

Lab 16

INTRODUCTION TO CLOUDWATCH

STEP 1: Log In to the Amazon Web Service Console

This laboratory experience is about Amazon Web Services and you will use the AWS Management Console in order to complete all the lab steps.

The screenshot shows the AWS Management Console interface. At the top, there's a header bar with the AWS logo, a 'Services' dropdown menu, and user information: 'Antonio Ang', 'Oregon', and a 'Support' link. Below the header, the main content area is titled 'Amazon Web Services' and is divided into several columns of service categories. The 'Compute' column includes EC2 (Virtual Servers in the Cloud) and Lambda (Run Code in Response to Events). The 'Storage & Content Delivery' column includes S3 (Scalable Storage in the Cloud), Storage Gateway (Integrates On-Premises IT Environments with Cloud Storage), Glacier (Archive Storage in the Cloud), and CloudFront (Global Content Delivery Network). The 'Database' column includes RDS (MySQL, Postgres, Oracle, SQL Server, and Amazon Aurora), DynamoDB (Predictable and Scalable NoSQL Data Stores), ElastiCache (In-Memory Cache), and Redshift (Managed Petabyte-Scale Data Warehouse Service). The 'Networking' column includes VPC (Isolated Cloud Resources), Direct Connect (Dedicated Network Connection to AWS), and Route 53 (Scalable DNS and Domain Name Registration). The 'Administration & Security' column includes Directory Service (Managed Directories in the Cloud), Identity & Access Management (Access Control and Key Management), Trusted Advisor (AWS Cloud Optimization Expert), CloudTrail (User Activity and Change Tracking), Config (Resource Configurations and Inventory), and CloudWatch (Resource and Application Monitoring). The 'Deployment & Management' column includes Elastic Beanstalk (AWS Application Container), OpsWorks (DevOps Application Management Service), CloudFormation (Templated AWS Resource Creation), and CodeDeploy (Automated Deployments). The 'Analytics' column includes EMR (Managed Hadoop Framework), Kinesis (Real-time Processing of Streaming Big Data), and Data Pipeline (Orchestration for Data-Driven Workflows). The 'Application Services' column includes SQS (Message Queue Service), SWF (Workflow Service for Coordinating Application Components), AppStream (Low Latency Application Streaming), Elastic Transcoder (Easy-to-use Scalable Media Transcoding), SES (Email Sending Service), and CloudSearch (Managed Search Service). The 'Mobile Services' column includes Cognito (User Identity and App Data Synchronization), Mobile Analytics (Understand App Usage Data at Scale), and SNS (Push Notification Service). The 'Enterprise Applications' column includes WorkSpaces (Desktops in the Cloud) and Zocalo (Secure Enterprise Storage and Sharing Service). To the right of the service categories, there are 'Additional Resources' including 'Getting Started' (See our documentation to get started and learn more about how to use our services.), 'AWS Console Mobile App' (View your resources on the go with our AWS Console mobile app, available from Amazon Appstore, Google Play, or iTunes.), 'AWS Marketplace' (Find and buy software, launch with 1-Click and pay by the hour.), 'Service Health' (All services operating normally. Updated: Nov 20 2014 12:57:00 GMT-0800. Service Health Dashboard), and 'Set Start Page' (Console Home).

The AWS Management Console is a web control panel for managing all your AWS resources, from EC2 instances to SNS topics. The console enables cloud management for all aspects of the AWS account, including managing security credentials, or even setting up new IAM Users.

Log in to the AWS Management Console

In order to start the laboratory experience, open the Amazon Console by clicking this button:

[Open AWS Console](#)

Log in with the username **xxxxx** and the password **xxxxx**



Account:

User Name:

Password:

☐ I have an MFA Token ([more info](#))

Sign In

[Sign-in using root account credentials](#)

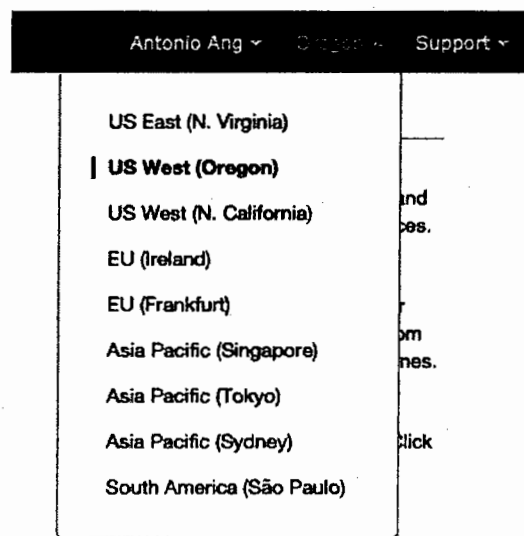
[Terms of Use](#) [Privacy Policy](#)
© 1996-2014, Amazon Web Services, Inc. or its affiliates.

Select the right AWS Region

Amazon Web Services is available in different regions all over the world, and the console lets you provision resources across multiple regions. You usually choose a region that best suits your business needs to optimize your customer's experience, but you must use the region **US**

West (Oregon) for this laboratory.

You can select the **US West (Oregon)** region using the upper right dropdown menu on the AWS Console page.



STEP 2: Explore CloudWatch

Key Concepts

There are a few concepts to define before working with CloudWatch. AWS has done a great job defining these concepts in the documentation, so let's stick with the official version:

Metrics

A metric is the fundamental concept in CloudWatch and represents a time-ordered set of data points. These data points can be either your custom metrics or metrics from other services in AWS. You or AWS products publish metric data points into CloudWatch and you retrieve statistics about those data points as an ordered set of time-series data. Metrics exist only in the region in which they are created.

Think of a metric as a variable to monitor, and the data points represent the values of that variable over time. For example, the CPU usage of a particular Amazon EC2 instance is one metric, and the latency of an Elastic Load Balancing load balancer is another.

CloudWatch stores your metric data for two weeks. You can publish metric data from multiple sources, such as incoming network traffic from dozens of different Amazon EC2 instances, or requested page views from several different web applications. You can request statistics on metric data points that occur within a specified time window.

Namespaces

CloudWatch namespaces are containers for metrics. Metrics in different namespaces are isolated from each other, so that metrics from different applications are not mistakenly aggregated into the same statistics.

Namespace names are strings you define when you create a metric. The names must be valid XML characters, typically containing the alphanumeric characters "0-9A-Za-z" plus "." (period), "-" (hyphen), "_" (underscore), "/" (slash), "#" (hash), and ":" (colon). AWS namespaces all follow the convention `AWS/<service>`, such as `AWS/EC2` and `AWS/ELB`.

There is no default namespace. You must specify a namespace for each data element you put into CloudWatch.

Dimensions

A dimension is a name/value pair that helps you to uniquely identify a metric. Every metric has specific characteristics that describe it, and you can think of dimensions as categories for those characteristics. Dimensions help you design a structure for your statistics plan. Because dimensions are part of the unique identifier for a metric, whenever you add a unique name/value pair to one of your metrics, you are creating a new metric.

