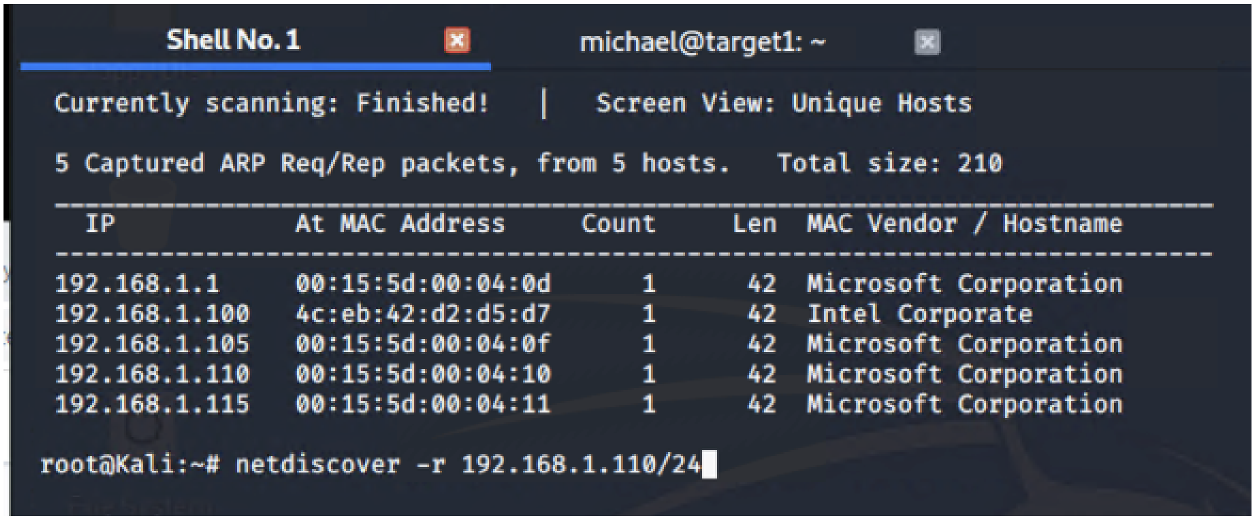
Blue Team: Summary of Operations

Table of Contents

Network Topology

The following machines were identified on the network:

- ELK
  - **Operating System**: Ubuntu 18.04.4 LTS
  - **Purpose**: To set up alerts - Holds Kibana dashboards
  - **IP Address**: 192.168.1.100/24
- Target 1
  - **Operating System**: Windows 6.1 (Samba 4.3.14-Debian)
  - **Purpose**: Vulnerable WordPress server to compromise (SQL databases)
  - **IP Address**: 192.168.1.110/24
- Capstone
  - **Operating System**: Ubuntu 18.04.4 LTS
  - **Purpose**: Test Alerts - Filebeat and Metricbeat are installed and will forward logs to the ELK machine
  - **IP Address**: 192.168.105/24
- Kali
  - **Operating System**: Kali GNU/Linux
  - **Purpose**: Penetration Test Machine
  - **IP Address**: 192.168.1.90/24

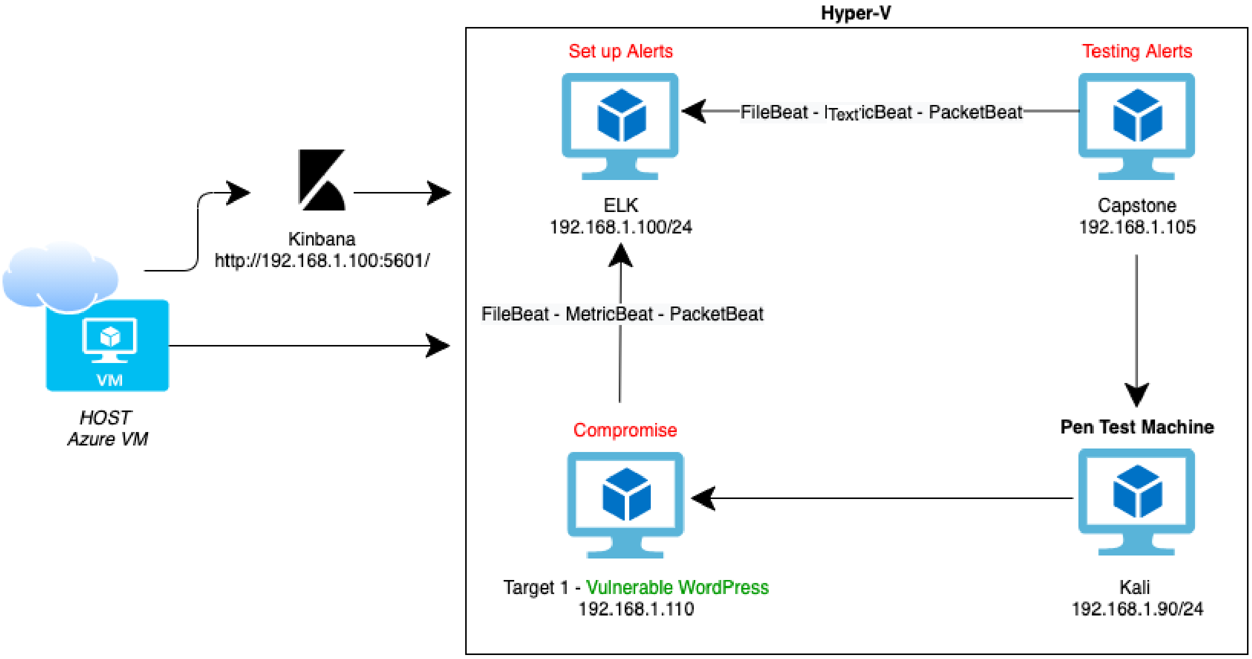


```
 Currently scanning: Finished!   |   Screen View: Unique Hosts

 5 Captured ARP Req/Rep packets, from 5 hosts.   Total size: 210
 _____________________________________________________________________________
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 192.168.1.1     00:15:5d:00:04:0d      1      42  Microsoft Corporation
 192.168.1.100   4c:eb:42:d2:d5:d7      1      42  Intel Corporate
 192.168.1.105   00:15:5d:00:04:0f      1      42  Microsoft Corporation
 192.168.1.110   00:15:5d:00:04:10      1      42  Microsoft Corporation
 192.168.1.115   00:15:5d:00:04:11      1      42  Microsoft Corporation

root@Kali:~# netdiscover -r 192.168.1.110/24
```

