**TCP / IP**

1. **OSI模型与TCP/IP模型的结构功能，都有那些协议。**

OSI（7层）：物理层、数据链路层、网络层、传输层、会话层、表示层、应用层。

TCP/IP（4层）：网络接口层、网际层、传输层、应用层。

五层协议：物理层、数据链路层、网络层、传输层、应用层。

应用层：通过进程间通信来实现特定的网络应用。应用层协议有：http、ftp、DNS..

表示层：数据格式处理，压缩与解压缩。

会话层：建立和维护会话。

传输层：为两进程间通信提供数据传输服务。常见传输协议：TCP面向链接、UDP..

功能：确定协议、确定目标端口。

网络层：负责处理网络上面不同主机间的通信服务。传输单位IP数据报常见协议IP。

功能：把本地IP地址转换成公网IP地址、并确定路由路径 (路由选择算法)。

数据链路层：将IP数据报封装成帧，进行MAC寻址、在相邻节点间的链路上传输帧。

物理层：以二进制形式在物理介质上传输数据。

1. **TCP和UDP的区别**

TCP：可靠的、面向连接的传输协议，TCP保证数据正确性，通过校验与确认、失败重传、滑动窗口、拥塞控制等机制，保证数据可靠不会丢包；传输速度比较慢、适合一些对数据有完整性、安全性需求的传输。 - 文件、邮件、超文本

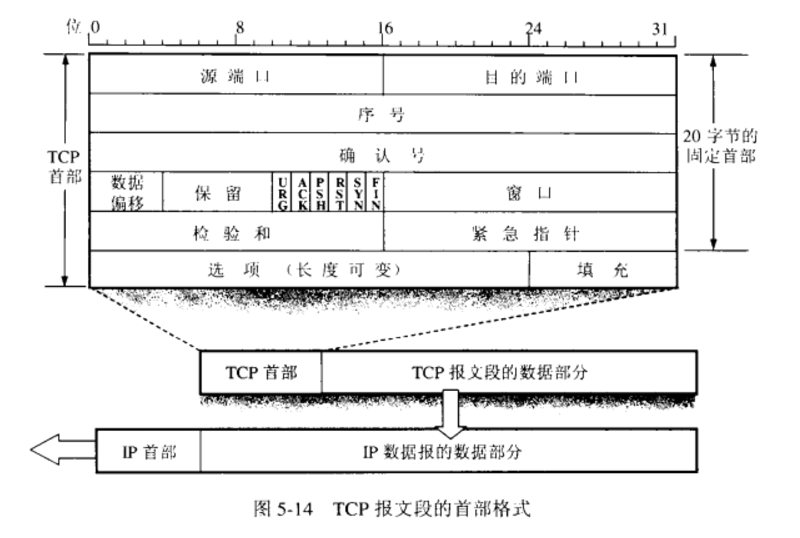
UDP：不可靠、面向无连接的传输协议，UDP不保证数据的正确性，可能丢包传输速度比较快、适合一些 - 视频、语音、直播

