PostgreSQL (System) Administration

SCALE12X - 2014
Los Angeles, CA

Stephen Frost
sfrost@snowman.net
Stephen Frost

- PostgreSQL
  - Major Contributor, Committer
  - Implemented Roles in 8.3
  - Column-Level Privileges in 8.4
  - Contributions to PL/pgSQL, PostGIS
- Resonate, Inc.
  - Principal Database Engineer
  - Online Digital Media Company
- We're Hiring! - techjobs@resonateinsights.com
Do you read...

• planet.postgresql.org
Agenda

• Terms
• Installation
• Initial configuration
• Getting connected
• Users / Roles
• Permissions
• Backups
• Monitoring
• Extensions
Terms

- "Cluster" ; aka "Instance"
  - One PG server
  - one "postmaster" - listens on one port
  - One set of data files (including tablespaces)
  - Users/Roles and tablespaces at cluster level
  - Replication at cluster level
Terms (continued)

• "Tablespace"
  • Alternate directory/filesystem for PG to store data
  • Perms must be 0700, owned by postgres
  • Must explicitly GRANT create rights
  • Can contain objects from any database

• "Database"
  • Lives inside a cluster
  • Schemas at the database level

• "Schema"
  • Lives inside a database
• Tables, views, functions at the schema level
• Default 'public' schema allows anyone to create
Installation

• Debian/Ubuntu/etc
  • apt.postgresql.org
  • Add PGDG sources.list.d
• RedHat/CentOS/etc
  • yum.postgresql.org
  • Download & Install PGDG RPM
• Multiple Major Versions
Debian Install

- Configs in /etc/postgresql/X.Y/main/
- Initial DB in /var/lib/postgresql/X.Y/main
- Binaries into /usr/lib/postgresql/X.Y/bin
- Logs into /var/log/postgresql/
- Startup logs in /var/log/postgresql also
- One init script starts all major versions
Debian "Clusters"

• Debian provides wrappers and helper scripts
• `pg_lsclusters` - lists all PG clusters
• `pg_ctlcluster` - Control specific clusters
• `--cluster` option - Specify specific cluster
  • `psql --cluster 9.2/main`
  • `pg_dump --cluster 9.2/main`, etc ...
RedHat Install

• Configs in data directory
• Default DB in /var/lib/pgsql/X.Y/data
• Create DB with 'service postgresql-9.2 initdb'
• Binaries into /usr/pgsql-X.Y/bin
• Logs into /var/lib/pgsql-X.Y/data/pg_log
• Startup logs in /var/lib/pgsql-X.Y/pgstartup.log
• Init script per major version
PostgreSQL Data Directory

• "Some thing in here do not react well to bullets."
• On Debian, just stay out of it
• On RedHat, be careful to only modify
  • postgressql.conf
  • pg_hba.conf
  • pg_ident.conf
  • pg_log/
• Do NOT touch files in pg_xlog or other dirs
• pg_xlog is PG's WAL- not just normal log files
Initial `postgresql.conf`

- `listen_addresses = '*'` (for external access)
- `checkpoint_segments = 30+`
  - Uses more disk space in `pg_xlog`
  - Never let that partition run out of space!
- `checkpoint_completion_target = 0.9`
  - Targets finishing in 90% of time given
- `effective_cache_size = half the RAM`
  - Never allocated, just for planning
- `max_wal_senders = 3`
- More later...
Logging

- `postgresql.conf`
  - `log_connections = on`
  - `log_disconnections = on`
  - `line_prefix= '%m [%p]: %q [%l-1] %d %u@%r %a'`
  - `log_lock_waits = on`
  - `log_statement = 'ddl'`
  - `log_min_duration_statement = 100`
  - `log_temp_files = 0`
  - `log_autovacuum_min_duration = 0`
pg_hba.conf

• Controls *how* users are authenticated

<table>
<thead>
<tr>
<th>Type</th>
<th>Database</th>
<th>User</th>
<th>Address</th>
<th>Method [Options]</th>
</tr>
</thead>
<tbody>
<tr>
<td>local</td>
<td>DATABASE</td>
<td>USER</td>
<td></td>
<td>METHOD [OPTIONS]</td>
</tr>
<tr>
<td>host</td>
<td>DATABASE</td>
<td>USER</td>
<td>ADDRESS</td>
<td>METHOD [OPTIONS]</td>
</tr>
<tr>
<td>hostssl</td>
<td>DATABASE</td>
<td>USER</td>
<td>ADDRESS</td>
<td>METHOD [OPTIONS]</td>
</tr>
<tr>
<td>hostnossl</td>
<td>DATABASE</td>
<td>USER</td>
<td>ADDRESS</td>
<td>METHOD [OPTIONS]</td>
</tr>
</tbody>
</table>

• Read in order, top-to-bottom, first match is used
• 'hostssl' requires SSL connection, no is not SSL
• Special DBs - 'all', 'sameuser', 'replication'
• Special Users - 'all', '+' prefix for role membership
• Address can be IPv4 or IPv6, can include CIDR mask
• Special 'reject' method
Authentication Methods

