The Internet of Things on AWS – Official Blog

Connecting home appliances with a smart home solution built on AWS in the AWS China Region

by 殷实 | on 11 MAR 2021 | in Amazon API Gateway, Amazon API Gateway, Amazon Cognito, Amazon RDS, Amazon Simple Storage Service (S3), AWS IoT Core, AWS IoT Device Management, AWS IoT Greengrass, AWS Lambda, Customer Solutions, Internet Of Things, RDS For MySQL | Permalink | ➦ Share

This blog post introduces how the manufacturers of home appliances can use AWS Services to build and maintain their smart home solutions. These solutions are both the platforms that power their connected products as well as the applications consumers use to control those products. This blog post illustrates a real use case from a customer that manufactures home appliances in China and sells them throughout the world. We will walk through a reference architecture that outlines their end-to-end solution and describe how the customer uses AWS IoT together with other AWS Services for their core use cases. You'll learn how the customer built a secure way for smart home appliances to share information with the families that use them, and how the customer securely manages their fleet of home appliances at scale within the AWS China Region. This blog post includes Python code snippets so you can implement a similar solution, where a device sends data to a smart home IoT platform and securely shares multimedia files from the device to customer-facing mobile or web applications.

Home appliances are becoming smarter as their users interact with them

With the continuous development of IoT and AI, smart home device growth has accelerated. The trend towards smarter devices and the smart home as a whole is primarily for two objectives:

1. Intelligent and personalized interaction between home appliances and end users
2. Ease of interoperability between two or more intelligent home appliances

AWS IoT makes it easy for you to build scalable IoT applications that collect, process, analyze, and act on data generated by connected home devices without having to manage any infrastructure. AWS IoT also integrates with other AWS Services, so you can easily build complete smart home solutions and focus your efforts on delivering new experiences and adding even more value to your consumers.

This blog post focuses on two key services within the AWS IoT service portfolio – AWS IoT Core and AWS IoT Greengrass. In addition to AWS IoT Core and Greengrass, AWS has broad and deep IoT services, from the edge to the cloud, and provides a powerful and comprehensive ecosystem of technology and solutions that help customers build, manage, and continually improve their IoT devices and platforms. For more information about AWS IoT, refer to https://aws.amazon.com/iot/.

AWS IoT Core

AWS IoT Core is a fully managed service that lets you connect IoT devices to the AWS Cloud and to other IoT devices without the need to provision or manage servers. For more information about AWS IoT Core, refer to the AWS IoT Technical Documentation: https://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html

AWS IoT Greengrass

AWS IoT Greengrass seamlessly extends AWS Services to physical devices to enable them to operate locally on the data they generate while still using the AWS Cloud for management, analysis, and persistent storage. For more information about AWS IoT Greengrass, refer to the AWS IoT Greengrass Technical Documentation: https://docs.aws.amazon.com/Greengrass

In the next section, we will describe a number of other AWS Services that this customer used to build their IoT infrastructure for their smart home use case.

Smart Home platform powered by AWS IoT services

In this section, we'll discuss the real customer use-case and illustrate how AWS IoT services play a main role in their smart home solution.

Solution Background

First, let's understand the background of what challenges the customer sought to overcome, and the technical architecture of their smart home solution. This customer is a world-class manufacturer and seller of home appliances. They wanted to build a new IoT infrastructure so that they could easily connect those appliances to the cloud, manage them at scale, and make their products more intelligent. To achieve these goals, they built a solution using a number of AWS Services, such as:

- AWS IoT Core: easily and securely connect devices to the cloud, and reliably scale to billions of devices and trillions of messages
- AWS IoT Device Management: register, organize, monitor, and remotely manage connected devices at scale
- AWS IoT Greengrass: bring local compute, messaging, data management, sync, and ML inference capabilities to edge devices
- Amazon Cognito: offer simple and secure user sign-up, sign-in, and access control
- Amazon API Gateway: create, maintain, and secure APIs at any scale
- AWS Lambda: run code without thinking about servers or clusters. Only pay for what you use
- Amazon S3: object storage built to store and retrieve any amount of data from anywhere
- Amazon Relational Database Service (RDS): set up, operate, and scale a relational database in the cloud with just a few clicks

Solution Architecture

The technical architecture of the customer's solution is composed of edge devices (such as refrigerators), a mobile app for remote management, and the cloud-based smart home IoT platform used for home appliance fleet management at scale. AWS IoT Greengrass was deployed on the refrigerators which act as a gateway between other smart home devices, such as a microwave oven, and the smart home IoT platform. Authentication of users in the mobile app is handled through an integration of the customer's existing SSO solution with an Amazon Cognito Identify Pool. The smart home IoT platform is built with AWS IoT Core and AWS Lambda to provide customer device registration, easy device fleet management at scale, and the interconnection of end users and the home appliances connected to this platform.

