《计算机网络》课程设计内容

计算机网络课程设计含以下 3 个部分，所选题目需要达到 60 分：

（1） 题目一：网络设计（30 分）

（2） 题目二：网络编程（30 分）

（3） 创新题：SDN 网络（60 分）

课程设计采取小组的形式，原则上每组 3-4 人，共同完成设计内容。课程设计报告以小组形式提交，小组成员在课程设计中分担的工作量在报告中体现，按照承担任务的复杂性和工作量的不同，最后评分会不同。

提交报告的格式见文件。

1. 网络设计（30 分，每个题目最多四个小组选择）

网络设计部分说明：所有设计可使用 Cisco Packet Tracer 模拟器进行仿真实现，如果感兴趣的小组可考虑使用 EVE-NG 或 GNS3 模拟器进行仿真实现（加分项）

题目 1：企业访问外部网络（难度：★★★）

1.需求描述

有些企业除了需求访问 Internet 外，还需要访问行业专用网，对于这种企业，存在使用相同的私有 IP 地址访问 Internet 和行业专用网的需求。假定某个企业由两个内部网络组成：一个内部网络连接管理员终端，另外一个内部网络连接员工终端。企业同时连接两个外部网络：一个是 Internet，另一个是行业服务网。Internet 和行业服务网都对该企业分配了全球 IP 地址，但无论是 Internet 还是行业服务网都只负责到达分配给该企业网的 IP 地址的路由功能。

允许所有人员访问 Internet，但只允许管理员访问行业服务网，要求完成该企业网络的设计和配置。

2.分配的 IP 信息

Internet 分配给企业网的网络信息如下：IP 地址为 192.1.1.1，子网掩码为 255.255.255.0，默认网关为 192.1.1.2，域名服务器地址为 192.1.2.253。

行业服务网分配给企业网的网络信息如下：IP 地址为 200.1.1.1，子网掩码为 255.255.255.0，默认网关为 200.1.1.2，域名服务器地址为 200.1.2.253，行业服务网的域名为 b.com。

Internet 的全球 IP 地址范围是任意的，行业服务网的全球 IP 地址范围是 200.1.2.0/24。

参考拓扑结构图如下所示。

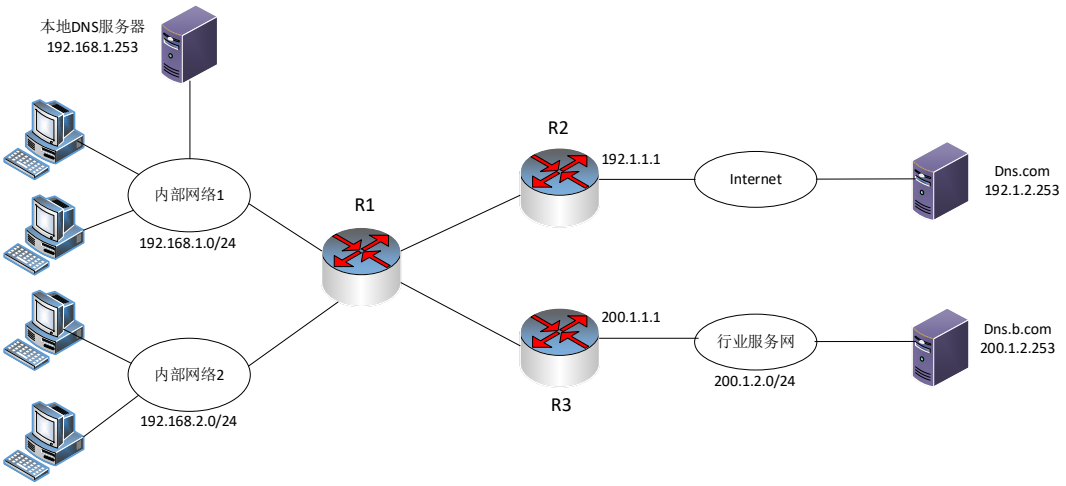

图 1 企业访问外部网络拓扑图

3.详细需求

1）企业网中各 PC 通过 DHCP 自动获取网络信息，由路由器 R1 提供 DHCP 服务器功能。

2）企业网中的域名服务器能够判断某个需要解析的完全合格的域名是 Internet 中服务器的域名，还是行业服务器中服务器的域名。

3）内部网络 1 既可访问 Internet 也可访问行业服务网，而内部网络 2 只能访问 Internet。企业网中的 PC 访问 Internet 或行业服务网时，需要使用 Internet 或行业服务网分配给企业网的 IP 地址（提示：在相应边缘路由器上进行 NAT）。

4）企业网中的 PC 可通过 R1 自动获取网络信息，为了防御 DHCP 欺骗攻击，需要启动相关交换机的防 DHCP 欺骗功能。

5）其它网络安全或可靠性设计。

注意：R1 和 R2 作为企业边界路由器，不应向外部通告自己企业内部的路由信息。

题目 2：远程用户接入企业网（难度：★★★）

1.需求描述

企业有时需要允许远程用户通过 VPN 接入企业网，对企业网中的信息资源进行访问。某企业网有 4 个 VLAN，分别是 VLAN2-VLAN5，其中 VLAN2 属于生产管理部门，VLAN3 属于销售部门，VLAN4 属于财务部门，VLAN5 属于信息服务部门。企业网和 Internet 互连，连接在 Internet 上的终端可以通过 VPN 访问 VLAN5 中的信息资源。为了安全，要求企业网实施以下安全策略：

1）属于财务部门的 PC 不允许访问 Internet；

2）属于财务部门的 VLAN4 与属于信息服务部门的 VLAN5 之间不能相互通信；

3）允许 VLAN2 和 VLAN3 中的 PC 发起访问 Internet；

4）连接在 Internet 上的终端如果需要访问企业网，必须先通过 VPN 接入企业网，且只能访问 VLAN5 中的信息资源，不能与其它 VLAN 中的 PC 相互通信。

