


# Network Programming II

龚欣洋

# IP

32位；大端法；点分十进制表示 (ipv4)

标识设备；向中转节点指示数据包的传输路径

镜像	内网IP	浮动IP	云主机类型	状态
	10.129.123.45 2001:0db8:85a 3:0000:0000:8a 2e:0370:7334 192.168.56.78	-	I2	<span>●</span> 运行中

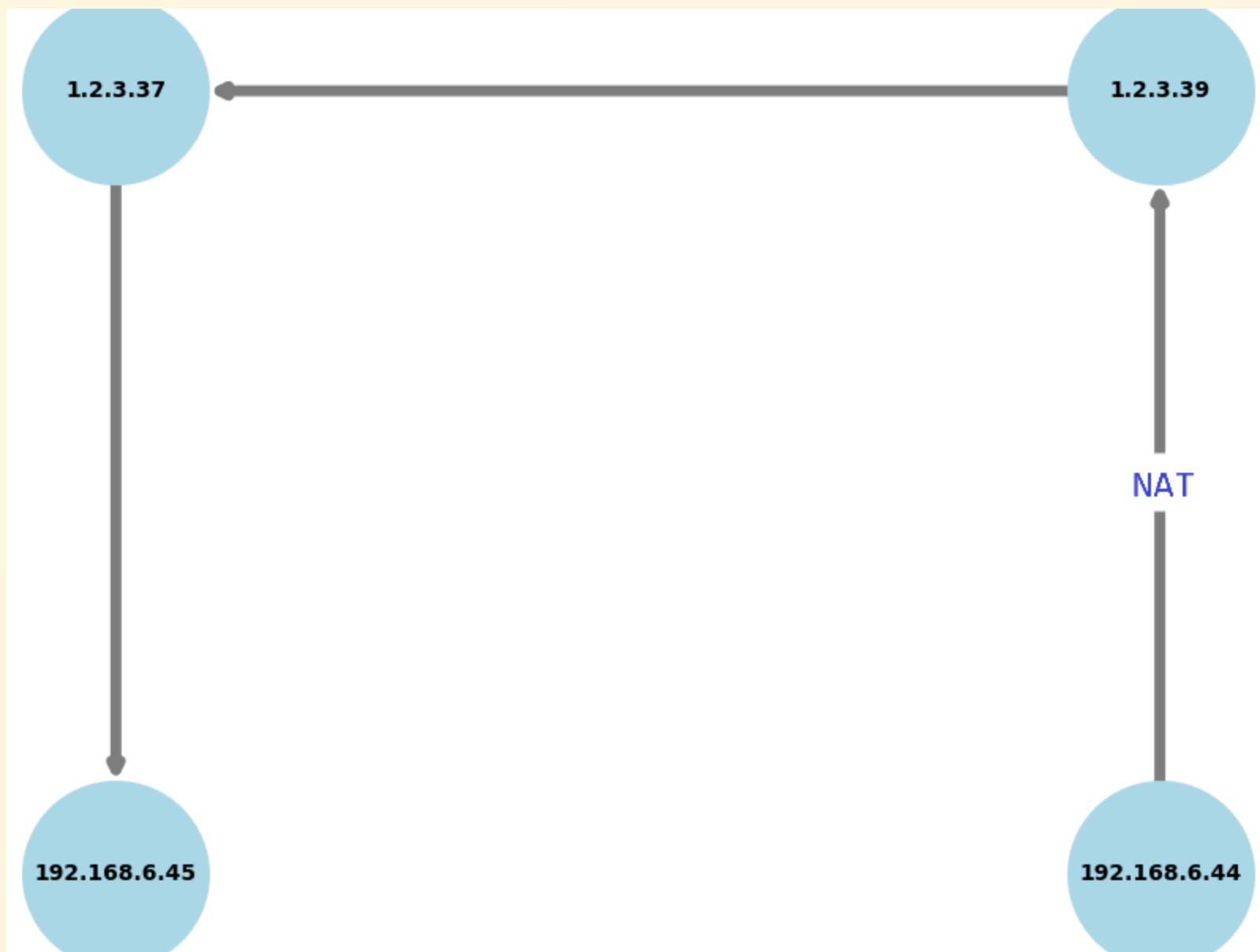
# 私有IP

用于内部网络

- 10.0.0.0 ~ 10.255.255.255
- 172.16.0.0 ~ 172.31.255.255
- 192.168.0.0 ~ 192.168.255.255

