# codis AIO

2017年4月20日
14:04

## codis3.2.8 AIO 单机环境部署 （by：一苇）

目录

# 零：部署环境

## 部署组件：

go
java
zookeeper
codis

Codis 3.x 由以下组件组成：

* **Codis Server**：基于 redis-3.2.8 分支开发。增加了额外的数据结构，以支持 slot 有关的操作以及数据迁移指令。具体的修改可以参考文档 [redis 的修改](redis_change_zh.md)。

* **Codis Proxy**：客户端连接的 Redis 代理服务, 实现了 Redis 协议。 除部分命令不支持以外([不支持的命令列表](unsupported_cmds.md))，表现的和原生的 Redis 没有区别（就像 Twemproxy）。

    + 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。

* **Codis Dashboard**：集群管理工具，支持 codis-proxy、codis-server 的添加、删除，以及据迁移等操作。在集群状态发生改变时，codis-dashboard 维护集群下所有 codis-proxy 的状态的一致性。

    + 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。

* **Codis Admin**：集群管理的命令行工具。

    + 可用于控制 codis-proxy、codis-dashboard 状态以及访问外部存储。

* **Codis FE**：集群管理界面。

    + 多个集群实例共享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。

* **Storage**：为集群状态提供外部存储。

    + 提供 Namespace 概念，不同集群的会按照不同 product name 进行组织；
    + 目前仅提供了 Zookeeper、Etcd、Fs 三种实现，但是提供了抽象的 interface 可自行扩展。

### 软件环境：

本机 IP：10.0.5.140
系统：centos6.8
软件：codis-release3.2.zip  go1.8.linux-amd64.tar.gz  zookeeper-3.4.10.tar.gz  java-1.8.0-openjdk

软件下载地址：
codis： https://github.com/CodisLabs/codis
zookeeper： https://zookeeper.apache.org/
go： http://golangtc.com/download

## 一. 部署 zookeeper 集群

### 1.1 安装 java 环境

```
# yum install java-1.8.0-openjdk-devel
```

配置 JAVA_HOME

```
# vim /etc/profile.d/java.sh
export JAVA_HOME=/usr

# source /etc/profile.d/java.sh

# java -version
openjdk version "1.8.0_91"
OpenJDK Runtime Environment (build 1.8.0_91-b14)
OpenJDK 64-Bit Server VM (build 25.91-b14, mixed mode)
```

### 1.2. 搭建 zookeeper 集群

```
# mkdir /{app,appdata}    //软件安装目录和数据目录
# cd /app
# mkdir -pv ./{zk1,zk2,zk3}/{data,log}    //zookeeper 安装目录，zookeeper 数据和日志目录
# tar -xf ./zookeeper-3.4.10.tar.gz -C ./zk1/
# tar -xf ./zookeeper-3.4.10.tar.gz -C ./zk2/
# tar -xf ./zookeeper-3.4.10.tar.gz -C ./zk3/
# ln -sv ./zk1/zookeeper-3.4.10 ./zookeeper
# ln -sv ./zk2/zookeeper-3.4.10 ./zookeeper
# ln -sv ./zk3/zookeeper-3.4.10 ./zookeeper
```

配置 zk1：

```
# cd /app
# cp /app/zk1/zookeeper/conf/zoo_sample.cfg /app/zk1/zookeeper/conf/zoo.cfg

# grep "^[^#].*" ./zoo.cfg
tickTime=2000
initLimit=10
syncLimit=5
dataDir=/app/zk1/data
#dataLogDir=/app/zk1/log    \\可以不写
clientPort=2181
server.1=10.0.5.140:2881:3888
server.2=10.0.5.140:2882:3888
server.3=10.0.5.140:2883:3888
```

生成 myid

```
# echo "1" > /app/zk1/data/myid
```

配置 zk2、zk3：
示例配置 zk3，zk2 类似：

```
# cd /app
# cp ./zk1/zookeeper/conf/zoo.cfg ./zk3/zookeeper/conf/
# sed -i 's/zk1/zk3/g' ./zk3/zookeeper/conf/zoo.cfg
# grep "^[^#].*" ./zk3/zookeeper/conf/zoo.cfg
tickTime=2000
initLimit=10
syncLimit=5
dataDir=/app/zk3/data
#dataLogDir=/app/zk3/log
```

```
clientPort=2183
server.1=10.0.5.140:2881:3888
server.2=10.0.5.140:2882:3888
server.3=10.0.5.140:2883:3888
# echo "3" > /app/zk3/data/myid
```

启动、关闭、查看 zookeeper 服务：

```
# /app/zk1/zookeeper/bin /zkServer.sh start|stop|status|restart
```

开机自启动 zookeeper 服务：

```
# grep "^/app.*" /etc/rc.local
/app/zk1/zookeeper/bin/zkServer.sh start
/app/zk2/zookeeper/bin/zkServer.sh start
/app/zk3/zookeeper/bin/zkServer.sh start
```

zookeeper 集群状态：

```
[root codies140 07:54:21] /app
-- # ./zk3/zookeeper/bin/zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /app/zk3/zookeeper/bin/../conf/zoo.cfg
Mode: follower
[root codies140 07:54:50] /app
-- # ./zk1/zookeeper/bin/zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /app/zk1/zookeeper/bin/../conf/zoo.cfg
Mode: follower
[root codies140 07:54:55] /app
-- # ./zk2/zookeeper/bin/zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /app/zk2/zookeeper/bin/../conf/zoo.cfg
Mode: leader
```

zookeeper 客户端连接：

```
# /app/zk1/zookeeper/bin/zkCli.sh  -server 10.0.5.140:2181
Connecting to 10.0.5.140:2181
```

```
2017-04-20 09:38:48,858 [myid:] - INFO  [main:Environment@100] - Client environment:zookeeper.version=3.4.10-
39d3a4f269333c922ed3db283be479f9deacaa0f, built on 03/23/2017 10:13 GMT
...
complete on server 10.0.5.140/10.0.5.140:2181, sessionid = 0x15b88a1c4070002, negotiated timeout = 30000

WATCHER::

WatchedEvent state:SyncConnected type:None path:null
[zk: 10.0.5.140:2181(CONNECTED) 0] ls /
[zookeeper]
[zk: 10.0.5.140:2181(CONNECTED) 1]
[zk: 10.0.5.140:2181(CONNECTED) 1] help
ZooKeeper -server host:port cmd args
	stat path [watch]
	set path data [version]
	ls path [watch]
	delquota [-n|-b] path
	ls2 path [watch]
	setAcl path acl
	setquota -n|-b val path
	history
	redo cmdno
	printwatches on|off
	delete path [version]
	sync path
	listquota path
	rmr path
	get path [watch]
	create [-s] [-e] path data acl
	addauth scheme auth
	quit
	getAcl path
	close
	connect host:port
```

