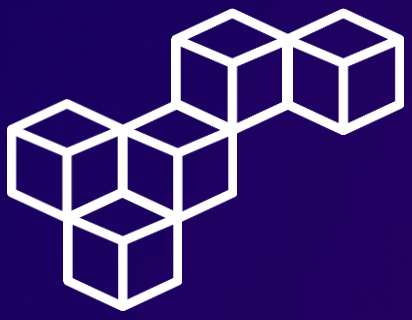
AWS re:Start

LAB

Managing Services

WEEK 2

aws re/start



Overview

Managing services involves overseeing software applications within a system for optimal functionality, including starting, stopping, and monitoring services efficiently. Tools like top provide real-time insights into resource usage, while platforms such as AWS CloudWatch offer comprehensive monitoring and management features for AWS resources and applications, ensuring system stability.

In Linux systems, the systemctl command is vital for service management via systemd, allowing administrators to control service statuses, start or stop services, and manage configurations effectively. Integrating systemctl with monitoring tools and cloud solutions creates a robust toolkit for service supervision, ensuring smooth system performance.

Note: This lab was made using Windows Subsystem for Linux.

Topics covered

- 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

Task 1

Use SSH to connect to an Amazon Linux EC2 instance

Initial Preparations

In the AWS Management Console, select the EC2 instance and make note of the **Public IPv4 address**.

Download the **private key file** labsuser.pem. Change to the Downloads directory and modify the permissions on the key to be read-only (r--------).

Connect to the instance using SSH

Establish a connection to the EC2 instance using the ssh command, the key and the instance's public IPv4 address.

```
support@HP-Pavilion-Laptop:~/Downloads$ ssh -i labsuser.pem ec2-user@34.219.150.124
The authenticity of host '34.219.150.124 (34.219.150.124)' can't be established.
ED25519 key fingerprint is SHA256:Ll2v7E7covSnBiamtcfAPwPIuILpUYrPVmh6lPHH8GU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '34.219.150.124' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2025-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-10-0-10-165 ~]$
```

aws re/start

Task 2

Check the Status of the httpd Service

Step 1: Check the httpd service status

Check the status of the httpd service with the systemctl status command. The output says that the httpd service is inactive.

```
[ec2-user@ip-10-0-10-165 ~]$ sudo systemctl status httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
   Active: inactive (dead)
     Docs: man:httpd.service(8)
[ec2-user@ip-10-0-10-165 ~]$
```

Step 2: Start the httpd service

Start the httpd service with the systemctl start command. Then, check the status of the httpd service again with the systemctl status command. The output says that the httpd service is now active (running).

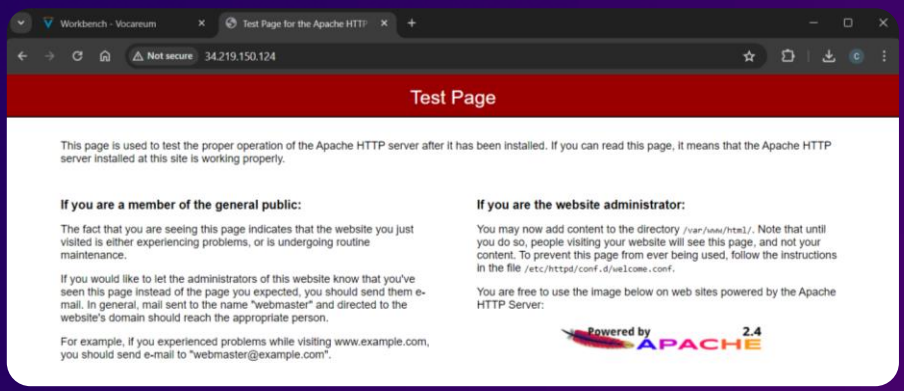
```
[ec2-user@ip-10-0-10-165 ~]$ sudo systemctl start httpd.service
[ec2-user@ip-10-0-10-165 ~]$ sudo systemctl status httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
   Active: active (running) since Tue 2024-04-09 20:33:47 UTC; 29s ago
     Docs: man:httpd.service(8)
 Main PID: 2591 (httpd)
   Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec:   0 B/sec"
   CGroup: /system.slice/httpd.service
           ├─2591 /usr/sbin/httpd -DFOREGROUND
           ├─2592 /usr/sbin/httpd -DFOREGROUND
           ├─2594 /usr/sbin/httpd -DFOREGROUND
           ├─2599 /usr/sbin/httpd -DFOREGROUND
           ├─2601 /usr/sbin/httpd -DFOREGROUND
           └─2606 /usr/sbin/httpd -DFOREGROUND

Apr 09 20:33:47 ip-10-0-10-165.us-west-2.compute.internal systemd[1]: Starting The Apache HTTP Server...
Apr 09 20:33:47 ip-10-0-10-165.us-west-2.compute.internal systemd[1]: Started The Apache HTTP Server.
[ec2-user@ip-10-0-10-165 ~]$
```

