23. Komunikační protokol MQTT

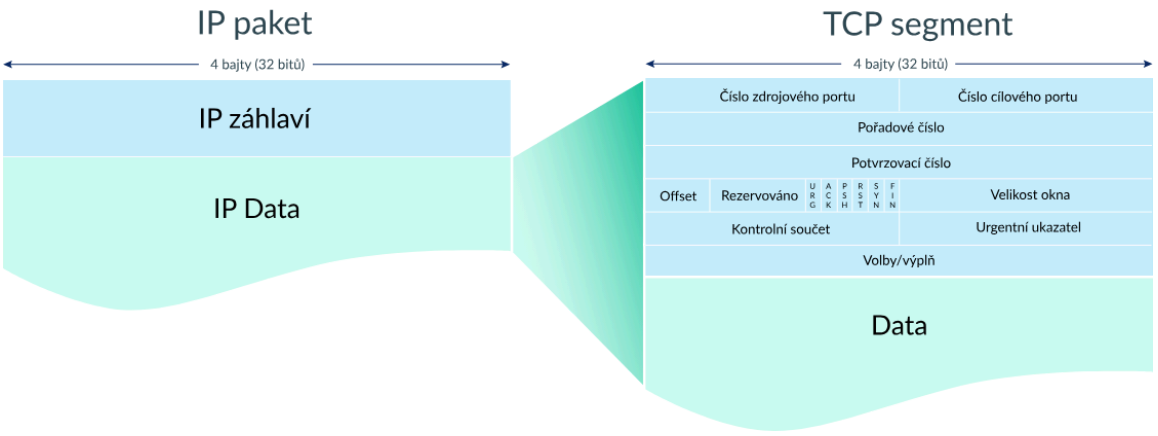
Created	@April 26, 2025 7:11 PM
Tags	Done
Kdo vypracoval	Robb

Všechno ohledně sekce "Obecné znalosti" je podle švihly zbytečné ale já mu nevěřím tak to tu nechám. Podle švihly si stačí přečíst všechno od nadpisu "MQTT"

Obecné znalosti

Komunikační protokol, který umožnuje předávání správ klientů pomocí centrálního bodu (**broker**).

- Přenos probíhá pomocí **TCP/IP** (Transmission Control Protocol / Internet Protocol)
  - Zajištuje spolehlivé spojení mezi zařízeními
  - Formát packetu:



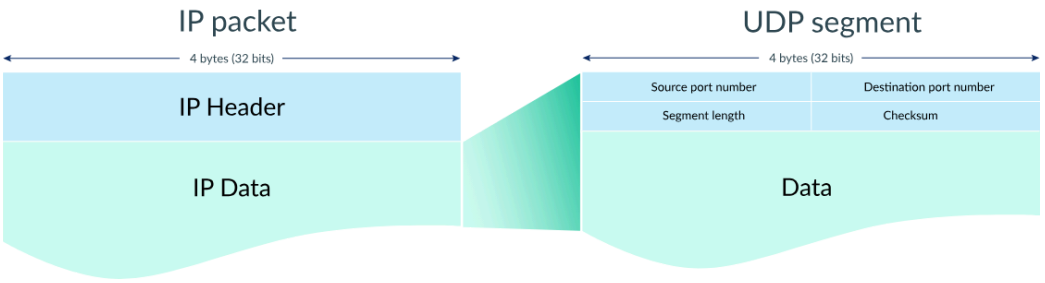
TCP/IP

TCP/IP protokol sdílí některá pole s **UDP** protokolem

1. Číslo zdrojového portu = Source port number

2. Číslo cílového portu = Destination port number
3. Kontrolní součet = Checksum

- **UDP** (User Datagram Protocol)
  - Funguje nad IP
  - Přináší mechanismus pro detekci korupce dat v packetech ale korupci jen oznamuje, nijak ji neřeší
  - Jednoduchý a rychlý, často aplikovaný v případech kdy je důležitější rychlost než přesnost
  - Formát packetů



  - **Checksum**
    - Poslední dva bajty UDP headeru
    - Prověření že nejsou data corruptnuté

    1. Odesílatel před posláním packetu rozdělí správu na dvě půlky, sečte je, uloží do packetu a odešle
    2. Po přijmutí packetu udělá příjemce to samé a porovná vypočítaný **Checksum** s tím, který je uložený v packetu

