
华中科技大学计算机学院

《计算机通信与网络》实验报告

班级_____ 姓名_____ 学号_____

项目	数据可靠传输协议设计 (50%)	CPT 组网 (30%)	平时成绩 (20%)	总分
得分				

教学目标达成情况一览表

实验 \ 目标	目标 1	目标 2	目标 3	目标 4	目标 5	目标 6	目标 7	合计
数据可靠传输 协议设计	/25		/15	/4	/2	/2	/2	/50
CPT 组网	/5	/20		/3	/2			/30
小计	/30	/20	/15	/7	/4	/2	/2	/80

教师评语：

教师签名：

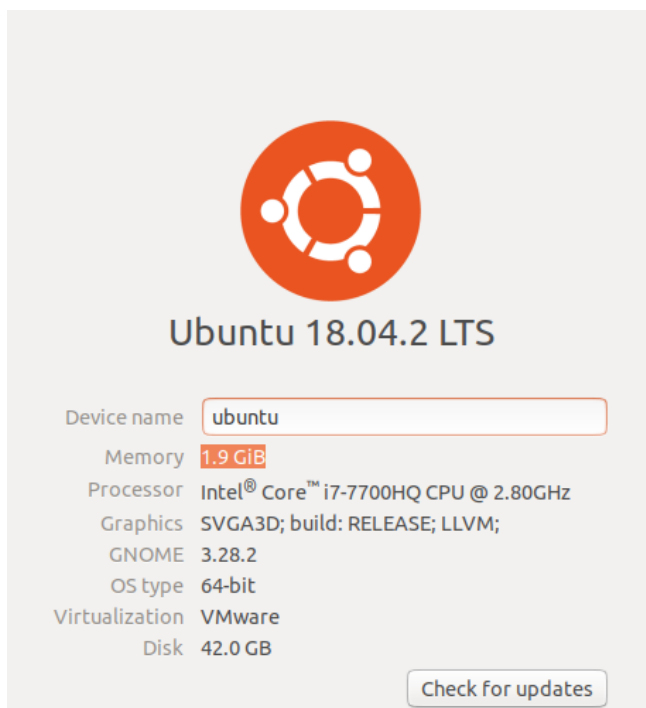
给分日期：

目 录

1. 实验一 数据可靠传输协议设计实验	3
1.1 环境	3
1.2 实验要求	3
1.3 协议的设计、验证及结果分析	3
1.4 教学目标达成情况说明	8
1.5 其它需要说明的问题	9
2. 实验二 基于 CPT 的组网实验	10
2.1 环境	10
2.2 实验要求	10
2.3 基本部分实验步骤说明及结果分析	10
2.4 综合部分实验设计、实验步骤及结果分析	20
2.5 教学目标达成情况说明	24
2.6 其它需要说明的问题	24
3. 心得体会与建议	25
3.1 心得体会	25
3.2 建议	25

1. 实验一 数据可靠传输协议设计实验

1.1 环境



1.2 实验要求

本实验包括三个级别的内容，具体包括：

- (1) 实现基于 GBN 的可靠传输协议，分值为 50%。
- (2) 实现基于 SR 的可靠传输协议，分值为 30%。
- (3) 在实现 GBN 协议的基础上，根据 TCP 的可靠数据传输机制（包括超时后只重传 最早发送且没被确认的报文、快速重传）实现一个简化版的 TCP 协议。报文段格式、 报文段序号编码方式和 GBN 协议一样保持不变，不考虑流量控制、拥塞控制，不需 要估算 RTT 动态调整定时器 Timeout 参数。分值 20%。

1.3 协议的设计、验证及结果分析

1.3.1 GBN 协议的设计、验证及结果分析

Sender 类

```
1.  int base;      //基序号，最早的未确认分组的序号
2.  int nextSeqnum; //下一个待发分组的序号
3.  const int wndsize; //滑动窗口大小，实验建议为 4
4.  const int seqsize; //序号大小，实验建议位数为 3 位，即 0~7
5.  Packet *const sendBuf; //发送缓冲区，保存发送的报文，用于重传
```

Sender 方法

```
bool GBNRdtSender::send(Message & message)
```

1. 将应用层 message 封装为传输层数据包 Packet，Packet 中除包含 message 所有数据外，还包含当前发送的滑动窗口序号、数据包校验和；
2. 由于 GBN 中采取累计确认机制，因此只需要一个计时器，因此当前滑动窗口 base 序号和 nextSeqnum 序号相等时，表示所有接收方接收到了所有的数据包，因此再次发送数据包时应重新启动定时器；
3. 调用 sendToNetworkLayer 发送数据包；
4. 发送完毕，更新滑动窗口状态，即 nextSeqnum 序号循环右移

```
void GBNRdtSender::receive(Packet &ackPkt)
```

1. 通过数据包校验和检查发来的 ack 是否损坏，如果损坏，不做处理；
2. 更新滑动窗口状态，由于采取累计确认机制，接收到 ack 表明 ack 之前序号的所有数据包接收方已经全部接收完毕，因此 base 序号直接循环右移到 ack+1 处，同样，如果此时 base 序号和 nextSeqnum 序号相同，应停止计时器计时，表明这一轮所有数据包接收方已全部接收完毕；
3. 打印滑动窗口变化。

```
void GBNRdtSender::timeoutHandler(int seqNum)
```

1. 由于采取累计确认机制，接收方没有缓存不会保留顺序，因此必须重发所有已经发送且未确认的分组；
2. 重新启动定时器；
3. 打印超时重传调试信息。

```
bool GBNRdtSender::getWaitingState()
```

1. 该函数返回 true 时，表明发送方滑动窗口已满，上层应用不应该再把 message 交由传输层处理；
2. return (base + wndsize) % seqsize == (nextSeqnum) % seqsize;

Receiver 类

1. `int expectedSeqNum;` //下一个待接收的数据包
2. `const int seqsize;` //窗口大小, 实验建议为 8
3. `Packet lastAckPkt;` //保存上一次成功接收对应 `ack` 包

Receiver 方法

`void GBNRdtReceiver::receive(Packet & packet)`

1. 查看数据包 `packet` 校验和是否正确, 数据包损坏则发送上一次 `ack` 数据包;
2. 查看数据包 `packet` 序号是否是自己期待接收的包, 如果不是也即发送上一次 `ack` 数据包;
3. 将传输层数据包 `packet` 解包成应用层 `message`, 并调用 `delivertoAppLayer` 递交 给上层应用;
4. 构造 `ack` 数据包, `ack` 序号为 `packet` 中序号 `seqnum`, 通知发送方;
5. 更新 `expectedSeqnum`, 由于接收方没有缓存机制, 因此 `expectedSeqnum` 加一即可。

