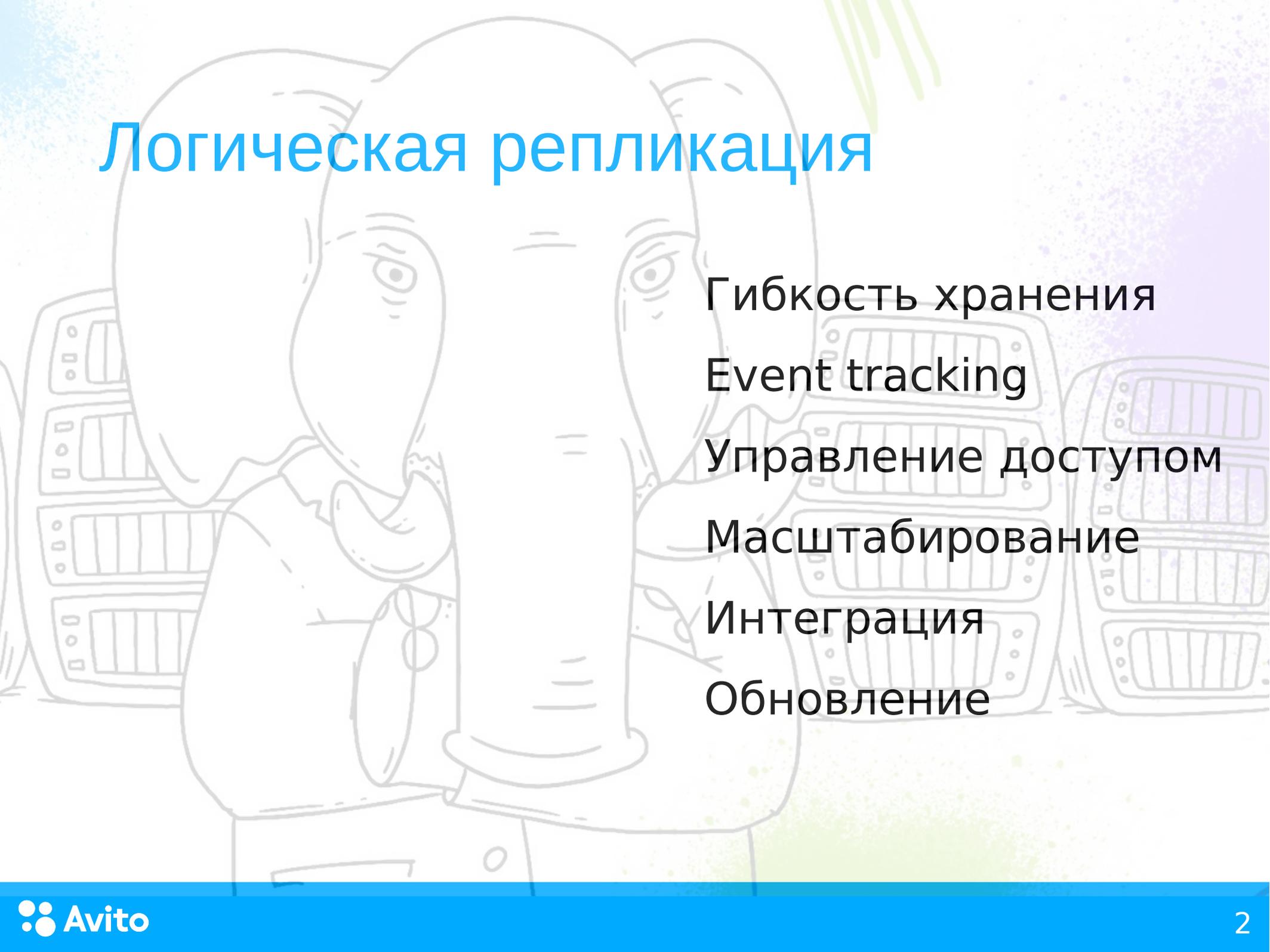




Кейсы использования логической репликации при восстановлении данных в PostgreSQL 10

Костантин Евтеев
Михаил Тюрин

Логическая репликация

A stylized line-art illustration of two men in suits and ties, one with his hand to his chin in a thoughtful pose, standing in front of several server racks. The background is a light purple and green gradient with some abstract brushstrokes.

Гибкость хранения

Event tracking

Управление доступом

Масштабирование

Интеграция

Обновление

Logical replication in Avito

Building data streams (PGDAY 2016) goo.gl/1ZT4Bb

Поток данных в Авито (PgConf.Russia 2016) goo.gl/mwhwya

Логическая репликация в Avito (HL 2017 <https://goo.gl/xSBXeT>)

Доставка справочников

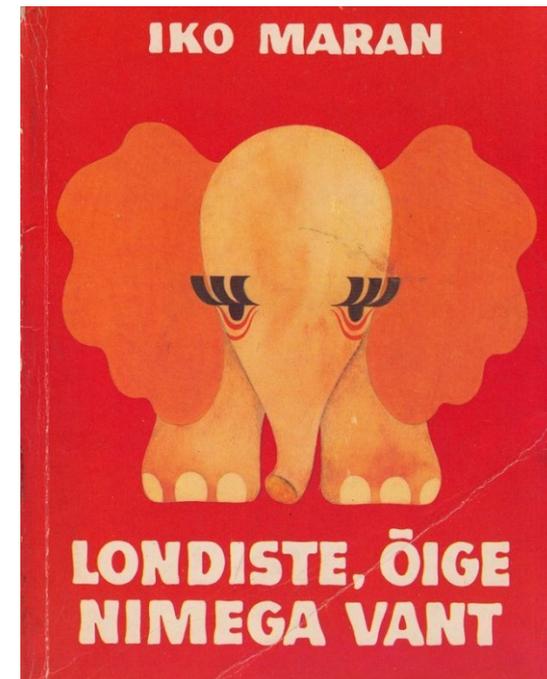
Распределение нагрузки

Partial replication в сервисы

Доставка данных в поисковые системы

Persistent queue

Межсервисное взаимодействие

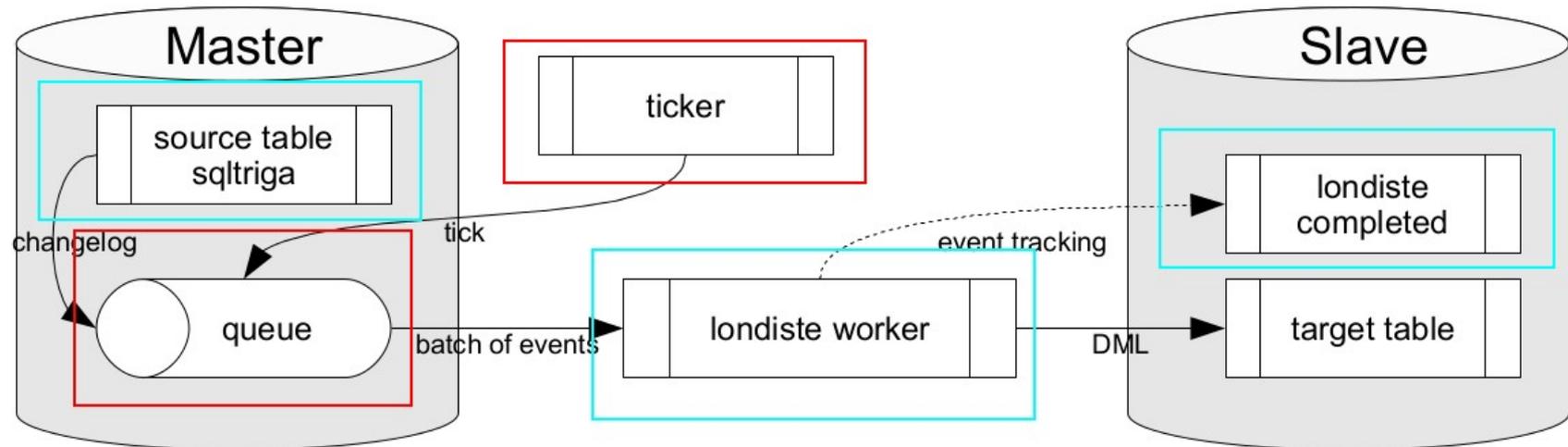




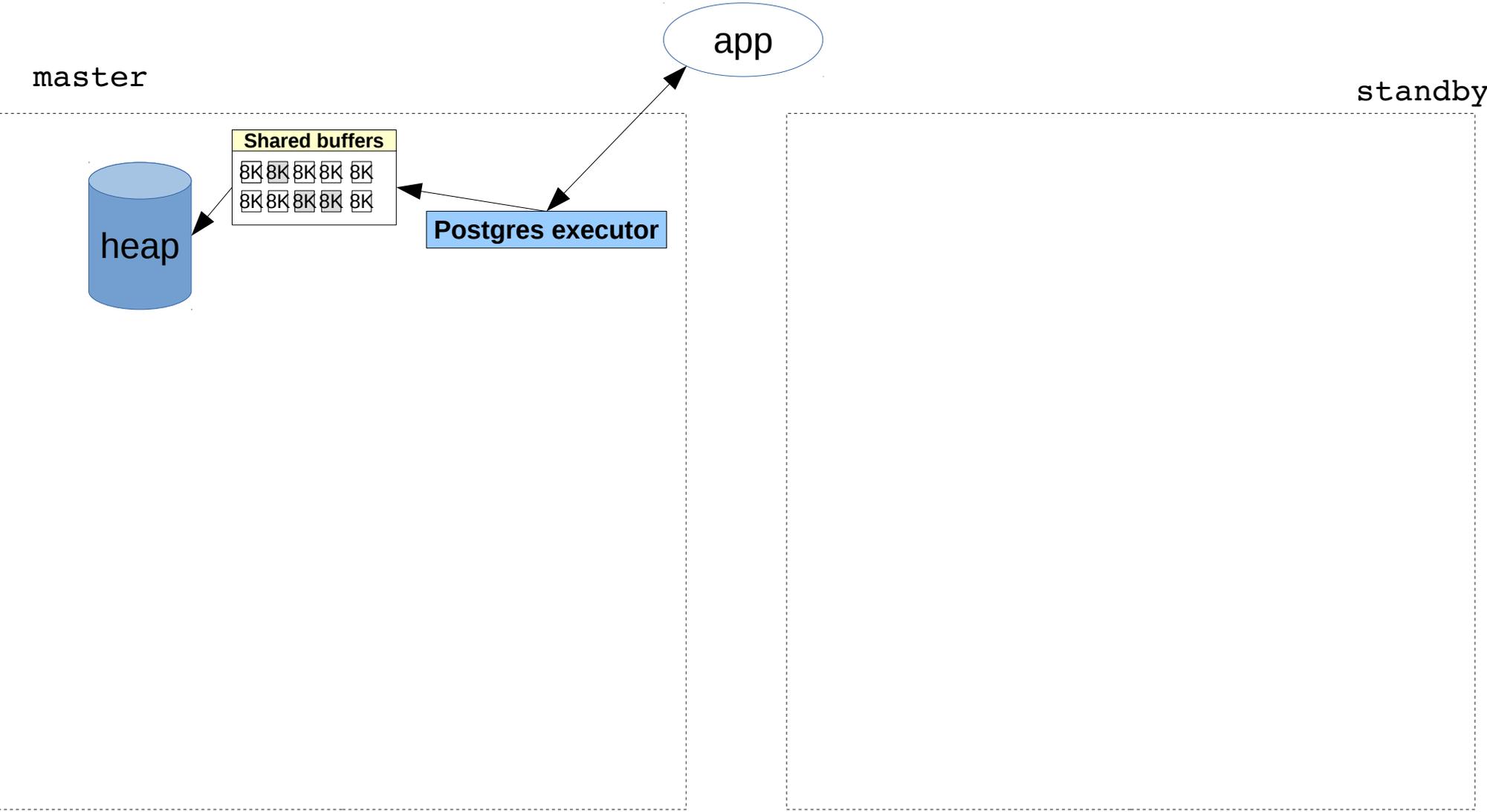
Londiste

https://www.pgcon.org/2009/schedule/attachments/91_pgq.pdf

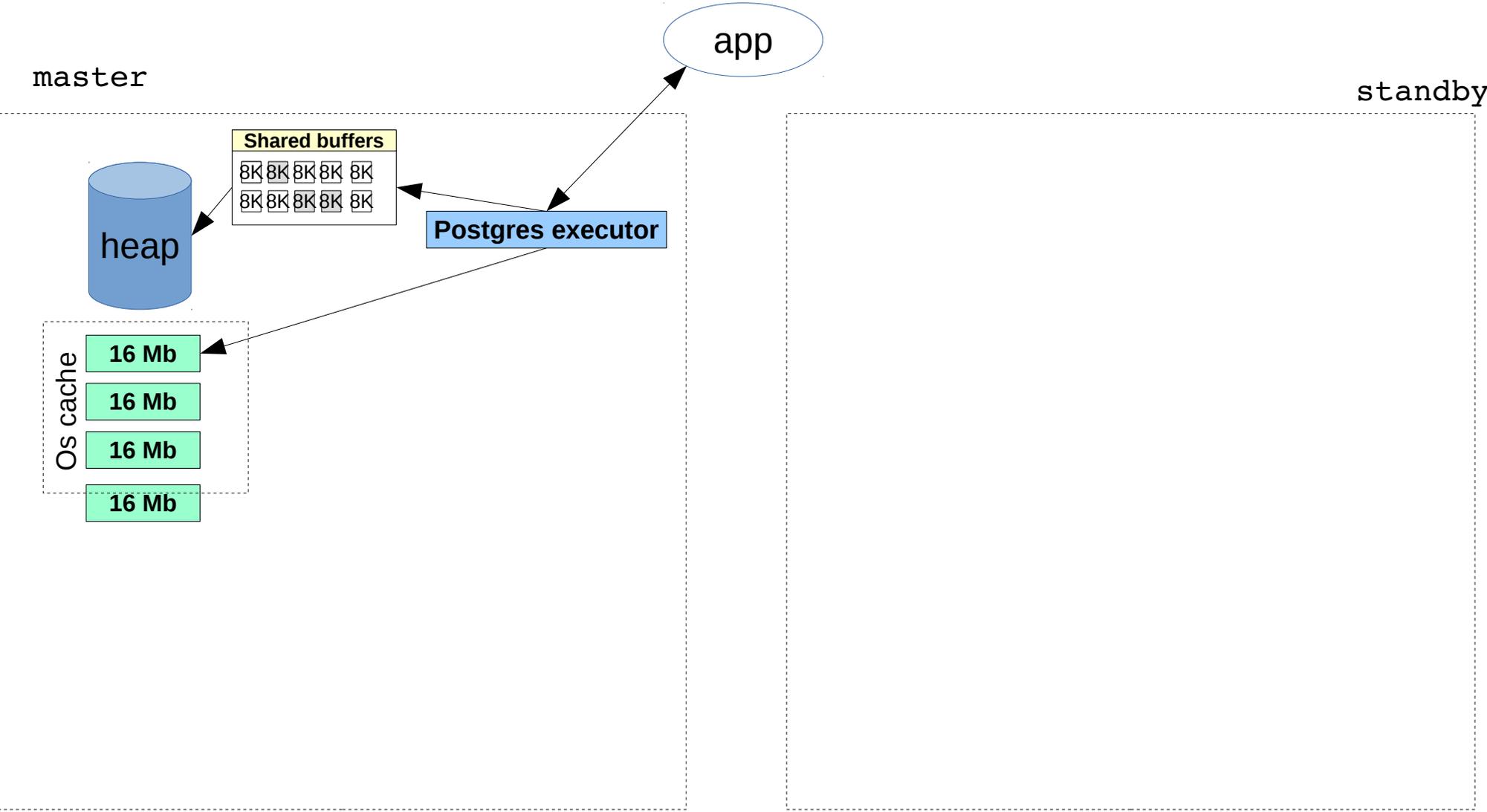
<https://github.com/avito-tech/skytools>



