Microsoft Azure

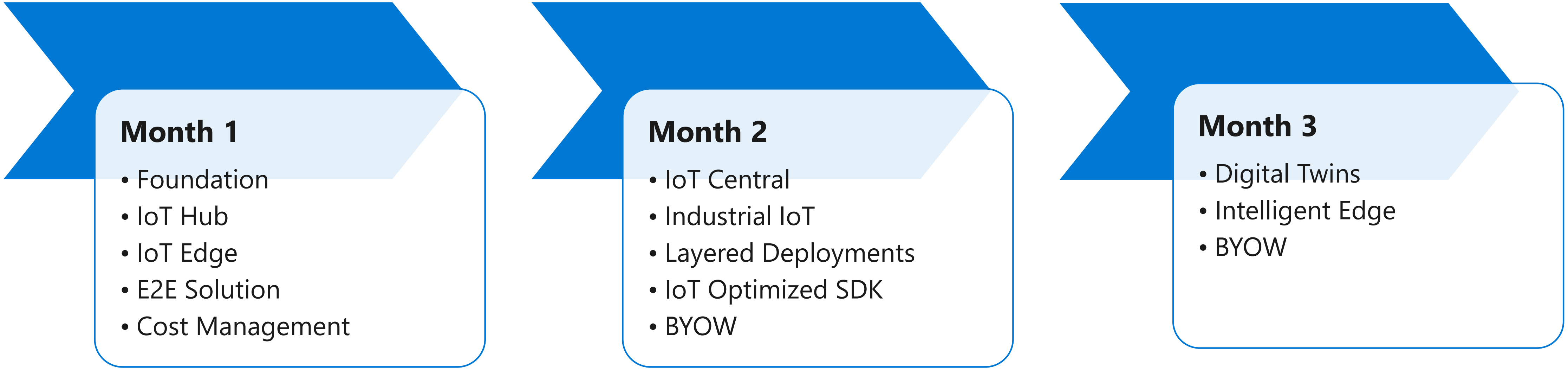
Azure IoT Academy

Transforming your business

Marianela Ramsdell
Sr. Technical Specialist - Global Black Belt
Microsoft

Manisha Kumari
Sr. Technical Specialist – Global Black Belt
Microsoft

IoT Academy Journey



Month 1

- Foundation
- IoT Hub
- IoT Edge
- E2E Solution
- Cost Management

Month 2

- IoT Central
- Industrial IoT
- Layered Deployments
- IoT Optimized SDK
- BYOW

Month 3

- Digital Twins
- Intelligent Edge
- BYOW

Agenda (All timings are in EST)

- 11:00am- 1:00pm EST : HOLs
- 1:00pm - 1:45pm EST : Lunch Break
- 1:45pm - 3:15pm EST : HOLs
- 3:15pm - 3:30pm EST : Coffee Break
- 3:30pm - 5:00pm EST : HOLs

Lesson 01 – Defining IoT

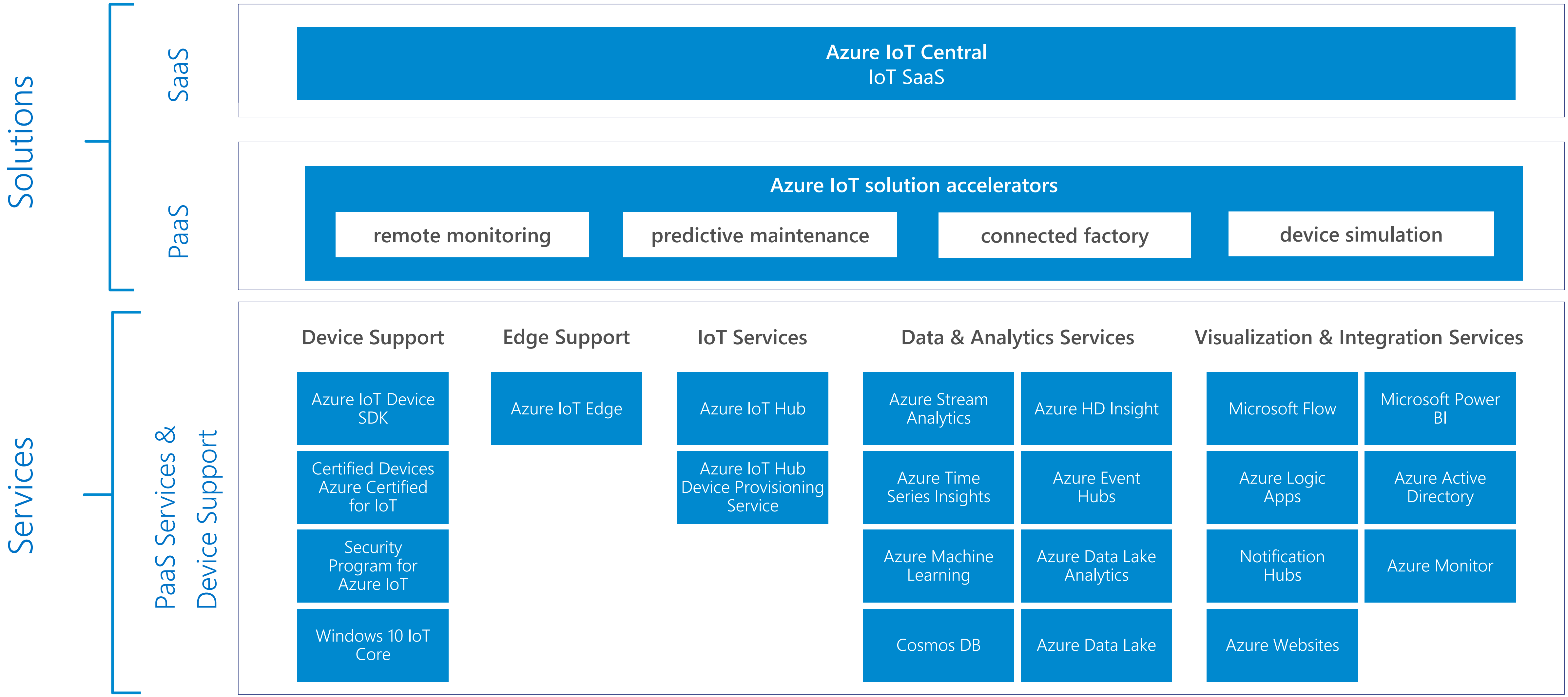
Defining IoT

Things

Insights

Action

Comprehensive Set of Capabilities for IoT Solutions



Solutions

SaaS

Azure IoT Central
IoT SaaS

PaaS

Azure IoT solution accelerators

- remote monitoring
- predictive maintenance
- connected factory
- device simulation

Services

PaaS Services & Device Support

Device Support

- Azure IoT Device SDK
- Certified Devices Azure Certified for IoT
- Security Program for Azure IoT
- Windows 10 IoT Core

Edge Support

- Azure IoT Edge

IoT Services

- Azure IoT Hub
- Azure IoT Hub Device Provisioning Service

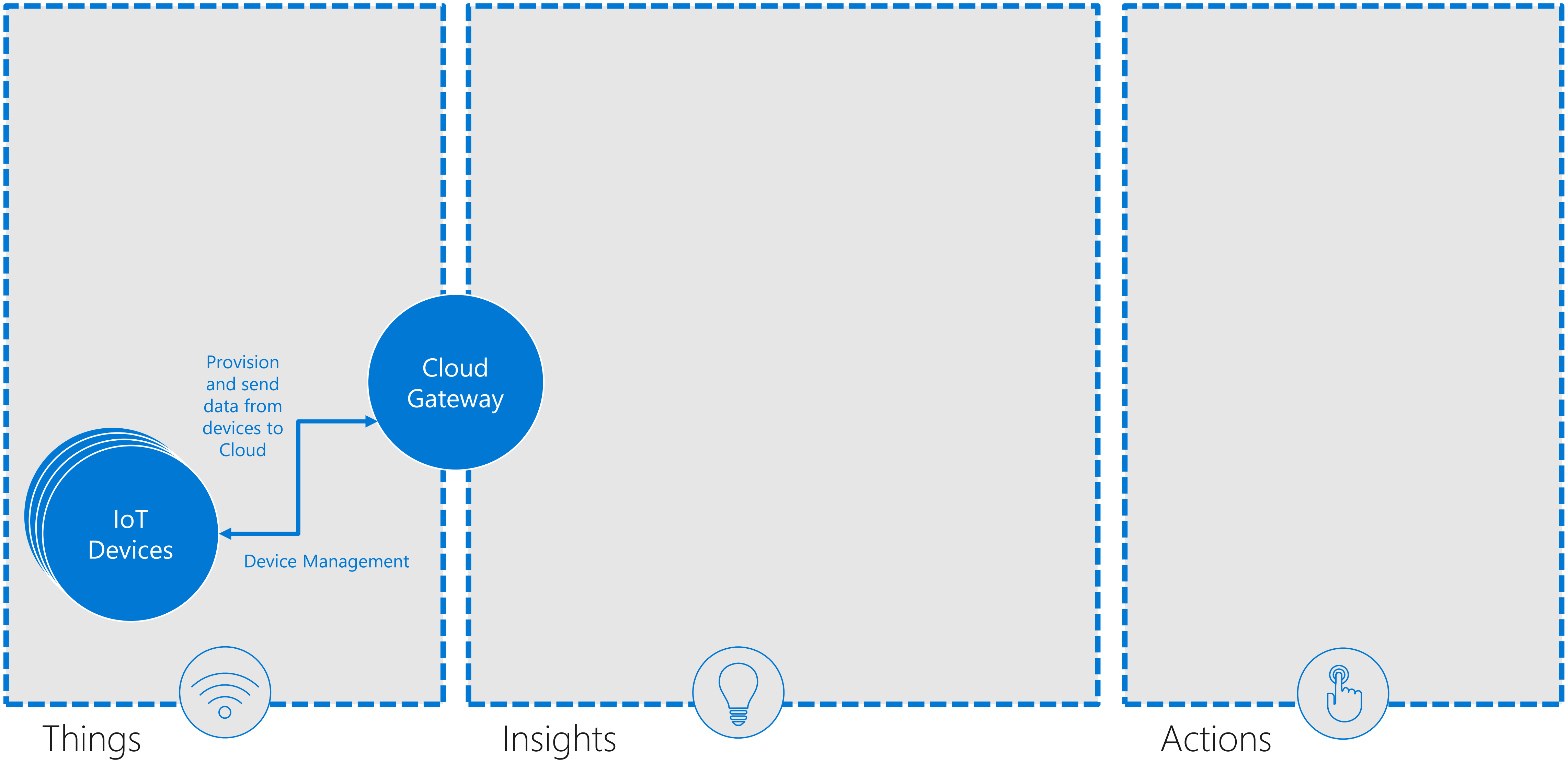
Data & Analytics Services

- Azure Stream Analytics
- Azure HD Insight
- Azure Time Series Insights
- Azure Event Hubs
- Azure Machine Learning
- Azure Data Lake Analytics
- Cosmos DB
- Azure Data Lake

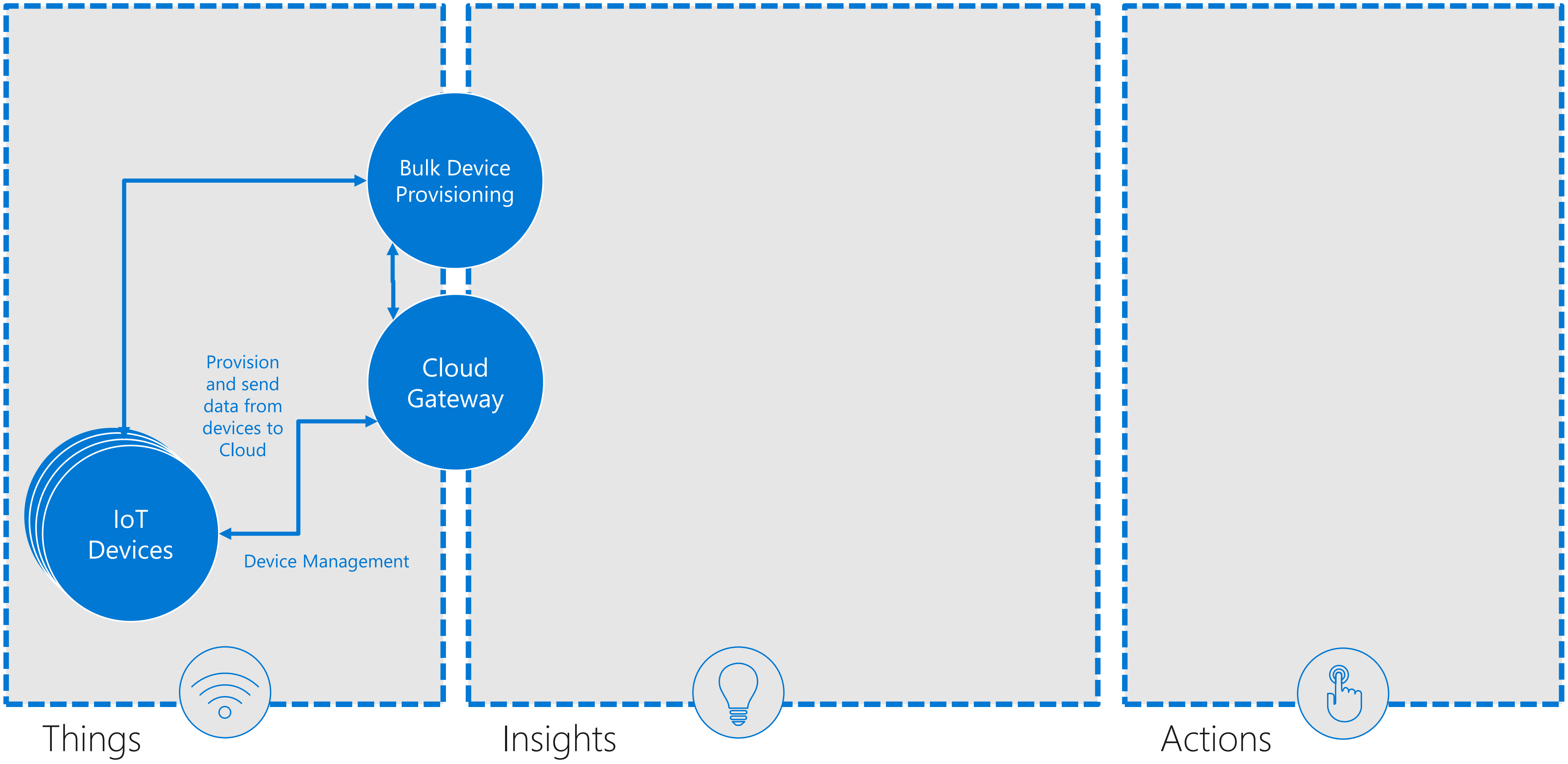
Visualization & Integration Services

- Microsoft Flow
- Microsoft Power BI
- Azure Logic Apps
- Azure Active Directory
- Notification Hubs
- Azure Monitor
- Azure Websites

