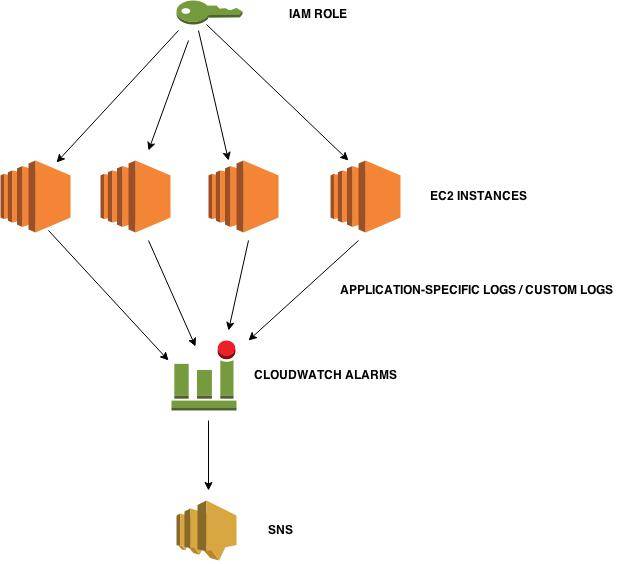
Logs Monitoring Using AWS CloudWatch

## AWS CloudWatch is a monitoring and alerting service that integrates with most AWS services like **EC2 or RDS**. It can monitor system performance in near real time and generate alerts based on thresholds you set.

### Some Basic Terms

Before going any further, let’s talk about two important concepts.  
CloudWatch Logs are arranged in what’s known as *Log Groups* and *Log Streams*. Basically, a log stream represents the source of your log data. For example, Nginx error logs streaming to CloudWatch will be part of one log stream. Java logs coming from app servers will be part of another log stream, database logs would form another stream and so on. In other words, each log stream is like a channel for log data coming from a particular source.  
Log groups are used to classify log streams together. A log group can have one or multiple log streams in it. Each of these streams will share the same retention policy, monitoring setting or access control permissions. For example, your “Web App” log group can have one log stream for web servers, one stream for app servers and another for database servers. You can set a retention policy of, say, two weeks for this log group, and this setting will be applied to each of the log streams.

Amazon CloudWatch now provides us the flexibility to monitor, maintain, store and access our custom log files, log files from EC2 Servers, CloudTrail and other resources. We can also generate alerts on those logs. This will help us in troubleshooting our servers by monitoring all the appplication-specific logs on CloudWatch in real time.



The following image shows the different AWS resources monitored by Amazon CloudWatch.

## Amazon CloudWatch -Edureka

Amazon CloudWatch allows administrators to easily monitor multiple instances and resources from one console by performing the below tasks :

* Enables robust monitoring of resources like :
  1. Virtual instances hosted in Amazon EC2
  2. Databases located in Amazon RDS
  3. Data stored in Amazon S3
  4. Elastic Load Balancer
  5. Auto-Scaling Groups
  6. Other resources
* Monitors, stores and provides access to system and application log files
* Provides a catalog of standard reports that you can use to analyze trends and monitor system performance
* Provides various alert capabilities, including rules and triggers high resolutions alarms and sends notifications
* Collects and provides a real-time presentation of operational data in form of key metrics like CPU utilization, disk storage etc.

Now we know why users choose CloudWatch, which is, for its automatic integration with AWS services, its flexibility, and its ability to scale quickly. But how does Amazon CloudWatch achieve this?

## ****Amazon CloudWatch In Action****

Before learning how Amazon CloudWatch operates there are certain primary concepts that you need to know. Let’s have a look at those concepts.

### ****Metrics****

* ***Metrics*** represents a time-ordered set of data points that are published to CloudWatch
* You can relate metric to a variable that is being monitored and data points to the value of that variable over time
* Metrics are uniquely defined by a name, a namespace, and zero or more dimensions
* Each data point has a time-stamp.

### ****Dimensions****

* A ***dimension*** is a name/value pair that uniquely identifies a metric
* Dimensions can be considered as categories of characteristics that describe a metric
* Because dimensions are unique identifiers for a metric, whenever you add a unique name/value pair to one of your metrics, you are creating a new variation of that metric.

