** UNIVERSITATEA TEHNICĂ “GHEORGHE ASACHI” IAŞI, FACULTATEA DE AUTOMATICĂ ȘI A picture containing text

Description automatically generated**

**CALCULATOARE, SPECIALIZAREA CALCULATOARE ŞI TEHNOLOGIA INFORMAŢIEI**

**Rețele de calculatoare - proiect**

**Aplicație pentru monitorizarea resurselor SO**

**(Client MQTT)**

**Coordonator: Studenti:**

Prof. Nicolae-Alexandru Botezatu Belciug Răzvan, grupa 1308A

Vecliuc Draia-Teodora, grupa 1307B

**Cuprins documentație:**

**Capitolul 1: Descriere generală**

* 1.1 MQTT
* 1.2 Istoric
* 1.3 Conceptele de baza
* 1.4 Beneficiile
* 1.5 Autentificare și autorizare cu Mosquito

**Capitolul 2: Client MQTT**

2.1  Informații generale/Pachetele de control

2.2  Mecanism Keep Alive

2.3  Quality of Service:

* 2.3.1 QoS 0
* 2.3.2 QoS 1
* 2.3.3 QoS 2

2.4  Last Will Testament

**Capitolul 3 : Bibliografie**

**Capitolul 1 : Descriere Generală**

1.1. MQTT

MQTT(MQ Telemetry Transport) este un protocol de rețea de tip publish-subscribe care permite transportul de mesaje între două dispozitive. Este ideal pentru conectarea la distanță a dispozitivelor a căror limitări impun dimensiuni reduse ale codului sursă sau există o rețea cu o lățime de bandă limitată.

1.2 Istoric

Protocolul MQTT a fost inventat de Andy Stanford-Clark și Arlen Nipper (Arcom, acum Cirrus Link) în anul 1999. Scopul inițial al metodei de comunicare a fost acela de a permite dispozitivelor de monitorizare utilizate în industria petrolului și a gazelor să își trimită datele către serverele la distanță. În multe cazuri, astfel de dispozitive de monitorizare au fost utilizate în locații îndepărtate, unde orice fel de linie fixă, conexiune cu fir sau conexiune de transmisie radio ar fi dificilă sau imposibil de stabilit. La acea vreme, singura opțiune pentru astfel de cazuri era comunicațiile prin satelit, care erau foarte scumpe. Cu mii de senzori în domeniu, industria a avut nevoie de o formă de comunicare care să poată oferi date fiabile pentru utilizare, folosind în același timp o lățime de bandă minimă.

 „MQ” din MQTT provine din seria IBM MQ, un produs dezvoltat de IBM pentru a suporta transportul de telemetrie MQ. Când Andy și Arlen și-au creat protocolul, l-au numit după produsul IBM. MQTT nu este o soluție tradițională de așteptare a mesajelor deși este posibil să puneți mesaje în coadă în anumite cazuri. Timp de  zece ani, IBM a folosit protocolul intern până când a lansat MQTT 3.1 ca versiune free în 2010.

1.3 Conceptele MQTT-ului:

* Publish/Subscribe - este conceptul care stă la baza MQTT-ului. Unul dintre cele două dispozitive publică mesajele către un topic, iar altul se abonează la acel topic pentru a primi mesajele trimise.
* Mesajele - sunt informațiile pe care vrem să le schimbăm între dispozitive.
* Topic-urile - reprezintă o modalitate în care specificăm locul unde este publicat un mesaj, iar în momentul în care publicăm să specificăm ce mesaje se doresc a fi recepționate atunci când facem subscribe.

* Broker - este cel responsabil de primirea și filtrarea mesajelor. De asemenea, el este cel care decide cine e interesat de ele și după le publica tuturor clienților care au dat subscribe.

1.4 Beneficiile

* transmisie eficientă a datelor și rapid de implementat datorită faptului că este un protocol ușor;
* utilizare redusă a rețelei, datorită pachetelor de date minimizate;
* distribuirea eficientă a datelor;
* implementarea cu succes a teledetecției (complex de tehnici de utilizare pentru prelucrarea de la distanță a unor date cu privire la obiecte sau fenomene ) și controlului;
* livrarea rapidă și eficientă a mesajelor;
* utilizează cantități mici de putere, ceea ce este bun pentru dispozitivele conectate;
* optimizează lățimea de bandă a rețelei.

1.5 Autentificare și autorizare cu Mosquitto

Mosquito este un broker de mesaje care implementează protocolul MQTT. Este un broker lightweight și poate fi utilizat de către toate dispozitivele cu o singura placă de putere redusă la servere complete.

Acesta permite crearea și utilizarea unor metode de autentificare și autorizare prin id (nume de utilizator) și parolă.

**Capitolul 2 : Client MQTT**

2.1 Informații generale/Pachetele de control

Un client MQTT reprezintă o instanță ce se poate conecta cu un broker MQTT utilizând diferite pachete.

