Linux系统简介、安装RHEL7系统、RHEL7基本操作、总结和答疑

查看内核版本、主机名、IP/MAC地址:cat /etc/redhat-release

列出内核版本:uname -r

查看CPU型号/频率、内存大小:lscpu

检查内存大小、空闲情况:cat /proc/meminfo

重启当前系统:reboot

命令行基础、目录和文件管理、教学环境介绍、总结和答疑

快捷键

1）清理编辑的命令行

快速清屏：Ctrl + l

从当前光标处删除到行首：Ctrl + u

从当前光标处往前删除一个单词：Ctrl + w

2）放弃编辑的命令行

中止当前命令行：Ctrl + c

3）参数复用

在当前光标处粘贴上一条命令行的最后一个参数：Esc + .

SSH远程访问：ssh -X 用户@位置

ssh添加了-X选项（大写字母X），这是为了在执行远程主机的图形程序时，能够将图形界面在客户机上显示，方便用户操作

软件包管理、配置网络、文本/文件查找、总结和答疑

列出yum库:yum repolist

移除不可用的yum库配置文件:rm -rf /etc/yum.repos.d/\*.repo

添加指定的yum仓库配置:yum-config-manager --add-repo yum源

清理yum缓存：yum clean all

升级Linux内核： rpm -ivh 内核文件 重启系统：reboot 登入系统，确认使用的内核已是新版本:uname -r

查看网络连接、连接详情：nmcli con show "连接名"

配置IP地址、默认网关、DNS地址，并配置自动连接：mcli connection modify "网卡名" ipv4.method manual（手动）|auto(自动) ipv4.addresses "IP/掩码长度 默认网关" ipv4.dns DNS connection.autoconnect（配置自动连接） yes

找出所有用户 student 拥有的文件:find / -user student -type f

拷贝找到的文件到目标文件夹：find / -user student -type f -exec cp -p {} /目标文件夹 \;

使用grep命令查找指定的关键词，并通过重定向输出保存到指定的文件：grep '条件' /查找路径 > /重定向保存路径

管理用户和组、tar备份与恢复、NTP时间同步、cron计划任务、

添加组：groupadd [-g组ID] 组名称

删除组成员：groupadd -d 用户名 组名

添加组成员：groupadd -a 用户名 组名

修改组属性：groupmod [-g 组ID] [-n 新组名] 组名

删除组：groupdel 组名

添加用户：useradd [选项] 用户名 【常用选项-u 用户ID -d 用户家目录 -s 登陆shell -g基本组 -G附加组】

设置用户密码：echo “密码” |passwd --stdin 用户名

修改用户属性：usermod [选项] 用户名 【常用选项-u 用户ID -d 用户家目录 -s 登陆shell -g基本组 -G附加组】

检查用户ID信息：id 用户名

删除用户：userdel [-r] 用户名

Tar工具常用选项：-c:创建归档 -x:释放归档 -f:指定归档文档名称 -z j J：调用.gz .bz2 .xz格式的工具 -t：显示压缩包里面的东西 -P

：保持压缩包内的文件的绝对路径

制作归档压缩包：tar -zPcf 备份文件.tar.gz 文档.... tar -jPcf 备份文件.tar.bz2 文档.... tar -JPcf 备份文件.tar.xz 文档....

查看归档压缩包：tar -tf 备份文件

释放归档压缩包：tar -xf 备份文件 [-C 目标目录]

（示例）使用tar命令制作归档备份，结合-j选项调用bzip2压缩工具，保留绝对路径：tar jcPf 压缩名 要压缩的文件

配置NTP网络时间客户端：yum -y install chrony(装包) vim /etc/chrony.conf(配置server classroom.example.com iburst )timedatectl（查看现有状态）timedatectl set-ntp yes（启用NTP同步）systemctl restart chronyd（启动服务）

Cron定时任务：分 时 日 月 周 任务命令行（绝对路径） systemctl restart crond(启动服务)crontab -e -u 用户名(为用户XX添加定是任务)

恢复系统日期时间:hwclock -s

权限和归属

权限设置：chmod rwx 文件

文档归属：chown -m u:用户：权限 文件路径

Acl访问控制策略:setfacl -m [u|g]:权限 文件、

分区规划及使用、LVM逻辑卷、总结和答疑

硬盘分区及格式化： fdisk /dev/vdb（进行分区） mkfs.ext4 /dev/vdb2（将分区/dev/vdb2格式化为EXT4文件系统）

Fdisk分区指令

m：列出指令帮助

p：查看当前的分区表信息

n：新建分区

d：删除分区

t：更改分区标识

q：放弃分区更改并退出

w：保存对分区表所做的更改

分区开机自动挂载：/dev/vdb2 挂载点 文件类型 defaults 0 0

新建一个逻辑卷：vgcreate 卷组名 物理设备.. .. lvcreate -L 大小 -n 逻辑卷名 卷组名

系统安全保护 配置用户环境 配置高级连接 防火墙策略管理 总结和答疑

启用SELinux保护：查看当前模式getenforce setenforce 1

为SELinux运行模式建立固定配置:vim /etc/selinux/config

定义别名:alias 别名='复杂的命令行'

