AWS GOV to Sentinel (using Logstash)

Last updated by | Edgar Concepcion | Jan 25, 2021 at 1:58 PM EST

AWS to Sentinel (using Logstash)

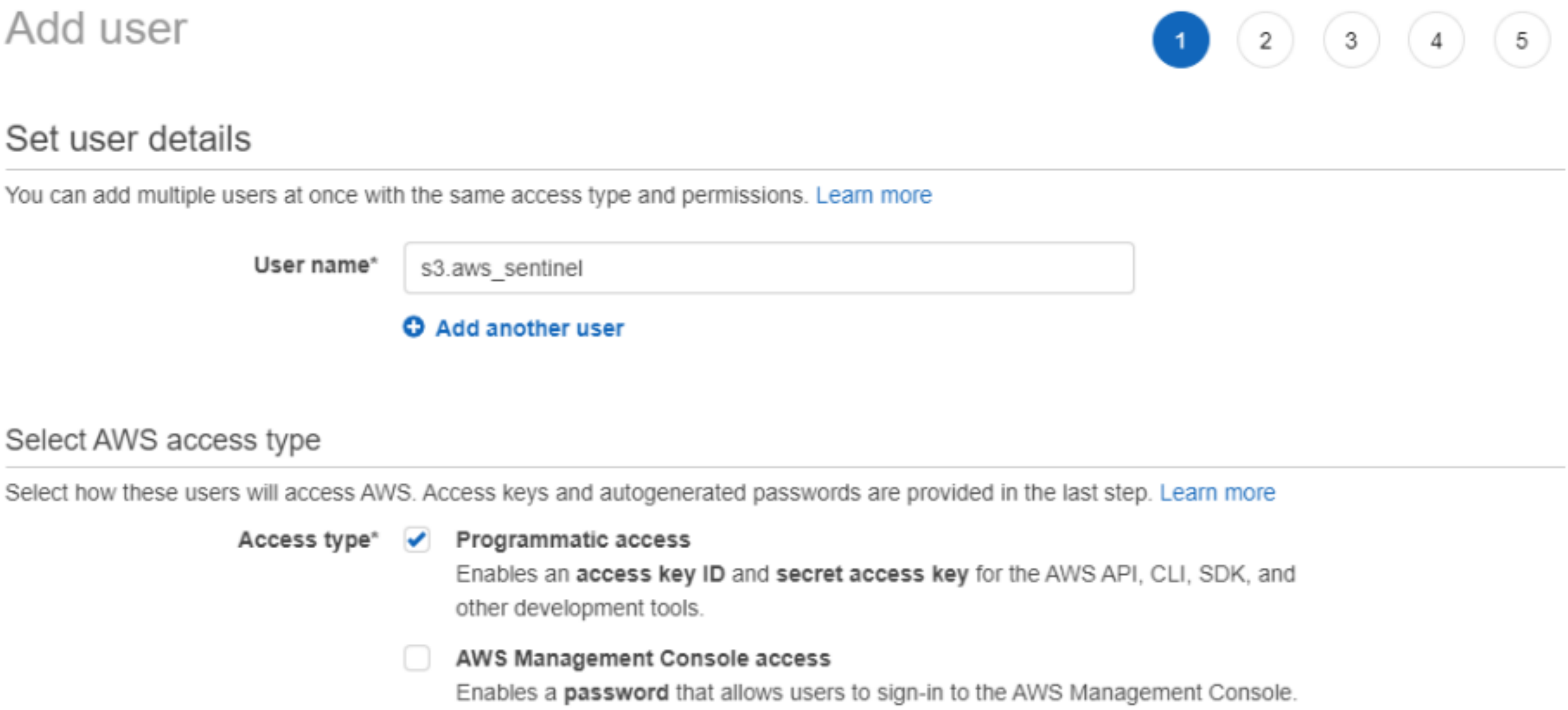
Goal:

This tutorial will go through the step-by-step process of ignesting a client's AWS CloudTrail and GuardDuty logs to a Sentinel instance.

Set up AWS

1. Log into Gov site: https://console.amazonaws-us-gov.com
2. Create a new user account
    - Go to IAM
    - Under **Access Management** click **Users**
    - At the top of the page click **Add user**

- You'll get to the add user page. Fill it in accordingly:



Add user

Set user details

You can add multiple users at once with the same access type and permissions. Learn more

User name* s3.aws_sentinel

Add another user

Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. Learn more

Access type*

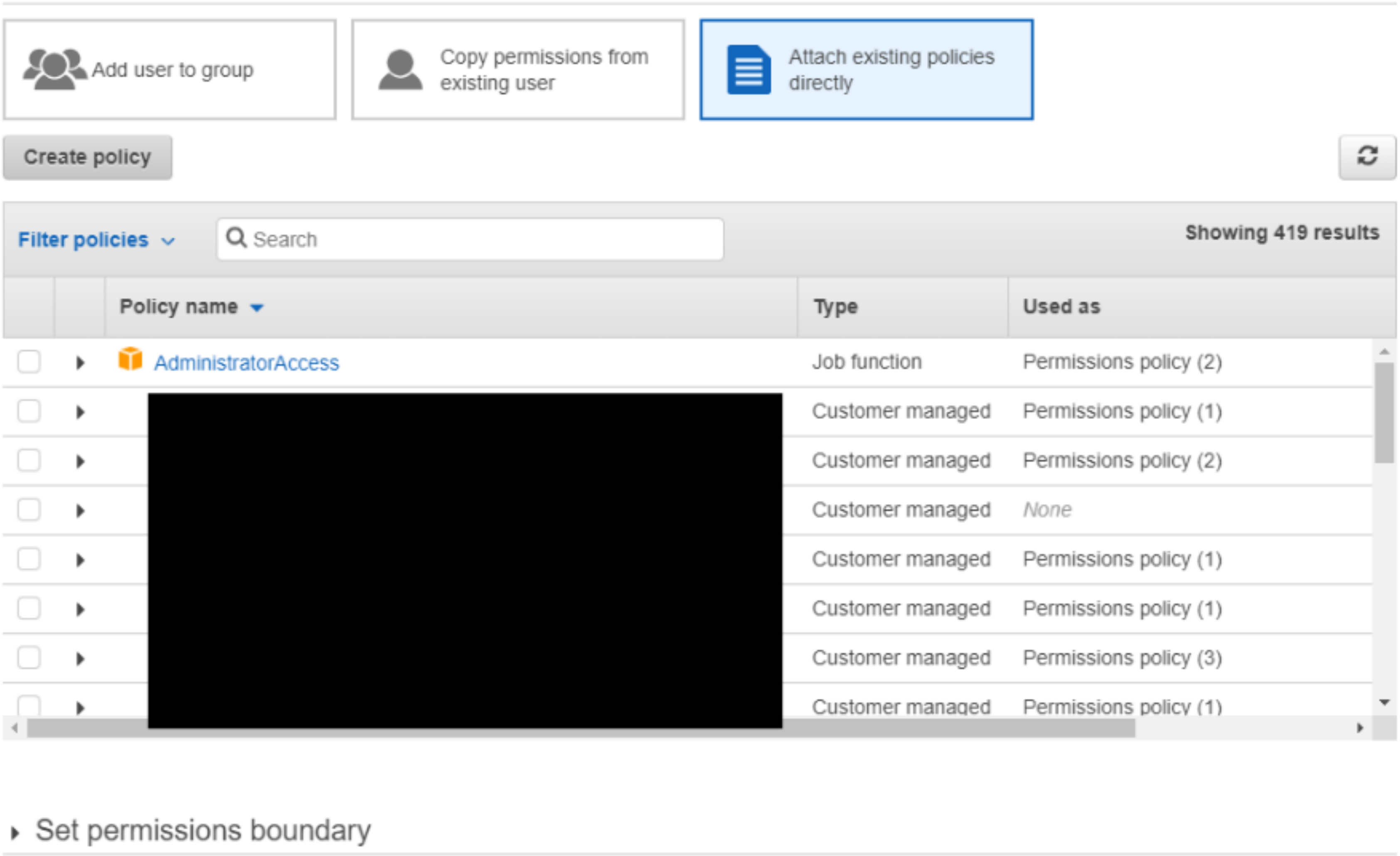
☑ **Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

☐ **AWS Management Console access**
Enables a **password** that allows users to sign-in to the AWS Management Console.

- Click Next. You'll be taken to the Permissions page
- Click on **Attach existing policies directly** then **Create policy**

