

UDP C/S 结构程序开发—以 Ping 程序为例

71118415 叶宏庭

东南大学软件学院

Email: 213182964@seu.edu.cn

May 17, 2021

1 实验目的

了解 UDP 通信原理、UDP 通信与 TCP 通信的区别，掌握 UDP C/S 结构程序开发。了解 Ping 指令的原理，完成 Ping 程序开发。

2 实验环境

2.1 操作系统:

Ubuntu 20.04

2.2 辅助软件:

CTEX(用于编写 tex 报告)

3 实验内容

3.1 UDP 通信原理:

UDP 是 OSI 参考模型中一种无连接的传输层协议，它主要用于不要求分组顺序到达的传输中，分组传输顺序的检查与排序由应用层完成，提供面向事务的简单不可靠信息传送服务。UDP 协议基本上是 IP 协议与上层协议的接口。UDP 协议适用端口分别运行在同一台设备上的多个应用程序。

UDP 报文没有可靠性保证、顺序保证和流量控制字段等，可靠性较差。但是正因为 UDP 协议的控制选项较少，在数据传输过程中延迟小、数据传输效率高，适合对可靠性要求不高的应用程序，或者可以保障可靠性的应用程序，如 DNS、TFTP、SNMP 等

3.2 UDP vs TCP:

1. 基于连接与无连接

TCP 是面向连接的协议，而 UDP 是面向无连接的协议。

2. 对系统资源的要求

TCP 是面向连接的可靠协议，因此对系统资源提出了更高的要求，而 UDP 因为无连接不可靠的特性，所以对系统资源要求低。

3. 程序结构复杂度

基于 TCP 协议的程序相对更为复杂，而 UDP 程序则更为简单。

4. 流模式与数据报模式

TCP 的传输是基于流模式，而 UDP 的传输是基于数据报。

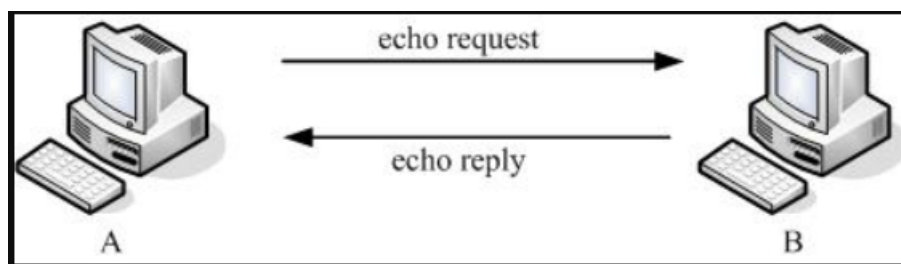
5. 可靠性

TCP 通过应答机制，实现了数据传输的正确性、可靠性，而 UDP 并没有应答机制，所以容易产生丢包。

3.3 Ping 指令原理:

Ping 命令利用 ICMP 协议进行工作，ICMP 是 Internet 控制消息协议，用于在主机和路由器之间传递控制消息。Ping 命令利用了 ICMP 两种类型的控制消息：“echo request”（回显请求）、“echo reply”（回显应答）。

比如在主机 A 上执行 ping 命令，目标主机是 B。在 A 主机上就会发送“echo request”（回显请求）控制消息，主机 B 正确接收后即发回“echo reply”（回显应答）控制消息，从而判断出双方能否正常通信。其工作原理如下图所示。



如果在 A 主机上能够 ping 通 B 主机，那么主机 A 上显示的信息就是从主机 B 上返回来的“回显应答”。如果不能 ping 通，主机 A 上显示的信息则是由系统自身所产生的错误提示。

3.4 Ping 程序开发:

统计 ICMP 报文结果

```

1 void statistics(int signo){
2     printf("\n-----PING statistics-----\n");
3     printf("%d packets transmitted, %d received, %%%d lost\n", nsend, nreceived,

```

```

4         (nsend-nreceived)/nsend*100);
5     close(sockfd);
6     exit(1);
7 }

