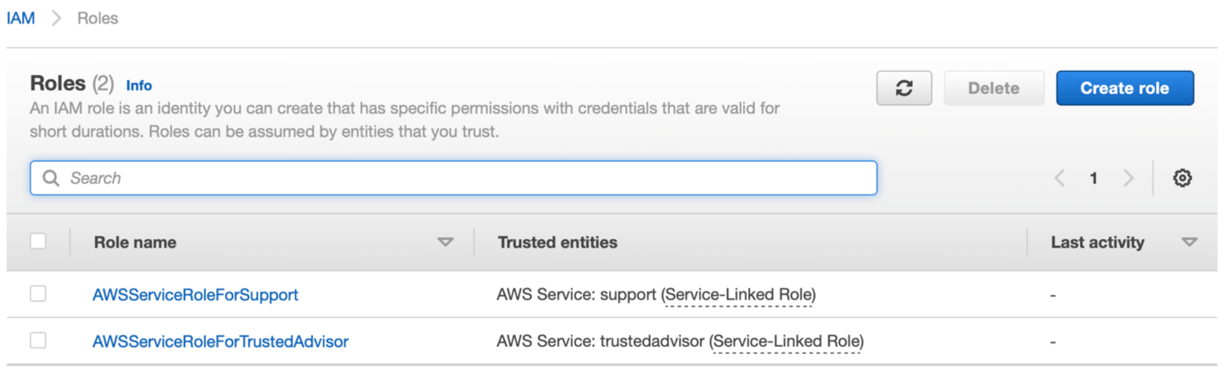
Demo

Step1: Initial stage of AWS services

Initially in our AWS account in the eu-central-1 region, we don't have any services created. Below images shows that we don't have any s3 bucket or any other services. In the next step we will execute the code to create these resources.

Figure 1: IAM roles before

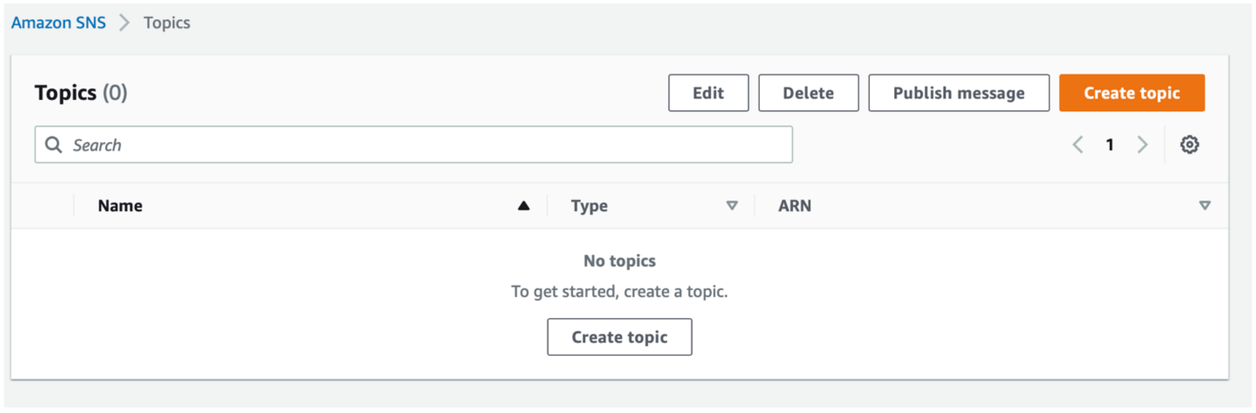


Roles (2)

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Role name	Trusted entities	Last activity
AWSServiceRoleForSupport	AWS Service: support (Service-Linked Role)	-
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service-Linked Role)	-

Figure 2: SNS before

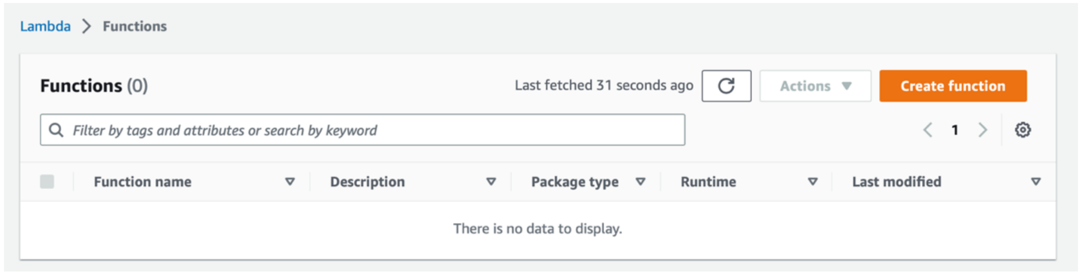


Topics (0)

No topics

To get started, create a topic.

