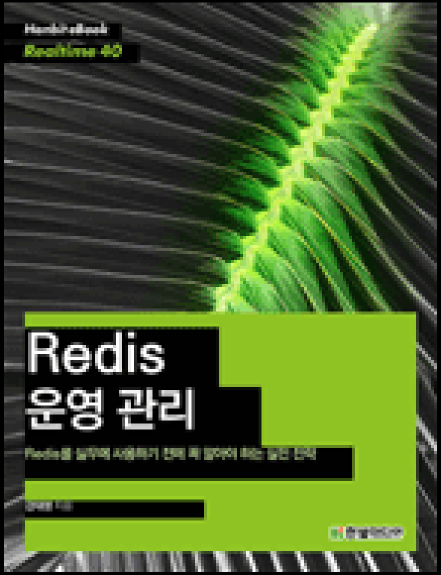
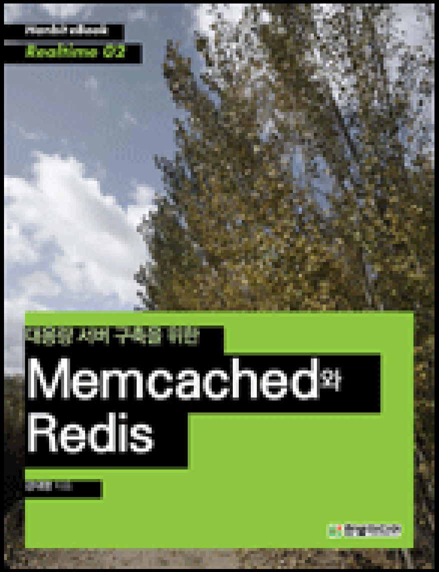
Redis Sentinel

charsyam@naver.com

Redis.io



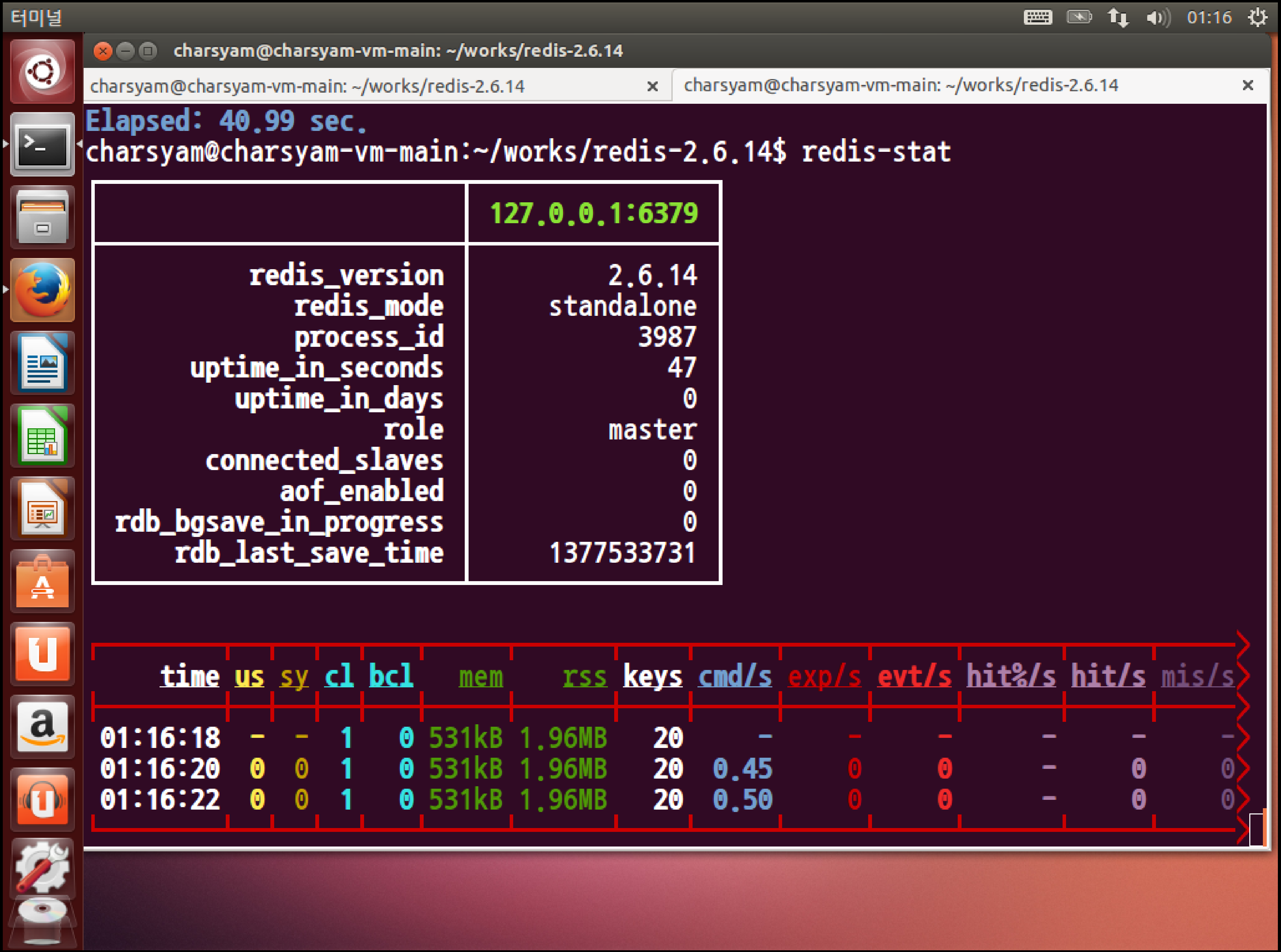


Redis/Twemproxy

contributor

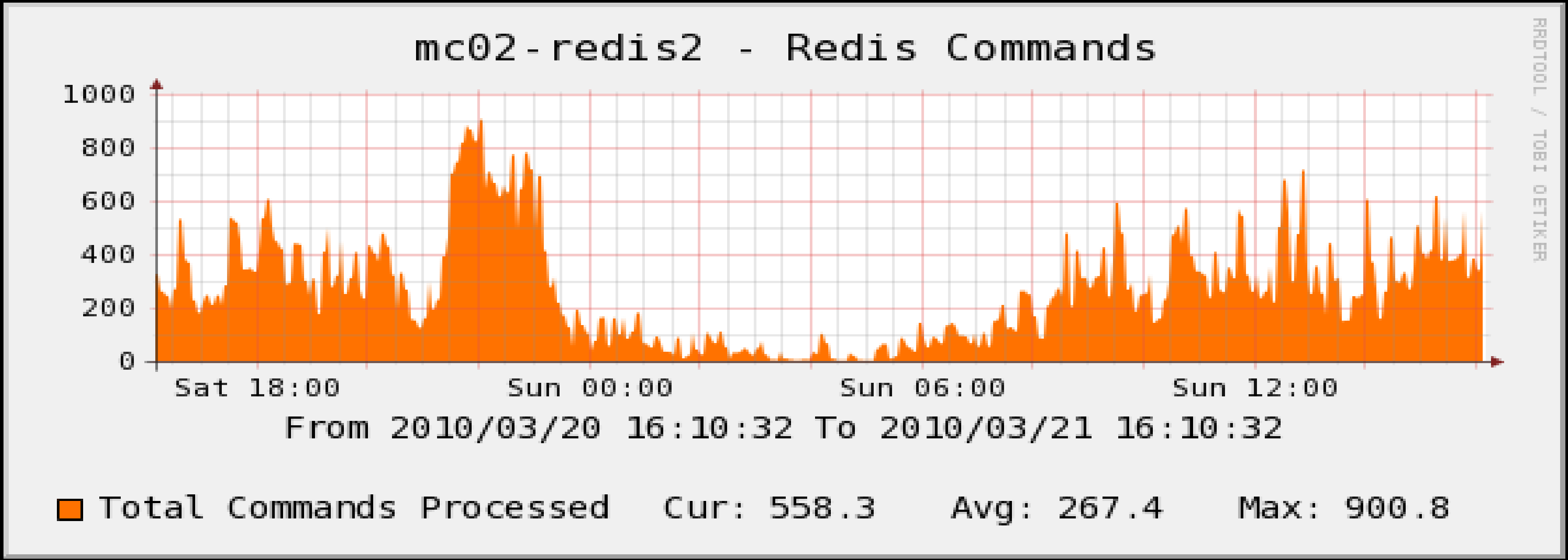
카카오 홈 개발자

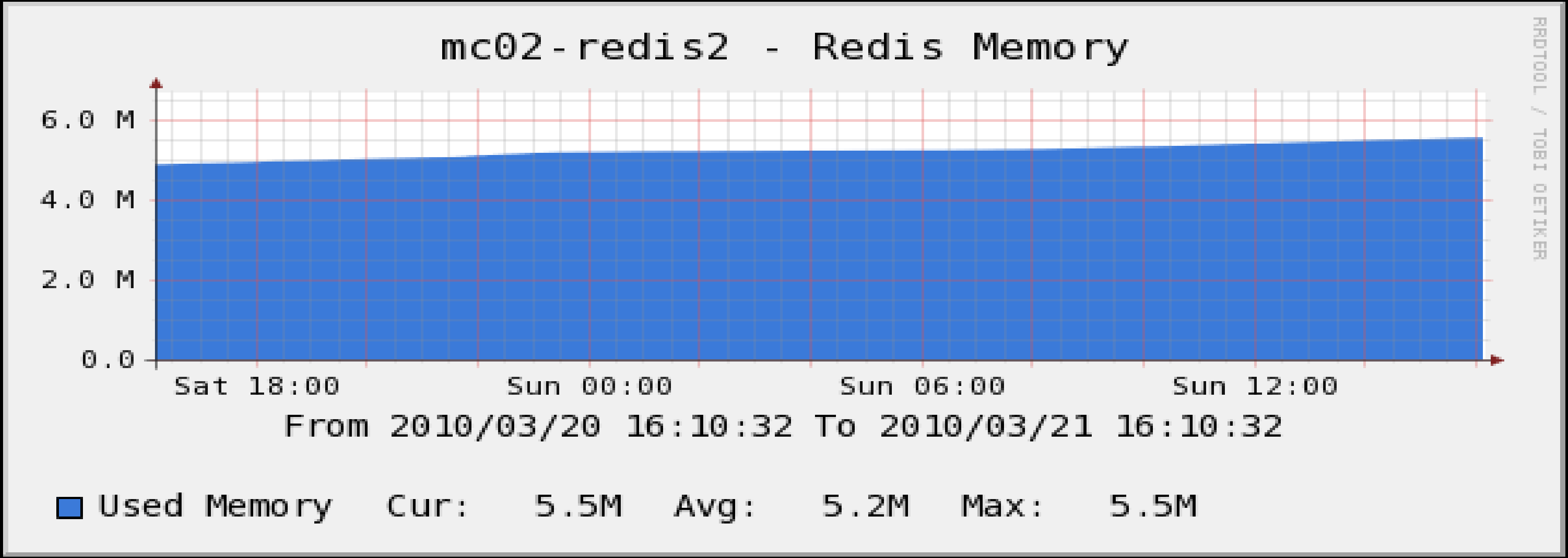
Redis Monitoring?

