PROCESSROBOT Redis INSTALLATION GUIDE

Enterprise Robotic Process Automation 2019

softomotive

We talk automation

processrobot

Building and Installing Redis

In this document, the steps for building and installing Redis are described.

Install dependencies

```
apt install -y curl build-essential tcl
```

```
root@Redis-3:/home/pr_user# apt install -y curl build-essential tcl
```

Change to the temporary directory

```
cd /tmp
```

```
Setting up manpages-dev (4.04-2) ...
Setting up tcl8.6 (8.6.5+dfsg-2) ...
Setting up tcl (8.6.0+9) ...
Processing triggers for libc-bin (2.23-0ubuntu11) ...
root@Redis-3:/home/pr_user# cd /tmp
```

Download the latest stable Redis release

```
curl -O http://download.redis.io/redis-stable.tar.gz
```

```
Setting up tcl (8.6.0+9) ...
Processing triggers for libc-bin (2.23-0ubuntu11) ...
root@Redis-3:/home/pr_user# cd /tmp
root@Redis-3:/tmp# curl -O http://download.redis.io/redis-stable.tar.gz
```

Extract the downloaded files

```
tar xzvf redis-stable.tar.gz
```

```
root@Redis-3:/tmp# curl -O http://download.redis.io/redis-stable.tar.gz
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 1958k  100 1958k    0     0   297k      0  0:00:06  0:00:06 --:--:--  482k
root@Redis-3:/tmp# tar xzvf redis-stable.tar.gz
```

Change to the Redis directory

```
cd redis-stable
```

```
redis-stable/MANIFESTO
redis-stable/CONTRIBUTING
redis-stable/redis.conf
redis-stable/runtest-cluster
root@Redis-3:/tmp# cd redis-stable
```

Compile and Install Redis

```
make
```

```
redis-stable/redis.conf
redis-stable/runtest-cluster
root@Redis-3:/tmp# cd redis-stable
root@Redis-3:/tmp/redis-stable# make
```

```
make test
```

```
Hint: It's a good idea to run 'make test' ;)

make[1]: Leaving directory '/tmp/redis-stable/src'
root@Redis-3:/tmp/redis-stable# make test
```

The above command should complete without any errors. An errorless execution would prompt with the message shown below.

```
  36 seconds - unit/dump
  23 seconds - integration/psync2-reg
  26 seconds - integration/psync2
  8 seconds - unit/wait
  14 seconds - unit/pendingquerybuf
  51 seconds - unit/type/stream
  48 seconds - integration/replication-4
  33 seconds - unit/geo
  50 seconds - integration/replication-3
  59 seconds - unit/maxmemory
  77 seconds - unit/aofrw
  56 seconds - unit/hyperloglog
  71 seconds - unit/memefficiency
  99 seconds - unit/type/list-3
  83 seconds - unit/obuf-limits
  104 seconds - integration/replication-psync
  131 seconds - integration/replication

\o/ All tests passed without errors!

Cleanup: may take some time... OK
make[1]: Leaving directory '/tmp/redis-stable/src'
root@Redis-3:/tmp/redis-stable#
```

```
make install
```

```
\o/ All tests passed without errors!

Cleanup: may take some time... OK
make[1]: Leaving directory '/tmp/redis-stable/src'
root@Redis-3:/tmp/redis-stable# make install
cd src && make install
make[1]: Entering directory '/tmp/redis-stable/src'

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

    INSTALL install
    INSTALL install
    INSTALL install
    INSTALL install
    INSTALL install
make[1]: Leaving directory '/tmp/redis-stable/src'
root@Redis-3:/tmp/redis-stable#
```

Create the directory that will hold the Redis config.

```
mkdir -p /etc/redis
mkdir -p /var/redis
```

Copy configs to their respective dirs.

```
cp redis.conf /etc/redis/redis.conf
cp utils/redis_init_script /etc/init.d/redis
```

