What is streaming and why does it matter?

STREAMING DATA WITH AWS KINESIS AND LAMBDA

Maksim Pecherskiy
Data Engineer

datacamp

Batch vs stream

Batch vs stream

Batch

- ~~"Better"~~
- Larger datasets
- More complex analysis
- Slower moving data
- Ex. Daily sales report
- Ex. Forecasting next month's sales
- Ex. Churn prediction

Stream

- ~~"Cooler"~~
- Simpler analysis: aggregation / filtering
- Individual records / micro batches
- Data moves FAST
- Ex. Fraud detection
- Ex. Monitoring wind turbines
- Ex. Real time alerting

Cody and the fleet

Cody

The Fleet

Telematics streaming

Amazon Kinesis

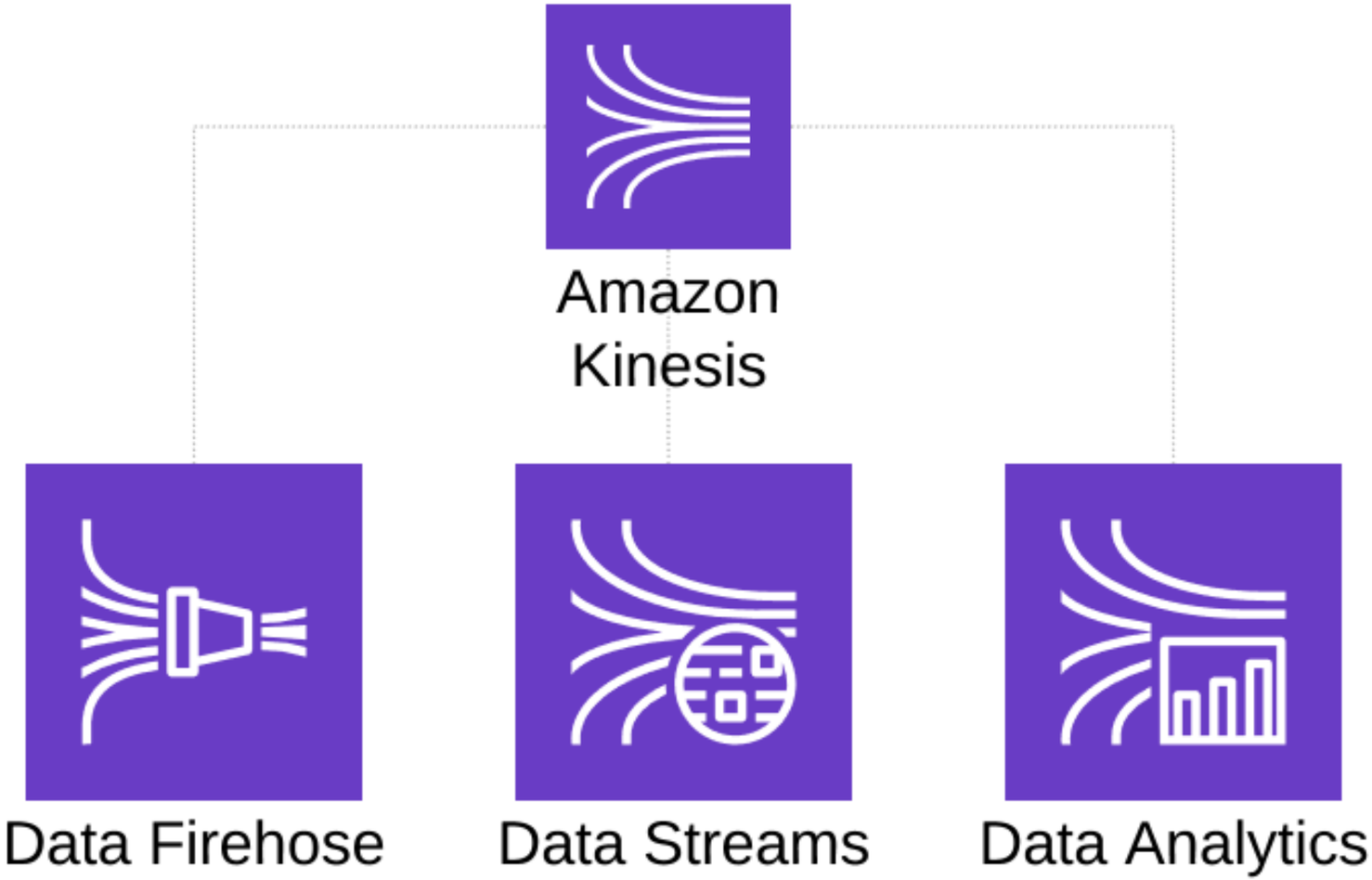
AWS Lambda

Amazon S3

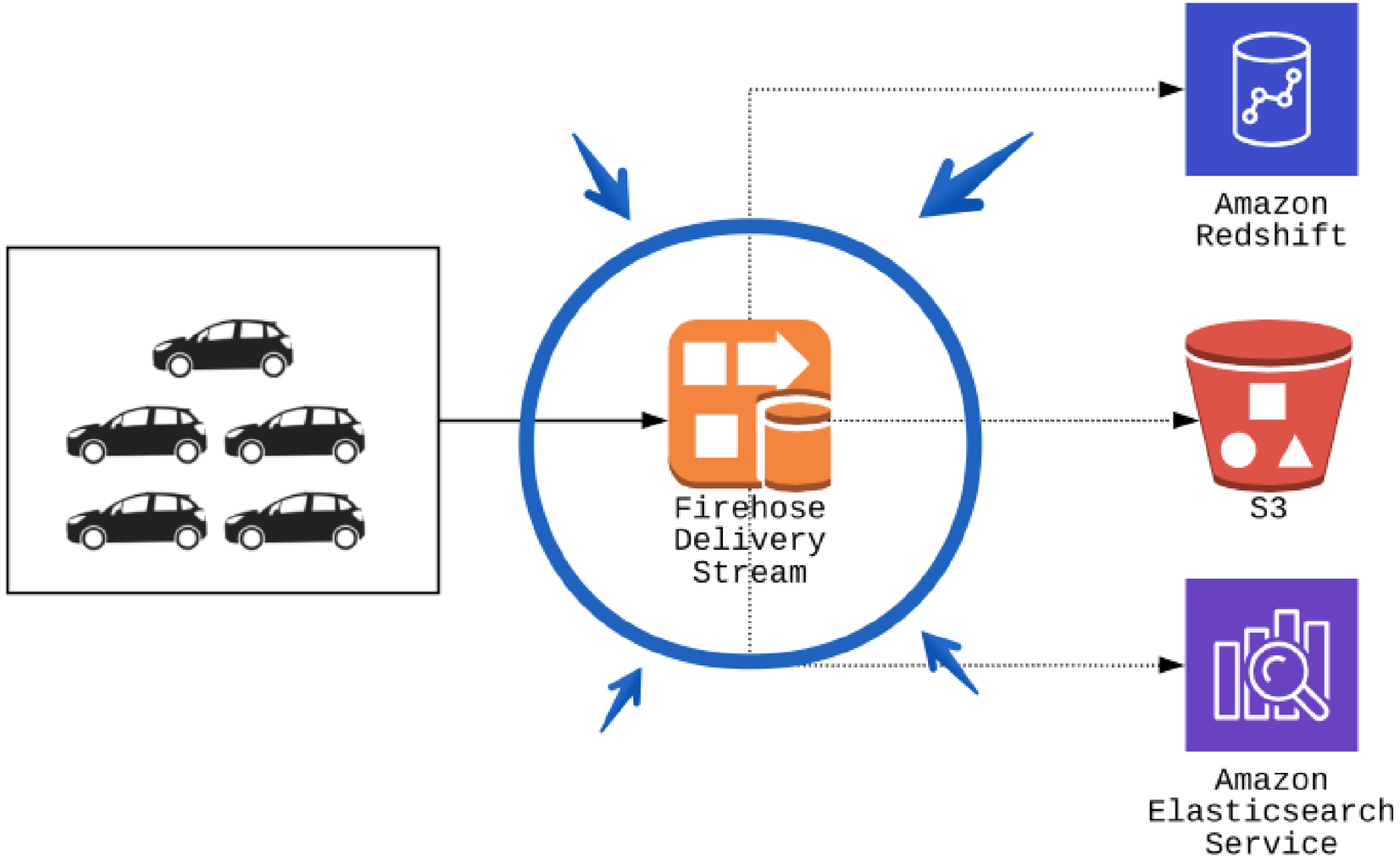
IAM

Amazon
SNS

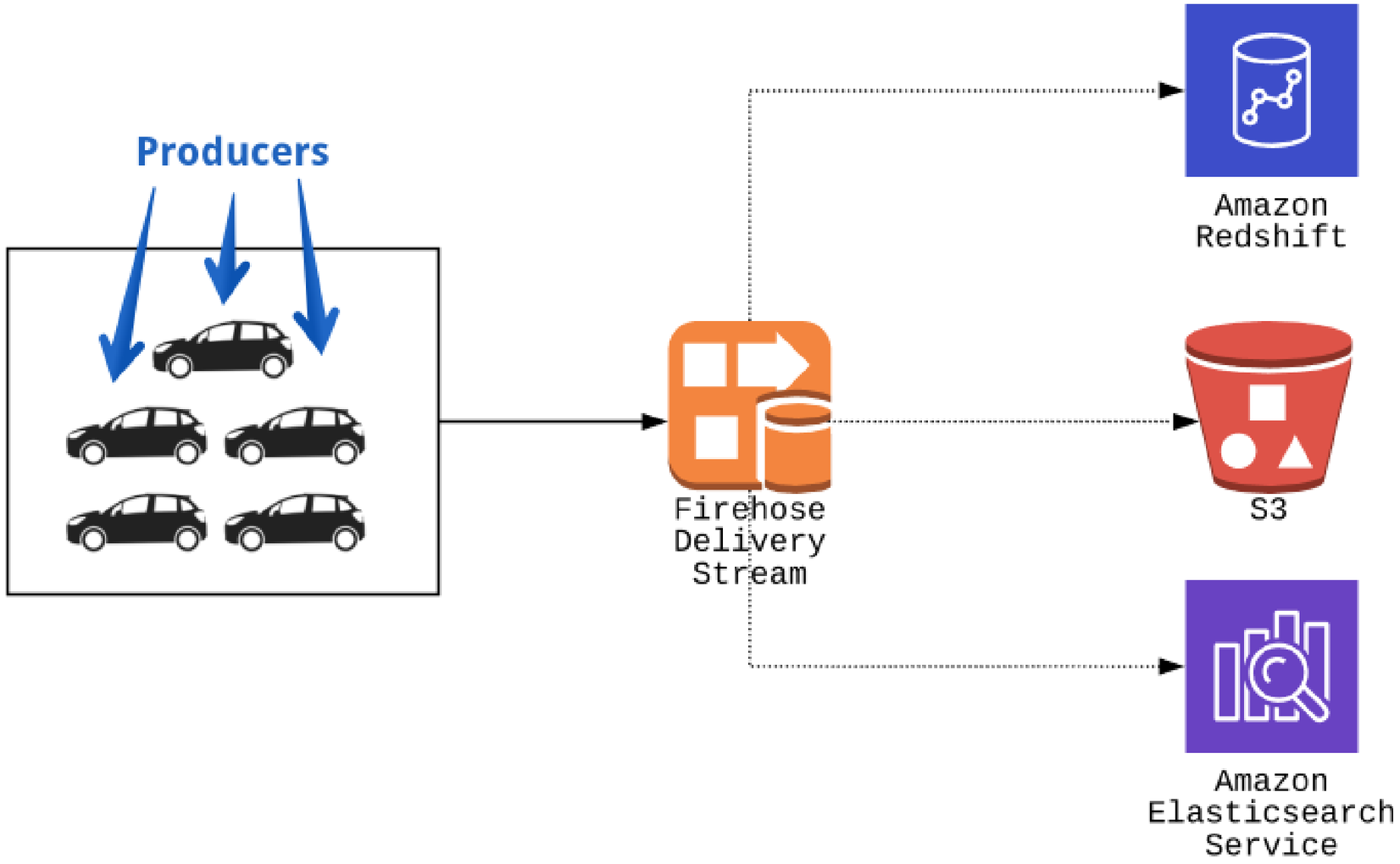
AWS Kinesis



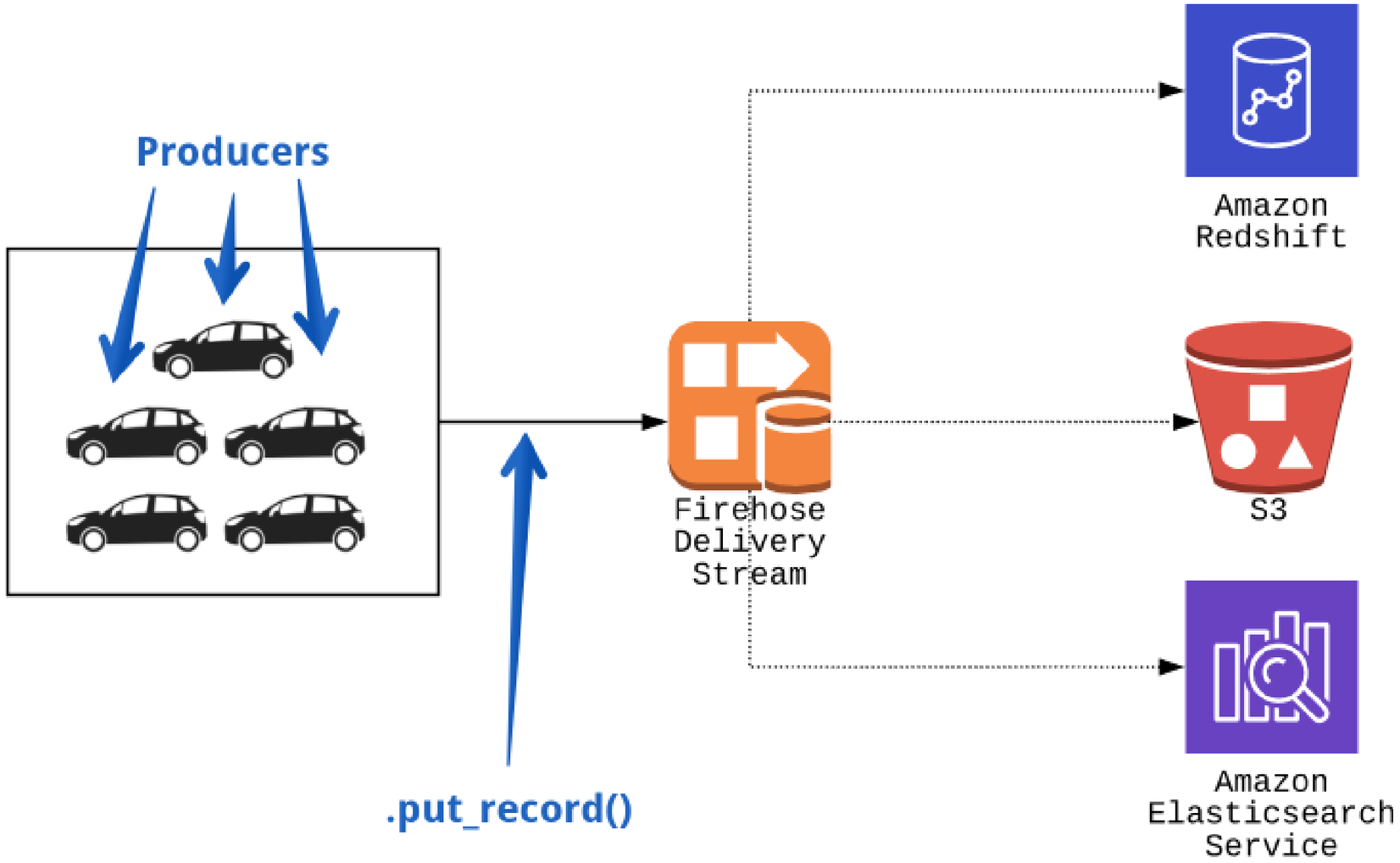
Data Firehose



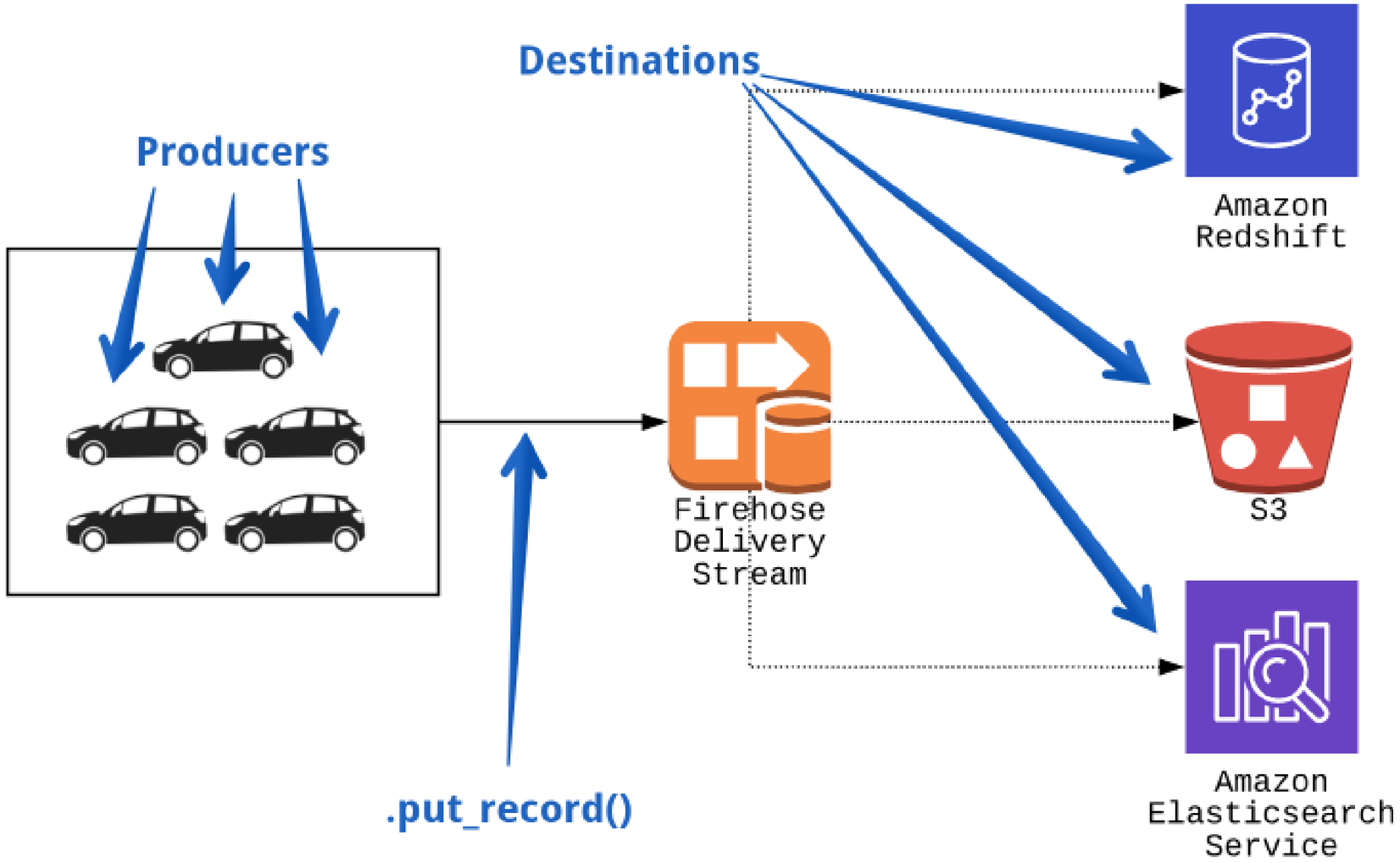
Delivery streams



Delivery streams



Delivery streams



Creating a Firehose client

```
import boto3
firehose = boto3.client('firehose',
                        aws_access_key_id=AWS_KEY_ID,
                        aws_secret_access_key=AWS_SECRET,
                        region_name='us-east-1')
```

Working with delivery streams

```
# Show created delivery streams
response = firehose.list_delivery_streams()
print(response['DeliveryStreamNames'])
```

```
['old-delivery-stream1', 'a-test-stream']
```

Delete streams

```
# Show created delivery streams
response = firehose.list_delivery_streams()
# Delete them all!
for stream_name in response['DeliveryStreamNames']:
      firehose.delete_delivery_stream(DeliveryStreamName=stream_name)
```

Review

- Batch vs stream
- Cody and telematics collection
- AWS Kinesis
- Kinesis Firehose Delivery Streams
- AWS Kinesis Data Firehose
- List and delete Firehose delivery streams
- Producer -> data generator
- Destination -> where the data is going

Let's practice!

