**面试**

# 1 Http

## 1.1http协议介绍

http是一个基于请求与响应模式的、无状态、应用层的协议，基于TCP的连接方式。特点：无连接，限制每次连接只处理一个请求，节省传输时间；无状态，无状态是指协议对于事务处理没有记忆能力，即如果后续处理需要前面的信息，必须重传。

* *我们在传输数据时，可以只使用(传输层)TCP/IP协议，但是那样的话，如果没有应用层，便无法识别数据内容。如果想要使传输的数据有意义，则必须使用到应用层协议。应用层协议有很多，比如HTTP、FTP、TELNET等，也可以自己定义应用层协议。WEB使用HTTP协议作应用层协议，以封装HTTP文本信息，然后使用TCP/IP做传输层协议将它发到网络上*
* *socket是对TCP/IP协议的封装，Socket本身并不是协议，而是一个调用接口(API)；TCP/IP只是一个协议栈，就像操作系统的运行机制一样，必须要具体实现，同时还要提供对外的操作接口。*

*这个就像操作系统会提供标准的编程接口，比如win32编程接口一样，TCP/IP也要提供可供程序员做网络开发所用的接口，这就是Socket编程接口。*

* *传输层的TCP是基于网络层的IP协议的，而应用层的HTTP协议又是基于传输层的TCP协议的，而Socket本身不算是协议，它只是提供了一个针对TCP或者UDP编程的接口。*
* *TCP的三次握手：****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;****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。握手过程中传送的包里不包含数据，三次握手完毕后，客户端与服务器才正式开始传送数据。*

## 1.2 http请求

* http请求由三部分组成：请求行、消息报头、请求正文
* 请求方法：
  + get：请求获取request-URL所标识的资源
  + post：在request-URL所标识的资源后附加新的数据
  + head：获取资源响应消息报头，方法常用于测试超链接的有效性，是否可以访问，以及最近是否更新。
  + put：请求服务器存储一个资源，并用request-URL作为其标识
  + delete：请求服务器删除request-URL所标识的资源
  + trace：请求服务器回送收到的请求信息，主要用于测试或诊断
  + options：请求查询服务器的性能，或者查询与资源相关的选项和需求，返回服务器支持的请求方法
* header内容：
  + Accept:image/gif/text/html
  + Content-length:22
  + Connection:keep-alive
  + Cache-Control:no-cache

## 1.3 http响应

* 响应状态码：
  + 1xx : 指示信息，表示请求已接收，继续处理
  + 2xx： 成功，表示请求已被成功接收、理解、接受
  + 3xx： 重定向，要完成请求必须进行进一步的操作；301-moved permanently；302-
  + 4xx：客户端错误，请求语法错误或请求无法实现；400-bad request；401-Unauthorized；403-Forbidden；404-Not Found；
  + 5xx：服务器错误，服务器未能实现合法的请求；500-Internal Server Error；502-bad gateway；504-gateway timeout

## 1.4其他（网关代理通道）

* **代理(Proxy)**：一个中间程序，它可以充当一个服务器，也可以充当一个客户机，为其它客户机建立请求。请求是通过可能的翻译在内部或经过传递到其它的 服务器中。一个代理在发送请求信息之前，必须解释并且如果可能重写它。代理经常作为通过防火墙的客户机端的门户，代理还可以作为一个帮助应用来通过协议处 理没有被用户代理完成的请求。
* **网关(Gateway)**：一个作为其它服务器中间媒介的服务器。与代理不同的是，网关接受请求就好象对被请求的资源来说它就是源服务器；发出请求的客户机并没有意识到它在同网关打交道。网关经常作为通过防火墙的服务器端的门户，网关还可以作为一个协议翻译器以便存取那些存储在非HTTP系统中的资源。
* **通道(Tunnel)**：是作为两个连接中继的中介程序。一旦激活，通道便被认为不属于HTTP通讯，尽管通道可能是被一个HTTP请求初始化的。当被中继 的连接两端关闭时，通道便消失。当一个门户(Portal)必须存在或中介(Intermediary)不能解释中继的通讯时通道被经常使用。

