Monitoring & Logging

AWS CLoudWatch : Publish metrics

Objective:
AWSCloudwatch service is used to monitor AWS resources.

AWSCloudwatch is a

- Monitoring/Alerting service
- Metrics gathering service
- Logging service
- Graphing service

CloudWatch stores data about a metric as a series of data points and each data point can be published as an aggregated set called statistic set.

High-Resolution Metrics

Each metric is one of the following:

- Standard resolution, with data having a one-minute granularity
- High resolution, with data at a granularity of one second

Metrics produced by AWS services are standard resolution by default. When you publish a custom metric, you can define it as either standard resolution or high resolution. When you publish a high-resolution metric, CloudWatch stores it with a resolution of 1 second, and you can read and retrieve it with a period of 1, 5, 10, 30 seconds, or any multiple of 60 seconds.

Immediate insight into your application's sub-minute activity is made possible by High-resolution metrics. Every `PutMetricData` call for a custom metric is charged, so calling `PutMetricData` more often on a high-resolution metric can lead to higher charges.

Using Dimensions

In custom metrics, the `--dimensions` parameter is common. A dimension further clarifies what the metric is and what data it stores. You can have up to 10 dimensions in one metric, and each dimension is defined by a name and value pair.

How you specify a dimension is different when you use different commands. With put-metric-data, you specify each dimension as *`MyName`*=*`MyValue`*.

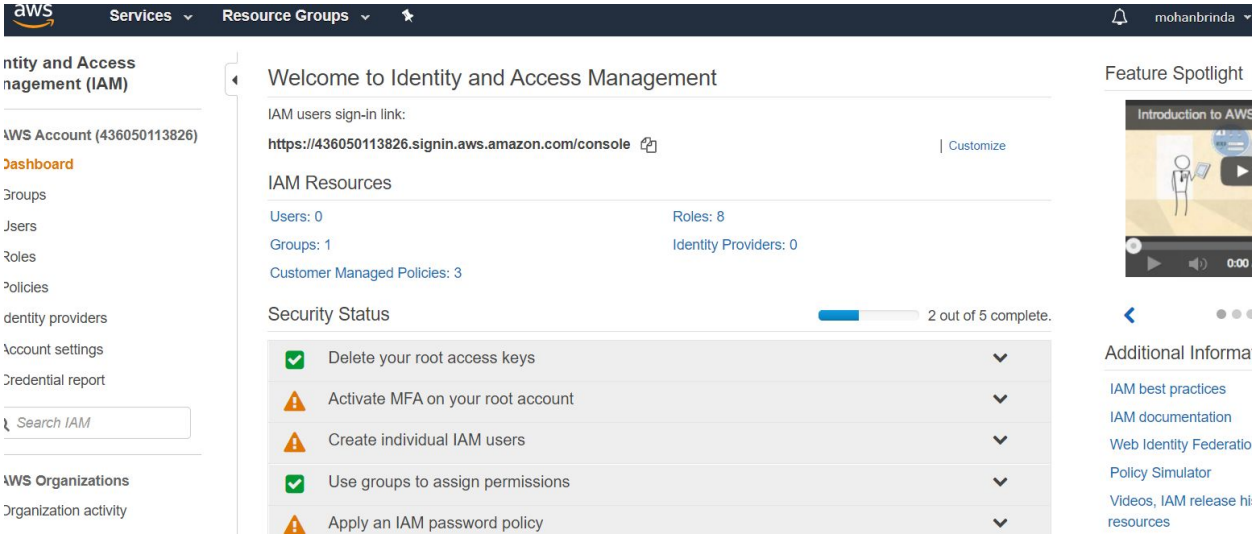
Publishing Statistic Sets

Prior to publishing data to CloudWatch, data can be aggregated. Aggregating data minimizes the number of calls to put-metric-data when there are multiple data points per minute. For example, using the `--statistic-values` parameter instead of calling **put-metric-data** multiple times for three data points that are within 3 seconds of each other, the data can be aggregated into a statistic set that you publish with one call.

