前言

整理到网络！！！

参考链接

参考作者链接地址
https://www.nowcoder.com/discuss/125248
https://blog.csdn.net/shanghairuoxiao/article/details/72876248

网络技术(TCP/IP)

- OSI七层网络模型
- TCP三次握手过程，为什么需要三次？
    a. 首先Client向Server发送请求报文段，同时同步自己的SYN(x)，Client进入SYN_SENT状态。
    b. Server接收到Client的请求报文段，返回CLient自己的seq(y)及ack(x+1)，Server进入SYN_REVD状态。
    c. CLinet收到Server的SYN+ACK包，向服务器发送一个序列号seq(x+1)，确认号为ack(y+1)，此包发送完毕，Client和Server进入ESTABLISHED(TCP连接成功)状态，完成三次握手。
    > 需要三次的原因：防止已失效的报文段出现在本连接中。
- TCP四次挥手的过程
    a. 客户端发送断开TCP连接请求的报文，其中报文中包含seq序列号，是由发送端随机生成的，并且还将报文中的FIN字段置为1，表示需要断开TCP连

接。（FIN=1，seq=x，x由客户端随机生成）

b. 服务端会回复客户端发送的TCP断开请求报文，其包含seq序列号，是由回复端随机生成的，而且会产生ACK字段，ACK字段数值是在客户端发过来的seq序列号基础上加1进行回复，以便客户端收到信息时，知晓自己的TCP断开请求已经得到验证。（FIN=1，ACK=x+1，seq=y，y由服务端随机生成）

c. 服务端在回复完客户端的TCP断开请求后，不会马上进行TCP连接的断开，服务端会先确保断开前，所有传输到A的数据是否已经传输完毕，一旦确认传输数据完毕，就会将回复报文的FIN字段置1，并且产生随机seq序列号。（FIN=1，ACK=x+1，seq=z，z由服务端随机生成）

d. 客户端收到服务端的TCP断开请求后，会回复服务端的断开请求，包含随机生成的seq字段和ACK字段，ACK字段会在服务端的TCP断开请求的seq基础上加1，从而完成服务端请求的验证回复。（FIN=1，ACK=z+1，seq=h，h为客户端随机生成）
至此TCP断开的4次挥手过程完毕

- 为什么TCP建立连接需要三次握手，而断开连接需要四次挥手？
因为当Server端收到Client端的SYN连接请求报文后，可以直接发送SYN+ACK报文。其中ACK报文是用来应答的，SYN报文是用来同步的。但是关闭连接时，当Server端收到FIN报文时，很可能并不会立即关闭SOCKET，所以只能先回复一个ACK报文，告诉Client端，”你发的FIN报文我收到了”。只有等到我Server端所有的报文都发送完了，我才能发送FIN报文，因此不能一起发送。故需要四步握手。
- TIME_WAIT的意义，为什么等于2MSL？
MSL是最长报文段寿命，设置的目的是：
- 保证A发送的最后一个ACK能够到达B
- 防止已失效的报文段出现在本连接中
- TCP头部校验的原理，安全吗，可以仿造吗？
TCP校验和是一个端到端的校验和，由发送端计算，然后由接收端验证。其目的是为了发现TCP首部和数据在发送端到接收端之间发生的任何改动。如果接收方检测到校验和有差错，则TCP段会被直接丢弃。
可以仿造。
- TCP、UDP的区别？服务器和客户端建立的过程？
TCP---传输控制协议,提供的是面向连接、可靠的字节流服务。当客户和服务器彼此交换数据前，必须先在双方之间建立一个TCP连接，之后才能传输数据。TCP提供超时重发，丢弃重复数据，检验数据，流量控制等功能，保证数据能顺序地

从一端传到另一端。

UDP---用户数据报协议，是一个简单的面向数据报的运输层协议。UDP不提供可靠性，它只是把应用程序传给IP层的数据报发送出去，但是并不能保证它们能到达目的地。由于UDP在传输数据报前不用在客户和服务器之间建立一个连接，且没有超时重发等机制，不保证数据按顺序传递，故而传输速度很快。

- UDP编程的服务器端一般步骤

  a. 创建一个socket，用函数socket()；

  b. 设置socket属性，用函数setsockopt();* 可选

  c. 绑定IP地址、端口等信息到socket上，用函数bind();

  d. 循环接收数据，用函数recvfrom();

  e. 关闭网络连接；

- UDP编程的客户端一般步骤是

  a. 创建一个socket，用函数socket()；

  b. 设置socket属性，用函数setsockopt();* 可选

  c. 绑定IP地址、端口等信息到socket上，用函数bind();* 可选

  d. 设置对方的IP地址和端口等属性;

  e. 发送数据，用函数sendto();

  f. 关闭网络连接；

- TCP编程的服务器端一般步骤是

  a. 创建一个socket，用函数socket()；

  b. 设置socket属性，用函数setsockopt(); * 可选

  c. 绑定IP地址、端口等信息到socket上，用函数bind();

