DTCC

第十四届中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA

数智赋能 共筑未来

IT168.com ChinaUnix ITPUB

北京国际会议中心 | 2023/8/16-18

PostgreSQL的高可用架构设计与实践

海能达通信股份有限公司+数据库团队负责人+崔鹏

高可用的本质

为什么要高可用?

增加系统风险控制能力!

抵御天灾人祸之能力!

万物皆非100%可靠-内部因素

从运维、开发人员层面：都可能误操作。

从信息系统层面：软件都是有可能有BUG的。

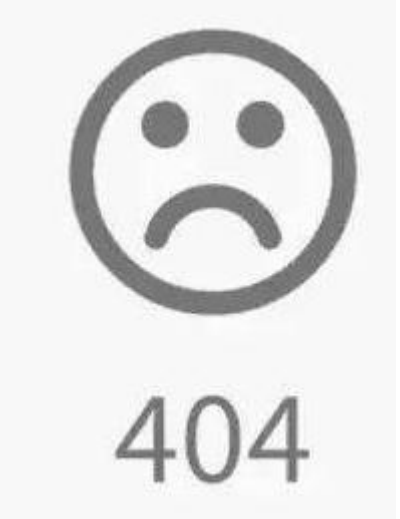
从硬件层面：硬件都是有可能会坏的。

万物皆非100%可靠-外部因素

从企业角度： 无高可用，企业线上系统服务被中断,使企业蒙受损失,经济和商誉两方面受损。

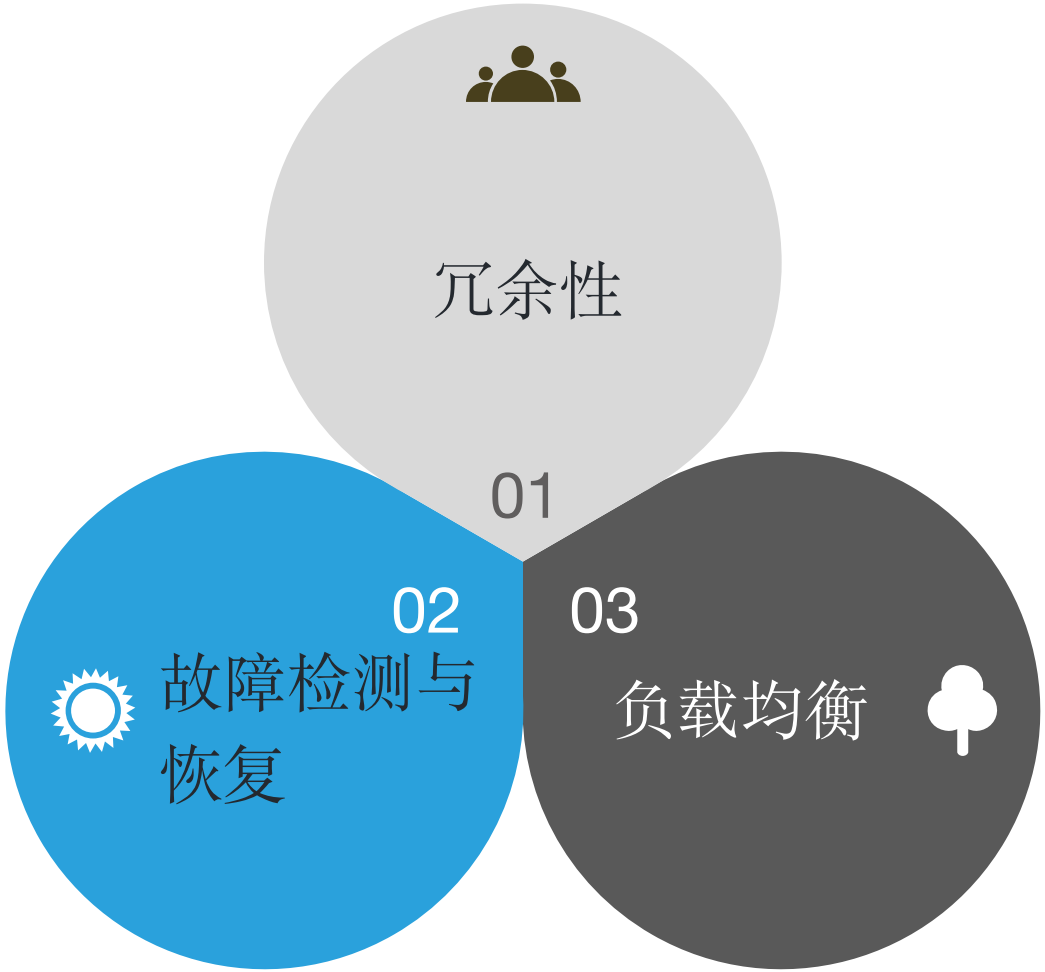
从用户层面： 无高可用，客户会的投资以及利益蒙受损失。

从社会维度： 无高可用，社会秩序可能受影响。



万物皆非100%可靠-核心

确保系统在面对各种故障和
中断情况时仍能提供持续可靠的服务。



如何思考高可用-引入冗余

通过使用冗余设备或系统组件来提高可用性。

例如，使用冗余服务器、网络设备和存储设备等。

当一个设备或组件故障时，可以切换到备用设备或组件上，以保持系统的连续运行。

如何思考高可用-负载均衡

通过将负载分布到多个服务器或节点上，确保系统能够处理更多的请求。

负载均衡可以通过硬件设备（如负载均衡器）或软件（如反向代理）来实现。

这样可以避免单点故障，并提高系统的整体性能和可用性。

如何思考高可用-容错设计

在系统架构和应用程序设计中，采用容错机制可以预防和处理故障。

例如，使用错误检测和纠正技术，实现数据备份和恢复机制，

以及实施事务处理和回滚机制等。

如何思考高可用-自动化运维

自动化是提高可用性的关键因素之一。通过自动化运维工具和脚本，

可以减少人为错误和操作失误的可能性，并快速响应故障。

自动化还可以加快故障诊断和恢复的速度，降低系统停机时间。

如何思考高可用-监控和警报

建立全面的监控系统来实时监测关键组件和指标，

以便及时发现和解决潜在问题。合理设置警报机制，

可以及时通知相关人员并采取措施，从而减少系统故障对业务的影响。

如何思考高可用-容量规划

对系统进行合理的容量规划，确保系统能够处理当前和未来的负载。

避免过度使用资源或资源瓶颈，以防止系统崩溃或性能下降。

如何思考高可用-预防措施

除了上述措施外，还应采取一些预防措施来提高可用性。

例如，定期进行备份和灾难恢复测试，定期进行软件和硬件的升级和维护，

以及培训和准备应急响应团队等。

PostgreSQL主流高可用技术对比解析

高可用的核心需求

数据复制需求

1.同步复制
2.异步复制
3.级联复制
4.不同集群间的数据同步

故障转移

1.支持自动故障转移
2.支持脚本回调

部署需求

1.支持k8s, docker等容器化环境部署
2.满足不同级别项目的快速部署

其它需求

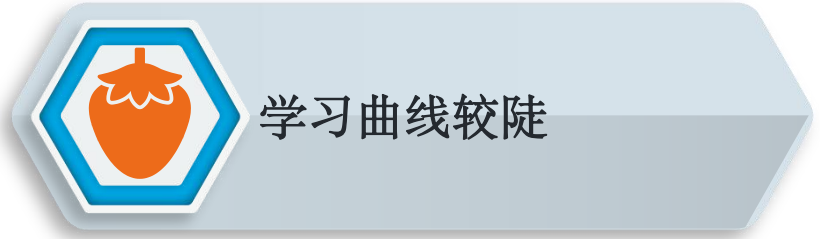
1.支持通过pg_rewind自动修复旧主
2.支持通过watchdog防止脑裂
3.支持多种方式初始化集群和重建备机

Patroni优点

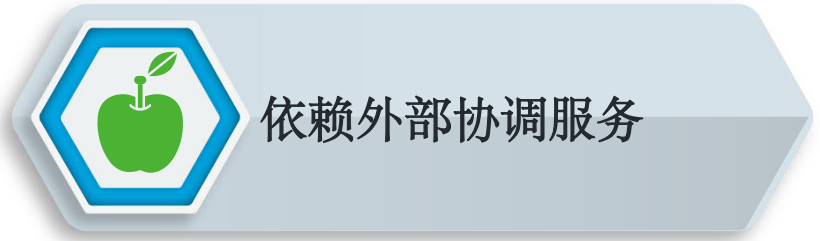
Patroni提供了简单的配置和管理界面，使得部署和维护PostgreSQL高可用性集群变得容易。它采用YAML配置文件，可以快速定义和修改集群配置，同时提供了命令行工具和RESTful API，方便管理和监控集群状态。

Patroni缺点

对于没有使用过类似工具的用户来说，Patroni可能需要一些时间来学习和理解其配置和管理方式。尽管Patroni提供了详细的文档和示例，但有一定的技术门槛，需要一定的经验和知识。



Patroni使用外部协调服务（如ZooKeeper、etcd或Consul）来实现主节点选举和故障检测等功能。这意味着要部署和配置额外的组件，并确保它们的可用性和稳定性。如果协调服务发生故障或出现网络问题，可能会影响到整个集群的正常运行。



Patroni专注于管理PostgreSQL数据库集群，因此对于其他数据库引擎或系统的支持相对有限。如果需要管理多种不同类型的数据库集群，可能需要考虑其他适合的工具或解决方案。

Repmgr优点

简要说明

repmgr提供了简洁而直观的命令行工具和API，使得设置和管理PostgreSQL的主从复制变得容易。它使用INI格式的配置文件，允许用户快速定义和修改复制拓扑结构，并提供了多种选项来适应不同的需求和环境。

Repmgr缺点

简要说明

repmgr的配置相对复杂，特别是对于新手用户来说可能会有一定的学习曲线。它需要进行详细的配置和设置，包括复制节点的身份、连接信息、拓扑结构等。在某些复杂的情况下，可能需要更深入的了解和调试，以确保正确的配置和运行。

01 配置复杂性

02 对网络和存储依赖

03 数据同步延迟

04 限于PostgreSQL

主流开源PostgreSQL高可用方案

概览	stolon	pgpool	repmgr	patroni
开源协议	Apache 2.0	BSD	GPL	MIT
支持PG版本	9.4 to 12	9.1 to 13	9.5 to 13	9.3 to 13
开发语言	Go	C	C	Python
测试情况	使用案例、资料较少	性能损耗较大。 容易出现脑裂。	扩展使用灵活性较差	部署简单 灵活性较高 源码可读性较好

PostgreSQL高可用选型-DCS

	Zookeeper	etcd	Consul
产生时间	长	短	短
原生语言	JAVA	Go	Go
算法	Paxos	Raft	Raft
多数据中心	不支持	不支持	支持
健康检查	支持	不支持	支持
web管理界面	支持	不支持	支持
http协议	较为复杂	支持	支持
DNS协议	较为复杂	不支持	支持

三种DCS软件各有优缺点,由于整体业务基础架构使用K8S+etcd做微服务容器管理,故选择etcd。

应用如何连接数据库集群?