The topology diagram below shows Target 1 web server machine which the Blue team will be configuring for system hardening. The pen testing machine Kali, will attack Target 1 and uncover vulnerabilities. ELK VM was configured to set up alerts and the Capstone machine is another VM designed to test the alerts.



Description of Targets

The target of this attack was: Target 1 (IP Address: 192.168.1.110/24).

Target 1 is an Apache web server and has SSH enabled, so ports 80 and 22 are possible ports of entry for attackers. As such, the following alerts have been implemented:

Monitoring the Targets (added the alert to the logs and run watch every 1 minute)

Traffic to these services should be carefully monitored. To this end, we have implemented the alerts below:

1) Aler1: Excessive HTTP Errors

Excessive HTTP Errors is implemented as follows:

- **Metric**: HTTP Errors 'http.response.status_code'
- **Threshold**: Above 400 for the last 5 minutes
- **Vulnerability Mitigated**: Brute Force Attacks
- **Reliability**: high reliability, the alerts were triggered several times by brute force attacks. This rule is reliable with acceptable false positives

2) Aler2: HTTP Request Size Monitor

HTTP Request Size Monitor is implemented as follows:

- **Metric**: http.response.bytes OVER all documents
- **Threshold**: IS ABOVE 3500 FOR THE LAST 1 minute
- **Vulnerability Mitigated**: Denial of Service Attacks
- **Reliability**: high reliability, similar to alert 1 above

3) Alert3: CPU Usage Monitor

CPU Usage Monitor is implemented as follows:

- **Metric**: system.process.cpu.total.pct
- **Threshold**: OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes
- **Vulnerability Mitigated**: CPU usage/DoS by scripts overwhelming the system
- **Reliability**: medium reliability, it is a good rule in conjunction with the other alerts

Suggestions for Going Further (Optional)

- Each alert above pertains to a specific vulnerability/exploit. Recall that alerts only detect malicious behavior, but do not stop it. For each vulnerability/exploit identified by the alerts above, suggest a patch. E.g., implementing a blocklist is an effective tactic against brute-force attacks. It is not necessary to explain *how* to implement each patch.

The logs and alerts generated during the assessment suggest that this network is susceptible to several active threats, identified by the alerts above. In addition to watching for occurrences of such threats, the network should be hardened against them. The Blue Team suggests that IT implement the fixes below to protect the network:

- Vulnerability 1 - **Excessive HTTP Errors**
  - **Patch**: To prevent Brute Force Attacks a security package like **SSHGuard** may be used. Install SSHGuard with *sudo apt-get install sshguard*
  - **Why It Works**: SSHGuard blocks users after a few unsuccessful attempts, then releases the lock after some time. It also protects other services like sendmail, exim, dovecot, vsftpd, etc.

https://www.unixmen.com/prevent-brute-force-attacks-using-these-tools/

- Vulnerability 2 - **HTTP Request Size Monitor**
  - **Patch**: To prevent DDOS attacks a tool name **mod_evasive Apache** may be used. Install mod_evasive Apache in several steps:
    - Install Apache Web Server Utility: sudo apt update and s*udo apt-get install apache2-utils*
    - Install mod_evasive: s*udo apt-get install libapache2-mod-evasive*
  - **Why It Works**: mod_evasive works by monitoring incoming server requests watching for suspicious activity like: many requests for the same page in one second, more than 50 simultaneous requests per second, and request form a blacklisted IP address

https://www.devmanuals.net/install/ubuntu/ubuntu-12-04-lts-precise-pangolin/install-apache2-utils.html

- Vulnerability 3 - **CPU Usage Monitor**
  - **Patch**: To limit CPU usage, a tool named cpulimit can be used. Install by following these steps: sudo apt-get update and *sudo apt-get install cpulimit*
  - **Why It Works**: cpulimit is a tool that limits the CPU usage of a process by percentage. Our alert in this case was set up to trigger an alarm when usage reached more than 50%. This tool is useful to control batch jobs. The goal of cpulimit is to prevent a process from running more than the specified time ratio.

https://www.howtoforge.com/how-to-limit-cpu-usage-with-cpulimit-on-ubuntu-linux