信息网络与协议

实验指导书

（2023 秋季学期）

课 程 教 师

卢 汉 成　　hclu@ustc.edu.cn

助　教

李 伦 升　　**lunsheng23@mail.ustc.edu.cn**

龙 万 青　　**madoren@mail.ustc.edu.cn**

孟　　琪　　**mengqi7788@mail.ustc.edu.cn**

陈 雨 昂　　**yuangchen21@mail.ustc.edu.cn**

注意事项

1) 本课程一共包含 4 个实验，分五周完成。
2) 实验报告通过邮箱进行提交，具体提交方式见实验准备（4）。报告提交截止时间为每周三，迟交会有记录。
3) 第四次实验时间为两周，可以提前完成并提交实验报告。
4) 四次实验内容是彼此独立的，单次实验完成之后请务必点击结束实验按钮释放资源。每次实验直接启动建立好的实验，不需要重复创建。
5) 本课程实验基于未来网实验设施平台进行，如实验过程中遇到问题，请及时通过课程群与助教联系。

实验准备

本次实验基于未来网络实验设施平台 http://ceni.ustc.edu.cn 进行，请通过校园网登录该网站进行实验。关闭网页不会影响实验数据。

一、 登录

在浏览器输入地址 http://ceni.ustc.edu.cn 登录实验平台。

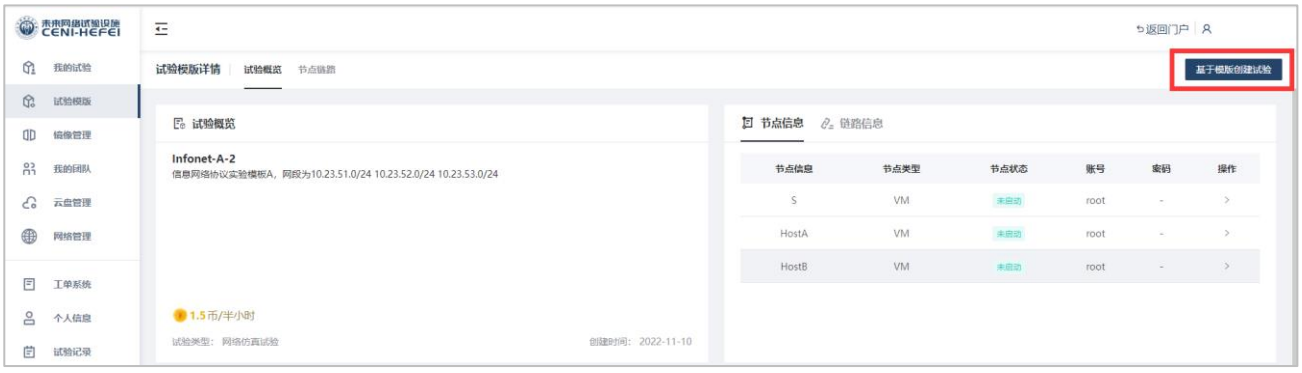
二、 实验创建

由于 ipv4 地址同一网段地址不足够支持所有同学进行实验，现在提供了两套网段，其对应关系如下

主机名	HostA/B	S	S HostA/B
网段 1	10.23.51.0/24	10.23.52.0/24	10.23.53.0/24
网段 2	10.23.54.0/24	10.23.55.0/24	10.23.56.0/24

