Investigate High Availability performance problems

adyen

engineered
for ambition

EDB

HIGH AVAILABLE CONFIGURATIONS ARE VERY COMMON FOR POSTGRESQL. BUT HOW DO YOU INVESTIGATE PERFORMANCE PROBLEMS WHEN THE STANDBY CAN'T KEEP UP?

adyen

engineered
for ambition

EDB

Boriss Mejias

Senior solution architect

EDB

adyen

engineered
for ambition

EDB

Derk van Veen

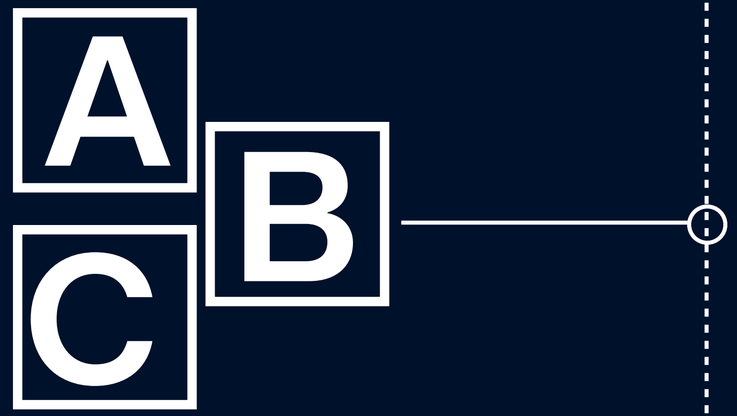
Data science specialist

Adyen

adyen

engineered
for ambition

Agenda



Concepts

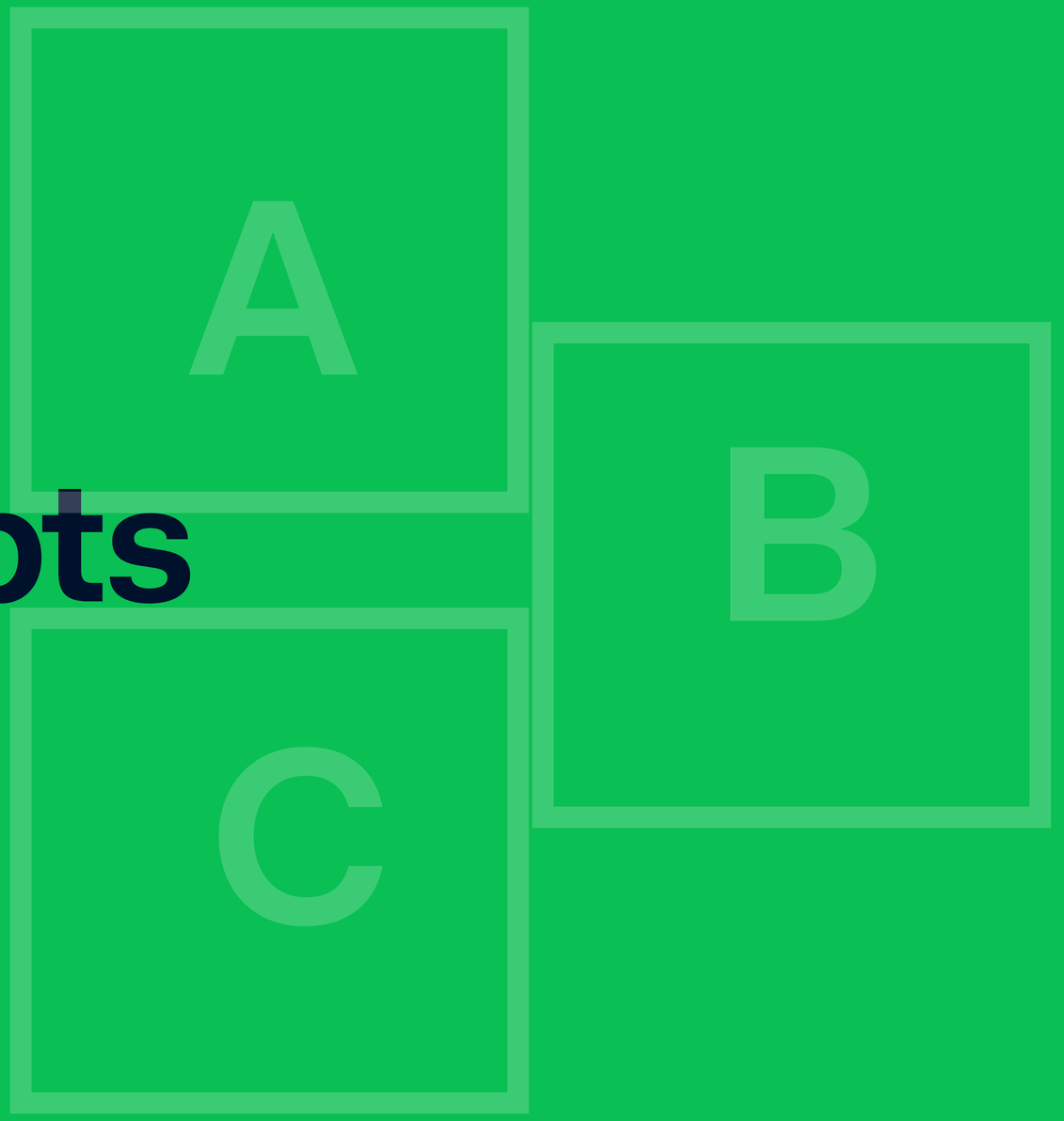
Problem

Investigation

Solution

Lessons learned

Concepts

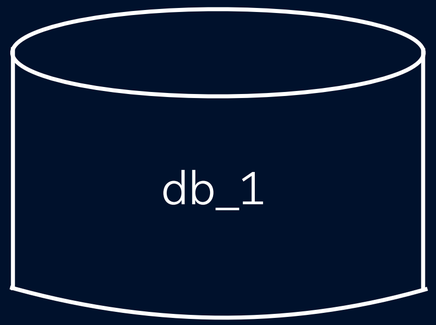