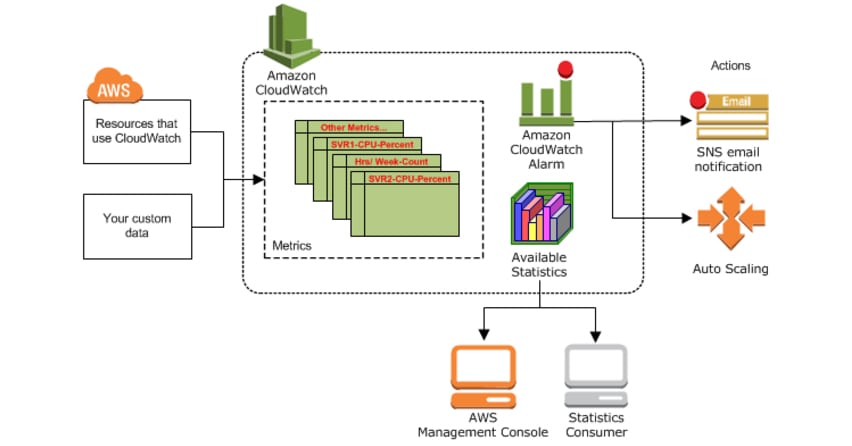
### ****Statistics****

* **Statistics** are metric data aggregations over specified periods of time
* Aggregations are made using the namespace, metric name, dimensions within the time period you specify
* Few available statistics are maximum, minimum, sum, average and sample count.

### ****Alarm****

* An **alarm** can be used  to automatically initiate actions on your behalf
* It watches a single metric over a specified time period and performs one or more specified actions
* The action is a simply a  notification that is sent to Amazon SNS topic.

Now let’s have a look at how Amazon CloudWatch works. The following diagram shows the conceptual view of how CloudWatch provides robust monitoring.



Amazon CloudWatch has system-wide visibility into your AWS resources and applications. It will monitor your resource files and generate key metrics based on your application’s log files. Key metrics include CPU usage, CPU latency, Network traffic, Disk storage etc. Based on these metrics it provides a real-time summary of system activity and individual resources.

CloudWatch also provides a comprehensive at-a-glance view of AWS infrastructure to keep track of application performance, spot trends and troubleshoot operational issues. In addition, Amazon CloudWatch configures high-resolution alarms and sends real-time notifications in case of sudden operational changes in AWS environment.

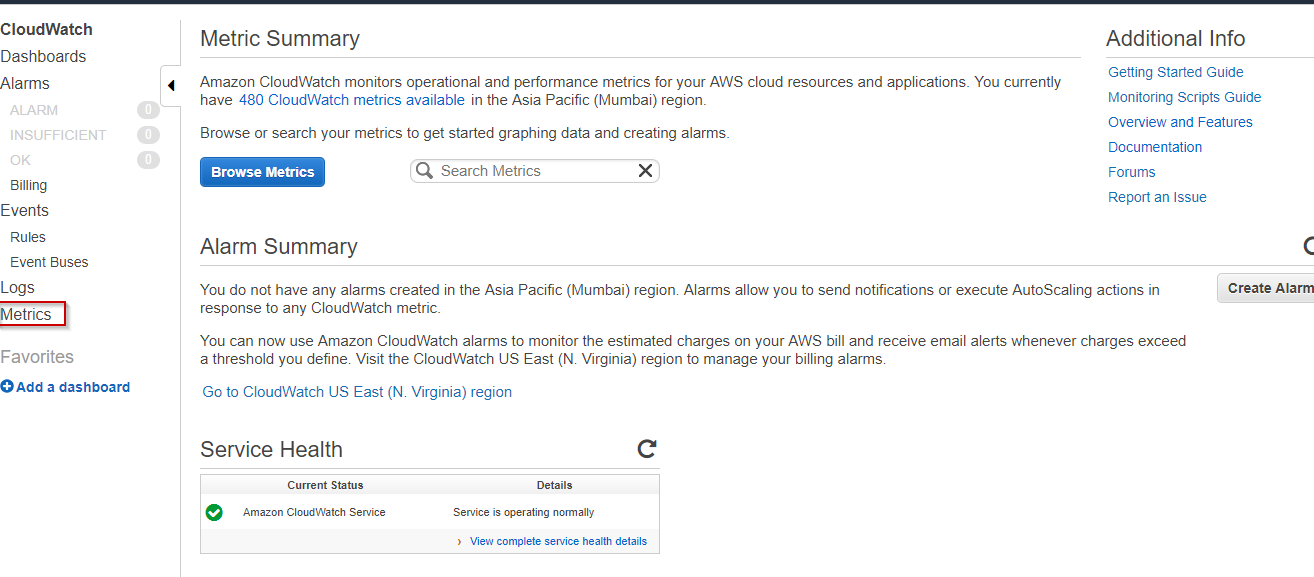
Now that you are familiar with Amazon CloudWatch concepts and its operation let’s have a look at how you can use Amazon CloudWatch to monitor your Amazon EC2 instance.

