| Lab 241

Administrative Services

Student: Mane Zakarian
Bootcamp: Forge AWS re/Start UYMON5
Date: 2023

Objectives

In this lab, you will:

- Check the status of the service **httpd** to ensure that it is running, and that you can make an http connection to the local host IP address
- You will also learn how to monitor your Amazon Linux 2 EC2 instance
  - Using the Linux **top** command
  - Using **AWS CloudWatch**

Accessing the AWS Management Console

1. At the top of these instructions, choose Start Lab to launch your lab. A **Start Lab** panel opens, and it displays the lab status.

   Tip: If you need more time to complete the lab, choose the Start Lab button again to restart the timer for the environment.

2. Wait until you see the message *Lab status: ready*, then close the **Start Lab** panel by choosing the **X**.
3. At the top of these instructions, choose AWS. This opens the AWS Management Console in a new browser tab. The system will automatically log you in.

   Tip: If a new browser tab does not open, a banner or icon is usually at the top of your browser with a message that your browser is preventing the site from opening pop-up windows. Choose the banner or icon and then choose **Allow pop ups**.

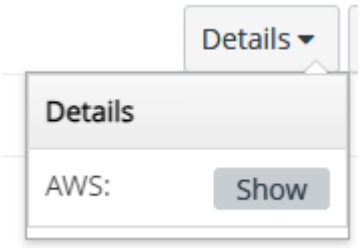
4. Arrange the AWS Management Console tab so that it displays alongside these instructions. Ideally, you will be able to see both browser tabs at the same time so that you can follow the lab steps more easily.

Task 1: Use SSH to connect to an Amazon Linux EC2 instance

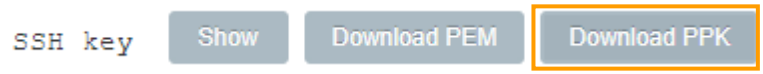
In this task, you will connect to a Amazon Linux EC2 instance. You will use an SSH utility to perform all of these operations.

Windows Users: Using SSH to Connect

1. Select the Details drop-down menu above these instructions you are currently reading, and then select Show. A Credentials window will be presented.



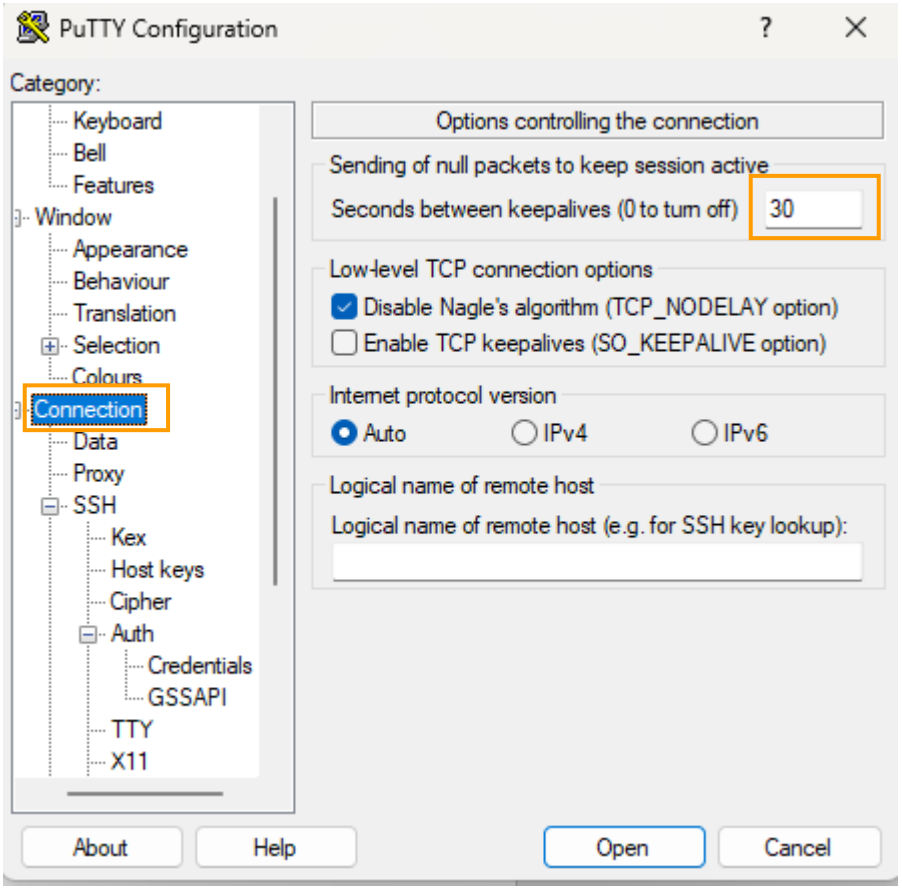
2. Select the **Download PPK** button and save the **labsuser.ppk** file.



