Created By:
https://guyeldad.github.io/
linkedin.com/in/guy-eldad/

Creating Custom Memory Profiles for Volatility2

What is the Linux kernel version of this memory dump?

We need to run the command “vol -f '/root/Desktop/ChallengeFile/MyW3B.vmem' banners.Banners” to extract the banner information from the Linux system captured in the memory dump.

```
root@ip-172-31-27-224:~/Desktop/ChallengeFile# vol -f '/root/Desktop/ChallengeFile/MyW3B.vmem' banners.Banners
Volatility 3 Framework 2.7.0
WARNING  volatility3.framework.layers.vmware: No metadata file found alongside VMEM file. A VMSS or VMSN file may b
e required to correctly process a VMEM file. These should be placed in the same directory with the same file name,
e.g. MyW3B.vmem and MyW3B.vmss.
Progress:  100.00               PDB scanning finished
Offset  Banner

0x1b2001a0      Linux version 5.4.0-150-generic (buildd@bos03-amd64-012) (gcc version 7.5.0 (Ubuntu 7.5.0-3ubuntu1~
18.04)) #167~18.04.1-Ubuntu SMP Wed May 24 00:51:42 UTC 2023 (Ubuntu 5.4.0-150.167~18.04.1-generic 5.4.233)
0x1c197e14      Linux version 5.4.0-150-generic (buildd@bos03-amd64-012) (gcc version 7.5.0 (Ubuntu 7.5.0-3ubuntu1~
18.04)) #167~18.04.1-Ubuntu SMP Wed May 24 00:51:42 UTC 2023 (Ubuntu 5.4.0-150.167~18.04.1-generic 5.4.233)
0x7bd00010      Linux version 5.4.0-150-generic (buildd@bos03-amd64-012) (gcc version 7.5.0 (Ubuntu 7.5.0-3ubuntu1~
18.04)) #167~18.04.1-Ubuntu SMP Wed May 24 00:51:42 UTC 2023 (Ubuntu 5.4.0-150.167~18.04.1-generic 5.4.233)
root@ip-172-31-27-224:~/Desktop/ChallengeFile#
```

As we can see, the Linux kernel version is "5.4.0-150-generic" and the Ubuntu version is "Ubuntu 5.4.0-150.167~18.04.1". This indicates that the memory dump was acquired from Ubuntu OS version 18.04.1.

Now, we need to create a custom Linux profile for Volatility. For this, we need to install a brand-new machine with the same Ubuntu OS version “18.04.1” we found in the previous question.

Search on Google “ubuntu 18.04.1” and look for a download.

I downloaded the ISO image from the link http://ftp.au.debian.org/ubuntu-releases/18.04.1.0/

Look for the file “ubuntu-18.04.1-desktop-amd64.iso” and download it.

Name	Last modified	Size
Parent Directory		-
FOOTER.html	2018-11-30 11:02	810
HEADER.html	2018-11-30 11:03	4.1K
MD5SUMS	2018-11-30 10:27	140
MD5SUMS-metalink	2018-11-30 10:27	150
MD5SUMS-metalink.gpg	2018-11-30 10:27	916
MD5SUMS.gpg	2018-11-30 10:27	916
SHA1SUMS	2018-11-30 10:27	156
SHA1SUMS.gpg	2018-11-30 10:27	916
SHA256SUMS	2018-11-30 10:27	204
SHA256SUMS.gpg	2018-11-30 10:27	916
ubuntu-18.04.1-desktop-amd64.iso	2018-07-25 13:22	1.8G

Now, we need to create a new virtual machine and insert the ISO file to begin installing the Ubuntu OS.

