Blue Team: Summary of Operations

Submitted by: Katerina Alenicheva

Table of Contents

Network Topology

The following machines were identified on the network:

Kali

- Operating System: Debian Kali 5.4.0
- Purpose: Attacking VM
- IP Address: 192.168.1.90

ELK

- Operating System: Ubuntu 10.04
- Purpose: The ELK Stack (Elasticsearch and Kibana)
- IP Address: 192.168.1.100

Capstone

- Operating system: Ubuntu 18.04
- Purpose: The Vulnerable Web Server
- IP Address: 192.168.1.105

Target 1

- Operating System: Debian GNU/Linux 8
- Purpose: The WordPress Host
- IP Address: 192.168.1.110

Target 2

- Operating system: Debian GNU/Linux 8
- Purpose: The WordPress Host
- IP Address: 192.168.1.110

Description of Targets

The target of this attack was: Target 1 192.168.1.110 and Target 2 192.168.1.115 with better security hardening and exploited differently.

Target 1 and Target 2 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

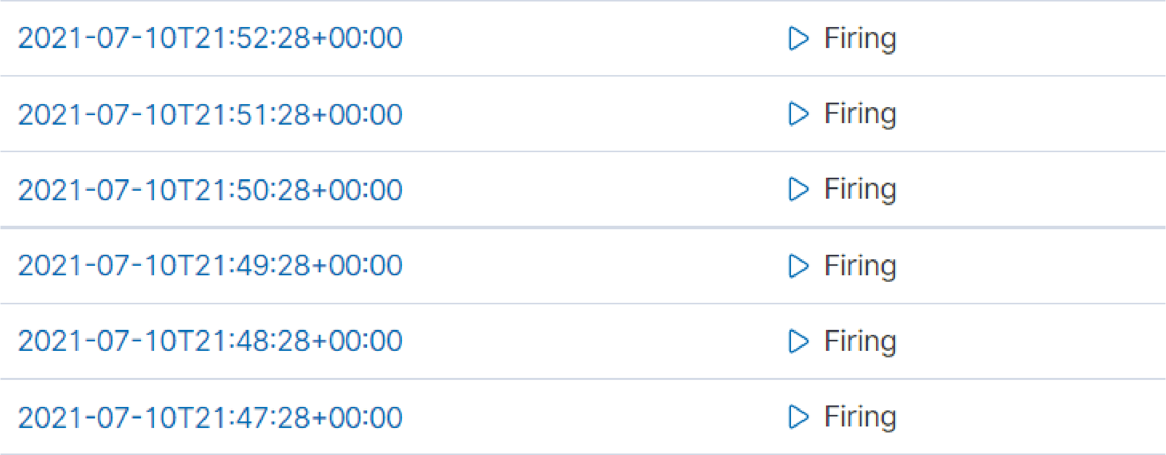
Traffic to these services should be carefully monitored. To this end, we have implemented the alerts below:

Excessive HTTP Errors

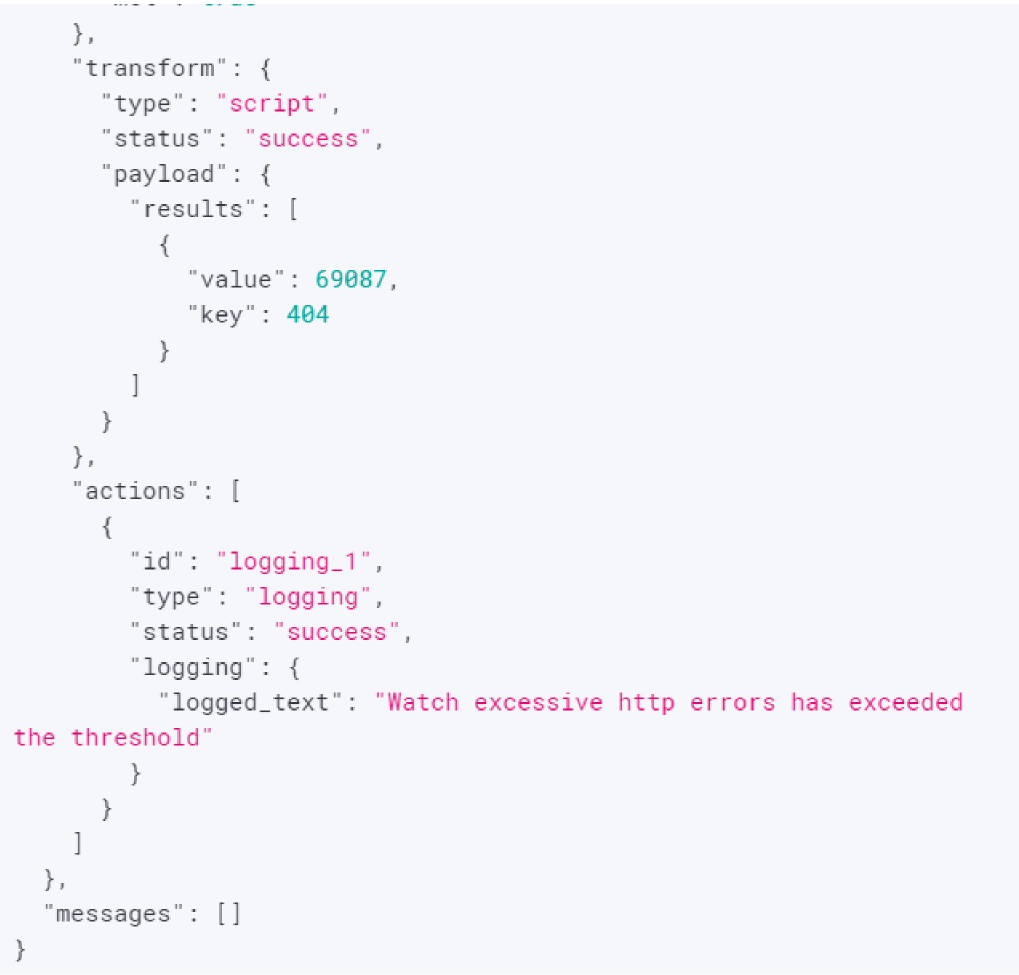
Excessive HTTP Errors is implemented as follows:

```
WHEN count() GROUPED OVER top 5 'http.response.status_code' IS ABOVE 400
FOR THE LAST 5 minutes
```

- Metric: **WHEN count() GROUPED OVER top 5'http.response.status_code'**
- Threshold: **IS ABOVE 400**
- Vulnerability Mitigated: **Enumeration/Brute Force**
- Reliability: The alert is highly reliable. Measuring by error code above 400 will filter normal responses and will trigger alert that goes above high rate.



2021-07-10T21:52:28+00:00	▷ Firing
2021-07-10T21:51:28+00:00	▷ Firing
2021-07-10T21:50:28+00:00	▷ Firing
2021-07-10T21:49:28+00:00	▷ Firing
2021-07-10T21:48:28+00:00	▷ Firing
2021-07-10T21:47:28+00:00	▷ Firing