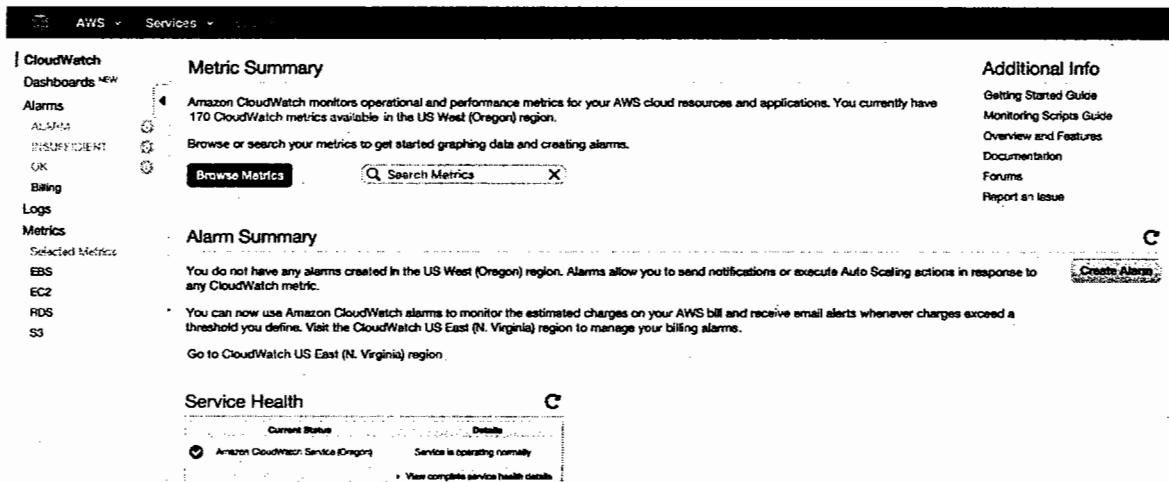
You can take a look at the full version of the documentation [in here](#).

Exploring CloudWatch

Go on the CloudWatch console and make sure you are in the **US West (Oregon)** region.

Management Tools

-  **CloudWatch**
Monitor Resources and Applications
-  **CloudFormation**
Create and Manage Resources with Templates
-  **CloudTrail**
Track User Activity and API Usage
-  **Config**
Track Resource Inventory and Changes
-  **OpsWorks**
Automate Operations with Chef
-  **Service Catalog**
Create and Use Standardized Products
-  **Trusted Advisor**
Optimize Performance and Security



The screenshot shows the AWS CloudWatch console interface. On the left is a navigation menu with options: CloudWatch, Dashboards, Alarms, Metrics, Billing, Logs, and Metrics. The main content area is divided into three sections: Metric Summary, Alarm Summary, and Service Health. The Metric Summary section provides an overview of available metrics and includes a search bar. The Alarm Summary section informs the user about the current state of alarms in the region. The Service Health section displays the current status of the Amazon CloudWatch service.

Metric Summary

Amazon CloudWatch monitors operational and performance metrics for your AWS cloud resources and applications. You currently have 170 CloudWatch metrics available in the US West (Oregon) region.

Browse or search your metrics to get started graphing data and creating alarms.

[Browse Metrics](#)

Alarm Summary

You do not have any alarms created in the US West (Oregon) region. Alarms allow you to send notifications or execute Auto Scaling actions in response to any CloudWatch metric.

You can now use Amazon CloudWatch alarms to monitor the estimated charges on your AWS bill and receive email alerts whenever charges exceed a threshold you define. Visit the CloudWatch US East (N. Virginia) region to manage your billing alarms.

Go to CloudWatch US East (N. Virginia) region.

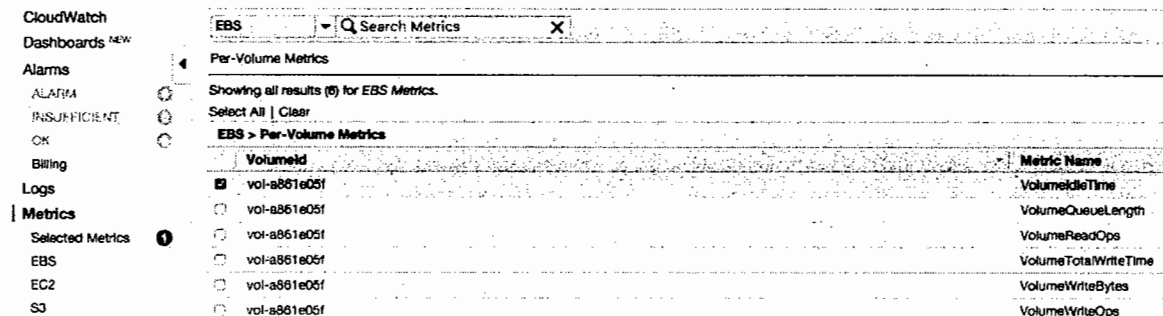
Service Health

Current Status: Amazon CloudWatch Service (Oregon) Service is operating normally. [View complete service health details](#)

Additional Info

- Getting Started Guide
- Monitoring Scripts Guide
- Overview and Features
- Documentation
- Forums
- Report an Issue

Take time to see what Metrics and Namespaces look like in the CloudWatch console. Simply select a namespace and then a particular metric:



This screenshot shows the CloudWatch console with the 'EBS' namespace selected. It displays a list of metrics for EBS volumes. The table includes columns for 'VolumeId' and 'Metric Name'. The first metric, 'VolumeIdleTime', is selected.

EBS

Per-Volume Metrics

Showing all results (6) for EBS Metrics.

Select All | Clear

EBS > Per-Volume Metrics

VolumeId	Metric Name
<input checked="" type="radio"/> vol-a861e05f	VolumeIdleTime
<input type="radio"/> vol-a861e05f	VolumeQueueLength
<input type="radio"/> vol-a861e05f	VolumeReadOps
<input type="radio"/> vol-a861e05f	VolumeTotalWriteTime
<input type="radio"/> vol-a861e05f	VolumeWriteBytes
<input type="radio"/> vol-a861e05f	VolumeWriteOps





STEP 3: Monitoring EC2 Instances

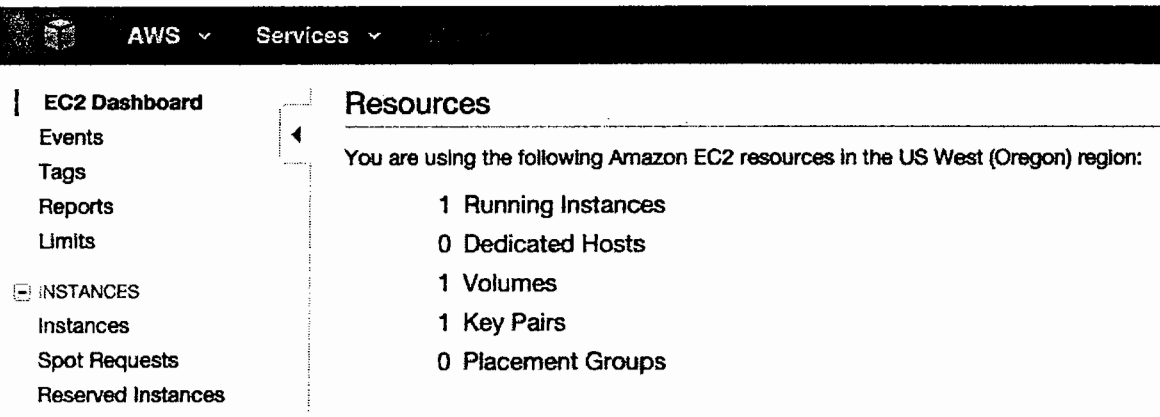
Standard Metrics

CloudWatch doesn't monitor everything out of the box, for EC2 instances, for example, it doesn't monitor things like disk space or memory usage, let's take a look at the metrics that CloudWatch monitors by default for EC2 instances, but this time in the EC2 console:

Amazon Web Services

Compute

-  **EC2**
Virtual Servers in the Cloud
-  **EC2 Container Service**
Run and Manage Docker Containers
-  **Elastic Beanstalk**
Run and Manage Web Apps
-  **Lambda**
Run Code in Response to Events



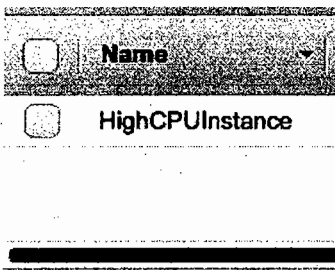
Resources

You are using the following Amazon EC2 resources in the US West (Oregon) region:

