100 Days of DevOps — Day 2 -Introduction to Simple Notification Service(SNS)

*Welcome to Day 2 of 100 Days of DevOps, Let extend the journey of DevOps Monitoring and Alerting*[*https://medium.com/@devopslearning/100-days-of-devops-day-1-introduction-to-cloudwatch-metrics-b04be36307a8*](https://medium.com/@devopslearning/100-days-of-devops-day-1-introduction-to-cloudwatch-metrics-b04be36307a8)*with alerting concept Simple Notification Service(SNS).*

**Problem Statement**

* *To send out a notification via Email, SMS.. when an event occurs.*

**Solution**

*This can be achieved via in one of the three ways*

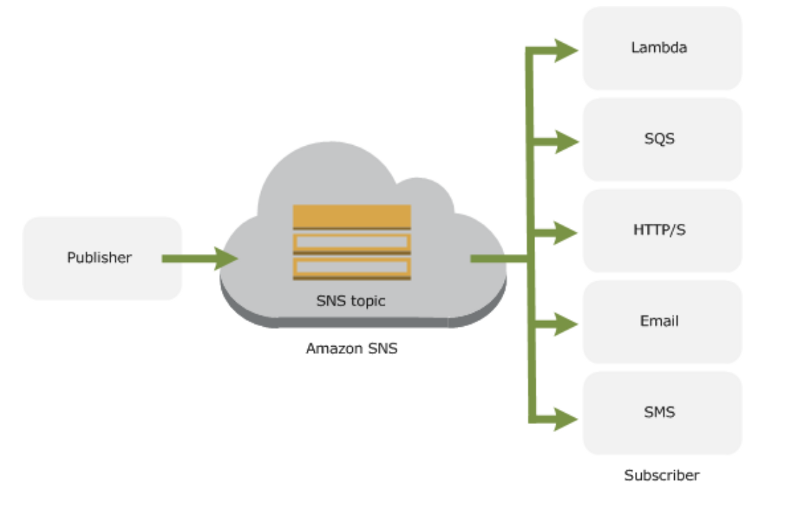
* *AWS Console*
* *AWS CLI*
* *Terraform*

*NOTE: This is not the complete list and there are more ways to achieve the same*

What is SNS?

*As per official documentation*

AWS SNS is a web service that coordinates and manages the delivery or sending of messages to subscribing endpoints or clients.



*We already saw this in action with CloudWatch(high CPU utilization or System/Instance Status Check) when the certain event occurs and SNS is used to send a notification. CloudWatch in combination with SNS creates a full monitoring solution with notifies the administrator in case of any environment issue(high CPU, Downtime…).*

*Reference*

[What is Amazon Simple Notification Service? - Amazon Simple Notification Service  
Describes the Amazon SNS web service that coordinates and manages the delivery or sending of messages to subscribing…docs.aws.amazon.com](https://docs.aws.amazon.com/sns/latest/dg/welcome.html)

***SNS has three major components***

**Publisher**

* *The entity that triggers the sending of a message(eg: CloudWatch Alarm, Any application or S3 events)*

**Topic**

* *Object to which you publish your message****(≤256KB)***
* *Subscriber subscribe to the topic to receive the message*
* *Soft limit of 10 million subscribers*

**Subscriber**

*An endpoint to a message is sent. Message are simultaneously pushed to the subscriber*

* *As you can see it follows the publish-subscribe****(pub-sub)****messaging paradigm with notification being delivered to the client using a push mechanism that eliminates the need to periodically check or poll for new information and updates.*
* *To prevent the message from being lost, all messages published to Amazon SNS are stored redundantly across multiple Availability Zones.*

**E***nough of theory, let see SNS in action*

**Using the AWS Console**

**Step 1: Create a topic**

* *In the*[*Amazon SNS console*](https://console.aws.amazon.com/sns/v2/home)*, choose****Create topic****.*
* *Create a topic*

