Objektovo orientované programovanie - *pokročilí*

Učiteľ:
Ing. Jozef Wagner, PhD.

Učebnica:
https://oop.wagjo.com/

OPGP
Pokročilí 20

1. Transmission Control Protocol
2. Fázy spojenia
3. Nadviazanie spojenia
4. Three-way handshake

Transmission Control Protocol

TCP - protokol na 4. vrstve OSI

Stream-oriented - používa súvislý prúd bajtov - **stream**.

Connection oriented - client a server musia najprv nadviazať spojenie

Používame štandardný Python modul `socket`

Transmission Control Protocol

Vlastnosti TCP

- *Spoľahlivosť* - stratené segmenty sa automaticky znovu pošlú
- *Poradie* - dáta prídu presne v tom poradí, v akom boli poslané
- *Obojsmerný stream* - obe strany môžu posielať a prijímať súčasne

Transmission Control Protocol

Nevýhody TCP oproti UDP

- vyššie oneskorenie a overhead
- nevhodné pre realtime komunikáciu a online hry
- vyššie nároky na server, pomalší štart
- nepodporuje broadcast/multicast

TCP – fázy spojenia

Fázy spojenia

1. **Nadviazanie spojenia** - používame funkcie `listen()`, `connect()` a `accept()`
2. **Prenos dát** - používame `sendall()` a `recv()`
3. **Ukončenie spojenia** - používame `shutdown()` a `close()`

Nadviazanie spojenia

Server počúva (listen) na svojom porte a zachytí (accept) žiadosti o spojenie.

Client žiada server o nadviazanie spojenia (connect).

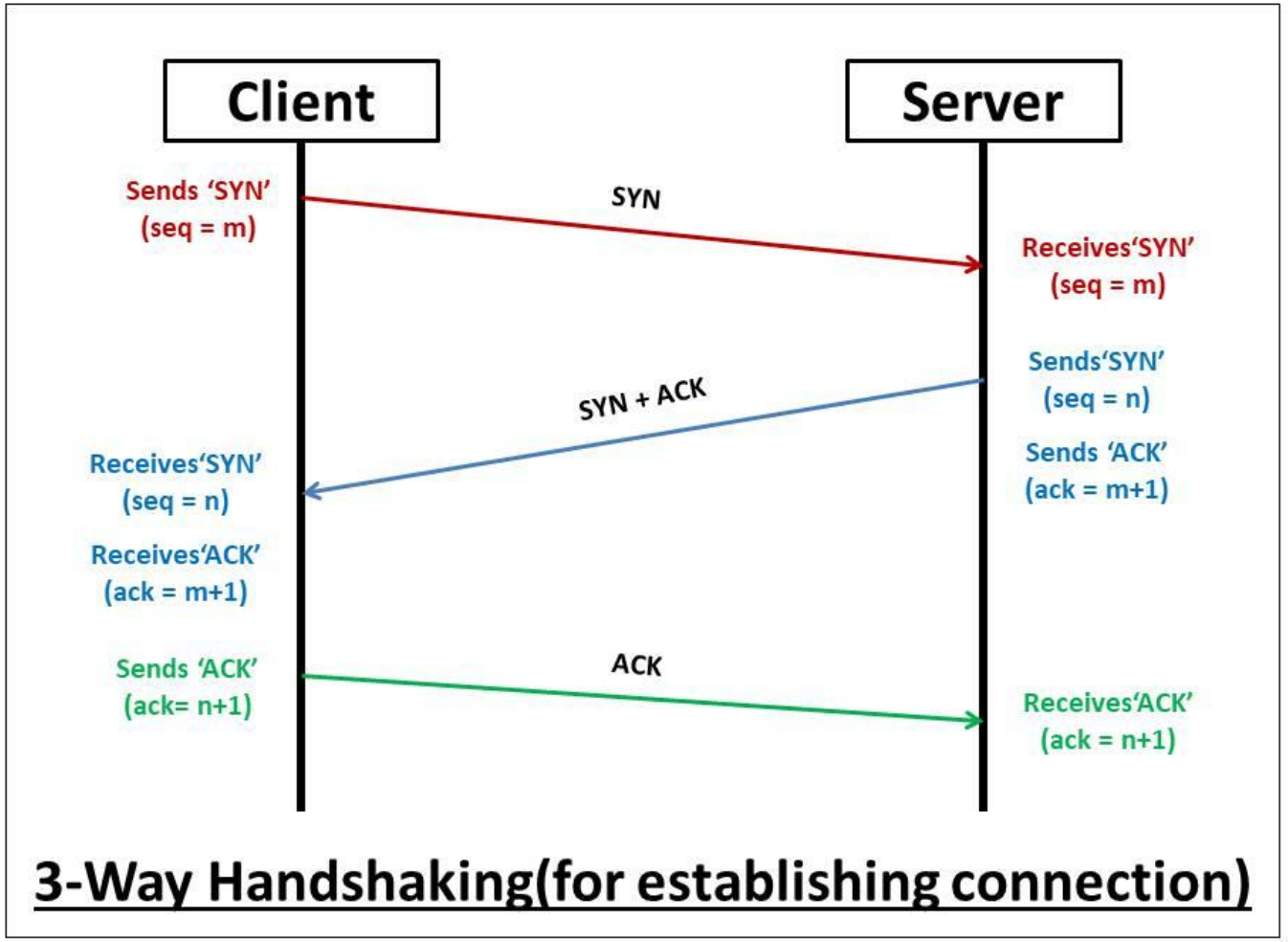
Mechanizmus nadviazania spojenia sa volá **Three-way handshake**

Nadviazanie spojenia

Three-way handshake:

1. Klient pošle **SYN** segment
2. Server odpovie segmentom **SYN+ACK**
3. Klient nakoniec odošle **ACK** segment

SYN segmenty obsahujú aj rôzne nastavenia spojenia



TCP Client

```
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM);

# 1. Nadviazanie spojenia - CONNECT (THREE-WAY HANDSHAKE)
sock.connect(('127.0.0.1', 65432))
# 2. Pošleme správu serveru
message = "Ahoj server! Toto je test spojenia."
sock.sendall(message.encode('utf-8'))
# 3. Prijmeme odpoveď
response = sock.recv(1024)
if response:
    print(f"Odpoveď od servera: {response.decode('utf-8')}")
# 4. Shutdown - oznámime serveru, že už nebudeme posielať
sock.shutdown(socket.SHUT_WR)
# 6. Zatvorenie spojenia
sock.close()
```

TCP Server

```
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
# 1. Viažeme socket na IP + port
sock.bind((HOST, PORT))
# 2. Začneme počúvať (5 je max čakajúcich klientov na pripojenie)
sock.listen(5)
# 3. Prijatie spojenia (THREE WAY HANDSHAKE)
conn, addr = sock.accept()
# Máme aktívne spojenie - 4. Server čaká na správu od klienta
data = conn.recv(1024)
if data:
    print(f"Prijaté od klienta: {data.decode('utf-8')}")
    conn.sendall(data) # 5. Poslanie správy naspať klientovi
# 6. Zatvorenie serveru
conn.close()
sock.close()
```