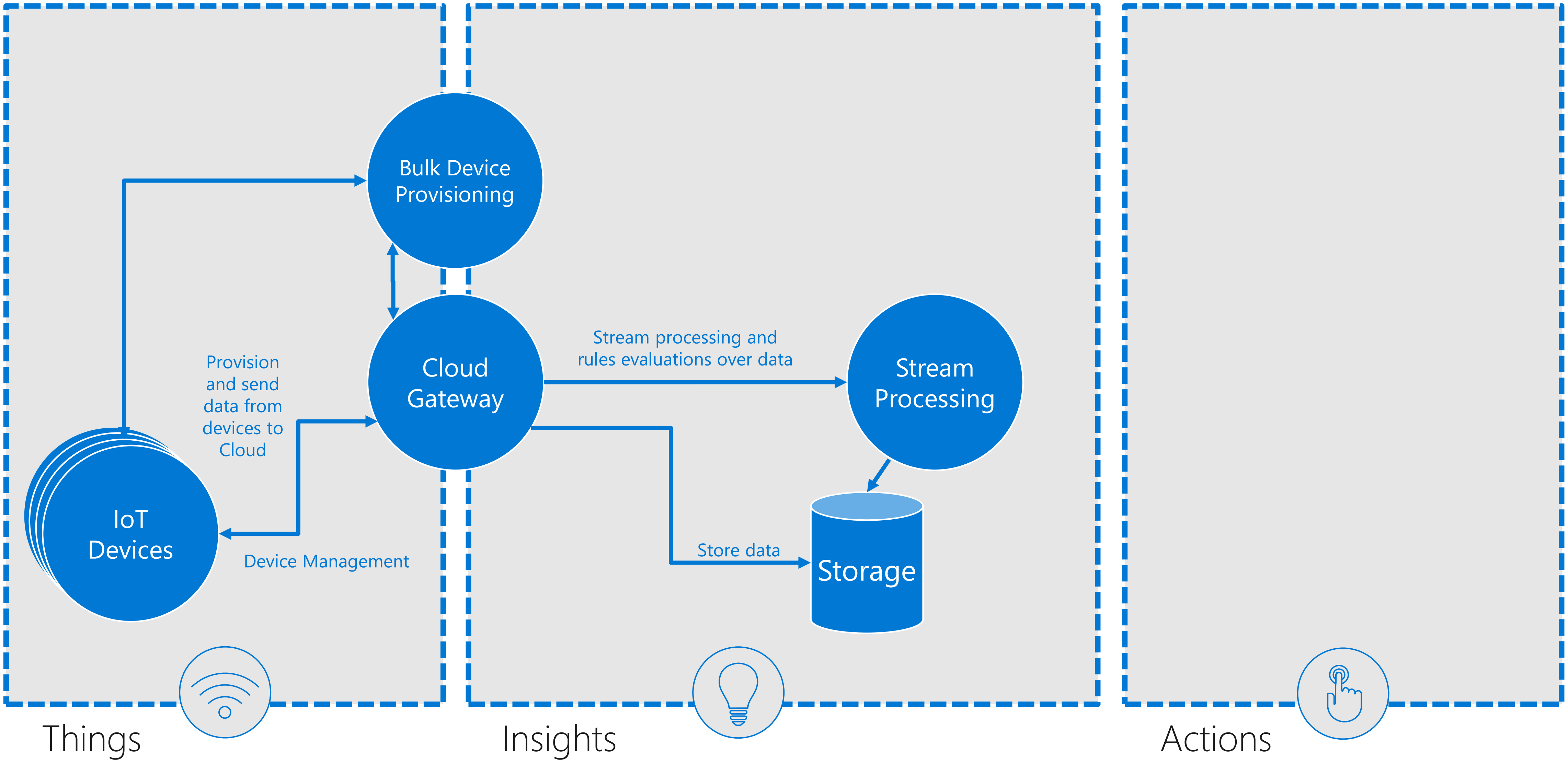
High-level IoT Architecture



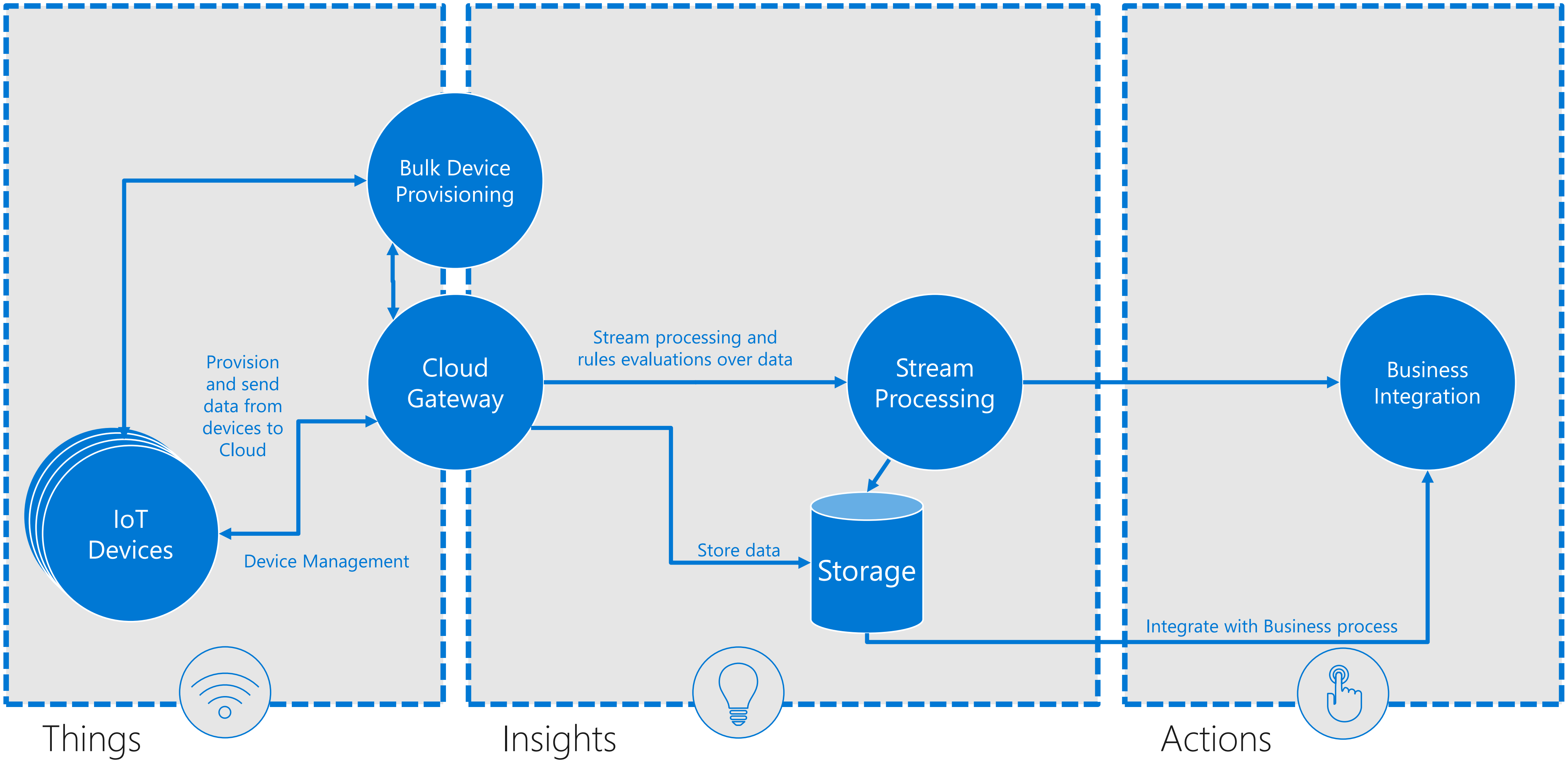
High-level IoT Architecture



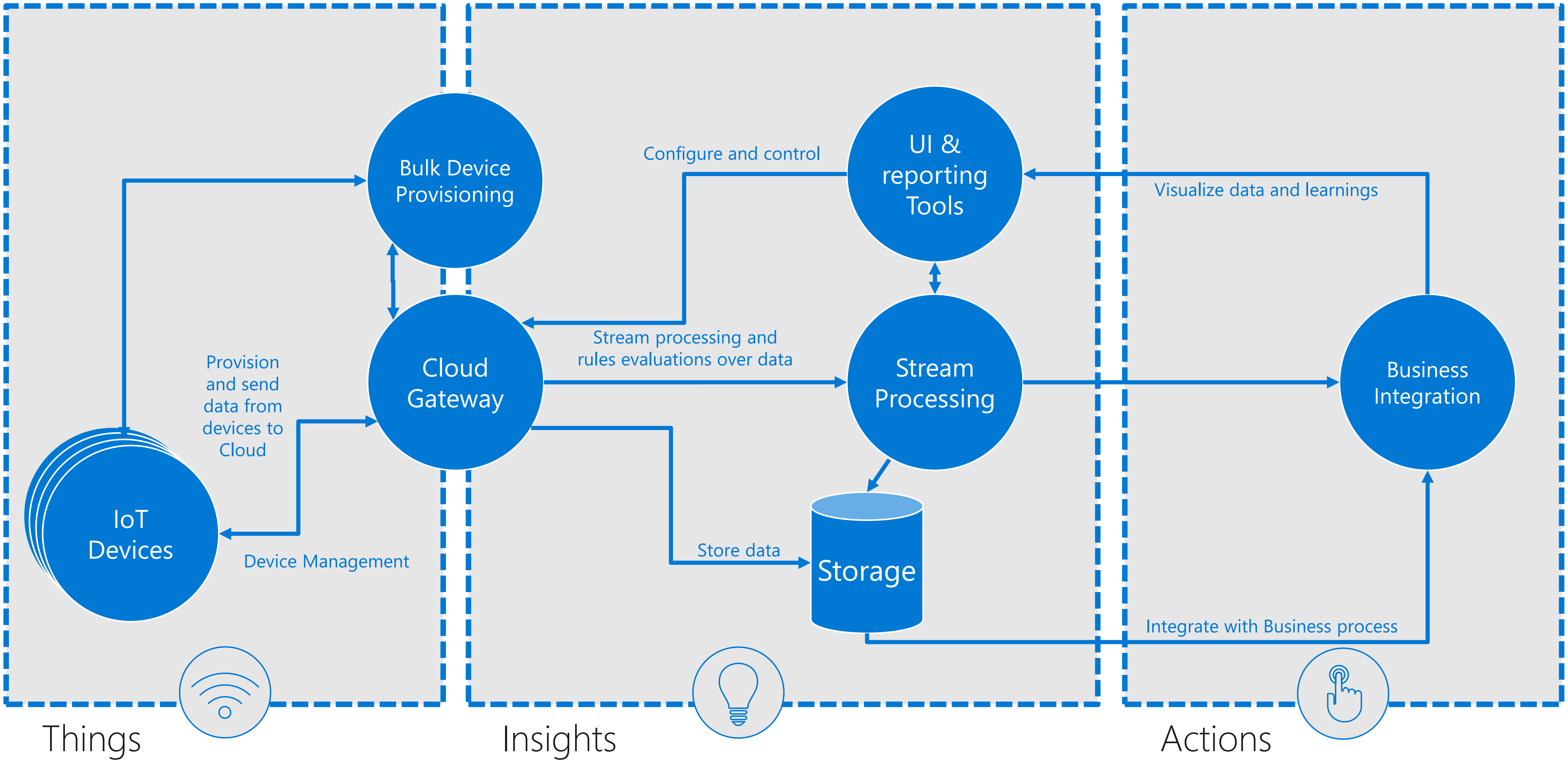
High-level IoT Architecture



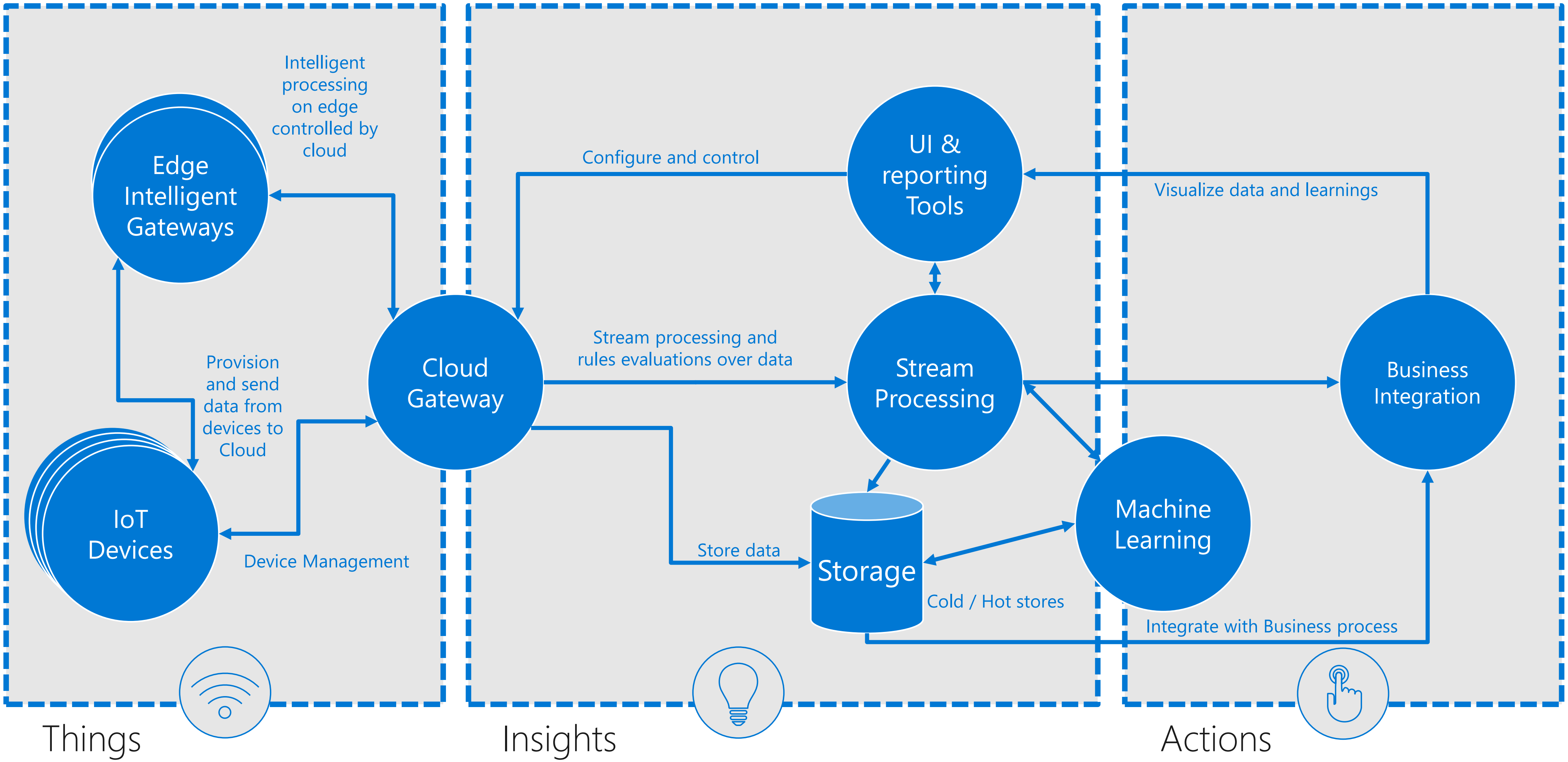
High-level IoT Architecture



High-level IoT Architecture



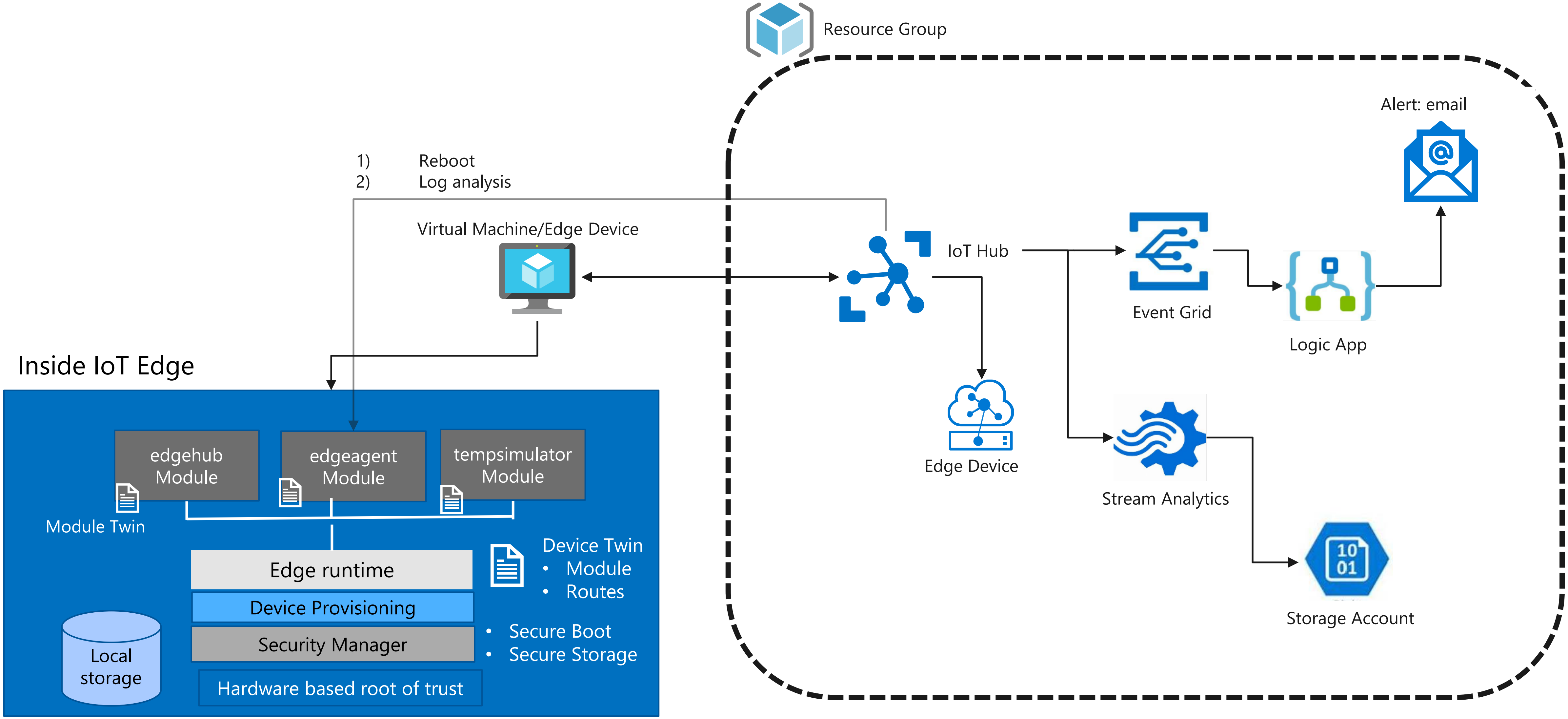
High-level IoT Architecture



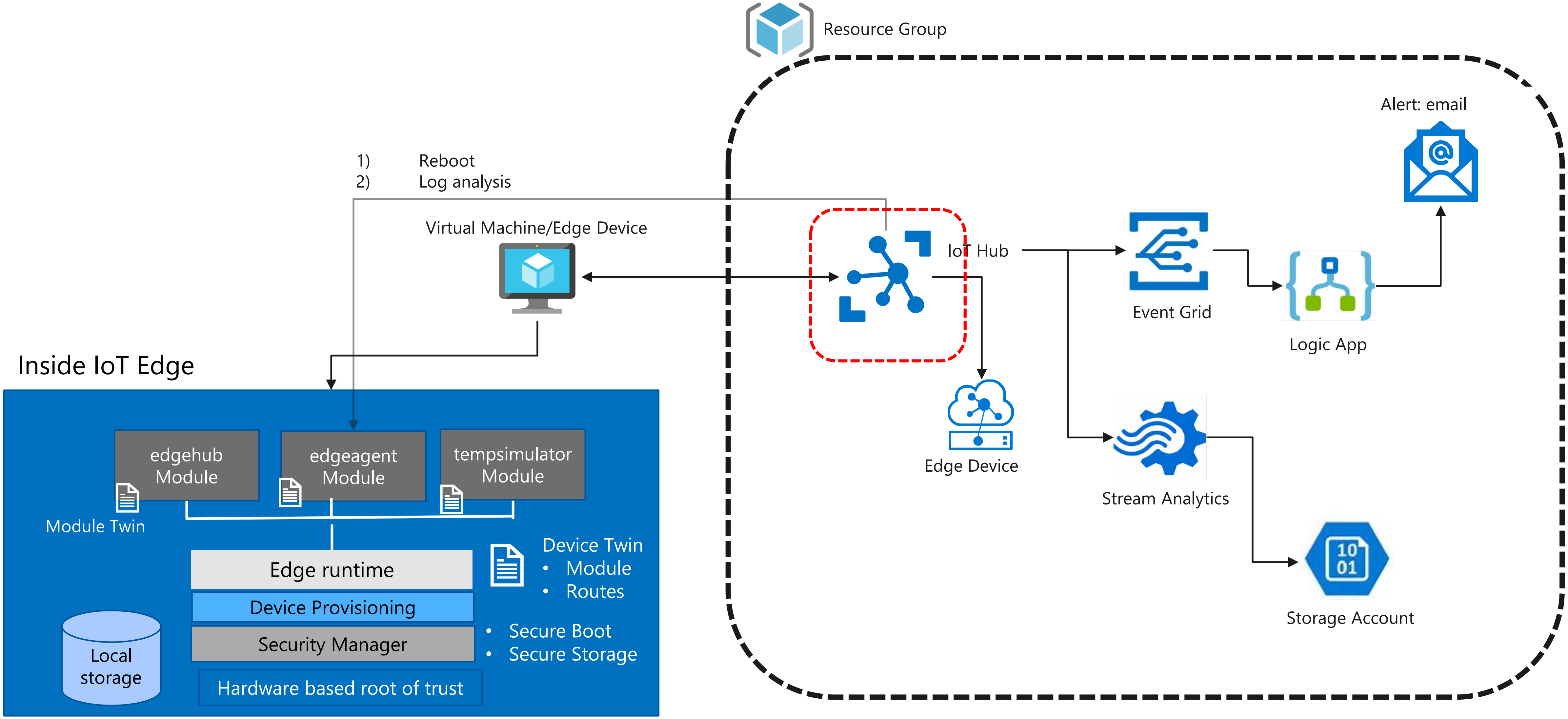
Intelligent processing on edge controlled by cloud
Edge Intelligent Gateways
Configure and control
UI & reporting Tools
Visualize data and learnings
Provision and send data from devices to Cloud
Cloud Gateway
Stream processing and rules evaluations over data
Stream Processing
Business Integration
IoT Devices
Device Management
Store data
Storage
Cold / Hot stores
Machine Learning
Integrate with Business process
Things
Insights
Actions

Hands-On Lab

Today's E2E Architecture



Exercise #1


Resource Group
1) Reboot
2) Log analysis
Virtual Machine/Edge Device
IoT Hub
Alert: email
Event Grid
Logic App
Edge Device
Stream Analytics
Storage Account
Inside IoT Edge
edgehub Module
edgeagent Module
tempsimulator Module
Module Twin
Edge runtime
Device Twin
• Module
• Routes
Device Provisioning
Security Manager
• Secure Boot
• Secure Storage
Local storage
Hardware based root of trust

IoT Developer Tools Overview

- Software Development Kits (SDKs)
- Visual Studio
- Visual Studio Code
- Command-Line Interfaces (CLIs)

Exercise #1

IoT Hub Create Resource Options

- MS Portal
- CLI
- Tools: VS Code

Features of Azure IoT Hub

Azure IoT Hub provides feature support in the following areas:

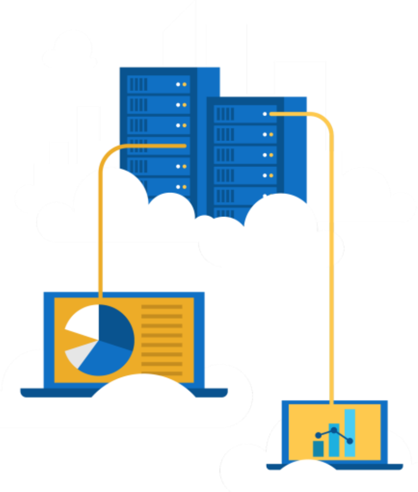
- Security
- Scalability
- Routing
- Service Integration
- Device Management
- Monitoring

Recap

IoT Hub Create Resource Options

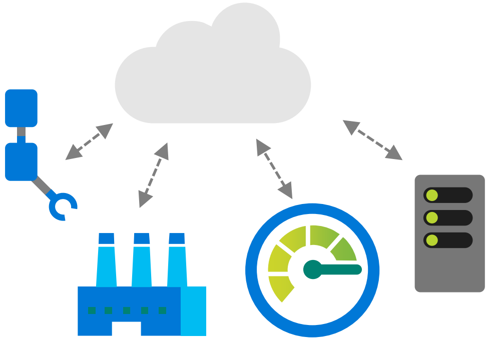
- MS Portal
- CLI
- Tools: VS Code

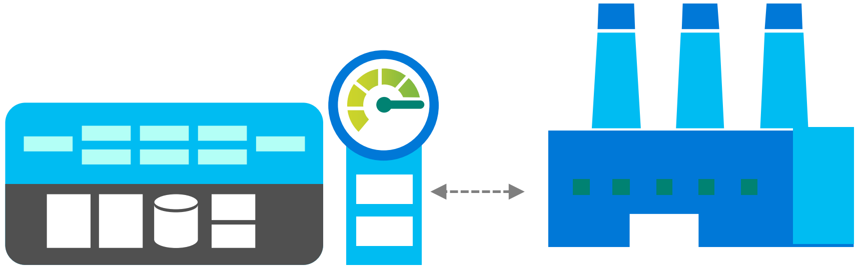
Lesson 02 - Devices



IoT Edge Hardware - Samples

IoT in the Cloud and on the Edge





IoT in the Cloud

Remote monitoring and management

Merging remote data from multiple IoT devices

Infinite compute and storage to train machine learning and other advanced AI tools

IoT on the Edge

Low latency tight control loops require near real-time response

Protocol translation & data normalization

Privacy of data and protection of IP

Symmetry

Enabling the Intelligent Edge Spectrum

Sensor Tier

Constrained Tier

Interactive Tier

Industrial Tier

Gateway Tier

GPU/DSP

FPGA

Accelerated Tier

LOW POWER CAPABILITIES

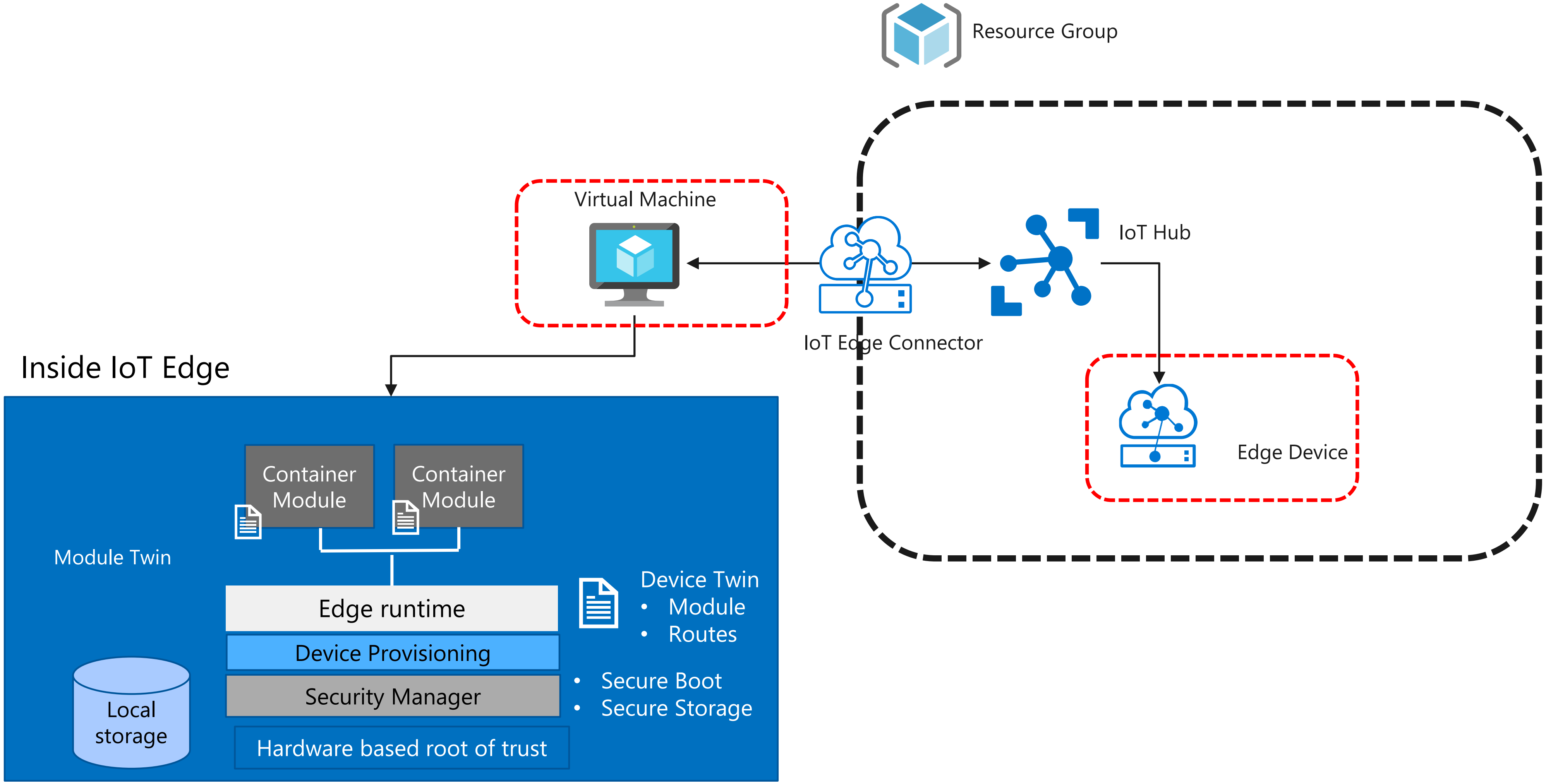
HIGH POWER CAPABILITIES

Edge hardware requirements

- Rich OS – Windows or Linux
- Flexible HW – ARM or x64
- Moby-compatible container runtime
- Hardware based security – HSM or Enclave
- Hardware sizing depends on workload

Hands-On Lab

Exercise #2

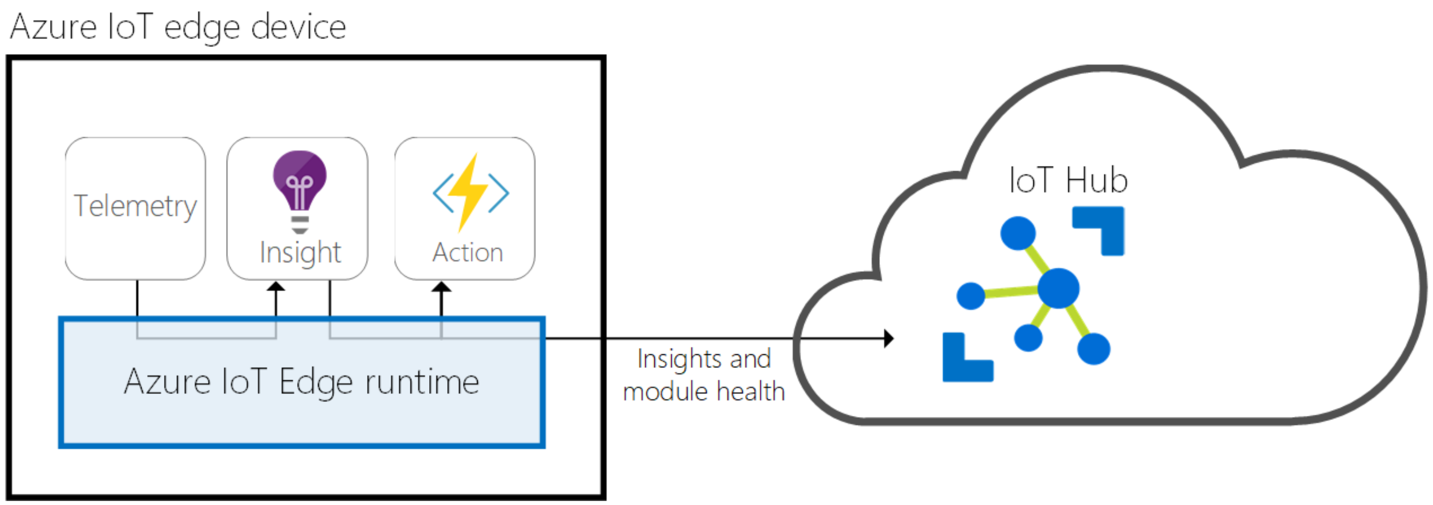

Resource Group
Virtual Machine
IoT Edge Connector
IoT Hub
Edge Device
Inside IoT Edge
Container Module
Container Module
Module Twin
Edge runtime
Device Provisioning
Security Manager
Hardware based root of trust
Local storage
Device Twin
• Module
• Routes
• Secure Boot
• Secure Storage

Exercise #2

Create and Edge Device

- Create an Edge Device
- Set up the Edge Runtime
- Connect your Edge Device to IoT Hub

