Telekommunikációs Hálózatok

5. gyakorlat

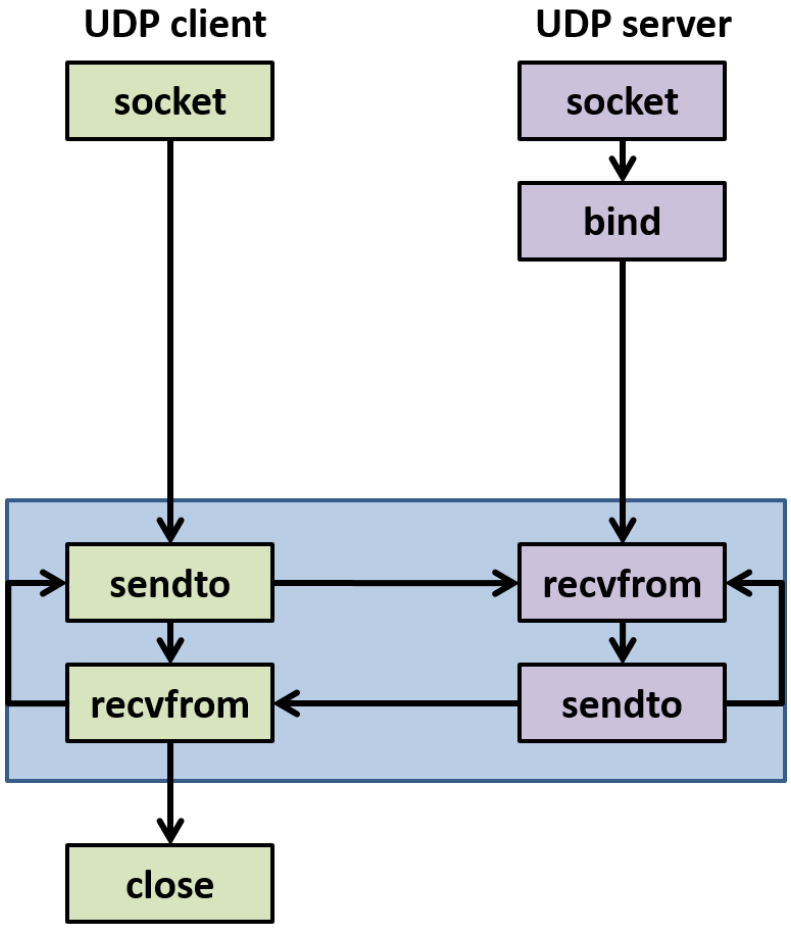
PYTHON SOCKET - UDP

Ismétlés

A kommunikációs csatorna kétféle típusa

- Kapcsolat-orientált modell (analógia: telefonbeszélgetés)
  - csomagok megérkeznek jó sorrendben
  - ilyen protokoll a TCP
  - kapcsolódó típus: stream socket
- **Kapcsolat-nélküli modell (analógia: postai levelezés)**
  - **csomagok nem biztos, hogy sorrend helyesen érkeznek, sőt el is veszhetnek**
  - **előnye a jobb teljesítmény**
  - **ilyen protokoll a UDP**
  - **kapcsolódó típus: datagram socket**