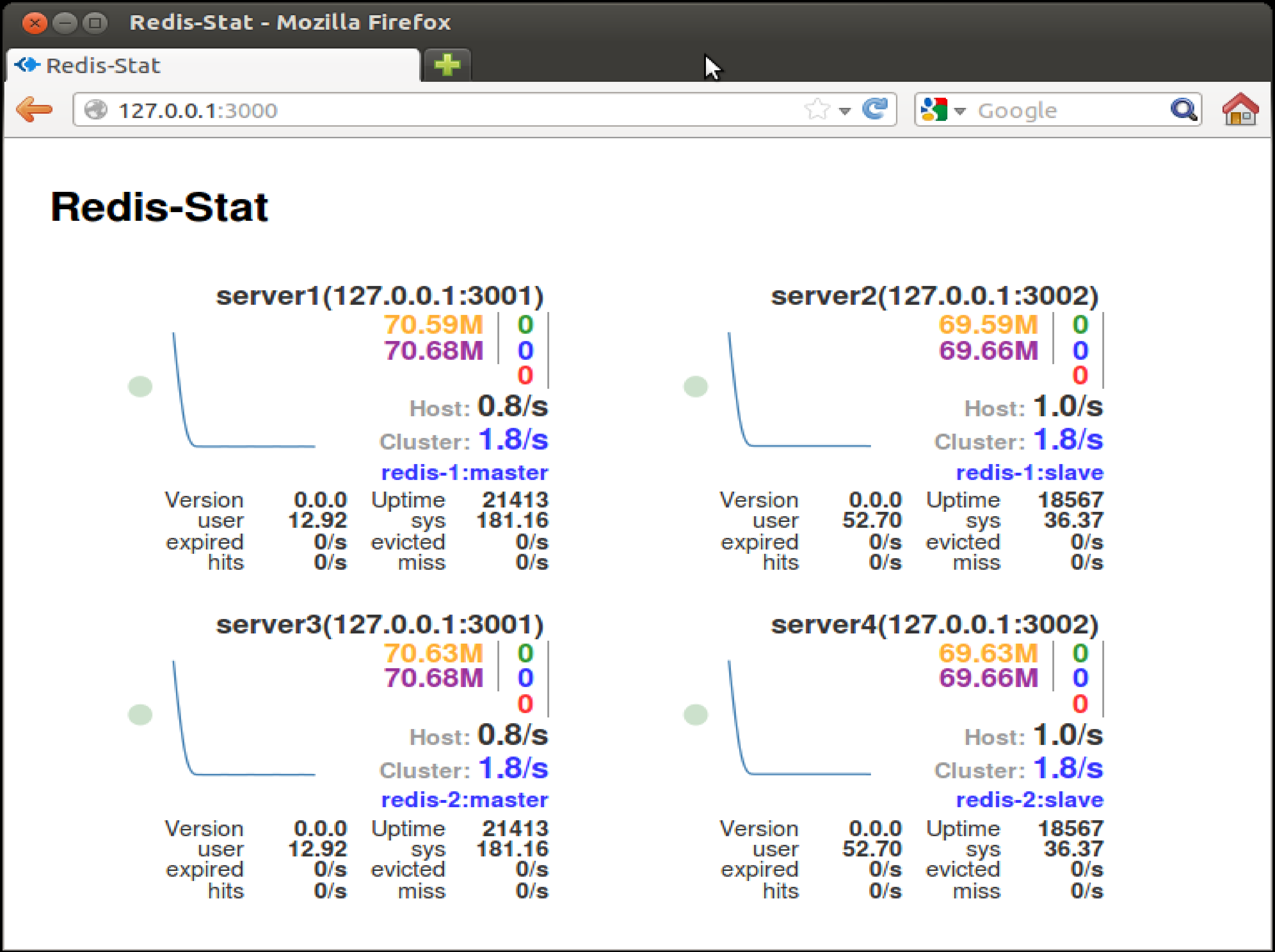

터미널
01:16
charsyam@charsyam-vm-main: ~/works/redis-2.6.14
charsyam@charsyam-vm-main: ~/works/redis-2.6.14
charsyam@charsyam-vm-main: ~/works/redis-2.6.14

```
Elapsed: 40.99 sec.
charsyam@charsyam-vm-main:~/works/redis-2.6.14$ redis-stat
┌──────────────────────────┬────────────────┐
│                          │ 127.0.0.1:6379 │
├──────────────────────────┼────────────────┤
│           redis_version  │         2.6.14 │
│              redis_mode  │     standalone │
│              process_id  │           3987 │
│       uptime_in_seconds  │             47 │
│          uptime_in_days  │              0 │
│                    role  │         master │
│        connected_slaves  │              0 │
│             aof_enabled  │              0 │
│  rdb_bgsave_in_progress  │              0 │
│     rdb_last_save_time   │     1377533731 │
└──────────────────────────┴────────────────┘

     time  us  sy  cl  bcl    mem    rss keys cmd/s exp/s evt/s hit%/s hit/s mis/s
 01:16:18   -   -   1    0  531kB 1.96MB   20     -     -     -      -     -     -
 01:16:20   0   0   1    0  531kB 1.96MB   20  0.45     0     0      -     0     0
 01:16:22   0   0   1    0  531kB 1.96MB   20  0.50     0     0      -     0     0
```


mc02-redis2 - Redis Commands
1000
800
600
400
200
0
Sat 18:00
Sun 00:00
Sun 06:00
Sun 12:00
From 2010/03/20 16:10:32 To 2010/03/21 16:10:32
Total Commands Processed Cur: 558.3 Avg: 267.4 Max: 900.8
RRDTOOL / TOBI OETIKER


mc02-redis2 - Redis Memory
6.0 M
4.0 M
2.0 M
0.0
Sat 18:00
Sun 00:00
Sun 06:00
Sun 12:00
From 2010/03/20 16:10:32 To 2010/03/21 16:10:32
Used Memory Cur: 5.5M Avg: 5.2M Max: 5.5M
RRDTOOL / TOBI OETIKER



Redis-Stat

```
server1(127.0.0.1:3001)                        server2(127.0.0.1:3002)
               70.59M | 0                                     69.59M | 0
               70.68M | 0                                     69.66M | 0
                      | 0                                            | 0
             Host: 0.8/s                                    Host: 1.0/s
          Cluster: 1.8/s                                 Cluster: 1.8/s
         redis-1:master                                  redis-1:slave
 Version    0.0.0   Uptime   21413          Version    0.0.0   Uptime   18567
    user    12.92      sys  181.16             user    52.70      sys   36.37
 expired      0/s  evicted     0/s          expired      0/s  evicted     0/s
    hits      0/s     miss     0/s             hits      0/s     miss     0/s

server3(127.0.0.1:3001)                        server4(127.0.0.1:3002)
               70.63M | 0                                     69.63M | 0
               70.68M | 0                                     69.66M | 0
                      | 0                                            | 0
             Host: 0.8/s                                    Host: 1.0/s
          Cluster: 1.8/s                                 Cluster: 1.8/s
         redis-2:master                                  redis-2:slave
 Version    0.0.0   Uptime   21413          Version    0.0.0   Uptime   18567
    user    12.92      sys  181.16             user    52.70      sys   36.37
 expired      0/s  evicted     0/s          expired      0/s  evicted     0/s
    hits      0/s     miss     0/s             hits      0/s     miss     0/s
```

Checking Liveness

Checking Liveness

```
servers = [ ('localhost', 6379),
            ('localhost', 6380),
            ('localhost', 6381) ]


conns = []


def check():
    for conn in conns:
        print conn.ping()
```

checking it works

Checking it works

```
def check():
    for conn in conns:
        i = conn.info()
        print i['run_id'],
              i['used_memory_human'],
              i['total_commands_processed']
```

Is it Reliable?

Is Server really Failed?

Split Network.

Server is busy.

Monitoring is not the one we really want.

HA

High Availity

what do you do?

when your servers failed

1. Recognize Redis is down.

2. choose new master candidate

3. Promote slave to Master.

4. make client connect to new master

What Redis Sentinel do?

