计算机网络：

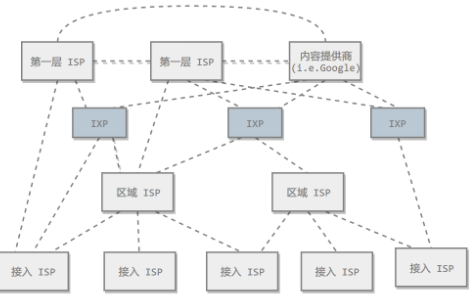
**互联网与网络**：

网络是把主机连接起来，而互联网是把多种不同的网络连接起来，互联网是网络的网络，是全球范围的互联网。

**ISP互联网服务提供商:**

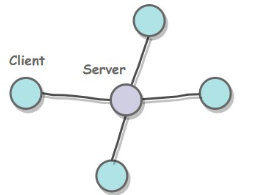
ISP从互联网管理机构获得许多的IP地址,同时拥有通信线路及路由器等联网设备,个人或者机构向ISP缴纳一定的费用就可以连入互联网；

目前采用的是一种多层次的ISP,ISP根据覆盖面积的大小分为第一层ISP,区域ISP和接入ISP.互联网交换点IXP运行两个ISP直接相连而不用经过第三个ISP。

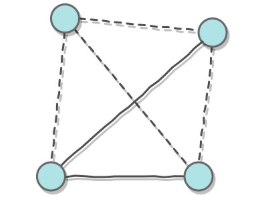


**主机间的通信**：

(1)客户-服务器(C/S):客户是服务的请求方,服务器是服务的提供方



(2)对等P2P:不区分客户和服务器.



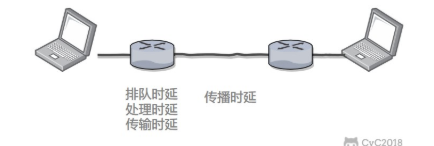
**电路交换和分组交换**：

(1)电路交换:电路交换用于电话通信系统,两个用户要通信之前需要建立一条专用的物理链路,并且在整个通信过程中始终占用该链路.由于通信的过程中不可能一直在使用传输线路,因此电路交换对线路的利用率很低,不到10%.

(2)分组交换:每个分组都有首部和尾部,包含源地址和目的地址等控制信息,在同一个传输线路上同时传输多个互相分组互相不会影响,因此同一条传输线路允许同时传输多个分组,也就是分组交换不需要占用传输路线.在一个邮局通信系统中,邮局收到一份邮件之后,先存储下来,然后把相同目的地的邮件一起转发给下一个目标,这个过程就是存储转发,分组交换也使用了存储转发过程

**时延**：

(1)总的时延=排队时延+处理时延+传输时延+传播时延



(2)排队时延:分组在路由器的输入队列和输出队列中排队等待的时间,取决于网络当前的通信量

(3)处理时延:主机或者路由器受到分组时进行处理所需要的时间,例如分析首部/从分组中提取数据,进行差错检验或者查找适当的路由等.

(4)传输时延:主机或者路由器传输数据帧所需要的时间



l表示的是数据帧的长度,v表示的是传输速率

(5)传播时延:电磁波在信道中传播所需要的花费的时间,电磁波传播速度接近光速:



l为线路长度,v为电磁波在信道中的传播速度

**计算机网络的体系结构**：



(1)应用层:为特定程序提供数据传输服务,例如http/dns协议.数据单位为报文

(2)传输层:为进程提供通用数据传输服务.由于应用层的协议很多,定义通用的传输层协议就可以支持不断增多的应用层协议.传输层包括两种协议:

传输控制协议TCP:提供面向连接/可靠数据传输服务/数据单位为报文段;

用户数据报协议UDP:提供无连接/尽最大努力数据传输服务,数据单位为用户数据报.

TCP主要是提供完整性服务,UDP主要是提供及时性服务.

//OSI模型

(1)表示层:数据压缩,加密及数据描述,使得应用程序不必关心各个主机中数据内部格式不同的问题

(2)会话层:建立和管理会话

//数据在各层之间的传递

(1)在向下传递的过程中,需要添加下层协议所需要的首部和尾部,在向上传递的过程需要不断拆开首部和尾部

(2)路由器是三层设备,工作在网络层,不需要为进程或者应用程序提供服务.

**物理层：**

//通信方式:

(1)单工通信:单向传输

(2)半双工通信:双向交替传输

(3)全双工通信:双向同时传输

//带通调制

模拟信号是连续信号,数字信号是离散信号,带通调制把数字信号转成模拟信号.分为:调幅和调相

**数据链路层：**

(1)基本问题

(2)信道分类

(3)信道复用技术

(4)CSMD/CD协议

(5)PPP协议

(6)MAC地址

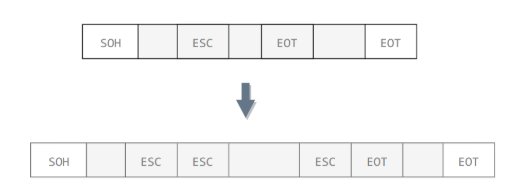
(7)局域网

(8)以太网

(9)交换机

(10)虚拟局域网

**基本问题**：

(1)封装成帧  
将网络层传输下来的分组添加首部和尾部,用于标记帧的开始和结束  
file://c:\users\admini~1\appdata\local\temp\tmplhfp_m\1.png  
(2)透明传输  
透明表示一个实际存在的事物好像不存在一样  
帧使用首部和尾部进行定界,如果帧的数据部分含有和尾部首部相同的内容,那么帧的开始和结束有可能造成误判;需要在数据部分出现首部和尾部相同内容的地方插入转义字符,如果数据部分出现转义字符,那么就在转义字符前面再加一个转义字符.在接收端处理的时候可以还原成原数据.透明传输的内容是转义字符,用户感觉不到转移字符的存在.  
  
(3)差错检测  
循环冗余检验(CRC)来检查比特差错

**信道分类**：

(1)广播信道

一对多通信,一个节点发送的数据能够被广播信道上所有节点收到;所有节点都在同一个广播信道上发送数据,因此需要有专门的控制方法进行协调,避免发生冲突

主要有两种控制方法:一个是信道复用技术;另外是CSMA/CD协议

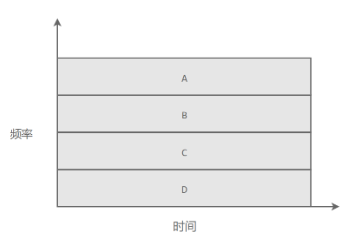
(2)点对点信道

一对一通信,因为不会发生碰撞,比较简单,使用PPP协议进行控制

**信道复用技术：**

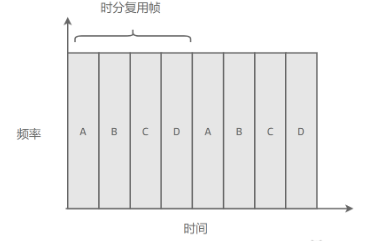
(1)频分复用

所有主机在相同的时间占用不同频率带宽资源



(2)时分复用

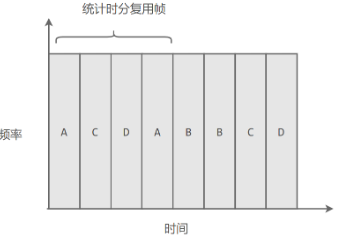
所有主机在不同时间占用相同频率的频率带宽资源



使用频分复用和时分复用的时候,在通信过程中主机会一直占用一部分信道资源,由于计算机数据的突发性质,通信过程中没必要一直占着信道资源不让出给其他用户使用,因此这两种方式对信道的利用率都不高

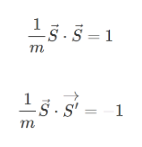
(3)统计时分复用

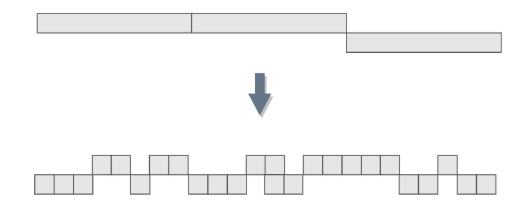
对时分复用的改进,不固定内个用户在时分复用帧中的位置,只要有数据就集中起来组成统计时分复用帧然后发送.



(4)波分复用

光的频分复用,由于光的频率很高,习惯上用波长来表示光载波

(5)码分复用  
为每个用户分配m bit的码片,并且所有的码片正交,对任意两个码片file://c:\users\admini~1\appdata\local\temp\tmplhfp_m\1.png有:  
file://c:\users\admini~1\appdata\local\temp\tmplhfp_m\2.png  
为了讨论方便,取m=8,设码片s为00011011.在拥有该码片的用户发送比特1时就发送该码片,发送比特0时就发送该码片的反码11100100.  
在计算的时候将00011011记作(-1 -1 -1 +1 +1 -1 +1 +1)可以得到  
  
file://c:\users\admini~1\appdata\local\temp\tmplhfp_m\4.png  
利用上式可知,当接收端使用码片S对接收到的数据进行内积运算的时候,结果为0的是其他用户发送的数据,结果为1的是用户发送的比特1,结果为-1的是用户发送的比特0.  
码分复用需要发送的数据量为原先的m倍.



**CSMA/CD表示载波监听多点接入/碰撞检测:**

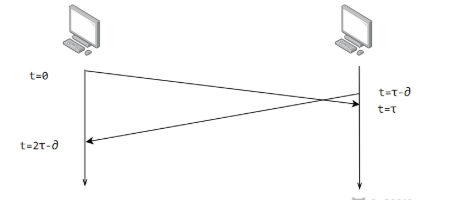
(1)多点接入:说明这是总线型网络,许多主机以多点的方式连接到总线上

(2)载波监听:每个主机不停的监听信道,在发送前,如果监听到信道正在使用,就必须等到.

(3)碰撞检测:在发送中,如果监听到信道已经有其他主机正在发送数据,就发生了碰撞,虽然每个主机发送数据之前都已经监听到信道为空闲,但是由于电磁波传播的时延的存在,还是有可能发生碰撞

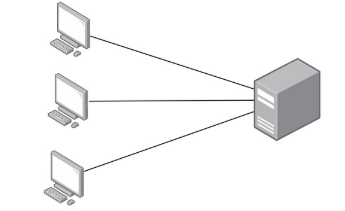
记端到端的传播时延为t,最先发送的站点最多经过2t就可以知道就可以知道是否发送了碰撞,称2t为争用期.只有经过争用期之后还没有检测到碰撞,才肯定这次发送不会碰撞,

当发生碰撞时,站点要停止发送,等待一段时间在发送.这个时间采用截断二进制指数避让算法来确定.从离散的整数集合{0,1,2...(2k-1)}中随机取出一个数,记作r,然后取r倍的争用期作为重传等待时间.



**PPP协议:**

//互联网用户需要连接到某个ISP之后才能连接到互联网,PPP协议是用户计算机和ISP进行通信时所使用的数据链路层协议



PPP帧格式:

(1)F字段为帧的界定符

(2)A和C字段暂时没有意义

(3)FCS字段是用于CRC检验的

(4)信息部分长度不超过1500

file://c:\users\admini~1\appdata\local\temp\tmplhfp_m\1.png

**MAC地址:**

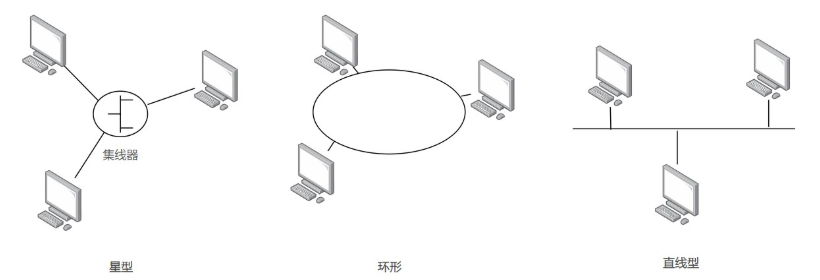
MAC地址是链路层地址,长度为6个字节(48位),用于唯一标示网络适配器(网卡).一台主机拥有多个网络适配器就有多个MAC地址.

