SIGPwny

General FA2025 • 2025-09-18

Reverse Engineering Setup

Cameron Asher

Cameron Asher

- President
- Statistics & Computer Science
- I got my discord profile picture commissioned for free in 2018. It's based on the default youtube pfps.

Announcements

- **Fall CTF 2025**
  - Intro hacking competition run by SIGPwny
  - September 21st, 12–6 PM
  - Visit https://2025.fallctf.com for more information and to register

- CSAW CTF 2025
  - We qualified for CSAW CTF Finals 2025!
  - Thank you to anyone who participated.

ctf.sigpwny.com

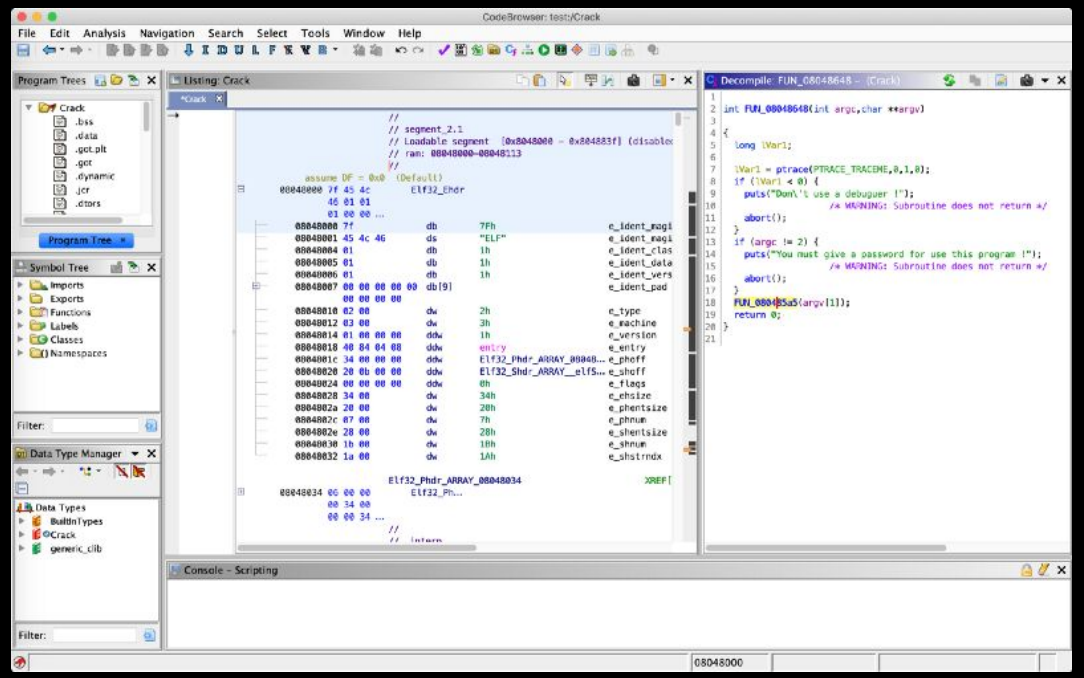
sigpwny{fallctf_is_on_9/21}

Reverse Engineering Setup

Survey of the room

What is Ghidra?

- Ghidra is a reverse engineering toolkit developed by the NSA and made open source
- Allows you to disassemble applications - essentially turn an unreadable application into readable code



Installing Java (Windows/macOS)

Check if you have Java JDK, and if so what version; should be ≥21

```
~ ❯ java -version
openjdk version "20.0.1" 2023-04-18
OpenJDK Runtime Environment Homebrew (build 20.0.1)
OpenJDK 64-Bit Server VM Homebrew (build 20.0.1, mixed mode, sharing)
~ ❯
```

Note: we recommend installing JDK and Ghidra on Windows, not WSL

Installing Java (Windows/macOS)

Install JDK 21+ (not JRE!) from Oracle (or package manager, if applicable)

https://www.oracle.com/java/technologies/downloads/#java25

Java 25, Java 21, and earlier versions available now

JDK 25 is the latest *Long-Term Support (LTS)* release of the Java SE Platform.

JDK 21 is the previous Long-Term Support (LTS) release of the Java SE Platform.

Earlier JDK versions are available below.

