AWS re:Start

LAB

Monitor an EC2 Instance

WEEK 4

aws re/start

Overview

Logging and monitoring are techniques implemented to achieve a common goal. They work together to help ensure that a system's performance baselines and security guidelines are always met.

Logging refers to recording and storing data events as log files. Logs contain low-level details that can give you visibility into how your application or system performs under certain circumstances. From a security standpoint, logging helps security administrators identify red flags that are easily overlooked in their system.

Monitoring is the process of analyzing and collecting data to help ensure optimal performance. Monitoring helps detect unauthorized access and helps align your services' usage with organizational security.

In this lab, you create an Amazon CloudWatch alarm that initiates when the Amazon Elastic Compute Cloud (Amazon EC2) instance exceeds a specific central processing unit (CPU) utilization threshold. You create a subscription using Amazon Simple Notification Service (Amazon SNS) that sends an email to you if this alarm goes off. You log in to the EC2 instance and run a stress test command that causes the CPU utilization of the EC2 instance to reach 100 percent.

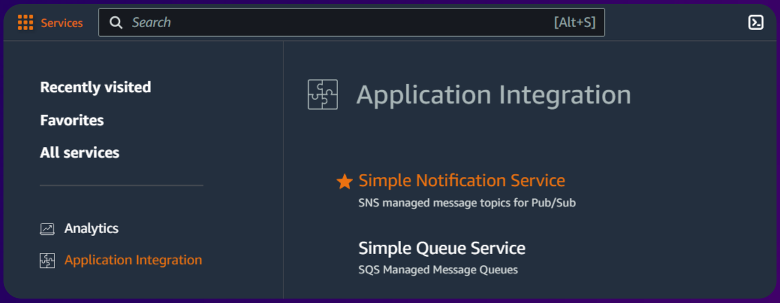
This test simulates a malicious actor gaining control of the EC2 instance and spiking the CPU. CPU spiking has various possible causes, one of which is malware.

Task 1

Configure Amazon SNS

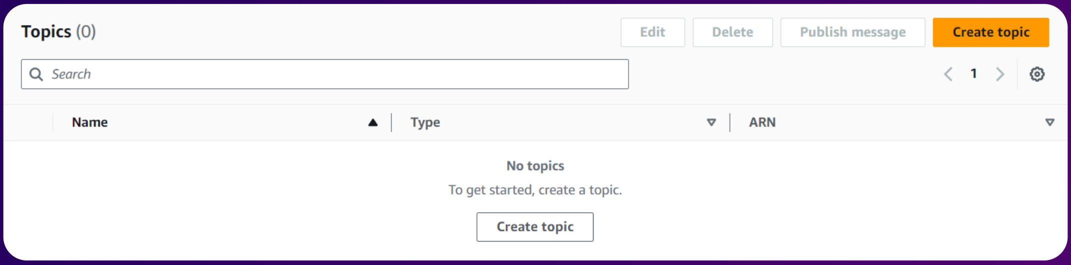
Step 1: Access the Simple Notification Service

Open the AWS Management Console, and select Simple Notification Service.



Step 2: Create topic

Navigate to the **Topics** section and select Create topic.



No topics

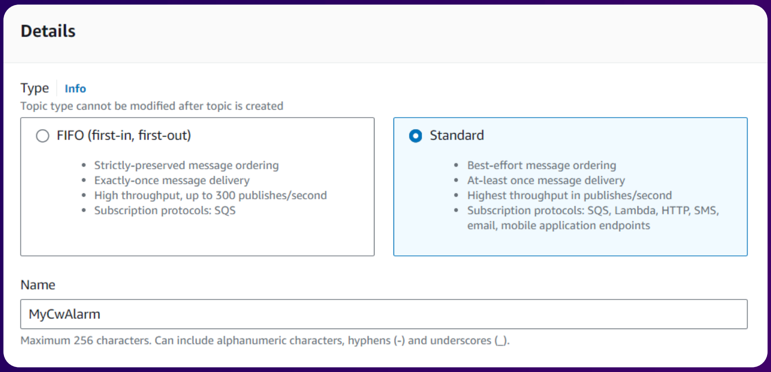
To get started, create a topic.

Task 1

Configure Amazon SNS

Step 3: Topic Details

On the **Create topic** page in the **Details** section, configure the following options.