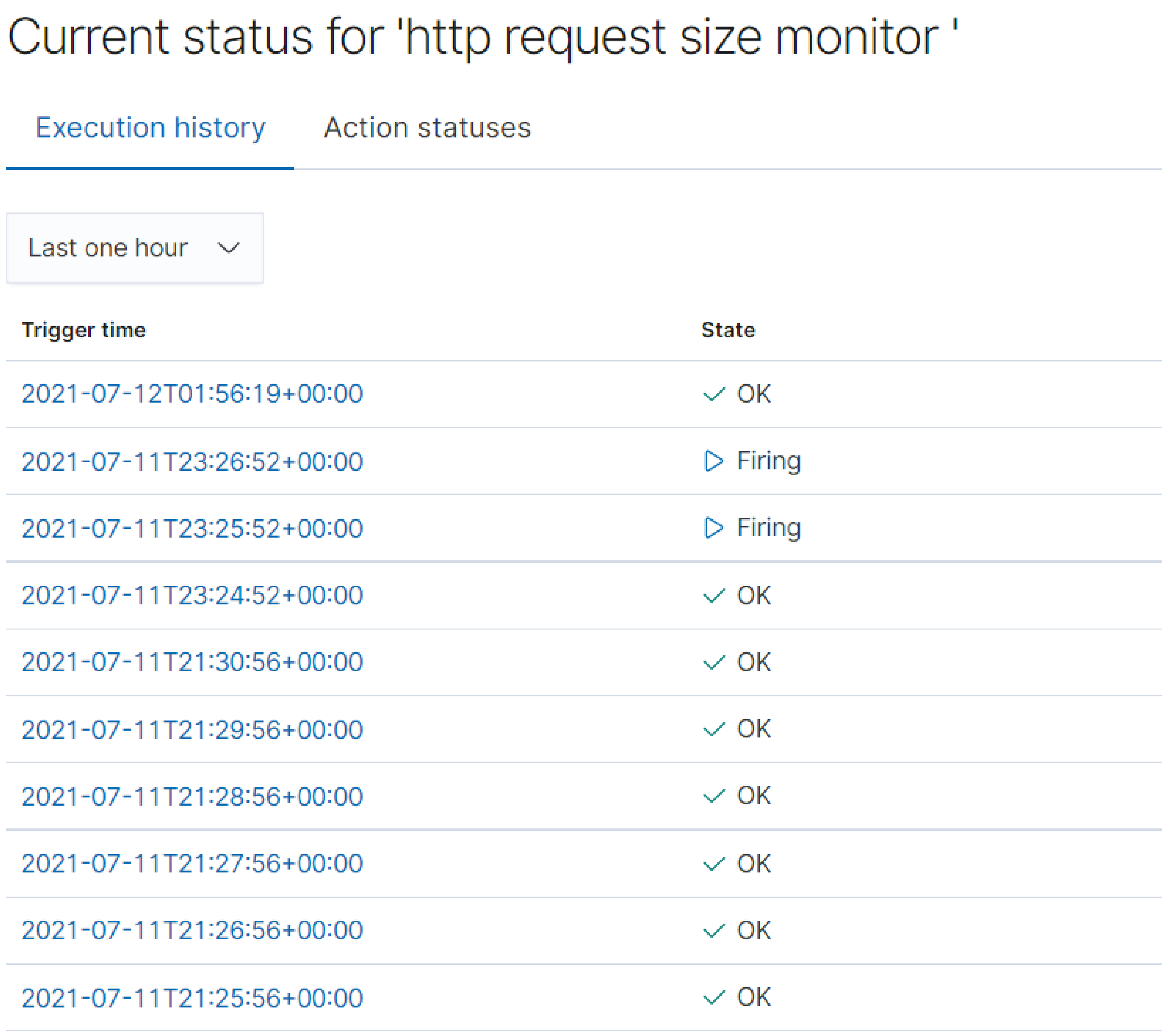
```
    },
    "transform": {
      "type": "script",
      "status": "success",
      "payload": {
        "results": [
          {
            "value": 69087,
            "key": 404
          }
        ]
      }
    },
    "actions": [
      {
        "id": "logging_1",
        "type": "logging",
        "status": "success",
        "logging": {
          "logged_text": "Watch excessive http errors has exceeded
the threshold"
        }
      }
    ]
  },
  "messages": []
}
```

HTTP Request Size Monitor

HTTP Request Size Monitor is implemented as follows:

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

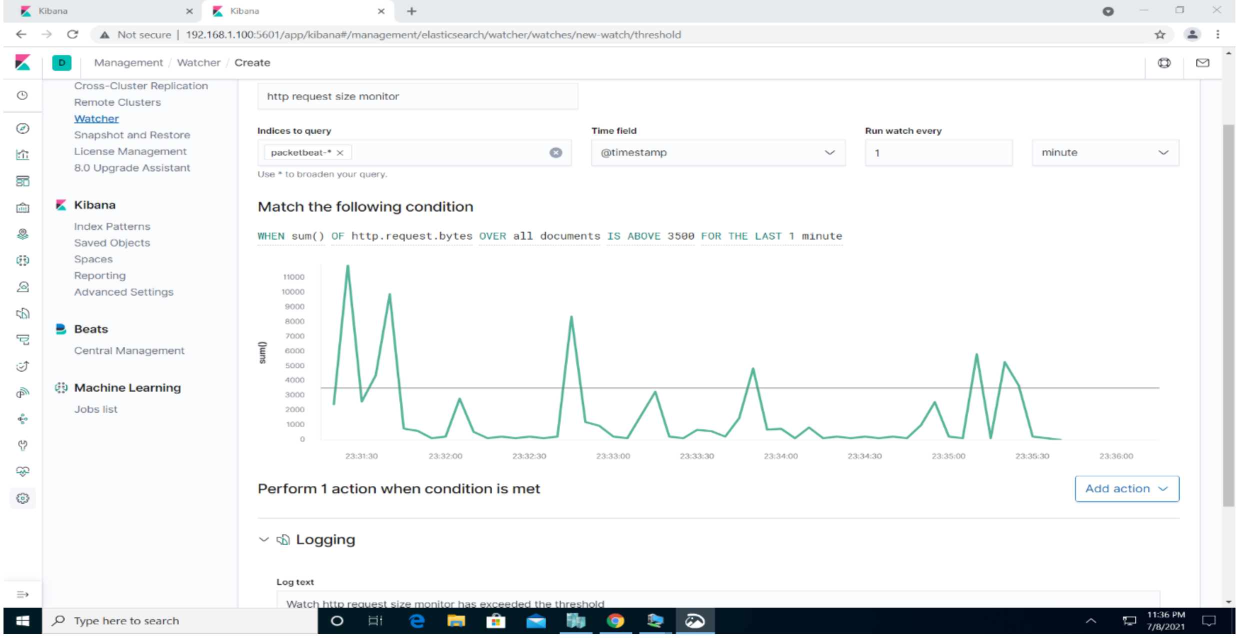
- Metric: **WHEN sum() of http.request.bytes OVER all documents**
- Threshold: **IS ABOVE 3500**
- Vulnerability Mitigated: **Code injection XXS in HTTP requests or DDOS**
- Reliability: The alert is medium reliability. It could create false positives when legitimate HTTP requests occurs in large amounts.



Current status for 'http request size monitor '

Trigger time	State
2021-07-12T01:56:19+00:00	✓ OK
2021-07-11T23:26:52+00:00	▷ Firing
2021-07-11T23:25:52+00:00	▷ Firing
2021-07-11T23:24:52+00:00	✓ OK
2021-07-11T21:30:56+00:00	✓ OK
2021-07-11T21:29:56+00:00	✓ OK
2021-07-11T21:28:56+00:00	✓ OK
2021-07-11T21:27:56+00:00	✓ OK
2021-07-11T21:26:56+00:00	✓ OK
2021-07-11T21:25:56+00:00	✓ OK

```
    },
    "condition": {
      "type": "script",
      "status": "success",
      "met": true
    },
    "transform": {
      "type": "script",
      "status": "success",
      "payload": {
        "result": 24821
      }
    },
    "actions": [
      {
        "id": "logging_1",
        "type": "logging",
        "status": "success",
        "logging": {
          "logged_text": "Watch http request size monitor has exceeded the threshold"
        }
      }
    ]
  },
  "messages": []
}
```



http request size monitor

Indices to query: packetbeat-*

Use * to broaden your query.

