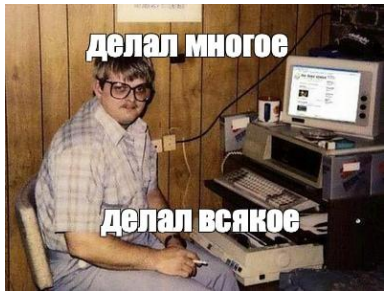
Билдь или не билдь...

…или Как достойно собрать PostgreSQL из исходников

Вадим Пономарев

“Тантор Лабс”

О себе



20+ лет в IT

Архитектор
Linux-инженер
DBA
Руководитель
Могу копать
Могу не копать

IBS
Рубитех
Скала-Р
Озон

Теперь делаю tantor XData PostgreSQL as a Service on premises
(коробочное решение для частного облака с сервисом Postgres).

А можно ничего не делать? Можно!

PostgreSQL release cycle (https://www.postgresql.org/docs/release/)

- 1 major release per year
- 1 minor release per quarter
- + extra releases to fix security issues

Готовые пакеты можно взять

- ваш дистрибутив (обычно, с лагом)
- PGDG (см. **https://yum.postgresql.org/**, **https://wiki.postgresql.org/wiki/Apt**)

Тогда какой смысл?! А он таки есть!

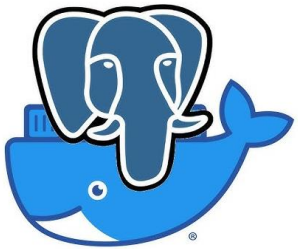
- обучение
- разработка
- bugfix
- feature backport

- тесты нового/необычного
- нет нужной версии PG
  …под вашу операционную систему

Дальше будет …

- **не трогаем сборку под Windows,**
- **только на Linux и для Linux,**
- **конкретно в примерах - Ubuntu 22.04**
- **… на WSL.**
- **Но лучше, конечно, в контейнере.**

```
$ sudo docker run --rm -it ubuntu:22.04 bash
```



- **не трогаем автоматизацию - сборка вручную**
- **… зато тестируем и патчим!**

- **про разработку/дебаг тоже не говорим,**
- **только сборка, только хардкор!**

Tutorial

https://github.com/vbp1/pgbootcamp23

Подготовка

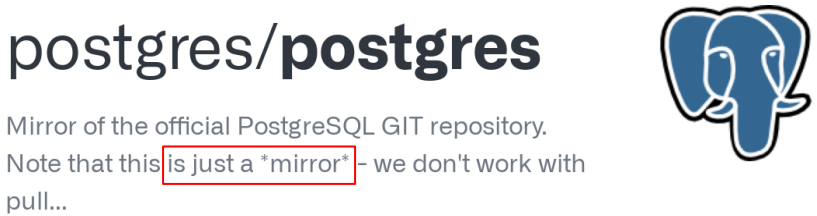
- **IDE** (Visual Studio Code) - оно неплохо
  - … но необязательно!
- **dev tools**:
  - git, autoconf, make, gcc,
  - tar, readline, zlib, Flex, Bison, Gettext
- **optional**:
  - Python, Perl, Tcl, OpenSSL,
  - MIT Kerberos, OpenLDAP,
  - PAM, LZ4, Zstandard, LLVM, Clang
  - libxml2, libxslt
- **for docs**:
  - DocBook, FOP, xmllint, xsltproc
- **internalization**:
  - locales, libICU

```
$ export DEBIAN_FRONTEND=noninteractive
$ sudo apt-get update
$ sudo apt-get install -y \
              git tree nano \
              build-essential cpanminus slapd ldap-utils libldap2-
dev \
              autoconf bison clang-11 devscripts dpkg-dev flex \
              libldap2-dev libdbi-perl libgssapi-krb5-2 libicu-dev
\
              krb5-kdc krb5-admin-server libssl-dev libpam0g-dev \
              libkrb5-dev krb5-user libcurl4-openssl-dev \
              perl perl-modules libipc-run-perl libtest-simple-
perl \
       libtime-hires-perl liblz4-dev libpam-dev libreadline-dev \
       libselinux1-dev libsystemd-dev libxml2-dev libxslt-dev \
       libzstd-dev llvm-11-dev locales-all pkg-config \
              python3-dev uuid-dev zlib1g-dev \
              openjade docbook-xml docbook-xsl opensp libxml2-
utils \
       xsltproc libjson-perl curl time

$ sudo cpanm TAP::Harness::Archive\
                    TAP::Parser::SourceHandler::pgTAP\
                     IPC::Run
```

