Streaming data case study

STREAMING DATA WITH AWS KINESIS AND LAMBDA

Maksim Pecherskiy
Data Engineer

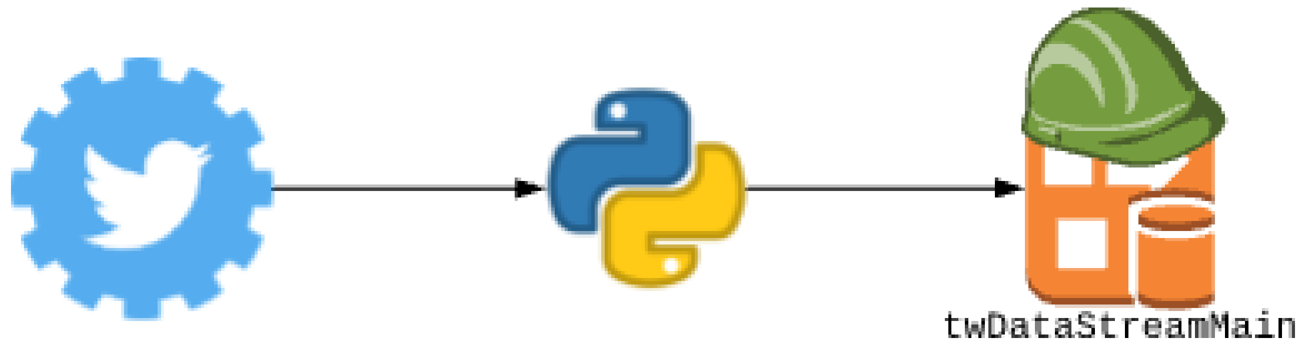
This chapter

- Send incoming data to Firehose
- Store data
- Visualize data
- Set alerts in real-time
- Monitor the stream
- Meet a set of requirements

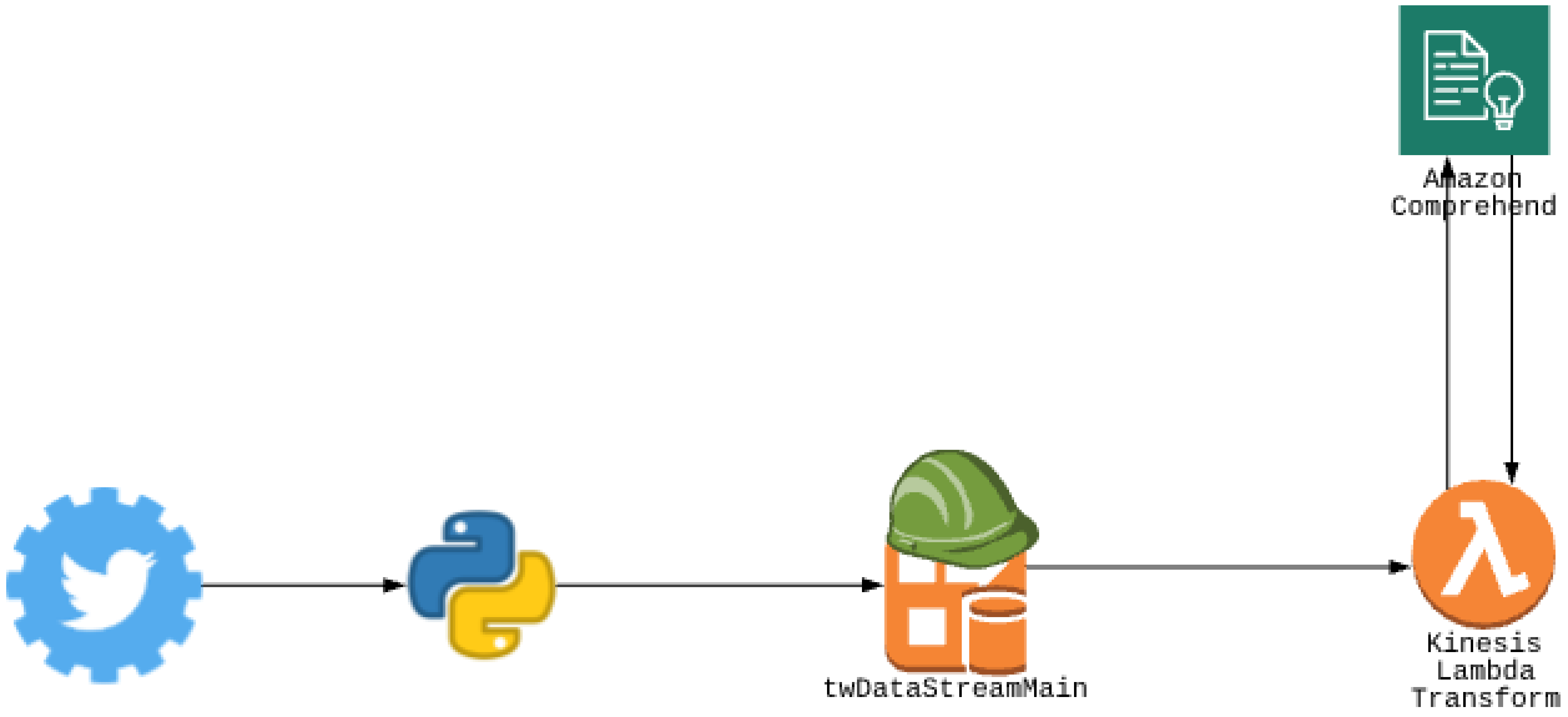
Requirements

- Tweets must include #sandiego hashtag
- Tweets must come in real time
- Tweets must come enriched with sentiment
- Visualize last 15 minutes of data
- Notify manager if >3 negative tweets in 5 minute interval
- The stream should minimize data loss due to downtime
- Data must persist to be analyzed later

