25 | 高可用存储架构：双机架构

2018-06-23 李运华

从0开始学架构 进入课程 >

讲述：黄洲君
时长 15:29 大小 7.10M

存储高可用方案的本质都是通过将数据复制到多个存储设备，通过数据冗余的方式来实现高可用，其复杂性主要体现在如何应对复制延迟和中断导致的数据不一致问题。因此，对任何一个高可用存储方案，我们需要从以下几个方面去进行思考和分析：

数据如何复制？

各个节点的职责是什么？

如何应对复制延迟？

如何应对复制中断？

常见的高可用存储架构有主备、主从、主主、集群、分区，每一种又可以根据业务的需求进行一些特殊的定制化功能，由此衍生出更多的变种。由于不同业务的定制功能难以通用化，

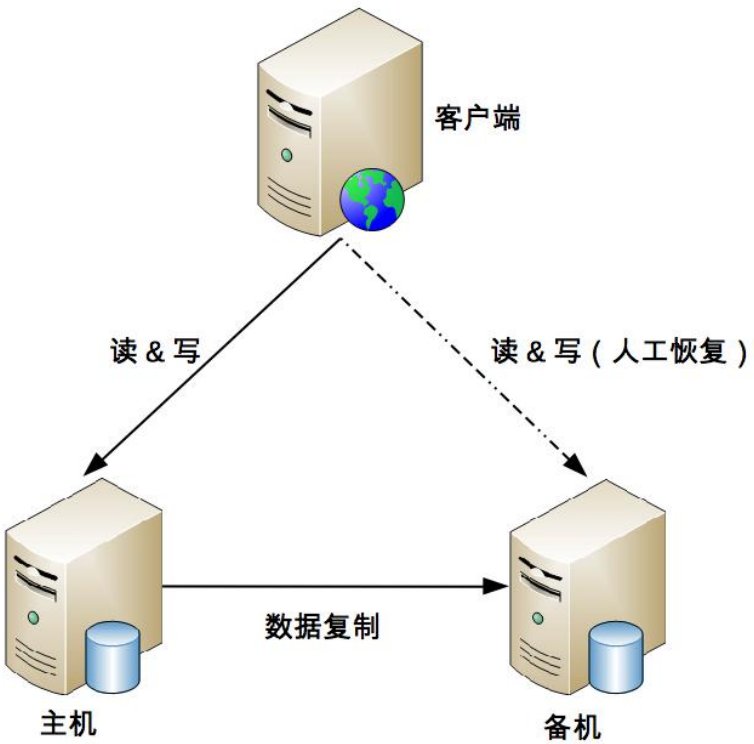
今天我将针对业界通用的方案，来分析常见的双机高可用架构：主备、主从、主备 / 主从切换和主主。

主备复制

主备复制是最常见也是最简单的一种存储高可用方案，几乎所有的存储系统都提供了主备复制的功能，例如 MySQL、Redis、MongoDB 等。

1. 基本实现

下面是标准的主备方案结构图：



其整体架构比较简单，主备架构中的“备机”主要还是起到一个备份作用，并不承担实际的业务读写操作，如果要把备机改为主机，需要人工操作。

2. 优缺点分析

主备复制架构的优点就是简单，表现有：

对于客户端来说，不需要感知备机的存在，即使灾难恢复后，原来的备机被人工修改为主机后，对于客户端来说，只是认为主机的地址换了而已，无须知道是原来的备机升级为主机。

对于主机和备机来说，双方只需要进行数据复制即可，无须进行状态判断和主备切换这类复杂的操作。

主备复制架构的缺点主要有：

备机仅仅只为备份，并没有提供读写操作，硬件成本上有浪费。

故障后需要人工干预，无法自动恢复。人工处理的效率是很低的，可能打电话找到能够操作的人就耗费了 10 分钟，甚至如果是深更半夜，出了故障都没人知道。人工在执行恢复操作的过程中也容易出错，因为这类操作并不常见，可能 1 年就 2、3 次，实际操作的时候很可能遇到各种意想不到的问题。

