Final Engagement
Attack, Defense & Analysis of a Vulnerable Network

Table of Contents

This document contains the following resources:

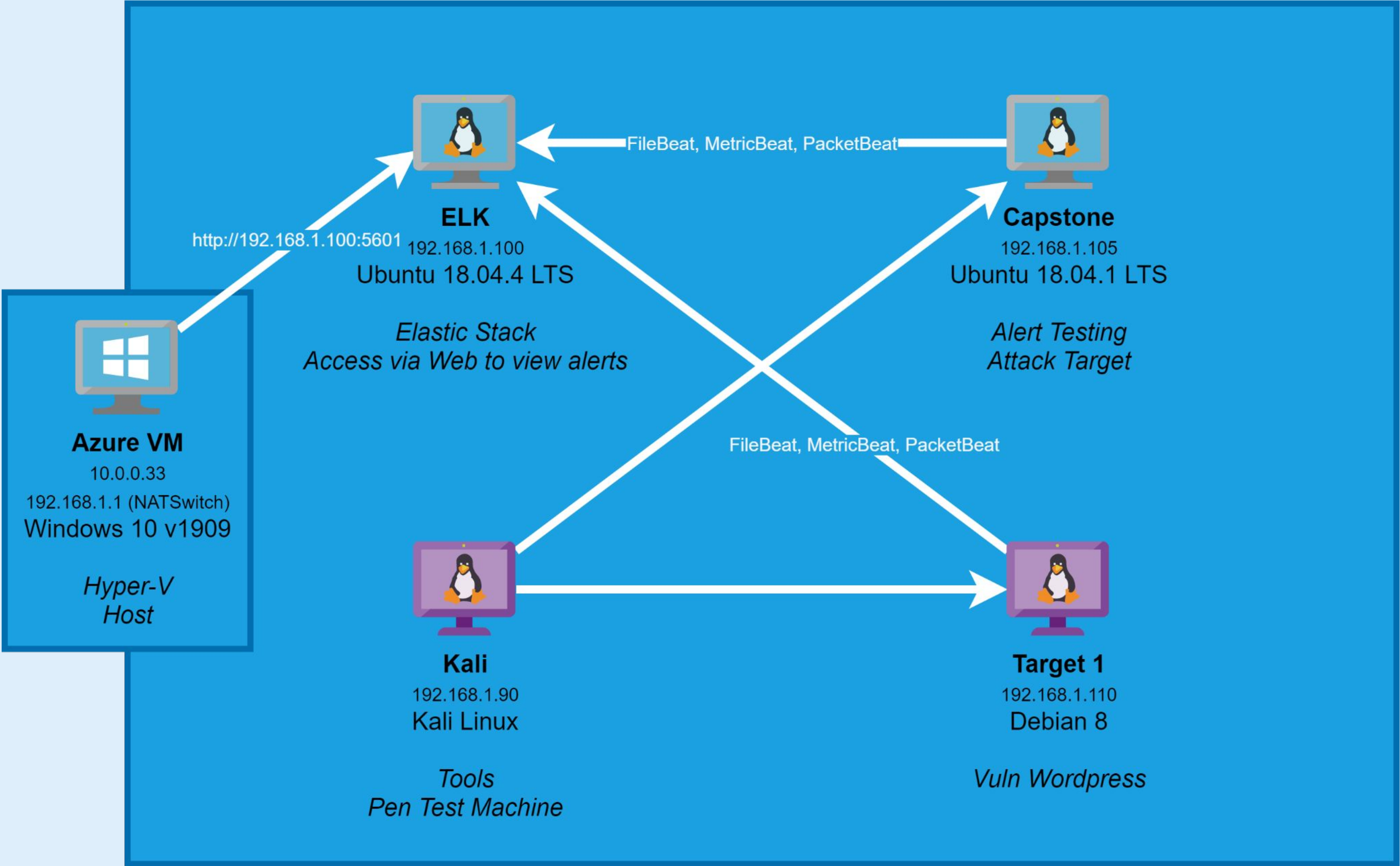**Network Topology & Critical Vulnerabilities**

