Patroni
使用维护手册

神州飞象（北京）数据科技有限公司

目录

关于本手册.....4

提供内容.....4

读者.....4

如何使用本手册.....4

1　Patroni 集群说明.....5

1.1　软件说明.....5

1.2　组件说明.....5

1.3　Patroni 流程.....6

1.4　使用优势.....7

1.5　使用限制.....8

2　集群的启停.....8

2.1　启动.....8

2.1.1　启动 zookeeper（root）.....9

2.1.2　启动 Patroni（普通用户）.....9

2.2　关闭.....9

2.2.1　关闭 patroni（普通用户）.....10

2.2.2 关闭 zookeeper（root）.....10
2.3 重启.....11
3 Patroni 参数的更改.....11
4 Patronictl 集群维护命令.....15
4.1 查看集群状态.....16
4.2 发送一条 SQL 语句.....17
4.3 获取主节点 dsn 信息.....17
4.4 重启集群.....17
4.5 手动执行主备切换.....19
4.6 手动 failover 一个节点.....20
4.7 在 DCS 中删除集群信息.....21
4.8 重新初始化节点.....21
5 Postgresql 数据库插件的加载.....22
6 Watchdog 功能.....22
6.1 适用场景:.....22
6.2 Watchdog 配置步骤.....22
6.3 开机启动服务.....24
7 添加节点.....24
7.1 添加 patroni 与数据库节点.....24
8 日志级别调整.....25
9 故障.....27
9.1 故障检查流程.....27
9.2 检查项.....27

关于本手册

提供内容

patroni 使用维护手册将向读者介绍 patroni 高可用集群的使用方法。通过该手册，您将学会如何使用 patroni 高可用集群。

读者

- 本指南适用于数据库管理员、应用工程师、系统工程师等能独立完成部署的人员。

如何使用本手册

本手册包含以下几章：

启停，参数的分类以及修改，patroni 集群的维护命令，添加 patroni 节点，故障检测流程等。

在文档使用过程中，您可以顺序阅读每一章， 也可以根据目录，寻找您需要的部分进行使用。具体的故障测试请查看测试文档。祝您阅读愉快。

1 Patroni 集群说明

1.1 软件说明

patroni 是一款运用 dcs 存储集群来存储信息、主备状态与配置,通过 patroni 来检测并且实现主备库自动切换的软件。使用一套模板化的配置文件来自动搭建初始化数据库流复制集群以及配置数据库。patroni 高可用集群由 postgresql,patroni,dcs 存储组成。。

