ENPM665 Exercise - AWS

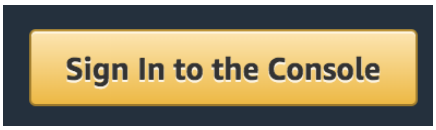
Version 2.5 – October 11th, 2022

Monitoring Security Events in AWS

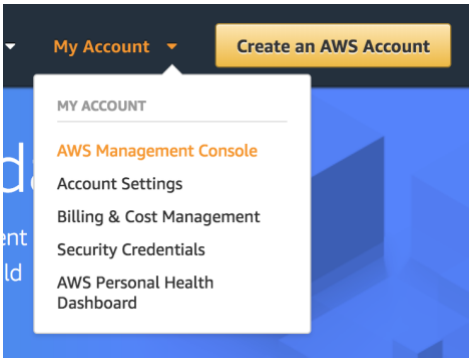
Background: In this example we'll use the AWS Console and CloudFormation templates to create a CloudWatch alarm that is triggered when a specific AWS API call is made. For this example, we will use certain S3 API calls that might be one of interest for someone tasked with monitoring the security of an AWS environment.

Login to AWS Console

1. Open a web browser and go to https://aws.amazon.com/
2. If you have previous logged in to the AWS console click "**Sign in to the Console**" (on the top right)

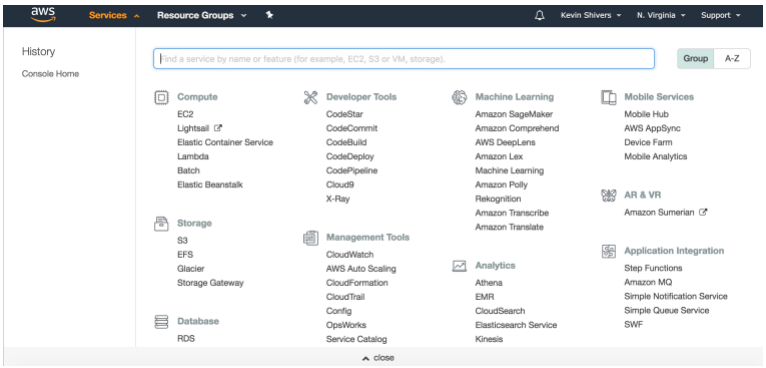


If not then click "My Account" and then "**AWS Management Console**" (on the top right)



3. Login using the user name and password you created. (Do the MFA step if you enabled MFA as well.)

You can view all of the AWS services by clicking the "Services" link at the top left of the console. It's a large list!

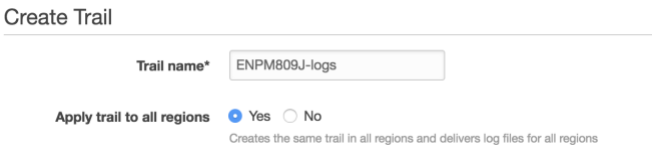


Enable Cloud Trail logging

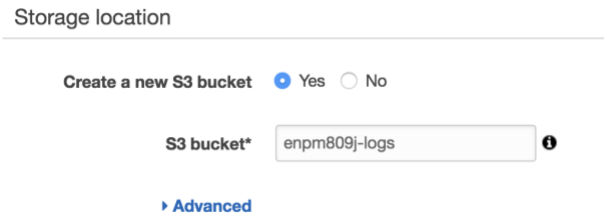
1. In the "**Management & Governance**" section click "**CloudTrail**"
2. Click the "**Create trail**" button

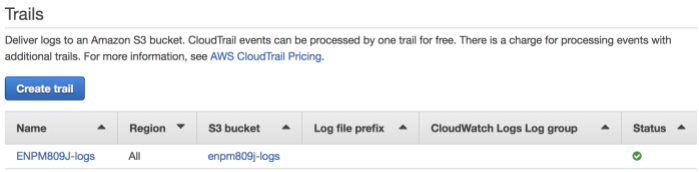
3. Give the trail a name (ex: "**ENPM809J-logs**")

4. Under "**Storage location**" create a new S3 bucket (I used "enpm809j-logs" so you'll need to use something different for a unique S3 bucket name.)

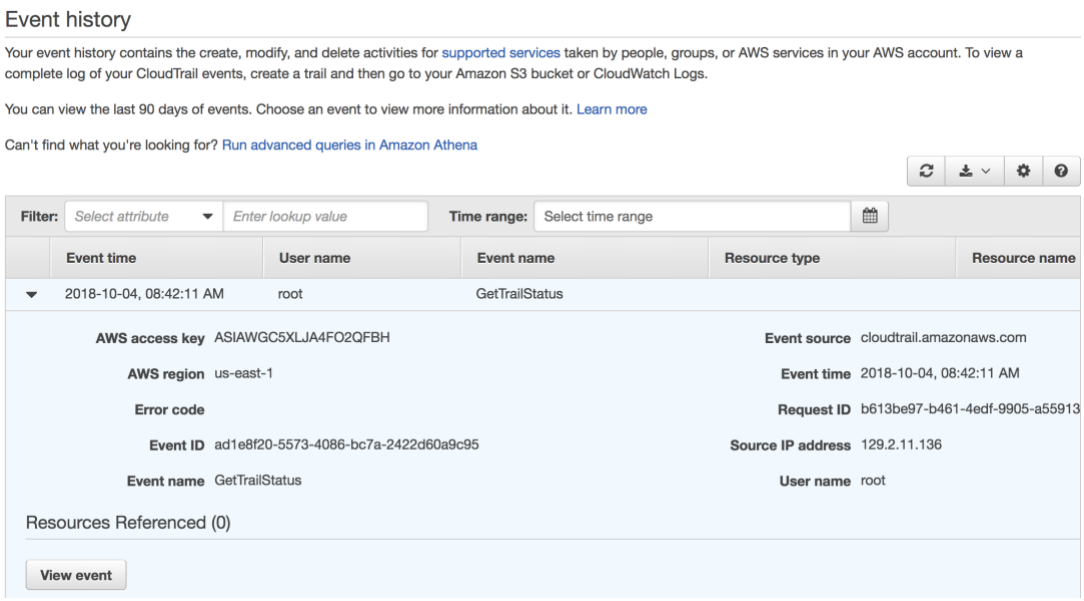
   

5. Leave the other settings as default
6. Click "**Create**"
7. When complete you'll see your newly created Cloudtrail and the S3 bucket results will go to.

Logs are exported on a scheduled basis so you may need to wait 10-15 minutes to see logs show up in the S3 bucket.