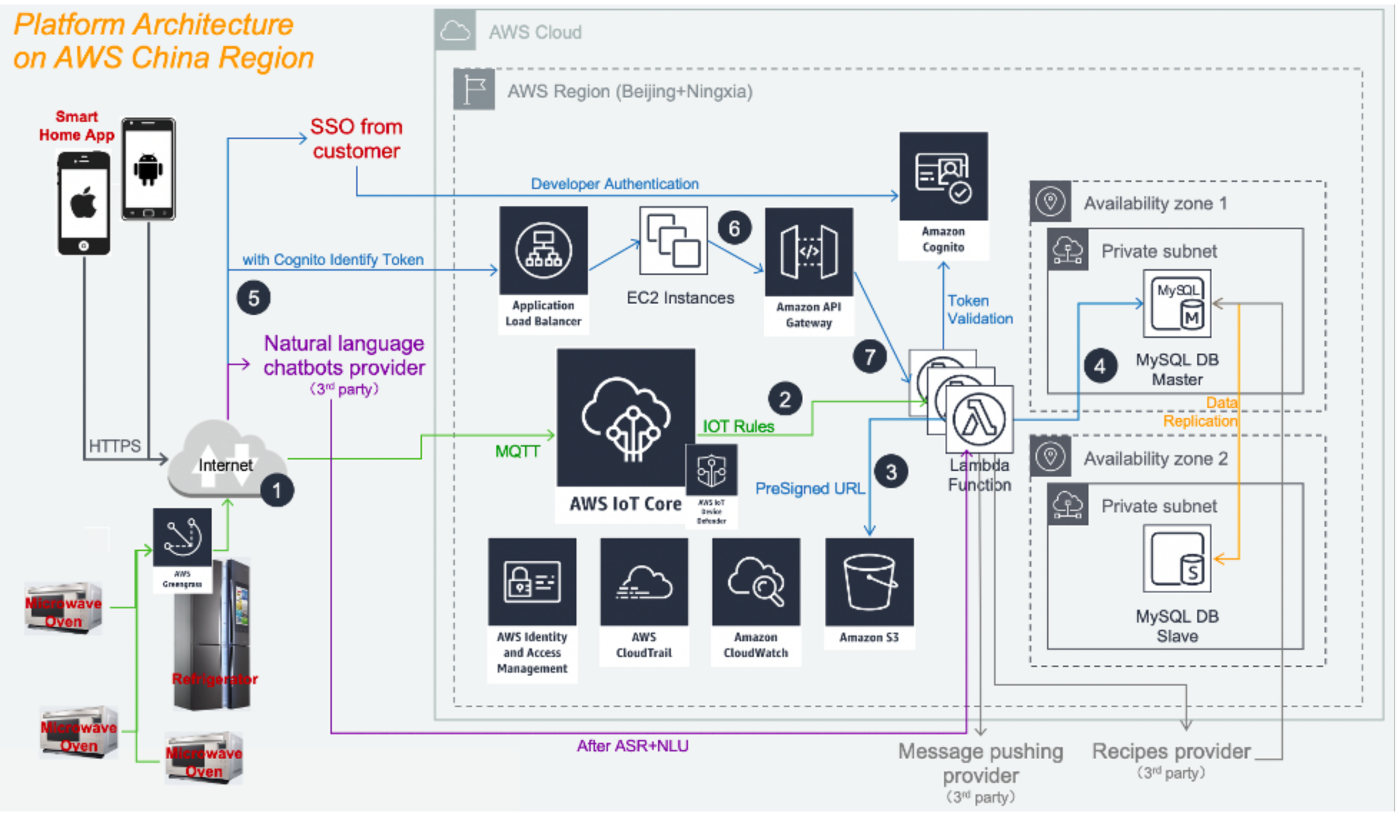


Figure 1: IoT platform architecture built using AWS Services in the AWS China Region

