

计算机网络编程

第13章 基于UDP的客户机/服务器程序

信息工程学院 方徽星

fanghuixing@hotmail.com

大纲

- 设计目的
- 相关知识
- 例题分析

1. 设计目的

- **通过基于UDP的客户机与服务器程序设计**
 - **了解UDP协议的基本概念与主要功能**
 - **掌握这类网络应用的设计思路与编程方法**

2. 相关知识

- UDP协议的基本概念

UDP主要用于对传输效率要求高的应用层协议

应用层	BOOTP、RIP、SNMP、 TFTP、DNS、FTP...
传输层	TCP、 UDP 、SCTP
网络层	IP、ARP、IGMP、ICMP

无连接 + 不可靠的传输 + 数据包

2. 相关知识

- **UDP协议的基本概念**
 - **UDP校验和可选，没有保障传输可靠性的措施**
 - **UDP协议检测出数据报出错，则直接丢弃，既不确认，也不会要求重传**
- **使用场景**
 - **实时性要求高**
 - **部分数据包丢失不会造成重大影响**

2. 相关知识

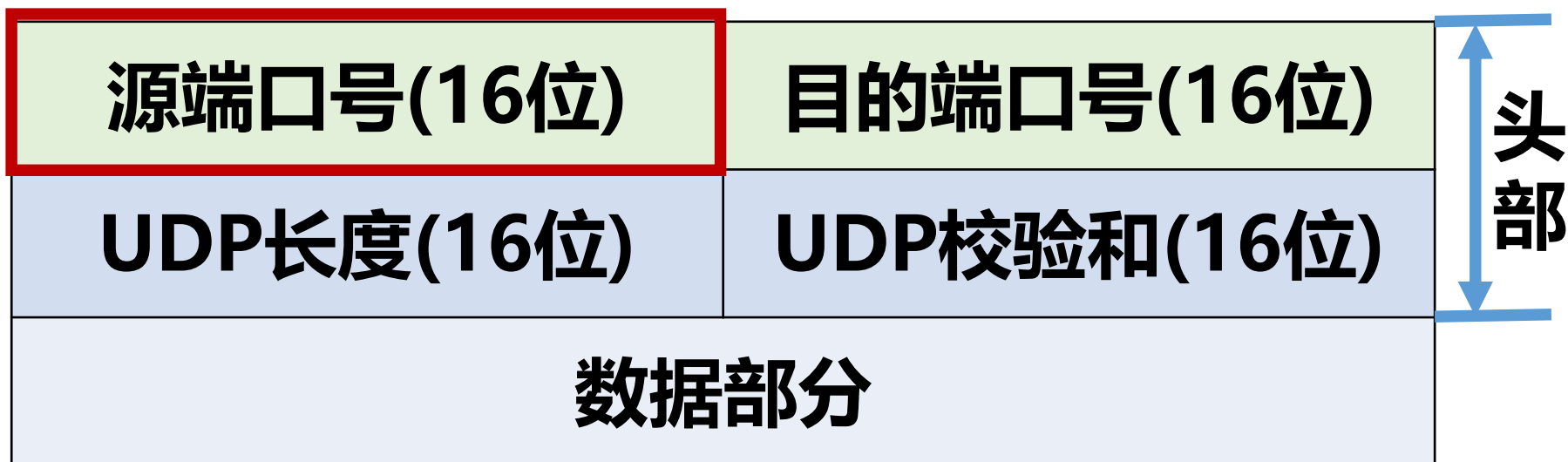