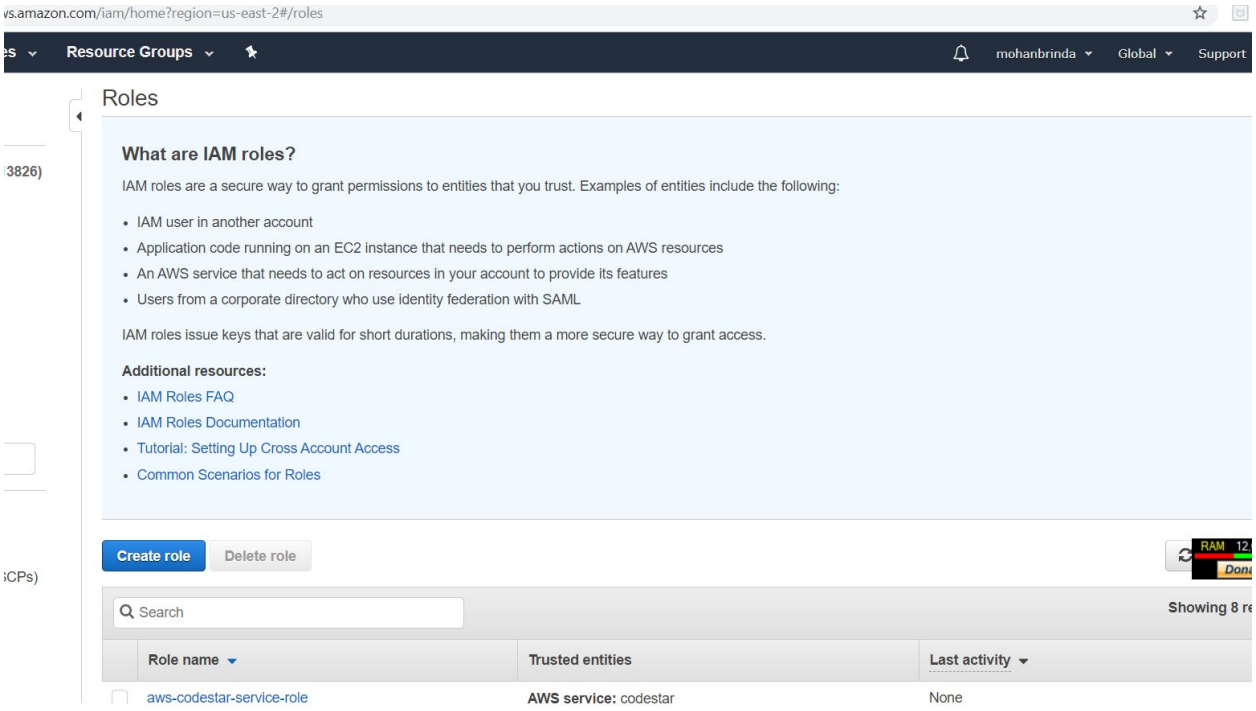
AWSCloudWatchMetrics

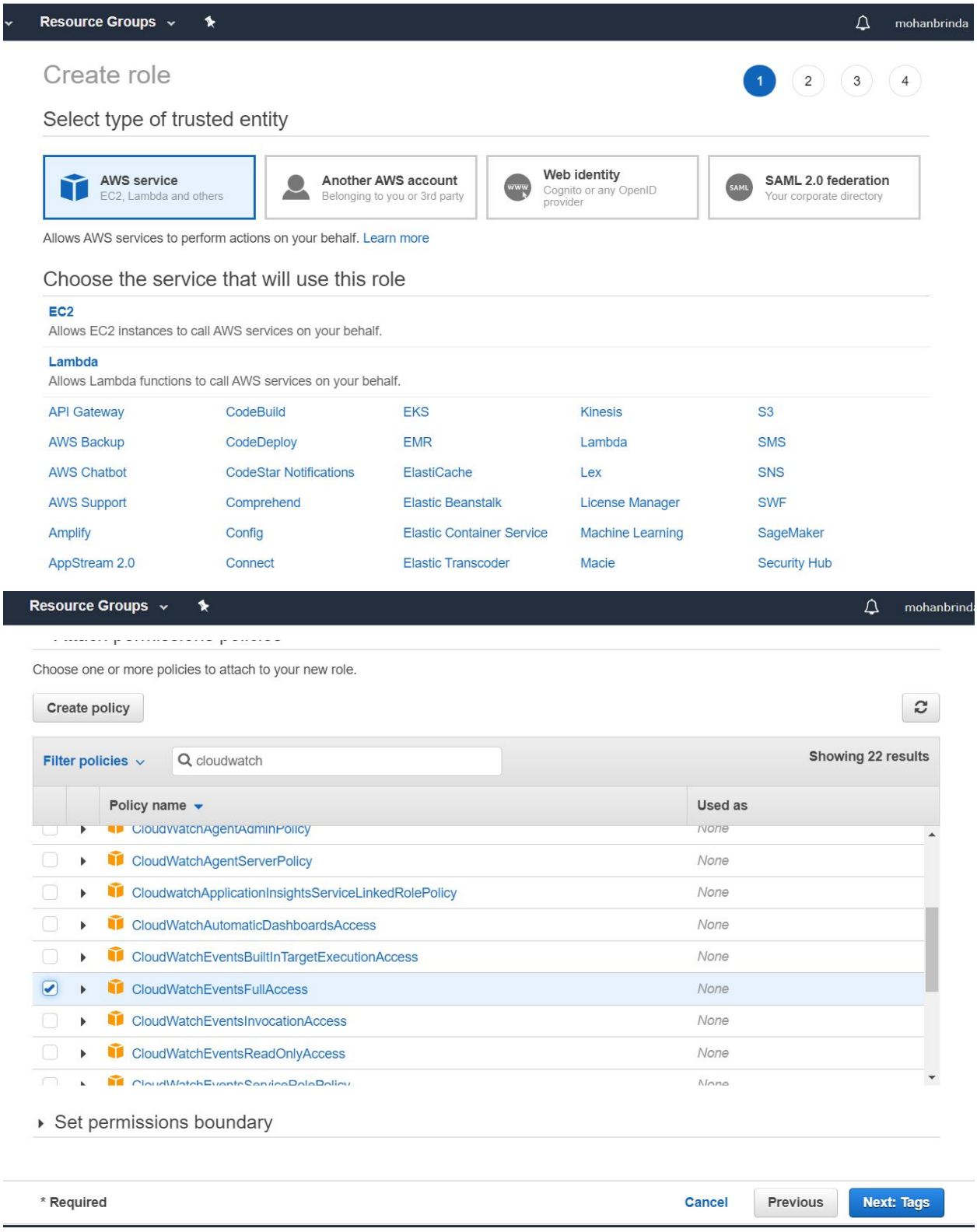
- First, Login to the IAM console and give ec2 instance and access to write to cloudwatch
- Next, SSH into ec2 instance and start pushing some data in order to graph it.

