

# **CS44500 COMPUTER SECURITY**

---

**Fall 2024**

# Syllabus

- Office hours

**Name:** Haytham Idriss (he/him)

**Office Hours:** ETCS 125A TR: 1:30 – 2:30 p.m.

**Email:** hidriss@pfw.edu

**Phone:** (260) 481 – 6817

*\*Reach out by email or phone to set up a meeting outside office hours\**

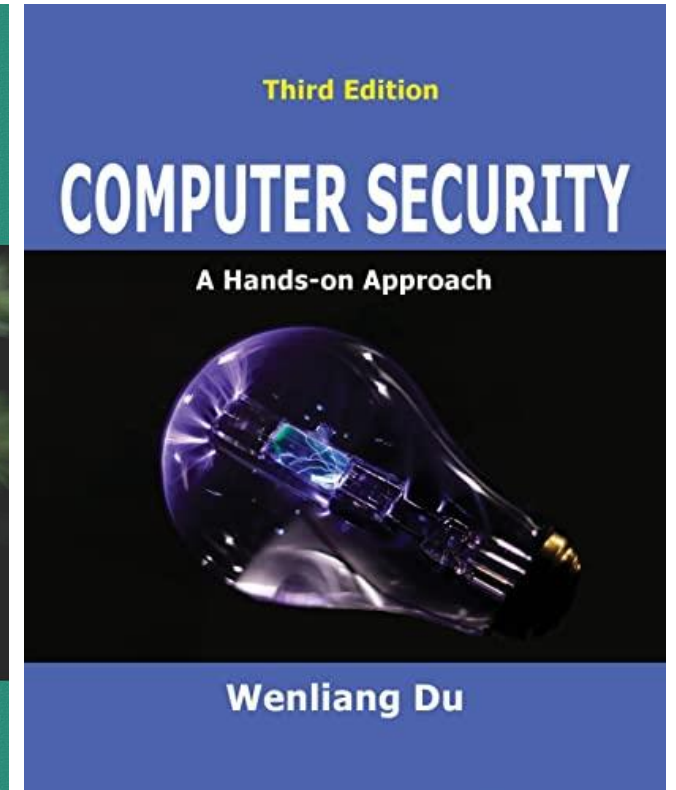
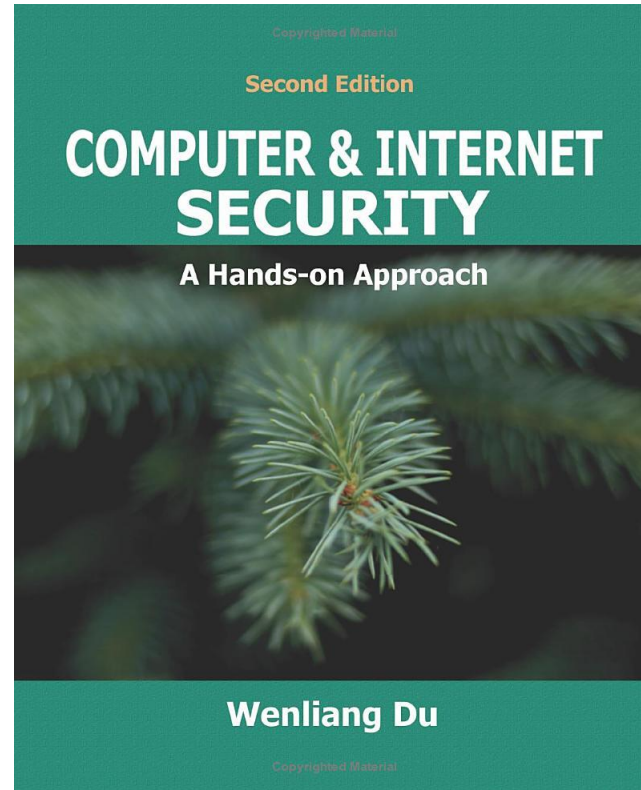
**Classroom & Time:** SB-G24, TR 3:00 p.m. – 4:15 p.m.

# Syllabus

- Office hours
- Email
  - Include CS 445 in subject.

# Syllabus

- Office hours
- Email
  - Include CS 445 in the subject.
- Textbook



*Computer & Internet Security: A Hands-on Approach* by Wenliang Du. Independently Published (2022).  
Third Edition (ISBN: 978-17330039-4-0).

