# Redis Sentinel

Sentinel是Redis的一种高可用方案。作用是通过一个或者多个Sentinel实例组成的Sentinel集群来监控多个Redis主备服务器。当Redis主服务器因为异常无法正常提供服务的情况下，将某个从服务器升级为主服务器，自动完成主备切换。

## 常用配置

sentinel monitor <master> <ip> <port> <quorum>

配置sentinel需要监控的master信息。

其中quorum就是指需要至少多少个sentinel判定该master为主观下线，才能判定该master为客观下线。

sentinel down-after-milliseconds <master> <time>

配置sentinel监控的master多长时间没有正常响应sentinel的PING消息，就被判定为主观下线。

sentinel parallel-syncs <master> <time>

配置当出现故障转移的时候，同一时间能有多少个redis从服务器向redis主服务器发起同步。配置为1表示每次只有1个redis从服务器处于不能处理命令请求的状态。

## 关键数据结构

### sentinelRedisInstance

typedef struct sentinelRedisInstance {

int flags;

redisAsyncContext \*\*cc;

redisAsyncContext \*\*pc;

dict \*slave;

dict \*sentinels;

…

}sentinelRedisInstance;

**flags：**标记服务器类型的标记位，SRI\_MASTER表示主服务器，SRI\_SLAVE表示从服务器，SRI\_SENTINEL表示sentinel服务器。

**cc**：指向命令连接的指针。

**pc**：指向订阅连接的指针。

**slave**：该redis实例对应的slave redis的字典。

**sentinels**：监视当前redis的所有sentinel的字典。

## 一、Sentinel启动

### 1、启动命令

redis-sentinel /…/…/sentinel.conf 或

redis /…/…/sentinel.conf –sentinel

### 创建与Redis主服务器的连接

sentinel.conf配置文件中需要指定sentinel监控的redis主服务器信息。对于每个需要被监控的redis主服务器，sentinel会与之建立两条连接，一条命令连接，一条订阅连接，订阅redis主服务器的\_\_sentinel\_\_:hello频道。

#### 关键函数

##### sentinelReconectInstance

入参sentinelRedisInstance：需要建立连接的redis实例。

出参：无。

## Sentinel与Redis通信

### INFO命令

Sentinel默认每10s通过命令连接向被监视的redis主服务发送INFO命令。通过redis主服务器INFO命令的应答获取以下信息：

1. redis主服务器的run\_id。
2. redis主服务器对应的从服务器列表。
3. 每个从服务器的ip、port、状态(online/offline)、复制偏移量等。

获取到的从服务器信息会加入主服务器的slave字典中，如果存在则是更新。对于新加入的从服务器，sentinel也会与新加入的从服务器建立命令连接和订阅连接。主服务器实例中的name字段来源于配置，而从服务器实例中的name字段是根据从服务器的ip、port设置的。

这也是为什么sentinel的配置文件只是指定了需要监控的主服务器而没有指定需要监控的从服务器。

与从服务器建立命令连接和订阅连接之后，sentinel也会向从服务器发送INFO命令。频率与主服务器一致。从服务器INFO命令获取以下信息：

1. 从服务器run\_id。
2. 从服务器所属的主服务器ip、port、link\_status、优先级、复制偏移量等。

#### 关键函数

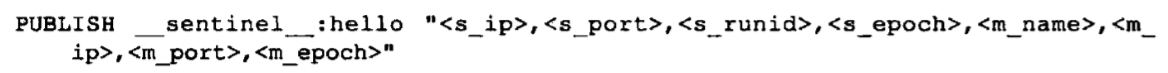
##### sentinelRefreshInstanceInfo

入参\*ri：redis实例，不一定是redis主服务器的实例，也有可能是redis从服务器的实例。

入参\*info：需要被解析的INFO命令回复的数据。

### PUBLISH命令

Sentinel每2s通过命令连接向所有主从服务器发送publish命令，格式如下：



s\_开头的参数表示sentinel本身的信息。m\_开头的参数表示主服务器的信息(如果是发送给从服务器的，那么就是其对应的主服务器信息)。

#### 关键函数

### SUBSCRIBE命令

Sentinel通过订阅连接向所有主从服务器发送subscribe命令，格式如下：



相当于sentinel向所有主从服务器订阅了\_\_sentinel\_\_:hello这个频道的信息。这样结合上面的PUBLISH命令可知，sentinel既订阅了hello频道，又会给hello发送消息。这样相当于一个sentinel发送的publish消息会被其他sentinel收到。这样相当于每个sentinel都知道了其他sentinel的信息。