Concept – Azure IoT Edge Runtime

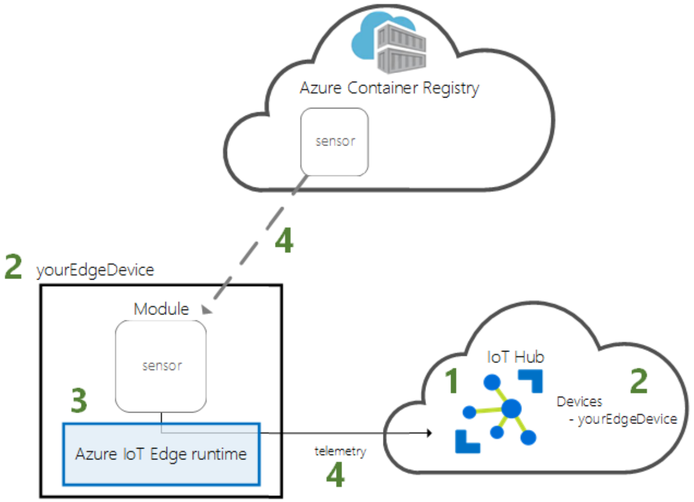


- Installs and updates workloads on the device.
- Maintains Azure IoT Edge security standards on the device.
- Ensures that IoT Edge modules are always running.
- Reports module health to the cloud for remote monitoring.
- Facilitates communication between downstream leaf devices and the IoT Edge device.
- Facilitates communication between modules on the IoT Edge device.
- Facilitates communication between the IoT Edge device and the cloud

System Modules

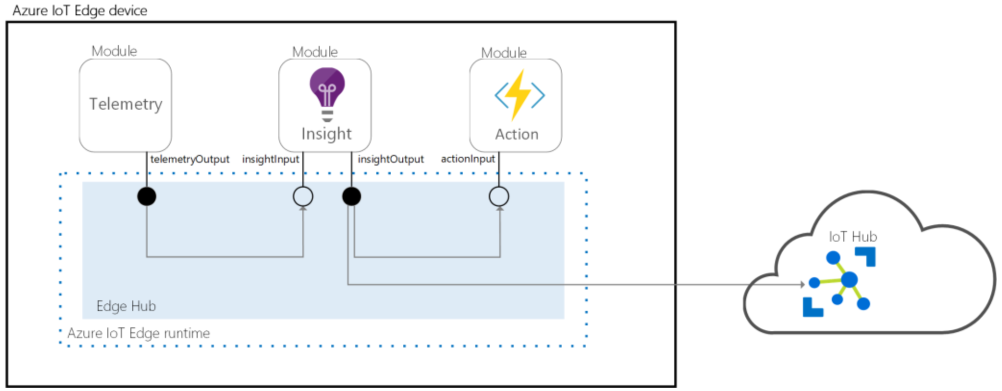
edge-agent:

Deployment & Container orchestration
Ensures module uptime

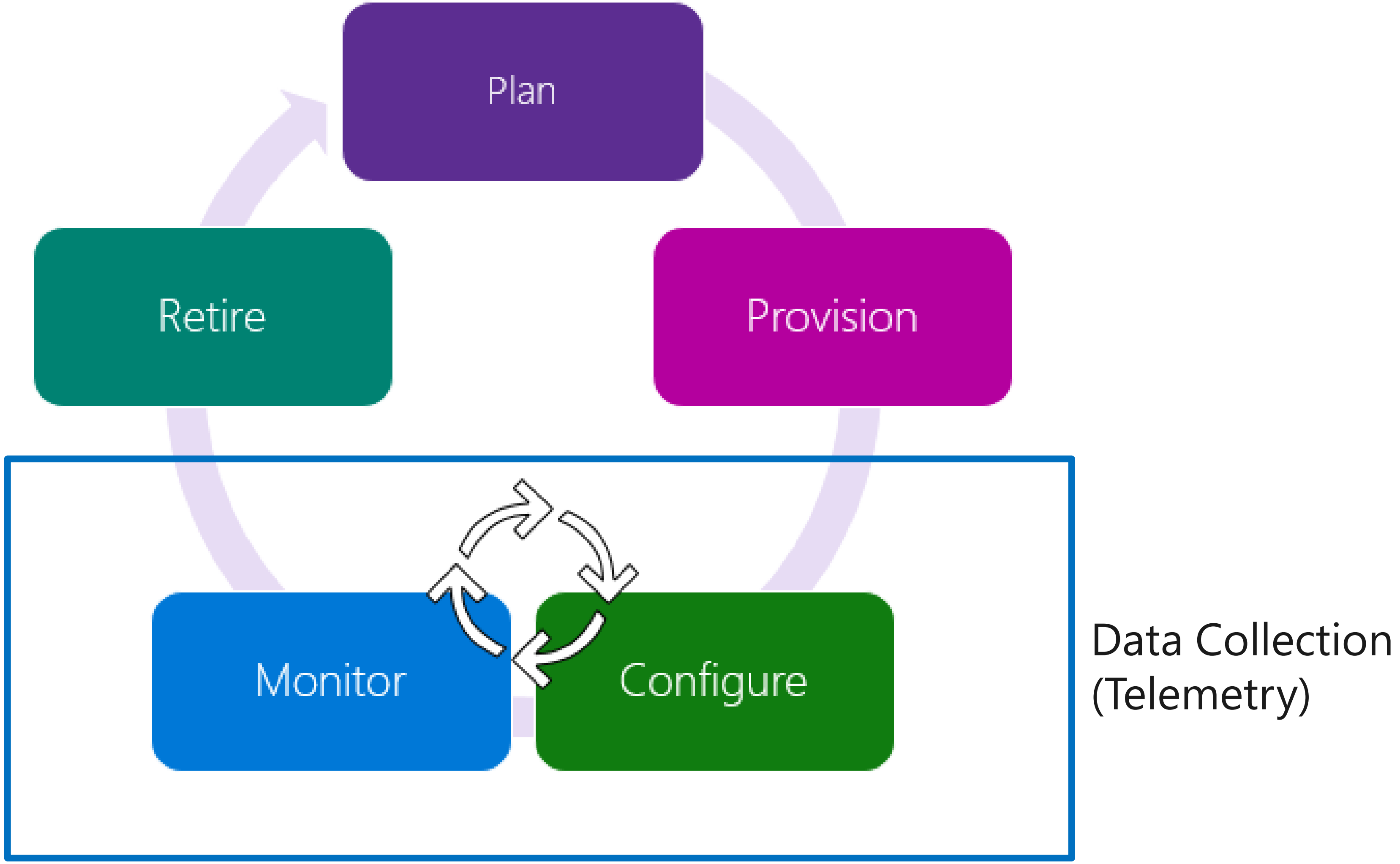


edge-hub:

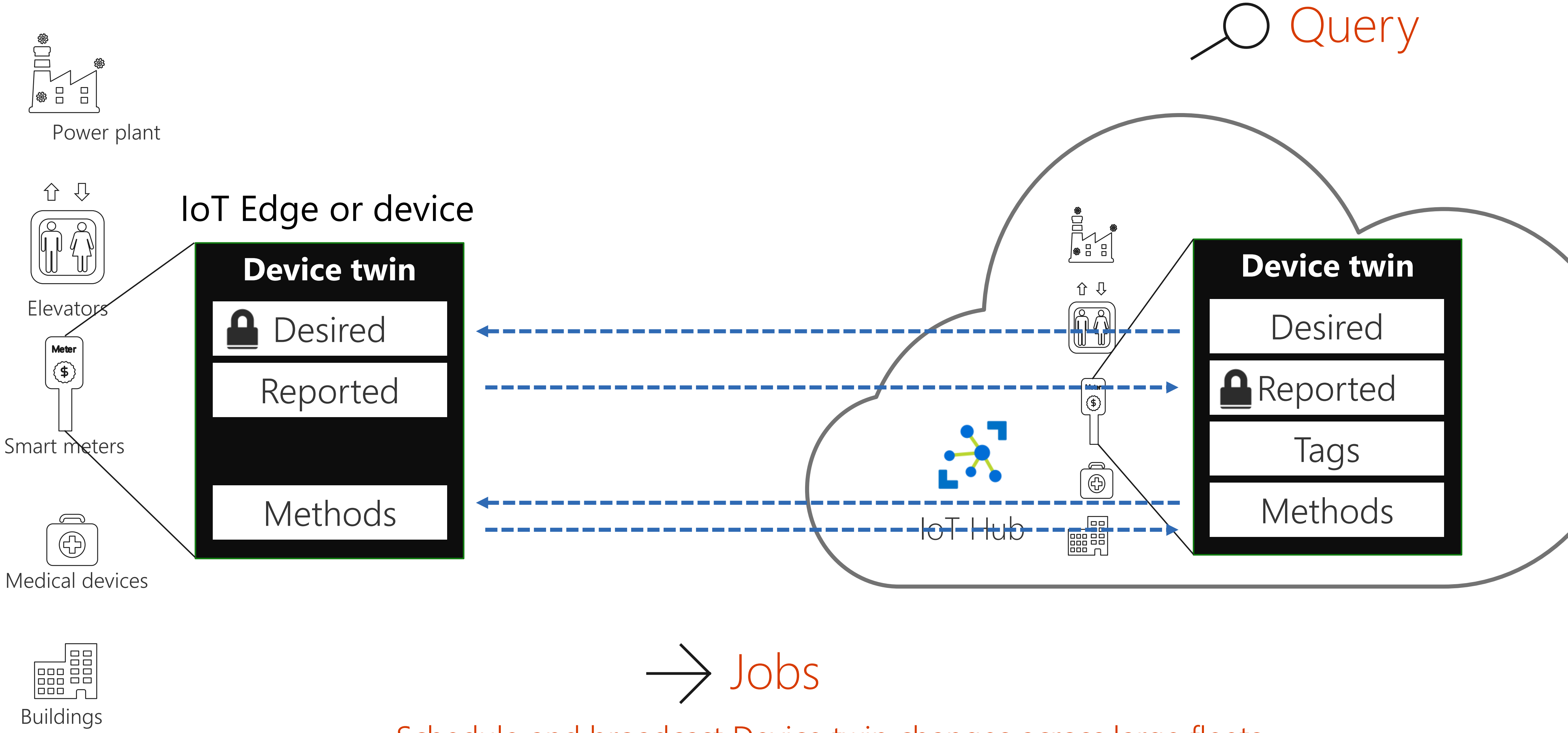
Communication to/from Azure IoT Hub
Inter-module communication