**localhost**

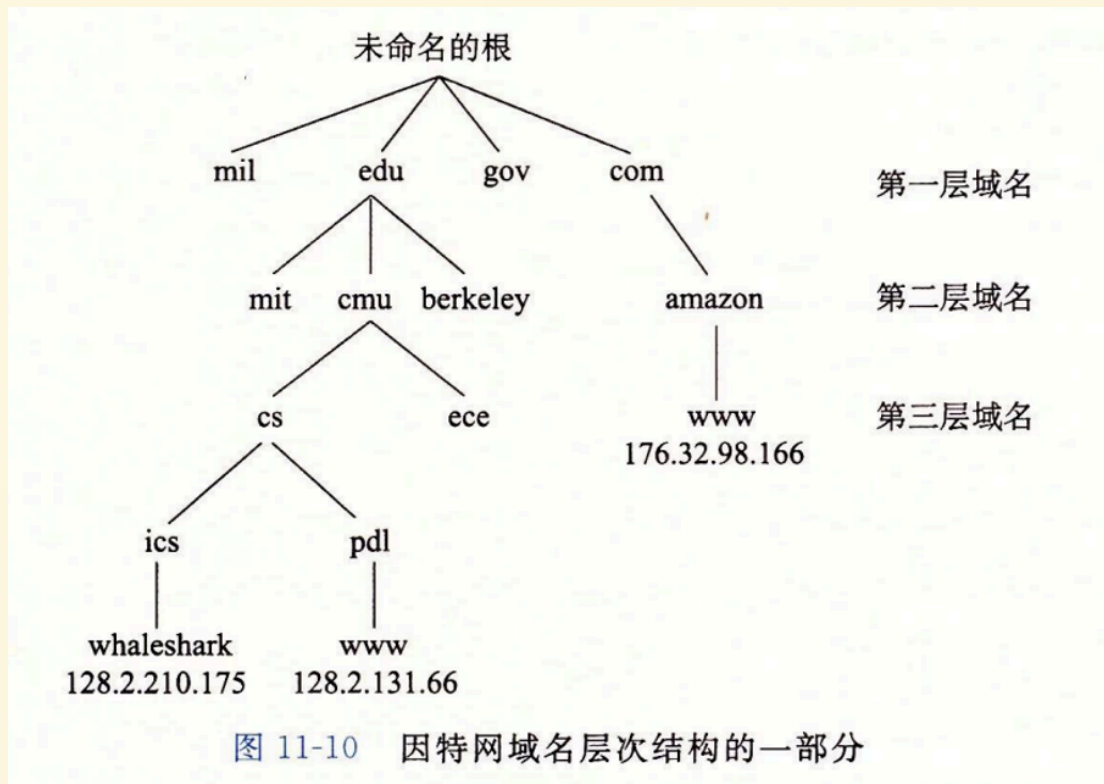
127.0.0.1



# 因特网域名

## DNS域名系统

维护域名集合和IP地址的映射



# 端口

16位整数

服务	端口号
echo	7
ftp	21
ssh	22
smtp	25
http	80
https	443

# 套接字连接

一个套接字是连接的一个端点

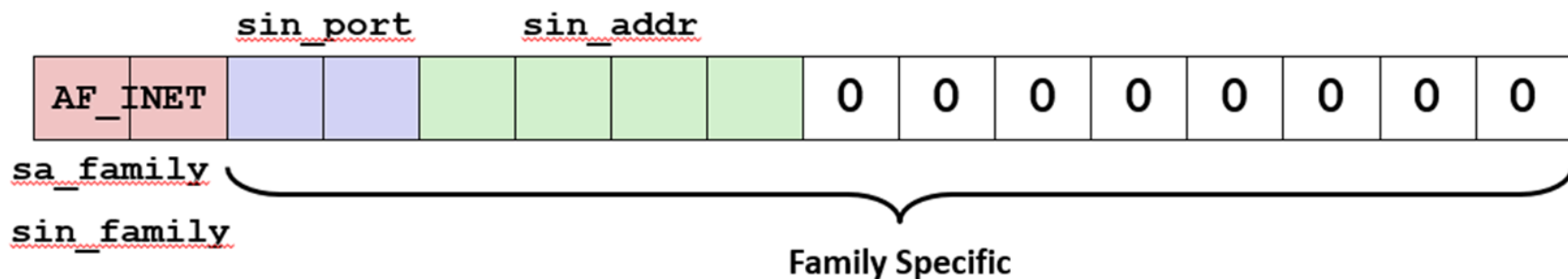
套接字地址： 因特网地址： 端口

套接字对 `(cliaddr:cliport, servaddr:servport)`

# 套接字接口

## 套接字地址

```
struct sockaddr_in {  
    uint16_t    sin_family; /* Protocol family (always AF_INET) */  
    uint16_t    sin_port;   /* Port num in network byte order */  
    struct in_addr sin_addr; /* IP addr in network byte order */  
    unsigned char sin_zero[8]; /* Pad to sizeof(struct sockaddr) */  
};
```





# 1. 创建 socket

## 客户端

## 服务器

调用 `socket()` 函数

配置地址族（如 IPv4）和通信类型（如 TCP）

```
server_socket =  
socket(AF_INET, SOCK_STREAM, 0);
```

## 2. 绑定地址（服务器端特有）

客户端

服务器

调用 `bind()` 函数将 socket 与 IP 和端口绑定。

```
struct sockaddr_in server_addr;  
server_addr.sin_family = AF_INET;  
  
// 任意本地 IP  
server_addr.sin_addr.s_addr = INADDR_ANY;  
// 转为网络字节序  
server_addr.sin_port = htons(8080);  
bind(server_socket,  
      (struct sockaddr*)&server_addr,  
      sizeof(server_addr));
```

# 3. 监听连接

## 客户端

## 服务器

调用 `listen()` 函数。  
进入监听状态，等待客户端连接。

```
listen(server_socket, 5);  
//在套接字接受队列中可以排队的最大连接数为5
```

## 4. 创建 socket

客户端

服务器

```
client_socket =  
socket(AF_INET, SOCK_STREAM, 0);
```

# 5. 建立连接

## 客户端

调用 `connect()` 函数，连接服务器的 IP 和端口。

```
connect(client_socket,  
        (struct sockaddr*)&server_addr,  
        sizeof(server_addr));
```

## 服务器

调用 `accept()` 函数，接受连接。  
需要用到监听的socket  
返回一个新的 socket

```
conn = accept(server_socket,  
              (struct sockaddr*)&client_addr,  
              &client_len);
```

