Redis SENTINEL Introduction

Sentinel 센티널 시작하기

센티널이란

- 운영환경에서 레디스는 일반적으로 마스터와 복제로 구성됩니다. 운영중 예기치 않게 마스터가 다운되었다면, 관리자가 이를 감지해서 복제를 마스터로 올리고 클라이언트들이 새로운 마스터에 접속할 수 있도록 해 주어야 합니다. 센티널은 마스터와 복제를 감시하고 있다가 마스터가 다운되면 이를 감지해서 관리자의 개입없이 자동으로 복제를 마스터로 올려줍니다.
- 센티널은 다음과 같은 기능을 합니다.
  - **모니터링 Monitoring** : 센티널은 레디스 마스터, 복제들을 제대로 동작하는지 지속적으로 감시합니다.
  - **자동 장애조치 Automatic Failover** : 센티널은 레디스 마스터가 예기치 않게 다운되었을 때 복제를 새로운 마스터로 승격시켜 줍니다. 그리고 복제가 여러 대 있을 경우 이 복제들이 새로운 마스터로 부터 데이터를 받을 수 있도록 재 구성하고, 다운된 마스터가 재 시작했을 때 복제로 전환되어 새로운 마스터를 바라볼 수 있도록 합니다.
  - **알림 Notification** : 센티널은 감시하고 있는 레디스 인스턴스들이 failover 되었을 때 Pub/Sub으로 Application(client)에게 알리거나 shell script로 관리자에게 이메일이나 SMS로 알릴 수 있습니다.

준비

- 이 데스트는 레디스 버전 4.0에서 하였고, 아래 일부분 3.2.0의 버그가 표시되어 있습니다.
- 테스트 환경은 같은 머신에서 운영하는 것으로 가정하고 괄호안에 포트를 표시했습니다.

  마스터(6382), 복제(6383), 센티널(5000)

- 마스터 구성 설정: 센티널을 위해서 따로 구성할 사항은 없습니다.
- 복제 구성 설정: 센티널을 위해서 따로 구성할 사항은 없습니다. 다만 복제로 구성하여야 합니다. redis.conf 에 다음 내용을 입력하세요. 복제 replication 구성 설정은 여기를 참조하세요.

  ```
  127.0.0.1:6383> slaveof 127.0.0.1 6382
  127.0.0.1:6383> replicaof 127.0.0.1 6382 (5.0부터)
  ```

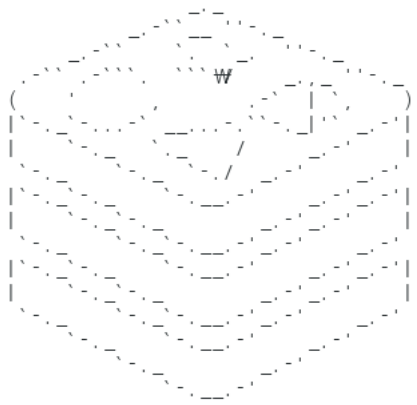
- 센티널 구성 설정: sentinel.conf에 다음 사항을 수정합니다.
  - **port 5000** : default는 26379, 여기서는 5000으로 설정합니다.
  - **sentinel monitor mymaster 127.0.0.1 6382 1** : IP와 port 그리고 마지막 숫자를 1로 입력합니다. IP와 port는 감시할 마스터의 주소입니다. 마스터 주소만 설정하면 됩니다. 복제는 마스터에서 정보를 가져옵니다. 마지막 숫자(default 2)는 quorum(의사진행에 필요한 정족수)이라고 합니다. 예를 들어 센티널이 3대 일때 quorum이 2이면 2대 이상의 센티널에서 해당 레디스 마스터가 다운되었다고 인식하면 장애조치(failover) 작업을 진행합니다. 테스트 환경에서는 센티널을 1대로 했기 때문에 quorum을 1로 설정합니다. 이 명령은 나중에 센티널에 접속해서 실행해도 됩니다. 반드시 sentinel.conf에 있을 필요는 없습니다.

