# **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**

The following machines were identified on the network:

* Host Computer
  + **Operating System**: Microsoft Windows 2008 XP
  + **Purpose**: The host computer that houses Hyper-V and VMs for the project.
  + **IP Address**: 192.168.1.1/24
* Kali
  + **Operating System**: Kali Linux
  + **Purpose**: The attackers machine that was used to PenTest the Target1 and Target 2 machines.
  + **IP Address** : 192.168.1.90/24
* ELK Stack
  + **Operating System**: Intel Computer
  + **Purpose**: The ELK Stack machine that acts as the Defensive’s team resource to set up alerts.
  + **IP Address**: 192.168.1.100/24
* Capstone
  + **Operating System**: Microsoft Corporation
  + **Purpose**: To test alerts and provide FileBeat, MetricBeat, and PacketBeat to the ELK Stack machine (192.168.1.1).
  + **IP Address:** 192.168.1.105/24
* Target 1
  + **Operating System**: Microsoft running a Linux Kernel
  + **Purpose**: The vulnerable WordPress machine that is open and accessible for attacks.
  + **IP Address** : 192.168.1.110/24

### **Description of Targets**

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**

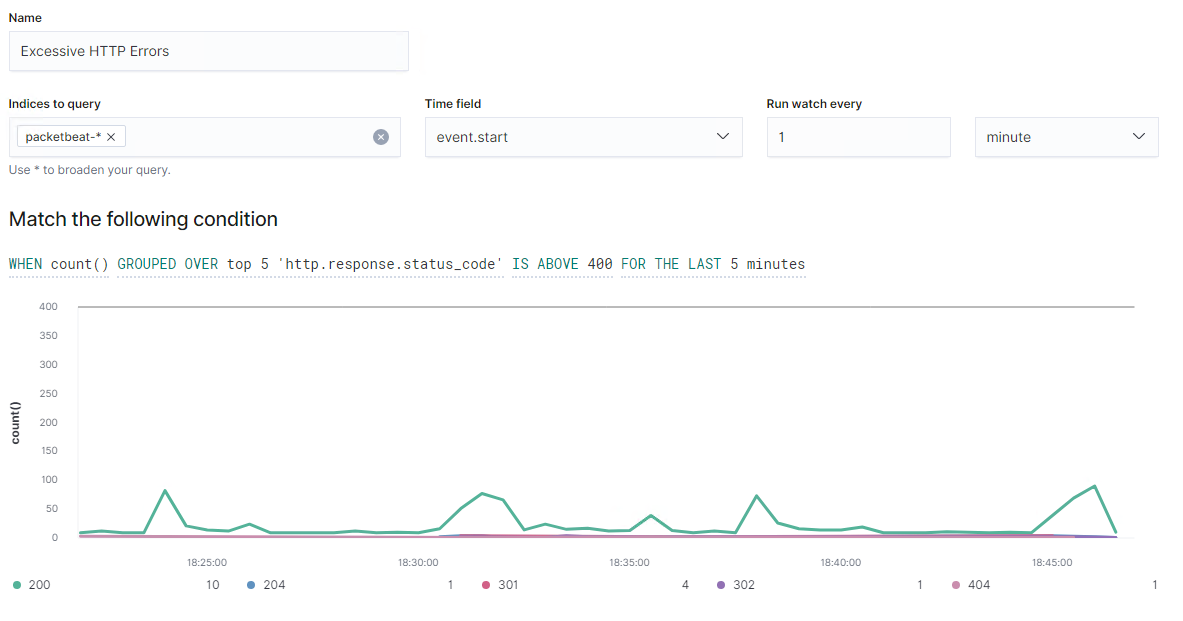
Traffic to these services should be carefully monitored. To this end, we have implemented the alerts below:

#### **Excessive HTTP Errors**

This alert monitors the number of HTTP Errors doled out from the server in 5 minute increments. If HTTP Errors are included in the Top 5 HTTP Responses, an alert will trigger.

