# 准备篇

## 常用命令

* 1. **Cp**
  2. **df**

**查看目录使用情况**

**#df –h**

* 1. **mv**
  2. **mkdir**
  3. **touch**
  4. **tar**
  5. **useradd**

**添加用户**

* 1. **groupadd**

添加www 用户组

#groupadd www

## 用户设置

**检查mysql组和用户是否存在，如无创建。**

# cat /etc/group | grep mysql

# cat /etc/passwd | grep mysql

**// useradd -r参数表示mysql用户是系统用户，不可用于登录系统。**

# groupadd mysql

# useradd -r -g mysql mysql

**创建用户:**

# useradd zhang

**设置密码:**

# passwd zhang

**输入两次密码即可**

**设置用户登陆的shell :**

# useradd -d /www -s /usr/bin/passwd zhang

删除用户 ： # userdel zhang

## IP地址设置

* 1. **编辑文件:**

#vi /etc/sysconfig/network-scripts/ifcfg-eth0

HWADDR="00:15:5D:07:F1:02" #网卡地址，也就是MAC地址

TYPE="Ethernet" #网络类型：这里是以太网

DEFROUTE="yes"

PEERDNS="yes"

PEERROUTES="yes"

IPV4\_FAILURE\_FATAL="no"

IPV6INIT="no" #是否启用IPV6协议

IPV6\_AUTOCONF="yes"

IPV6\_DEFROUTE="yes"

IPV6\_PEERDNS="yes"

IPV6\_PEERROUTES="yes"

IPV6\_FAILURE\_FATAL="no"

NM\_CONTROLLED=yes # network mamager的参数，实时生效，

USERCTL=no #是否允许非root用户控制该设备 no只能用root用户更改

NAME="eth0" # 网卡名称

UUID="bb3a302d-dc46-461a-881e-d46cafd0eb71"