**局域网**:

局域网是一种典型的广播信道,主要特点:

(1)网络为一个单位所拥有,且地理范围和站点数目均有限

主要有以太网,令牌环网,FDDI和ATM等局域网技术,目前以太网占领有限局域网市场.可以按照网络拓补结构对局域网划分.



**虚拟局域网**:

虚拟局域网可以建立与物理位置无关的逻辑组,只有在同一个虚拟局域网中的成员才会收到链路层广播信息.

如下图中(A1,A2,A3,A4)属于同一个局域网,A1发送的广播会被A2,A3,A4收到,而其他站点收不到.

使用Vlan干线建立的局域网,每台交换机上的一个特殊接口被设置成干线接口,以互连vlan交换机.IEEE定义了一种扩展的以太网帧格式802.1Q,它以标准的以太网加上4字节首部Vlan标签,用于表示该帧属于哪一个虚拟局域网.



**以太网**:

//以太网是一种星型拓补结构局域网

早期使用集线器连接,集线器是一种物理层设备,作用于比特而不是帧,当一个比特到达接口时,集线器重新生成这个这个比特,并将其能量强度放大,从而扩大网络传输距离,之后再将这个比特发送给其他所有接口,如果集线器同时受到两个不同接口的帧,就发生了碰撞

目前使用交换机代替集线器,交换机是一种链路层设备,不会发生碰撞,能够根据mac地址进行存储转发

以太网帧格式:

(1)类型:标记上层使用的协议

(2)数据:长度在46-1500之间,如果太小则需要填充;

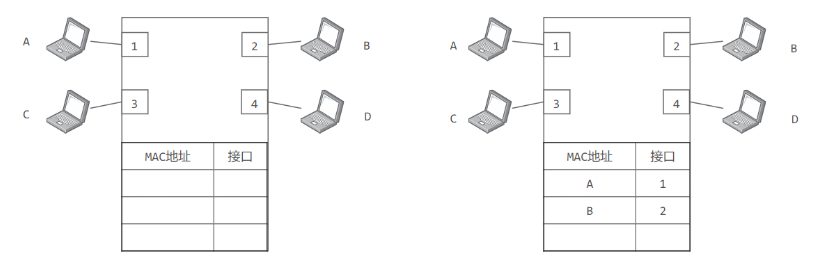
(3)FCS:帧检验序列,使用的是CRC检验



**交换机:**

//交换机具有自学习能力,学习的目的是交换表的内容,交换表中存储的mac地址到接口的映射.

正是这种自学习的能力,因此交换机是一种即插即用的设备,不需要网络管理员手动配置交换表的内容.



上图中,交换机有4个接口,主机A向B发送数据帧时,交换机把主机A到接口1的映射写入交换表中.为了使得数据发送到B,先查交换表,此时没有B的地址,那么A就发送广播帧,主机C和D会丢弃该帧,主机B回应该帧向主机A发送数据包时,交换机查找交换表得到主机A的映射接口为1,就发送数据帧到接口1,同时交换机添加主机B到接口2的映射.

**网路层：**

//网络层是整个互联网的核心,网络层向上只提供简单灵活的,无连接的,尽最大努力交互的数据服务.

使用IP协议可以把异构的物理网络连接起来,使得网络看起来像一个统一的网络

与IP协议配套的有三个协议

(1)地址解析协议ARP

(2)网际控制报文协议ICMP

(3)网际组管理协议IGMP

**IP协议**：

//IP数据报格式

(1)版本:IPV4/IPV6

(2)首部长度:占4位,因此最大值为15,值为1表示是1个32位字的长度,也就是4个字节.因为固定部分长度为20字节,因此该值最小为5.如果可选字段的长度不是4字节的整数倍,就用尾部的填充部分来填充

(3)区分服务:用于获取更好的服务,一般情况下不使用

(3)总长度:包括首部长度和数据部分长度

(4)生存时间:TTL,存在的意义为了防止无法交付的数据在互联网中不断传输,以路由器跳数为单位,当TTL为0就将数据报丢弃.

(5)协议:指出携带的数据应该上交给那个协议进行处理,例如ICMP,TCP,UDP等

(6)首部检验和:因为数据报每经过一个路由器,都需要重新计算检验和,因此检验和不包含数据部分可以减少计算的工作量

(7)标识:在数据报长度过长从而发生分片的情况下,相同数据报不同的分片具有相同的标识符

(8)片偏移:和标识符一起,用于发生分片的情况.片偏移的单位为8字节.

//IP地址编址方式

(1)分类

IP地址由两部分组成:网路号和主机号

(2)子网划分:

(3)无分类

**地址解析协议ARP**：

//网络层实现主机之间的通信,链路层实现每段链路之间的通信.因此在通信过程中IP数据报中的源地址和目的地址是不会改变的,MAC地址随着链路改变而改变

ARP实现的是由IP地址得到MAC地址；每个主机都有一个ARP高速缓存,里面有本局域网上的各个主机和路由器和路由器的IP地址到MAC地址的映射表.

如果主机A直到主机B的IP地址,但是ARP高速缓存中没有该IP地址到mac地址的映射,此时主机A通过广播的方式发送ARP请求分组,主机B受到请求之后会发送ARP响应分组给A告知其MAC地址,随后主机A向高速缓存中写入主机B的IP地址到MAC地址的映射.

**网际控制报文协议ICMP：**

//ICMP是为了更有效的转发IP数据报和提高交付成功的机会.其封装在IP数据报中,但不属于高层协议.

ICMP报文分为:差错报文和询问报文.

(1)ping

ping是ICMP的一个重要应用,用于测试两台主机的联通性

原理是通过向目的主机发送ICMP Echo请求报文,目的主机受到之后会发送Echo回答报文,ping会根据时间和成功响应的次数估算出数据报往返时间以及丢包率

(2)Traceroute

traceroute是ICMP的另一个应用,用来跟踪一个分组从源到终点的路径

traceroute发送的ip数据报封装是无法交付的UDP用户数据包,并由目的主机发送终点不可达差错报告报文.

源主机项目的主机发送一连串的IP数据报,第一个数据报P1生存时间TTL设置为1,当P1到达路径上第一个路由器R1时,收下,此时TTL等于0,此路由器将数据报丢弃,并向原主机发送一个ICMP时间超过差错报告报文

源主机接着发送第二个数据包P2,将TTL设置成2,类似上面.

不断执行该操作,直到最后一个数据报刚刚达到目的主机,主机不转发数据报,也不把TTL的值减少1.但是因为数据报封装的是无法交付的UDP,因此目的主机向源主机发送ICMP终点不可达差错报文报告.

之后源主机直到目的主机所经过的路由器的IP地址以及每个路由器的往返时间.

**虚拟专用网VPN**：

//虚拟专用网VPN

由于IP地址的紧缺,机构内的计算机可以使用仅在本地有效的IP地址(专用网络).

有三个专用地址块:

(1)10.0.0.0~10.255.255.255

(2)172.16.0.0~172.31.255.255

(3)192.168.0.0~192.168.255.255

VPN使用公用的互联网作为嗯机构的专用网之间的通信载体.专用指机构内主机只与本机构内的其他主机通信;虚拟指的是好像是,而实际上并不是,他有经过公用的互联网

场所 A 和 B 的通信经过互联网，如果场所 A 的主机 X 要和另一个场所 B 的主机 Y 通信，IP 数据报的源地址是 10.1.0.1，目的地址是 10.2.0.3。数据报先发送到与互联网相连的路由器 R1，R1 对内部数据进行加密，然后重新加上数据报的首部，源地址是路由器 R1 的全球地址 125.1.2.3，目的地址是路由器 R2 的全球地址 194.4.5.6。路由器 R2 收到数据报后将数据部分进行解密，恢复原来的数据报，此时目的地址为 10.2.0.3，就交付给 Y。

**网络地址转换NAT：**

专用网络内部主机使用IP地址想和互联网上的主机通信,使用NAT将本地IP转换成全球IP;

在以前,NAT将本地IP和全球IP一一对应,这种方式下拥有n个全球IP地址的专用网内最多同时拥有n台主机连入互联网.现在常用NAT转换表将传输层的端口号用上,使得多个专用网络内部主机共用一个全球IP,使用端口号的NAT也叫作网络地址与端口转换NAPT.

**路由器：**

//路由器的结构

(1)功能:路由选择与分组转发

(2)分组转发三部分构成:交换结构,一组输入端口,一组输出端口

//路由器分组转发流程

(1)从数据报首部提取目的主机的IP地址,得到目的网络地址N

(2)若N就是与此路由器直接连接的某个网络地址,则进行直接交付

(3)若路由表中有目的地址D的特定主机路由,则把数据报传送给表中所指明的下一步路由

(4)若路由表中有到达网络N的路由,则把数据报传送给路由表中所指明的下一跳路由;

(5)若路由表中有一个默认的路由,则把数据报给表中默认的路由

(6)报告转发分组出错

//路由选择协议

路由选择是自适应的,能随着网络通信量和拓扑结构的变化而自适应地进行相应的调整.

互联网可以划分成许多较小的自治系统AS,一个AS可以使用一种和别的AS不同的选择协议

可以把路由选择协议分成两大类

(1)自治系统内部路由的选择:RIP和OSPF

(2)自治系统间的路由选择:BGP

**内部网关协议RIP：**

RIP 是一种基于距离向量的路由选择协议。距离是指跳数，直接相连的路由器跳数为 1。跳数最多为 15，超过 15 表示不可达.

RIP 按固定的时间间隔仅和相邻路由器交换自己的路由表，经过若干次交换之后，所有路由器最终会知道到达本自治系统中任何一个网络的最短距离和下一跳路由器地址。

距离向量算法：

(1) 对地址为 X 的相邻路由器发来的 RIP 报文，先修改报文中的所有项目，把下一跳字段中的地址改为 X，并把所有的距离字段加 1；

(2) 对修改后的 RIP 报文中的每一个项目，进行以下步骤：

(3)若原来的路由表中没有目的网络 N，则把该项目添加到路由表中；

(4) 否则：若下一跳路由器地址是 X，则把收到的项目替换原来路由表中的项目；否则：若收到的项目中的距离 d 小于路由表中的距离，则进行更新（例如原始路由表项为 Net2, 5, P，新表项为 Net2, 4, X，则更新）；否则什么也不做。

(5)若 3 分钟还没有收到相邻路由器的更新路由表，则把该相邻路由器标为不可达，即把距离置为 16。

RIP 协议实现简单，开销小。但是 RIP 能使用的最大距离为 15，限制了网络的规模。并且当网络出现故障时，要经过比较长的时间才能将此消息传送到所有路由器。

**内部网关协议OSPF：**

//开放最短路径优先OSPF,为了克服RIP缺点

//开放表示 OSPF 不受某一家厂商控制，而是公开发表的；最短路径优先表示使用了 Dijkstra 提出的最短路径算法 SPF。

//OSPF特点

(1)向本治系统中所有路由器发送消息,采用洪泛法

(2)发送的信息就是相邻路由器的链路状态,链路状态包括与那些路由器相连以及链路的度量,度量的费用,距离,时延,带宽来表示

(3)只有当链路状态发生变化时,路由器才会发送信息

所有路由器都具有全网拓扑结构图,并且是一致的,相比于RIP,OSPF的更新过程收敛的很快

**外部网关协议BGP**：

//BGP边界网关协议

AS之间的路由选择很困难,主要由于:

(1)互联网规模很大

(2)各个AS内部使用不同的路由选择协议,无法准确定义路径的度量

(3)AS之间的路由选择必须考虑有关的策略,比如有些AS不愿意让其他AS经过.

