Dmall MySQL集群高可用实践

冯光普 @ 多点Dmall

2019-04

About me

- 『前』阿里MySQL内核研发
  - Thread pool / Statement timeout
  - SELECT … FROM UPDATE / Thread running ctl
  - TokuDB / Galera Cluster
- 多点Dmall数据库负责人
  - MySQL、PG、Redis、MongoDB
  - 数据库自动化运维平台建设

多点 dmall.com

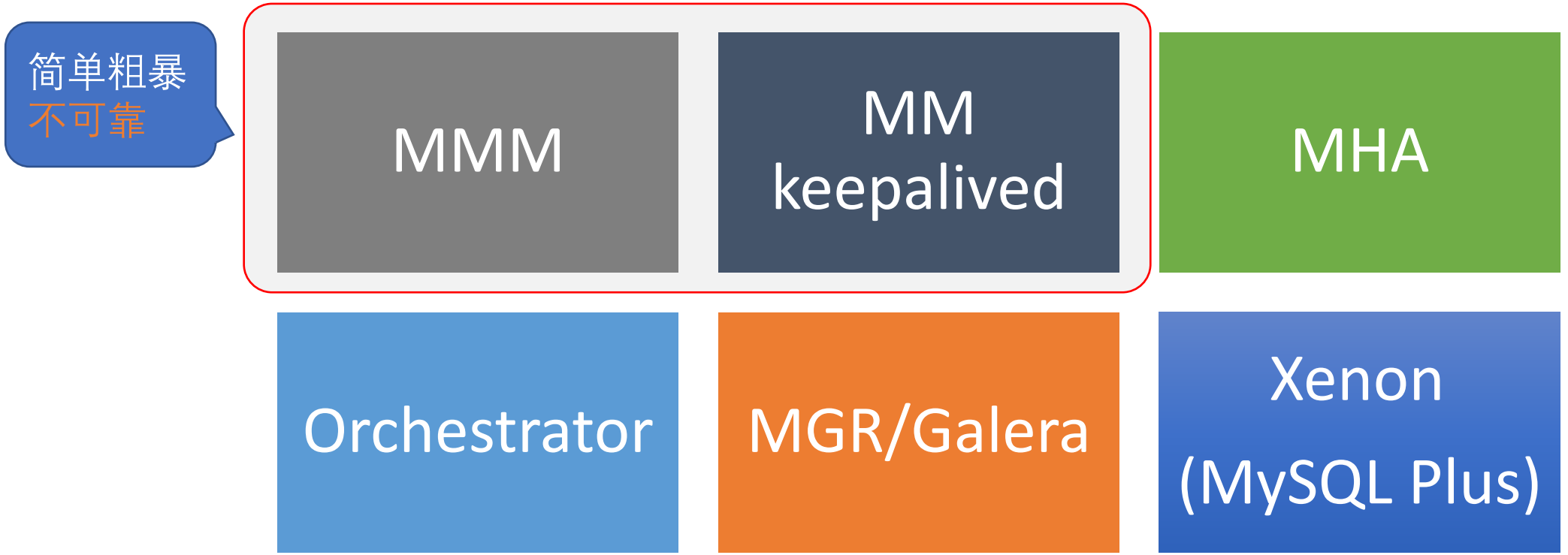
主要内容

- MySQL高可用方案选型
- DNS api & NamesHA
- MHA切换流程
- MHA自动化及监控
- MHA问题及总结
- MGR集群高可用

高可用方案选型

MMM	MM keepalived	MHA
Orchestrator	MGR/Galera	Xenon (MySQL Plus)

