

# **Final Engagement**

**Attack, Defense & Analysis of a Vulnerable Network**

**Prepared by Dmitriy Baimakov on December 2nd, 2021**

# Table of Contents

---

This document contains the following resources:



**Network Topology & Critical Vulnerabilities**



**Alerts Implemented**



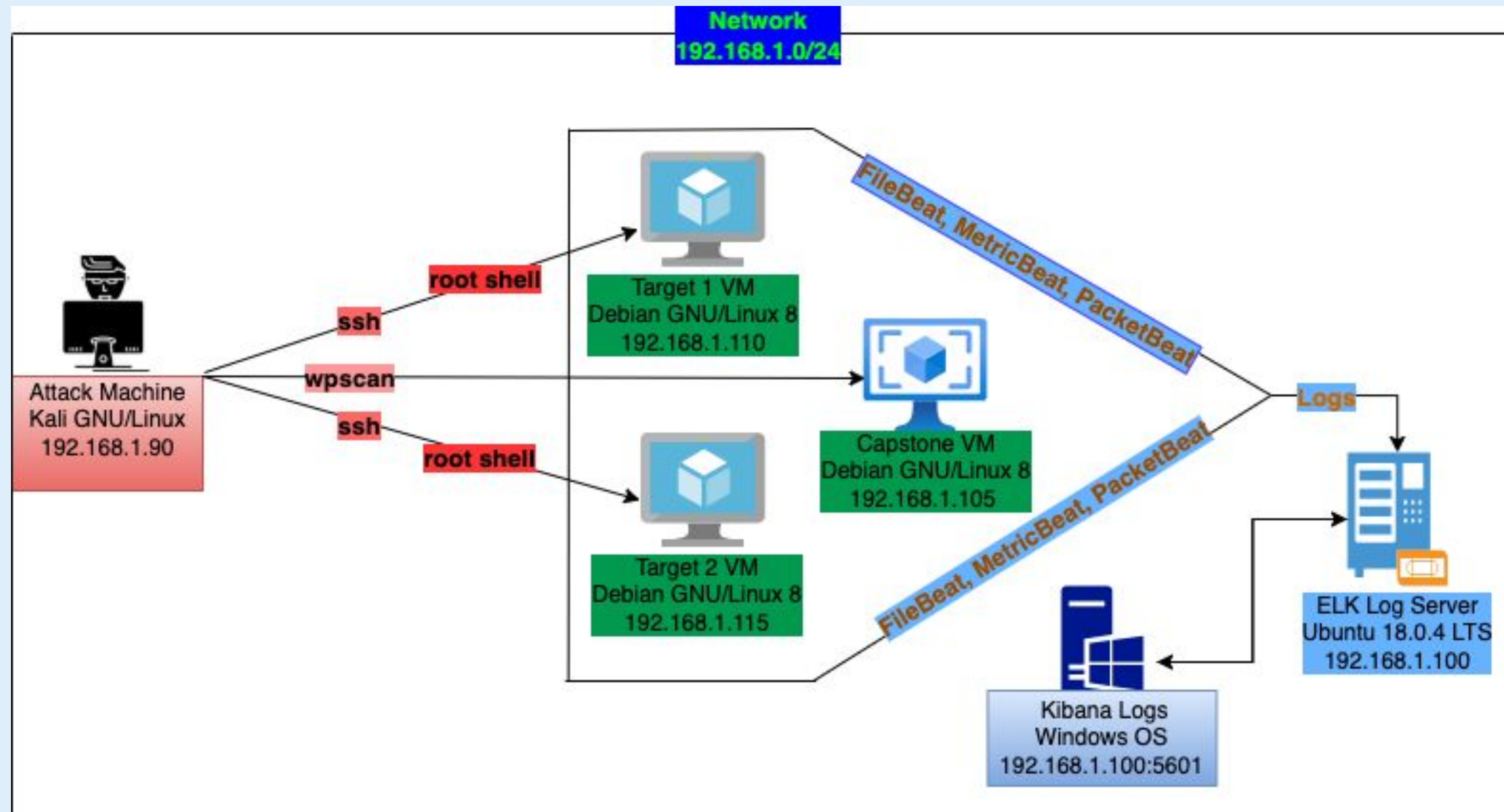
**Hardening**



**Implementing Patches**

# 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: 192.168.1.90  
OS: Linux  
Hostname: Kali

IPv4: 192.168.1.110  
OS: Linux  
Hostname: Target 1

IPv4: 192.168.1.115  
OS: Linux  
Hostname: Target 2

IPv4: 192.168.1.105  
OS: Linux  
Hostname: Capstone

IPv4: 192.168.1.100  
OS: Ubuntu  
Hostname: ELK



# Vulnerabilities: Target 1

---

Our assessment uncovered the following vulnerabilities in the **Target 1**

Vulnerability	Description	Impact
Open access via port 22	Vulnerable due to unfiltered access to port 22, which increases the number of various services to be exploited	Attackers can configure the network and exploit programs running on this exposed port.
wordpress mysql access, using WordPress Enumeration	Vulnerable to the use of wp-scan enumeration with the following access gain to mysql database	This vulnerability enumerates user names that can be used by the attack to gain ssh access, and obtaining secret hashes from wordpress database
Root Shell escalation	Vulnerable to sudo -l and python escalation command	Potential for gaining root shell user privileges