1.2 组件说明

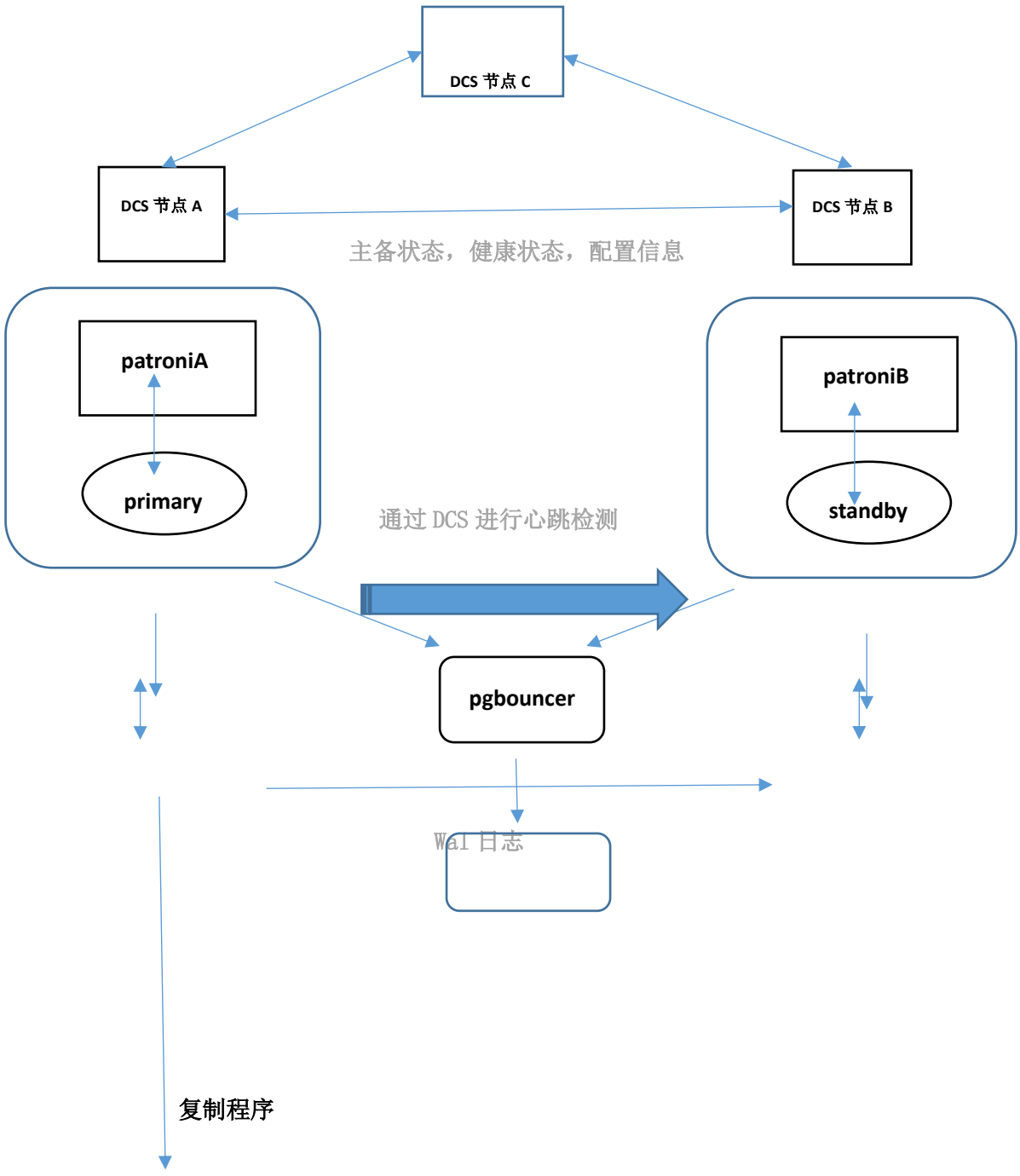
组件分别的作用(不包括 pg 数据库):

- patroni:通过参数文件来配置自动初始化数据库搭建流复制（配置 pg 参数文件、创建用户、可以配置预加脚本），指定 zookeeper 节点等。负责通过一个 api 接口连接到 dcs(分布式存储系统集群),向其插入键值记录 patroni 参数、数据库参数、主备信息以及连接信息。平时通过对 zookeeper 中的信息进行更新、读取来判断集群的健康状态。在主备切换或者做恢复时通过向 zookeeper 读取主备信息来判断各节点的状态进行切换。

- zookeeper:最少需要三个节点且为奇数来进行 leader 选举（脑裂发生时 zookeeper 集群会僵死等待恢复，不会发生都认为自己是主的情况）。存储并在各个节点上同步键值信息。

1.3 Patroni 流程

dcs+patroni+postgres+pgbouncer+复制程序基本架构图



基本流程：

- patroni 自动创建主备流复制集群并且向 zookeeper 读取以及更新键值
- zookeeper 存储,同步键值信息
- patroni 进行循环检测，如果发现当前节点或者主节点发生异常，会执行相对的应对措施（重启节点、主备切换等）

