codis 3.1 安装搭建

作者：夏末终年
Codis交流群：343595434

一.基本信息

1. 服务器基本信息

ip地址	安装服务
172.16.200.71	zk1、codis-dashboard、codis-fe、codis-ha、 codis-proxy1、group1_M(6379)、group2_S(6380)
172.16.200.72	zk2、codis-proxy2、group2_M(6379)、group3_S(6380)
172.16.200.73	zk3、codis-proxy3、group3_M(6379)、group1_S(6380)

参考文档： https://github.com/CodisLabs/codis/blob/release3.1/doc/tutorial_zh.md

2. 环境信息

2.1 codis版本：3.1 版本

```
git clone https://github.com/CodisLabs/codis.git -b release3.1
```

2.2 Go使用版本：go1.5.2.Linux-amd64.tar.gz

```
https://golang.org/doc/install?download=go1.5.2.linux-amd64.tar.gz
```

2.3 jdk版本：jdk1.8.0_11

```
http://download.oracle.com/otn-pub/java/jdk/8u111-b14/jdk-8u111-linux-x64.tar.gz
```

2.4 zookeeper版本：zookeeper-3.4.8.tar.gz

```
http://mirrors.cnnic.cn/apache/zookeeper/zookeeper-3.4.6/zookeeper-3.4.6.tar.gz
```

二.部署codis

1. 相关组件安装配置

1.1 安装所需依赖包

三台主机上执行

```
yum install -y gcc make gcc-c++ automake lrzsz openssl-devel zlib-* bzip2-* readline* zlib-* bzip2-* git nmap unzip wget lsof xz net-tools mercurial vim
```

1.2 修改内核

三台主机上执行

```
vi /etc/sysctl.conf
vm.overcommit_memory = 1
sysctl vm.overcommit_memory=1
```

手工执行：

```
echo never > /sys/kernel/mm/transparent_hugepage/enabled
```

并加到 /etc/rc.local中

2.安装go(三台主机上执行）

2.1 下载go

```
cd /usr/local/src
wget https://golang.org/doc/install?download=go1.5.2.linux-amd64.tar.gz
```

2.2 解包

```
cd /usr/local/src
tar -C /usr/local -xzf go1.5.2.linux-amd64.tar.gz
```

2.3 新建gopath

```
mkdir /usr/local/gopkg
```

2.4 配置go环境变量

```
vim /etc/profile
```

添加如下信息

```
export GOROOT=/usr/local/go
export GOPATH=/usr/local/gopkg
export PATH=$GOROOT/bin:$PATH
```

刷新配置文件：

```
source /etc/profile
```

查看go版本：

```
[root@codis01 ~]#go version
go version go1.5.2 linux/amd64
```

3.安装jdk(三台主机上执行）

```
cd /usr/loca/src/
tar -C /usr/local/ -xzf /usr/local/src/jdk-8u111-linux-x64.tar.gz
```

配置java环境变量

```
vim /etc/profile
```

添加如下信息

```
export JAVA_HOME=/usr/local/jdk1.8.0_111
export PATH=$JAVA_HOME/bin:$PATH
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
```

刷新配置文件：

```
source /etc/profile
```

查看go版本：

```
[root@codis01 ~]#java -version
java version "1.8.0_111"
Java(TM) SE Runtime Environment (build 1.8.0_111-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.111-b14, mixed mode)
```

4.安装zookeeper(三台主机上执行)

4.1 安装zookeeper

```
cd /usr/local/src/
tar -C /usr/local/ -xzf zookeeper-3.4.8.tar.gz
cd /usr/local/zookeeper-3.4.8
ln -s zookeeper-3.4.8 zookeeper
```

4.2 生成配置文件

```
cd /usr/local/zookeeper
```

```
cp conf/zoo_sample.cfg conf/zoo.cfg
```