Pgbouncer/Pgpool/Haproxy/VIP/DNS/JDBC或其它语言支持连接层配置多IP地址。

JDBC 配置多IP地址

- jdbc:postgresql://node1,node2,node3/accounting?targetServerType=master

订阅etcd中的Leader Key变化

- **Leader Key变化**

VIP

- 同中心提供虚拟机访问方式,patroni的callback脚本

PostgreSQL主流连接池、备份工具对比解析

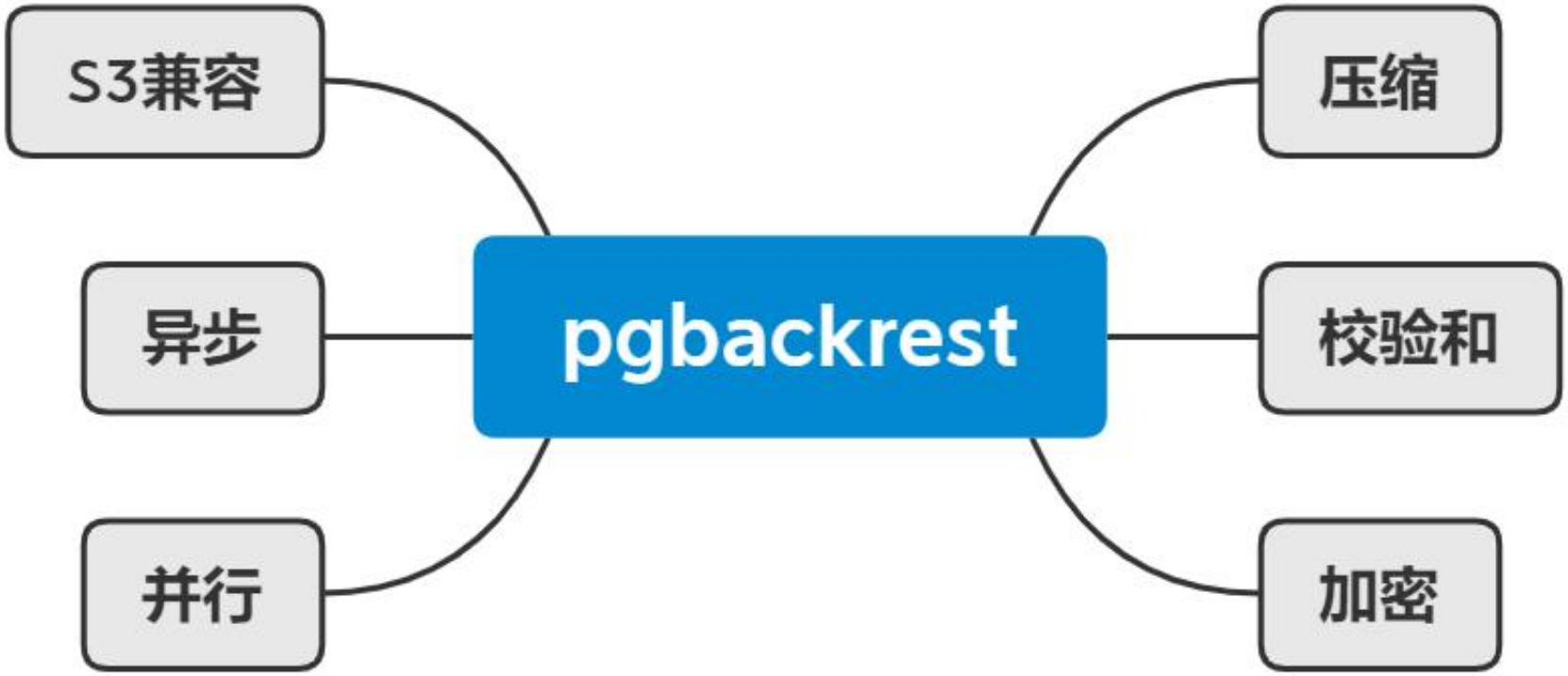
Pgbackrest

特点：Pgbackrest是一个强大且功能丰富的备份和恢复工具。它使用基于硬链接的增量备份策略，提供了高性能和低备份窗口的特点。Pgbackrest支持并行备份和恢复操作，并提供了各种配置选项来满足不同需求。

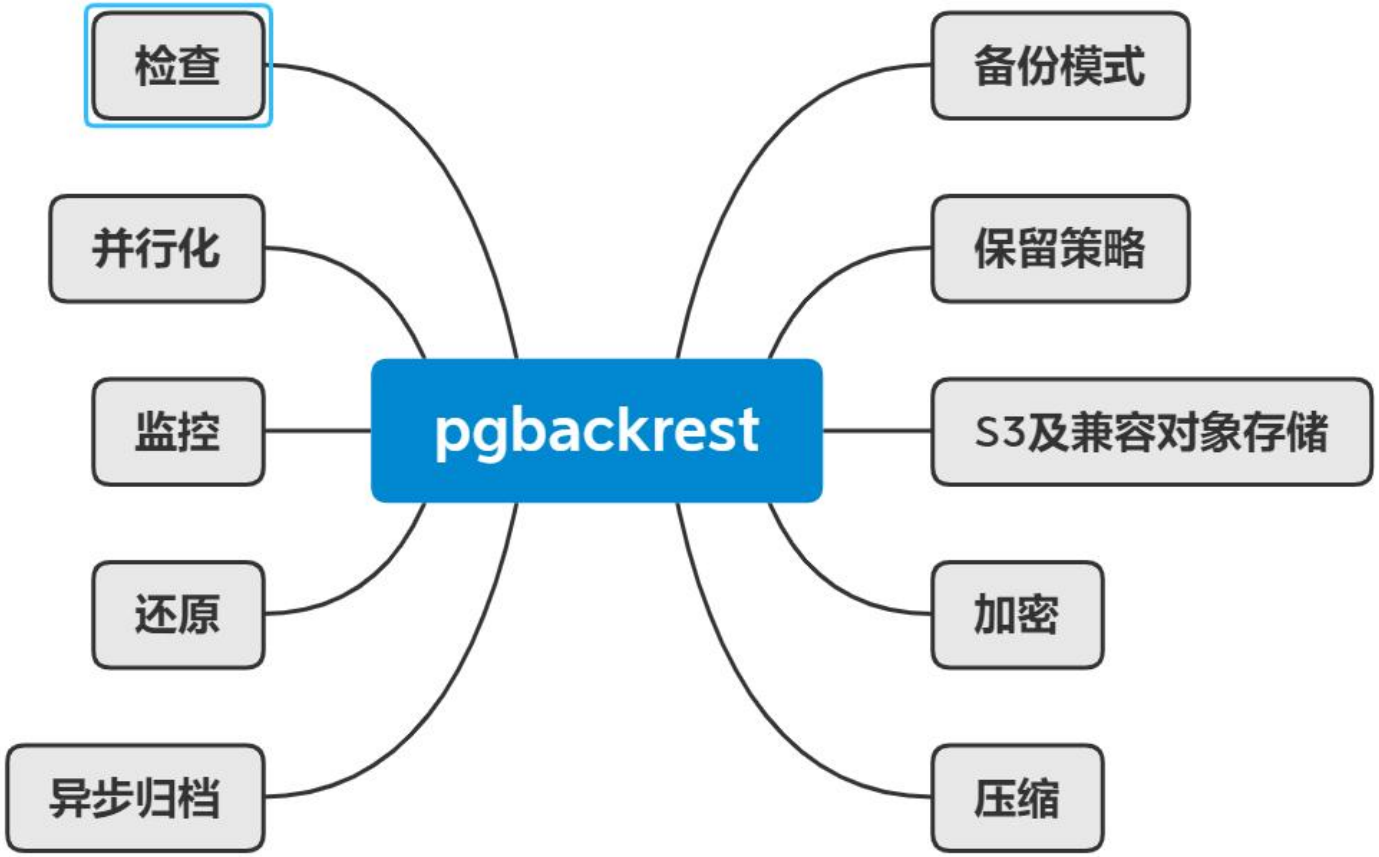
优点：Pgbackrest具有出色的性能和并行处理能力，并提供了高级功能，如增量备份、备份完整性校验、远程复制等。它的文档和社区支持也相对较好。

缺点：Pgbackrest的配置比较复杂，对初学者而言可能需要花费一些时间来进行学习和理解。此外，Pgbackrest的备份文件格式并非与PostgreSQL兼容，因此在恢复时可能需要使用特定的工具进行操作。

Pgbackrest技术特性



Pgbackrest功能特性



pg_probackup

特点：pg_probackup是由Postgres Professional开发的备份和恢复工具。它支持全量备份和增量备份，通过使用PostgreSQL内部的技术（如并发控制和热备份）来实现高性能备份。

优点：pg_probackup具有与PostgreSQL高度集成的特点，可以直接从数据库实例进行备份和恢复。它支持并行备份和恢复操作，并提供了管理工具和命令行界面来简化操作。

缺点：pg_probackup在某些方面相对较新，可能在稳定性和成熟度方面略逊一筹。此外，它的文档和社区支持相对较少，可能需要更多自主学习和摸索。

pg_rman

特点：pg_rman是一个基于归档日志的备份和恢复工具。它使用PostgreSQL的归档日志来实现备份和增量备份功能，支持全量和差异备份，并可与物理备份（如文件级备份）结合使用。

优点：pg_rman相对简单易用，它的备份文件与PostgreSQL兼容，恢复时可以直接使用PostgreSQL工具进行操作。它提供了可靠的备份和恢复机制，并支持WAL文件的轮转和管理。

缺点：pg_rman在并行备份和恢复方面的能力比较有限，可能无法满足高吞吐量和大规模数据的需求。此外，pg_rman的社区支持相对较少，更新和改进的速度可能较慢。

pg_basebackup

特点：
1.全量备份：pg_basebackup执行全量备份，将数据库的所有数据文件复制到目标位置。
2.基于流复制协议：它使用PostgreSQL的流复制协议（Streaming Replication Protocol）进行备份，可以实现高效的数据传输。
3.支持增量备份：结合归档日志，pg_basebackup可以实现增量备份的功能。
4.简单易用：pg_basebackup是PostgreSQL自带的工具，使用起来相对简单，不需要额外安装和配置。

优点：
1.数据一致性：pg_basebackup在备份过程中会自动处理并发操作，保证备份数据的一致性。
2.故障恢复简单：由于备份是全量的，因此在故障恢复时只需要拷贝备份文件到新的数据库实例即可，恢复速度较快。
3.集成性强：pg_basebackup与PostgreSQL紧密集成，可以与其他PostgreSQL工具和功能无缝配合使用。

缺点：
1.备份窗口较长：由于pg_basebackup是全量备份，当数据量较大时，备份过程可能相对耗时，导致备份窗口较长。
2.不支持并行操作：pg_basebackup不支持并行备份操作，无法充分利用多个CPU核心或网络带宽来提高备份速度。

连接池工具概述

1.PgBouncer：PgBouncer是一个轻量级的连接池工具，它可以在应用程序和数据库之间充当中间件。PgBouncer支持连接池的复用和管理，可以提高连接的效率和性能，并节省数据库资源。它支持事务，并提供了各种配置选项和灵活性。

2.pgpool-II：pgpool-II是一个功能强大的连接池和负载均衡工具。除了连接池功能外，pgpool-II还提供了查询缓存、主备复制、故障切换等高级功能。它可以实现对多个数据库服务器的负载均衡和故障恢复，提供了高可用性和扩展性。

3.Apache Tomcat JDBC Pool：Apache Tomcat JDBC Pool是适用于Java应用程序的连接池实现。虽然它最初是为Tomcat服务器设计的，但也可以作为独立的连接池工具使用。它支持连接池的管理和复用，提供了一些高级功能，如连接泄漏检测、空闲连接回收等。

