

CoAP in space

draft-gomez-core-coap-space-00

Intended Status: Informational

Carles Gomez

Universitat Politècnica de Catalunya

Sergio Aguilar

Sateliot

1. Introduction

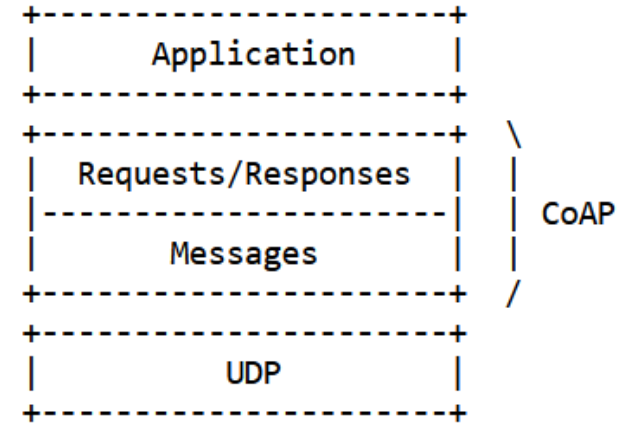
- Deep space communication:
 - Between devices on/orbiting different celestial bodies
 - Long delays, intermittent comm. opportunities...
- IP stack was considered unsuitable for deep space
 - Led to DTN architecture (RFC 4838) and BP (RFC 5050, RFC 9171)
- draft-many-deepspace-ip-assessment:
 - Revisits the assessment on the IP stack for deep space
 - Application layer: HTTP (over QUIC), also CoAP
- Constrained Application Protocol (CoAP):
 - Designed for constrained-node network environments (IoT)
 - Lightweight operation, asynch. message exchanges, flexibility
 - Based on the REST architecture of the web

3.1. Overview and underlying transport

- Originally designed on top of UDP (RFC 7252)

- Message sublayer:
 - Optional reliability
 - Simple congestion control

- Message types:
 - Confirmable (CON)
 - » Needs an Acknowledgment (ACK)
 - » Stop & wait
 - » Timer-based retransmission with exponential back-off
 - Non-confirmable (NON)
 - ACK and RST



- Subsequently, CoAP over reliable transports (RFC 8323)
 - TCP, TLS, WebSockets
 - Significant performance penalty in deep space (e.g., initial handshakes)
 - Reliability is not optional
 - No support for multicast
 - Greater header size

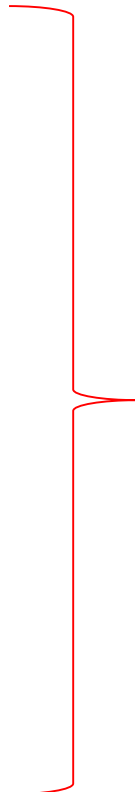
Deep space: CoAP over UDP

3.2. Main CoAP parameters and times relevant to deep space (I/II)

- NSTART
 - Max number of outstanding interactions
 - Default value: 1
 - Greater values possible when some mechanism ensures congestion safety
- ACK_TIMEOUT (AT), ACK_RANDOM_FACTOR (AF)
 - Initial RTO, randomly chosen from $[AT, AT \cdot AF]$
 - Default values (respectively): 2 s, 1.5
 - ACK_TIMEOUT needs to be set to at least the RTT
 - ACK_RANDOM_FACTOR intended to avoid synchronization effects
- MAX_RETRANSMIT
 - Default value: 4
 - Due to exponential back-off, lower than default may be suitable
 - Congestion control, needed in deep space environments?

3.2. Main CoAP parameters and times relevant to deep space (II/II)

- MAX_LATENCY
 - Max time since a datagram is sent until it is received
 - Defined as 100 s
- EXCHANGE_LIFETIME
 - Max time since first transmission attempt of a CON until its ACK
 - Default value: 247 s
- NON_LIFETIME
 - Max time since a NON message is sent until it is received
 - Default value: MAX_LATENCY (i.e., 100 s)



At least 2
orders of
magnitude
greater for
deep space

- Note: CoAP implementations using 8-bit timers may need to be adapted for deep space

4. Observe

- CoAP option (RFC 7641)
- Allows a server to send notifications
 - Representation of the current state of a resource to observers
 - Observers need to initially register their interest
- The client does not have to send a request to receive each notification
 - Beneficial in deep space

5.1. Block-wise transfers overview

- Block-wise transfers allow to carry large payloads
- Two Block-wise specifications:
 - RFC 7959
 - Based on CON messages
 - Follows a stop & wait pattern
 - RFC 9177
 - Based on NON messages
 - No stop & wait limitation
 - Recovery of multiple missing blocks
 - » Can be reported at once in a single CoAP message
 - More suitable for deep space

5.2. Block-wise main parameters

- MAX_PAYLOADS
 - Max number of consecutive blocks without response
 - Default value: 10
 - Value for deep space?
- NON_TIMEOUT_RANDOM
 - Min time between two consecutive sets of MAX_PAYLOADS blocks
 - $\text{NON_TIMEOUT_RANDOM} = \text{NON_TIMEOUT} * \text{ACK_RANDOM_FACTOR}$
 - NON_TIMEOUT is ACK_TIMEOUT (Default value: 2 s)
 - ACK_RANDOM_FACTOR is set to 1.5
- NON_RECEIVE_TIMEOUT
 - Initial time a receiver waits for a missing block before requesting retransmission
 - Default value: $2 * \text{NON_TIMEOUT}$
- NON_MAX_RETRANSMIT
 - Maximum number of requests for retries
 - Default value: MAX_RETRANSMIT

Same as Section 3.2 apply for deep space

6. Security

- CoAP base spec (RFC 7252) defines a binding to DTLS
 - DTLS security modes:
 - NoSec, PreSharedKey, RawPublicKey, and Certificate
 - Mandatory to implement: NoSec, RawPublicKey
- Also, CoAP option (RFC 8613):
 - Object Security for Constrained RESTful Environments (OSCORE)
 - End-to-end application-layer payload protection
 - Used to protect CoAP group communication
 - Shared security context, may be based on pre-shared materials
 - Avoids initial handshake and related performance penalty, critical in deep space

Other

- Forward Error Correction (FEC) for CoAP?
 - No effort in the IETF
 - Discussions on it have been reported
 - In the context of firmware updates
- No IANA considerations

Thanks!

Questions? Comments?

Carles Gomez

Universitat Politècnica de Catalunya

Sergio Aguilar

Sateliot