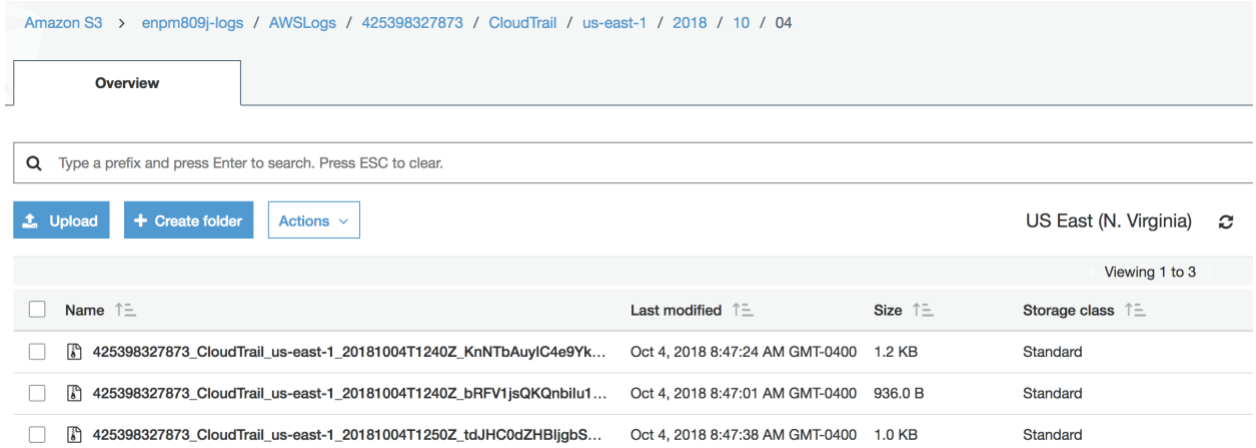
You can view events under the "**Event history**" section of Cloudtrail or wait for them to be written to S3 and view there.



Click "View event" to get the full log from the event in question.

In S3 you can view the logs via the console and scroll down the directory tree to access the logs saved in .json.gz format.

Directory tree format is: **S3 bucket** / AWS logs / **account number** / CloudTrail / **region** / **year** / **month** / **day** (where the bolded items will be specific to the account/date you are looking for)



To view them with the AWS CLI the command would look like:
`aws s3 ls s3://enpm809j-logs/AWSLogs/425398327873/CloudTrail/us-east-1/2018/10/04/` (Replace the S3 bucket name, account number, and date as needed)

```
itmc244524:scripts kts$ aws s3 ls s3://enpm809j-logs/AWSLogs/425398327873/CloudTrail/us-east-1/2018/
10/04/
2018-10-04 08:51:42       2305 425398327873_CloudTrail_us-east-1_20181004T1240Z_HEBU60cURFRU6S41.jso
n.gz
2018-10-04 08:47:24       1234 425398327873_CloudTrail_us-east-1_20181004T1240Z_KnNTbAuyIC4e9YkF.jso
n.gz
2018-10-04 08:47:01        936 425398327873_CloudTrail_us-east-1_20181004T1240Z_bRFV1jsQKQnbiIu1.jso
n.gz
2018-10-04 08:50:23        589 425398327873_CloudTrail_us-east-1_20181004T1245Z_xqK5kqh4mYYGl5HH.jso
n.gz
2018-10-04 08:47:38       1024 425398327873_CloudTrail_us-east-1_20181004T1250Z_tdJHC0dZHBIjgbSK.jso
n.gz
```

To save one of the files you can download it via the Console UI or use the AWS CLI tools.

Ex: **`aws s3 cp s3://enpm809j-logs/AWSLogs/425398327873/CloudTrail/us-east-1/2018/10/04/425398327873_CloudTrail_us-east-1_20181004T1250Z_tdJHC0dZHBIjgbSK.json.gz .`** (Replace the S3 bucket name, account number, date, file name as needed)

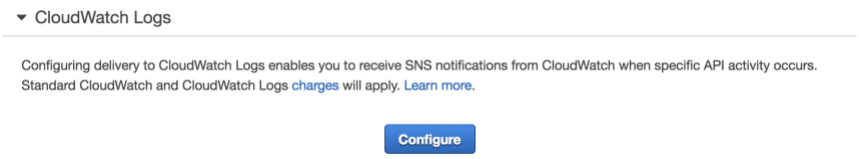
```
itmc244524:scripts kts$ aws s3 cp s3://enpm809j-logs/AWSLogs/425398327873/CloudTrail/us-east-1/2018/
10/04/425398327873_CloudTrail_us-east-1_20181004T1250Z_tdJHC0dZHBIjgbSK.json.gz .
download: s3://enpm809j-logs/AWSLogs/425398327873/CloudTrail/us-east-1/2018/10/04/425398327873_Cloud
Trail_us-east-1_20181004T1250Z_tdJHC0dZHBIjgbSK.json.gz to ./425398327873_CloudTrail_us-east-1_20181
004T1250Z_tdJHC0dZHBIjgbSK.json.gz
itmc244524:scripts kts$ gzip -d 425398327873_CloudTrail_us-east-1_20181004T1250Z_tdJHC0dZHBIjgbSK.js
on.gz
itmc244524:scripts kts$ cat 425398327873_CloudTrail_us-east-1_20181004T1250Z_tdJHC0dZHBIjgbSK.json
{"Records":[{"eventVersion":"1.06","userIdentity":{"type":"Root","principalId":"425398327873","arn":
"arn:aws:iam::425398327873:root","accountId":"425398327873","accessKeyId":"ASIAWGC5XLJA4FO2QFBH","se
ssionContext":{"attributes":{"creationDate":"2018-10-04T12:32:14Z","mfaAuthenticated":"true"}}},"eve
ntTime":"2018-10-04T12:42:11Z","eventSource":"cloudtrail.amazonaws.com","eventName":"GetTrailStatus"
,"awsRegion":"us-east-1","sourceIPAddress":"129.2.11.136","userAgent":"console.amazonaws.com","reque
stParameters":{"name":"arn:aws:cloudtrail:us-east-1:425398327873:trail/ENPM809J-logs"},"responseElem
```

In the screenshot above I also decompressed the file (**`gzip -d`**) and then ran **`cat`** to view the contents, as you can see it's in raw JSON.

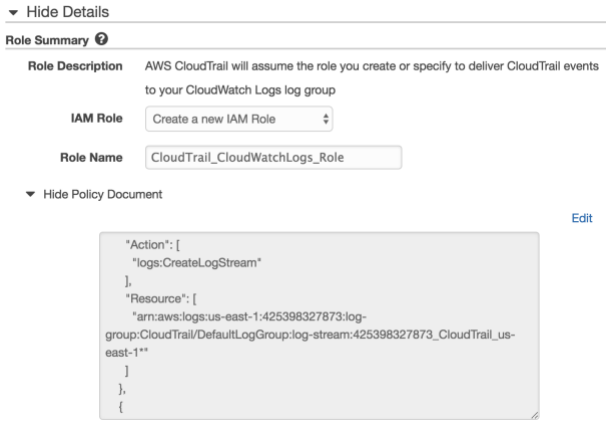
Create a CloudWatch log group

CloudTrail uses a CloudWatch Logs log group as a delivery endpoint for log events.

1. In the AWS Management Console, Under "**Management & Governance"**, Select "**CloudTrail"**
2. Select the trail we created above (ex: "**ENPM809J-logs**") by clicking on the trail name
3. Scroll down to the "**CloudWatch Logs**" section and click "**Configure**"



4. In the New or existing log group box, keep the **DefaultLogGroup** and then click **Continue**.
5. **Leave the defaults on the next page**. Click View Details and look at the Role Name box. Expand View Policy Document. The default role policy contains the permissions required for creating a CloudWatch Logs log stream in a log group that you specify and for delivering CloudTrail events to that log stream.