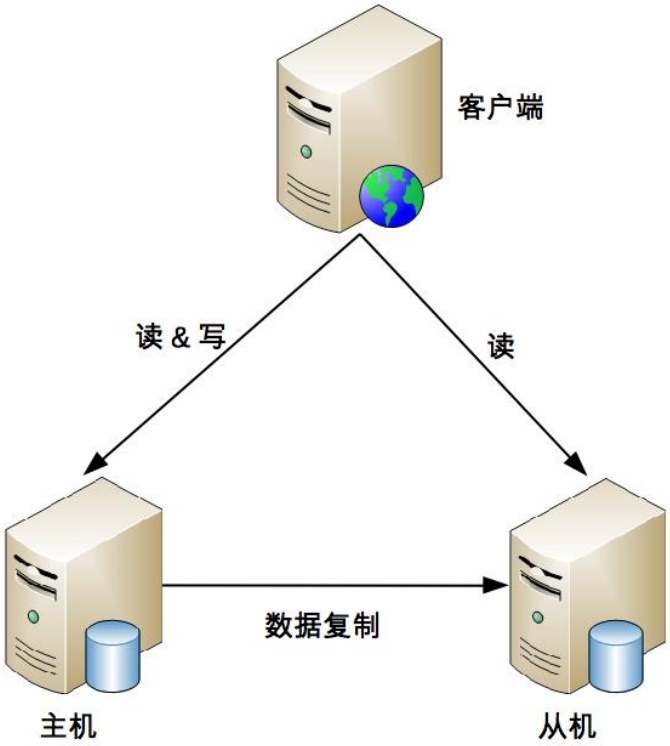
综合主备复制架构的优缺点，内部的后台管理系统使用主备复制架构的情况会比较多，例如学生管理系统、员工管理系统、假期管理系统等，因为这类系统的数据变更频率低，即使在某些场景下丢失数据，也可以通过人工的方式补全。

主从复制

主从复制和主备复制只有一字之差，“从”意思是“随从、仆从”，“备”的意思是备份。我们可以理解为仆从是要帮主人干活的，这里的干活就是承担“读”的操作。也就是说，主机负责读写操作，从机只负责读操作，不负责写操作。

1. 基本实现

下面是标准的主从复制架构：



与主备复制架构比较类似，主要的差别点在于从机正常情况下也是要提供读的操作。

2. 优缺点分析

主从复制与主备复制相比，优点有：

主从复制在主机故障时，读操作相关的业务可以继续运行。

主从复制架构的从机提供读操作，发挥了硬件的性能。

缺点有：

主从复制架构中，客户端需要感知主从关系，并将不同的操作发给不同的机器进行处理，复杂度比主备复制要高。

主从复制架构中，从机提供读业务，如果主从复制延迟比较大，业务会因为数据不一致出现问题。

故障时需要人工干预。

综合主从复制的优缺点，一般情况下，写少读多的业务使用主从复制的存储架构比较多。例如，论坛、BBS、新闻网站这类业务，此类业务的读操作数量是写操作数量的 10 倍甚至 100 倍以上。

双机切换

1. 设计关键

主备复制和主从复制方案存在两个共性的问题：

主机故障后，无法进行写操作。

如果主机无法恢复，需要人工指定新的主机角色。

双机切换就是为了解决这两个问题而产生的，包括主备切换和主从切换两种方案。简单来说，这两个方案就是在原有方案的基础上增加“切换”功能，即系统自动决定主机角色，并完成角色切换。由于主备切换和主从切换在切换的设计上没有差别，我接下来以主备切换为例，一起来看看双机切换架构是如何实现的。

要实现一个完善的切换方案，必须考虑这几个关键的设计点：

主备间状态判断

主要包括两方面：状态传递的渠道，以及状态检测的内容。

状态传递的渠道：是相互间互相连接，还是第三方仲裁？

状态检测的内容：例如机器是否掉电、进程是否存在、响应是否缓慢等。

切换决策

主要包括几方面：切换时机、切换策略、自动程度。

切换时机：什么情况下备机应该升级为主机？是机器掉电后备机才升级，还是主机上的进程不存在就升级，还是主机响应时间超过 2 秒就升级，还是 3 分钟内主机连续重启 3 次就升级等。

切换策略：原来的主机故障恢复后，要再次切换，确保原来的主机继续做主机，还是原来的主机故障恢复后自动成为新的备机？

自动程度：切换是完全自动的，还是半自动的？例如，系统判断当前需要切换，但需要人工做最终的确认操作（例如，单击一下“切换”按钮）。

数据冲突解决

当原有故障的主机恢复后，新旧主机之间可能存在数据冲突。例如，用户在旧主机上新增了一条 ID 为 100 的数据，这个数据还没有复制到旧的备机，此时发生了切换，旧的备机升级为新的主机，用户又在新的主机上新增了一条 ID 为 100 的数据，当旧的故障主机恢复后，这两条 ID 都为 100 的数据，应该怎么处理？

