Масштабируемость PostgreSQL

Дмитрий Васильев

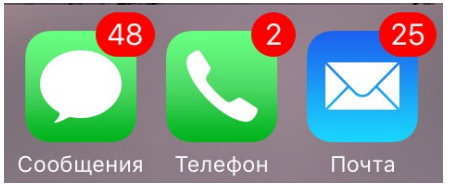
www.postgrespro.ru

Когда рабочий день начинается так

504 Gateway Time-out

---

nginx



План оптимизации БД

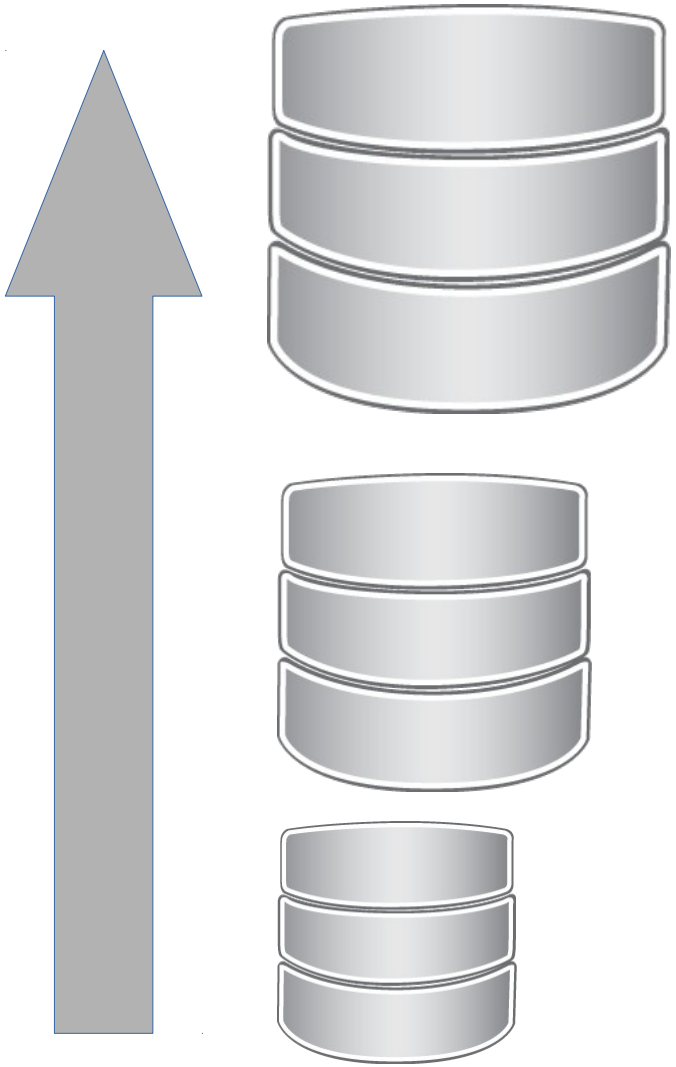
* Оптимизация запросов
* Настройки ОС
* Изменение архитектуры приложения
* Настройка железа
* Модернизация железа

Postgres PROFESSIONAL

Масштабируемость -

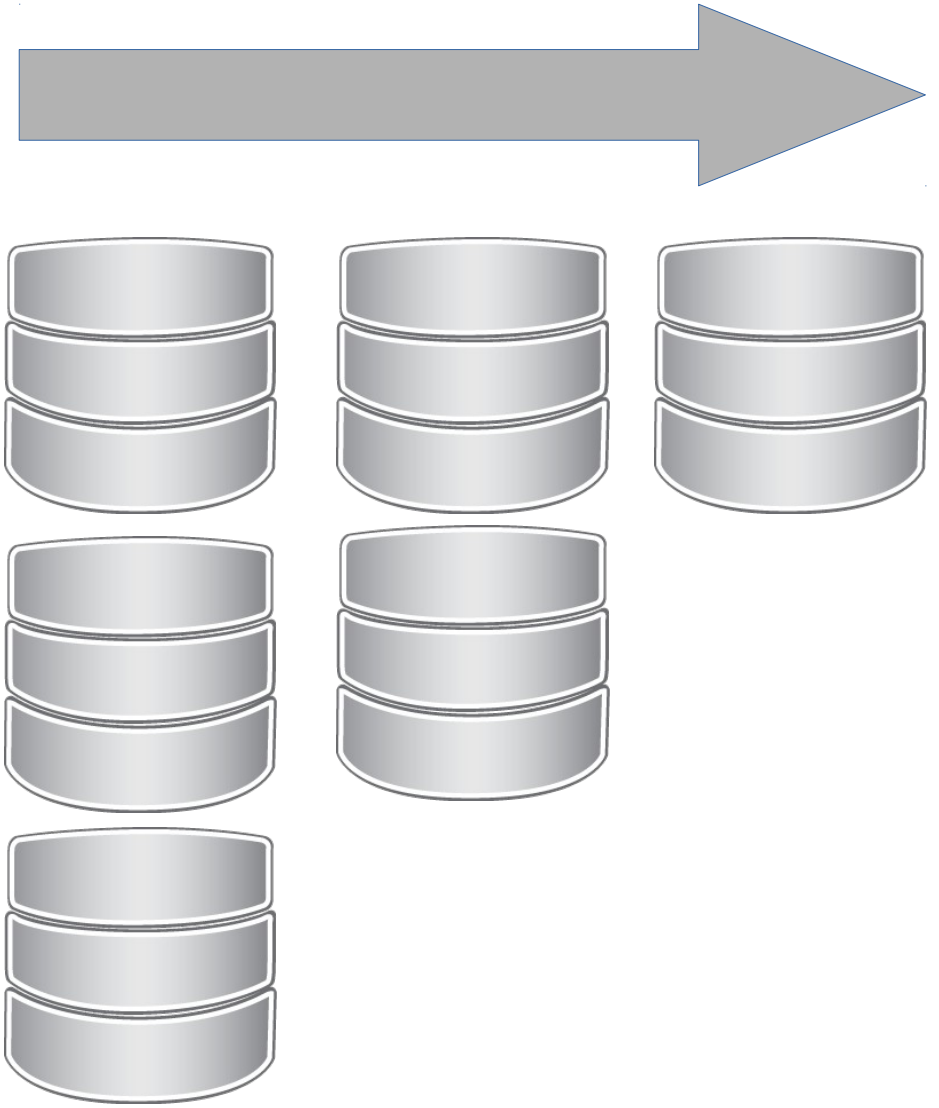
способность увеличивать производительность при добавлении ресурсов

