Internet stvari

SVEUČILIŠTE U ZAGREBU
Fakultet elektrotehnike i računarstva

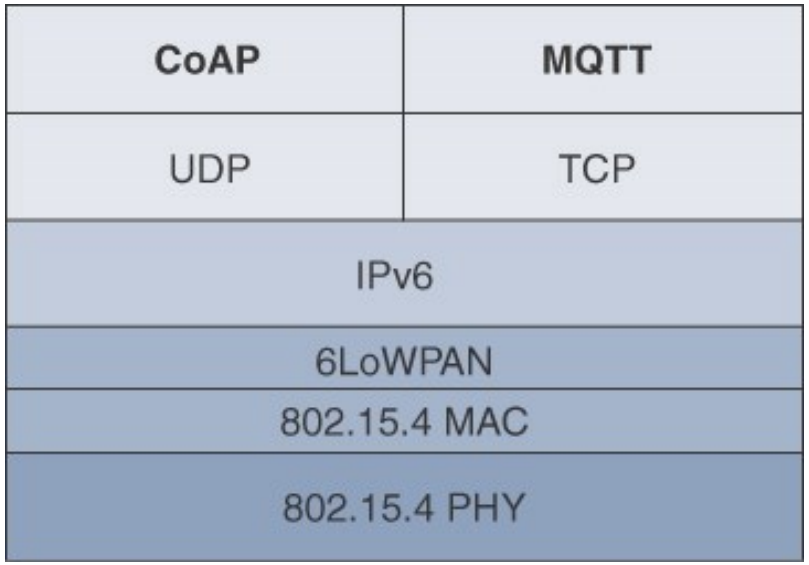
Diplomski studij
Računarstvo
Znanost o mrežama
Programsko inženjerstvo i informacijski sustavi
Računalno inženjerstvo
Informacijska i komunikacijska tehnologija
Automatika i robotika
Informacijsko i komunikacijsko inženjerstvo
Elektrotehnika i informacijska tehnologija
Audiotehnologije i elektroakustika
Elektroenergetika
(Izborni predmet profila)

4. Aplikacijski sloj: MQTT, CoAP

Ak. god. 2022./2023.

Sadržaj

- CoAP i MQTT
- Detaljni pregled dva aplikacijska protokola za uređaje i mreže ograničenih resursa



Constrained Application Protocol (CoAP)

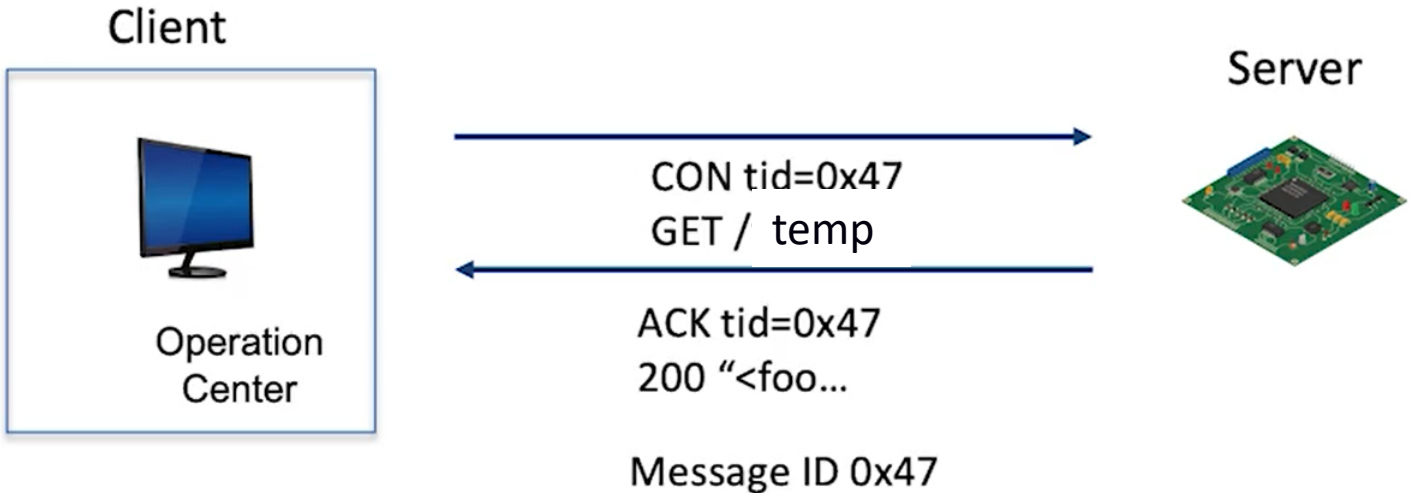
- **CoAP *framework*** definira jednostavne i fleksibilne načine za slanje i primanje podataka s IoT-uređaja te njihovo upravljanje
- IETF working group Constrained RESTful Environments (CoRE)
  - **RFC 6690:** Constrained RESTful Environments (CoRE) Link Format
  - **RFC 7252:** The Constrained Application Protocol (CoAP)
  - **RFC 7641:** Observing Resources in the Constrained Application Protocol (CoAP)
  - **RFC 7959:** Block-Wise Transfers in the Constrained Application Protocol (CoAP)
  - **RFC 8075:** Guidelines for Mapping Implementations: HTTP to the Constrained Application Protocol (CoAP)
  - Ostali relevantni dokumenti: http://datatracker.ietf.org/wg/core/

CoAP

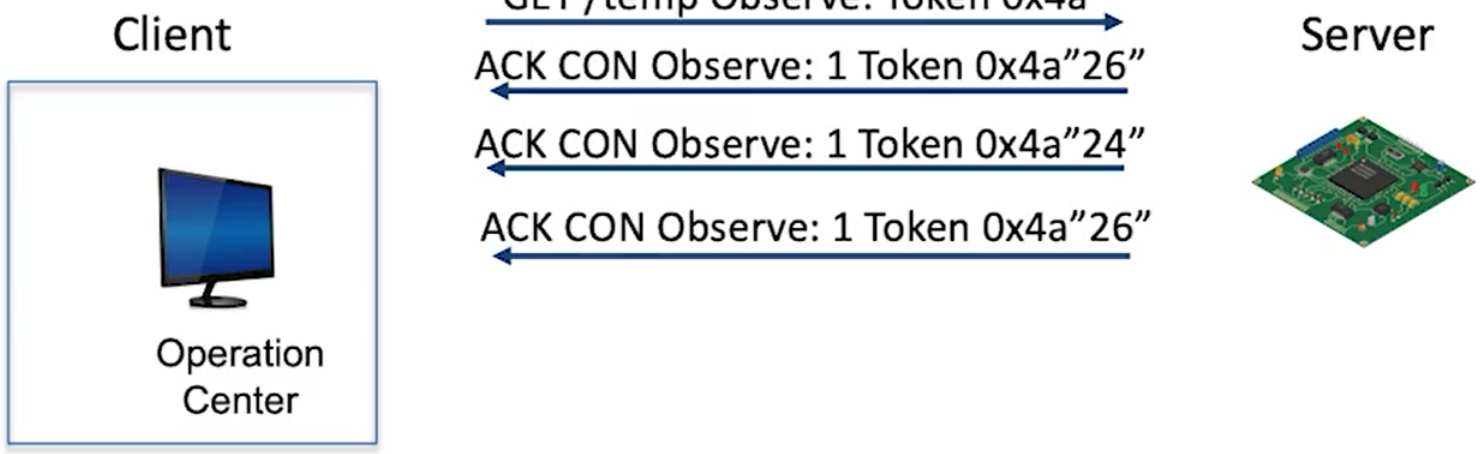
- Protokol je definiran u RFC 7252
- Resursi se identificiraju pomoću URI-ja, IoT-uređaj postaje poslužitelj koji nudi resuse
- Temelji se na REST-u (metode **GET**, **POST**, **PUT** i **DELETE**)
- Klijenti pristupaju resursima pomoću asinkronog mehanizma zahtjev-odgovor
- Koristi UDP kao transportni protokol i poseban “message layer” za retransmisiju izgubljenih paketa
- Koristi tehnike za kompresiju podataka
- Sigurna komunikacija omogućena protokolom DTLS (Datagram Transport Layer Security)

coap://mymeter.com:5683/ temp

1. način rada: zahtjev-odgovor



2. način rada: promatraj



CoAP URI

- URI se sastoji od sljedećih segmenata

coap[s]://<host>[:<port>]/<path>[?<query>]

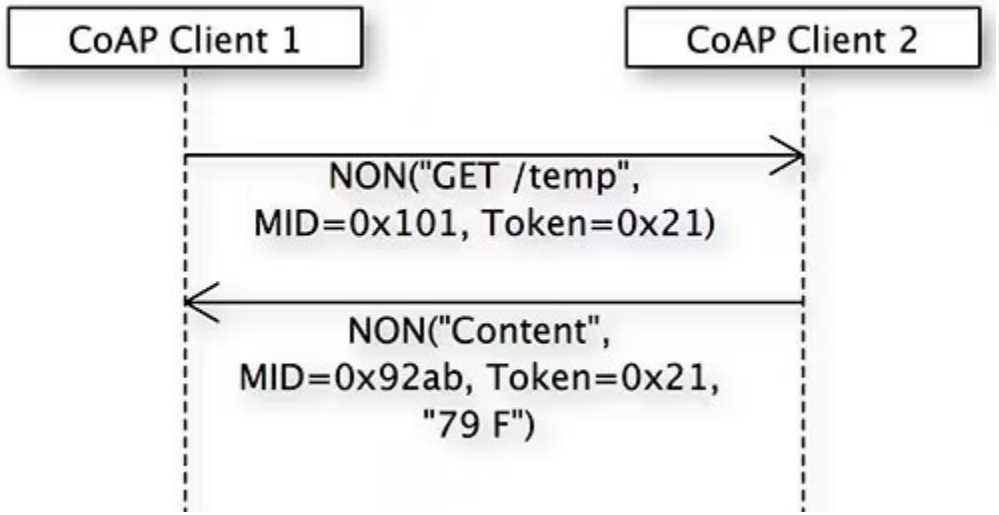
URI schema + authority + path + querry

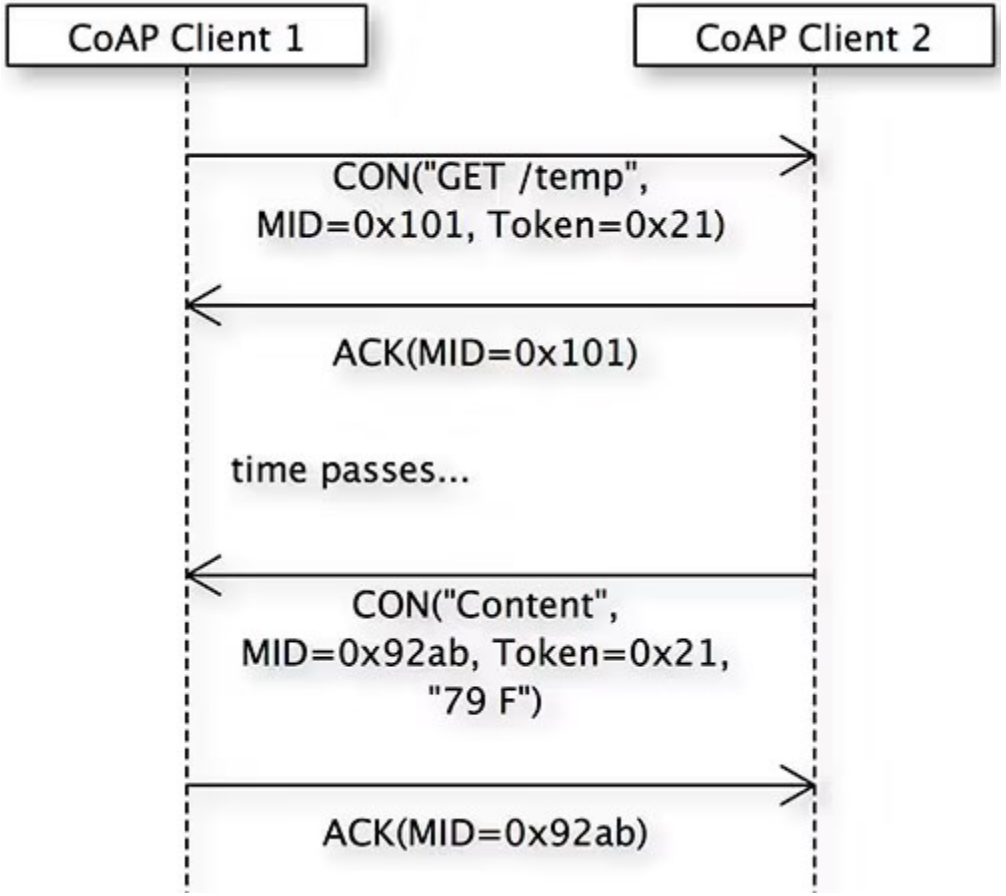
- CoAP URI
  - coap://example.smarthome.com:5683/sensors
- CoAPs URI
  - coaps://example.smarthome.com:5684/activate_alarm

Vrste CoAP poruka

Type	Description
CON	Confirmable Message Each confirmable message is acknowledged either by a message type of "Ack" or "Reset".
NON	Non-Confirmable Message For messages that do not require an acknowledgement. This is particularly true for messages that are repeated regularly for application requirements, such as repeated readings from a sensor where eventual success is sufficient.
Ack	Acknowledgement Message An Ack acknowledges that a specific CON message arrived. It does not indicate success or failure of any encapsulated request.
Reset	Reset Message A Reset message indicates that a specific message (CON or NON) was received, but some context is missing to properly process it. This condition is usually caused when the receiving node has rebooted and has forgotten some state that would be required to interpret the message. Provoking a Reset message (e.g., by sending an empty Confirmable message) is also useful as an inexpensive check of the liveness of an endpoint ("CoAP ping").