Time field: @timestamp

Run watch every: 1 minute

Match the following condition

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

Perform 1 action when condition is met

Logging

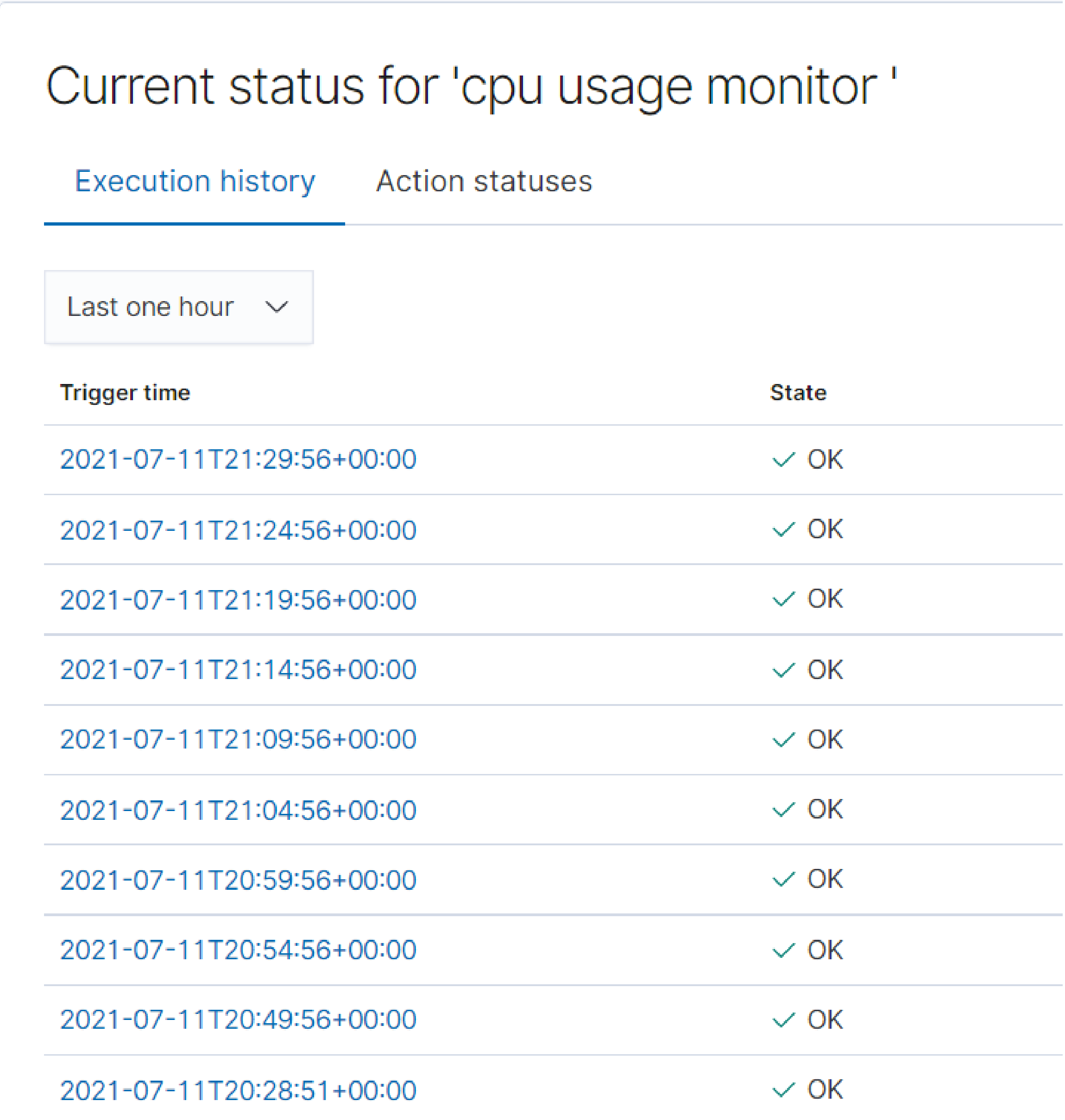
Log text: Watch http request size monitor has exceeded the threshold

CPU Usage Monitor

CPU Usage Monitor is implemented as follows:

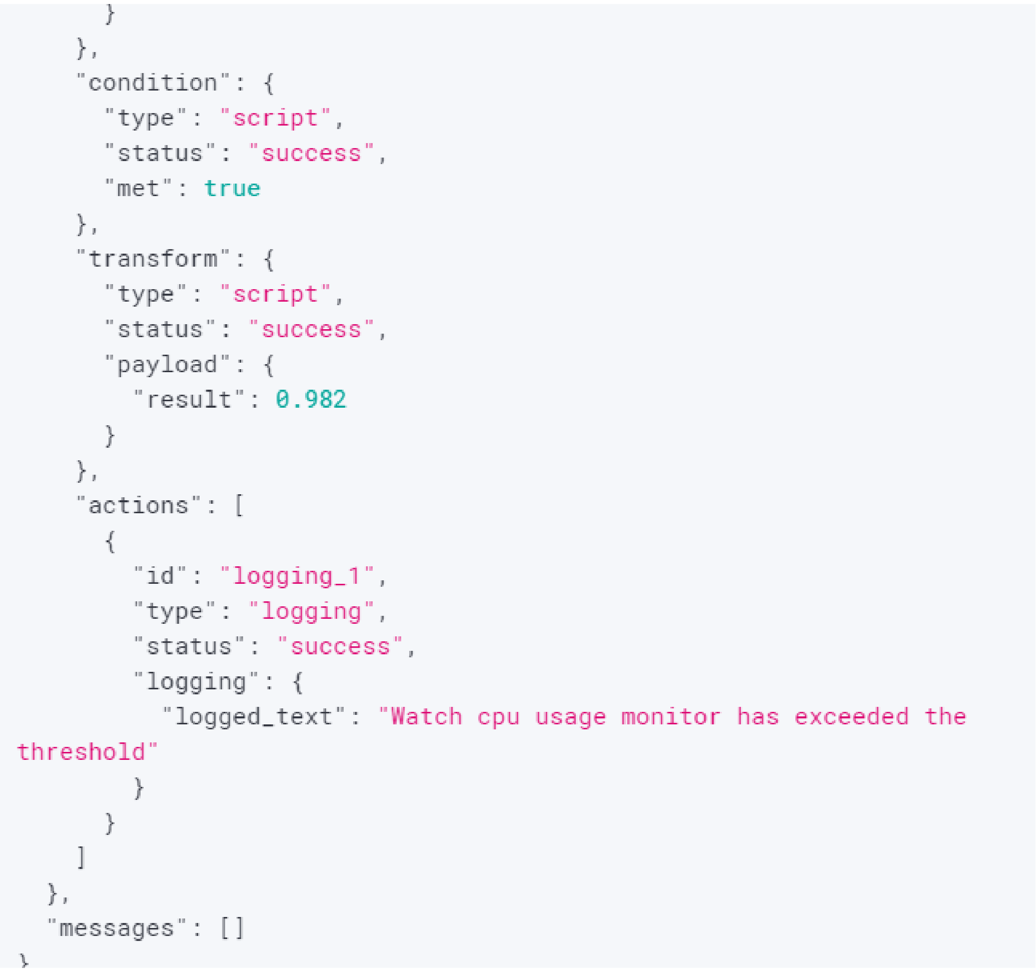
WHEN max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes

- Metric: **WHEN max() OF system.process.cpu.total.pct OVER all documents**
- Threshold: **IS ABOVE 0.5**
- Vulnerability Mitigated: Malicious code execution on target server increase CPU usage.
- Reliability: The alert is highly reliable. It can detect any software or program running on server that taking up resources as well as the malicious software that will trigger the alert.

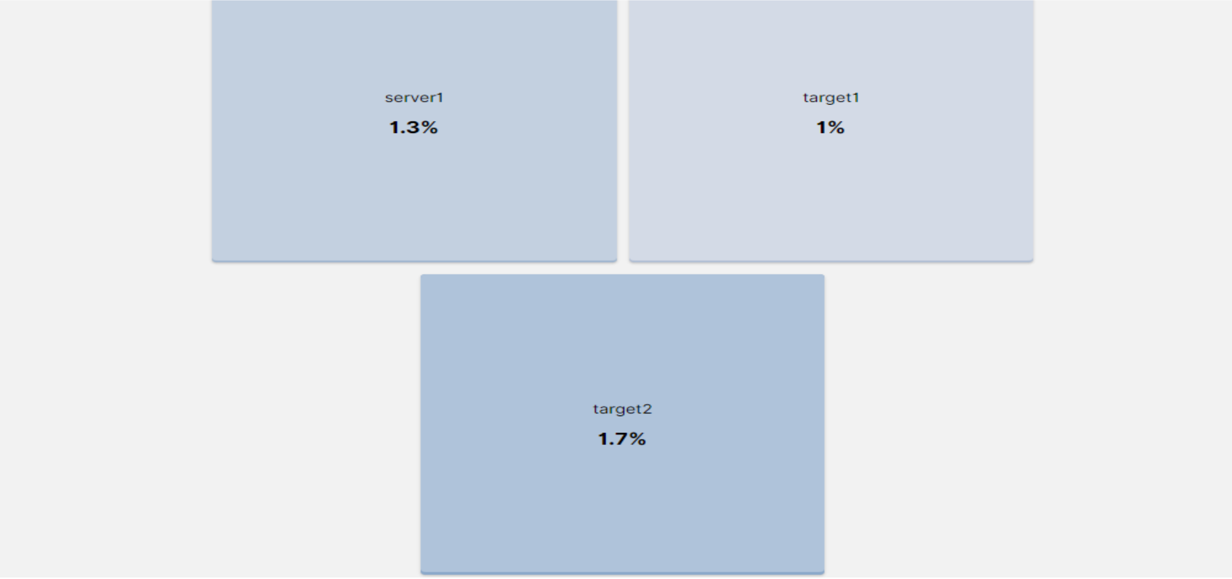


Current status for 'cpu usage monitor '