- UDP数据包的结构

源端口号(16位)	目的端口号(16位)	头部
UDP长度(16位)	UDP校验和(16位)	
数据部分		

头部固定8字节，数据部分长度可变

2. 相关知识

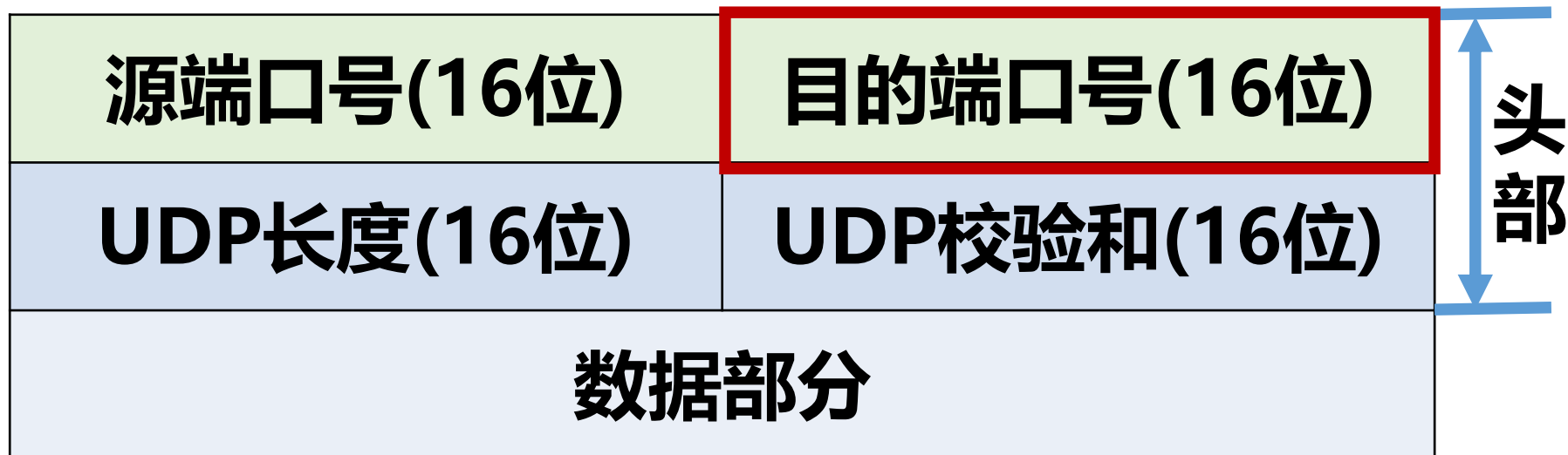
- UDP数据包的结构



源端口号：发送端进程使用的UDP端口号

2. 相关知识

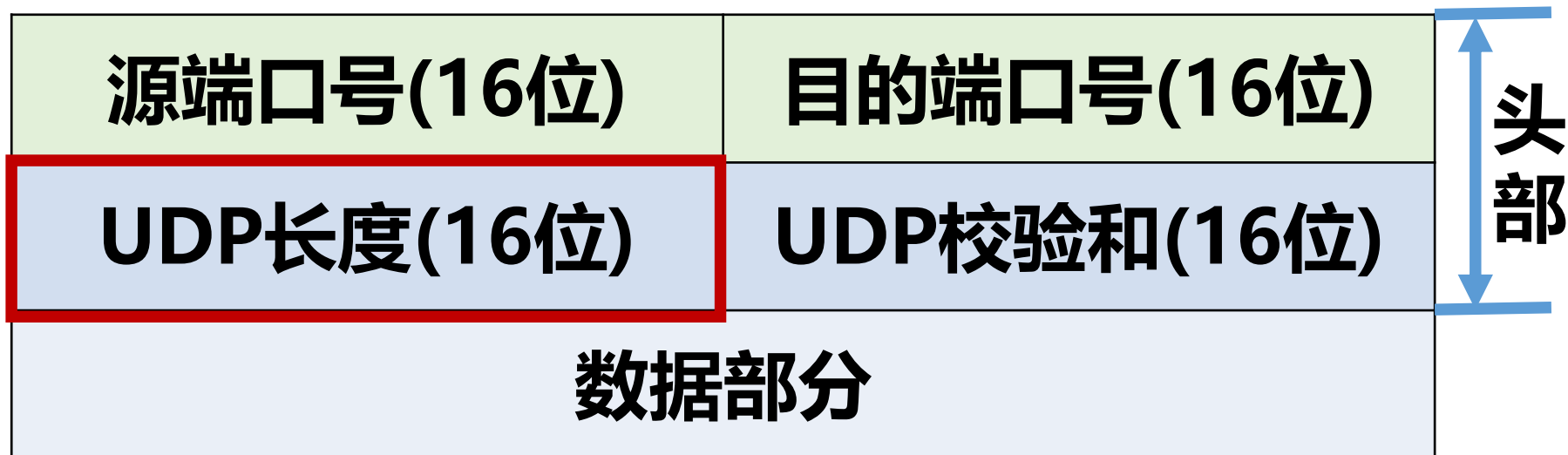
- UDP数据包的结构



目的端口号：接收端进程使用的UDP端口号

2. 相关知识

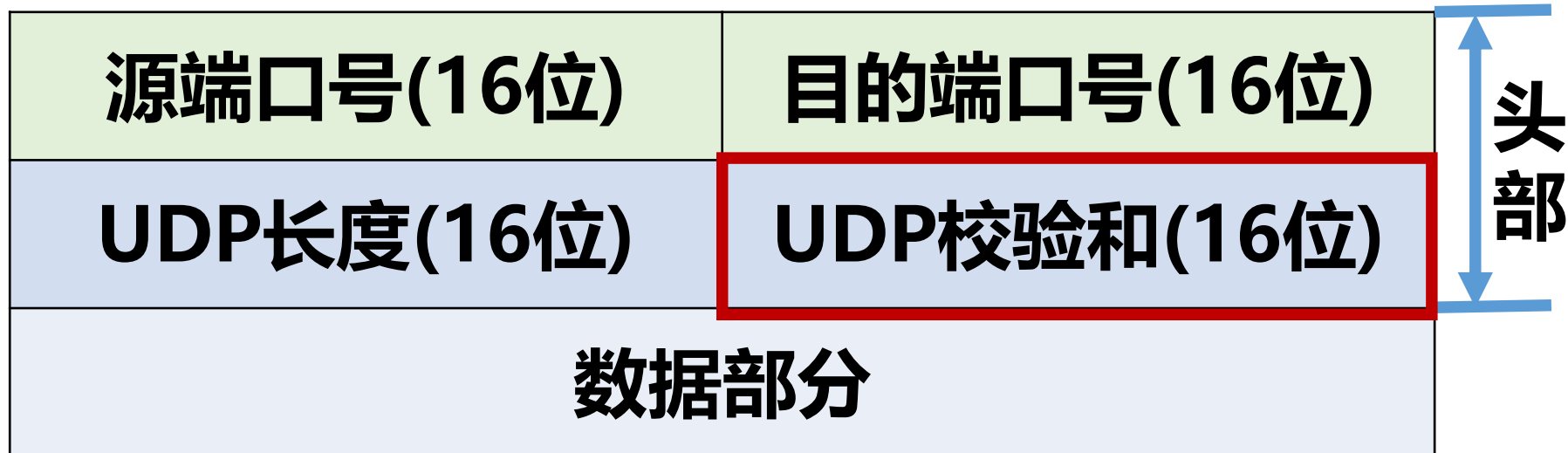
- UDP数据包的结构



UDP长度：包括头部在内的UDP数据包总长度
8B ~ 65535B

2. 相关知识

- UDP数据包的结构



UDP校验和： 校验范围包括伪头部、UDP头部和UDP数据； 可以不使用校验和

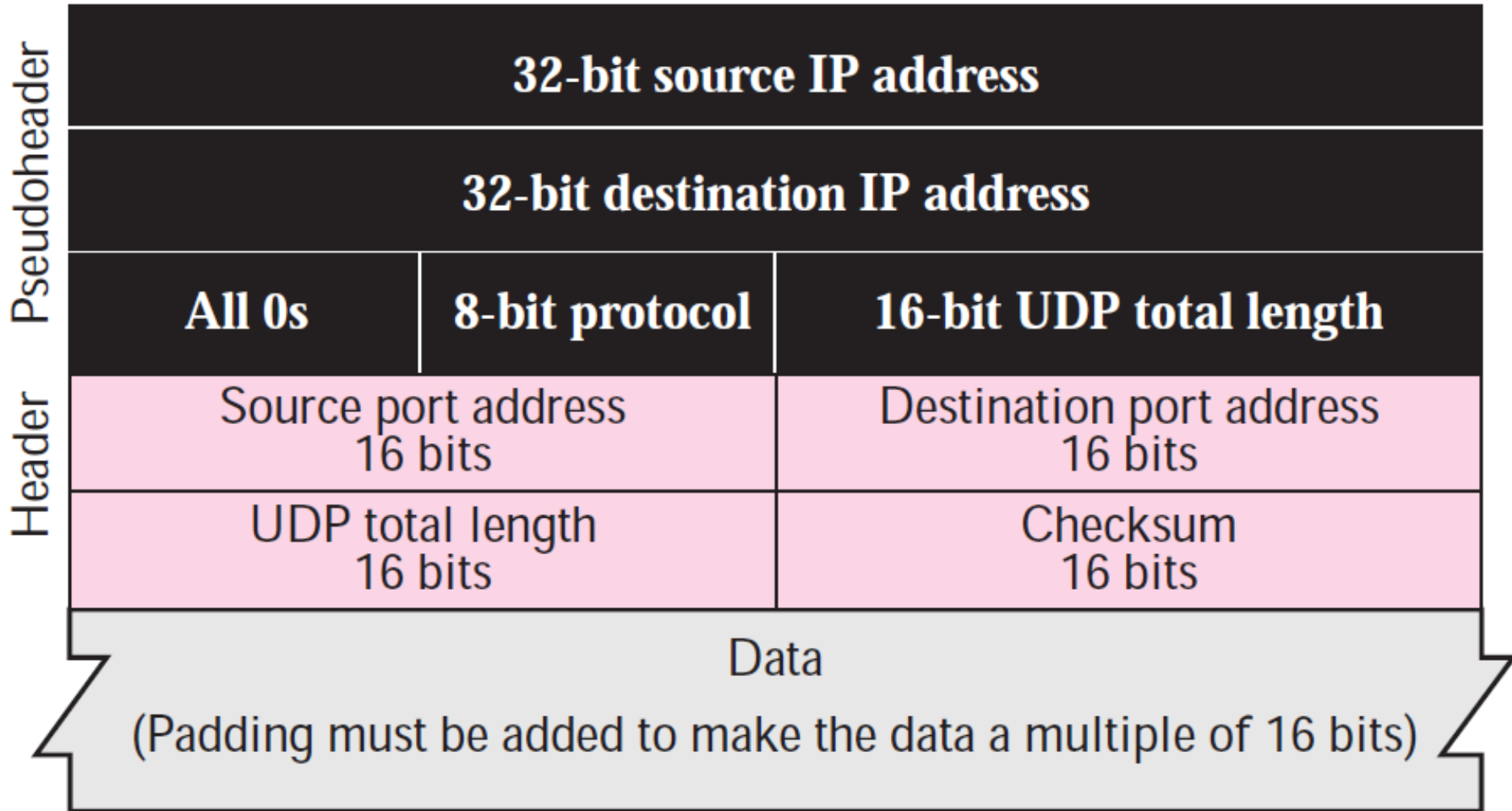
2. 相关知识