3. Make a note of the **PublicIP** address.

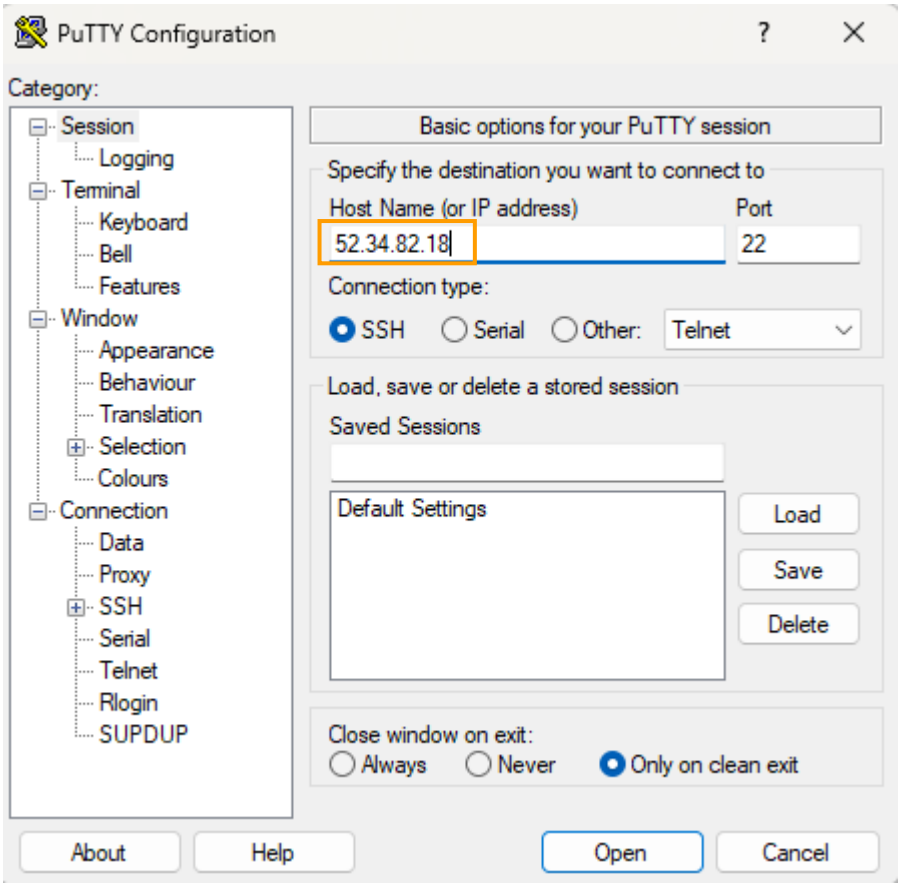
4. Then exit the Details panel by selecting the **X**.
5. Download **PuTTY** to SSH into the Amazon EC2 instance. If you do not have PuTTY installed on your computer.
6. Open **putty.exe**
7. Configure PuTTY timeout to keep the PuTTY session open for a longer period of time.:

- Select **Connection**
- Set **Seconds between keepalives** to `30`

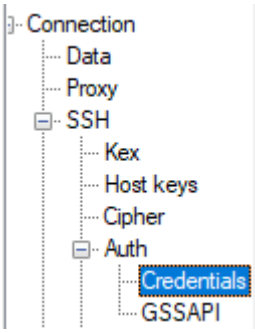


8. Configure your PuTTY session:
   - Select **Session**

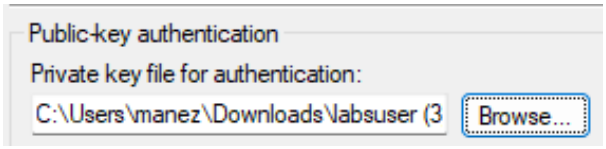
- **Host Name (or IP address):** Paste the **Public DNS or IPv4 address** of the instance you made a note of earlier. Alternatively, return to the EC2 Console and select **Instances**. Check the box next to the instance you want to connect to and in the *Description* tab copy the **IPv4 Public IP** value



- Back in PuTTY, in the **Connection** list, expand **SSH** and select **Auth** *(don't expand it)*

