**SSH Barrage**

Task: Generate a high amount of failed SSH login attempts and verify that Kibana is picking up

this activity.

* Log into your Jump Box.
* Create a bash script to generate failed login attempts in a loop to all three Web VM servers.

RedAdmin@Jump-Box-Provisioner:~$ cat ssh\_barrage\_all.sh

#!/bin/bash

# Script to generate failed login attempts in a loop to all Web VM servers.

while true

do

for i in {7..9}

do

ssh ttadmin@10.0.0.$i

done

done

* Execute script.

RedAdmin@Jump-Box-Provisioner:~$ ./ssh\_barrage\_all.sh

* + Hit CTRL + c to exit from loop!
* From the left pane in Kibana, select ‘Logs’.
  + Search for ‘system.auth’ or the time that this event occurred.

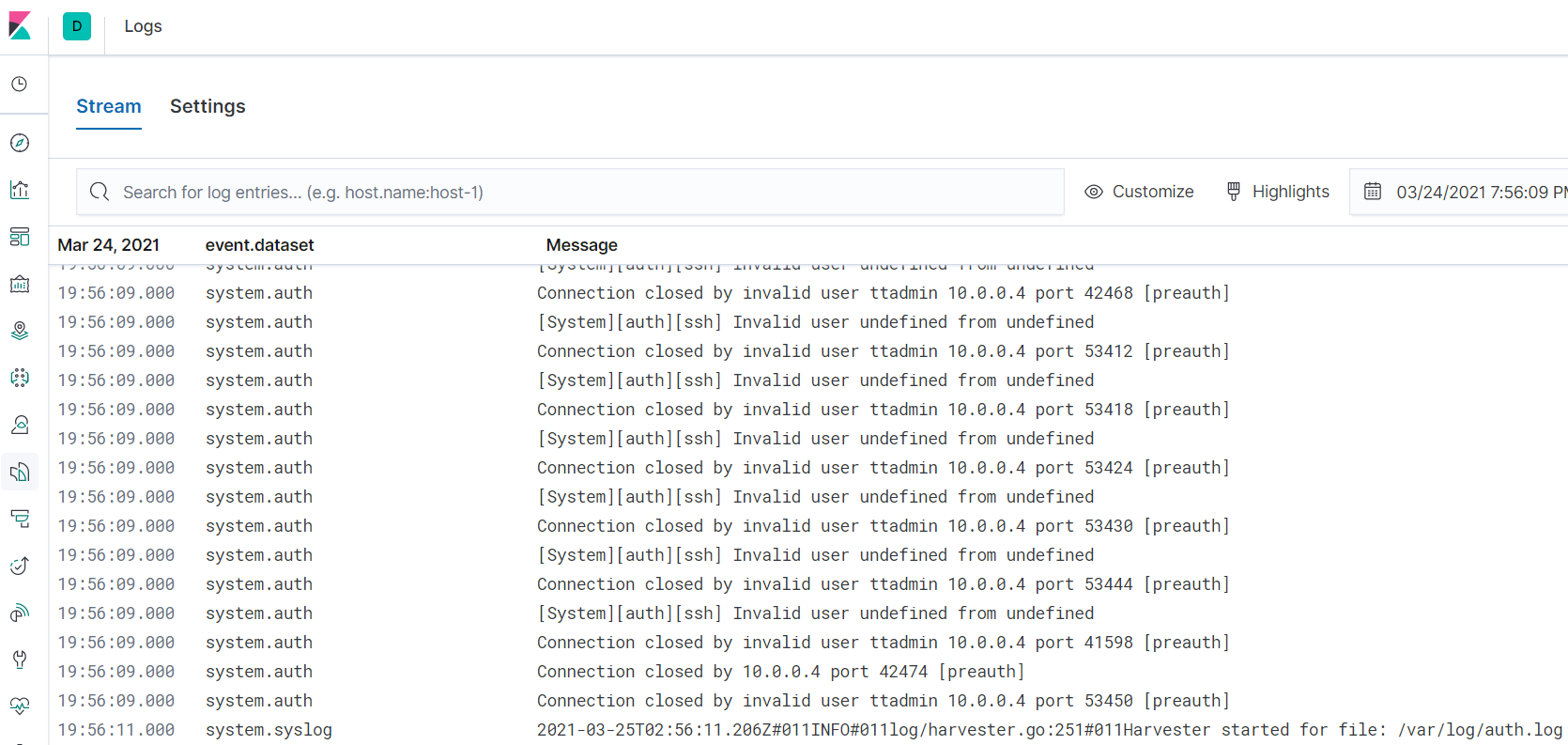
**SSH Barrage (con’t)**

Task: Generate a high amount of failed SSH login attempts and verify that Kibana is picking up

this activity.

