

ЭКСПЕРИМЕНТЫ С TIMESCALEDB ZABBIX И POSTGRESQL С АВТОМАТИЧЕСКИМ ПАРТИЦИРОВАНИЕМ



Гуцин Андрей Николаевич
ZABBIX Инженер технической поддержки

ZABBIX 2019
Conference
RUSSIA

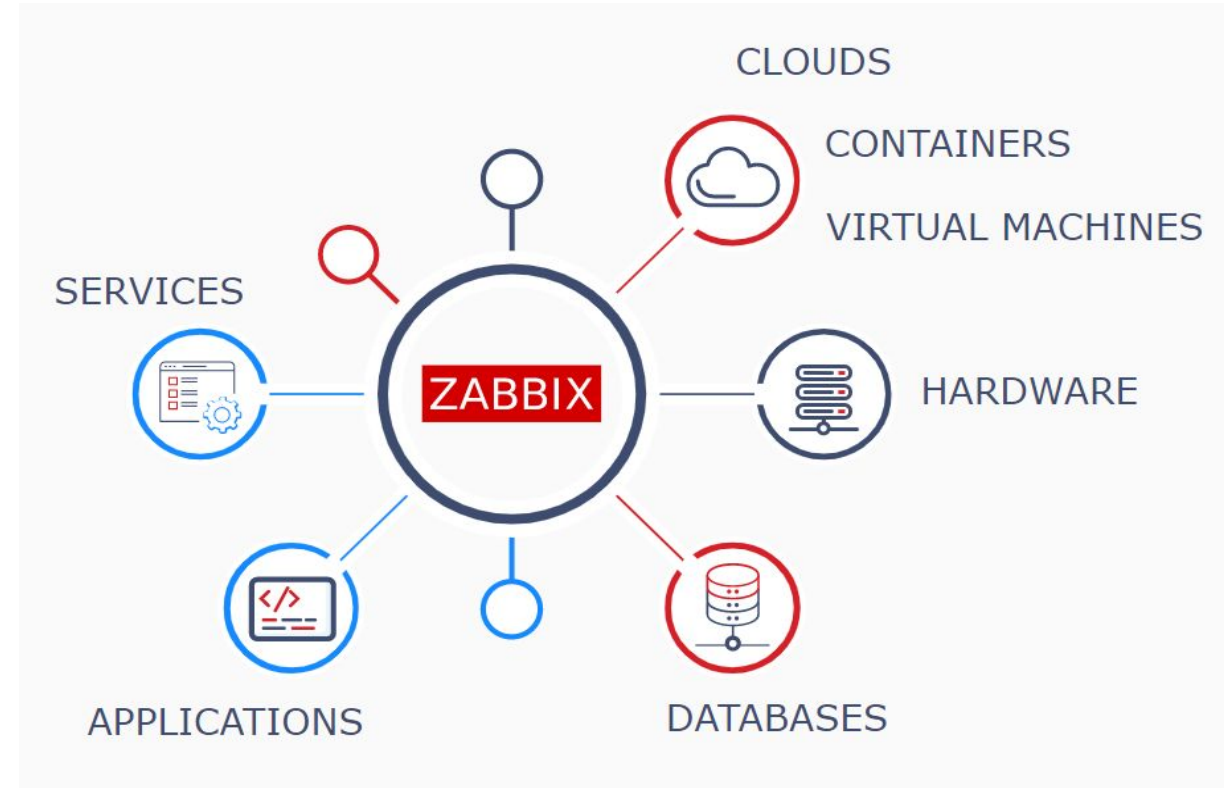
ПРИЧИНЫ ПЛОХОЙ ПРОИЗВОДИТЕЛЬНОСТИ

ZABBIX 2019
Conference
RUSSIA

Первый вопрос проблем производительности

Быстрый сбор и обработка.

Хорошая система мониторинга должна получать, обрабатывать и сохранять получаемые данные очень быстро.