**2.1 tcp如何保证可靠的，丢包如何处理？**

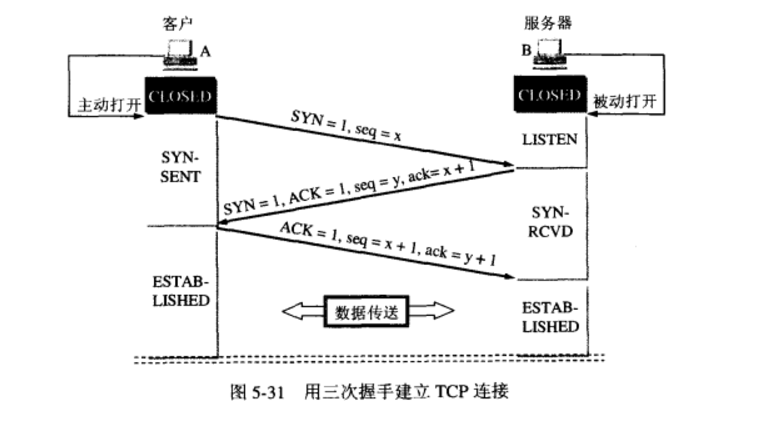
Tcp是面向连接的，只有在双方建立了连接后才能够进行数据的交互。

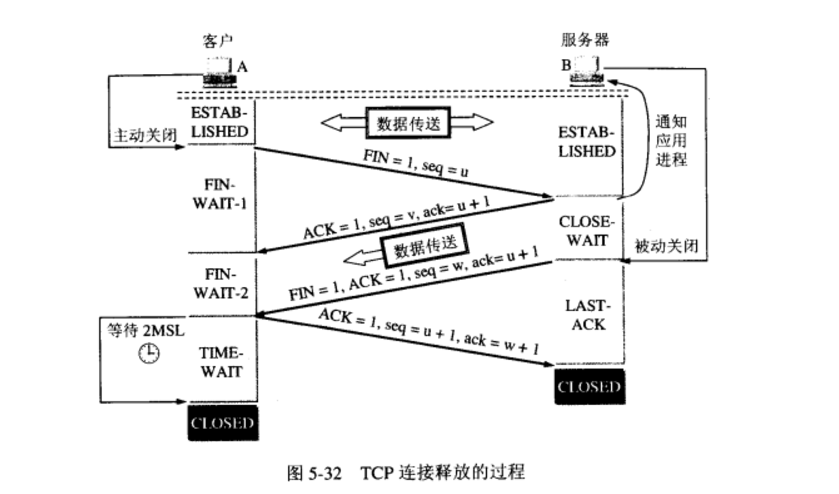
而且每一次数据接收都会进行数据包确认，超时或数据未接收都会触发重传机制。

1. **TCP报文结构**



1. **TCP的三次握手和四次挥手过程，各状态名称和含义，TIME-WAIT作用**