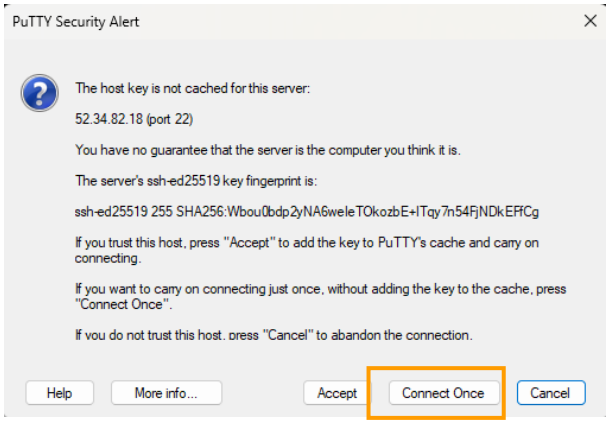


- Select **Browse** and select the lab#.ppk file that you downloaded

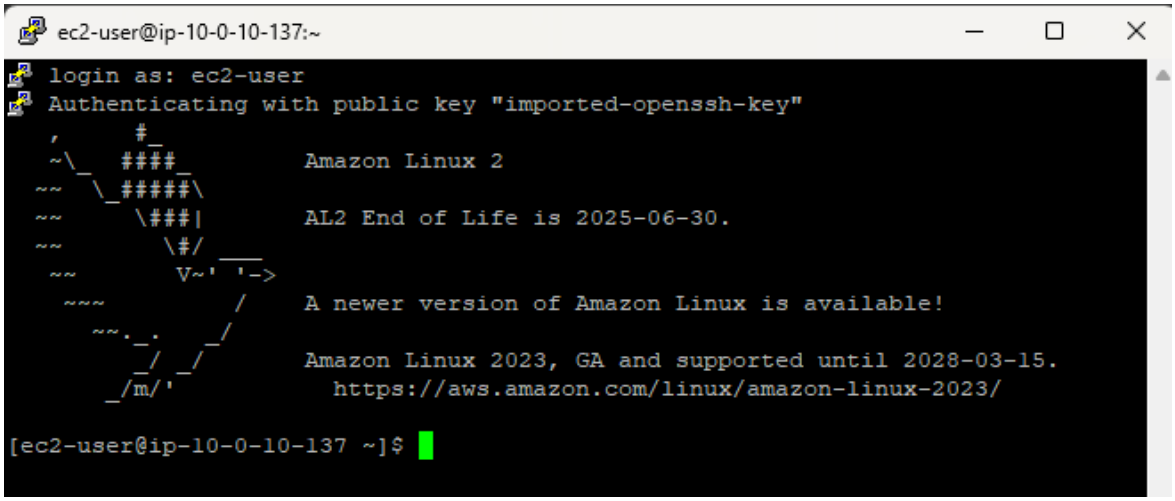


- Select **Open** to select it and then select **Open** again.

9. Select **Yes**, to trust and connect to the host.



10. When prompted **login as**, enter: `ec2-user` This will connect you to the EC2 instance.



Task 2: Check the Status of the httpd Service

Httpd is the service for the Apache http server that is installed on your host. This is a lightweight web server like the ones that run your favorite websites (think let's .... say amazon.com) . In this exercise you check the status of the httpd service, and start it using the **systemctl** command and verify the service is working.

Helpful Hint

You may have to use **sudo** to complete this exercise if you are not root.

24. Check the status of the httpd service by using the **systemctl** commands as shown below and pressing ENTER

```
sudo systemctl status httpd.service
```

Expected Output:

```
[ec2-user@ip-10-0-10-102 ~]$ sudo systemctl status httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor prese
t: disabled)
   Active: inactive (dead)
     Docs: man:httpd.service(8)
```

Figure: The command sudo systemctl status httpd.service and the output says that the httpd service section is inactive (dead).

This indicates that the **httpd** service is **loaded**, which means it is installed and ready to work but is **inactive**. So the next step is to start it

25. Check the status of the httpd service by using the **systemctl** commands as shown below and pressing ENTER.

```
sudo systemctl start httpd.service
```

```
[ec2-user@ip-10-0-10-102 ~]$ sudo systemctl start httpd.service
[ec2-user@ip-10-0-10-102 ~]$
```

26. Check again the status of the httpd service by using the **systemctl** commands as shown below and pressing ENTER

```
sudo systemctl status httpd.service
```