DDoS attack <a href="#">CVE:2014-5266</a>	Vulnerable to msfconsole's DDoS	Can be used to slow down and crash the website

# Vulnerabilities: Target 1

---

Vulnerability	Description	Impact
Authentication bypass vulnerability	During recon we discovered an and enumerated the users. Hydra was used to brute force.	Obtained passwords for Michael and Steven to ssh.
Brute Force with WPScan against wordpress via XML-RPC <a href="#">CVE:2020-28036</a>	Accompanied with a wpscan for enumeration, there is enough information to brute force with the same tool.	Allows attackers to gain privileges by using the xmlrpc.php file to make post requests by sending usernames and passwords for authentication. Obtained passwords for Michael and Steven.
Weak Passwords	Michael has a simple account password.	A threat actor can brute force a simple password in a matter of seconds, gaining access to sensitive data.
Apache httpd allows remote attackers to read secret data from process memory. <a href="#">CVE:2017-9798</a>	The attacker sends an unauthenticated OPTIONS HTTP request when attempting to read secret data.	Secret data can be accessed. Can result in an Optionsbleed.



Alerts Implemented

# CPU Usage Monitor

## CPU usage Monitor Watcher Alert:

- This alerts monitors system.process.cpu.total.pct using metricbeat logs
- To avoid false positive flags threshold of >70% is used

CPU Usage Monitor

Indices to query

metricbeat-\* X

Time field

@timestamp

Run watch every

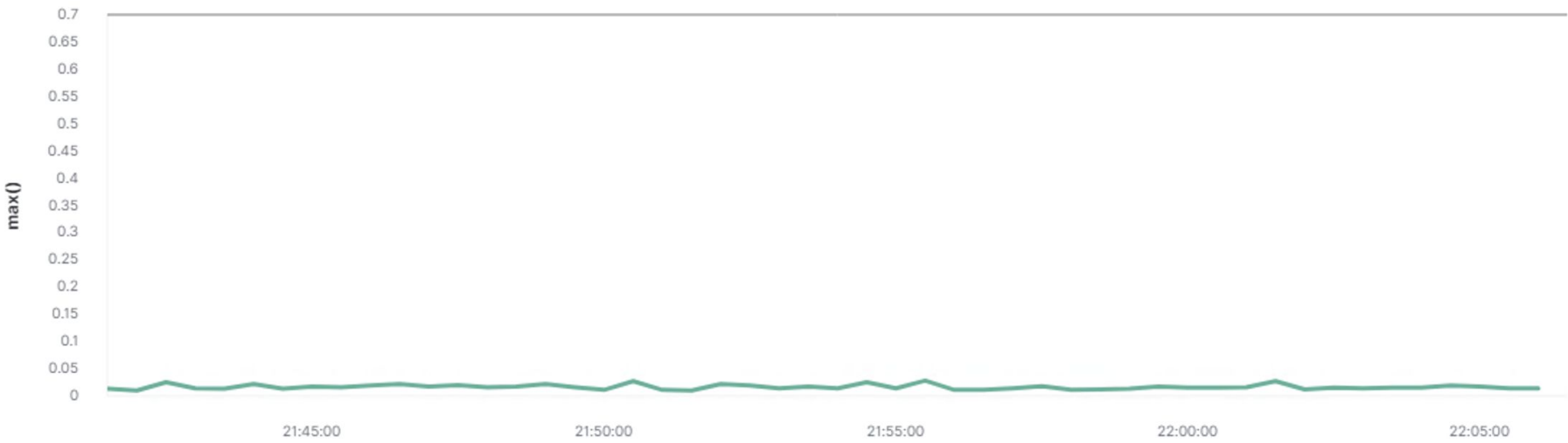
1

minute

Use \* to broaden your query.