Login to AWS console and select IAM.


Services
Resource Groups
mohanbrinda
ntity and Access
nagement (IAM)
WS Account (436050113826)
Dashboard
Groups
Users
Roles
Policies
dentity providers
Account settings
Credential report
Search IAM
WS Organizations
Organization activity
Welcome to Identity and Access Management
IAM users sign-in link:
https://436050113826.signin.aws.amazon.com/console
Customize
IAM Resources
Users: 0
Roles: 8
Groups: 1
Identity Providers: 0
Customer Managed Policies: 3
Security Status
2 out of 5 complete.
Delete your root access keys
Activate MFA on your root account
Create individual IAM users
Use groups to assign permissions
Apply an IAM password policy
Feature Spotlight
Introduction to AWS
0:00
Additional Informa
IAM best practices
IAM documentation
Web Identity Federatio
Policy Simulator
Videos, IAM release hi
resources

Create role


ws.amazon.com/iam/home?region=us-east-2#/roles
Resource Groups
mohanbrinda
Global
Support
Roles
What are IAM roles?
IAM roles are a secure way to grant permissions to entities that you trust. Examples of entities include the following:
IAM user in another account
Application code running on an EC2 instance that needs to perform actions on AWS resources
An AWS service that needs to act on resources in your account to provide its features
Users from a corporate directory who use identity federation with SAML
IAM roles issue keys that are valid for short durations, making them a more secure way to grant access.
Additional resources:
IAM Roles FAQ
IAM Roles Documentation
Tutorial: Setting Up Cross Account Access
Common Scenarios for Roles
Create role
Delete role
Search
Showing 8 re
Role name
Trusted entities
Last activity
aws-codestar-service-role
AWS service: codestar
None


Resource Groups
mohanbrinda
Create role
1
2
3
4
Select type of trusted entity
AWS service
EC2, Lambda and others
Another AWS account
Belonging to you or 3rd party
Web identity
Cognito or any OpenID provider
SAML 2.0 federation
Your corporate directory
Allows AWS services to perform actions on your behalf. Learn more
Choose the service that will use this role
EC2
Allows EC2 instances to call AWS services on your behalf.
Lambda
Allows Lambda functions to call AWS services on your behalf.
API Gateway
AWS Backup
AWS Chatbot
AWS Support
Amplify
AppStream 2.0
CodeBuild
CodeDeploy
CodeStar Notifications
Comprehend
Config
Connect
EKS
EMR
ElastiCache
Elastic Beanstalk
Elastic Container Service
Elastic Transcoder
Kinesis
Lambda
Lex
License Manager
Machine Learning
Macie
S3
SMS
SNS
SWF
SageMaker
Security Hub
Resource Groups
mohanbrinda
Choose one or more policies to attach to your new role.
Create policy
Filter policies
cloudwatch
Showing 22 results
Policy name
Used as
CloudWatchAgentAdminPolicy
None
CloudWatchAgentServerPolicy
None
CloudwatchApplicationInsightsServiceLinkedRolePolicy
None
CloudWatchAutomaticDashboardsAccess
None
CloudWatchEventsBuiltInTargetExecutionAccess
None
CloudWatchEventsFullAccess
None
CloudWatchEventsInvocationAccess
None
CloudWatchEventsReadOnlyAccess
None
Set permissions boundary
* Required
Cancel
Previous
Next: Tags

Create role

1 2 3 4

Add tags (optional)

IAM tags are key-value pairs you can add to your role. Tags can include user information, such as an email address, or can be descriptive, such as a job title. You can use the tags to organize, track, or control access for this role. Learn more

Key	Value (optional)	Remove
Add new key		

You can add 50 more tags.

Cancel Previous Next: Review

Resource Groups mohanbrinda

Create role

1 2 3 4

Review

Provide the required information below and review this role before you create it.

Role name* Cloudwatch

Use alphanumeric and '+=,.@-_' characters. Maximum 64 characters.

Role description Allows EC2 instances to call AWS services on your behalf.

Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.

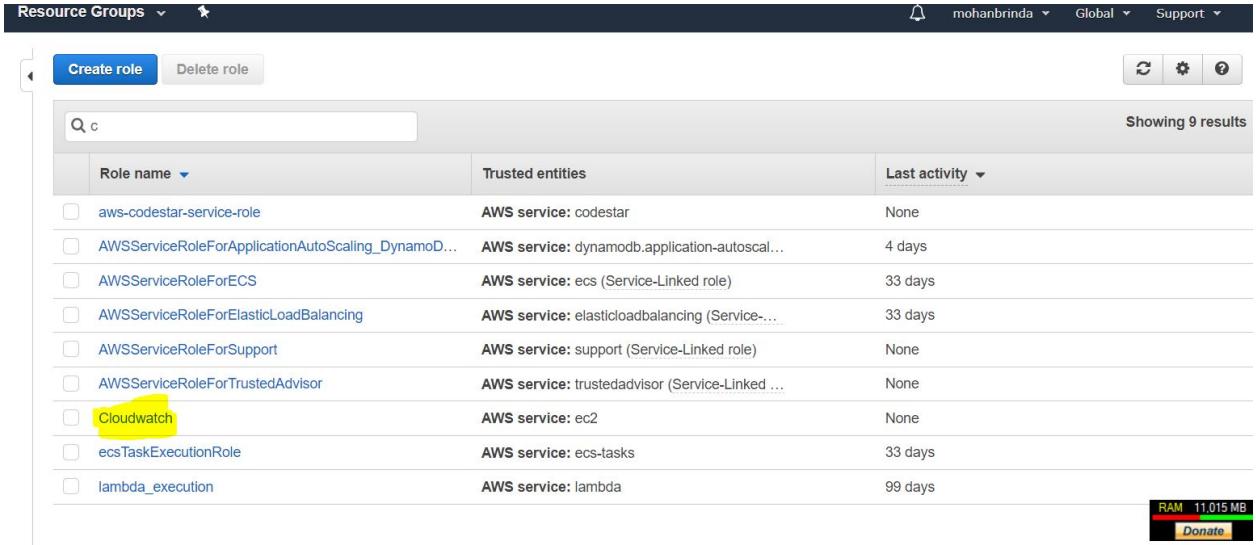
Trusted entities AWS service: ec2.amazonaws.com

Policies CloudWatchEventsFullAccess

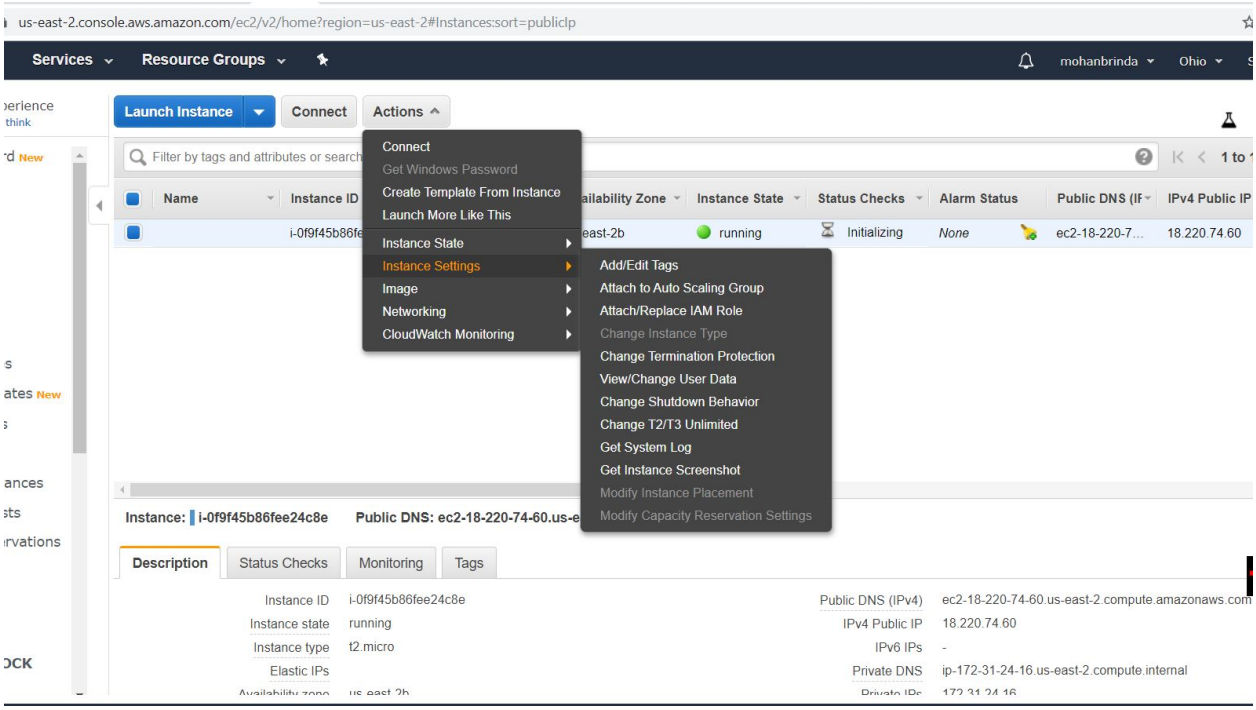
Permissions boundary Permissions boundary is not set

