* **System architecture (device to hub and hub to cloud)**

Ein Bild, das Text, Screenshot, Diagramm enthält.

KI-generierte Inhalte können fehlerhaft sein.

* **Authentication and authorization strategy**

Authentication

1. **HTTP Basic Auth**

HttpClient is sending username and password in Authorization-Header for RegisterDeviceAsync and SendHeartbeatAsync.

1. **Client-Zertificates (TLS Mutual Auth)**

HttpClientHandler is using X509Certificate2 to secure transmission.

MQTT connection is secured via StaticCertificateProvider.

1. **MQTT Username/Password**

Mqtt connection is also secured with credentials / name and password.

Authentication strategy:

-REST API uses Basic Authentication over HTTPS, combined with mutual TLS (client certificate)

- MQTT uses user/password authentication over TLS (port 8883).

- Devices identify themselves using a unique client ID and client certificates.

Authorization

1. **Topic-based authorization on the MQTT-broker.**

Authorization can be managed by ACL configuration in Mosquitto.

1. **Server side roles**

REST: /register can check if a device is entitled to register.

JWT can extend the token based authentication.

Authorization Strategy:

- Devices are registered via a /register endpoint and authenticated using known credentials.

- Each device may be restricted to specific MQTT topics (e.g., topic filters per device).

- The system can evolve toward token-based or role-based access control for scalability.

* **Encryption Methods**

**TLS (Transport Layer Security) – Asymmetric + Symmetric**

**Used for:**

* HTTPS-Connections (HttpsClient to <https://hub.io>)
* MQTT .WithTlsOptions (Port 8883)

Asymmetric encryption for key exchange / Symmetric Encryption for transmission.

**Client-Certificate – Asymmetric Authentication**

For mTLS enabled HTTP-Client and MQTT.

**Basic Auth (over HTTPS) – No encryption**

Over TLS the transmission is secure.

* **Threat Model and mitigation strategies**

1.)Device is communicating via MQTT + HTTPS with the Hub in local Network

Authentication: Basic Auth + Certificates

Communication: Sensor <-> Hub via local Network

2.)Hub is sending data to Azure Event Hub via Internet.

Hub <-> Azure via Internet. Azure Event Hub provides security layer. e.g.(TLS, SAS-Tokens, Azure AD)

|  |  |  |
| --- | --- | --- |
| **Category** | **Thread** | **Mitigation** |
| Sniffing (Eavesdropping) | Attacker logs MQTT/HTTP-traffic | TLS 1.2 on both protocols |
| Man-in-the-Middle (MitM) | Attacker fakes server or intercepts traffic | Certificate validation and mTLS (Client-Certs) |
| Replay- Attack | Attackers send old invalid packages | Time-based tokens or Nonces optional (unique id/counter) |
| Device impersonation | An attacker is emulating your device | DeviceId and a device certificate issued by a trusted root CA |
| Credential leak | Basic Auth is leaked (e.g. in code or log files) | Keep secrets in a separate file, not hard coded |
| Denial of Service (DoS) | The device or server is being flooded with requests and payloads. | Rate Limits, Topic-Filter on Broker. Use Reverse proxy. |
| MQTT Topic Hijacking | Attacker publishes on foreign topics | Usage of MQTT ACLs |