

R-Pi

Team Emertxe



IoT Protocols

CoAP



CoAP

Introduction

- Is one of the latest application layer protocol developed by IETF for smart devices to connect to Internet
- Many devices exist as components in vehicles and buildings with constrained resources, it leads a lot of variation in power computing, communication bandwidth etc
- lightweight protocol CoAP is intended to be used and considered as a replacement of HTTP for being an IoT application layer protocol

Table 1 protocols in different layers

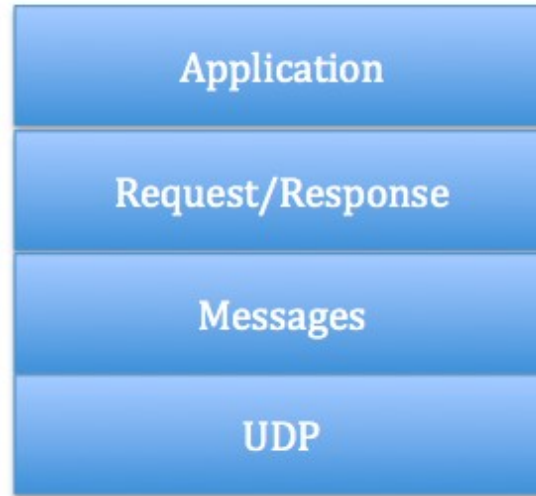
Application layer	HTTP, CoAP, EBHTTP, LTP, SNMP, IPfix, DNS, NTP, SSH, DLMS, COSEM, DNP, MODBUS
Network/Communication layer	IPv6/IPv4, RPL, TCP/UDP, uIP, SLIP, 6LoWPAN,
PHY/MAC layer	IEEE 802.11 Series, 802.15 Series, 802.3, 802.16, WirelessHART, Z-WAVE, UWB, IrDA, PLC, LonWorks, KNX



- Constrained web protocol fulfilling M2M requirements
- Security binding to DTLS (Datagram Transport Layer Security)
- Asynchronous message exchanges
- Low header overhead and parsing complexity
- URI and content type support
- Simple proxy and caching capabilities
- UDP binding with optional reliability supporting unicast and multicast requests

CoAP

Structure Model



- CoAP interactive model is similar to HTTP's client/server model
- The bottom layer is Message layer that has been designed to deal with UDP and asynchronous switching
- The request/response layer concerns communication method and deal with request/response message

CoAP: Structure Model

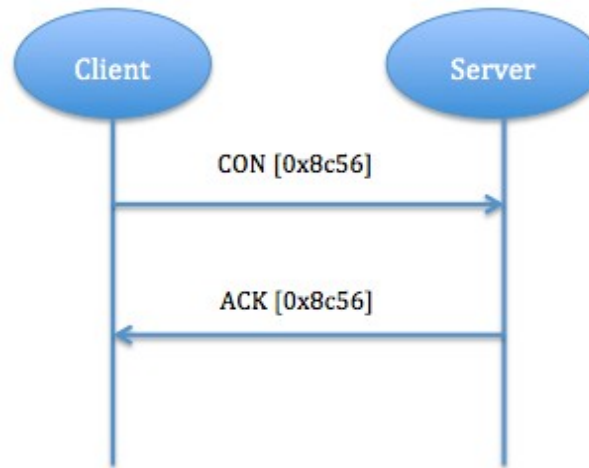
Message Layer Model

- Message Layer supports 4 types of messages
 - CON (confirmable)
 - NON (non-confirmable)
 - ACK (Acknowledgement)
 - RST (Reset)

CoAP: Structure Model

Message Layer Model

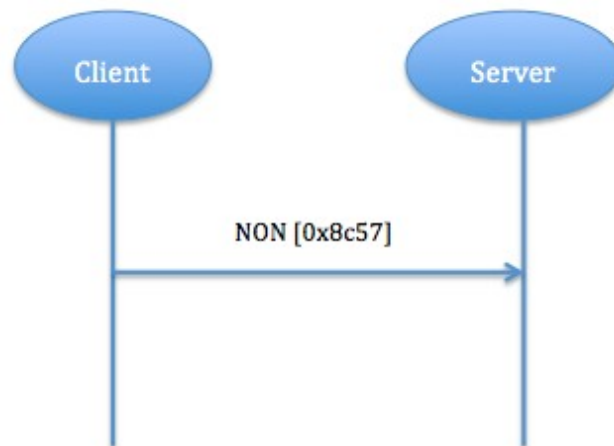
- Reliable message transport
 - Keep retransmission until get ACK with the same message ID (like 0x8c56 in fig.)
 - If recipient fail to process message, it responses by replacing ACK with RST.
 - Fig shows a reliable message transport