Device Lifecycle Terms and Concepts



Concept – Device Management



Query

Power plant

Elevators

Smart meters

Medical devices

Buildings

IoT Edge or device

Device twin

🔒 Desired

Reported

Methods

IoT Hub

Device twin

Desired

🔒 Reported

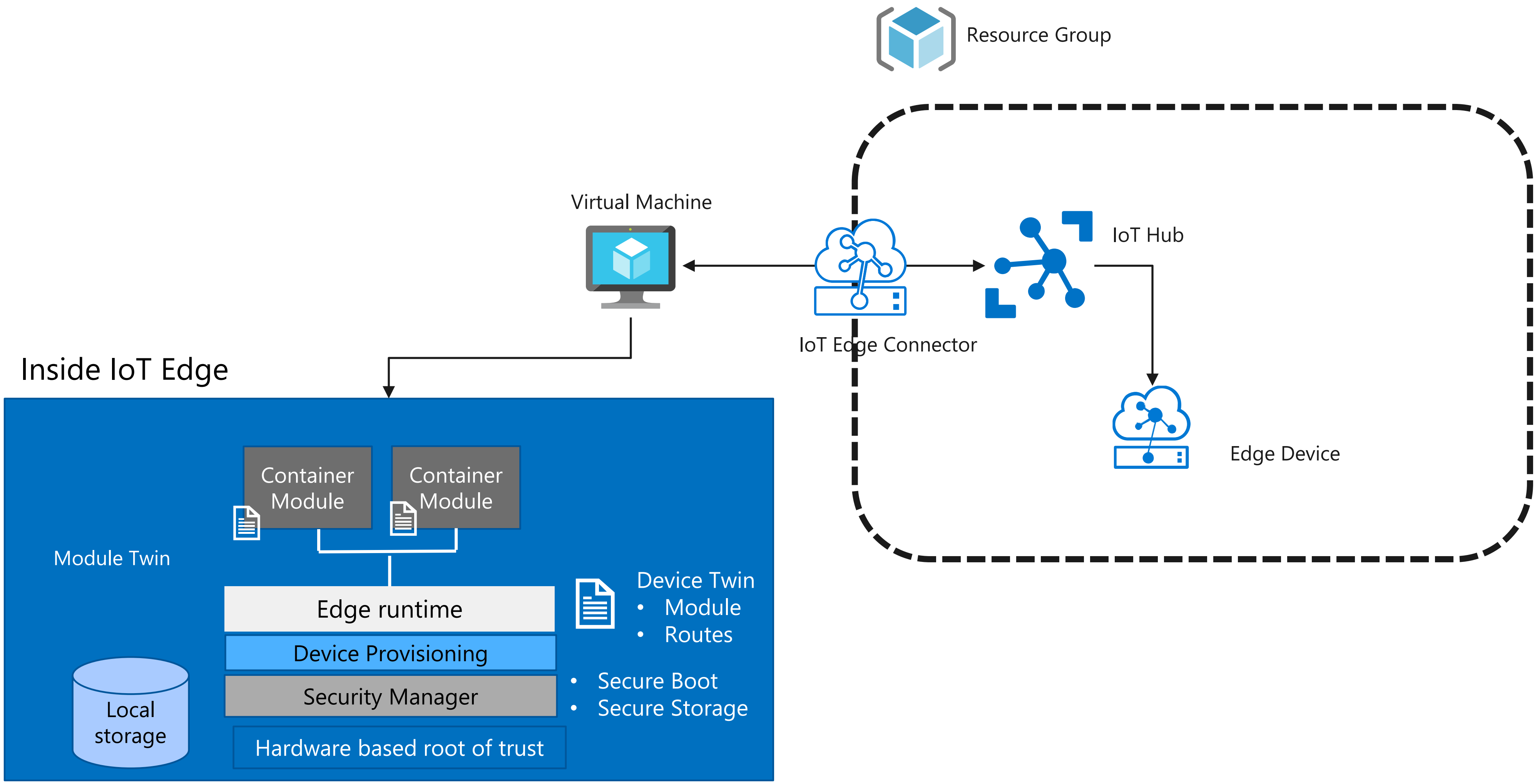
Tags

Methods

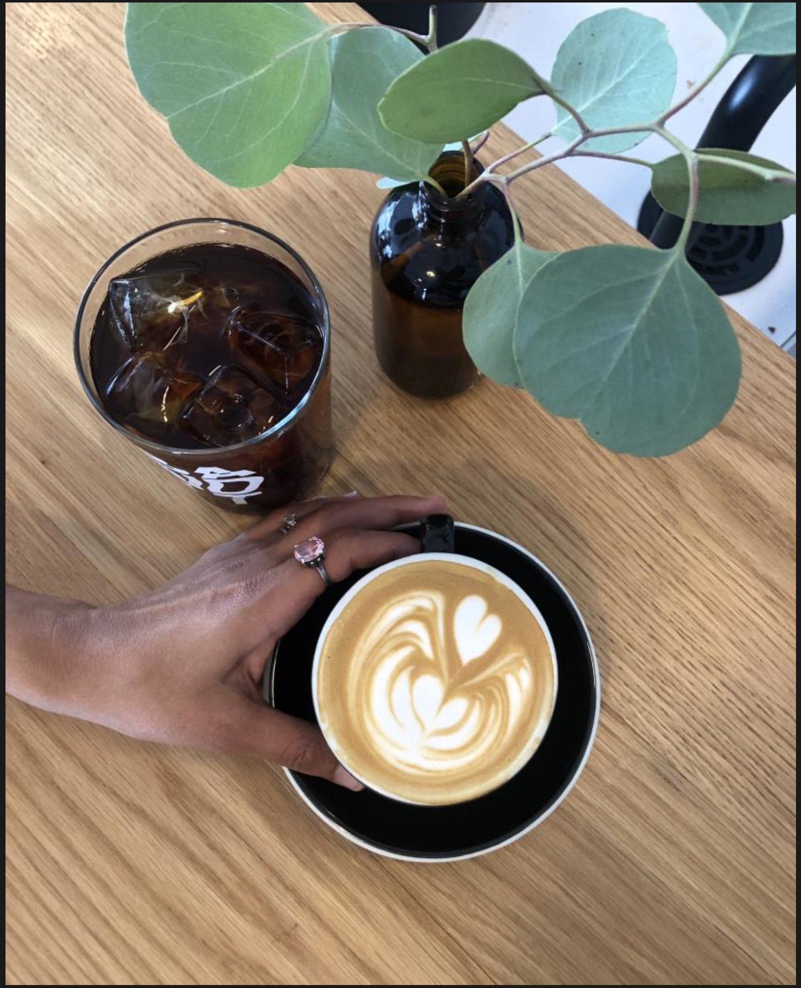
Jobs

Schedule and broadcast Device twin changes across large fleets

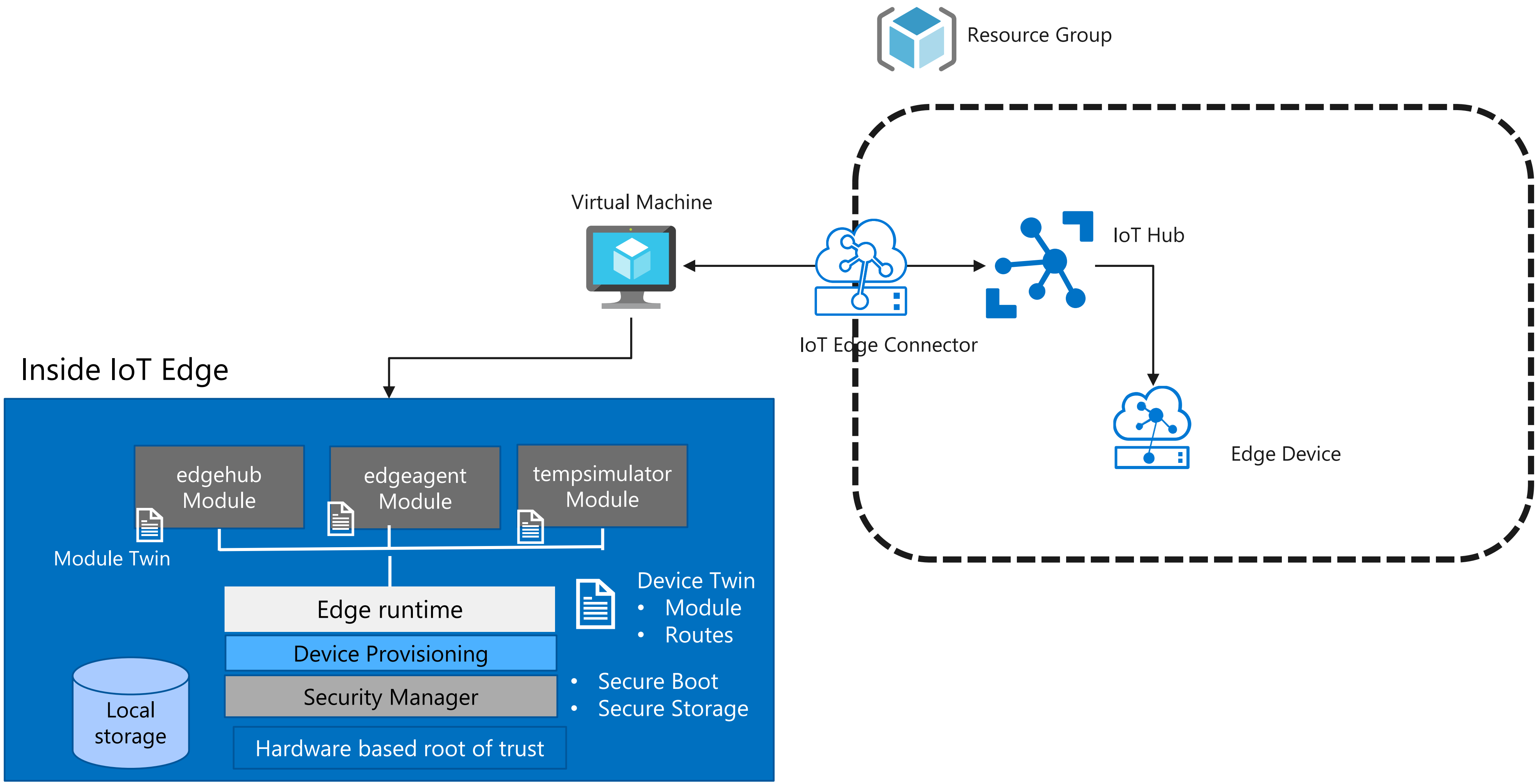
Recap - Exercise #2



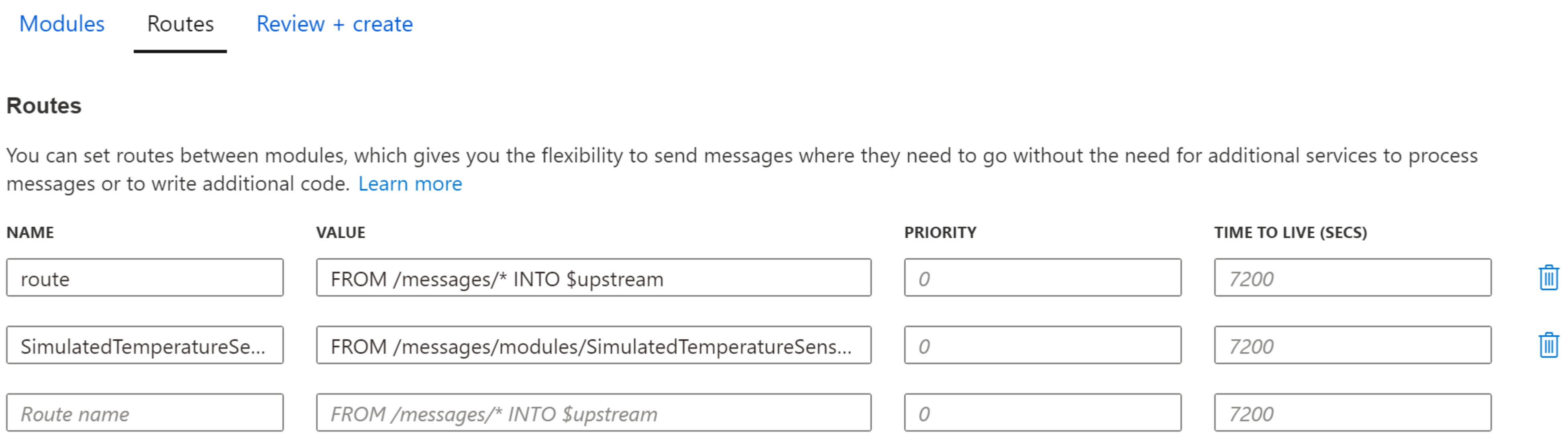
Break Time!



Exercise #3 – Deploy Temperature Simulator



Routes

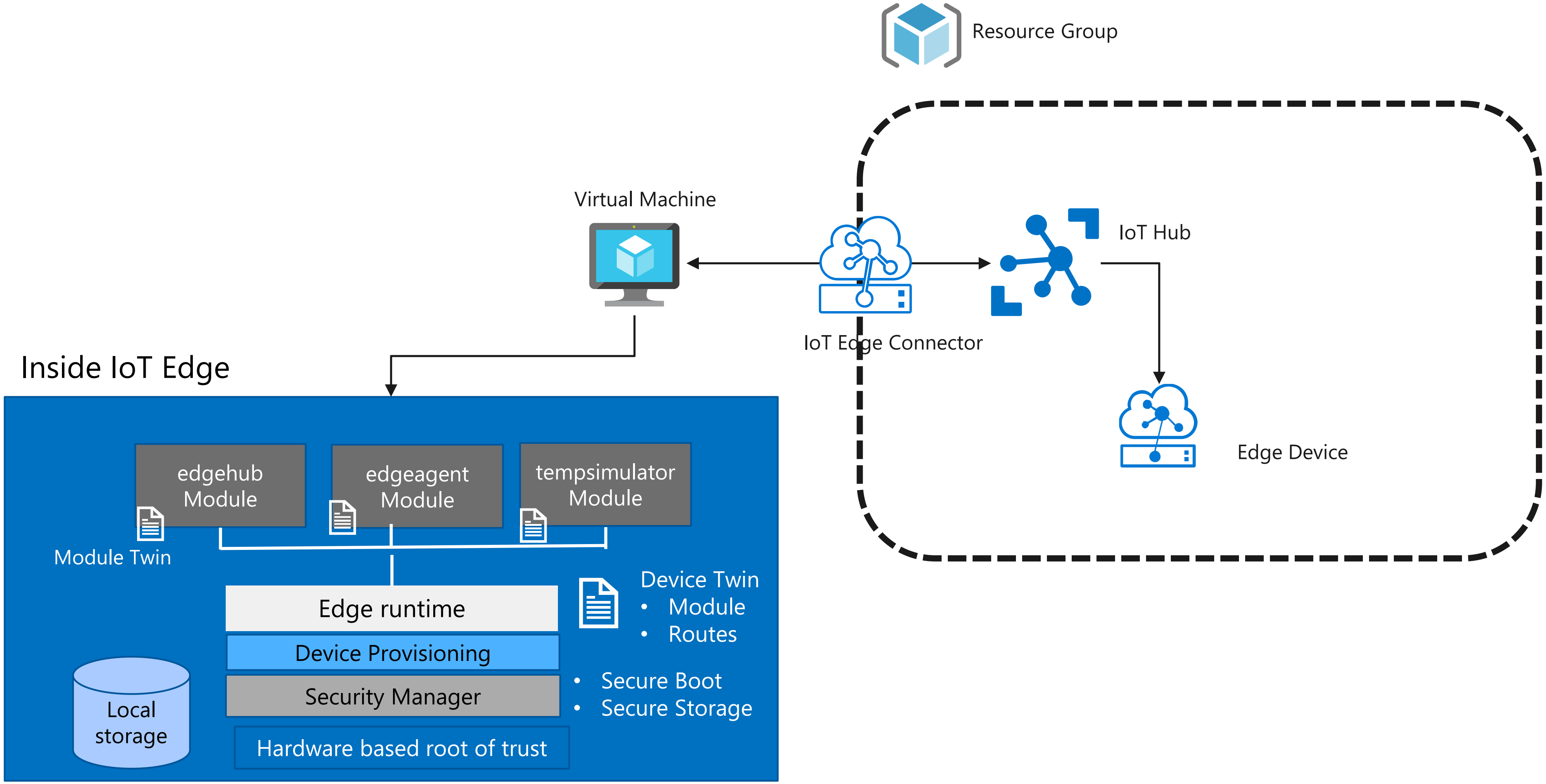


Routes

You can set routes between modules, which gives you the flexibility to send messages where they need to go without the need for additional services to process messages or to write additional code. Learn more

NAME	VALUE	PRIORITY	TIME TO LIVE (SECS)
route	FROM /messages/* INTO $upstream	0	7200
SimulatedTemperatureSe...	FROM /messages/modules/SimulatedTemperatureSens...	0	7200
Route name	FROM /messages/* INTO $upstream	0	7200

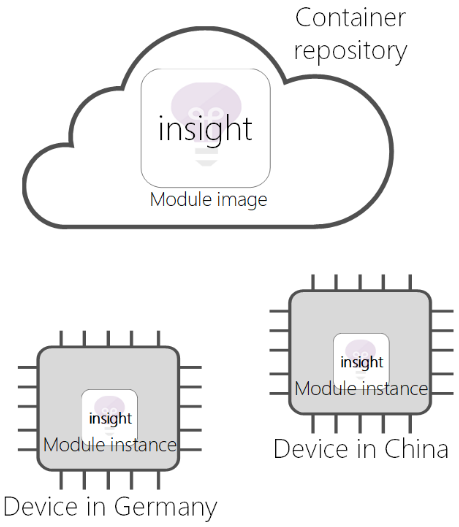
Recap Exercise #3 – Deploy Temperature Simulator

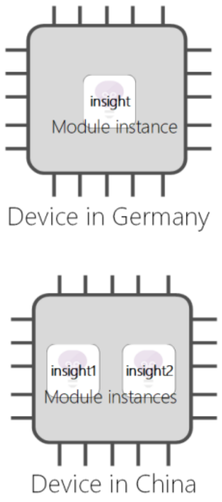


Concept – Module

Module image
lives in the cloud

Module instances
run on-premises





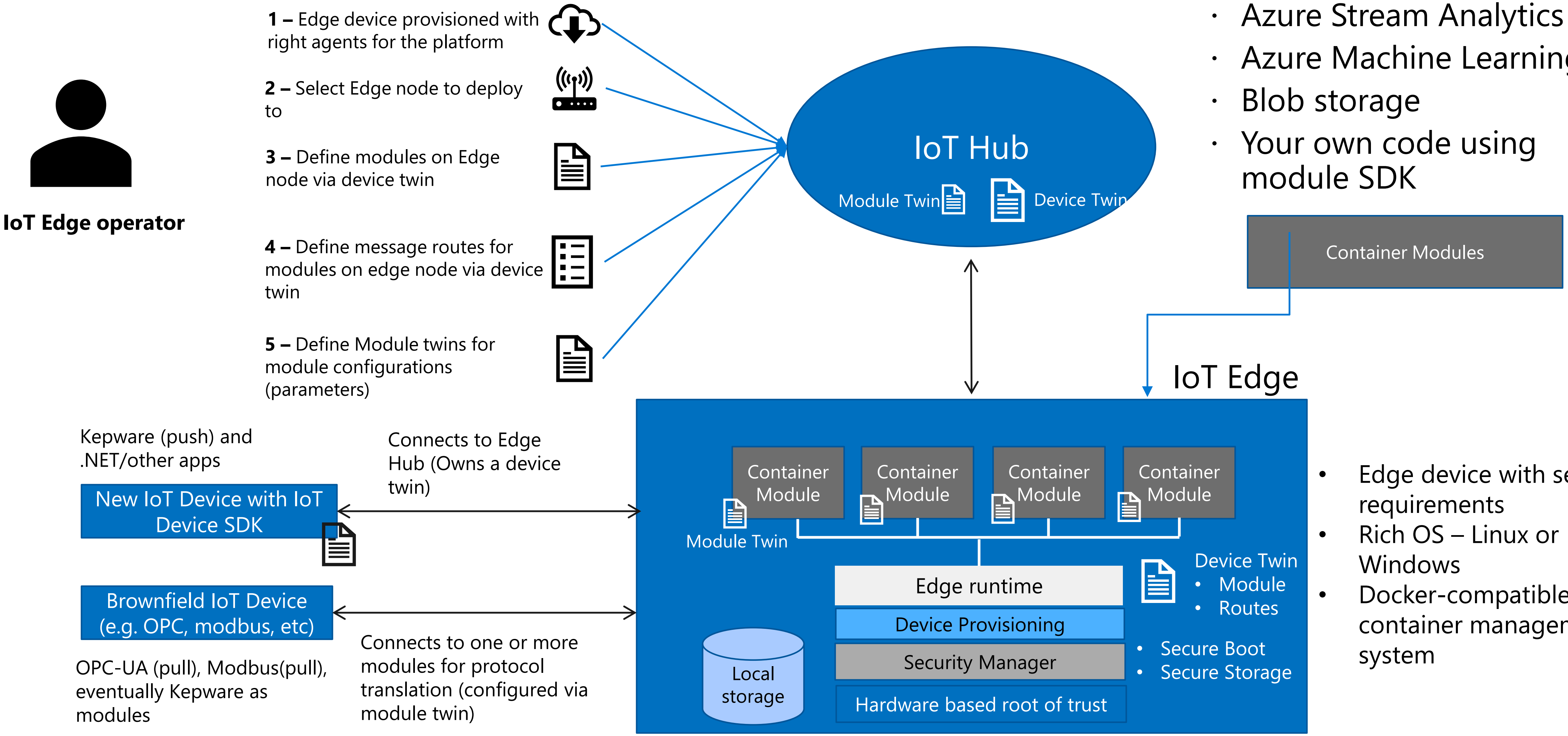
Device	/devices/Hannover01
Modules	/devices/Hannover01/modules/insight

Device	/devices/Shenzhen01
Modules	/devices/Shenzhen01/modules/insight1 /devices/Shenzhen01/modules/insight2

- A **module image** is a package containing the software that defines a module.
- A **module instance** is the specific unit of computation running the module image on an IoT Edge device. The module instance is started by the IoT Edge runtime.
- A **module identity** is a piece of information (including security credentials) stored in IoT Hub, that is associated to each module instance.
- A **module twin** is a JSON document stored in IoT Hub, that contains state information for a module instance, including metadata, configurations, and conditions.
- SDKs to develop custom modules in multiple languages (C#, C, Python, Java, Node.JS)

IoT Edge in action

- Container based workloads
- Cognitive Services
- Azure Functions
- Azure Stream Analytics
- Azure Machine Learning
- Blob storage
- Your own code using module SDK



IoT Edge operator

1 – Edge device provisioned with right agents for the platform

2 – Select Edge node to deploy to

3 – Define modules on Edge node via device twin

4 – Define message routes for modules on edge node via device twin

5 – Define Module twins for module configurations (parameters)

Device Twin
- Module
- Routes

- Secure Boot
- Secure Storage

Kepware (push) and .NET/other apps

Connects to Edge Hub (Owns a device twin)

OPC-UA (pull), Modbus(pull), eventually Kepware as modules

Connects to one or more modules for protocol translation (configured via module twin)

- Edge device with security requirements
- Rich OS – Linux or Windows
- Docker-compatible container management system

Azure IoT Edge - Packages Services in Containers

Azure Stream Analytics

In Line experience in the ASA web portal

Azure Functions

Develop functions for your scenario and package as container

AI & Azure Machine Learning

Package AI and ML model as module in a container after training using ML studio. Deploy packaged ML modules to the IoT edge.

