The First International Conference on Cross-Domain Security
in Distributed, Intelligent and Critical Systems (CROSS-SEC 2026)

Secure-by-Design Prototyping of an IoT Access-Control System

Oliver Vainikko[1], **Ulrich Norbisrath**[1], **Ruben Jubeh**[2]

[1] Department of Computer Science, University of Tartu, Estonia

[2] Faculty of Computer Science and Mathematics, OTH Regensburg, Germany

Presenter: Prof. Dr. Ruben Jubeh · ruben.jubeh@oth-regensburg.de

About the Presenter

Prof. Dr. Ruben Jubeh
Faculty of Computer Science and Mathematics, OTH Regensburg, Germany

Research focus: Internet of Things, Software Engineering, Edge AI, Embedded Machine Learning, and IoT security

Active collaboration with the **University of Tartu** (Estonia) on the IoTempower framework

1 The Prototyping–Security Tension

Rapid IoT prototyping in universities and industry training delivers **functional systems fast** — but routinely ships with insecure defaults that persist into final demonstrations.

Frameworks like **IoTempower**, Tasmota, and ESPHome abstract away networking and messaging — and, with it, security configuration.

> Security is *deferred* until "later" — and "later" never comes before the deadline.

Rapid Iteration

Declarative config, OTA updates, MQTT messaging — prototypes run in hours

Security Debt

Plaintext messaging, shared passwords, no device identity — left as-is

2 Research Questions

RQ1 How can a **secure-by-design architecture** for access control be integrated into prototyping workflows?

RQ2 Which security mechanisms can be introduced **in phases** to reduce developer friction?

RQ3 How can **collaborative threat-mapping** support security awareness in mixed-experience student teams?

RQ4 Which **framework-level extensions** meaningfully reduce recurring misconfigurations?

> Goal: Make secure operation the **expected end state** of educational prototypes — without a last-minute hardening sprint.

3 IoTempower: Educational IoT Framework

Open-source framework for **ESP8266/ESP32** rapid prototyping, used at several European universities.