Osnovna interakcija: zahtjevi NON i CON





Izvor: https://blogs.oracle.com/javamagazine/post/java-coap-constrained-application-protocol-introduction

Metode CoAP-a

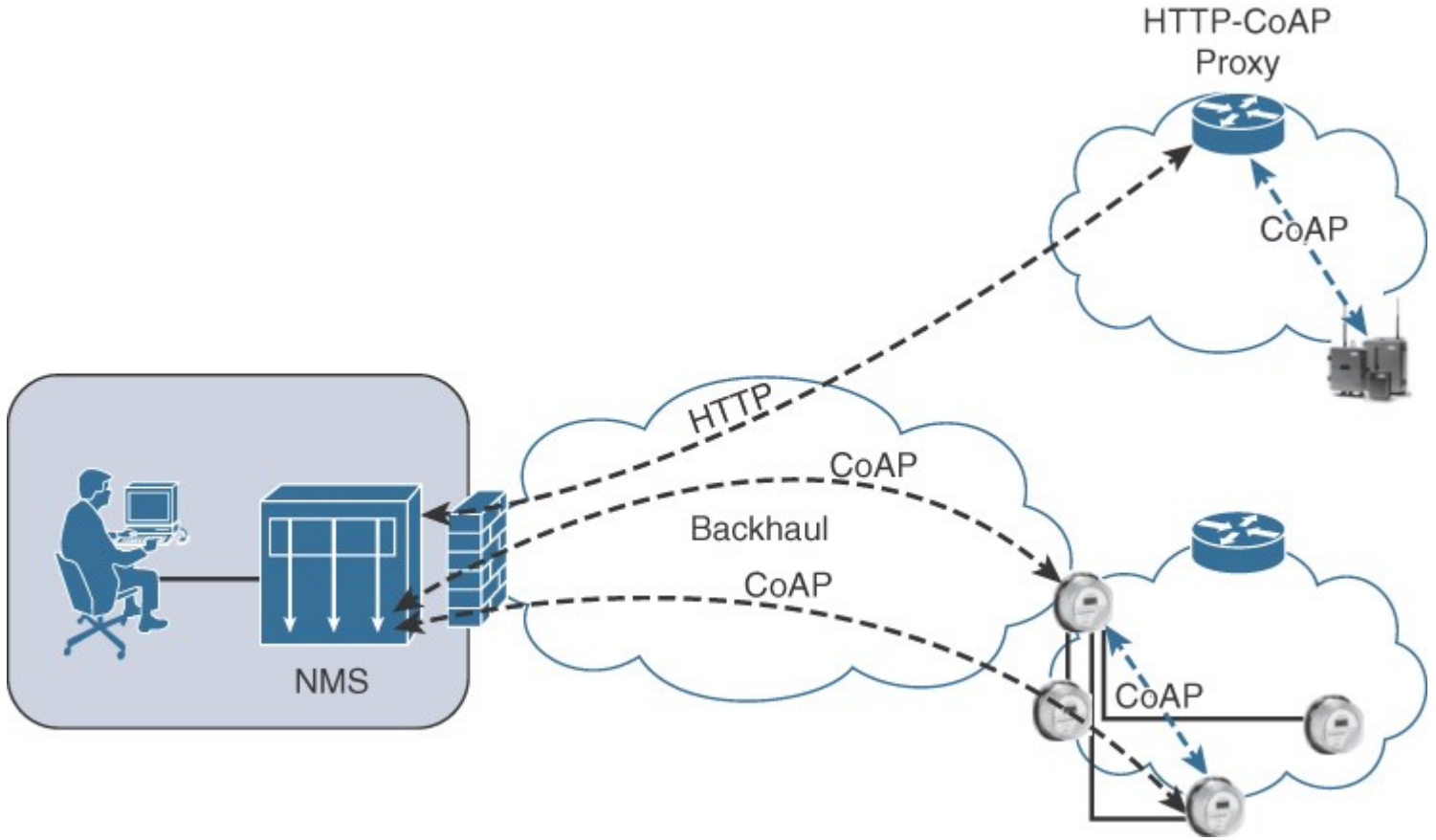
- **GET**
  - The GET method retrieves a representation for the information that currently corresponds to the resource identified by the request URI.
  - The GET method is safe and idempotent.
- **POST**
  - The POST method requests that the representation enclosed in the request be processed.
  - POST is neither safe nor idempotent.
- **PUT**
  - The PUT method requests that the resource identified by the request URI be updated or created with the enclosed representation.
  - PUT is not safe, but is idempotent.
- **DELETE**
  - The DELETE method requests that the resource identified by the request URI be deleted.
  - DELETE is not safe, but is idempotent.

Prijenos poruka

- Poruka nosi sljedeće: zahtjev, odgovor ili je prazna
- CoAP *endpoint* (IP address, UDP port) : izvor ili odredište poruke
- Poruke se razmjenjuju asinkrono
- Mehanizam za pouzdanu isporuku poruka je relativno jednostavan
  - *stop-and-wait-retransmission*, vrijeme čekanja na poruke se eksponencijalno povećava (vrijedi samo za poruke tipa *Confirmable*)
  - koristi se i brojač MAX_RETRANSMIT
  - detekcija duplikata (za poruke tipa *Confirmable* i *Non Confirmable*)

CoAP: protokolni složaj i „suživot” s HTTP-om

- Komunikacija među uređajima unutar jedne podmreže ili putem javnog Interneta
- Definiran je Proxy mehanizam u RFC 8075 koji mapira HTTP poruke u CoAP poruke



Prednosti i nedostaci CoAP-a

Prednosti

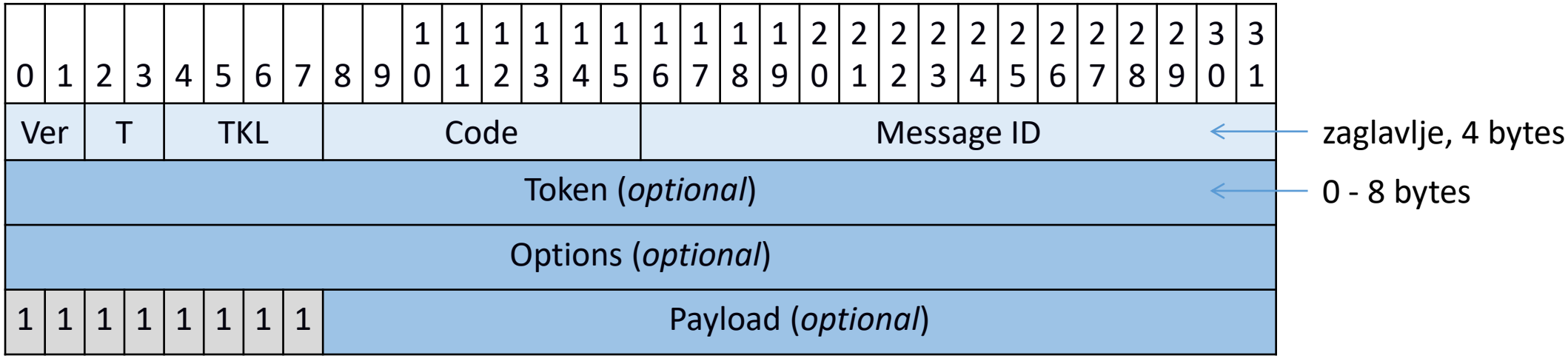
- Prilagođen uređajima i mrežama ograničenih resursa (koristi UDP)
- Vrlo niska potrošnja energije, može se koristiti u okolinama s ograničenim napajanjem (baterija)
- Temelji se na REST-u
- Omogućuje potvrdu poruke (QoS)
- DTSL za šifriranje poruka

Nedostaci

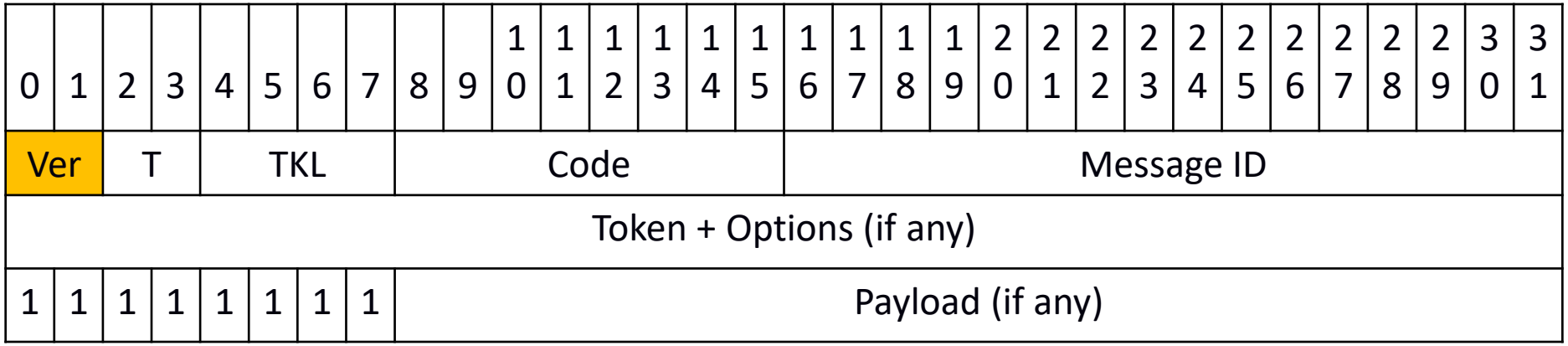
- Komunikacija je uvijek 1 na 1 (jedan izvor i jedno odredište)

Format CoAP poruke

- RFC 7252 definira sadržaj svakog CoAP okvira
  - **coap** (non-secured CoAP) koristi vrata 5683
  - **coaps** (DTLS-secured CoAP) koristi vrata 5684
  - podrška i za ostale transportne protokole: SMS, TCP, SCTP,...