Add user

Set permissions

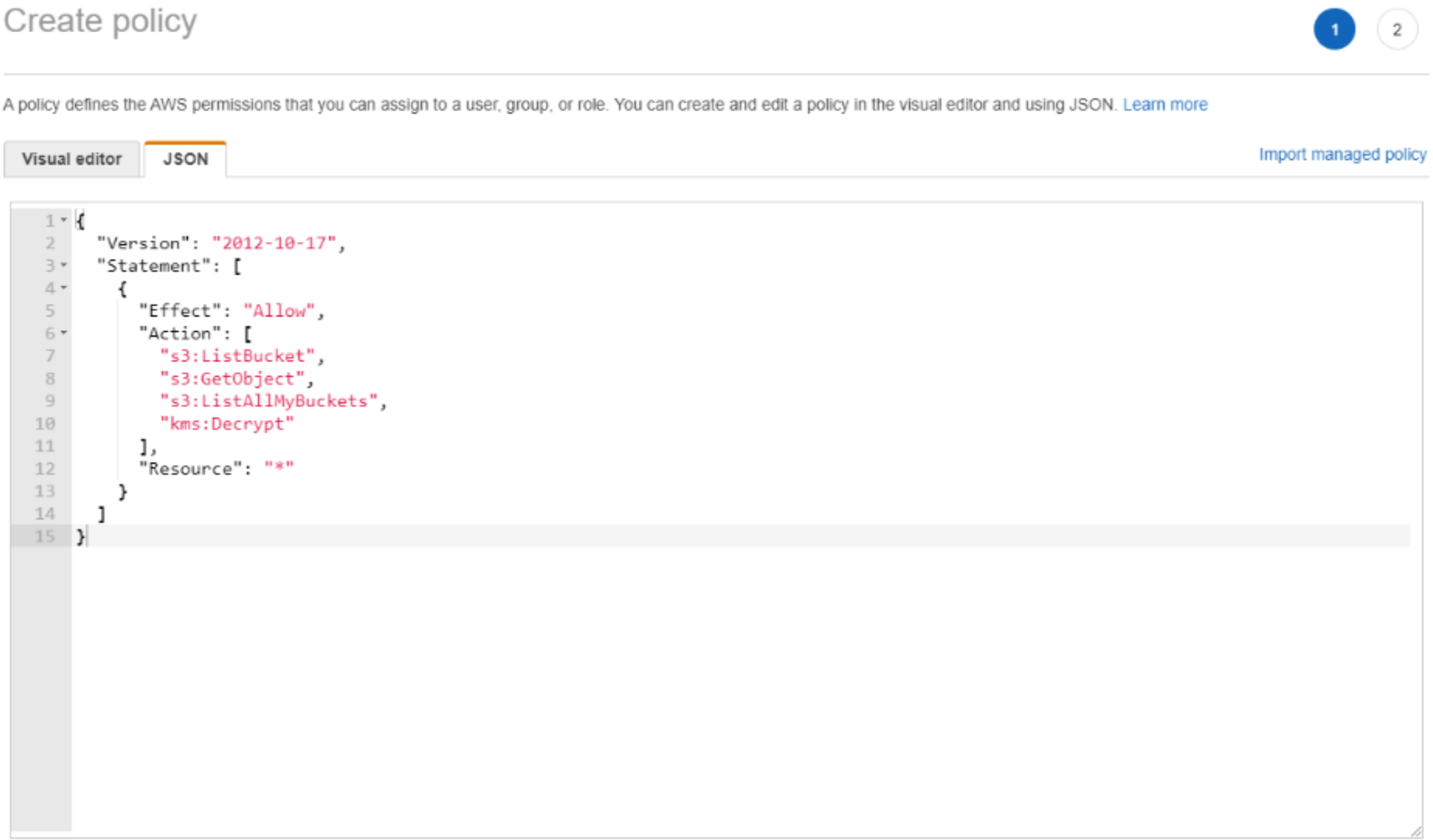


Showing 419 results

		Policy name	Type	Used as
□	▸	AdministratorAccess	Job function	Permissions policy (2)
□	▸	[redacted]	Customer managed	Permissions policy (1)
□	▸	[redacted]	Customer managed	Permissions policy (2)
□	▸	[redacted]	Customer managed	*None*
□	▸	[redacted]	Customer managed	Permissions policy (1)
□	▸	[redacted]	Customer managed	Permissions policy (1)
□	▸	[redacted]	Customer managed	Permissions policy (3)
□	▸	[redacted]	Customer managed	Permissions policy (1)

- A new tab will open. Which takes you to the **Create policy** page
- Create the new policy as JSON and paste the following JSON:

```
{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "s3:ListBucket", "s3:GetObject", "s3:ListAllMyBuckets",
"kms:Decrypt" ], "Resource": "*" } ] }
```