验证

```
[SENDER]发送前窗口 [0* 1 2 3] 4 5 6 7
*****模拟网络环境*****：启动定时器, 当前时间 = 0.8925, 定时器报文序号 = 0, 定时器Time
*****模拟网络环境*****：发送方的数据包将在9.4625到达对方, 数据包为-->seqnum = 0, ack
[SENDER]发送后窗口 [0 1* 2 3] 4 5 6 7

*****模拟网络环境*****：向上递交给应用层数据：AAAAAAAAAAAAAAAAAAAAA
[Debug]接收方正确并发送到上层APP：seqnum = 0, acknum = -1, checksum = 29556, AAAAAA
[Debug]接收方发送确认报文：seqnum = -1, acknum = 0, checksum = 12851, .....
*****模拟网络环境*****：接收方的确认包将在15.4425到达对方, 确认包为-->seqnum = -1, a

[RECEIVER]确认号ack：0

*****模拟网络环境*****：关闭定时器, 当前时间 = 15.4425, 定时器报文序号 = 0

[SENDER]收到ack, 滑动窗口移动：0 [1* 2 3 4] 5 6 7
```

图 1-1 滑动窗口变化

```
[Debug]发送超时
*****模拟网络环境*****：启动定时器, 当前时间 = 1165.04, 定时器报文序号 = 0, 定时器Timeout时间 = 1185.04
*****模拟网络环境*****：发送方的数据包将在1168.67到达对方, 数据包为-->seqnum = ZZZZZZZZZZ -1,checksum = 52420,payload = YYYYYYY
[Debug]超时重传的分组：seqnum = 5, acknum = -1, checksum = 52420, YYYYYYY
ZZZZZZZZZZ
*****模拟网络环境*****：发送方的数据包将在1175.43到达对方, 数据包为-->seqnum = AAAAAAAAAA -1,checksum = 17217,payload = ZZZZZZZZ
[Debug]超时重传的分组：seqnum = 6, acknum = -1, checksum = 17217, ZZZZZZZZ
AAAAAAAAAA
*****模拟网络环境*****：发送方的数据包将在1179.48到达对方, 数据包为-->seqnum = BBBB BBBB -1,checksum = 42652,payload = AAAAAAAA
[Debug]超时重传的分组：seqnum = 7, acknum = -1, checksum = 42652, AAAAAAAA
BBBBBBBBBB
*****模拟网络环境*****：发送方的数据包将在1181.69到达对方, 数据包为-->seqnum = CCCCCCCCC -1,checksum = 39582,payload = BBBB BBBB
[Debug]超时重传的分组：seqnum = 0, acknum = -1, checksum = 39582, BBBB BBBB
CCCCCCCC
[Debug]重发数据包完毕
```

图 1-2 超时重传

1.3.2 SR 协议的设计、验证及结果分析

Sender 数据结构

```
1.  const int seqsize; //发送序号大小
2.  const int wndsize; //发送窗口大小
3.  Packet *const sendBuf; //发送缓冲区，避免反复构造析构
4.  bool *const bufStatus; //数据包确认状态
5.  int base, nextSeqnum; //窗口标志序号
```

Sender 类

boolSRRdtSender::send(Message&message)

1. 将应用层 message 封装为传输层数据包 Packet，Packet 中除包含 message 所有数据外，还包含当前发送的滑动窗口序号、数据包校验和；
2. 由于 SR 中采取选择重传机制，因此需要为每一个发送的数据包设置一个计时器；
3. 发送完毕，更新滑动窗口状态

cvoidSRRdtSender::receive(Packet&ackPkt)

1. 通过数据包校验和检查发来的 ack 是否损坏，如果损坏，不做处理；
2. 更新滑动窗口状态，由于采取选择重传机制，因此可以将 base 序号循环右移到最后一个未确认的数据包；
3. 打印滑动窗口变化。

dvoidSRRdtSender::timeoutHandler(intseqnum)

由于采取选择重传机制，接收方设置有缓存，因此只需将超时的数据包 seqnum 重传即可。

boolSRRdtSender::getWaitingState()

1. 该函数返回 true 时，表明发送方滑动窗口已满，上层应用不应该再把 message 交由传输层处理；
2. 因此 return(base+wndsize)%seqsize==(nextSeqnum)%seqsize;即可。

Receiver 数据结构

```
1.  const int seqsize; //发送序号大小
2.  const int wndsize; //发送窗口大小
3.  Packet lastAckPkt; //上一个接收成功的 ack 包
4.  Packet *const recvBuf; //接收缓存
5.  bool *const bufStatus; //数据包接受状态
6.  int base;
```

receiver 方法

void SRRdtReceiver::receive(Packet&packet)

1. 查看数据包 packet 校验和是否正确，数据包损坏则发送上一次 ack 数据包；
2. 查看数据包 packet 序号是否是自己期待接收的包，如果不是也即发送上一次 ack 数据包，判断是否为期待接收的数据包即判断数据包序号是否在接收方滑动窗口内；
3. 将接收缓存 recvBuf 里可用的数据包全部解包为应用层 message，并调用 deliverToAppLayer 递交给上层应用；
4. 构造 ack 数据包，ack 序号为 packet 中序号 seqnum，通知发送方；

验证

```
[SENDER]Window before sending[Available Available Available Available ]Unavailable Unavailable Unavailable Unavailable
*****模拟网络环境*****：发送方的数据包将在7.8到达对方，数据包为-->seqnum = 0, acknum = -1,checksum = 29556,payload = AAAAAAAAAAAAAAAAAAAAAA
*****模拟网络环境*****：启动定时器，当前时间 = 1.18, 定时器报文序号 = 0, 定时器Timeout时间 = 21.18
[SENDER]Window after sending[Sended Available Available Available ]Unavailable Unavailable Unavailable Unavailable

[Debug]Getting ACK: seqnum = 0, acknum = 0, checksum = 12852, .....
*****模拟网络环境*****：接收方的确认包将在11.03到达对方，确认包为-->seqnum = 0, acknum = 0,checksum = 12852,payload = .....
*****模拟网络环境*****：向上递交给应用层数据：AAAAAAAAAAAAAAAAAAAAA
[Debug]Delivering to the upper layer:: seqnum = 0, acknum = -1, checksum = 29556, AAAAAAAAAAAAAAAAAAAAAA

[RECEIVER]Receive and Move window:Unavailable [Expected Expected Expected Expected ]Unavailable Unavailable Unavailable
[Debug]ACK checksum error: seqnum = 0, acknum = 0, checksum = 12852, .....
```

