CloudWatch Auto Alarms

CloudFormation template for Creation of Cloudwatch Alarms for EC2, RDS and ALB services.

aws partner network

Advanced
Consulting Partner

DevOps Competency
Well Architected
Solution Provider
Public Sector Partner
SaaS Partner

Submitted By

Rapyder Cloud Solutions Pvt Ltd

Bangalore | Delhi | Mumbai

Contents

About the Solution

This CloudFormation template deploys a solution which creates Cloudwatch alarms for EC2 instances including Autoscaling instances, RDS instances including Cluster instances and Application load balancer alarms.

The solution makes use of EventBridge Rules designated for triggering Lambda function which creates and deletes Cloudwatch alarms upon certain conditions met which will be discussed later in this doc.

Architecture Overview of templates

The present template launches the below mentioned AWS resources:

The template deploys the below components:

1. AWS SSM Parameter
   - Cloudwatch Metric Configuration for Linux instances.
   - Cloudwatch Metric Configuration for Windows instances.
2. AWS Lambda Function with Role
   - CloudwatchAutoAlarm Lambda function which creates EC2, RDS and ALB alarms
   - IAM Role for execution of Lambda function with required permissions.
3. AWS EventBridge Rule (Lambda Triggers)
   - Rule for EC2 instances in Running and Terminated State through CloudTrail Api.
   - Rule for RDS instances when Creation and Deletion Occurs through RDS Events.
   - Rule for ELB on Deletion of ELBv2 Load Balancers through CloudTrail Api.

Working and Features Involved

Instructions on deploying the solution are in the next section. This section provides details about the services involved and how they interact with each other.

EC2 Alarms:

EC2 instances Cloudwatch alarms are created when instances in launched and and comes in a "Running" state. This state is defined in EventBridge Rule which Triggers CloudwatchAutoAlarm Lambda Function when instances comes into the Running State. The process is similar for when a instance comes a "Terminated"

*Instance comes to **Running** State:*

- Whenever an EC2 instance comes into running state, the Lambda Function is triggered via Event Bridge.
- It checks if Alarm is already present for the instance, if it exists, it will do nothing and exit.
- Next, it installs the CloudWatch Agent via SSM Run command, (if already installed it will do nothing)
- As part of next step, it configures CloudWatch Agent with Document stored in SSM, and configure it to send Memory and disk metrics to CloudWatch.
- W.R.T disk utilization metrics, it will ignore the following File Systems (“sysfs", "devtmpfs", "tmpfs", "squashfs", "vfat”)
- Alarms will be created for ***CPU Utilization*, *Memory Utilization*, *Disk Utilization*, *Instance Status check*, *System Status check*.**

- Alarms for instances launched through AutoScaling Group will be created for only ***CPU Utilization and Memory Utilization metrics***. There will be only one alarm for mentioned metrics for a particular AutoScaling Group.
- Naming Convention for EC2 alarms is ***<ProjectName>#<InstanceName>_<InstanceId> :: <MetricName>#<Threshold>***

*Instance comes to **Terminated** State:*

- Again, Lambda will be triggered via Event Bridge.
- It will check if there are any Alarms for the instance which is created by this solution and will delete them.

RDS Alarms:

RDS DB instance Cloudwatch alarms are created when DB instance gets created successfully. This RDS Creation event is defined in EventBridge Rule which Triggers CloudwatchAutoAlarm Lambda Function when RDS DB instance in created. The process is similar for when a DB instance is deleted.

*RDS DB Instance is **Created**:*

- Whenever an RDS DB instance gets created, the Lambda Function is triggered via Event Bridge.
- It checks if Alarm is already present for RDS DB instance, if it exists, it will do nothing and exit.
- Alarms will be created for ***CPU Utilization***, ***Free Memory***, ***Free Storage***, ***DiskQueueDepth*** and ***Max DB Connections.***
- Alarms will be created only at DB instance level and not Cluster level, irrespective of whether the instance belongs to a Cluster.
- RDS DB instances which are part of a Cluster will not have ***Free Storage*** alarm.
- Naming Convention for RDS DB instance alarms is ***<ProjectName>#<DBIdentifier> :: <MetricName>#<Threshold>***

*RDS DB Instance is **Deleted**:*

- Again, Lambda will be triggered via Event Bridge.
- It will check if there are any Alarms for the RDS DB instance which is created by this solution and will delete them.