参考：

https://zookeeper.apache.org/doc/r3.4.10/zookeeperStarted.html
http://blackproof.iteye.com/blog/2039040

## 二：部署 codis-server（redis-server）

### 2.1. go 环境部署

```
# mkdir /{app,appdata}
# cd /app
# tar -xf ./go1.8.linux-amd64.tar.gz
# mkdir ./gopkg
```

配置 GOROOT、GOPATH：

```
# vim /etc/profile.d/go.sh
export GOROOT=/app/go
export GOPATH=/app/gopkg
export PATH=$PATH:$GOROOT/bin

# source /etc/profile.d/go.sh
```

查看 go 版本：

```
# go version
go version go1.8 linux/amd64
```

### 2.2. codis-server 部署

#### *2.2.1. codis-server 编译*

配置编译环境：

```
# yum install -y gcc make gcc-c++ automake lrzsz openssl-devel zlib-* bzip2-* readline* git nmap unzip wget lsof xz net-tools mercurial vim  //具体软件按实际需求
```

创建 codis 编译目录：

```
# mkdir -pv /app/gopkg/src/github.com/CodisLabs/
```

下载 codis 软件包到编译目录：

```
# cd /app/gopkg/src/github.com/CodisLabs/
# wget https://github.com/CodisLabs/codis/archive/release3.2.zip
# unzip release3.2
```
或者
```
# git clone https://github.com/CodisLabs/codis.git -b release3.2

# ln -sv ./codis-release3.2 ./codis
# cd ./codis
# make   //如出现报错 ，可换成  make MALLOC=libc
```
确认编译无报错即可

编译完成后，bin 目录下生成如下文件：
```
[root codies140 08:50:46] /app/gopkg/src/github.com/CodisLabs/codis
-- # ll ./bin
total 83188
drwxr-xr-x 4 root root     4096 Apr 20 08:18 assets
-rwxr-xr-x 1 root root 15206342 Apr 20 08:18 codis-admin
-rwxr-xr-x 1 root root 16775582 Apr 20 08:17 codis-dashboard
-rwxr-xr-x 1 root root 14938086 Apr 20 08:18 codis-fe
-rwxr-xr-x 1 root root 18968628 Apr 20 08:18 codis-proxy
-rwxr-xr-x 1 root root  7982779 Apr 20 08:17 codis-server
-rwxr-xr-x 1 root root  5580447 Apr 20 08:17 redis-benchmark
-rwxr-xr-x 1 root root  5712403 Apr 20 08:17 redis-cli
-rw-r--r-- 1 root root       94 Apr 20 08:17 version
```

此处的/bin 目录会被后续 codis 各角色所使用；

### *2.2.2. codis-server 实例部署*
此处单机部署两个 redis 实例，一主一从为一组，建议部署多组；

创建 codis-server 实例所需目录：
```
mkdir -pv /app/codis/redis/{7001,7002}
```

复制 codis 的 bin 目录和 redis 的配置文件：
```
# cp -r /app/gopkg/src/github.com/CodisLabs/codis/bin /app/codis/
# cp /app/gopkg/src/github.com/CodisLabs/codis/extern/redis-3.2.8/redis.conf /app/codis/redis/7001/redis.conf
# cp /app/gopkg/src/github.com/CodisLabs/codis/extern/redis-3.2.8/redis.conf /app/codis/redis/7002/redis.conf
```

修改 codis-server 的配置文件：

```
# cd /app/codis/redis
```

配置 redis7001（主库）：

```
[root codies140 09:09:39] /app/codis/redis
-- # grep "^[^#].*" ./7001/redis.conf
bind 0.0.0.0
protected-mode no
port 7001
tcp-backlog 511
timeout 60
tcp-keepalive 300
daemonize yes
supervised no
pidfile /tmp/redis_7001.pid
loglevel notice
logfile "/app/codis/redis/7001/redis_7001.log"
databases 16
save 900 1
save 300 10
save 60 10000
stop-writes-on-bgsave-error yes
rdbcompression yes
rdbchecksum yes
dbfilename dump_7001.rdb
dir /app/codis/redis/7001/
masterauth codis
slave-serve-stale-data yes
slave-read-only yes
repl-diskless-sync no
repl-diskless-sync-delay 5
repl-disable-tcp-nodelay no
slave-priority 100
requirepass codis
```

```
maxmemory 2gb
appendonly yes
appendfilename "appendonly.aof"
appendfsync everysec
no-appendfsync-on-rewrite no
auto-aof-rewrite-percentage 100
auto-aof-rewrite-min-size 64mb
aof-load-truncated yes
lua-time-limit 5000
slowlog-log-slower-than 10000
slowlog-max-len 128
latency-monitor-threshold 0
notify-keyspace-events ""
hash-max-ziplist-entries 512
hash-max-ziplist-value 64
list-max-ziplist-size -2
list-compress-depth 0
set-max-intset-entries 512
zset-max-ziplist-entries 128
zset-max-ziplist-value 64
hll-sparse-max-bytes 3000
activerehashing yes
client-output-buffer-limit normal 0 0 0
client-output-buffer-limit slave 256mb 64mb 60
client-output-buffer-limit pubsub 32mb 8mb 60
hz 10
aof-rewrite-incremental-fsync yes
```