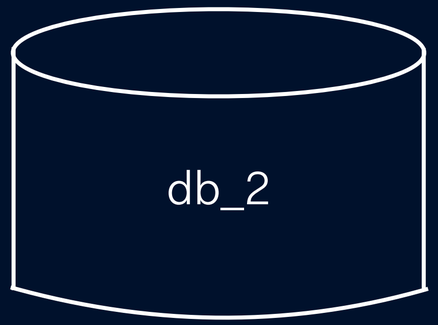
Cluster

Wal sender

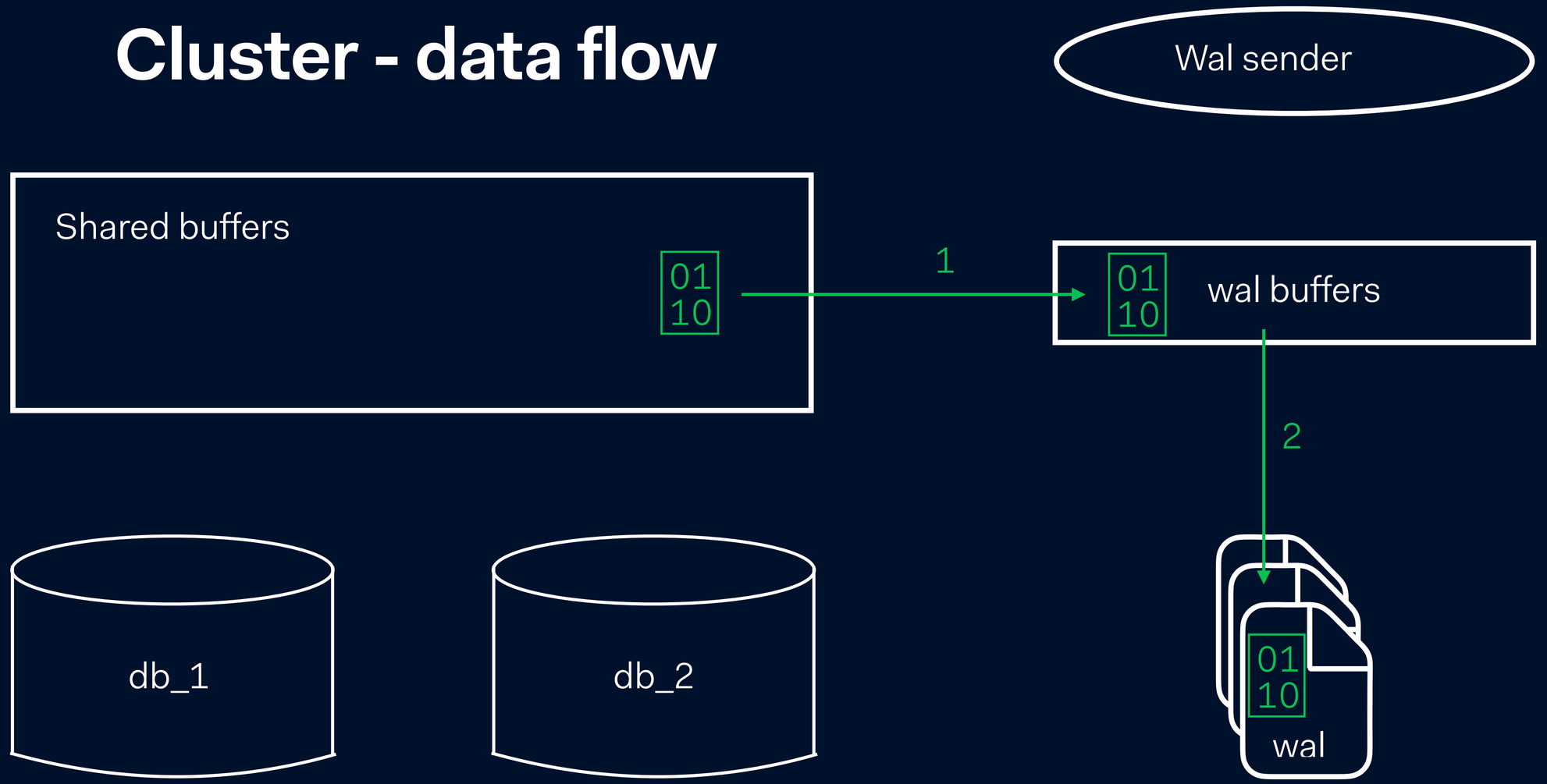
Shared buffers

wal buffers

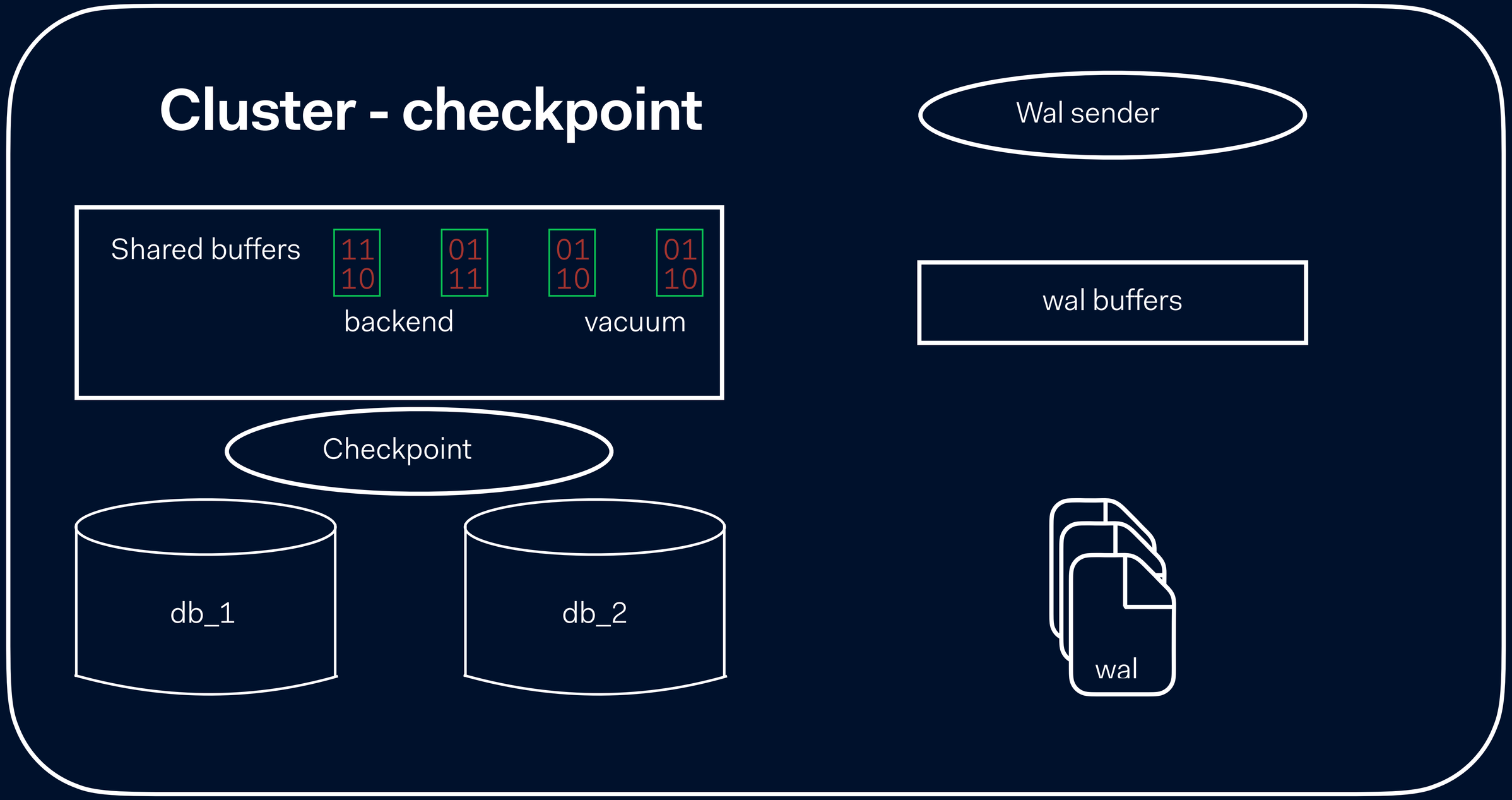




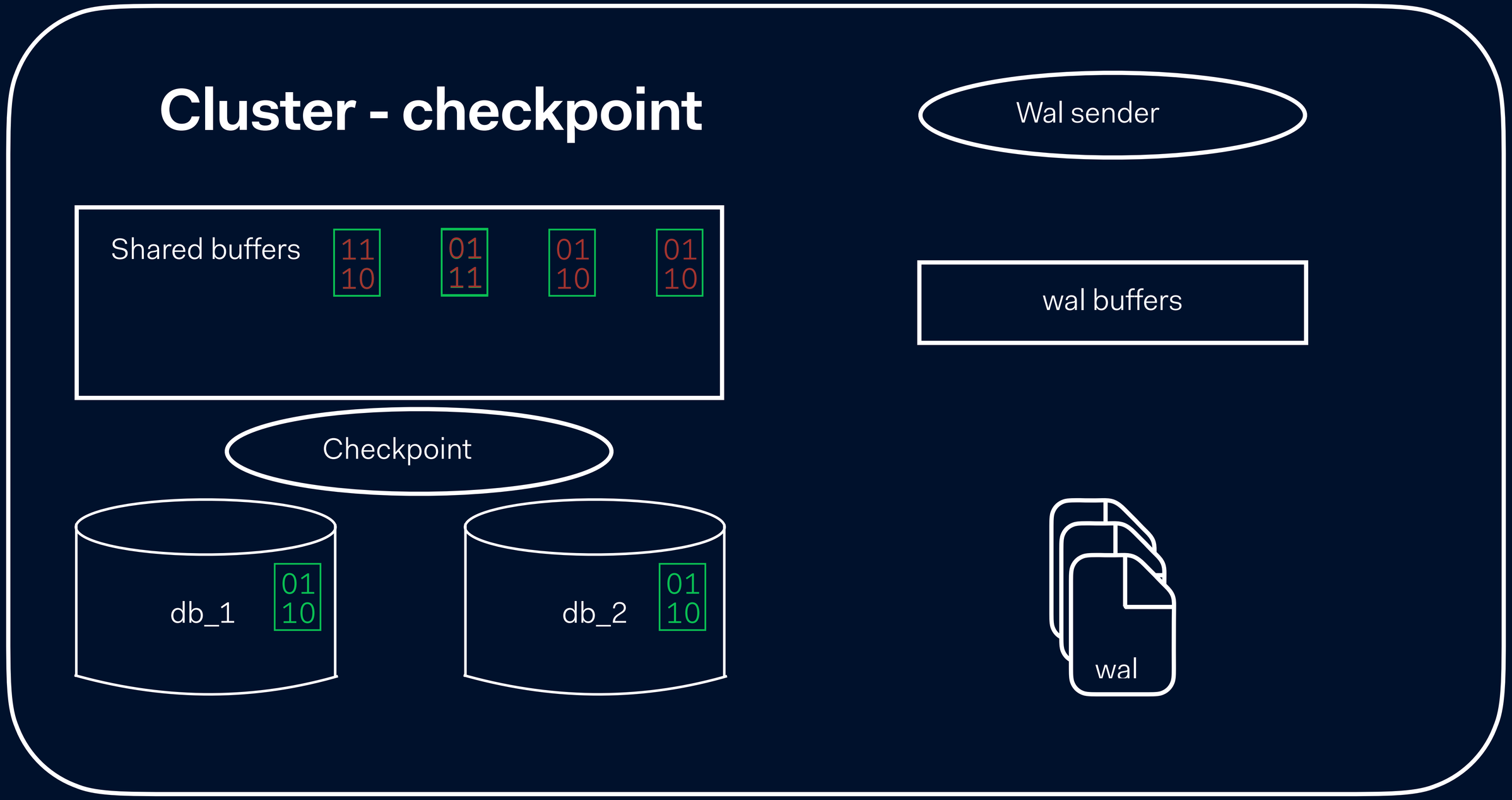
Cluster - data flow



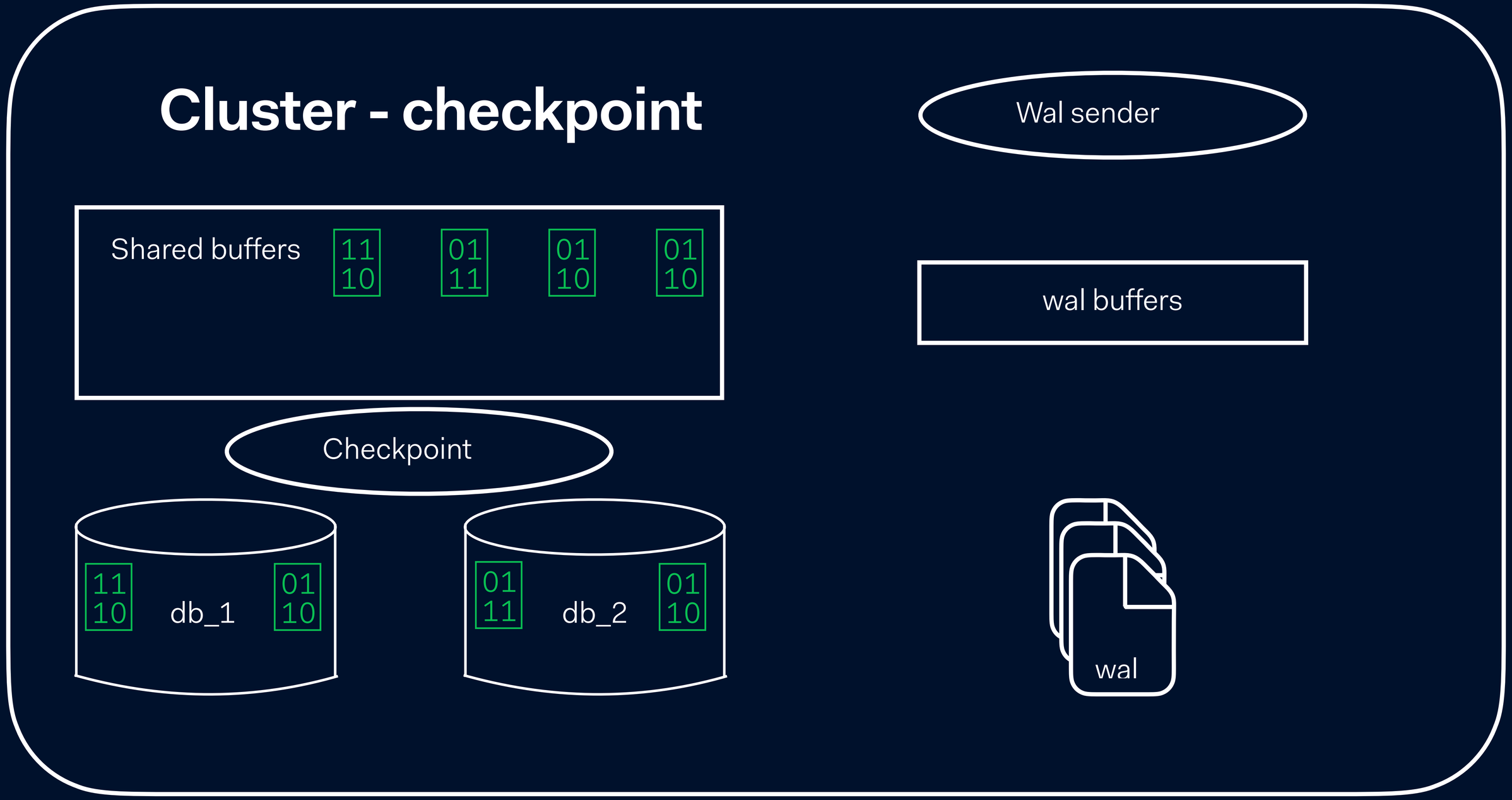
Cluster - checkpoint



Cluster - checkpoint



Cluster - checkpoint

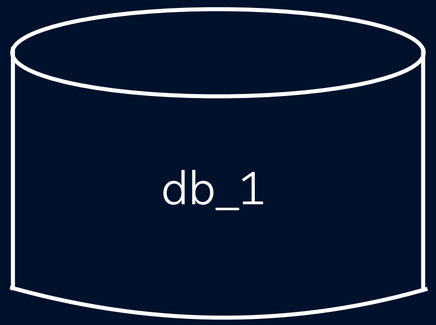


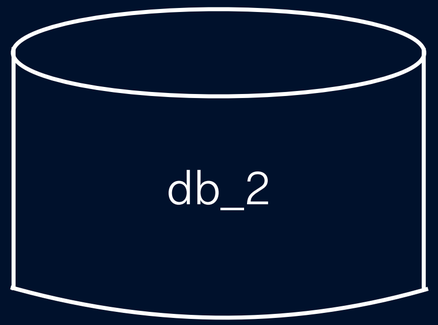
Cluster

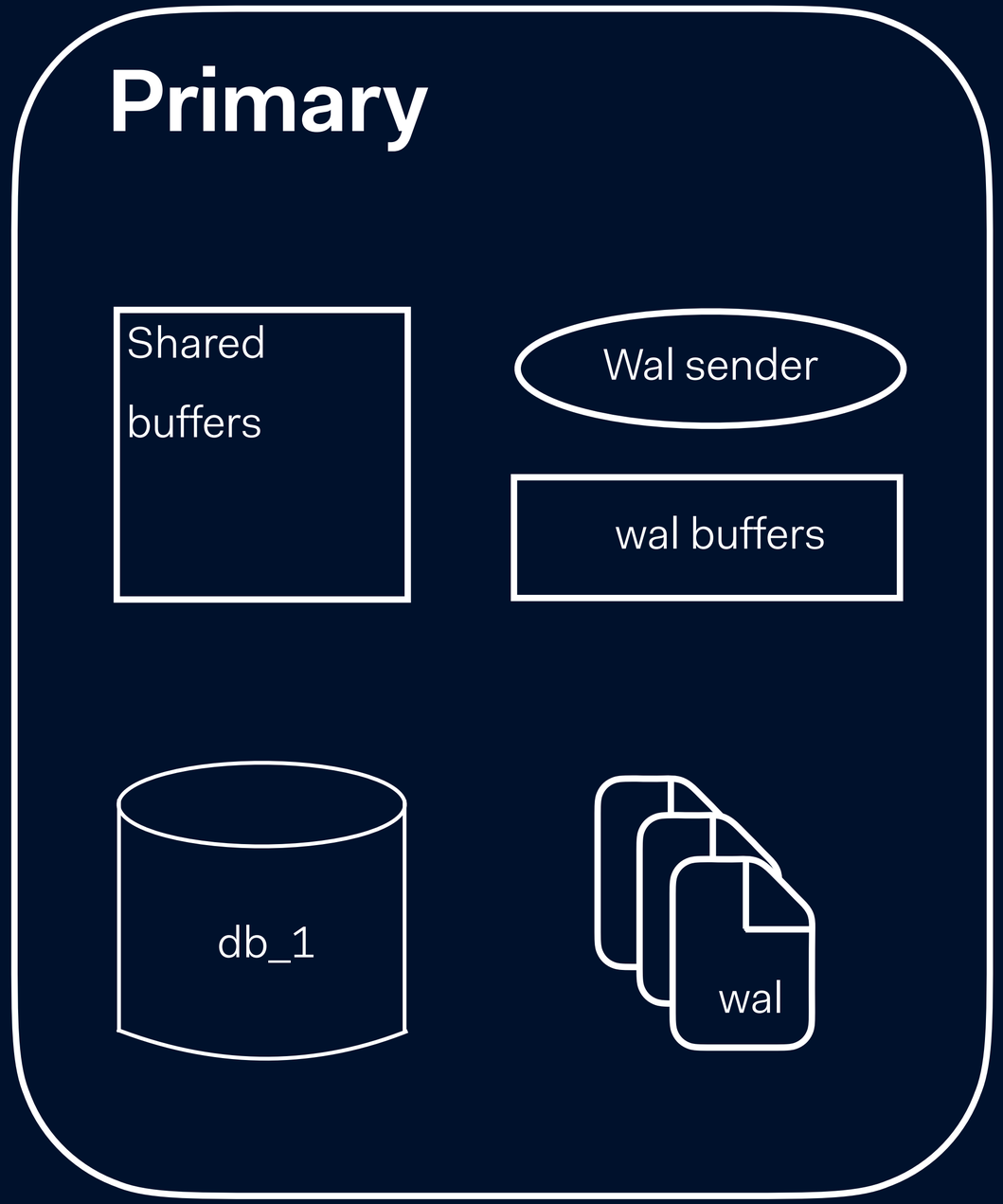
Wal sender

Shared buffers

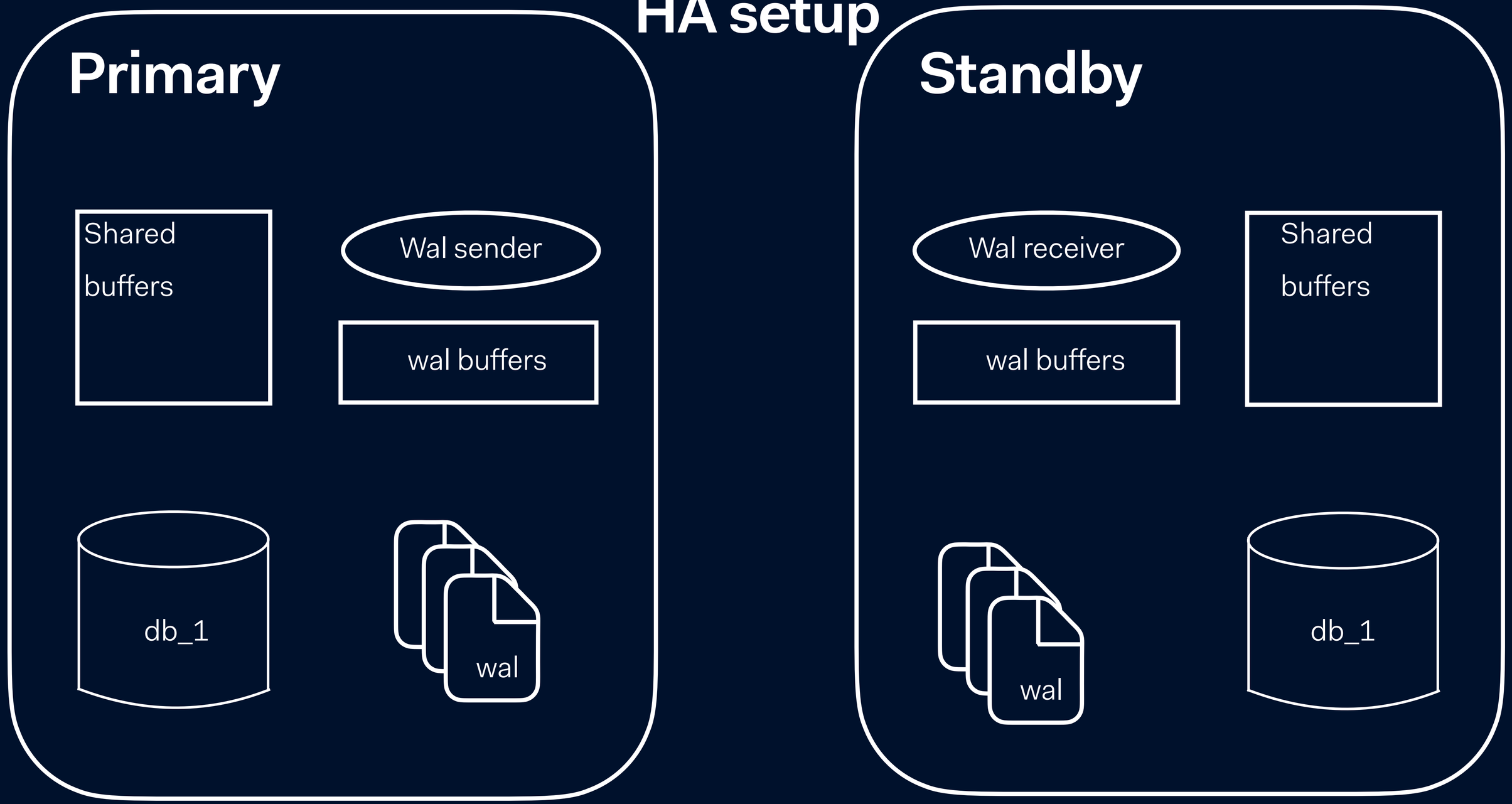
wal buffers



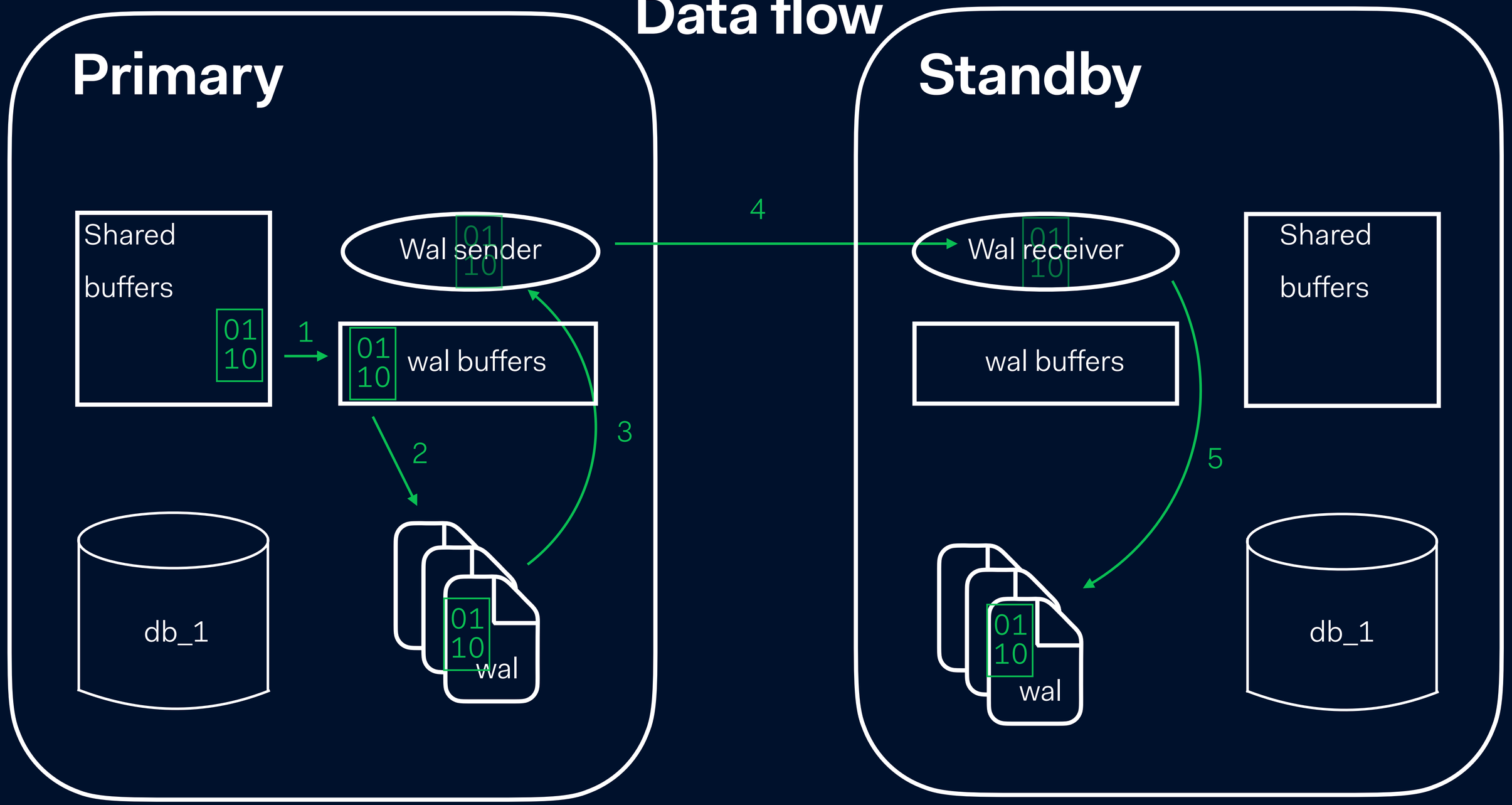


