Static analysis, test coverage, and other 12+ letter words

A tour of ways to find and prevent bugs in PostgreSQL

Peter Eisentraut
peter@eisentraut.org
@petereisentraut

meetme

PGCon 2014

Compiler Warnings

Compiler Warnings

```
-Wall
-Wmissing-prototypes
-Wpointer-arith
-Wdeclaration-after-statement
-Wendif-labels
-Wmissing-format-attribute
-Wformat-security
```

Compiler Warnings

My options:

```
-D_FORTIFY_SOURCE=2
-Wlogical-op

-Wignored-qualifiers
-Wmissing-parameter-type
-Wold-style-declaration
-Woverride-init
-Wuninitialized
-Wunused-but-set-parameter

-Wno-tautological-compare
-Wloop-analysis
-Wuninitialized

-Wheader-guard
-Wlogical-not-parentheses
```

Using -Werror

```
pgconfigure() {
    ./configure "$@"
    if [ ! -e src/Makefile.custom ]; then
      cat <<EOF >src/Makefile.custom
      COPT = -Werror
      EOF
    fi
}
```

Compiler Warnings

	9.4	9.3	9.2	9.1	9.0	8.4	8.3
gcc-4.9	36	36	43	67	121	232	230
gcc-4.8	0	0	0	25	79	192	194
gcc-4.7	0	0	0	25	79	192	194
gcc-4.6	0	0	0	25	79	192	210
gcc-4.5	0	0	0	0	4	9	35
gcc-4.4	0	0	0	0	0	0	145
gcc-4.3	0	0	0	0	0	0	145
gcc-4.2	0	0	0	0	0	0	6
clang-3.4	0	0	10	8	9	31	50
clang-3.3	0	0	7	8	9	31	50
clang-3.2	0	0	7	8	9	31	50

Coverity

Coverity

- `https://scan.coverity.com/`
- 71 commits mentioning Coverity, since 2005
- currently 1501 issues in master

Scan Build

http://clang-analyzer.llvm.org/scan-build.html

Scan Build

How to

```
$ scan-build -o $PWD ./configure --enable-cassert ...
$ scan-build -o $PWD make world
...
All of PostgreSQL successfully made. Ready to install.
scan-build: 272 bugs found.
scan-build: Run 'scan-view ...' to examine bug reports.
$ scan-view ...
```

Scan Build

	*	9.4	9.3	9.2	9.1	9.0	8.4	8.3
scan-build	211	272	244	883	1393	1388	1386	1230

Address Sanitizer

https://code.google.com/p/address-sanitizer/
http://clang.llvm.org/docs/AddressSanitizer.html

Address Sanitizer

How to

```
./configure CC=clang CFLAGS='-O1 -g -fsanitize=address
    -fno-omit-frame-pointer -fno-optimize-sibling-calls'
# or gcc-4.8
make
make check  # will crash or not
asan_symbolize.py < log/postmaster.log
```

Address Sanitizer

```
diff --git a/src/backend/access/common/tupdesc.c
     b/src/backend/access/common/tupdesc.c
index 11c31d8..bf47640 100644
--- a/src/backend/access/common/tupdesc.c
+++ b/src/backend/access/common/tupdesc.c
@@ -501,12 +507,12 @@
     /*
      * Note: attributeName can be NULL, because the planner doesn't always
      * fill in valid resname values in targetlists, particularly for resjunk
-     * attributes.
+     * attributes. Also, do nothing if caller wants to re-use the old attname.
      */
-    if (attributeName != NULL)
-        namestrcpy(&(att->attname), attributeName);
-    else
+    if (attributeName == NULL)
         MemSet(NameStr(att->attname), 0, NAMEDATALEN);
+    else if (attributeName != NameStr(att->attname))
+        namestrcpy(&(att->attname), attributeName);

     att->attstattarget = -1;
     att->attcacheoff = -1;
```

Address Sanitizer

```
diff --git a/src/backend/utils/adt/ruleutils.c b/src/backend/utils/adt/ruleutils.c
index 74b573b..dffac7c 100644
--- a/src/backend/utils/adt/ruleutils.c
+++ b/src/backend/utils/adt/ruleutils.c
@@ -632,7 +632,7 @@ static char *generate_function_name(Oid funcid, int nargs,
     * Get the pg_rewrite tuple for the view's SELECT rule
     */
    args[0] = ObjectIdGetDatum(viewoid);
-   args[1] = PointerGetDatum(ViewSelectRuleName);
+   args[1] = DirectFunctionCall1(namein, CStringGetDatum(ViewSelectRuleName));
    nulls[0] = ' ';
    nulls[1] = ' ';
    spirc = SPI_execute_plan(plan_getviewrule, args, nulls, true, 2);)
```