配置 redis7002（从库）：

```
[root codies140 09:10:18] /app/codis/redis
-- # grep "^[^#].*" ./7002/redis.conf
bind 0.0.0.0
protected-mode no
port 7002
tcp-backlog 511
timeout 60
```

```
tcp-keepalive 300
daemonize yes
supervised no
pidfile /tmp/redis_7002.pid
loglevel notice
logfile "/app/codis/redis/7002/redis_7002.log"
databases 16
save 900 1
save 300 10
save 60 10000
stop-writes-on-bgsave-error yes
rdbcompression yes
rdbchecksum yes
dbfilename dump_7002.rdb
dir /app/codis/redis/7002/
# slaveof 10.0.5.140 7001  //此处不需要指定主库，后续会在 codis-fe 管理界面里指定；
masterauth codis
slave-serve-stale-data yes
slave-read-only yes
repl-diskless-sync no
repl-diskless-sync-delay 5
repl-ping-slave-period 10
repl-timeout 60
repl-disable-tcp-nodelay no
repl-backlog-size 1mb
slave-priority 100
requirepass codis
maxmemory 2gb
appendonly yes
appendfilename "appendonly.aof"
appendfsync everysec
no-appendfsync-on-rewrite no
auto-aof-rewrite-percentage 100
auto-aof-rewrite-min-size 64mb
aof-load-truncated yes
lua-time-limit 5000
```

```
slowlog-log-slower-than 10000
slowlog-max-len 128
latency-monitor-threshold 0
notify-keyspace-events ""
hash-max-ziplist-entries 512
hash-max-ziplist-value 64
list-max-ziplist-size -2
list-compress-depth 0
set-max-intset-entries 512
zset-max-ziplist-entries 128
zset-max-ziplist-value 64
hll-sparse-max-bytes 3000
activerehashing yes
client-output-buffer-limit normal 0 0 0
client-output-buffer-limit slave 256mb 64mb 60
client-output-buffer-limit pubsub 32mb 8mb 60
hz 10
aof-rewrite-incremental-fsync yes
```

注意：codis-server 配置文件里不要设置密码，否则会出现 codis-fe 管理界面添加 codis-server 报错；

启动 codis-server 实例：

```
# /app/codis/bin/codis-server /app/codis/redis/7001/redis.conf
# /app/codis/bin/codis-server /app/codis/redis/7002/redis.conf
```

codis-server 读写测试：

```
[root codies140 09:17:49] /app/codis/bin
-- # ./redis-cli -h 10.0.5.140 -p 7001
10.0.5.140:7001> AUTH codis
OK
10.0.5.140:7001> SET name tom
OK
10.0.5.140:7001> GET na
```

```
[root codies140 09:18:53] /app/codis/bin
-- # ./redis-cli -h 10.0.5.140 -p 7002
10.0.5.140:7002> AUTH codis
OK
10.0.5.140:7002> SET name tom
OK
10.0.5.140:7002> GET name
"tom"
```

可以看到两个 codis-server 实例读写数据均 OK;

## 三：部署 codis-proxy

### 3.1. 生成 codis-proxy 配置文件

```
# cd /app/codis/bin/
# /app/codis/bin/codis-proxy --default-config |tee proxy.toml (proxy.conf)
#
# vim ./proxy.toml

##################################################
#                                                #
#                  Codis-Proxy                   #
#                                                #
##################################################

# 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
#   1. jodis_name is short for jodis_coordinator_name, only accept "zookeeper" & "etcd".
#   2. jodis_addr is short for jodis_coordinator_addr
#   3. proxy will be registered as node:
#        if jodis_compatible = true (not suggested):
#          /zk/codis/db_{PRODUCT_NAME}/proxy-{HASHID} (compatible with Codis2.0)
#        or else
#          /jodis/{PRODUCT_NAME}/proxy-{HASHID}
jodis_name = "zookeeper"
jodis_addr = "10.0.5.140:2881,10.0.5.140:2882,10.0.5.140:2883"  //zookeeper 地址
jodis_timeout = "20s"
jodis_compatible = false

# Set datacenter of proxy.
proxy_datacenter = ""

# Set max number of alive sessions.
proxy_max_clients = 1000

# Set max offheap memory size. (0 to disable)
proxy_max_offheap_size = "1024mb"

# Set heap placeholder to reduce GC frequency.
proxy_heap_placeholder = "256mb"

# Proxy will ping backend redis (and clear 'MASTERDOWN' state) in a predefined interval. (0 to disable)
backend_ping_period = "5s"

# Set backend recv buffer size & timeout.
backend_recv_bufsize = "128kb"
backend_recv_timeout = "30s"

# Set backend send buffer & timeout.
backend_send_bufsize = "128kb"
backend_send_timeout = "30s"
```

```
# Set backend pipeline buffer size.
backend_max_pipeline = 1024

# Set backend never read replica groups, default is false
backend_primary_only = false

# Set backend parallel connections per server
backend_primary_parallel = 1
backend_replica_parallel = 1
```

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 端口口 |
| proto_type | Redis 端口口类型，接受 tcp/tcp4/tcp6/unix/unixpacket |
| proxy_addr | Redis 端口口地址或者路路径 |
| 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 的路路径 |
| backend_ping_period | 与 codis-server 探活周期，单位 second，0 表示禁止止 |
| session_max_timeout | 与 client 连接最大大读超时，单位 second，0 表示禁止止 |
| session_max_bufsize | 与 client 连接读写缓冲区大大小小，单位 byte |
| session_max_pipeline | 与 client 连接最大大的 pipeline大大小小 |
| session_keepalive_period | 与 client 的 tcp keepalive 周期，仅 tcp 有效，0 表示禁止止 |

## 3.2. 管理 codis-proxy 服务

启动 codis-proxy：

```
[root codies140 10:11:36] /app/codis/bin
-- # nohup /app/codis/bin/codis-proxy --ncpu=4 --config=proxy.toml --log=proxy.log --log-level=WARN &
```

正常关闭 codis-proxy：

```
[root codies140 10:12:47] /app/codis/bin
-- # /app/codis/bin/codis-admin --proxy=10.0.5.140:11080 --auth="xxx"（有密码就加，没有就不加） --shutdown
```

```
# ./codis-proxy -h
Usage:
     codis-proxy [--ncpu=N [--max-ncpu=MAX]] [--config=CONF] [--log=FILE] [--log-level=LEVEL] [--host-admin=ADDR] [--host-proxy=ADDR]
     [--dashboard=ADDR|--zookeeper=ADDR|--etcd=ADDR|--filesystem=ROOT|--fillslots=FILE] [--ulimit=NLIMIT] [--pidfile=FILE]
     codis-proxy  --default-config
     codis-proxy  --version

Options:
     --ncpu=N                    set runtime.GOMAXPROCS to N, default is runtime.NumCPU().
     -c CONF, --config=CONF      run with the specific configuration.
     -l FILE, --log=FILE         set path/name of daliy rotated log file.
     --log-level=LEVEL           set the log-level, should be INFO,WARN,DEBUG or ERROR, default is INFO.
     --ulimit=NLIMIT             run 'ulimit -n' to check the maximum number of open file descriptors.
```