Match the following condition

WHEN max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.7 FOR THE LAST 5 minutes



Perform 1 action when condition is met

Add action

☒ Logging

Log text

Watch [{{ctx.metadata.name}}] has exceeded the threshold

Log a sample message

## Watcher

Watch for changes or anomalies in your data and take action if needed.

[Watcher docs](#)

Q Search...

Create

ID	Name	State	Last fired	Last triggered	Comment	Actions
<input type="checkbox"/> fc29e5bb-98a0-4316-98f6-f156ce90dd06	CPU Usage Monitor	✓ OK	5 hours ago	a few seconds ago		<a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> 647cf8cf-83b5-4c1d-b900-1d4d2d2078be	HTTP Request Size Monitor	✓ OK	3 minutes ago	a few seconds ago		<a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> 56038dcd-d469-4977-8a3c-ca3b9f68b675	Excessive HTTP Errors	✓ OK	an hour ago	a few seconds ago		<a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> ce947a78-9ed4-433f-a28b-5822887e80bc	Port Scan Alert	▶ Firing	a few seconds ago	a few seconds ago		<a href="#">Edit</a> <a href="#">Delete</a>

Rows per page: 10

< 1 >

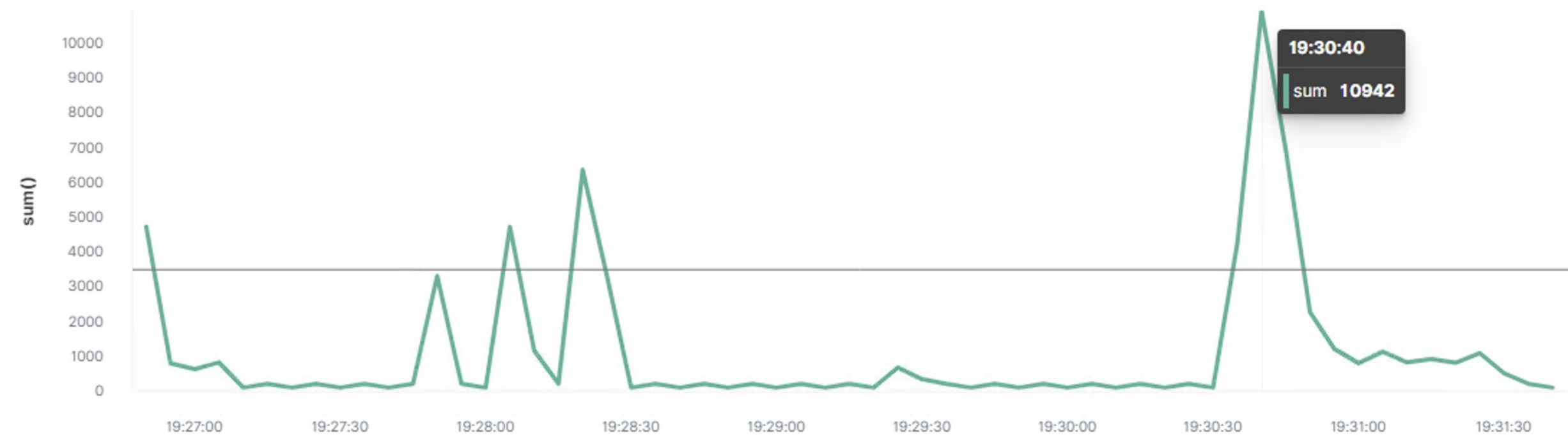


# HTTP Request Size Monitor

## HTTP Request Size Monitor Specifications:

- This alert uses http.request.bytes metric using packetbeat's logs
- Threshold is set to >3500 per minute to mitigate against potential Bruteforce and DDoS attacks

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



### Edit HTTP Request Size Monitor

Send an alert when your specified condition is met. Your watch will run every 1 minute.