1.4 使用优势

- 自动检测主备状态进行切换
- 统一模板配置
- 不会发生因双主而照成的脑裂现象，主节点是哪个节点记录在了 dcs 中，恢复后 pg_rewind 会同步主备的时间线
- 在线添加 zookeeper、patroni 节点以及数据节点
- 支持同步异步流复制，级联流复制
- 异步流复制可设置最小丢失数据量

- 使用 pg_rewind 进行恢复，缩短恢复时间
- 拥有 watchdog 机制来解决节点数据库因内存资源超出而照成的崩溃或者是高负载系统下 patroni 被卡死这样的单点故障

1.5 使用限制

- 需要至少三个以上且为奇数的 zookeeper 节点
- 底层基于的是流复制
- 部分 patroni 自带参数需要通过更改 dcs 中键值来修改
- 因故障发生而未提交的事物会回滚，会话需要客户端重新发起连接
- 在复制程序开启的情况下 pg_ctl 命令关闭主库，

2 集群的启停

2.1 启动

zookeeper ⇨ Patroni

2.1.1 启动 zookeeper（root）

```
开启命令

三个zookeeper节点均需执行：

/root/zookeeper-3.3.6/bin/start

查看zookeeper运行状态：

/root/zookeeper-3.3.6/bin/status
```

2.1.2 启动 Patroni（普通用户）

Patroni 所在节点均需执行:

输出到屏幕运行

```
/home/postgres/patroni-1.4.3/patroni.py /home/postgres/patroni-
1.4.3/postgresmq.yml
```

后台运行

```
nohup /home/postgres/patroni-1.4.3/patroni.py
/home/postgres/patroni-1.4.3/postgresmq.yml  >
/home/postgres/logfile/patroni_log 2>&1 &
```

2.2 关闭

建议：

- 正常来说，**强烈不建议去关闭集群**。高可用集群启动后是不需要关闭的。组件或节点故障都会有相应的高可用机制。

如果遇到特殊情况需要关闭集群，最完美的关闭情况请依照以下顺序关闭:

确认数据库再没有业务连接进来，并且备机 wal 日志更新没有延迟

Patroni ⇨ zookeeper

2.2.1 关闭 patroni（普通用户）

node1，node2 均需执行

首先关闭每个节点的 patroni 进程

```
[yd@node1 ~]$ ps -ef|grep patroni
postgres 2853  2823  0 14:32 pts/1    00:00:39 python ./patroni.py
postgres000.yml
postgres 5873  5840  0 16:16 pts/0    00:00:00 grep patroni
[yd@node1 ~]$ kill -9 2853
```

关闭数据库，先关闭主库，再关闭从库

```
主库:
pg_ctl -D /home/postgres/flyingdb-v3-logical/data stop
从库:
pg_ctl -D /home/postgres/flyingdb-v3-logical/data stop
```

2.2.2 关闭 zookeeper（root）

三个节点均需执行

```
开启命令
```

三个zookeeper节点均需执行：

```
/root/zookeeper-3.3.6/bin/stop
```

查看zookeeper运行状态：

```
/root/zookeeper-3.3.6/bin/status
```

2.3 重启

重启整个集群

```
/home/postgres/patroni-1.4.3/patronictl.py  -c
/home/postgres/patroni-1.4.3/postgresmq.yml restart batman5
```

重启单个节点

```
/home/postgres/patroni-1.4.3/patronictl.py  -c
/home/postgres/patroni-1.4.3/postgresmq.yml restart batman5
postgresql0
```

3 Patroni 参数的更改

Patroni 参数和数据库参数配置有 3 种类型：

- 必须在 dcs 中更改的**数据库参数**:

这些参数在 patroni 参数文件中配置，包括了流复制参数和部分连接参数。

```
vi postgresql.conf

# Do not edit this file manually!

# It will be overwritten by Patroni!

include 'postgresql.base.conf'


cluster_name = 'batman5'

hot_standby = 'on'

listen_addresses = '0.0.0.0'

max_connections = '100'

max_locks_per_transaction = '64'

max_prepared_transactions = '0'

max_replication_slots = '10'

max_wal_senders = '10'

max_worker_processes = '8'

port = '5433'

track_commit_timestamp = 'off'

unix_socket_directories = '/tmp'

wal_keep_segments = '8'

wal_level = 'logical'

wal_log_hints = 'on'

hba_file = '/home/postgres/flyingdb-v3-logical/data/pg_hba.conf'

ident_file = '/home/postgres/flyingdb-v3-logical/data/pg_ident.conf'
```

更改方式:

依据字典格式在 dcs 中更改，更改完后在全局生效，依据该参数是动态或是静态参数来选择是否需要重启数据库

zk 的更改格式

进入 zk

```
[root@test2 bin]# cd /root/zookeeper-3.3.6/bin

[root@test2 bin]# ./zkCli.sh -server localhost:2181
```

查看参数命令，能看到下面的这些参数

```
[zk: localhost:2181(CONNECTED) 0] get /service/batman5/config

{"ttl":30,"maximum_lag_on_failover":1048576,"retry_timeout":10,"postg
resql":{"use_pg_rewind":true,"parameters":{"wal_level":"logical"}},"l
oop_wait":10}

cZxid = 0x120000002a

ctime = Fri Mar 16 11:02:43 CST 2018

mZxid = 0x1c0000017c

mtime = Thu Mar 22 11:37:56 CST 2018

pZxid = 0x120000002a

cversion = 0

dataVersion = 10

aclVersion = 0

ephemeralOwner = 0x0

dataLength = 151

numChildren = 0
```

更改例子：

异步切同步

```
set /service/batman5/config
{"ttl":30,"maximum_lag_on_failover":1048576,"synchronous_mode":"true"
,"retry_timeout":10,"postgresql":{"use_pg_rewind":true,"parameters":{
"wal_level":logical,"max_connections":100,"max_locks_per_transaction"
:64,"max_prepared_transactions":0,"max_replication_slots":10,"max_wal
_senders":10,"max_worker_processes":8,"track_commit_timestamp":"off",
"wal_keep_segments":8,"wal_log_hints":"on"}},"loop_wait":10}
```

同步切异步

```
set /service/batman5/config
{"ttl":30,"maximum_lag_on_failover":1048576,"synchronous_mode":null,"
retry_timeout":10,"postgresql":{"use_pg_rewind":true,"parameters":{"w
al_level":logical,"max_connections":100,"max_locks_per_transaction":6
4,"max_prepared_transactions":0,"max_replication_slots":10,"max_wal_s
enders":10,"max_worker_processes":8,"track_commit_timestamp":"off","w
al_keep_segments":8,"wal_log_hints":"on"}},"loop_wait":10}
```

改完后自动生效。

- 必须在 dcs 中更改的 patroni 参数

这类参数修改方法与上类参数一致，如 `ttl`，`maximum_lag_on_failover`，`synchronous_mode`，`retry_timeout`

Patroni 参数：

ttl – 当一段时间内没有人更新 dcs 中 leader key，则视作到期，删除并重新选举新的主节点。 默认 30 seconds

loop_wait – Patroni 多久循环一次 HA loop。 默认 10 seconds

retry_timeout – 如果更新 dcs 中 leader key 失败, Patroni 将在这段时间内尝试再更新。默认 10 seconds

maximum_lag_on_failover – 每一次的 HA loop, 主节点把 wal_position 写入到 dcs 中. 而从属于这个主的每个从节点将会用自己的最后一个 wal_position 与主节点的进行对比。如果主从的差距大于 maximum_lag_on_failover ， patroni 便不会再让这个差距变的更大。默认 1048576bit

master_start_timeout – 主节点启动超时。默认 300 seconds

synchronous_mode – 同步流复制配置选项。默认 false

- 可以在数据库配置文件中更改的参数

在 postgresql.base.conf 中修改参数，包括了绝大部分数据库参数。修改的参数不能和 postgresql.conf 中的参数重复

更改例子：

主备修改postgresql.base.conf文件

```
work_mem = 1MB
```

重启patroni集群，在任意patroni节点执行：

```
cd /home/postgres/patroni-1.4.2

[postgres@test1 patroni-1.4.2]$ ./patronictl.py -c postgresmq.yml
restart batman5
```