Expected Output

```
[ec2-user@ip-10-0-10-102 ~]$ sudo systemctl status httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor prese
t: disabled)
   Active: active (running) since Sun 2023-10-29 19:49:17 UTC; 33s ago
     Docs: man:httpd.service(8)
 Main PID: 2596 (httpd)
   Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes se
rved/sec:    0 B/sec"
   CGroup: /system.slice/httpd.service
           ├─2596 /usr/sbin/httpd -DFOREGROUND
           ├─2597 /usr/sbin/httpd -DFOREGROUND
           ├─2599 /usr/sbin/httpd -DFOREGROUND
           ├─2604 /usr/sbin/httpd -DFOREGROUND
           ├─2606 /usr/sbin/httpd -DFOREGROUND
           └─2611 /usr/sbin/httpd -DFOREGROUND

Oct 29 19:49:17 ip-10-0-10-102.us-west-2.compute.internal systemd[1]: Startin...
Oct 29 19:49:17 ip-10-0-10-102.us-west-2.compute.internal systemd[1]: Started...
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-10-0-10-102 ~]$
```

Figure: The command sudo systemctl status httpd.service and the output says that the httpd service section is active (running).

27. Now that the **httpd** is running, let's check it works correctly. Open a new tab on your browser and enter : **http://<publicip>** . Replace <publicip> with the public ip that you retrieved at the beginning of the course.

    http://34.218.81.35

    Expected Output

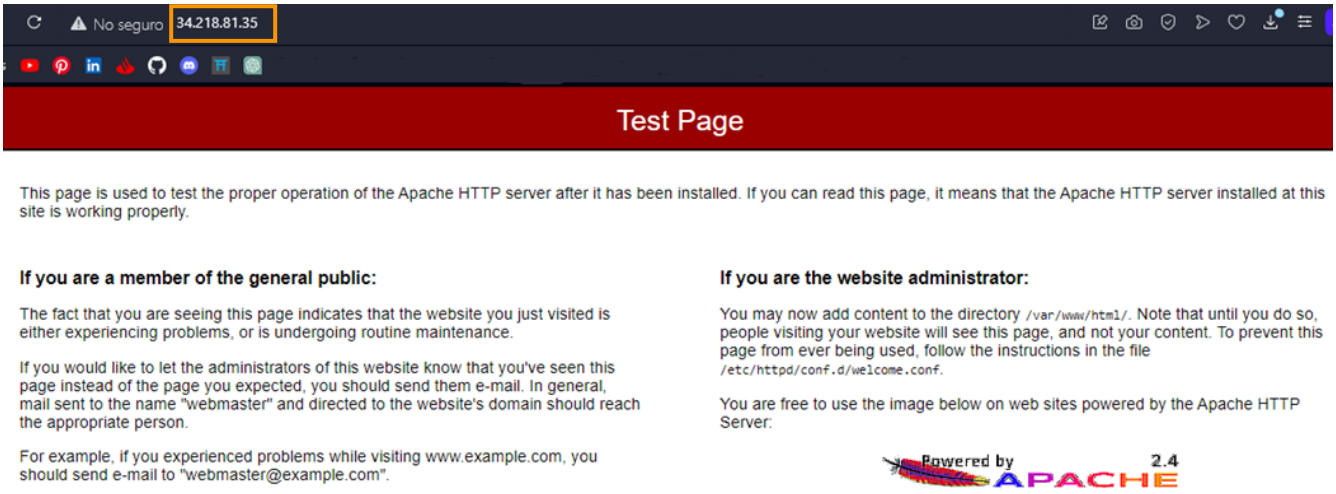
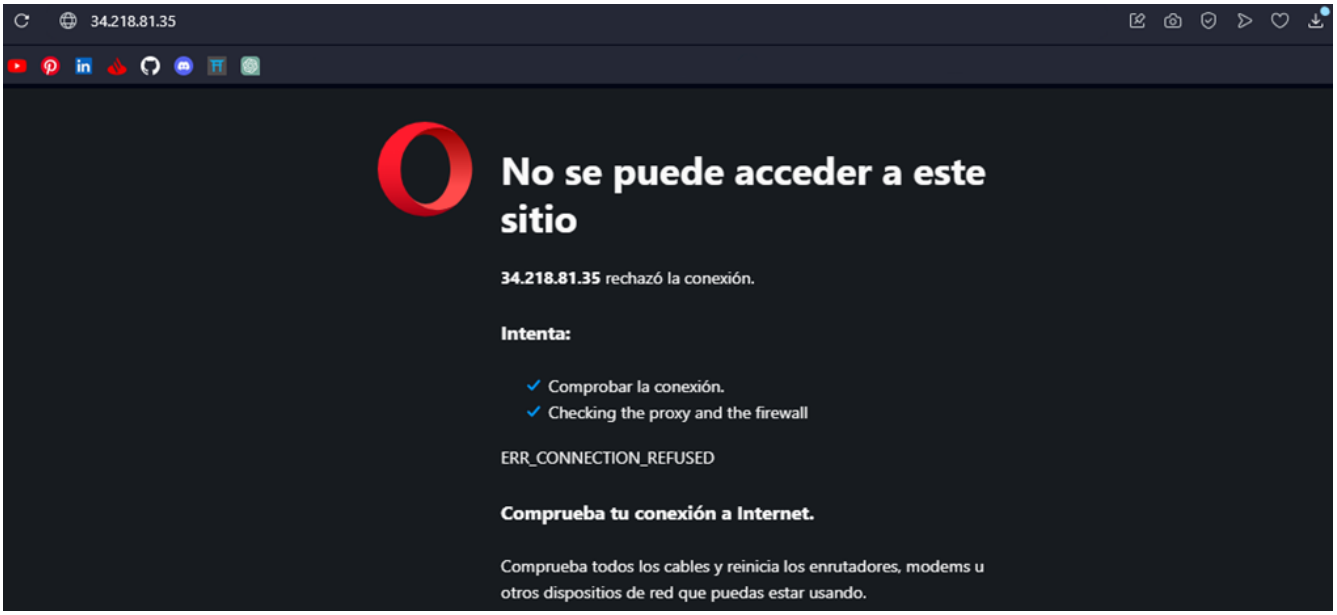