4.HAProxy是一个开源的高性能、可靠的负载均衡器和代理服务器。它被广泛应用于分布式系统架构中，可以将网络流量分发到多个后端服务器，实现负载均衡和高可用性。

Pgbouncer

特点：
1.连接池：Pgbouncer可以维护一个连接池，将数据库连接复用，并通过复用连接减少与数据库的建立和断开连接的开销。

2.代理服务器：Pgbouncer作为一个中间层代理，可以将客户端请求转发给后端的PostgreSQL服务器。这样可以分担数据库服务器的负载，并提高系统的并发性能。

3.高性能：Pgbouncer采用异步I/O模型，具有较低的资源消耗和响应延迟，可以处理大量的并发连接。

4.配置灵活：Pgbouncer的配置非常灵活，可以根据实际需求进行调整和优化，包括连接池大小、超时设置、认证方式等。

5.高可用性：Pgbouncer支持故障检测和自动恢复功能。当后端的数据库服务器发生故障时，Pgbouncer可以自动将连接切换到其他可用的服务器上，保持系统的连续可用性。

Pgbouncer

优点：
1.节省资源：通过连接池的使用，Pgbouncer减少了数据库建立和断开连接的开销，节省了资源和网络带宽。
2.提高性能：Pgbouncer通过复用连接和异步I/O模型，提高了系统的并发处理能力和响应速度。
3.简化连接管理：Pgbouncer管理连接池的创建、销毁和超时等，简化了数据库连接的管理和维护工作。
4.可靠性：Pgbouncer具有故障检测和恢复功能，可以自动切换到可用的数据库服务器，提供高可用性和容错能力。

缺点：
1.单点故障：由于Pgbouncer本身是一个中间代理，如果Pgbouncer出现故障，可能会导致整个系统无法访问数据库。
2.有限的功能：相对于完整的数据库服务器，Pgbouncer的功能较为有限，它主要关注连接管理和负载均衡，并不支持一些数据库特定的功能和扩展。

Pgpool

特点：

1.连接池：Pgpool可以维护一个连接池，复用数据库连接，减少建立和断开连接的开销，并提高性能。

2.负载均衡：Pgpool作为负载均衡器，可以将客户端请求分发给后端的多个PostgreSQL数据库节点，平衡负载，并增加系统的并发处理能力。

3.自动故障检测和切换：Pgpool可以监测后端数据库节点的可用性，并在某一节点发生故障时自动切换到其他可用节点，保证系统的高可用性。

4.并行查询：Pgpool可以将一个查询请求分解为多个子查询，并在后端数据库节点之间并行执行，提高查询性能和响应时间。

Pgpool

优点：
1.提高性能：通过连接池和负载均衡机制，Pgpool可以提高系统的并发处理能力和响应速度，减少数据库的负载压力。

2.高可用性：Pgpool支持故障检测和自动切换功能，当后端数据库节点发生故障时，Pgpool可以自动将连接切换到其他可用节点，保证系统的连续可用性。

3.负载均衡：Pgpool可以将客户端请求均匀地分发到多个数据库节点，平衡负载，提高系统的扩展性和稳定性。

4.并行查询：Pgpool的并行查询功能可以将查询请求分解为多个子查询，并在多个节点上并行执行，加快查询速度。

Pgpool

缺点：

1.配置复杂：Pgpool的配置相对复杂，需要了解和理解PostgreSQL集群的工作原理和参数设置，以确保正确和最佳的配置。

2.单点故障：如果Pgpool本身出现故障或性能瓶颈，可能会影响整个数据库集群的访问。

3.不支持所有PostgreSQL特性：Pgpool并不完全支持所有PostgreSQL的特性和扩展，某些复杂的操作可能需要直接与数据库节点交互。

4.切换时数据一致性：由于切换时的数据同步延迟，可能会导致在发生故障切换时部分数据的丢失或不一致。

Haproxy

特点：

1.高性能：HAProxy采用事件驱动的模型和多线程技术，具有卓越的性能和响应速度，能够处理大量并发连接和高流量负载。

2.支持多种负载均衡算法：HAProxy支持多种负载均衡算法，如轮询、最小连接数、IP散列等，可以根据需求进行灵活配置。

3.健康检查：HAProxy能够定期检查后端服务器的健康状态，通过监测服务器的可用性来确保只将请求转发到正常运行的服务器上。

4.SSL/TLS终端处理：HAProxy可以作为SSL/TLS终端进行加密和解密，提供安全的通信通道，并支持负载均衡和代理功能。

Haproxy

优点：

1.高可用性：HAProxy支持热备份和故障自动切换，当一个或多个后端服务器发生故障时，HAProxy能够自动将请求转发到其他健康的服务器上，确保系统的连续可用性。

2.灵活的配置：HAProxy的配置文件简单而灵活，可以通过配置文件进行动态调整和扩展，以适应不同的负载均衡需求。

3.监控和统计：HAProxy提供丰富的监控和统计信息，可以实时监测负载均衡器和后端服务器的性能指标，并进行故障排除和性能优化。

4.支持WebSocket和HTTP/2：HAProxy支持WebSocket协议和HTTP/2协议，能够处理现代Web应用的通信需求。

Haproxy

缺点：

1.单点故障：由于HAProxy本身是一个集中式的负载均衡器，如果HAProxy发生故障，可能会导致整个系统无法访问。

2.学习成本：HAProxy的配置相对复杂，需要了解和理解各项配置参数和负载均衡算法，初学者可能需要一些时间来学习和掌握。

3.有限的高级功能：HAProxy的重点是负载均衡和代理，相对于完整的Web服务器，它可能缺少某些高级的Web服务功能。

追寻极致的SLA、RTO、RPO

概念简介

SLA（Service Level Agreement）、RTO（Recovery Time Objective）和RPO（Recovery Point Objective）是与业务连续性和灾难恢复相关的重要概念。

SLA：服务级别协议，定义了服务提供商与客户之间的约定，包括服务可用性、性能指标和支持水平等。SLA通常规定了系统的最低运行时间和可接受的故障恢复时间。较高的SLA意味着更高的可靠性和业务连续性。

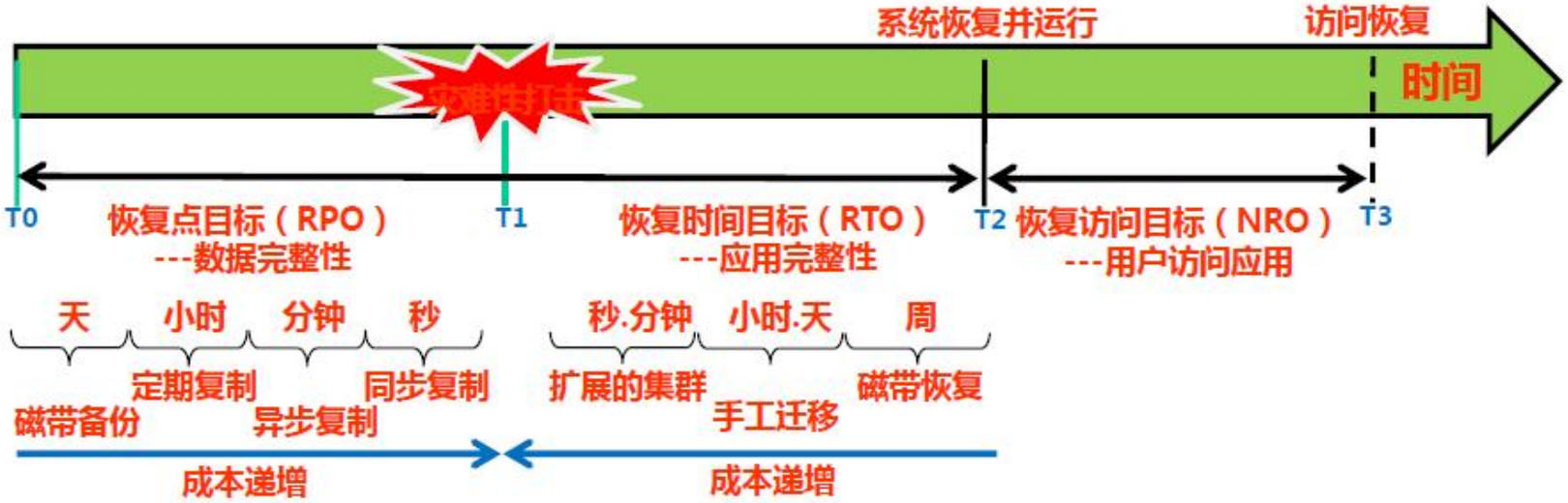
RTO：恢复时间目标，指在发生故障或灾难后，系统需要恢复到正常运行状态所需的时间。RTO衡量了系统从故障中恢复功能的速度。较短的RTO表示系统能够更快地恢复，并减少业务中断时间。

RPO：恢复点目标，指在发生故障或灾难时，系统能够恢复到的数据状态的时间点。RPO定义了业务对数据丢失的容忍程度。较小的RPO表示系统能够更少地丢失数据，并降低业务损失。

