简要说明ISO七层模型分别的作用是什么？

### **HTTP和HTTPS的区别**

首先HTTP和HTTPS的默认端口号就不一样，HTTP的默认端口号为80，HTTPS的默认端口号为443，HTTP在传输过程中使用的是明文 传输，内容可能被窃取，而且无法验证通信方的身份，还有可能遭遇身份伪装，而HTTPS在应用层和传输层之间增加了ssl协议用来加密 内容，因此通过证书验证来验证身份，即使数据被窃取也无法解密，数据的传输更加安全。

### **ssl加密使用了那种算法，如何加密**

### **TCP三次握手的过程，为什么是三次而不是两次或者四次？**

第一次握手：客户端发送一个syn（同步）包（syn=x）给服务器，进入SYN\_SEND状态，等待服务器确认

第二次握手：服务端收到客户端发送的同步包，确认客户端的同步请求（ack=x+1）,同时也发送一个同步包， 也就是一个ACK包+SYN包服务器进入SYN\_RECV状态

第三次握手：客户端收到服务器的SYN+ACK包，向服务器发送一个确认包，此包发送完毕，客户端和服务器进入 ESTABLISHED状态，完成三次握手

不是两次是为了防止已经失效的连接请求报文段突然又传送到了服务端，因而产生错误，比如有一个因网络延迟的请求 发送到了服务端，服务端收到这个同步报文之后进行确认，如果此时是两次握手，那么此时连接建立，但是客户端并没有发出 建立连接的请求，服务端却一直等待客户端发送数据，这样服务端的资源就白白浪费了。

不是四次的话是因为完全没有必要，三次已经足够了

### **TCP的四次挥手**

第一次：主动关闭方发送一个FIN包，用来关闭主动关闭方到被动关闭方的数据传送，也就是告诉另一方我不再发送数据了，但此时仍可以接收数据

第二次：被动关闭方收到FIN包之后，发送一个确认（ACK）包给对方

第三次：被动关闭方发送一个FIN包，告诉对方不带发送数据

第四次：主动关闭方收到FIN包之后，发送一个ACK包给对方，至此完成四次挥手

### **HTTP报文的格式，传输中以何种方式传输**

HTTP报文分为三个部分，起始行、首部和主体，其中起始行和首部以一个回车和换行符分隔，首部和主体以一个空行分隔，其中起始行是对这次HTTP请求或者响应 的描述，请求报文的起始行包括使用的HTTP方法、请求的url地址、HTTP版本，响应报文的起始行包括HTTP的版本，HTTP状态码，http状态码的描述，首部也 就是常说的HTTP头部，如Date、Cookie、Content-Type等，主体是这次请求或响应的数据，传输中以明文传输。

### **常见的HTTP头部**

可以将HTTP首部分为通用首部、请求首部、响应首部、实体首部，通用首部表示一些通用信息，如Date表示报文创建时间，请求首部就是请求报文中 独有的，如cookie、和缓存相关的If-Modified-Since，响应首部就是响应报文中独有的，如set-cookie和重定向有关的location，实体首部用来 描述实体部分，如Allow用来描述可执行的请求方法，Content-Type描述主体类型，Content-Encoding描述主体的编码方式

### **HTTP状态的简要分类**

可以按照HTTP状态码的第一个数字分类，1xx表示信息，2xx表示成功，3xx表示重定向，这里需要注意的是304，表示未修改， 4xx表示客户端错误，最常见的是404，5xx表示服务端错误

### **HTTP状态码101、200、301、302、304的具体含义**

101：切换协议 200：正常，OK，301：永久重定向，302：临时重定向，304：未修改

### **301和302的区别**

### **.简要介绍一次302的过程**

用户请求一个url，服务器处理这个url，设置这个url需要重定向，返回用户一个302响应，并在http头部设置location字段为新的地址， 浏览器得到这个响应，根据location中新的地址重新发起一次请求。

### **.HTTP2.0的简要介绍**

### **用户登陆过程的简要说明，如何判断用户是否登录？**

用户输入用户名和密码，通过post请求将密码和用户名发送给服务器，服务器比对收到的用户名、密码和数据库中的数据进行比对，不一致则做出响应， 反馈信息给客户端，如果比对一致则服务端生成一个session，这个session可以存储在内存、文件、数据库中，同时生成一个与之一一对应的sessionID 作为cookie发送给客户端，比对成功之后反馈信息，这时一般会进行一次重定向，重定向至登陆之后的默认页面。判断用户登录则是根据这个sessionID，每次请求 会先检查有没有这次类似sessionID的cookie发送过来，没有则认为没有登录，有则是否有相应的session，这个session是否过期等，来判断用户是否登录， 登录是否过期。

### **tcp和udp的区别**

TCP面向连接的、提供可靠传输的协议，而UDP则是面向非连接。不可靠传输的协议，之所以说TCP是可靠的传输协议是因为TCP协议在传输数据之前有一个确认双方是否 连接的过程，而UDP没有，也正是因此在传输速度方面，UDP更快。因此需要可靠传输需要选用TCP，不需要可靠传输情况下选择UDP。

