1. Installing and Running VMware Player
2. VM installation and configure (Ubuntu /Debian).
3. Redis installation in Ubuntu / Debian .
4. Redis configuration standalone
5. Radis Configuring High Availability/ Redise replication (Master /Salve)

1.Installing and Running VMware Player

To install VMware Player on a Windows host

1 If you are installing VMware Player from a CD and autorun is enabled, follow the instructions displayed

when you insert the CD in your CD‐ROM drive.

If you are installing VMware Player from a CD and autorun is not enabled, double‐click the

VMware-player-<xxxx-xxxx>.exe installer file in the Windows directory of the CD. (In the filename,

<xxxx-xxxx> is a series of numbers representing the version and build numbers.)

2 On the Welcome page, click Next.

3 On the Destination Folder page, optionally choose an alternate location for VMware Player files, and click

Next.

4 Optionally, deselect any shortcuts on the Configuration Shortcuts page, and click Next.

5 Click Install to begin the installation.

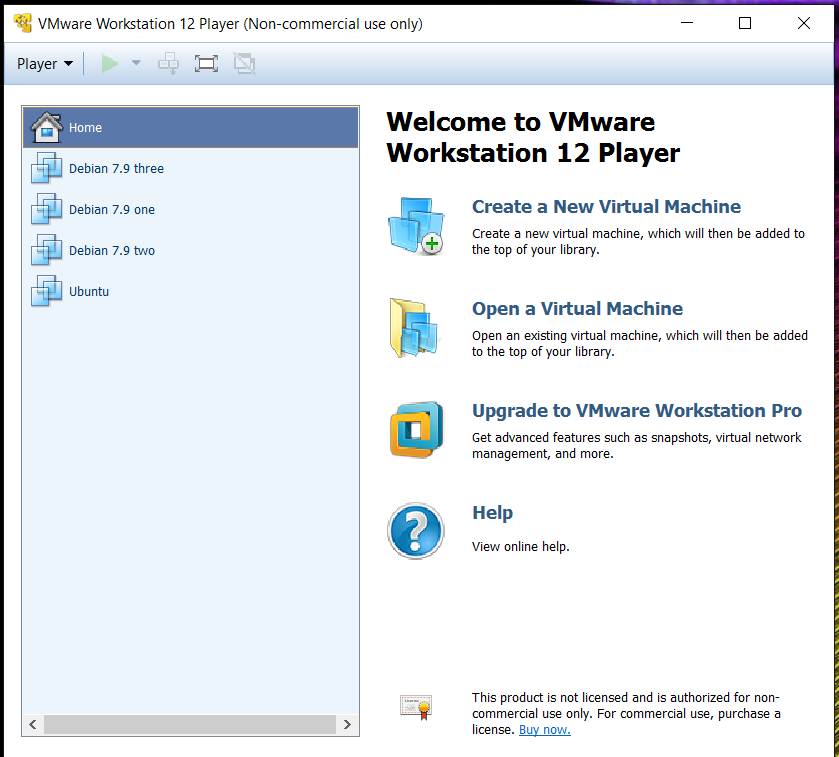
VMware Player and any shortcuts you selected are installed on your host machine.

6 Click Finish.

2. Installation and configure (Ubuntu /Debian) in VM Player .

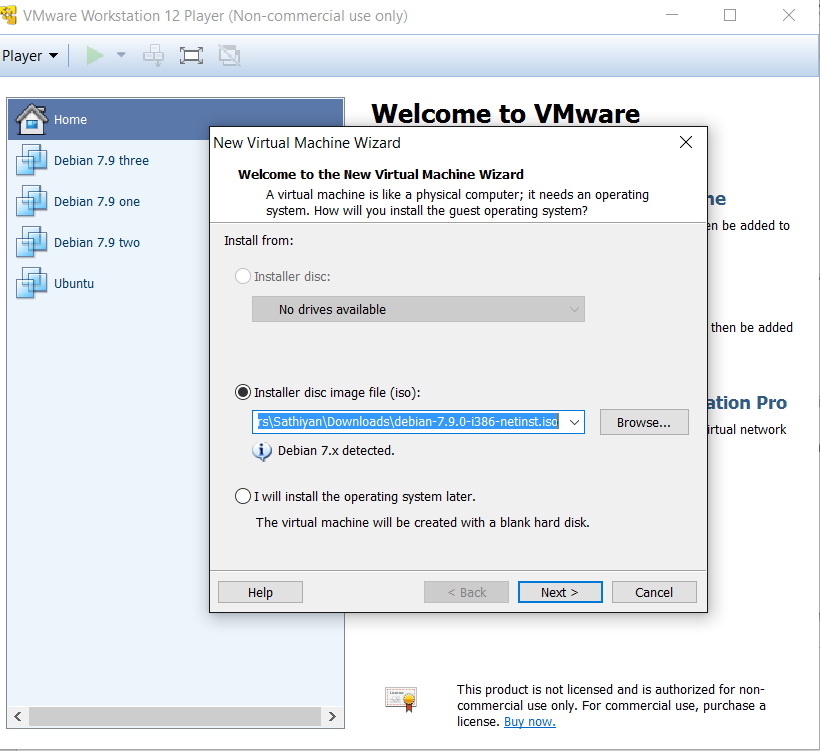
To install a new virtual machine in VMware Player and VMware Workstation Player:

1. Open VMware Player/VMware Workstation Player.

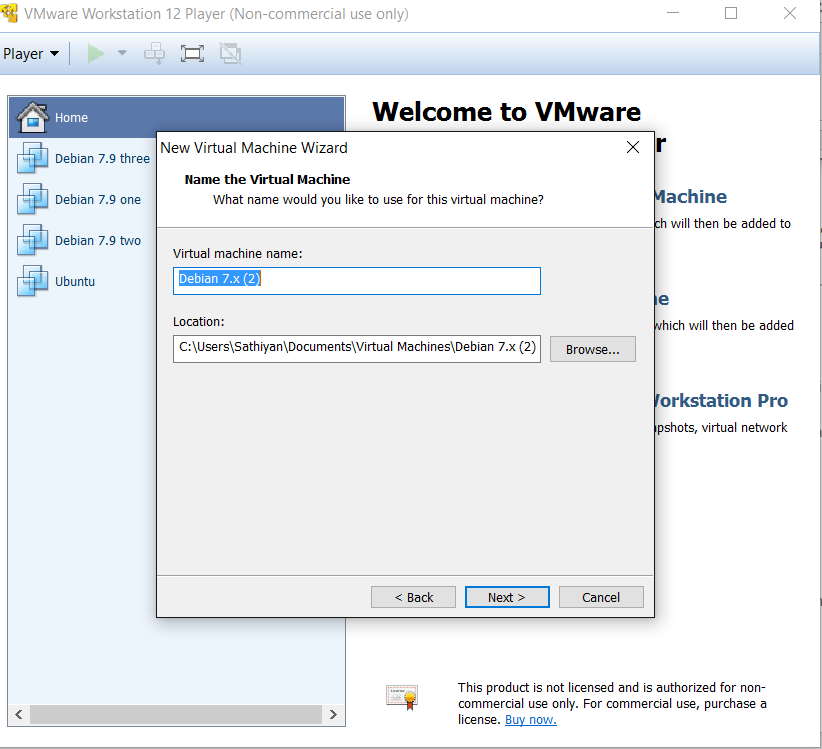


1. Insert your operating system install disc into your CD/DVD drive.  
     
   **Note**: If you do not have an install disc, use a disc image file ( .iso) you can download in respective site .Ubuntu : <http://www.ubuntu.com/download/ubuntu-kylin>. And Debian : <https://www.debian.org/releases/wheezy/debian-installer/>
2. For Player 4.x and below: Click **File** > **Create a New Virtual Machine**.  
   For Player 5.x and above: Click **Player > File > New Virtual Machine**.

The New Virtual Machine Wizard opens and recognizes the operating system on the install disc or image file.