Figure: When httpd is successfully running, an Apache Test page will apear with general information about the proper operation of the Apache HTTP server.

28. You can now stop the service by entering the command below and pressing ENTER

```
sudo systemctl stop httpd.service
```

```
[ec2-user@ip-10-0-10-102 ~]$ sudo systemctl stop httpd.service
```



Task 3: Monitoring a Linux EC2 instance

In this exercise you will use Linux commands to monitor the Amazon Linux2 EC2 instance. You will also open the AWS Console and log into CloudWatch to see how this service can provide you with data to monitor your instance.

Helpful Hint

You may have to use **sudo** to complete this exercise if you are not root.

29. Display the list of running processes by entering the command below and pressing ENTER

```
top
```

Expected Output

```
top - 20:00:52 up 16 min,  1 user,  load average: 0.00, 0.00, 0.00
Tasks:  89 total,   1 running,  47 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 s
KiB Mem :   966816 total,   444924 free,    76504 used,   445388 buff/cache
KiB Swap:        0 total,        0 free,        0 used.   747980 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
    1 root      20   0  123620   5476   3856 S   0.0  0.6   0:00.99 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kthreadd
    4 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/0+
    5 root      20   0       0      0      0 I   0.0  0.0   0:00.08 kworker/u+
    6 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 mm_percpu+
    7 root      20   0       0      0      0 S   0.0  0.0   0:00.02 ksoftirqd+
    8 root      20   0       0      0      0 I   0.0  0.0   0:00.08 rcu_sched
    9 root      20   0       0      0      0 I   0.0  0.0   0:00.00 rcu_bh
   10 root      rt   0       0      0      0 S   0.0  0.0   0:00.00 migration+
   11 root      rt   0       0      0      0 S   0.0  0.0   0:00.00 watchdog/0
   12 root      20   0       0      0      0 S   0.0  0.0   0:00.00 cpuhp/0
   13 root      20   0       0      0      0 S   0.0  0.0   0:00.00 cpuhp/1
   14 root      rt   0       0      0      0 S   0.0  0.0   0:00.00 watchdog/1
   15 root      rt   0       0      0      0 S   0.0  0.0   0:00.20 migration+
   16 root      20   0       0      0      0 S   0.0  0.0   0:00.03 ksoftirqd+
   17 root      20   0       0      0      0 I   0.0  0.0   0:00.00 kworker/1+
   18 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/1+
```

Figure: When running the command top, the output will show the current running processes and resource usages.

Top displays the processes currently running as well as the resource usage like CPU usage and memory usage. Hit **q** to exit and return to the shell. In the next step you are going to run a script that simulates a workload on the CPU.

```
[ec2-user@ip-10-0-10-102 ~]$
```

30. Run the **stress.sh** script that simulates a heavy workload on the EC2 instance by entering the following command and pressing ENTER

```
./stress.sh & top
```

```
top - 20:10:07 up 25 min,  1 user,  load average: 8.86, 2.55, 0.88
Tasks: 105 total,  15 running,  50 sleeping,   0 stopped,   0 zombie
%Cpu(s): 61.4 us, 38.6 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   966816 total,   288268 free,   232896 used,   445652 buff/cache
KiB Swap:        0 total,        0 free,        0 used.   591508 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 2718 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:08.54 stress
 2719 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:08.52 stress
 2721 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:08.54 stress
 2723 ec2-user  20   0  138656  53796    212 R  14.3  5.6   0:08.54 stress
 2725 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:08.54 stress
 2726 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:08.54 stress
 2728 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:08.54 stress
 2729 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:08.55 stress
 2730 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:08.57 stress
 2731 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:08.55 stress
 2720 ec2-user  20   0  138656  61816    212 R  14.0  6.4   0:08.52 stress
 2722 ec2-user  20   0    7580     96      0 R  14.0  0.0   0:08.52 stress
 2724 ec2-user  20   0    7580     96      0 R  14.0  0.0   0:08.54 stress
 2727 ec2-user  20   0    7580     96      0 R  13.6  0.0   0:08.55 stress
    1 root      20   0  123620   5476   3856 S   0.0  0.6   0:00.99 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kthreadd
    4 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/0+
```