Task 2

Check the Status of the httpd Service

Step 3: Check the httpd service

On your browser, visit http://34.219.150.124 to check if the httpd web service works correctly.



Step 4: Stop the httpd service

Stop the httpd service with the systemctl stop command. Then, check the status of the httpd service again with the systemctl status command. The output says that the httpd service is now inactive (dead).

```
[ec2-user@ip-10-0-10-165 ~]$ sudo systemctl stop httpd.service
[ec2-user@ip-10-0-10-165 ~]$ sudo systemctl status httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
   Active: inactive (dead)
     Docs: man:httpd.service(8)

Apr 09 20:33:47 ip-10-0-10-165.us-west-2.compute.internal systemd[1]: Starting The Apache HTTP Server...
Apr 09 20:33:47 ip-10-0-10-165.us-west-2.compute.internal systemd[1]: Started The Apache HTTP Server.
Apr 09 20:37:17 ip-10-0-10-165.us-west-2.compute.internal systemd[1]: Stopping The Apache HTTP Server...
Apr 09 20:37:18 ip-10-0-10-165.us-west-2.compute.internal systemd[1]: Stopped The Apache HTTP Server.
[ec2-user@ip-10-0-10-165 ~]$
```

aws re/start

Task 3

Monitoring a Linux EC2 instance

Step 1: The top command

The top command displays the processes currently running as well as the resource usage like CPU usage and memory usage.

```
[ec2-user@ip-10-0-10-165 ~]$ top
top - 20:38:51 up 7 min,  1 user,  load average: 0.00, 0.07, 0.06
Tasks:  87 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 st
KiB Mem :   966808 total,   444688 free,    76676 used,   445444 buff/cache
KiB Swap:        0 total,        0 free,        0 used.   747888 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
    1 root      20   0  123620   5452   3848 S   0.0  0.6   0:01.00 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kthreadd
```

Step 2: Simulate a heavy workload

Run the **stress.sh** script to simulate a heavy workload on the EC2 instance, and simultaneously execute the top command to monitor system resources. The command prompt shows a high CPU usage after running the script.

```
[ec2-user@ip-10-0-10-165 ~]$ ./stress.sh & top
[1] 2662
stress: info: [2664] dispatching hogs: 8 cpu, 4 io, 2 vm, 0 hdd
top - 20:40:59 up 9 min,  1 user,  load average: 11.37, 4.03, 1.49
Tasks: 105 total,  15 running,  49 sleeping,   0 stopped,   0 zombie
%Cpu(s): 62.1 us, 37.9 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
KiB Mem :   966808 total,   254516 free,   266688 used,   445604 buff/cache
KiB Swap:        0 total,        0 free,        0 used.   557872 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 2672 ec2-user  20   0    7580     92      0 R  14.7  0.0   0:13.36 stress
 2665 ec2-user  20   0    7580     92      0 R  14.3  0.0   0:13.35 stress
 2666 ec2-user  20   0    7580     92      0 R  14.3  0.0   0:13.34 stress
 2667 ec2-user  20   0  138656  94080    272 R  14.3  9.7   0:13.34 stress
 2668 ec2-user  20   0    7580     92      0 R  14.3  0.0   0:13.35 stress
 2669 ec2-user  20   0    7580     92      0 R  14.3  0.0   0:13.34 stress
 2670 ec2-user  20   0  138656  94080    272 R  14.3  9.7   0:13.35 stress
 2675 ec2-user  20   0    7580     92      0 R  14.3  0.0   0:13.36 stress
 2676 ec2-user  20   0    7580     92      0 R  14.3  0.0   0:13.36 stress
 2677 ec2-user  20   0    7580     92      0 R  14.3  0.0   0:13.45 stress
 2678 ec2-user  20   0    7580     92      0 R  14.3  0.0   0:13.36 stress
 2671 ec2-user  20   0    7580     92      0 R  14.0  0.0   0:13.34 stress
 2673 ec2-user  20   0    7580     92      0 R  14.0  0.0   0:13.35 stress
 2674 ec2-user  20   0    7580     92      0 R  14.0  0.0   0:13.35 stress
```