- **sentinel down-after-milliseconds mymaster 3000** : 단위는 millisecond 입니다. 이 값은 마스터가 다운되었다고 인지하는 시간입니다. 즉, 마스터 서버에 정기적으로 PING을 보내는데, 이 시간 동안 응답이 없으면 다운된 것으로 판단하고 장애조치(failover) 작업을 시작합니다. default는 30초로 상당히 긴 편입니다. 여기서는 3초에서 5초 사이로 설정합니다.
- 다른 파라미터는 수정할 필요없이 그대로 진행하면 됩니다.

시작하기

- 레디스 마스터와 복제를 시작합니다. 마지막으로 센티널을 시작합니다. 시작 화면은 레디스와 유사합니다.

```
$ src/redis-sentinel   sentinel.conf
```



```
Redis 4.0.6 (00000000/0) 64 bit

Running in sentinel mode
Port: 5000
PID: 20659

      http://redis.io
```

```
20659:X 02 Jan 09:54:28.723 # Sentinel ID is 1628154ec050e3ba3bfdf1e598eaa2d95851214e
20659:X 02 Jan 09:54:28.723 # +monitor master mymaster 127.0.0.1 6382 quorum 1
20659:X 02 Jan 09:54:28.726 * +slave slave 127.0.0.1:6383 127.0.0.1 6383 @ mymaster
127.0.0.1 6382
```

- 이제 센티널은 레디스 노드들을 감시하기 시작합니다.
- 센티널 메시지를 간단하게 살펴 봅시다.
  20659 : process id 입니다. Port 다음 줄에 있는 PID와 동일합니다.
  X : Sentinel을 나타냅니다. 레디스 마스터일때는 M, 복제일때는 S, 마스터나 복제에서 RDB나 AOF를 Rewrite하기 위해 자식 프로세스를 생성하면 C로 표시됩니다.
- 센티널 클라이언트는 레디스 클라이언트와 동일합니다. 포트만 센티널 포트를 지정하면됩니다.
  INFO sentinel 명령으로 센티널 상태와 모니터링하는 마스터 정보를 조회합니다.

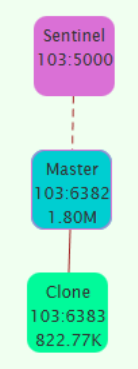
```
$ src/redis-cli -p 5000
127.0.0.1:5000> info sentinel
# Sentinel
sentinel_masters:1
sentinel_tilt:0
sentinel_running_scripts:0
sentinel_scripts_queue_length:0
sentinel_simulate_failure_flags:0
master0:name=mymaster,status=ok,address=127.0.0.1:6382,slaves=1,sentinels=1
```

- 추가로 아래처럼 환경에 맞게 몇 개 파라미터를 수정하거나 추가한 후 센티널을 재 시작합니다.
  bind 192.168.56.103 127.0.0.1
  protected-mode yes
  dir "/home/redis/redis-4.0.6/5000"
  logfile "sentinel.log"
  daemonize yes

장애조치(failover)

Master -> Slave1 테스트

- 준비 상태



- 레디스 마스터를 중지시키고 센티널이 복제를 마스터로 승격시키는지 봅시다.
  아래는 레디스 마스터를 중지시켰을때 센티널에서 나오는 메시지입니다. 중지시키는 방법은 레디스 클라이언트에서 shutdown을 입력합니다. 이 메시지는 중지 후 3초 이후에 나오기 시작합니다. 이유는 sentinel.conf 에 입력한 'sentinel down-after-milliseconds mymaster 3000'에 따라서 3초간 마스터 서버로 부터 응답을 받지 못하면 서버 다운을 판단하고 장애조치(failover)가 진행되기 때문입니다.