1. Check Redis Liveness

1. Check Redis Liveness

    -Subjective Down

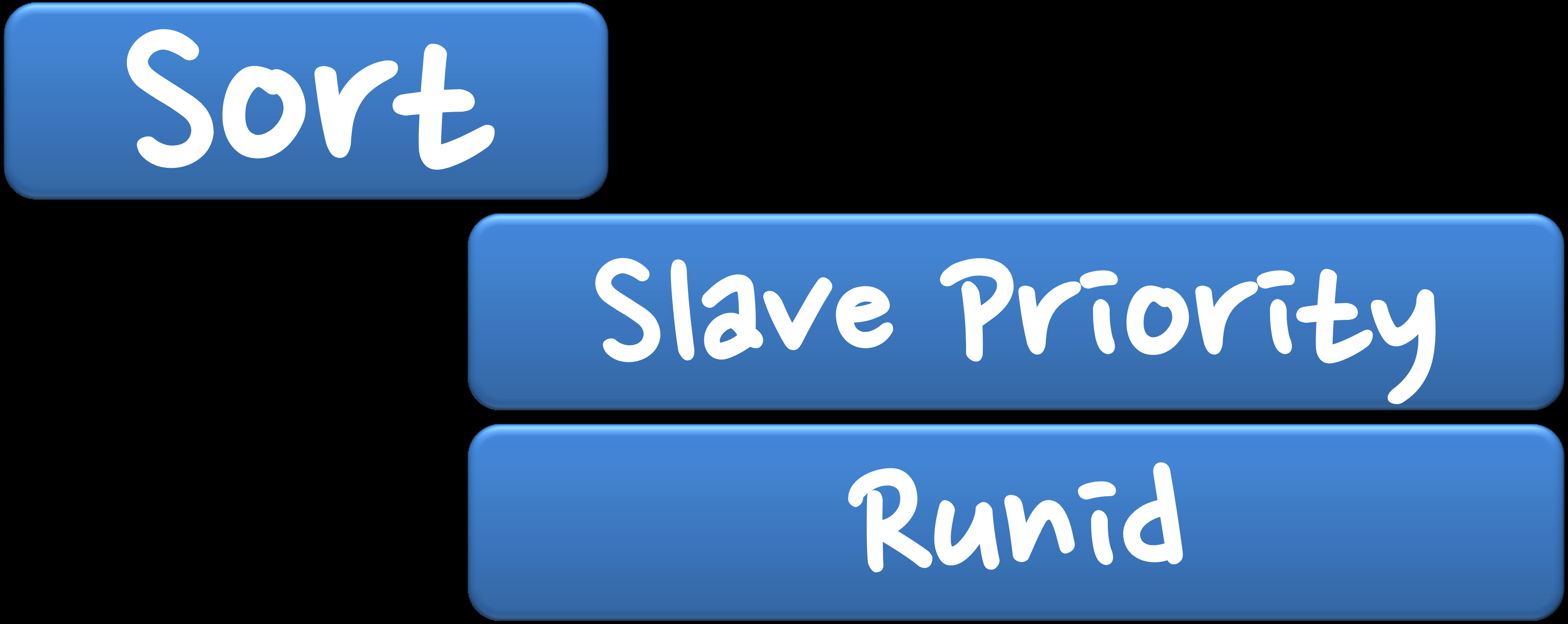
    -Objective Down

2. choose good slave.

Good Slave

- NOT SDOWN, ODOWN, DISCONNECTED
- NOT DEMOTE
- Ping reply > info_validity_time
- Slave_priority != 0
- Info reply > info_validity_time

