# Blue Team: Summary of Operations

## Table of Contents

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

## Network Topology

TODO: Fill out the information below.

The following machines were identified on the network:

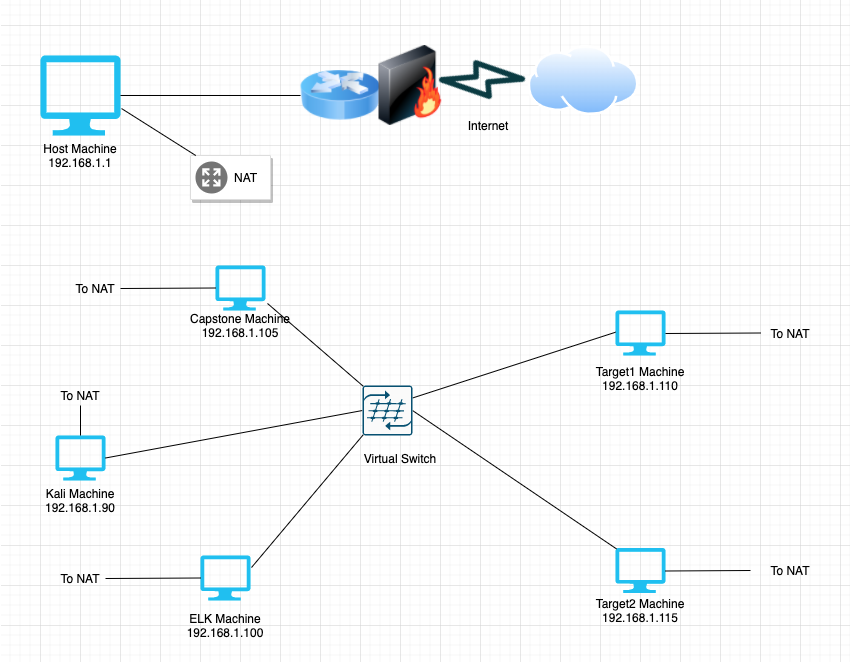
**[Target 1]**

* Operating System: Debian 8
* Purpose: WebServer
* IP Address: 192.168.1.110

**[Target 2]**

* Operating System: Debian 8
* Purpose: WebServer
* IP Address: 192.168.1.115

Including a Gliffy or draw.io diagram is optional but highly encouraged.



**Network**

Address Range: 192.168.1.0/24

Netmask: 255.255.255.0

Gateway: 192.168.1.1

**Machines**

IPv4: 192.168.1.1

OS: Windows

Hostname: ML-RefVm-684427

IPv4: 192.168.1.100

OS: Linux

Hostname: ELK

IPv4: 192.168.1.105

OS: Linux

Hostname: Capstone

IPv4: 192.168.1.90

OS: Linux 2.6.32

Hostname: Kali

IPv4: 192.168.1.110

OS: Windows

Hostname: Target1

Description of Targets

Fill in the following:

* Two VMs on the network were vulnerable to attack: Target 1 [TODO: 192.168.1.110] and Target 2 [TODO: 192.168.1.115].
* Each VM functions as an Apache web server and has SSH enabled, so ports 80 and 22 are possible ports of entry for attackers.

## Monitoring the Targets

This scan identifies the services below as potential points of entry:

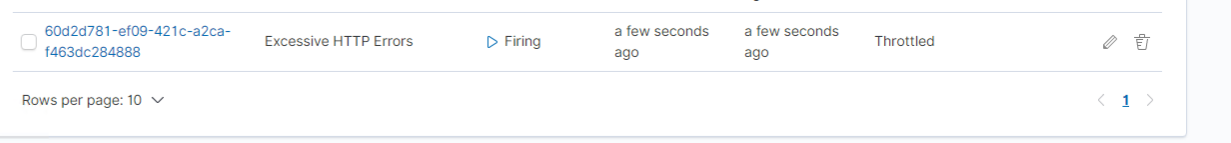
* **Target 1**
  + List of
  + Potentially vulnerable
  + Services
* **Target 2**
  + List of
  + Potentially vulnerable
  + Services

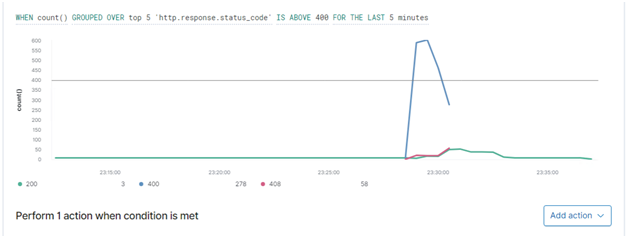
Traffic to these services should be carefully monitored. To this end, we have implemented the alerts below: (Note: Add at least three alerts. You can add more if time allows.)