Загрузка исходного кода

- **git clone**
  - https://www.postgresql.org/ftp/source/
  - git://git.postgresql.org/git/postgresql.git
  - **https://github.com/postgres/postgres**



- **опционально:**
  - **git branch -a**
  - **git checkout origin/{branch_name}**

Конфигурирование и сборка

- [CC=compiler] ./configure [options]
- make [world, world-bin] [-j X]

```
$ export WORKSPACE_PATH=/home/vponomarev
$ ./configure --prefix=$WORKSPACE_PATH/postgres_bin CFLAGS='-O2 -pipe'\
            --with-python --with-llvm\
           --with-lz4 --with-zstd --with-ssl=openssl --enable-nls='ru'\
            --enable-tap-tests --enable-depend --enable-debug
$ make world-bin -j $(nproc)
```

Don't modify files generated by configure!

src/include/pg_config.h

src/include/pg_config.h.in

Тестирование-1

- [PG_TEST_EXTRA='kerberos ldap ssl wal_consistency_checking'] [LANG=ru_RU.UTF-8] [LC_COLLATE=en_US.utf8 LC_CTYPE=fr_CA.utf8] [LANG=C ENCODING=EUC_JP]
  make [check, check-world] [-j X]
  - make check PGOPTIONS="-c debug_parallel_query=regress -c work_mem=50MB"
  - echo 'log_checkpoints = on' > test_postgresql.conf
  - echo 'work_mem = 50MB' >> test_postgresql.conf
  - make check EXTRA_REGRESS_OPTS="--temp-config=test_postgresql.conf"

- опционально TAP-тесты:
  - make -C src/bin check (нужен perl + Perl module IPC::Run + ./configure … --enable-tap-tests)

- Посмотреть результаты:
  - less ./src/test/recovery/tmp_check/log/regress_log*
  - less ./src/test/regress/regression.diffs

Установка, тестирование-2, очистка

- Установка
  - $ sudo make install [install-world install-world-bin]

- Запуск
  - $ PATH=$WORKSPACE_PATH/postgres_bin/bin:$PATH
  - $ LD_LIBRARY_PATH=$WORKSPACE_PATH/postgres_bin/lib
  - $ PGDATA=$WORKSPACE_PATH/postgres_data
  - $ initdb -D $PGDATA
  - $ pg_ctl -D $PGDATA start

- Тестирование
  - $ make installcheck [installcheck-parallel installcheck-world]
- …
- Очистка
  - $ make distclean

Документация и автоматизация

Manuals:

- https://www.postgresql.org/docs/current/install-make.html
- https://www.postgresql.org/docs/current/regress.html

Scripts and automatization

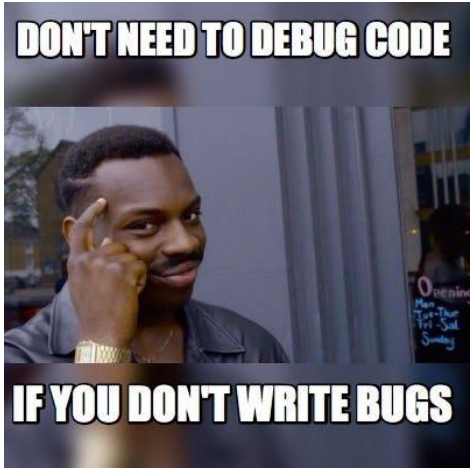
- https://github.com/afiskon/pgscripts
- https://github.com/postgres/postgres/tree/master/src/tools/ci