重启集群后参数生效

4 Patronictl 集群维护命令

我们用 **patronictl** 命令可以:

- **查看集群状态**
- **发送一条 SQL 语句**
- **获取主节点 dsn 信息**
- **重启集群**
- **手动执行主备切换**
- **手动 failover 一个节点**
- **在 DCS 中删除集群信息**
- **重新初始化节点**

```
Usage: patronictl.py [OPTIONS] COMMAND [ARGS]...


Options:
  -c, --config-file TEXT  Configuration file
  -d, --dcs TEXT          Use this DCS
  --help                 Show this message and exit.
```

```
Commands:
  configure    Create configuration file
  dsn          Generate a dsn for the provided member,...
  edit-config  Edit cluster configuration
  failover     Failover to a replica
  flush        Flush scheduled events
  list         List the Patroni members for a given Patroni
  pause        Disable auto failover
  query        Query a Patroni PostgreSQL member
  reinit       Reinitialize cluster member
  remove       Remove cluster from DCS
  restart      Restart cluster member
  resume       Resume auto failover
  scaffold     Create a structure for the cluster in DCS
  show-config  Show cluster configuration
  switchover   Switchover to a replica
  version      Output version of patronictl command or a...
```

4.1 查看集群状态

方法一:patronictl 命令

–c 指定 patroni 配置文件 batman 是默认集群名字在 patroni 参数文件设置

```
[postgres@test3 patroni-1.4.2]$ ./patronictl.py  -c postgresmq.yml
list
+---------+-------------+---------------+--------+---------+-------
----+
| Cluster |    Member   |      Host     |  Role  |  State  | Lag in MB
|
+---------+-------------+---------------+--------+---------+-------
----+
```

```
| batman5 | postgresql0 | 192.168.1.143 | Leader | running |
|
| batman5 | postgresql1 | 192.168.1.142 |        | running |
|
+---------+-------------+---------------+--------+---------+-------
----+
```

4.2 发送一条 SQL 语句

```
[postgres@test3 patroni-1.4.3]$ ./patronictl.py -c postgresmq.yml
query batman5 --command 'select count(*) from test'

44
```

4.3 获取主节点 dsn 信息

```
[postgres@test1 patroni-1.4.3]$ ./patronictl.py -c postgresmq.yml
dsn batman5

host=192.168.1.143 port=5432
```

4.4 重启集群

```
重启整个集群

[postgres@test3 patroni-1.4.2]$ ./patronictl.py  -c postgresmq.yml
restart batman5



+---------+-------------+---------------+--------+---------+-------
----+
| Cluster |    Member   |     Host      |  Role  |  State  | Lag in MB
|
+---------+-------------+---------------+--------+---------+-------
----+
| batman5 | postgresql0 | 192.168.1.143 |        | running |
|
```

```
| batman5 | postgresql1 | 192.168.1.142 | Leader | running |
|
+---------+-------------+---------------+--------+---------+-------
----+
Are you sure you want to restart members postgresql0, postgresql1?
[y/N]: y
Restart if the PostgreSQL version is less than provided (e.g.
9.5.2)  []:
When should the restart take place (e.g. 2015-10-01T14:30)  [now]:
Success: restart on member postgresql0
Success: restart on member postgresql1
```

