# 课程介绍

* 1、OSI模型和TCP/IP模型;（知道）
* 2、局域网基础概念；（了解）
* 3、UDP编程；（掌握）

1. OSI模型和TCP/IP模型
   1. OSI模型的提出

开放式系统互联通信参考模型（Open System Interconnection Reference Model，缩写为 OSI），简称为OSI模型（OSI model），一种概念模型，由国际标准化组织（ISO）提出，一个试图使各种计算机在世界范围内互连网络的标准框架。

* + 1. ISO/OSI模型图



* + 1. ISO/OSI模型的影响：

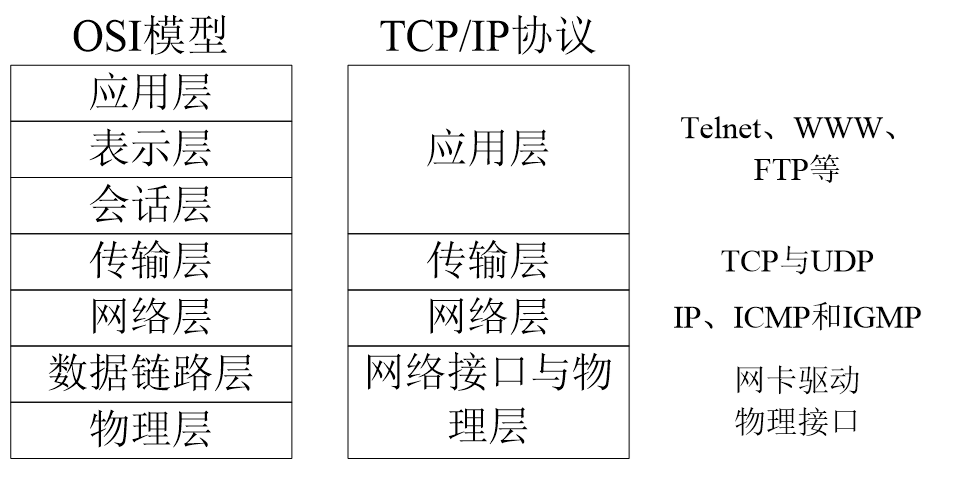
OSI是一个定义良好的协议规范集，并有许多可选部分完成类似的任务。它定义了开放系统的层次结构、层次之间的相互关系以及各层所包括的可能的任务，作为一个框架来协调和组织各层所提供的服务。

OSI参考模型并没有提供一个可以实现的方法，而是描述了一些概念，用来协调进程间通信标准的制定。即OSI参考模型并不是一个标准，而是一个在制定标准时所使用的概念性框架。

* 1. TCP/IP模型

最早的TCP/IP由文顿·瑟夫和罗伯特·卡恩两位开发，慢慢地通过竞争战胜其他一些网络协议的方案，比如国际标准化组织ISO的OSI模型。

TCP/IP的蓬勃发展发生在1990年代中期。当时一些重要而可靠的工具的出世，例如页面描述语言HTML和浏览器Mosaic，促成了互联网应用的飞速发展。



* + 1. 什么是协议

有的说英语，有的说中文，有的说德语，说同一种语言的人可以交流，不同的语言之间就不行了。为了解决不同种族人之间的语言沟通障碍，现规定国际通用语言是英语，这就是一个规定，这就是协议。

* + 1. 常用的TCP/IP协议

|  |  |  |
| --- | --- | --- |
| OSI中的层 | 功能 | TCP/IP协议族 |
| 应用层 | 文件传输，邮件，文件服务，虚拟终端 | TFTP，HTTP，SNMP，FTP，SMTP，DNS，Telnet |
| 表示层 | 数据格式化，代码转换，数据加密 | 没有协议 |
| 会话层 | 解除或建立与别的节点的联系 | 没有协议 |
| 传输层 | 提供端对端的接口 | TCP，UDP |
| 网络层 | 为数据包选择路由 | IP，ICMP，RIP，OSPF，BGP，IGMP |
| 数据链路层 | 传输有地址的帧以及错误检测功能 | SLIP，CSLIP，PPP，ARP，RARP，MTU |
| 物理层 | 以二进制数据形式在物理媒体上传输数据 | ISO2110，IEEE802.1,EEE802.2 |

