目录

1.一些概念：
①会话：客户端打开与服务器的连接发出请求到服务器响应客户端请求的全过程称之为会话。会话跟踪指对同一个用户对服务器连续的请求和接受响应的监视。OSI模型的会话层提供。
②会话跟踪常用方法：

- URL重写：在URL结尾添加一个附加数据（sessionID？）标识该会话，以便在服务器端进行识别不同的用户。
- 隐藏表单域，将会话ID添加到HTML表单元素中提交到服务器，此表单元素不在客户端显示。
- Cookie，web服务器发送给客户端的一小段信息，可以选择临时保存、永久保存（有有效期），或禁止保存。客户端访问服务器时，可以读取cookie并发送到服务器端，进行用户的识别。
- session，每个用户都有一个不同的session，不同的用户之间不能共享。在服务器端会创建一个session对象，并产生一个sessionID来标识这个session对象，然后把这个sessionID放入到cookie中发送到客户端，下一次访问时，sessionID会发送到服务器，在服务器端进行识别不同的用户。

使用Token防止伪造请求攻击：Tooken是随机的、一次性的（每次请求后要更新token）、需要保密的，例如敏感操作使用Post，防止Token出现在URL中。

cookie session、token的参考链接:https://www.cnblogs.com/moyand/p/9047978.html
cookie和session的参考链接:https://www.cnblogs.com/andy-zhou/p/5360107.html

---

2.HTTP协议：
概念：以明文方式发送内容，不提供任何方式的数据加密，如果攻击者截取了Web浏览器和网站服务器之间的传输报文，就可以直接读懂其中的信息。

HTTP报文格式

请求报文和响应报文的格式：分为请求行（响应行）、请求头部（响应头部）、空白行、消息主体。

- 请求行：请求行分为请求方法、协议版本号、资源URL
- 状态行：状态行分为协议版本号、状态码、状态消息

HTTP方法和状态码详解

基本方法：主要是GET、POST、PUT、DELETE四种方法，对应对这个资源的查、改、增、删四个操作。

- **GET：**用于信息获取，不会改动资源，提交数据跟在URL之后，可提交数据量受URL长度限制（浏览器接收URL长度有限）。
  - 条件GET：客户端发送一个包，询问在上一次访问完后是否更改了页面，若没有更改，则客户端只需要使用本地缓存，否则，服务器发送更新的网页给客户端。
- **POST：**非幂等的，向指定资源提交数据进行处理请求（例如提交表单或者上传文件），提交数据必须在body部分中，需和服务器统一Content-Type，编码方式。POST请求可能会导致新的资源的建立和/或已有资源的修改。
- **PUT：**幂等的（每次提交相同资源，最终结果一致），从客户端向服务器传送的数据取代指定文档中的内容，每次请求，若对应URL位置没有资源则创建一个，否则会覆盖原先的资源。
- **DELETE：**请求服务器删除指定的页面
- **HEAD：**类似GET请求，只不过返回的响应中没有具体的内容，用于获取报头。
- **CONNECT：**HTTP1.1协议中预留给能够将连接改为管道方式的代理服务器。
- **TRACE：**回显服务器的请求，主要用于测试或诊断。
- **OPTIONS：**允许客户端查看服务器的性能。

状态码：大致可分为1xx，2xx，3xx，4xx，5xx五类

- 1xx，接受的请求正在处理。代表请求已被接受，需要继续处理，这类响应是临时响应，只包含状态行和某些可选的响应头信息，并以空行结束。
- 2xx，成功。这一类状态码，代表请求已成功被服务器接收、理解、并接受。
- 3xx，重定向。这类状态码代表需要客户端采取进一步的操作才能完成请求。
- 4xx，请求错误。这类的状态码代表了客户端看起来可能发生了错误，妨碍了服务器的处理。
- 5xx、6xx，服务器错误。这类状态码代表了服务器在处理请求的过程中有错误或者异常状态发生，也有可能是服务器意识到以当前的软硬件资源无法完成对请求的处理。
- 常见状态码
  - 200 OK //客户端请求成功

- 400 Bad Request //客户端请求有语法错误，不能被服务器所理解
- 401 Unauthorized //请求未经授权，这个状态码必须和Authenticate报头域一起使用
- 403 Forbidden //服务器收到请求，但是拒绝服务
- 404 Not Found //请求资源不存在，eg：输入了错误的URL
- 500 Internal Server Error //服务器发生不可预期的错误
- 503 Server Unavailable //服务器当前不能处理客户端的请求，一段时间后可能恢复正常

HTTP的版本更新和功能

④Transfer—Encoding，标示HTTP报文传输格式的头部值，HTTP规范里只定义了该值为chunked，表示消息体分块传输的传输方式。HTTP2不支持，因为这个版本的协议有自己的streaming传输方式。

⑤HTTP的连接状态（HTTP本身是无连接的）

- 非持续连接：每次连接只处理一个请求，服务端处理完客户端一次请求，等到客户端做出回应之后便断开连接。这种方式有利于节省传输时间，但随着互联网的发展，一台服务器同一时间处理的请求越来越多，如果依然采用原来的方式，将会在建立和断开连接上花费大量的时间。

为了避免这一劣势，在HTTP/1.0提出了持久连接。HTTP/1.1的默认状态是持续连接的流水线工作方式。

- 持久连接(连接复用)：当一个TCP连接服务器多次请求，客户端会在请求的Header中携带Connection：Keep—Alive，向服务器请求持久连接。该模式可使客户端到服务器端的连接在一段时间内和有限次请求次数内持续有效，避免了频繁地新建连接。

在HTTP/1.1时代，持久连接成为了默认的连接方式，但这个模式的弊端也暴露出来了，即所有的连接都是串行的，当某一个请求阻塞时就会导致同一条连接的后续请求被阻塞。为了解决这一问题，提出了Pipelining的概念。

- HTTP Pipelining（管线化、流水线）：是将多个HTTP请求整批提交的技术，在传递过程中不需要等待服务器端的回应。值得注意的是，管线化通过持久连接完成，因此HTTP/1.0不支持该技术。另外，管线化不影响响应到来的顺序，即按请求的顺序响应。

到了SPDY和HTTP/2时代，多路复用（multiplexing）技术出现；能够让多个请求和响应的传输完全混杂在一起进行，通过streamID来互相识别。

而对于HTTP的无连接，是指HTTP借助于底层的TCP虚拟连接，HTTP协议本身无需连接。

⑥HTTP协议的无状态

是指服务端对于客户端每次发送的请求都认为它是一个新的请求，上一次会话和下一次会话；这种特性既有优点也有缺点，优点在于解放了服务器，每一次请求都“点到为止”不会造成不必要连接占用，缺点在于每次请求会传输大量重复的内容信息。