Figure 3: Lambda functions before

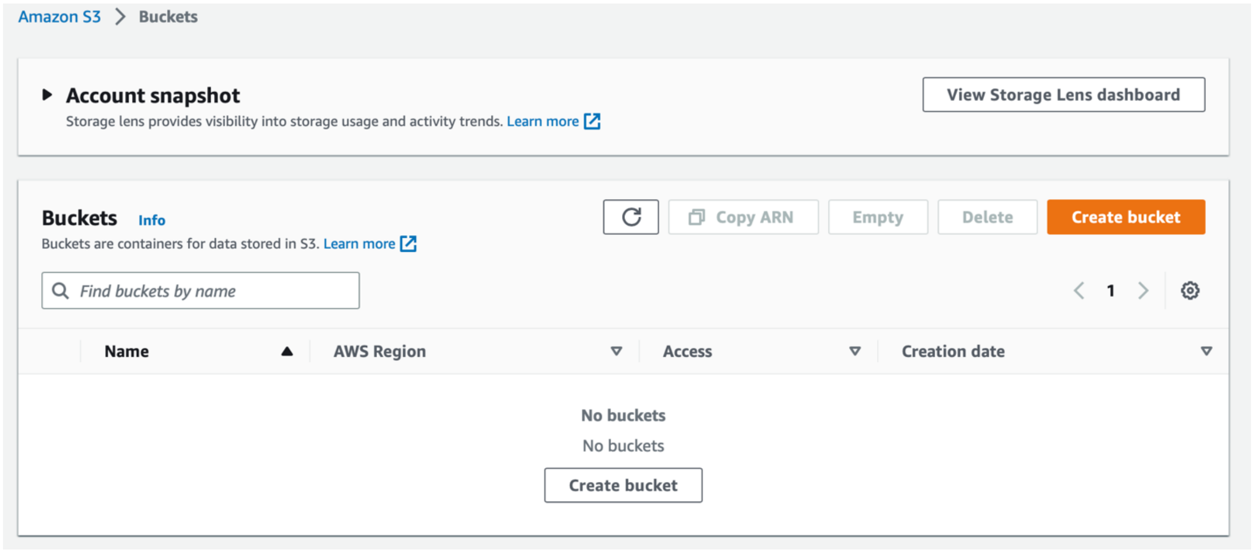


Functions (0)

Last fetched 31 seconds ago

There is no data to display.