## 1.5 缓存

### 直接本地缓存

直接使用浏览器端的Cache，不会发出真正的request，直到Cache过期；最省流量和服务器资源，但可能存在数据不一致，适合实时性不高的场景。

Header信息：Cache-Control: public, max-age=31536000

* public : 用户浏览器和中间proxy都会cache
* private : 只有用户端会cache
* max-age=xxx : 设置cache的最大存活时间，单位s
* no-store: 不要缓存
* no-cache: 同上，但是浏览器的支持可能不一致，最好两个一起用
* Expires: Mon, 19 Nov 2012 08:40:01 GMT   
  指定cache的*绝对*过期时间，和Cache-Control一起使用时以后者为准

### 304校验缓存

将浏览器端的Cache信息发送到服务器校验，如果服务器认为依然有效，则返回304，浏览器使用Cache；会发出一次请求且需要服务器校验，没有第一种方式那么有效，但可以保证数据实时性。

校验的两种方式：

* 基于内容ETag，ETag是Entity Tag的缩写，它由服务器端指定，通常是一个资源的内容的hash。

服务器返回给浏览器：Etag: 427fe7b6442f2096dff4f92339305444

浏览器请求传回给服务器：If-None-Match: 427fe7b6442f2096dff4f92339305444

服务器收到请求将两个值对比，若不一致，返回新数据，否则使用缓存

* 基于最后修改时间：Last-Modified

Request将本地缓存的最后修改时间通过 If-Modified-Since 发送到服务器

## 1.6 cookie与session区别

<https://www.zhihu.com/question/19786827>

sessionID是服务器发送给客户端的用于标识同一用户，比如购物车、结算类似需要验证是哪一个用户对服务器的请求，sessionID在服务器可存入内存、数据库、文件等等，客户端则会存入cookie中，在每次请求中都会把sessionID发送到服务器，若客户端禁用了cookie功能，则会通过一种重写URL的技术来识别用户（在每次请求的URL后追加sid=xxx的参数）。Cookie可以记录用户的信息（如登录账号密码，在下次访问同一网页时自动填入账号密码）

* session在服务器，cookie在客户端
* session默认被存在服务器的一个文件里（不是内存）
* session的运行依赖sessionID 而sessionID是存在cookie中的，也就是说，如果浏览器禁用了cookie 同时session也会失效（但可通过在URL中传递sessionID的方式实现）
* session可放在文件、数据库、或内存中
* 用户验证这种场合一般会用session

# linux命令

Linux系统Linux是一种[自由和开放源代码](https://zh.wikipedia.org/wiki/%E8%87%AA%E7%94%B1%E5%8F%8A%E5%BC%80%E6%94%BE%E6%BA%90%E4%BB%A3%E7%A0%81%E8%BD%AF%E4%BB%B6" \o "自由及开放源代码软件)的[类UNIX](https://zh.wikipedia.org/wiki/%E7%B1%BBUnix%E7%B3%BB%E7%BB%9F" \o "类Unix系统)[操作系统](https://zh.wikipedia.org/wiki/%E4%BD%9C%E6%A5%AD%E7%B3%BB%E7%B5%B1)。一些流行的主流Linux发布版，包括**Debian**（及其派生版本Ubuntu、Linux Mint）、**Fedora**（及其相关版本Red Hat Enterprise Linux、CentOS）和**openSUSE**等。

