

Phase 1: Create the Infrastructure Using CloudFormation CFN template

Step 1: Download the Cloud formation (CFN) template on your local Machine

https://gitlab.aws.dev/rajeabh/s3bucketkey-poc/-/blob/main/cfn/cfn_s3_bucket_key_poc.json

Step 2: CFN Stack

Login to AWS console and go to CloudFormation → Create Stack → Chose the downloaded cfn_s3_bucket_key_poc file as shown in diagram below.

The screenshot shows the 'Create stack' wizard in the AWS CloudFormation console. On the left, a sidebar lists the steps: Step 1: Specify template (active), Step 2: Specify stack details, Step 3: Configure stack options, and Step 4: Review. The main area is titled 'Create stack'. Under 'Prerequisite - Prepare template', there are three radio buttons: 'Template is ready' (selected), 'Use a sample template', and 'Create template in Designer'. Below this, the 'Specify template' section explains that a template is a JSON or YAML file. It has two options for 'Template source': 'Amazon S3 URL' and 'Upload a template file' (selected). Under 'Upload a template file', there is a 'Choose file' button. A callout box with an arrow points to this button, containing the text 'Chose the downloade template' (note the typo). The file name 'cfn_s3_bucket_key_v5.json' is visible next to the button.

Step 3: Specify Stack Details: Enter the Stack Name and Bucket Names and click Next. Please note: CFN template will create two buckets. First bucket with S3 Bucket Keys DISABLED and Second Bucket with S3 Bucket Keys ENABLED.

Specify stack details

The screenshot shows the 'Specify stack details' step of the 'Create stack' wizard. The 'Stack name' field is filled with 'S3BucketKeyPOC'. Below it, a note states: 'Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-)'. The 'Parameters' section is titled 'Parameters' and includes a description: 'Parameters are defined in your template and allow you to input custom values when you create or update a stack.' There are two parameter fields: 's3bucket1' with the value 'rajeabh-0462' and 's3bucket2' with the value 'rajeabh-0462-bucketkey'. At the bottom right, there are three buttons: 'Cancel', 'Previous', and 'Next'.

Step 4: Configure Stack Option, you don't need to change anything, just need to click Next

Advanced options

You can set additional options for your stack, like notification options and a stack policy. [Learn more](#)

► **Stack policy**
Defines the resources that you want to protect from unintentional updates during a stack update.

► **Rollback configuration**
Specify alarms for CloudFormation to monitor when creating and updating the stack. If the operation breaches an alarm threshold, CloudFormation rolls it back. [Learn more](#)

► **Notification options**

► **Stack creation options**

[Cancel](#) [Previous](#) [Next](#)

Step 5: Review the stack and acknowledge that CFN template is creating IAM role.

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.

[Cancel](#) [Previous](#) [Create change set](#) [Create stack](#)

Step 6: Note down the stack output values

Please note that **values** for KMSKeyARN1 and KMSKeyARN2 these values required in Athena query to analyze the KMS traffic

S3BucketKeysPOC

[Delete](#) [Update](#) [Stack actions](#) [Create stack](#)