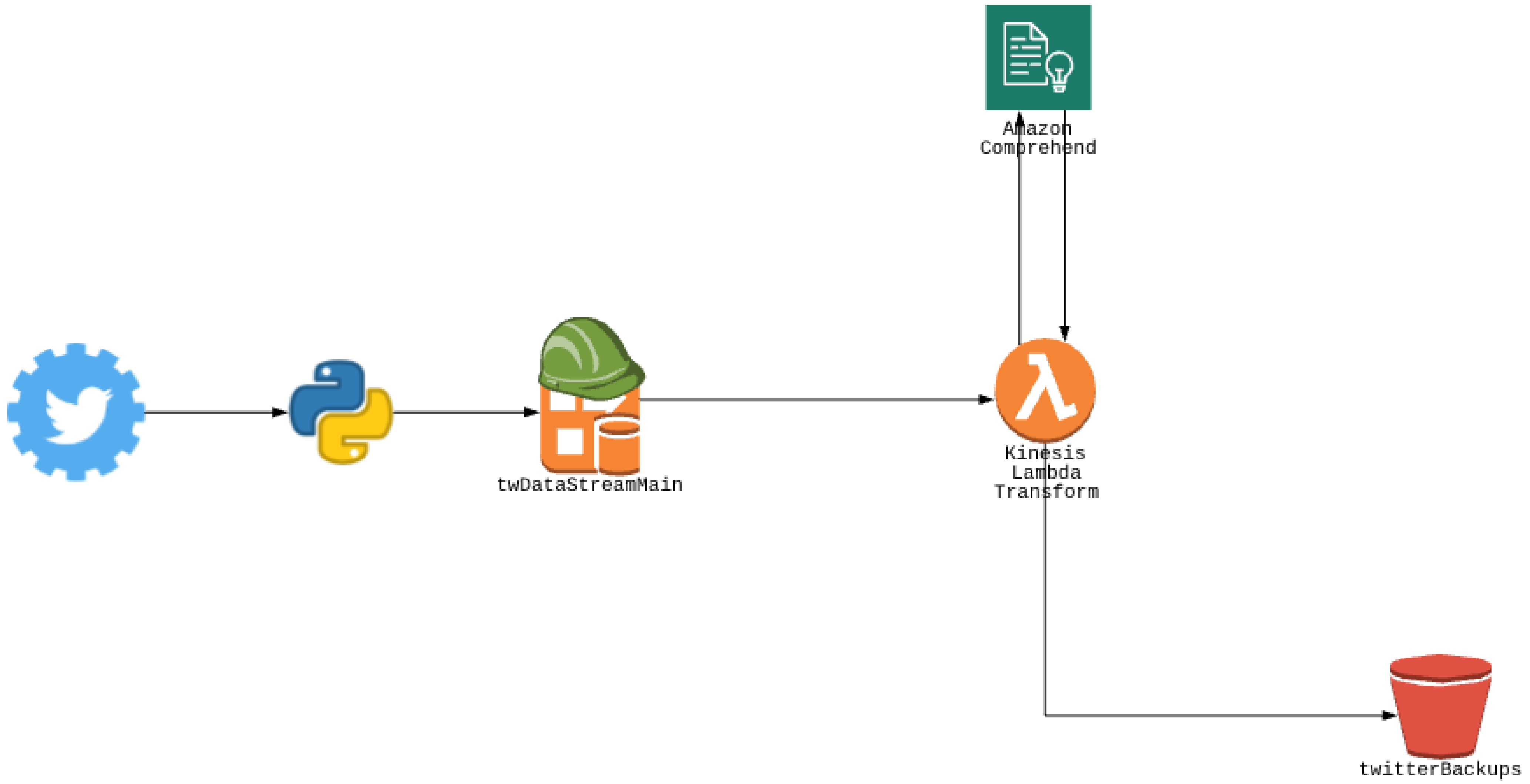
Tweets come in real-time



Enriched with sentiment



Data must persist for later analysis



Visualize last 15 minutes

Amazon Redshift

Amazon Elasticsearch Service

Redshift vs Elasticsearch

Redshift

- Designed for storing clean tables of data
- Schema is defined in database
- SQL for queries
- Works great with BI tools like Tableau

Elasticsearch

- Schemaless - good for logs and text
- Schema is created during query
- Uses its own language for queries
- Has its own UI - Kibana

Let's practice!

STREAMING DATA WITH AWS KINESIS AND LAMBDA

Creating an Elasticsearch cluster

STREAMING DATA WITH AWS KINESIS AND LAMBDA

Maksim Pecherskiy
Data Engineer

Let's practice!

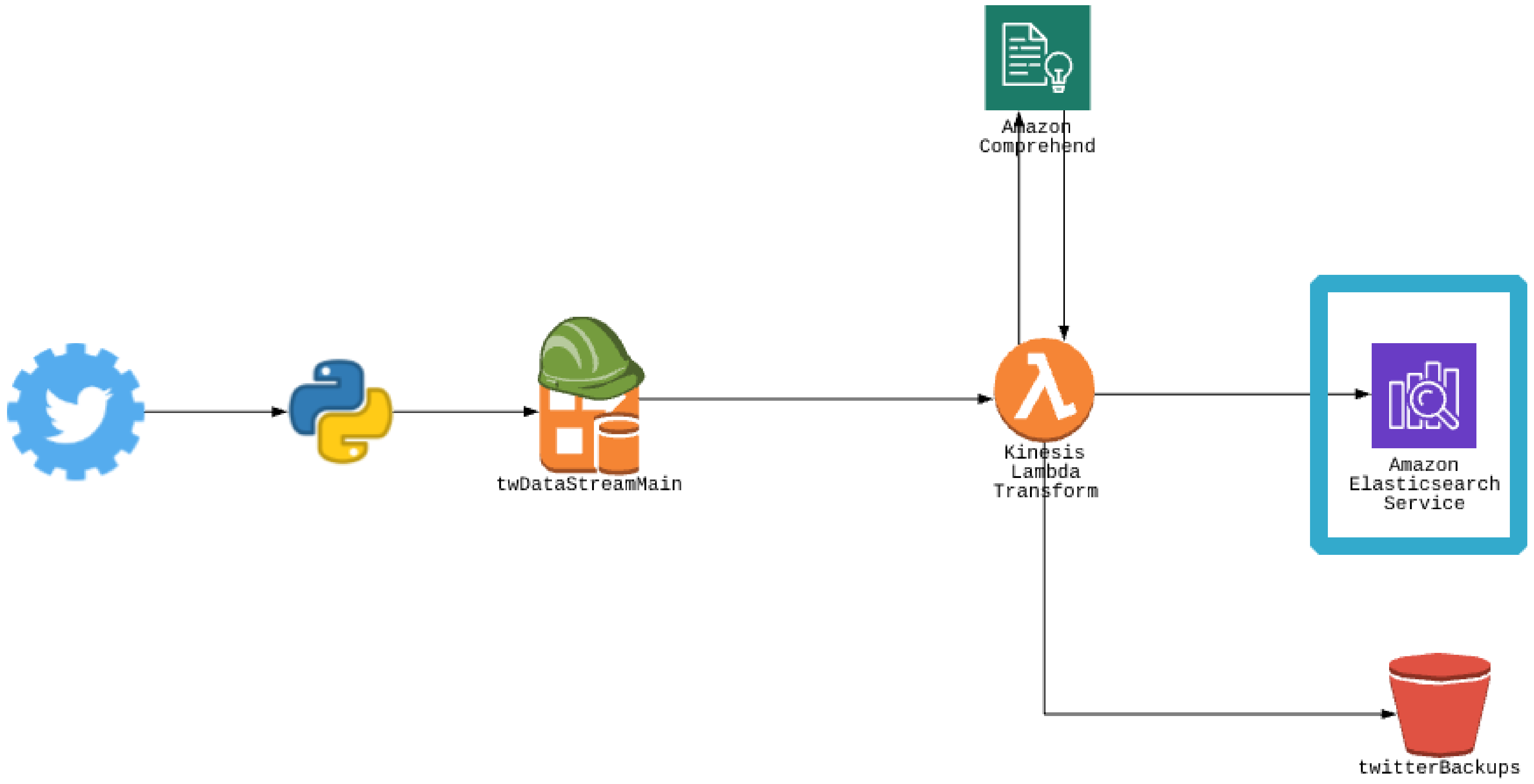
STREAMING DATA WITH AWS KINESIS AND LAMBDA

