Performance Evaluation of the CoAP Protocol with Security Support for IoT Environments

Victor Seoane, Florina Almenares, Celeste Campo ,
Carlos Garcia-Rubio

uc3m | Universidad Carlos III de Madrid
Departamento de Ingeniería Telemática

Index

- Introduction
- Related Work
- Methodology
- CoAP analysis
- Conclusions and future work

Index

- **Introduction**
- Related Work
- Methodology
- CoAP analysis
- Conclusions and future work

Introduction

- Internet of Things (IoT) revolution.
  - Connecting everything to the network.
  - Challenges:
    - Handling a huge number of connections while assuring the Quality of Service (QoS).
    - Constrained devices. Typically, power and storage constraints.

- Lightweight application protocols are designed to face these challenges:

  → **CoAP** (Constrained Application Protocol).

Introduction

- Securing IoT communications
  - Threats: message forgery, tampering or eavesdropping.
  - IoT scenarios may include critical systems or carry sensitive data.

- Recommendations from **RFC 7925** to secure IoT communications.
  - Network overhead and consumption increase.
  - Security and performance trade-off.

Introduction

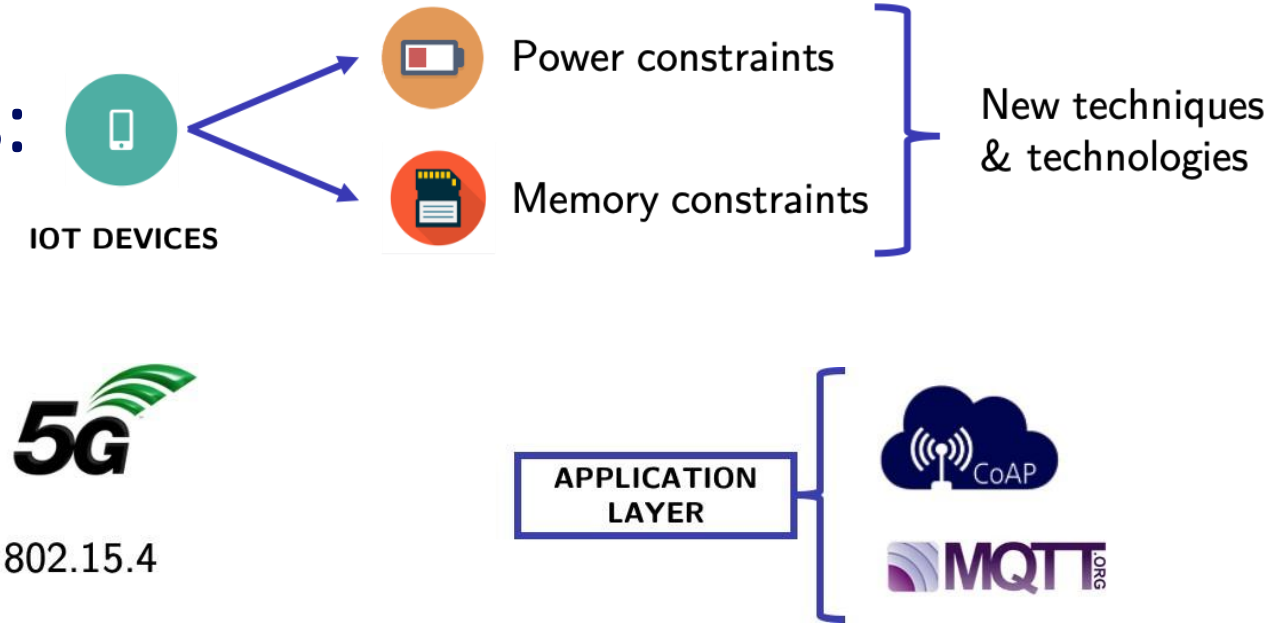
- Goals:
  - Latency, consumption and bandwidth analysis for CoAP.
    - Realistic network scenario.
    - Open-source implementations.
  - Study of performance degradation with security support.

Index

- Introduction
- **Related Work**
- Methodology
- CoAP analysis
- Conclusions and future work

Related work

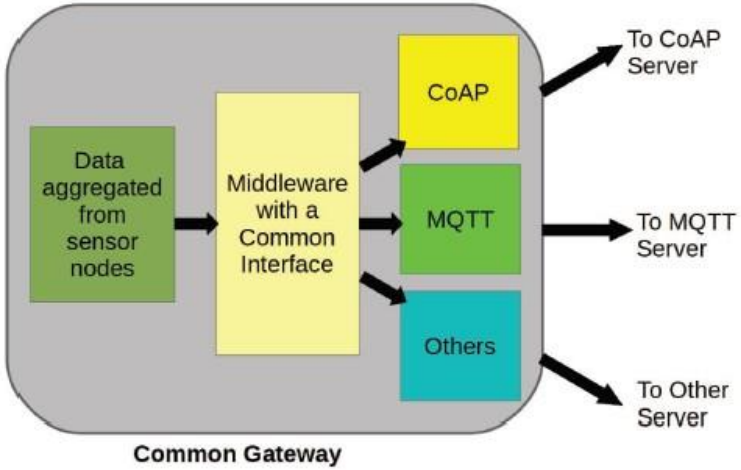
- Internet of Things:



  – Application Layer is crucial for:
    - Reliability.
    - Quality of Service (QoS).
    - Logical architecture of underlying network.