As shown in Figure 1, the solution implementation is:

- **Step 1:** Refrigerators connect to the internet and send messages to AWS IoT Core through MQTT protocol. Microwave ovens connect to AWS Greengrass deployed on the refrigerators and then connect to AWS IoT Core via the internet and MQTT protocol.
- **Step 2:** AWS IoT Core uses a rule engine to extract necessary data from the messages and pass the data to AWS Lambda function
- **Step 3:** AWS Lambda functions generate and then return an Amazon S3 presigned URL to store multimedia such as images, audio files, and videos
- **Step 4:** AWS Lambda functions write data into Amazon RDS MySQL
- **Step 5:** Mobile apps request identity token from the Amazon Cognito account previously integrated with the customer's AWS Directory Service, and then send requests to AWS ALB
- **Step 6:** Application-layer firewall deployed on Amazon EC2 instances protect APIs supported by Amazon API Gateway
- **Step 7:** Amazon API Gateway receive requests from Mobile apps and trigger AWS Lambda function to fulfill the requests.

Taking Action and Driving insights at the Edge

This solution uses an Android tablet on a refrigerator door to provide an end user interface and several cameras installed on the inside of the refrigerator door. The Android interface supports interactions such as typing, verbal commands, and visual via the in-door cameras. These cameras leverage an image recognition solution from a 3rd party to detect what food is being added and removed from the refrigerator. That data is updated in an Amazon RDS MySQL database. Then, the customer surfaced the data to the end users via the Android Tablet or on their mobile phone app so that the end users can gain real-time insights about what food is available in their fridge.

