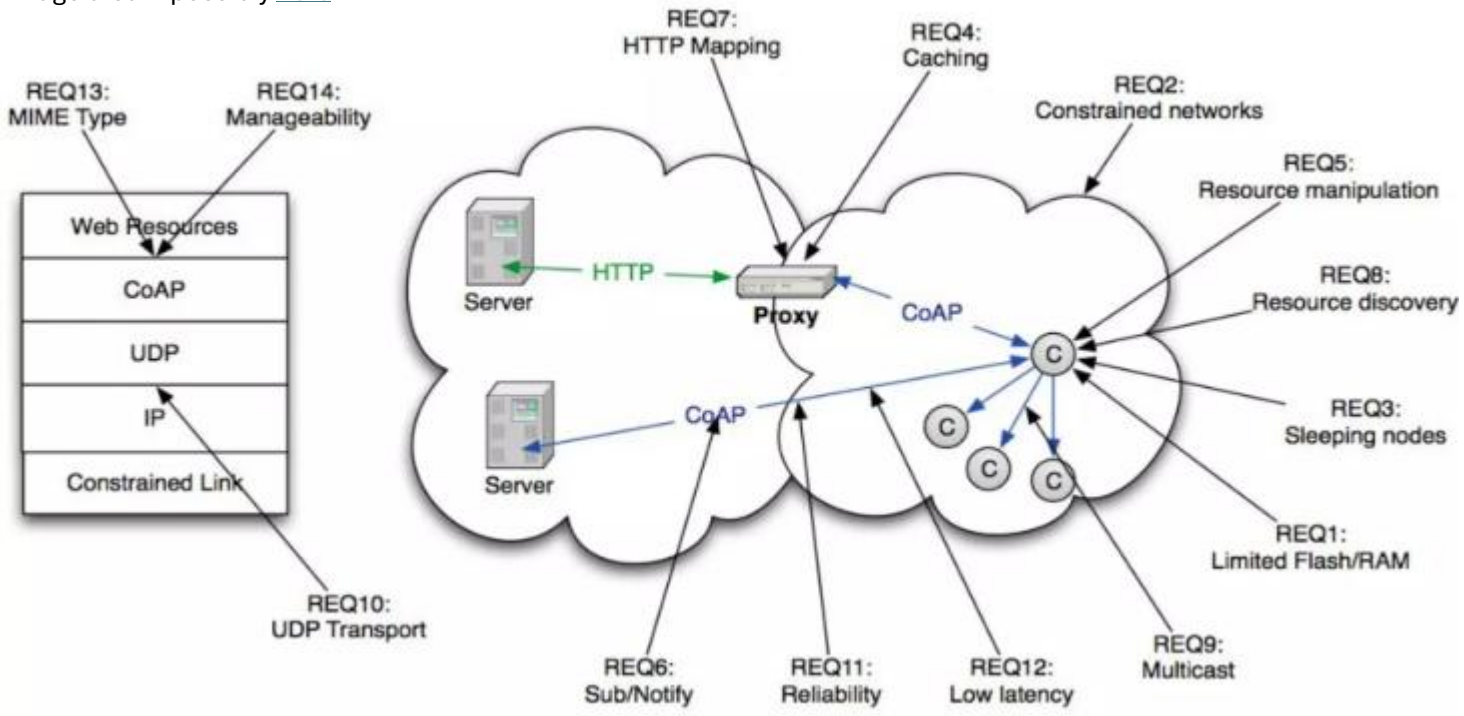
Constrained Application Protocol (CoAP): Intro + Use for Network Telemetry

Image credit: possibly here



Esko Dijk
CoRE WG / IoTconsultancy.nl

IETF 123, Madrid – opsarea
July 24, 2025

Background

- For various use cases a lightweight UDP protocol may be needed:
  - Network telemetry – today's focus
  - Management of low-bitrate (802.15.4) IPv6 mesh networks – my background
  - Management of control plane for memory-constrained routers – ANIMA WG topic
  - Cellular IoT device fleet management

- Often, the choice is to create a custom protocol over UDP or re-use CoAP (RFC 7252)

