

Cloudwatch Alarm Events, and Logging

Part 1

Launching an Instance

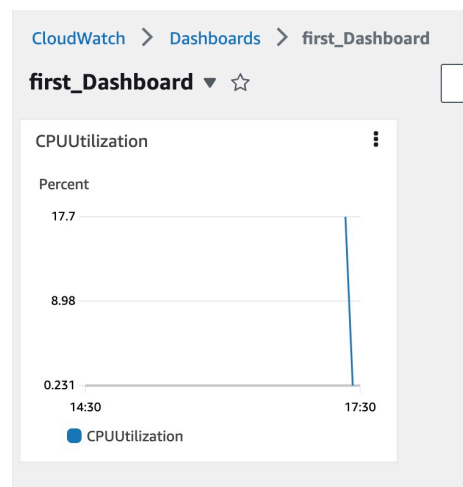
I created an EC2 instance named Cloudwatch-instance , AMI : Amazon Linux 2, t2.micro. I checked “enable CloudWatch detailed monitoring” under Advanced Details and pasted the code was given. I chose http and ssh open to anywhere under the security group section.

Instances (1) Info								Refresh	Connect	Instance state ▼	Actions ▼	Launch instances	▼
<input type="text" value="Find instance by attribute or tag (case-sensitive)"/>													
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone						
<input type="checkbox"/>	Cloucwath-instance	i-05813aa29e6a5f0c1	Running	t2.micro	-	No alarms +	us-east-1d						

Part 2

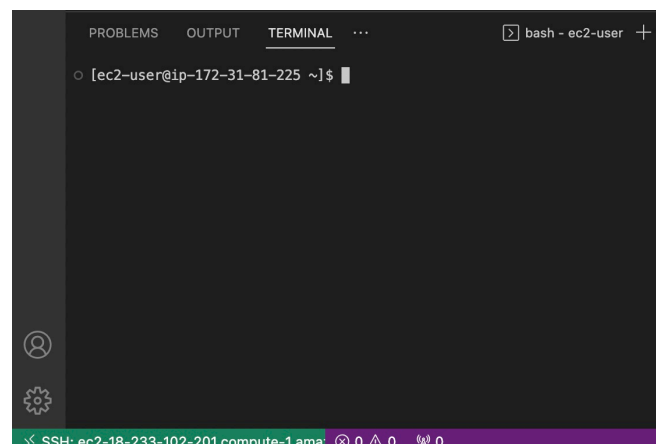
Creating a Cloudwatch dashboard

I selected Dashboard on the Cloudwatch and created one named ‘first_Dashboard’. Then I chose line, metrics, ec2 as a metrics. I selected ‘per-instance’ and CPUUtilization and then clicked ‘create widget’.



Upload Stress tool on EC2

I connected to my EC2 via ssh.



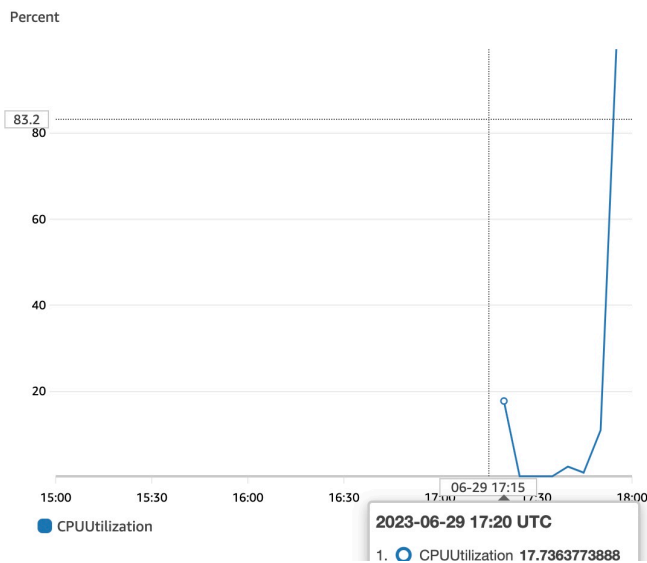
I uploaded the stress tool with the code 'sudo yum install -y stress'.

```
Installed:
  stress-1.0.4-28.amzn2023.0.2.x86_64

Complete!
[ec2-user@ip-172-31-81-225 ~]$
```

ec2-18-233-102-201.compute-1.ama: 0 0 0

I want to force the CPU. In order to force I used the code given. "stress --cpu 80 --timeout 20000"