查看别名：alias、alias 别名

取消别名：unalias 别名、unalias -a

配置聚合连接:nmcli con add con-name team0 type team ifname team0 config '{ "runner":{ "name":"activebackup" } }'

Connection 'team0' (8e61d730-50ff-4a7b-8ca0-fcf5955f6ea7) successfully added.

新建聚合成员连接:nmcli con add con-name team0-p1 type team-slave ifname eth1 master team0

Connection 'team0-p1' (a62d23a2-9a2a-4855-8fbc-60ce1fd43f0b) successfully added.

配置firewalld防火墙:firewall-cmd --set-default-zone=trusted

public：仅允许访问本机的sshd等少数几个服务

trusted：允许任何访问

block：阻塞任何来访请求

drop：丢弃任何来访的数据包

封锁指定的IP网段:firewall-cmd --permanent --zone=block --add-source=IP网段

实现5423-->80端口转发:firewall-cmd --permanent --zone=trusted --add-forward-port=port=5423:proto=tcp:toport=80

**NGINX配置与调优**

**NGINX配置路径/usr/local/nginx/conf/nginx.conf**

访问Web页面需要进行用户认证:

下载软件httpd-tools.x86\_64

htpasswd -c /usr/local/nginx/pass tom //创建密码文件

auth\_basic "Input Password:"; //认证提示符

auth\_basic\_user\_file "/usr/local/nginx/pass"; //认证密码文件

基于域名的虚拟主机

server {

listen 80; //端口

server\_name www.b.com; //域名

location / {

root www; //指定网站根路径

index index.html index.htm;

}

}

配置加密网站的虚拟主机

cd /usr/local/nginx/conf

openssl genrsa > cert.key //生成私钥

openssl req -new -x509 -key cert.key > cert.pem //生成证书

ssl\_certificate cert.pem; #这里是证书文件

ssl\_certificate\_key cert.key; #这里是私钥文件

通过修改Nginx配置文件实现对PHP页面的支持

location / {

root html;

index index.php index.html index.htm;

#设置默认首页为index.php，当用户在浏览器地址栏中只写域名或IP，不说访问什么页面时，服务器会把默认首页index.php返回给用户

}

location ~ \.php$ {

root html;

fastcgi\_pass 127.0.0.1:9000; #将请求转发给本机9000端口，PHP解释器

fastcgi\_index index.php;

#fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

include fastcgi.conf;

}

Nginx的默认访问日志文件为/usr/local/nginx/logs/access.log

Nginx的默认错误日志文件为/usr/local/nginx/logs/error.log

PHP默认错误日志文件为/var/log/php-fpm/www-error.log

如果动态网站访问失败，可用参考错误日志，查找错误信息。

地址重写

rewrite 旧地址 新地址 [选项]

修改配置文件(访问a.html重定向到b.html):rewrite /a.html /b.html;

访问a.html重定向到b.html(跳转地址栏):rewrite /a.html /b.html redirect;

修改配置文件(访问192.168.4.5的请求重定向至www.tmooc.cn)

server {

listen 80;

server\_name localhost;

rewrite ^/ http://www.tmooc.cn/;

location / {

root html;

index index.html index.htm;

# rewrite /a.html /b.html redirect;

}

}

修改配置文件(访问192.168.4.5/下面子页面，重定向至www.tmooc.cn/下相同的页面)

server {

listen 80;

server\_name localhost;

rewrite ^/(.\*)$ http://www.tmooc.cn/$1;

location / {

root html;

index index.html index.htm;

# rewrite /a.html /b.html redirect;

}

}

地址重写格式【总结】

rewrite 旧地址 新地址 [选项];

last 不再读其他rewrite

break 不再读其他语句，结束请求

redirect 临时重定向

permament 永久重定向

配置Nginx服务器，添加服务器池，实现反向代理功能

http {

.. ..