1. 局域网基础
   1. IP地址的概念

**互联网协议地址**（Internet Protocol Address，又译为网际协议地址），缩写为IP地址（IP Address），是分配给网络上使用网际协议（IP）的设备的数字标签。

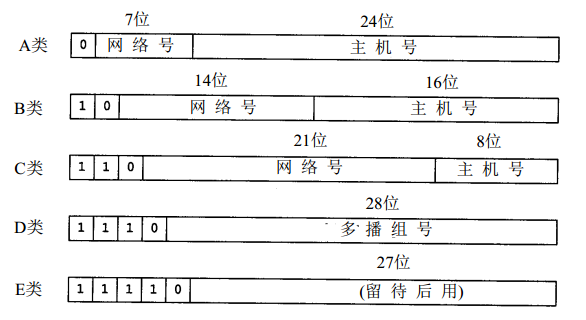
常见的IP地址分为IPv4与IPv6两大类。

* 1. IPv4

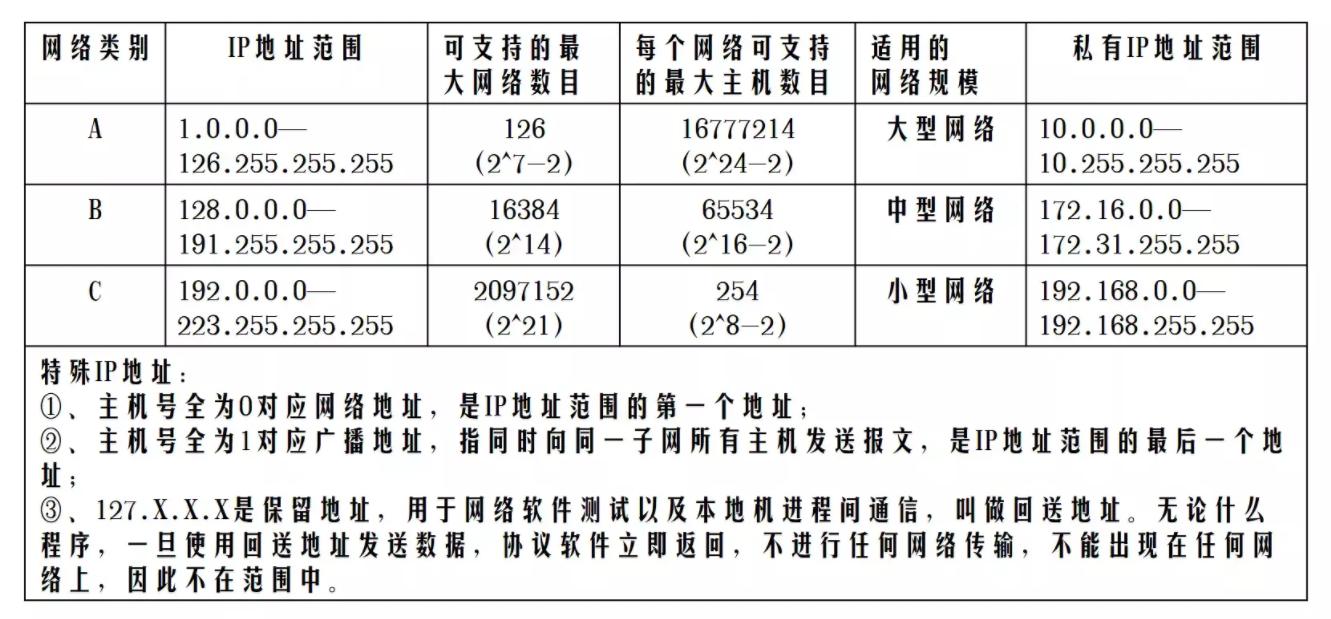
由32位二进制数组成，为便于使用，常以XXX.XXX.XXX.XXX形式表现，每组XXX代表小于或等于255的10进制数。