choose new master



Sort

Slave Priority

Runid

3. Promote it

Promoting

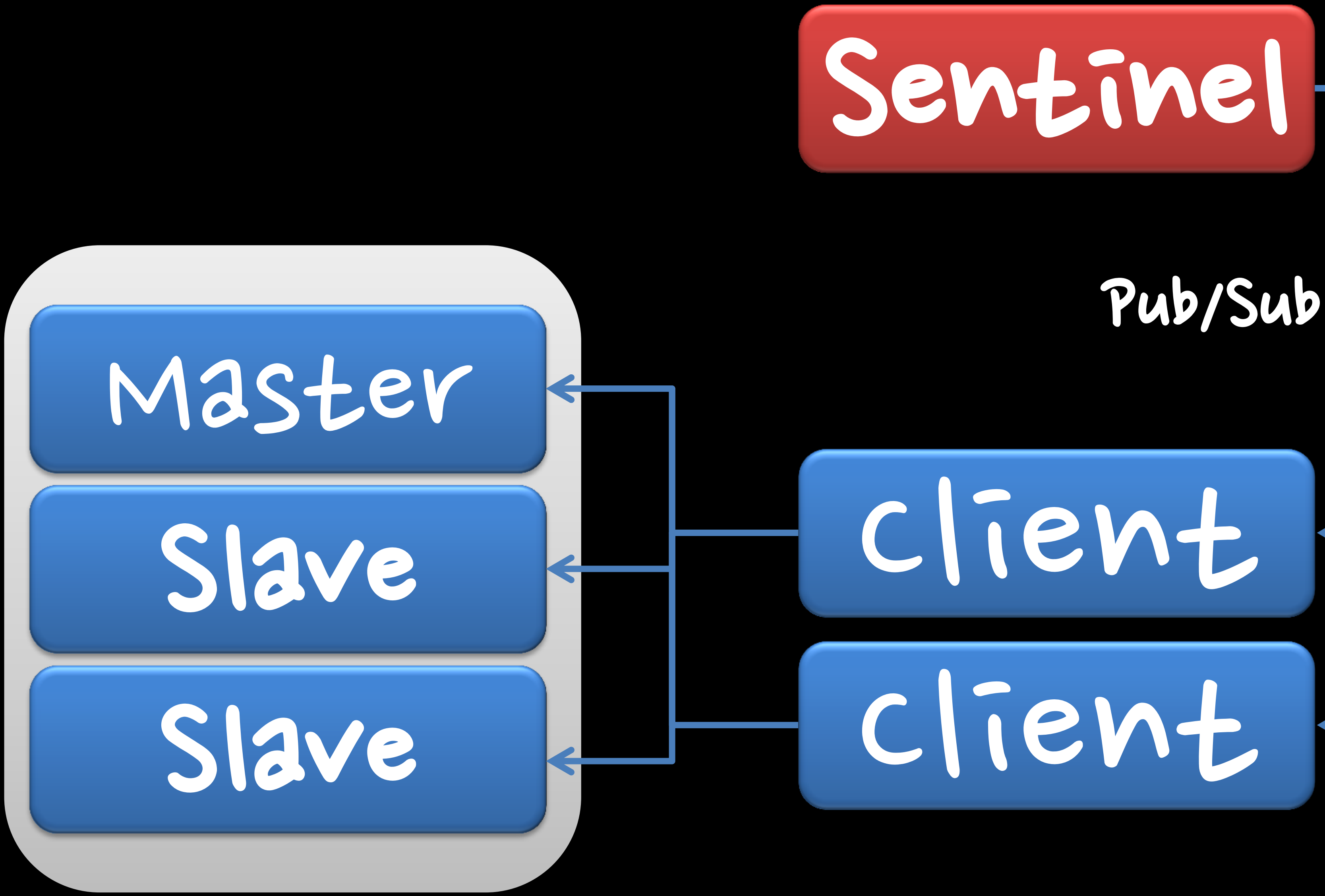
Send Slaveof no one to new master

Send Slaveof [new master ip] [addr] to other redis

Notify new master to clietns +switch-master

Set DEMOTE mark to old-Master

Failover Scenario



Sentinel

Pub/Sub

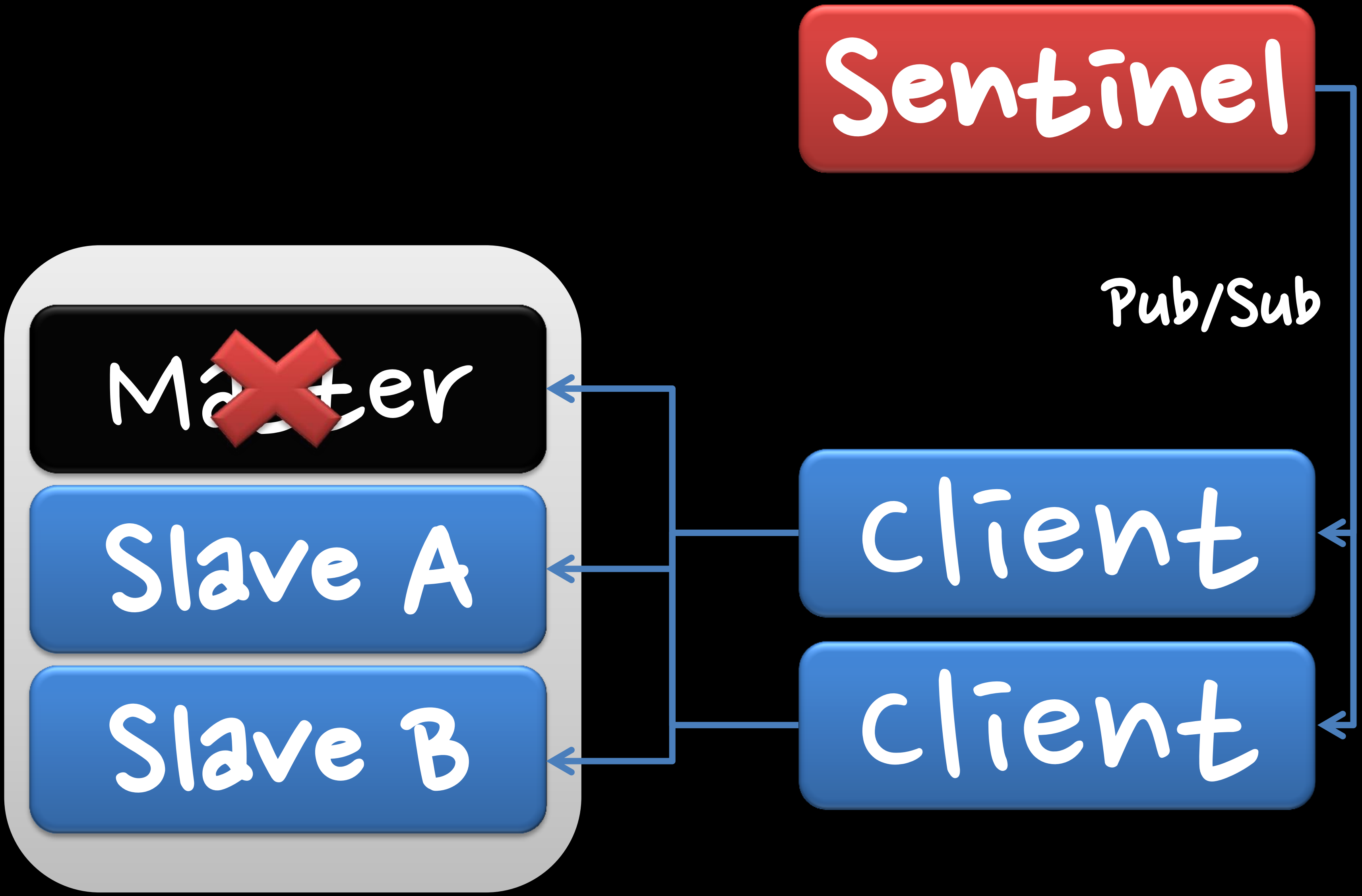
Master

Slave

Slave

Client

Client



Sentinel

Pub/Sub

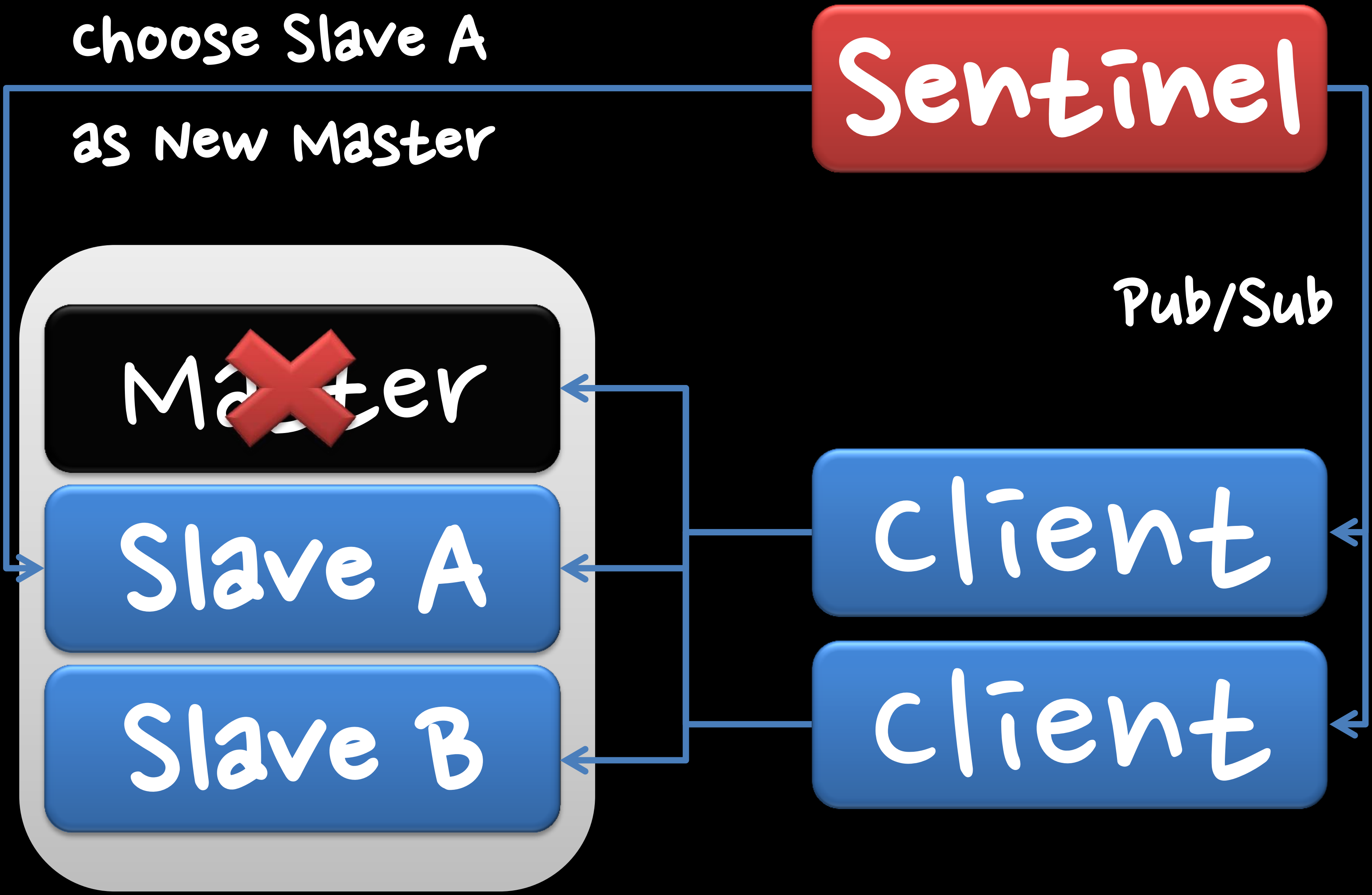
Master

Slave A

Slave B

Client

Client



choose Slave A as New Master

Sentinel

Pub/Sub

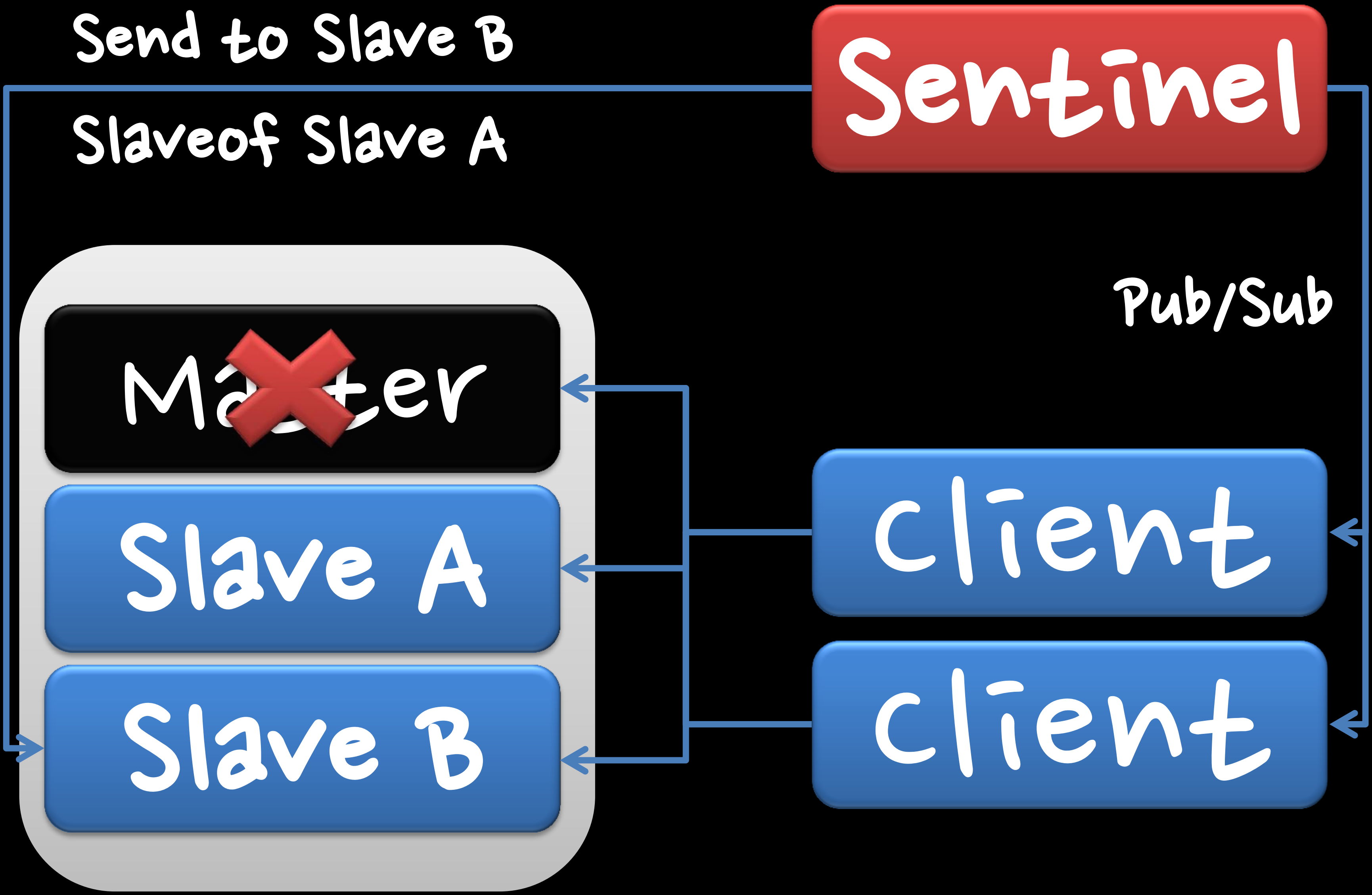
Master

Slave A

Slave B

client

client



Send to Slave B

Slaveof Slave A

Sentinel

Pub/Sub

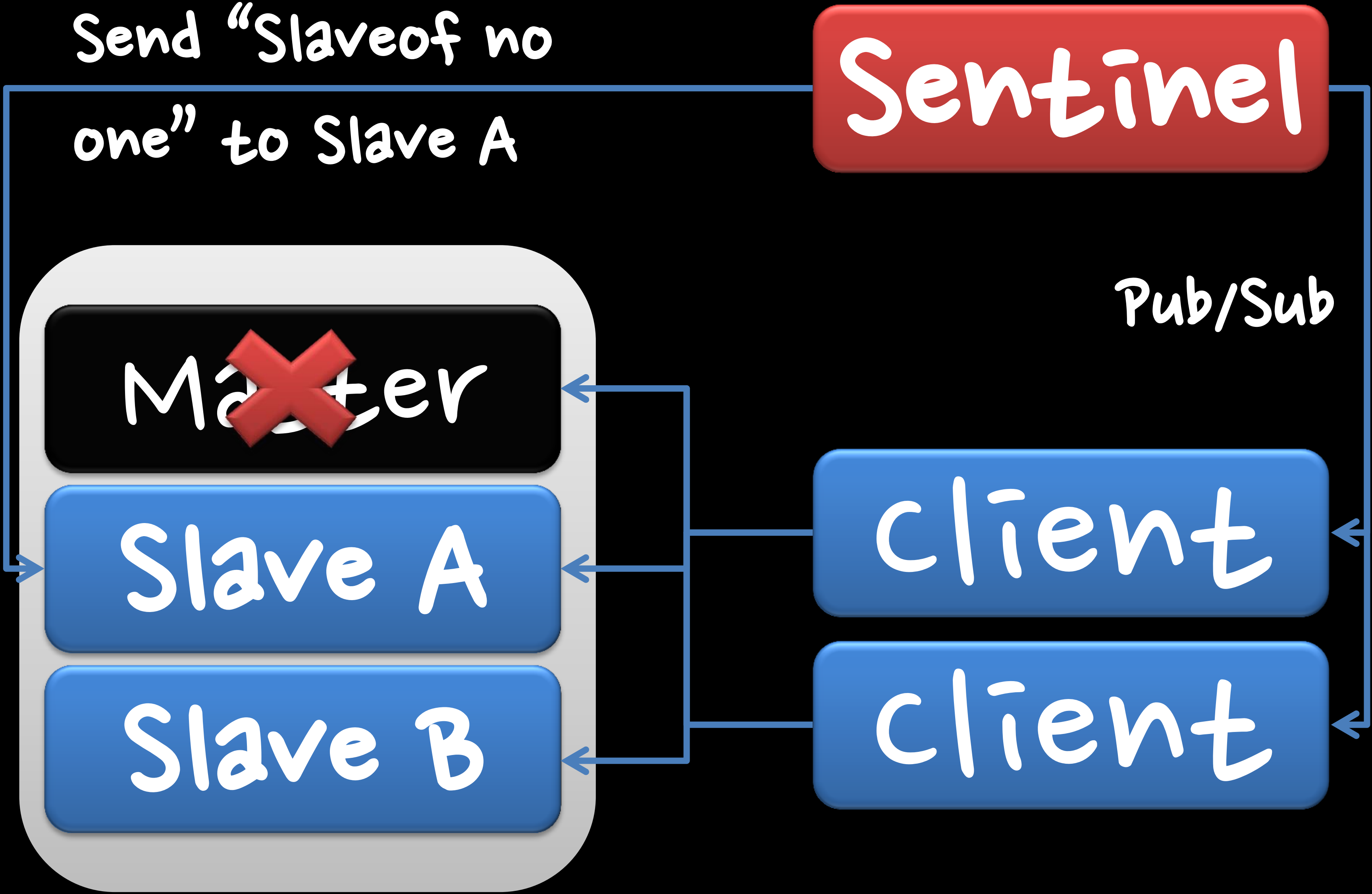
Master

Slave A

Slave B

Client

Client



Send “Slaveof no one” to Slave A

Sentinel

Pub/Sub

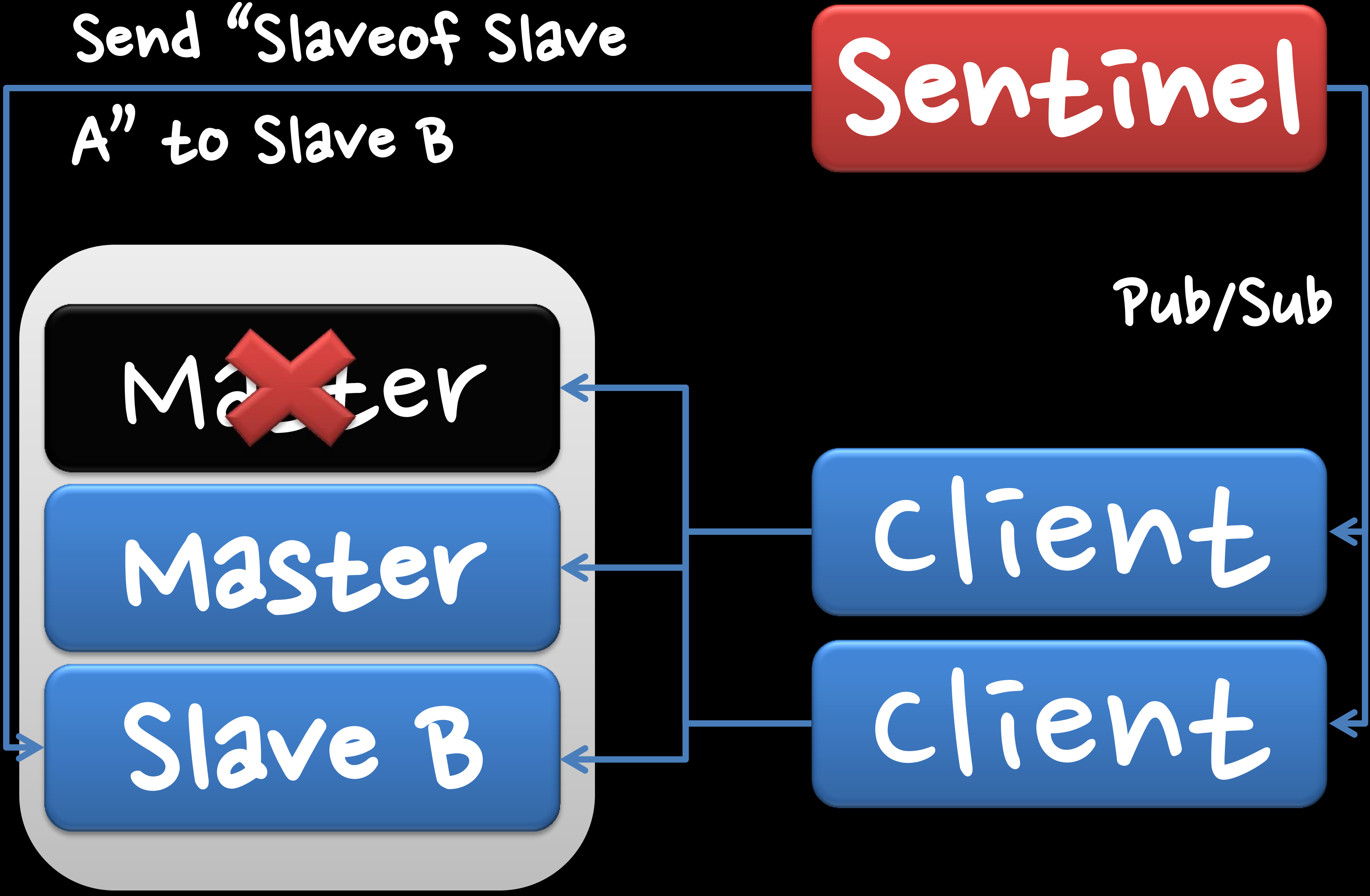
Master

Slave A

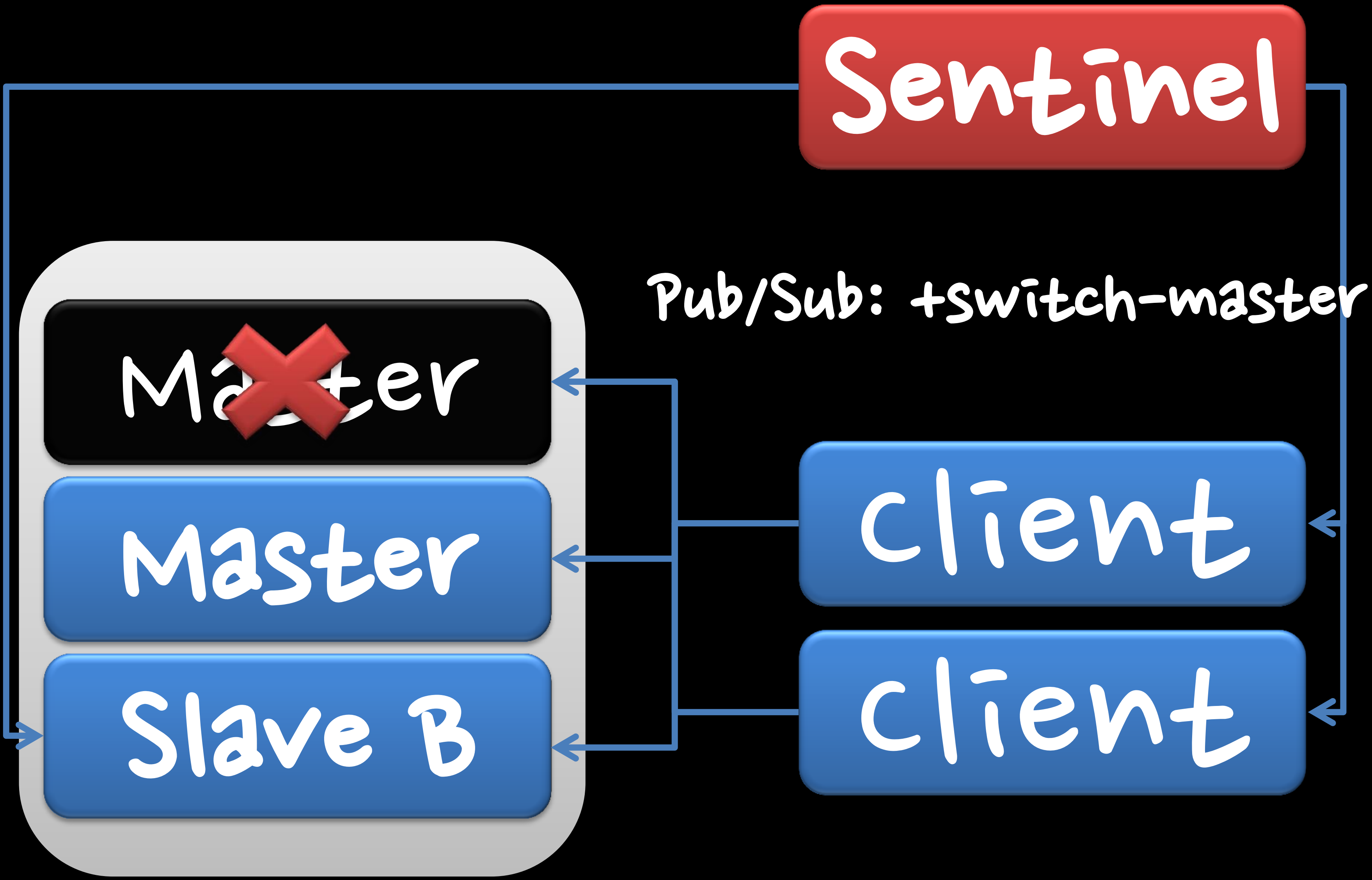
Slave B

Client

Client



Send "Slaveof Slave A" to Slave B



Sentinel
Pub/Sub: +switch-master
Master
Master
Slave B
Client
Client

Sentinel Internal

