

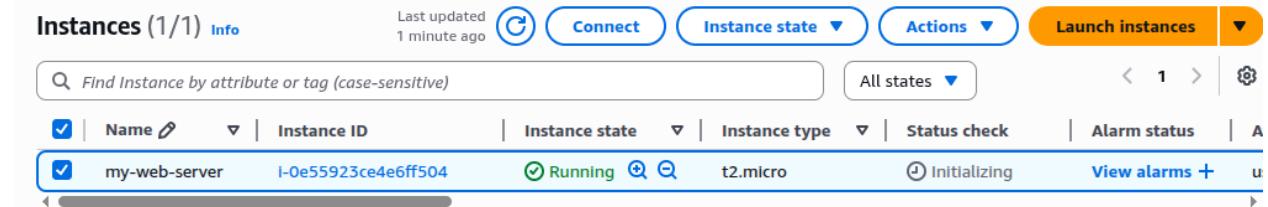
Implementation of CloudWatch CPU Alarm and SNS Email Notification

Overview

Amazon CloudWatch alarm that monitors the CPU usage of an EC2 instance. When the CPU exceeds a threshold (80%), the alarm triggers an SNS email notification so the support team can take action.

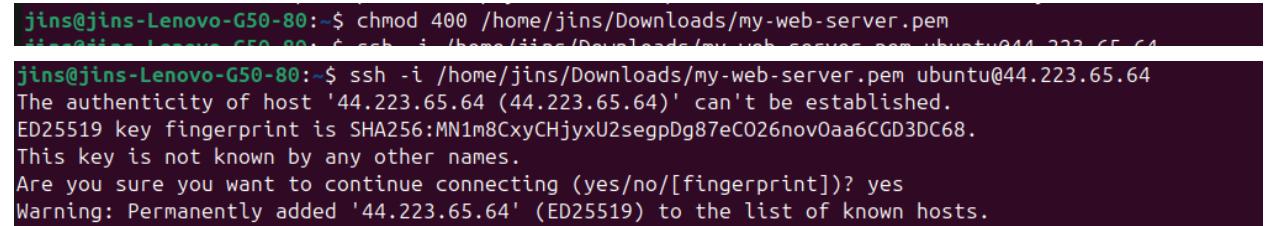
Prerequisites

- 1.AWS EC2 instance (t2.micro) with port 22 allowed (allow only your IP) for ssh access and port 80 allowed (for http traffic from the internet).
2. Select OS ubuntu 24.04
- 3.Create key value pair for ssh access change permission to 400 for maintaining best security practice.



The screenshot shows the AWS CloudWatch Instances console. At the top, there's a header with 'Instances (1/1)' and a 'Info' button. Below the header are several filters: 'Name' (with a search icon), 'Instance ID' (showing 'i-0e55923ce4e6ff504'), 'Instance state' (showing 'Running' with a green checkmark), 'Instance type' (showing 't2.micro'), 'Status check' (showing 'Initializing'), and 'Alarm status' (showing 'View alarms +'). A search bar at the top says 'Find Instance by attribute or tag (case-sensitive)' and a dropdown says 'All states'. On the right, there are buttons for 'Launch instances' and other navigation controls. Below the filters, a table lists the instance details.

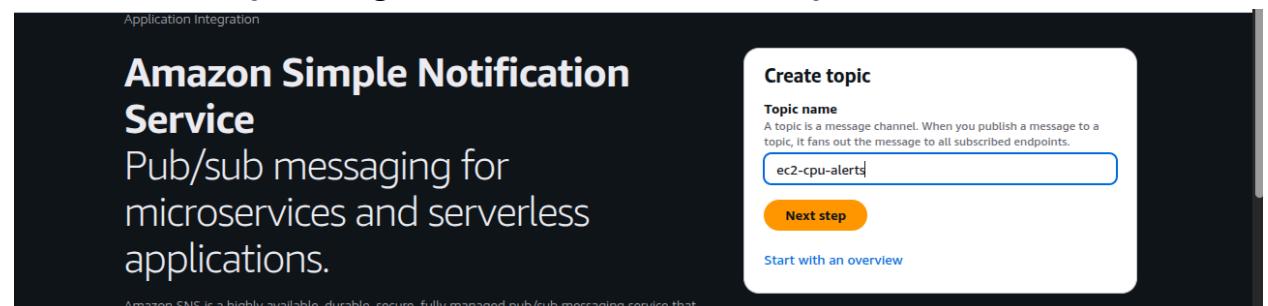
Name	Instance ID	Instance state	Instance type	Status check	Alarm status
my-web-server	i-0e55923ce4e6ff504	Running	t2.micro	Initializing	View alarms +