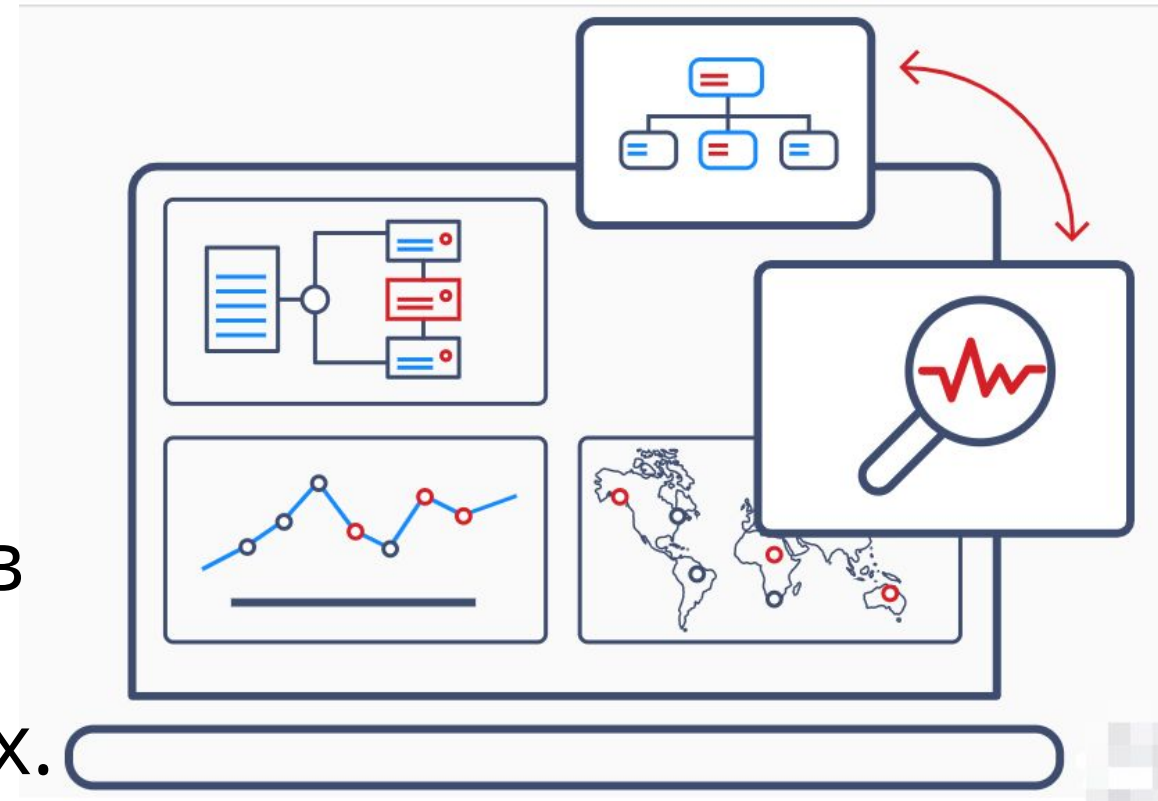


Второй вопрос проблем производительности

Хранение истории.

Хорошая система мониторинга должна предоставлять удобный доступ к огромным массивам истории.

История активно используется в отчетах, графиках, триггерах и вычисляемых элементах данных.

