PROBLEM STATEMENT

Design a system to collect and report upon data for in-water temperatures.

We are supporting a team of scientists who are working on Climate Change and have developed a unique device to measure the water temperature and report via the internet.

The device runs on solar power and is anchored to ensure it doesn't move. In addition, it contains a small cellular radio that communicates with a major telco to ensure it can connect to the internet. However, it's not a guarantee that the device will always communicate, without fail.

Every X seconds, the device will measure the water temperature and send the information, along with the unique device-id, to a given REST endpoint. The timer for the transmission event starts when the device is turned on.

Using lots of these devices, our scientists have defined a list of GPS coordinates for the devices to be deployed. This deployment will be performed in the Puget Sound, and will be dropped by a contractor on their boat. The devices have been preconfigured with the endpoint and the interval for transmitting temperatures.

We need to create a system to handle the incoming data, process it and answer several questions for our scientists. The system should scale from 1 to 1m devices, should be resilient and allow us to answer the following questions:

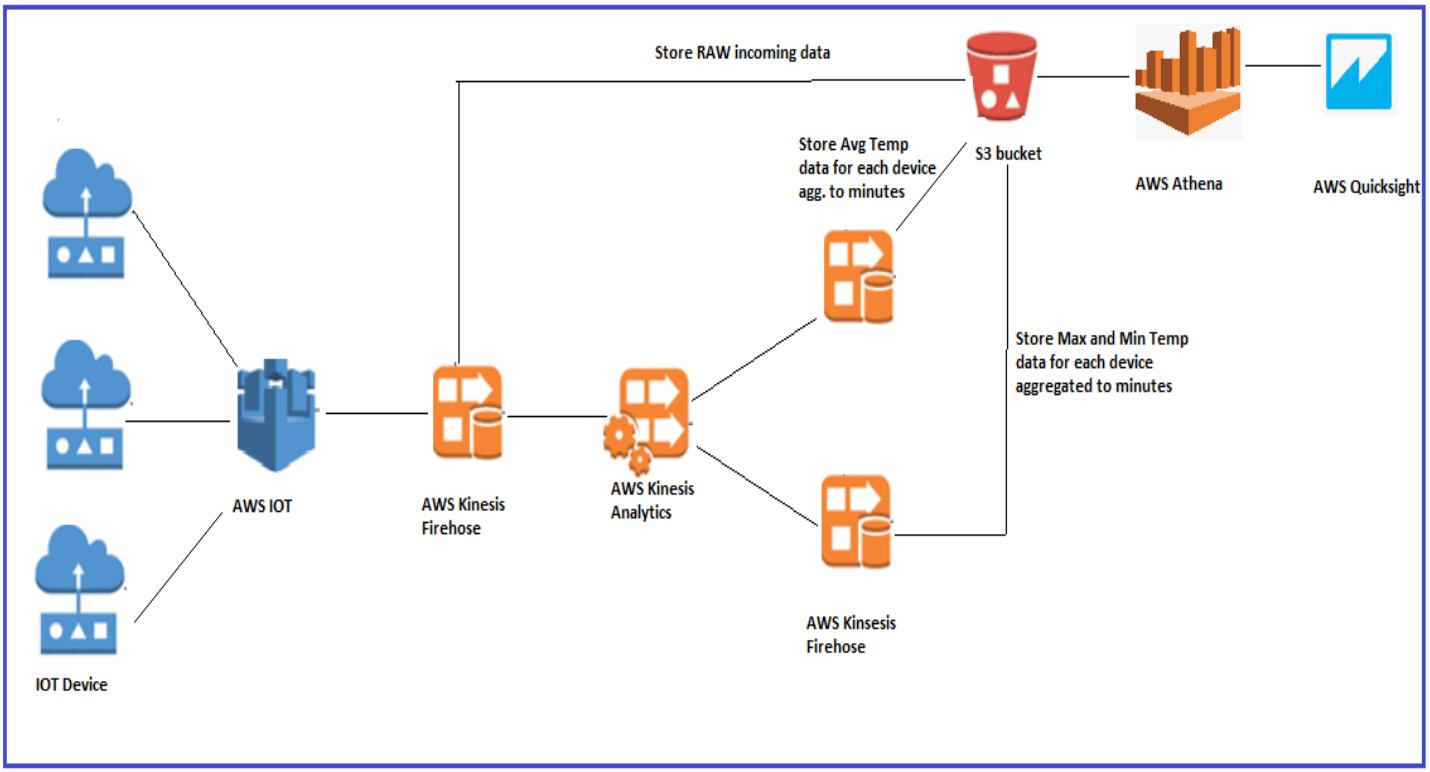
- What system architecture do we need to handle the volume of data?
- What are the scaling concerns?
- How do we compute Min/Max temp across a list or all of devices for a given HH:MM?
- How do we compute Avg temp across a list or all devices for a given HH:MM?