(con’t)

Logs should look like the following:



**SSH Barrage (con’t)**

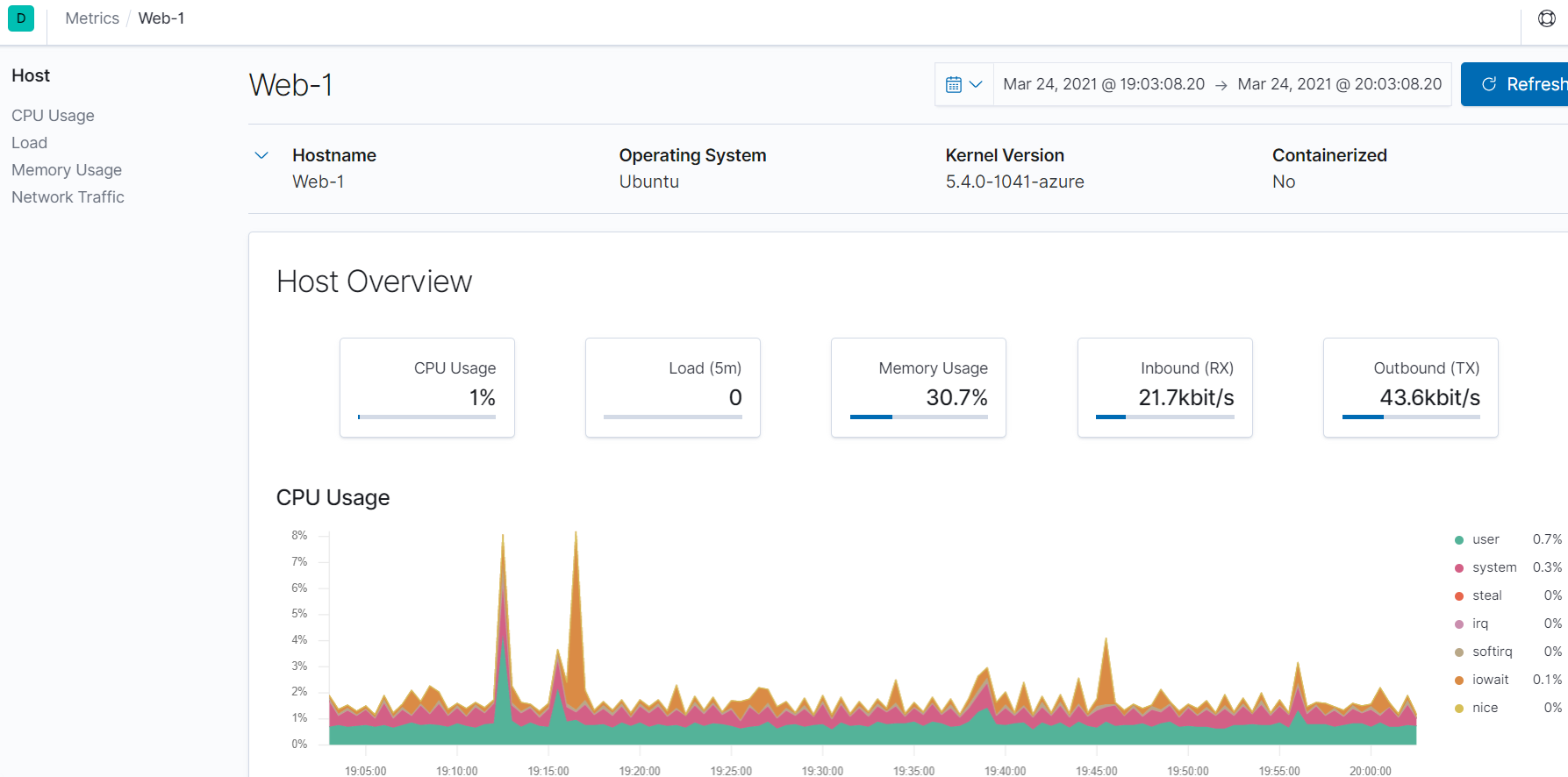
Task: Generate a high amount of failed SSH login attempts and verify that Kibana is picking up

this activity.