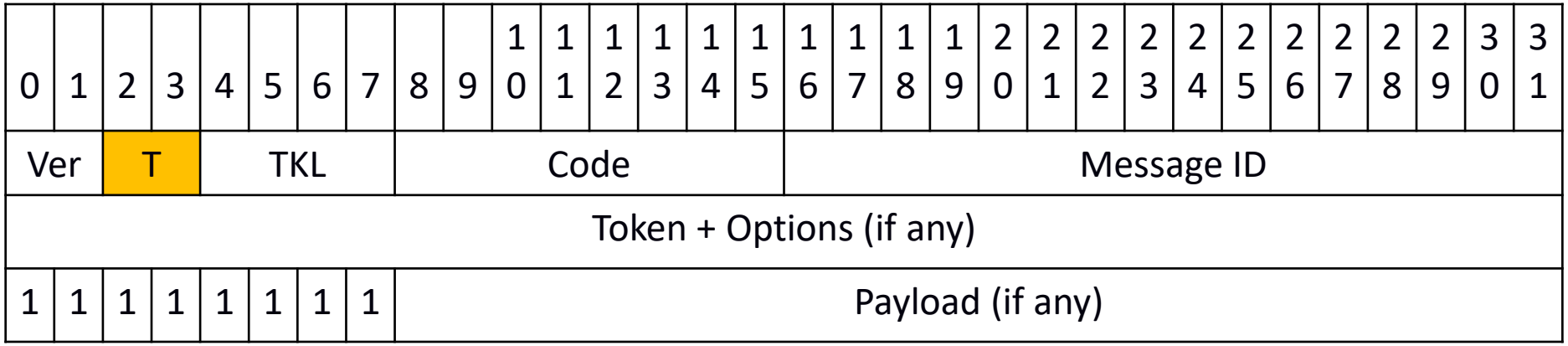


Objašnjenje zaglavlja CoAP poruke



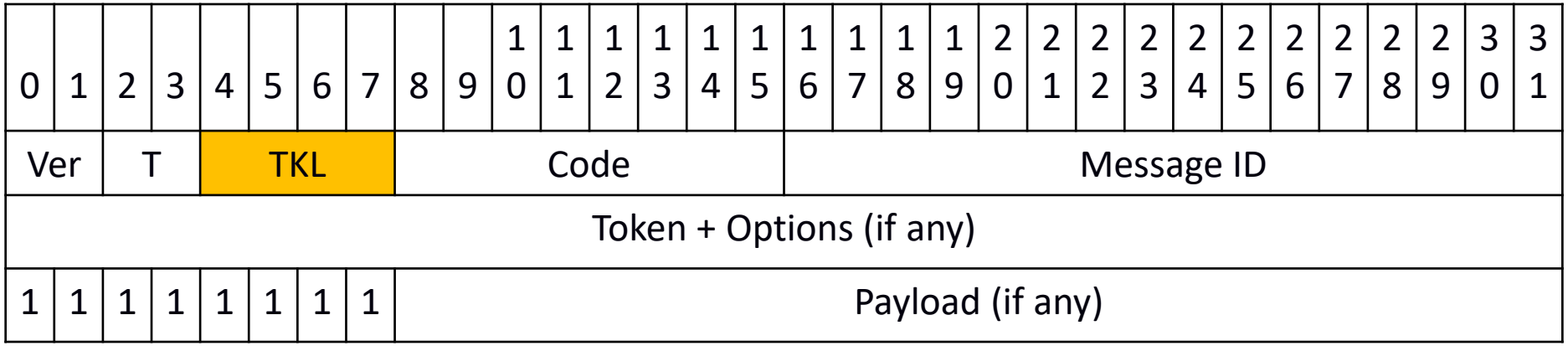
- ***Version*** ***(Ver)***
  - 2-bitni prirodni broj
  - predstavlja verziju CoAP protokola
  - prema trenutnoj specifikaciji mora biti postavljen na 01
  - nespecificirane verzije MORAJU biti ignorirane

Objašnjenje zaglavlja CoAP poruke



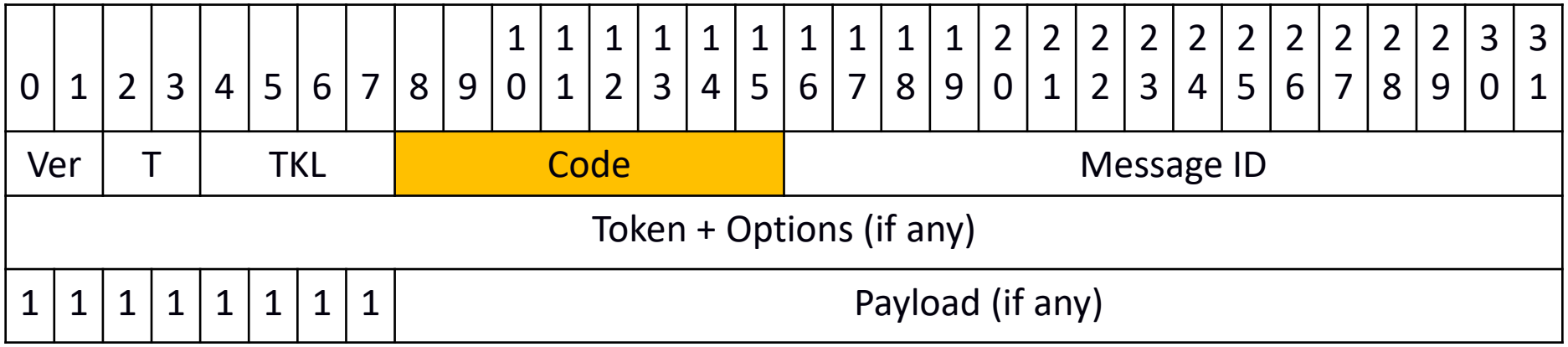
- ***Type*** ***(T)***
  - 2-bitni prirodni broj
  - predstavlja jedan od tipova poruka:
    - Comfirmable (0)
    - Non-confirmable (1)
    - Acknowledgement (2)
    - Reset(3)

Objašnjenje zaglavlja CoAP poruke



- ***Token Length** (TKL)*
  - 4-bitni prirodni broj
  - predstavlja duljinu polja Token
  - Token se koristi za povezivanje zahtjeva i njegovog odgovora te je neovisan o *Message ID*
  - *A token is intended for use as a client-local identifier for differentiating between concurrent requests; it could have been called a "request ID". The client SHOULD generate tokens in such a way that tokens currently in use for a given source/destination endpoint pair are unique.*

Objašnjenje zaglavlja CoAP poruke



- 8-bitni prirodni broj
- podjela u dvije cjeline: *3-bit class* (najznačajniji bitovi) i *5-bit detail (manje značajni bitovi)*
- notacija: „*c.dd*” gdje je *c* broj od 0 – 7, a *dd* broj od 00 – 31
- klase mogu biti:
  - Request (0)
  - Success response (2)
  - Client error response (4)
  - Server error response (5)
  - Posebni slučaj poruke je 0.00 koji predstavlja praznu poruku (*Empty message*)
- u slučaju upita (*Request*) polje *Code* predstavlja *Request code*, a u slučaju odgovora (*Response*) predstavlja *Response code*

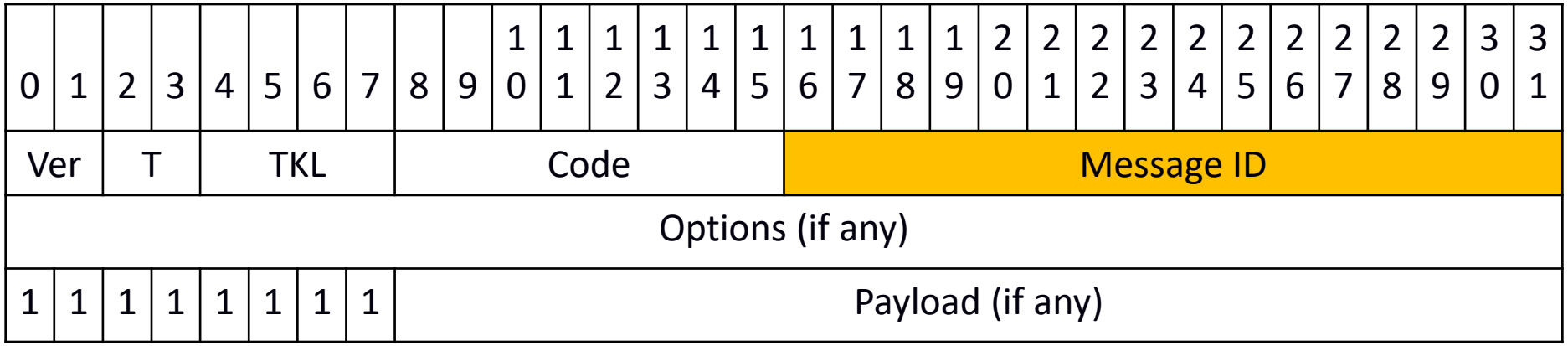
CoAP kodovi zahtjeva i odgovora

Općenita podjela	
Kod	Opis
0.00	Kod prazne poruke
0.01 – 0.31	Kodovi zahtjeva
1.00 – 1.31	*Rezervirani kodovi*
2.00 – 5.31	Kodovi odgovora
6.00 – 7.31	*Rezervirani kodovi*

Kodovi zahtjeva	
Kod	Opis
0.01	GET
0.02	POST
0.03	PUT
0.04	DELETE
0.05 – 1.31	*Nedodijeljeni kodovi*

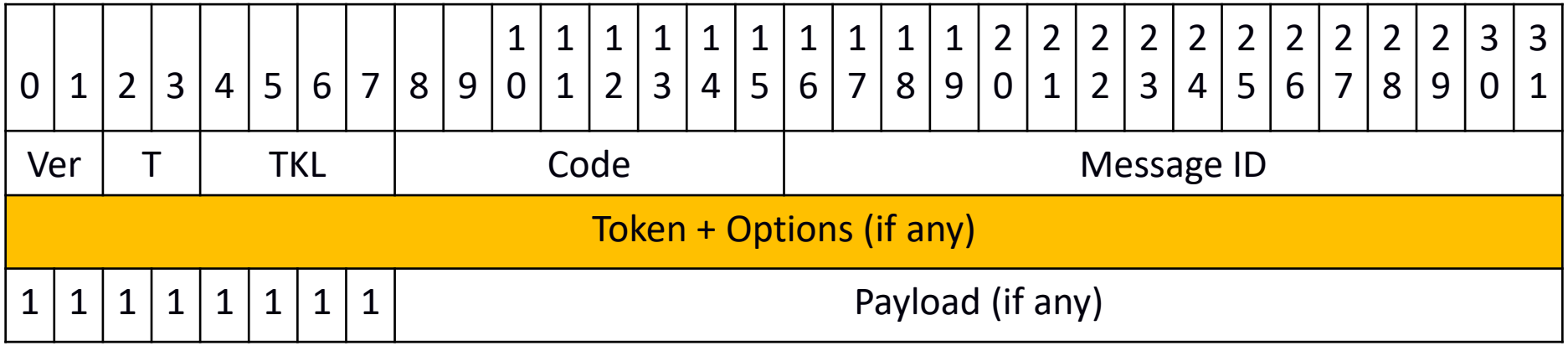
Kodovi odgovora					
Kod	Opis	4.01	Unauthorized	4.14	*Nedodijeljen kod*
2.00	*Nedodijeljen kod*	4.02	Bad option	4.15	Unsupported Content-Format
2.01	Created	4.03	Forbidden	4.16 – 4.31	*Nedodijeljeni kodovi*
2.02	Deleted	4.04	Not found	5.00	Internal Poslužitelj Error
2.03	Valid	4.05	Method not allowed	5.01	Not Implemented
2.04	Changed	4.06	Not acceptable	5.02	Bad Gateway
2.05	Content	4.07 – 4.11	*Nedodijeljeni kodovi*	5.03	Service Unavailable
2.06 – 2.31	*Nedodijeljeni kodovi*	4.12	Precision failed	5.04	Gateway Timeout
3.00 – 3.31	*Rezervirani kodovi*	4.13	Request entity too large	5.05	Proxying Not Supported
4.00	Bad request			5.06 – 5.31	*Nedodijeljeni kodovi*

Objašnjenje zaglavlja CoAP poruke



- ***Message ID***
  - 16-bitni prirodni broj
  - za detekciju višestrukih poruka i povezivanje poruka tipa *Acknowldgement/Reset s* porukama tipa *Confirmable/Non-confirmable*