- UDP伪头部的结构

伪头部只是在计算校验和时临时和UDP数据包相加，长度12B



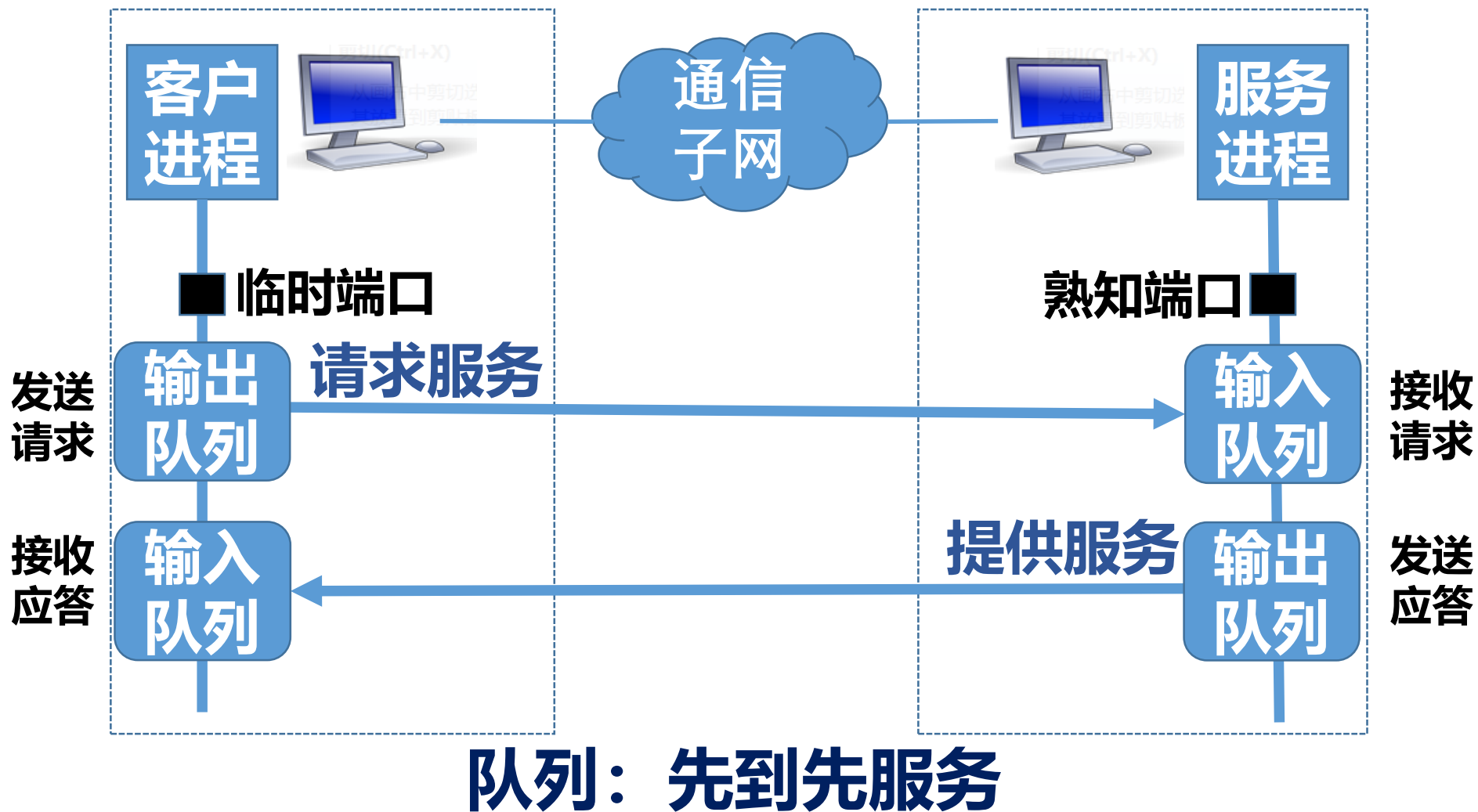
2. 相关知识



2. 相关知识

- **基于UDP的客户机/服务器编程**

基于UDP的客户机/服务器结构



主要的UDP熟知端口号

<i>Port</i>	<i>Protocol</i>	<i>Description</i>
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
53	Domain	Domain Name Service (DNS)
67	Bootps	Server port to download bootstrap information
68	Bootpc	Client port to download bootstrap information
69	TFTP	Trivial File Transfer Protocol
111	RPC	Remote Procedure Call
123	NTP	Network Time Protocol