Copy the rest of the files to their respective directories

```
cp src/redis-server /usr/local/bin/
cp src/redis-sentinel /usr/local/bin/
cp src/redis-cli /usr/local/bin/*
```

Open the Redis Configuration file for editing

```
nano /etc/redis/redis.conf
```

Change the Bind Address to include the address that Redis will listen to. In this case, the IP address is (192.168.10.151) and the default port is 6379.

```
bind 127.0.0.1 192.168.10.151
```

By default Redis does not run as a daemon. Use 'yes' if you need it.

```
daemonize yes
```

The location of the two files that are created are shown.

```
pidfile /var/run/redis.pid
```

```
logfile /var/log/pr-redis.log
```

The filename where to dump the DB

```
dbfilename redis-dump.rdb
```

The working directory.

```
dir /var/redis
```

If the master is password protected (using the "requirepass" configuration directive below) it is possible to tell the replica to authenticate before starting the replication synchronization process, otherwise the master will refuse the replica request.

The Master Password should be long enough to prevent brute force attacks for two reasons:

Redis is very fast at serving queries. Many passwords per second can be tested by an external client. The Redis password is stored inside the redis.conf file and inside the client configuration, so it does not need to be remembered by the system administrator, and thus it can be very long.

The security of the Redis installation should comply with the Redis best practices (https://redis.io/topics/security)

```
masterauth foobar
```

Require clients to issue AUTH <PASSWORD> before processing any other commands. This might be useful in environments in which you do not trust others with access to the host running Redis-server.

This step should be completed in both the master and the slaves instances.

```
requirepass foobar
```

```
appendonly yes
```

When configuring a Redis replica, the ip address of the master should be used.

```
replicaof <masterip> <masterport>
```

```
ex. replicaof 192.168.10.214 6379
```

Create the init.d script for the Redis server.

```
nano /etc/init.d/redis.conf
```

Note: This is a sample script. Audit and adjust it for your environment prior to submitting.

```
#!/bin/sh
#
# Simple Redis init.d script conceived to work on Linux systems
# as it does use of the /proc filesystem.

### BEGIN INIT INFO
# Provides:     redis_6379
# Required-Start: $syslog
# Required-Stop: $syslog
# Default-Start:        2 3 4 5
# Default-Stop:         0 1 6
# Short-Description:    Redis data structure server
# Description:          Redis data structure server. See
https://redis.io
### END INIT INFO
```

```
REDISPORT=6379
EXEC=/usr/local/bin/redis-server
CLIEXEC=/usr/local/bin/redis-cli

PIDFILE=/var/run/redis.pid
CONF="/etc/redis/redis.conf"

case "$1" in
    start)
        if [ -f $PIDFILE ]
        then
                echo "$PIDFILE exists, process is already running or
crashed"
        else
                echo "Starting Redis server..."
                $EXEC $CONF
        fi
        ;;
    stop)
        if [ ! -f $PIDFILE ]
        then
                echo "$PIDFILE does not exist, process is not running"
        else
                PID=$(cat $PIDFILE)
                echo "Stopping ..."
                $CLIEXEC -p $REDISPORT -a foobar shutdown
                while [ -x /proc/${PID} ]
                do
                    echo "Waiting for Redis to shutdown ..."
                    sleep 1
                done
                echo "Redis stopped"
        fi
        ;;
    *)
        echo "Please use start or stop as first argument"
        ;;
esac
```

Once this file is created, run the following command to reload units.

```
systemctl daemon-reload
```