Learn about Java SE Subscription

JDK 25 JDK 21

Java SE Development Kit 21.0.8 downloads

JDK 21 binaries are free to use in production and free to redistribute, at no cost, under the Oracle No-Fee Terms and Conditions (NFTC).

JDK 21 will receive updates under the NFTC, until September 2026, a year after the release of the next LTS. Subsequent JDK 21 updates will be licensed under the Java SE OTN License (OTN) and production use beyond the limited free grants of the OTN license will require a fee.

Linux macOS Windows

Product/file description	File size	Download
ARM64 Compressed Archive	186.07 MB	https://download.oracle.com/java/21/latest/jdk-21_linux-aarch64_bin.tar.gz (sha256)
ARM64 RPM Package	185.76 MB	https://download.oracle.com/java/21/latest/jdk-21_linux-aarch64_bin.rpm (sha256) (OL 9 GPG Key)
x64 Compressed Archive	187.89 MB	https://download.oracle.com/java/21/latest/jdk-21_linux-x64_bin.tar.gz (sha256)
x64 Debian Package	159.73 MB	https://download.oracle.com/java/21/latest/jdk-21_linux-x64_bin.deb (sha256)
x64 RPM Package	187.55 MB	https://download.oracle.com/java/21/latest/jdk-21_linux-x64_bin.rpm (sha256) (OL 9 GPG Key)

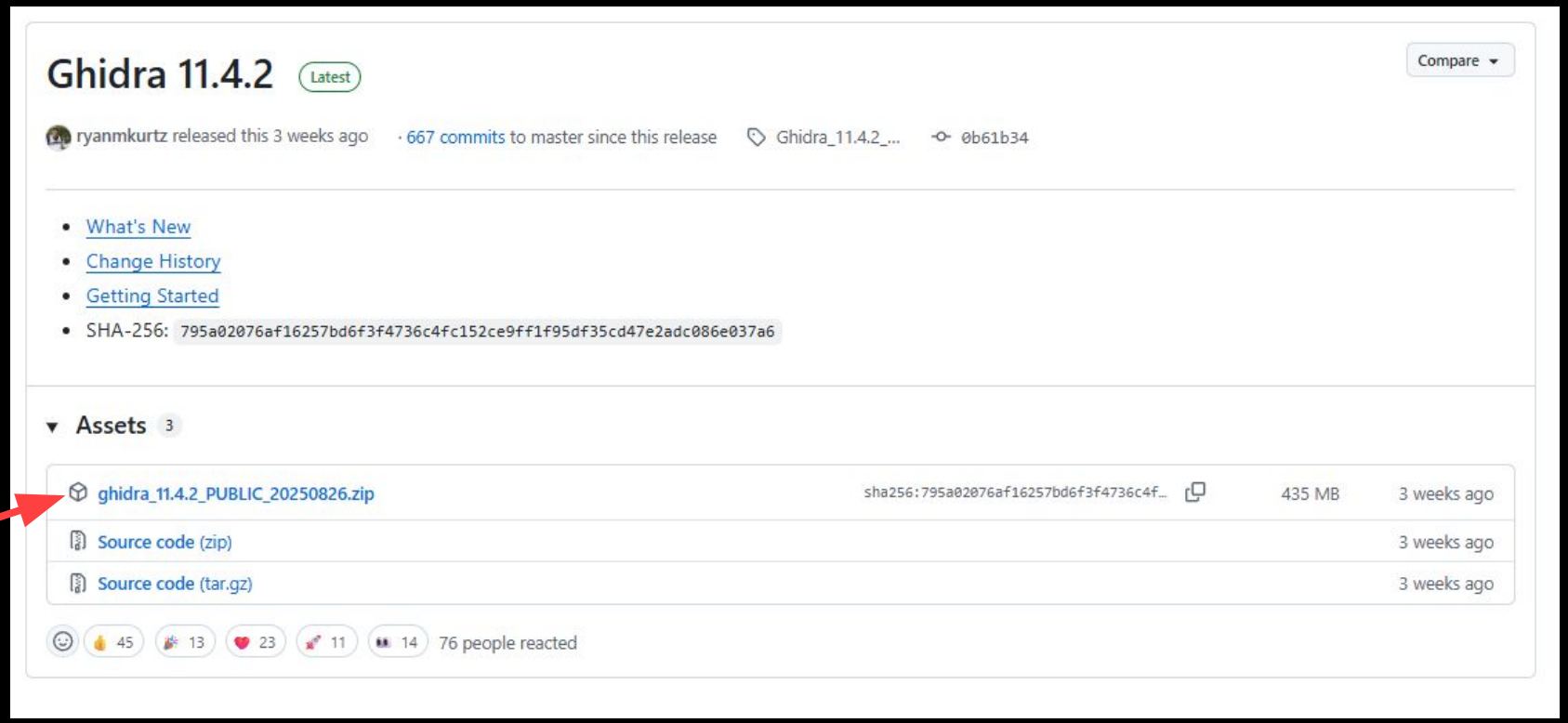
Installing JDK (Linux)

```
sudo apt update
sudo apt install openjdk-21-jdk
```