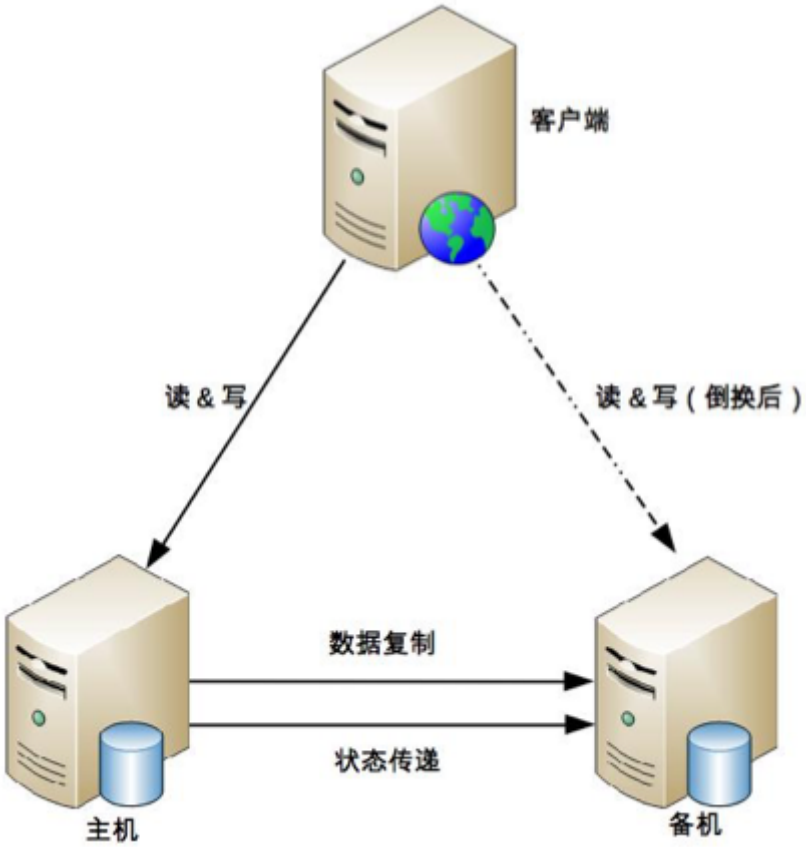
以上设计点并没有放之四海而皆准的答案，不同的业务要求不一样，所以切换方案比复制方案不只是多了一个切换功能那么简单，而是复杂度上升了一个量级。形象点来说，如果复制方案的代码是 1000 行，那么切换方案的代码可能就是 10000 行，多出来的那 9000 行就是用于实现上面我所讲的 3 个设计点的。