Alerts Implemented

Hardening

Implementing Patches

Network Topology
& Critical Vulnerabilities

Network Topology

Critical Vulnerabilities: Target 1

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

| Vulnerability	Description	Impact
CWE-200: Exposure of Sensitive Information to an Unauthorized Actor **Port Scanning** Src: https://cwe.mitre.org/data/definitions/200.html	The product exposes sensitive information to an actor that is not explicitly authorized to have access to that information. Some kinds of sensitive information include: private, personal information, such as personal messages, financial data, health records, geographic location, or contact details, system status and environment, such as the operating system and installed packages, business secrets and intellectual property, network status and configuration, the product's own code or internal state, metadata, e.g. logging of connections or message headers, indirect information, such as a discrepancy between two internal operations that can be observed by an outsider.	Port scanning allows a malicious actor to identify where the targets weakness is for entry, and list potential exploits that can be used.
CWE-521: Weak Password Requirements Src: https://www.cvedetails.com/cwe-details/521/Weak-Password-Requirements.html	Authentication mechanisms often rely on a memorized secret (also known as a password) to provide an assertion of identity for a user of a system. It is therefore important that this password be of sufficient complexity and impractical for an adversary to guess.	Passwords that are easily guessed can lead to unauthorised access into a system or network and expose sensitive data.
CWE-284: Improper Access Control Src: https://www.cvedetails.com/cwe-details/284/Access-Control-Authorization-Issues.html	The software does not restrict or incorrectly restricts access to a resource from an unauthorized actor. Access control involves the use of several protection mechanisms such as authentication, authorization and accountability.	When any mechanism is not applied or otherwise fails, attackers can compromise the security of the software by gaining privileges, reading sensitive information, executing commands, evading detection, etc.

Alerts Implemented
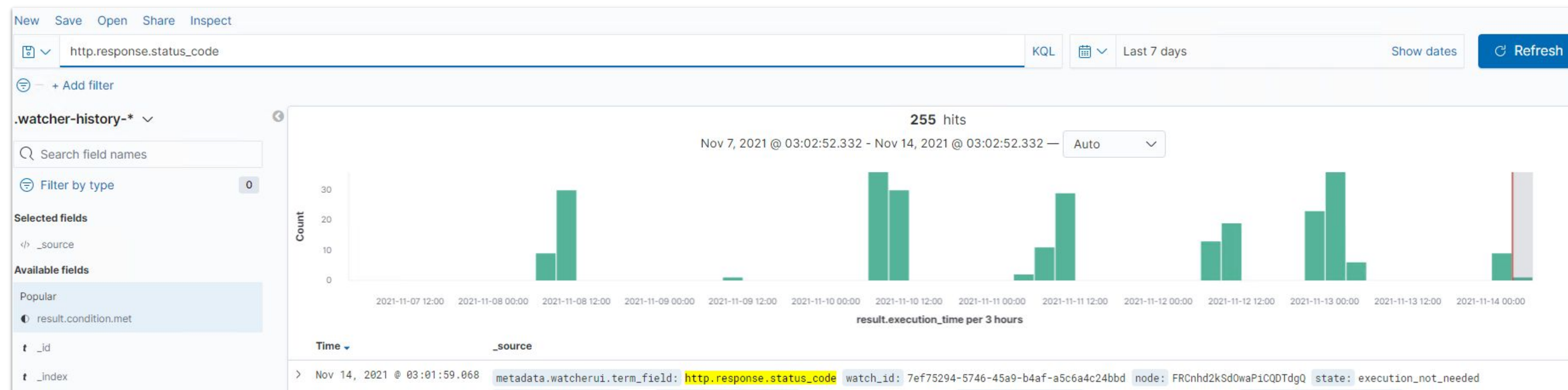
HTTP Request Size Monitor

Summarize the following:

- Which **metric** does this alert monitor? http.request.bytes
- What is the **threshold** it fires at? Above 3,500 for the last minute
- Provide a screenshot of the alert in action.
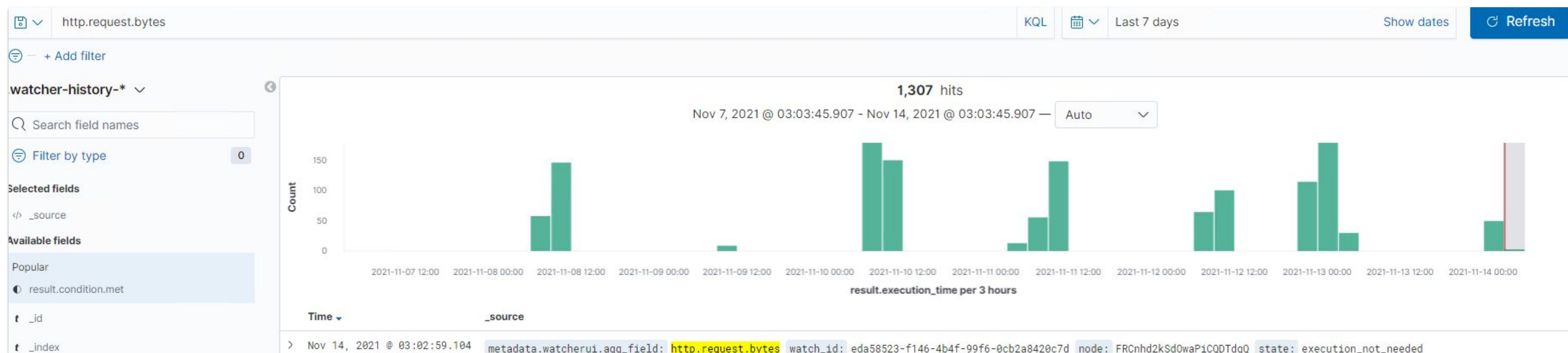
Excessive HTTP Errors

Summarize the following:

- Which **metric** does this alert monitor? http.response.status_code
- What is the **threshold** it fires at? Above 400 for the last 5 minutes
- Provide a screenshot of the alert in action.

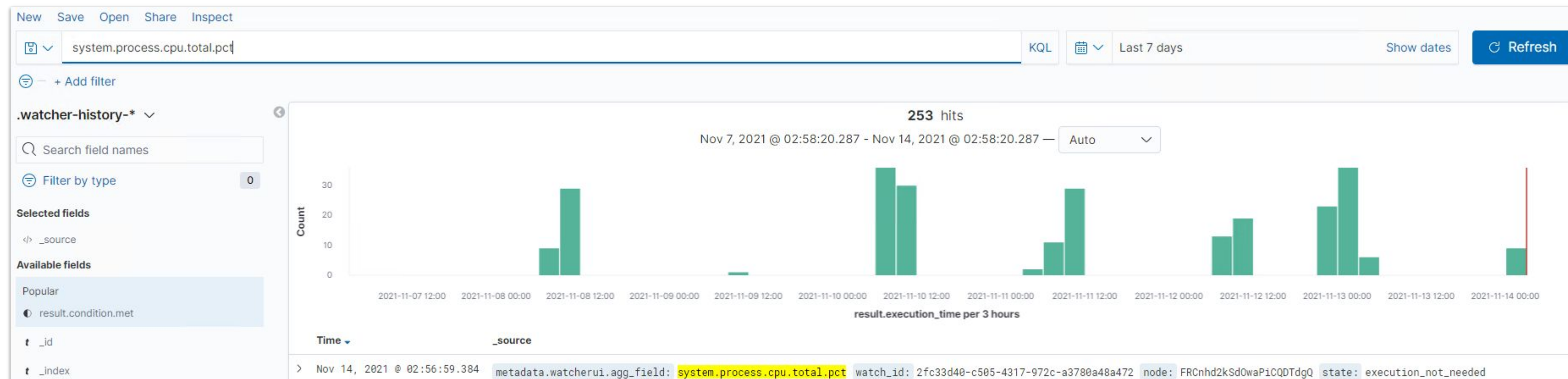# CPU Usage Monitor

Summarize the following:

- Which **metric** does this alert monitor? system.process.cpu.total.pct

- What is the **threshold** it fires at? Above 0.5 for the last 5 minutes

- Provide a screenshot of the alert in action.

Hardening

Hardening Against Weak Authentication on Target 1

Explain how to patch Target 1 against Weak Authentication. Include:

PASSWORD POLICY:

- Create a strong, Long Passphrase including the use of upper/lowercase, special characters and numeric characters.
- Mandatory periodic password change
- Passwords cannot contain the username or any identifiable details of the user.
- Enforce a password history policy to eliminate the repeated use of a password.
- Suspend Accounts after 45 days without a valid login
- Limit failed Login attempts
- Monitor Failed Login Attempts

ADDITIONAL LAYERS OF AUTHENTICATION:

- Implement Multi-Factor Authentication
- RSA Hard Token
- CAPTCHA tools to differentiate between real users and automated users
- Use of Password Managers (LastPass)

Account lockout/Authentication Policy hardening.