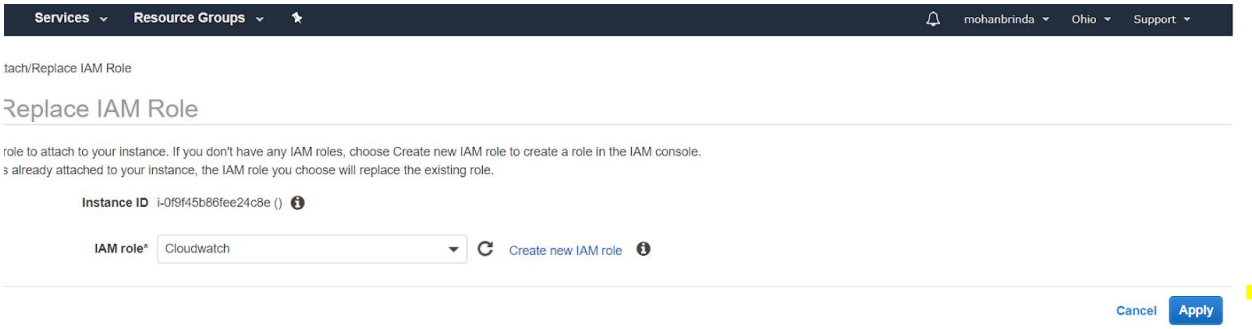
No tags were added.

* Required Cancel Previous Create role

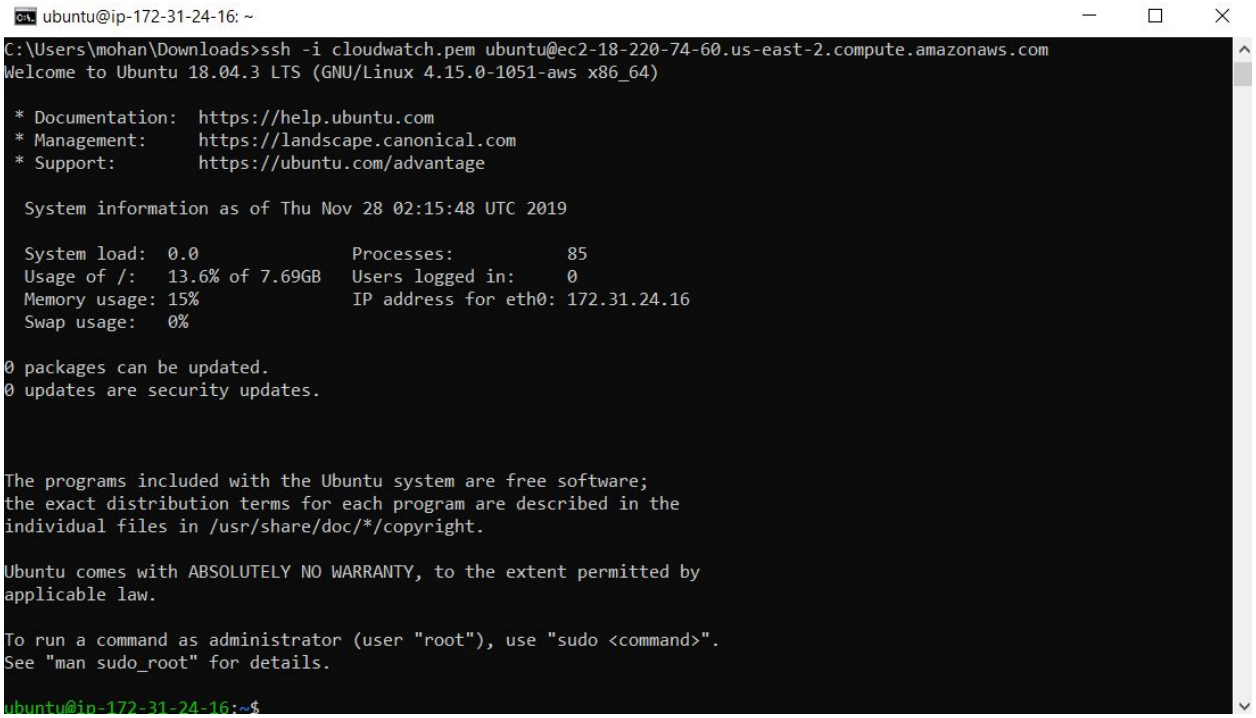


The role has been created.
Next assign the role (Cloudwatch) to Ec2 instance.