BOOTPROTO="static" #开机协议，有none,static,dhcp,bootp

ONBOOT="yes" #开机启用本配置

IPADDR=192.168.7.106 #静态IP

GATEWAY=192.168.7.1 #默认网关

NETMASK=255.255.255.0 #子网掩码

DNS1=192.168.7.1 #DNS 配置

* 1. **重启网络服务:**

# systemctl restart network

* 1. **查看IP地址：**

#ip addr

## 主机名设置

## 时间同步

* 1. **安装ntpdate工具**

# yum -y install ntp ntpdate

* 1. **设置系统时间与网络时间同步**

# ntpdate cn.pool.ntp.org

time.nuri.net

* 1. **将系统时间写入硬件时间**

# hwclock --systohc

## 关闭FireWall

停止firewall

#systemctl stop firewalld.service

禁止firewall开机启动

#systemctl disable firewalld.service

查看默认防火墙状态（关闭后显示notrunning，开启后显示running）

#firewall-cmd –state

## 关闭SELINUX

#vi /etc/selinux/config

#SELINUX=enforcing #注释掉

#SELINUXTYPE=targeted #注释掉

SELINUX=permissive #增加

:wq! #保存退出

使配置立即生效

##设置SELinux 成为permissive模式

##setenforce 1 设置SELinux 成为enforcing模式

#setenforce 0

## 设置iptables

* 1. **安装iptables**

#yum install iptables-services

编辑防火墙配置文件

#vi /etc/sysconfig/iptables

**\*filter**

**:INPUT ACCEPT [0:0]**

**:FORWARD ACCEPT [0:0]**

**:OUTPUT ACCEPT [0:0]**

**-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT**

**-A INPUT -p icmp -j ACCEPT**

**-A INPUT -i lo -j ACCEPT**

**-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT**

**-A INPUT -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT**

**-A INPUT -p tcp -m state --state NEW -m tcp --dport 3306 -j ACCEPT**

**-A INPUT -p tcp -m state --state NEW -m tcp --dport 10000 -j ACCEPT**

**-A INPUT -j REJECT --reject-with icmp-host-prohibited**

**-A FORWARD -j REJECT --reject-with icmp-host-prohibited**

**COMMIT**

:wq! #保存退出

最后重启防火墙使配置生效

#systemctl restart iptables.service

设置防火墙开机启动

#systemctl enable iptables.service

## 工具安装

* 1. **wget**
     1. **安装wget**

# yum -y install wget

* + 1. **使用**

#wget url

* 1. **zip,unzip**
     1. **安装**

# yum -y install zip unzip

* + 1. **使用**

压缩mydata目录

# zip -r mydata.zip mydata

mydata.zip解压到mydatabak目录里面

# unzip mydata.zip -d mydatabak

* 1. **RZ,SZ**
     1. **安装**

#yum –y install lrzsz

* + 1. **使用**
  1. **telnet**

查看是否安装

# rpm -qa | grep telnet

# rpm -qa | grep xinetd

# yum -y install xinetd telnet telnet-server

设置开机启动

# chkconfig telnet on

将xinetd服务加入开机自启动:

# systemctl enable xinetd.service

将telnet服务加入开机自启动：

#systemctl enable telnet.socket

telnet服务也是由xinetd守护的，所以安装完telnet-server，要启动telnet服务就必须重新启动xinetd

# systemctl start telnet.socket

# systemctl start xinetd

* 1. **gcc-c++**

# yum -y install libevent

## JDK

* 1. **安装**

检测jdk和卸载已安装:

# rpm -qa|grep jdk

# yum -y remove java-1.8.0-openjdk-headless-1.8.0.65-3.b17.el7.x86\_64

上传tar文件到/usr/local/software

# rz

创建安装目录 :

# mkdir /usr/local/java

解压文件到/usr/local/java

# tar -zxvf jdk-8u162-linux-x64.tar.gz -C /usr/local/java

* 1. **设置环境变量**

# vi /etc/profile

在/etc/profile文件末尾添加:

**JAVA\_HOME=/usr/local/java/jdk1.8.0\_66**

**JRE\_HOME=/usr/local/java/jdk1.8.0\_66/jre**

**CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar:$JRE\_HOME/lib**

**PATH=$JAVA\_HOME/bin:$PATH**

**export PATH JAVA\_HOME JRE\_HOME CLASSPATH**

使/etc/profile生效:

# source /etc/profile

检测是否安装成功:

# java -version

成功显示：

**java version "1.8.0\_162"**

**Java(TM) SE Runtime Environment (build 1.8.0\_162-b12)**

**Java HotSpot(TM) 64-Bit Server VM (build 25.162-b12, mixed mode)**

# 应用服务器篇

## Tomcat

* 1. **Tomcat8.0.48**

**解压压缩包:**

#tar -zxvf apache-tomcat-8.0.26.tar.gz

**启动tomcat:**

# /usr/local/tomcat/apache-tomcat-8.0.48/bin/startup.sh

**停止tomcat:**

# /usr/local/tomcat/apache-tomcat-8.0.48/bin/shutdown.sh

**修改tomcat 端口为80 ：**

# vi /usr/local/tomcat/apache-tomcat-8.0.48/conf/server.xml

**增加80端口到防火墙:**

# vi /etc/sysconfig/iptables

**-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT**

**保存配置信息:**

#service iptables save

**重启配置信息 ：**

# service iptables restart

**访问页面验证是否成功 ： http://192.168.128.127:8080**

* 1. **Tomcat7.0.82**

## WebLogic

## JBOSS

## RESIN

# 数据库篇

## MySql

1. **检查并卸载系统自带的 mariadb-lib，mysql**

# rpm -qa | grep mysql

# rpm -e mysql --nodeps

# rpm -qa|grep mariadb

**mariadb-libs-5.5.44-2.el7.centos.x86\_64**

# rpm -e mariadb-libs-5.5.44-2.el7.centos.x86\_64 –nodeps

**创建mysql文件目录**

#mkdir /usr/local/java/mysql

1. **RPM 文件安装**

**获取文件：**

#wget <http://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-5.7.16-1.el7.x86_64.rpm-bundle.tar>

