Client CoAP
Aplicație demonstrativă

Proiect (în curs de a fi) realizat de:
Butnaru Raimond-Eduard (1306B)
Zacordoneț Andrei (1306B)

Îndrumător proiect:
ș.l. dr.ing. Botezatu Nicolae-Alexandru

Cuprins

Ce este CoAP?

1. Caracteristici generale

CoAP (Constrained Application Protocol) este un protocol specializat pentru aplicațiile dispozitivelor care sunt constrânse de diferiți factori. Unul din scopurile principale CoAP este de a proiecta un protocol web generic pentru comunicarile de tip machine-to-machine (M2M), care sunt limitate fie de capacitățile hardware (cum ar fi limitări RAM și ROM) sau de limitările de mediu (căderi de tensiune, pierdere de pachete etc.).

Alte caracteristici principale CoAP sunt:

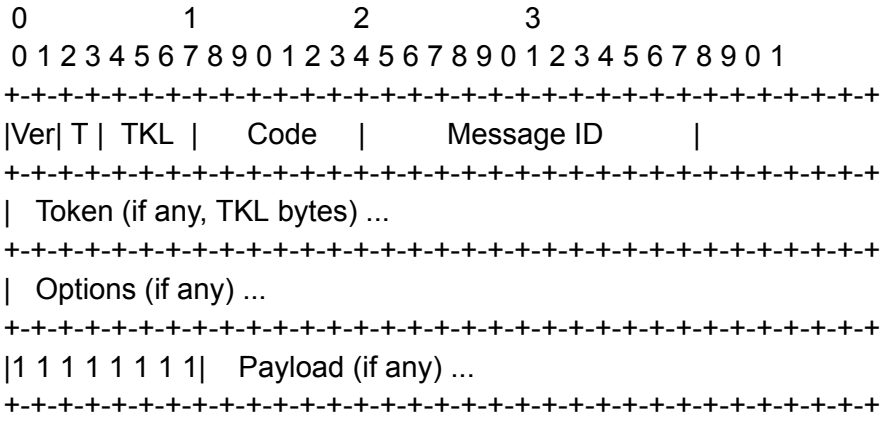
- Schimbul de mesaje asincron;
- Complexitatea analizei (parsing) redusă;
- Capacitatea de a integra proxy și cache într-un mod cât mai simplu.

Comparat deseori cu HTTP, CoAP pune în practic și interschimbarea de mesaje în mod asincron. Față de HTTP, interacțiunile CoAP conferă mașinilor care comunică între ele sa aibă atât rol de client cât și rol de server.

Modelul de mesaje CoAP se bazează pe schimbul de mesaje utilizând pachete de tip UDP (User Datagram Protocol).

2. Formatul mesajelor

Mesajele sunt criptate într-un format binar simplu. Header-ul mesajului are lungime fixă de 4 octeți, urmat de un „Token” cu lungime variabilă (între 0 la 8 octeți). După token se poate găsi un câmp cu opțiuni CoAP, cât și un câmp de „Payload”, ambele fiind de natură opțională.



2.1. Header

Este alcătuit din 4 octeți, după cum urmează:

A. Octet 1 - are 3 câmpuri în alcătuire (versiunea, tipul mesajului și lungimea token-ului):
   a. Biții 0-1 reprezintă versiunea protocolului (01 respectând RFC-7252)
   b. Biții 2-4 reprezintă tipul mesajului (Confirmable 00, Non-confirmable 01, Acknowledge 10, Reset 11)
   c. Biții 4-7 reprezintă lungimea token-ului (lungimile 1001-1111 sunt rezervate)
B. Octet 2 - format din 2 câmpuri (class și detail)indică codul de Request/Response în formatul zecimal cu virgulă class.detail (c.dd), codul 0.00 reprezentând Empty Message:
   a. Biții 0-2 intră în alcătuirea clasei (Request 0, Success Response 2, Client Error Response 4, Server Error Response 5)
   b. Biții 3-7 aparțin campului de detail:
      i. În cazul clasei Response (0): GET 0.01, POST 0.01, PUT 0.03, DELETE 0.04
      ii. În cazul clasei Request (2): CREATED 2.01, DELETED 2.02, VALID 2.03
C. Octeții 3-4 - câmpul Message ID: utilizat pentru identificarea mesajelor duplicate, permițând și fiabilitatea transmisiunii

2.2. Token

Rolul token-ului este acela de a corela cererile cu răspunsurile. Astfel, cererea respectiv răspunsul primit vor avea același token. Având lungimea maximă de 8 octeți, din motive de securitate, token-ul trebuie să conțină cel puțin 32 de biți generați aleator în cadrul conexiunii la internetul uzual.

2.3. Opțiuni

Respectând formatul TLV (type-length-value), fiecărei opțiuni îi este asignat un număr (onum - option number). Astfel, în alcătuirea primului octet intră:

A. Option Delta (biții 0-3):
   - OptDelta < 13: Cazul normal
   - OpDelta = 13: OpDelta = nextByteValue + 13
   - OpDelta = 14: OpDelta = next2BytesValue + 255 + 14
   - OpDelta = 15: Payload Marker arata sfarsitul sectiunii de optiuni
B. Option Length (biții 4-7)
   - OpLen < 13: OpValue are lungimea precizata si urmeaza fix dupa
   - OpLen = 13: OpLen = nextByteValue + 13
   - OpLen = 14: OpLen = next2BytesValue + 255 + 14
   - OpLen = 15: Eroare

Următorii octeți sunt ocupați după cum urmează: Option Delta Extended (0-2 octeți), Option Length Extended (0-2 octeți) și Option Value (0 sau mai mulți octeți).