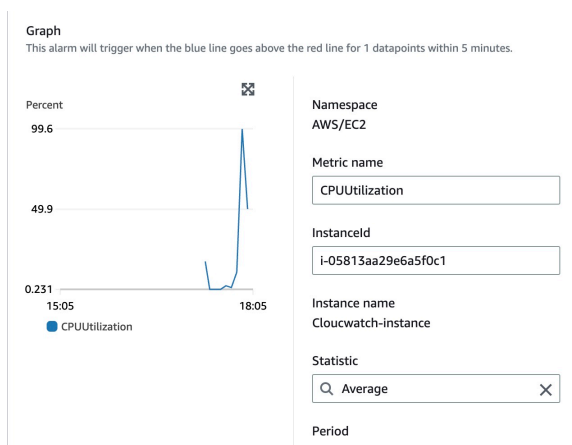


Part 3

Create an Alarm

I selected 'create alarm' and 'select metric'

I selected EC2, Per Instance Metrics, CPUUtilization and select metric.



I chose period for 1 minute and kept remaining as default. Threshold Type is static, whenever CPUUtilization is greater, than 60.

Alarm state trigger is 'In alarm', created a new topic as 'my-alarm', email endpoints I put my email address.

Under ec2 action; alarm state trigger ; in alarm ; select 'Stop this instance' and next. Alarm named as my first cloud watch alarm.

Conditions

Threshold type

Static

Whenever CPUUtilization is

Greater (>)

than...

60

► Additional configuration

Alarms (1)

☐ Hide Auto Scaling alarms

Clear selection

Create composite alarm

Actions ▾

Create alarm

Q Search

Any state ▾

Any type ▾

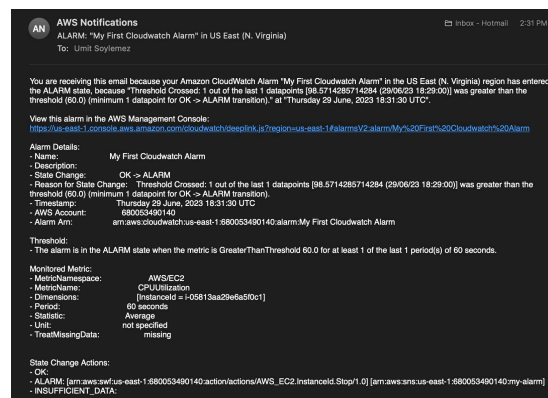
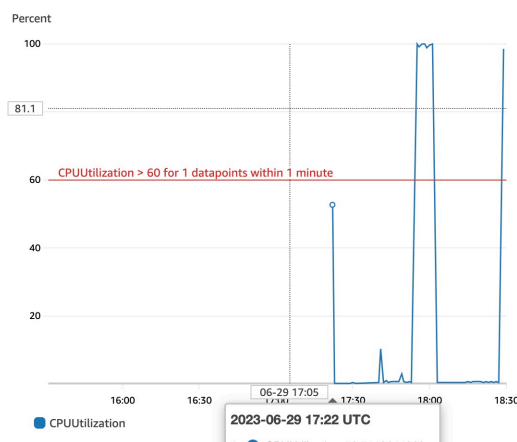
Any actions ... ▾

< 1 >

<input type="checkbox"/>	Name ▾	State ▾	Last state update ▾	Conditions	Actions ▾
<input type="checkbox"/>	My First Cloudwatch Alarm	Insufficient data	2023-06-29 18:27:25	CPUUtilization > 60 for 1 datapoints within 1 minute	Actions enabled

Now again the same code I run 'stress --cpu 80 --timeout 20000' on VS code.