STREAMING DATA WITH AWS KINESIS AND LAMBDA

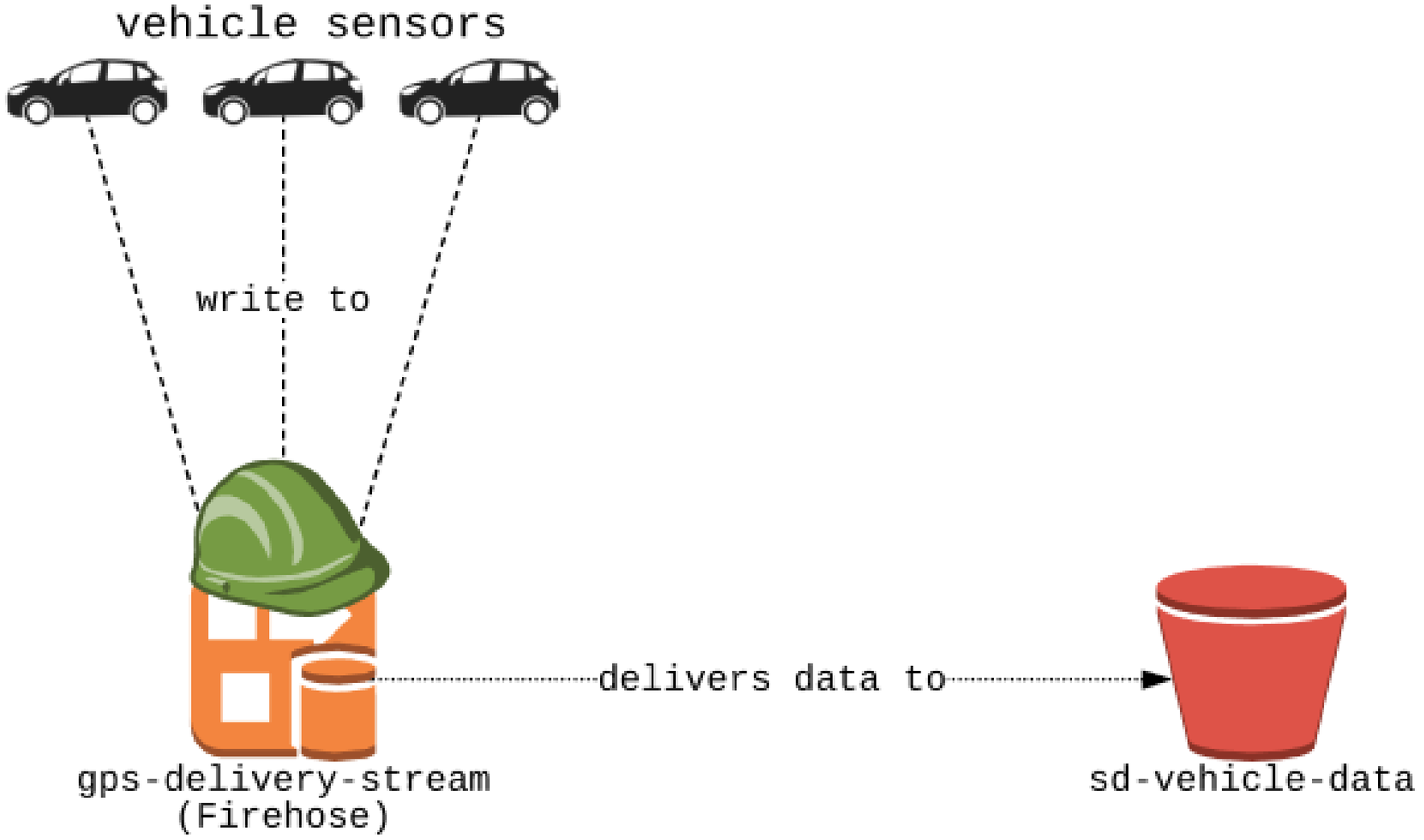
Getting ready for the first stream

STREAMING DATA WITH AWS KINESIS AND LAMBDA

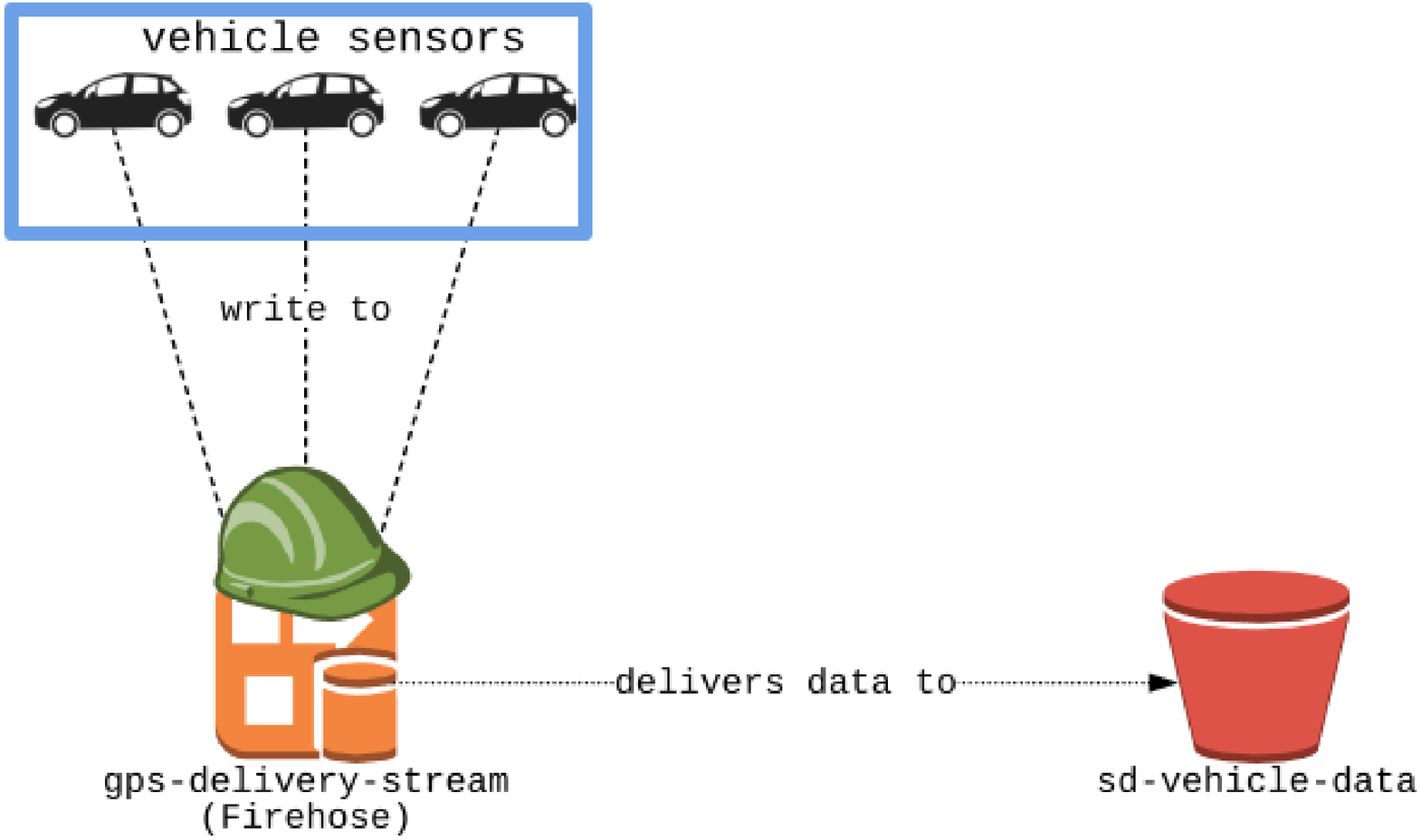
Maksim Pecherskiy
Data Engineer

datacamp

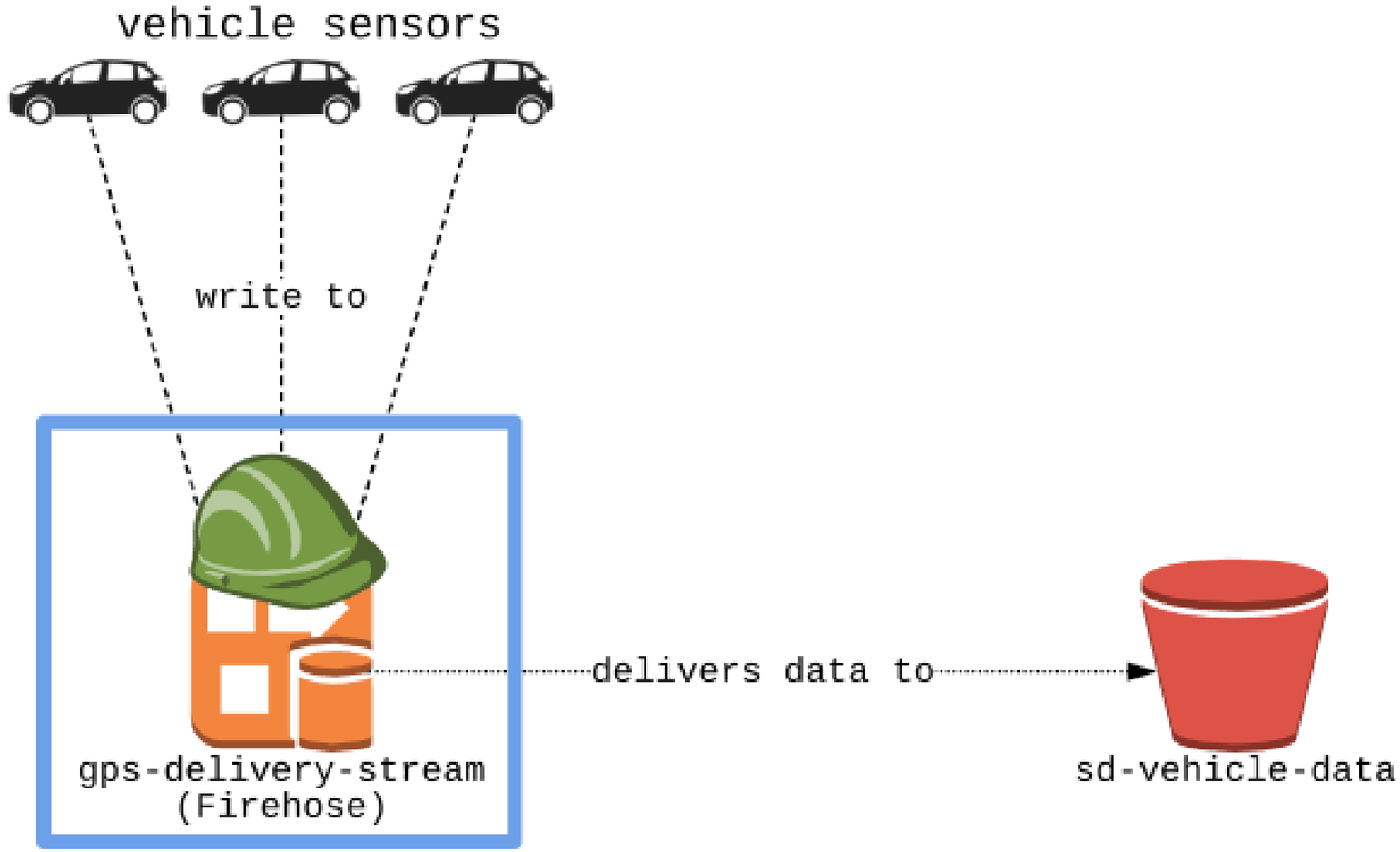
End goal


vehicle sensors
write to
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

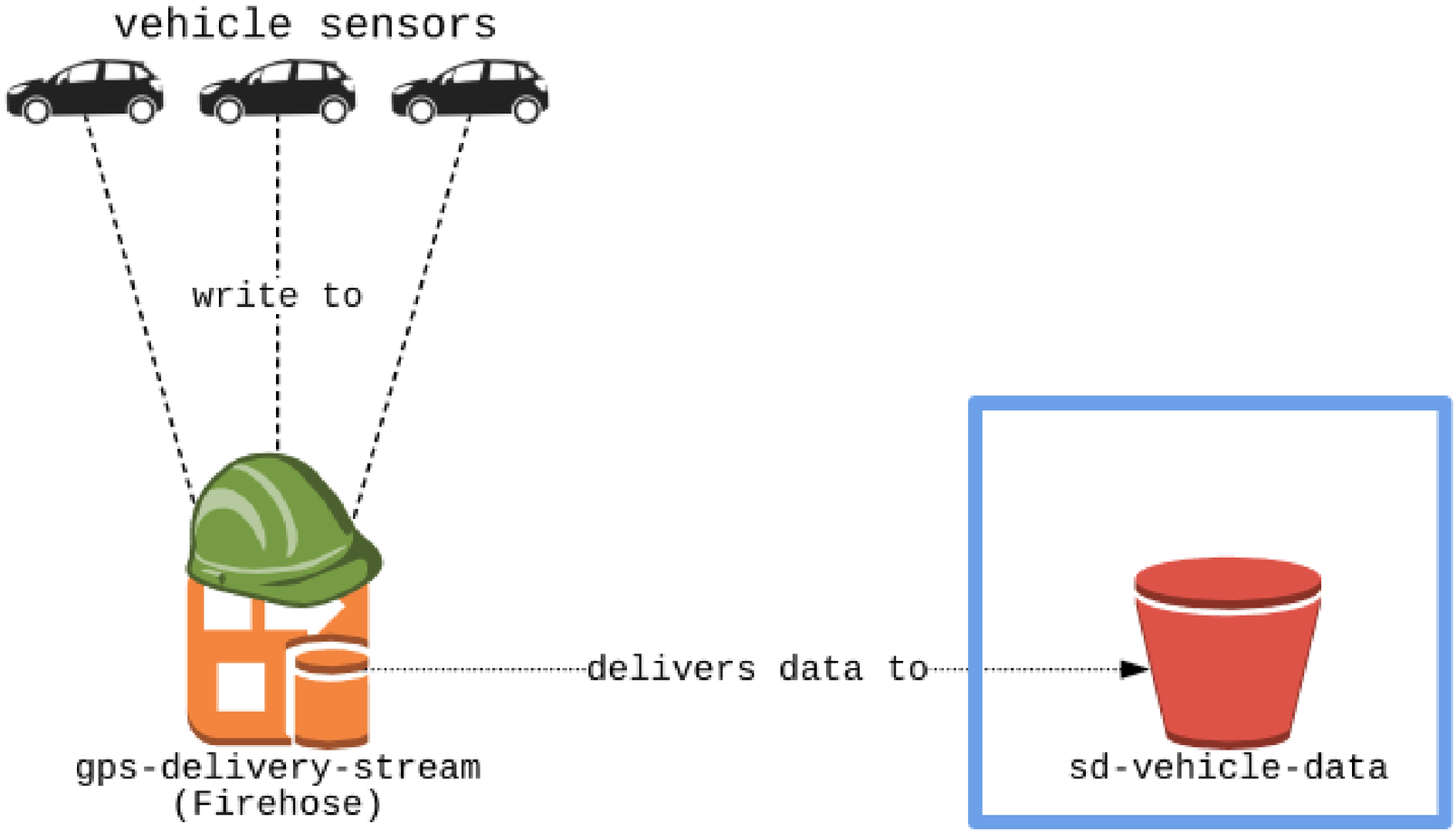
End goal


vehicle sensors
write to
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

End goal


vehicle sensors
write to
gps-delivery-stream
(Firehose)
delivers data to
sd-vehicle-data

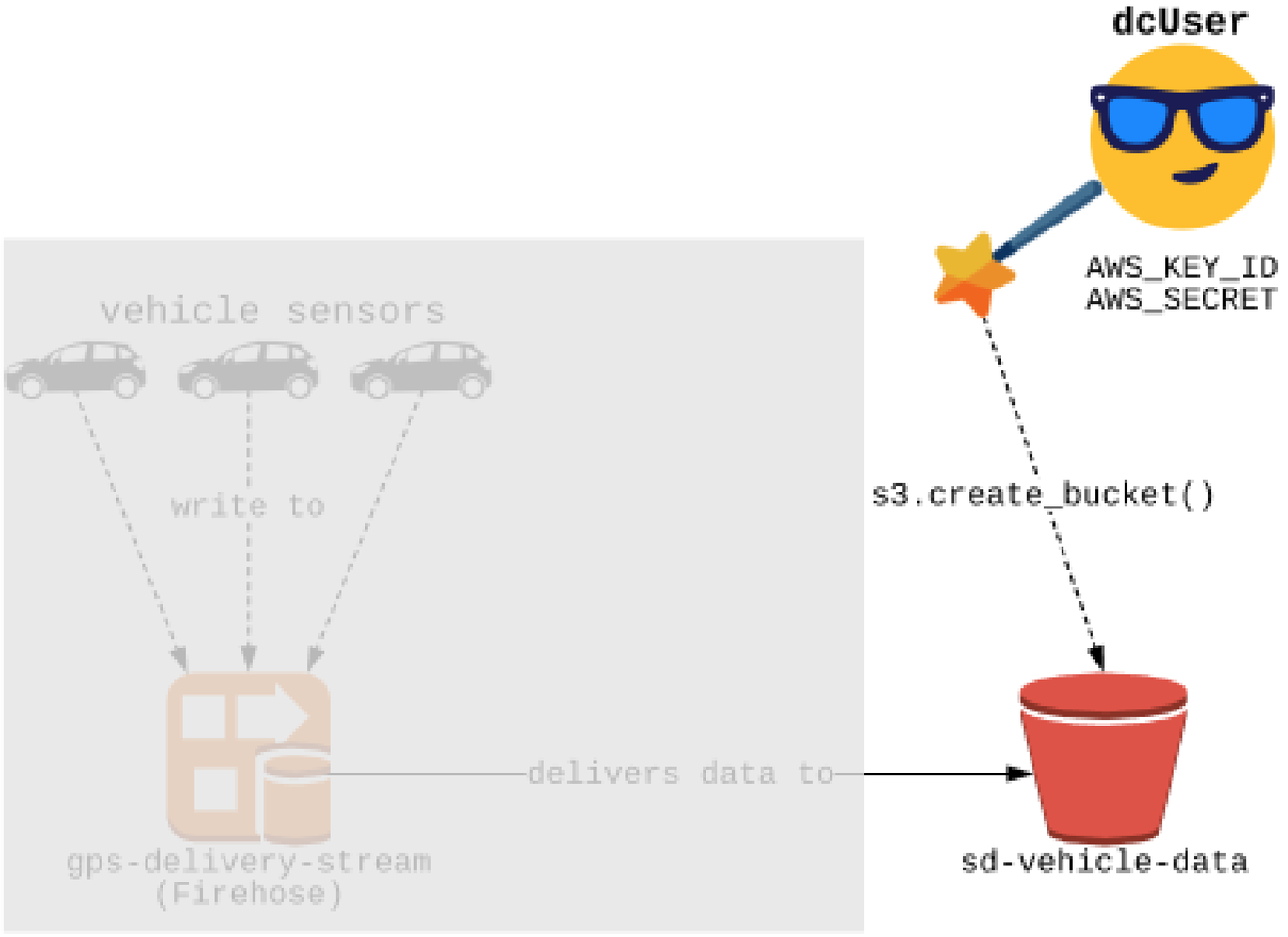
End goal


vehicle sensors
write to
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

dcUser

AWS_KEY_ID
AWS_SECRET

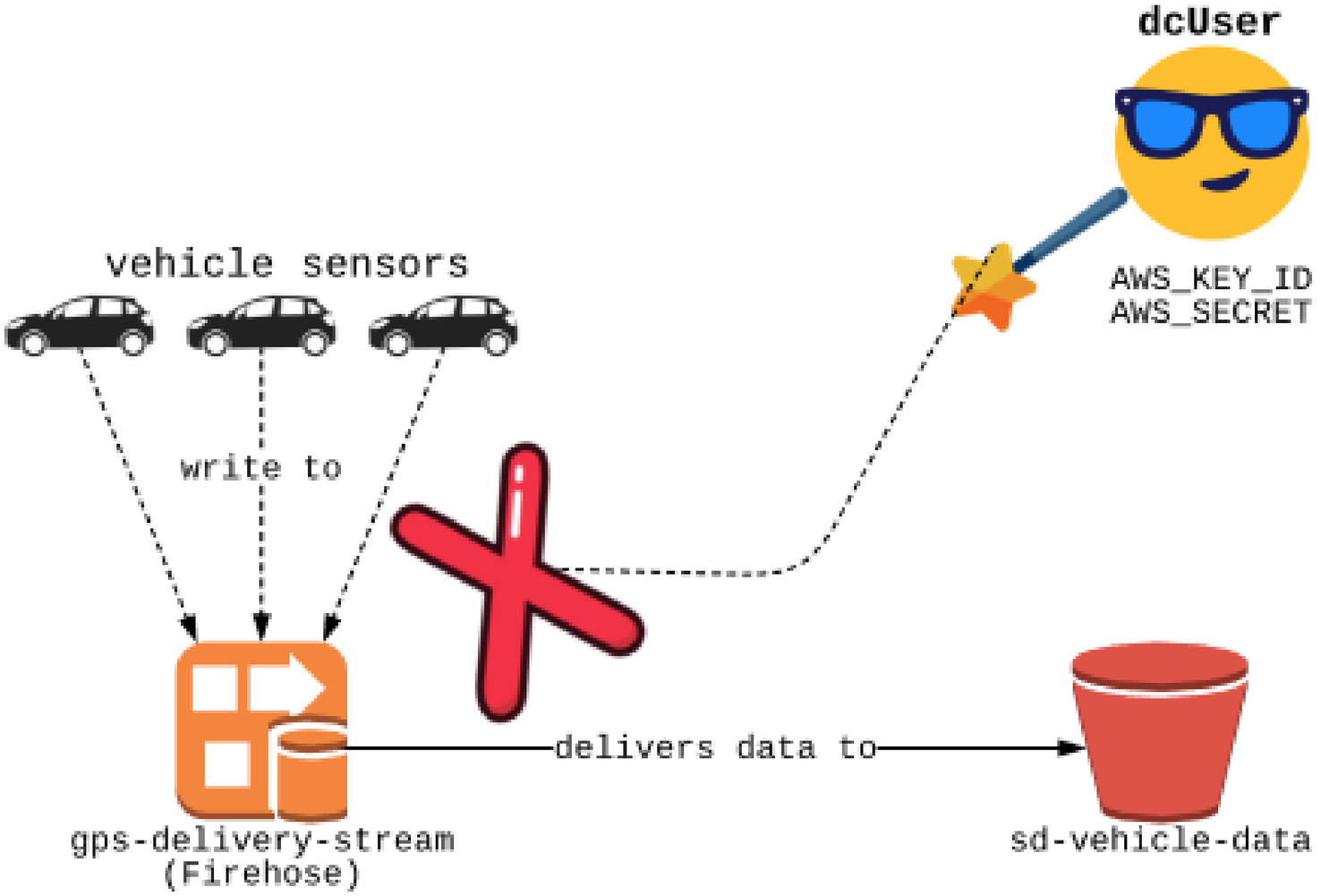
dcUser

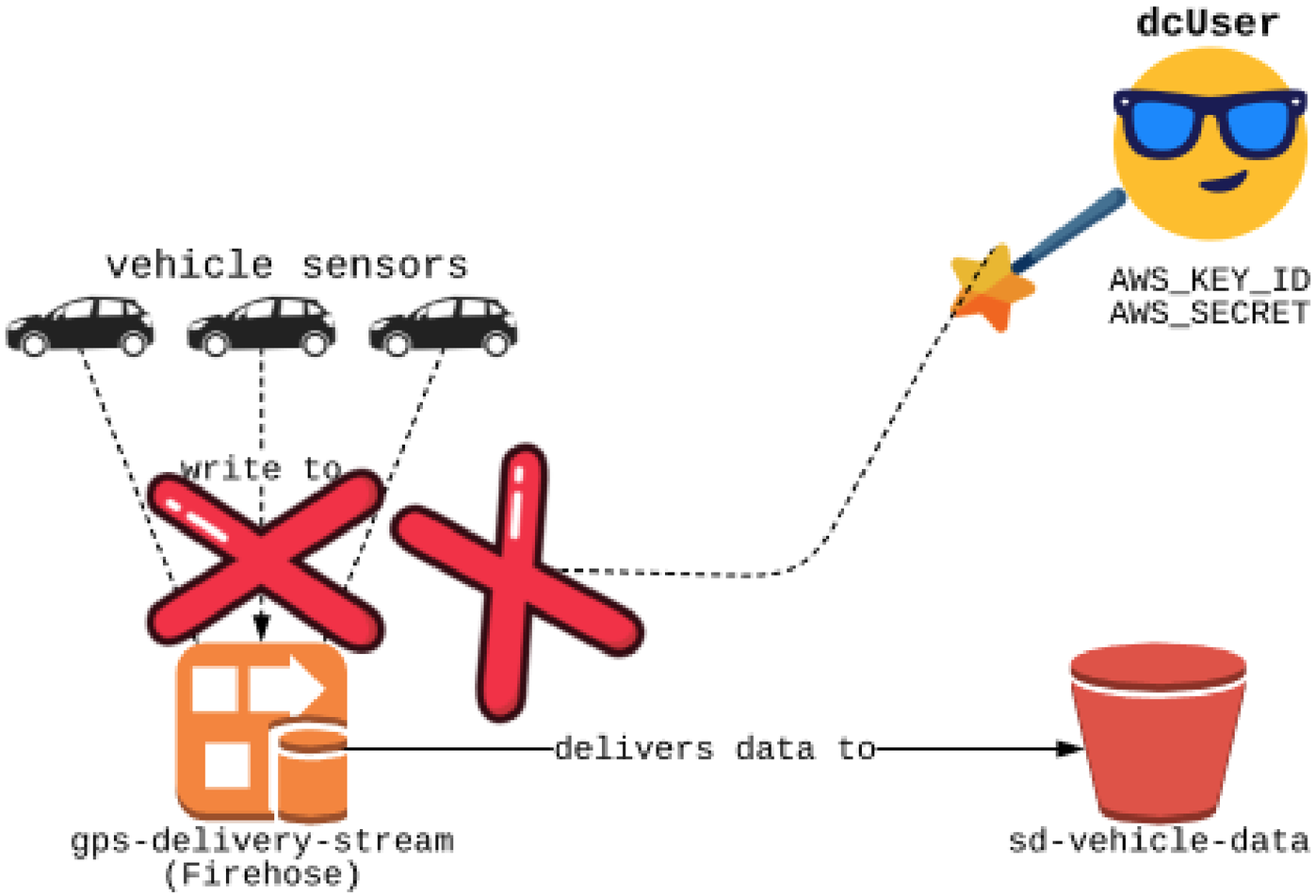

dcUser
AWS_KEY_ID
AWS_SECRET
s3.create_bucket()
vehicle sensors
write to
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