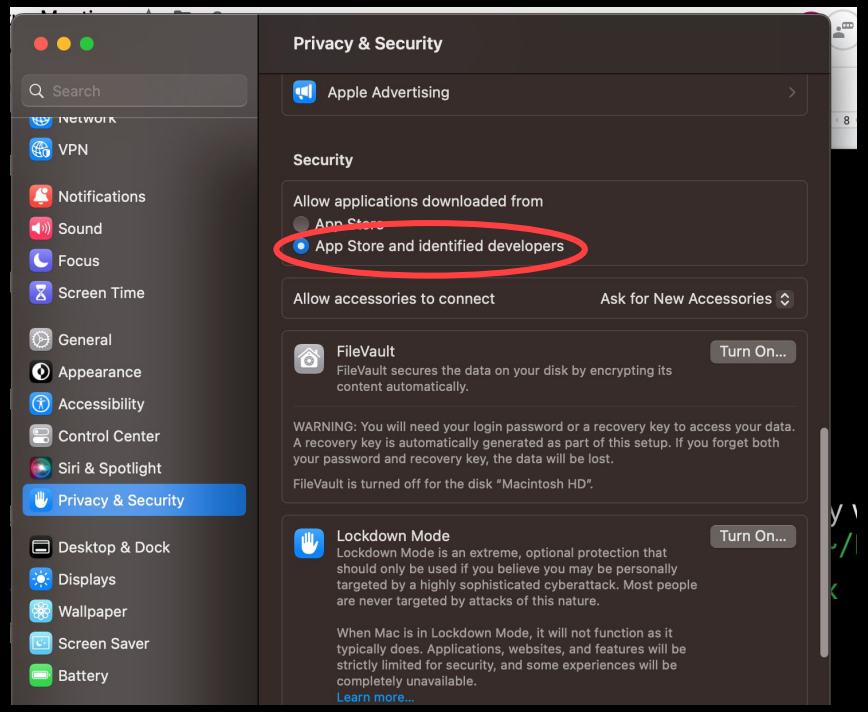
Downloading Ghidra (All Platforms)

https://github.com/NationalSecurityAgency/ghidra/releases

Download the public archive in assets for the latest release (`ghidra_X.X.X_PUBLIC_XXXXXXXX.zip`, not `Source code.zip`)



Running Ghidra



Windows:

Double click `ghidraRun.bat`

Mac/Linux:

```
$ cd ~/Downloads
$ unzip ghidra_??.?.?_PUBLIC_*.zip && cd ghidra_??.?.?_PUBLIC
$ chmod +x ghidraRun && ./ghidraRun
```

Running Ghidra (macOS)

The Ghidra distributable on GitHub is unsigned and needs permission to run the decompiler binaries

1. Open an x86 binary and run through the default decompiler
2. When you receive an error, go back to the "Privacy & Security" tab of settings, and hit "allow" on the binary that appears there
3. Repeat until you receive no errors when decompiling

See https://support.apple.com/kb/PH25088?locale=en_US for more clear instructions if you're having trouble.