Typical Setup

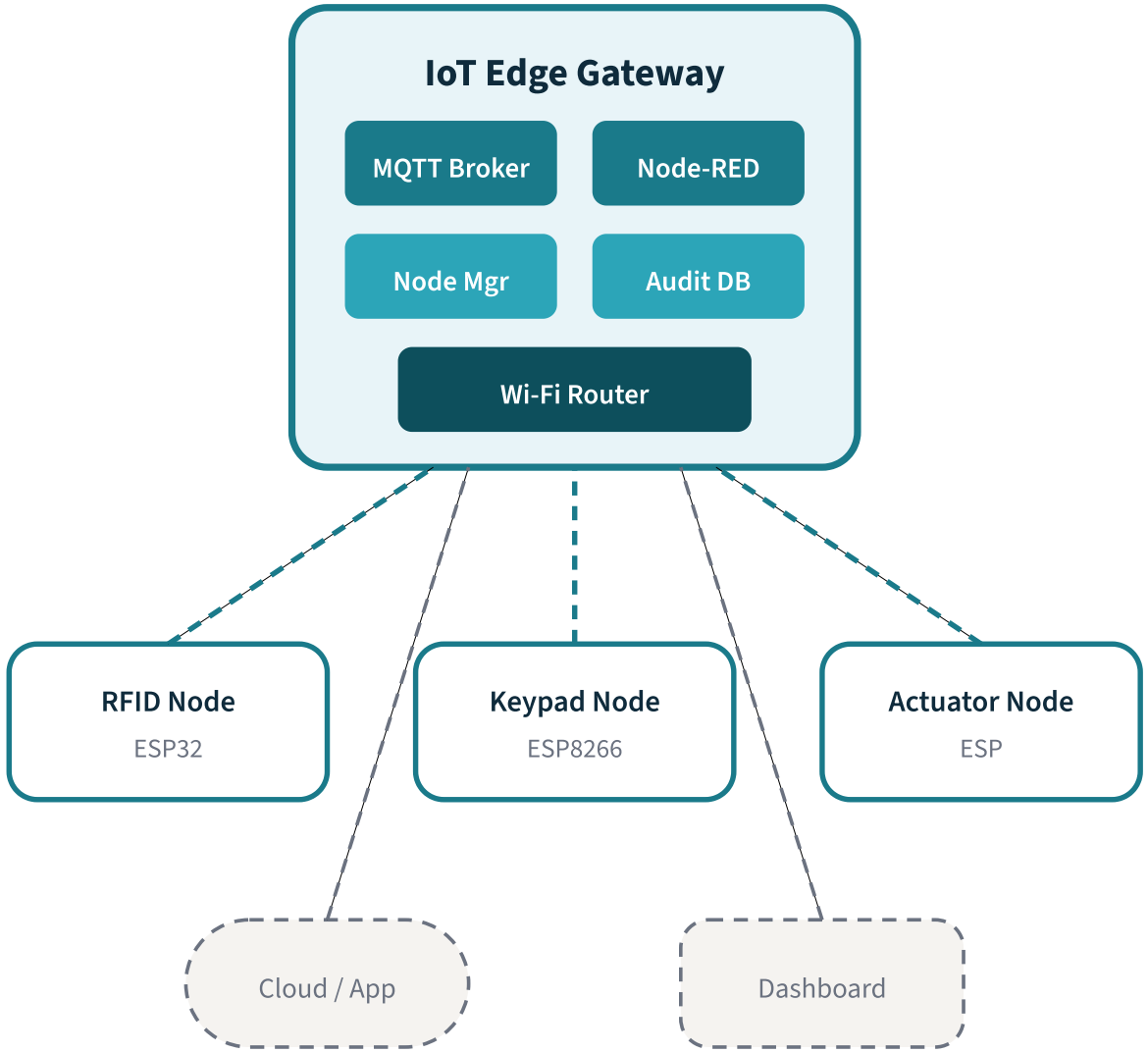
Gateway (Raspberry Pi / laptop) + dedicated Wi-Fi router + MQTT broker + Node-RED integration

Developer Experience

Declarative node configs → IoTempower handles networking, MQTT pub/sub, OTA flashing

Key Features

Device Management, flexible integration, works offline



4 Access-Control Use Case

Edge-local RFID/NFC door access control — suitable for a lab or office environment.

Trust boundary: Gateway = trusted infrastructure. Nodes = exposed (attacker may physically access them). MQTT broker = shared message bus — if you can reach it, you can observe and inject.

5 Insecure Prototype Baseline

Three recurring properties observed in typical classroom IoTempower deployments:

Plaintext MQTT

Port 1883 with weak or shared authentication — all messages visible on the network