**解压文件**

#tar -xvf mysql-5.7.16-1.el7.x86\_64.rpm-bundle.tar

mysql-community-libs-compat-5.7.16-1.el7.x86\_64.rpm  
mysql-community-devel-5.7.16-1.el7.x86\_64.rpm  
mysql-community-minimal-debuginfo-5.7.16-1.el7.x86\_64.rpm  
mysql-community-libs-5.7.16-1.el7.x86\_64.rpm  
mysql-community-common-5.7.16-1.el7.x86\_64.rpm  
mysql-community-embedded-compat-5.7.16-1.el7.x86\_64.rpm  
mysql-community-test-5.7.16-1.el7.x86\_64.rpm  
  mysql-community-embedded-devel-5.7.16-1.el7.x86\_64.rpm  
  mysql-community-server-minimal-5.7.16-1.el7.x86\_64.rpm  
  mysql-community-server-5.7.16-1.el7.x86\_64.rpm  
  mysql-community-client-5.7.16-1.el7.x86\_64.rpm  
  mysql-community-embedded-5.7.16-1.el7.x86\_64.rpm

**依次执行（几个包有依赖关系，所以执行有先后）下面命令安装：**

rpm -ivh mysql-community-common-5.7.16-1.el7.x86\_64.rpm

rpm -ivh mysql-community-libs-5.7.16-1.el7.x86\_64.rpm

rpm -ivh mysql-community-client-5.7.16-1.el7.x86\_64.rpm

rpm -ivh mysql-community-server-5.7.16-1.el7.x86\_64.rpm

**在安装rpm -ivh mysql-community-server-5.7.16-1.el7.x86\_64.rpm的时候报错如下：**  
  warning: mysql-community-server-5.7.16-1.el7.x86\_64.rpm: Header V3 DSA/SHA1 Signature, key ID 5072e1f5: NOKEY  
  error: Failed dependencies:  
  libaio.so.1()(64bit) is needed by mysql-community-server-5.7.16-1.el7.x86\_64  
  libaio.so.1(LIBAIO\_0.1)(64bit) is needed by mysql-community-server-5.7.16-1.el7.x86\_64  
  libaio.so.1(LIBAIO\_0.4)(64bit) is needed by mysql-community-server-5.7.16-1.el7.x86\_64  
  net-tools is needed by mysql-community-server-5.7.16-1.el7.x86\_64

  这个报错的意思是需要安装libaio包和net-tools包：可以yum安装一下，安装 libaio-0.3.107-10.el6.x86\_64.rpm，下载地址：

wget <http://mirror.centos.org/centos/6/os/x86_64/Packages/libaio-0.3.107-10.el6.x86_64.rpm>然后yum安装net-tools yum install net-tools 完成之后就可以顺利安装

1. **YUM安装：**

**获取yum安装源：**

# wget <https://repo.mysql.com//mysql57-community-release-el7-> 11.noarch.rpm

# rpm -Uvh mysql57-community-release-el6-11.noarch.rpm

**查看安装结果:**

# yum repolist enabled | grep "mysql.\*-community.\*"

# yum repolist all | grep mysql

# yum repolist enabled | grep mysql

**修改yum默认安装版本:**

# vi /etc/yum.repos.d/mysql-community.repo

**修改 enable=1 ,其他的enable=0**

[mysql57-community]

name=MySQL 5.7 Community Server

baseurl=http://repo.mysql.com/yum/mysql-5.7-community/el/6/$basearch/

enabled=1

gpgcheck=1

gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-mysql

# yum install mysql-community-server

**查看并设置开机启动:**

# systemctl list-unit-files | grep mysql

# systemctl enable mysqld

**守护进程启动**

# systemctl daemon-reload

1. **授权远程连接:**

#GRANT ALL PRIVILEGES ON \*.\* TO 'root'@'%' IDENTIFIED BY 'yourpassword' WITH GRANT OPTION;