UDP

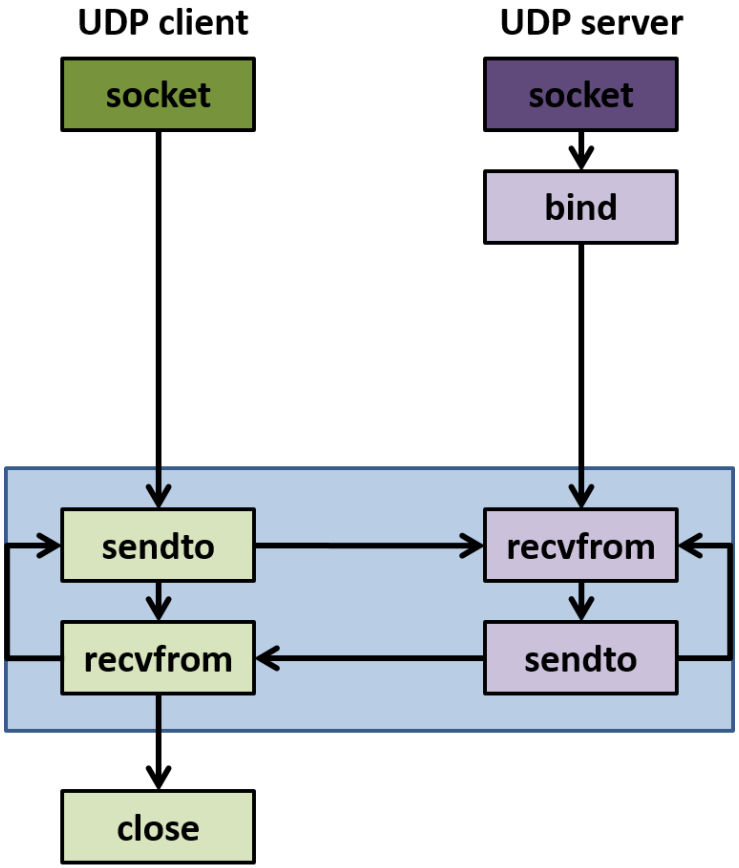

UDP client
UDP server
socket
socket
bind
sendto
recvfrom
recvfrom
sendto
close

Socket leíró beállítása

- socket.**socket**([*family*
  [, *type*
  [, *proto*]]])
- *family* : socket.AF_INET → IPv4 (AF_INET6 → IPv6)
- ***type* : socket.SOCK_DGRAM → UDP**
- *proto* : 0

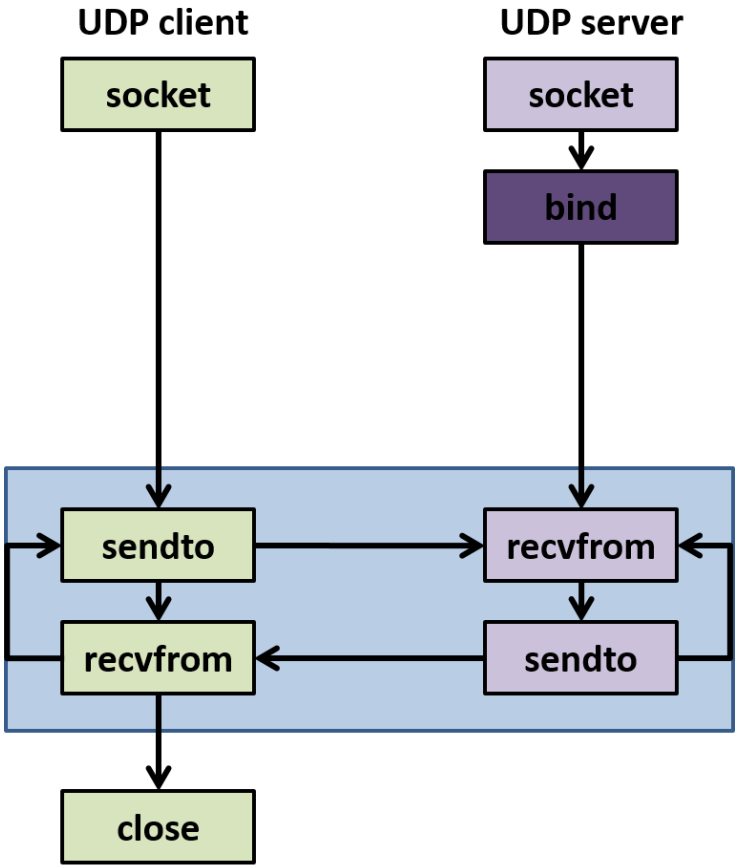
(alapértelmezett protokoll lesz)

- visszatérési érték: egy socket objektum, amelynek a metódusai a különböző socket rendszer hívásokat implementálják



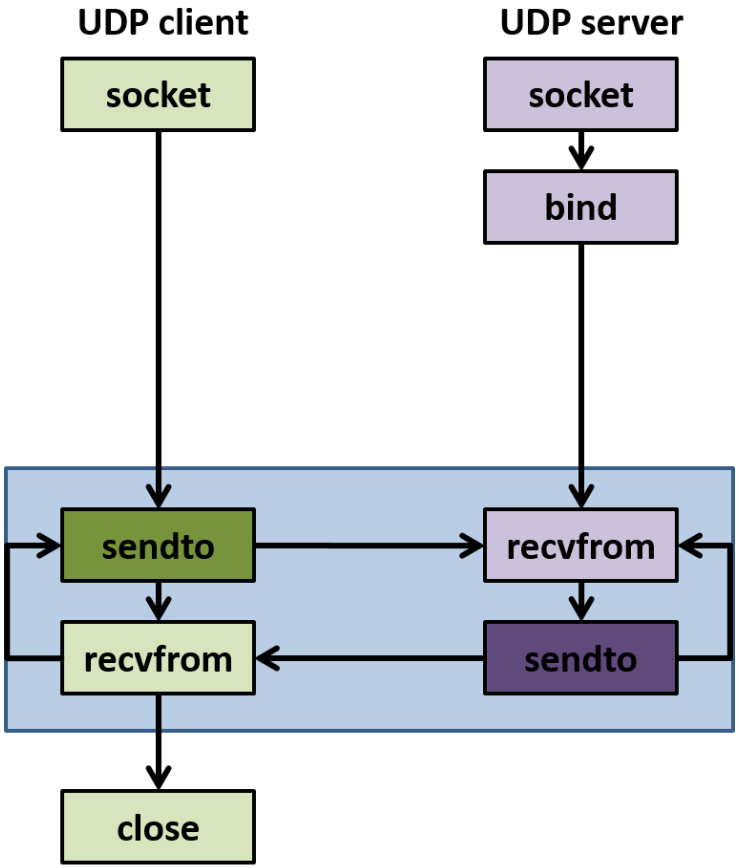
Bindolás – ismétlés

- socket.socket.**bind**(*address*)
- A socket objektum metódusa
- *address* : egy tuple, amelynek az első eleme egy hosztnév vagy IP cím (sztring reprezentációval), második eleme a portszám