Monitoring performance

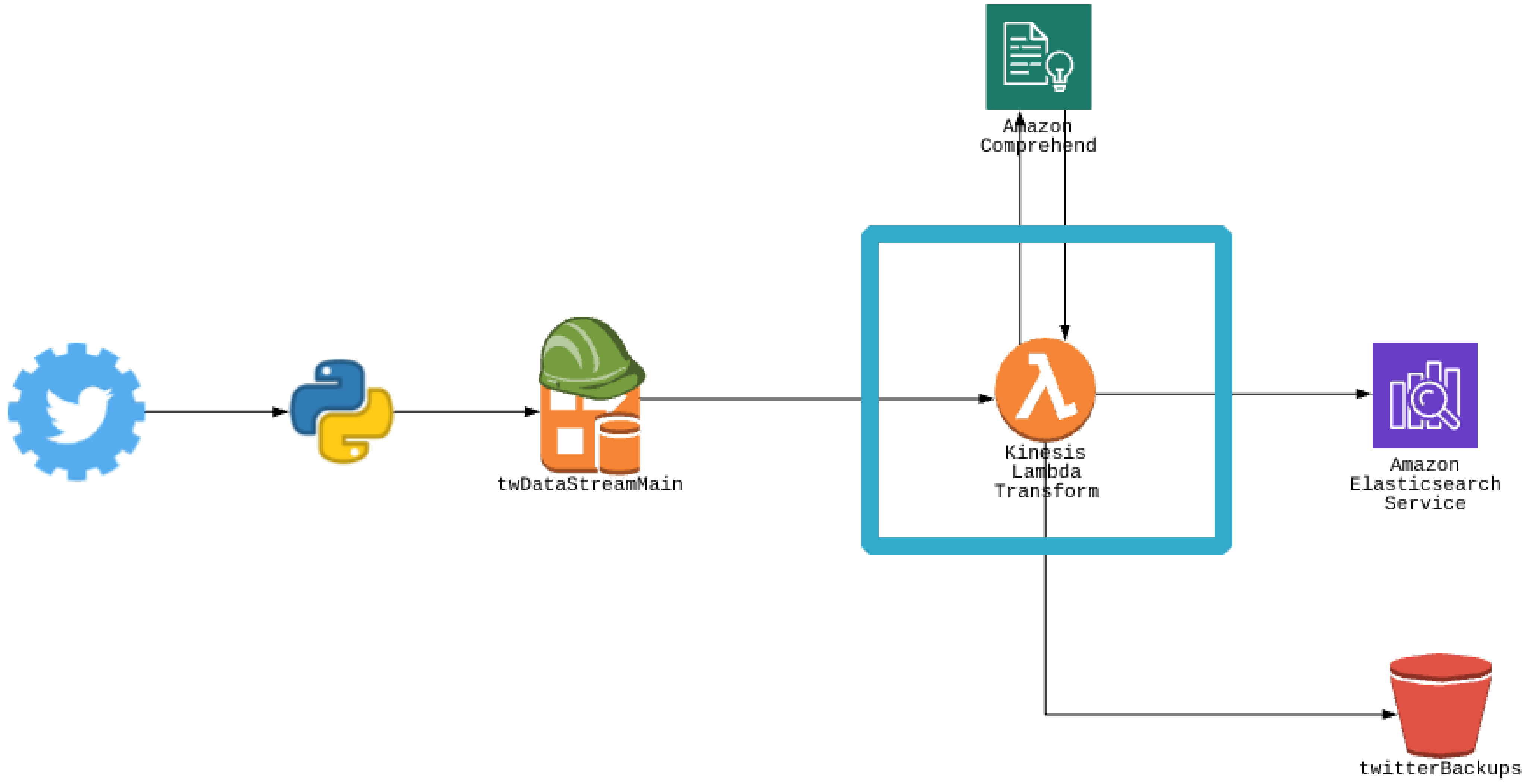
STREAMING DATA WITH AWS KINESIS AND LAMBDA

Maksim Pecherskiy
Data Engineer

Last lesson


Amazon Comprehend
twDataStreamMain
Kinesis Lambda Transform
Amazon Elasticsearch Service
twitterBackups

Lambda transform



Lambda transform

```
def lambda_handler(event, context):
    comprehend = boto3.client('comprehend',
                              region_name='us-east-1',
                              aws_access_key_id = AWS_KEY,
                              aws_secret_access_key=AWS_SECRET)
    output = []
    for record in event['records']:
        ...
```

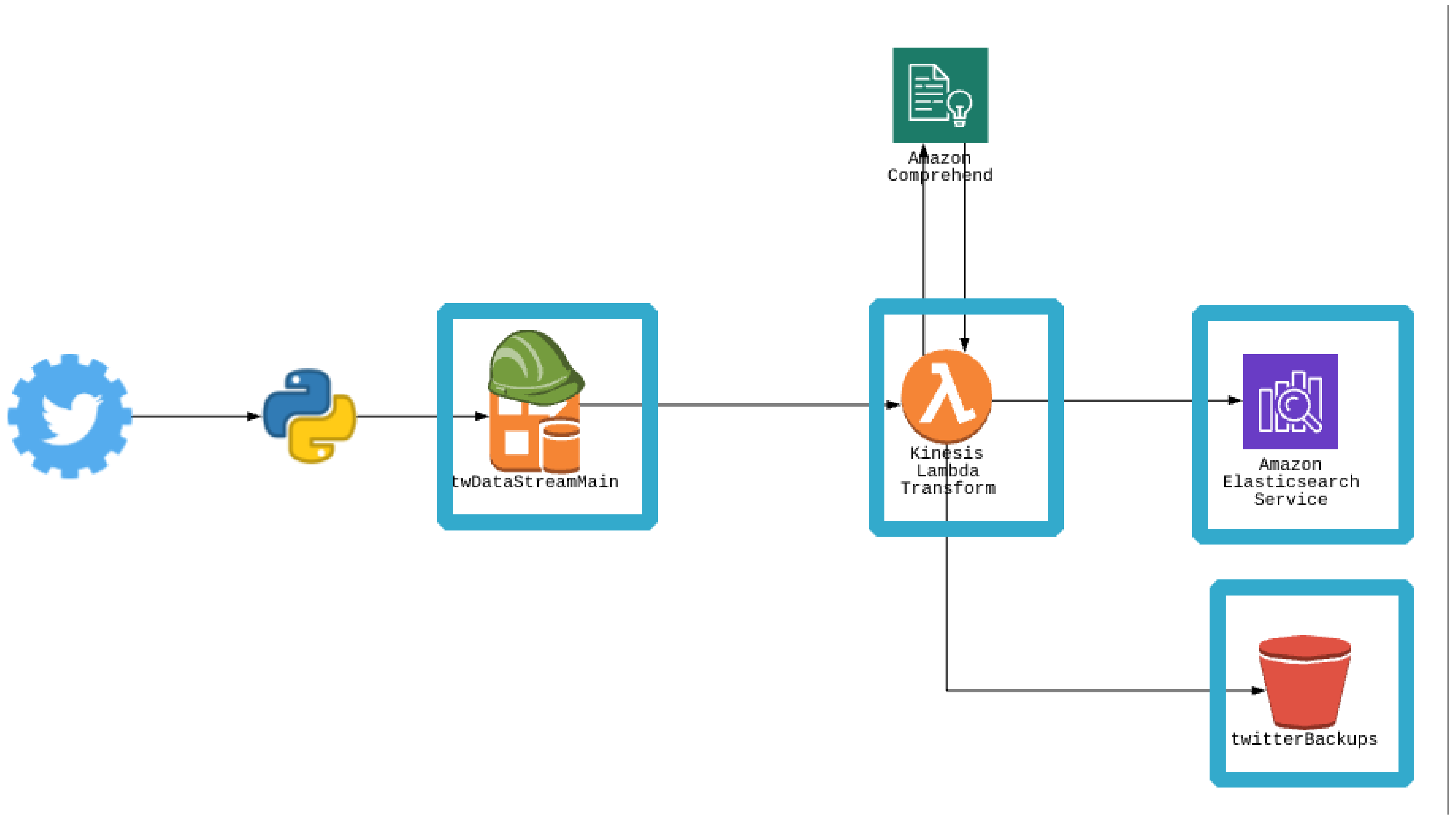
Lambda transform

```python
def lambda_handler(event, context):
    ...
    for record in event['records']:
        dict_data = base64.b64decode(record['data']).decode('utf-8').strip()
        dict_data = json.loads(dict_data)
        sentiment_all = comprehend.detect_sentiment(
          Text=dict_data['text'],
          LanguageCode=dict_data['lang'])
        dict_data['sentiment'] = sentiment_all['Sentiment']
        ...
```