#FLUSH PRIVILEGES

1. **数据库密码修改**

# systemctl start mysqld.service

**查看状态**

# systemctl status mysqld.service

# mysql -uroot –p

**查找root密码:**

# sudo grep 'temporary password' /var/log/mysqld.log

**修改root密码：**

ALTER USER 'root'@'localhost' IDENTIFIED BY 'MyNewPass4!';

**如果你设置的密码过于简单也会报错；**

1. **数据库my.cnf配置**

[mysqld]

Basedir = path # 使用给定目录作为根目录(安装目录)。

Datadir = path # 从给定目录读取数据库文件。

pid-file = filename # 为mysqld程序指定一个存放进程ID的文件

socket = /tmp/mysql.sock # 为MySQL客户程序与服务器之间的本地通信指定一个套接字文件(Linux下默认是/var/lib/mysql/mysql.sock文件)

port = 3306 # 指定MsSQL侦听的端口

key\_buffer = 384M # key\_buffer是用于索引块的缓冲区大小，增加它可得到更好处理的索引(对所有读和多重写)。 索引块是缓冲的并且被所有的线程共享，key\_buffer的大小视内存大小而定。

table\_cache = 512 # 为所有线程打开表的数量。增加该值能增加mysqld要求的文件描述符的数量。可以避免频繁的打开数据表产生的开销

sort\_buffer\_size = 2M # 每个需要进行排序的线程分配该大小的一个缓冲区。增加这值加速ORDER BY或GROUP BY操作。 注意：该参数对应的分配内存是每连接独占！如果有100个连接，那么实际分配的总共排序缓冲区大小为100×6=600MB

read\_buffer\_size = 2M # 读查询操作所能使用的缓冲区大小。和ort\_buffer\_size一样，该参数对应的分配内存也是每连接独享。

query\_cache\_size = 32M # 指定MySQL查询结果缓冲区的大小

read\_rnd\_buffer\_size = 8M # 改参数在使用行指针排序之后，随机读用的。

myisam\_sort\_buffer\_size =64M # MyISAM表发生变化时重新排序所需的缓冲

thread\_concurrency = 8 # 最大并发线程数，取值为服务器逻辑CPU数量×2，如果CPU支持H.T超线程，再×2

thread\_cache = 8 #缓存可重用的线程数

skip-locking # 避免MySQL的外部锁定，减少出错几率增强稳定性。

[mysqldump]

max\_allowed\_packet =16M # 服务器和客户端之间最大能发送的可能信息包

1. **优化配置：**

[mysqld]