ALB Alarms:

ALB Cloudwatch alarms are created through ***Scan*** event which is discussed in detail in next section. The scan event takes in name of the Load Balancers and creates Cloudwatch alarms for the given ALB. The process is different for Deletion of ALB cloudwatch alarms which gets Triggered through EventBridge when an ALB is deleted and deletes all the alarms made by the solution.

*ALB alarms through **Scan** Event:*

- We need to pass name of the Application Load Balancers in the scan which we want to create Cloudwatch Alarm for. This is the manual scan event which is discussed in the next section.
- It checks if Alarm is already present for ALB , if it exists, it will do nothing and exit.
- Alarms will be created for ***TargetResponseTime, UnhealthyHostCount, HttpCodeTarget3xxCount***, ***HttpCodeTarget4xxCount*** and ***HttpCodeTarget5xxCount.***

- Alarms for ***UnhealthyHostCount*** will be created per Target Group, i.e. , if ALB has 2 Target groups, there will be 2 alarms created for UnhealthyHostCount per Target Group.
- Naming Convention for RDS DB instance alarms is ***<ProjectName>#<ALBName> :: <MetricName>#<Threshold>*** except for UnhealthyHostCount metric which has Naming Convention ***<ProjectName>#<ALBName> :: <MetricName>_<TargetGroupName>#<Threshold>***

*ALB is **Deleted**:*

- Lambda Function will be triggered via Event Bridge.
- It will check if there are any Alarms for the ALB which is created by this solution and will delete them.

Scan Event:

```
{
  "action": "scan",
  "instances": [
    "i-123xxxxxxx",
    "i-456xxxxxxx"
  ],
  "rds": [
    "db-instance-identifier-2",
    "db-instance-identifier-2"
  ],
  "alb": [
    "alb-name-1",
    "alb-name-2"
  ],

}
```

Scan Event is a custom event for creating Cloudwatch alarms for existing EC2 instances, RDS DB instances and ALB. You can refer to the above Json event in which the field **"action"** and value **"scan"** are necessary for triggering the lambda. The other fields take in multiple values inside a list. Field **"Instances"** takes in instance-ids for EC2 alarm, field **"rds"** takes in db-instance-identifier for RDS DB instance alarms and field **"alb"** takes in Application load balancer name for ALB alarms.

Pre-requisites and Limitations

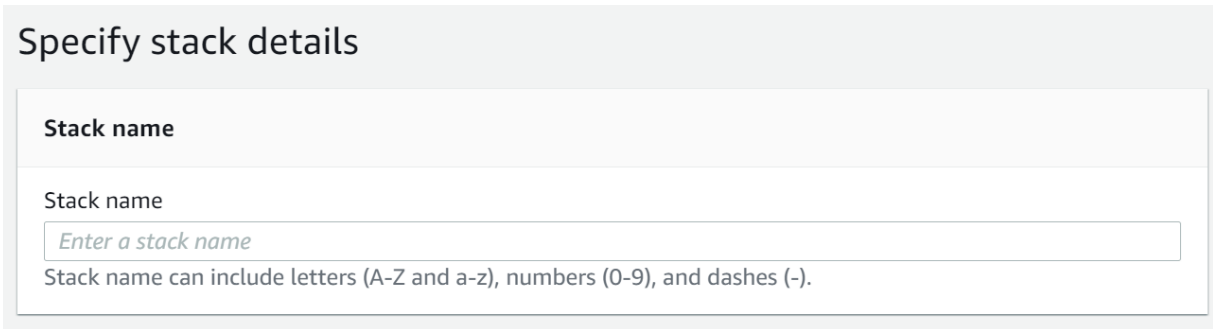
1. After CFT is launched Enable EventBridge Triggers (This is disabled by default for accidental launches)
2. Run Test Event in Lambda Function for Scan Event for existing workload as defined in documentation.
3. For Custom AMI Creation, make sure you mention OS name as part of the description with no abbreviation. For example, if Linux/Unix type of OS, please mention Ubuntu or Amazon Linux, etc. as part of description.

4. CW metric configuration for alarm creation would be updated while running this automation. If we want the existing configuration for CW logs, Alarms creation should be disabled and alarms should be made manually for respective instances or autoscaling group instances.
5. If EC2 instance is running under Autoscaling Group, the solution will aggregate average the Memory, CPU metrics and push it to CloudWatch. For ASG, it will not create Disk Alarms.
6. For Autoscaling Group instances, it will not automatically delete the alarm.
7. For EC2 instances, OS Supported: Amazon Linux, Ubuntu, RHEL, Centos, SUSE Linux, Windows.
8. For Scan Event, you can only run one service per event, i.e., you can create alarms for either ec2 instance or rds db instance or alb.
9. For creating Disk Alarms for additional volumes of ec2 instance, you need to delete the existing alarms present for the ec2 instance if any and then create alarm for the ec2 instance using Scan Event.