The terminal session starts with the command 'ssh -i /home/jins/Downloads/my-web-server.pem ubuntu@44.223.65.64'. It then displays a warning about the host fingerprint and asks if the user wants to continue connecting. The user responds with 'yes' and a warning message is shown about adding the host to the list of known hosts.

```
jins@jins-Lenovo-G50-80:~$ chmod 400 /home/jins/Downloads/my-web-server.pem
jins@jins-Lenovo-G50-80:~$ ssh -i /home/jins/Downloads/my-web-server.pem ubuntu@44.223.65.64
The authenticity of host '44.223.65.64 (44.223.65.64)' can't be established.
ED25519 key fingerprint is SHA256:MN1m8CxyCHjyxU2segpDg87eC026nov0aa6CGD3DC68.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '44.223.65.64' (ED25519) to the list of known hosts.
```

1.Create SNS topic using console and create subscription



The screenshot shows the 'Create topic' step of the AWS SNS wizard. The left side has a title 'Amazon Simple Notification Service' and a subtitle 'Pub/sub messaging for microservices and serverless applications.' The right side is a form with a 'Topic name' field containing 'ec2-cpu-alerts'. Below the field is a description: 'A topic is a message channel. When you publish a message to a topic, it fans out the message to all subscribed endpoints.' There are two buttons: 'Next step' and 'Start with an overview'.

While creating a subscription select protocol i.e the communication method. I have selected email and provided my email as the endpoint.

New Feature
Amazon SNS now supports High Throughput FIFO topics. [Learn more](#)

Create subscription

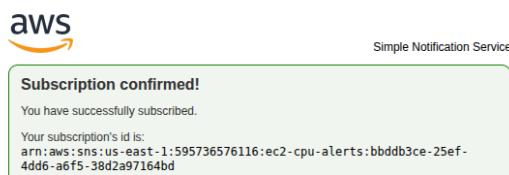
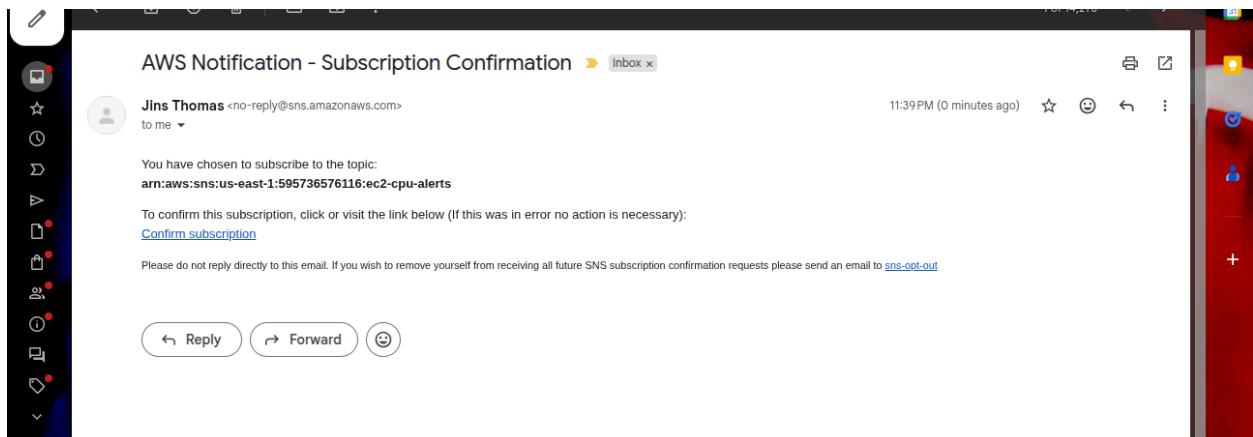
Details