Shared Identities

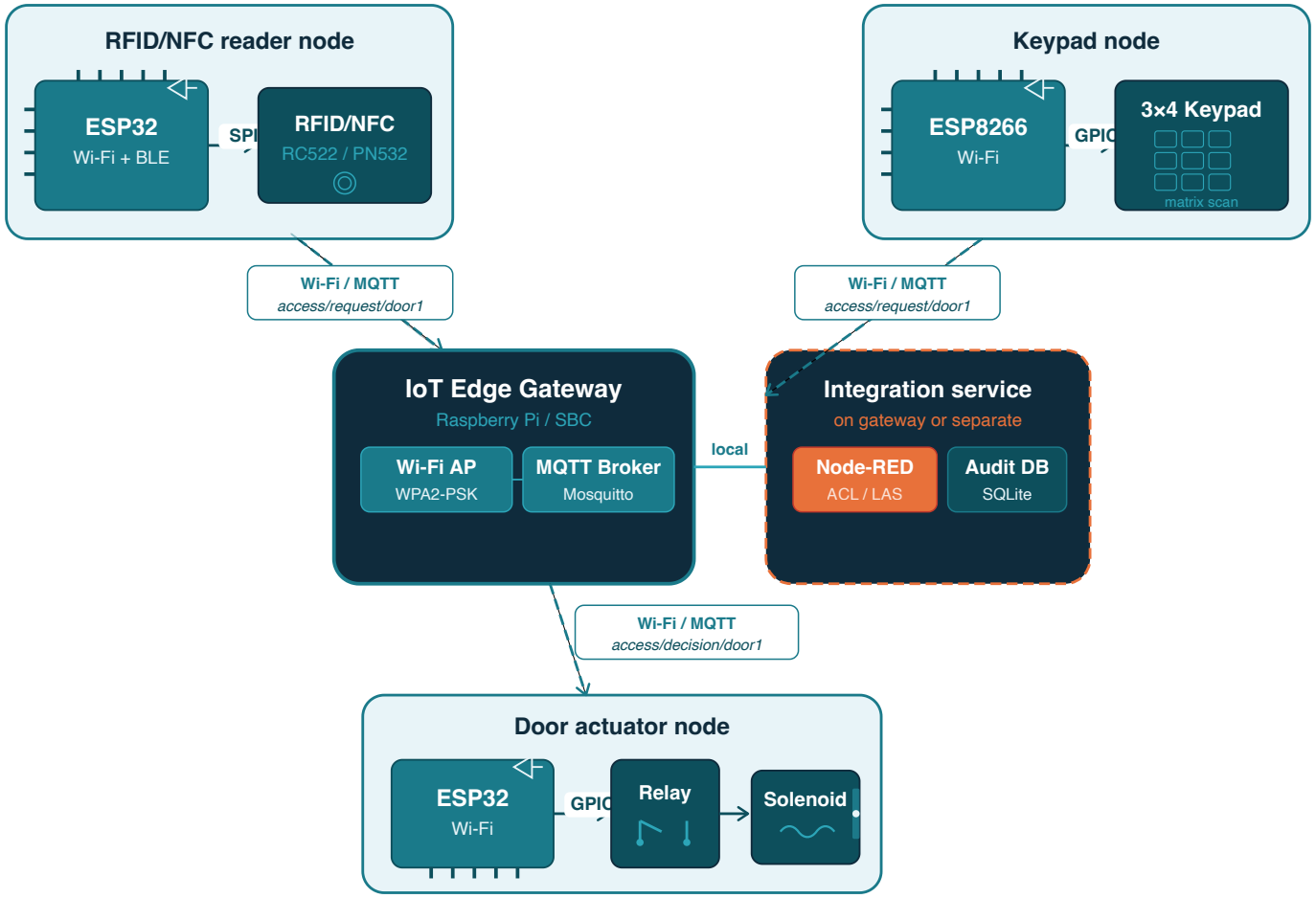
All nodes share the same Wi-Fi PSK, MQTT credentials, and OTA password

Weak OTA Protection

Updates protected only by a password hash; sensitive config embedded in firmware

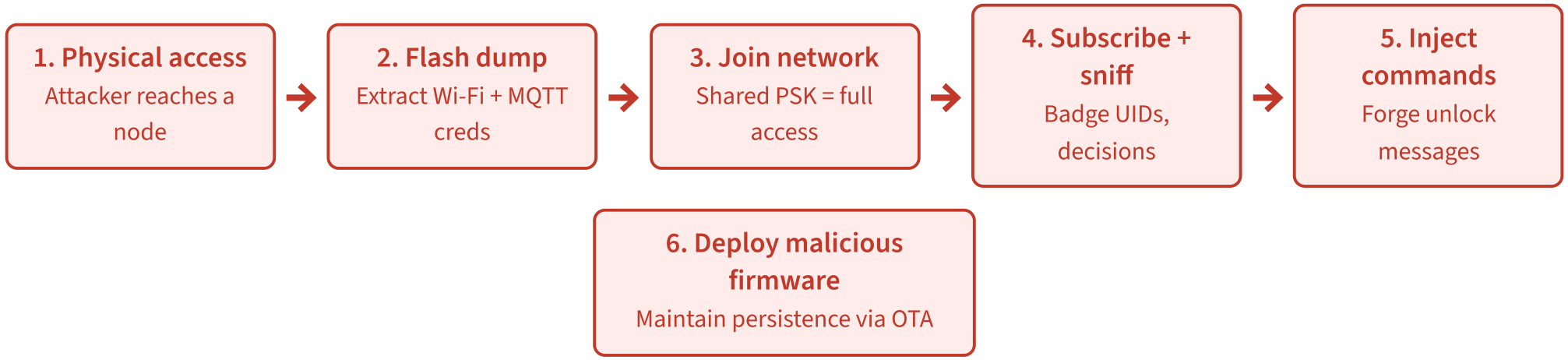
> IoTempower is **secure-capable** but not **secure-by-default** — TLS and ACLs are supported but rarely configured in time.

