Technology Solutions Professional @Microsoft

Application Development and IoT

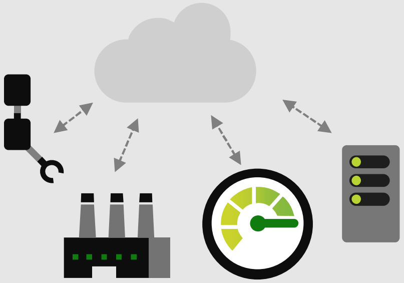
@vslepakov

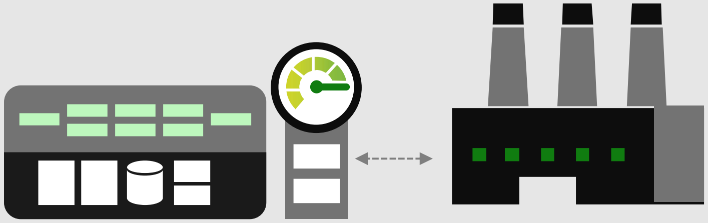
vitaliy.slepakov@microsoft.com

Agenda

- IoT Edge 101
- Inner-loop development workflow
- IoT Edge DevSecOps with Azure DevOps

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

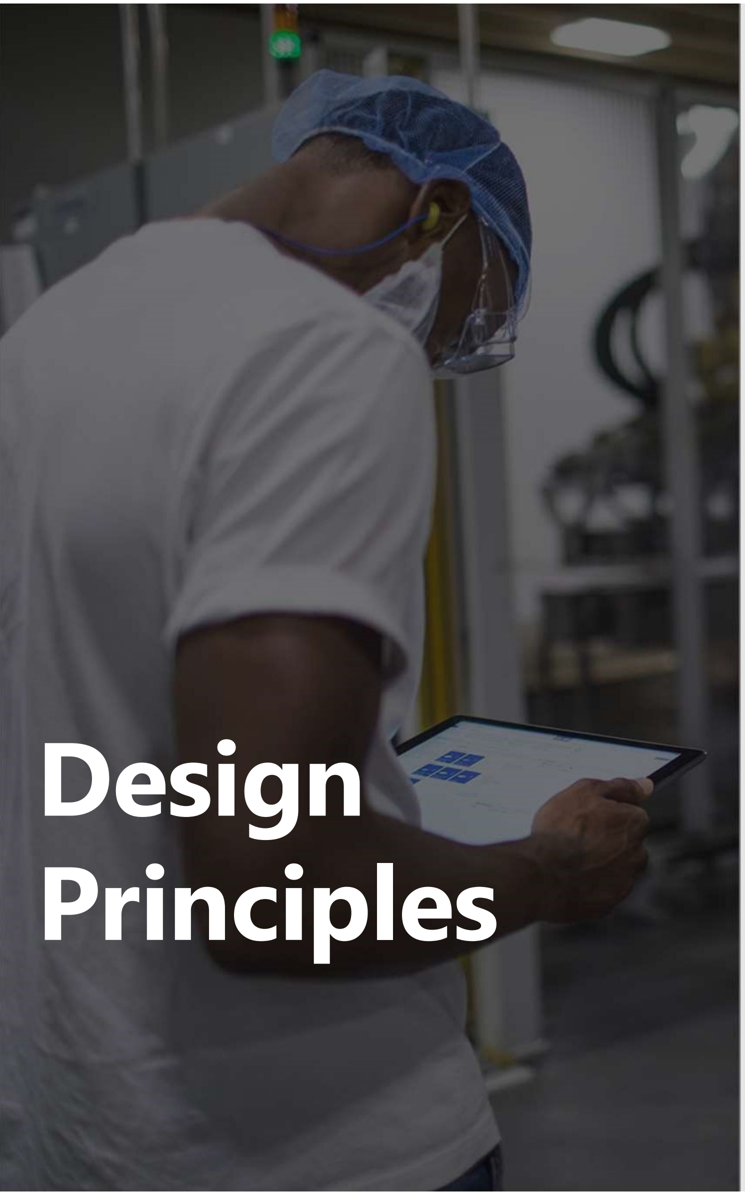
IoT on the Edge

Low latency tight control loops require near real-time response

Protocol translation & data normalization

Privacy of data and protection of IP

Symmetry



Design Principles

Secure

Provides a secure connection to the Azure IoT Edge, update software/firmware/configuration remotely, collect state and telemetry and monitor security of the device

Cloud managed

Enables rich management of Azure IoT Edge from Azure provide a complete solution instead of just an SDK

Cross-platform

Enables Azure IoT Edge to target the most popular edge operating systems, such as Windows and Linux