OR run this one-liner to remove Ghidra from "quarantine":

```
sudo xattr -d -r com.apple.quarantine $GHIDRA_ROOT
```

$GHIDRA_ROOT - where you downloaded ghidra to

Python and pwntools

"Now is better than never." (*The Zen of Python*, aphorism 15)

What is pwntools?

pwntools is a CTF framework and exploit development library written in Python

It makes scripting exploits much simpler/less tedious

```
>>> sh = process('/bin/sh')
>>> sh.sendline(b'sleep 3; echo hello world;')
>>> sh.recvline(timeout=1)
b''
>>> sh.recvline(timeout=5)
b'hello world\n'
>>> sh.close()
```

Installing Python

pyenv allows you to easily manage and switch between different Python versions (e.g. 3.12 and 3.8)

This is **preferred** over a system installation of Python - but both will work fine!

```
$ curl https://pyenv.run | bash
- add the EXPORT ... snippet in output to the
  end of your ~/.bashrc OR ~/.zshrc
$ pyenv install 3.11
$ source ~/.bashrc / source ~/.zshrc
$ pyenv global 3.11
```

Installing pwntools

```
python3 -m pip install pwntools
```

If you get a "command not found", you may need to refresh the shell environment:

```
source ~/.bashrc
source ~/.zshrc # zsh is default on macOS
```

on Apple silicon (M1, etc.) run this first!

```
$ brew install cmake pkg-config qemu
```

GDB + pwndbg

For those times where `printf` doesn't cut it

Computer Architectures

M-series Macbook

```
60      ;IF-THEN   WITH COM
61      ;      IF (R0 <= 20 || R
62      MOV    R0, #-2
63      CMP    R0,#20
64      BLE    S_THEN
65      CMP    R0,#25
66      BLT    S_ENDIF
67 S_THEN MOV    R1,#1
68 S_ENDIF
```

aarch64 / arm64
“arm, 64 bit”

You cannot run x86 programs normally* on arm64, or vice versa!

i9-morbillion laptop

```
_start:

    mov   edx, len
    mov   ecx, msg
    mov   ebx, 1
    mov   eax, 4
    int   0x80

    mov   eax, 1
    int   0x80
```

x86 / x86_64
“x86, 64 bit”