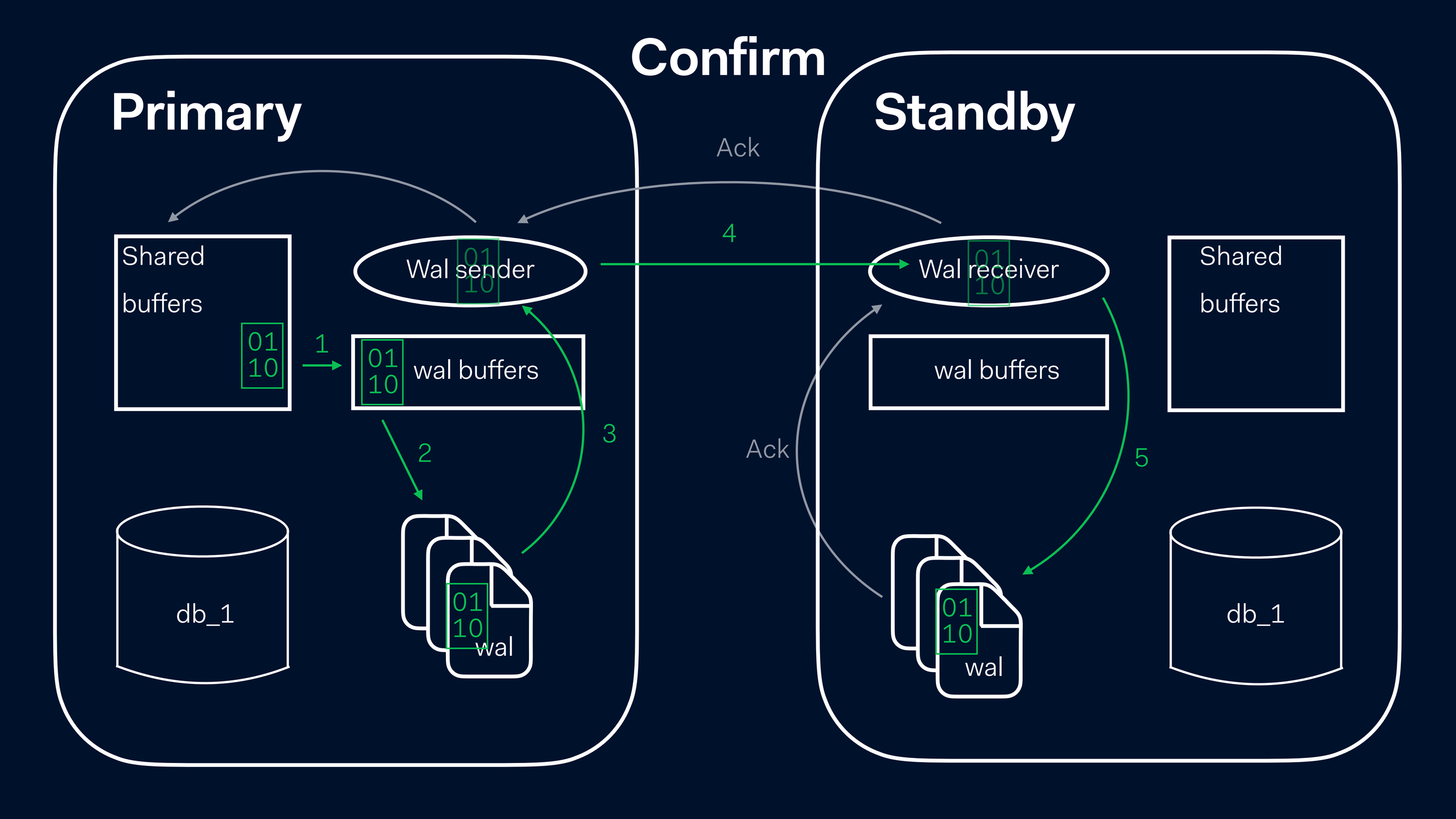

Primary
Shared
buffers
Wal sender
wal buffers
db_1
wal

HA setup

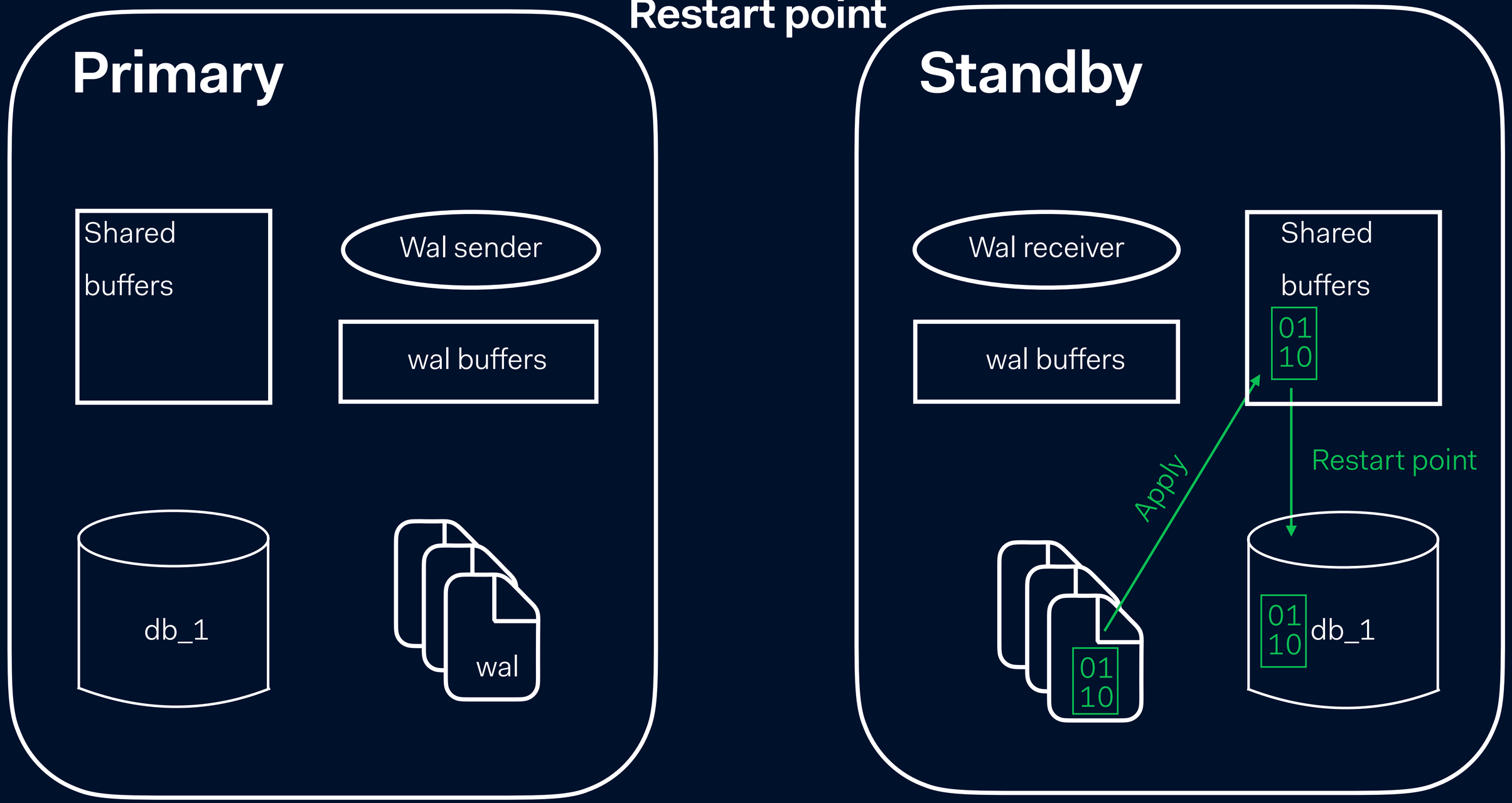

Primary
Shared buffers
Wal sender
wal buffers
db_1
wal
Standby
Wal receiver
wal buffers
Shared buffers
wal
db_1

Data flow


Primary
Standby
Shared buffers
01 10
1
wal buffers
01 10
Wal sender
01 10
2
3
4
Wal receiver
01 10
wal buffers
5
Shared buffers
db_1
01 10
wal
01 10
wal
db_1

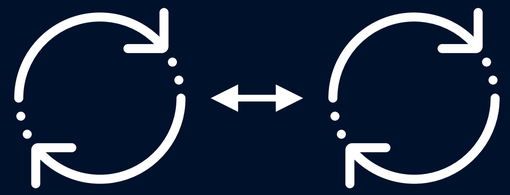

Confirm
Primary
Standby
Ack
Shared
buffers
01
10
1
Wal sender
01
10
4
Wal receiver
01
10
Shared
buffers
01
10
wal buffers
wal buffers
2
3
Ack
5
db_1
01
10
wal
01
10
wal
db_1