pid-file=/var/run/mysqld/mysqld.pid

datadir=/var/lib/mysql

socket=/var/lib/mysql/mysql.sock

symbolic-links=0

########basic settings########

server-id = 11

port = 3306

user = mysql

autocommit = 1

character\_set\_server=utf8mb4

skip\_name\_resolve = 1

max\_connections = 800

max\_connect\_errors = 1000

transaction\_isolation = READ-COMMITTED

join\_buffer\_size = 128M

tmp\_table\_size = 64M

tmpdir = /tmp

max\_allowed\_packet = 64M

interactive\_timeout = 1200

wait\_timeout = 600

read\_buffer\_size = 16M

read\_rnd\_buffer\_size = 8M

sort\_buffer\_size = 8M

########log settings########

log-error=/var/log/mysqld.log

slow\_query\_log = 1

long\_query\_time = 6

slow\_query\_log\_file = /var/logs/mysql/slow.log

log\_queries\_not\_using\_indexes = 1

log\_slow\_admin\_statements = 1

log\_slow\_slave\_statements = 1

log\_throttle\_queries\_not\_using\_indexes = 10

expire\_logs\_days = 90

min\_examined\_row\_limit = 100

########replication settings########

master\_info\_repository = TABLE

relay\_log\_info\_repository = TABLE

log\_bin = bin.log

sync\_binlog = 5

gtid\_mode = on

enforce\_gtid\_consistency = 1

log\_slave\_updates = 1

binlog\_format = mixed

relay\_log = /var/logs/mysql/relay.log

relay\_log\_recovery = 1

binlog\_gtid\_simple\_recovery = 1

slave\_skip\_errors = ddl\_exist\_errors

innodb\_page\_size = 16K

innodb\_buffer\_pool\_size = 1

innodb\_buffer\_pool\_instances = 4G

innodb\_buffer\_pool\_load\_at\_startup = 1

innodb\_buffer\_pool\_dump\_at\_shutdown = 1

innodb\_lru\_scan\_depth = 2000

innodb\_io\_capacity = 4000

innodb\_io\_capacity\_max = 8000

innodb\_lock\_wait\_timeout = 30

innodb\_log\_group\_home\_dir = /var/logs/mysql/

innodb\_file\_format = Barracuda

innodb\_file\_format\_max = Barracuda

innodb\_strict\_mode = 1

innodb\_purge\_threads = 1

innodb\_large\_prefix = 1

innodb\_thread\_concurrency = 8

innodb\_print\_all\_deadlocks = 1

innodb\_sort\_buffer\_size = 16M

## Oracle

## SqlServer

# NoSql缓存篇

## Redis

1. **安装和使用**

#mkdir /usr/local/java/

#wget http://download.redis.io/releases/redis-4.0.8.tar.gz

# tar xzf redis-4.0.8.tar.gz

# cd redis-4.0.8

# make

**启动服务端 :配置文件redis.conf**

**#** src/redis-server redis.conf

**启动客户端：**You can interact with Redis using the built-in client:

**#** src/redis-cli

set foo bar

get foo

1. **redis.conf 配置项说明如下：**

1. Redis默认不是以守护进程的方式运行，可以通过该配置项修改，使用yes启用守护 进程

**daemonize no**

2. 当Redis以守护进程方式运行时，Redis默认会把pid写入/var/run/redis.pid文件， 可以通过pidfile指定

**pidfile /var/run/redis.pid**

3. 指定Redis监听端口，默认端口为6379，作者在自己的一篇博文中解释了为什么选 用6379作为默认端口，因为6379在手机按键上MERZ对应的号码，而MERZ取 自意大利歌女Alessia Merz的名字

**port 6379**

4. 绑定的主机地址

**bind 127.0.0.1**

5.当 客户端闲置多长时间后关闭连接，如果指定为0，表示关闭该功能

**timeout 300**

6. 指定日志记录级别，Redis总共支持四个级别：debug、verbose、notice、warning， 默认为verbose

**loglevel verbose**

7. 日志记录方式，默认为标准输出，如果配置Redis为守护进程方式运行，而这里又配 置为日志记录方式为标准输出，则日志将会发送给/dev/null

**logfile stdout**

8. 设置数据库的数量，默认数据库为0，可以使用SELECT <dbid>命令在连接上指定数 据库id

**databases 16**

9. 指定在多长时间内,有多少次更新操作,就将数据同步到数据文件,可以多个条件配合

**save <seconds> <changes>**

Redis默认配置文件中提供了三个条件：

**save 900 1**

**save 300 10**

**save 60 10000**

分别表示900秒（15分钟）内有1个更改，300秒（5分钟）内有10个更改以及 60秒内有10000个更改。

10. 指定存储至本地数据库时是否压缩数据，默认为yes，Redis采用LZF压缩，如果为 了节省CPU时间，可以关闭该选项，但会导致数据库文件变的巨大

**rdbcompression yes**

11. 指定本地数据库文件名，默认值为dump.rdb

**dbfilename dump.rdb**

12. 指定本地数据库存放目录

**dir ./**

13. 设置当本机为slav服务时，设置master服务的IP地址及端口，在Redis启动时， 它会自动从master进行数据同步

**slaveof <masterip> <masterport>**

14. 当master服务设置了密码保护时，slav服务连接master的密码

**masterauth <master-password>**

15. 设置Redis连接密码，如果配置了连接密码，客户端在连接Redis时需要通过AUTH <password>命令提供密码，默认关闭

**requirepass foobared**

16. 设置同一时间最大客户端连接数，默认无限制，Redis可以同时打开的客户端连接 数为Redis进程可以打开的最大文件描述符数，如果设置 maxclients 0，表示不作 限制。当客户端连接数到达限制时，Redis会关闭新的连接并向客户端返回max number of clients reached错误信息