4.3 修改zookeeper配置文件

```
vim /usr/local/zookeeper/conf/zoo.cfg
```

修改以下内容

```
maxClientCnxns=60
tickTime=2000
initLimit=10
syncLimit=5
dataDir=/usr/local/zookeeper/data
dataLogDir=/data/logs/zookeeper
clientPort=2181
server.1=172.15.200.71:2888:3888
server.2=172.15.200.72:2888:3888
server.3=172.15.200.73:2888:3888
2888表示zookeeper程序监听端口，3888表示zookeeper选举通信端口。
```

4.4 创建所需文件夹

```
mkdir -p /usr/local/zookeeper/data
mkdir -p /data/logs/zookeeper
```

4.5 生成myid

主机(172.16.200.71)

```
echo "1" >/usr/local/zookeeper/data/myid  ##生成ID，这里需要注意， myid对应的zoo
.cfg的server.ID，比如第二台zookeeper主机对应的myid应该是2
```

主机(172.16.200.72)

```
echo "2" >/usr/local/zookeeper/data/myid
```

主机(172.16.200.73)

```
echo "3" >/usr/local/zookeeper/data/myid
```

4.6 启动zookeeper

```
cd /usr/local/zookeeper/bin
./zkServer.sh start#
```

4.7 关闭zookeeper

```
cd /usr/local/zookeeper/bin
./zkServer.sh stot
```

4.8 查看zk状态

```
cd /usr/local/zookeeper/bin
./zkServer.sh status
```

4.9 查看相关信息：

```
/usr/local/zookeeper/bin/zkCli.sh -server 127.0.0.1:2181
```

5.安装codis(三台主机上执行)

5.1 下载codis

```
mkdir -p /usr/local/gopkg/src/github.com/CodisLabs/
cd /usr/local/gopkg/src/github.com/CodisLabs/
git clone https://github.com/CodisLabs/codis.git -b release3.1
```

5.2 编译codis

```
cd /usr/local/gopkg/src/github.com/CodisLabs/codis
```

```
make
```

5.3 查看bin

```
[root@codis01 codis]# ll bin/
总用量 82728
drwxr-xr-x 4 root root      111 12月 18 21:54 assets
-rwxr-xr-x 1 root root 18261200 12月 18 21:54 codis-admin
-rwxr-xr-x 1 root root 19101304 12月 18 21:54 codis-dashboard
-rwxr-xr-x 1 root root 17655424 12月 18 21:54 codis-fe
-rwxr-xr-x 1 root root 10032096 12月 18 21:54 codis-ha
-rwxr-xr-x 1 root root 11202080 12月 18 21:54 codis-proxy
-rwxr-xr-x 1 root root  4167892 12月 18 21:54 codis-server
-rwxr-xr-x 1 root root  2073794 12月 18 21:54 redis-benchmark
-rwxr-xr-x 1 root root  2197701 12月 18 21:54 redis-cli
-rw-r--r-- 1 root root      148 12月 18 21:54 version
[root@zt-redis01 codis]#
```

执行全部指令后，会在 bin 文件夹内生成 codis-admin、codis-dashboard、codis-fe、codis-ha、codis-proxy、codis-server 六个可执行文件。
另外, bin/assets 文件夹是 codis-dashboard http 服务需要的前端资源, 需要和codis-dashboard 放置在同一文件夹下

5.4 创建codis所需目录

```
mkdir -p /usr/local/codis
mkdir -p /data/logs/codis
mkdir -p /data/codis/redis_conf
mkdir -p /data/components/redis
```

5.5 拷贝codis的bin目录

```
cp -r /usr/local/gopkg/src/github.com/CodisLabs/codis/bin  /usr/local/codis/
```

由于codis 本身只有codis-server ， 没有Redis-cli，需要把redis 2.8.21 安装包里面的redis-cli copy到/usr/local/codis/bin 下面:

```
cd /usr/local/gopkg/src/github.com/CodisLabs/codis/extern/redis-2.8.21/src
cp redis-cli /usr/local/codis/bin
```

