**Exploring Kibana**

​

​

**Instructions**

​

1. After adding a sample of web log data to Kibana, we noticed:

​

- In the last 7 days, how many unique visitors were located in India?

236 unique visitors were located in India

​

- In the last 24 hours, of the visitors from China, how many were using Mac OSX?

7 visitors from China were using Mac OSX

​

- In the last 2 days, what percentage of visitors received 404 errors? How about 503 errors?

404: 100%

503: 0%

- In the last 7 days, what country produced the majority of the traffic on the website?

China (CN)

- Of the traffic that's coming from that country, what time of day had the highest amount of activity?

10 am and 12 pm

- List all the types of downloaded files that have been identified for the last 7 days, along with a short

description of each file type (use Google if you aren't sure about a particular file type).

Gz files: A GZ file is an archive file compressed by the standard GNU zip (gzip)

Css files: A CSS file is a cascading style sheet ([CSS](https://techterms.com/definition/css)) file used to format the contents of a webpage

Deb files: deb file is used to signify a collection of files managed by the Debian packages management system

Zip files: ZIP is an archive file format that supports lossless data compression

Rpm files: RPM originally stands for Red Hat Package Manager, is a free and open-source package management system

​

3. Now that you have a feel for the data, Let's dive a bit deeper. Look at the chart that shows Unique Visitors Vs. Average Bytes.

- Locate the time frame in the last 7 days with the most amount of bytes (activity).

3 hours

- In your own words, is there anything that seems potentially strange about this activity?

It is suspicious that the average amount of bytes transferred doubled from the previous activity from 8,682 to 15,709

​

4. Filter the data by this event.

- What is the timestamp for this event? May 23, 2021 @ 22:57:28.552

- What kind of file was downloaded? rpm

- From what country did this activity originate? India

- What HTTP response codes were encountered by this visitor? 200

​

5. Switch to the Kibana Discover page to see more details about this activity.

- What is the source IP address of this activity?

35.143.166.159

- What are the geo coordinates of this activity?

geo.coordinates: { "lat": 43.34121, "lon": -73.6103075 }

- What OS was the source machine running?

Win 8

- What is the full URL that was accessed?

https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-6.3.2-i686.rpm

- From what website did the visitor's traffic originate?

​ http://facebook.com/success/jay-c-buckey

6. Finish your investigation with a short overview of your insights.

​

- What do you think the user was doing?

The user is trying to masquerade himself and make the victim appears to be the imposter.

- Was the file they downloaded malicious? If not, what is the file used for?

No, the file is not malicious. The downloaded file I686.rpm is an rpm file. The name RPM variously refers to the .rpm file format, files in this format, software packaged in such files, and the package manager itself.

- Is there anything that seems suspicious about this activity?

Yes, “request and GET and 200” tells us that something successfully happened

​

​

**Kibana Metrics and Logs Orientation**

​

​Generating a high amount of failed SSH login attempts and verify that Kibana from Jump-Box

- Running failed SSH command in a loop to generate failed login log entries.

​

for i in {1..10}; do ssh [azureuser@10.0.0.5](mailto:azureuser@10.0.0.5); done

Text

Description automatically generated

​

- Generated failed login attempts captured on Kibana

​

Text

Description automatically generated

\*\*Bonus\*\*:

Nested loop that generates SSH login attempts across all three of your VM's.

​for i in {1..10}; do ssh azureuser@10.0.0.5; ssh azureuser@10.0.0.6; ssh azureuser@10.0.0.7; done

​

Failed SSH login attempted captured on Kibana by running the Nested loop across the 3 VMs:

Text

Description automatically generated

​

</details>

​

**#### Linux Stress**

​

Task: Generate a high amount of CPU usage on the pentesting machines and verify that Kibana picks up this data.

​

<details>

​

<summary> Activity File: Linux Stress </summary>

​

​

#### Scenario

​

- You are a cloud architect that has been tasked with setting up an ELK server to gather logs for the Incident Response team to use for training.

​

- Before you hand over the server to the IR team, your senior architect has asked that you verify the ELK server is working as expected and pulling both logs and metrics from the pen-testing web servers.

​

​

\*\*Your Task\*\*: Generate a high amount of CPU usage on the pentesting machines and verify that Kibana picks up this data.

