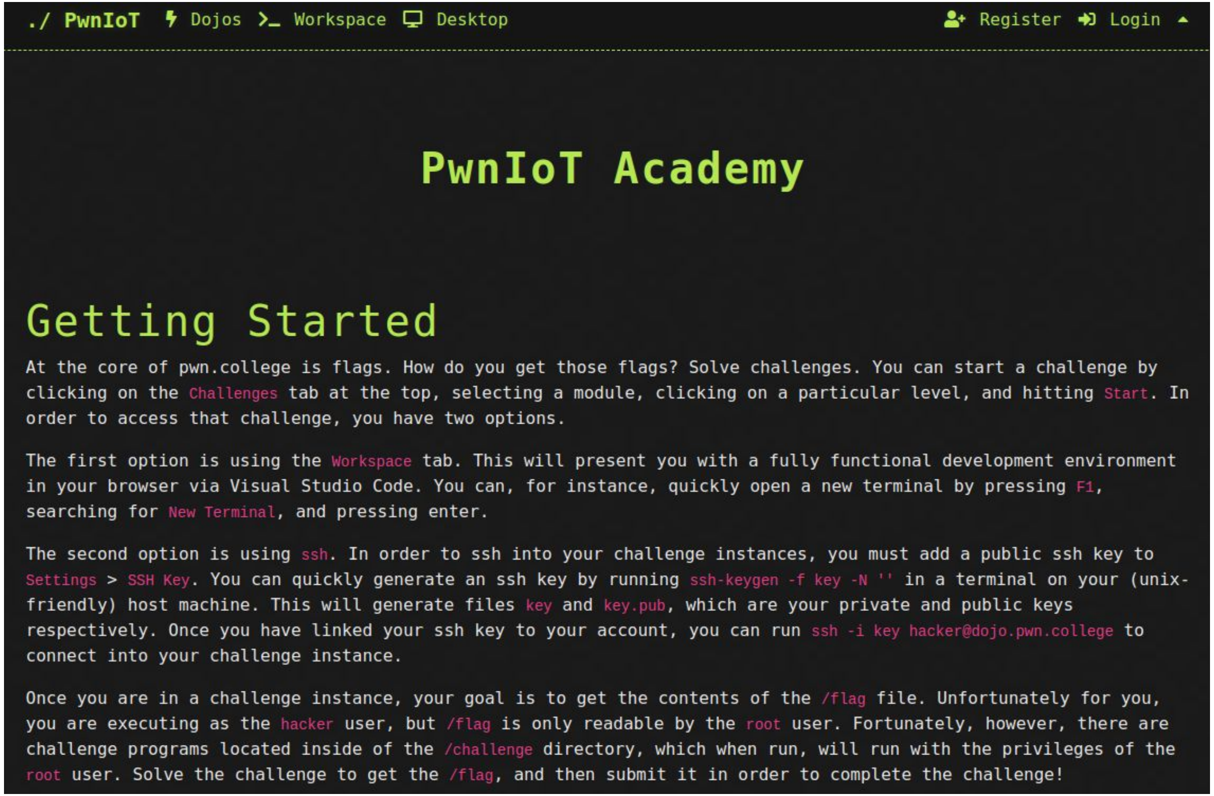
Welcome to CY5770 CTF Platform!

The CY5770 CTF Platform was created by Ziming Zhao and members of CactiLab at the Northeastern University.

New Environment Under Construction

Only NEU students can access this website. If you are off-campus, you need to VPN to connect to NEU network to access

Register an account with your NEU username and email address.



PwnIoT Academy

Getting Started

At the core of pwn.college is flags. How do you get those flags? Solve challenges. You can start a challenge by clicking on the `Challenges` tab at the top, selecting a module, clicking on a particular level, and hitting `Start`. In order to access that challenge, you have two options.

The first option is using the `Workspace` tab. This will present you with a fully functional development environment in your browser via Visual Studio Code. You can, for instance, quickly open a new terminal by pressing `F1`, searching for `New Terminal`, and pressing enter.