A destination S3 bucket

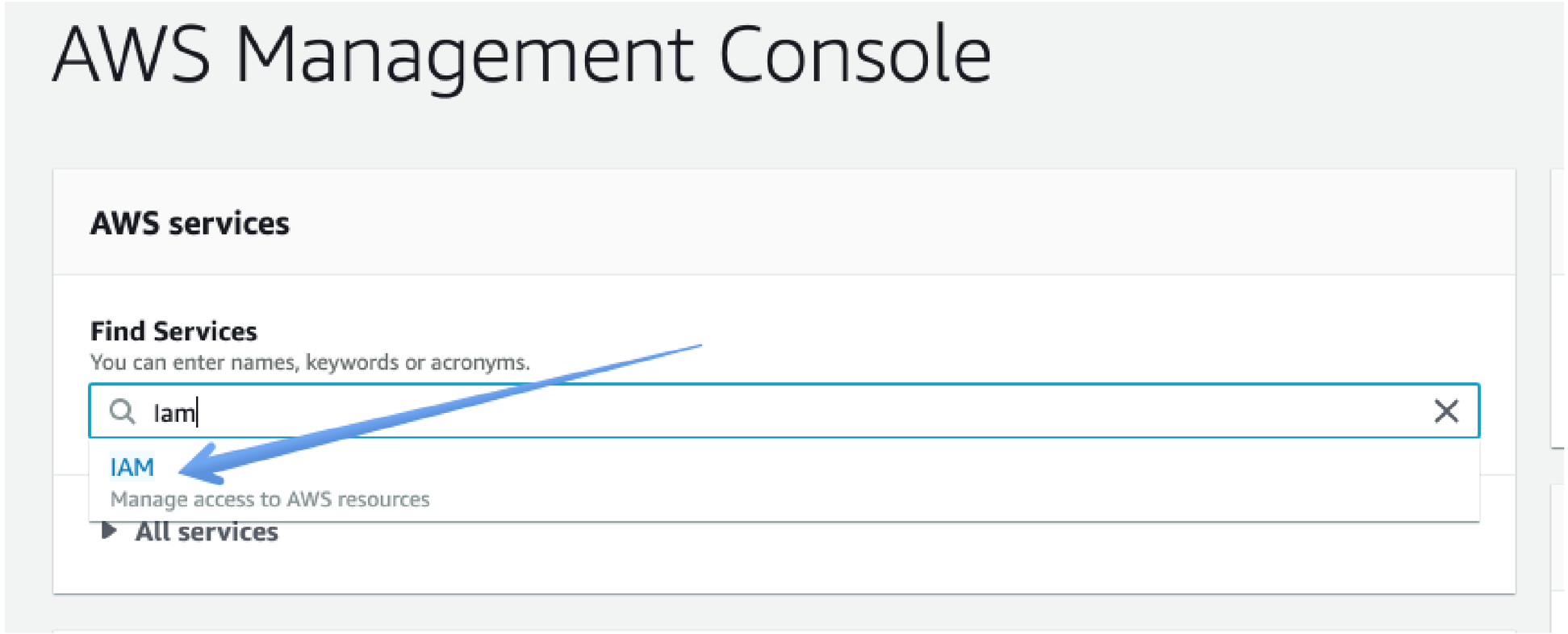
```
s3 = boto3.client('s3',
                  aws_access_key_id=AWS_KEY_ID,
                  aws_secret_access_key=AWS_SECRET,
                  region_name='us-east-1')
```

```
s3.create_bucket(Bucket='sd-vehicle-data')
```

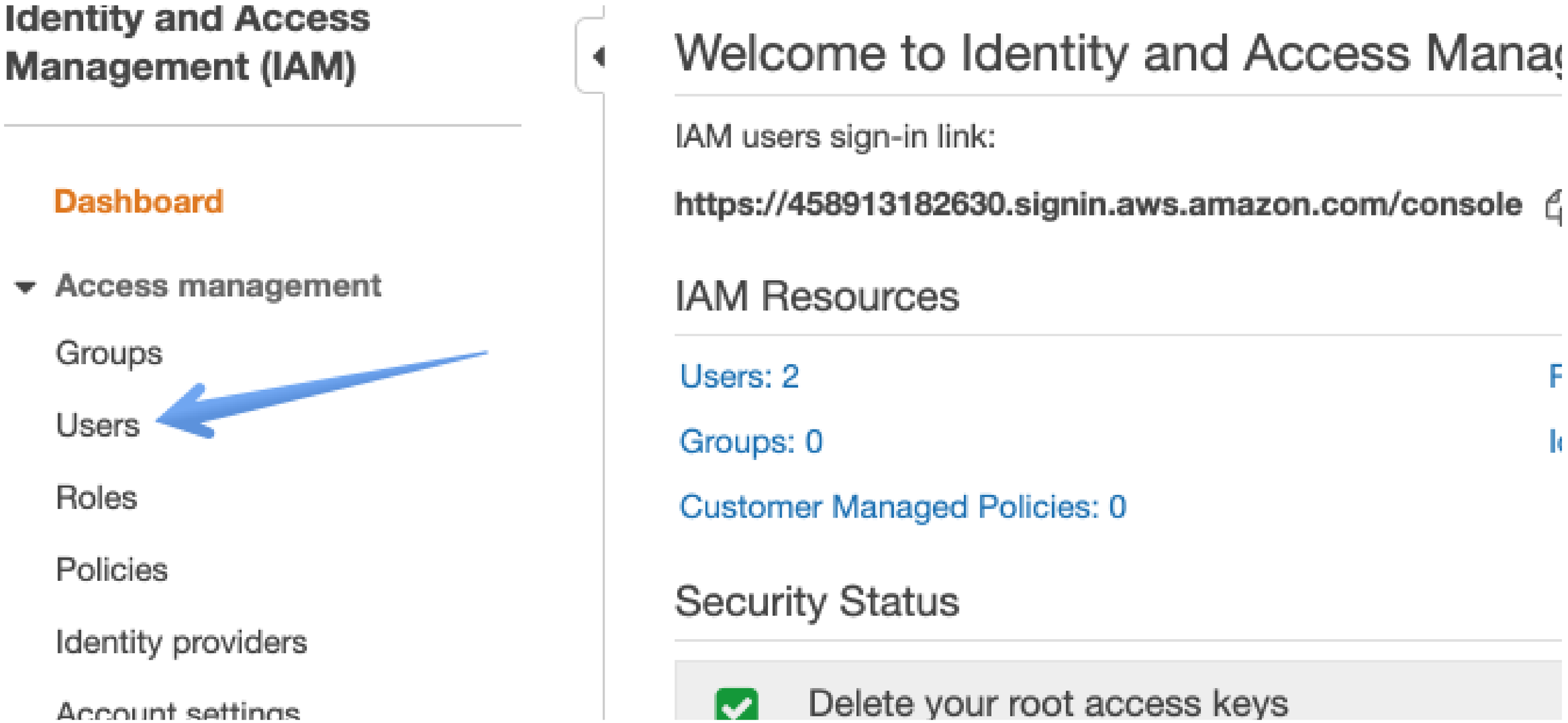

dcUser
AWS_KEY_ID
AWS_SECRET
vehicle sensors
write to
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data


dcUser
AWS_KEY_ID
AWS_SECRET
vehicle sensors
write to
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