Details

Type Info
Topic type cannot be modified after topic is created

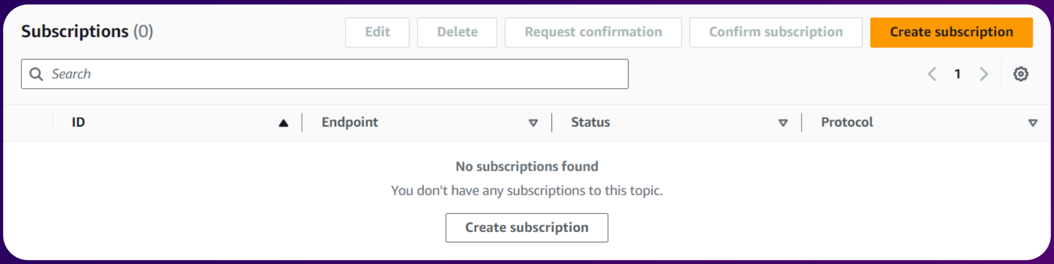
○ FIFO (first-in, first-out)
- Strictly-preserved message ordering
- Exactly-once message delivery
- High throughput, up to 300 publishes/second
- Subscription protocols: SQS

● Standard
- Best-effort message ordering
- At-least once message delivery
- Highest throughput in publishes/second
- Subscription protocols: SQS, Lambda, HTTP, SMS, email, mobile application endpoints

Name
MyCwAlarm
Maximum 256 characters. Can include alphanumeric characters, hyphens (-) and underscores (_).

Step 4: Create subscription

On the **MyCwAlarm** details page, choose the **Subscriptions** tab, and then choose Create subscription.



Subscriptions (0)

ID	Endpoint	Status	Protocol

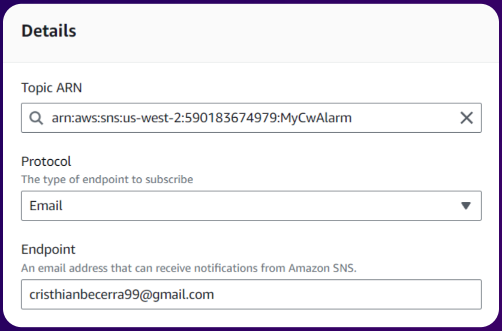
No subscriptions found
You don't have any subscriptions to this topic.

Task 1

Configure Amazon SNS

Step 5: Subscription Details

On the **Create subscription** page in the **Details** section, configure the following options.



Details

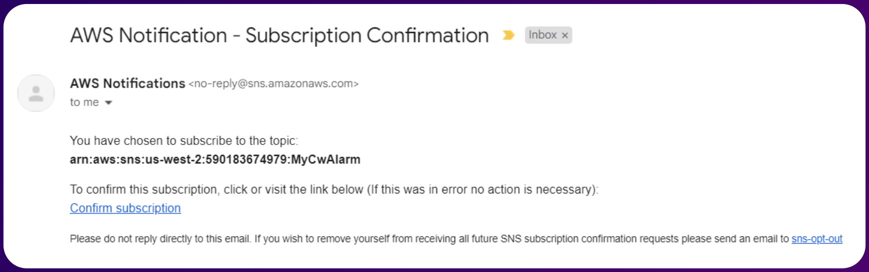
Topic ARN
arn:aws:sns:us-west-2:590183674979:MyCwAlarm

Protocol
The type of endpoint to subscribe
Email

Endpoint
An email address that can receive notifications from Amazon SNS.
cristhianbecerra99@gmail.com

Step 6: Subscription Confirmation

Open the email that you received with the Amazon SNS subscription notification, and choose Confirm subscription.



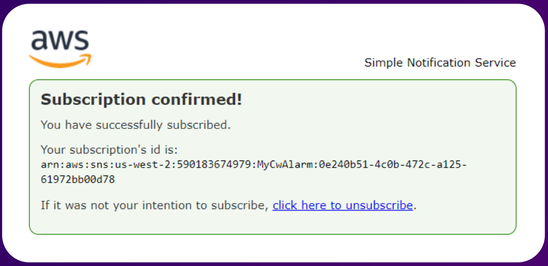
AWS Notification - Subscription Confirmation Inbox ×

AWS Notifications <no-reply@sns.amazonaws.com>
to me

You have chosen to subscribe to the topic:
arn:aws:sns:us-west-2:590183674979:MyCwAlarm

To confirm this subscription, click or visit the link below (If this was in error no action is necessary):
Confirm subscription

Please do not reply directly to this email. If you wish to remove yourself from receiving all future SNS subscription confirmation requests please send an email to sns-opt-out



aws Simple Notification Service

Subscription confirmed!

