

CLOUD COMPUTING 2020

11th International Conference on Cloud Computing, GRIDs, and Virtualization

An IoT Crypto Gateway for Resource-Constrained IoT Devices

Ahmed Alqattaa  
Daniel Loebenberger 

OTH Amberg-Weiden, Germany



📍 Nice, France

📅 October, 2020

1 Introduction and Background

2 IoT crypto Gateway

3 Conclusion

1

Introduction & Background

👍 IoT Challenges

👍 QUIC Introduction

👍 QUIC Layers

👍 QUIC vs DTLS

👍 HTTP vs MQTT

IoT resource-constrained is a wide concept, which faces many challenges:

- limited energy
- limited memory and bandwidth
- communication latency
- communication security
- achieving of security goals: authentication, confidentiality and integrity
- computations
- patching
- etc . . .

IoT resource-constrained is a wide concept, which faces many challenges:

- limited energy
- limited memory and bandwidth
- communication latency
- communication security
- achieving of security goals: authentication, confidentiality and integrity
- computations
- patching
- etc . . .

Using QUIC, the developed IoT gateway aims to:

- make it cheap
- keep it secure

👍 QUIC, TCP vs UDP



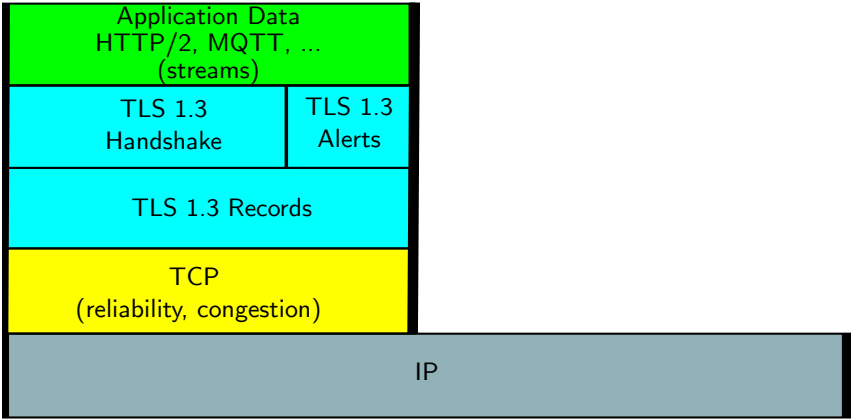


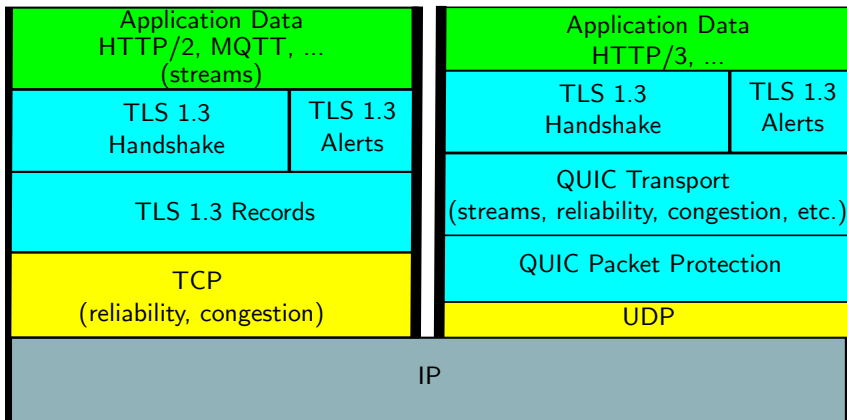
QUIC (Quick UDP Internet Connections):

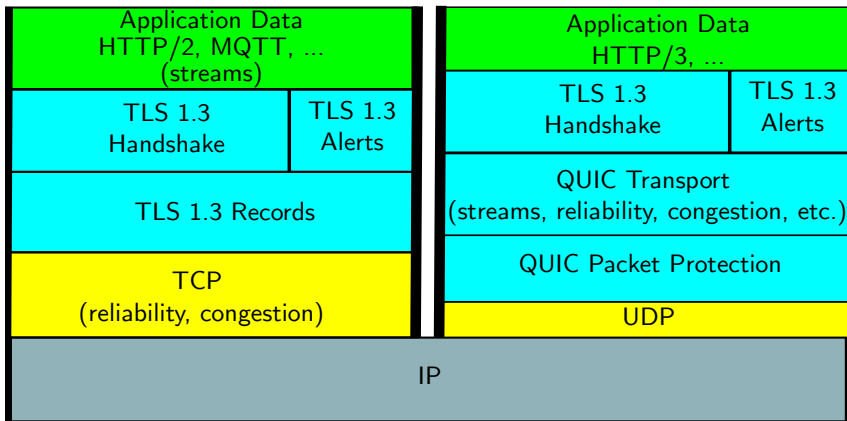
- UDP-based multiplexed and secure general-purpose transport protocol
- Stream multiplexing
- Stream- and connection-level flow control
- Low-latency connection establishment
- Authenticated and encrypted header and payload using TLS 1.3

- In 2013, QUIC was announced by Jim Roskind at Google.
- In 2015, QUIC was submitted for standardization to the IETF.
- In 2016, the IETF established a working group of QUIC.
- Until now, the IETF is still working on developing QUIC.