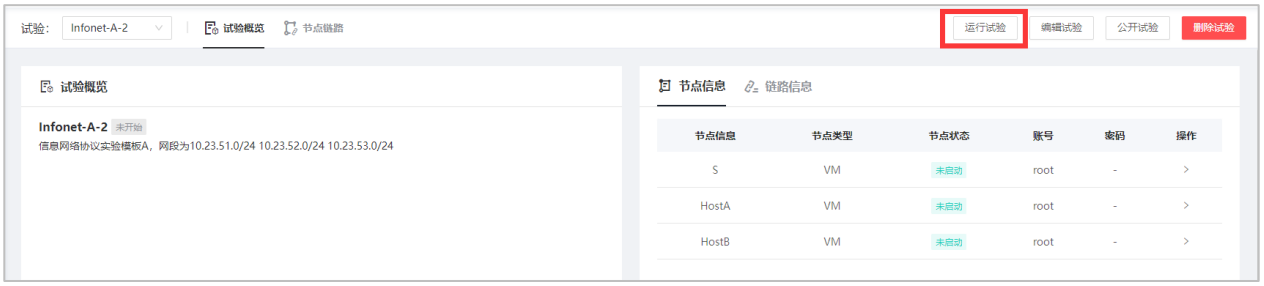
请**学号最后一位数字**为**奇数**的同学选择第一个网段，**偶数**的同学选择第二个网段进行实验。

直接使用实验模板创建实验，网段1实验模板为Infonet-A-2，网段2模板为InfonetB-2。

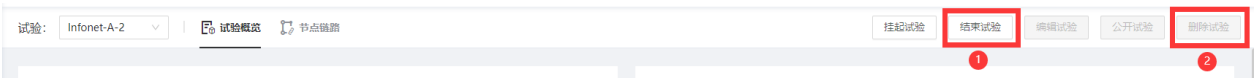


三、 实验运行

点击刚刚新创建的实验，点击运行（此过程可能需要等待一段时间），创建完成之后打开控制台，即可开始实验。请选择以**ustc账号**进行登录，其登录密码为**ustc1958**



注意：四次实验内容是彼此独立的，单次实验完成之后请务必点击**结束实验**按钮释放资源。下次实验直接启动就好，不需要重复创建实验



四、 结果提交

实验报告统一提交至邮箱: ustc_infonet@163.com ， 标题和文件命名规则为：**姓名-学号-信网第x次实验**。前三次实验请提交PDF文件，第四次将报告和代码打包为一个压缩包。

实验四：IPv6 网络编程实验——自由设计

一、 实验目标

1. 了解多进程编程
2. 掌握基于 socket 的全双工通信
3. 掌握基于 socket 的文件传输

二、 实验原理

1. 多进程编程

进程 (process) 是操作系统的概念，为程序的一次执行在操作系统中或内存中的映像，同时伴随着资源的分配和释放。并具有以下特征的活动单元。

- 一组执行的指令序列
- 一个当前状态
- 相关的系统资源集合

Linux 内核通过唯一的进程标识符 PID 来标识每个进程。PID 存放进程描述符的 pid 字段中，新创建的 PID 通常是前一个进程的 PID 加 1，不过 PID 的值有上限。

使用 fork() 函数复制创建新进程：

```
1  //函数头文件及函数原型
2  #include <unistd.h>
3  pid_t fork(void);
```

在 Linux 中创建一个新进程的唯一方法是使用 fork() 函数。fork() 函数是 Linux 中一个非常重要的函数，用于从已存在的进程中创建一个新进程。新进程称为子进程，而原进程称为父进程。使用 fork() 函数得到的子进程是父进程的一个复制品，它从父进程处继承了整个进程的地址空间，包括进程的上下文、代码段、进程堆栈、内存信息、打开的文件描述符、符号控制设定、进程优先级、进程组号、当前工作目录、根目录、资源限制、信号处理方式和控制终端等，而子进程所独有的只有它的进程号、资源使用和计时器等。实际是在父进程中执行 fork() 函数时，父进程会复制一个子进程，而且父子进程的代码从 fork() 函数的返回开始分别在两个地址空间中同时运行，从而使两个进程分别获得所属 fork() 函数的返回值，其中在父进程中的返回值是子进程的进程号，而在子进程中返回 0，若出错返回-1。

2. 全双工通信

双工（Full Duplex）是通讯传输的一个术语。通信允许数据在两个方向上同时传输，它在能力上相当于两个单工通信方式的结合。全双工指可以同时（瞬时）进行信号的双 5.3 实验内容向传输（AB 且 BA）。指 AB 的同时 BA，是瞬时同步的。