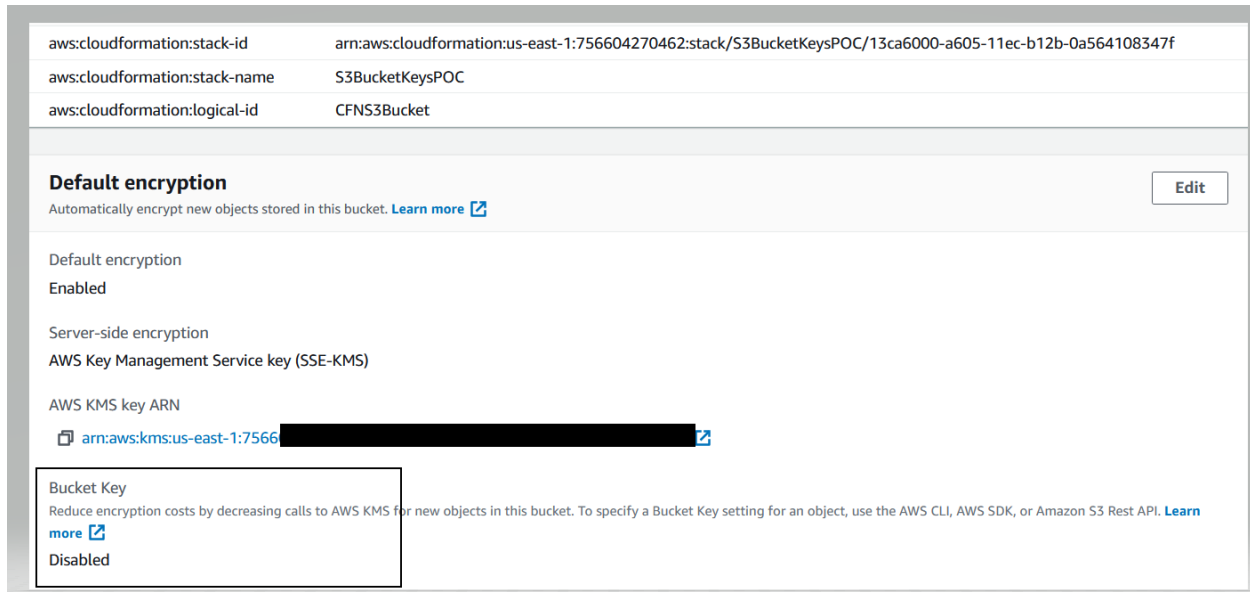
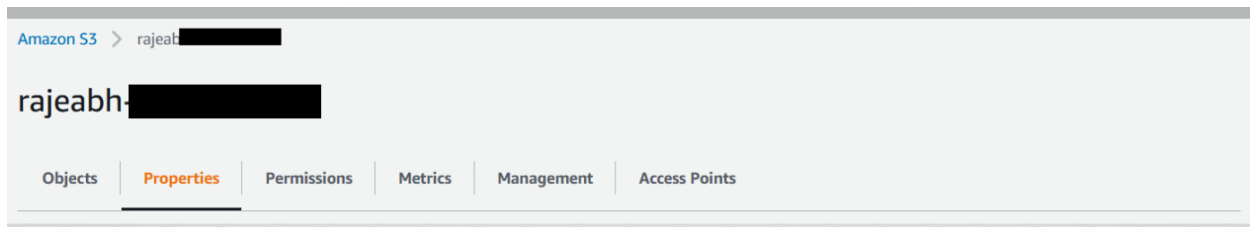
[Stack info](#) [Events](#) [Resources](#) [Outputs](#) [Parameters](#) [Template](#) [Change sets](#)

Key	Value	Description	Export name
CFNIAMRoleOut	arn:aws:iam::756604270462:role/S3BucketKeysPOC-CFNIAMRole-1510XRI981DEK	-	-
KMSKeyARN1	arn:aws:kms:us-east-1:756604270462:key/a2f17c4c426d	KMS key ARN used for encrypting S3 bucket with NO S3 Bucket Keys	-
KMSKeyARN2	arn:aws:kms:us-east-1:756604270462:key/f454-4190-98b6-769f5affec1c	KMS key ARN used for encrypting S3 bucket with S3 Bucket Keys ENABLED	-
LambdaFnOut	s3_bucket_test_simulator	Lambda function to generate traffic on both buckets	-
S3BUCKETAWSKMS	rajeabh-0462	S3 Bucket with NO S3 Bucket Keys.	-
S3BUCKETAWSKMS2	rajeabh-0462-s3bucketkey	S3 Bucket with S3 Bucket Keys ENABLED.	-