Sentinel Failover State Machine

State	내용	Next State
NONE		
WAIT_START		SELECT_SLAVE
SELECT_SLAVE		SEND_SLAVEOF
SEND_SLAVEOF		WAIT_PROMOTION
WAIT_PROMOTION		
RECONF_SLAVES		
DETECT_END		

Starting Point Of Sentinel

- sentinelTimer in sentinel.c
  - called every 100ms
  - sentinelHandleRedisInstance

SentinelRedisInstance

- master
  - If it is master, it is null.
- slaves
- sentinels

sentinelHandleDictOfRedisInstances

```
while((de = dictNext(di)) != NULL) {
       sentinelRedisInstance *ri = dictGetVal(de);


       sentinelHandleRedisInstance(ri);
       if (ri->flags & SRI_MASTER) {
           sentinelHandleDictOfRedisInstances(ri->slaves);
           sentinelHandleDictOfRedisInstances(ri->sentinels);
           ......
       }
}
```

sentinelHandleRedisInstance

- Reconnect to Instances
- Ping
- Asking Master State to other Sentinel
- check SDOWN
- check ODOWN

How to check Subjective Down

- When Sentinel get Ping Reply
  - Set last_avail_time as current
- check
  - e = current -last_avail_time
  - e > last_avail_time
    - Subjective Down.