<https://www.handsonsecurity.net/>

[https://seedsecuritylabs.org/Labs\\_20.04/](https://seedsecuritylabs.org/Labs_20.04/)

# Grades

- Assignments (10) (49%)
- Midterm (20%)
- Final (25%)
- Attendance (6%)

For credit on a given assignment, it must be submitted by the due date and time. Late submission without advanced approval by the instructor will be penalized as follows:

- 1 day late: 5% deduction
- 2 days late: 10% deduction
- 3 days late: 15% deduction
- **Submissions not accepted after 3 day**

# Make-ups and incompletes

## **Make-ups and incompletes**

Make-ups and incompletes will be given only in extreme circumstances. To schedule a make-up exam, you must contact either the instructor or the Department office prior to the date and time of the exam. There will be no make-up Quizzes. The instructor reserves the right to either allow a make-up with a penalty, a make-up without penalty, or deny a make-up as it relates to the circumstances and the promptness of notice. A student should provide documentation for the make-up. In other circumstances, the Final Examinations policy described in PFW Undergraduate Catalog will be applied.

# Course Objective

- Understand the principle of least privilege
- Understand **software security** attacks through environment variables
- Understand the stack memory layout and the attacks that exploit the stack, as well as their countermeasures
- Understand the **Web security** attacks and countermeasures
- Understand the **network security** attacks and countermeasures
- Understand the basic **cryptography**

# Software Security

- Set-UID programs
- Environment variables and attacks
- Buffer overflow attacks
- Return-to-libc attacks

Cover chapters 2 – 5 in textbook



# Web Security

- Cross Site Request Forgery
- Cross-Site Scripting Attack
- SQL Injection Attack
- Shellshock attacks
- Reverse Shell

Cover chapters 12 – 14, 16, and 10 in textbook

# Network Security

- Network Security Basics\*
  - Packet sniffing and spoofing
- Attacks on the TCP protocol\*

*\*Computer & Internet Security*

Cover chapters 19 – 20 in textbook

# Cryptography

- Secret-Key Encryption
- Public-Key Cryptography

Cover chapters 24 and 26 in textbook

# How to Win a Satisfactory Grade

- **Get** the textbook ASAP
- **Attend** the class
- **Understand** slides and class examples
- **Work independently** on labs
- **Form** study group
- **Ask** me
- **Don't** wait until the last minute to work on labs!

# Lab Environment Setup

- Virtual machine software
  - VirtualBox
- Pre-built virtual machine images
  - Ubuntu 20.04
- <https://seedsecuritylabs.org/labsetup.html>
- SEED Labs
  - [https://seedsecuritylabs.org/Labs\\_20.04/](https://seedsecuritylabs.org/Labs_20.04/)
- ET-109 and ET-111 labs

# CHAPTER 1: LINUX SECURITY BASICS

---

Users and Groups

# Users

- In Linux, each user is assigned a unique user ID
- User ID is stored in `/etc/passwd`

```
root:x:0:0:root:/root:/bin/bash
seed:x:1000:1000:SEED,,,:/home/seed:/bin/bash
```

- Find user ID

```
seed@VM:~$ id
uid=1000(seed) gid=1000(seed) groups=1000(seed)

root@VM:~# id
uid=0(root) gid=0(root) groups=0(root)
```

# Add Users & Switch to Other Users

- Add users
  - Directly add to /etc/passwd
  - Use “adduser” command
- Switch to another user

```
seed@VM:~$ su bob  
Password:  
bob@VM:/home/seed$
```



# Group

- Represent a group of users
- Assigning permissions based on group
- A user can belong to multiple groups
- A user's primary group is in /etc/passwd

```
root:x:0:0:root:/root:/bin/bash
seed:x:1000:1000:SEED,,,:/home/seed:/bin/bash
bob:x:1001:1001:Bob,,,:/home/bob:/bin/bash
alice:x:1002:1003:Alice,,,:/home/alice:/bin/bash
```

# Which Group Does a User Belong To?

```
seed@VM:~$ grep seed /etc/group
adm:x:4:syslog,seed
sudo:x:27:seed
plugdev:x:46:seed
lpadmin:x:120:seed
lxd:x:131:seed
seed:x:1000:
docker:x:136:seed
```

```
seed@VM:~$ groups
seed adm sudo plugdev lpadmin lxd docker
```

```
seed@VM:~$ id
uid=1000(seed) gid=1000(seed) groups=1000(seed), 4(adm), 27(sudo),
46(plugdev), 120(lpadmin), 131(lxd), 136(docker)
```