**Use Case**: Configure Amazon CloudWatch to send a notification when CPU Utilization of an instance is lower than 15%.

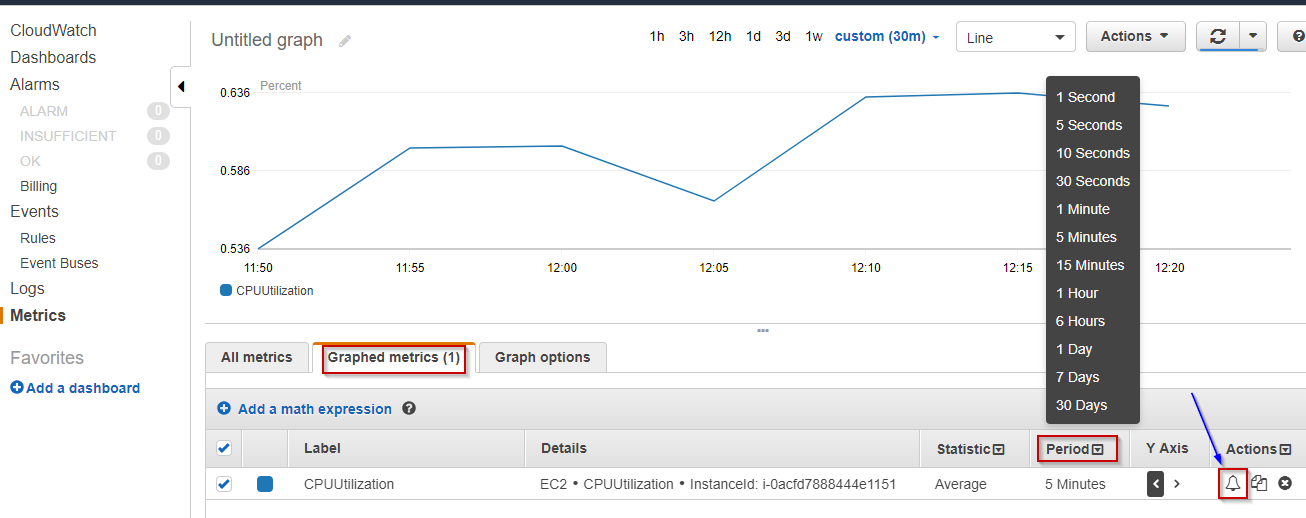
Lets go through various steps involved.

### ****Step 1 : Creating a CPU utilization metric****

* Go to Amazon CloudWatch Management Console and select metrics from the navigation pane.