BGP只能寻找一条比较好的路由,而不是最佳路由

每个AS都必须配置BGP发言人,通过两个相邻BGP发言人之间建立的TCP连接来交换路由信息.

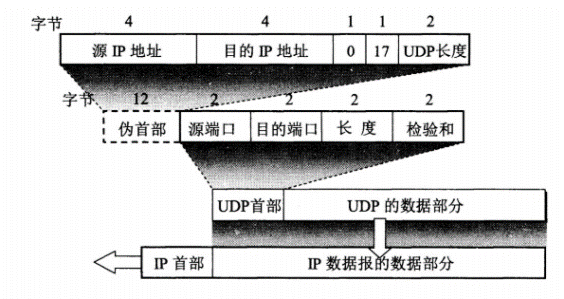
**传输层：**

//网络层只是把分组发送到目的地址，但是真正通信的并不是主句而是主机中的进程，传输层提供了进程间的逻辑通信，传输层向高层用户屏蔽了下面网络层的核心细节，使得应用程序看起来像是在两个传输层实体之间有一条端到端的逻辑通信信道。

**UDP与TCP：**

//用户数据报协议UDP:是无连接的,尽可能最大交付,没有拥塞控制,面向报文(对于应用程序传下来的报文不合并也不拆分,只是添加UDP首部),支持一对一,一对多,多对一和多对多的通信

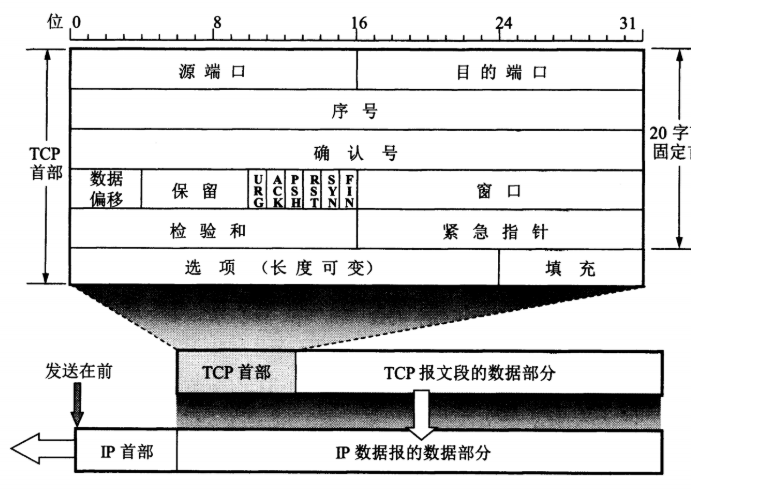
UDP首部格式:



首部字段只有8个字节,包括源端口,目的端口,长度,检验和,12字节的伪首部是为了计算检验和临时添加的

//传输控制协议TCP:是面向连接的,提供可靠交付,有流量控制,拥塞控制,提供全双工通信,面向字节流,(把应用层传输的报文当做字节流,把字节流组织成大小不等的数据块),每一条TCP连接都是点对点的(一对一).

TCP首部格式:



序号:用于对字节流进行编号

确认号:期望收到下一个报文段的序号,例如B正确收到A发来的一个报文段,序号为501,携带的数据长度为200,因此B期望的下一个报文段的序号为701,B发送给A的确认报文段中确认号就是701;

数据偏移:指的是数据部分距离报文段起始处的偏移量,实际上指的就是首部长度

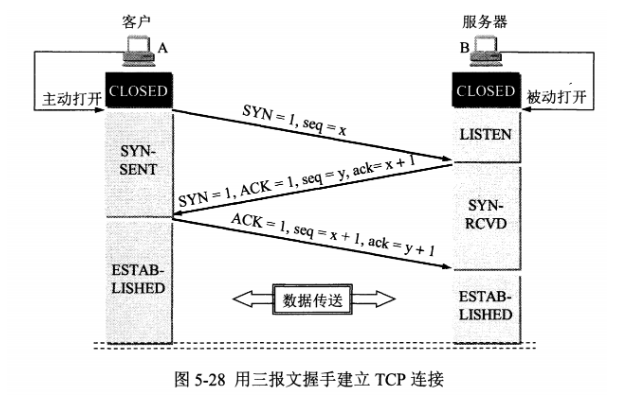
确认ACK:当ACK=1的时候确认字段有效,否则无效.TCP规定,在建立连接之后多有传送的报文段必须把ACK置为1;

同步SYN:在连接建立时用来同步序号.当SYN=1,ACK=0表示这是一各连接请求报文段,若对方统一建立,则响应报文为SYN=1,ACK=1;

终止FIM:用来释放一个连接,当FIN=1,表示此报文段发送方的数据已经发送完毕,并要求释放连接

窗口:窗口值作为接收方让发送方设置其发送窗口的依据,之所以需要这个,是因为接收方的数据缓存空间是有限的.

**TCP的三次握手**：



假设A为客户端,B为服务器端

(1)首先B处于监听状态,等待客户的连接请求

(2)A向B发送连接请求报文,SYN=1,ACK=0,选择一个初始的序号x.

(3)B收到了连接请求报文,如果同意连接,则向A发送连接确认报文,SYN=1,SCK=1,确认号为x+1;,同时也选择一个初始的序号

(4)A收到B的连接确认信号之后,还要向B发出确认,确认号为y+1,序号为x+1;

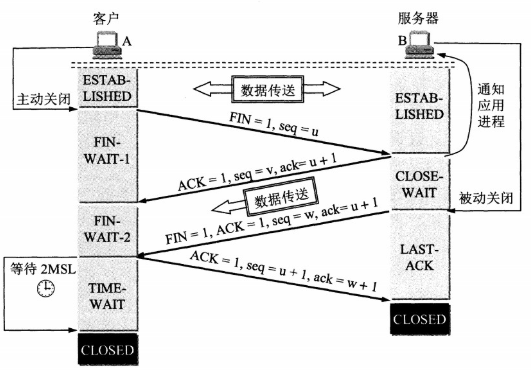
(5)B收到A的确认之后,连接建立

原因:

第三次握手的原因是防止失效的连接请求达到服务器,让服务器错误打开连接

客户端发送请求如果在网络中滞留,那么就会隔很长一段时间才能收到服务器端发挥的连接确认.客户端等待第一个超时重传的时间之后,就会重新请求连接.但是这个滞留的连接请求最后还是会到达服务器,如果不进行第三次握手,那么服务器会打开两个连接.如果有第三次握手,客户端会忽略之后发送的对滞留连接请求的确认,不进行第三次握手,就不会打开连接.

**TCP的四次挥手**：



(1)A发送连接释放报文,FIN=1;

(2)B收到之后发出确认,此时TCP处于半关闭状态,B能向A发送数据,但是A不能向B发送数据

(3)当B不需要连接的时候,发送释放报文,FIN=1;

(4)A收到之后发出确认,进入TIME-WAIT状态,等待2MSL(最大报文存活时间)后释放连接

(5)B收到A的确认之后释放连接

四次挥手释放的原因:

客户端发送了FIN连接释放报文之后,服务器受到了这个报文,就进入CLOSE-WAIT状态,这个状态是为了让服务器继续发送未发送完的报文,传送完毕之后,服务器发送FIN释放报文

**TIME\_WAIT**

客户端接收到服务器FIN报文之后进入这个状态,此时并不是直接进入CLOSED状态,还需要等待2MSL时间:

(1)确保最后一个报文能够到达.如果B没收到A发送来的确认报文,那么就会重新发送连接释放请求,A等待时间处理这种情况

(2)为了让本连接持续时间内产生的所有报文在网络中消失,使得下一个连接不会出现旧的连接请求报文.

**TCP可靠传输(需要补充)：**

//TCP利用超时重传来实现可靠传输:如果一个已经发送的报文在超时时间内没有收到确认,那么就重传这个报文段.

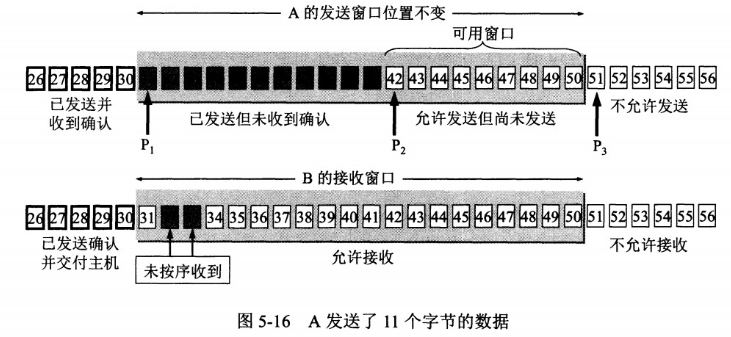
一个报文段从发送再到接收确认所经过的时间称为往返时间RTT,加权平均往返时间RTTS计算如下:

**TCP滑动窗口**：

窗口是缓存的一部分,用来暂时存放字节流.发送端和接收端各有一个窗口,接收方通过TCP报文段中窗口字段高速发送端自己的窗口大小,发送端根据这个值来设置自己的窗口大小

发送窗口内的字节都允许被发送,接收窗口内的字节都被允许接收.如果发送窗口左部的字节已经发送并且收到了确认,那么就将发送窗口右滑动一定距离,直到左部第一个字节不是已发送并且已确认状态;接收窗口的滑动类似,接收窗口左部字节已经发送确认并交付主机,就向右滑动窗口.

接收窗口只会对窗口内最后一个按序到大的字节进行确认,例如接收窗口接收到字节{31,34,35},其中{31}按序达到,{34,35}就不是,因此只对{31}进行确认.发送方得到第一个字节的确认之后,就知道这个字节之前的所有字节都被接收了.



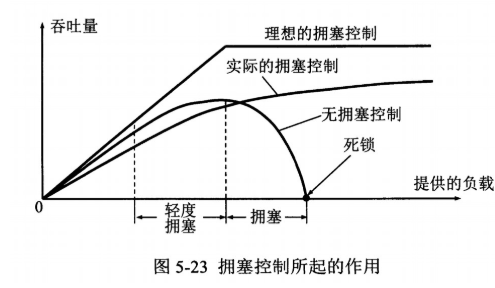
**TCP流量控制：**

流量控制为了控制发送方发送的速率,保证接收方来的及接收

接收方发送的确认报文中的窗口字段可以用来控制发送窗口的大小,从而影响其发送的速率,将窗口字段设置为0,则发送方不能发送数据.

**拥塞控制**：

//如果网络出现了拥塞,分组将会丢失,此时发送方会继续重传,从而导致网络拥塞程度更高.因此,当拥塞出现的时候,应当控制发送方的速率,这一点和流量控制很像,但是出发点不同.流量控制是为了让接收方来得及接收,而拥塞控制是为了降低整个网络的拥塞程度。



TCP主要通过四个算法来进行拥塞控制

(1)慢开始

(2)拥塞避免

(3)快重传

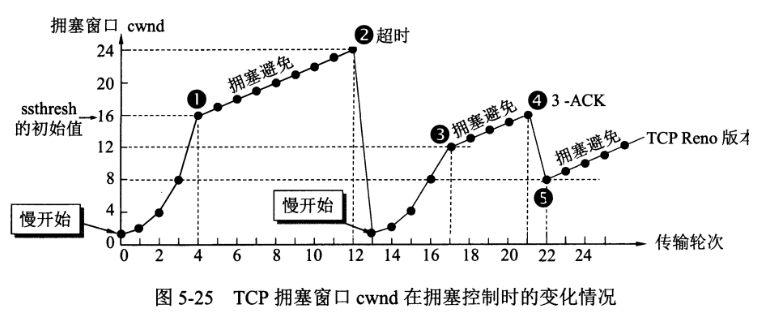
(4)块恢复

发送方需要维护一个叫做拥塞窗口的状态变量,注意拥塞窗口与发送窗口的区别:拥塞窗口只是一个状态变量,实际决定发送方能够发送多少数据的是发送方的窗口

做出如下假设:

(1)接收方有足够大的接受缓存,因此不会发生流量控制

(2)虽然TCP的窗口基于字节,但是这里窗口的大小单位是报文段



1 慢开始与拥塞

发送的最初执行慢开始,令cwnd=1,发送方只能发送一个报文段;当收到确认之后,将cwnd加倍,因此之后能发送2,4,8,

注意慢开始每次都要将cwnd加倍,这样会让cwnd增长速度非常快,从而使得发送方发送的速度增长的特别快,网络拥塞的可能性更高.设置一个阈值,当cwnd超过阈值之后,进入拥塞避免,每个轮次只将cwnd+1.