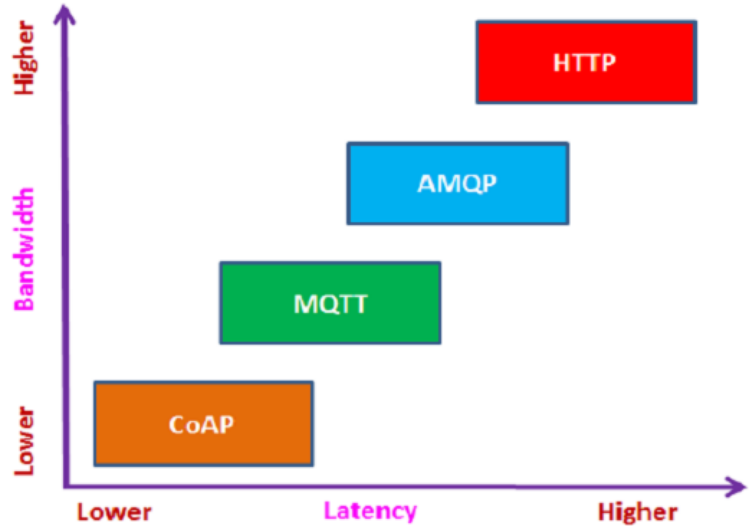
Related work

- "Performance Evaluation of MQTT and CoAP via a Common Middleware" (Thangavel et.al, 2014).
  - √ Complete hardware description
  - √ Lossy environments testing
  - × Reliability-performance trade-off
  - × Security-performance trade-off



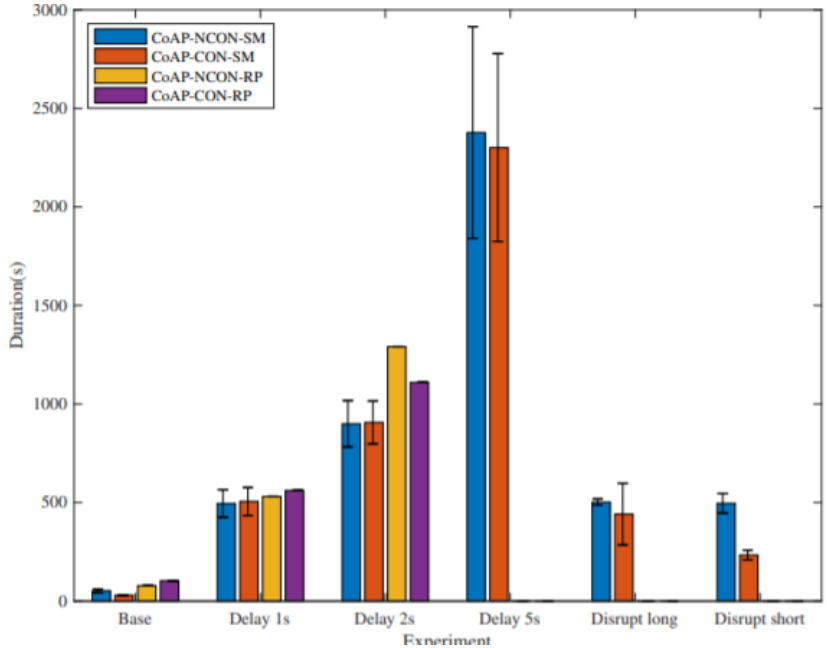
Related work

- "Choice of Effective Messaging Protocols for IoT Systems: MQTT, CoAP, AMQP and HTTP" (Naik, 2017).
  - √ Good guide for choosing
  - √ Many indicators comparisons
  - × Virtual hardware scenario
  - × Lossless scenarios



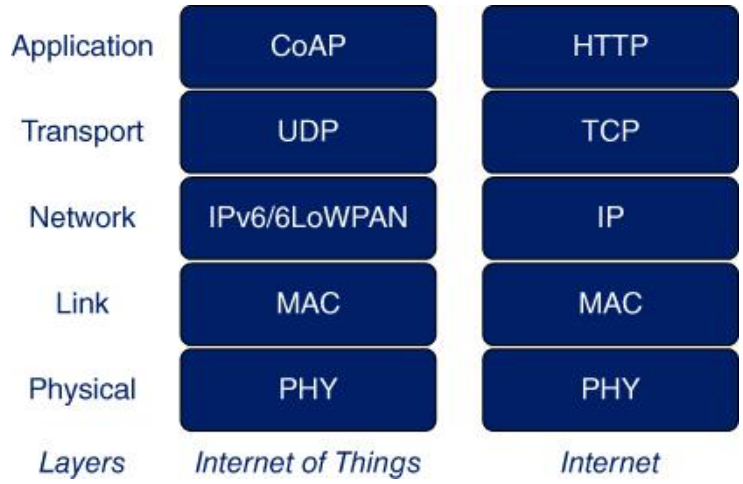
Related work

- "Robustness of IoT Application Protocols to Network Impairments" (Liri, 2018).
  - √ analyzes four protocols
  - √ experiments run using Raspberry-Pis
  - × not considers the effects of different security mechanisms



CoAP overview

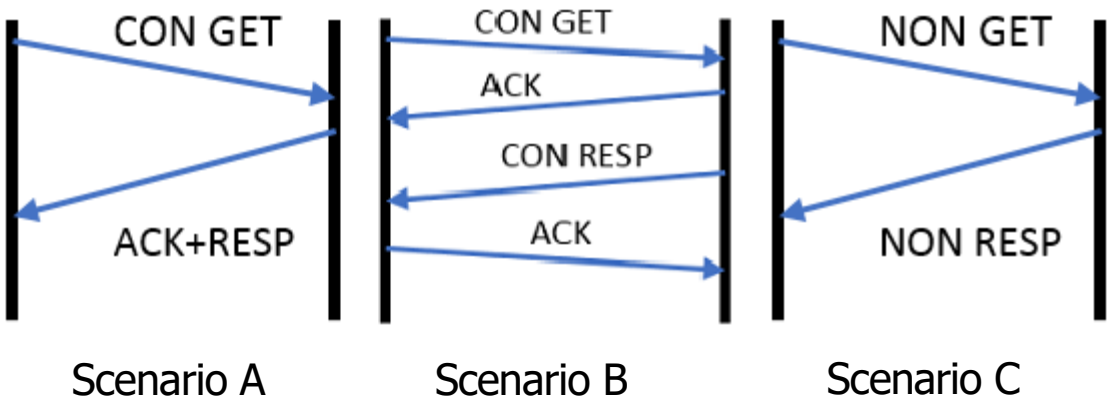
- **Constrained Application Protocol (CoAP)**
  - RFC 7252 (June 2014)
  - Important extensions in various stages of standardization.
  - Request-response architecture.
    - based on HTTP REST.
    - Conversion straightforward.



