---------------------------------------------------------------物理层

**物理层主要做什么？**

利用传输介质为通信的两端建立物理连接。

物理层主要屏蔽物理传输介质和传输方式 让上面的网络层更多的注重协议和服务 还有主要是实现比特流的透明传输

**主机之间的通信方式？**

**全双工通信**

支持通信发送方和接收方同时传输信息和接收信息 同一时间一个信道可以有多个信息传输 比如打电话

**半双工通信**

支持通信发送方和接收方同时互相传输信息 但是同一时间一个信道只能有一个信息在传输

**单工通信**

发送方和接收方同一时间只有一方发送和接收信息 信息是单向传输

-------------------------------链路层

**数据链路层的三个基本问题**

**1. 透明传输**

在帧的首部和尾部添加转义字符 用户是察觉不到的 比如字节填充算法 在每一个帧的结束时 添加ESC字符 已表示一个帧传输的结束

并且会将此帧弹出 实现了透明传输。

**2. 差错检测**

CRC检测传输比特差错

**3. 封装成帧**

从网络层传下来的数据packet 封装成frame 添加头部和首部 说明了帧大小的界限

**MAC地址和IP地址分别作用？**

MAC地址是电子设备的物理地址 用于区别不同的设备 IP地址是计算机的逻辑地址 用于区别网络中不同的计算机

**为什么有了MAC地址还要IP地址？**

如果拿MAC地址进行寻址 非常耗费内存 满世界的找

同一个子网内的设备 给其分配的IP地址前缀是一样的 路由器只用按照IP地址前缀 就可以知道设备来自哪个子网了

**为什么有了IP地址还要MAC地址？**

知道了MAC地址才能为其所在的地域分配IP地址

只有IP地址的话 区分不了不同的物理设备

**私网地址和公网地址之间进行转换后 同一局域网内两个私网转换后 从外面看一样吗？**

动态转换 - 一个私网对应一个公网 经过转换后的公网IP不一样

端口复用 - 每个子网对应一个公网 端口不一样

**以太网中的CSMA/CD协议？**

载波监听

总线每个节点 都在监听互相之间连接的信道 如果信道空闲 立即传输数据 如果忙碌中 则等待

检测碰撞

如果发现两个以上的节点传输数据发生碰撞 则停止传输 二进制避让策略算法

**PPP协议**

用于建立网桥、路由器、主机之间的通信 实现点对点通信 不可靠协议 简易实现

动态分配IP 身份验证

---------------------------------网络层

**IP协议的定义和作用？**

Internet Protocol: 互联网协议 支持互相连接网络之间的数据报协议。

提高网络的可扩展性 IP属于无连接 不可靠 尽力而为的数据包传输服务。

寻址和路由

IP地址携带源IP地址和目的IP地址来标识该数据包的源主机和目的主机。

数据报在传输过程中 每个中间路由器会根据路由表来选择合适的路径

分段和重组

IP数据报在传输过程中 会流经不同的网络 不同的网络对数据报的最大长度限制是不同的

IP协议通过给每个IP数据报分配标识符以及组装的信息 使得数据包能够分段传输 到达源头后 恢复初原来的IP数据报

**IPv4地址不够如何解决？**

IPv6 12进制 IPv4的下一个版本 星球级的IP地址 就算一个星球的每个沙子都是一个地址 都能够用完

NAT 地址网络转换协议 能将

DHCP 动态分配主机协议

**域名和 IP 的关系，一个 IP 可以对应多个域名吗**

域名是绰号 一个IP是一个身份证 一个IP可以对应多个域名

**路由器的分组转发流程？**

1 从IP数据报里取出目的地IP地址 找到其所在的网络

2. 判断IP地址所在的网络是否与本路由器直接相连

3. 从路由器的路由表里查找是否存在之前访问过的路径 如果有那么就按照这条路径走就好了

4.如果没有在表中 那么去要找到匹配路由 发送到下一跳的路由器

5. 如果路由表有设置默认路由 默认路由转发

6 没有合适路由路径 报错

**路由器和交换机的区别？**

交换机： 通过硬件实现 同一局域网下 主要通过利用设备的物理设备来建立通信 确定数据的发送方和目的地

路由器： 通过软件实现 利用不同设备的IP逻辑网络地址 来确定数据的发送的接收

**ICMP协议概念/作用**

辅助IP协议完成可靠传输服务 验证网络通畅

**ICMP的应用**

Ping 即因特网包探测器，是一种工作在网络层的服务命令，主要用于测试网络连接量

(echo响应报文)