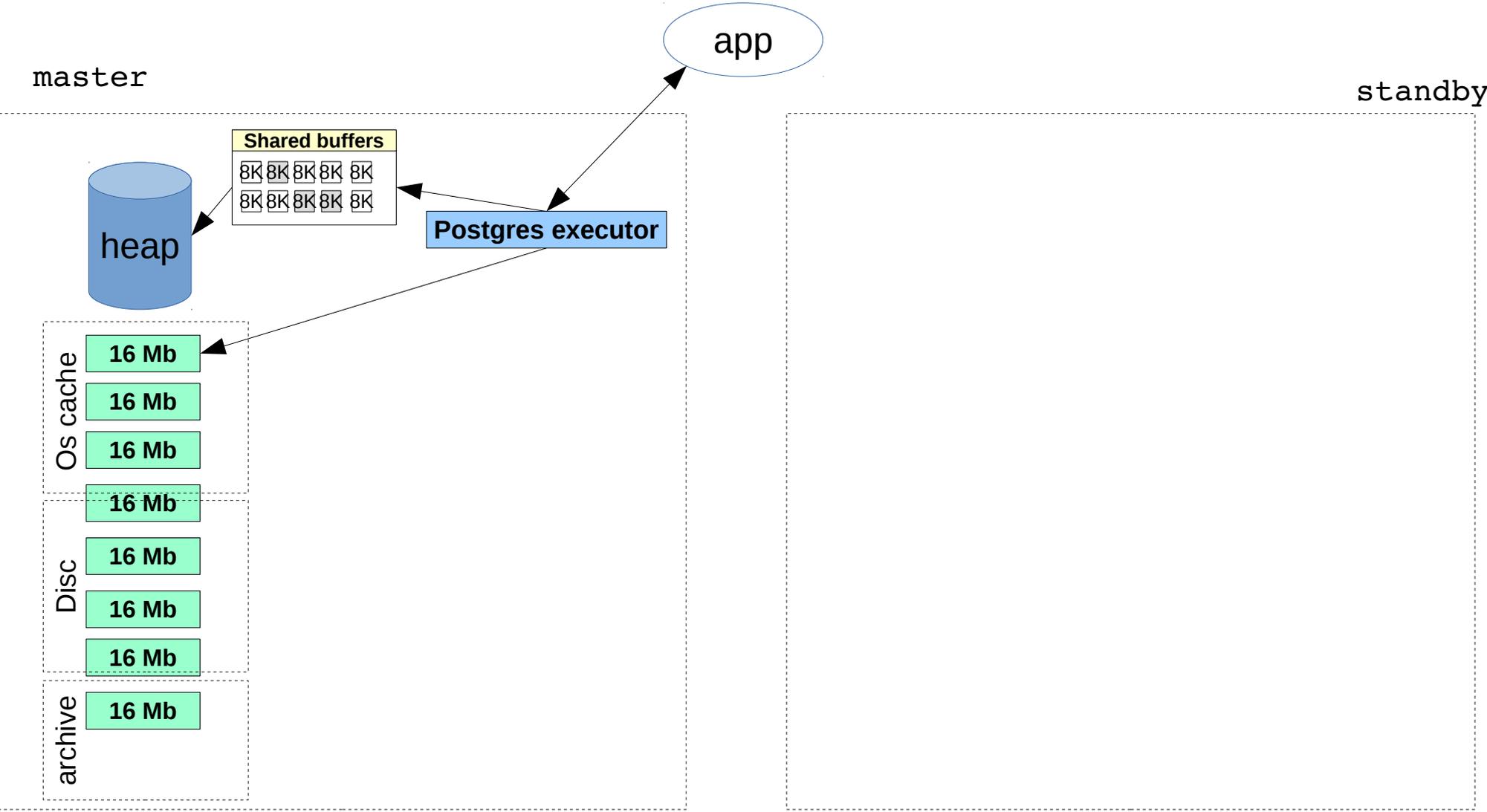
Streaming



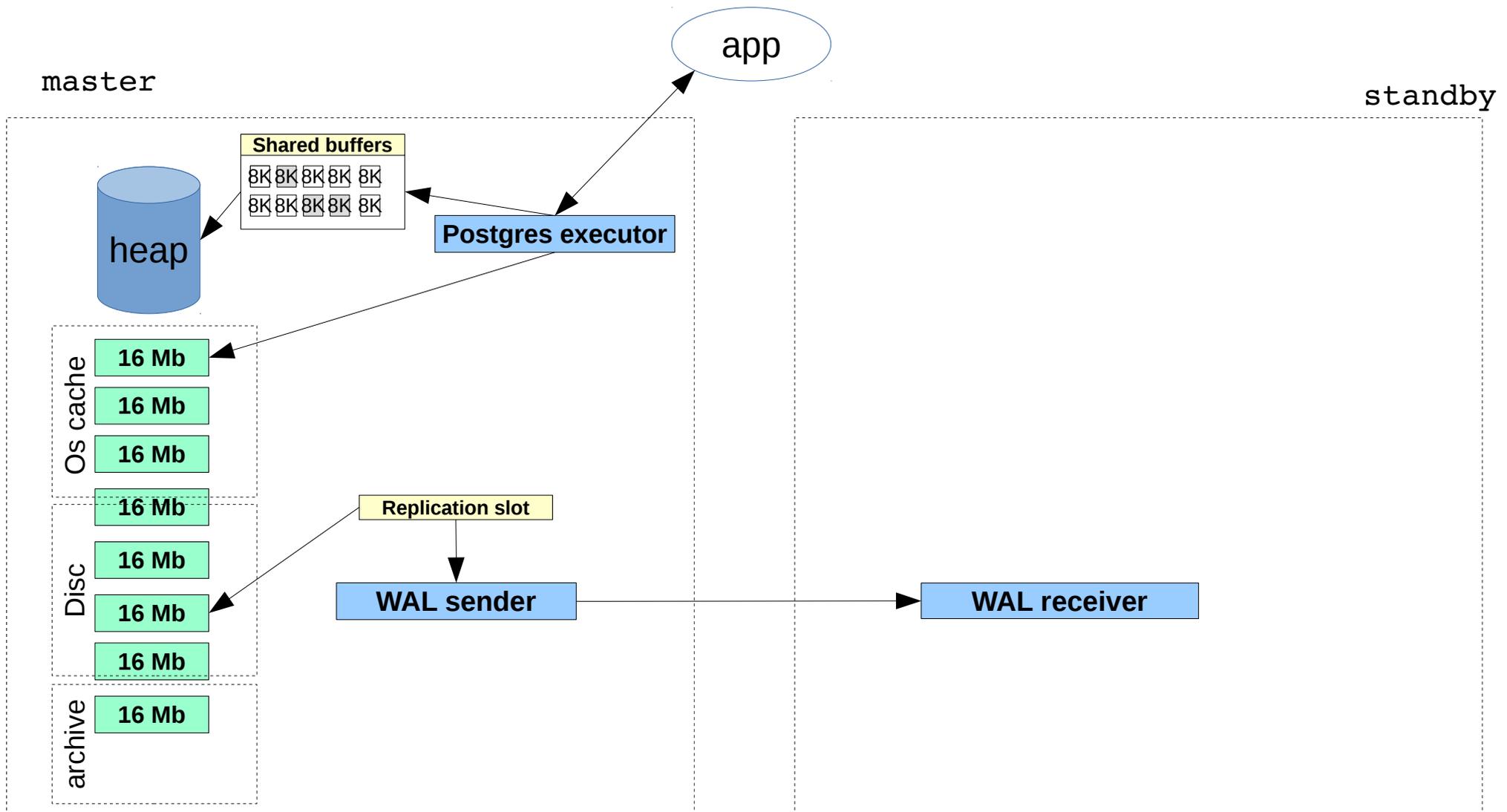
Streaming



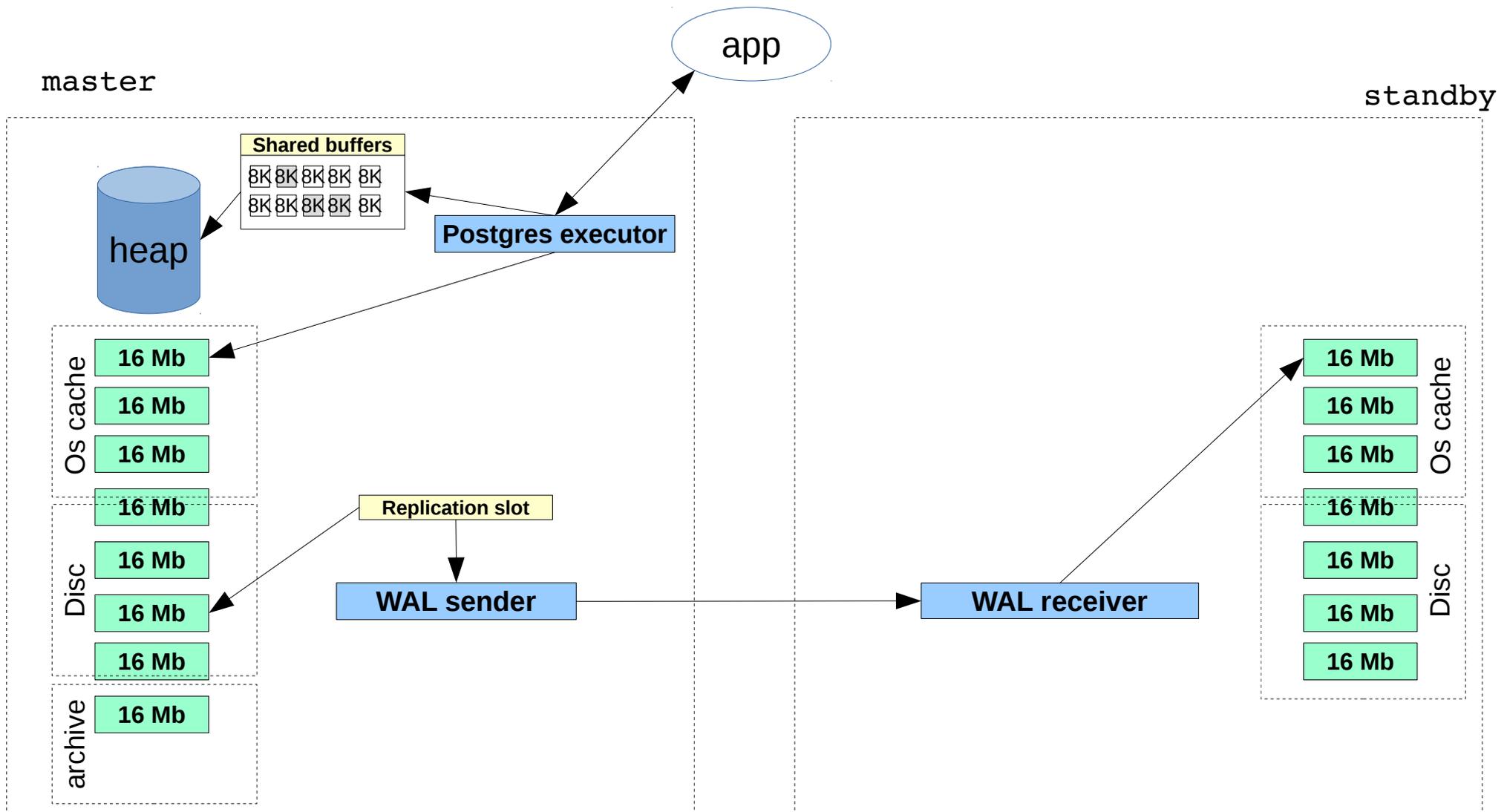
Streaming



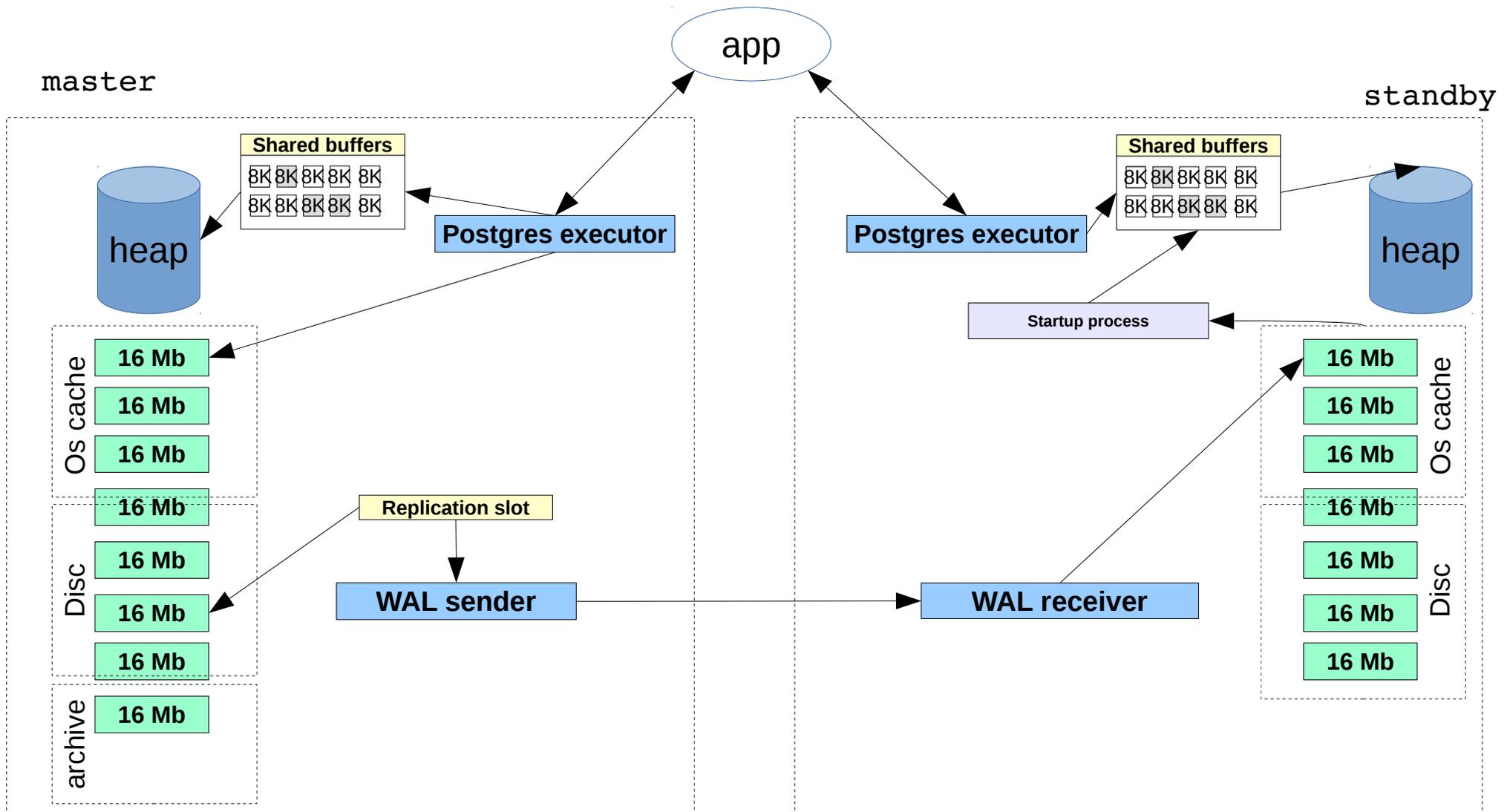
Streaming



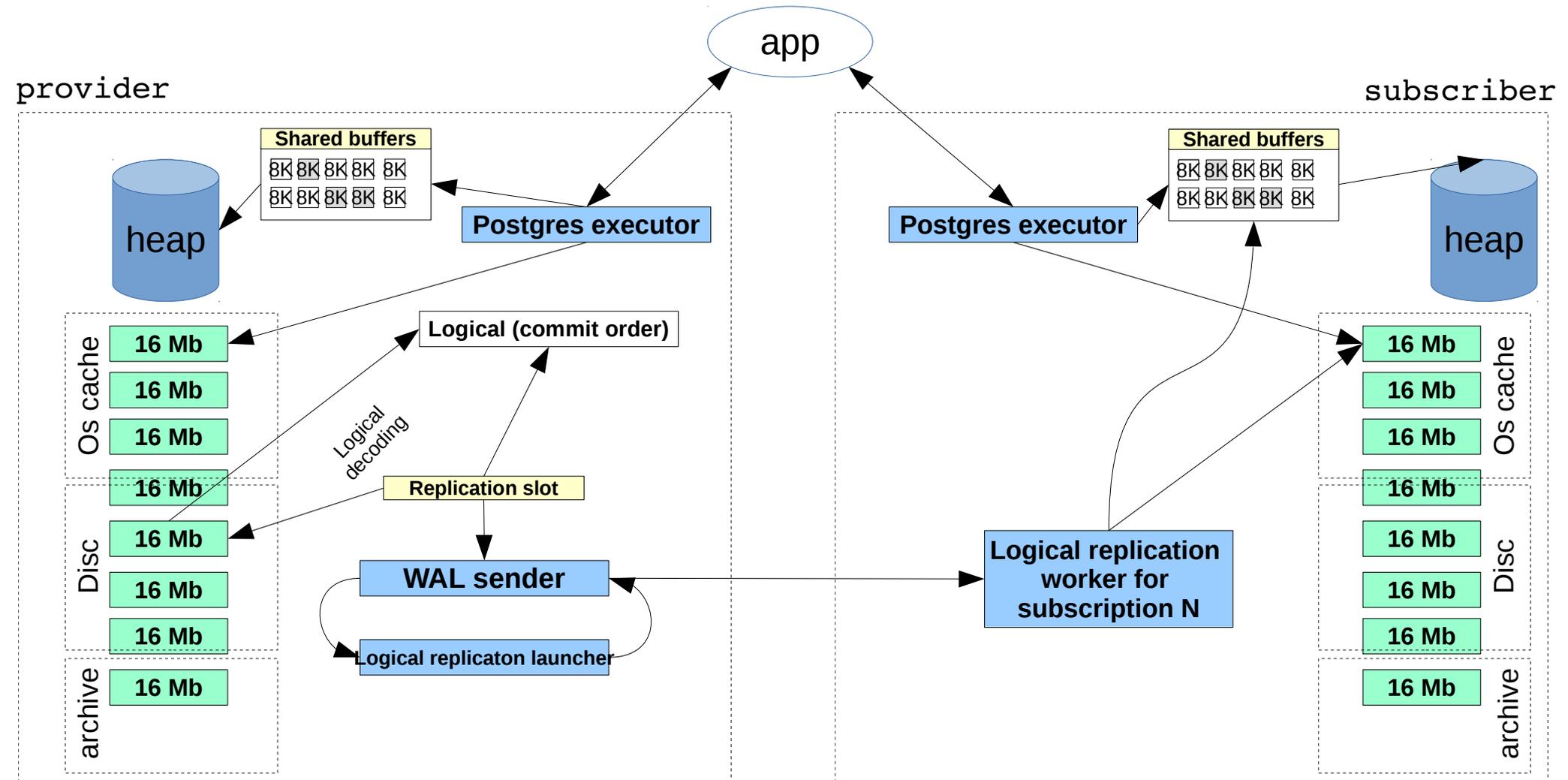
Streaming



Streaming



Logical

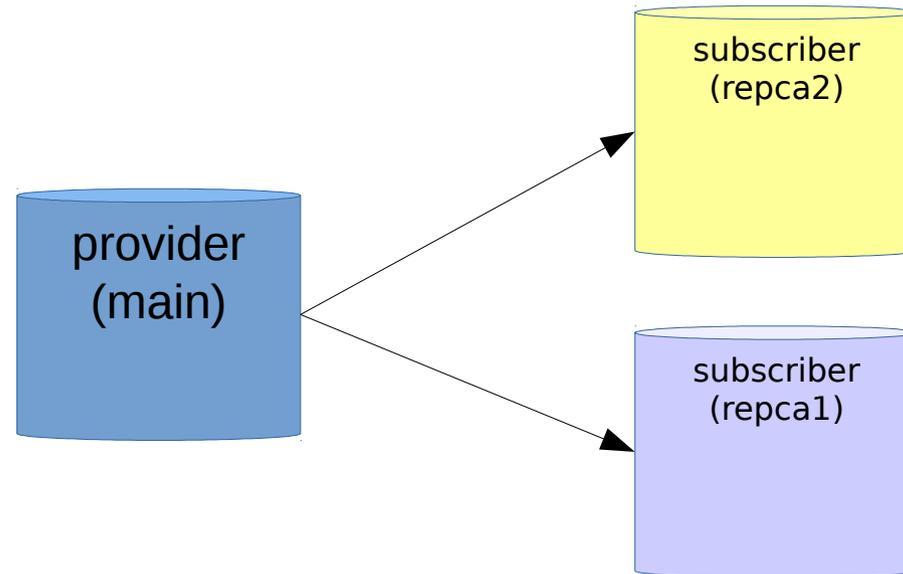


Кейсы восстановления

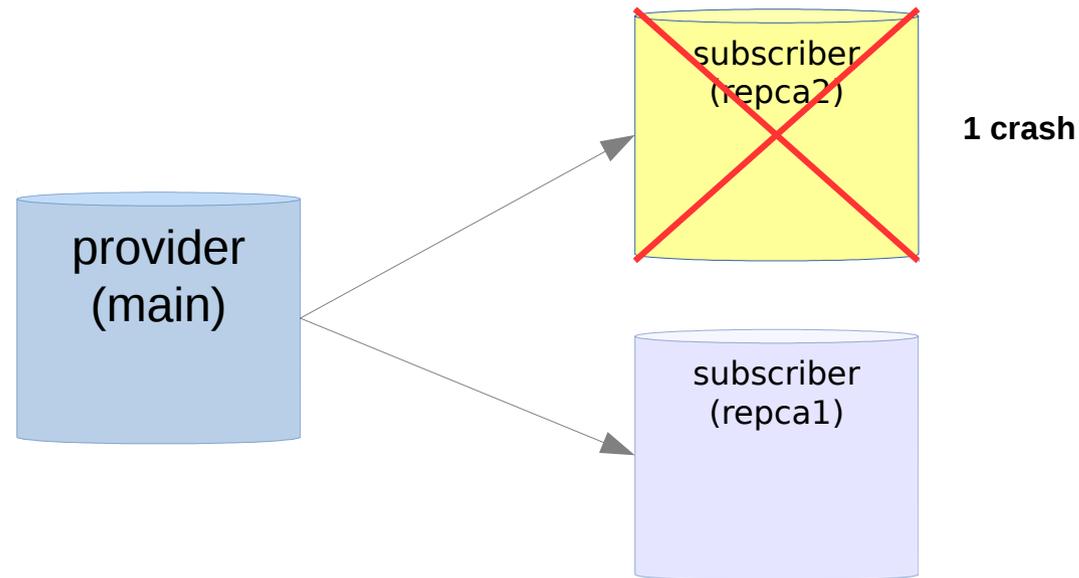