Name

HTTP Request Size Monitor

Indices to query

packetbeat-\*

Time field

@timestamp

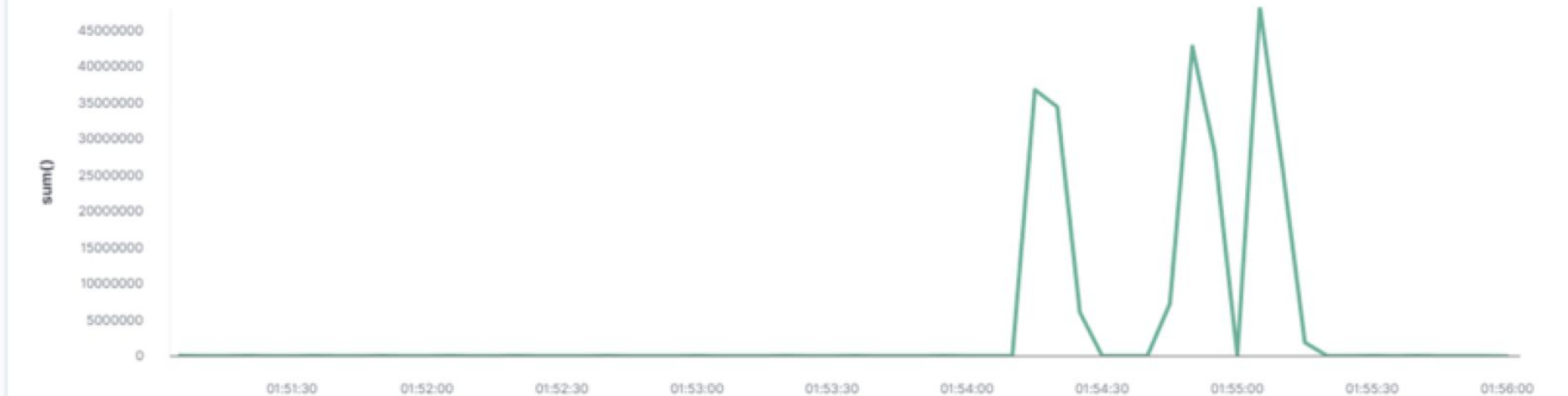
Run watch every

1

minute

Match the following condition

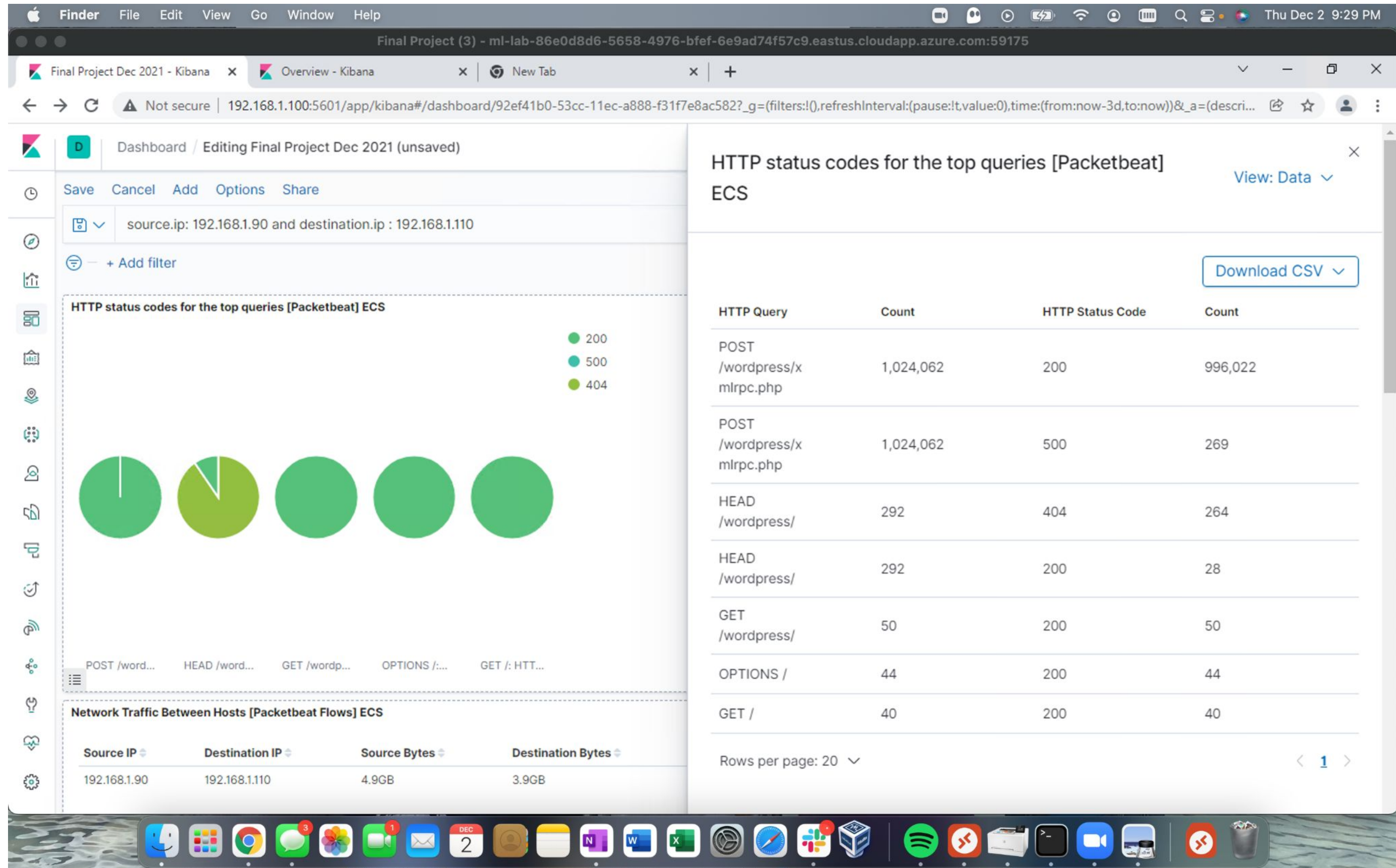
WHEN sum() OF http.request.bytes OVER all documents IS ABOVE 10000 FOR THE LAST 1 minute