Deploying the Solution

This solution is deployed using CloudFormation Template with following steps:

1. In **Stack Name,** enter a name for your stack according to the mentioned pattern. This should be done for each template separately.


Specify stack details

Stack name

Stack name

Enter a stack name

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

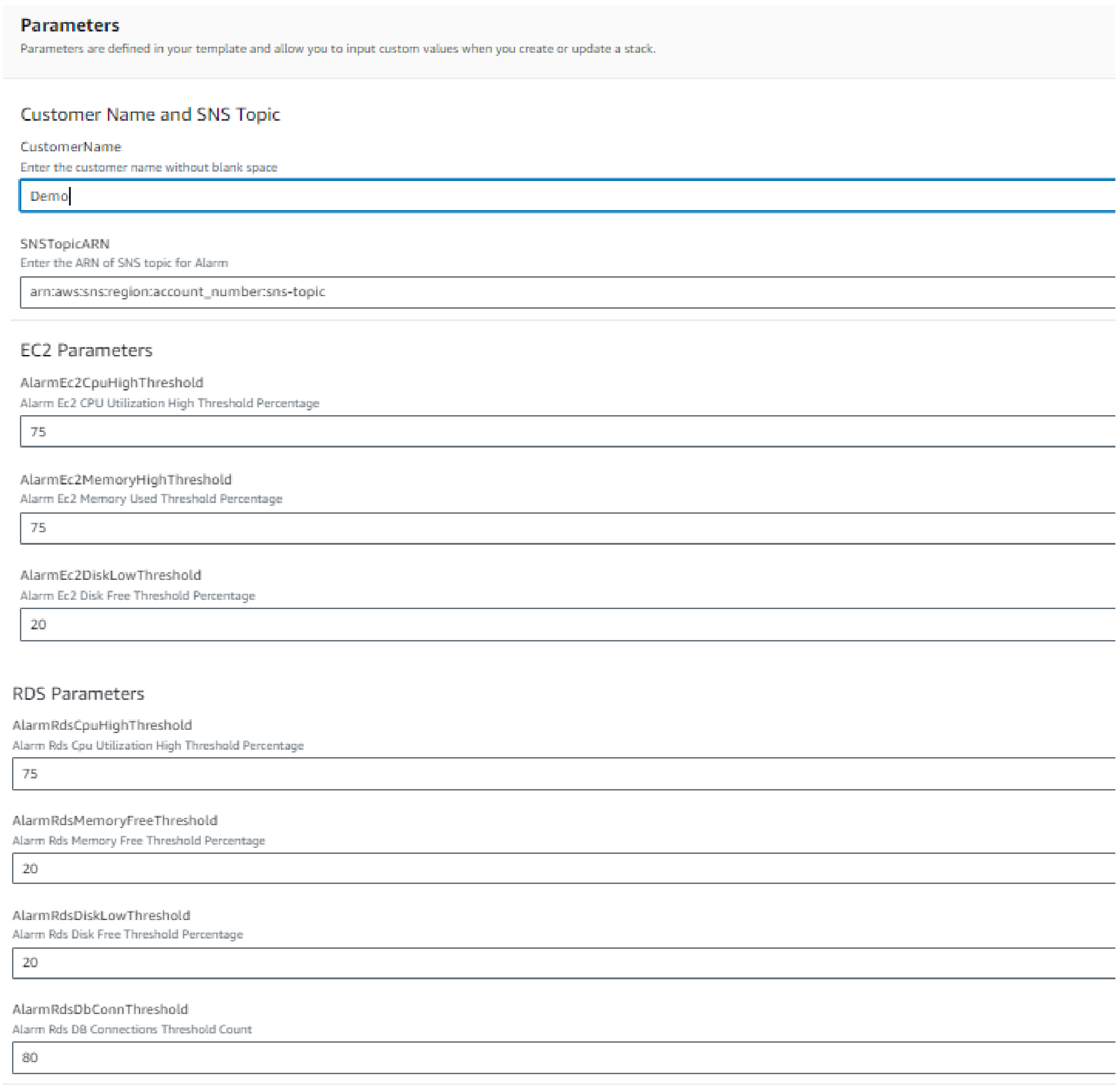
Below we mention the **template details and parameters** for each template that is used for the delivery acceleration:

Parameters:

The below image shows the Parameters page for the CloudFormation.