- 1 Running Instances
- 0 Dedicated Hosts
- 1 Volumes
- 1 Key Pairs
- 0 Placement Groups

Click on instances to see more details:

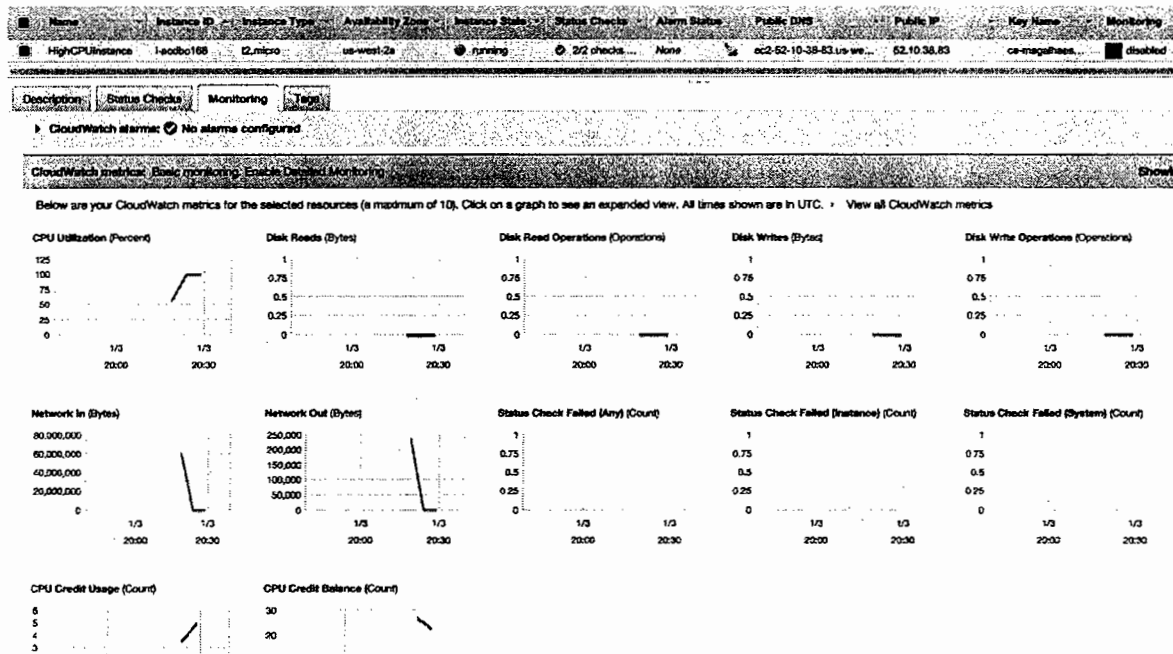
- Reports
- Limits
- INSTANCES
- Instances**
- Spot Requests
- Reserved Instances
- Commands
- Dedicated Hosts



<input type="checkbox"/>	Name
<input type="checkbox"/>	HighCPUInstance

Select an instance above

There is already an instance called **HighCPUInstance** running, select this instance and then click in the **Monitoring** tab and take a look at the standard metrics:



These are the standard metrics that CloudWatch monitors for all your EC2 instances, please refer to the [documentation](#) for more details.

You should be aware that all the metrics in this tab related to Disk (Disk Reads, Disk Read Operations, Disk Writes, Disk Write Operations) pertain to ephemeral storage disks. Those metrics will not represent anything if you have launched an EBS backed instance.

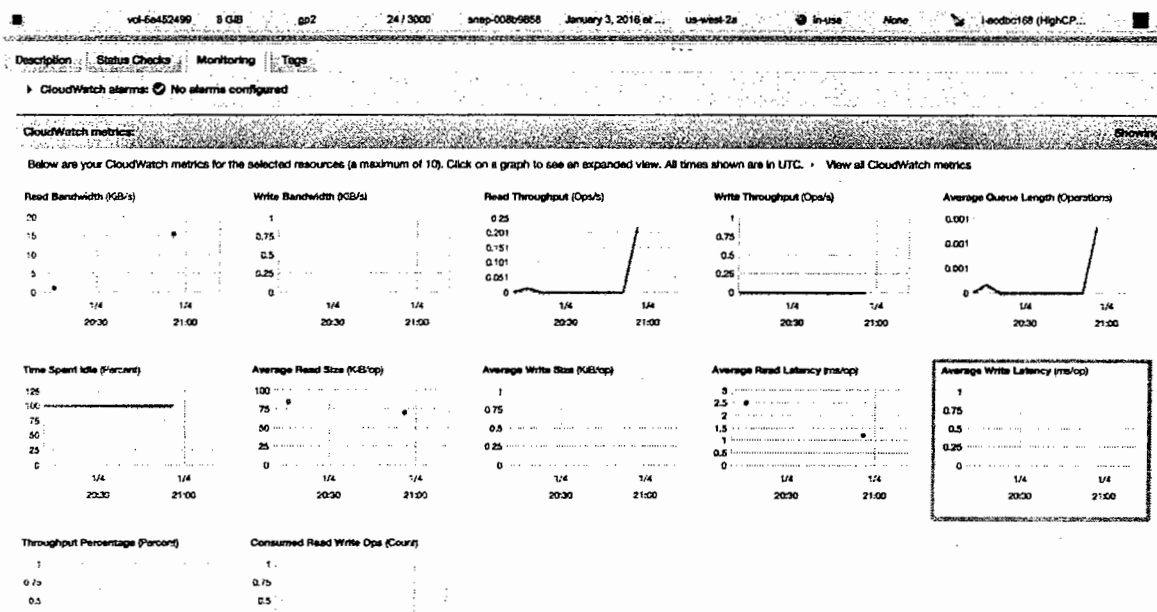
To see the metrics related to EBS volumes you need to go somewhere else. Let's take a look at the metrics of the EBS volume from this particular instance. You can do it by selecting the instance, clicking on the **Description** tab, click on the **Root** device then on the EBS ID:

EBS-optimized False
Root device type ebs
Root device /dev/xvda
Block devices

Block Device /dev/xvda

EBS ID `vol-46a12171`
Root device type EBS
Attachment time 2016-01-03T20:22:17.000Z
Block device status attached
Delete on termination True

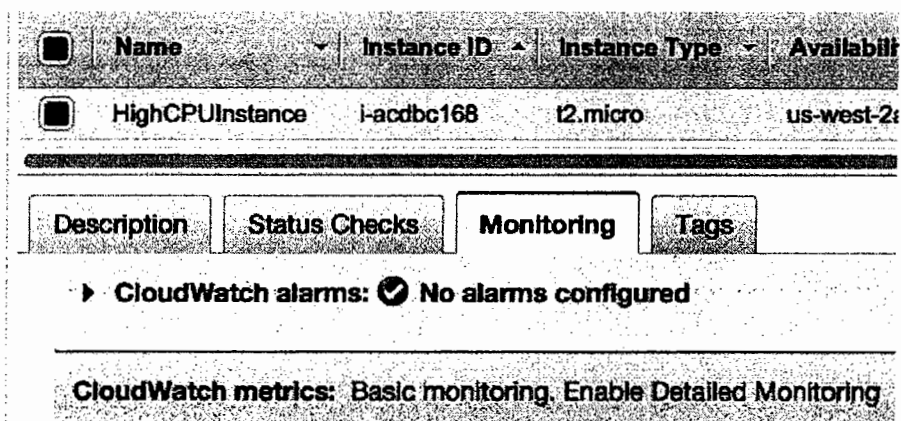
Now with the volume selected, click on the **Monitoring** tab to see the metrics for this EBS volume:



You can learn even more about these metrics by taking a look at the [documentation](#).