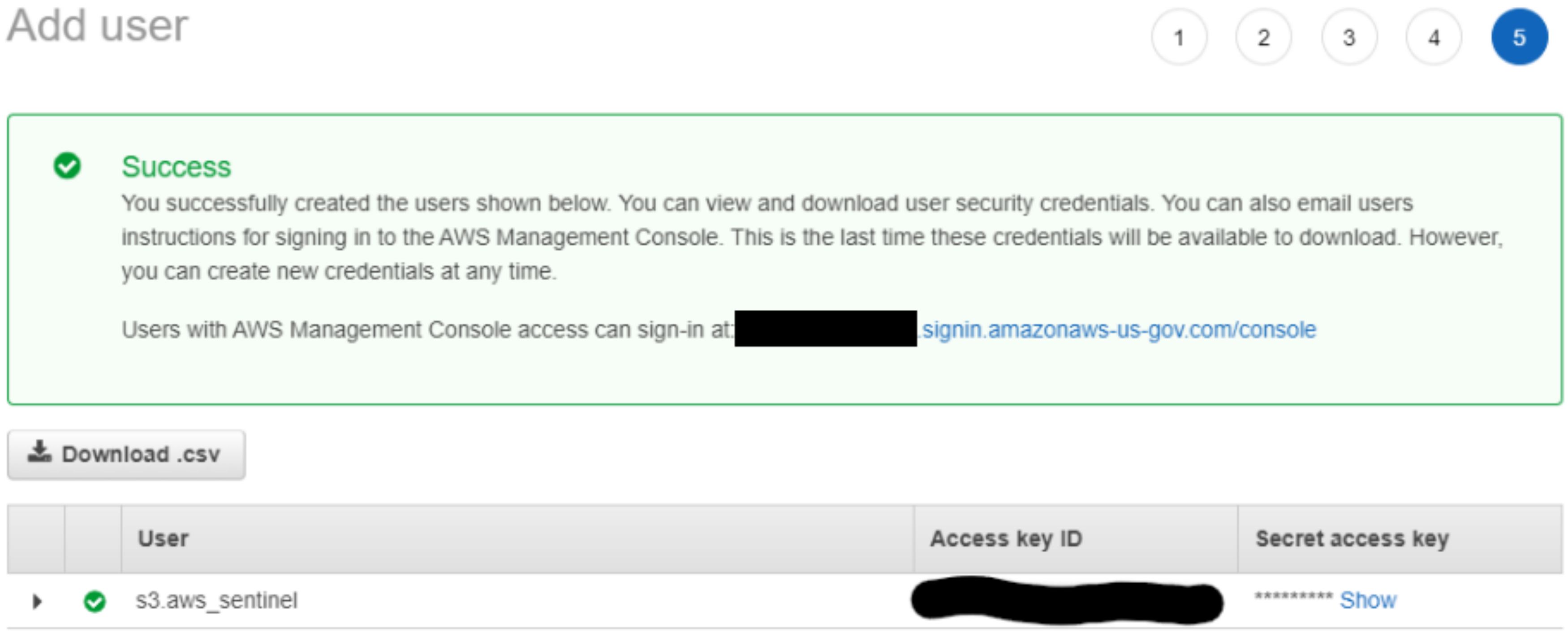
Create policy

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. Learn more

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetObject",
        "s3:ListAllMyBuckets",
        "kms:Decrypt"
      ],
      "Resource": "*"
    }
  ]
}
```

- Click the review policy button at the bottom of the page
- Fill in the Name and description, then create the policy

- Go back to the **Add user** page and refresh the policies so that the new one shows up
- Attach the newly created policy to the user that is being created
- Click next, you'll be taken to the tags page
- Add tags (if needed)
- Click next, you'll be takenb to the review page
- If everything looks good, click **Create user**
- You'll be taken to the credentials page for the user. MAKE SURE TO COPY THE **Access key ID** AND THE **Secret access key** in a notepad temporarily (or download the .csv):



Success

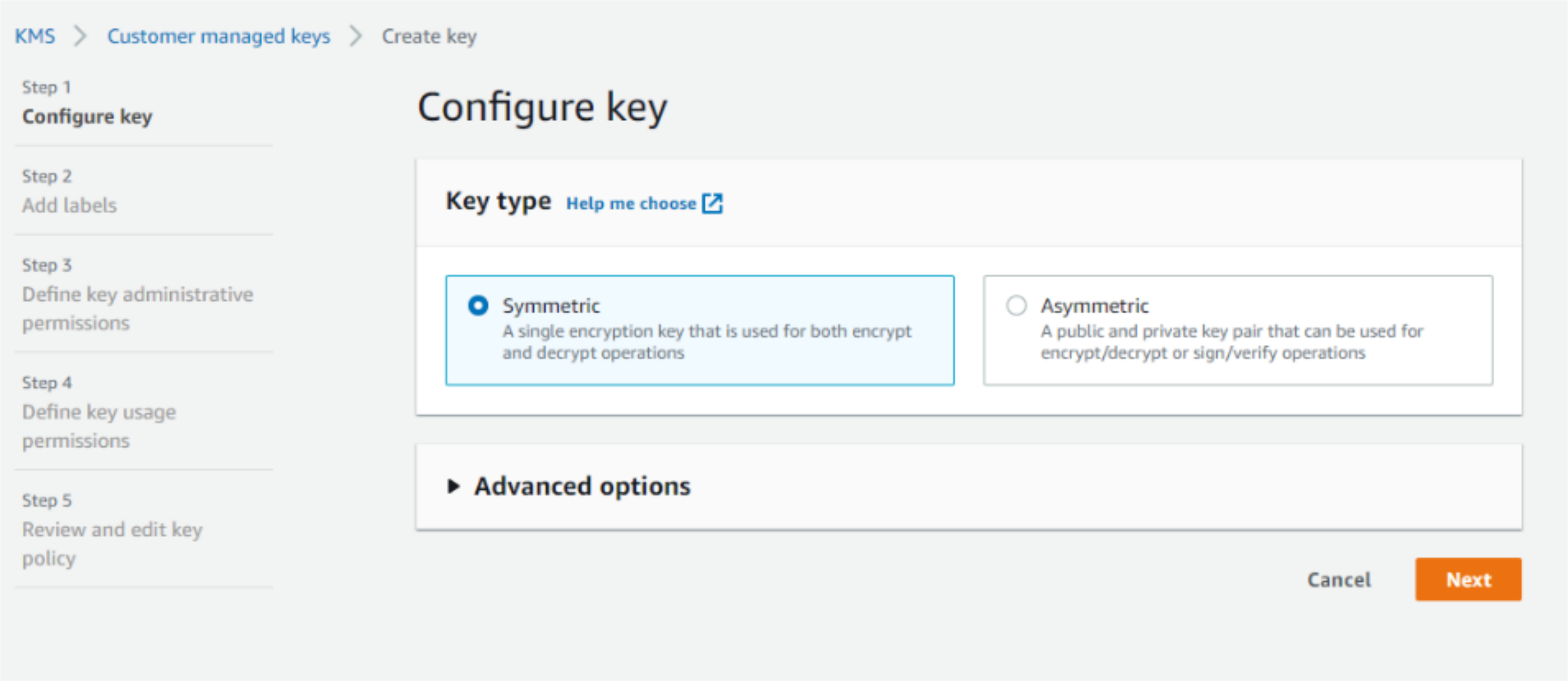
You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: [redacted].signin.amazonaws-us-gov.com/console

		User	Access key ID	Secret access key
▸	✓	s3.aws_sentinel	[redacted]	********* Show

- The user has been created

3. Create a symmetric encryption key\s\s
   We're creating a symmetric key instead of an asymmetric key because of the logstash s3 plugin. It throws errors as the documentation states that the asymmetric key is used for signing and verifying operations which are not necessary for this use case.
   - Go to KMS and click on **Create a Key**
   - Make sure **Symmetric** is selected and click next:



KMS > Customer managed keys > Create key

Step 1
Configure key

Step 2
Add labels

Step 3
Define key administrative permissions

Step 4
Define key usage permissions

Step 5
Review and edit key policy

Configure key

Key type Help me choose

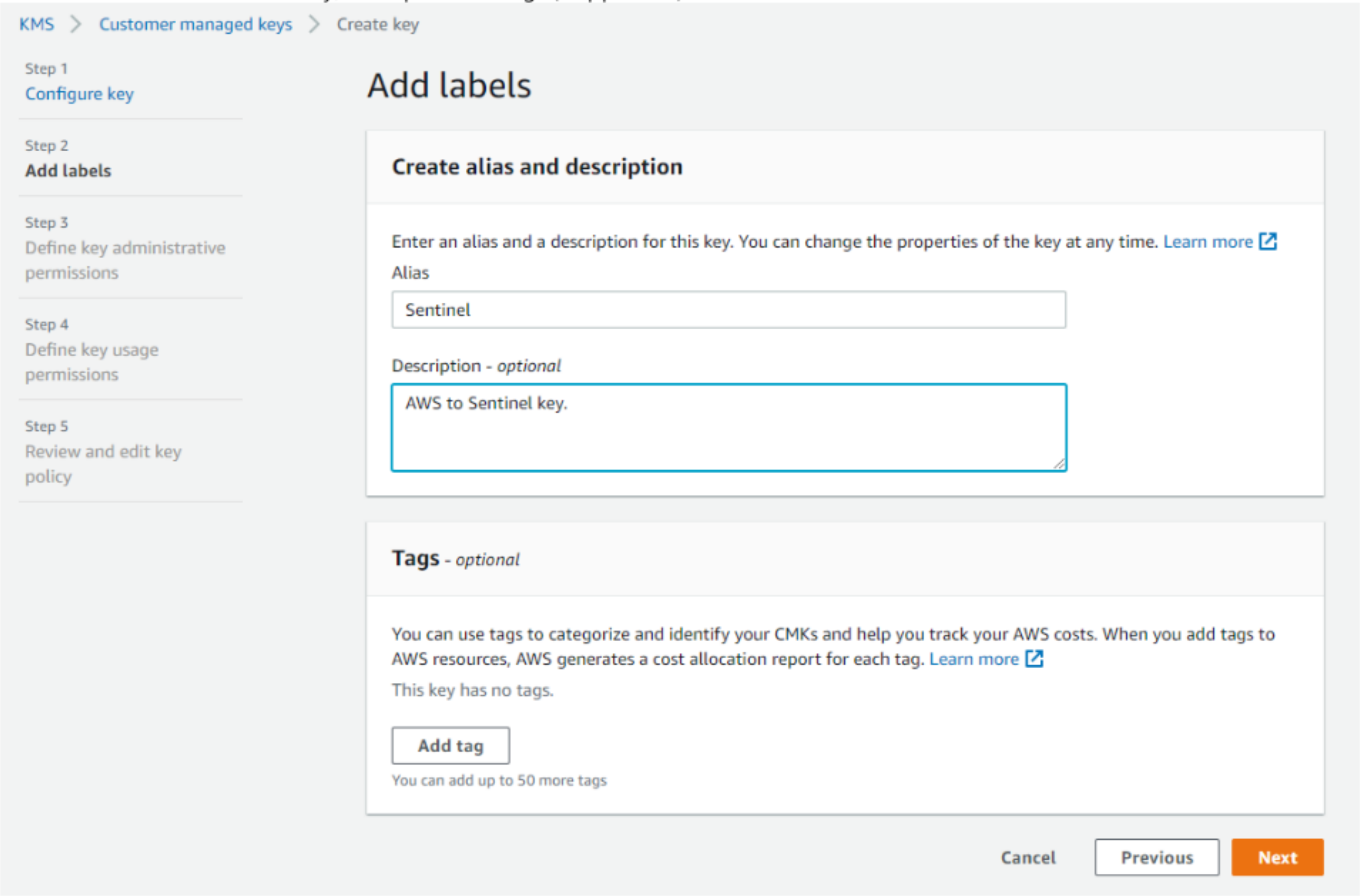
● Symmetric
A single encryption key that is used for both encrypt and decrypt operations

○ Asymmetric
A public and private key pair that can be used for encrypt/decrypt or sign/verify operations

▶ **Advanced options**

   - You'll be taken to the **Add labels** page

- Fill it out with the Alias for the key, description and tags (if applicable) then click next:



KMS > Customer managed keys > Create key
Step 1
Configure key
Step 2
Add labels
Step 3
Define key administrative permissions
Step 4
Define key usage permissions
Step 5