    12745:X 22:51:10.590 # +**sdown** master mymaster 127.0.0.1 6382
    12745:X 22:51:10.590 # +**odown** master mymaster 127.0.0.1 6382 #quorum 1/1
    12745:X 22:51:10.590 # +new-epoch 180001
    12745:X 22:51:10.590 # +try-failover master mymaster 127.0.0.1 6382
    12745:X 22:51:10.611 # +vote-for-leader 968dbf46cf52b2ba8bdb8a641020eaeded1a1b0d 180001
    12745:X 22:51:10.611 # +elected-leader master mymaster 127.0.0.1 6382
    12745:X 22:51:10.611 # +failover-state-select-slave master mymaster 127.0.0.1 6382
    12745:X 22:51:10.677 # +selected-slave slave 127.0.0.1:6383 127.0.0.1 6383 @ mymaster 127.0.0.1 6382
    12745:X 22:51:10.677 * +failover-state-send-**slaveof-noone** slave 127.0.0.1:6383 127.0.0.1 6383 @ mymaster 127.0.0.1 6382
    12745:X 22:51:10.760 * +failover-state-wait-promotion slave 127.0.0.1:6383 127.0.0.1 6383 @ mymaster 127.0.0.1 6382
    12745:X 22:51:11.658 # +promoted-slave slave 127.0.0.1:6383 127.0.0.1 6383 @ mymaster 127.0.0.1 6382
    12745:X 22:51:11.658 # +failover-state-reconf-slaves master mymaster 127.0.0.1 6382
    12745:X 22:51:11.697 # +failover-end master mymaster 127.0.0.1 6382
    12745:X 22:51:11.697 # +**switch-master** mymaster 127.0.0.1 6382 127.0.0.1 6383
    12745:X 22:51:11.697 * +slave slave 127.0.0.1:6382 127.0.0.1 6382 @ mymaster 127.0.0.1 6383
    12745:X 22:51:14.786 # +sdown slave 127.0.0.1:6382 127.0.0.1 6382 @ mymaster 127.0.0.1 6383

    - **SDOWN** : Subjectively down(주관적 다운)이라고 합니다. 이는 센티널에서 주기적으로 마스터에 보내는 PING과 INFO 명령의 응답이 3초(down-after-milliseconds 에서 설정한 값) 동안 오지 않으면 주관적 다운으로 인지합니다. 이는 센티널 한 대에서만 판단한 것입니다. 주관적 다운만으로는 장애조치를 진행하지 않습니다.
    - **ODOWN** : Objectively down(객관적 다운)이라고 합니다. 이는 설정한 quorum 이상의 센티널에서 해당 마스터가 다운되었다고 인지하면 객관적 다운으로 인정하고 장애조치를 진행합니다.
    - 중간에 **slaveof-noone** 메시지가 현재 복제인 6383 서버에 slaveof no one 명령을 실행시켜 마스터로 승격시키는것을 나타냅니다.
    - **switch-master**는 6383 서버가 마스터로 승격이 완료되었음을 나타냅니다.
    - 마지막 라인은 6382가 복제로 전환되었고, 다운 상태라는 것을 나타냅니다. sentinel.conf 파일에 6382가 이제 복제라고 다음과 같이 기록되어 있습니다. **'sentinel known-slave mymaster 127.0.0.1 6382'**
- 이 상태에서 센티널에 접속해서 마스터를 확인해 보면 6383으로 나옵니다.

    127.0.0.1:5000> info sentinel
    # Sentinel
    sentinel_masters:1
    sentinel_tilt:0
    sentinel_running_scripts:0

sentinel_scripts_queue_length:0
sentinel_simulate_failure_flags:0
master0:name=mymaster,status=ok,address=127.0.0.1:6383,slaves=1,sentinels=1

- 이제 복제에서 마스터로 승격된 6383 서버의 메시지를 봅시다.

  12735:S 22:51:07.524 # Connection with master lost.
  12735:S 22:51:07.524 * Caching the disconnected master state.
  12735:S 22:51:08.418 * Connecting to MASTER 127.0.0.1:6382
  12735:S 22:51:08.418 * MASTER <-> SLAVE sync started
  12735:S 22:51:08.418 # Error condition on socket for SYNC: Connection refused
  12735:S 22:51:09.419 * Connecting to MASTER 127.0.0.1:6382
  12735:S 22:51:09.419 * MASTER <-> SLAVE sync started
  12735:S 22:51:09.419 # Error condition on socket for SYNC: Connection refused
  12735:S 22:51:10.421 * Connecting to MASTER 127.0.0.1:6382
  12735:S 22:51:10.421 * MASTER <-> SLAVE sync started
  12735:**S** 22:51:10.421 # Error condition on socket for SYNC: Connection refused
  12735:**M** 22:51:10.760 * Discarding previously cached master state.
  12735:M 22:51:10.760 * **MASTER MODE enabled** (user request)
  12735:M 22:51:10.761 # **CONFIG REWRITE** executed with success.