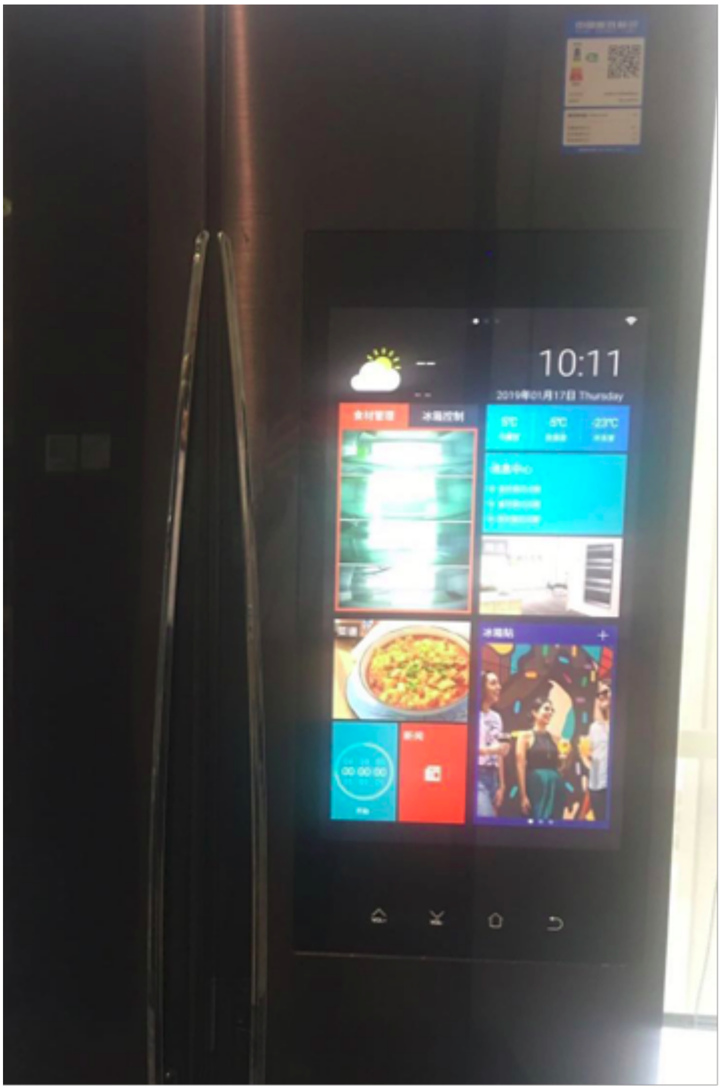

10:11
2019年01月17日 Thursday
食材管理
冰箱控制
5°C
-5°C
-23°C
冰箱贴
00:00:00

Figure 2: Tablet interface and in-door cameras in refrigerator

Solutions for Use Cases

Let’s walk through the core use cases for this solution. The use cases are

1. End users register their refrigerators using the mobile app, which adds the device to the customer’s smart home platform.
2. End users sign in to the mobile app to interact with the smart home platform.
3. End users search existing ingredients and add new ones in the fridge using their tablet and/or mobile app.
4. Customer securely manages the fleet of refrigerators registered on the smart home platform.

Solution for Use Case 1: Users Register the refrigerator

First, we will outline the process of how the refrigerators are registered to the smart home IoT platform. Each refrigerator has a preset certificate generated and placed in the tablet by the manufacturer to access to AWS IoT Core for the smart home IoT platform. The preset certificates are only granted the permissions needed to create a Thing in AWS IoT Core, which results in a new device added to the customer’s smart home IoT platform, as well as to download certificates to the refrigerator, which means security to the customer. The refrigerator uses these preset certificates to connect to AWS IoT Core and create a thing for itself, and save the client ID and download certificates to its local storage. These new certificates are not activated until the refrigerator is paired to a mobile app user. The end user uses the mobile app to scan the barcode located on the refrigerator. The barcode contains the refrigerator's device information such as a unique serial number and hardware types. Once the mobile app scans the barcode, the information is sent to the smart home IoT platform and verified. Once the verification succeeds the refrigerator is recorded in the database of the customer's device management service and the certificates of the thing for the refrigerator is activated in AWS IoT Core. Then the refrigerator completes the registration process in the smart home IoT platform. This process is highlighted in the figure below.