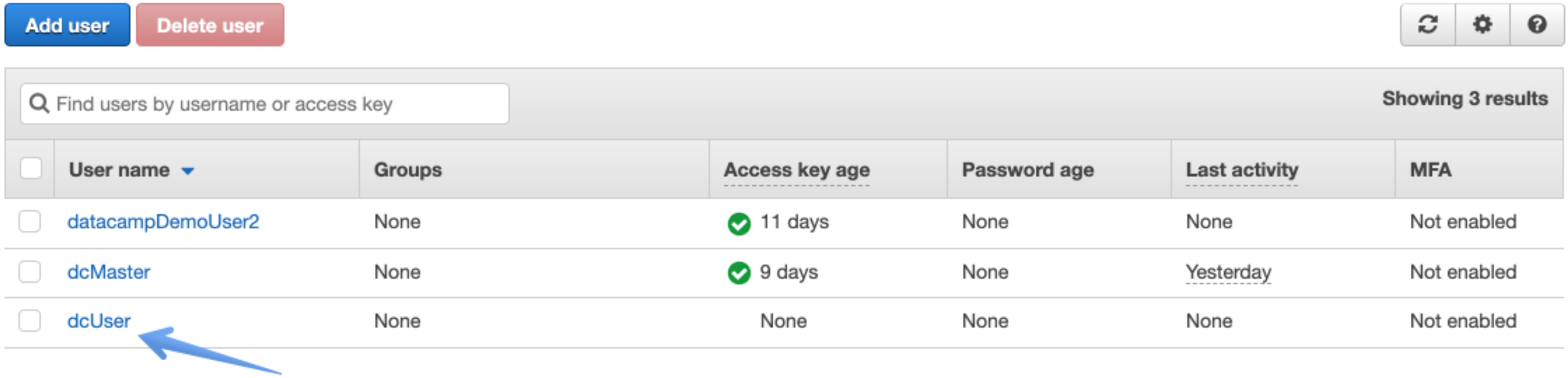
Add permissions to user



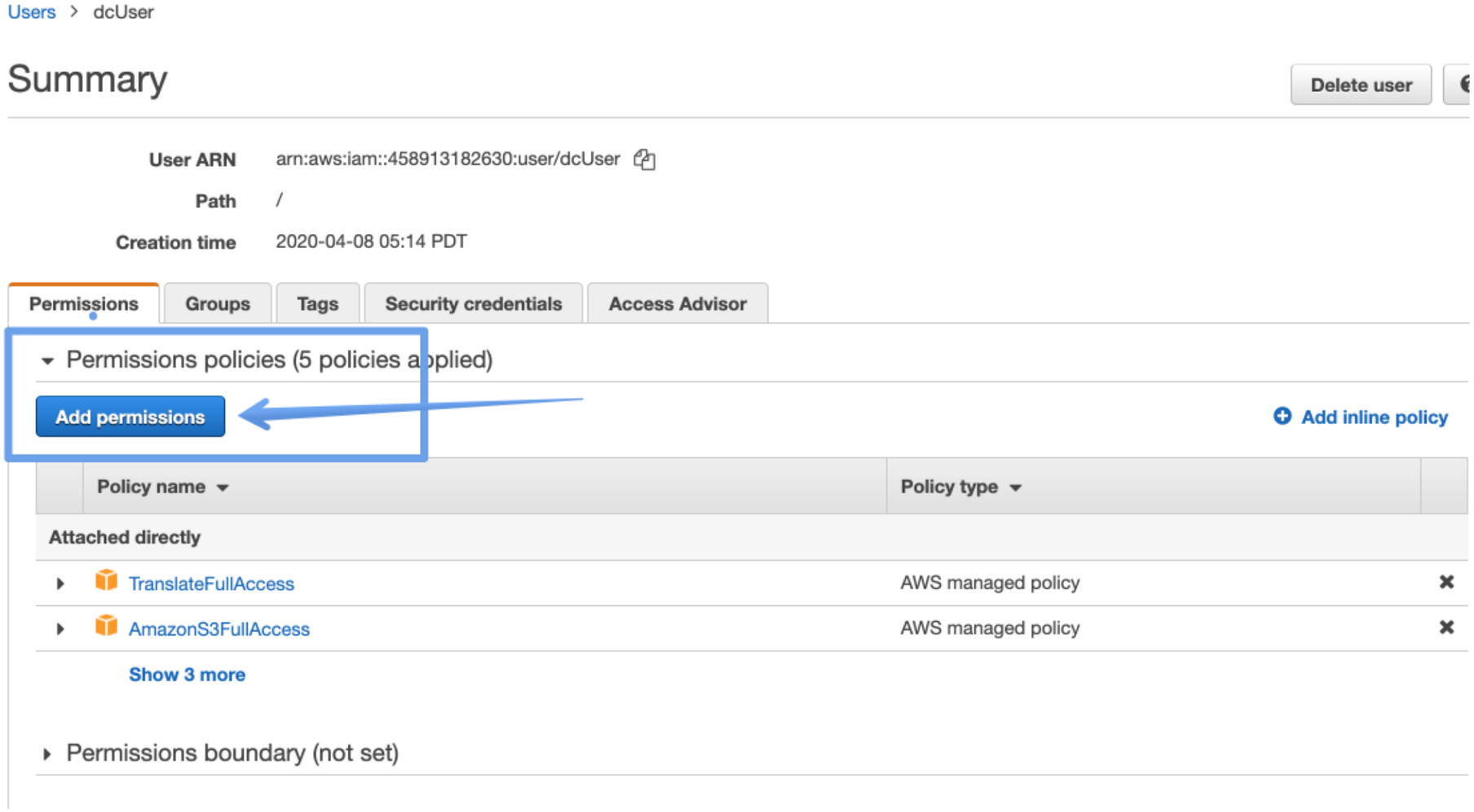
Add permissions to user



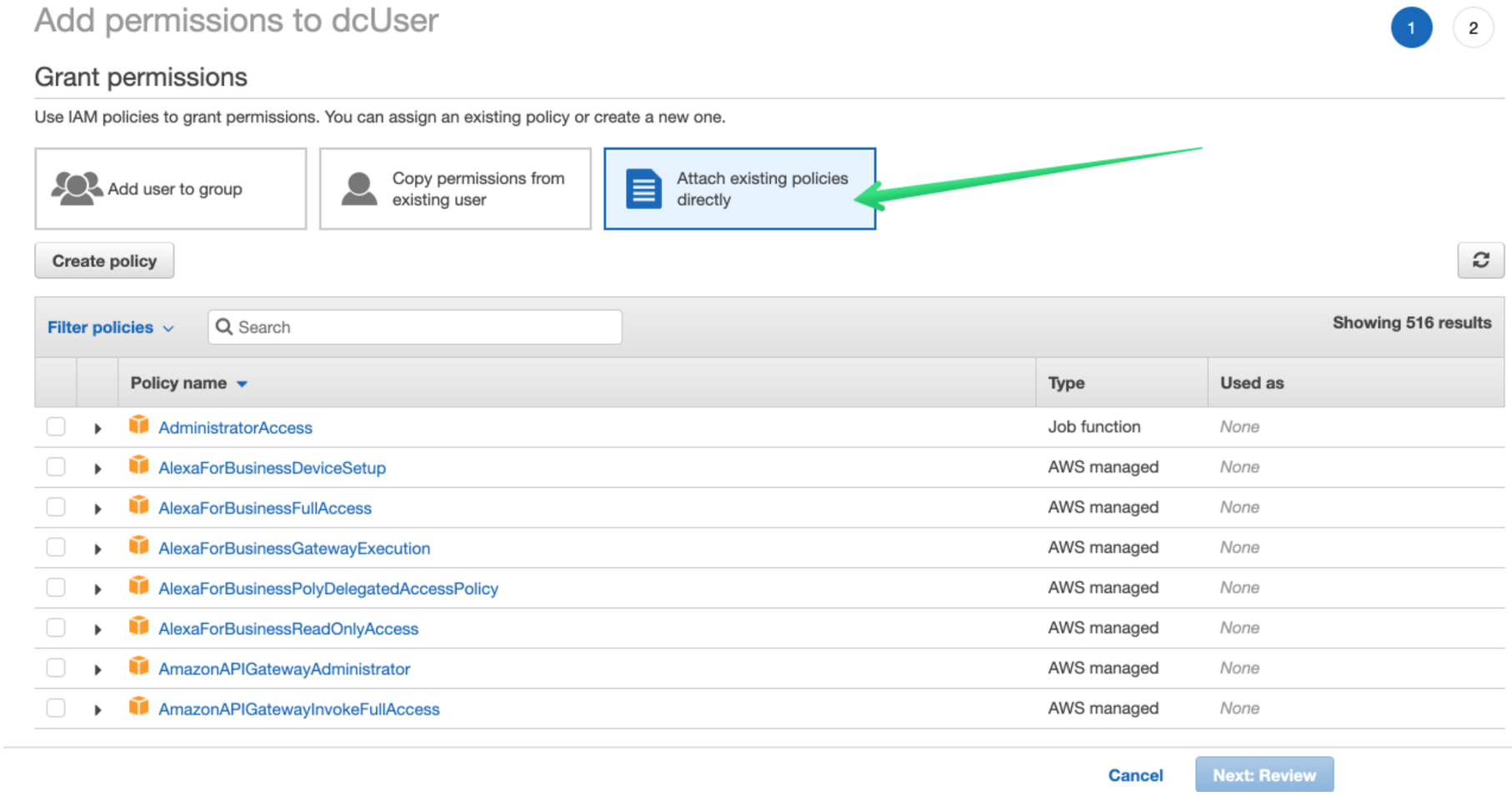
Add permissions to user



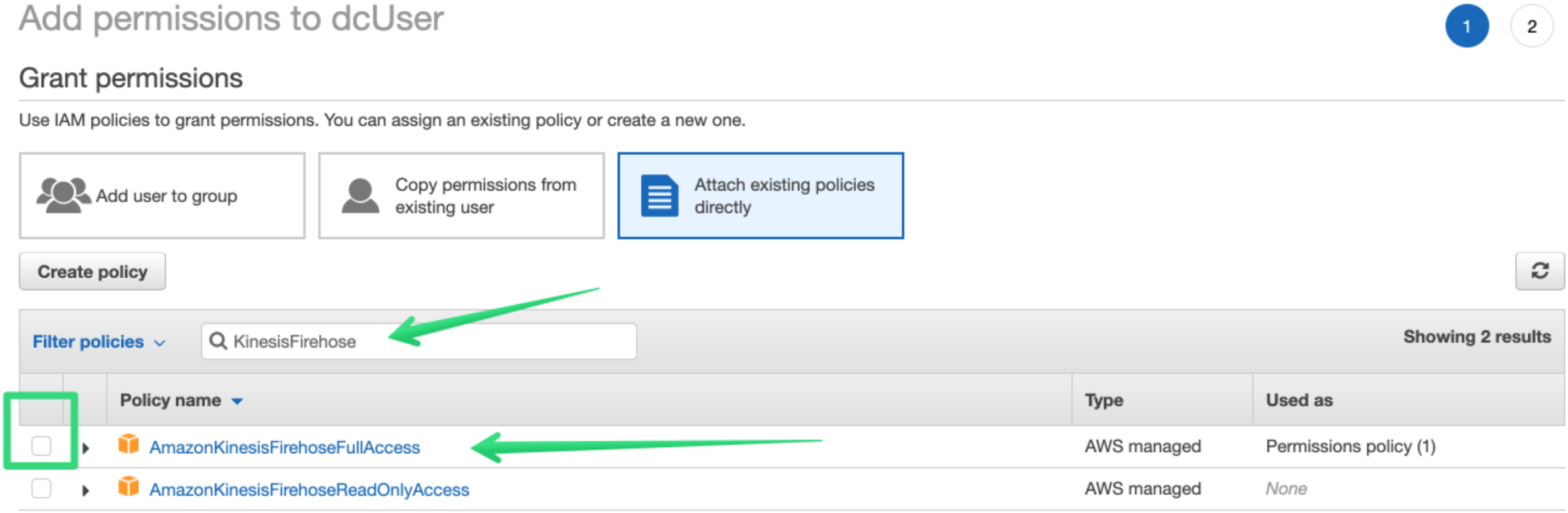
Add permissions to user



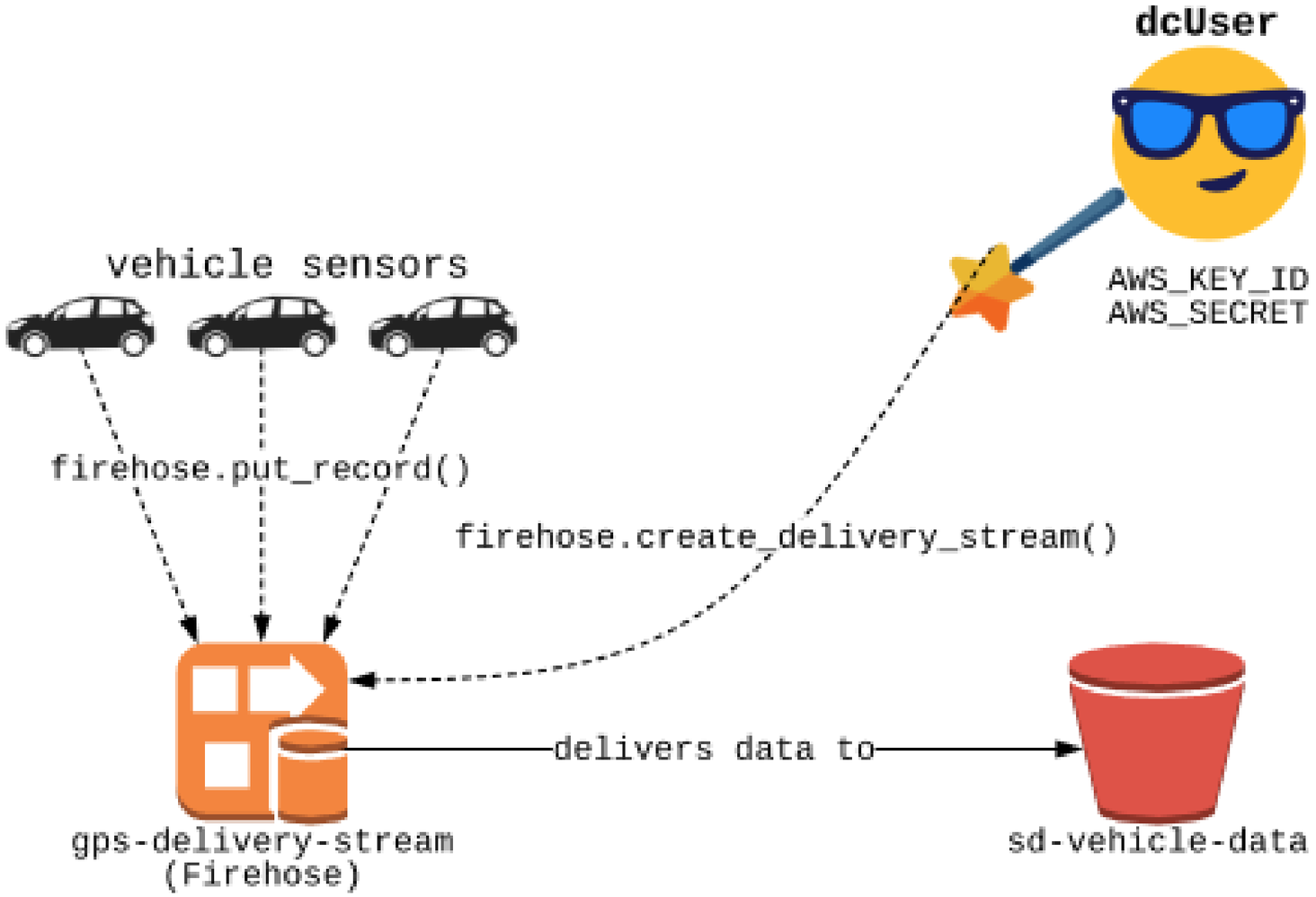
Add permissions to user



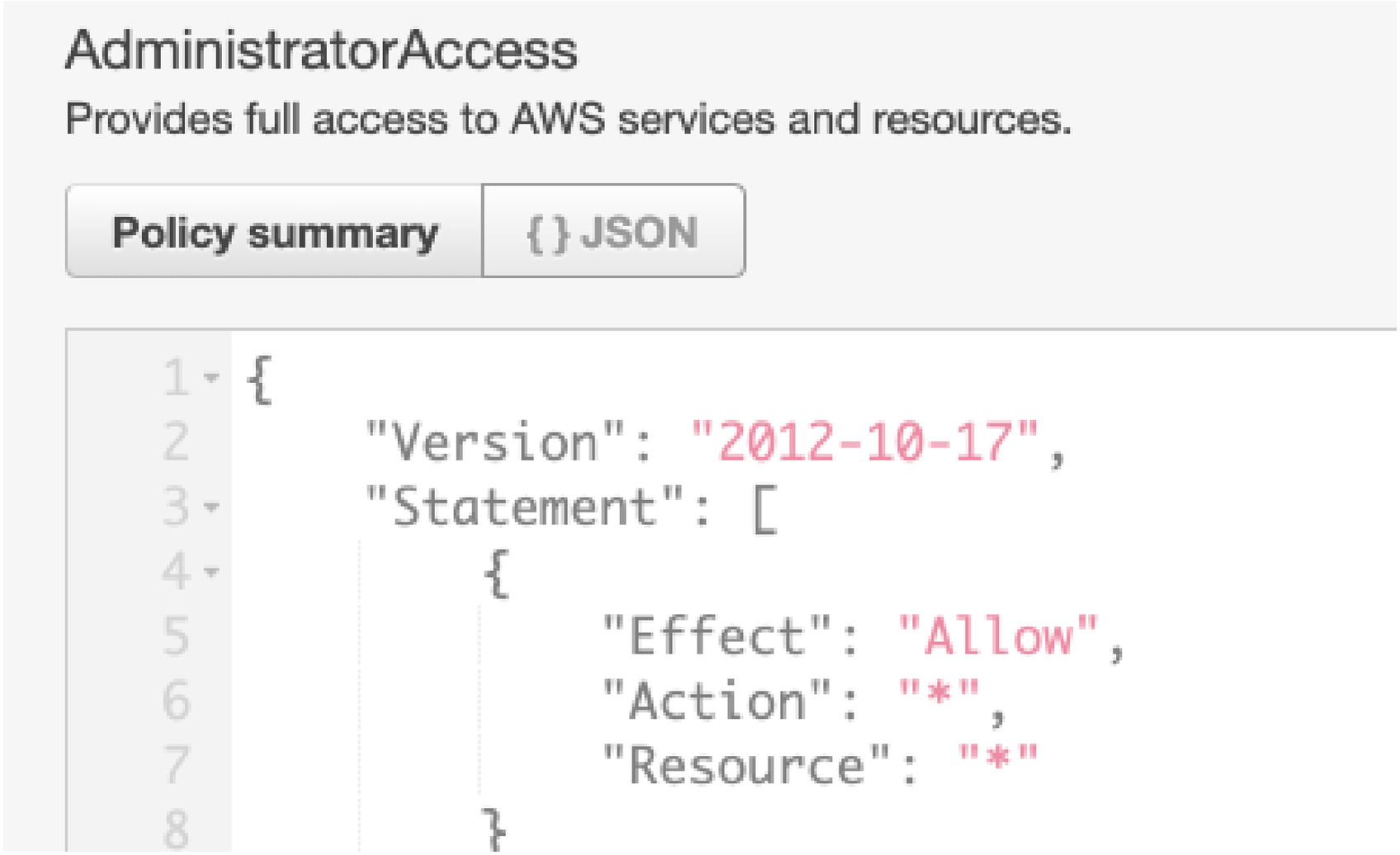
Add permissions to user



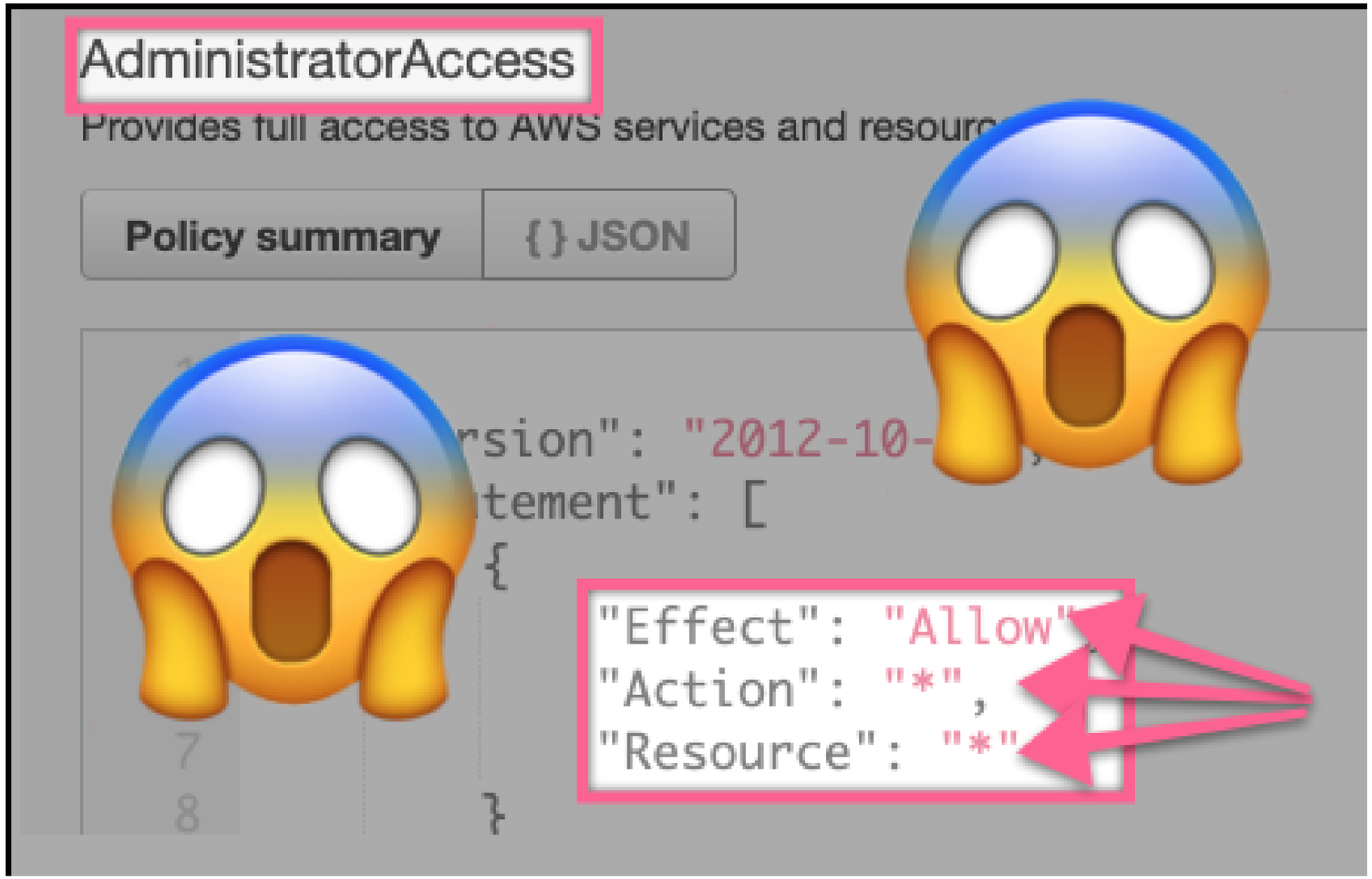
New powers


dcUser
AWS_KEY_ID
AWS_SECRET
vehicle sensors
firehose.put_record()
firehose.create_delivery_stream()
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

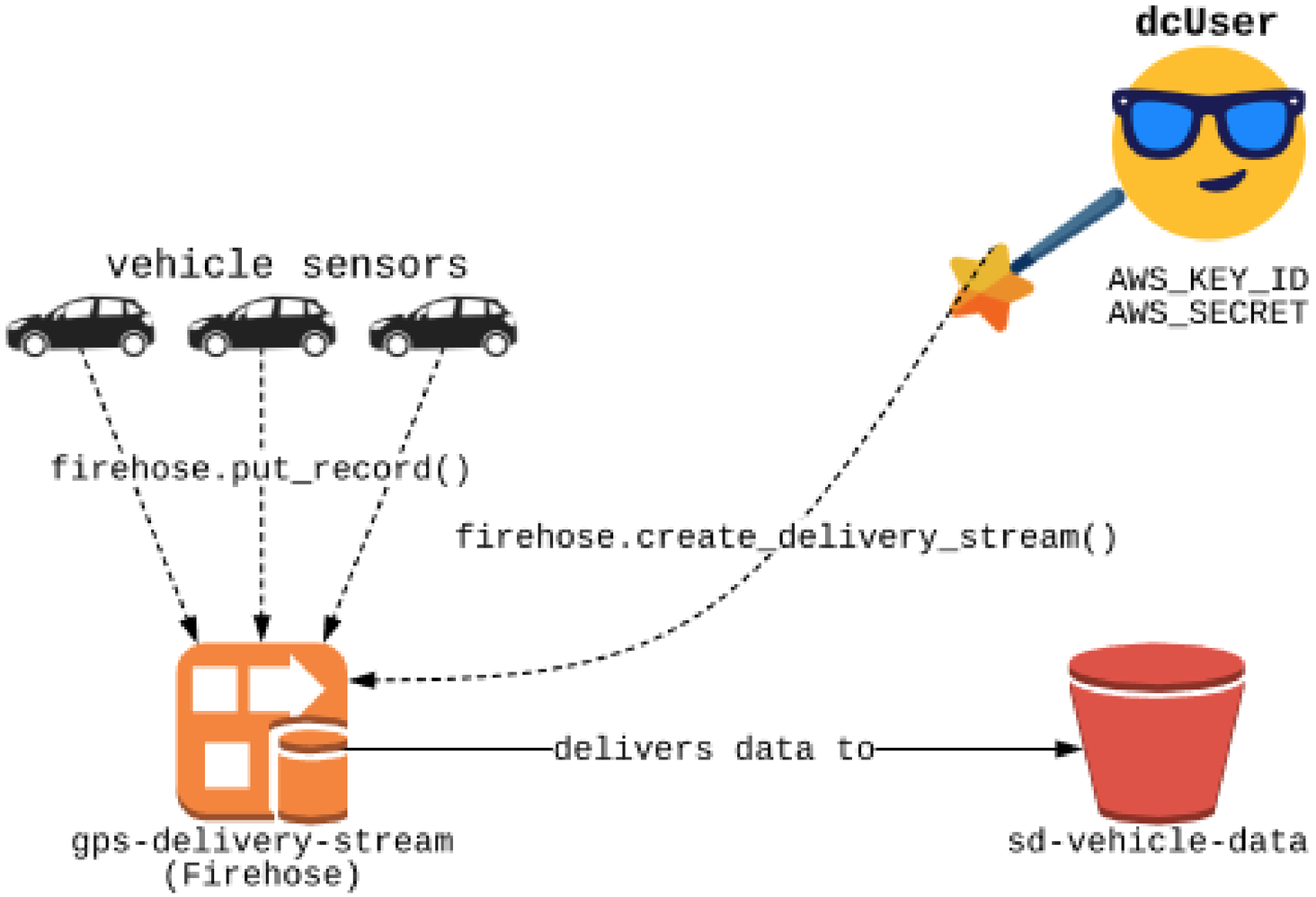
A note on security

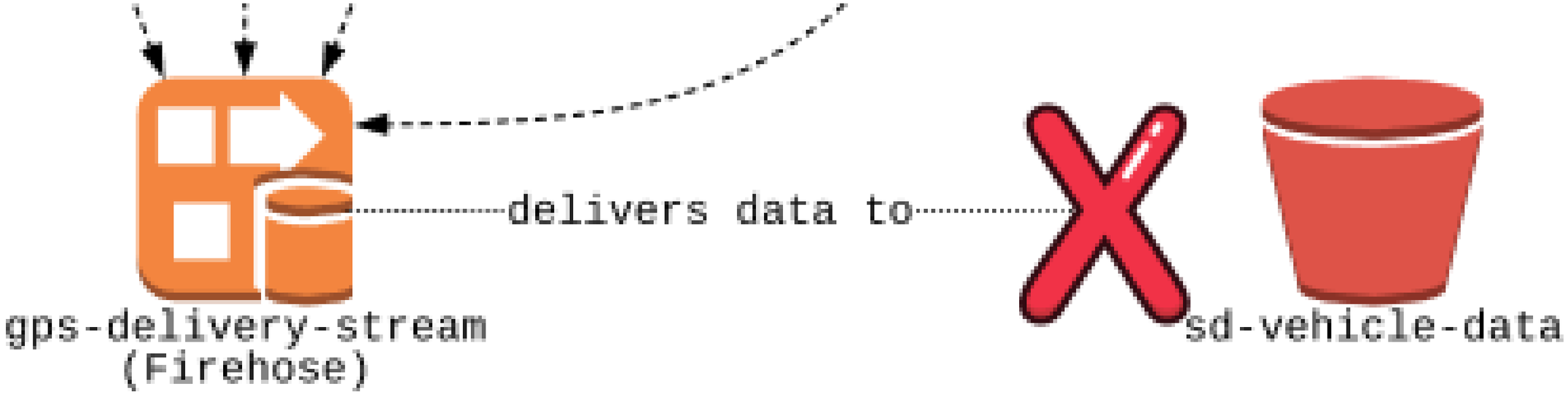


A note on security

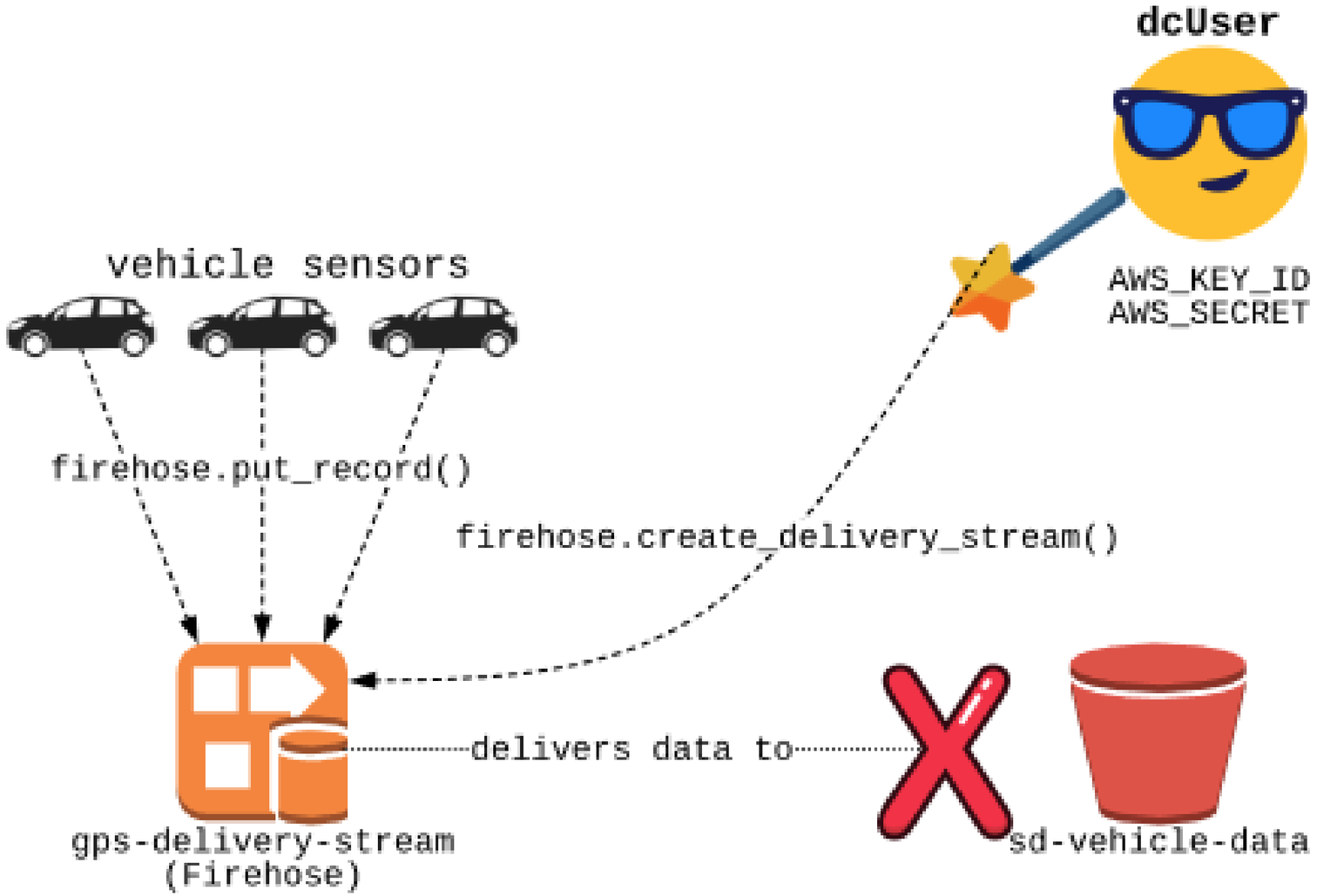


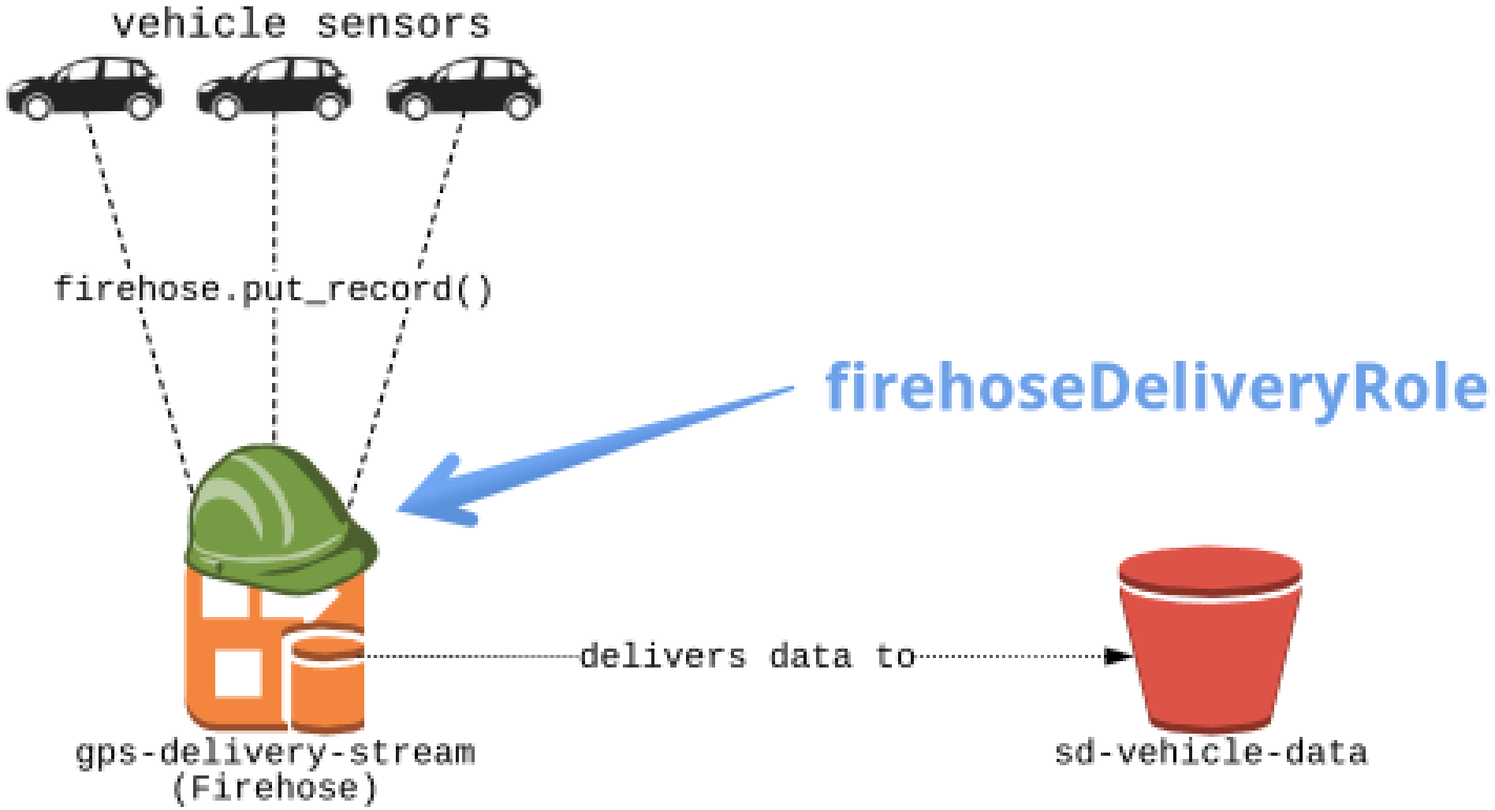
New powers

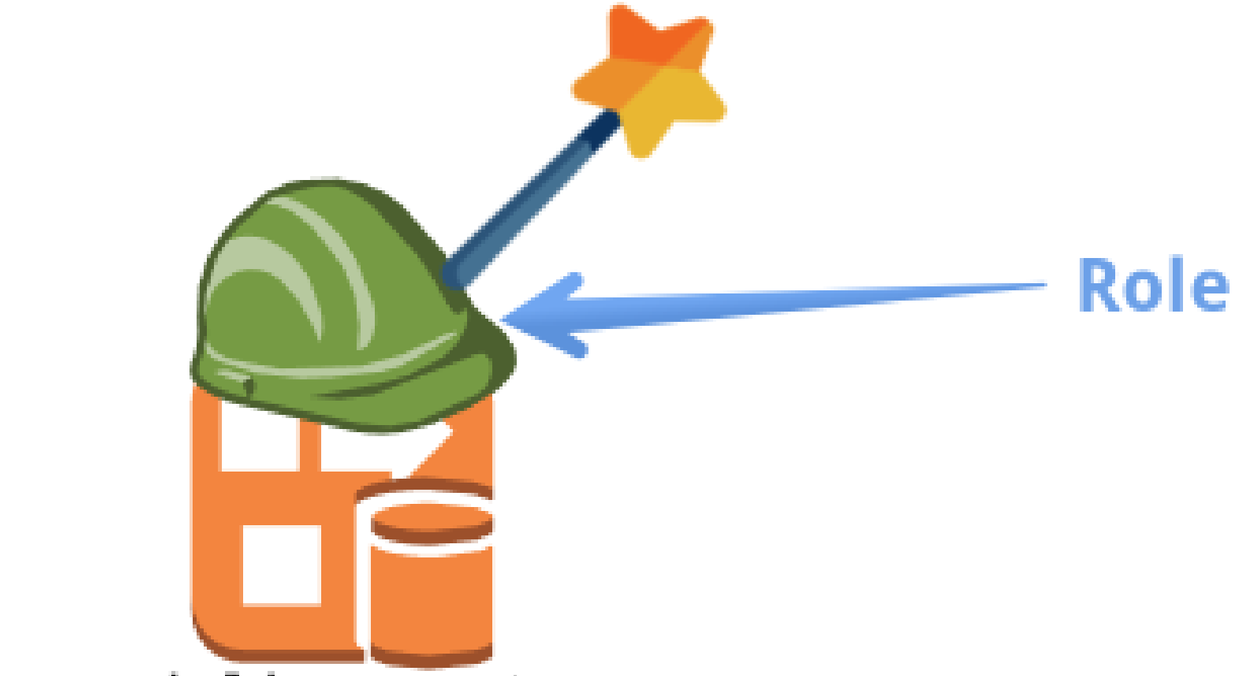

dcUser
AWS_KEY_ID
AWS_SECRET
vehicle sensors
firehose.put_record()
firehose.create_delivery_stream()
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