2.IP 地址分配

Internet 分配给该企业的网络信息如下：IP 地址为 192.1.1.1，子网掩码为 255.255.255.0，默认网关地址为 192.1.1.2。

参考网络拓扑结构如下图所示。

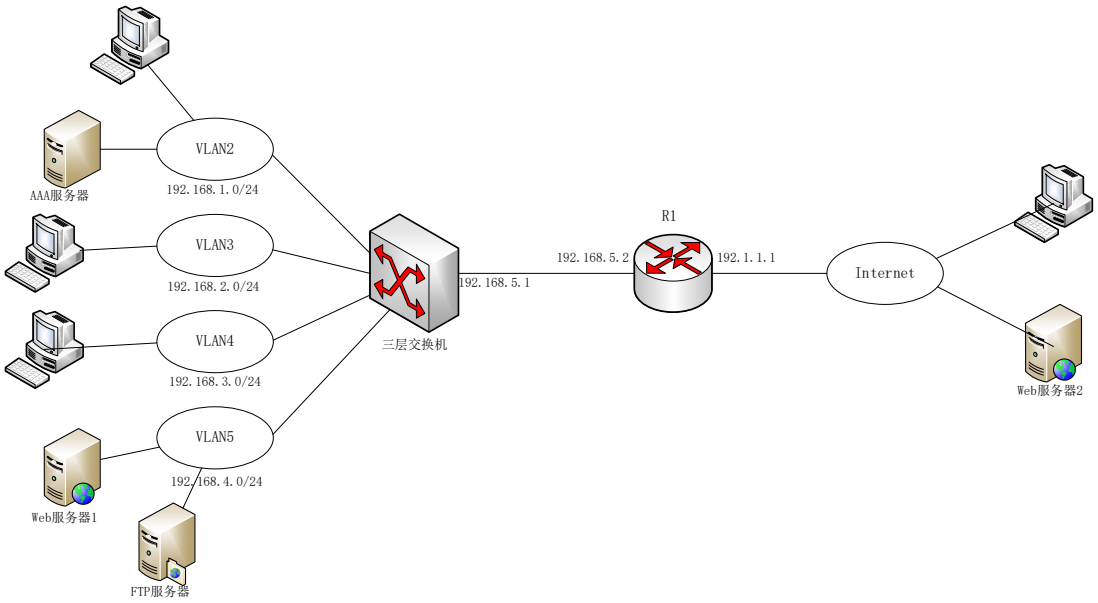

图 2 远程用户接入企业网拓扑图

3.详细需求

1）允许 VLAN2 和 VLAN3 访问 Internet（提示：需要在边缘路由器上启动 NAT）。

2）将边缘路由器作为 VPN 接入服务器，完成以下功能：对远程接入用户进行身份鉴别；为远程终端分配属于网络地址 192.168.6.0/24 的私有 IP 地址，同时在路由表中创建一项将该远程终端和边缘路由器之间的 IP 隧道与分配给该远程终端的私有 IP 地址绑定在一起的动态路由项；建立远程终端与边缘路由器之间的双向安全关联，实现远程终端与边缘路由器之间的安全传输；企业网设置 AAA 服务器，由它统一完成注册用户的身份鉴别过程。

3）财务部门 VLAN4 只能与 VLAN2 和 VLAN3 进行通信。

4）分配属于网络地址 192.168.6.0/24 的私有 IP 地址的远程终端只能与 VLAN5 之间进行通信，需要在远程终端上访问 Web 服务器和 FTP 服务器。（提示：需要在三层交换机上配置访问控制策略）

5）其它网络安全或可靠性设计。

注意：R1 作为企业边界路由器，不应向外部通告自己企业内部的路由信息。

题目 3：中小企业网络设计（难度：★★）

1.需求描述

某公司下设研发部、市场部、人力资源部、网络运维部 4 个部门，其中研发部有计算机 30 台、市场部有计算机 20 台，人力资源部有计算机 5 台，网络运维部有计算机 3 台，另外有 4 台服务器组成的服务器群提供 Web、FTP、DNS 等各种服务。要求完成该企业网络的设计、架构、配置和管理功能。

2.IP 地址分配

ISP 给公司分配了 15 个公网 IP 地址：202.0.0.1－15/29。

公司内部两台服务器对外分别提供 Web 服务和 FTP 服务，其公网 IP 地址分别为 202.0.0.1 和 202.0.0.2，对应的内网地址分别为 192.168.4.1 和 192.168.4.2。