How to check Objective Down

- Ask to other sentinels is it down.
  - sentinelAskMasterStateToOtherSentinels
- If other sentinels reply it is down.
  - count them, if it is bigger than Quorum.
  - It is objective Down.

How to find Redis

INFO Command

```
# Replication
role:master
connected_slaves:1
slave0:ip=127.0.0.1,port=6380,state=online,offset=1,lag=0
master_repl_offset:1
repl_backlog_active:1
repl_backlog_size:1048576
repl_backlog_first_byte_offset:2
repl_backlog_histlen:0
```

INFO Command

```
# Replication
role:slave
master_host:127.0.0.1
master_port:6379
```

+redirect-master

How to find Sentinel

Find Sentinel

Subscribe SENTINEL_HELLO_CHANNEL
In Master

Publish Sentinel Information
In sentinelPingInstance

Add New Sentinel
In sentinelReceiveHelloMessages

Using Pub/sub

```
*4
$8
pmessage
$1
*
$18
__sentinel__:hello
$58
127.0.0.1:26379:7d717f945afde99e6f82f825de052f17cab7e6f3:1
```

Is Sentinel Really Good?

Not Mature #1

Sentinel Conf

```
port 26379


sentinel monitor mymaster 127.0.0.1 6379 2
sentinel down-after-milliseconds mymaster 2000
sentinel can-failover mymaster yes
sentinel parallel-syncs mymaster 1
sentinel failover-timeout mymaster 900000
```

Quorum Count

```
sentinel monitor mymaster 127.0.0.1 6379 2
```

We should check this count.

Sentinel Conf

```
sentinel down-after-milliseconds mymaster 2000
```

If this value is Too small,

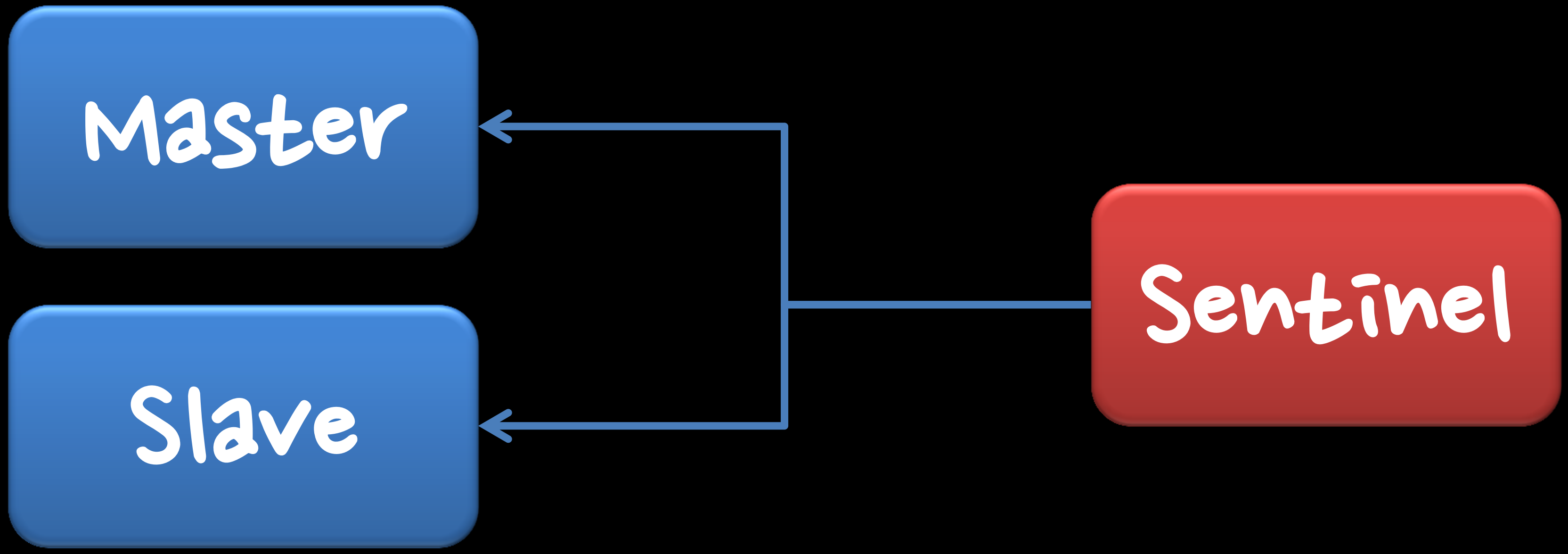
It can cause some trouble.

-sdown/+sdown loop

Compare with Zookeeper Conf

```
tickTime=2000
dataDir=/var/zookeeper
clientPort=2181
initLimit=5
syncLimit=2
server.1=z001:2888:3888
server.2=z002:2888:3888
server.3=z003:2888:3888
```