**Excessive HTTP Errors**

Excessive HTTP Errors is implemented as follows:

* Metric: http.response.status.code
* Threshold: Grouped over top 5 for the last five minutes
* Vulnerability Mitigated: Denial of Service Attack or Brute Force attacks
* Reliability: This alert is highly reliable and fired during nmap, wpscan, gobuster and nikto scans.



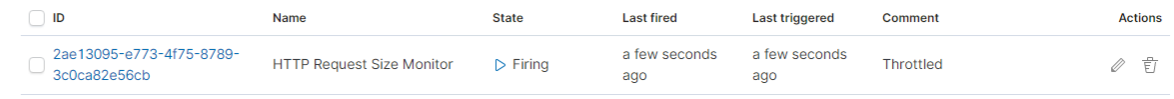


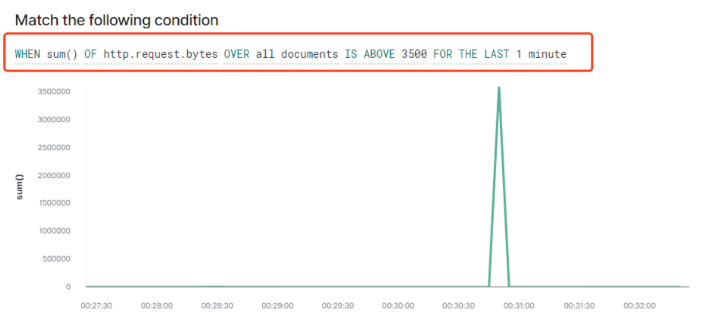
**HTTP Request Size Monitor**

HTTP Request Size Monitor is implemented as follows:

* Metric: http.response.body.bytes
* Threshold: sum of bytes is above 3500 for a minute
* Vulnerability Mitigated: Payload Delivery Attempt
* Reliability: Highly reliable. Had to use a kali application called slowhttptest to fire the alert. See command below