# 6. 数据传输

## 客户端

```
Rio_writen(client_fd, msg, strlen(msg));  
Rio_readinitb(&rio, buf, MAXLINE);
```

## 服务器

```
Rio_readinitb(&rio, buf, MAXLINE);  
Rio_writen(conn, rsp, strlen(rsp));
```

# 7. 关闭连接

## 客户端

```
close(client_socket);
```

## 服务器

```
close(conn);
```

```
//在最终  
close(server_socket);
```

# 套接字接口的封装

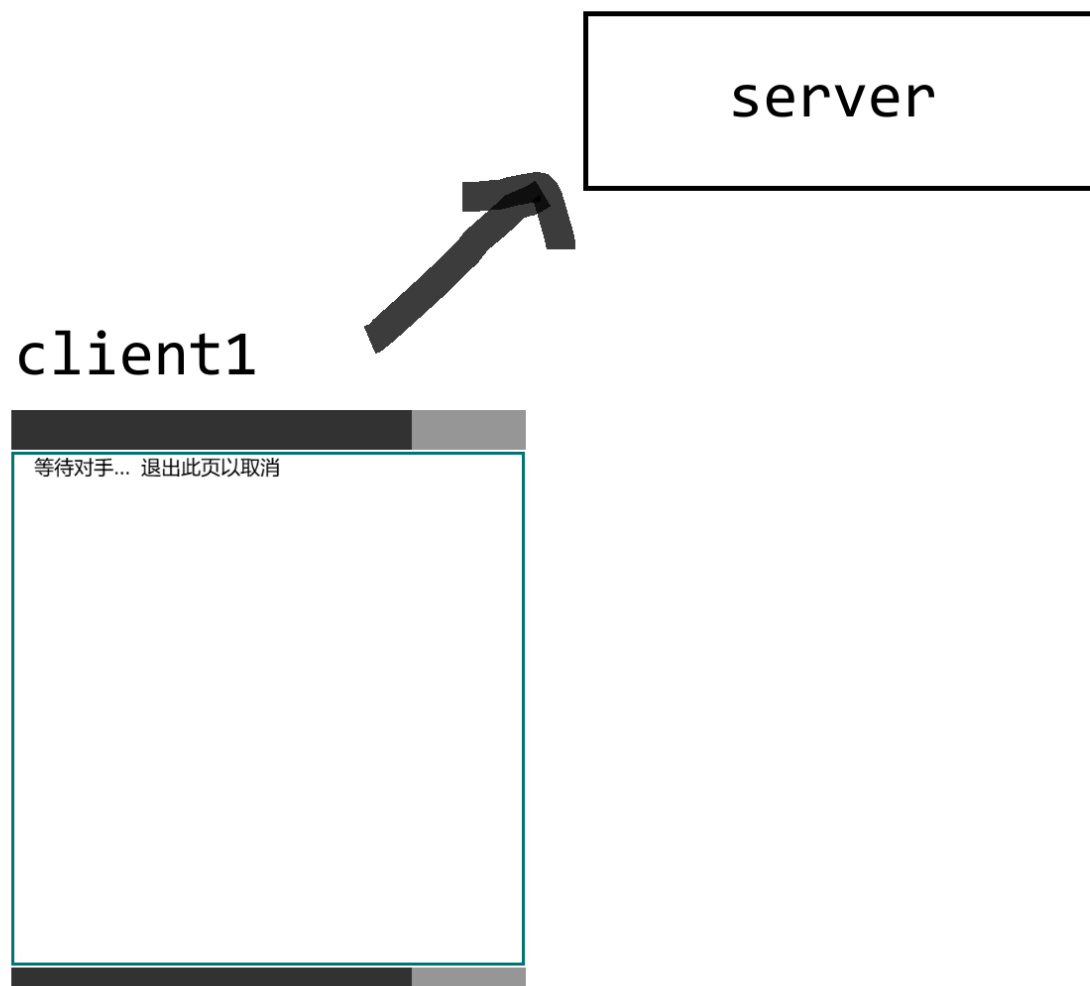