5.6 配置和启动各组件

配置和启动 Codis的Redis
配置文件 : /usr/local/codis/redis_conf/redis6379.conf
设置密码: xxxxx
考虑性能，主库关闭aof和rdp，从库只开启aof:

```
cd /usr/local/gopkg/src/github.com/CodisLabs/codis/extern/redis-2.8.21/
cp redis.conf /usr/local/codis/conf/redis6379.conf
cp redis.conf /usr/local/codis/conf/redis6380.conf
```

> 注：以下配置文件仅供参考

将redis6379.conf更改(主库)：

```
daemonize yes
pidfile /usr/locai/codis/run/redis6379.pid
port 6379
timeout 86400
tcp-keepalive 60
loglevel notice
logfile /data/logs/codis/redis6379.log
databases 16
save ""
save 900 1
save 300 10
save 60 10000
stop-writes-on-bgsave-error no
rdbcompression yes
dbfilename dump6379.rdb
dir /data/codis/redis_data_6379
masterauth "xxxxx"
slave-serve-stale-data yes
repl-disable-tcp-nodelay no
slave-priority 100
requirepass "xxxxx"
maxmemory 10gb
maxmemory-policy allkeys-lru
appendonly no
appendfsync everysec
no-appendfsync-on-rewrite yes
auto-aof-rewrite-percentage 100
auto-aof-rewrite-min-size 64mb
```

```
lua-time-limit 5000
slowlog-log-slower-than 10000
slowlog-max-len 128
hash-max-ziplist-entries 512
hash-max-ziplist-value 64
list-max-ziplist-entries 512
list-max-ziplist-value 64
set-max-intset-entries 512
zset-max-ziplist-entries 128
zset-max-ziplist-value 64
client-output-buffer-limit normal 0 0 0
client-output-buffer-limit slave 0 0 0
client-output-buffer-limit pubsub 0 0 0
hz 10
aof-rewrite-incremental-fsync yes
repl-backlog-size 33554432
```

将redis6380.conf更改(从库)：

```
daemonize yes
pidfile /usr/local/codis/run/redis6380.pid
port 6380
timeout 86400
tcp-keepalive 60
loglevel notice
logfile /data/logs/codis/redis6380.log
databases 16
save ""
save 900 1
save 300 10
save 60 10000
stop-writes-on-bgsave-error no
rdbcompression yes
dbfilename dump6379.rdb
dir /data/codis/redis_data_6380
masterauth "xxxxx"
slave-serve-stale-data yes
repl-disable-tcp-nodelay no
slave-priority 100
requirepass "xxxxx"
maxmemory 10gb
maxmemory-policy allkeys-lru
appendonly no
appendfsync everysec
no-appendfsync-on-rewrite yes
auto-aof-rewrite-percentage 100
auto-aof-rewrite-min-size 64mb
```

```
lua-time-limit 5000
slowlog-log-slower-than 10000
slowlog-max-len 128
hash-max-ziplist-entries 512
hash-max-ziplist-value 64
list-max-ziplist-entries 512
list-max-ziplist-value 64
set-max-intset-entries 512
zset-max-ziplist-entries 128
zset-max-ziplist-value 64
client-output-buffer-limit normal 0 0 0
client-output-buffer-limit slave 0 0 0
client-output-buffer-limit pubsub 0 0 0
hz 10
aof-rewrite-incremental-fsync yes
repl-backlog-size 33554432
```

5.7 启动redis

```
/usr/local/codis/bin/codis-server  /usr/local/codis/redis_conf/redis6379.con
f &
/usr/local/codis/bin/codis-server  /usr/local/codis/redis_conf/redis6380.con
f &
```

三. 配置codis

1. Codis Dashboard

Coddis3.0的dashboard与codis 2.0有所不同，作为集群管理工具，它支持codis-proxy,codis-server的添加、删除以及数据迁移等操作。在集群状态发生改变时，codis-dashboard 维护集群下所有 codis-proxy 的状态一致性。有以下两点注意事项：
* 对于同一个业务集群而言，同一个时刻codis-dashboard只能有0个或者1个；
* 所有对集群的修改都必须通过codis-dashboard完成。