At a high level, there are two main interactions between the TLS and QUIC components:

- TLS sends/receives messages via QUIC providing a reliable stream abstraction to TLS.
- TLS provides a series of updates to QUIC, including
 - new packet protection keys to install
 - state changes such as handshake completion, the server certificate, etc. ...









- DTLS stands for Datagram Transport Layer Security protocol.
- It provides communications privacy for UDP.
- DTLS is based on TLS.

So ...

- Why do we need QUIC when we have DTLS?
- What difference does it make for the IoT devices?

DTLS vs QUIC (TLS) - - -> Full Handshake

DTLS-Client

DTLS-Server

ClientHello

...

---1--->

<---2---

HelloRetryRequest
+ cookie

ClientHello

...

---3--->

+ cookie

ServerHello

...

<---4---

Finished
[Application Data*]

...

Finished

---5--->

[Application Data]

<---6---

[ACK]
[Application Data*]

[Application Data]

<----->

[Application Data]

👍 DTLS vs QUIC (TLS) - - -> Full Handshake

DTLS-Client

DTLS-Server

TLS-Client

TLS-Server

ClientHello

...

---1--->

<---2-----

HelloRetryRequest
+ cookie

ClientHello

...

+ cookie

---3--->

ServerHello

...

Finished

[Application Data*]

<---4-----

...

Finished

[Application Data]

-----5-->

<---6-----

[ACK]

[Application Data*]

[Application Data]

<----->

[Application Data]

ClientHello

...

---1--->

ServerHello

...

Finished

[Application Data*]

<---2-----

...

Finished

[Application Data]

-----3-->

<----->

[Application Data]

Protocol	HTTP	MQTT
Architecture	Request/ Response	PUBLISH/ SUBSCRIBE
Message Format	ASCII	binary
Message Size	large	small
Complexity	complex	simple
Service Levels	1	3
Distribution	one-to-one	one-to-many
SSL/TLS Support	Yes	No, but possible
QUIC	Work in Progress	No
Power Demand	high	low
Usage, e.g.	big data, browsers	small data, embedded systems

Protocol	HTTP	MQTT
Architecture	Request/ Response	PUBLISH/ SUBSCRIBE
Message Format	ASCII	binary
Message Size	large	small
Complexity	complex	simple
Service Levels	1	3
Distribution	one-to-one	one-to-many
SSL/TLS Support	Yes	No, but possible
QUIC	Work in Progress	No
Power Demand	high	low
Usage, e.g.	big data, browsers	small data, embedded systems

HTTP vs MQTT

Protocol	HTTP	MQTT
Architecture	Request/ Response	PUBLISH/ SUBSCRIBE
Message Format	ASCII	binary
Message Size	large	small
Complexity	complex	simple
Service Levels	1	3
Distribution	one-to-one	one-to-many
SSL/TLS Support	Yes	No, but possible
QUIC	Work in Progress	No
Power Demand	high	low
Usage, e.g.	big data, browsers	small data, embedded systems

Protocol	HTTP	MQTT
Architecture	Request/ Response	PUBLISH/ SUBSCRIBE
Message Format	ASCII	binary
Message Size	large	small
Complexity	complex	simple
Service Levels	1	3
Distribution	one-to-one	one-to-many
SSL/TLS Support	Yes	No, but possible
QUIC	Work in Progress	No
Power Demand	high	low
Usage, e.g.	big data, browsers	small data, embedded systems