Objašnjenje zaglavlja CoAP poruke

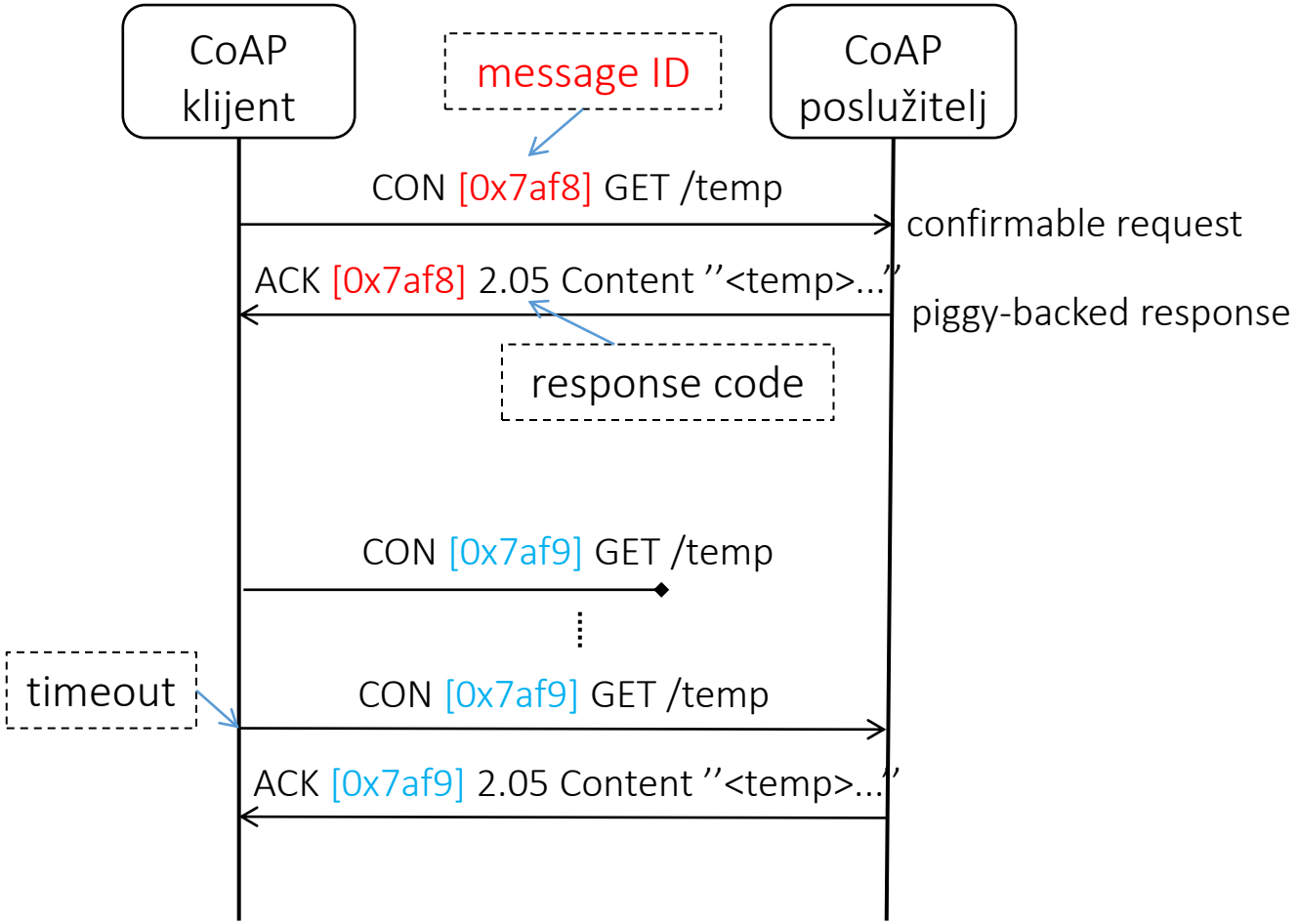


- Token se koristi za povezivanje zahtjeva i njegovog odgovora te je neovisan o *Message ID*
- svaka opcija mora biti u skladu sa specifikacijom, a sastoji se od:
  - Option number (delta) – 4-bitni prirodni broj
  - Option length – 4-bitni prirodni broj
  - Option value – sadrži jednu od sljedećih opcija:
    - Empty – niz bitova duljine 0
    - Opaque – nedefinirana struktura niza okteta
    - Uint – konačan niz okteta prikazan u polju *Option length*
    - String – niz znakova kodiran koristeći UTF8 (RFC3629)

CoAP defines a single set of options that are used in both requests and responses: ***Content-Format****,* ***Etag****, Location-Path, Location-Query, Max-Age, Proxy-Uri, Proxy-Scheme, Uri-Host, Uri-Path, Uri-Port, Uri-Query, Accept, If-Match, If-None-Match, Size*

Npr. CoAP Content-Formats: text, utf-8, json, xml…

CoAP – confirmable request



Interakcija: zahtjev-odgovor

Request:

Header: GET (T=CON, Code=1, MID=0x7af8)

Uri-Path: "temp"

8-bitni zapis odgovora 2.05

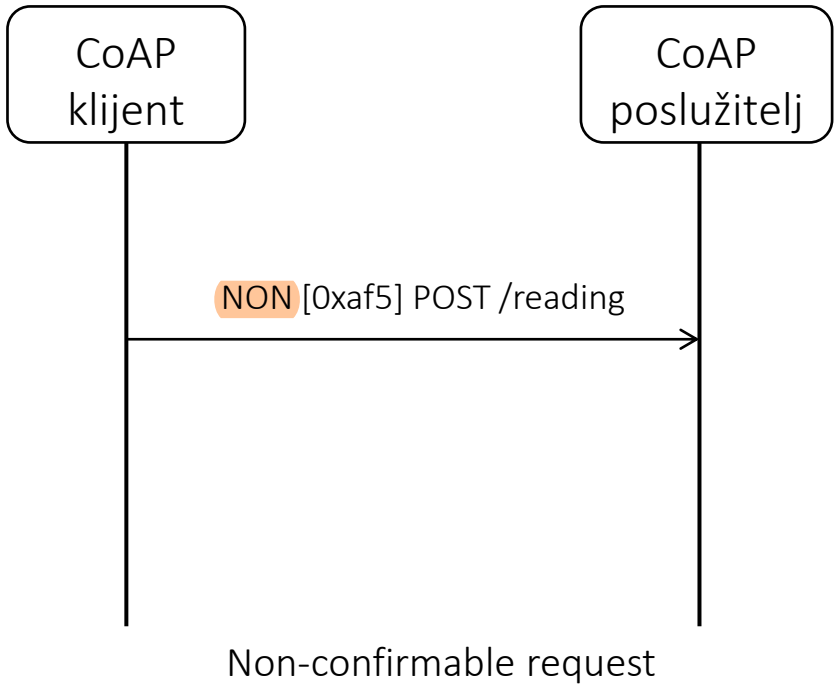
Response:

Header: 2.05 Content (T=ACK, Code=69, MID=0x7af8)

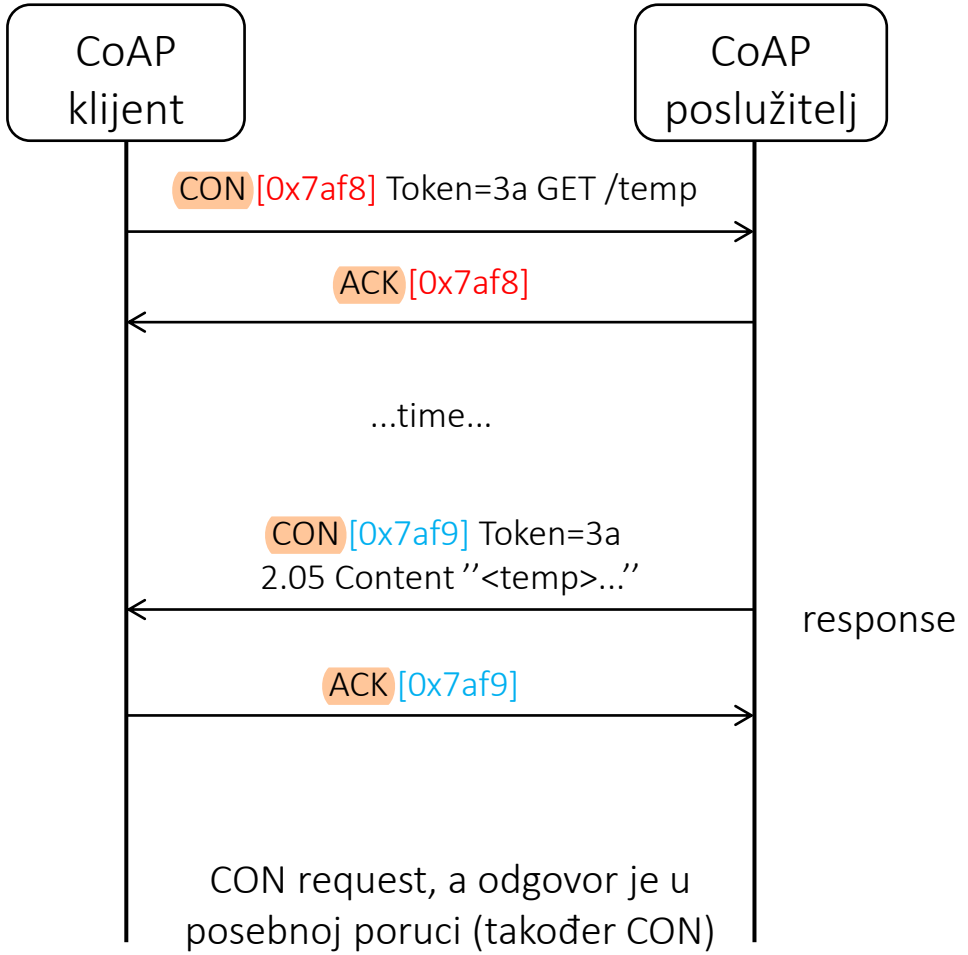
Payload: "<temp>22.3 C</temp>"