Масштабирование

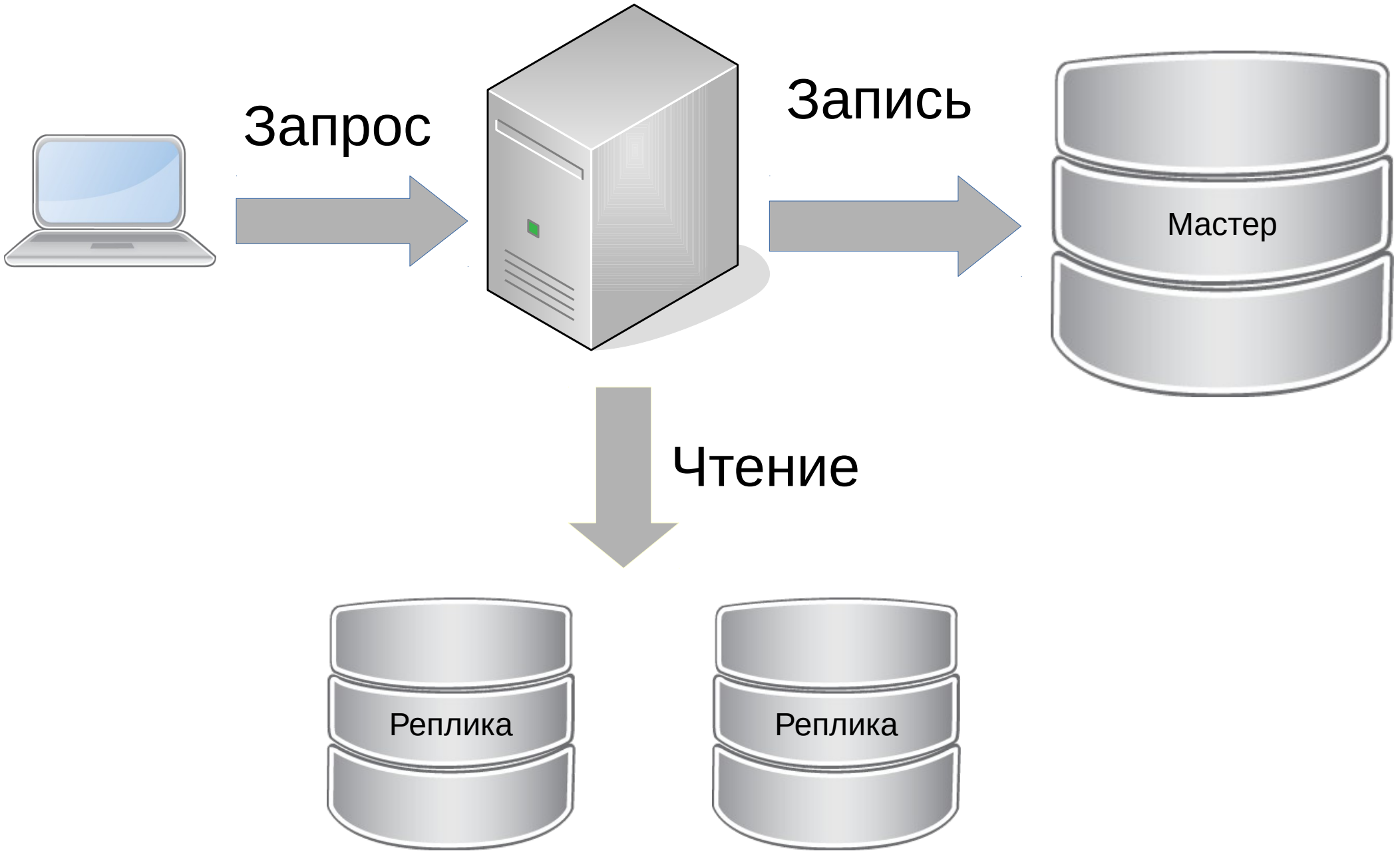


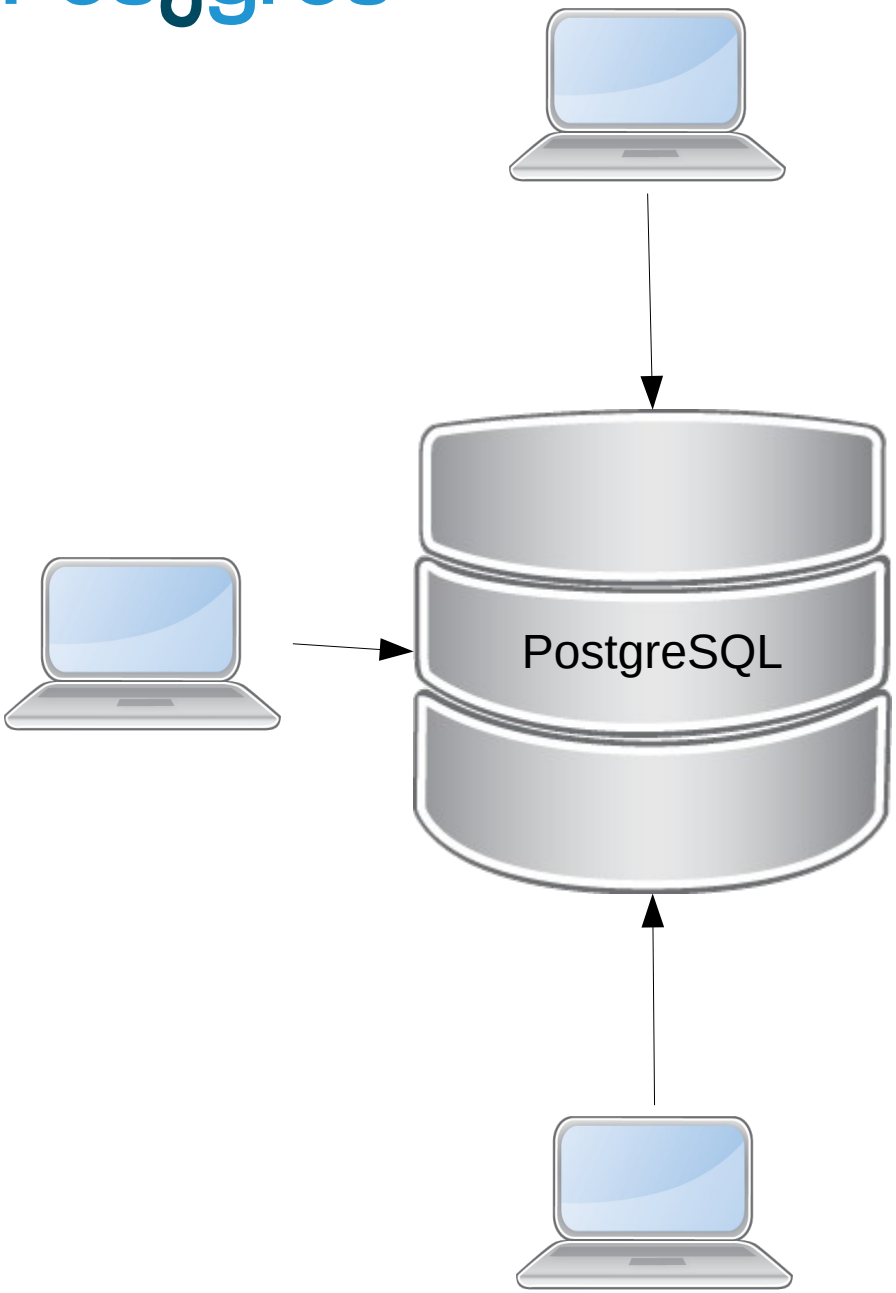
Вертикальное

Горизонтальное



Вертикальное





* Каждый клиент порождает 1 процесс бакэнда (до 9.6)

* Каждый бакэнд однопоточный и может утилизировать 1 логический CPU

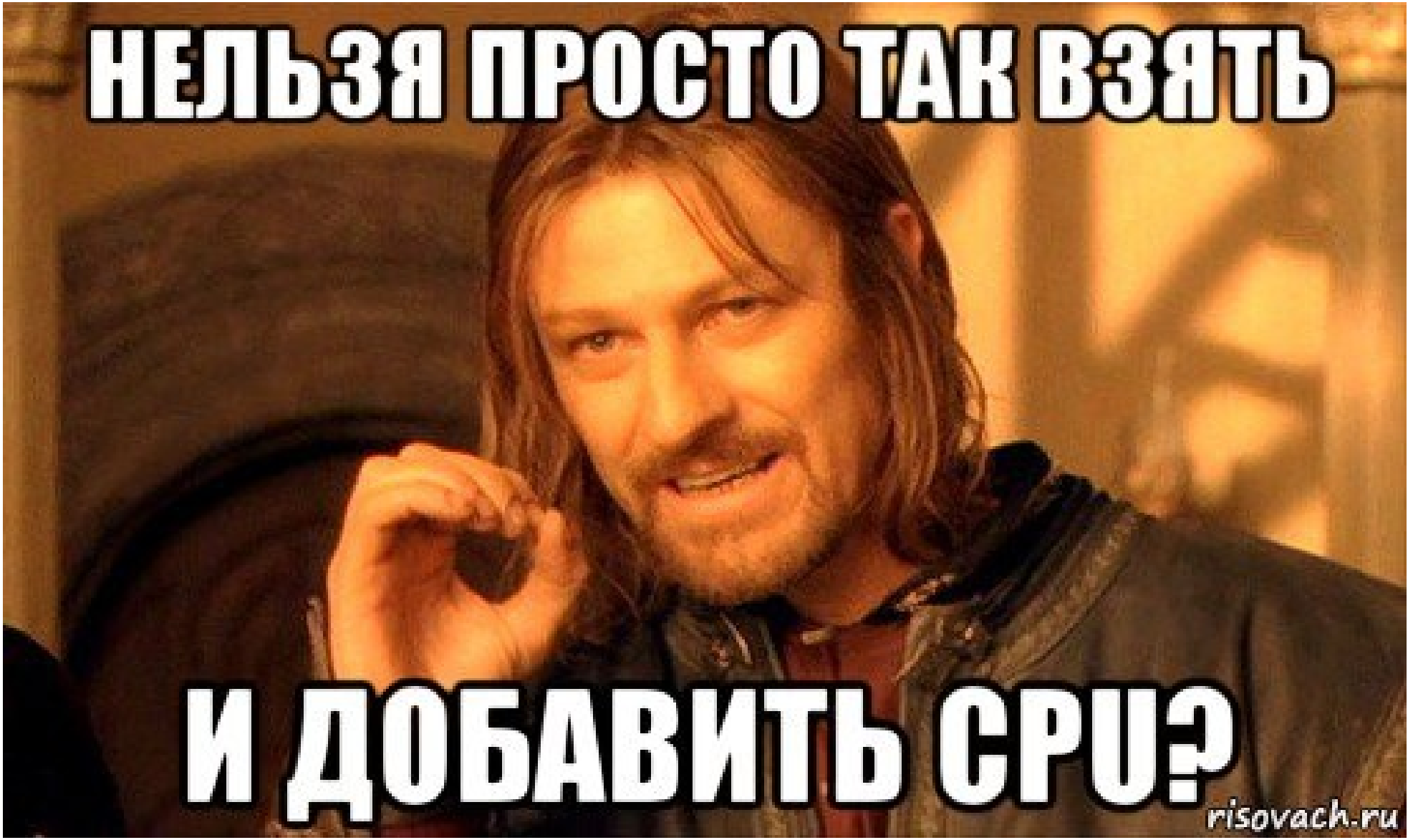
* По каждому запросу составляется план выполнения, которому нужна информация по объектам и статистика

* Бакэнды взаимодействуют с данными, раположеными на дисках через общую разделяемую память, в процессе этой работы данные могут быть записаны, обновлены или вытеснены из разделяемой памяти

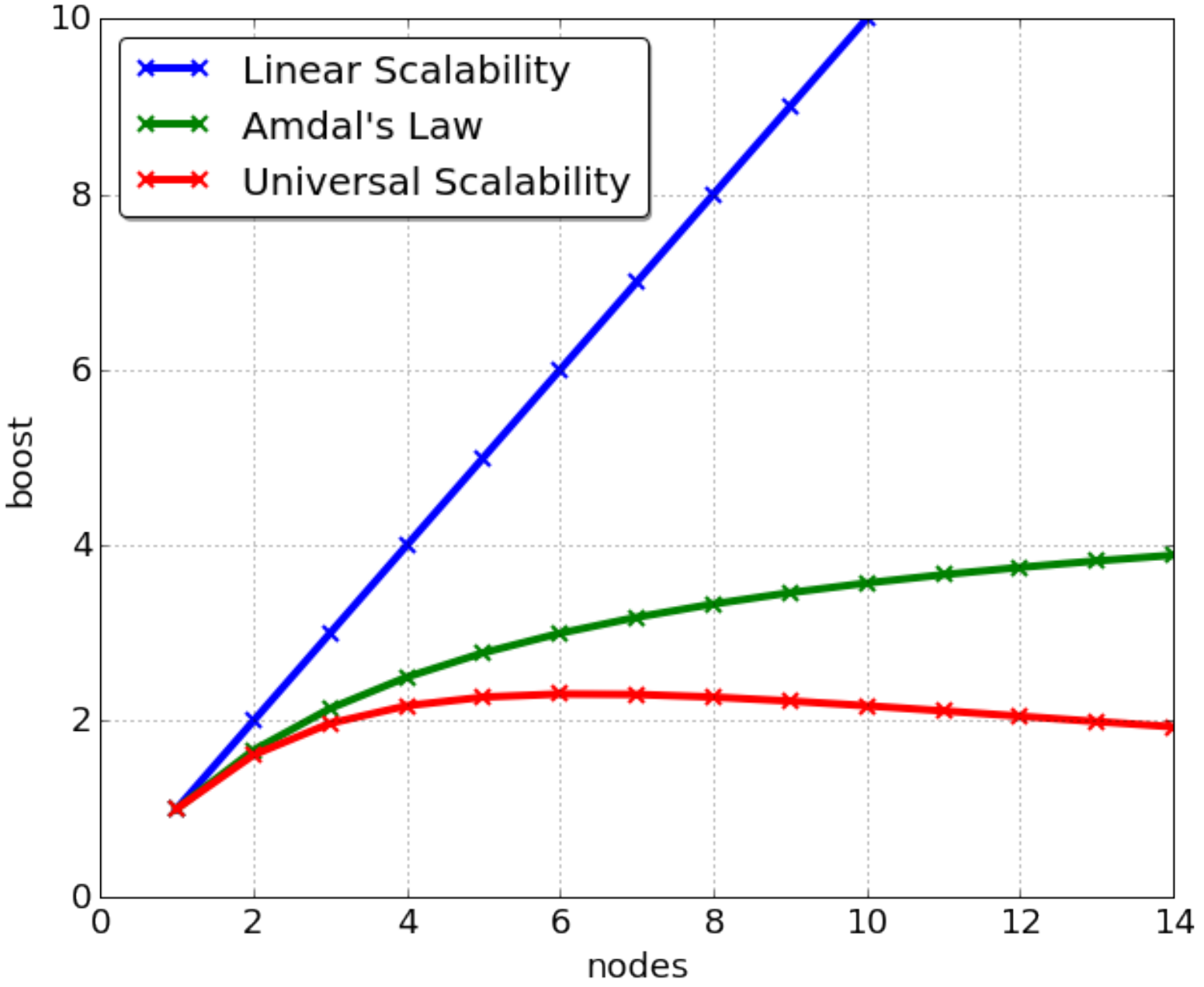
* Существуют фоновые процессы, которые взаимодействуют с данными и разделяемой памятью

Postgres PROFESSIONAL

План оптимизации БД



Математические модели



Линейное масштабирование

$$B(N) = c * N$$

N — Количество вычислителей (в нашем случае CPU)

B — Полученое ускорение

Закон Амдала

$$B(N) = \frac{N}{1 + a * (N - 1)}$$

a - не распараллеленный участок

Postgres PROFESSIONAL

Не распараллеленный участок

Не распараллеленный участок кода

Критическая секция

- участок кода, который происходит в эксклюзивном режиме доступа к данным или ресурсу

Общий закон масштабируемости

$$B(N) = \frac{N}{1 + a*(N-1) + b*N*(N-1)}$$

a — не распараллеленный участок

b — время, потраченное на синхронизацию исполнителей

Блокировки и синхронизация

Главные враги масштабируемости:

* Время проведеное в блокировке

* Синхронизация событий между исполнителями

Локи

```
db=# \d+ pg_locks
                        View "pg_catalog.pg_locks"
       Column        |   Type   | Modifiers | Storage  |
---------------------+----------+-----------+----------+
 locktype            | text     |           | extended |
 database            | oid      |           | plain    |
 relation            | oid      |           | plain    |
 page                | integer  |           | plain    |
 tuple               | smallint |           | plain    |
 virtualxid          | text     |           | extended |
 transactionid       | xid      |           | plain    |
 classid             | oid      |           | plain    |
 objid               | oid      |           | plain    |
 objsubid            | smallint |           | plain    |
 virtualtransaction  | text     |           | extended |
 pid                 | integer  |           | plain    |
 mode                | text     |           | extended |
 granted             | boolean  |           | plain    |
 fastpath            | boolean  |           | plain    |
```

https://wiki.postgresql.org/wiki/Lock_Monitoring

Postgres PROFESSIONAL

HWLocks

- Работают на уровне объектов
- Логические блокировки
- Детектор dead-lock

Локи в PostgreSQL

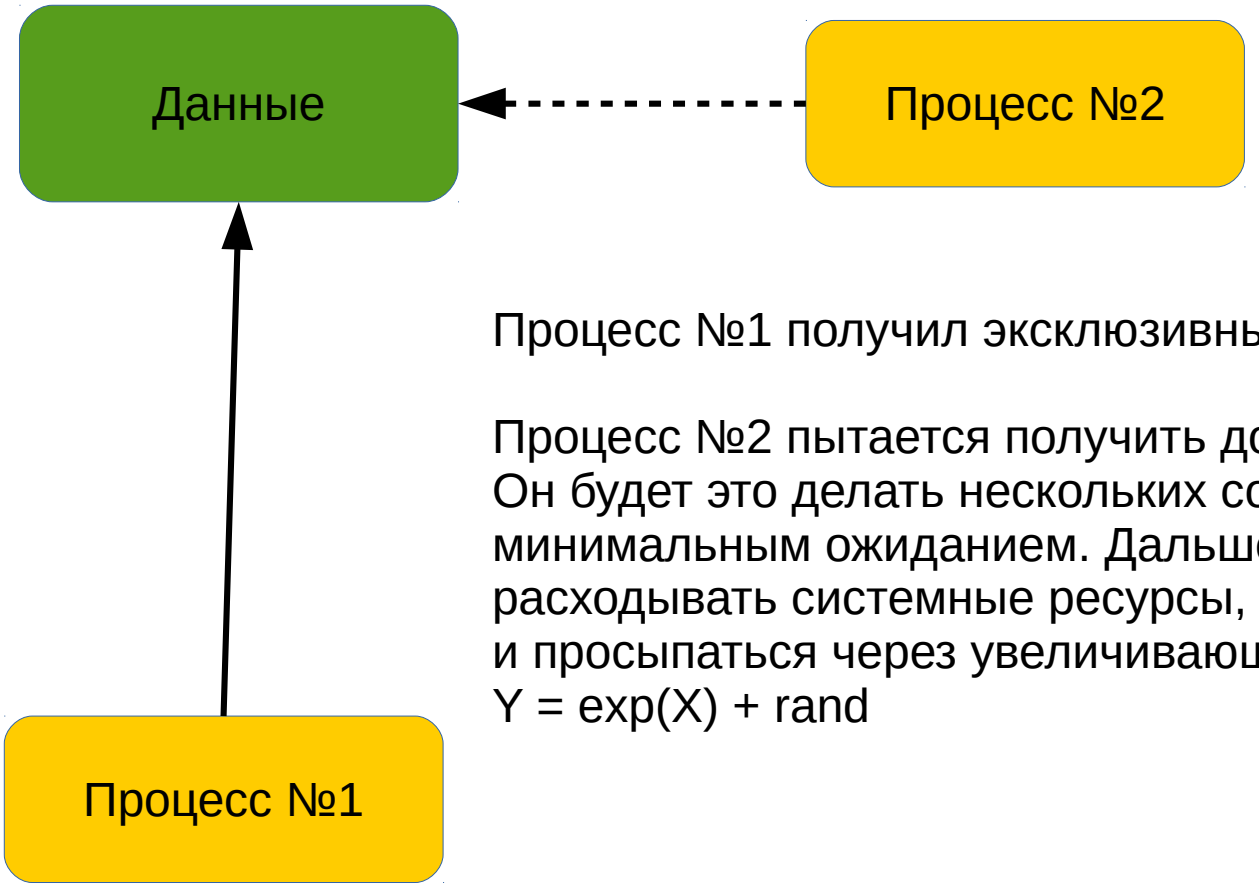
- Тяжелые локи (HWLock)
- Легкие локи (LWLock)
- Spin Lock

Также: Row-Level, Predicate, Advisory Locks

SpinLock

- Для коротких операций
- Только эксклюзивный режим
- Нет детектора dead-locks
- Нет очередей

SpinLock



Процесс №1 получил эксклюзивный доступ к объекту

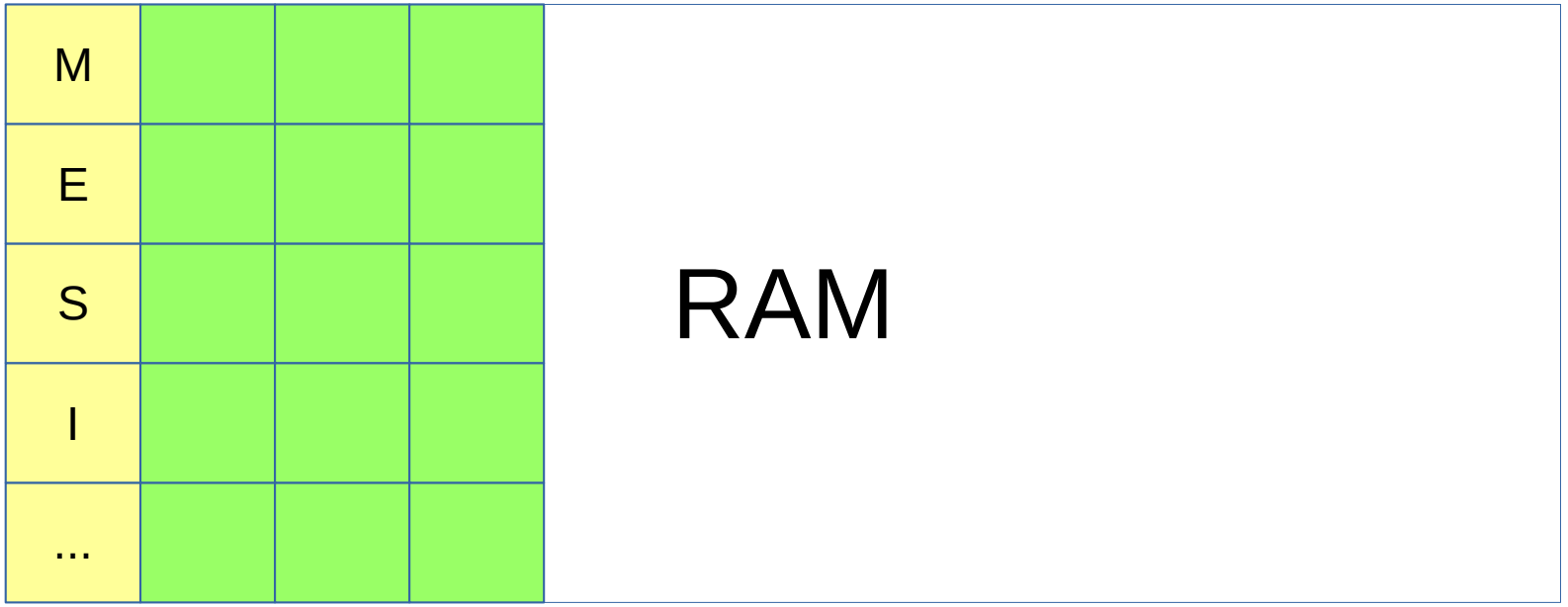
Процесс №2 пытается получить доступ к объекту. Он будет это делать нескольких сотен раз с минимальным ожиданием. Дальше, что бы не расходывать системные ресурсы, он будет засыпать и просыпаться через увеличивающиеся интервалы:
Y = exp(X) + rand

SpinLock: Цикл с Atomic TAS

```
int
s_lock {
  int spin = 0;
  while TAS(&lock) {
     SPIN_DELAY(); // nop
      if (++spin > MAX) {
        pg_usleep(...);
      }
  }
  ...
}
```

```
#define LOCKED 1

int
TAS(int* lockPtr) {
   int oldValue;

   atomically {
      oldValue = *lockPtr;
      *lockPtr = LOCKED;
   }

   return oldValue;
}
```

Atomic operation



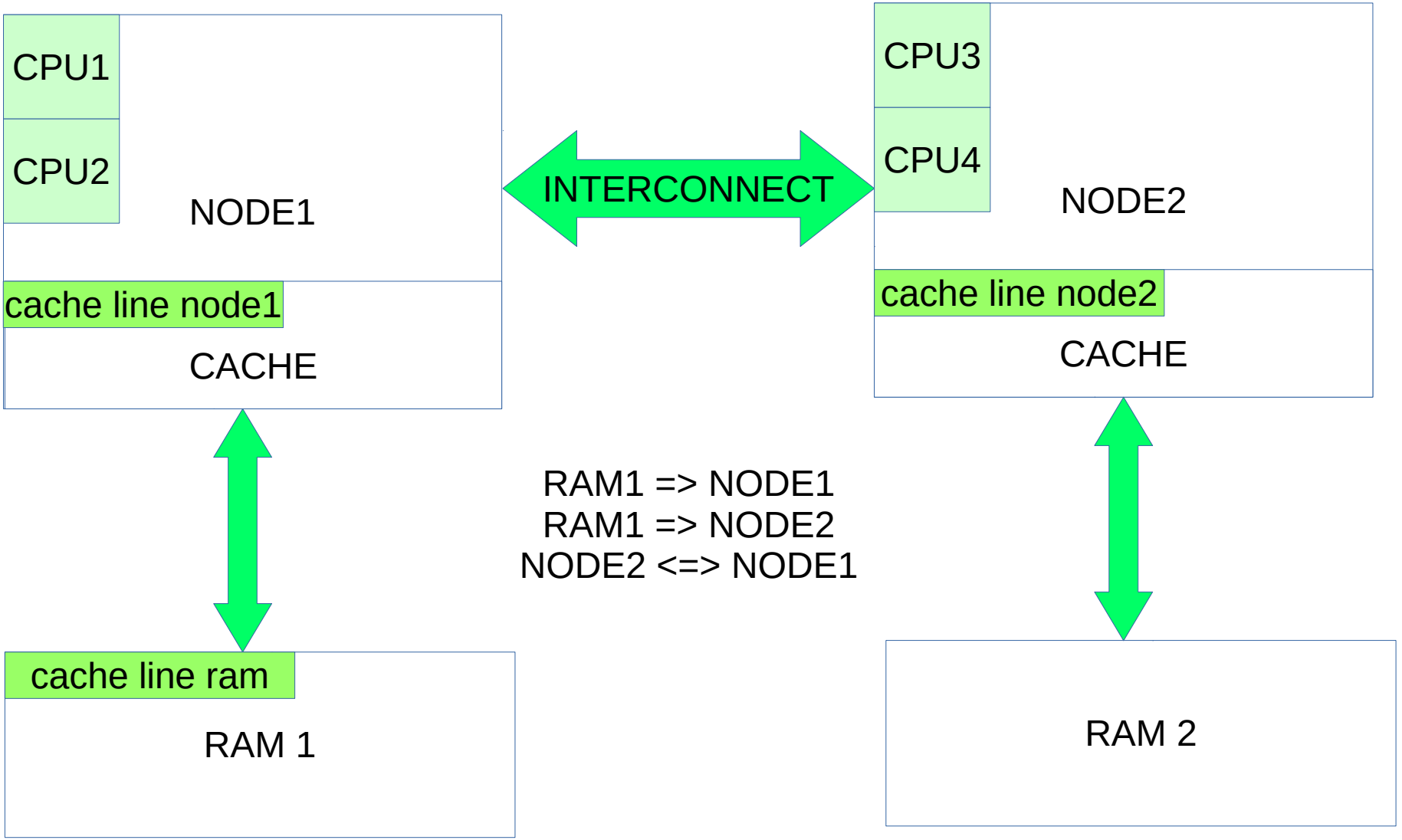
* CPU читает из основной памяти страницами cache-line