高可用方案选型



高可用方案选型

MMM

MM keepalived

MHA

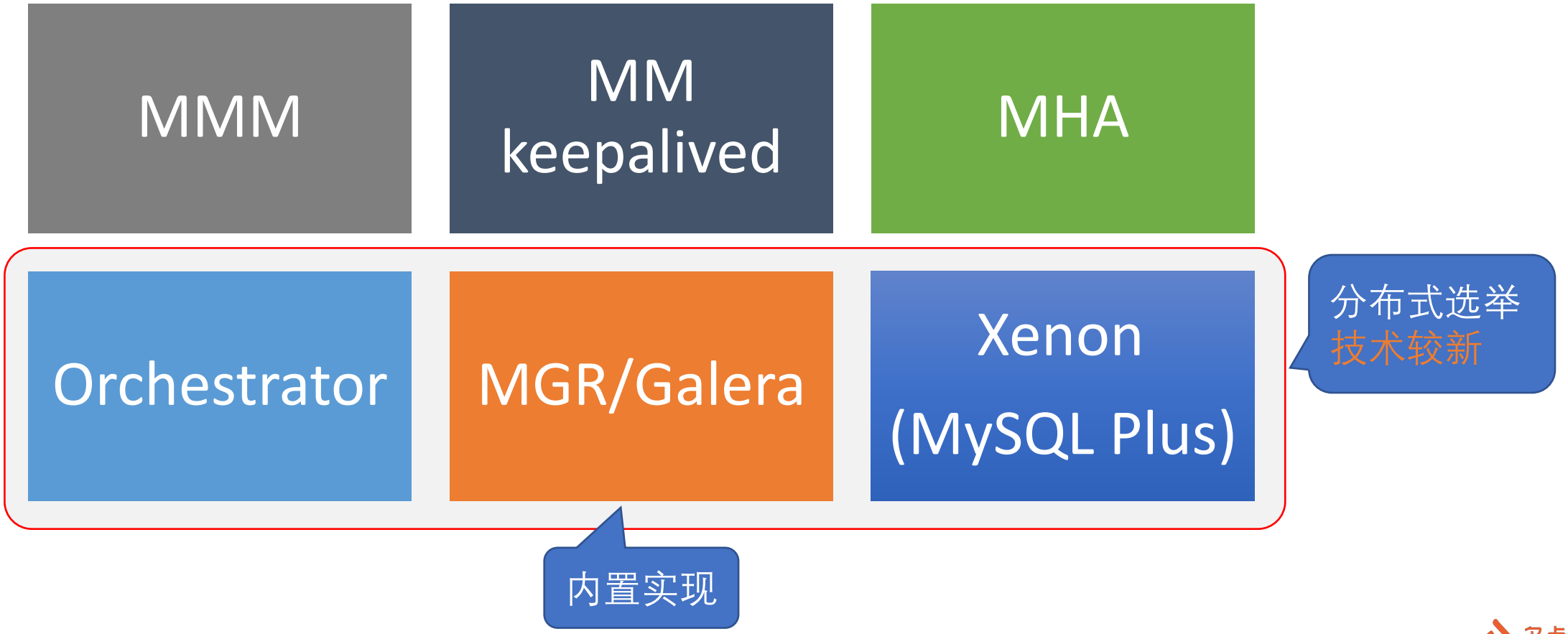
成熟流行
补全数据
停止维护
单点

Orchestrator

MGR/Galera

Xenon (MySQL Plus)

高可用方案选型


MMM
MM keepalived
MHA
Orchestrator
MGR/Galera
Xenon (MySQL Plus)
分布式选举
技术较新
内置实现

高可用方案选型

简单稳定

Classic & GTID

补数据

Binlog Server

配置丰富

MHA

5.6 经典复制
5.7 GTID复制

NamesHA

读写分离
无状态proxy

MGR

强一致性
读多写少
多节点写

Paxos

不丢数据

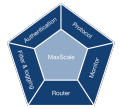
最终一致

Multi master

Oracle

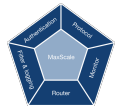
高可用 / 服务暴露

- 目标：failover对业务透明

VIP	DNS	配置中心+SDK	Proxy中间件
• 漂移迅速 • 增加维护成本，浪费ip地址 • 多实例部署难度大 • Master/standy需在相同网段，不适合混合云	• 通用性好 • 可读性好 • 可操作性好 • DNS TTL等待 • 需搭建内网DNS服务，并构建DNS api	• 安全性提高 • 需实现高度可靠的分布式配置中心 • 提供不同语言SDK • 成本高，周期长	• Load balancer或更多功能 • 响应时间增加 • Proxy维护成本 • Proxy本身高可用  ProxySQL 

高可用 / 服务暴露

- 目标：failover对业务透明

VIP	DNS	配置中心+SDK	Proxy中间件
• 漂移迅速 • 增加维护成本，浪费ip地址 • 多实例部署难度大 • Master/standy需在相同网段，不适合混合云	• 通用性好 • 可读性好 • 可操作性好 • DNS TTL等待 • 需搭建内网DNS服务，并构建DNS api	• 安全性提高 • 需实现高度可靠的分布式配置中心 • 提供不同语言SDK • 成本高，周期长	• Load balancer或更多功能 • 响应时间增加 • Proxy维护成本 • Proxy本身高可用  ProxySQL 