- Examples of standards that adopted CoAP: LWM2M, Thread

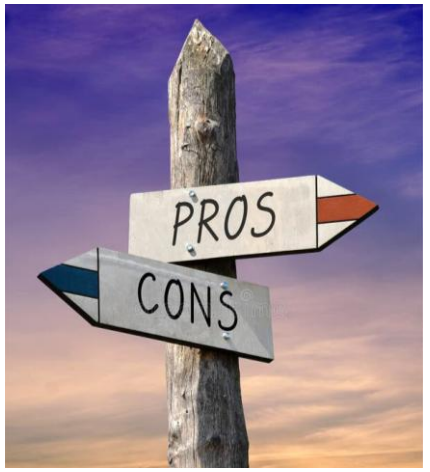


Image credit: Google search
(sorry, no more details)

CoAP Features 1/2

- IANA-assigned port numbers 5683, 5684 ☺
- Compact encoding (headers/options/payloads)
- Multicast support – unlike HTTP
- Selectable message reliability
- Optional security: DTLS, TLS or OSCORE (intro)
  (OSCORE minimizes size, memory & round trips – used in the KNX IoT standard)

CoAP Features 2/2

- Highly modular & extensible – a protocol toolkit

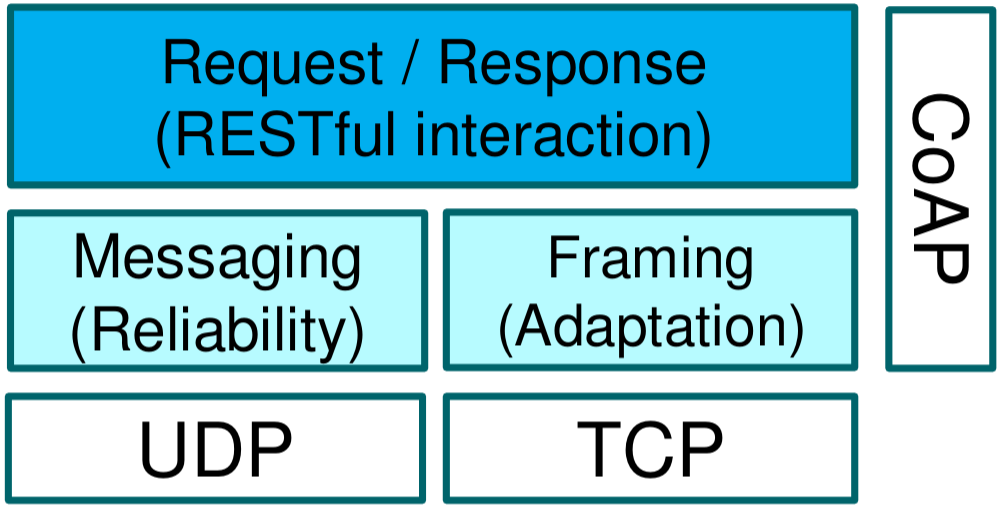
  options, content-formats, protocol parameters, methods, resources, …
  proxies, blockwise data transfer, subscription model (“observe”), …

- REST communication paradigm (see next slide)
- Multiple transports including UDP/TCP (see next slide)

Slide credit: Marco Tiloca, RISE

CoAP

- **Constrained Application Protocol – RFC 7252**
  - Web-transfer protocol, at the application layer
  - Think of it as a “lightweight alternative to HTTP”
  - URI scheme: *coap* (Default port 5683)
    - *coap://<host>[:<port>]/<path>[?<query>]*