You have successfully subscribed.

Your subscription's id is:
arn:aws:sns:us-west-2:590183674979:MyCwAlarm:0e240b51-4c0b-472c-a125-61972bb00d78

If it was not your intention to subscribe, click here to unsubscribe.

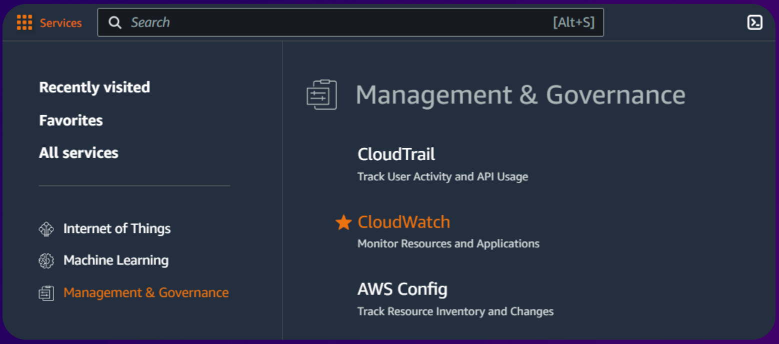
aws re/start

Task 2

Create a CloudWatch alarm

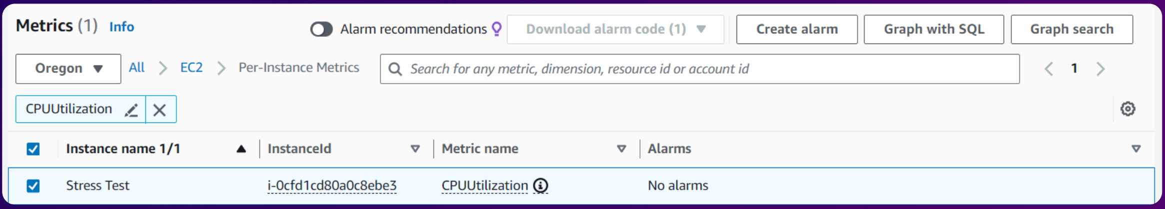
Step 1: Access the CloudWatch console

Open the AWS Management Console, and select CloudWatch.



Step 2: Review Metrics

On the **Metrics** page, choose EC2, and choose Per-Instance Metrics. Select the CPUUtilization metric for the Stress Test EC2 instance.



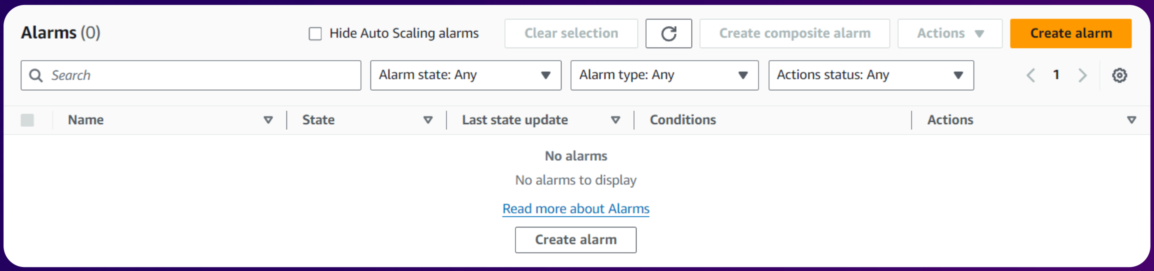
✓	Instance name 1/1	InstanceId	Metric name	Alarms
✓	Stress Test	i-0cfd1cd80a0c8ebe3	CPUUtilization	No alarms

Task 2

Create a CloudWatch alarm

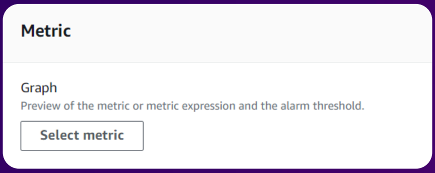
Step 3: Create alarm

Navigate to the **Alarms** section and select Create alarm.