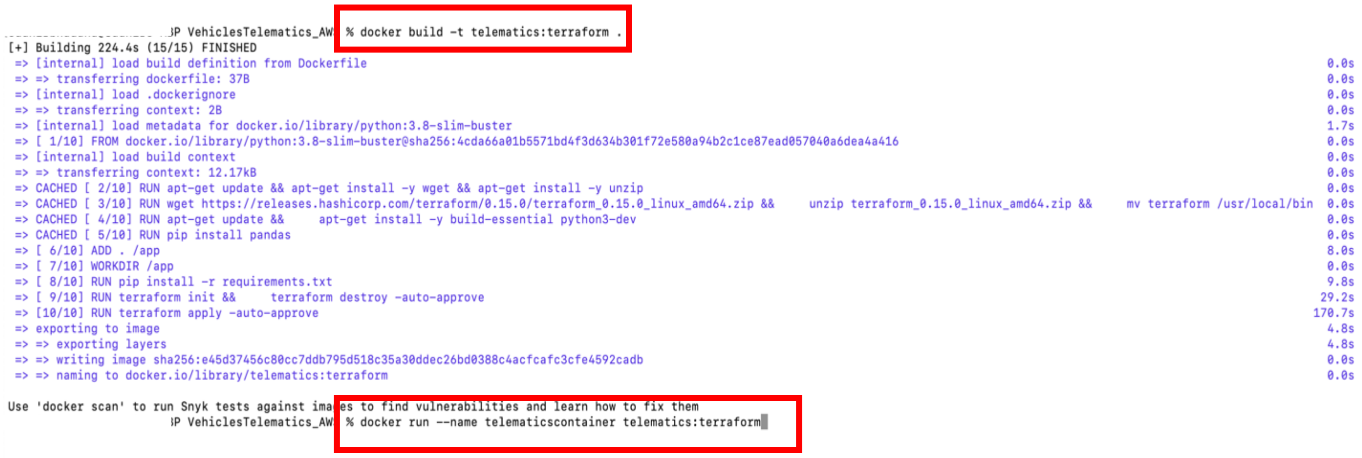
Figure 4: S3 buckets before code


Amazon S3 > Buckets
Account snapshot
Storage lens provides visibility into storage usage and activity trends. Learn more
View Storage Lens dashboard
Buckets Info
Buckets are containers for data stored in S3. Learn more
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
1
Name
AWS Region
Access
Creation date
No buckets
No buckets
Create bucket

Step2: Build docker image

We have created a docker file to execute terraform files to create infrastructure on AWS. While building the docker image ("docker build -t imagename:imagetag ."), we execute terraform apply command. In the below images, we can see our complete architecture is created in AWS. Once image is built, we ran the container which sends the fake data to our services.

Figure 5: Build docker image and run container



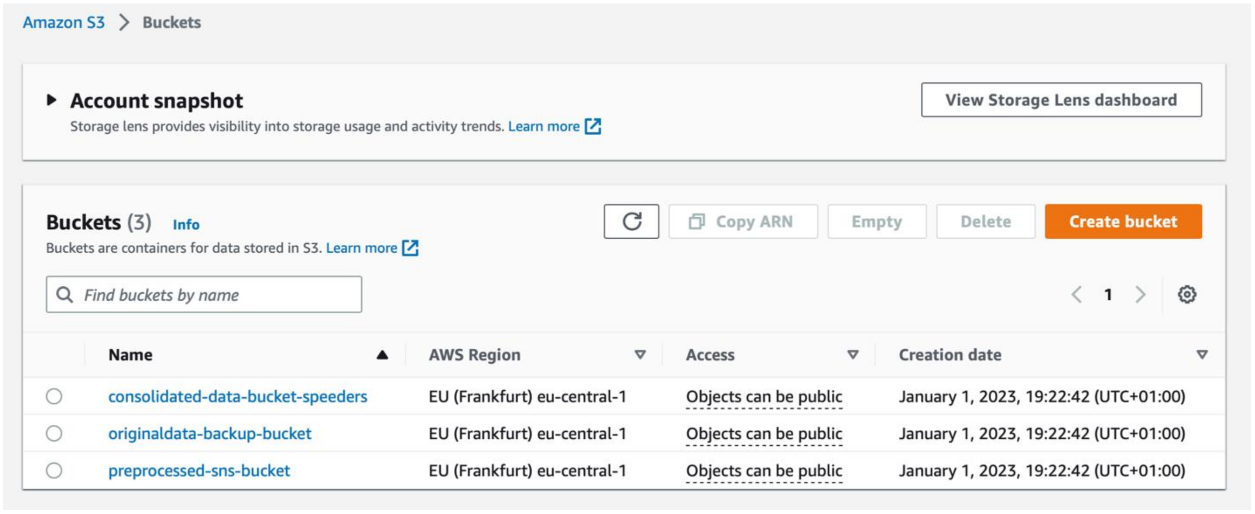
```
[redacted]3P VehiclesTelematics_AWS % docker build -t telematics:terraform .
[+] Building 224.4s (15/15) FINISHED
 => [internal] load build definition from Dockerfile                                                                                                                   0.0s
 => => transferring dockerfile: 37B                                                                                                                                    0.0s
 => [internal] load .dockerignore                                                                                                                                      0.0s
 => => transferring context: 2B                                                                                                                                        0.0s
 => [internal] load metadata for docker.io/library/python:3.8-slim-buster                                                                                              1.7s
 => [ 1/10] FROM docker.io/library/python:3.8-slim-buster@sha256:4cda66a01b5571bd4f3d634b301f72e580a94b2c1ce87ead057040a6dea4a416                                       0.0s
 => [internal] load build context                                                                                                                                      0.0s
 => => transferring context: 12.17kB                                                                                                                                   0.0s
 => CACHED [ 2/10] RUN apt-get update && apt-get install -y wget && apt-get install -y unzip                                                                           0.0s
 => CACHED [ 3/10] RUN wget https://releases.hashicorp.com/terraform/0.15.0/terraform_0.15.0_linux_amd64.zip &&     unzip terraform_0.15.0_linux_amd64.zip &&     mv terraform /usr/local/bin  0.0s
 => CACHED [ 4/10] RUN apt-get update &&     apt-get install -y build-essential python3-dev                                                                            0.0s
 => CACHED [ 5/10] RUN pip install pandas                                                                                                                              0.0s
 => [ 6/10] ADD . /app                                                                                                                                                 8.0s
 => [ 7/10] WORKDIR /app                                                                                                                                               0.0s
 => [ 8/10] RUN pip install -r requirements.txt                                                                                                                        9.8s
 => [ 9/10] RUN terraform init &&     terraform destroy -auto-approve                                                                                                 29.2s
 => [10/10] RUN terraform apply -auto-approve                                                                                                                        170.7s
 => exporting to image                                                                                                                                                 4.8s
 => => exporting layers                                                                                                                                                4.8s
 => => writing image sha256:e45d37456c80cc7ddb795d518c35a30ddec26bd0388c4acfcafc3cfe4592cadb                                                                           0.0s
 => => naming to docker.io/library/telematics:terraform                                                                                                                0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
[redacted]3P VehiclesTelematics_AWS % docker run --name telematicscontainer telematics:terraform
```