全双工方式在发送设备的发送方和接收设备的接收方之间采取点到点的连接，通信系统的每一端都设置了发送器和接收器，因此，能控制数据同时在两个方向上传送。全双工方式无需进行方向的切换，因此，没有切换操作所产生的时间延迟，这对那些不能有时间延误的交互式应用（例如即时聊天、远程监测和控制系统）十分有利。

三、 实验内容

本实验是在第 4 章实验三的基础之上的扩展实验。根据扩展功能的不同，本实验有两个题目，分别为：

- 基于 socket 的全双工通信编程
- 基于 socket 的文件传输编程

同学们可以在对实验三中基于 socket 的基础网络编程的充分理解之上，自行设计 Client/Server 结构通信中的扩展功能并实现。附录 A 中列出了可能会用到的功能的示例代码（C 语言版本）。

(一)题目 1

要求编写一个程序，实现两个主机之间的文本聊天通信。通信需要在 IPv6 环境下完成 (即这两个主机均使用 IPv6 地址)。

人数限制：此题目仅允许 1 个人单独完成。

基本要求：此程序不需要有图形界面，即图形界面不作为评分依据。两个主机之间的聊天由一方作为发起方开始，而后不断进行直到任意一方发送“END”结束。要求聊天过程中两人都可以随时向对方发送文本消息，即需要实现两个主机之间的全双工通信。如果提交的代码只能实现两个主机轮流发送文本消息给对方将会酌情扣分。

扩展要求：此题目扩展要求自拟，比如可以实现“群聊”、“@”、“文件传输”功能。选做扩展要求者根据所选择的扩展要求的实现难度以及完成质量酌情加分。

(二) 题目 2

要求编写一个程序，将一个不小于 20M 的文件（不一定是文本文件）从一个主机发送到另外一个主机，通信需要在 IPv6 环境下完成 (即这两个主机均使用 IPv6 地址)。

人数限制：此题目可以由 1-4 个成员共同完成。

基本要求：此程序不需要有图形界面，即图形界面不作为评分依据。主机 B 收到的由主机 A 发来的文件必须与主机 A 上的文件一致，即发送前后文件的校验值 (CRC32) 相同。需要可以控制文件传输的速度 (即在程序开始运行时询问用户以多大的速度发送文件，而后一直以给定的速率传输)。要求在文件发送开始前接收主机可以选择拒绝或者同意接收文件。

扩展要求：要求在文件发送过程中两个主机都能看到传输的进度，即已传大小、百分比、文件传输速度、预计传输时间等信息，在文件发送过程中接收主机可以选择暂停、继续、终止接收文件。完成此要求者根据所选择的扩展要求的实现难度以及完成质量酌情加分。

四、 实验报告

报告要求：电子版报告，内容包括实验题目、实验原理 (对相关功能的实现方式及程序代码进行必要的解释)、实验结果 (程序运行截图)、实验收获。实验代码及相应的可执行程序需要一并提交。如果采用脚本语言 (比如 Ruby、Python) 则无需提交可执行程序。

注意

- 以上两题**任选一题**完成即可。
- 可以采用你熟悉的**任何编程语言**实现上述要求。
- 提交的代码会经由代码 Review 工具检查，如果检查后发现两组或两个人的代码重合度高于 50% 则算作抄袭，抄袭与被抄袭者本实验记为 0 分。简单修改变量名称与语句顺序不能降低重合度！
- 实验报告、实验代码、可执行程序请打包成 **ZIP 格式**提交。

实验提交截止日期 12 月 9 日 23 时 59 分 59 秒

附录B 网络编程参考资料

说明：以下内容均节选自 [The GNU C Library Reference Manual, Sandra Loosemore et.al]，此手册已放于 Host 虚拟机桌面。