31. As in step 1, display the list of running processes by entering the **top** command and pressing ENTER

Expected Output

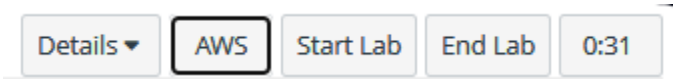
```
top - 20:11:03 up 26 min,  1 user,  load average: 11.95, 4.48, 1.64
Tasks: 105 total,  15 running,  50 sleeping,   0 stopped,   0 zombie
%Cpu(s): 61.1 us, 38.9 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   966816 total,   339896 free,   181264 used,   445656 buff/cache
KiB Swap:        0 total,        0 free,        0 used.   643140 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 2718 ec2-user  20   0    7580     96      0 R  14.6  0.0   0:16.49 stress
 2721 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:16.49 stress
 2722 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:16.48 stress
 2723 ec2-user  20   0  138656  21424    212 R  14.3  2.2   0:16.50 stress
 2724 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:16.50 stress
 2725 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:16.50 stress
 2726 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:16.50 stress
 2727 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:16.51 stress
 2728 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:16.50 stress
 2730 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:16.52 stress
 2731 ec2-user  20   0    7580     96      0 R  14.3  0.0   0:16.51 stress
 2719 ec2-user  20   0    7580     96      0 R  14.0  0.0   0:16.47 stress
 2720 ec2-user  20   0  138656  83200    212 R  14.0  8.6   0:16.47 stress
 2729 ec2-user  20   0    7580     96      0 R  14.0  0.0   0:16.50 stress
    1 root      20   0  123620   5476   3856 S   0.0  0.6   0:00.99 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kthreadd
    4 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/0+
```

Figure: The command prompt shows a high CPU usage after running a script. It shows the user as ec2-user the percentage of CPU at 14-14.3 and the command used was stress.

You can see that the process you just ran has a high CPU usage. The script is designed to run for 6 minutes, before stopping.

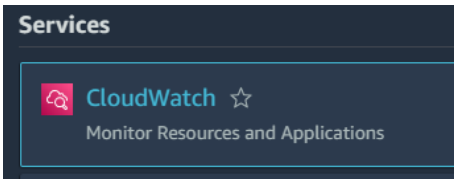
In the next steps you open the **AWS Management Console** and start the **AWS CloudWatch** application that will give you better insight into you EC2 instance.

32. On the top right of you screen select the `AWS` button. This displays the **AWS Management Console** in a new tab.



33. In the Search bar on the top of the screen, enter **CloudWatch** and press ENTER.

Expected Output



The AWS console includes a search bar that you can use to search for services.

34. On the left section of the navigation pane, select **Dashboard**, then select **Automatic dashboards**. In the **Automatic dashboards** list, select **EC2**