# Group Management

## How to add users

```
$ sudo groupadd alpha          # create a group alpha
$ sudo usermod -a -G alpha seed # add seed to alpha
$ sudo usermod -a -G alpha bob  # add bob to alpha
```

# LINUX SECURITY BASICS

---

**permissions and access control**

# Traditional Permission Model

- Types of access on files
  - **read (r)**: user can view the contents of the file
  - **write (w)**: user can change the contents of the file
  - **execute (x)**: user can execute or run the file if it is a program or script
- Types of access on directories
  - **read (r)**: user can list the contents of the directory (e.g., using `ls`)
  - **write (w)**: user can create files and sub-directories inside the directory
  - **execute (x)**: user can enter that directory (e.g., using `cd`)

# File Permissions

permissions  
- rwX rwX rwX seed abc 1802 Feb 6 11:39 xyz  
owner group other owner group owner file name

# Default File Permissions

- umask value: decides the default permissions for new files
- Example

Initial	(0666)	rw-	rw-	rw-
		110	110	110
umask	(0022)	000	010	010
-----				
Final permission		110	100	100
		rw-	r--	r--

Initial AND NOT umask

# Examples (umask)

```
$ umask                                umask (short for user file-creation mode mask)
0002
$ touch t1

$ umask 0022
$ touch t2
$ umask 0777
$ touch t3

$ ls -l t*
-rw-rw-r-- 1 seed seed 0 Feb  6 16:23 t1
-rw-r--r-- 1 seed seed 0 Feb  6 16:24 t2
----- 1 seed seed 0 Feb  6 16:24 t3
```



# Access Control List

- Fine grained ACL
- Assign permissions to individual users/groups
- Coexist with the traditional permission model
- Example

```
$ getfacl example
# file: example
# owner: seed
# group: seed
user::rw-
group::rw-
other::r--
```

# ACL Commands

```
setfacl {-m, -x} {u, g}:<name>:[r, w, x] <file, directory>
```

```
$ setfacl -m u:alice:r-- example
$ setfacl -m g:faculty:rw- example
$ getfacl example
# file: example
# owner: seed
# group: seed
user::rw-
user:alice:r--
group::rw-
group:faculty:rw-
mask::rw-
other::r--
```

①

```
-rw-rw-r--+ 1 seed seed 1050 Feb 7 10:57 example
  ↙ indicating that ACLs are defined
```

# LINUX SECURITY BASICS

---

## **Running Command With Privilege**

# Running command with privilege

- Three command mechanisms
  - sudo
  - Set-uid programs (covered in a separate chapter)
  - POSIX capabilities

# Using sudo

- sudo: Super-user Do
- Run commands as a superuser
- A user must be authorized (/etc/sudoers)
- Here is how the seed user is allowed to run sudo

```
In /etc/sudoers
%sudo ALL=(ALL:ALL) ALL
    ↖ group name

In /etc/group
sudo:x:27:seed ← the sudo group
```

# Getting Root Shell

- In Ubuntu 20.04, the root user account is locked
- Cannot log into the root account
- There are many ways to get a root shell
  - `sudo -s`
  - `sudo bash`
  - `sudo su`
- It is not recommended to run commands using a root shell. Instead, use `sudo` to run individual commands.

# Running Command Using Another User

- Run command using another user (instead of root, default)

```
$ sudo -u bob id  
uid=1001(bob) gid=1001(bob) groups=1001(bob),1004(alpha)
```

# POSIX Capabilities

- Divide the root privilege into smaller privilege units
- Known as capabilities
- Use “man capabilities” to find all the capabilities
- Examples

```
CAP_CHOWN:           Make arbitrary changes to file UIDs and GIDs.  
CAP_DAC_OVERRIDE:    Bypass file read/write/execute permission checks.  
CAP_DAC_READ_SEARCH: Bypass file read permission checks ...  
CAP_NET_RAW:         Use RAW and PACKET sockets ...
```



# Setting File Capabilities (1)

- Before

```
$ cp /bin/bash ./mybash
$ ./mybash
$ cat < /etc/shadow
mybash: /etc/shadow: Permission denied    ← Failed
```

- Setting the capabilities

```
$ sudo setcap CAP_DAC_READ_SEARCH=ep mybash
$ ./mybash
$ getpcaps $$      # List the capability of the current process
65331: = cap_dac_read_search+ep    ← The process has the capability
```