#使用upstream定义后端服务器集群，集群名称任意(如webserver)

#使用server定义集群中的具体服务器和端口

upstream webserver {

server 192.168.2.100:80;

server 192.168.2.200:80;

weight可以设置后台服务器的权重，max\_fails可以设置后台服务器的 失败次数，fail\_timeout可以设置后台服务器的失败超时时间。

server 192.168.2.100 weight=1 max\_fails=1 fail\_timeout=30;

server 192.168.2.200 weight=2 max\_fails=2 fail\_timeout=30;

server 192.168.2.101 down;

#weight设置服务器权重值，默认值为1

#max\_fails设置最大失败次数

#fail\_timeout设置失败超时时间，单位为秒

#down标记服务器已关机，不参与集群调度

}

upstream webserver {

#通过ip\_hash设置调度规则为：相同客户端访问相同服务器

ip\_hash;

server 192.168.2.100 weight=1 max\_fails=2 fail\_timeout=10;

server 192.168.2.200 weight=2 max\_fails=2 fail\_timeout=10;

}

server {

listen 80;

server\_name localhost;

location / {

#通过proxy\_pass将用户的请求转发给webserver集群

proxy\_pass http://webserver;

}

}

配置Nginx服务器，添加服务器池，实现TCP/UDP反向代理功能

stream {

upstream backend {

server 192.168.2.100:22; //后端SSH服务器的IP和端口

server 192.168.2.200:22;

}

server {

listen 12345; //Nginx监听的端口

proxy\_connect\_timeout 1s;

proxy\_timeout 3s;

proxy\_pass backend;

}

}

http {

.. ..

}

启用Nginx服务并查看监听端口状态

netstat命令可以查看系统中启动的端口信息，该命令常用选项如下：

-a显示所有端口的信息

-n以数字格式显示端口号

-t显示TCP连接的端口

-u显示UDP连接的端口

-l显示服务正在监听的端口信息，如httpd启动后，会一直监听80端口

-p显示监听端口的服务名称是什么（也就是程序名称）

注意：在RHEL7系统中可以使用ss命令替代netstat命令，功能一样，选项一样。

**优化Nginx并发量:**

**修改Nginx配置文件，增加并发量**

**worker\_processes 2; //与CPU核心数量一致**

**events {**

**worker\_connections 65535; //每个worker最大并发连接数**

**use epoll;**

**}**

**.. ..**

**ulimit -a //查看所有属性值**

**ulimit -Hn 100000 //设置硬限制（临时规则）**

**ulimit -Sn 100000 //设置软限制（临时规则）**

**vim /etc/security/limits.conf**

**.. ..**

**\* soft nofile 100000**

**\* hard nofile 100000**

**优化Nginx数据包头缓存**

**http {**

**client\_header\_buffer\_size 1k; //默认请求包头信息的缓存**

**large\_client\_header\_buffers 4 4k; //大请求包头部信息的缓存个数与容量**

**.. ..**

**}**

**浏览器本地缓存静态数据**

**server {**

**listen 80;**

**server\_name localhost;**

**location / {**

**root html;**

**index index.html index.htm;**

**}**

**location ~\* \.(jpg|jpeg|gif|png|css|js|ico|xml)$ {**

**expires 30d; //定义客户端缓存时间为30天**

**}**

**}**

**自定义报错页面**

**error\_page 404 /40x.html;**

**常见http状态码**

****

**查看服务器状态信息**

**编译安装时使用--with-http\_stub\_status\_module开启状态页面模块**

**location /status {**

**stub\_status on;**

**}**

**查看状态页面信息curl http://192.168.4.5/status**

**Active connections：当前活动的连接数量。**

**Accepts：已经接受客户端的连接总数量。**

**Handled：已经处理客户端的连接总数量（一般与accepts一致，除非服务器限制了连接数量）。**

**Requests：客户端发送的请求数量。**

**Reading：当前服务器正在读取客户端请求头的数量。**

**Writing：当前服务器正在写响应信息的数量。**

**Waiting：当前多少客户端在等待服务器的响应。**

**对页面进行压缩处理**

**http {**

**.. ..**

**gzip on; //开启压缩**

**gzip\_min\_length 1000; //小文件不压缩**

**gzip\_comp\_level 4; //压缩比率**

**gzip\_types text/plain text/css application/json application/x-javascript text/xml application/xml application/xml+rss text/javascript;**