(con’t)

* From the left pane in Kibana, select ‘Metrics’.

Metrics should look like the following:



**Linux Stress**

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

* Log into your Jump Box.
* Log into one of your VM containers.
* Install the ‘stress’ program

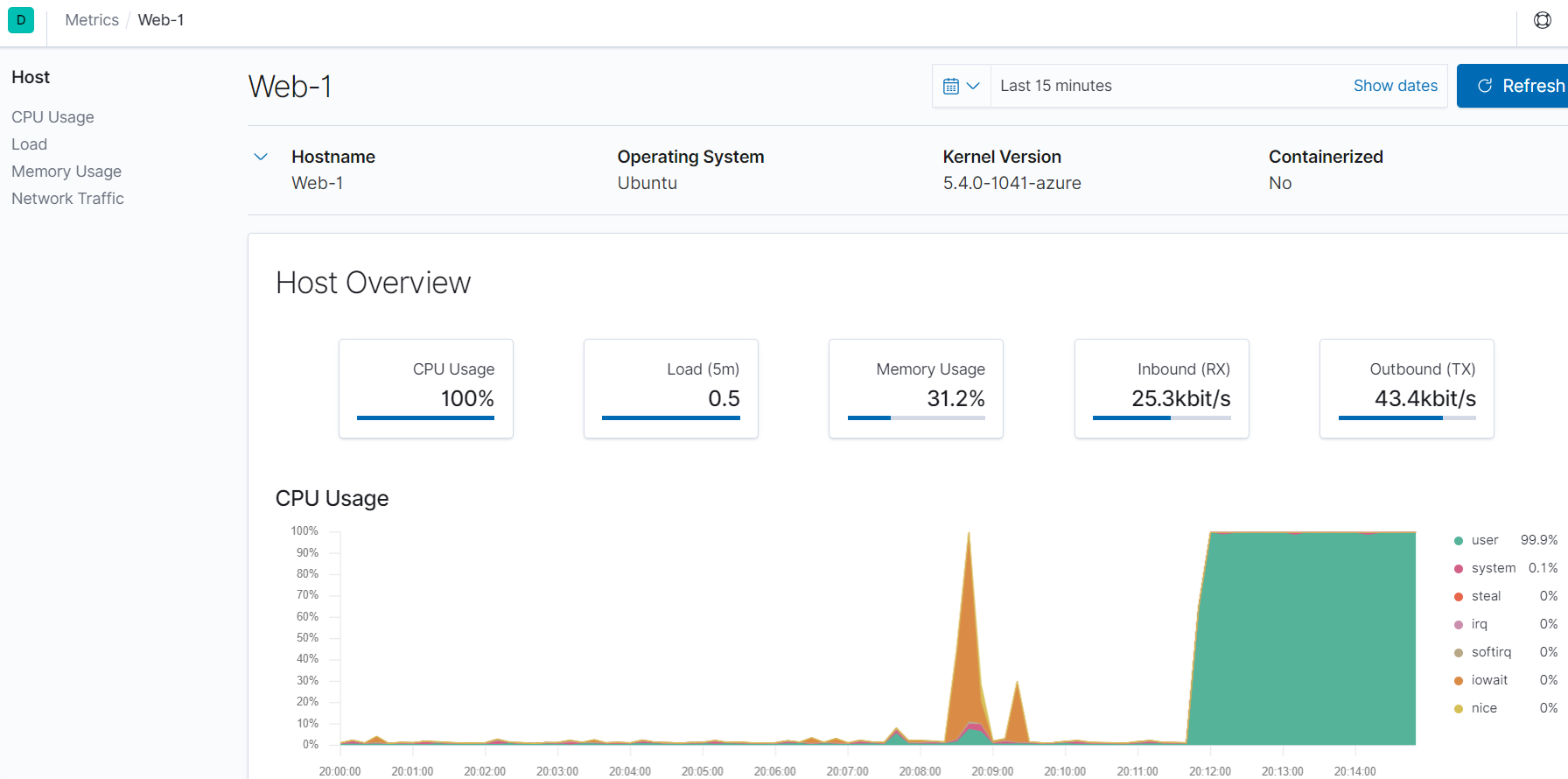
sysadmin@Web-1:~$ sudo apt install stress