如果出现了超时,则令阈值ssthresh=cwnd/2,然后重新开始慢执行.

2 快重传与快恢复

在接收方,要求每次接受到报文段都应该对最后一个收到的有序报文进行确认.例如已经接收到了M1和M2,此时接收到M4,应当立即发送M2的确认.

在发送方,如果收到三个重复确认,那么可以知道下一个报文段丢失,此时执行快重传,立即重传下一个报文段.例如收到三个M2,则M3丢失,立即重传M3.

在这种情况下,只是丢失了个别报文段,而不是网络拥塞.因此执行快速恢复,令ssthresh=cwnd/2,cwnd=ssthresh,此时直接进入拥塞避免.

慢开始和快恢复的快慢指的是cwnd的设置值,而不是cwnd的增长速度,慢开式初始cwnd=1;块恢复cwnd=sstresh.

**应用层：**

• 域名系统

• 文件传送协议

• 动态主机配置协议

• 远程登录协议

• 电子邮件协议• 1. SMTP

• 2. POP3

• 3. IMAP

常用端口

• Web 页面请求过程• 1. DHCP 配置主机信息

• 2. ARP 解析 MAC 地址

• 3. DNS 解析域名

• 4. HTTP 请求页面

**域名系统DNS：**

//DNS是一个分布式的数据库,提供了主机名和IP地址之间相互转换的服务.这里分布式数据库指的是每个站点只保留他自己的那部分数据

域名具有层次结构,从上到下依次为:根域名,顶级域名,二级域名

DNS可以使用UDP或者TCP进行传输,使用的端口号为53.大多数情况下DNS使用的是UDP传输,这就要求域名解析服务器和域名服务器都必须自己处理超时和重传从而保证可靠性.在两种情况下使用TCP传输

(1)如果返回的响应超过512字节(UDP最大支持512字节的传输)

(2)区域传送(区域传送是主域名服务器向辅域名服务器传送变化的那部分数据)

**文件传送协议FTP:**

FTP使用TCP进行连接,他需要两个连接来传送一个文件:

(1)控制连接:服务器打开端口号21等待客户端的连接,客户端主动建立连接之后,使用这个连接将客户端的命令传送给服务器,并传回服务器的应答.

(2)数据连接:用来传送一个文件数据.

//根据数据连接是否由服务器主动建立,FTP分为主动和被动两种模式:

(1)主动模式:服务器端主动建立数据连接,其中服务器端的端口号为20,客户端的端口号随机,但是必须大于1024,因为0~1023都是熟知端口号

(2)被动模式:客户端建立数据连接,其中客户端的端口号由客户端自己指定,服务器的端口随机.

主动模式要求客户端开放端口号给服务器端，需要去配置客户端的防火墙。被动模式只需要服务器端开放端口号即可，无需客户端配置防火墙。但是被动模式会导致服务器端的安全性减弱，因为开放了过多的端口号。

**动态主机配置协议DHCP**:

DHCP提供了即插即用的联网方式,用户不需要手动配置IP地址等信息

DHCP配置的不仅是IP地址,还包括子网掩码,网关IP地址等

DHCP工作过程:

(1)客户端发送Discover报文,该报文目的地址为255.255.255.255:67,源地址为0.0.0.0:68,被放入UDP中,该报文被广播到同一个子网的所有主机上.如果客户端和DHCP服务器不在同一个子网,就需要使用中继代理

(2)DHCP服务器收到Discover报文之后,发送Offer报文给客户端,该报文包含了客户端所需要的信息.因为客户端可能收到多个DHCP服务器的信息,因此客户端需要进行选择.

(3)如果客户端选择某个DHCP服务器提供的信息,那么就发送request报文给DHCP服务器

(4)DHCP服务器发送ACK报文,表示客户端此时可以使用提供给他的信息

**远程登录协议TELNET:**

TELNET用于登录远程主机上,并且远程主机上的输出也会返回

TELNET可以适应许多计算机和操作系统的差异,例如不同操作系统的换行符定义

**电子邮件协议:**

一个电子邮件系统由但部分组成

(1)用户代理

(2)邮件服务器

(3)邮件协议

邮件协议包含发送协议和读取协议,发送协议常用SMTP,读取协议常用POP3和IMAP.

//SMTP:

只能发送ASCII码,而互联网邮件扩充MIME可以发送二进制文件.MIME并没有改动或者取代SMTP,而是增加邮件主体的结构,定义了非ASCII码的编码规则.

//POP3

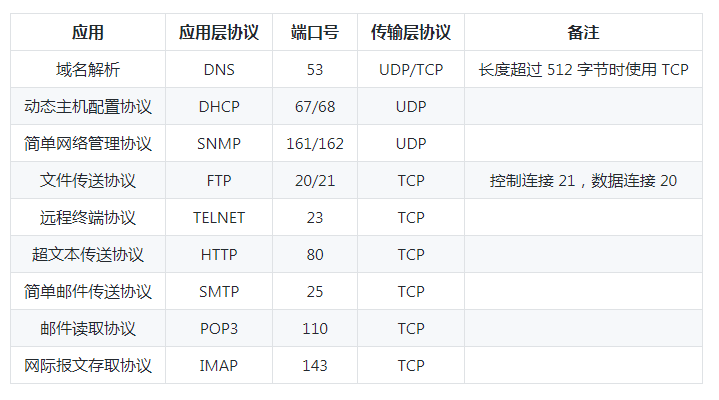
特点是只要用户从服务器上读取了邮件,就把该邮件删除.最新的POP3可以不删除邮件

//IMAP

客户端和服务器上的邮件保持同步,如果不手动删除邮件,那么服务器上的邮件也不会被删除,可以让用户随时随地的去访问服务器的邮件

**常用端口:**

TFTP：简单文本传输协议：69端口，UDP



**Web页面请求过程：**

//1 DHCP配置主机信息

(1)假设主机开始没有IP地址及其他信息,那么就需要先使用DHCP来获取.

(2)主机生成一个DHCP请求报文,并且将报文放入具有目的端口67和源端口68的UDP报文段中

(3)该报文段则被放入在一个具有广播IP目的地址(255.255.255.255)和源IP地址(0.0.0.0)的IP数据报中

(4)该数据包则被放置在MAC帧中,该帧具有MAC地址FF:FF:FF:FF:FF,将广播到与交换机连接的所有设备中

(5)连接交换机的DHCP服务器收到广播帧之后,不断的向上分解IP数据报,UDP报文段,DHCP请求报文,之后生成DHCP ACK报文,该报文包含信息:IP地址,DNS服务器的IP地址,默认网关路由器的IP地址和子网掩码.该报文被放入UDP报文段中,UDP报文段有被放入IP数据报中,最后放入MAC帧中.

(6)该帧的目的地址是请求主机的mac地址,因为交换机的自学习的能力,之前主机发送了广播帧之后就记录了mac地址到其转发接口的交换表中,因此交换机就可以直接指导应该向那个接口发送该数据帧

(7)主机受到该帧之后,不断分解得到DHCP报文,之后配置他的IP地址,子网掩码,DNS服务器的IP地址,并在其IP转发表中默认安装网关.

//2.ARP解析MAC地址

(1)主机通过浏览器生成一个TCP套接字,套接字向HTTP服务器发送HTTP请求.为了生成套接字,主机需要知道网站的域名对应的IP地址

(2)主机生成一个DNS查询报文,该报文具有53号端口,因为DNS服务器的端口号为53.

(3)该DNS查询报文被放入目的地址为DNS服务器IP地址的IP数据报中

(4)该IP数据报被放入一个以太网帧中,该帧将发送到路由器网关

(5)DHCP过程只知道网关路由器的IP地址,为了获取网关路由器的MAC地址,需要使用ARP协议.

(6)主机生成一个包含目的地址为网关路由器IP地址的ARP查询报文,将该ARP查询报文放入一个具有广播地址的以太网帧中,并向交换机发送该以太网帧,交换机向连接的设备群发,包括路由器网关

(7)网关路由器收到该帧后,不断向上分解ARP报文,发现其中IP地址与其接口匹配,因此就发送一个ARP回答报文,包含自己的MAC地址,发送给主机.

//3.DNS解析域名

(1)知道网关路由器的MAC地址之后,就可以继续DNS解析

(2)网关路由器接收到包含DNS查询报文的以太网帧之后,抽取IP数据报,并根据转发表决定该IP数据包应该转发的路由器

(3)因为路由器具有内部网关协议（RIP、OSPF）和外部网关协议（BGP）这两种路由选择协议，因此路由表中已经配置了网关路由器到达 DNS 服务器的路由表项。

(4)到达 DNS 服务器之后，DNS 服务器抽取出 DNS 查询报文，并在 DNS 数据库中查找待解析的域名

(5)找到 DNS 记录之后，发送 DNS 回答报文，将该回答报文放入 UDP 报文段中，然后放入 IP 数据报中，通过路由器反向转发回网关路由器，并经过以太网交换机到达主机

//4. HTTP请求页面

(1)有了 HTTP 服务器的 IP 地址之后，主机就能够生成 TCP 套接字，该套接字将用于向 Web 服务器发送 HTTP GET 报文

(2)在生成 TCP 套接字之前，必须先与 HTTP 服务器进行三次握手来建立连接。生成一个具有目的端口 80 的 TCP SYN 报文段，并向 HTTP 服务器发送该报文段

(3)HTTP 服务器收到该报文段之后，生成 TCP SYN ACK 报文段，发回给主机

(4)连接建立之后，浏览器生成 HTTP GET 报文，并交付给 HTTP 服务器

(5)HTTP 服务器从 TCP 套接字读取 HTTP GET 报文，生成一个 HTTP 响应报文，将 Web 页面内容放入报文主体中，发回给主机。

(6)浏览器收到 HTTP 响应报文后，抽取出 Web 页面内容，之后进行渲染，显示 Web 页面

**选择题刷题：**

//用户账户由用户名和密码标识

//程序计数器PC用于存放下一条指令所在单元的地址的地方;

.指令寄存器IR是临时放置从内存里面取得的程序指令寄存器

.状态寄存器PSW是计算机系统的核心,用来保存运算器的结果状态,程序运行时的工作状态及机器的状态信息

.地址寄存器AR用来保存当前CPU所访问的内存单元地址.

//布线系统与网络体系结构密切相关

//路由器具有网络互联、网络路由和网络隔离等功能，并且路由器总是占有两个或两个以上的IP地址.

//PPP 的两种认证方式中，CHAP 比 PAP 更安全

//当网络通信符合增大的时候，由于冲突增多，网络吞吐率下降，传输延迟增加，因此CSMA/CD方法一般用于通信负荷较轻的应用环境中.