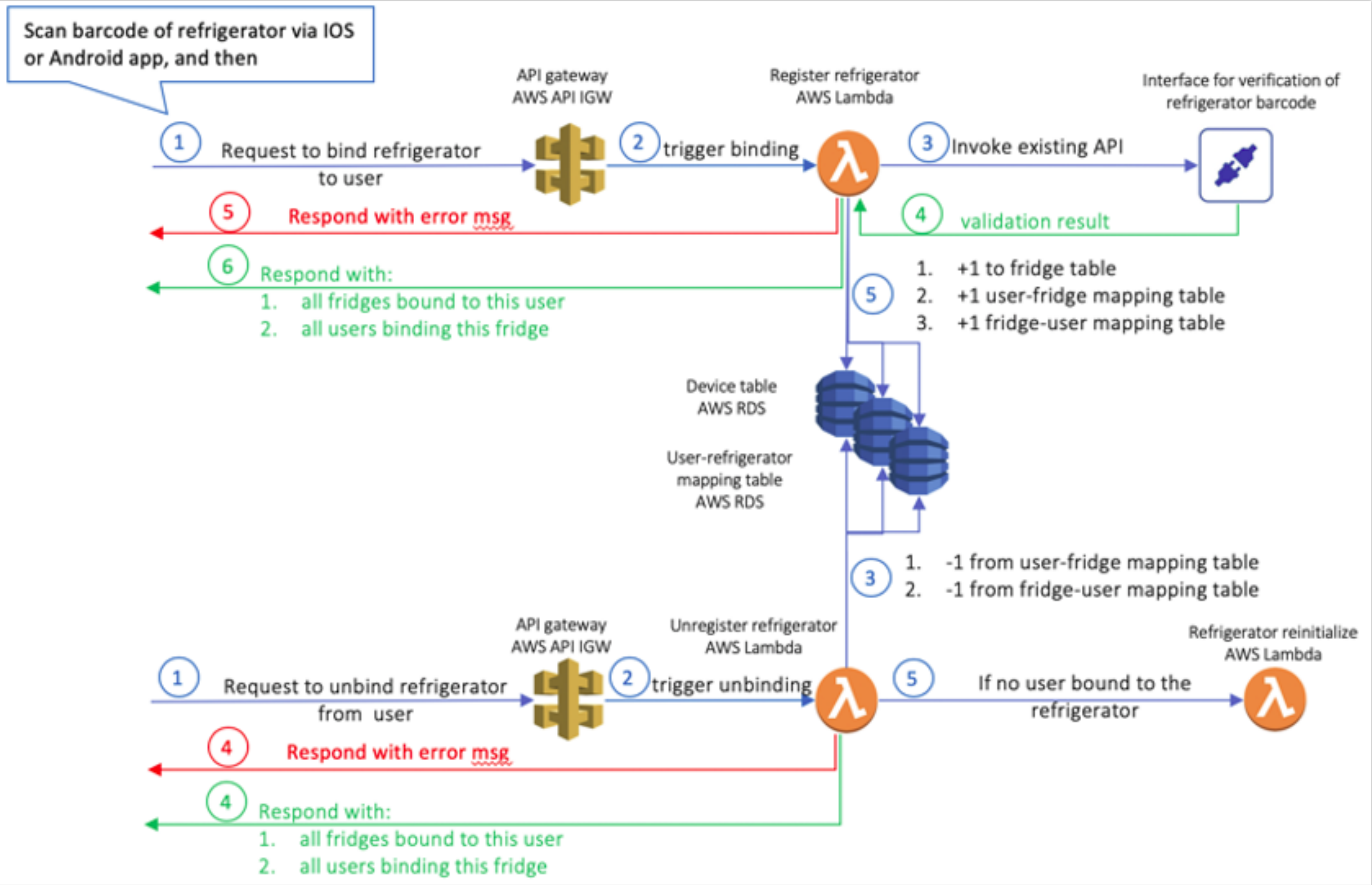

Scan barcode of refrigerator via IOS or Android app, and then
API gateway AWS API IGW
Register refrigerator AWS Lambda
Interface for verification of refrigerator barcode
1 Request to bind refrigerator to user
2 trigger binding
3 Invoke existing API
4 validation result
5 Respond with error msg
6 Respond with:
1. all fridges bound to this user
2. all users binding this fridge
5
1. +1 to fridge table
2. +1 user-fridge mapping table
3. +1 fridge-user mapping table
Device table AWS RDS
User-refrigerator mapping table AWS RDS
3
1. -1 from user-fridge mapping table
2. -1 from fridge-user mapping table
API gateway AWS API IGW
Unregister refrigerator AWS Lambda
Refrigerator reinitialize AWS Lambda
1 Request to unbind refrigerator from user
2 trigger unbinding
5 If no user bound to the refrigerator
4 Respond with error msg
4 Respond with:
1. all fridges bound to this user
2. all users binding this fridge

Figure 3: End users register their refrigerators to the smart home platform

Solution for Use Case 2: End users sign in to the smart home platform

End users need to register and log in to the smart home IoT platform in order to control the refrigerators remotely. After they register the device as described in the first use case solution, end users can use the Android tablet to perform operations from the front of the refrigerator, such as sending a voice message, without needing to log in to the smart home IoT platform. From the perspective of user management, this solution treats the end users who control the same refrigerator as a family group. For example, the husband can use his iPhone as an end user to log into the IoT platform's iOS app to control the refrigerator, and the wife uses her Android phone app as another end user to log into the IoT platform and control the same refrigerator. More importantly, the Android tablet on the refrigerator is also treated as an end user who can control the refrigerator but does not need to log into the IoT platform. Instead, the tablet uses certificates as its identity to communicate with the smart home IoT platform. These authentication approaches help the smart home IoT platform fulfill different authentication requirements to keep the refrigerator and the data it generates secure, while also optimizing the end user experience.