重启单个节点

```
[postgres@test3 patroni-1.4.2]$ ./patronictl.py  -c postgresmq.yml
restart batman5 postgresql0
+---------+-------------+---------------+--------+---------+-------
----+
| Cluster |   Member    |     Host      |  Role  |  State  | Lag in MB
|
+---------+-------------+---------------+--------+---------+-------
----+
| batman5 | postgresql0 | 192.168.1.143 |        | running |
|
| batman5 | postgresql1 | 192.168.1.142 | Leader | running |
|
+---------+-------------+---------------+--------+---------+-------
----+
Are you sure you want to restart members postgresql0? [y/N]: y
Restart if the PostgreSQL version is less than provided (e.g.
9.5.2)  []:
When should the restart take place (e.g. 2015-10-01T14:30)  [now]:
```

```
Success: restart on member postgresql0Success: restart on member
postgresql2
```

4.5 手动执行主备切换

```
[postgres@test3 patroni-1.4.2]$ ./patronictl.py -c postgresmq.yml
switchover batman5

Master [postgresql0]:

Candidate ['postgresql1'] []:

When should the switchover take place (e.g. 2015-10-01T14:30)
[now]:

Current cluster topology

+---------+-------------+---------------+--------+---------+-------
----+

| Cluster |   Member    |     Host      |  Role  |  State  | Lag in MB
|

+---------+-------------+---------------+--------+---------+-------
----+

| batman5 | postgresql0 | 192.168.1.143 | Leader | running |
|

| batman5 | postgresql1 | 192.168.1.142 |        | running |
|

+---------+-------------+---------------+--------+---------+-------
----+

Are you sure you want to switchover cluster batman5, demoting
current master postgresql0? [y/N]: y

2018-03-21 11:06:28.71009 Successfully failed over to "postgresql1"

+---------+-------------+---------------+--------+---------+-------
----+

| Cluster |   Member    |     Host      |  Role  |  State  | Lag in MB
|

+---------+-------------+---------------+--------+---------+-------
----+
```

```
| batman5 | postgresql0 | 192.168.1.143 |       | stopped |
unknown |
| batman5 | postgresql1 | 192.168.1.142 | Leader | running |
|
+---------+-------------+---------------+--------+---------+-------
----+
```

4.6 手动 failover 一个节点

```
[postgres@test3 patroni-1.4.3]$ ./patronictl.py -c postgresmq.yml
failover  batman5
Candidate ['postgresql1'] []: postgresql1
Current cluster topology
+---------+-------------+---------------+-------------+---------+-
----------+
| Cluster |    Member   |      Host     |     Role    |  State  | Lag
in MB |
+---------+-------------+---------------+-------------+---------+-
----------+
| batman5 | postgresql0 | 192.168.1.143 |    Leader   | running |
|
| batman5 | postgresql1 | 192.168.1.142 | Sync standby | running |
|
+---------+-------------+---------------+-------------+---------+-
----------+
Are you sure you want to failover cluster batman5, demoting current
master postgresql0? [y/N]: y
2018-03-29 11:03:36.98125 Successfully failed over to "postgresql1"
+---------+-------------+---------------+--------+---------+-------
----+
| Cluster |    Member   |      Host     |  Role  |  State  | Lag in MB
|
+---------+-------------+---------------+--------+---------+-------
----+
```

```
| batman5 | postgresql0 | 192.168.1.143 |       | stopped |
unknown |
| batman5 | postgresql1 | 192.168.1.142 | Leader | running |
|
+---------+-------------+---------------+--------+---------+-------
----+
```

4.7 在 DCS 中删除集群信息

```
[postgres@test3 patroni-1.4.2]$ ./patronictl.py -c postgresmq.yml
remove  batman1


+---------+--------+------+------+-------+-----------+
| Cluster | Member | Host | Role | State | Lag in MB |
+---------+--------+------+------+-------+-----------+
+---------+--------+------+------+-------+-----------+
Please confirm the cluster name to remove: batman1
You are about to remove all information in DCS for batman1, please
type: "Yes I am aware": Yes I am aware
```