Azure blob storage

Breakthrough intelligence capabilities, in the cloud and on the edge

Azure SQL database Edge

Enabling intelligent data at the edge

Features of the Device Provisioning Service (DPS)

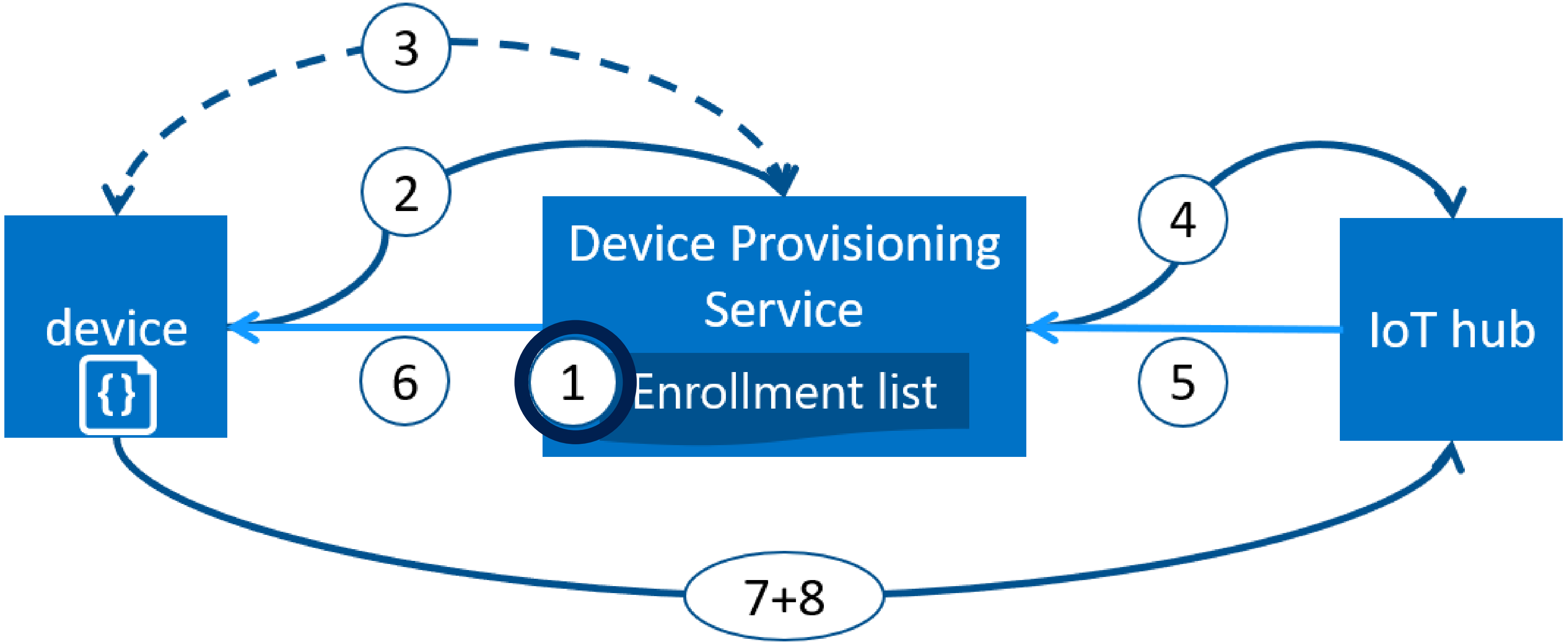
- Secure attestation support (X.509 and TPM-based identities)
- A configurable, updatable enrollment list containing the complete record of devices/groups of devices that may at some point register
- Multi-hub support (including across subscriptions and regions), assigned by multiple allocation policies
- Monitoring and diagnostics logging to make sure everything is working properly.
- Cross-platform support
  - A variety of operating systems
  - SDKs across multiple languages
  - HTTPS, AMQP, and MQTT protocol support (Service SDK is HTTPS only)

When to use the Device Provisioning Service

If you want any of these capabilities, the DPS is a good choice:

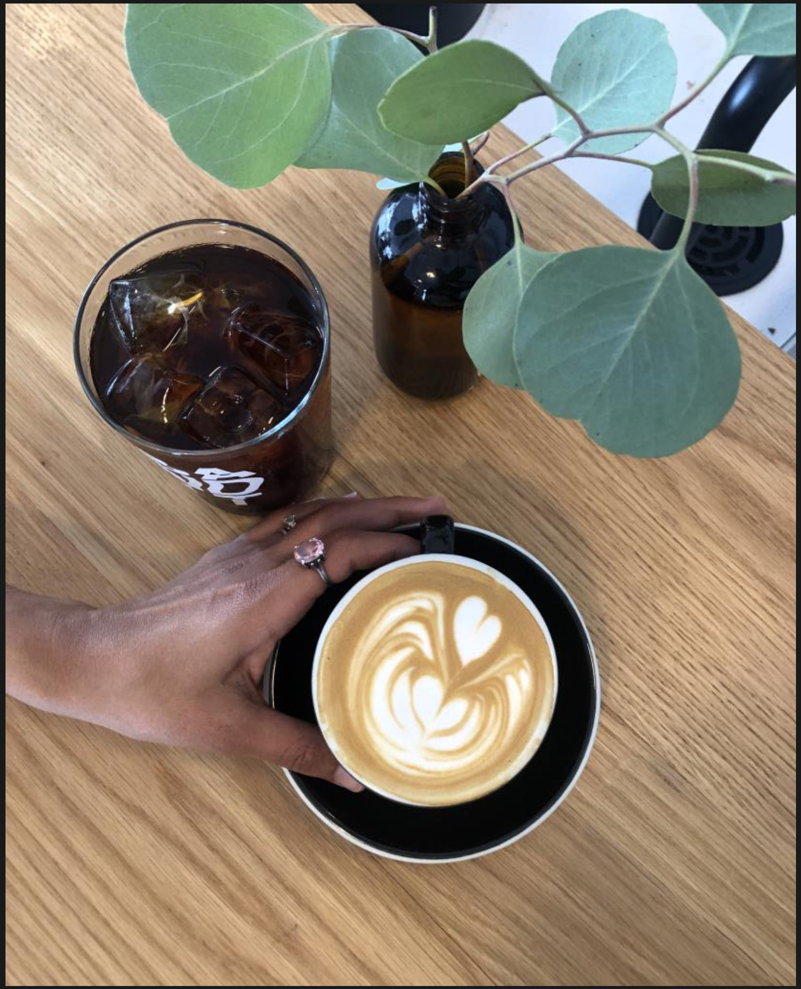
- Zero-touch provisioning
- Load balancing
- Connecting devices (multitenancy, solution isolation, geo-sharding)
- Reprovisioning
- Rolling keys

DPS Auto-Provisioning Behind the Scenes

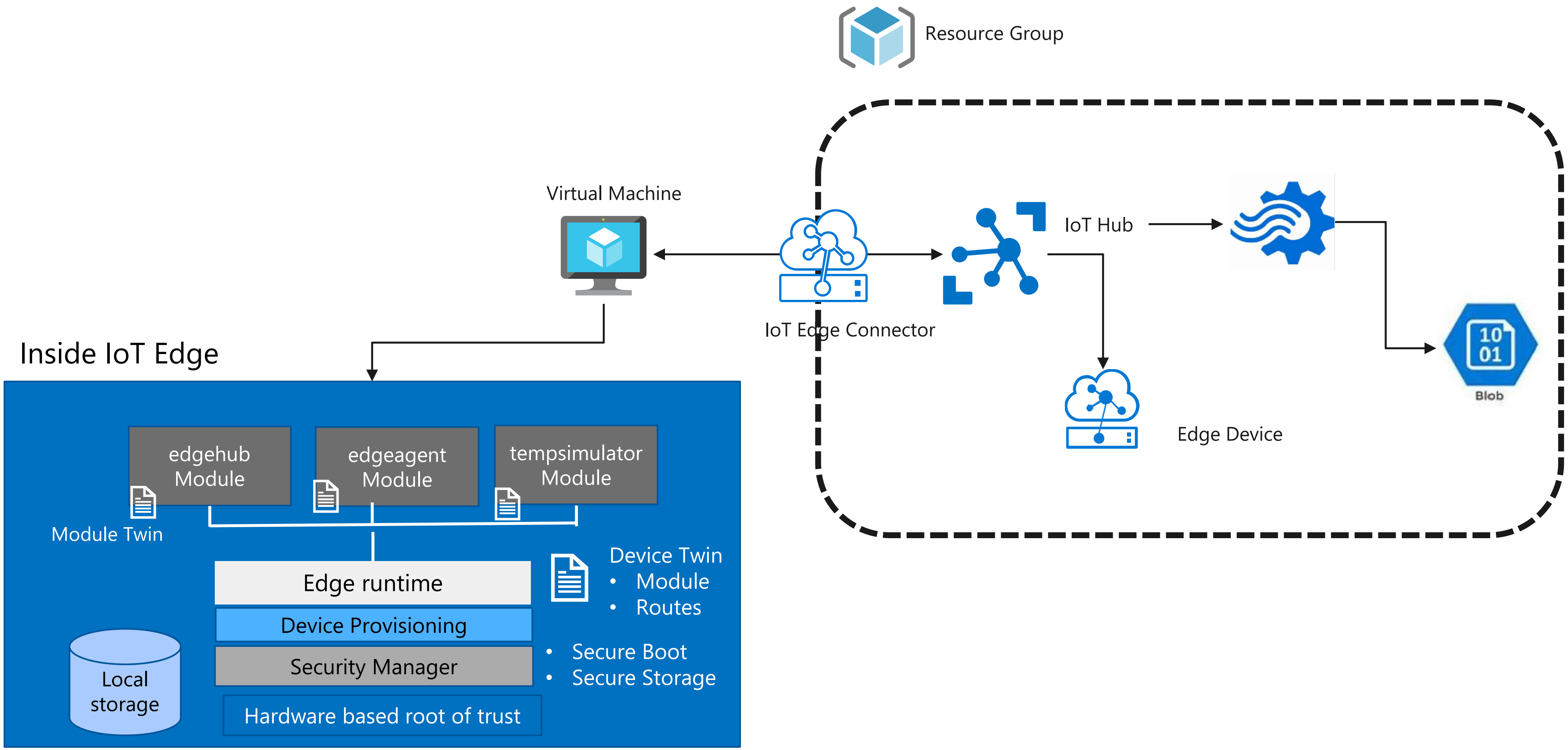


[illegible]

Break Time!



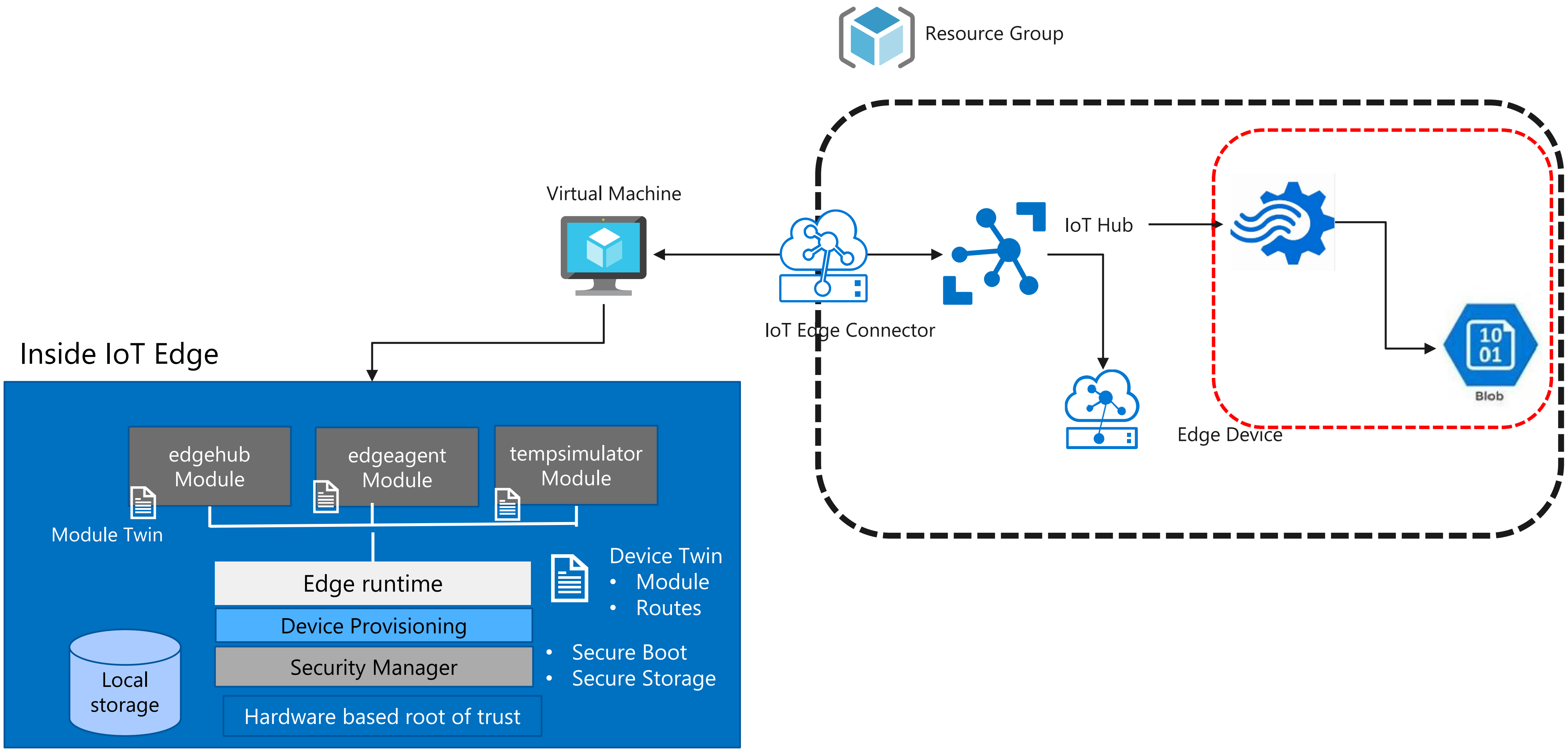
Exercise #4 – Telemetry Data



Azure Stream Analytics Data Flow

- An Azure Stream Analytics *job* consists of an *input*, *query*, and an *output*
- Input – ASA can ingest data from Azure Event Hubs, Azure IoT Hub, or Azure Blob Storage
- Query – ASA uses a SQL-like query language that includes support for filtering, sorting, aggregating, joining, and user-defined functions
- Output – ASA can output to many targets