DNS设计

mysql.master.3306.oltp.shop-xyz-01.internal.dmall.com

1. DB类型
2. 实例角色
3. 端口号：递增且唯一，实例迁移到其他机器无端口冲突
4. 场景类型
5. 业务描述（可选）
6. 内网域名后缀

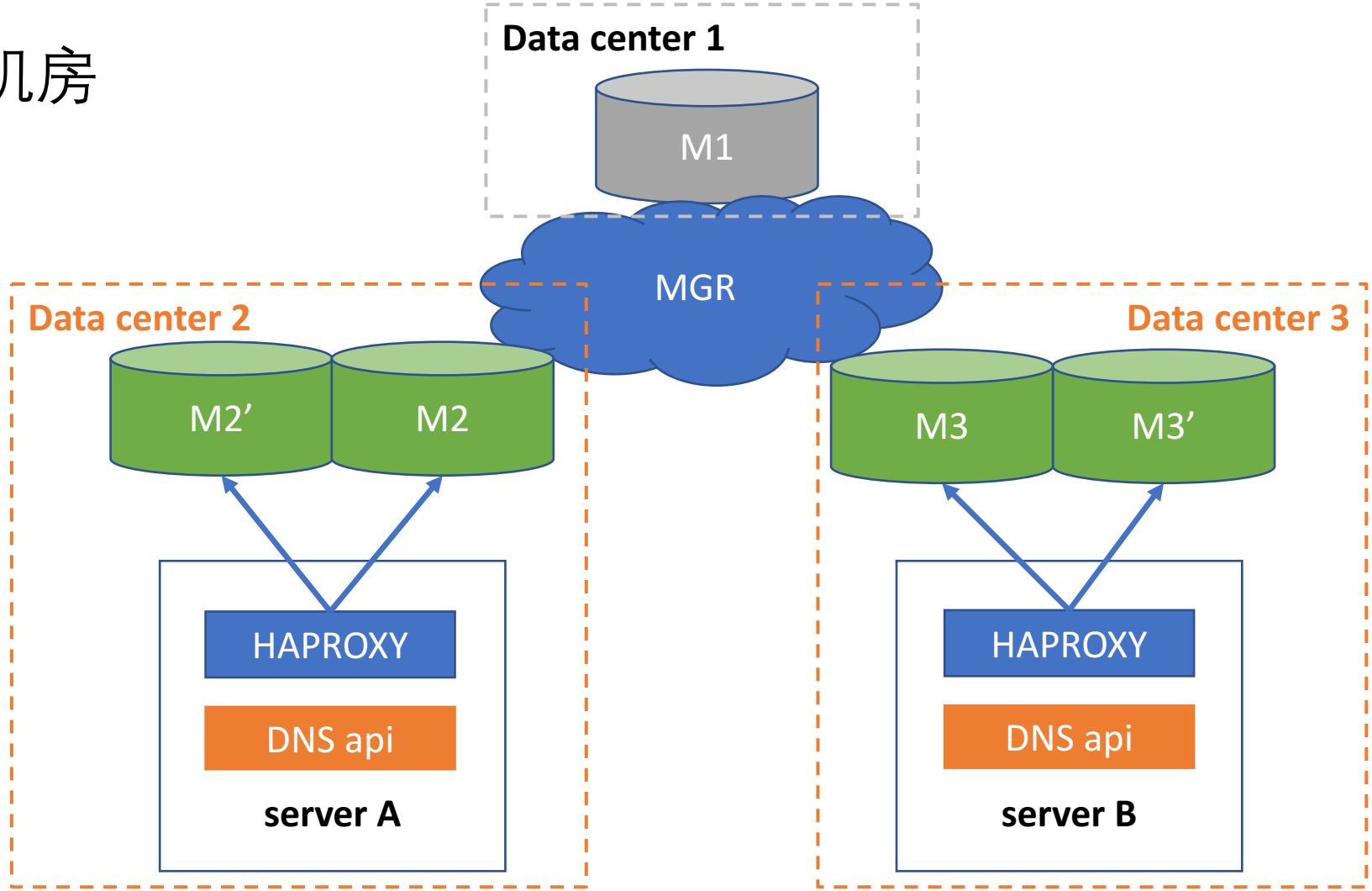
DNS api

- 基于bind实现http接口
- 使用MySQL存储
- 默认TTL 30s，小于MHA切换完成时间

API	HTTP方法	URI	参数
获取DNS解析	GET	/	pattern
添加DNS解析	POST	/add	token, name, type, addr, ttl
删除DNS解析	POST	/delete	token, name, type, addr
更新DNS解析	POST	/update	token, name, type, addr_list, ttl

