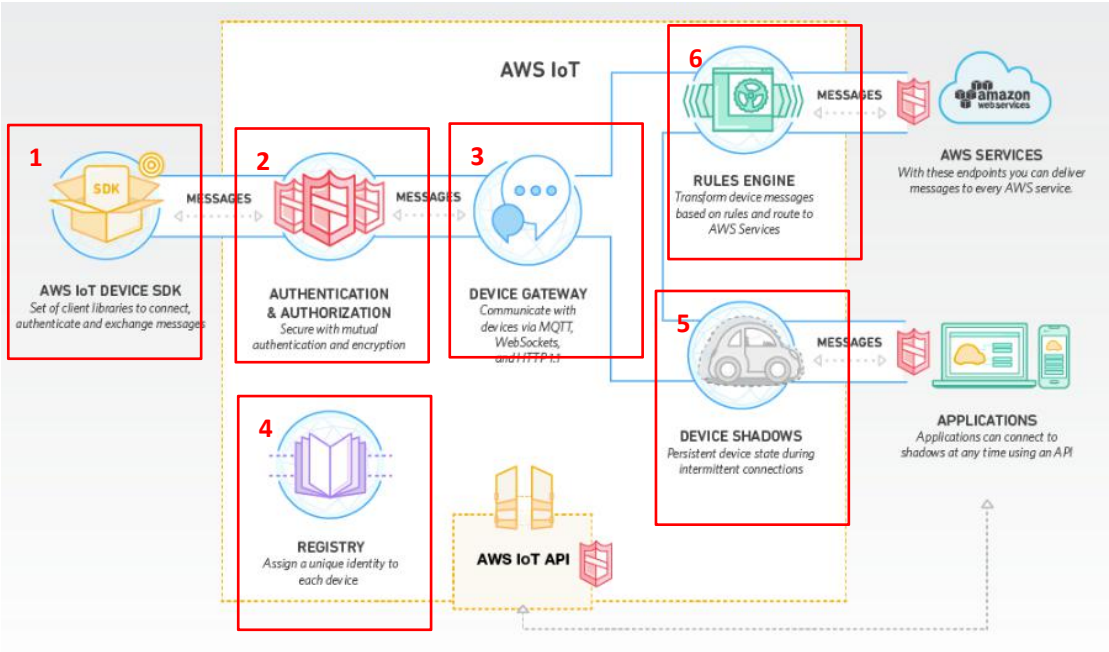
AWS IoT Platform

1. AWS IoT features

AWS IoT is a platform that enables us to connect devices to AWS Services and other devices, secure data and interactions, process and act upon device data, and enable applications to interact with devices even when they are offline.



1. 1 AWS IoT Device SDK

AWS IoT provides an SDK to help us easily and quickly connect wer hardware device or wer mobile application. The AWS IoT Device SDK enables our devices to connect, authenticate, and exchange messages with AWS IoT using the MQTT, HTTP, or WebSockets protocols.

*AWS SDK: C++, Java,.Net,Node.js,Python,PHP,Ruby,Go,ios/android
*Device SDK:embedded C, Java, Node.js,Python,Arduino Yun,ios/android

1.2 Authentication and Authorization

AWS IoT provides mutual authentication and encryption at all points of connection, so that data is never exchanged between devices and AWS IoT without proven identity. AWS IoT supports the AWS method of authentication (called 'SigV4') as well as X.509 certificate based authentication. Connections using HTTP can use either of these methods, while connections using MQTT use certificate based authentication, and connections using WebSockets can use SigV4. With AWS IoT we can use AWS IoT generated certificates, as well as those signed by our preferred Certificate Authority (CA). We can map wer choice of role and/or policies to each certificate, so that we can authorize devices or applications to have access, or change our mind and revoke access altogether without ever touching the device.

AWS IoT also supports connections from users' mobile apps using Amazon Cognito, which takes care of all the steps necessary to create a unique identifier for our app's users and retrieve

temporary, limited-privilege AWS credentials.

1.3 Device Gateway

The AWS IoT Device Gateway enables devices to securely and efficiently communicate with AWS IoT. The Device Gateway can exchange messages using a publication/subscription model, which enables one-to-one and one-to-many communications. With this one-to-many communication pattern AWS IoT makes it possible for a connected device to broadcast data to multiple subscribers for a given topic. **The Device Gateway supports MQTT, WebSockets, and HTTP 1.1 protocols.** The Device Gateway scales automatically to support over a billion devices without provisioning infrastructure.