  - 첫 번째 줄은 6382가 다운되어 마스터와의 연결이 끊어진것을 나타낸다. 시간을 보면 51분 7초로 센티널의 메시지 51분 10초 보다 3초 빠르다. 이 메시지는 연결이 끊어지자 마자 바로 나온 것이고, 센티널의 메시지는 설정 값(down-after-milliseconds)에 따라 3초 늦게 나온 것이다.
  - 세번째부터 다섯번째 줄까지 세줄은 1초마다 반복적으로 나온다.
  - 3초 후에 센티널이 6383 서버를 마스터로 승격시키면 서버의 종류를 나타내는 S 가 M 으로 변경된다.
  - 마지막 라인에 **CONFIG REWRITE** 가 실행되면 6383 서버의 redis.conf 에 있는 **slaveof 127.0.0.1 6382**를 삭제한다. 그러므로 **문제 발생 후 레디스를 점검할때 꼭 redis.conf 에서 slaveof를 확인해 보야 한다**.

- 여기까지 진행된 현재 상태

  마스터(6382) -> Down
  복제(6383) -> 마스터로 승격

- 다운되어 있는 6382 서버를 다시 시작해보자.

  13744:M 23:12:16.881 # Server started, Redis version 4.0.6
  13744:M 23:12:16.882 * DB loaded from append only file: 0.000 seconds
  13744:M 23:12:16.882 * The server is now ready to accept connections on port 6382
  13744:S 23:12:26.939 * SLAVE OF 127.0.0.1:6383 enabled (user request)
  13744:S 23:12:26.940 # CONFIG REWRITE executed with success.
  13744:S 23:12:27.900 * Connecting to MASTER 127.0.0.1:6383

  - 서버가 시작하면 마스터로 수행된다. 메시지에 **M**으로 표기된 것을 보면 알 수 있다. 마스터인 상태에서 AOF나 RDB파일을 읽어 들인다. 이 데이터는 이후에 새로운 마스터로 부터 데이터를 다시 받으므로 버려진다. 그러므로 서버를 빠르게 띄우고 싶으면 데이터 디렉토리에서 해당 AOF나 RDB 파일을 다른 곳으로 옮겨놓는 것이 좋은 방법일 것이다.
  - 10초 후에 센티널에 의해서 복제로 전환된다. **M**이 **S**로 바뀌었다.
  - **CONFIG REWRITE**가 실행되어, 6382의 redis.conf에 slaveof 명령이 기록된다.
  - 새로운 마스터에 접속해서 데이터를 받아 온다.

- **주의사항** : 레디스 버전 3.2.0은 6382 서버를 재 시작했을 때 버그로 인하여 복제로 전환되지 않는다. 이는 3.2.1에서 수정되었다. 2016년 6월 17일에 발표된 3.2.1 릴리즈 노트에 있는 버그 관련 내용

  ==
  Redis 3.2.1 Released Fri Jun 17 15:01:56 CEST 2016
  ==

  Upgrade urgency HIGH: Critical fix to Redis Sentinel, due to 3.2.0 regression
  compared to 3.0.

> Hey, this is Redis 3.2.1, and this release should bring some grain of maturity to Redis 3.2. The list of commits following this note will tell you the details, but the main things addressed in this release are the following:
>
> 1. A critical bug in Sentinel was hopefully fixed. During the big 3.2 refactoring of Redis Sentinel, in order to implement connection sharing to make Sentinel able to scale better (few Sentinels to monitor many masters), a bug was introduced that mis-counted the number of pending commands in the Redis link. This in turn resulted into an inability to talk with certain Redis instances. A common result of this bug was the inability of Redis Sentinel to reconfigure back the old master, after a failover, when it is reachable again, as the slave of the new master. This was due to the inability to talk with the old master at all.
>
> https://github.com/antirez/redis/commit/6ad0371c9b4206a7a6692d50c9a301457baf9b6d

- 여기까지 진행된 현재 상태

  마스터(6382) -> Down -> 복제
  복제(6383) -> 마스터로 승격 -> 마스터

장애조치(failover) 시 주의할 사항

- **get-master-addr-by-name** : 마스터, 복제 모두 다운되었을 때, 센티널에 접속해서 마스터 서버 정보를 요청하면 이미 다운된 서버 정보를 리턴한다. 이 정보를 받은 클라이언트/어플리케이션은 이미 다운된 서버에 접속하려고 시도할 것이다. INFO sentinel 명령으로 마스터의 status를 확인해야 한다.