参考网络拓扑结构如下所示。

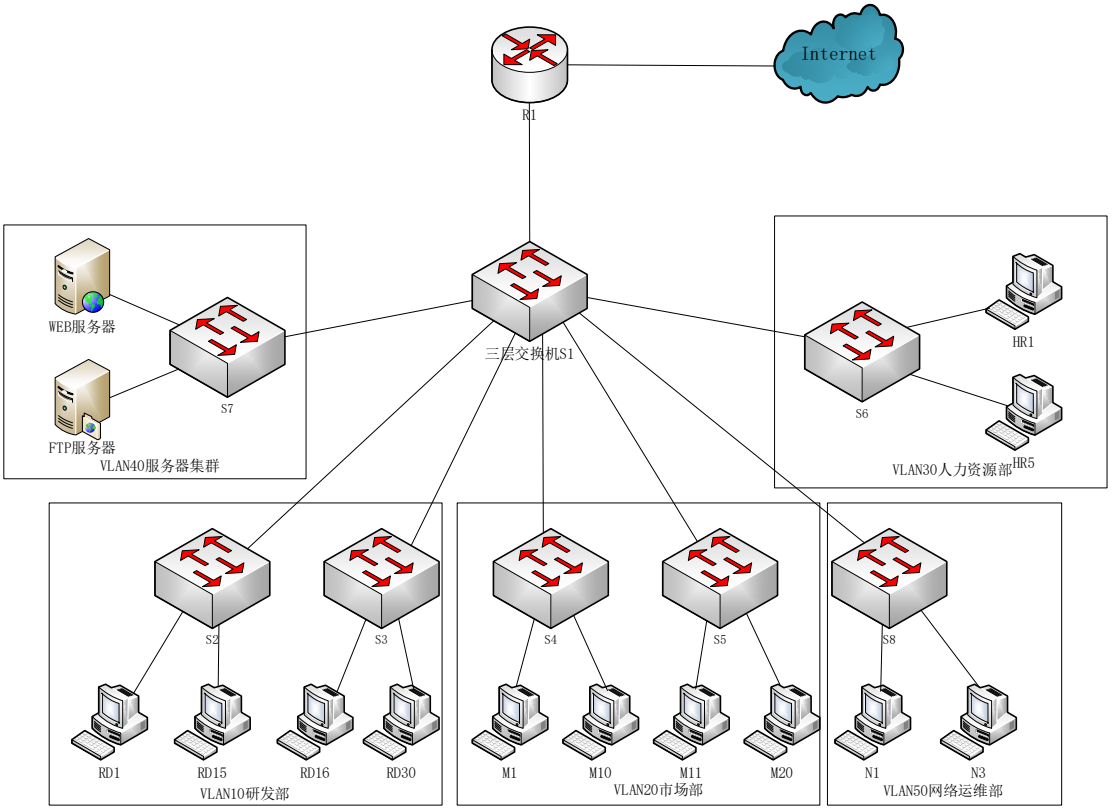

图 3 中小企业网络拓扑图

3.详细要求

1）为所有交换机和路由器配置远程管理权限，Telnet 终端登录，由网络运维部所有 PC 可以进行远程管理；

2）所有接入到公司内部局域网的计算机必须经过 802.1x 认证才能联网；

3）研发部、市场部、人力资源部互相之间不能访问；

4）研发部和人力资源部不能访问 Internet，市场部可以访问 Internet；

5）所有部门的计算机均可访问服务器群中任一台服务器；

6）要求公司内部局域网访问 Internet 使用 NAT 地址转换；

7）要求公司内部两台服务器对外分别提供 Web 服务和 FTP 服务（分配的 IP 见 IP 地址分配）。

8）其它网络安全或可靠性设计。

注意：R1 作为企业边界路由器，不应向外部通告自己企业内部的路由信息。

题目 4：防火墙安全设计（难度：★★★）

企业网的设计目标是开放和安全，开放要求企业网能够与外部网络实现相互通信，安全要求外部网络访问企业网的过程实施严格控制。企业网安全常采用防火墙结构，来实现内部网络和外部网络的隔离，该结构由内部网、DMZ 非军事区和外部网组成，内部网主要实现内部终端与内部服务器之间互连，而内部网中的设备对于其他网络是透明的，DMZ 是企业网向外发布信息及实现与外部网之间信息交换的窗口，其他网络中的用户允许发起对 DMZ 中 Web 服务器的访问，DMZ 中的邮件服务器需要与其他网络中的邮件服务器交换信件。该结构参考拓扑结构如下所示。

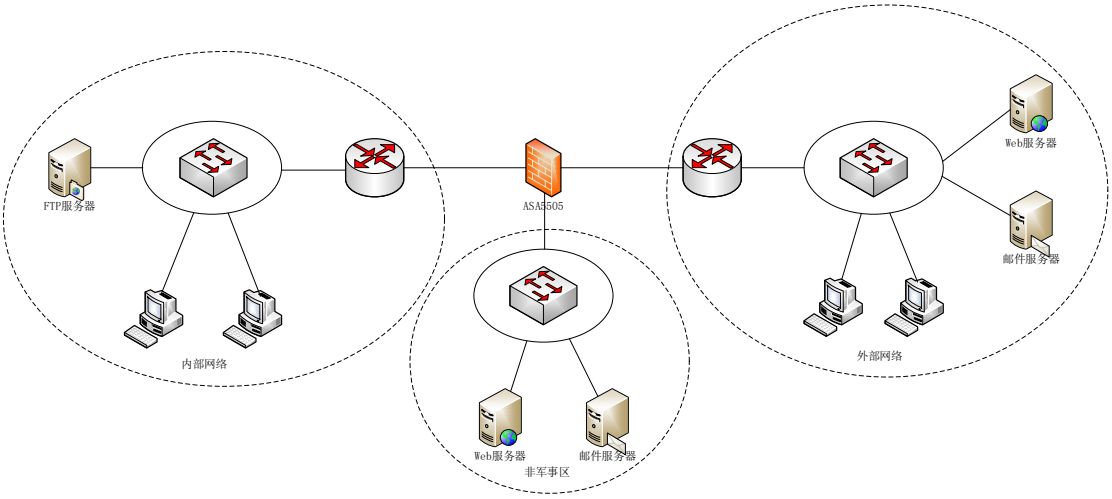

图 4 防火墙设计拓扑图

请根据以下要求完成详细设计及配置。

1）内部网络划分为两种 VLAN，VLAN10 可以访问外部网络，VLAN20 不可以访问外部网络，FTP 服务器只能被内部网络访问；

2）允许外部网络中的终端访问非军事区中的 Web 服务器；

3）允许非军事区中的邮件服务器与外部网络中的邮件服务器之间相互交换 SMTP 消息；

4）禁止其它网络间的通信；

5）其它网络安全或可靠性设计。

题目 5：学校实验机房网络设计（难度：★★★）

1.需求描述

以某学校计算机学院实验楼网络工程项目的应用需求为背景，规划一个约 10 个机房 500 台计算机的实验教学网络。要求将各个机房连成一个相对独立的局域网，保证网络互相连通，同时网络连通性可控，如某些机房考试时能够禁止该机房访问互联网。另外网络中心没有足够的公网地址分配给每台计算机，需要采用地址转换技术进行网络规划。

2.IP 地址分配

地址规划采用私网地址 10.0.0.0/8 网段，第二个字节代表楼层，第三个字节代表房间号，第四个字节表示其在房间中的位置编号，这样便于网络管理。如从 IP 地址 10.5.4.16 可以定位该计算机位于 5 层 504 机房 16 号机位。

网络中心给实验机房提供了 115.25.141.129~255/25 这个地址段共 128 个公有网络地址，其中将 115.25.141.193~254/26 作为公网地址，另外一段地址为其他服务器等设备使用。