- 简化使用
- 隐藏底层细节
- 提高代码可读性

`open_clientfd(hostname, port)`, `open_listenfd(port)`

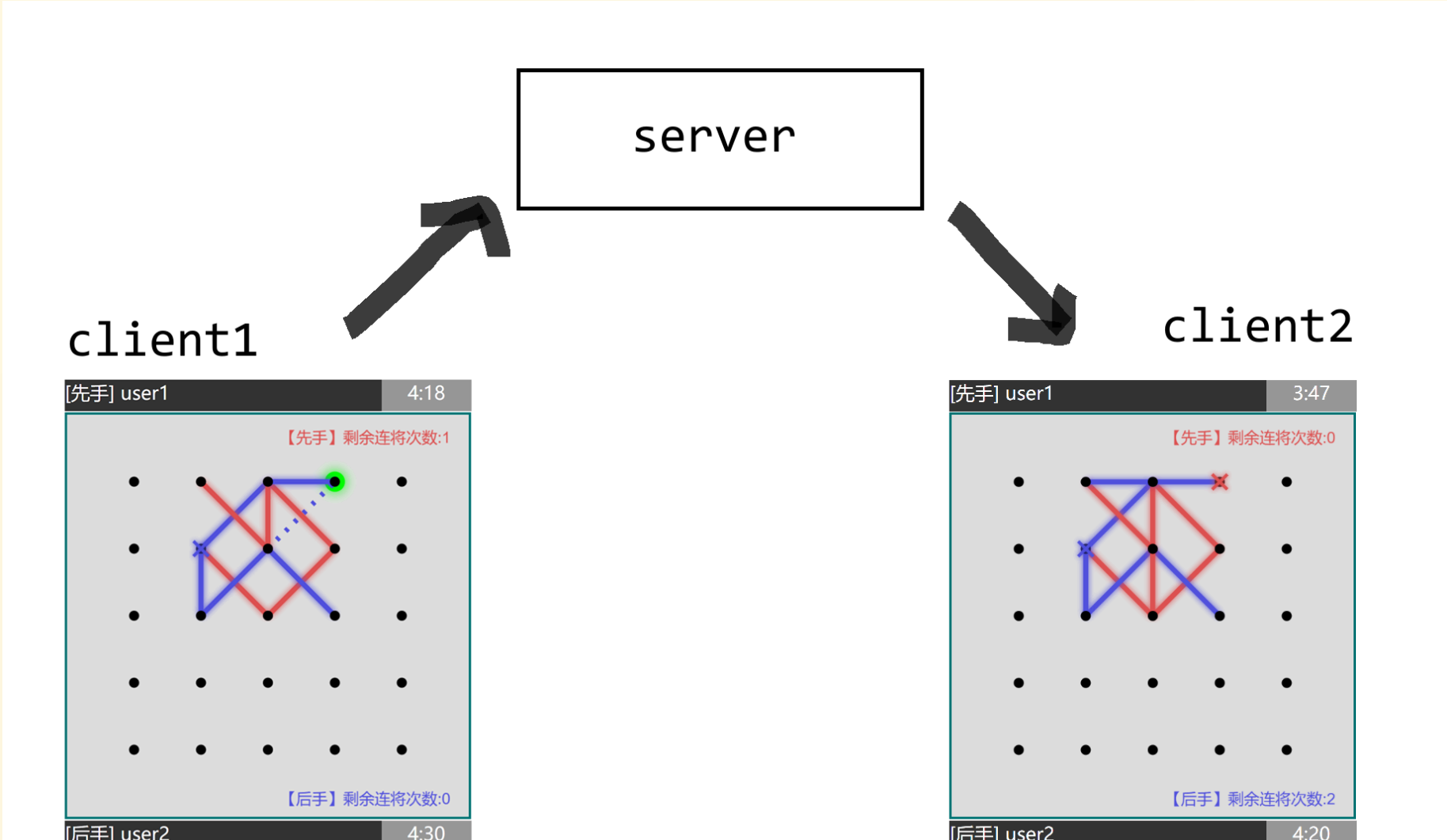
```
# Python
socket = create_socket("TCP")
connect(socket, "192.168.1.1", 8080)
send(socket, "Hello, World!")
response = receive(socket)
close(socket)
```



# 案例：联机棋类游戏



# 案例：联机棋类游戏



**thanks**