三次握手：CLOSE > SYN\_SENT > SYN\_RCVD > ESTABLASH(c) > ESTABLEASH(S)..

四次挥手：ESTABLASH > FIN WAIT-1 > CLOSE-WAIT > FIN WAIT-2

> CLOSE(s) > CLOSE(c)

TIME-WAIT作用：确保最后client发送的确认信息ACK能够到达service

为什么3次握手？

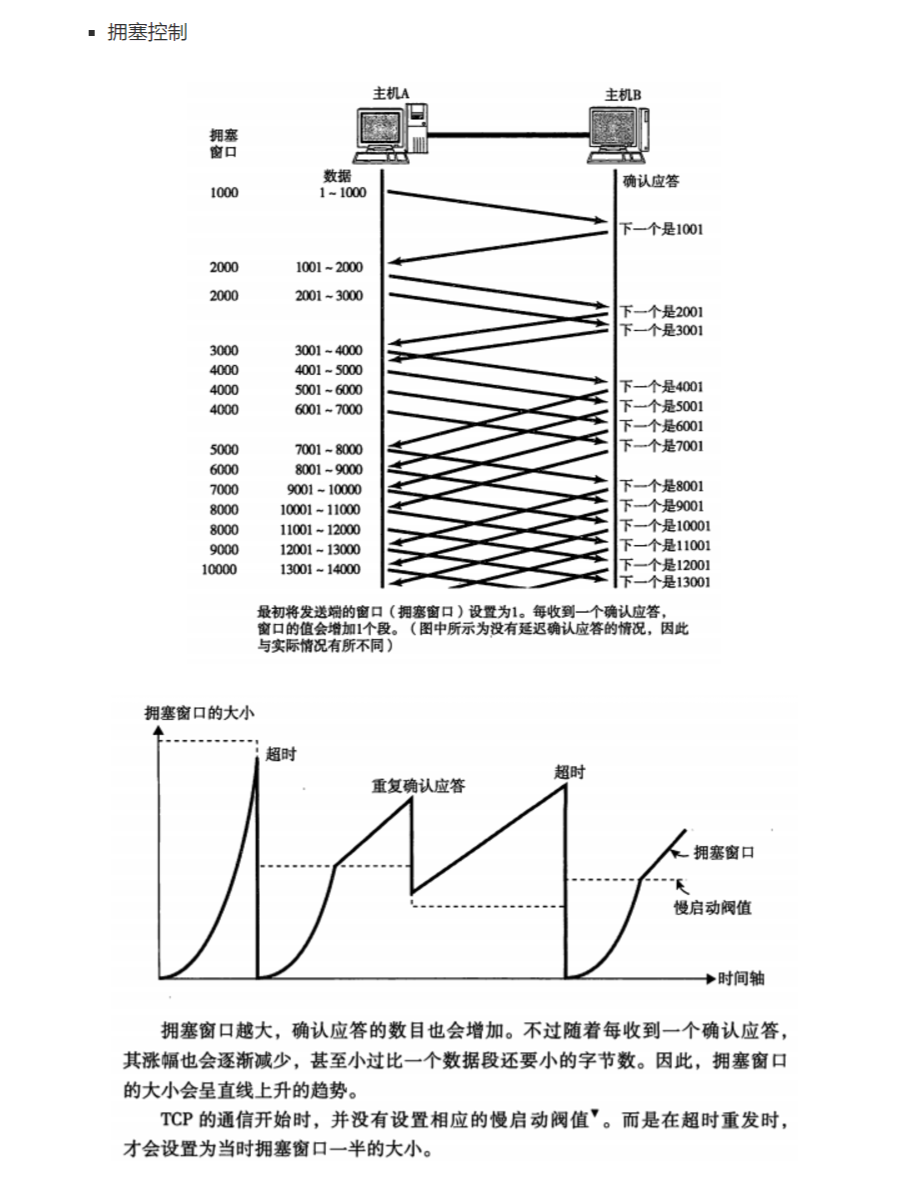
--- 为了防止失效的请求报文在错误的时间送到服务器端，防止server端一直等待，浪费资源。