Restart point

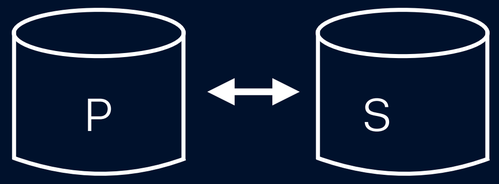


Wait events

Wait events

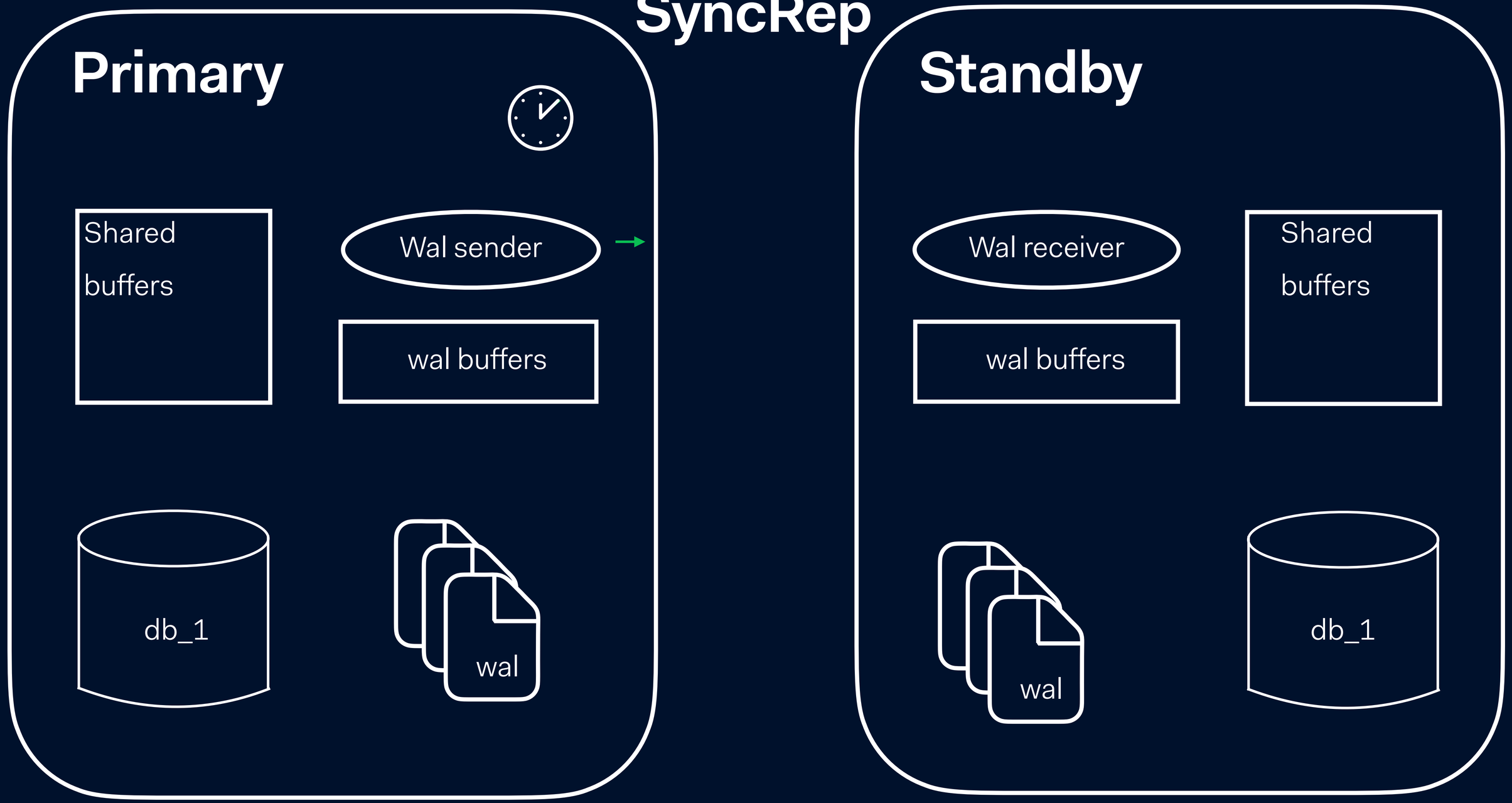


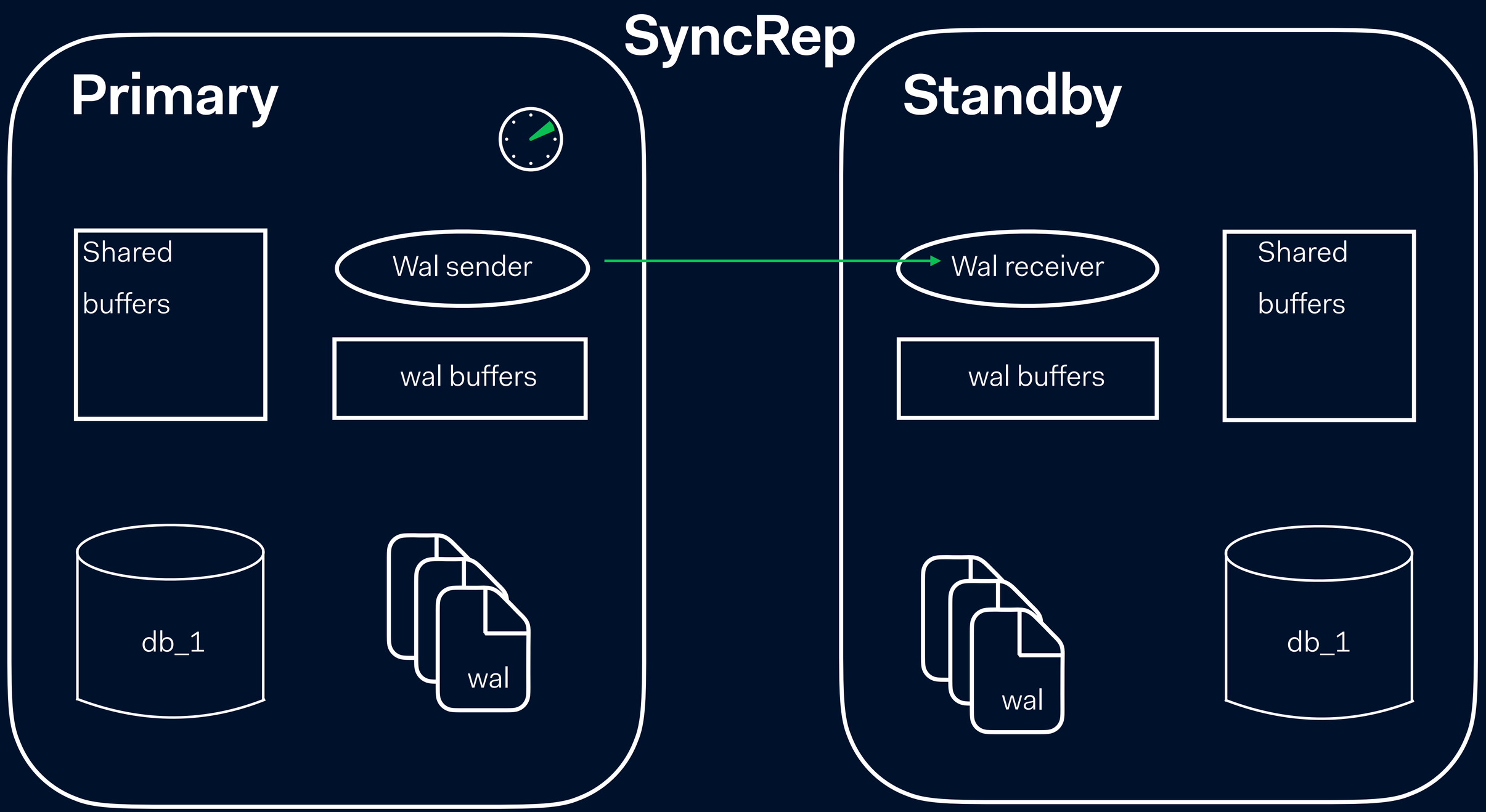
IPC wait events

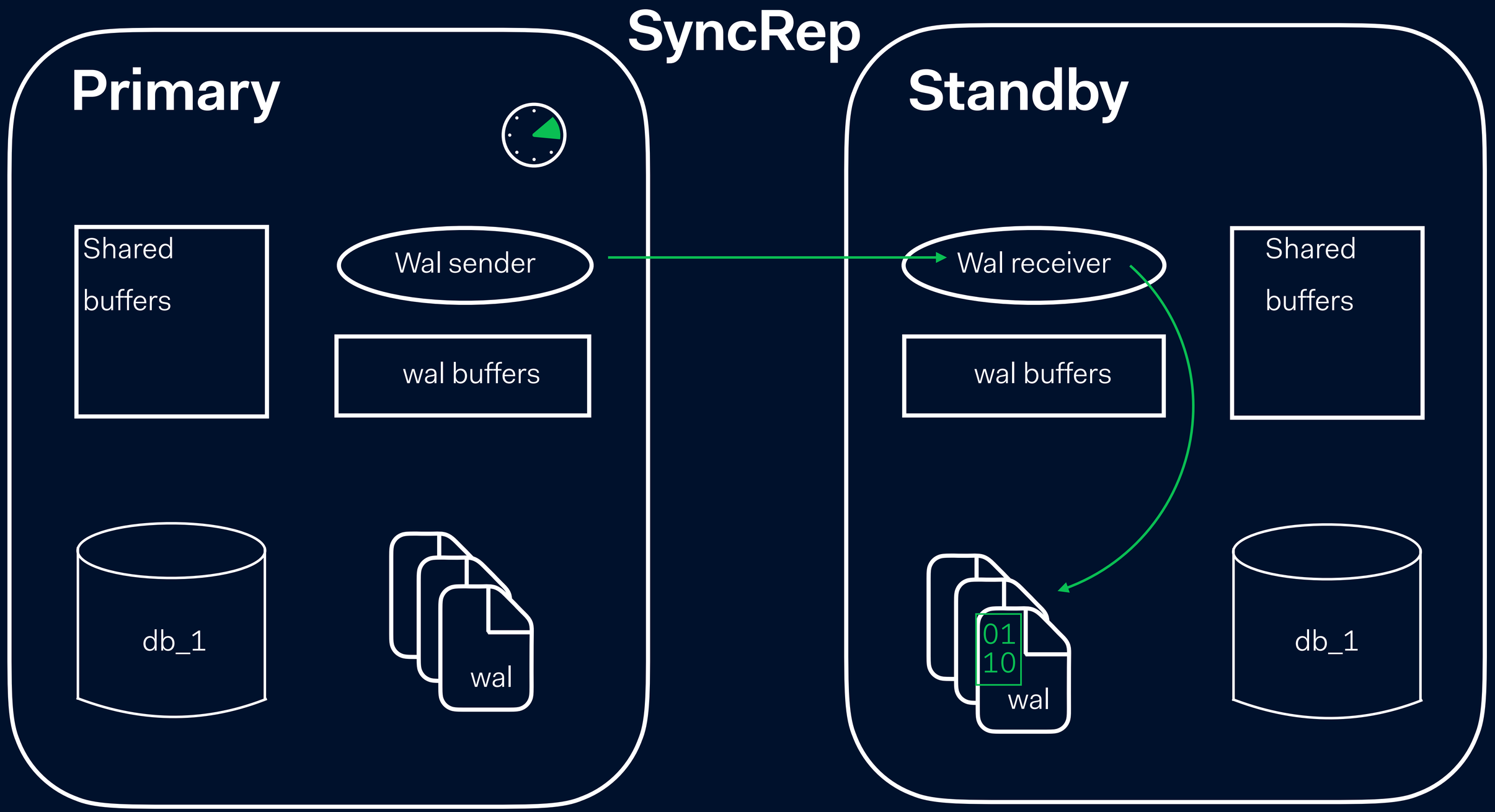


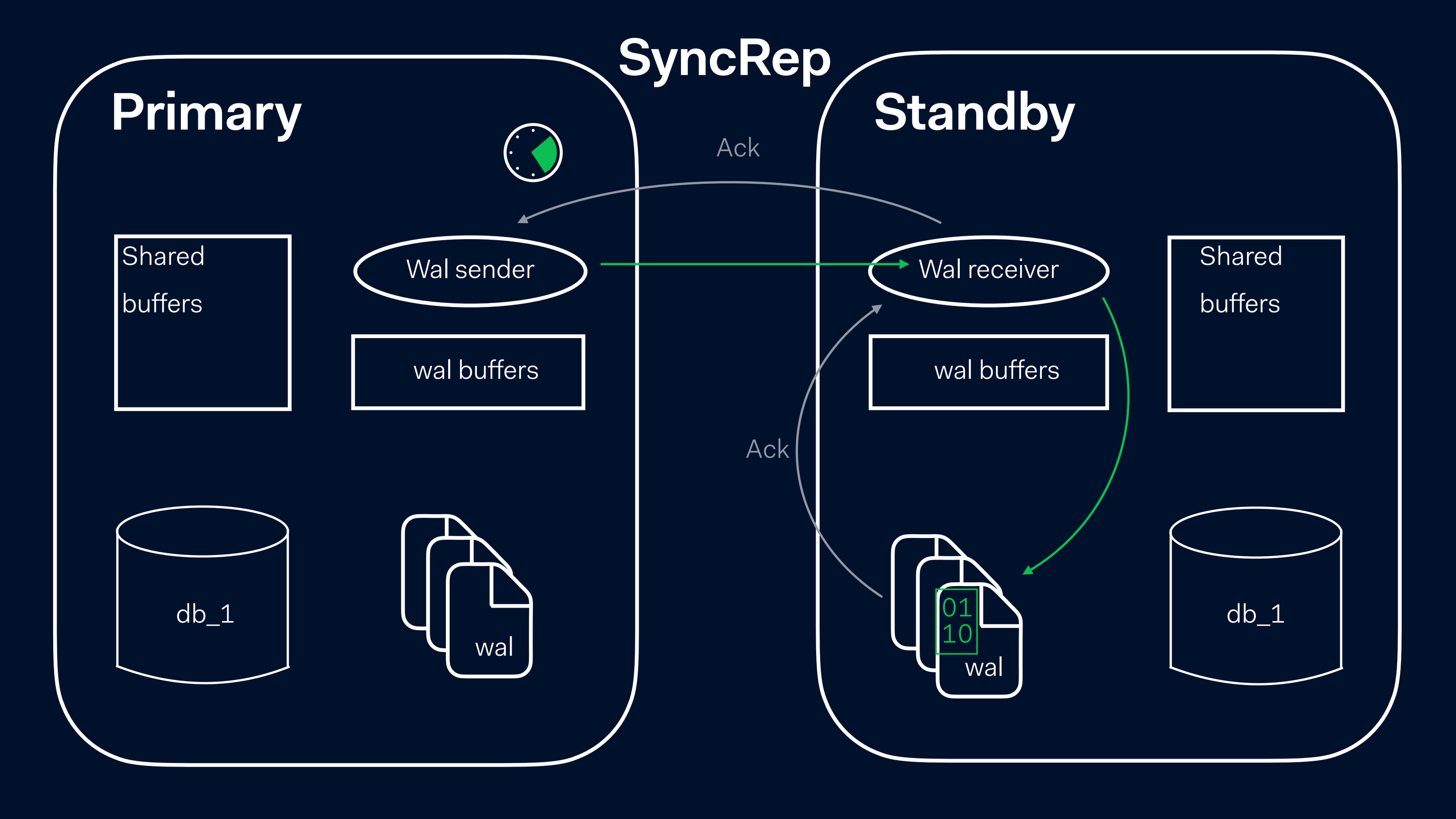
Syncrep wait events

SyncRep


Primary
Shared buffers
Wal sender
wal buffers
db_1
wal
Standby
Wal receiver
wal buffers
Shared buffers
wal
db_1


SyncRep
Primary
Standby
Shared buffers
Wal sender
Wal receiver
Shared buffers
wal buffers
wal buffers
db_1
wal
wal
db_1


SyncRep
Primary
Standby
Shared
buffers
Wal sender
wal buffers
Wal receiver
wal buffers
Shared
buffers
db_1
wal
01
10
wal
db_1

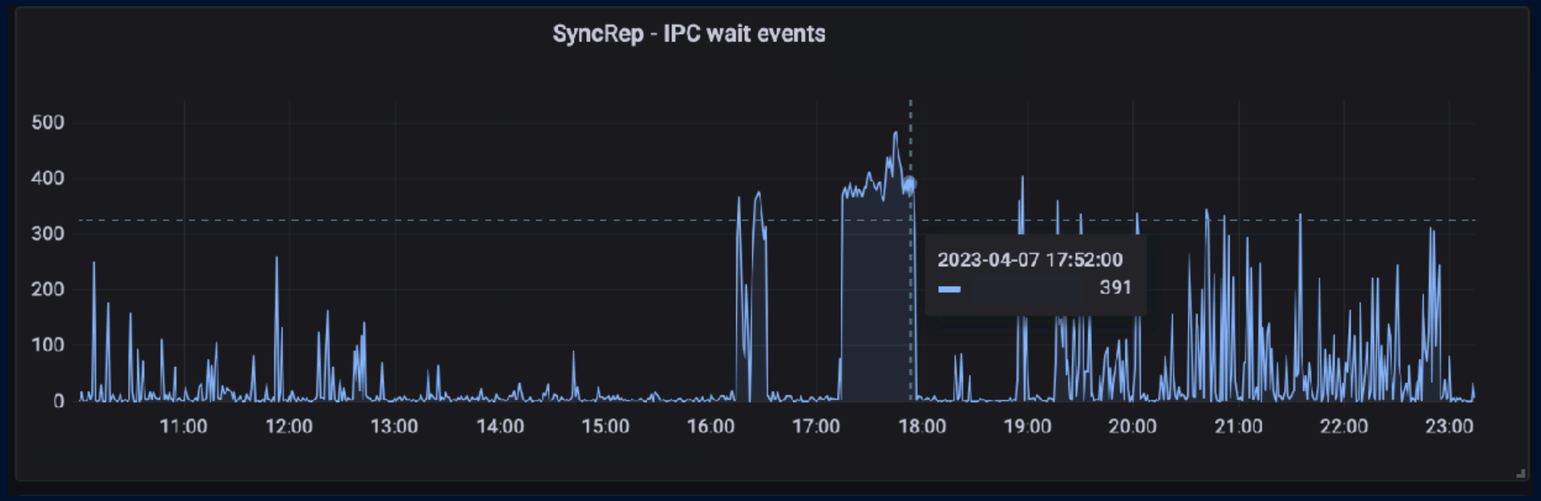

SyncRep
Primary
Standby
Ack
Shared buffers
Wal sender
Wal receiver
Shared buffers
wal buffers
wal buffers
Ack
db_1
wal
01
10
wal
db_1