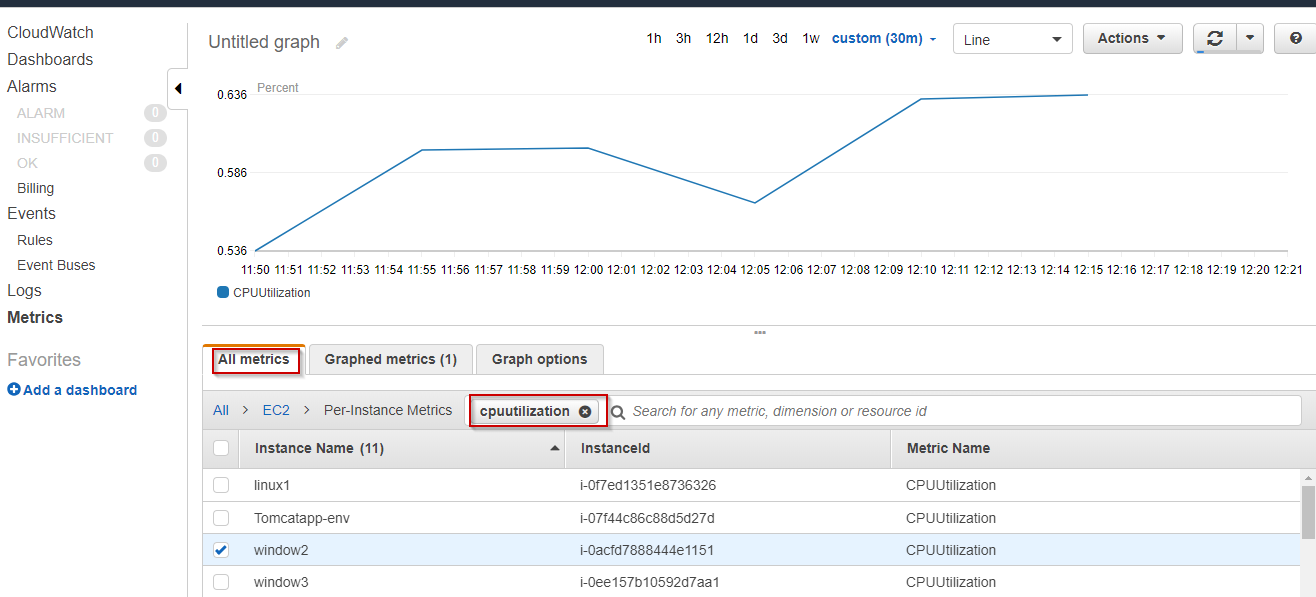


* On the metrics page type CPU Utilization in the search bar.
* From the displayed list of instances choose the instance for which you want to create a metric.

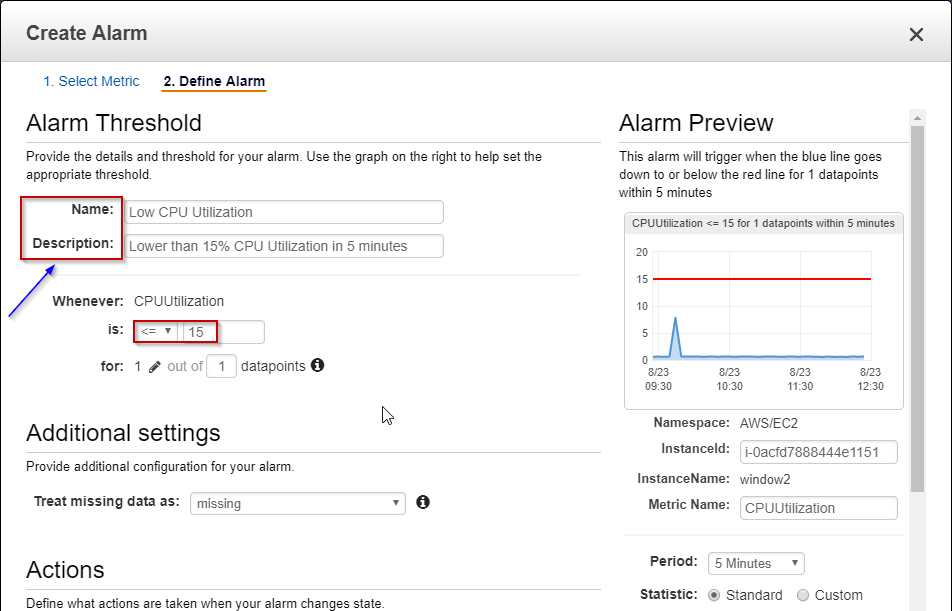


### ****Step 2 : Creating an alarm to notify when CPU Utilization metric of the instance is lower than 15%****

* Now select the Graphed Metrics option on the same page. Then set the time period according to your need. And choose an alarm icon located beside the selected instance.