参考网络拓扑结构如下图所示。

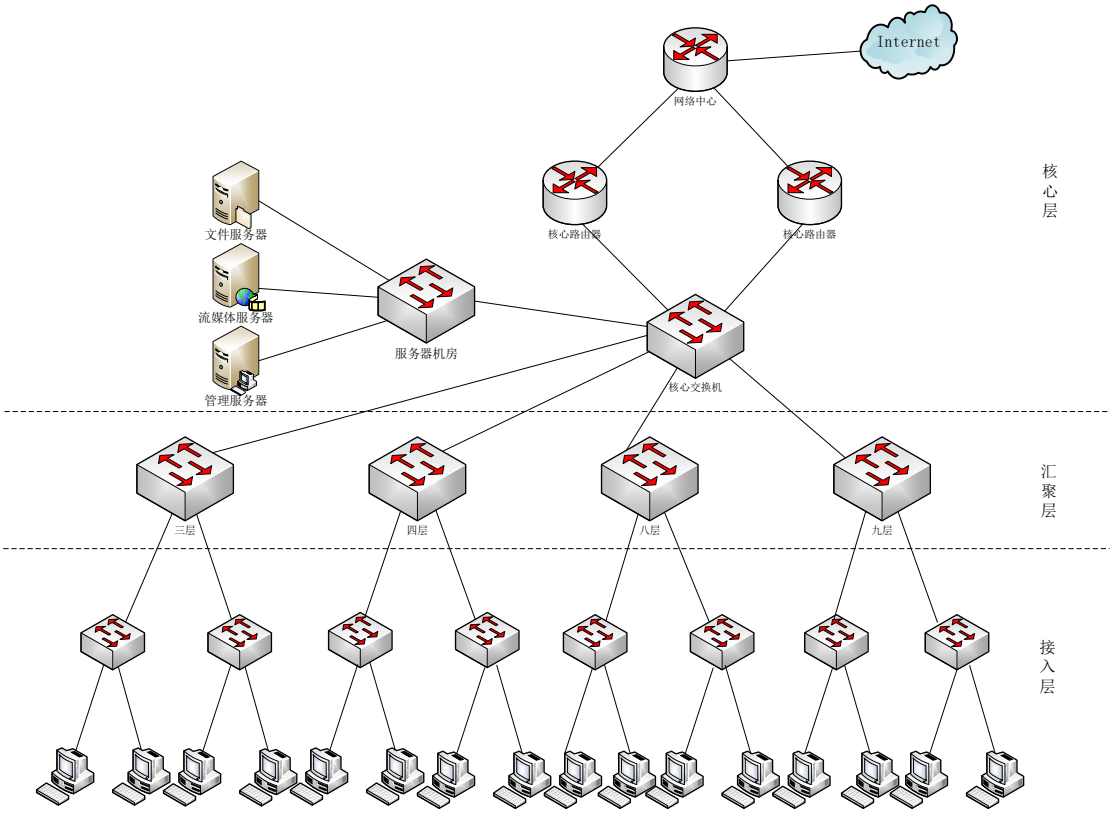

图 5 学校实验机房网络拓扑图

3.详细要求

1）将所有机房连成一个局域网，保证网络互联互通，合理分配 IP 和划分 VLAN；

2）公网地址向网络中心申请一个至少包括 128 个公网地址的地址池，采用 NAT 技术实现众多学生同时上网的需求；

3）核心链路考虑采用可靠性设计，如链路备份、设备备份和路由备份技术；

4）服务器机房和 Internet 不能互通，仅能与各机房之间通信；

5）其它网络安全或可靠性设计。

注意：网络中心路由器作为边界路由器，不应向外部通告自己内部的路由信息。汇聚层交换机应采用三层交换机并开启路由功能。

题目 6：无线网络设计规划（难度：★★★）

随着移动终端的普及，无线局域网日益成为使用最广泛的局域网。无线局域网的无线传输特性，要求 AP 和无线路由器必须对需要对其建立关联的终端进行身份鉴别，同时需要加密终端与 AP 和无线路由器之间传输的数据。

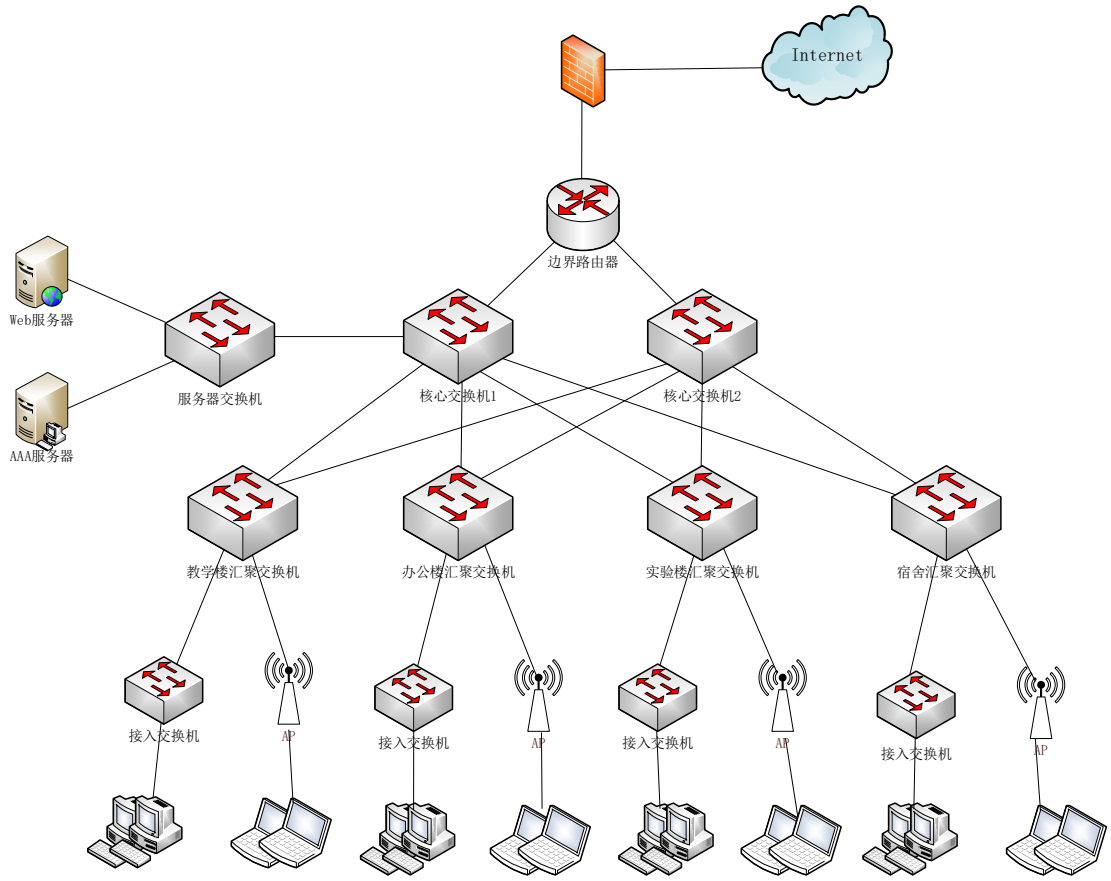

图 6 无线网络设计拓扑图

1）合理划分 VLAN 及分配 IP 地址，办公楼 VLAN 不能与其它 VLAN 相互访问，其它 VLAN 之间可以相互通信，各楼终端均可访问 web 服务器及 internet；

2）两台核心交换机上实现故障切换和负载均衡；

3）选择某种安全机制实现各终端与 AP 之间的连接通信；

4）设置 AAA 服务器，由它统一完成注册用户的身份鉴别过程；

5）其它网络安全或可靠性设计。

注意：企业边界路由器，不应向外部通告自己企业内部的路由信息。

题目 7：学校网络整体规划（难度：★★★）

1.需求描述

以某大学网络为例，以图书馆子网为例接入主干网。网络结构为典型的三层结构，核心层由三台高性能的三层交换机组成环形拓扑结构，核心层之间采用两条链路聚合方式以增加它们校园网络主干带宽。

