Smart-home-as-a-service with AWS Project Report

Tristan Brisker
College of Engineering
Florida Gulf Coast University
Fort Myers, USA
tsbrisker0470@eagle.fgcu.edu

Yailin Dominguez Santos
College of Engineering
Florida Gulf Coast University
Fort Myers, USA
ydominguezsantos6907@eagle.fgcu.edu

Amber Jones
College of Engineering
Florida Gulf Coast University
Fort Myers, USA
anjones4463@eagle.fgcu.edu

Yanisley Mendiola
College of Engineering
Florida Gulf Coast University
Fort Myers, USA
ymendiola2865@eagle.fgcu.edu

Erick Rodriguez
College of Engineering
Florida Gulf Coast University
Fort Myers, USA
erodriguez5736@eagle.fgcu.edu

***Abstract*—This project aims to simplify the monitoring of smart home devices for elderly individuals using a combination of AWS Lambda, Prometheus, Grafana, and AWS EC2 cloud services. The system uses simulated data from AWS Lambda, which gets pulled by Prometheus, and then is visualized by Grafana. Notifications will be sent to the user via Discord of whether or not their appliances are running or not. While the integration of the Amazon Smart Plug would've been ideal, it posed challenges due to its lack of scraping capabilities. In the future, we would've liked to expand the system to be able to support a wider range of appliances.**

I. Introduction

The rapid growth of IoT these days has led to the widespread adoption of smart home devices. They offer increased convenience, automation, and energy efficiency. However, for elderly individuals or not as tech-savvy people, it can be challenging to manage and monitor these devices. Some solutions may not always be user-friendly or provide real-time feedback which can make it difficult to ensure that their devices are functioning as intended. This project seeks to address these challenges by creating a simple and effective way to monitor your smart home devices through a cloud-based system. By integrating AWS Lambda, Prometheus, Grafana and AWS EC2, the system we created provides an intuitive solution for users to track their appliances' status in real-time. AWS Lambda simulates our data from various devices, which then is scraped by Prometheus for storage and retrieval. Grafana serves as our visualization platform, which displays the data in a clear and accessible way. The user is then notified via Discord alerts, informing them that their devices are firing, with the appropriate message depending on the appliance. This ensures that the users can easily manage their smart home devices without requiring technical expertise.

II. Project Description

A. Smart-home-as-a-service with AWS Project Report

The primary goal of this project is to create an accessible and user-friendly platform for elderly individuals and others who need to monitor their smart home devices in real-time. The solution we created integrates all four: AWS Lambda, Prometheus, and Grafana to provide a cloud-based monitoring system.

III. Architectures

A. Prometheus

When developing our smart home device watch system, Prometheus was key for the collection and monitoring of the data. An interval of fifteen seconds was set up to scrape the data from Lambda. Prometheus comes with its own AlertManager architecture; however, it was not used for this project.

B. Grafana

After Prometheus completed its role in gathering the data, it was then transferred to Grafana to display visualizations of our metrics in graphical form. The data was collected from appliances such as the washing machine, refrigerator, air conditioning, and TV. The green power consumption line represented air conditioning, which ranged from one thousand to fifteen hundred watts. The red line indicated the TV consumption between three hundred to eight hundred watts. The yellow line was for the refrigerator and that was from zero to two hundred watts. Lastly, the blue line signified the washing machine and that ranged from zero to fourteenth hundred watts. Grafana comes with its own AlertManager just like Prometheus but it has a simpler alert management configuration. In addition, Grafana AlertManager has more methods of communication between AlertManager and the user compared to Prometheus (just email). For this project, we decided to use Discord as the only contact point since it was the only communication method of our group; so, it made it easier to test the alerts.

C. AWS Lambda

AWS Lambda was used to simulate real-time data power consumption on all four appliances. The appliances were put together inside one function for the program we created. The metrics function generated randomized values within a set range, which were used to account for power consumption. A list was initialized inside that stored the metrics in a format compatible with Prometheus. When Lambda got triggered, the handler function would send data for the generated power as a plain text to Prometheus. In the future, we would like to separate each appliance and put them into four different functions when simulating the data.

D. API Gateway

This is used to create the link between the functions in our AWS Lambda code to our Prometheus server. Since there is no way to directly link the code to the Grafana dashboard to be displayed, it has to be connected to Prometheus so that it can be run and compiled into a graph. Essentially in order to do this, it exposes the AWS Lambda functions as an HTTP endpoint which allows Prometheus to access the information. Using this system ensures that we get the data that is needed for the dashboard with little integration issues and security of the information (so that outside systems cannot access it).

E. AlertManager

The AlertManager was used in sending notifications to the client (in our instance, we used discord, but a more convenient use is a phone number or email address) that would notify if there were any drastic changes in the power consumption. It would also send notifications if appliances (like the TV) were left on for an extended period. This system is integrated into Grafana so that it can read the data directly from the graphs being displayed by the metrics it is collecting from Prometheus data source. Using the data, it will send notifications based on the given parameters of each appliance. These notifications consist of vital information such as a summary of the appliance being used, status of that appliance, and description of what the alert is for.

IV. Project Process

A. First Stage

Before working directly with smart devices, we reviewed the available options and identified two that aligned with our project goals: the Smart Home Hub and the Arlo Ultra 2 Outdoor Camera. While waiting to get our hands the devices, we did some research on the systems we would use: AWS Lambda, Grafana, Prometheus, and Alertmanager. With this knowledge, we developed a general plan: integrate the outdoor camera with Grafana, display the camera's live feed on the dashboard, configure motion detection alerts via Prometheus' Alertmanager, and have it all connected to the Smart Home Hub.