Perform 1 action when condition is met

Add action

# HTTP request





# Excessive HTTP Errors

Excessive HTTP Errors specifications:

- This alert checks http.response.status\_code for any alert that returns Errors 400 and above
- Alert triggers when any response error has a tag of 400 and above

Edit Excessive HTTP Errors

Send an alert when your specified condition is met. Your watch will run every 1 minute.

Name

Excessive HTTP Errors

Indices to query

packetbeat-\*

Time field

@timestamp

Run watch every

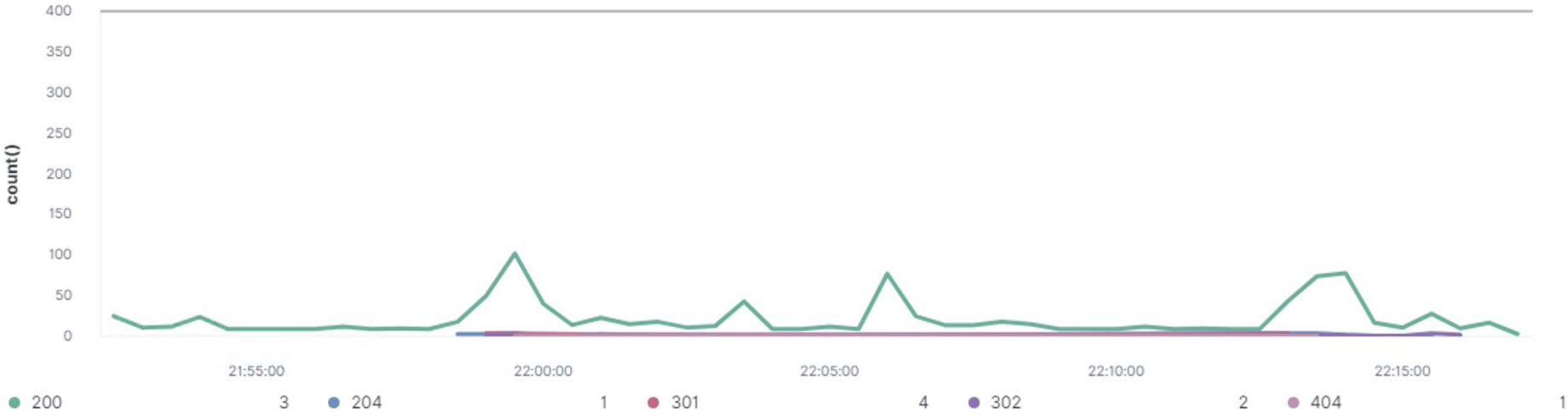
1

minute

Use \* to broaden your query.