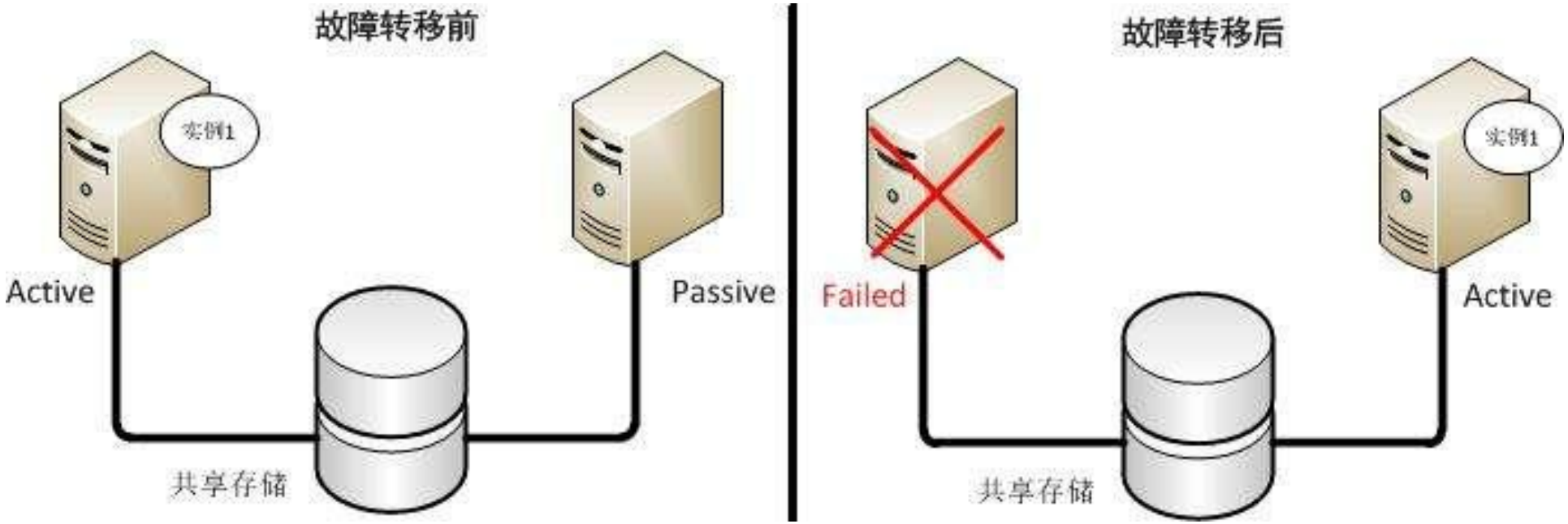
容灾恢复-指标等级

国标灾难恢复能力与业务恢复能力

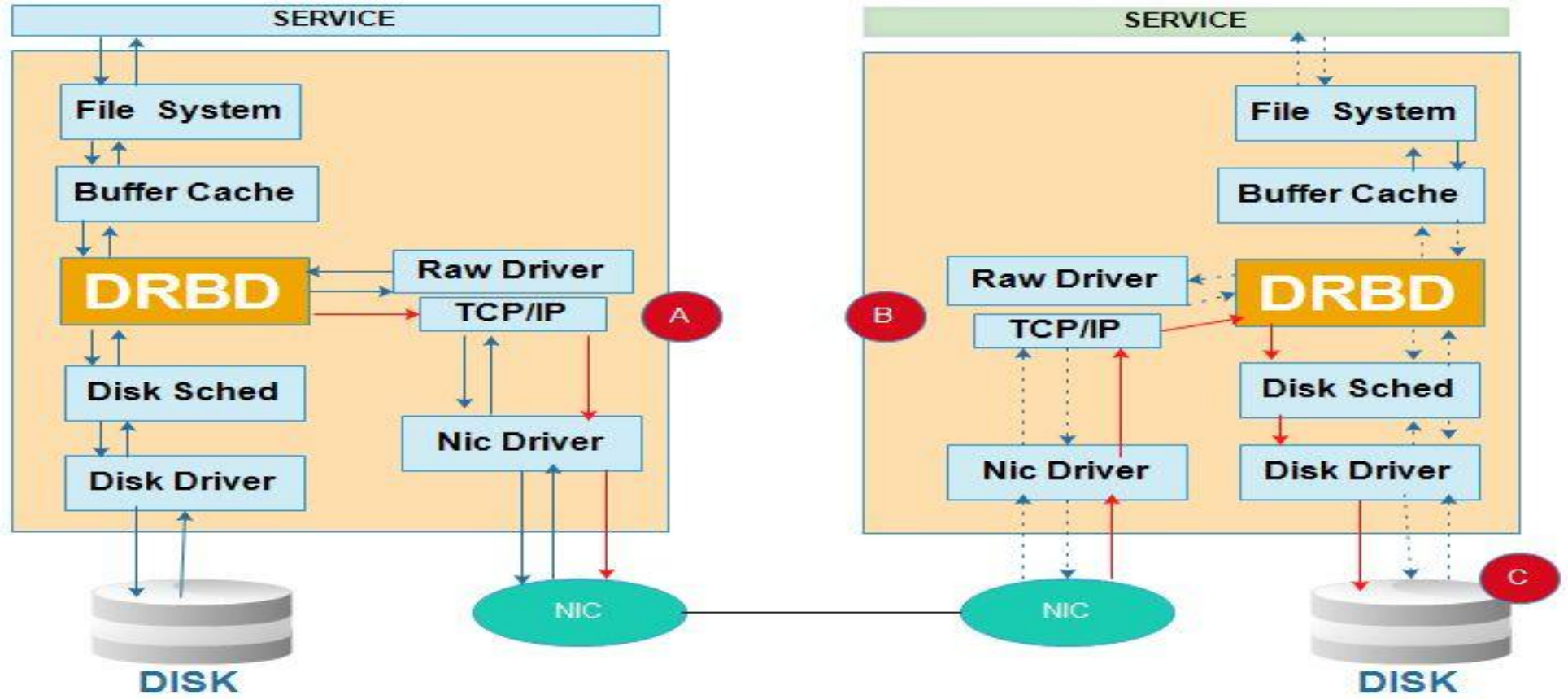
	容灾等级	RTO	RPO
数据大集中初期建设目标	第1级	2天以上	1天至7天
	第2级	24小时以上	1天至7天
	第3级	12小时以上	数小时至1天
	第4级	数小时至2天	数小时至1天
最终目标	第5级	数分钟至2天	0至30分钟
	第6级	数分钟	0