2. 常见架构

根据状态传递渠道的不同，常见的主备切换架构有三种形式：互连式、中介式和模拟式。

互连式

故名思议，互连式就是指主备机直接建立状态传递的渠道，架构图请注意与主备复制架构对比。



你可以看到，在主备复制的架构基础上，主机和备机多了一个“状态传递”的通道，这个通道就是用来传递状态信息的。这个通道的具体实现可以有很多方式：

可以是网络连接（例如，各开一个端口），也可以是非网络连接（用串口线连接）。

可以是主机发送状态给备机，也可以是备机到主机来获取状态信息。

可以和数据复制通道共用，也可以独立一条通道。

状态传递通道可以是一条，也可以是多条，还可以是不同类型的通道混合（例如，网络 + 串口）。

为了充分利用切换方案能够自动决定主机这个优势，客户端这里也会有一些相应的改变，常见的方式有：

为了切换后不影响客户端的访问，主机和备机之间共享一个对客户端来说唯一的地址。例如虚拟 IP，主机需要绑定这个虚拟的 IP。

客户端同时记录主备机的地址，哪个能访问就访问哪个；备机虽然能收到客户端的操作请求，但是会直接拒绝，拒绝的原因就是“备机不对外提供服务”。

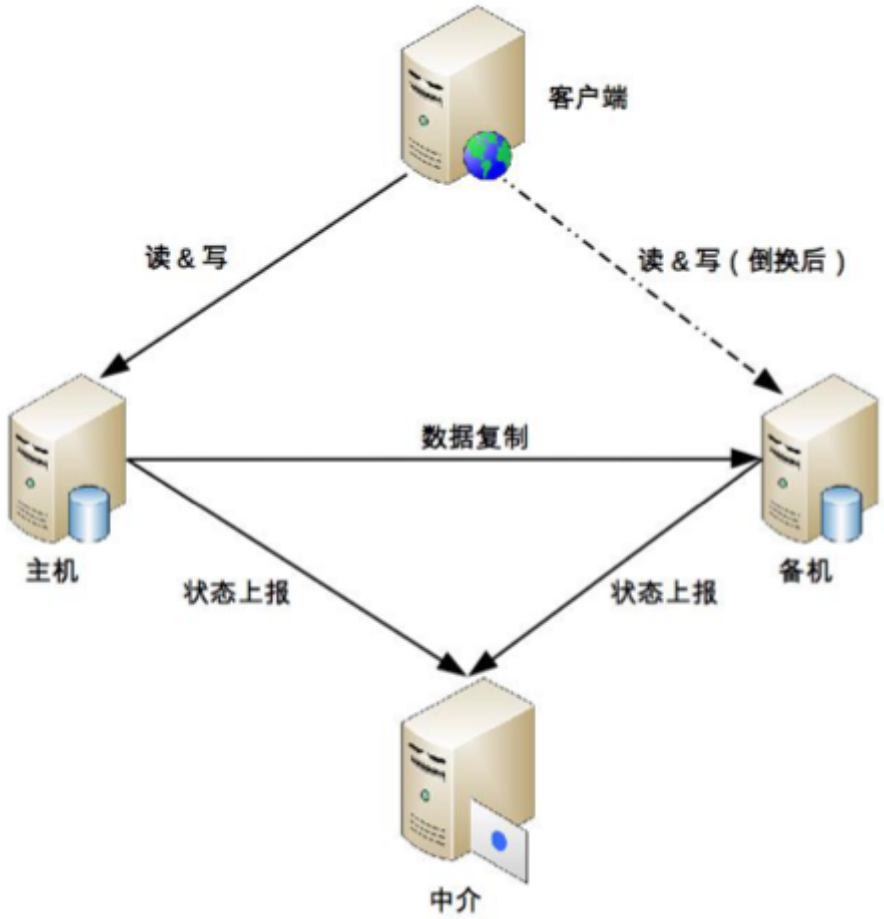
互连式主备切换主要的缺点在于：

如果状态传递的通道本身有故障（例如，网线被人不小心踢掉了），那么备机也会认为主机故障了从而将自己升级为主机，而此时主机并没有故障，最终就可能出现两个主机。

虽然可以通过增加多个通道来增强状态传递的可靠性，但这样做只是降低了通道故障概率而已，不能从根本上解决这个缺点，而且通道越多，后续的状态决策会更加复杂，因为对备机来说，可能从不同的通道收到了不同甚至矛盾的状态信息。

中介式

中介式指的是在主备两者之外引入第三方中介，主备机之间不直接连接，而都去连接中介，并且通过中介来传递状态信息，其架构图如下：



对比一下互连式切换架构，我们可以看到，主机和备机不再通过互联通道传递状态信息，而是都将状态上报给中介这一角色。单纯从架构上看，中介式似乎比互连式更加复杂了，首先要引入中介，然后要各自上报状态。然而事实上，中介式架构在状态传递和决策上却更加简单了，这是为何呢？

连接管理更简单：主备机无须再建立和管理多种类型的状态传递连接通道，只要连接到中介即可，实际上是降低了主备机的连接管理复杂度。

例如，互连式要求主机开一个监听端口，备机来获取状态信息；或者要求备机开一个监听端口，主机推送状态信息到备机；如果还采用了串口连接，则需要增加串口连接管理和数据读取。采用中介式后，主备机都只需要把状态信息发送给中介，或者从中介获取对方的状态信息。无论是发送还是获取，主备机都是作为中介的客户端去操作，复杂度会降低。

