Final Engagement

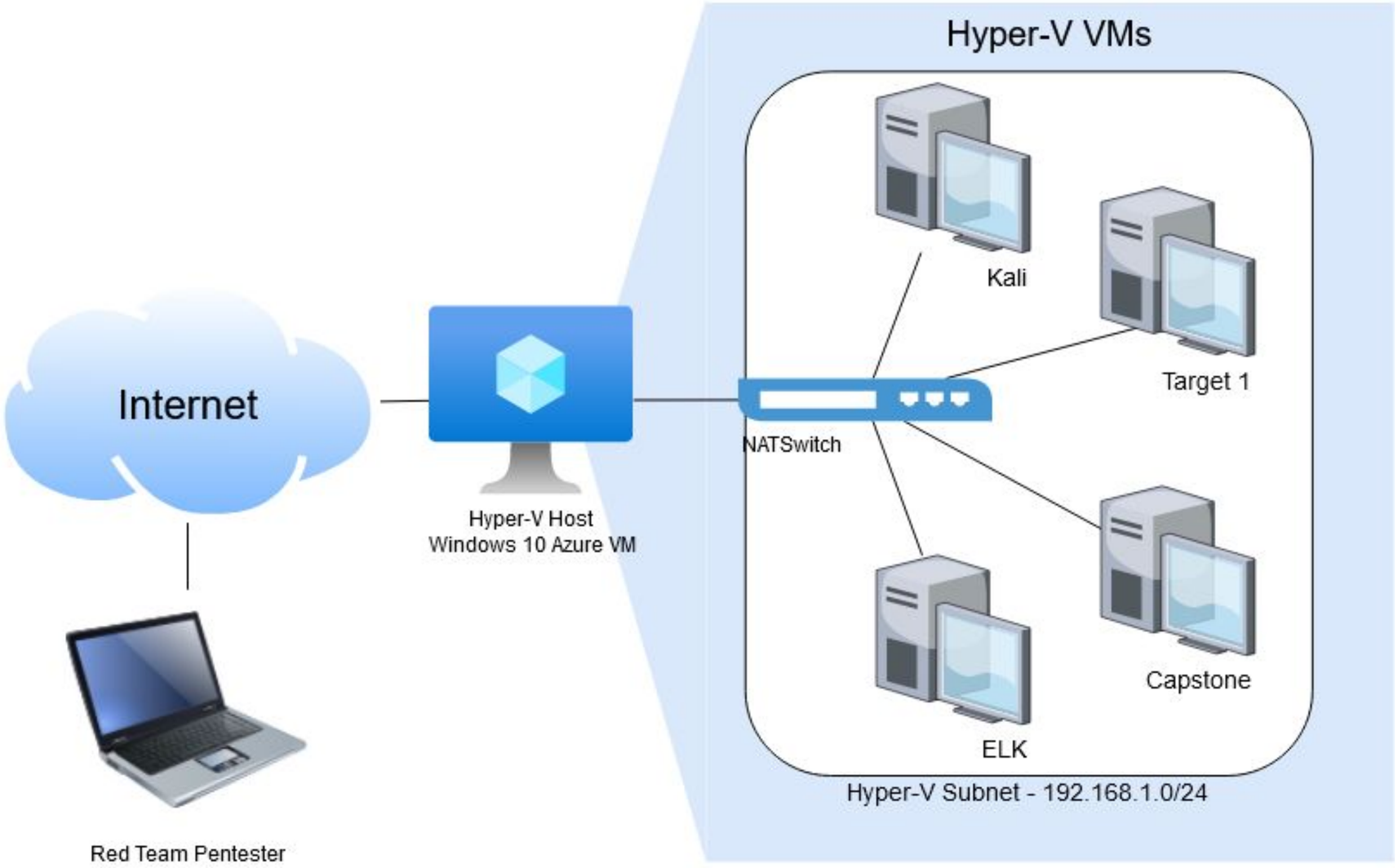
Attack, Defense & Analysis of a Vulnerable Network

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: Debian Kali 5.4.0
Hostname: Kali