Recap Exercise #4 – Telemetry Data



Resource Group

Virtual Machine

IoT Edge Connector

IoT Hub

Edge Device

Blob

10
01

Inside IoT Edge

edgehub Module

edgeagent Module

tempsimulator Module

Module Twin

Edge runtime

Device Provisioning

Security Manager

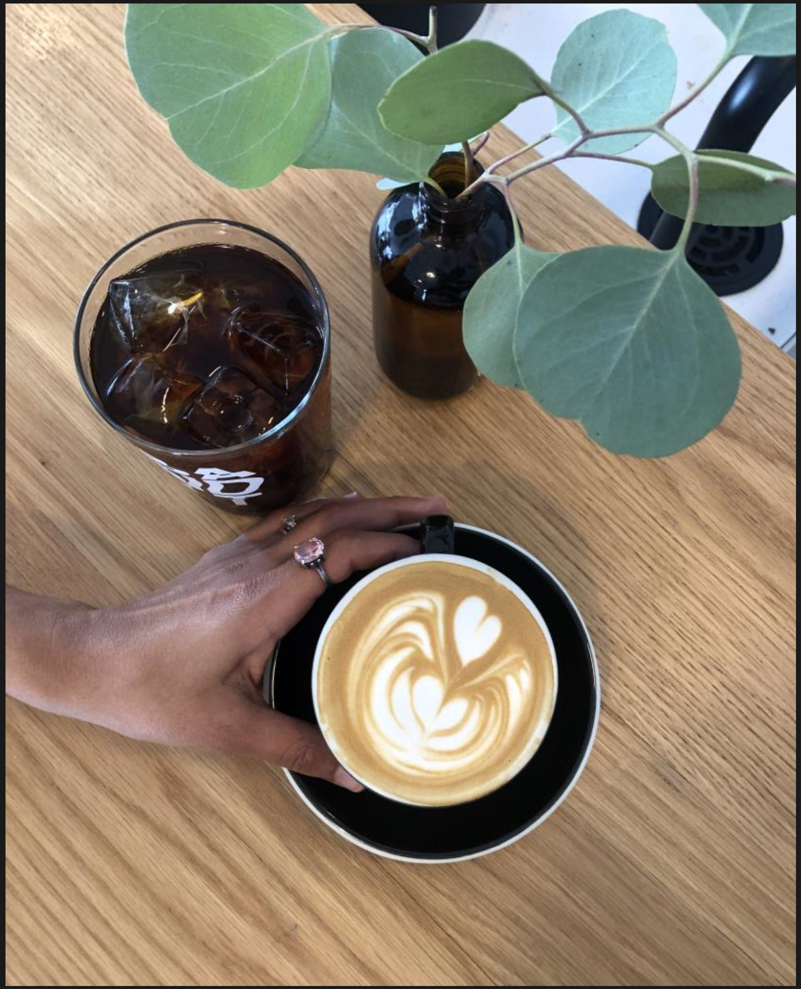
Hardware based root of trust

Local storage

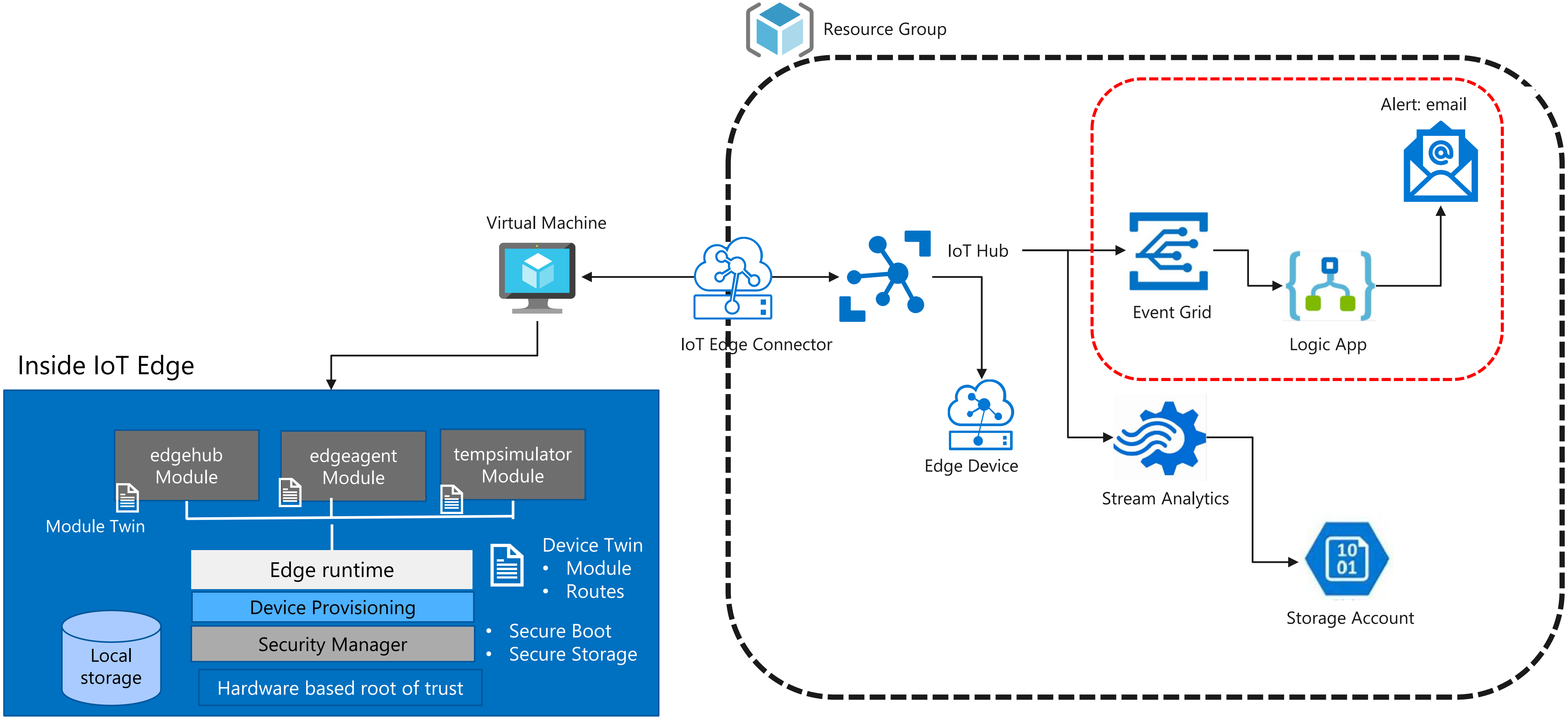
Device Twin
- Module
- Routes

- Secure Boot
- Secure Storage

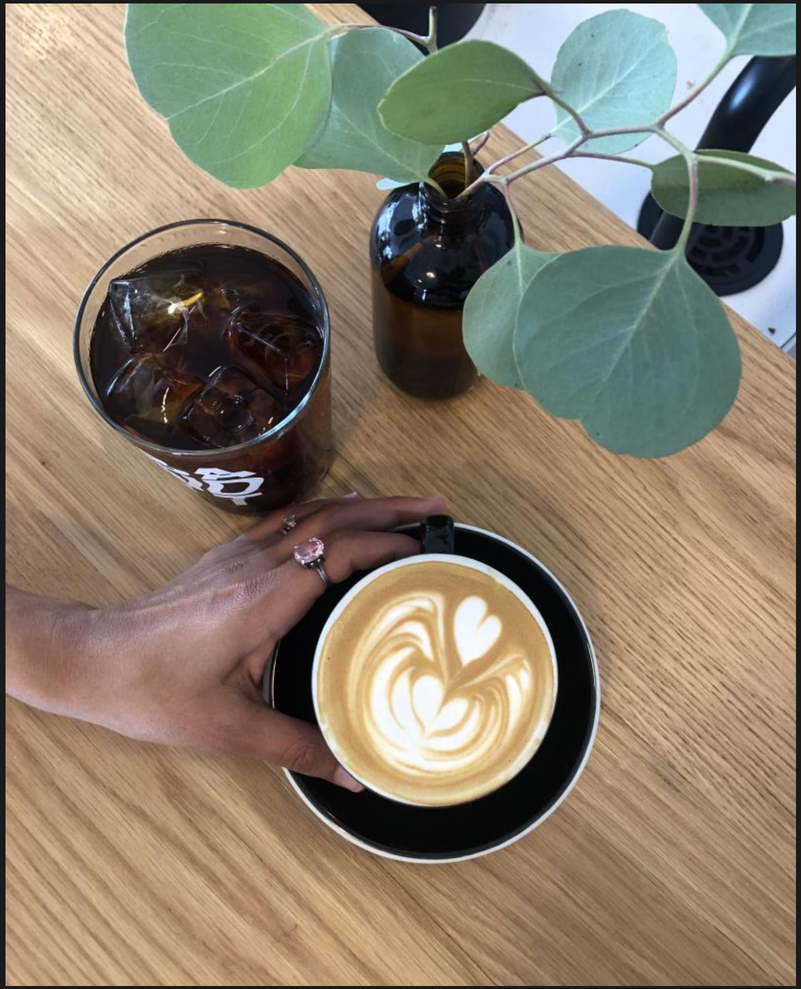
Break Time!



Exercise #5 – Monitoring Devices

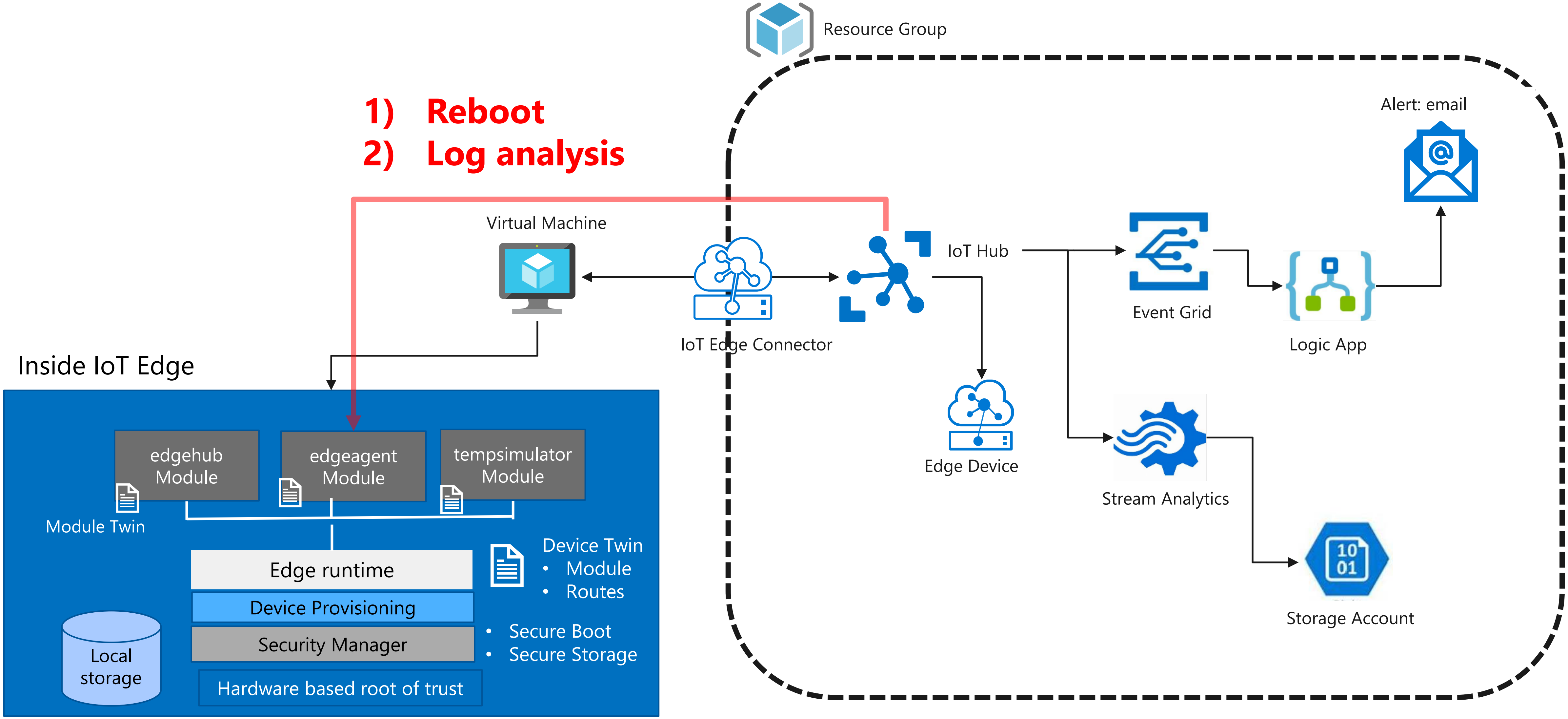


Break Time!



Exercise #6 – Interacting with remote Devices

1) **Reboot**
2) **Log analysis**



Direct Methods: Introduction

- *Direct methods* – request from the cloud to a device, executing code directly on the target
- Features
  - Each call targets a single device or module instance
  - Can be used by anyone with appropriate IoT Hub permissions
  - Follow a request-response pattern for immediate feedback
- Lifecycle
  - Called by a back-end application through an HTTPS URL pattern on the IoT Hub
  - Translated to MQTT or AMQP on the device side
  - Reply received from the device sent directly back to the back-end application

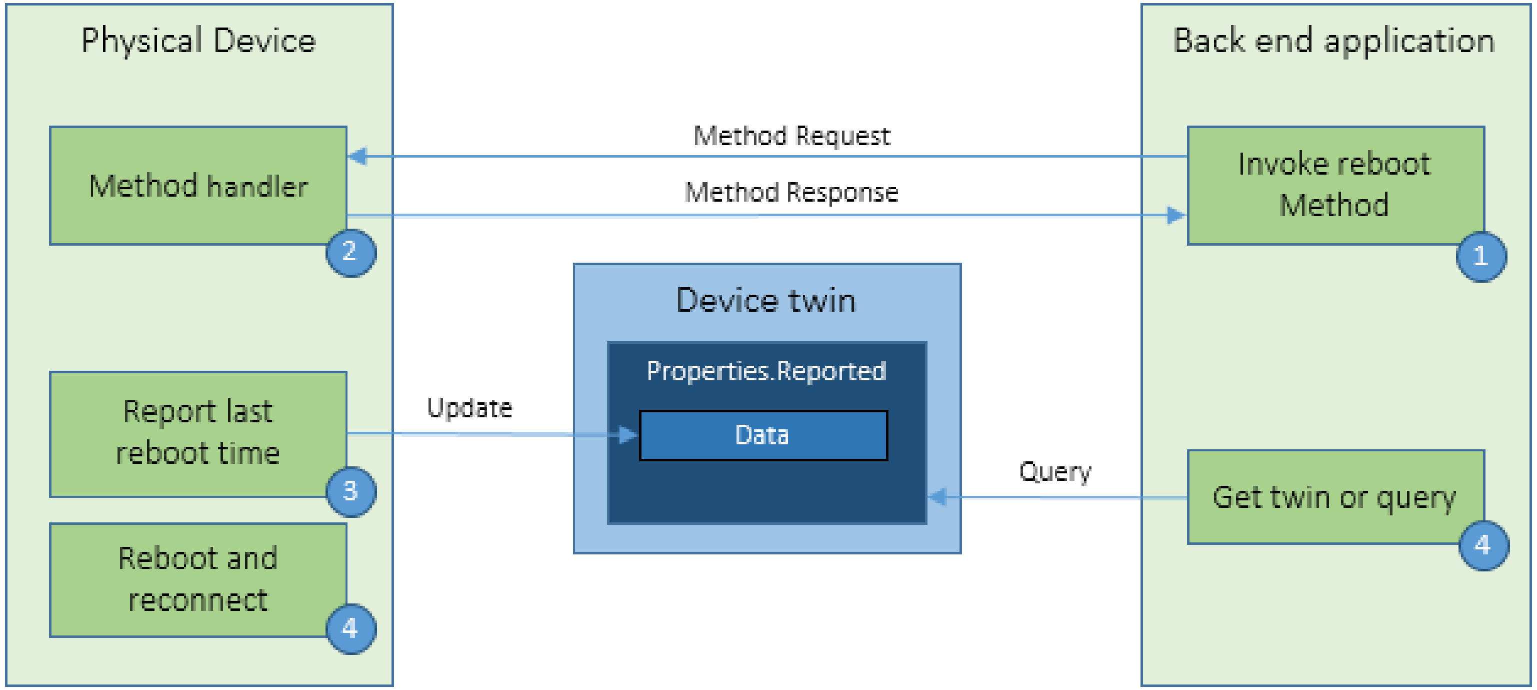
Comparing Device Management Approaches

	Direct Method Call	Device Twins	Cloud-to-device messages
Scenario	Requires immediate confirmation	Long-running desired state configuration	One-way notifications.
Data flow	Two-way with immediate response	One-way to the device	One-way to the device
Durability	Disconnected devices are not contacted. The solution back end is notified that the device is not connected.	Property values are preserved in the device twin. Device will read it at next reconnection. Property values are retrievable with the IoT Hub query language.	Messages can be retained by IoT Hub for up to 48 hours.
Targets	Single device using deviceId, or multiple devices using jobs.	Single device using deviceId, or multiple devices using jobs.	Single device by deviceId.
Size	Payload maximum is 128 KB.	Desired properties maximum is 8 KB.	Up to 64 KB messages.
Frequency	High	Medium	Low
Protocol	MQTT or AMQP.	MQTT or AMQP.	MQTT, AMQP, HTTPS