161	SNMP	Simple Network Management Protocol
162	SNMP	Simple Network Management Protocol (trap)

<https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>

3. 例题分析：设计要求

- 基于UDP的客户机/服务器工作模式，编写服务器程序接收客户机的命令
- 并根据命令向客户机做出响应
 - 当客户机向服务器发送getfile命令时，服务器向客户机发送指定文件中的数据
 - 当客户机向服务器发送gettime命令时，服务器向客户机发送当前系统时间

3. 例题分析：具体要求

- 要求程序为命令程序

UdpServer server_port



**服务器侦听
的端口号**

3. 例题分析：具体要求

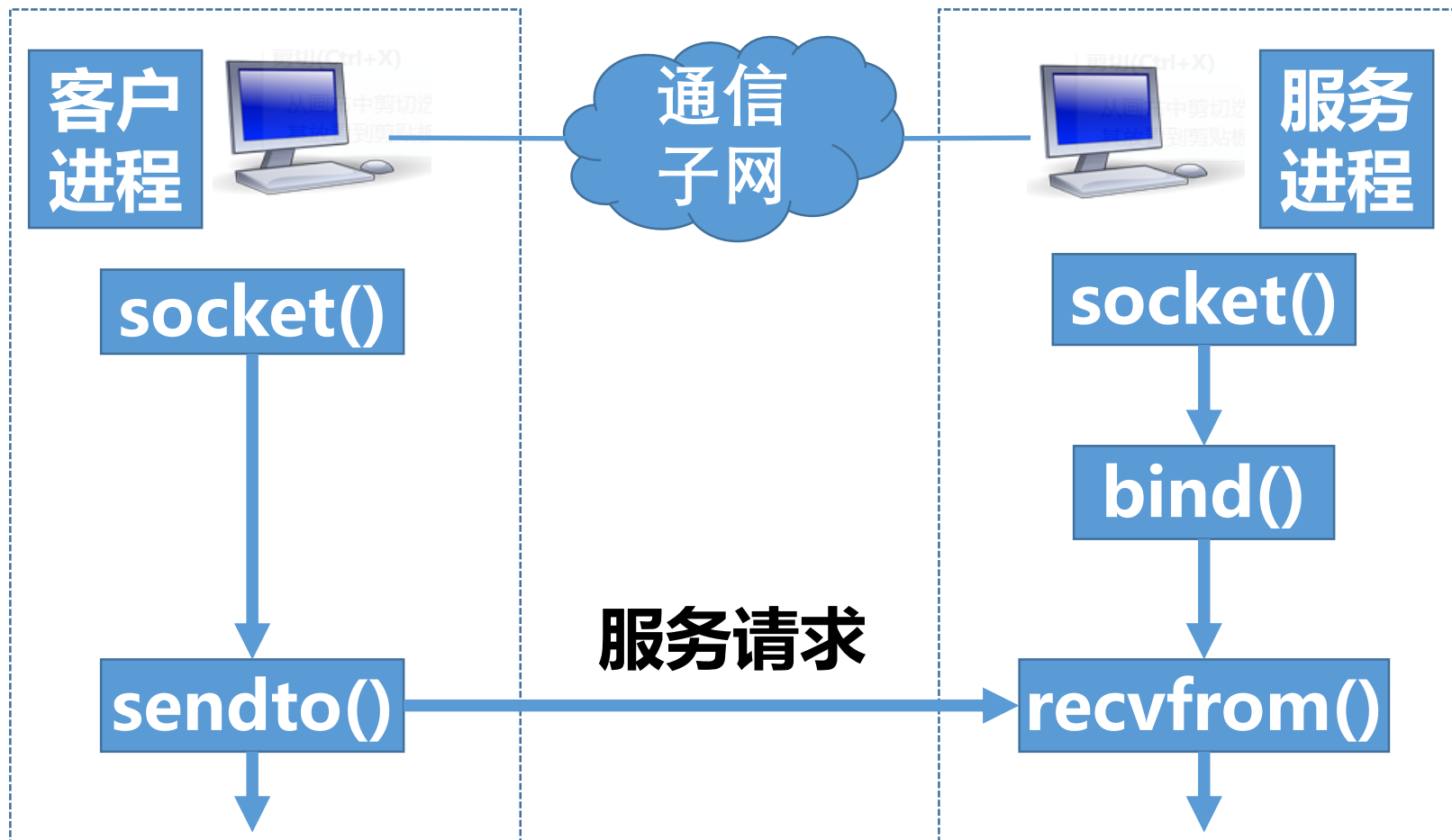
- 要求将服务器的状态显示在控制台上，具体格式：

UDP Server接收命令： ...

UDP Server发送数据： ...