Step 7: Verify S3 Buckets

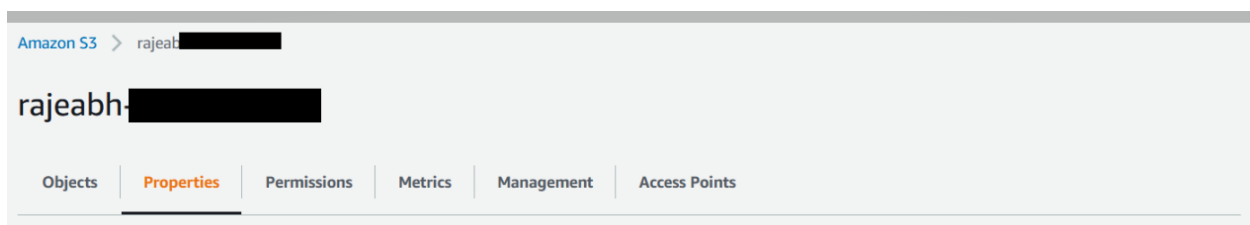
Go to S3 Bucket → First Bucket → Click on Properties → Default Encryption

Observe S3 Bucket Key is Disabled



Go to S3 Bucket → Second Bucket → Click on Properties → Default Encryption

Observe S3 Bucket Key is Enabled



aws:cloudformation:stack-id	arn:aws:cloudformation:us-east-1:756604270462:stack/S3BucketKeysPOC/13ca6000-a605-11ec-b12b-0a564108347f
aws:cloudformation:stack-name	S3BucketKeysPOC
aws:cloudformation:logical-id	CFNS3Bucket2

Default encryption
[Learn more](#)

Edit

Automatically encrypt new objects stored in this bucket.

Default encryption
Enabled

Server-side encryption
AWS Key Management Service key (SSE-KMS)

AWS KMS key ARN
arn:aws:kms:us-east-1:...

Bucket Key
Reduce encryption costs by decreasing calls to AWS KMS for new objects in this bucket. To specify a Bucket Key setting for an object, use the AWS CLI, AWS SDK, or Amazon S3 Rest API. [Learn more](#)

Enabled

Phase 2: Generate and analyze the KMS traffic

Now we have created all the Infrastructure we will enable the CloudTrail and Amazon Athena. Generate the traffic to both the S3 Buckets using Lambda function and Analyze the KMS API calls using Athena.

Step 8: Enable AWS CloudTrail account

AWS CloudTrail is a service that enables governance, compliance, operational auditing, and risk auditing of your AWS account. With AWS CloudTrail, you can log, continuously monitor, and retain account activity related to actions across your AWS. If you have already enabled CloudTrail, you can reuse the same. If CloudTrail is not enabled, you can enable CloudTrail as mentioned in the tutorial link below.

Please note, for the proposed solution you need to enable CloudTrail for management events only. You don't need to enable CloudTrail for data events or insight events. Also, please note that we need only single cloud trail and creating duplicate cloud trails can increase the cost. Please see the pricing page.

Tutorial:<https://docs.aws.amazon.com/awsCloudTrail/latest/userguide/CloudTrail-tutorial.html#tutorial-step2>

CloudTrail pricing: <https://aws.amazon.com/CloudTrail/pricing/>

Please note that you can analyze the data in Athena only when the Cloudtrail data is available. it takes up to 15 minutes for events to get to CloudTrail, and up to 5 minutes for CloudTrail to write to S3

Step 9: Create Amazon Athena table to query the CloudTrail data

Amazon Athena is an interactive query service that makes it easy to analyze data in Amazon S3 using standard SQL. Amazon Athena is Serverless, so there is no infrastructure to manage, and you pay only for the queries that you run.

Create Amazon Athena table in any database or default database in a region where your hub account S3 Data Lake bucket resides.

If you are using Athena first time follow the steps below to create database.

<https://docs.aws.amazon.com/athena/latest/ug/getting-started.html>

Open the Athena built-in query editor, copy below query, modify as suggested below, and run the query.

In the `LOCATION` and `storage.location.template` clauses, replace the bucket with CloudTrail Bucket, account-id with your account's account ID, and aws-region with region where CloudTrail bucket is located. For `projection.timestamp.range`, replace 2020/01/01 with the starting date that you want to use .