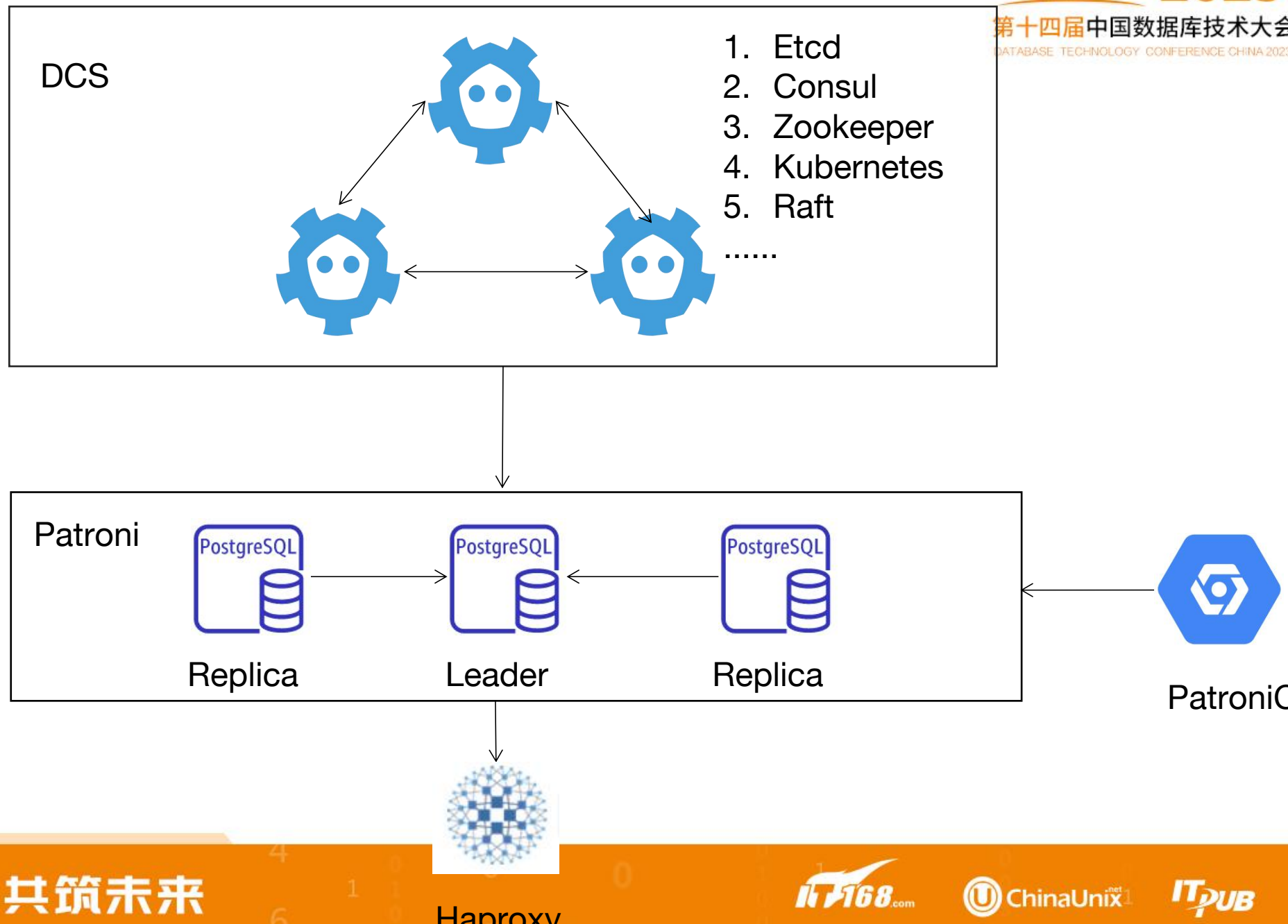
共享磁盘架构



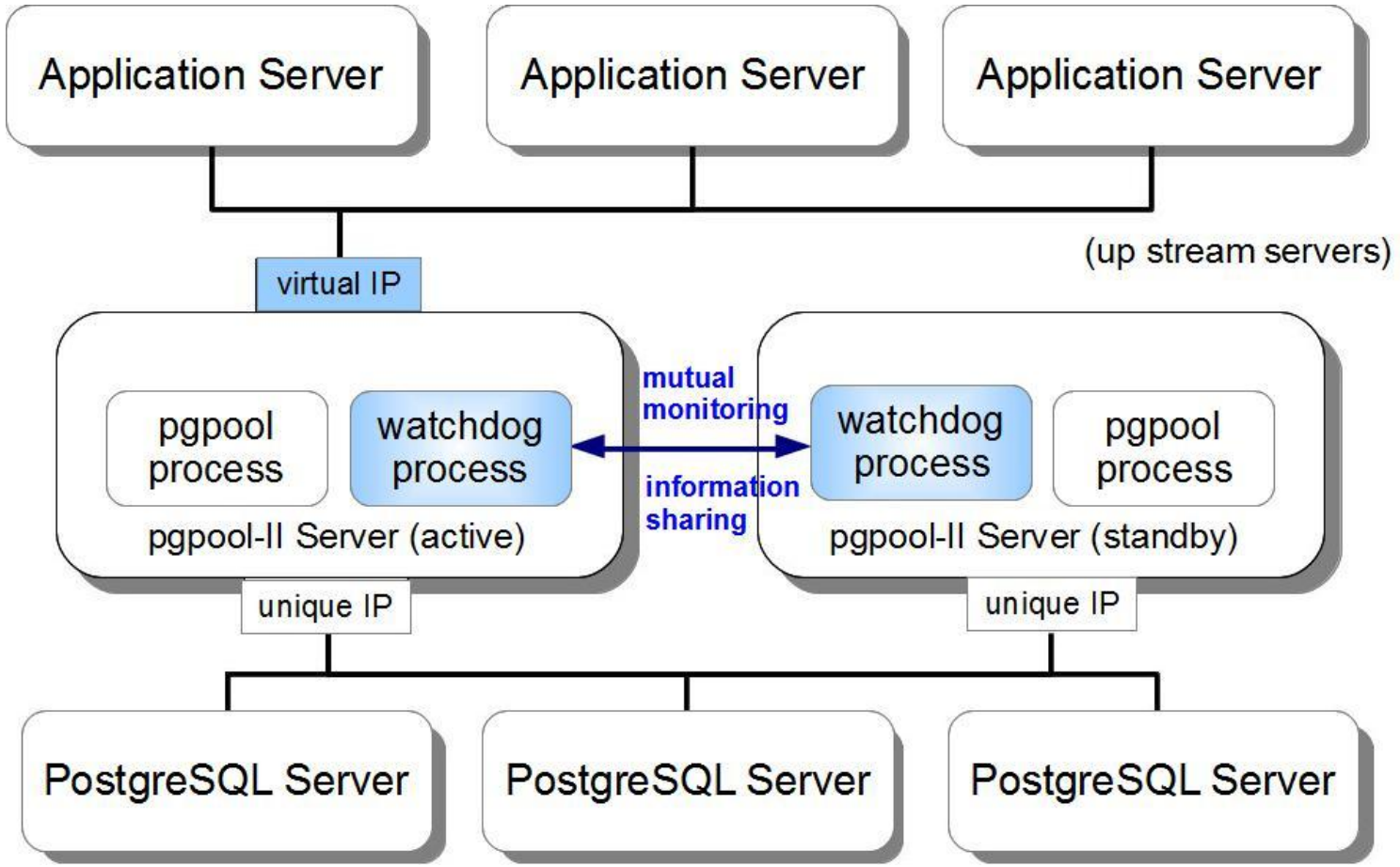
DRBD 分布式复制块设备



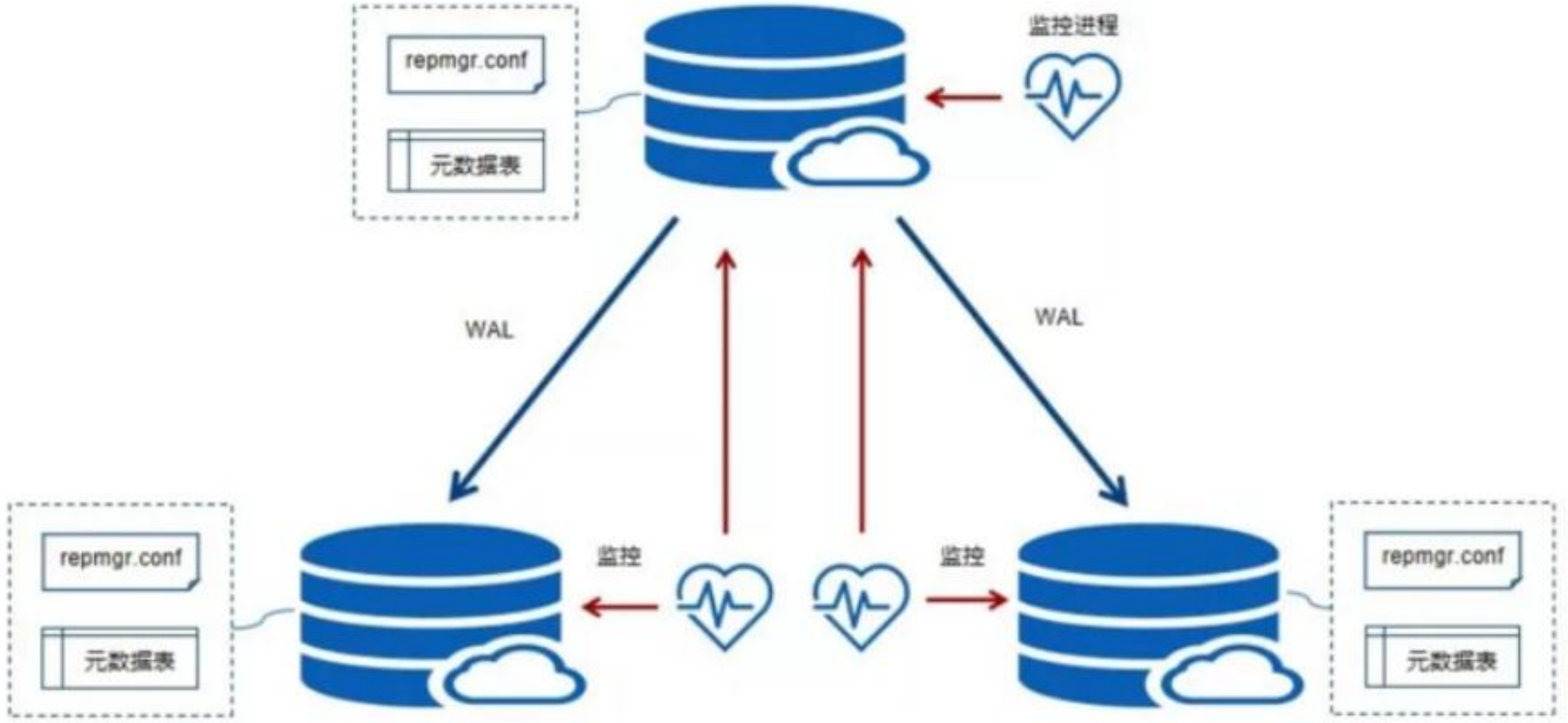
Patroni


DCS
1. Etcd
2. Consul
3. Zookeeper
4. Kubernetes
5. Raft
.....
Patroni
PostgreSQL
PostgreSQL
PostgreSQL
Replica
Leader
Replica
PatroniCt
Haproxy

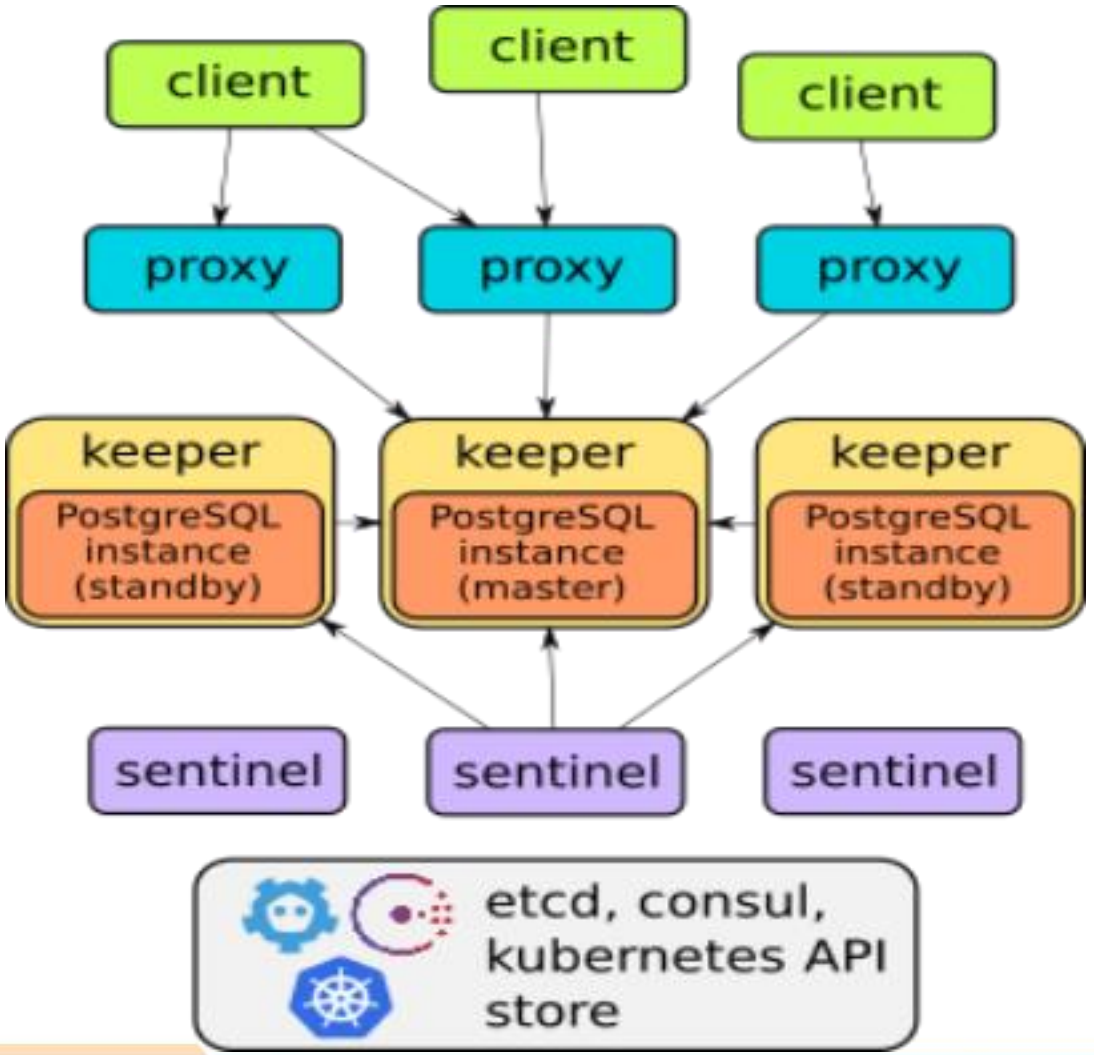
Pgpool-ii


Application Server
Application Server
Application Server
(up stream servers)
virtual IP
pgpool process
watchdog process
pgpool-II Server (active)
mutual monitoring
information sharing
watchdog process
pgpool process
pgpool-II Server (standby)
unique IP
unique IP
PostgreSQL Server
PostgreSQL Server
PostgreSQL Server

Repmgr


repmgr.conf
元数据表
监控进程
WAL
WAL
repmgr.conf
元数据表
监控
监控
repmgr.conf
元数据表

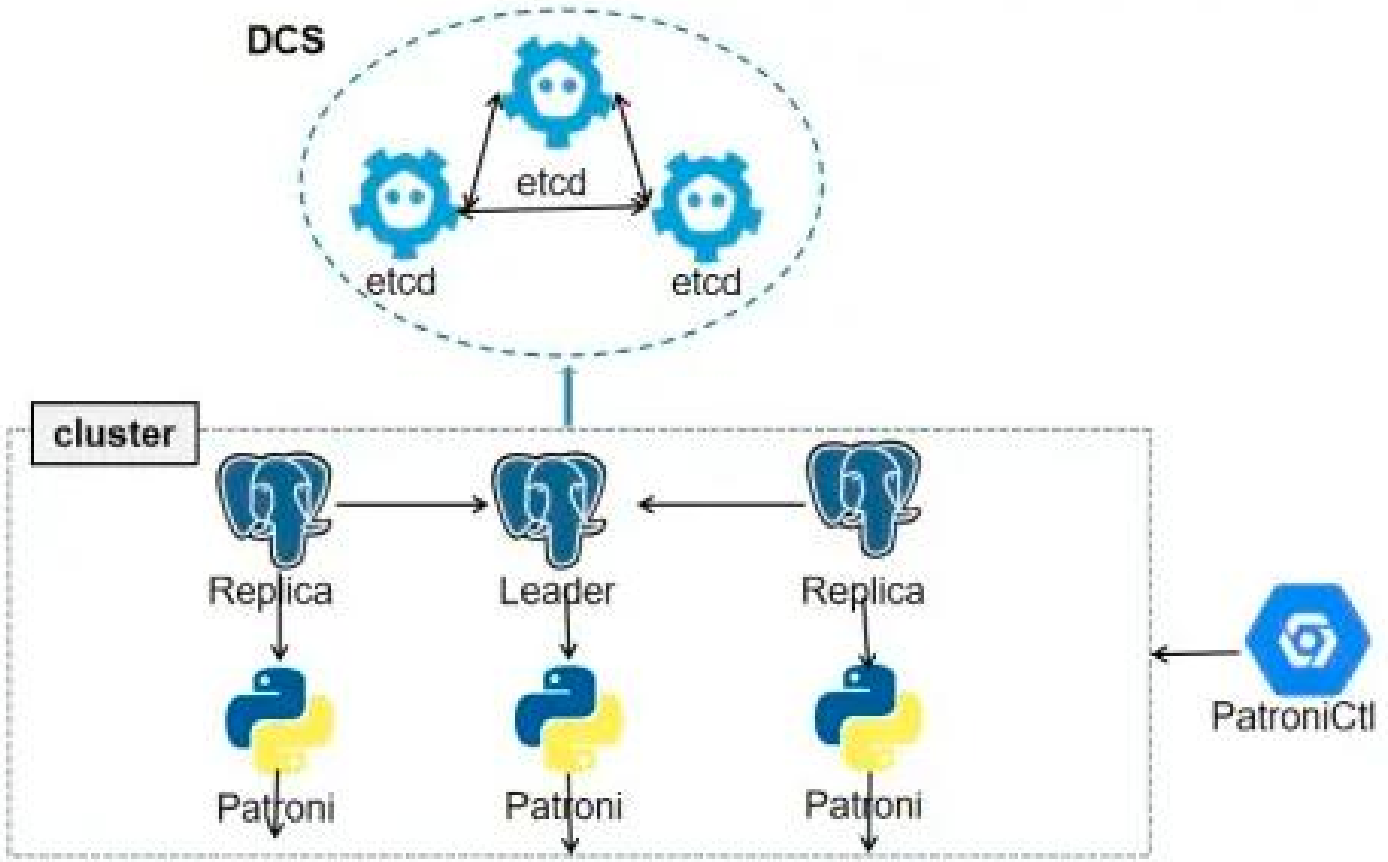
stolon


client
client
client
proxy
proxy
proxy
keeper
PostgreSQL instance (standby)
keeper
PostgreSQL instance (master)
keeper
PostgreSQL instance (standby)
sentinel
sentinel
sentinel
etcd, consul, kubernetes API store