IP地址是唯一的。目前IP技术可能使用的IP地址最多可有4,294,967,296个，即232。不过，一些地址是为特殊用途所保留的，如专用网络（约1800万个地址）和多播地址（约2.7亿个地址），这减少了可在互联网上路由的地址数量。

随着互联网的快速成长，IPv4的42亿个地址的分配最终于2011年2月3日用尽。



A、B和C类有不同的网络类别长度，剩余的部分被用来识别网络内的主机，这就意味着每个网络类别有着不同的给主机编址的能力。D类被用于多播地址，E类被留作将来使用。



IP地址127．0．0．1 ~ 127．255．255．255用于回路测试，如：127.0.0.1可以代表本机IP地址。

* 1. 什么是子网掩码

子网掩码（subnet mask）又叫“网络掩码”， 它是一种用来指明一个IP地址的哪些位标识的是主机所在的网络地址以及哪些位标识的是主机地址的位掩码。

* 子网掩码是一个应用于TCP/IP网络的32位二进制值，每节8位，必须结合IP地址对应使用。
* 子网掩码32位都与IP地址32位对应，如果某位是网络地址，则子网掩码为1，否则为0。
* 子网掩码可以通过与IP地址“与”计算，分离出IP地址中的网络地址和主机地址，用于判断该IP地址是在局域网上，还是在广域网上。
* 子网掩码一般用于将网络进一步划分为若干子网，以避免主机过多而拥堵或过少而IP浪费。

假设IP地址为“192.168.1.1”子网掩码默认就为“255.255.255.0”，有时候为了方便，也可以写成192.168.1.1/24。

* 1. 端口

IP地址可以识别一个网络中的电脑是谁，但是电脑收到数据后，到底是给浏览器，还是给QQ这样的软件那？显然在识别一个数据的具体服务时，需要一个标识，而这个标识，就是端口。

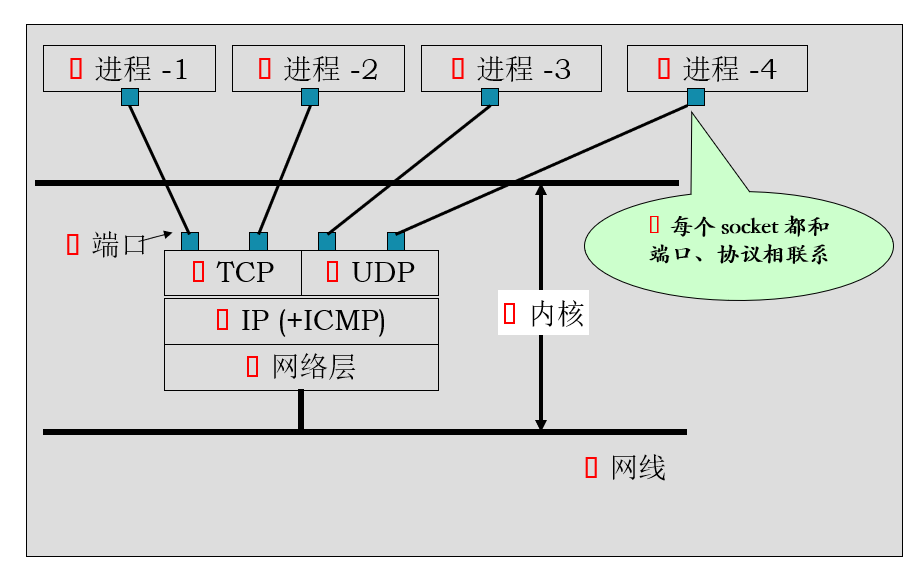
* 为了区分一台主机接收到的数据包应该转交给哪个进程来进行处理，使用端口号来区别；
* 端口是运行在传输层的概念，在TCP/IP协议簇中，UDP和TCP这2个协议具备独立的端口号；
* 端口号是16bit的，表示的范围是0-65535。其中小端口号已经被IANA (Internet Assigned Numbers Authority)定义：
* 0到1023号端口：标准服务的常用端口，大部分已经被内定了。
* 1024到49151号端口：已经被登记的端口。
* 49152到65535号端口：可以被动态申请的端口。

