1. TCP建立连接的过程。

三次握手：

1. 第一次握手(客户端发送syn包到服务器端)：客户端发送syn包到服务器端，进入syn_send状态，等待服务器端的确认；

2. 第二次握手(服务器返回syn+ack包给客户端)：服务器端收到客户端的syn包，发送syn+ack包给客户端，进入syn_recv状态；

3. 第三次握手(客服端返回ack包给服务端)：客户端收到服务器端的syn+ack包，发送个ack包到服务器端，至此，客户端与服务器端进入established状态；

4. 握手过程中传送的包不包含任何数据，连接建立后才会开始传送数据，理想状态下，TCP连接一旦建立，在通信双方的任何一方主动关闭连接前，TCP连接都会一直保持下去。

2. TCP断开连接的过程。

四次挥手：

1. 第一次挥手：主动关闭方发送fin包到被动关闭方，告诉被动关闭方我不会再给你发送数据了；

2. 第二次挥手：被动关闭方收到syn包，发送ack给对方，确认序号为收到序号+1；

3. 第三次挥手：被动关闭方也也发送fin包给主动关闭方，告诉对方我也不会给你发送数据了；

4. 第四次挥手：主动关闭方收到syn包，发送ack给对方，至此，完成四次挥手；

3. 浏览器发生302跳转背后的逻辑?