After successful execution of the query, you will see the `CloudTrail_logs` table created in Athena.

```
CREATE EXTERNAL TABLE cloudtrail_logs_region(  
  eventVersion STRING,  
  userIdentity STRUCT<  
    type: STRING,  
    principalId: STRING,  
    arn: STRING,  
    accountId: STRING,  
    invokedBy: STRING,  
    accessKeyId: STRING,  
    userName: STRING,  
    sessionContext: STRUCT<  
      attributes: STRUCT<  
        mfaAuthenticated: STRING,  
        creationDate: STRING>,  
      sessionIssuer: STRUCT<  
        type: STRING,  
        principalId: STRING,  
        arn: STRING,  
        accountId: STRING,  
        userName: STRING>>>,  
  eventTime STRING,  
  eventSource STRING,  
  eventName STRING,  
  awsRegion STRING,  
  sourceIpAddress STRING,  
  userAgent STRING,  
  errorCode STRING,  
  errorMessage STRING,  
  requestParameters STRING,  
  responseElements STRING,  
  additionalEventData STRING,  
  requestId STRING,  
  eventId STRING,  
  readOnly STRING,  
  resources ARRAY<STRUCT<  
    arn: STRING,  
    accountId: STRING,  
    type: STRING>>,  
  eventType STRING,
```

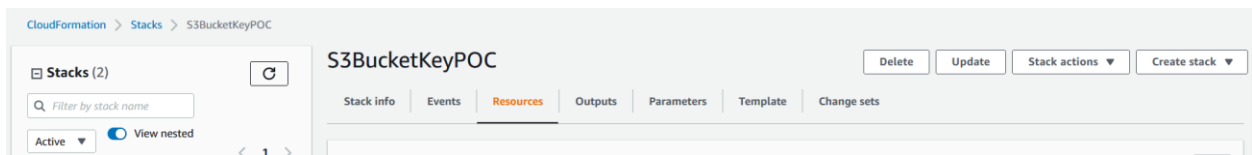
```

    apiVersion STRING,
    recipientAccountId STRING,
    serviceEventDetails STRING,
    sharedEventID STRING,
    vpcEndpointId STRING
  )
PARTITIONED BY (
  `timestamp` string)
ROW FORMAT SERDE 'com.amazon.emr.hive.serde.CloudTrailSerde'
STORED AS INPUTFORMAT 'com.amazon.emr.cloudtrail.CloudTrailInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  's3://bucket/AWSLogs/account-id/CloudTrail/aws-region'
TBLPROPERTIES (
  'projection.enabled'='true',
  'projection.timestamp.format'='yyyy/MM/dd',
  'projection.timestamp.interval'='1',
  'projection.timestamp.interval.unit'='DAYS',
  'projection.timestamp.range'='2020/01/01,NOW',
  'projection.timestamp.type'='date',
  'storage.location.template'='s3://bucket/AWSLogs/account-id/CloudTrail/aws-
region/${timestamp}')

```

Step 10: Run the Lambda function to generate the traffic to S3 Bucket which intern will generate KMS API calls.

- Go to Cloudformation → Stacks → Select S3 Bucket Key POC stack → Resource



- Click on Lambda function

Stack Info	Events	Resources	Outputs	Parameters	Template	Change sets
Resources (8)						
Search resources						
Logical ID	Physical ID	Type	Status	Status reason	Module	
CFNIAMRole	██████████	AWS::IAM::Role	CREATE_COMPLETE	-	-	
CFNKMSKey	██████████	AWS::KMS::Key	CREATE_COMPLETE	-	-	
CFNKMSKey2	██████████	AWS::KMS::Key	CREATE_COMPLETE	-	-	
CFNS3Bucket	rajeabh-0462	AWS::S3::Bucket	CREATE_COMPLETE	-	-	
CFNS3Bucket2	rajeabh-0462-s3bucketkey	AWS::S3::Bucket	CREATE_COMPLETE	-	-	
CFNS3BucketPolicy	S3BucketKeyPOC-CFNS3BucketPolicy-144N03BMS4HE9	AWS::S3::BucketPolicy	CREATE_COMPLETE	-	-	
CFNS3BucketPolicy2	S3BucketKeyPOC-CFNS3BucketPolicy2-1JLRR1MMCBKQV	AWS::S3::BucketPolicy	CREATE_COMPLETE	-	-	
MyLambdaFunction	s3_bucket_test_simulator	AWS::Lambda::Function	CREATE_COMPLETE	-	-	