Step 4: Select metric

On the **Metric** section, choose EC2, and choose Per-Instance Metrics. Select the CPUUtilization metric for the Stress Test EC2 instance.

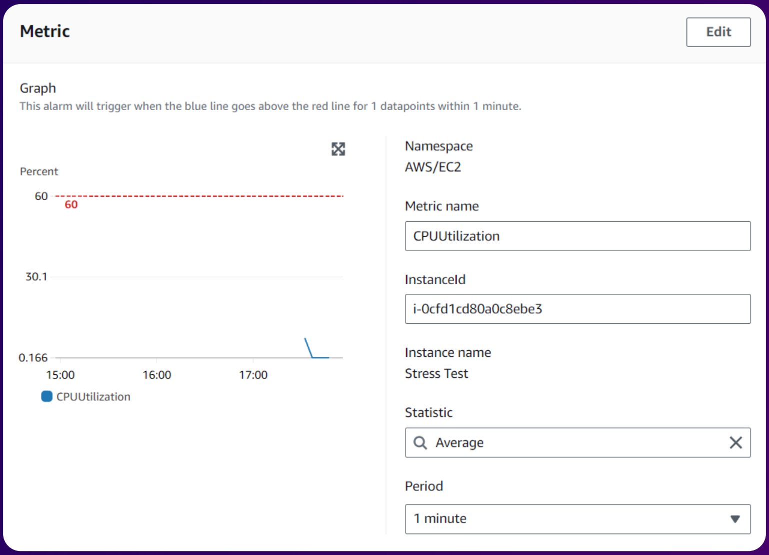

Metric
Graph
Preview of the metric or metric expression and the alarm threshold.
Select metric

Task 2

Create a CloudWatch alarm

Step 5: Specify metric and conditions

On the **Specify metric and conditions** page, configure the following options.



Metric

Graph
This alarm will trigger when the blue line goes above the red line for 1 datapoints within 1 minute.

Namespace: AWS/EC2

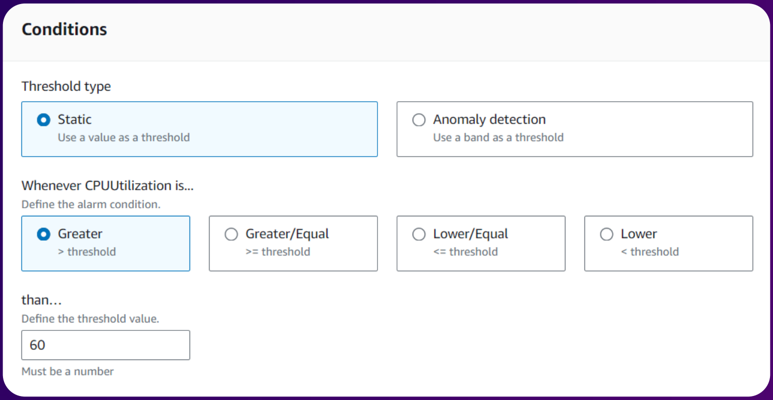
Metric name: CPUUtilization

InstanceId: i-0cfd1cd80a0c8ebe3

Instance name: Stress Test

Statistic: Average

Period: 1 minute



Conditions

Threshold type: [✓ Static — Use a value as a threshold] Anomaly detection — Use a band as a threshold

Whenever CPUUtilization is...
Define the alarm condition.
[✓ Greater — > threshold] Greater/Equal — >= threshold; Lower/Equal — <= threshold; Lower — < threshold