非版本信息参考链接:https://blog.csdn.net/holmofy/article/details/68492045

3.HTTPS协议

概念：为了解决HTTP的安全缺陷，HTTPS在HTTP的基础上加入了SSL/TLS协议，SSL协议位于TCP/IP协议与各种应用层协议之间，依靠证书来验证服务器的身份，并为浏览器和服务器之间的通信加密。

相对于HTTP协议，HTTPS协议添加了SSL协议，即安全套接层。SSL在传输层对网络连接进行加密，是为网络通信提供安全及数据完整性的一种安全协议。目前SSL已被广泛地用于Web浏览器和服务器之间的身份认证和加密数据传输。

SSL协议可分为两层：
SSL记录协议，它建立在可靠的传输协议（如TCP）之上，为高层协议提供数据封装、压缩、加密等基本功能的支持。
SSL握手协议，它建立在SSL记录协议之上，用于在实际的数据传输开始前，通讯双方进行身份认证、协商加密算法、交换加密密钥等。

建立连接前的握手阶段。
在浏览器中输入URL并回车后：

- 首先在DNS缓存中寻找有没有该url对应的ip，没有则通过dns服务获取url对应ip。
- tcp三次握手建立tcp连接。

http到这里就可以正常通信了，下面是https的握手阶段：

- 客户端发送请求到服务器（发送客户端支持的加密协议及协议版本）。
- 服务器从中选择合适的加密协议，没有则断开连接，否则返回公钥和证书到客户端。（协商加密算法）,证书里包含了公钥,防止公钥被篡改.
- 客户端验证证书的安全性/可信性，如果通过则会生成一个随机数，用公钥对其加密，发送到服务器。（验证证书和确定加密密钥）
- 服务器收到随机数加密后的数字后，用私钥解密得到真正随机数，然后用这个随机数当做密钥对需要发送的数据进行对称加密。
- 客户端在接收到加密的数据后，用之前保存的随机数对数据进行解密，将解密后的数据解析出来并对数据进行校验。（确认加密密钥交换成功）
- 完成握手，此阶段完成了协商加密算法，认证身份和交换密钥等需求。

PS：使用HTTPS前需要保证服务端配置了正确的证书。

参考链接:https://blog.csdn.net/clh604/article/details/22179907
参考链接2:https://www.cnblogs.com/digdeep/p/4832885.html

4.HTTP和HTTPS的区别

- https协议需要到ca申请SSL证书，一般免费证书较少，因而需要一定费用。
- http是超文本传输协议，信息是明文传输，https则是具有安全性的ssl加密传输协议。
- http和https使用的是完全不同的连接方式，用的端口也不一样，前者是80，后者是443。
- http是无连接，无状态的；HTTPS协议是有连接的，由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议，比http协议安全。

HTTPS协议的缺点

- HTTPS协议握手阶段比较费时，因为数据传输前，通讯双方需要进行身份认证、协商加密算法、交换加密密钥等；（时间开销）
- HTTPS连接缓存不如HTTP高效，会增加数据开销和功耗，甚至已有的安全措施也会因此而受到影响；（数据和缓存开销）
- SSL证书需要钱，功能越强大的证书费用越高，个人网站、小网站没有必要一般不会用。（金钱开销）
- SSL证书通常需要绑定IP，不能在同一IP上绑定多个域名，IPv4资源不可能支撑这个消耗。
- HTTPS协议的加密范围也比较有限，在黑客攻击、拒绝服务攻击、服务器劫持等方面几乎起不到什么作用。最关键的，SSL证书的信用链体系并不安全，特别是在某些国家可以控制CA根证书的情况下，中间人攻击一样可行。（加密范围有限，CA证书的信用问题）

参考资料：https://www.cnblogs.com/wqhwe/p/5407468.html

5.TCP协议：

概念：TCP协议是一种面向连接、面向字节流、的可靠传输层协议，TCP报文中不包括本地IP地址和目的IP地址。使用校验和、确认和重传机制来保证可靠传输。数据分节并排序，保证数据的顺序不变和非重复。滑动窗口实现流量控制，动态改变窗口大小进行拥塞控制。

TCP协议的特性：
1.面向连接：建立连接和释放连接都需要双方彼此通信才能完成操作。如下：

- 建立连接：三次握手。客户端和服务器之间总共发送三个包，目的是连接服务器指定端口，建立TCP连接，并同步连接双方的序列号和确认号，交换TCP窗口大小。
  - 第一次握手（SYN=1，Seq=x），客户端发送一个TCP的SYN标志值为1的包，指明客户端打算连接的服务器的端口、以及初始序号X，保存在包头的序列号字段里，发送完毕后，客户端进入SYN_SEND状态。
  - 第二次握手，SYN=1，ACK=1，seq=y，ACKnum=x+1。服务器发回确认包应答，服务器端选择自己的ISN序列号，放到Seq域里，同时将确认序号设置为客户的ISN+1，发送的信息还包括了窗口的大小。发送完毕后，服务器端进入SYN_RCVD状态。
  - 第三次握手，SYN=0，ACK=1，ACKnum=y+1，客户端再次发送确认包（ACK），并且在数据段方写ISN的+1，发送完毕后，客户端进入ESTABLISHED状态，当服务器端接受到这个包时，也进入ESTABLISHED状态，TCP握手结束。

