

Kops Nodes Alerts

find a solution to automatic alert (by slack or email) when our cluster is down (either master is all down or nodes is all down) you can use cloudwatch, or other free/open-source solution.

so the issue here is "we want to be alerted so we can take manual action when it's completely broken", you can think freely how we can solve the issue

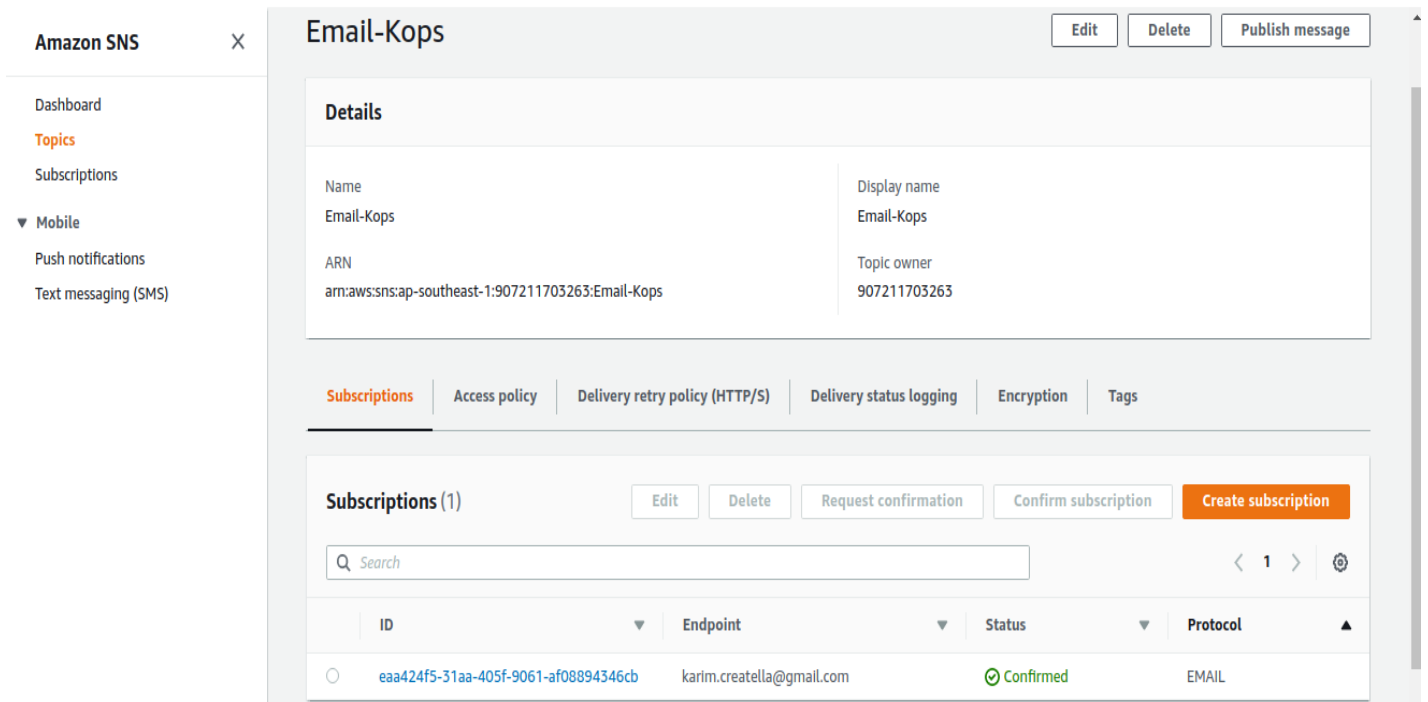
Solution

- 1.Using Cloudwatch to send Email Alert.
- 2.Using Cloudwatch with Slack to send Notification in Channel.

In Two options we will use “SNS” aws notification service to create topic and subscription, when cloudwatch detect alert “Terminate - Launch instances in autoscaling group of kops Cluster”.

1.Using Cloudwatch to send Email Alert.