​

---

​

#### Notes

​

The Metrics page for a single VM shows the CPU usage for that machine. This shows how much work the machine is doing. Excessively high CPU usage is typically a cause for concern, as overworked computers are at greater risk for failure.

​

- Metricbeat forwards data about CPU load to Elasticsearch, which can be visualized with Kibana.

​

- In this activity, you will intentionally stress the CPU of one of your VMs, then find evidence of the increased activity in Kibana.

​

Linux has a common, easy-to-use diagnostic program called `stress`. It is easy to use and can be downloaded via `apt`.

​

#### Instructions

​

1. From your jump box, start up your Ansible container and attach to it.

​

2. SSH from your Ansible container to one of your WebVM's.

​

3. Run `sudo apt install stress` to install the stress program.

​

4. Run `sudo stress --cpu 1` and allow `stress` to run for a few minutes.

​

5. View the Metrics page for that VM in Kibana. What indicates that CPU usage increased?

​

6. Run the `stress` program on all three of your VMs and take screenshots of the data generated on the Metrics page of Kibana.

​

- \*\*Note:\*\* The stress program will run until you quit with Ctrl+C.

</details>

​

​

**#### wget-DoS**

​

​

Task: Generate a high amount of web requests to your pen-testing servers and make sure that Kibana is picking them up.

​

<details>

​

<summary> Activity File: wget-DoS </summary>

​

​

#### Scenario

​

- You are a cloud architect that has been tasked with setting up an ELK server to gather logs for the Incident Response team to use for training.

​

- Before you hand over the server to the IR team, your senior architect has asked that you verify the ELK server is working as expected and pulling both logs and metrics from the pen-testing web servers.

​

\*\*Your Task\*\*: Generate a high amount of web requests to your pen-testing servers and make sure that Kibana is picking them up.

​

---

​

#### Instructions

​

The Metrics section for a single VM will show Load and Network Traffic data.

​

We can generate abnormal data to view by creating a DoS web attack. The command-line program `wget` can do this easily.

​

`wget` will download a file from any web server. Use man pages for more info on `wget`.

​

1. Log into your jump box.

​

2. Run `wget ip.of.web.vm`.

​

```bash

sysadmin@Jump-Box-Provisioner:~$ wget 10.0.0.5

--2020-05-08 15:44:00-- http://10.0.0.5/

Connecting to 10.0.0.5:80... connected.

HTTP request sent, awaiting response... 302 Found

Location: login.php [following]

--2020-05-08 15:44:00-- http://10.0.0.5/login.php

Reusing existing connection to 10.0.0.5:80.

HTTP request sent, awaiting response... 200 OK

Length: 1523 (1.5K) [text/html]

Saving to: ‘index.html’

​

index.html 100%[=======================>] 1.49K --.-KB/s in 0s

​

2020-05-08 15:44:00 (179 MB/s) - ‘index.html’ saved [1523/1523]

```

​ # wget 10.0.0.5

3. Run `ls` to view the file you downloaded from your web VM to your jump box.

​

```bash

sysadmin@Jump-Box-Provisioner:~$ ls

index.html

```

​

4. Run the `wget` command in a loop to generate many web requests.

​

- You can use a bash `for` or `while` loop, directly on the command line, just as you did with the SSH command.

​for i in {1..20}; do wget 10.0.0.5; done

5. Open the Metrics page for the web machine you attacked and answer the following questions:

- Which of the VM metrics were affected the most from this traffic?

​Network Traffic

Memory usage

CPU usage

\*\*Bonus\*\*: Notice that your `wget` loop creates a lot of duplicate files on your jump box.

​

- Write a command to delete \_all\_ of these files at once.

​ find . -name "\*.html" -type f or rm index.\*

- Find a way to run the `wget` command without generating these extra files.

for i in {1..20}; do wget -p 10.0.0.5; done

- Look up the flag options for `wget` and find the flag that lets you choose a location to save the file it downloads.

wget –P [destination\_directory] [URL] wget -P wgetdir 10.0.0.5

- Save that file to the Linux directory known as the "void" or the directory that doesn't save anything.

​ wget -P ~/dev/null 10.0.0.5

\*\*Bonus\*\*: Write a nested loop that sends your `wget` command to all three of your web VMs over and over.

wget 10.0.0.5 10.0.0.6 10.0.0.7

​

</details>