* CONNECT: utilizat pentru a loga utilizatorul la server
* CONNACK: răspunsul serverului la primirea pachetului CONNECT
* PUBLISH: publică mesajele către un anumit topic
* PUBACK : răspunsul la primirea pachetului PUBLISH cu QoS 1
* PUBREC: prima parte a răspunsului la primirea unui pachet PUBLISH cu QoS 2
* PUBREL:  a doua parte a răspunsului la primirea unui pachet PUBLISH cu QoS 2
* PUBCOMP:  a treia parte a răspunsului la primirea unui pachet PUBLISH cu QoS 2
* SUBSCRIBE: reprezinta  pachetul trimis către abonarea la diferite topicuri
* SUBACK: răspunsul la primirea pachetului SUBSCRIBE
* UNSUBSCRIBE: reprezinta pachetul trimis catre dezabonarea la diferite topicuri
* UNSUBACK: răspunsul  la primirea unui pachet de UNSUBSCRIBE
* PINGREQ: trimiterea unui ping către server
* PINGRESP: răspunsul serverului la primirea unui ping/ unui pachet de tip PINGREQ
* DISCONNECT: pachet de notificare de deconectare de la broker

2.2 : Mecanism KeepAlive

MQTT include o funcție de KeepAlive care oferă o soluție pentru rezolvarea problemei conexiunilor pe jumătate deschise. Mecanismul KeepAlive este cel care asigură legătura broker și client și faptul că aceasta este încă deschisă. De asemenea, ei sunt conștienți că sunt conectați. Când clientul stabilește o conexiune cu brokerul, clientul îi comunică acestuia un interval de timp. Brokerul ia intervalul și îl multiplică cu 1,5 pentru a determina durata de timp care poate trece între mesajele de la client. De  exemplu, pentru un interval de 30 de secunde va rezulta un interval de Keep Alive de 45 de secunde. Intervalul de KeepAlive este o valoare numerică care indică numărul de secunde în care un client poate fi inactiv, dar considerat că funcționează corect. Intervalul maxim de păstrare în viață care poate fi specificat este de 18 ore, 12 minute și 15 secunde. Setarea valorii la zero dezactivează funcționalitatea KeepAlive.

Este de responsabilitatea clientului să se asigure că intervalul între pachetele de control trimise nu depășește valoarea KeepAlive. Cât timp mesajele sunt schimbate frecvent și intervalul de KeepAlive nu este depășit, nu este nevoie să se trimită  un mesaj suplimentar pentru a stabili dacă conexiunea este încă deschisă.

În absența trimiterii oricăror alte pachete de control, clientul trebuie să trimită un pachet de tip PINGREQ pentru a informa brokerul că este disponibil și pentru a verifica dacă  și brokerul este. Brokerul răspunde la mesajul PINGREQ trimițând clientului un mesaj PINRESP.  Brokerii care nu primesc, cel puțin, un mesaj PINGREQ de la un client în intervalul calculat de Keep Alive  întrerup conexiunea.

Funcționalitatea de Keep Alive este încă un motiv pentru care MQTT este o alegere excelentă pentru rețelele moderne distribuite și comunicarea între dispozitive. Cu o configurație adecvată, puteți proiecta o implementare de rețea viabilă, care poate rezista cazurilor de conectivitate nesigură (un senzor conectat la rețea) .

2.3 Quality of Service

Quality of Service (QoS) este un acord între expeditorul  și receptorul unui mesaj care definește garanția de livrare pentru un mesaj specific. Există 3 nivele de în MQTT:

* At most once (QoS 0)
* At least once (QoS 1)
* Exactly once (QoS 2)

Atunci când un client publică mesajul către broker definește un QoS,  dacă acesta este  mai mare decât QoS-ul pe care îl  are un  client abonat  brokerul decide să transmită mesajul cu QoS cel mai scăzut.

**2.3.1 At most once (QoS 0)**

QoS 0 este cunoscut și sub numele de  “fire and forget‘ mod de trimitere a mesajelor. O dată ce  un expeditor a trimis un  mesaj nu este interesat dacă mesajul lui a ajuns la destinatar sau nu. În plus, nu stabilește vreun mecanism de retrimitere a mesajului. Mesajele la acest nivel au cea mai mare eficiență de transmisie, dar există posibilitatea ca acestea să nu fie  livrate o dată din acest motiv nu există nicio garanție de livrare.

Diagram

Description automatically generated

Acest nivel este folosit atunci când conexiunea în mare parte este completă și stabilă, de exemplu: conectarea unui client de testare sau a unei aplicații front-end direct la un broker MQTT. Pierderea unor mesaje cu  date  nu sunt  atât de  importante pentru noi sau sunt transmise la un interval scurt imediat.