Click Here

- Run the s3_bucket_test_simulator Lambda function

s3_bucket_test_simulator

Throttle Copy ARN Actions

This function belongs to an application. [Click here](#) to manage it.

Function overview

s3_bucket_test_simulator

Layers (0)

+ Add trigger

+ Add destination

Description

-

Last modified

19 minutes ago

Function ARN

arn:aws:lambda:us-east-1:██████████

Application

S3BucketKeyPOC

Code Test Monitor Configuration Aliases Versions

Code source

Upload from

File Edit Find View Go Tools Window Test Deploy

Go to Anything (Ctrl-P)

Environment

s3_bucket_test_simulator

index.py

Execution results

Test Event Name

Test

Response

null

Function Logs

uploading File -->tripdata_48.csv

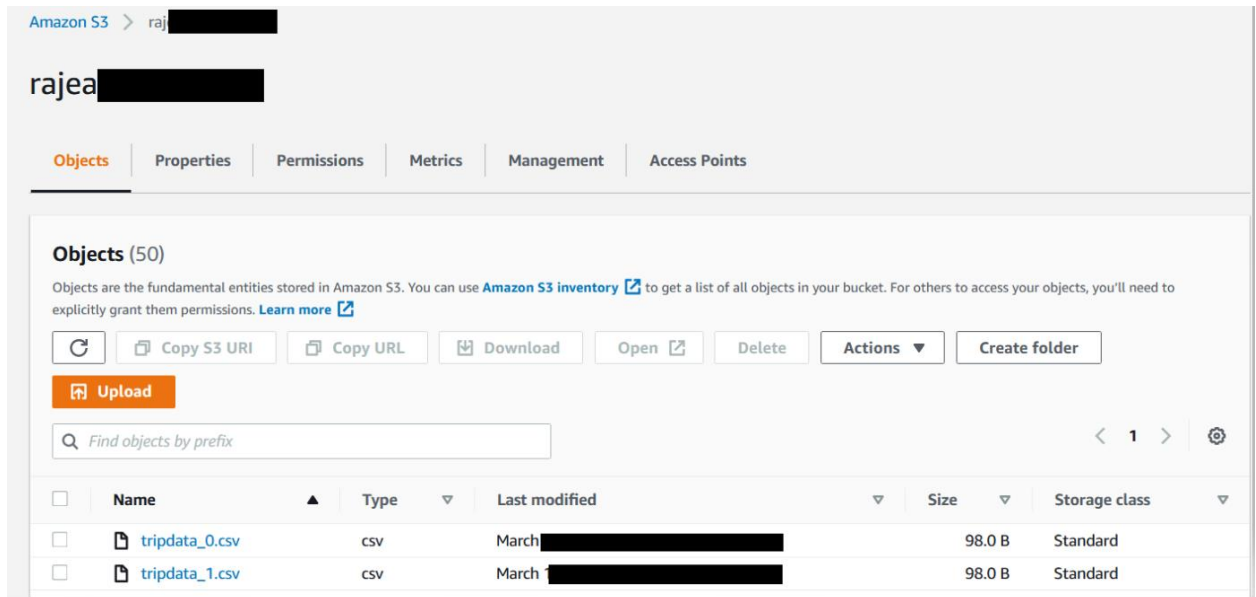
uploading File -->tripdata_49.csv

uploading File -->tripdata_49.csv

uploading File -->tripdata_49.csv

Status: Succeeded Max memory used: 70 MB Time: 15782.41 ms

Once you run the lambda function you will see traffic generated on S3 Bucket. You will 50 test files are uploaded in both the S3 Buckets.