状态决策更简单：主备机的状态决策简单了，无须考虑多种类型的连接通道获取的状态信息如何决策的问题，只需要按照下面简单的算法即可完成状态决策。

无论是主机还是备机，初始状态都是备机，并且只要与中介断开连接，就将自己降级为备机，因此可能出现双备机的情况。

主机与中介断连后，中介能够立刻告知备机，备机将自己升级为主机。

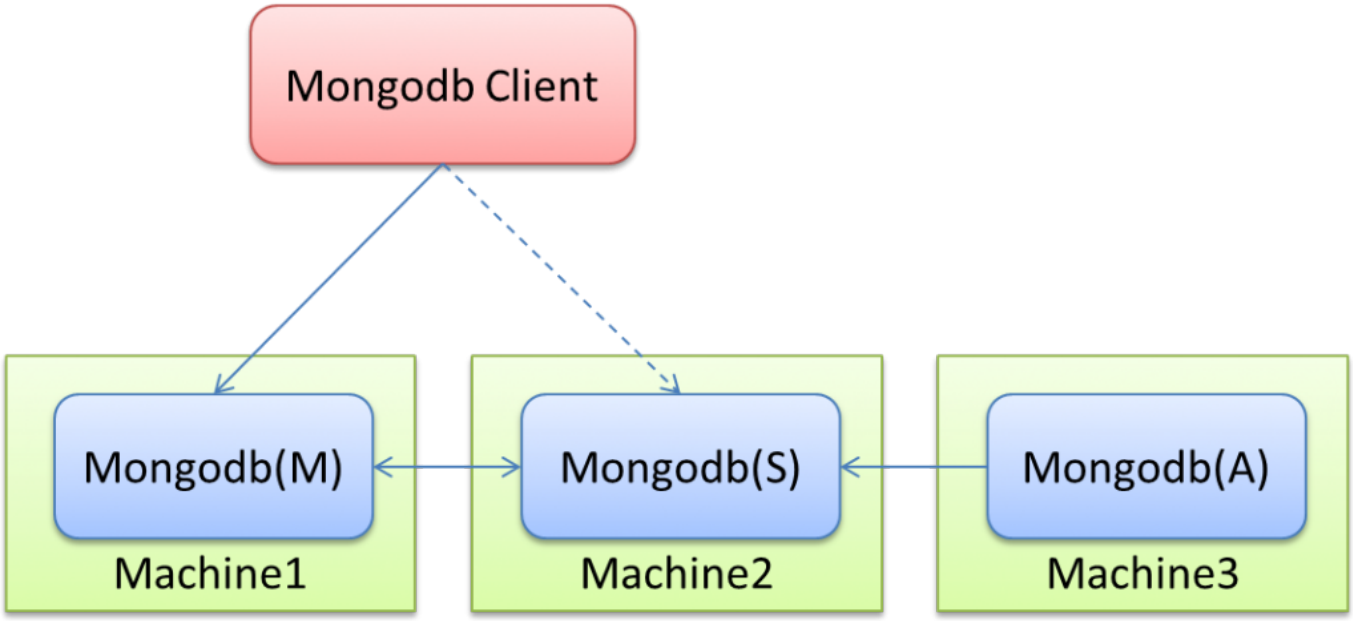
如果是网络中断导致主机与中介断连，主机自己会降级为备机，网络恢复后，旧的主机以新的备机身份向中介上报自己的状态。

如果是掉电重启或者进程重启，旧的主机初始状态为备机，与中介恢复连接后，发现已经有主机了，保持自己备机状态不变。

主备机与中介连接都正常的情况下，按照实际的状态决定是否进行切换。例如，主机响应时间超过 3 秒就进行切换，主机降级为备机，备机升级为主机即可。

虽然中介式架构在状态传递和状态决策上更加简单，但并不意味着这种优点是没有代价的，其关键代价就在于如何实现中介本身的高可用。如果中介自己宕机了，整个系统就进入了双备的状态，写操作相关的业务就不可用了。这就陷入了一个递归的陷阱：为了实现高可用，我们引入中介，但中介本身又要求高可用，于是又要设计中介的高可用方案……如此递归下去就无穷无尽了。