在Linux系统下，使用netstat -an查看端口

1. socket网络基础概述
   1. 什么是socket

socket本身是一种进程间通信的方式，最开始可以作为本地的进程之间数据的传输方式。在本地可以通过进程PID来唯一标识一个进程，但是在网络中这是行不通的。

而非本地的进程，该怎么标识自己那，就是前面我们说的IP地址和端口号了。



在python中使用socket模块封装了系统底层的socket操作。

* 1. python的socket模块基础

socket.socket(family=AF\_INET, type=SOCK\_STREAM, proto=0, fileno=None)

实际开发中，主要是family和type参数需要使用

|  |  |  |
| --- | --- | --- |
| 参数 | 选项 | 含义 |
| family | AF\_INET | 用于IPV4的通信 |
| AF\_INET6 | 用于IPV6的通信 |
| AF\_UNIX | 类unix的进程间通信 |
| type | SOCK\_STREAM | 面向连接的操作，TCP/IP里指代TCP协议 |
| SOCK\_DGRAM | 面向数据包的操作，TCP/IP里指代UDP协议 |
| SOCK\_RAW | 原始套接字，比如ICMP协议 |
| 返回值 | socket对象 | 网络通信的API访问全部通过这个对象来访问 |

常用写法：

sock = socket.socket() # 创建TCP的socket

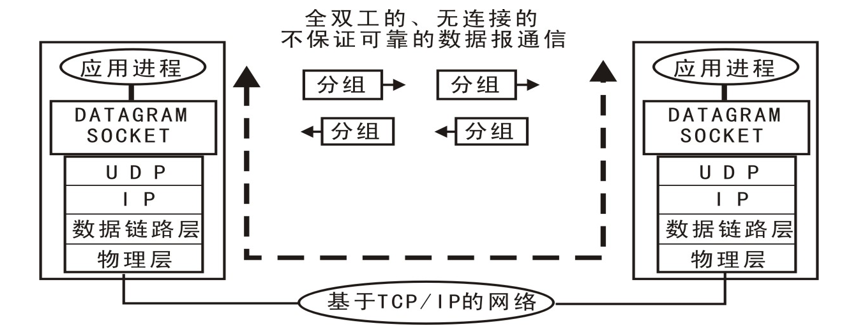
sock = socket.socket(AF\_INET, SOCK\_DGRAM) # 创建UDP的socket

其他socket的接口函数将在具体的TCP/UDP编程中介绍。

1. python的UDP编程基础
   1. 什么是UDP

UDP --- 用户数据报协议，是一个无连接的简单的面向数据报的运输层协议。UDP不提供可靠性，它只是把应用程序传给IP层的数据报发送出去，但是并不能保证它们能到达目的地。由于UDP在传输数据报前不用在客户和服务器之间建立一个连接，且没有超时重发等机制，故而传输速度很快。

UDP是一种面向无连接的协议，每个数据报都是一个独立的信息，包括完整的源地址或目的地址，它在网络上以任何可能的路径传往目的地，因此能否到达目的地，到达目的地的时间以及内容的正确性都是不能被保证的。



* 1. UDP的使用场景

UDP是面向消息的协议，通信时不需要建立连接，数据的传输自然是不可靠的，UDP一般用于多点通信和实时的数据业务，比如：

* 发送小尺寸数据（如对DNS服务器进行IP地址查询时）
* 在接收到数据，给出应答较困难的网络中使用UDP。（如：无线网络）
* 适合于广播/组播式通信中。
* MSN/QQ/Skype等即时通讯软件的点对点文本通讯以及音视频通讯通常采用UDP协议
* 流媒体、VOD、VoIP、IPTV等网络多媒体服务中通常采用UDP方式进行实时数据传输
  1. C/S模式

客户端-服务器（Client/Server）架构、C/S架构，是一种网络架构，它把客户端 (Client) （通常是一个采用图形用户界面的程序）与服务器 (Server) 区分开来。每一个客户端软件的实例都可以向一个服务器或应用程序服务器发出请求。有很多不同类型的服务器，例如文件服务器、游戏服务器等。