2

IoT crypto Gateway

👍 Motivation

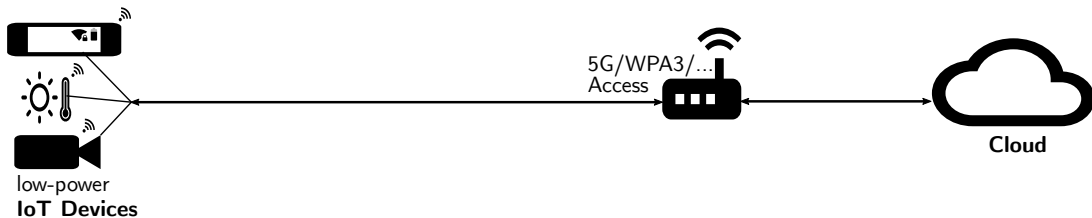
👍 Issue

👍 Contribution

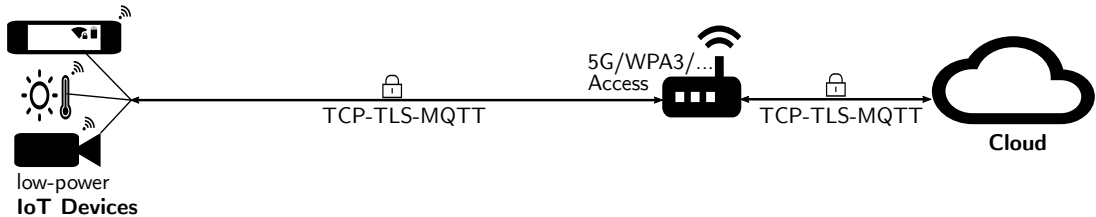
IoT resource-constrained has many challenges:

- limited energy
- limited memory and bandwidth
- communication latency
- communication security
- computations
- etc ...

Two scenarios as examples, which are used nowadays ...

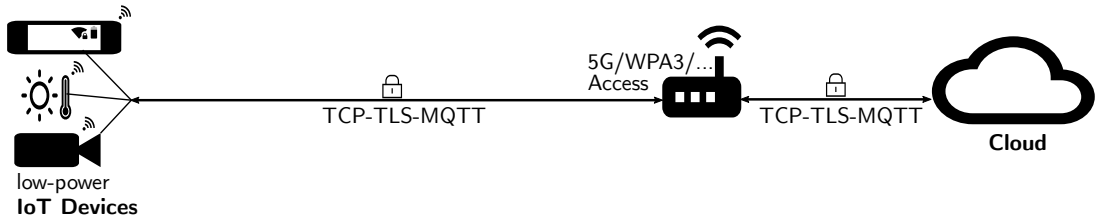


Problem in Performance



First scenario: the IoT devices directly secure the connection with the cloud.

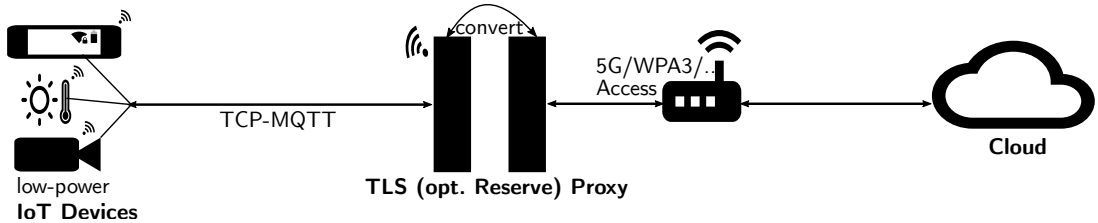
Problem in Performance



First scenario: the IoT devices directly secure the connection with the cloud.

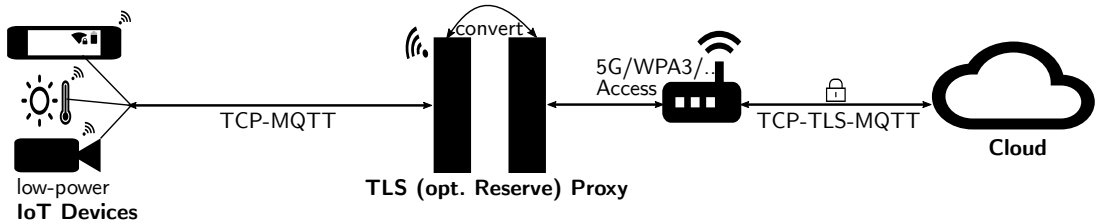
Problem: secure but not cheap

Problem in Security



Second scenario: a TLS (opt. reserve) proxy between the IoT devices and the cloud.

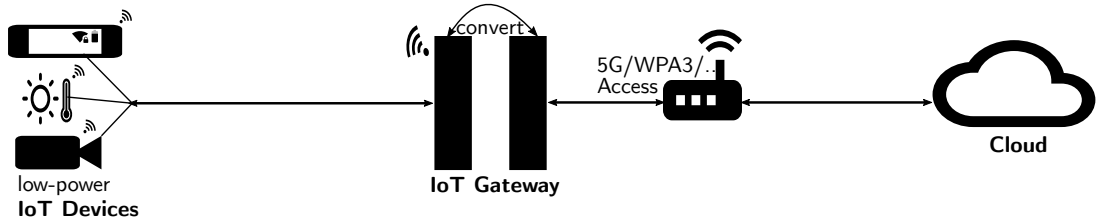
Problem in Security



Second scenario: a TLS (opt. reserve) proxy between the IoT devices and the cloud.