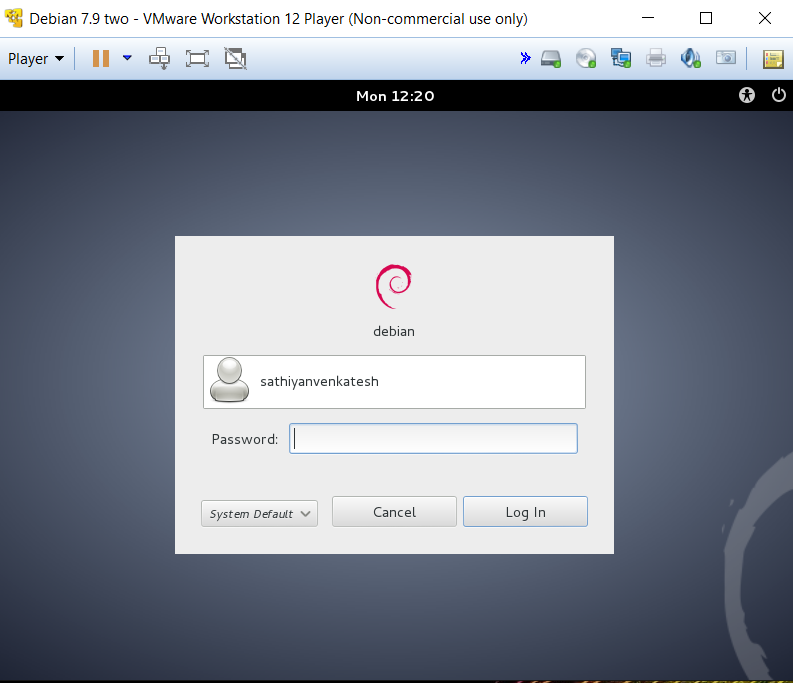


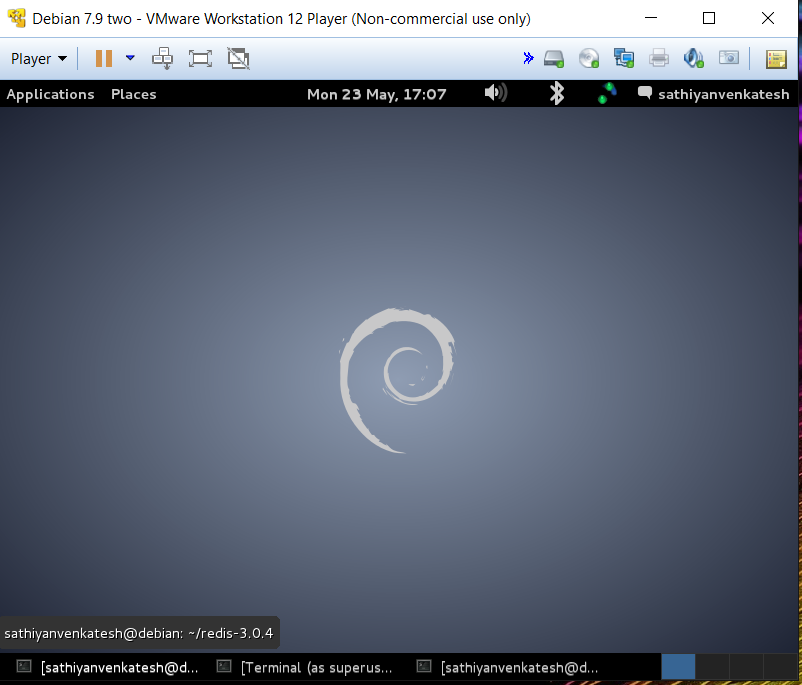
1. Click **Next**.
2. Enter the required information, such as user name, password, and license key (if required). Click **Next**.



1. Provide a name for the virtual machine and choose a location to save. Click **Next**
2. Adjust the disk size as required and choose either to store the file as a single file or to split it into multiple files. Click **Next**.
3. To make changes to the virtual hardware, click **Customize Hardware** and modify as required.  
     
   **Note**: You can also retain the default values.
4. Click **Finish**.  
     
   VMware Player starts installing your new virtual machine. The screen flickers a few times as the virtual machine restarts to complete the installation. After the operating system installation completes, VMware Player continues with the installation of VMware Tools inside the virtual machine. After the VMware Tools installation completes, use your newly configured virtual machine.

10.After booting you will see the screen like below





3.Reise installation in Ubuntu / Debian: .

### **What is Redis?**

Redis is a flexible, open-source, key value data store. Following in the footsteps of other NoSQL databases, such as Cassandra, CouchDB, and MongoDB, Redis allows the user to store vast amounts of data without the limits of a relational database. Additionally, it has also been compared to memcache and can be used, with its basic elements as a cache with persistence.

### **Set-up Environment For Installation.**

Before installing redis, there are a couple of prerequisites that need to be downloaded in-order to proceed with the installation.

**Update all of the apt-get packages:**

$>Sudo apt-get update

Download a compiler with build essential in-order to install Redis from source.:

$>sudo apt-get install build-essential

Install Redis

After all of the prerequisites and dependencies downloaded to the server, you can proceed with the Redis installation from source.