I received an alarm message to my email and this message triggered to stop the ec2 instance.



Instances (1) Info

Connect

Instance state ▾

Actions ▾

Launch instance

Find instance by attribute or tag (case-sensitive)

< 1

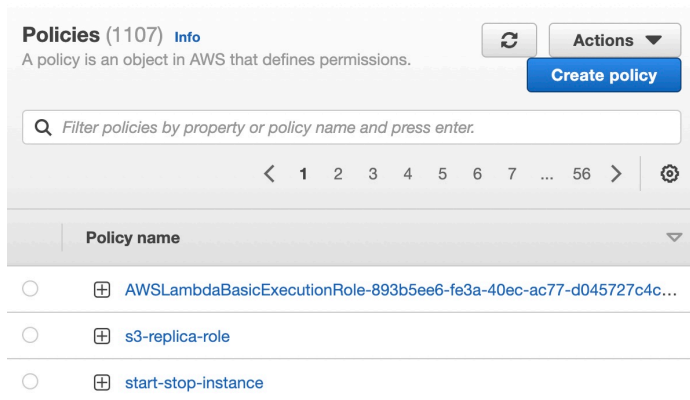
<div><input type="checkbox"/></div>	Name ▾	Instance ID	Instance state ▾	Instance type ▾	Status check	Alarm status	Availab
<div><input type="checkbox"/></div>	Clouwatch-instance	i-05813aa29e6a5f0c1	<div><div><div></div></div>Stopped<div><div></div><div></div></div></div>	t2.micro	-	<div><div></div>1/1 in al +</div>	us-east

Part 4

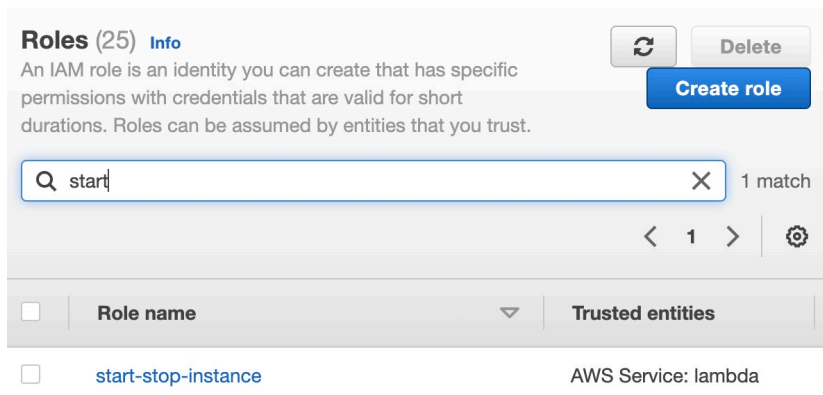
CloudWatch Events with Lambda

Create Role

I created a Policy named 'start-stop-instance' and pasted the json script.



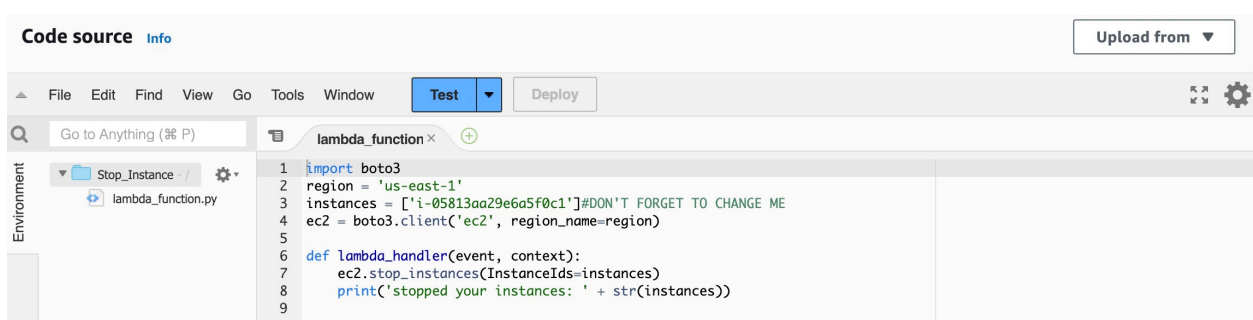
Then I created an IAM Role that will be used in Lambda Function. Role named start-stop-instance.