  d. 开启监听，用函数listen()；

  e. 接收客户端上来的连接，用函数accept()；

  f. 收发数据，用函数send()和recv()，或者read()和write();

  g. 关闭网络连接；

  h. 关闭监听；

- TCP编程的客户端一般步骤是

  a. 创建一个socket，用函数socket()；

  b. 设置socket属性，用函数setsockopt();* 可选

c. 绑定IP地址、端口等信息到socket上，用函数bind();* 可选

d. 设置要连接的对方的IP地址和端口等属性；

e. 连接服务器，用函数connect()；

f. 收发数据，用函数send()和recv()，或者read()和write();

g. 关闭网络连接；

- socket中的close是一次就关闭的吗？半关闭状态是怎么产生的？
  不是，当A发送给B控制FIN的时候，A到B这个方向的连接就关闭了，这个时候处于半关闭的状态，但是B到A这个方向的连接并没有关闭，因为B要等到将数据全部发送完毕之后才会发送FIN给A。
- TCP拥塞控制
  重点掌握慢开始、拥塞避免、快重传、快恢复。
- TCP流量控制,采用滑动窗口会用什么问题？
  流量控制是为了让发送方的发送速率不要太快，要让接收方来得及接收。
  Nagle算法：①当发送方首都哦啊哦对第一个数据字符的确认后，再把发送缓存中的所有数据组装成一个报文段发送出去，同时继续对随后到到达的数据进行缓存。②当到达的数据已达到发送窗口大小的一半或已达到报文段的长度的时候就立即发送一个报文段。
  糊涂窗口综合征：就是由于发送端和接收端上的处理不一致，导致网络上产生很多的小包，结果报文段包含了一个大大的头部，携带数据很少。数据传输效率低。处理方法是等待窗口大小满足一定的条件之后(能够接收一个最大报文，或者缓冲区的一半)，再来发送窗口通告，这样就不会产生小报文。

  > 滑动窗口机制为端到端设备间的数据传输提供了可靠的流量控制机制。然而，它只能在源端设备和目的端设备起作用，当网络中间设备（例如路由器等）发生拥塞时，滑动窗口机制将不起作用。

- 拥塞控制和流量控制的区别？

  a. 拥塞控制就是防止过多的数据注入到网络中，这样可以使网络中的路由器或链路不会过载

  b. 流量控制往往是点对点通信量的控制，是一个端到端的问题，流量控制要做的是抑制发送端发送数据的速率，以便接收端来得及接收。

- TCP怎么保证可靠性？

  a. 应用数据被分割成TCP认为最适合发送的数据块

b. 超时重传：当TCP发出一个段后，它启动一个定时器，等待目的端确认收到这个报文段。如果不能及时收到一个确认，将重发这个报文段

c. TCP给发送的每一个包进行编号，接收方对数据包进行排序，把有序数据传送给应用层

d. 校验和：TCP将保持它首部和数据的检验和。这是一个端到端的检验和，目的是检测数据在传输过程中的任何变化。如果收到段的检验和有差错，TCP将丢弃这个报文段和不确认收到此报文段

e. TCP的接收端会丢弃重复的数据

f. 流量控制：让发送方的发送速率不要太快，要让接收方来得及接收

g. 拥塞控制：当网络拥塞时，减少数据的发送

- TCP滑动窗口协议

  a. “窗口”对应的是一段可以被发送者发送的字节序列，其连续的范围称之为“窗口”

  b. “滑动”则是指这段“允许发送的范围”是可以随着发送的过程而变化的，方式就是按顺序“滑动”

- http协议与TCP协议的联系
  TPC协议是传输层协议，主要解决数据如何在网络中传输，而HTTP是应用层协议，主要解决如何包装数据。
  Http协议是建立在TCP协议基础之上的，当浏览器需要从服务器获取网页数据的时候，会发出一次Http请求。Http会通过TCP建立起一个到服务器的连接通道，当本次请求需要的数据完毕后，Http会立即将TCP连接断开，这个过程是很短的。所以Http连接是一种短连接，是一种无状态的连接。