## 四：部署 codis-dashboard

### 4.1.配置 codis-dashboard

生成 codis-dashboard 配置文件：

```
# cd /app/codis/bin
# /app/codis/bin/codis-dashboard --default-config |tee dashboard.toml (dashboard.conf)
#
# vim ./dashboard.toml

##################################################
#                                                #
#                  Codis-Dashboard               #
#                                                #
##################################################

# Set Coordinator, only accept "zookeeper" & "etcd" & "filesystem".
# Quick Start
coordinator_name = "zookeeper"  \\ 外部存储类型，接受 zookeeper|etcd
coordinator_addr = "10.0.5.140:2181,10.0.5.140:2182,10.0.5.140:2183"   \\ zookeeper 外部存储地址
#coordinator_name = "zookeeper"
```

```
#coordinator_addr = "127.0.0.1:2181"

# 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"

# Set arguments for data migration (only accept 'sync' & 'semi-async').
migration_method = "semi-async"
migration_parallel_slots = 100
migration_async_maxbulks = 200
migration_async_maxbytes = "32mb"
migration_async_numkeys = 500
migration_timeout = "30s"

# Set configs for redis sentinel.
sentinel_quorum = 2
sentinel_parallel_syncs = 1
sentinel_down_after = "30s"
sentinel_failover_timeout = "5m"
sentinel_notification_script = ""
sentinel_client_reconfig_script = ""
```

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，满足足正则 \w[\w\.\-]*
product_auth 集群密码，默认为空
admin_addr RESTful API 端口口

## 4.2. 管理 codis-dashboard 服务

启动 codis-dashboard：

```
[root codies140 09:57:19] /app/codis/bin
```

```
-- # nohup /app/codis/bin/codis-dashboard --ncpu=4 --config=dashboard.toml --log=dashboard.log --log-level=WARN &
```

正常关闭 codis-dashboard：

```
# /app/codis/bin/codis-admin --dashboard=10.0.5.140:18080 --shutdown


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

Options:
	--ncpu=N                    set runtime.GOMAXPROCS to N, default is runtime.NumCPU().
	-c CONF, --config=CONF      run with the specific configuration.
	-l FILE, --log=FILE         set path/name of daliy rotated log file.
	--log-level=LEVEL           set the log-level, should be INFO,WARN,DEBUG or ERROR, default is INFO.
```

## 五：部署 codis-fe

### 5.1.生成 codis-fe 配置文件

```
# cd /app/codis/bin
# /app/codis/bin/codis-admin --dashboard-list --zookeeper=10.0.5.140:2181 |tee ./codis.json
...
[
    {
        "name": "codis-demo",
        "dashboard": "10.0.5.140:18080"
    }
]
...
```

### 5.2.管理 codis-fe 服务

启动 codis-fe 服务：

```
[root codies140 10:21:00] /app/codis/bin
-- # nohup /app/codis/bin/codis-fe --ncpu=4 --log=fe.log --log-level=WARN --dashboard-list=codis.json --listen=0.0.0.0:18090 &
```

和 codis 相关的服务及端口有：

```
 # ss -tunlp |grep codis
tcp    LISTEN     0      128                     :::18080                  :::*      users:(("codis-dashboard",13795,6))
tcp    LISTEN     0      128                     :::11080                  :::*      users:(("codis-proxy",13851,6))
tcp    LISTEN     0      128                     :::18090                  :::*      users:(("codis-fe",13881,5))
tcp    LISTEN     0      128                      *:19000                   *:*      users:(("codis-proxy",13851,4))
tcp    LISTEN     0      128                      *:7001                    *:*      users:(("codis-server",13601,4))
tcp    LISTEN     0      128                      *:7002                    *:*      users:(("codis-server",13786,4))
```

## 六：redis-sentinel 部署

### 6.1.部署多实例的 redis-sentinel

此处部署三个 redis-sentinel 实例；

创建 redis-sentinel 目录，复制 redis-sentinel 配置文件：

```
# mkdir /app/codis/sentinel/{27001,27002,27003}
# cp /app/gopkg/src/github.com/CodisLabs/codis/extern/redis-3.2.8/sentinel.conf /app/codis/sentinel/
```

```
# vim  /app/codis/sentinel/sentinel.conf
bind 0.0.0.0
protected-mode no
port 27001
dir /app/codis/sentinel/27001/
```

注意：此时不用指定 monitor 节点，后续会在 codis-fe 管理界面里添加 sentinel，然后向 sentinel 指定需要 monitor 的节点；

将此配置文件复制到各 sentinel 实例目录下，修改监听端口和数据目录：

sentinel 实例 1：

```
# cp /app/codis/sentinel/sentinel.conf /app/codis/sentinel/27001/
# grep "^[^#].*" /app/codis/sentinel/27001/sentinel.conf
bind 0.0.0.0
protected-mode no
port 27001
dir /app/codis/sentinel/27001/
```

sentinel 实例 2：

```
# cp /app/codis/sentinel/sentinel.conf /app/codis/sentinel/27002/
# sed -i "s/27001/27002/g" /app/codis/sentinel/27002/sentinel.conf
# grep "^[^#].*" ./27002/sentinel.conf
bind 0.0.0.0
protected-mode no
port 27002
dir /app/codis/sentinel/27002/
```

sentinel 实例 3：

```
# cp /app/codis/sentinel/sentinel.conf /app/codis/sentinel/27003/
# sed -i "s/27001/27003/g" /app/codis/sentinel/27003/sentinel.conf
# grep "^[^#].*" /app/codis/sentinel/27003/sentinel.conf
bind 0.0.0.0
protected-mode no
port 27003
dir /app/codis/sentinel/27003/
```

codis/bin 下没有 redis-sentinel 命令，需要将 redis-xxx/src 目录下 redis-sentinel 命令复制到 codis/bin 目录下：

```
# cp /app/gopkg/src/github.com/CodisLabs/codis/extern/redis-3.2.8/src/redis-sentinel /app/codis/bin/
```