- 释放连接：四次挥手。释放TCP连接需要发送四个包，因此称为四次挥手，客户端或服务器端均可主动发起挥手动作。
  - 第一次挥手（FIN=1，seq=x），假设客户端想要关闭连接，客户端发送一个FIN=1，seq=x（序列号），表示已经没有数据可以发送了，但是仍可以接受数据。发送完毕后，客户端进入FIN_WAIT_1状态。
  - 第二次挥手（ACK=1，ACKnum= x+1），服务器确认客户端的FIN包，发送一个确认包，表示自己接受到了客户端关闭连接的请求，但还没有准备关闭连接，发送完毕后，服务器进入CLOSE_WAIT状态，客户端接收到这个确认包后进入FIN_WAIT_2状态，等到服务器关闭连接。
  - 第三次挥手（FIN=1，seq=y），服务器端做好关闭连接的准备，向客户端发送结束连接请求，发送完毕后，服务器端经过LAST_ACK状态之后进入LISTEN状态，等待来自客户端的最后一个ACK。
  - 第四次挥手（ACK=1，ACKnum=y+1），客户端接受到来自服务器端的关闭请求，发送一个确认包，并进入TIME_WAIT状态，等待可能出现的要求重传的ACK包(因为有可能第四次挥手的ACK丢失了,这种情况下服务器经过一段时间没有收到到ACK,会认为第三次挥手的报文已丢失,因此会重发之前的FIN+ACK报文)。服务器接受到这个确认包后，关闭连接，进入CLOSED状态，客户端等待了某个固定时间之后，没有收到来自服务器端的ACK，认为服务器端已经正常关闭，于是自己也关闭连接，进入CLOSED状态。
- 保持定时器和时间等待计时器：和TCP连接的运行状态以及连接释放中可能存在的问题有关。
  - 保持定时器：用来防治TCP连接长时间处于空闲状态。在服务器中设置定时器，当服务器收到客户端信息时，就将定时器复位，超时假设设置为2小时，若服务器过了2小时没有收到客户端的信息，它就发送探测报文，如果发送了10个报文（时间间隔75s）还没有响应，终止该连接。
  - 时间等待定时器：在连接终止期间使用，当TCP关闭一个连接时，它并不认为这个连接马上就关闭了。在时间等待过程中，连接还处于一种过渡状态。等待器的值一般设置为一个报文的寿命的两倍。

TCP为什么需要三次握手和四次挥手—参考链接：https://www.cnblogs.com/lms0755/p/9053119.html

2.支持字节流传输：应用层不需要规定特点的数据格式。

3.支持全双工服务：TCP通信进程的任何一方都不需要专门发送确认报文，而可以在发送数据时顺便把确认信息捎带传输（TCP报头中的确认序号）。另外，通信双方都设置有发送和接受缓冲区，高层应用程序在它合适的时候到缓冲区中读取数据。支持并发连接，即同时建立和维持多个TCP连接。

4.支持可靠传输服务：使用校验和和确认机制检查数据是否安全、完整地到达。另外，还提供了拥塞控制和重传功能。

- 流量控制：滑动窗口。TCP使用以字节为单位的滑动窗口协议来控制字节流的发送、接受、确认和重传过程。
  - TCP使用两个缓存和两个窗口来控制字节流的传输过程。
  - 接收方通过TCP报头通知发送方：已经正确接收的字节号，以及发送方还能够继续发送的字节流。
  - TCP协议通过报头的序号来表示发送的字节，用确认号表示哪些字节已经被正确接收。
  - TCP通过滑动窗口去跟踪和记录发送字节的状态，以实现差错控制的功能。
  - 可以将发送的字节分为以下四种类型：
    - 第一类：已发送且已被确认的字节。
    - 第二类：已发送但未被确认的字节。
    - 第三类：未发送但接收方已做好准备接收的字节。
    - 第四类：未发送且对方未做好准备接收的字节。
  - 发送窗口：表示发送方已经发送但没有被确认，以及可以随时发送的总字节数。其长度等于第二类和第三类字节数之和。
  - 可用窗口：表示发送方可以随时发送的字节数。其长度等于第三类字节数。
  - 发送方发送可用窗口的第三类字节流，则第三类字节变成第二类字节，等待接收方确认。
  - 经过一段时间后，接收方向发送方发送报文，确认之前发送的报文。如果保持发送窗口为原长度，则将窗口向左滑动。
  - 接收方可以在任何时候发送确认，窗口大小可以由接收方根据自身需要增大或减少，但发送窗口值不能够超过接受窗口值，发送方可以根据自身的需要来决定。
  - TCP滑动窗口是面向字节的，它可以起到差错控制和流量控制作用。

- 利用滑动窗口进行流量控制的过程：
  - 当接收方从缓存中读取字节的速度大于或等于字节到达的速度，那个接收方将在每个确认中发出一个非零的窗口通告，否则，接收方发送零窗口的通告。
  - 当发送方接受到一个零窗口通告时，停止发送，直到下一次接受到接受方新发送一个非零窗口通告为止。

- 坚持定时器：接受方发出了零窗口通告之后，发送方就停止发送，这个过程直到接受方再发送一个非零窗口通告为止。但如果下一个非零窗口通告丢失，那么发送方将无休止地等待接收方的通知，这就造成了死锁。为了避免这种现象，TCP为每个连接使用一个坚持定时器，当发送方收到一个零窗口通告时，启动坚持定时器。当坚持定时器的倒计时为0时，发送方发送一个探测报文，探测报文的作用是提示接收方的TCP：确认已丢失，必须重传。若第一次探测没有收到接收方的回答，将坚持定时器的值加倍和复位，再次探测，直到定时器的值到达阈值（60s）后，每隔60s探测一次，直到窗口重新打开。

- 确认重传：在互联网中，报文段丢失是不可避免的。对接受的字节流序号不连续的处理方法有两种：回退方式和选择重传方式。
  - 回退方式：在丢失最久的一个报文段时，不管之后的报文段是否已经正确接收，从丢失的当前报文开始，重发所有的报文段。这种方法效率较低。
  - 选择重传方式：当接收到序号不连续的字节时，如果这些字节的序号都在接收窗口内，则首先接收这些字节，然后将丢失的字节流序号通知发送方，发送方只需要重发丢失的报文段即可。

- 重传计时器：用来处理报文确认和等待重传的时间。当发送方TCP发送一个报文时，首先将它的一个报文的副本放入重传队列，同时启动一个重传定时器，若在重传定时器倒计时到0之前收到确认，表示报文传输成功，否则表示该报文传输失败，准备重传该报文。需要注意的是重传时间的设置，由于不同的时间和不同的网络环境，所以应选择使用一个动态的自适应重传方法。
  - 计算重传时间：Timeout = b*RTT，RTT = a*旧RTT + (1–a)*新的RTT测量值

- 拥塞控制：用于防止由于过多的报文进入网络而造成路由器与链路过载情况的发生。网络出现拥塞的条件是：对网络资源的需求>网络可用资源。
  - 流量控制可以很好地协调发送方和接收方之间的端到端报文发送和处理速度，但是无法控制进入网络的总体流量。
  - 理想的拥塞控制是在网络负载达到饱和点之前，网络吞吐量一直保持线性增长，而到达饱和点之后网络吞吐量维持不变。
  - TCP的拥塞控制方法分为：慢开始、拥塞避免、快重传和快恢复。
  - 使用滑动窗口实现拥塞控制：发送窗口=Min（通知窗口，拥塞窗口）
  - 在慢开始和拥塞避免的算法中，网络是否出现拥塞是根据路由器是否丢弃分组来确定的。