Logical (10):

- Пересоздание subscriber из другого subscriber
- UNDO (авария на provider и переключение на provider's standby)
- REDO (авария на subscriber и переключение на subscriber's standby)

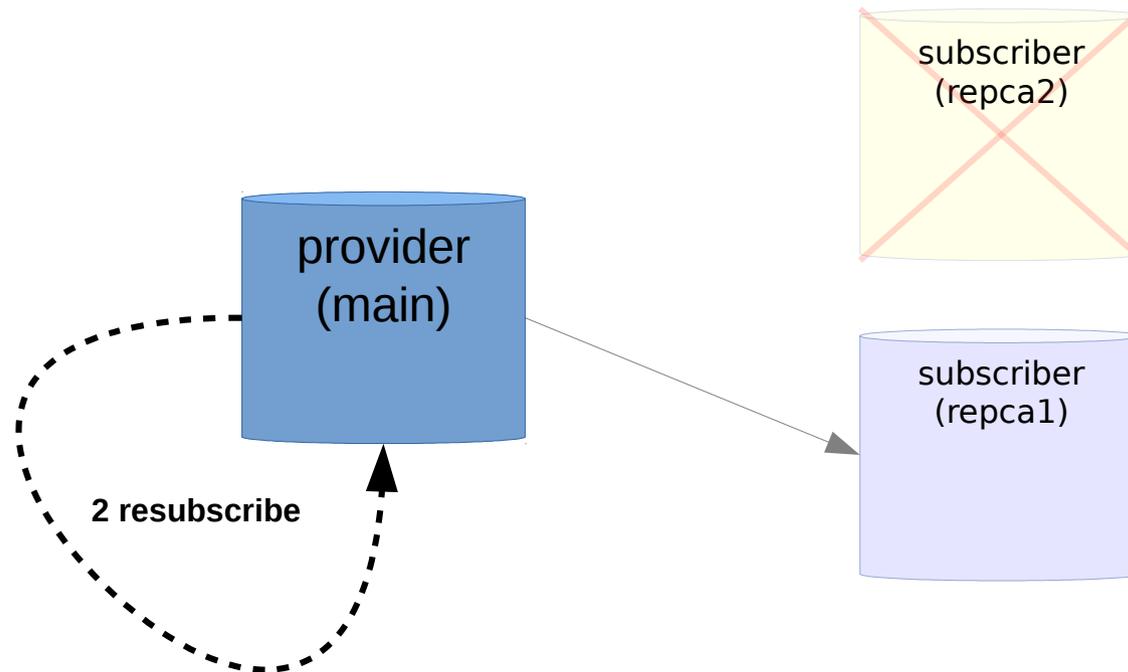
Пересоздание subscriber из другого subscriber



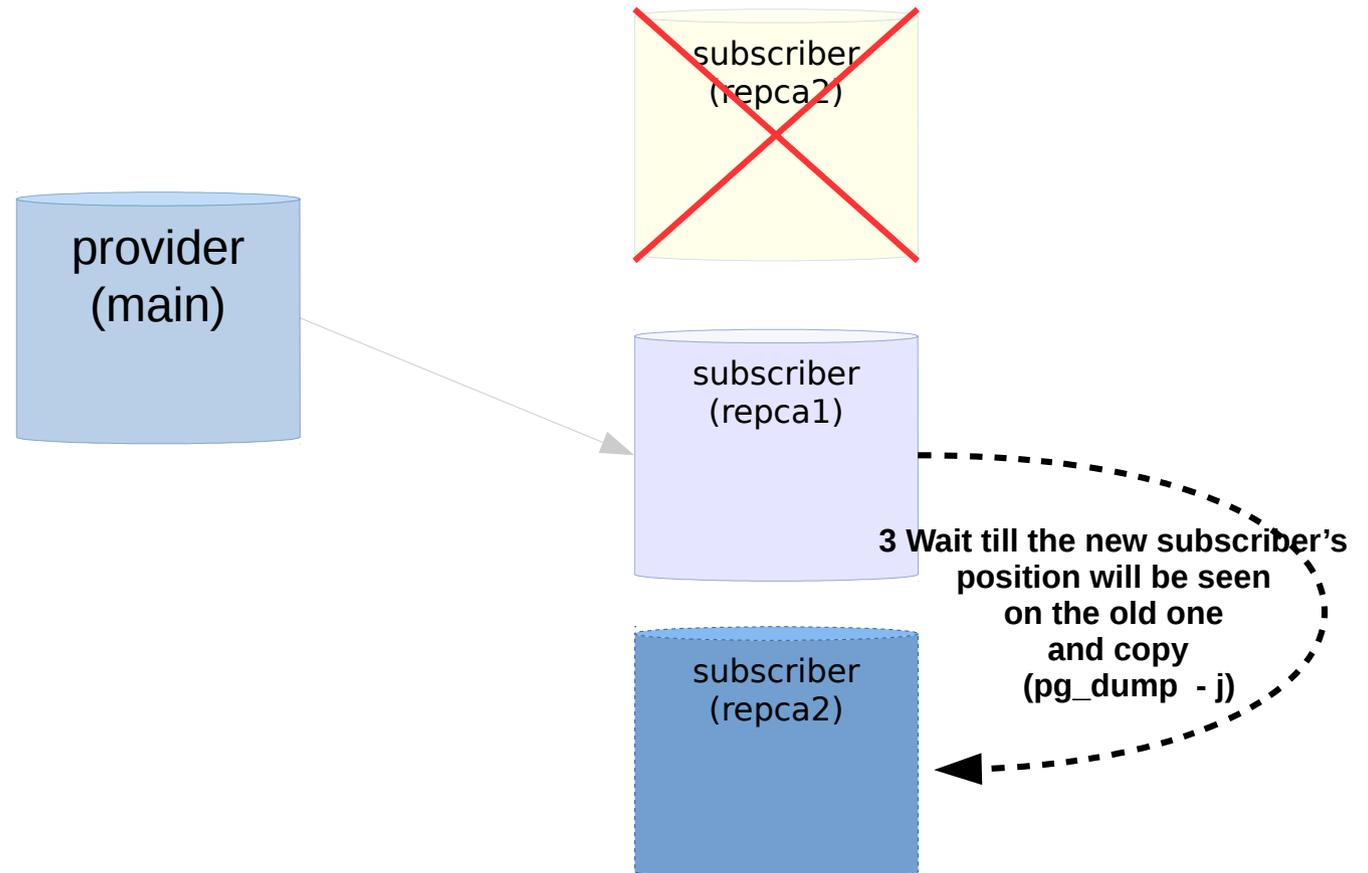
Пересоздание subscriber из другого subscriber