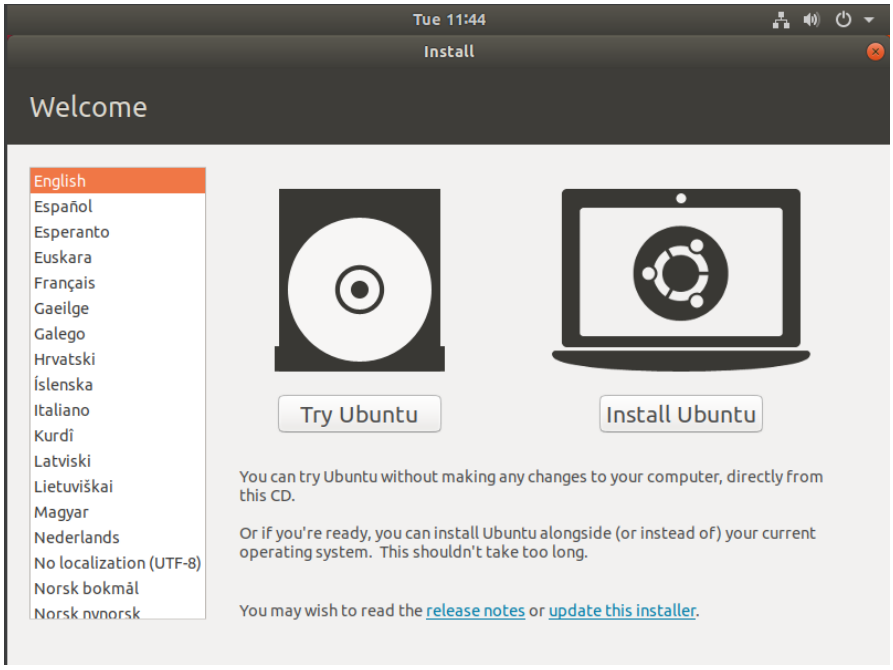
You can use your preferred VM tools. I will be using Oracle VM VirtualBox Manager.

Open VirtualBox – Machine – New and choose the name of the VM, the path, and the version Ubuntu (64-bit) and click Finish.

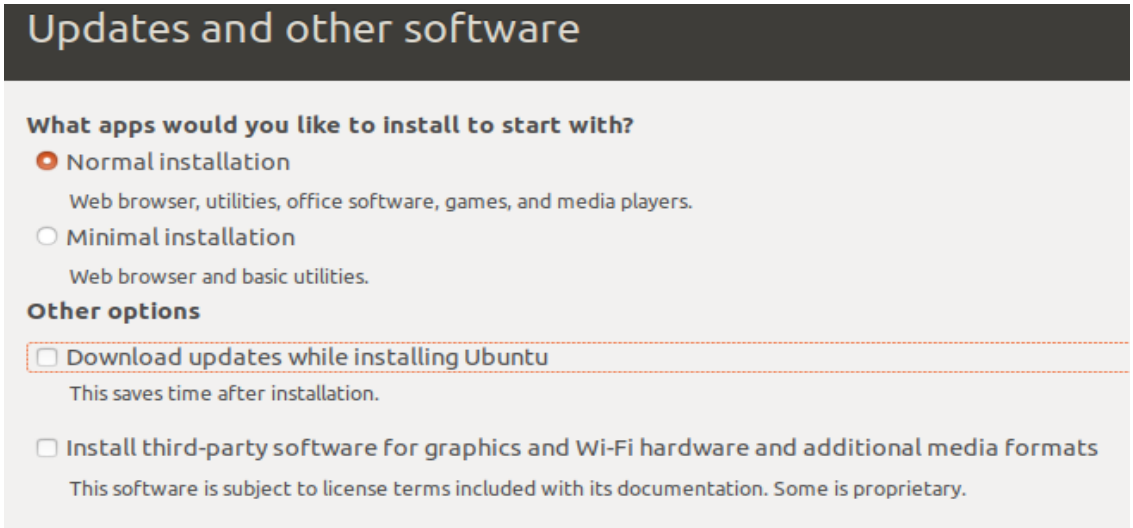
Note: You can also allocate more VRAM and Processors to your machine under the "Hardware" settings.

Start the VM, and a pop-up window will prompt you to select the ISO image we just downloaded.

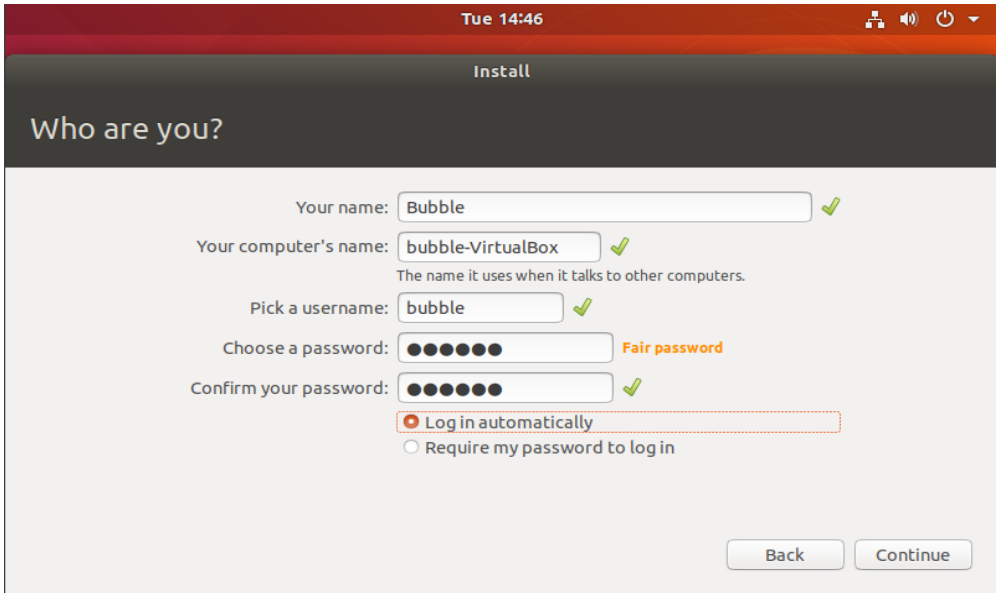
Click Install Ubuntu and continue.