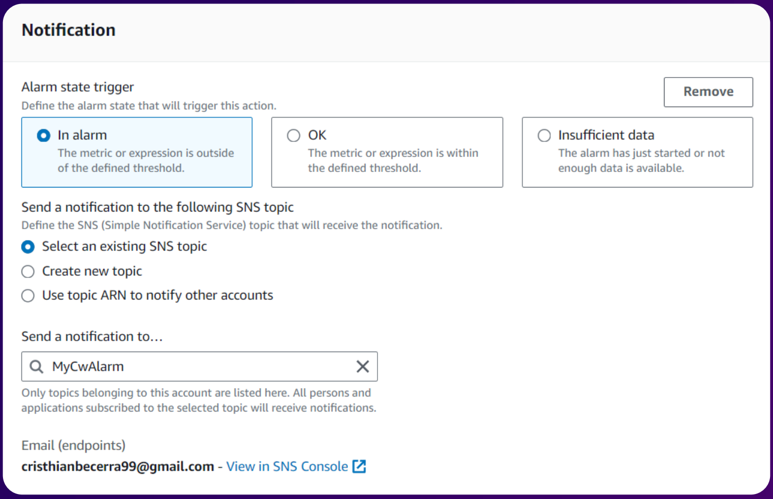
than...
Define the threshold value.
60
Must be a number

Task 2

Create a CloudWatch alarm

Step 6: Configure actions

On the **Notification** section, configure the following options.



Notification

Alarm state trigger
Define the alarm state that will trigger this action.

[✓ In alarm — The metric or expression is outside of the defined threshold.] OK — The metric or expression is within the defined threshold. Insufficient data — The alarm has just started or not enough data is available.

Send a notification to the following SNS topic
Define the SNS (Simple Notification Service) topic that will receive the notification.

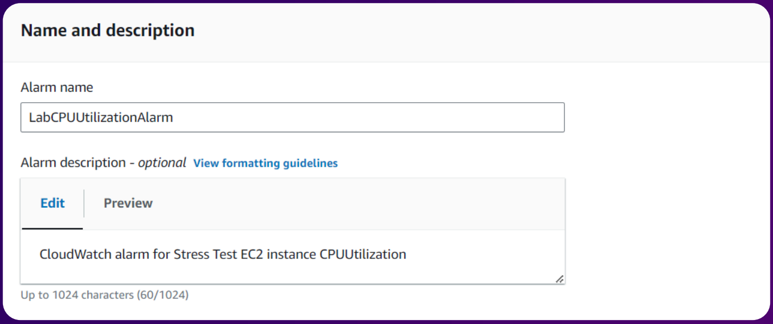
[✓ Select an existing SNS topic] Create new topic / Use topic ARN to notify other accounts

Send a notification to...
MyCwAlarm
Only topics belonging to this account are listed here. All persons and applications subscribed to the selected topic will receive notifications.

Email (endpoints)
cristhianbecerra99@gmail.com - View in SNS Console

Step 7: Add name and description

On the **Name and description** section, configure the following options.



Name and description

Alarm name: LabCPUUtilizationAlarm

Alarm description - *optional* View formatting guidelines

CloudWatch alarm for Stress Test EC2 instance CPUUtilization

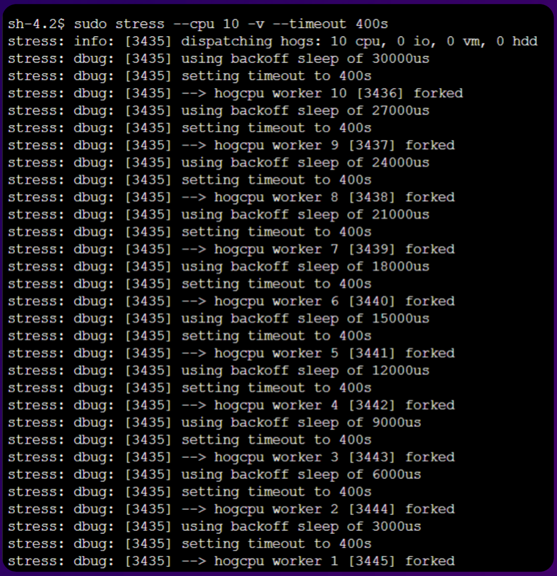
Up to 1024 characters (60/1024)