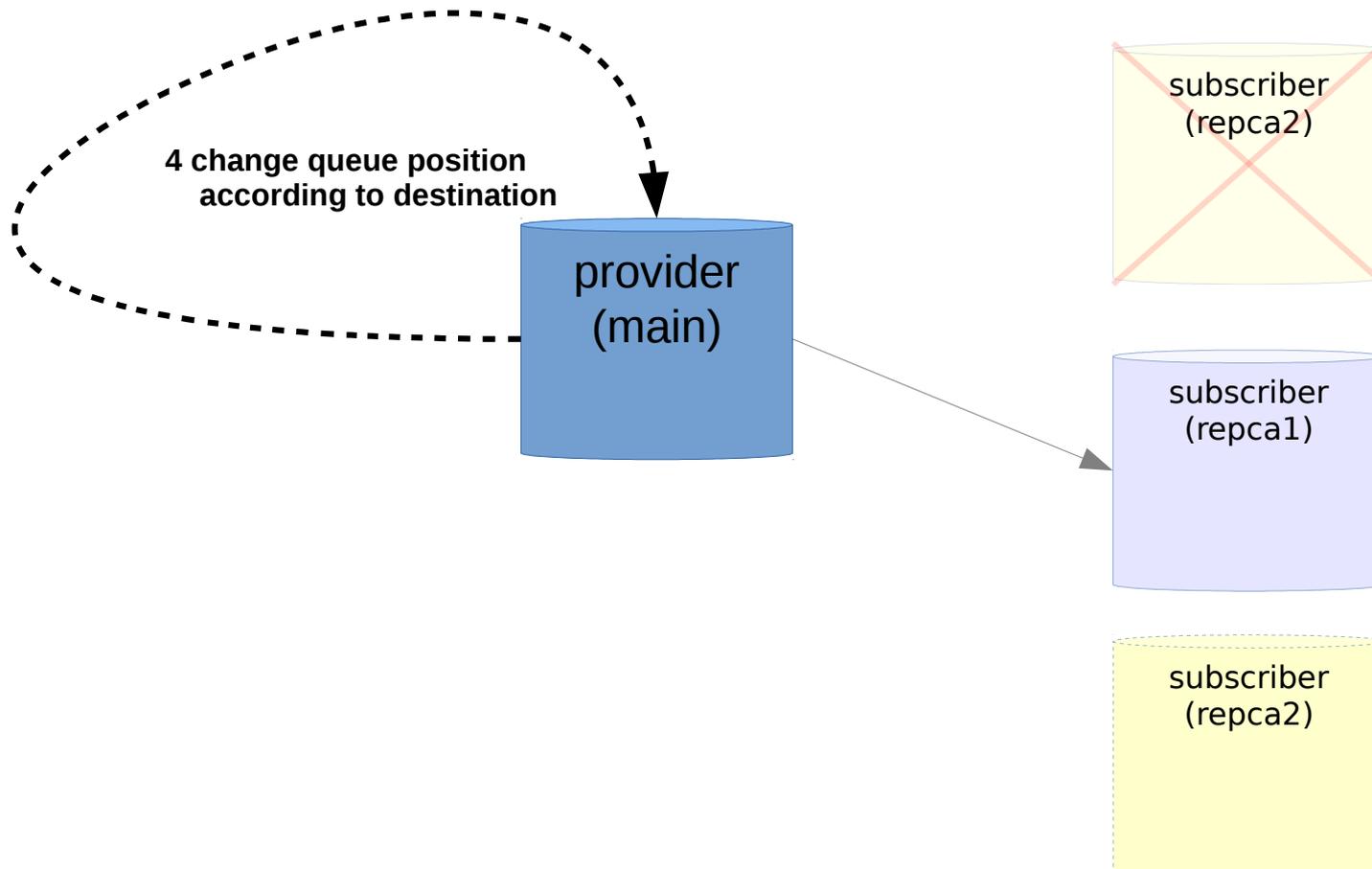
Пересоздание subscriber из другого subscriber



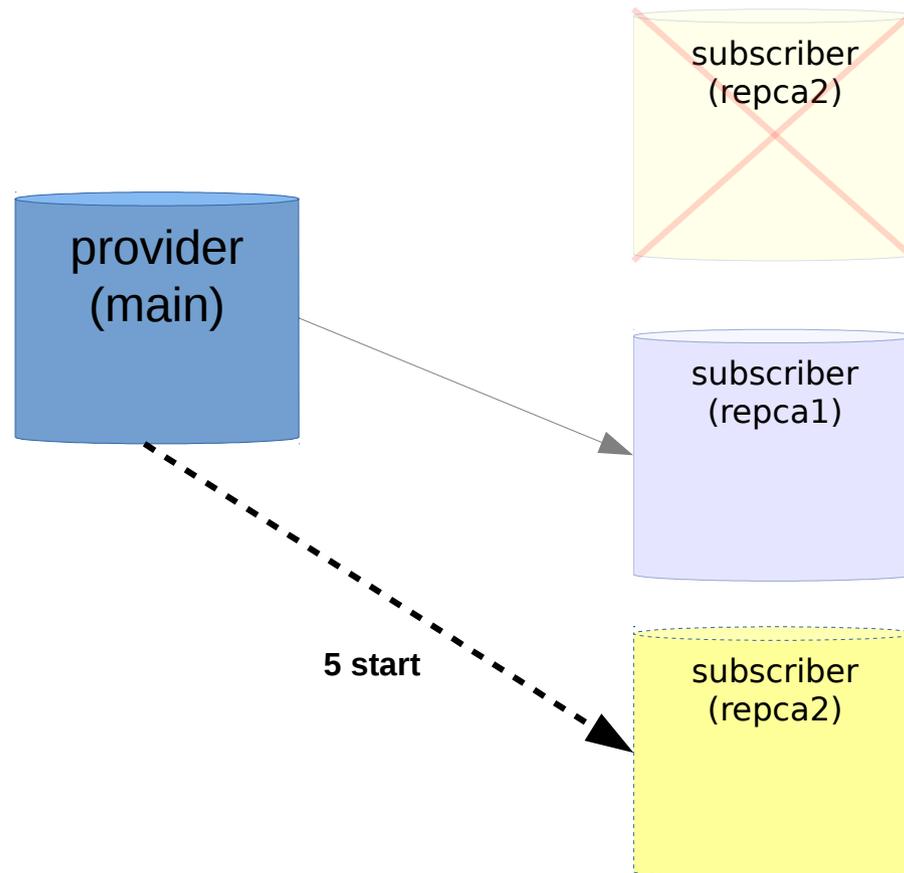
Пересоздание subscriber из другого subscriber



Пересоздание subscriber из другого subscriber



Пересоздание subscriber из другого subscriber



Пересоздание subscriber из другого subscriber

1. Слот для новой реплики:

provider
(main)

```
pg_recvlogical --cluster 10/main -U postgres -d src --create-slot -S repca2 -P pgoutput  
psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f			561	0/30409D0	0/3040A08
repca1	pgoutput	logical	16384	src	f	t	16117		561	0/30409D0	0/3040A08
standby		physical			f	t	15871			0/3040A08	

Пересоздание subscriber из другого subscriber

1. Слот для новой реплики:

provider
(main)

```
pg_recvlogical --cluster 10/main -U postgres -d src --create-slot -S repca2 -P pgoutput  
psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f			561	0/30409D0	0/3040A08
repca1	pgoutput	logical	16384	src	f	t	16117		561	0/30409D0	0/3040A08
standby		physical			f	t	15871			0/3040A08	

2. Отключаем активную подписку:

subscriber
(repca1)

```
psql --cluster 10/repca1 -U postgres -X -d dst -c 'alter subscription repca1 disable'
```

Пересоздание subscriber из другого subscriber

1. Слот для новой реплики:

provider
(main)

```
pg_recvlogical --cluster 10/main -U postgres -d src --create-slot -S repca2 -P pgoutput  
psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f			561	0/30409D0	0/3040A08
repca1	pgoutput	logical	16384	src	f	t	16117		561	0/30409D0	0/3040A08
standby		physical			f	t	15871			0/3040A08	

2. Отключаем активную подписку:

subscriber
(repca1)

```
psql --cluster 10/repca1 -U postgres -X -d dst -c 'alter subscription repca1 disable'
```

3. Текущий LSN::

subscriber
(repca1)

```
psql --cluster 10/repca1 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
```

local_id	external_id	remote_lsn	local_lsn
1	pg 16430	0/3040998	0/1679D80

Пересоздание subscriber из другого subscriber

```
4. pg_dump --cluster 10/repca1 -U postgres -Fc --serializable-deferrable --no-subscriptions -d dst \  
  | pg_restore --cluster 10/repca2 -U postgres -C -d postgres
```

Пересоздание subscriber из другого subscriber

4. `pg_dump --cluster 10/repca1 -U postgres -Fc --serializable-deferrable --no-subscriptions -d dst \`
`| pg_restore --cluster 10/repca2 -U postgres -C -d postgres`

5. параллельно:

`psql --cluster 10/repca1 -U postgres -X -d dst -c 'alter subscription repca1 enable'`

subscriber
(repca1)

`psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'`

provider
(main)

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f			561	0/30409D0	0/3040A08
repca1	pgoutput	logical	16384	src	f	t	16727		562	0/30410A8	0/30411C0
standby		physical			f	t	15871			0/30411C0	

Пересоздание subscriber из другого subscriber



restart_lsn =



restart_lsn

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f			561	0/30409D0	0/3040A08
repca1	pgoutput	logical	16384	src	f	t	16117		561	0/30409D0	0/3040A08
standby		physical			f	t	15871			0/3040A08	



restart_lsn >



restart_lsn

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f			561	0/30409D0	0/3040A08
repca1	pgoutput	logical	16384	src	f	t	16727		562	0/30410A8	0/30411C0
standby		physical			f	t	15871			0/30411C0	

Пересоздание subscriber из другого subscriber

6. Создаем подписку

subscriber
(repca2)

```
psql --cluster 10/repca2 -U postgres -X -d dst -f- <<'EOF'  
create subscription repca2 connection 'port=5432 dbname=src'  
    publication pub with (enabled = false, create_slot = false, copy_data = false);  
  
select oid, * from pg_subscription;  
EOF
```

```
CREATE SUBSCRIPTION  
 oid | subdbid | subname | subowner | subenabled | subconninfo | subslotname | subsynccommit | subpublications  
-----+-----+-----+-----+-----+-----+-----+-----+-----  
16558 | 16384 | repca2 | 10 | f | port=5432 dbname=src | repca2 | off | {pub}
```

Пересоздание subscriber из другого subscriber

6. Создаем подписку

subscriber
(repca2)

```
psql --cluster 10/repca2 -U postgres -X -d dst -f- <<'EOF'  
create subscription repca2 connection 'port=5432 dbname=src'  
    publication pub with (enabled = false, create_slot = false, copy_data = false);  
  
select oid, * from pg_subscription;  
EOF
```

```
CREATE SUBSCRIPTION  
 oid | subdbid | subname | subowner | subenabled | subconninfo | subslotname | subsynccommit | subpublications  
-----+-----+-----+-----+-----+-----+-----+-----+-----  
16558 | 16384 | repca2 | 10 | f | port=5432 dbname=src | repca2 | off | {pub}
```

7. Сдвигаем LSN

subscriber
(repca2)

```
psql --cluster 10/repca2 -U postgres -X -d dst -c "select pg_replication_origin_advance('pg_16558', '0/3040998')"
```

```
# psql --cluster 10/repca1 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
```

subscriber
(repca1)

```
local_id | external_id | remote_lsn | local_lsn  
-----+-----+-----+-----  
1 | pg 16430 | 0/3040998 | 0/1679D80
```

Пересоздание subscriber из другого subscriber

8. Проверим состояние:

subscriber
(repca2)

```
psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
```

local_id	external_id	remote_lsn	local_lsn
1	pg_16558	0/3040998	0/0

Пересоздание subscriber из другого subscriber

8. Проверим состояние:

subscriber
(repca2)

```
psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
```

```
local_id | external_id | remote_lsn | local_lsn  
-----+-----+-----+-----  
1 | pg_16558 | 0/3040998 | 0/0
```

9. Включаем подписку:

```
psql --cluster 10/repca2 -U postgres -X -d dst -c 'alter subscription repca2 enable'
```

provider
(main)

```
psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

```
slot_name | plugin | slot_type | datoid | database | temporary | active | active_pid | xmin | catalog_xmin | restart_lsn | confirmed_flush_lsn  
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----  
repca2 | pgoutput | logical | 16384 | src | f | t | 18501 | | 562 | 0/30410A8 | 0/30411C0  
repca1 | pgoutput | logical | 16384 | src | f | t | 16727 | | 562 | 0/30410A8 | 0/30411C0  
standby | | physical | | | f | t | 15871 | | | 0/30411C0 |
```

Пересоздание subscriber из другого subscriber

8. Проверим состояние:

subscriber
(repca2)

```
psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
```

```
local_id | external_id | remote_lsn | local_lsn  
-----+-----+-----+-----  
1 | pg_16558 | 0/3040998 | 0/0
```

9. Включаем подписку:

```
psql --cluster 10/repca2 -U postgres -X -d dst -c 'alter subscription repca2 enable'
```

```
psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

provider
(main)

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	t	18501		562	0/30410A8	0/30411C0
repca1	pgoutput	logical	16384	src	f	t	16727		562	0/30410A8	0/30411C0
standby		physical			f	t	15871			0/30411C0	

subscriber
(repca2)

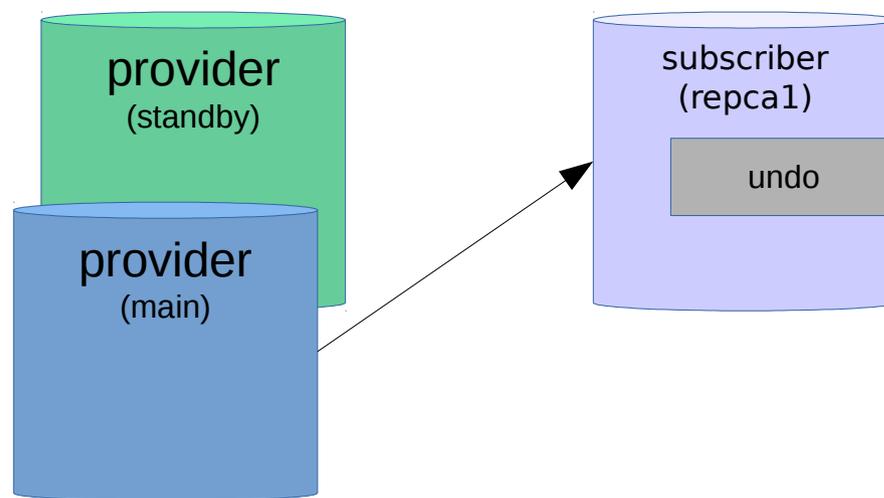
```
oghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'  
local_id | external_id | remote_lsn | local_lsn  
-----+-----+-----+-----  
1 | pg_16558 | 0/30410A8 | 0/16EA5A0  
(1 строка)
```

subscriber
(repca1)

