计算机网络

基础

Q：五层协议的体系结构分别是什么？每一层都有哪些协议？

技术点：网络模型、协议
思路：分条解释每层名字以及协议
参考回答：

物理层
数据链路层：逻辑链路控制LLC、媒体接入控制MAC
网络层：IP协议、地址解析协议ARP、逆地址解析协议RARP、因特网控制报文协议ICMP
传输层：传输控制协议TCP、用户数据报协议UDP
应用层：文件传输协议FTP、远程登录协议TELNET、超文本传输协议HTTP、域名系统DNS、简单邮件协议SMTP、简单网络管理协议SNMP

Q：为何有MAC地址还要IP地址?

技术点：MAC地址、IP地址
参考回答：

每台主机在出厂时都有一个唯一的MAC地址，但是IP地址的分配是根据网络的拓朴结构，得以保证路由选择方案建立在网络所处的拓扑位置基础而不是设备制造商的基础上
使用IP地址更方便数据传输。数据包在这些节点之间的移动都是由ARP协议负责将IP地址映射到MAC地址上来完成的。

TCP

Q：TCP和UDP的区别?

技术点：传输层协议对比
参考回答：

TCP传输控制协议：面向连接；使用全双工的可靠信道；提供可靠的服务，即无差错、不丢失、不重复且按序到达；拥塞控制、流量控制、超时重发、丢弃重复数据等等可靠性检测手段；面向字节流；每条TCP连接只能是点到点的；用于传输可靠性要求高的数据
UDP用户数据报协议：无连接；使用不可靠信道；尽最大努力交付，即不保证可靠交付；无拥塞控制等；面向报文；支持一对一、一对多、多对一和多对多的交互通信；用于传输可靠性要求不高的数据

Q：拥塞控制和流量控制都是什么，两者的区别?

技术点：拥塞控制、流量控制
参考回答：

拥塞控制：对网络中的路由和链路传输进行速度限制，避免网络过载；包含四个过程：慢启动、拥塞避免、快重传和快恢复
流量控制 ：对点和点/发送方和接收方之间进行速度匹配，由于接收方的应用程序读取速度不一定很迅速，加上缓存有限，因此需要避免发送速度过快；相关技术：TCP滑动窗口、回退N针协议

Q：谈谈TCP为什么要三次握手？为什么要四次挥手?