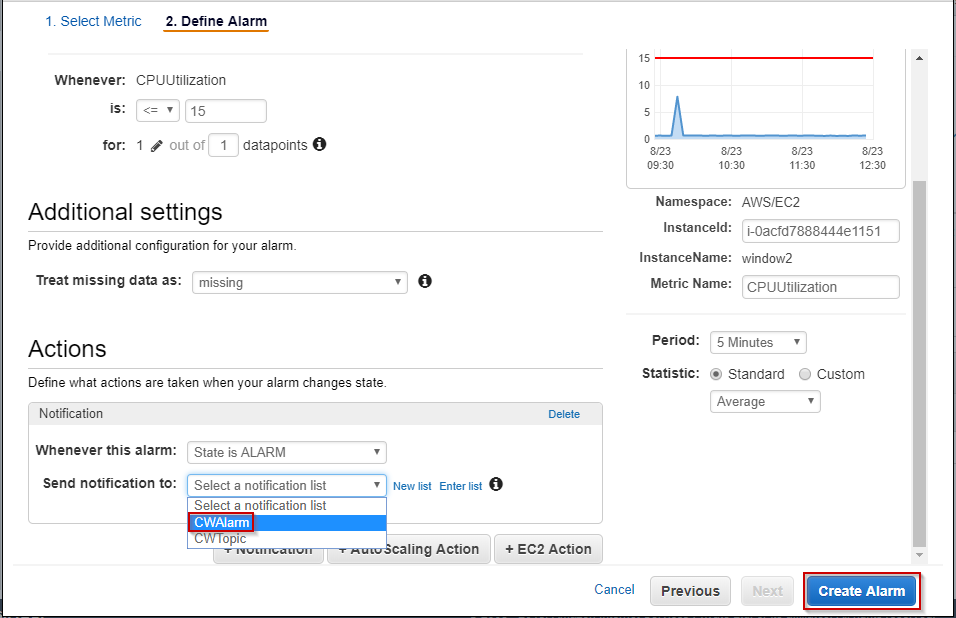


* Configure the alarm in the displayed dialog box. Give your alarm a name and description. Set the Threshold condition.



You want AWS to send you an email notification whenever the alarm condition is satisfied. The notification is sent through Amazon SNS Topic.

* Select New List option if want to add new email recipient, or If you want to choose the existing one, choose Enter List and enter the name of SNS topic.



* Click Create Alarm.

**Congratulations, you have successfully configured Amazon CloudWatch Alarm** to monitor your instance. You will receive the notification through an e-mail on the mail-id you have specified when the alarm condition is met.

Want To Be A Certified AWS Architect?Enroll Now

Now we will talk about the two most important segments of Amazon CloudWatch, which are :

* Amazon CloudWatch Events
* Amazon CloudWatch Logs

## ****Amazon CloudWatch Events****

Amazon CloudWatch Events deliver a real-time stream of system events from AWS resources to AWS Lambda functions, Amazon SNS Topics, Amazon SQS queues, and other target types.

CloudWatch Events enable you to create a set of rules that you can match certain events with. Then you can route these events to one or more targets like Lambda Function, SNS Topic etc. Whenever there are operational changes in your AWS environment, CloudWatch Events capture these changes and perform remedial actions by sending notifications, activating Lambda functions etc.

Let’s talk about certain topics that you need to understand before using CloudWatch Events.

### ****Events****

An event indicates a change in the AWS environment. AWS resources generate events when their state changes. Amazon allows you to generate your own custom application-level events and publish them to CloudWatch Events.

### ****Rules****

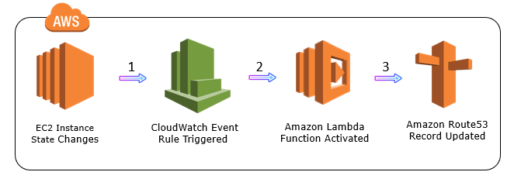
Rules are nothing but constraints. They evaluate every incoming event to determine if out-of-bounds scenario exists. If yes the event is then routed to target for processing. A single rule can route to multiple targets, all of which are processed in parallel.

### ****Targets****

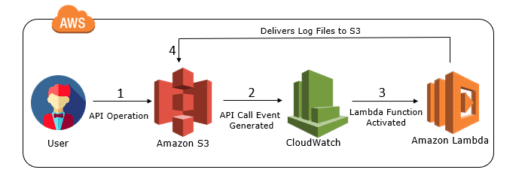
A target processes events. Targets can include Amazon EC2 instances, AWS Lambda functions, Kinesis streams, Amazon ECS tasks, Amazon SNS topics, Amazon SQS queues, and built-in targets. A target receives events in JSON format.

Now let’s have a look at situations where we can use Amazon CloudWatch Events.

**Use Case 1**: You can log the changes in the state of an Amazon EC2 instance by using CloudWatch Events with the assistance of AWS Lambda function.



**Use Case 2**: You can log the object-level API operations on your S3 buckets using CloudWatch Events. But prior to that, you should use AWS CloudTrail to set up a trail configured to receive these operations.



Well, these are just two use-cases which I have specified here so that you will have an idea about the capability of Amazon CloudWatch Events. To describe Amazon CloudWatch Events in one sentence, it is a service that allows you to track changes to your AWS resources with less overhead and more efficiency.