脑裂

如何产生?
1.网络问题
2.主备切换异常
3.网络孤岛
4.原节点以主角色加入集群

处理处理?
1.网关
2.仲裁节点
3.DCS软件(ETCD/ZK)
4.发生网络隔离时,节点暂停



检测

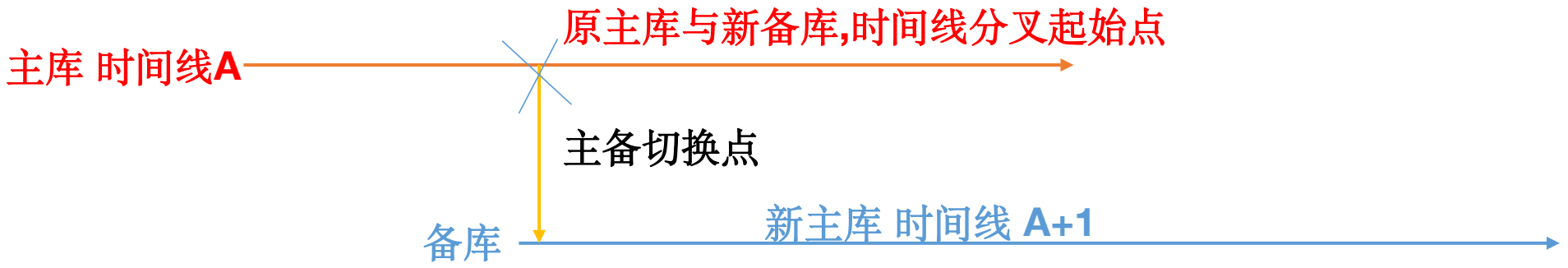
数据库实例活动状态检测,如何更准确?

常用方式
1.pg_isready
2.pg_ctl staus
3.pg_controldata
4.ps - ef |grep postgres
5.psql - c ‘select 1;’

当实例运行过程中,部分文件损坏后,
以上几种方式无法感知检测。

```
'select count(*) from pg_authid;',
'select count(*) from pg_class;',
'select count(*) from pg_auth_memb
ers;',
'select count(*) from pg_database;',
'select count(*) from pg_index;',
'select count(*) from pg_tablespace;',
'select count(*) from pg_namespace;',
'select count(*) from pg_attribute;',
'select count(*) from pg_depend;',
'select count(*) from pg_statistic;',
'select count(*) from pg_stat_replicati
on;'
```

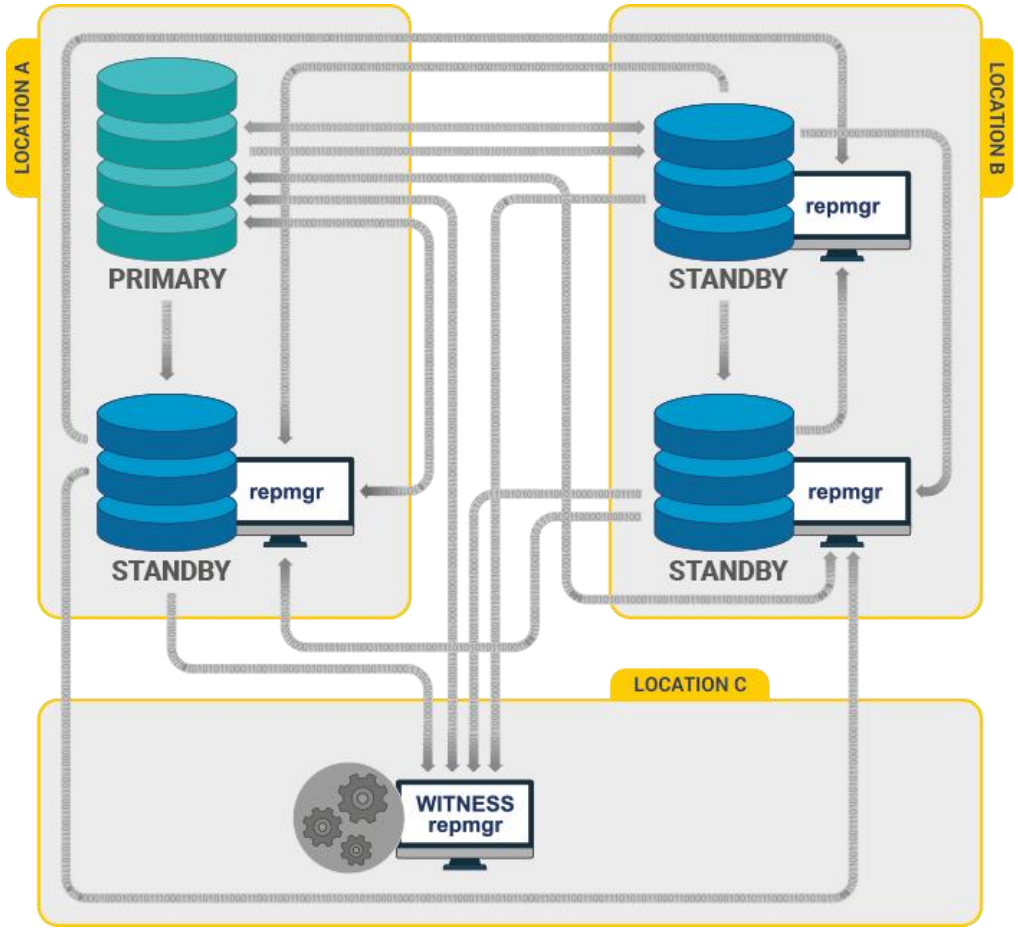
时间线分叉

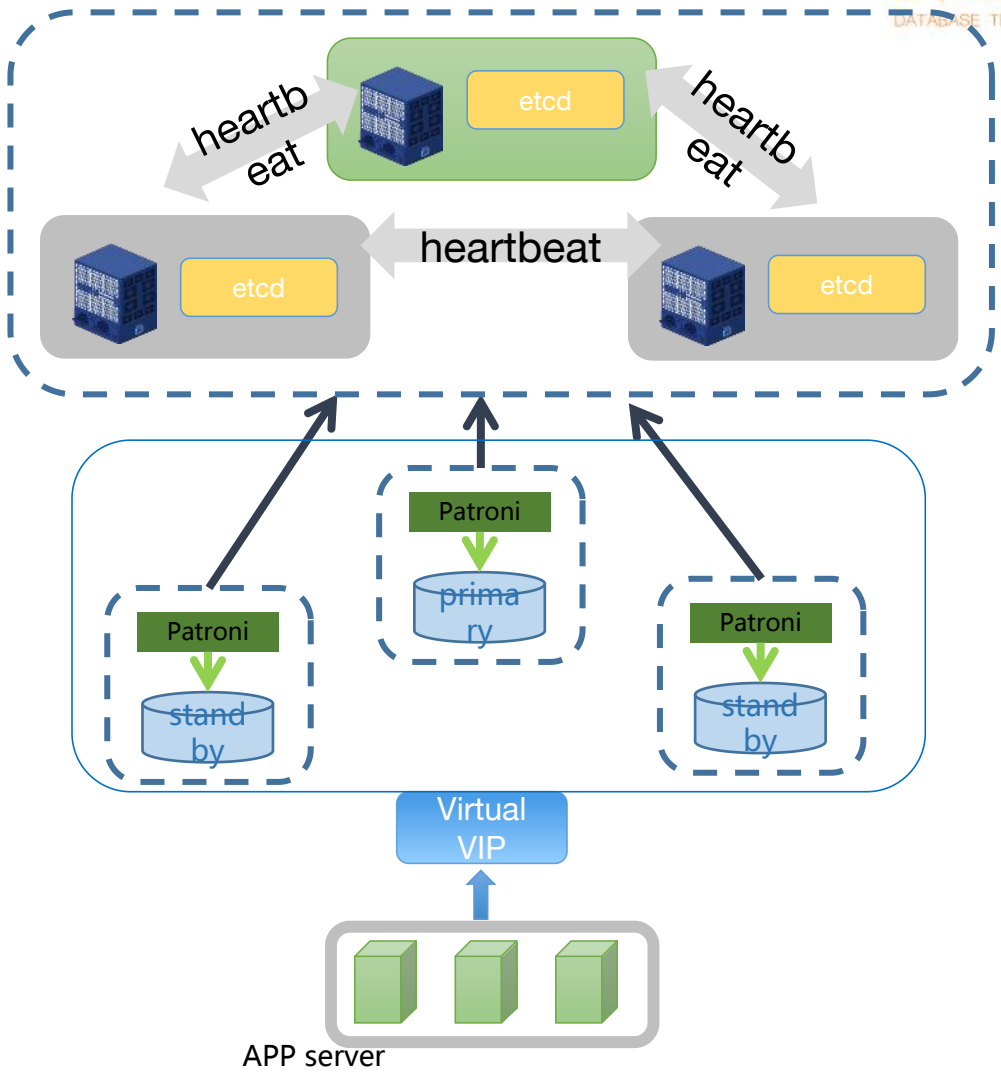


pg_rewind 拉齐时间线

使用限制 需要开启配置参数full_page_writes、wal_log_hints或者initdb时指定checksum

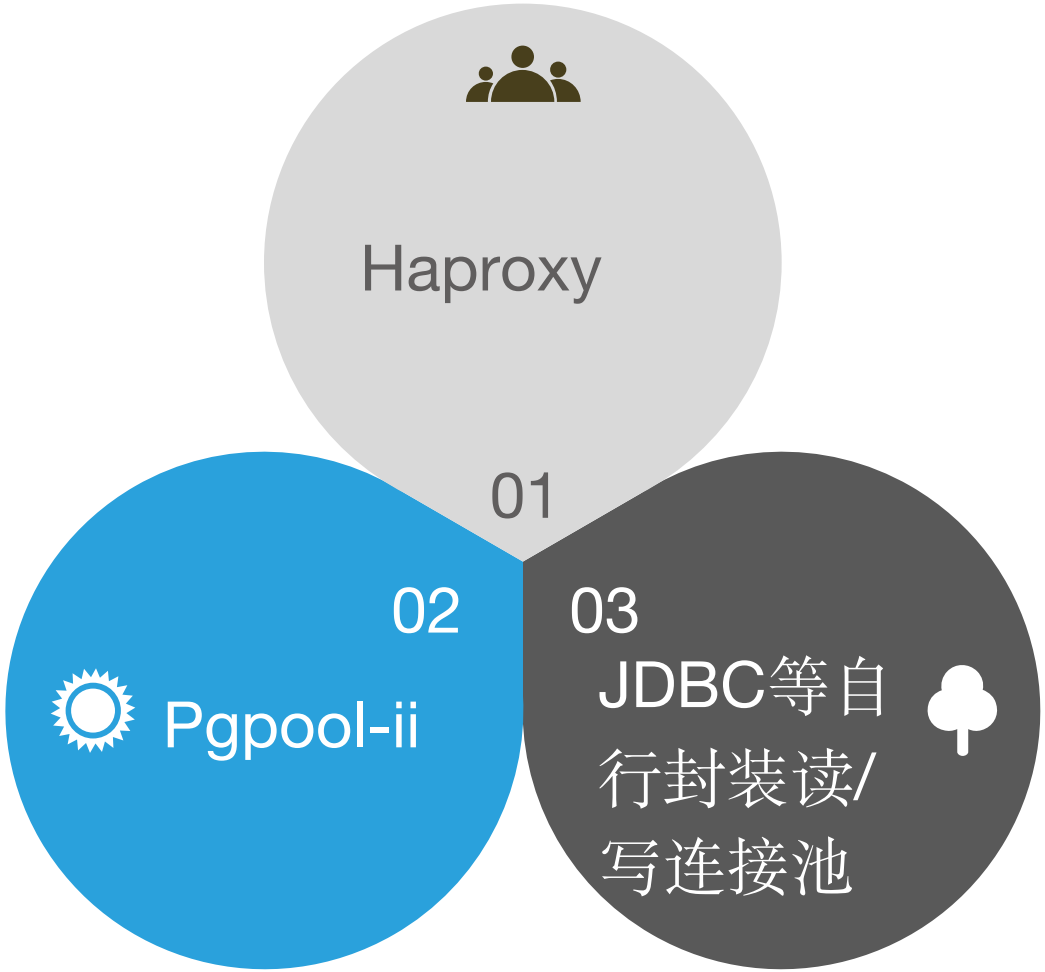
选举仲裁


LOCATION A
LOCATION B
LOCATION C
PRIMARY
STANDBY
STANDBY
STANDBY
repmgr
repmgr
repmgr
WITNESS
repmgr