1. Access AWS console and Create topic called “Email-Kops” and subscription type “Email” to send mail for example to my Email “karim.creatella@gmail.com”



Note : you will ask for confirmation in your mail to allow aws send mails “Notifications” to your mail, so you need to confirm that.

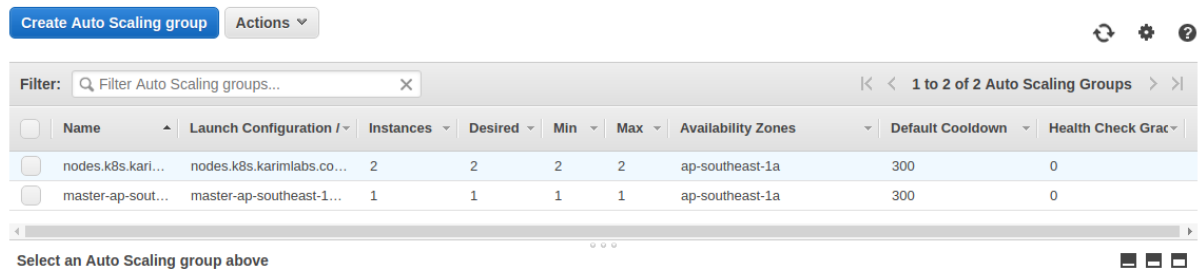
2.Create Notification for AWS Kops cluster Instances groups.

In AWS Console Open Autoscaling group and check the instances groups.

In my case i used this Repo to deploy Kops

<https://github.com/karimfadi/vagrant-kops-ansible> that consist of two instances groups “1 Master have one instances” “1 Nodes have 2 Nodes”

If you use the same example you will find the instances group as the following:



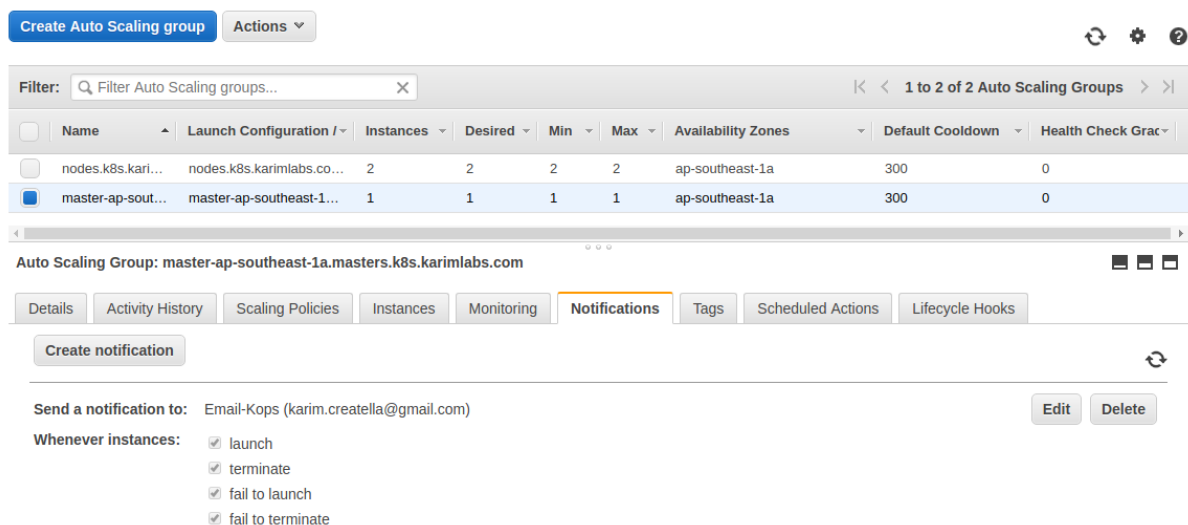
Filter:

1 to 2 of 2 Auto Scaling Groups

<input type="checkbox"/>	Name	Launch Configuration /	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health Check Grac
<input type="checkbox"/>	nodes.k8s.kari...	nodes.k8s.karimlabs.co...	2	2	2	2	ap-southeast-1a	300	0
<input type="checkbox"/>	master-ap-sout...	master-ap-southeast-1...	1	1	1	1	ap-southeast-1a	300	0

Select an Auto Scaling group above

In Both of them “Master and Nodes” you will click to instances group, chose “Notifications” and mark to all options.



