Problem

> You are the lead architect for a new connected train. The CEO of your company is convinced that your trains should collect as much data as possible so that it can be used for predictive maintenance and to improve safety

1. How would you **collect the data** that the trains are generating?
2. Your CEO has asked you to design a **low-cost** system that will **"let the train know"** whether it finds any **anomalies in the data** it collects. How would this best be accomplished?
3. How would you **share the collected sensor data** with your team of train engineers and data scientists?

The answer could be… :

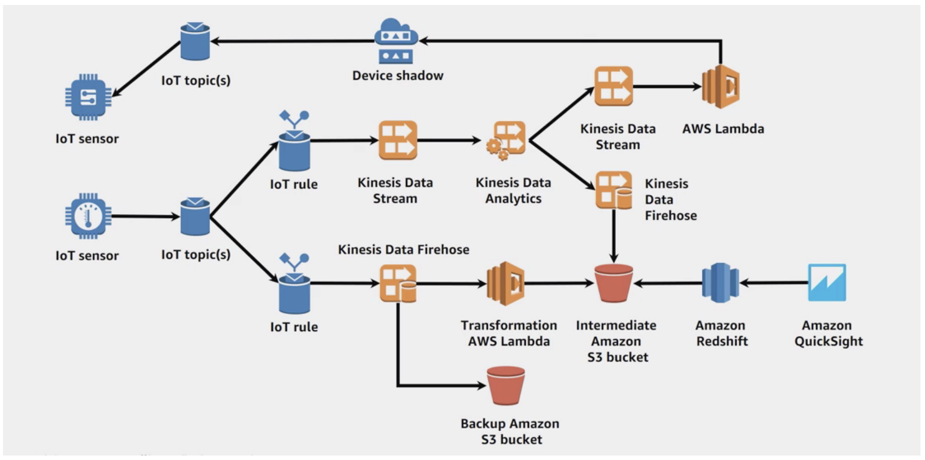

Figure 1: image

Collecting Data with IoT core

IoT Sensor -> IoT topic -> IoT rule -> Kinesis Data Stream

Train will occasionally lose internet connection. So we need some kind of register (could be achieved by IoT device shadow)

Your device would rely on slow, unreliable internet connection.

IoT core uses MQTT message protocol, which is lightweight so it could function over limited band width, and it can automatically synchronize from the dead connections.

Components of IoT core includes:

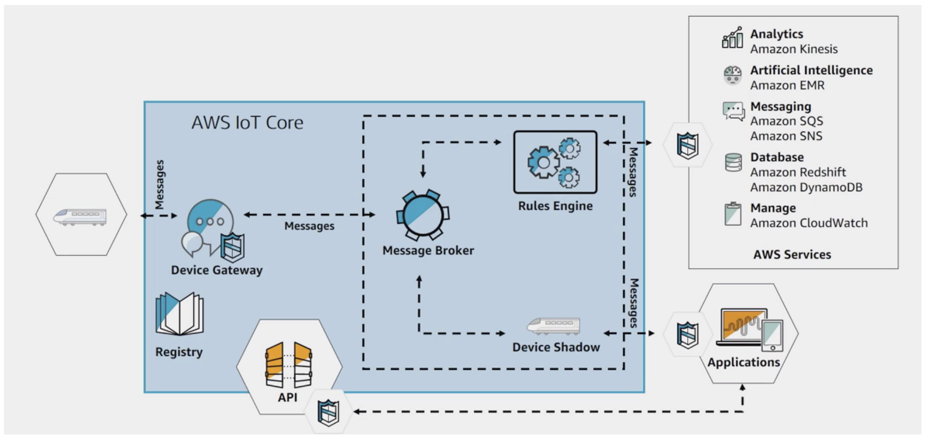

Figure 2: image

- Device Gateway : Manages connection between remote device and AWS IoT core.
- Registry: establishes identity for devices and tracks metadata such as attributes of devices (what is the temperature, health check…)
- Message Broker : manages any messages that pass through the device gate way. Enables sending / receiving messages to/from IoT.
- Rules Engine : Enables building the application that gather, process, and analyze the data generated from the device in **global scale** without having to manage any infrastructure.