**maxclients 128**

17. 指定Redis最大内存限制，Redis在启动时会把数据加载到内存中，达到最大内存 后，Redis会先尝试清除已到期或即将到期的Key，当此方法处理 后，仍然到达最 大内存设置，将无法再进行写入操作，但仍然可以进行读取操作。Redis新的vm机 制，会把Key存放内存，Value会存放在swap区

**maxmemory <bytes>**

18. 指定是否在每次更新操作后进行日志记录，Redis在默认情况下是异步的把数据写 入磁盘，如果不开启，可能会在断电时导致一段时间内的数据丢失。因为 redis本 身同步数据文件是按上面save条件来同步的，所以有的数据会在一段时间内只存 在于内存中。默认为no

**appendonly no**

19. 指定更新日志文件名，默认为appendonly.aof

**appendfilename appendonly.aof**

20. 指定更新日志条件，共有3个可选值：

no：表示等操作系统进行数据缓存同步到磁盘（快）

always：表示每次更新操作后手动调用fsync()将数据写到磁盘（慢，安全）

everysec：表示每秒同步一次（折衷，默认值）

**appendfsync everysec**

21. 指定是否启用虚拟内存机制，默认值为no，简单的介绍一下，VM机制将数据分页 存放，由Redis将访问量较少的页即冷数据swap到磁盘上，访问多的页面由磁盘 自动换出到内存中（在后面的文章我会仔细分析Redis的VM机制）

**vm-enabled no**

22. 虚拟内存文件路径，默认值为/tmp/redis.swap，不可多个Redis实例共享

vm-swap-file /tmp/redis.swap

23. 将所有大于vm-max-memory的数据存入虚拟内存,无论vm-max-memory设置多 小,所有索引数据都是内存存储的(Redis的索引数据 就是keys),也就是说,当vm- max-memory设置为0的时候,其实是所有value都存在于磁盘。默认值为0

**vm-max-memory 0**

24. Redis swap文件分成了很多的page，一个对象可以保存在多个page上面，但一个 page上不能被多个对象共享，vm-page-size是要根据存储的 数据大小来设定的， 作者建议如果存储很多小对象，page大小最好设置为32或者64bytes；如果存储 很大大对象，则可以使用更大的page，如果不 确定，就使用默认值