* Execute the following command.

sysadmin@Web-1:~$ sudo stress --cpu 1

* + Let this execute for a few minutes and then hit CTRL + c to exit.

Metrics should look like the following:

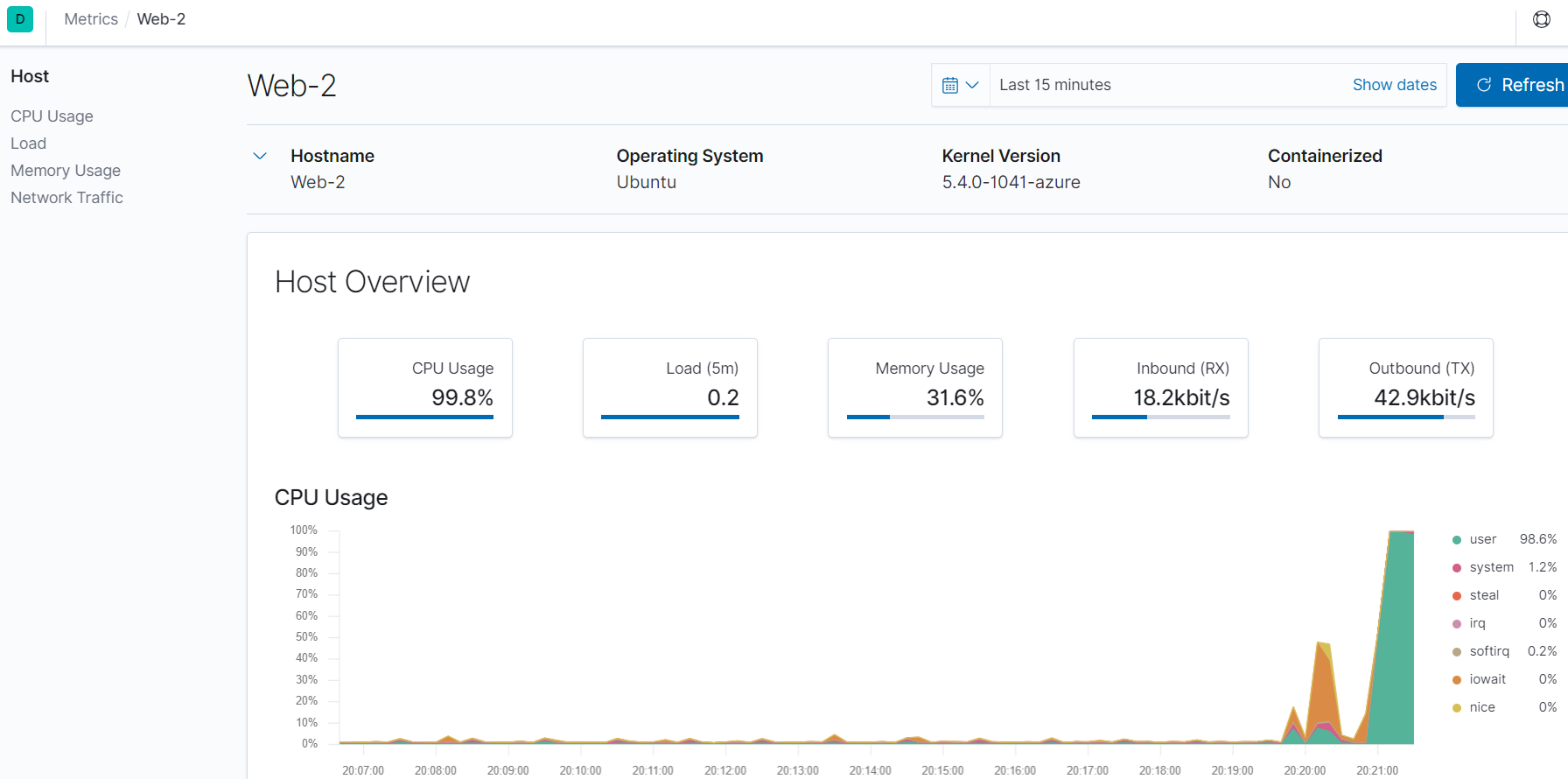


**Linux Stress (con’t)**

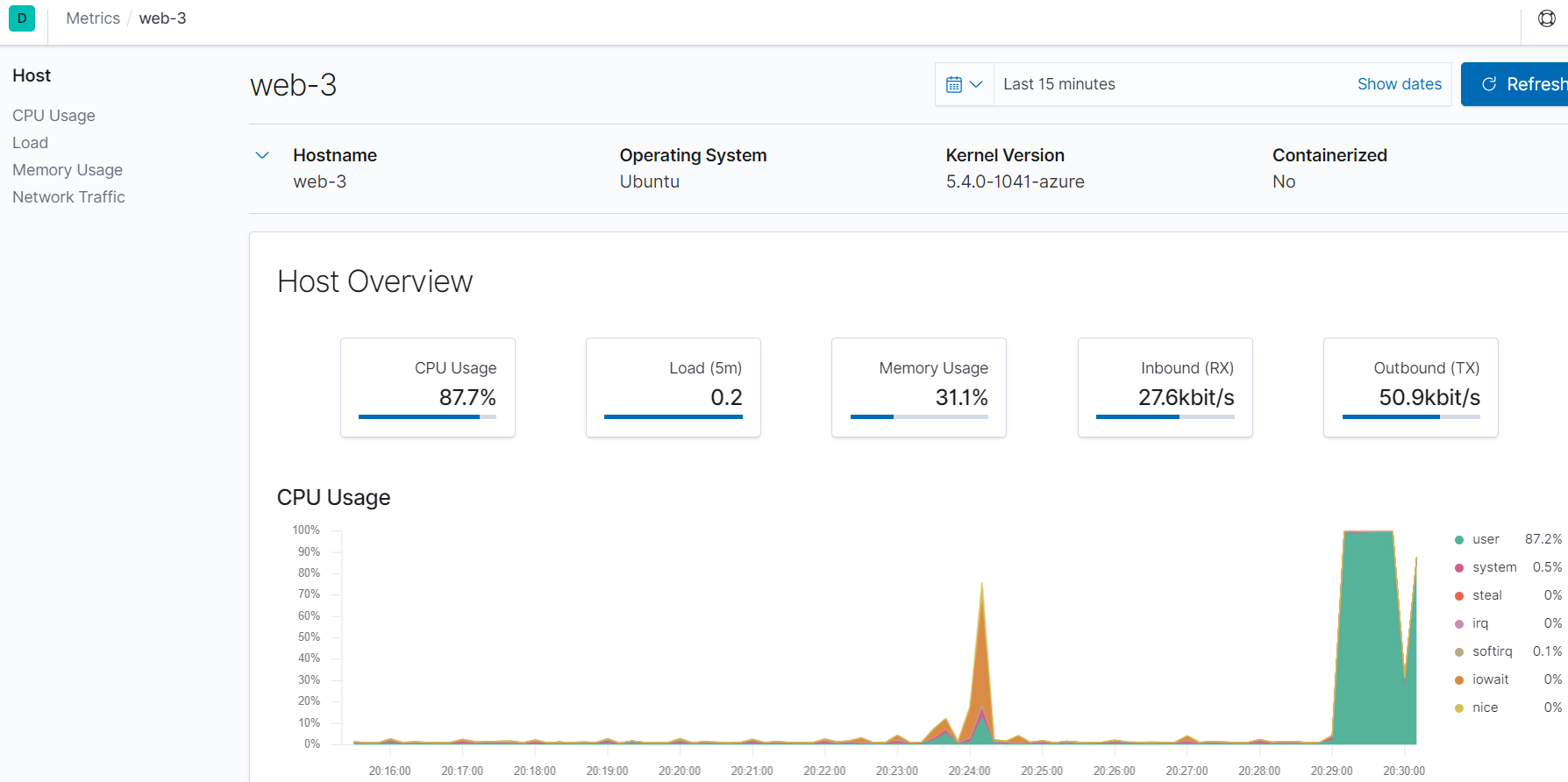
Task: Generate a high amount of CPU usage on the pentesting machines and verify that Kibana picks up this data.