Portable

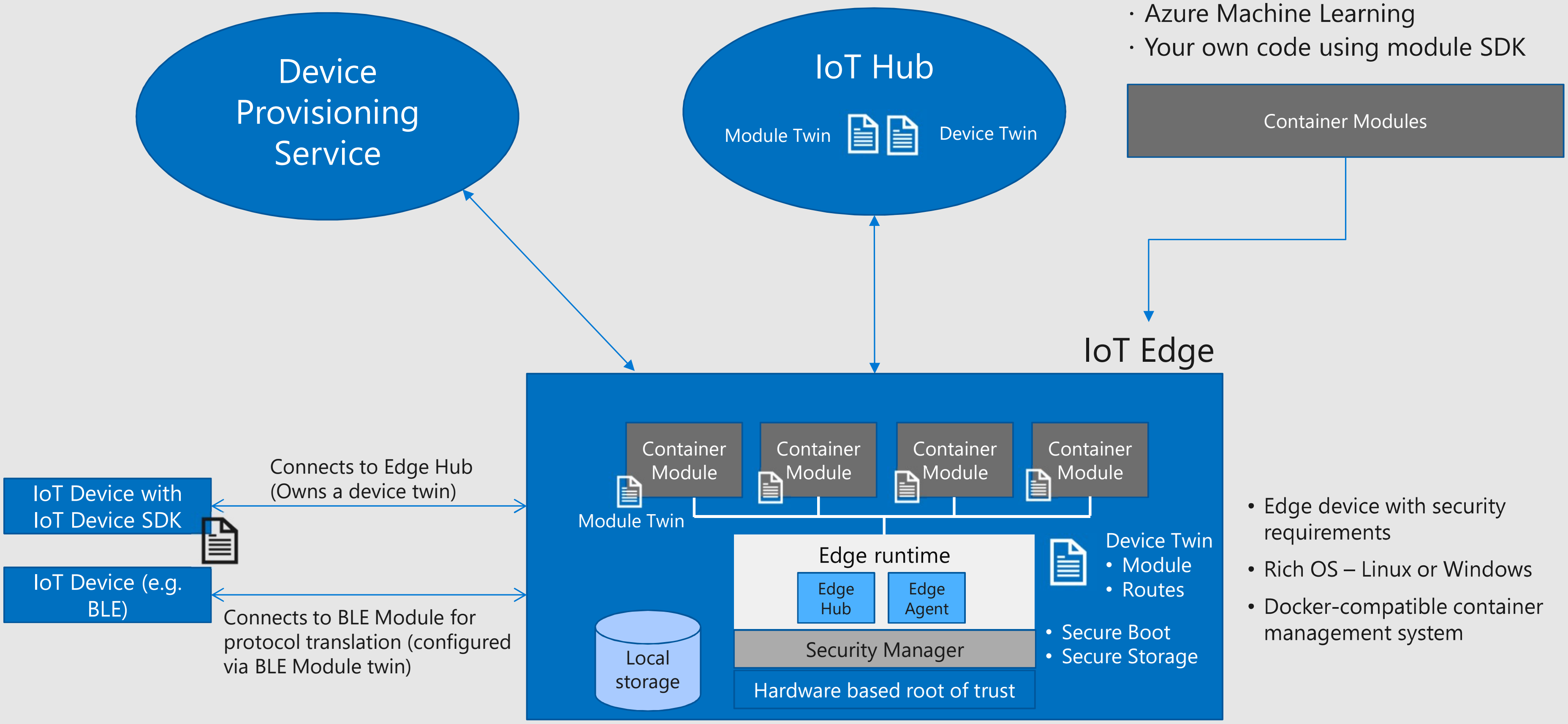
Enables Dev/Test of edge workloads in the cloud with later deployment to the edge as part of a continuous integration / continuous deployment pipeline

Extensible

Enables seamless deployment of advanced capabilities such as AI from Microsoft, and any third party, today and tomorrow

IoT Edge in action

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



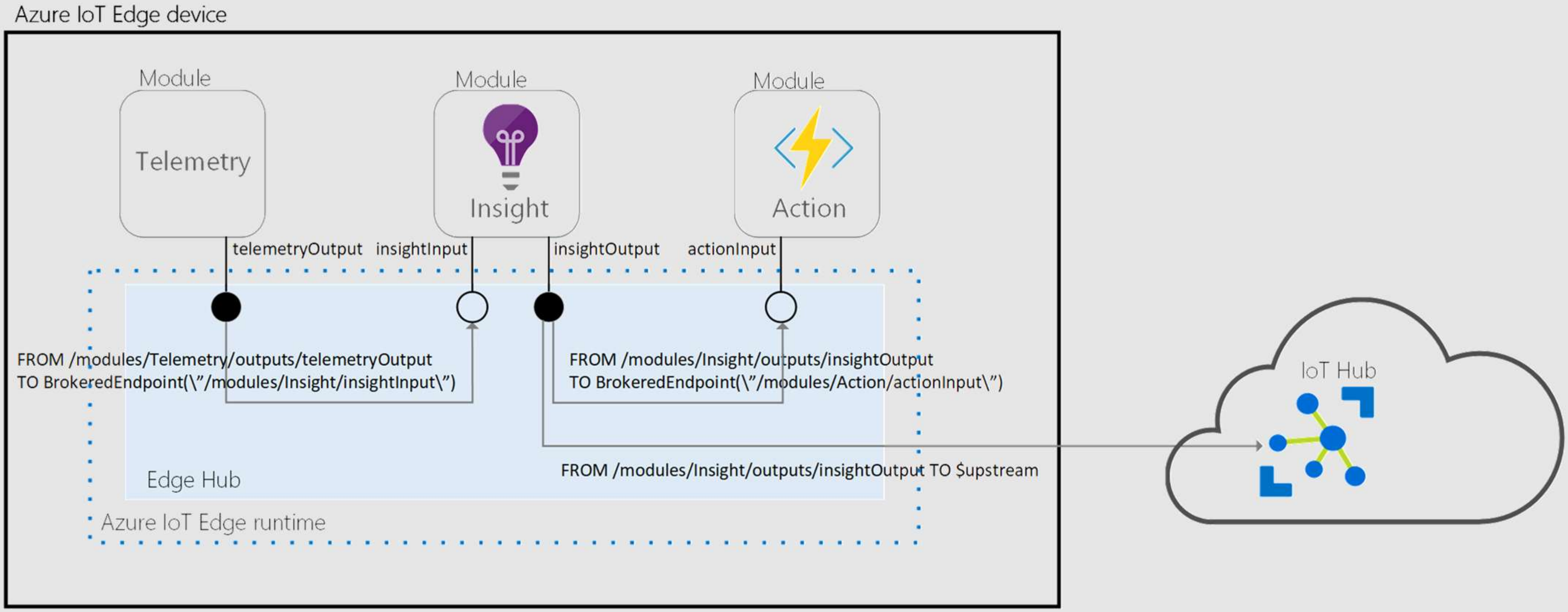
Connects to Edge Hub (Owns a device twin)

