Over-the-Air Update System for Microcontrollers
Full Metal Update
Team: Lerzan Cengiz, Connor Kafka, Anusha Kamat
Industry Mentors: Cedric Vincent, Adrien Leravat

PROBLEM STATEMENT
In today’s world, there are so many active devices within the Internet of Things (IoT) field. We need a mechanism to constantly update, upgrade, and maintain these devices. We need an automated process that can be initiated from a single location to simplify device management and provisioning.

REQUIREMENTS
- IOT devices need to be managed and updated remotely.
- Provide security updates.
- Add new features.
- Monitor device health.
- Reach devices located in remote areas.
- Reduce logistic delay for updates.

IMPLEMENTATION
- Upload multiple binary files.
- Schedule update rollouts to many devices.
- Monitor device status.
- Device usage and uptime.
- Command updates.
- Share status with server.
- Apply an update.
- Run customer application code.

TECHNOLOGIES
- Eclipse Callimorgen COAP/HTTP Proxy
- Eclipse Leshan LW/M2M server
- Zephyr RTOS LW/M2M library
- Foundry ID LW/M2M client implementation

FUNCTIONAL DESCRIPTION
- Hawkit interacts with the front-end GUI.
- Hawkit stores update files.
- Hawkit provides a REST API for Leshan to poll.
- The device management server polls the update management server for available updates and forwards the download link to the microcontroller.
- The proxy translates COAP requests to HTTP requests.
- Ethernet connectivity
- Bootloader configured for image swap mechanism.

FUTURE WORK
- Send only a difference file in case of small changes. Patch the difference file to already present firmware in the microcontroller and create updated version.
- Reduces transmission bandwidth.
- Reduces power consumption.
- Send the firmware using the Transfer Layer Security (TLS) protocol for secure binary file transfers.

RESULTS
- ZephyrOS runs on the microcontroller, which has a secure bootloader in place.
- The client software runs on the microcontroller and checks if a valid firmware update is received from a known source.
- Once the client can determine that there is a firmware update, the client sends a COAP request, which is translated into an HTTP request to the server to request and download the update image.
- The server runs and manages the deployment of the firmware image to the appropriate client device.
- The update management application (Hawkit) provides a GUI to queue the firmware updates for various client devices.
- Lightweight Machine-to-Machine (LWM2M) protocol is used to communicate.

CONCLUSION
- We were able to send firmware images successfully over the air to microcontrollers.
- The GUI displays which devices are attached to the system.
- The server monitors resources on the devices and enables the user to execute a firmware update remotely.
- The client can send a request to retrieve the update image.
- Client GET requests are translated via a proxy to be compatible with an HTTP server.
- The client verifies the source of the incoming firmware image and its validity.
- The secure bootloader enables rollback mechanism in case of a faulty firmware image.

REFERENCES