- **Bound to UDP** (not TCP) by default.
  - This makes it more suitable for the IoT applications.
  - Unicast and multicast support.
- **Low overhead**.
  - Header: 4 bytes + variable.

CoAP overview

- **Reliability** mechanisms.
  - CoAP messages may arrive out of order, appear duplicated, or go missing without notice.
  - For this reason, CoAP implements a lightweight reliability mechanism.
  - Three possible reliability modes of operation:



CoAP overview

- **Security Support** → Optional DTLS (Datagram Transport Layer Security) over UDP (RFC 6347).
  - Four security modes.
    - NoSec.
    - PreSharedKey (PSK)
    - RawPublicKey.
    - Certifica
- **Implementation**
  - DTLS: OpenSSL.
  - CoAP: Erbium, microcoap, libcoap, cantcoap, californium.

CoAP overview

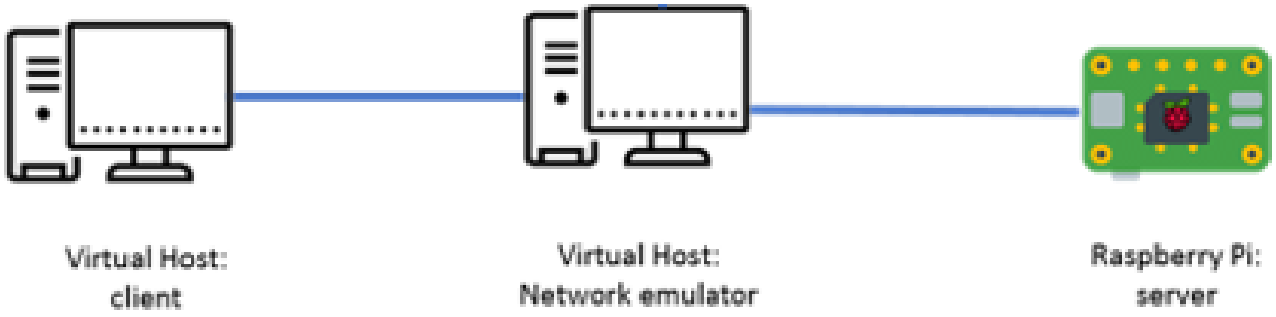
- Pros and cons:
  - √ ↓ Delay and power consumption.
  - √ End-to-end principle.
  - √ Asynchronous.
  - × Unreliability.
  - × DTLS overhead.

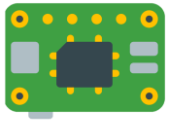
Index

- Introduction
- Related Work
- **Methodology**
- CoAP analysis
- Conclusions and future work

Scenario setup

- Realistic network scenario.



  – "Constrained" device → 
  – Real purpose: retrieve temperature from sensor
  – Network conditions → NetEm