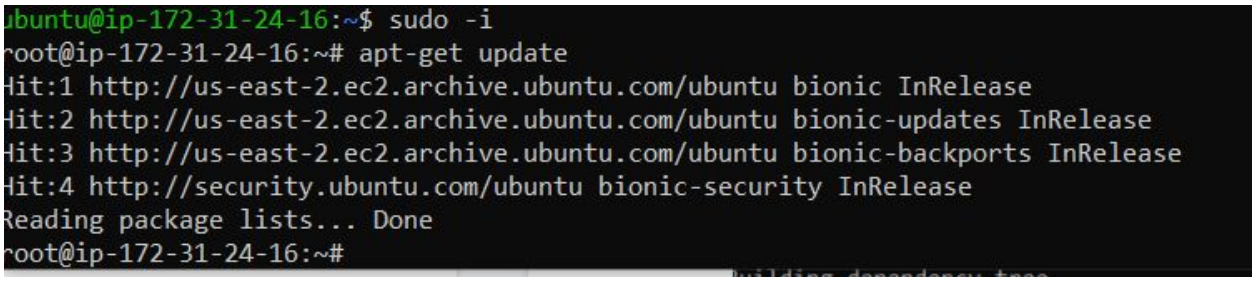




SSH and connect to the instance



Update and Install awscli



```
oot@ip-172-31-24-16:~# apt-get install awscli
eading package lists... Done
uilding dependency tree
eading state information... Done
wscli is already the newest version (1.14.44-1ubuntu1).
 upgraded, 0 newly installed, 0 to remove and 53 not upgraded.
oot@ip-172-31-24-16:~#
```

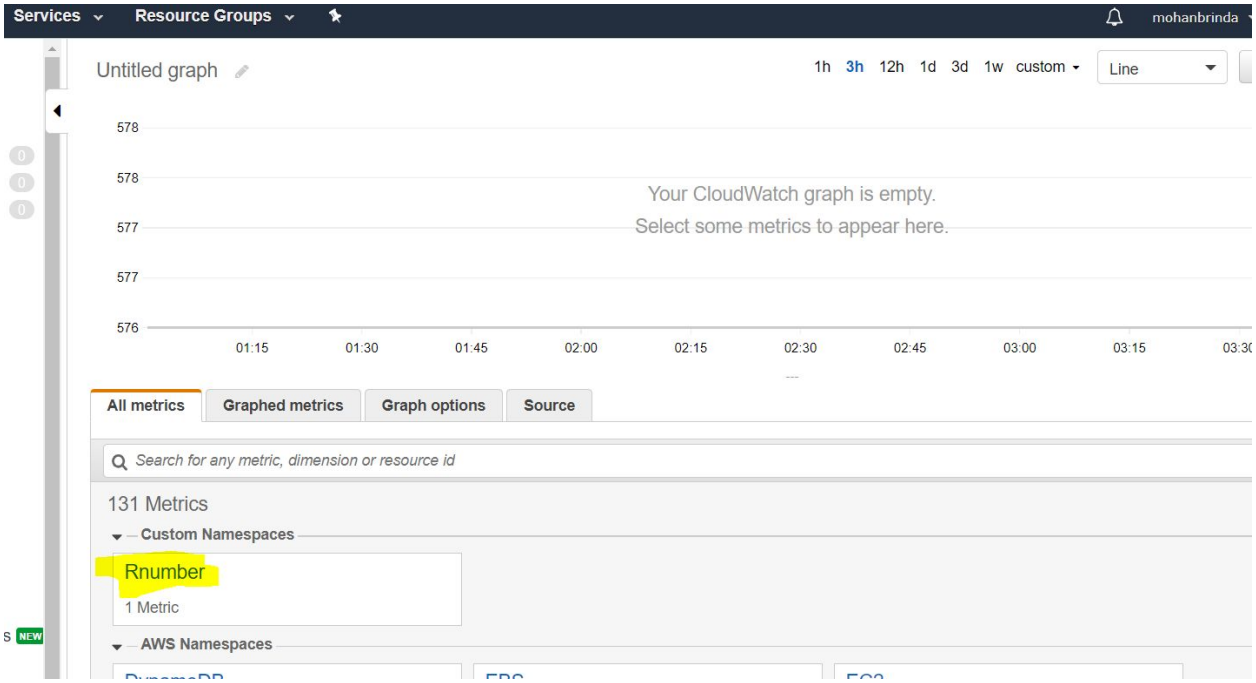
Generate some random numbers between 1 and 1000 using shuf command and push it to cloudwatch. A graph will be presented with the given data points.

```
root@ip-172-31-24-16:~# shuf -i 1-1000 -n 1
776
root@ip-172-31-24-16:~#
root@ip-172-31-24-16:~# shuf -i 1-1000 -n 1
549
root@ip-172-31-24-16:~# shuf -i 1-1000 -n 1
173
root@ip-172-31-24-16:~# shuf -i 1-1000 -n 1
223
root@ip-172-31-24-16:~# shuf -i 1-1000 -n 1
113
root@ip-172-31-24-16:~# shuf -i 1-1000 -n 1
230
root@ip-172-31-24-16:~# shuf -i 1-1000 -n 1
863
root@ip-172-31-24-16:~#
```