Trigger time	State
2021-07-11T21:29:56+00:00	✓ OK
2021-07-11T21:24:56+00:00	✓ OK
2021-07-11T21:19:56+00:00	✓ OK
2021-07-11T21:14:56+00:00	✓ OK
2021-07-11T21:09:56+00:00	✓ OK
2021-07-11T21:04:56+00:00	✓ OK
2021-07-11T20:59:56+00:00	✓ OK
2021-07-11T20:54:56+00:00	✓ OK
2021-07-11T20:49:56+00:00	✓ OK
2021-07-11T20:28:51+00:00	✓ OK



```
      }
    },
    "condition": {
      "type": "script",
      "status": "success",
      "met": true
    },
    "transform": {
      "type": "script",
      "status": "success",
      "payload": {
        "result": 0.982
      }
    },
    "actions": [
      {
        "id": "logging_1",
        "type": "logging",
        "status": "success",
        "logging": {
          "logged_text": "Watch cpu usage monitor has exceeded the
threshold"
        }
      }
    ]
  },
  "messages": []
}
```



server1
1.3%

target1
1%

target2
1.7%

Suggestions for Going Further

Each alert above pertains to a specific vulnerability/exploit. Recall that alerts only detect malicious behavior, but do not stop it. For each vulnerability/exploit identified by the alerts above, suggest a patch. E.g., implementing a blocklist is an effective tactic against brute-force attacks. 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, 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:

Excessive HTTP Errors

- Patch: WordPress Hardening
  - Implement regular updates to WordPress
    - WordPress Core
    - PHP version
    - Plugins
  - Install security plugin(s)
    - Ex. Wordfence (adds security functionality)
  - Disable unused WordPress features and settings like:
    - WordPress XML-RPC (on by default)
    - WordPress REST API (on by default)
  - Block requests to /?author= by configuring web server settings
  - Remove WordPress logins from being publicly accessible specifically:
    - /wp-admin
    - /wp-login.php
- Why It Works:
  - Regular updates to WordPress, the PHP version and plugins is an easy way to implement patches or fixes to exploits/vulnerabilities.
  - Depending on the WordPress security plugin it can provide things like:
    - Malware scans
    - Firewall
    - IP options (to monitor/block suspicious traffic)
  - REST API is used by WPScan to enumerate users
    - Disabling it will help mitigate WPScan or enumeration in general
  - XML-RPC uses HTTP as it's method of data transport
  - WordPress links (permalinks) can include authors (users)
    - Blocking request to view the all authors (users) helps mitigate against user enumeration attacks
  - Removal of public access to WordPress login helps reduce the attack surface

HTTP Request Size Monitor

- Patch: Code Injection/DDOS Hardening
    - Implementation of HTTP Request Limit on the web server
        - Limits can include a number of things:
            - Maximum URL Length
            - Maximum length of a query string
            - Maximum size of a request
    - Implementation of input validation on forms
- Why It Works:
    - If an HTTP request URL length, query string and over size limit of the request a 404 range of errors will occur.
        - This will help reject these requests that are too large.
    - Input validation can help protect against malicious data anyone attempts to send to the server via the website or application in/across a HTTP request.

CPU Usage Monitor

- Patch: Virus or Malware hardening
    - Add or update to a good antivirus.
    - Implement and configure Host Based Intrusion Detection System (HIDS)
        - Ex. SNORT (HIDS)
- Why It Works:
    - Antiviruses specialize in removal, detection and overall prevention of malicious threats against computers.
        - Any modern antivirus usually covers more than viruses and are a robust solution to protecting a computer in general.
    - HIDS monitors and analyzes internals of computing systems.
        - They also monitor and analyze network packets.
        -

Implementing hardening and patches

We could also implement Ansible playbook that would easily configure the WordPress configuration files and properly assign permissions to users.

Playbook must include: install prerequisites, install php extensions, install LAMP packages, Apache configuration, download latest WordPress, set ownership, permissions for directories and files, set up wp-config.