CoAP: Structure Model

Message Layer Model

- UnReliable message transport
 - A message that does not require reliable transmission can be sent as a Non-confirmable Message (NON).
 - Transporting with NON type message.
 - It doesn't need to be ACKed, but has to contain message ID for supervising in case of retransmission.
 - Messages is not acknowledged, but has a message ID for duplicatedetection
 - If recipient fail to process message, server replies RST.
 - Fig shows a reliable message transport



CoAP: Structure Model

Message Layer Model

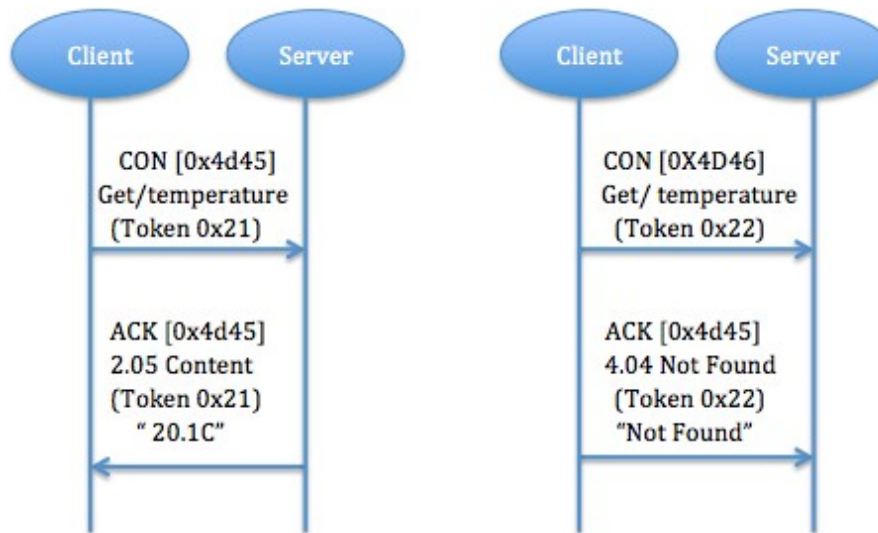


- UnReliable message transport
 - Unlike HTTP, CoAP deals with these interchanges asynchronously over adatagram-oriented transport, such as User Datagram Protocol (UDP), and thus, a NON might get lost without the client and the server noticing it.

CoAP: Structure Model

Request/Response Layer Model

- Piggy-backed
 - Client sends request using CON type or NON type message and receives response ACK with confirmable message immediately
 - fig. shows successful response, ACK contain response message (identify by using token), for failure response, ACK contain failure response code

