CoAP

For GPRO project, just a page to collect the presentation material, links, etc, for the CoAP simple intro. This is not intended to be a perfect presentation (CoAP is a minor protocol in this project and there are surely other better presentations and pages - please add more links if you find them)

Summary

IETF standard, RFC 7252:
"[CoAP is...] similar to the client/server model of HTTP. However, machine-to-machine interactions typically result in a CoAP implementation acting in both client and server roles"

Summary: CoAP is like HTTP for constrained environments. If you want to use HTTP/RESTful services but more bandwidth efficient, runs on unreliable networks.

- CoAP typically runs on UDP, (HTTP on TCP).
- Thus, no guaranteed delivery of packets. Asynchronous operation.
- Implementation includes a messaging layer, which optionally adds reliability/resend/etc. - Message types: Confirmable, Non-confirmable, Acknowledgement, Reset.
- Logical protocol built on top of those messages: 4 method types, modeled after HTTP: GET, PUT, POST, DELETE
- Command is text in HTTP but binary message code in CoAP. 4-byte header, whole message efficient binary encoding: E.g. [3, slide 16]
- CoAP server role returns response code modeled after HTTP: (201 = Created, 403 = forbidden, 404 = not found, etc.)
- Payload has a known Content-Format, like HTTP Content-Type. HTTP: Send Text-based MIME type. CoAP: Single type code, predefined in specification (binary efficient)
- Proxy between CoAP (IoT devices) and HTTP (rest of web, server-to-server communication) is anticipated. Good picture: [3, slide 10]
- Proxy might include cache (each response can include an expiry time "Max-Age")
- Piggybacked messages (E.g. response message is sent together with ACK)
- Tokens included to enumerate requests (0 to 8 bytes long), referenced by the Ack.
- ? No limits on number of outstanding messages? (I did not see see any mention of a sliding window protocol [2], although the tokens seem to be possible to use that way).
- Randomized tokens suggested to avoid spoofing (seems not a comprehensive countermeasure IMHO)
- URLs are coap(s)://<something>. DTLS (Datagram TLS) for encryption
- Various security related discussion in RFC
- Various additional details

Links to more information

[0] http://coap.technology/
[1] RFC: https://tools.ietf.org/html/rfc7252
[2] https://en.wikipedia.org/wiki/Sliding_window_protocol
[3] https://www.slideshare.net/aniruddha.chakrabarti/coap-web-protocol-for-iot
[4] https://www.slideshare.net/HamdamboyUrunov/the-constrained-application-protocol-coap

⚠️ N.B. The intended license on some of the linked slide decks on slideshare.net is not clear to me. Therefore, no pictures or texts from the slide decks are reproduced here. When using the slideshare links, make your own evaluation on their respective copyright and licensing.