

# **Final Engagement**

**Attack, Defense & Analysis of a Vulnerable Network**

# Table of Contents

---

This document contains the following resources:



**Network Topology & Critical Vulnerabilities**



**Alerts Implemented**



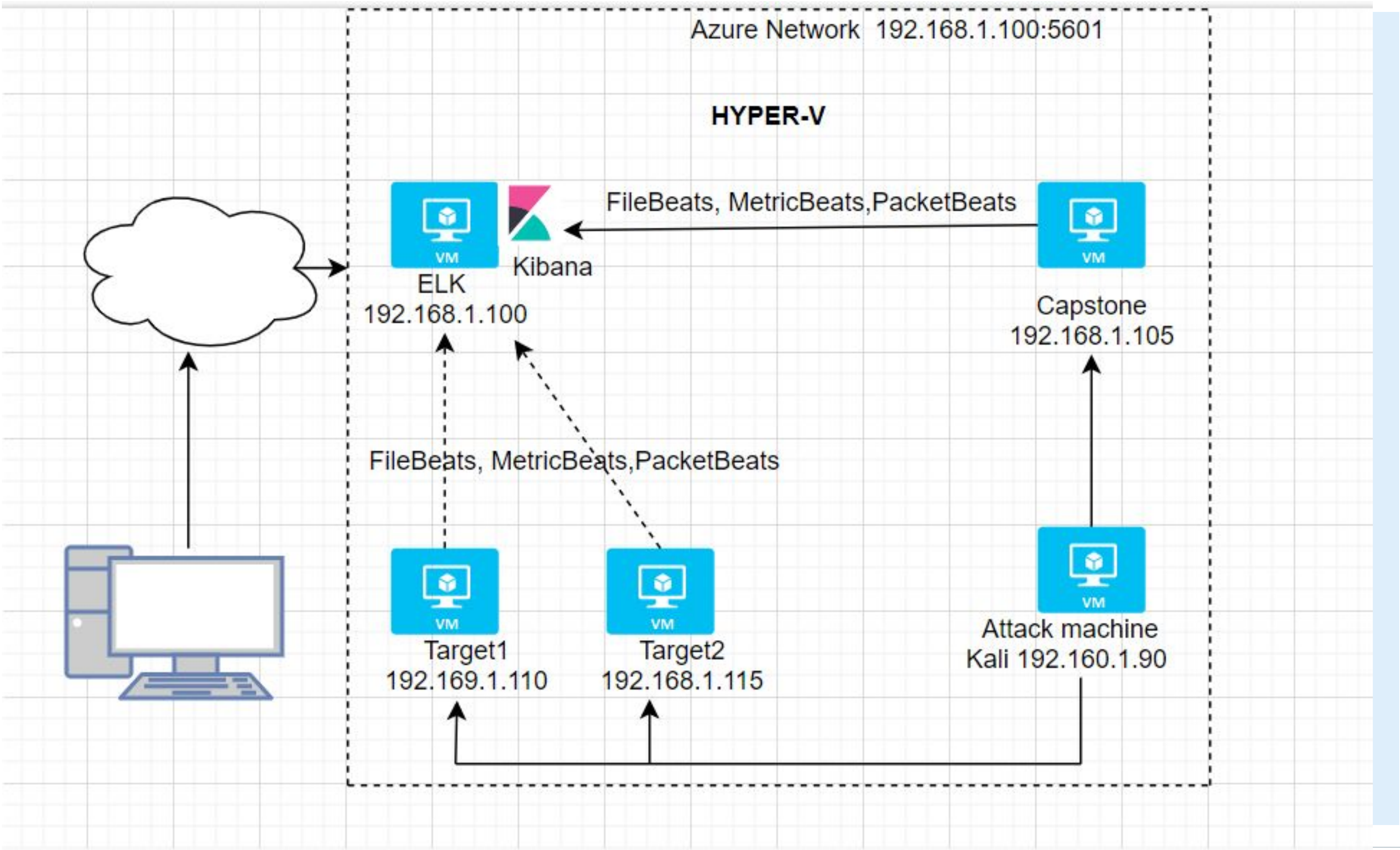
**Hardening**



**Implementing Patches**

# Network Topology & Critical Vulnerabilities

# Network Topology



## Network

Address Range:  
192.168.1.0/24  
Netmask: 255.255.255.0  
Gateway: 10.0.0.1

## Machines:

IPv4: 192.168.1.100  
OS: Ubuntu 18.04.4 LTS  
Hostname: ELK

IPv4: 192.168.1.105  
OS: Ubuntu 10.04.1 LTS  
Hostname: Capstone

IPv4: 192.168.1.90  
OS: Kalu GNU/Linux rolling  
Hostname: Kali

IPv4: 192.168.1.110  
OS: Debian GNU/Linux 8  
Hostname: Target1

IPv4: 192.168.1.115  
OS: Debian GNU/Linux 8  
Hostname: Target2



# Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Weak user Password	short, common words or something that could be rapidly guessed by executing a brute force attack	passwords can be easily cracked and attacker will gain an access to the account.
Unsalted password hash	If a password is not salted it can be cracked by online tools or programs like “John the ripper”	If an attacker already know the username after cracking the password he have a full access to the user account.
Wordpress User Enumeration	an attacker scans a web application to discover the login name of the WordPress based web application	Allows attacker to gather usernames to gain access to the web server.

# Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
MySQL Database Access	I was able to find a file that contains login information for MySQL database	an attacker can gain access to MySQL database .
Privilege Escalation	Used Stevens sudo python -c 'import pty;pty.spawn("/bin/bash")' escalate to root	an attacker is able to gain root privilege



Alerts Implemented

# Excessive HTTP Errors

Summarize the following:

- Which **metric** does this alert monitor? http.response.status\_code
- What is the **threshold** it fires at? The TOP 5 get ABOVE 400 for the last 5 minutes
- Provide a screenshot of the alert in action.

Edit Excessive HTTP Errors

Send an alert when your specified condition is met. Your watch will run every 1 minute.

Name

Excessive HTTP Errors

Indices to query

packetbeat-\*

Time field

event.created

Run watch every

1

minute

Match the following condition

WHEN count() GROUPED OVER top 5 'http.response.status\_code' IS ABOVE 400 FOR THE LAST 5 minutes

No data

Your index and condition did not return any data.

Perform 1 action when condition is met

Add action

> Logging

Current status for 'Excessive HTTP Errors'

Deactivate

Delete

Execution history

Action statuses

Last one hour

Trigger time	State	Comment
2022-06-11T17:11:49+00:00	✓ OK	
2022-06-11T17:10:49+00:00	✓ OK	
2022-06-11T17:09:49+00:00	✓ OK	
2022-06-11T17:08:49+00:00	✓ OK	
2022-06-11T17:07:49+00:00	✓ OK	
2022-06-11T17:06:48+00:00	✓ OK	
2022-06-11T17:05:49+00:00	✓ OK	



# HTTP Request Size Monitor

Summarize the following:

- Which **metric** does this alert monitor? WHEN sum() of http.request.bytes OVER all documents
- What is the **threshold** it fires at?ABOVE 350 FOR THE LAST 1 minute
- Provide a screenshot of the alert in action.

Edit http request size monitor

Send an alert when your specified condition is met. Your watch will run every 1 minute.

Name

http request size monitor

Indices to query

packetbeat-\* x

Time field

@timestamp

Run watch every


1

minute

Match the following condition

WHEN sum() OF http.request.bytes OVER all documents IS ABOVE 350 FOR THE LAST 1 minute

sum()



Perform 1 action when condition is met

Add action

Logging

Save alert

Cancel

Show request

Current status for 'http request size monitor'

Deactivate

Delete

Execution history

Action statuses

Last 7 days

Trigger time	State	Comment
2022-06-11T17:14:49+00:00	✓ OK	
2022-06-11T17:13:49+00:00	✓ OK	
2022-06-11T17:12:49+00:00	✓ OK	
2022-06-11T17:11:49+00:00	✓ OK	
2022-06-11T17:10:49+00:00	✓ OK	
2022-06-11T17:09:49+00:00	✓ OK	
2022-06-11T17:08:49+00:00	✓ OK	
2022-06-11T17:07:49+00:00	✓ OK	
2022-06-11T17:06:48+00:00	✓ OK	
2022-06-11T17:05:49+00:00	✓ OK	

Rows per page: 10

<

1

2

3

4

5

...

108

>

9

# CPU Usage Monitor

Summarize the following:

- Which **metric** does this alert monitor? WHEN max() OF system.process.cpu.total.pct OVER all documents
- What is the **threshold** it fires at? ABOVE 0.5 FOR THE LAST 5 minutes
- Provide a screenshot of the alert in action.

Edit cpu usage monitor

Send an alert when your specified condition is met. Your watch will run every 1 minute.