Uncheck the option "Download updates while installing Ubuntu"



When you reach the “Who are you?” window, choose a name for the computer, set a password, and continue.



Now, after the installation is complete, we need to install VirtualBox Guest Additions so that we can use Fullscreen mode and also utilize the copy and paste feature on the virtual machine.

Note: You can also take a Snapshot, allowing you to revert to a fresh start of the machine at any time.

On the virtual machine menu above, click on “Devices” and then choose “Insert Guest Additions CD Image”. The installation will start automatically.

After the installation is complete, make sure to restart the machine.

Click again on “Devices” in the VM settings above, and choose “Bidirectional” for both “Shared Clipboard” and “Drag and Drop” options. This allows us to use copy and paste functionality and transfer files in and out of the virtual machine.

Now, open the Terminal and use the command “sudo apt install linux-headers-$(uname -r) build-essential dkms”

This command installs several packages that are often needed for building and compiling kernel modules on a Linux system.

After the installation is complete, make sure to restart the virtual machine once more. Then, you can use Fullscreen mode or stretch the window to your preference.

Now that we have everything set up, we need to download Volatility onto the machine. Open the Terminal and use the command "**sudo apt install git**".

This command allows us to download git repositories.

Install the volatility using the command:

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

```
bubble@bubble-VirtualBox:~$ git clone https://github.com/volatilityfoundation/volatility.git
Cloning into 'volatility'...
remote: Enumerating objects: 27411, done.
remote: Total 27411 (delta 0), reused 0 (delta 0), pack-reused 27411
Receiving objects: 100% (27411/27411), 21.10 MiB | 13.14 MiB/s, done.
Resolving deltas: 100% (19758/19758), done.
```

Now we need to install the version of the Linux image.

Use the command "**sudo apt update; sudo apt install linux-image-5.4.0-150-generic**"

This command updates the package list on our system and then installs the Linux kernel version 5.4.0-150-generic.

After the installation finished, restart the machine and use "**uname -a**" to verify the kernel is installed.

```
bubble@bubble-VirtualBox:~$ uname -a
Linux bubble-VirtualBox 5.4.0-150-generic #167~18.04.1-Ubuntu SMP Wed May 24 00:51:42 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux
```

Note: Sometimes the kernel is not updated after one restart, so restart again, and then it should work.

Now go to "**volatility/tools/linux**" and install dwarfdump.

Use the command "**sudo apt install dwarfdump**"

"dwarfdump" is a command-line tool used to display information about the DWARF debugging information format contained in compiled binary files (executables, object files, shared libraries, etc.). DWARF is a widely-used format for debugging data in compiled code, providing detailed information about the program's source code, variables, data structures, and more.

```
bubble@bubble-VirtualBox:~$ cd volatility/tools/linux/
bubble@bubble-VirtualBox:~/volatility/tools/linux$ sudo apt install dwarfdump
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  dwarfdump
0 upgraded, 1 newly installed, 0 to remove and 741 not upgraded.
Need to get 249 kB of archives.
After this operation, 643 kB of additional disk space will be used.
Get:1 http://il.archive.ubuntu.com/ubuntu bionic/universe amd64 dwarfdump amd64 20180129-1 [249 kB]
Fetched 249 kB in 1s (243 kB/s)
Selecting previously unselected package dwarfdump.
(Reading database ... 132860 files and directories currently installed.)
Preparing to unpack .../dwarfdump_20180129-1_amd64.deb ...
Unpacking dwarfdump (20180129-1) ...
Setting up dwarfdump (20180129-1) ...
Processing triggers for man-db (2.8.3-2) ...
```