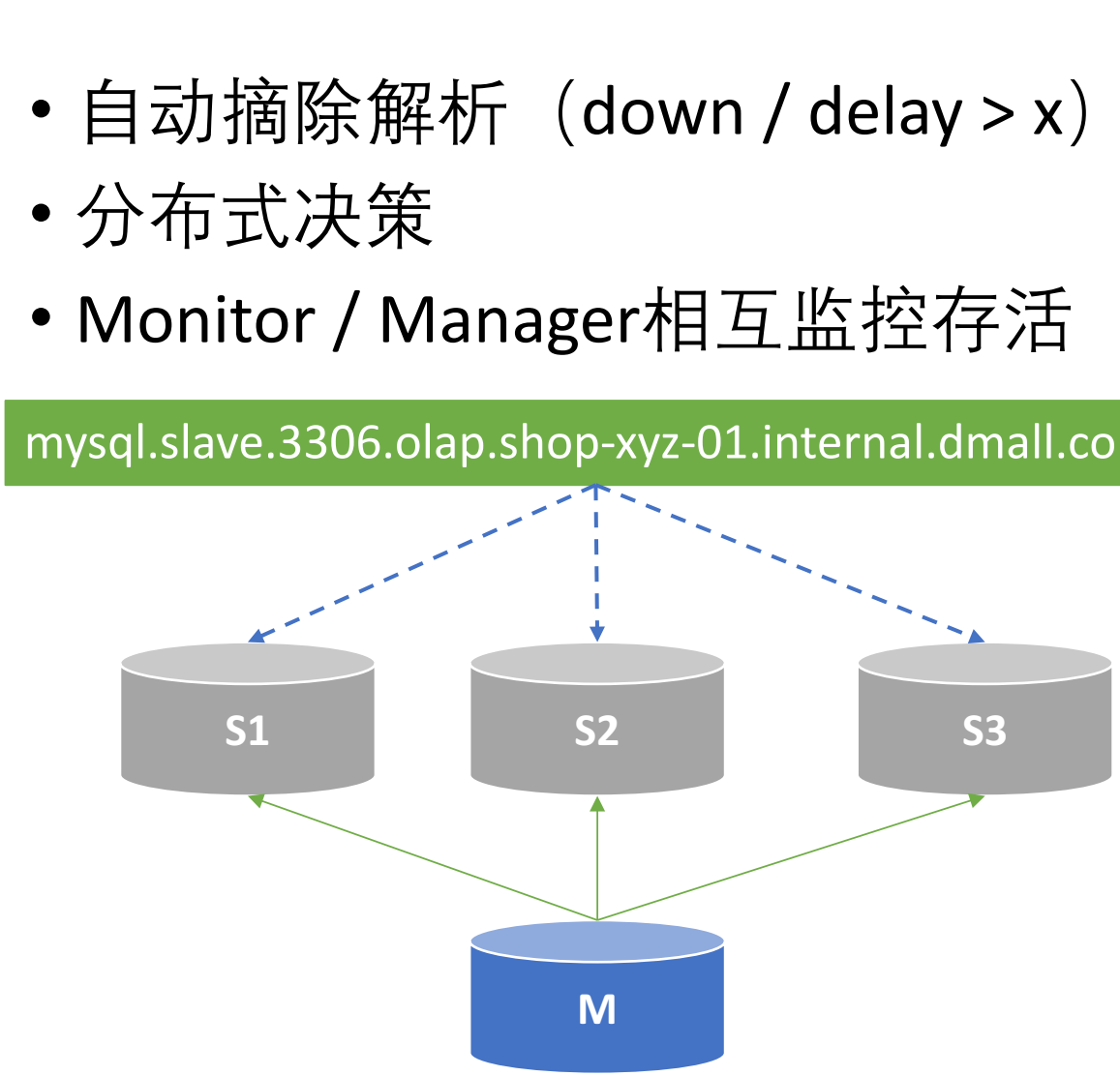
DNS高可用

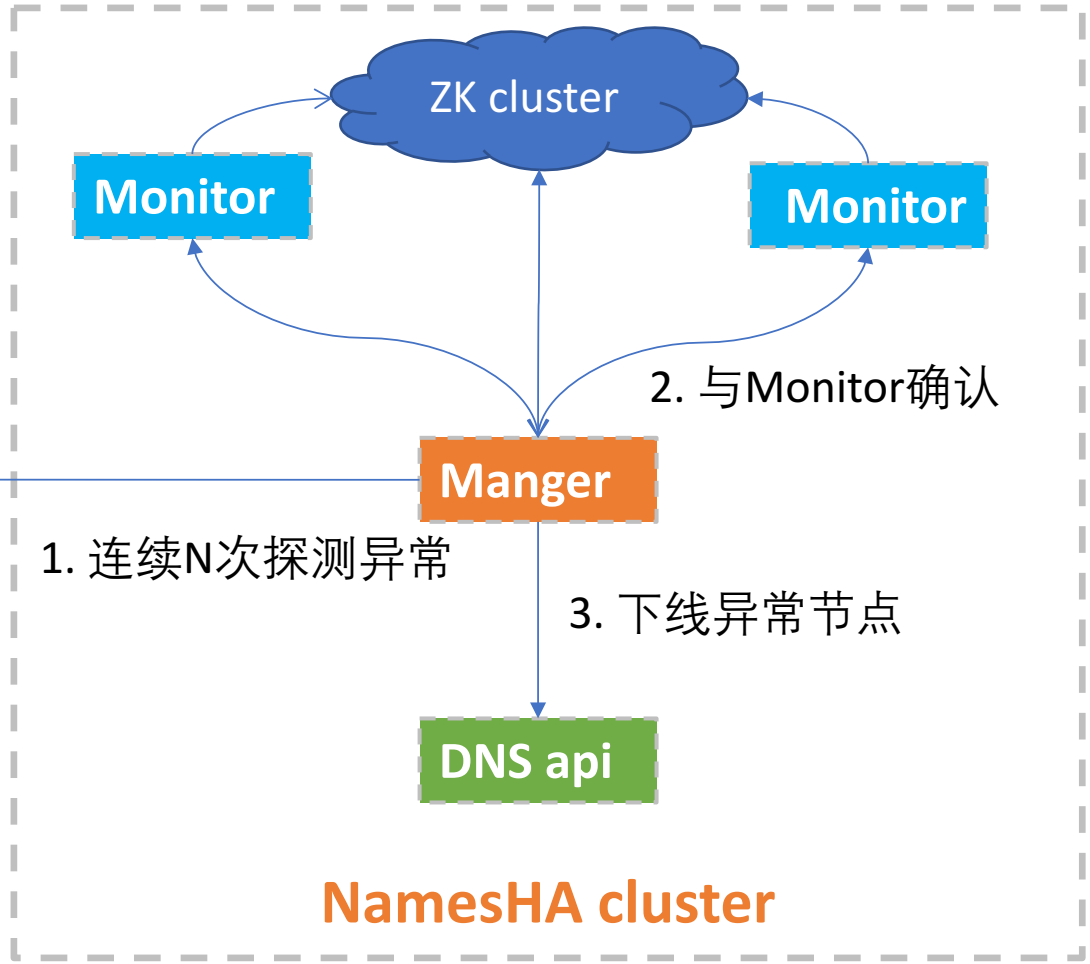
- DNS解析走本地机房
- 容许单机房故障
- MGR
  - ✓读多写少
  - ✓高一致性
  - ✓多活