- http/1.0和http/1.1的区别？

  a. http1.1提供永久性连接，即1.0使用非持久连接

  b. http1.1增加host头

  c. http1.1还提供了身份认证，状态管理和cache缓存机制等相关的请求头和响应头。

- http请求方法有哪些？get和post的区别

  a. OPTIONS
  返回服务器针对特定资源所支持的HTTP请求方法，也可以利用向web服务器发送‘*’的请求来测试服务器的功能性

b. HEAD

向服务器索与GET请求相一致的响应，只不过响应体将不会被返回。这一方法可以再不必传输整个响应内容的情况下，就可以获取包含在响应小消息头中的元信息。

c. GET

向特定的资源发出请求。注意：GET方法不应当被用于产生“副作用”的操作中，例如在Web Application中，其中一个原因是GET可能会被网络蜘蛛等随意访问。Loadrunner中对应get请求函数：web_link和web_url

d. POST

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）。数据被包含在请求体中。POST请求可能会导致新的资源的建立和/或已有资源的修改。 Loadrunner中对应POST请求函数：
web_submit_data,web_submit_form

e. PUT

向指定资源位置上传其最新内容

f. DELETE

请求服务器删除Request-URL所标识的资源

g. TRACE

回显服务器收到的请求，主要用于测试或诊断

h. CONNECT

HTTP/1.1协议中预留给能够将连接改为管道方式的代理服务器。

> (1)根据HTTP规范，GET用于信息获取，而且应该是安全的和幂等的
> (2)根据HTTP规范，POST表示可能修改变服务器上的资源的请求

- get/post 区别

a. get重点在从服务器上获取资源，post重点在向服务器发送数据；

b. get传输数据是通过URL请求，以field（字段）= value的形式，置于URL后，并用"?"连接，多个请求数据间用"&"连接，如 http://127.0.0.1/Test/login.action?name=admin&password=admin，这个过程用户是可见的；
post传输数据通过Http的post机制，将字段与对应值封存在请求实体中发送给服务器，这个过程对用户是不可见的；

c. Get传输的数据量小，因为受URL长度限制，但效率较高；Post可以传输大量数据，所以上传文件时只能用Post方式；

d. get是不安全的，因为URL是可见的，可能会泄露私密信息，如密码等；post较get安全性较高；

- http状态码
  状态码有很多，可以看下数字代表什么意思。
- http和https的区别？由http升级到https需要哪些操作？
  HTTP 指的是超文本传输协议，https 指的是超文本传输安全协议。HTTPS 就是将 HTTP 中的传输内容进行了加密，然后通过可靠的连接，传输到对方的机器上。加密的协议是 TLS,其前身是 SSL。
- https具体怎么实现？，怎么确保安全性？
- http中浏览器一个URL的流程，这个过程中浏览器做些什么，URL包括哪三个部分？

  a. 浏览器向DNS服务器查找输入URL对应的IP地址。

  b. DNS服务器返回网站的IP地址。

  c. 浏览器根据IP地址与目标web服务器在80端口上建立TCP连接

  d. 浏览器获取请求页面的html代码。

  e. 浏览器在显示窗口内渲染HTML。

  f. 窗口关闭时，浏览器终止与服务器的连接。
  URL包括：①协议（或称为服务方式）；②存有该资源的主机IP地址（有时也包括端口号）；③主机资源的具体地址，如目录和文件名等。

  > 浏览器中输入URL，首先浏览器要将URL解析为IP地址，解析域名就要用到DNS协议，首先主机会查询DNS的缓存，如果没有就给本地DNS发送查询请求。DNS查询分为两种方式，一种是递归查询，一种是迭代查询。如果是迭代查询，本地的DNS服务器，向根域名服务器发送查询请求，根域名服务器告知该域名的一级域名服务器，然后本地服务器给该一级域名服务器发送查询请求，然后依次类推直到查询到该域名的IP地址。DNS服务器是基于UDP的，因此会用到UDP协议。
  > 得到IP地址后，浏览器就要与服务器建立一个http连接。因此要用到http协议，http协议报文格式上面已经提到。http生成一个get请求报文，将该报文传给TCP层处理。如果采用https还会先对http数据进行加密。
  > TCP层如果有需要先将HTTP数据包分片，分片依据路径MTU和MSS。
  > TCP的数据包然后会发送给IP层，用到IP协议。IP层通过路由选路，一跳一跳发送到目的地址。当然在一个网段内的寻址是通过以太网协议实现(也可以是其他物理层协议，比如PPP，SLIP)，以太网协议需要直到