浏览器在原请求地址的响应的Location域找到要跳转的URI执行跳转。

浏览器输入URL后发生了什么

1.DNS域名解析；

2.建立TCP连接；

3.发送HTTP请求；

4.服务器处理请求；

5.返回响应结果；

6.关闭TCP连接；

7.浏览器解析HTML；

8.浏览器布局渲染；

4. HTTP协议的交互流程。 HTTP和HTTPS的差异， SSL的交互流程?

a. Http协议

1、建立TCP连接；

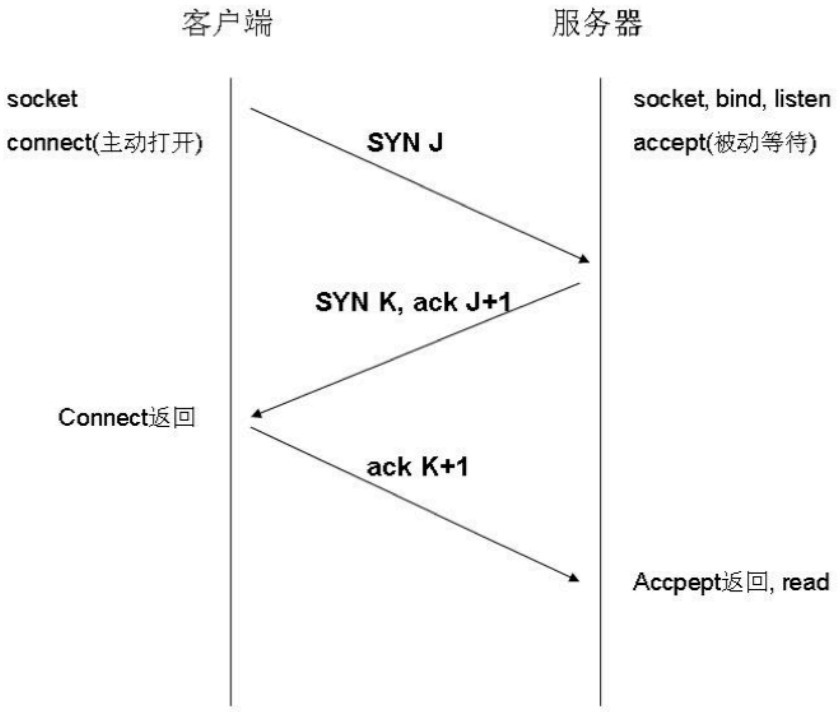
2、发送HTTP请求；

3、服务器处理请求；

4、返回响应结果；

5、关闭TCP连接；

b. http三次握手：



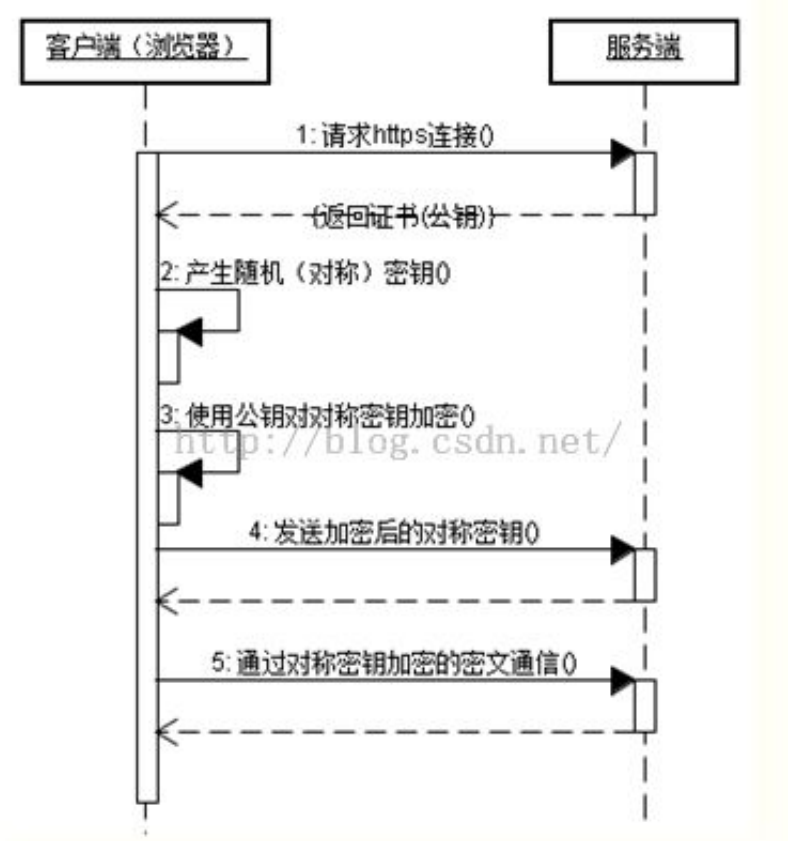
i. 第一次握手：客户端发送syn包(syn=j)到服务器，并进入SYN_SEND状态，等待服务器确认；

ii. 第二次握手：服务器收到syn包，必须确认客户的SYN（ack=j+1），同时自己也发送一个SYN包（syn=k），即SYN+ACK包，此时服务器进入SYN_RECV状态；

iii. 第三次握手：客户端收到服务器的SYN＋ACK包，向服务器发送确认包ACK(ack=k+1)，此包发送完毕，客户端和服务器进入ESTABLISHED状态，完成三次握手。

c. HTTPS协议

HTTPS协议就是基于SSL的HTTP协议
HTTPS使用与HTTP不同的端口（HTTP:80 ， HTTPS:443）
提供了身份验证与加密通信方法，被广泛用于互联网上安全敏感的通信。



http://blog.csdn.net/

1、客户端请求SSL连接，并将自己支持的加密规则发给网站。
2、服务器端将自己的身份信息以证书形式发回给客户端。证书里面包含了网站地址，加密公钥，以及证书的颁发机构。
3、获得证书后，客户要做以下工作

- 验证证书合法性
- 如果证书受信任，客户端会生成一串随机数的密码，并用证书提供的公钥进行加密。
- 将加密好的随机数发给服务器。

4、获得到客户端发的加密了的随机数之后，服务器用自己的私钥进行解密，得到这个随机数，把这个随机数作为对称加密的密钥。（利用非对称加密传输对称加密的密钥）
5、之后服务器与客户之间就可以用随机数对各自的信息进行加密，解密。

注意的是：证书是一个公钥，这个公钥是进行加密用的。而私钥是进行解密用的。公钥任何都知道，私钥只有自己知道。这是非对称加密。

而对称加密就是钥匙只有一把，我们都知道。

之所以用到对称加密，是因为对称加密的速度更快。而非对称加密的可靠性更高。

客户端请求--服务端发送证书（公钥）--客户端验证证书，并生成随机数，通过公钥加密后发送给服务端--服务端用私钥解密出随机数--对称加密传输数据。

d. HTTP与HTTPS的区别

1、HTTPS协议需要申请证书。

2、HTTP是明文传输；HTTPS使用的是具有安全性的SSL加密传输协议

3、HTTP端口是80；HTTPS端口号是443

4、HTTP连接简单无状态；HTTPS由SSL+HTTP协议构件的可进行加密传输、身份验证的网络协议。

5. Rest和Http什么关系？大家都说Rest很轻置，你对Rest风格如何理解？

a. Http是一种协议，Rest是一种软件架构风格。

b. URL定位资源，用HTTP动词（GET,POST,DELETE,DETC）描述操作。

c. GET表示查询、POST表示新建、PUT表示更新、DELETE表示删除等。

```
GET /api/v1/user 获取用户列表
GET /api/v1/user/1 获取ID为1的用户
POST /api/v1/user 新建用户
PUT /api/v1/user/1 更新ID为1的用户信息
DELETE /api/v1/user/1 删除ID为1的用户
```