**//对特定文件压缩，类型参考mime.types**

**.. ..**

**}**

**服务器内存缓存**

**http {**

**open\_file\_cache max=2000 inactive=20s;**

**open\_file\_cache\_valid 60s;**

**open\_file\_cache\_min\_uses 5;**

**open\_file\_cache\_errors off;**

**//设置服务器最大缓存2000个文件句柄，关闭20秒内无请求的文件句柄**

**//文件句柄的有效时间是60秒，60秒后过期**

**//只有访问次数超过5次会被缓存**

**}**

**Tomcat部署虚拟主机**

**修改server.xml配置文件，创建两个域名的虚拟主机**

**<Host name="www.a.com" appBase="a" unpackWARS="true" autoDeploy="true">**

**</Host>**

**使用docBase参数可以修改默认网站首页路径**

**<Host name="www.b.com" appBase="b" unpackWARS="true" autoDeploy="true">**

**<Context path="" docBase="base" reloadable="true"/>**

**</Host>**

**<Host name="www.b.com" appBase="b" unpackWARS="true" autoDeploy="true">**

**</Host>**

**当用户访问http://www.a.com/test打开/var/www/html目录下的页面**

**<Host name="www.a.com" appBase="a" unpackWARS="true" autoDeploy="true">**

**<Context path="/test" docBase="/var/www/html/" />**

**</Host>**

**配置Tomcat支持SSL加密网站**

**keytool -genkeypair -alias tomcat -keyalg RSA -keystore /usr/local/tomcat/keystore //提示输入密码为:123456**

**//-genkeypair 生成密钥对**

**//-alias tomcat 密钥别名**

**//-keyalg RSA 定义密钥算法为RSA算法**

**//-keystore 定义密钥文件存在:/usr/local/tomcat/keystore**

**再次修改server.xml配置文件**

**<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"**

**maxThreads="150" SSLEnabled="true" scheme="https" secure="true"**

**keystoreFile="/usr/local/tomcat/keystore" keystorePass="123456" clientAuth="false" sslProtocol="TLS" />**

**//备注，默认这段Connector被注释掉了，打开注释，添加密钥信息即可**

**配置Tomcat日志**

<Host name="www.a.com" appBase="a" unpackWARS="true" autoDeploy="true">

<Context path="/test" docBase="/var/www/html/" />

<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"

prefix=" aa\_access" suffix=".txt"

pattern="%h %l %u %t &quot;%r&quot; %s %b" />

</Host>

<Host name="www.bb.com" appBase="bb" unpackWARS="true" autoDeploy="true">

<Context path="" docBase="base" />

<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"

prefix=" bb\_access" suffix=".txt"

pattern="%h %l %u %t &quot;%r&quot; %s %b" />

</Host>

**使用Varnish加速Web**

**步骤一：构建Web服务器**

**步骤二：部署Varnish缓存服务器**

**步骤三：客户端测试**

**制作RPM包**

**步骤一：安装rpm-build软件**

**yum -y install rpm-build**

**rpmbuild -ba nginx.spec**

**准备工作，将源码软件复制到SOURCES目录**

**cp nginx-1.12.2.tar.gz /root/rpmbuild/SOURCES/**

**创建并修改SPEC配置文件**

**vim /root/rpmbuild/SPECS/xxxx.spec**

**yum -y install gcc pcre-devel zlib-devel openssl-devel**

**rpmbuild -ba /root/rpmbuild/SPECS/nginx.spec**

**........OK**

**Subversion基本操作**

**YUM安装subversion软件**

**创建版本库**

**mkdir /var/svn/**

**svnadmin create /var/svn/project**

**ls /var/svn/project/**

**conf/ db/ format hooks/ locks/ README.txt**

**本地导入初始化数据**

**cd /usr/lib/systemd/system/**

**svn import . file:///var/svn/project/ -m "Init Data"**

**修改配置文件，创建账户与密码**

**vim /var/svn/project/conf/svnserve.conf**

**anon-access = none**

**//19行，匿名无任何权限**

**auth-access = write**

**//20行，有效账户可写**

**password-db = passwd**

**//27行，密码文件**

**authz-db = authz**

**//34行，ACL访问控制列表文件**

**vim /var/svn/project/conf/passwd**

**[users]**

**harry = pass**

**//用户名和密码**

**tom = pass**

**//用户名和密码**

**vim /var/svn/project/conf/authz**

**[/] //定义ACL访问控制**

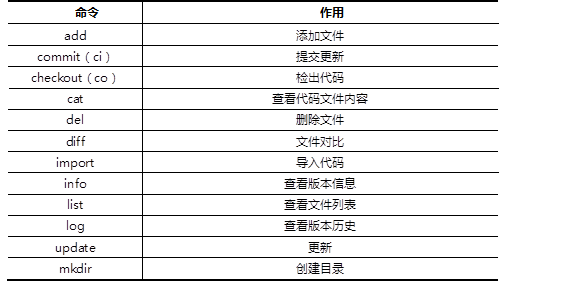
**harry = rw //用户对项目根路径可读可写**

**tom = rw**

**\* = r //其他人只读**

**svnserve -d -r /var/svn/project1 //启动服务**

**使用svn命令测试svnserver服务时可以使用的命令列表如表所示。**

****

**备份版本库数据**

**svnadmin dump /var/svn/project > project.bak //备份**

**svnadmin load /var/svn/project2 < project.bak //还原**