另一种目前广范使用的网络架构类型是点对点架构（P2P），不同于主从式架构，网络上的每个使用端或程序的实体都拥有相同的档次，同时扮演客户端与服务器的角色。

在初学者写代码前，一定要先确定谁是服务器，谁是客户端。

* 1. UDP简单服务端

根据C/S模型的要求，首先应该先有一个服务器，等待客户端的请求，然后再根据请求内容给予服务，那么UDP的服务器的设计流程就是：

1. **申请对象**：从内核中申请一个具有UDP特性的socket对象；
2. **配置结构**：用元组结构来表示服务器要监听的端口和IP地址；
3. **绑定结构**：使用bind系统调用，向内核告知开放哪个端口和哪些IP地址；
4. **等待数据**：使用recvfrom来接收数据，若没有收到数据，阻塞等待；

import socket

# **申请对象**，使用TCP/IP协议中的IP和UDP协议层

udp\_client = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

# **配置结构**，监听所有IP网络，接收8899这个端口的数据

src\_addr = ('', 8899)

# **绑定结构**，通过bind把配置好的结构告知给内核

udp\_client.bind(src\_addr)

# **等待数据**，以1K为单位接收UDP的数据

data, remote\_addr = udp\_server.recvfrom(1024)

print("获取到{}从{}".format(data.encode('utf-8'), remote\_addr))

* 1. UDP简单客户端

有了服务器后，只需要设计一个客户端，向服务器发起请求，就完成了C/S的模型了。

那么UDP的客户端设计流程是：

1. **申请对象**：从内核中申请一个具有UDP特性的socket对象；
2. **配置目标**：作为客户端，需要知道服务器的身份信息，才可能向指定的地方发送数据；
3. **发送数据**：使用sendto向目标地址发送数据；

import socket

# 申请对象

udp\_server = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

# 配置目的地的IP和端口

dest\_addr = ('192.168.199.128', 8899)

# 调用sendto发送数据

udp\_server.sendto('你好'.encode('utf-8'), dest\_addr)

* 1. 网络诊断的方法

在实际编程中，很多时候会遇到，收不到数据的情况，下面我们说明下遇到这种问题后，调试的思路和工具。这种方法不仅适用于本地虚拟机测试，也适用于爬虫，网站后台开发等业务。

整体思路：

根据TCP/IP的四层模型，我们的程序是应用层开发，如果程序收不到数据，那可以肯定是下面3层出了问题，所以建议使用排除法，逐一缩小问题的范围，来调整代码逻辑。建议从下到上，逐一排查。

* + 1. 物理接口层

这层最大的问题，就是网线没有插好，对于本机外网数据获取，肯定是看物理机的网口灯是否有亮或者闪烁。

对于虚拟机开发来说，就要点虚拟机的设置，看网卡是否有“上电”。

这层难度到不大，主要是要有这个意识。

* + 1. 网络层

本层最大的问题就是：

* 当访问的目的地址是局域网地址时，本机和目的机没有在一个网段。
* 当访问的目的地址是外网地址时，本机可能没有连接外网的权限。

本层的检测工具是：ping命令

只需要在本机ping 目标地址，如果有回应，说明联通，若超时，那么说明网络层不通，开始排查错误。排查错误的方法是：

ifconfig查看本地的IP地址是否获得

若是局域网通信，判断本地IP地址是否和目的IP地址一个网段

若是外网通信，先ping百度，若不通，说明不能连接外网；若通，说明连接的服务器出问题了，问题在对方，而不是自己。

连接外网，若是虚拟机使用，大部分原有是NAT服务忘记开了。

【总结】：

本层问题排除点：

局域网通信还是外网通信

ping的通吗？注意不要ping windows的IP，尽量使用window ping linux或linux互ping。

* + 1. 传输层

只要双方能ping通，那么如还收不到数据，解决起来就简单多了。

传输层的问题，一般就是服务器开放的端口号，和客户端访问的端口号不一样，那么：

首先在服务端netstat -tuan判断下端口是否开放了，开放的端口号是多少。如果这项判断没问题，那么还有一种可能性就是服务端的防火墙打开了。

关闭防火墙的办法，只要简单搜索下百度就可以了。

1. 课程总结
   1. 重点
      * 1. TCP/IP模型、OSI模型，以及他们的关系。（面试常考，注意不能说错顺序）
        2. IP和端口的意义，网络不通时，排查的方法。
        3. UDP通信的特点和简单的UDP的C/S模型。
   2. 难点
      * 1. IP地址的分类，子网掩码的作用
        2. IP地址和子网掩码的作用
        3. 网络诊断的方法（逐步熟悉）
   3. 如何掌握？
      * 1. 抄写代码，画出网络通信模拟图
        2. 故意写错代码，根据错误提示，不断累积经验
2. 课后练习
   * + 1. 抄写下面名词解释：