Topic ARN
arn:aws:sns:us-east-1:595736576116:ec2-cpu-alerts

Protocol
The type of endpoint to subscribe
Email

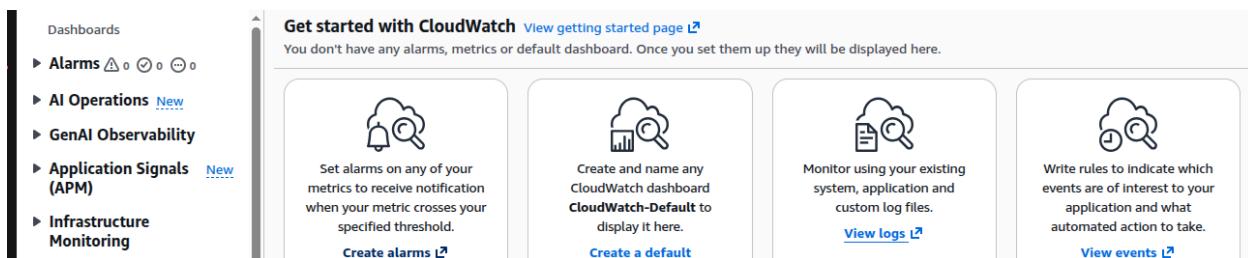
Endpoint
An email address that can receive notifications from Amazon SNS.
jinsthomas920@gmail.com

Once the subscription is created make sure you confirm the subscription received in your mail. Else AWS will not send the alert notification.



