UC Berkeley School of Information

W205 Storage and Retrieval

Project Proposal

Fall 2016

Cloud-based IOT Server for Energy Savings

Team: W205-3: Matthew Burke, Jan Forslow, Vyas Swaminathan and Xiao Wu

Project and Industry Background

The initial phase of Smart Meter deployments occurred in the United States around 2007-2010 thanks to government incentives that became available at that time. The main target use case at that time was to automate billing reads. As such the systems deployed were largely optimized for low velocity data (mainly once a day data collection). The servers were largely on premise and used standard relational databases for data storage.

Since then a lot have changed. Distributed Energy Generation (e.g. PV solar) has taken off and created a two-way power grid in need for greater control. Energy Efficiency programs are being launched (e.g. using NEST smart thermostats) requiring more granular validation of energy consumption based on time of use. Novel new ideas to save energy are proposed within the utilities themselves (e.g. Volt/VAR Optimization) requiring sub 10-minute instantaneous voltage from a subset of bellwether Smart Meters on the distribution feeder line.

This results in that a new backend system is required that is more scalable and can receive and analyze streaming data at high velocity. In this project we propose to design a demonstrator of such a Cloud-based IoT Server for Energy Savings programs using the building blocks available inside AWS.

Potential Data Sources

Two different data sources have been examined so far: live feeds and simulator-generated Smart Meter data.

There are several live feeds from Smart Meters available. The following web site (https://data.sparkfun.com/streams/QGxRll4341u0019rMbXM) is one example from the Netherlands that updates its live feed every 10 minutes. The proposal is to amplify this feed to reflect millions of Smart Meters and use as a continuous background feed into the IoT Server.

On top of this we will layer of more advanced event data stream using an IoT Device Simulator. The events coming from the simulator will largely be based on the Smart Meter ANSI C12.19 standards. Example events are shown below.

Low Battery	Under Voltage	Real Time Pricing Activation	Remote Disconnect Switch Open	Tamper Detection
High Temperature	Over Voltage	Real Time Pricing Deactivation	Remote Disconnect Switch Closed	Tamper Detection Cleared

Potential System Architecture

As mentioned previously, we will attempt to use the available building blocks from within AWS to create a scalable, cloud-based IoT Server so that the demonstrator can more easily be taken to commercialization if generating interest. Specifically, we will attempt to leverage the following components:

- **AWS IoT** (Amazon Internet of Things) is an Amazon Web Services platform that collects and analyzes data from internet-connected devices and sensors and connects that data to AWS cloud applications.
- **Amazon Kinesis Firehose** is a platform for streaming data on AWS, offering powerful services to make it easy to load and analyze streaming data; and also providing the ability for you to build custom streaming data applications for specialized needs
- **Amazon Elasticsearch** is a popular open-source search and analytics engine for use cases such as data analytics, log analytics, real-time application monitoring, and click stream analytics.

We plan to use **AWS Lambda** to create server-less APIs and code snippets in between the different building blocks.

Potential Challenges and Initial Plan

We will spend some time initially to create a simulator. We will do this within the framework of AWS as an own IoT Device Simulator EC2 instance. We believe that getting the right simulator algorithm could be a challenge and may revert back to use static data logs as input.

The second phase will be focused on the data storage and retrieval architecture, including selecting the right database for this use case, e.g. **Amazon S3**, **Amazon DynamoDB** (noSQL), **Amazon EMR** (Hadoop) and/or **Amazon Redshift** (Columnar Data Warehouse). We will spend a fair amount of time on this and document the pros and cons with the different options. We will then work through ER diagrams, DAGs and associated query optimization strategy for the selected setup.

We will provide some analytics and dashboard functions, but these will largely use the easily accessible capabilities in **Amazon Elasticsearch**, including Kibana, and **Amazon Machine Learning**. We will not aim to spend too much time on this aspect of the project as this can be tailored in subsequent commercialization phase by domain experts.

Finally, we will scale out and test the system.