  마스터(6382) -> 마스터 -> Down
  복제(6383) -> Down -> Down

  127.0.0.1:5000> sentinel get-master-addr-by-name mymaster
  1) "127.0.0.1"
  2) "6382"
  127.0.0.1:5000> info sentinel
  master0:name=mymaster,**status=odown**,address=127.0.0.1:6383,slaves=1,sentinels=1

- **복제 -> 마스터 승격 안되는 경우** : 복제가 먼저 다운되고, 마스터가 다운된 다음, 복제(6383)가 시작되면 이 서버는 마스터로 전환되지 않는다. 즉, 복제의 redis.conf 에 자신이 복제로 되어 있고, 센티널도 이 서버(6383)를 복제라고 인식하고 있을 경우 마스터로 전환하지 않는다. **해결책**은 복제(6383)을 시작하기 전에 redis.conf 에서 slaveof 를 삭제하는 것이다.

  마스터(6382) -> 마스터 -> **Down** -> Down
  복제(6383) -> **Down** -> Down -> **Start(복제)**

장애조치(failover) 후 구성 형태가 바뀌는 경우

- 1차, 2차 복제가 있는 상황에서 진행 후 Master/Slave간 연결이 처음 상황과 다르게 구성되는 경우이다.
- 처음 상황

  Master -> Slave 6383(1차 복제) -> Slave 6384(2차 복제)

- 장애조치 진행

  Master(6382) -> Down -> Start(Slave)
  Slave1(6383) -> Master -> Master
  Slave2(6384) -> Slave -> Slave

- 장애조치 후 마스터에 복제 2대가 연결되어있는 상황이 되었다.

  Slave 6382(1차 복제) <- Master 6383 -> Slave 6384(1차 복제)

- 처음과 같은 형태로 구성하려면 6382 서버에 접속해서 **slaveof 127.0.0.1 6384** 명령을 실행한다.

1차 복제에서만 후보를 선정

- 센티널은 1, 2차 복제가 있을 경우 1차 복제만 인식하고 2차 이상은 인식하지 않는다. 그러므로 마스터 다운 시 1차 복제만 새 마스터 후보에 오를 수 있다.

Slave-priority 설정

- 이 파라미터의 디폴트 값은 100이다. 같은 마스터를 바라보는 복제 중에서 가장 적은 값을 가진 복제가 마스터에 선정된다. 즉, 복제가 3대인데 slave-priority 값이 각각 80, 90, 100 이면 마스터 다운 시 80인 복제가 새 마스터가 된다.
- slave-priority를 0으로 설정할 수 있는데, 0으로 하면 마스터로 승격되지 않는다. 특별한 이유로 이 복제를 마스터로 승격시키지 않으려면 0으로 설정한다.

마스터와 1차 복제가 거의 동시에 다운되면?

- 2차 복제가 있어도 Failover에 실패하게 된다. 정확히 표현하면 2차 복제는 마스터 후보가 아니다. 왜냐하면 센티널은 마스터와 1차 복제 정보는 가지고 있지만, 2차 복제 정보는 가지고 있지 않다. 그러므로 2차 복제가 있으도 마스터로 승격되지 못한다.

  Master(6382) -> Down
  Slave1(6383) -> Down
  Slave2(6384) -> Slave 마스터로 승격되지 못하고 복제로 남아 있다.

마스터와 1차 복제가 10초 차이로 다운되면?

- 마스터(6282)가 다운되면 1차 복제(6383)가 새로운 마스터(6383)로 승격된다. 그러면 처음에 2차 복제(6384)였던 서버가 1차 복제(6384)가 된다. 이 상태에서 마스터(6383)가 다운되면 6384가 마스터가 된다.
- 여기서 예로 든 10초는 down-after-milliseconds 값에 따라 다르다. 이 파라미터 값을 3초로 설정했기 때문에 대부분의 경우 4초면 복제가 마스터로 승격된다. 그러면 센티널은 새 마스터의 복제 정보를 가져오기 때문에 새 마스터가 다운되어도 가지고 있는 복제에서 다시 새 마스터로 승격시킨다.