Enabling / Disabling Detailed Monitoring

There is also another important thing to consider, CloudWatch typically monitors your EC2 instances every 5 minutes. If you need more frequent monitoring, you can configure CloudWatch to monitor your instances every minute. To make this happen, you need to enable something called **Detailed Monitoring**. You can do this during the instance launch or change it anytime. To enable or disable detailed monitoring after an instance is launched select the instance, click on the **Monitoring** tab, and then on **Enable** (or **Disable**) **Detailed Monitoring**:



For better experience during this lab you should **Enable Detailed Monitoring** for the **HighCPUInstance**.

If you are done exploring and understand what the standard metrics mean, then you should proceed to the next step.

STEP 4: Install the EC2 Monitoring Scripts

In the last step we learned that CloudWatch only monitors a few EC2 metrics per default. That may be enough for many cases, but sometimes you will want more complete monitoring. In this step let's enhance the power of CloudWatch by installing some monitoring scripts in an EC2 instance. These scripts will run every 5 minutes and send custom metrics to CloudWatch.

Install the EC2 Monitoring Scripts

Let's launch a new instance and configure the monitoring scripts during the instance's launch by employing User Data.

User Data is a script that you can configure to run once, during the instance launch. This is very useful for bootstrapping and with that we can customize our instances without accessing it via SSH or RDP for Windows instances.

Let's launch our instance and see how it works. Go to the EC2 console:

Amazon Web Services

Compute



EC2

Virtual Servers in the Cloud

On the Dashboard click on **Launch instance**:

AWS Services

EC2 Dashboard

- Events
- Tags
- Reports
- Limits
- INSTANCES
 - Instances
 - Spot Requests
 - Reserved Instances
 - Commands
 - Dedicated Hosts
- IMAGES
 - AMIs
 - Bundle Tasks

Resources

You are using the following Amazon EC2 resources in the US West (Oregon) region:

1 Running Instances	0 Elastic IPs
0 Dedicated Hosts	0 Snapshots
1 Volumes	0 Load Balancers
1 Key Pairs	21 Security Groups
0 Placement Groups	

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 Instance.

Launch Instance

On step 1, select Amazon linux as our AMI:

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 1: Choose an Amazon Machine Image (AMI)

Cancel and Exit

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.


Quick Start |< < 1 to 22 of 22 AMIs > >|

My AMIs

AWS Marketplace

Community AMIs

☐ Free tier only ⓘ


Amazon Linux
Free tier eligible

Amazon Linux AMI 2015.09.1 (HVM), SSD Volume Type - ami-f0091d91
Select
64-bit

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.

Root device type: ebs Virtualization type: hvm

On step 2, select t2.micro as our instance size:

1. Choose AMI **2. Choose Instance Type** 3. Configure Instance 4. Add Storage 5. Tag Instance

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. In storage, and networking capacity, and give you the flexibility to choose the appropriate mix of needs.

Filter by: **All instance types** **Current generation** **Show/Hide Columns**

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB me

	Family	Type	vCPUs ⓘ	Mem
<input type="checkbox"/>	General purpose	t2.nano	1	
<input checked="" type="checkbox"/>	General purpose	t2.micro Free tier eligible	1	

Now, on step 3, is where the magic happens. First of all, you need to expand the advanced area of this step by clicking on the little arrow, just to the left of where you see **Advanced Details**:

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of 1 to the instance, and more.

Number of instances	1	Launch into Auto Scaling Group
Purchasing option	<input type="checkbox"/> Request Spot Instances	
Network	vpc-cf7555aa (172.31.0.0/16) (default)	Create new VPC
Subnet	No preference (default subnet in any Availability Zone)	Create new subnet
Auto-assign Public IP	Use subnet setting (Enable)	
IAM role	None	Create new IAM role
Shutdown behavior	Stop	
Enable termination protection	<input type="checkbox"/> Protect against accidental termination	
Monitoring	<input type="checkbox"/> Enable CloudWatch detailed monitoring Additional charges apply.	
Tenancy	Shared - Run a shared hardware instance Additional charges will apply for dedicated tenancy.	

▼ Advanced Details

User data	<input checked="" type="radio"/> As text <input type="radio"/> As file <input type="checkbox"/> Input is already base64 encoded
(Optional)	

Now we will insert the code that we are going to execute during the instance launch, however in order to send metrics to CloudWatch, we need to configure some credentials first, you can use either Access Keys or IAM roles for this task, in this lab we will follow the best practices and use IAM roles. There is an instance role already created in your account, so click on IAM role and select it. It will have a name that looks like **cwinitconf-EC2MonitoringRole-XXXXXXXXXX**:

Auto-assign Public IP	Use subnet setting (Enable)
IAM role	<div>None aws-opsworks-ec2-role ✓ CF-EC2MonitoringRole-1KVQ5IWVBQQSW EMR_EC2_DefaultRole</div> Create new IAM role

Then copy and paste this code in the User Data text field:

```
#!/bin/bash
yum install -y perl-DateTime perl-Sys-Syslog perl-LWP-Protocol-https
wget http://aws-cloudwatch.s3.amazonaws.com/downloads/CloudWatchMonitoringScripts-1.2.1.zip
unzip CloudWatchMonitoringScripts-1.2.1.zip
```

```
rm CloudWatchMonitoringScripts-1.2.1.zip
echo "*/*5 * * * * /aws-scripts-mon/mon-put-instance-data.pl --mem-util --disk-space-
util --disk-path=/ --from-cron" > monitoring.txt
crontab monitoring.txt
rm monitoring.txt
```

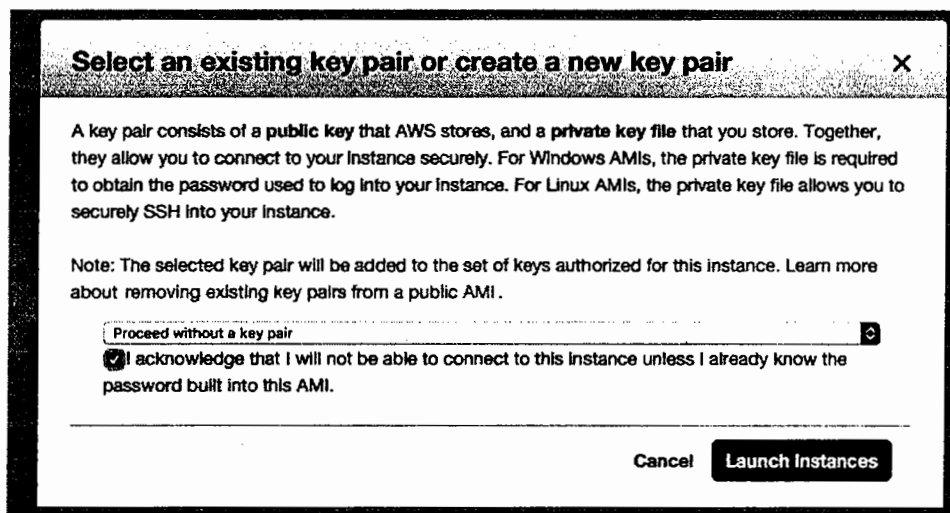
▼ Advanced Details

User data ⓘ

☒ As text ☐ As file ☐ Input is already base64 encoded

```
#!/bin/bash
yum install -y perl-DateTime perl-Sys-Syslog perl-LWP-Protocol-https
wget http://aws-
cloudwatch.s3.amazonaws.com/downloads/CloudWatchMonitoringScripts-
1.2.1.zip
```

Now we are ready to proceed. Select **Review and Launch**, then press **Launch**. A dialog box will appear asking for a Key Pair. Select **Proceed without a key pair**, mark the "I acknowledge that I will not be able to connect to this instance unless I already know the password built into this AMI." checkbox and finally click on **Launch Instances**.