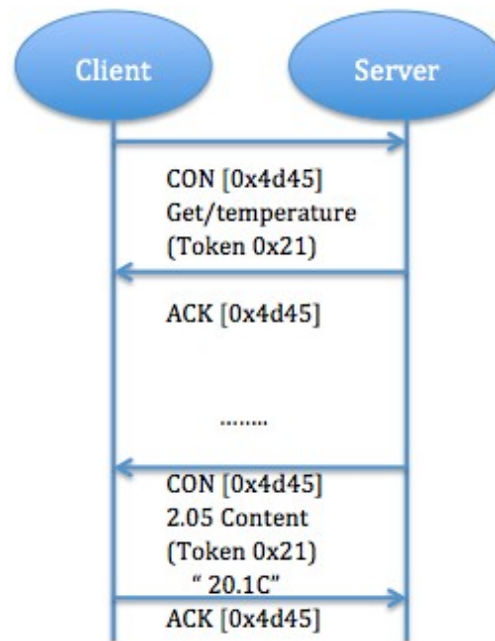


The successful and failure response results of GET method

CoAP: Structure Model

Request/Response Layer Model

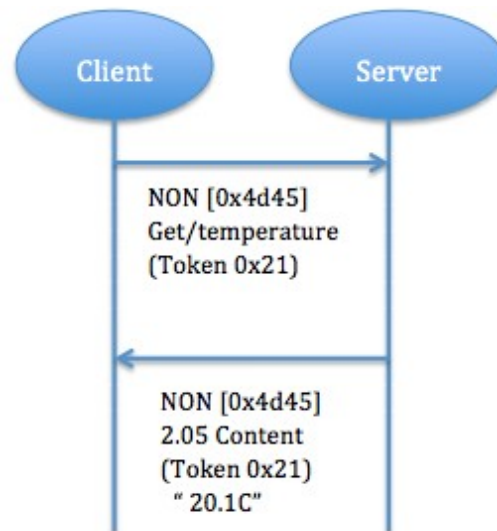
- Separate response
 - If server receive a CON type message but not able to response this request immediately, it will send an empty ACK in case of client resend this message
 - When server ready to response this request, it will send a new CON to client and client reply a confirmable message with acknowledgment
 - ACK is just to confirm CON message, no matter CON message carry request or response



CoAP: Structure Model

Request/Response Layer Model

- Non confirmable request and response
 - unlike Piggy-backed response carry confirmable message, in Non confirmable request client send NON type message indicate that Server don't need to confirm
 - Server will resend a NON type message with response



CoAP: Structure Model

Message Format

- CoAP is based on the exchange of compact messages that, by default, are transmitted over UDP (i.e. each CoAP message occupies the data section of one UDP datagram)
- Message of CoAP uses simple binary format
- Message= fixed-size 4-byte header plus a variable-length Token plus a sequence of CoAP options plus payload

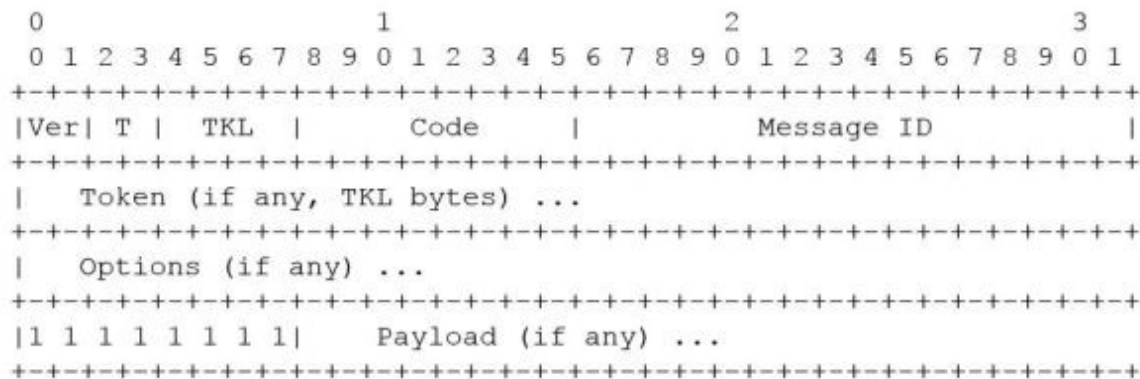


Figure 7: Message Format

CoAP: Structure Model

Message Format

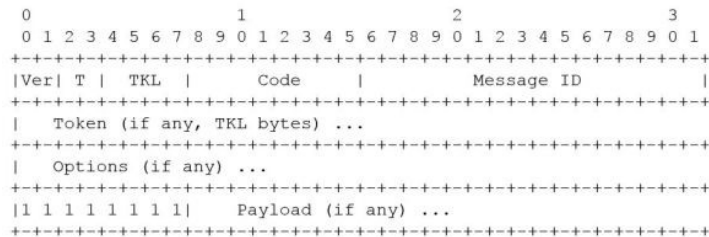


Figure 7: Message Format

Ver	It is a 2 bit unsigned integer indicating the version
T	2 bit unsigned integer indicating the message type: 0 confirmable, 1 non-confirmable
OC / TKL	Token Length is the token 4 bit length
Code	It is the code response (8 bit length)
MessageID	It is the message ID expressed with 16 bit

