Redis是一种高级key-value数据库。它跟memcached类似，不过数据可以持久化，而且支持的数据类型很丰富。有字符串，链表，集 合和有序集合。支持在服务器端计算集合的并，交和补集(difference)等，还支持多种排序功能。所以Redis也可以被看成是一个数据结构服务器。

Redis的所有数据都是保存在内存中（效率高），然后不定期的通过异步方式保存到磁盘上(这称为“半持久化模式”)；也可以把每一次数据变化都写入到一个append only file(aof)里面(这称为“全持久化模式”)。

在这里我仅仅给出Redis在[CentOS](http://www.linuxidc.com/topicnews.aspx?tid=14" \o "CentOS" \t "http://www.linuxidc.com/Linux/2017-05/_blank) 7上的安装部署。

**步骤一：下载Redis安装包**

首先从官网下在redis正式版的压缩包redis-2.8.19.tar.gz：

cd /home/downloads

wget http://download.redis.io/releases/redis-2.8.19.tar.gz

**步骤二：编译源程序**

解压和编译

tar -zxvf redis-2.8.19.tar.gz

cd redis-2.8.19/src

//编译

make

…….此处是大量编译过程，省略。可能有一些警告，不去官它们………….   
CC setproctitle.o   
CC hyperloglog.o   
CC latency.o   
CC sparkline.o   
LINK redis-server   
INSTALL redis-sentinel   
CC redis-cli.o   
LINK redis-cli   
CC redis-benchmark.o   
LINK redis-benchmark   
CC redis-check-dump.o   
LINK redis-check-dump   
CC redis-check-aof.o   
LINK redis-check-aof

Hint: It’s a good idea to run ‘make test’ ;)

进入src进行安装:

cd src

make install

//安装过程提示   
Hint: It’s a good idea to run ‘make test’ ;)