Now we can give a name to our instance, it's not mandatory, but it will help us in the next step:

<input type="checkbox"/>	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
<input type="checkbox"/>	HighCPUInstance	i-acdbc168	t2.micro	us-west-2a	● running	✔ 2/2 checks ...
<input checked="" type="checkbox"/>	Monitoring Scripts	i-fa52b43d	t2.micro	us-west-2a	● running	✔ 2/2 checks ...

Wait until the Instance State change to "running", then select the instance and go to the monitoring tab. Notice that nothing is different in there. This happens because we are not adding anything special to CloudWatch. We are just configuring our instance to send custom

metrics each five minutes to the CloudWatch service, using the credentials associated with the IAM role in the EC2 instance.

To check the results, go to the CloudWatch console. Notice that now we have a new Namespace in here, it is called **Linux System**. This name is configured when you send the custom metrics and this particular name is the standard for these monitoring scripts. If you don't see the new Namespace wait a few minutes and refresh the page because CloudWatch takes some time to display the information in the dashboard.

Once the new Namespace appears you should click on it.

CloudWatch

Dashboards NEW

Alarms

ALARM

INSUFFICIENT

OK

Billing

Logs

Metrics

Selected Metrics

EBS

EC2

S3

Linux System

Metric Summary

Amazon CloudWatch monitoring metrics available

Browse or search your metrics

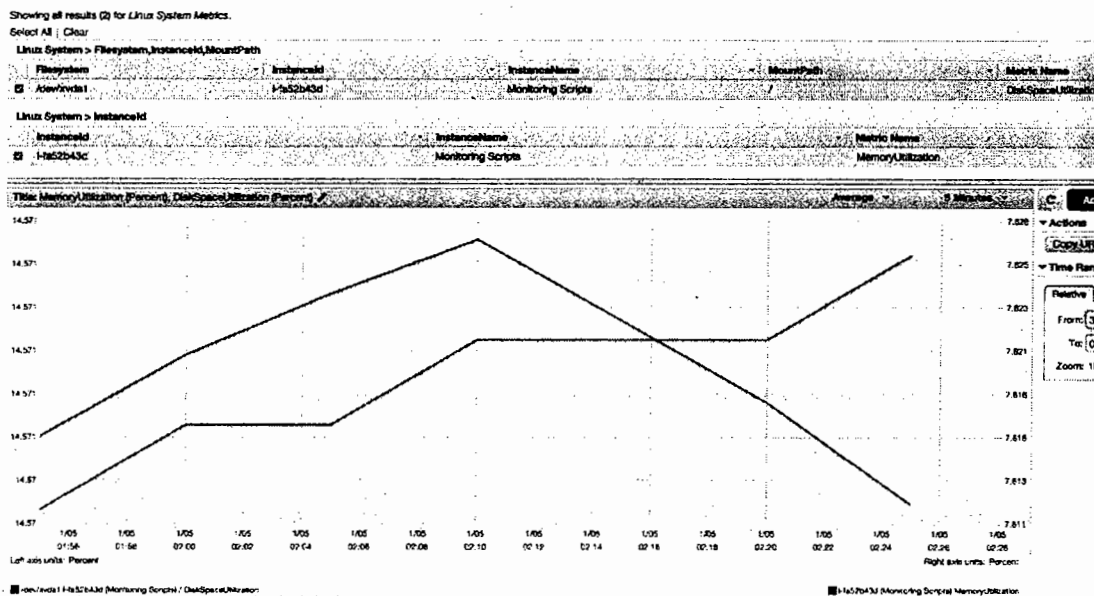
Browse Metrics

Alarm Summary

You do not have any alarm on any CloudWatch metric.

You can now use Amazon CloudWatch thresholds you define. Visit the documentation for more information.

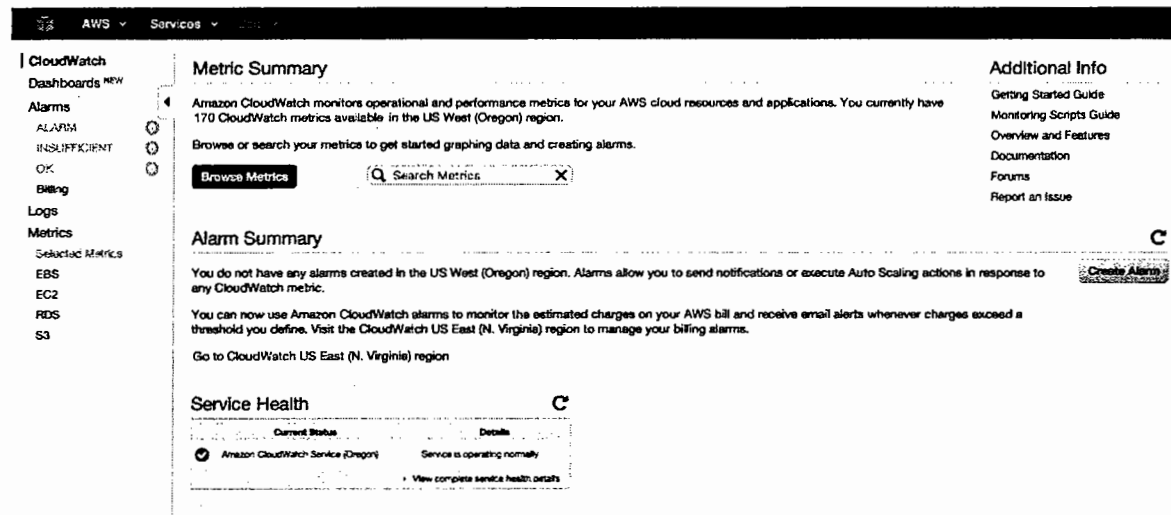
Now we have two new metrics being monitored by CloudWatch, **MemoryUtilization** and **DiskSpaceUtilization**. Select one of them, or both, and see what appears in the graph.



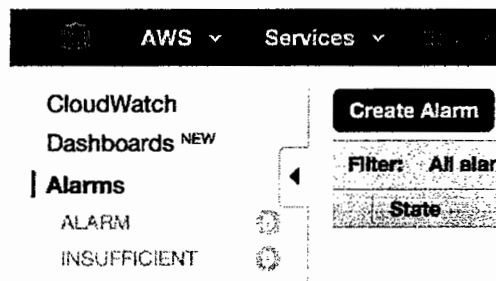
We will continue exploring the Graph tool in a further step.

STEP 5: Create your first CloudWatch alarm

Go to the CloudWatch console and click on **Alarms**:



In the Alarms page click **Create Alarm**:



A pop-up will appear on the screen. Now you need to select a particular metric to create the alarm. We will create an alarm for the CPUUtilization metric of the HighCPUInstance, so we'll need to select this particular metric first. To do this, click on **Per-Instance Metrics** right below **EC2 Metrics**:

Create Alarm

1. Select Metric

2. Define Alarm

Browse Metrics

Q Search Metrics

X

CloudWatch Metrics by Category

Your CloudWatch metric summary has loaded. Total metrics: 44

EBS Metrics: 16

EC2 Metrics: 24

S3 Metrics: 2

Per-Volume Metrics: 16

Per-Instance Metrics: 24

Storage Metrics: 2

Linux System Metrics: 2

Filesystem, InstanceId, MountPath: 1

InstanceId: 1

Then look for the **CPUUtilization** metric related to the **HighCPUInstance**:

Create Alarm

1. Select Metric

2. Define Alarm

EC2

Q

Search Metrics

X

1 to 24 of 24 Metrics

<>

InstanceId	InstanceName	Metric Name
<input type="checkbox"/> i-4b86678c	Monitoring Scripts	CPUCreditBalance
<input type="checkbox"/> i-4b86678c	Monitoring Scripts	CPUCreditUsage
<input type="checkbox"/> i-4b86678c	Monitoring Scripts	CPUUtilization
<input type="checkbox"/> i-4b86678c	Monitoring Scripts	DiskReadBytes
<input type="checkbox"/> i-4b86678c	Monitoring Scripts	DiskReadOps
<input type="checkbox"/> i-4b86678c	Monitoring Scripts	DiskWriteBytes
<input type="checkbox"/> i-4b86678c	Monitoring Scripts	DiskWriteOps
<input type="checkbox"/> i-4b86678c	Monitoring Scripts	NetworkIn
<input type="checkbox"/> i-4b86678c	Monitoring Scripts	NetworkOut
<input type="checkbox"/> i-4b86678c	Monitoring Scripts	StatusCheckFailed
<input type="checkbox"/> i-4b86678c	Monitoring Scripts	StatusCheckFailed_Instance
<input type="checkbox"/> i-4b86678c	Monitoring Scripts	StatusCheckFailed_System
<input type="checkbox"/> i-7a8465bd	HighCPUInstance	CPUCreditBalance
<input type="checkbox"/> i-7a8465bd	HighCPUInstance	CPUCreditUsage
<input checked="" type="checkbox"/> i-7a8465bd	HighCPUInstance	CPUUtilization
<input type="checkbox"/> i-7a8465bd	HighCPUInstance	DiskReadBytes
<input type="checkbox"/> i-7a8465bd	HighCPUInstance	DiskReadOps

Title: CPUUtilization (Percent)

Average

5 Minutes

Update Graph

Time Range

Relative

Absolute

UTC (GMT)

From: 12

hours ago

To: 0

minutes ago

Zoom: 1h | 3h | 6h | 12h | 1d | 3d | 1w | 2w

Left axis units: Percent

i-7a8465bd (HighCPUInstance)

Cancel

Previous

Next

Create Alarm

And now we are ready to click **Next**.

An Alarm is basically a metric that is being monitored over time. Once the metrics meet a customizable rule it will change the alarm's state. There are 3 possible states for an alarm:

OK—The metric is within the defined threshold

ALARM—The metric is outside of the defined threshold

INSUFFICIENT_DATA—The alarm has just started, the metric is not available, or not enough data is available for the metric to determine the alarm state

We should also set a period of time when the metric will be evaluated. And we can set one or multiple Actions for an alarm. However, it is not mandatory. After filling in all the information needed, we should have something like this:

Create Alarm

1. Select Metric

2. Define Alarm

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

Name:

Description:

Whenever: CPUUtilization

Is:

for: consecutive period(s)

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line for a duration of 5 minutes

Namespace: AWS/EC2

InstanceId:

InstanceName: HighCPUInstance

Metric Name: CPUUtilization

Period:

Statistic:

Actions

Define what actions are taken when your alarm changes state.

Notification

Delete

Whenever this alarm:

Send notification to: Select list

Email list:

+ Notification

+ AutoScaling Action

+ EC2 Action

Cancel

Previous

Next

Create Alarm

Notice that in the **Actions** area, you must click **New list** in order to change the form. After you must provide a new SNS topic name and also your personal email. The below image is how the screen should appear prior to clicking on **New list**:

Actions

Define what actions are taken when your alarm changes state.

Notification
Delete

Whenever this alarm: State is ALARM

Send notification to: Select a notification list
New list Enter list ⓘ

+ Notification + AutoScaling Action + EC2 Action

Now we're ready to create the Alarm. Click **Create Alarm**.

You have just created a new alarm, along with a new SNS topic -- if you have also inserted your email into the Email list. Go to your email account because you should have received an email from AWS. This email is a subscription confirmation. Just click the provided link and you're ready to receive emails from this topic. Every time the state of the alarm switches to **ALARM**, you'll receive a new email.

Now that the alarm is created, you can check it out, don't worry about the state of the alarm right now, we will come back to it soon.

CloudWatch
Dashboards NEW
Alarms
ALARM
INSUFFICIENT
OK
Billing
Logs
Metrics

✓ Your alarm High CPU Alarm has been saved.

Create Alarm Modify Copy Delete

Filter: All alarms Search Alarms X

State	Name
<input type="checkbox"/> ALARM	High CPU Alarm

By selecting the alarm you can see some useful information about the alarm itself. In the **Details** tab you have a general overview about the alarm, and in the **History** tab you can see the last 50 states of the alarm:

State: ALARM Name: High CPU Alarm Threshold: CPUUtilization >= 50 for 5 minutes Config Status: No notifications

1 Alarm selected

Alarm: High CPU Alarm

Details History

State Details: State changed to ALARM at 2016/01/05. Reason: Threshold Crossed: 1 datapoint (36.94%) was greater than or equal to the threshold (50.0).

Description: When CPU Utilization > 50%

Threshold: CPUUtilization >= 50 for 5 minutes

Actions: In ALARM: Send message to topic "AlarmTopic" Send message to topic "AlarmTopic" (Warning: the alarm is not configured to notify. Please modify this alarm and add an email address.)

Namespace: AWS/EC2

Metric Name: CPUUtilization

Dimensions: InstanceId: i-7a5485cd (HighCPUInstance)

Statistic: Average

Period: 5 minutes

High CPU Alarm
CPUUtilization >= 50

ALARM	High CPU Alarm	CPUUtilization >= 50 for 5 minutes	No notifications
-------	----------------	------------------------------------	------------------

1 Alarm selected		
Alarm: High CPU Alarm		
Details	History	

Showing all history entries (3)

Date	Type	Description
2016-01-05 02:48 UTC-2	Action	Successfully executed action arn:aws:iam::us-west-2:568965021270:AlarmTopic
2016-01-05 02:48 UTC-2	State update	Alarm updated from INSUFFICIENT_DATA to ALARM
2016-01-05 02:48 UTC-2	Configuration update	Alarm "High CPU Alarm" created





Take some time exploring the latest alarm changes and try to understand what is going on in each change. You can see more details of the changes by clicking the little arrow next to the date/time of the changes.

STEP 6: Create an Alarm using the EC2 console

Go to the EC2 console and click on **Instances**:

Amazon Web Services

Compute

-  **EC2**
Virtual Servers in the Cloud
-  **EC2 Container Service**
Run and Manage Docker Containers
-  **Elastic Beanstalk**
Run and Manage Web Apps
-  **Lambda**
Run Code in Response to Events

AWS ▾ Serv

EC2 Dashboard

Events

Tags

Reports

Limits

INSTANCES

Instances

Spot Requests

Reserved Instances

Commands

Dedicated Hosts

IMAGES

Select the **Monitoring Scripts** instance. Then go to the **Monitoring** tab and then click **Create Alarm**:

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public
Monitoring Scripts	i-4b86678c	t2.micro	us-west-2b	running	2/2 checks ...	None	ec2-54
HighCPUInstance	i-7a8465bd	m3.medium	us-west-2b	running	2/2 checks ...	ALARM	ec2-54

Description | Status Checks | Monitoring | Tags

CloudWatch alarms: ☒ No alarms configured Create Alarm

CloudWatch metrics: Basic monitoring, Enable Detailed Monitoring Showing data for: Last Hour

Below are your CloudWatch metrics for the selected resources (a maximum of 10). Click on a graph to see an expanded view. All times shown

A screen will appear to configure the details of the alarm. It is similar to the screen we saw in the last step and you should feel free to configure the details for yourself. In the end, you should have something similar to it:

Create Alarm

You can use CloudWatch alarms to be notified automatically whenever metric data reaches a level you define.

To edit an alarm, first choose whom to notify and then define when the notification should be sent.