IPv4: 192.168.1.110
OS: Debian GNU/Linux 8
Hostname: Target 1

IPv4: 192.168.1.105
OS: Ubuntu 18.04
Hostname: Capstone

IPv4: 192.168.1.100
OS: Ubuntu 18.04
Hostname: ELK

Critical Vulnerabilities: Target 1

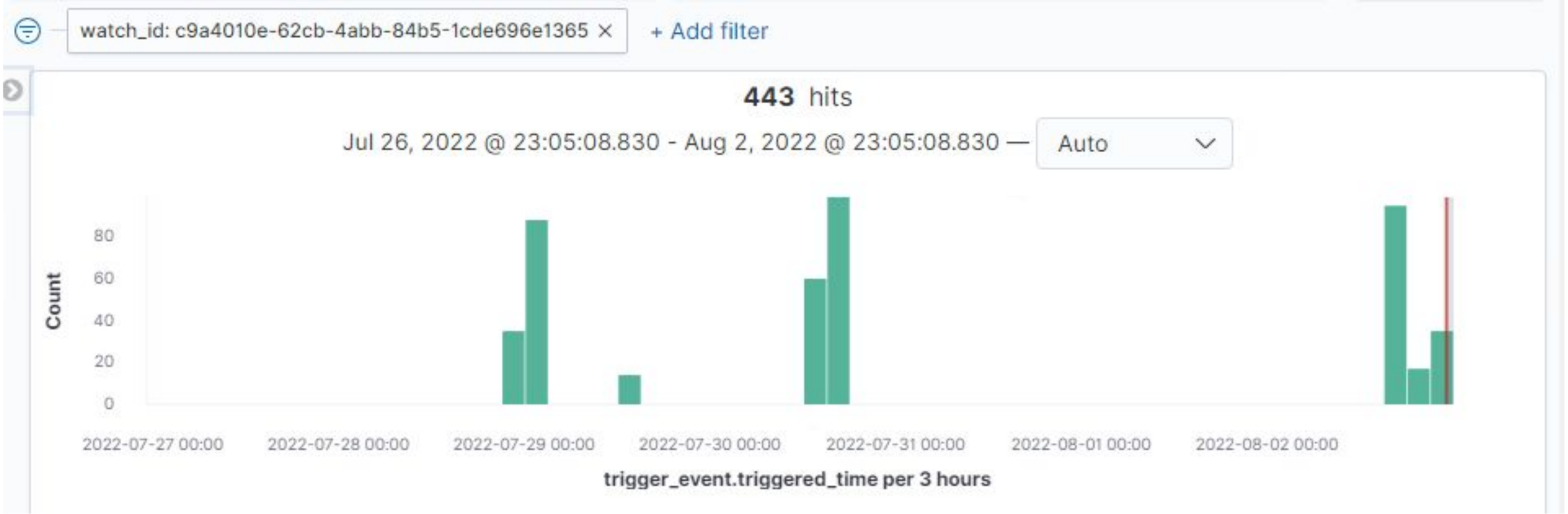
Our assessment uncovered the following critical vulnerabilities in **Target 1**.

Vulnerability	Description	Impact
SSH Port 22 Open, No Keyfile Required	SSH connections are allowed with basic username/password authentication	Able to gain easy terminal access
Unsalted Password Hashes	Easy to reverse hashes	Able to use John to crack hashes
Basic Password for Michael’s Account (CWE-521)	Easy to guess or bruteforce	Able to SSH into Michael’s account
Wordpress Enumeration (CVE-2009-2335)	WPS scan enumeration	Able to discover users
Python Sudo Rights for Steven’s Account	Python can be run as sudo using Steven’s account	Able to escalate privileges
Exposed MySQL Credentials	MySQL database credentials were stored in a config file in plan-text	Able to query the MySQL database