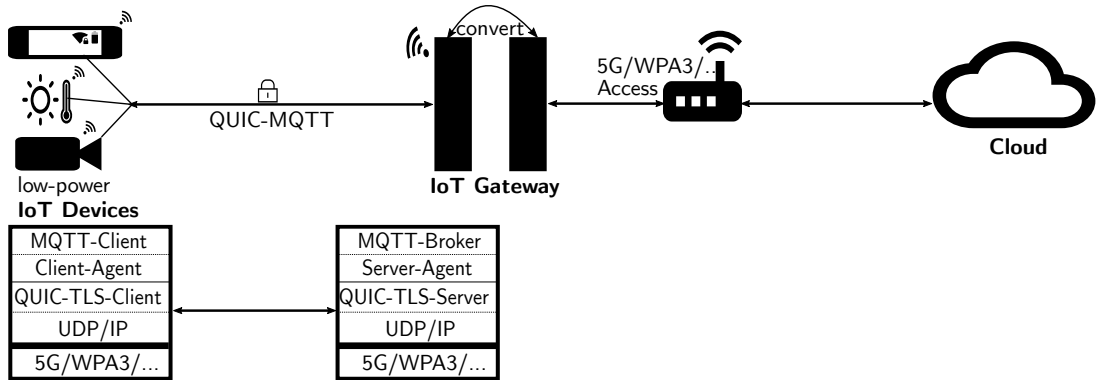
Problem: cheap but not secure enough

Contribution - - -> Introduction

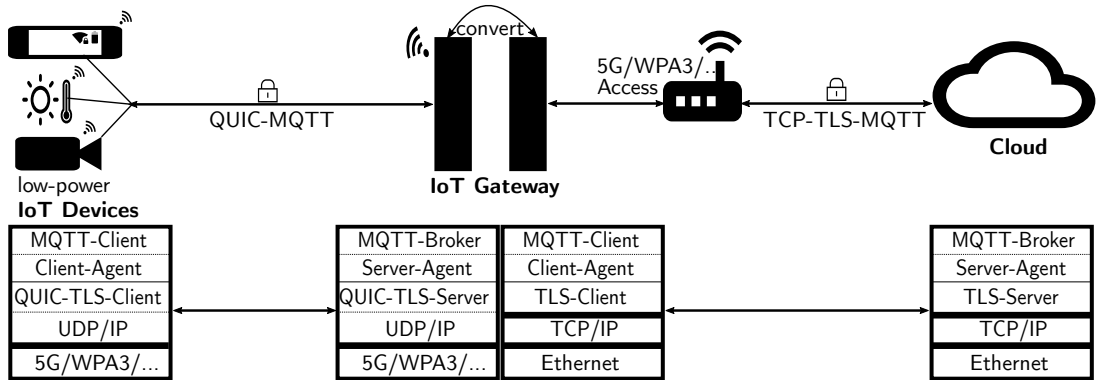


Contribution: establishing a connection between the IoT crypto gateway and the IoT device.