//ARQ协议，即自动重传请求（Automatic Repeat-reQuest），是OSI模型中数据链路层和传输层的错误纠正协议之一。它通过使用确认和超时这两个机制，在不可靠服务的基础上实现可靠的信息传输。简单地说，就是发送方在发送完数据之后，没有收到确认帧，它通常会重发请求。

//如果环上的某个工作站收到令牌并且有信息发送，它就改变令牌中的一位（该操作将令牌变成一个帧开始序列），添加想传输的信息，然后将整个信息发往环中的下一工作站。信息帧沿着环传输直到它到达目的地，目的地创建一个副本以便进一步处理。信息帧继续沿着环传输直到到达发送站时便可以被删除（因此是在发送站回收）。发送站可以通过检验返回帧以查看帧是否被接收站收到并且复制

//物理层，数据链路层，网络层提供设备与设备之间的通信(点到点)，传输层及以上各层提供程序与程序之间的通信(端到端，这里的端是源端和目的端)

//拥塞控制算法大部分是闭环控制

//ATM异步传输模式:ATM中异步是指周期性地插入ATM信元，所有含相同地址的信元形成一虚电路

//TCP存在粘包问题,UDP不存在粘包问题.UDP不存在粘包问题，是由于UDP发送的时候，没有经过Nagle算法优化，不会将多个小包合并一次发送出去。另外，在UDP协议的接收端，采用了链式结构来记录每一个到达的UDP包，这样接收端应用程序一次recv只能从socket接收缓冲区中读出一个数据包。也就是说，发送端send了几次，接收端必须recv几次（无论recv时指定了多大的缓冲区）

//UDP最大负荷是1472字节,TCP最大的是1460字节

//Internet起源于美国最初的一个分组交换网ARPANET,其目的是将不同的主机以一种对等的方式连接起来

//TCP滑动窗口流量控制实际是对对方发送的数据流量的控制。

//Socket是一组接口。而web socket是基于tcp的协议

**TCP与UDP的区别**：

TCP:是面向连接的,可靠的数据流传输,传输单位是TCP报文段

UDP:不连接的不可靠的数据流传输,传输单位是用户数据报,传输快.

TCP的协议

(1)FTP:定义文件传输,21和20端口

(2)Telent:远程登录端口,23端口

(3)SMTP:邮件传输协议,25号端口

(4)POP3:邮件110端口

(5)HTTP:80端口

UDP协议

(1)DNS:域名解析服务:53端口

(2)SNMP:简单网络管理协议:161端口

(3)TFTP:简单的文件传输.69端口使用UDP协议

**MAC地址与IP地址的作用：**

MAC地址是一个硬件地址，用来定义网络设备的位置，主要由数据链路层负责。而IP地址是IP协议提供的一种统一的地址格式，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。

**TCP怎么保证可靠性**：

（1）序列号，确认应答，超时重传

（2）窗口控制与快重传，快恢复

（3）拥塞控制

**TCP/IP数据链路层交互过程**：

网络层等到数据链层用mac地址作为通信目标，数据包到达网络等准备往数据链层发送的时候，首先会去自己的arp缓存表(存着ip-mac对应关系)去查找改目标ip的mac地址，如果查到了，就讲目标ip的mac地址封装到链路层数据包的包头。如果缓存中没有找到，会发起一个广播：who is ip XXX tell ip XXX,所有收到的广播的机器看这个ip是不是自己的，如果是自己的，则以单拨的形式将自己的mac地址回复给请求的机器

**IP怎么知道报文应该交给那个程序**：

根据端口区分；看ip头中的协议标识字段，17是udp，6是tcp。

**阻塞，非阻塞，同步异步**：

阻塞和非阻塞：调用者在事件没有发生的时候，一直在等待事件发生，不能去处理别的任务这是阻塞。调用者在事件没有发生的时候，可以去处理别的任务这是非阻塞。同步和异步：调用者必须循环自去查看事件有没有发生，这种情况是同步。调用者不用自己去查看事件有没有发生，而是等待着注册在事件上的回调函数通知自己，这种情况是异步

**DNS的工作原理：**

当DNS需要在程序中使用名称的时候,他会查询DNS服务器来解析名称

客户机发送的每条查询信息包含:指定的DNS域名,指定的查询类型,DNS指定的类别.

基于UDP服务53端口.

**TCP三次握手,四次挥手：**

(1)三次握手

第一次握手:客户端发送syn=1包(syn=x)到服务器,并进入syn\_send状态，等待服务器应答

第二次握手：服务器受到syn1包，必须确认客户的syn（ack=syn0+1）,同时自己也发送一个syn(syn=y),即syn+ack包,此时服务器进入syn\_recv状态

第三次握手:客户端受到服务器的syn1+ack包,向服务器发送确认报ack(ack=syn1+1),发送完毕,客户端和服务器进入建立连接的状态,完成三次握手

握手传输的包里面没有数据,三次握手完毕之后才传输数据,理想状态下TCP一旦建立连接,在通信双方主动关闭连接之前,TCP都会一直保持下去

(2)四次挥手

第一次挥手:主动关闭方发送一个FIN,用来关闭主动方到被动方的传输数据,也就是主动关闭告诉被动方关闭;我已经不传送数据了,但是主动关闭方还可以接受数据

第二次挥手:被动关闭方收到FIN之后,发送一个ACK给对方,确认序号为受到序号+1

第三次挥手:被动关闭方发送一个FIN,用来关闭被动关闭方到主动关闭方的数据传输,就是告诉主动方我的数据发完了.

第四次挥手:主动关闭方收到FIN之后,发送一个ACK给被动关闭方,确认序号为受到序号+1,完成

**为什么采取三次握手,两次可以吗：**

(1)主机A向B发送连接请求;主机B向A发送报文段确认;主机A发送确认的确认

采用三次的方式是为了防止失效的连接请求报文段突然又传送到服务器,因而产生错误.失效的连接请求指的是:主机A发送了连接请求没有收到B的确认,过一段时间后A重发连接请求,且建立 连接,顺序完成数据传输.如果申请这时候延迟发送给了主机B,主机b以为A又要发起连接,就向A发送确认信号,但是A不理会,于是第三次握手不会产生

如果不进行第三次握手,B就会一直等待A的信号.

因为TCP是全双工的,两次只能保证单向通信正常；四次降低了效率

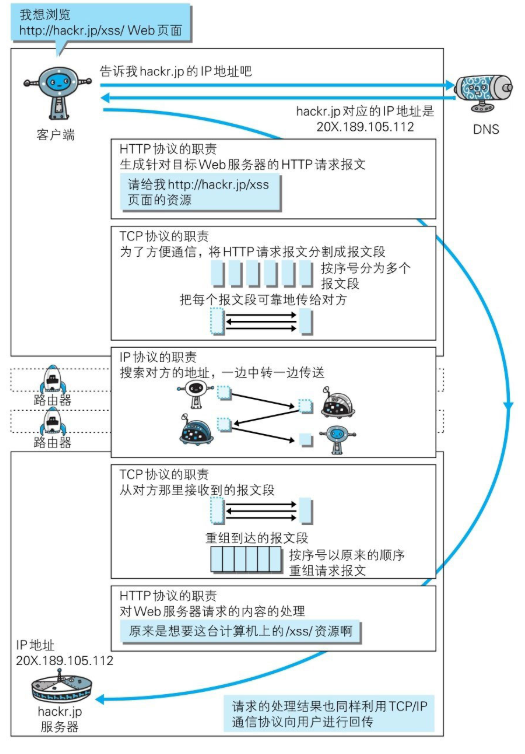
**浏览器访问网站全过程**：

(1)客户浏览器通过DNS解析网址的IP地址,通过IP地址找到客户端到服务器的路径.客户端浏览器发起一个HTTP会话到这个IP地址.然后通过TCCP封装数据报,输入到网络层

(2)在传输层,把HTTP会话请求分成报文段,添加源和目的端口,如80端口等

(3)客户端的网络层主要是查找路由表确定如何到达服务器,期间可能经过多个路由器.

(4)客户端的链路层,包通过链路层发送到路由器,通过ARP协议查找给定IP地址的MAC地址,然后发送ARP请求查找目的地址,得到回应知乎,就可以传输。



**IP地址的分类：**

(1)A类地址:以0开头,第一个字节1~126:(1.0.0.0~126.255.255.255)

(2)B类地址:以10开头,第一个字节范围128~191;(128.0.0.0~191.255.255.255)

(3)C类地址:以110开头,第一个字节范围192~223(192.0.0.0~223.255.255.255)

(4)D类地址:以1110开头,第一个字节范围224~239(224.0.0.0`239.255.255.255)

(5)E类地址,保留

ABC是基本类,D类用于多播.

私有地址:

A:10.0.0.0~10.255.255.255

B:172.16.0.0~172.31.255.255

C:192.168.0.0~192.168.255.255

**OSI层数和TCP层数及协议**：

(1)OSI七层

(2)TCP/IP四层:网络接口层,网络层,传输层,应用层

(3)五层协议:物理层,数据链路层,网络层,传输层,应用层

物理层:RJ45,IEEE802.3;确定机械及电气标准

数据链路层:PPP,FR,HDLC,VLAN,MAC;网桥,交换机,将比特组装成帧和点到点传递

网络层:IP,ICMP,ARP,IGMP,RARP,OSPF,IPX,RIP,IGRP(路由器)数据报的的传递和网际互联

传输层:TCP,UDP,SPX,提供端到端的可靠传输

会话层:NFS,SQL,RPC,建立,管理,终止会话

表示层:JPEG,MPEG,ASII,翻译和加密

应用层:FTP,HTTP,SMTP,TELENT,WWW,NFS.允许访问的手段

**ARP工作原理**：

ARP地址解析协议:

(1)每个主机都会在自己的ARP缓存区建立一个ARP列表.形成IP地址与MAC地址的对应

(2)当源主机发送数据的时候,首先检查ARP列表中是否有对应IP地址的MAC地址,如果有,直接发送,如果没有,就向本网段的所有主机发送ARP数据报,数据报的内容:源主机的IP地址,源主机的MAC地址,目的主机的IP地址

(3)当本网络主机受到之后,首先检查目的IP是不是自己,如果不是,忽略本次数据包,如果是,首先从数据包中提取出源主机的IP和mac地址写入自己的ARP表中,如果已经存在,则覆盖,然后将自己的MAC地址写入ARP响应包中,告诉源主机自己的MAC地址.

(4)源主机受到ARP响应包之后,将MAC地址写入ARP列表中,并利用此信息发送数据.

广播发ARP请求,单播发送ARP响应.

**RARP协议：**

RARP逆地址解析协议:把MAC地址解析成IP地址

(1)在网络中配置一台RARP服务器,里面保存着IP地址和MAC地址的映射关系,

(2)当无盘工作站启动后,就封装一个RARP的数据包,里面有其MAC地址,然后广播到网络上

(3)服务器受到请求之后,就查找对应的mac地址的IP地址装入响应报文中发回给请求者.

RARP具有广播能力

**协议介绍：**

ICMP协议:因特网控制报文协议:用于IP主机,路由器之间传递控制消息

TFTP协议:简单的文件传输协议

HTTP协议:超文本传输协议,简单快捷

NAT协议:网络地址转换属接入广域网技术,将私有地址转换为合法的IP地址

DHCP:动态主机配置协议,让系统得以连接网络,给内部网络或网络服务供应商自动分配IP.