Coredumps и debug-symbols

Enable core dumps

Core Dumps — How to enable them?
Check & change core dump size limit:
ulimit -a
ulimit -S -c unlimited
Check & change core_pattern:
sysctl kernel.core_pattern
sudo sysctl -w kernel.core_pattern=/var/crash/core-%e-%s-%u-%g-%p-%t

warning!
non persistent changes!



Add debug symbols

при сборке: configure … --enable-debug …
при установке из пакетов: поставить пакеты …-dbgsym, например
$ apt list | grep postgresql-15-dbgsym; apt-get install postgresql-15-dbgsym

Патчи. Disclaimer

Далее будет довольно искусственный и тривиальный пример (но у него есть большое достоинство - он крайне простой!).

Тот кому хочется творчества может попробовать забэкпортить функциональность **On client login event trigger** (патч https://commitfest.postgresql.org/44/2900/) в PG15 или 16.

Патчи

Есть Проблема!

Исследуем проблему: gdb и stack trace

Debugging with GDB
postgres=# select pg_backend_pid();
pg_backend_pid

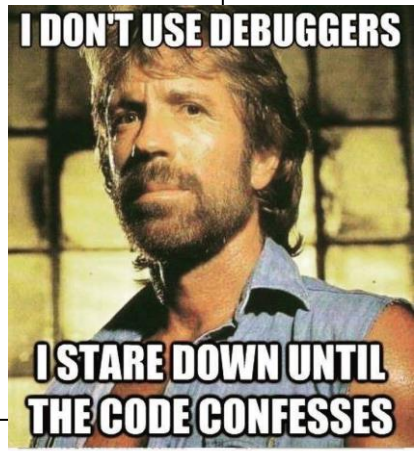
2512670
Connect with gdb:
gdb -p 2512670
break ProcessUtility
break ProcessQuery
https://wiki.postgresql.org/wiki/Developer_FAQ#gdb
Getting a stack trace of a running PostgreSQL backend on Linux/BSD

```
(gdb) bt
#0  0x00007f8d0140bf9a in epoll_wait (epfd=5, events=0x562c2abac680, maxevents=1, tim
#1  0x0000562c2a04669b in WaitEventSetWaitBlock (set=set@entry=0x562c2abac608, cur_ti
    at latch.c:1529
#2  0x0000562c2a04725a in WaitEventSetWait (set=0x562c2abac608, timeout=timeout@entry
    wait_event_info=wait_event_info@entry=100663296) at latch.c:1475
#3  0x0000562c29ee1b6f in secure_read (port=0x562c2aba4320, ptr=0x562c2a4f4460 <PqRec
#4  0x0000562c29ee9caf in pq_recvbuf () at pqcomm.c:939
#5  0x0000562c29eea949 in pq_getbyte () at pqcomm.c:982
#6  0x0000562c2a0712ca in SocketBackend (inBuf=0x7ffe7d542030) at postgres.c:336
#7  0x0000562c2a072e09 in ReadCommand (inBuf=inBuf@entry=0x7ffe7d542030) at postgres.
#8  0x0000562c2a076211 in PostgresMain (dbname=<optimized out>, username=<optimized o
#9  0x0000562c29fccf61 in BackendRun (port=port@entry=0x562c2aba4320) at postmaster.c
#10 0x0000562c29fd007f in BackendStartup (port=port@entry=0x562c2aba4320) at postmast
#11 0x0000562c29fd022d in ServerLoop () at postmaster.c:1779
#12 0x0000562c29fd173a in PostmasterMain (argc=argc@entry=3, argv=argv@entry=0x562c2a
#13 0x0000562c29eed54c in main (argc=3, argv=0x562c2ab6f810) at main.c:200
(gdb)
```

```
postgres=# select pg_backend_pid();
 pg_backend_pid
----------------
        2512670
```