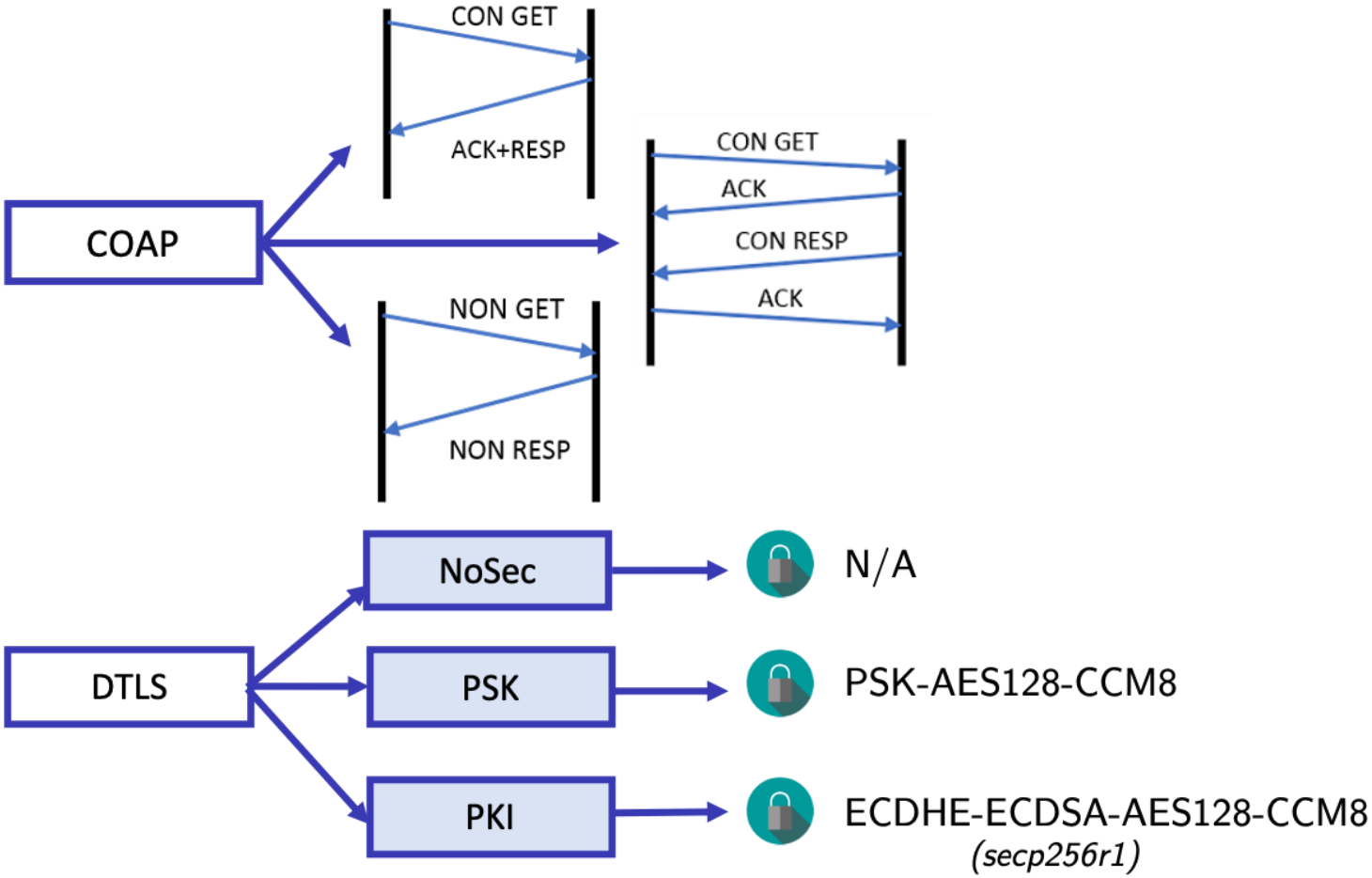
Scenario setup

- DTLS decryption.
  - Necessary to match requests and responses. Ideas:
    - × SSLKEYLOGFILE.
    - √ Intercepting the SSL library.
- Consumption measurement
  - CPU cycles → PERF.
  - Number of packets.

Implementation

- libcoap modifications
  - Send temperature from sensor.
  - Specify the cipher-suites (RFC-7925).

Implementation



- PKI mode includes a self-signed server X509 certificate

Implementation

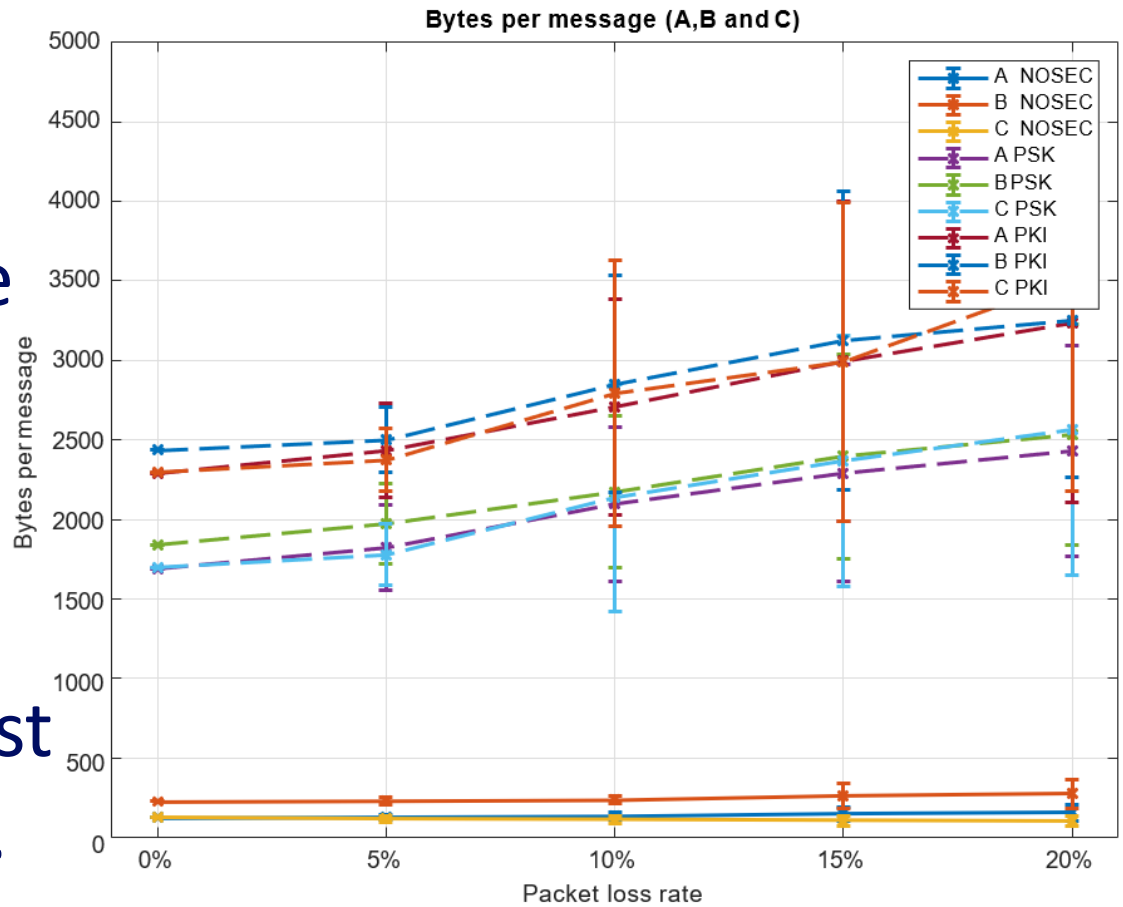
- 9 possible combinations CoAP-DTLS modes.
- 5 different packet loss rates (0%,5%,10%,15% and 20%).
- 45 simulations of 500 requests each.