☒ Send a notification to: AlarmTopic (wadawd@john.com) create topic

☐ Take the action:

- ☐ Recover this instance
- ☐ Stop this instance
- ☐ Terminate this instance
- ☐ Reboot this instance

Whenever: Average of CPU Utilization

 Is: Percent

 For at least: 1 consecutive period(s) of 5 Minutes

 Name of alarm: High CPU - Monitoring Scripts

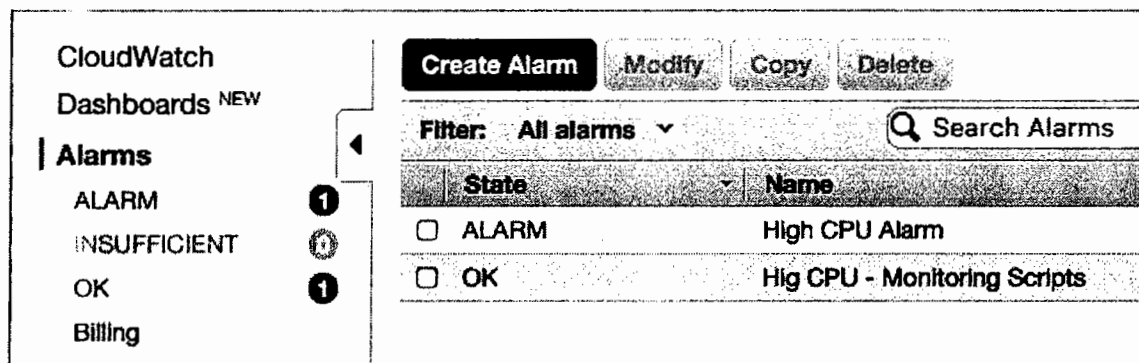
Cancel
Create Alarm

CPU Utilization Percent

Click on **Create Alarm** and a new screen will appear. A message will appear saying that the alarm was created. You can click on **Close**.

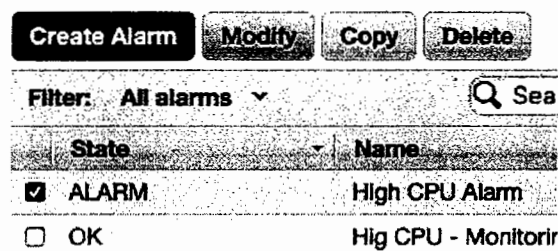
Setting EC2 Actions to the Alarms

Now that we have explored two ways of creating an alarm, let's make sure that they are helping us somehow. Let's go back to the **Alarms** page in the **CloudWatch console** and we can see that the first alarm we created is stuck in the ALARM state:



This happens because this instance is running an application that consumes 100% of the CPU utilization. You probably won't have anything like this in a production environment. But, let's imagine that we are managing a production environment and we have an instance that is becoming unavailable intermittently because of the CPU utilization. We sure would like to receive a notification every time the CPU utilization is high, but this can happen anytime, in the middle of the night, or during a weekend, so it would be nice to have a pre-defined action in these cases -- at least until we find a definitive solution for the problem.

To help us with that, we can set EC2 actions to our alarms. Now, let's modify the **High CPU Alarm** that we created for the **HighCPUInstance**. To do this, just select the alarm that you want to edit, and click on **Modify**:



A screen will appear for us. In the **Actions** area, we need to click on + **EC2 Action** to configure our action.

To make our alarm more suitable to our needs, let's change the threshold from 50% of CPU Utilization to 70% and set a new **EC2 Action** to **Reboot this instance** whenever the state of this alarm is ALARM. Once you fill in all the necessary info, you should have something like this:

Modify Alarm

1. Select Metric
2. Define Alarm

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

Name:

Description:

Whenever: CPUUtilization

Is:

For: consecutive period(s)

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line for a duration of 5 minutes

High CPU Alarm
CPUUtilization >= 70

Namespaces: AWS/EC2

InstanceId:

Metric Name:

Period:

Statistic:

Actions

Define what actions are taken when your alarm changes state.

Notification

Whenever this alarm:

Send notification to: [New list](#) [Enter list](#)

This notification list is managed in the SNS console.

EC2 Action

Whenever this alarm:

Take this action:

- ☒ Recover this instance
- ☐ Stop this instance
- ☐ Terminate this instance
- ☐ Reboot this instance

This will reboot your EC2 instance (i-381b98e2)

[+ Notification](#)
[+ AutoScaling Action](#)
[+ EC2 Action](#)

[Cancel](#)
[Previous](#)
[Next](#)
[Save Changes](#)

Click **Save Changes**.

Now we need to change the state of the alarm. This is needed because the current state of the alarm is ALARM, so CloudWatch will not take any actions until the state changes to something else. Let's stop our instance to change the state of the alarm to INSUFICIENT DATA. To do that, go to the EC2 console, and in the Instances page, select the HighCPUInstance and stop it for a few minutes:

EC2 Dashboard

Events

Tags

Reports

Limits

INSTANCES

Instances

Spot Requests

Reserved Instances

Commands

Launch Instance

Connect

Actions

Filter by tags and attributes or

	Name	Instance	Availability Zone
<input type="checkbox"/>	Monitoring Scripts	i-4b866	
<input checked="" type="checkbox"/>	HighCPUInstance	i-7a846	

Connect

Launch More Like This

Instance State

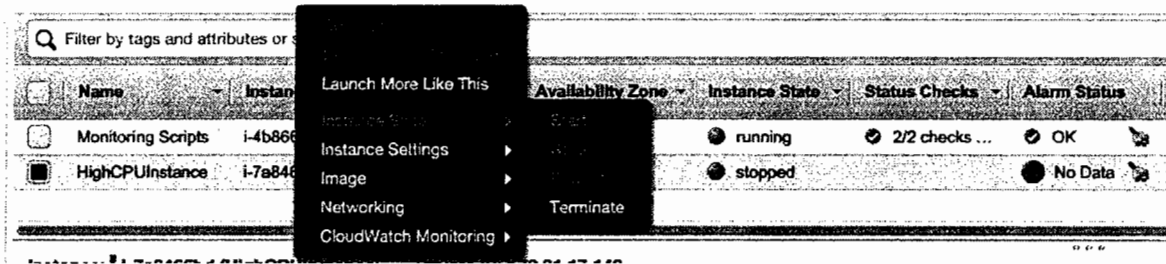
Instance Settings

Image

Networking

CloudWatch Monitoring

Once the **Alarm Status** of the instance changes to **No Data** you can start the instance again:

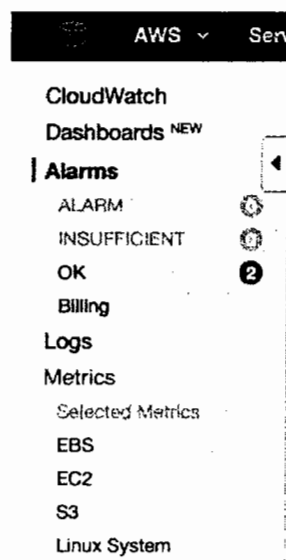


If you feel that it is taking too much time you can delete the current alarm and create a new one from scratch with an email notification and an EC2 Action. That will do the job as well.