Connect with gdb:

```
gdb -p 2512670
    break ProcessUtility
    break ProcessQuery
```



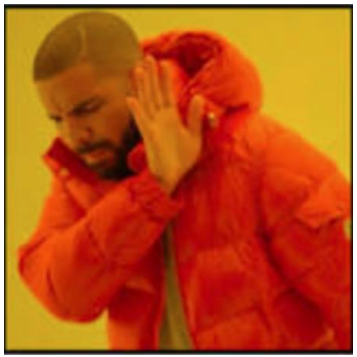

```
$ gdb -p pid --ex bt --batch
```

```
$ gdb /path/to/executable /path/to/coredump --ex 'bt full' --batch
```

Патчи

Pgsql Hackers (https://www.postgresql.org/list/pgsql-hackers/)
Commitfest (https://commitfest.postgresql.org/),
Google и ChatGPT в конце концов!

Главное - не включать голову;)



Патчи: PostgreSQL commitfest

https://commitfest.postgresql.org/
http://commitfest.cputube.org/

Commitfest 2021-01

Search/filter | Shortcuts

Status summary: Needs review: 188. Waiting on Author: 28. Ready for Committer: 22. Committed: 19. Withdrawn: 3. Total: 260.

Active patches

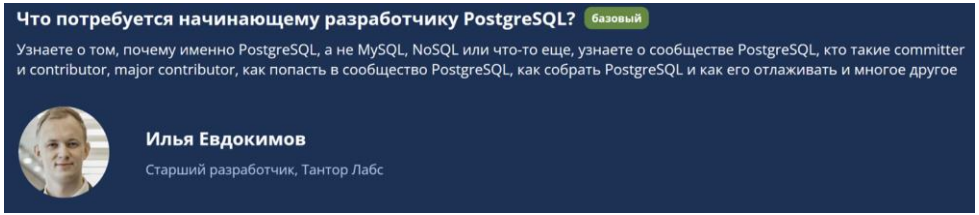
Patch	Status	Ver	Author	Reviewers	Committer	Num cfs	Latest activity	Latest mail
faster execution of CALL statement from plpgsql	Needs review		Pavel Stehule (okbobcz)			1	2020-11-02 12:33	2021-01-01 08:15
Functions for sorting GiST build of gist_btree types	Needs review		Andrey Borodin (x4m)	Heikki Linnakangas (heikki)		1	2020-11-05 16:15	2020-12-10 10:16
Pageinspect functions for GiST	Needs review		Heikki Linnakangas (heikki)	Andrey Borodin (x4m)		1	2020-11-05 16:15	2020-12-10 10:16
pgbench stopped supporting large number of client connections on Windows	Waiting on Author		Marina Polyakova (whipping.top1991)			1	2020-11-08 16:35	2020-11-08 14:59
Extended statistics / estimate Var op Var clauses	Needs review		Tomas Vondra (fuzzycz)			1	2020-11-16 14:34	2020-11-13 01:14
Extended statistics on expressions	Needs review		Tomas Vondra (fuzzycz)	Justin Pryzby (justinpryzby)		1	2020-11-16 14:34	2020-12-11 20:17
Add nullif case for eval_const_expressions	Needs review		hou zhijie (houzj)			1	2020-11-17 01:24	2020-11-10 11:31

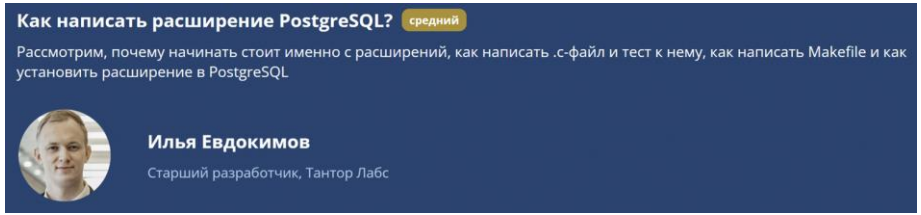
Патчи

- Патч решает мою проблему?! - git apply …
- (Не вышло?! Адаптируем патч!)
- Тест
- …
- PROFIT!!!