-
  - 慢开始：当主机开始发送数据时，它对网络的负载状态不了解，这时用试探的方法，由小到大逐步地增大拥塞窗口。当出现拥塞时，拥塞窗口就立即减小。
  - 假设拥塞窗口大小为cwnd，慢开始的阈值为SST，那么：
    - 当cwnd<SST，使用慢开始算法
    - 当cwnd>SST，停止使用慢开始算法，使用拥塞避免算法
    - 当Cwnd=SST，既可以使用慢开始算法，也可使用拥塞避免算法
  - 拥塞避免算法：当cwnd>SST时，然后每增加一个往返就将拥塞窗口值cwnd加1。等到出现一个网络拥塞时，SST设置为出现的cwnd最大值的1/2，然后重新开始慢开始和拥塞避免的阶段。

- 快重传：当个别数据报丢失，但后面的数据报都正确接收时，不能根据丢失的数据报超时而简单地判断网络出现拥塞。这种情况下，应采用快重传与快恢复拥塞控制算法。
  - 当接收到m1、m2、m4报文，丢失m3报文时，这时接收方不能对m4进行确认，因为m4属于乱序的报文。根据快重传算法，接收方应立刻向发送方连续三次发出对m2的重复确认，要求发送方尽早重传未被重传的报文，发送方收到三个重复确认后，立刻重传，不必等待设置的重传计数器时间到期。
  - 快恢复：与快重传算法配合的算法。它规定：
    - 当发送方收到第一个对m2的重复确认时，发送方立即将拥塞窗口cwnd设置为最大拥塞窗口的1/2，执行拥塞避免算法，拥塞窗口按线性方式增长。

Socket：Socket是对TCP/IP协议族的一种封装，它把复杂的TCP/IP协议族隐藏在Socket接口后面，利用三元组（ip地址、协议、端口），就可以唯一标识网络中的进程。

UDP协议

概念：UDP是一种无连接、不可靠、面向报文的的传输层协议。UDP对报文除了提供一种可选的校验和外，几乎没有其它保证数据传输可靠性的措施。如果UDP检测出在收到的分组中有差错，它就丢弃这个分组。因此在QoS较差的网络环境中，UDP几乎无法运行，由于丢包、重复、失序等问题太严重。

计算校验和：UDP–IP协议组合中，保证数据正确性的唯一手段。分三部分：

- 伪报头：UDP数据报不包括伪报头。
- UDP报头：不包含目的IP地址和本地IP地址。
- 应用层数据：计算校验和。

多播和广播：UDP支持一对一、一对多、多对多的交互式通信，这一点TCP不支持。
双工性：支持全双工通信。

TCP和UDP的区别

1.是否有连接。TCP是面向连接的，三次握手建立连接，四次挥手解除连接。UDP无连接。
2.传输是否可靠。TCP依靠连接、重传、流量控制和报文序列号保证了传输的可靠性，与之相反的是，UDP的不可靠。
3.TCP的传输是面向字节流的，传输报文太大时会分成多个报文传输。UDP的传输是面向报文的，传过来多大的报文，都一次传输出去。
4.传输速度。TCP较慢，主要是拥塞控制、重传机制和确认机制都会消耗大量的时间，而且每个连接的管理与维护都要消耗计算机的内存和CPU资源。UDP较快，没有拥塞控制，对实时应用很有用。
5.TCP支持点对点的通信，UDP支持一对一、一对多、多对一、多对多的交互通信。

图解区别与联系：

	TCP	UDP
可靠性	可靠	不可靠
连接性	面向连接	无连接
报文	面向字节流	面向报文（保留报文的边界）
效率	传输效率低	传输效率高
双工性	全双工	一对一、一对多、多对一、多对多
流量控制	有（滑动窗口）	无
拥塞控制	有（慢开始、拥塞避免、快重传、快恢复）	无
传输速度	**慢**	**快**
应用场合	对效率要求相对低，但对准确性要求相对高；或者是要求有连接的场景	对效率要求相对高，对准确性要求相对低的场景
应用示例	TCP一般用于文件传输（FTP http 对数据准确性要求高，速度可以相对慢），发送或接收邮件（pop imap SMTP 对数据准确性要求高，非紧急应用），远程登录（telnet SSH 对数据准确性有一定要求，有连接的概念）等等；	UDP一般用于即时通信（QQ聊天 对数据准确性和丢包要求比较低，但速度必须快），在线视频（rtsp 速度一定要快，保证视频连续，但是偶尔花了一个图像帧，人们还是能接受的），网络语音电话（VoIP 语音数据包一般比较小，需要高速发送，偶尔断音或串音也没有问题）等等。

两种协议都是传输层协议，为应用层提供信息载体。tcp协议是基于连接的可靠协议，有流量控制和差错控制，也正因为有可靠性的保证和控制手段，所以传输效率比UDP低；UDP协议是基于无连接的不可靠协议，没有控制手段，仅仅是将数据发送给对方，因此效率比tcp要高。

参考资料：https://blog.csdn.net/u013777351/article/details/49226101

Cookie和Session的区别

前言：当客户端向服务端发起请求时，会要求服务器端产生一个session，服务器端先检查一下，客户端的请求报文中的cookie里面有没有sessionID，ID是否过期。如果存在可使用的sessionID，服务器会根据这个ID从服务器中检索出session。如果ID不可用，服务器端会重新建立一个。

1.Cookie存放在客户端，Session存放在服务器，但Cookie中包含了SessionID。
2.cookie不是很安全，别人可以分析存放在本地的cookie并进行cookie欺骗。
3.Session会在一定时间内保存在服务器上，当访问增多，会比较占用你服务器的性能，考虑到减轻服务器性能方面，应当使用cookie。
4.在浏览器中，每个站点的cookie的保存数量有限制。
5.重要信息应放在session中，其它信息如果需要保留，可以放在cookie中。
6.禁止掉cookie后，session还可以用，但是需要通过别的方式去传递这个sessionID。

参考资料：https://blog.csdn.net/wuhuagu_wuhuaguo/article/details/78552633

OSI模型、TCP/IP模型和五层协议

- **OSI模型：**即开放式系统互联，一般称为OSI参考模型，具有七层，如下：

  应用层：为应用程序提供服务
  表示层：数据格式转化、数据加密
  会话层：建立、管理和维护会话
  传输层：建立、管理和维护端到端的连接
  网络层：IP选址、分组转发及路由选择
  数据链路层：提供介质访问和链路管理
  物理层：信号的传输