- This will make it more difficult for an intruder to circumvent the authentication process.

- Implementation of the Password Policy includes but is not to limited to the following;
  - To set the minimum password length:
    - *sudo nano /etc/pam.d/common-password*
      - *password [success=2 default=ignore] pam_unix.so obscure sha512* **minlen=12**
  - *To enforce complexity of passwords:*
    - *sudo apt-get install libpam-pwquality*
    - *sudo nano /etc/pam.d/common-password*
      - To set 1 character as uppercase:
        - *password   requisite   pam_pwquality.so retry=3 ucredit=-1*
  - To set expiry dates:
    - sudo nano /etc/login.defs
      - *PASS_MAX_DAYS 100*
      - *PASS_MIN_DAYS   0*
      - *PASS_WARN_AGE 7*

Hardening Against SSH authentication Vulnerability on Target 1

- Implement SSH only via SSH-key and not password
  - SSH-key will only allow administrators with the SSH-key access to the server
    - Command to disable SSH via password
      - *sudo vi /etc/ssh/sshd_config*
        - then set

          ChallengeResponseAuthentication to no
          PasswordAuthentication to no
          UsePAM to no
          PermitRootLogin to no
          PermitRootLogin to prohibit-password

- Setting up firewall rule - allow only SSH from a list of whitelisted IP addresses
  - The firewall rule will only allow access from the administrators IP address
    - Command to set up firewall to allow whitelisted IP address
      - *sudo ufw allow from [admin's IP address] proto tcp to any port 22*

Hardening Against folder enumeration Vulnerability on Target 1

Explain how to patch Target 1 against Vulnerability 3. Include:

With the implementation of POLP (Principle of Least Privilege) the access to the folders would be restricted and by which Michael wouldn't be able to access wp-config.php where database username and password were found.

- Why the patch works. - As it reduces the attack surface, and only allows required access, it would have eliminated the exposure of mysql root credentials.

- How to install it (include commands)

  - chmod 400 [filename/foldername]

    - chmod 400 wp-config.php

Implementing Patches

Implementing Patches with Ansible

Playbook Overview

```
- name: Run a script to patch(free form)

    ansible.builtin.script: /patch.sh
```

```
GNU nano 2.2.6                                          File: ./patch.sh

#!/bin/bash

# set variables
wp_location="/var/www/html/wordpress"
stevens_ip="192.168.1.136" #picked a random one for the example

#############################
# Fix for Weak Authntication #
#############################

# backup the files
cp -f /etc/pam.d/common-password /etc/pam.d/common-password.bak
cp -f /etc/login.defs /etc/login.defs.bak

# create the updated stream
cat /etc/pam.d/common-password | sed -E 's/pam_unix.so obscure sha512/am_unix.so obscure sha512 minlen=8/g' | sed -E 's/pam_permit.so/pam_permit.so minlen=12 lcredit=1 ucredit=1 dcredit=2 ocredit=1/g' > /etc/pam.d/common-password.new
# cat /etc/login.defs | sed -E 's/PASS_MAX_DAYS */PASS_MAX_DAYS 100 /g' > /etc/login.defs.new

sed -i -e '/PASS_MAX_DAYS/s/99999/90/' /etc/login.defs

# replace the originals
mv -f /etc/pam.d/common-password.new /etc/pam.d/common-password
#mv -f /etc/login.defs.new /etc/login.defs


#############################
# Fix for SSH Authentication #
#############################

# Change SSH deamon settings
sed -i -e '/ChallengeResponseAuthentication/s/yes/no/' /etc/ssh/ssd_config
sed -i -e '/PasswordAuthentication/s/yes/no/' /etc/ssh/ssd_config
sed -i -e '/UsePAM/s/yes/no/' /etc/ssh/ssd_config
sed -i -e '/PermitRootLogin/s/yes/no/' /etc/ssh/ssd_config
#sed -i -e '/PermitRootLogin/s//no/' /etc/ssh/ssd_config

# Configure UFW rules
sudo ufw allow from $stevens_ip proto tcp to any port 22

#################################
# Fix for wp-config file permissions #
#################################

chmod 400  $wp_location/wp-config.php
```

15