Index

- Introduction
- Related Work
- Methodology
- **CoAP analysis**
- Conclusions and future work

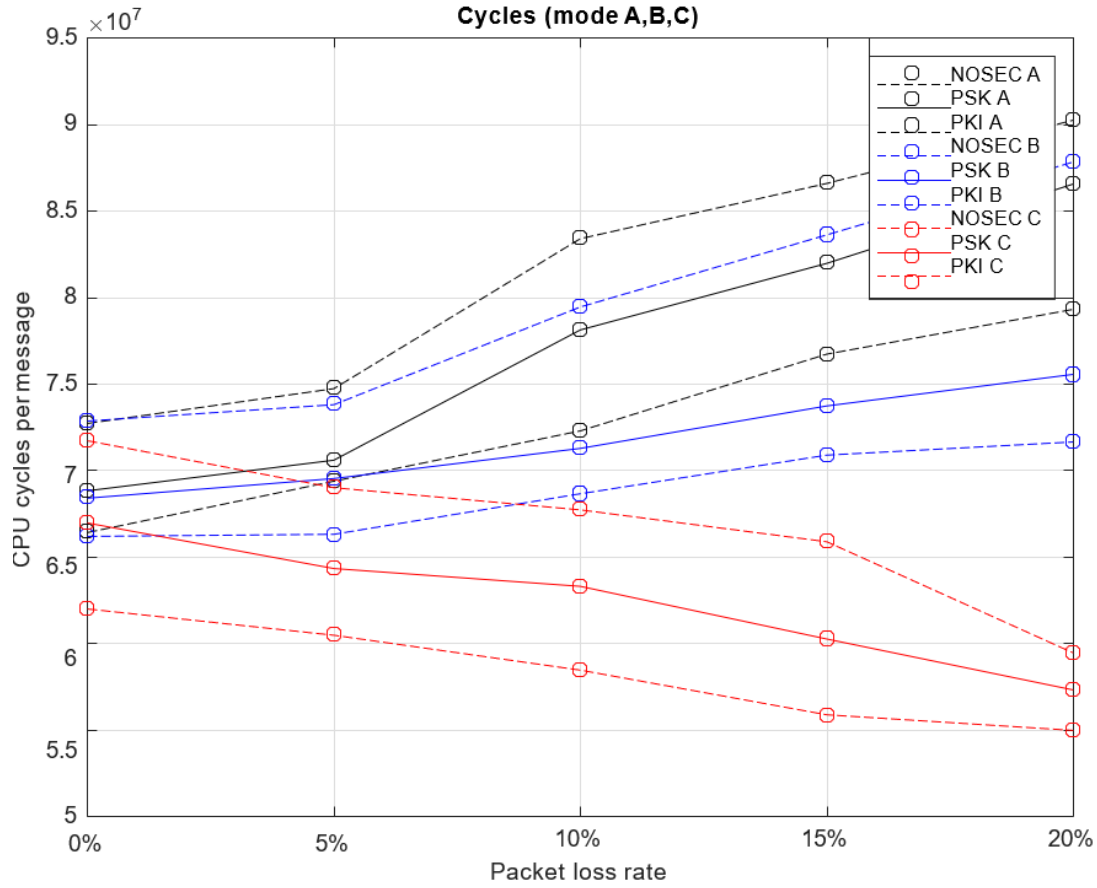
Performance results

- Bandwidth consumption:
  - Modes A and C behave similarly.
    Mode B demands more
  - PSK and PKI modes more adversely affected by losses
  - PKI ciphering is the most bandwidth demanding.



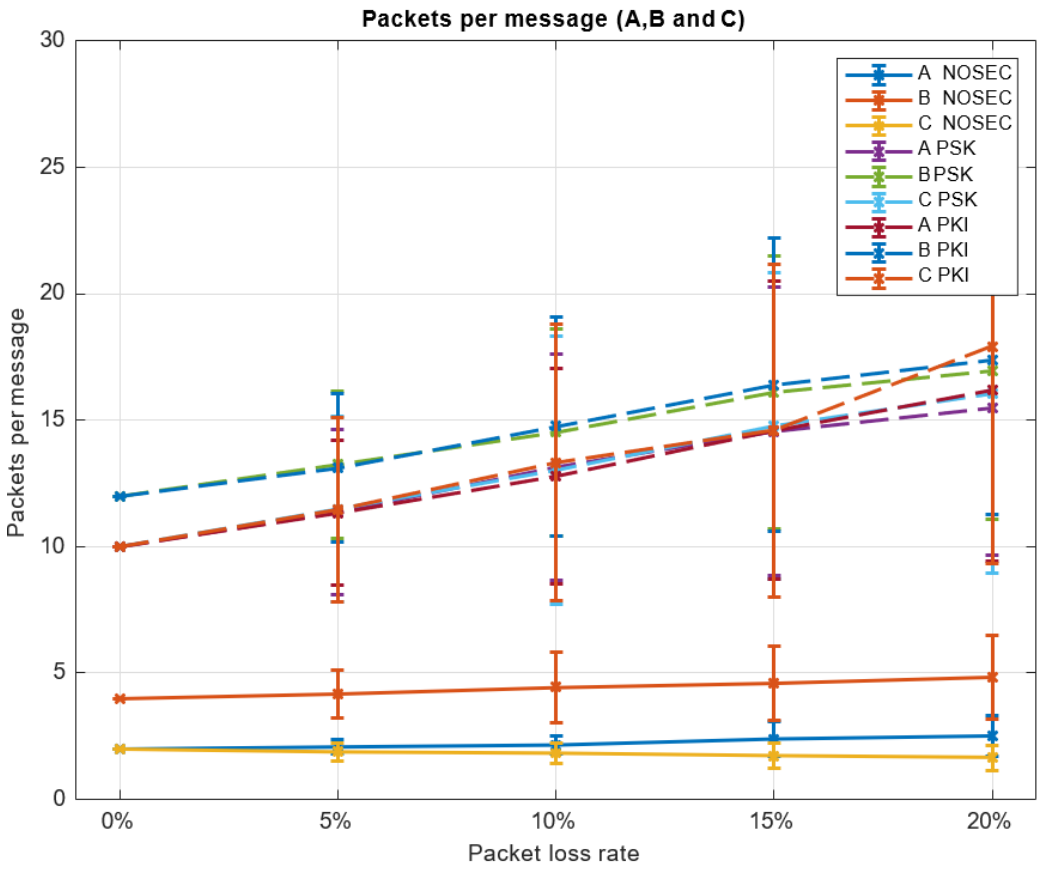
Performance results

- CPU cycles:
  - Immediate response requires additional effort on server (mode A).
  - Mode C is unreliable → lost packets are not processed.
  - Ciphering significantly increases consumption (*PKI > PSK > NoSec*)