Auto Scaling Group: master-ap-south-east-1a.karimlabs.com

Details Activity History Scaling Policies Instances Monitoring Notifications Tags Scheduled Actions Lifecycle Hooks

Create notification

Send a notification to: Email-Kops (karim.creatella@gmail.com) Edit Delete

Whenever instances:

- ☒ launch
- ☒ terminate
- ☒ fail to launch
- ☒ fail to terminate



Note : you will receive Email that notify this action will send to you in the future ignore it.

Now we will do a simple test to see if it's working or not.

We will terminate one of Kops EC2 nodes and see what will happen.

3. Test Cloudwatch Email Notification in case any Node Terminated or Launched.

When i terminated the instance i received mail like the following:



Auto Scaling: termination for group "nodes.k8s.karimlabs.com"  

Inbox x

Email-Kops no-reply@sns.amazonaws.com [via](#) amazonses.com 1:24 AM (47 minutes ago) ☆ ↶ ⋮
to me ▾

Service: AWS Auto Scaling
Time: 2019-07-05T23:24:39.893Z
RequestId: cbc5af11-d0ca-431f-d94f-c0f5b49b5e30
Event: autoscaling: [EC2_INSTANCE_TERMINATE](#)
AccountId: 907211703263
AutoScalingGroupName: [nodes.k8s.karimlabs.com](#)
AutoScalingGroupARN: arn:aws:autoscaling:ap-southeast-1:907211703263:autoScalingGroup:cd6e8d5-5f9a-46d9-83b4-ec759ddb21e2:autoScalingGroupName/[nodes.k8s.karimlabs.com](#)
ActivityId: cbc5af11-d0ca-431f-d94f-c0f5b49b5e30
Description: Terminating EC2 instance: i-0d7b0bb5864e44443
Cause: At 2019-07-05T23:24:37Z an instance was taken out of service in response to a EC2 health check indicating it has been terminated or stopped.
StartTime: 2019-07-05T23:24:37.904Z
EndTime: 2019-07-05T23:24:39.893Z
StatusCode:InProgress
StatusMessage:
Progress: 50
EC2InstanceId: i-0d7b0bb5864e44443
Details: {"Subnet ID":"subnet-01e46cfa124ccb896","Availability Zone":"ap-southeast-1a"}

Kops detected that and start to launch a new instance and i received a new mail for launching the new instances like the following:

Auto Scaling: launch for group "nodes.k8s.karimlabs.com"  

Inbox x

Email-Kops no-reply@sns.amazonaws.com [via](#) amazonses.com 1:26 AM (46 minutes ago) ☆ ↶ ⋮
to me ▾

Service: AWS Auto Scaling
Time: 2019-07-05T23:26:08.665Z
RequestId: 68e5af11-d296-c459-7a1b-b73266169971
Event: autoscaling: [EC2_INSTANCE_LAUNCH](#)
AccountId: 907211703263
AutoScalingGroupName: [nodes.k8s.karimlabs.com](#)
AutoScalingGroupARN: arn:aws:autoscaling:ap-southeast-1:907211703263:autoScalingGroup:cd6e8d5-5f9a-46d9-83b4-ec759ddb21e2:autoScalingGroupName/[nodes.k8s.karimlabs.com](#)
ActivityId: 68e5af11-d296-c459-7a1b-b73266169971
Description: Launching a new EC2 instance: i-08a0062604a723721
Cause: At 2019-07-05T23:25:07Z a difference between desired and actual capacity changing the desired capacity, increasing the capacity from 1 to 2.
StartTime: 2019-07-05T23:25:07.377Z
EndTime: 2019-07-05T23:26:08.665Z
StatusCode:InProgress
StatusMessage:
Progress: 50
EC2InstanceId: i-08a0062604a723721
Details: {"Subnet ID":"subnet-01e46cfa124ccb896","Availability Zone":"ap-southeast-1a"}

2.Using Cloudwatch with Slack to send Notification in Channel.