Connects to BLE Module for protocol translation (configured via BLE Module twin)

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

Concept

Routing



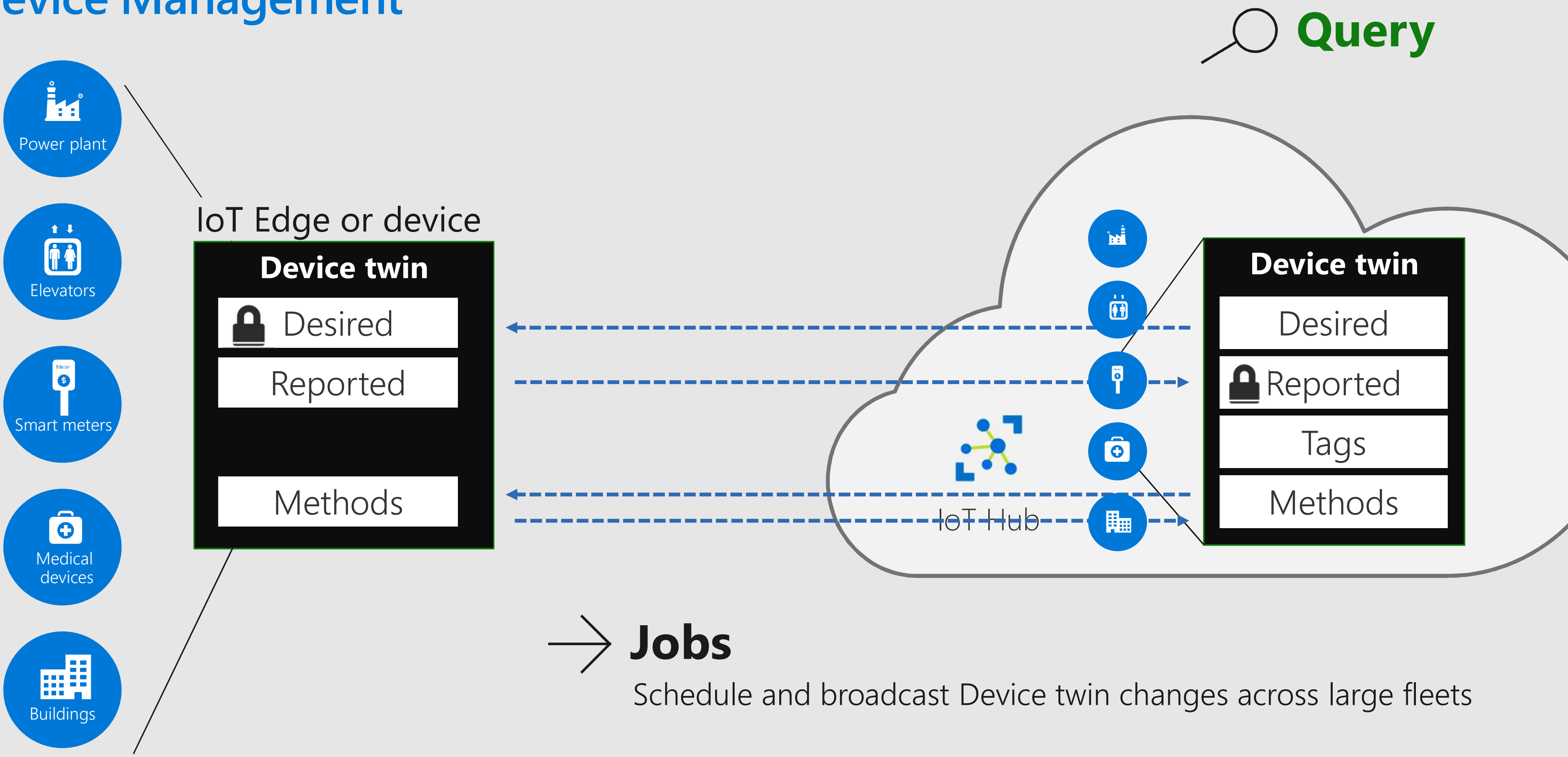
FROM /modules/Telemetry/outputs/telemetryOutput
TO BrokeredEndpoint(\"/modules/Insight/insightInput\")