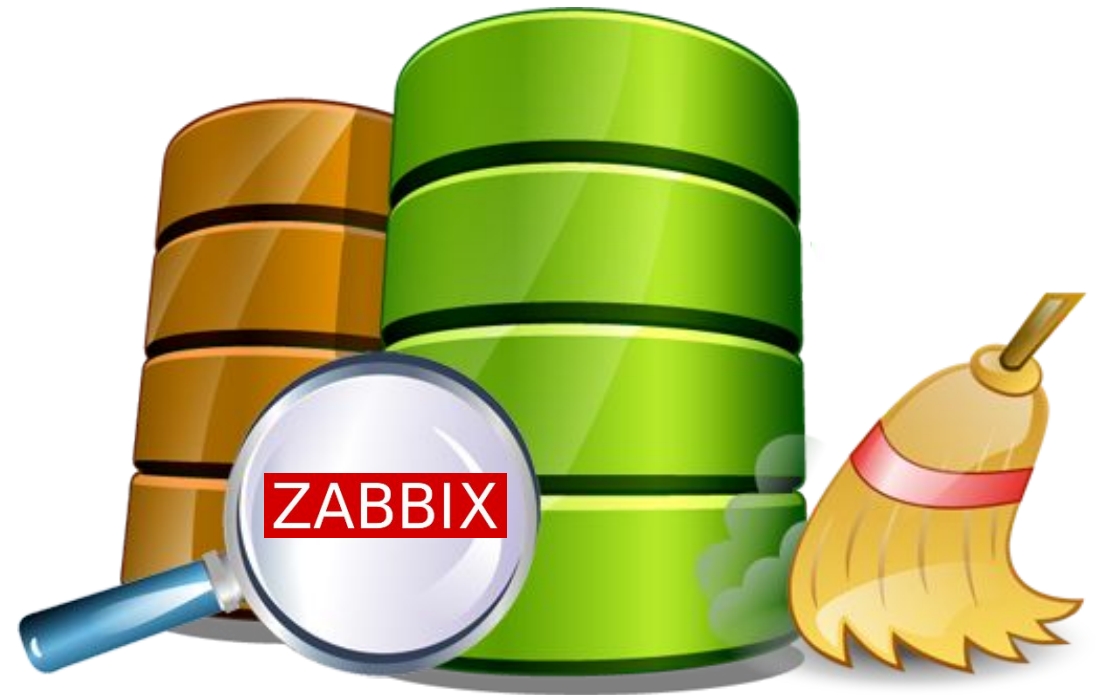


Третий вопрос проблем производительности

Очистка истории.

Хорошая система мониторинга должна хранить исторические данные. И время от времени их удалять.

Процедура удаления истории влияет на производительность очень сильно.



КАК ИХ РЕШИТЬ ?

ZABBIX 2019
Conference
RUSSIA

- Сбор и обработка данных
- Использование истории в триггерах, графиках, вычисляемых элементах данных.

Кэши Zabbiх и Базы Данных.

Кэширование значений –
обработка данных и
вычисление триггеров

Кэш конфигурации -
Сбор данных

Буффер значений перед
записью в Базу данных

Другие вспомогательные
кэши



MySQL
PostgreSQL
Oracle
DB2

Innodb buffer pool

Shared buffers

Effective cache size

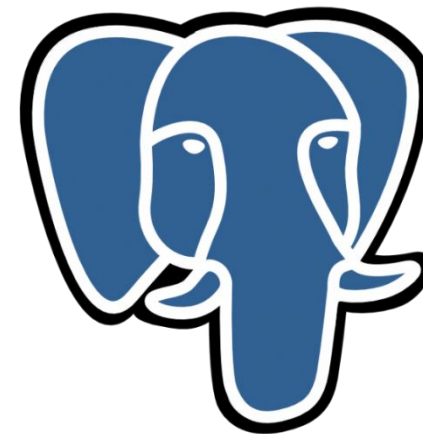
Shared pool... и прочие

Тюнинг параметров базы данных обязателен!
Как при первом запуске, так и на регулярной основе.