4.8 重新初始化节点

```
[postgres@test3 patroni-1.4.2]$ ./patronictl.py -c postgresmq.yml
reinit batman5 postgresql1


+---------+-------------+---------------+--------+---------+-------
----+
| Cluster |    Member   |      Host     |  Role  |  State  | Lag in MB
|
+---------+-------------+---------------+--------+---------+-------
----+
```

```
| batman5 | postgresql0 | 192.168.1.143 | Leader | running |
|
| batman5 | postgresql1 | 192.168.1.142 |        | running |
|
+---------+-------------+---------------+--------+---------+-------
----+
Are you sure you want to reinitialize members postgresql1? [y/N]: y
Success: reinitialize for member postgresql1
```

5 Postgresql 数据库插件的加载

每个节点都对插件编译后再在主节点 create extension，或者 patronictl -c postgresmq.yml query batman5 --command 'create extension pg_prewarm'在给集群每一个节点发送指令即可。

6 Watchdog 功能

6.1 适用场景:

- 主节点 patroni 进程被 kill
- 主节点 patroni 因内存资源超出而照成的崩溃或者是高负载系统下 patroni 被卡死这样的单点故障
- 网络故障

处理方式:

当遇到上述情景时，watch 会触发主节点系统重启，启动后用开机服务来自动开启 dcs 和 patroni,postgres。从库在主库 down 掉后提升为主，原主在重启完毕恢复后降级为备机。

6.2 Watchdog 配置步骤

安装（root 用户）:

```
yum install watchdog -y
```

开启并授权：（如果机器重启必须执行如下 2 步）（root 用户）

两个节点都需要执行，不然使用切换命令会认为备节点不是一个可以提升的节点

```
modprobe softdog

chown postgres /dev/watchdog
```

patroni 的 yaml 文件添加（普通用户）,更改完毕后重启 patroni 生效（kill 再开启）：

```
watchdog:

 mode: required # Allowed values: off, automatic, required

 device: /dev/watchdog

 safety_margin: 5
```

注意：主备都需进行以上操作，如果主库配置不正确或会导致出现如下信息

```
  File "./patroni.py", line 6, in <module>

    main()

  File "/home/postgres/patroni-1.4.3/patroni/__init__.py", line 176,
in main

    return patroni_main()

  File "/home/postgres/patroni-1.4.3/patroni/__init__.py", line 145,
in patroni_main

    patroni.run()

  File "/home/postgres/patroni-1.4.3/patroni/__init__.py", line 114,
in run

    logger.info(self.ha.run_cycle())

  File "/home/postgres/patroni-1.4.3/patroni/ha.py", line 1134, in
run_cycle

    info = self._run_cycle()

  File "/home/postgres/patroni-1.4.3/patroni/ha.py", line 1058, in
_run_cycle

    return self.post_bootstrap()

  File "/home/postgres/patroni-1.4.3/patroni/ha.py", line 978, in
post_bootstrap

    self.cancel_initialization()
```

```
  File "/home/postgres/patroni-1.4.3/patroni/ha.py", line 957, in
cancel_initialization

    raise PatroniException('Failed to bootstrap cluster')

patroni.exceptions.PatroniException: 'Failed to bootstrap cluster'
```

如果备库 watchdog 配置不正确在发生切换时会切换不了，有如下信息：

```
INFO: following a different leader because i am not the healthiest
node
```

如果 watchdog 配置成功，重新启动 patroni 时主库会出现如下信息：

```
INFO: Software Watchdog activated with 25 second timeout, timing
slack 15 seconds
```

6.3开机启动服务

自启动配置（root 用户）

```
chmod +x /etc/rc.d/rc.local

vi /etc/rc.d/rc.local
```

添加四行（root 用户）

```
/root/zookeeper-3.3.6/bin/zkServer.sh start

modprobe softdog

chown postgres /dev/watchdog

su - postgres -c "nohup /home/postgres/patroni-1.4.3/patroni.py
/home/postgres/patroni-1.4.3/postgresmq.yml  >
/home/postgres/logfile/patroni_log 2>&1 &"
```