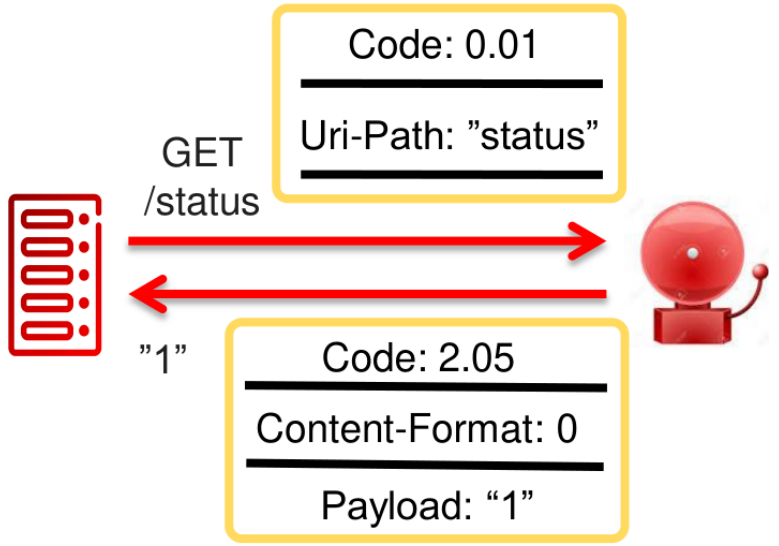
- **Supported transports**
  - Most used is UDP over IP
  - TCP or WebSockets (RFC 8323)
  - SMS, CIoT, Bluetooth, LoRaWAN, IEEE 802.15.4, …

- **RESTful protocol**
  - What matters is the resource status on the server and its transfer
  - Supported methods: GET, POST, PUT, DELETE
  - More recent ones: FETCH and PATCH (RFC 8132)

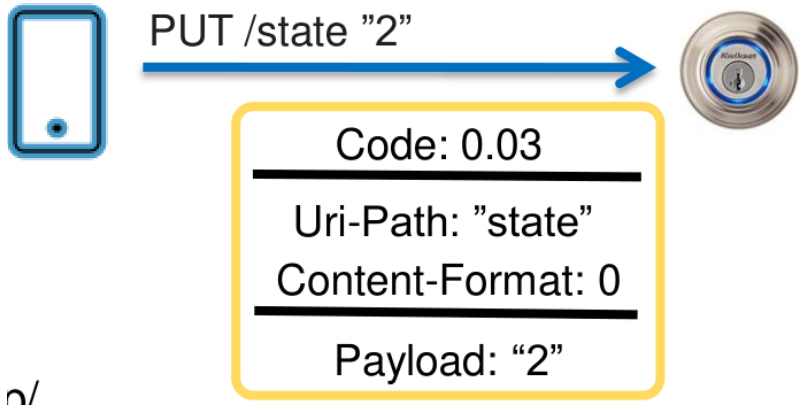
Image credit: Marco Tiloca, RISE

CoAP Examples

- Controller gets status of an “alarm” node
  GET /status



- Phone opens a smart lock
  PUT /state “2”