Problem

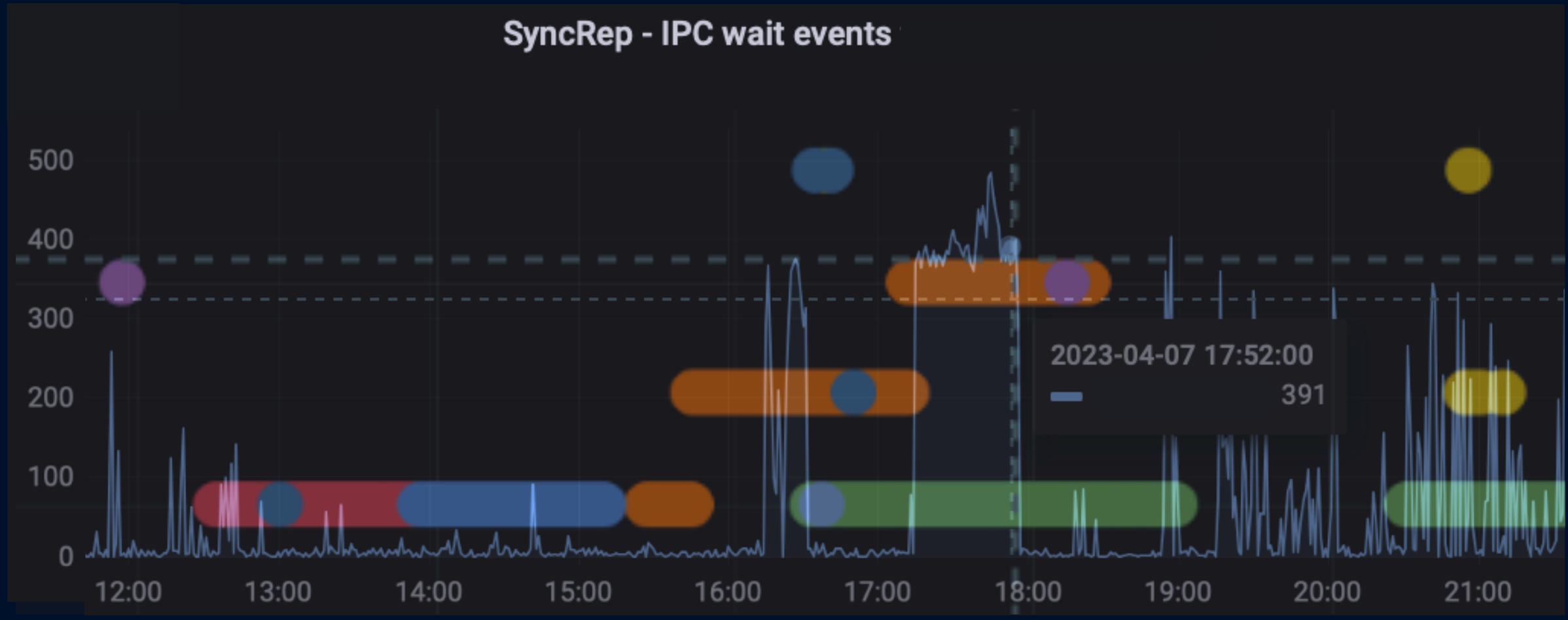
Job is delayed

Job is delayed

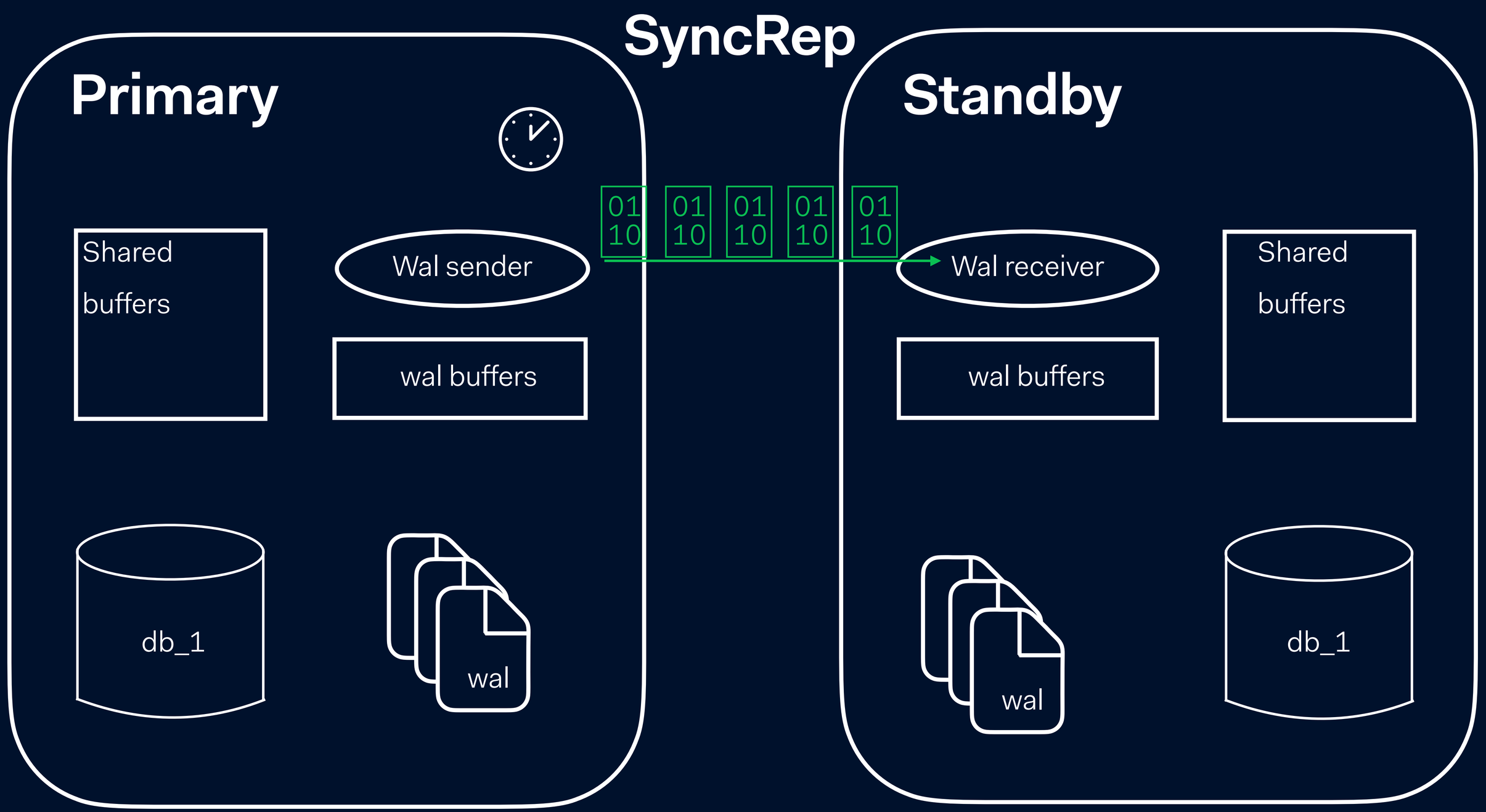
Job is delayed



Job is delayed

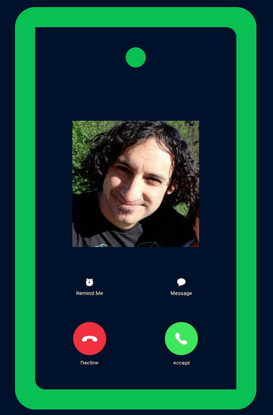


Mitigation: make vacuum slower


SyncRep
Primary
Standby
Shared buffers
Wal sender
wal buffers
01 10
01 10
01 10
01 10
01 10
Wal receiver
wal buffers
Shared buffers
db_1
wal
wal
db_1


Remind Me
Message
Decline
Accept

Problem description

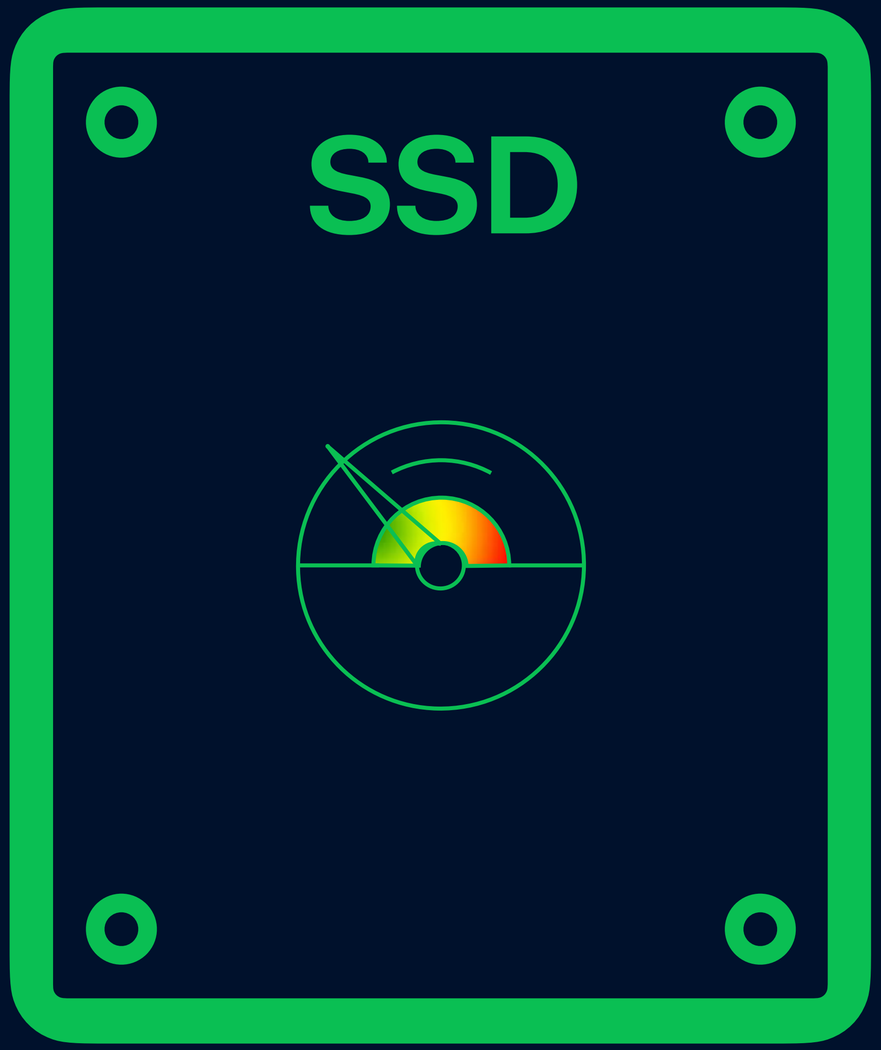


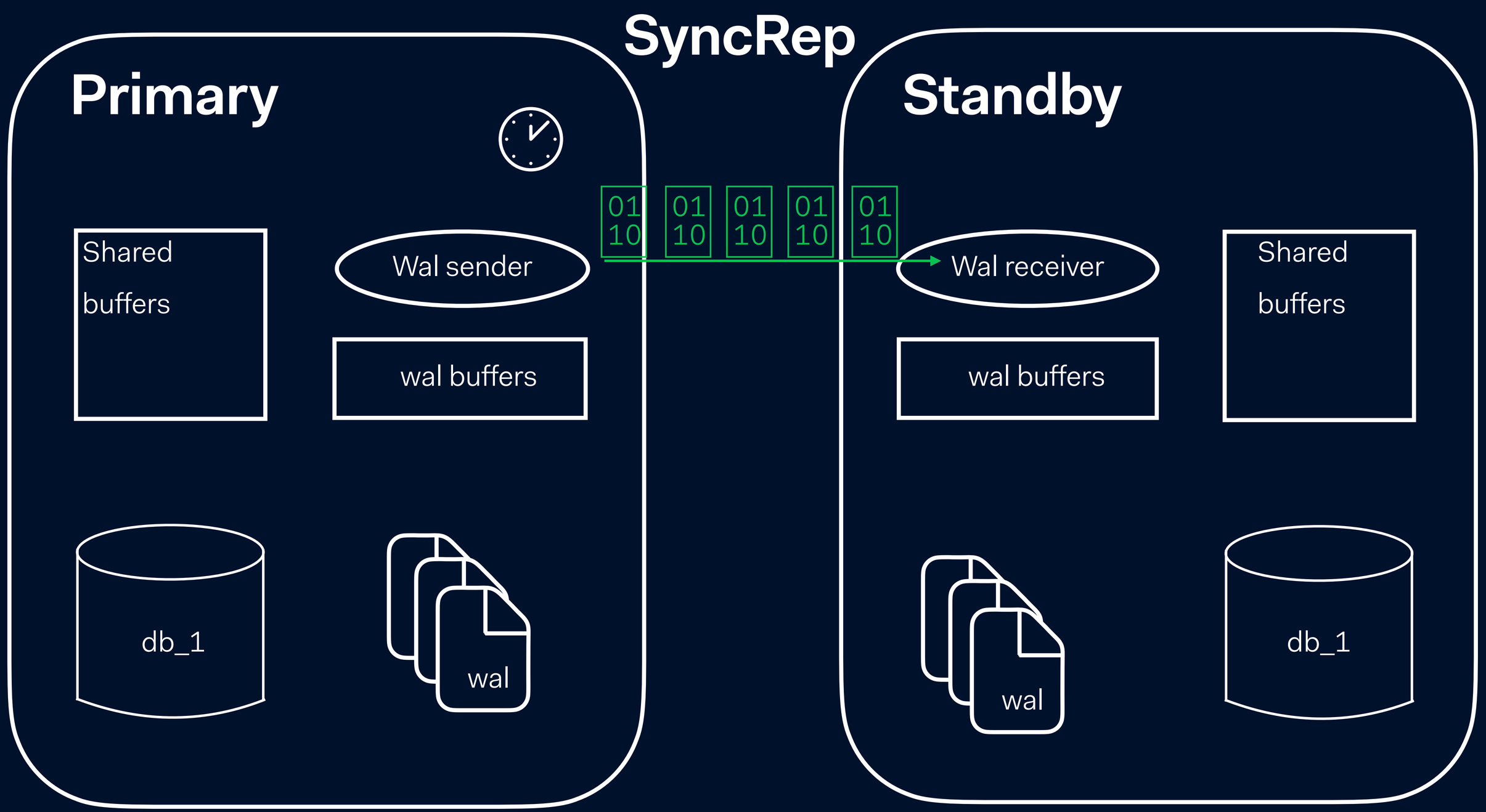
- We have significant job delays related to SyncRep
- Strong correlation with vacuum on large tables
