Efficiently Backing up Terabytes of Data with pgBackRest

David Steele

LISA 2015
November 11, 2015
About the Speaker

- Senior Data Architect at Crunchy Data Solutions, the PostgreSQL company for secure enterprises.
- Actively developing with PostgreSQL since 1999.
Agenda

- Why Backup?
- Living Backups
- How to Backup?
- pgBackRest Design
- Performance
- Demo
Why Backup?

- Hardware Failure
  - No amount of redundancy can prevent it
- Replication
  - WAL archive for when async streaming gets behind
  - Sync replica from backup instead of master
- Corruption
  - Can be caused by hardware or software
  - Detection is of course a challenge
- Accidents
  - So you dropped a table?
  - Deleted your most important account?
Why Backup? - Continued

- **Development**
  - No more realistic data than production!
  - May not be practical due to size / privacy issues

- **Reporting**
  - Use backups to standup an independent reporting server

- **Forensics**
  - Recover important data that was removed on purpose
Schrödinger’s Backup

The state of any backup is unknown until a restore is attempted.
Living Backups

• Find a way to use your backups:
  • Syncing / New Replicas
  • Offline reporting
  • Offline data archiving
  • Development

• Unused code paths will not work when you need them unless they are tested:
  • Regularly scheduled automated failover using backups to restore the old primary
  • Regularly scheduled disaster recovery (during a main window if possible) to test restore techniques
How to Backup?

• `pg_dump`
• `pg_basebackup`
• Manual
• ThirdParty
  • OmniPITR
  • Barman
  • WAL-E

• `pgBackRest`?
pgBackRest Design - Say No to Rsync

• Rsync powers many database backup solutions but it has some serious limitations:
  • Single-threaded
  • One second timestamp resolution
  • No destination compression
  • Incremental backups require previous backup to be uncompressed.

• pgBackRest does not use rsync, tar or any other tools of that type:
  • Protocol supports local/remote operation
  • Solves timestamp resolution issue
pgBackRest Design - Features

- Compression is performed and checksums are calculated in-stream
- Asynchronous compression and transfer for WAL archiving
- Remote or local operation
- Threading for parallel compression and transfer
- Full, differential, and incremental support
- Backup and archive expiration policies
- Resumable backups
- Optional hard-linking of diff and incr backups
- Works with PostgreSQL >= 8.3
pgBackRest Design - Backup Structure

- Clear simple structure
- Plaintext manifest
- Valid Postgres data directory
- Postgres can be started in the backup directory if no compression is used
- Archive logs needed to make the backup consistent can optionally be copied to pg_xlog (no need to use recovery.conf or have access to the archive logs)
### pgBackRest Performance vs Rsync

<table>
<thead>
<tr>
<th>Parameters</th>
<th>PgBackRest</th>
<th>Rsync</th>
</tr>
</thead>
<tbody>
<tr>
<td>threads: 1</td>
<td>141.0 seconds</td>
<td>124.5 seconds</td>
</tr>
<tr>
<td>network compression: l3</td>
<td></td>
<td>.13X Faster</td>
</tr>
<tr>
<td>destination compression: none</td>
<td></td>
<td></td>
</tr>
<tr>
<td>threads: 2</td>
<td>84.1 seconds</td>
<td>N/A</td>
</tr>
<tr>
<td>network compression: l3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>destination compression: none</td>
<td>1.48X Faster</td>
<td></td>
</tr>
<tr>
<td>(than 1 rsync thread)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>threads: 1</td>
<td>334.4 seconds</td>
<td>510.3 seconds</td>
</tr>
<tr>
<td>network compression: l6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>destination compression: l6</td>
<td>1.52X Faster</td>
<td></td>
</tr>
<tr>
<td>threads: 2</td>
<td>174.4 seconds</td>
<td>N/A</td>
</tr>
<tr>
<td>network compression: l6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>destination compression: l6</td>
<td>2.93X Faster</td>
<td></td>
</tr>
<tr>
<td>(than 1 rsync thread)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Do you think they backup?
Demo Time!

• Live Demo, this will be fun...
Thank You! Questions?

website: www.pgbackrest.org

email: david@pgbackrest.org
email: david@crunchydata.com

release page: https://github.com/pgmasters/backrest/releases

slides & demo: https://github.com/dwsteele/conference/releases