Agenda

On the Community

Environment setup

Compilation

Regression tests

Being a committer
On The Community
Mailing lists

• Everything is mail-based.
• No bug tracker
• Hacker places
  ▪ pgsql-hackers
  ▪ pgsql-bugs
  ▪ pgsql-docs
• https://www.postgresql.org/list/
• Private lists: security, packagers, etc.
Sending a patch

• Recommended: git format-patch -1 -vN.
  ▪ Better when adding/removing files.
  ▪ Better for large patch sets.
  ▪ Includes commit message.

• Personal take: OK as long as patch -pN can work.

• [https://wiki.postgresql.org/wiki/Submitting_a_Patch](https://wiki.postgresql.org/wiki/Submitting_a_Patch)

• Discuss first, never rush writing code.
Commit fest

• Patch tracker:
  ▪ review, commit or rejection.
  ▪ Author one patch => Review one patch of “equal” difficulty.

• Includes bugs
• Needs community account.
• https://commitfest.postgresql.org/
Development cycle

- 5 commit fests: July, Sept, Nov, Jan, March.
- Feature freeze in April.
- Begins after beta 2 of previous major version => Stable branch around end of June.
- Major version in September/October.
Environment setup
Build scripts

- [https://github.com/michaelpq/home/](https://github.com/michaelpq/home/)
- Lots of environment variables.
- Dot files:
  - Public and Private (keys, etc.).
  - Configuration profiles.
- `pg_compile`
  - Source folder
  - Installation folder
  - All stable branches builds at hand.
Git

- Aliases
  - `graph = log --graph --date-order -C -M --pretty=format:\"<%h> %cd [%cn] %Cgreen%d%Creset %s\" --all --date=short`
  - `list-files = diff-tree --no-commit-id --name-only -r`

- Custom scripts
  - `git-menage`, cleans up all the files of a repo. Not from $HOME as base.
  - `git-open`, `git-find`.

- Use git prompt!

- `core.exincludefiles`, `diff.orderFile`
Aliases

- alias pgkill='killall -9 postgres'
- alias pgcheck="cd \$HOME_POSTGRES_SRC && make check-world -j 4 PROVE_FLAGS='-j 4' && echo result: \$?"
- alias pgall="cd \$HOME_POSTGRES_SRC && git menage && pg_compile 2> compile.txt && pgcheck"
- alias pgdocs="cd \$HOME_POSTGRES_SRC && git menage && pg_compile -c && cd doc/ && make html"
- Version-based aliases as well (12~16), like pg12all.
Test scripts

- `pg_start [-s N]
  - Streaming replication optional. (No cascading)
  - Archiving.
  - Configuration profiles.

- `pg_upgrade_test`
  - Tests for `pg_upgrade`.
  - `src/bin/pg_upgrade/` recommended (more later).
Compilation
CFLAGS - Mandatory

- -DENFORCE_REGRESSION_TEST_NAME_RESTRICTIONS
- -DWRITE_READPARSE_PLAN_TREES
- -DCOPY_PARSE_PLAN_TREES
- -DRAW_EXPRESSION_COVERAGE_TEST
- -Og -ggdb -g3, for gdb
- -fno-omit-frame-pointer, for perf profiles
CFLAGS – Use with care!

• Performance impact (mainly initdb):
  ▪ -DCLOBBER_CACHE_ALWAYS
  ▪ -DRELCACHE_FORCE_RELEASE -DCATCACHE_FORCE_RELEASE
  ▪ “initdb” **without** them, then run “postgres” **with** them.
  ▪ Give up “make check/installcheck”. Be frugal, only isolated tests.
• -m32, for 32-bit builds. Only gcc.
• -DEXEC_BACKEND, no fork() like WIN32.
Meson/ninja - 1

• New in Postgres 16.
• Automatic detection (-Dxx=auto) by default
  ▪ Can be a trap.
  ▪ Enable/disable to force your way.
• Feeds on CFLAGS and LDFLAGS.
• https://wiki.postgresql.org/wiki/Meson
Meson/ninja - Windows