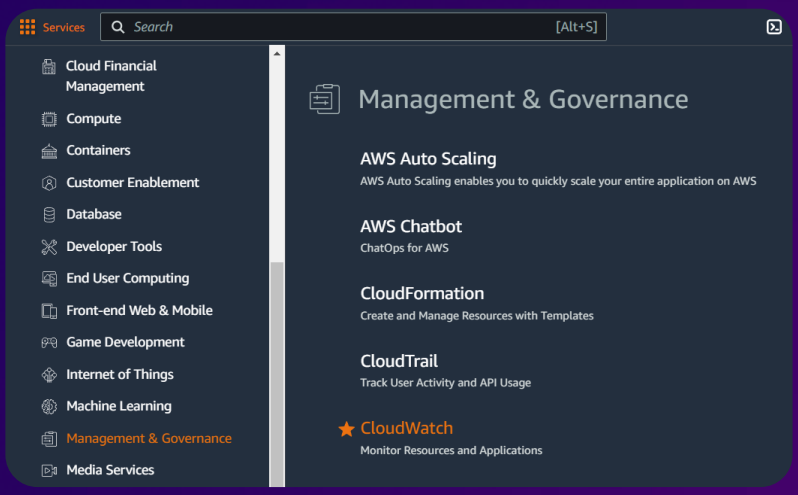
aws re/start

Task 3

Monitoring a Linux EC2 instance

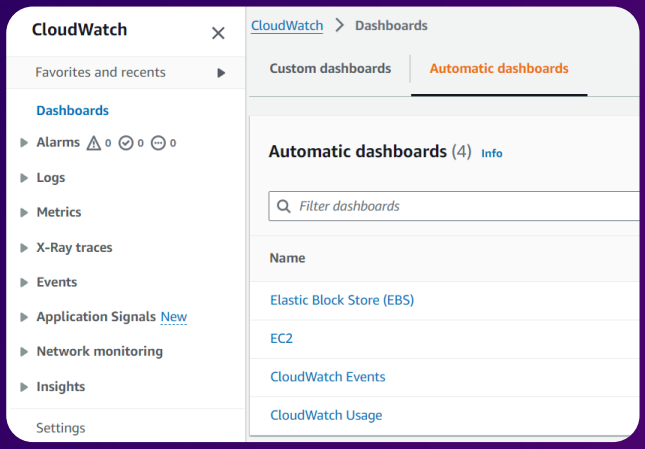
Step 3: AWS CloudWatch

Open the AWS Management Console and start AWS CloudWatch.



Step 4: Automatic dashboards

Within CloudWatch's automatic dashboards, select EC2.