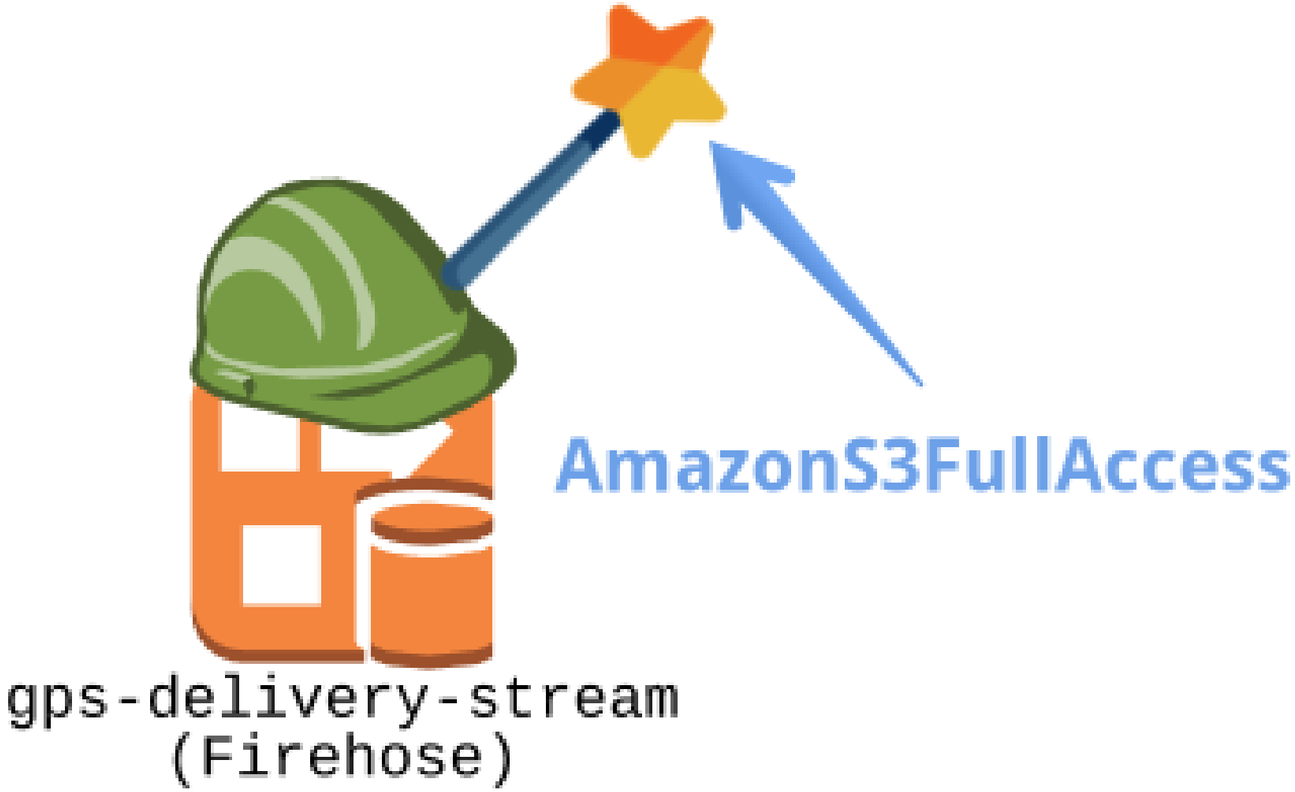

delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

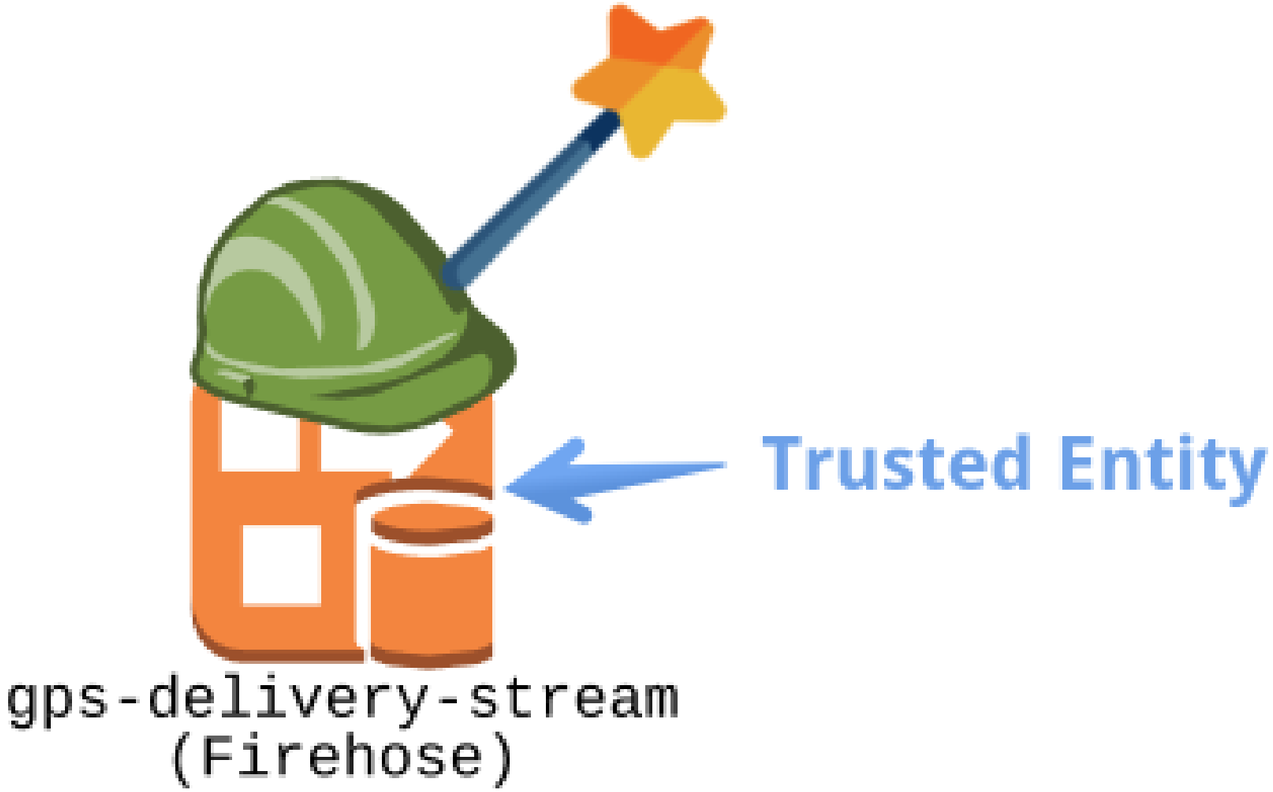
Firehose stream permissions




vehicle sensors
firehose.put_record()
firehoseDeliveryRole
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data


Role
gps-delivery-stream
(Firehose)


AmazonS3FullAccess
gps-delivery-stream
(Firehose)

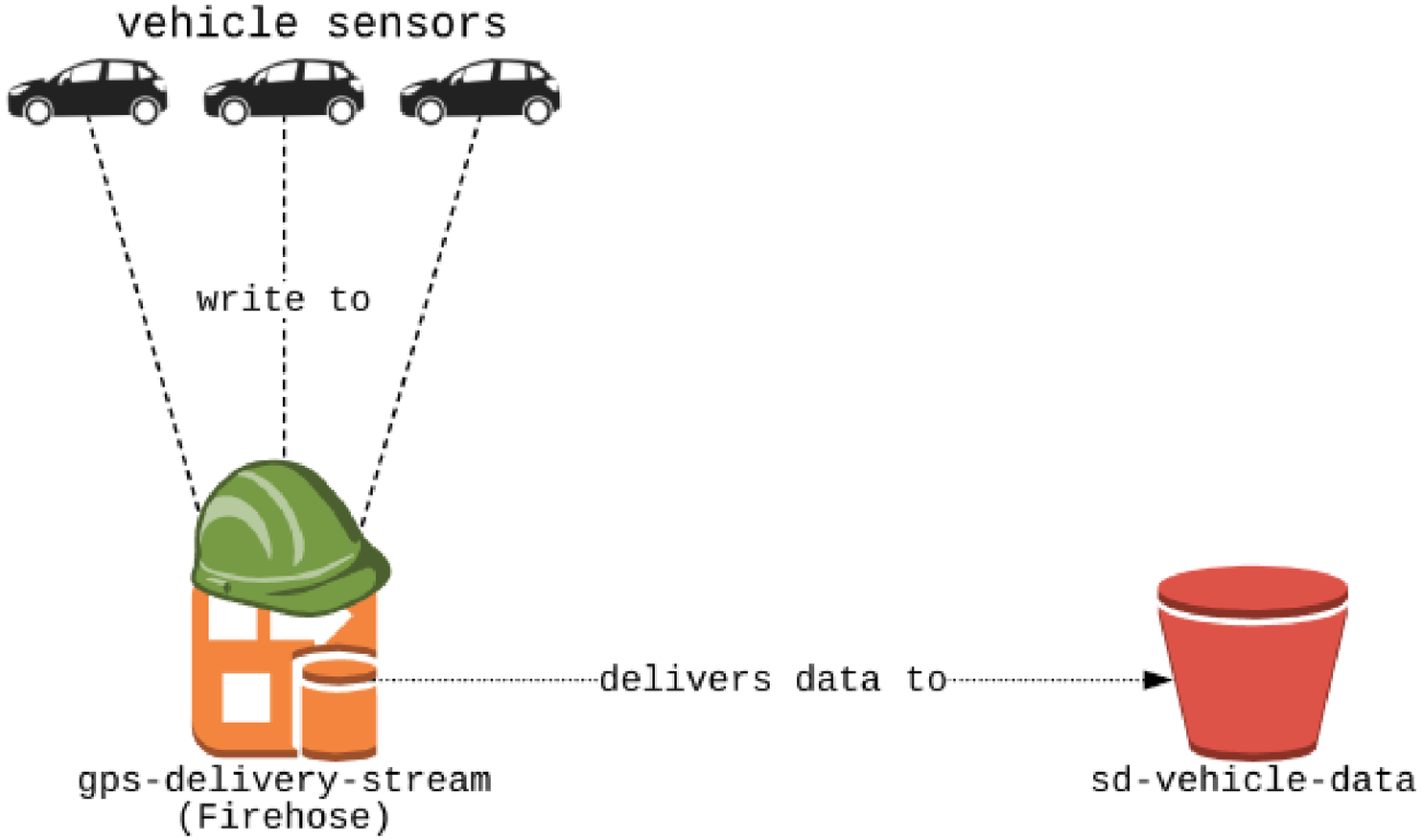

Trusted Entity
gps-delivery-stream
(Firehose)

Roles vs users

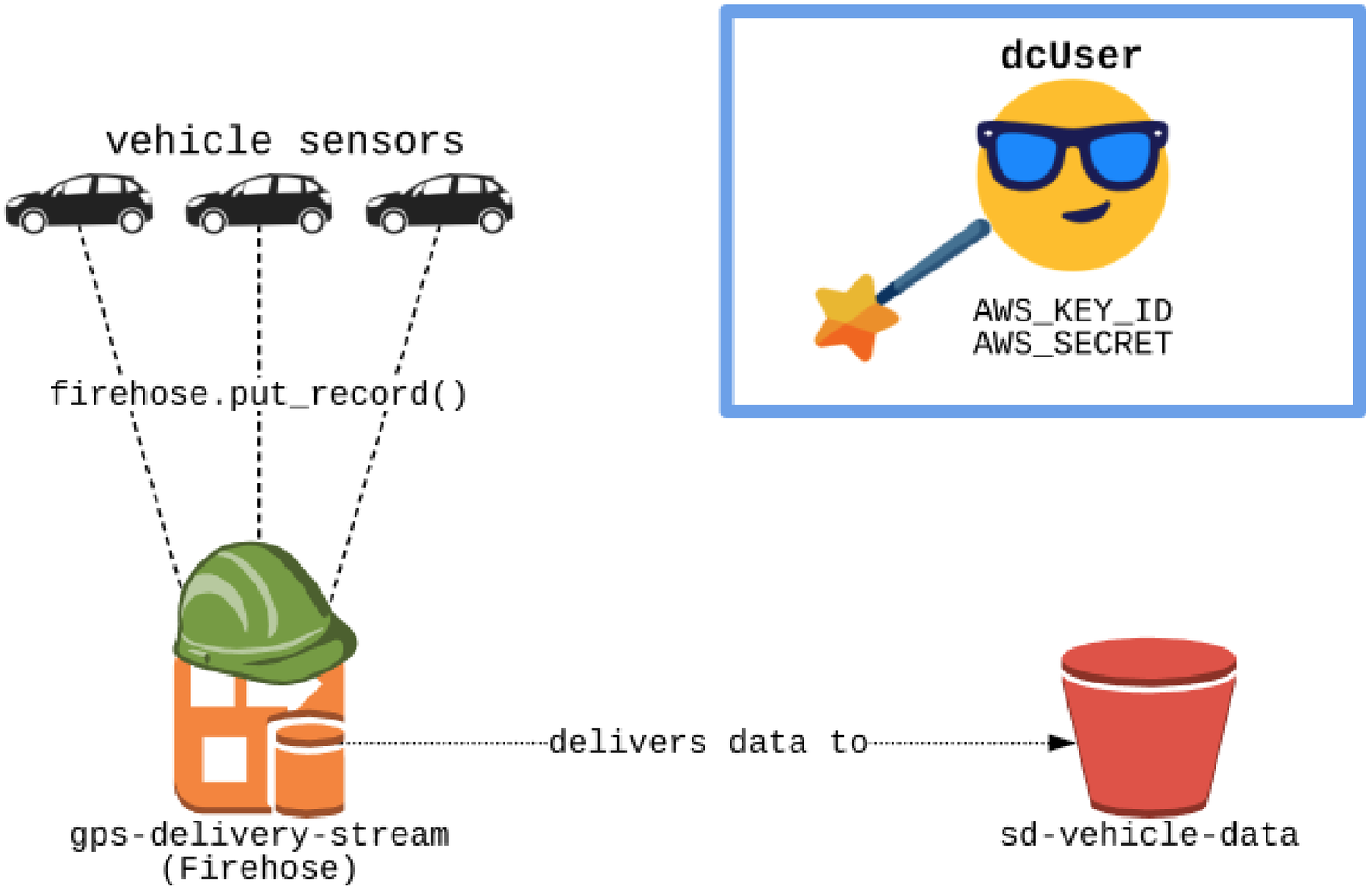
- Live in IAM
- Have permissions policies
- **Only attach to other services**
- **Do not have keys nor login**

- Live in IAM
- Have permissions policies
- **Can act on their own**
- **Can have keys or login**

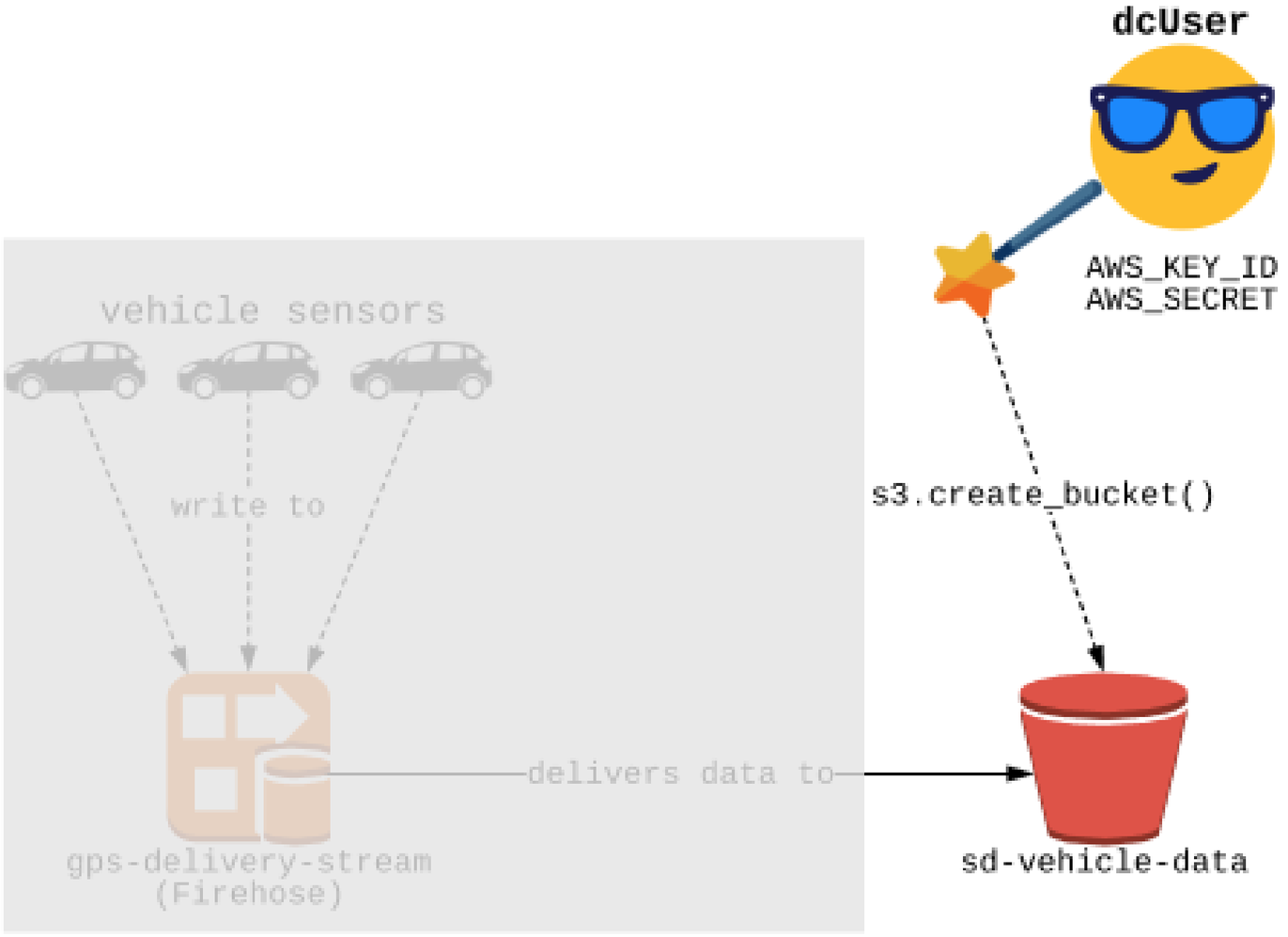
Review - end goal



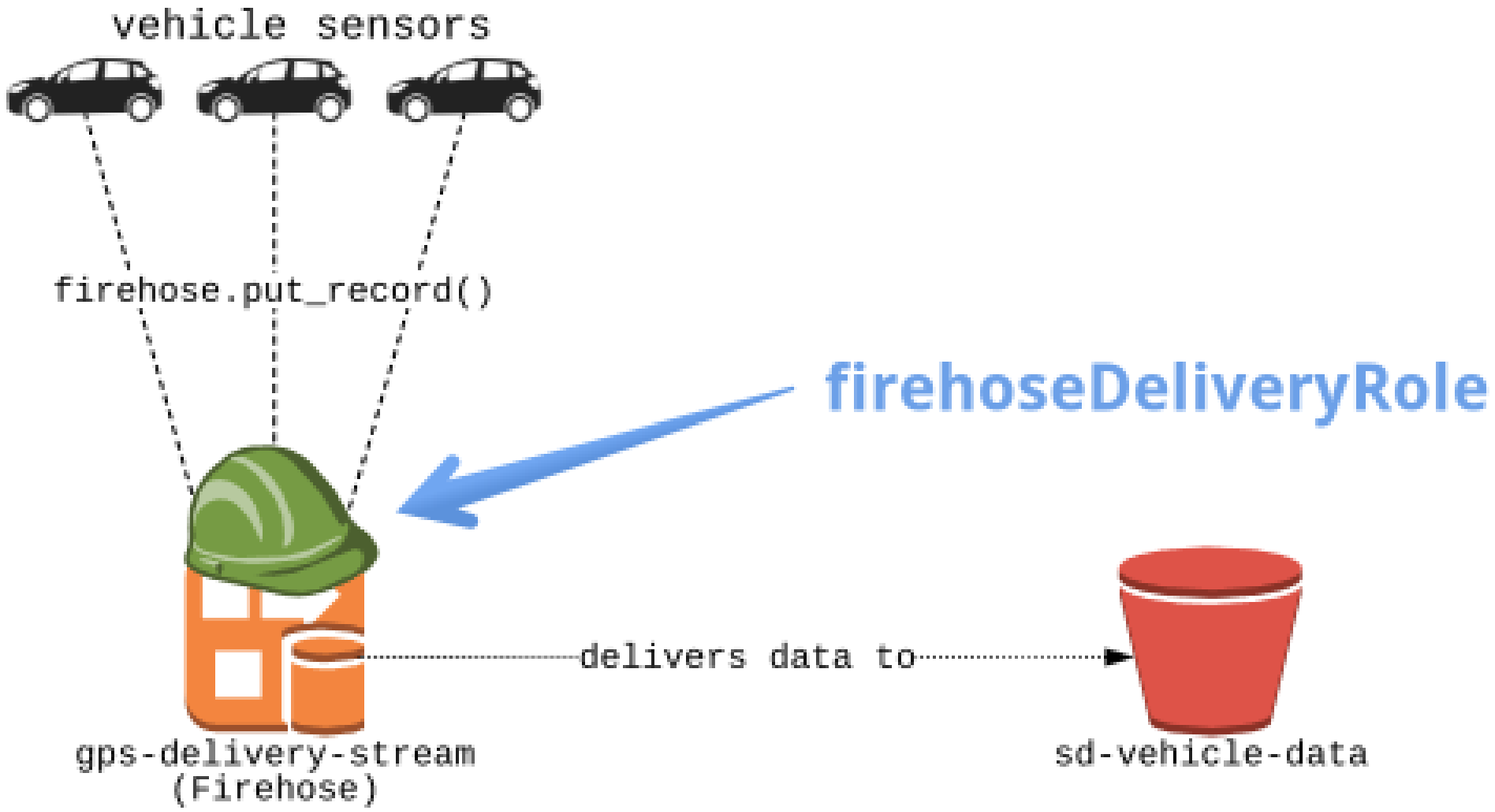
Review


vehicle sensors
firehose.put_record()
dcUser
AWS_KEY_ID
AWS_SECRET
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

Review


dcUser
AWS_KEY_ID
AWS_SECRET
s3.create_bucket()
vehicle sensors
write to
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

Review


vehicle sensors
firehose.put_record()
firehoseDeliveryRole
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

Let's do it!