METHODOLOGY

- **System Architecture and Scaling:**
  In order to continuously stream the temperature data from nearly 1 millions devices, we have to ensure that system architecture is highly scalable and fully managed. For this analysis, I would go with a solution which is serverless and fully managed so that we can focus more on generating business insights and less on maintaining the backend infrastructure. Popular choice would be to use the **AWS serverless and fully managed services however Microsoft Azure is also preferred choice**. Following AWS services can be leveraged for this use case

- **AWS IOT :** to ingest Device data
- **S3:** store incoming and processed data
- **AWS Kinesis Firehose:** process, transform and load data to S3 bucket
- **AWS kinesis Analytics:** process the stream data using standard SQL
- **AWS Athena:** SQL interface for quick analysis on stored data on S3
- **AWS quicksight (optional) :** Data visualization

High Level system Architecture:



1. **First step** in the architecture is to make sure that IOT device is registered as a Thing in the AWS IOT. This will ensure that devices are authenticated and can easily communicate and send data to AWS IOT service. We also have to configure rule and actions in AWS IOT to make sure that as soon as the rule is triggered then action gets executed which in turn sends data to Kinesis firehose

2. **Second step** is to create 3 firehose delivery streams
   - Source_Raw_data : sends the raw incoming data from IOT device to a specified location in the S3 bucket
   - Avg_temp_data : Sends minute level aggregate temperature data for each device to a specified S3 bucket
   - Min_Max_Data: Sends minute level aggregate Min and Max temperature data for each device to a specified S3 bucket

   Firehose auto scale based on incoming throughput and therefore we do not have to be concerned about scaling. Firehose has near real time data processing buffer time of 60 seconds which means that latency is almost near real time ~60 seconds

3. **Third Step** is to ensure that AWS IOT receives and sends data to the Firehose delivery stream (Source_Raw_Data). We have to create a new rule and action to make sure that we route the incoming data from the IOT device to the Source_Raw_Data firehose delivery stream created in step 2. It also stores the raw incoming data to S3 bucket

4. **Fourth step** is to process raw incoming data coming from the third step by building an analytics application using Kinesis analytics. Kinesis analytics auto discovers the schema on the data and has SQL editor which can be leveraged to build sql query to create two output streams
   - Calculating Avg temperature for each device aggregated to a minute level
   - Calculating Min and Max temperature for each device aggregated at the minute level

5. **Fifth step** is to take the output stream data created above and send it to the following Kinesis firehose streams (created in step 2) which in turn store the aggregated data to a specified folder in the S3 bucket as CSV file. S3 bucket is organized/partitioned into year,month,day,hour,minute for faster access to data
   - Avg_temp_data
   - Min_Max_Data

6. **Sixth Step** is to create Athena tables partitioned by year,month,day,hour,minute on top of the data that is stored in the S3 bucket folders for faster query to data
   Athena gives a schema to data stored in the S3 bucket. We can also build precomputed views which contain business logic in a denormalized fashion . These views can be sources to data scientists for easy access to data or to the data visualization service like AWS Quicksight

   For example for if the data scientist would like to see and analyze the data for Avg temperature,
   Engineers can configure the Athena tables to specify the S3 folder location that stores the Avg temp data for each device aggregated to the minute level
   `s3://<YOUR_BUCKET_NAME>/data/<YEAR>/<MONTH>/<DAY>/<HOUR>/<MINUTE>`

   Also if the data scientist would like to see and analyze the Min and Max temperature data,
   Engineers can configure the Athena tables to specify the S3 folder location that stores the Avg temp data for each device aggregated to the minute level
   `s3://<YOUR_BUCKET_NAME>/data/<YEAR>/<MONTH>/<DAY>/<HOUR>/<MINUTE>`

7. **Seventh Step** is an optional step to do data visualization on the data by connecting Quicksight to Athena