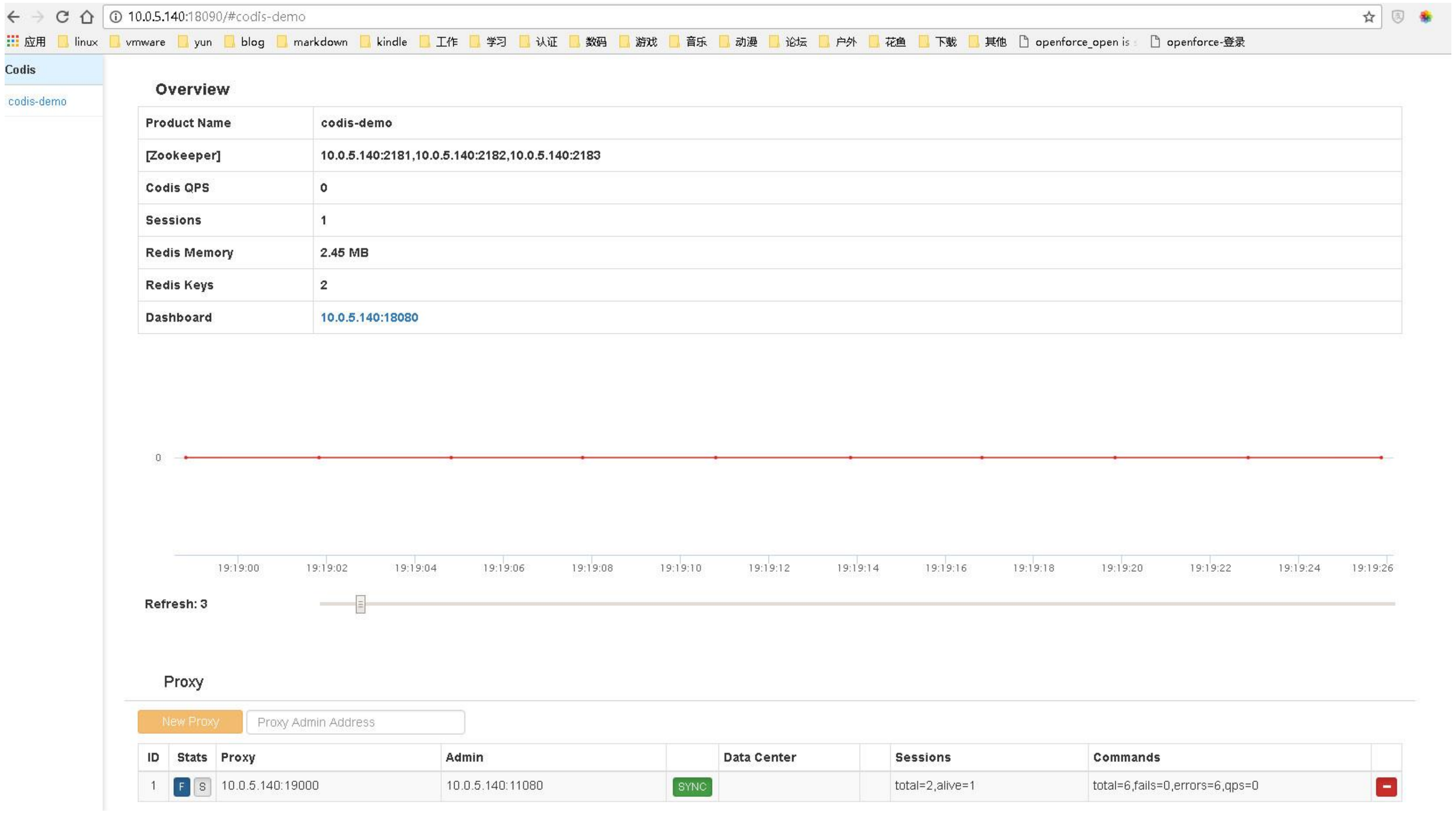
启动 sentinel 服务：

```
# /app/codis/bin/codis-server /app/codis/sentinel/27001/sentinel.conf
# /app/codis/bin/codis-server /app/codis/sentinel/27002/sentinel.conf
# /app/codis/bin/codis-server /app/codis/sentinel/27003/sentinel.conf
```

# 七：使用 codis-fe 管理 codis 集群环境

## 7.1. codis-fe 管理界面

使用浏览器打开 codis-fe 管理界面： http://10.0.5.140:18090



Overview

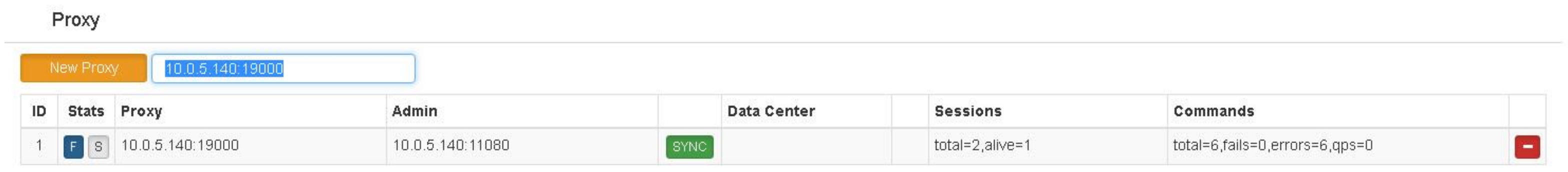
| | |
|---|---|
| Product Name | codis-demo |
| [Zookeeper] | 10.0.5.140:2181,10.0.5.140:2182,10.0.5.140:2183 |
| Codis QPS | 0 |
| Sessions | 1 |
| Redis Memory | 2.45 MB |
| Redis Keys | 2 |
| Dashboard | 10.0.5.140:18080 |

Proxy

| ID | Stats | Proxy | Admin | | Data Center | | Sessions | Commands | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | F S | 10.0.5.140:19000 | 10.0.5.140:11080 | SYNC | | | total=2,alive=1 | total=6,fails=0,errors=6,qps=0 | |

## 7.2. 添加 codis-proxy、codis-server、redis-sentinel

添加 codis-proxy：

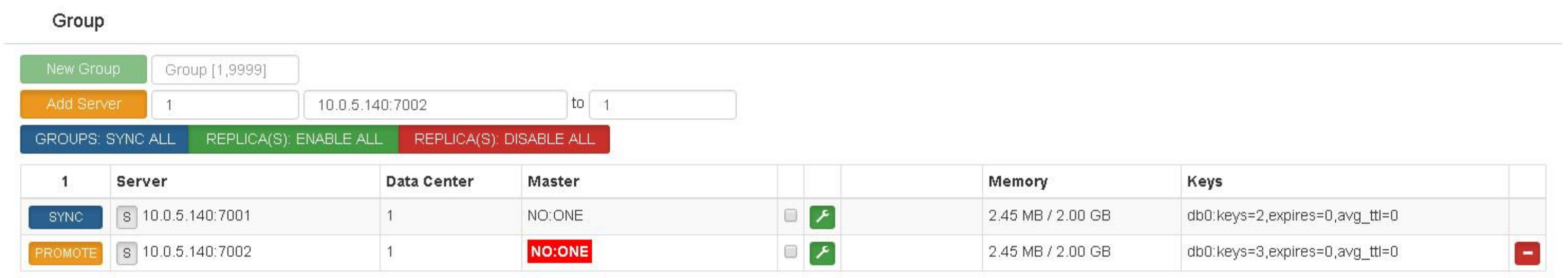


Proxy

| ID | Stats | Proxy | Admin | | Data Center | | Sessions | Commands | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | F S | 10.0.5.140:19000 | 10.0.5.140:11080 | SYNC | | | total=2,alive=1 | total=6,fails=0,errors=6,qps=0 | |