  Master(6382) -> Down -> Down
  Slave1(6383) -> Master -> Down
  Slave2(6384) -> Slave(1차 복제) -> Master

- 서버가 다운되는 시간 차에 따라 Failover가 성공할 수도 있고 실패할 수도 있음을 설명하기 위해서 위와 같은 상황을 제시한 것이다.

센티널 대수와 Quorum 값

- Quorum 2 이면 적어도 2대 이상의 센티널이 마스터가 다운되었다고 인지해야 failover를 진행한다. 만약 quorum 2 이고 센티널이 2대인 상황에서 센티널 1대가 다운되었다면 failover는 결코 진행되지 않는다.
- 최소 권장 사항은 센티널 3대, quorum 2 이다. quorum은 sentinel.conf **sentinel monitor mymaster 127.0.0.1 6382 2** 에서 마지막 값이다.

Sentinel 명령

센티널 명령은 사용자가 사용할 수도 있지만, 많은 명령이 센티널 내부적으로 사용된다.

- Sentinel Data Structure : 센티널 Data Structure 설명
- INFO SENTINEL : 센티널에 대한 전반적인 정보를 조회한다.
- ROLE : 마스터들의 이름을 조회한다.
- SENTINEL MASTERS : 마스터들의 자세한 상태 정보를 조회한다.

- SENTINEL MASTER : 지정한 마스터의 자세한 상태 정보를 조회한다.
- SENTINEL SLAVES : 슬레이브의 상태 정보를 조회한다.
- SENTINEL REPLICAS : 복제노드의 상태 정보를 조회한다.
- SENTINEL SENTINELS : 센티널의 상태 정보를 조회한다.
- SENTINEL RESET : 지정한 마스터의 상태 정보를 초기화(RESET)한다.
  위에서 설명한 INFO 명령에서 다운된 복제 개수도 포한되는데, 제거할 때 이명령을 사용한다. RESET 명령 사용 직후에는 0이 되는데 곧 정상 정보로 채워진다.
- SENTINEL MONITOR : 지정한 마스터에 대한 모니터링을 시작한다.
- SENTINEL REMOVE : 지정한 마스터를 모니터링 대상에서 제거한다. 더 이상 모니터링하지 않는다.
- SENTINEL FAILOVER : 명령으로 강제 장애조치를 진행한다.
- SENTINEL CKQUORUM : 쿼럼값이 올바른지 체크한다.
- SENTINEL SIMULATE-FAILURE : 장애조치 중간에 리더를 다운시키는 시뮬레이션을 설정한다.
- SENTINEL SET : 센티널 설정값을 변경한다.
- SENTINEL FLUSHCONFIG : 센티널 설정(configuration)파일을 다시 쓴다.
- SENTINEL INFO-CACHE : INFO 명령으로 가져온 정보를 저장한다.
- SENTINEL PENDING-SCRIPTS : 현재 대기 또는 실행중인 스크립트 정보를 조회한다.
- SENTINEL NOTIFICATION : 알림기능에 대한 설명.
- SENTINEL ELECTION : 센티널 리더를 선출하는 방법과 장애조치 설명

Sentinel 클라이언트

redis.io에 소개된 Sentinel clients

- C# - csredis : C# 용 레디스, 센티널 클라이언트 Async (and sync) client for Redis and Sentinel
- Go - gore : Go 용 레디스, 센티널 클라이언트 A full feature redis Client for Go. Supports Pipeline, Transaction, LUA scripting, Pubsub, Connection Pool, Sentinel and client sharding
- Node.js - ioredis : Node.js 용 레디스, 센티널, 클러스터 클라이언트 A delightful, performance-focused and full-featured Redis client. Supports Cluster, Sentinel, Pipelining and Lua Scripting
- PHP - PHP Sentinel Client : PHP용 센티널 클라이언트 A PHP sentinel client acting as an extension to your regular redis client

Sentinel Sentinel Data Structure >>

redisgate@gmail.com 02.503.2235
서울시 강남구 강남대로 342 역삼빌딩 5층 (역삼동) 우 06242