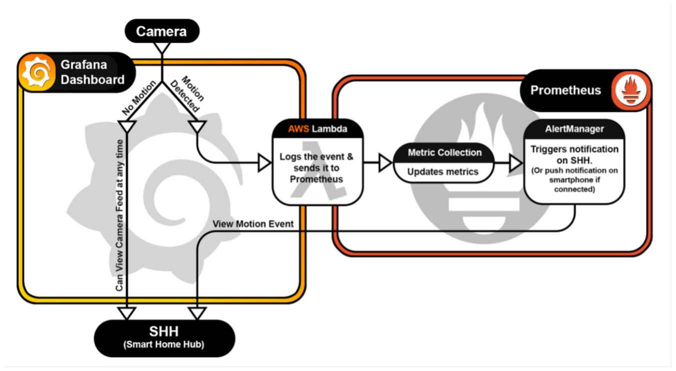
Camera
Grafana Dashboard
No Motion
Motion Detected
Can View Camera Feed at any time
AWS Lambda
Logs the event & sends it to Prometheus
Metric Collection
Updates metrics
Prometheus
AlertManager
Triggers notification on SHH.
(Or push notification on smartphone if connected)
View Motion Event
SHH
(Smart Home Hub)

Fig. 1. Initial diagram for the project.

B. Second Stage

Due to availability issues, we switched to alternative devices: the Amazon Echo Show 15 and Amazon Smart Plug. After some research on the new devices, we revised our plan to focus on the smart plug's ability to measure the power consumption of appliances connected to it. The data collected by the smart plug would be processed by an AWS Lambda function and scraped by Prometheus and turned into a dataset, enabling Grafana to display the data on a graph. The dashboards would be viewed on the Echo Show 15' via the Amazon Silk browser. For practice, we started working with Grafana, creating dashboards with dummy data to display on graphs.

C. Third Stage

To ensure that we have a demonstration by the deadline, we divided the workload into three tasks: collecting smart plug data, enabling cloud access for the Grafana dashboard, and configuring Alertmanager. However, we encountered a limitation, the Amazon Smart Plug couldn't collect power data without extensive modifications. Due to time constraints, we decided to simulate the data instead. Once simulated, it was scraped by Prometheus and turned into a dataset. We then created a Grafana dashboard with one graph that visualizes power consumption of the four appliances. We also configured Alertmanager to send Discord notifications when an appliance has hit the max or minimum consumption and when they are off. This setup was finalized for the demonstration.

D. Final Stage

In the final stage, we enhanced the Grafana dashboard by visualizing each appliance's power consumption on individual graphs. Four separate graphs were created, each accessing the same dataset but configured to only display the data for their designed appliance, ensuring a polished presentation for the demonstration.

V. Project Challenges

A. Major Challenges

The major challenges for this project were the availability outside of class, the work strategy within the team, and the devices given to us for the project. It was extremely difficult to get the group together, especially in person, because we all had different schedules and could not find time to work on it. A couple of students had to miss other classes to attend the in-person meetings. Virtual meetings also did not work out very well until the last two stages of the project because some team members had busy schedules outside of school. This is the reason we decided to change our work strategy after Stage 2 and distribute the workload: two group members worked on configuring an AWS Lambda function to send metrics to Prometheus, two group members worked on setting up the connections between Grafana and Prometheus and configuring the graphs, and one team member worked on configuring the alerts on Grafana AlertManager. By the end of Stage 3, we were almost done with this project simulation, and the change in work strategy was successful. Lastly, the devices given to us — an Arlo camera, an Amazon Smart Plug, and an Amazon SmartHub — did not come with capabilities to obtain metrics from any of the devices. It was impossible to get the metrics from them and use them for our project. Finally, we relied on AWS Lambda to create a function that simulated the data since we could not connect to the Alexa API from the SmartPlug.

B. Minor Challenges

The only minor issue we encountered on this project was the lack of knowledge about Prometheus, Grafana, AlertManager, AWS Lambda, and API Gateway. None of the group members had worked with any of these architectures before, so we all needed to do research on them before starting the project, and we had to rely on documents from the architectures' websites to find solutions

References

[1] "Manage dashboards: Grafana Documentation," Grafana Labs, https://grafana.com/docs/grafana/latest/dashboards/manage-dashboards/(accessed Nov. 24, 2024).

[2] "Get Started with Grafana and Prometheus," Grafana Labs, https://grafana.com/docs/grafana/latest/getting-started/get-started-grafana-prometheus/ (accessed Nov. 22, 2024)

[3] "Queries and condiditons," Grafana Labs, https://grafana.com/docs/grafana/latest/alerting/fundamentals/alert-rules/queries-conditions/ (acessed Nov. 22, 2024)

[4] "Data soueces," Grafana Labs, https://grafana.com/docs/grafana/latest/datasources/

[5] Amazon API Gateway," AWS documentation, https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html

[6] lux4rd0, "GitHub - lux4rd0/kasa-collector: Kasa Collector provides a way of collecting real-time energy data from Kasa Smart Plugs. These Grafana dashboards offer visualizations for their Current, Voltage, Power, and Total Watt Hours.," GitHub, Oct. 03, 2024. https://github.com/lux4rd0/kasa-collector (accessed Nov. 18, 2024).