> 目的IP地址的物理地址，有需要ARP协议。

- http四个会话过程？

  a. 建立tcp连接

  b. 发出请求文档

  c. 发出响应文档

  d. 释放tcp连接

  e. 网页解析的过程

- 一个机器能使用的端口号上限是多少？为什么？可以改变吗？如果想要用的端口超过这个限制怎么办？
- 对称加密和非对称加密
  对称加密(也叫私钥加密)指加密和解密使用相同密钥的加密算法。有时又叫传统密码算法，就是加密密钥能够从解密密钥中推算出来，同时解密密钥也可以从加密密钥中推算出来。
  非对称加密算法需要两个密钥:公开密钥(publickey)和私有密钥(privatekey)。公开密钥与私有密钥是一对，如果用公开密钥对数据进行加密，只有用对应的私有密钥才能解密;如果用私有密钥对数据进行加密，那么只有用对应的公开密钥才能解密。因为加密和解密使用的是两个不同的密钥，所以这种算法叫作非对称加密算法。
- 数字证书的了解（高频）
- 客户端为什么信任第三方证书？
- RSA加密算法，MD5原理
- 单条记录高并发访问的优化
- 介绍一下ping的过程，分别用到了那些协议？
  ping用来测试两台主机之间的连通性。ICMP协议
- TCP/IP分片粘包过程
  a. 正常情况：如果Socket Client 发送的数据包，在Socket Server端也是一个一个完整接收的，那个就不会出现粘包和分包情况，数据正常读取。
  b. 粘包情况：Socket Client发送的数据包，在客户端发送和服务器接收的情况下都有可能发送，因为客户端发送的数据都是发送的一个缓冲buffer，然后由缓冲buffer最后刷到数据链路层的，那么就有可能把数据包2的一部分数据结合数据包1的全部被一起发送出去了，这样在服务器端就有可能出现这样的情况，导致读取的数据包包含了数据包2的一部分数据，这就产生粘包，

当然也有可能把数据包1和数据包2全部读取出来。

c. 分包情况：意思就是把数据包2或者数据包1都有可能被分开一部分发送出去，接着发另外的部分，在服务器端有可能一次读取操作只读到一个完整数据包的一部分。

d. 在数据包发送的情况下，有可能后面的数据包分开成2个或者多个，但是最前面的部分包，黏住在前面的一个完整或者部分包的后面，也就是粘包和分包同时产生了。

- 有没有抓过TCP包，描述一下
- 一个IP配置多个域名，靠什么识别？
  主机头
- 路由器的工作原理和作用，交换机的工作原理和作用
- 对路由协议的了解与介绍。
- 路由协议使用的算法
- 服务器攻击（DDos攻击）

操作系统

- 什么是临界区？进程进入临界区的调度原则是？
  临界区是一段对共享资源的保护代码，该保护代码在任意时刻只允许一个线程对共享资源访问。
  进程进入临界区的调度原则是：
  ①如果有若干进程要求进入空闲的临界区，一次仅允许一个进程进入。②任何时候，处于临界区内的进程不可多于一个。如已有进程进入自己的临界区，则其它所有试图进入临界区的进程必须等待。③进入临界区的进程要在有限时间内退出，以便其它进程能及时进入自己的临界区。④如果进程不能进入自己的临界区，则应让出CPU，避免进程出现“忙等”现象。
- 互斥对象、临界区和事件的区别？
  互斥是一种用途非常广泛的内核对象。能够保证多个线程对同一共享资源的互斥访问。
  事件对象也是属于内核对象，它的主要成员包括：1.使用计数 2.指明该事件是一个自动重置事件还是一个人工重置事件的布尔值3.指明该事件处于已通知状态还是未通知状态的布尔值。
- 互斥对象、事件对象与临界区的比较:
  互斥对象、事件对象都是属于内核对象，利用内核对象进行线程同步，速度较

慢，但可以在多个进程中的多个线程间可以进行同步。
临界区属于在用户模式下，同步速度较快，但是很容易进入死锁状态，因为在等待进入临界区时无法设定超时值。

- 进程和线程的区别？进程和程序的区别？
  进程是资源（CPU、内存等）分配的基本单位，它是程序执行时的一个实例。程序运行时系统就会创建一个进程，并为它分配资源，然后把该进程放入进程就绪队列，进程调度器选中它的时候就会为它分配CPU时间，程序开始真正运行。
  线程是程序执行时的最小单位，它是进程的一个执行流，是CPU调度和分派的基本单位，一个进程可以由很多个线程组成，线程间共享进程的所有资源，每个线程有自己的堆栈和局部变量。线程由CPU独立调度执行，在多CPU环境下就允许多个线程同时运行。同样多线程也可以实现并发操作，每个请求分配一个线程来处理。

  - 进程和程序的区别
    进程是动态的，而程序是静态的。
    进程有一定的生命期，而程序是指令的集合，本身无“运动”的含义。没有建立进程的程序不能作为1个独立单位得到操作系统的认可。
    1个程序可以对应多个进程，但1个进程只能对应1个程序。进程和程序的关系犹如演出和剧本的关系。

- 一个进程可以创建多少个线程？和什么有关？
  一个进程可以创建的线程数由可用虚拟空间和线程的栈的大小共同决定
- 僵尸进程？
- 什么是死锁？死锁产生的原因？死锁四个必要条件？死锁的解除、死锁控制？
  死锁是指多个进程因竞争资源而造成的一种僵局（互相等待），若无外力作用，这些进程都将无法向前推进。
- 死锁产生的原因

  - 系统资源的竞争
    系统资源的竞争导致系统资源不足，以及资源分配不当，导致死锁。
  - 进程运行推进顺序不合适
    进程在运行过程中，请求和释放资源的顺序不当，会导致死锁。