INSTALL install

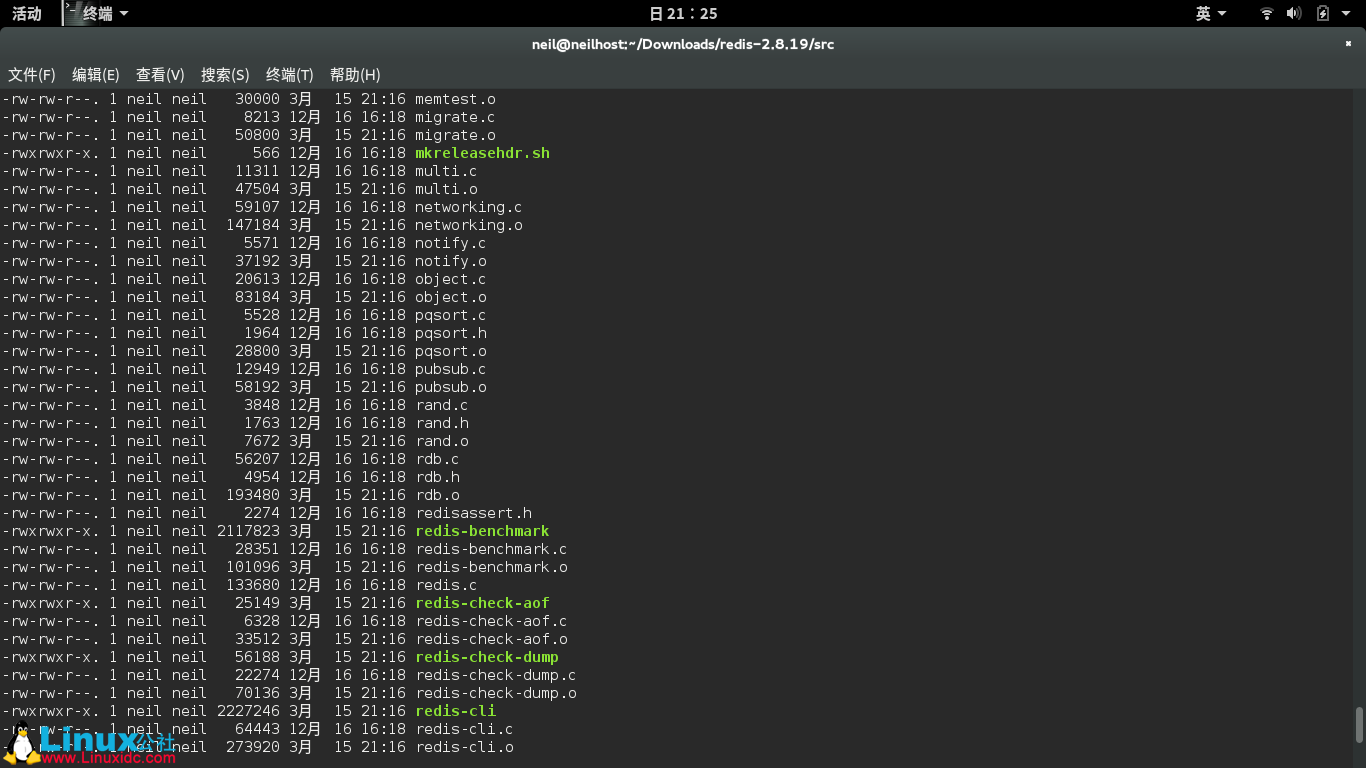
INSTALL install

INSTALL install

INSTALL install

INSTALL install

这时候，我们可以看看src下的文件：



可以看到此时，src文件夹下出现了一些绿色的文件，这些文件就是我们以后需要用到的命令文件

**步骤三：移动文件**

移动文件，便于管理：(所有源代码安装的软件都安装在/usr/local下，如apache等)

创建两个文件夹，bin用于存放命令，etc拥有存放配置文件。

mkdir -p /usr/local/redis/etcmkdir -p /usr/local/redis/etc

-p是递归创建。

接下来，将redis-2.8.19文件夹下的redis.conf复制到/usr/local/redis/etc/

并将src目录下的7个命令文件（绿色的），移动到/usr/local/redis/bin/

[linuxidc@localhost src]$ cd ..

[linuxidc@localhost redis-2.8.19]$ ls00-RELEASENOTES COPYING Makefile redis.conf sentinel.conf utilsBUGS deps MANIFESTO runtest srcCONTRIBUTING INSTALL README runtest-sentinel tests

[linuxidc@localhost redis-2.8.19]$ mv ./redis.conf /usr/local/redis/etc/

[linuxidc@localhost redis-2.8.19]$ cd src

[linuxidc@localhost src]$ mv mkreleasehdr.sh redis-benchmark redis-check-aof redis-check-dump redis-cli redis-sentinel redis-server /usr/local/redis/bin/

**步骤四：启动Redis服务：**

首先进入刚才安装redis的目录：

[linuxidc@localhost src]$ cd /usr/local/redis/bin

[linuxidc@localhost bin]$ ls

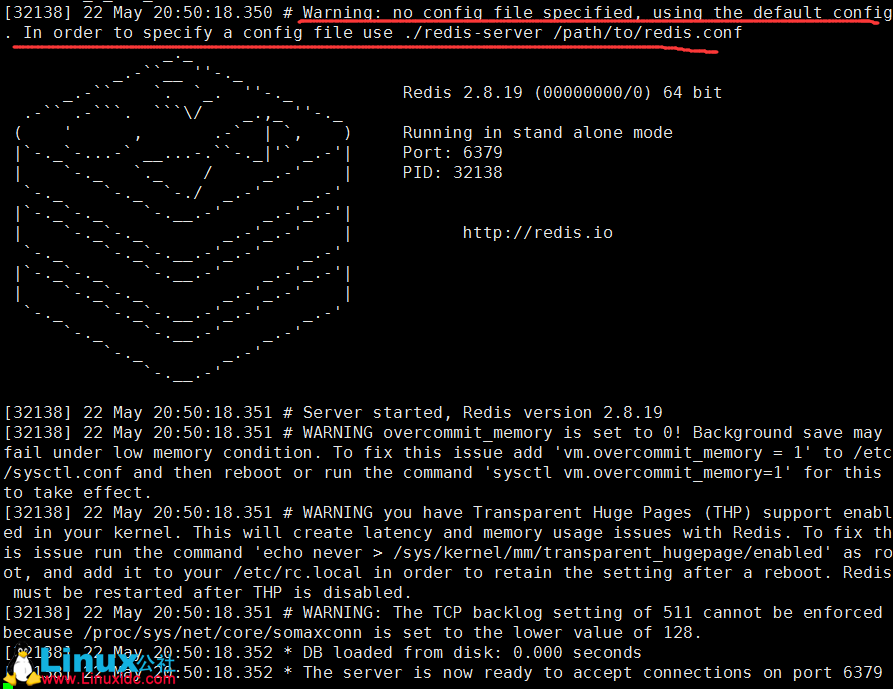
mkreleasehdr.sh redis-check-aof redis-cli redis-server

