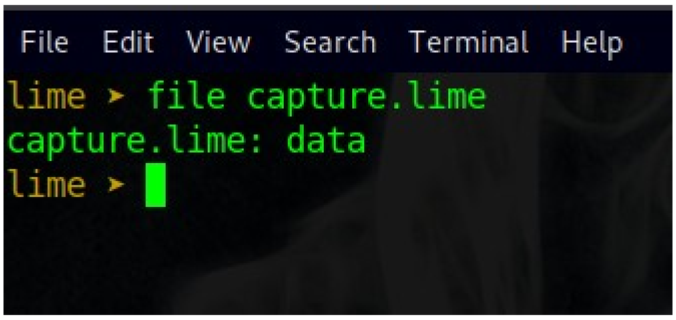
Memory Forensics Writeup
Vault: Forensics @ AfricaHackon Quals

As my first time playing the Africahackon ctf I would say I really enjoyed playing the challenges. The Forensics vault was one that took hours of my ctf time lol, but was worth the time at the end.
This is a writeup of the process I used to solve the challenge.

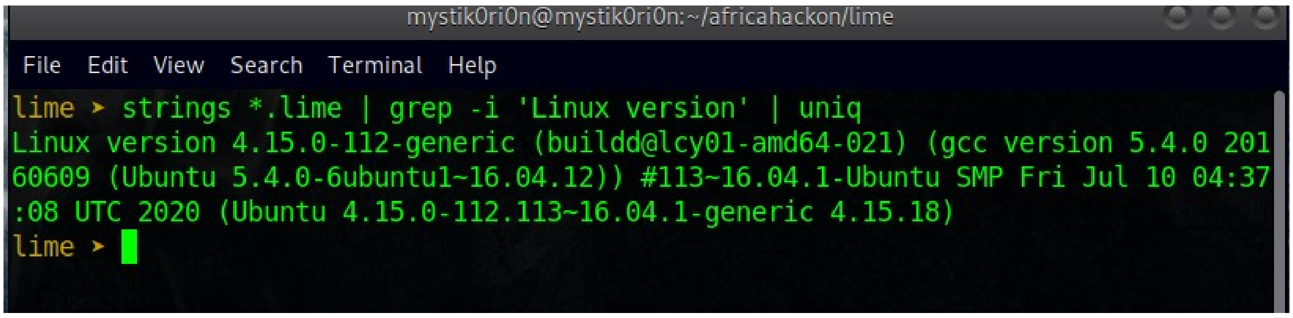
Analysis

A file was provided with a .lime extension which from some previous ctf I learned that such files are the result of extracting memory from a linux system using the Linux Memory Extractor tool → LiME. Using the command file on the memory dump shows that it contains data.



```
lime ➤ file capture.lime
capture.lime: data
lime ➤
```

Since the file we are working on is a memory dump, the go to tool to analyse it is Volatility. But for it to analyse the file , it needs to know from what system the memory dump was fetched. Since we know from Lime that it is obviously a linux system, we know that running a profile search with volatility would yield nothing. Unless we know what linux distribution it belongs to which we can identify by:
strings *.lime | grep -i 'Linux version' | uniq


mystik0ri0n@mystik0ri0n:~/africahackon/lime
File Edit View Search Terminal Help

```
lime ➤ strings *.lime | grep -i 'Linux version' | uniq
Linux version 4.15.0-112-generic (buildd@lcy01-amd64-021) (gcc version 5.4.0 201
60609 (Ubuntu 5.4.0-6ubuntu1~16.04.12)) #113~16.04.1-Ubuntu SMP Fri Jul 10 04:37
:08 UTC 2020 (Ubuntu 4.15.0-112.113~16.04.1-generic 4.15.18)
lime ➤
```

From the screenshot we can identify it as Ubuntu 16.04.12, so for our first flag for the most suitable profile;
AH{Ubuntu160412}.
Now we have a distribution and also a kernel version, **Ubuntu16.04** and kernel version **Linux version 4.15.0-112-generic.**

Creating an ubuntu1604 volatility profile

We first clone the volatility 2 from the repo;

$ git clone https://github.com.volatilityfoundation/volatility

$ cd volatility/tools/linux

Change the kernel detection value in the Makefile to match the linux kernel we identified, so for our case;

sed -i 's/initial-kernel-value/4.15.0-112-generic/g' Makefile

Something to keep in mind while creating volatility 2 profiles, you only need the Linux headers and a system map. So, goodbye VM and hello docker :) .