Creating Stop Lambda Functions

I create lambda function under functions. I selected author from scratch named Stop_Instance, runtime python 3.9.

I chose existing role 'start-stop-instance' under Change Default execution Role and then hit the create function. I pasted the function code given but I changed the instance ID in the Code and then clicked DEPLOY



Testing the function- Create test event

I selected 'Create new test event'. Event name is teststop, event template is 'hello-world'. Then I clicked test the instance had stopped.

The screenshot shows the AWS Lambda console interface. At the top, there's a 'Code source' section with an 'Info' link and an 'Upload from' button. Below this is a menu bar with 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', 'Window', and buttons for 'Test' and 'Deploy'. A search bar 'Go to Anything (% P)' is on the left. The main area is divided into 'Environment' and 'Execution results'. The 'Environment' pane shows a folder 'Stop_Instance' with a file 'lambda_function.py'. The 'Execution results' pane shows a successful test event named 'teststop' with a 'Response' of 'null'. It includes 'Function Logs' showing the start and end of the function execution, and a 'Request ID' of '618db2d3-c870-41b1-905e-c001c1161699'. Below the console, the 'Instances (1)' table is visible, showing one instance named 'Cloucdwatch-instance' with ID 'i-05813aa29e6a5f0c1', state 'Stopped', type 't2.micro', and availability zone 'us-east-1d'.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
Cloucdwatch-instance	i-05813aa29e6a5f0c1	Stopped	t2.micro	-	1 alarms	us-east-1d

Creating Start Lambda Functions

I create lambda function under functions. I selected author from scratch named Start_Instance, runtime python 3.9.

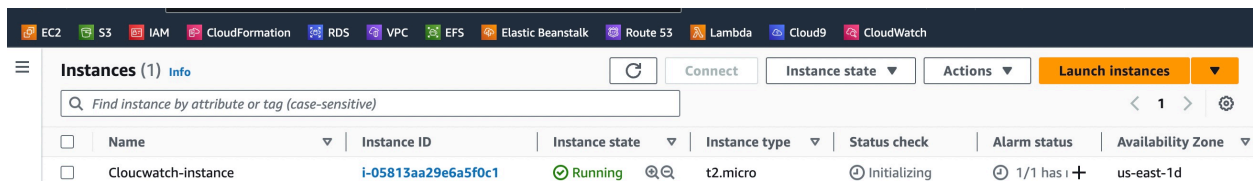
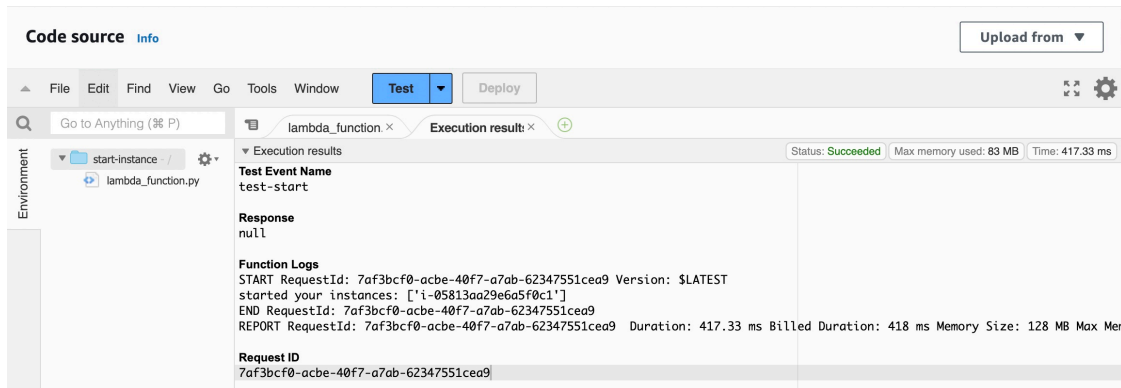
I chose existing role 'start-stop-instance' under Change Default execution Role and then hit the create function. I pasted the function code given but I changed the instance ID in the Code and then clicked DEPLOY