message ID omogućuje detekciju duplih poruka

CoAP – non-confirmable request i separate response

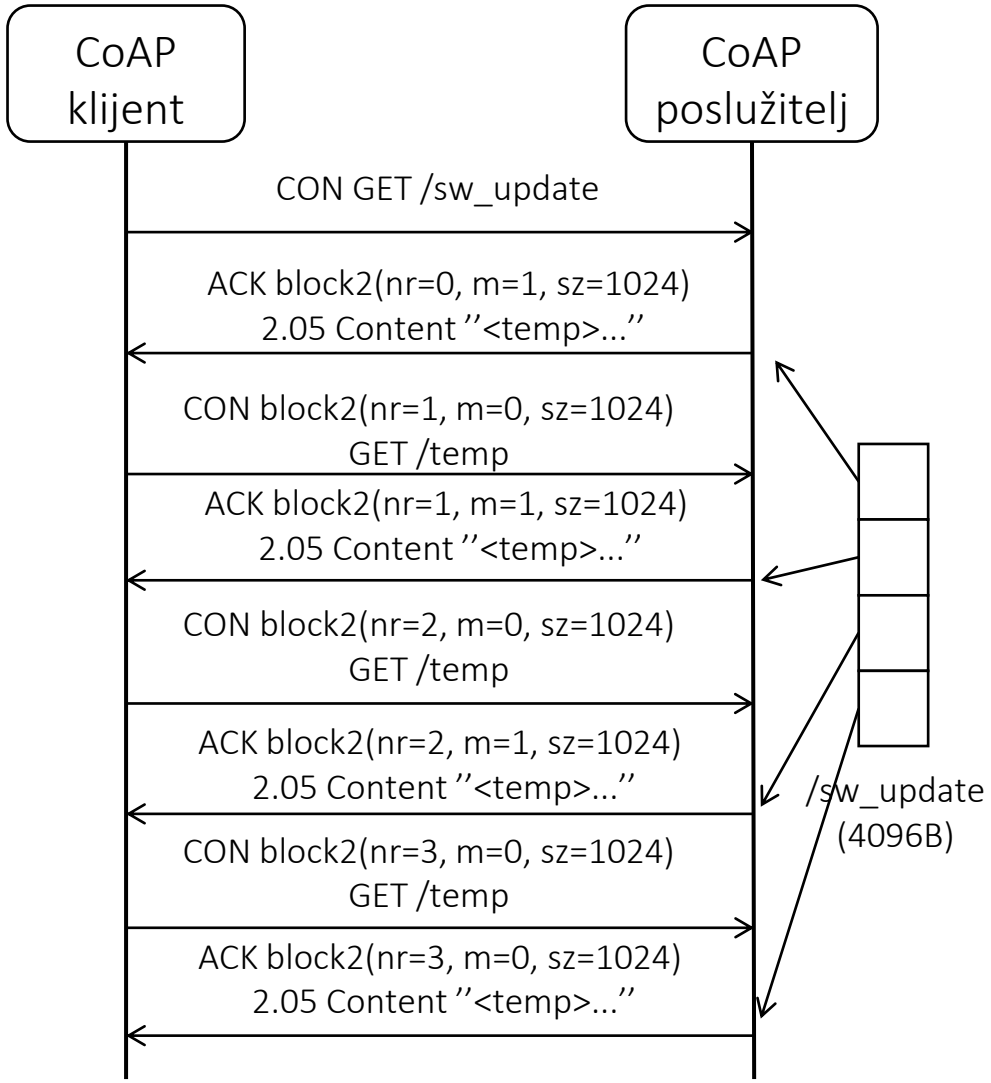


Non-confirmable request



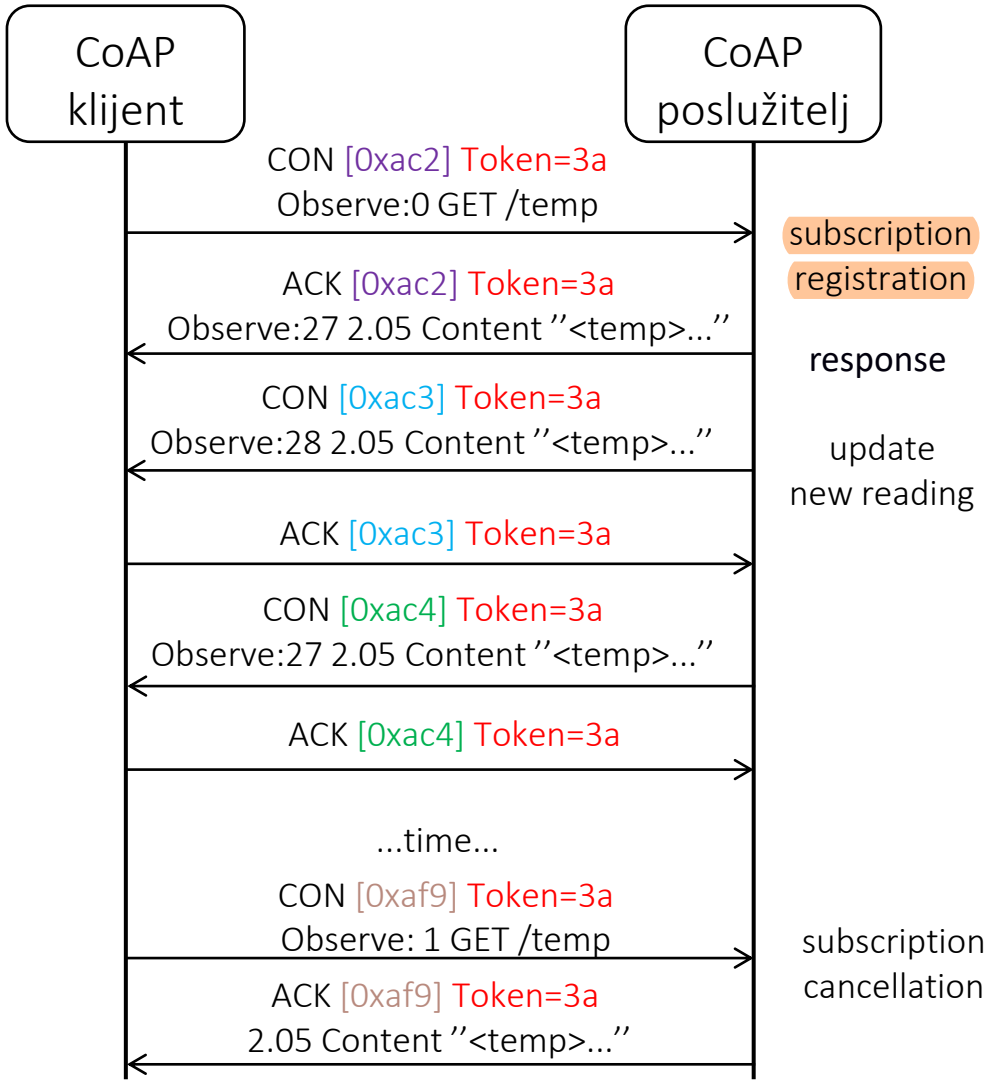
CON request, a odgovor je u posebnoj poruci (također CON)

CoAP – block transfer



If you're required to send content that exceeds the amount of data possible to embed in a datagram, CoAP allows you to divide your content into **blocks**.
Whenever you send data using CoAP, you can also provide a **block size**. If the content exceeds this block size, the CoAP layer divides it into blocks and transmits them individually, while the receiving end assembles the blocks and delivers the complete payload to its application at the other end. Block-wise transfers using CoAP is defined in RFC 7959: https://tools.ietf.org/html/rfc7959.

CoAP - observation



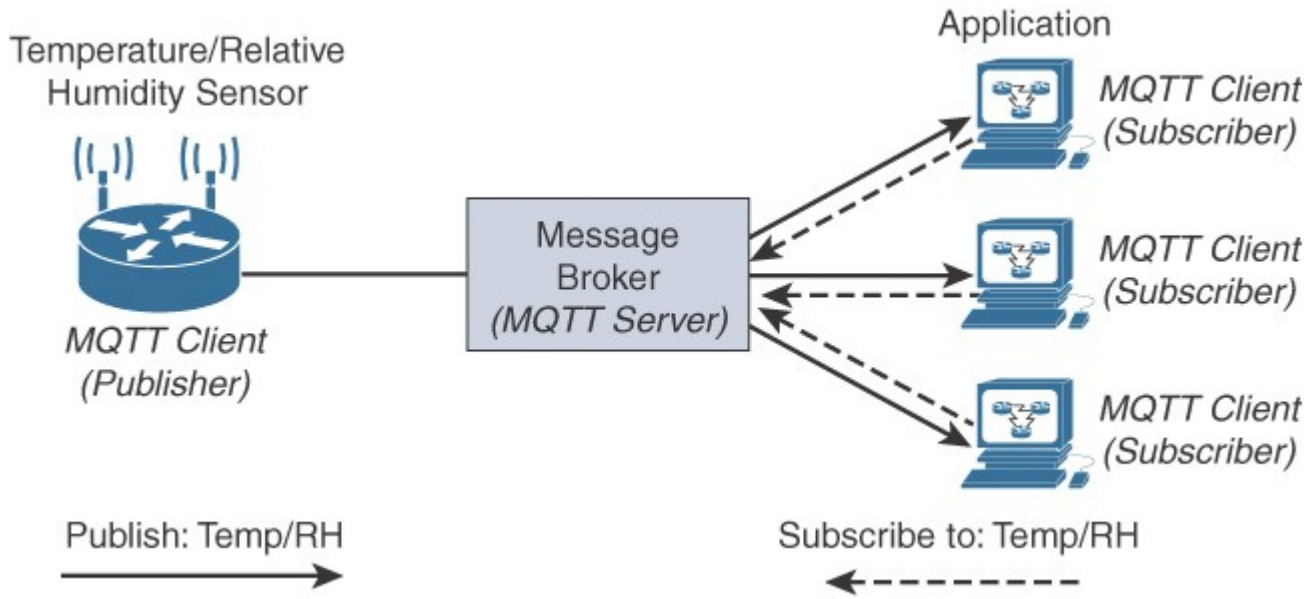
- Klijent registrira pretplatu
- U odgovoru (ACK) se prenosi očitanje, a sljedeći odgovori se šalju bez eksplicitnog zahtjeva dok ne stigne poruka za otkazivanje pretplate
- Smanjuje se generirani promet, očitanja se dostavljaju u trenutku nastanka
- Resursi koji podržavaju ovaj pattern se zovu *observable resources,* definirano u RFC 7641: https://tools.ietf.org/html/rfc7641

Message Queuing Telemetry Transport (MQTT)

- Jednostavan protokol za prijenos poruka na načelu **objavi-pretplati** prilagođen uređajima i mrežama s ograničenim resursima
- Dodaje mali transportni overhead (header je veličine 2 byte), „*lightweight protocol*”
- Koristi TCP/IP na mrežnom sloju (najčešće, moguće izvedbe putem UDP-a)
- http://mqtt.org
- MQTT je standard koji definira OASIS (Organization for the Advancement of Structured Information Standards)
  - posljednja verzija standarda MQTT v5.0, 7.3.2019.
  - prethodna verzija MQTT v3.1.1, 10.12.2015.

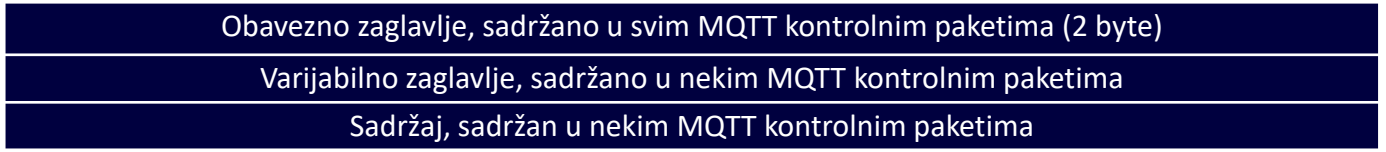
Osnovni komunikacijski mehanizam

- **MQTT Broker**: posrednik između objavljivača i pretplatnika (poslužitelj)
- **MQTT Client**
  - *Publisher*: objavljuje poruke na *Topic*
  - *Subscriber*: pretplaćuje se na *Topic*, prima objavljene poruke putem Brokera nakon što ih posrednik primi od objavljivača
  - 1 poruka se potencijalno isporučuje do *n* pretplatnika



Kontrolni paketi

- Sastoje se od zaglavlja fiksne duljine (2 byte) nakon čega može slijediti:
  - *varijabilno zaglavlje*
  - *sadržaj* (do 256 MB)



- MQTT prenosi kontrolne pakete putem TCP-a (port 1883)
- TCP osigurava ispravan prijenos paketa bez gubitaka
- MQTT se može koristiti u kombinaciji s TLS-om (port 8883)
- Za prijenos se također može koristiti WebSocket (definirano u RFC 6455)
- *Durable Sessions*: omogućuju perzistentnost poruka (*Store and Forward*)

MQTT-sjednica

- Ostvaruje se između svakog klijenta i brokera
- Sastoji se od 4 faze:
  - *session establishment*,
  - *authentication*,
  - *data exchange*,
  - *session termination*.
- Svaki klijent se identificira jedinstvenim identifikatorom koji služi i za identifikaciju otvorene sjednice
- Prilikom isporuke poruke do više klijenata, broker će svaku isporuku obaviti neovisno o ostalim sjednicama

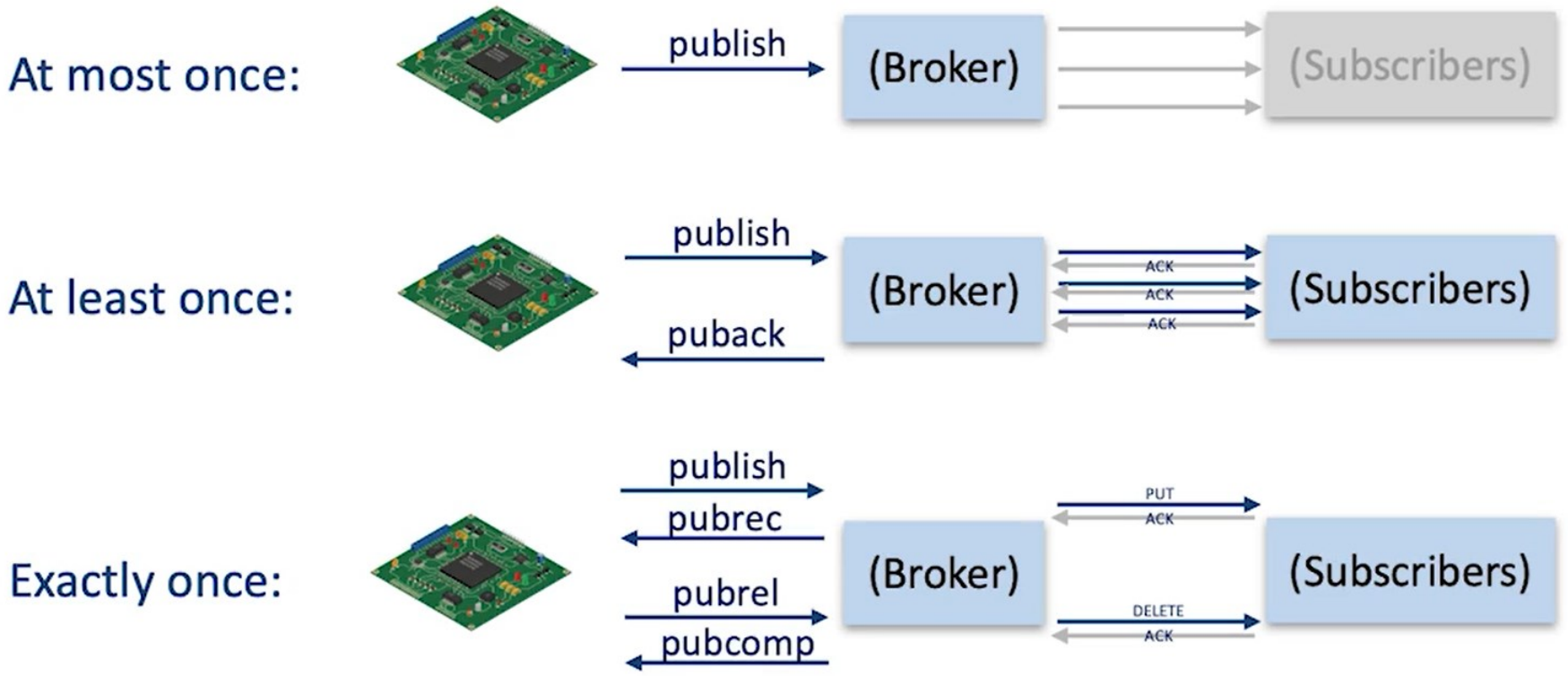
Vrste poruka

- Pretplate na temu generiraju kontrolne pakete SUBSCRIBE/SUBACK
- Brisanje pretplate na temu se ostvaruje pomoću UNSUBSCRIBE/UNSUBACK
- Slanje podataka i potvrda primanja PUBLISH/PUBACK
- Za zatvaranje konekcije se koristi DISCONNECT
  - jedina poruka iza koje ne slijedi ACK
  - slanjem client ID-ja se ostvaruje reconnect
- Za održavanje sjednice (zbog konekcije TCP-a) se koriste PINGREQ/PINGRESP
  - provjera da je klijent još uvijek dostupan
- PUBREC i PUBREL (objašnjene na sl. 29)