1、概念：REST（英文：Representational State Transfer，简称REST，表现层状态转化），指的是一组架构约束条件和原则。满足这些约束条件和原则的应用程序或设计就是 RESTful。

2、一种软件架构风格，设计风格而不是标准，只是提供了一组设计原则和约束条件。它主要用于客户端和服务器交互类的软件。基于这个风格设计的软件可以更简洁，更有层次，更易于实现缓存等机制。

3、Restful架构：

（1）每一个URI代表一种资源；

（2）客户端和服务器之间，传递这种资源的某种表现层；

（3）客户端通过四个HTTP动词(GET用来获取资源，POST用来新建资源（也可以用于更新资源），PUT用来更新资源，DELETE用来删除资源。)，对服务器端资源进行操作，实现"表现层状态转化”。

6. ~~TCP的滑动窗口协议有什么用？讲讲原理。~~

滑动窗口协议是传输层进行流控的一种措施，接收方通过通告发送方自己的窗口大小，从而控制发送方的发送速度，从而达到防止发送方发送速度过快而导致来不及接受。

7. HTTP协议都有哪些方法?

1. GET 请求获取由Request-URI所标识的资源。
2. POST 在Request-URI所标识的资源后附加新的数据。
3. HEAD 请求获取由Request-URI所标识的资源的响应消息报头。
4. OPTIONS 请求查询服务器的性能，或查询与资源相关的选项和需求。
5. PUT 请求服务器存储一个资源，并用Request-URI作为其标识。
6. DELETE 请求服务器删除由Request-URI所标识的资源。
7. TRACE 请求服务器回送收到的请求信息，主要用语测试或诊断。

8. 交换机与路由器的区别?

1、工作层次不同

- 最初的交换机工作在OSI模型中的数据链路层，工作原理简单
- 路由器工作在OSI模型中的网络层，得更多协议信息，做更智能的转发决策

2、数据转发所依据的对象不同

- 交换机是利用物理地址（MAC地址），确定转发的目的地址。（MAC固化硬件，一般不可更改）
- 路由器是利用IP地址，确定转发的目的地址。（IP通常为网关或系统自动分配的）

3、是否可以分割广播域

- 传统的交换机可以分割冲突域，不能分割广播域，而路由器可以分割广播域
- 由交换机连接的网段仍然属于同一广播域，广播数据报会在交换机连接的所有网段上传播，某些情况导致通信拥挤和安全漏洞。连

接到路由器上的网段被分配成不同的广播域，所以，广播数据不穿过路由器

虽然三层交换机可以分割广播域，但是子广播域之间不能通信，还是需要路由器

4、路由器提供了防火墙的服务

- 路由器仅仅转发特定地址的数据包，不传送不支持路由协议的数据包，不传送未知目标网络数据包，从而可以防止广播风暴

5、表

二层交换机上存在MAC表，三层交换机上存在路由表、MAC表、ARP表，路由器上存在路由表和ARP表。

总之，交换机在具体的城域网中扮演着VLAN透传的角色，就是桥。路由器的每一个端口都是一个独立的广播域和冲突域，而交换机是只有一个广播域和端口数量的冲突域。

9. Socket网络通信、NIO流以及多线程处理技术，Netty、Mina?

1、socket网络通信：

NIO流以及多线程处理技术：

BIO:阻塞式，线程池初始时创建一定量线程，超过则等待；

NIO:非阻塞式，不同的线程干专业的事情，提高系统吞吐量；

NIO+异步处理：让少量的线程做大量的事情；

2、Mina：Apache Mina是一个能够帮助用户开发高性能和高伸缩性网络应用程序的框架。它通过Java nio技术基于TCP/IP和UDP/IP协议提供了抽象的、事件驱动的、异步的API。采用非阻塞方式的异步传输，支持批量传输数据。mina框架简单高效，完成了底层的线程管理，内置编码器能够满足大多数用户的需求，省去了消息编码和解码的工作。

3、Netty：本质是JBoss开发的一个jar包，目的是开发高性能、高可靠性的网络服务和客户端服务；提供异步非阻塞的、事件驱动的网络应用程序的NIO框架和工具；处理socket；通过Future-Listener机制，用户可以方便的主动获取或者通过通知机制获得IO操作结果。