Review and edit key policy

Add labels

Create alias and description

Enter an alias and a description for this key. You can change the properties of the key at any time. Learn more

Alias: Sentinel

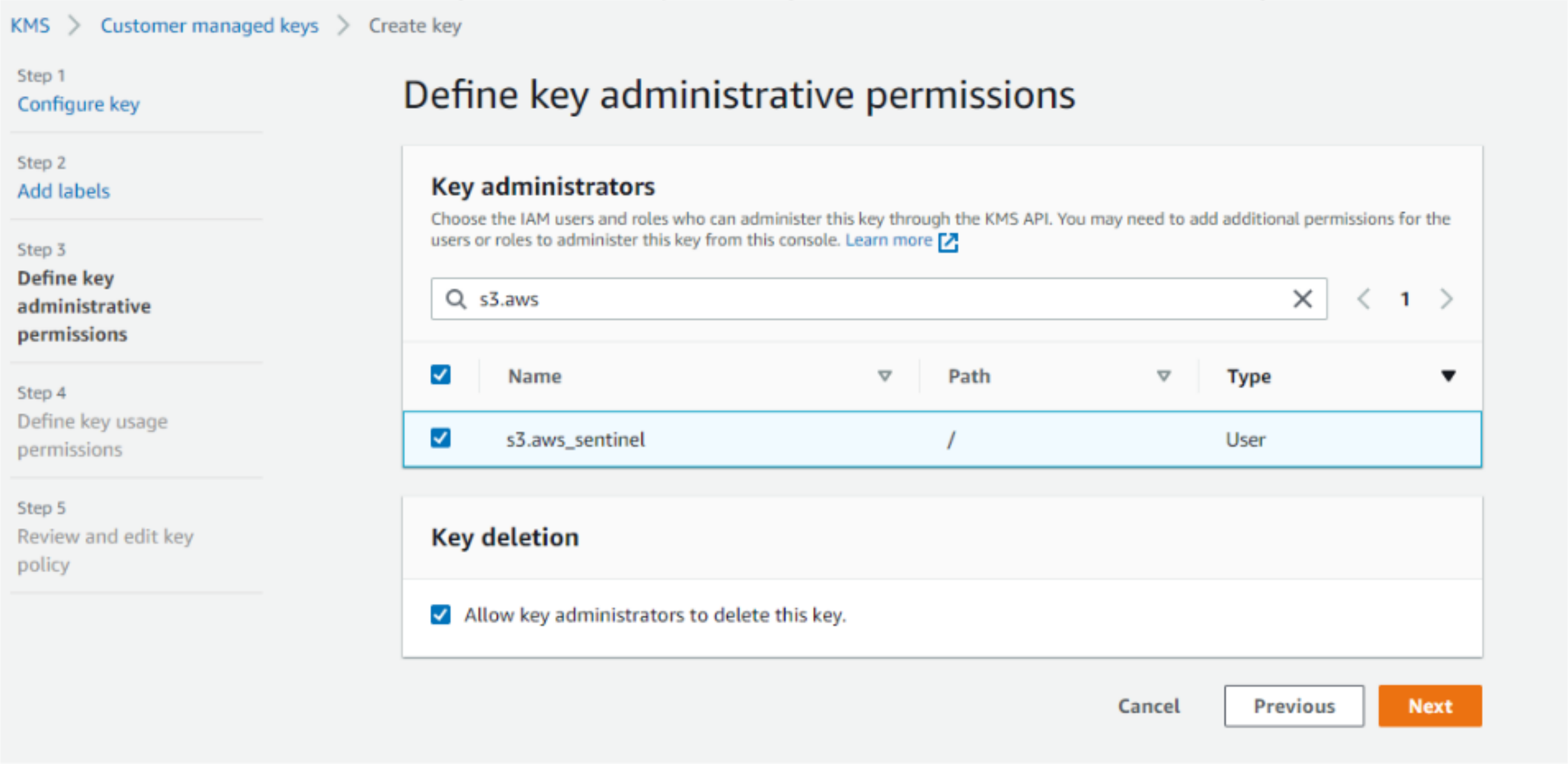
Description - *optional*: AWS to Sentinel key.

Tags - *optional*

You can use tags to categorize and identify your CMKs and help you track your AWS costs. When you add tags to AWS resources, AWS generates a cost allocation report for each tag. Learn more

This key has no tags.

- You'll be taken to the **Define key administrative permissions** page
- Here, search for and select the new user you created in step 2 (and any other account that should administer keys) then click next:



KMS > Customer managed keys > Create key

Step 1
Configure key

Step 2
Add labels

Step 3
Define key administrative permissions

Step 4
Define key usage permissions

Step 5
Review and edit key policy

Define key administrative permissions

Key administrators

Choose the IAM users and roles who can administer this key through the KMS API. You may need to add additional permissions for the users or roles to administer this key from this console. Learn more

s3.aws

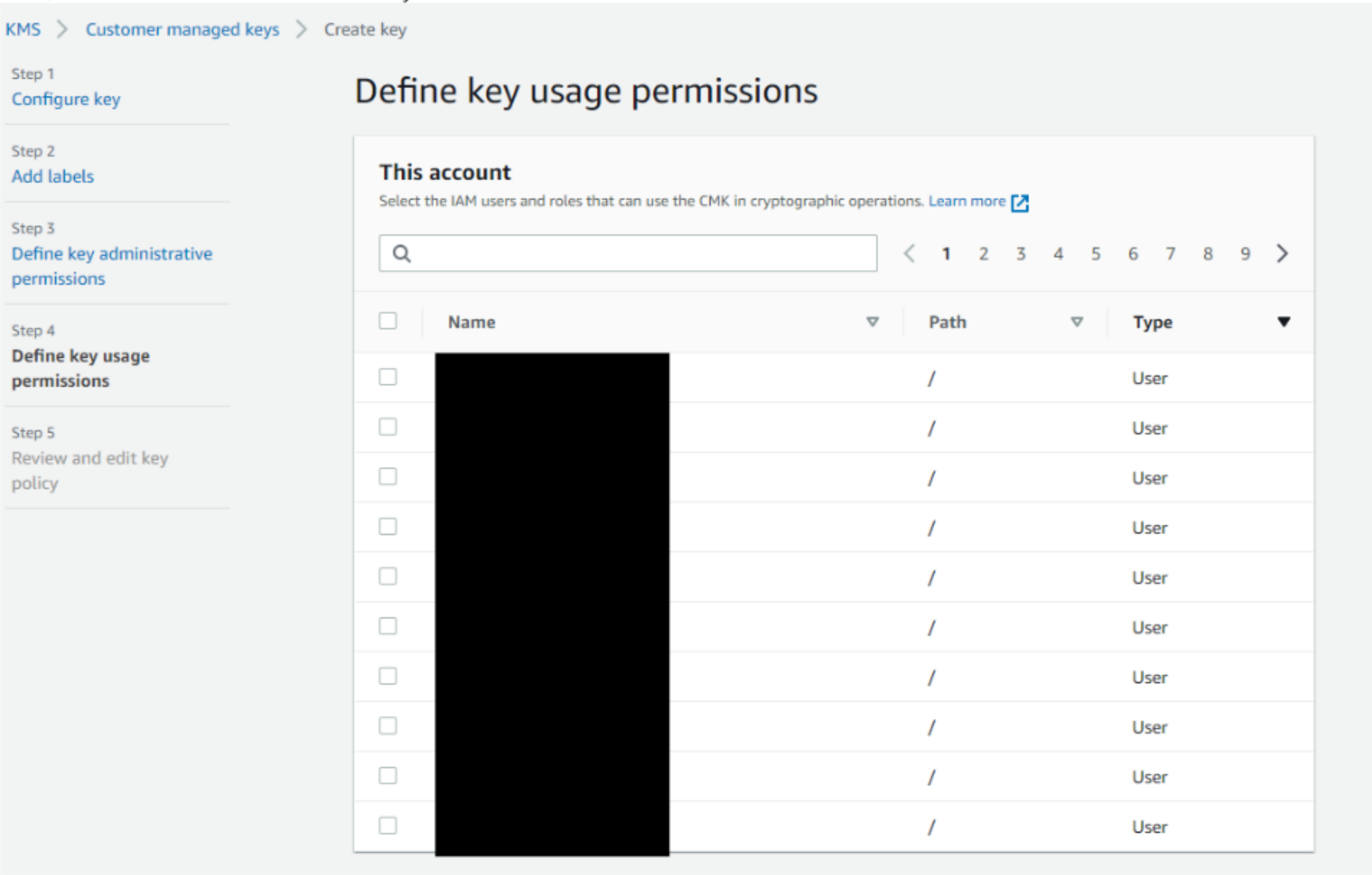
☑	Name	Path	Type
☑	s3.aws_sentinel	/	User

Key deletion

☑ Allow key administrators to delete this key.

- You'll be taken to the **Define key usage permissions**

- Here, search for and select the GuardDuty service role and click next:



KMS > Customer managed keys > Create key

Step 1
Configure key

Step 2
Add labels

Step 3
Define key administrative permissions

Step 4
Define key usage permissions

Step 5
Review and edit key policy

Define key usage permissions

This account

Select the IAM users and roles that can use the CMK in cryptographic operations. Learn more

< 1 2 3 4 5 6 7 8 9 >

□	Name	Path	Type
□	[redacted]	/	User
□	[redacted]	/	User
□	[redacted]	/	User
□	[redacted]	/	User
□	[redacted]	/	User
□	[redacted]	/	User
□	[redacted]	/	User
□	[redacted]	/	User
□	[redacted]	/	User
□	[redacted]	/	User

- You'll be taken to the **Review and edit key policy page**

- Here, paste the following two JSON formatted permissions:

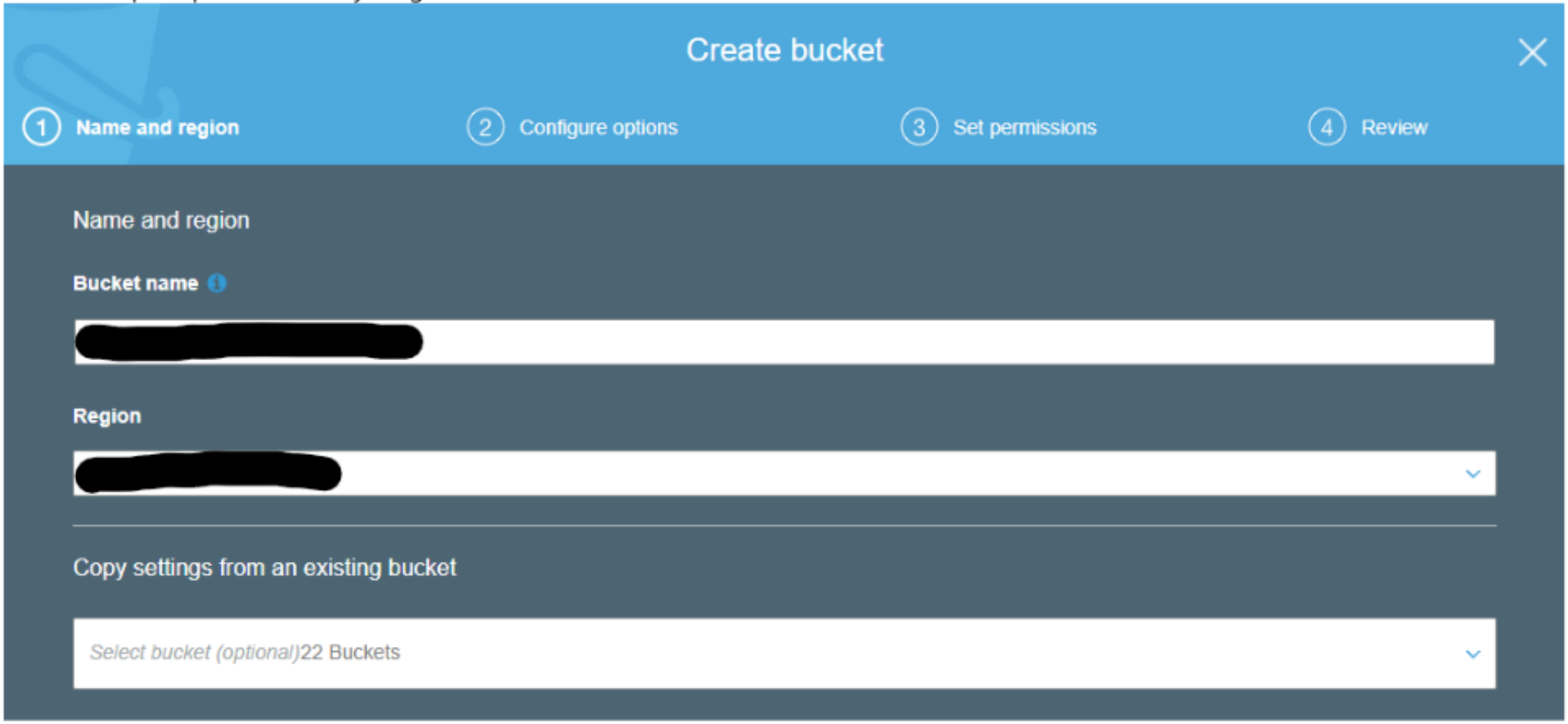
  ```
  { "Sid": "Allow GuardDuty to use the key", "Effect": "Allow", "Principal": { "Service": "guardduty.amazonaws.com" }, "Action":
  "kms:GenerateDataKey", "Resource": "*" }
  ```

  and