FROM /modules/Insight/outputs/insightOutput
TO BrokeredEndpoint(\"/modules/Action/actionInput\")

FROM /modules/Insight/outputs/insightOutput TO $upstream

Concept

Device Management



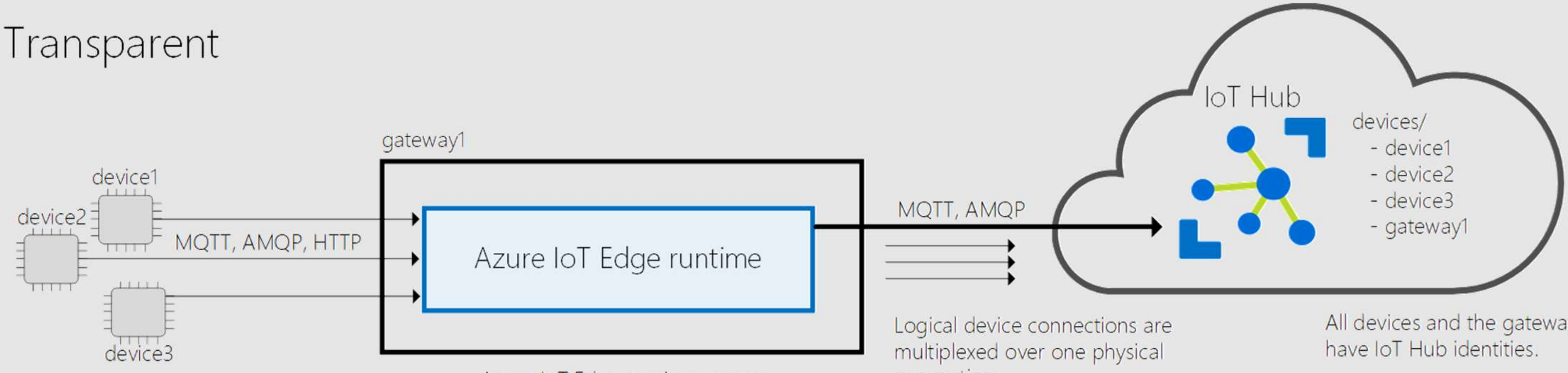
Query

→ **Jobs**

Schedule and broadcast Device twin changes across large fleets

Transparent Gateway

Transparent



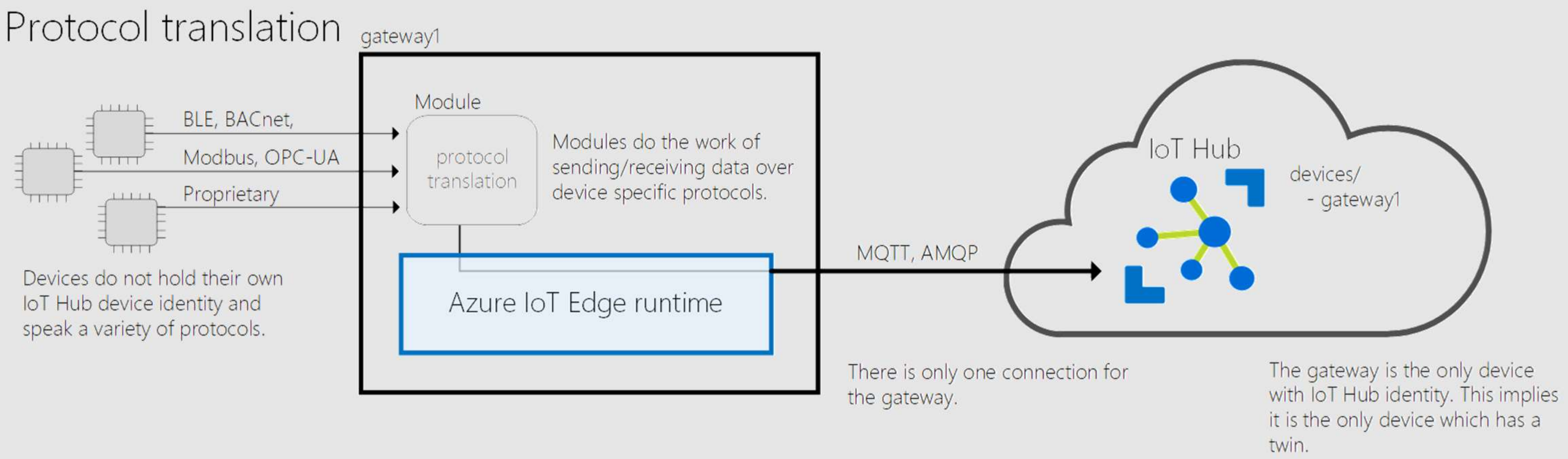
Devices hold their own IoT Hub device identity and speak a protocol understood by IoT Hub.