- Increase in SyncRep directly after checkpoint starts
- While SyncRep piles up, transaction rate goes down
- Terminate vacuum helps
- Delay is equal for both standbys

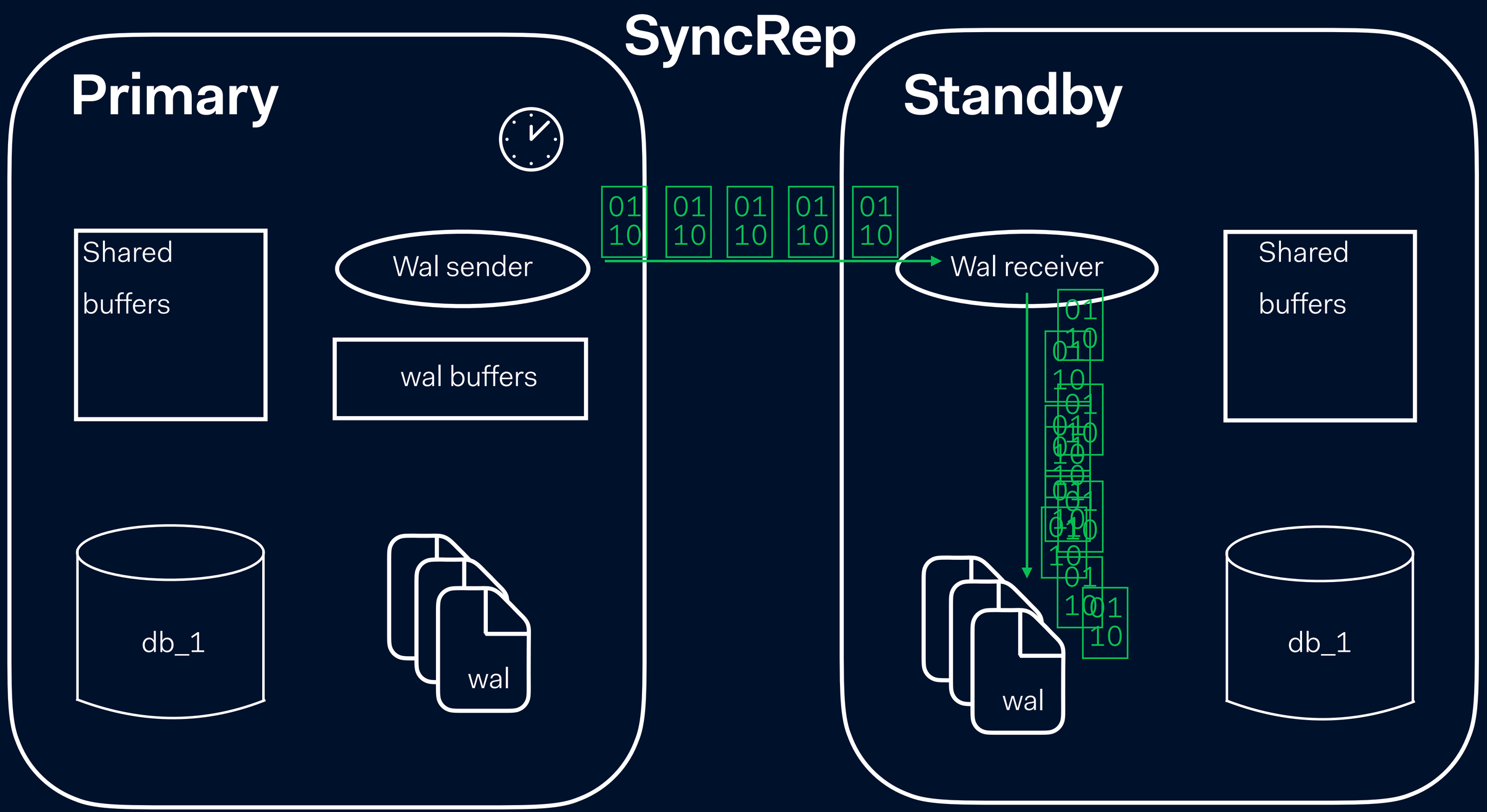
Problem investigation

Lets blame the firewall/ network

Lets blame the storage


SSD


SyncRep
Primary
Standby
Shared buffers
Wal sender
wal buffers
01 10
01 10
01 10
01 10
01 10
Wal receiver
wal buffers
Shared buffers
db_1
wal
wal
db_1


SyncRep
Primary
Standby
Shared buffers
Wal sender
wal buffers
db_1
wal
Wal receiver
Shared buffers
wal
db_1

If everything is sequential

- Tcp
- Wal lsn
- OS writes
- Commits

What is the theoretical limit?