2.Creating the cloudwatch alarm

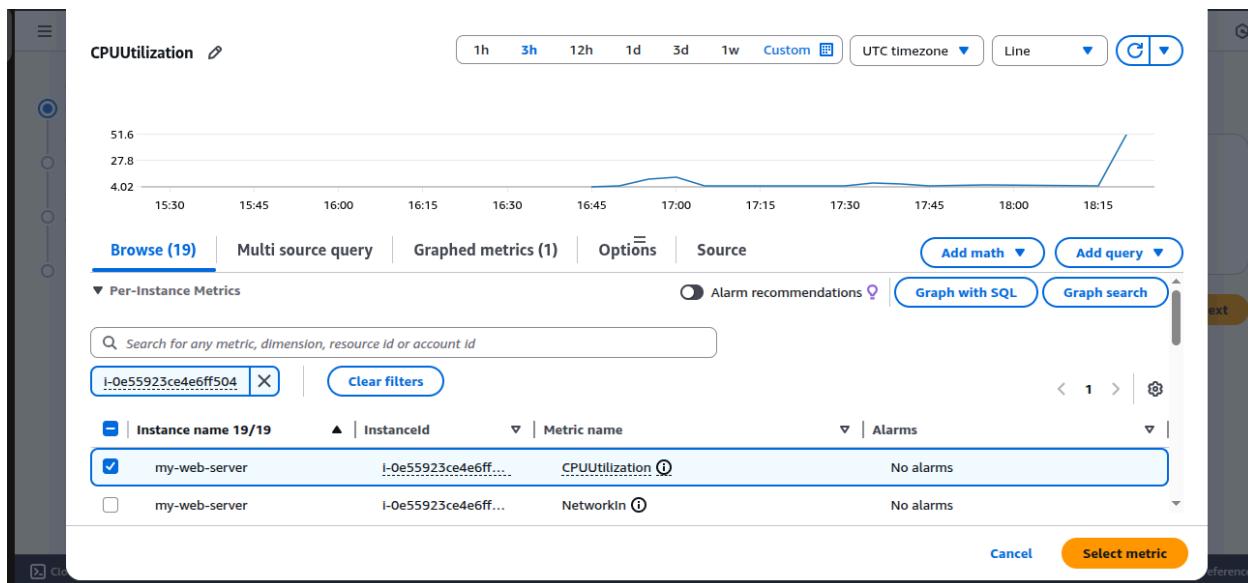
Select create alarm



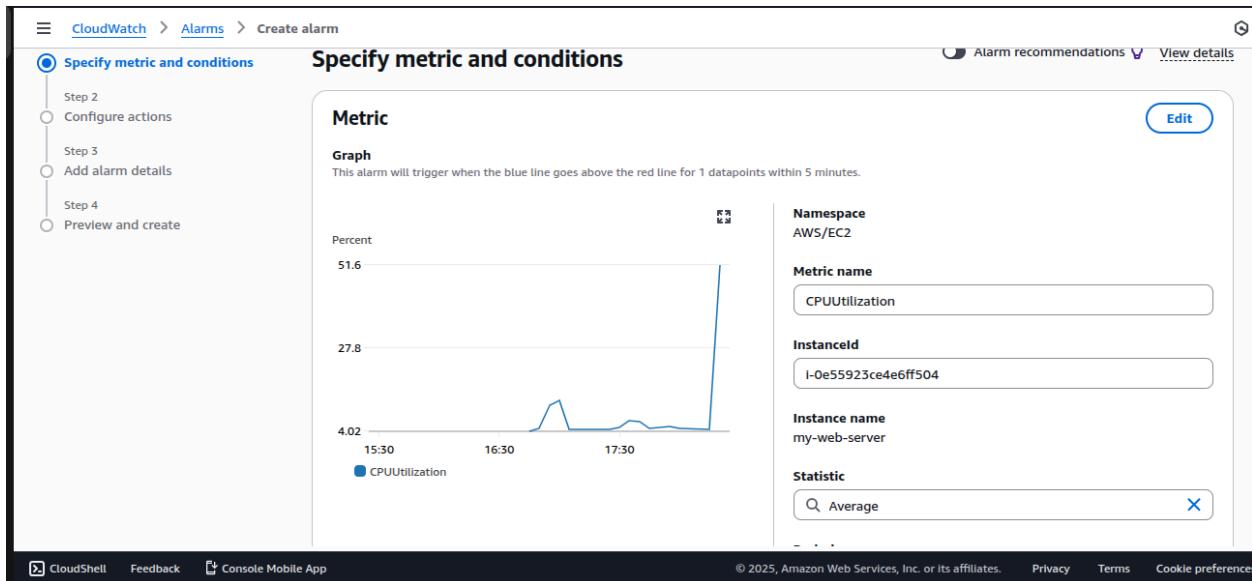
The screenshot shows the CloudWatch Metrics Dashboard. On the left, a sidebar lists navigation options: Dashboards, Alarms (0), AI Operations (New), GenAI Observability, Application Signals (APM) (New), and Infrastructure Monitoring. The main area is titled "Get started with CloudWatch" with a link to "View getting started page". It includes four cards:

- Create alarms**: Set alarms on any of your metrics to receive notifications when your metric crosses your specified threshold. [Create alarms](#)
- Create a default**: Create and name any CloudWatch dashboard **CloudWatch-Default** to display it here. [Create a default](#)
- View logs**: Monitor using your existing system, application and custom log files. [View logs](#)
- View events**: Write rules to indicate which events are of interest to your application and what automated action to take. [View events](#)

Select the EC2 instance by identifying the instance id and select the desired metric. In this case i have selected cpu utilization of my EC2 instance(my-web-server).



Next we will specify the metric condition. The metric name in our case will be CPU utilization select statistics as average.



Now we will define the conditions for firing our alarm to receive an email notification on our endpoint which is my email w.r.t high CPU utilization for our configured instance. so in our case it will be fired when the cpu utilization is ≥ 50 (this condition is just for hands-on practice and exploring the service).