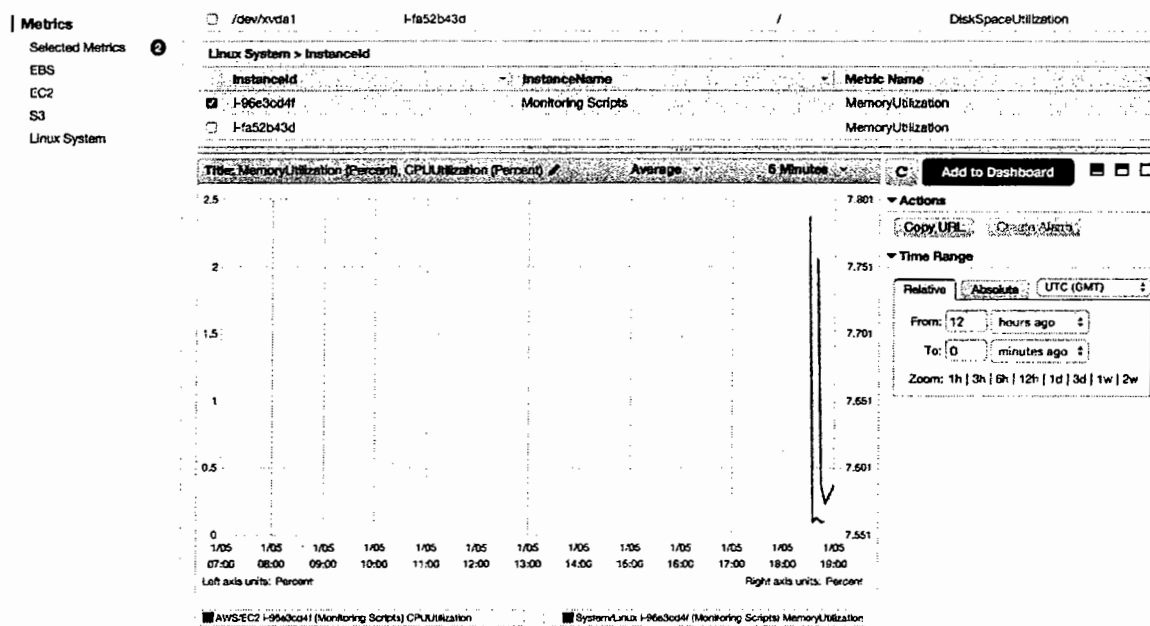
Now you can remain in the **Instances** page in the EC2 console and watch CloudWatch reboot the instance every time the **Alarm Status** changes to ALARM. Later on, you can go to the **Alarms** in the CloudWatch console and see the History of the alarm. Take some time here to watch things and adjust the settings in the alarm and change its behavior. I encourage you to not to limit yourself to these alarms only, create more alarms and test them by yourself.

STEP 7: Analyzing CloudWatch metrics over time

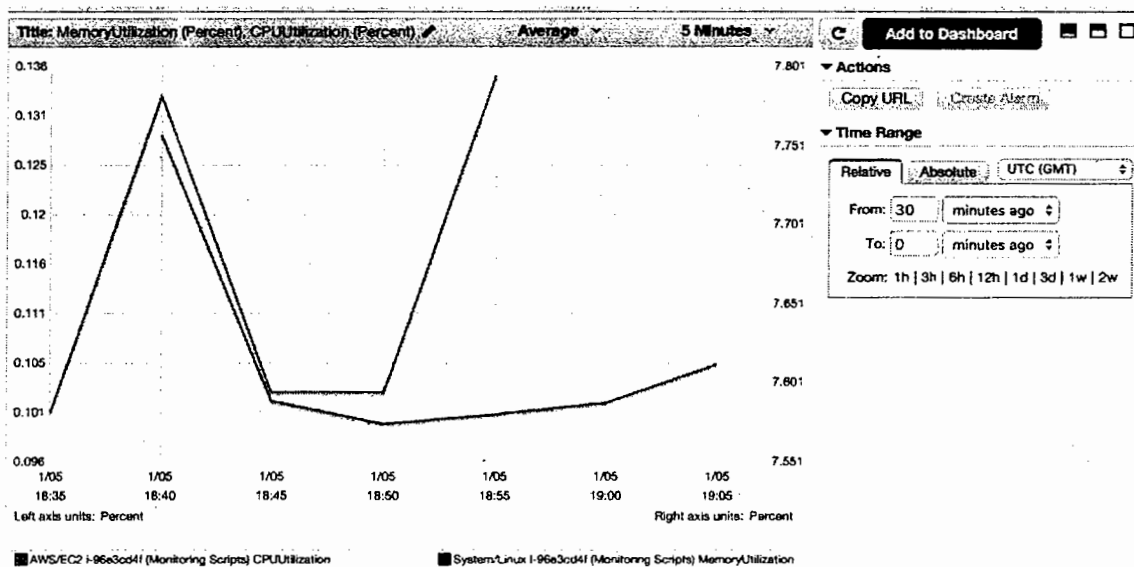
Go to the CloudWatch console and click on Metrics:



Now Select two metrics. In this case I will select the **CPUUtilization** and **MemoryUtilization** of the **Monitoring Scripts** instance. In the end you should have something like this:

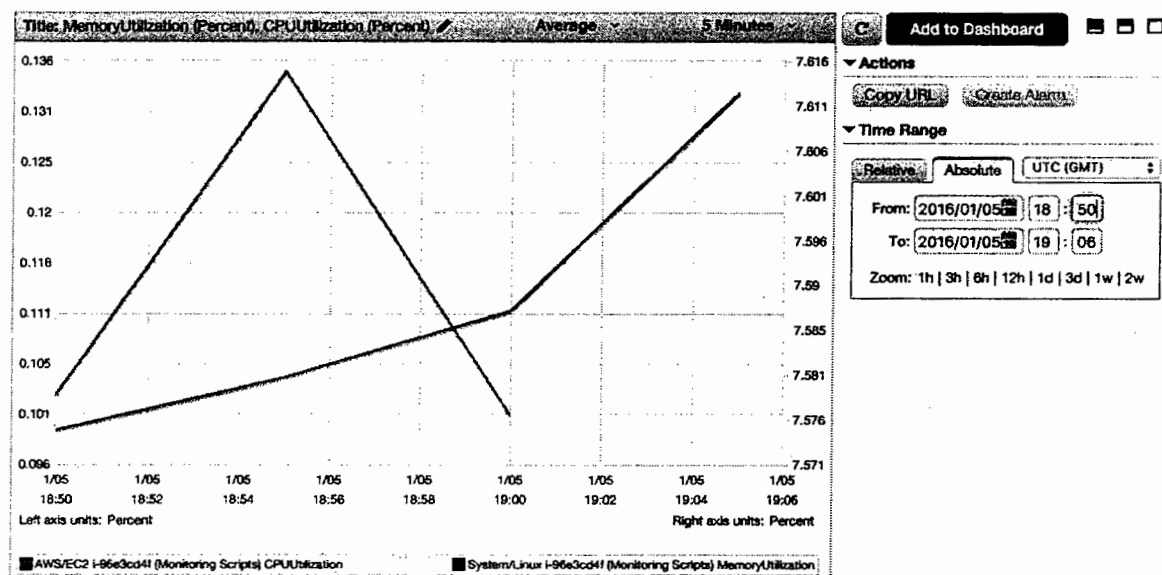


Now let's play with this graph. We will change the Relative time to show what happened in just the last 30 minutes. To do this, simply change the values and hit enter on the number field once you're done:



Now we can see, in greater detail, what happened in the last 30 minutes, and analyze what happens with the memory utilization and the CPU utilization goes up and down.

Another cool feature is the ability to set an Absolute time to check the metrics. This is useful in case we want to troubleshoot what happened during an outage, like in the last step's example, we could use it to examine what happened with a particular instance when the High CPU Alarms changed its state to ALARM. Do this by clicking on the **Absolute** tab and selecting the exact time that you want to review:



As soon as you find something interesting, you can share it with others. To do that click on Copy URL, and notice that you need to be logged in the AWS console in order to see the link. That means you can only share links with people who have access to your AWS account.

Take your time exploring how to use the Graph tool and how to see metrics over time. An interesting thing to watch might be what happens with the EBS volume associated with the HighCPUInstance when the CPUUtilization goes up.