6. Click **Allow**.

When you are finished with these steps in the console, the CloudTrail trail will be set up to use the log group and role you specified to send events to CloudWatch Logs. If the trail you configured to use CloudWatch Logs receives log files across regions, events from all regions will be sent to the CloudWatch Logs log group that you specified.

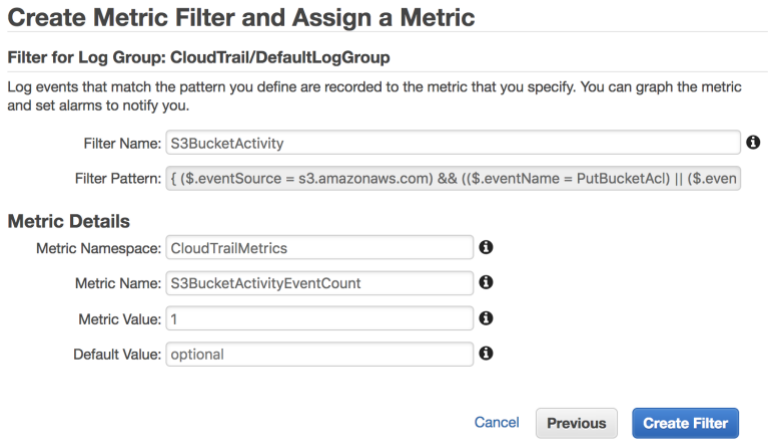
Create a CloudWatch Metric Filter

1. In the AWS Management Console, Under **Management & Governance** Tools, Select **CloudWatch**
2. In the navigation pane on left, click **Logs**
3. In the list of log groups, select the radio button next to the log group that you created for CloudTrail log events. (ex: "**CloudTrail/DefaultLogGroup**")
4. Click the **Create Metric Filter** button
5. On the Define Logs Metric Filter screen enter the following in **Filter Pattern**:

```
{ ($.eventSource = s3.amazonaws.com) && (($.eventName = PutBucketAcl)
|| ($.eventName = PutBucketPolicy) || ($.eventName = PutBucketCors) ||
($.eventName = PutBucketLifecycle) || ($.eventName =
PutBucketReplication) || ($.eventName = DeleteBucketPolicy) ||
($.eventName = DeleteBucketCors) || ($.eventName =
DeleteBucketLifecycle) || ($.eventName = DeleteBucketReplication)) }
```

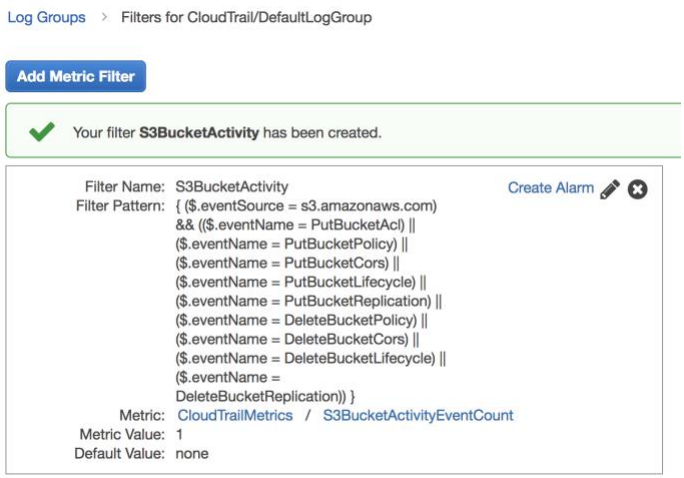
Review this filter pattern and take a note of what it is doing. Notice that a number of S3 bucket specific events are captured. Steps are provided for testing one such events but you may want to test additional filters.

6. Click **Assign Metric**
7. On the Create Metric Filter and Assign a Metric screen, in the Filter Name box, delete existing text and enter **S3BucketActivity**
8. Under Metric Details, in the Metric Namespace box enter **CloudTrailMetrics**.
9. In the Metric Name field, enter **S3BucketActivityEventCount**
10. Click **Show advanced metric settings**
11. Click **Metric Value**, and then ensure that the value is **1**.
12. Click **Create Filter**



Create an Alarm for the Metric Filter

1. When you finish the step above you'll see a confirmation screen with a "Create Alarm" link at the top right



2. Click "**Create Alarm**"
3. On the **Specify metric and conditions** page, provide the following values

   Name: **S3 Bucket Activity**
   Metric Name: **S3BucketActivityEventCount**
   Whenever S3BucketActivityEventCount is >= 1 for 1 datapoints
   Period: **1 Minute**

   Under "Additional configuration"

   Missing data treatment: **Treat missing data as good (not breaching threshold**)

   Under Actions, Click **New list** for Send notification to:, provide a topic name such as **Notify** and **enter your email address**.

   Everything should look like this:

Metric

Edit

Graph

This alarm will trigger when the blue line goes above the red line for 1 datapoints within 1 minute

No unit

3
2.5
2
1.5
1
11:00 12:00 13:00

S3BucketActivityEventCount

Namespace

CloudTrailMetrics

Metric name

S3BucketActivityEventCount

Statistic

Sum

Period

1 minute

Conditions

Whenever S3BucketActivityEventCount is...

Define the alarm condition

- Greater (> threshold)
- Greater/Equal (>= threshold)
- Lower/Equal (<= threshold)
- Lower (< threshold)

than...

Define the threshold value

1

Must be a number

Additional configuration

Datapoints to alarm

Define the number of datapoints within the evaluation period that must be breaching to cause the alarm to go to ALARM state.

1 out of 1

Missing data treatment

How to treat missing data when evaluating the alarm

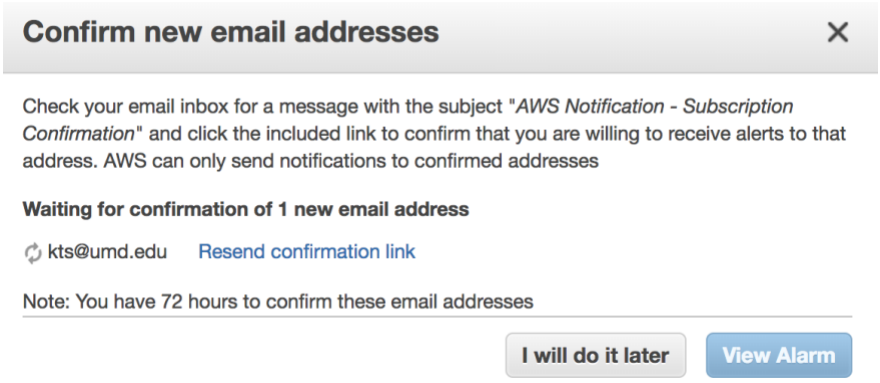
Treat missing data as good (not breaching threshold)

Cancel Next

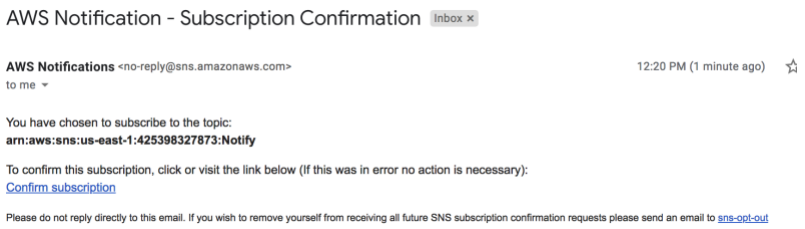
4. Click "**Next**"

5. On the **Configure actions** page select “Create new topic” under “Select an SNS topic” and enter your email address so you receive alarms and then click the “**Create topic**” button.