Expected Output

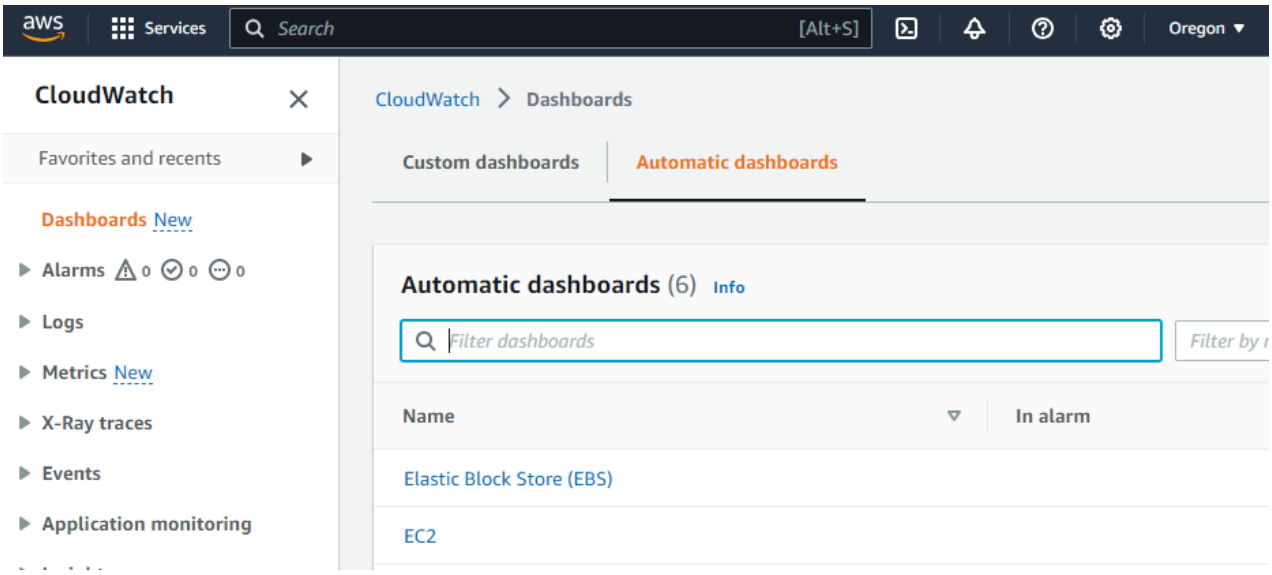


Figure: The CloudWatch dashboards shows CloudWatch Events, CloudWatch Logs, and EC2 as the top three dashboards.

This opens up the **EC2 dashboard** created for you by **AWS**.

Expected Output

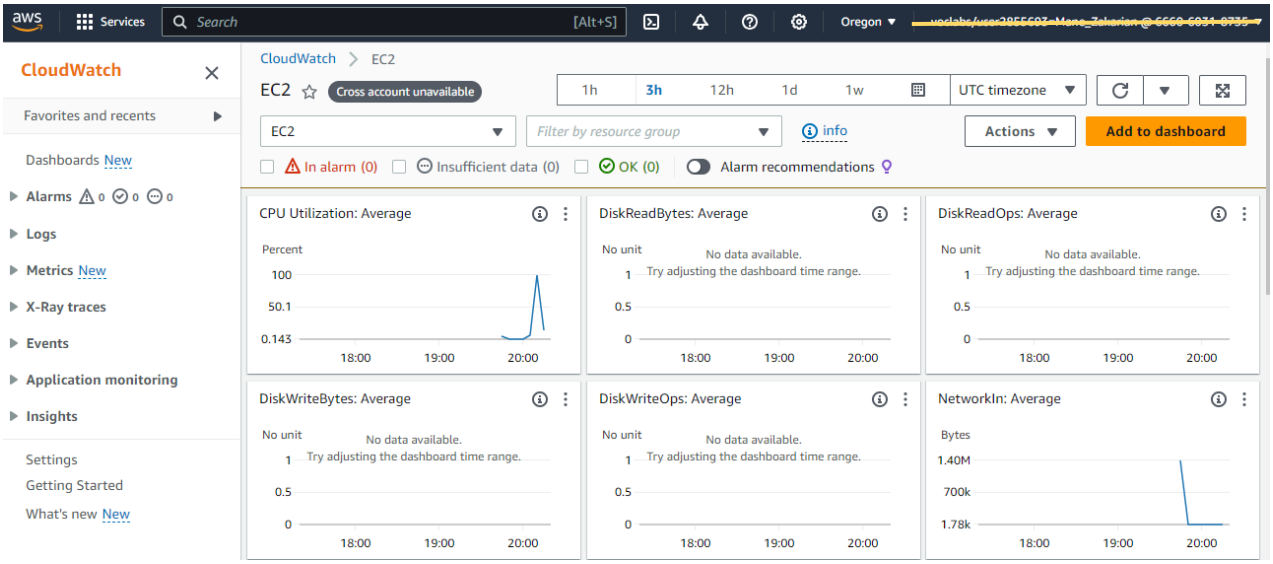
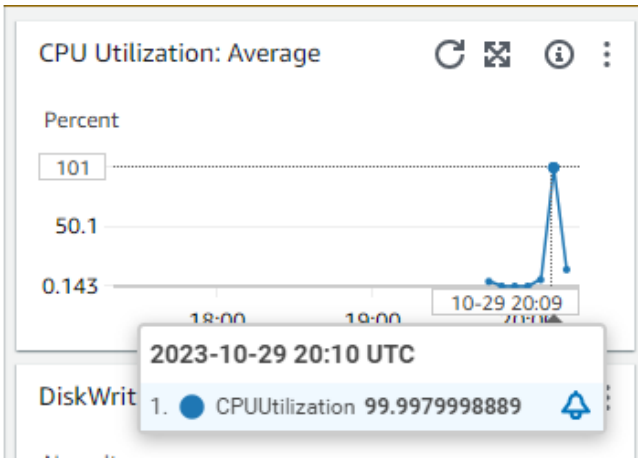


Figure: The graphs shown are CPU Utilization, DiskReadBytes, DiskReadOps, DiskWriteBytes, DiskWriteOps, and NetworkIn for the account's EC2 instances.