```
oghack@pghack-debian-8:~$ psql --cluster 10/repca1 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'  
local_id | external_id | remote_lsn | local_lsn  
-----+-----+-----+-----  
1 | pg_16430 | 0/30410A8 | 0/17218A0  
(1 строка)
```

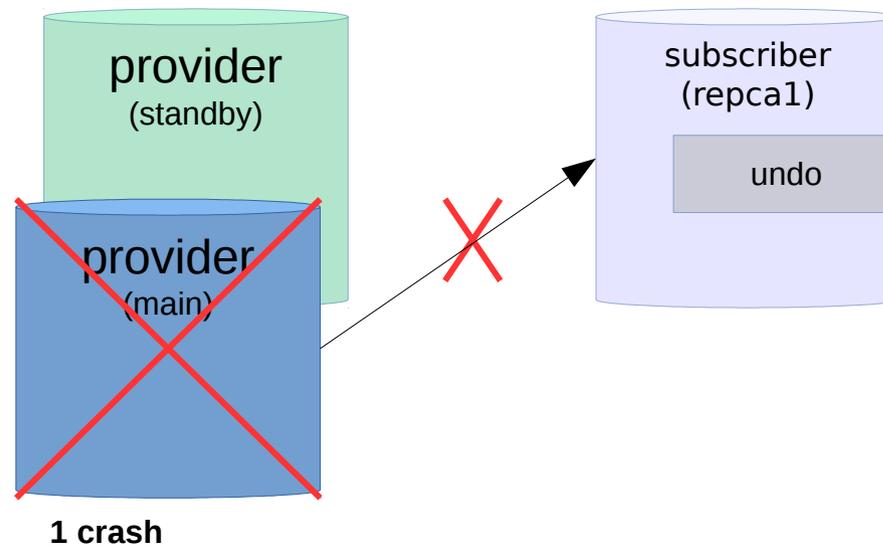
UNDO

(авария на provider и переключение на provider's standby)



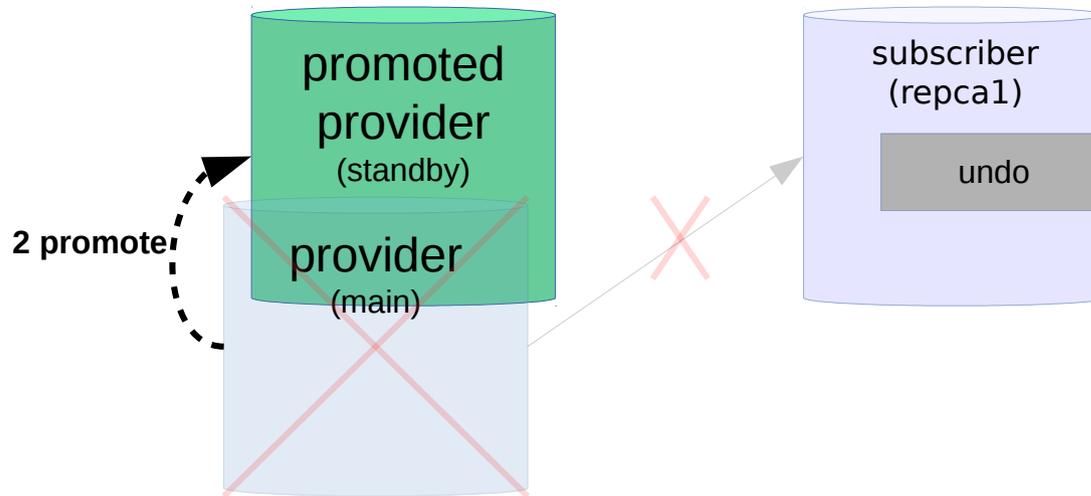
UNDO

(авария на provider и переключение на provider's standby)



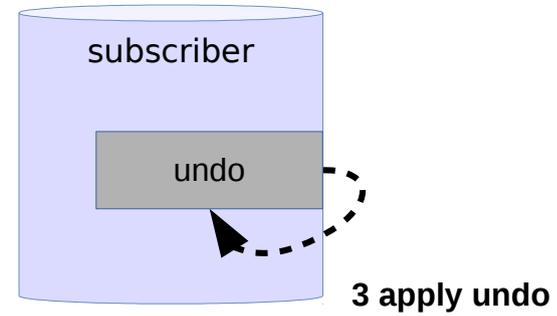
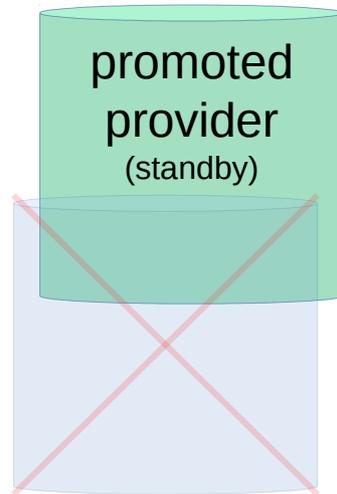
UNDO

(авария на provider и переключение на provider's standby)



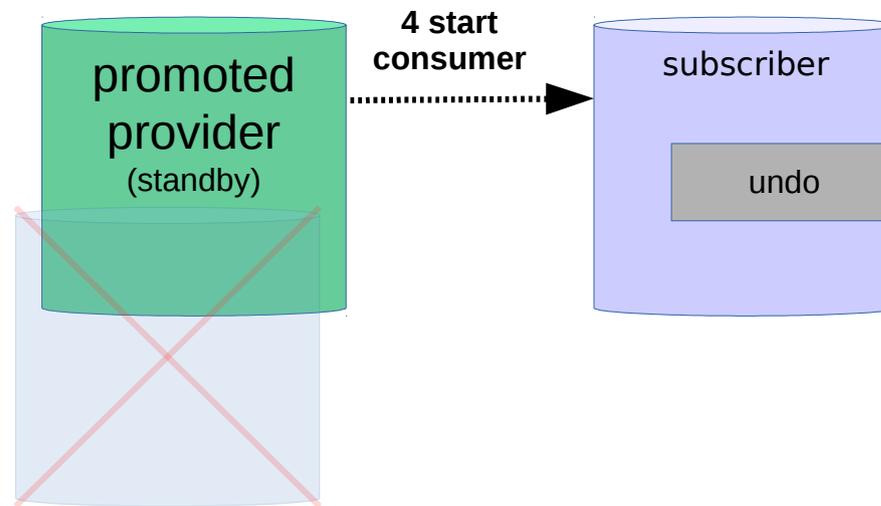
UNDO

(авария на provider и переключение на provider's standby)



UNDO

(авария на provider и переключение на provider's standby)



UNDO

(авария на provider и переключение на provider's standby)

<https://github.com/avito-tech/dba-docs/blob/master/2017-highload/plan.text>

```
create table undo_log (  
    id bigserial primary key,  
    txtime timestamptz default now(),  
    lsn pg_lsn default pg_replication_origin_session_progress(false),  
    dst_schema name, dst_table name,  
    undo_cmd char, cmd_data hstore, cmd_pk hstore -- identity  
);
```

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca1 -U postgres -X -d dst -c 'select * from undo_log order by id'
```

id	txtime	lsn	dst_schema	dst_table	undo_cmd	cmd_data	cmd_pk
1	2018-01-07 16:53:08.07825+03	0/3040998	public	cats	D		"cat_id"=>"3"
2	2018-01-07 16:53:08.07825+03	0/3040998	public	items	D		"item_id"=>"4"
3	2018-01-07 18:49:37.703375+03	0/30410A8	public	cats	D		"cat_id"=>"4"
4	2018-01-07 18:49:37.703375+03	0/30410A8	public	cats	D		"cat_id"=>"5"
5	2018-01-07 18:49:37.703375+03	0/30410A8	public	cats	D		"cat_id"=>"6"
6	2018-01-07 19:06:21.510662+03	0/3041728	public	cats	I	"cat_id"=>"4", "cat_name"=>"категория 10"	
7	2018-01-07 19:06:21.510662+03	0/3041728	public	cats	I	"cat_id"=>"5", "cat_name"=>"категория 20"	
8	2018-01-07 19:06:21.510662+03	0/3041728	public	cats	I	"cat_id"=>"6", "cat_name"=>"категория 30"	
9	2018-01-07 19:07:22.8155+03	0/3041AB8	public	cats	D		"cat_id"=>"7"
10	2018-01-07 19:07:22.8155+03	0/3041AB8	public	cats	D		"cat_id"=>"8"
11	2018-01-07 19:07:22.8155+03	0/3041AB8	public	cats	D		"cat_id"=>"9"

UNDO

(авария на provider и переключение на provider's standby)

1. Slot не реплицируются на стендбай, поэтому нельзя включать трафик сразу после промотирования стендбая

provider
(standby)

```
pghack@pghack-debian-8:~$ sudo pg_ctlcluster 10 standby promote  
pghack@pghack-debian-8:~$ pg_recvlogical --cluster 10/standby -U postgres -d src --create-slot -S repcal -P pgoutput
```

UNDO

(авария на provider и переключение на provider's standby)

1. Slot не реплицируются на стендбай, поэтому нельзя включать трафик сразу после промотирования стендбая

provider
(standby)

```
pghack@pghack-debian-8:~$ sudo pg_ctlcluster 10 standby promote
pghack@pghack-debian-8:~$ pg_recvlogical --cluster 10/standby -U postgres -d src --create-slot -S repcal -P pgoutput
```

2. Запомнить позицию стендбая перед промотированием

provider
(standby)

```
pghack@pghack-debian-8:~$ psql --cluster 10/standby -U postgres -X -d src -c 'select pg_last_wal_replay_lsn()'
pg_last_wal_replay_lsn
-----
0/3041840
```

UNDO

(авария на provider и переключение на provider's standby)

1. Slot не реплицируются на стэндбай, поэтому нельзя включать трафик сразу после промотирования стэндбая

provider
(standby)

```
pghack@pghack-debian-8:~$ sudo pg_ctlcluster 10 standby promote
pghack@pghack-debian-8:~$ pg_recvlogical --cluster 10/standby -U postgres -d src --create-slot -S repcal -P pgoutput
```

2. Запомнить позицию стэндбая перед промотированием

provider
(standby)

```
pghack@pghack-debian-8:~$ psql --cluster 10/standby -U postgres -X -d src -c 'select pg_last_wal_replay_lsn()'
pg_last_wal_replay_lsn
-----
0/3041840
```

3. Undo table

subscriber
(repca1)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca1 -U postgres -X -d dst -c 'select * from undo_log order by id'
```

id	txtime	lsn	dst_schema	dst_table	undo_cmd	cmd_data	cmd_pk
1	2018-01-07 16:53:08.07825+03	0/3040998	public	cats	D		"cat_id"=>"3"
2	2018-01-07 16:53:08.07825+03	0/3040998	public	items	D		"item_id"=>"4"
3	2018-01-07 18:49:37.703375+03	0/30410A8	public	cats	D		"cat_id"=>"4"
4	2018-01-07 18:49:37.703375+03	0/30410A8	public	cats	D		"cat_id"=>"5"
5	2018-01-07 18:49:37.703375+03	0/30410A8	public	cats	D		"cat_id"=>"6"
6	2018-01-07 19:06:21.510662+03	0/3041728	public	cats	I	"cat_id"=>"4", "cat_name"=>"категория 10"	
7	2018-01-07 19:06:21.510662+03	0/3041728	public	cats	I	"cat_id"=>"5", "cat_name"=>"категория 20"	
8	2018-01-07 19:06:21.510662+03	0/3041728	public	cats	I	"cat_id"=>"6", "cat_name"=>"категория 30"	
9	2018-01-07 19:07:22.8155+03	0/3041AB8	public	cats	D		"cat_id"=>"7"
10	2018-01-07 19:07:22.8155+03	0/3041AB8	public	cats	D		"cat_id"=>"8"
11	2018-01-07 19:07:22.8155+03	0/3041AB8	public	cats	D		"cat_id"=>"9"

UNDO

(авария на provider и переключение на provider's standby)

4. Текущий LSN subscriber

subscriber
(repca1)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca1 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
```

local_id	external_id	remote_lsn	local_lsn
1	pg_16430	0/3042000	0/1722E90

UNDO

(авария на provider и переключение на provider's standby)

4. Текущий LSN subscriber

subscriber
(repca1)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca1 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
 local_id | external_id | remote_lsn | local_lsn
-----+-----+-----+-----
      1 | pg_16430    | 0/3042000 | 0/1722E90
```

5. Применить undo

subscriber
(repca1)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca1 -U postgres -X -d dst -c "select run_undo('0/3041840')"
```

ЗАМЕЧАНИЕ: last applied LSN: 0/3042000
ЗАМЕЧАНИЕ: locking public.cats ...
ЗАМЕЧАНИЕ: disable triggers public.cats
ЗАМЕЧАНИЕ: undo events with LSN >= 0/3041728
ЗАМЕЧАНИЕ: undo (11 | 0/3041AB8): DELETE FROM ONLY public.cats d WHERE d.cat_id = '9'
ЗАМЕЧАНИЕ: undo (10 | 0/3041AB8): DELETE FROM ONLY public.cats d WHERE d.cat_id = '8'
ЗАМЕЧАНИЕ: undo (9 | 0/3041AB8): DELETE FROM ONLY public.cats d WHERE d.cat_id = '7'
ЗАМЕЧАНИЕ: undo (8 | 0/3041728): INSERT INTO public.cats SELECT (populate_record(NULL::public.cats, \$1)).*
ЗАМЕЧАНИЕ: undo (7 | 0/3041728): INSERT INTO public.cats SELECT (populate_record(NULL::public.cats, \$1)).*
ЗАМЕЧАНИЕ: undo (6 | 0/3041728): INSERT INTO public.cats SELECT (populate_record(NULL::public.cats, \$1)).*
ЗАМЕЧАНИЕ: set current replayed LSN to 0/3041728
ЗАМЕЧАНИЕ: enable triggers public.cats
run_undo

6

UNDO

(авария на provider и переключение на provider's standby)

4. Текущий LSN subscriber

subscriber
(repcal)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repcal -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
 local_id | external_id | remote_lsn | local_lsn
-----+-----+-----+-----
      1 | pg_16430    | 0/3042000 | 0/1722E90
```

5. Применить undo

subscriber
(repcal)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repcal -U postgres -X -d dst -c "select run_undo('0/3041840')"
```

ЗАМЕЧАНИЕ: last applied LSN: 0/3042000
ЗАМЕЧАНИЕ: locking public.cats ...
ЗАМЕЧАНИЕ: disable triggers public.cats
ЗАМЕЧАНИЕ: undo events with LSN >= 0/3041728
ЗАМЕЧАНИЕ: undo (11 | 0/3041AB8): DELETE FROM ONLY public.cats d WHERE d.cat_id = '9'
ЗАМЕЧАНИЕ: undo (10 | 0/3041AB8): DELETE FROM ONLY public.cats d WHERE d.cat_id = '8'
ЗАМЕЧАНИЕ: undo (9 | 0/3041AB8): DELETE FROM ONLY public.cats d WHERE d.cat_id = '7'
ЗАМЕЧАНИЕ: undo (8 | 0/3041728): INSERT INTO public.cats SELECT (populate_record(NULL::public.cats, \$1)).*
ЗАМЕЧАНИЕ: undo (7 | 0/3041728): INSERT INTO public.cats SELECT (populate_record(NULL::public.cats, \$1)).*
ЗАМЕЧАНИЕ: undo (6 | 0/3041728): INSERT INTO public.cats SELECT (populate_record(NULL::public.cats, \$1)).*
ЗАМЕЧАНИЕ: set current replayed LSN to 0/3041728
ЗАМЕЧАНИЕ: enable triggers public.cats
run_undo