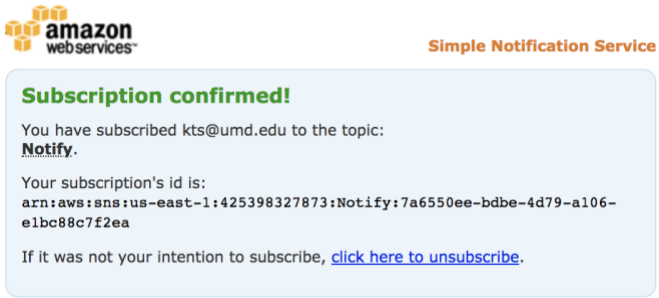
6. Click “**Next**”
7. In the “Add a description” page enter something for the Alarm name like “**S3 Bucket Activity**” and add a description if you like.
8. Click “**Next”**
9. Review everything one last time and then click “**Create Alarm**”
10. If asked to confirm email address check your email and confirm your email address.



Email:



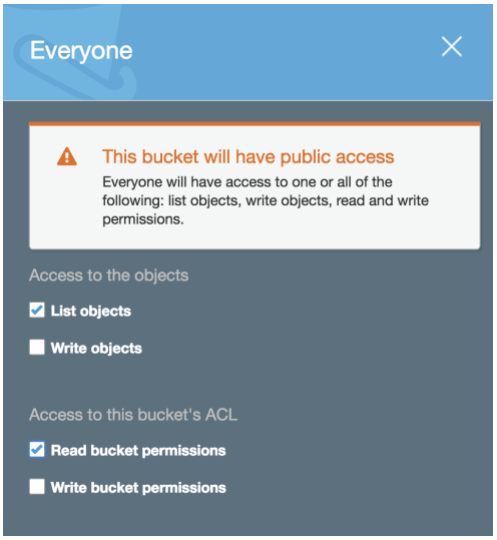
Confirmed:



11. Once complete you can click “View alarm”

Generate some S3 activity

1. In the AWS Console select **Services** and then **S3**
2. Create a new S3 bucket logs (ex enpm809j-test)
3. Click the Permissions tab
4. Under “Public access” select “Everyone” and select “List objects” and “Read bucket permissions”

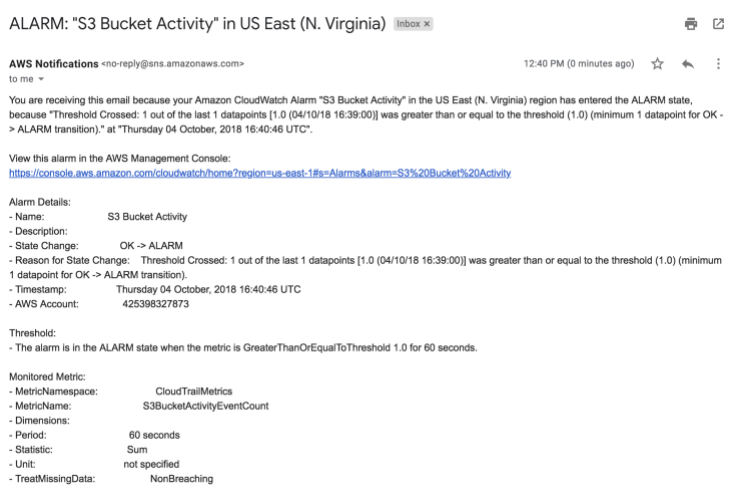


5. Click “Save”
6. You should receive an Alarm S3 bucket Activity via email in a few minutes once the CloudTrail log has been written and the alarm has been processed.

Note: - If you have not received email notification, navigate to AWS Console, Services, CloudWatch and click on Alarms. If this shows Config Status as Pending confirmation then that means you have not yet confirmed SNS subscription yet. Refer to your email and subscribe to notifications from this module.

Repeat steps 3 through 5 but unselect the public options to remove the public access. You can also create a new bucket and then delete it as additional examples.

Sample alarm email:



ALARM: "S3 Bucket Activity" in US East (N. Virginia) Inbox ×

AWS Notifications <no-reply@sns.amazonaws.com> 12:40 PM (0 minutes ago)
to me

You are receiving this email because your Amazon CloudWatch Alarm "S3 Bucket Activity" in the US East (N. Virginia) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [1.0 (04/10/18 16:39:00)] was greater than or equal to the threshold (1.0) (minimum 1 datapoint for OK -> ALARM transition)." at "Thursday 04 October, 2018 16:40:46 UTC".

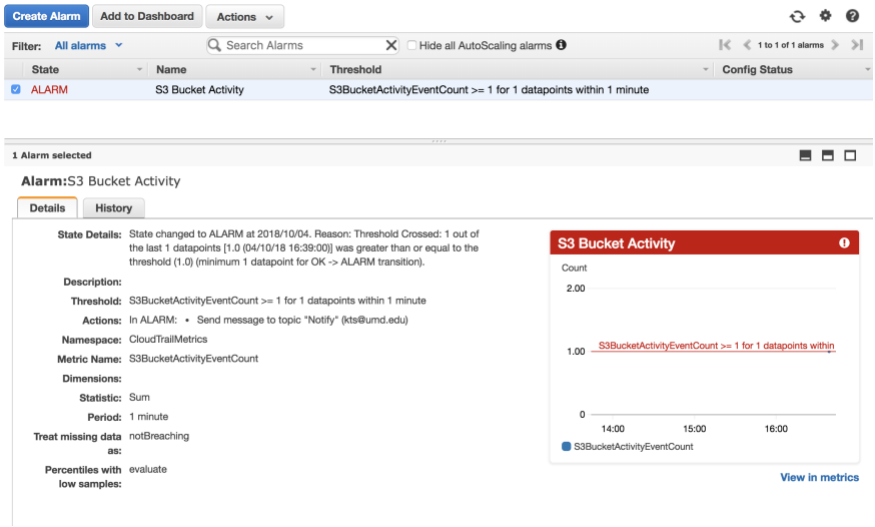
View this alarm in the AWS Management Console:
https://console.aws.amazon.com/cloudwatch/home?region=us-east-1#s=Alarms&alarm=S3%20Bucket%20Activity

Alarm Details:
- Name: S3 Bucket Activity
- Description:
- State Change: OK -> ALARM
- Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [1.0 (04/10/18 16:39:00)] was greater than or equal to the threshold (1.0) (minimum 1 datapoint for OK -> ALARM transition).
- Timestamp: Thursday 04 October, 2018 16:40:46 UTC
- AWS Account: 425398327873

Threshold:
- The alarm is in the ALARM state when the metric is GreaterThanOrEqualToThreshold 1.0 for 60 seconds.