6 Access Control Reference Architecture



7 Attack Scenario: Single-Device Compromise

Shared secrets + plaintext messaging → one compromise cascades system-wide.



Root cause: One device breach = total breach when all nodes share the same credentials.

8 Assets & Adversary Actions

Node Credentials

Wi-Fi, MQTT, OTA, certs — gate access to network, broker, updates.
→ *Extract from flash; impersonate nodes; pivot laterally*

MQTT Topics

Request, grant/deny, logs — carry badge data and unlock commands.
→ *Eavesdrop; replay; inject forged grant messages*

Gateway Configs

Broker ACLs, LAS logic, keys — defines authorization and trust anchors.
→ *Modify allowlists/flows; steal keys; abuse defaults*

Firmware / OTA

Determines device behavior and may embed secrets in code.
→ *Upload malicious firmware; downgrade; backdoor*

Shared Wi-Fi

Same network for all devices and gateway — spoofing enables capture for replay. → *Eavesdrop, replay, spoof MAC*

9 Gap Analysis: Standards vs. Practice

Mapping baseline prototype against **ETSI EN 303 645** and **NISTIR 8259A** expectations:

Default Credentials

Standard: No universal default passwords.

Gap: All nodes share the same Wi-Fi PSK and MQTT creds.

Transport Security

Standard: Secure communications required.

Gap: MQTT on plaintext port 1883 is the norm.

Device Identity

Standard: Unique device identification.

Gap: Non-unique identities across the fleet.

Secure Updates

Standard: Protected update mechanisms.

Gap: OTA protected only by shared password hash.

> Not aiming for formal compliance — identifying **highest-leverage gaps** to close first.

Secure-by-Design Reference Architecture

Addressing baseline weaknesses with three core principles

10 Architecture: Four Pillars

Unique Device Identity

Each node gets a unique credential (X.509 client certificate + key). Broker authenticates per-device → enables individual revocation. Breaks "one breach = total breach."

TLS-Only MQTT

Broker accepts connections only over TLS; plaintext rejected. Mutual authentication prevents unauthorized clients even with Wi-Fi access.

Least-Privilege Authorization

Broker-side ACLs restrict each identity to specific topics and operations. Compromised reader can't publish unlock; compromised actuator can't subscribe to other doors.

OTA & Config Hygiene

Per-device update secrets (not fleet-wide). Firmware signing where possible. Gateway logs auth failures and denied publishes for auditing.

11 Unique Identity + TLS-Only MQTT

Per-Device Certificates

Each ESP node is provisioned with its own **X.509 client certificate and private key** during Phase 2. The MQTT broker authenticates based on these credentials.

Key Benefit

Compromise of one node does not yield credentials for any other node. Individual devices can be revoked without disrupting the fleet.

TLS Enforcement

The broker **rejects all plaintext connections** — port 1883 is disabled entirely. Mutual TLS ensures both broker and device verify each other.

Defense in Depth

Even if an attacker gains Wi-Fi access, they cannot connect to the broker without a valid client certificate.