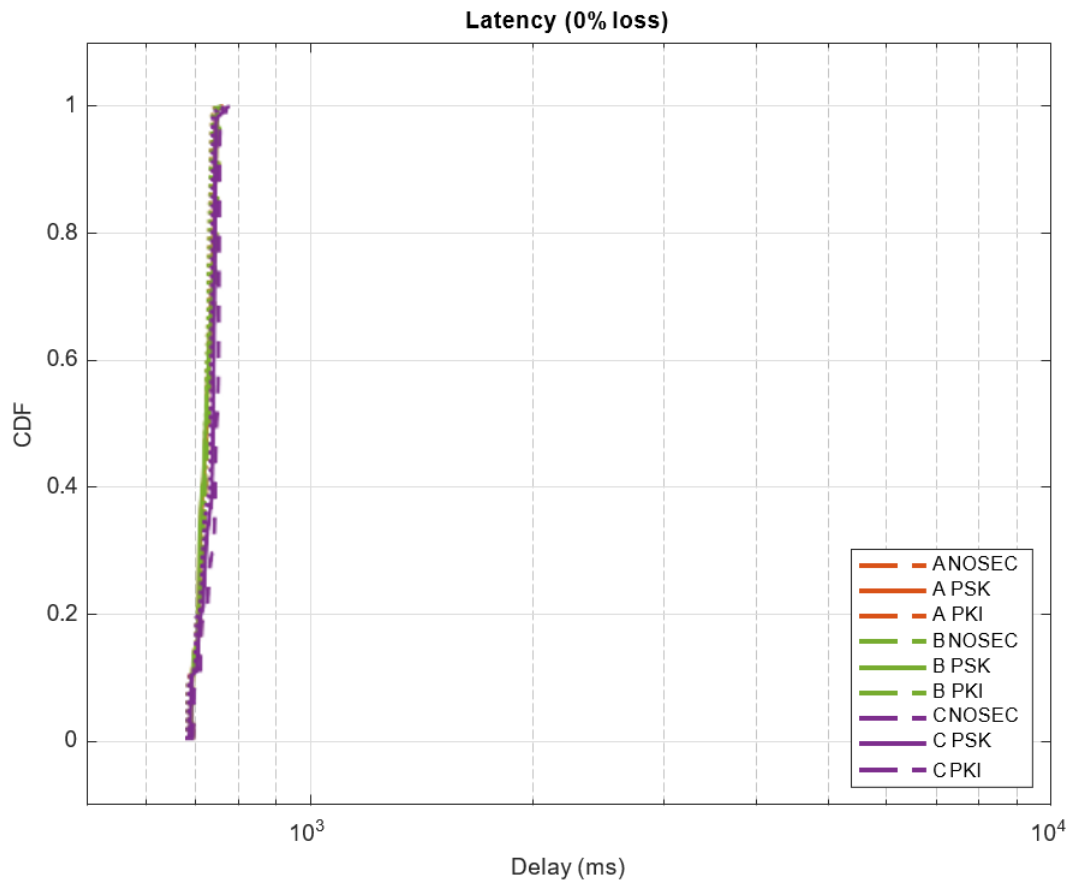
Performance results

- Number of packets:
  - Significant increase when applying ciphering (DTLS overhead).
  - Mode B stands out from the rest of modes.
  - PSK and PKI behave similarly.



