HA in MySQL Replication

mnstory.net

因为 aCloud 原有实现就是基于半同步机制的，所以我们讲了《MySQL Semi-Synchronous Replication》，明白了半同步机制，我们再来看看目前在复制机制下的高可用实现。

为了实现 HA（High Availability，高可用），引入了 VIP（Virtual IP，虚拟 IP）。

VIP

在 master/slave 的复制模型下，master 对外提供写服务，slave 可以不提供任何服务，只是做备份。

也有人将 slave 利用起来，提供读服务，就叫做读写分离，读写分离可以减轻单台服务器的压力。

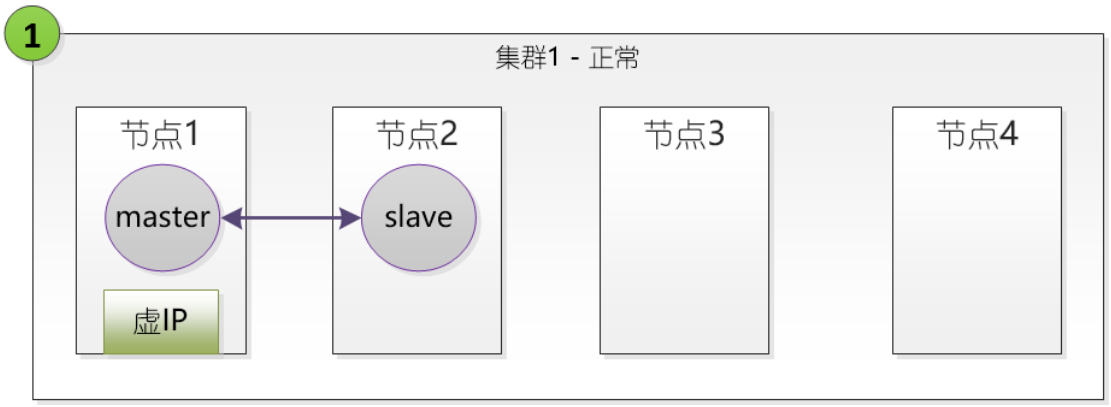
不管怎么说，半同步机制是单 master 多 slave 的模式，这种模式下，只能在 master 上提供写服务，如果 master 挂了，如何保证业务连续性？

VIP 原理类似 DNS，在 DNS 中我们每次链接到一个域名，每次解析域名得到可提供服务的 IP。VIP 的话，我们每次在能提供服务的主机上，配置一个固有 IP，如果这台主机出问题了，例如宕机了，就在其他能提供服务的节点，例如从节点配置该 IP，这样用户每次链接同一个 IP，也能实现高可用。

HA 场景

高可用并不代表无中断，它可能是短暂中断，例如，master 宕机时，VIP 要换节点，这个时候用户的事务就会提交失败，但是一般来说如果事务写得没有问题，这里不会存在啥问题，因为失败会返回到用户界面，相当于用户需要重试。