6

6. Включить репликацию

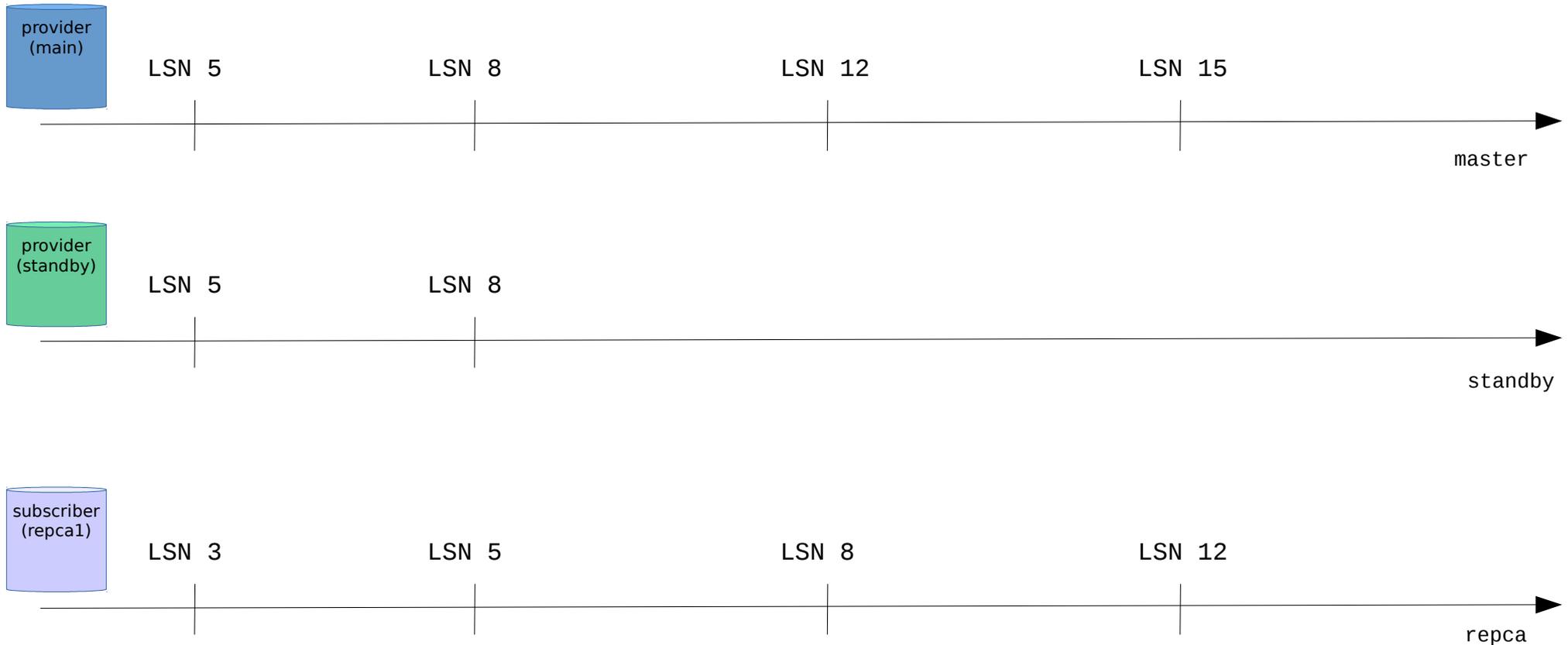
subscriber
(repcal)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repcal -U postgres -X -d dst -f- <<'EOF'  
> alter subscription repcal CONNECTION 'port=5433 dbname=src';  
> EOF  
ALTER SUBSCRIPTION
```

UNDO

(авария на provider и переключение на provider's standby)

`pg_replication_origin_session_progress(false)` - маркирует записи в undo - предыдущий lsn



UNDO

(авария на provider и переключение на provider's standby)

Несколько источников и undo

- В trigger получаем имя подписки и маркируем записи в undo логе

```
select subname
from
    pg_stat_subscription p
where
    p.pid = pg_backend_pid()
```

UNDO

(авария на provider и переключение на provider's standby)

Несколько источников и undo

- В trigger получаем имя подписки и маркируем записи в undo лог

```
select subname
from
    pg_stat_subscription p
where
    p.pid = pg_backend_pid()
```

- На publication невозможно узнать кто потребляет слот. Так как мы не знаем связи хосты подписчика <—> публикация — то мы должны держать внешний список логических подписчиков (как и для londiste), чтобы делать apply undo на них, при падении их источника.

UNDO

(авария на provider и переключение на provider's standby)

Несколько источников и undo

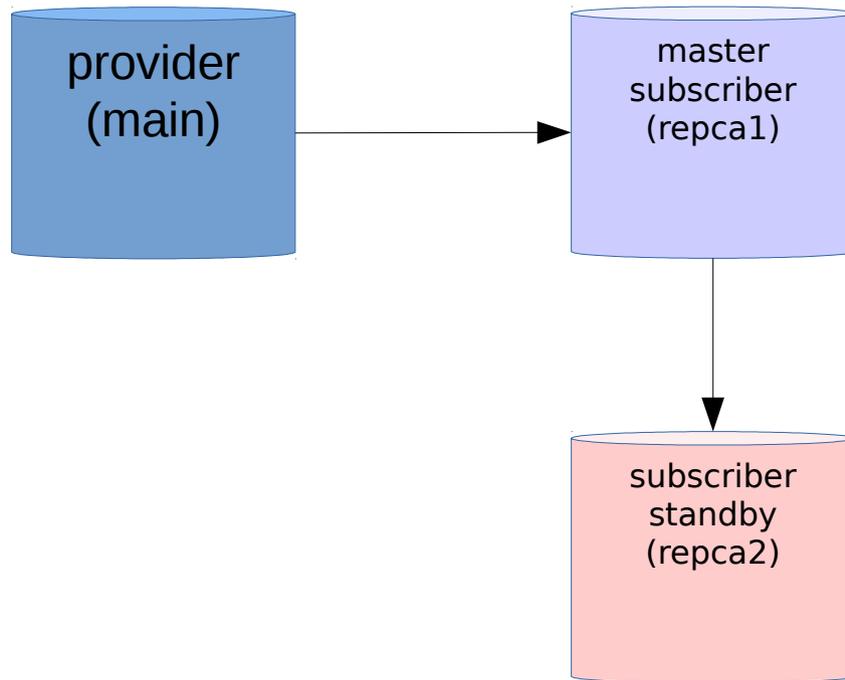
- В trigger получаем имя подписки и маркируем записи в undo лог

```
select subname
from
    pg_stat_subscription p
where
    p.pid = pg_backend_pid()
```

- На publication невозможно узнать кто потребляет слот. Так как мы не знаем связи хосты подписчика <—> публикация — то мы должны держать внешний список логических подписчиков (как и для londiste), чтобы делать apply undo на них, при падении их источника.
- Соответственно имя подписки должно быть определённого формата, что бы по нему можно было определить к какому источнику она относится

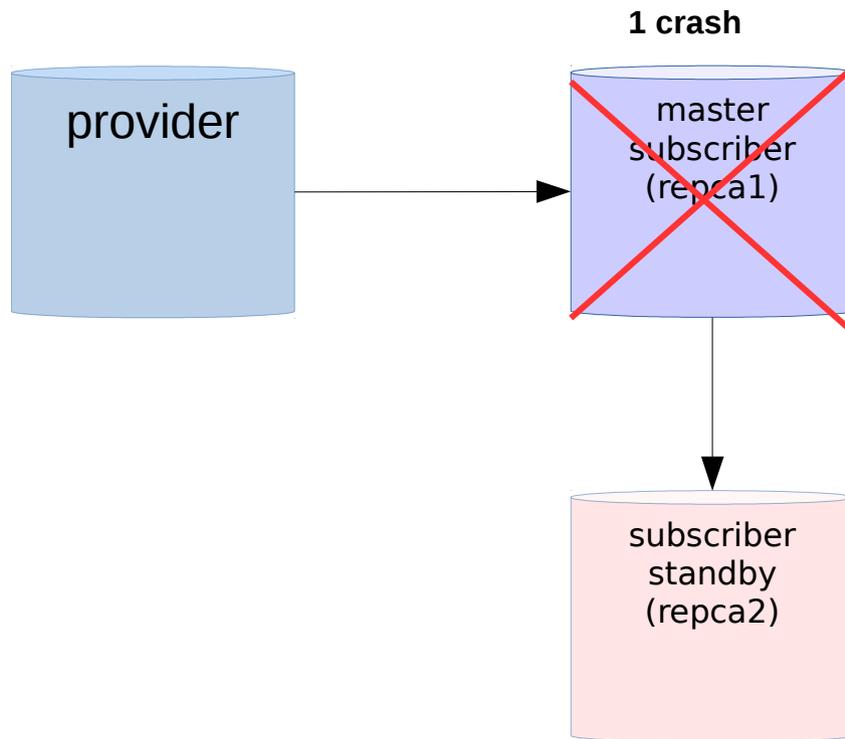
REDO

(авария subscriber и promote subscriber standby)



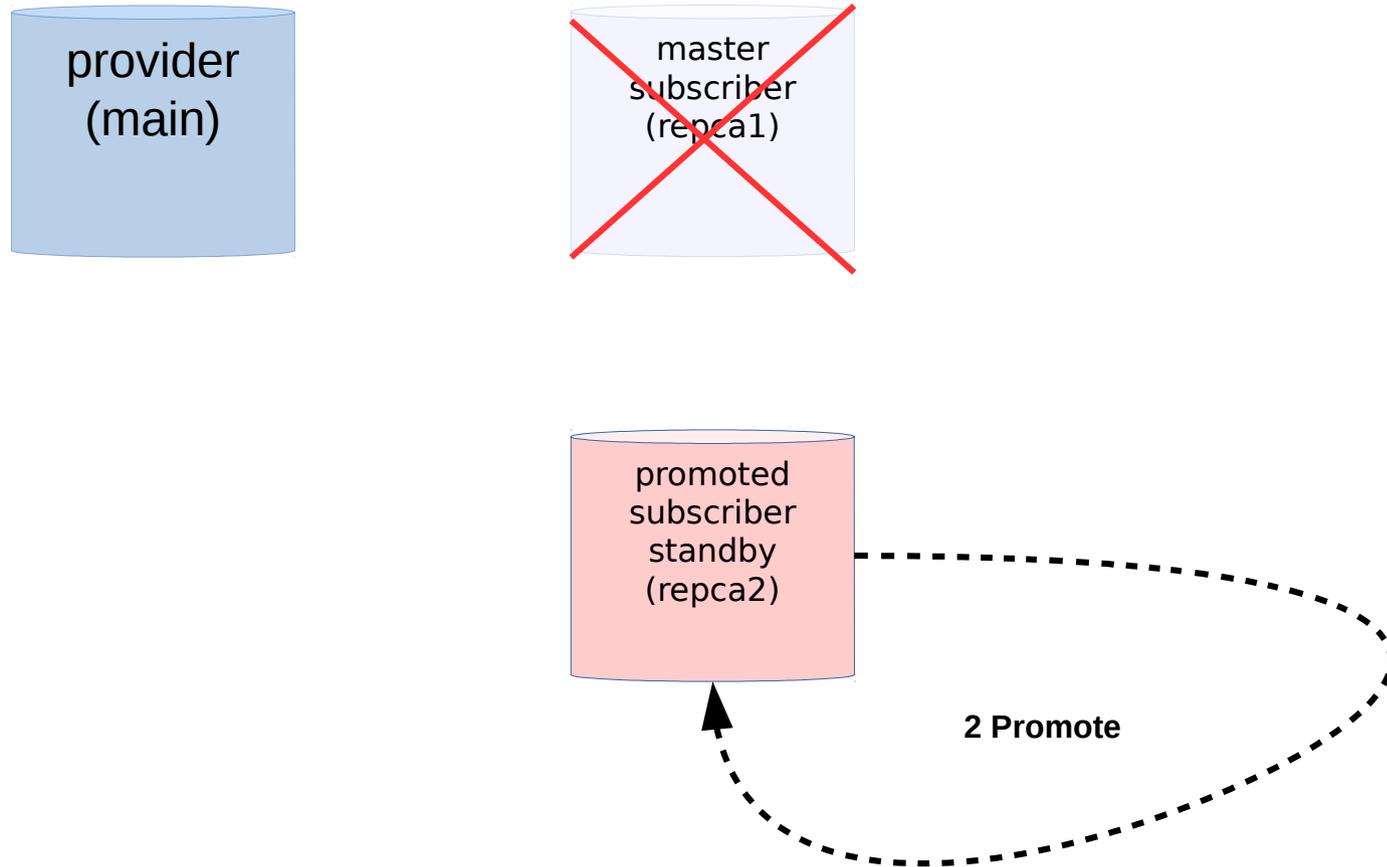
REDO

(авария subscriber и promote subscriber standby)



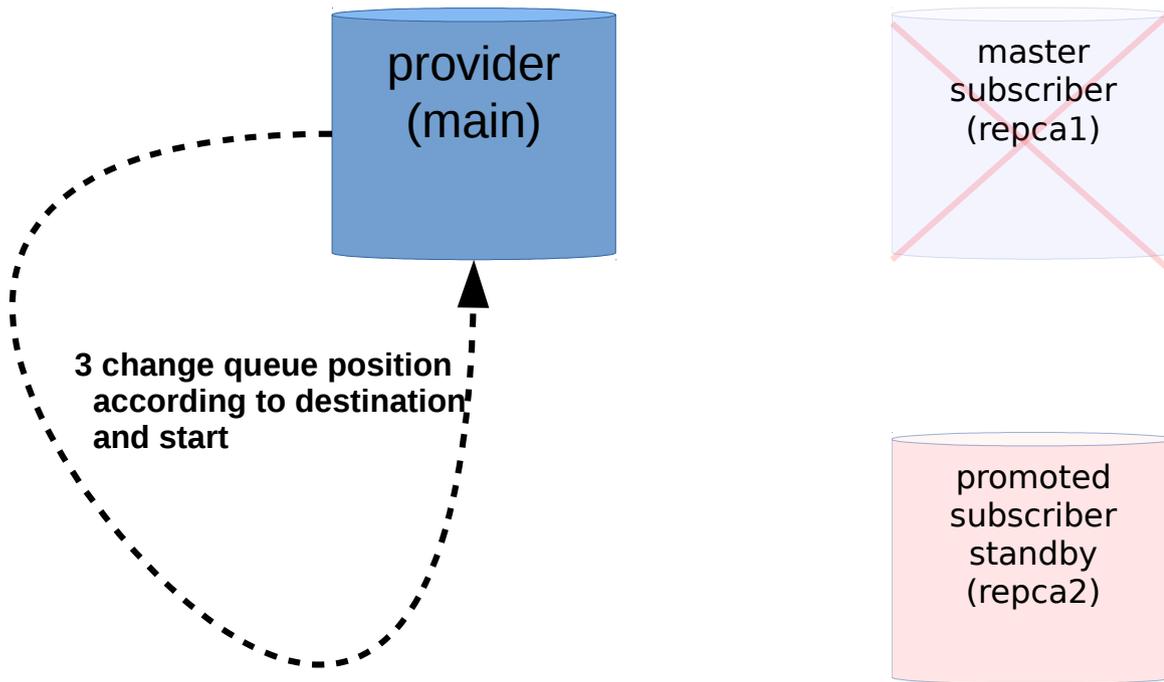
REDO

(авария subscriber и promote subscriber standby)



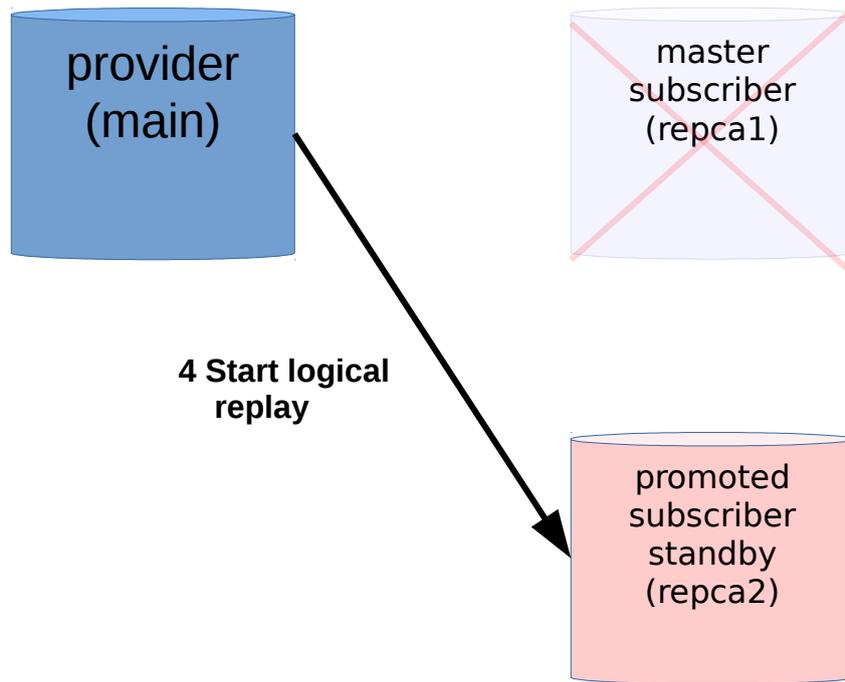
REDO

(авария subscriber и promote subscriber standby)



REDO

(авария subscriber и promote subscriber standby)



REDO

(авария subscriber и promote subscriber standby)

Команда для консумера, который будет препятствовать ротации данных:

```
psql --cluster 10/standby -U postgres -X -d src
-c "
select * from pg_logical_slot_get_binary_changes(
    'repca2'::name,
    '0/38AFCC0'::pg_lsn,
    null::int,
    variadic array['proto_version', '1', 'publication_names',
'pub' ]
)"
```

REDO

(авария subscriber и promote subscriber standby)

1. Слот который держит wal на provider для стэндбая subscriber

subscriber
standby
(repca2)

```
oghack@pghack-debian-8:~$ pg_recvlogical --cluster 10/main -U postgres -d src --create-slot -S repca2 -P pgoutput
oghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f				559 0/3037800	0/3037838
repca1	pgoutput	logical	16384	src	f	t	27782			559 0/3037800	0/3037838
standby		physical			f	t	27621			0/3037838	

REDO

(авария subscriber и promote subscriber standby)

1. Слот который держит wal на provider для стэндбая subscriber

subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ pg_recvlogical --cluster 10/main -U postgres -d src --create-slot -S repca2 -P pgoutput
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalogxmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f				559 0/3037800	0/3037838
repca1	pgoutput	logical	16384	src	f	t	27782			559 0/3037800	0/3037838
standby		physical			f	t	27621			0/3037838	

2. Добавим запись на источник

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -1 -X -d src -f- <<'EOF'
```

```
> insert into cats (cat_name) values ('категория 3');
```

```
> EOF
```

```
INSERT 0 1
```

REDO

(авария subscriber и promote subscriber standby)

1. Слот который держит wal на provider для стэндбая subscriber

subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ pg_recvlogical --cluster 10/main -U postgres -d src --create-slot -S repca2 -P pgoutput
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f				559 0/3037800	0/3037838
repca1	pgoutput	logical	16384	src	f	t	27782			559 0/3037800	0/3037838
standby		physical			f	t	27621				0/3037838

2. Добавим запись на источник

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -l -X -d src -f- <<'EOF'
> insert into cats (cat_name) values ('категория 3');
> EOF
INSERT 0 1
```

3. Проверим pg_replication_origin status на subscriber и стэндбая subscriber

subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
```

local_id	external_id	remote_lsn	local_lsn
1	pg_16415	0/3037B68	0/30C1BE8

(1 строка)

subscriber
(repca1)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca1 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
```

local_id	external_id	remote_lsn	local_lsn
1	pg_16415	0/3037B68	0/30C1BE8

(1 строка)

REDO

(авария subscriber и promote subscriber standby)

4. Слот для стэндбая подписчика в прошлом

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f			559	0/3037800	0/3037838
repca1	pgoutput	logical	16384	src	f	t	27782		560	0/3037B68	0/3037BA0
standby		physical			f	t	27621			0/3037BA0	

(3 строки)

REDO

(авария subscriber и promote subscriber standby)

4. Слот для стэндбая подписчика в прошлом

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f			559	0/3037800	0/3037838
repca1	pgoutput	logical	16384	src	f	t	27782		560	0/3037B68	0/3037BA0
standby		physical			f	t	27621			0/3037BA0	

(3 строки)

5. "Двигаем" слот с помощью SQL протокола

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c "select * from pg_logical_slot_get_binary_changes('repca2'::name, '0/3037B68'::pg_lsn, null::int, variadic array['proto_version', '1', 'publication_names', 'pub'])"
```

lsn	xid	data
0/3037838	559	\x420000000003037b38000205949a2d1e2a0000022f
0/30378A0	559	\x52000040037075626c6963006361747300640002016361745f69640000000017ffffff006361745f6e616d650000000019ffffff
0/30378A0	559	\x49000040034e00027400000001337400000014d0bad0b0d182d0b5d0b3d0bed180d0b8d18f2033
0/3037B68	559	\x43000000000003037b380000000003037b68000205949a2d1e2a

(4 строки)

REDO

(авария subscriber и promote subscriber standby)

4. Слот для стендбая подписчика в прошлом

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f				559 0/3037800	0/3037838
repca1	pgoutput	logical	16384	src	f	t	27782			560 0/3037B68	0/3037BA0
standby		physical			f	t	27621			0/3037BA0	

(3 строки)

5. "Двигаем" слот с помощью SQL протокола

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c "select * from pg_logical_slot_get_binary_changes('0/3037B68',:pg_lsn, null::int, variadic array['proto_version', '1', 'publication_names', 'pub'])"
```

lsn	xid	data
0/3037838	559	\x4200000000003037b38000205949a2d1e2a0000022f
0/30378A0	559	\x520000400037075626c6963006361747300640002016361745f69640000000017ffffff006361745f6e616d650000000019ffffff
0/30378A0	559	\x490000400034e00027400000001337400000014d0bad0b0d182d0b5d0b3d0bed180d0b8d18f2033
0/3037B68	559	\x43000000000003037b380000000003037b68000205949a2d1e2a

(4 строки)

6. LSN совпал

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f				559 0/3037800	0/3037B68
repca1	pgoutput	logical	16384	src	f	t	27782			560 0/3037B68	0/3037BA0
standby		physical			f	t	27621			0/3037BA0	

(3 строки)

REDO

(авария subscriber и promote subscriber standby)

7. “Эмулируем задержку стэндбая subscriber”

subscriber
(repca1)

```
pghack@pghack-debian-8:~$ sudo pg_ctlcluster 10/repca1 reload
pghack@pghack-debian-8:~$ psql --cluster 10/repca1 -U postgres -X -d dst -c "select pg_terminate_backend(active_pid) from pg_replication_slots where slot_name = 'repca2'"
pg_terminate_backend
-----
t
```

REDO

(авария subscriber и promote subscriber standby)

7. “Эмулируем задержку стэндбая subscriber”

subscriber
(repca1)

```
pghack@pghack-debian-8:~$ sudo pg_ctlcluster 10/repca1 reload
pghack@pghack-debian-8:~$ psql --cluster 10/repca1 -U postgres -X -d dst -c "select pg_terminate_backend(active_pid) from pg_replication_slots where slot_name = 'repca2'"
pg_terminate_backend
-----
t
```

8. Делаем еще 1 запись

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -1 -X -d src -f- <<'EOF'
> insert into cats (cat_name) values ('категория 4');
> EOF
INSERT 0 1
```

REDO

(авария subscriber и promote subscriber standby)

7. “Эмулируем задержку стэндбая subscriber”

subscriber
(repca1)

```
pghack@pghack-debian-8:~$ sudo pg_ctlcluster 10/repca1 reload
pghack@pghack-debian-8:~$ psql --cluster 10/repca1 -U postgres -X -d dst -c "select pg_terminate_backend(active_pid) from pg_replication_slots where slot_name = 'repca2'"
pg_terminate_backend
-----
t
```

8. Делаем еще 1 запись

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -1 -X -d src -f- <<'EOF'
> insert into cats (cat_name) values ('категория 4');
> EOF
INSERT 0 1
```

9. Стэндбай subscriber отстает

subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from cats'
 cat_id | cat_name
-----+-----
      1 | категория 1
      2 | категория 2
      3 | категория 3
(3 строки)
```

subscriber
(repca1)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca1 -U postgres -X -d dst -c 'select * from cats'
 cat_id | cat_name
-----+-----
      1 | категория 1
      2 | категория 2
      3 | категория 3
      4 | категория 4
(4 строки)
```

REDO

(авария subscriber и promote subscriber standby)

10. Авария на мастере subscriber, стендбай все еще отстают

provider
(main)

```
pghack@pghack-debian-8:~$ sudo pg_ctlcluster -m immediate 10 repca1 stop -- -m immediate
Notice: extra pg_ctl/postgres options given, bypassing systemctl for stop operation
pghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from cats'
 cat_id |  cat_name
-----+-----
      1 | категория 1
      2 | категория 2
      3 | категория 3
(3 строки)
```

subscriber
standby
(repca2)

REDO

(авария subscriber и promote subscriber standby)

10. Авария на мастере subscriber, стендбай все еще отстает

provider
(main)

```
pghack@pghack-debian-8:~$ sudo pg_ctlcluster -m immediate 10 repca1 stop -- -m immediate
Notice: extra pg_ctl/postgres options given, bypassing systemctl for stop operation
pghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from cats'
 cat_id |  cat_name
-----+-----
       1 | категория 1
       2 | категория 2
       3 | категория 3
(3 строки)
```

subscriber
standby
(repca2)

11. Удалили слот для аварийного мастера subscriber

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c "select pg_drop_replication_slot('repca1')"
```

pg_drop_replication_slot

(1 строка)

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn	
repca2	pgoutput	logical	16384	src	f	f				559	0/3037800	0/3037B68
standby		physical			f	t	27621				0/3038068	

REDO

(авария subscriber и promote subscriber standby)

12. Проверяем pg_replication_origin_status

subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
```

local_id	external_id	remote_lsn	local_lsn
1	pg_16415	0/3037B68	0/30C1BE8

REDO

(авария subscriber и promote subscriber standby)

12. Проверяем pg_replication_origin_status

subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
 local_id | external_id | remote_lsn | local_lsn
-----+-----+-----+-----
      1 | pg_16415    | 0/3037B68 | 0/30C1BE8
```

13. “Докручиваем слот”

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c "select * from pg_logical_slot_get_binary_changes('repca2'::name, '0/3037B68'::pg_lsn, null::int, variadic array['proto_version', '1', 'publication_names', 'pub'])"
 lsn | xid | data
-----+-----+-----
(0 строк)
```

REDO

(авария subscriber и promote subscriber standby)

12. Проверяем pg_replication_origin_status

subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
 local_id | external_id | remote_lsn | local_lsn
-----+-----+-----+-----
      1 | pg_16415   | 0/3037B68 | 0/30C1BE8
```

13. “Докручиваем слот”

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c "select * from pg_logical_slot_get_binary_changes('repca2'::name, '0/3037B68'::pg_lsn, null::int, variadic array['proto_version', '1', 'publication_names', 'pub'])"
 lsn | xid | data
-----+-----+-----
(0 строк)
```

14. в таблице не появилось новых данных

subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from cats'
 cat_id | cat_name
-----+-----
      1 | категория 1
      2 | категория 2
      3 | категория 3
```

REDO

(авария subscriber и promote subscriber standby)

15. Промотируем subscriber standby

promoted
subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ sudo pg_ctlcluster 10 repca2 promote
pghack@pghack-debian-8:~$ sudo tail /var/log/postgresql/postgresql-10-main.log
2018-01-12 18:29:06.275 MSK [28314] postgres@src ПОДРОБНОСТИ: передача транзакций, фиксируемых после 0/3037B68, чтение WAL с 0/3037800
2018-01-12 18:29:06.275 MSK [28314] postgres@src ОПЕРАТОР: select * from pg_logical_slot_get_binary_changes('repca2'::name, '0/3037B68'::pg_lsn, nu
ll::int, variadic array['proto version', '1', 'publication_names', 'pub'])
2018-01-12 18:29:06.275 MSK [28314] postgres@src СООБЩЕНИЕ: процесс логического декодирования достиг точки согласованности в 0/3037800
2018-01-12 18:29:06.275 MSK [28314] postgres@src ПОДРОБНОСТИ: Больше активных транзакций нет.
2018-01-12 18:29:06.275 MSK [28314] postgres@src ОПЕРАТОР: select * from pg_logical_slot_get_binary_changes('repca2'::name, '0/3037B68'::pg_lsn, nu
ll::int, variadic array['proto version', '1', 'publication_names', 'pub'])
2018-01-12 18:29:39.832 MSK [28346] postgres@src ОШИБКА: слот репликации "repca1" не существует
2018-01-12 18:29:44.846 MSK [28350] postgres@src ОШИБКА: слот репликации "repca1" не существует
2018-01-12 18:29:49.854 MSK [28353] postgres@src ОШИБКА: слот репликации "repca1" не существует
2018-01-12 18:29:54.867 MSK [28357] postgres@src ОШИБКА: слот репликации "repca1" не существует
2018-01-12 18:29:59.879 MSK [28359] postgres@src ОШИБКА: слот репликации "repca1" не существует
pghack@pghack-debian-8:~$ sudo tail /var/log/postgresql/postgresql-10-repca2.log
2018-01-12 18:30:19.928 MSK [27677] СООБЩЕНИЕ: рабочий процесс: logical replication worker for subscription 16415 (PID 28374) завершился с кодом вы
хода 1
2018-01-12 18:30:24.935 MSK [28378] СООБЩЕНИЕ: запускается применяющий процесс логической репликации для подписки "repca1"
2018-01-12 18:30:24.938 MSK [28378] ОШИБКА: не удалось начать трансляцию WAL: ОШИБКА: слот репликации "repca1" не существует
2018-01-12 18:30:24.940 MSK [27677] СООБЩЕНИЕ: рабочий процесс: logical replication worker for subscription 16415 (PID 28378) завершился с кодом вы
хода 1
2018-01-12 18:30:29.948 MSK [28380] СООБЩЕНИЕ: запускается применяющий процесс логической репликации для подписки "repca1"
2018-01-12 18:30:29.951 MSK [28380] ОШИБКА: не удалось начать трансляцию WAL: ОШИБКА: слот репликации "repca1" не существует
2018-01-12 18:30:29.952 MSK [27677] СООБЩЕНИЕ: рабочий процесс: logical replication worker for subscription 16415 (PID 28380) завершился с кодом вы
хода 1
2018-01-12 18:30:34.960 MSK [28384] СООБЩЕНИЕ: запускается применяющий процесс логической репликации для подписки "repca1"
2018-01-12 18:30:34.963 MSK [28384] ОШИБКА: не удалось начать трансляцию WAL: ОШИБКА: слот репликации "repca1" не существует
2018-01-12 18:30:34.964 MSK [27677] СООБЩЕНИЕ: рабочий процесс: logical replication worker for subscription 16415 (PID 28384) завершился с кодом вы
хода 1
```