**HTTPS：**

（1）HTTP协议是以明文的方式在网络中传输数据，HTTPS是经过TLS加密后，HTTPS具有更高的安全性

（2）HTTPS在TCP三次握手阶段后，需要进行SSL的handsnake，协商加密使用的对称加密秘钥

（3）HTTPS协议需要服务端申请证书，浏览器安装对应的根证书

（4）HTTP的端口是80，HTTPS的端口是443

**http与HTTPS：**

HTTP协议和HTTPS协议区别如下：1）HTTP协议是以明文的方式在网络中传输数据，而HTTPS协议传输的数据则是经过TLS加密后的，HTTPS具有更高的安全性

2）HTTPS在TCP三次握手阶段之后，还需要进行SSL 的handshake，协商加密使用的对称加密密钥

3）HTTPS协议需要服务端申请证书，浏览器端安装对应的根证书

4）HTTP协议端口是80，HTTPS协议端口是443

HTTPS优点：

HTTPS传输数据过程中使用密钥进行加密，所以安全性更高

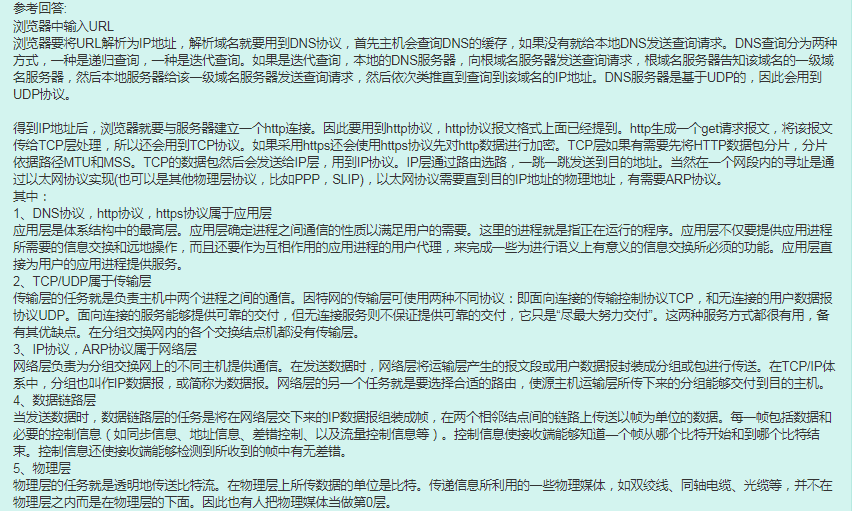
HTTPS协议可以认证用户和服务器，确保数据发送到正确的用户和服务器

HTTPS缺点：

HTTPS握手阶段延时较高：由于在进行HTTP会话之前还需要进行SSL握手，因此HTTPS协议握手阶段延时增加

HTTPS部署成本高：一方面HTTPS协议需要使用证书来验证自身的安全性，所以需要购买CA证书；另一方面由于采用HTTPS协议需要进行加解密的计算，占用CPU资源较多，需要的服务器配置或数目高

**搜索网页的层，及协议：**



**HTTP返回状态吗：**

（1）1xx：表示请求已经接收，继续处理

（2）2xx：成功：

（3）3xx：重定向：要完成请求必须进一步操作

（4）4xx：客户端错误，语法错误或者无法请求

（5）5xx：服务器端错误。

**HTTP与HTTPS：**

**URI**：统一资源标识符,表示web上每一种可用的资源,如html文档,图像等都有一个URI进行标识.

组成:

(1)资源的命名机制

(2)存放资源的主机名

(3)资源自身的名称

PFC2396（URI的通用语法）：对URI进行了定义：

uniform：统一的个数可以方便处理不同类型的资源。

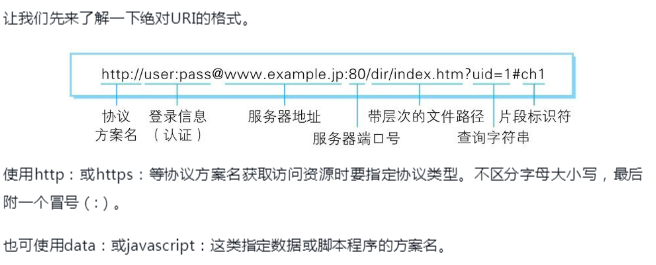
Resource：可识别的全部资源

Identifier:可标识的对象，标识符



**URL**：URL是URI的子集,URL是URI的一种实现方法.

统一资源定位符,对互联网上得到的资源的位置和访问方法的简洁表示,是互联网上标准资源的地址.互联网上每一个文件都有唯一的URL,包含的信息指出文件的位置以及浏览器的处理方法.



登录信息（认证）：指定用户名密码作为服务器端获取资源的必要认证信息，可选选项

服务器地址：使用局对URI必须指定待访问的服务器地址。地址可以是DNS可解析的域名，或者是IP地址。

服务器端口号：指定服务器连接的网络端口号，若省略用默认端口号

带层次的文件路径：指定服务器中的文件路径定位特指资源。

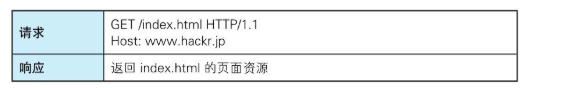
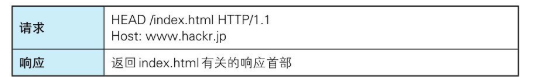
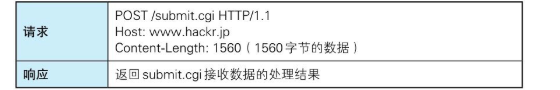
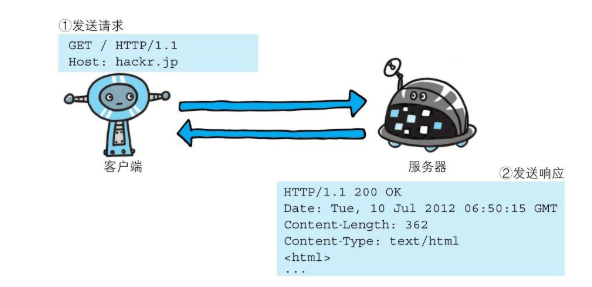
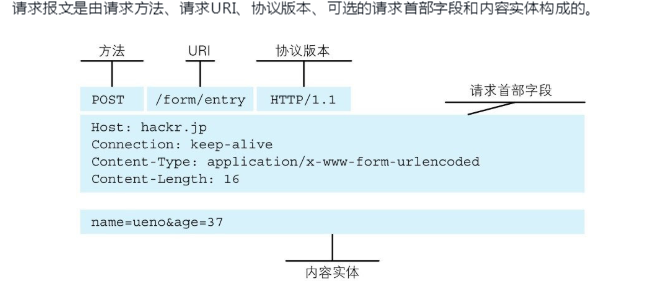
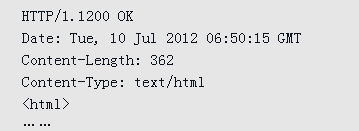
查询字符串：针对已指定的文件路径内的资源，使用字符串传入任意参数，此项可选。

片段标识符：使用片段标识符通常可以标记出已经获取资源中的子资源，可选

**URN**：URN也是URI的子集.

URL的一种更新形式，统一资源名称(URN, Uniform Resource Name)不依赖于位置，并且有可能减少失效连接的个数。但是其流行还需假以时日，因为它需要更精密软件的支持

**HTTP函数：**

(1)get:获取资源  
  
(2)HEAD:获取报文首部,不返回报文实体主体部分,主要用于确认URL的有效性个资源更新的日期时间等.  
  
(3)POST:传输实体主体:主要用于传输数据.不是为了获取响应的主题内容  
  
(4)PUT:上传文件:不带验证机制,任何人都能上传,安全性问题,不推荐使用  
  
(5)PATCH:对资源进行部分修改:PUT修改时完全替代资源,PATCH可以对部分修改  
(6)DELETE:删除文件,与PUT功能相反,也不带验证机制  
  
(7)OPTIONS:查询支持的方法:查询指定的URL能够支持的方法,会返回 Allow: GET, POST, HEAD, OPTIONS 这样的内容  
  
(8)CONNECT:要求在代理服务器通信时建立隧道,使用SSL(安全套接层)和TLS(传输层安全)协议把通信内容加密后经网络隧道传送.  
  
(9)TRACE:追踪路径:服务器会将通信路径返回给客户端,发送请求的时候,在Max-Forwards首部填入数值,每次经过服务器就减1,当数值为0就停止传输.通常不会使用trace,因为容易受到XST供给(CST,跨站追踪).  
  
  
HTTP协议规定：请求必须从客户端发起，先从客户端开始建立通信，服务器端在没有接收到请求之前不会发送响应。  
  
（1）请求发送：GET指明操作；HTTP/1.1表明版本  
  
  
（2）服务器响应：  
  
先是版本号，200OK代表的状态成功；下一行显示日期；接着以一空行分隔，之后的内容为资源主体。  


**GET和POST：**







**HTTP状态码：**



1xx信息:100continue:表明到目前为止都很正常,客户端可以继续发送请求或者忽略该响应

2xx:成功

200:OK：表示客户端的请求正在被正常处理

204NO Content:请求已经成功处理,但是返回响应报文不包括实体的主体部分.一般只需要从客户端往服务器发送信息,而不需要返回数据的时候使用.

206 Partial Content:表示客户端进行范围请求,响应报文包含Content-Range指定范围的实体内容

3xx重定向:

301Moved Permanently:永久性重定向

302Found:临时重定向

303 See Other:和302功能相同,但是3030明确对象要求客户端采用GET方法获取资源

注：虽然 HTTP 协议规定 301、302 状态下重定向时不允许把 POST 方法改成 GET 方法，但是大多数浏览器都会在 301、302 和 303 状态下的重定向把 POST 方法改成 GET 方法。

304Not Modified:如果请求报文首部包含一些条件,如果不满足条件.服务器返回304状态码

307 Temporary Redirect:临时重定向,与302类似,但是307要求浏览器不会吧重定向请求的POST方法改成GET方法

4xx客户端错误

400Bad Request:请求报文存在语法错误

401Unauthorized:表示发送的请求需要有认证消息.如果之前已经进行过一次请求,则表示该用户认证失败

403 Forbidden:请求被拒绝

404Not Found

5xx:服务器错误:

500 Internet Server Error:服务器正在执行请求时发生错误

503Service Unavailable:服务器暂时处于超负载或者正在进行停机维护,现在无法处理请求

**http是不保存状态的协议**：

是一种无状态的协议，HTTP/1.1虽然是无状态协议，但为了实现期望的保持状态功能，于是引入了Cookie技术。有了Cookie再用HTTP协议通信，就可以管理状态了。

**连接管理：**



HTTP协议的初始版本中，每进行一次HTTP通信就要断开一次TCP连接

比如，使用浏览器浏览一个包含多张图片的HTML页面时，在发送请求访问HTML页面资源的同时，也会请求该HTML页面里包含的其他资源。因此，每次的请求都会造成无谓的TCP连接建立和断开，增加通信量的开销。

在HTTP/1.1中，所有的连接默认都是持久连接，但在HTTP/1.0内并未标准化

**管线化：**

持久连接使得多数请求以管线化（pipelining）方式发送成为可能。从前发送请求后需等待并收到响应，才能发送下一个请求。管线化技术出现后，不用等待响应亦可直接发送下一个请求。这样就能够做到同时并行发送多个请求，而不需要一个接一个地等待响应了。

**Cookie：**

HTTP协议是无状态的,主要是为了让HTTP协议尽可能简单,使得能够处理大量事务.HTTP/1.1引入Cookie来保存状态信息

Cookie是服务器发送到用户浏览器并保存在本地的一小块数据,他会在浏览器之后向统一服务器再次发送请求时被携带上,用于告知服务端两个请求是否来自于同一个浏览器,会带来额外开销.

(1)用途

会话状态管理:记录登录状态等

个性化设置:自定义

浏览器行为跟踪:

(2)创建过程:

服务器发送响应报文包含Set-Cookie首部字段,客户端得到响应报文后把Cookie内内容保存带浏览器中

客户端之后对同一个服务器发送请求时,会从浏览器中取出Cookie信息并通过Cookie请求首部字段发送给服务器

(3)分类

会话期Cookie:浏览器关闭之后自动删除.