技术点：TCP可靠保证
参考回答：
（1）建立TCP连接：TCP的三次握手

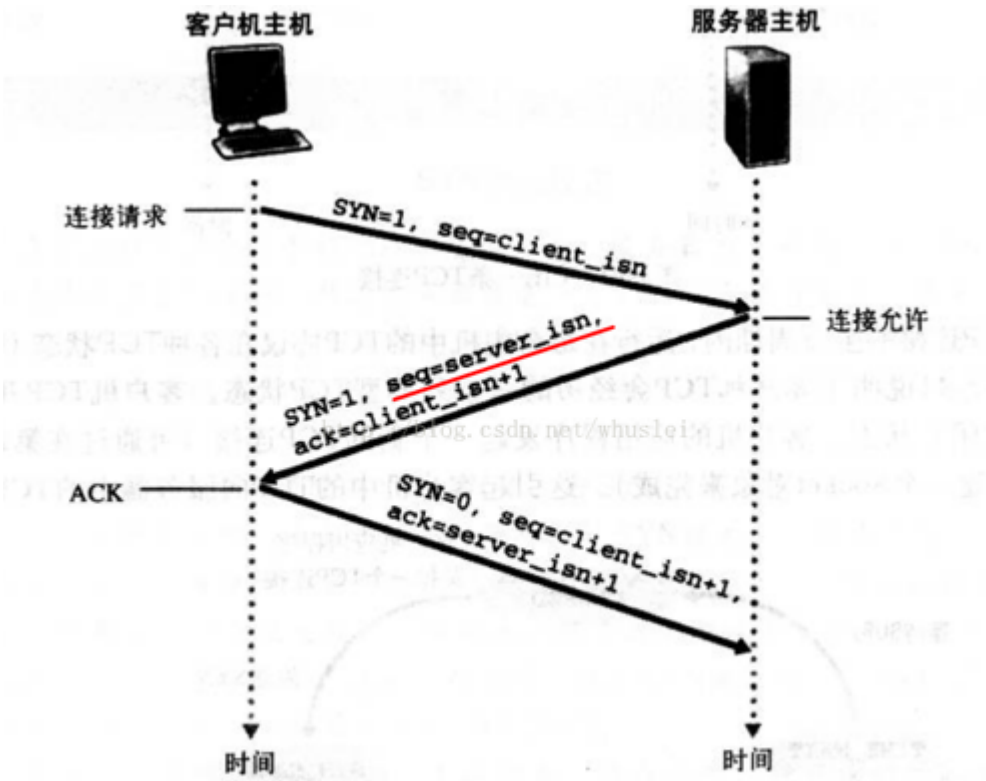

图3-39 TCP三次握手：报文段交换

客户端向服务端发送一个表示建立连接的报文段SYN报文段；一旦包含SYN报文段的IP数据报到达服务器主机，服务器从IP数据报中提取出TCP、SYN报文段，为该TCP连接分配需要的缓存和变量，并向客户端发送表示允许连接的报文段ACK；在收到ACK报文段之后，客户端也要给该连接分配缓存和变量，客户端向服务器再发送一个报文段ACK，表示对允许连接的报文段进行了确认。自此完成一次TCP连接。
第三次握手可以避免由于客户端延迟的请求连接的请求，使得服务端无故再次建立连接。