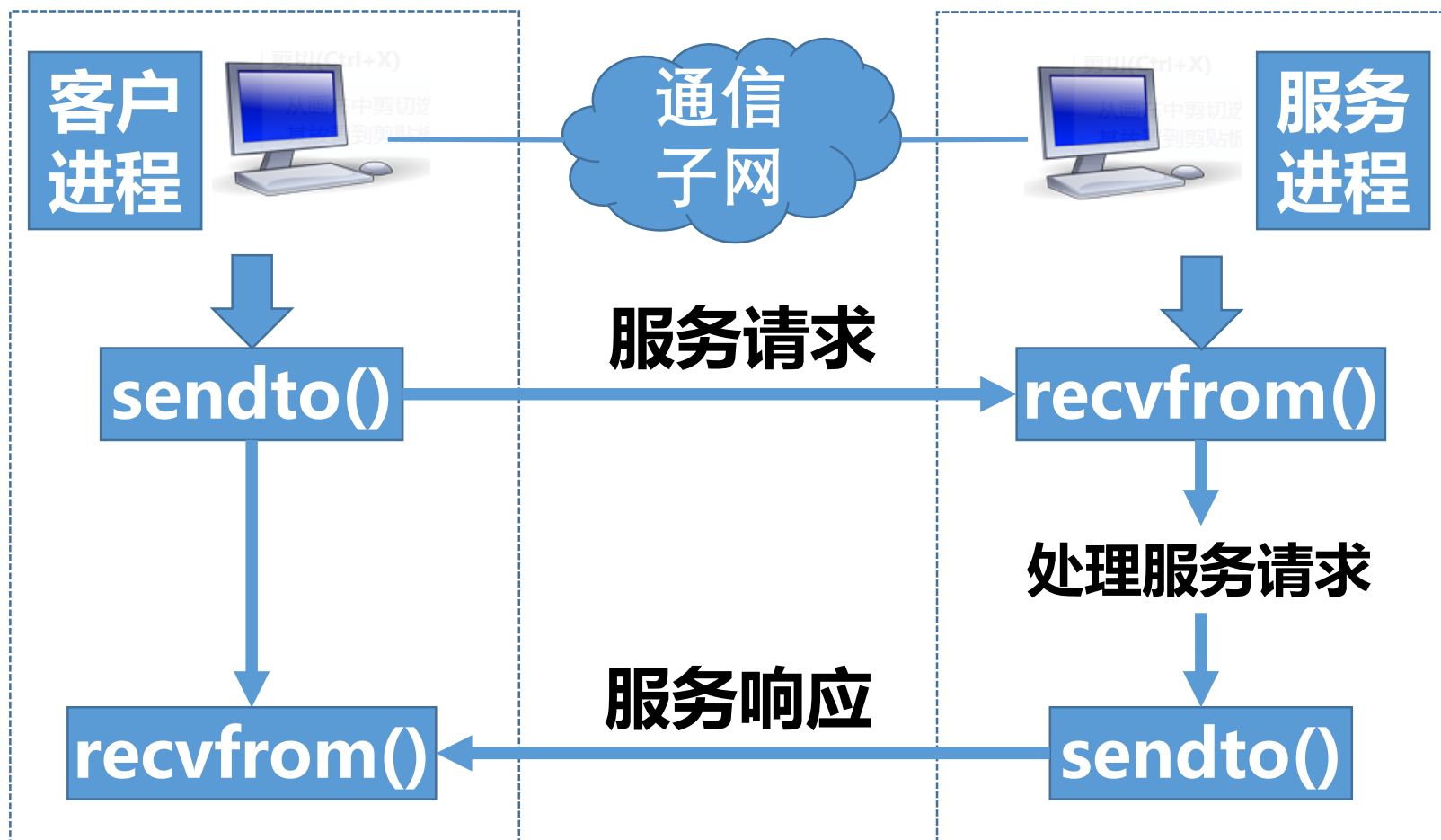
3. 例题分析：关键问题

• 基本编程模式分析



3. 例题分析：关键问题

• 基本编程模式分析



3. 例题分析： 关键问题

• 创建数据报套接字

```
//创建数据报套接字
sock=socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);

//填充本地套接字地址
sockaddr_in serveraddr;
serveraddr.sin_family = AF_INET;
serveraddr.sin_port = htons((unsigned short)atoi(argv[1]));
serveraddr.sin_addr.S_un.S_addr=htonl(INADDR_ANY);

//将套接字与套接字地址绑定
bind(sock, (sockaddr*)&serveraddr, sizeof(serveraddr));
```

3. 例题分析：关键问题

- 在主循环中发送和接收数据

```
while(true)
{
    初始化接收缓冲区
    从客户机接收请求
    初始化发送缓冲区
    判断请求的类型
    数据写入发送缓冲区
    向客户机发送数据
}
```

3. 例题分析：关键问题

- 在主循环中发送和接收数据

```
//初始化接收缓冲区  
char recvbuf[20];  
memset(recvbuf, 0, sizeof(recvbuf));  
...
```

3. 例题分析：关键问题

- 在主循环中发送和接收数据

...

//从客户机接收请求

nRecv = recvfrom(

 sock, //用于接收数据的UDP套接字句柄

 recvbuf, //保存接收数据的缓冲区

 sizeof(recvbuf), //可接收的最大字节数

 0, //可选项参数，若没有则传入0

 (sockaddr*)&clientaddr, //发送端地址信息

 &clientaddrlen);

...