* Каждая cache-line кроме данных имеет тэг хранящее ее состояние

* Тэг меняет/читает CPU при обращении или записи

* Тэг может принимать значения: Modified, Exclusive, Shared, Invalid …

Cache coherency protocol

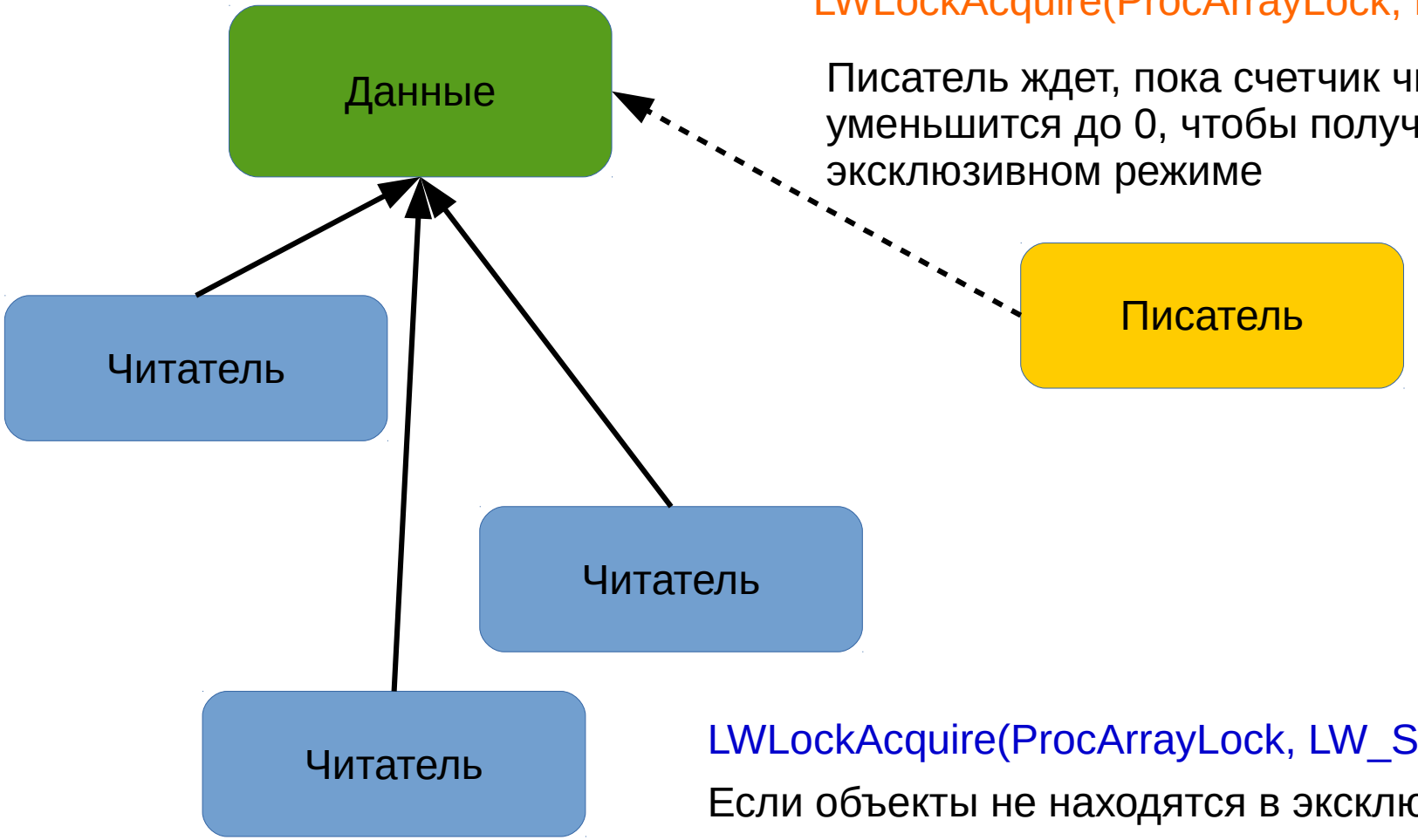


Postgres PROFESSIONAL

LWLocks

- Очереди: Shared / Exclusive
- Нет dead-lock детектора
- В 9.5 избавились от SpinLock

Shared и Exclusive: ProcArrayLock

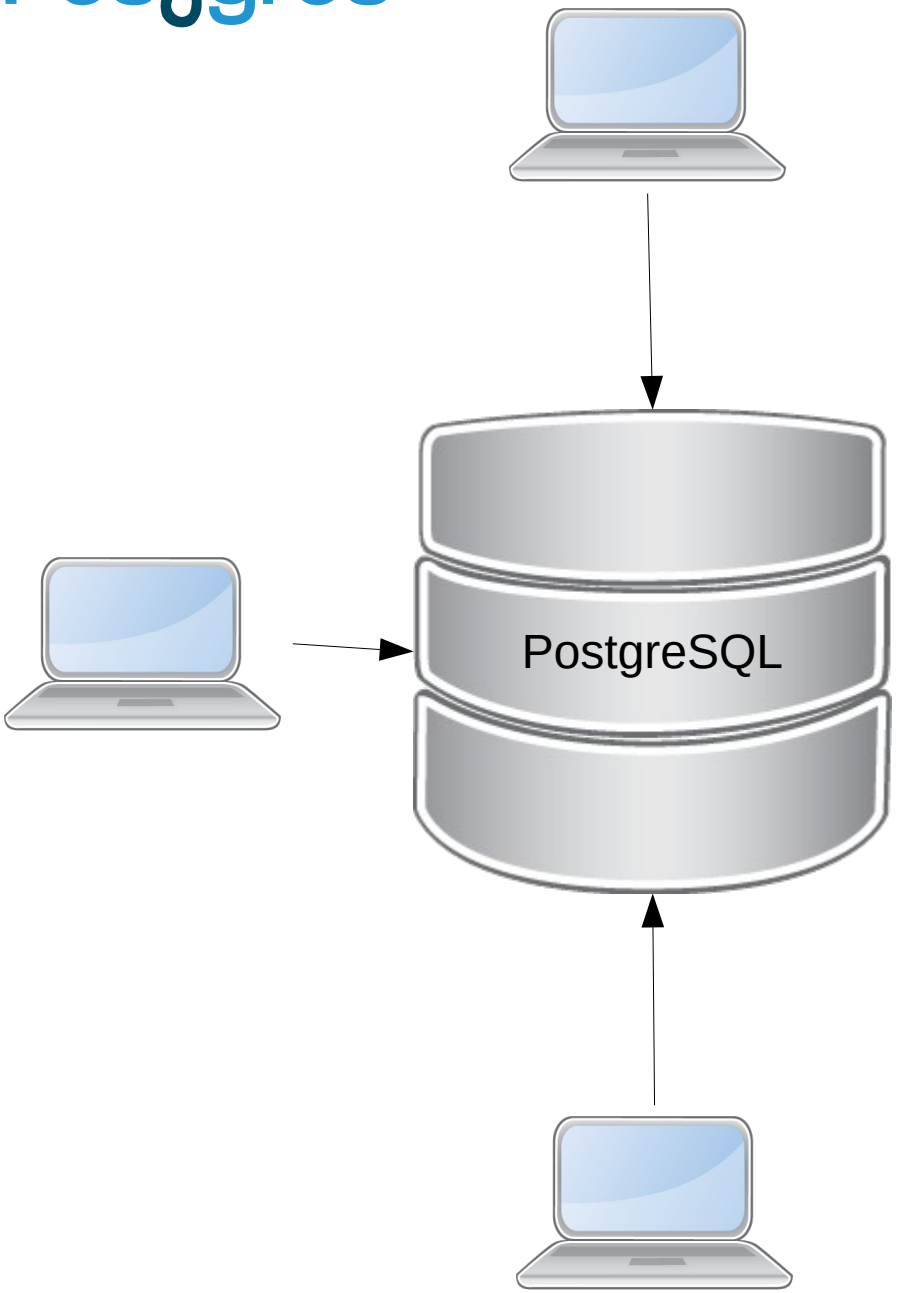


LWLockAcquire(ProcArrayLock, LW_EXCLUSIVE)

Писатель ждет, пока счетчик читателей не уменьшится до 0, чтобы получить доступ в эксклюзивном режиме

LWLockAcquire(ProcArrayLock, LW_SHARED);

Если объекты не находятся в эксклюзивном режиме, то читатели только инкрементируют счетчик читателей.