（2）断开TCP连接：TCP的四次挥手

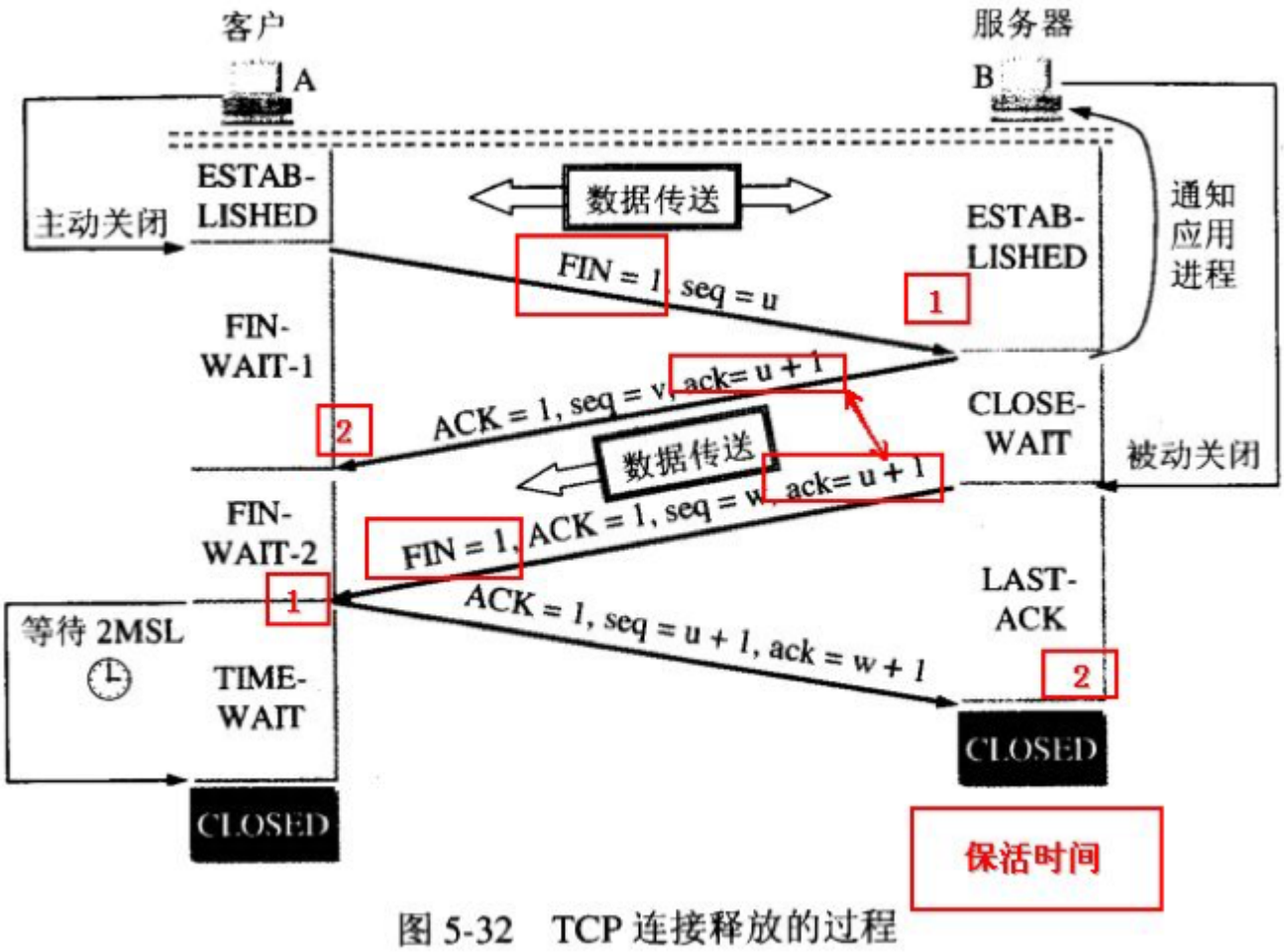

图 5-32　TCP 连接释放的过程

由于TCP连接是全双工的，因此每个方向都必须单独关闭。客户端在数据发送完毕后发送一个结束数据段FIN，且服务端也返回确认数据段ACK，此时结束了客户端到服务端的连接；然后客户端接收到服务端发送的FIN，且服务端也收到了ACK之后，自此双方的数据通信完全结束。简单说来是“先关读，后关写”，一共需要四个阶段：服务器读通道关闭->客户机写通道关闭->客户机读通道关闭->服务器写通道关闭。

引申：谈谈客户端到达的TIME_WAIT状态时间是MaximumSegmentLifetime的两倍，而不是直接进入CLOSED状态的原因。（保证TCP协议的全双工连接能够可靠关闭、保证本次连接的重复数据段从网络中消失）

Q：播放视频用TCP还是UDP？为什么？

技术点：传输层协议适用场景
参考回答：播放视频适合用UDP。UDP适用于对网络通讯质量要求不高、要求网络通讯速度能尽量快的实时性应用；而TCP适用于对网络通讯质量有要求的可靠性应用。而且视频区分关键帧和普通帧，虽然UDP会丢帧但如果只是丢普通帧损失并不大，取而代之的是高速率和实时性。
引申：TCP、UDP适用的场景

HTTP

Q：了解哪些响应状态码？

技术点：响应状态码

思路：
参考回答：状态码由三位数字组成，第一位数字表示响应的类型，常用的状态码有五大类：

1xx：表示服务器已接收了客户端请求，客户端可继续发送请求
2xx：表示服务器已成功接收到请求并进行处理

200 OK：表示客户端请求成功

3xx：表示服务器要求客户端重定向
4xx：表示客户端的请求有非法内容

400 Bad Request：表示客户端请求有语法错误，不能被服务器所理解
401 Unauthonzed：表示请求未经授权，该状态代码必须与 WWW-Authenticate 报头域一起使用
403 Forbidden：表示服务器收到请求，但是拒绝提供服务，通常会在响应正文中给出不提供服务的原因
404 Not Found：请求的资源不存在，例如，输入了错误的URL

5xx：表示服务器未能正常处理客户端的请求而出现意外错误

500 Internal Server Error：表示服务器发生不可预期的错误，导致无法完成客户端的请求
503 Service Unavailable：表示服务器当前不能够处理客户端的请求，在一段时间之后，服务器可能会恢复正常

Q：get和post的区别?