In This Case we will create SNS with Slack through Lambda Function.

I used the following Resource for that:

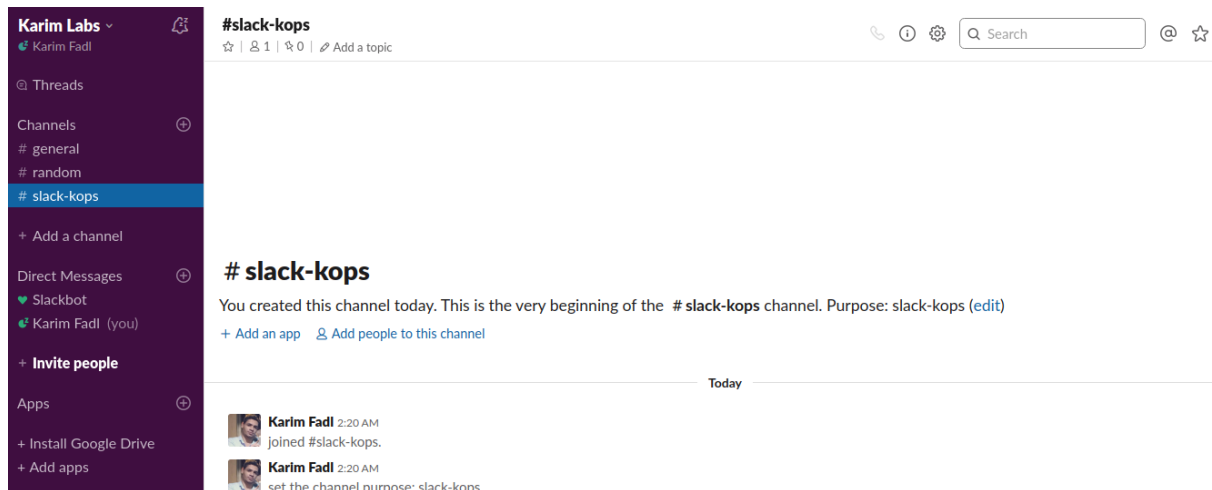
https://dev.to/alex_barashkov/how-to-send-aws-cloudwatch-alarms-to-slack-596e

Let's Follow the steps:

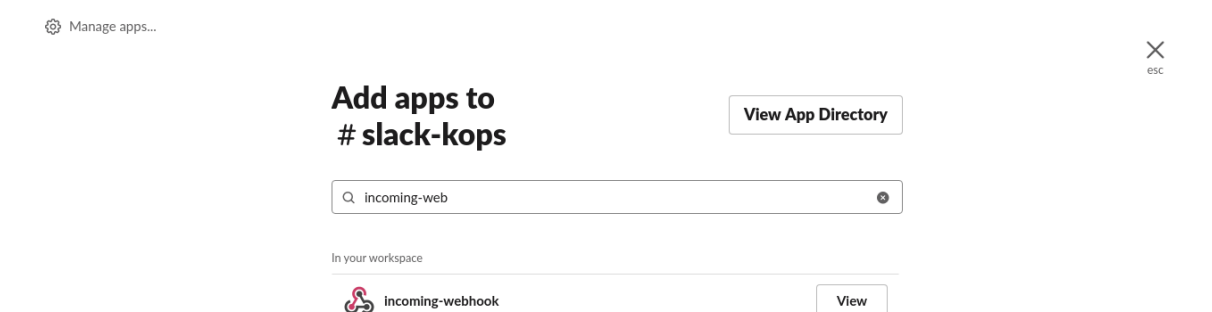
- 1.Create Your Slack channel That will receive the Notifications.
- 2.Create AWS Role To execute Lambda Function.
- 3.Create Lambda Functions to be Part from SNS Topic and Subscription.
- 4.Test The Slack Notification if you terminated an instance.