Razine QoS-a (1/2)

- **Razina 0** (najniža, *best effort*) uključuje isporuku poruke najviše jedan put, pri čemu je moguće da poruka uopće ne stigne do odredišta.
  - *At most once*, koristi se samo PUBLISH
- **Razina 1** uključuje isporuku barem jedan put, pri čemu je moguće primanje više od jedne poruke. Moguće su duple poruke.
  - *At least once,* koriste se PUBLISH i PUBACK, a u varijabilnom zaglavlju piše *packetID*.
  - Ako ne stigne PUBACK, PUBLISH se ponavlja.
- **Razina 2** uključuje isporuku poruke točno jedan put.
  - *Exactly once*, ova razina ima značajan *overhead* i zahtijeva ACK u dva koraka

Razine QoS-a (2/2)



Poruke za QoS=2

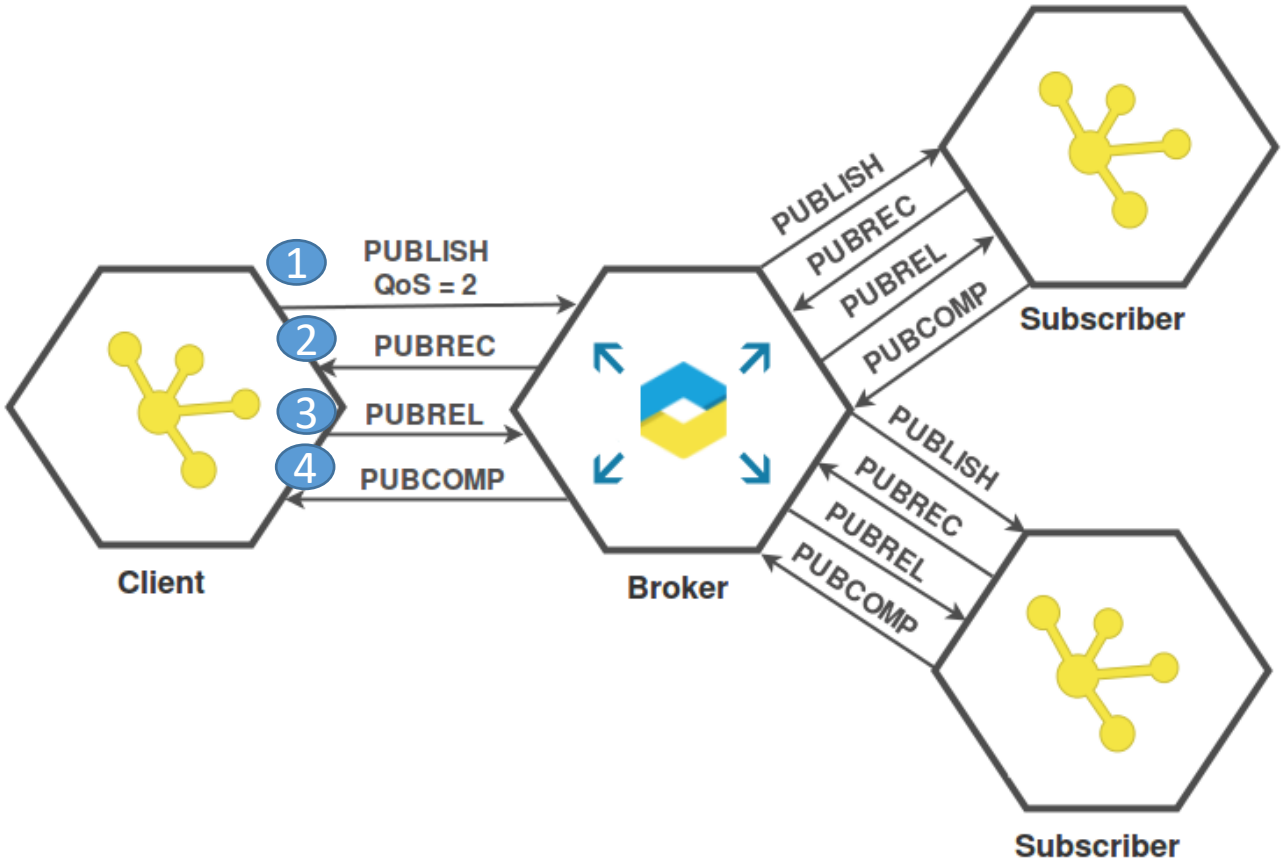
1. Klijent šalje PUBLISH s QoS=2.

2. Broker pohranjuje poruku i odgovara s PUBREC klijentu.

3. Klijent šalje brokeru PUBREL.

4. Broker pohranjuje poruku i odgovara s PUBCOMP.

5. Nakon primitka poruke, broker ponavlja isporuku u dva koraka PUBLISH/PUBREC i PUBREL/PUBCOMP do svakog pretplatnika.



https://www.iotbroker.cloud/blog/MQTT/Publish%20message%20flow%20and%20levels%20of%20QoS%20in%20MQTT

Jezik za pretplate

- Temelji se na hijerarhiji tema (*topic*), teme u hijerarhiji se odvajaju znakom /
  - Npr. `FER/C08-18/sensors/temperature`, `FER/C08-18/sensors/humidity`
  - Klijent-objavljivač ne mora prije slanja prve poruke na temu konfigurirati novu temu na brokeru, samo šalje poruku i definira temu
- Primjeri pretplate
  - `FER/C08-18/sensors/temperature` (samo na jednu temu)
  - `FER/#` (sve teme koje počinju s FER, # označava sve preostale stupnjeve hijerarhije)
  - `FER/C08-18/#` (sve teme registrirane za FER/C08-18, npr. svi senzori i aktuatori)
  - `FER/+/sensors/+` (pretplata na sve senzore na FER-u, + označava samo jedan stupanj u hijerarhiji)

Dodatni koncepti

- *Clean Session*
  - ako je *clean session flag* = 1 kada se ostvaruje sjednica, sve definirane pretplate tijekom sjednice će biti uklonjene s brokera nakon njenog zatvaranja
  - ako je *clean session flag* = 0 pretplate ostaju definirane i nakon zatvaranja sjednice
- *Durable Connections*
  - Kada je *clean session flag* = 0, sve poruke isporučene brokeru za vrijeme kada je klijent odspojen koje imaju QoS>0 će biti pohranjene na brokeru i isporučene nakon što se klijent ponovno spoji
- *Will flag*
  - definira pravila u slučaju neočekivanog prekida konekcije, postavlja klijent kada kreira sjednicu
  - MQTT v5.0 definira *Session expiry* i *Message expiry*

MQTT: format zaglavlja

Zaglavlje, 2 byte

bit	7	6	5	4	3	2	1	0
byte 1	Message Type				DUP flag	QoS level		RETAIN
byte 2	Remaining Length							

- **CONNECT** (klijent -> poslužitelj) – klijent poslužitelju šalje zahtjev za otvaranje konekcije;
- **CONNACK** (poslužitelj -> klijent) – poslužitelj klijentu šalje potvrdu o stvaranju konekcije;
- **PUBLISH** (klijent -> poslužitelj, poslužitelj -> klijent) – slanje poruke;
- **PUBACK** (klijent -> poslužitelj, poslužitelj -> klijent) – potvrda o slanju poruke;
- **PUBREC*** (klijent -> poslužitelj, poslužitelj -> klijent) – PUBlish RECeived, potvrda o primanju poruke;
- **PUBREL*** (klijent -> poslužitelj, poslužitelj -> klijent) – PUBlish RELease, potvrda za otpuštanje poruke;
- **PUBCOMP*** (klijent -> poslužitelj, poslužitelj -> klijent) – PUBlish COMPlete, potvrda o uspješnom prijenosu poruke, posljednja potvrda ra QoS=2;

*poruke se koriste kod sigurnog isporučivanja (engl. *assured delivery*), vezano uz QoS level

MQTT: format zaglavlja

Zaglavlje, 2 byte

bit	7	6	5	4	3	2	1	0
byte 1	Message Type				DUP flag	QoS level		RETAIN
byte 2	Remaining Length							

- **SUBSCRIBE** (klijent -> poslužitelj) – klijent poslužitelju šalje zahtjev za pretplatom;
- **SUBACK** (poslužitelj -> klijent) – poslužitelj klijentu šalje potvrdu o pretplati;
- **UNSUBSCRIBE** (klijent -> poslužitelj) – klijent poslužitelju šalje zahtjev za brisanjem pretplate;
- **UNSUBACK** (poslužitelj -> klijent) – poslužitelj klijentu šalje potvrdu o brisanju pretplate;
- **PINGREQ** (klijent -> poslužitelj) – klijent šalje poslužitelju PING zahtjev (za održavanje TCP konekcije);
- **PINGRESP** (poslužitelj -> klijent) – poslužitelj odgovara klijentu na PING zahtjev;
- **DISCONNECT** (klijent -> poslužitelj) – klijent zatvara konekciju prema poslužitelju.

MQTT: kodovi za Message Type

Mnemonic	Enumeration	Description
Reserved	0	Reserved
CONNECT	1	Client request to connect to Server
CONNACK	2	Connect Acknowledgment
PUBLISH	3	Publish message
PUBREC	5	Publish Received (assured delivery part 1)
PUBREL	6	Publish Release (assured delivery part 2)
PUBCOMP	7	Publish Complete (assured delivery part 3)
SUBSCRIBE	8	Client Subscribe request
SUBACK	9	Subscribe Acknowledgment
UNSUBSCRIBE	10	Client Unsubscribe request
UNSUBACK	11	Unsubscribe Acknowledgment
PINGREQ	12	PING Request
PINGRESP	13	PING Response
DISCONNECT	14	Client is Disconnecting
Reserved	15	Reserved