Task 3

Test the CloudWatch alarm

Step 1: Increase the CPU load

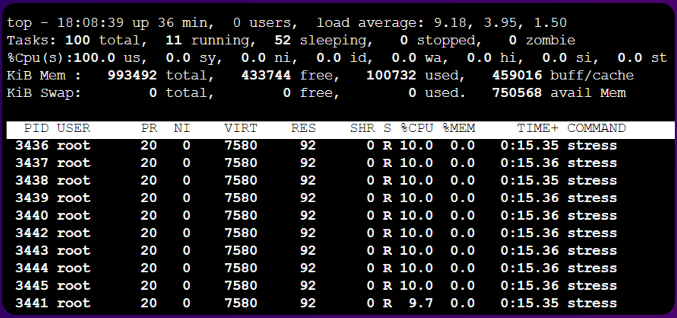
Log in to the Stress Test EC2 instance and run the following command to manually stress the CPU load to 100 percent.



```
sh-4.2$ sudo stress --cpu 10 -v --timeout 400s
stress: info: [3435] dispatching hogs: 10 cpu, 0 io, 0 vm, 0 hdd
stress: dbug: [3435] using backoff sleep of 30000us
stress: dbug: [3435] setting timeout to 400s
stress: dbug: [3435] --> hogcpu worker 10 [3436] forked
stress: dbug: [3435] using backoff sleep of 27000us
stress: dbug: [3435] setting timeout to 400s
stress: dbug: [3435] --> hogcpu worker 9 [3437] forked
stress: dbug: [3435] using backoff sleep of 24000us
stress: dbug: [3435] setting timeout to 400s
stress: dbug: [3435] --> hogcpu worker 8 [3438] forked
stress: dbug: [3435] using backoff sleep of 21000us
stress: dbug: [3435] setting timeout to 400s
stress: dbug: [3435] --> hogcpu worker 7 [3439] forked
stress: dbug: [3435] using backoff sleep of 18000us
stress: dbug: [3435] setting timeout to 400s
stress: dbug: [3435] --> hogcpu worker 6 [3440] forked
stress: dbug: [3435] using backoff sleep of 15000us
stress: dbug: [3435] setting timeout to 400s
stress: dbug: [3435] --> hogcpu worker 5 [3441] forked
stress: dbug: [3435] using backoff sleep of 12000us
stress: dbug: [3435] setting timeout to 400s
stress: dbug: [3435] --> hogcpu worker 4 [3442] forked
stress: dbug: [3435] using backoff sleep of 9000us
stress: dbug: [3435] setting timeout to 400s
stress: dbug: [3435] --> hogcpu worker 3 [3443] forked
stress: dbug: [3435] using backoff sleep of 6000us
stress: dbug: [3435] setting timeout to 400s
stress: dbug: [3435] --> hogcpu worker 2 [3444] forked
stress: dbug: [3435] using backoff sleep of 3000us
stress: dbug: [3435] setting timeout to 400s
stress: dbug: [3435] --> hogcpu worker 1 [3445] forked
```

Step 2: Review CPU usage

Run the top command to show the live CPU usage.



```
top - 18:08:39 up 36 min,  0 users,  load average: 9.18, 3.95, 1.50
Tasks: 100 total,  11 running,  52 sleeping,   0 stopped,   0 zombie
%Cpu(s):100.0 us,  0.0 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   993492 total,   433744 free,   100732 used,   459016 buff/cache
KiB Swap:        0 total,        0 free,        0 used.   750568 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 3436 root      20   0    7580     92      0 R 10.0  0.0   0:15.35 stress
 3437 root      20   0    7580     92      0 R 10.0  0.0   0:15.36 stress
 3438 root      20   0    7580     92      0 R 10.0  0.0   0:15.35 stress
 3439 root      20   0    7580     92      0 R 10.0  0.0   0:15.36 stress
 3440 root      20   0    7580     92      0 R 10.0  0.0   0:15.36 stress
 3442 root      20   0    7580     92      0 R 10.0  0.0   0:15.36 stress
 3443 root      20   0    7580     92      0 R 10.0  0.0   0:15.36 stress
 3444 root      20   0    7580     92      0 R 10.0  0.0   0:15.36 stress
 3445 root      20   0    7580     92      0 R 10.0  0.0   0:15.36 stress
 3441 root      20   0    7580     92      0 R  9.7  0.0   0:15.35 stress
```