- 死锁的四个条件

  a. 互斥条件：一个资源每次只能被一个进程使用，即在一段时间内某 资源仅为一个进程所占有。此时若有其他进程请求该资源，则请求进程只能等待。

b. 请求与保持条件：进程已经保持了至少一个资源，但又提出了新的资源请求，而该资源 已被其他进程占有，此时请求进程被阻塞，但对自己已获得的资源保持不放。

c. 不可剥夺条件:进程所获得的资源在未使用完毕之前，不能被其他进程强行夺走，即只能 由获得该资源的进程自己来释放（只能是主动释放)。

d. 循环等待条件: 若干进程间形成首尾相接循环等待资源的关系
这四个条件是死锁的必要条件，只要系统发生死锁，这些条件必然成立，而只要上述条件之一不满足，就不会发生死锁。

- 死锁的避免与预防

a. 死锁避免的基本思想
系统对进程发出每一个系统能够满足的资源申请进行动态检查,并根据检查结果决定是否分配资源,如果分配后系统可能发生死锁,则不予分配,否则予以分配。这是一种保证系统不进入死锁状态的动态策略。
理解了死锁的原因，尤其是产生死锁的四个必要条件，就可以最大可能地避免、预防和解除死锁。所以，在系统设计、进程调度等方面注意如何让这四个必要条件不成立，如何确定资源的合理分配算法，避免进程永久占据系统资源。此外，也要防止进程在处于等待状态的情况下占用资源。因此，对资源的分配要给予合理的规划。

b. 死锁避免和死锁预防的区别
死锁预防是设法至少破坏产生死锁的四个必要条件之一,严格的防止死锁的出现,而死锁避免则不那么严格的限制产生死锁的必要条件的存在,因为即使死锁的必要条件存在,也不一定发生死锁。死锁避免是在系统运行过程中注意避免死锁的最终发生。

- 死锁的解除
一旦检测出死锁，就应立即采取相应的措施，以解除死锁。死锁解除的主要两种方法：

a. 抢占资源。从一个或多个进程中抢占足够数量的资源，分配给死锁进程，以解除死锁状态。

b. 终止（或撤销）进程。终止（或撤销）系统中的一个或多个死锁进程，直至打破循环环路，使系统从死锁状态解脱出来。

- Windows内存管理方式

a. 分页存储

用户程序的逻辑地址空间被划分成若干固定大小的区域，称为“页”或者“页面”，相应地，内存物理空间也分成相对应的若干个物理块，页和块的大小相等。提高了内存的利用率。

b. 分段存储
将用户程序地址空间分成若干个大小不等的段，每段可以定义一组相对完整的逻辑信息。提高程序的逻辑性。

c. 段页式存储
两者结合。作业的地址空间首先被分成若干个逻辑分段，每段都有自己的段号，然后再将每段分成若干个大小相等的页。

- 一个程序从开始运行到结束的完整过程（四个过程）
预处理，编译，汇编，链接。

a. 源代码.c 文件先经过预处理器，生成一个中间文件.i 文件，这个阶段有两个作用，一是把include的头文件内容进行替换，二是处理宏定义。

b. .i 文件经过编译生成汇编.s 文件

c. .s 的汇编文件经过汇编器生成.obj 的目标文件

d. .obj 经过链接器和 lib（静态链接库） dll（动态链接库）文件生成 exe 可执行程序

- 头文件在编译过程中的作用？（网易游戏）
头文件并不参加链接和编译。编译器第一步要做的就是简单的把头文件在包含它的源文件中展开。不知你是否能理解这句话。也就是头文件里面有什么内容，通通把它移到包含这个头文件的源文件里。（我觉得这是个很重要的概念，可以帮助我们简化理解编译链接的过程，包括理解头文件中定义静态变量或静态函数是怎么回事）。编译器经过这一步转换后剩下什么呢？就是一堆cpp文件了。而头文件已经不再是编译器需要关心的东西了。编译器接下来就要处理这一堆cpp文件了。
所以第一个阶段是预处理阶段，在正式的编译阶段之前进行。预处理阶段将根据已放置在文件中的预处理指令来修改源文件的内容。如#include指令就是一个预处理指令，它把头文件的内容添加到.cpp文件中。 第二个阶段编译、优化阶段。

- 进程间通信的方法？

a. 管道pipe：管道是一种半双工的通信方式，数据只能单向流动，而且只能在具有亲缘关系的进程间使用。进程的亲缘关系通常是指父子进程关系。

b. 命名管道FIFO：有名管道也是半双工的通信方式，但是它允许无亲缘关系进程间的通信。

c. 消息队列MessageQueue：消息队列是由消息的链表，存放在内核中并由消息队列标识符标识。消息队列克服了信号传递信息少、管道只能承载无格式字节流以及缓冲区大小受限等缺点。

d. 共享存储SharedMemory：共享内存就是映射一段能被其他进程所访问的内存，这段共享内存由一个进程创建，但多个进程都可以访问。共享内存是最快的 IPC 方式，它是针对其他进程间通信方式运行效率低而专门设计的。它往往与其他通信机制，如信号两，配合使用，来实现进程间的同步和通信。