## ****Amazon CloudWatch Logs****

*Amazon CloudWatch Logs is used to monitor, store and access log files from AWS resources like Amazon EC2 instances, Amazon CloudTrail, Route53, and others.*

Let’s take a look at a few basic concepts of Amazon CloudWatch Logs. The below table gives an overview of those concepts.

|  |  |
| --- | --- |
| Log Events | Log Event is a record of some activity recorded by the application or resource being monitored |
| Log Streams | A log stream is a sequence of log events that share the same source. It represents the sequence of events coming from the application instance |
| Log Groups | Log groups represent groups of log streams that share the same retention, monitoring, and access control settings. Each log stream has to belong to one log group. |

With Amazon CloudWatch Logs you can troubleshoot your system errors and maintain and store the respective log files automatically. You can configure an alarm so that a notification will be sent when some error occurs in your system log. You can then troubleshoot the errors within minutes by accessing the original log data stored by CloudWatch Logs. Moreover, you can use Amazon CloudWatch Logs to:

* Store your log data in the highly durable storage
* Monitor your application log files in real-time for specific phrases, values or patterns
* Log information about the DNS queries that Route 53 receives
* Adjust the retention policy for each log group, by choosing a retention period between 10 years and one day.

Now that we have a foundation of Amazon CloudWatch lets go ahead and look at few reasons as to why it is the most famous cloud monitoring tool.

## ****Benefits of Amazon CloudWatch****

* Amazon CloudWatch allows you to access all your data from a single platform. It is natively integrated with more than 70 AWS services. Vodafone company uses Amazon CloudWatch with Auto Scaling groups to monitor CPU usage and to scale from three Amazon EC2 instances to nine during peak periods automatically.
* Provides real-time insights so that you can optimize operational costs and AWS resources. Kellogg company uses Amazon CloudWatch for monitoring, which helps the company make better decisions around the capacity they need, so that they can avoid wastage.
* Provides complete visibility across your applications, infrastructure stack, and AWS services. Atlassian uses Amazon CloudWatch to monitor RAM usage and bandwidth, so they can more easily optimize their application.

### Implementation:-

**Step 1 :-**

Firstly we need two policies attached to an IAM role which we will assign to EC2 instances so that the logs from the instances can be pushed to the CloudWatch.

There is a need of an CloudWatch agent which will do the task to push logs onto the CloudWatch.An agent-configuration file is necessary which we can store in our S3 bucket and at the time of launching an instance we will use that agent-configuration file.

The two necessary policies are :-

### Policy 1 :-

This policy will allow your EC2 instance to access the agent-configuration file stored in your S3 bucket.Here you would give the name of your S3 bucket.

{    
   “Version”:”2012-10-17″,  
   “Statement”:[    
      {    
         “Effect”:”Allow”,  
         “Action”:[    
            “logs:\*”,  
            “s3:GetObject”  
         ],  
         “Resource”:[    
            “arn:aws:logs:\*:\*:\*”,  
            “arn:aws:s3:::your\_bucket\_name/\*”  
         ]  
      }  
   ]  
}

### Policy 2 :-

This policy will allow your EC2 instance to push the log file stored in your CloudWatch.So here I am assigning all the permissions to my EC2 instance so that it can create log group,log stream and other necessary files.

{

“Version”:”2012-10-17″,”Statement”:[

{    
         “Effect”:”Allow”,  
         “Action”:[    
            “logs:\*”  
         ],  
         “Resource”:[    
            “arn:aws:logs:\*:\*:\*”  
         ]  
      }  
   ]  
}

So create an IAM role and assign two policies to it and at the time of launching EC2 instance you will be assigning this role to your EC2 instance.

**Step 2 :- Understanding Agent-Configuration File (stored on S3)**

Now, create an agent configuration file Cloudwatch\_agent\_conf  and paste the content of  following file  and edit it accordingly and upload it to your S3 bucket.

[general]  
state\_file = /var/awslogs/state/agent-state

[/var/log/syslog]  
file = /var/log/nginx/access.log  
log\_group\_name = nginx\_server  
log\_stream\_name = nginx\_access\_logs  
datetime\_format = %b %d %H:%M:%S

### Parameters in agent-configuration file ****:-****

**1.file :-**The file specifies the file in which your actual logs are stored on your EC2 instances. This is the log file whose content you want to push on CloudWatch logs . I want to push my nginx access logs onto the CloudWatch so I am specifiying the path of nginx access log file.

**2.log\_group\_name :-**It refers to the destination log group. A log group will be created automatically if no log group exists in your CloudWatch.