The screenshot shows the AWS Lambda console interface for a function named 'start-instance'. The 'Code source' section at the top has an 'Info' link and an 'Upload from' button. The menu bar includes 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', 'Window', and 'Test' and 'Deploy' buttons. A search bar 'Go to Anything (% P)' is on the left. The 'Environment' pane shows a folder 'start-instance' with a file 'lambda_function.py'. The main area displays the code for the lambda function, which imports boto3, sets the region to 'us-east-1', and defines a lambda_handler function that starts instances and prints the instance IDs.

```
1 import boto3
2 region = 'us-east-1'
3 instances = ['i-05813aa29e6a5f0c1'] # Burayi kendi instance'ina gore optimize et
4 ec2 = boto3.client('ec2', region_name=region)
5
6 def lambda_handler(event, context):
7     ec2.start_instances(InstanceIds=instances)
8     print('started your instances: ' + str(instances))
```

Testing the function- Create test event

I selected 'Create new test event'. Event name is teststart, event template is 'hello-worl. Than I clicked test the instance had started.

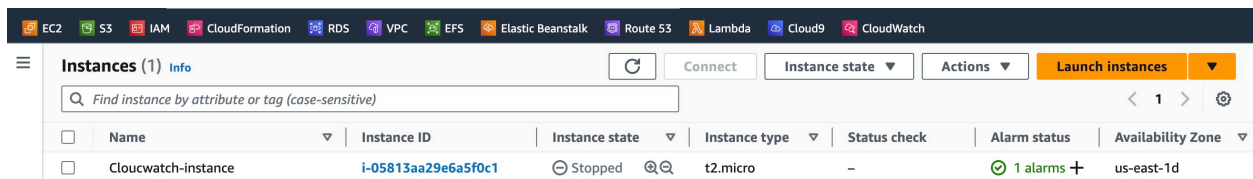


Creating Stop-Cloudwatch Event

I went to the CloudWatch Console and under the Event sub-section Rules and Create Rules.

I selected 'Schedule' and 1 mins for cron expression under event source. Targets; Function: Stop_Instance.

I named Event_Stop under the configure details. Description: This event provides stop action state I chose enabled.

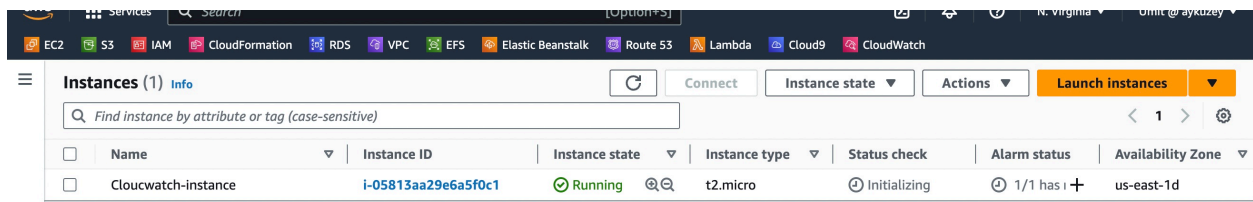


Creating Terminate- Cloudwatch Event

I went to the CloudWatch Console and under the Event sub-section Rules and Create Rules.

I selected 'Schedule' and 1 mins for cron expression under event source. Targets; Function: Start_Terminate.

I named Event_Terminate under the configure details. Description: This event provides start action state I chose enabled.



Instances (1) Info						
<input type="text" value="Find instance by attribute or tag (case-sensitive)"/>						
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	Cloucwatch-instance	i-05813aa29e6a5f0c1	Running	t2.micro	Initializing	1/1 has 1