主备切换设计文档

修订历史

版本	修订日期	修订描述	作者	备注
1.0	2015-12-31	文档编写	张晨东	

1 系统设计

1.1 综述

主备切换是作为集群选主功能实现的一个支持过程，主集群发生异常导致重新选举产生新的主集群，这时所有的集群就会发起由主向备切换或者由备向主切换的过程，必须等待所有的集群完成此切换过程后，集群才能正常对外提供服务。

1.2 名词解释

主备集群切换过程：在多集群部署模式下，集群由主切换为备或者由备切换为主的过程。

集群角色：每个集群保存有自己的角色，主集群的角色为OBI_MASTER，备集群的角色为OBI_SLAVE，刚启动时未接收到任何角色设置的集群角色为OBI_INIT。

Raft角色：每个集群中的RootServer会保存自己在选举过程中所得到的角色，角色标记对应Raft一致性算法中的三种角色为：领导者（Leader），则保存OB_LEADER；如果是候选者（Candidate），则保存OB_CANDIDATE；如果是跟随者（FOLLOWER），则保存OB_FOLLOWER。

RootServer选举状态：RootServer在运行过程中会根据当前的选举功能所处的状态和自己的Raft角色来为自己设定一个选举状态，以此作为Raft角色变化与集群切换流程的衔接。RootServer包括三个选举状态：初始化状态（INIT），选举进行状态（DURING_ELECTION），有主状态（AFTER_ELECTION）。

1.3 功能

RootServer中Raft模块会标记当前RootServer的角色，当这个角色发生变化时RootServer就会进行相应的集群角色切换，此小节内容介绍在Raft角色发生变化时主备集群切换的过程。

多集群部署模式下每个集群也会有自己的角色，可以是主集群角色（OBI_MASTER）、备集群角色（OBI_SLAVE）或者初始集群角色（OBI_INIT）。主集群是唯一的，它能提供强一致性的查询服务；备集群可以有多个，它们只能提供弱一致性的查询服务；初始化集群不能提供任何查询服务。

Raft模块中的角色变化会引起集群角色的变化，进而调起集群切换的流程。当所有的RootServer都是处于刚启动的状态时，在Raft模块中每个RootServer标记的角色都是OB_FOLLOWER，它们的集群角色都是OBI_INIT，必须通过人工指定Raft角色为OB_LEADER，这时每个集群都会发生集群角色的改变，集群角色从OBI_INIT改变为OBI_MASTER或者OBI_SLAVE。RootServer的主线程会每隔10ms检查一次当前的Raft角色是否发生变化，一旦发生改变，相当于集群角色改变，进而会引起集群的切换。

1.4 性能指标

待添加

2 模块设计

在实现此模块时，可以分为集群角色变化感知，集群切换，以及RootServer与UpdateServer之间的租约机制实现三部分。

集群角色变化感知

- 主线程每隔100ms会检查当前的节点角色是否发生变化，主线程中会根据情况分别调用由主切换为备或者由备切换为主的函数，由主切换为备和由备切换为主分别对应如下两个函数：start_master_cluster()，start_slave_cluster()。 当前是使用三个状态来标记RootServer所处的状态:INIT，DURING_ELECTION，AFTER_ELECTION。

```
for (;;)
{
    switch(cur_role.get_role())
        switch(check_rselection_thread_.get_election_state())
        {
            //根据cur_role和election_state来决定执行什么操作
        }
}
```

集群切换

- 集群角色感知由主线程负责，当集群角色发生变化时，主线程中会根据情况分别调用由主切换为备或者由备切换为主的函数，由主切换为备和由备切换为主分别对应如下两个函数：

- **start_master_cluster()**
  在start_as_master_cluster()函数实现从备变成主的切换流程：
  1. 将当前RootServer的集群角色设为OBI_MASTER
  2. 设置原来的主集群中RS的集群角色为OBI_SLAVE
  3. 设置当前RootServer的关于master_root_server_ip和master_root_server_port的config信息
  4. 设置其他RootServer的关于master_root_server_ip和master_root_server_port的config信息
  5. 执行SQL语句：
     `ALTER system set master_root_server_ip='{当前ip}' server_type=rootserver,master_root_server_port={当前端口} server_type=rootserver;`
     以更新系统表中关于主RootServer的信息。

```
int ObRootWorker::start_master_cluster()
{
    // 1.将当前RootServer的集群角色设为OBI_MASTER
    role.set_role(common::ObiRole::MASTER);
    root_server_.set_obi_role(role);
    // 2.设置原来的主集群中RS的集群角色为OBI_SLAVE
    root_server_.set_slave_cluster_obi_role();
    // 3.设置当前RootServer的关于master_root_server_ip和master_root_server_port的config信息
    config_.add_extra_config(config_str, true);
    root_server_.commit_task(OBI_ROLE_CHANGE, OB_ROOTSERVER, rt_master_, rt_master_.get_port(), "rootserver", 1);
    // 4.设置其他RootServer的关于master_root_server_ip和master_root_server_port的config信息
    set_master_root_server_config_on_slave(config_string);
    // 5.更新系统表中主RootServer的信息
    root_server_.commit_task(CHANGE_MASTER_CLUSTER_ROOTSERVER, OB_ROOTSERVER, rs, rs.get_port(), server_version);
}
```