MongoDB 的 Replica Set 采取的就是这种方式，其基本架构如下：



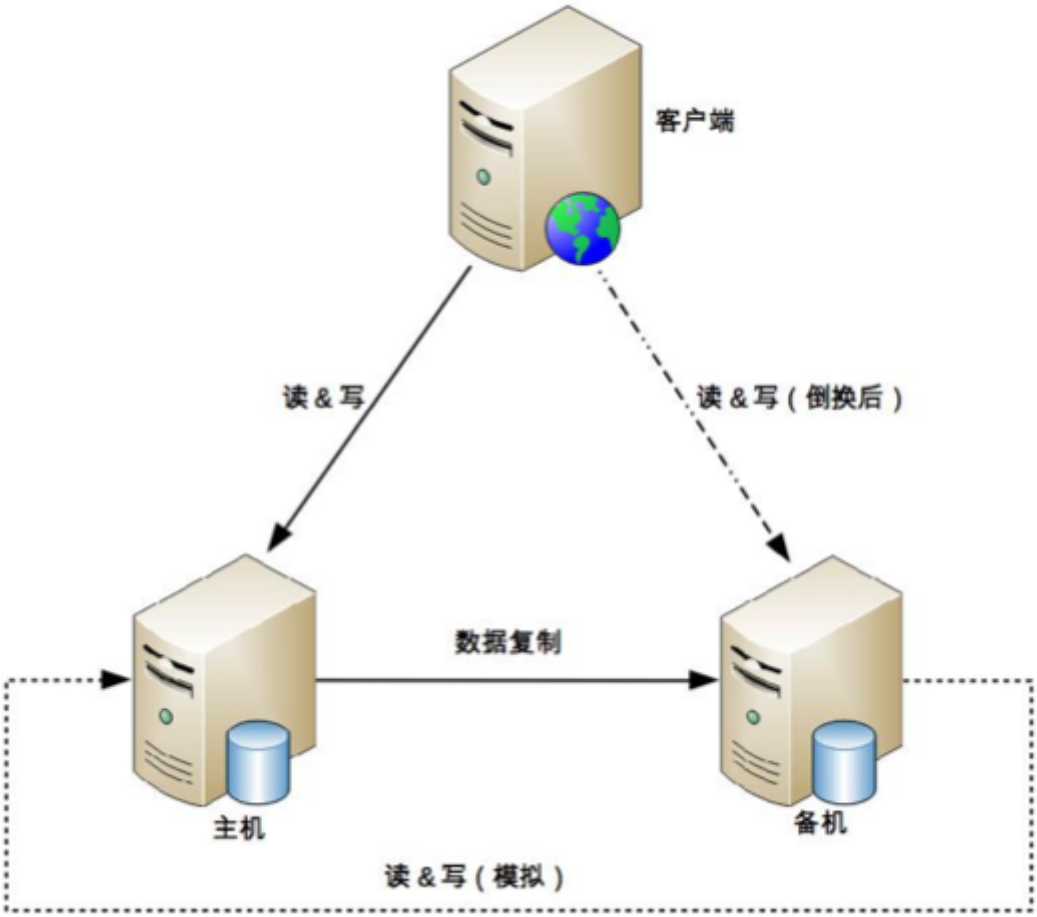
（http://img.my.csdn.net/uploads/201301/13/1358056331_2790.png）

MongoDB(M) 表示主节点，MongoDB(S) 表示备节点，MongoDB(A) 表示仲裁节点。主备节点存储数据，仲裁节点不存储数据。客户端同时连接主节点与备节点，不连接仲裁节点。

幸运的是，开源方案已经有比较成熟的中介式解决方案，例如 ZooKeeper 和 Keepalived。ZooKeeper 本身已经实现了高可用集群架构，因此已经帮我们解决了中介本身的可靠性问题，在工程实践中推荐基于 ZooKeeper 搭建中介式切换架构。

模拟式

模拟式指主备机之间并不传递任何状态数据，而是备机模拟成一个客户端，向主机发起模拟的读写操作，根据读写操作的响应情况来判断主机的状态。其基本架构如下：



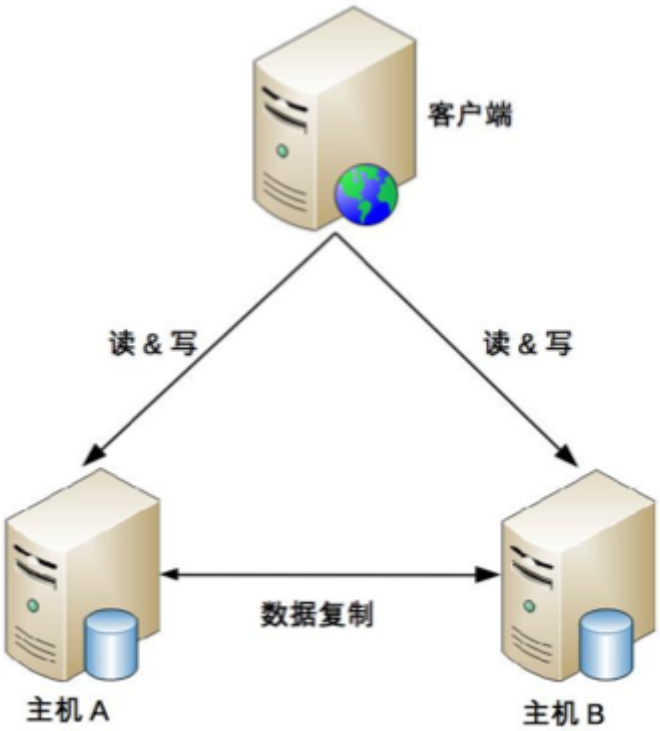
对比一下互连式切换架构，我们可以看到，主备机之间只有数据复制通道，而没有状态传递通道，备机通过模拟的读写操作来探测主机的状态，然后根据读写操作的响应情况来进行状态决策。