图 1-3 窗口滑动

```
[Debug]Package Time out:4
*****模拟网络环境*****：发送方的数据包将在537.473到达对方，数据包为-->seqnum =04, acknum = -1,checksum = 64507,payload = TTTTTTTTTTTTTTTTTT
*****模拟网络环境*****：启动定时器，当前时间 = 509.378, 定时器报文序号 = 4, 定时器Timeout时间 = 529.378
[Debug]Finish resending:4
```

图 1-4 超时重传

1.3.3 简单 TCP/IP 协议的设计、验证及结果分析 (Optional)

TCP Sender(Based on GBN)

数据结构

相比 GBN，增加 dupAckNum 检测冗余情况：

1. `int base;` //基序号，最早的未确认分组的序号
2. `int nextSeqnum;` //下一个待发分组的序号
3. `const int wndsize;` //滑动窗口大小= 4
4. `const int seqsize;` //序号大小，实验建议位数为 3 位，即 0~7
5. `Packet *const sendBuf;` //发送缓冲区，保存发送的报文，用于重传，大小应该是 seqsize
6. `int dupAckNum;` //收到 3 个冗余 ack 快速重传

1. boolTcpRdtSender::send(Message&message)

和 GBN::Sender 方法完全相同，不再赘述。

2. voidTcpRdtSender::receive(Packet&ackPkt)

和 GBN 稍有不同的是，TCP 接收到窗口外 ack 即冗余 ack 时，判断冗余 ack 是否连续到了三个，若是如此，表明当前网络环境不好，数据包传输过程中会超时，这个时候可以手动重传未确认的包而不需要等到数据包超时再重传。

3. voidTcpRdtSender::timeoutHandler(intseqNum)

TCP 采用快速重传，配合 receiver 方法，数据包超时时只重传最早未确认的包（即 base）即可。

4. boolTcpRdtSender::getWaitingState()

和 GBN 完全相同。

验证

```
*****模拟网络环境*****：发送方发送的数据包丢失：seqnum = 3, acknum = -1,checksum = C
*****模拟网络环境*****：关闭定时器，当前时间 = 84.7225, 定时器报文序号 = 0
*****模拟网络环境*****：启动定时器，当前时间 = 84.7225, 定时器报文序号 = 0, 定时器Timeout时间 = 104.722
[Debug]重发数据包完毕
```

图 1-5 超时重传,只重传最早的未确认的包

```
*****模拟网络环境*****：接收方的确认包将在185.875到达对方，确认包为-->seqnum = -1, acknum = 5,checksum = 12846,payload = .....
[Debug]不是期望的数据分组：seqnum = 7, acknum = -1, checksum = 28260, GGGGG
*****模拟网络环境*****：接收方的确认包将在187.435到达对方，确认包为-->seqnum = -1, acknum = 5,checksum = 12846,payload = .....
*****模拟网络环境*****：发送方的数据包将在193.895到达对方，数据包为-->seqnum = GGGGGGGGGGGGGGGG,checksum = 29553,payload = FFFF
[SENDER]收到连续三个冗余ack，快速重传
```

图 1-6 三个冗余 ACK 的快速重传

1.4 教学目标达成情况说明

1.4.1 目标 1 达成情况

【目标说明】能够基于计算机科学原理并采用科学方法对计算机复杂工程问题能进行问题抽象、制定实验方案、搭建计算机软硬件实验环境、正确采集和整理实验数据,进行实验验证。

【达成情况】这个可以有

1.4.2 目标 3 达成情况

【目标说明】能利用集成开发环境、开源及第三方资源进行计算机平台与工具的开发、调试与测试。

【达成情况】应该有

1.4.3 目标 4 达成情况

【目标说明】理解计算机相关领域工程实践和复杂工程解决方案设计中应承担的社会责任,具有主动采取应对措施减少不良影响意识和行动。

【达成情况】有吧

1.4.4 目标 5 达成情况

【目标说明】正确理解和评价解决计算机复杂工程问题的专业实践对客观世界和社会可持续发展的影响;理解用技术手段降低其负面影响的基本方法、作用及其局限性。

【达成情况】有

1.4.5 目标 6 达成情况

【目标说明】掌握计算机工程项目生命周期各过程管理的基本内容、方法和技术。

【达成情况】以后慢慢培养

1.4.6 目标 7 达成情况

【目标说明】能将管理原理、经济方法应用于计算机工程项目的开发、设计和流程优化等过程中。

【达成情况】有一点

1.5 其它需要说明的问题

暂无

2. 实验二 基于 CPT 的组网实验

2.1 环境

- Windows
- CPT

2.2 实验要求

熟悉 Cisco Packet Tracer 仿真软件，利用 Cisco Packet Tracer 仿真软件完成实验内容

2.3 基本部分实验步骤说明及结果分析

2.3.1 IP 地址规划与 Vlan 分配实验的步骤及结果分析

按照要求连接好模拟机器。

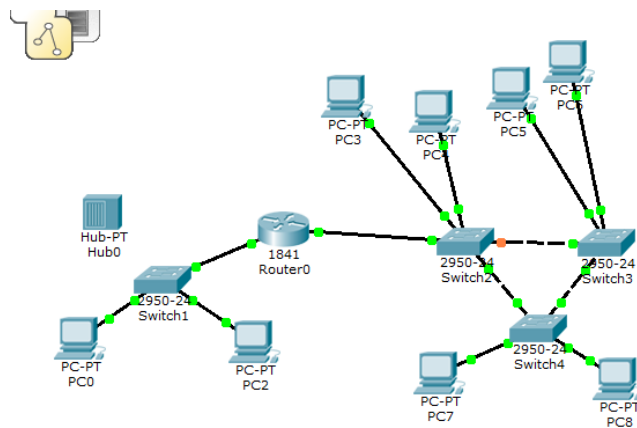


图 2-1

以配置 Route0 为例配置路由:

IP 配置为 192.168.0.2
子网掩码为 255.255.255.0
网关配置:

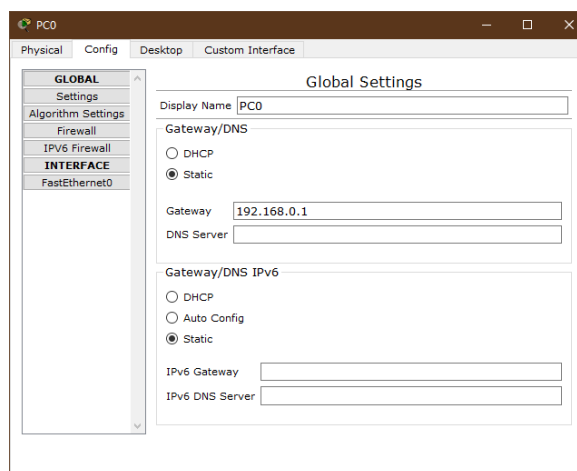


图 2-5

网关配置为上层路由地址 192.168.0.1
配置完成后尝试连接 PC0 和 PC8

PC>ping 192.168.1.7

Pinging 192.168.1.7 with 32 bytes of data:

Reply from 192.168.1.7: bytes=32 time=0ms TTL=127
Reply from 192.168.1.7: bytes=32 time=0ms TTL=127
Reply from 192.168.1.7: bytes=32 time=0ms TTL=127
Reply from 192.168.1.7: bytes=32 time=0ms TTL=127

Ping statistics for 192.168.1.7:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>ping 192.168.0.2

Pinging 192.168.0.2 with 32 bytes of data:

Reply from 192.168.0.2: bytes=32 time=0ms TTL=127
Reply from 192.168.0.2: bytes=32 time=0ms TTL=127
Reply from 192.168.0.2: bytes=32 time=1ms TTL=127
Reply from 192.168.0.2: bytes=32 time=2ms TTL=127

Ping statistics for 192.168.0.2:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum = 0ms, Maximum = 2ms, Average = 0ms

除了可以使用命令提示符进行连通性测试以外，还可以使用图形界面进行测试，其效果与命令提示符相同，在今后的测试中，都将使用图形界面。在 packet tracer 界面的右侧单击图标，在需要通信的两台 pc 机上分别点击一次，就代表将信息从 pc1 发向 pc2,通信是否成功可以在界面下方的 PDU 列表窗口查看。如图 3-9, pc1 向 pc2 发送消息，然后 pc2 向 pc1 发送消息，两次连接的状态均为“成功”。

	Successful	PC0	PC6	ICMP		0.000	N	0	(edit)	(delete)
	Successful	PC2	PC8	ICMP		0.000	N	1	(edit)	(delete)
	Successful	PC2	PC3	ICMP		0.000	N	2	(edit)	(delete)
	Successful	PC0	PC2	ICMP		0.000	N	3	(edit)	(delete)
	Successful	PC0	PC2	ICMP		0.000	N	4	(edit)	(delete)

图 2-6

配置 IP 后，两个网段内部都可以进行通信，第二次配置 IP 以后，三个网段内部的内部也可以分别进行通信，符合实验要求。网段每个网段内部的 pc 机之间都是通过交换机连接起来的，内部通信不需要经过路由器就可以直接发送至对方，所以不管有没有给路由器端口分配 IP 地址、有没有给 PC 机配置网关，都可以进行内部通信。

内容 2

完成基本内容 1 以后，要求将 pc4、pc6、pc8 从子网 192.168.1.0 网段中分离出去，编入子网 192.168.2.0，所以需要对这三台 pc 重新配置 IP。

- 不需要删除网关!!!

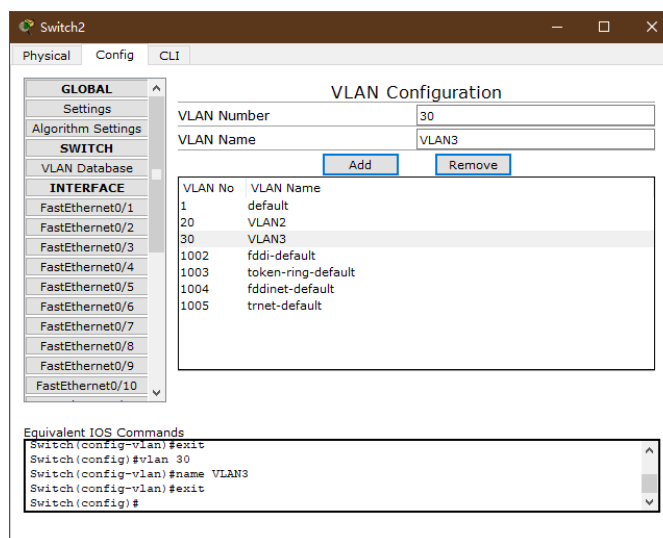


图 2-7

- 同样配置 6,8
- 交换机的 vlan 可用图形界面配置。题目要求将右侧的 6 台 pc 机划分到两个 vlan 中，所以需要在所有的路由器上都添加两个 vlan，vlan 号可以任意选择,这里选择 20 和 30 作为 vlan 号，两个 vlan 名分别为 vlan2 和 valn3。以交换机 2 为例，选择在：配置->交换配置->vlan 数据库，添加两个 vlan.

并修改接口。对于交换机与交换机相连，或者交换机与路由器相连的链路，都称为主干链路，选中该链路对应的端口，选择链路类型为“trunk”，trunk 链路的 vlan 默认包含了所有的 vlan，不需要修改。

- 配置 PC3,5,7 连接的交换机接口为 Access,VLAN2
- 配置 PC4,6,8 连接的交换机接口为 Access,VLAN3
- 对内部主机进行测试:











	Successful	PC1	PC2	ICMP		0.000	N	0	(edit) (delete)
	Successful	PC3	PC5	ICMP		0.000	N	1	(edit) (delete)
	Successful	PC3	PC7	ICMP		0.000	N	2	(edit) (delete)
	Successful	PC4	PC6	ICMP		0.000	N	3	(edit) (delete)
	Successful	PC4	PC8	ICMP		0.000	N	4	(edit) (delete)

图 2-8

- 对域间主机进行测试:结果如图:







	Failed	PC1	PC7	ICMP		0.000	N	0	(edit) (delete)
	Failed	PC3	PC4	ICMP		0.000	N	1	(edit) (delete)
	Failed	PC5	PC6	ICMP		0.000	N	2	(edit) (delete)

图 2-9

- 路由器 vlan 配置:由于路由器只有两个端口，fa0/1 端口所管理的网络中有两个网段的 pc 机，所以要为该端口分配子端口，并为子端口分配 vlan，才能使两个 vlan 的主机可以通信。对路由器 A 的快速以太网接口 0/1，创建子接口 fa0/1.1，并将其划分到 vlan2 中，ip 地址设置为 192.168.1.1。这些都只能用命令行配置。

```

1. Router>enable
2. Router#config terminal
3. Enter configuration commands, one per line. End with CNTL/Z.
4. Router(config-if)#int fa0/1.1
5. Router(config-subif)#encap dot1q 20
6. Router(config-subif)#ip addr 192.168.1.1 255.255.255.0
7. Router(config-subif)#exit

```

- 备注:在路由器上需要做的:
 1. 删除原有的 0/0,0/1 接口的 IP 信息
 2. 按照上述命令配置:
 1. 0/1.1:dot1q 20,192.168.1.1
 2. 0/1.2:dot1q 30 192.168.2.1
 3. 0/0.1:dot1q 10 192.168.0.1
- 测试连通性:证明域间网路联通.