Excessive HTTP Errors is implemented as follows:

* **Metric**: http.response.status\_code
* **Threshold**: Top 5 HTTP response codes contain response codes greater than 400 for the last 5 minutes
* **Vulnerability Mitigated**: Brute Force Attack
* **Reliability**: Medium Reliability. This alert has a medium reliability since we configured the alert to trigger if the top 5 HTTP response codes contain responses greater than 400 (which comprise HTTP error codes) within five minutes. This much HTTP Error Code activity would notify the SOC team of suspicious activity and a potential brute force attack. However, there could be instances where error codes are being generated that are not connected to cyberattacks.

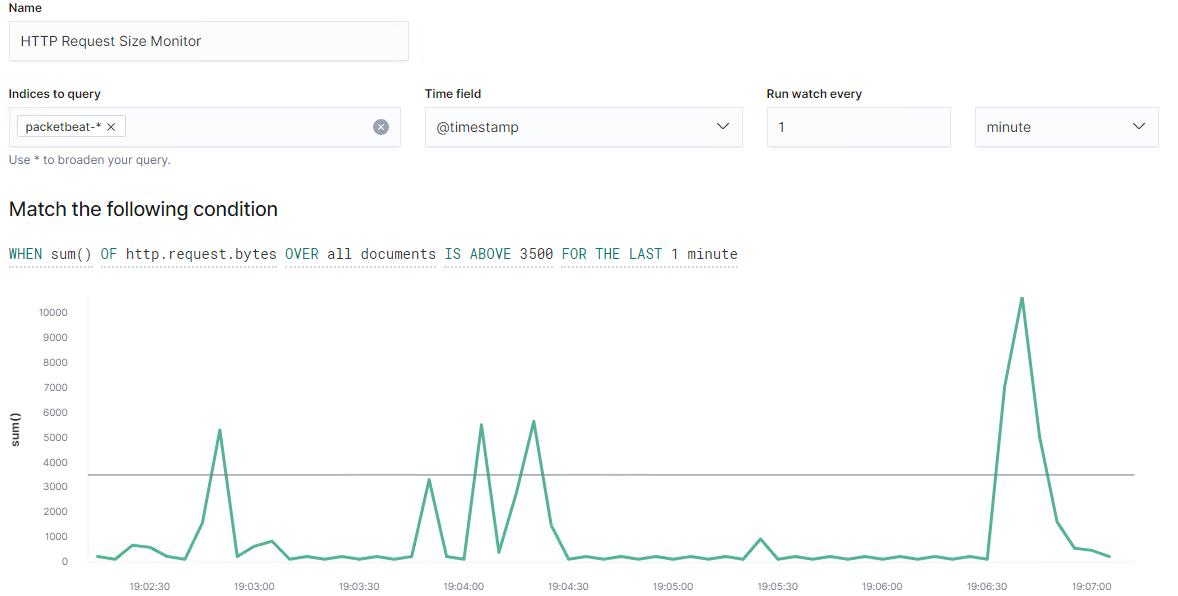


#### **HTTP Request Size Monitor**

This alert monitors the number of HTTP requests being sent to and attempting to access the server.

HTTP Request Size Monitor is implemented as follows:

* **Metric**: http.request.bytes
* **Threshold**: ABOVE 3500 FOR THE LAST 1 minute
* **Vulnerability Mitigated**: DoS Attack, DDoS Attack
* **Reliability**: High Reliability. This alert has a high threshold set above 3,500 HTTP Requests within one minute. It is unlikely there will be false positives since that much activity is abnormal.