当某个sentinel收到了其他sentinel的信息之后，会更新sentinelRedisInstance结构中sentinels字典保存的内容。sentinels字典的key是sentinel的ip:port的键值对，也就是sentinel的名字。sentinel如果收到了自己发送的消息，则会忽略该消息。

所以sentinel的集群不是通过配置指定的，而是通过这种方式做到互相发现的。

#### 关键函数

## Sentinel与Sentinel通信

sentinel通过上述方式互相发现之后，sentinel会与其他sentinel创建一个命令连接。sentinel与sentinel之间不会创建订阅连接。

### 主观下线状态

sentinel默认每1s通过命令连接向其他所有实例（包括redis主从服务器，其他sentinel）发送PING命令，根据回复来判断其他实例是否在线。如果判断实例不在线，则称为**主观下线**。

配置down-after-milliseconds指定了判断实例主观下线的时间长度。如果一个实例在配置时间内没有回复或连续发送无效回复，则sentinel判定该实例主观下线。由于时间是通过配置制定的，那么就存在多个sentinel配置了不同的主观下线时间长度。这种情况，每个sentinel都按照自己配置的主观下线时间判定主观下线，不会受到其他sentinel判定的影响。

对于PING命令的有效回复为+PONG、-LOADING、-MASTERDOWN三种。除此之外的其他回复或者不回复都为无效回复。

### 客观下线状态

当某个sentinel判定某个实例为主观下线后，就会向其他sentinel发起询问，看是否也认为该实例为主观下线，如果得到了足够数量的sentinel认可，则判定该服务器为客观下线。



该命令用于询问其他sentinel是否判定ip:port所指向的redis主服务器主观下线。该消息的应答包括三个参数：down\_state（1表示下线，0表示未下线）、leader\_runid、leader\_epoch。

收到应答的sentinel要判断有多少个其他sentinel判定redis主服务器主观下线，当这个数量大于等于sentinel monitor master 中指定的数量时，就判定该redis主服务器客观下线。此时就要启动故障转移的过程。

### 选举Sentinel

故障转移过程需要一个固定的sentinel来执行，因此首先要选举一个sentinel。选举过程如下：

1. 每个监视该redis主服务器的sentinel都可以被选举为leader sentinel，每次选举不一定能有结果，但不论选举成功与否，所有sentinel的epoch都会自增一次（epoch本质就是一个计数器，确保所有的sentinel都在做同一件事情）。
2. 每个epoch周期内，只能有一个sentinel被设置为leader sentinel，即选举一旦完成在这个epoch周期之内就不能更改。
3. 通过上面说的SENTINEL is-master-down-by-addr进行leader sentinel的选举，最后一个参数runid如果不是\*，而是sentinel的runid时，表示要求将该sentinel选举为leader sentinel。
4. 一个epoch周期内可能会有多个sentinel发送这个命令，但是对于其他接收者来说，先接到哪一个sentinel发送的命令，就将命令中runid指定的sentinel设置为leader sentinel，后来再介绍这个命令携带的runid会被忽略。
5. 上面应答消息中的leader\_runid和leader\_epoch则说明了接收者在leader\_epoch这个周期内，选择了leader\_runid所对应的sentinel为leader sentinel。
6. 当超过半数以上的sentinel都选择了某个sentinel，那么该sentinel就为leader sentinel。
7. 如果在一定的时间范围内，没有哪个sentinel获得了半数以上的投票，那么间隔一段时间之后会再次开始选举。

#### 关键函数

### 故障转移

故障转移分为三步：

1. 在判定客观下线的主服务器的所有从服务器中，选择一个称为新的主服务器。选择从服务器为新主服务器的过程如下：
2. 删除所有下线、断线或者5s内没有回复过leader sentinel发送的INFO消息的从服务器。
3. 删除所有与原主服务器断开超过10 \* down-after-milliseconds的从服务器，确保从服务器中的数据都比较新。
4. 根据剩余从服务器优先级，选取优先级最高的从服务器。
5. 优先级相同，则选择复制偏移量最大的从服务器。
6. 优先级相同，复制偏移量相同，则选择runid最小的。
7. 选中之后，向该从服务器发送SALVEOF no one的命令，然后leader sentinel每1s一次向选中的从服务器发送INFO的命令，并根据应答中role字段的变化，来判断从服务器是否顺利变为主服务器。
8. 将其他从服务器的主服务器设定为新的主服务器。上一步完成之后，leader sentinel向从服务器发送SALVEOF ip port，ip和port对应新的主服务器。
9. 将已下线的主服务器设置为新的主服务器的从服务器。这一步属于sentinel内部的操作，改变饿了旧主服务器的数据结构。当旧主服务器再次上线的时候，sentinel就会向其发送SALVEOF命令，将其变为新主服务器的从服务器。

#### 关键函数