Successful	PC1	PC4	ICMP	0.000	N	0	(edit) (delete)
Successful	PC1	PC6	ICMP	0.000	N	1	(edit) (delete)
Successful	PC2	PC3	ICMP	0.000	N	2	(edit) (delete)
Successful	PC2	PC7	ICMP	0.000	N	3	(edit) (delete)
Successful	PC2	PC6	ICMP	0.000	N	4	(edit) (delete)
Successful	PC4	PC7	ICMP	0.000	N	5	(edit) (delete)

图 2-10

- 在交换机上划分 vlan 以后，各个主机被分入不同的 vlan，一个 vlan 是一个逻辑上的整体，他们不受空间和子网的限制，所以 pc1、pc2 可以跟 pc3 处于同一个 vlan 中。若不在路由器上进行 vlan 配置，一个 vlan 内部的主机可以进行通信，但是他们不能和其他 vlan 的主机通信，相当于这个 vlan 是与世隔绝的。只有在路由器上进行 vlan 配置，一个 vlan 的主机向其他 vlan 主机发送的消息才能被路由器识别并转发，否则，路由器不清楚消息该发向何处，因为路由器本地没有相关 vlan 可以选择。
- 目前路由的配置情况：

Port	Link	VLAN	IP Address	IPv6 Address	MAC Address
FastEthernet0/0	Up	--	<not set>	<not set>	0060.3E47.D301
FastEthernet0/0.1	Up	--	192.168.0.1/24	<not set>	0060.3E47.D301
FastEthernet0/1	Up	--	<not set>	<not set>	0060.3E47.D302
FastEthernet0/1.1	Up	--	192.168.1.1/24	<not set>	0060.3E47.D302
FastEthernet0/1.2	Up	--	192.168.2.1/24	<not set>	0060.3E47.D302
Vlan1	Down	1	<not set>	<not set>	00D0.BC38.8C23
Hostname: Router					
Physical Location: 城际, 城市家园, 公司办公室, Wiring Closet					

图 2-11

2.3.2 路由配置实验的步骤及结果分析

绘制网络拓扑图

根据给定的拓扑图，在 cpt 中绘制等效拓扑图，如。图中有 4 台 pc 机，3 台交换机和 4 台路由器。注意使用 Serial DTE 线之前要在路由器上加装 HWIC-2T 模块。

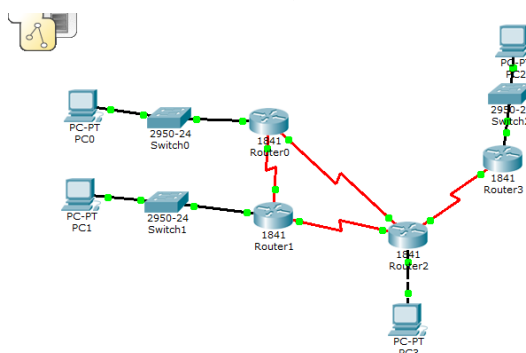


图 2-12

配置路由器 IP

图中有四个路由器，每个路由器管理一个子网，四个子网分别处于 192.168.1.0、192.168.2.0、192.168.3.0、

192.168.4.0 网段，所以路由器与 pc 机或者交换机连接的端口 IP 应该配置为对应网段的 IP。路由器 A 的左端口 IP 为 192.168.1.1，路由器 B 的左端口 IP 为 192.168.2.1，路由器 C 的下方端口 IP 为 192.168.3.1，路由器 D 的上方端口 IP 为 192.168.4.1，图 3-27 所示为路由器 A 的 fa 端口 IP 配置，其他三个路由器与之类似。

配置的 serial 端口。这些端口的 IP 选择比较自由，只要不与 PC 机所处网段的 IP 重叠即可，我们不妨选择 192.168.5.0、192.168.6.0、192.168.7.0、192.168.8.0 四个子网为路由器之间的四条链路进行 IP 分配，链路两端的 IP 分别设为 xxx.xxx.xxx.1 和 xxx.xxx.xxx.2，比如路由器 A 的 serial0/1/0 端口 IP 为 192.168.5.1，与之相连的路由器 B 的 serial0/1/0 端口 IP 为 192.168.5.2，其他端口 IP 可以类推得到。

路由的 DCE/DTE 配置

路由器 DCE、DTE 端口配置 路由器之间连接时，如果选用自动类型的连线，软件会报错，无法成功连接。只能选用 DCE 串口线，这种线是有方向的，先连的一端为 DCE，后连的一端为 DTE，在 DCE 一端需要设置时钟频率，DTE 一端则不需要修改，自动设置为默认值。以路由器 A 为例，其 serial0/1/0 串口为 DCE 端，选择“配置”选项，将该端口的时钟频率选为 64000。

路由器 RIP 协议配置

RIP 协议是用于自治系统内的动态路由协议，它只和与自己相连的路由器交换信息，所以每个路由器配置该协议时，只需要把自己每个端口的 IP 地址所在网段填上即可。以路由器 A 为例，它的 fa 端口 IP 为 192.168.1.1，两个 serial 端口 IP 分别为 192.168.5.1 和 192.168.6.1，所以应该配置这三个网段。选择：配置->路由配置->RIP，在方框中填入三个 IP 地址对应的网段，点击添加。该路由器 RIP 协议配置成功。

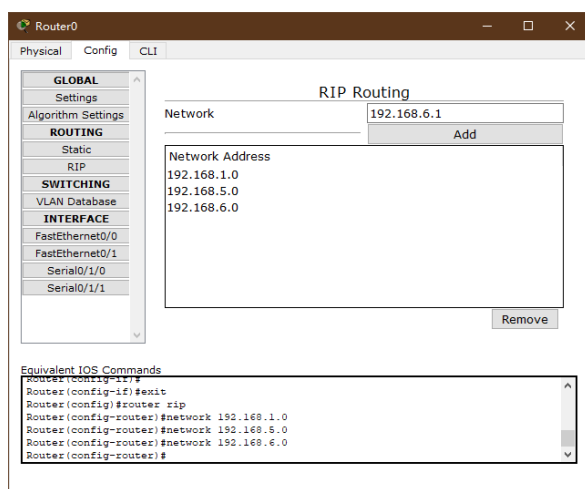


图 2-13

配置 PC 机 IP 地址和网关

Pc1-pc4 分别处于 192.168.1.0、192.168.2.0、192.168.3.0、192.168.4.0 网段，可将其 IP 地址分别设置为 192.168.1.2、192.168.2.2、192.168.3.2、192.168.4.2。