1. Download the Stable Redis release from their website. google code. The latest stable version is 3.0.4

$>wget http://download.redis.io/releases/redis-3.0.4.tar.gz

2. Extract the downloaded .tar file, and switch into that directory.

$ tar xzf redis-3.0.4.tar.gz

$ cd redis-3.0.4

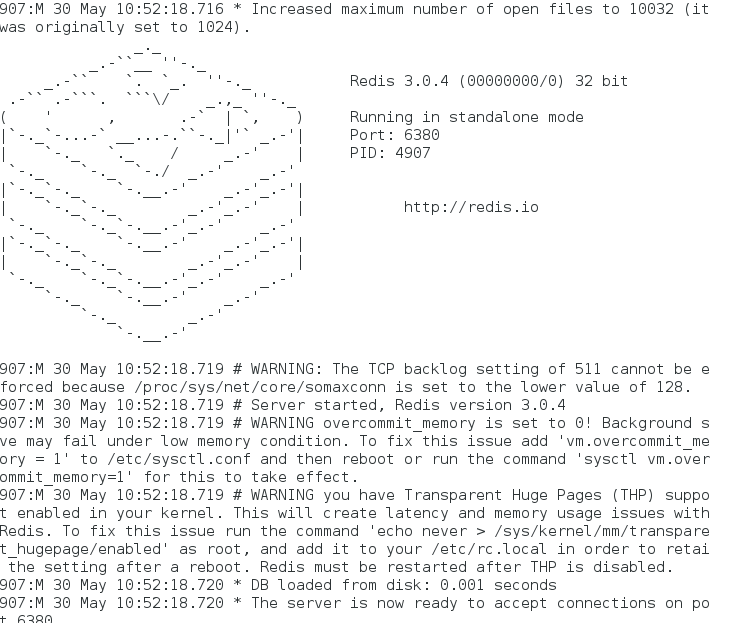
$ make

4.Redis configuration standalone.

For standalone default port is 6379 and ip 127.0.0.1

**Running server:**

$>src/redis-server



Running client :

$>src/redis-cli -h <localhost>

Ip:port>

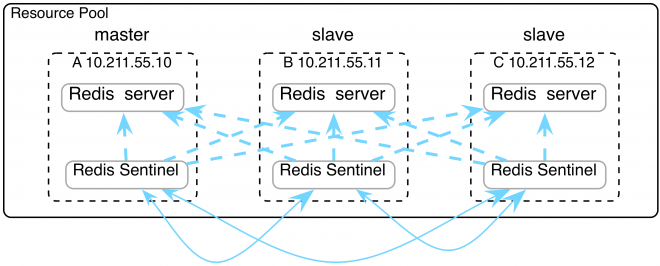
To test type

127.0.0.1:6379>info

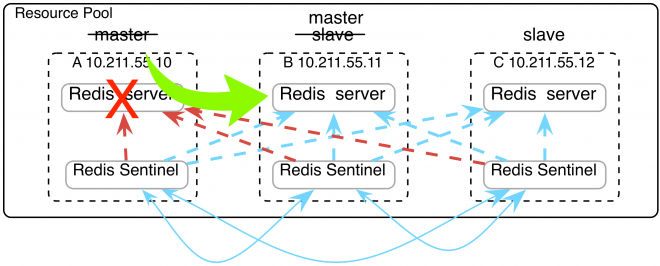
Information u can see about the server.

1. Radis Configuring High Availability/ Redise replication (Master /Salve)

This article describes how to configure active-passive high availability(HA) for a group of Redis instances. I avoid using cluster because [Redis Cluster](http://redis.io/topics/cluster-tutorial) is the mechanism to distribute shards to multiple Redis instances . Active-passive means there is only one Redis instance, the master, serving requests at all time. Data is replicated from the master to the rest of the Redis instances, the slaves. One of the slaves would replace the master when the master is down. Therefore HA essentially needs to take care of two major issues, data availability and service availability. Redis replication provides data availability.  Redis [Sentinel](http://redis.io/topics/sentinel) ensures service availability. Redis Sentinel monitors Redis instances. For example, consider a group of three Redis instances, A, B, and C, with A as the initial master. The following figure illustrate how Redis Sentinel monitors Redis instances.