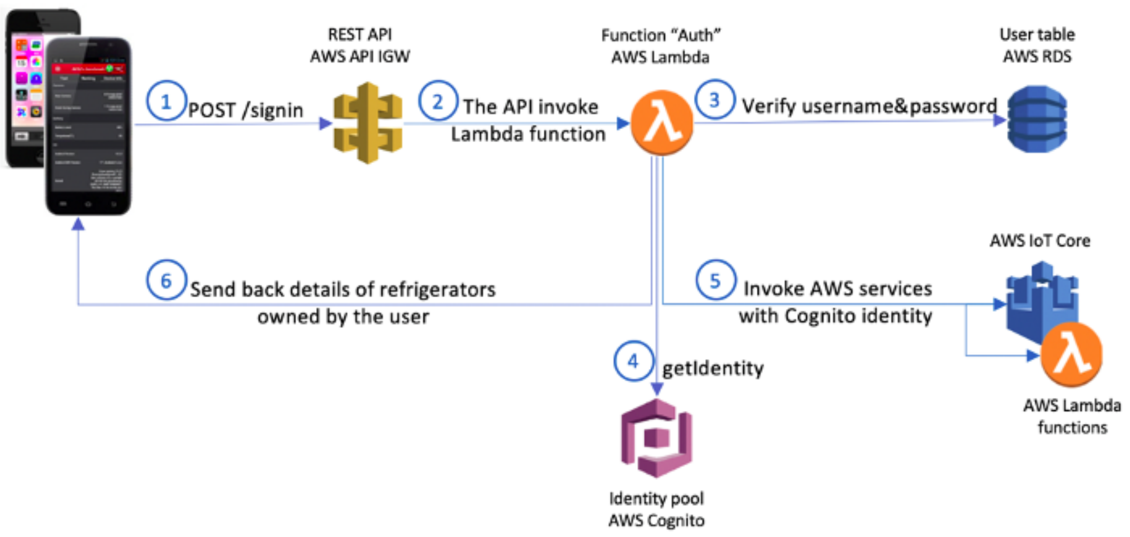

REST API
AWS API IGW
Function "Auth"
AWS Lambda
User table
AWS RDS
1 POST /signin
2 The API invoke Lambda function
3 Verify username&password
AWS IoT Core
6 Send back details of refrigerators owned by the user
5 Invoke AWS services with Cognito identity
4 getIdentity
AWS Lambda functions
Identity pool
AWS Cognito

Figure 5: End users sign into the smart home platform

Solution for Use Case 3: End users add and search ingredients from the tablet and mobile app

When a user closes the refrigerator door, an AWS Lambda function of "add ingredients" is triggered and is invoked by a REST API managed with the Amazon API Gateway service. The 3rd party image recognition solution detects if new ingredients were added to or removed from the fridge, and if any were added or removed they are inserted into/deleted from a database using Amazon RDS for MySQL. Then another Lambda function 'ingredients search' will retrieve all ingredients currently in the refrigerator and send them back to the tablet installed on the refrigerator and the user's mobile app connected to the smart home IoT platform. Both the tablet and mobile app can check for ingredients in the refrigerator in real time by video data streamed from the in-door cameras, so a user can browse what food is available in the fridge without opening the door, saving energy and preserving food quality, or when they are on-the-go from anywhere, such as at the grocery store.