Monitored Metric:
- MetricNamespace: CloudTrailMetrics
- MetricName: S3BucketActivityEventCount
- Dimensions:
- Period: 60 seconds
- Statistic: Sum
- Unit: not specified
- TreatMissingData: NonBreaching

Alarm in the Console:



Create Alarm | Add to Dashboard | Actions

Filter: All alarms | Search Alarms | Hide all AutoScaling alarms | 1 to 1 of 1 alarms

State	Name	Threshold	Config Status
ALARM	S3 Bucket Activity	S3BucketActivityEventCount >= 1 for 1 datapoints within 1 minute	

1 Alarm selected

Alarm:S3 Bucket Activity

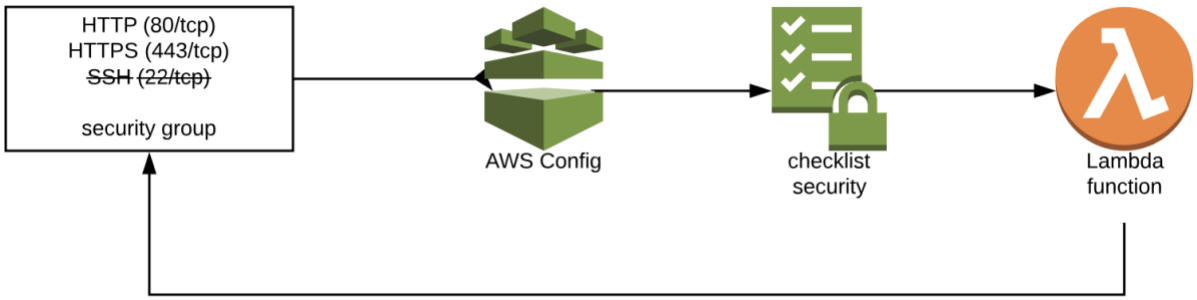
Details | History

State Details: State changed to ALARM at 2018/10/04. Reason: Threshold Crossed: 1 out of the last 1 datapoints [1.0 (04/10/18 16:39:00)] was greater than or equal to the threshold (1.0) (minimum 1 datapoint for OK -> ALARM transition).
Description:
Threshold: S3BucketActivityEventCount >= 1 for 1 datapoints within 1 minute
Actions: In ALARM: • Send message to topic "Notify" (kts@umd.edu)
Namespace: CloudTrailMetrics
Metric Name: S3BucketActivityEventCount
Dimensions:
Statistic: Sum
Period: 1 minute
Treat missing data as: notBreaching
Percentiles with low samples: evaluate

View in metrics

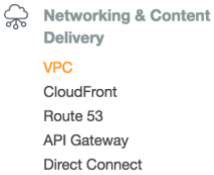
Monitoring Security Groups with AWS Config

In this exercise we'll use AWS Config Rules with an AWS Lambda function to monitor the ingress ports associated with an EC2 security group. The Lambda function will be triggered whenever the security group is modified. If the ingress rule configuration differs from that which is coded in the function, the Lambda function will revert the ingress rules back to the appropriate configuration. The activity from the Lambda function can then be viewed through Amazon CloudWatch Logs.



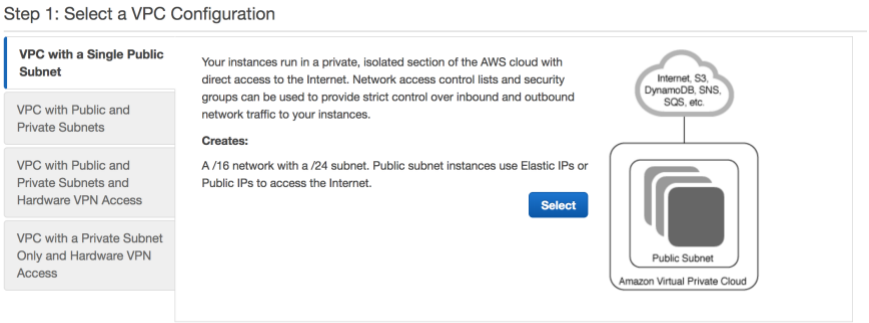
Create a VPC

1. In the Services drop down scroll down to "**Networking & Content Delivery**" and click "**VPC**"

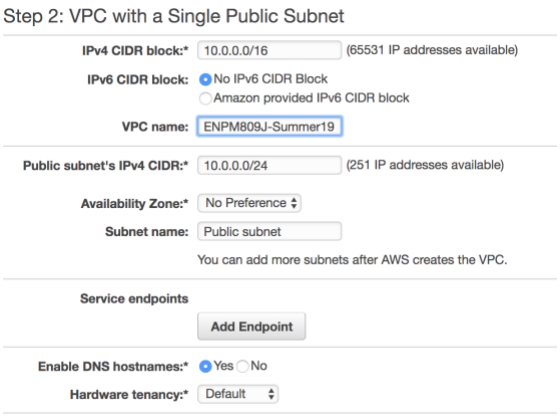


2. Click "**Launch VPC Wizard**"

3. By default the "VPC with a Single Public Subnet" tab is selected, click "**Select**"

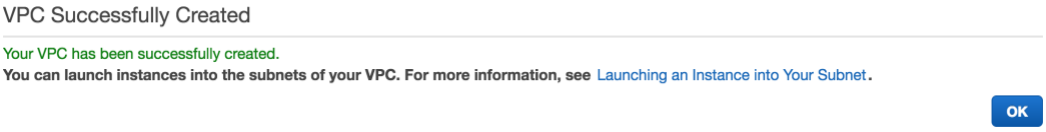


4. Leave the defaults, put something for "VPC name", I'll use **ENPM809J-Summer19** for my example



5. Click "**Create VPC**"

6. You will be taken to a page titled "VPC Successfully Created", click "**Ok**"



Create a Security Group

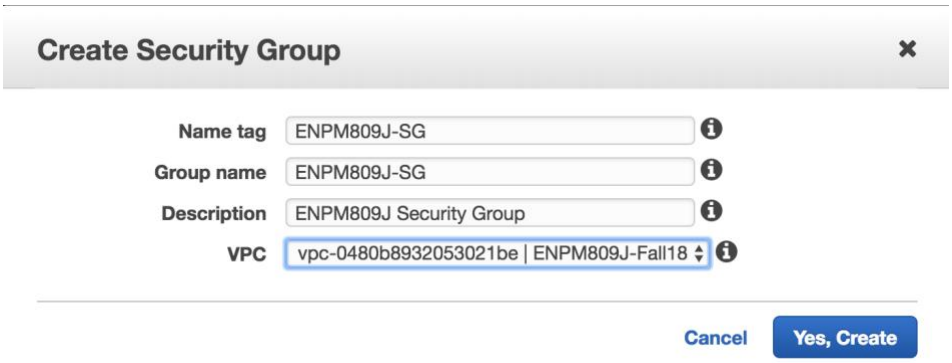
1. On the menu on the left scroll down and select "**Security Groups**" under "**Security**"

2. Click "**Create Security Group**"

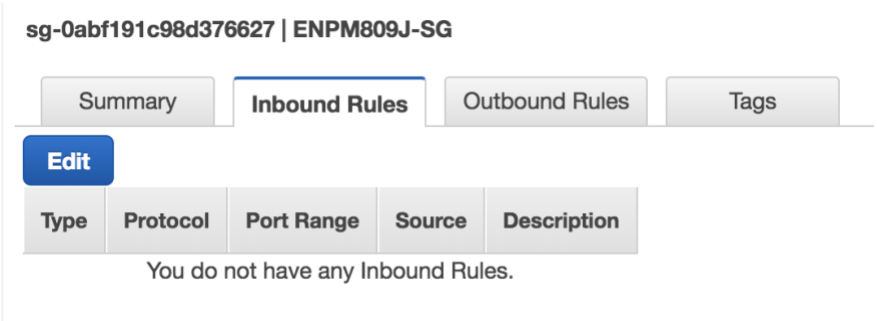
3. In the popup enter the following:

Name tag: **ENPM809J-SG**
Group name**: ENPM809J-SG** (this will auto fill for you after you enter the Name tag)
Description: **ENPM809J Security Group**
VPC: The name of the VPC you created above.