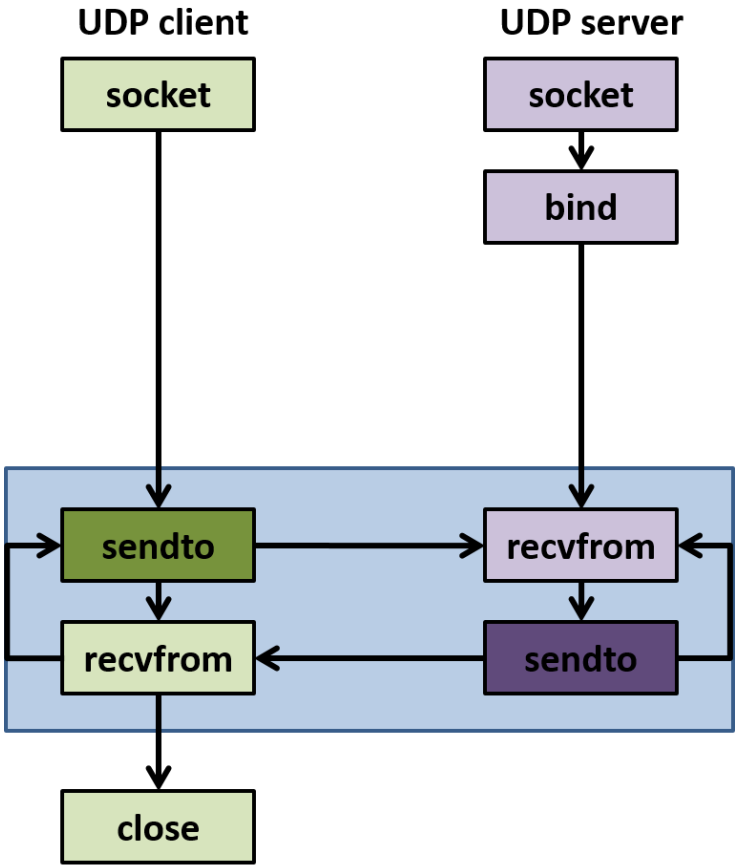
sendto

- socket.socket.**sendto**(*bytes* , ***address***)
- socket.socket.**sendto**(*bytes* , *flags* , ***address***)
- A socket objektum metódusai
- Adatküldés (*bytes*) a socketnek
- *flags* : 0 (nincs flag meghatározva)
- **A socketnek előtte nem kell csatlakozni a távoli sockethez, mivel azt az *address* meghatározza**