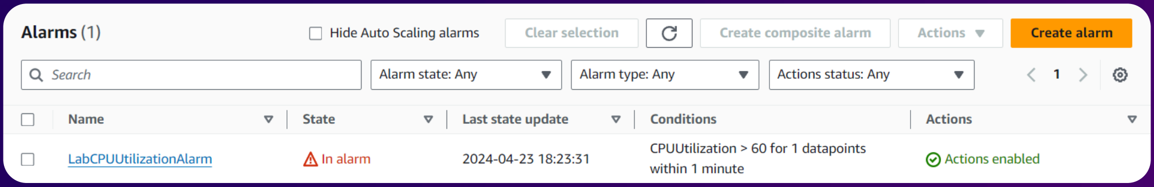
aws re/start

Task 3

Test the CloudWatch alarm

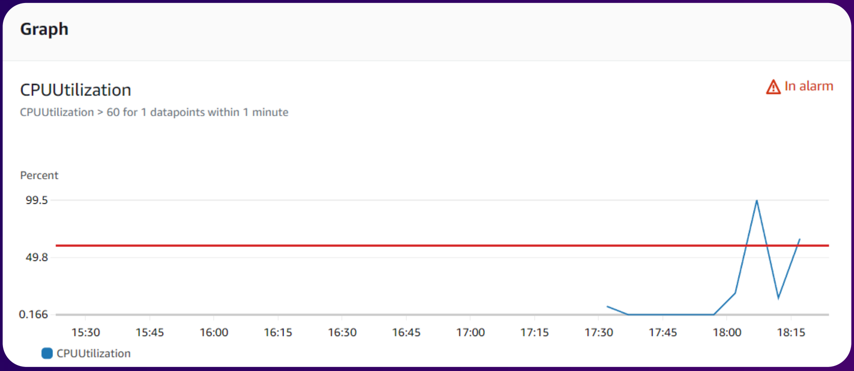
Step 3: Review alarm state

Review the LabCPUUtilizationAlarm. The alarm state is In alarm.



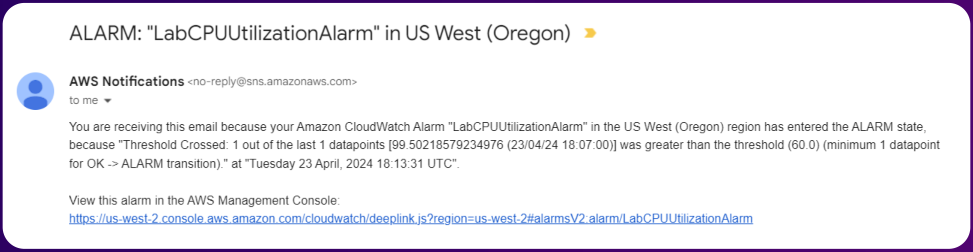
Alarms (1)

	Name	State	Last state update	Conditions	Actions
☐	LabCPUUtilizationAlarm	⚠ In alarm	2024-04-23 18:23:31	CPUUtilization > 60 for 1 datapoints within 1 minute	✓ Actions enabled



Step 4: Verify alarm notification

Review the new email from AWS Notifications in your inbox.



ALARM: "LabCPUUtilizationAlarm" in US West (Oregon)

AWS Notifications <no-reply@sns.amazonaws.com>
to me

You are receiving this email because your Amazon CloudWatch Alarm "LabCPUUtilizationAlarm" in the US West (Oregon) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [99.50218579234976 (23/04/24 18:07:00)] was greater than the threshold (60.0) (minimum 1 datapoint for OK -> ALARM transition)." at "Tuesday 23 April, 2024 18:13:31 UTC".

View this alarm in the AWS Management Console:
https://us-west-2.console.aws.amazon.com/cloudwatch/deeplink.js?region=us-west-2#alarmsV2:alarm/LabCPUUtilizationAlarm