Пример конфигурации PostgreSQL

```
max_connections = 600
shared_buffers = 32GB
effective_cache_size = 96GB
work_mem = 7MB
maintenance_work_mem = 2GB
effective_io_concurrency = 200
max_worker_processes = 59
max_parallel_workers_per_gather = 24
max_parallel_workers = 48
wal_buffers = 16MB
max_wal_size = 16GB
min_wal_size = 4GB
checkpoint_completion_target = 0.9
random_page_cost = 1.1
default_statistics_target = 500 # range 1-10000
```

```
# TimescaleDB part
shared_preload_libraries = '/usr/pgsql-10/lib/timescaledb.so'
timescaledb.max_background_workers = 80
timescaledb.last_tuned = '2019-06-20T10:22:41+03:00'
timescaledb.last_tuned_version = '0.6.0'
```



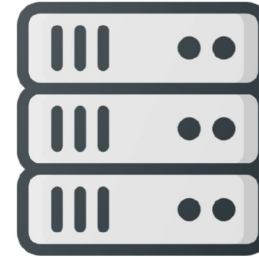
PostgreSQL



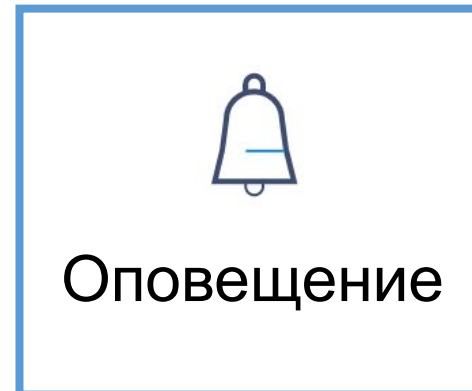
Производительность Базы данных



MySQL
PostgreSQL
Oracle
DB2



Zabbix Server



Производительность Базы данных

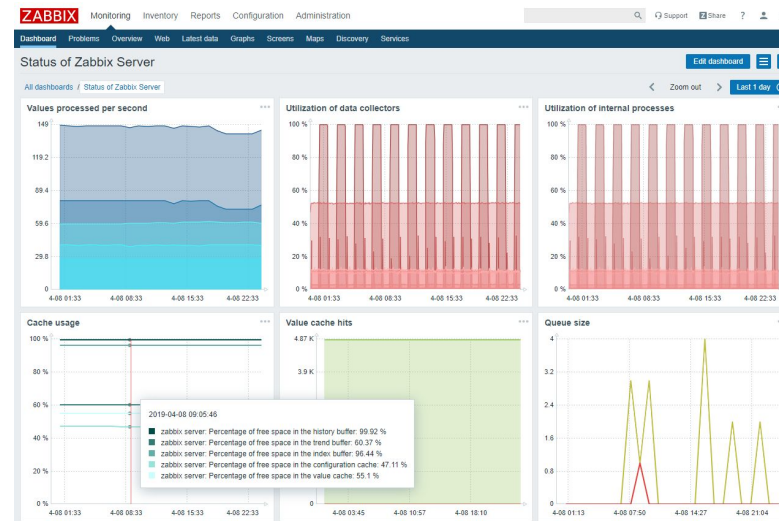


MySQL
PostgreSQL
Oracle
DB2

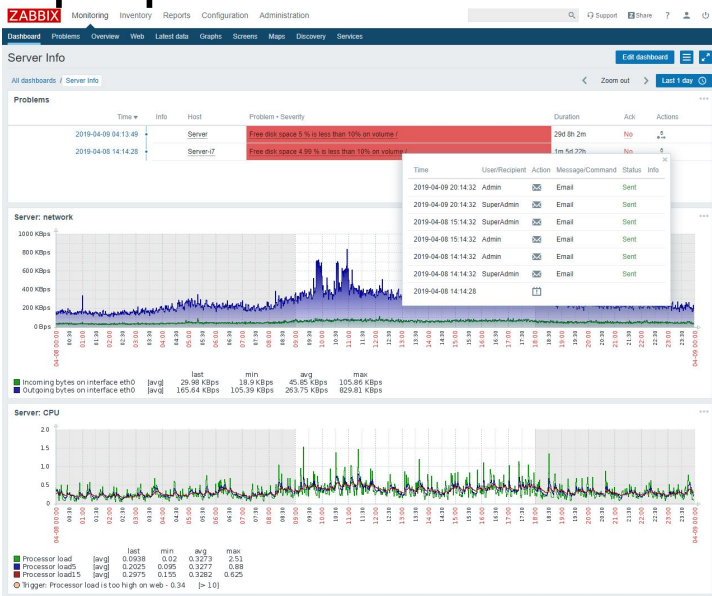


Zabbix Server

Веб интерфейс Zabbix



