26.10.2022

Cursul 4

TCP < protocol orientat pe conexiune
      < reliable

Socket-uri UDP

Emițător (client)

Receptor (server) (roluri mai puțin egale; interschimbabile

C = socket (AF_INET, SOCK_DGRAM

S = socket (AF_INET, SOCK_DGRAM)

* UDP - nu se poate face deosebire între cazurile când { - se pierd date de la E la R
                                                       { - nu există un proces cu care să se comunice
                                                       - se pierde rezultatul / răspunsul

* 64 Kb - dimensiunea maximă a unui packet
  (MTU - Maximum Transfer Unit)

Situații UDP (neverificate, ca în cazul TCP) < pierderi de date
                                             < ajung datele, neordonate (out of order)

↓
este mai importantă rapiditatea
(modelul interogare - răspuns)

~~connect~~

sendto (c, de unde dău mesajul, cât, flag, structura, sizeof(structura))
- IP dest
- port dest

bind (S, ~~accept~~, ~~listen~~)

recvfrom (S, unde, cât, flag, structură, sizeof(str))
- IP emițător
- port emițător

recvfrom (

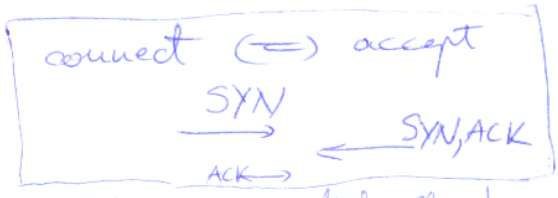
close (c)

Sendto (

* sendto trimite datele unui partener ale cărui date sunt de identificare completate de recvfrom

(Dacă fac serverul concurent, după recvfrom generez un port random (pt. identificare), fac un fiu, iar „comunicarea" se face pe baza portului nou generat.)

TCP:



(UDP/IP), TCP packet structure

- Port { Sursă / Destinație
- Număr de secvență
- Număr de confirmare

⇝ pachete de control → fără date
(ex: se trimit pt. a confirma că încă are loc conexiunea și se primesc date din sens invers)

un pid de la close → FIN

→ confirmările nu se fac instant (ar supraaglomera fluxul de conexiune / rețeaua cu pachete goale / de control)

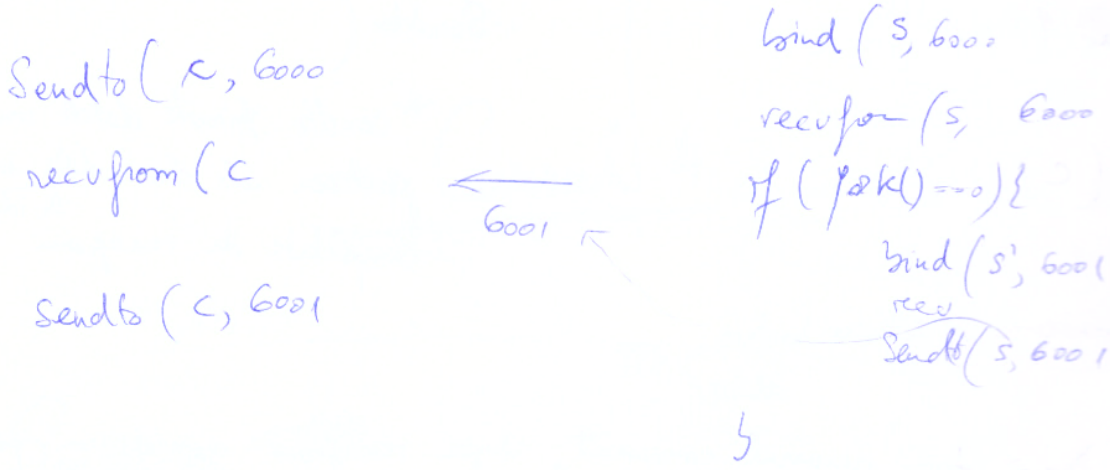
↓

se confirmă colecții de date

ex:

```
c = socket (AF_INET, SOCK_STREAM)
connect (c)
...
close (c)
c = socket (AF_INET, SOCK_DGRAM)
```

„reciclarea" lui c → e nu e în același timp socket TCP și UDP


Sendto(c, 6000
recvfrom(c
6001
Sendto(c, 6001
bind(s, 6000
recvfrom(s, 6000
if(fork() == 0){
bind(s', 6001
recv
Sendto(s, 6001
}