12 Least-Privilege ACLs + OTA Hygiene

Topic Authorization

Broker-side ACLs map each device identity to a **constrained set of topics and operations** (publish vs. subscribe).

```
rfid_reader_door1 → PUB access/request/door1
actuator_door1   → SUB access/decision/door1
las_service     → SUB access/request/#
                  PUB access/decision/#
```

Topics are **explicit interfaces**, not "secret strings." Naming conventions reinforce role separation.

OTA Protection

Update secrets are **per-device** — not shared across the fleet. Firmware signing ensures only trusted binaries are accepted.

Fail-Secure Default

Classroom prototypes default to fail-secure: do not unlock on missing authorization. Real deployments may need different policies for life safety.

Audit Trail

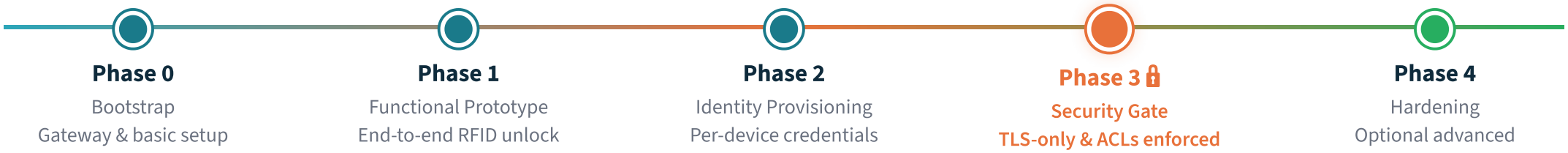
Gateway logs authentication failures and denied publishes — teachable evidence during exercises.

Phased Security Model

Aligning security upgrades with functional milestones

13 Phased Security Model: Overview

Security controls are layered incrementally, with a **mandatory security gate** at Phase 3.



> Phase 3 is a **precondition** for final demonstrations or grading — a system that "works" must also "work securely."

14 Phases 0–1: Bootstrap → Functional Prototype

Phase 0 — Bootstrap

Bring up gateway, MQTT broker, initial nodes. Temporary development secrets are acceptable.

Already enforced:

- Basic network segmentation
- Clear, consistent topic naming
- Avoid unstructured growth later

Phase 1 — Functional Prototype

End-to-end door workflow running: badge read → LAS decision → door unlocks.

Hygiene introduced:

- Separate request and decision topics
- Enable logging of access events
- Remove wildcard subscriptions

Friction is kept low — teams focus on **making the system work** while laying groundwork for later security layers.

15 Phases 2–3: Identity → Security Gate

Phase 2 — Identity Provisioning

Prepare for stronger guarantees without yet forcing all traffic into TLS.

- Generate per-device certificates/credentials
- Pre-create broker accounts
- Define ACL entries per device identity

Phase 3 — Security Gate 🔒

The mandatory checkpoint — insecure fallbacks are removed.

- MQTT switched to **TLS-only**
- ACLs **enforced** on broker
- Plaintext listeners removed
- Shared test credentials removed

→ Precondition for grading / demo

16 Phase 4: Optional Hardening

For teams with capacity or higher assurance goals — beyond the mandatory baseline.

OTA Firmware Signing

Only cryptographically signed binaries are accepted by devices, preventing malicious firmware uploads.

Credential Rotation

Procedures for periodic rotation of device certificates and keys — limits exposure window of compromised credentials.

Monitoring & Anomaly Detection

Hooks for detecting suspicious behavior at runtime — unusual topic access, repeated auth failures, unexpected traffic patterns.

17 Collaborative Threat Mapping

Peer-review exercise at the Phase 3 security gate — teams review each other's systems.

Review Bundle

Each team prepares:

- Architecture diagram
- Structured list of MQTT topics & roles
- Selected config excerpts (no private keys)
- Description of security controls implemented

Outcome

Vulnerability report with concrete findings + proposed mitigations. Original team must address issues or provide justified explanation before final assessment.