**Step2: Subscribe to a Topic**

* *Choose to****Create a subscription****.*
* *The****Create Subscription****dialog box appears.*
* *Go to your email and confirm subscription*

**Step3: Publish to the topic**

* *Choose the****Publish to the topic****button.*
* *The****Publish a Message****page appears.*

**Using AWS CLI**

* ***Create a Topic***

*$ aws sns create-topic --name "my-demo-sns-topic"*

*{*

*"TopicArn": "arn:aws:sns:us-west-2:1234556667:my-demo-sns-topic"*

*}*

* ***Subscribe to a Topic***

*$ aws sns subscribe --topic-arn arn:aws:sns:us-west-2:123456667:my-demo-sns-topic --protocol email --notification-endpoint test@gmail.com*

*{*

*"SubscriptionArn": "pending confirmation"*

*}*

* ***Publish to a Topic***

*$ aws sns publish --topic-arn arn:aws:sns:us-west-2:1234567:my-demo-sns-topic --message "hello from sns"*

*{*

*"MessageId": "d651b7d5-2d66-58c8-abe4-e30822a3aa3e"*

*}*

* ***To list all the subscriptions***

*$ aws sns list-subscriptions*

*{*

*"Subscriptions": [*

*{*

*"Owner": "1234567889",*

*"Endpoint": "test@gmail.com",*

*"Protocol": "email",*

*"TopicArn": "arn:aws:sns:us-west-2:1234567788:HighCPUUtilization",*

*"SubscriptionArn": "arn:aws:sns:us-west-2:1234567788:HighCPUUtilization:a28e2be8-40cd-4f8b-83d9-33b2c858749d"*

*},*

* ***Unsubscribe from a Topic***

*aws sns unsubscribe --subscription-arn arn:aws:sns:us-west-2:1234567899:my-demo-sns-topic:f28124be-850b-4a2e-8d3e-a3dc4f7cca1a*

* ***Delete a topic***

*$ aws sns delete-topic --topic-arn arn:aws:sns:us-west-2:1234567788:my-demo-sns-topic*

* ***List a topic***

*$ aws sns list-topics*

*{*

*"Topics": [*

*{*

*"TopicArn": "arn:aws:sns:us-west-2:123333345555:mydemosnstopic"*

*}*

*]*

*}*

[100daysofdevops/100daysofdevops  
Contribute to 100daysofdevops/100daysofdevops development by creating an account on GitHub.github.com](https://github.com/100daysofdevops/100daysofdevops/blob/master/sns_via_aws_cli)

**Using Terraform**

***# main.tf***

*resource "aws\_sns\_topic" "alarm" {  
 name = "alarms-topic"  
  
 delivery\_policy = <<EOF  
{  
 "http": {  
 "defaultHealthyRetryPolicy": {  
 "minDelayTarget": 20,  
 "maxDelayTarget": 20,  
 "numRetries": 3,  
 "numMaxDelayRetries": 0,  
 "numNoDelayRetries": 0,  
 "numMinDelayRetries": 0,  
 "backoffFunction": "linear"  
 },  
 "disableSubscriptionOverrides": false,  
 "defaultThrottlePolicy": {  
 "maxReceivesPerSecond": 1  
 }  
 }  
}  
EOF  
  
 provisioner "local-exec" {  
 command = "aws sns subscribe --topic-arn ${self.arn} --protocol email --notification-endpoint ${var.alarms\_email}"  
 }  
}*

***# variables.tf***

*variable "alarms\_email" {  
 default = "laprashant@gmail.com"  
}*

***# outputs.tf***

*output "sns\_topic" {  
 value = "${aws\_sns\_topic.alarm.arn}"  
}*

*GitHub link*

[100daysofdevops/100daysofdevops  
Contribute to 100daysofdevops/100daysofdevops development by creating an account on GitHub.github.com](https://github.com/100daysofdevops/100daysofdevops/tree/master/monitoring_and_alerting/sns_alert)