  各层的解读：
  应用层：应用层有着诸多的网络服务协议：HTTP、HTTPS、FTP、POP3、SMTP等，应用层可以通过这些服务协议实现传输服务。
  表示层：表示层提供各种用于应用层数据的编码和转换功能，确保一个系统的应用层发送的数据能被另一个系统的应用层识别。另外，加密服务也在这一层实现。
  会话层：负责建立、还礼和终止表示层实体之间的通信会话。该层的通信由不同设备中的应用程序之间的服务请求组成。

传输层：传输层建立了主机端到端的链接。传输层的作用是为上层协议提供端到端的可靠和透明的数据传输服务，包括处理差错控制和流量控制等问题。TCP、UDP协议在这一层。

网络层：本层通过IP寻址来建立两个节点之间的连接，将源端的传输层送来的报文段或用户数据报封装成分组，选择合适的路由和交换节点，正确无误地按照地址传送给目的端的传输层。通常所说的IP层。一般的路由器是工作在网络层的。

数据链路层：还是主机之间的数据传输服务，为同一链路的主机提供数据传输服务，解决相邻主机通信问题，把分组封装成帧：为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。使用链路层地址（以太网使用MAC地址）来访问介质，并广泛使用了循环冗余检验（CRC）进行差错控制。常用的交换机工作在数据链路层。

物理层：定义物理设备标准,实际最终信号的传输是通过物理层实现的，通过物理介质传输比特流。

通信特点：对等通信，为了使数据分组从源传送到目的地，源端OSI模型的每一层都必须与目的端与对等层进行通信，这种通信方式称为对等层通信。

- **TCP/IP模型**

TCP/IP四层模型是我们实际上使用的模型，具有四层，如下：

应用层：为特定应用程序提供数据传输服务，对应OSI中的应用层、表示层和会话层。应用层确定进程之间通信的性质选择不同的协议以满足用户的需要。

传输层：为进程提供数据传输服务，解决进程间的通信。

网络层：为主机提供数据传输服务，解决跨网络的主机通信问题。

网络接口层：与OSI参考模型中的物理层和数据链路层相对应，它负责监视数据在主机和网络之间的交换。

- 五层体系结构
  - 概念:五层只是OSI和TCP/IP的综合,实际应用还是TCP/IP的四层结构.
  - 应用层
  - 运输层
  - 网络层
  - 数据链路层
  - 物理层

参考资料：https://www.cnblogs.com/qishui/p/5428938.html
三种体系结构的比较:https://www.cnblogs.com/wxd0108/p/7597216.html

物理层：

1.根据信息在传输线上的传送方向，分为以下三种通信方式：

- 单工通信：单向传输
- 半双工通信：双向交替传输
- 全双工通信：双向同时传输

2.除此之外，物理层还使用了带通调制，把离散的数字信号转换为连续的模拟信号。

数据链路层：

1.封装成帧：为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。

2.循环冗余检验：使用CRC协议检查比特差错

3.信道分类

一对多通信（广播信道）

冲突解决：频分复用、时分复用或CSMA/CD协议

频分服用（FDMA）：不同主机在相同时间占用不同的频率带宽资源。

时分复用（TDMA）：不同主机在不同时间占用相同的频率带宽资源。

这两种服用对信号的利用率不高，优化后的策略：统计时分复用、波分复用、码分复用（CDMA）

CSMA/CA协议(以太网使用)：表示载波监听多点接入/碰撞检测。

- 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。
- 载波监听：每个主机都必须不停地监听信道。在发送前，如果监听到信号正在使用，就必须等待。
- 碰撞检测：在发送中，如果监听到信道已有其它主机正在发送数据，就表示发生了碰撞。虽然每个主机在发送数据之前都已经监听到信道为空闲，但是由于电磁波的传播时延的存在，还是有可能会发生碰撞。当发生碰撞时，站点要停止发送，等待一段时间再发送。

一对一通信

PPP协议：互联网用户通常需要连接到某个 ISP 之后才能接入到互联网，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。

4.MAC地址：链路层地址，用于唯一标识网络适配器（网卡）。一台主机有多少个网络适配器就有多少个MAC地址，例如笔记本电脑普遍存在无线网络适配器和有线网络适配器，因此就有两个MAC地址。

5.局域网：现在广泛使用的是以太网，一种星型拓扑结构局域网。

需要注意的是，以太网使用交换机替代了集线器，交换机是一种链路层设备，它不会发生碰撞，能根据MAC地址进行存储转发。另外，交换机具有自学习能力，学习的是交换表的内容，交换表中存储着 MAC 地址到接口的映射

6.MTU:有效载荷的大小,最大为1500,不包括帧头和帧尾

网络层：

1.IP协议：

使用IP协议，把异构的物理网络连接起来，使得在网络层看起来好像是一个统一的网络。

与IP协议配套使用的还有三个协议：

- 地址解析协议ARP：ARP实现由IP地址得到MAC地址.主机发送信息时,将包含目标IP地址的ARP请求广播到网络上的所有主机,并接受返回信息,以此确定目标的物理地址.收到返回信息后将该IP地址和物理地址存入本机ARP缓存中并保存一定时间,下次请求时直接查询ARP缓存以节约资源。
- 网络控制报文协议ICMP：为了更有效地转发IP数据包和提高交付成功的机会的一个协议。它封装在IP数据包中，但不属于高层协议。应用：Plng，测试两台主机之间的连通性。
- 网际组管理协议IGMP