### **udp的阻塞机制，如何处理**

### **简要介绍一下socket协议**

### **.什么是正向代理？什么是反向代理？**

正向代理就是客户端向代理服务器发送请求，并且指定目标服务器，之后代理向目标服务器转交并且将获得的内容返回给客户端。比如翻墙 反向代理的话代理会判断请求走向何处，并将请求转交给客户端，客户端只会觉得这个代理是一个真正的服务器。如负载均衡。

### **介绍一下HTTPS的连接过程**

### **和缓存有关的请求头有哪些？优先级是怎样的？**

和缓存有关的请求头有Cache-Control、If-Match、If-None-Match、If-Modified-Since、If-Unmodified-Since，在缓存中 总体来说是Cache-Control优先于Expires，Cache-Control中会需要检测Cache-Control是否过期，过期的话检验会优先检测Etag， 也就是If-Match、If-None-Match,不一致则验证Last-Modify请求头也就是If-Modified-Since、If-Unmodified-Since。

### **介绍一下DNS的查找过程？**

### **http连接性能优化，长连接，keep-alive**

### **https的详细过程，使用的加密算法，是对称加密算法还是非对称加密算法。md5、SHA、AES分别是对称加密的还是非对称加密的**

### **https没有大规模应用的原因**

### **http2有哪些新特性？**

### **端口号的作用是什么？**

HTTP协议的主要特点

* 简单快速：每个URI是固定的
* 灵活：可以完成不同数据类型的传输
* \*无连接：连接一次就会断掉，不会一直连着
* \*无状态：http不能区分两次连接的不同

HTTP报文的组成部分

请求报文

* 请求行（http方法+页面地址+http协议版本）
* 请求头（key：value值）
* 空行（头和体的分隔标识）
* 请求体

响应报文

* 状态行（http协议+版本+状态码）
* 响应头
* 空行
* 响应体

HTTP方法

* GET->获取资源
* POST->传输资源
* PUT->更新资源
* DELETE->删除资源
* HEAD->获得报文首部

POST和GET的区别

* GET在浏览器回退时是无害的，而POST会再次提交请求
* GET产生的URI地址可以被收藏，而POST不可以
* GET请求会被浏览器主动缓存，而POST不会，除非手动设置
* GET请求参数会被完整保留在浏览器历史记录里，而POST中的参数不会保留
* GET请求在URL中传输的参数是有长度限制的，而POST没有限制
* GET参数通过URL传递，POST放在Request body中

HTTP状态码

* 1xx：指示信息-表示请求已接收，继续处理
* 2xx：成功-表示请求已被成功接收
* 3xx：重定向-要完成请求必须进行更进一步的操作
* 4xx：客户端错误-请求有语法错误或请求无法实现
* 5xx：服务器错误-服务器未能实现合法的请求
* 200：服务器成功返回
* 206：客户端表明自己只需要目标URL上的部分资源
* 301：永久重定向
* 302：临时重定向
* 304：缓存(服务器告诉客户，原来缓冲的文档还可以继续使用)
* 400：请求出错
* 401：被请求的页面需要用户名和密码
* 403：服务器拒绝访问
* 404：资源找不到
* 500：服务器错误
* 503：请求未完成。服务器临时过载或当机

什么是持久连接

HTTP协议采用“请求-应答”模式，当使用普通模式，即非keep-alive模式时，每次连接都需要重新建立

* 持久连接只有在HTTP1.1版本才支持

什么是管线化

持久连接的情况下，请求和连接是对应的，即：  
请求1-->响应1-->请求2-->响应2

管线化是指：将请求和响应都进行了打包进行发送  
请求1、2、3-->响应1、2、3

* 管线化是通过持久连接完成的，至少1.1版本
* 只有GET和HEAD请求可以进行管线化，POST有所限制
* 第一次建立时不应该启动管线化，当前版本不一定支持持久连接
* 开启管线化很可能不会带来大幅度的性能提升，而且很多服务器端和代理程序对管线化的支持并不好，因此现代浏览器，如Chrome和Firefox是默认关闭管线化的

常见HTTP首部字段

通用首部字段（请求报文与响应报文都会使用的首部字段）

* Date：创建报文时间
* Connection：连接的管理
* Cache-Control：缓存的控制
* Transfer-Encoding：报文主体的传输编码方式

请求首部字段（请求报文会使用的首部字段）

* Host：请求资源所在服务器
* Accept：可处理的媒体类型
* Accept-Charset：可接收的字符集
* Accept-Encoding：可接受的内容编码
* Accept-Language：可接受的自然语言

响应首部字段（响应报文会使用的首部字段）

* Accept-Ranges：可接受的字节范围
* Location：令客户端重新定向到的URI
* Server：HTTP服务器的安装信息

实体首部字段（请求报文与响应报文的的实体部分使用的首部字段）

* Allow：资源可支持的HTTP方法
* Content-Type：实体主类的类型
* Content-Encoding：实体主体适用的编码方式
* Content-Language：实体主体的自然语言
* Content-Length：实体主体的的字节数
* Content-Range：实体主体的位置范围，一般用于发出部分请求时使用

HTTPS对于HTTP的优点

* 通信使用明文不加密，内容可能被窃听
* 不验证通信方身份，可能遭到伪装
* 无法验证报文完整性，可能被篡改

HTTPS就是HTTP加上加密处理（一般是SSL安全通信线路）+认证+完整性保护