(con’t)

Metrics stress test for Web-2 server.



Metrics stress test for Web-3 server.



**wget-DoS**

Task: Generate a high amount of CPU usage on the pentesting machines and verify that Kibana is picking them up.

* Execute the following command:

RedAdmin@Jump-Box-Provisioner:~$ wget 10.0.0.9

--2021-03-25 03:32:11-- http://10.0.0.9/

Connecting to 10.0.0.9:80... connected.

HTTP request sent, awaiting response... 302 Found

Location: login.php [following]

--2021-03-25 03:32:11-- http://10.0.0.9/login.php

Reusing existing connection to 10.0.0.9:80.

HTTP request sent, awaiting response... 200 OK

Length: 1415 (1.4K) [text/html]

Saving to: ‘index.html’

index.html 100%[=================================================>] 1.38K --.-KB/s in 0s

2021-03-25 03:32:11 (233 MB/s) - ‘index.html’ saved [1415/1415]

* + An HTML file has been downloaded
* List all files in the current directory.

File ‘index.html’ has been created.

RedAdmin@Jump-Box-Provisioner:~$ ls -l

total 12

-rw-rw-r-- 1 RedAdmin RedAdmin 1415 Mar 25 03:32 index.html

RedAdmin@Jump-Box-Provisioner:~$ ls -l

* Create a bash script to generate many web requests.

RedAdmin@Jump-Box-Provisioner:~$ cat web\_requests.sh

#!/bin/bash

# Script to generate many web requests.

for i in {1..1000}:

do

wget 10.0.0.9

done

**wget-DoS (con’t)**

Task: Generate a high amount of CPU usage on the pentesting machines and verify that Kibana is picking them up.