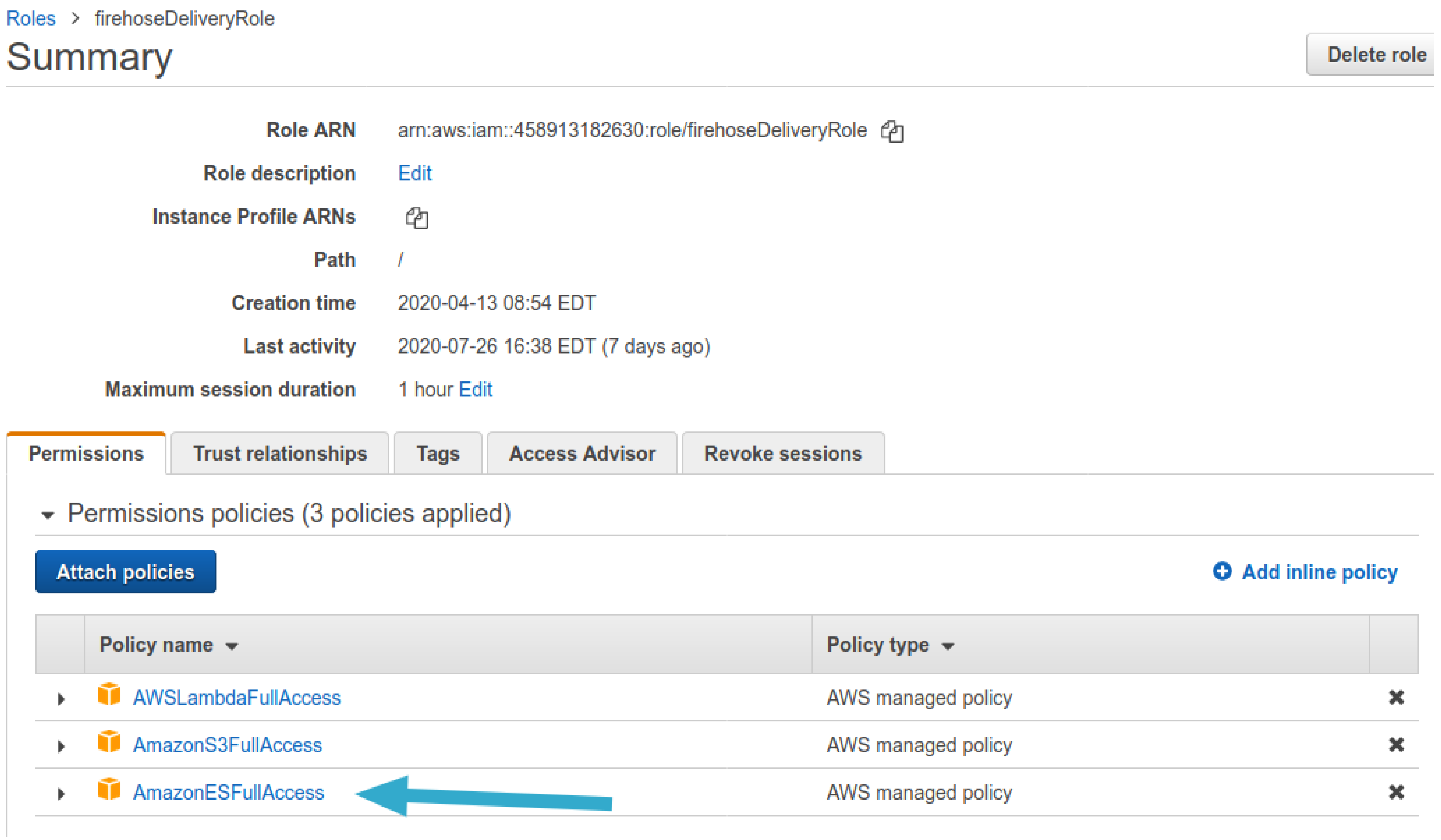
Lambda transform

```
def lambda_handler(event, context):
    ...
    for record in event['records']:
        ...
        output_record = {
            'recordId': record['recordId'],
            'result': 'Ok',
            'data': base64.b64encode(json.dumps(dict_data).encode('utf-8'))
        }
        output.append(output_record)
    return {'records': output}
```

Wiring it up


Amazon Comprehend
twDataStreamMain
Kinesis Lambda Transform
Amazon Elasticsearch Service
twitterBackups