Полезности

- https://www.postgresql.org/docs/current/installation.html
- https://www.postgresql.org/docs/current/regress.html
- https://github.com/postgres/postgres/blob/master/src/tools/ci/README
- книга Егора Рогова “PostgreSQL 15 изнутри” (https://postgrespro.ru/education/books/internals)
- доклады на PG BootCamp Russia 2023:





- курс Hacking PostgreSQL (https://postgrespro.ru/education/courses/hacking)

Stay tuned

В следующих сериях:

- Упаковка собранного PG в пакеты
- Автоматизация сборки PG (как это делаем мы?*)

* Только правой!

Спасибо!

https://t.me/vbponomarev

p.s. Готовы к следующему DIY проекту? ;)

Потроха PG: структура

Directory	Description
config	Config system for driving the build
contrib	Source code for Contrib Modules, aka, Extensions
doc	Documentation (SGML)
src/backend	PostgreSQL Server ("Back-End")
src/bin	psql, pg_dump, initdb, pg_upgrade, etc ("Front-End")
src/common	Code common to the front and back ends
src/fe_utils	Code useful for multiple front-end utilities
src/include	Header files for PG, mainly back-end
src/include/catalog	Definition of the PostgreSQL catalog tables (pg_catalog.* tables)
src/interfaces	Interfaces to PG, including libpq, ECPG
src/pl	Core Procedural Languages (plpgsql, plperl, plpython, tcl)
src/port	Platform-specific hacks
src/test	Regression tests
src/timezone	Timezone code from IANA
src/tools	Developer tools (including pgindent)

Потроха PG: структура

Directory	Description
access	Methods for accessing different types of data (heap, btree indexes, gist/gin, etc).
bootstrap	Routines for running PostgreSQL in "bootstrap" mode (by initdb)
catalog	Routines used for modifying objects in the PG Catalog (pg_catalog.*)
commands	User-level DDL/SQL commands (CREATE/ALTER, VACUUM/ANALYZE, COPY, etc)
executor	Executor, runs queries after they have been planned/optimized
foreign	Handles Foreign Data Wrappers, user mappings, etc
jit	Provider independent Just-In-Time Compilation infrastructure
lib	Code useful for multiple back-end components
libpq	Backend code for talking the wire protocol
main	main(), determines how the backend PG process is starting and starts right subsystem
nodes	Generalized "Node" structure in PG and functions to copy, compare, etc
optimizer	Query optimizer, implements the costing system and generates a plan for the executor
parser	Lexer and Grammar, how PG understands the queries you send it
partitioning	Common code for declarative partitioning in PG
po	Translations of backend messages to other languages

Потроха PG: структура

Directory	Description
port	Backend-specific platform-specific hacks
postmaster	The "main" PG process that always runs, answers requests, hands off connections
regex	Henry Spencer's regex library, also used by TCL, maintained more-or-less by PG now
replication	Backend components to support replication, shipping WAL logs, reading them in, etc
rewrite	Query rewrite engine, used with RULEs, also handles Row-Level Security
snowball	Snowball stemming, used with full-text search
statistics	Extended Statistics system (CREATE STATISTICS)
storage	Storage layer, handles most direct file i/o, support for large objects, etc
tcop	"Traffic Cop"- this is what gets the actual queries, runs them, etc
tsearch	Full-Text Search engine
utils	Various back-end utility components, cacheing system, memory manager, etc