Contribution - - -> Communication Layers by Gateway

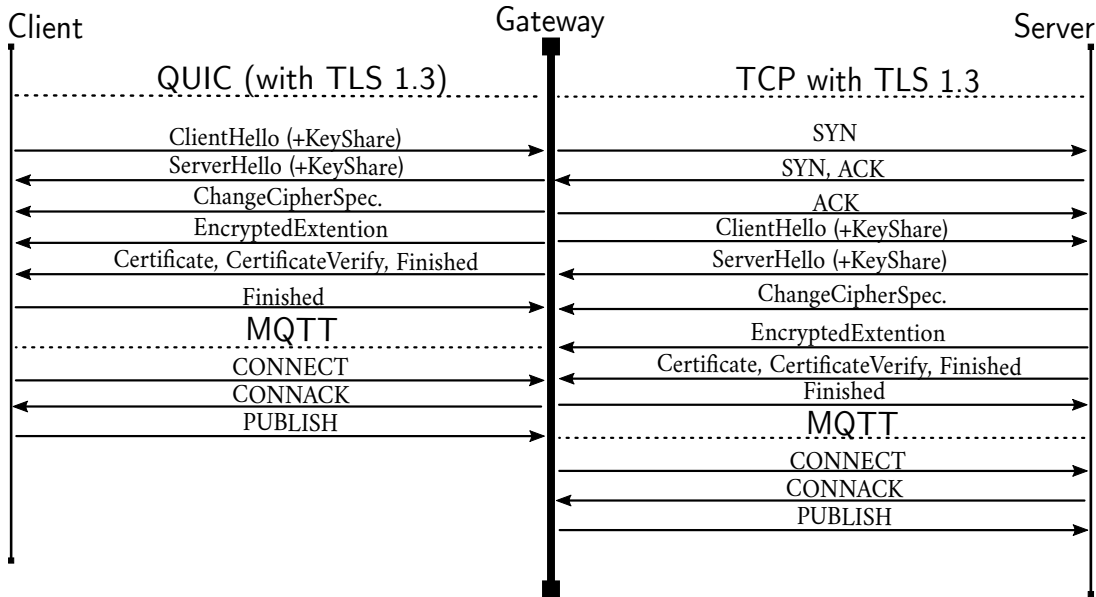


Contribution: the gateway secures endpoints connection implementing different layer protocols.



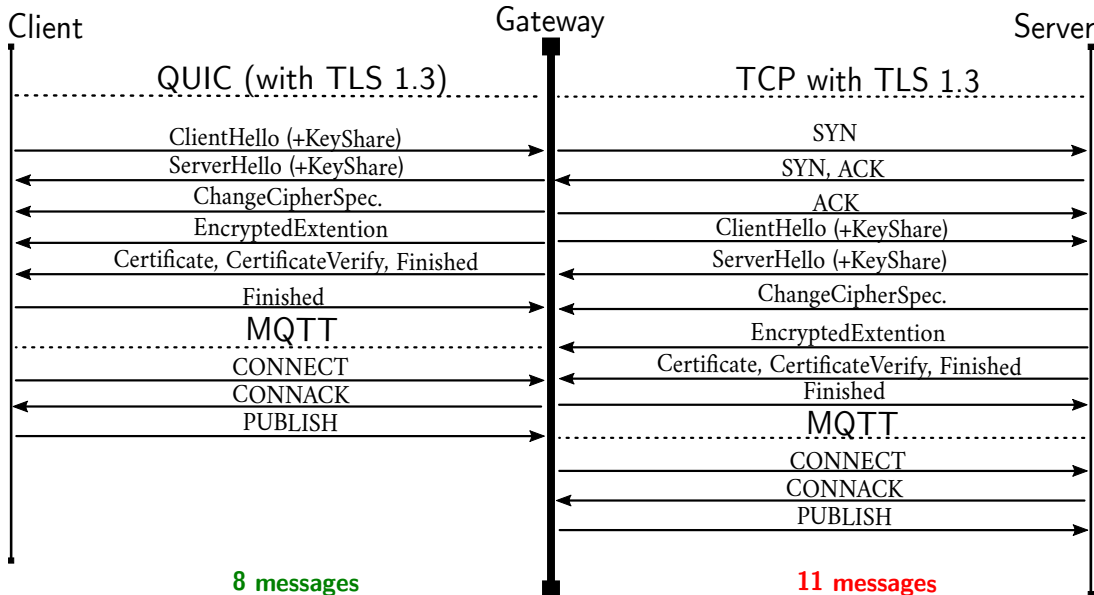
Contribution: the gateway secures endpoints connection implementing different layer protocols.