## 软件包管理

yum命令是**Fedora**和**Redhat**及**suse**中基于rpm的软件包管理器

选项：

-h：帮助信息

-y：对所有提问回答yes

-c：指定配置文件

-q：安静模式

-v：详细模式

-d：设置调试等级（0-10）

-e：设置错误等级（0-10）

-R: 设置yum处理一个命令的最大等待时间

-C：完全从缓存中运行，不去下载或更新任何头文件

参数：

install 安装

update 更新

check-update 检查是否有可用的rpm软件包

remove 删除指定的软件包

list 显示软件包信息

search 检查软件包信息

info 显示指定的软件包的描述信息和概要信息

clean 清理yum过期的缓存

shell 进入yum的shell提示符

resolvedep 显示rmp软件包的依赖关系

localinstall 安装本地的rmp软件包

localupdate 显示本地软件包进行更新

deplist 显示rpm软件包的所有依赖关系

示例：

yum install package1

yum info package1

yum remove package1

yum clean package1

apt-get命令是**Debian** Linux发行版本中的apt软件包管理工具,deb包相当于Windows上的安装文件。

apt-get 选项 参数

选项：

-c 指定配置文件

参数：

apt-get update 更新

apt-get install packagename

apt-get remove packagename

apt-get –purge remove packagename 卸载一个已安装的软件包（删除配置文件）

apt-get autoclean apt 删除已经删掉的软件

apt-get clean 命令会删掉已安装的软件的备份，不过不会影响软件的使用。

Apt-get 更新所有已安装的软件包

Apt-get dist-upgrade 将系统升级到新版本

## Wget

**选项：**

-q 不显示指令执行过程

-v 显示详细执行过程

-V 显示版本信息

-c 继续执行上次中断的任务

-d 调试模式运行指令

-i <文件> 从指定文件获取要下载的URL地址

-nc 文件存在时下载文件不覆盖原有文件

-nv 下载时只显示更新和出错信息，不显示指令的详细执行过程

**示例：**

wget <http://www.linux.net/test.zip> /下载过程会显示进度条，下载文件名默认为最后是个/的内容,即test.zip

wget -O newname.zip <http://www.linux.net/test.zip> 下载已不同的文件名保存

wget --limit-rate=300k <http://www.linux.net/test.zip> 限速下载

wget -c <http://www.linux.net/test.zip> 断点续传

wget -b <http://www.linux.net/test.zip> 后台下载 tail -f wget-log 查看下载进度

wget --user-agent=”xxx” <http://www.linux.net/test.zip> 伪装代理名称下载

wget --spider url 测试下载链接是否有效

wget --tries=40 url 增加重试次数

wget –i filelist.txt 下载多个文件

wget --mirror –p –convert-links –P ./LOCAL url 镜像网站

wget --reject=gif url 过滤格式下载

wget -o download.log url 下载信息存入日志文件

wget -r –A.pdf url 下载指定格式的文件（如下载一个网站的所有视频文件）

wget ftpurl 匿名下载 wget --ftp-user=xxx --ftppassword=xxx url 指定用户名密码

将当前目录下的文件a变成隐藏文件：修改文件名也可用mv a aa

mv a .a

## ps

用于列出系统中当前运行的那些进程，是当前进程的快照，想要动态的显示进程信息可用top命令

Linux进程的5种状态：对应ps工具的状态码

运行 R

中断 S

不可中断 D

僵死 Z

停止 T

选项：

x 显示所有程序不以终端机来区分

a 显示现行终端机下的所有程序，包含其他用户的程序

-a 显示所有终端机下执行的程序

c 列出程序时显示每个程序真正的指令名称，不包含路径、选项、常驻服务的标识

e 列出程序时，显示每个程序所使用的环境变量

-e /-A 显示所有程序

-f 显示UID PPID C STIME 栏位

L 列出栏位的相关信息

-l 采用详细的格式显示程序状况

m 显示所有执行绪

n 以数字来表示USER WCHAN栏位

-p <程序识别码> 指定程序识别码，并列出该程序的状况

示例：

ps aux： 列出目前所有的正在内存当中的程序,通常与grep一起用

ps –ef：显示所有进程信息，连同命令行

ps aux | egrep '(cron|syslog)' :找出与 cron 与 syslog 这两个服务有关的 PID 号码

ps –aux|more 分页查看

ps –aux > ps001.txt 输出到文件

ps –o pid,ppid,pgrp,session,tpgid,comm 输出指定的字段

## tar

参数：

-c 创建新备份文件

-d 记录文件的差别

-x 从备份文件中还原文件

-t 列出备份文件的内容

-z 通过gzip处理文件

-j 支持bzip2解压缩

-v verbose显示指令执行过程

-r 添加文件到已压缩的文件