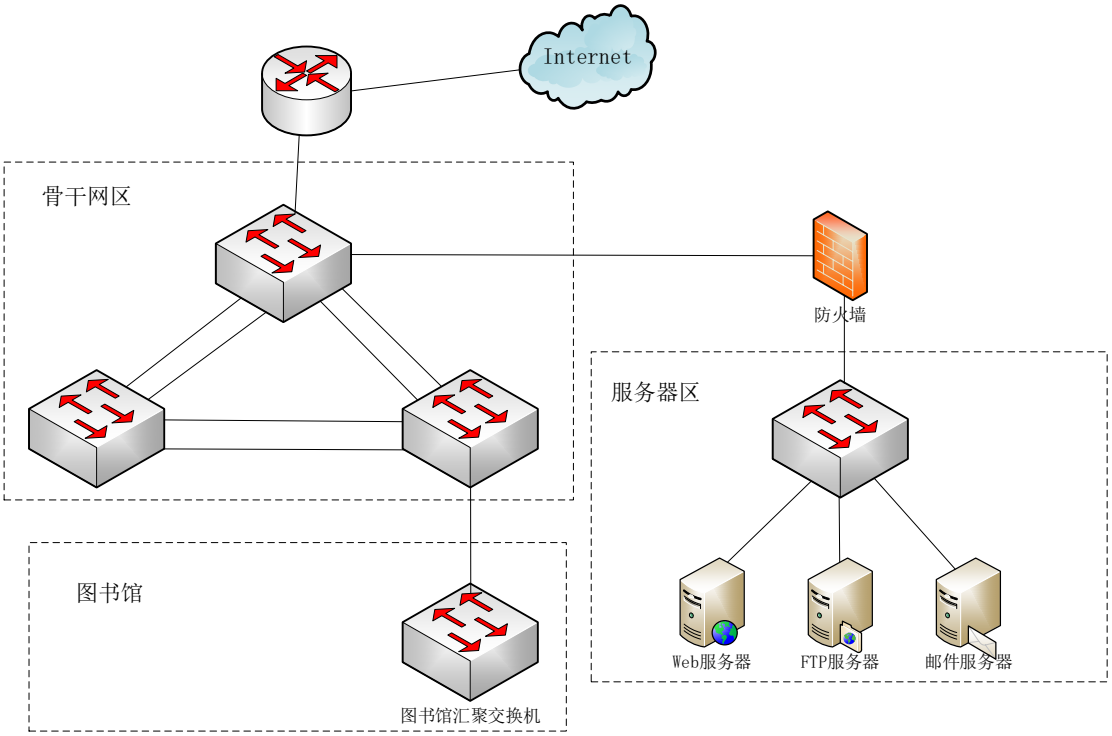

图 7 学校网络整体规划拓扑图

2.详细要求

1）根据参考网络拓扑完成网络拓扑结构设计，规划 IP 地址及设备端口；

2）图书馆分为两类用户，一类可通过 NAT 方式访问 Internet，一类禁止访问 Internet，仅可访问校内网络，合理规划 VLAN 划分及配置；

3）核心层之间运行 OSPF 动态路由协议，且采用两条链路聚合方式增加主干网带宽；

4）防火墙上只允许 Web 服务器和邮件服务器可以被网络上的其它计算机访问，FTP 服务器只允许内部网络访问；

5）其它网络安全及可靠性设计。

注意：边界路由器不应向外部通告内部的路由信息。

题目 8：城域网组网设计（难度：★★★）

1.需求描述

某省现有中学 80 所，随着教育信息化的逐步深入，教育城域网的建设成为校园互通的重点，同时也成为实现远程教学、多媒体教学及教学资源共享的关键。

建设教育城域网，希望通过网络建立教学辅助系统，为课堂教学提供丰富的实时音频、视频素材，并且希望网络建成后，不仅满足当前的业务需求，同时还要具备良好的可扩展性，能适应未来业务的发展和智能升级。