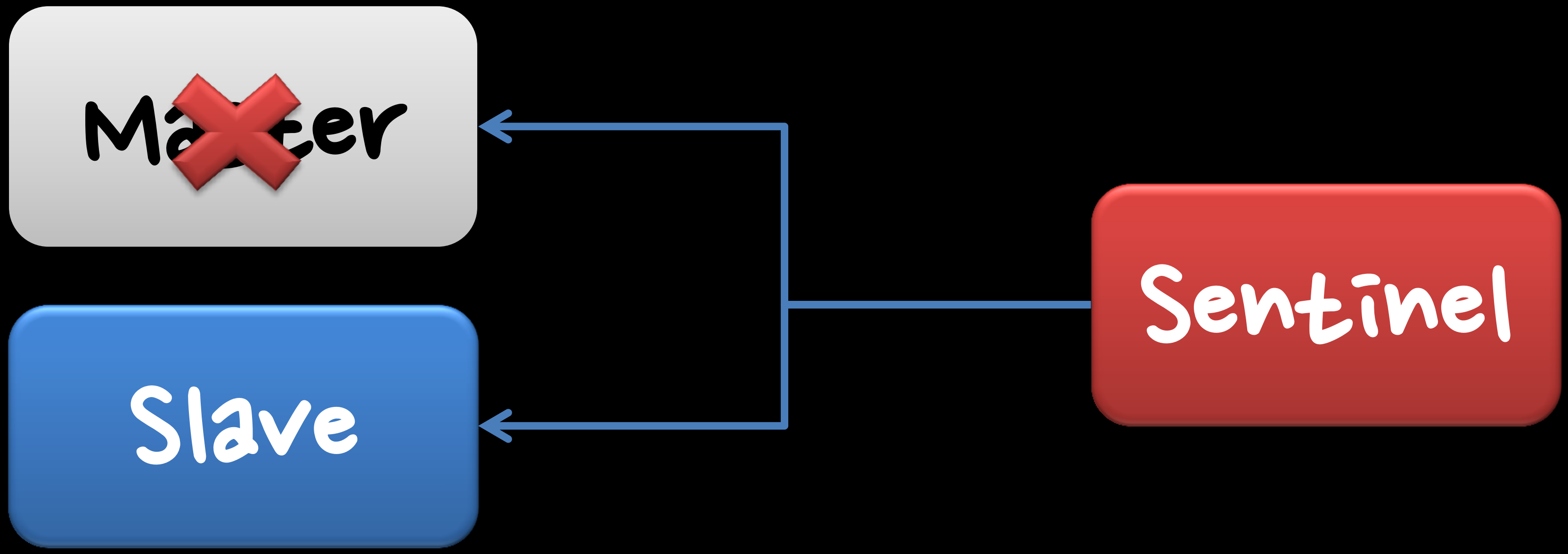
Not Mature #2

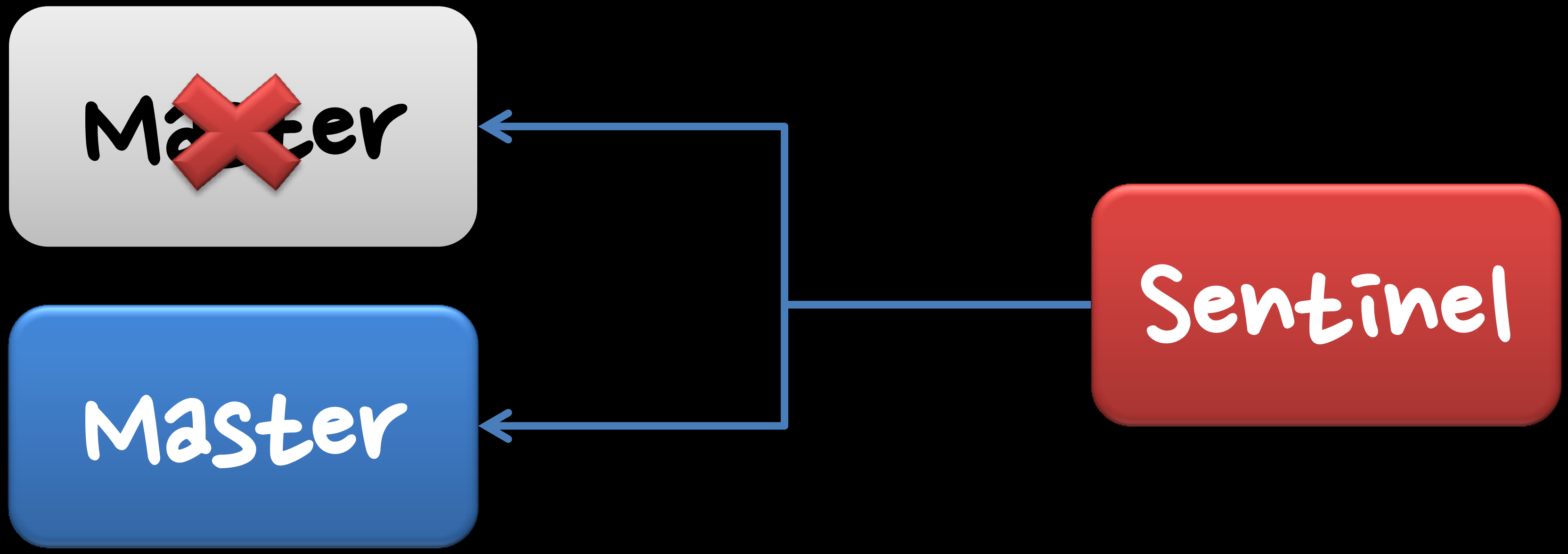


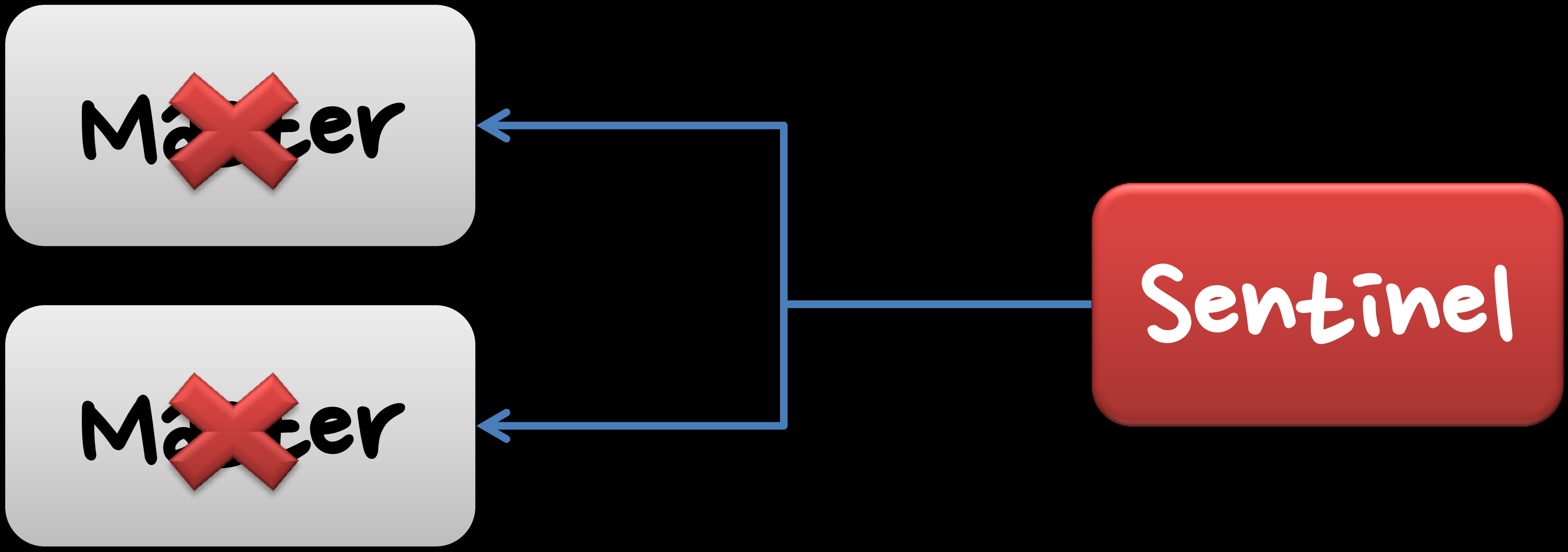
Master

Slave

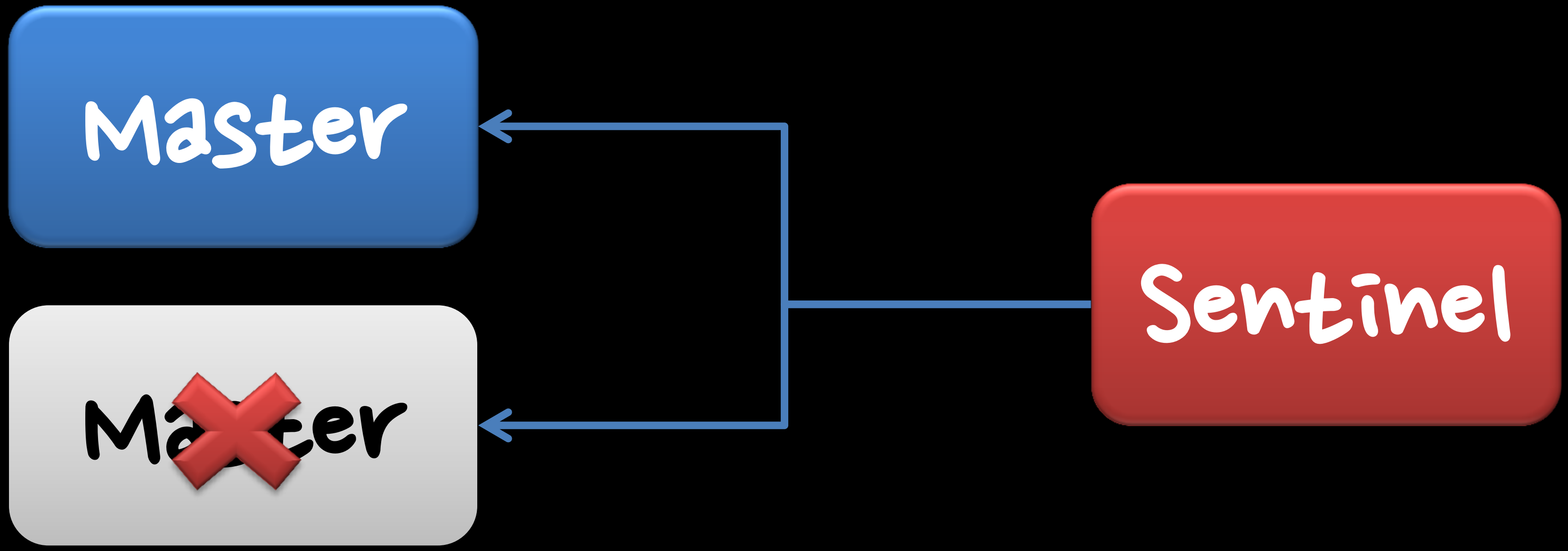
Sentinel


Master
Slave
Sentinel


Master
Master
Sentinel

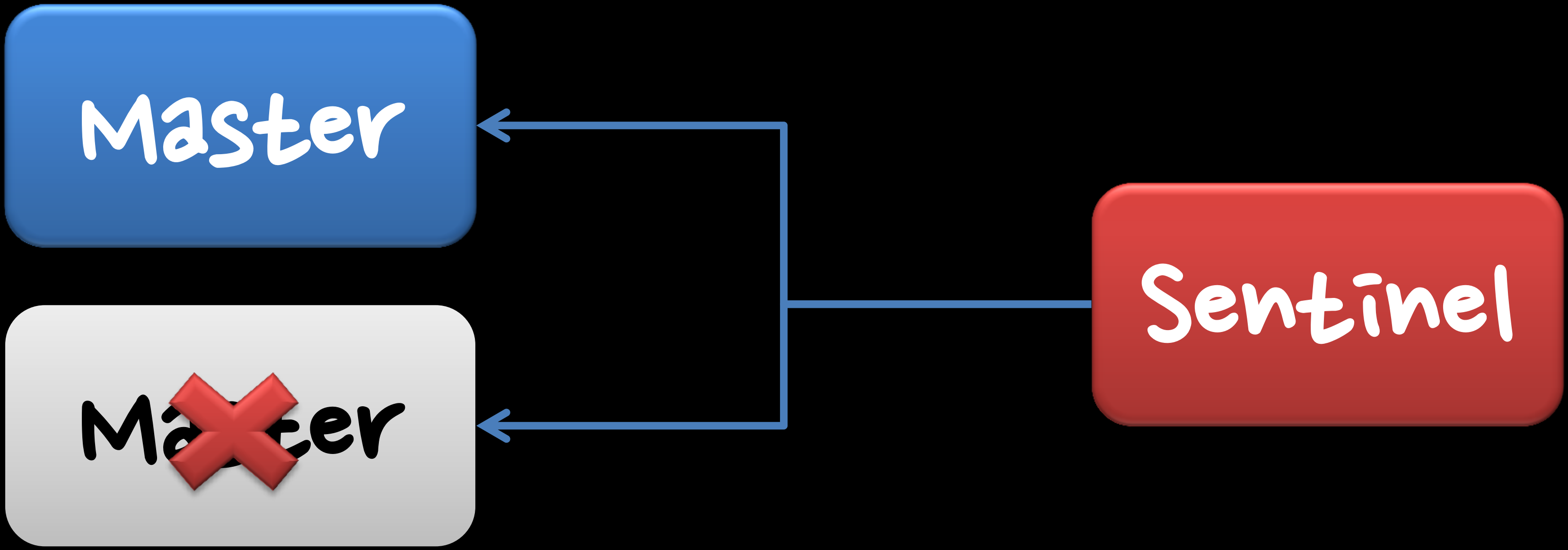

Master
Master
Sentinel

case #1

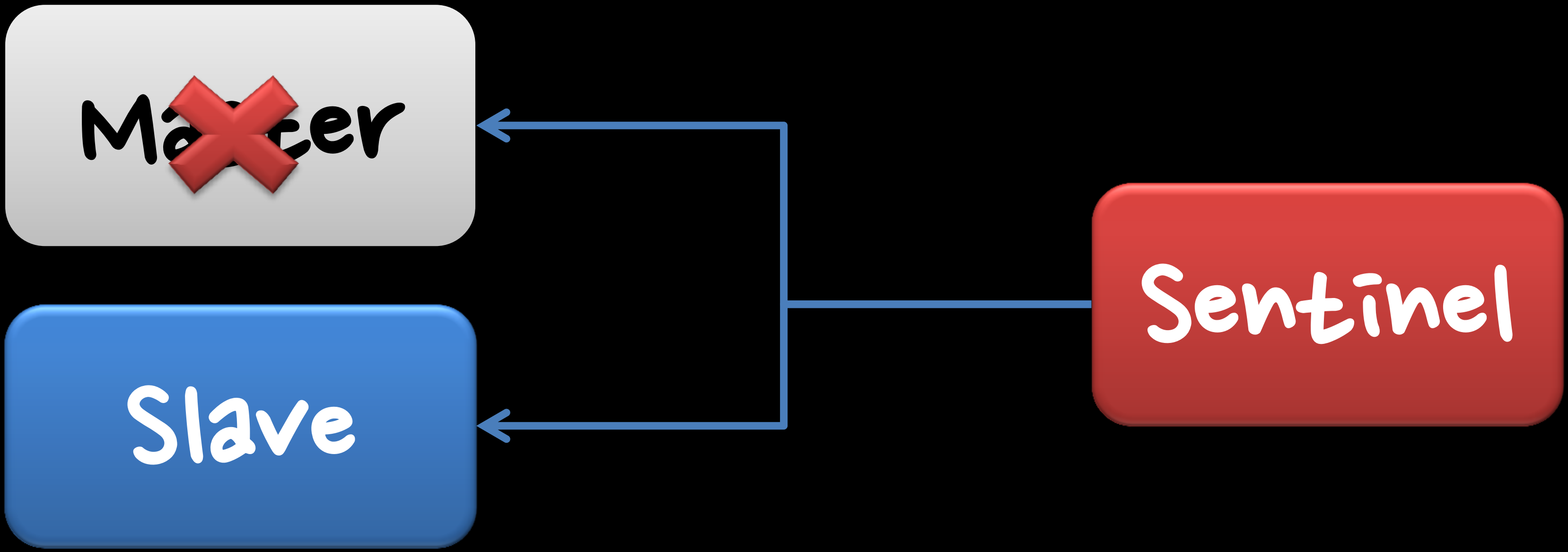


It's ok.

case #1



case #2

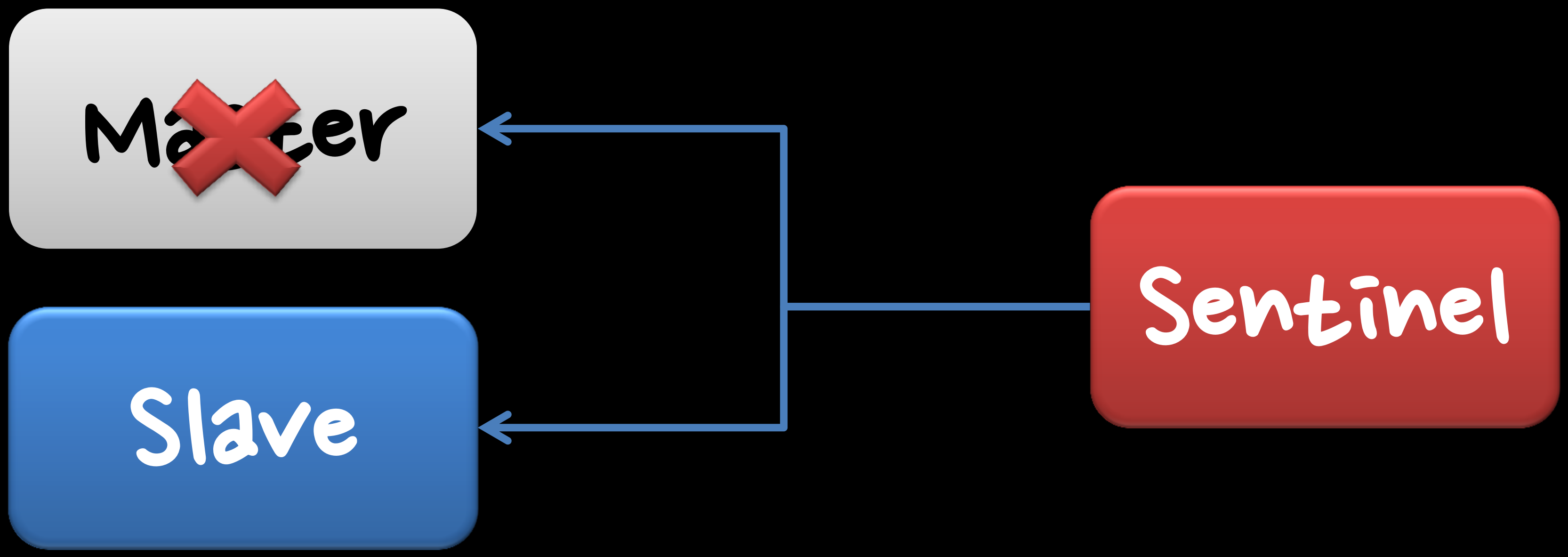


~~Master~~

Slave

Sentinel

case #2



Master

Slave

Sentinel

Sentinel can't promote

Slave to Master

why?

Sentinel Mechanism.

When master is changed to Slave,
Reconf with new Master.

Sentinel Mechanism.

When master is changed to Slave,
Reset information with new Master.

But, if new master failed, sentinel will
not get any information from it.

Sentinel Mechanism.

```
if ((ri->flags & SRI_MASTER) && role == SRI_SLAVE && ri->slave_master_host)
    {
        sentinelEvent(REDIS_WARNING,"+redirect-to-master",ri,
            "%s %s %d %s %d",
            ri->name, ri->addr->ip, ri->addr->port,
            ri->slave_master_host, ri->slave_master_port);
        sentinelResetMasterAndchangeAddress(ri,ri->slave_master_host,