The second option is using `ssh`. In order to ssh into your challenge instances, you must add a public ssh key to `Settings` > `SSH Key`. You can quickly generate an ssh key by running `ssh-keygen -f key -N ''` in a terminal on your (unix-friendly) host machine. This will generate files `key` and `key.pub`, which are your private and public keys respectively. Once you have linked your ssh key to your account, you can run `ssh -i key hacker@dojo.pwn.college` to connect into your challenge instance.

Once you are in a challenge instance, your goal is to get the contents of the `/flag` file. Unfortunately for you, you are executing as the `hacker` user, but `/flag` is only readable by the `root` user. Fortunately, however, there are challenge programs located inside of the `/challenge` directory, which when run, will run with the privileges of the `root` user. Solve the challenge to get the `/flag`, and then submit it in order to complete the challenge!

The Hacking Environment

Intel x86
x86-64, a.k.a amd64
ARM Cortex-A, Cortex-M
Linux (Ubuntu)

Pwngdb
Pwntools
GDB peda
NSA Ghidra
Binary Ninja

Homework

Reading: book chapter, whitepaper, paper, blog, etc.
Hands-on: hacking, debugging, etc.

Submit before a class on Canvas. We may discuss homework at the beginning of each class.

30% penalty if you submit within 10 mins after class starts. 0 points after 10 mins.

0 points for homework if plagiarising is found. No exceptions.

Disability Access Services

If you need DAS, please inform me in the first two weeks.

Hacking Assignment Rules

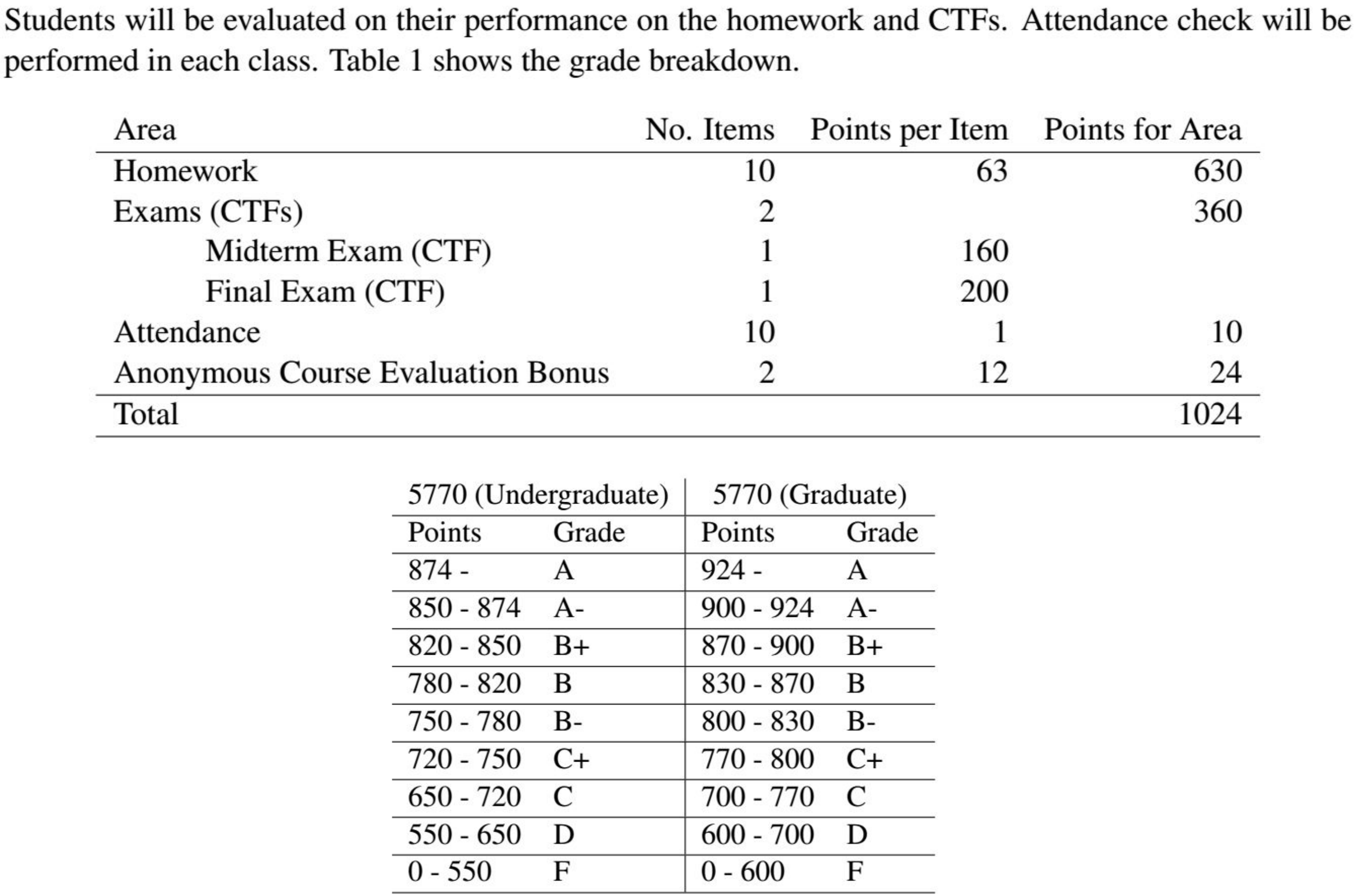
- For each hacking assignment, you will submit your exploit, a simple write-up, and screenshots to show it works
    - Simple write-up:
        - Briefly describe how you solve the challenge
        - Mention who you worked with if any in the write-up