1. 正常情况
   我们理想的情况，应该是这样的：

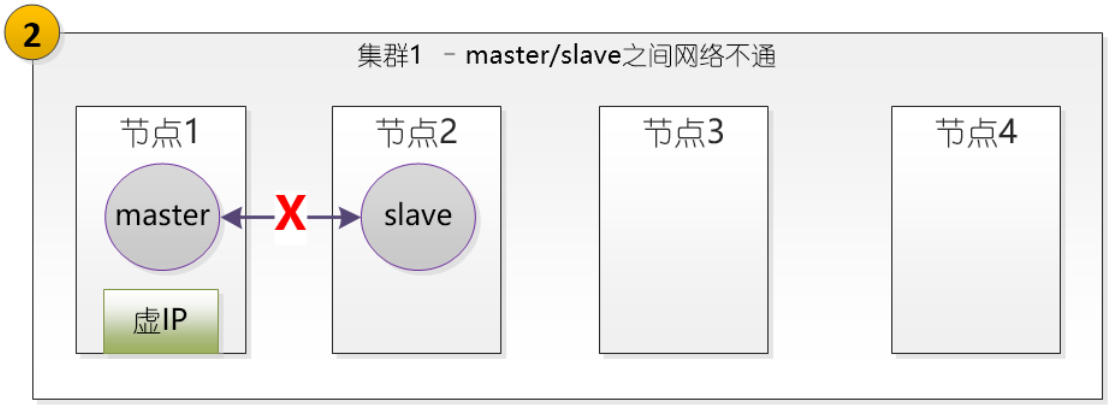


但是，也可能发生一些意外。

2. master/slave 之间网络不通

影响：这是最容易解决的意外，不通就不通也行，唯一带来的问题就是 slave 节点同步不到最新的 master 节点数据。

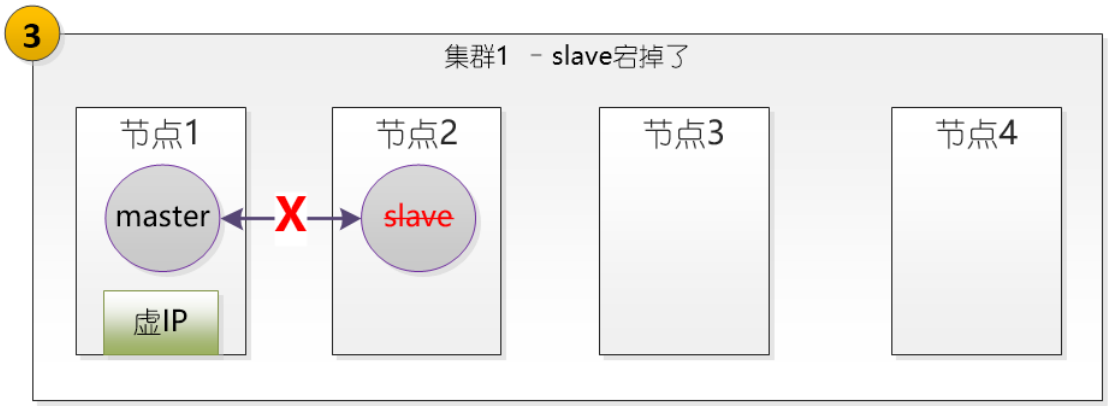
解决方法：如果出现这类问题，超出一定时间，给个告警到用户，让其检测节点之间硬件连接或者软件配置（例如防火墙）是比较好的解决方法。



3. slave 宕掉了

影响：slave 宕掉了，会导致不能同步最新的 master 数据，增加了整个高可用的风险，例如 master 再宕掉，整个集群就没法提供服务了。

解决方法：重新启动 slave 上的 MySQL 或者重新启动节点即可，当然，如果节点没办法恢复了，这时候，需要用其他节点来替代该节点，提供 slave 功能，如果找不到可用节点，应该及时通知用户。如果 slave 还提供了读服务，那还要考虑将读服务转移到 master。