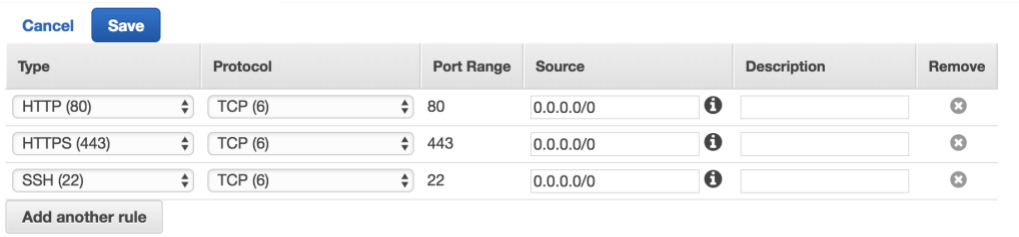
Then click "**Yes, Create**"



4. Click the "**ENPM809J-SG**" row
5. Select the "**Inbound Rules**" tab
6. Click "**Edit**"



7. Jot down the Security Group identifier (ex: **sg-0abf191c98d376627** in my example yours will be different) somewhere as we'll need it later
8. Add the following rules and then click "**Save**"

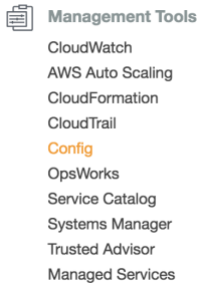


What we've done is created a Security Group that allows in traffic from ports 80/tcp (HTTP), port 443/tcp (HTTPS), and port 22/tcp (SSH) from the world. (The world = 0.0.0.0/0 = all IP addresses)

In a production environment you should not allow SSH (port 22/tcp) open to the world, typically the only world accessible service you would see would be HTTP/HTTPS.

Enable AWS Config

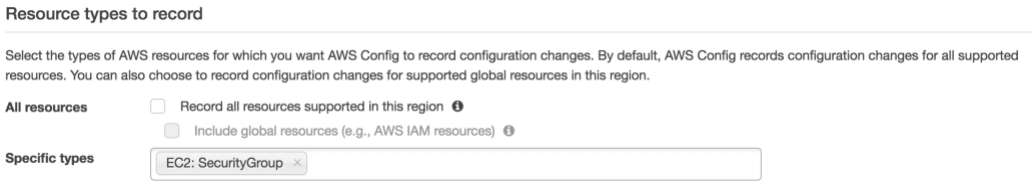
1. On the Services menu, under **Management & Governance** click **Config**



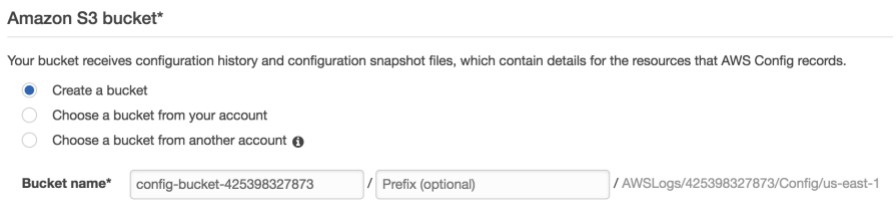
2. Click **Get Started** if you see a button with that text, else click **Settings**.

3. Under Resource types to record, **uncheck** the box "Record all resources supported in this region"

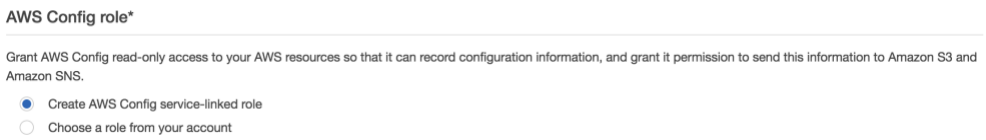
4. Click the **Specific types** box. A scroll box field will appear. Scroll down to the EC2 section and click **SecurityGroup**. You should see **EC2: Security Group** appear in the Specific types box. Click outside of the box to close the scroll box field.



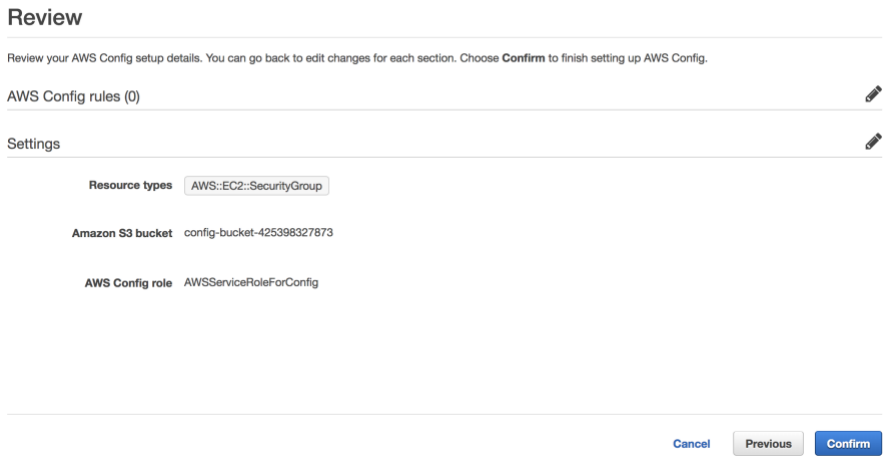
5. Under Amazon S3 bucket, select **Create a bucket**. In the Bucket name field, use the default name that is provided. Leave the Prefix (optional) text box empty. Make sure that the Bucket Name is not already created else you will get a bucket already exist error.



6. Under AWS Config Role, select **Create AWS Config service-linked role**



7. Click the **Next** button at the bottom right of the web page.
8. On the AWS Config Rules page, do not select any rules. You will add a custom rule later. Click **Next**.
9. On the Review page, click **Confirm**.