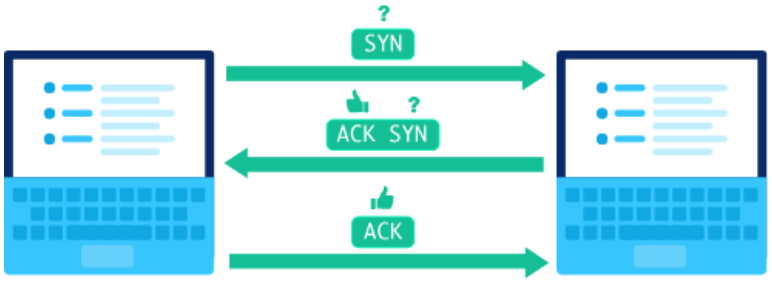
       Sedí → Všechno ok

       Nesedí → Něco je v piči

- **Třístupňové ověření (Navázání spojení)**

  1. První počítač odesílá packet s bitem SYN (SYN = “Synchronizovat?”) nastaveným na 1
  2. Druhý počítač odesílá zpět packet s bitem ACK nastaveným na 1 (ACK = “Potvrzená”) a s bitem SYN také nastaveným na 1
  3. První počítač odpovídá bitem ACK

4. Jakmile jsou počítače s ověřením hotovy jsou připraveny přijímat data



Třístupňové ověření

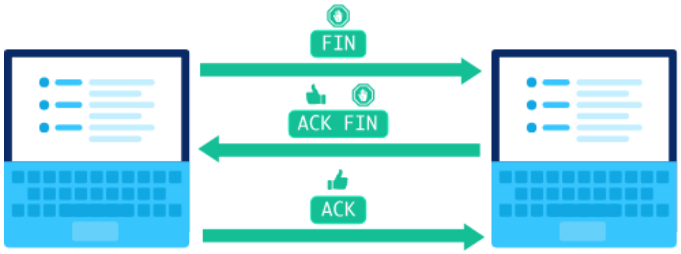
- **Odesílání packetů dat**

  Když je odeslán packet dat musí vždy příjemce potvrdit co obdržel

  První počítač posílá packet s daty a pořadovým číslem. Druhý potvrdí použitím ACK bitu a a zvýšením potvrzovacího čísla o délku obdržených dat

- **Ukončení spojení**

  1. První počítač odešle packet s bitem FIN nastaveným na 1.
  2. Druhý odpoví packetem ACK a FIN.
  3. Na to první počítač odpoví ACK.



- **Detekce ztracených packetů**

  Detekuje ztracené pakety pomocí časovým limitem

  1. Po odeslání spustí odesílatel časovač a umístí packet do fronty na přeposílání
  2. Když časovač vyprší a příjemce packet neobdržel pošle se znovu

  Může vést k tomu že příjemce obdrží duplicitní packety

- **Zpracovávání dat v nesprávném pořadí**

1. Pokud příjemce vidí vyšší pořadové číslo → něco chybí
2. Příjemce dává odesílateli vědět že něco chybí a odesílá packet s potvrzovacím číslem nastaveným jako pořadové číslo

Někdy packet cestuje delší cestou a dorazí s opožděním. Muže se stát že

Někdy se po cestě ztratí a musí se odeslat znovu

MQTT

- Návrhový vzor **publisher/subscriber**
- Broker (centrální bod) třídí zprávy podle témata (topic) a zařízení buď:
  - **Publikuje** v daném tématu (publisher) a odesílá zprávu brokeru, který ukládá a přeposílá zprávu zařízením, které mají odběr (subscribtion) na dané téma
  - Je **přihlášeno k odběru** na dané téma a broker zasílá tomuto zařízení všechny zprávy daného téma
- V protokolu se posílají zprávy (**Message** nebo také **Payload**) a s ní téma (**Topic**)
- Klient může publikovat i v topicu, u kterého má subscribe. Zpráva se samozřejmě odešle všem subscriberům, takže i klientovi, který publikoval. Irl se to ale takto nedělá. Nejlépe se klient změní ze subscribera na publishera, zprávu pošle a poté se vrátí na stav subscribera.
- Témata
  - místo kam “putuje” zpráva
  - Jedno zařízení může mít **odběr nebo publisher** u **více témat najednou**. Zpráva může patřit **právě do jednoho tématu**
  - **Publisher nemusí zakládat nové téma**. Pokud broker přijme zprávu s novým tématem automaticky jej založí
  - Témata jsou řetězce UTF-8 (diakritika není problém)