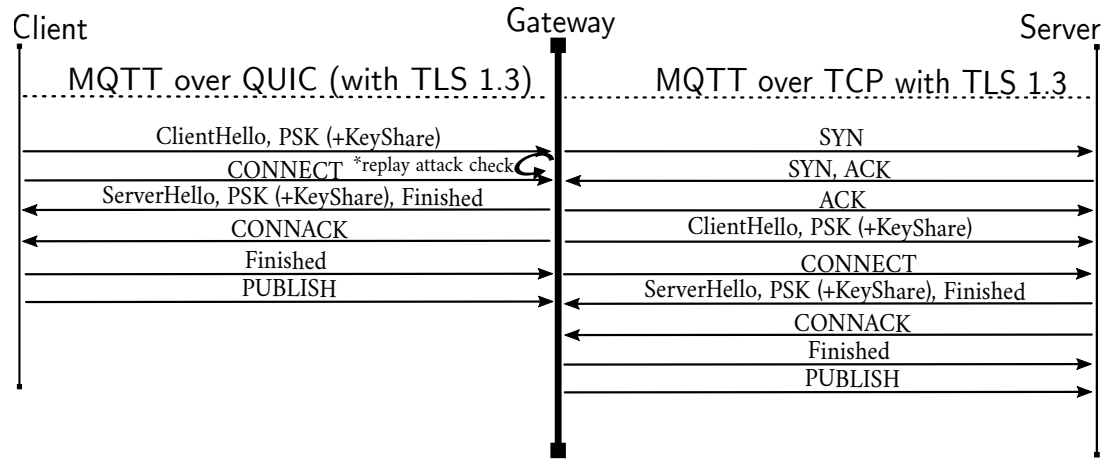
Benefit: cheap and secure

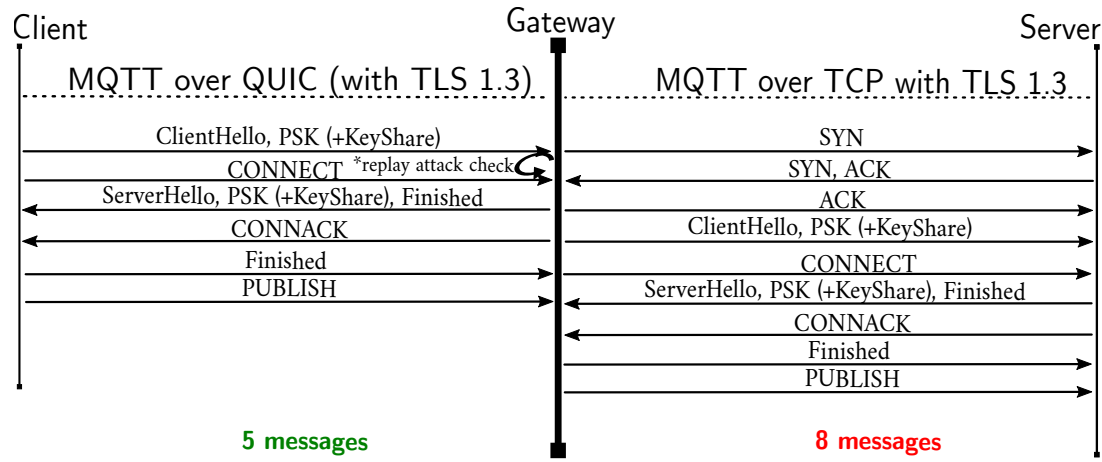




Packet Exchange - -> TLS 1.3 FH to TLS 1.3 FH