为什么4次挥手？

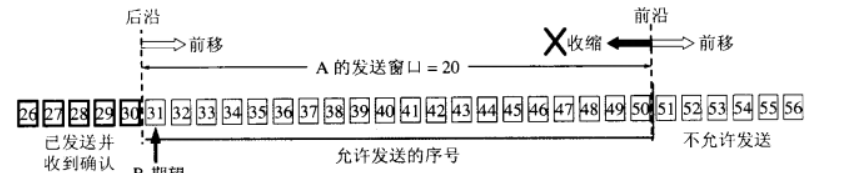
Tcp是全双工模式的，头两次挥手目的是确认发送端无数据要发送。而接收端此时不确定无数据返回。 因此需要把数据都返回通过2次挥手确认关闭链接。

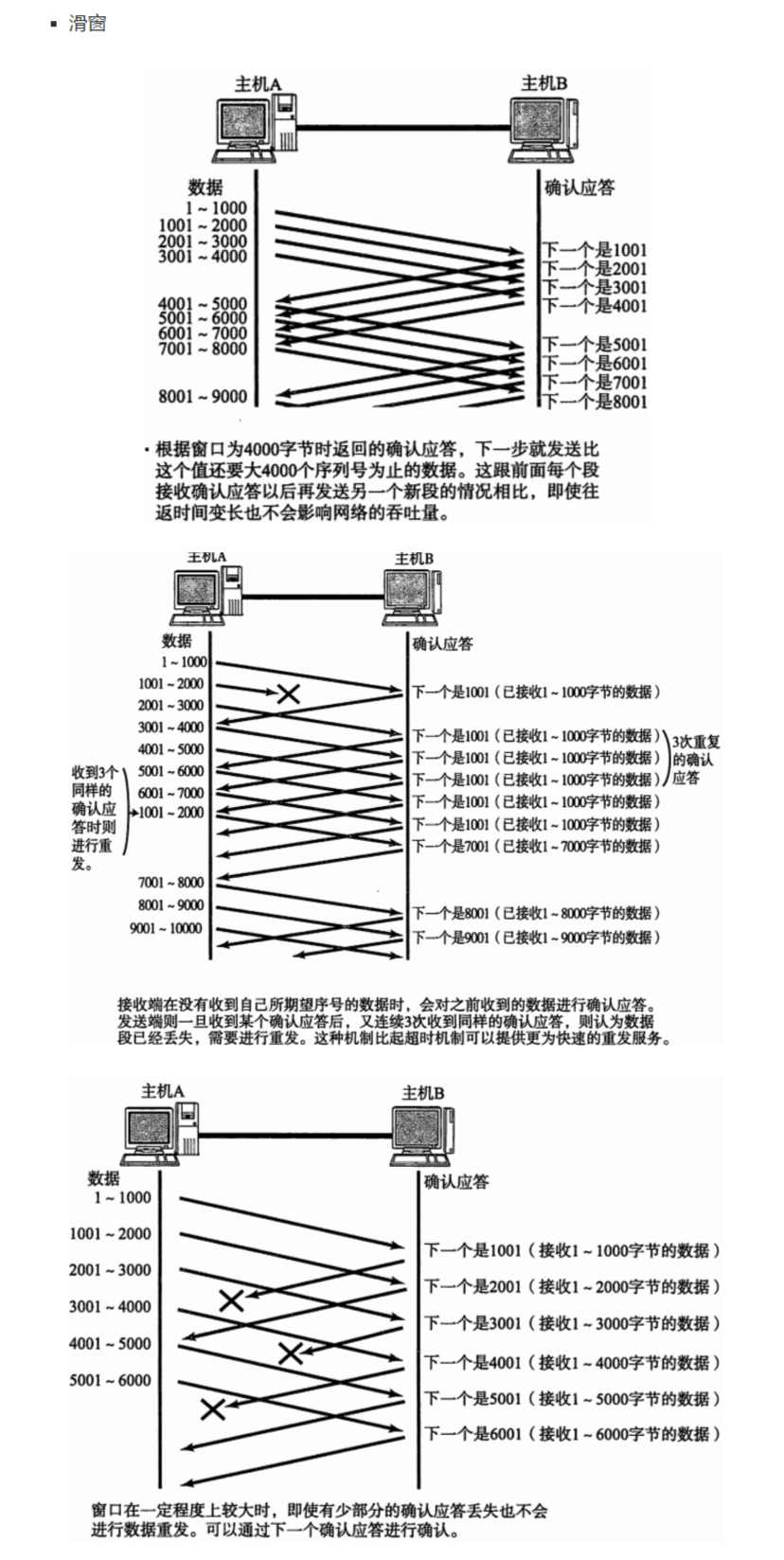
1. **TCP拥塞控制**

慢开始 2^k.. -> 拥塞避免 +1 >( 三个重复确认 )> 快开始



1. **TCP滑动窗口与回退N针协议**

接收-发送(缓冲），



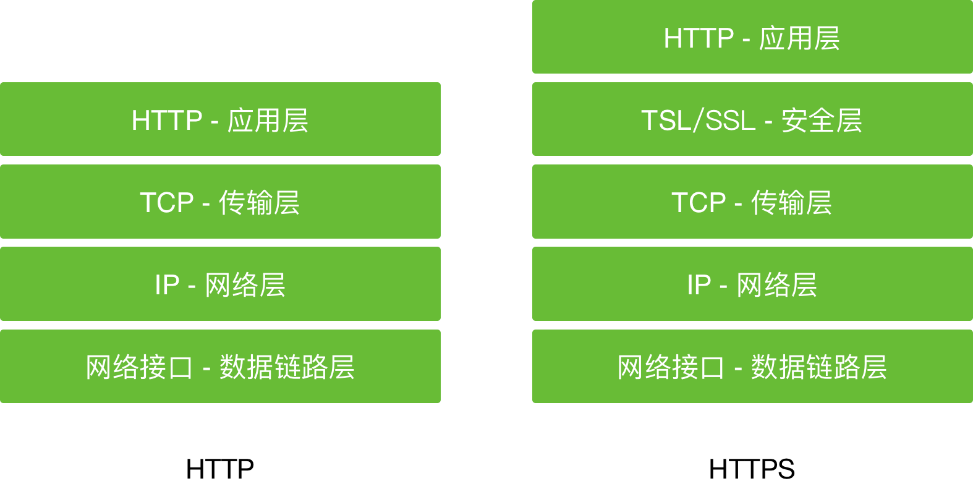
**回退N帧：**一次性发送窗口大小的数据帧，若某一帧出现错误，出现超时重传，就要回退到该帧，并重新发送该帧后面的所有数据帧。