  ```
  { "Sid": "Allow Cloudtrail to use the key", "Effect": "Allow", "Principal": { "Service": "cloudtrail.amazonaws.com" }, "Action":
  "kms:GenerateDataKey", "Resource": "*" }
  ```

- Click **Finish**

4. Create new S3 bucket
   - Go to S3 service, then click on **Create bucket**
   - You will be prompted with everything that needs to be filled in:



Name and region

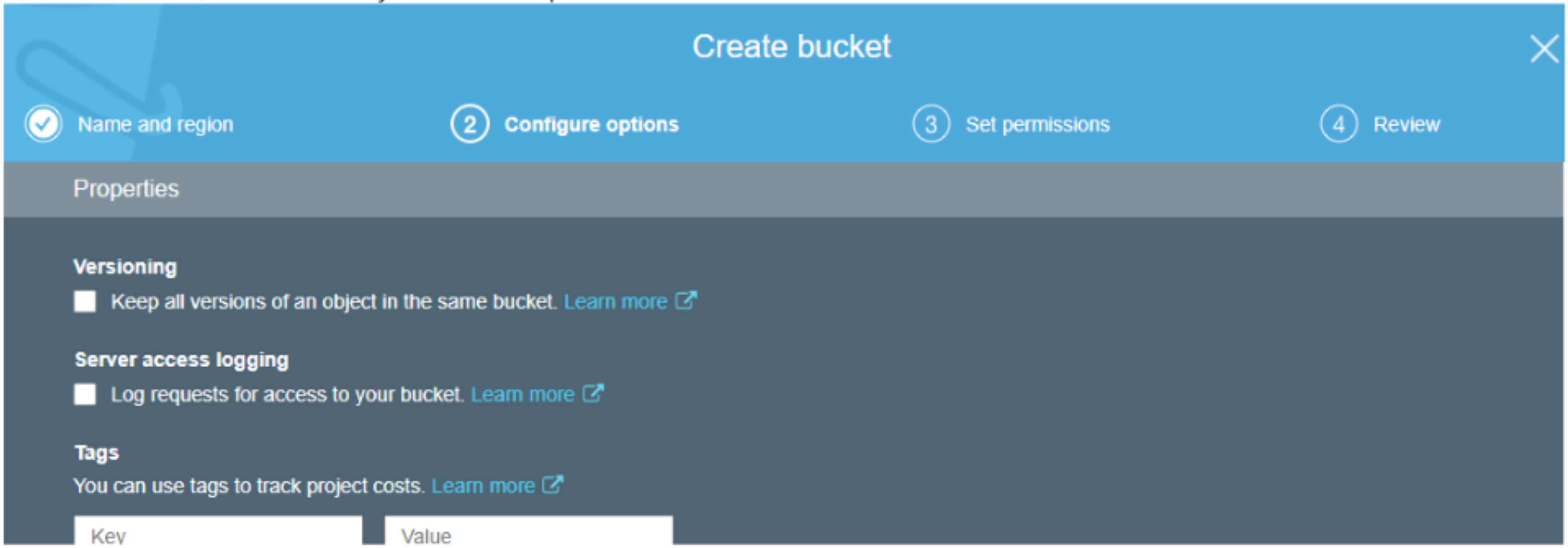
Bucket name: [redacted]

Region: [redacted]

Copy settings from an existing bucket

Select bucket (optional)22 Buckets

- Fill everything in and click next
- You will be taken to the **Properties tab**, fill everything in and enable **Default encryption**
- Select **AWS-KMS**, and use the key created in step 3



Versioning

□ Keep all versions of an object in the same bucket. Learn more

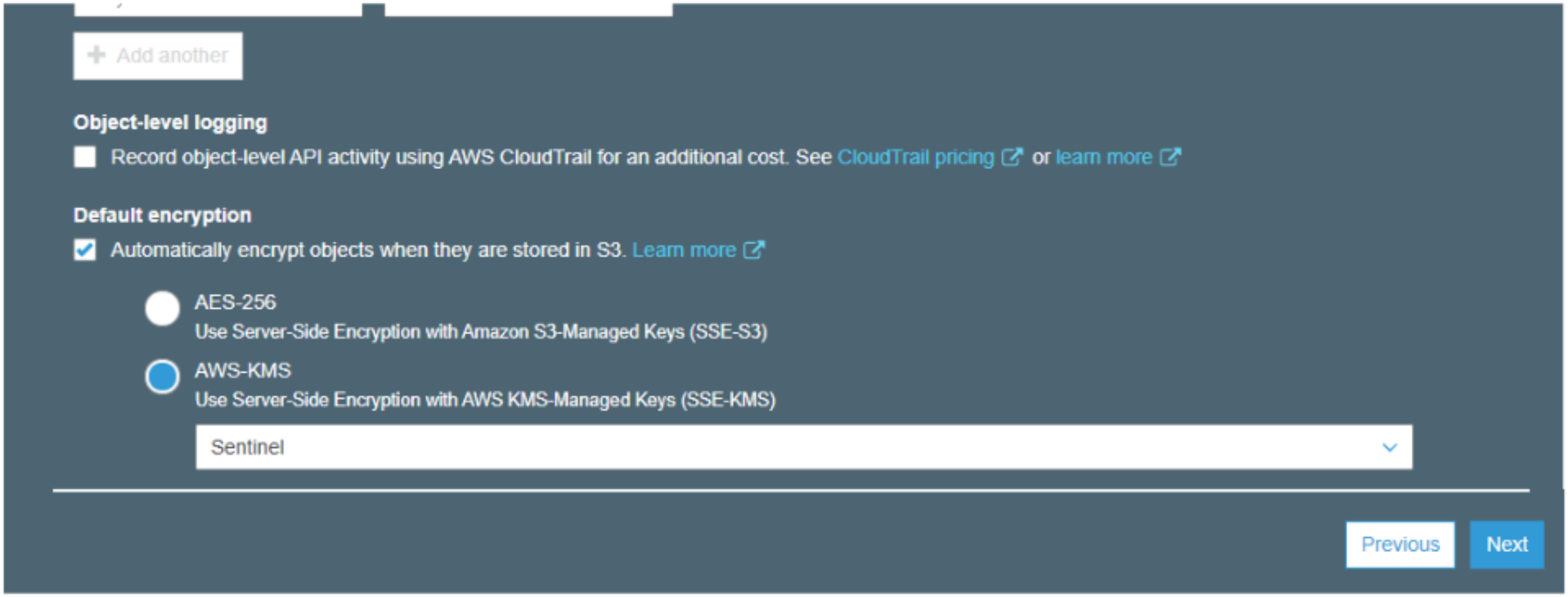
Server access logging

□ Log requests for access to your bucket. Learn more

Tags

You can use tags to track project costs. Learn more

Key | Value



Object-level logging

□ Record object-level API activity using AWS CloudTrail for an additional cost. See CloudTrail pricing or learn more

Default encryption

☑ Automatically encrypt objects when they are stored in S3. Learn more

○ AES-256
Use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)

● AWS-KMS
Use Server-Side Encryption with AWS KMS-Managed Keys (SSE-KMS)

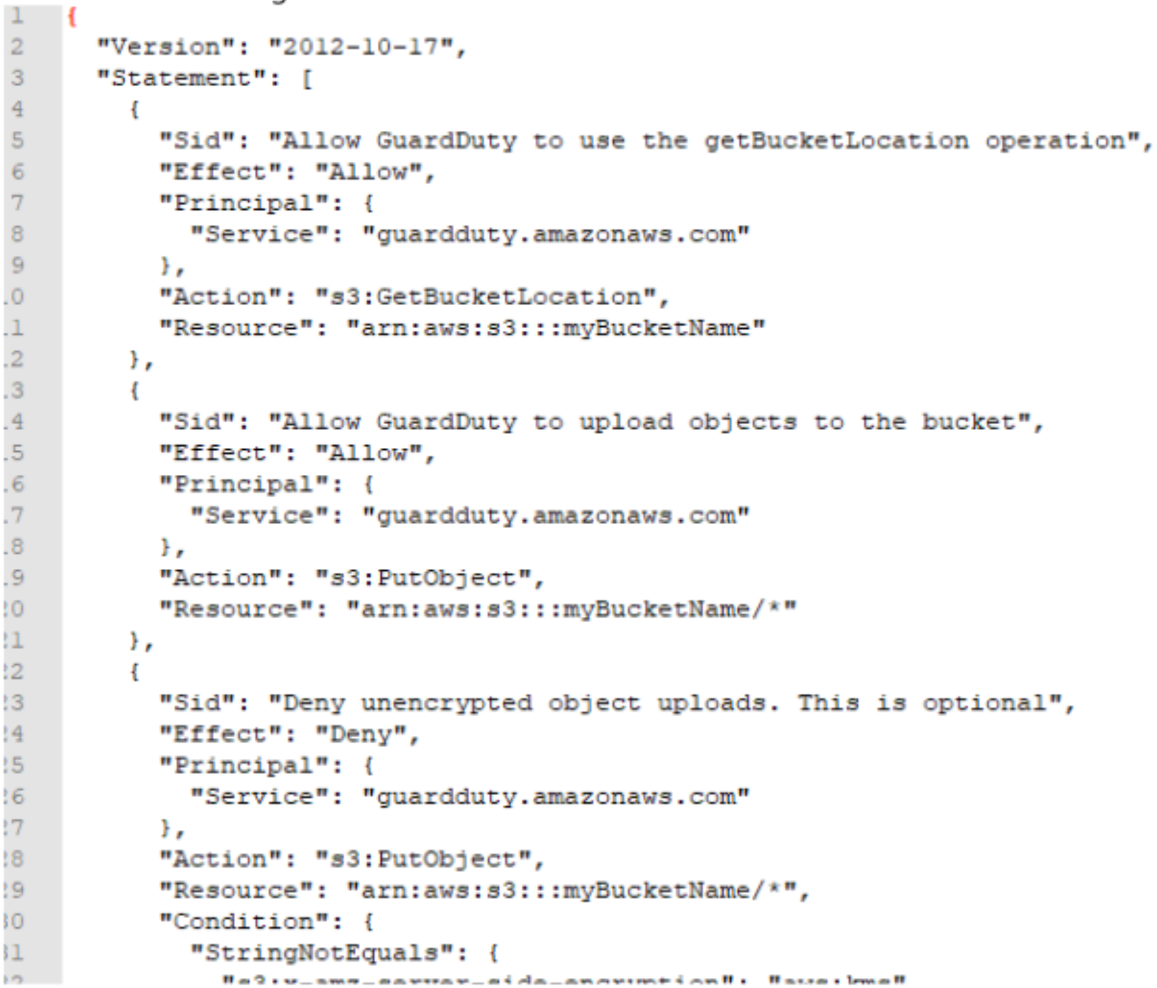
Sentinel

- Then click next, you'll be taken to the bucket settings for public access
- Leave **Block *all* public access** selected and click next
- Review the bucket permissions and click **Create bucket** if everything is fine
- Go to the new bucket that was created and click on **Permissions**
- Then click on **Bucket Policy** where we will have to add a few custom permissions for GuardDuty to be able to do all the bucket operations needed.
- Use Notepad++ or some other editor and copy the JSON currently being used for the **Bucket policy**
- Then inside of the **Statement: []** key, include the following JSON:

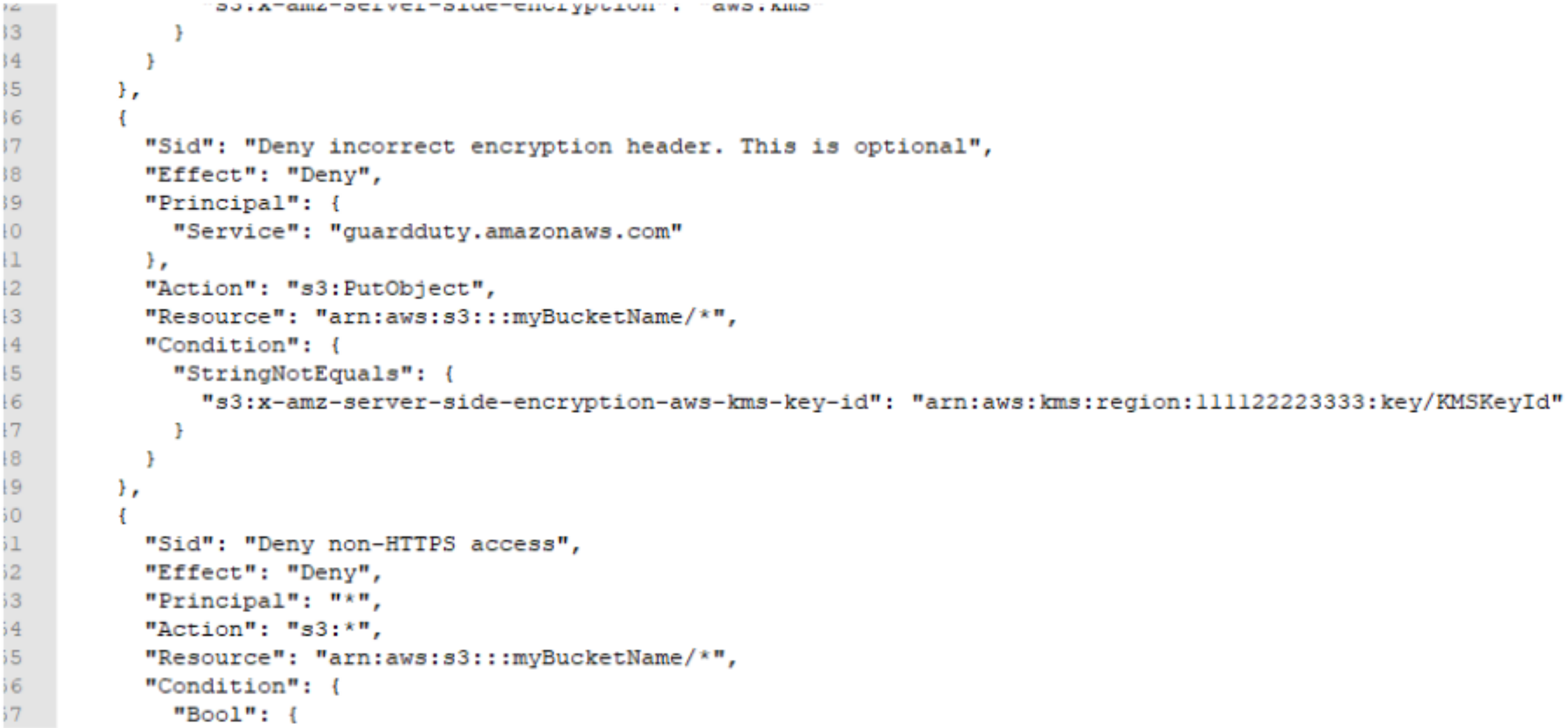
```
{ "Sid": "Allow GuardDuty to use the getBucketLocation operation", "Effect": "Allow", "Principal": { "Service": "guardduty.amazonaws.com"
}, "Action": "s3:GetBucketLocation", "Resource": "arn:aws:s3:::myBucketName" }, { "Sid": "Allow GuardDuty to upload objects to the
bucket", "Effect": "Allow", "Principal": { "Service": "guardduty.amazonaws.com" }, "Action": "s3:PutObject", "Resource":
"arn:aws:s3:::myBucketName/*" }, { "Sid": "Deny unencrypted object uploads. This is optional", "Effect": "Deny", "Principal": { "Service":
```