Then use "sudo apt install build-essential"

This command installs a package that provides a comprehensive set of tools required for compiling and building software on Debian-based Linux systems

```
bubble@bubble-VirtualBox:~/volatility/tools/linux$ sudo apt install build-essential
Reading package lists... Done
Building dependency tree
Reading state information... Done
build-essential is already the newest version (12.4ubuntu1).
0 upgraded, 0 newly installed, 0 to remove and 741 not upgraded.
```

Then use the following commands:

sudo ln -s /usr/src/linux-headers-5.4.0-150-generic /lib/modules/5.4.0-150-generic/build

This command creates a symbolic link in the kernel modules directory, pointing to the kernel headers. This setup is needed for building kernel modules.

sudo apt-get install linux-headers-5.4.0-150-generic

This command installs the Linux kernel headers for version 5.4.0-150-generic. Kernel headers are necessary for building and compiling kernel modules.

```
bubble@bubble-VirtualBox:~/volatility/tools/linux$ sudo ln -s /usr/src/linux-headers-5.4.0-150-generic /lib/modules/5.4.0-150-generic
/build
bubble@bubble-VirtualBox:~/volatility/tools/linux$ sudo apt-get install linux-headers-5.4.0-150-generic
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  linux-hwe-5.4-headers-5.4.0-150
The following NEW packages will be installed:
  linux-headers-5.4.0-150-generic linux-hwe-5.4-headers-5.4.0-150
0 upgraded, 2 newly installed, 0 to remove and 741 not upgraded.
Need to get 12.3 MB of archives.
After this operation, 86.0 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://il.archive.ubuntu.com/ubuntu bionic-updates/main amd64 linux-hwe-5.4-headers-5.4.0-150 all 5.4.0-150.167~18.04.1 [11.0 M
B]
Get:2 http://il.archive.ubuntu.com/ubuntu bionic-updates/main amd64 linux-headers-5.4.0-150-generic amd64 5.4.0-150.167~18.04.1 [1,32
1 kB]
Fetched 12.3 MB in 2s (5,480 kB/s)
Selecting previously unselected package linux-hwe-5.4-headers-5.4.0-150.
(Reading database ... 132880 files and directories currently installed.)
Preparing to unpack .../linux-hwe-5.4-headers-5.4.0-150_5.4.0-150.167~18.04.1_all.deb ...
Unpacking linux-hwe-5.4-headers-5.4.0-150 (5.4.0-150.167~18.04.1) ...
Selecting previously unselected package linux-headers-5.4.0-150-generic.
Preparing to unpack .../linux-headers-5.4.0-150-generic_5.4.0-150.167~18.04.1_amd64.deb ...
Unpacking linux-headers-5.4.0-150-generic (5.4.0-150.167~18.04.1) ...
Setting up linux-hwe-5.4-headers-5.4.0-150 (5.4.0-150.167~18.04.1) ...
Setting up linux-headers-5.4.0-150-generic (5.4.0-150.167~18.04.1) ...
/etc/kernel/header_postinst.d/dkms:
 * dkms: running auto installation service for kernel 5.4.0-150-generic
   ...done.
```

Now, use the command “**make**” and then “**ls**”. We can see that we created the file module.dwarf

The “make” command help create a custom Linux profile for Volatility, enabling it to analyze memory dumps from systems running specific kernel version.

"module.dwarf" is a file containing debugging information crafted during the compilation of a kernel module through the "make" command.

```
bubble@bubble-VirtualBox:~/volatility/tools/linux$ make
make -C //lib/modules/5.4.0-150-generic/build CONFIG_DEBUG_INFO=y M="/home/bubble/volatility/tools/linux" modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-150-generic'
  CC [M]  /home/bubble/volatility/tools/linux/module.o
  Building modules, stage 2.
  MODPOST 1 modules
WARNING: modpost: missing MODULE_LICENSE() in /home/bubble/volatility/tools/linux/module.o
see include/linux/module.h for more information
  CC [M]  /home/bubble/volatility/tools/linux/module.mod.o
  LD [M]  /home/bubble/volatility/tools/linux/module.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-150-generic'
dwarfdump -di module.ko > module.dwarf
make -C //lib/modules/5.4.0-150-generic/build M="/home/bubble/volatility/tools/linux" clean
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-150-generic'
  CLEAN   /home/bubble/volatility/tools/linux/Module.symvers
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-150-generic'
```

```
bubble@bubble-VirtualBox:~/volatility/tools/linux$ ls
kcore  Makefile  Makefile.enterprise  module.c  module.dwarf
```

Now, let’s check if the System.map file is installed on our /boot directory.