Match the following condition

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



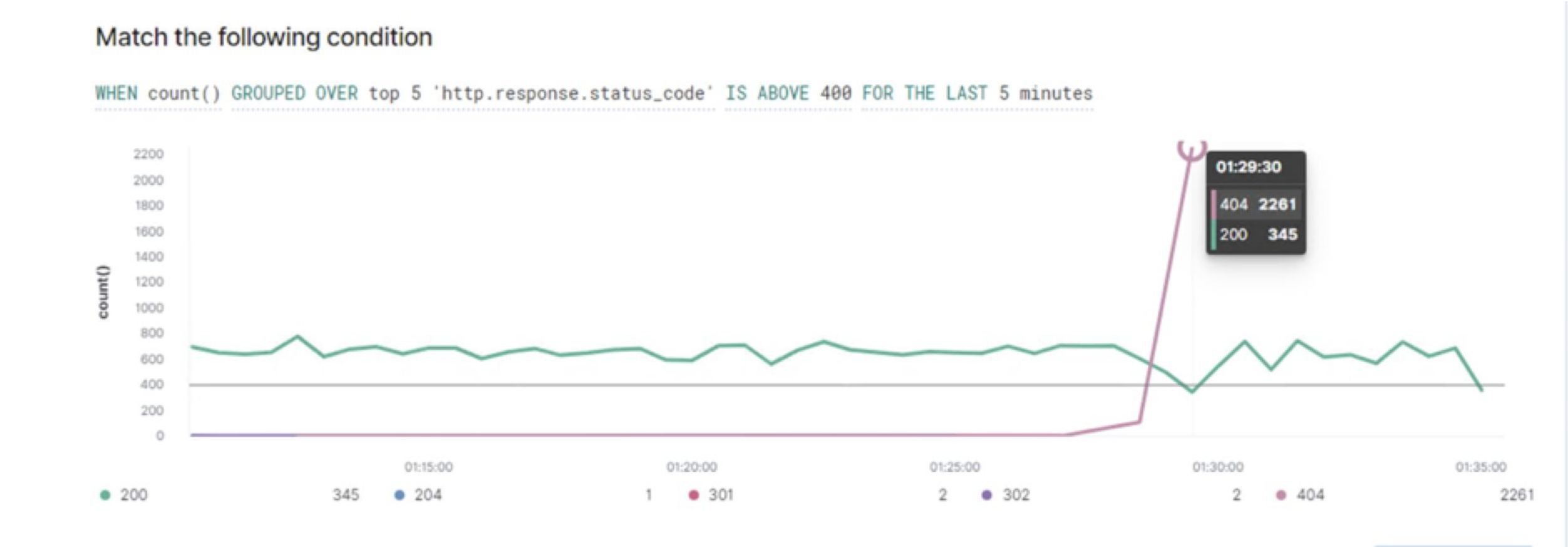
Perform 1 action when condition is met

Add action

Logging

Log text

Watch {{{ctx.metadata.name}}} has exceeded the threshold



# Hardening



# Hardening Against SSH logings on Target 1

---

To patch the exposed port 22 vulnerability on Target 1 the following could be implemented:

- Disable non root users from being able to ssh in by editing with **nano /etc/ssh/sshd\_config**, search for Port and set it to something random like port 899
- Use TCP Wrappers by configuring two files; first **nano -w /etc/hosts.allow**, replace sshd : your secure IP
- Disable root login by editing sshd\_config file, search for PermitRootLogin no

```
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin no_
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile      %h/.ssh/authorized_keys
```

# Hardening Against DDoS CVE:2014-5266, 7494 on Target 1

---

To patch against WordPress and DDoS vulnerabilities

- Earlier versions of samba is vulnerable to remote code execution, allowing a malicious client to upload a shared library to a writable share, and then cause the server to load and execute which can initiate a DDoS attack
- Patch the WordPress services to the current versions and disable XML-RPC api settings
- Patch the Samba client to the current version

# Hardening Against File Editing and File Access on Target 1

---

To prevent file to be edited and accessed implement the following:

- Block access to wp-config.php by changing file permissions with **find**  
**/path/to/your/wordpress/install/ -type d -exec chmod 755 {} \;** for files **find**  
**/path/to/your/wordpress/install/ -type f -exec chmod 644 {} \;**
- Set WordPress to perform an automatic update

# Implementing Patches



# Implementing Patches with Ansible

---

## Playbook Overview

Explain which vulnerability each task in the playbook patches.