e. 信号量Semaphore：信号量是一个计数器，可以用来控制多个进程对共享资源的访问。它常作为一种锁机制，防止某进程正在访问共享资源时，其他进程也访问该资源。因此，主要作为进程间以及同一进程内不同线程之间的同步手段。

f. 套接字Socket：套解口也是一种进程间通信机制，与其他通信机制不同的是，它可用于不同及其间的进程通信。

g. 信号（sinal）： 信号是一种比较复杂的通信方式，用于通知接收进程某个事件已经发生。

- 线程同步方法？
- 锁机制
  a. 互斥锁：提供了以排它方式阻止数据结构被并发修改的方法。
  b. 读写锁：允许多个线程同时读共享数据，而对写操作互斥。
  c. 条件变量：可以以原子的方式阻塞进程，直到某个特定条件为真为止。对条件测试是在互斥锁的保护下进行的。条件变量始终与互斥锁一起使用。
- 信号量机制：包括无名线程信号量与有名线程信号量
- 信号机制：类似于进程间的信号处理。
  线程间通信的主要目的是用于线程同步，所以线程没有象进程通信中用于数据交换的通信机制。
- 线程创建的方式有几种？
- 进程调度算法？
  a. 先来先去服务
  b. 短作业(进程)优先调度算法SJ(P)F
  c. 轮转法
  d. 多级反馈队列算法
- 页面置换方法

a. 最优页面置换算法

b. 最近未使用页面置换算法（NRU）

c. 先进先出页面置换算法（FIFO）及其改进

d. 时钟页面置换算法（clock）

e. 最近最少使用页面置换算法（LRU）

f. 工作集算法

- 布隆过滤器的优点与缺点
- 布隆过滤器处理大规模问题时的持久化，包括内存大小首先、磁盘换入换出问题
- 文件读写使用的系统调用
- 线程池的了解、优点、调度处理方式和保护任务队列的方式
  于是为了避免一个程序需要大量创建线程时的不必要浪费，也就是最好的去避免线程创建与线程销毁的时间浪费，此时线程池就出现了。线程池的实现就是在初始的时候创建一些线程（业界通常认为创建CPU核心数的两倍为最佳，也有说是两倍+1），创建的线程为挂起状态（就绪），当我们有任务要处理的时候，我们就激活一个就绪的线程去完成任务，完成任务后，线程又变为就绪态进行继续等待任务的到来。这样过程使得每个线程一次创建，多次使用，如果你的程序并没有多次任务处理，使得线程池中的线程长时间处于就绪态，此时就建议你直接使用一个线程就好，不必使用线程池。
- 线程池怎么创建？
- 怎么回收线程
- 多线程同步（项目中可能会问）
- mencache
- 异常和中断的区别
- 如何保证线程安全？

Linux

- 进程与线程
  (1) 进程与线程区别？
  (2) 线程比进程具有哪些优势？
  (3) 什么时候用多进程？什么时候用多线程？
  (4) LINUX中进程和线程使用的几个函数？
  (5) 线程同步？

在Windows下线程同步的方式有：互斥量，信号量，事件，关键代码段
在Linux下线程同步的方式有：互斥锁，自旋锁，读写锁，屏障(并发完成同一项任务时，屏障的作用特别好使)
知道这些锁之间的区别，使用场景？

- 进程调度
  Linux进程分为两种，实时进程和非实时进程；
  优先级分为静态优先级和动态优先级，优先级的范围；
  调度策略，FIFO，LRU，时间片轮转
  交互进程通过平均睡眠时间而被奖励；
- fork与vfork区别
  fork和vfork都用于创建子进程。但是vfork创建子进程后，父进程阻塞，直到子进程调用exit()或者excle()。
  对于内核中过程fork通过调用clone函数，然后clone函数调用do_fork()。
  do_fork()中调用copy_process()函数先复制task_struct结构体，然后复制其他关于内存，文件，寄存器等信息。fork采用写时拷贝技术，因此子进程和父进程的页表指向相同的页框。但是vfork不需要拷贝页表，因为父进程会一直阻塞，直接使用父进程页表。
- exit()与_exit()区别
  exit()清理后进入内核，_exit()直接陷入内核。
- 孤儿进程与僵死进程
  孤儿进程是怎么产生的？
  僵死进程是怎么产生的？
  僵死进程的危害？
  如何避免僵死进程的产生？
- Linux的I/O模型介绍以及同步异步阻塞非阻塞的区别？
  a. 阻塞IO
  b. 非阻塞IO
  c. IO复用
  d. 信号驱动IO
  e. 异步IO
- Linux进程管理
- Linux内核的进程调度
- fork返回值是什么？