Update firehoseDeliveryRole



Roles > firehoseDeliveryRole

Summary

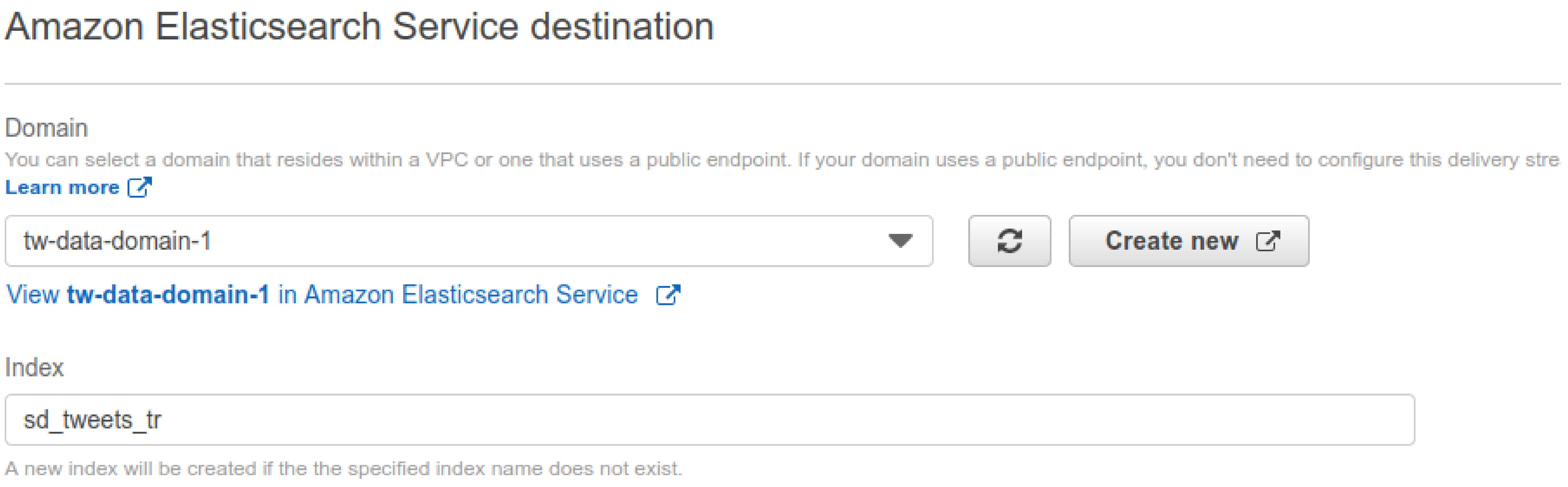
Delete role

Role ARN	arn:aws:iam::458913182630:role/firehoseDeliveryRole
Role description	Edit
Instance Profile ARNs	
Path	/
Creation time	2020-04-13 08:54 EDT
Last activity	2020-07-26 16:38 EDT (7 days ago)
Maximum session duration	1 hour Edit

Permissions policies (3 policies applied)

Policy name	Policy type
AWSLambdaFullAccess	AWS managed policy
AmazonS3FullAccess	AWS managed policy
AmazonESFullAccess	AWS managed policy

Create delivery stream



Amazon Elasticsearch Service destination

Domain

You can select a domain that resides within a VPC or one that uses a public endpoint. If your domain uses a public endpoint, you don't need to configure this delivery stre

Learn more

tw-data-domain-1

Create new

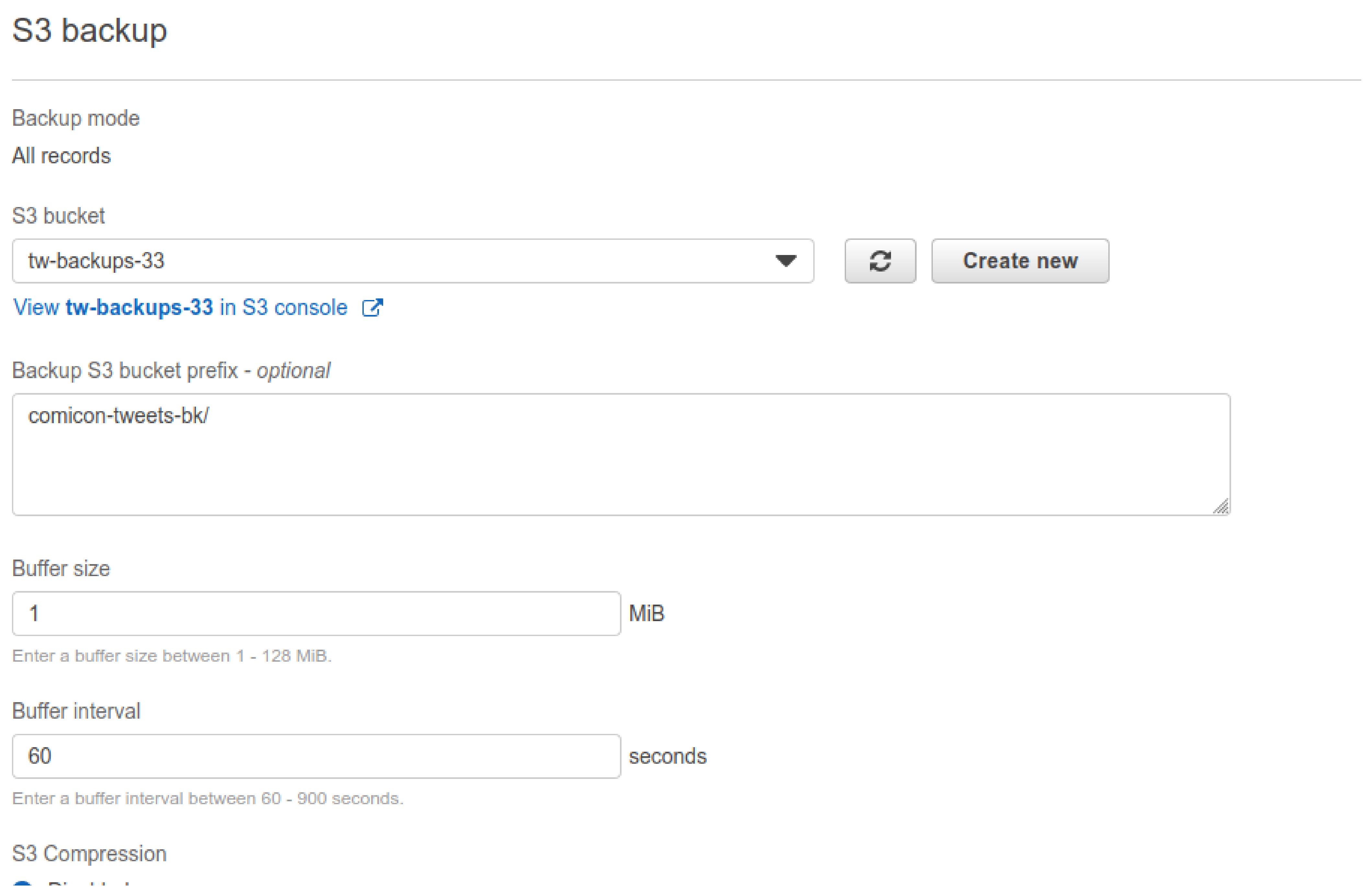
View **tw-data-domain-1** in Amazon Elasticsearch Service

Index

sd_tweets_tr

A new index will be created if the the specified index name does not exist.