- The ones you *should* use ...
  - peer
    - Secure, unix-socket-based auth
    - Checks the Unix username of the user
  - gss (Kerberos)
    - Integrates w/ MIT/Heimdal Kerberos and AD
    - Recommended for Enterprise deployments
  - cert (SSL Certificate)
    - Client-side certificate based authentication
    - Use pg_ident to map CNs to PG usernames
Authentication Methods

• Acceptable, but not ideal...
• md5
  • Stock username/password
  • Use SSL if you're worried about security
• pam
  • Modules run as postgres user
  • Can't be used directly w/ pam_unix
  • saslauthd can make it work (pam_sasl, saslauthd)
• radius
  • Use SSL if you're worried about security
Auth Method Don'ts

- trust - Never use this- *no auth done*
- password - Password sent in cleartext
- sspi
  - Windows-specific
  - Uses Kerberos/GSSAPI underneath
- ident
  - Insecure, don't trust it- use 'peer' for local
- ldap
  - Auths against an LDAP server
  - Use Kerberos/GSSAPI if you can
pg_ident.conf

• Defines mappings which are used in pg_hba

<table>
<thead>
<tr>
<th>map-name</th>
<th>auth-user</th>
<th>pg-user</th>
</tr>
</thead>
<tbody>
<tr>
<td>kerbnames</td>
<td><a href="mailto:sfrost@SNOWMAN.NET">sfrost@SNOWMAN.NET</a></td>
<td>sfrost</td>
</tr>
<tr>
<td>certname</td>
<td>stephen.frost</td>
<td>sfrost</td>
</tr>
</tbody>
</table>

• External-user to PG-user mappings
• Unix user 'joe' can be PG user 'bob'
• Regexps can be used- but be careful
• Also works for Kerberos, client certs, etc.
Debian configs

• Extra config files in Debian/Ubuntu
  • start.conf
    • Controls start of this cluster
    • Can be 'auto', 'manual', 'disabled'
  • pg_ctl.conf
    • Options to pass to pg_ctl
    • Generally don't need to modify it
• environment
  • Controls environment PG starts in
  • Generally don't need to modify it
RedHat configs

- Basically just the init.d scripts.
Connecting

• sudo su - postgres
• psql
• \? to see backslash-commands
• \h to get help on SQL queries/commands
• Exit with \q or ctrl-d
• psql -h localhost
Looking around

- `table pg_stat_activity; - aka 'w'
- `\l` - list databases

<table>
<thead>
<tr>
<th>Name</th>
<th>Owner</th>
<th>Encoding</th>
<th>Collate</th>
<th>Ctype</th>
<th>Access privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>postgres</td>
<td>postgres</td>
<td>UTF8</td>
<td>en_US.UTF-8</td>
<td>en_US.UTF-8</td>
<td></td>
</tr>
<tr>
<td>template0</td>
<td>postgres</td>
<td>UTF8</td>
<td>en_US.UTF-8</td>
<td>en_US.UTF-8</td>
<td>=c/postgres</td>
</tr>
<tr>
<td>template1</td>
<td>postgres</td>
<td>UTF8</td>
<td>en_US.UTF-8</td>
<td>en_US.UTF-8</td>
<td>=c/postgres</td>
</tr>
</tbody>
</table>

- `\dn` - list schemas

<table>
<thead>
<tr>
<th>Name</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>public</td>
<td>postgres</td>
</tr>
</tbody>
</table>

- `\db` - list tablespaces

<table>
<thead>
<tr>
<th>Name</th>
<th>Owner</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>pg_default</td>
<td>postgres</td>
<td></td>
</tr>
<tr>
<td>pg_global</td>
<td>postgres</td>
<td></td>
</tr>
</tbody>
</table>
User setups

- `createuser` / `CREATE USER`
- `\password` to set passwords
- Privileges
  - Superuser: Do not give this out
  - `CreateRole`: Creation *and* modification of roles
  - `CreateDatabase`: Allows database creation
  - `Login`: Allows user to connect to DB
  - `Replication`: Only for replication/system user
  - `Admin`: Allows changing role memberships
  - `Inherit`: Automatically get privileges
Roles

• Users are really roles
• Groups are implemented with roles
• CREATE ROLE (or just createuser --nologin)
  • Same privilege options
  • Can start as nologin, then be granted login
  • Can cascade
• Any role can be GRANT'd to any other role
• Inherit is default, acts like group privs
• Noinherit means user must run 'set role', ala sudo
Permissions

• 'public' means 'all users'
• GRANT / REVOKE to give/take away privs, roles, etc
• CONNECT privs on the database (public by default)
• schemas - CREATE, USAGE
  • recommend dropping 'public' or revoke CREATE
  • Use per-user or per-app schemas
• tables - SELECT/INSERT/UPDATE/DELETE/TRUNCATE
• view - same (incl update!); execute as view owner
• columns - SELECT/INSERT/UPDATE
• functions - 'SECURITY DEFINER' are akin to setuid
Default perms

