Memcached安装说明

## 安装

**1、需要的安装包**

libevent-2.0.22-stable.tar.gz

memcached-1.4.22.tar.gz

**2、安装步骤**

先安装libevent。这个东西在配置时需要指定一个安装路径，即./configure –prefix=/usr/local；然后make；然后make install；

再安装memcached，只是需要在配置时需要指定libevent的安装路径即./configure –with-libevent=/usr/local；然后make；然后make install；

**3、安装详情**

**先安装libevent：**  
# tar zxvf libevent-2.0.22-stable.tar.gz  
# cd libevent-2.0.22-stable  
# ./configure –prefix=/usr/local  
# make  
# make install

测试libevent是否安装成功：  
# ls -al /usr/local/lib | grep libevent  
  
每个版本显示的不一样，只要出现一些上面的信息就说明安装成功。

安装memcached，同时需要安装中指定libevent的安装位置：   
# tar zxvf memcached-1.4.22.tar.gz  
# cd memcached-1.4.22  
# ./configure –with-libevent=/usr/local  
# make  
# make install  
如果中间出现报错，请仔细检查错误信息，按照错误信息来配置或者增加相应的库或者路径。  
安装完成后会把memcached放到 /usr/local/bin/memcached ，

测试是否成功安装memcached：  
# ls -al /usr/local/bin/mem\*  


**4、启动memcached**

启动命令

# /usr/local/bin/memcached -d -m 1024 -u root -l 10.1.21.235 -p 10000 -c 4096 -P /tmp/memcached.pid

-d选项是启动一个守护进程，  
-m是分配给Memcache使用的内存数量，单位是MB，我这里是10MB，  
-u是运行Memcache的用户，我这里是root，  
-l是监听的服务器IP地址，如果有多个地址的话，我这里指定了服务器的IP地址10.1.21.235，  
-p是设置Memcache监听的端口，我这里设置了10000，最好是1024以上的端口，  
-c选项是最大运行的并发连接数，默认是1024，我这里设置了256，按照你服务器的负载量来设定，  
-P是设置保存Memcache的pid文件，我这里是保存在 /tmp/memcached.pid，

如果要结束Memcache进程，执行：

# kill `cat /tmp/memcached.pid`

也可以启动多个守护进程，不过端口不能重复。

**5、测试memcached**

[root@localhost /]# telnet 10.1.21.235 10000

Trying 10.1.21.235...

Connected to 10.1.21.235 (10.1.21.235).

Escape character is '^]'.

set key1 0 60 4

zhou

STORED

get key1

VALUE key1 0 4

zhou

END

**6、常见问题:**

**1.如果启动Memcached服务的时候遇到了**

/usr/local/bin/memcached: error while loading shared libraries: libevent-1.2.so.1: cannot open shared object file: No such file or directory;

解决方案:

[root@localhost bin]# LD\_DEBUG=libs memcached -v   
[root@localhost bin]# ln -s /usr/lib/libevent-1.2.so.1 /usr/lib64/libevent-1.2.so.1  
[root@localhost bin]# /usr/local/bin/memcached -d -m 100 -u root -p 12000 -c 1000 -P /tmp/memcached.pid  
[root@localhost bin]# ps -aux

可以看到启动的Memcached服务了.

**2.把Memcached服务加载到Linux的启动项中.万一机器断电系统重启.那么Memcached就会自动启动了.**

假如启动Memcache的服务器端的命令为：  
# /usr/local/bin/memcached -d -m 10 -u root -l 10.1.21.235 -p 10000 -c 256 -P /tmp/memcached.pid容来自17jquery

想开机自动启动的话，只需在/etc/rc.d/rc.local中加入一行，下面命令  
/usr/local/memcached/bin/memcached -d -m 10 -p 10000 -u apache -c 256   
上面有些东西可以参考一下：即，ip不指定时，默认是本机，用户:最好选择是：apache 或 deamon  
这样，也就是属于哪个用户的服务，由哪个用户启动

## Memcached状态查看说明

查看memcached运行状态命令

[root@localhost /]# telnet 10.1.21.235 10000

Trying 10.1.21.235...

Connected to 10.1.21.235 (10.1.21.235).