IP ： 指的是互联网协议（Internet Protocol）

IP地址 ：IP地址是IP协议提供的一种统一的地址格式，它为互联网上的每一个网络和每一台主机分配一个逻辑地址，以此来屏蔽物理地址的差异。

ICMP : 互联网控制消息协议（Internet Control Message Protocol），ping命令使用了该协议。属于网络层协议

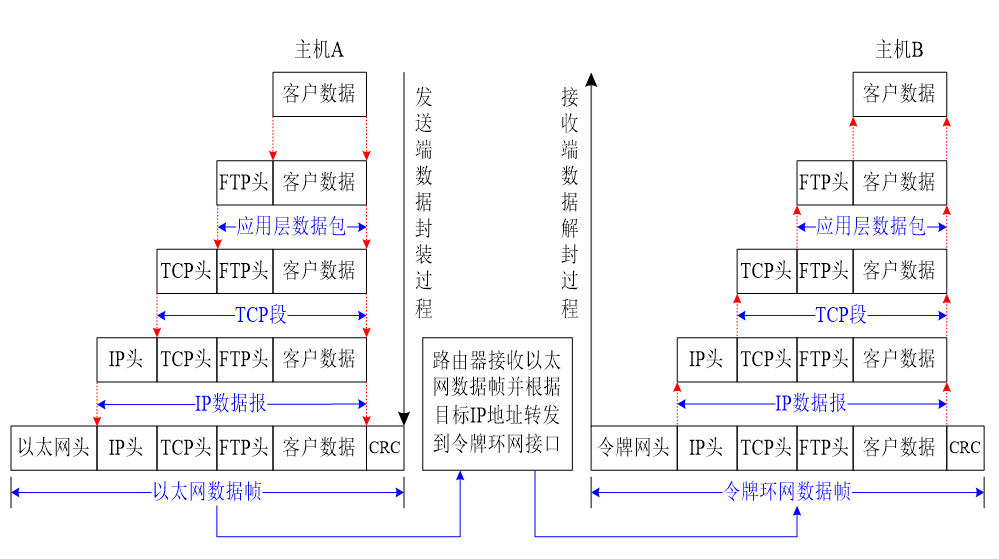
UDP ：用户数据报协议（User Datagram Protocol），属于传输层

DNS ：域名系统（Domain Name System），通过它完成域名和IP地址转换的功能，属于应用层。

DHCP ：动态主机配置协议（Dynamic Host Configuration Protocol），目的是自动获取IP地址。属于应用层协议。

* + - 1. 抄写最简单的UDP程序和UDP服务端、客户端的步骤流程，尽量自己不看参考来写。
      2. 使用UDP协议，写一个具有echo功能的服务器和客户端
      3. (扩展题)使用UDP协议，写一个C/S的聊天程序。

1. 面试题
   * + 1. 请说明TCP/IP协议的分层模型？
       2. 端口是什么，他的作用是什么？
2. 扩展知识或课外阅读推荐（可选）
   1. TCP/IP数据流



* 1. 子网划分
     1. 为什么需要划分子网那？

两台计算机要通讯，首先要判断是否处于同一个广播域内，即网络地址是否相同。如果网络地址相同，表明接受方在本网络上，那么可以把数据包直接发送到目标主机，否则就需要路由网关将数据包转发送到目的地，当需要将一个给定的网络划分为各个互不相关的网络时，就需要进行子网划分了。

