A beginners guide to setting up a CTF environment

Introduction

This guide will help you set up a CTF environment for the mini CTF. This guide is intended for beginners and will walk you through the process of setting up your environment and the tools that you may need.
It is intended as a guide for beginners, so if you already have some experience feel free to ignore us, and do your own thing. That being said, pay attention to the python setup guide; this is the most common source of bugs when playing CTFs as your packages will begin to conflict.

The prerequisites

If you're running Windows we heavily recommend installing and using WSL (Windows Subsystem for Linux). The many security tools don't provide support for windows, and experience with the linux commandline will become a requirement further into your degree and into CTFing.

If you're on a mac, some steps my differ. A number of committee members use mac, so we can assist with any issues you may have,

If you run into any problems DM @social_anthrax on discord or message in the sigint discord.

Python environments

During CTFs it's common for different challenges or programs to require different versions of python at the same time, and for libraries to require different versions of the same dependency. For this reason we recommend using pyenv, pipx and poetry.

pyenv

Pyenv is a simple python version management tool that allows you to have any number of python versions installed at the same time without breaking your system python.

For macos we recommend using the homebrew install method, and for linux/WSL we recommend the automatic installer:

https://github.com/pyenv/pyenv?tab=readme-ov-file#getting-pyenv

After installing pyenv, you can install a python version with the following command:

```
pyenv install 3.12
pyenv global 3.12 # sets the default python version
pyenv local 3.11 # sets the python version for the current directory
```

pipx

pipx is a tool used to run python programs in isolated environments. This is useful for running tools that you don't want to install system-wide, or that have conflicting dependencies (as everything always does).

Before installing pipx we recommend following this guide first to make sure that

pipx will use the correct pythong environment: https://gabnotes.org/how-use-pipx-pyenv/

After that you can install pipx by following the steps on the github repo: https://github.com/pypa/pipx

Just to make sure that everything is set up correctly, run the command in the guide above to make sure that pipx is still using the correct python version.

poetry

Poetry is a python package manager that is used to manage dependencies and virtual environments. It's a bit more user-friendly than pipenv, and is the recommended tool for managing python dependencies.

You can install poetry by following the instructions on the github repo using pipx: https://python-poetry.org/docs/#installation

After installing poetry create a directory for CTFs and create a file called pyproject.toml with the following contents:

```
[tool.poetry]
name = "pwn-env"
version = "0.1.0"
description = ""
authors = ["social_anthrax <me@anthrax.social>"]
readme = "README.md"
packages = []

[tool.poetry.dependencies]
python = "^3.11"
pwntools = "^4.9.0"
angr = "^9.2.51"
monkeyhex = "^1.7.4"
```

```
ipython = "^8.11.0"
ipykernel = "^6.23.3"
pyelftools = "0.29"
requests = "^2.31.0"
ropper = "^1.13.8"
ropgadget = "^7.4"
jupyter = "^1.0.0"

[tool.poetry.group.data_analysis]
# Honestly most of the time we don't want to touch pandas anyways
optional = true

[tool.poetry.group.data_analysis.dependencies]
pandas = "^2.1.0"
tabulate = "^0.9.0"

[build-system]
requires = ["poetry-core"]
build-backend = "poetry.core.masonry.api"
```

Then run the following and you're good to go!

```
poetry install
poetry shell
```

Docker

Docker is a containerisation tool that we will only be using for crypto challenges in miniCTF.
It is a very useful tool to have in your toolkit, and is used in industry for deploying applications and getting environments set up quickly.
We recommend installing docker using the official guide: https://docs.docker.co

m/get-docker/

Tools

Depending on which disciplines you're interested in, you may need different tools. Here are some of the common tools we expect you to need during our minictf:

Reverse engineering

- Ghidra: https://ghidra-sre.org/ (free, and all you need for most of the CTF)
  - Some alternatives:
    - IDA Pro: https://www.hex-rays.com/products/ida/ (Only if you already have it)
      - Binary Ninja: https://binary.ninja/ (Only if you already have it)
      - Python (follow the guide above)

Pwn

- Pwntools (Installed with the poetry guide above)
- Ghidra: https://ghidra-sre.org/ (free, and all you need for most of the CTF)
- GDB: https://www.gnu.org/software/gdb/ (this will fail on mac, contact us for help if you want to go ahead with pwn)
  - pwndbg (GDB plugin): https://github.com/pwndbg/pwndbg

Web

- Burp Suite: https://portswigger.net/burp
- Wireshark: https://www.wireshark.org/
- Python (follow the guide above)

Crypto

Linux and WSL

There is a great docker container developed by our friends at cryptohack.org (thank's hyperreality) that has all the tools you need for crypto challenges.

```
docker run -it --platform linux/amd64 -p 127.0.0.1:8888:8888 -v $(pwd):/home/sage/ctf  hyperreality/cryptohack:latest
```

MacOS

The cryptohack docker container is unfortunately not built for MacOS and the emulation fails to run it correctly. Instead you can use the following command to run a sagemath container that has all the tools you need for crypto challenges. This is missing some of the more ctf focussed tools that are required for more advanced CTFs but should be more than adequate for our miniCTF.

```
docker run -it --platform linux/amd64 -p 127.0.0.1:8888:8888 -v $(pwd):/home/sage/ctf  sagemath/sagemath:latest "sage -n jupyter --NotebookApp.token='' --no-browser --ip='0.0.0.0' --port=8888"
```

Misc

If you're doing misc then seek any and every god that will listen. There is no hope for you. 👍