1. CustomerName: Enter the customer name without blank space
2. SNSTopicARN: Enter the ARN of SNS topic for Alarm
3. AlarmEc2MemoryHighThreshold: Alarm Ec2 Memory Used Threshold Percentage
4. AlarmEc2CpuHighThreshold: Alarm Ec2 CPU Utilization High Threshold Percentage
5. AlarmEc2DiskLowThreshold: Alarm Ec2 Disk Free Threshold Percentage
6. AlarmRdsCpuHighThreshold: Alarm Rds Cpu Utilization High Threshold Percentage
7. AlarmRdsMemoryFreeThreshold: Alarm Rds Memory Free Threshold Percentage

8. AlarmRdsDiskLowThreshold: Alarm Rds Disk Free Threshold Percentage
9. AlarmRdsDbConnThreshold: Alarm Rds DB Connections Threshold Count
10. AlarmAlbTargetResTimeThreshold: Alarm Alb Target Response Threshold Time in Seconds
11. AlarmAlbUnhealthyHostThreshold: Alarm Alb Unhealthy Host Threshold Count
12. AlarmAlbHttpTarget5xxThreshold: Alarm Alb Http Target 5xx Threshold Count
13. AlarmAlbHttpTarget3xxThreshold: Alarm Alb Http Target 3xx Threshold Count
14. AlarmAlbHttpTarget4xxThreshold: Alarm Alb Http Target 4xx Threshold Count


Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

Customer Name and SNS Topic

CustomerName
Enter the customer name without blank space
Demo

SNSTopicARN
Enter the ARN of SNS topic for Alarm
arn:aws:sns:region:account_number:sns-topic

EC2 Parameters

AlarmEc2CpuHighThreshold
Alarm Ec2 CPU Utilization High Threshold Percentage
75

AlarmEc2MemoryHighThreshold
Alarm Ec2 Memory Used Threshold Percentage
75

AlarmEc2DiskLowThreshold
Alarm Ec2 Disk Free Threshold Percentage
20

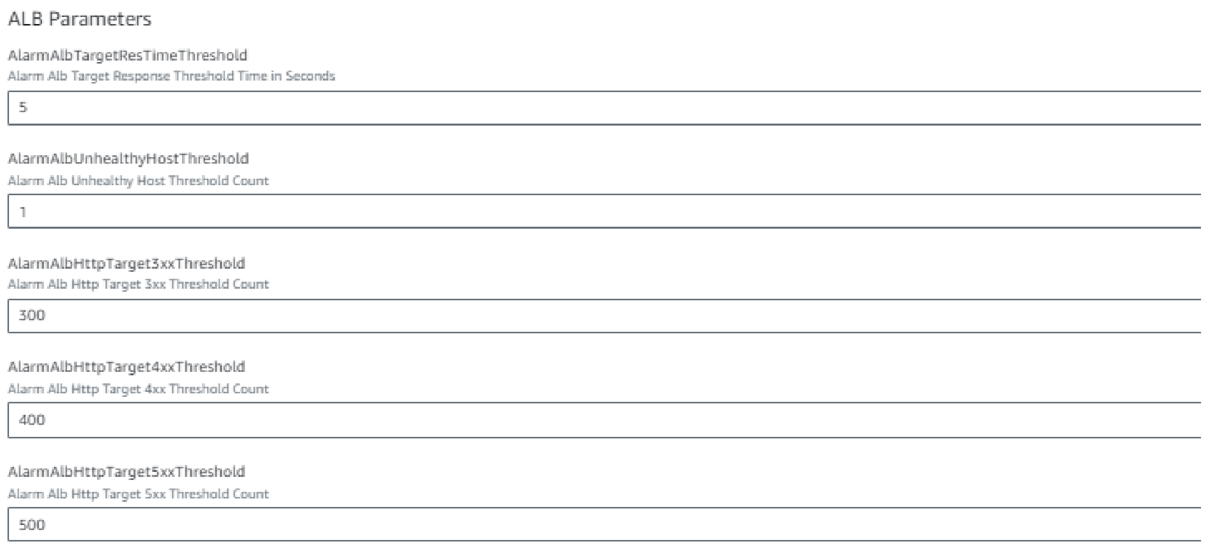
RDS Parameters

AlarmRdsCpuHighThreshold
Alarm Rds Cpu Utilization High Threshold Percentage
75

AlarmRdsMemoryFreeThreshold
Alarm Rds Memory Free Threshold Percentage
20

AlarmRdsDiskLowThreshold
Alarm Rds Disk Free Threshold Percentage
20

AlarmRdsDbConnThreshold
Alarm Rds DB Connections Threshold Count
80



ALB Parameters

AlarmAlbTargetResTimeThreshold
Alarm Alb Target Response Threshold Time in Seconds
5