Azure IoT Edge runtime passes communications between devices and IoT Hub.

Logical device connections are multiplexed over one physical connection.

All devices and the gateway have IoT Hub identities.

Protocol translation



Protocol translation

gateway1

Module

protocol translation

BLE, BACnet,

Modbus, OPC-UA

Proprietary

Modules do the work of sending/receiving data over device specific protocols.

Azure IoT Edge runtime

MQTT, AMQP

IoT Hub

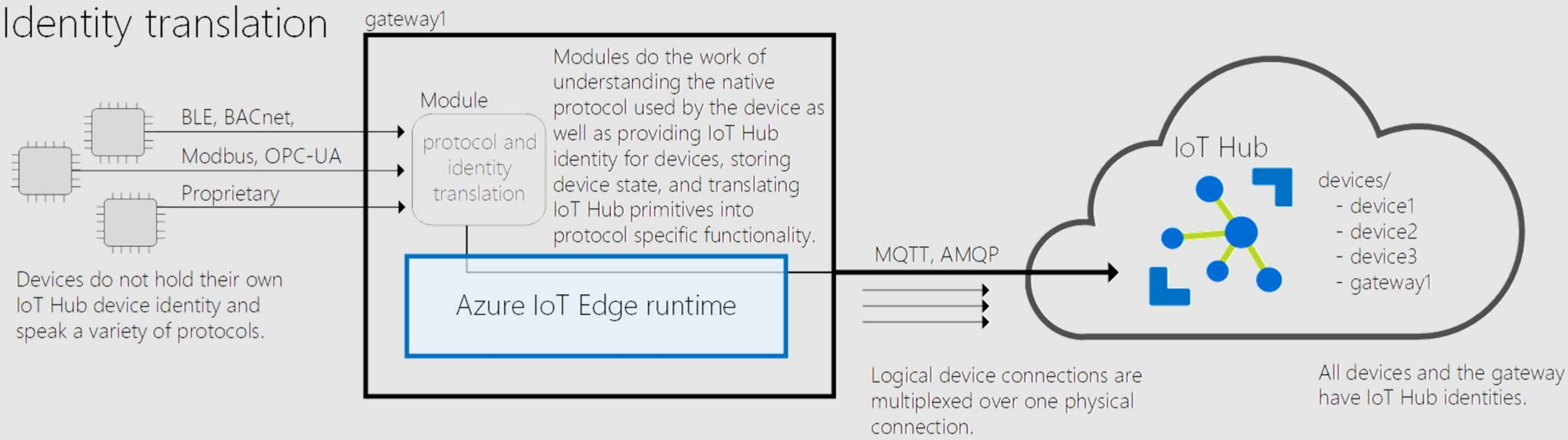
devices/
- gateway1

Devices do not hold their own IoT Hub device identity and speak a variety of protocols.

There is only one connection for the gateway.

The gateway is the only device with IoT Hub identity. This implies it is the only device which has a twin.

Identity translation



Modules do the work of understanding the native protocol used by the device as well as providing IoT Hub identity for devices, storing device state, and translating IoT Hub primitives into protocol specific functionality.

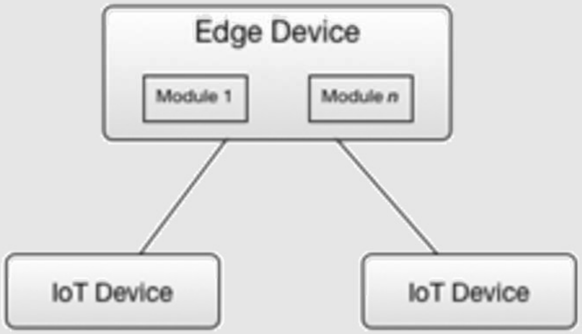
Devices do not hold their own IoT Hub device identity and speak a variety of protocols.

Logical device connections are multiplexed over one physical connection.

All devices and the gateway have IoT Hub identities.

Extended offline operation

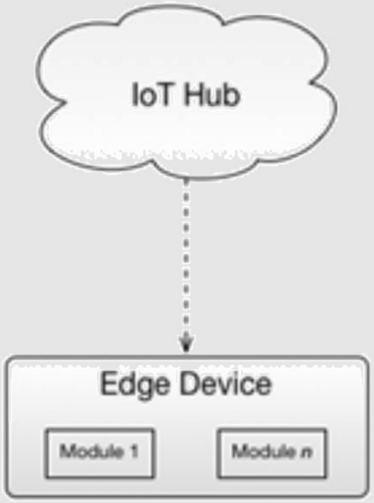
① Assign child devices to Edge device



- ✓ Establish parent-child relationship in IoT Hub portal
- ✓ Local modules are extended offline capable out-of-the-box