Create delivery stream



S3 backup

Backup mode

All records

S3 bucket

tw-backups-33

View **tw-backups-33** in S3 console

Backup S3 bucket prefix - *optional*

comicon-tweets-bk/

Buffer size

1 MiB

Enter a buffer size between 1 - 128 MiB.

Buffer interval

60 seconds

Enter a buffer interval between 60 - 900 seconds.

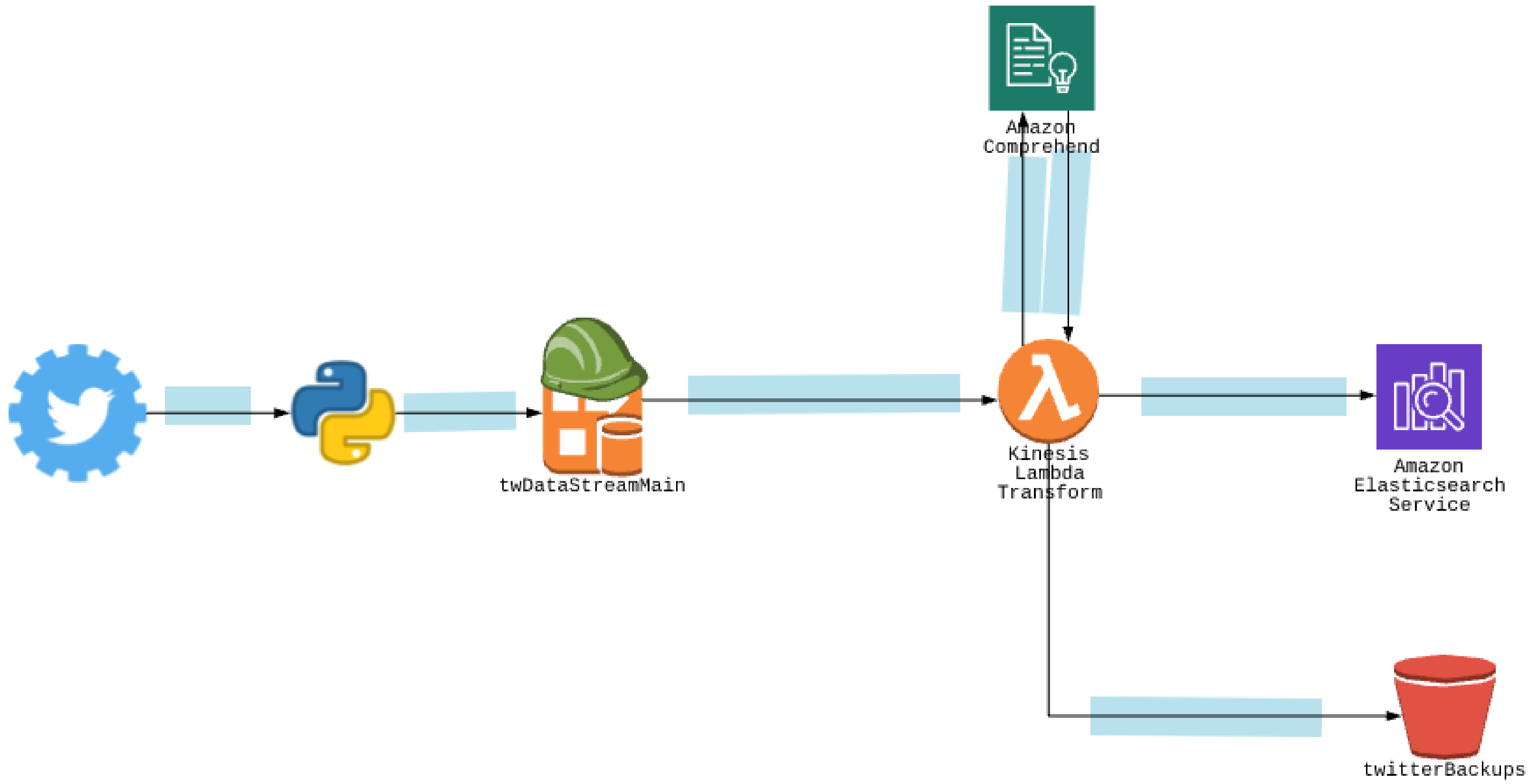
S3 Compression

Cloudwatch

> The stream should minimize data loss due to downtime.

- Logs (raw data)
- Metrics (measures of various activities of the service)
- Alarms (notifications when a metric is out of a specified range)
- Dashboards (metrics visualization)

Failure points



Let's practice!

STREAMING DATA WITH AWS KINESIS AND LAMBDA

Cloudwatch dasbhoards and alarms

STREAMING DATA WITH AWS KINESIS AND LAMBDA

Maksim Pecherskiy
Data Engineer

Let's practice!

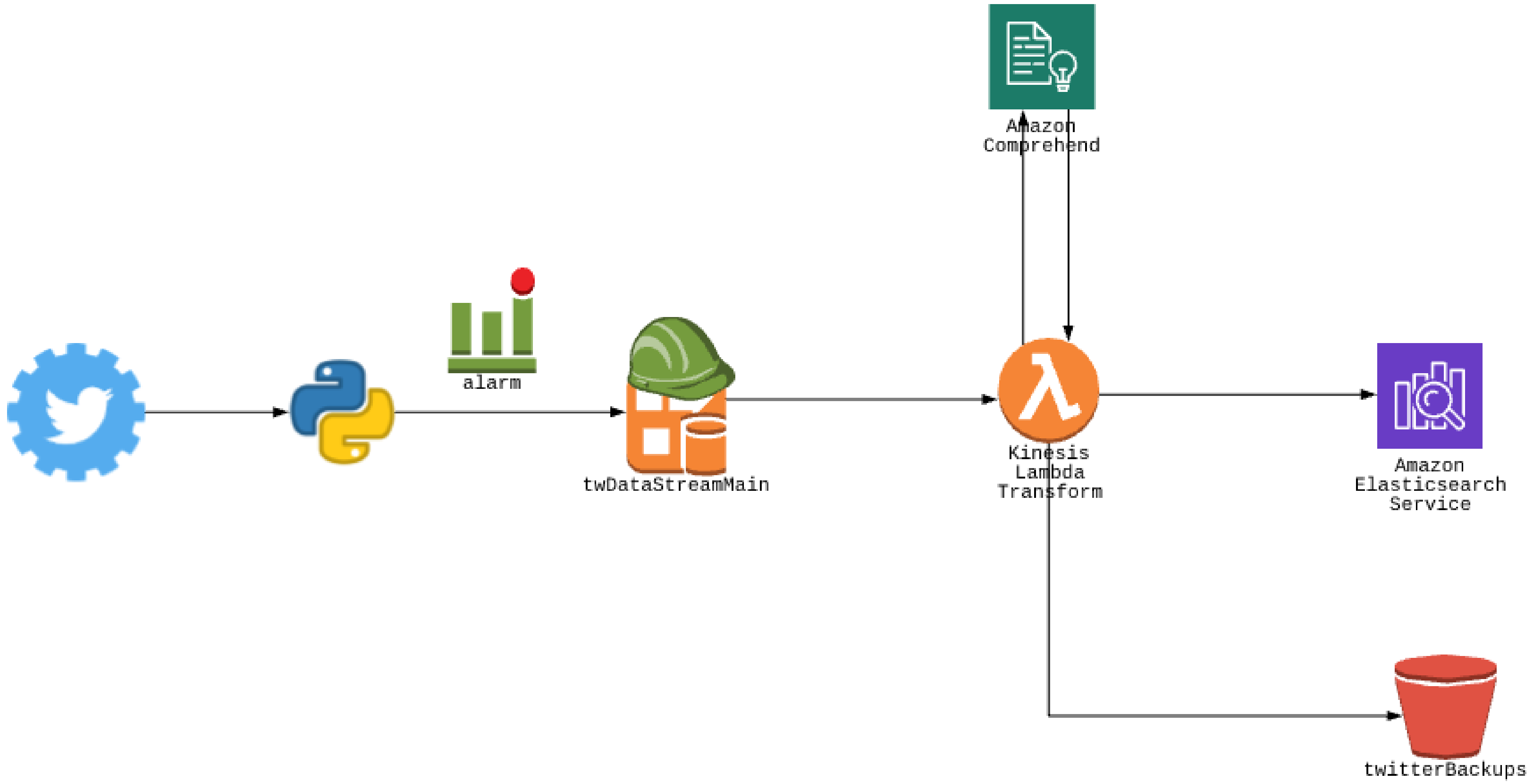
STREAMING DATA WITH AWS KINESIS AND LAMBDA