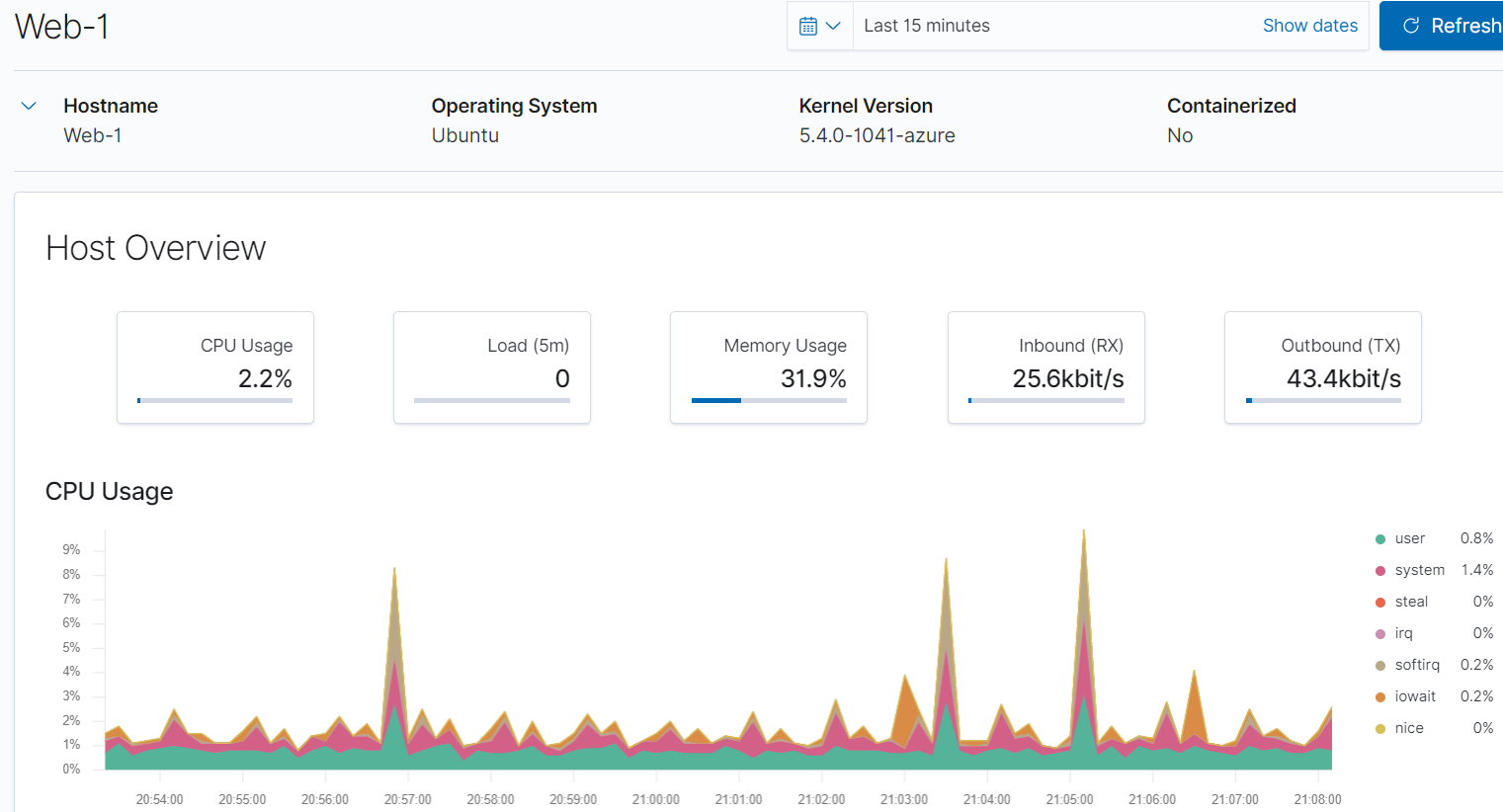
(con’t)

* Execute script.

RedAdmin@Jump-Box-Provisioner:~$ ./web\_requests.sh

* + Hit CTRL + c to exit from loop!
* From the left pane in Kibana, select ‘Metrics’.

The following metrics for Web-1 server.