Process

1. Team A completes Phase 3 baseline
2. Prepares review bundle
3. Team B applies STRIDE-lite checklist
4. Submits vulnerability report
5. Team A addresses findings

18 STRIDE-Lite Checklist for IoT Access Control

S Spoofing

Can a rogue client impersonate a device or the LAS to publish unlock commands?

T Tampering

Can MQTT messages or allowlists be modified in transit or on the gateway?

R Repudiation

Are door events logged so actions can be attributed to identities?

I Info Disclosure

Do badge IDs, credentials, or configs leak over the network or in storage?

D Denial of Service

What happens if Wi-Fi is jammed or the broker is flooded — does the door fail-secure?

E Elev. of Privilege

Can a low-privilege node publish admin topics or bypass LAS decisions?

> Adapted from Microsoft's STRIDE — tailored to devices, topics, and trust boundaries rather than enterprise IT.

19 IoTempower Extensions for Secure Defaults

Making the **secure path the default** — reducing the effort of "doing the right thing."

Secure Project Scaffolding

Default broker config enables TLS, disables anonymous access, includes example ACL file reflecting role separation. Plaintext MQTT is no longer the starting point.

Credential Generation Utility

Single command creates a local CA, broker/server certs, and per-device client credentials with device-ID mapping. Eliminates the temptation of shared secrets.

Embedded Security Checklist

Structured checklist + threat-mapping worksheet aligned with development phases. Milestone reminders prompt teams at phase transitions.

Classroom Mode

Instructors can selectively enable temporary insecure settings for demonstrating attacks, while the default remains aligned with the secure baseline.

20 Preliminary Observations & Evaluation Plan

Observations to Date

Across prior course iterations, recurring issues consistently appeared: plaintext MQTT, shared credentials, over-permissive topic access. These motivated introducing the **Phase 3 security gate**.

Planned Evaluation

Upcoming course iteration: ~20 students across 5 teams.

Quantitative

Number and severity of vulnerabilities found in peer review; quality of implemented fixes — collected across multiple courses.

Qualitative

Surveys and interviews on perceived friction, security awareness, and usefulness of framework extensions.

21 Discussion & Limitations

Threat Coverage

Focuses on MQTT-related threats (spoofing, tampering, lateral movement). Does not fully mitigate DoS (broker overload, Wi-Fi jamming), supply-chain compromises, or sophisticated physical attacks. Intentional for pedagogical scope.

Peer Review Variability

Quality of threat mapping depends on team experience levels. STRIDE-lite checklists standardize the process, but uneven depth is possible. Requires instructor calibration and iterative checklist refinement.

Isolation ≠ Security

Even local-only deployments have exposed surfaces: Node-RED UI, MQTT, SSH, OTA port. Teams must document and verify their secure setup through measurements (traffic analysis, flash extraction tests).

Usability–Enforcement Balance

The phased model defers resource-intensive tasks until the system is stable, while ensuring insecure fallbacks cannot persist into final demonstrations.

22 Conclusion & Future Work

Key Takeaway

Secure operation can become the **expected end state** of educational IoT prototypes — without sacrificing development speed. The integration of secure defaults, a phased security gate, and collaborative threat mapping makes this practical for novice teams.

Contributions

- Secure-by-design reference architecture for edge-local RFID access control
- Phased security model with mandatory gate before deployment
- STRIDE-lite collaborative threat-mapping exercise for student teams
- Lightweight IoTempower extensions encoding secure defaults

Future Work

- Empirical evaluation at scale across multiple course iterations
- Tighter per-device credential provisioning and revocation support
- Signed firmware update integration
- Upstream open-source contribution into IoTempower core framework

Thank You

Questions & Discussion

Oliver Vainikko · Ulrich Norbisrath · Ruben Jubeh

{oliver.vainikko | ulrich.norbisrath}@ut.ee · ruben.jubeh@oth-regensburg.de

University of Tartu · OTH Regensburg

github.com/iotempire/iotempower