*We will talk about an exception on Macs called Rosetta

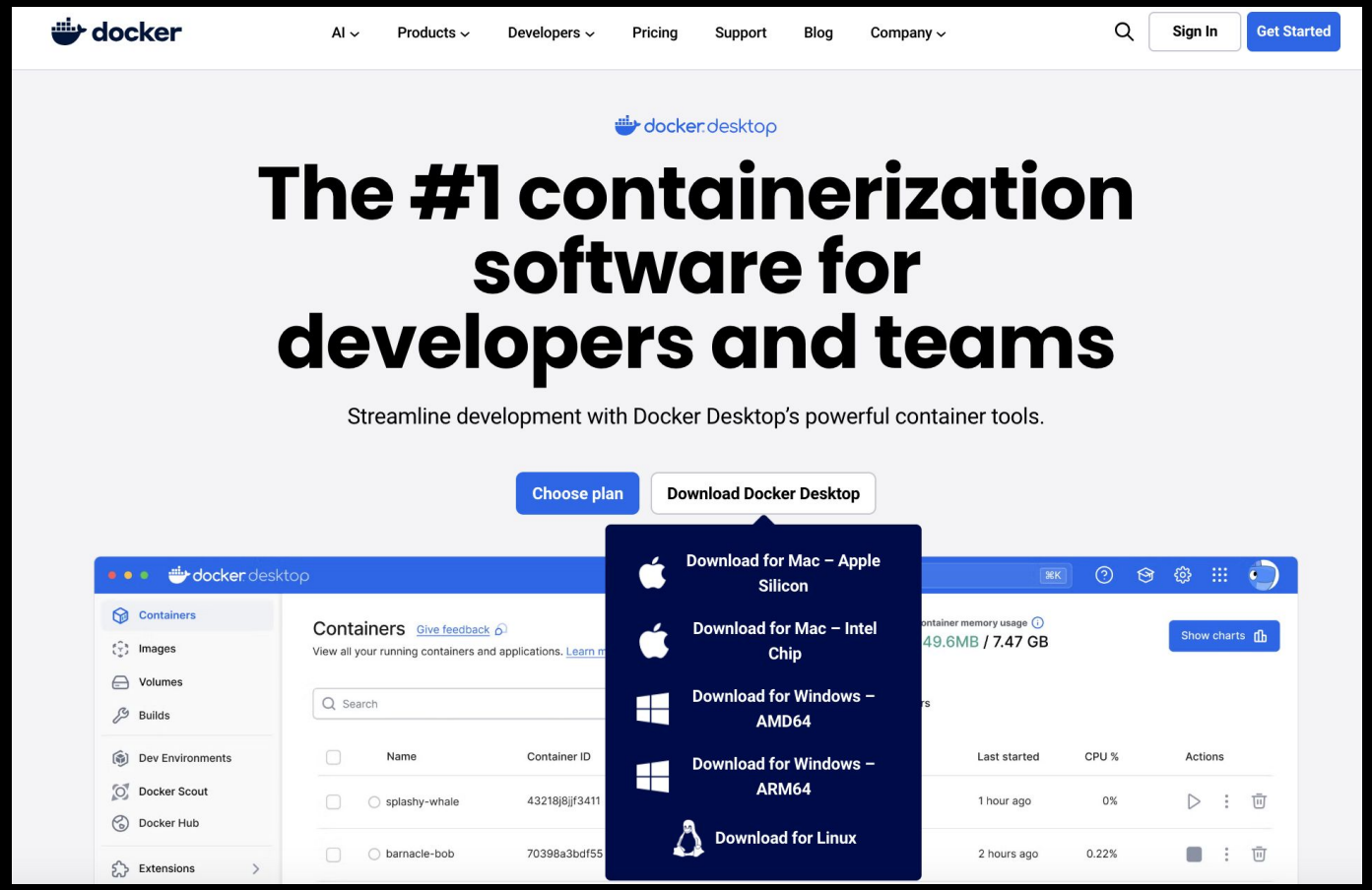
**Otherwise, you can use QEMU

What do I have?

- Windows
  - You are almost certainly running an Intel x86_64 cpu
- Mac
  - You are probably running on an ARM cpu

Installing Docker

- Install Docker Desktop at https://www.docker.com/products/docker-desktop/



What is GDB?

- The **G**NU **D**e**B**ugger allows you to inspect and modify execution of programs
- We will teach you how to debug **x86** binaries in **Rev II: x86 Reversing**!
- **pwndbg** is a "plugin" (`gdbinit`) for GDB that adds lots of nice features that are useful for binary exploitation and reverse-engineering

Installing GDB + pwndbg

macOS:

- GDB cannot debug **native** programs on Apple silicon (`aarch64-darwin`), *but can still debug binaries for other platforms (including x86)*
- Use our Docker container!

WSL/Linux:

```
$ sudo apt install gdb
$ git clone https://github.com/pwndbg/pwndbg && cd pwndbg && ./setup.sh
```

pwn-docker

For debugging and running x86 applications on **arm64 macs**

- if you have e.g. a windows arm machine, talk to us after the meeting

Installation

You must be running macOS 13 or newer!

Enable Rosetta:

```
$ /usr/sbin/softwareupdate --install-rosetta --agree-to-license
```

Download the latest Docker Desktop and:

- Enable '**Use Virtualization Framework**' in 'Settings > General'
- Enable '**Use Rosetta for x86/amd64 on Apple Silicon**' in 'Settings > Features in Development'

Clone `pwn-docker`:

```
git clone https://github.com/sigpwny/pwn-docker.git
```

pwn-docker Usage

`./create.sh` - Run this to start your container. Type 'y' to initialize a permanent container, or 'n' for a temporary container. Don’t start in background – still WIP.

`./connect.sh` - Connect to your permanent container after it has been stopped

GDB *should* work, ask in Discord if you run into a problem

```
$ file ./challenge
challenge: ELF 64-bit LSB pie executable, x86-64, ...
$ ROSETTA_DEBUGSERVER_PORT=1234 ./challenge
$ gdb ./challenge -ex 'target remote localhost:1234'
```

Next Meetings

2025-09-21 • This Sunday

- Fall CTF 2025
- Beginner CTF competition with prizes, game badges, free food, recruiters, and fun challenges!

2025-09-25 • Next Thursday

- Open Source Intelligence (OSINT)
- Learn how to use publicly available resources to gather valuable information.

ctf.sigpwny.com

sigpwny{fallctf_is_on_9/21}

Meeting content can be found at sigpwny.com/meetings.

SIGPwny