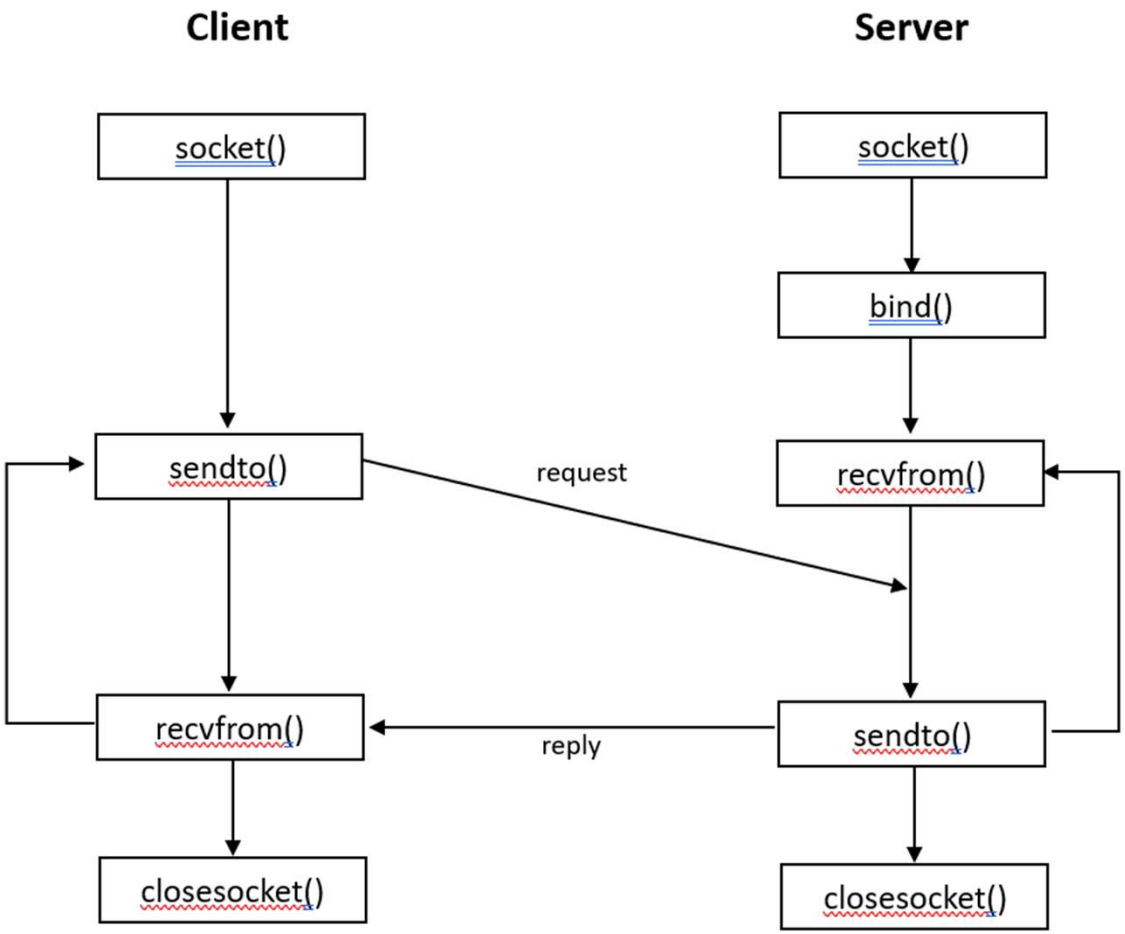
LẬP TRÌNH MẠNG

1 Lập trình WinSock

- Khởi tạo Socket
- Lập trình Socket cho ứng dụng TCP cơ bản
- Lập trình Socket cho ứng dụng UDP cơ bản
- Thiết kế giao thức ứng dụng

UDP SOCKET


Client
Server
socket()
socket()
bind()
sendto()
request
recvfrom()
recvfrom()
reply
sendto()
closesocket()
closesocket()

HÀM NHẬN DỮ LIỆU

```
int recv(
    SOCKET s,
    char FAR* buf,
    int len,
    int flags
);
```

Socket không kết nối

```
int recvfrom(
    SOCKET s,
    char FAR* buf,
    int len,
    int flags,
    struct sockaddr FAR* from,
    int FAR* fromlen
);
```