模拟式切换与互连式切换相比，优点是实现更加简单，因为省去了状态传递通道的建立和管理工作。

简单既是优点，同时也是缺点。因为模拟式读写操作获取的状态信息只有响应信息（例如，HTTP 404，超时、响应时间超过 3 秒等），没有互连式那样多样（除了响应信息，还可以包含 CPU 负载、I/O 负载、吞吐量、响应时间等），基于有限的状态来做状态决策，可能出现偏差。

主主复制

主主复制指的是两台机器都是主机，互相将数据复制给对方，客户端可以任意挑选其中一台机器进行读写操作，下面是基本架构图。



相比主备切换架构，主主复制架构具有如下特点：

两台都是主机，不存在切换的概念。

客户端无须区分不同角色的主机，随便将读写操作发送给哪台主机都可以。

从上面的描述来看，主主复制架构从总体上来看要简单很多，无须状态信息传递，也无须状态决策和状态切换。然而事实上主主复制架构也并不简单，而是有其独特的复杂性，具体表现在：如果采取主主复制架构，必须保证数据能够双向复制，而很多数据是不能双向复制的。例如：

用户注册后生成的用户 ID，如果按照数字增长，那就不能双向复制，否则就会出现 X 用户在主机 A 注册，分配的用户 ID 是 100，同时 Y 用户在主机 B 注册，分配的用户 ID 也是 100，这就出现了冲突。

库存不能双向复制。例如，一件商品库存 100 件，主机 A 上减了 1 件变成 99，主机 B 上减了 2 件变成 98，然后主机 A 将库存 99 复制到主机 B，主机 B 原有的库存 98 被覆盖，变成了 99，而实际上此时真正的库存是 97。类似的还有余额数据。

因此，主主复制架构对数据的设计有严格的要求，一般适合于那些临时性、可丢失、可覆盖的数据场景。例如，用户登录产生的 session 数据（可以重新登录生成）、用户行为的日志数据（可以丢失）、论坛的草稿数据（可以丢失）等。

小结

今天我为你讲了高可用存储架构中常见的双机架构，分析了每类架构的优缺点以及适应场景，希望对你有所帮助。

这就是今天的全部内容，留一道思考题给你吧，如果你来设计一个政府信息公开网站的信息存储系统，你会采取哪种架构？谈谈你的分析和理由。

欢迎你把答案写到留言区，和我一起讨论。相信经过深度思考的回答，也会让你对知识的理解更加深刻。（编辑乱入：精彩的留言有机会获得丰厚福利哦！）

上一篇 24 | FMEA方法，排除架构可用性隐患的利器

下一篇 26 | 高可用存储架构：集群和分区

精选留言 (46)

写留言

空档滑行

2018-06-23

30

政府信息网站使用主备或者主从架构就可以了。信息都是人工录入，可以补录。数据本来对实时性要求不高，所以出了故障人工修复也来得及。所以主备就够了，如果为了照顾形象可以用主从，保证主机故障后仍然可以查，不能新发

展开

作者回复: 分析正确

文竹

2018-08-22

5

政府信息公开网站的特点：
1、用户量和QPS不会很大
2、其次读/写都非常少，读相较于写多
3、可忍受一定时间范围的不可用
…

展开

作者回复: 正确

姜戈

2018-12-13

4

网上搜索了一下：
在软件系统的高可靠性（也称为可用性，英文描述为HA，High Available）里有个衡量其可靠性的标准——X个9，这个X是代表数字3~5。X个9表示在软件系统1年时间的使用过程中，系统可以正常使用时间与总时间（1年）之比，我们通过下面的计算来感受下X个9在不同级别的可靠性差异。…

展开

南友力max...

2018-07-16

4

单机就可以了，搞那么复杂

展开

作者回复: 单机可靠性只有2个9

今夕是何年 4
2018-06-23

政府信息网站使用主从就行了，因为读的请求多，写的请求少。
网站挂掉影响也不大，所以可以不用主从切换。

展开

忠厚 2
2018-06-27

数据持久化信息我可能会选择主备模式，备机主做数据备份不提供读写操作。

添加一个redis缓存全量信息数据，做一个哨兵模式，实现故障切换，提高网站的可用性

应用上再使用个Ehcache堆外缓存，主要把热点信息放到应用里提升性能....