You can see that by default the **EC2 CloudWatch dashboard** displays several metrics such as the CPU utilization, Disk reads and writes....

You can see a spike in the CPU utilization that matches the time when you started the stress script earlier.

> **Note** > > Dashboards are customizable so you can add or remove widgets, > reorganize them, customize colors... **AWS CloudWatch** offers many > more features such as alarms or events triggers that you will discover > later that makes it a key **AWS Service** to monitor your applications > in real time. Update the 5 minutes average to 1 second to second review > updates more quickly.

35. Wait 5 minutes and go back to the AWS CloudWatch dashboard. You see that the CPU utilization dropped

    Expected Output

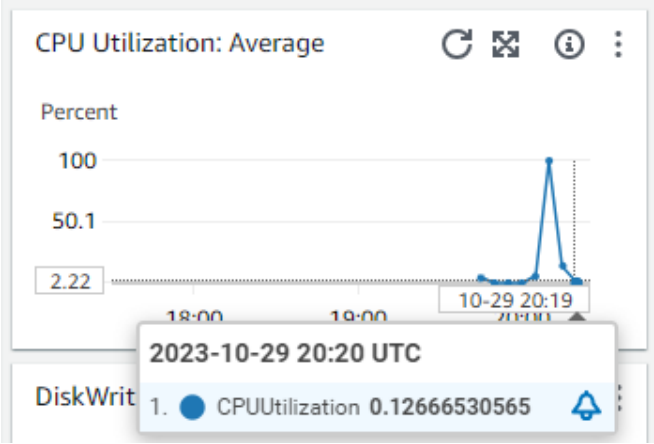
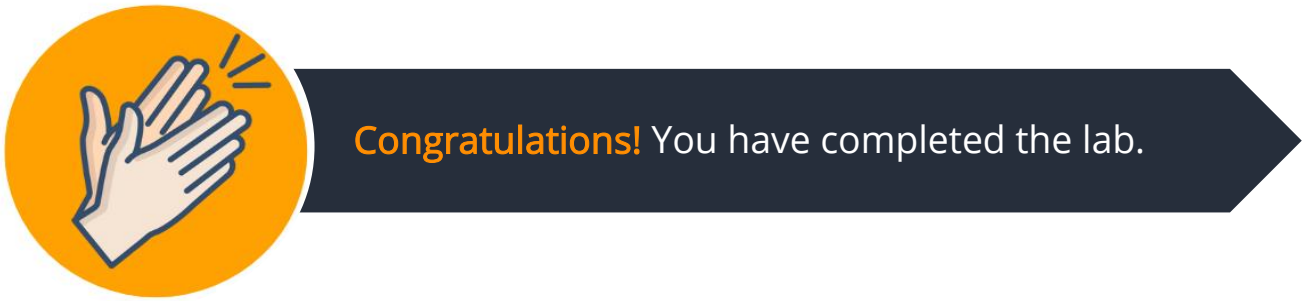


Figure: The graph shown is the CPU Utilization Average. In this average, the highest percent it reaches is 62.6 percent at around 8 minutes and hovers around 33 percent then drops back down right aftger the 9 minute mark.

> **Note** > > By default AWS CloudWatch aggregates data for 5 minutes before processing them. This is setup that can be changed

Lab Complete



36. Select End Lab at the top of this page and then select Yes to confirm that you want to end the lab. A panel will appear, indicating that "DELETE has been initiated... You may close this message box now."
37. Select the X in the top right corner to close the panel.

Commands Used:

On this lab we used several commands to perform different tasks. Here is a summary of the commands used:

Command	*Description*
pwd	Prints the current working directory
sudo	Execute commands with superuser (administrator) privileges.
top	Displays information about the processes that are currently using system resources, including CPU and memory usage.

What we learned?

In this lab, we learned how to manage and monitor an Amazon Linux 2 EC2 instance. We started by checking, starting, and verifying the Apache HTTP Server (httpd) service. Then, we explored system monitoring using the top command and introduced AWS CloudWatch for advanced monitoring and custom dashboards. This lab equipped us with essential skills for managing and monitoring EC2 instances in AWS and understanding system performance.