智能DNS / NamesHA

- 自动摘除解析（down / delay > x）
- 分布式决策
- Monitor / Manager相互监控存活





MHA切换流程

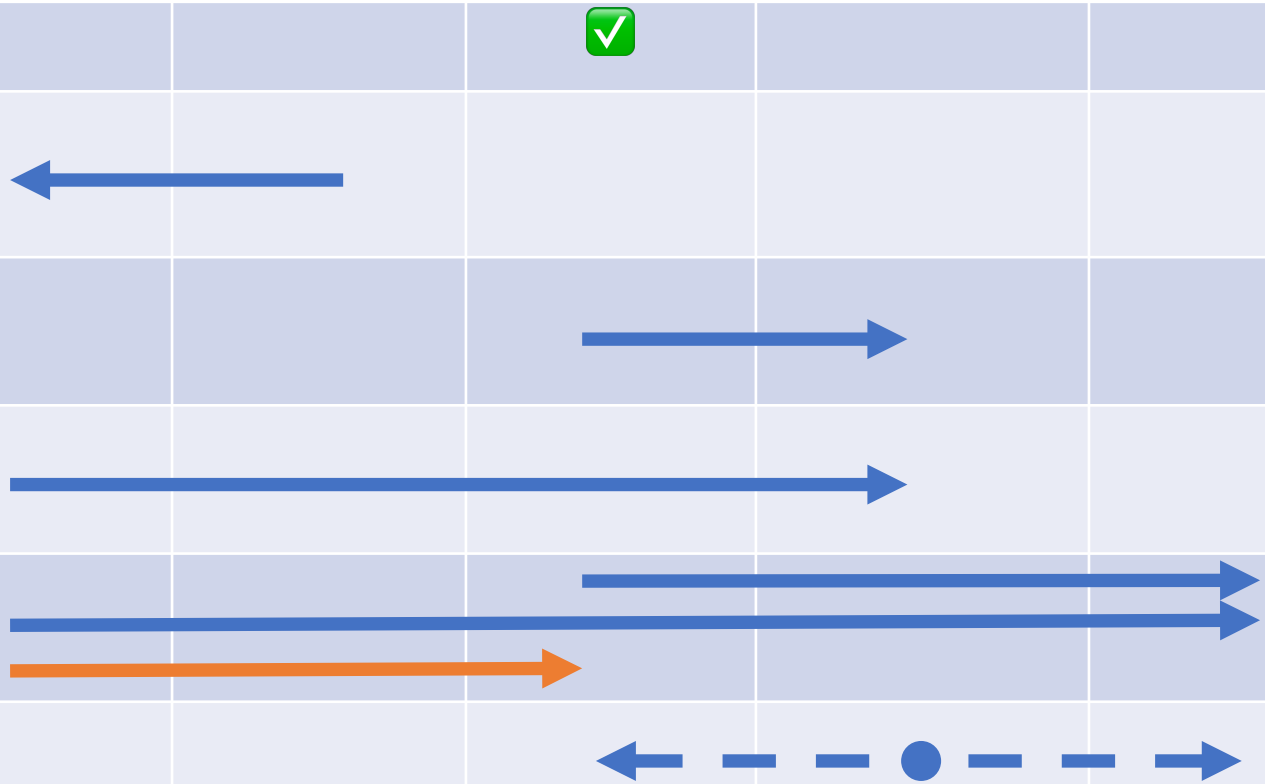
* Phase 1: Configuration Check Phase..
* Phase 2: Dead Master Shutdown Phase.. => (master_ip_failover stopssh)
* Phase 3: Master Recovery Phase..
  * Phase 3.1: Getting Latest Slaves Phase..
  * Phase 3.2: Saving Dead Master's Binlog Phase..
  * Phase 3.3: Determining New Master Phase..
  * Phase 3.3: New Master Diff Log Generation Phase..
  * Phase 3.4: Master Log Apply Phase.. => (master_ip_failover start)
* Phase 4: Slaves Recovery Phase..
  * Phase 4.1: Starting Parallel Slave Diff Log Generation Phase..
  * Phase 4.2: Starting Parallel Slave Log Apply Phase..
* Phase 5: New master cleanup phase..