**3.log\_stream\_name :-**It refers to the destination log stream.A log stream can be {instance\_id}, {hostname}, {ip\_address} or a combination of these.

**4.datetime\_format :-**It specifies how the timestamp is extracted from logs.

%b specifies month (Jan,Feb..)

%d specifies day of month (01,02..)

%H specifies Hour (24-hour clock)

%M specifies Minutes (01,02..59)

%S specifies Seconds (01,02..59)

**Step 3 :-**Now, we need a script which we can pass in the User data at the time of launching an EC2 instance which will make all the configurations for CloudWatch agent to push logs to the CloudWatch.

#!/bin/bash  
wget https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/awslogs-agent-setup.py

chmod +x ./awslogs-agent-setup.py

 ./awslogs-agent-setup.py -n -r us-east-1 -c s3://your\_bucket\_name/Cloudwatch\_agent\_conf

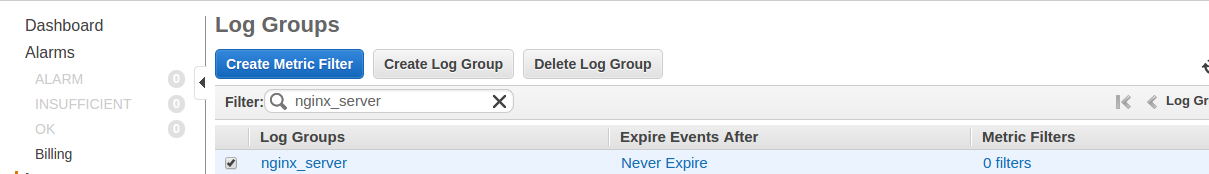
 sudo service awslogs restart

**Step 4 :-**We are almost done with the configuration . Now, launch a new EC2 instance, assign it the IAM role with two policies which you have created above and pass the above script as a userdata to the instance.

You can also append other packages’ installation steps after these lines in userdata script like installation of nginx or apache.

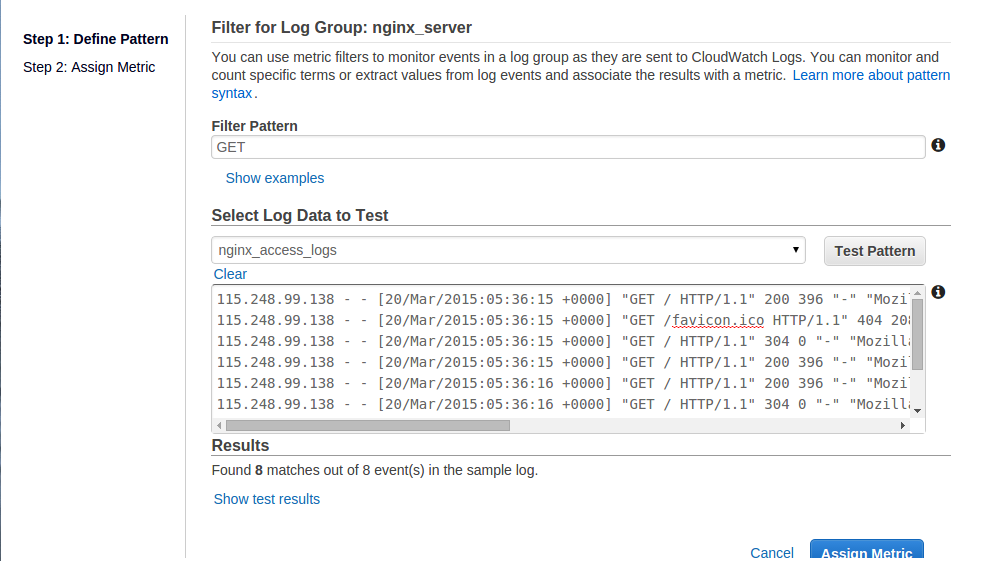
Now go to your AWS CloudWatch console.Go to “Logs” in Dashboard, you will be able to see the log group name which you mentioned in your agent-configuration file.