“Standby can’t keep up with the amount of wal writes”

Lets start some debug

- SAR report
- IOStat
- SOSReport
- ss
- perf
- ioping
- pidstat
- Netstat
- iotop

Lets start some debug

- SAR report
- IOStat
- SOSReport
- ss
- perf
- ioping
- pidstat
- netstat
- iotop

ss

"A socket is one endpoint of a two-way communication link between two programs running on the network"

Socket Statistics

Round Trip Time	average round trip time with mean deviation
Recv-Q	receive buffer, application needs to retreive it
Send-Q	send buffer; not sent or sent but not ACKed
lastsnd	How long since last package sent

Socket Statistics

rtt:5.06/0.428 send 254.1Mbps

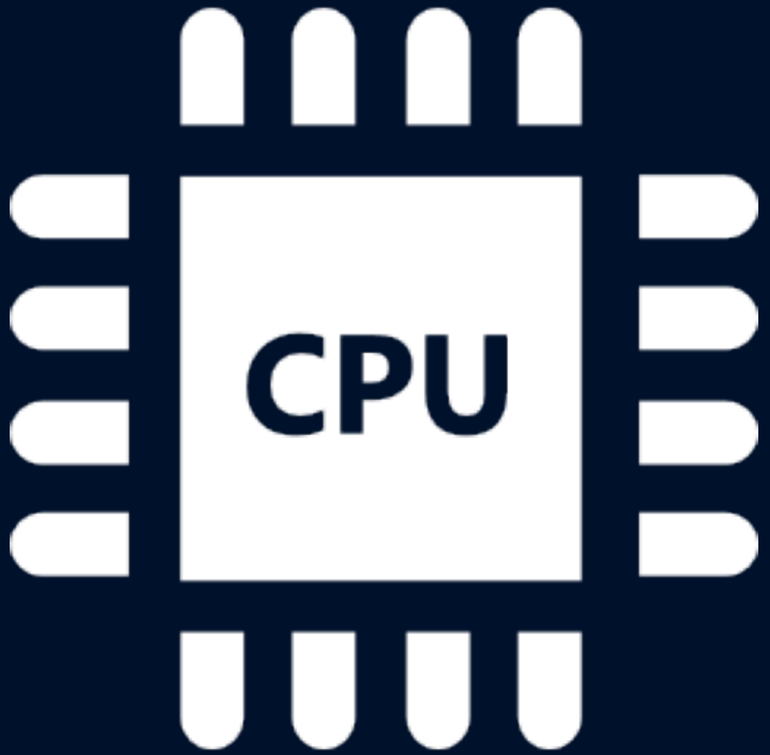
rtt:2.657/2.413 send 457.8Mbps

Socket Statistics

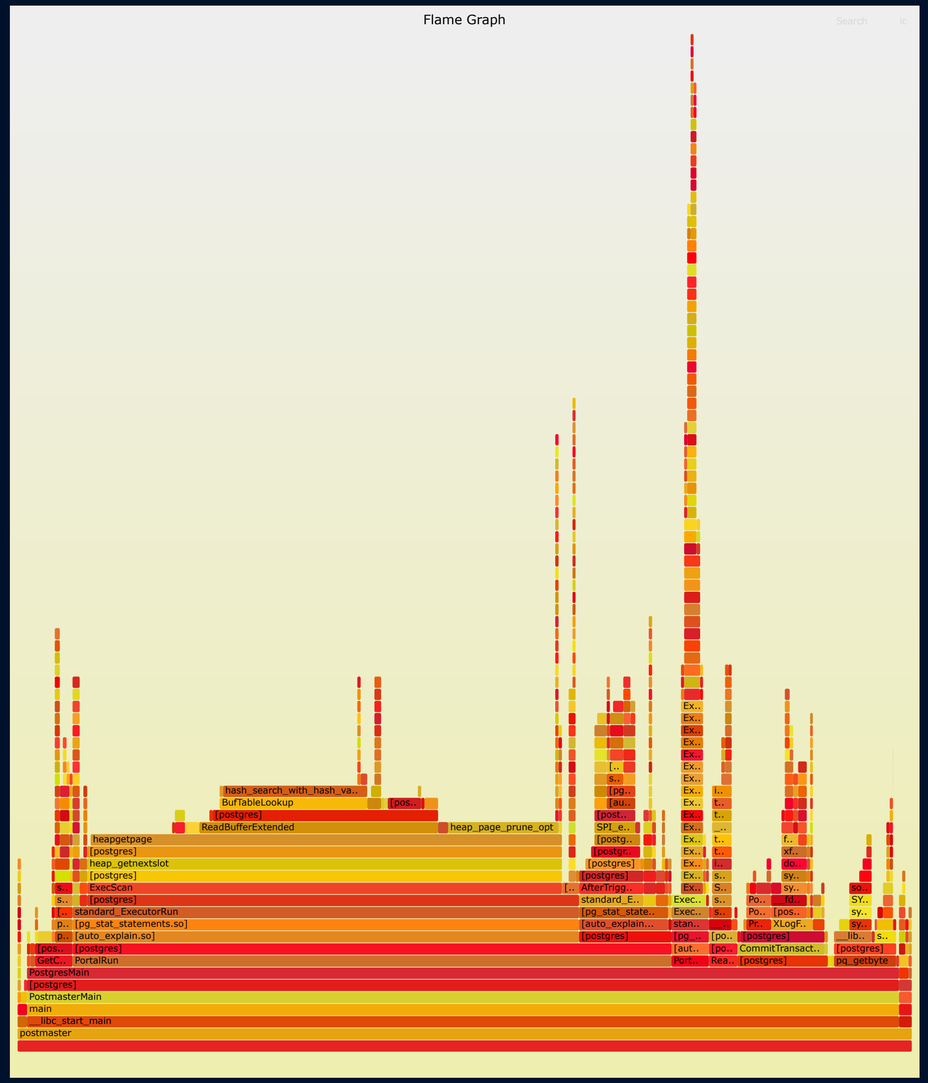
```
skmem:(
  r6142208,
  rb6291456,
  t0,
  tb332800,
  f161536,
  w0,
  o0,
  bl0,
  d580)
```

97.6% rec buffer full

Perf



Perf flame-graph



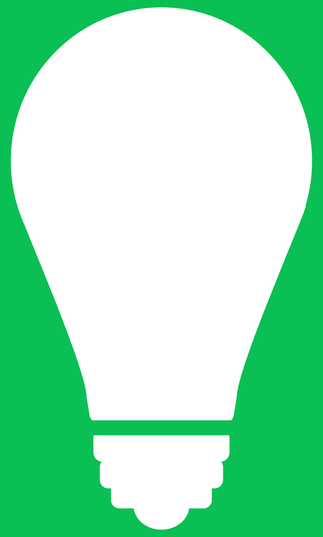

hash_search_with_hash_va..
BufTableLookup
[pos..
[postgres]
ReadBufferExtended
heap_page_prune_opt
heapgetpage
[postgres]
heap_getnextslot
[postgres]
s..
ExecScan
[..
s..
[postgres]
[..
standard_ExecutorRun
p..
[pg_stat_statements.so]
p..
[auto_explain.so]
[pos..
[postgres]
GetC..
PortalRun
PostgresMain
[postgres]
PostmasterMain
main
__libc_start_main
postmaster
[..
s..
[pg..
[au..
[post..
SPI_e..
[postg..
[postgr..
[postgres]
AfterTrigg..
standard_E..
[pg_stat_state..
[auto_explain...
[postgres]
Ex..
Ex..
Ex..
Ex..
Ex..
Ex..
Ex..
Ex..
Ex..
Ex..
Ex..
Ex..
Ex..
Exec..
Exec..
stan..
[pg_..
[aut..
Port..
i..
t..
t..
_..
t..
t..
i..
s..
S..
s..
s..
__..
[po..
[po..
Rea..
Po..
__fd..
Po..
[pos..
Pr..
XLogF..
[postgres]
CommitTransact..
[postgres]
f..
xf..
do..
sy..
sy..
sy..
sy..
so..
SY..
sy..
__lib..
s..
[postgres]
pq_getbyte

SS & Perf output

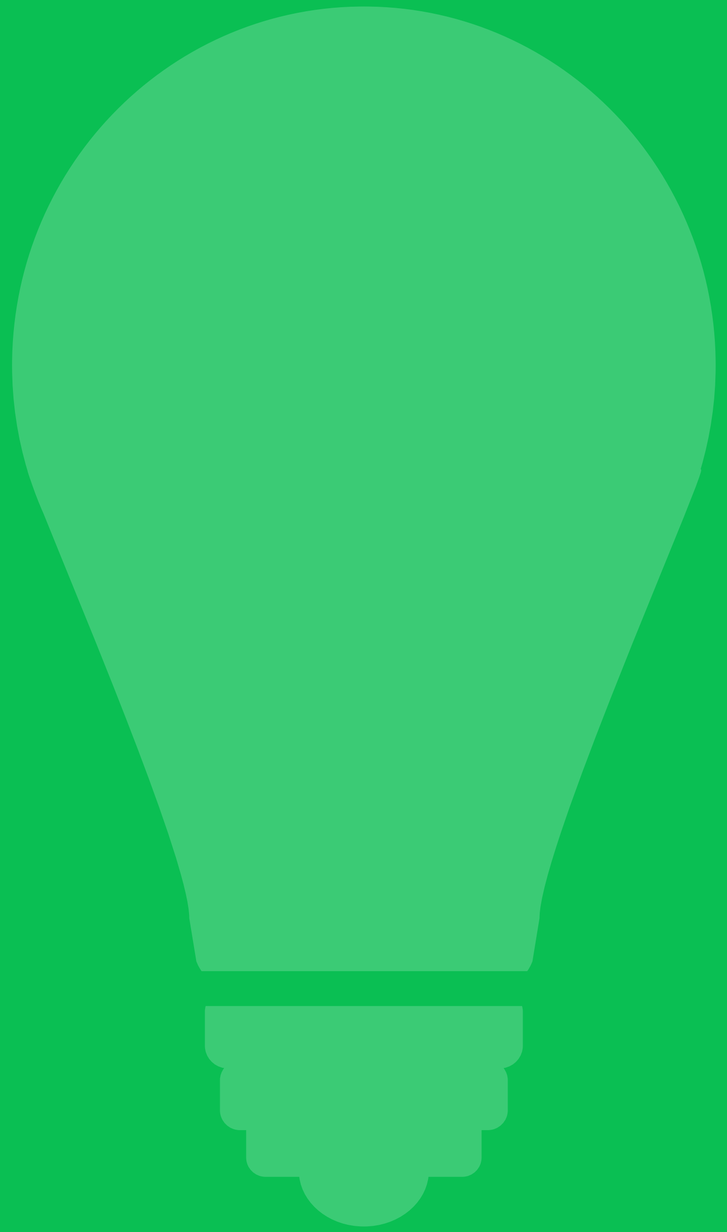
TIME	TCP_RECVQ	TCP_RBUFPCT	TCP_RTT	TCP_SENDRATE	TOPSYSCALL
21:00:32	899	0.0%	0.695/0.058	50.0Mbps	13.74%--sys_fdatasync
21:02:04	0	0.0%	0.877/0.19	39.6Mbps	13.43%--sys_fdatasync
21:03:37	20896	0.7%	0.823/0.214	42.2Mbps	12.80%--sys_fdatasync
21:05:09	0	0.0%	0.765/0.143	45.4Mbps	14.50%--sys_fdatasync
21:06:41	0	0.0%	0.777/0.083	44.7Mbps	13.53%--sys_fdatasync
21:08:13	0	0.0%	0.804/0.087	43.2Mbps	14.01%--sys_fdatasync
21:09:45	139	0.0%	0.808/0.133	43.0Mbps	13.38%--sys_fdatasync
21:11:19	0	0.0%	0.79/0.11	44.0Mbps	11.99%--sys_fdatasync
21:12:51	0	0.0%	0.702/0.109	49.5Mbps	13.87%--sys_fdatasync
21:14:23	1211	0.0%	0.732/0.105	47.5Mbps	12.21%--sys_fdatasync
21:15:53	125633	4.0%	0.86/0.121	40.4Mbps	36.60%--sys_pwrite64
21:17:24	1379822	44.6%	0.835/0.175	41.6Mbps	34.97%--sys_pwrite64
21:18:55	683034	21.9%	0.791/0.122	43.9Mbps	35.22%--sys_pwrite64
21:20:25	1790854	57.3%	0.673/0.082	51.6Mbps	34.73%--sys_pwrite64
21:21:56	3288552	99.5%	1.361/0.406	25.5Mbps	34.99%--sys_pwrite64
21:23:26	716453	23.0%	0.679/0.16	51.2Mbps	36.18%--sys_pwrite64
21:24:57	500137	16.1%	0.777/0.134	44.7Mbps	34.08%--sys_pwrite64
21:26:28	365872	11.4%	0.81/0.21	42.9Mbps	25.82%--sys_pwrite64
21:27:59	4131	0.1%	0.628/0.118	55.3Mbps	26.19%--sys_pwrite64
21:29:30	838693	26.7%	0.674/0.062	51.6Mbps	27.37%--sys_pwrite64
21:31:00	115372	3.6%	0.846/0.173	41.1Mbps	35.10%--sys_pwrite64
21:32:31	139553	4.4%	0.719/0.164	48.3Mbps	35.50%--sys_pwrite64
21:34:03	0	0.0%	0.81/0.182	42.9Mbps	13.52%--sys_fdatasync
21:35:35	6803	0.2%	0.637/0.086	54.6Mbps	14.31%--sys_fdatasync