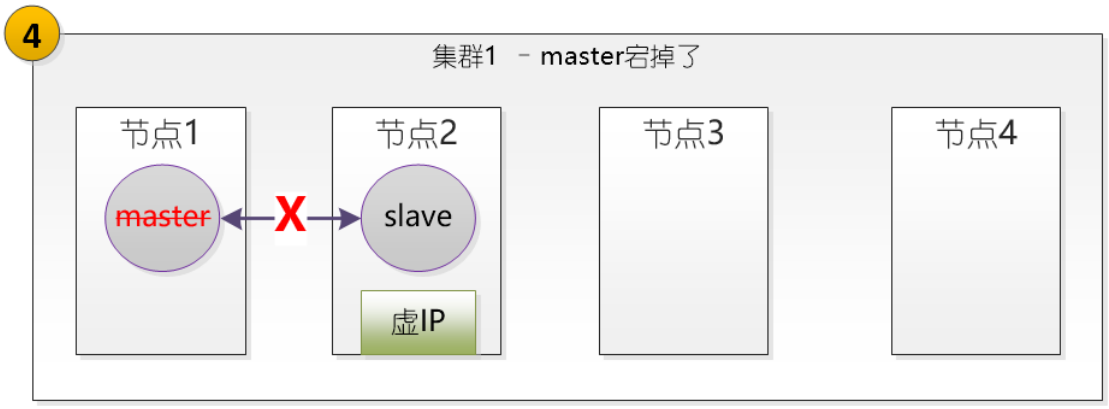


4. master 宕掉了

半同步模型下，可能存在 slave 和 master 之间有延迟，导致数据丢失，如果数据非常重要，建议设置 sync_binlog 和 useinnodb_flush_log_at_trx_commit 为 1，这样设置了，也不一定能保证数据不丢，master 端最多影响一条日志，slave 端不一定，看 IO 同步线程是否能赶得上日志记录速度。

影响： 短暂的切换过程不能提供服务，并且可能出现数据丢失。

解决方法：在第三方监控的帮助下(例如 keepalive),slave 可以顺利接替 master 提供服务，VIP