**2.3.2 At least once (QoS 1)**

Când QoS este 1, se poate garanta că mesajul va fi  publicat cel puțin o dată. MQTT garantează QoS 1 printr-un mecanism simplu de ACF.  Acknowledgment (ACF ) este un semnal care este transmis între procesele de comunicare, computere   sau dispozitive pentru a transmite confirmarea sau primirea unui mesaj, ca parte dintr-un protocol de comunicații.

Expeditorul  trebuie mai întâi să stocheze mesajul apoi  să-l trimită receptorului. El trebuie  să aștepte să primească un pachet de PUBACK de la receptor. Dacă după un interval de timp,  expeditorul nu primește confirmarea de la receptor acesta va trimite din nou mesajul pe care îl are stocat până când va primi confirmarea. Expeditorul folosește identificatorul de pachet din fiecare pachet pentru a face corespondența între pachetul PUBLISH și pachetul PUBACK adecvat.

Diagram

Description automatically generated

Când un receptor primește un mesaj cu QoS 1, îl poate procesa imediat. De exemplu, dacă receptorul este un broker, brokerul trimite mesajul tuturor clienților abonați și apoi răspunde cu un pachet PUBACK.

Acest nivel este folosit deseori deoarece garantează  că mesajul ajunge cel puțin o dată sau  când aplicația mea nu suportă costul general al unui QoS 2 și are nevoie să livreze mesajele repede.

**2.3.2 Exactly once (QoS 2)**

QoS 2 este cel mai înalt nivel de serviciu din MQTT, cel mai sigur, dar cel mai lent nivel de calitate  a serviciului dintre toate. Această garanție este oferită de cel puțin două fluxuri de cerere/ răspuns între expeditor și destinatar.

Expeditorul  trebuie mai întâi să stocheze mesajul apoi  să-l trimită receptorului. Receptorul primește pachetul de la expeditor, stochează și procesează mesajul și îi răspunde prin trimiterea unui pachet PUBREC.  În cazul în care expeditorul nu primește un pachet PUBREC de la receptor va trimite din nou pachetul până când va primi confirmarea.   După ce primește un pachet PUBREC expeditorul poate elimina mesajul în siguranță, după aceea trebuie să  stocheze starea pachetului PUBREC  și să-i răspundă receptorului printr-un pachet de tip PUBREL. După ce receptorul primește pachetul PUBREL, poate renunța la toate stările stocate și să  răspundă cu un pachet PUBCOMP (același lucru este valabil atunci când expeditorul primește PUBCOMP). Până când receptorul finalizează procesarea și trimite pachetul PUBCOMP înapoi expeditorului, receptorul stochează o referință la identificatorul de pachet al pachetului PUBLISH original. Acest pas este important pentru a evita procesarea mesajului a doua oară. Dacă un pachet se pierde pe parcurs, expeditorul este responsabil să retransmită mesajul

Când fluxul QoS 2 este complet, ambele părți sunt sigure că mesajul este livrat și expeditorul are confirmarea livrării.

Diagram

Description automatically generated

Acest nivel este folosit când este important pentru aplicația mea  să primească  toate mesajele exact o dată. Acest caz este adesea folosit  când  o livrare cu date duplicate poate dăuna utilizatorilor aplicației sau clienților abonați.

2.4 Mecanismul LAST WILL TESTAMENT

MQTT folosește mecanismul Last Will & Testament (LWT) pentru a notifica deconectarea incorectă a unei client (de exemplu: defecțiune la  bateria dispozitivului pe care îl folosește clientul, clientul nu reușește să comunice în perioada KeepAlive, clientul nu trimite un pachet DISCONNECT înainte de a închide conexiunea) către alți clienți. În acest  mecanism,  când un client se conectează la un broker îi poate specifica acestuia un  LWT în mesajul CONNECT, cel care inițiază conexiunea dintre client și broker. LWT  este un mesaj normal MQTT cu QoS, topic, retained flag & payload pe care brokerul îl stochează până când detectează că clientul s-a deconectat incorect. Dacă clientul se deconectează corect cu un mesaj DISCONNECT brokerul renunță la mesajul LWT stocat.

Acest mecanism este foarte util în cazul unei aplicații care trebuie să urmărească un fel de prezență a dispozitivelor participante.

**Capitolul 4 : Bibliografie**

[**https://www.emqx.com/en/blog/introduction-to-mqtt-qos**](https://www.emqx.com/en/blog/introduction-to-mqtt-qos)

[**https://en.wikipedia.org/wiki/Acknowledgement\_(data\_networks)**](https://en.wikipedia.org/wiki/Acknowledgement_(data_networks))

[**https://medium.com/jungletronics/mqtt-qos-ef1ef4498405**](https://medium.com/jungletronics/mqtt-qos-ef1ef4498405)

[**https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/**](https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/)

[**https://en.wikipedia.org/wiki/MQTT**](https://en.wikipedia.org/wiki/MQTT)

[**MQTT Interface (devicewise.com)**](https://docs.devicewise.com/Content/Products/IoT_Portal_API_Reference_Guide/MQTT_Interface/MQTT-Interface.htm)