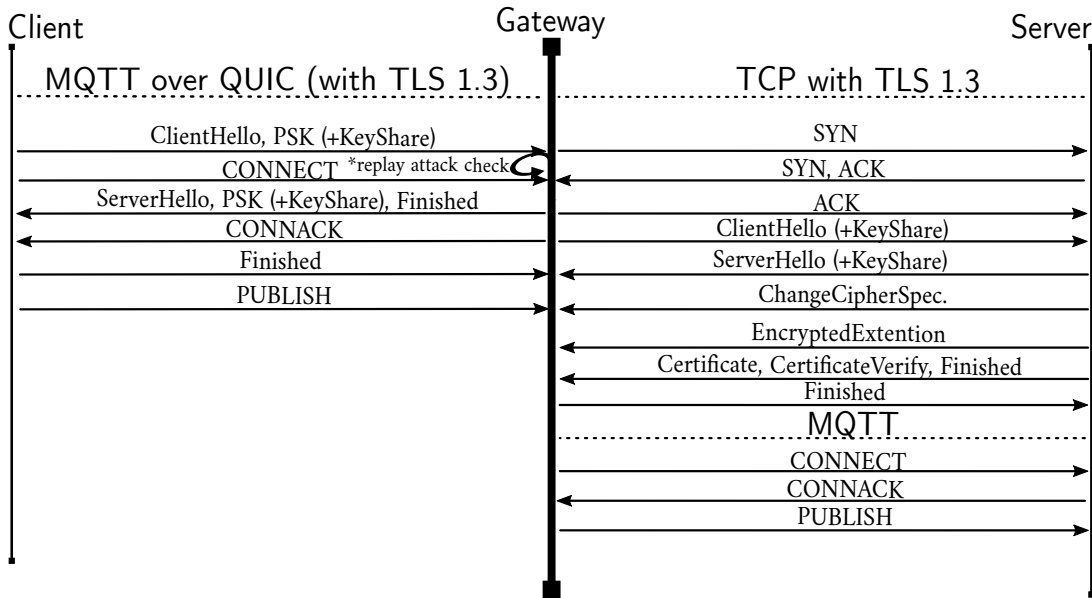






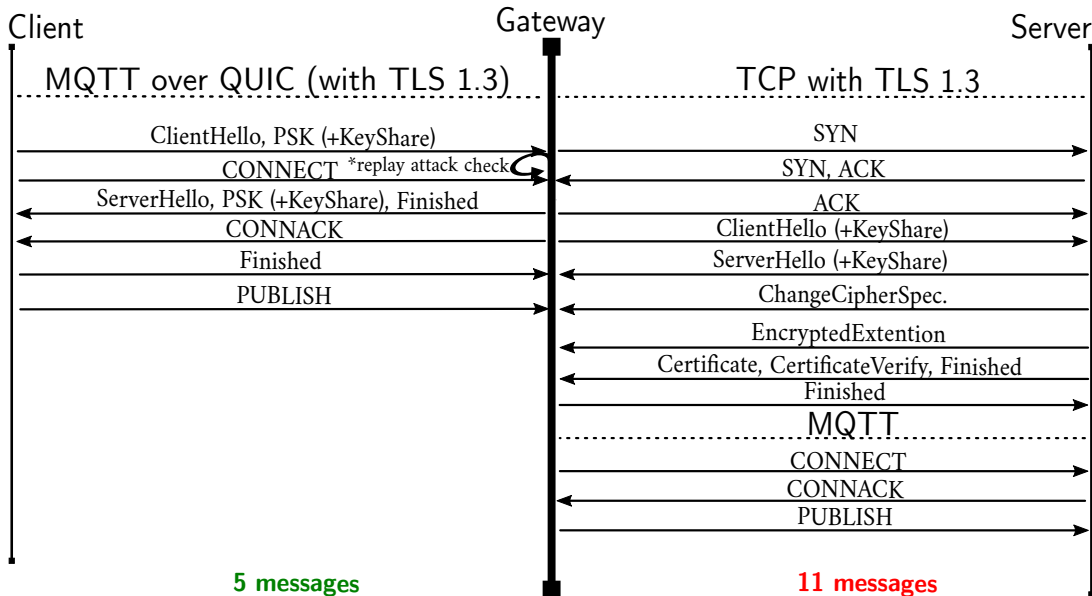


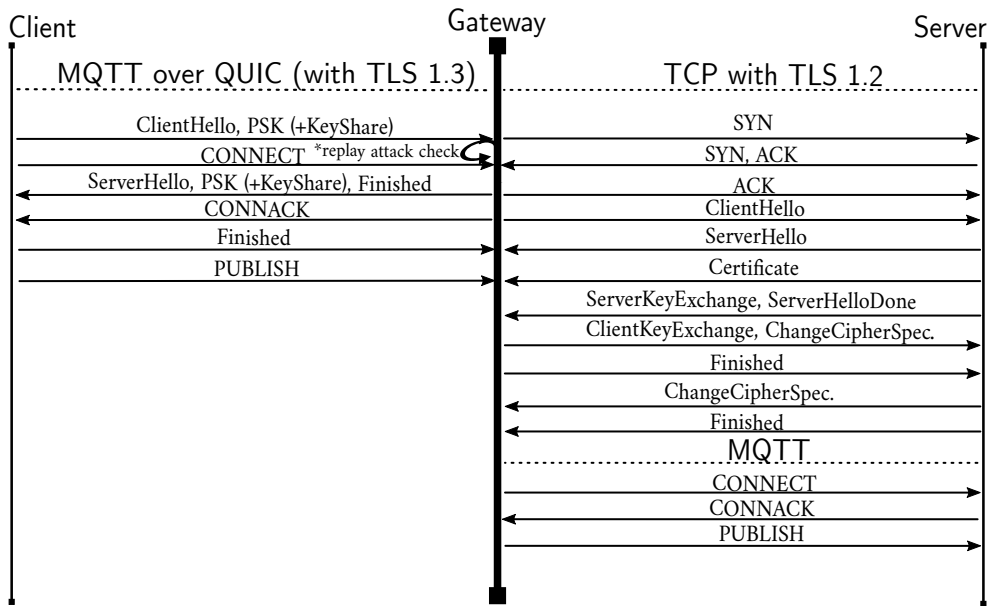
Packet Exchange - - -> TLS 1.3 0-RTT to TLS 1.3 FH