**选择重传：**是回退N帧的改良版，使用缓冲技术，在某一帧出现错误重新回传时，只需回传该帧。而后面的数据帧都保留在接收端的缓冲区中。

**7.X．HTTP与HTTPS有什么区别？**

Http传输协议，数据信息是明文传输的，而Https则是具有安全性的SSL加密协议封装，而且Http跟Https使用的是完全不同的端口，一个是80，一个是443；

**简单的理解就是Https在Http的基础上做了一个SSL加密协议的封装，使我们数据的传输变得更安全；**



**1.** https协议需要到CA申请证书或自制证书。

**2.** http的信息是明文传输，https则是具有安全性的SSL加密。

**3.** http是直接与TCP进行数据传输，而https是经过一层SSL（OSI表示层），用的端口也不一样，前者是80（需要国内备案），后者是443。

**4.** http的连接很简单，是无状态的；HTTPS协议是由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。

5、Https会进行CA证书**验证阶段使用非对称加密**、在具体的数**据传递过程中会使用对称加密**。

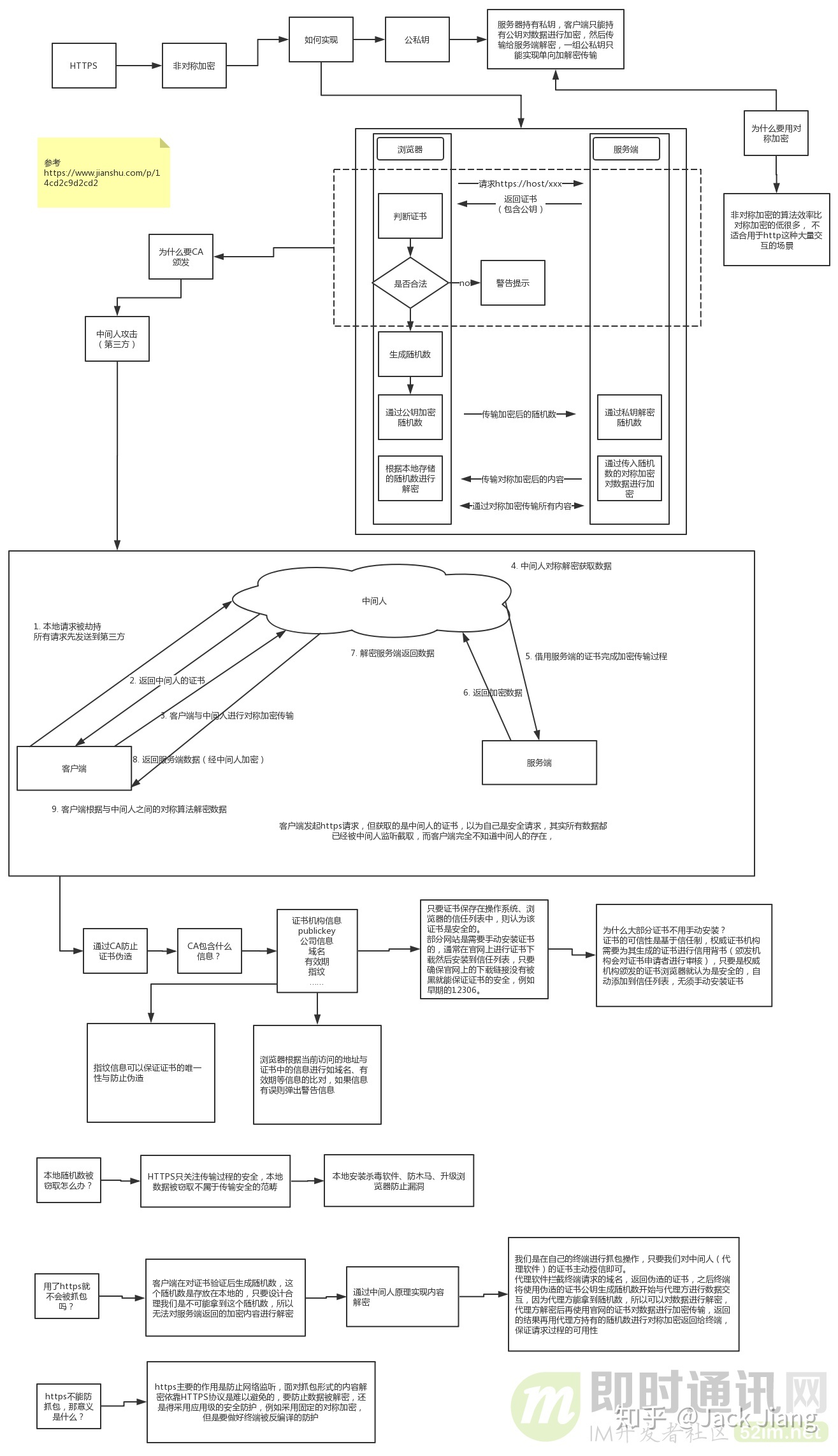
**Ps：SSL协议包含对称加密和非对称加密算法。**

**Https能够避免用户不知情的情况下被中间人监听网络数据，但是在用户授权的情况下(用户手动将中间人的安全证书安装进客户端)，中间人即可完成抓包或监听，比如fiddler、charles（即冒充服务器角色、又冒充客户端角色）**

**那么如何防止抓包？？