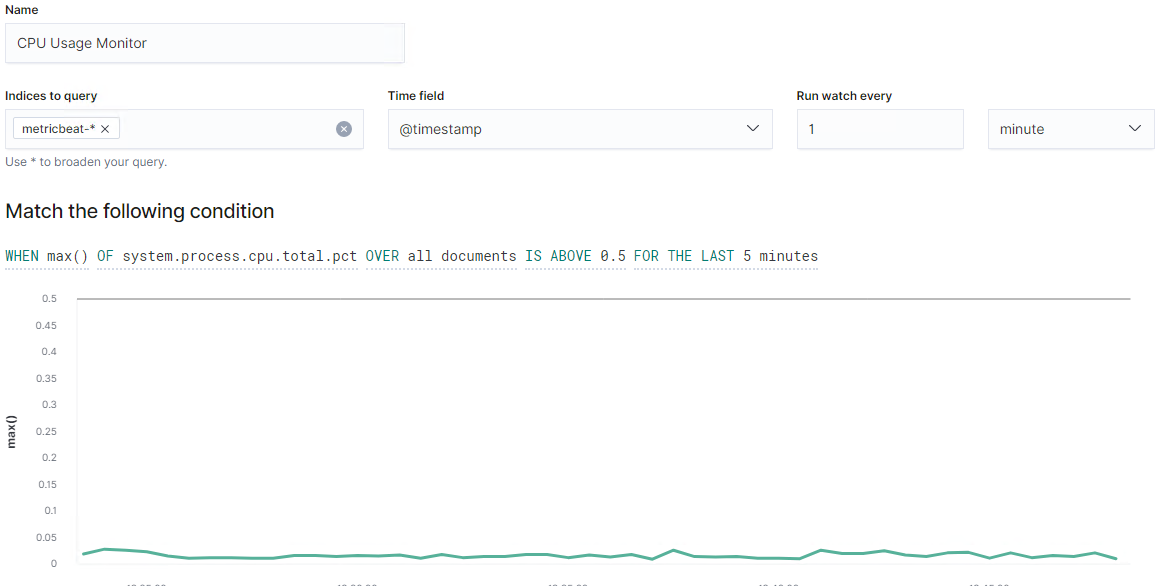


#### **CPU Usage Monitor**

This alert monitors the percentage of CPU spent by a system’s processes. If CPU usage goes above 50%, an alert will be triggered.

CPU Usage Monitor is implemented as follows:

* **Metric**: system.process.cpu.total.pct
* **Threshold**: ABOVE 0.5 FOR THE LAST 5 minutes
* **Vulnerability Mitigated**: Malware
* **Reliability**: Low Reliability. This alert helps to determine if malware is running on the machine, but this alert could also trigger false positives since high CPU usage could be the result of many scenarios such as outdated drivers, lengthy downloads, and more.



### **Suggestions for Going Further**

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: Malware
  + **Patch**: Blacklist Malware Hashes (Blocking Blacklist and Create Alerts)
  + **Why It Works**: A blacklist of malware hashes (unique identifiers) will be set up to block most common malware attacks. Instead of using CPU Usage rates as a reliable alert indicator of malware, alerts should be focused on hash field. Alerts will be triggered when a blacklist of hashes are attempting to be utilized. If time and resources are available a whitelist can be created to prevent an unknown hashes to interact with the network, although this is incredibly expensive and time consuming.
* Vulnerability 2: HTTP Floods (DoS/DDoS Attempts)
  + **Patch**: Size Limits on HTTP Headers and Limit Request Rate of Request Sources:
    - HTTP header size limits can be adjusted or added to the *application.properties* file. (*server.max-http-header-size=[*desired#*]*)
    - IPs that surpass HTTP request thresholds within a specific time frame to any path under the domain name will be blocked.
  + **Why It Works**: HTTP floods occur when an attacker exploits legitimate HTTP GET or POST requests to attack a web server or application. This vulnerability is difficult to patch because they can run below rate-based thresholds. Placing size limits on HTTP headers constricts what can be received despite speeds, while blocking IPs sending rapid HTTP requests will prevent a D/Dos attack. Finally, setting connection timeout restrictions can help prevent slow HTTP flood attempts.
* Vulnerability 3: Brute Force Attack through Unlimited SSH Attempts: These conditions can be created when an unauthorized remote attacker attempts to make many SSH connections.
  + **Patch**: Limiting Port Access Attempts from External Devices within a Specific Timeframe: Inbound Port Rule [Windows] or *sudo apt-get install ufw* *&& sudo ufw enable*[Linux]
  + **Why It Works**: Limiting port access attempts (example: 3-6 attempts within 30 seconds) from external devices/IPs prevents rapid, brute force attacks allowing the attacker to try making many SSH connections in an attempt to gain access to a user’s account. This can also create conditions for a DoS vulnerability if too many connection attempts are being made in a short period of time, triggering high CPU usage alerts.