调用DNS api
删除Dead master解析

调用DNS api
添加New master解析

MHA切换流程 / 补数据 (file:position)

#	步奏	Manager	Dead master	Latest slave	New master	Other slaves
0	找出latest slave			✅		
1	尝试从dead master补latest slave上缺失的binlog -> d1					
2	从latest slave上补new master上缺失的binlog -> d2					
3	从manager上拉取d1 应用d2，d1					
4	并行从latest slave上补缺失的binlog，d1，并应用					
5	Change master… Start slave					



MHA切换流程 / 补数据原理

Slave IO线程读取到master位点	Slave SQL线程执行对应master位点	Slave relay log位点
Master_Log_File: mysql-bin.000220 Read_Master_Log_Pos: 33754	Relay_Master_Log_File: mysql-bin.000220 Exec_Master_Log_Pos: **33754**	Relay_Log_File: **relay-bin.000074** Relay_Log_Pos: 725

- Relay log中end_log_pos记录了对应master位点

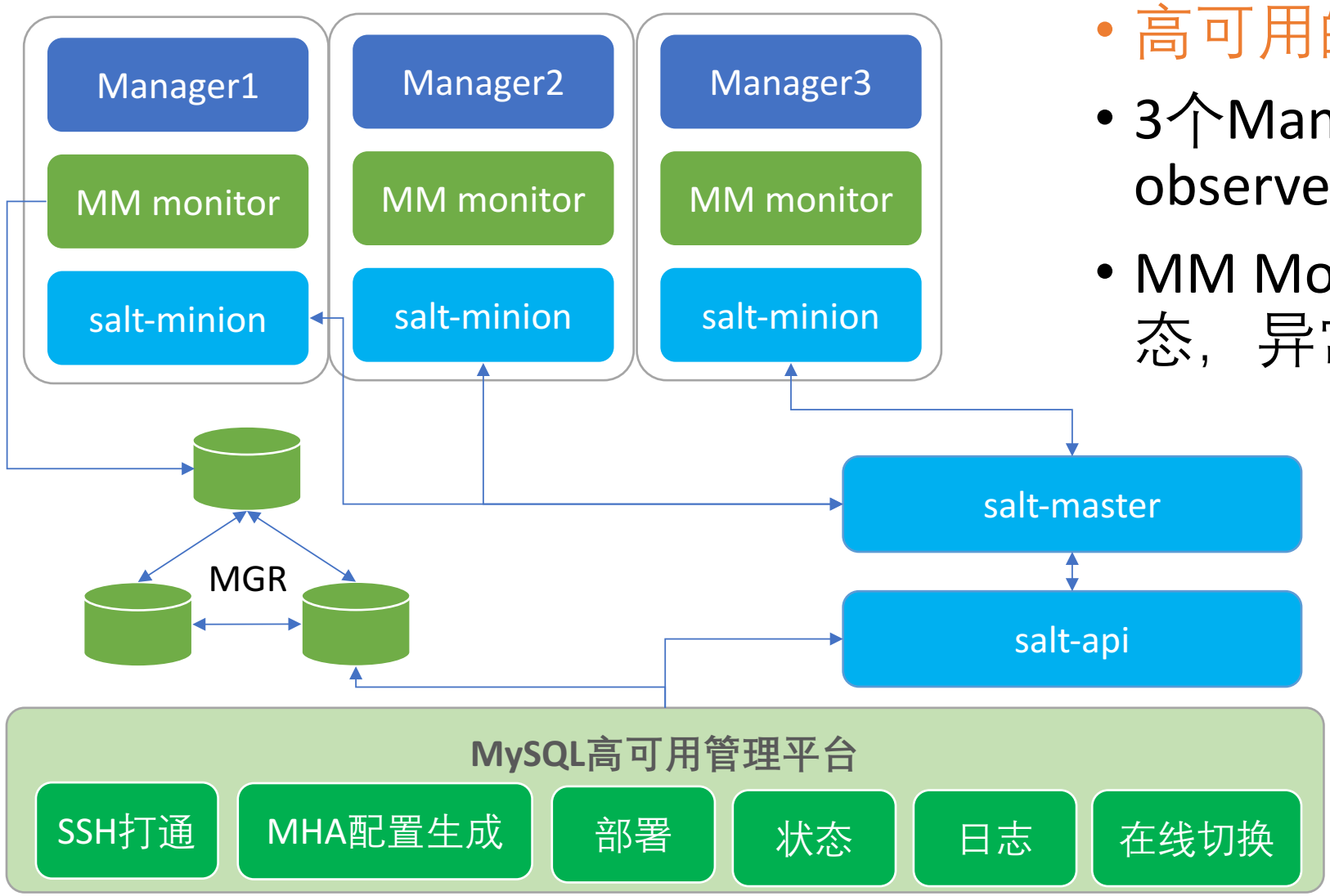
```
root@slave # mysqlbinlog -vvvv relay-bin.000074


…
# at 694
#190412 15:15:36 server id 1653666  end_log_pos 33754 CRC32 0x9bc2e127
        Xid = 1137590
COMMIT/*!*/;
```