持久性Cookie:指定过期时间,或者有效期之后就成为持久性的Cookie

(4)作用域

Domain标识指定那些主机可以接受Cookie.如果不指定,默认当前文档的主机.如果指定了Domain,则一般包含子域名.

Path标识指定了主机下的那些路径可以接受Cookie(该URL路径必须存在于请求URL中).(/)作为路径分隔符,子路径也会被匹配.

(5)JavaScript

浏览器通过document.cookie属性可以创建新的Cookie，也可以通过该属性访问非HttpOnly标识的Cookie

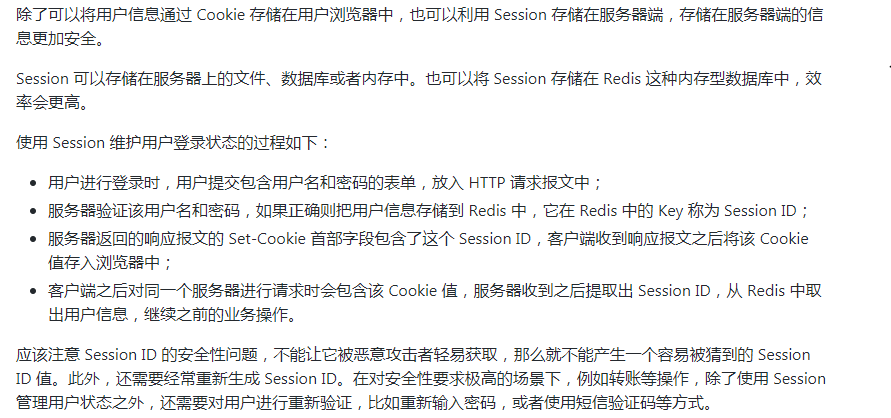
（6）HttpOnly

标记为 HttpOnly 的 Cookie 不能被 JavaScript 脚本调用。跨站脚本攻击 (XSS) 常常使用 JavaScript 的 document.cookie API 窃取用户的 Cookie 信息，因此使用 HttpOnly 标记可以在一定程度上避免 XSS 攻击。

(7)Secure

标记为 Secure 的 Cookie 只能通过被 HTTPS 协议加密过的请求发送给服务端。但即便设置了 Secure 标记，敏感信息也不应该通过 Cookie 传输，因为 Cookie 有其固有的不安全性，Secure 标记也无法提供确实的安全保障。

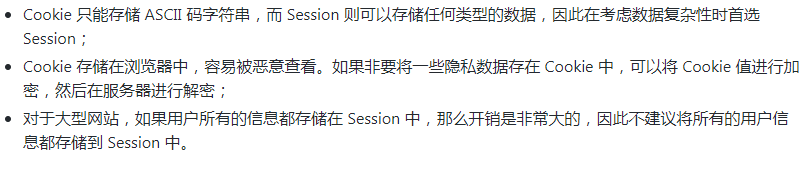
(8)Session



(9)浏览器禁用Cookie:

此时无法使用Cookie来保存用户信息,只能使用Session.除此之外,不能再将Session ID存放在Cookie中,而是使用URL重写技术,将Session ID作为URL的参数进行传递

(10)Cookie和Session选择



**缓存：**

(1)优点

缓解服务器压力;

降低客户端获取资源的延迟;缓存通常位于内存中,读取缓存的速度更快.并且缓存服务器在地理位置上也有可能比源服务器来的近.

(2)实现方法

让代理服务器进行缓存

让客户端浏览器进行缓存

(3)Cache-Control

Http/1.1通过Cache-Control首部字段来控制缓存

1)禁止进行缓存:no-store:规定不能对请求或者响应的任何人以部分缓存

2)强制确认缓存:no-cache:缓存服务器需要先向源服务器验证缓存资源的有效性,只有当缓存资源有效的时候才能使用该缓存对客户端的请求进行响应.

3)私有缓存和公有缓存

private指定将资源作为私有缓存,智能被单独使用,一般存储在用户浏览器中

public:指令规定了将资源作为公共缓存,可以被多个用户使用,一般存储在代理服务器中.

4)缓存过期机制

max-age:出现在请求斑纹,并且缓存资源的缓存时间小于该指令指定的时间,就能接收该缓存.

max-age指令出现在响应报文,表示缓存资源在缓存服务器中保存的时间

Expires首部字段也可以用于告知缓存服务器该资源什么时候过期

在Http/1.1,会优先处理max-age指令

在Http/1.0中,max-age指令会被忽略掉

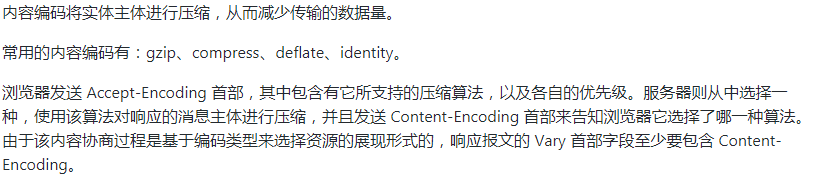
(4)缓存验证



**内容协商：**



**内容编码：**



**范围请求：**



**分块传输编码：**

file://c:\users\admini~1\appdata\local\temp\tmplhfp_m\1.png

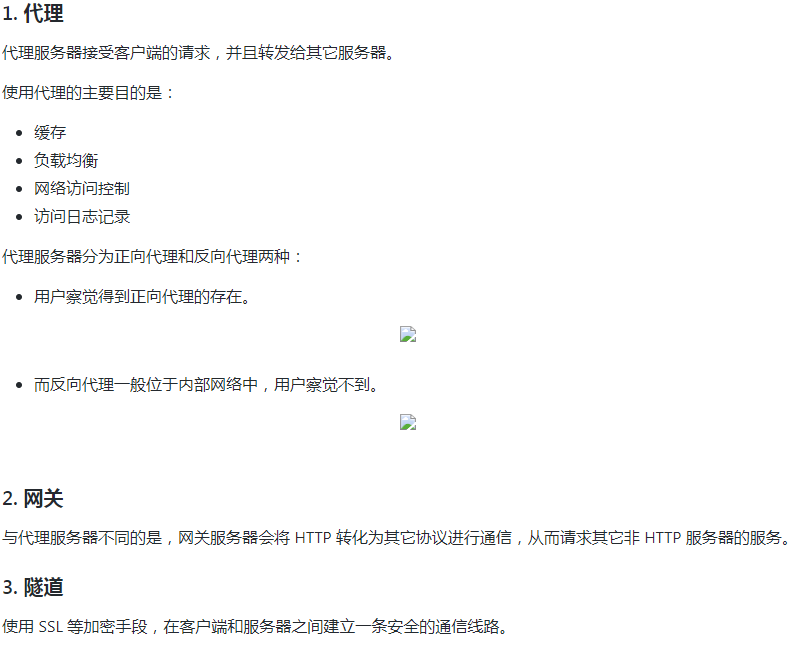
**多部分对象集合：**



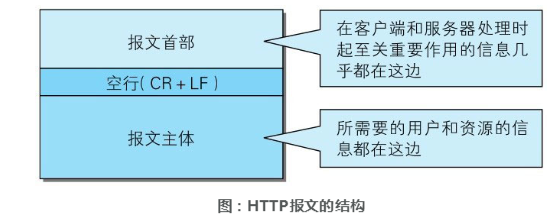
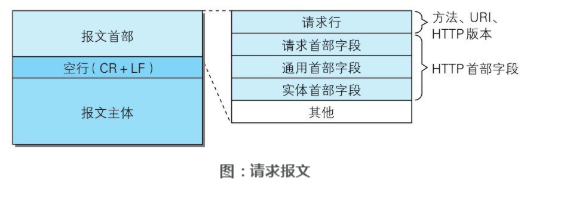
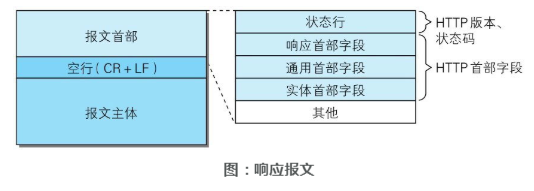
**虚拟主机：**

file://c:\users\admini~1\appdata\local\temp\tmplhfp_m\1.png

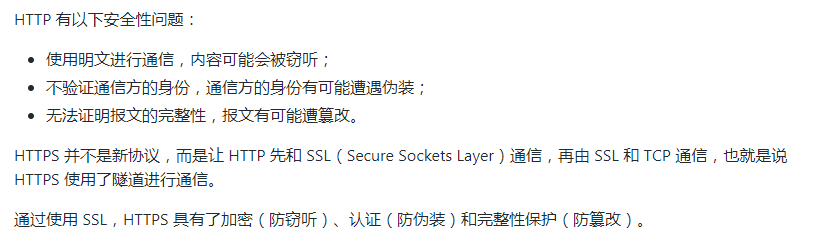
**通信数据转发：**

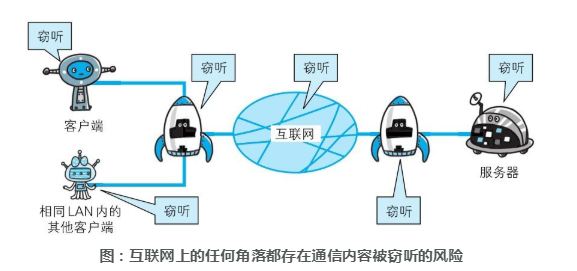


**HTTP报文：**

  
  
请求报文：  
  
响应报文：  


**HTTPS：**



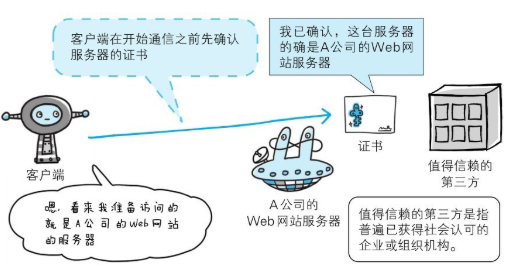


**加密：**

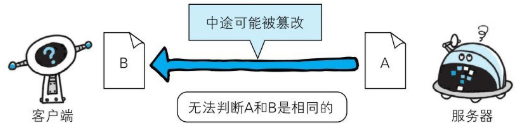


SSL不仅提供加密处理，而且还使用了一种被称为证书的手段，可以用于确定方。

证书由第三方值得信赖的机构颁发，用于证明服务器和客户端是实际存在的，另外，伪造证书从技术角度上说是非常困难的。所以只要能够确认通信方（服务器或者客户端）的证书，就可以判断通信方的真实意图。



由于HTTP无法证明报文的完整性，所以，在请求发出之后知道对方接受到之前的这段时间内，及时请求或者响应内容被篡改，也没有办法获悉。就是没有办法确认发出的请求/响应和接收到的请求/响应前后是相同的。



如果中间被人篡改了，叫做中间人攻击。

于是就有了HTTPS=HTTP+加密+认证+完整性保护

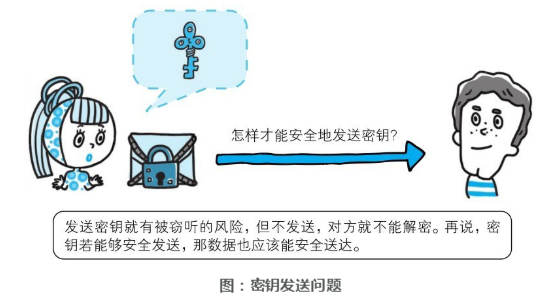
在采用SSL之后，HTTP就具有加密，证书和完整性保护的功能。其他协议也可以与SSL结合，不仅仅是SSL。