Opțiunile se împart în 2 categorii:

1. Elective: - opțiunile nerecunoscute sunt ignorate silențios
2. Critical: -CON => se trimite 4.02 (Bad Option) și un diagnostic în payload
   -NON => mesajul este respins

2.4. Payload

Aici este situat mesajul propriu-zis, lungimea acestuia trebuie să fie corelată cu datagram length (1024 octeți). Totodată, primul octet se numește payload marker și are valoarea 0xFF (11111111_2)

3. Transmiterea mesajelor

3.1. Tipuri de mesaje

În cadrul CoAP există 4 tipuri de mesaje: Confirmable (CON), Non-confirmable (NON), Acknowledge (ACK) și Reset (RST):

A. Confirmable (CON):
   - este trimis până la primirea unui mesaj de tipul ACK sau RST;
   - cere trimiterea unui mesaj de tipul ACK sau RST, acestea din urmă trebuind să aibă același ID.

B. Non-confirmable (NON):
   - trimiterea lui nu necesită un mesaj de tip ACK sau RST.

C. Acknowledge (ACK):
   - nu indică succesul sau reușita unei cereri, arată doar faptul că cererea a ajuns la endpoint.

D. Reset (RST):
   - indică primirea unei cereri (CON sau NON) dar anumite detalii necesare lipsesc.

3.2. Fiabilitate

Proprietatea de fiabilitate a mesajelor se datorează celor 2 tehnologii implementate la nivelul clientului și anume Exponential Backoff și Duplicate Detection.

A. Exponential Backoff:
   - este un procedeu prin care mesajele de tip CON sunt retransmise cu o frecvență mai redusă odată cu trecerea timpului;

- 2 parametrii modelează această funcție: timeout și retransmission counter:
    i. Timeout: număr random în intervalul (2, 2 * 1.5) exprimat în secunde. După expirarea sa se retransmite masajul si parametrul își dublează valoarea (timeout *= 2).
    ii. Retransmission Counter: începe de la 0 și se incrementează la fiecare retransmitere a mesajului până este atinsă valoarea MAX_RETRANSMIT = 4.

B. Duplicate Detection: daca mai multe cereri au același ID atunci ce va executa o singură dată mesajul (excepție fac mesajele idempotente)

Tot aici apare și noțiunea de control al congestionării, după EXCHANGE_LIFETIME = 247 s clientul nemaiașteptând un răspuns la o cerere de tip CON.

3.3. Modelul Cerere/Răspuns

A. Cereri

Există 4 tipuri de cereri GET, POST, PUT și DELETE.

1. GET: - returnează resursa specificată de URI (uniform resource identifier)
   - în caz de succes se returnează codurile 2.05 (Content) și 2.03 (Valid)
   Această opțiune este sigură și idempotentă.
2. POST: - returnează 2.01 (Created) și URI-ul resursei create
   Această opțiune nu este nici sigură nici idempotentă.
3. PUT: - returnează 2.04 (Changed) iar dacă resursa nu exista returnează 2.01 (Created), în orice alt caz returnând un cod de eroare
   Această opțiune nu este sigură dar este idempotentă.
4. DELETE: - returnează 2.02 (Deleted)
   Această opțiune nu este nici sigură nici idempotenta.

B. Răspunsuri

1. Piggybacked Response: - răspunsul este returnat în mesajul ACK

2. Separate Response: - răspunsul este oferit după o perioadă de timp într-un mesaj CON sau NON
3. NON: - dacă cererea este de NON și răspunsul trebuie să fie tot NON

3.4. Schema URI

URI (Uniform Resource Identifier) este o secvență de caractere care identifică unic fiecare resursa.

CoAP-URI = “coap:” “//” host [“:”port] path-abempty [“?” query]

Scopul proiectului

Scopul proiectului este implementarea unei aplicații demonstrative care să implementeze protocolul CoAP, partea de client.

La nivelul aplicației, clientul va putea accesa, prin intermediul interfeței, o gamă de operații standard pentru accesarea fișierelor și directorilor stocate pe server (acces la fișier, descărcare, creare, încărcare, ștergere, mutare etc.).

Tot la nivelul clientului, va fi implementată și navigarea prin fișiere și directori prin intermediul interfeței.

Vor fi implementate și codurile CoAP, metodele GET, POST și o a treia metodă propusă (TBA) alături de codurile de răspuns relevante pentru aplicație.

Bibliografie

CoAP Client

https://en.wikipedia.org/wiki/Constrained_Application_Protocol

https://github.com/Tanganelli/CoAPthon

https://pypi.org/project/aiocoap/

https://www.rfc-editor.org/rfc/rfc7252.html

https://cqr.company/wiki/protocols/constrained-application-protocol-coap/

https://github.com/akib1162100/mac/blob/master/The%20TCP-IP%20Guide%20-%20A%20Comprehensive%20Illustrated%20Internet%20Protocols%20Reference.pdf

https://docs.iotify.io/temp/protocol-settings/coap

https://jonathanberi.medium.com/a-field-guide-to-coap-part-1-75576d3c768b

https://datatracker.ietf.org/doc/html/rfc7252

RFC 7959 (blockwise transfer)

github.com/nicolaebotezatu

https://github.com/TUIASI-AC-IoT/proiect-11

https://docs.python.org/3/library/select.html