MHA切换流程 / GTID

- 不会尝试从Dead master补数据
- 若配置有Binlog server，则尝试从Binlog server补数据
- 利用GTID配置其他slave

MHA自动化及监控



- 高可用的监控异常重要
- 3个Manager互为彼此的observer
- MM Monitor定期检查运行状态，异常发短信
  - ✓ Manager进程异常
  - ✓ 集群节点变更
  - ✓ 与DB中状态不一致

MHA问题及总结

- binlog_sync设置为1
  - 否则可能：master binlog没落地，但同步到slave，老master起来后缺数据
- GTID模式下，补数据需要配置binlog server
- 重要参数
  - check_repl_delay：设置为1时，延迟超过100MB，拒绝切换
  - ping_interval：检查间隔，默认3s，最大10s，设置超过10s后报异常
  - client_bindir / client_libdir：支持MariaDB非GTID模式
- 业务需设置正确的DNS TTL缓存过期策略

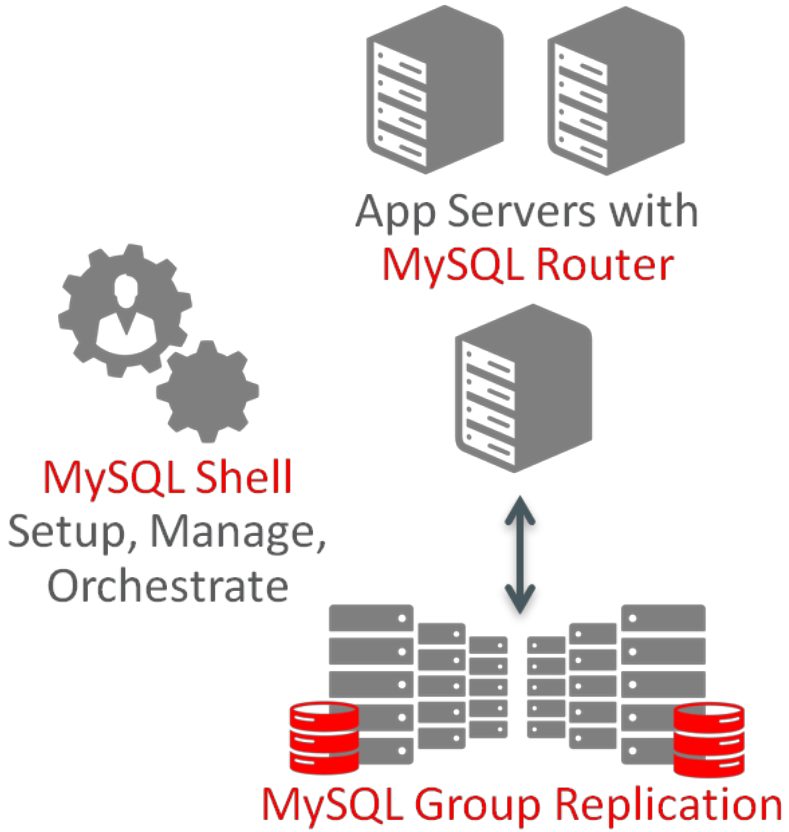
```
sun.net.inetaddr.ttl=30
networkaddress.cache.ttl=30
```