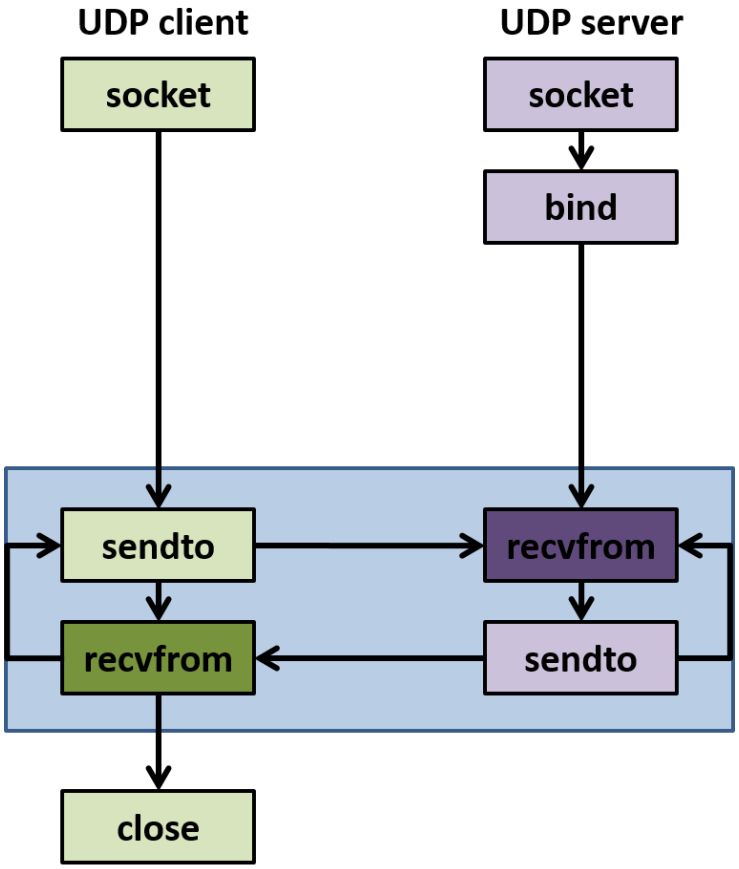
sendto

- **Fontos, hogy egy UDP üzenetnek bele kell férni egy egyszerű csomagba (ez IPv4 esetén kb. 65 KB-ot jelent)**
- visszatérési érték: az átküldött bájtok száma
  - az alkalmazásnak kell ellenőrizni, hogy minden adat átment-e
  - ha csak egy része ment át: újra kell küldeni a maradékot



recvfrom

- socket.socket.**recvfrom**(*bufsize [, flags]*)
- A socket objektum metódusa
- Üzenet fogadása
- *bufsize* : a max. adatmennyiség, amelyet egyszerre fogadni fog
- *flags* : 0 (nincs flag meghatározva)
- **visszatérési érték: egy (*bytes*, *address*) tuple, ahol a fogadott adat bytes reprezentációja és az adatküldő socket címe szerepel**



UDP

- socket

```
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
```

- recvfrom()

```
data, address = sock.recvfrom(4096)
```

- sendto()

```
sent = sock.sendto(data, address)
```

Feladat 1

Készítsünk egy kliens-szerver alkalmazást, amely UDP protokollt használ. A kliens küldje a ‚Hello Server' üzenetet a szervernek, amely válaszolja a ‚Hello Kliens' üzenetet.

Nézzük meg a megoldást!

Udp video streaming példa

A Fájlok / Gyakorlat05 könyvtárból töltsük le az alábbi fájlokat:

videograbber.py, video.mp4,
udp_stream_client_gyak5.py,
udp_stream_server_gyak5.py

cv2 telepítés:

```
pip install opencv-python
```

Benti laboros gépeken:

```
py –m pip install --user opencv-python
```

Forrás: https://docs.opencv.org

Udp video streaming példa

```
cv2.namedWindow(winname)
```

- **winname** – Name of the window in the window caption that may be used as a window identifier.
- The function **namedWindow** creates a window that can be used as a placeholder for images and trackbars. Created windows are referred to by their names.

Forrás: https://docs.opencv.org

Udp video streaming példa

```
cv2.imdecode(buf, flags)
```

- **buf** – Input array or vector of bytes.
- **flags** – Flags specifying the color type of a loaded image (1: Return a 3-channel color image.)
- The function reads an image from the specified buffer in the memory.

Forrás: https://docs.opencv.org

Udp video streaming példa

```
cv2.imshow(winname, mat)
```

- **winname** – Name of the window.
- **mat** – Image to be shown.
- The function **imshow** displays an image in the specified window.

Forrás: https://docs.opencv.org

Udp video streaming példa

```
cv2.imencode(ext, img[, params])
```

- **ext** – File extension that defines the output format.
- **img** – Image to be written.
- **params** – Format-specific parameters.
- The function compresses the image and stores it in the memory buffer that is resized to fit the result.

Forrás: https://docs.opencv.org

Udp video streaming példa

```
cv2.VideoCapture(filename)
```

- **filename** – name of the opened video file
- VideoCapture constructor.

```
cv2.VideoCapture.read()
```

- This is the most convenient method for reading video files or capturing data from decode and return the just grabbed frame.

Feladat 2 - Számológép UDP felett

Készítsünk egy szerver-kliens alkalmazást, ahol a kliens elküld 2 számot és egy operátort a szervernek, amely kiszámolja és visszaküldi az eredményt. A kliens üzenete legyen struktúra. Használjunk UDP protokollt!

Feladat 3 - Képküldés UDP felett

Egy kliens küldjön át egy képet UDP segítségével a szervernek:

- 200 bájtonként küldjünk
- Ha vége a fájlnak akkor küldjünk üres sztringet
- Minden kapott üzenetre OK legyen a válasz

Feladat 4: Chat UDP-vel

- Készítsünk egy chat alkalmazást, amelynél egy chat szerveren keresztül tudnak a kliensek beszélni egymással!
- A kliensek először csak elküldik a nevüket a szervernek
- A szerver szerepe, hogy a kliensektől jövő üzenetet minden más kliensnek továbbítja névvel együtt: [<név>] <üzenet> ; pl. [Józsi] Kék az ég!
- A kliensek a szervertől jövő üzeneteket kiírják a képernyőre.

VÉGE
KÖSZÖNÖM A FIGYELMET!