Step3: Final stage of AWS services

Our terraform code has created the architecture and with the help of faker python library we have sent data to our AWS microservice architecture. We can see the original data send to kinesis by our python faker code. Once processed and filtered, all the over-speeding vehicles can be seen in our final s3 bucket.

Figure 6: S3 buckets created by terraform



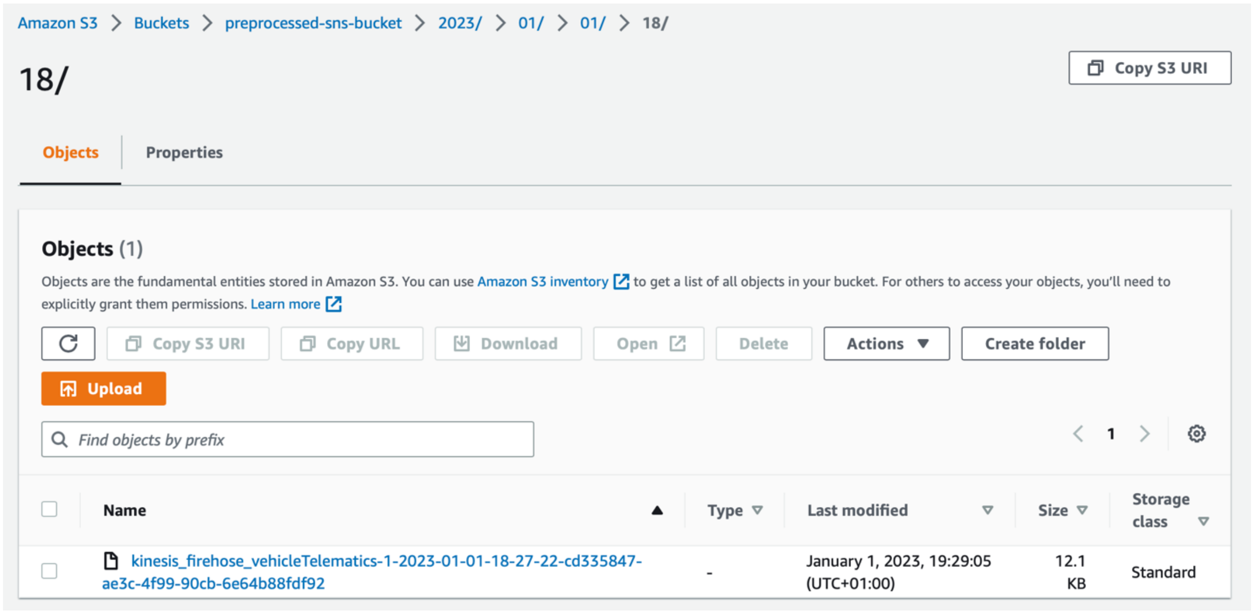
Storage lens provides visibility into storage usage and activity trends. Learn more

