Livrable 4 — Campus IoT (Groupe 3 FISA INFO 2024/2027)

Projet : Campus IoT (CESI Nancy – Bâtiment Orion)

Auteurs : BANIZETTE Matthieu, GACHENOT Antoine, PELLIZZARI Théo

Date : 06/02/2026

1) Dossier de compréhension et d'analyse (2–3 pages)

1.1 Reformulation du contexte et de ce qui pose question

Le projet **Campus IoT** vise à instrumenter les salles du bâtiment Orion afin de collecter des mesures environnementales (température, humidité, CO2, présence) et piloter des actionneurs (moteur, speaker) en temps réel. L'architecture actuelle s'appuie sur des capteurs/actuateurs connectés à une **gateway** locale, un **broker MQTT (Mosquitto)** et un **backend FastAPI** exposant des APIs et des WebSockets pour le front-end. Le protocole MQTT a été spécifié (arborescence de topics, QoS, retain, session persistante) et des tests de fiabilité/latence ont été menés.

Le cas d'usage principal est la **supervision environnementale** (qualité de l'air, confort thermique) et la **réaction automatique** (alertes sonores, ouverture/fermeture motorisée). Les utilisateurs visés sont les équipes techniques, les responsables de site et les occupants. Le système doit donc concilier **temps réel**, **disponibilité** et **fiabilité** tout en restant sobre en ressources.

Les questions clés portent sur la **fiabilité des données**, la **sécurité des échanges** et la **sobriété énergétique** dans un contexte Edge/IoT. Plus précisément :

- Comment garantir l'intégrité et l'authenticité des mesures et commandes ?
- Comment limiter les pertes de messages lors des micro-coupures réseau ?
- Comment réduire le trafic tout en conservant la réactivité en cas d'anomalie ?
- Comment tracer les actions et les événements critiques de manière robuste ?

Ces interrogations exigent une **architecture durable et sécurisée**, cohérente avec les contraintes IoT et les recommandations ETSI EN 303 645. Elles impliquent également de préciser les responsabilités entre **objet**, **gateway** et **backend**, et de définir un périmètre clair de **données critiques** (alertes CO2, commandes d'actuateurs, accès administratifs).

1.2 Enjeux techniques, énergétiques et de confiance

Enjeux techniques

- **Fiabilité de transport** : garantir la livraison des messages sensibles (QoS 1 + sessions persistantes).
- **Interopérabilité** : topics normalisés, payloads cohérents, compatibilité capteurs/actuateurs.
- **Scalabilité** : capacité à ajouter des salles ou des typologies de capteurs sans refonte.
- **Observabilité** : logs et métriques pour diagnostiquer latence, pertes, charge.
- **Résilience** : tolérance aux coupures courtes, reprise automatique et synchronisation via retain.
- **Qualité de données** : gestion des valeurs aberrantes, horodatage fiable, filtrage côté edge.

Enjeux énergétiques

- **Sobriété réseau** : limiter la fréquence d'émission, utiliser des modes adaptatifs (ex. BME280 en « fast » seulement en anomalie).
- **Sobriété côté capteurs** : minimiser les transmissions inutiles, privilégier les deltas et les heartbeats.
- **Empreinte infrastructure** : un broker local limite les allers-retours cloud et réduit la latence.
- **Compromis précision/sobriété** : fréquence adaptée par typologie de capteurs et niveau d'alerte.

Enjeux de confiance

- **Authentification forte** des objets et des utilisateurs (JWT, rôles, secrets).
- **Chiffrement des flux** pour éviter l'espionnage ou l'injection de commandes.
- **Intégrité et non-répudiation** des données (HMAC, horodatage, traçabilité).
- **Conformité aux bonnes pratiques** (ETSI EN 303 645 : mots de passe, rôles, mises à jour, durcissement).
- **Traçabilité** des décisions automatiques pour expliquer les actions (audit, responsabilité).

1.3 Vision globale des éléments à fiabiliser, sécuriser ou améliorer

1. **Fiabilisation des échanges MQTT**
   - Maintenir QoS 1, retain pour les mesures critiques, sessions persistantes pour la continuité.

- Définir des limites de débit et une stratégie de back-off en cas de congestion.
- Mettre en place un « last will » pour signaler l'indisponibilité d'un capteur ou d'une gateway.

2. **Sécurisation des communications**
   - Passage à **MQTTS** (TLS) entre gateway et broker.
   - Gestion de certificats (CA interne) et rotation des secrets.
   - Segmentation réseau (VLAN IoT), filtrage firewall et réduction de la surface exposée.
3. **Authentification et autorisation**
   - Authentification des objets via **certificats ou tokens** signés.
   - ACL sur les topics MQTT pour éviter les publications non autorisées.
   - Politique de moindre privilège pour les rôles applicatifs.
4. **Intégrité et validation**
   - HMAC-SHA256 + timestamp côté capteurs/gateway, vérification côté backend.
   - Détection d'anomalies (valeurs aberrantes, replays).
   - Normalisation des unités et contrôles de cohérence (seuils, variation maximale).
5. **Traçabilité**
   - Journalisation immuable des événements clés (alertes, commandes d'actuateurs, connexions).
   - Registre distribué ou journal append-only pour renforcer la preuve et l'audit.
   - Conservation adaptée aux exigences de conformité et d'investigation.
6. **Sobriété énergétique**
   - Rythmes d'émission adaptatifs, delta-based, heartbeat minimal.
   - Optimisation du trafic et limitation des duplications.
   - Mise en veille des capteurs non critiques hors horaires de forte occupation.