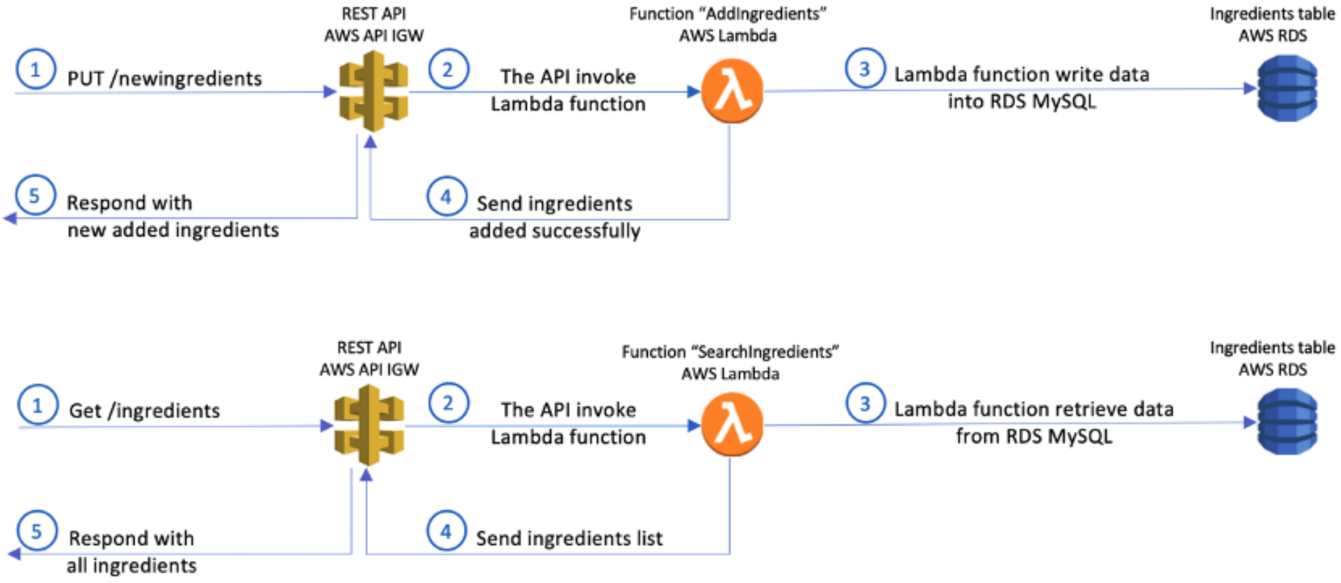
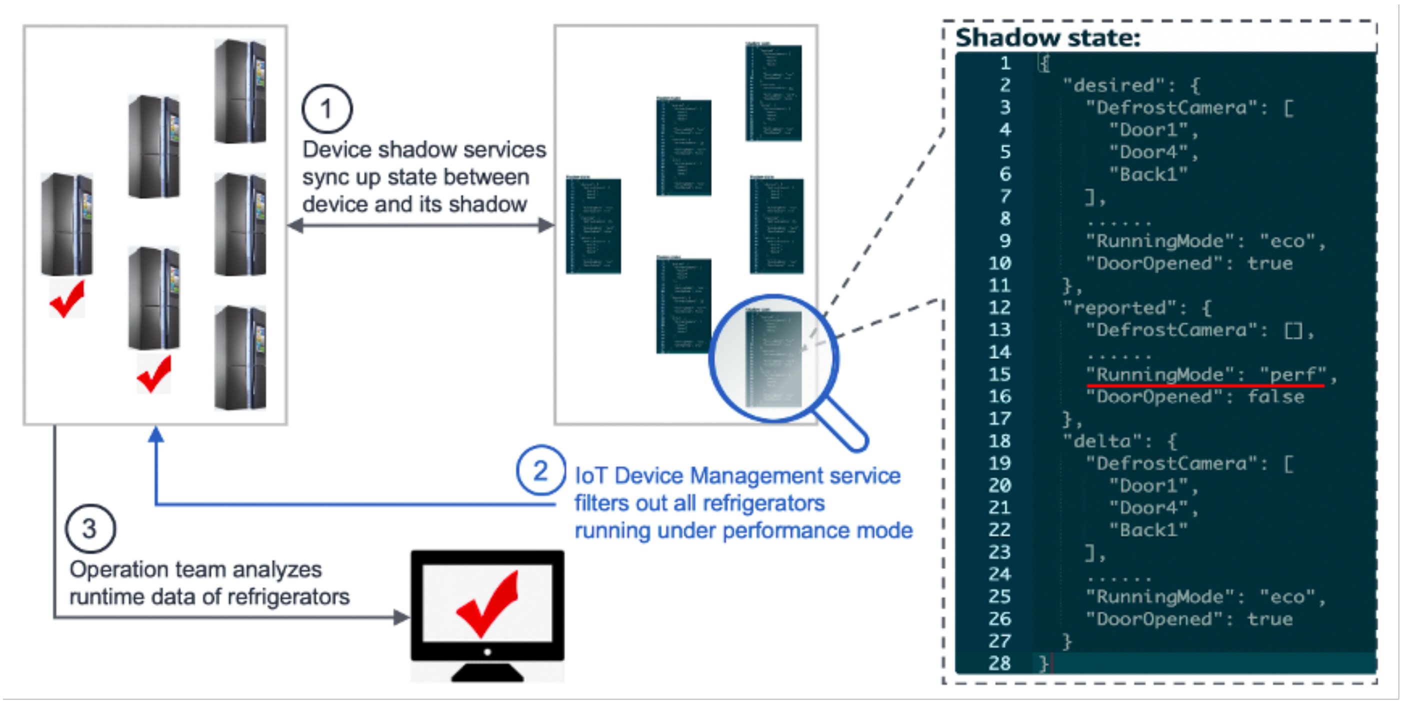