Traceroute

跟踪寻找从发送端到接收端所经过路由的最短的TTL时间的路径

**两台电脑连起来ping不通 怎么办？**

1. 检查防火墙

2. 延迟过高？

3. 网卡驱动

**ARP地址解析协议的原理？**

将IP地址解析为物理地址

**网络地址转换NAT？**

将内部私网地址翻译成公网IP地址 解决了IP地址不够用的问题 并且隐藏和保护网络内部主机 避免来自外部的攻击。

**TTL是什么？有什么用？**

表示一个数据报的最大生存时间 每个数据报文在路由器转发一次后 TTL都会-1 当TTL=0时 数据报死亡

**传输层和网络层协议的区别？**

传输层实现了进程之间的逻辑网络通信 但是网络层实现了主机之间的逻辑网络通信

**OSI七层模型**

OSI 将计算机网络体系结构划分为七层，每一层实现各自的功能和协议，并完成与相邻层的接口通信。OSI 的服务定义详细说明了各层所提供的服务。

① 应用层

应用层位于 OSI 参考模型的第七层，其作用是通过应用程序间的交互来完成特定的网络应用。

② 表示层

表示层的作用是使通信的应用程序能够解释交换数据的含义

③ 会话层

会话层就是负责建立、管理和终止表示层实体之间的通信会话

④ 传输层

传输层的主要任务是为两台主机进程之间的通信提供服务。

⑤ 网络层

网络层的主要任务就是选择合适的网间路由和交换节点，确保数据按时成功传送。

⑥ 数据链路层

数据链路层通常也叫做链路层，在物理层和网络层之间。两台主机之间的数据传输，总是在一段一段的链路上传送的，这就需要使用专门的链路层协议。

⑦ 物理层

作为 OSI 参考模型中最低的一层，物理层的作用是实现计算机节点之间比特流的透明传送，尽可能屏蔽掉具体传输介质和物理设备的差异。

**OSI 模型和 TCP/IP 模型异同比较**

相同点

① OSI 参考模型与 TCP/IP 参考模型都采用了层次结构。

② 都能够提供面向连接和无连接两种通信服务机制。

不同点

② TCP/IP 参考模型没有对网络接口层进行细分，只是一些概念性的描述； OSI 参考模型对服务和协议做了明确的区分。

③ OSI 先有模型，后有协议规范，适合于描述各种网络；TCP/IP 是先有协议集然后建立模型，不适用于非 TCP/IP 网络。

**I/O多路复用（multiplexing**

的本质是通过一种机制（系统内核缓冲I/O数据），让单个进程可以监视多个文件描述符，一旦某个描述符就绪（一般是读就绪或写就绪），能够通知程序进行相应的读写操作

select、poll 和 epoll 都是 Linux API 提供的 IO 复用方式。

而select、poll、epoll本质上也都是同步I/O，因为他们都需要在读写事件就绪后自己负责进行读写，也就是说这个读写过程是阻塞的。

I/O多路复用技术的最大优势是系统开销小，系统不必创建进程/线程，也不必维护这些进程/线程，从而大大减小了系统的开销。

**select运行机制**

select()的机制中提供一种fd\_set的数据结构，实际上是一个long类型的数组，每一个数组元素都能与一打开的文件句柄（不管是Socket句柄,还是其他文件或命名管道或设备句柄）建立联系，建立联系的工作由程序员完成，当调用select()时，由内核根据IO状态修改fd\_set的内容，由此来通知执行了select()的进程哪一Socket或文件可读。

**Poll**

poll的机制与select类似，与select在本质上没有多大差别，管理多个描述符也是进行轮询，根据描述符的状态进行处理，但是poll没有最大文件描述符数量的限制。

**Epoll**

epoll在Linux2.6内核正式提出，是基于事件驱动的I/O方式，相对于select来说，epoll没有描述符个数限制，使用一个文件描述符管理多个描述符，将用户关心的文件描述符的事件存放到内核的一个事件表中，这样在用户空间和内核空间的copy只需一次。