Use “**ls /boot**”. This should show us our System.map file.

```
bubble@bubble-VirtualBox:~/volatility/tools/linux$ ls /boot
abi-4.15.0-20-generic     initrd.img-4.15.0-20-generic  memtest86+_multiboot.bin      vmlinuz-4.15.0-20-generic
config-4.15.0-20-generic  initrd.img-5.4.0-150-generic  retpoline-4.15.0-20-generic   vmlinuz-5.4.0-150-generic
config-5.4.0-150-generic  memtest86+.bin                System.map-4.15.0-20-generic
grub                      memtest86+.elf                System.map-5.4.0-150-generic
```

Here we can see our “System.map-5.4.0-150-generic” file, which we created using the make command.

Now, navigate back two directories to the volatility directory and use the command:

“**sudo zip volatility/plugins/overlays/linux/Ubuntu540.zip tools/linux/module.dwarf /boot/System.map-5.4.0-150-generic**”

This command creates a zip file named "Ubuntu540.zip". It includes the "module.dwarf" file from the "tools/linux" directory and the "System.map-5.4.0-150-generic" file from the "/boot" directory. These files are located in the specified path within the volatility directory.
The purpose of this command is to generate a specific module tailored for the Ubuntu Linux kernel version specified by "System.map-5.4.0-150-generic".

```
bubble@bubble-VirtualBox:~/volatility/tools/linux$ cd ../..
bubble@bubble-VirtualBox:~/volatility$ sudo zip volatility/plugins/overlays/linux/Ubuntu540.zip tools/linux/module.dwarf /boot/System
.map-5.4.0-150-generic
  adding: tools/linux/module.dwarf (deflated 91%)
  adding: boot/System.map-5.4.0-150-generic (deflated 79%)
```

Now, install Python so we can use Volatility. Use the command "**sudo apt install python**".
After the installation is complete, use the command "**python vol.py --info | grep Linux**" to find the profile we created.

```
bubble@bubble-VirtualBox:~/volatility$ python vol.py --info | grep Linux
Volatility Foundation Volatility Framework 2.6.1
LinuxUbuntu540x64      - A Profile for Linux Ubuntu540 x64
LinuxAMD64PagedMemory          - Linux-specific AMD 64-bit address space.
linux_aslr_shift           - Automatically detect the Linux ASLR shift
linux_banner               - Prints the Linux banner information
linux_yarascan             - A shell in the Linux memory image
```

As we can see, “LinuxUbuntu540x64” is the profile we created.

Now, let's try running a command to see if our profile is working.
python vol.py -f '/home/bubble/Challenge/MyW3B.vmem' --profile=LinuxUbuntu540x64 linux_pslist

```
bubble@bubble-VirtualBox:~/volatility$ python vol.py -f '/home/bubble/Challenge/MyW3B.vmem' --profile=LinuxUbuntu540x64 linux_pslist
Volatility Foundation Volatility Framework 2.6.1
Offset             Name                 Pid             PPid            Uid             Gid    DTB                 Start Time
------------------ -------------------- --------------- --------------- --------------- ------ ------------------- ----------
0xffff9c503b7e8000 systemd              1               0               0               0      0x0000000079718000 2023-09-28 19:00:58 UTC+0000
0xffff9c503b7eaf00 kthreadd             2               0               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503b7e9780 rcu_gp               3               2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503b7ede00 rcu_par_gp           4               2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503ac11780 kworker/0:0H         6               2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503ac14680 mm_percpu_wq         8               2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503ac10000 ksoftirqd/0          9               2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503ac12f00 rcu_sched            10              2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503ac1c680 migration/0          11              2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503ac18000 idle_inject/0        12              2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503afaaf00 cpuhp/0              14              2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503afa9780 cpuhp/1              15              2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503afade00 idle_inject/1        16              2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503afac680 migration/1          17              2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503afa8000 ksoftirqd/1          18              2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503afb8000 kworker/1:0H         20              2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503afbaf00 kdevtmpfs            21              2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503afb9780 netns                22              2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
0xffff9c503afbde00 rcu_tasks_kthre      23              2               0               0      ------------------ 2023-09-28 19:00:58 UTC+0000
```

As we can see, the plugin linux_pslist is working with no errors, which indicates that our profile is functioning correctly.

Created By:
linkedin.com/in/guy-eldad/
https://guyeldad.github.io/