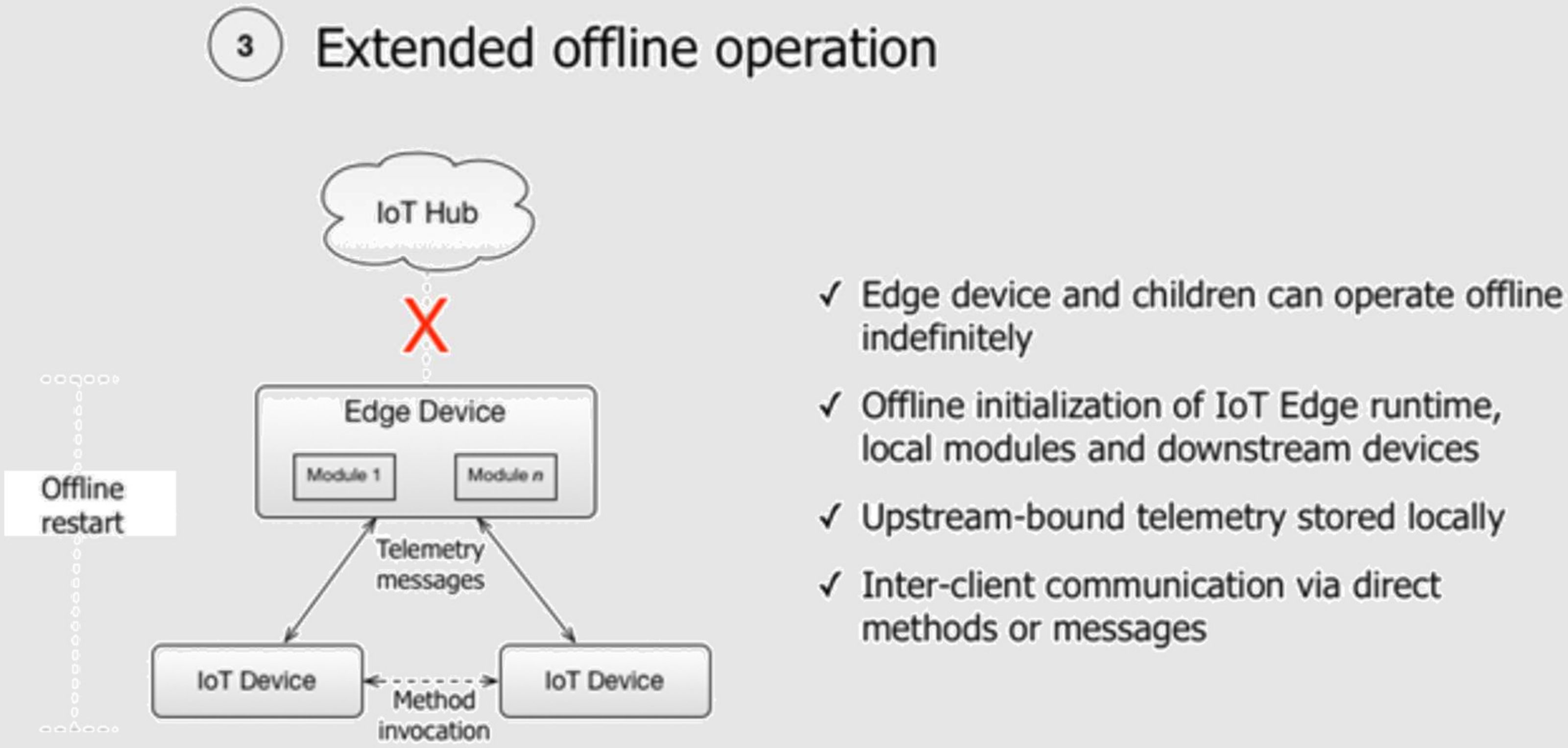
Extended offline operation

2 One-time sync with IoT Hub



- ✓ Get details of child devices
- ✓ Securely update local cache to enable offline operation
- ✓ Retrieve settings for local storage of telemetry messages

Extended offline operation

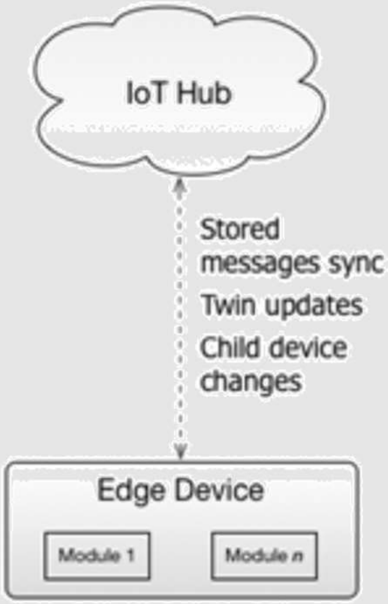


3 Extended offline operation

- ✓ Edge device and children can operate offline indefinitely
- ✓ Offline initialization of IoT Edge runtime, local modules and downstream devices
- ✓ Upstream-bound telemetry stored locally
- ✓ Inter-client communication via direct methods or messages

Extended offline operation

4 Re-sync with IoT Hub



- ✓ Locally stored messages delivered to IoT Hub
- ✓ Desired/Reported property changes reconciled
- ✓ Child device updates (add/remove) synced