Name

cpu usage monitor

Indices to query

metricbeat-\* x

Time field

@timestamp

Run watch every

1

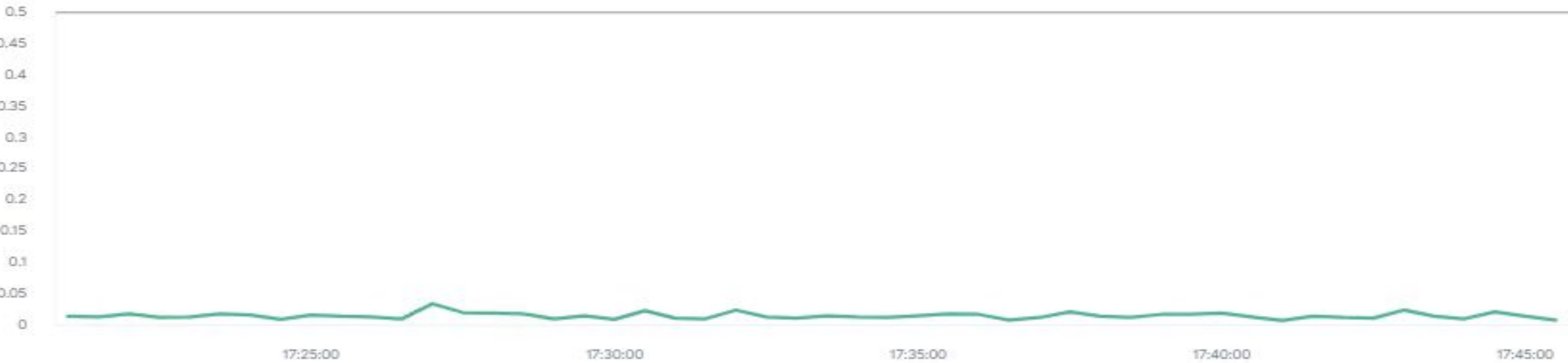
minute

Use \* to broaden your query.

Match the following condition

WHEN max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes

max()



Perform 1 action when condition is met

Add action

> Logging

Save alert

Cancel

Show request

Current status for 'Excessive HTTP Errors'

DeactivateDelete

Execution history

Action statuses

Last 7 days

Trigger time	State	Comment
2022-06-11T17:18:48+00:00	✓ OK	
2022-06-11T17:17:49+00:00	✓ OK	
2022-06-11T17:16:49+00:00	✓ OK	
2022-06-11T17:15:49+00:00	✓ OK	
2022-06-11T17:14:49+00:00	✓ OK	
2022-06-11T17:13:49+00:00	✓ OK	
2022-06-11T17:12:49+00:00	✓ OK	
2022-06-11T17:11:49+00:00	✓ OK	
2022-06-11T17:10:49+00:00	✓ OK	
2022-06-11T17:09:49+00:00	✓ OK	

Rows per page: 10

<

1

2

3

4

5

...

109

>

# Hardening



# Hardening Against Weak User Password on Target 1

---

- Set up lockouts on multiple bad password attempts
- `sudo nano /etc/pam.d/common-auth` and updating the deny setting for lockout
- Institute complex password requirements with expirations on passwords
- This can be accomplished by setting the character requirements in the `/etc/pam.d/common-password` file
- `sudo nano /etc/pam.d/common-password` to set the requirements which includes character requirements `minlen`, `maxrepeat`, `ucredit`, `lcredit`, `dcredit`, `ocredit`, and `difok`.
- Expiration can be set in the `/etc/login.defs`
- `sudo nano /etc/login.defs` to set `Pass_Max_Days` to how many days.



# Hardening Against Unsalted password hash on Target 1

---

Explain how to patch Target 1 against Vulnerability 2. Include:

- Salt all hashes
- Salting works by adding random values to the hashing, making them significantly more difficult to decrypt
- This is implemented at the creation of your password, using the option “-m” you can specify the hashing method
- `mkpasswd -m sha512crypt`

# Hardening Against Wordpress User Enumeration on Target 1

---

Explain how to patch Target 1 against Vulnerability 3. Include:

- To prevent User enumeration, you can edit the functions.php file
- add command

```
if (!is_admin()) {  
    // default URL format  
    if (preg_match('/author=([0-9]*)/i', $_SERVER['QUERY_STRING'])) die(); add_filter('redirect_canonical', 'shapeSpace_check_enum', 10, 2);  
}  
  
function shapeSpace_check_enum($redirect, $request) {  
    // permalink URL format  
    if (preg_match('/\?author=([0-9]*)(</i>)/i', $request)) die(); else return $redirect;  
}
```

- This would check if the request contains an integer and block it if it does

# Hardening Against MySQL Database Access on Target 1

---

- update the file wp-config.php to only be readable by root and the owner of the file.
- `chmod 700 wp-config.php`

# Implementing Patches



# Implementing Patches with Ansible

---

## Playbook Overview

You can use Ansible to script the adding of lines to files and updating lines in files to the desired state. With this, you can script the update of the recommended files in the hardening section.

Example of updating wordpress playbook commands:

```
- name: get latest wordpress
  unarchive:
    src: https://wordpress.org/latest.zip
    dest: /tmp/
    remote_src: yes
    become: true

- name: Wait until wordpress has been downloaded
  wait_for:
    path: /tmp/wordpress/index.php
    state: present

- name: copy wordpress to website
  shell: /bin/cp -rf /tmp/wordpress/* /var/www/html/
  become: true
```