**vm-page-size 32**

25. 设置swap文件中的page数量，由于页表（一种表示页面空闲或使用的bitmap） 是在放在内存中的，，在磁盘上每8个pages将消耗1byte的内存。

**vm-pages 134217728**

26. 设置访问swap文件的线程数,最好不要超过机器的核数,如果设置为0,那么所有对 swap文件的操作都是串行的，可能会造成比较长时间的延迟。默认值为4

**vm-max-threads 4**

27. 设置在向客户端应答时，是否把较小的包合并为一个包发送，默认为开启

**glueoutputbuf yes**

28. 指定在超过一定的数量或者最大的元素超过临界值时，采用特殊的哈希算法

**hash-max-zipmap-entries 64**

**hash-max-zipmap-value 512**

29. 指定是否激活重置哈希，默认为开启（后面在介绍Redis的哈希算法时具体介绍）

**activerehashing yes**

30. 指定包含其它的配置文件，可以在同一主机上多个Redis实例之间使用同一份配置 文件，而同时各个实例又拥有自己的特定配置文件

**include /path/to/local.conf**

## MongoDB

## Memcahed

**1.获取文件**： # wget http://memcached.org/files/memcached-1.5.4.tar.gz

**2.解压文件**： # tar -zxvf memcached-1.5.4.tar.gz

**3.进入目录**： # cd memcached-1.5.4

**4.安装：** # ./configure && make && make test && sudo make install

**5. configure: error: libevent is required**

**检查libevent**  : # rpm -qa|grep libevent

**没有安装:** # yum -y install libevent libevent-devel

**6.启动memcached:**

# /usr/local/bin/memcached -d -m 512 -l 127.0.0.1 -p 10000 -u root

表示已守护进程的方式启动，监听于 127.0.0.1 的10000端口，使用root用户，最大 使用512M内存

可以同时开多个memcached，但要监听在不同的端口.

说明：如果没有带 -u root 的话就会报：

can't run as root without the -u switch

解决：带-u root就行！

**备注**：

$ /usr/local/memcached/bin/memcached -d -u root -l 192.168.56.10 -p 32054 -m 10 - c 100 -P /tmp/memcached\_32054.pid

#选项说明，这里只列出比较重要的选项，具体选项说明使用memcached -h来查阅

-p TCP端口，默认为11211，可以不设置

-U UDP端口，默认为11211，0为关闭

-l 监听的ip地址

-d 守护进程（daemon）作为daemon在后台启动

-u 指定用户，如果当前为 root ，需要使用此参数指定用户

-m 最大内存，单位MB。默认64MB，32位操作系统，每个进程最多只能使用2GB， 64位无限制

-M 禁止LRU策略，内存耗尽时返回错误，而不是删除数据

-c 最大连接数，默认是1024

-vv 用very vrebose模式启动，调试信息和错误输出到控制台

-P memcache的pid文件，结束memcache进程：kill `cat /tmp/memcached\_32054.pid`

-f 增长因子，默认1.25

-n 初始chunk=key+suffix+value+32结构体，默认48字节

-L 启用大内存页，可以降低内存浪费，改进性能

-t 线程数，默认4。由于memcached采用NIO，所以更多线程没有太多作用

-R 每个event连接最大并发数，默认20

-C 禁用CAS命令（可以禁止版本计数，减少开销）

-I 每次申请内存的页的大小（page），默认1M，最小1k，最大128M

-F 禁用flush\_all

# 消息队列篇

## JMS

## ActiveMQ

## Kafka

## RabbitMQ

# 负载均衡篇

## Nginx

**1.切换文件目录**

# cd /usr/local/src/nginx

# tar zxvf nginx-1.5.0.tar.gz

# cd nginx-1.5.0

**2配置。通常将软件安装在/usr/local/目录下**

**添加www组**

**#**  groupadd www

**创建nginx运行账户www并加入到www组，不允许www用户直接登录系统**

**#** useradd -g www www -s /bin/false

# ./configure --user=www --group=www --prefix=/usr/local/nginx --with- http\_stub\_status\_module --with-http\_ssl\_module --with-http\_realip\_module

**3编译 安装**

# make

# make install

4 检查是否安装成功

# cd /usr/local/nginx/sbin

# ./nginx -t

**结果显示：**

nginx: the configuration file /usr/local/nginx/conf/nginx.conf syntax is ok

nginx: configuration file /usr/local/nginx/conf/nginx.conf test is successful

**启动**

# /usr/local/nginx/sbin/nginx

**停止/重启**

# /usr/local/nginx/sbin/nginx -s stop(quit、reload)

**命令帮助**

# /usr/local/nginx/sbin/nginx -h

**验证配置文件**

# /usr/local/nginx/sbin/nginx -t

**配置文件**

# vi /usr/local/nginx/conf/nginx.conf

**PS:**

每一个server就是一个虚拟主机，我们有一个当作web服务器来使用

listen 80;代表监听80端口

server\_name xxx.com;代表外网访问的域名

location / {};代表一个过滤器，/匹配所有请求，我们还可以根据自己的情况定义不同的过滤，比如对静态文件js、css、image制定专属过滤

root html;代表站点根目录

index index.html;代表默认主页

负载均衡功能往往在接收到某个请求后分配到后端的多台服务器上，那我们就需要upstream{}块来配合使用

upstream xxx{};upstream模块是命名一个后端服务器组，组名必须为后端服务器站点域名，内部可以写多台服务器ip和port，还可以设置跳转规则及权重等等