```
"guardduty.amazonaws.com" }, "Action": "s3:PutObject", "Resource": "arn:aws:s3:::myBucketName/*", "Condition": { "StringNotEquals": {
"s3:x-amz-server-side-encryption": "aws:kms" } } }, { "Sid": "Deny incorrect encryption header. This is optional", "Effect": "Deny",
"Principal": { "Service": "guardduty.amazonaws.com" }, "Action": "s3:PutObject", "Resource": "arn:aws:s3:::myBucketName/*", "Condition": {
"StringNotEquals": { "s3:x-amz-server-side-encryption-aws-kms-key-id": "arn:aws:kms:region:111122223333:key/KMSKeyId" } } }, { "Sid":
"Deny non-HTTPS access", "Effect": "Deny", "Principal": "*", "Action": "s3:*", "Resource": "arn:aws:s3:::myBucketName/*", "Condition": {
"Bool": { "aws:SecureTransport": "false" } } }
```

- It will look something like this:



```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow GuardDuty to use the getBucketLocation operation",
      "Effect": "Allow",
      "Principal": {
        "Service": "guardduty.amazonaws.com"
      },
      "Action": "s3:GetBucketLocation",
      "Resource": "arn:aws:s3:::myBucketName"
    },
    {
      "Sid": "Allow GuardDuty to upload objects to the bucket",
      "Effect": "Allow",
      "Principal": {
        "Service": "guardduty.amazonaws.com"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::myBucketName/*"
    },
    {
      "Sid": "Deny unencrypted object uploads. This is optional",
      "Effect": "Deny",
      "Principal": {
        "Service": "guardduty.amazonaws.com"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::myBucketName/*",
      "Condition": {
        "StringNotEquals": {
          "s3:x-amz-server-side-encryption": "aws:kms"
```

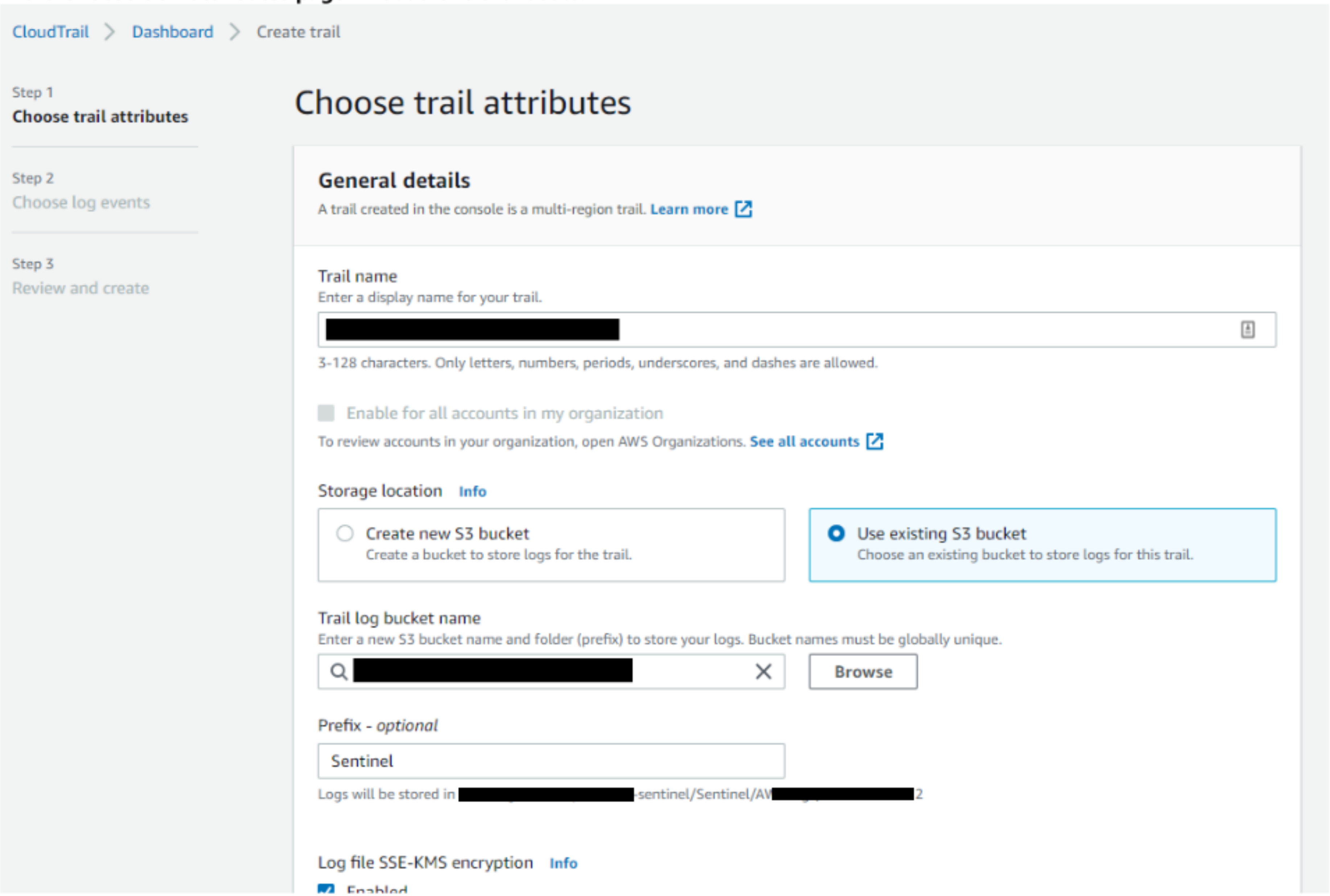