-p 用原来的文件权限还原文件

-C 指定解压缩的目录路径

示例：

tar –cvf log.tar test.log 仅打包，不压缩

tar –zcvf log.tar.gz test.log 打包后以gzip压缩

tar –jcvf log.tar.bz2 test.log 打包后，以bzip2压缩

tar –ztvf log.tar 查看tar包内的文件

tar –zxvf log.tar.gz 解压缩文件

tar –zxvf log.tar.gz test.log 只将tar内的部分文件解压出来

tar –jxvf log.tar.bz2 –C test/tmp/

➜ test **tar -cvf tt.tar tt.log**

a tt.log

➜ test ls

rr.log tt.log tt.tar

➜ test **tar -cvf tt2.tar tt.log rr.log**

a tt.log

a rr.log

➜ test ls

rr.log tt.log tt.tar tt2.tar

➜ test **tar -ztvf tt2.tar**

-rw-r--r-- 0 ralphliu staff 0 6 21 16:30 tt.log

-rw-r--r-- 0 ralphliu staff 0 6 21 16:30 rr.log

## chmod

u user 文件或目录的拥有者

g group 文件或目录的所属群组

o other 文件或目录拥有者和群组之外的

a all 全部用户

r 读取权限4

w 写入权限2

x 执行或切换权限1

- 不具有任何权限0

s 特殊功能说明：变更文件或目录的权限

选项：

-c 类似-v参数

-R 递归处理，将指令目录下的所有文件及子目录一并处理

-v 显示指令执行过程

-f 不显示错误信息

示例：

系统中所有的账号与一般身份使用者及root相关信息都记录在/etc/passwd文件中；每个人的密码记录在/etc/shadow文件中；所有组群名称记录在/etc/group中。

chmod u+x,g+w file01 为文件file01设置自己可执行，组员可写入的权限

chmod u=rwx,g=rw,o=r file01

chomd 764 file01

chmod a+x file01

chown user:market file01 把文件file01给user，添加到market组

chown –R 9drug\_test:www /home/web\_root/www.9drug.com/\*

## rm

选项：

-f 强制删除文件或目录

-i 删除已有文件或目录之前先询问用户

-r或-R 递归处理，将指定目录下的所有文件与子目录一并处理

-v 显示指令详细执行过程

-d 把要删除的目录的硬连接数据删除成0，删除该目录

实例：

rm –i test test1 删除两个文件（交互式）

rm –r \* 删除当前目录下除隐含文件外的所有文件和子目录

## sh

sh命令是shell命令语言解释器，执行命令从标准输入或文件中读取。

bash [options] [file]

选项：

-c string ：命令从-c后的字符串读取

-i 实现脚本交互

-x 实现shell脚本逐条语句的跟踪

实例：

sh –x xxx.sh 打印出所执行的每一行命令以及当前状态

## echo

用于在shell中打印shell变量的值，或直接输出指定的字符串。

选项：

-e 激活转义字符，如\b \n 特殊处理，不会当成一般文字输出

## netstat

-p 显示正在使用socket的程序识别码和程序名称

-o 显示计时器

-n 直接使用ip地址，不通过域名服务器numeric

-i 显示网络界面信息表单interfaces

-l 显示监控中的服务器的socket listening

-t tcp连接

-u udp连接

-v 显示指令执行过程 verbose

-V version

-r 显示routing table

-s 显示网络工作信息统计表

netstat –a 列出所有端口

netstat –at 列出所有tcp端口

netstat –au 列出所有udp端口

netstat –l 只显示监听端口

netstat –lt 所有监听tcp端口 –lu –lx UNIX端口

netstat –s 显示所有端口的统计信息

netstat –pt 输出中显示pid和进程名称

netstat –an | grep ‘:80’ 找出运行在指定端口的进程

netstat –i 显示网络接口列表

netstat –anpo | grep “php-cgi” | wc -1 查看phpcgi进程数，如果接近预设值，说明不够需要增加

## wc

文件过滤与和并，计算文件byte数、字数、列数

-c 只显示bytes数 chars

-l 列数 lines

-w 字数 words

## sudo

以其他身份来执行命令，预设的身份为root 在/etc/sudoers 中设置了可执行sudo指令的用户。

选项：

sudo -l 列出目前用户可执行与无法执行的指令

-s<shell> 执行指定的shell

-u<用户> 以指定的用户作为新的身份，若不加参数则预设以root作为新的身份

-v 延长密码有效期限5分钟

示例：

配置sudo必须通过编辑/etc/sudoers文件，只有超级用户才可以修改，必须使用visudo编辑：visudo可防止两个用户同时修改它；可进行有限的语法检查。

xxx ALL=(ALL) ALL

第一个all指网络中的主机；第二个all是指目标用户，即以谁的身份去执行命令；最后一个all当然就是指所执行的命令：Foobar linux=(jack,root) /bin/kill