- 什么是虚拟内存？
- 文件系统的理解（EXT4，XFS，BTRFS）
    a. EXT4：使用广泛
    b. XFS：XFS 文件系统是扩展文件系统的一个扩展，XFS 文件系统有一些缺陷，例如它不能压缩，删除大量文件时性能低下。
    c. btrfs：有很多好用的功能，例如写复制、扩展校验、快照、清洗、自修复数据、冗余删除以及其它保证数据完整性的功能。
- Linux的内存管理？
- Linux基本命令？
    a. 与CPU，内存，磁盘相关的命令(top，free, df, fdisk)
    b. 网络相关的命令netstat，tcpdump等
    c. sed, awk, grep三个超强大的命名，分别用与格式化修改，统计，和正则查找
    d. ipcs和ipcrm命令
    e. 查找当前目录以及字母下以.c结尾的文件，且文件中包含”hello world”的文件的路径
    f. 创建定时任务

命令	作用
pwd	显示当前目录
rm	删除
touch	生成文件
cat	读取指定文件的内容并打印到终端输出
mkdir	新建目录make directory
file	查看文件类型
whereis，which，find 和 locate	查找
chown	改变文件所有者
df	查看磁盘容量
wc	计数工具
tr	删除一段文本信息中的某些文字。或者将其进行转换
join	连接两个文件
paste	它是在不对比数据的情况下，简单地将多个文件合并一起，以Tab隔开

- grep、awk、sed掌握程度？
  a. grep命令用于打印输出文本中匹配的模式串，它使用正则表达式作为模式匹配的条件。
  b. sed用于过滤和转换文本的流编辑器。
  c. AWK是一种用于处理文本的编程语言工具。
- Linux是如何避免内存碎片的
  a. 伙伴算法，用于管理物理内存，避免内存碎片;
  b. 高速缓存Slab层用于管理内核分配内存，避免碎片。
- 文件权限的查看与修改？

a. 文件权限查看ls -l，查看文件所有者，所属组，其他的文件权限，rwx为777

b. 修改使用chmod命令

- Linux如何打开一个文件？如何处理一个正在动态增长的文件？
- IO复用的三种方法,poll、epoll和select的特点和区别
  select，poll，epoll都是IO多路复用的机制。I/O多路复用就通过一种机制，可以监视多个描述符，一旦某个描述符就绪（一般是读就绪或者写就绪），能够通知程序进行相应的读写操作。但select，poll，epoll本质上都是同步I/O，因为他们都需要在读写事件就绪后自己负责进行读写，也就是说这个读写过程是阻塞的，而异步I/O则无需自己负责进行读写，异步I/O的实现会负责把数据从内核拷贝到用户空间。
- select：是最初解决IO阻塞问题的方法。用结构体fd_set来告诉内核监听多个文件描述符，该结构体被称为描述符集。由数组来维持哪些描述符被置位了。对结构体的操作封装在三个宏定义中。通过轮寻来查找是否有描述符要被处理，如果没有返回。
  存在的问题：

  a. 内置数组的形式使得select的最大文件数受限与FD_SIZE；

  b. 每次调用select前都要重新初始化描述符集，将fd从用户态拷贝到内核态，每次调用select后，都需要将fd从内核态拷贝到用户态；

  c. 轮寻排查当文件描述符个数很多时，效率很低；

- poll：通过一个可变长度的数组解决了select文件描述符受限的问题。数组中元素是结构体，该结构体保存描述符的信息，每增加一个文件描述符就向数组中加入一个结构体，结构体只需要拷贝一次到内核态。poll解决了select重复初始化的问题。轮寻排查的问题未解决。
- epoll：轮寻排查所有文件描述符的效率不高，使服务器并发能力受限。因此，epoll采用只返回状态发生变化的文件描述符，便解决了轮寻的瓶颈。
- 为什么使用IO多路复用，最主要的原因是什么？
- epoll有两种触发模式？这两种触发模式有什么区别？编程的时候有什么区别？
- 上一题中编程的时候有什么区别，是在边缘触发的时候要把套接字中的数据读干净，那么当有多个套接字时，在读的套接字一直不停的有数据到达，如何保证其他套接字不被饿死。
- select/poll/epoll区别
  https://segmentfault.com/a/1190000003063859

- 几种网络服务器模型的介绍与比较
  https://www.ibm.com/developerworks/cn/linux/l-cn-edntwk/index.html?ca=drs-
- epoll为什么这么快(搞懂这篇文章，关于IO复用的问题就信手拈来了)
  http://www.jianshu.com/p/b5bc204da984
- GDB调试
- Linux进程和线程如何创建、退出？进程退出的时候，自己没有释放的资源（如内存没有free）会怎样？