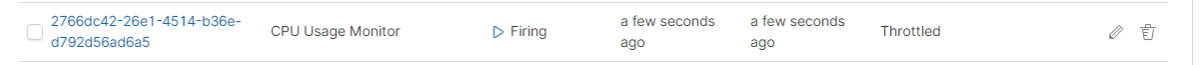


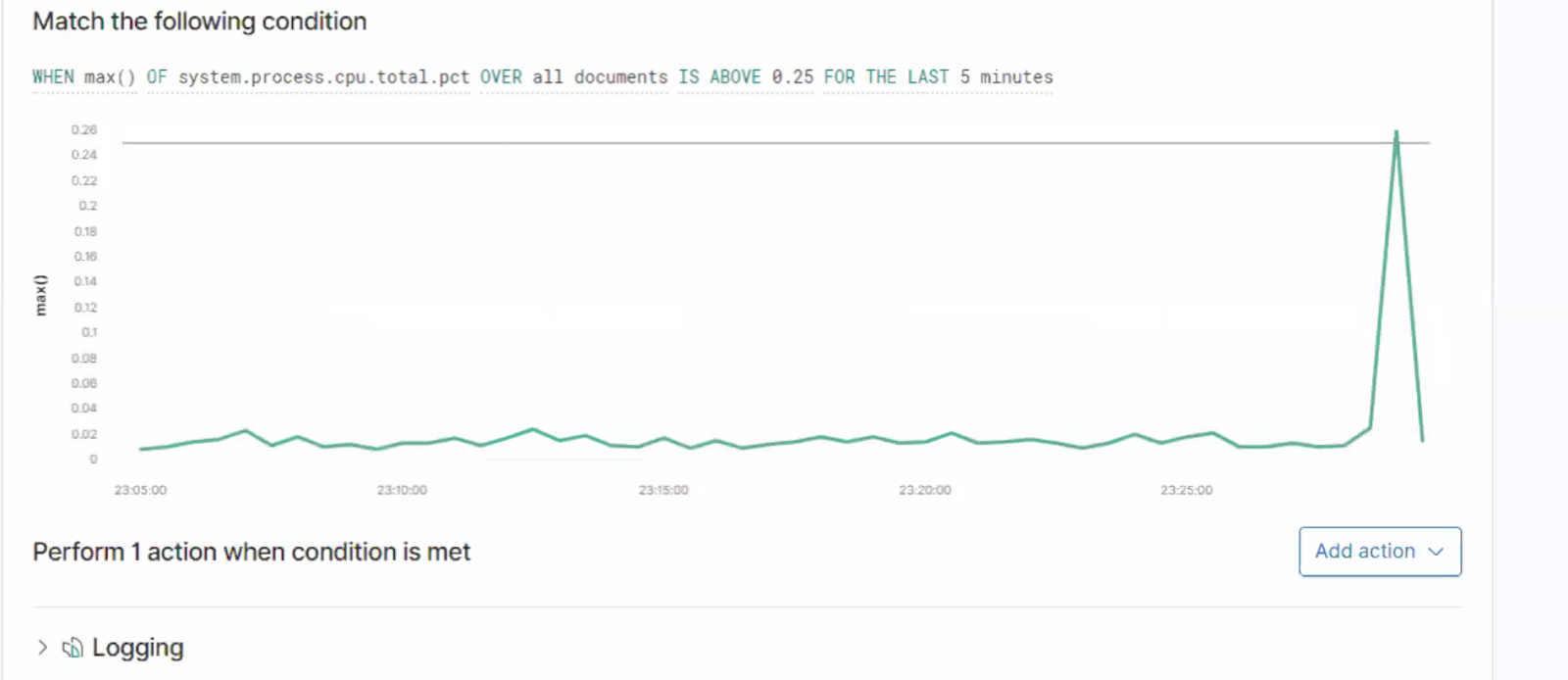


**CPU Usage Monitor**

CPU Usage Monitor is implemented as follows:

* Metric: system.process.cpu.total.pct
* Threshold: Reaches a max of 25% in the last five minutes
* Vulnerability Mitigated: Denial of Service or a backdoor attack.
* Reliability: There were false negatives on this alert when it was originally set to 50%. All the offensive tools used on the target caused spike up above 25% but below 50% which is why the threshold was changed to 25%.





## Suggestions for Going Further

**Suggest a patch for each vulnerability identified by the alerts above.** Remember: alerts only detect malicious behavior. They do not prevent it.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. 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**

* Patch: Immediately force Michael's password to expire and deny michael as an ssh user.
* Why It Works: It works because it closes that attack vector immediately and michael will be forced to change his password when he tries to login next.
* Ansible Playbook Entry:

-name: Expire Michael's Password & Force him to change it

command: passwd -e michael

-name: Disable ssh for Michael

command: echo “Deny Users michael” >> /etc/ssh/sshd\_config

-name: Restart ssh service

systemd:

name: ssh

state: reloaded

**Vulnerability 2**

* Patch: Changing Password Complexity
* Why It Works: It works against brute force dictionary- based attacks as we did with joh the ripper.
* Ansible Playbook Entry:

-name: Change Password Policy

lineinfile:

path: /etc/security/pwquality.conf

regexp: ‘{{item.From}}’

line: ‘{{item.To}}’

state: present

with\_items:

- { From: ‘minlen: 9’, To: ‘minlen: 12’}

- { From: ‘dcredit: 1’, To: ‘dcredit: 2’}

- { From: ‘lcredit: 1’, To: ‘lcredit: 2’}

- { From: ‘ucredit: 1’, To: ‘ucredit: 2’}

**Vulnerability 3**

* Patch: Setup a firewall to deny all traffic before enabling ports 80 and 22 only.
* Why It Works: This works because it prevents leaving open ports to implement a backdoor listener as we did using port 4444 with netcat.
* Ansible Playbook Entry:

-name: Deny everything and enable UFW

ufw:

state: enabled

policy: deny

-name: Allow ssh and http

ufw:

rule: allow

port: ‘80’

port: ‘22’

**Vulnerability 4**

* Patch: Uninstall the python pty module
* Why It Works: This works because it prevents root privilege escalation using the python script.
* Ansible Playbook Entry:

-name: Uninstall python terminals

-pip:

extra\_args: uninstall

port: ‘80’

name: pty