**

**如果用户手动将中间人的伪造的CA证书装到客户端上，那么中间人就能够抓包或监听到我们传输的数据。就是说SSL原有的加密逻辑其实失效了，中间人能监听到我们传输的内容了。但是，我们客户端一般会有自己公司内部的一套加密算法，公司服务端也有一套相应的解密算法，这套算法一般是保密的不会公开的，这样我们就能够在请求的时候本地加密一些敏感的信息、比如账号密码等，这样即使中间人监听到拿到这些数据，也没办法对这些敏感信息进行解密，因为解密的秘钥在我们服务端那边。**

**也就是说我们在SSL协议加密算法的基础上，对我们请求或响应的敏感数据进行本地的加解密操作（公司内部的加密算法），这样中间人即使能抓到包也没法解析这些加密数据。**



1. **Http的报文结构**

请求报文：

请求行 --请求方法、请求uri、协议版本

请求头 --**key:val字段**（Connection:Keep-Alive，Cache-Control:no-cache，Accept-Encoding、Cookie、**If-Modified-Since：xxx）**

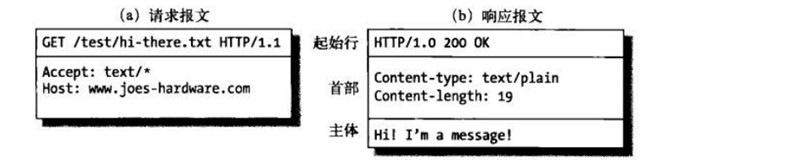
请求体 --内容实体

相应报文：

状态行 --状态码、协议版本

响应头 --..**key:val字段** （Set-Cookie、Age、Content-Type/Language、Cache-Control: max-age=3600、Last-Modified: Tue, 15 Nov 1994）