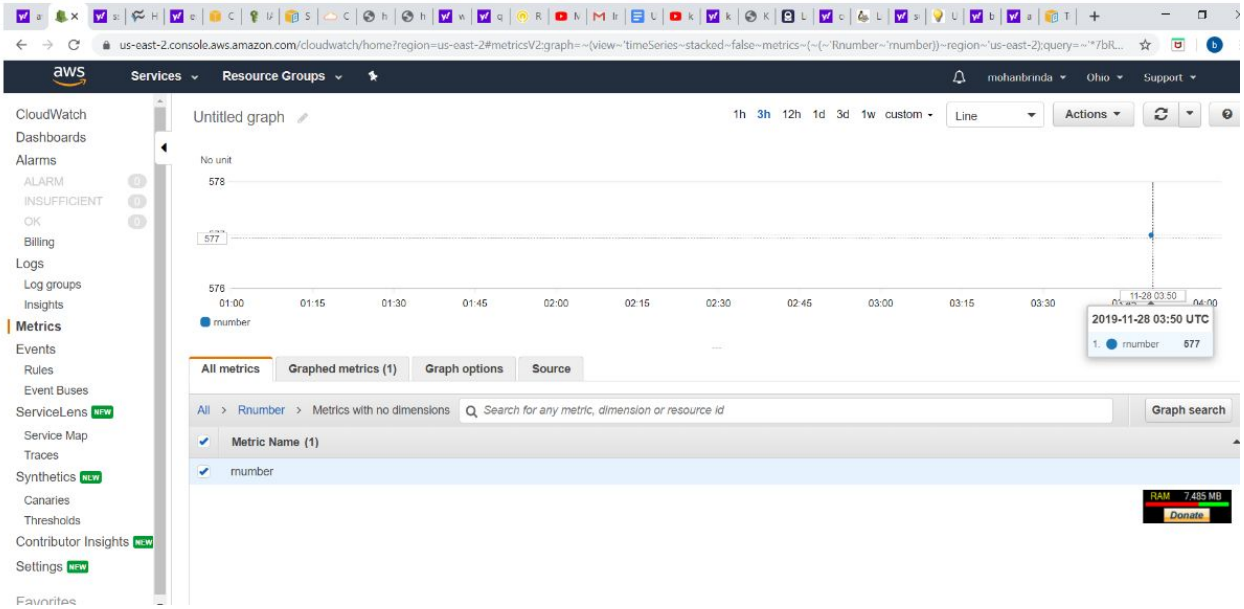
NEXT, put the data points into cloudwatch using the following command, the command inside the backtix symbol(command substitution) will be replaced with the results in the command.

```
root@ip-172-31-24-16:~#
root@ip-172-31-24-16:~# aws cloudwatch put-metric-data --metric-name rnumber --namespace Rnumber --value  ` shuf -i 1-10
00 -n 1 ` --region=us-east-2
root@ip-172-31-24-16:~# aws cloudwatch put-metric-data --metric-name rnumber --namespace Rnumber --value  ` shuf -i 1-10
00 -n 1 ` --region=us-east-2
root@ip-172-31-24-16:~# aws cloudwatch put-metric-data --metric-name rnumber --namespace Rnumber --value  ` shuf -i 1-10
00 -n 1 ` --region=us-east-2
root@ip-172-31-24-16:~# aws cloudwatch put-metric-data --metric-name rnumber --namespace Rnumber --value  ` shuf -i 1-10
00 -n 1 ` --region=us-east-2
root@ip-172-31-24-16:~#
```

In the AWS console, choose Cloudwatch and select merics, the newly created Rnumber will be displayed along with the data point.



The following graph shows one datapoint with random number 577.