1.1 配置Codis Dashboard

默认配置文件dashboard.toml可由codis-dashboard生成。

```
#/usr/local/codis/bin/codis-dashboard  --default-config | tee  dashboard.tom
l（就是dashboard.conf）
```

生成dashboard.toml文件，可自行配置。

```
# Set Coordinator, only accept"zookeeper"&"etcd"
coordinator_name = "zookeeper"
coordinator_addr = "172.16.200.71:2181,172.16.200.72:2181,172.16.200.73:2181
"        #zookeeper是集群的话就写多个ip和端口用逗号隔开
# Set Codis Product {Name/Auth}
product_name = "codis-demo"
product_auth = ""
# Set bind address for admin(rpc), tcp only.
admin_addr = "0.0.0.0:18080"
```

参数说明：

```
coordinator_name               外部存储类型，接受 zookeeper/etcd
coordinator_addr                 外部存储地址
product_name                     集群名称，满足正则 \w[\w\.\-]*
product_auth                       集群密码，默认为空
admin_addr                         RESTful API 端口
```

1.2 启动Codis Dashboard

```
nohup   /usr/local/codis/bin/codis-dashboard   --ncpu=4    --config=dashboar
d.toml（这里指定dashboard.conf也可以）    --log=dashboard.log      --log-level=W
ARN  &
```

```
#bin/codis-dashboard   -h
Usage:
codis-dashboard [--ncpu=N][--config=CONF] [--log=FILE] [--log-level=LEVEL] [
--host-admin=ADDR]
codis-dashboard  --default-config
codis-dashboard  --version
```

参数说明：

```
 --ncpu=N                                  最大使用 CPU 个数
```

```
 -c  CONF, --config=CONF       指定启动配置文件
 -l   FILE, --log=FILE                  设置 log 输出文件
--log-level=LEVEL              设置log输出等级：INFO,WARN,DEBUG,ERROR；默认INFO,
推荐WARN
正常关闭dashboard命令：      bin/codis-admin   --dashboard=172.16.200.71:18080
  --shutdown
```

2. Codis Proxy

> 对于同一个业务集群而言，可以同时部署多个codis-proxy 实例；
> 不同 codis-proxy 之间由 codis-dashboard 保证状态同步。

2.1 配置proxy

默认配置文件 proxy.toml 可由 codis-proxy 生成。

```
/usr/local/codis/bin/codis-proxy --default-config | tee proxy.toml (proxy.co
nf)
```

生成proxy.toml可自行配置。

```
# Set Codis Product {Name/Auth}.
product_name = "codis-demo"
product_auth = ""
# Set bind address for admin(rpc), tcp only.
admin_addr = "0.0.0.0:11080"
# Set bind address for proxy, proto_type can be "tcp","tcp4", "tcp6", "unix"
 or "unixpacket".
proto_type = "tcp4"
proxy_addr = "0.0.0.0:19000"
 # Set jodis address & session timeout.
jodis_addr = ""
jodis_timeout = 10
# Proxy will ping-pong backend redis periodly to keep-alive
backend_ping_period = 5
# If there is no request from client for a long time, the connectionwill be
droped. Set 0 to disable.
session_max_timeout = 1800
# Buffer size for each client connection.
session_max_bufsize = 131072
# Number of buffered requests for each client connection.
# Make sure this is higher than the max number of requests for eachpipeline
request, or your client may be blocked.
```

```
session_max_pipeline = 1024
# Set period between keep alives. Set 0 to disable.
session_keepalive_period = 60
```

参数说明：

```
product_name 集群名称，参考dashboard参数说明
product_auth 集群密码，默认为空
admin_addr RESTfulAPI 端口
proto_type Redis 端口类型，接受tcp/tcp4/tcp6/unix/unixpacket
proxy_addr Redis 端口地址或者路径
jodis_addr Jodis注册zookeeper地址
jodis_timeout Jodis注册sessiontimeout时间，单位second
jodis_compatible Jodis注册 zookeeper 的路径
backend_ping_period 与codis-server 探活周期，单位second，0表示禁止
session_max_timeout 与client 连接最大读超时，单位second，0表示禁止
session_max_bufsize 与client 连接读写缓冲区大小，单位byte
session_max_pipeline 与client 连接最大的pipeline大小
session_keepalive_period 与client 的 tcp keepalive周期，仅tcp有效，0表示禁止
```