Address Sanitizer

http://pgci.eisentraut.org/jenkins/job/postgresql_master_sanitize_address/390/artifact/src/test/regress/log/postmaster.log.asan/*view*/

```
==22216==ERROR: AddressSanitizer: stack-buffer-overflow on address 0x7fff9b3db294 at
      pc 0x60d5c1 bp 0x7fff9b3da910 sp 0x7fff9b3da908
READ of size 1 at 0x7fff9b3db294 thread T0
    #0 0x60d5c0 in XLogInsert src/backend/access/transam/xlog.c:1049
    #1 0x5d2d3b in doPickSplit src/backend/access/spgist/spgdoinsert.c:1389
    #2 0x5c875a in spgdoinsert src/backend/access/spgist/spgdoinsert.c:2006
    #3 0x5bcc33 in spginsert src/backend/access/spgist/spginsert.c:238
    #4 0xd2518c in FunctionCall6Coll src/backend/utils/fmgr/fmgr.c:1437
    #5 0x57821e in index_insert src/backend/access/index/indexam.c:226
    #6 0x8530a8 in ExecInsertIndexTuples src/backend/executor/execUtils.c:1104
    #7 0x87e743 in ExecInsert src/backend/executor/nodeModifyTable.c:274
    #8 0x87db1f in ExecModifyTable src/backend/executor/nodeModifyTable.c:1024
    #9 0x836306 in ExecProcNode src/backend/executor/execProcnode.c:377
    #10 0x82fac6 in ExecutePlan src/backend/executor/execMain.c:1475
    #11 0x82f8b1 in standard_ExecutorRun src/backend/executor/execMain.c:308
    #12 0x82f638 in ExecutorRun src/backend/executor/execMain.c:256
    #13 0xaf58df in ProcessQuery src/backend/tcop/pquery.c:185
    #14 0xaf4633 in PortalRunMulti src/backend/tcop/pquery.c:1287
    #15 0xaf33c8 in PortalRun src/backend/tcop/pquery.c:816
    #16 0xaed221 in exec_simple_query src/backend/tcop/postgres.c:1045
    #17 0xaec507 in PostgresMain src/backend/tcop/postgres.c:4004
    #18 0xa00c71 in BackendRun src/backend/postmaster/postmaster.c:4104
    #19 0xa00011 in BackendStartup src/backend/postmaster/postmaster.c:3778
    #20 0x9fc233 in ServerLoop src/backend/postmaster/postmaster.c:1569
    #21 0x9f9760 in PostmasterMain src/backend/postmaster/postmaster.c:1222
    #22 0x8caea7 in main src/backend/main/main.c:223
    #23 0x2accb99ca994 in __libc_start_main ??:?
    #24 0x49a66c in _start ??:?
Address 0x7fff9b3db294 is located in stack of thread T0 at offset 724 in frame
    #0 0x5cd24f in doPickSplit src/backend/access/spgist/spgdoinsert.c:678
```

Valgrind

http://valgrind.org/

Valgrind

How to

```
./configure CFLAGS='-O0 -g' ...
make # COPT='-DUSE_VALGRIND'
make install

valgrind --leak-check=yes --log-file=val.log
    pg-install/bin/pg_dump regression >/dev/null

valgrind \
  --trace-children=yes --quiet \
  --track-origins=yes --leak-check=no \
  --read-var-info=yes \
  --suppressions=src/tools/valgrind.supp \
  pg-install/bin/postgres
```

https://wiki.postgresql.org/wiki/Valgrind

Test Coverage

Test Coverage

How to

```
./configure --enable-coverage ...
make all
make coverage-clean
make check MAX_CONNECTIONS=1
make coverage-html
$BROWSER coverage/index.html
```

goto fail

goto fail

```
diff --git a/src/backend/libpq/pqcomm.c
    b/src/backend/libpq/pqcomm.c
index 605d891..94fd5e6 100644
--- a/src/backend/libpq/pqcomm.c
+++ b/src/backend/libpq/pqcomm.c
@@ -1384,6 +1384,7 @@ pq_putmessage(char msgtype, const
    char *s, size_t len)
    if (msgtype)
        if (internal_putbytes(&msgtype, 1))
            goto fail;
+           goto fail;
    if (PG_PROTOCOL_MAJOR(FrontendProtocol) >= 3)
    {
        uint32      n32;
```

Other Things to Try

- `clang -fsanitize=undefined`
- `clang -fsanitize=memory`
- `clang -fsanitize=thread`
- Splint
- PVS-Studio (Viva64)

SLOC

V	SLOC		C	MsR	CsR	C/M	h/d
9.4	747,766	(+4.7%)	18	–	–	–	24
9.3	713,926	(+4.4%)	18	8	353	44	24
9.2	683,557	(+5.5%)	18	20	592	29	24
9.1	648,217	(+8.3%)	19	32	783	24	24
9.0	598,076	(+5.8%)	21	44	911	20	24
8.4	565,206	(+6.7%)	22	59	995	16	24
8.3	529,585		20	60	936	15	24

Summary

- Always build with `-Werror`
- Let's fix some scan-build problems and scan-build
- Automate Valgrind tests
- Fix ASAN
- Add more tests
- Immediately jump on the bandwagon of every new tool that comes out.