shift 到原 slave 节点(节点 2)，原 slave(节点 2)变成 master，当原 master(节点 1)再次启动的时候，最好是以 slave 身份启动。



5. 脑裂

这里出现了集群脑裂，我们不认为是用户参与并分裂的集群，而是一个临时状态，两者的差异在于：

如果是用户参与并分裂的集群，我们可以很明确，两个集群之间可以互不干涉，各自配置自己的 VIP 和 master/slave。

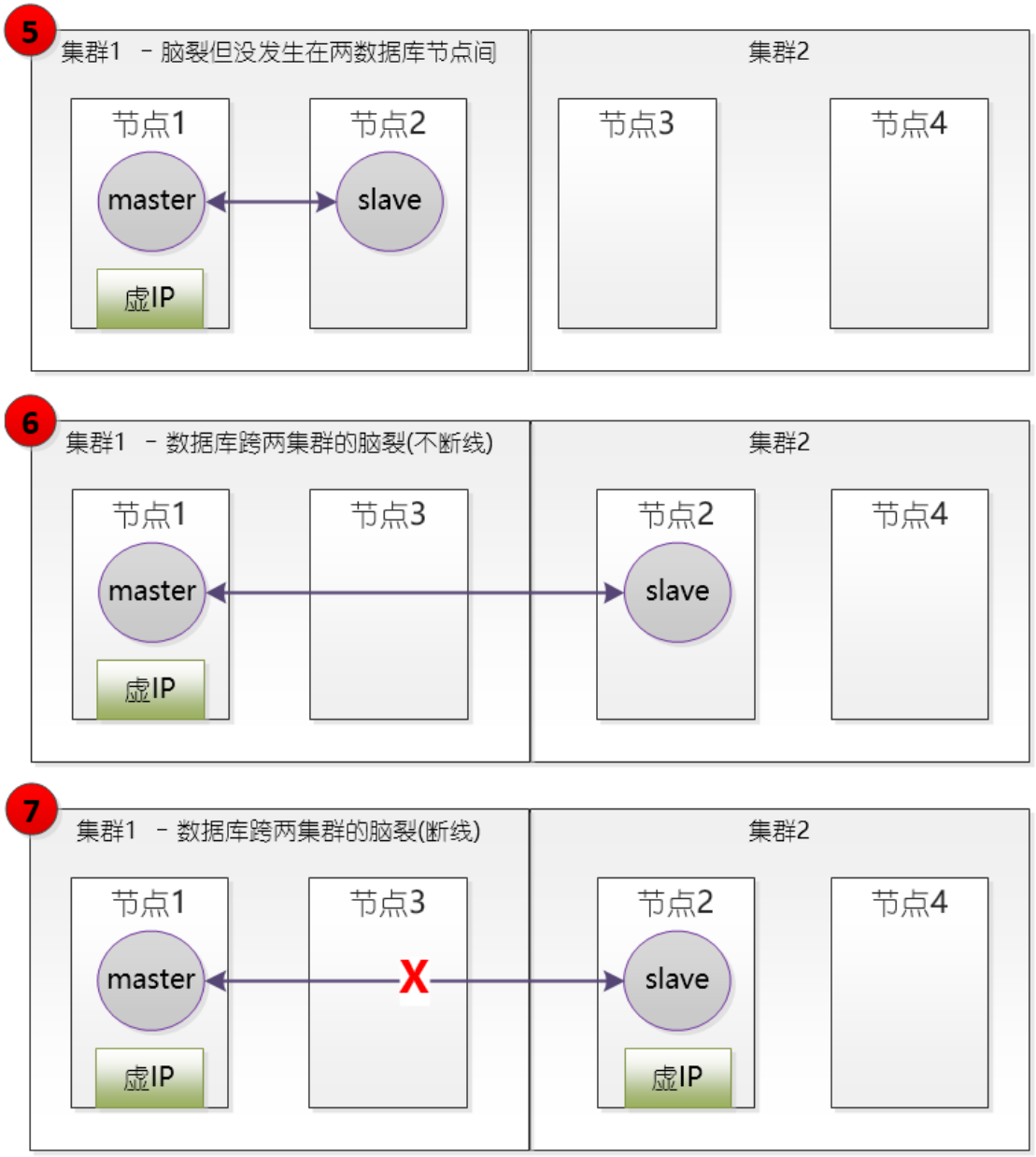
如果是临时状态，我们的目的是尽可能保证数据不出问题的情况下部分功能可用，后面故障排除后，需要重新合并。在用户参与之前或自动恢复之前，都是出于临时状态。