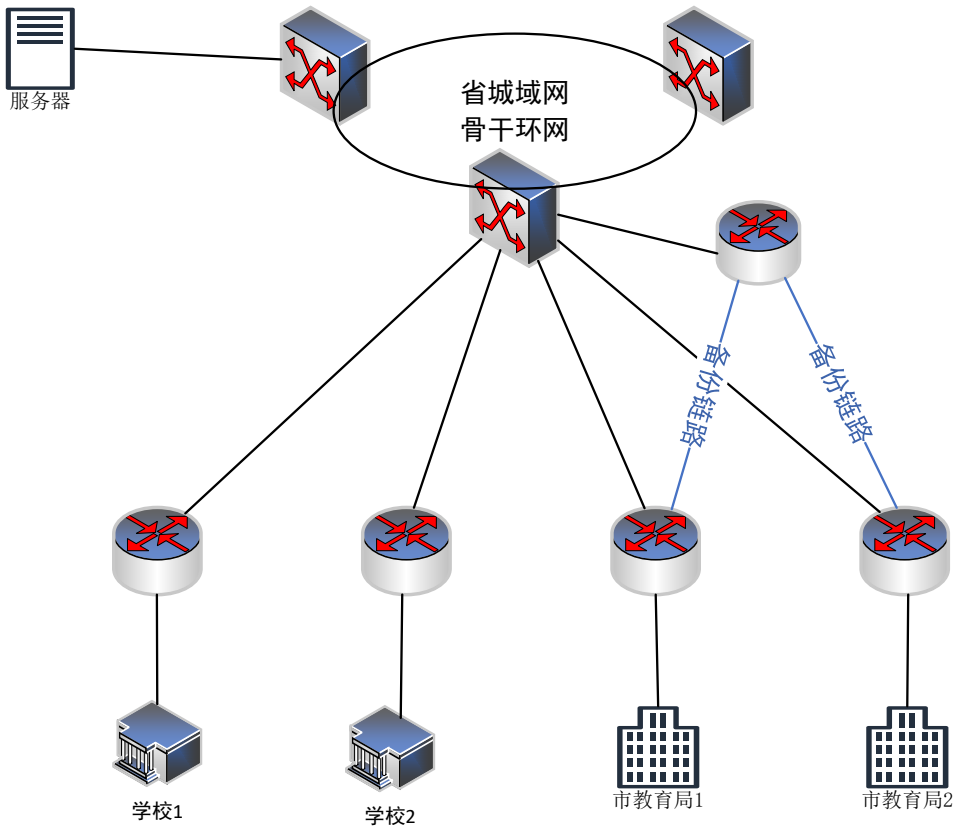

图 8 教育城域网拓扑图

2.详细要求

1）合理规划 vlan 和分配 ip 地址，各学校和各市教育采用私有 ip 地址；

2）各学校和各市教育局对外访问地址为 10.0.0.0/28；

3）要求每个地市教育局连接到省教育局增加一条备份链路，保证地市教育局到省教育局业务不中断，合理配置备份链路的主和关系；

4）各学校和各市教育局可访问位于省教育局的 web 服务器及 internet 和 ftp 服务器；

5）各学校和教育局可通过 telnet 管理本单位的网络设备；

6）省城域网可采用三台高性能三层交换机组成骨干环网来提高可靠性。

注意：各单位内部的边界路由器不向外部通告内部路由信息。

2. 网络编程要求（30 分，每个题目最多三个小组选择）

题目 1：ARP 数据包的封装发送和解析、ARP 欺骗（难度：★★★）

课程设计目的：

- ARP 协议用于完成 IP 地址与 MAC 地址之间的转换。通过封装与发送 ARP 数据包，加深对网络协议的理解，掌握 ARP 帧结构和工作原理及其对协议栈的贡献。另外，既然我们可以自己填充数据包，就可以玩一些基于 ARP 的“小把戏”，如基于 ARP 欺骗的监听。

课程设计要求：

- 编写程序，根据 ARP 帧的结构，封装数据包发送到局域网中。另外要捕获网络中的 ARP 数据包，解析数据包的内容，并将结果显示，并同时写入日志文件。
- 以命令行形式运行或图形界面
- 在标准输出中显示捕获的 ARP 报文的首部字段的内容。
- 模拟实现 ARP 欺骗的过程。如某台计算机 C 和计算机 A\B 位于同一个局域网中，那么 C 给 A 发送伪造的 ARP 应答包，对 A 说 B 的 ip 对应的 MAC 为 C 的 MAC 地址，那么 C 将获得 A 向 B 传输的数据。当时因为 ARP 缓存是动态的，有超时时间，所以必须每隔一段时间就给 A 或 B 发送一个 ARP 应答包。

课程设计分析：

- 使用原始套接字或者 winpcap，**捕获 ARP 数据包，并进行解析**
- 定义 ARP 首部的数据结构
- 自定义并填充数据包，发送数据包，捕获数据包，**实验测试要求发送 ARP 请求报文，然后通过 wireshark 获取 ARP 响应报文**
- 模拟实现 ARP 欺骗（单向或双向）

题目 2：IP 数据包的封装发送和解析和 IP 流量分析（难度：★★★）

课程设计目的：

- IP 协议是网络层的重要协议，通过封装与发送 IP 数据包，加深对网络协议的理解，掌握 IP 数据包结构和工作原理及其对协议栈的贡献。IP 包流量对于表征网络运行状态至关重要，学习和掌握网络性能分析的基础是学习 IP 包流量分析程序编程。

课程设计要求：

- 编写程序，根据 IP 的结构，封装数据包发送到局域网中。另外要捕获网络中的 IP 数据包，解析数据包的内容，并将结果显示，并同时写入日志文件
- 在标准输出中显示捕获的 IP 报文的首部字段的内容
- 编写程序，监控本地网络，捕获一段时间内以本机为源地址或目的地址的 IP 数据包，统计 IP 数据包的信息，列出本机与其他主机之间不同协议类型的 IP 数据包数量。程序统计信息包括源地址、目的地址、协议类型以及本机与不同主机之间不同协议类型 IP 数据包的数量。
- 以命令行形式运行或图形界面

课程设计分析：

- 使用原始套接字或者 winpcap，捕获 IP 数据包，**并进行解析**

- 定义 IP 首部的数据结构
- 自定义并填充数据包，发送数据包，捕获数据包，**实验测试要求使用 ping 命令发送报文，通过 wireshark 捕获响应报文**
- 在捕获符合条件的数据包之后，进行 IP 数据包信息的统计

题目 3：TCP 数据包的封装发送和解析、发现 TCP 服务（难度：★★★）

课程设计目的：

- TCP 是运输层的可靠传输协议。通过封装与发送这些数据包，加深对网络协议的理解，掌握 TCP 帧结构和工作原理及其对协议栈的贡献。
- 端口扫描往往用于获取目标系统上的端口信息，用于识别其上具有的 TCP 和 UDF 服务，常用的端口扫描技术有很多种，如 TCP connect 扫描、TCP SYN 扫描和 TCP FIN 扫描等。通过编程，可以了解客户机/服务器与端口扫描的工作原理。

课程设计要求：

- 编写程序，根据 TCP 帧的结构，封装数据包发送到局域网中。另外要捕获网络中的 TCP 数据包，解析数据包的内容，并将结果显示，并同时写入日志文件。
- 以命令行形式运行或图形界面
- 在标准输出中显示捕获的 TCP 的首部字段的内容。
- 编写程序发现服务器中开启的 TCP 服务，输出开启的 TCP 服务端口号。

课程设计分析：

- 使用原始套接字或者 winpcap，捕获 TCP 数据包，**并进行解析**
- 定义 TCP 首部的数据结构
- 自定义并填充数据包，发送数据包，捕获数据包，**实验测试可以考虑三次握手中的建立连接阶段，要求发送 TCP 连接报文，通过 wireshark 捕获数据包并分析各字段是否正确**
- 可考虑采用多线程来加快扫描速度

题目 4：FTP 客户机（难度：★★）

设计目的：

- 设计并实现一个 FTP 客户机的程序，了解 FTP 服务的基本原理和 FTP 协议的工作过程，加深对 TCP 协议和流式套接字编程的理解。

设计要求：

- 要求在 FTP 客户机程序中至少实现目录变换（CWD），列表（LIST），下载（RETR）功能。

- 以命令行形式或图形界面运行
- 输出内容：FTP 客户机与服务器交互过程中的命令与应答信息。下载指定 FTP 网站的指定文件（学校 FTP 网址：202.204.60.11）