REDO

(авария subscriber и promote subscriber standby)

16. Подписчик все еще в “прошлом”

promoted
subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from cats'
 cat_id | cat_name
-----+-----
      1 | категория 1
      2 | категория 2
      3 | категория 3
(3 строки)
```

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from cats'
 cat_id | cat_name
-----+-----
      1 | категория 1
      2 | категория 2
      3 | категория 3
      4 | категория 4
(4 строки)
```

REDO

(авария subscriber и promote subscriber standby)

16. Подписчик все еще в “прошлом”

promoted
subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from cats'
 cat_id | cat_name
-----+-----
      1 | категория 1
      2 | категория 2
      3 | категория 3
(3 строки)
```

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from cats'
 cat_id | cat_name
-----+-----
      1 | категория 1
      2 | категория 2
      3 | категория 3
      4 | категория 4
(4 строки)
```

17. Меняем слот

promoted
subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -f- <<'EOF'
> alter subscription repca1 set (slot_name = repca2);
> EOF
ALTER SUBSCRIPTION
```

REDO

(авария subscriber и promote subscriber standby)

18. Слот стал активным и стал потребляться

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
реpsa2	pgoutput	logical	16384	src	f	t	28579		561	0/3038030	0/3038148
standby		physical			f	t	27621			0/3038148	

(2 строки)

REDO

(авария subscriber и promote subscriber standby)

18. Слот стал активным и стал потребляться

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	t	28579		561	0/3038030	0/3038148
standby		physical			f	t	27621			0/3038148	

(2 строки)

19. Subscriber получил изменения

promoted
subscriber
standby
(repca2)

```
pghack@pghack-debian-8:~$ psql --cluster 10/repca2 -U postgres -X -d dst -c 'select * from pg_replication_origin_status'
```

local_id	external_id	remote_lsn	local_lsn
1	pg_16415	0/3038030	0/30C2830

(1 строка)

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from cats'
```

cat_id	cat_name
1	категория 1
2	категория 2
3	категория 3
4	категория 4

REDO

Алгоритм потребления слота для стэндбая подписчика:

- Получить состояние слота для мастера подписчика (pg_replications_slots)

provider
(main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f			559	0/30378A0	0/3037B68
repca1	pgoutput	logical	16384	src	f	t	27782		560	0/3037B68	0/3037BA0
standby		physical			f	t	27621			0/3037BA0	

REDO

Алгоритм потребления слота для стэндбая подписчика:

- Получить состояние слота для мастера подписчика (pg_replications_slots)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f			559	0/30378A0	0/3037B68
repca1	pgoutput	logical	16384	src	f	t	27782		560	0/3037B68	0/3037BA0
standby		physical			f	t	27621			0/3037BA0	

- Проверить pg_replication_origin на мастере подписчика

provider
(main)

subscriber
(repca1)

REDO

Алгоритм потребления слота для стэндбая подписчика:

- Получить состояние слота для мастера подписчика (pg_replications_slots)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f				559 0/30378A0	0/3037B68
repca1	pgoutput	logical	16384	src	f	t	27782			560 0/3037B68	0/3037BA0
standby		physical			f	t	27621			0/3037BA0	

provider
(main)

subscriber
(repca1)

subscriber
standby
(repca2)

- Проверить pg_replication_origin на мастере подписчика
- Дождаться позиции pg_replication_origin мастера подписчика на стэндбае подписчика

REDO

Алгоритм потребления слота для стендбэя подписчика:

- Получить состояние слота для мастера подписчика (pg_replications_slots)

provider (main)

```
pghack@pghack-debian-8:~$ psql --cluster 10/main -U postgres -X -d src -c 'select * from pg_replication_slots'
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn
repca2	pgoutput	logical	16384	src	f	f			559	0/30378A0	0/3037B68
repca1	pgoutput	logical	16384	src	f	t	27782		560	0/3037B68	0/3037BA0
standby		physical			f	t	27621			0/3037BA0	

- Проверить pg_replication_origin на мастере подписчика
- Дождаться позиции pg_replication_origin мастера подписчика на стендбэе подписчика

- Потребить слот для стендбэя подписчика

provider (main)

```
select * from pg_logical_slot_get_binary_changes('repca2'::name,  
'0/3037B68'::pg_lsn, null::int, variadic array['proto_version', '1',  
'publication_names', 'pub'])"
```

	SkyTools	PostgreSQL10
Undo	OK	OK
		To do 1: простая реализации из коробки
		To do 2: Logical Slot на standby(Слот не реплицируется на стендбай), дырка после promote, если трафик пойдет
Redo	OK	OK
		To do 3: Прогресс потребления слота на подписчике
		To do 4: Нет проверки позиции потребления между subscriber и provider
		To do 5: Чтобы pg_replication_origin_advance сдвигал слот на src
Пересоздание subscriber из другого subscriber	OK	OK
		To do 6: pg_replication_origin не транзакционный
		To do 7: pg_dump (replication_origin, pg_subscription)
Event tracking	OK	OK
		overhead на proxy таблицу/ писать custom decoder + pg_logical_emit_message
Unlogged table	OK	-

Creating a subscription that connects to the same database cluster

<https://www.postgresql.org/docs/10/static/sql-createsubscription.html>

To make this work, create the replication slot separately (using the function `pg_create_logical_replication_slot` with the plugin name `pgoutput`) and create the subscription using the parameter `create_slot = false`. This is an implementation restriction that might be lifted in a future release.

Creating a subscription that connects to the same database cluster

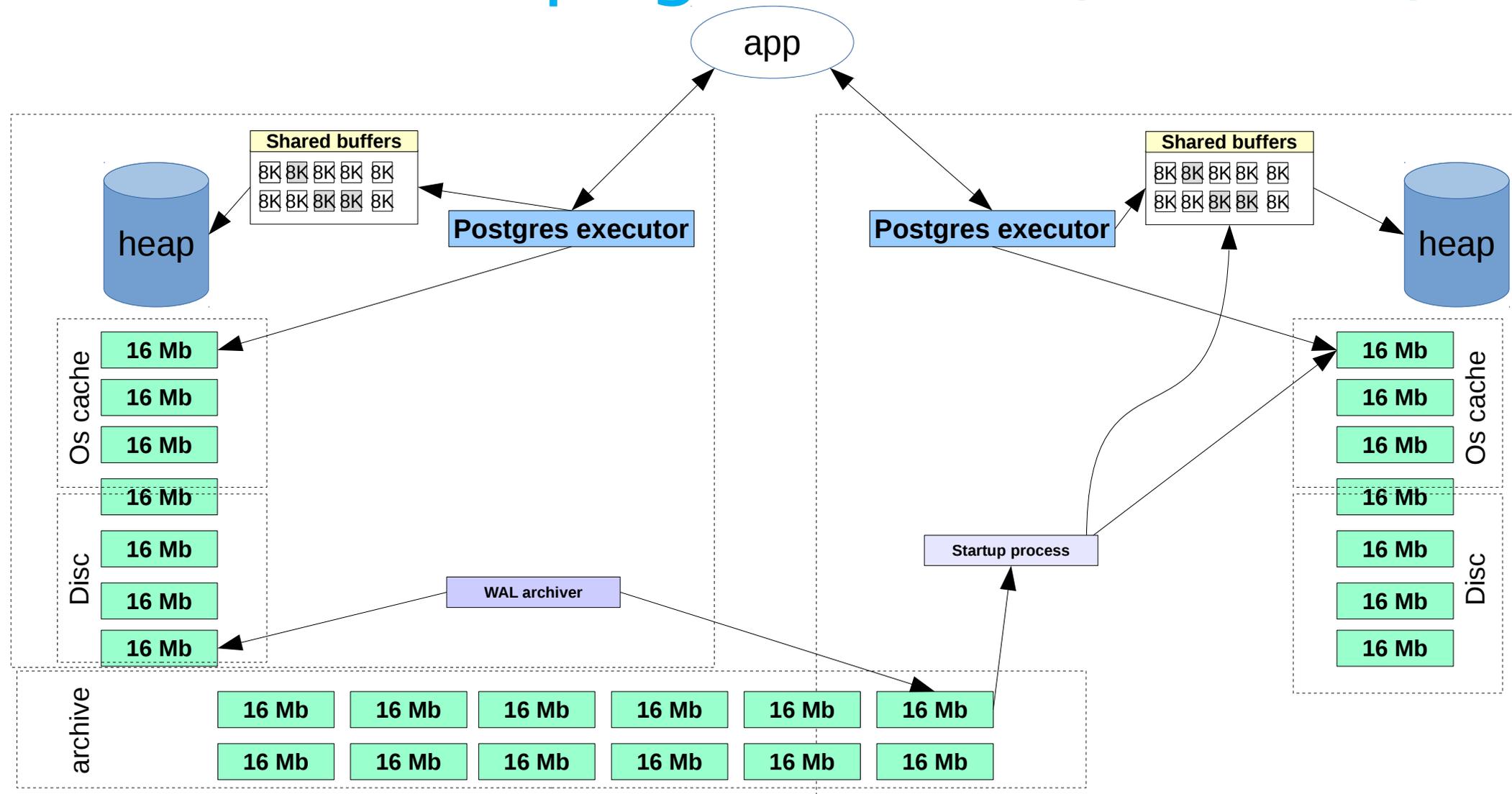
<https://www.postgresql.org/docs/10/static/sql-createsubscription.html>

To make this work, create the replication slot separately (using the function `pg_create_logical_replication_slot` with the plugin name `pgoutput`) and create the subscription using the parameter `create_slot = false`. This is an implementation restriction that might be lifted in a future release.

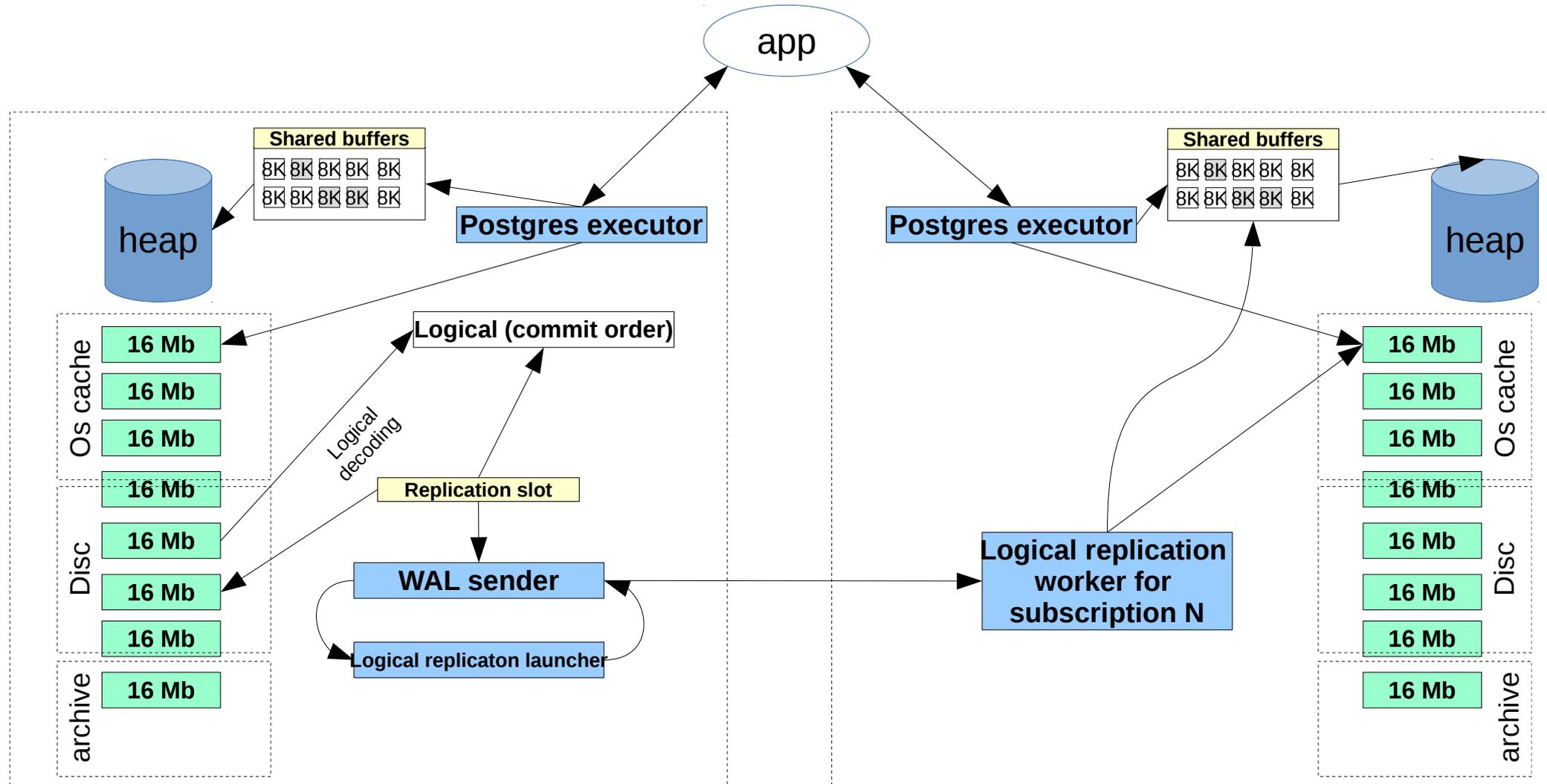
Мы подумали и сделали

- provider и subscriber на 1 базе
- Добавили триггер – получили некий event loop, который можно использовать для “размазывания нагрузки”, обработки очередей и тд

WALs out of page cache (archive)



WALs out of page cache (logical)



Спасибо за внимание!

Вопросы?



<https://github.com/avito-tech>

kevteev@avito.ru