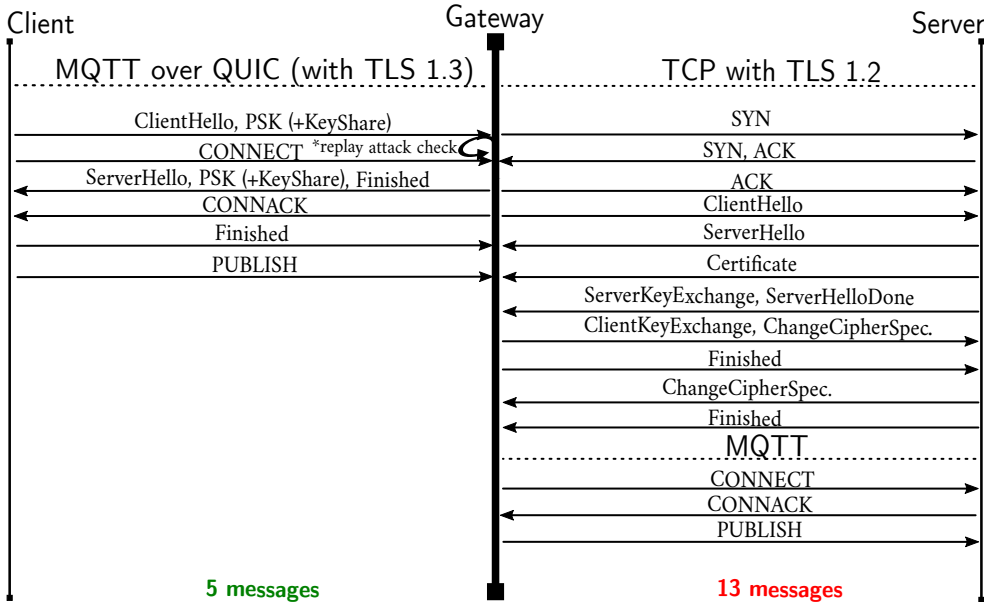
Packet Exchange - - -> TLS 1.3 0-RTT to TLS 1.3 FH







Packet Exchange - - -> TLS 1.3 0-RTT to TLS 1.2 FH



3

Conclusion

👍 Project Comparison

👍 Working on!

#	IoT devices secured via QUIC	via TLS directly
Security	high*	high*
Latency		
Resource usage		
Battery life		
Computations		

*Assuming that both connections use the actual versions of TLS.

#	IoT devices secured via QUIC	via TLS directly
Security	high*	high*
Latency	lower	higher
Resource usage		
Battery life		
Computations		

*Assuming that both connections use the actual versions of TLS.

#	IoT devices secured via QUIC	via TLS directly
Security	high*	high*
Latency	lower	higher
Resource usage	lower	higher
Battery life		
Computations		

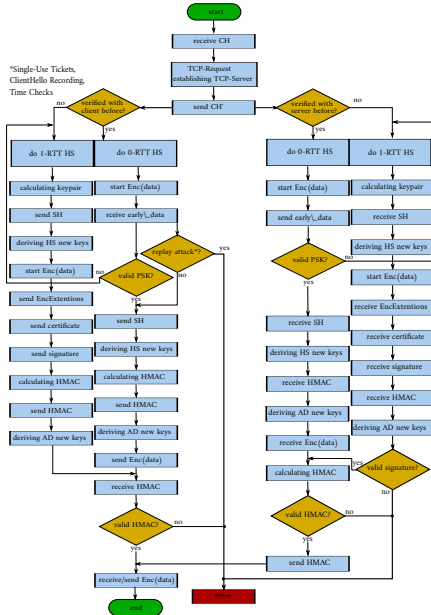
*Assuming that both connections use the actual versions of TLS.

#	IoT devices secured via QUIC	via TLS directly
Security	high*	high*
Latency	lower	higher
Resource usage	lower	higher
Battery life	longer	shorter
Computations		

*Assuming that both connections use the actual versions of TLS.

#	IoT devices secured via QUIC	via TLS directly
Security	high*	high*
Latency	lower	higher
Resource usage	lower	higher
Battery life	longer	shorter
Computations	lower	higher

*Assuming that both connections use the actual versions of TLS.





📍 Nice, France

📅 October, 2020



 Ahmed Alqattaa
a.alqattaa@oth-aw.de



 Daniel Loebenberger
d.loebenberger@oth-aw.de

- [1] M. Bishop.
Hypertext Transfer Protocol Version 3 (HTTP/3).
Internet-draft, Internet Engineering Task Force, September 2020.
Work in Progress, retrieved: October 2020.
- [2] J. Iyengar and M. Thomson.
QUIC: A UDP-Based Multiplexed and Secure Transport.
Internet-draft, Internet Engineering Task Force, September 2020.
Work in Progress, retrieved: October 2020.
- [3] E. Rescorla, H. Tschofenig, and N. Modadugu.
The Datagram Transport Layer Security (DTLS) Protocol Version 1.3.
Internet-draft, Internet Engineering Task Force, May 2020.
Work in Progress, retrieved: October 2020.
- [4] M. Thomson and S. Turner.
Using TLS to Secure QUIC.
Internet-draft, Internet Engineering Task Force, September 2020.
Work in Progress, retrieved: October 2020.