1.4 Registry

The Registry establishes an identity for devices and tracks metadata such as the devices' attributes and capabilities. The Registry assigns a unique identity to each device that is consistently formatted regardless of the type of device or how it connects. It also supports metadata that describes the capabilities of a device, for example whether a sensor reports temperature, and if the data are Fahrenheit or Celsius.

The Registry lets us store metadata about our devices at no additional charge, and metadata in the Registry does not expire as long as we access or update wer registry entry at least once every 7 years.

1.5 Device Shadows

With AWS IoT we can create a persistent, virtual version, or "shadow," of each device that includes the device's latest state so that applications or other devices can read messages and interact with the device. The Device Shadows persist the last reported state and desired future state of each device even when the device is offline. we can retrieve the last reported state of a device or set a desired future state through the API or using the rules engine.

Device Shadows make it easier to build applications that interact with our devices by providing always available REST APIs. In addition, applications can set the desired future state of a device without accounting for the devices current state. AWS IoT will compare the difference between the desired and last reported state, and command the device to make up the difference.

The AWS IoT Device SDK makes it easy for our device to synchronize its state with its shadow, and to respond to desired future states set via the shadow.

Device Shadows let us store the state of wer devices for up to a year for free. Device Shadows persist forever if we update them at least once per year, otherwise they expire.

1.6 Rules Engine

The Rules Engine makes it possible to build IoT applications that gather, process, analyze and act on data generated by connected devices at global scale without having to manage any infrastructure. The Rules Engine evaluates inbound messages published into AWS IoT and

transforms and delivers them to another device or a cloud service, based on business rules we define. A rule can apply to data from one or many devices, and it can take one or many actions in parallel.

The Rules Engine can also route messages to AWS endpoints including **AWS Lambda, Amazon Kinesis, Amazon S3, Amazon Machine Learning, Amazon DynamoDB, Amazon CloudWatch, and Amazon Elasticsearch Service with built-in Kibana integration.** External endpoints can be reached using AWS Lambda, Amazon Kinesis, and Amazon Simple Notification Service (SNS).

we can author rules within the management console or write rules using a SQL-like syntax. Rules can be authored to behave differently depending upon the content of the message.
The Rules Engine provides dozens of available functions that can be used to transform our data, and it's possible to create infinitely more via AWS Lambda. Rules can also trigger the execution of our Java, Node.js or Python code in AWS Lambda, giving us maximum flexibility and power to process device data.

2. AWS IoT Pricing

With AWS IoT, you pay for only what you use and there are no minimum fees. Prices are based on the number of messages published to AWS IoT (Publishing Cost), and the number of messages delivered by AWS IoT to devices or applications (Delivery Cost).

AWS IoT does not charge for deliveries to the following AWS services:
Amazon S3
Amazon DynamoDB
AWS Lambda
Amazon Kinesis
Amazon SNS
Amazon SQS.

The AWS IoT free tier gets you started with 250,000 free messages (published or delivered) per month, for 12 months.
https://amazonaws-china.com/free/?nc2=h_l2_cc

MQTT Message Metering

MQTT Connect	Metered as the size of the message including "Will" topic size and "Will" message payload
MQTT PubAck (received from device)	Metered as a single message of 512 bytes
MQTT Ping	Includes both Ping request and Ping response. Both together are metered as a single message of 512 bytes
MQTT Subscribe	Metered on the size of the topic list submitted in the Subscribe message
MQTT Publish (inbound)	Metered on the size of the payload and topic in bytes
MQTT Publish (outbound)	Metered on the size of the payload and topic in bytes

The following MQTT messages are excluded from metering:

MQTT Disconnect

MQTT ConnAck

MQTT PubAck (sent by service)

MQTT SubAck

MQTT Unsubscribe

MQTT messages transferred inside a WebSocket connection are metered like MQTT messages inside a mutually authenticated TLS connection.

References:

https://amazonaws-china.com/iot-platform/how-it-works/?nc1=h_ls

https://amazonaws-china.com/cn/about-aws/events/webinar/iot-solutions-based-on-aws-iot-platform06062017/

https://amazonaws-china.com/iot-platform/pricing/?nc1=h_ls