PostgreSQL Backups
the Modern Way

PGConf.ASIA 2016
Tokyo, Japan

Magnus Hagander
magnus@hagander.net
Magnus Hagander

• Redpill Linpro
  ▪ Infrastructure services
  ▪ Principal database consultant

• PostgreSQL
  ▪ Core Team member
  ▪ Committer
  ▪ PostgreSQL Europe
So, backups...

- Do you make them?
Backups

- Are *not* superseded by replication
- Or cloud
- Or containers
- ..
Backups

• Are boring
• But I'm glad you have them
Backups

• When did you last restore?
Ok, enough generic
What about backups in PostgreSQL?
pg_dump options:

-FC = custom format
-Z = compression
-j = parallel
-a = data only, -s = schema only
-n = schema, -t = table
...
pg_dump

- Don't use for backups
  - Has other good use cases
- Too slow to restore
- Too much overhead
- No PITR
- Exceptions, of course
Physical backups

- Base backups
- With or without log archive
- Fast restore
- Full cluster only
- Platform specific
#!/bin/bash
set -e

psql -U postgres -q "SELECT pg_start_backup('foo')"

```
tar cfz /backup/$ (date +%Y%m%d).tar.gz /var/lib/pgsql/data
```

```
psql -U postgres -q "SELECT pg_stop_backup()"
```
Base backups

- So many ways to get that wrong
  - Spot one?
Base backups

- This used to be the only way
- Many scripts around that does it
- Many of those are broken...
pg_basebackup

- Base backup over replication protocol
- Safe
- Error handling and recovery
- For most cases
  - (we'll cover other options later)
#!/bin/bash
set -e

pg_basebackup -D /backup/$\{(date +\%Y\%m\%d)\} -Ft -x
Needs replication

- Defaults need to change
- But for now:

```sql
wal_level = hot_standby
max_wal_senders = 5
```

```text
local  replication  postgres  peer
```
Backup formats

- **plain**
  - Safe copy of data directory
  - Not good with multiple tablespaces
- **tar**
  - Destination still a directory
  - Each tablespace gets one file
    - base.tar
Transaction log

- xlog required to restore backup
- From beginning of backup to end
- In the log archive, right?
Including xlog

- Always use `-x` or `-X` to include xlog
- Makes backup *independently consistent*
  - With or without log archive
  - May back up xlog twice
- Use even with log archive!
Including xlog

-X fetch
- Fetches xlog at end of backup
- Can fail if xlog rotated

-X stream
- Replicates xlog over secondary connection
- Fewer failure scenarios
- Does not work with tar (until version 10)
Backup compression

```
pg_basebackup -Z
```

- Compression happens in `pg_basebackup`
- Tar format only
- CPU usage
- Remote server?
Transfer compression

- SSL compression
  - Much harder these days
- ssh tunneling

```
ssh mydbserver -c "pg_basebackup -Ft -D -Z9" > backup.tgz
```
That's it!

- With that, you have backups
- That work
- And are (reasonably) safe
PITR

- Point in time recovery
- You all want it
- A bit more setting up
To use PITR, we use log archiving like this?

```
archive_command =
    'test ! -f /mnt/archivedir/%f && cp %p /mnt/archivedir/%f'
```
Don't do that!
pg_receivexlog

- Runs on archive server
- Uses streaming replication
- Generates log archive
pg_receivexlog

- More granular recovery
- Safe against server restarts
- Can follow timeline switches on master
**pg_receivexlog**

- *Always* use with replication slot
  - As of 9.4
  - But we said modern..
- Backups *should* block
Ensure it's restarted!
Backup retention

- How long to keep around?
- What granularity?
- ...

Backup retention

- Recovery needs:
  - Base backup
  - All xlog from start to end
  - All xlog from end to pitr
- (that's why we use -x!)
Backup retention

- `find` is often enough
- Delete logs older than X, base older than Y
  - Safe if `-x` was used!

```bash
#!/bin/bash

find /var/backups/basebackup -type f -mtime +30 -print0 | xargs -0 -r /bin/rm

find /var/backups/xlog -type f -mtime +7 -print0 | xargs -0 -r /bin/rm
```
Not enough?

- Handles the simple cases
- But has limitations
- Particularly in management
Other tools

- Barman
- pgBackRest
Barman

- Backup scheduling
- Log archiving
- Retention management
- Multi-server
- Restore shortcuts
Barman

- Developed by 2ndQuadrant
- Python
- GPLv3
- Primarily ssh+rsync
  - 1.6 learned about pg_receivexlog!
  - 2.0 learned about pg_basebackup
    - Before that, no (safe) concurrent backup support
pgBackRest

- Backup scheduling
- Log archiving
- Retention management
- Multi-server
- Restore shortcuts
pgBackRest

- Developed by CrunchyData
- Perl
- MIT license
- ssh but not rsync
pgBackRest

- Custom protocol
- Parallel backup sessions
- Full/Differential/Incremental
  - Segment based
pgBackRest

- No pg_receivexlog support
- No concurrent backup support
- Yet
Summary
Don't roll your own!
Don't roll your own

- Too many pitfalls
- Both base backups and archiving
- Backups are too important!
Don't roll your own

- Primary choice
  - Built-in
  - If it's enough
- Secondary choice
  - pgBackRest
  - Barman
- Tertiary choice
  - Restart from top of slide
Thank you!

Magnus Hagander
magnus@hagander.net
@magnushagander
http://www.hagander.net/talks/