2.2 启动proxy

```
nohup /usr/local/codis/bin/codis-proxy --ncpu=4 --config=proxy.toml \
--log=proxy.log --log-level=WARN &
```

codis-proxy启动后，处于 waiting 状态，监听proxy_addr 地址，但是不会accept连接。添加到集群并完成集群状态的同步，才能改变状态为online。添加的方法有以下两种：

> 通过codis-fe添加：通过Add Proxy按钮，将admin_addr加入到集群中；
> 通过codis-admin命令行工具添加，方法如下：

最好采用通过codis-fe添加

```
/usr/local/codis/bin/codis-admin --dashboard=172.16.200.71:18080 --create-pr
oxy -x 172.16.200.71:11080
```

其中172.16.200.71:18080 以及172.16.200.71:11080 分别为dashboard和proxy的admin_addr地址。
添加过程中，dashboard会完成如下一系列动作：
① 获取 proxy 信息，对集群name以及auth进行验证，并将其信息写入到外部存储中；

② 同步slots状态；
③ 标记proxy状态为online，此后proxy开始accept连接并开始提供服务。

正常关闭proxy操作

```
/usr/local/codis/bin/codis-admin --proxy=172.16.200.71:11080 --auth="xxxxx"(有就加，没有就不加) --shutdown
```

```
/usr/local/codis/bin/codis-proxy -h
Usage:
codis-proxy [--ncpu=N][--config=CONF] [--log=FILE] [--log-level=LEVEL] [--host-admin=ADDR]
[--host-proxy=ADDR] [--ulimit=NLIMIT]
codis-proxy --default-config
codis-proxy --version
 Options:
    --ncpu=N 最大使用 CPU 个数
    -c CONF, --config=CONF 指定启动配置文件
    -l FILE, --log=FILE 设置 log 输出文件
    --log-level=LEVEL 设置 log 输出等级：INFO,WARN,DEBUG,ERROR；默认INFO，推荐WARN
    --ulimit=NLIMIT 检查ulimit -n的结果，确保运行时最大文件描述不少于NLIMIT
```

3. Codis FE

3.1 集群管理界面

> 多个集群实例可以共享同一个前端展示页面；
> 通过配置文件管理后端codis-dashboard列表，配置文件可自动更新。

3.2 配置codis-fe

配置文件codis.json（fe.conf）可以手动编辑，也可以通过codis-admin从外部存储中拉取。

```
/usr/local/codis/bin/codis-admin --dashboard-list --zookeeper=172.16.200.71:2181 | tee codis.json
```

```
[
```

```
    {
"name": "codis-demo",
"dashboard": "127.0.0.1:18080"
    }
]
```

3.3 启动codis-fe

```
nohup /usr/local/codis/bin/codis-fe --ncpu=4 --log=fe.log --log-level=WARN \
--dashboard-list=codis.json --listen=0.0.0.0:18090 &#(这里指定端口号为18090是为了防止和codis-dashboard的端口号18080冲突)
```

```
/usr/local/codis/bin/codis-fe -h
Usage:
    codis-fe [--ncpu=N][--log=FILE] [--log-level=LEVEL] --dashboard-list=LIST --listen=ADDR
    codis-fe --version
Options:
--ncpu=N 最大使用 CPU 个数
-d LIST,--dashboard-list=LIST 配置文件，能够自动刷新
-l FILE, --log=FILE 设置 log 输出文件
--log-level=LEVEL 设置 log 输出等级：INFO,WARN,DEBUG,ERROR；默认INFO，推荐WARN
--listen=ADDR HTTP 服务端口
```

打开浏览器，在地址栏里输入http://172.16.200.71:18090，通过管理界面操作Codis。