题目 5：局域网文字聊天室（难度：★★）

设计目的：

- 设计并实现一个局域网内的文字聊天小程序，使用 Winsocket API 或 Winpcap 技术，完成局域网内主机之间文字信息传送

设计要求：

- 实现一对一文字对话
- 实现一对多文字通话
- 以命令行形式或图形界面运行

题目 6：路由跟踪小程序（难度：★★）

设计目的：

- 设计并实现一个基于 IP 的路由跟踪小程序，使用 IP 协议，记录该数据包经过的路由器，记录 IP 地址和往返时间，类似于 tracert。

设计要求：

- 输入：目的 IP 或主机名，输出：IP 报文由本机出发到达目的主机所经过的路由信息
- 以命令行形式或图形界面运行
- 如有余力，可考虑如果无法得到某路由器的 IP 地址，有什么替代办法

题目 7：子网内文件传送（难度：★★）

设计目的：

- 设计并实现一个局域网内部的文件传送工具，使用 TCP 协议进行可靠文字传输。可参考飞鸽传书。

设计要求：

- 以命令行形式或图形界面运行
- 不同结点上文件自动同步

题目 8：统计局域网内流量与网内各活动主机状态（难度：★★）

设计目的：

- 设计并实现一个监听 TCP/IP 网络流量的小程序，统计该网卡子网内的不同协议流量。
- 记录局域网内活动主机数量和活动端口，记录该活动主机的 IP 地址和端口号。

设计要求：

- 以命令行形式或图形界面运行

题目 9：简单邮件代理器（难度：★★）

设计目的：

- 设计并实现一个邮件收发器（类似于 outlook），使用 SMTP 协议和 POP3 协议，使用 socket 编程。

设计要求：

- 命令行或图形界面运行
- 按照指定地址，完成文字发送

题目 10：简单防火墙程序实现（难度：★★★）

设计目的：

- 防火墙能够监控本机访问网络的进程，根据用户需求限制特定进程网络访问权限，编程实现防火墙以深入理解防火墙的原理，掌握 windows 下应用层过滤数据包的基本技术，掌握进行简单数据过滤的基本方法。

设计要求：

- 编写一款简单的防火墙软件，要求能够以协议类型、IP 地址和端口为判别条件过滤网络上其他主机发给本主机的数据包；
- 要求能够明确显示当前有效的过滤条件，并将过滤事件写入日志文件保存；
- 若有余力可实现用户能够自由定义过滤条件。

3. 创新题：SDN 网络（60 分）

题目 1：新型网络架构 SDN 智能路由（难度：★★★★★）

设计目的：

- SDN 网络是新型网络架构，通过将网络设备的控制面与数据面分离开来，从而实现了网络流量的灵活控制，使网络作为管道变得更加智能，为核心网络及应用的创新提供了良好的平台；这样可以在控制面中设计不同的算法来控制数据包的转发，如采用深度学习、强化学习等方法，对数据包进行智能转发
- 设计并实现一个智能路由算法，能够结合网络 QoS 的一些指标对网络性能进

行优化。

设计要求：

- 在控制面实现机器学习（或深度学习、强化学习）算法的智能路由算法，基于 Mininet 搭建仿真环境进行实验，控制器可采用 RYU 等控制器（不限）
- 基于胖树拓扑（基础要求）和更加复杂的拓扑（可选）进行测试，和传统的 OSPF 算法进行对比，性能有何提升？对结果进行可视化分析

题目 2：基于 SDN 的 DDoS 攻击检测和防御技术（难度：★★★★★）

设计目的：

- 熟悉 SDN 架构及 mininet 仿真环境搭建
- 模拟 DDoS 攻击并实现一种 DDoS 攻击的检测方法
- 利用 sFlow 验证 DDoS 攻击的防御功能

设计要求：

- 基本要求：基于 Mininet 搭建仿真环境进行实验，控制器可采用 RYU 等控制器（不限），部署 sFlow 工具，获取接口流量信息并解析数据，对解析后的数据进行分析判断，能够检测出 DDoS 攻击，并对比防御未开启和开启后的流量变化，要求有可视化结果
- 进阶要求： 对上述工作进行改进，可考虑优化的点有：利用机器学习等方法检测 DDoS 攻击（检测精度）、更加复杂的防御策略，实验模拟正常的背景流量和攻击流量对 DDoS 攻击检测和防御方法进行测试

题目 3：基于 SDN 的数据中心流量负载均衡技术研究（难度：★★★★★）

设计目的：

- 熟悉 SDN 架构及 mininet 仿真环境搭建
- 了解负载均衡的概念、作用和原理，掌握基于 SDN 的负载均衡实现方法

设计要求：

- 基本要求：基于 Mininet 搭建仿真环境进行实验，控制器可采用 RYU 等控制器（不限），对链路负载进行计算，然后规划最优路径（使用最短路径算法完成），基于胖树拓扑进行测试并验证方法的有效性。
- 进阶要求：对上述工作进行改进，可考虑优化的点有：预测链路负载状况、考虑为用户制定个性化 QoS 服务、自适应的进行负载均衡等（可考虑一个或多个算法的创新），最后基于胖树拓扑（基础）和更加复杂的拓扑进行测试，给出性能提升可视化结果。

创新题提示：可阅读相关论文并复现论文的工作，同时可基于 Github 上的源码进行二次开发，最终答辩和报告中需要阐明自己的工作。

二、课程设计相关参考

1. Winpcap 结构体和函数说明

a) 结构体

在程序中，将要进行分析的各数据包头格式用结构体进行定义。这样便于对数据包的解析，使每个字段清楚易懂。

如：

- 以太帧头格式结构体，共 14 个字节：

```
typedef struct ether_header {
    unsigned char ether_dhost[6];      //目的 MAC 地址
    unsigned char ether_shost[6];      //源 MAC 地址
    unsigned short ether_type;             //协议类型
}
```

- IPv4 报头格式结构体，共 20 个字节：

```
typedef struct ipv4_header {
unsigned char ver_ihl;              //版本 (4 bits) + 首部长度 (4 bits)
unsigned char tos;                  //服务类型
unsigned short tlen;                //数据报总长度
unsigned short identification;      //标识
unsigned short flags_fo;            //标志 (3 bits) + 片偏移 (13 bits)
    unsigned char ttl;                  //生存时间
unsigned char proto;                //协议
unsigned short crc;                 //首部校验和
u_char ip_src[4];                   //源 IP 地址
u_char ip_dst[4];                   //目的 IP 地址
}
```

- IPv6 报头格式结构体，共 40 个字节：

```
typedef struct ipv6_header {
    u_char ver_tf;                  //版本号（4 bit）
    u_char traffic;                 //优先级（8 bit）
    u_short label;                  //流标识（20 bit）
    u_char length[2];               //报文长度（16 bit）
    u_char next_header;             //下一头部（8 bit）
    u_char limits;                  //跳数限制（8 bit）
```

```
    u_char Srcv6[16];              //源 IPv6 地址（128 bit）
    u_char Destv6[16];             //目的 IPv6 地址（128 bit）
}
```