```

校验和算法

```

1 unsigned short cal_chksum(unsigned short *addr,int len)
2 {
3     int nleft=len;
4     int sum=0;
5     unsigned short *w=addr;
6     unsigned short answer=0;
7
8     /*
9      The ICMP header binary data is added up in units of 2 bytes.
10    */
11    while(nleft>1)
12    {
13        sum+=*w++;
14        nleft-=2;
15    }
16    /*
17     If the ICMP header has an odd number of bytes, the last byte is left.
18     Consider the last byte as the high byte of a 2-byte data,
19     the low byte of which is 0, and continue to add
20    */
21    if(nleft==1)
22    {
23        *(&answer)=*(unsigned char *)w;
24        sum+=answer;
25    }
26    sum=(sum>>16)+(sum&0xffff);
27    sum+=(sum>>16);
28    answer=~sum;
29    return answer;
30 }

```

设置 ICMP 报头

```

1 int pack(int pack_no)
2 {
3     int i,packsize;
4     struct icmp *icmp;
5     struct timeval *tval;
6
7     icmp=(struct icmp*)sendpacket;

```

```

7     icmp->icmp_type=ICMP_ECHO;
8     icmp->icmp_code=0;
9     icmp->icmp_cksum=0;
10    icmp->icmp_seq=pack_no;
11    icmp->icmp_id=pid;
12    packsize=8+datalen;
13    tval= (struct timeval *)icmp->icmp_data;
14    gettimeofday(tval, NULL);    /*记录发送时间*/
15    icmp->icmp_cksum=cal_chksum( (unsigned short *)icmp, packsize); /*校验算法*/
16    return packsize;
17 }

```

发送 3 个 ICMP 报文

```

1 void send_packet()
2 {
3     int packsize;
4     while( nsend<MAX_NO_PACKETS)
5     {
6         nsend++;
7         packsize=pack(nsend); /*设置ICMP报头*/
8         if( sendto(sockfd, sendpacket, packsize, 0,
9             (struct sockaddr *)&dest_addr, sizeof(dest_addr)) < 0 )
10        {
11            perror("sendto error");
12            continue;
13        }
14        sleep(1); /*每隔一秒发送一个ICMP报文*/
15    }
16 }

```

两个 timeval 结构相减

```

1 void tv_sub(struct timeval *out, struct timeval *in)
2 {
3     if( (out->tv_usec-=in->tv_usec)<0)
4     {
5         —out->tv_sec;
6         out->tv_usec+=1000000;
7     }
8     out->tv_sec-=in->tv_sec;
9 }

```

接受 ICMP 报文与剥去报文头部分请详见完整代码。

主函数部分

```

1 main(int argc, char *argv[])
2 {
3     struct hostent *host;

```

```

3      struct protoent *protocol;
4      unsigned long inaddr=0l;
5      int waittime=MAX_WAIT_TIME;
6      int size=50*1024;
7
8      if(argc<2)
9      {
10         printf("usage:%s hostname/IP address\n",argv[0]);
11         exit(1);
12     }
13
14     if( (protocol=getprotobyname("icmp") )==NULL)
15     {
16         perror("getprotobyname");
17         exit(1);
18     }
19     /*生成使用ICMP的原始套接字,这种套接字只有root才能生成*/
20     if( (sockfd=socket(AF_INET,SOCK_RAW,protocol->p_proto) )<0)
21     {
22         perror("socket error");
23         exit(1);
24     }
25     /* 回收root权限,设置当前用户权限*/
26     setuid(getuid());
27     /*扩大套接字接收缓冲区到50K这样做主要为了减小接收缓冲区溢出的
28     的可能性,若无意中ping一个广播地址或多播地址,将会引来大量应答*/
29     setsockopt(sockfd,SOL_SOCKET,SO_RCVBUF,&size,sizeof(size));
30     bzero(&dest_addr,sizeof(dest_addr));
31     dest_addr.sin_family=AF_INET;
32
33     /*判断是主机名还是ip地址*/
34     if( inaddr=inet_addr(argv[1])==INADDR_NONE)
35     {
36         if((host=gethostbyname(argv[1]) )==NULL) /*是主机名*/
37         {
38             perror("gethostbyname error");
39             exit(1);
40         }
41         memcpy( (char *)&dest_addr.sin_addr,host->h_addr,host->h_length);
42     }
43     else /*是ip地址*/
44     memcpy( (char *)&dest_addr,(char *)&inaddr,4); //修改host->h_length为4
45     /*获取main的进程id,用于设置ICMP的标志符*/
46     pid=getpid();
47     printf("PING %s(%s): %d bytes data in ICMP packets.\n",argv[1],
48     inet_ntoa(dest_addr.sin_addr),datalen);

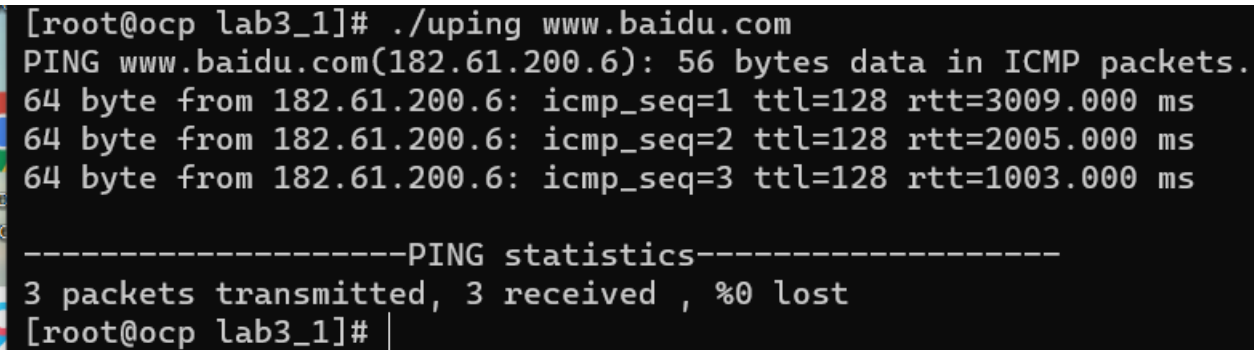
```

```
44     send_packet(); /*发送所有ICMP报文*/
45     recv_packet(); /*接收所有ICMP报文*/
46     statistics(SIGALRM); /*进行统计*/
47
48     return 0;
49
50 }
```