• Generally 'secure-by-default'
  • Except functions - EXECUTE granted by default
  • Owners have all rights on their objects
  • Membership in owning role == ownership
• ALTER DEFAULT PRIVILEGES - for roles
  • FOR ROLE ... IN SCHEMA ... GRANT
  • Can't be applied to just a schema
• GRANT ... ON ALL ... IN SCHEMA
  • For tables, views, sequences, functions
  • One-time operation, new tables will not have privs
Tuning

• For a dedicated server
• shared_buffers
  • Will be dedicated to PG for caching
  • Up to half of main memory
  • Try 2G on larger servers, more may not help
  • Pre-9.3, need to bump sysctl params
  • Post-9.3, you don't!
• Defaults to 128MB
Tuning (continued)

- **work_mem**
  - Used for in-memory hashing, sorts, etc
  - Can be increased inside a given connection
  - Used many times over- *not* a hard limit
  - Per connection, so be careful
  - Defaults to 1MB (wayyy too small..)

- **maintenance_work_mem**
  - Used for building indexes
  - Make it larger before building an index
  - Defaults to 16MB (that's a very small index)
Tuning (continued)

- **effective_cache_size**
  - Tells PG how much of the DB is in memory
  - Half of main memory
  - Never allocated, only for planning purposes
  - Defaults to 128MB
- **autovacuum**
  - On a high-rate server, make it more aggressive
  - Increase max_workers
  - Decrease autovacuum_vacuum_cost_delay
  - Defaults are for lightly loaded systems
Config Bump-Ups

- `max_connections = 100`
  - Consider using `pg_bouncer`
  - # connections == # of CPUs is ideal
- `shared_buffers = couple gig`
  - Probably not more than 3-4G (Test!)
- `maintenance_work_mem = maybe a gig`
  - Used for building indexes
- `max_locks_per_transaction = 128`
  - More if you have lots of objects
  - # locks available is actually this * max_conn
Backups

• Extremely important!
• pg_basebackup w/ WAL recieve
  • Binary-based backup
  • MUST have WAL files backed up also!
  • Needs to connect to 'replication' DB
• pg_dump
  • Logical, text-based backup
  • Does not back up indexes, must rebuild
  • Requires lightweight locks on everything
• Test restoring your data!
Monitoring

- check_postgres.pl
- Useful with Nagios, Icinga, MRTG, etc.
- Provides metrics as well as monitoring
- Allows custom query for monitoring
- Minimum set of checks

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>archive_ready (if doing WAL archiving)</td>
<td>Number of WAL .ready files</td>
</tr>
<tr>
<td>autovac_freeze</td>
<td>How close to Autovacuum Max Freeze</td>
</tr>
<tr>
<td>backends (Metric)</td>
<td>Number of Backends running</td>
</tr>
<tr>
<td>dbstats (Metrics)</td>
<td>Lots of different stats</td>
</tr>
<tr>
<td>listener (If using LISTEN/NOTIFY)</td>
<td>Checks if anyone is LISTEN'ing</td>
</tr>
<tr>
<td>locks (Metric)</td>
<td>Number of locks held</td>
</tr>
<tr>
<td>pgbouncer options (if using pgbouncer)</td>
<td>Various pgbouncer checks</td>
</tr>
<tr>
<td>txn_idle</td>
<td>Transactions idle for X time</td>
</tr>
<tr>
<td>txn_time</td>
<td>Transactions longer than X time</td>
</tr>
<tr>
<td>txn_wraparound</td>
<td>How close to transaction wraparound</td>
</tr>
</tbody>
</table>
Extensions

• Install -contrib package
• Use PGXN - http://pgxn.org
• table pg_available_extensions;

<table>
<thead>
<tr>
<th>name</th>
<th>default_version</th>
<th>installed_version</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>file_fdw</td>
<td>1.0</td>
<td></td>
<td>foreign-data wrapper for flat file access</td>
</tr>
<tr>
<td>dblink</td>
<td>1.0</td>
<td></td>
<td>connect to other PostgreSQL databases from within a database</td>
</tr>
<tr>
<td>plpgsql</td>
<td>1.0</td>
<td>1.0</td>
<td>PL/pgSQL procedural language</td>
</tr>
<tr>
<td>pg_trgm</td>
<td>1.0</td>
<td></td>
<td>text similarity measurement and index searching based on trigrams</td>
</tr>
<tr>
<td>adminpack</td>
<td>1.0</td>
<td></td>
<td>administrative functions for PostgreSQL</td>
</tr>
<tr>
<td>ip4r</td>
<td>2.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>hstore</td>
<td>1.1</td>
<td></td>
<td>data type for storing sets of (key, value) pairs</td>
</tr>
</tbody>
</table>

• adminpack allows superuser to change anything..
• \dx lists installed extensions
Thank you!

Stephen Frost
sfrost@snowman.net
@net_snow