* + 1. 怎么划分子网？

**将IP地址中的主机号部分再拿出某几位来作为网络号，剩下的部分作为主机号。**

这个知识点，在笔试中是非常容易考到的。

下面看一个例子：

**现有一个C类网210.55.23.X。由于有四个不同的部门要使用这段IP，为了不让各个部门之间相互干扰，我们需要将原来给出的一个网络划分为四个子网，以使得各个子网间互不影响。**

这个网段能够为自己使用的就是最后一个X，他代表8个bit。为了划分4个网段，需要2个bit来标记，所以我们将IP地址的主机部分中（8位）拿出前面2位来作为我们的子网网络号部分，因此用作主机的位数就只有剩下的6位

210.55.23.xxxx xxxx

|  |  |  |
| --- | --- | --- |
| 网络号 | 主机号二进制范围 | 主机范围 |
| 210.55.23.00xx xxxx | **00**00 0001 -- **00**11 1110 | 1 - 62 |
| 210.55.23.01xx xxxx | **01**00 0001 -- **01**11 1110 | 65 - 126 |
| 210.55.23.10xx xxxx | **10**00 0001 -- **10**11 1110 | 129 - 190 |
| 210.55.23.11xx xxxx | **11**00 0001 -- **11**11 1110 | 193 - 254 |

由于我们将原来IP地址中主机号的前两位用来作为了网络号部分，因此，为了让计算机能知道这两位是网络号，所以我们需要将相应的子网掩码中对应的这两位设置为1。

所以子网掩码应设置为255.255.255.192。

* + 1. 总结

根据子网ID借用的主机位数，我们可以计算出划分的子网数、掩码、每个子网主机数，以C类网络为例：

|  |  |  |  |
| --- | --- | --- | --- |
| 借用位数 | 子网掩码 | 子网数 | 每个子网的主机数 |
| 2 | 255.255.255.192 | 2 | 62 |
| 3 | 255.255.255.224 | 6 | 30 |
| 4 | 255.255.255.240 | 14 | 14 |
| 5 | 255.255.255.248 | 30 | 6 |
| 6 | 255.255.255.252 | 62 | 2 |

网络面试常用知识点：http://www.cnblogs.com/zyf-zhaoyafei/p/4716297.html

* 1. 网络抓包工具
     1. 什么是wireshark

wireshark是非常流行的网络封包分析软件，功能十分强大。可以截取各种网络封包，显示网络封包的详细信息。

wireshark是开源软件，可以放心使用。 可以运行在Windows和Mac OS上。

为了安全考虑，wireshark只能查看封包，而不能修改封包的内容，或者发送封包。

在使用时，务必右键点击图标，使用“管理员权限”打开。

* + 1. wireshark的使用

wireshark是捕获机器上的某一块网卡的网络包，当你的机器上有多块网卡的时候，你需要选择一个网卡。

使用wireshark最重要的是使用过滤器来查看我们需要的数据包，使用过滤是非常重要的， 初学者使用wireshark时，将会得到大量的冗余信息，在几千甚至几万条记录中，以至于很难找到自己需要的部分。搞得晕头转向。

过滤器有两种：

一种是显示过滤器，就是在捕获的记录中找到所需要的记录。

一种是捕获过滤器，用来过滤捕获的封包，以免捕获太多的记录。

* + 1. 过滤表达式的规则
* 协议过滤

比如TCP，只显示TCP协议。

* IP 过滤

比如 ip.src ==192.168.1.102 显示源地址为192.168.1.102，

ip.dst==192.168.1.102, 目标地址为192.168.1.102

* 端口过滤

tcp.port ==80, 端口为80的

tcp.srcport == 80, 只显示TCP协议的源端口为80的。

* http模式过滤

http.request.method=="GET", 只显示HTTP GET方法的。

* 逻辑运算符为 AND（&&）OR（||）
  + 1. wireshark抓包结构



各行信息说明：

* Frame: 物理层的数据帧概况；
* Ethernet II: 数据链路层以太网帧头部信息；
* Internet Protocol Version 4: 互联网层IP包头部信息；
* Transmission Control Protocol: 传输层的数据段头部信息，此处是TCP；
* Hypertext Transfer Protocol: 应用层的信息，此处是HTTP协议；