Start the Redis service.

```
/etc/init.d/redis start
or
service redis start
```

Next, run the following command to confirm that the service started successfully.

```
service redis status
```

```
root@Redis-3:/home/pr_user# service redis status
● redis.service - LSB: Redis data structure server
   Loaded: loaded (/etc/init.d/redis; bad; vendor preset: enabled)
   Active: active (running) since Thu 2019-04-25 17:31:14 EEST; 17s ago
     Docs: man:systemd-sysv-generator(8)
  Process: 956 ExecStart=/etc/init.d/redis start (code=exited, status=0/SUCCESS)
    Tasks: 4
   Memory: 3.3M
      CPU: 52ms
   CGroup: /system.slice/redis.service
           └─1004 /usr/local/bin/redis-server 127.0.0.1:6379

Apr 25 17:31:14 Redis-3 systemd[1]: Starting LSB: Redis data structure server...
Apr 25 17:31:14 Redis-3 redis[956]: Starting Redis server...
Apr 25 17:31:14 Redis-3 systemd[1]: Started LSB: Redis data structure server.
root@Redis-3:/home/pr_user#
```

Next login to the Redis command line interface.

```
redis-cli -a "your password here"
```
ex: redis-cli -a foobar

Once the connection has established, type ping. The response should be “PONG”.

After the above procedure has completed successfully for the remaining Redis replicas, it’s time to check the replication.

Set a variable in the master Redis node by typing the following command.

```
set var1 "Hello World"
```

Login to each replica and type the following command to confirm the replication is successful.

```
Get var1
```

The value “Hello World” should return on the two replicas.

Preparation of sentinels

Follow the below procedure, on each Redis server.

First, create a directory for the sentinel.

```
nano /etc/redis/sentinel.conf
```

Copy and paste the below(The bind command indicates the local machine's IP address while the Port it's port.):

```
bind 192.168.10.214 127.0.0.1
port 26379
dir "/var/sentinel"
sentinel monitor mymaster 192.168.10.214 6379 2
sentinel down-after-milliseconds mymaster 2000
sentinel failover-timeout mymaster 10000
sentinel auth-pass mymaster foobar
```

Then create the init.d file for the sentinel

```
nano /etc/init.d/sentinel
```

Pasting the following:

```
#!/bin/sh
#
# Simple Sentinel init.d script conceived to work on Linux systems as
it does use of the /proc filesystem.

### BEGIN INIT INFO
#Provides: sentinel
#Required-Start: $syslog
#Required-Stop: $syslog
#Default-Start: 2 3 4 5
#Default-Stop: 0 1 6
#Short-Description: start and stop sentinel
#Description: Sentinel daemon
### END INIT INFO


SENTINELPORT=26379
EXEC=/usr/local/bin/redis-sentinel
CLIEXEC=/usr/local/bin/redis-cli
```

```
PIDFILE=/var/run/sentinel.pid
CONF="/etc/redis/sentinel.conf"


case "$1" in
    start)
        if [ -f $PIDFILE ]
        then
                echo "$PIDFILE exists, process is already running or
crashed"
        else
                echo "Starting Sentinel server..."
                nohup $EXEC $CONF >> /var/log/redis-sentinel.log 2>&1 &
                echo $! > "${PIDFILE}";
        fi
        ;;
    stop)
        if [ ! -f $PIDFILE ]
        then
                echo "$PIDFILE does not exist, process is not running"
        else
                PID=$(cat $PIDFILE)
                echo "Stopping ..."
                $CLIEXEC -p $SENTINELPORT shutdown
                while [ -x /proc/${PID} ]
                do
                    echo "Waiting for Sentinel to shutdown ..."
                    sleep 1
                done
                rm -rf $PIDFILE
                echo "Sentinel stopped"
        fi
        ;;
    *)
        echo "Please use start or stop as first argument"
        ;;
esac
```

Change the permissions of the newly created file:

```
chmod 755 /etc/init.d/sentinel
```

Then run systemctl daemon-reload to reload the files.