STREAMING DATA WITH AWS KINESIS AND LAMBDA

Creating roles

STREAMING DATA WITH AWS KINESIS AND LAMBDA

Maksim Pecherskiy
Data Engineer

datacamp

Let's practice!

STREAMING DATA WITH AWS KINESIS AND LAMBDA

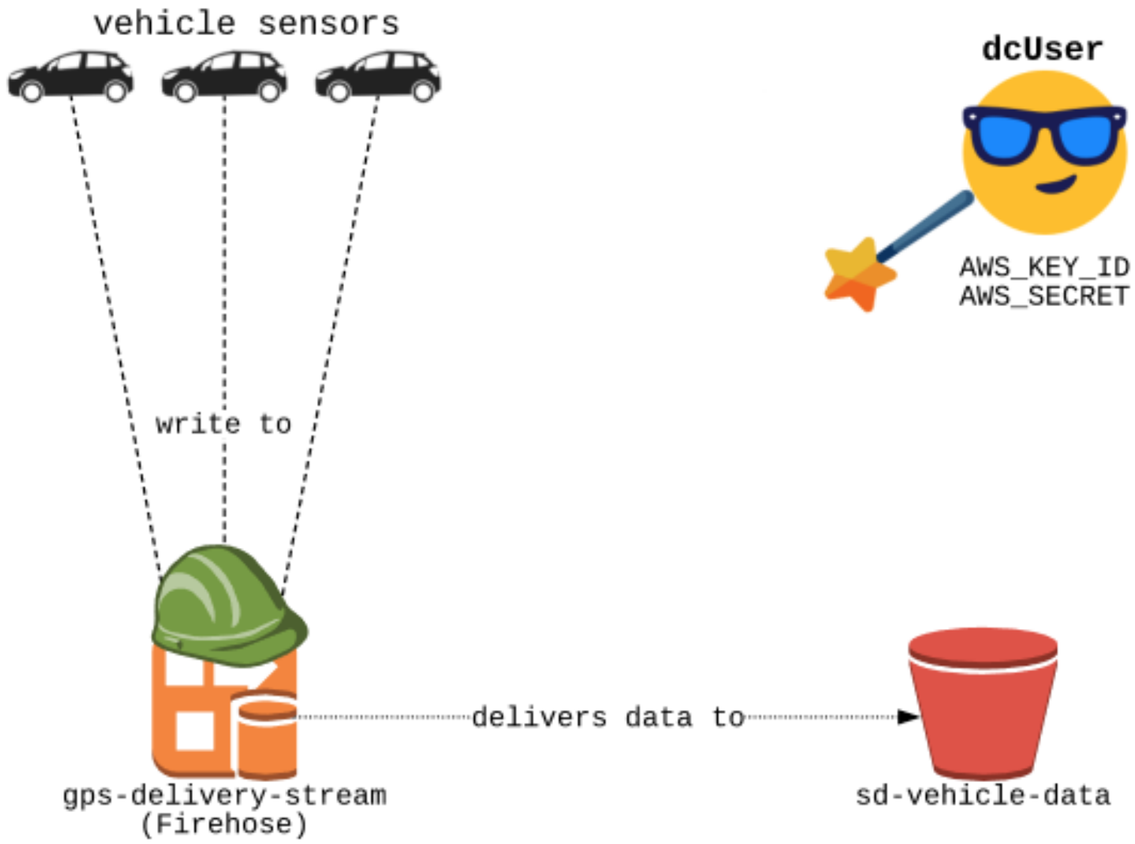
Working with the Firehose delivery stream

STREAMING DATA WITH AWS KINESIS AND LAMBDA

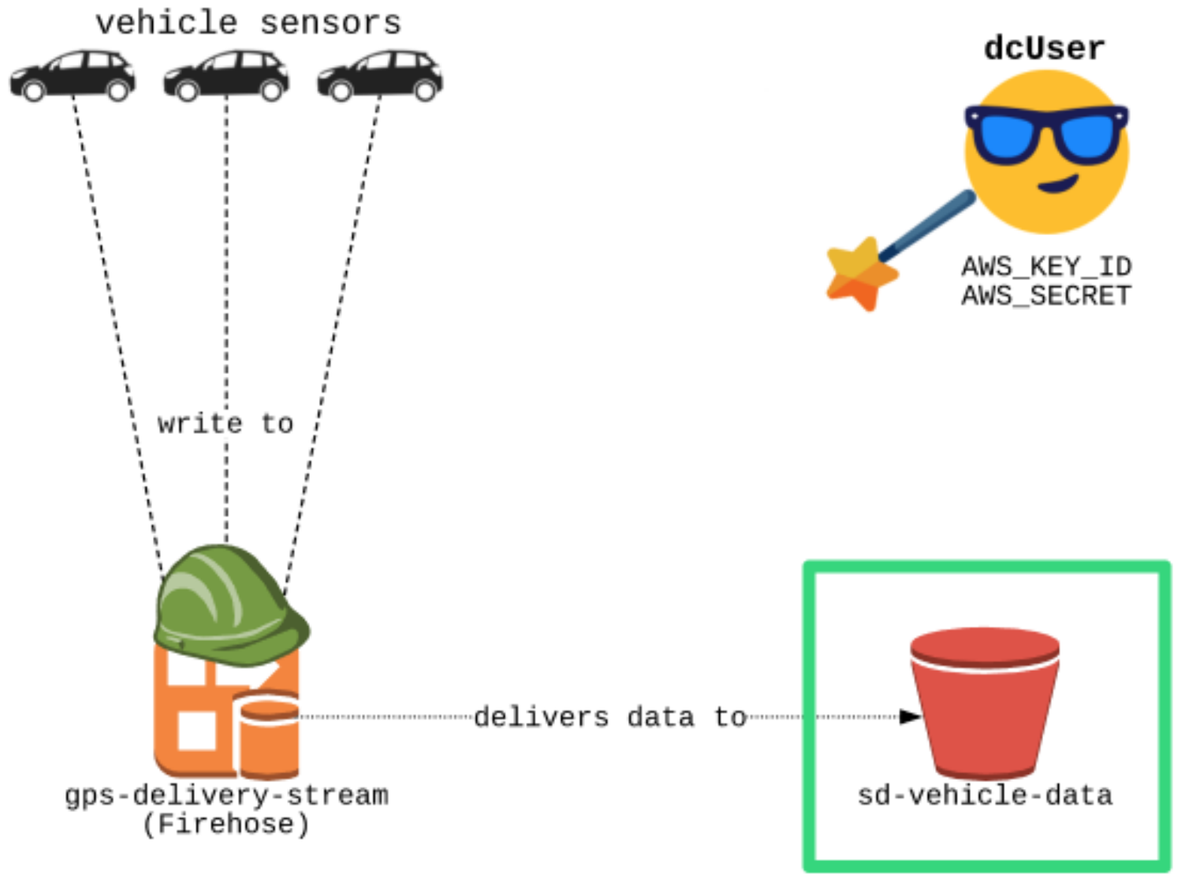
Maksim Pecherskiy
Data Engineer

datacamp

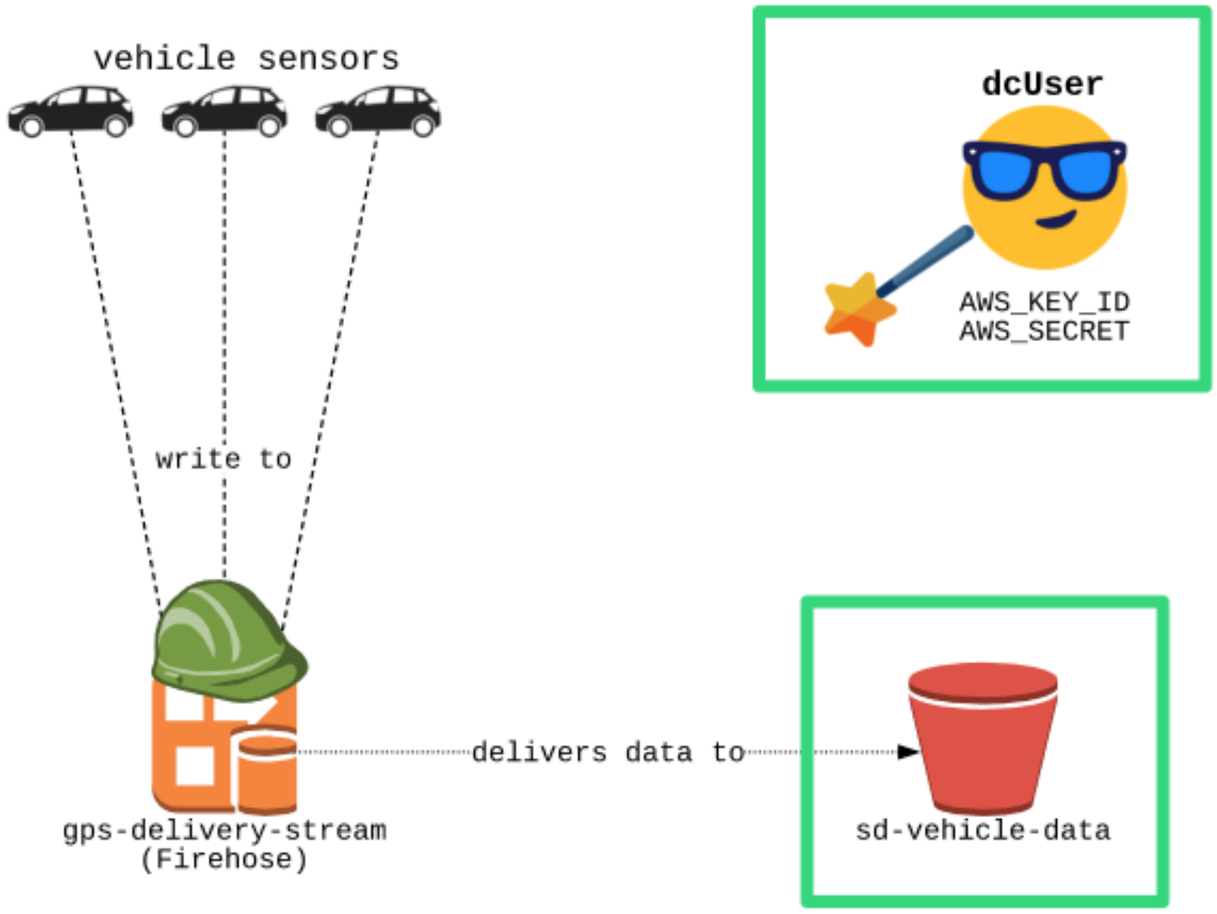
Ready to create stream



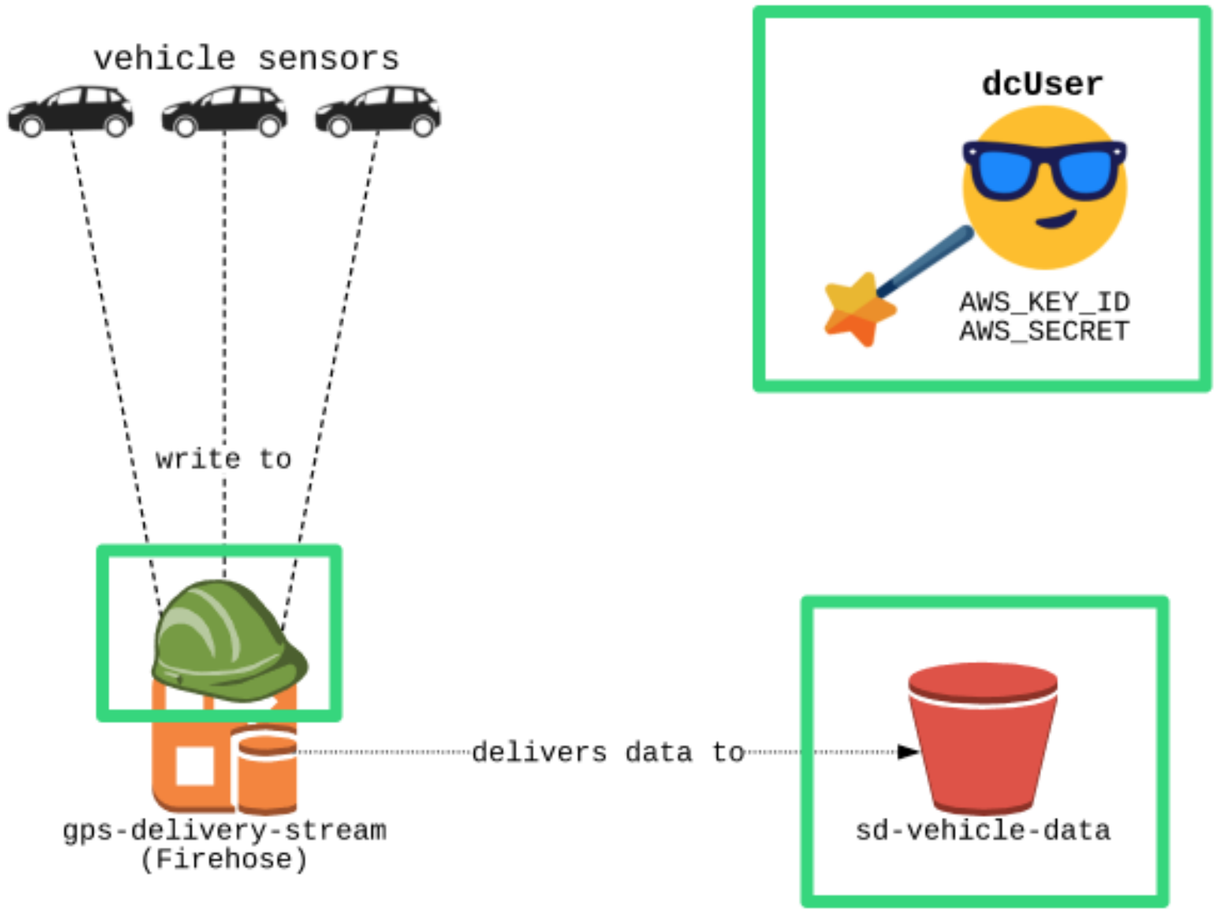
Ready to create stream



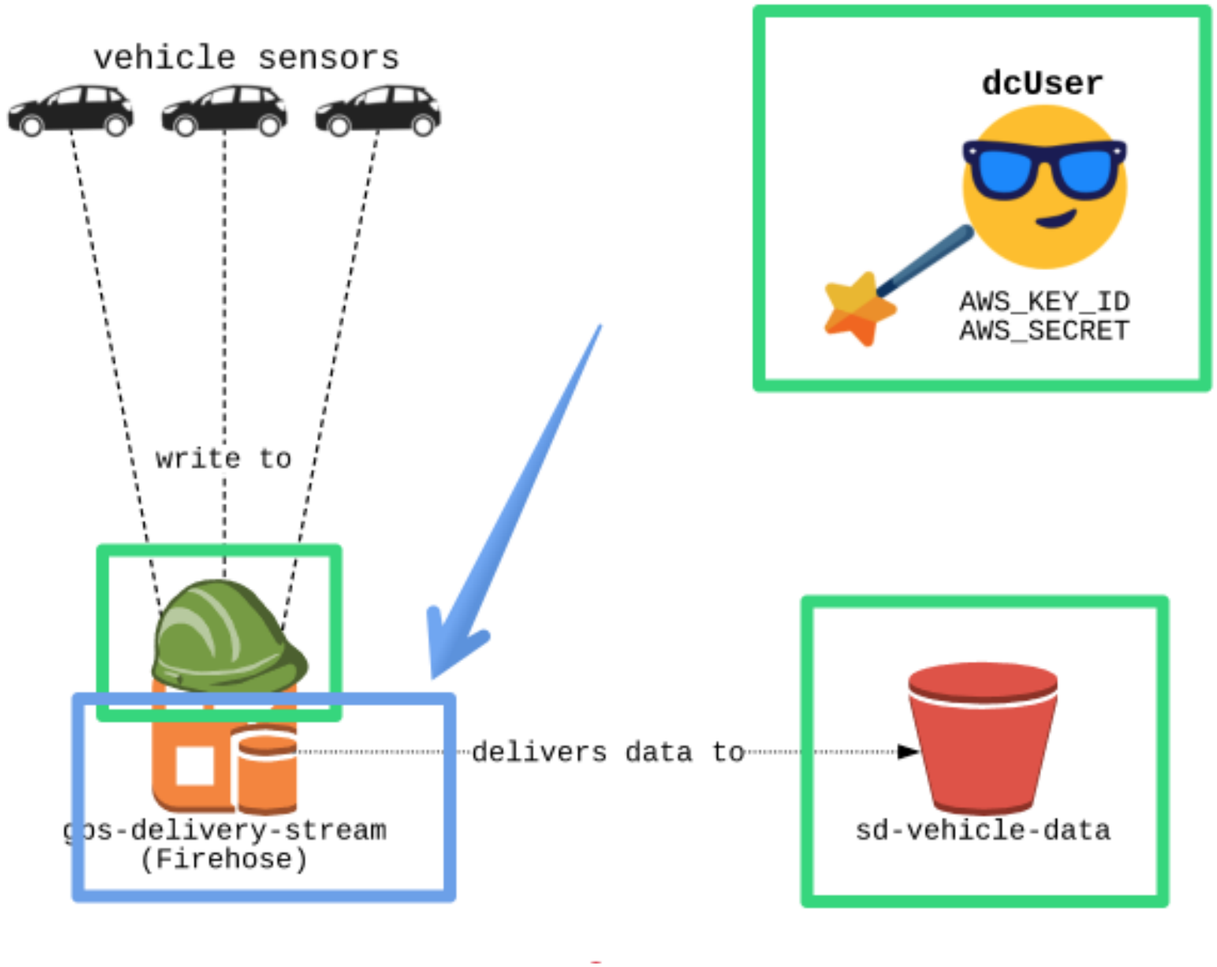
Ready to create stream



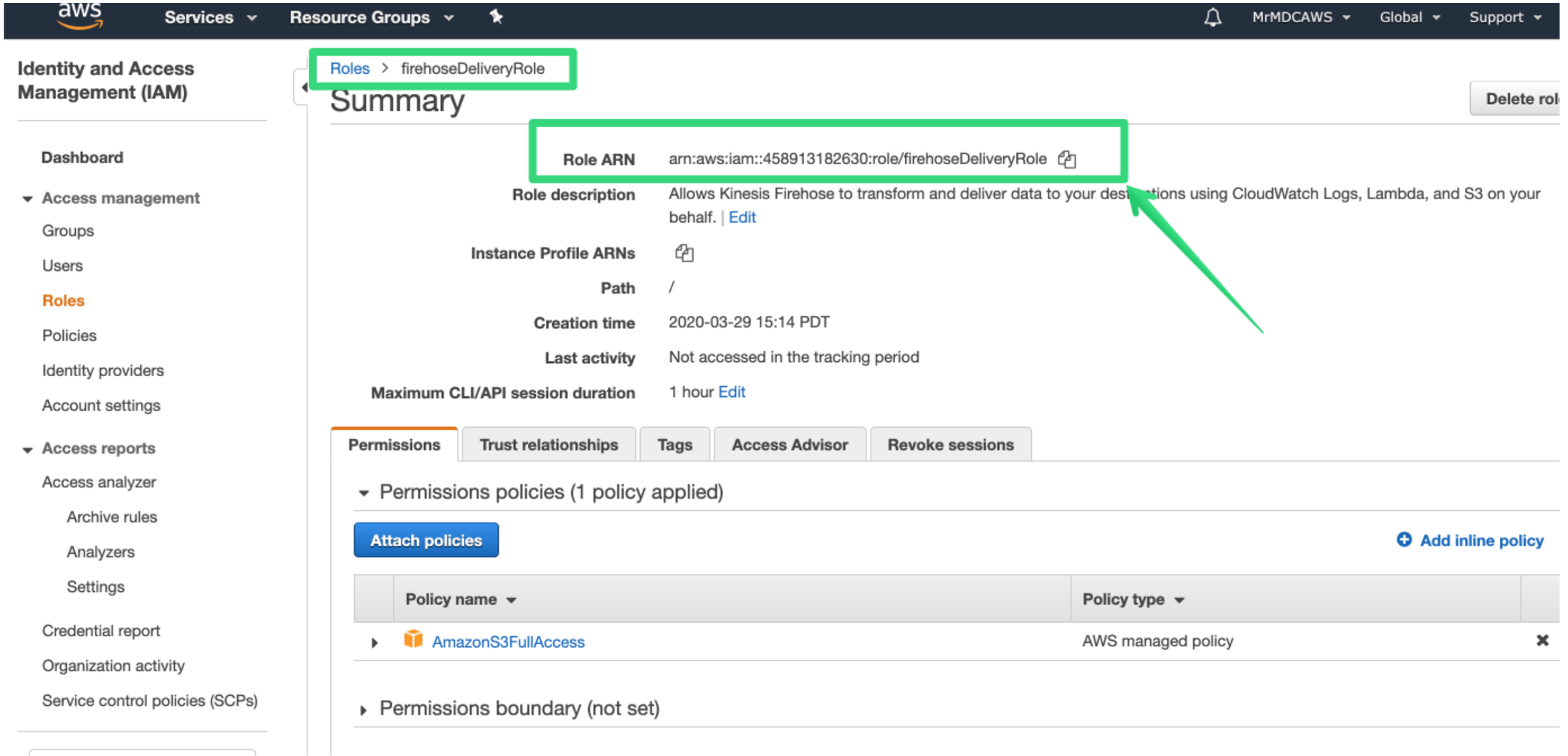
Ready to create stream



Ready to create stream



Get Role ARN


aws
Services
Resource Groups
MrMDCAWS
Global
Support
Identity and Access Management (IAM)
Dashboard
Access management
Groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access analyzer
Archive rules
Analyzers
Settings
Credential report
Organization activity
Service control policies (SCPs)
Roles > firehoseDeliveryRole
Summary
Delete rol
Role ARN
arn:aws:iam::458913182630:role/firehoseDeliveryRole
Role description
Allows Kinesis Firehose to transform and deliver data to your dest ations using CloudWatch Logs, Lambda, and S3 on your behalf. | Edit
Instance Profile ARNs
Path
/
Creation time
2020-03-29 15:14 PDT
Last activity
Not accessed in the tracking period
Maximum CLI/API session duration
1 hour Edit
Permissions
Trust relationships
Tags
Access Advisor
Revoke sessions
Permissions policies (1 policy applied)
Attach policies
Add inline policy
Policy name
Policy type
AmazonS3FullAccess
AWS managed policy
Permissions boundary (not set)

Initialize boto3 client

```
import boto3

firehose = boto3.client('firehose',
                        aws_access_key_id=AWS_KEY_ID,
                        aws_secret_access_key=AWS_SECRET,
                        region_name='us-east-1')
```

Create the stream!