Visualizing streaming data

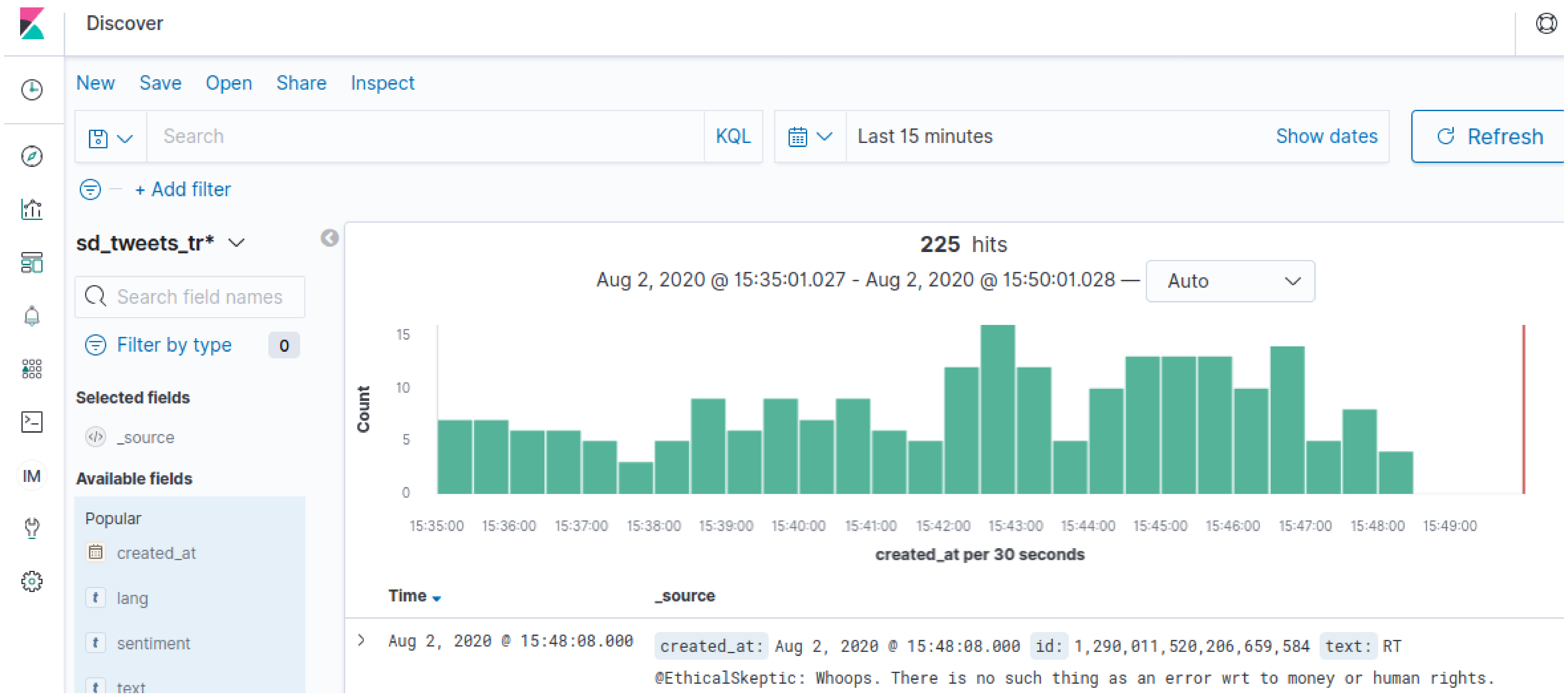
STREAMING DATA WITH AWS KINESIS AND LAMBDA