- **Tham số:**
  - "s" : **[IN]** socket đã được tạo
  - "buf" : **[OUT]** bộ đệm chứa dữ liệu nhận
  - "len" : **[IN]** kích thước của biến buf
  - "flag" : **[IN]** cờ thường là 0 (0, MSG_DONTROUTE, or MSG_OOB)
  - "from" : **[OUT]** địa chỉ gửi
  - "fromlen": **[OUT]** kích thước của cấu trúc địa chỉ

HÀM NHẬN DỮ LIỆU

```
int recv(
    SOCKET s,
    char FAR* buf,
    int len,
    int flags
);
```

Socket không kết nối

```
int recvfrom(
    SOCKET s,
    char FAR* buf,
    int len,
    int flags,
    struct sockaddr FAR* from,
    int FAR* fromlen
);
```

- **Trả về:**
  - <int>: Thành công và trả về bytes dữ liệu nhận
  - SOCKET_ERROR: Kết nối thất bại

HÀM GỬI DỮ LIỆU

```
int send(
    SOCKET s,
    const char FAR * buf,
    int len,
    int flags
);
```

Socket không kết nối

```
int sendto(
    SOCKET s,
    const char FAR * buf,
    int len,
    int flags,
    const struct sockaddr FAR * to,
    int tolen
);
```

- **Tham số:**
  - "s" : **[IN]** socket đã được tạo
  - "buf" : **[IN]** bộ đệm chứa dữ liệu gửi
  - "len" : **[IN]** kích thước của biến buf
  - "flag" : **[IN]** cờ thường là 0 (0, MSG_DONTROUTE, or MSG_OOB)
  - "to" : **[IN]** địa chỉ đích/nhận
  - "tolen" : **[IN]** kích thức của cấu trúc địa chỉ

HÀM GỬI DỮ LIỆU

```
int send(
    SOCKET s,
    const char FAR * buf,
    int len,
    int flags
);
```

Socket không kết nối

```
int sendto(
    SOCKET s,
    const char FAR * buf,
    int len,
    int flags,
    const struct sockaddr FAR * to,
    int tolen
);
```

- **Trả về:**
  - <int>: Thành công và trả về bytes dữ liệu đã gửi
  - SOCKET_ERROR: Kết nối thất bại

BT: TRUYỀN/NHẬN DỮ LIỆU DÙNG UDP

- **Viết chương trình ứng dụng phân giải tên miền sau:**
  - Server:
    - Viết chương trình serverUDP.cpp
    - Thực thi lệnh: serverUDP.exe <port_number>, trong đó <port_number> tuỳ chọn
    - Nhận tên miền của website từ client và tiến hành chuyển sang tên dạng địa chỉ IPv4 gồm 4 số phân tách bởi dấu '.'
    - Hiển thị và trả kết quả về cho client
  - Client:
    - Viết chương trình clientUDP.cpp
    - Thực thi lệnh: clientUDP.exe <server_IP> <port_number> <domain_name>, trong đó: <server_IP> và <port_number> tương ứng là địa chỉ IP và số hiệu cổng tùy chọn của server, <domain_name> là tên miền của site cần xác định địa chỉ IP
    - Chờ kết quả từ server và hiển thị

HÀM LẤY THÔNG TIN ĐỊA CHỈ

```
int getaddrinfo(
        const char FAR *nodename,
        const char FAR *servname,
        const struct addrinfo FAR *hints,
        struct addrinfo FAR *FAR *res
);
```

- **Yêu cầu: Thêm tiêu đề *<ws2tcpip.h>***
- **Tham số:**
  - "nodename" : **[IN] (**NULL-terminated) tên miền/địa chỉ IP máy chủ
  - "servname" : **[IN] (**NULL-terminated) số hiệu cổng hoặc tên dịch vụ (ftp, telnet)
  - "hints" : **[IN]** cấu trúc gợi ý nhằm phân giải tên
  - "res" : **[OUT]** danh sách liên kết kiểu ***addrinfo*** chứa thông tin về địa chỉ vừa phân giải được