CoAP: Security + Application

Why use DTLS for CoAP Security



- CoAP is now becoming the standard protocol for IoT applications
- Security is important to protect the communication between devices
- A security protocol DTLS is introduced
- There are three main elements when considering security, namely integrity, authentication and confidentiality, DTLS can achieve all of them
- DTLS employ TCP, which is too complex
- DTLS solves two problems
 - reordering and packet lost
- It adds three implements
 - packet retransmission
 - assigning sequence number within the handshake
 - replay detection

CoAP: Security + Application

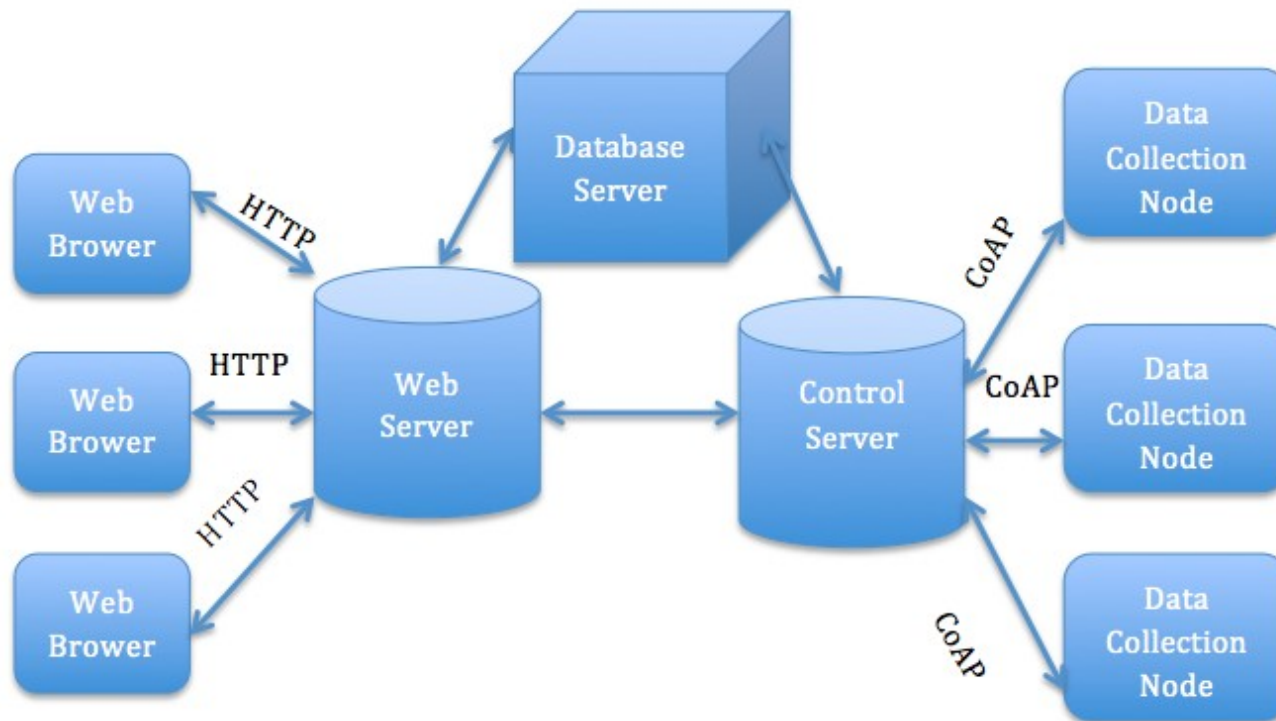
Why use DTLS for CoAP Security

- DTLS in application layer protect end-to-end communication
- No end-to-end communication protection will make it easy for attacker to access to all text data that passes through a compromised node
- DTLS also avoids cryptographic overhead problems that occur in lower layer security protocols

Application (CoAP, XML)
Security (DTLS)
Transport (UDP)
Network (IPv6)
PHY/MAC (IEEE 802.15.4)

CoAP: Security + Application

Application for Smart Homes



THANK YOU