You can route them to many other AWS services like SNS topic, SQS, kinesis data stream, Lambda function

- Device Shadow : Your application can interact with device shadow. Persistent, virtual version of the device. It enables you to synchronize the action of the device after it reconnects to the internet.

IoT encryption

It encrypts the data in transit using TLS with client authentication. The server authenticates with the client with the X.509 certificate. The copy of the servers' root certificate is uploaded to the device in order to check the validity of the server's certificate. Client authentication is additional layer of security. In this particular scenario, the client certificate, which you uploaded to the device, is sent to the server.

Whenever there is an internet connection, the train can securely publish its data to AWS IoT core. Those updates will be forwarded into Kinesis Data Stream.

Detecting anomalies and reporting the event back to the train ("Let the train know")

Kinesis Data Stream -> Kinesis Data Analytics -> Kinesis Data Stream -> AWS Lambda -> Device Shadow -> IoT topic -> IoT sensor

In order to let the train know the processed data as soon as possible, you need to send the data back very quickly.

1. How can you query the collected streaming data (by Kinesis Data Analytics!)
2. How can you identify anomalous behaviour in that data (RANDOM_CUT_FOREST in Kinesis Data Analytics!)
3. How can you quickly report any identified anomalies (by sending the result to again, Kinesis Data Stream!)

Kinesis Data Analytics can:

1. Capture streaming data (source data could be Kinesis Stream, Firehose, and even reference data source in S3) (supports many input formats such as JSON, CSV, variable column, and unstructured text) (**each input has a schema, which is inferred, but you can also edit it**)
2. Run standard SQL queries on that data (ability to build streaming application with one-to-many SQL statements) (support for: robust SQL and advanced analytic functions (e.g. random sampling, anomaly detection), "At least once" processing semantics. Extension to the SQL standard, to work seamlessly with streaming data)
3. Send the processed data to analytics tools (multiple destinations available : S3, Redshift, ES (through firehose), Kinesis Data Streams (with AWS lambda integration for custom destinations)) (End-to-end processing speed as low as a sub-second, separation of processing and data delivery)

You can detect anomalies using advanced algorithm (RANDOM_CUT_FOREST) in Kinesis Data Analytics. Then, how would you "let the train know"?

Device shadow will send that data to IoT sensor as soon as it establishes network connection.

Class Discussion 1

Q: How many input streams can you have in an Amazon Kinesis Analytics application?

A: only one stream per kinesis analytics application

Class Discussion 2

Q: Your CEO wants a real-time view of the results of your anomalies detection. How would you provide that?

A: Build your own custom lambda function to put the data into Amazon ElasticSearch cluster. (note: you can't use kinesis firehose! (minimum 60 seconds latency))

Making the information available to our team of engineers and data scientists (sharing)

IoT rule -> Kinesis Data Firehose -> transformation with AWS Lambda (optionally diverse to backup amazon S3 bucket) -> intermediate amazon S3 bucket (which also stores 1. The result of kinesis analytics, 2. Amazon redshift from quicksight)

It is not a no-brainer like giving everyone an access to raw data. Different users have different expectations on how data looks like.

- Data scientist: need access to raw sensor data (=> S3)
- Engineers: Need access to processed data (=> Redshift)
- Business users: Need access to a few simple, interactive reports (=> QuickSight)

Sensor Data DR (disaster recovery)

- Consider using cross-region replication is S3
- Create a lifecycle policy
- Use amazon glacier for data affordable archiving and long-term backup

Sharing the Sensor Data on Amazon S3

- Encrypt the data at rest by CSE, SSE-KMS, SSE-C, …
- Protect bucket contents by IAM policies, bucket policies
- Audit access by AWS cloudtrail

Storing Processed data on Redshift

Since transforming data is at risk, you can source records to backup S3 bucket.