1. Create Your Slack channel That will receive the Notifications.


In My Slack workspace i created a new channel called “slack-kops”, to receive all kops notification on it.



Next step i will click to “+add an app” to add “incoming webhook” app that will integrate with AWS SNS with token.



After you installed it you will start to Configure it by adding the app to our slack channel “slack-kops”




Incoming WebHooks

Send data into Slack in real-time.

Incoming Webhooks are a simple way to post messages from external sources into Slack. They make use of normal HTTP requests with a JSON payload, which includes the message and a few other optional details described later.

[Message Attachments](#) can also be used in Incoming Webhooks to display richly-formatted messages that stand out from regular chat messages.

**New to Slack integrations?**

Check out our [Getting Started](#) guide to familiarize yourself with the most common types of integrations, and tips to keep in mind while building your own. You can also [register as a developer](#) to let us know what you're working on, and to receive future updates to our APIs.

Post to Channel

Start by choosing a channel where your Incoming Webhook will post messages to.

slack-kops

or [create a new channel](#)

Add Incoming WebHooks integration

After that it's automatically generate a token for your integration you must save it for future steps.

Also you can change some configuration like change slack channel logo.

Post to Channel

Messages that are sent to the incoming webhook will be posted here.

slack-kops

or [create a new channel](#)

Webhook URL

Send your JSON payloads to this URL.
[Show setup instructions](#)

<https://hooks.slack.com/services/TK7BM4KPB/BKUTRPHD1/OqEcc24MIhfdCw>

[Copy URL](#) • [Regenerate](#)

Descriptive Label

Use this label to provide extra context in your list of integrations (optional).

Optional description of this integration


2.Create AWS Role To execute Lambda Function.


In this Step we create AWS custom role for Lambda function that will running to create SNS.


Through AWS console >> IAM >> Role >> chose AWS service >> Lambda.


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.

Chose “AWSLambdaBasicExecutionRole” Policy to attach to this Role.

Create role 1 2 3 4

▼ Attach permissions policies


Choose one or more policies to attach to your new role.

Create policy ↺

Filter policies ▼

Q basic

Showing 1 result

	Policy name ▼	Used as	Description
<input checked="" type="checkbox"/>	 AWSLambdaBasicExecutionRole	None	Provides write permissions to CloudWatc...

Add Role Name for example “cloudwatch-slack”

Review

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

Role name*

cloudwatch-slack

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

Role description

Allows Lambda functions to call AWS services on your behalf.

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

Trusted entities

AWS service: lambda.amazonaws.com

Policies



AWSLambdaBasicExecutionRole [↗](#)

Permissions boundary

Permissions boundary is not set

* Required

[Cancel](#)

[Previous](#)

[Create role](#)

In Summary section Save the ARN of this Role we will use it in the next steps.

[Roles](#) > cloudwatch-slack

Summary

[Delete role](#)

Role ARN

arn:aws:iam::907211703263:role/cloudwatch-slack [↗](#)

Role description

Allows Lambda functions to call AWS services on your behalf. | [Edit](#)

Instance Profile ARNs



Path

/

Creation time

2019-07-06 02:36 UTC+0200

Maximum CLI/API session duration

1 hour [Edit](#)

Permissions

[Trust relationships](#)

[Tags](#)

[Access Advisor](#)

[Revoke sessions](#)

▼ Permissions policies (1 policy applied)

[Attach policies](#)

[+ Add inline policy](#)

Policy name ▼	Policy type ▼	
AWSLambdaBasicExecutionRole	AWS managed policy	✕

3.Create Lambda Functions to be Part from SNS Topic and Subscription.

To deploy the AWS Lambda function, you need to clone the Repo:

<https://github.com/assertible/lambda-cloudwatch-slack>

and have Node.js installed on your local machine.

Especially for that guide, we contributed significantly in the repository to get a better configuration process.

```
git clone git@github.com:assertible/lambda-cloudwatch-slack.git
cd lambda-cloudwatch-slack
cp .env.example .env
```

Open your .env file and fill in the environment variables.