每台 pc 的网关都应该是与它距离最近的路由器的 IP 端口的地址，设为 192.168.1.1、192.168.2.1、192.168.3.1、192.168.4.1，这分别是路由器 A、B、C、D 的 fa 端口地址

访问测试

任选两台 PC 机相互通信，均可进行访问,符合实验预期。







	Successful	PC0	PC2	ICMP		0.000	N	0	(edit)	(delete)
	Successful	PC1	PC3	ICMP		0.000	N	1	(edit)	(delete)
	Successful	PC3	PC2	ICMP		0.000	N	2	(edit)	(delete)

图 2-14

OSPF 改 RIP

除配置路由 OSPF 协议与 RIP 协议不同外，其他配置均与 RIP 协议相同。OSPF 协议是区别于 RIP 协议的另一种选路协议，同样可以用于以太网的通信，我们采用命令行来为路由器配置 OSPF 协议。以路由器 A 为例，任选一个数字作为进程号，为路由器配置 OSPF，network 开头的命令将路由器端口的 IP 地址和子网掩码绑定到路由器上。该路由器有三个端口，所以使用了三条这样的语句。配置完成后，用 copyrunstartup 语句建立配置。

其他三个路由器的配置与之相似，只需要改变进程号和端口 IP。

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router ospf 1
Router(config-router)#network 192.168.1.0 0.0.0.255 area 0
Router(config-router)#network 192.168.5.0 0.0.0.255 area 0
Router(config-router)#network 192.168.6.0 0.0.0.255 area 0
Router(config-router)#end
Router#
%SYS-5-CONFIG_I: Configured from console by console
Router#copy run startup
Destination filename [startup-config]?
Building configuration...
[OK]
```

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router ospf 1
Router(config-router)#network 192.168.2.0 0.0.0.255 area 0
Router(config-router)#network 192.168.5.0 0.0.0.255 area 0
Router(config-router)#network 192.168.7.0 0.0.0.255 area 0

Router(config-router)#end
Router#
%SYS-5-CONFIG_I: Configured from console by console
Router#copy run startup
Destination filename [startup-config]?
Building configuration...
```

[OK]

enable

Router#config terminal

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#router ospf 1

Router(config-router)#network 192.168.3.0 0.0.0.255 area 0

Router(config-router)#network 192.168.6.0 0.0.0.255 area 0

Router(config-router)#network 192.168.7.0 0.0.0.255 area 0

Router(config-router)#network 192.168.7.0 0.0.0.255 area 0

00:14:37: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.7.1 on Serial0/1/0 from LOADING to FULL, Loading Done

00:14:38: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.6.1 on Serial0/0/1 from LOADING to FULL, Loading Done

Router(config-router)#network 192.168.8.0 0.0.0.255 area 0

Router(config-router)#end

Router#

%SYS-5-CONFIG_I: Configured from console by console

Router#copy run startup

Destination filename [startup-config]?

Building configuration...

[OK]

Router>enable

Router#configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#router ospf 1

Router(config-router)#network 192.168.4.0 0.0.0.255 area 0

Router(config-router)#network 192.168.8.0 0.0.0.255 area 0

Router(config-router)#end

Router#

%SYS-5-CONFIG_I: Configured from console by console

00:18:36: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.8.2 on Serial0/1/0 from LOADING to FULL, Loading Done

copy run startup

Destination filename [startup-config]?

Building configuration...

[OK]

联通测试

pc1 对 pc2, pc2 对 pc3, pc4 对 pc2, pc3 对 pc1 都可以进行访问, 所以我们可以认为, 网络中的任意两台 pc 都可以互相访问, OSPF 协议配置成功.







	Successful	PC1	PC0	ICMP		0.000	N	0	(edit) (delete)
	Successful	PC1	PC2	ICMP		0.000	N	1	(edit) (delete)
	Successful	PC1	PC3	ICMP		0.000	N	2	(edit) (delete)

图 2-15

访问控制

用命令行对路由器 A 进行配置。要使得 pc0 无法访问其他网段，而且不能被其他网段访问，应该在路由器 A 与交换机 0 相连的端口进行配置。先用 access-list 命令创建访问控制列表，可选用 100 以内的数字作为 acl 编号，deny 表示屏蔽某网段的消息，permit 表示接受某网段的消息，这里使用了 deny，因为我们要屏蔽 pc1 的通信。命令中的 ip 地址是要屏蔽或接受的网段，最后一个参数是子网掩码的反码。创建 acl 完成，打开端口 fa0/0，用 access-group 命令把 acl 绑定到路由器上，acl 就配置完成了。

Router0:

```
Router(config)#access-list 1 deny 192.168.1.0 0.0.0.255
```

```
Router(config)#int fa 0/0
```

```
Router(config-if)#ip access-group 1 out
```

```
Router(config-if)#exit
```

测试

PC0 不能访问其他终端.





	Failed	PC0	PC1	ICMP		0.000	N	0	(edit) (delete)
	Failed	PC0	PC3	ICMP		0.000	N	1	(edit) (delete)

图 2-16

模拟原因触发了 deny 列表.

1. The receiving port has an inbound traffic access-list with an ID of 1. The router checks the packet against the access-list.
2. The packet matches the criteria of the following statement: deny 192.168.1.0 0.0.0.255. The packet is denied and dropped.

图 2-17

ACL 配置

Pc0 不能访问 pc1，但能访问其他 pc。第一次配置 acl 使只使用了 deny 指令，所以只能屏蔽作用，这里既需要屏蔽一部分消息，又需要允许一部分消息通过，所以还要使用 permit 指令。

仍然对 Router0 进行配置，使用 6 作为 acl 号，当然也可以使用其他数字。首先 deny 来自 pc1，也就是 192.168.2.0 网段的信息，acl 列表就创建好了，然后绑定到端口 fa0/0 上。

```
Router(config)#access-list 6 deny 192.168.2.0 0.0.0.255
```

```
Router(config)#int fa 0/0
```

```
Router(config-if)#ip access-group 6 in
```

```
Router(config-if)#exit
```

测试

Failed	PC1	PC0	ICMP	0.000	N	0	(edit) (delete)
Successful	PC0	PC3	ICMP	0.000	N	1	(edit) (delete)
Failed	PC0	PC1	ICMP	0.000	N	2	(edit) (delete)
Failed	PC1	PC0	ICMP	0.000	N	3	(edit) (delete)
Failed	PC0	PC1	ICMP	0.000	N	4	(edit) (delete)
Successful	PC0	PC2	ICMP	0.000	N	5	(edit) (delete)