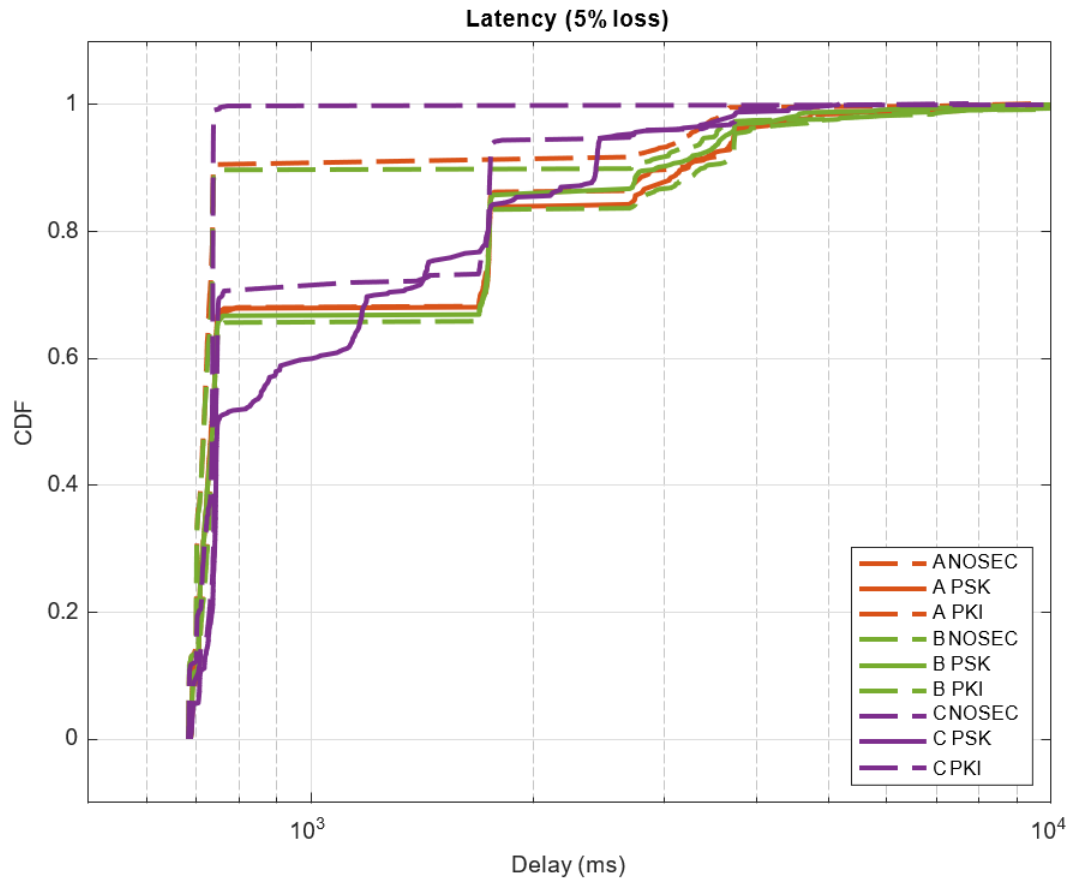
Performance results

- Latency (0% loss).



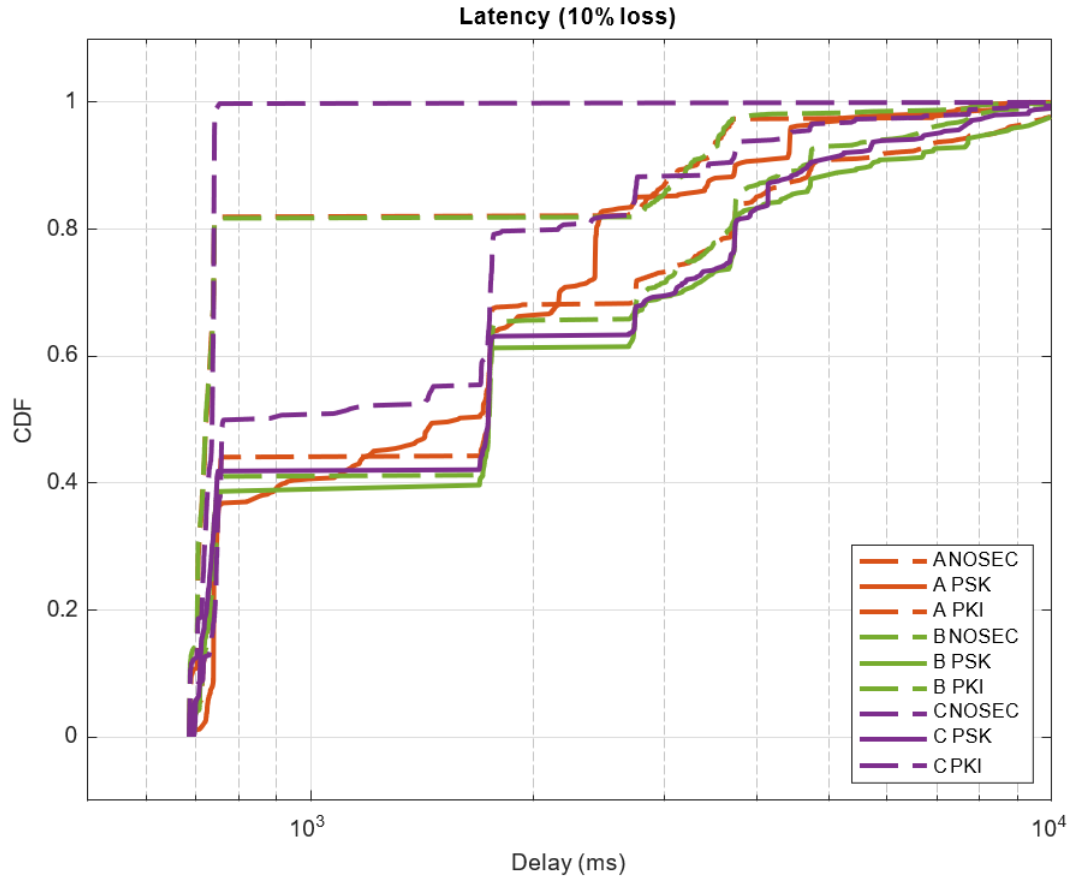
Performance results

- Latency (5% loss).