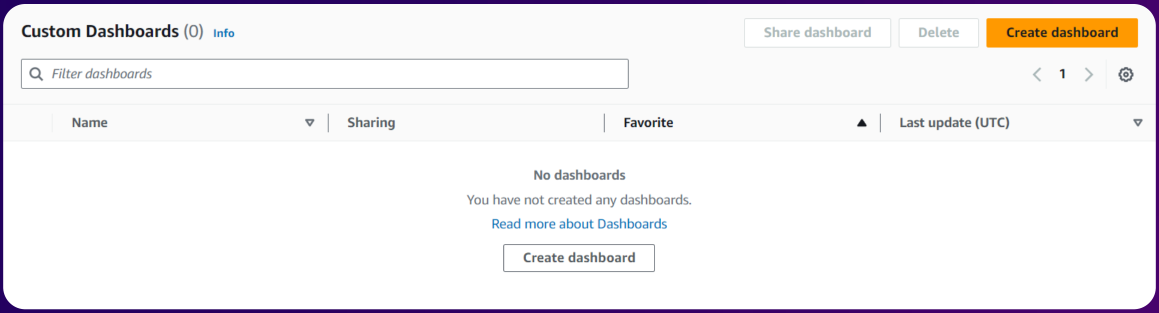
aws re/start

Task 4

Create a CloudWatch dashboard

Step 1: Create dashboard

Navigate to the **Dashboards** section and select Create dashboard.



Custom Dashboards (0) Info

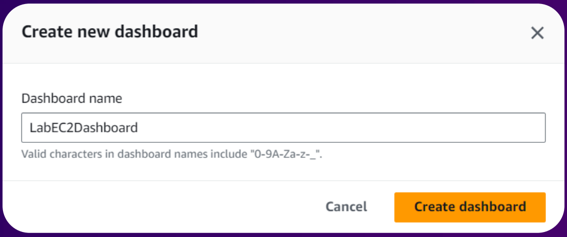
No dashboards

You have not created any dashboards.

Read more about Dashboards

Step 2: Create new dashboard

For Dashboard name, enter LabEC2Dashboard.



Create new dashboard

Dashboard name: LabEC2Dashboard

Valid characters in dashboard names include "0-9A-Za-z-_".

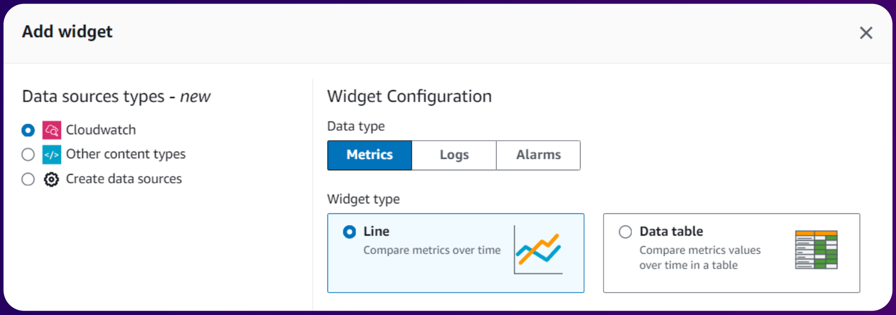
aws re/start

Task 4

Create a CloudWatch dashboard

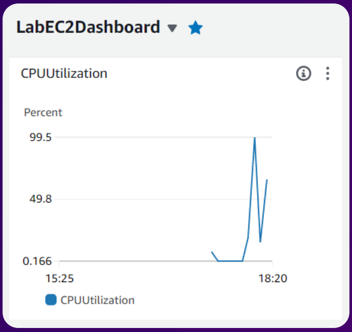
Step 3: Add widget

For Widget type, choose Line. For Metrics, choose EC2, and choose Per-Instance Metrics. Select the CPUUtilization metric for the Stress Test EC2 instance.



Step 4: Review dashboard

Now you have created a quick access shortcut to view the **CPUUtilization** metric for the Stress Test instance.



Conclusions

Simple Notification Service

Simple Notification Service (SNS) facilitates seamless communication by sending notifications via email, SMS, or other messaging protocols.

CloudWatch

CloudWatch serves as a monitoring and management service, providing insights into AWS resources and applications through logs, metrics, and alarms.

CloudWatch Metrics

CloudWatch Metrics collect and visualize data points for AWS resources, enabling real-time monitoring and analysis of performance metrics.

CloudWatch Alarms

CloudWatch Alarms notify users of specific conditions or thresholds within metrics, allowing for proactive response to potential issues or anomalies.

CloudWatch Dashboards

CloudWatch Dashboards provide customizable views and visualizations of metrics and alarms, offering a centralized platform for monitoring and analysis.

aws re/start

Cristhian Becerra

cristhian-becerra-espinoza

+51 951 634 354

cristhianbecerra99@gmail.com

Lima, Peru

aws re/start