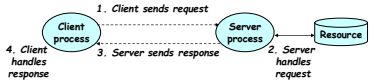
Lập trình mạng (Network Programming)

Chương 5. Socket

Mô hình Client/Server

- Client gửi yêu cầu (request) – Server cung cấp dịch vụ (response)
- Thông thường: Single server – multiple clients
- Server không cần biết về client
- Client phải biết một số thông tin về server (nơi đặt server)



Socket

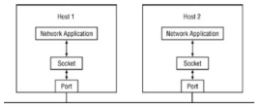
- Socket là một giao diện lập trình ứng dụng mạng
  - Qua giao diện này ta có thể lập trình điều khiển việc truyền thông giữa hai máy sử dụng các giao thức TCP, UDP...
- Một socket có thể mở/đóng một kết nối
  - Có thể gửi/nhận dữ liệu qua kết nối này
  - Dữ liệu thường gửi theo khối (packet)
- Dữ liệu truyền trên internet phải sử dụng giao thức internet
  - Packet phải có địa chỉ IP nguồn/đích và cổng (port)
  - Số hiệu cổng: 1 → 65535 (nên sử dụng cổng > 1024)

Socket

- Hai loại socket
  - Stream socket: dựa trên giao thức TCP (tranmission control protocol)
    - Chỉ truyền dữ liệu khi đã thiết lập kết nối
    - Đảm bảo truyền tin cậy, đúng thứ tự
    - Có cơ chế quản lý luồng và chống tắc nghẽn
  - Datagram socket: dựa trên giao thức UDP (user datagram protocol)
    - Truyền dữ liệu không cần thiết lập kết nối
    - Truyền không tin cậy
    - Tốc độ cao

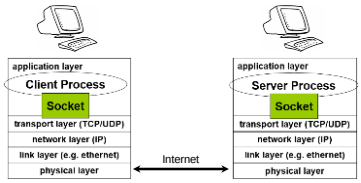
Giao diện socket

- Khi socket được tạo – nó phải gắn (bound) với một địa chỉ mạng và cổng
  - → Có thể truyền/nhận dữ liệu qua mạng



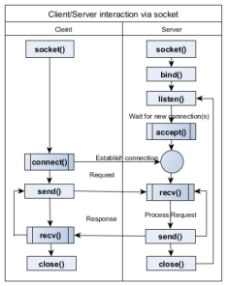
Giao diện socket

- Socket trong ứng dụng mạng



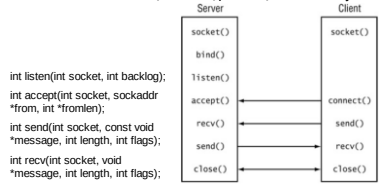
Giao diện socket

- Client/server giao tiếp qua socket
- Request/response loop
  - Client gửi dữ liệu tới server
  - Server xử lý và trả về cho client
  - Kết thúc: client đóng kết nối
  - Server trở về trạng thái listen



Socket hướng kết nối (connection-oriented)

- Sử dụng giao thức TCP để thiết lập kết nối
  - Khi kết nối được thiết lập: dữ liệu có thể truyền/nhận



TCP socket

- Import thư viện socket
- Gọi lớp socket

```
from socket import*
import socket
#tạo socket TCP
s = socket.socket(family=AF_INET, type=SOCK_STREAM, proto=0)
print('Socket created')
```

  - Socket family: domain của socket: AF_INET, AF_UNIX, AF_BLUETOOTH
  - Socket type: TCP – SOCK_STREAM, UDP – SOCK_DGRAM
  - Protocol: xác định biến thể của giao thức và kiểu, thường để 0

TCP socket

- Trường hợp lỗi: không được phép truy cập đến cổng/host
  - try … except socket.error as err
- Kết nối: socket.connect(host, int(port))
- Ví dụ: kết nối với server socket
  - s.connect(host, int(port)
  - Đóng kết nối: s.shutdown(2)

Client

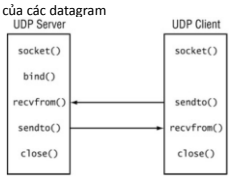
- Gửi dữ liệu đến server
- Dữ liệu: data = ‘abc’
- Gửi dữ liệu: s.send(data.encode(‘utf-8’))
- Nhận dữ liệu: data = s.recv(buffsize)
- In dữ liệu: print(data.decode(‘utf-8’)

Server

- Tạo socket:
  - s = socket.socket(family=AF_INET, type=SOCK_STREAM, proto=0)
- Server cần phải gắn (bind) tới socket và lắng nghe (listen)
  - s.bind(host, port)
  - s.listen(5)
  - client_socket, addr = s.accept()
- Đặt option:
  - s.setsockopt(socket.SOL_SOCKET,socket.SO_REUSEADDR, 1)

Socket phi kết nối (connectionless)

- Sử dụng giao thức UDP
  - Không có thông tin kết nối được gửi giữa hai thiết bị
  - Không kiểm tra lỗi của các datagram



Socket phi kết nối (connectionless)

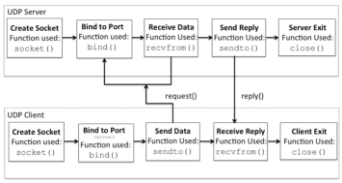
- Client gửi message đến server
  - UDP không đảm bảo chuyển đúng các gói tin (có thể mất) → không tin cậy
  - Connectionless → Không có dòng dữ liệu (streaming) giữa UDP server và UDP client
  - Tốc độ nhanh hơn TCP

- Lệnh

s = socket.socket(family=socket.AF_INET, type=socket.SOCK_DGRAM)

Byte = s.recvfrom(bufferSize)

sendto(bytesToSend, address)



Socket phi kết nối (connectionless)

- Khai báo
  - s = socket(AF_INET,SOCK_DGRAM)
- Server
  - s.bind((‘ ‘,12345))
  - Nhận dữ liệu: data, addr = sock.recvfrom(bufsize)
  - Gửi dữ liệu: s.sendto(msg.encode(),addr)
- Client
  - Khai báo socket
  - Gửi/nhận dữ liệu tương tự

Socket phi kết nối