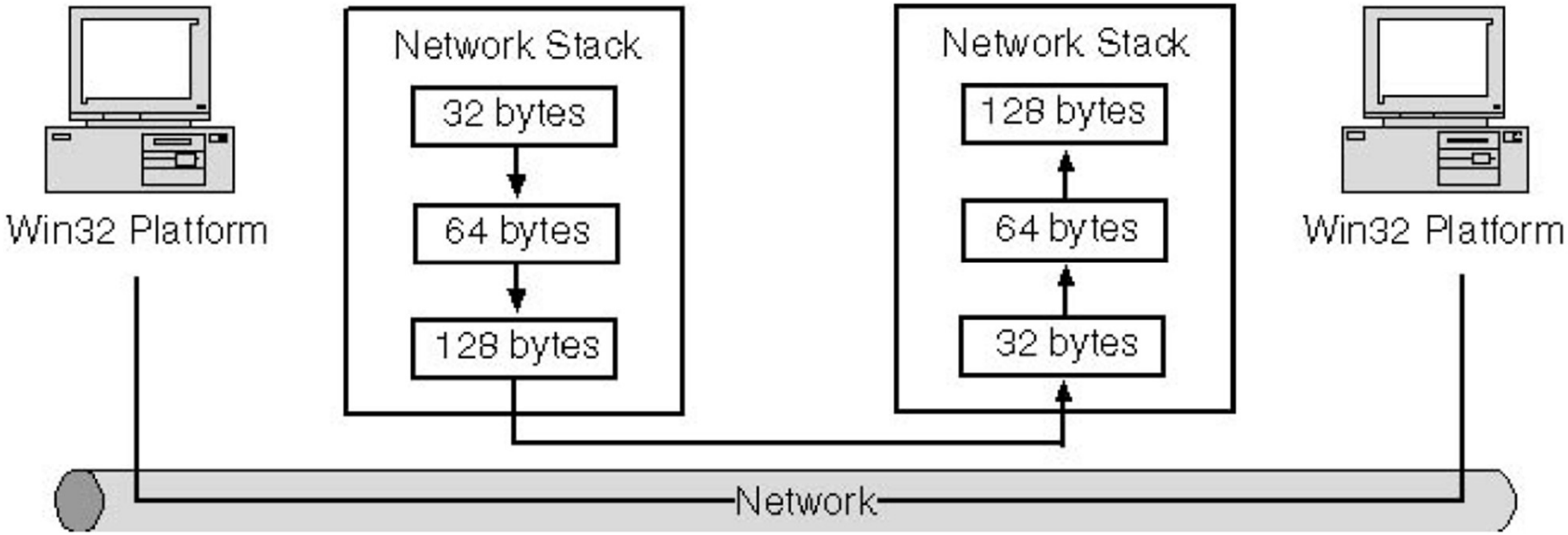
BT: TRUYỀN/NHẬN DỮ LIỆU DÙNG UDP

```
typedef struct addrinfo {
    int ai_flags; //cờ tùy chọn của hàm getaddrinfo()
    int ai_family; //họ giao thức
    int ai_socktype; //kiểu socket
    int ai_protocol; //giao thức lớp giao vận
    size_t ai_addrlen; //kích thước cấu trúc
    char *ai_canonname; //tên miền phụ
    struct sockaddr *ai_addr; //địa chỉ socket
    struct addrinfo *ai_next; //phần tử tiếp theo
} ADDRINFOA, *PADDRINFOA;
```