Escape character is '^]'.

stats

STAT pid 11243

STAT uptime 87353

STAT time 1425523156

STAT version 1.4.22

STAT libevent 2.0.22-stable

STAT pointer\_size 64

STAT rusage\_user 0.016997

STAT rusage\_system 0.028995

STAT curr\_connections 5

STAT total\_connections 17

STAT connection\_structures 16

STAT reserved\_fds 20

STAT cmd\_get 18

STAT cmd\_set 5

STAT cmd\_flush 0

STAT cmd\_touch 0

STAT get\_hits 8

STAT get\_misses 10

STAT delete\_misses 0

STAT delete\_hits 0

STAT incr\_misses 0

STAT incr\_hits 0

STAT decr\_misses 0

STAT decr\_hits 0

STAT cas\_misses 0

STAT cas\_hits 0

STAT cas\_badval 0

STAT touch\_hits 0

STAT touch\_misses 0

STAT auth\_cmds 0

STAT auth\_errors 0

STAT bytes\_read 2655

STAT bytes\_written 3108

STAT limit\_maxbytes 2147483648

STAT accepting\_conns 1

STAT listen\_disabled\_num 0

STAT threads 4

STAT conn\_yields 0

STAT hash\_power\_level 16

STAT hash\_bytes 524288

STAT hash\_is\_expanding 0

STAT malloc\_fails 0

STAT bytes 0

STAT curr\_items 0

STAT total\_items 2

STAT expired\_unfetched 0

STAT evicted\_unfetched 0

STAT evictions 0

STAT reclaimed 0

STAT crawler\_reclaimed 0

STAT lrutail\_reflocked 0

|  |  |
| --- | --- |
| pid | 进程id |
| uptime | 总的运行时间，秒数 |
| time | 当前时间 |
| version | 版本号 |
| libevent | libevent版本号 |
| pointer\_size | 当前操作[系统](http://www.2cto.com/os/)的指针大小（32位系统一般是32bit,64就是64位操作系统） |
| rusage\_user | 进程的累计用户时间 |
| rusage\_system | 进程的累计系统时间 |
| curr\_connections | 当前连接数 |
| total\_connections | 从服务器启动以后的总共连接数 |
| connection\_structures | 服务器分配的连接构造数 |
| reserved\_fds |  |
| cmd\_get | 总获取次数 |
| cmd\_set | 总的写入次数 |
| cmd\_flush | flush命令请求次数 |
| cmd\_touch | touch命令请求次数 |
| get\_hits | 总的命中次数 |
| get\_misses | 获取失败次数 |
| delete\_misses | delete命令未命中次数 |
| delete\_hits | delete命令命中次数 |
| incr\_misses | incr命令未命中次数 |
| incr\_hits | incr命令命中次数 |
| decr\_misses | decr命令未命中次数 |
| decr\_hits | decr命令命中次数 |
| cas\_misses | cas命令未命中次数 |
| cas\_hits | cas命令命中次数 |
| cas\_badval | cas使用擦拭次数 |
| touch\_hits | touch命令命中次数 |
| touch\_misses | touch命令未命中次数 |
| auth\_cmds | 认证命令处理的次数 |
| auth\_errors | 认证失败数目 |
| bytes\_read | 总共读取的流量字节数 |
| bytes\_written | 总的写入流量字节 |
| limit\_maxbytes | 最大允许使用的内存量，字节 |
| accepting\_conns |  |
| listen\_disabled\_num | 失效的监听数 |
| threads | 当前线程数 |
| conn\_yields | 连接操作主动放弃数目 |
| hash\_power\_level |  |
| hash\_bytes |  |
| hash\_is\_expanding |  |
| malloc\_fails |  |
| bytes | 所有的缓存使用的内存量 |
| curr\_items | 当前缓存中的KeyValue数量 |
| total\_items | 曾经总共经过缓存的KeyValue数量 |
| expired\_unfetched |  |
| evicted\_unfetched |  |
| evictions | 为获取空闲内存而删除的items数（分配给memcache的空间用满后需要删除旧的items来得到空间分配给新的items) |
| reclaimed | 已过期的数据条目来存储新数据的数目 |
| crawler\_reclaimed |  |
| lrutail\_reflocked |  |

安装magent（[magent-0.6](https://storage.googleapis.com/google-code-archive-downloads/v2/code.google.com/memagent/magent-0.6.tar.gz)有问题，建议使用[magent-0.5](https://storage.googleapis.com/google-code-archive-downloads/v2/code.google.com/memagent/magent-0.6.tar.gz)）

https://code.google.com/archive/p/memagent/downloads

1. mkdir magent
2. cd ./magent
3. wget http://memagent.googlecode.com/files/magent-0.5.tar.gz
4. tar zxvf magent-0.5.tar.gz
5. /sbin/ldconfig
6. sed -i "s#LIBS = -levent#LIBS = -levent -lm#g" Makefile
7. make
8. cp magent /usr/bin/magent

在make的时候可能会报undefined reference to `clock\_gettime'错误，此时可如下修改Makefile文件（添加-lrt参数）：

magent: $(STPROG)

$(CC) $(CFLAGS) -o $@ $^ $(LIBS) -lrt

启动：

./magent -u root -n 51200 -p 10000 -s 127.0.0.1:11211 -b 127.0.0.1:11212

./magent -u root -n 51200 -p 11000 -s 127.0.0.1:11212 -b 127.0.0.1:11211