# **Blue Team: Summary of Operations**

## **Table of Contents**

* Network Topology
* Description of Targets
* Monitoring the Targets
* Patterns of Traffic & Behavior
* Suggestions for Going Further

### **Network Topology**

*TODO: Fill out the information below.*

The following machines were identified on the network:

* Name of VM 1 Kali
  + **Operating System**:Kali Linux
  + **Purpose**:Attacking Machine
  + **IP Address**:192.168.1.90
* Name of VM 2 Target 1
  + **Operating System**: Linux
  + **Purpose**: Target Machine 1
  + **IP Address**: 192.168.1.110
* Name of VM 3 Target 2
  + **Operating System**: Linux
  + **Purpose**: Target Machine 2
  + **IP Address**: 192.168.1.110
* Name of VM 4 Capstone Server
  + **Operating System**: Ubuntu Linux
  + **Purpose**: Web server
  + **IP Address**: 192.168.1.105
* Name of VM 5 ELK Server
  + **Operating System**: Ubuntu Linux
  + **Purpose**: monitoring and logging Capstone server
  + **IP Address**: 192.168.1.100

### **Description of Targets**

*TODO: Answer the questions below.*

The target of this attack was: Target 1 (192.168.1.110).

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**

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

#### **Excessive HTTP errors**

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

Alert 1 is implemented as follows:

* **Metric**: WHEN count() GROUPED OVER top 5 'http.response.status\_code'
* **Threshold**: Above 400
* **Vulnerability Mitigated**: Enumeration/Brute Force
* **Reliability**: This alert has a high reliability as a high frequency rate of error codes 400 and above, is highly unlikely to occur through normal use and traffic. Especially 400 error codes are of concern here as they indicate client and server errors.

#### **HTTP Request Size Monitor**

Alert 2 is implemented as follows:

WHEN sum() of http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute

* **Metric**: WHEN sum() of http.request.bytes OVER all documents
* **Threshold**: Above 3500
* **Vulnerability Mitigated**: Code injection through HTTP requests
* **Reliability**: This alert could be considered medium reliable as it has the potential to trigger false positives when large non-malicious HTTP requests are made to the server.

#### **CPU Usage monitor**

Alert 3 is implemented as follows:

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

* **Metric**: WHEN max() OF system.process.cpu.total.pct OVER all documents
* **Threshold**: Above 0.5
* **Vulnerability Mitigated**: Malware running in the background using up CPU power
* **Reliability**: This alert has a low reliability as large programs or services running on the server could easily trigger false positives.

### **Suggestions for Going Further (Optional)**

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:

* HTTP Request Errors (Brute Force)
  + **Patch**: *Wordpress hardening.*
    - *Install regular updates*
    - *Make use of Wordpress plugins suchs as Wordpress firewall and backup*
    - *Password protect your Wordpress admin Directory*
    - *Implement two-factor authentication*
  + **Why It Works**:
    - Installing regular updates will minimise the risk of having known vulnerabilities exploited
    - Using security plugins will help protect your Wordpress server
    - Requiring two-factor authentication will make it harder for an attacker to brute force their way into your server
    - Protecting your admin directory and logins will provide an added layer of defense in case of a breach
* HTTP request size monitor (Code Injection)
  + **Patch**:
    - Install regular updates
    - Limit HTTP requests to web server in size and/or length
    - Make use of input validation
  + **Why It Works**:
    - Keeping your application up to date will minimise the risk of k known vulnerabilities being exploited
    - Limiting the size of the HTTP requests will prevent overly large requests from going through, thus preventing this type of attack.
    - Input validation of your forms help prevent any malicious data or scripts from being uploaded through your site to the server.
* Vulnerability 3
  + **Patch**:
    - Implement a host based intrusion system (HIDS)
    - Invest in good anti-malware software and ensure it stays up to date.
  + **Why It Works**:
    - A HIDS will scan for malware and suspicious activity specifically on your machine, not just on the network. This means that it will be able to detect and locate malware running on your machine and prevent it from altering or infecting local files and/or data.
    - A robust anti-malware program will allow you to more easily detect, and prevent, any malware intrusions into your system.