我们设想一个极端的情况，管理面的脑裂和数据库脑裂，是可交错叠加的。

如图 5、图 6，为管理面的脑裂，数据库层面并没有脑裂。
而图 7 为管理面和数据库层面都脑裂的情况。

如果要保证脑裂后的两个集群可用，那数据库必须提供服务，但是考虑都后续还要对集群进行合并，所以要保证数据无冲突，那么数据库不能和集群一样分裂为两套。相对于保服务还是保数据一致性，一般倾向于保数据一致性。

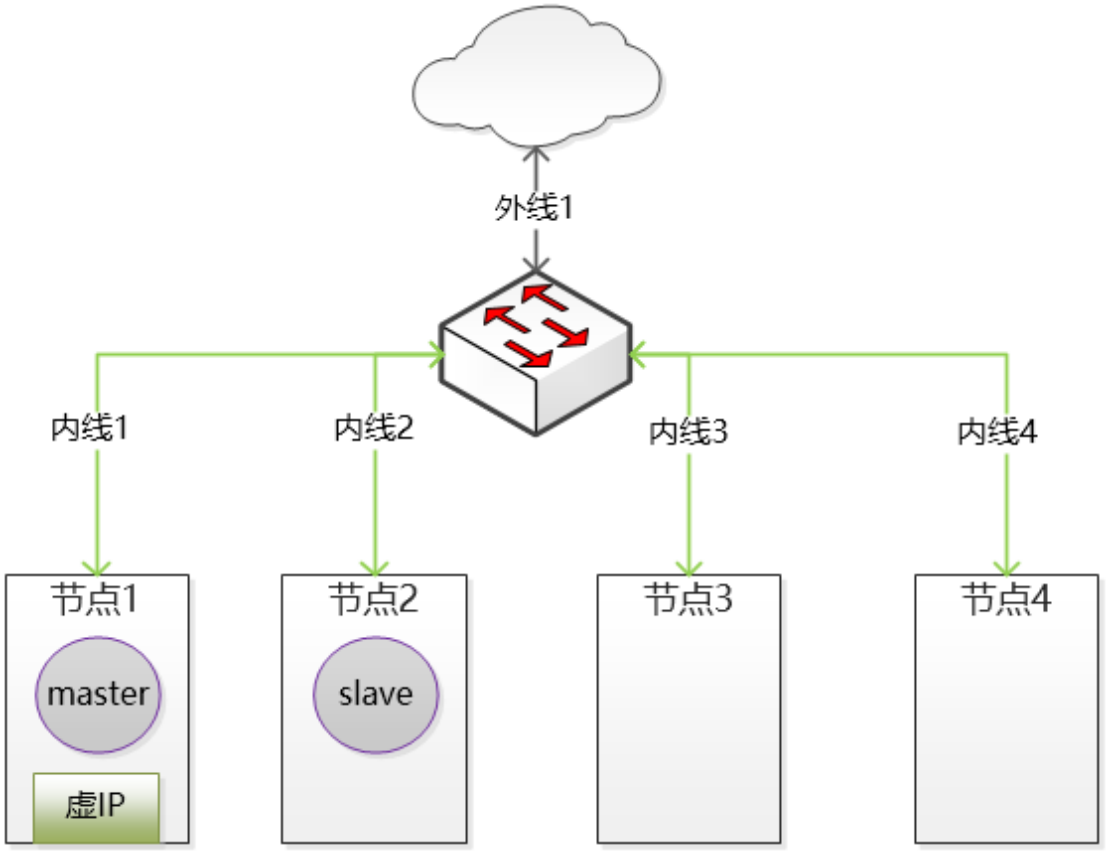
影响：如果集群 2 能访问 VIP，可能出现集群 2 和集群 1 写入数据冲突的情况，导致数据一致性收到影响。如果主从之间也不能访问，那么会降低高可用的抵抗能力。
解决方法：利用防火墙规则，阻止集群外的访问，保一致性丢服务可用性。