Buckets (3) Info

Buckets are containers for data stored in S3. Learn more

Name	AWS Region	Access	Creation date
consolidated-data-bucket-speeders	EU (Frankfurt) eu-central-1	Objects can be public	January 1, 2023, 19:22:42 (UTC+01:00)
originaldata-backup-bucket	EU (Frankfurt) eu-central-1	Objects can be public	January 1, 2023, 19:22:42 (UTC+01:00)
preprocessed-sns-bucket	EU (Frankfurt) eu-central-1	Objects can be public	January 1, 2023, 19:22:42 (UTC+01:00)

Figure 7: Processed data in S3 bucket

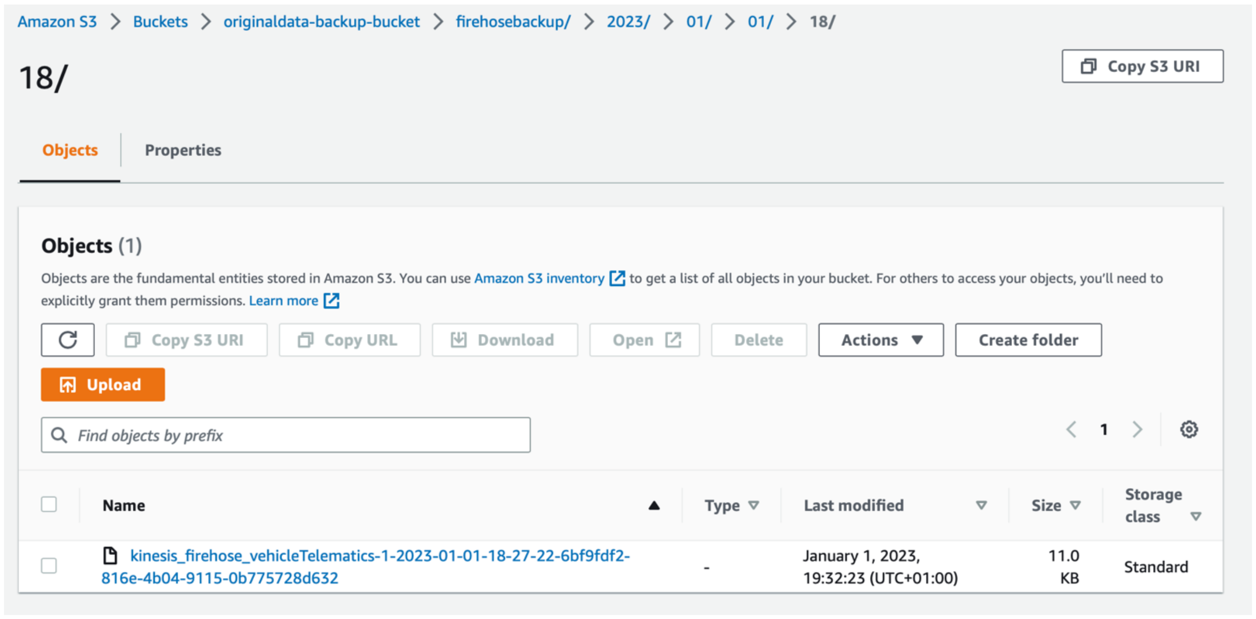


Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
kinesis_firehose_vehicleTelematics-1-2023-01-01-18-27-22-cd335847-ae3c-4f99-90cb-6e64b88fdf92	-	January 1, 2023, 19:29:05 (UTC+01:00)	12.1 KB	Standard