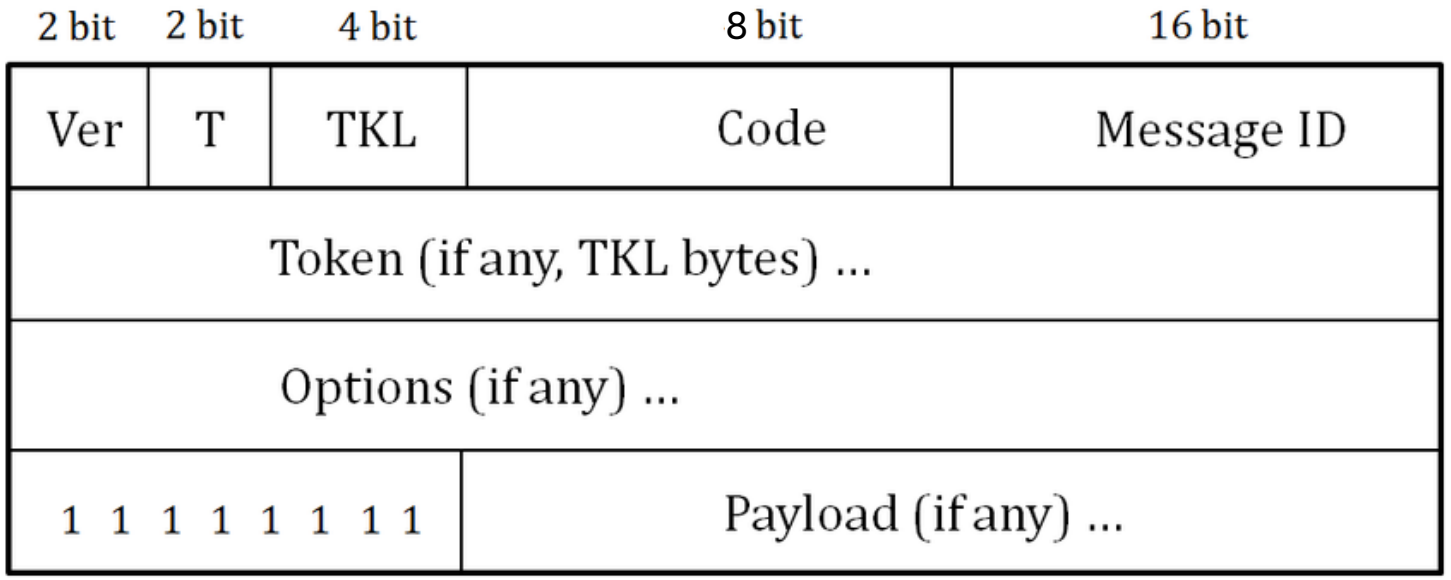


- GET and PUT are the REST methods, used for these examples.

CoAP Header

- UDP Header version shown!
- Ver = Version (0x1)
- T = Type (*message type*)
- Min size: 4 bytes
  No Token, no Options, no Payload
- Typical size: > 10 bytes,
  due to Token + Options + Payload,
  or security.

Image credit: Idrees Zaman



Optional Payload Marker
(present if payload)

```
> IEEE 802.15.4 Data, Dst: 0x7000, Src: 0xd800
> 6LoWPAN, Src: fd00:db9::ff:fe00:d800, Dest: fd00:db9::ff:fe00:fc00
> Internet Protocol Version 6, Src: fd00:db9::ff:fe00:d800, Dst: fd00:db9::ff:fe00:fc00
> User Datagram Protocol, Src Port: 61631, Dst Port: 61631
v Constrained Application Protocol, Confirmable, POST, MID:14302
    01.. .... = Version: 1
    ..00 .... = Type: Confirmable (0)
    .... 0010 = Token Length: 2
    Code: POST (2)
    Message ID: 14302
    Token: 7bfb
  v Opt Name: #1: Uri-Path: a
      Opt Desc: Type 11, Critical, Unsafe
      1011 .... = Opt Delta: 11
      .... 0001 = Opt Length: 1
      Uri-Path: a
  v Opt Name: #2: Uri-Path: sd
      Opt Desc: Type 11, Critical, Unsafe
      0000 .... = Opt Delta: 0
           0010 = Opt Length: 2
```

Example RESTful CoAP: CORECONF

- CoRE WG develops CORECONF
  - a lightweight, COAP version of RESTCONF
  - See draft-ietf-core-comi
  - Includes subscription to Notifications – uses CoAP's "observe" functionality

- Lengthy JSON / XML encodings replaced by compact CBOR (RFC 9254)
  - All string names from YANG models are substituted by numeric IDs

REST-less CoAP? → For telemetry

- Some systems/standards use CoAP without its RESTful paradigm
- This is enabled, for example:
  1. Sender is CoAP client
  2. Receiver is CoAP server
  3. Client sends **POST** request:

     POST /c “<*binary-data*>”

  4. Optionally, server **suppresses sending** the (REST) response,
     e.g. if the request is not using reliable transport.

     → Achieves one-way fire-and-forget communication.

Network Telemetry - Requirements

- Anycast addressing
  - DSCP bits, QoS
  - Dest endpoint change while preserving session state → **Looks possible**
- Minimal message (size) overhead
- Varying message length
  - Permit segmentation over multiple packets → **CoAP block-wise RFC 9177**
- Zero to little storage capability in the network itself
- Cheap, reliable, highly scalable, maintainable, resilient

Benefits CoAP over “new UDP protocol” ?

- For the network telemetry use case

- No definitive answer, but these elements should be considered:
  1. Security, or the option to add lightweight security (OSCORE)
  2. Available Block-wise transfer methods (2 options)
  3. Do we accept the overhead (some # of bytes) –
     dedicated protocol can be more efficient
  4. Does the REST communication pattern help us, or hinder us?