Maksim Pecherskiy
Data Engineer

Our pipeline so far

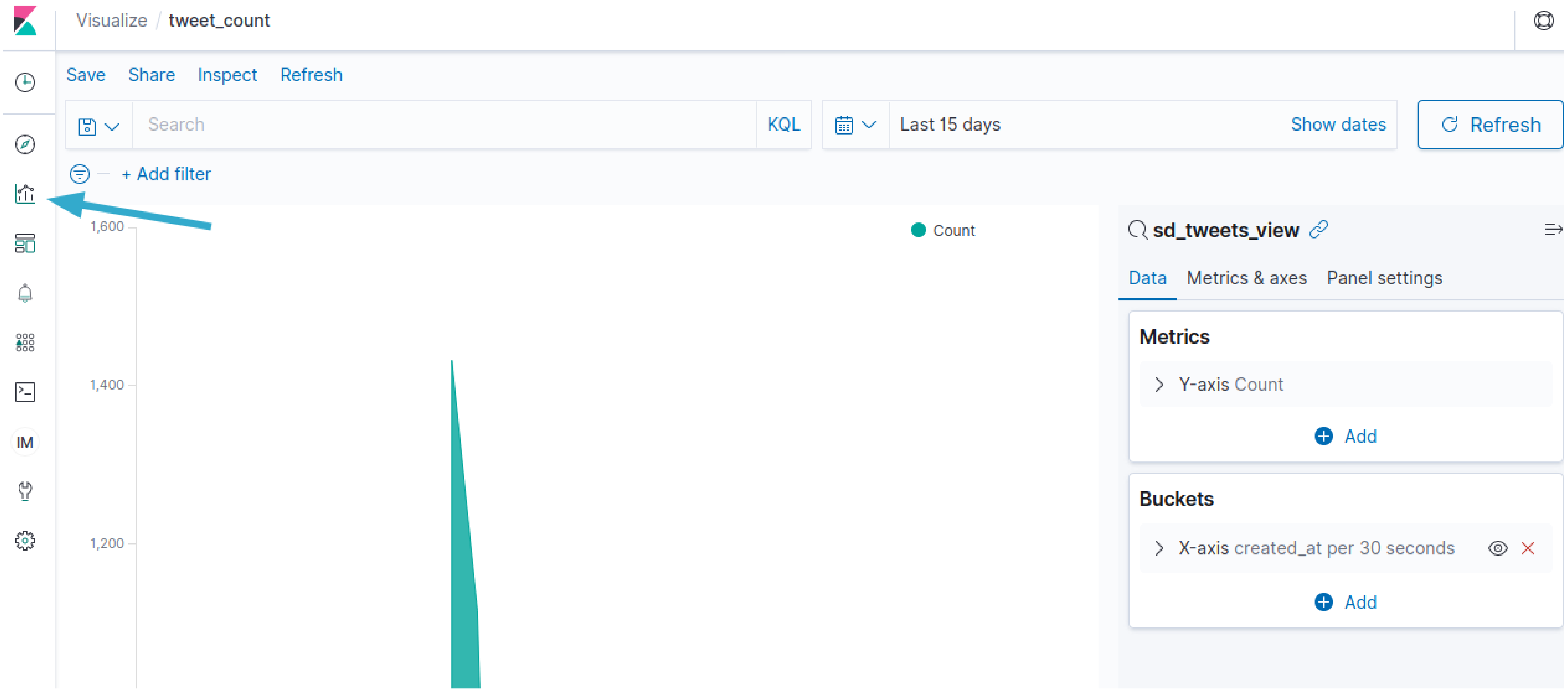


Kibana Discover view



```
Aug 2, 2020 @ 15:48:08.000    created_at: Aug 2, 2020 @ 15:48:08.000 id: 1,290,011,520,206,659,584 text: RT
                              @EthicalSkeptic: Whoops. There is no such thing as an error wrt to money or human rights.
```

Kibana visualizations



sd_tweets_view

Data | Metrics & axes | Panel settings

Metrics

Y-axis Count

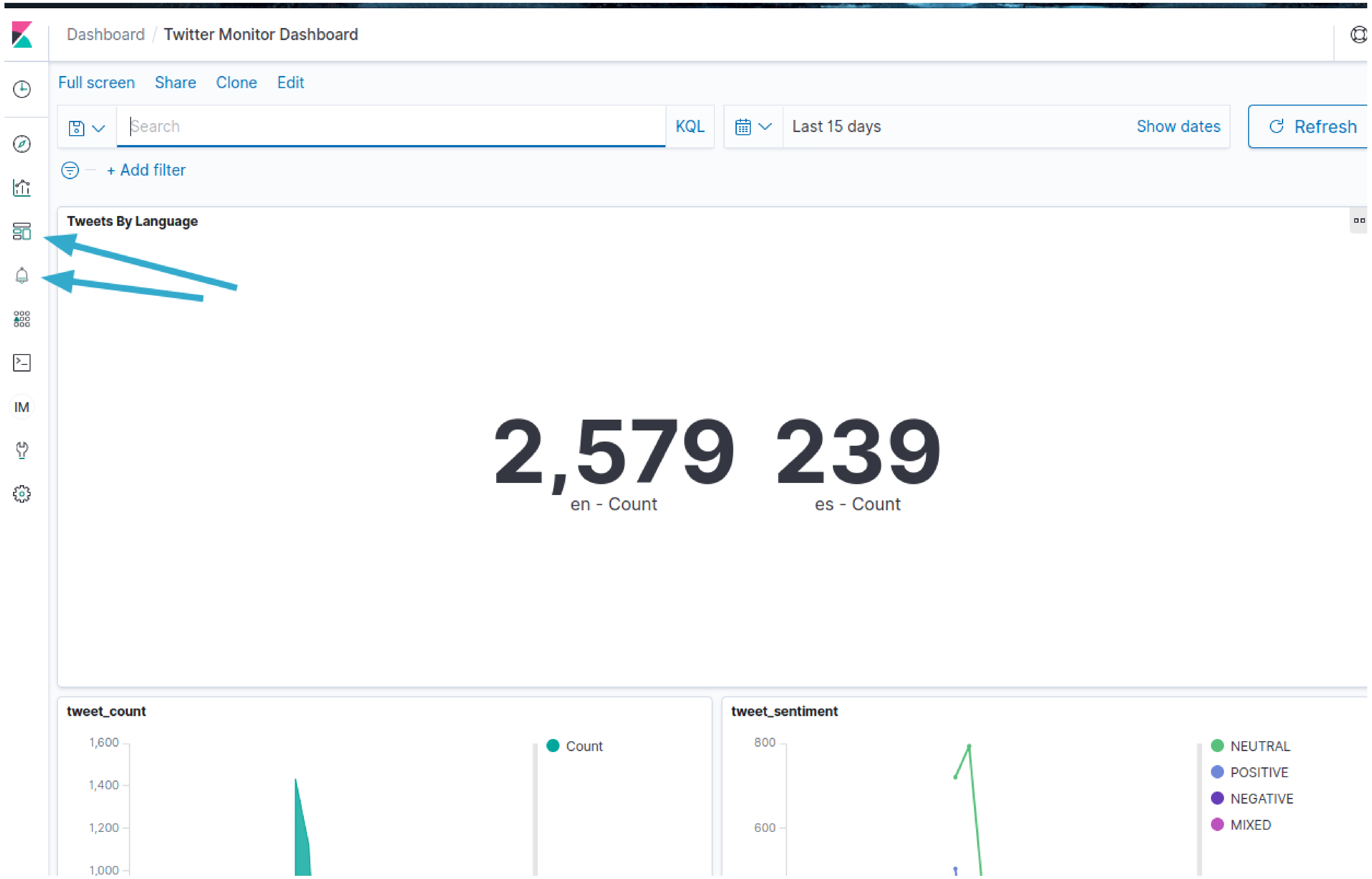
Add

Buckets

X-axis created_at per 30 seconds

Add

Dashboards and alerts



Elasticsearch vs CloudWatch

CloudWatch

- AWS centric
- Can accept custom data, but not primary use case
- Great for working with logs

Elasticsearch

- General, open source tool
- Accepts custom data, including logs
- Better viz than CloudWatch
- Robust plugin ecosystem

Let's practice!

STREAMING DATA WITH AWS KINESIS AND LAMBDA

Working with ElasticSearch using Kibana

STREAMING DATA WITH AWS KINESIS AND LAMBDA

Maksim Pecherskiy
Data Engineer

Let's practice!

STREAMING DATA WITH AWS KINESIS AND LAMBDA