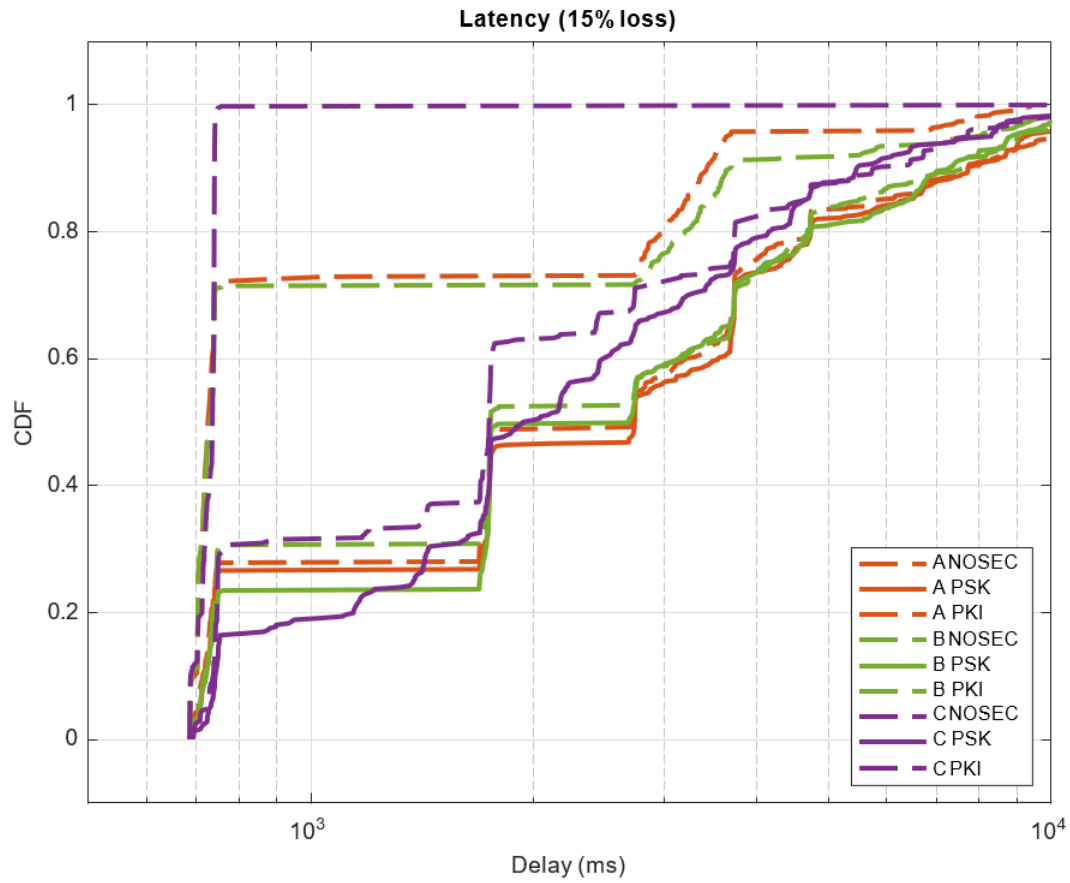
Performance results

- Latency (10% loss).



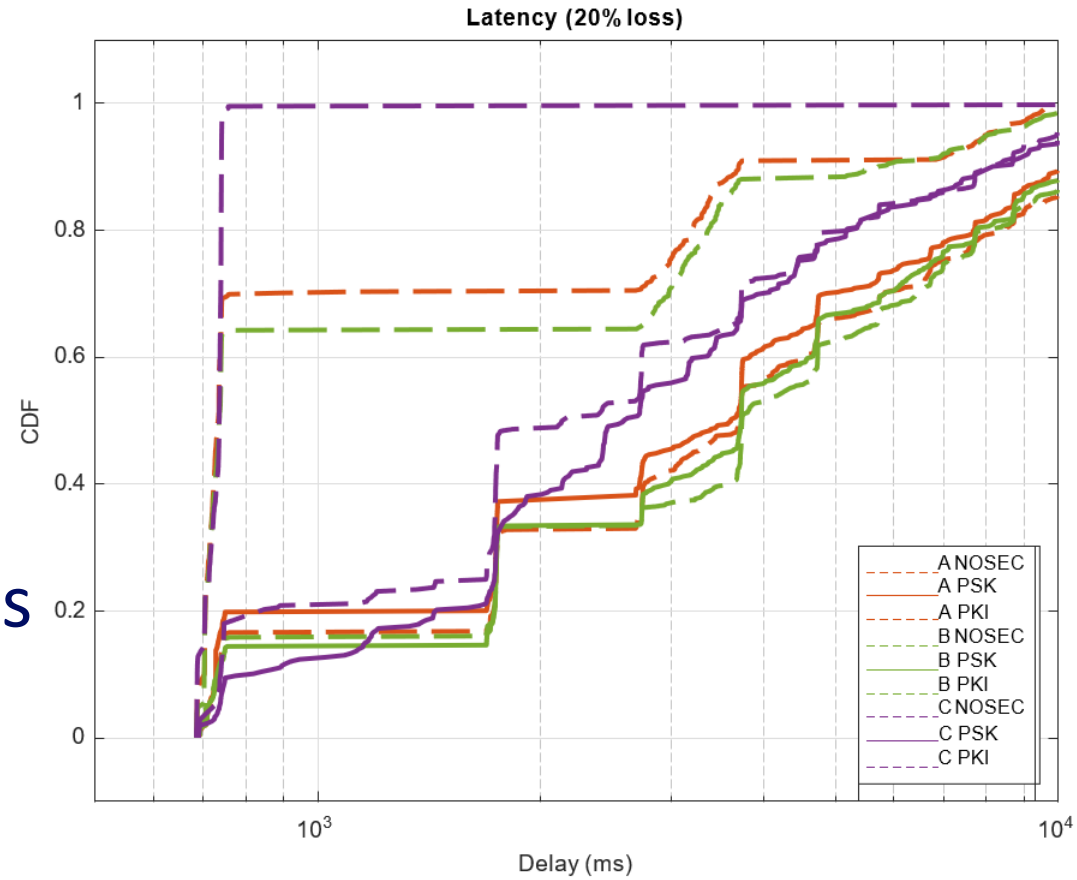
Performance results

- Latency (15% loss).



Performance results

- Latency (20% loss).
  - In lossless network, effect of ciphering is despicable (≈ 1%).
  - Infinite waits of mode C are not considered (lower effect).
  - Ciphered communications are more adversely affected by losses.



Index

- Introduction
- Related Work
- Methodology
- CoAP analysis
- **Conclusions and future work**

Conclusions

- Secure communications, i.e., ciphering methods, increases significantly:
  - the bandwidth consumption (more than 1000% in most cases).
  - CPU cycles (about 3.5% in PSK and 11.5% in PKI).
- But in turn high security level is guaranteed.
- In lossless networks, the effect of the ciphering modes on the latency is almost despicable (about 1%).
- But with higher loss rates secure communications are more adversely affected than no-secured ones.

Future work

- Will estimate the power consumption of the Raspberry-Pi through power models or using specific hardware.
- In addition, we are also assessing MQTT performance in order to compare results of both protocols.
  - Alternative cipher-suites
  - Version for sensor networks of MQTT (MQTT-SN)

Thanks you for your attention

Any questions?