- **Wildcards:**
  - **Single level** = "+"
    - odběr celé jedné úrovně témat
    - pokud odbíráme téma: *myhome/groundfloor/+/temperature*

  - **Multi level** = "#"
    - odběr všech úrovní témat
    - pokud odbíráme téma: *myhome/groundfloor/#*

- Normálně broker **nevidí strukturu** topicu (topic je prostě string, Broker nevidí závorky jako oddělovače úrovně)
- Když broker **detekuje wildcard**, rozdělí si zprávu podle lomítek a vytvoří si routovací tabulku, podle které zasílá nové zprávy subscriberům
- Protokol je "payload agnostic" = **formát** dat / zpráv je **irelevantní** (nejčastěji JSON nebo BSON). Obsah zprávy je omezen na **256 MB**.

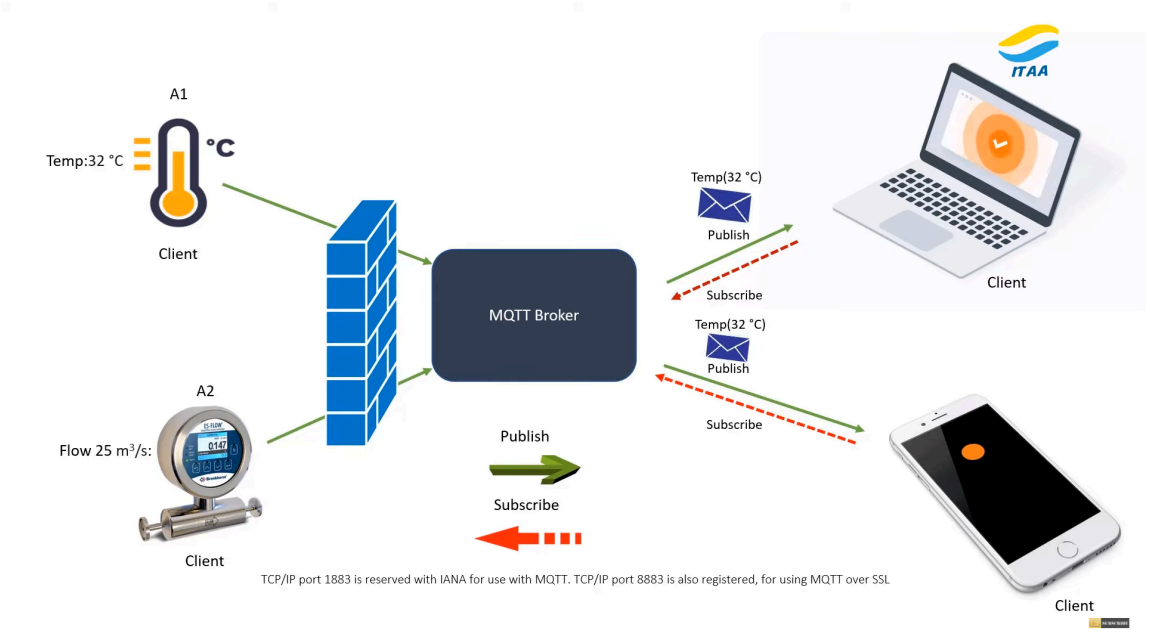
QoS

Tři úrovně QoS (Quality of Service). Klient nemusí všechny podporovat

= Potvrzení o dodání zprávy / packetu

1. Zpráva je odeslána bez potvrzení a **není zaručeno doručení**
2. Zpráva je doručena **alespoň jednou**
3. Každá zpráva je doručena **právě jednou**

Struktura MQTT



Obrázek z videa viz. Odkazy v sekci MQTT

Ukázka programu MQTT v Micropythonu:

```
import network
import time
from machine import Pin
from umqtt.simple import MQTTClient

wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect("ssid","pass")
time.sleep(5)
print(wlan.isconnected())

sensor = Pin(16, Pin.IN)

mqtt_server = 'broker.hivemq.com'
client_id = 'sauron'
topic_pub = b'middle_earth/sauron'
topic_sub = b'middle_earth/sauron'
topic_msg = b'Ach nach utunbagul'
```

```
def callback(topic, msg):
    print("Message received: ", msg)
    print("Topic: ", topic)
```

```
def mqtt_connect():
    client = MQTTClient(client_id, mqtt_server, keepalive=3600) #keepalive o
    client.connect()
    print('Connected to %s MQTT Broker'%(mqtt_server))
    return client
```