设置默认目标用户：Defaults:footbar runas\_default=jack

设置不需要密码执行命令：footbar localhost=NOPASSWD: /bin/cat, /bin/ls

## set

显示系统中已经存在的shell变量，以及设置shell变量的新变量值。不能够定义新的shell变量，定义新变量使用declare xx=xx

选项：

-a 标识已修改的变量，输出到环境变量

-v 显示shell所读取的输入值

示例：

declare xx=’hello’ 定义新环境变量

set –a xx 设置为环境变量

env | grep xx 查看环境变量

## env

显示系统中已存在的环境变量，以及在定义的环境中执行命令

-i 开始一个新的空的环境

-u<变量名> 从当前环境中删除指定的变量

## export

-n 删除指定的变量，变量实际并未删除，只是不会输出到后续指令的执行环境中

-p 列出所有的shell赋予程序的环境变量

export PATH=$PATH:/opt/xx/xx 设置环境变量

退出这个命令行就失效了。要想永久生效，需要把这行添加到环境变量文件里。有两个文件可 选：“/etc/profile”和用户主目录下的“.bash\_profile”，“/etc/profile”对系统里所有用户都有效，用户主目录下 的“.bash\_profile”只对这个用户有效。

## Source

source命令也称为“点命令”，也就是一个点符号（.）。source命令通常用于重新执行刚修改的初始化文件，使之立即生效，而不必注销并重新登录。source filename

## find

find . 列出当前目录及子目录下的所有文件和文件夹

find /home –name “\*.log” 在/home目录下查找以.log结尾的文件名

find /home –iname “\*.log” 忽略大小写

find . –name “\*.log” –o –name “\*.txt” 当前目录及子目录下查找log txt文件

find /usr/ -path “\*local” 查找usr目录下路径名以local结尾的目录

find . –regex xxx 基于正则表达式的方式查找

find /home ! –name “\*.txt” 不以txt结尾的文件

find –type f –atime -7 7天内被访问过的文件 +7 ，7

find . –empty 列出长度为0的文件

## 通用命令

mv cp mkdir touch which whereis rm ls

# 提高产品质量

## 产品需求设计的合理性

失败的产品设计上线后会经常性的变动需求，导致整个团队频繁的修改和测试，尤其在上线前还在修改产品逻辑的场景。需要PM前期做很充分的需求调研。

## 技术架构设计，逻辑实现要合理

技术框架设计不合理，导致在线上用户量达到量级时，爆发很严重的问题。

## 开发编码规范

开发对变量定义不规范、对输入输出参数的不规范、对异常处理的不规范、对日志处理的不规范，导致出现类似空指针、数组越界、崩溃这样低级的bug而且很难找到引起bug的原因。

## 需求评审

应起到实际的评审目的，提前将评审内容发给团队相关人，提前查阅、记录问题，带着问题去评审，效率更高、效果也会更好。

## 测试流程的规范

根据公司自己的实际情况制定，参考流程：测试需求分析、case编写、case评审、接口测试、冒烟测试、正式4轮测试、beta测试、monkey测试、提交testin兼容众测、线上环境回归测试、发布版本后安装卸载升级流程测试。

## 开发流程规范

开发代码的管理同步，代码分支开发、合并主干，提交代码review，SQL上线进行必要的review

## 上线流程规范

管理提交代码，确保分支没问题的代码才能合并到主干，再做主干整体回归测试。

## 优化功能测试的范围界定

考虑新功能对其他模块的影响，对老版本的兼容，每一次上线一定要对高频功能、核心功能做最充分的回归测试。

## 静态代码扫描

Checkstyle、FindBugs、PMD、Jtest、php depend、PHPMD等，仅通过分析或检查源程序的语法、结构、过程、接口等来检查程序的正确性，找出代码隐藏的错误和缺陷，如参数不匹配，有歧义的嵌套语句，错误的递归，非法计算，可能出现的空指针引用、错误的变量定义等等。使用静态代码分析工具自动化执行代码检查和分析，能够极大地提高软件可靠性并节省软件开发和测试成本。

## 接口测试、单元测试

一般情况下后端完成接口开发，就可以提前提测，尽早的暴露隐含的问题，让开发及时解决，方便后面的功能测试顺利进行，可缩短产品的研发周期，提高效率。

## UI走查

最好在冒烟测试时就走查一遍，test环境测试完毕走查一遍，确保不因后期修改bug带来UI问题。

## Bug预防

对bug进行分析，找出真正原因，及对应的预防避免措施，提醒测试、开发人员注意。

## 运维监控

## 复盘总结

每次项目结束进行及时的复盘总结，针对项目中出现的问题，及时做出调整，避免犯同样的错误。

## 沟通机制

产品与开发与测试之间的信息同步