添加 codis-server：

group：一组主从 codis-server 为一组；



Group

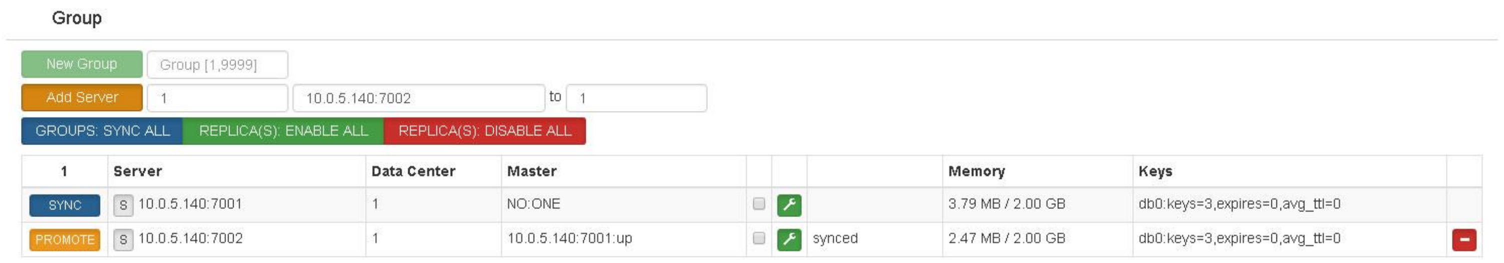
New Group | Group [1,9999]

Add Server | 1 | 10.0.5.140:7002 | to | 1

GROUPS: SYNC ALL | REPLICA(S): ENABLE ALL | REPLICA(S): DISABLE ALL

| 1 | Server | Data Center | Master | | | | Memory | Keys | |
|---|---|---|---|---|---|---|---|---|---|
| SYNC | S 10.0.5.140:7001 | 1 | NO:ONE | | | | 2.45 MB / 2.00 GB | db0:keys=2,expires=0,avg_ttl=0 | |
| PROMOTE | S 10.0.5.140:7002 | 1 | NO:ONE | | | | 2.45 MB / 2.00 GB | db0:keys=3,expires=0,avg_ttl=0 | |

设置 codis-server 主从：



Group

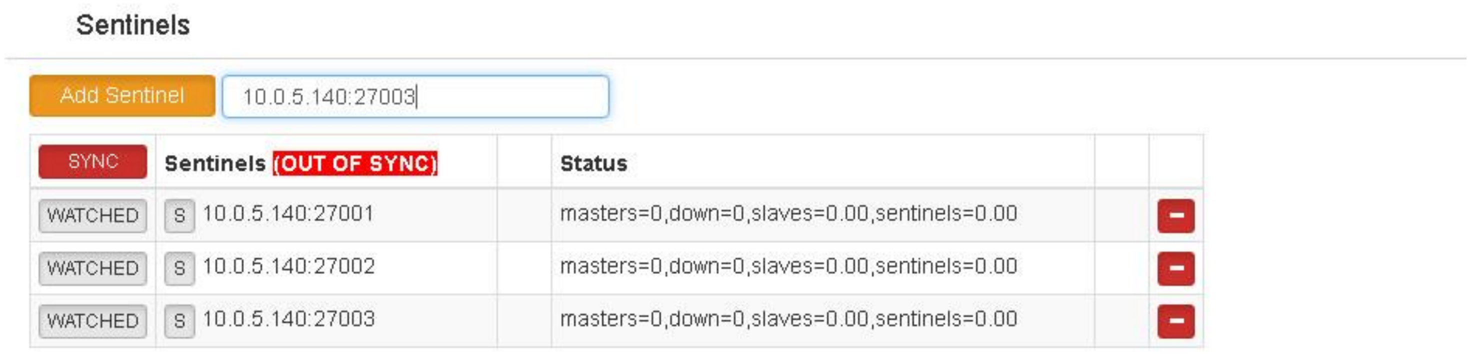
New Group | Group [1,9999]

Add Server | 1 | 10.0.5.140:7002 | to | 1

GROUPS: SYNC ALL | REPLICA(S): ENABLE ALL | REPLICA(S): DISABLE ALL

| 1 | Server | Data Center | Master | | | | Memory | Keys | |
|---|---|---|---|---|---|---|---|---|---|
| SYNC | S 10.0.5.140:7001 | 1 | NO:ONE | | | | 3.79 MB / 2.00 GB | db0:keys=3,expires=0,avg_ttl=0 | |
| PROMOTE | S 10.0.5.140:7002 | 1 | 10.0.5.140:7001:up | | | synced | 2.47 MB / 2.00 GB | db0:keys=3,expires=0,avg_ttl=0 | |