ip\_hash;代表使用ip地址方式分配跳转后端服务器，同一ip请求每次都会访问同一台后端服务器

server;代表后端服务器地址

server{};server模块依然是接收外部请求的部分

server\_name;代表外网访问域名

location / {};同样代表过滤器，用于制定不同请求的不同操作

proxy\_pass;代表后端服务器组名，此组名必须为后端服务器站点域名

server\_name和upstream{}的组名可以不一致，server\_name是外网访问接收请求的域名，upstream{}的组名是跳转后端服务器时站点访问的域名

# 框架篇

## Spring

## Spring MVC

## Spring Boot

## MyBatis

## Hibernate

## WebService

## Shiro

# 大数据篇

## Hadoop

## Spark

## Apache Storm

# 文本搜索篇

## Apache Lucene

## Solr

## ElasticSearch

# 工具篇

## Svn

1. **安装**

# sudo yum install subversion

subv**ersion安装在/bin目录：**

# which svnserve

/bin/svnserve

**检查一下subversion是否安装成功:**

#svnserve --version  
svnserve, version 1.7.14 (r1542130)  
 compiled Nov 20 2015, 19:25:09

1. **建立版本库**

**subversion默认以/var/svn作为数据根目录，可以通过/etc/sysconfig/svnserve修改这个默认位置。**

# systemctl cat svnserve.service

**修改/etc/sysconfig/svnserver将默认目录指定到/opt/svn。**

# cat /etc/sysconfig/svnserve

OPTIONS="-r /var/svn"

**使用svnadmin建立版本库yltd**

# sudo mkdir -p /var/svn  
# sudo svnadmin create /var/svn/yltd

1. **配置**

**编辑用户文件passwd，新增两个用户：admin和guest：**

# cat /opt/svn/spring-hello-world/conf/passwd

[users]

admin = admin

guest = guest

**编辑权限文件authz，用户admin设置可读写权限，guest设置只读权限：**

# cat /opt/svn/spring-hello-world/conf/authz

[/]

admin = rw

guest = r

**编辑svnserve.conf：**

# cat /opt/svn/spring-hello-world/conf/svnserve.conf

[general]

anon-access = none #控制非鉴权用户访问版本库的权限

auth-access = write #控制鉴权用户访问版本库的权限

password-db = passwd #指定用户名口令文件名

authz-db = authz #指定权限配置文件名

realm = spring-hello-world #指定版本库的认证域，即在登录时提示的认证域名称

1. **SVN服务**

**启动SVN服务**

# sudo systemctl start svnserve.service

#svnserve -d -r /mnt/westos --listen-port 3691（centos 6）

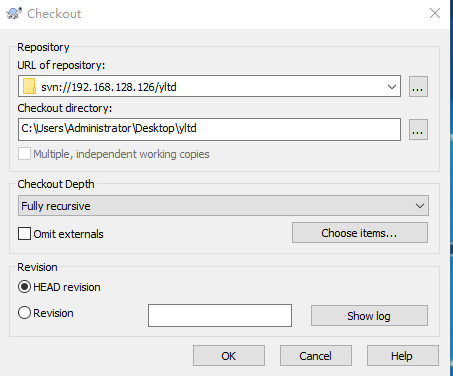
**检查服务是否启动成功**

# ps aux | grep svn

**设置成开机启动**

# sudo systemctl enable svnserve.service

1. **客户端测试**



PS:

svnserve.conf:12: Option expected”

因为subversion读取配置文件svnserve.conf时，无法识别有前置空格的配置文件

# 工作流

## Activiti

# 其它