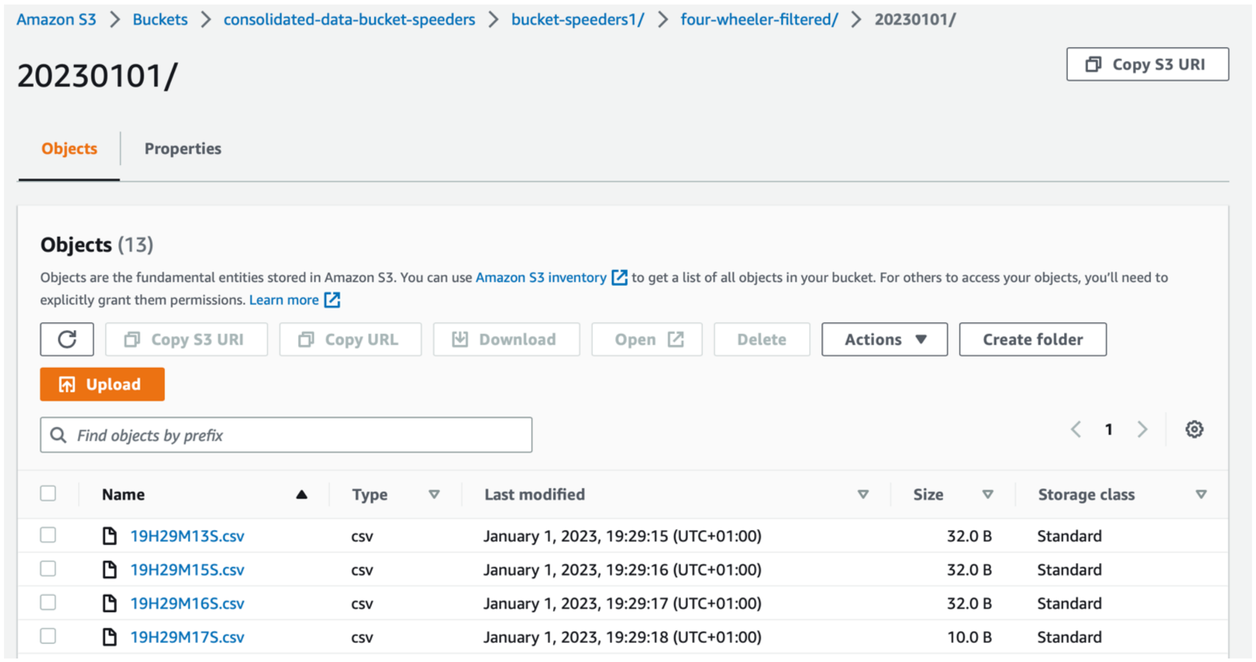
Figure 8: Back up of original data in S3 bucket



Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
kinesis_firehose_vehicleTelematics-1-2023-01-01-18-27-22-6bf9fdf2-816e-4b04-9115-0b775728d632	-	January 1, 2023, 19:32:23 (UTC+01:00)	11.0 KB	Standard

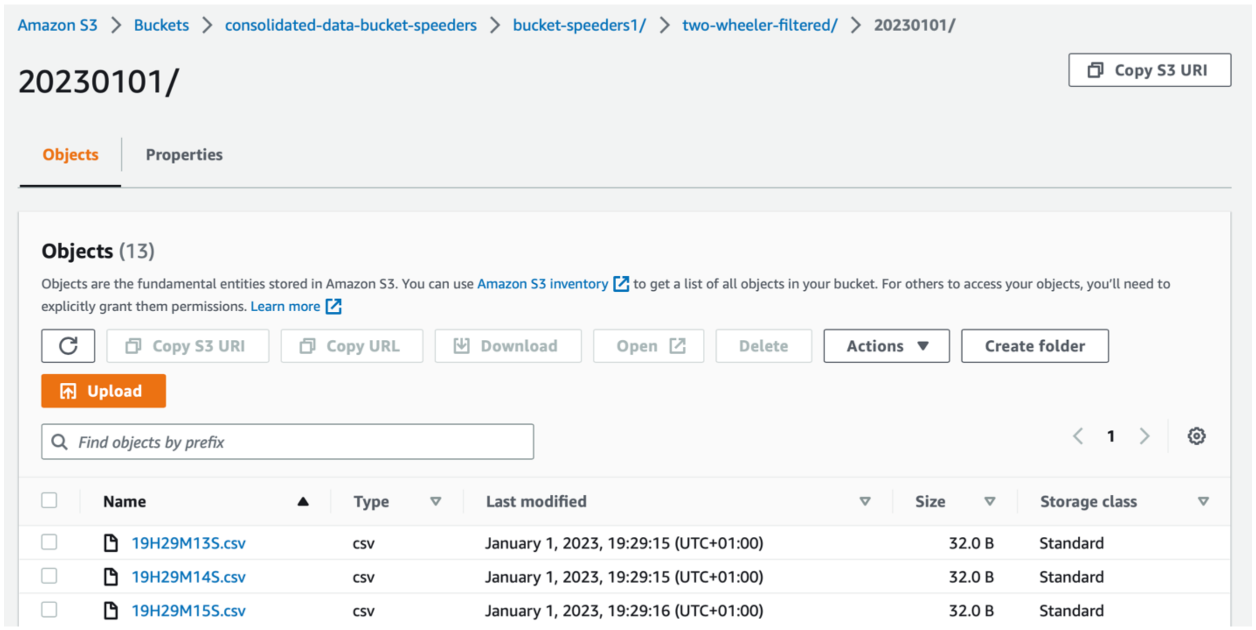
Figure 9: Overspeeding four wheeler filtered



Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
19H29M13S.csv	csv	January 1, 2023, 19:29:15 (UTC+01:00)	32.0 B	Standard
19H29M15S.csv	csv	January 1, 2023, 19:29:16 (UTC+01:00)	32.0 B	Standard
19H29M16S.csv	csv	January 1, 2023, 19:29:17 (UTC+01:00)	32.0 B	Standard
19H29M17S.csv	csv	January 1, 2023, 19:29:18 (UTC+01:00)	10.0 B	Standard

Figure 10: Overspeeding two wheeler filtered



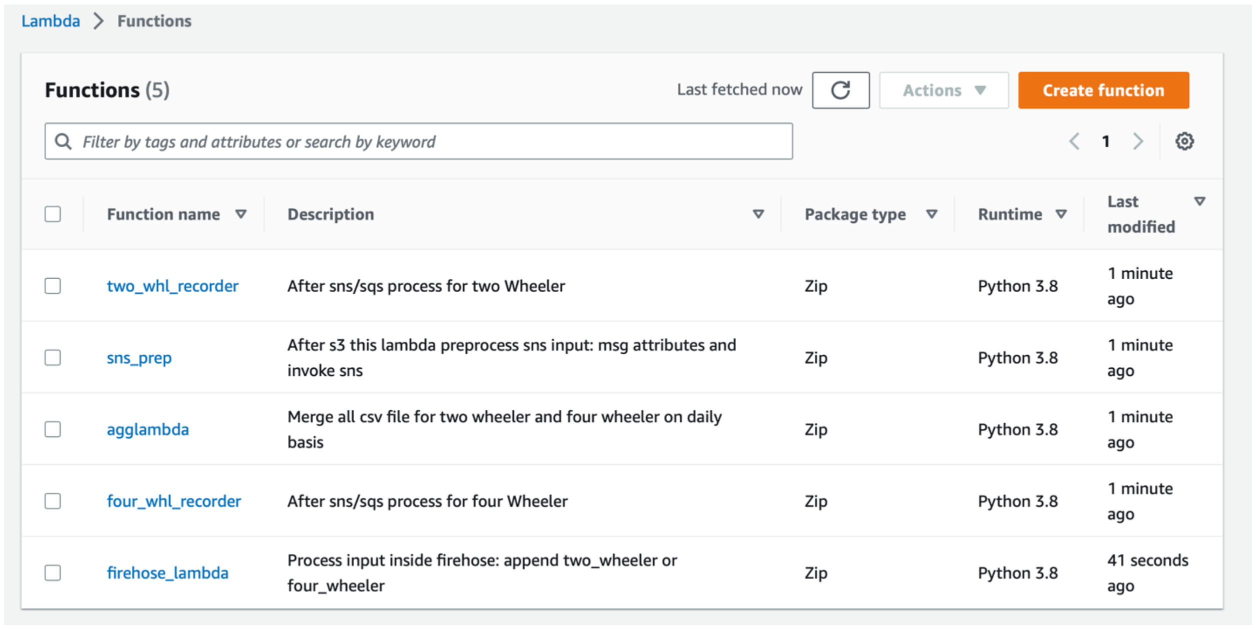
20230101/

Objects (13)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
19H29M13S.csv	csv	January 1, 2023, 19:29:15 (UTC+01:00)	32.0 B	Standard
19H29M14S.csv	csv	January 1, 2023, 19:29:15 (UTC+01:00)	32.0 B	Standard
19H29M15S.csv	csv	January 1, 2023, 19:29:16 (UTC+01:00)	32.0 B	Standard