添加 redis-sentinel：

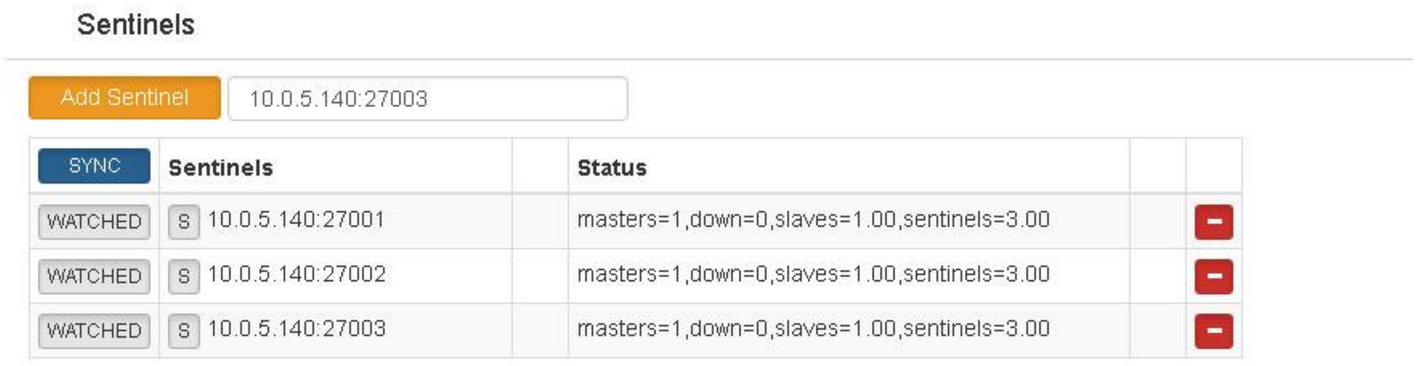


Sentinels

Add Sentinel | 10.0.5.140:27003

| SYNC | Sentinels (OUT OF SYNC) | | Status | | |
|---|---|---|---|---|---|
| WATCHED | S 10.0.5.140:27001 | | masters=0,down=0,slaves=0.00,sentinels=0.00 | | |
| WATCHED | S 10.0.5.140:27002 | | masters=0,down=0,slaves=0.00,sentinels=0.00 | | |
| WATCHED | S 10.0.5.140:27003 | | masters=0,down=0,slaves=0.00,sentinels=0.00 | | |

同步 redis-sentinel：

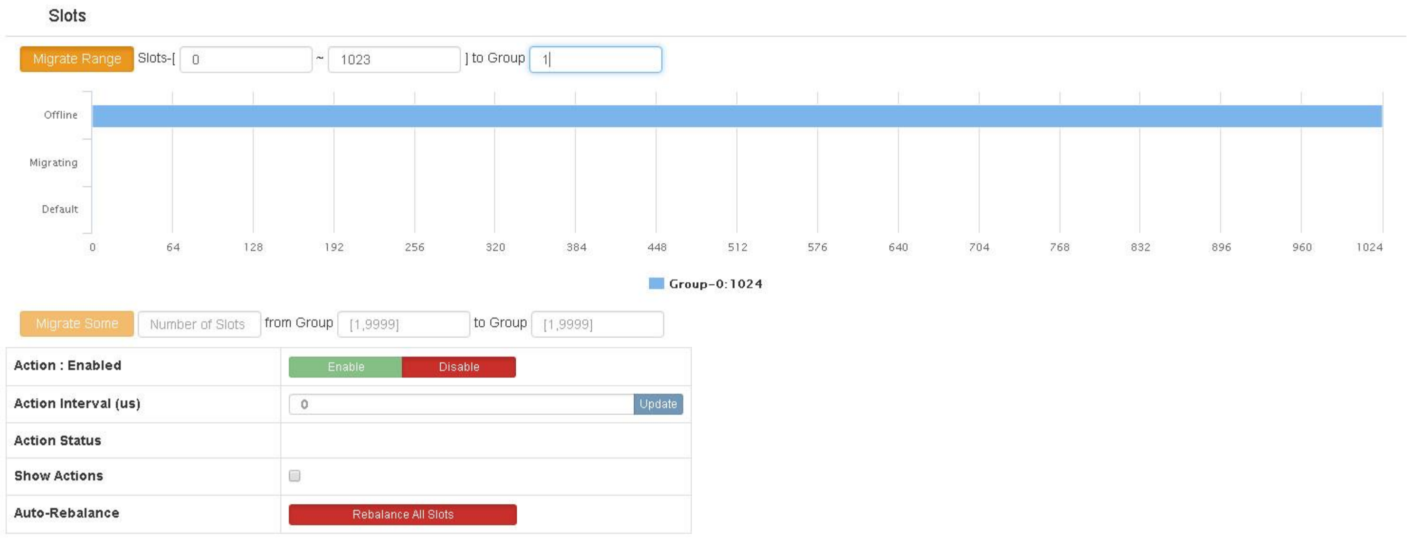

Sentinels

Add Sentinel 10.0.5.140:27003

| SYNC | Sentinels | | Status | | |
|---|---|---|---|---|---|
| WATCHED | S 10.0.5.140:27001 | | masters=1,down=0,slaves=1.00,sentinels=3.00 | | − |
| WATCHED | S 10.0.5.140:27002 | | masters=1,down=0,slaves=1.00,sentinels=3.00 | | − |
| WATCHED | S 10.0.5.140:27003 | | masters=1,down=0,slaves=1.00,sentinels=3.00 | | − |

分配 slot：

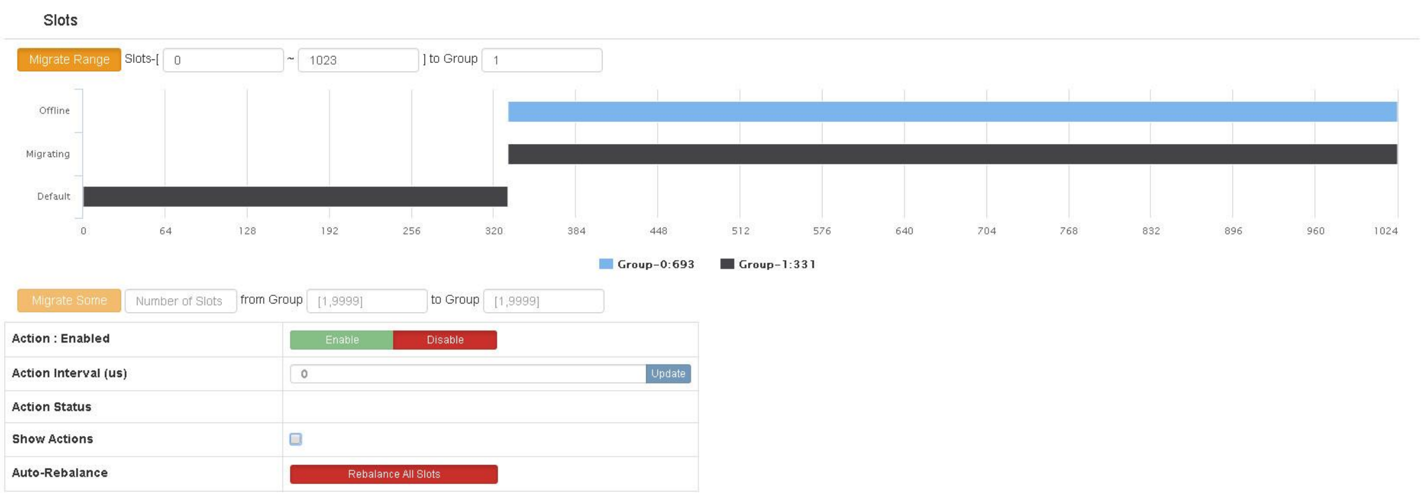
slot：数据槽，在 codis-server 间数据分片的单元；建议将数据槽均分于多个 group；