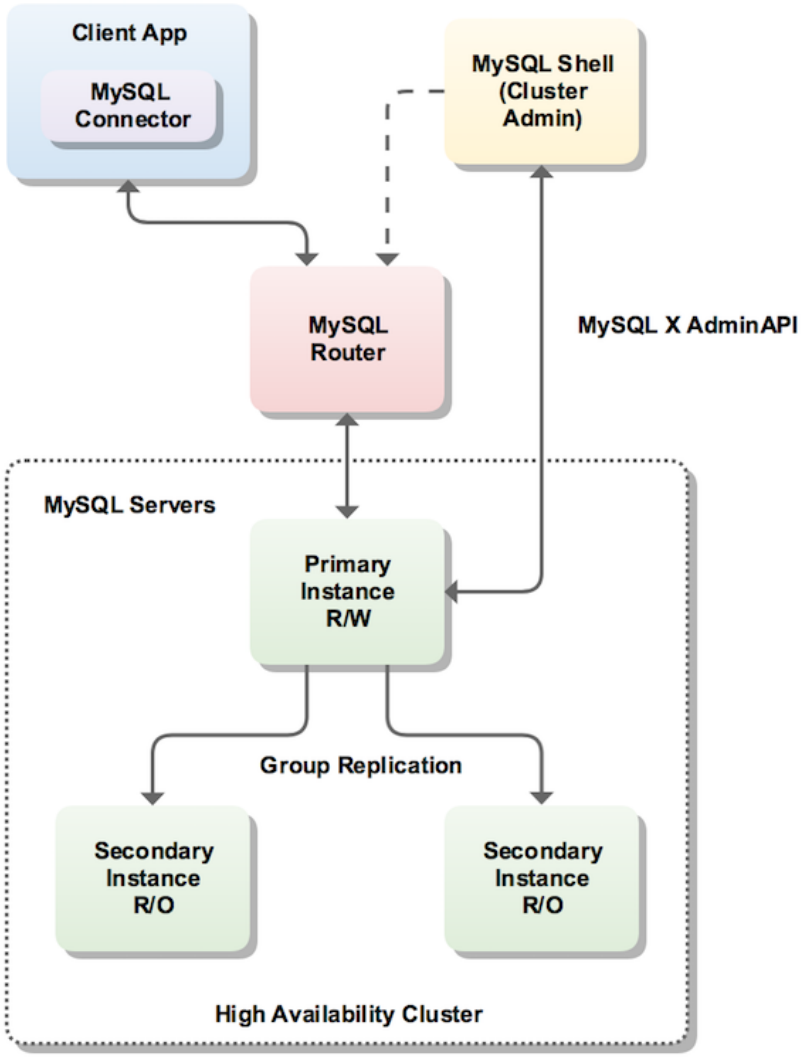
MGR集群高可用

多节点写 | Paxos协议 | 最终一致 | Failover不丢数据

- 使用场景
  - 读多写少，数据一致性要求高，多活
- 集群内部基于Paxos投票选举，但缺接入层高可用
  - 期待类似MongoDB driver

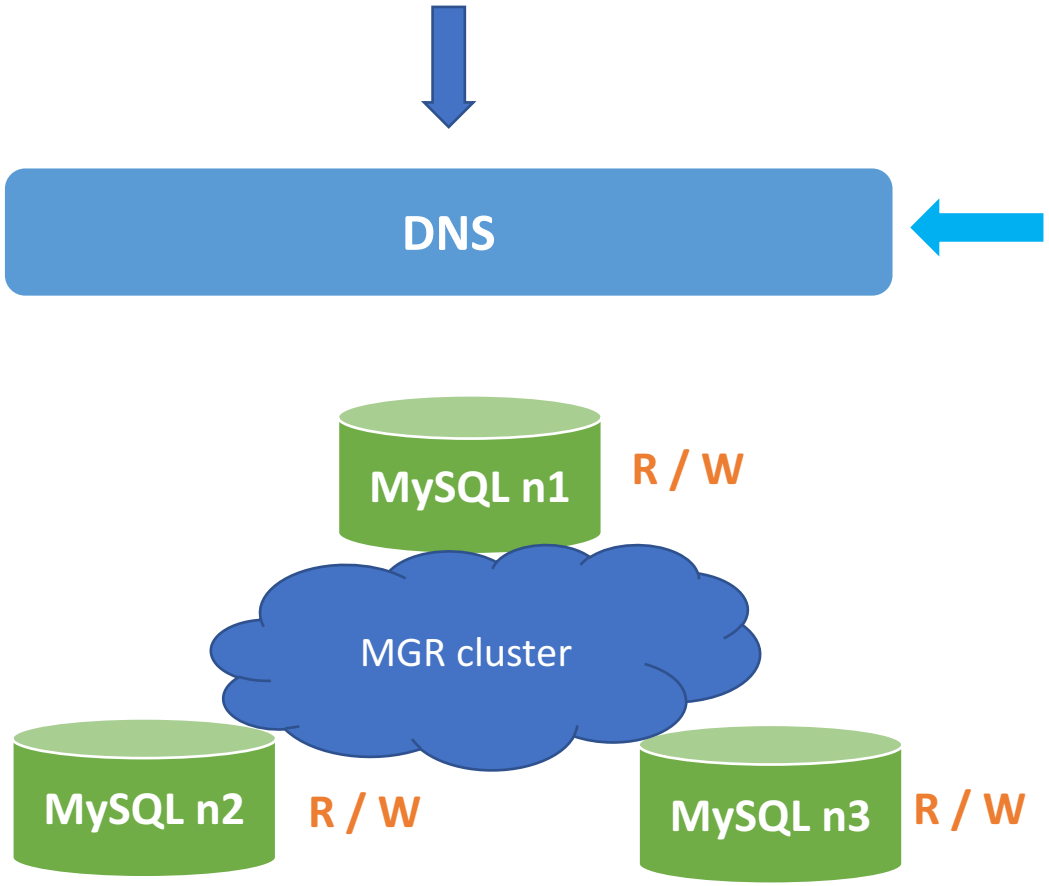
MGR集群高可用





- Oracle官方架构
- 不适用于已配置好的MGR集群
- 架构复杂
- 稳定性待验证

MGR集群高可用



Multi-write

- group_replication_exit_state_action
  - ABORT_SERVER / READ_ONLY
  - 8.0.12+, 5.7.24+

Q & A

Reference

- https://github.com/yoshinorim/mha4mysql-manager/wiki
- https://www.slideshare.net/matsunobu/automated-master-failover
- https://dev.mysql.com/doc/refman/5.7/en/group-replication.html

Thanks!