Step 11: See the difference of KMS API call reduction when S3 bucket Key is ENABLED.

Go to Athena query Editor and run the following query. Replace “KMSKeyARN1-Value” and “KMSKeyARN2-value” generated from Cloud formation template. Change the timestamp between as appropriate

```
SELECT
    resources[1].arn as key_arn,
    count(resources) as count
FROM cloudtrail_logs_region
WHERE eventsource='kms.amazonaws.com'
    AND timestamp between '2021/04/01' and '2022/12/31'
    AND eventname in ('Decrypt','Encrypt','GenerateDataKey')
    AND json_extract(json_extract(requestparameters , '$.encryptionContext'),'$.aws:s3:arn') is not
null
AND resources[1].arn IN (KMSKeyARN1-Value ',' KMSKeyARN2-value')
GROUP BY resources[1].arn
ORDER BY key_arn,count desc
```

Result Show demonstrate that by uploading 50 file objects and by downloading 100 files objects on both SSE-KMS encrypted bucket .

S3 Bucket with Bucket Key **Enabled** made 4 KMS API calls
S3 Bucket with Bucket Key **Disabled** has 150 KMS API calls

In this example there is **97.33% reduction in KMS API calls**.

The screenshot displays the Amazon Athena console interface. At the top, there are tabs for 'Query 1' and 'Query 2'. The SQL query is as follows:

```
1 SELECT
2     resources[1].arn as key_arn,
3     count(resources) as count
4 FROM cloudtrail_logs_region
5 WHERE eventsource='kms.amazonaws.com'
6     AND timestamp between '2021/04/01' and '2022/12/31'
7     AND eventname in ('Decrypt','Encrypt','GenerateDataKey')
8     AND json_extract(json_extract(requestparameters, '$.encryptionContext'), '$.aws:s3:arn') is not null
9 AND resources[1].arn IN ('arn:aws:kms:us-east-1:756[REDACTED]', 'arn:aws:kms:us-east-1:756[REDACTED]')
10 GROUP BY resources[1].arn
11 ORDER BY key_arn,count desc
12
```

Below the query editor, there are buttons for 'Run again', 'Cancel', 'Save', 'Clear', and 'Create'. The status bar indicates 'Completed' with a green checkmark, and performance metrics: 'Time in queue: 0.153 sec', 'Run time: 1.348 sec', and 'Data scanned: 65.46 KB'.

The 'Results (2)' section shows a table with two columns: 'key_arn' and 'count'. The results are as follows:

#	key_arn	count
1	arn:aws:kms:us-east-1:756[REDACTED]	4
2	arn:aws:kms:us-east-1:756[REDACTED]	150

Phase 3: Clean Up

- **Step1: Empty S3 Buckets**

Empty both the S3 buckets that were created as part of Step-3

1. Sign in to the AWS Management Console and open the Amazon S3 console at <https://console.aws.amazon.com/s3/>
2. In the Bucket name list, select the option next to bucket name (that you entered as part of **Step 3**) and then choose Empty.
3. On the Empty bucket page, confirm that you want to empty the bucket by entering the bucket name into the text field, and then choose Empty.
4. Monitor the progress of the bucket emptying process on the Empty bucket: Status page.
5. Follow 1-4 for other S3 bucket

- **Step 2: Delete the Cloudformation Stack**

6. Open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation>
7. On the Stacks page in the CloudFormation console, select the stack **S3BucketKeyPOC** to delete.
8. In the stack details pane, choose Delete.
9. Select Delete stack when prompted.

- **Step 3: Delete the CloudTrail created**

If you have **created CloudTrail specifically for this solution**, you can delete the CloudTrail by following the instructions provided in link.

<https://docs.aws.amazon.com/awsCloudTrail/latest/userguide/CloudTrail-delete-trails-console.html>

- **Step 4: Drop the Amazon Athena table**

1. Log in to the Amazon Athena console and run the following drop table query:
2. Drop table < CloudTrail_logs_aws_region>