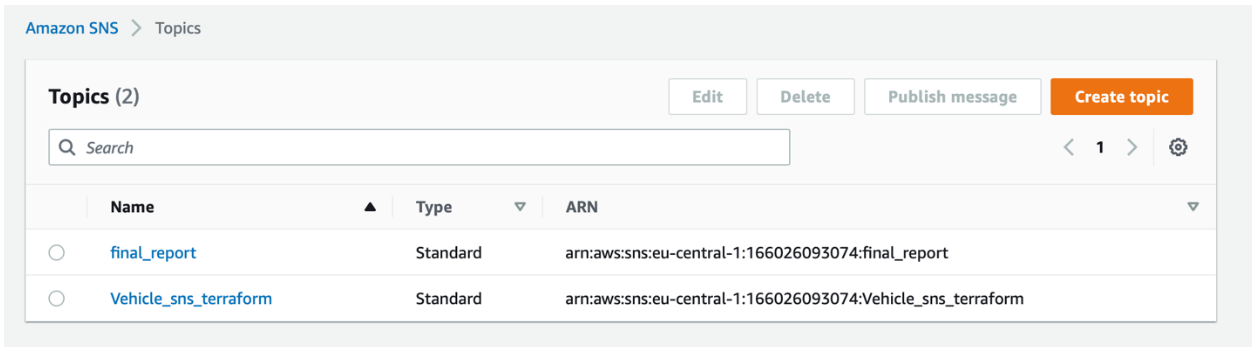
Figure 11: Lambda functions created by terraform



Functions (5)

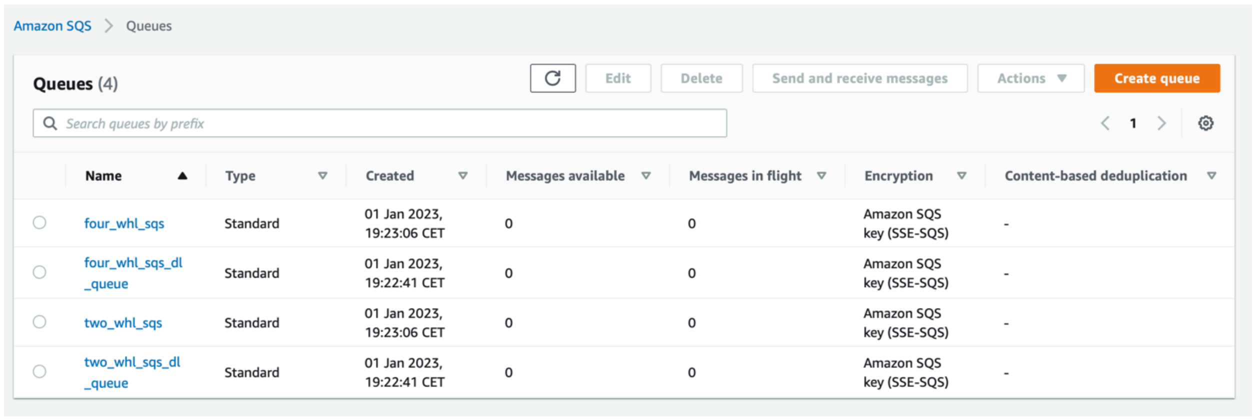
Function name	Description	Package type	Runtime	Last modified
two_whl_recorder	After sns/sqs process for two Wheeler	Zip	Python 3.8	1 minute ago
sns_prep	After s3 this lambda preprocess sns input: msg attributes and invoke sns	Zip	Python 3.8	1 minute ago
agglambda	Merge all csv file for two wheeler and four wheeler on daily basis	Zip	Python 3.8	1 minute ago
four_whl_recorder	After sns/sqs process for four Wheeler	Zip	Python 3.8	1 minute ago
firehose_lambda	Process input inside firehose: append two_wheeler or four_wheeler	Zip	Python 3.8	41 seconds ago

Figure 12: SNS created by terraform



Name	Type	ARN
final_report	Standard	arn:aws:sns:eu-central-1:166026093074:final_report
Vehicle_sns_terraform	Standard	arn:aws:sns:eu-central-1:166026093074:Vehicle_sns_terraform

Figure 13: SQS created by terraform



Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplication
four_whl_sqs	Standard	01 Jan 2023, 19:23:06 CET	0	0	Amazon SQS key (SSE-SQS)	-
four_whl_sqs_dl_queue	Standard	01 Jan 2023, 19:22:41 CET	0	0	Amazon SQS key (SSE-SQS)	-
two_whl_sqs	Standard	01 Jan 2023, 19:23:06 CET	0	0	Amazon SQS key (SSE-SQS)	-
two_whl_sqs_dl_queue	Standard	01 Jan 2023, 19:22:41 CET	0	0	Amazon SQS key (SSE-SQS)	-