When the master is down, Redis Sentinel automatically elects a new master and failover to the the new master as shown in the following figure.



Host IP and ports.

Machin A :

Ip:192.168.139.131 Redis port:6380 Sentinel port :26380

Machin B:

IP: 192.168.139.229 Redis port :6379 Sentinel port :26379

Machin C:

**Open Machin A:**

Open 4 terminals

**Terminal 1- to run redis server :**

$>cd redis-server

Redis-server>sudo src/redis-server redis.conf

**Terminal 2-To run redis client**

$>cd redis-server

Redis-server>sudo src/redis-cli -h<ip> -p<port>

Ip:port>info

**Terminal 3-To run Sentinel server**

Configure sentinel:

Enable and do the below changes in sentinel.conf file

sentinel monitor mymaster 10.211.55.10 6379 2

sentinel down-after-milliseconds mymaster 60000

sentinel failover-timeout mymaster 180000

sentinel parallel-syncs mymaster 1

and run below command

$>cd redis-server

Redis-server>sudo src/redis-server sentinel.conf –sentinel

Terminal 4 – To run sentinel monitor

$>cd redis-server

Redis-server>sudo src/redis-cli -p <sentinel port>

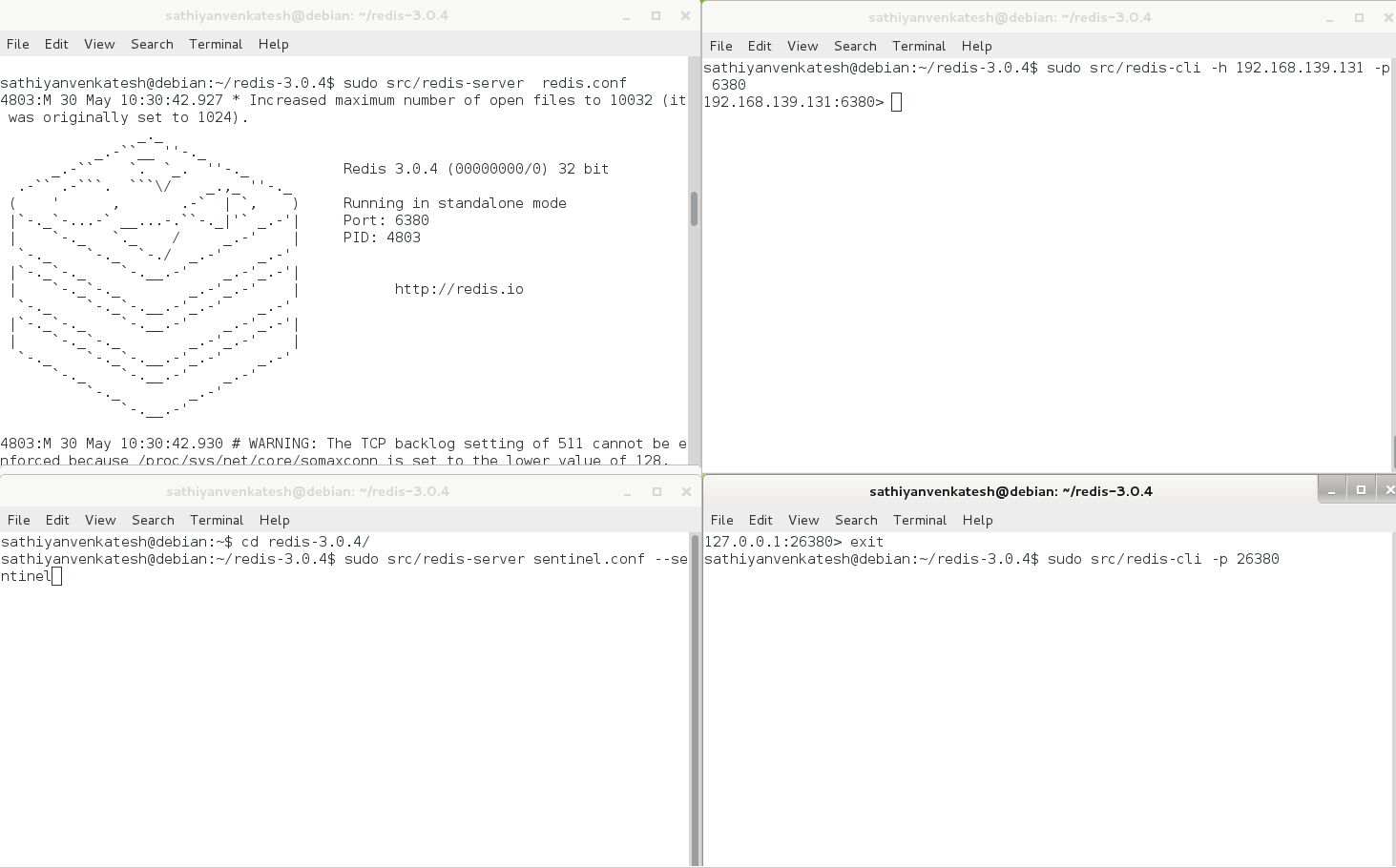
Eg:sudo src/redis-cli -p 26379

**Note: open 4 terminals in each Machin and repeat the above steps**

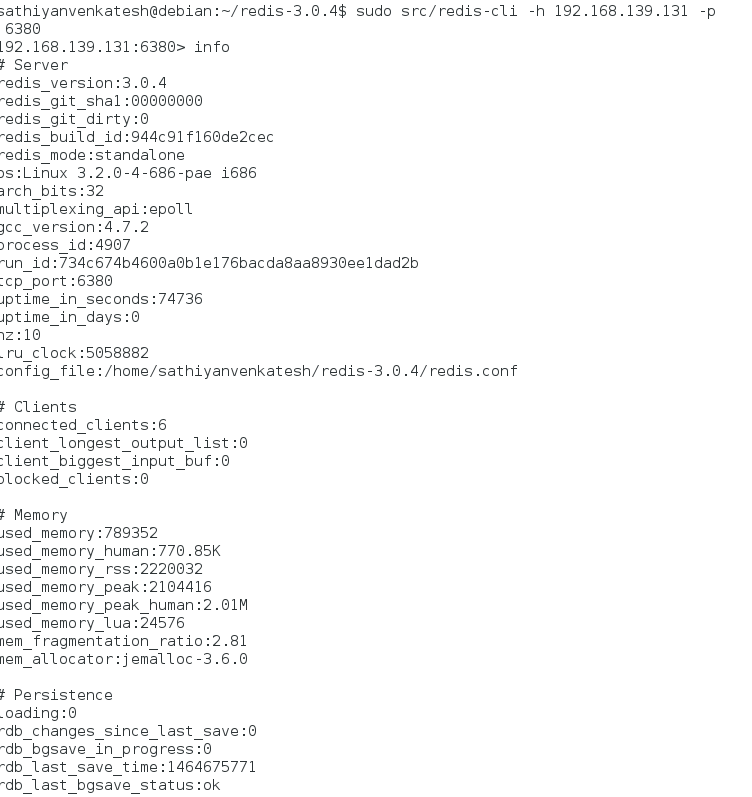
**Reference Link:**

[**http://redis.io/download**](http://redis.io/download)

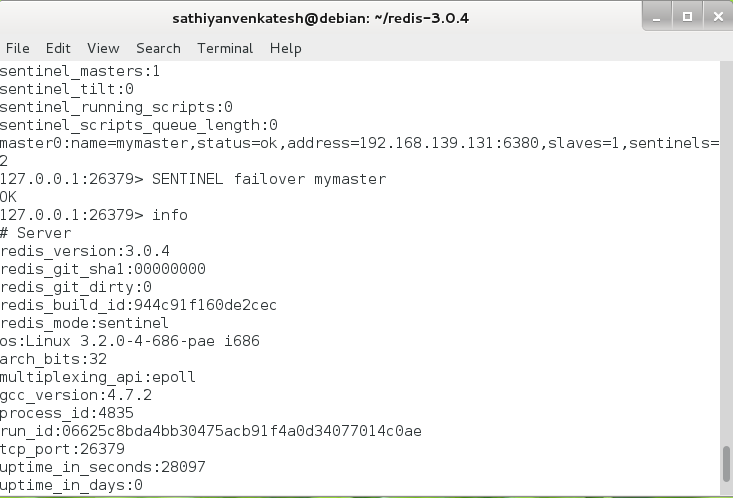
[**http://avenshteinohad.blogspot.ae/2016/01/redis-jedis-quickstart.html**](http://avenshteinohad.blogspot.ae/2016/01/redis-jedis-quickstart.html)

**To start redis server – client and sentinel server – client**  

To check redis client:



To check sentinel client





### **Redis & Jedis Quickstart**

To get started, you can download redis installation from:

<http://redis.io/download>

If you use windows, use MSOpenTech version from:

<https://github.com/MSOpenTech/redis>

Jedis (Redis java provider):

<https://github.com/xetorthio/jedis>

### Connect to redis server

redis-cli -h [host] -p [port]

(default port is 6379)

### Useful commands

INFO - Prints general details about the current server

GET [Key] - Gets a value of a key

SET [Key] [Value] - Sets a value of a key

SETEX [Key] [Seconds] [Value] – Sets key & value with expiration

RENAME [OldKeyName] [NewKeyName] - Renames a key

DEL [Key] [Key] .... - Removes given keys and returns the number of keys removes

APPEND [Key] [Value] – Append a value to a key

EXISTS [Key] - Returns 1 if the given key exists, 0 if not.

KEYS [regex] - Prints all keys matches a given regex pattern

KEYS \* - Prints all keys

KEYS hello\* - Prints all keys that starts with "hello"

TTL [Key] – Gets the remaining time of a key to live.

TIME  [Key] – Returns the time of the current server.

DBSIZE - Number of keys in db

FLUSHALL - Removes all keys & values

SLAVEOF [host] - Creates a slave to a master

SLAVEOF NO ONE - Promotes slave to master

ROLE - Tells whether current server is master or slave

CLUSTER INFO - Prints details regarding the redis cluster parameters like state, size etc.

CLUSTER NODES - Prints details regarding servers in the cluster

CONFIG GET  \* - Prints all configuration file data

CONFIG SET [Key] [Value] - Sets a value to a config key (Can be use in runtime)

CLIENT SETNAME [This client name] – Assigns a name to the current connection

CLIENT GETNAME – Gets the current connection name

CLIENT LIST - Prints information about client connections to the current server

QUIT - Closes connections to the server

### SENTINEL:

"Redis Sentinel is a distributed system:

Sentinel itself is designed to run in a configuration where there are multiple

Sentinel processes cooperating together. The advantages of having multiple

Sentinel processes cooperating are the following:

1. Failure detection is performed when multiple Sentinels agree about the

fact a given master is no longer available. This lowers the probability of

false positives.

2. Sentinel works even if not all the Sentinel processes are working,

making the system robust against failures. There is no fun in having a

fail over system which is itself a single point of failure, after all."

(Taken from official redis documentation <http://redis.io/topics/sentinel>)

**Starting a redis server in sentinel mode:**

redis-server /path/to/sentinel.conf --sentinel

**Setting a sentinel option:**

sentinel <option\_name> <master\_name> <option\_value>

**Setting a sentinel master:**

sentinel monitor <master-group-name> <ip> <port> <quorum>

**All these configuration should be placed in "sentinel.conf":**

sentinel monitor mymaster 127.0.0.1 6379 2

sentinel down-after-milliseconds mymaster 60000

sentinel failover-timeout mymaster 180000

sentinel parallel-syncs mymaster 1

***First config row -*** Sentinel configuration INCLUDING master (slaves are auto-discovered).

The last value ("2") is the quorum number (number of Sentinels that need to agree about the fact the master is not reachable).

***Last config row -***parallel-syncs is the number of slaves which will participate in configuring the new master after failover.

(Bigger number = Faster sync, but if the data is old in some of them, it might be written to the new master)

### Jedis Java Provider

**Common Pool Constructors:**

JedisPool(Config poolConfig, String host)

JedisPool(Config poolConfig, String host, int host)

JedisSentinelPool(String masterName, Set<String> sentinels)

***Parameters:***

*masterName* – Name of the sentinel master given in "sentinel monitor" line in

"sentinel.conf". In the example above it's "mymaster".

*sentinels*- Entire set of IP addresses of all sentinels (Master + Slaves).

*For Example:*

HashSet<String> sentinels = new HashSet<String>();

sentinels.add("127.0.0.1:1234");

JedisSentinelPool jedisPool = new JedisSentinelPool("mymaster", sentinels);

**Get key example:**

Jedis jedis = jedisPool.getResource();

String result = jedis.get(key);

jedisPool.returnResource(jedis);

**Set key example:**

Jedis jedis = jedisPool.getResource();

jedis.setex(key,secondsTillExpired, value); // setting a key and a value with expiration

jedisPool.returnResource(jedis);