4 实验结果与分析

首先给出程序的运行结果：

4.1 Ping www.baidu.com



```
[root@ocp lab3_1]# ./uping www.baidu.com
PING www.baidu.com(182.61.200.6): 56 bytes data in ICMP packets.
64 byte from 182.61.200.6: icmp_seq=1 ttl=128 rtt=3009.000 ms
64 byte from 182.61.200.6: icmp_seq=2 ttl=128 rtt=2005.000 ms
64 byte from 182.61.200.6: icmp_seq=3 ttl=128 rtt=1003.000 ms

-----PING statistics-----
3 packets transmitted, 3 received , %0 lost
[root@ocp lab3_1]# |
```

从图中可以清楚看到，本程序可以通过主机名顺利 Ping 通对应的主机，并且输出信息与电脑自带的 ping 结果相同。(包含 IP 地址、序列号、RTT 延迟等)

4.2 Ping 58.192.118.142(东南大学主机地址)

```
[root@ocp lab3_1]# ./uping www.seu.edu.cn
PING www.seu.edu.cn(58.192.118.142): 56 bytes data in ICMP packets.
64 byte from 58.192.118.142: icmp_seq=1 ttl=128 rtt=3010.000 ms
64 byte from 58.192.118.142: icmp_seq=2 ttl=128 rtt=2006.000 ms
64 byte from 58.192.118.142: icmp_seq=3 ttl=128 rtt=1003.000 ms

-----PING statistics-----
3 packets transmitted, 3 received , %0 lost
[root@ocp lab3_1]# ./uping 58.192.118.142
PING 58.192.118.142(0.0.0.0): 56 bytes data in ICMP packets.
64 byte from 127.0.0.1: icmp_seq=1 ttl=64 rtt=3019.000 ms
64 byte from 127.0.0.1: icmp_seq=2 ttl=64 rtt=2014.000 ms
64 byte from 127.0.0.1: icmp_seq=3 ttl=64 rtt=1012.000 ms

-----PING statistics-----
3 packets transmitted, 3 received , %0 lost
```

我们首先采用 Ping 主机名的方式，获取 www.seu.edu.cn 的 IP 地址，再直接 Ping 对应的 IP 地址。

4.3 总结

通过本次实验，首先了解了 UDP 的通信原理，并且掌握了其 C/S 结构的程序设计。其次还深入了解了 Ping 指令的原理。实现了一个自己的 Ping 程序，并且取得了良好的结果。