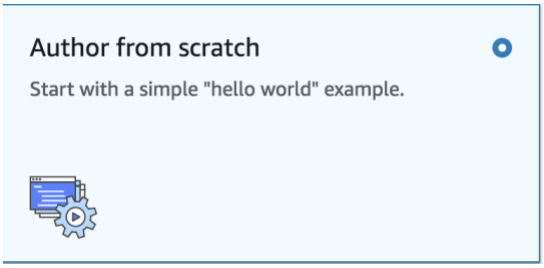
10. After a while, you will see the Config Dashboard page appear. Click "**Add Rule**"

11. The Add rule page will appear, Click the **Add custom rule** button.

12. Enter the following:

    Name: **EC2SecurityGroup**
    Description: **Restrict ingress ports to HTTP and HTTPS**

13. Click "**Create AWS Lambda function**". This will open up Lambda in a new tab.
14. Click "**Author from scratch**"



15. Enter the following:

    Name: **awsconfig_lambda_security_group**
    Runtime: **Python 2.7**
    Role: **Create a custom role** (**Note:** this will open up a new tab/window)

16. In this new tab/window enter the following:

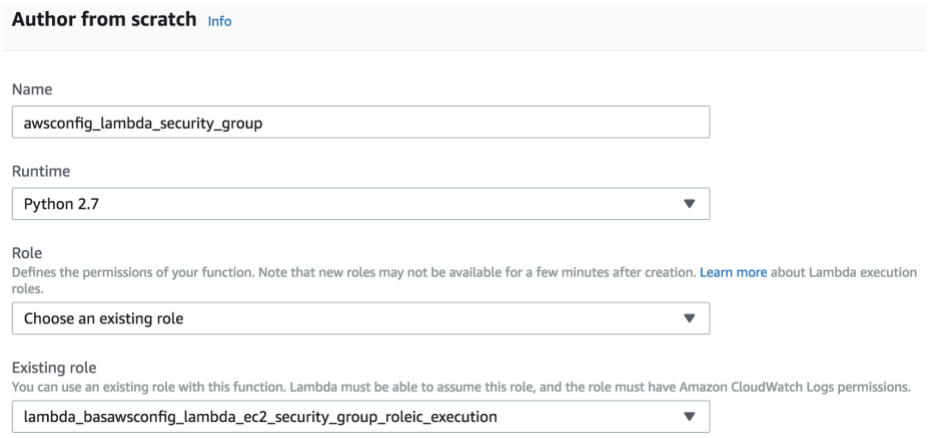
    IAM Role: **Create a new IAM Role**
    Role Name: **awsconfig_lambda_ec2_security_group_role**

17. Click **"View Policy Document"** to open the policy drop down
18. Click **Edit**. (Click **Ok** if a warning message appears about reading the documentation.)
19. In the policy window erase the existing content and enter the following:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "config:PutEvaluations",
                "ec2:DescribeSecurityGroups",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": "*"
        }
    ]
}
```

The page should look like:

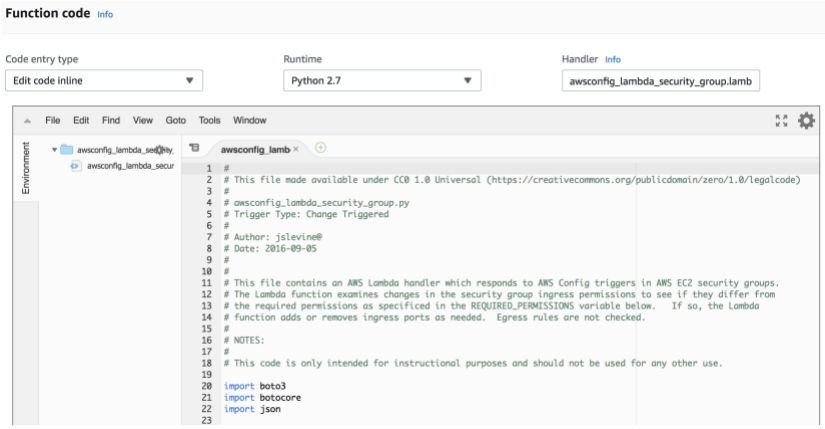
20. Click the **Allow** button. The page will close and you will return to the Lambda Basic Information page
21. Notice the Role has been changed to "**Choose an existing role**" and the Existing role has been changed to **awsconfig_lambda_ec2_security_group_role**



22. Click **Create function**

23. Scroll down to the **Function code** section
24. In the "Code entry type" drop down select "**Upload a .zip file**"
25. Download this file: https://github.com/kts262/enpm809j/blob/master/aws-config/awsconfig_lambda_security_group.py.zip and save it on your computer.
26. Click the **Upload button** under Function Package and upload the file you just downloaded.
27. In the Handler field enter **awsconfig_lambda_security_group.lambda_handler**.
28. Leave the Memory (MB) field under Basic Settings field with the default value of **128**.
29. In the Timeout fields, set min to **1** and sec to **0**
30. Lambda functions can run for a maximum of fifteen minutes. This is particular function typically takes less than five seconds to run so allowing one minute should be more than adequate.
31. Leave the other default settings as they are.
32. Click the "**Save**" button at the top right of the page.

33. Review under Function code that the Python code has been added, I should look something like this:



34. Scroll down to the "**REQUIRED_PERMISSIONS**" section and notice it lists port 80 and 443. For this example those are the only ports we want to have open. (And not 22/tcp – SSH which we also added earlier.)