Start the sentinel service:

```
service sentinel start
```

And Check if the service started successfully:

```
service sentinel status
```

Having a correct configuration the connection should be active

```
root@Redis-3:/home/pr_user# service sentinel status
● sentinel.service - LSB: start and stop sentinel
   Loaded: loaded (/etc/init.d/sentinel; bad; vendor preset: enabled)
   Active: active (running) since Thu 2019-04-25 18:11:09 EEST; 2s ago
     Docs: man:systemd-sysv-generator(8)
  Process: 1516 ExecStop=/etc/init.d/sentinel stop (code=exited, status=0/SUCCESS)
  Process: 1523 ExecStart=/etc/init.d/sentinel start (code=exited, status=0/SUCCESS)
    Tasks: 4
   Memory: 1.2M
      CPU: 9ms
   CGroup: /system.slice/sentinel.service
           └─1525 /usr/local/bin/redis-sentinel 192.168.10.151:26379 [sentinel
```

Next, check if the sentinels work properly using one of the two commands:

```
redis-sentinel /etc/redis/sentinel.conf
```

```
redis-server /etc/redis/sentinel.conf --sentinel
```

Please note, that it is mandatory to use a configuration file when running Sentinel, as this file will be used by the system to save the current state that will be reloaded in case of restarts.

Sentinels will simply refuse to start if no configuration file is given or if the configuration file path is not writable.

Sentinels by default run listening for connections to TCP port 26379, so for Sentinels to work, port 26379 of your servers must be open to receive connections from the IP addresses of the other Sentinel instances. Otherwise Sentinels can't talk and can't agree about what to do, so failover will never be performed.

For further information, please read thoroughly Redis' official Sentinel documentation:
https://redis.io/topics/sentinel

Login to the sentinel by typing the following command:

```
redis-cli -p 26379
```

All the information of the local sentinel should appear.

```
● sentinel.service - LSB: start and stop sentinel
   Loaded: loaded (/etc/init.d/sentinel; bad; vendor preset: enabled)
   Active: active (running) since Thu 2019-04-25 18:11:09 EEST; 2s ago
     Docs: man:systemd-sysv-generator(8)
  Process: 1516 ExecStop=/etc/init.d/sentinel stop (code=exited, status=0/SUCCESS)
  Process: 1523 ExecStart=/etc/init.d/sentinel start (code=exited, status=0/SUCCESS)
    Tasks: 4
   Memory: 1.2M
      CPU: 9ms
   CGroup: /system.slice/sentinel.service
           └─1525 /usr/local/bin/redis-sentinel 192.168.10.151:26379 [sentinel

Apr 25 18:11:09 Redis-3 systemd[1]: Starting LSB: start and stop sentinel...
Apr 25 18:11:09 Redis-3 sentinel[1523]: Starting Sentinel server...
Apr 25 18:11:09 Redis-3 systemd[1]: Started LSB: start and stop sentinel.
root@Redis-3:/home/pr_user# redis-cli -p 26379
127.0.0.1:26379> sentinel master mymaster
 1) "name"
 2) "mymaster"
 3) "ip"
 4) "192.168.10.214"
 5) "port"
 6) "6379"
 7) "runid"
 8) "b7824e6996391b991b9f8808bea7948194e08a40"
 9) "flags"
10) "master"
11) "link-pending-commands"
12) "0"
13) "link-refcount"
14) "1"
15) "last-ping-sent"
16) "0"
17) "last-ok-ping-reply"
18) "72"
19) "last-ping-reply"
20) "72"
21) "down-after-milliseconds"
22) "2000"
23) "info-refresh"
24) "7728"
25) "role-reported"
26) "master"
27) "role-reported-time"
28) "108066"
29) "config-epoch"
30) "0"
31) "num-slaves"
32) "1"
33) "num-other-sentinels"
34) "0"
35) "quorum"
36) "2"
37) "failover-timeout"
38) "10000"
39) "parallel-syncs"
40) "1"
127.0.0.1:26379>
```

1. The sentinel is running on the Local IP address and the port configured earlier.
2. The name of the master Redis node.
3. The IP of the master Redis node.
4. The port of the master Redis node.