注：在集群发生切换时，所有的集群中的RootServer会重新向自己所在集群的lms汇报自己的集群角色，以dml类型的SQL语句修改内部表__all_cluster来修改主备集群的内部表信息。这是由RootServer发送replace类型的SQL语句给lms，由lms来执行它以此修改对应的内部表中相应记录。
在修改系统表中关于集群角色的信息过程中，并没有采用各个RootServer各自分别发送dml的方式来实现，而是由新的主RootServer去统一设置所有的系统表中关于集群角色的信息，这是为了防止有备RootServer不能正常执行dml而导致集群角色信息不能更新。主RootServer需要知道所有的备集群的RootServer的信息，为此增加了一个ObClusterMgr类来保存所有的备集群信息，这个类在全局中只有一个实例，它提供了多线程安全的访问函数和修改函数。

- **start_slave_cluster()**
  在start_as_slave_cluster()函数中只是将当前RootServer的角色设为OBI_SLAVE。

```
int ObRootWorker::start_slave_cluster()
{
    role.set_role(common::ObiRole::SLAVE);
    root_server_.set_obi_role(role);
}
```

- 当Raft角色发生变化时，就会调用start_master_cluster()或者start_slave_cluster()：
  - **OB_FOLLOWER => OB_LEADER**: start_master_cluster()
  - **OB_LEADER => OB_FOLLOWER**: start_slave_cluster()

租约机制

- 当主集群发生故障或者主集群所在的局部网络发生问题，原来的Leader会在租约过期后转变为Follower，同时选举模块会重新选举一个新的Leader。
- 在之前实现的主备切换功能中，可能会出现这样一个场景：当所有的RootServer同时感知到自己的Raft角色发生变化，就会同时发起切换，原来的主集群会将自己由主集群切换为备集群，新的Leader也会将自己由备集群切换为主集群，如果前者的速度没有后者快，就会发生同一时间内出现多个主集群的问题。
  为了解决这个问题，我们采用了这样一个方法，在主UpdateServer与主RootServer之间维持一个租约时间，这个租约时间比主RootServer与其他备RootServer之间的租约时间短200ms，也就是让主UpdateServer与主RootServer之间的租约过期时间点比主RootServer与备RootServer之间的租约过期时间点早200ms，当前者这个租约过期后，主UpdateServer会将自己切换为备UpdateServer，也就是让主UpdateServer比主RootServer提前200ms发生租约过期。同时，备RootServer在检查与主RootServer之间的租约时也会放宽300ms，也就是让备RootServer比主RootServer延后300ms过期。这样就可以让主RootServer与备RootServer之间的主备切换过程时长之差放宽至500ms之内，根据当前的测试运行结果统计得知主RootServer与备RootServer之间的切换时长之差总是小于500ms。
  这样当集群发生新的选举时就不会出现多个主集群同时存在的情况了。

3 模块外部接口

- 在集群初始化完成后，集群中的选举功能是被屏蔽的，需要DBA人工来设定主集群，选举模块中提供了首次设主的用户接口，调用方式为：
  `rs_admin -r {new_master_ip} -p {new_master_port} set_obi_master_first`

这个接口会设定RootServer的Raft角色，并打开选举模块的功能。

- 为了便于DBA维护集群，选举模块中提供一个用户接口来强制设定主的接口，接口调用方式为：
  `rs_admin -r {new_master_ip} -p {new_master_port} set_obi_master_rs`
  这个接口会设定每个集群的集群角色，并执行主备切换过程。

4 使用限制条件和注意事项

- 不支持故障自动恢复，当旧的主RootServer发生故障后重新选出新的主RootServer后，旧的主RootServer无法自动恢复，需要重启发生故障的Server来排除故障。
- 基础版本中当主RootServer发生故障后会直接宕机或者切换为备RootServer，此时系统将会进入无主的状态，必须等待人工手动指定新的主RootServer。当前实现的功能将会使备RootServer主动切换为主RootServer，可能产生的影响是若系统的网络不稳定时，频繁发生选举过程，就会频繁发生备RootServer切换为主RootServer的过程。