IoT Edge DevOps Challenges

Code and Dependency Security

Integration Testing

Inner Loop Development Workflow

Modules Maintained by Different Teams

Continuous Deployment and Tracking of New Releases

HA/DR

Container Images That Can Be Trusted by Edge Devices

Monitoring

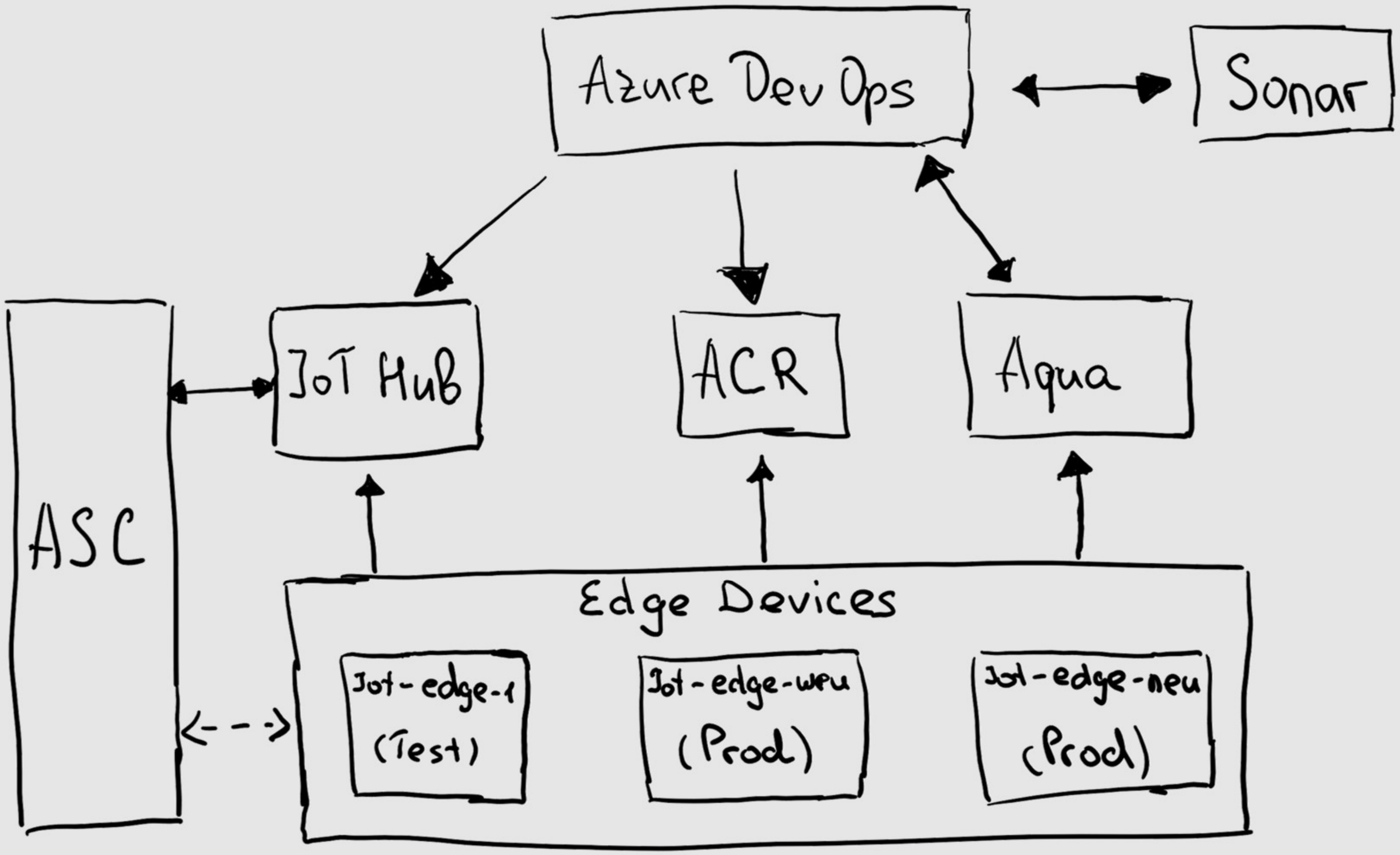
Container Image Security

Deploy Only to Specific Devices

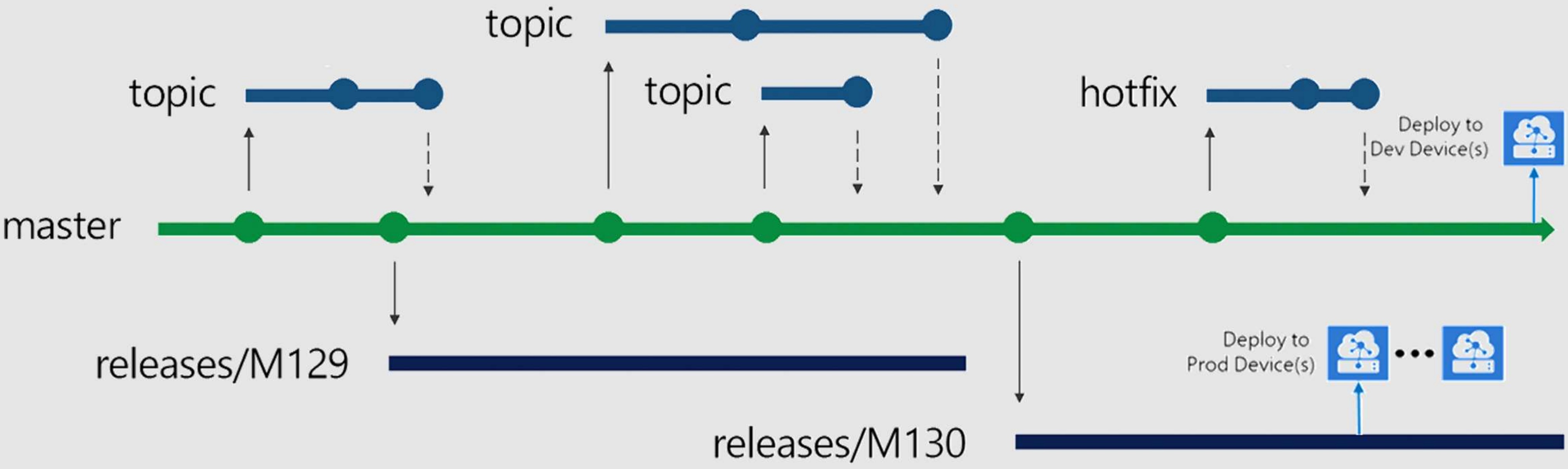
Device Identity and Provisioning

Continuous OS and Framework Base Image Patching

Setup

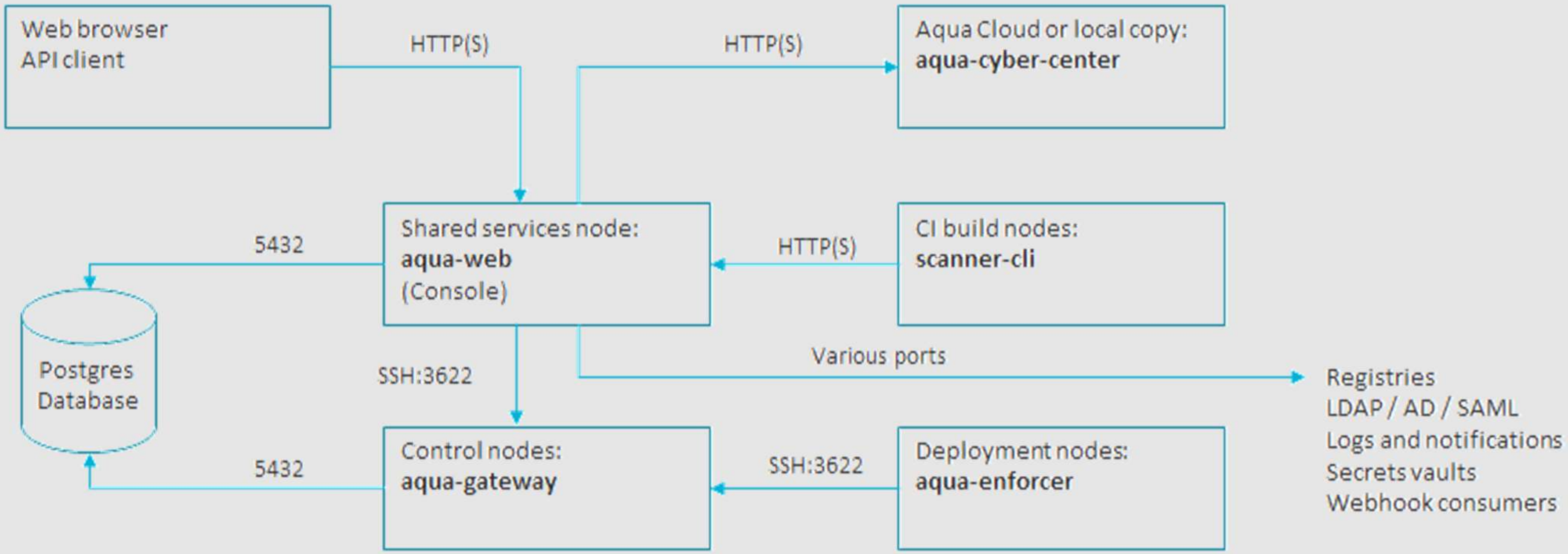

Azure Dev Ops
Sonar
IoT Hub
ACR
Aqua
ASC
Edge Devices
Iot-edge-1
(Test)
Iot-edge-weu
(Prod)
Iot-edge-neu
(Prod)

IoT Edge DevOps Example (Release Flow)



Aqua Security

Note: Default port numbers are shown; they can be changed.



Web browser
API client

HTTP(S)

HTTP(S)

Aqua Cloud or local copy:
aqua-cyber-center

5432

Shared services node:
aqua-web
(Console)

HTTP(S)

CI build nodes:
scanner-cli

Postgres
Database

SSH:3622

Various ports

Registries
LDAP / AD / SAML
Logs and notifications
Secrets vaults
Webhook consumers

5432

Control nodes:
aqua-gateway

SSH:3622

Deployment nodes:
aqua-enforcer

Microsoft