Appendix

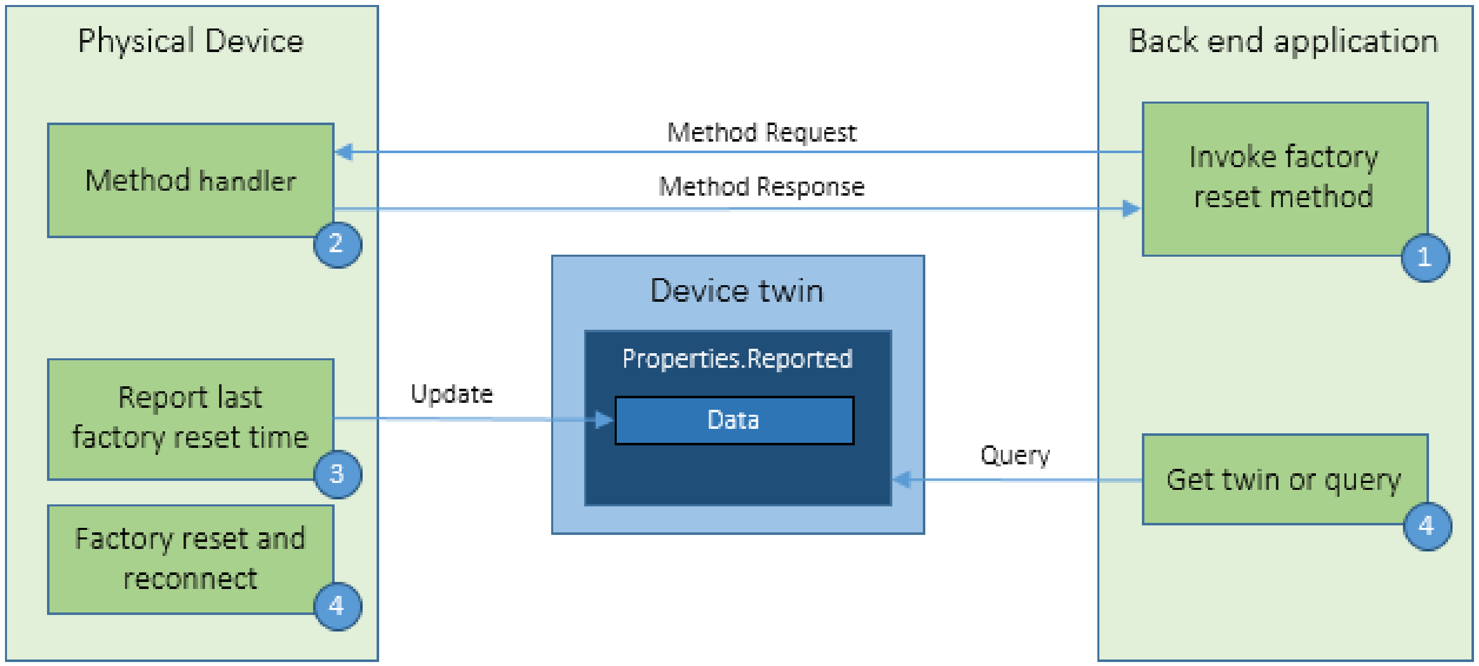
Device Management Patterns

- Reboot
- Factory Reset
- Configuration
- Firmware Update
- Reporting Progress and Status



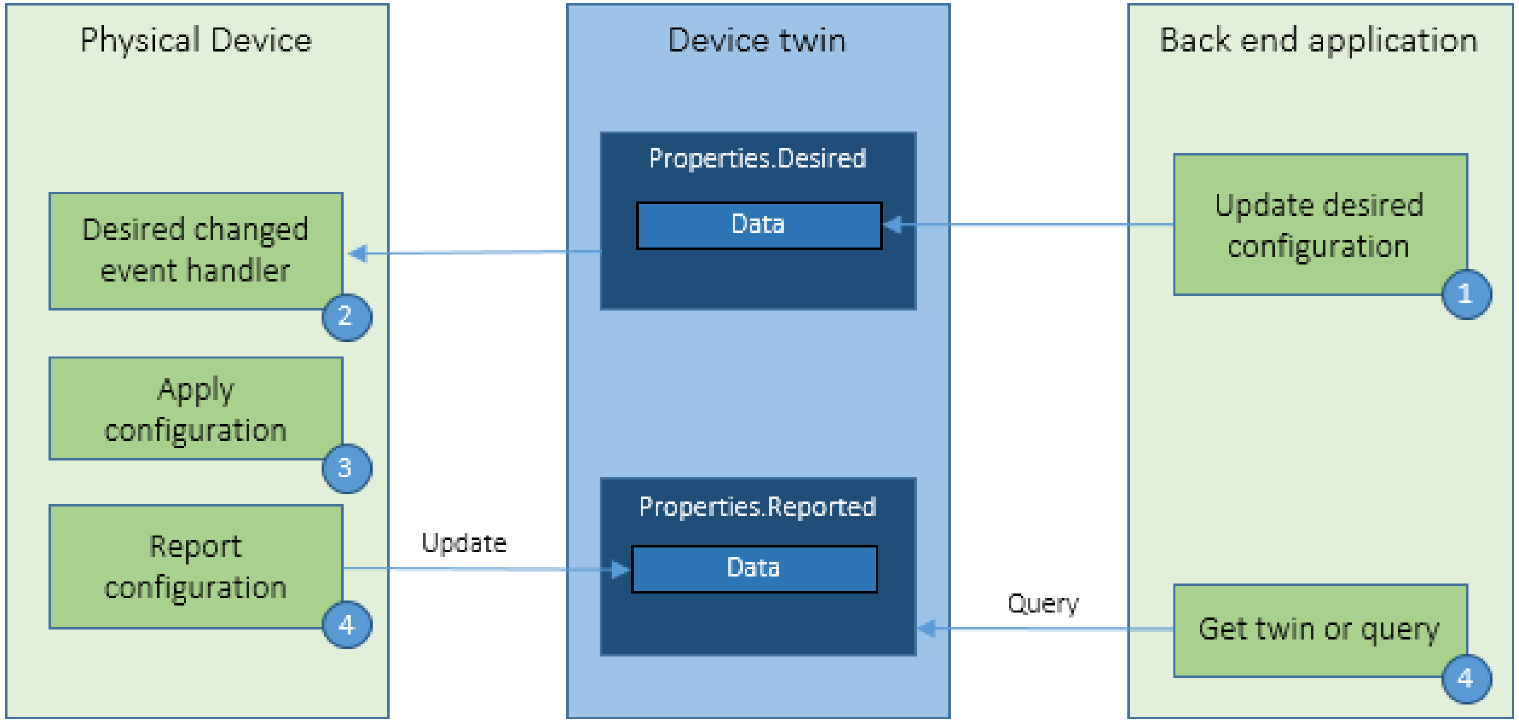
Device Management Patterns

- Reboot
- Factory Reset
- Configuration
- Firmware Update
- Reporting Progress and Status



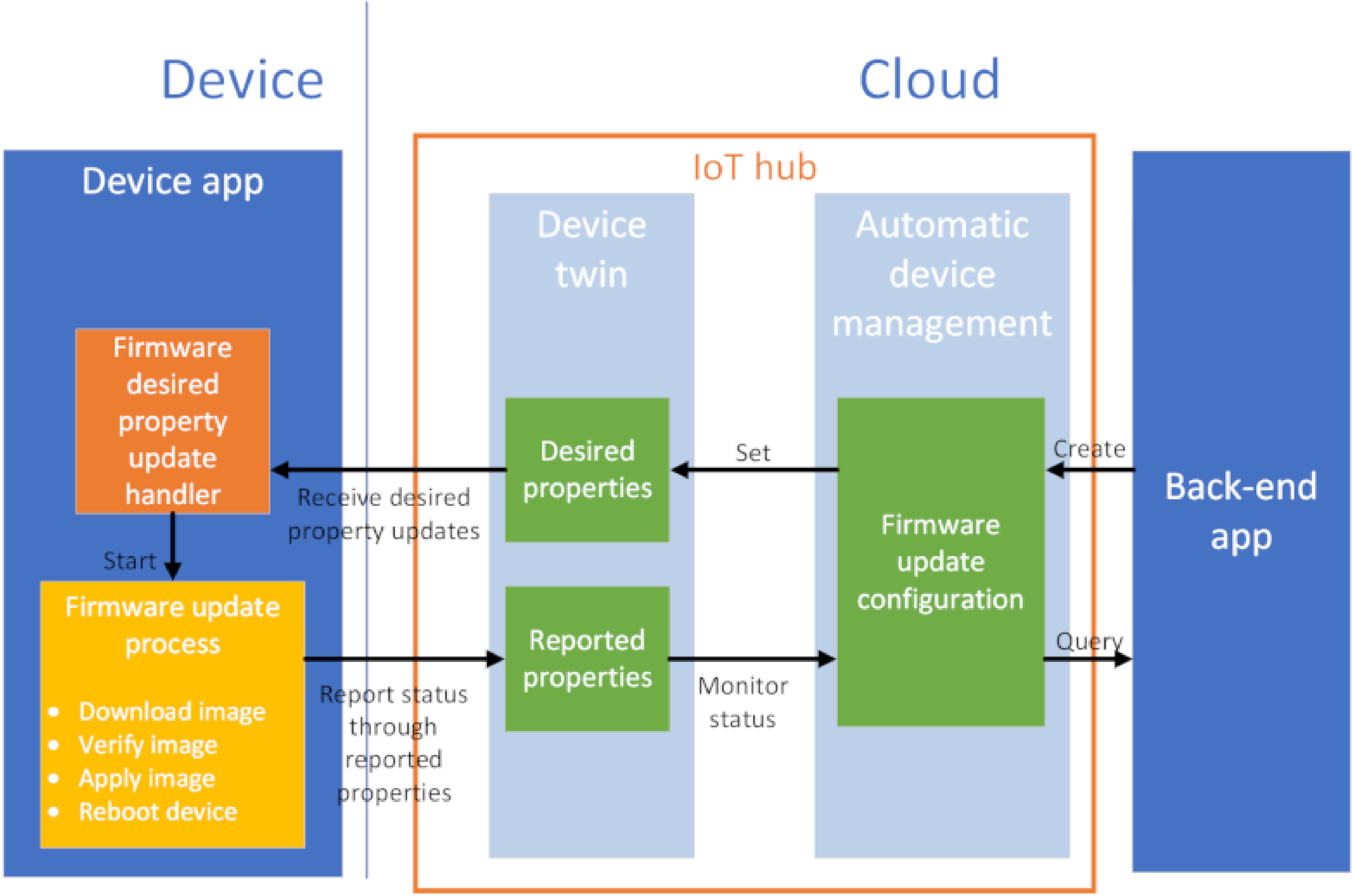
Device Management Patterns

- Reboot
- Factory Reset
- **Configuration**
- Firmware Update
- Reporting Progress and Status



Device Management Patterns

- Reboot
- Factory Reset
- Configuration
- **Firmware Update**
- Reporting Progress and Status



Introduction to IoT Device Software

- OS Support
  - Windows 10 IoT
  - Ubuntu Core
  - Riot
  - QNX
  - Android Automotive
  - etc.
- Software Development Kits
  - Device SDKs
  - Service SDKs
  - Device Provisioning SDKs
- Programming Language Support
  - C/C++
  - Java
  - C#
  - Python
  - etc.

Features of Azure IoT Hub Device Provisioning Service

Features of the Device Provisioning Service include:

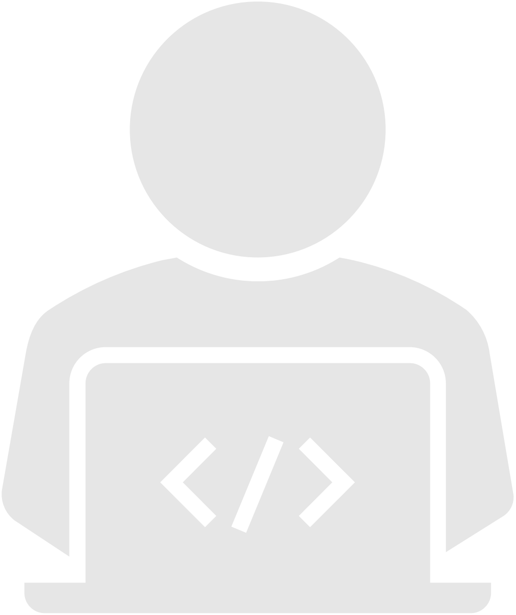
- Secure attestation
- Enrolment list
- Allocation policies
- Monitoring
- Multi-hub support

Azure IoT Hub SDKs

- Benefits of using the SDKs
  - Develop a “future-proof” solution with minimal code
  - Leverage features designed for a complete software solution and focus on your specific need
  - Develop with your preferred language for different platforms
  - Benefit from the flexibility of open source with support from Microsoft and community
- Included SDKs
  - IoT Hub Device SDKs
  - IoT Hub Service SDKs
- Platform support
  - C, .NET (C#), Node.js, Java, and Python

Azure IoT Hub Device SDKs: Languages

- C (easily ported!)
- C#
- Java
- Node.js
- Python



Azure IoT Hub Device SDKs: Sample Platforms

- Linux (Ubuntu, Debian, Raspbian)
- Windows
- MBED
- Arduino
  - Huzzah, ThingDev, FeatherM0
  - FreeRTOS (ESP32, ESP8266)
- .NET Variations
  - .NET Framework 4.5
  - PCL (Profile 7 – UWP, Xamarin.iOS, Xamarin.Android)
  - .NET Standard 1.3
- Intel Edison

Azure IoT Hub Device SDKs: Protocols

- MQTT
- MQTT over WebSockets
- AMQP
- AMQP over WebSockets
- HTTPS

Communication Protocols: Protocol Comparison

Protocol	**Port**	**When you should use this protocol**
MQTT	8883	Use on all devices that do not require to connect multiple devices (each with its own per-device credentials) over the same TLS connection.
MQTT over WebSockets	443	
AMQP	5671	Use on field and cloud gateways to take advantage of connection multiplexing across devices.
AMQP over Websockets	443	
HTTPS	443	Use for devices that cannot support other protocols.

Azure IoT Hub Service SDKs

- Coding language support
  - C
  - C#
  - Java
  - Node.js
  - Python
- Backend Scenarios
  - Identity registry
  - Cloud-to-device messaging
  - Direct method operation
  - Querying
  - Jobs

Visual Studio Code Extensions

- Azure IoT Tools collection
  - Azure IoT Hub Toolkit
  - Azure IoT Edge
  - Azure IoT Device Workbench

Azure CLI Tools

- Added by Azure CLI extensions for IoT
  - `az extension add --name azure-iot`
- Hub Commands
  - `create, delete, show-connection-string, etc.`
- Subgroup commands
  - `device-identity, device-twin, etc.`
- Running CLI commands
  - Example
    - `az iot hub create --resource-group MyResourceGroup --name MyIotHub`
  - `Getting help`
    - `az iot hub <command name> --help`