```
REQUIRED_PERMISSIONS = [
{
    "IpProtocol" : "tcp",
    "FromPort" : 80,
    "ToPort" : 80,
    "UserIdGroupPairs" : [],
    "IpRanges" : [{"CidrIp" : "0.0.0.0/0"}],
    "PrefixListIds" : [],
    "Ipv6Ranges": []
},
{
    "IpProtocol" : "tcp",
    "FromPort" : 443,
    "ToPort" : 443,
    "UserIdGroupPairs" : [],
    "IpRanges" : [{"CidrIp" : "0.0.0.0/0"}],
    "PrefixListIds" : [],
    "Ipv6Ranges": []
}]
```

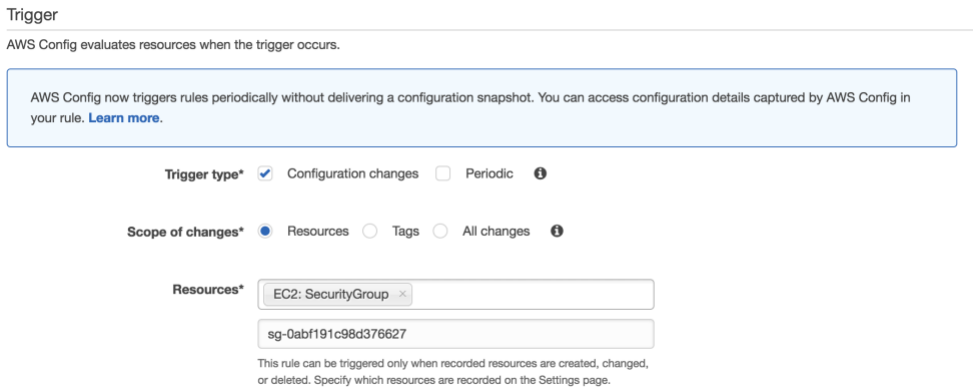
Inside the code there are 2 functions authorize_security_group_ingress() and revoke_security_group_ingress() to add or remove permissions as appropriate. Therefore, we should expect that the SSH (22/tcp) permissions should be removed when we run this function. And if we removed say HTTP (80/tcp) then those permissions would be added back in when this function runs.

35. On the upper right part of the page you should some text following ARN. Copy the text beginning with arn:aws:lambda all the way to the end into scratch text file or leave it in your copy/paste buffer. It should look something like this:

    arn:aws:lambda:us-east-1:425398327873:function:awsconfig_lambda_security_group

36. Go back to the AWS Config page that should still be open to Add custom rule
37. In the AWS Lambda function ARN field, enter the arn:aws:lambda value that you copied in step 35.

38. For Trigger type select **Configuration changes**
39. For Scope of changes select **Resources**
40. Click in the Resources text box and scroll box will appear select **EC2: SecurityGroup**.
41. For the "Resource identifier" enter the security group identifier you copied earlier ex: **sg-0abf191c98d376627** from step 7 of "Create a Security Group" above.



42. In Rule parameters, enter the following:

    Key: **debug**
    Value: **true** (this will generate additional data you can look at later if you choose.)

Rule parameters

Rule parameters define attributes for which your resources are evaluated; for example, a required tag or S3 bucket.

Key	Value	
debug	true	
Key	Value	

43. Click **Save**.
44. This will return you to the AWS Config Rules page. Under the Compliance column, you will see the function has been submitted for an initial evaluation. This initial evaluation may take several minutes to complete. This same evaluation will also take place whenever the security group is changed again in the future.
45. Click the refresh button at the top right periodically as well to update the evaluation status.

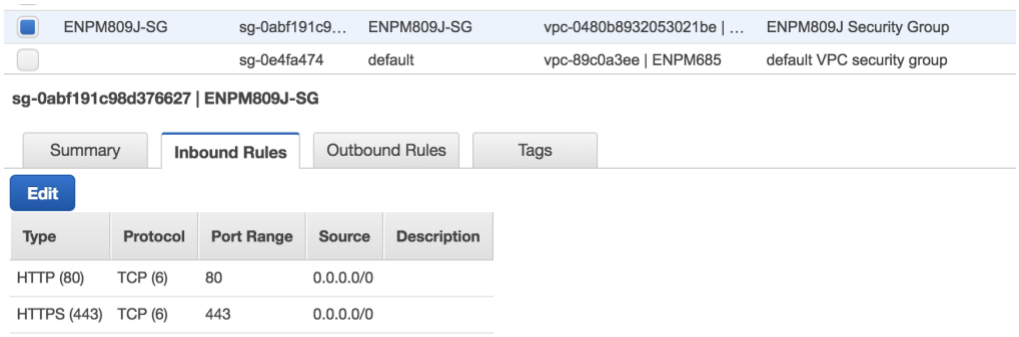
46. Once the compliance evaluation has taken place, you should see the following

Rule name	Compliance
EC2SecurityGroup	Compliant

Verify AWS Config Actually Did Something

We will now examine the VPC security group that we had previously created to allow HTTP, HTTPS, and SSH traffic in from the world.

1. Click the Services menu and select **VPC** under “Networking & Content Delivery”. The VPC Dashboard will appear.
2. On the left hand side of the window under Security click **Security Groups**.
3. Click the Security Group you created (ex. **ENPM809J-SG**)
4. Click the Inbound tab that appears below.
5. Notice that only HTTP and HTTPS traffic are permitted as shown below.



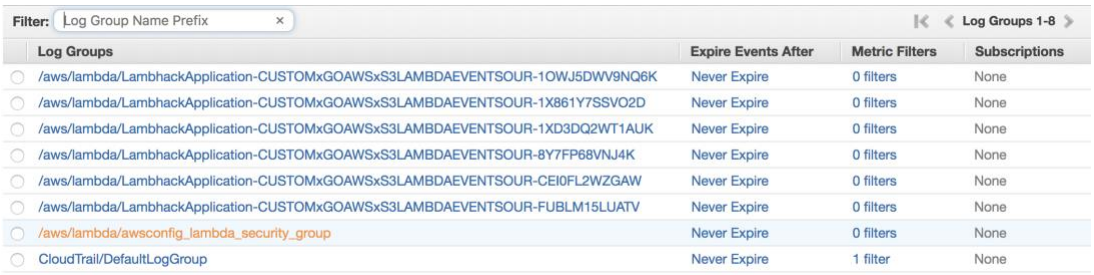
6. **AWS Config did a thing!** In this case, the detection/modification happened during the initial AWS Config rule validation. If you were to modify the security group again, a compliance evaluation would be triggered which would again invoke the Lambda function and the changes would be reverted. (Go ahead and try it, remove port 80, or add in a different port and then review a few minutes later to see if the Lambda function has restored it back to just port 80 and 443.)

Verify Actions with Amazon CloudWatch Logs

Let's review the Amazon CloudWatch Logs to see what the Lambda function did.

1. Click the Services menu and select **CloudWatch** under "**Management & Governance**"
2. On the left side of page, select Logs.

3. Click on the Log Group that contains **awsconfig_lambda_security_group**.



4. Under Log Streams, beginning with the top link, click each link until you see an entry that contains the words revoking for and expand the entry. You should see something similar to the screenshot below This shows that the entry for ports 22/tcp (SSH) was removed.

What Have We Learned?

- You have successfully created a Trail in AWS CloudTrail console
- Created a log group in CloudWatch console that receives logs from CloudTrail
- Created a number of metric filters and corresponding alarms for automated notifications
- You have also learned how to automate the steps via AWS CloudFormation.
- You enabled AWS Config
- Uploaded a Lambda function to support a rule for AWS Config that evaluations permissions on an EC2 security group.
- Modified the default VPC Security group to contain both compliant and noncompliant permissions
- Enabled the AWS Config Rule and observed the results
- Examined the activity of the Lambda function using Amazon Cloudwatch Logs.

In a real world scenario we would probably use something more like Splunk or ELK to review logs from AWS or alert on important events vs AWS native tools. Something we'll take a look at in a few weeks. (I can see using CloudWatch Alarms/SNS and/or Splunk/ELK for alerting, that's more of an organizational preference.)

Let's Undo All This Work Now…

Let's undo all of this work so you won't be charged for any of this after you have completed this exercise.

1. In the AWS Management Console, on the Services menu, click CloudWatch
2. Under Alarms, select S3 Bucket Activity
3. click on Actions, select Delete
4. click on Yes, Delete
5. In the AWS Management Console, on the Services menu, click CloudTrail
6. Open the trail that you created in this exercise
7. In top right, next to Logging, click on "On/Off" switch
8. Click Continue in the popup window
9. Click on the "trash can" icon to delete the Trail
10. Click Delete in the popup window
11. Delete the IAM role "awsconfig_lambda_ec2_security_group_role".
12. Delete the AWS Config Rule "EC2SecurityGroup".
13. If AWS Config was not enabled, turn off AWS Config in Config Settings.
14. Delete the Lambda function "awsconfig_lambda_security_group".
15. Delete S3 bucket for config if created during the exercise
16. Delete Config role if created "config-role-"
17. Delete CloudWatch logs group "/aws/lambda/awsconfig_lambda_security_group