Parametr “**keepalive**” označuje interval v sekundách, během kterého musí odesílatel odeslat packet PINGREQ aby nebylo připojení ukončeno. Broker po přijetí odpovídá PINGRESP packet, který potvrzuje že je připojení pořád aktivní

```
def reconnect():
    print('Failed to connect to the MQTT Broker. Reconnecting...')
    time.sleep(5)
    machine.reset()

try:
    client = mqtt_connect()
    client.set_callback(callback)
      client.subscribe(topic_sub) #zaregistruje schránku, kterou poslouchá a
except OSError as e:
    reconnect()
while True:
    client.publish(topic_pub, topic_msg) #topic_pub = jméno schránky
                                         #topic_msg = odesílané data
    time.sleep(3)
```

ASdamdwandwjdajůdawdbawjdbawjůdjjadůadakjdjakdj

Odkazy

- TCP

Khan Academy

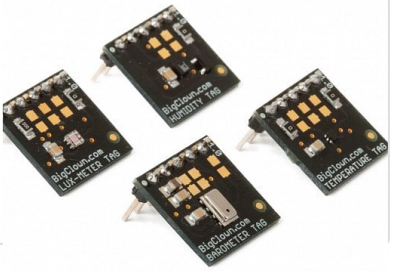
https://cs.khanacademy.org/computing/informatika-pocitace-a-internet/x8887af37e7f1189a:internet/x8887af37e7f1189a:tcp-protokol/a/transmission-control-protocol--tcp

- MQTT

Protokol MQTT: komunikační standard pro IoT - Root.cz

Dnes odbočíme od představení BigClown k obecnějšímu tématu. Představíme si komunikační protokol, který se stal velmi rychle „lingua franca“ pro internet...

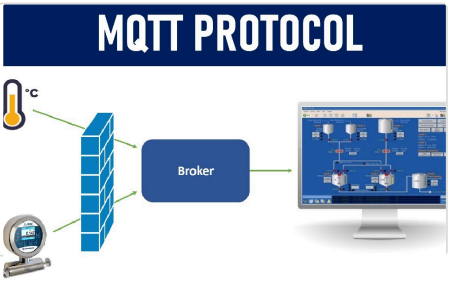
https://www.root.cz/clanky/protokol-mqtt-komunikacni-standard-pro-iot/



What is MQTT Protocol ? How it works ? | 2022

What is MQTT Protocol? How it works | 2022 #mqtt #2022 #protocol #industrial #iiot
IIOT protocol MQTT

https://youtu.be/NXyf7tVsi10?si=QTkVaIjmBGpxHIL1&t=117



- UDP

Khan Academy

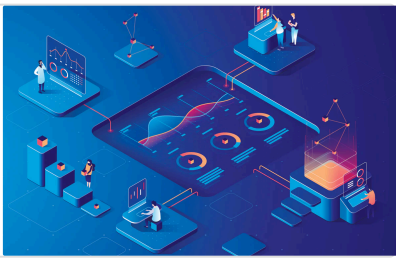
https://en.khanacademy.org/computing/computers-and-internet/xcae6f4a7ff015e7d:the-internet/xcae6f4a7ff015e7d:transporting-packets/a/user-datagram-protocol-udp

Nepoužité ale mohly by se hodit:

Co je MQTT a k čemu slouží ve IIoT? Popis protokolu MQTT

MQTT je kompaktní a otevřený protokol výměny dat určený pro přenos dat na vzdálená místa, kde je vyžadována malá velikost kódu a existují omezení šířky pásma.

https://ipc2u.cz/blogs/news/mqtt-protokol



MQTT - The Standard for IoT Messaging

A lightweight messaging protocol for small sensors and mobile devices, optimized for high-latency or unreliable networks, enabling a Connected World and the Internet of Things

https://mqtt.org/

přihlášení a zabezpečení

qos

Topologie