CloudWatch CLI

Contents

Overview

Amazon **CloudWatch** monitors your Amazon Web Services (AWS) resources and the applications you run on AWS in real-time. You can use CloudWatch to collect and track metrics, which are variables you can measure for your resources and applications.

The CloudWatch home page automatically displays metrics about every AWS service you use. You can additionally create custom dashboards to display metrics about your custom applications and display custom collections of metrics that you choose.

You can create alarms that watch metrics and send notifications or automatically make changes to the resources you are monitoring when a threshold is breached. For example, you can monitor the CPU usage and disk reads and writes of your Amazon EC2 instances and then use this data to determine whether you should launch additional instances to handle the increased load. You can also use this data to stop under-used instances to save money. With CloudWatch, you gain system-wide visibility into resource utilization, application performance, and operational health.

CloudWatch Key Concepts

Metrics

A metric is a 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. Metrics are generated by AWS products and users, they both publish metric data points into CloudWatch, this data is then used by CloudWatch to create statistics, these are retrievable by users as an ordered set of time-series data. Metrics exist only in the region in which they are created.

Namespaces

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

Alarms

You can use an alarm to automatically initiate actions on your behalf. An alarm watches a single metric over a specified time period, and performs one or more specified actions, based on the value of the metric relative to a threshold over time. The action is a notification sent to an Amazon SNS topic or an Auto Scaling policy. You can also add alarms to dashboards.

Alarms invoke actions for sustained state changes only. CloudWatch alarms will not invoke actions simply because they are in a particular state. The state must have changed and been maintained for a specified number of periods.

Tutorial

This tutorial will cover how to set up an alarm that triggers a notification whenever the CPU utilisation
exceeds 70%.

1. We need to make sure we have an instance to monitor.

```
# Run an EC2 instance
aws ec2 run-instances --image-id ami-08a2aed6e0a6f9c7d --count 1 --instance-type
t2.micro --key-name (your key-pair name)
```

Make sure you are launching this instance while your aws has been configured to eu-west-1 region.

Note down the instance id, as this will be required to defined our cloudwatch metric.

2. We need to now create a notification which will be used to notify a specific person when the metric
has been set to the ALARM STATE.

```
# Create Topic
aws sns create-topic --name qa-sample-alarm-topic
```

Note down the topic ARN.

```
# Create subscription
aws sns subscribe --topic-arn (your topic arn) --protocol email --notification-
endpoint (your email address)
```

This will output the subscription ARN and for the first time, this will say **"pending confirmation"**.
You will need to go to the email address used for the endpoint and make sure you subscribe.
Check your spam folder as well in case you can't find it.

3. Create the alarm that will monitor and alert the person subscribed to your topic.

```
# Create your alarm
aws cloudwatch put-metric-alarm --alarm-name qa-sample-cpu-alarm --alarm-
description "QA Sample Alarm when CPU exceeds 70 %" --metric-name CPUUtilization
--namespace AWS/EC2 --statistic Average --period 300 --threshold 70 --unit
Percent --comparison-operator GreaterThanThreshold --dimensions
"Name=InstanceId,Value=(your instance id)" --evaluation-periods 2 --alarm-
actions (your topic arn)
```

There will be no output, but you can navigate to your AWS CloudWatch Alarms overview, and you will see your new
alarm created.

4. Lets test out this out. SSH into the EC2 instance that is being monitored and set up as part of the alarm.

Remember, if you're unable to SSH, it means you have not opened the SSH port for your instance security group.

```
# Run a CPU stress test
sudo yum update -y && sudo amazon-linux-extras install epel -y
sudo stress --cpu 8 --timeout 600
```

We will be using the yum package manager, as the image id provided in this tutorial is the AMI for Amazon Linux 2.

After some time, your instance will go in the alarm state and you will be notified via email.

5. We need to create a role and attach the role to our EC2 instance. This role should have access to AWS CloudWatch.

```
# Create policy
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
cat << EOF > assume-policy.json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::${ACCOUNT_ID}:root",
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
```

```
# Create role on new policy
aws iam create-role --role-name ec2-cloudwatch-role --assume-role-policy-
document file://assume-policy.json
```

```
# Attach policy to role
aws iam attach-role-policy --role-name ec2-cloudwatch-role --policy-arn
arn:aws:iam::aws:policy/CloudWatchFullAccess
```

```
# Create instance profile
aws iam create-instance-profile --instance-profile-name EC2CW
```

```
# Add role to instance profile
aws iam add-role-to-instance-profile --instance-profile-name EC2CW --role-name
ec2-cloudwatch-role
```

```
# Attach role to instance
aws ec2 associate-iam-instance-profile --iam-instance-profile Name=EC2CW --
instance-id (your instance id)
```

5. Lets push custom metrics, as the default metrics are not so useful.

```
# Install dependencies
sudo yum update -y
sudo yum install -y perl-Switch perl-DateTime perl-Sys-Syslog perl-LWP-Protocol-
https perl-Digest-SHA.x86_64 unzip
```

```
# Download cloudwatch agent zip file
cd /home/ec2-user/
curl https://aws-
cloudwatch.s3.amazonaws.com/downloads/CloudWatchMonitoringScripts-1.2.2.zip -O
```

```
# Extract zip file
unzip CloudWatchMonitoringScripts-1.2.2.zip
rm -rf CloudWatchMonitoringScripts-1.2.2.zip
```

```
# Push custom metrics to CloudWatch
/home/ec2-user/aws-scripts-mon/mon-put-instance-data.pl --mem-util --mem-used --
mem-avail
```

This will return a successful message stating metric has been reported. Navigate to the CloudWatch metrics and you should see a Custom Namespace called "System/linux". If you go on in there
-> InstanceId, there should be a list of metric names. This will show a single data point as we only sent
one metric up.

6. Clean up your AWS environment.

```
# Terminate EC2 instance
aws ec2 terminate-instances --instance-ids i-0ac03a92eda9216e2
```

```
# Detaching IAM role
aws iam detach-role-policy --role-name ec2-cloudwatch-role --policy-arn
arn:aws:iam::aws:policy/CloudWatchFullAccess
```

```
# Removing Role from instance profile
aws iam remove-role-from-instance-profile --instance-profile-name EC2CW --role-
name ec2-cloudwatch-role
```

```
# Deleting IAM Role
aws iam delete-role --role-name ec2-cloudwatch-role
```

```
# Unsubscribe from topic
aws sns unsubscribe --subscription-arn (your subscription arn)
```

```
# Delete topic
aws sns delete-topic --topic-arn (your topic arn)
```

```
# Delete alarm
aws cloudwatch delete-alarms --alarm-names qa-sample-cpu-alarm
```

There is no option to deleting the new custom metric as of writing this tutorial. Wait a couple of weeks, and if no data is being pushed it will automatically delete the metric.

Exercises

1. Modify the command to constantly push these metrics up. Hint: CronJob.