IP数据报格式：

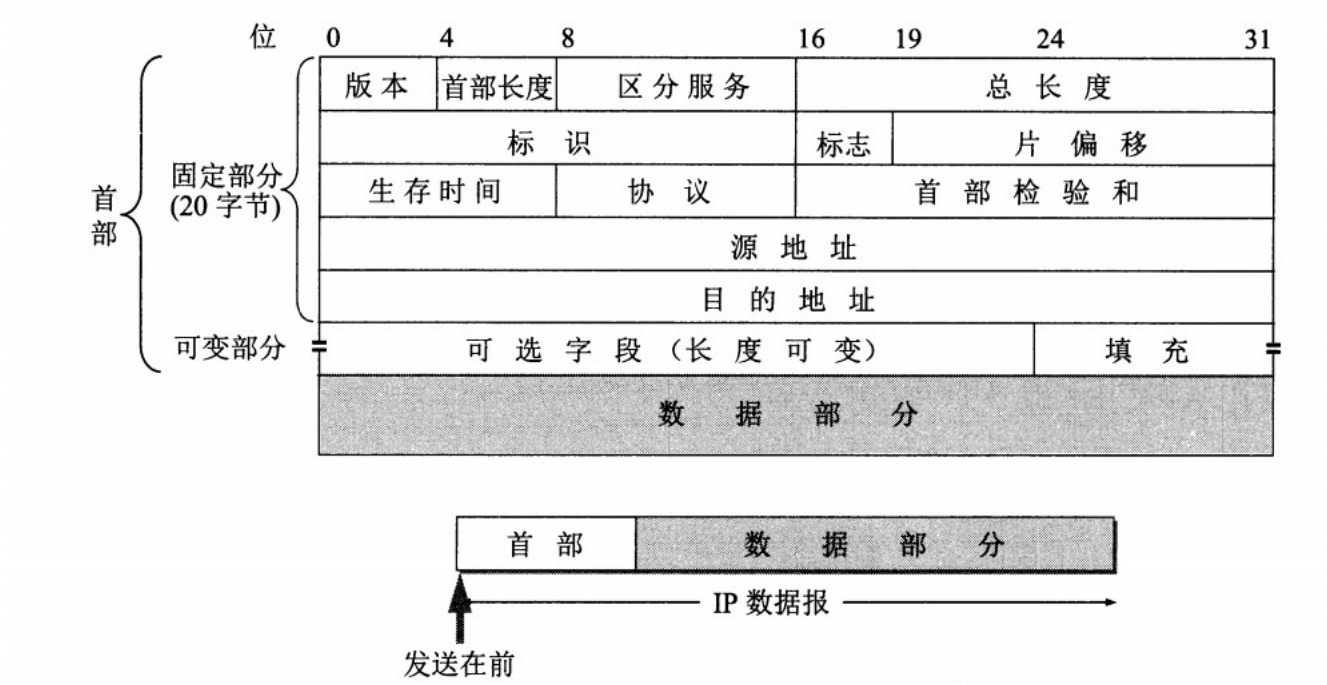

图 4-13 IP 数据报的格式

- **版本** : 有 4（IPv4）和 6（IPv6）两个值；
- **首部长度** : 占 4 位，因此最大值为 15。值为 1 表示的是 1 个 32 位字的长度，也就是 4 字节。因为首部固定长度为 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。
- **区分服务** : 用来获得更好的服务，一般情况下不使用。
- **总长度** : 包括首部长度和数据部分长度。
- **生存时间** ： TTL，它的存在是为了防止无法交付的数据报在互联网中不断兜圈子。以路由器跳数为单位，当 TTL 为 0 时就丢弃数据报。
- **协议** ： 指出携带的数据应该上交给哪个协议进行处理，例如 ICMP、TCP、UDP 等。
- **首部检验和** ： 因为数据报每经过一个路由器，都要重新计算检验和，因此检验和不包含数据部分可以减少计算的工作量。
- **标识** : 在数据报长度过长从而发生分片的情况下，相同数据报的不同分片具有相同的标识符。
- **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。

2.IP地址编址方式的三个历史阶段：

- 分类：由两部分组成，网络号和主机号，其中不同分类（A类、B类。。。）具有不同的网络号长度，并且是固定的。
- 子网划分：通过在主机号字段中拿一部分作为子网号，把两级IP地址划分为三级IP地址。注意，要使用子网，必须配置子网掩码。
- 无分类：无分类编址CIDR消除了传统A类、B类和C类地址以及划分子网的概念，使用网络前缀和主机号来对IP地址进行编码，网络前缀的长度可以根据需要变化。

3.路由器的结构：

路由器的功能包括：路由选择和分组转发

分组转发结构由三个部分组成：

- 交换结构
- 一组输入端口
- 一组输出端口

- **IP分组的概念**
  - 物理网络层一般要限制每次发送数据帧的最大长度，这个最大长度称为最大传输单元（MTU），如果待发送的IP数据报大小大于MTU，则需要进行分片，分片既可以发生在原始发送端主机上，也可以发生在中间路由器上，但数据报的重组只在目的主机端进行。
  - IP数据报分片后成为一个个的分组，这些分组都拥有各自的IP首部，路由选择也是相互独立的。需要注意的是，传输层首部只出现在第一片数据中。

分组转发流程：

- 从数据报的首部提取目的主机的 IP 地址 D，得到目的MAC网络地址 N。
- 若 N 就是与此路由器直接相连的某个网络地址，则进行直接交付；
- 若路由表中有目的地址为 D 的特定主机路由，则把数据报传送给表中所指明的下一个 路由器；
- 若路由表中有到达网络地址 N 的路由，则把数据报传送给路由表中所指明的下一个路由器；
- 若路由表中有一个默认路由，则把数据报传送给路由表中所指明的默认路由器；
- 报告转发分组出错。

路由选择使用的协议有：

- 内部网关协议RIP，一种基于距离向量的路由选择协议。
- 内部网关协议OSPF，开放最短路径优先OSPF，是为了克服RIP的缺点而开发出来的。
- 外部网关协议BGP，边界网关协议，BGP只能寻找一条比较好的路由路径，而不是最佳路由。

传输层

概念：网络层只是把分组发送到目的主机，但是真正通信的并不是主机而是主机中的进程。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。