The data from intermediate S3 bucket is loaded into Redshift using COPY command (massively parallelized) COPY (S3, EMR, DynamoDB, multiple data source from remote hosts (SSH) => RedShift)

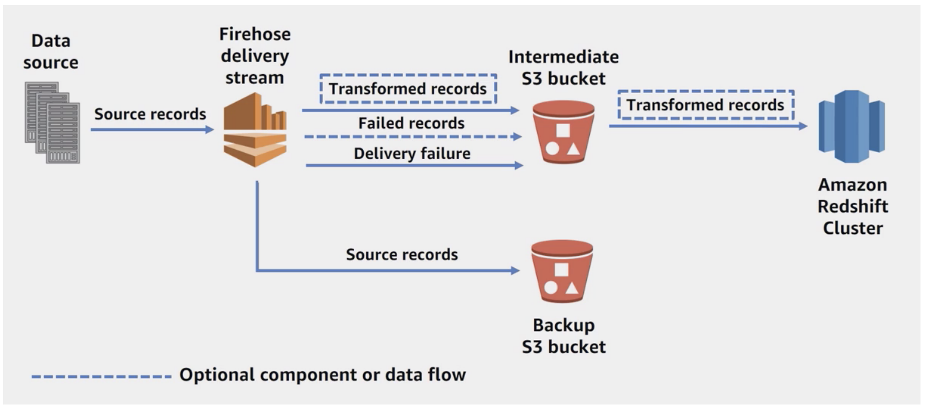

Figure 3: image

Sharing Processed data on Redshift (security on RedShift)

Cluster

- Is in VPC and a private subnet
- can be accessed via AWS direct connect
- Is encrypted with AWS KMS / AWS CloudHSM
- Obtain data in Amazon S3 through S3 endpoint

Access

- restricted by security group
- clients use TLS
- access via user name and password or SSO (single sign-on) from you Identity Provider

Setup

- A nonstandard port
- Amazon CloudWatch alarms

Creating reports on QuickSight

1. Authorize connections to your Amazon Redshift cluster (by creating Security group allowing connection from VPC where redshift resides)
2. Connect to Amazon RedShift and select a table in QuickSight
3. Analyze in Amazon QuickSight

Class Discussion 3

Q: How can you grant access to an S3 bucket for all users behind a corporate firewall?

A: implement a **bucket policy** based on the corporate firewall IP address since the request is for bucket-centered (not for grained objects).

Sample Exam question 1

You are setting up a new Amazon ES cluster outside a VPC and must give access to a team to build Kibana dashboards. You want to configure the Amazon ES Access policy to allow access to Kibana from instances inside a public-facing VPC.

You should grant access to Kibana based on:

a) ~~The IAM role used by the Amazon EC2 instance profile~~ (Kibana is not an AWS service so it doesn't have corresponding IAM role)
b) The elastic IP address used by the VPC manages NAT gateway
c) ~~user name and password that you create for each user~~ (pain stalking process to do)
d) ~~the membership of an active directory built with the AWS directory service~~ (Kibana doesn't integrate with active directory)

Sample Exam question 2

What configuration details should be investigated when tuning a Redshift "SELECT" query against a single table? Select TWO

a) the enumber and type of columns in the table.
b) ~~primary and secondary key constraints~~ (not interested)
c) alignment of table's sort key with the predicates in the "SELECT" statement.
d) ~~the number of rows in the table~~ (columnar storage, so it doesn't have such kind of info)
e) ~~the partition scheme for the database table~~ (partitioning is in the "cluster" level. It doesn't support partitioning things in "data" level within database objects)

Sample Exam question 3

What is the best way to incrementally load data from SQL server running on Amazon EC2 to Amazon Redshift?

Answer: Use AWS DMS to copy data into Amazon S3 and use Amazon RedShift Spectrum to query the data

Pitfall option is, Use AWS DMS to load data directly into Amazon Redshift (DMS copies the source DB to target DB, also loading has some risk. It is always better to copy the data to keep in sync)