- **tcp和udp的区别，线程和进程的区别。**
- tcp需要经过三次握手建立客户端与服务端的连接，而udp不需要建立连接。
- tcp建立完连接才进行数据传输，因此数据的安全性、完整性、无重复性都很高；而udp只是尽可能交付，并不保证数据的有效传输。
- 但是udp具有较好的实时性，工作效率比tcp高，它适合于对高速传输和实时性有较高的要求的通信或广播通信。
- 每一条的tcp连接是一对一的，而udp支持一对一，一对多，多对一和多对多的交互通信。
- tcp对系统资源要求较多，udp对系统资源要求较少。
- 进程是资源调度的基本单位，它是在系统中并发执行的。
- 线程是进程中执行运算的最小单位。一个线程只能属于一个进程，而一个进程可以有多个线程，但至少有一个线程。
- 线程是操作系统能识别的最小执行和调度单位，系统将资源分配给进程，同一进程的所有线程共享该进程的资源 。
- 在创建或撤销进程时，由于需要分配和回收资源，所以占用的系统开销要明显大于创建或撤销线程的。
- **ddos攻击的原理，ddos攻击的防范措施。**
- ddos就是分布式拒绝服务攻击。它的攻击方式有很多种，基本原理就是利用合理的服务请求来占用服务器的大量资源，从而影响到正常用户的访问。最常见的ddos攻击就是syn洪水攻击了，它是利用了tcp协议实现上的一个缺陷，首先它发送大量的syn请求给服务端，但是在握手的第三步时不给服务端发送确认连接的信息，导致服务器不停地重试并等待一个同步超时的时间，这样就会占用服务器大量的资源和带宽等等，严重影响到正常用户的访问。常见的ddos攻击还有cc攻击，它对那些比较消耗资源的页面进行大量的访问，从而使服务器资源消耗殆尽直至宕机。
- 到目前为止并没有对ddos攻击完美防御的方法。但是我们能尽量减少它带来的影响。
- 首先我们应该尽量减少不必要的服务和端口开放，减少被ddos攻击的概率
- 限制同时打开的syn半连接数目
- 缩短syn半连接的timeout时间
- 我们可以对服务进行负载均衡来达成分布式集群防御
- 还可以通过ddos硬件防火墙对异常流量的清洗过滤、数据包的规则过滤、数据流指纹检测过滤等等顶尖技术来判断外来访问流量是否正常，然后将异常流量进行拦截
- **为什么要做动静态分离，动态数据可以分离吗？静态为什么用nginx?**
- 因为tomcat处理动态页面效率很高，但是对静态文件的处理却不如专门的静态服务器，比如apache和nginx，将动静态分离的话会大大提高用户的访问速度，而且

在高并发的情况下还能减少tomcat的并发压力。

- apache和nginx都是静态方面十分优秀的，但是会更多的企业选择nginx作为静态。因为nginx配置更简单，占用的资源更少，它采用的epoll网络模型，而apache采用的是select模型，所以它的抗并发能力更强。epoll模型和select模型区别很大，select是通过轮询检测的方式找到所有状态改变的描述符，然后再进行网络io处理，epoll模型是这样的，它无需将所有的描述符集进行检测，只要处理那些被内核io事件异步唤醒而加入ready队列的描述符就行了。因此，随着并发量的提高，select的机制会使性能急速下降，而epoll模型却几乎不会受到影响。
- **现在我们需要搭一个并发10000的网站，大概需要多少服务器，各层怎么分配，你能给一个关于架构的方案吗?**
- 关于并发的话，我之前有测过tomcat的。在内存5G，首页110k，1000M带宽，标准为响应时间500ms的配置下达到过800多并发。其它的没测过，但是根据我的了解，一台nginx的并发大概是5000，squid大概是3000，而haproxy能达到8000.所以我的架构是外围是CDN，做四个squid的缓存，源站需要一台haproxy的七层代理，下面需要10台tomcat的负载均衡以及两台nginx的静态服务器做一个动静分离。数据库方面需要做一个主从架构实现读写分离，前面还需要一个redis缓存，从服务器需要haproxy+keepalived的四层代理以及高可用。其实外部的CDN能为网站挡住大部分的并发，我的架构已经是考虑过缓存命中过低的情况了。
- **有一个nginx/apache的日志，你如何将里面访问量最大的ip找出来**
- cat access.log | awk '{print $1}' | uniq -c | sort -n | tail -n1
- **如何查看哪个进程占用了8080端口，如何查看当前服务有多少并发**
- netstat -tpln | gawk '/8080/{a[$1]++}END{for(i in a)print a[i],i}'
- netstat -tpln | grep -i esta |wc -l 或者
- ss -tpln | grep -i esta | wc -l
- **如何将本地的80端口的请求转发到本地8080端口（远程的，两种情况）**
- 本地:
- iptables -t nat -A PREROUTING -p tcp -d 127.0.0.1 --dport 80 -j DNAT --to 127.0.0.1:8080
- 远程:
- iptables -t nat -A PREROUTING -p tcp -d 127.0.0.1 --dport 80 -j DNAT --to-destination 远程ip:8080
- iptables -t nat -A POSTROUTING -p tcp -d 远程ip --sport 8080 -j SNAT --to-source 本地ip
- **你常用的监控系统性能的命令有哪些?**

- 监控内存的命令有free -m/vmstat,监控cpu的命令有sar，监控带宽状态的有iftop/ifstat，监控io状态有iotop/iostat，监控负载情况有uptime，监控端口、并发有netstat/ss。
- **如果服务器负载过大，你会怎么处理?**
- 服务器负载过大的话我们可以从内存、cpu、磁盘、网络四个方面查看。使用top命令可以查看具体某个进程占用的内存、cpu情况。iostat/iotop命令可以查看磁盘io情况，一般如果不是密集写入读取型的业务，io不会有太大的问题。
- 使用ifstat/iftop命令可以查看网络带宽情况，如果带宽占用过高，说明并发很高或者遭到了ddos攻击。可以使用netstat -tpln ｜gawk '/80/{a[$6]++}END{for (i in a) print a[i],i}'查看establish和time_wait的情况。如果establish过多说明并发很高，我们可以通过分析access日志找到ip访问频率，如果是固定的几个ip频率很高，则有可能是恶意攻击造成的。如果time_wait过多，我们需要将端口范围加大，修改/proc/sys/net/ipv4/ip_local_port_range的值将最大值加大，最小值建议1024，因为1024以内的端口一般是系统级进程占用的；修改/proc/sys/net/ipv4/tcp_tw_recycle的值为1，表示打开回收，将time_wait进程进行回收。当然，如果有图形监控的话最好了，也能最直观的发现问题，比如zabbix监控等等。