To setup a container that matches our target os(Ubuntu 16.04);

$ docker run –it –rm -v $PWD:/volatility ubuntu:16.04 *bin/*bash

This will sometimes cause an error that mount point doesn't exist, A temporary work around would be;

$ sudo mkdir /sys/fs/cgroup/systemd && sudo mount -t cgroup -o none,name=systemd cgroup /sys/fs/cgroup/systemd

Then run the docker command again;

Install the necessary packages on the container;

# cd volatility

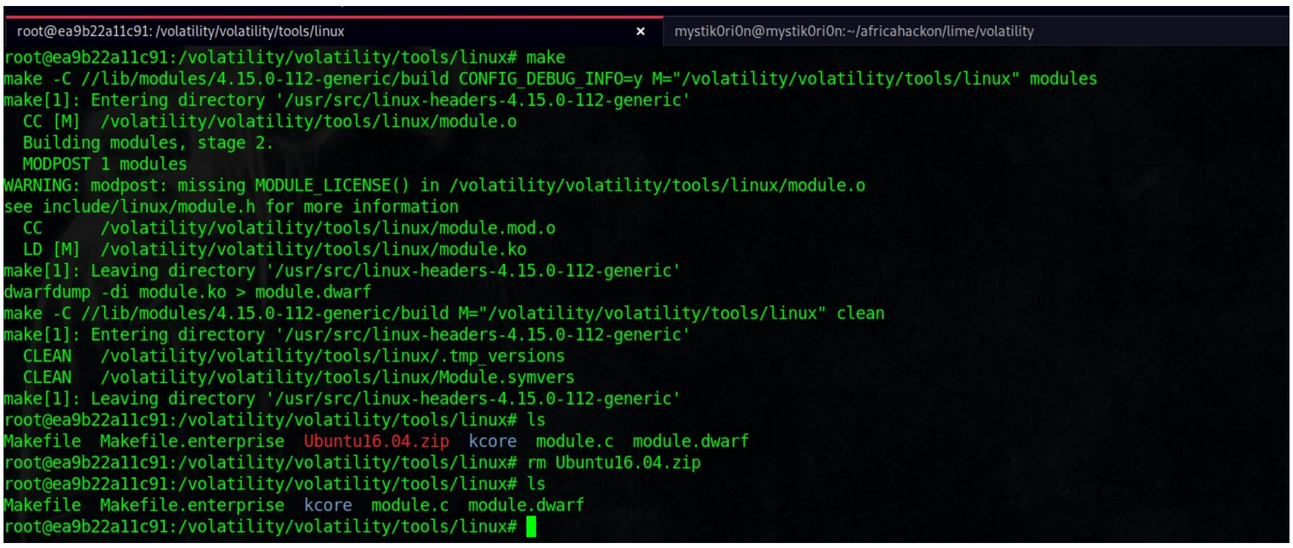
# apt update

# apt install build-essential linux-headers-4.15.0-112-generic dwarfdump make zip linux-image-4.15.0-112-generic

# cd /volatility/tools/linux

Create the dwarf file using the volatility tool

# make


root@ea9b22a11c91: /volatility/volatility/tools/linux
mystik0ri0n@mystik0ri0n:~/africahackon/lime/volatility

```
root@ea9b22a11c91:/volatility/volatility/tools/linux# make
make -C //lib/modules/4.15.0-112-generic/build CONFIG_DEBUG_INFO=y M="/volatility/volatility/tools/linux" modules
make[1]: Entering directory '/usr/src/linux-headers-4.15.0-112-generic'
  CC [M]  /volatility/volatility/tools/linux/module.o
  Building modules, stage 2.
  MODPOST 1 modules
WARNING: modpost: missing MODULE_LICENSE() in /volatility/volatility/tools/linux/module.o
see include/linux/module.h for more information
  CC      /volatility/volatility/tools/linux/module.mod.o
  LD [M]  /volatility/volatility/tools/linux/module.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.15.0-112-generic'
dwarfdump -di module.ko > module.dwarf
make -C //lib/modules/4.15.0-112-generic/build M="/volatility/volatility/tools/linux" clean
make[1]: Entering directory '/usr/src/linux-headers-4.15.0-112-generic'
  CLEAN   /volatility/volatility/tools/linux/.tmp_versions
  CLEAN   /volatility/volatility/tools/linux/Module.symvers
make[1]: Leaving directory '/usr/src/linux-headers-4.15.0-112-generic'
root@ea9b22a11c91:/volatility/volatility/tools/linux# ls
Makefile  Makefile.enterprise  Ubuntu16.04.zip  kcore  module.c  module.dwarf
root@ea9b22a11c91:/volatility/volatility/tools/linux# rm Ubuntu16.04.zip
root@ea9b22a11c91:/volatility/volatility/tools/linux# ls
Makefile  Makefile.enterprise  kcore  module.c  module.dwarf
root@ea9b22a11c91:/volatility/volatility/tools/linux# 
```

Zip the dwarf file and the System Map

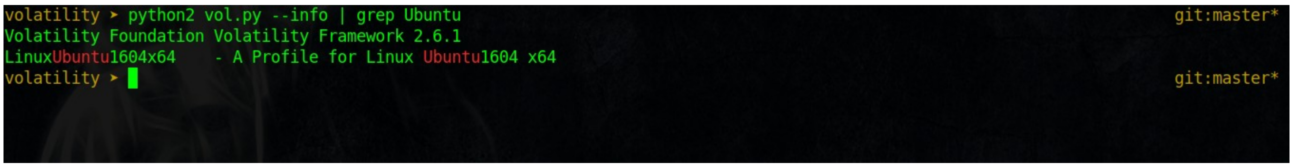
# zip Ubuntu16.04.zip module.dwarf *boot*/system.map-4.15.0.112-generic

# exit

$ cp ubuntu1604.zip <volatility-tool-folder>/volatility/plugins/overlays/linux

Now we have a profile, and for the moment of truth,whether it works for us or back to the drawing board lol!