10. http协议（报文结构，断点续传，多线程下载，什么是长连接）

a. 概念：

i. HTTP协议是Hyper Text Transfer Protocol（超文本传输协议）的缩写,是用于从万维网（WWW:World Wide Web ）服务器传输超文本到本地浏览器的传送协议。

ii. HTTP是一个基于TCP/IP通信协议来传递数据（HTML 文件, 图片

文件, 查询结果等）。

iii. HTTP是一个属于应用层的面向对象的协议，由于其简捷、快速的方式，适用于分布式超媒体信息系统。它于1990年提出，经过几年的使用与发展，得到不断地完善和扩展。目前在WWW中使用的是HTTP/1.0的第六版，HTTP/1.1的规范化工作正在进行之中，而且HTTP-NG(Next Generation of HTTP)的建议已经提出。

iv. HTTP协议工作于客户端-服务端架构为上。浏览器作为HTTP客户端通过URL向HTTP服务端即WEB服务器发送所有请求。Web服务器根据接收到的请求后，向客户端发送响应信息。

b. 主要特点：

1、简单快速：客户向服务器请求服务时，只需传送请求方法和路径。请求方法常用的有GET、HEAD、POST。每种方法规定了客户与服务器联系的类型不同。由于HTTP协议简单，使得HTTP服务器的程序规模小，因而通信速度很快。

2、灵活：HTTP允许传输任意类型的数据对象。正在传输的类型由Content-Type加以标记。

3.无连接：无连接的含义是限制每次连接只处理一个请求。服务器处理完客户的请求，并收到客户的应答后，即断开连接。采用这种方式可以节省传输时间。

4.无状态：HTTP协议是无状态协议。无状态是指协议对于事务处理没有记忆能力。缺少状态意味着如果后续处理需要前面的信息，则它必须重传，这样可能导致每次连接传送的数据量增大。另一方面，在服务器不需要先前信息时它的应答就较快。

5、支持B/S及C/S模式。

11. get与post区别

1. 表单的method如果为get,那么所有的参数信息都会显示在浏览器的地址栏，当我们使用浏览器地址栏输入网址的方式来发送请求时,那么该请求一定是get方式

2. 对于get方式,底层是将所有参数附加在请求资源的后面一起传递的,对于post方式,底层是将所有参数附加在请求参数的最后一行的下一行的下一行

Get请求的数据是被附在url之后（HTTP协议头中），POST请求数据则放置在HTTP包的包体head中；

3. 对于get,post方式,servlet不同处理：doGet()，doPost();

4、浏览器处理：重复访问使用GET方法请求的页面，浏览器会使用缓存处理后续请求。使用POST方法的form提交时，浏览器基于POST将产生永久改变的假设，将让用户进行提交确认。

12. Nginx性能调优：

1、增加同时打开文件数open files数量；

2、处理大量静态文件的磁盘I/O时(此时worker进程是单线程的)，增加CPU核心数Worker Processes数量，提高计算能力；

3、如果网站流量很高，则可以提升worker进程连接数Worker Connections(默认1024)；

4、控制keep alive在10s-20s之间，减少连接的时间开销；

5、合理设置open file cache时间提高处理效率；

6、提高网路带宽；

7、开启压缩传输数据gzip，设置压缩级别gzip_comp_level为1-2，再高效果不明显并且浪费CPU了；

8、扩展机器数量。

13. rpc和http的区别，使用场景：

a. 区别：

- 传输协议
  - RPC，可以基于TCP协议，也可以基于HTTP协议
  - HTTP，基于HTTP协议
- 传输效率
  - RPC，使用自定义的TCP协议，可以让请求报文体积更小，或者使用HTTP2协议，也可以很好的减少报文的体积，提高传输效率
  - HTTP，如果是基于HTTP1.1的协议，请求中会包含很多无用的内容，如果是基于HTTP2.0，那么简单的封装以下是可以作为一个RPC来使用的，这时标准RPC框架更多的是服务治理
- 性能消耗，主要在于序列化和反序列化的耗时
  - RPC，可以基于thrift实现高效的二进制传输
  - HTTP，大部分是通过json来实现的，字节大小和序列化耗时都比thrift要更消耗性能
- 负载均衡
  - RPC，基本都自带了负载均衡策略
  - HTTP，需要配置Nginx，HAProxy来实现
- 服务治理（下游服务新增，重启，下线时如何不影响上游调用

者）

- RPC，能做到自动通知，不影响上游
- HTTP，需要事先通知，修改Nginx/HAProxy配置

a. 总结：RPC主要用于公司内部的服务调用，性能消耗低，传输效率高，服务治理方便。HTTP主要用于对外的异构环境，浏览器接口调用，APP接口调用，第三方接口调用等。

14.