Slots

Migrate Range Slots-[ 0 ~ 1023 ] to Group 1

Migrate Some Number of Slots from Group [1,9999] to Group [1,9999]

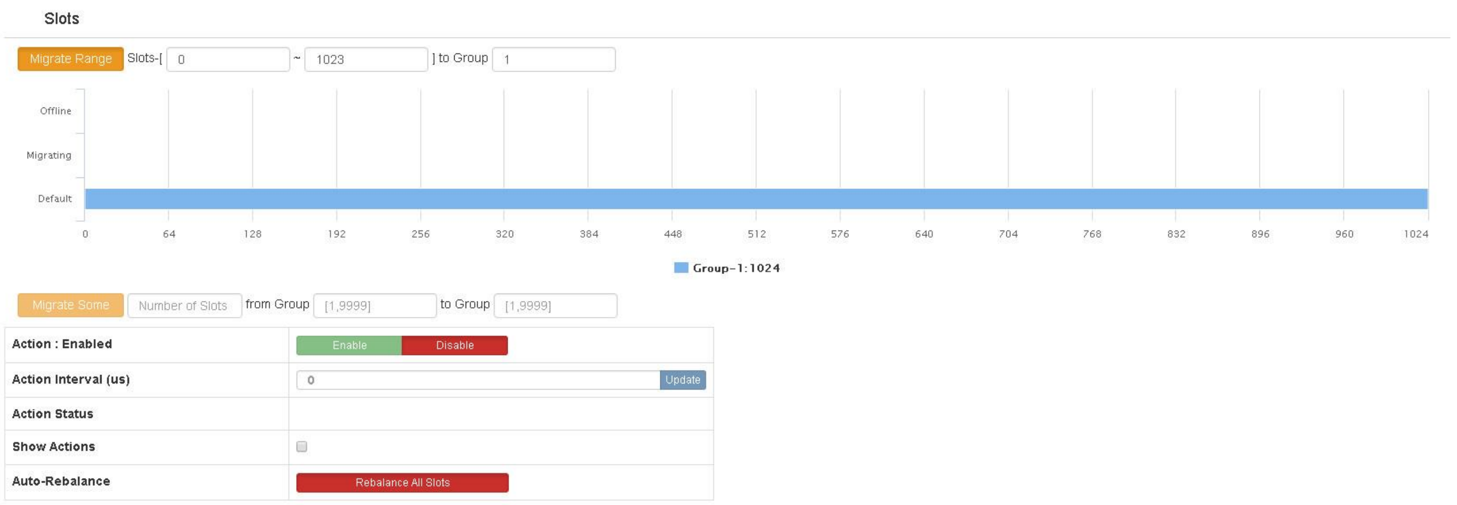
| | |
|---|---|
| Action : Enabled | Enable Disable |
| Action Interval (us) | 0 Update |
| Action Status | |
| Show Actions | ☐ |
| Auto-Rebalance | Rebalance All Slots |



Slots-[ 0 ~ 1023 ] to Group 1

Number of Slots from Group [1,9999] to Group [1,9999]

| | |
|---|---|
| **Action : Enabled** | Enable Disable |
| **Action Interval (us)** | 0 Update |
| **Action Status** | |
| **Show Actions** | ☐ |
| **Auto-Rebalance** | Rebalance All Slots |



Slots-[ 0 ~ 1023 ] to Group 1

Number of Slots from Group [1,9999] to Group [1,9999]

| | |
|---|---|
| **Action : Enabled** | Enable Disable |
| **Action Interval (us)** | 0 Update |
| **Action Status** | |
| **Show Actions** | ☐ |
| **Auto-Rebalance** | Rebalance All Slots |

在 codis-fe 管理界面添加 redis-sentinel，并同步，同步后 redis-sentinel 配置文件会更新：

```
# grep "^[^#].*" /app/codis/sentinel//27001/sentinel.conf
bind 0.0.0.0
protected-mode no
```

```
port 27001
dir "/app/codis/sentinel/27001"
sentinel myid da5cefe099c9dbfa7583998ce33051e24bcf8284
sentinel monitor codis-demo-1 10.0.5.140 7001 2
sentinel failover-timeout codis-demo-1 300000
sentinel config-epoch codis-demo-1 0
sentinel leader-epoch codis-demo-1 0
sentinel known-slave codis-demo-1 10.0.5.140 7002
sentinel known-sentinel codis-demo-1 10.0.5.140 27003 a2235596194dd9c81b941cbbaa7685326a848d10
sentinel known-sentinel codis-demo-1 10.0.5.140 27002 23dc0dfe33f9a513a5b0111e388371353b849974
sentinel current-epoch 0
```

可以看到配置文件新增了 monitor 节点信息;

至此完成单机环境的 codis 集群部署;

## 八：环境总结

实验目录结构：

```
# tree -L 2 /app
/app
├── codis
│   ├── bin
│   ├── redis
│   └── sentinel
├── codis-release3.2.zip
├── go
│   ├── api
│   ├── AUTHORS
│   ├── bin
│   ├── blog
│   ├── CONTRIBUTING.md
│   ├── CONTRIBUTORS
│   ├── doc
│   ├── favicon.ico
│   ├── lib
│       ├──── LICENSE
│       ├──── misc
│       ├──── PATENTS
│       ├──── pkg
│       ├──── README.md
│       ├──── robots.txt
│       ├──── src
│       ├──── test
│       └──── VERSION
├──── go1.8.linux-amd64.tar.gz
├──── gopkg
│       ├──── pkg
│       └──── src
├──── zk1
│       ├──── data
│       ├──── log
│       ├──── zookeeper -> ./zookeeper-3.4.10
│       └──── zookeeper-3.4.10
├──── zk2
│       ├──── data
│       ├──── log
│       ├──── zookeeper -> ./zookeeper-3.4.10
│       └──── zookeeper-3.4.10
├──── zk3
│       ├──── data
│       ├──── log
│       ├──── zookeeper -> ./zookeeper-3.4.10
│       └──── zookeeper-3.4.10
├──── zookeeper-3.4.10.tar.gz
└──── zookeeper.out
```