网络环境中的进程识别：五元组，传输层协议、本地IP、本地端口、目的IP、目的端口。

1.UDP协议

UDP首部格式：

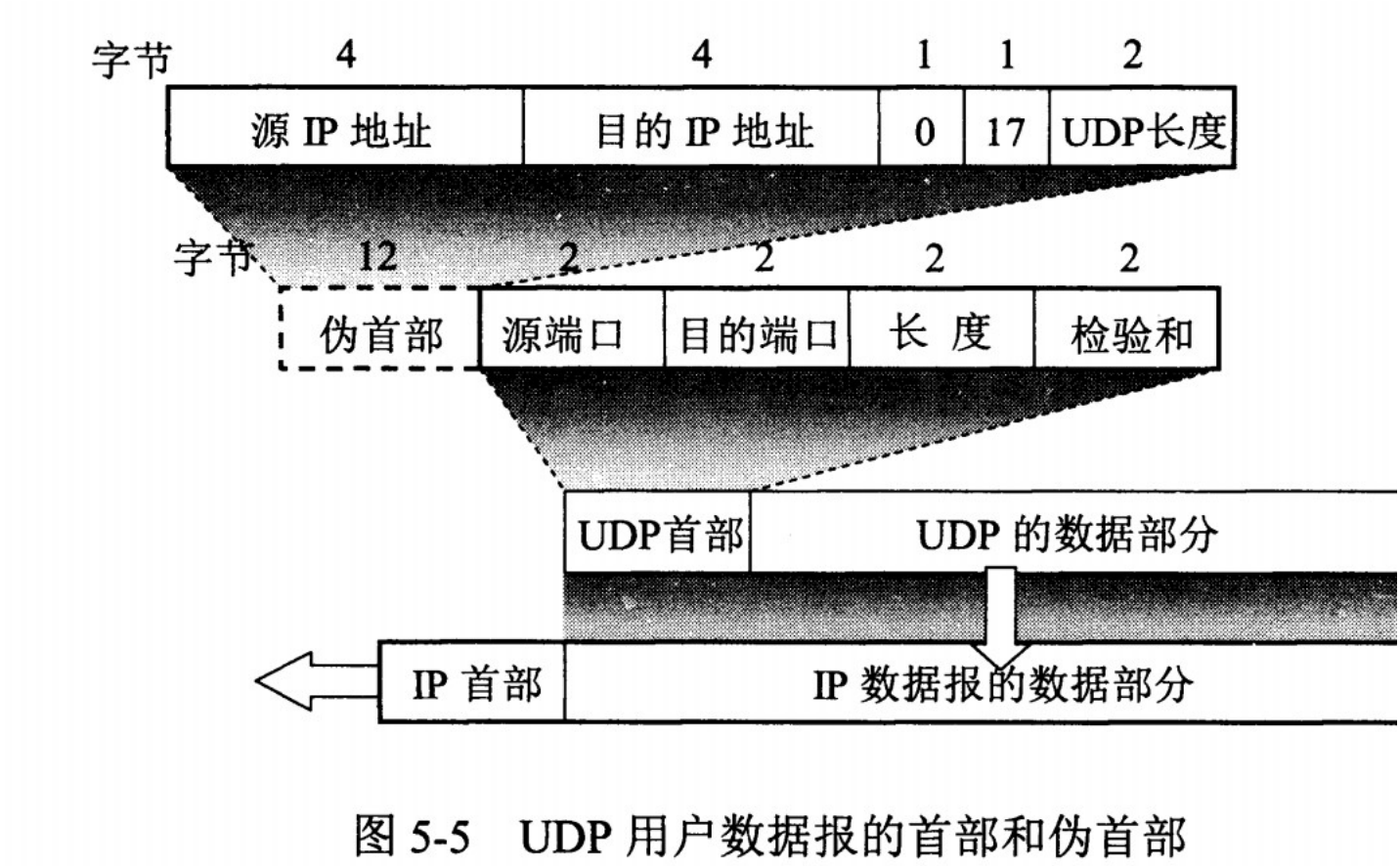

图 5-5 UDP 用户数据报的首部和伪首部

首部字段只有八个字节，包括源端口、母的端口、长度、校验和。12字节的伪首部是一个虚拟的数据结构，其中的信息是从数据报所在IP分组头的分组头中提取的，既不向下传送也不向上递交，仅仅是为了计算校验和临时添加的。