技术点：HTTP请求方法
参考回答：

GET：当客户端要从服务器中读取某个资源时使用GET；一般用于获取/查询资源信息；GET参数通过URL传递，传递的参数是有长度限制，不能用来传递敏感信息
POST：当客户端给服务器提供信息较多时可以使用POST；POST会附带用户数据，一般用于更新资源信息；POST将请求参数封装在HTTP 请求数据中，可以传输大量数据，传参方式比GET更安全

Q：HTTP1.0、HTTP1.1、HTTP2.0的区别?

技术点：HTTP协议发展
参考回答：
（1）HTTP1.0和HTTP1.1的区别：

HTTP1.0默认使用短连接，HTTP1.1开始默认使用长连接
HTTP1.1增加更多的请求头和响应头来改进和扩充HTTP1.0的功能，比如身份认证、状态管理和Cache缓存等

（2）HTTP2.0和HTTP1.X相比的新特性：

新的二进制格式：HTTP2.0的协议解析决定采用二进制格式，实现方便且健壮，不同于HTTP1.x的解析是基于文本
多路复用：连接共享，即每一个request都是是用作连接共享机制的
服务端推送：服务器主动向客户端推送消息

引申：长连接和短连接的优缺点和适用场景，HTTP 长连接和短连接

Q：HTTP和TCP的区别

技术点：HTTP、TCP
参考回答：
TCP是传输层协议，定义数据传输和连接方式的规范。通过三次握手建立连接、四次挥手释放连接。
HTTP是应用层协议，定义的是传输数据的内容的规范。HTTP的连接使用"请求-响应"方式。基于TCP协议传输，默认端口号是80

Q：HTTP和HTTPS的区别

技术点：HTTP、HTTPS
HTTP（超文本传输协议）：运行在TCP之上；传输的内容是明文；端口是80
HTTPS（安全为目标的HTTP）：运行在SSL/TLS之上，SSL/TLS运行在TCP之上；传输的内容经过加密；端口是443

Q：HTTP和Socket的区别

技术点：HTTP、Socket
参考回答：

HTTP是应用层协议；基于TCP协议；使用“请求–响应”方式建立连接，在请求时需要先建立连接且客户端要先发出请求，可见服务器需要等到客户端发送一次请求后才能将数据传回给客户端
Socket（套接字）是对TCP/IP协议的封装，是接口而不是协议；创建Socket连接时可以指定传输层协议TCP或UDP；
Socket建立连接过程三步骤：服务器监听->客户端请求->连接确认，可见服务器可以直接将数据传送给客户端（HTTP2.0也增加了服务端推送的功能）。

Q：在地址栏打入URL会发生什么?

技术点：理解网络模型
参考回答：在浏览器地址栏键入URL，按下回车之后会经历以下流程：

浏览器向DNS服务器请求解析该URL中的域名所对应的IP地址
解析出IP地址后，根据该IP地址和默认端口80，和服务器建立TCP连接
浏览器发出读取文件的HTTP请求，该请求报文作为TCP三次握手的第三个报文的数据发送给服务器
服务器对浏览器请求作出响应，并把对应的html文本发送给浏览器
释放TCP连接，若connection模式为close，则服务器主动关闭TCP连接，客户端被动关闭连接，释放TCP连接；若connection模式为keepalive，则该连接会保持一段时间，在该时间内可以继续接收请求
客户端将服务器响应的html文本解析并显示

操作系统

Q：操作系统中进程和线程的区别?

技术点：进程、线程
参考回答：
进程是操作系统分配和管理资源的单位，线程是CPU调度和管理的单位，是CPU调度的最小单元
进程拥有独立的地址空间，而线程间共享地址空间
进程创建的开销比较大，线程创建的开销小
引申：可谈谈安卓系统中对进程和线程的理解

Q：进程死锁的产生和避免?