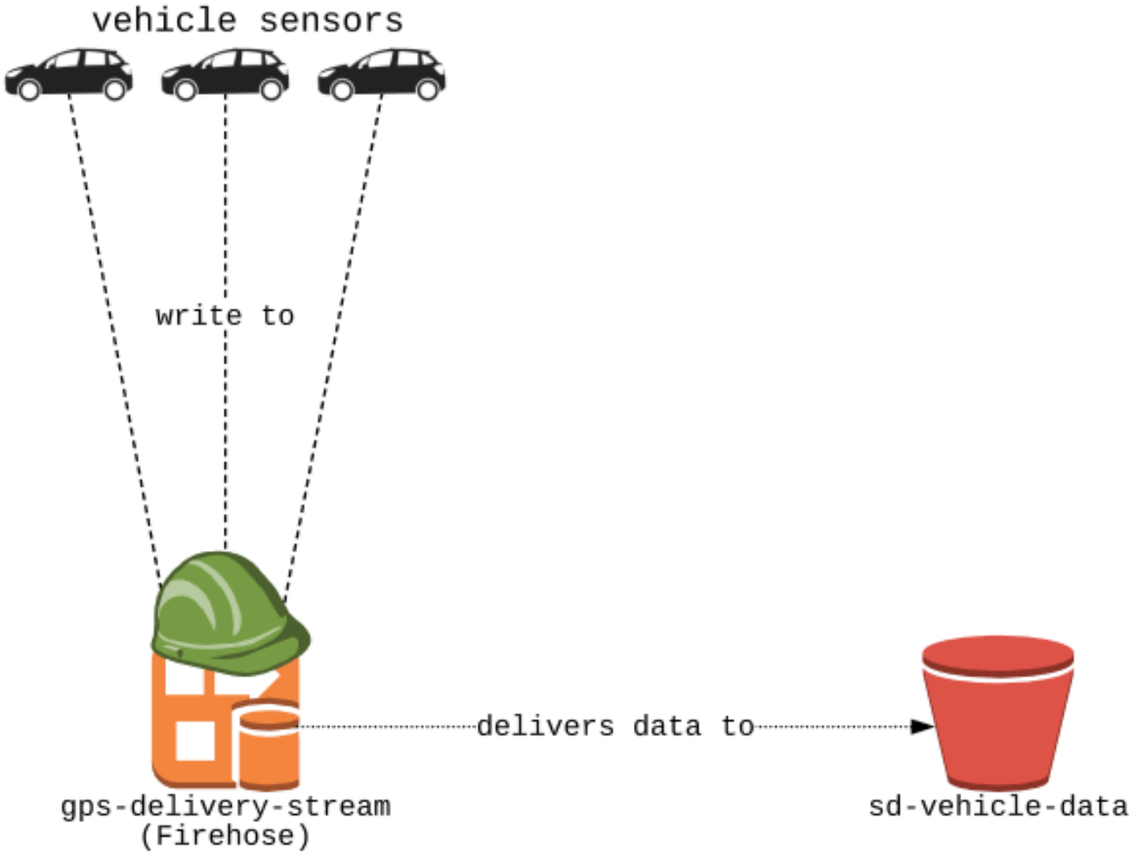
```
res = firehose.create_delivery_stream(
    DeliveryStreamName = "gps-delivery-stream",
    DeliveryStreamType = "DirectPut",
    S3DestinationConfiguration = {
        "RoleARN": "arn:aws:iam::0000000:role/firehoseDeliveryRole",
        "BucketARN": "arn:aws:s3:::sd-vehicle-data"
  }
)
```

Create stream response

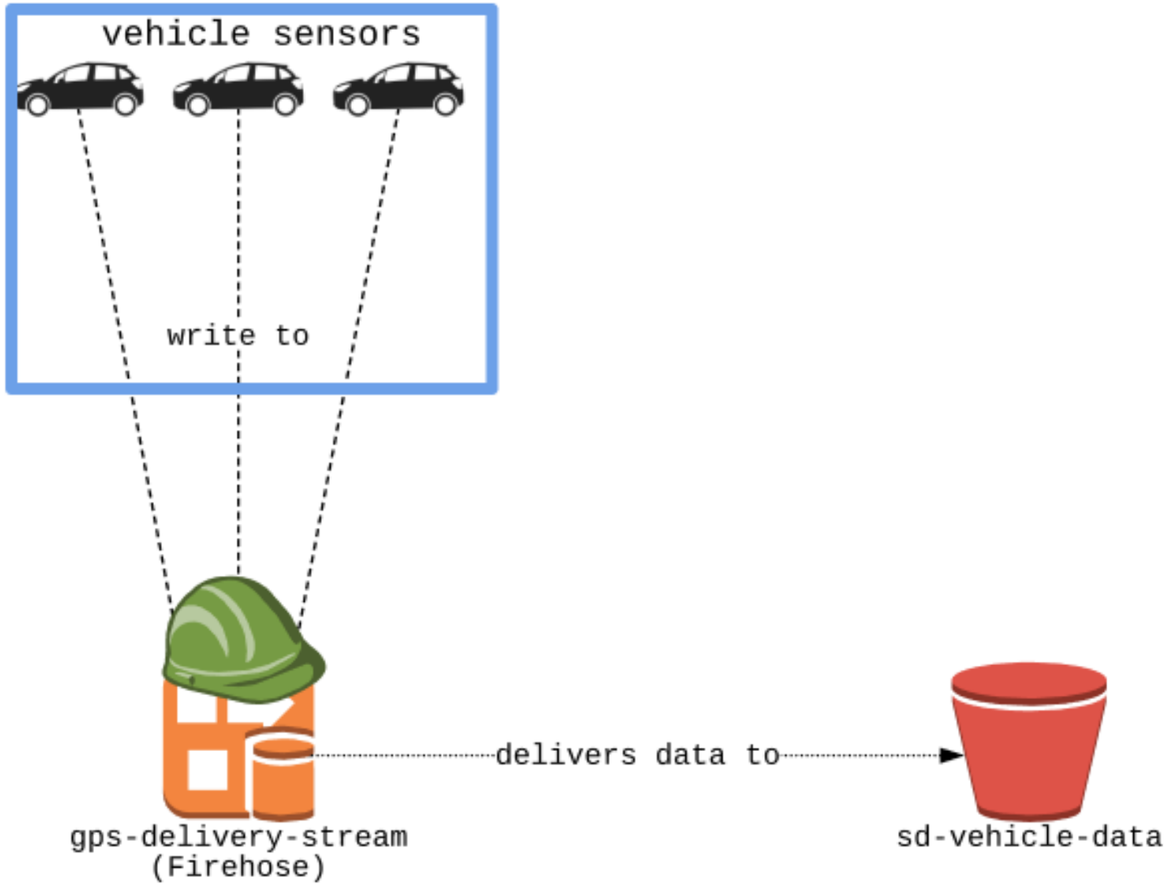
```
print(res['DeliveryStreamARN'])
```

```
# New Stream's ARN
"arn:aws:firehose:us-east-1:0000000:deliverystream/gps-delivery-stream"
```

Stream is ready

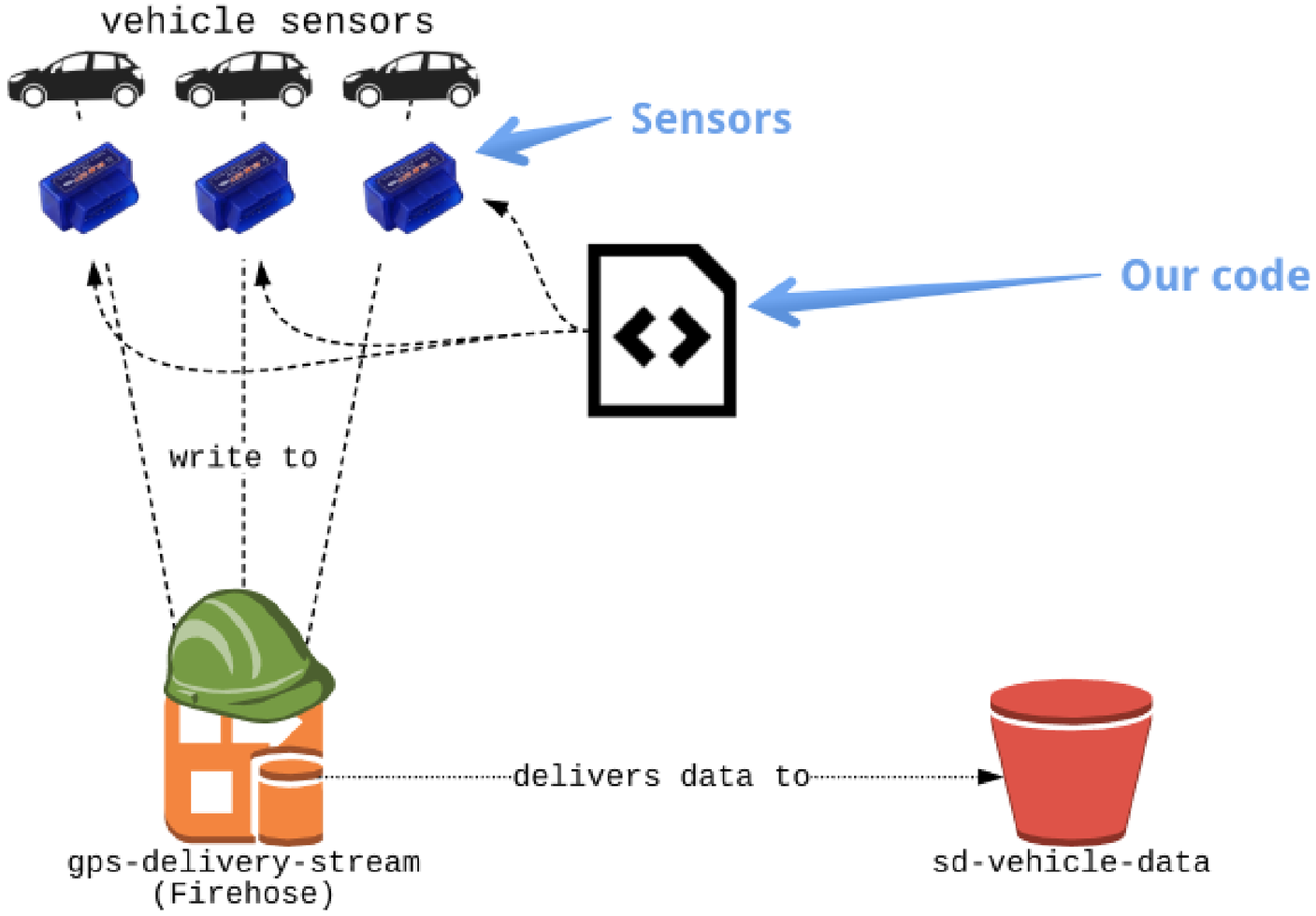


Writing to stream



Telematics hardware

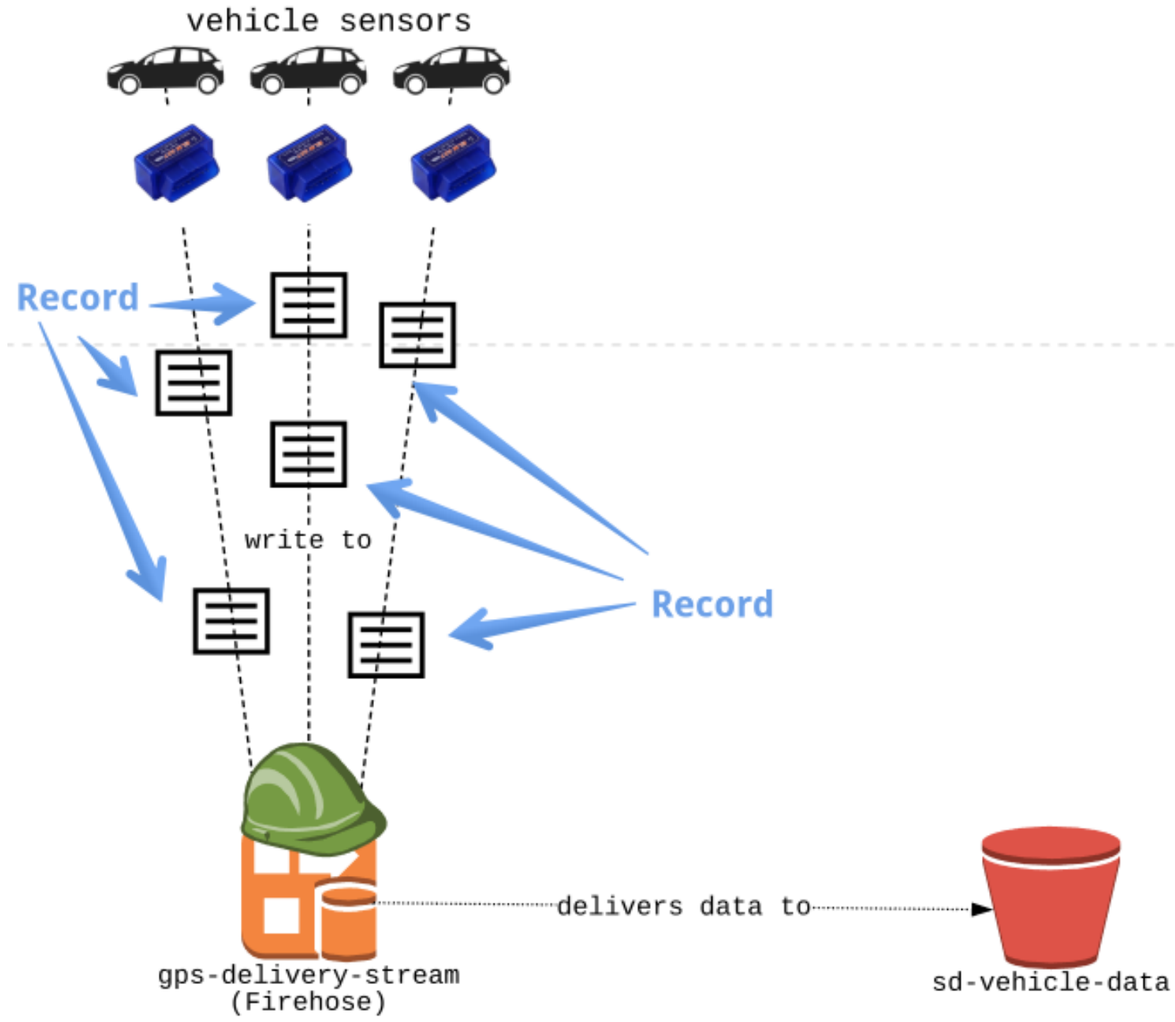
Telematics data send



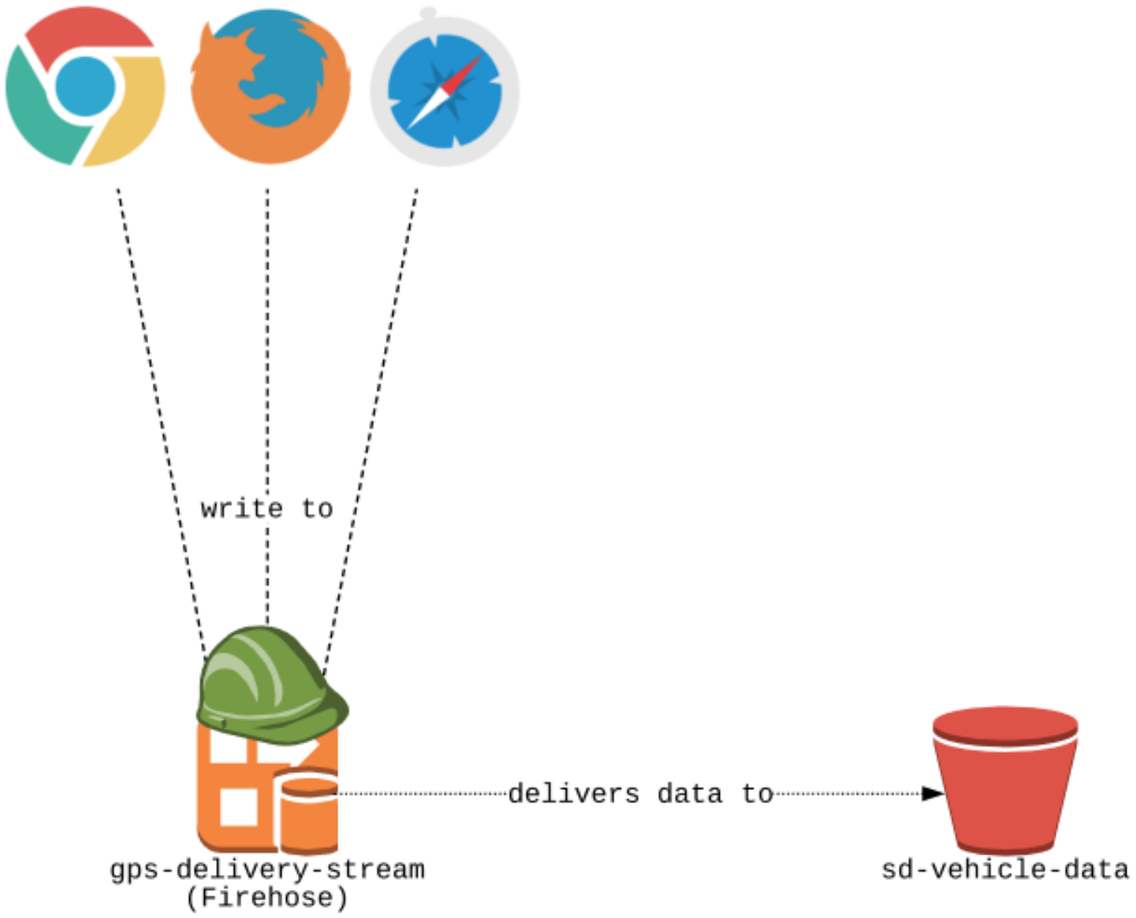
Single record

```
{
    'record_id': '939ed1d1-1740-420c-8906-445278573c7f', # <-- Unique record id
    'timestamp': '4:25:06.000', # <-- time of measurement
    'vin': '4FTEX4944AK844294', # <-- vehicle id
    'lon': 106.9447146, # <-- vehicle location longitude
    'lat': -6.3385652, # <-- vehicle location latitude
    'speed': 25 # <-- vehicle speed
}
```

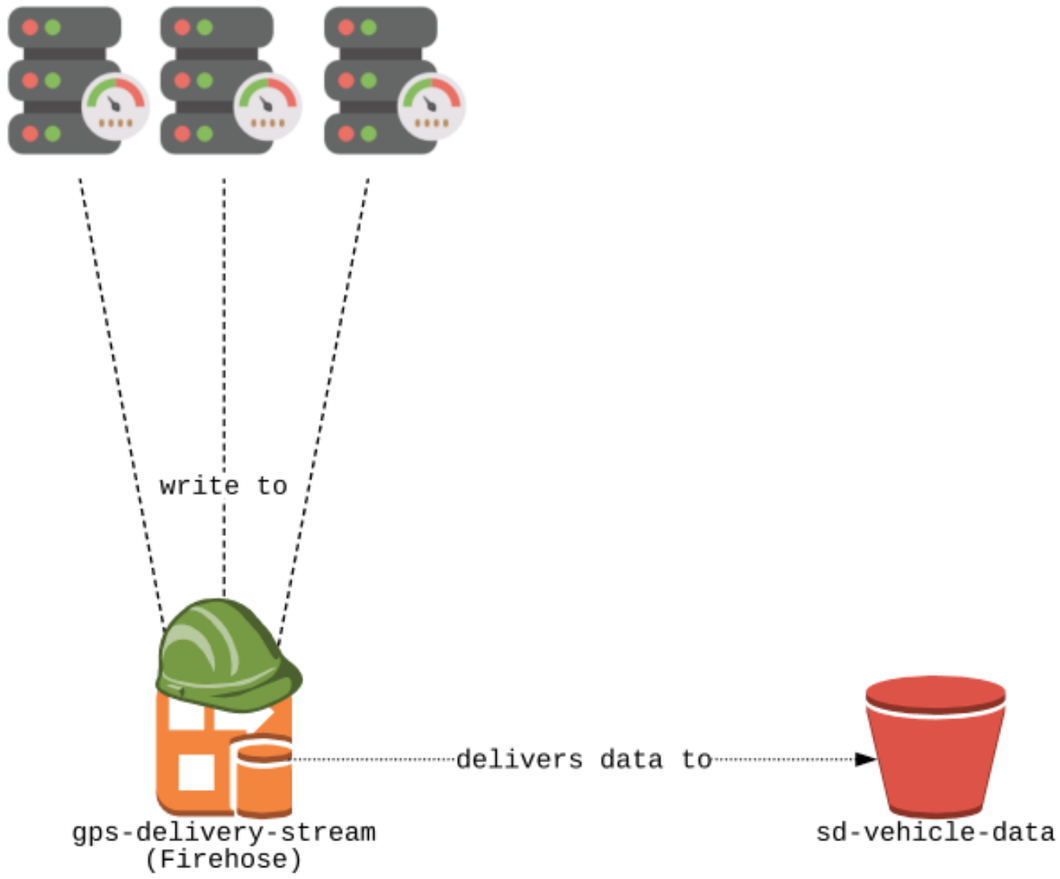
Records coming in



Another use case



Another use case



Patterns

Sending a record

```
res = firehose.put_record(
    DeliveryStreamName='gps-delivery-stream',
    Record = {
        'Data': payload
    }
)
```