### SNS Integration on Logs to create alerts

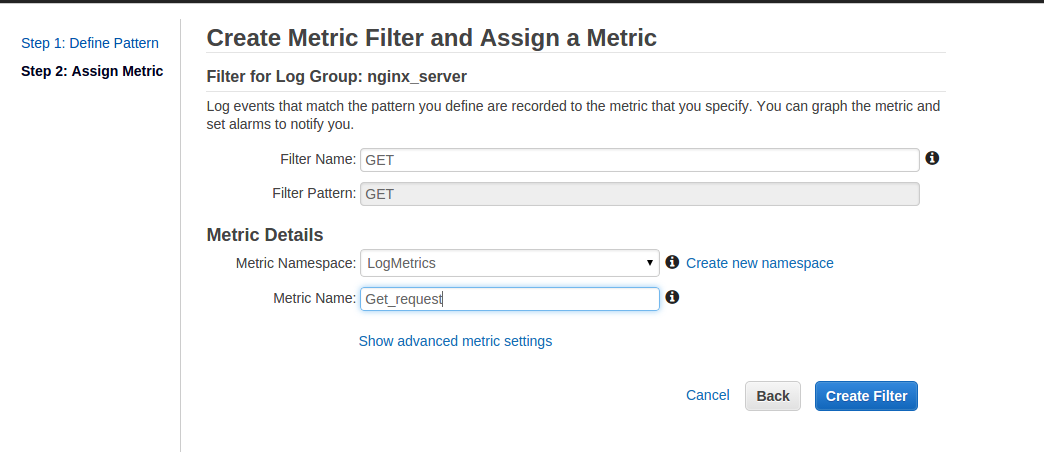
**Step 1 :-**Go to the Log Groups in your AWS CloudWatch console .Click on Create Metric Filter.

**Step 2 :-** Now, you can set pattern to be searched in logs.

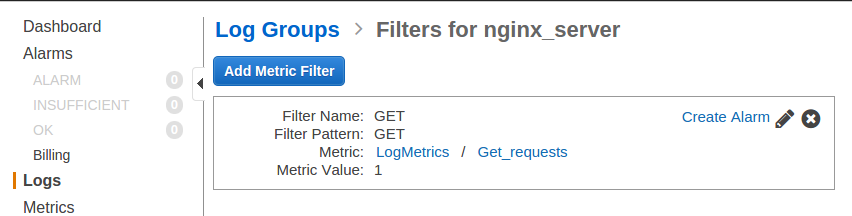
For Example, I want to create alarm according to my GET requests.Click on Assign Metric.



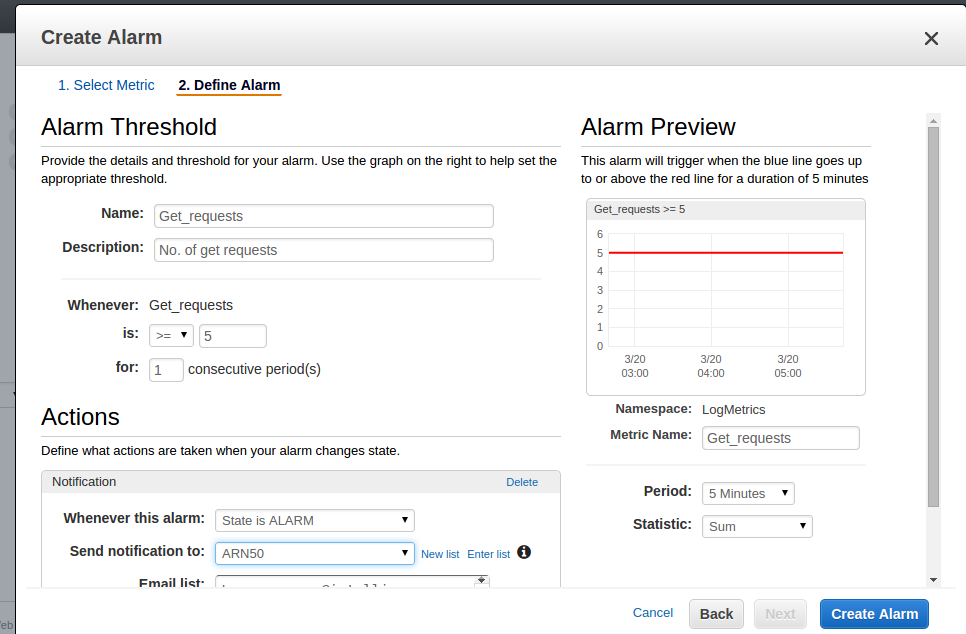
**Step 3 :-**Now, assign a metric to your metric filter.



**Step 4 :-**Your metric filter is created.Now click on Create Alarm .



**Step 5 :-**Here you can create alarm by setting thresholds and ARN for your SNS.



Finally your alarms are configured on your logs.You can check your alarms in CloudWatch alarms.