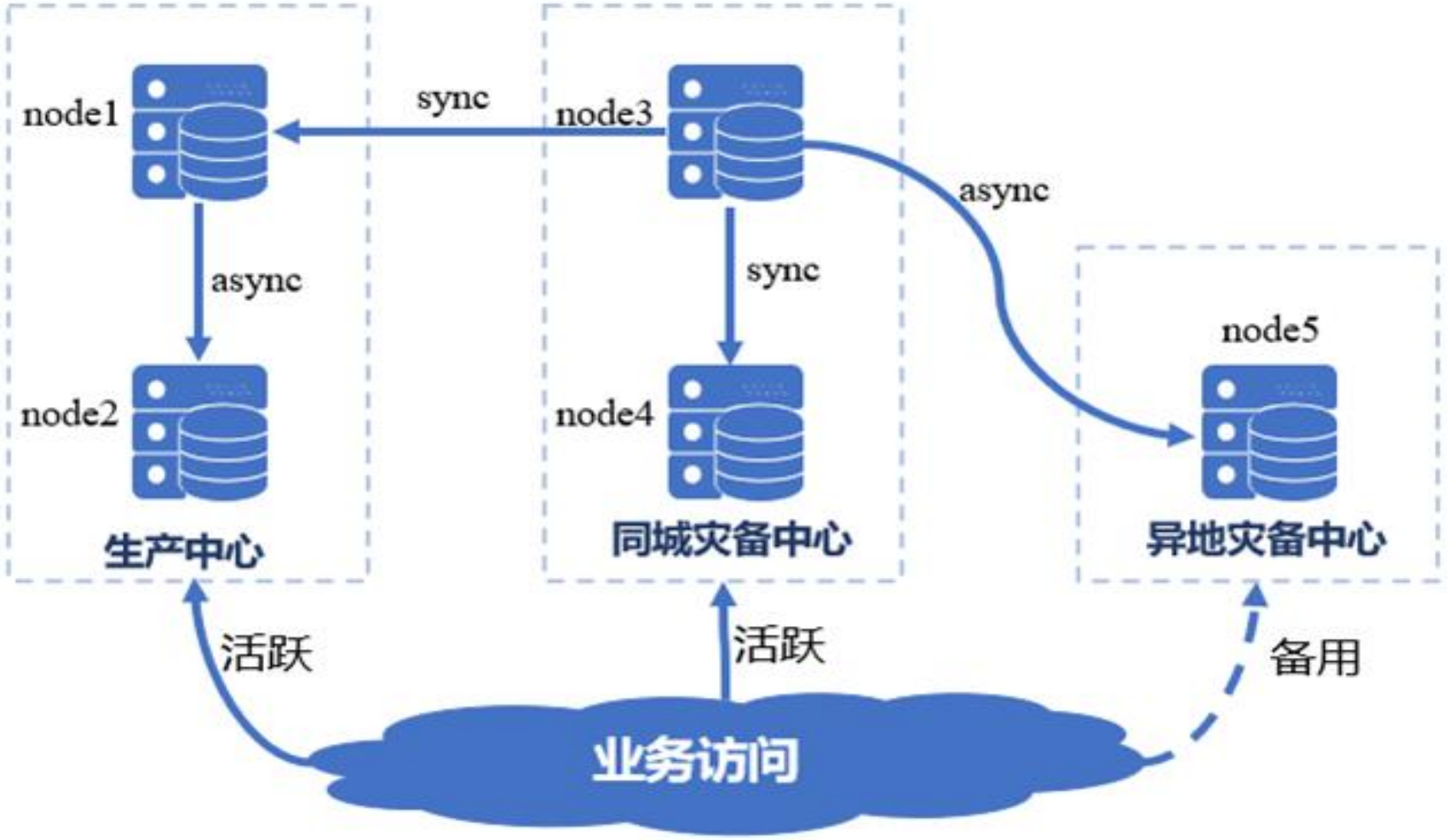

heartb
eat
etcd
heartb
eat
etcd
heartbeat
etcd
Patroni
Patroni
Patroni
prima
ry
stand
by
stand
by
Virtual
VIP
APP server

读写分离

- 性能损耗?
- 业务实时性要求,二八原则?
- 主从切换后,如何快速识别?



两地三中心


node1
sync
node3
async
async
sync
node5
node2
node4
生产中心
同城灾备中心
异地灾备中心
活跃
活跃
备用
业务访问

实现极致的SLA、RTO和RPO

需要综合考虑以下因素:

1.高可用性架构：采用冗余的硬件、网络和服务配置，以确保系统在故障发生时能够持续提供服务。

2.数据备份与复原：定期备份和存储数据，并实施有效的数据恢复策略，以最小化数据丢失并快速恢复。

3.故障容错与负载均衡：使用负载均衡器、冗余服务器和故障切换技术，以提高系统的可用性和故障恢复能力。

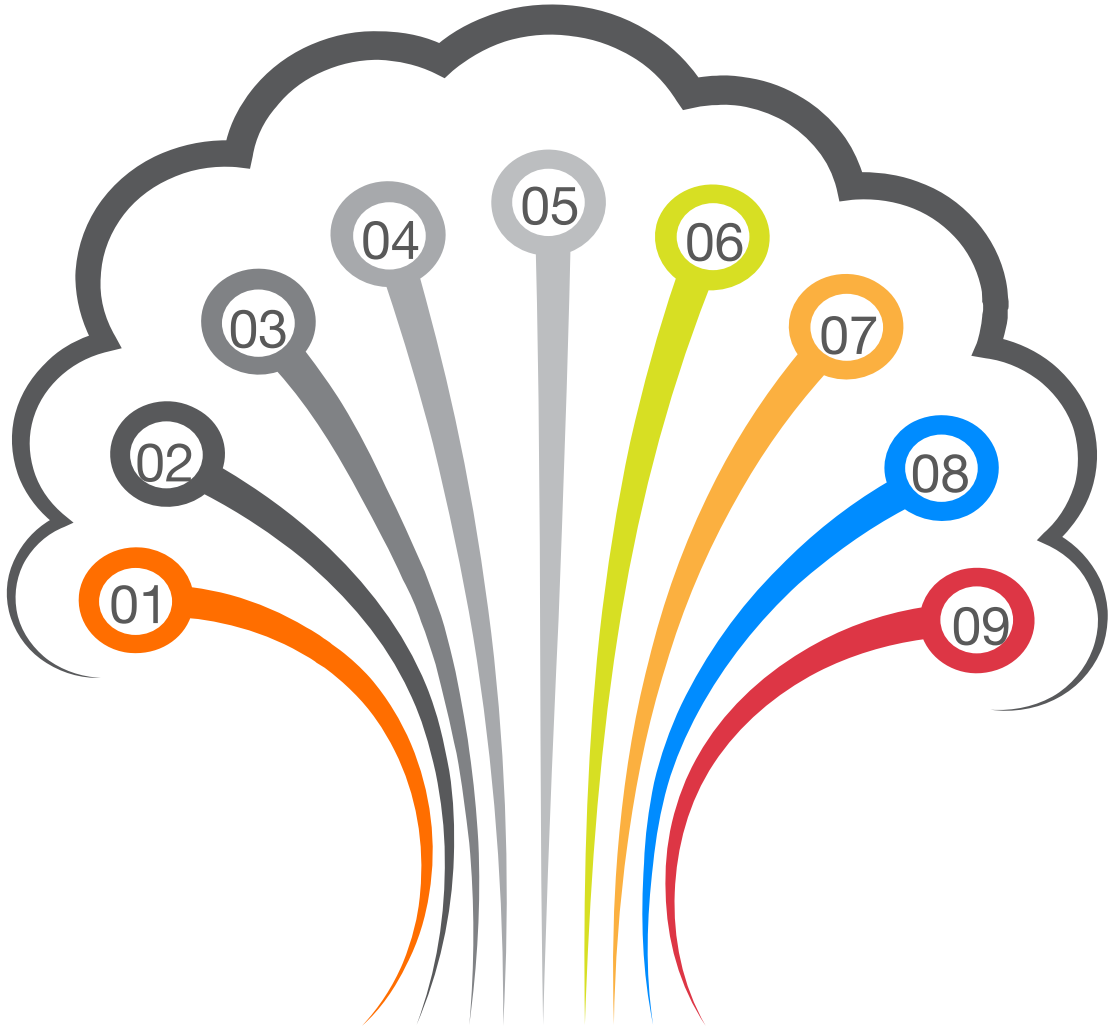
4.灾难恢复计划（DRP）：制定详细的灾难恢复计划，包括故障检测、通信、数据恢复和业务优先级。

5.监控与自动化：实施全面的监控机制，及时检测故障并采取自动化的恢复措施，缩短故障处理时间。

6.测试与演练：定期进行灾难恢复演练和测试，以验证恢复策略和流程的有效性，并及时修正发现的问题。

PostgreSQL高可用技术在专网通信领域的实践与应用

Why choose PostgreSQL ?



01 多副本同步复制.

02 丰富的安全控制机制 .

03 丰富的外部扩展支持.

04 完善的SQL标准支持.

05 FDW外部表支持.

06 维护和使用成本低.

07 全文检索.

08 空间数据库.

09 活跃的社区支持.

Why choose Patroni?