**select,poll,epoll的区别：**

|  |  |  |  |
| --- | --- | --- | --- |
|  | select | poll | epoll |
| 操作方式 | 遍历 | 遍历 | 回调 |
| 底层实现 | 数组 | 链表 | 红黑树 |
| IO效率 | 每次调用都进行线性遍历，时间复杂度为O(n) | 每次调用都进行线性遍历，时间复杂度为O(n) | 事件通知方式，每当fd就绪，系统注册的回调函数就会被调用，将就绪fd放到readyList里面，时间复杂度O(1) |
| 最大连接数 | 1024(x86) | 无上限 | 无上限 |

**原型模式（Prototype）**

Intent

使用原型实例指定要创建对象的类型，通过复制这个原型来创建新对象。

**工厂方法（Factory Method）**

Intent

定义了一个创建对象的接口，但由子类决定要实例化哪个类。工厂方法把实例化操作推迟到子类。

Class Diagram

在简单工厂中，创建对象的是另一个类，而在工厂方法中，是由子类来创建对象。

**观察者（Observer）**

Intent

定义对象之间的一对多依赖，当一个对象状态改变时，它的所有依赖都会收到通知并且自动更新状态。

主题（Subject）是被观察的对象，而其所有依赖者（Observer）称为观察者。

Class Diagram

主题（Subject）具有注册和移除观察者、并通知所有观察者的功能，主题是通过维护一张观察者列表来实现这些操作的。

观察者（Observer）的注册功能需要调用主题的 registerObserver() 方法。

**哈希表 原理 冲突解决方法**

散列冲突：不同的关键字经过散列函数的计算得到了相同的散列地址。

1）开放定址法：

这种方法也称再散列法，其基本思想是：当关键字key的哈希地址p=H（key）出现冲突时，以p为基础，产生另一个哈希地址p1，如果p1仍然冲突，再以p为基础，产生另一个哈希地址p2，…，直到找出一个不冲突的哈希地址pi ，将相应元素存入其中。

2) 再哈希法

这种方法是同时构造多个不同的哈希函数

3）链地址法

这种方法的基本思想是将所有哈希地址为i的元素构成一个称为同义词链的单链表，并将单链表的头指针存在哈希表的第i个单元中，因而查找、插入和删除主要在同义词链中进行。

4) 建立公共溢出区

将哈希表分为基本表和溢出表两部分，凡是和基本表发生冲突的元素，一律填入溢出表

**冒泡排序（Bubble Sort）On^2 O1 稳定**

比较相邻的元素。如果第一个比第二个大，就交换它们两个；

对每一对相邻元素作同样的工作，从开始第一对到结尾的最后一对，这样在最后的元素应该会是最大的数；

针对所有的元素重复以上的步骤，除了最后一个；

**选择排序（Selection Sort）On^2 O1 不稳定**

工作原理：首先在未排序序列中找到最小（大）元素，存放到排序序列的起始位置，然后，再从剩余未排序元素中继续寻找最小（大）元素，然后放到已排序序列的末尾。以此类推，直到所有元素均排序完毕。

**归并排序（Merge Sort）Onlogn On 稳定**

将已有序的子序列合并，得到完全有序的序列；即先使每个子序列有序，再使子序列段间有序

把长度为n的输入序列分成两个长度为n/2的子序列；

对这两个子序列分别采用归并排序；

将两个排序好的子序列合并成一个最终的排序序列。

**快速排序 Onlogn Onlogn 不稳定**

**堆排序 O(nlogn)，O1就地排序 不稳定**

将待排序序列构造成一个大顶堆，此时，整个序列的最大值就是堆顶的根节点。将其与末尾元素进行交换，此时末尾就为最大值。然后将剩余n-1个元素重新构造成一个堆，这样会得到n个元素的次小值。如此反复执行，便能得到一个有序序列了

步骤一 构造初始堆。将给定无序序列构造成一个大顶堆

步骤二 将堆顶元素与末尾元素进行交换，使末尾元素最大。然后继续调整堆，再将堆顶元素与末尾元素交换，得到第二大元素。

如此反复进行交换、重建、交换。