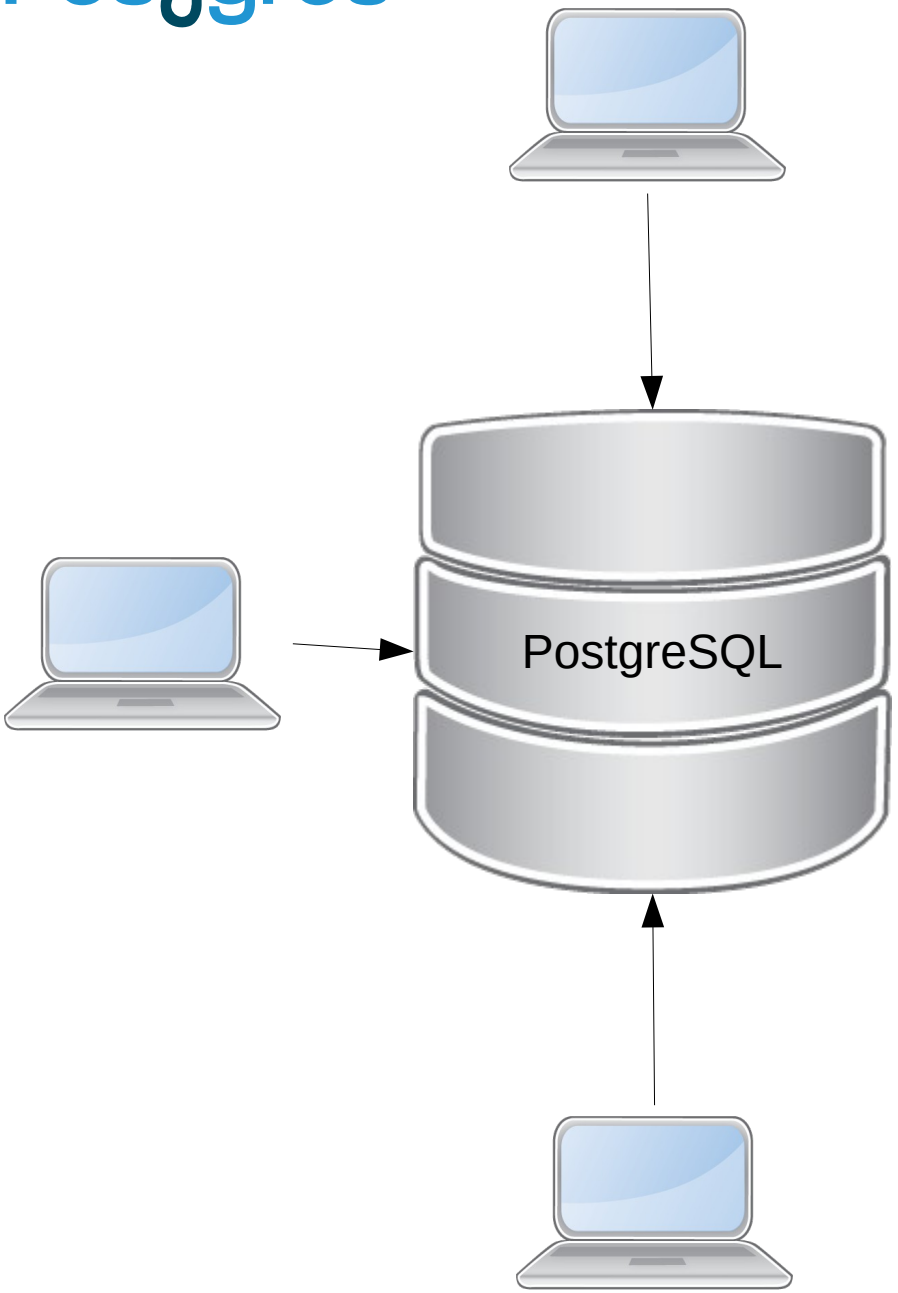
Каждый клиент порождает 1 процесс бакэнда:

Ваша система может быть нагружена так сильно, что получить ProcArrayLock в эксклюзивном режиме новый бакэнд не сможет.

Проблемы использования SHARED

file: **contrib/pg_buffercache/pg_buffercache_pages.c**

```
for (i = 0; i < NUM_BUFFER_PARTITIONS; i++)
    LWLockAcquire(BufMappingPartitionLockByIndex(i), LW_SHARED);
```



По каждому запросу составляется план выполнения, которому нужна информация по объектам и статистика:

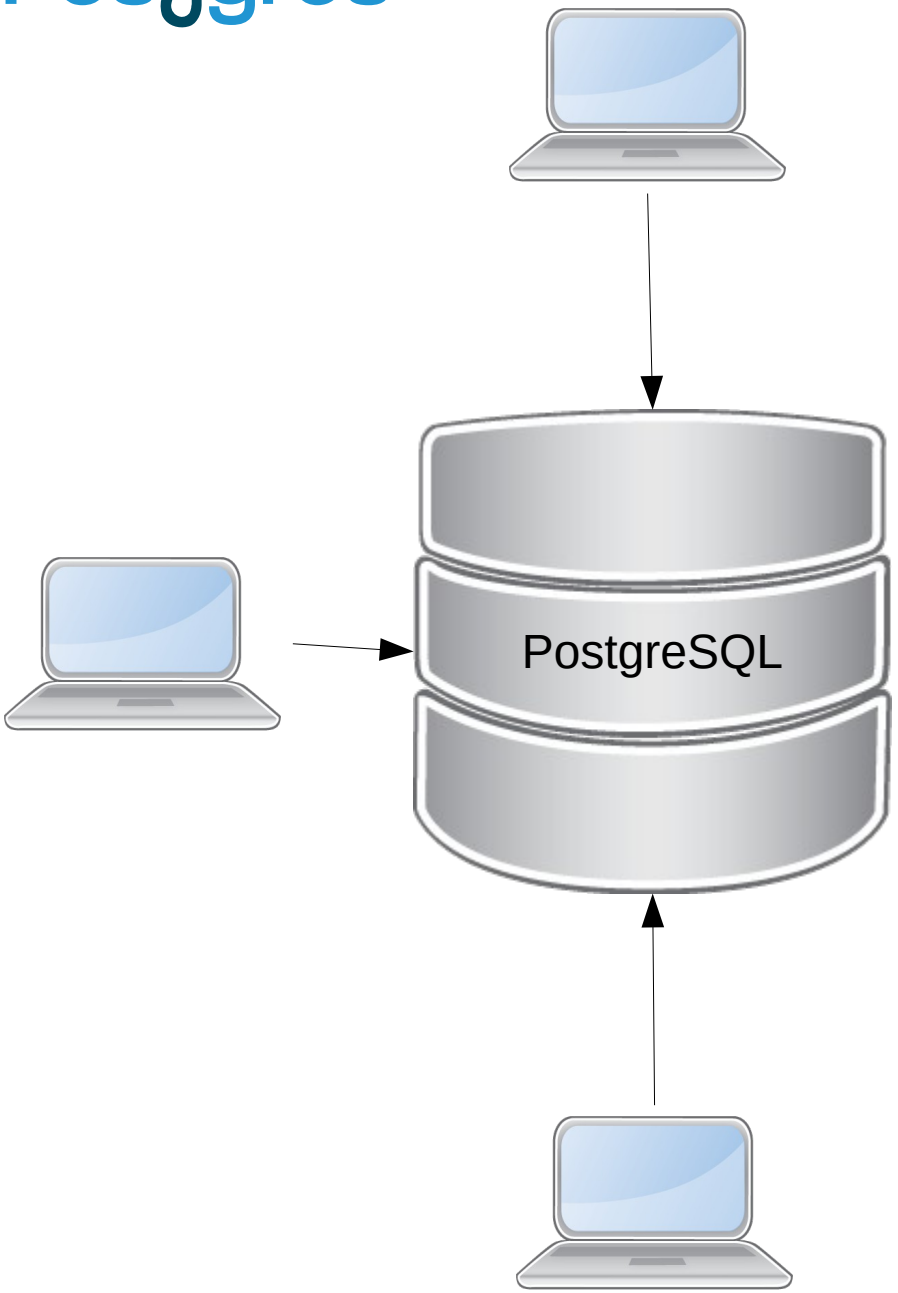
Локи на этом этапе мешают использовать наследование для построения партицирования:

Planning time >> Execution Time

Минутка рекламы

pathman:

- Автоматическое создание партиций
- Работает на уровне планера
- Hash - партицирование



Бакэнды взаимодействуют с данными, расположенными на дисках, через общую разделяемую памятью.

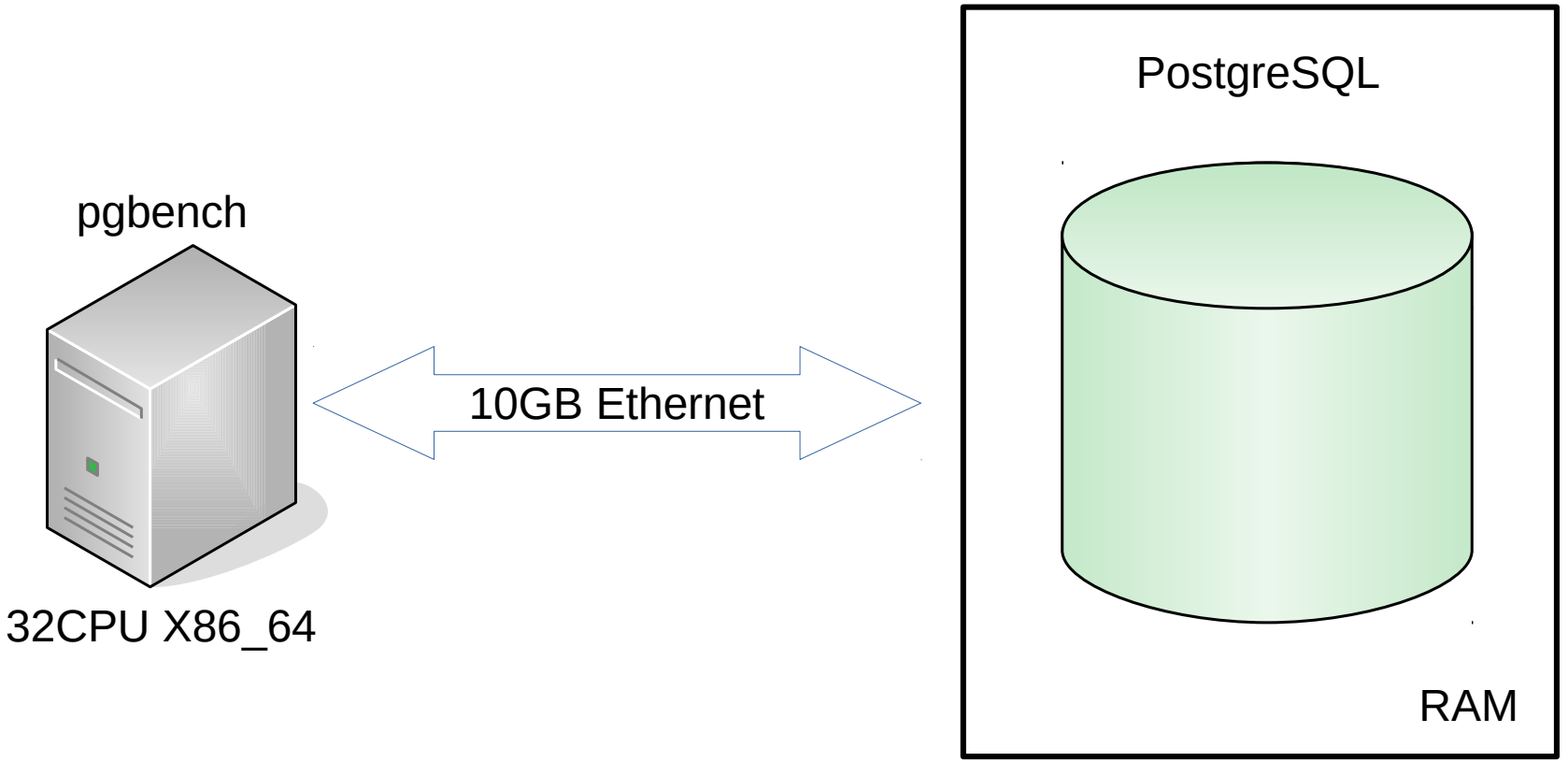
Бакэнд может писать и читать из разделяемой памяти.