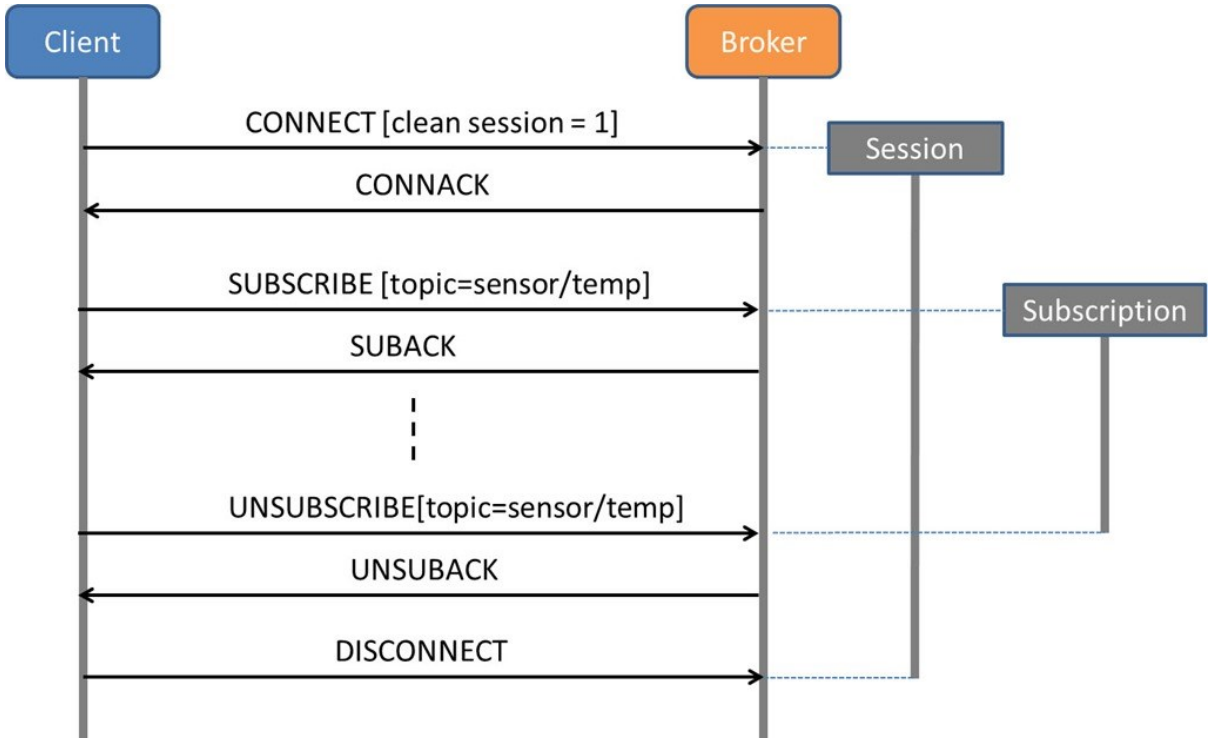
MQTT: format zaglavlja

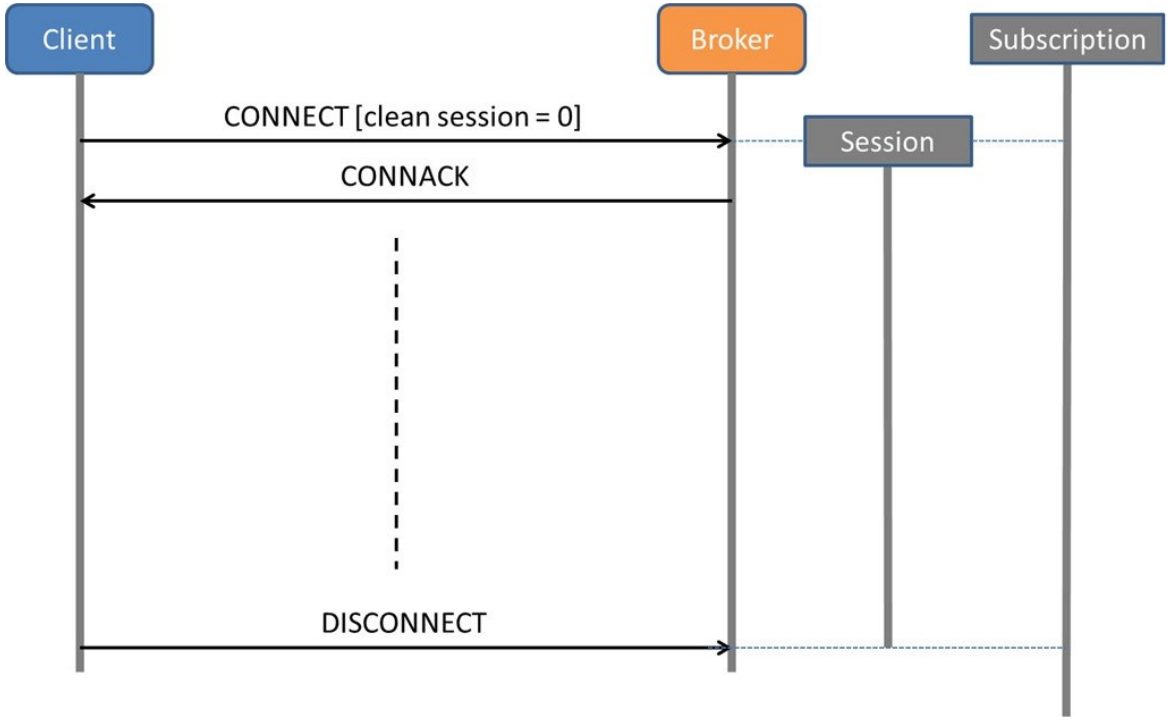
Zaglavlje, 2 byte

bit	7	6	5	4	3	2	1	0
byte 1	Message Type				DUP flag	QoS level		RETAIN
byte 2	Remaining Length							

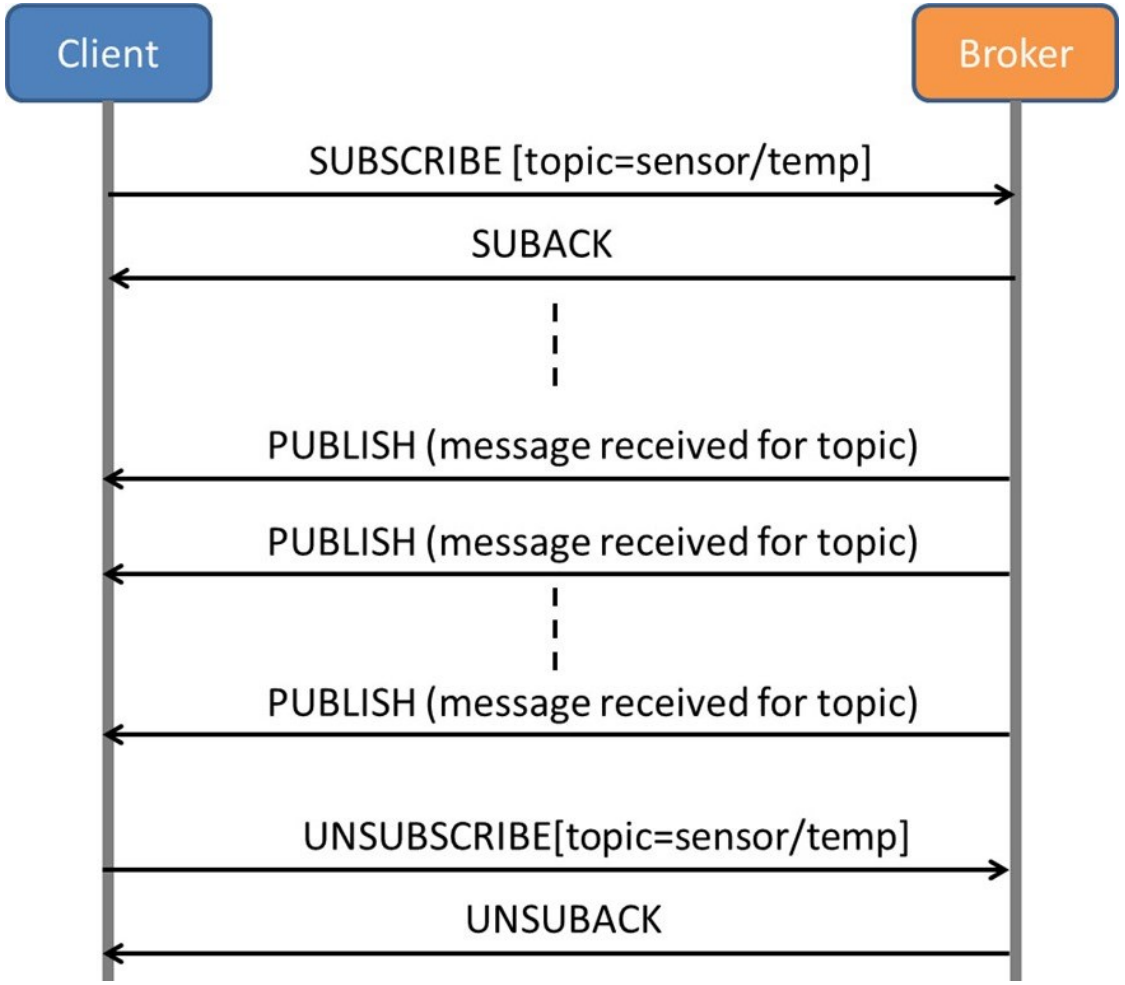
- **DUP flag** – ponovljeno slanje
- **QoS** – definira razinu kvalitete usluge (0, 1 ili 2)
- **RETAIN** – koristi se samo u porukama PUBLISH
  - Kada klijent šalje PUBLISH do brokera, ako je retain flag = 1, broker treba pohraniti poruku, poruka će biti isporučena novim pretplatnicima s odgovarajućom pretplatom (novi pretplatnici ne čekaju novu obavijest, već primaju posljednju poznatu vrijednost).
  - tzv. *retained messages*