Confirm that the profile is read by volatility

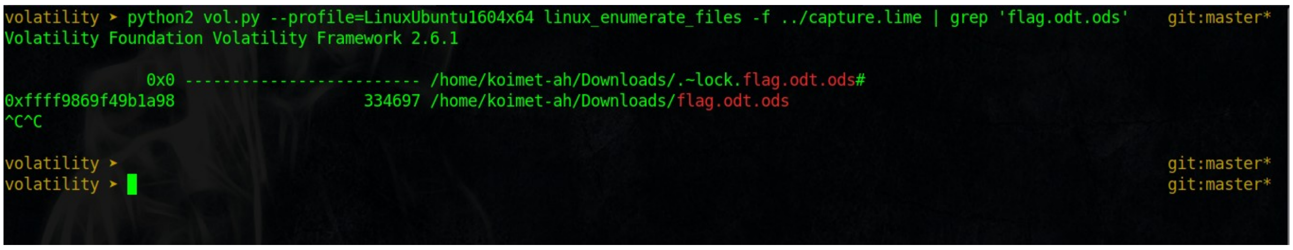


```
volatility ➤ python2 vol.py --info | grep Ubuntu                                                              git:master*
Volatility Foundation Volatility Framework 2.6.1
LinuxUbuntu1604x64    - A Profile for Linux Ubuntu1604 x64
volatility ➤                                                                                                  git:master*
```

Solving with volatility

Conduct a file search for flag.odt.ods

python2 vol.py --profile=LinuxUbuntu1604x64 linux_enumerate_files -f ../capture.lime | grep 'flag.odt.ods'

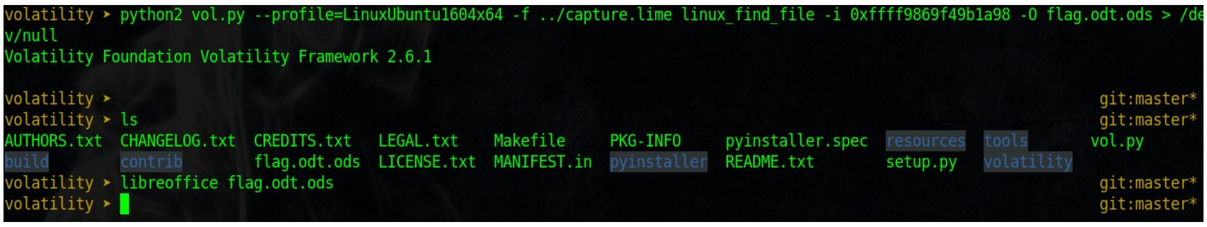


```
volatility ➤ python2 vol.py --profile=LinuxUbuntu1604x64 linux_enumerate_files -f ../capture.lime | grep 'flag.odt.ods'    git:master*
Volatility Foundation Volatility Framework 2.6.1

             0x0 ------------------------ /home/koimet-ah/Downloads/.~lock.flag.odt.ods#
0xffff9869f49b1a98                   334697 /home/koimet-ah/Downloads/flag.odt.ods
^C^C

volatility ➤                                                                                                  git:master*
volatility ➤                                                                                                  git:master*
```

Found it! , now to dump it and read the flag

python2 vol.py --profile=LinuxUbuntu1604x64 -f ../capture.lime linux_find_file -i 0xffff9869f49b1a98 -O flag.odt.ods

a h { h o p e _ y o u _ b u i l t _ a _ c u s t o m _ p r o f i l



```
volatility ➤ python2 vol.py --profile=LinuxUbuntu1604x64 -f ../capture.lime linux_find_file -i 0xffff9869f49b1a98 -O flag.odt.ods > /de
v/null
Volatility Foundation Volatility Framework 2.6.1

volatility ➤                                                                                                  git:master*
volatility ➤ ls                                                                                               git:master*
AUTHORS.txt  CHANGELOG.txt  CREDITS.txt   LEGAL.txt    Makefile     PKG-INFO     pyinstaller.spec  resources  tools       vol.py
build        contrib        flag.odt.ods  LICENSE.txt  MANIFEST.in  pyinstaller  README.txt        setup.py   volatility
volatility ➤ libreoffice flag.odt.ods                                                                         git:master*
volatility ➤                                                                                                  git:master*
```

flag → **ah {hope_you_built_a_custom_profile}**

References

Visit the link below for additional information on creating both volatility-2 and volatility-3 profiles. Using volatility 3 requires symbol tables instead of profiles thus the need for another approach.

https://beguier.eu/nicolas/articles/security-tips-3-volatility-linux-profiles.html