redis-benchmark redis-check-dump redis-sentinel

之后我们启动redis服务。启动redis服务需要用到命令redis-server

[linuxidc@localhost bin]$ ./redis-server

结果：



但是，这样做的话，我们并没有使用etc的下的配置文件进行启动（图中红线部分）。

如果希望通过指定的配置文件启动，需要在启动时指定配置文件：

这里我们先用ctrl+C来终止服务，然后查看redis服务是否终止干净了，之后通过设置配置文件来启动服务：

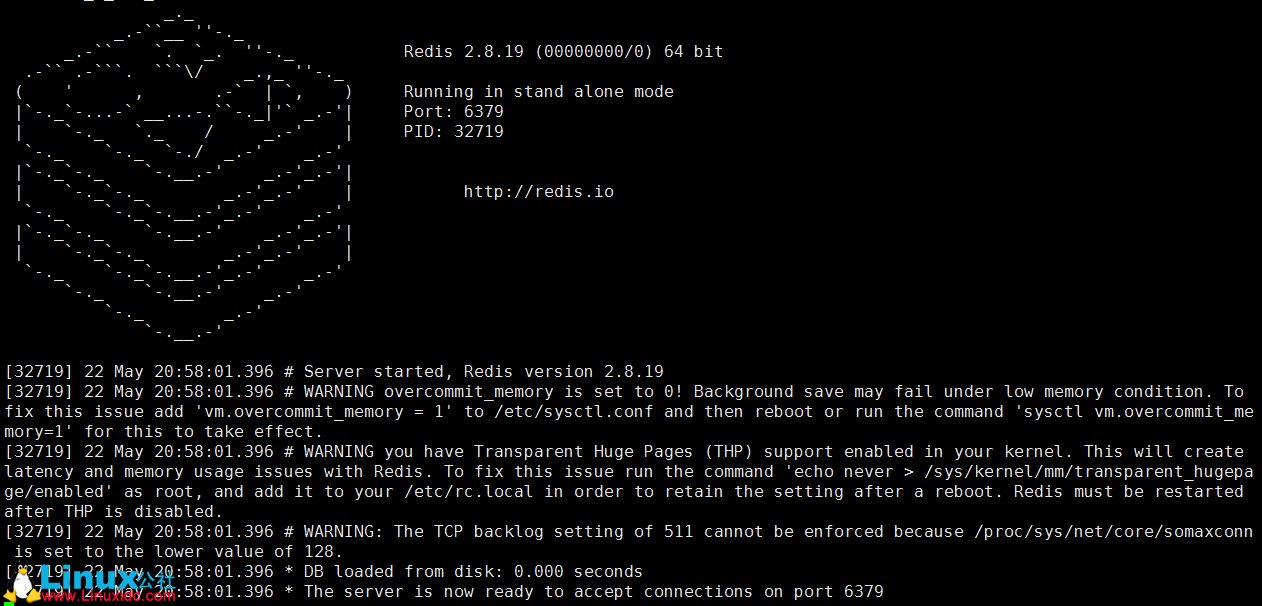
按下ctrl+c后(^C):

^C[32138 | signal handler] (1463921540) Received SIGINT scheduling shutdown...[32138] 22 May 20:52:20.380 # User requested shutdown...[32138] 22 May 20:52:20.380 \* Saving the final RDB snapshot before exiting.[32138] 22 May 20:52:20.393 \* DB saved on disk[32138] 22 May 20:52:20.393 # Redis is now ready to exit, bye bye...

运行：pstree -p | grep redis 发现redis服务已经被终止干净

现在我们带上配置文件 /usr/local/etc/redis.conf 运行redis

[linuxidc@localhost bin]$ ./redis-server /usr/local/redis/etc/redis.conf



但是，现在redis仍然是在前台运行。

如果要后台启动该怎么办呢？

在这里我们用—-直接将命令丢到后台中“执行”的 & ？可以吗？

答案是不可以的，Redis的后台启动并运行需要通过配置文件中的参数设置。如果需要在后台运行，把daemonize配置项改为yes

vim /usr/local/redis/etc/redis.conf

搜索 ：’\daemonize’   
把daemonize配置项改为yes   
保存退出

之后我们再次使用配置文件启动redis-server。

可以看到，redis是后台启动了，并且通过ps命令可以查看到redis正在运行。

[linuxidc@localhost bin]\$ ./redis-server /usr/local/redis/etc/redis.conf

[linuxidc@localhost bin]\$ ps -ef | grep redis

root 13154 1 0 22:53 ? 00:00:00 ./redis-server \*:6379

neil 13162 8143 0 22:54 pts/0 00:00:00 grep --color=auto redis

[linuxidc@localhost bin]$ pstree -p | grep redis

|-redis-server(13154)-+-{redis-server}(13156)

| `-{redis-server}(13157)

Redis服务端默认连接端口是6379.