图 2-18

2.4 综合部分实验设计、实验步骤及结果分析

2.4.1 实验设计

1. 一个宿舍 200 台,使用 8 位子网编码.学校可以使用 10 位子网码,211.69.4/24、211.69.5/24、211.69.6/24,所以限定高位 00,01,10 为三个宿舍,11 用作其他用途.
2. 图书馆 100 台,需要 7 位,设计为 211.69.7.0/25 段
3. 三个学院一个 20 台,分别设计为 211.69.7.0b(100 00000), 211.69.7.0b(101 00000), 211.69.7.0b(110 00000). 211.69.7.0b(111 00000),不使用.-> 211.69.7.128/27, 211.69.7.160/27,211.69.7.192/27

表格 1

	Static IP Address Sgement						
	24~31	16~23	9~15	8~9	7	5~6	0~4
Dormitory	211	69	1	0b00			
	211	69	1	0b01			
	211	69	1	0b10			
Library	211	69	1	0b11	0		
College	211	69	1	0b11	1	0b00	211.69.7.129
	211	69	1	0b11	1	0b01	211.69.7.161
	211	69	1	0b11	1	0b10	211.69.7.193

从理论上来说,只需要一个路由器就可以解决全校的通信问题,所有的交换机之间也不需要相连,直接往路由器上连接即可。但是充分结合实际,所有宿舍楼、学院都应有自己的路由器,即便没有,也会与临近的宿舍或学院共享一台路由器,绝不会全校使用一台,因为这样更便于管理,更方便维修。

网络拓扑图的大致结构是:图书馆一台路由器、三个学院共用一台路由器、三个宿舍共用一台路由器。每 23 台主机共用一台交换机(因为一台交换机最多只有 24 个接口),一个学院内的所有交换机之间有一定的连接,保证每台交换机都是可达的,最终通过一台交换机与路由器相连。图书馆要求无线上网,所以既有无线上网的笔记本,也有有线上网的 pc 机,此外还有一台无线路由器。宿舍和学院尝试了两种布局方式,即单点布局和树状布局.明显树状布局冗余更多.由于 STP 协议的限制不能产生交换机的环路否则会产生广播洪泛。

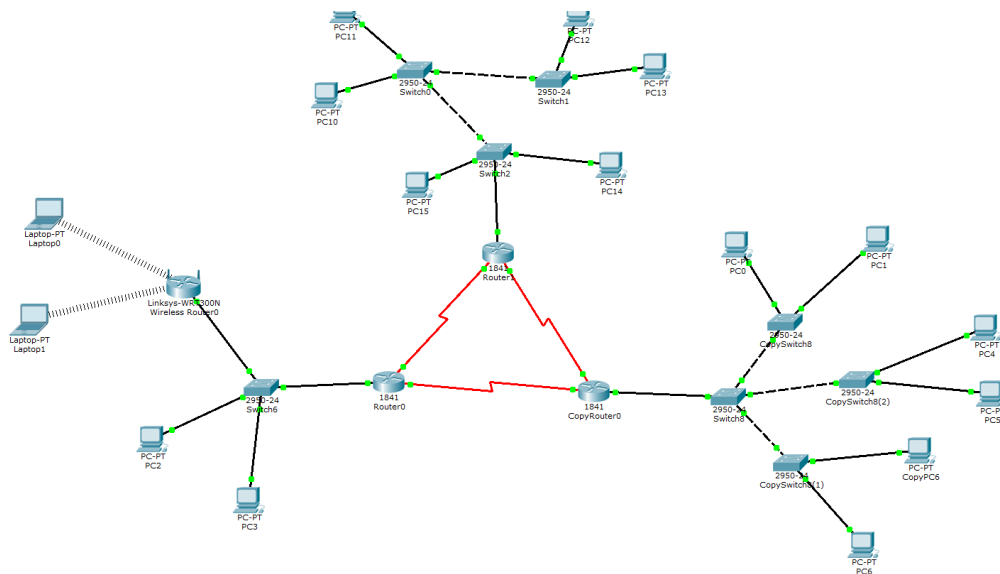


图 2-19

三个学院是相对独立的子网，三个宿舍也是相对独立的子网，然而他们连接在路由器的同一个端口上，所以需要路由器分配子端口。各学院之间要能够通信，所以还需要划分成功三个 vlan，同理三个宿舍也要划分 vlan。学院和宿舍要跨过路由器访问图书馆，需要在路由器上配置 RIP 协议或者 OSPF 协议进行选路。要禁止学院和宿舍互相访问，需要进行 ACL 配置。

2.4.2 实验步骤

配置网段

学院的每台交换机都增加三个 vlan，分别编号为 2、3、4，对应三个学院。宿舍的每台交换机也增加三个 vlan，编号 5、6、7，对应三个宿舍。将交换机之间、交换机与路由器之间的链路设为 trunk 链路，交换机与 pc 机之间的链路设为 access 链路，vlan 设为该 pc 机对应的宿舍楼或者学院的 vlan，这样，就把三个宿舍、三个学院分成了 6 个 vlan。图书馆不需要划分 vlan。

路由器间配置

给每个路由器添加两个 serial 接口，用于路由器之间的连接。路由器之间相连的接口 IP 使用 192.168.xxx.xxx 系列，不能使用 211.69.xx.xx 系列，否则会与学校的 pc 机 IP 相冲突。

学院和宿舍的路由器与交换机相连的接口，都分出三个子接口，子接口的 IP 从三个学院、三个宿舍的 IP 地址块中选出一个进行分配。图书馆的有线路由器不需要子接口，其 fa 接口 IP 分配为图书馆的 IP 地址块中的任一个

学院和宿舍的路由器分出的子接口都需要分别划分到对应学院和宿舍的 vlan 中，六个子接口分别对应 6 个 vlan。路由器之间的 DCE 端需要设置时钟频率为 64000。

对学院和宿舍的路由器还需要加装子接口;详见 2.3.1.

无线模块配置

将无线路由器的 IP 地址设置为图书馆所在网段中的一个地址 211.69.7.1，子网掩码为 255.255.255.128。

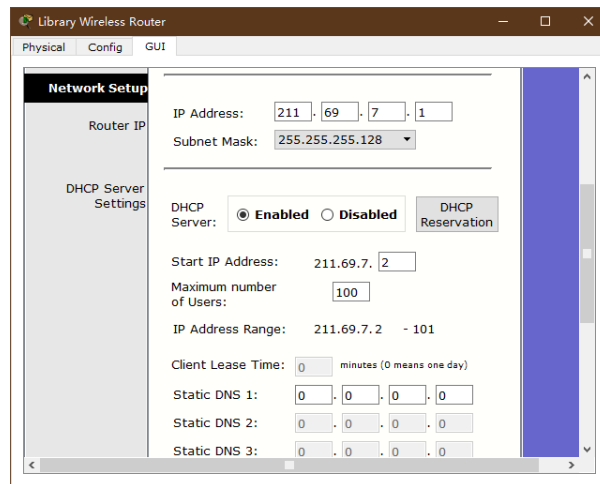


图 2-20

配置 Laptop 终端,无电下加装无线网卡模块.