Локи в BufferManager, является темой нашего дальнейшего расказа

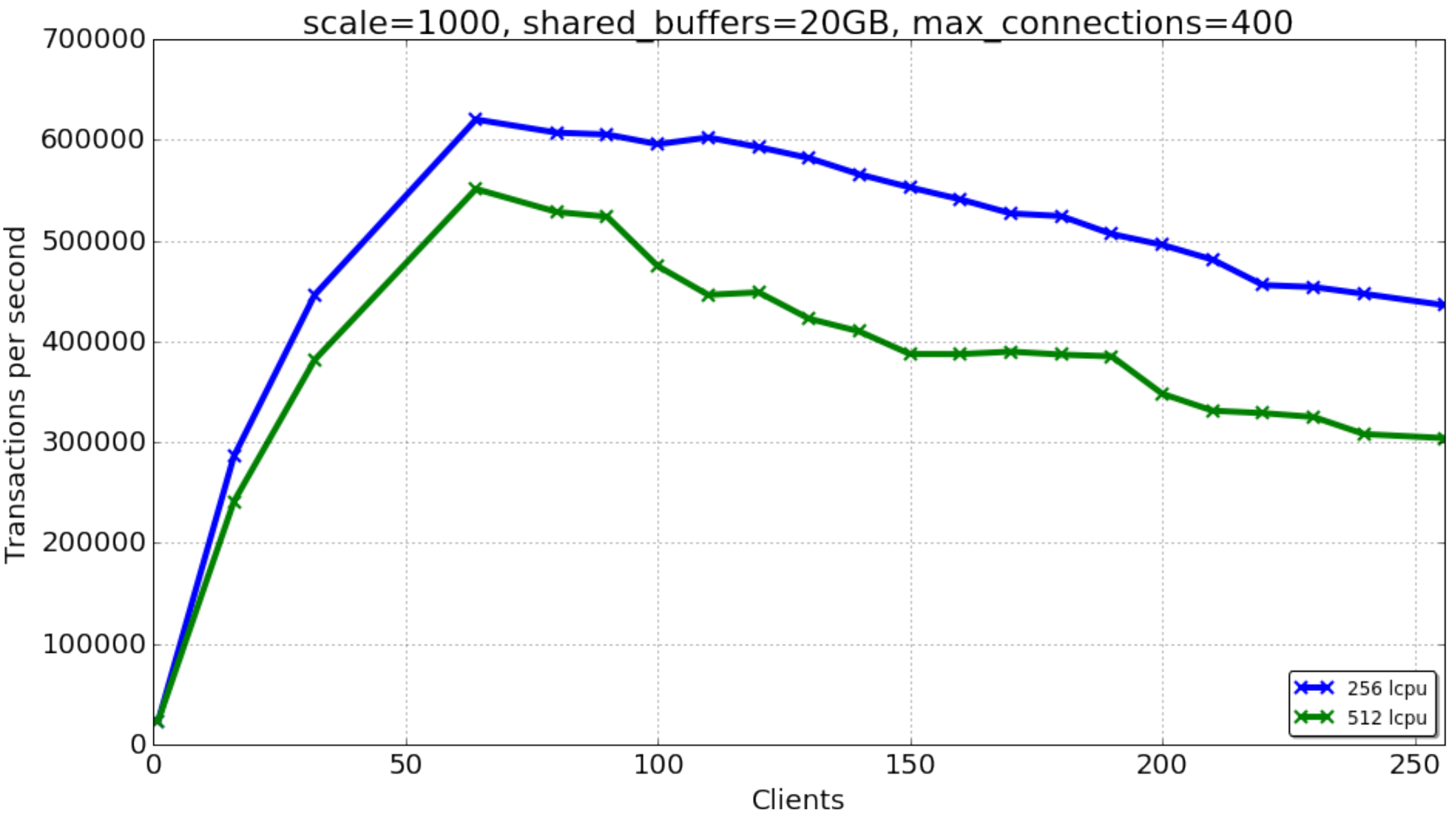
IBM E880

- 2 x 2U Ноды
- Ноды соединены «HyperConnect»
- Одна нода: 4 Сокета с Power8
- Один сокет: 8 Ядер (Core)
- Одно ядро: 8 логических ядер (SMT)

Схема тестирования



256 vs 512 LCPU



Инструменты: perf

https://perf.wiki.kernel.org

* perf record: запись событий
* perf report: построение отчета
* perf top: просмотр отчета «live»