Alerts Implemented

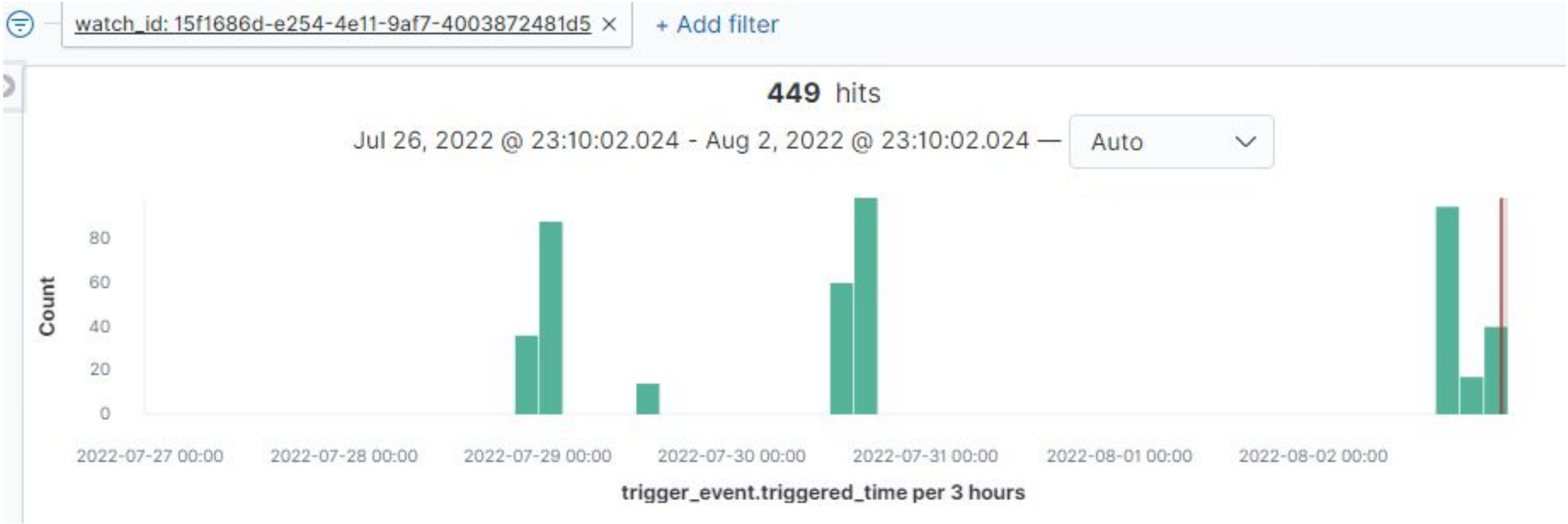
CPU Usage Monitor

- **Metric =** WHEN max ( ) OF `system.process.cpu.total.pct` OVER all documents
- **Threshold** = IS ABOVE 0.5 FOR THE LAST 5 minutes.



