**1.** **TCP断头最小长度是\_\_\_字节？** **答案：20字节**

**2.** **简述TCP三次握手的过程？**

**答案：**

**在TCP/IP协议中，TCP协议提供可靠的连接服务，采用三次握手建立一个连接。第一次握手：建立连接时，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。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。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。完成三次握手，客户端与服务器开始传送数据**

**简版：首先A向B发SYN（同步请求），然后B回复SYN+ACK（同步请求应答），最后A回复ACK确认，这样TCP的一次连接（三次握手）的过程就建立了**

**3.** **dmesg命令中看到ip\_conntrack: table full， dropping packet.，如何解决？**

**答案：**

**ip\_conntrack表的最大值由参数ip\_conntrack\_max控制，查看当前设置的最大值可通过命令： cat /proc/sys/net/ipv4/ip\_conntrack\_max**

**redhat默认大小是65536，这个值的大小是有机器的内存决定的，65536是1G内存的大小，如果你的内存不止1G，那么设置的数值大小就是65536的倍数。2G是131072，4G是262144。数值可以比当前内存可设置的最大值小，但是不能超过最大值，要不然设置之后是不生效的。**

**要查看当前系统ip\_conntrack的大小可以通过命令：**

**cat /proc/sys/net/ipv4/netfilter/ip\_conntrack\_count**

**设置方法如下：**

**方式一：**

**echo 131072 >/proc/sys/net/ipv4/ip\_conntrack\_max**

**要实现每次开机后都生效的话，要加到开机启动脚本/etc/rc.local里，其他开机自动运行的脚本也行。**

**方式二：**

**echo 'net.ipv4.ip\_conntrack\_max=131072'  >> /etc/sysctl.conf**

**sysctl -p**

**这样每次开机都能自动设置该参数**

**4.** **简要描述Linux的启动过程？**

**答案：（这个版本有点老了，但很详细，可以帮助你复习很多知识点，面试的时候，可以简单说一下，这个答案主要是理解为主）**

**A.** **启动第一步 加载BIOS**

**当你打开计算机电源，计算机会首先加载BIOS信息，BIOS信息是如此的重要，以至于计算机必须在最开始就找到它。这是因为BIOS中包含了CPU的相关信息、设备启动顺序信息、硬盘信息、内存信息、时钟信息、PnP特性等等。在此之后，计算机心里就有谱了，知道应该去读取哪个硬件设备了。**

**B.** **启动第二步 读取MBR**

**众所周知，硬盘上第0磁道第一个扇区被称为MBR，也就是Master Boot Record，即主引导记录，它的大小是512字节，别看地方不大，可里面却存放了预启动信息、分区表信息。**

**系统找到BIOS所指定的硬盘的MBR后，就会将其复制到0×7c00地址所在的物理内存中。其实被复制到物理内存的内容就是Boot Loader，而具体到你的电脑，那就是lilo或者grub了。**

**C.** **启动第三步** **Boot Loader**

**Boot Loader** **就是在操作系统内核运行之前运行的一段小程序。通过这段小程序，我们可以初始化硬件设备、建立内存空间的映射图，从而将系统的软硬件环境带到一个合适的状态，以便为最终调用操作系统内核做好一切准备。**

**Boot Loader有若干种，其中Grub、Lilo和spfdisk是常见的Loader。**

**我们以grub为例来讲解吧，毕竟用lilo和spfdisk的人并不多。**

**系统读取内存中的grub配置信息（一般为menu.lst或grub.lst），并依照此配置信息来启动不同的操作系统。**

**D.** **启动第四步 加载内核**

**根据grub设定的内核映像所在路径，系统读取内存映像，并进行解压缩操作。此时，屏幕一般会输出“Uncompressing Linux”的提示。当解压缩内核完成后，屏幕输出“OK, booting the kernel”。**

**系统将解压后的内核放置在内存之中，并调用start\_kernel()函数来启动一系列的初始化函数并初始化各种设备，完成Linux核心环境的建立。至此，Linux内核已经建立起来了，基于Linux的程序应该可以正常运行了。**

**E.** **启动第五步 用户层init依据inittab文件来设定运行等级**

**内核被加载后，第一个运行的程序便是/sbin/init，该文件会读取/etc/inittab文件，并依据此文件来进行初始化工作。**

**其实/etc/inittab文件最主要的作用就是设定Linux的运行等级，其设定形式是“：id:5:initdefault:”，这就表明Linux需要运行在等级5上。**

**Linux的运行等级设定如下：**

**0：关机**

**1：单用户模式**

**2：无网络支持的多用户模式**

**3：有网络支持的多用户模式**

**4：保留，未使用**

**5：有网络支持有X-Window支持的多用户模式**

**6：重新引导系统，即重启**

**F.** **启动第六步** **init进程执行rc.sysinit**

**在设定了运行等级后，Linux系统执行的第一个用户层文件就是/etc/rc.d/rc.sysinit脚本程序，它做的工作非常多，包括设定PATH、设定网络配置（/etc/sysconfig/network）、启动swap分区、设定/proc等等。如果你有兴趣，可以到/etc/rc.d中查看一下rc.sysinit文件，里面的脚本够你看几天的**

**G.** **启动第七步 启动内核模块**

**具体是依据/etc/modules.conf文件或/etc/modules.d目录下的文件来装载内核模块。**

**H.** **启动第八步 执行不同运行级别的脚本程序**

**根据运行级别的不同，系统会运行rc0.d到rc6.d中的相应的脚本程序，来完成相应的初始化工作和启动相应的服务。**

**I.** **启动第九步 执行/etc/rc.d/rc.local**

**你如果打开了此文件，里面有一句话，读过之后，你就会对此命令的作用一目了然：**

**# This script will be executed \*after\* all the other init scripts.**

**# You can put your own initialization stuff in here if you don’t**

**# want to do the full Sys V style init stuff.**

**rc.local就是在一切初始化工作后，Linux留给用户进行个性化的地方。你可以把你想设置和启动的东西放到这里。**

**J.** **启动第十步 执行/bin/login程序，进入登录状态**

**此时，系统已经进入到了等待用户输入username和password的时候了，你已经可以用自己的帐号登入系统了。**

**5.** **在Linux下，如何分析一个程序达到性能瓶颈的原因，请分别从CPU、内存、IO、网络的角度判断是谁导致的瓶颈？注意现在的机器CPU是多核。**

**答案：**

**A. 用w命令看系统负载高不高，如果高可能是CPU不够用，进程太多运行太慢，也可能是磁盘I/O太高了。**

**B. 用vmstat 1命令来查看各个指标，着重分析r,b,swpd,si,so,bi,bo,us,wa. 若r列或者us列数值偏高则说明cpu有瓶颈，若b列或者wa列数值偏高同时bi或bo的数字很大，则说明磁盘有压力，若swpd一直变化，并且si和so一直不为0，则说明内存不够了。**

**C. 用top命令来分析哪个进程耗费cpu最多，默认按使用cpu多少排序，按M也可以按内存使用多少排序。**

**D. 用sar -n DEV 1 10 或者 nload 命令可以查看网卡的流量（若没有nload命令，请使用yum install epel-release; yum install -y nload 安装）**

**E. 用iotop命令可以查看磁盘读写速度 （yum install -y iotop 安装）**