Change The following Env :

UNENCRYPTED_HOOK_URL >> Slack URL WebHook

AWS_FUNCTION_NAME >> Choose any Name

AWS_REGION >> for example : ap-southeast-1

AWS_ROLE >> "arn:aws:iam::907211703263:role/cloudwatch-slack"

AWS_ACCESS_KEY_ID

AWS_SECRET_ACCESS_KEY

Now You can Run the script with two Steps:

```
npm install
```

```
npm run deploy
```

Now check the function that created, Open your AWS console in AWS service Lambda Function and see your function.

AWS Lambda ×

Dashboard
Applications
Functions
Layers

Lambda > Functions

Functions (1) Actions ▾ Create function

Filter by tags and attributes or search by keyword

	Function name ▾	Description	Runtime ▾	Code size ▾	Last modified ▾
<input type="radio"/>	slack-sns	Better Slack notifications for AWS CloudWatch	Node.js 8.10	6.2 MB	56 minutes ago

Now we can create a new SNS topic called for example “Slack-Kops” with “Lambda” type subscription.

Amazon SNS ×

Dashboard
Topics
Subscriptions
▼ Mobile
Push notifications
Text messaging (SMS)

Slack-Kops Edit Delete Publish message

Details

Name Slack-Kops	Display name Slack-Kops
ARN arn:aws:sns:ap-southeast-1:907211703263:Slack-Kops	Topic owner 907211703263

Subscriptions | Access policy | Delivery retry policy (HTTP/S) | Delivery status logging | Encryption | Tags

Subscriptions (1) Edit Delete Request confirmation Confirm subscription Create subscription

Search

	ID ▾	Endpoint ▾	Status ▾	Protocol ▲
<input type="radio"/>	d329c4c5-63f6-4b5b-8e93-8cc501e1e220	arn:aws:lambda:ap-southeast-1:907211703263:function:slack-sns	Confirmed	LAMBDA

4. Test The Slack Notification if you terminated an instance.

Like we did in SNS of Email with Autoscaling, we open Autoscaling section in AWS Console and create a Notification “For Master and Nodes Instances Groups” but in this time to “Slack-Kops” Topic and see what will happen.

The screenshot shows the AWS Management Console interface for an Auto Scaling Group named 'master-ap-south-east-1a.masters.k8s.karimlabs.com'. The 'Notifications' tab is selected, displaying a 'Create notification' button and a configuration section. The configuration section shows the notification is sent to 'Slack-Kops (arn:aws:lambda:ap-southeast-1:907211703263:function:slack-sns)'. The 'Whenever instances:' section has four checked options: 'launch', 'terminate', 'fail to launch', and 'fail to terminate'.

Name	Launch Configuration	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health Check Grace
nodes.k8s.kari...	nodes.k8s.karimlabs.co...	2	2	2	2	ap-southeast-1a	300	0
master-ap-sout...	master-ap-southeast-1...	1	1	1	1	ap-southeast-1a	300	0

You can check your Slack Channel, you will find a Notification for that.

The screenshot shows a Slack channel named '#slack-kops' with two incoming-webhook notifications from 'AWS AutoScaling Notification'. The first notification is for the group 'master-ap-south-east-1a.masters.k8s.karimlabs.com' and the second is for the group 'nodes.k8s.karimlabs.com'. Both notifications include a 'Message', 'Description', 'Event' (autoscaling:TEST_NOTIFICATION), and 'Cause'.