Sending a record

```
Record = {
    'Data': payload
}
```

Sending a record

What our Record Looks Like

```
record = {
 'record_id': '939ed1d1-1740-420c-8906-445278573c7f',
 'timestamp': '4:25:06.000','vin': '4FTEX4944AK844294',
 'lon': 106.9447146,'lat': -6.338565200000001,
 'speed': 25}
```

What we want to send (one string)

```
"939ed1d1-1740-420c-8906-445278573c7f 4:25:06.000
4FTEX4944AK844294 106.9447146 -6.338565200000001 25"
```

Sending a record

```
payload = " ".join(
    str(value) for value in record.values()
)
```

```
print(payload)
```

```
"939ed1d1-1740-420c-8906-445278573c7f 4:25:06.000
4FTEX4944AK844294 106.9447146 -6.338565200000001 25"
```

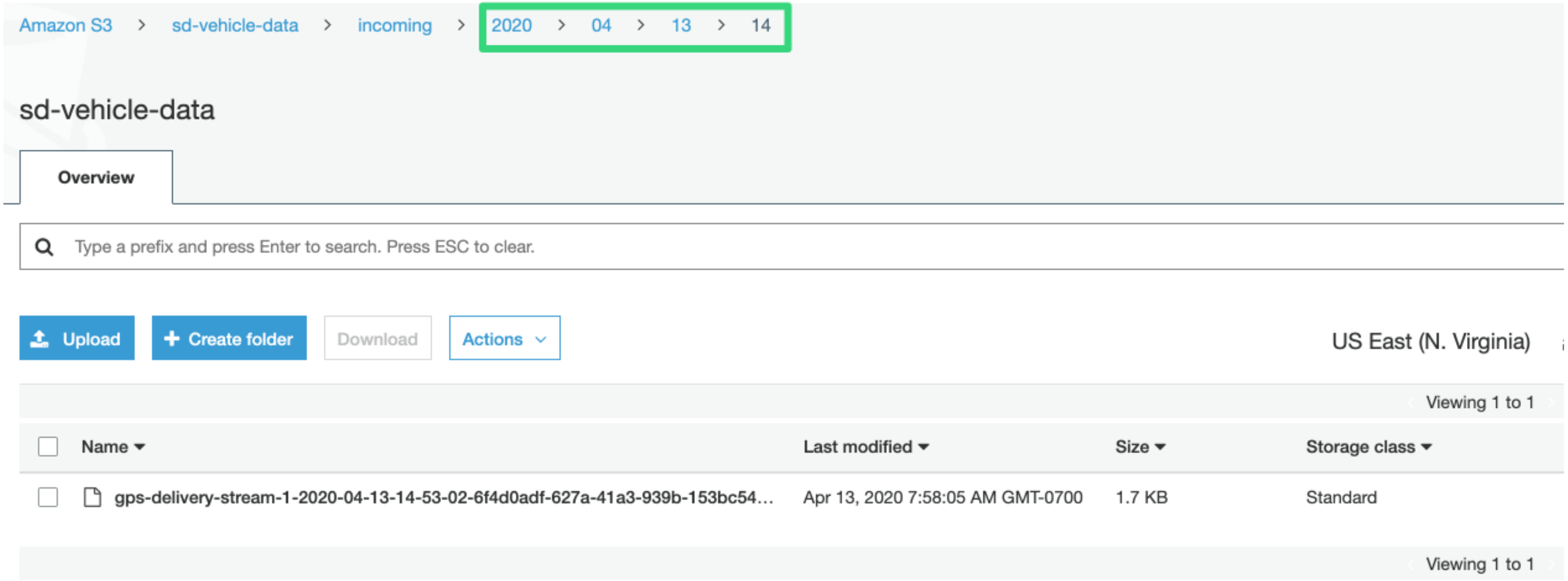
Putting it together

```
record = {
 'record_id': '939ed1d1-1740-420c-8906-445278573c7f',
 'timestamp': '4:25:06.000','vin': '4FTEX4944AK844294',
 'lon': 106.9447146,'lat': -6.338565200000001, 'speed': 25}
payload = " ".join(
    str(value) for value in record.values()
)
#"939ed1d1-1740-420c-8906-445278573c7f 4:25:06.000 4FTEX4944AK844294 106.9447146 -6.
```

Putting it together

```
res = firehose.put_record(
    DeliveryStreamName='gps-delivery-stream',
    Record = {
        'Data': payload + "\n" #<-- Line break!
    }
)
```

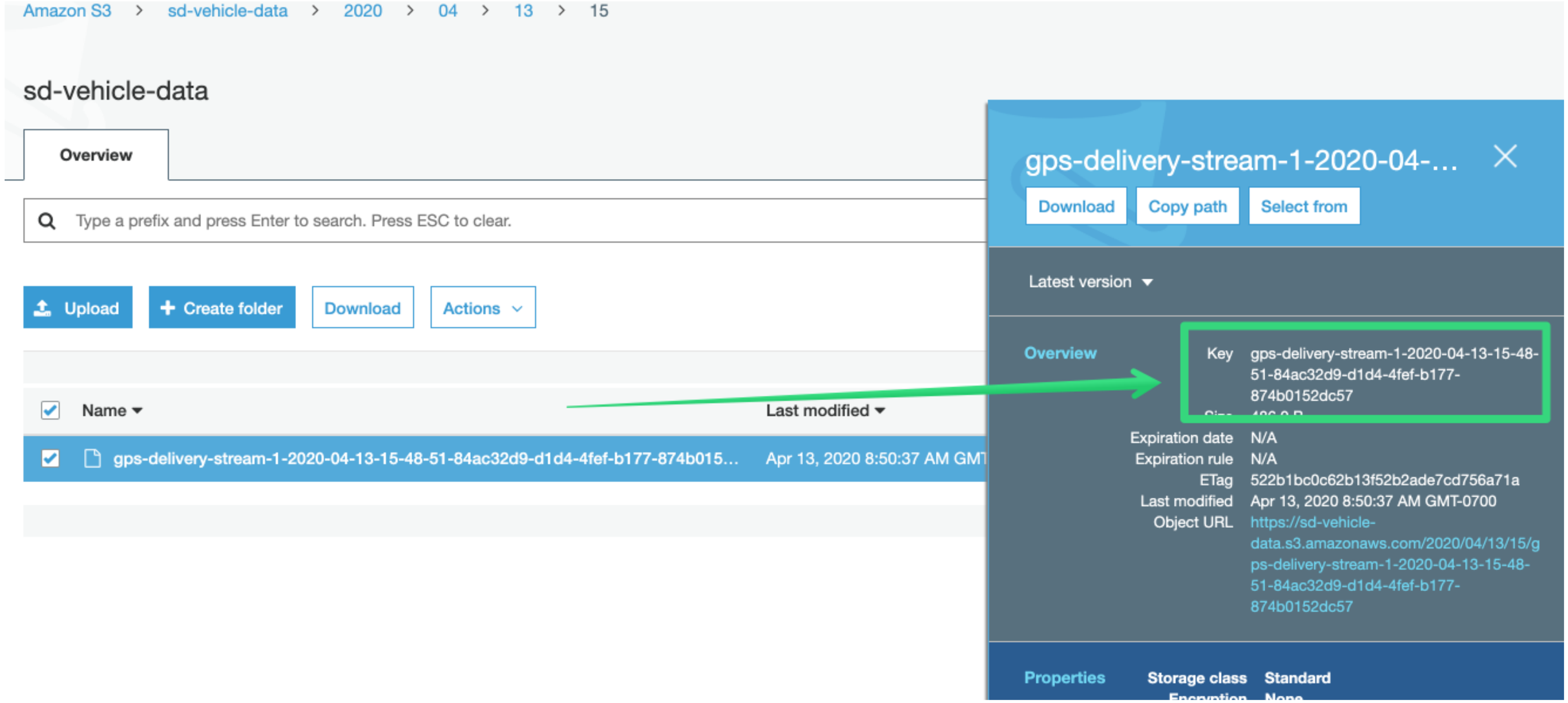
Created files



Sample data

```
939ed1d1-1740-420c-8906-445278573c7f 4:25:06.000 4FTEX4944AK844294 106.9447146 -6.338565200000001 25
f29a5b3d-d0fa-43c0-9e1a-e2a5cdb8be7a 8:10:47.000 3FTEX1G5XAK844393 108.58068100000001 34.79925 37
ff8e7131-408d-463b-8d07-d016419b0656 20:26:44.000 2LAXX1C8XAK844292 114.39239199999999 36.097577 90
bc75da5f-1bf6-444c-80ad-49c180e1b8de 23:16:06.000 3FTEX1G5XAK844393 -76.6990172 2.481207 40
7bdcf779-444e-4313-83da-140461933aeb 22:28:44.000 5FTEX1MAXAK844295 -47.0145295 -21.4649238 40
```

Created files



Create S3 client

```
# Create boto3 S3 client.
s3 = boto3.client('s3',
                  aws_access_key_id=AWS_KEY_ID,
                  aws_secret_access_key=AWS_SECRET,
                  region_name='us-east-1')
```

Read data into DataFrame

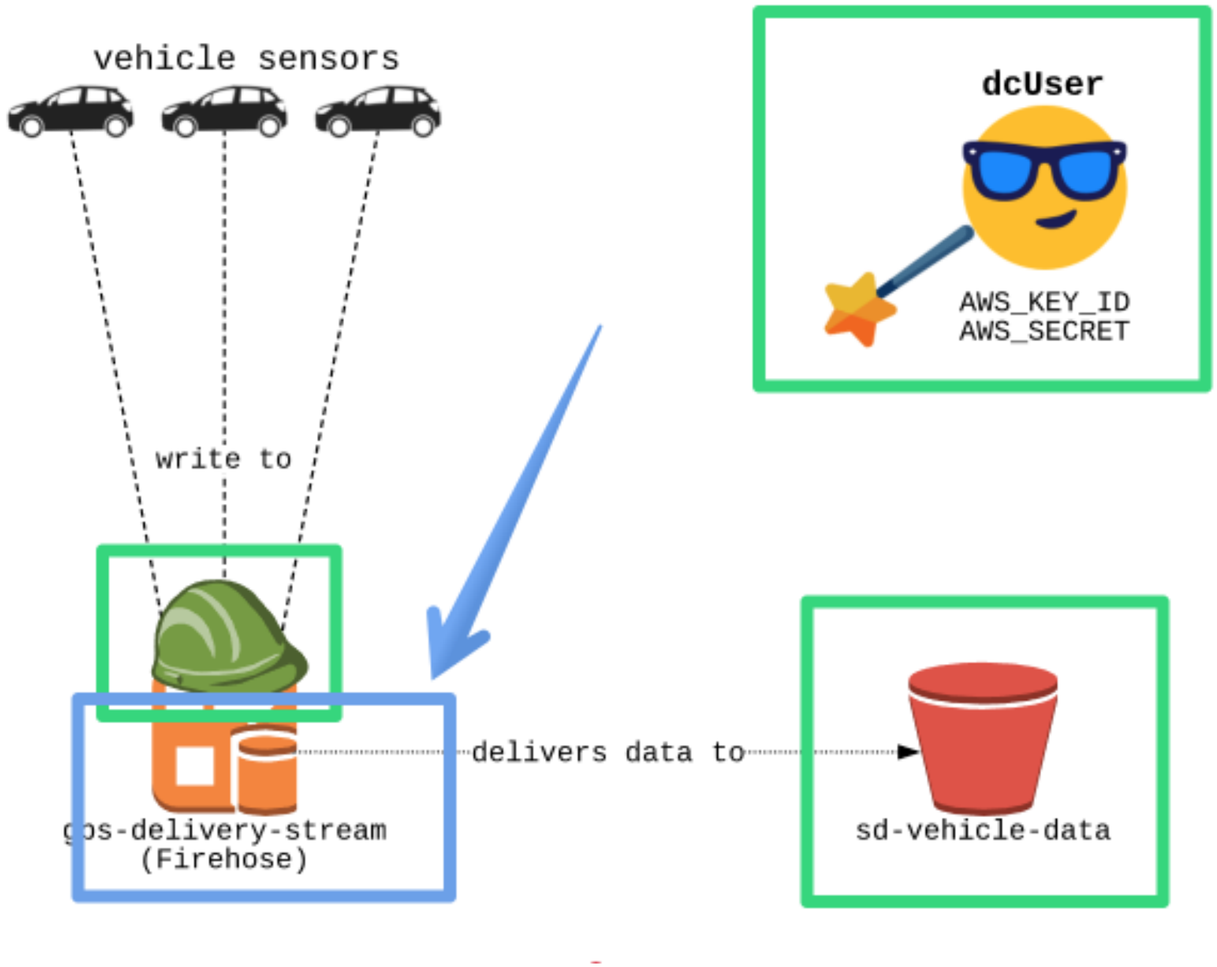
```
# Get the object from S3
obj_data = s3.get_object(Bucket='sd-vehicle-data', Key=KEY_YOU_COPIED)
```

```
# Read read the object into a DataFrame
vehicle_data = pd.read_csv(
    data['Body'],
    delimiter = " ",
    names=["record_id", "timestamp", "vin", "lon", "lat", "speed"]))
```

vehicle_data

	record_id	timestamp	vin	lon	lat	speed
0	939ed1d1...	4:25:06.000	4FTEX4944AK844294	106.945	-6.33857	25
1	f29a5b3d...	8:10:47.000	3FTEX1G5XAK844393	108.581	34.7993	37
2	ff8e7131...	20:26:44.000	2LAXX1C8XAK844292	114.392	36.0976	90
3	bc75da5f...	23:16:06.000	3FTEX1G5XAK844393	-76.699	2.48121	40
4	7bdcf779...	22:28:44.000	5FTEX1MAXAK844295	-47.0145	-21.4649	40

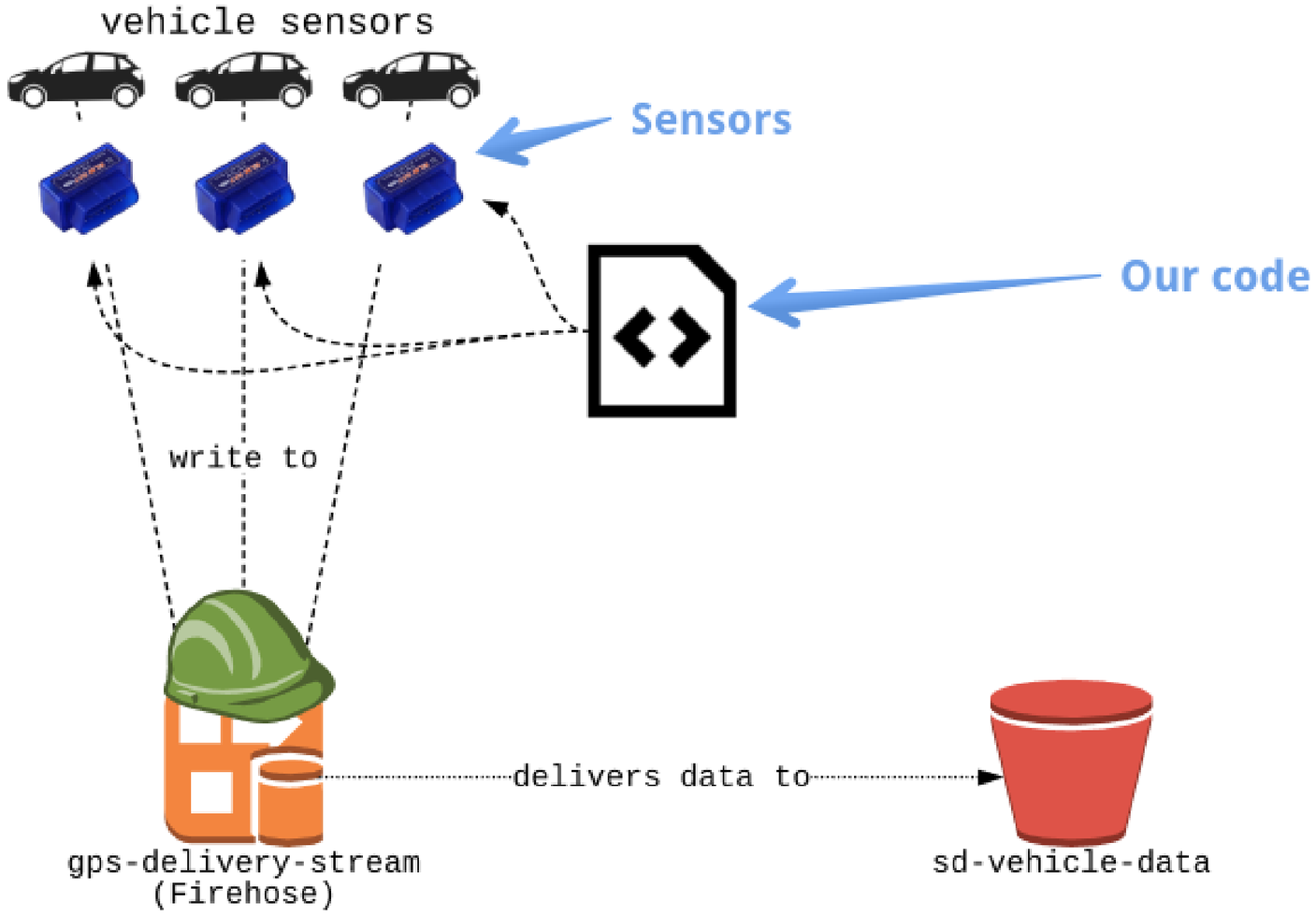
Review


vehicle sensors
dcUser
AWS_KEY_ID
AWS_SECRET
write to
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

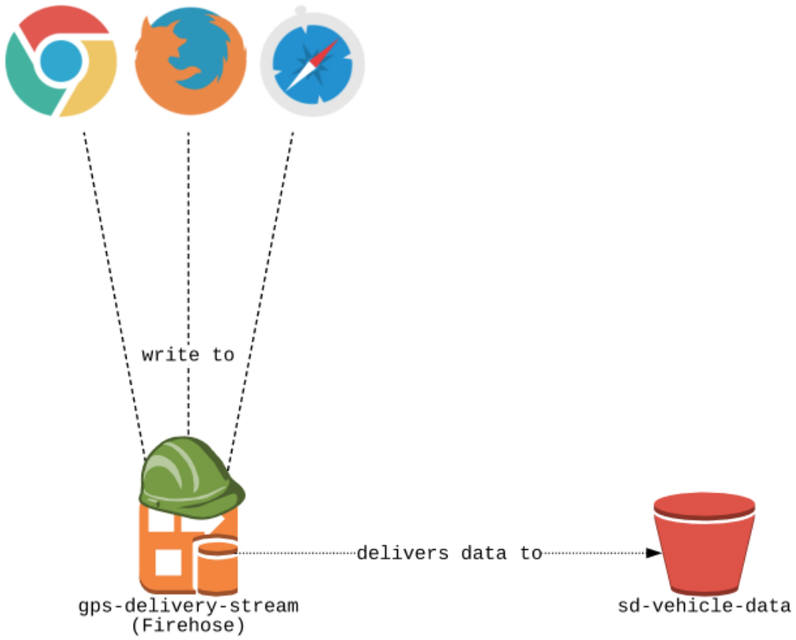
Review

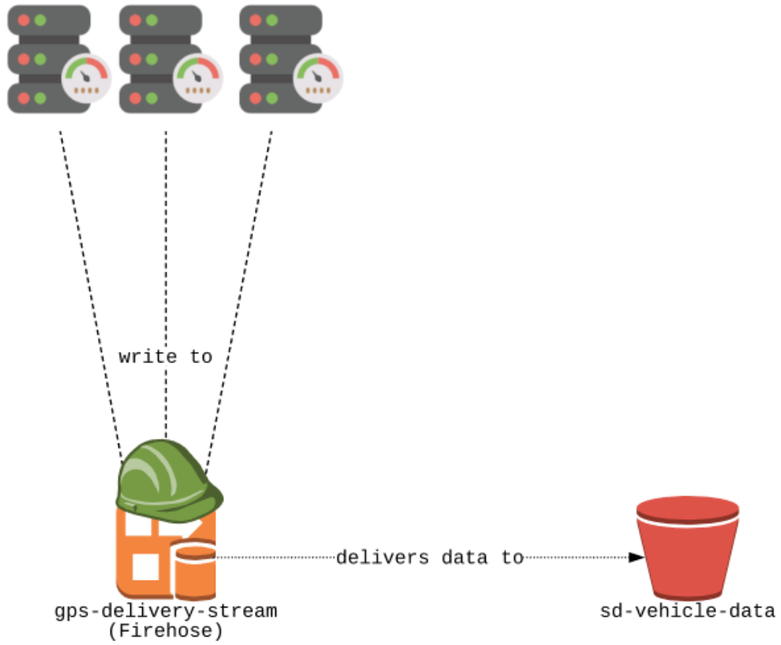
```
res = firehose.create_delivery_stream(
  DeliveryStreamName =  "gps-delivery-stream",
  DeliveryStreamType = "DirectPut",
  S3DestinationConfiguration = {
    "RoleARN": "arn:aws:iam::0000000:role/firehoseDeliveryRole",
    "BucketARN": "arn:aws:s3:::sd-vehicle-data",
  }
)
```

Review


vehicle sensors
Sensors
Our code
write to
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

Review


write to
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data


write to
delivers data to
gps-delivery-stream
(Firehose)
sd-vehicle-data

Review

	record_id	timestamp	vin	lon	lat	speed
0	939ed1d1...	4:25:06.000	4FTEX4944AK844294	106.945	-6.33857	25
1	f29a5b3d...	8:10:47.000	3FTEX1G5XAK844393	108.581	34.7993	37
2	ff8e7131...	20:26:44.000	2LAXX1C8XAK844292	114.392	36.0976	90
3	bc75da5f...	23:16:06.000	3FTEX1G5XAK844393	-76.699	2.48121	40
4	7bdcf779...	22:28:44.000	5FTEX1MAXAK844295	-47.0145	-21.4649	40

Let's practice!

STREAMING DATA WITH AWS KINESIS AND LAMBDA