展开

作者回复: 缓存设计得比较复杂了，我认为ehcache没有必要

叶伟 1
2018-06-24

政府网站特殊性，更多属于公告通知类的内容，使用主备即可，读的高频率可以用缓存、cdn等其他方式实现，遇到突发事件访问量过大时，避免让压力直接落在db上

gen_jin 1
2018-06-23

我认为对政府信息系统：
1. 由于数据写少读多（1：10000）：采用主从复制（利用从机读）而不是主备。
2.由于面对公众性，最好24小时无间断工作，出现故障最好采用自动双机切换；而考虑将来扩容，开始是一主一从 后面是一主多从，对一主多从 实现简单看最好中介式（使用zk或LVS+Keepalived的架构 实现一主多从）。...

展开

作者回复: 用了主从复制即可，没必要切换，因为写很少

阿鼎
2018-06-23

1

主备机与第三方中介断网，存在的双备问题，可以使系统中所有节点当做仲裁者，过半仲裁主备状态。这就好像paxos，raft中的选举，但数据仅在主备之间一致。

展开

作者回复: 系统只有两个节点，你说的是集群方案，后面会介绍

张飞洪
2019-05-12

请问李老师，假如我使用的是阿里云数据库，还有必要考虑高可用吗？云本身有自带数量备份了吧？

作者回复: 可以考虑怎么用

行者
2019-03-06

大开眼界。

展开

gkb111
2019-03-06

存储高可用，可以分为主备，主从，读远大于写。故障需要人工干预，双机切换，中介式，如zookeeper，

发条橙子 ...
2019-02-03

主从还要读写分离 ，相对来说会比主备复杂 。 并且也不需要读写分离的场景 。并且政府部门都比较有钱， 所以主备就可以了 。

展开

张汉桂-东...

2019-02-01

我有个疑问，最后的主主复制不是可以用分布式事务实现的更简单些吗？所有的插入和更新操作同时在两台机上执行，要么一起成功，要么一起失败。若果我的想法不对，望指点迷津。

作者回复: 1. 不能充分利用机器性能
2. 分布式事务不简单，很复杂，你尝试考虑各种情况试试，例如一个成功一个失败，然后撤销的时候有有失败

小狮子辛巴

2018-11-19

主 + 从 + （备份）

写信息的请求很少，远远小于读信息的量。
而且我感觉对写的高可用要求不大，
一主一从，就可以。...

展开

劉阳河

2018-11-04

老师我是这么理解的，数据库主从实现读写分离保证了数据库的高性能，但是没有保证数据库的高可用，主机崩掉后，数据库不在提供写服务，而从库只能进行读操作，拒绝写服务。所以需要在主从之间加入主从切换的规则，当主机崩掉后，从机可以进行自主升为主机，从而保证高可用，这种机制就可以保证数据库的高性能和高可用，但是感觉这种复杂度太高了，真的会使用这种架构吗？

展开

作者回复: 用得很多啊😄

大熊

2018-10-29

老师，你好。数据冲突解决，下文好像没怎么讲具体的解决方案，能否帮忙讲解下呢，谢谢

作者回复: 目前出现数据冲突都是靠人工修复，或者覆盖一些数据，或者直接丢失一些数据，异地多活的章节会讲

奋斗心

2018-09-28

不知道oracle rac属于上面的哪个模式？政府网站感觉用oracle rac加磁盘阵列应该能满足要求

作者回复: 对Oracle不熟悉，你可以自己查查，理解会更深刻。政府网站用oracle rac有点大材小用，成本太高

Geek_5b3cc...

2018-09-26

我看到您说单机可靠性只有两个9，请问些两个9是怎么算出来的？我们平时说的3-5个9具体怎么计算出来呢？

作者回复: 两个9不是算出来的，统计出来的，平常说的几个9，具体上网搜“可靠性指标”

Godaday

2018-09-19

政府信息公开网站的信息系统
1. 读多写少
2. 保证读的稳定，涉及到民众对权力机构的印象。
3.采用人工恢复，可能延时比较高。
所以DB使用主从复制的架构，程序读写分离的方式来保证。Web 做三台 ，nginx 做负...
展开

作者回复: 这样也可以，但还有更简单的更合适的，不用读写分离，用缓存就可以了