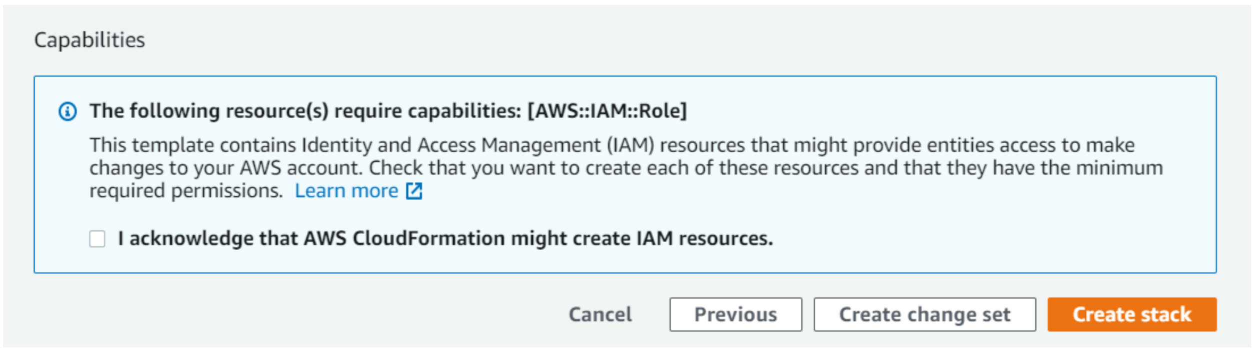
AlarmAlbUnhealthyHostThreshold
Alarm Alb Unhealthy Host Threshold Count
1

AlarmAlbHttpTarget3xxThreshold
Alarm Alb Http Target 3xx Threshold Count
300

AlarmAlbHttpTarget4xxThreshold
Alarm Alb Http Target 4xx Threshold Count
400

AlarmAlbHttpTarget5xxThreshold
Alarm Alb Http Target 5xx Threshold Count
500

Press "Next" and let the default values be the same. When on the last page, tick mark the last section to acknowledge the IAM role creation by CloudFormation. Click the "Create Stack" to begin stack creation.



Capabilities

The following resource(s) require capabilities: [AWS::IAM::Role]

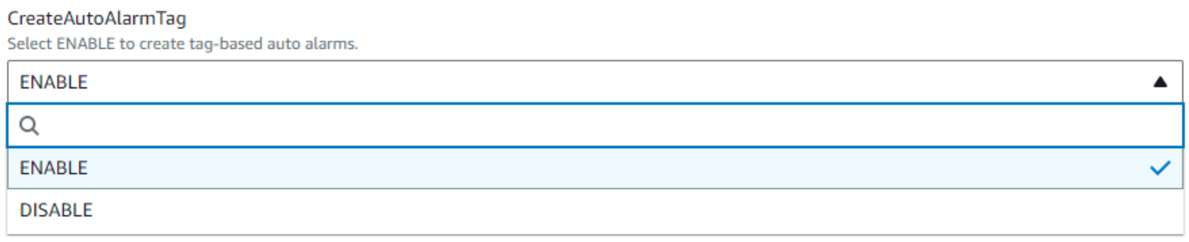
This template contains Identity and Access Management (IAM) resources that might provide entities access to make changes to your AWS account. Check that you want to create each of these resources and that they have the minimum required permissions. Learn more

☐ **I acknowledge that AWS CloudFormation might create IAM resources.**

Solution Updates

version 1.5:

1. New Feature added for EC2 alarms which enables whether tag-based alarm will be created or not. For this, there is a new parameter update while lauching cloudformation template named as **CreateAutoAlarmTag** having values **[ENABLE, DISABLE]**. Selecting Enable will create alarms only for instance which are tagged.



CreateAutoAlarmTag
Select ENABLE to create tag-based auto alarms.
ENABLE

2. Selecting Disable will create alarms for every EC2 instance which is launched irrespective of whether they are tagged or not.

3. Alarms will only get created for Instance tagged with Key as **Rapyder_Auto_Alarm** and Value as **On.**
4. If the value of **CreateAutoAlarmTag** was wrongly selected and CFT was launched, you can simply update the CFT stack with the same template and override the parameter value.