附录 B1. 常用网络编程函数及数据结构

名称	功能	页码
[Function] socket	This function creates a socket and specifies communication style ***style***.	489
[DataType] sockaddr_in6	This is the data type used to represent socket addresses inthe Internet namespace.	474
[Function] listen	The listen function enables the socket ***socket*** to accept connections, thus making it a serversocket.	493
[Function] bind	The bind function assigns an address to the socket ***socket***.	469
[Function] accept	This function is used to accept a connection request on the server socket ***socket***.	494
[Function] connect	The connect function initiates a connection from the socket with file descriptor ***socket*** to the socket whose address is specified by the ***addr*** and ***length*** arguments.	492
[Function] shutdown	The shutdown function shuts down the connection of socket ***socket***.	490

附录 B2. 网络字节顺序及转换函数

Different kinds of computers use different conventions for the ordering of bytes within a word. Some computers put the most significant byte within a word first (this is called "big-endian" order), and others put it last ("little-endian" order). So that machines with different byte order conventions can communicate, the Internet protocols specify a canonical byte order convention for data transmitted over the network. This is known as network byte order. When establishing an Internet socket connection, you must make sure that the data in the **sin_port** and **sin_addr** members of the **sockaddr_in** structure are represented in network byte order. If you are encoding integer data in the messages sent through the socket, you should convert this to network byte order too. If you don't do this, your program may fail when running on or talking to other kinds of machines.[pp.485]

名称	功能	页码
[Function] htons	This function converts the uint16_t integer ***hostshort*** from host byte order to network byte order.	486
[Function] ntohs	This function converts the uint16_t integer ***netshort*** from network byte order to host byte order.	486
[Function] htonl	This function converts the uint32_t integer ***hostlong*** from host byte order to network byte order. This is used for IPv4 Internet addresses.	486
[Function] ntohl	This function converts the uint32_t integer ***netlong*** from network byte order to host byte order. This is used for IPv4 Internet addresses.	486

附录 B3. 常用 IP 地址转换函数

名称	功能	页码
[Function] inet_addr	This function converts the IPv4 Internet host address name from the standard numbers-and-dots notation into binary data.	478
[Function] inet_aton	This function converts the IPv4 Internet host address name from the standard numbers-and-dots notation into binary data and stores it in the struct in_addr that ***addr*** points to.	477
[Function] inet_ntoa	This function converts the IPv4 Internet host address addr to a string in the standard numbers-and-dots notation.	478
[Function] inet_pton	This function converts an Internet address (either IPv4 or IPv6) from presentation (textual) to network (binary) format.	479
[Function] inet_ntop	This function converts an Internet address (either IPv4 or IPv6) from network (binary) to presentation (textual) form.	479

附录 B4. 字节处理函数

名称	功能	页码
[Function] memcpy	The memcpy function copies ***size*** bytes from the object beginning at ***from*** into the object beginning at ***to***.	118
[Function] memset	This function copies the value of ***c*** (converted to an unsigned char) into each of the first ***size*** bytes of the object beginning at ***block***.	121
[Function] memcmp	The function memcmp compares the ***size*** bytes of memory beginning at ***a1*** against the ***size*** bytes of memory beginning at ***a2***.	130
[Function] memmove	memmove copies the ***size*** bytes at ***from*** into the ***size*** bytes at ***to***, even if those two blocks of space overlap.	120
[Function] bzero	This is a partially obsolete alternative for memset, derived from BSD. Note that it is not as general as memset, because the only value it can store is zero.	123
[Function] bcopy	This is a partially obsolete alternative for memmove, derived from BSD. Note that it is not quite equivalent to memmove, because the arguments are not in the same order and there is no return value.	123

附录 B5. 文件描述符操作函数

名称	功能	页码
[Function] open	The open function creates and returns a new file descriptor for the file named by ***filename***.	356
[Function] close	The function close closes the file descriptor ***filedes***.	358
[Function] write	The write function writes up to ***size*** bytes from ***buffer*** to the file with descriptor ***filedes***.	361
[Function] send	The send function is like write, but with the additional flags ***flags***.	495
[Function] recv	The recv function is like read, but with the additional flags ***flags***.	496