物理布局

抛开物理布局说高可用，虽然场景考虑的比较全，但是还是比较理想化，而现实中布局非常复杂，我们选择当前最常用的两种场景来说一下，故障可能发生的位置。

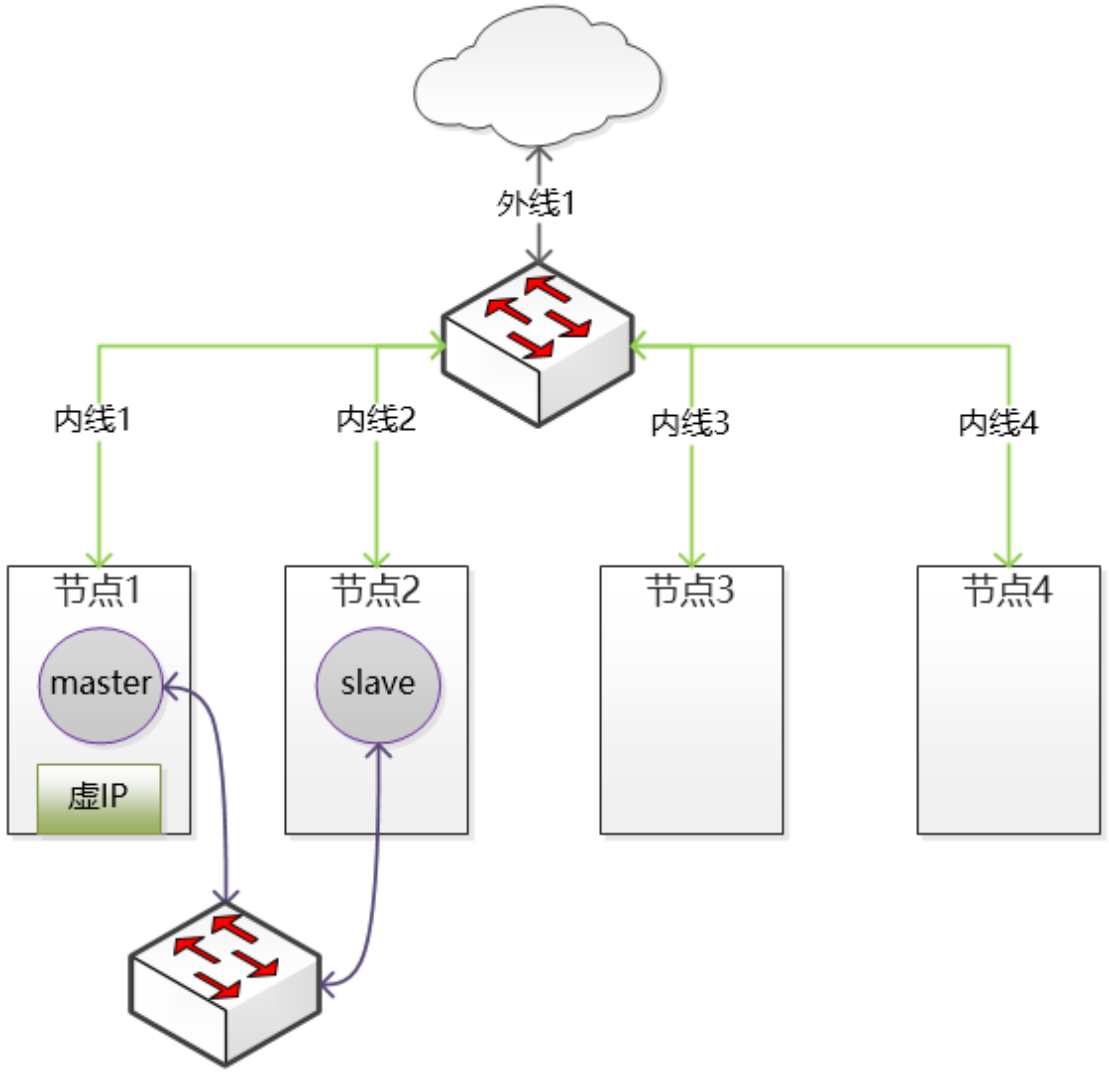
管理口和数据口同交换



1．在集群正常的时候，数据库的 Master 和 Slave 选择和数据库服务的启停，都由主控统一管理。
2．如果人工介入，假定所有问题都能解决。
3．在集群不正常的时候，参考下表处理：

序号	Master现象	Master发生时行为	Slave现象	Slave行为	集群检测到现象消除后	描述
1	同任何节点都不通	停止数据库	同Master不通 除Master外的非所有节点能通	不做任何处理	原Slave变成Master，为其找新的Slave	Master节点的网口坏掉、节点与交换机之间的连线坏掉、者交换机上该节点的网口坏掉
2	同任何节点都不通	停止数据库	同Master不通 除Master外的所有节点能通	变成新的Master	保持新Master身份不变，为其找新的Slave	Master节点的网口坏掉、节点与交换机之间的连线坏掉、者交换机上该节点的网口坏掉
3	同Slave不通	不做任何处理	同任何节点都不通	不做任何处理	不做任何处理，会自动重新同步	Slave节点的网口坏掉、节点与交换机之间的连线坏掉、者交换机上该节点的网口坏掉
4			同Master不通 除Master外的所有节点能通	变成新的Master	保持新Master身份不变，为其找新的Slave	Master系统坏了或者宕机了
5	同Slave不通 除Slave外的所有节点能通	找新的Slave			不做任何处理	Slave系统坏了或者宕机了
6	同任何节点都不通	停止数据库	同任何节点都不通	不做任何处理	重新启动Master和Slave	交换机坏了

管理口和数据口不同交换



1．如果数据通道正常，管理通道不正常，不需要检测。
2．如果两者皆不正常，按照管理口与数据口同交换方式处理。
3．如果管理口的交换机正常，而数据库的交换机或线路不正常：
   a）管理面评估出到底哪些节点之间数据通道正常的，如果满足大于等于两个节点的数据通道正常（选择节点的时候优先选择原来的主从，如果原来的主从都不正常，数据需要从管理网络传到其他节点）。
   b）如果上面方法行不通，再让 VIP 走管理通道，然后等待人工修复。

2016/11/23