2) Proposition d'architecture IoT sécurisée et durable

2.1 Architecture IoT fonctionnelle (objets, communications, services)

Objets IoT

- Capteurs : BME280 (temp/hum), capteur CO2, capteur de présence (HC-SR04), potentiomètre.
- Actuateurs : moteur (ouverture/fermeture), speaker (alertes).
- Capteurs configurés pour des **seuils de confort** et un mode « alerte » déclenchant une fréquence plus élevée.

Gateway locale

- Agrégation des capteurs, pré-traitement (filtrage/deltas), signature HMAC.
- Routage vers le broker MQTT.
- Buffer local en cas de coupure, reprise avec ré-émission contrôlée.

Broker MQTT (Mosquitto)

- Topics structurés : campus/orion/{ROOM}/…
- QoS 1, retain sur capteurs, sessions persistantes.
- ACL par client, limitation de débit, LWT pour l'état de connexion.

Backend (FastAPI)

- Validation HMAC + timestamp.
- Gestion des alertes, anomalies, règles métier.
- APIs REST + WebSocket pour le front.
- Historisation des mesures, calcul d'indicateurs (moyennes, dérives, pics).

Frontend

- Supervision (tableaux de bord, alertes, historique).

Base de données

- Stockage des mesures, alertes, journaux d'audit.
- Indexation par salle/horodatage pour faciliter les analyses et audits.

2.2 Intégration des mécanismes de sécurité IoT

Authentification

- Objets : certificats X.509 ou tokens signés (JWT) avec durée de vie courte.
- Utilisateurs : JWT + rôles (admin/technician/manager/user).
- Processus d'enrôlement des objets (provisioning) et révocation des identités.

Chiffrement

- MQTTS (TLS 1.2+), certificats par gateway.
- HTTPS pour le backend, WSS pour le front.
- Paramètres TLS durcis (ciphers recommandés, désactivation des suites faibles).

Intégrité

- HMAC-SHA256 sur les messages capteurs et commandes.

- Validation stricte des timestamps et des signatures.
- Rejet des messages hors fenêtre temporelle et contrôle anti-replay.

Autorisation

- ACL MQTT : un capteur ne publie que sur ses topics.
- Backend : RBAC selon rôles.
- Traçabilité des actions administratives (qui, quoi, quand).

Durcissement

- Rotation des secrets, suppression des credentials par défaut.
- Mise à jour régulière des firmwares.
- Surveillance des vulnérabilités et plan de patching.

2.3 Traçabilité / registre distribué (quand pertinent)

Pour les événements critiques (alertes CO2, commandes d'ouverture/fermeture, actions administratives), un **registre append-only** est pertinent :

- **Option 1 : journal immuable local** (hash chaîné) stocké en base.
- **Option 2 : registre distribué léger** (par ex. Hyperledger Fabric ou un service de journalisation immuable) si besoin de preuve multi-parties.

Objectif : fournir une **preuve d'intégrité** et faciliter les audits, sans alourdir l'architecture pour les messages non critiques. Un schéma simple de chaînage peut être utilisé : chaque entrée contient le hash de l'entrée précédente, l'horodatage, l'identifiant de salle et l'événement (alerte/commande), ce qui permet une vérification rapide en audit.

3) Restitution synthétique et argumentée

3.1 Choix réalisés

- **MQTT QoS 1 + retain + sessions persistantes** : compromis fiabilité/latence adapté à l'Edge.
- **Rythmes d'émission adaptatifs** : sobriété énergétique en régime normal et réactivité en anomalie.
- **HMAC + timestamps** : intégrité et lutte contre la falsification/replay.
- **TLS (MQTTS + HTTPS/WSS)** : confidentialité et protection contre l'injection.
- **Journalisation immuable** : traçabilité des actions critiques.
- **Segmentation réseau et ACL** : confinement des flux et réduction du risque d'intrusion.

3.2 Bénéfices attendus

- **Fiabilité accrue** des mesures et commandes, même lors de micro-coupures.
- **Réduction du trafic** et meilleure autonomie des capteurs.
- **Confiance renforcée** grâce à l'authentification et l'intégrité des données.
- **Audit facilité** pour les événements sensibles.
- **Meilleure qualité de service** avec une latence stable et une reprise rapide.

3.3 Limites et risques

- **Complexité accrue** (gestion des certificats, ACL, rotation des clés).
- **Coût opérationnel** pour la maintenance de la sécurité.
- **Risque de dérive** si les règles de qualité de données ne sont pas suivies.
- **Risque opérationnel** en cas de mauvaise gestion des certificats ou de rotation de clés.

3.4 Cohérence globale

La proposition s'appuie sur les choix validés au Livrable 3 (MQTT, QoS 1, retain, sessions persistantes) et les complète par une **couche sécurité robuste**, tout en conservant la **sobriété énergétique**. L'architecture est évolutive, conforme aux bonnes pratiques ETSI, et cohérente avec les contraintes d'un campus IoT réel. Les mécanismes proposés restent pragmatiques et compatibles avec une montée en charge progressive (ajout de salles, nouveaux capteurs, nouveaux usages).