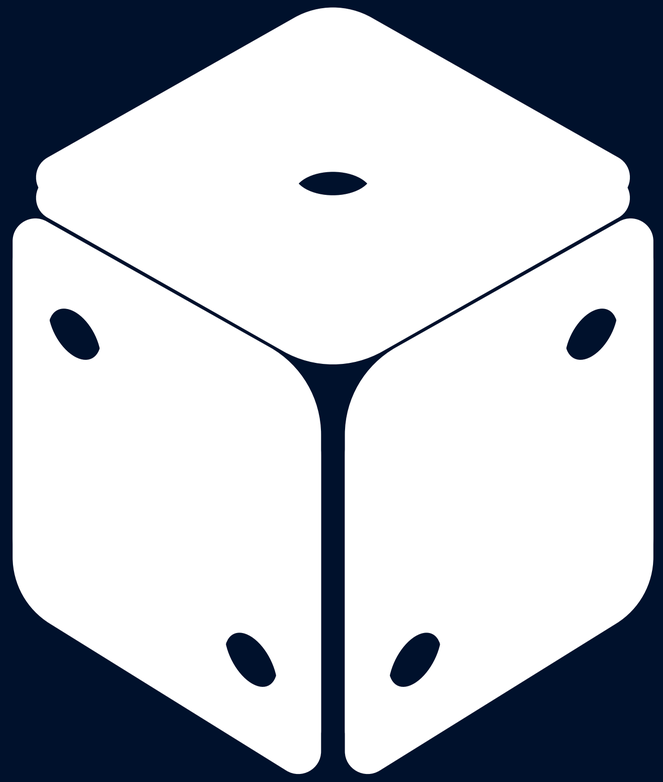
TIME	TCP_RECVQ	TCP_RBUFPCT	TCP_RTT	TCP_SENDRATE	TOPSYSCALL
21:05:09	0	0.0%	0.765/0.143	45.4Mbps	14.50%--sys_fdatasync
21:06:41	0	0.0%	0.777/0.083	44.7Mbps	13.53%--sys_fdatasync
21:08:13	0	0.0%	0.804/0.087	43.2Mbps	14.01%--sys_fdatasync
21:09:45	139	0.0%	0.808/0.133	43.0Mbps	13.38%--sys_fdatasync
21:11:19	0	0.0%	0.79/0.11	44.0Mbps	11.99%--sys_fdatasync
21:12:51	0	0.0%	0.702/0.109	49.5Mbps	13.87%--sys_fdatasync
21:14:23	1211	0.0%	0.732/0.105	47.5Mbps	12.21%--sys_fdatasync
21:15:53	125633	4.0%	0.86/0.121	40.4Mbps	36.60%--sys_pwrite64
21:17:24	1379822	44.6%	0.835/0.175	41.6Mbps	34.97%--sys_pwrite64
21:18:55	683034	21.9%	0.791/0.122	43.9Mbps	35.22%--sys_pwrite64
21:20:25	1790854	57.3%	0.673/0.082	51.6Mbps	34.73%--sys_pwrite64
21:21:56	3288552	99.5%	1.361/0.406	25.5Mbps	34.99%--sys_pwrite64
21:23:26	716453	23.0%	0.679/0.16	51.2Mbps	36.18%--sys_pwrite64
21:24:57	500137	16.1%	0.777/0.134	44.7Mbps	34.08%--sys_pwrite64
21:26:28	365872	11.4%	0.81/0.21	42.9Mbps	25.82%--sys_pwrite64
21:27:59	4131	0.1%	0.628/0.118	55.3Mbps	26.19%--sys_pwrite64
21:29:30	838693	26.7%	0.674/0.062	51.6Mbps	27.37%--sys_pwrite64
21:31:00	115372	3.6%	0.846/0.173	41.1Mbps	35.10%--sys_pwrite64
21:32:31	139553	4.4%	0.719/0.164	48.3Mbps	35.50%--sys_pwrite64
21:34:03	0	0.0%	0.81/0.182	42.9Mbps	13.52%--sys_fdatasync
21:35:35	6803	0.2%	0.637/0.086	54.6Mbps	14.31%--sys_fdatasync

“On high load caused by vacuum, restart point and activity, the standby filesystem can’t keep up but the network and the disk can”



Solution





Solution

wal_recycle = off

Solution

wal_init_zero = off

Solution

Pin process wal writer single cpu

```
taskset -pc 5 `pgrep -f walreceiver`
```

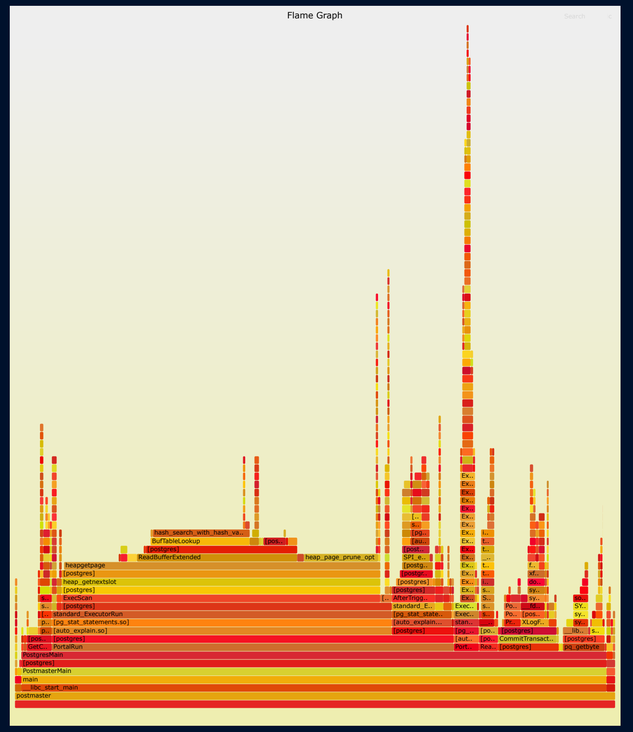
Lessons learned

Lessons learned

Lessons learned

0	0.0%	0.804/0.087
139	0.0%	0.808/0.133
0	0.0%	0.79/0.11
0	0.0%	0.702/0.109
1211	0.0%	0.732/0.105
125633	4.0%	0.86/0.121
1379822	44.6%	0.835/0.175
683034	21.9%	0.791/0.122
1790854	57.3%	0.673/0.082
3288552	99.5%	1.361/0.406
716453	23.0%	0.679/0.16
500137	16.1%	0.777/0.134
365872	11.4%	0.81/0.21
4131	0.1%	0.628/0.118
838693	26.7%	0.674/0.062
115372	3.6%	0.846/0.173
139553	4.4%	0.719/0.164
0	0.0%	0.81/0.182
6803	0.2%	0.637/0.086

SS & Perf



Lessons learned

Special thanks to

Adyen

Dmitry Fomin

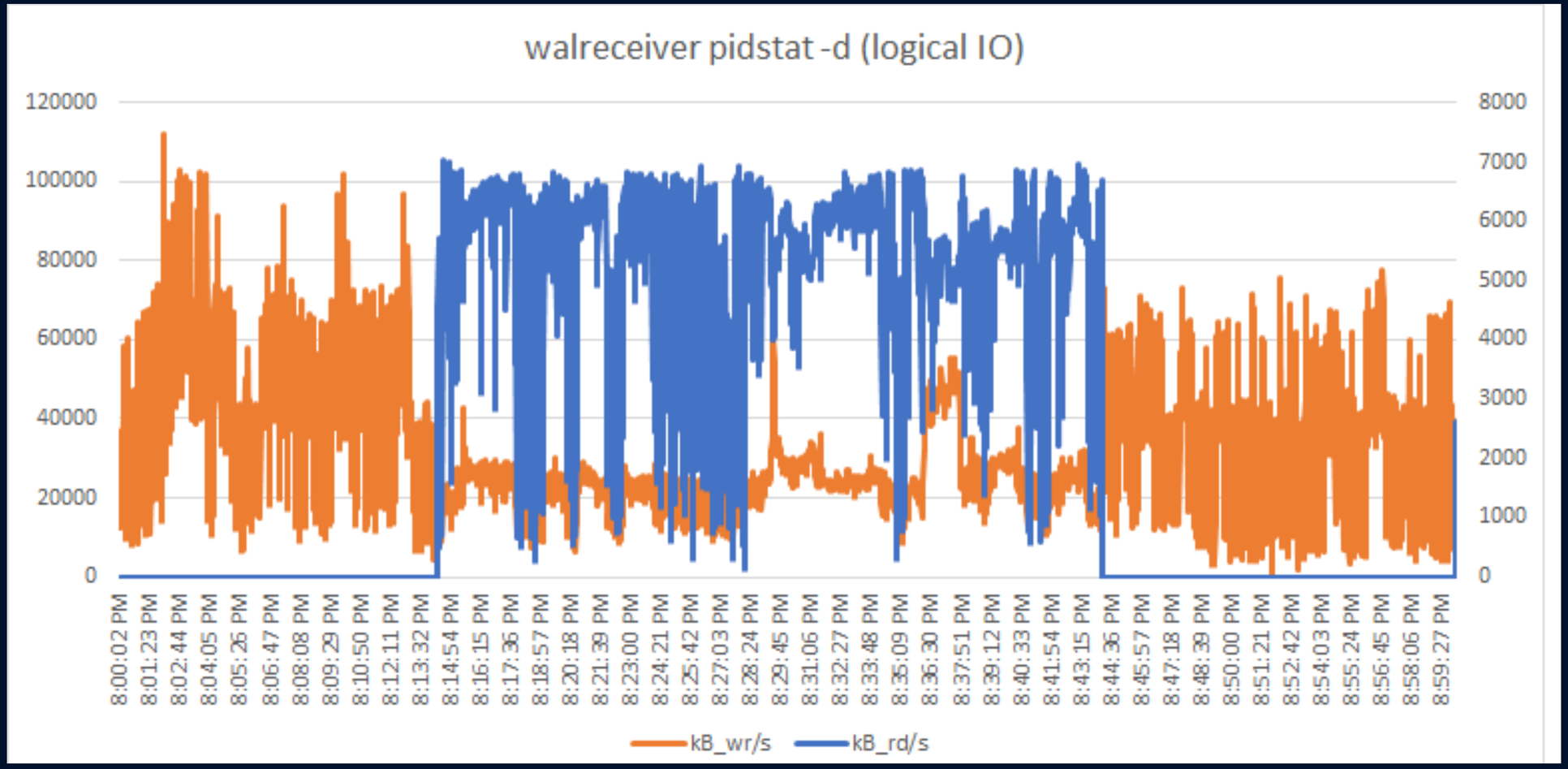
Milen Blagojevic

EDB

Jakub Wartak

Tomas Vondra

Álvaro Herrera Muñoz


walreceiver pidstat -d (logical IO)
120000
100000
80000
60000
40000
20000
0
8000
7000
6000
5000
4000
3000
2000
1000
0
8:00:02 PM
8:01:23 PM
8:02:44 PM
8:04:05 PM
8:05:26 PM
8:06:47 PM
8:08:08 PM
8:09:29 PM
8:10:50 PM
8:12:11 PM
8:13:32 PM
8:14:54 PM
8:16:15 PM
8:17:36 PM
8:18:57 PM
8:20:18 PM
8:21:39 PM
8:23:00 PM
8:24:21 PM
8:25:42 PM
8:27:03 PM
8:28:24 PM
8:29:45 PM
8:31:06 PM
8:32:27 PM
8:33:48 PM
8:35:09 PM
8:36:30 PM
8:37:51 PM
8:39:12 PM
8:40:33 PM
8:41:54 PM
8:43:15 PM
8:44:36 PM
8:45:57 PM
8:47:18 PM
8:48:39 PM
8:50:00 PM
8:51:21 PM
8:52:42 PM
8:54:03 PM
8:55:24 PM
8:56:45 PM
8:58:06 PM
8:59:27 PM
kB_wr/s
kB_rd/s

Questions