7 添加节点

7.1添加 patroni 与数据库节点

- 1 在新的节点上安装 postgresql、patroni

- 2 在对应的现有节点上提供新节点的认证（pg_hba.conf）
- 3 复制 yml 文件到新的节点

```
scp postgresmq.yml postgres@192.168.1.140:/home/postgres/patroni-
1.4.3/
```

- 4 修改新节点的 yml 文件

```
name: postgresql2

tags:

    replicatefrom: postgresql0    ###选择级联从节点，默认是向主同步


restapi:

  listen: 192.168.1.140:8009

  connect_address: 192.168.1.140:8009


postgresql:

  listen: 0.0.0.0:5432

  connect_address: 192.168.1.140:5432

  data_dir: /home/postgres/pg10/data
```

- 5 开启 patroni

8 日志级别调整

该功能只有在 patroni1.3.4 及以后的版本才加入

patroni1.3.4 新增日志等级介绍:

PATRONI_LOGLEVEL - sets the general logging level #基本日志输出

PATRONI_REQUESTS_LOGLEVEL - sets the logging level for all HTTP requests e.g. Kubernetes API call #接口接收日志

```
修改这两个参数需要在patroni/_init_.py 中修改,默认等级是INFO、WARNING
loglevel = os.environ.get('PATRONI_LOGLEVEL', 'INFO')
requests_loglevel = os.environ.get('PATRONI_REQUESTS_LOGLEVEL',
'WARNING')
```

```
def patroni_main():

    logformat = os.environ.get('PATRONI_LOGFORMAT',
'%(asctime)s %(levelname)s: %(message)s')

    loglevel = os.environ.get('PATRONI_LOGLEVEL', 'INFO')

    requests_loglevel = os.environ.get('PATRONI_REQUESTS_LOGLEVEL',
'WARNING')

    logging.basicConfig(format=logformat, level=loglevel)

    logging.getLogger('requests').setLevel(requests_loglevel)


    patroni = Patroni()

    try:

        patroni.run()

    except KeyboardInterrupt:

        pass

    finally:

        patroni.shutdown()
```

Python 中的日志等级:

CRITICAL

ERROR

WARNING

INFO

DEBUG

NOTSET

修改完毕后重启集群生效:

重启 patroni 集群，在任意 patroni 节点执行:

```
cd /home/postgres/patroni-1.4.3

./patronictl.py -c postgresmq.yml restart batman5
```

9 故障

9.1 故障检查流程

```
故障检测基本思路

查看patroni日志报出错误代码--|网络问题----|是否是数据库验证问题
                                |              |是否是网络单点故障或者多点故障
                                |              |检查dcs日志，dcs各节点状态
                                |
                                |切换问题----|检查dcs中leader键值是否改变
                                |              |检查dcs中members下的各个节点的信息
                                |              |检查数据库日志
                                |
                                |
                                |数据库问题---|检查数据库日志
```

9.2 检查项

- Patroni 日志

```
ll /home/postgres/logfile/patroni_log
```

- Patroni 集群状态

```
[postgres@test3 patroni-1.4.2]$ ./patronictl.py  -c postgresmq.yml
list
+---------+-------------+---------------+--------+---------+---------
--+
| Cluster |    Member   |      Host     |  Role  |  State  | Lag in MB |
+---------+-------------+---------------+--------+---------+---------
--+
| batman5 | postgresql0 | 192.168.1.143 | Leader | running |
|
| batman5 | postgresql1 | 192.168.1.142 |        | running |           |
```

```
+---------+-------------+---------------+--------+---------+---------
--+
```

- Dcs 日志

```
ll /home/postgres/logfile/patroni_log/version-2
```

- Dcs 集群状态

```
/root/zookeeper-3.3.6/bin/status
```

- 数据库日志

```
ll /home/postgres/flyingdb-v3-logical/data/log/*.log
```