技术点：死锁
思路：可从死锁含义、产生条件、解决办法、避免手段出发
参考回答：死锁是指多个进程因循环等待资源而造成无法执行的现象，它会造成进程无法执行，同时会造成系统资源的极大浪费。

死锁产生的条件：

互斥使用：指进程对所分配到的资源进行排它性使用，即在一段时间内某资源只由一个进程占用。如果此时还有其它进程请求资源，则请求者只能等待，直至占有资源的进程用毕释放。
不可抢占：指进程已获得的资源，在未使用完之前，不能被剥夺，只能在使用完时由自己释放。
请求和保持：指进程已经保持至少一个资源，但又提出了新的资源请求，而该资源已被其它进程占有，此时请求进程阻塞，但又对自己已获得的其它资源保持不放。
循环等待：指在发生死锁时，必然存在一个进程--资源的环形链，即进程集合{P0，P1，P2，···，Pn}中的P0正在等待一个P1占用的资源；P1正在等待P2占用的资源，……，Pn正在等待已被P0占用的资源。

解决死锁的策略：

银行家算法：判断此次请求是否造成死锁若会造成死锁，否则拒绝该请求
鸵鸟算法：忽略该问题，常用于在极少发生死锁的的情况

死锁的避免：通过合理的资源分配算法来确保永远不会形成环形等待的封闭进程链，即“如果一个进程的当前请求的资源会导致死锁，系统拒绝启动该进程；如果一个资源的分配会导致下一步的死锁，系统就拒绝本次的分配”

设计模式

Q：谈谈MVC、MVP和MVVM，好在哪里，不好在哪里?

技术点：MVC、MVP、MVVM
思路：详见MVP、MVVM模式

参考回答：

MVP的含义：

Model：数据层，负责存储、检索、操纵数据。

View：UI层，显示数据，并向Presenter报告用户行为。

Presenter：作为View与Model交互的中间纽带，从Model拿数据，应用到UI层，管理UI的状态，响应用户的行为。

MVP相比于MVC的优势：

分离了视图逻辑和业务逻辑，降低了耦合。
Activity只处理生命周期的任务，代码变得更加简洁。
视图逻辑和业务逻辑分别抽象到了View和Presenter的接口中去，提高代码的可阅读性。
Presenter被抽象成接口，可以有多种具体的实现，所以方便进行单元测试。
把业务逻辑抽到Presenter中去，避免后台线程引用着Activity导致Activity的资源无法被系统回收从而引起内存泄露和OOM。

MVVM的含义：与MVP类似，利用数据绑定(Data Binding)、依赖属性(Dependency Property)、命令(Command)、路由事件(Routed Event)等新特性，打造了一个更加灵活高效的架构。
MVVM相比于MVP的优势：在常规的开发模式中，数据变化需要更新UI的时候，需要先获取UI控件的引用，然后再更新UI，获取用户的输入和操作也需要通过UI控件的引用，但在MVVM中，这些都是通过数据驱动来自动完成的，数据变化后会自动更新UI，UI的改变也能自动反馈到数据层，数据成为主导因素。这样MVVM层在业务逻辑处理中只要关心数据，不需要直接和UI打交道，在业务处理过程中简单方便很多。

Q：如何理解生产者消费者模型?

技术点：生产者消费者模型
参考回答：生产者消费者模型通过一个缓存队列，既解决了生产者和消费者之间强耦合的问题，又平衡了生产者和消费者的处理能力。

具体规则：生产者只在缓存区未满时进行生产，缓存区满时生产者进程被阻塞；消费者只在缓存区非空时进行消费，缓存区为空时消费者进程被阻塞；当消费者发现缓存区为空时会通知生产者生产；当生产者发现缓存区满时会通知消费者消费。
实现关键：synchronized保证对象只能被一个线程占用；wait()让当前线程进入等待状态，并释放它所持有的锁；notify()&notifyAll()唤醒一个（所有）正处于等待状态的线程

Q：是否能从Android中举几个例子说说用到了什么设计模式?