响应体 --响应内容



1. **HTTP状态码含义**

1xx：请求正在处理

2xx：请求成功

200:ok

204:请求成功，无回复

206:部分资源请求成功

3xx：资源重定向

4xx：客户端请求错误，服务器无法处理请求。

404：服务器无法找到资源。

5xx：服务器端处理请求出错。

500：服务器内部资源故障

503：服务器超负荷，宕机。

1. **Http request的几种类型**

GET：向服务器请求资源

POST：向服务器发送数据

PUT：向服务器上传文件至指定路径uri

DELETE：在服务器指定路径uri删除文件

**x**. GET和POST区别

GET：从服务器上获取数据，GET的参数信息会写在URL上，可见的。2KB限制

POST：向服务器发送数据，POST将数据信息写在请求体上，不可见的。大小不限。

了解：PUT向指定URL发送文件，DELETE向指定路径删除文件。

1. **Http1.0 和 Http1.1区别**

**Http1.0：**不支持长连接、每次请求都重新创建新连接、不支持断点续传。主要使用header里的If-Modified-Since,Expires来做为缓存判断的标准

**Http1.1**：支持长连接（默认，即同一个tcp链接可以传送多个http请求和响应）、支持断点续传。

**SPDY：**对Http1.X的优化，在应用层HTTP之下、会话层SSL之上，支持Header压缩、多路复用下的请求优先级、支持Https的协议加密。

**http2.0**：支持头部压缩Gzip、多路复用、流量控制。

1. **Http如何处理长连接**

在http1.0中，需要添加请求头：connection：keep-alive

在http1.1中，默认是保持长连接的。

**Ps：Http-Get请求的缓存字段头：cache-control、expires(新鲜度)、If-Modified-Scince、Last-Modified校验过时时间及缓存是否可用。**

1. **Cookie和Session的作用原理**

Cookie是基于客户浏览器端的技术，Session是基于服务器端的技术。

Cookie和Session的作用都是为了或临时保存一些用户信息。

Session是服务器端的，安全性系数相对Cookie会高一些。（用户敏感信息）

都能够设置失效时间。

1. **电脑上访问 [www.baidu.com](http://www.baidu.com) 过程是怎样的？**

1 首先浏览器通过域名服务器DNS解析出该域名对应的IP地址。

2 浏览器针对ip地址发起一个Http的请求（请求头、请求体..）

3 通过TCP进行封装，将http请求分成报文段，添加源端口、目的端口。

4 将数据包传至网络层，由网路层进行路由寻址。

5 数据到达服务器由服务器（一层层解析）处理请求并返回响应（一层层封装）。

6 浏览器获取资源数据，进行数据解析并显示。

1. **Ping的整个过程，ICMP报文是什么？**

1 使用ping命令

2 向目标主机发送ICMP包

3 目标主机接收到ICMP包

4 目标主机回复ICMP

**ICMP报文是基于网络层ip的协议，用来诊断网络信息。**

1. **C/S模式下使用socket通信，几个关键函数。**

* 服务端

创建ServerSocket，监听服务器端指定端口。

监听并接收客户端跟服务器端的链接：serversocket.accept();