                                            ri->slave_master_port);
        return; /* Don't process anything after this event. */
    }
```

Not Mature #3

If redis is down, when redis is loading data(RDB, AOF), sentinel can’t register redis.

conclusion

Even though, Sentinel is not mature.

It is useful.

Redis Sentinel Tips

1. Subscribe all Sentinels And ignore duplicated message

2. Sentinel has explicit failover command

```
sentinel mymaster failover
```

But can’t select new master

Hacking is easy.

https://github.com/antirez/redis/pull/1126

3. Make Sentinel.conf same.

Sentinels can’t share their status

Deploy conf to all sentinels with Script

And restart all sentinels.

4. If you want not to promote specific server to master

Set Slave-Priority = 0

Thank you!

Just One more thing

Twemprox

Redis

Redis-sentinel

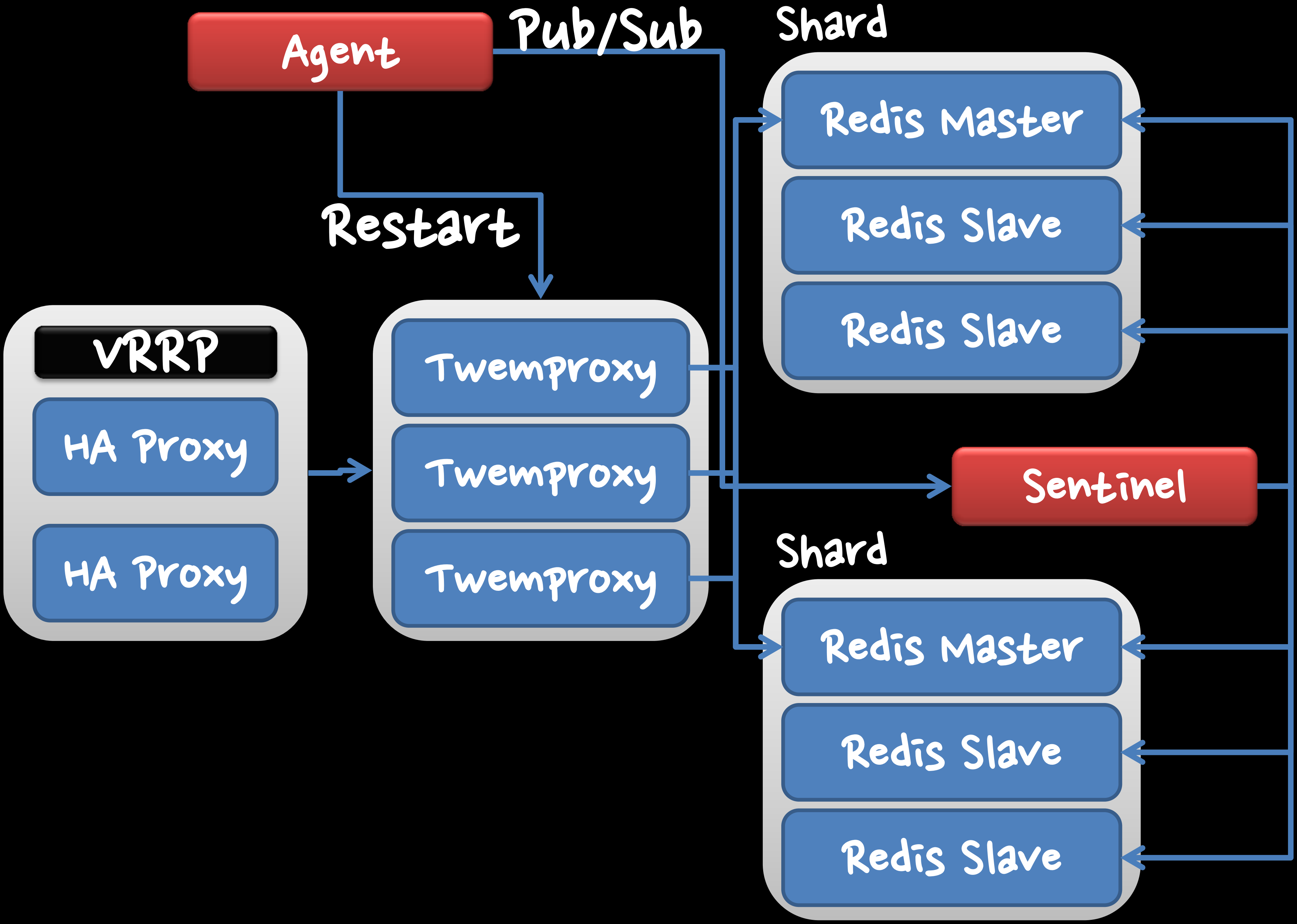
→

Sharded

Redis

Redis-Sentinel TwemProxy Agent

http://www.codeproject.com/Articles/656965/Redis-Sentinel-TwemProxy-Agent



Agent

Pub/Sub

Restart

Shard

Redis Master

Redis Slave

Redis Slave

VRRP

HA Proxy

HA Proxy

Twemproxy

Twemproxy

Twemproxy

Sentinel

Shard

Redis Master

Redis Slave

Redis Slave