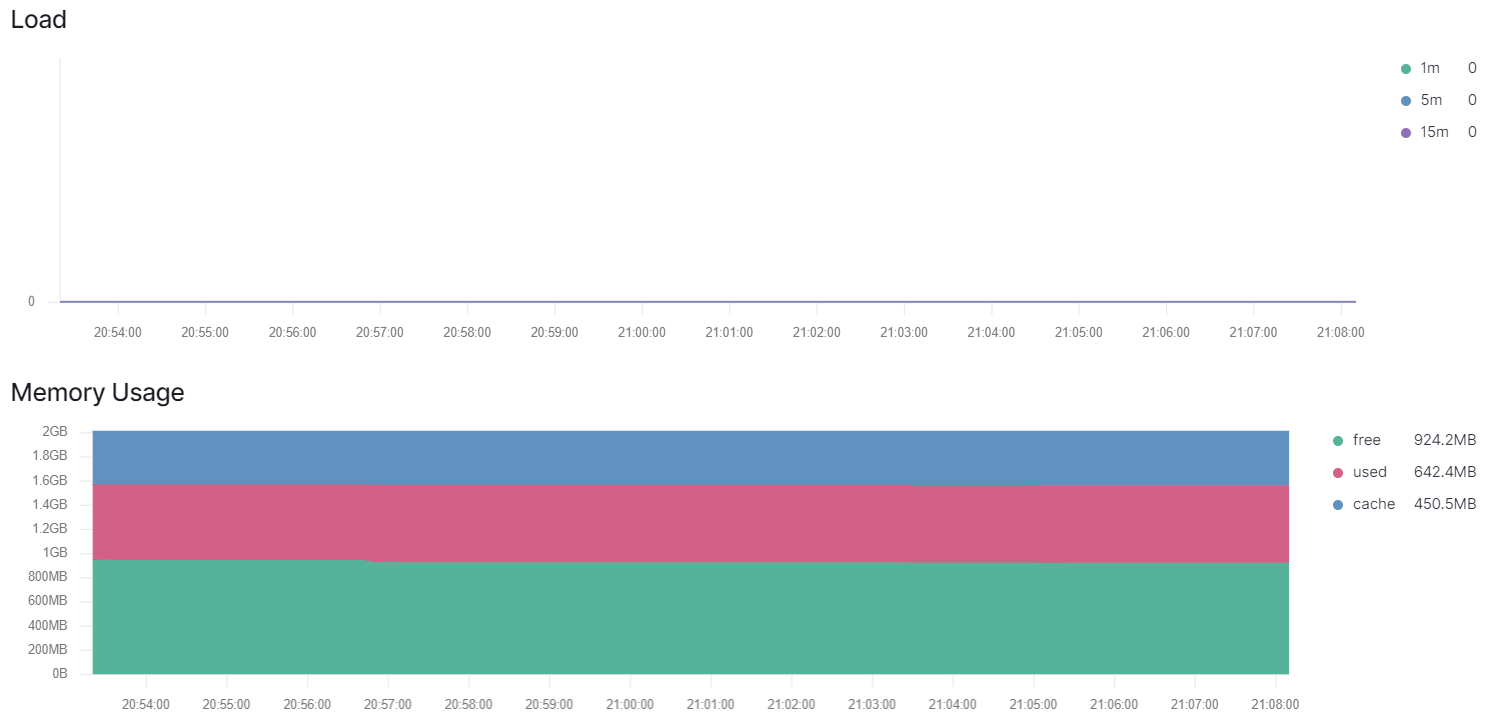


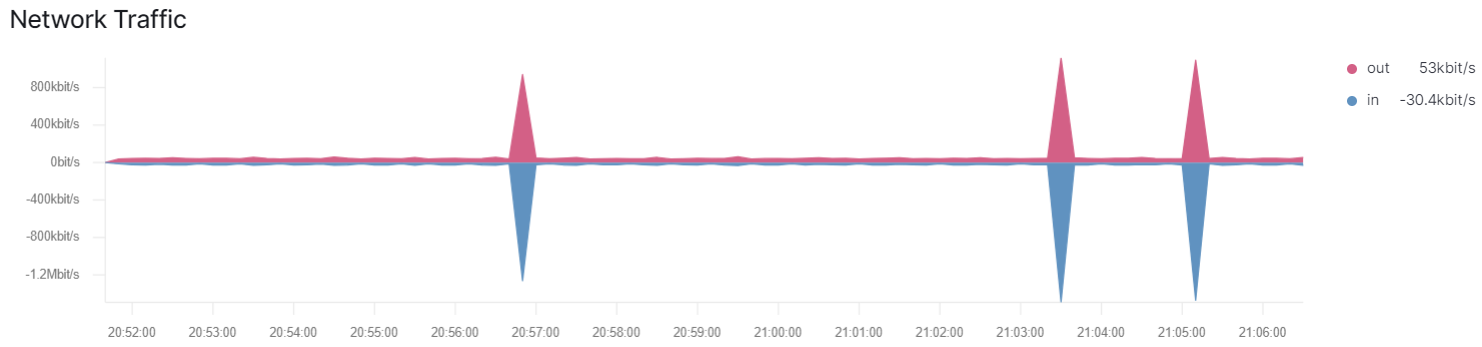
**wget-DoS (con’t)**

Task: Generate a high amount of CPU usage on the pentesting machines and verify that Kibana is picking them up.

(con’t)

The following metrics for Web-1 server.





**wget-DoS (con’t)**

Task: Generate a high amount of CPU usage on the pentesting machines and verify that Kibana is picking them up.

(con’t)

**NOTE:**

The bash script was edited to remove the duplicate ‘index.html’ files.

#!/bin/bash

# Script to generate many web requests.

for i in {1..1000}:

do

wget 10.0.0.9 -O /dev/null

done