* 客户端

创建一个Socket，指定对应的ip地址跟端口号。

通过socket发送字节/字符数据~

核心方法：

Accept() --侦听并接受到此套接字的连接。

Connect() --将此套接字连接到服务器。

Bind() --将套接字绑定到本地地址。

GetInputerStream --输入流

GetOutputStream --输出流

1. **IP地址分类**

32位二进制数表示，分为4字节，每个字节大小用10进制表示。

A类：以0开头的ip地址，前8位代表网络地址，后24位表示主机，0.0.0.0 ~ 127.0.0.0

B类：以10开头的ip地址，，前16位代表网络地址，后16位主机，128.0.0.0 ~ 191.255.0.0

C类：以110开头的ip地址，前24位表示网络地址，后8为表示主机。

192.0.0.0 ~ 223.255.255.0

D类：以1110开头，224.0.0.0 ~ 239.255.255.255

E类：240.0.0.0 ~ 255.255.255.255

1. **路由器和交换机的区别**

路由器：首先路由器是基于网路层的产品，作用是共享一个ip地址，让多台设备上网。

交换机：基于数据链路层的产品，简单来说，就是共享一条网线，接入不同的设备上网。

宿舍例子。

网关：连接两个不同的网络设备。不同协议网络的转换。

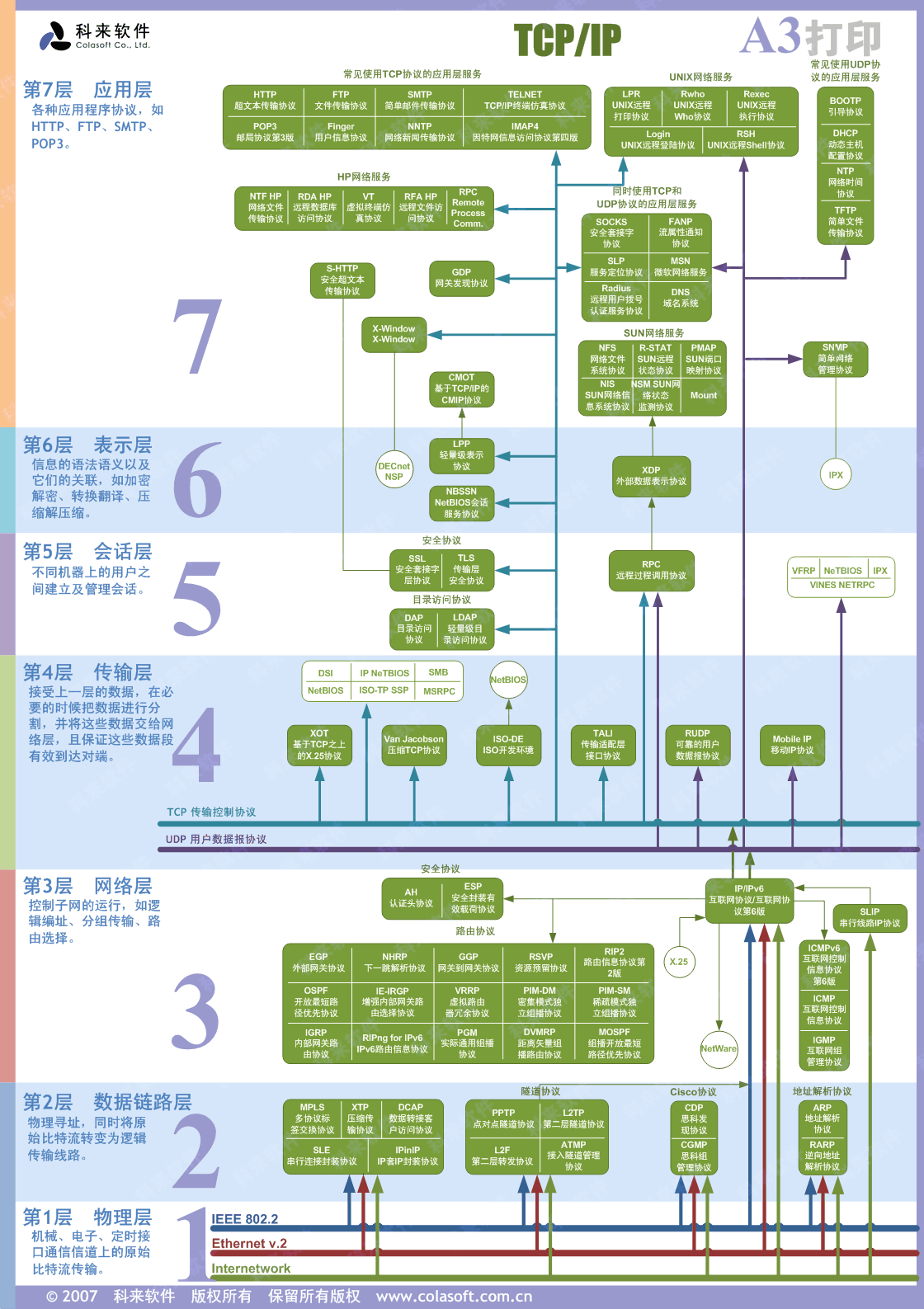
**Socket定义**

* 即套接字，**是一个对 TCP / IP协议进行封装 的编程调用接口（API）**
  1. 即通过Socket，我们才能在Andorid平台上通过 TCP协议进行开发
  2. Socket不是一种协议，而是一个编程调用接口（API），属于传输层（**主要解决数据如何在网络中传输的问题**）

**Socket是抽象的、不特指具体的协议，可以理解为是Java或JDK提供给我们的API，通过这API我们可以方便的进行网络数据通信，Socket本质上是对Tcp/ip协议的封装。**

**尝试去做一个网络框架（HttpUrlContextion、okHttp）：**

1. **底层通过Socket实现。**
2. **考虑线程创建、线程切换的问题、线程池的使用 (连接池的管理)。**
3. **考虑异常处理机制。**
4. **考虑数据缓存机制（缓存对于移动端是非常重要的存在、减少请求次数，减小服务器压力、本地数据读取速度更快、在无网络的情况下提供数据）。**
5. **数据返回的格式处理Converter。**
6. **完善API、暴露上层更易使用的接口方法。**
7. **Https。**



* 