就好比 MySQL 或 mariadb 服务端默认连接端口是3306

在平时，我们往往需要查看6379端口是否被占用。可以用以下命令：

netstat -tunpl | grep 6379

注意，redis服务需要 root 权限才能查看，不然���能检查到6379被某个进程占用，但是看不到进程名称。

至此，redis服务已经按照配置文件启动成功！！

**步骤五：客户端登录**

[linuxidc@localhost bin]$ /usr/local/redis/bin/redis-cli

回车后如果提示：

[linuxidc@localhost bin]$ /usr/local/redis/bin/redis-cli127.0.0.1:6379>//提示

则表明客户端登录成功

**步骤六：关闭Redis服务**

停止Redis实例

我们可以使用pkill redis-server

[linuxidc@localhost bin]$ pkill redis-server

[linuxidc@localhost bin]$ netstat -tunpl | grep 6379

[linuxidc@localhost bin]$

[linuxidc@localhost bin]$ pstree -p | grep redis

[linuxidc@localhost bin]$

[linuxidc@localhost bin]$

[linuxidc@localhost bin]$ /usr/local/redis/bin/redis-cli Could not connect to Redis at 127.0.0.1:6379: Connection refusednot connected> not connected> exit

关闭之后，发现6379就不再被占用了，redis的进程也都没有了。

客户登陆也无法成功了。

也可以使用/usr/local/redis/bin/redis-cli shutdown，这种方法使用客户端命令redis-cli 进行Redis服务的停止

[linuxidc@localhost bin]]$ ./redis-server /usr/local/redis/etc/redis.conf

[linuxidc@localhost bin]$ pstree -p | grep redis

|-redis-server(13509)-+-{redis-server}(13511)

| `-{redis-server}(13512)

[linuxidc@localhost bin]$ /usr/local/redis/bin/redis-cli shutdown

[linuxidc@localhost bin]$ pstree -p | grep redis

[linuxidc@localhost bin]$

[linuxidc@localhost bin]$ sudo netstat -tunpl | grep 6379

[linuxidc@localhost bin]$

当然关闭某个服务还可以使用 killall 和 kill -9.

**附录：更多的关于 /usr/local/redis/etc/redis.conf 的配置信息**

1、daemonize 如果需要在后台运行，把该项改为yes

2、pidfile 配置多个pid的地址 默认在/var/run/redis.pid

3、bind 绑定ip，设置后只接受来自该ip的请求

4、port 监听端口，默认是6379

5、loglevel 分为4个等级：debug verbose notice warning

6、logfile 用于配置log文件地址

7、databases 设置数据库个数，默认使用的数据库为0

8、save 设置redis进行数据库镜像的频率。

9、rdbcompression 在进行镜像备份时，是否进行压缩

10、dbfilename 镜像备份文件的文件名

11、Dir 数据库镜像备份的文件放置路径

12、Slaveof 设置数据库为其他数据库的从数据库

13、Masterauth 主数据库连接需要的密码验证

14、Requriepass 设置 登陆时需要使用密码

15、Maxclients 限制同时使用的客户数量

16、Maxmemory 设置redis能够使用的最大内存

17、Appendonly 开启append only模式

18、Appendfsync 设置对appendonly.aof文件同步的频率（对数据进行备份的第二种方式）

19、vm-enabled 是否开启虚拟内存支持 （vm开头的参数都是配置虚拟内存的）

20、vm-swap-file 设置虚拟内存的交换文件路径

21、vm-max-memory 设置redis使用的最大物理内存大小

22、vm-page-size 设置虚拟内存的页大小

23、vm-pages 设置交换文件的总的page数量

24、vm-max-threads 设置VM IO同时使用的线程数量

25、Glueoutputbuf 把小的输出缓存存放在一起

26、hash-max-zipmap-entries 设置hash的临界值

27、Activerehashing 重新hash