- 开箱即用高可用解决方案
- 降低运维成本，提升服务效率
  - 模板化快速部署
  - 避免PG集群脑裂发生
  - 提供备用集群功能
  - 一键故障切换
  - 故障自动转移
  - Watchdog机制

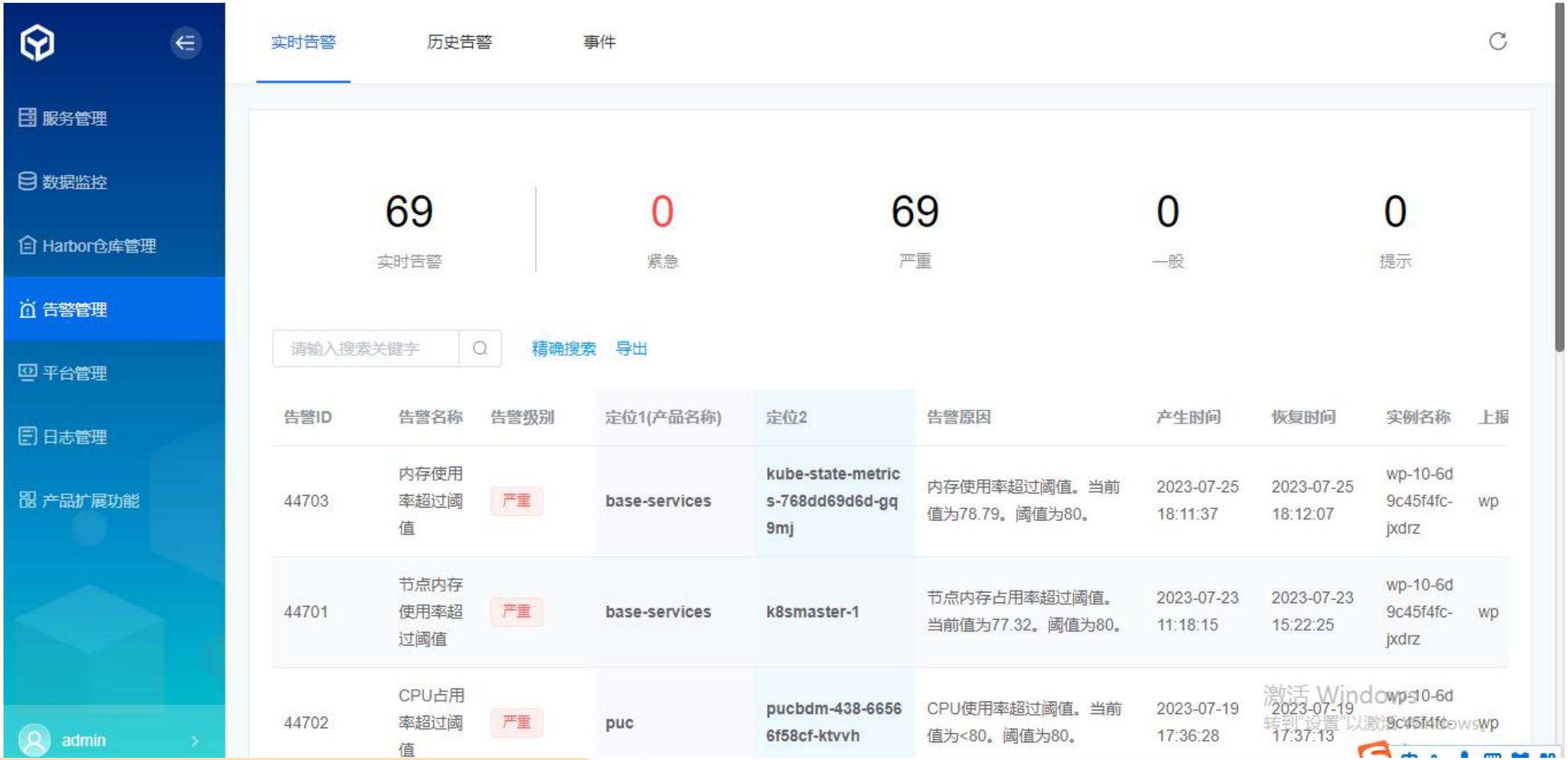
自研DCS-Agent

自研DCS-Agent集群管理程序
结合Patroni进行集群部署管理

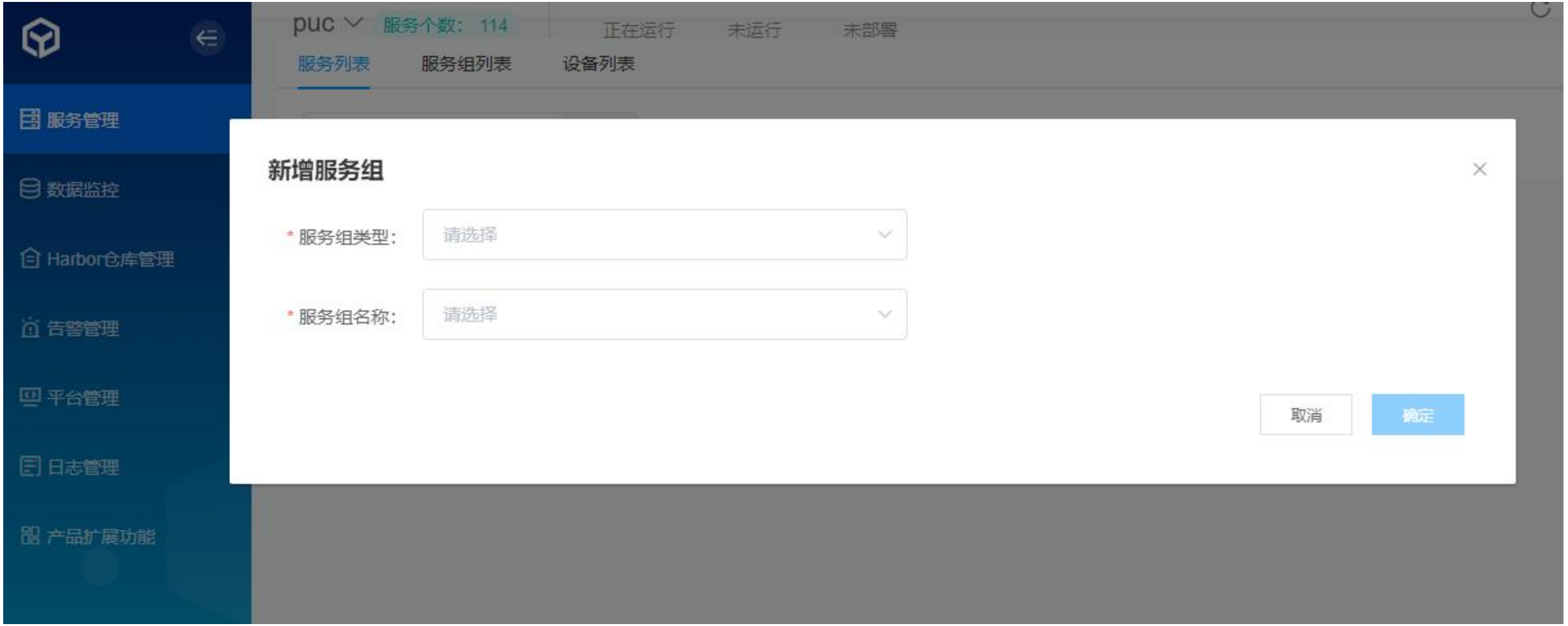
自动化运维


产品列表
K8S组列表
服务管理
数据监控
Harbor仓库管理
告警管理
平台管理
日志管理
产品扩展功能
请输入搜索关键字
base-services
服务个数：29
29
正在运行
0
未运行
0
未部署
ID： 1
类型： YsP
版本： V3.5.07.040
puc
服务个数：114
114
正在运行
0
未运行
0
未部署
ID： 356
类型： PUC
版本： V4.1.01.210

自动化运维

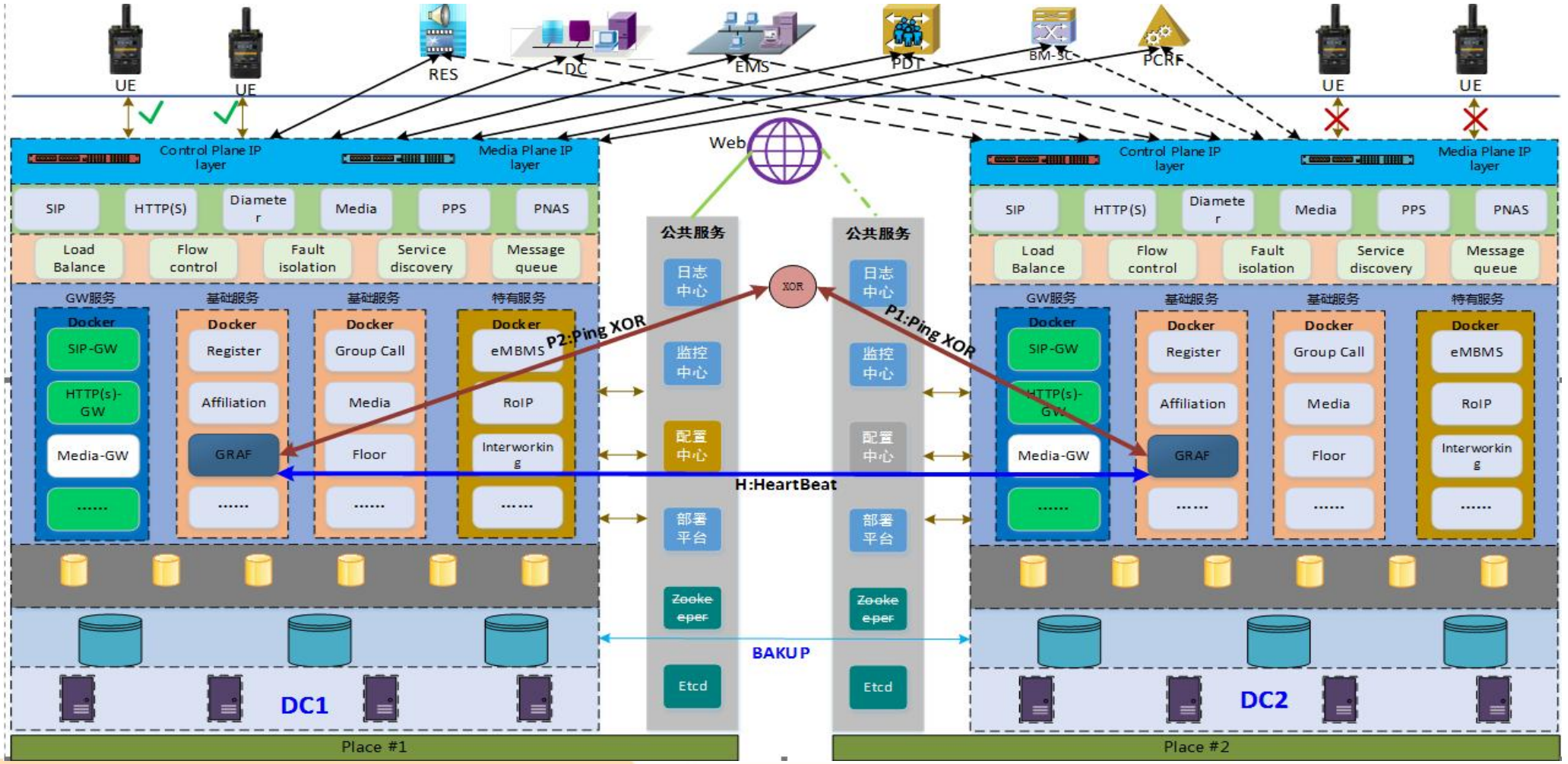


自动化运维


puc
服务个数: 114
正在运行
未运行
未部署
服务列表
服务组列表
设备列表
服务管理
数据监控
Harbor仓库管理
告警管理
平台管理
日志管理
产品扩展功能
新增服务组
* 服务组类型:
请选择
* 服务组名称:
请选择
取消
确定

总体架构

平台融合、业务架构融合、数据存储融合 多元融合架构



THANKS