# Setting File Capabilities (2)

- After

```
$ sudo setcap CAP_DAC_READ_SEARCH=ep mybash  
$ ./mybash
```

```
$ cat < /etc/shadow      # Bash will open this file for read  
root:!:18590:0:99999:7:::  
daemon*:18474:0:99999:7:::  
bin*:18474:0:99999:7:::  
sys*:18474:0:99999:7:::  
...
```

```
$ cat > /zzzz           # Bash will open this file for write  
mybash: /zzzz: Permission denied
```

# Authentication Methods

- A process to verify a user's identity
- Typical authentication methods
  - based on something the **user knows**: password
  - based on something the **user has**: ID card
  - based on something the **user is or does**: fingerprint
- Multi-factor authentication

# The Password File

- Each entry contains a user account information
- Password is not stored here (used to be)

```
root:x:0:0:root:/root:/bin/bash
seed:x:1000:1000:SEED,,,:/home/seed:/bin/bash
bob:x:1001:1001:Bob,,,:/home/bob:/bin/bash
alice:x:1002:1003:Alice,,,:/home/alice:/bin/bash
```

# First Command After Login

- The last field of each entry

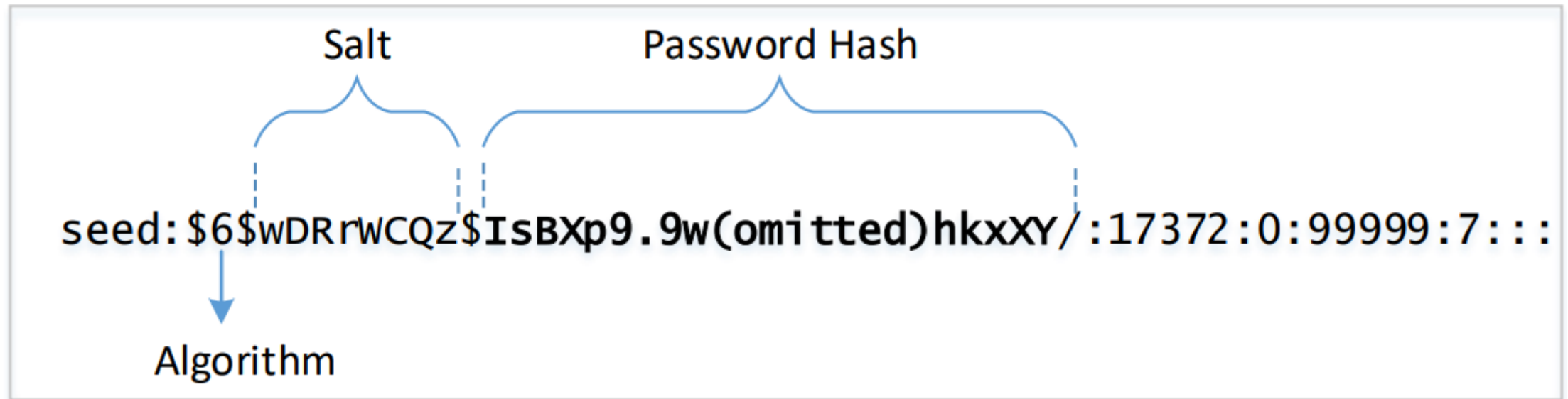
```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
gdm:x:125:130:Gnome Display Manager:/var/lib/gdm3:/bin/false
seed:x:1000:1000:SEED,,,:/home/seed:/bin/bash
bob:x:1001:1001:Bob,,,:/home/bob:/bin/bash
alice:x:1002:1003:Alice,,,:/home/alice:/bin/bash
```

```
$ sudo su bin
```

```
This account is currently not available.
```

# The Shadow File

- Store password, why not use /etc/password anymore?
- Structure for each entry



# The Purpose of Salt

- Defeat brute-force attacks
  - dictionary attack, rainbow table attack
- These 3 accounts have the same password

```
seed:$6$n8DimvsbIgU0OxbD$YZ0h1EA... (omitted) ...wFd0:18590:0:  
alice:$6$.1CMCeSFZd8/8QZl$QhfhId... (omitted) ...Sga.:18664:0:  
bob:$6$NOLhqomO3yNwyFsZ$K.Ql/KnP... (omitted) ...b8v.:18664:0:
```

SHA-512



# Locking Account

- Putting an invalid value in the password field
- The root account is locked

```
root:!:18590:0:99999:7:::
```