Brian O'Neil x23321156 Practicum Config

Env Info and Config README

1. Linux Env

Env Info	Version/ Type
Kernel Release	6.11.0-26-generic
Machine	x86_64
OS	GNU/Linux
Distro	Ubuntu 24.04.2 LTS
Install	Bare Metal

2. Environment set up For BCC

Complete instructions to install BCC can be found in the BCC project repository. Please see https://github.com/iovisor/bcc/blob/master/INSTALL.md. The Debian Binary is used from here - see section 2.2 below.

In short the following actions can be followed to install on Ubuntu.

2.1 Update packages and install build tools

Install essential build tools, it is assumed python 3 is asssumed python 3 and git are installed.

```
sudo apt update && sudo apt upgrade -y
```

```
sudo apt install -y build-essential cmake clang llvm libelf-dev gcc-multilib linux-headers-$(uname -r)
```

2.2 Install the BPF Compiler Collection (BCC)

Python bindings are also added in addition to the suggested packages from the install instructions on the bcc repo. Example scripts `libbcc-examples` appear to be defunct so were not installed.

```
sudo apt install -y bpfcc-tools libbpfcc libbpfcc-dev  python3-bpfcc
```

To confirm bcc is up and running:

```
# in one terminal:
sudo execsnoop-bpfcc

# in a second terminal
ls
```

From this you should expect output like so (you can ignore compiler warnings).

```
PCOMM            PID     PPID    RET ARGS
ls               36771   36750     0 /usr/bin/ls --color=auto
```

2.3 psutil

The python package psutil was installed globally in the environment using `sudo apt install python3-psutil`

2.4 pandas

The python package pandas was installed globally using `sudo apt install python3-pandas`. If installing this way other dependencies such as numpy, matplotlib and scipy should be installed automatically.

3. Installing the Rootkit

This project uses the CARAXES rootkit. The repository and install instructions can be found @ https://github.com/ait-aecid/caraxes/tree/mainhttps://github.com/ait-aecid/caraxes/tree/main

Please read the disclaimer carefully if you decide to install it.

3.1 Citation

CARAXES was developed in conjunction with the following publication and by the authors listed:

> Landauer, M., Alton, L., Lindorfer, M., Skopik, F., Wurzenberger, M., & Hotwagner, W. (2025). Trace of the Times: Rootkit Detection through Temporal Anomalies in Kernel Activity. Under Review.

Practicum Application README

The Env Info and Config README describes setting up the environment and its dependencies including installation of the rootkit and should be completed before executing and instructions below. Practicum README describes what to expect from the application and provides a how-to guide on how run it.

1. Running the Application

The application must be run with root permissions and the following command is the general format used.

```
sudo python3 run.py <run-type> <sched-policy>
```

The run.py script will take several options. These are twofold.

1.1 File Naming

Firstly, the `<run-type>` and `<sched-policy>` options are to aid with file naming. Data is required to be gathered for 'clean' runs where there is no rootkit present in the kernel. Files will be output into the `/csv_data` directory in the root of the application. They are formatted to be named: `<timestamp>_<sched_policy_type>_<run_type>.csv`

1.2 SCHED Policy

Secondly, a sched policy can be chosen. By default a the sched policy runs in what is known as 'normal' or 'other' mode. No `<sched-policy>` option is required or provided to run this.

1.3 Options

Options are listed below. Either `-c` or `-r` are required.

- `-c` Intended to output clean data and to be run without a rootkit present.
- `-r` Intended to output data that has been gathered with a rootkit present.
- `--sched-fifo` This is optional. If used it will run the SCHED_FIFO schedular policy in addition to naming the output file with the `sched_fifo` infix.

1.4 Examples

1. Running with the clean option.

```
sudo python3 run.py -c

# this will output a file like:
# 20250809_153148_sched_normal_clean_data.csv
```

2. Running with the rootkit option.

```
sudo python3 run.py -r

# This will output a file like:
# 20250809_193941_sched_normal_rookit_data.csv
```

3. Running with SCHED_FIFO and clean options.

```
sudo python3 run.py -c --sched-fifo

# This will output a file like:
# 20250809_194700_sched_fifo_clean_data.csv
```

4. Running with SCHED_FIFO and rootkit options.

```
sudo python3 run.py -r --sched-fifo

# This will output a file like:
# 20250809_195002_sched_fifo_rookit_data.csv
```

2. Running the Rootkit

As per instructions found at https://github.com/ait-aecid/caraxes/tree/main:

> Run ls in the directory - you should see several files as depicted below. Run sudo insmod caraxes.ko to load the rootkit into the kernel. Now, run ls again - all files that contain the magic word "caraxes" are hidden from the user. To make the files visible, just remove the rootkit from the kernel using sudo rmmod caraxes.

```
ubuntu@ubuntu:~/caraxes$ ls
LICENSE         README.md   caraxes.mod    caraxes.o         hooks.h
modules.order
Makefile        caraxes.c   caraxes.mod.c  caraxes_logo.svg
hooks_filldir.h     rootkit.h
Module.symvers  caraxes.ko  caraxes.mod.o  ftrace_helper.h
hooks_getdents64.h  stdlib.h
ubuntu@ubuntu:~/caraxes$ sudo insmod caraxes.ko
ubuntu@ubuntu:~/caraxes$ ls
LICENSE   Module.symvers  ftrace_helper.h  hooks_filldir.h
modules.order  stdlib.h
Makefile  README.md       hooks.h          hooks_getdents64.h  rootkit.h
ubuntu@ubuntu:~/caraxes$ sudo rmmod caraxes
ubuntu@ubuntu:~/caraxes$ ls
LICENSE         README.md   caraxes.mod    caraxes.o         hooks.h
modules.order
Makefile        caraxes.c   caraxes.mod.c  caraxes_logo.svg
hooks_filldir.h     rootkit.h
Module.symvers  caraxes.ko  caraxes.mod.o  ftrace_helper.h
hooks_getdents64.h  stdlib.h
ubuntu@ubuntu:~/caraxes$ make clean
```

The application itself will run the ls command after run.py is executed. If you wish to gather data for tracing targets while the rootkit is present is expected that the rootkit be run manually through the use of `sudo insmod caraxes.ko`.

2.1 Citation

CARAXES was developed in conjunction with the following publication and by the authors listed:

> Landauer, M., Alton, L., Lindorfer, M., Skopik, F., Wurzenberger, M., & Hotwagner, W. (2025). Trace of the Times: Rootkit Detection through Temporal Anomalies in Kernel Activity. Under Review.

3. Running Data Analysis

Scripts for analysing data analysis are found in `/data_summary_analysis` directory. Presently, the paths for data CVSs are hard coded and data is supplied in `/csv_data`. Running the application will produce more CVSs and name them automatically adding to the same folder, but they would need to be added to the required paths is it is desired to analyse new data.