3. 例题分析：关键问题

- 在主循环中发送和接收数据

```
...  
//初始化发送缓冲区  
char sendbuf[1500];  
memset(sendbuf, 0, sizeof(sendbuf));  
...
```

//判断客户机请求的类型

if(strcmp(recvbuf, "getfile")==0) //getfile命令

{

 //将文件数据写入发送缓冲区

 fstream infile;

 infile.open("input" , ios::in);

 infile.read(sendbuf, nlength);

 //向客户机发送数据

 nsend=sendto(sock,

 sendbuf, //保存待传数据的缓冲区

 sizeof(sendbuf), //待传字节数

 0, //可选项参数, 若没有则传递0

 (sockaddr*)&clientaddr, //目标地址

 clientaddrlen);

}

//判断客户机请求的类型

if(strcmp(recvbuf, "gettime")==0)//gettime命令

{

 //将系统时间写入发送缓冲区

 time_t CurTime;

 time(&CurTime);

 strftime(sendbuf,
 sizeof(sendbuf),
 "%Y-%m-%d %H:%M:%S" ,
 localtime(&CurTime));

 //向客户机发送数据

 nsend=sendto(sock, sendbuf, sizeof(sendbuf), 0,
 (sockaddr*)&clientaddr, clientaddrlen);

}

//设置当前日历时间到CurTime

time(&CurTime);

//将时间值格式化后，存储到sendbuf中

strftime(sendbuf,

sizeof(sendbuf),

"%Y-%m-%d %H:%M:%S" ,

localtime(&CurTime));

按照本地时区要求填充结构tm，然后返回

localtime(&CurTime)

Member	Type	Meaning	Range
tm_sec	int	seconds after the minute	0-60*
tm_min	int	minutes after the hour	0-59
tm_hour	int	hours since midnight	0-23
tm_mday	int	day of the month	1-31
tm_mon	int	months since January	0-11
tm_year	int	years since 1900	
tm_wday	int	days since Sunday	0-6
tm_yday	int	days since January 1	0-365
tm_isdst	int	Daylight Saving Time flag	

结构体tm部分成员说明

int tm_mon;

月份(从一月开始, 0代表一月), 取值区间为[0,11]

int tm_year;

年份, 其值等于实际年份减去1900

int tm_wday;

星期, 取值区间为[0,6], 其中0代表星期天, 1代表星期一, 以此类推

结构体tm部分成员说明

int tm_yday;

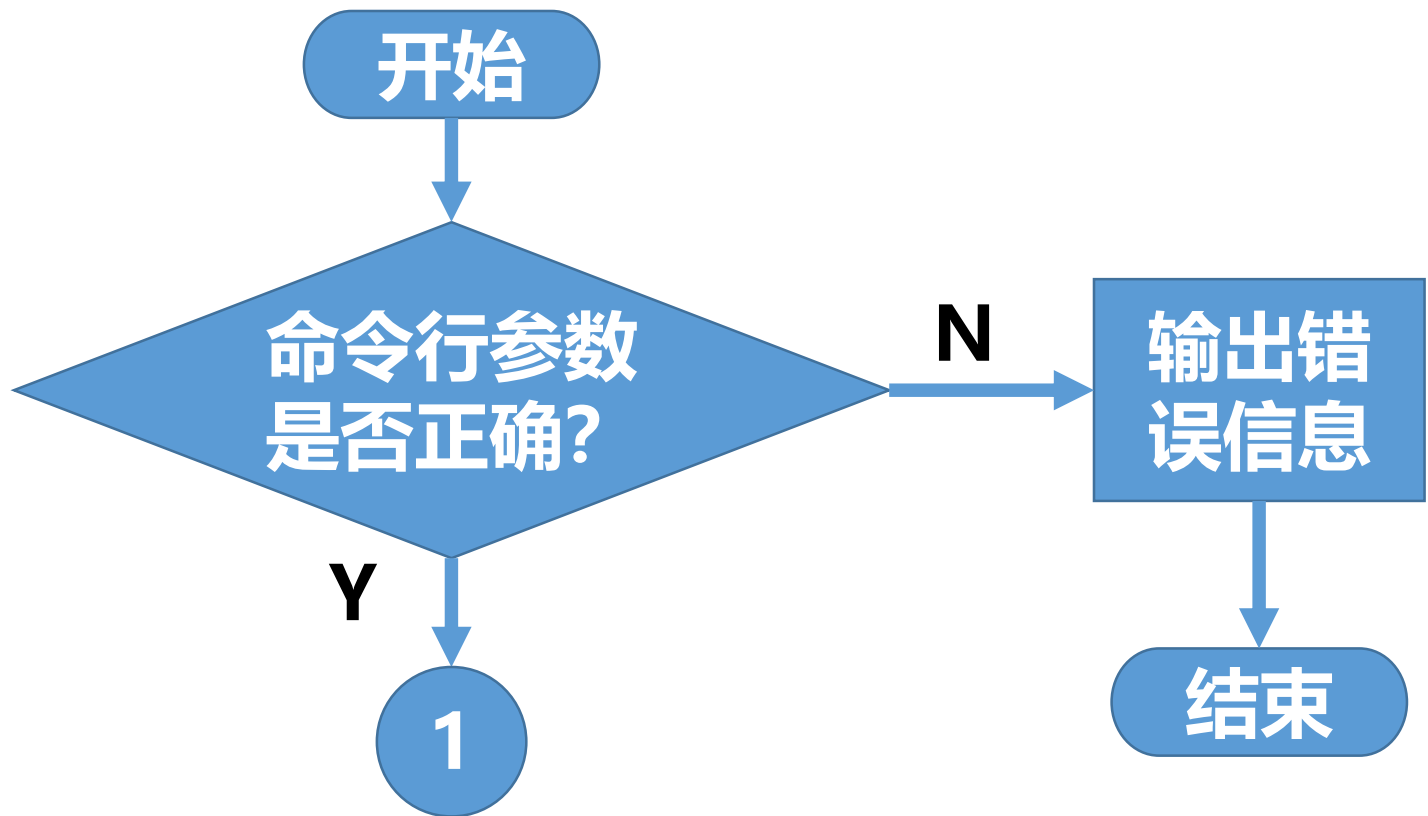
从每年的1月1日开始的天数，取值区间为[0,365]，其中0代表1月1日，1代表1月2日，以此类推