- TCP 报头格式结构体，共 20 个字节：

```
typedef struct tcp_header {
    WORD SourPort;          //源端口号
    WORD DestPort;          //目的端口号
    DWORD SeqNo;            //序号
    DWORD AckNo;            //确认序号
    BYTE HLen;              //首部长度（保留位）
    BYTE Flag;              //标识（保留位）
    WORD Window;            //窗口大小
    WORD ChkSum;            //校验和
    WORD UrgPtr;            //紧急指针
}
```

- UDP 报头格式结构体，共 8 个字节：

```
typedef struct udp_header {
    u_short sport;          //源端口号
    u_short dport;          //目的端口号
    u_short len;            //数据报长度
    u_short crc;            //校验和
}
```

b) 常用函数

```
LPPACKET PacketAllocatePacket(void)
VOID PacketInitPacket(LPPACKET lpPacket, PVOID Buffer, UINT Length)
VOID PacketFreePacket(LPPACKET lpPacket)
VOID PacketCloseAdapter(LPADAPTER lpAdapter)
BOOLEAN PacketGetAdapterNames(LPSTR pStr,PULONG BufferSize)
LPADAPTER PacketOpenAdapter(LPTSTR AdapterName)
BOOLEAN PacketReceivePacket(LPADAPTER AdapterObject,LPPACKET lpPacket, BOOLEAN
        Sync)
BOOLEAN PacketSetHwFilter(LPADAPTER AdapterObject,ULONG Filter)
BOOLEAN PacketGetNetInfoEx(LPTSTR AdapterNames,npf_ip_addr *buff, PLONG NEntries)
```

```
BOOLEAN PacketSendPacket(LPADAPTER AdapterObject,LPPACKET lpPacket, BOOLEAN Sync)

int pcap_findalldevs(pcap_if_t **alldevsp, char *errbuf)
char *pcat_lookupdev(char *errbuf)
int pcap_lookupnet(char *device, bpf_u_int32 *netp,bpf_u_int32 *maskp, char *errbuf)
pcap_dumper_t *pcap_dump_open(pcap_t *p,char *filename)
Pcap_t * pcap_open_live(char * DeviceName,int snaplen,int promisc,int to_ms,char *errbuf)
int pcap_compile (pcap_t *p, struct bpf_program *fp, char *str, int optimize, bpf_u_int32 netmask)
int pcap_setfilter (pcap_t *p, struct bpf_program *fp)
int pcap_dispatch(pcap_t *p, int cnt,pcap_handler callback, u_char *user)
int pcap_loop (pcap_t *p, int cnt, pcap_handler callback, u_char*user)
        pcap_read()
u_char *pcap_next(pcap_t *p, struct pcap_pkthdr *h)
void pcap_close (pcap_t *p)
int pcap_setbuff(pcap_t *p,int dim)
int pcap_setmode(pcap_t *p,int mode)
pcap_stats(pcap_t *p, struct pcap_stat *ps)
int pcap_sendpacket(pcap_t *p,char *buf,int size)
FILE *pcap_file(pcap_t *p)
int pcap_fileno(pcap_t *p)
```

2. WinSock 常用结构体和函数说明

a) 结构体：

```
struct sockaddr {
        u_short sa_family;              /* address family */
        char    sa_data[14];            /* up to 14 bytes of direct address */
};

struct sockaddr_in {
        short   sin_family;
        u_short sin_port;
        struct  in_addr sin_addr;
        char    sin_zero[8];
};

struct in_addr { //采用不同方式定义一个32比特无符号数
        union {
                struct { u_char s_b1,s_b2,s_b3,s_b4; } S_un_b;
                struct { u_short s_w1,s_w2; } S_un_w;
```

```
                    u_long S_addr;
            } S_un;
#define s_addr    S_un.S_addr
                                        /* can be used for most tcp & ip code */
#define s_host    S_un.S_un_b.s_b2
                                        /* host on imp */
#define s_net     S_un.S_un_b.s_b1
                                        /* network */
#define s_imp     S_un.S_un_w.s_w2
                                        /* imp */
#define s_impno S_un.S_un_b.s_b4
                                        /* imp # */
#define s_lh      S_un.S_un_b.s_b3
                                        /* logical host */
};

struct    hostent {
            char       * h_name;                /* official name of host */
            char       * * h_aliases;    /* alias list */
            short      h_addrtype;              /* host address type */
            short      h_length;                /* length of address */
            char       * * h_addr_list; /* list of addresses */
#define h_addr    h_addr_list[0]          /* address, for backward compat */
};
```

b) 常用 socket 函数

```
int WSAStartup(WORD wVersionRequired, LPWSADATA lpWSAData);
int WSACleanup(void);

SOCKET accept (SOCKET s, struct sockaddr *addr, int *addrlen);
int bind (SOCKET s, const struct sockaddr *addr, int namelen);
int closesocket (SOCKET s);
int connect (SOCKET s, const struct sockaddr *name, int namelen);

int ioctlsocket (SOCKET s, long cmd, u_long *argp);
int getsockopt (SOCKET s, int level, int optname,
                                    char * optval, int *optlen);
u_long htonl (u_long hostlong);
u_short htons (u_short hostshort);
unsigned long inet_addr (const char * cp);
char * inet_ntoa (struct in_addr in);
int listen (SOCKET s, int backlog);
u_long ntohl (u_long netlong);
u_short ntohs (u_short netshort);
```

```
int recv (SOCKET s, char * buf, int len, int flags);
int recvfrom (SOCKET s, char * buf, int len, int flags, struct sockaddr *from, int * fromlen);

int select (int nfds, fd_set *readfds, fd_set *writefds, fd_set *exceptfds, const struct timeval *timeout);

int send (SOCKET s, const char * buf, int len, int flags);

int sendto (SOCKET s, const char * buf, int len, int flags, const struct sockaddr *to, int tolen);

int setsockopt (SOCKET s, int level, int optname, const char * optval, int optlen);

int shutdown (SOCKET s, int how);

SOCKET socket (int af, int type, int protocol);

struct hostent * gethostbyaddr(const char * addr,    int len, int type);

struct hostent * gethostbyname(const char * name);

int gethostname (char * name, int namelen);

struct servent * getservbyport(int port, const char * proto);

struct servent * getservbyname(const char * name, const char * proto);

struct protoent * getprotobynumber(int proto);

struct protoent * getprotobyname(const char * name);
```