THIẾT KẾ GIAO THỨC ỨNG DỤNG

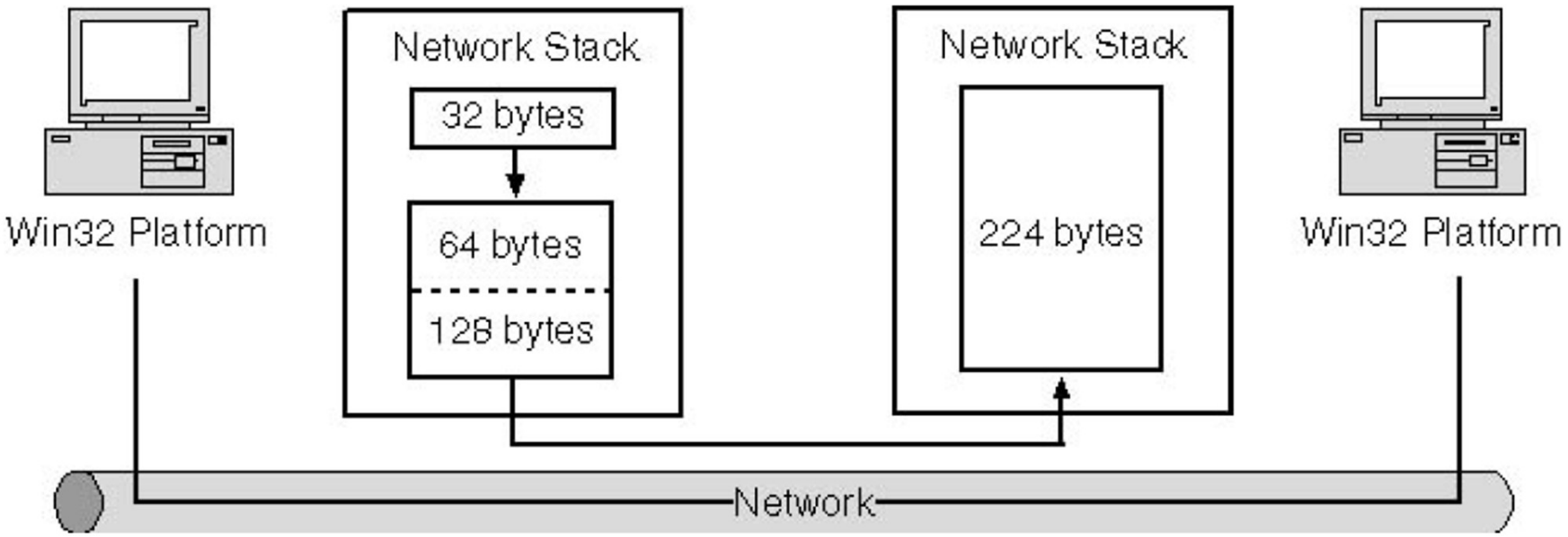
- **Đặc điểm giao thức ứng dụng:**
  - Hướng thông điệp (hướng tin), hướng luồng/hướng luồng giả
  - Hướng kết nối hay không kết nối
  - Độ tin cậy và trật tự dữ liệu
  - Broadcast/multicast và QoS

HƯỚNG THÔNG ĐIỆP



- **Đặc điểm:**
  - Máy trạm thực hiện ba lần gọi *recv* với bộ đệm 256 byte
  - Lần lượt từng cuộc gọitrả về 128, 64 và 32 byte

HƯỚNG LUỒNG



- **Đặc điểm:**
  - Sender gửi các gói 128, 64 và 32 byte
  - Ngăn xếp hệ thống cục bộ có thể tự do thu thập dữ liệu thành các gói lớn hơn

CÁC BƯỚC THIẾT KẾ

- Xác định các dịch vụ ứng dụng và mục tiêu của giao thức
- Lựa chọn mô hình truyền tin (client/server, P2P…)
- Định dạng thông điệp
- Cách thức truyền và xử lý thông điệp
- Tương tác với các giao thức khác

ĐỊNH DẠNG THÔNG ĐIỆP

- **Thông điệp thường có 2 phần: header và body**

- **Header: mô tả về thông điệp**
  - Loại thông điệp
  - Thao tác, lệnh
  - Kích thước phần thân(body)
  - Thứ tự của thông điệp
  - Số lần thử lại…

- **Body:**
  - Dữ liệu của ứng dụng

BT: THIẾT KẾ GIAO THỨC LOGIN/LOGOUT

- **Client: Gửi yêu cầu:**
  - Đăng nhập: Cần gửi thông tin id và password
  - Đăng xuất: Không cần gửi thông tin kèm theo

- **Server:**
  - Kiểm tra thông tin tài khoản và trạng thái. Gửi thông điệp trả lời tương ứng

BT: THIẾT KẾ GIAO THỨC LOGIN/LOGOUT

Client	Server
LOGIN <id> <password>	Trả về: ○ Thành công: Đăng nhập thành công !!! ○ Thất bại: • ID không tồn tại • Password không đúng • Sai định dạng thông điệp
LOGOUT	Trả về: ○ Thành công: Đăng xuất thành công !!! ○ Thất bại: • Chưa đăng nhập • Password không đúng • Sai định dạng thông điệp