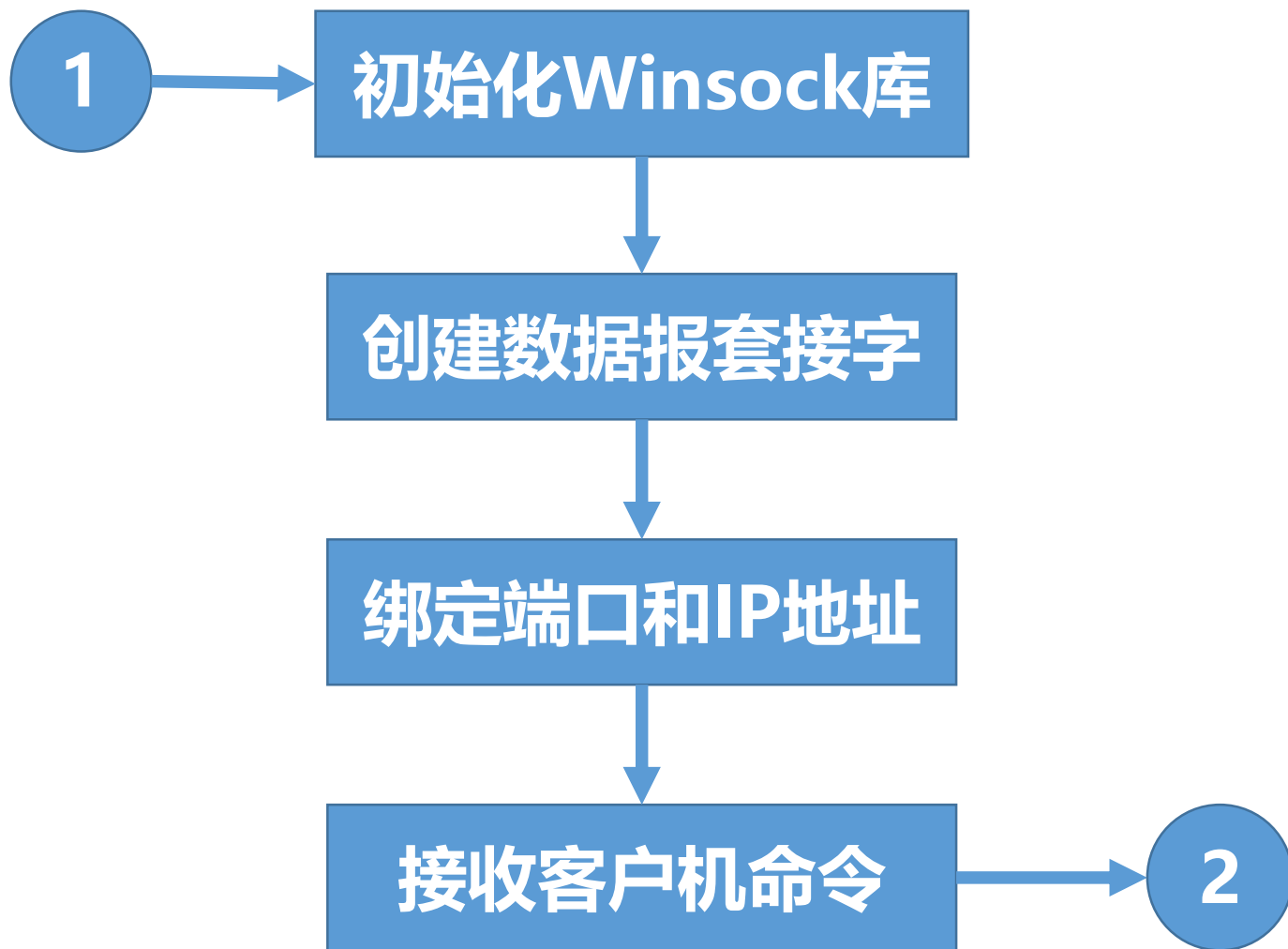
int tm_isdst;

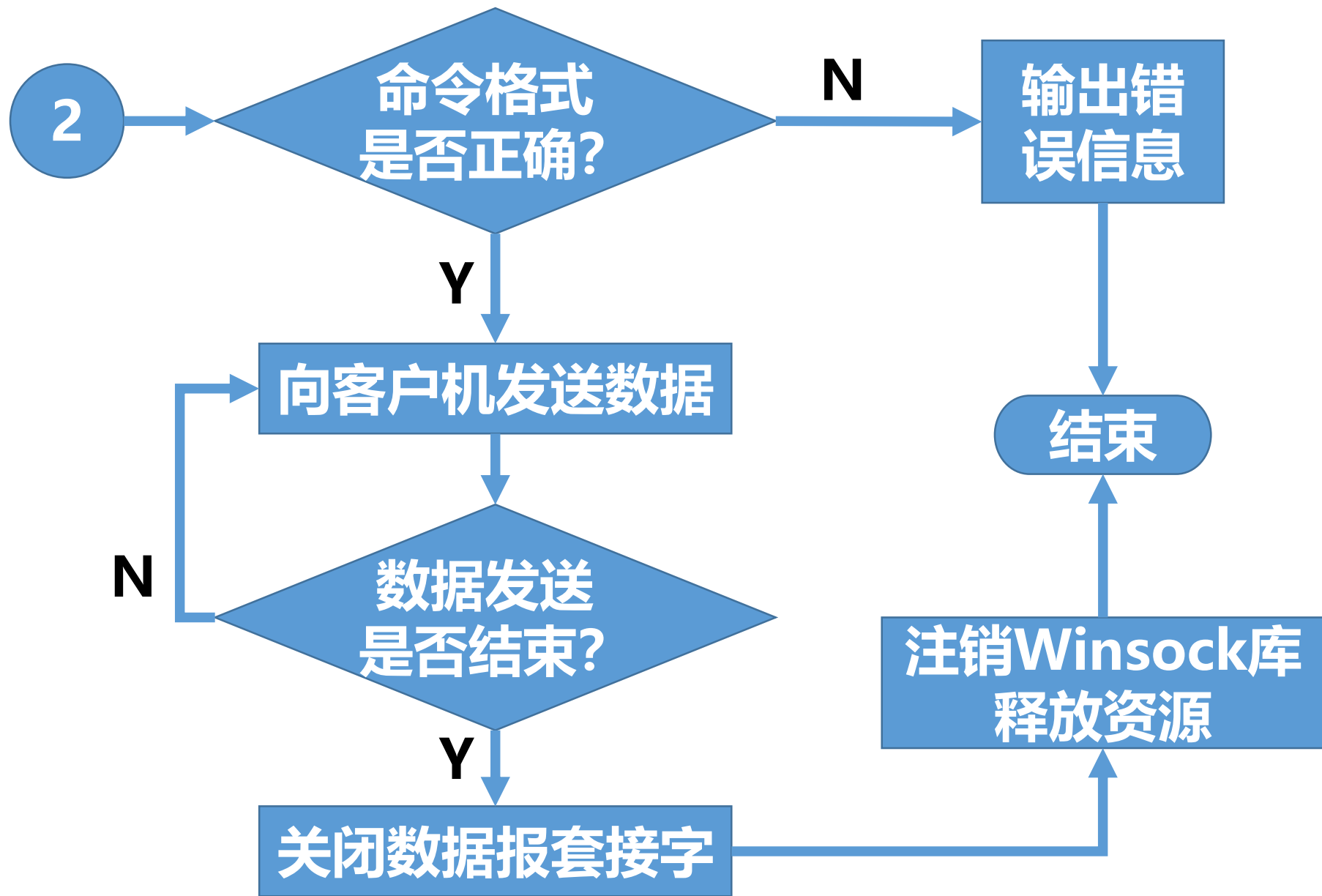
夏令时标识符，实行夏令时的时候，tm_isdst为正。不实行夏令时的进候，tm_isdst为0；不了解情况时，tm_isdst为负

