**Assignment Task #3   
Demonstrate how you can alert a support team when instance or server RAM usage is greater than 60%, disk space usage is greater than 80%, or CPU usage greater than 70%.  
  
Your choice of using instancing, monitoring, and alert tool mechanisms like the following:  
  
\* EC2 Instance  
\* Cloudwatch  
\* SNS  
\* Any website endpoint  
\* Uptime Robot  
\* Slack channel \***

Create a webhook for slack as below

Graphical user interface, text, application, chat or text message

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Below is the webhook urls

https://hooks.slack.com/services/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

<https://testcloudwatchgroup.slack.com/services/B02JJPR04DD?added=1>

curl -X POST --data-urlencode "payload={\"channel\": \"#cloudwatch\", \"username\": \"webhookbot\", \"text\": \"This is posted to #cloudwatch and comes from a bot named webhookbot.\", \"icon\_emoji\": \":ghost:\"}" https://hooks.slack.com/services/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

Graphical user interface, text, application, email

Description automatically generated

Create SNS topic

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Ceate a lamda function

Graphical user interface, text, application

Description automatically generated

Test if the webhook is working are not with below code.

import json

from urllib.parse import urlencode

from urllib.request import Request, urlopen

def lambda\_handler(event, context):

webhook\_url = 'https://hooks.slack.com/services/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' # Set destination URL here

slack\_data = {

'channel': 'cloudwatch',

'text': 'this is a simple test'

}

request = Request(

webhook\_url,

data=json.dumps(slack\_data).encode(),

headers={'Content-Type': 'application/json'}

)

response = urlopen(request)

return {

'statusCode': response.getcode(),

'body': response.read().decode()

}

Actual code:  
  
import json

from urllib.parse import urlencode

from urllib.request import Request, urlopen

from datetime import datetime

import boto3

session = boto3.session.Session()

class CloudWatchAlarmParser:

def \_\_init\_\_(self, msg):

self.msg = msg

self.timestamp\_format = "%Y-%m-%dT%H:%M:%S.%f%z"

self.trigger = msg["Trigger"]

if self.msg['NewStateValue'] == "ALARM":

self.color = "danger"

elif self.msg['NewStateValue'] == "OK":

self.color = "good"

def \_\_url(self):

return ("https://console.aws.amazon.com/cloudwatch/home?"

+ urlencode({'region': session.region\_name})

+ "#alarmsV2:alarm/"

+ self.msg["AlarmName"]

)

def slack\_data(self):

\_message = {

'text': '<!here|here>', # add @here to message

'attachments': [

{

'title': ":aws: AWS CloudWatch Notification :alarm:",

'ts': datetime.strptime(

self.msg['StateChangeTime'],

self.timestamp\_format

).timestamp(),

'color': self.color,

'fields': [

{

"title": "Alarm Name",

"value": self.msg["AlarmName"],

"short": True

},

{

"title": "Alarm Description",

"value": self.msg["AlarmDescription"],

"short": False

},

{

"title": "Trigger",

"value": " ".join([

self.trigger["Statistic"],

self.trigger["MetricName"],

self.trigger["ComparisonOperator"],

str(self.trigger["Threshold"]),

"for",

str(self.trigger["EvaluationPeriods"]),

"period(s) of",

str(self.trigger["Period"]),

"seconds."

]),

"short": False

},

{

'title': 'Old State',

'value': self.msg["OldStateValue"],

"short": True

},

{

'title': 'Current State',

'value': self.msg["NewStateValue"],

'short': True

},

{

'title': 'Link to Alarm',

'value': self.\_\_url(),

'short': False

}

]

}

]

}

return \_message

def lambda\_handler(event, context):

sns\_message = json.loads(event['Records'][0]['Sns']['Message'])

print(sns\_message)

webhook\_url = 'https://hooks.slack.com/services/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' # Set destination URL here

slack\_data = CloudWatchAlarmParser(sns\_message).slack\_data()

slack\_data["channel"] = 'cloudwatch'

request = Request(

webhook\_url,

data=json.dumps(slack\_data).encode(),

headers={'Content-Type': 'application/json'}

)

response = urlopen(request)

return {

'statusCode': response.getcode(),

'body': response.read().decode()

}

if \_\_name\_\_ == "\_\_main\_\_":

print(lambda\_handler(None, None))

if \_\_name\_\_ == "\_\_main\_\_":

print(lambda\_handler(None, None))

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

**Assignment Task #6  
  
Demonstrate the following server monitoring.  
  
1. List the top 5 unique IP addresses accessing your Apache webserver.  
2. List the past 10-days 4XX results of Apache's access log file, sorted by date with their IP address.  
3. Write a utility script to view the top 10 disk-space users of a given path across multiple hosts. \***

List the top 5 unique IP addresses accessing your Apache webserver.

awk '{ print $1}' access.log | sort -nr | uniq -c | sort -nr | head -n 10

sort the acces log by date

cat access.log | sort -t ' ' -k 4.9,4.12n -k 4.5,4.7M -k 4.2,4.3n -k 4.14,4.15n -k 4.17,4.18n -k 4.20,4.21n > website.log

print the value

awk '($9 ~ /403/)' website.log | awk '{print $1 " " $4$5 " " $9}' | sort | uniq -c

**Assignment Task #4  
  
Assume that you have gained unprivileged (non-root) access to a computer.  
  
How would you hog all resources and hang the device within 4 minutes? \***

<https://linuxconfig.org/how-to-crash-your-linux-system-with-fork-bomb>  
forkbomb(){ forkbomb | forkbomb & }; forkbomb