The screenshot shows the 'Conditions' step of creating an alarm. It asks for a 'Threshold type': 'Static' (selected) and 'Anomaly detection'. For the 'Whenever CPUUtilization is...' condition, 'Greater/Equal >= threshold' is selected, with the value set to 50. Other options include 'Greater > threshold', 'Lower/Equal <= threshold', and 'Lower < threshold'. A note says 'Must be a number.' At the bottom, there's a 'Cancel' button and a yellow 'Next' button.

Once we are done with alarm creation we will now configure the actions for our alarm. So in the notification part in alarm we select an SNS topic that we have already created by the name ec2-cpu-alerts.

The screenshot shows the 'Configure actions' step of the CloudWatch Create alarm wizard. On the left, a sidebar lists steps: Step 1 (Specify metric and conditions), Step 2 (Configure actions - highlighted in blue), Step 3 (Add alarm details), and Step 4 (Preview and create). The main area is titled 'Configure actions' and contains a 'Notification' section. It defines the alarm state trigger as 'In alarm' (the metric or expression is outside the defined threshold). Below this, it says 'Send a notification to...' and shows a search bar with 'ec2-cpu-alerts' typed in. A list of topics is shown, with 'jinstthomas920@gmail.com - View in SNS Console' selected. Other options include 'Select an existing SNS topic', 'Create new topic', and 'Use topic ARN to notify other accounts'. At the bottom right of the notification section is a 'Remove' button.

Next we will provide the name and description of the alarm. I have provided High CPU Utilization and a short description that we can see in the below screenshot. After this we have successfully created the alarm.

The screenshot shows the 'Add alarm details' step of the CloudWatch Create alarm wizard. On the left, a sidebar lists steps: Step 1 (Specify metric and conditions), Step 2 (Configure actions), Step 3 (Add alarm details - highlighted in blue), and Step 4 (Preview and create). The main area is titled 'Add alarm details' and contains a 'Name and description' section. Under 'Alarm name', the text 'High CPU Utilization' is entered. Under 'Alarm description - optional', there is a preview tab showing the message 'you ~~insatce~~ has reached the threshold value of ~~maximum~~ CPU usage that is 50'. The preview text is enclosed in a box with an 'Edit' tab and a 'Preview' tab. At the bottom of the description area, it says 'Up to 1024 characters (74/1024)'. A blue banner at the top of the page says '(i) Alarm recommendations available Turn on Recommendations to pre-populate the wizard with the recommended alarms.'

The screenshot shows the AWS CloudWatch Alarms console. On the left, there's a sidebar with 'CloudWatch' selected. The main area has a green banner at the top stating 'Successfully created alarm High CPU Utilization.' Below it, there's a search bar and filters for 'Alarm state: Any', 'Alarm type: Any', and 'Actions status: Any'. A table lists the single alarm: 'High CPU Utilization' with 'Insufficient data' status, last updated on '2025-12-02 18:31:56'. The condition is 'CPUUtilization >= 50 for 1 datapoints within 5 minutes'.

3. Testing the alarm by creating stress

```
ubuntu@ip-172-31-29-208: $ sudo apt install -y stress
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  stress
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
```

Final output

The screenshot shows the AWS CloudWatch Metrics Insights interface. It displays two panels: 'Alarms by AWS service' and 'Recent alarms'. In the 'Recent alarms' panel, the 'High CPU Utilization' alarm is shown with a graph of CPU utilization over time. The 'Percent' axis ranges from 4.02 to 51.6. A red line indicates a sharp spike reaching nearly 50% utilization. Below the graph, the threshold condition is listed as 'CPUUtilization >= 50 for 1 datapoint within 5 minutes'. An email notification is open, addressed to 'Jins Thomas <no-reply@sns.amazonaws.com>', informing them that the alarm has entered the ALARM state due to a threshold cross. The email includes a link to view the alarm in the AWS Management Console.

Note

For faster results and testing purpose keep the threshold value less and the time period less.