```
          "s3:x-amz-server-side-encryption": "aws:kms"
        }
      }
    },
    {
      "Sid": "Deny incorrect encryption header. This is optional",
      "Effect": "Deny",
      "Principal": {
        "Service": "guardduty.amazonaws.com"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::myBucketName/*",
      "Condition": {
        "StringNotEquals": {
          "s3:x-amz-server-side-encryption-aws-kms-key-id": "arn:aws:kms:region:111122223333:key/KMSKeyId"
        }
      }
    },
    {
      "Sid": "Deny non-HTTPS access",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::myBucketName/*",
      "Condition": {
        "Bool": {
```

- Make sure to Replace **myBucketName** with the name of the bucket that you're adding the bucket policy for
- Replace **region** with the Region that the KMS key is in
- Replace **111122223333** with the AWS account number of the account that owns the bucket
- Replace **KMSKeyId** with the key ID of the key that you chose for encryption
- Then paste the full JSON back into AWS's JSON bucket policy editor
- The end result should be around 86 lines of JSON
- Click **Save**

5. Set up CloudTrail
   - Go to the CloudTrail service and click **Create trail**

- In the **Choose trail attributes page** fill out the relevant data:



CloudTrail > Dashboard > Create trail

Step 1
Choose trail attributes

Step 2
Choose log events

Step 3
Review and create

Choose trail attributes

General details

A trail created in the console is a multi-region trail. **Learn more**

Trail name
Enter a display name for your trail.

[redacted]

3-128 characters. Only letters, numbers, periods, underscores, and dashes are allowed.

□ Enable for all accounts in my organization

To review accounts in your organization, open AWS Organizations. **See all accounts**

Storage location **Info**

○ Create new S3 bucket
Create a bucket to store logs for the trail.

● Use existing S3 bucket
Choose an existing bucket to store logs for this trail.

Trail log bucket name
Enter a new S3 bucket name and folder (prefix) to store your logs. Bucket names must be globally unique.

[redacted] ✕ **Browse**

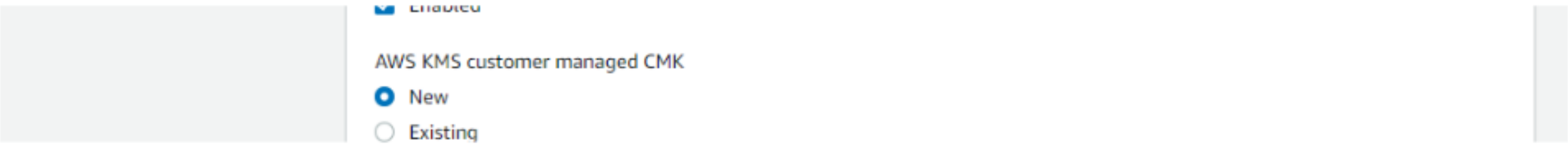
Prefix - *optional*

Sentinel

Logs will be stored in [redacted]-sentinel/Sentinel/AW[redacted]2

Log file SSE-KMS encryption **Info**

☑ Enabled



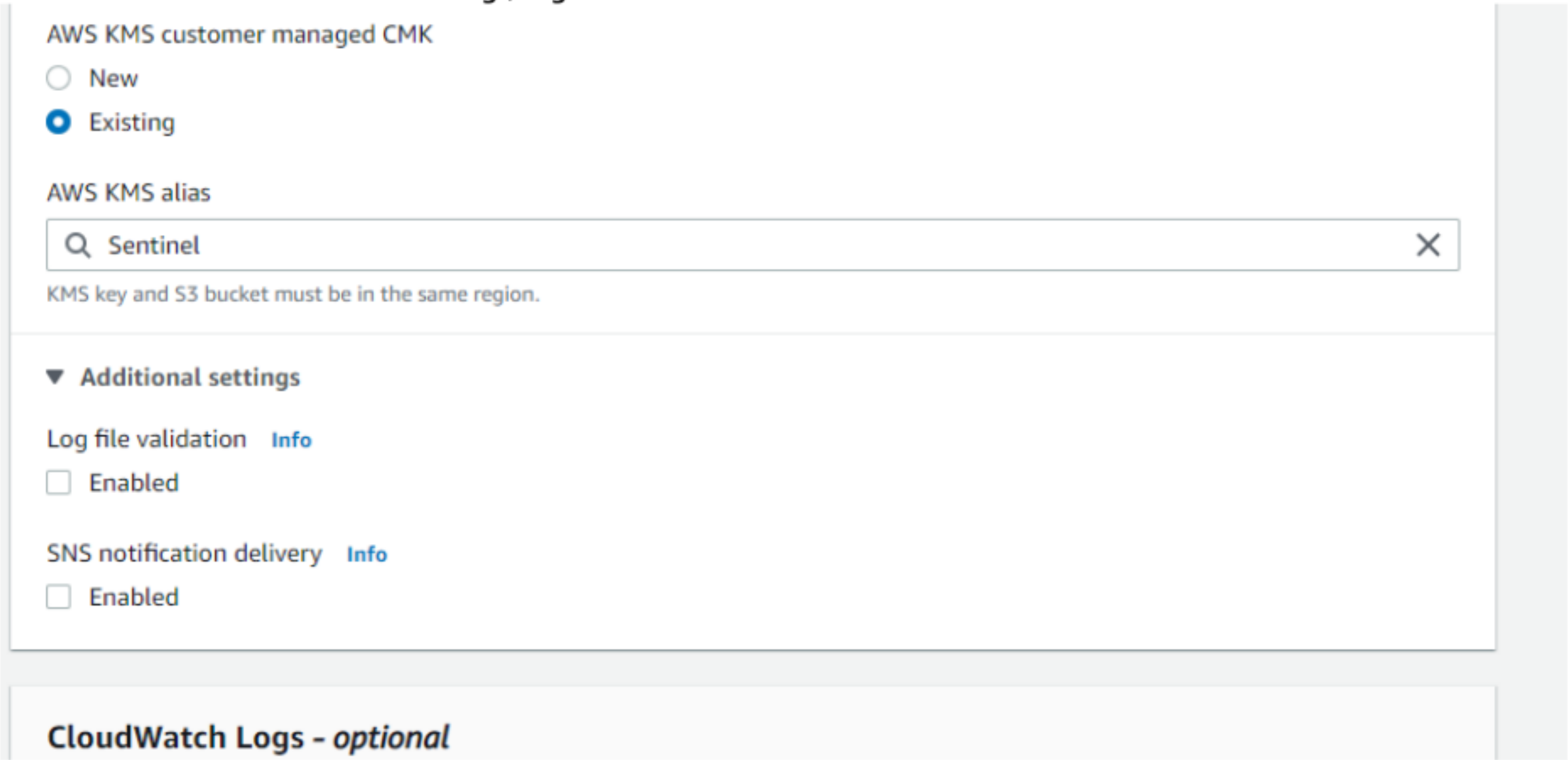
☑ Enabled

AWS KMS customer managed CMK

● New

○ Existing

- Make sure that you use the S3 bucket created in step 4
- For the **Prefix** write **Sentinel**
- Make sure that encryption is enabled
- Chooose an existing **AWS KMS customer managed CMK**, and search for the key we initially created
- Then make sure that in **Additional settings**, **Log file validation** is **disabled**



AWS KMS customer managed CMK

○ New

● Existing

AWS KMS alias

Sentinel

KMS key and S3 bucket must be in the same region.

▼ Additional settings

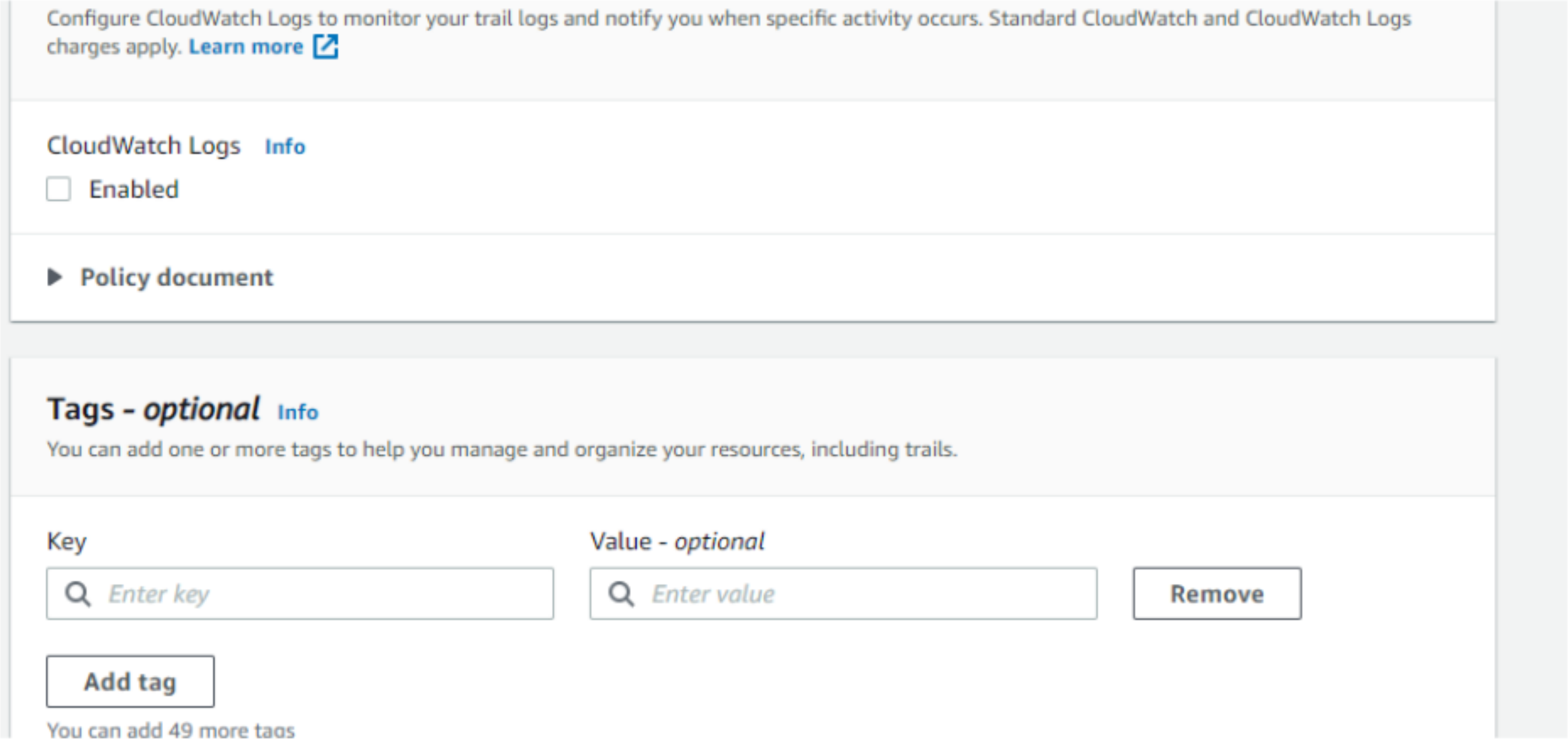
Log file validation Info

□ Enabled

SNS notification delivery Info

□ Enabled

CloudWatch Logs - *optional*



Configure CloudWatch Logs to monitor your trail logs and notify you when specific activity occurs. Standard CloudWatch and CloudWatch Logs charges apply. Learn more

CloudWatch Logs Info

□ Enabled

▶ Policy document

Tags - *optional* Info

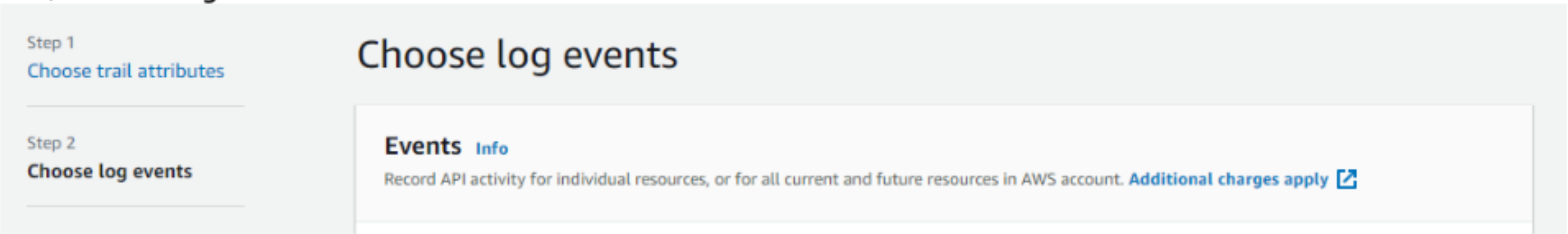
You can add one or more tags to help you manage and organize your resources, including trails.

Key: *Enter key*

Value - *optional*: *Enter value*

You can add 49 more tags

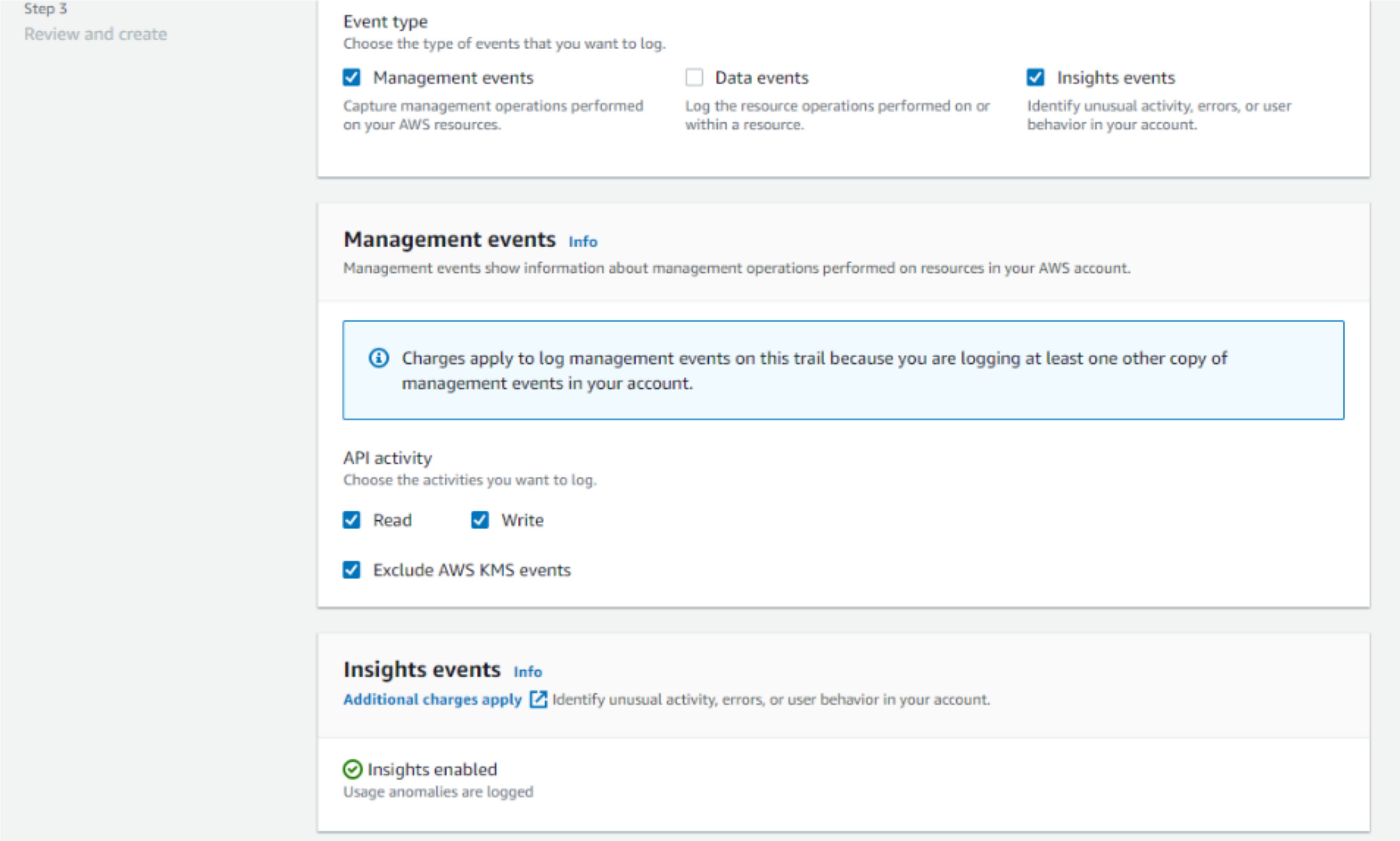
- Click next and you'll be taken to the **Choose log events** page
- Choose **Management events** and **Insight events** to be logged
- Also, under **Management events** select **Exclude AWS KMS events** and click next:



Choose log events

Events Info

Record API activity for individual resources, or for all current and future resources in AWS account. Additional charges apply



Event type

Choose the type of events that you want to log.

☑ Management events

Capture management operations performed on your AWS resources.

□ Data events

Log the resource operations performed on or within a resource.

☑ Insights events

Identify unusual activity, errors, or user behavior in your account.

Management events Info

Management events show information about management operations performed on resources in your AWS account.

ⓘ Charges apply to log management events on this trail because you are logging at least one other copy of management events in your account.

API activity

Choose the activities you want to log.

☑ Read ☑ Write

☑ Exclude AWS KMS events

Insights events Info

Additional charges apply Identify unusual activity, errors, or user behavior in your account.

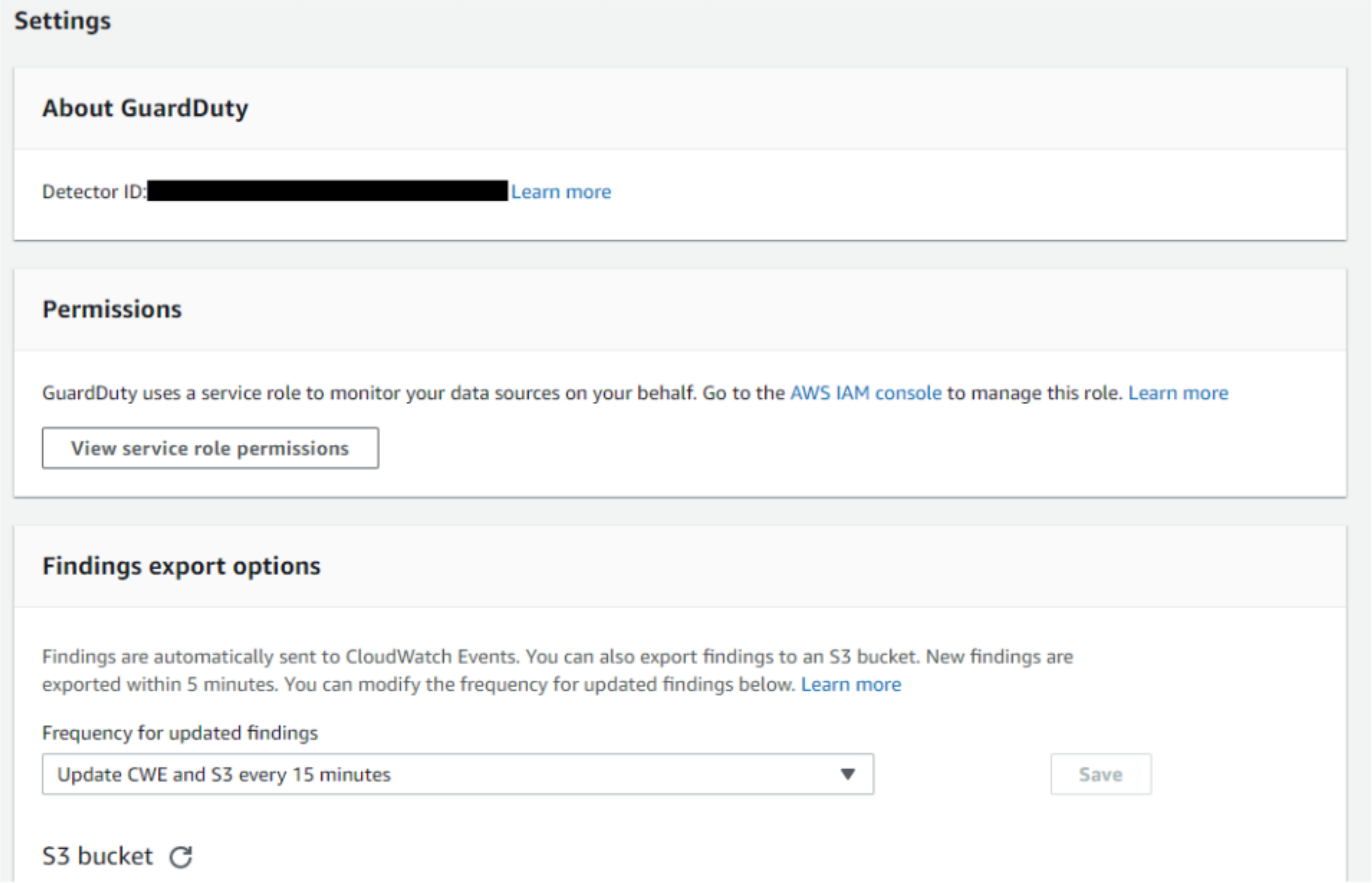
✓ Insights enabled

Usage anomalies are logged

- Review the new CloudTrail and if everything looks good, finish the creation

6. Set up GuardDuty
    - Go to GuardDuty

- On the left-hand side click **Settings**
- You will be taken to the settings where nothing should already be configured:



Settings

About GuardDuty

Detector ID: [redacted] Learn more

Permissions

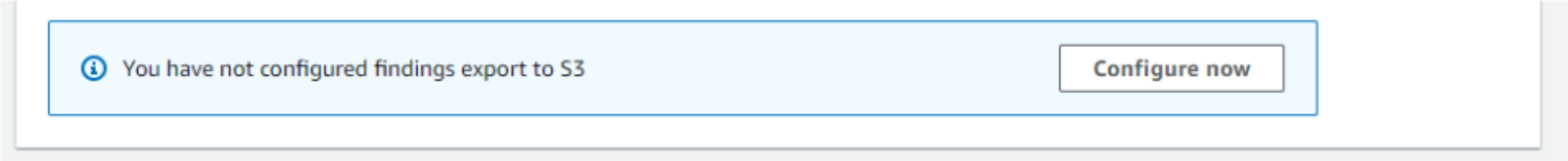
GuardDuty uses a service role to monitor your data sources on your behalf. Go to the AWS IAM console to manage this role. Learn more

Findings export options

Findings are automatically sent to CloudWatch Events. You can also export findings to an S3 bucket. New findings are exported within 5 minutes. You can modify the frequency for updated findings below. Learn more

Frequency for updated findings: Update CWE and S3 every 15 minutes

S3 bucket



You have not configured findings export to S3

   - Only change the **Finding export options**
   - In the dropdown select **Update CWE and S3 every 15 minutes**
   - In the **S3 bucket** section, click **Configure now**
   - Make sure to use the existing bucket that was created earlier
   - For the **Log file prefix** write **Sentinel**
   - KMS encryption: **choose key from your account** and search for the key created in step 3
   - Click **Save**

7. Set up CloudWatch (If CloudWatch is being used in the customer's environment)
   - **NOTES: Was able to get the requests module working with the lambda function that will send SNS messages to Sentinel.**
   - **TODO: Upload custom lambda function.zip, which has python environment dependencies.**

Set up Logstash

1. Create a new Ubuntu / CentOS machine instance to install Logstash
   - Minimum RAM requirements: 4GBs
   - Minimum 2CPUs
2. SSH into the machine with a privileged account and escalate to root with `sudo su`
3. Change to the /tmp directory to store the the onboarding script: `cd /tmp`
4. Download the set-up script script from `TODO [link]` and check the SHA1 hash with: `Curl [link] > logstash_setup.sh && echo "SHA1 Checksum:` sha1sum logstash_setup.sh `"`
   - The correct SHA1 hash should be: `fcfe3ec8a6a56bc7c626e42254c633e65ac4f2e4` as of 9/17/2020

5. If the hash checks out, make the script executable and run it with: `chmod logstash_setup.sh +x && ./logstash_setup.sh`
6. Edit configuration `vim /etc/logstash/conf.d/s3_sentinel.conf`
   - Make sure to fill in the access key id of the user created in the Setup AWS section
   - Also the secret access key from the same user created
   - The bucket section is the S3 bucket created previously
   - The prefix is "Sentinel/"
   - For the output, make sure to fill in the workspace ID and the workspace Key of the Sentinel instance
   - If the customer's azure instance is not a "gov" environment, comment out the `endpoint => "ods.opinsights.azure.us"` line
   - A special filter that works with large customers is now needed (the current config created by the script does not account for extremely large JSONs). This filter is needed because Sentinel by default cannot ingest such large documents. Therefore the logs created by AWS need to be split into events and those events need to be sent one by one inside of the "message" field to be parsed in KQL. Also, the GuardDuty logs do not come in the same format so we have to make sure that we prune out everything except the metadata and the message field from the GuardDuty logs. This is the filter:

```
# AWS S3 -> Logstash -> Elasticsearch pipeline.
# References:
#   https://www.elastic.co/guide/en/logstash/current/plugins-inputs-s3.html
#   https://www.elastic.co/blog/logstash-lines-inproved-resilience-in-S3-input
#   https://www.elastic.co/guide/en/logstash/current/working-with-plugins.html
input {
  s3 {
    "access_key_id" => "[redacted]*"
    "secret_access_key" => "[redacted]*"
    "region" => "us-gov-west-1"
    "bucket" => "[redacted]-sentinel"
    "prefix" => "Sentinel/AW[redacted]/[redacted]2"
    "interval" => "10"
    #"additional_settings" => {
    ## "force_path_style" => true
    ## "follow_redirects" => false
    #             }
  }
}

filter {
  json {
    source => "message"
  }
  if [Records] {
    mutate {
      remove_field => ["message"]
      add_tag => "CloudTrail"
    }
    split {
      field => "Records"
  #    add_field => {"message"}
  #    target => "message"
    }
    mutate {
      add_field => {"message" => "%{Records}"}
      remove_field => ["Records"]
    }
  }
  else if [service][serviceName] == "guardduty" {
    prune {
      whitelist_names => ["message", "@version", "@timestamp"]
    }
    mutate {
      add_tag => "GuardDuty"
    }
  }
}

output {
    #file {
    #  path => "/etc/logstash/test.log"
    #}
    microsoft-logstash-output-azure-loganalytics {
        codec => "json_lines"
        workspace_id => "[redacted]"
        workspace_key => "[redacted]"
        custom_log_table_name => "AWSCloudLogs"
        plugin_flush_interval => 5
        endpoint => "ods.opinsights.azure.us"
     }
    # for debug
    # stdout { codec => rubydebug }
}
```

7. Test configuration `/usr/share/bin/logstash -f s3_sentinel.conf -t`
8. If the configuration checks out, start logstash with `systemctl start logstash`