技术点：设计模式
参考回答：
View事件分发：责任链模式
BitmapFactory加载图片：工厂模式
Adapter：适配器模式
Builder：建造者模式
Adapter.notifyDataSetChanged()：观察者模式
Binder机制：代理模式

Q：装饰模式和代理模式有哪些区别?

技术点：装饰模式、代理模式
参考回答：

使用目的不同：代理模式是给目标对象提供一个代理对象，并由代理对象控制对目标对象的引用；装饰模式是在不必改变原类文件和使用继承的情况下，动态地扩展一个对象的功能
构造不同：代理模式内部保持对目标对象的引用；装饰模式是通过构造函数传参的方式得到目标对象

Q：实现单例模式有几种方法？懒汉式中双层锁的目的是什么？两次判空的目的又是什么?

技术点：单例模式
参考回答：实现单例模式常见的两种方式：
（1）懒汉式：延迟加载，同时也要保证多线程环境下会产生多个single对象（DCL）

```
        public class Singleton {

            private Singleton() {}
            private volatile static Singleton instance;//第一层锁：保证变量可见性

            public static Singleton getInstance() {
                if (single == null) {//第一次判空：无需每次都加锁，提高性能
                    synchronized (Singleton.class) {//第二层锁：保证线程同步
                        if (single == null) {//第二次判空:避免多线程同时执行getInstance()产生多
个single对象
                            single = new Singleton();
                        }
                    }
                }
                return single;
            }
        }
```

（2）饿汉式：在类加载初始化时就创建好一个静态的对象供外部使用：

```
        public class Singleton {

            private Singleton() {}
            private static Singleton single = new Singleton();

            public static Singleton getInstance() {
                return single;
            }
        }
```

Q：谈谈了解的设计模式原则？

技术点：设计模式原则
参考回答：

单一职责原则：一个类只负责一个功能领域中的相应职责
开放封闭原则：对扩展开放，对修改关闭
依赖倒置原则：抽象不应该依赖于细节，细节应当依赖于抽象。换言之，要针对接口编程，而不是针对实现编程
迪米特法则：应该尽量减少对象之间的交互，如果两个对象之间不必彼此直接通信，那么这两个对象就不应当发生任何直接的相互作用，如果其中的一个对象需要调用另一个对象的某一个方法的话，可以通过第三者转发这个调用
合成/聚合复用原则：要尽量使用合成/聚合，尽量不要使用继承

数据库

Q：数据库中的事务了解吗？事务的四大特性？

技术点：事务
参考回答：

事务是并发控制的单位，是用户定义的一个操作序列。它指这些操作要么都做，要么都不做，以便服务器保持数据的完整性。
事务通常是以BEGIN TRANSACTION开始，以COMMIT或ROLLBACK结束。
事务的四大特性（ACID特性）：原子性（Atomicity）表示事务中包括的诸操作要么全做，要么全不做；一致性（Consistency）表示事务执行的结果必须是使数据库从一个一致性状态变到另一个一致性状态；隔离性（Isolation）表示一个事务的执行不能被其他事务干扰；持续性（Durability）表示一个事务一旦提交，它对数据库中数据的改变就应该是永久性的

Q：如何理解数据库的范式?

技术点：范式
思路：详见实例讲解数据库的3大范式和5大约束

参考回答：

第一范式（1NF）：数据表中的每个字段必须是不可拆分的最小单元，即确保每一列的原子性
第二范式（2NF）：满足1NF后，要求表中的所有列，都必须依赖于主键，而不能有任何一列与主键没有关系
第三范式（3NF）：必须先满足第二范式，要求表中的每一列只与主键直接相关而不是间接相关，即表中的每一列只能依赖于主键