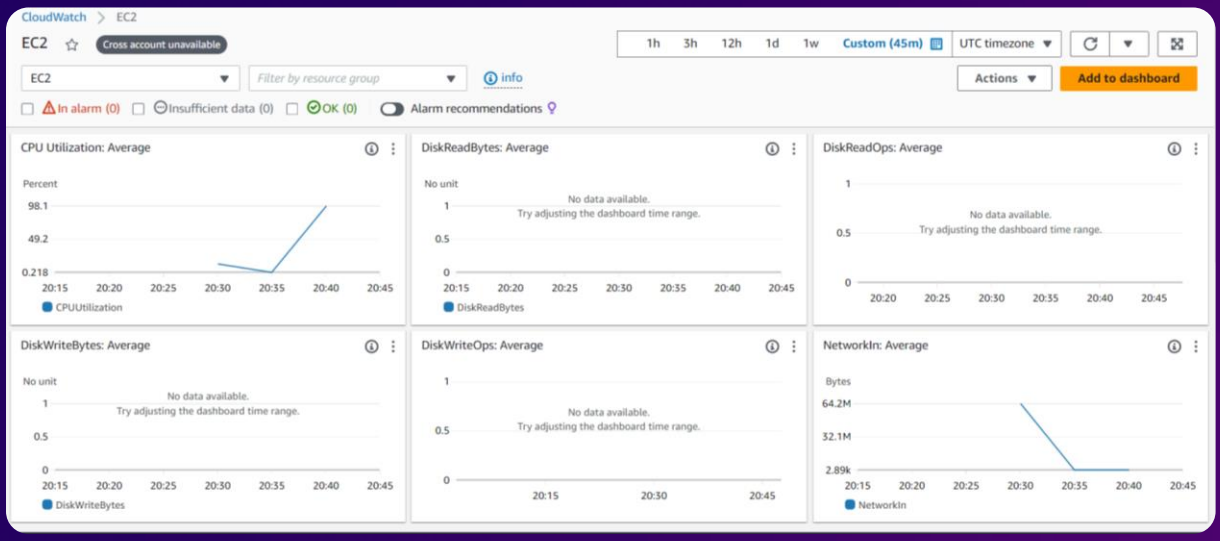
aws re/start

Task 3

Monitoring a Linux EC2 instance

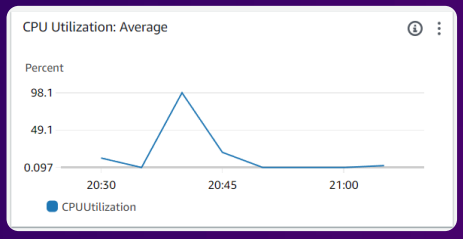
Step 5: EC2 CloudWatch dashboard

The EC2 CloudWatch dashboard displays several metrics such as the CPU utilization, network traffic, and disk I/O operations.

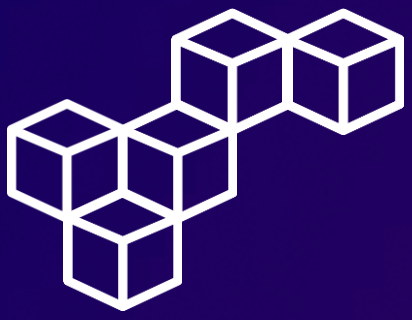


Step 6: Monitor CPU utilization

You can observe a spike in CPU utilization that correlates with the time when the stress script was initiated earlier, followed by a decrease in CPU utilization a few minutes later.



aws re/start



Conclusions

Managing services
Managing services involves overseeing software operations for optimal functionality within a system, including starting, stopping, and monitoring services.

The systemctl command
The systemctl command in Linux is vital for controlling service statuses, starting or stopping services, and managing configurations effectively.

Monitoring services
Monitoring services is essential for real-time insights into resource usage and performance.

The top command
The top command specifically helps monitor CPU and memory usage, aiding in identifying resource-intensive processes and performance bottlenecks.

AWS CloudWatch
AWS CloudWatch offers comprehensive monitoring and management for AWS resources, ensuring system stability and efficiency in cloud environments.

aws re/start

Cristhian Becerra

cristhian-becerra-espinoza

+51 951 634 354

cristhianbecerra99@gmail.com

Lima, Peru