交换密钥的公开密钥加密技术：

SSL采用一种叫做公开密钥加密的处理方法。加密的算法是公开的，密钥是保密的，加密和解密都会用到密钥。

（1）共享密钥加密

加密和解密运用同一个密钥叫做共享密钥，又叫做对称密钥加密。共享密钥加密必须将密钥也发送给对方，但是存在中间人获取密钥的风险。



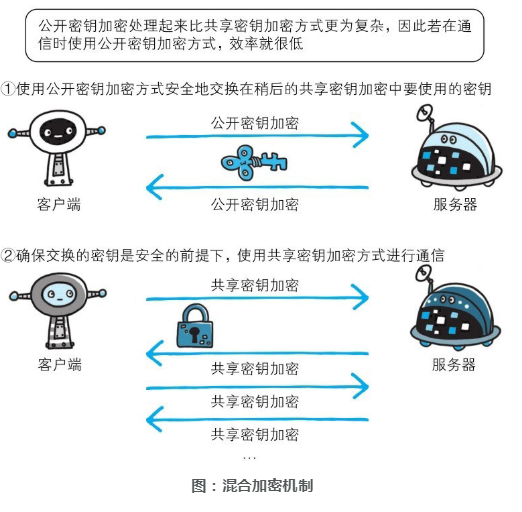
（2）使用两把密钥的公开密钥加密

公开密钥是一种非对称的密钥。一把叫做私有密钥，另一把叫做公有密钥。私有密钥不能让任何人知道，公有密钥是公开的。

使用共有密钥的加密方式，发送密文的一方使用对方的公有密钥急性加密处理，对方收到已经加密的信息之后，再使用自己的私有密钥解密。这样就不需要发送解密的私有密钥。

要想根据密文和公有密钥破解信息是异常困难的，因为解密的过程是对离散数进行求值，目前是很难做到的。

（3）HTTPS采用混合加密机制。



（4）证明公有密钥正确性的证书

必须要证明共有密钥是正确的。比如，准备和某台服务器建立公开密钥加密方式下的通信，如何证明收到的公有密钥就是原本预想的那台服务器的公开密钥。有可能在传输的过程中公有密钥被替换掉了。

因此，引入了数字证书认证机构(CA)和其相关机关颁发的公开密钥证书。

数字证书认证机构处于客户端和服务器双方都可信赖的第三方机构的立场上。首先服务器的运营人员向数字认证机构提出公开密钥的申请。数字认证机构判明之后，会对已申请的密钥做数字签名，然后分配这个签名了的密钥与公钥证书绑定在一起。

服务器将这份数字证书认证机构颁发的共有密钥发送给客户端，以进行公开密钥加密的方式通讯。公钥证书也可以叫做数字证书。

接到证书的客户端可以使用数字认证机构的公开密钥，对那张证书上数字签名进行验证，一旦验证成功：认证服务器的公开密钥是真是有效的；服务器的公开密钥是值得信赖的。

但是必须保证认证机关的公开密钥安全的转交给客户端？？多数浏览器开发商发布版本的时候，会事先在内部植入常用认证机关的公开密钥。

证书的一个作用是来证明作为通信一方的服务器是否规范，另外一个作用是确认对方服务器背后运行的企业是否真实存在。拥有该特性的证书就是EV SSL。

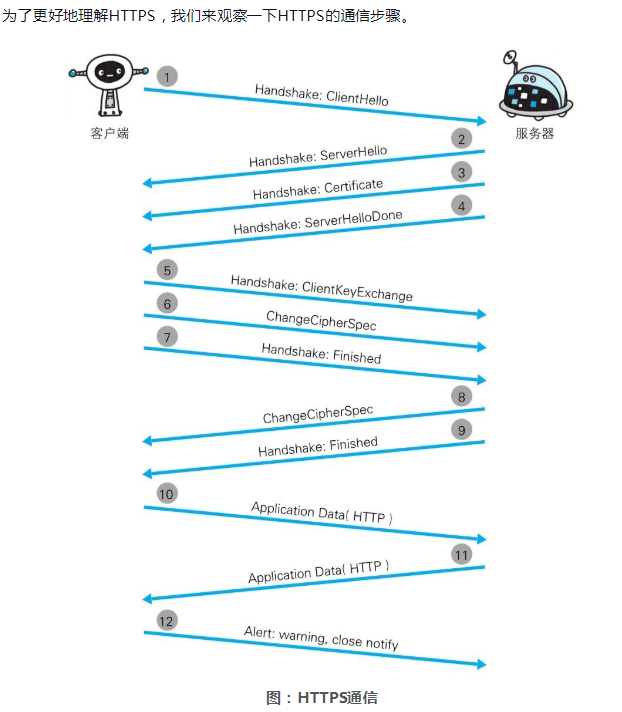
HTTPs还可以使用客户端证书，以客户端证书对客户端进行认证，证明和服务器正在通信的客户端是正确的客户端。但是客户端证书存在几处问题。

客户端证书的获取及发布？

想要获取证书，用户得自己安装客户端证书，但是由于证书是付费的，且证书价格较高。

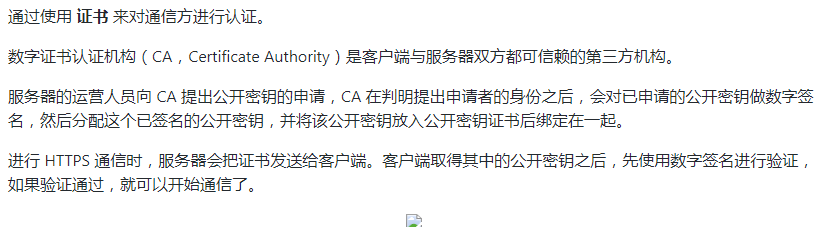
还存在的问题是，客户端证书只能证明客户端存在，不能证明用户本人的正确性。只要获得了安装有客户端证书的计算机的使用权限，就意味着拥有了使用客户端证书的权限。

OpenSSL可以每个人都构建一个自己的认证机构，但是不能作为互联网那个上的证书使用

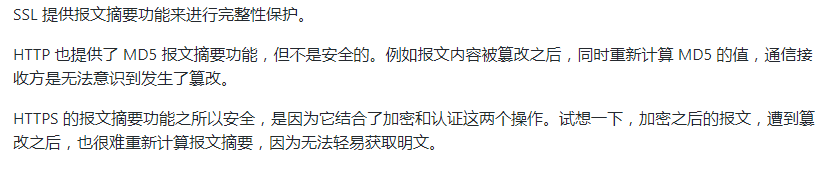




**认证：**



**完整性保护：**

**HTTPS缺点：**

(1)需要加密解密,速度更慢

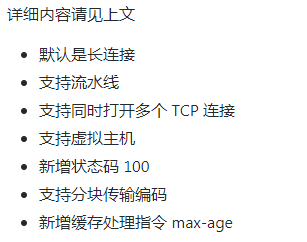
(2)需要支付证书授权的费用

（3）慢

**HTTP/2.0：**



**http/1.0特性：**

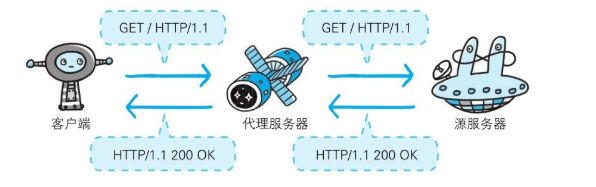


**代理，网关，隧道**：

代理：具有转发功能的应用程序，中间人角色，接收由客户端发送的请求并转发给服务器，同时也接收服务器返回的响应并且转发给客户端。

网关：转发其让服务器通信数据的服务器，接收从客户端发送来的请求，他就像自己拥有资源的源服务器一样对请求进行处理。有时候客户端都不知道自己通信的目标是一个网关。

隧道：隧道是在相隔甚远的客户端和服务器两者之间进行中转，并且保持双方通信连接的应用程序。



代理服务器的基本行为就是接受客户端发送的请求后转发给其他服务器。代理不改变请求URI，会直接发送给前方持有资源的目标服务器。持有资源的服务器称为源服务器。从源服务器返回的相应经过代理服务器后再传给客户端。可以有多级代理服务器。

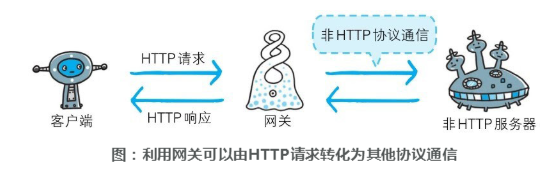
代理服务器存在的理由：利用缓存技术减少网络带宽的流量，组织内部针对特定网站的访问控制，以获取访问日志为主要目的，等等。

代理有多种方法，按两种基准分类。一种是是否使用缓存，另一种是是否会修改报文。

缓存代理：对预先将资源副本缓存在代理服务器中，当代理服务器接收到相同的资源请求的时候，就可以不从源服务器那里获取资源，而是将之前缓存的资源作为响应返回。

透明代理：转发请求或者响应时，不对报文做任何加工的代理类型叫透明代理。

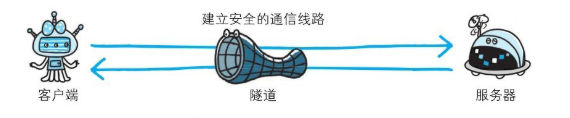
**网关：**



网关的工作机制和代理十分相似。而网关能使得通信线路上的服务器提供非HTTP协议服务。利用网关能提高通信的安全性，因为可以在客户端与网关之间的通信线路上加密确保连接的安全。比如网关可以连接数据库，使用SQL查询数据**。**

**隧道：**

隧道可按要求建立一条与其他服务器的通信线路，届时使用SSL等加密手段通信。隧道的目的是确保客户端与服务器的安全通信。隧道本身不会去解析HTTP请求。请求保持原样中转给之后的服务器。隧道会在通信双方断开连接时结束。隧道本身是透明的。



**缓存：**

缓存指的是代理服务器或者客户端在本地磁盘内保存的资源副本，利用缓存可以减少对源服务器的访问，节省通信流量和通信时间。

缓存服务器是代理服务器的一种，并归类在缓存代理类型中。

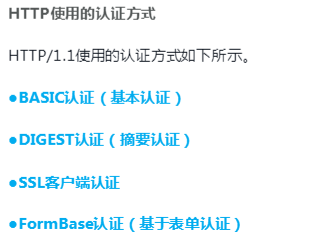
缓存需要有效期限，因为源服务器可能存在更新，若果判断缓存失败，就会从源服务器再次获取新的资源。

客户端的缓存：

客户端的缓存称之为临时网络文件。浏览器缓存如果有效，就不必向服务器再请求相同的资源，直接从本地磁盘内读取。

和缓存服务器相同，当缓存过期之后，会想源服务器确认资源的有效性，若失效，则请求新的资源。

**认证：**



（1）BASIC认证：采用Base64编码，但是不是加密，不需要任何附加信息就可以对其解码。明文解码之后就是用户的ID和密码。如果被窃听，极有可能被盗。

（2）DIGEST认证：质询/响应的方式，一开始一方会先发送认证要求给另一方，接着使用从另一方接收到的质询码生成响应码，最后将响应吗返回给对方进行认证的方式。



密码泄露可能性降低了。

（3）SSL客户端认证：

借由HTTPS的客户端证书完成认证的方式，凭借客户端证书认证，服务器可以确认访问是否来自于自己登陆的客户端。

（4）基于表单认证

**如果已经建立了连接，但是客户端突然出现故障了怎么办？**

TCP 还设有一个保活计时器，显然，客户端如果出现故障，服务器不能一直等下去，白白浪费资源。服务器每收到一次客户端的请求后都会重新复位这个计时器，时间通常是设置为 2 小时，若两小时还没有收到客户端的任何数据，服务器就会发送一个探测报文段，以后每隔 75 秒钟发送一次。若一连发送 10 个探测报文仍然没反应，服务器就认客户端出了故障，接着就关闭连接。