2.TCP协议

TCP首部格式：

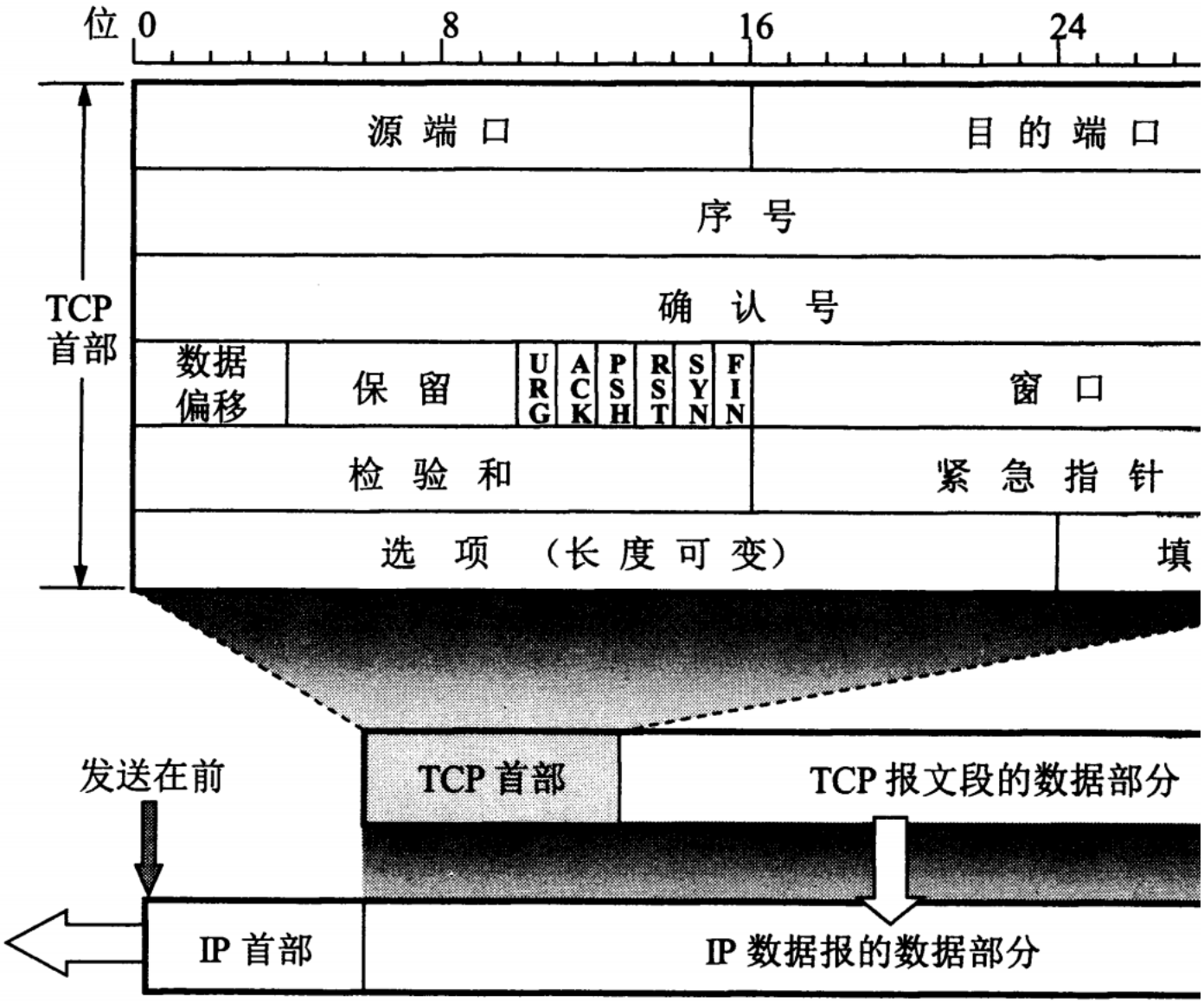

图 5-14 TCP 报文段的首部格式

TCP首部报文解读如下：

- 序号：用于对字节流进行编号，例如序号为301，表示第一个字节的编号为301，如果携带的数据长度为100字节，那么下一个报文段的序号应为401.
- 确认号：期望收到的下一个报文段的序号。
- 数据偏移：指的是数据部分距离报文段其实处的偏移量，实际上指的是首部的长度。
- 确认ACK：当ACK=1时确认字段有效，否则无效。TCP规定，在连接建立后所有的传送报文段都必须把ACK置为1.
- 同步SYN：在连接建立时用来同步序号，当SYN=1，ACK=0时表示这是一个连接请求报文段。若对方同意建立连接，则响应报文中，SYN=1，ACK=1.
- 终止FIN：用来释放一个连接，当FIN=1时，表示此报文段的发送方的数据已发送完毕，并要求释放连接。
- 窗口：窗口值是接收方设置的发送方发送窗口的大小。之所以要有这个限制，是因为接收方的数据缓存空间是有限的。

TCP可靠传输：TCP使用超时重传来实现可靠传输，即一个已经发送的报文段在设置的时间内没有收到确认，那么就重传。

TCP滑动窗口

窗口是缓存的一部分，用来暂时存放字节流。发送方和接收方各有一个窗口，接收方通过 TCP 报文段中的窗口字段告诉发送方自己的窗口大小，发送方根据这个值和其它信息设置自己的窗口大小。

TCP流量控制

流量控制可以通过设置窗口大小进行。

TCP拥塞控制

TCP主要通过四个算法来进行拥塞控制：慢开始、拥塞避免、快重传、快恢复。

应用层

- 域名系统（DNS）:DNS是一个分布式数据库，提供了主机名和IP地址之间相互转换的服务。这里的分布式数据库是指，每个站点只保留它自己的那部分数据。
- 文件传送协议（FTP）：使用TCP进行连接，它需要两个连接来传输一个文件：控制连接和数据连接
- 动态主机配置协议（DHCP）：提供了即插即用的联网方式。用户不再需要去手动配置IP地址等信息。DHCP配置的内容不仅是IP地址，还包括了子网掩码、网关IP地址。
- 远程登录协议（TELNET）：用于登录远程主机，并且远程主机上的输出也会返回。
- 电子邮件协议（SMTP、POP3和IMAP）

一个电子邮件系统由三部分组成：用户代理（客户端）、邮件服务器和邮件协议。邮件协议包括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：SMTP只能使用ASCII码，而MIME在SMTP的基础上，增加了邮件主题的结构，定义了非ASCII码的编码规则。

POP3:POP3的特点是只要用户从服务器上读取了邮件，就把该邮件删除。

IMAP：IMAP协议中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删除。IMAP这种做法可以让用户随时随地去访问服务器上的邮件。

6.Web网页请求过程

- DHCP配置主机信息
  - 假设主机最开始没有IP地址以及其它信息，那么就需要先使用DHCP来获取。
  - 主机生成一个DHCP请求报文，并将这个报文放入具有目的端口67和源端口68的UDP报文段中。
  - 该报文段则被放入一个具有广播IP目的地址和源IP地址的IP数据报中。
  - 该数据报则被放进在MAC帧中，该帧具有目的地址，将广播到与交换机连接的所有设备。
  - 连接在交换机的DHCP服务器收到广播帧后，不断向上分解得到IP数据报、UDP报文段、DHCP请求报文，之后生成DHCP ACK报文，该报文包含以下信息：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报文段又被放入IP数据报中，最后放入MAC帧中。
  - 该帧的目的地址就是请求主机的MAC地址，因为交换机具有自学习能力，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项，因此现在交换机就可以直接知道应该向哪个接口发送数据。
  - 主机在收到该帧后，向上分解得到DHCP报文。之后配置它的IP地址、子网掩码和DNS服务器的IP地址，并在其IP转发表中安装默认网关。

- ARP解析IP地址
  - 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。为了生成该套接字，主机需要知道网站的域名对应的IP地址。
  - 主机生成一个DNS查询报文，该报文具有53号端口，因为DNS服务器的端口是53.
  - 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。
  - 该IP数据报被放入一个以太网帧中，该帧将发送到网关路由器。
  - DHCP过程只知道网关路由器IP地址的ARP查询报文，为了获取网关路由器的MAC地址，需要使用ARP协议。
  - 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目的地址的以太网帧中，并向交换机发送以太网帧，交换机将该帧转发给所有的连接设备，包括网关路由器。
  - 网关路由器由接受到该帧后，向上分解得到ARP报文，得到其中的IP地址，若该IP地址与其接口的IP地址匹配，则发送一个ARP回答报文，包含了它的MAC地址，发回给主机。

- DNS解析域名
  - 知道了DNS网关路由器的MAC地址后，继续DNS的解析过程。
  - 网关路由器接受到包含DNS查询报文的以太网帧后，抽取出数据报，并根据转发表决定该IP数据报应该转发的路由器。（分组转发）
  - 因为路由器具有内部网关协议和外部网关协议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。
  - 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。
  - 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，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主机。

- HTTP请求页面
  - 获取到HTTP服务器的IP地址之后，主机生成TCP套接字，该套接字将用于向Web服务器发送HTTP GET报文。
  - 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。
  - 第一次握手，客户端生成一个具有目的端口80的TCP SYN=1报文段，并发送给HTTP服务器。
  - HTTP服务器收到该报文段后，生成TCP的SYN=1、ACK=1报文段，发回给主机。
  - 主机最后发送ACK=1确认报文给服务器，服务器收到后建立连接。
  - 连接建立后，主机可以根据需要发送GET、POST、PUT或DELETE报文给服务器，进行相应操作。

在最后，附上一个比较全面的，介绍网络协议的参考资料：
https://blog.csdn.net/qq1024884152/article/details/72312253