3. 例题分析：关键问题

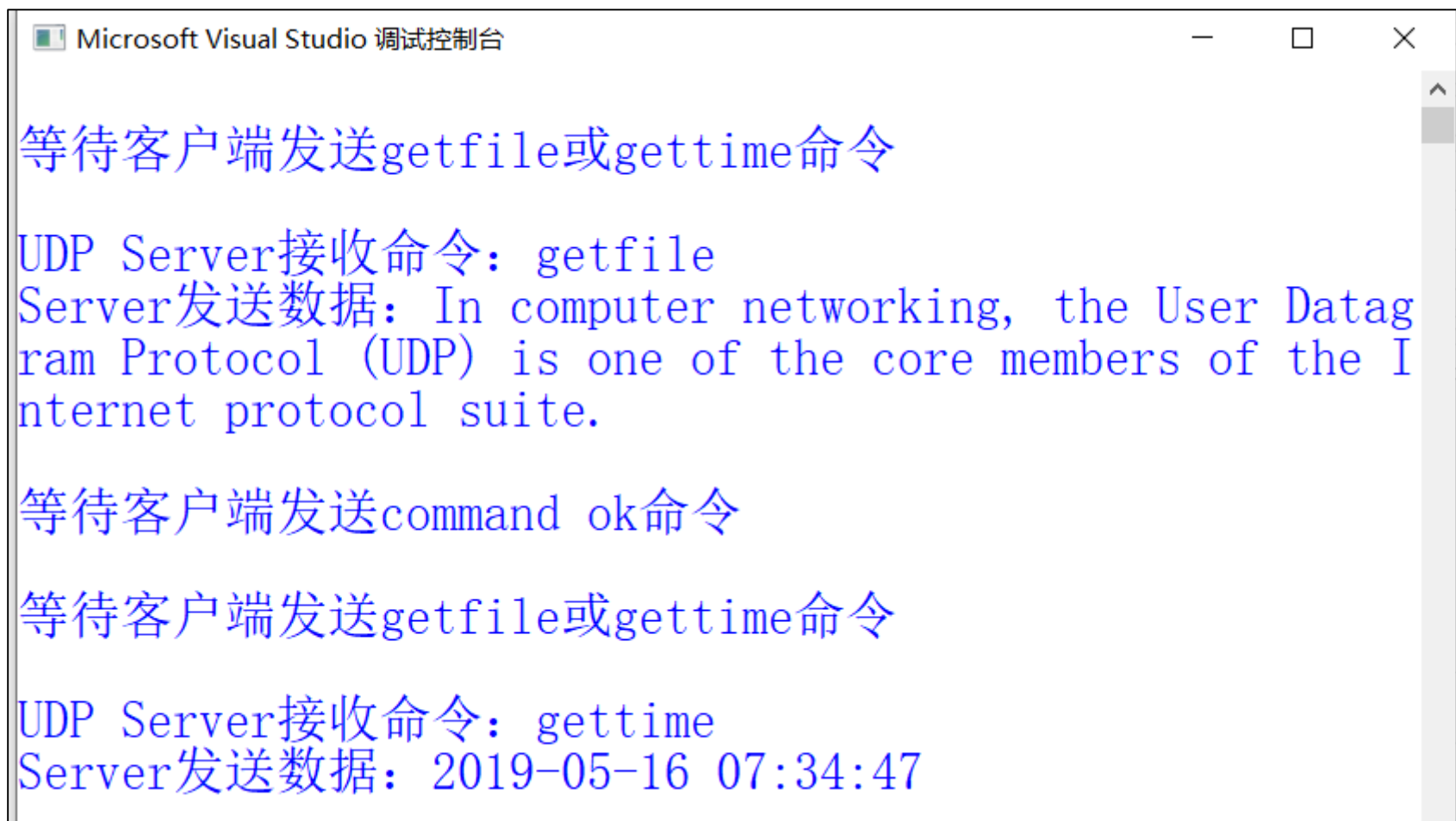
- 程序流程图







程序演示



```
Microsoft Visual Studio 调试控制台

等待客户端发送getfile或gettime命令

UDP Server接收命令: getfile
Server发送数据: In computer networking, the User Datagram Protocol (UDP) is one of the core members of the Internet protocol suite.

等待客户端发送command ok命令

等待客户端发送getfile或gettime命令

UDP Server接收命令: gettime
Server发送数据: 2019-05-16 07:34:47
```

本章小结

- **设计目的**

- **了解UDP协议的基本概念与主要功能，掌握这类网络应用的设计思路与编程方法**

- **相关知识**

- **UDP协议基本概念**
 - **UDP数据包结构**
 - **基于UDP的客户机/服务器编程**

- **例题分析**

- **基本编程模式分析、数据报套接字、发送和接收**