Инструменты: FlameGraph

http://www.brendangregg.com/flamegraphs.html

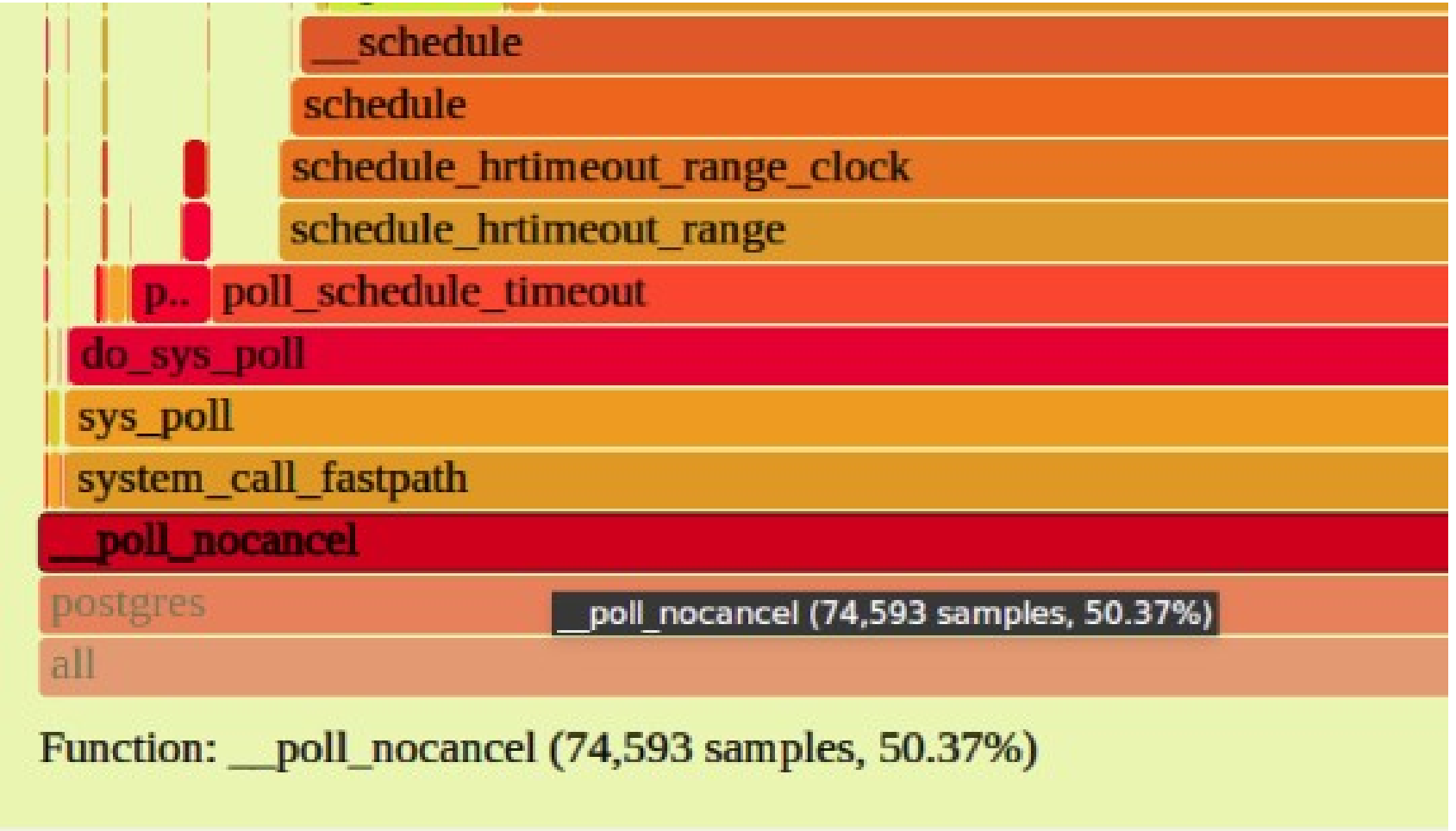
* Linux: perf, SystemTap, and ktap

* Solaris, illumos, FreeBSD: Dtrace

* Mac OS X: DTrace and Instruments

* Windows: Xperf.exe

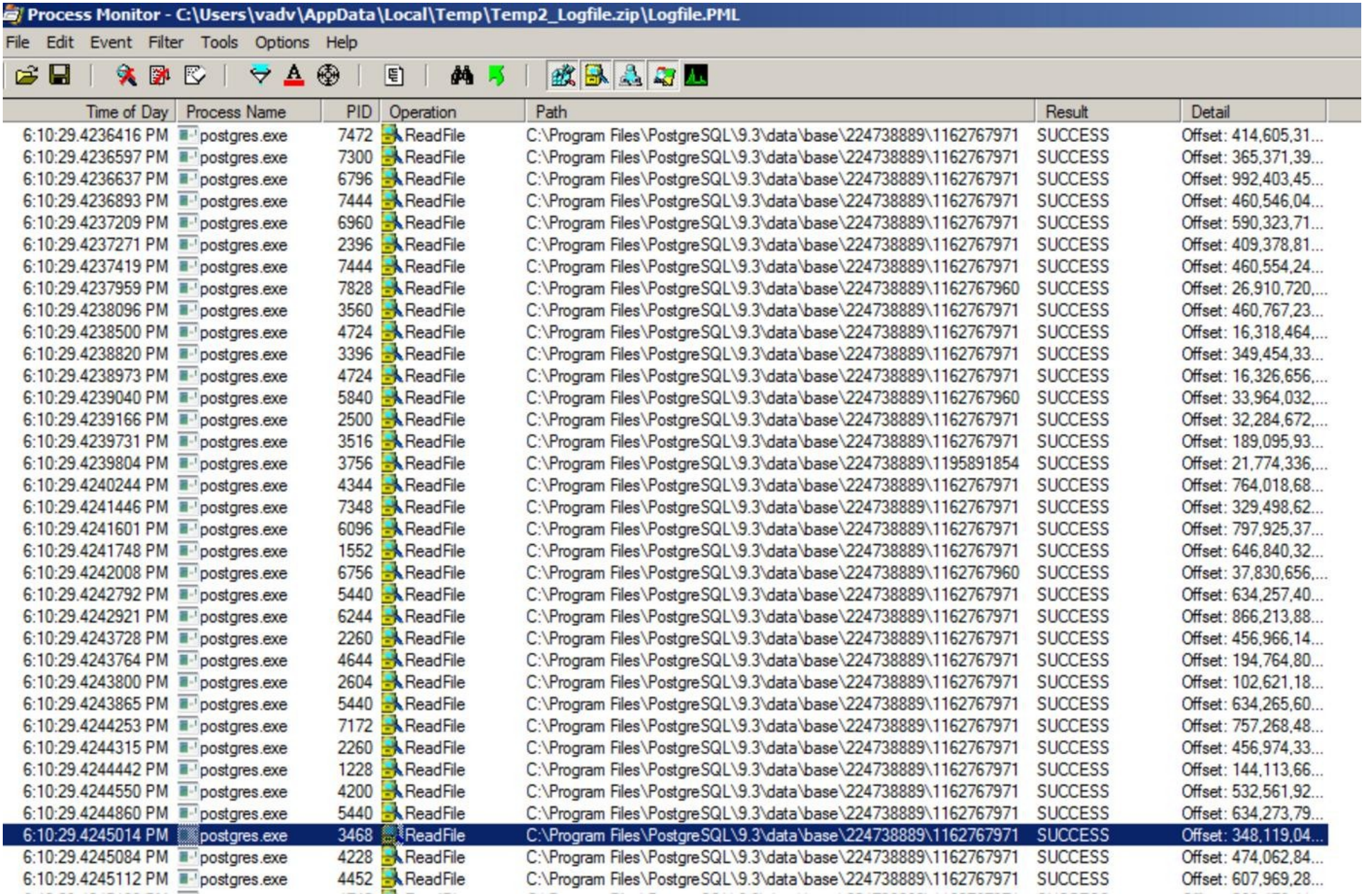
Инструменты: FlameGraph



Инструменты: FlameGraph

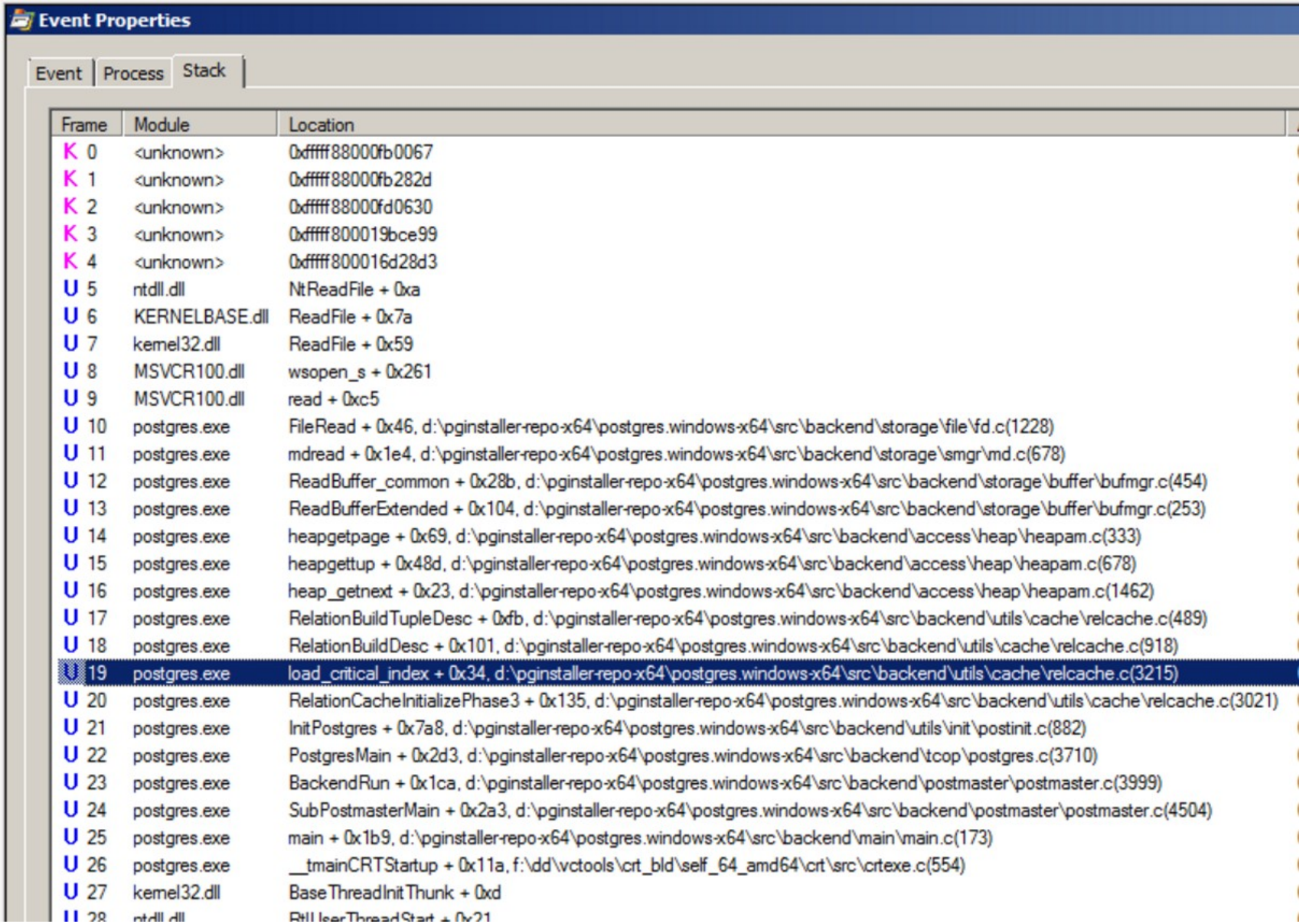
```
$ git clone https://github.com/brendangregg/FlameGraph

$ perf record -F 100 -a -g -u postgres

$ perf script | ./stackcollapse-perf.pl > out.perf-folded

$ ./flamegraph.pl out.perf-folded > perf-kernel.svg
```

Инструменты: Process Monitor



Time of Day	Process Name	PID	Operation	Path	Result	Detail
6:10:29.4236416 PM	postgres.exe	7472	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 414,605,31...
6:10:29.4236597 PM	postgres.exe	7300	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 365,371,39...
6:10:29.4236637 PM	postgres.exe	6796	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 992,403,45...
6:10:29.4236893 PM	postgres.exe	7444	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 460,546,04...
6:10:29.4237209 PM	postgres.exe	6960	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 590,323,71...
6:10:29.4237271 PM	postgres.exe	2396	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 409,378,81...
6:10:29.4237419 PM	postgres.exe	7444	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 460,554,24...
6:10:29.4237959 PM	postgres.exe	7828	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767960	SUCCESS	Offset: 26,910,720,...
6:10:29.4238096 PM	postgres.exe	3560	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 460,767,23...
6:10:29.4238500 PM	postgres.exe	4724	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 16,318,464,...
6:10:29.4238820 PM	postgres.exe	3396	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 349,454,33...
6:10:29.4238973 PM	postgres.exe	4724	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 16,326,656,...
6:10:29.4239040 PM	postgres.exe	5840	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767960	SUCCESS	Offset: 33,964,032,...
6:10:29.4239166 PM	postgres.exe	2500	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 32,284,672,...
6:10:29.4239731 PM	postgres.exe	3516	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 189,095,93...
6:10:29.4239804 PM	postgres.exe	3756	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1195891854	SUCCESS	Offset: 21,774,336,...
6:10:29.4240244 PM	postgres.exe	4344	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 764,018,68...
6:10:29.4241446 PM	postgres.exe	7348	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 329,498,62...
6:10:29.4241601 PM	postgres.exe	6096	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 797,925,37...
6:10:29.4241748 PM	postgres.exe	1552	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 646,840,32...
6:10:29.4242008 PM	postgres.exe	6756	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767960	SUCCESS	Offset: 37,830,656,...
6:10:29.4242792 PM	postgres.exe	5440	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 634,257,40...
6:10:29.4242921 PM	postgres.exe	6244	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 866,213,88...
6:10:29.4243728 PM	postgres.exe	2260	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 456,966,14...
6:10:29.4243764 PM	postgres.exe	4644	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 194,764,80...
6:10:29.4243800 PM	postgres.exe	2604	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 102,621,18...
6:10:29.4243865 PM	postgres.exe	5440	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 634,265,60...
6:10:29.4244253 PM	postgres.exe	7172	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 757,268,48...
6:10:29.4244315 PM	postgres.exe	2260	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 456,974,33...
6:10:29.4244442 PM	postgres.exe	1228	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 144,113,66...
6:10:29.4244550 PM	postgres.exe	4200	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 532,561,92...
6:10:29.4244860 PM	postgres.exe	5440	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 634,273,79...
6:10:29.4245014 PM	postgres.exe	3468	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 348,119,04...
6:10:29.4245084 PM	postgres.exe	4228	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 474,062,84...
6:10:29.4245112 PM	postgres.exe	4452	ReadFile	C:\Program Files\PostgreSQL\9.3\data\base\224738889\1162767971	SUCCESS	Offset: 607,969,28...

Инструменты: Process Monitor



Postgres PROFESSIONAL

Инструменты: perf

perf top -F 100 -u postgres

- 32.10% postgres [.] **s_lock**
- 7.77% postgres [.] GetSnapshotData
- 2.64% postgres [.] AllocSetAlloc
- 1.40% postgres [.] hash_search_with_hash_value
- 1.37% postgres [.] base_yyparse
- 1.36% postgres [.] SearchCatCache
- 1.32% postgres [.] PinBuffer
- 1.23% postgres [.] LWLockAcquire
- 1.05% postgres [.] palloc
- 1.01% postgres [.] ReadBuffer_common
- 0.99% postgres [.] LWLockRelease