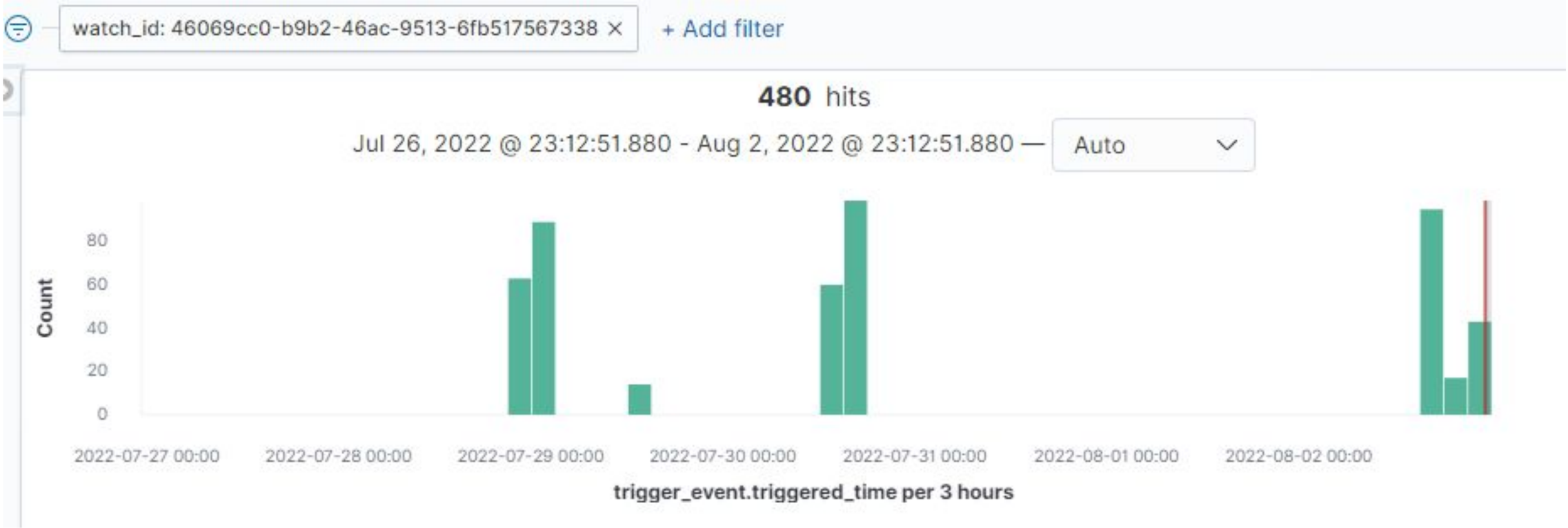
Excessive HTTP Errors

- **Metric =** WHEN counts ( ) GROUPED OVER top 5 `http.response.status_code`
- **Threshold =** IS ABOVE 400 FOR THE LAST 5 minutes



HTTP Request Size Monitor

- **Metric =** WHEN sum ( ) of `http.request.bytes` OVER all documents
- **Threshold** = IS ABOVE 3500 FOR THE LAST 1 minute



Hardening

Hardening Against Open Ports on Target 1

- Hackers will target open SSH servers on port 22. SSH receives incoming connections by default on port 22. Moving to another port will change the default to not listen for connections.
  - How to install it:
    - In /etc/ssh/sshd_config
    - Add Lines:
      - #Run SSH on a non standard port
      - Port 2025 #Change me
      - exit file
    - Run: service sshd restart
- Enable a firewall to allow SSH traffic only from designated management systems
- Require a keyfile for SSH connections

Hardening Against Basic Password on Target 1

- More complex password take much longer to bruteforce
- Add password parameters: numbers, symbols, uppercase, and lowercase letters
- Multi-factor authentication
- Password expiration dates (mandatory resets)
- No re-use of recently used passwords

Hardening Against Unsalted Password Hashes on Target 1

- Salted password hashes cannot be reversed due to the fact that they create unique hashes even when hashing the same phrase
- Apply salt whenever hashing is being performed to maintain security

Hardening Against WordPress User Enumeration

- Use a plug-in like WP Hardening to change the login error text and prevent user enumeration attacks
- Block IPs after too many failed login attempts, by plug-in or IPS
- Add a CAPTCHA to discourage login brute force attacks

Hardening Against Elevated Sudo Rights

- Unless needed, remove Steven’s sudo rights for Python
  - Using visudo, edit the sudoers file to remove those rights
- Where sudo rights are needed to run a script, grant rights to just the script (not all Python code)
  - Use visudo, change Steven’s access from all Python to the needed script

Hardening Against Exposed MySQL credentials

- Change the permissions on sensitive WordPress configuration files to prevent user accounts from accessing them
  - chmod [permissions] [file]

Implementing Patches

Implementing Patches with Ansible

This playbook can be used to implement the following patches:

1. change ssh port
2. hashing passwords, and salting hashes
3. hardening wp_config file

Changing SSH port

To edit the port parameter for SSH we can implement the following playbook:

```
- name: Setup alternative SSH port
  lineinfile:
    dest: “/etc/ssh/sshd_config”
    regexp: “^Port”
    line: Port 2025
  notify: Restart sshd
```

Using sha512 password hashing, and salting hashes

To get a sha512 password hash that also has a random salt, we can use the following playbook:

```
- name: Creating New Users
  hosts: all
  become: true
  gather_facts: false
  vars_files:
    - my_vault.yml
  tasks:
   - name: Create Users
       user:
         name: “{{item}}”
         password: “{{my_password | password_hash(‘sha512’) }}”
     shell: /bin/bash
   loop:
- user 1
- user 2
- user 3…
```

Hardening wp_config

Using the following playbook we can reinforce our wp_config file to make it more difficult for attackers to locate and extract sensitive information:

```
- hosts: servers
  roles:
    - { role: harden-wordpress, installations : … }
        wp_harden_block_uploads_php: true
        wp_harden_block_wpconfig: true
        wp_harden_disable_file_edits: true
        wp_harden_block_log_files: true
```