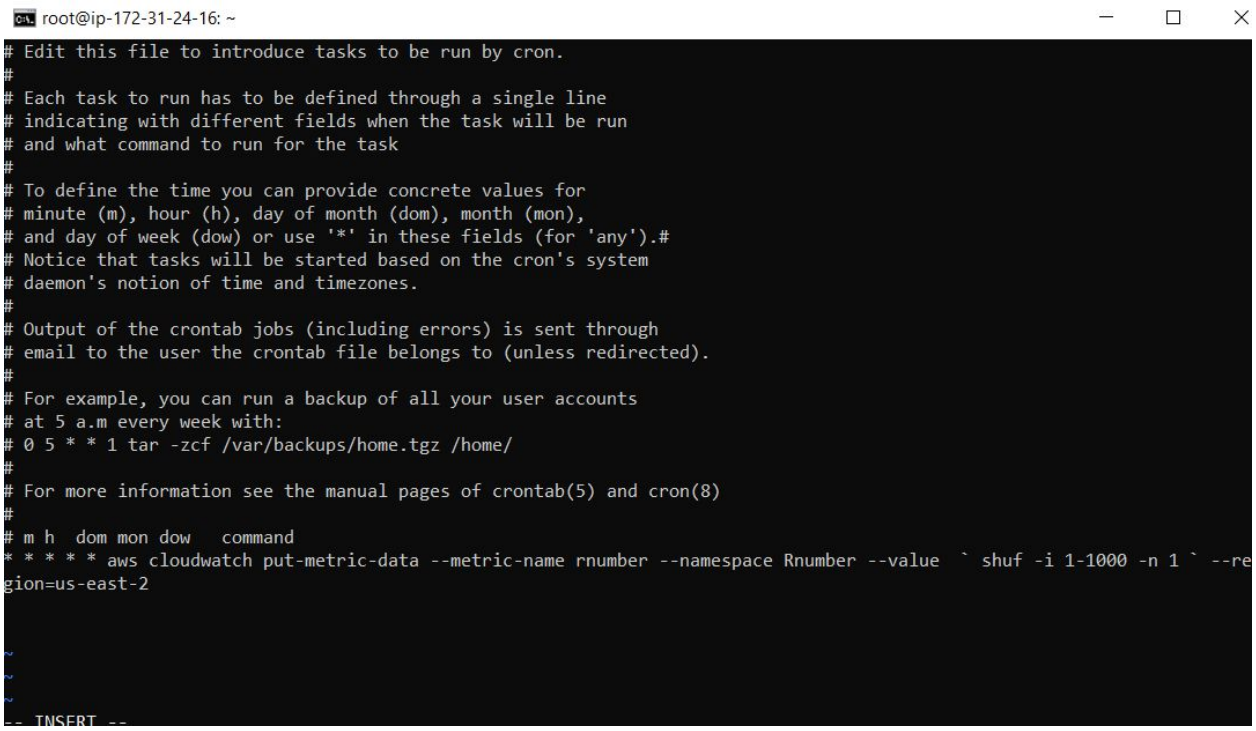
Next setup a cron job to push random number every minute to the graph.

```
root@ip-172-31-24-16:~# crontab -e
no crontab for root - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.basic
  3. /usr/bin/vim.tiny
  4. /bin/ed

Choose 1-4 [1]: 3
```

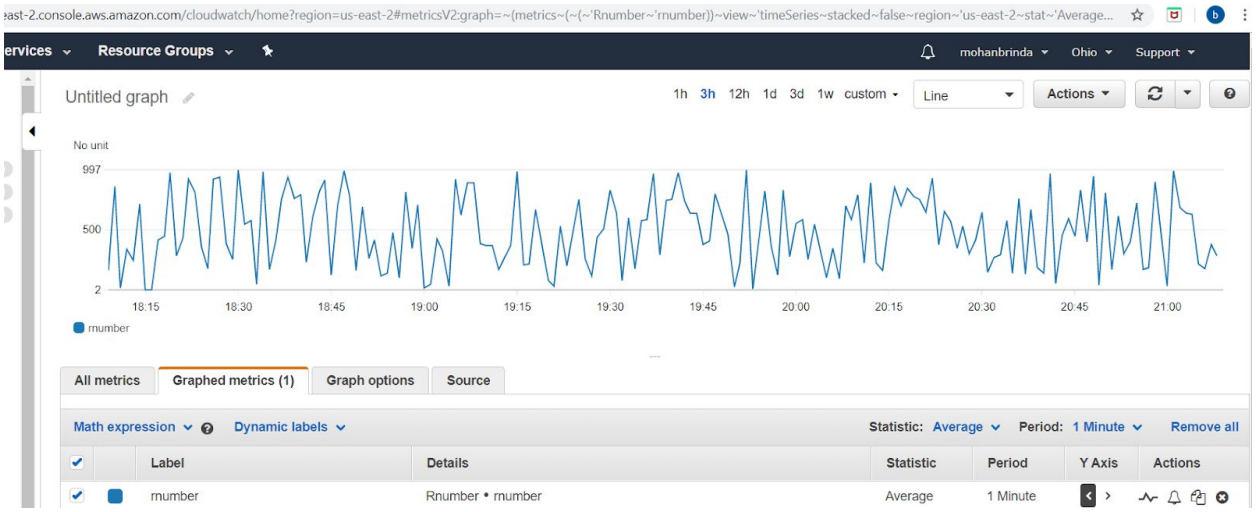
Edit the file by adding five * (minute, hour, day of month, every month, every day of the week) and paste the command for generating random numbers.



```
root@ip-172-31-24-16:~# crontab -e
no crontab for root - using an empty one

Select an editor.  To change later, run 'select-edit
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.basic
  3. /usr/bin/vim.tiny
  4. /bin/ed

Choose 1-4 [1]: 3
crontab: installing new crontab
"/tmp/crontab.XeBXl7/crontab":23: bad minute
errors in crontab file, can't install.
Do you want to retry the same edit? (y/n) y
crontab: installing new crontab
root@ip-172-31-24-16:~#
```



Thus a user can push their own metrics into cloudwatch, can make graphs, set alerts for example (set alarm when it reaches 1000) etc.

Reference:
Aws.amazon.com