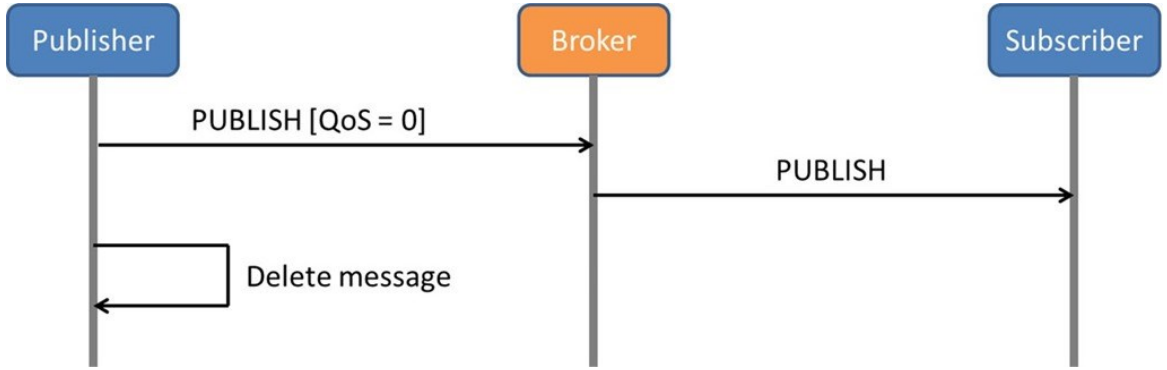
CONNECT/DISCONNECT





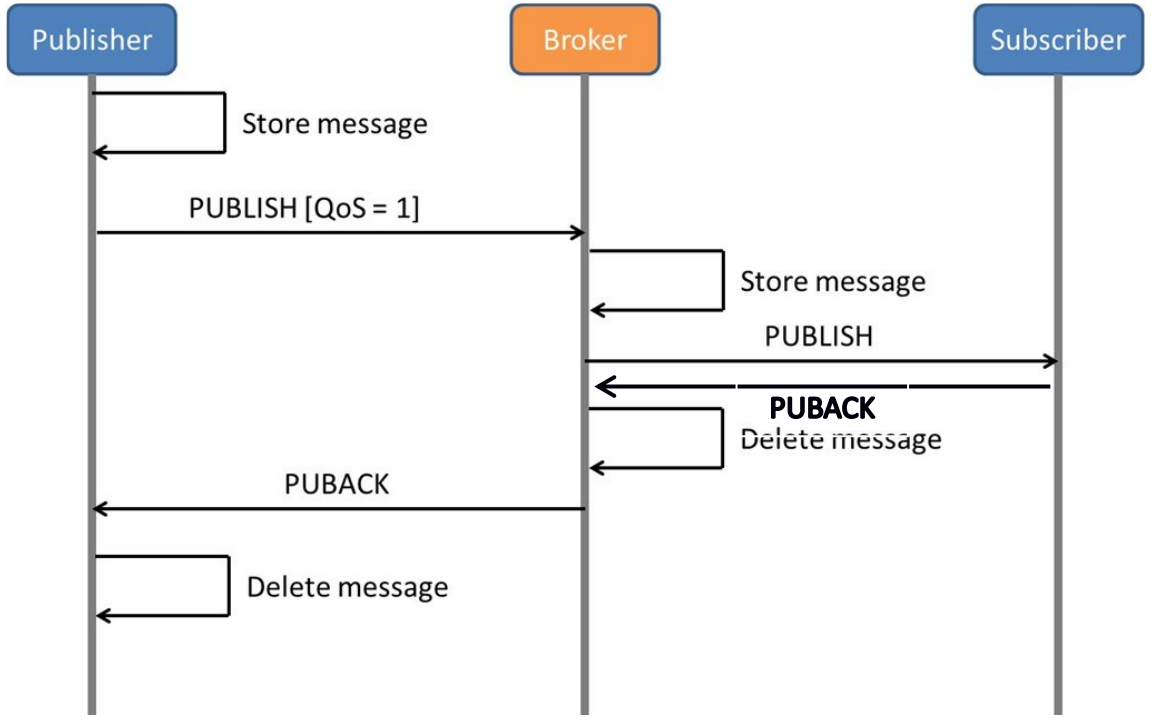
SUBSCRIBE/PUBLISH



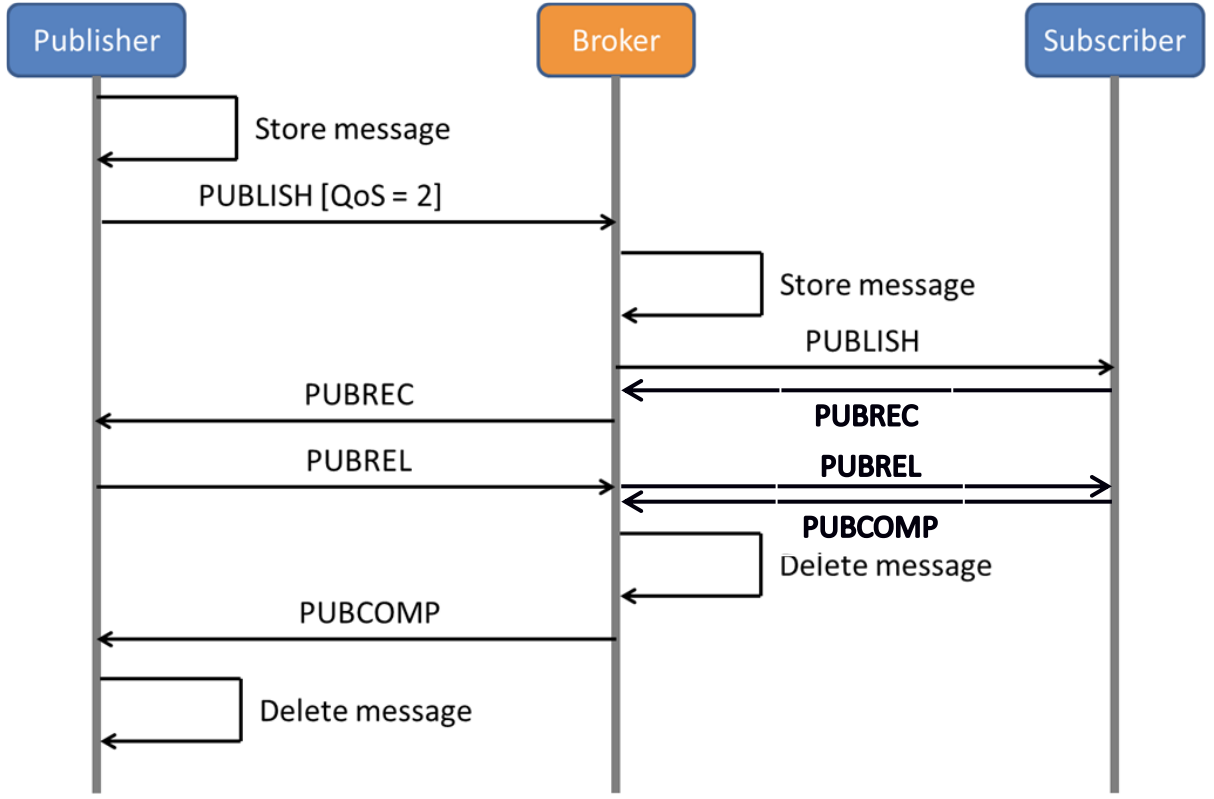


QoS = 0

PUBLISH, QoS>0

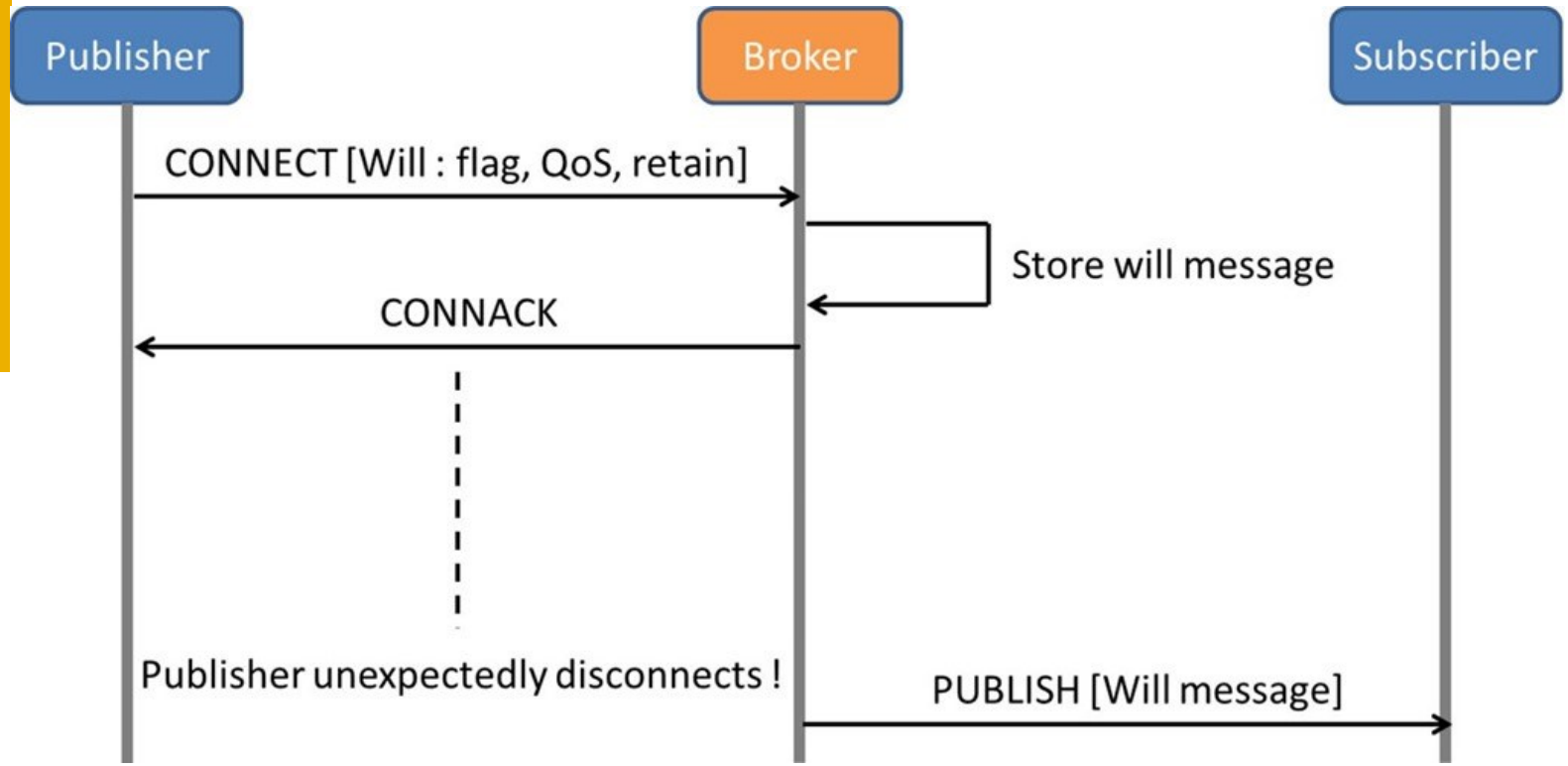


QoS = 1



QoS = 2

#Will message



- Publisher može prilikom stvaranja sjednice definirati *will flag*, *will topic* i *will payload*
- Ako dođe do njegovog nenadanog ispada, *will payload* se isporučuje pretplatnicima

Prednosti i nedostaci MQTT-a

Prednosti

- Messaging protokol prilagođen uređajima i mrežama ograničenih resursa
- Pouzdana isporuka poruka (s obzirom da se temelji na TCP-u) u jednoj sjednici
- Pretplata omogućuje isporuke poruke na više odredišta
- Različiti nivoi QoS-a: 1 i 2 osiguravaju isporuku poruke pretplatnicima

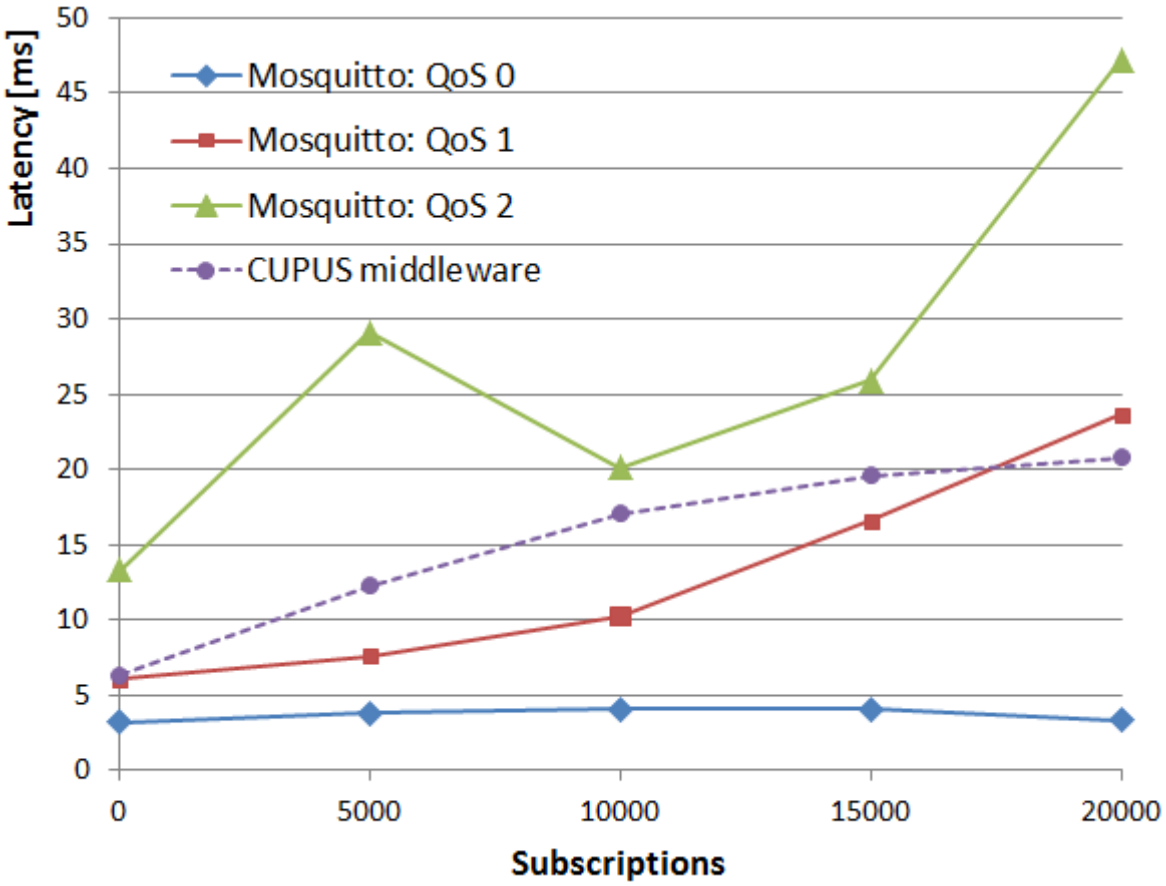
Nedostaci

- Korištenje TCP-a dodaje značajni overhead
  - Uređaj: podržava TCP/IP stack
  - Konekcija između klijenata i broker mora biti kontinuirano aktivna, čak i kada se ne šalju podaci ("keep alive")
- Značajna potrošnja energije na IoT-uređaju, nije pogodan za uređaje s ograničenom količinom energije
  - Postoji varijanta protokola koja se temelji na UDP-u, ali se rjeđe koristi

Usporedba CoAP-a i MQTT

	CoAP	MQTT
Communications Model	Request-Response, or Pub-Sub	Pub-Sub
RESTful	Yes	No
Transport Layer Protocol	Preferably UDP; TCP can be used	Preferably TCP; UDP can be used (MQTT-S)
Header	4 Bytes	2 Bytes
Number of message types	4	16
Messaging	Asynchronous and Synchronous	Asynchronous
Application Reliability	2 Levels	3 Levels
Security	IPSEC or DTLS	TLS
Intermediaries	Yes	Yes (MQTT-S)

Test performanci za MQTT



Usporedba vremena obrade i isporuke poruka svim pretplatnicima za QoS = 0, 1 i 2

MQTT server: Mosquitto
MQTT client: Paho

Izvor: Antonić, Aleksandar; Marjanović, Martina; Skočir, Pavle; Podnar Žarko, Ivana.
Comparison of the CUPUS middleware and MQTT protocol for smart city services // *Proceedings of the 13th International Conference on Telecommunications* / Plank, Thomas (ur.).
Graz : Graz University of Technology, 2015. 1-8

Literatura

1. David Hanes, Gonzalo Salgueiro, Patrick Grossetete, Robert Barton, and Jerome Henry. 2017. IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things (1st ed.). Cisco Press. (6. poglavlje)
2. MQTT Version 5.0, https://docs.oasis-open.org/mqtt/mqtt/v5.0/os/mqtt-v5.0-os.html