- Discussion is encouraged. But, you cannot share your code, exploits, write-ups to your classmates or post them online.

Exams, a.k.a, Capture-the-Flag (CTF) Hacking

Midterm CTF: 3 hours and 20 minutes
Final CTF: 3 hours and 20 minutes

Grades



Students will be evaluated on their performance on the homework and CTFs. Attendance check will be performed in each class. Table 1 shows the grade breakdown.

Area	No. Items	Points per Item	Points for Area
Homework	10	63	630
Exams (CTFs)	2		360
Midterm Exam (CTF)	1	160	
Final Exam (CTF)	1	200	
Attendance	10	1	10
Anonymous Course Evaluation Bonus	2	12	24
Total			1024

5770 (Undergraduate)		5770 (Graduate)	
Points	Grade	Points	Grade
874 -	A	924 -	A
850 - 874	A-	900 - 924	A-
820 - 850	B+	870 - 900	B+
780 - 820	B	830 - 870	B
750 - 780	B-	800 - 830	B-
720 - 750	C+	770 - 800	C+
650 - 720	C	700 - 770	C
550 - 650	D	600 - 700	D
0 - 550	F	0 - 600	F

Academic Integrity

Your first assignment is to to read the NEU academic integrity policies

Here are examples for your consideration

- you work on your laptop at a library with friends and step away from your computer without locking it
- you look at your neighbors' screen/papers during an exam, but don't copy their answers
- you take a piece of code from some website and give a link to the website at the end of the homework
- you work on a homework problem with friends, type the solution at home, but it's exactly the same as that of your friends

Academic Integrity

- Discussion is encourage. But, you cannot share your code, exploits to your classmates or post them online.

- The university, college, and department policies against academic dishonesty will be strictly enforced. To understand your responsibilities as a student read: UB Student Code of Conduct.

- Plagiarism or any form of cheating in homework, assignments, labs, or exams is subject to serious academic penalty.

- Any violation of the academic integrity policy will result in a 0 on the homework, lab or assignment, and even an **F** or **>F<** on the final grade. And, the violation will be reported to the Dean’s office.

ChatGPT/LLM Policy

- ChatGPT/LLM is forbidden in the midterm and final CTFs

Ethical Hacking

- Do not attempt to violate the law.
- If you discover real-world vulnerabilities using the knowledge you learn from this class, report the vulnerabilities responsibly. Companies may reward you for that.