• For MSVC or MinGW: Command Prompt.
• Avoid ActivePerl:
  ▪ Access to perl binary is tricky.
  ▪ Requires permanent online link.
• Chocolatey + Strawberry perl:
  choco install winflexbison sed gzip diff
  choco install strawberryperl
• MinGW good these days for binaries.
• Powershell with VS compiler faster?
Overlap meson/configure

- CFLAGS/LDFLAGS shared => environment variables.
- ./configure until removed from all stable branches.
- Meson/ninja is **much** nicer.
- Build options separated: --with-XX vs –Dxx=enabled
OpenSSL

- Down to 0.9.8 for 12~ (v17 dropped 1.0.1).
- Custom builds
  ./config -d –prefix=$HOME/stable/openssl/VERSION/ shared
  make install
  export CFLAGS+="$HOME/stable/openssl/VERSION/include "
  export LDFLAGS+="$HOME/stable/openssl/VERSION/lib "
  export PATH=$HOME/stable/openssl/VERSION/bin:$PATH
- OPENSSL: src/test/ssl/.
- Beware of python or openldap linking to OpenSSL.
Regression tests
General matters

• TAP tests mandatory (--enable-tap-tests)
• Parallelism:
  make check-world -j 32 PROVE_FLAGS="-j 32"
• PG_TEST_EXTRA
  ▪ Use PG_TEST_EXTRA='kerberos ldap ssl'
  ▪ Unsecure, still recommended.
• PG_TEST_NOCLEAN
• PGOPTIONS
• EXTRA_REGRESS_OPTS
installcheck

• Self-contained
• Repeatable.
• Some objects may stay around for pg_upgrade.
• Clean all shared objects:
  ▪ Roles.
  ▪ Replication slots, etc.
pg_upgrade – cross-version checks

• Based on TAP tests.
• Requires:
  ▪ Binaries of an old version.
  ▪ SQL Dump of old version, taken with old pg_dumpall.
• See src/bin/pg_upgrade/TESTING
• olddump=/to/dump/dump.sql \  
oldinstall=/to/install/ make check
• January 2023 (52585f8)
CI and github

- src/tools/ci/README
  - Create Postgres repo on github.
  - Enable cirrus-ci.
  - Push specific branches for tests.
  - Per-platform filtering possible (Only Windows, etc.).
- Used at http://cfbot.cputube.org/.
- Beware of costs.
Code coverage

• --enable-coverage
• ./configure
  ▪ make check
    make coverage-html
• Meson
  ▪ meson test
    ninja coverage-html
• See https://coverage.postgresql.org/
Being a Committer
What’s up?

• Since May 2018: 1400~ commits, ongoing series.
• Responsible for maintenance of own commits.
• Social contract.
• Errors:
  ▪ Are fine. Don’t do too many.
  ▪ Will happen anyway.
  ▪ Be responsive.
• Applies to authors and reviewers (thinner responsibility).
**git-worktree**

- One folder per stable branch:
  
  ```
  $ git worktree list
  /Users/paquier/git/postgres 2c7c6c417f [master]
  /Users/paquier/git/pg12 3b4580f5ce [REL_12_STABLE]
  /Users/paquier/git/pg13 154bcce3ba [REL_13_STABLE]
  /Users/paquier/git/pg14 4fada7def9 [REL_14_STABLE]
  /Users/paquier/git/pg15 985ac5ce29 [REL_15_STABLE]
  /Users/paquier/git/pg16 efaa50c783 [REL_16_STABLE]
  ```
Some rules

• “You shall not break the buildfarm.”
• src/tools/
  ▪ pgindent
  ▪ pgperlcritic, pgperltidy
• Catversion, WAL format, control file bumps.
  ▪ Author should not do that, less CF conflicts.
  ▪ Committer sole responsible.
• git cherry-pick: master, then stable.
External libraries

• Test and check all patches in isolation for each branch.
• ~7 stable branches * library versions supported.
• Use own builds to link to (recommended).
  ▪ CFLAGS+=“-I/path/to/include/ ”
  ▪ LDFLAGS+=“-L/path/to/lib/ ”
• Check patches without these libraries (--without-zlib, etc.)
Thank you!

Michael Paquier
Mastodon: @michaelpq@fosstodon.org
Twitter: michaelpq