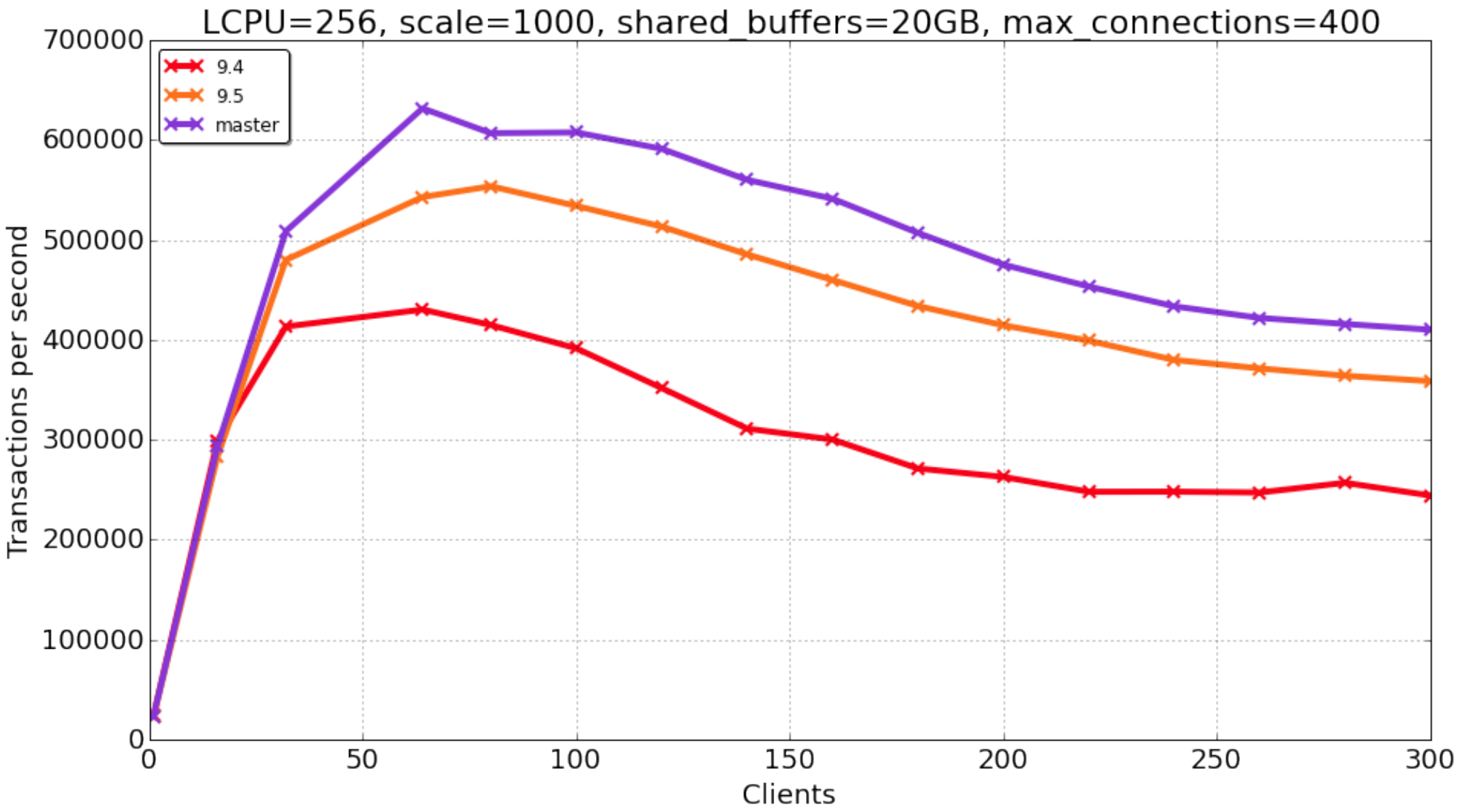
Инструменты: gdb

https://www.gnu.org/software/gdb/

gdb --batch --command=gdb.script —pid=XXX

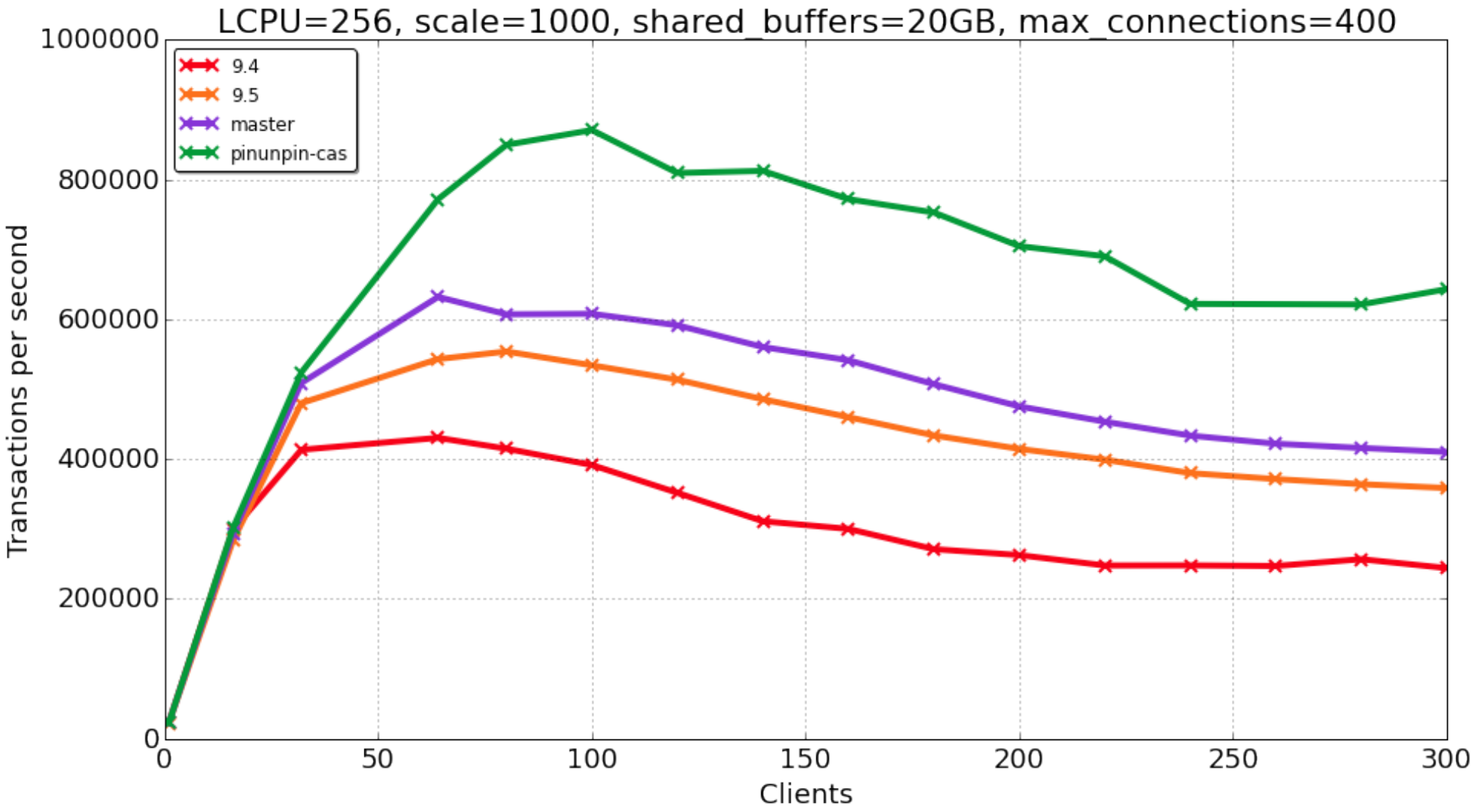
- #0 0x00003fffac40a858 in ___newselect_nocancel () from /lib64/power8/libc.so.6
- #1 0x00000000106105f0 in **pg_usleep** (microsec=<optimized out>) at pgsleep.c:53
- #2 0x00000000103e5f18 in **s_lock** (lock=0x3fe607980be0, file=0x10718398 "bufmgr.c", line=<optimized out>) at s_lock.c:110
- #3 0x00000000103aea10 in **UnpinBuffer** (buf=0x3fe607980bc0, fixOwner=1 '\001') at bufmgr.c:1540
- #4 0x00000000103b4910 in **ReleaseAndReadBuffer** (buffer=<optimized out>, relation=0x3fe6067073e0, blockNum=<optimized out>) at bufmgr.c:1401

9.4 vs 9.5 vs master



9.4 LWLocks: SpinLock
9.5 LWLocks: Atomic

Патчи Andres Freund



Патч: PinBuffer via CAS, UnPinBuffer via FetchAndAdd

Результаты оптимизаций

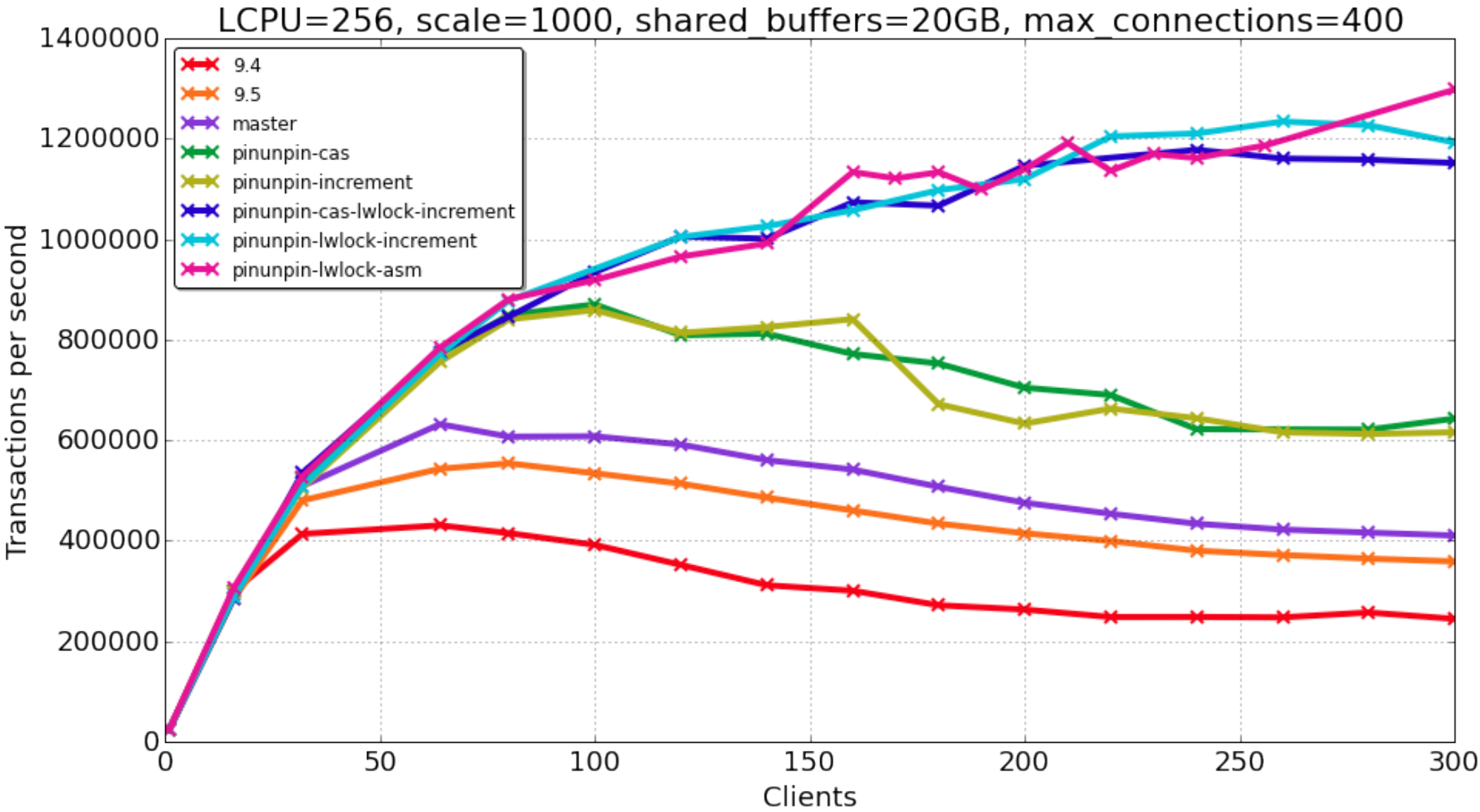
До:

- 33.48% postgres [.] s_lock
- 2.51% postgres [.] GetSnapshotData
- 1.82% postgres [.] PinBuffer

После:

- 13.75% postgres [.] GetSnapshotData
- 4.88% postgres [.] AllocSetAlloc
- 2.47% postgres [.] LWLockAcquire

PostgresPro патчи для 9.5



pinunpin-cas – PinBuffer: CAS
pinunpin-increment – PinBuffer: оптимистичный AtomicAdd
pinunpin-cas-lwlock-increment – PinBuffer: CAS, LWLockAttemptLock оптимистичный AtomicAdd
pinunpin-lwlock-increment – PinBuffer и LWLockAttemptLock: оптимистичный AtomicAdd
pinunpin-lwlock-asm – PinBuffer и LWLockAttemptLock: оптимизированный asm

Не распараллеленный участок

PostgreSQL 9.4	4.3 %
PostgreSQL 9.5	2.6 %
PostgreSQL 9.6	2.3 %
PostgresPro 9.5	1.6 %

Postgres PROFESSIONAL

Benchmark

```
Datum
incr_test(PG_FUNCTION_ARGS)
{
    int32 i = PG_GETARG_INT32(0),
          cnt;

    check_init();

    for (cnt = 0; cnt < 1000000; cnt++)
    {
        pg_atomic_add_fetch_u32(&atomics[i], 1);
    }

    PG_RETURN_VOID();
}
```

Postgres PROFESSIONAL

Мониторинг LWLock

pg_stat_wait:

- Профилирование
- История
- Трейс query

Postgres PROFESSIONAL

Время попробовать это!

http://repo.postgrespro.ru

* OpenSource

* Совместимо с vanilla

* Мониторинг с Zabbix из коробки!

В зале уже есть довольные пользователи наших патчей :)

Postgres PROFESSIONAL

Спасибо за внимание!

Postgres PROFESSIONAL

Вопросы?

Ссылки

http://habrahabr.ru/company/postgrespro/blog/

https://events.linuxfoundation.org/sites/events/files/slides/linuxcon-2014-locking-final.pdf

http://www.postgresql.eu/events/sessions/pgconfeu2015/session/1080-scaling-up-postgresql/

http://www.slideshare.net/chris1adkin/super-scaling-singleton-inserts-53947279

http://habrahabr.ru/post/190862/

http://www.slideshare.net/rivitli/waits-monitoring-in-postgresql

http://exadat.co.uk/2015/01/20/large-memory-pages-how-they-work-and-the-logcache_access-spinlock/

http://habrahabr.ru/company/ifree/blog/196548/