REST API
AWS API IGW
Function "AddIngredients"
AWS Lambda
Ingredients table
AWS RDS
1 PUT /newingredients
2 The API invoke Lambda function
3 Lambda function write data into RDS MySQL
5 Respond with new added ingredients
4 Send ingredients added successfully
REST API
AWS API IGW
Function "SearchIngredients"
AWS Lambda
Ingredients table
AWS RDS
1 Get /ingredients
2 The API invoke Lambda function
3 Lambda function retrieve data from RDS MySQL
5 Respond with all ingredients
4 Send ingredients list

Figure 6: End users add and search ingredients from tablet and mobile app

Solution for Use Case 4: Secure fleet management

The customer's smart home IoT platform leverages AWS IoT Device Management to organize and manage their fleet of refrigerators and store device records in a database. The thing shadows in AWS IoT Core sync with the state of the refrigerators, and AWS IoT Device Management indexes attributes, such as "RunningMode" and "DoorOpened", in thing shadows. Then the smart home IoT platform can filter out the refrigerators with specific attribute values and take action based on those attributes. For example, they can organize a things group of refrigerators with the "DoorOpened" state and trigger an alarm to both the Android tablet and the user's mobile app if the refrigerator has the "DoorOpened" state for more than 10 mins.

Figure 4: leverage AWS IoT Device Management to manage refrigerators


1
Device shadow services sync up state between device and its shadow
2
IoT Device Management service filters out all refrigerators running under performance mode
3
Operation team analyzes runtime data of refrigerators
Shadow state:

```
{
  "desired": {
    "DefrostCamera": [
      "Door1",
      "Door4",
      "Back1"
    ],
    ......
    "RunningMode": "eco",
    "DoorOpened": true
  },
  "reported": {
    "DefrostCamera": [],
    ......
    "RunningMode": "perf",
    "DoorOpened": false
  },
  "delta": {
    "DefrostCamera": [
      "Door1",
      "Door4",
      "Back1"
    ],
    ......
    "RunningMode": "eco",
    "DoorOpened": true
  }
}
```

Figure 4: leverage AWS IoT Device Management to manage refrigerators

Amazon S3 presigned URL is provided by the Amazon S3 service to temporarily share objects, such as multimedia like audio and visual files, owned by a user to other users without access credential to the objects. This allows users to upload or download objects, such as recipe cards or voice messages, to Amazon S3 buckets without permitting the users to have the right to read and write objects in the buckets. This is very important because the smart home IoT platform needs to support the scenario where end users do not provide access credentials to the smart home platform when sending multimedia information through the tablet on the refrigerators. To learn more about sharing an object with a resigned URL using Amazon S3, please refer to the technical documentation found here: https://docs.aws.amazon.com/sdk-for-java/v2/developer-guide/examples-s3-presign.html.

Summary

With a smart home IoT solution built on AWS IoT, you can provide your users with the real time status of home appliances, such as refrigerators and microwave ovens, and enable them to control these home appliances and execute operations from anywhere. You can make it easy for your users to securely share the status of their device with other family members, and to communicate with them using multimedia information such as images and videos. This makes it easier and more convenient for end users to plan out ingredients, compile recipes, and maintain a healthy diet all while enjoying more time spent with their families. We look forward to seeing how you use this example to start building the future of the smart home with AWS IoT! To learn more about AWS IoT solutions for the connected, smart home, refer to our solutions page: https://aws.amazon.com/iot/solutions/connected-home/ or contact us directly here: https://pages.awscloud.com/connectedhomecontact.html.

About the author

Shi Yin is an IoT consultant from AWS Professional Services, based in California. Shi worked with many big enterprises to leverage AWS IoT Services to build IoT platforms and connect sensors and devices to the platforms.

TAGS: AWS China Region, AWS IoT, AWS IoT Core, AWS IoT Device Management, AWS IoT for Connected Home, aws iot greengrass, connected devices, connected home, connected refrigerator, Consumer Devices, consumer IoT, device manufacturing, edge computing, IoT platform, smart home, smart home platform