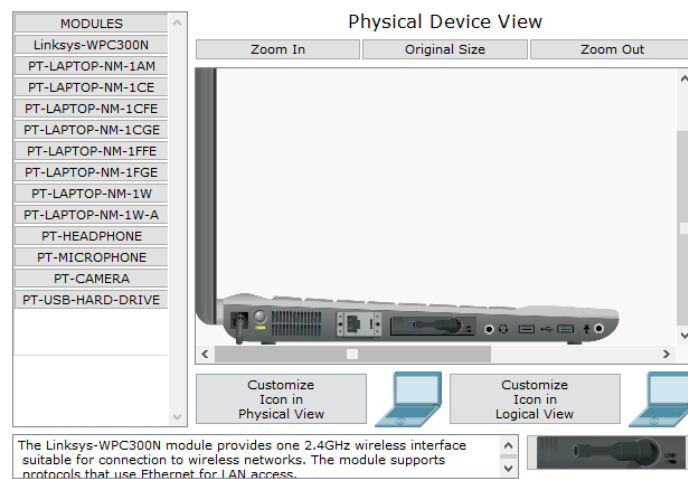


图 2-21

访问控制

对于除了笔记本以外的其他电脑,都将 IP 地址设置为其所在网段的任意值,只要不重复即可。网关设置为距离其最近的路由器端口的 ip 地址。

RIP 协议:在路由器上用图形界面配置 rip 协议,添加路由器所有端口所在的网段 IP 即可。

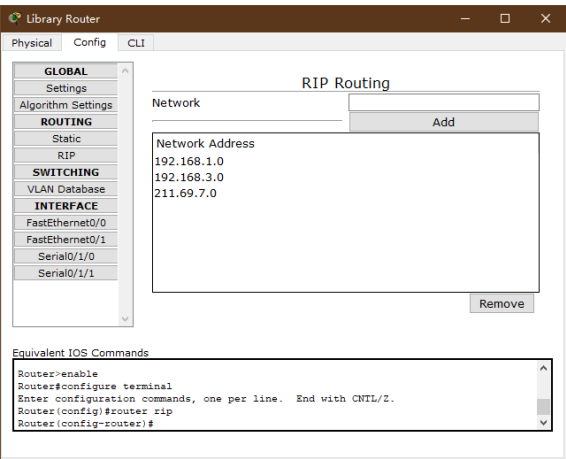


图 2-22

路由器 ACL 配置实验要求学院和宿舍不能互访，但是他们都可以访问图书馆，所以需要在学院和宿舍的路由器之间屏蔽掉对方的信号.操作方法同上个实验.

关键词是 out’!

2. 4. 3 结果分析

测试学校各个部门内部的访问权限。三个学院内部、三个宿舍内部和图书馆内部的主机都能互相访问。

	Successful	PC1	PC5	ICMP		0.000	N	0	(edit) (delete)
	Successful	PC6	PC4	ICMP		0.000	N	1	(edit) (delete)
	Successful	CopyPC6	PC5	ICMP		0.000	N	2	(edit) (delete)

图 2-23 Dom

	Successful	PC11	PC14	ICMP		0.000	N	3	(edit) (delete)
	Successful	PC10	PC13	ICMP		0.000	N	4	(edit) (delete)
	Successful	PC15	PC12	ICMP		0.000	N	5	(edit) (delete)

图 2-24 College

学院和宿舍都能访问图书馆的终端.

	Successful	PC1	PC5	ICMP		0.000	N	0	(edit) (delete)
	Successful	PC2	PC1	ICMP		0.000	N	1	(edit) (delete)
	Successful	PC4	Laptop1	ICMP		0.000	N	2	(edit) (delete)
	Successful	PC11	PC2	ICMP		0.000	N	3	(edit) (delete)

图 2-25

宿舍和学院之间不能相互访问.







	Failed	PC15	PC0	ICMP		0.000	N	0	(edit)	(delete)
	Failed	PC1	PC14	ICMP		0.000	N	1	(edit)	(delete)
	Failed	PC12	PC5	ICMP		0.000	N	2	(edit)	(delete)

图 2-26

根据以上三点测试可知，系统实现的功能达到实验要求的功能，即各部门内部可以互访，全校都可以访问图书馆，而且学院和宿舍不可以互相访问。整个系统综合考察了因特网通信的各个部分，首先要进行合理而且不重叠的子网划分，然后要进行网络拓扑设计。ACL 是保障网络安全的重要工具，它防止外界随意访问某些私密的信息，不然整个因特网都是“透明”的。

2.5 教学目标达成情况说明

2.5.1 目标 1 达成情况

【目标说明】能够基于计算机科学原理并采用科学方法对计算机复杂工程问题能进行问题抽象、制定实验方案、搭建计算机软硬件实验环境、正确采集和整理实验数据, 进行实验验证。

【达成情况】有吧…

2.5.2 目标 2 达成情况

【目标说明】能使用仿真工具对计算机相关理论进行验证，对系统设计方案进行分析和预测。

【达成情况】还行

2.5.3 目标 4 达成情况

【目标说明】理解计算机相关领域工程实践和复杂工程解决方案设计中应承担的社会责任，具有主动采取应对措施减少不良影响意识和行动。

【达成情况】这个必须有

2.5.4 目标 5 达成情况

【目标说明】正确理解和评价解决计算机复杂工程问题的专业实践对客观世界和社会可持续发展的影响;理解用技术手段降低其负面影响的基本方法、作用及其局限性。

【达成情况】可能有…

2.6 其它需要说明的问题

暂无, 见心得部分.

3. 心得体会与建议

3.1 心得体会

可靠数据传输实验相对难一些,但也让我对网络的传输层有了一个更深入的了解,知道传输层应该做些什么,为网络哪一层服务,发送方接收方应该遵守什么样的约定,双方应该怎么处理异常,等等.

第二次实验比较简单,操作却特别繁琐,但是我其实觉得第三次实验是和整个网络课程贴合最紧密的,因为涉及到了实际生产环境的拟真.当然拟真的环境也意味着复杂的设计和频发的故障,这个在综合设计部分尤为突出,调试错误花了大量的时间.很多方法都是提示中没有的,都是自己摸索总结出来的一些经验.

本次实验的相关资料已经开源至 [GitHub](#) 中.

3.2 建议

第一个实验希望能够明确指出要在哪里做什么,刚一拿到文件代码很难懂.
换个靠谱点的模拟软件吧!QAQ