数据库

- 存储过程
  我们常用的关系型数据库是MySQL，操作数据库的语言一般为SQL语句，SQL在执行的时候需要要先编译，然后执行，而存储过程（Stored Procedure）是一组为了完成某种特定功能的SQL语句集，经编译后存储在数据库中，用户通过指定存储过程的名字并给定参数（如果该存储过程带有参数）来调用执行它。
  一个存储过程是一个可编程的函数，它在数据库中创建并保存。它可以有SQL语句和一些特殊的控制结构组成。当希望在不同的应用程序或平台上执行相同的函数，或者封装特定功能时，存储过程是非常有用的。数据库中的存储过程可以看做是对面向对象方法的模拟，它允许控制数据的访问方式。
- 存储过程的优点：

  a. 存储过程增强了SQL语言的功能和灵活性：存储过程可以用流控制语句编写，有很强的灵活性，可以完成复杂的判断和较复杂的运算。

  b. 存储过程允许标准组件式编程：存储过程被创建后，可以在程序中被多次调用，而不必重新编写该存储过程的SQL语句。而且可以随时对存储过程进行修改，对应用程序源代码毫无影响。

  c. 存储过程能实现较快的执行速度：如果某一操作包含大量的Transaction-SQL代码或分别被多次执行，那么存储过程要比批处理的执行速度快很多。因为存储过程是预编译的。在首次运行一个存储过程时，优化器对其进行分析优化，并且给出最终被存储在系统表中的执行计划。而批处理的Transaction-SQL语句在每次运行时都要进行编译和优化，速度相对要慢一些。

  d. 存储过程能减少网络流量：针对同一个数据库对象的操作（如查询、修改），如果这一操作所涉及的Transaction-SQL语句被组织成存储过程，那么当在客户计算机上调用该存储过程时，网络中传送的只是该调用语句，从而大大增加了网络流量并降低了网络负载。

e. 存储过程可被作为一种安全机制来充分利用：系统管理员通过执行某一存储过程的权限进行限制，能够实现对相应的数据的访问权限的限制，避免了非授权用户对数据的访问，保证了数据的安全。

- 索引
  索引（Index）是帮助MySQL高效获取数据的数据结构；在数据之外，数据库系统还维护着满足特定查找算法的数据结构，这些数据结构以某种方式引用（指向）数据，可以在这些数据结构上实现高级查找算法，提高查询速度，这种数据结构，就是索引。
  B-Tree 索引：最常见的索引类型，大部分引擎都支持B树索引。
  HASH 索引：只有Memory引擎支持，使用场景简单。
  R-Tree 索引(空间索引)：空间索引是MyISAM的一种特殊索引类型，主要用于地理空间数据类型。
  Full-text (全文索引)：全文索引也是MyISAM的一种特殊索引类型，主要用于全文索引，InnoDB从MySQL5.6版本提供对全文索引的支持。
- 事物是什么？
  事务（Transaction）是并发控制的基本单位。所谓的事务，它是一个操作序列，由一条或者多条sql语句组成，这些操作要么都执行，要么都不执行，它是一个不可分割的工作单位。
- acid特性？

  a. 原子性（Atomicity）：指整个数据库事务是不可分割的工作单位。只有事务中所有的数据库操作都执行成功，整个事务的执行才算成功。事务中任何一个sql语句执行失败，那么已经执行成功的sql语句也必须撤销，数据库状态应该退回到执行事务前的状态。

  b. 一致性（Consistency）：事务应确保数据库的状态从一个一致状态转变为另一个一致状态。一致状态的含义是数据库中的数据应满足完整性约束，也就是说在事务开始之前和事务结束以后，数据库的完整性约束没有被破坏

  c. 隔离性（Isolation）：隔离性也叫做并发控制、可串行化或者锁。事务的隔离性要求每个读写事务的对象与其它事务的操作对象能相互分离，即该事务提交前对其它事务都不可见，这通常使用锁来实现多个事务并发执行时，一个事务的执行不应影响其他事务的执行。

  d. 持久性（Durability）：表示事务一旦提交了，其结果就是永久性的，也就是数据就已经写入到数据库了，如果发生了宕机等事故，数据库也能将数据恢复。

- 数据库中的“主属性”、“码”、“主码”的区别是什么？
    a. 在数据库的表（关系）中能够用于唯一区分开每个记录（元组）的属性或属性的集合，我们称之为码（候选码）。
    b. 当我们指定其中一个用来区分开每个记录（元组）的码为主码。
    c. 主属性是指包含在候选码中的属性。
    换句话说：主码和码的关系就像班长和班长候选人之间的关系。
    每个班长候选人，我们可称之为主属性，只不过在数据库中，候选码可能是多个属性共同组成的。

智力题

- 100层楼，2个鸡蛋，鸡蛋从某一个临界楼层丢下会摔碎，请设计方案，能用最小的次数找到临界楼层
- 用3,4,5,6,7,8组成不重复的4位奇数，按照从小到大的顺序排序，第62个数字是？
- 24点游戏
- 25匹马，5个跑道，如何通过最少的比赛次数确定前3名？
- 一家人过桥问题