#slack-kops

Today

set the channel purpose: slack-kops

Karim Fadl 3:26 AM
added an integration to this channel: incoming-webhook

Incoming-webhook APP 4:12 AM
AWS AutoScaling Notification

Message
Auto Scaling: test notification for group "master-ap-south-east-1a.masters.k8s.karimlabs.com"

Description

Event
autoscaling:TEST_NOTIFICATION

Cause

Today at 4:12 AM

AWS AutoScaling Notification

Message
Auto Scaling: test notification for group "nodes.k8s.karimlabs.com"

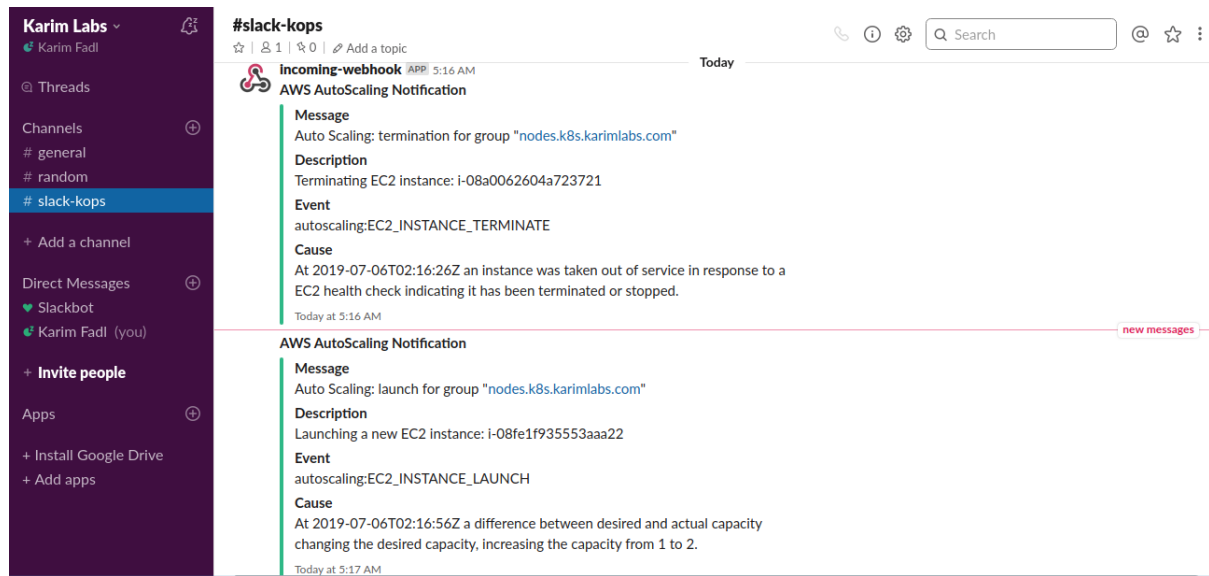
Description

Event
autoscaling:TEST_NOTIFICATION

Cause

Today at 4:12 AM

Let's Terminate one of nodes Instances and see if slack will receive that.



The screenshot shows a Slack interface with a sidebar on the left and a main channel view on the right. The sidebar includes a header for 'Karim Labs' with a dropdown arrow, a list of channels (#general, #random, #slack-kops), and options to add channels, direct messages, and apps. The main channel view is for '#slack-kops' and shows two messages from an 'incoming-webhook' app. The first message is an 'AWS AutoScaling Notification' with a green header bar. It contains a 'Message' section with the text 'Auto Scaling: termination for group "nodes.k8s.karimlabs.com"', a 'Description' section with 'Terminating EC2 instance: i-08a0062604a723721', an 'Event' section with 'autoscaling:EC2_INSTANCE_TERMINATE', and a 'Cause' section with 'At 2019-07-06T02:16:26Z an instance was taken out of service in response to a EC2 health check indicating it has been terminated or stopped.' The second message is also an 'AWS AutoScaling Notification' with a green header bar. It contains a 'Message' section with 'Auto Scaling: launch for group "nodes.k8s.karimlabs.com"', a 'Description' section with 'Launching a new EC2 instance: i-08fe1f935553aaa22', an 'Event' section with 'autoscaling:EC2_INSTANCE_LAUNCH', and a 'Cause' section with 'At 2019-07-06T02:16:56Z a difference between desired and actual capacity changing the desired capacity, increasing the capacity from 1 to 2.' A red 'new messages' badge is visible on the right side of the channel view.

We have it right Now Enjoy :)