

Redis High Availability Cluster Sharding setup with HAproxy Load balancer

Chapter 1: Redis Master - Slave setup (Manual Setup)

1. Create a 3 nodes redis cluster which has 1 master and 2 slaves. In my case below are the Ubuntu VM details:

IP Address	Port	Hostname	Role
10.0.1.4	6379	Ubuntu01	Master1
10.0.1.5	6380	Ubuntu02	Master2
10.0.1.6	6381	Ubuntu03	Master3
10.0.1.4	6381	Ubuntu01 Slave of Master3	
10.0.1.5	6379	Ubuntu02 Slave of Master1	
10.0.1.6	6380	Ubuntu03 Slave of Master2	
10.0.1.50	6381	Load Balancer	

2. Update Ubuntu packages and Install build-essential and tcl packages. Redis is build manually then make command needs these libraries.

```
sudo apt-get update
sudo apt-get install -y build-essential tcl
```

3. Download the redis file and extract it into a folder.

```
wget http://download.redis.io/releases/redis-5.0.3.tar.gz
tar xvzf redis-5.0.3.tar.gz
cd redis-5.0.3
```

4. Then with make command, build redis binary. If you wish by make test command, tests should be run.

```
make
make test
sudo make install
```

5. By providing the file locations and redis port to the ./utils/install_server.sh, necessary system services are created. Redis-server service will create after executing install.sh script.

```
cd utils
sudo sh install_server.sh
➤ sudo REDIS_PORT=<port>
➤ REDIS_CONFIG_FILE=/etc/redis/<port>.conf
➤ REDIS_LOG_FILE=/var/log/redis_<port>.log
➤ REDIS_LOG_FILE=/var/log/redis_<port>.log
➤ REDIS_EXECUTABLE='command -v redis-server' ./utils/install_server.sh
```

- Now we have 3 redis servers with default configurations, it is time to configure them as master slave model.
- Redis holds config file at location /etc/redis/<port>.conf

Redis High Availability Cluster Sharding setup with HAproxy Load balancer

6. Edit /etc/redis/6379.conf file on Master Redis – server and change below tags:

```
vi /etc/redis/<port>.conf
```

- daemonize yes
- port <port>
- bind 0.0.0.0
- dbfilename "dump.rdb"
- dir "/var/lib/redis"
- protected-mode no
- save 10 1
- cluster-enabled yes
- cluster-config-file node-<port>.conf
- cluster-node-timeout 15000

- **Now our 3 master servers configured successfully. Its time to configure 3 slave nodes.**
- **To configure 3 slave servers and to add it to the respective master node follow the same process as we have followed to configure the master and after we will add these 3 nodes to cluster as slave node with their master node.**

○

7. To add other 3 servers as slave nodes run below command:

```
Redis-cli -cluster add-node <slave-ip:port> <master-ip:port> --cluster-slave
```

8. To verify the Cluster Info and Cluster Nodes run below command:

```
Redis-cli -h <master-ip> -p <port> -c
```

- Cluster info (shows the cluster status)
- Cluster nodes (shows the master and slave nodes)

9. Execute command to set key and check the sharding in redis :

```
Set key1 "1st test key"
```

- Ok

```
Set key2 "2nd test key"
```

- Redirected to slot [16090] located at <other master node>
- Ok


10. If user or admin wants to know no of key slots assigned to every master node and till now total number of keys stored in all three servers as well as how many keys are stored in each master nodes, run below command.

```
redis-cli -cluster check <master:port>
```


Redis High Availability Cluster Sharding setup with HAproxy Load balancer

CHAPTER 2: Add New Node to Cluster

1. Before adding a new node, we need to configure redis to that node for that steps are given in Chapter 1.
2. After configuring the redis to the new node add that node to the existing cluster, for that run below command:

 `Redis-cli --cluster add-node <new node-ip:port> <existing node-ip:port>`

3. Slots are not assigned automatically after adding the new node to cluster, to assign the slots to the new node run below command:

 `Redis-cli --cluster resharding <new node-ip:port>`

- Give the number of slots wants to move to the new node
- Enter the receiving node node id
- Enter the source node id (from where we want to take the slots)

4. After running above command, verify that slots are assigned to the new node or not and also make sure that if any key present in that slots then that key should also move to the new node. We can verify the same by executing below command:

 `Redis-cli --cluster check <master node-ip:port>`

Redis High Availability Cluster Sharding setup with HAproxy Load balancer

Chapter 3: HAproxy and Keepalived setup for virtual ip and load balancing.

1. Install packages of haproxy and keepalived

```
sudo apt-get install keepalived haproxy
```

2. Tweak sysctl to allow haproxy to bind to the virtual ip, even if it is not assigned to the node its running on.

```
echo "net.ipv4.ip_nonlocal_bind=1" | sudo tee -a /etc/sysctl.conf  
sysctl -p
```

3. Configure HAproxy load balancer by editing /etc/haproxy/haproxy.conf

```
frontend redis  
#haproxy should listen on the virtual ip  
bind <load balancer-ip:port> name redis  
default_backend redis_backend  
backend redis_backend  
option tcp-check  
#haproxy will look for the following strings to determine the master  
tcp-check send PING\r\n  
tcp-check expect string +PONG  
tcp-check send info\r\n replication\r\n  
tcp-check expect string role:master  
tcp-check send QUIT\r\n  
tcp-check expect string +OK  
#these are the ip's of the two redis nodes  
server redis1 <master/slave-ip:port> check inter 1s
```

4. Reload the HAproxy service by running below command:

```
sudo /etc/init.d/haproxy reload
```

Redis High Availability Cluster Sharding setup with HAproxy Load balancer

5. Edit keepalived's configuration file (/etc/keepalived/keepalived.conf) and reload it.

```
✚ vrrp_script chk_haproxy {  
    script "killall -0 haproxy" # verify the pid existance  
    interval 2 # check every 2 seconds  
    weight 2 # add 2 points of prio if OK  
}  
vrrp_instance VI_1 {  
    interface eth0 # interface to monitor  
    state MASTER # backup for slaves  
    virtual_router_id 51 # Assign one ID for this route  
    priority 101 # 101 on master, 100 on backup  
    virtual_ipaddress {  
        <virtual ip> # the virtual IP  
    }  
    track_script {  
        chk_haproxy  
    }  
}
```

6. Start the keepalived service and reload it as shown below:

```
✚ sudo systemctl restart keepalived  
✚ sudo /etc/init.d/keepalived reload
```

7. Confirm redis is listening on the correct ip's using:

- `sudo netstat -tnlp`

8. Try to login with load balancer virtual IP, it should point to current master redis server:

- `redis-cli -h 10.0.0.50`