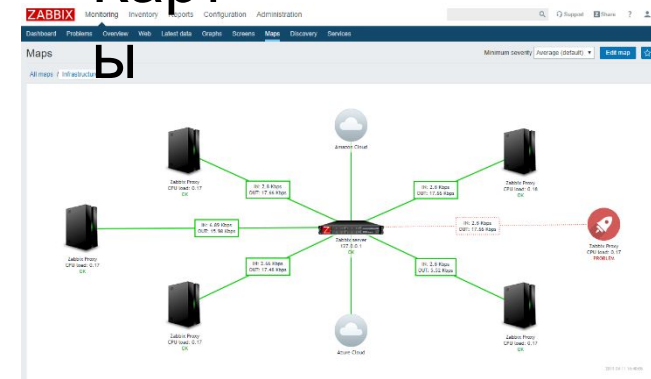
Графики



События и проблемы

Time	Severity	Info	Host	Problem	Duration	Ack	Actions	Tags
12:19:07	Warning		RTS-MT-WIFI	Interface RTS-MT-WIFI-CAPS-MAN-2.4GHz_S0N1: High bandw...	3h 25m 7s	No		
12:05:14	Warning		node-2	Processor load is too high on node-2	3h 30m	No		
11:19:16	Critical		built-server	CPU2 Temp Critical [91]	4h 24m 59s	No		
10:56:55	Average		Brother MFC-L2700DW	Drum Unit is low on printer Brother MFC-L2700DW	4h 47m 19s	No		
09:19:06	Warning		built-server	Disk array cache controller is in warning state	6h 25m 8s	No		
2019-04-08 11:17:50	Average		built-server	Free disk space is less than 12% on volume /	1d 4h 31m	Yes		
2019-04-08 10:44:12	Average		RTS-MT-WIFI					
2019-04-08 10:04:20	Information		node-6					
2019-04-08 09:24:45	Warning		FreeBSD 4.2					

Карты

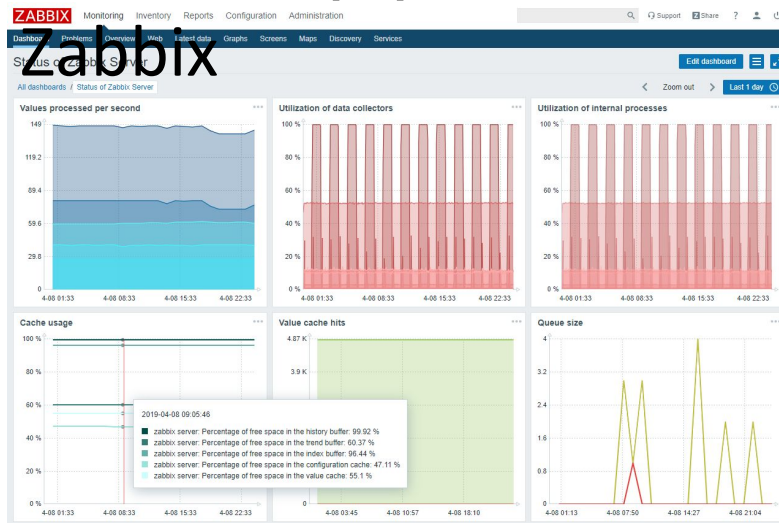


Производительность Базы данных



Zabbix Server

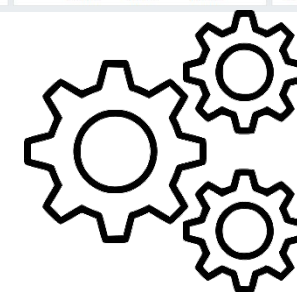
Веб интерфейс



MySQL
PostgreSQL
Oracle
DB2



Grafana

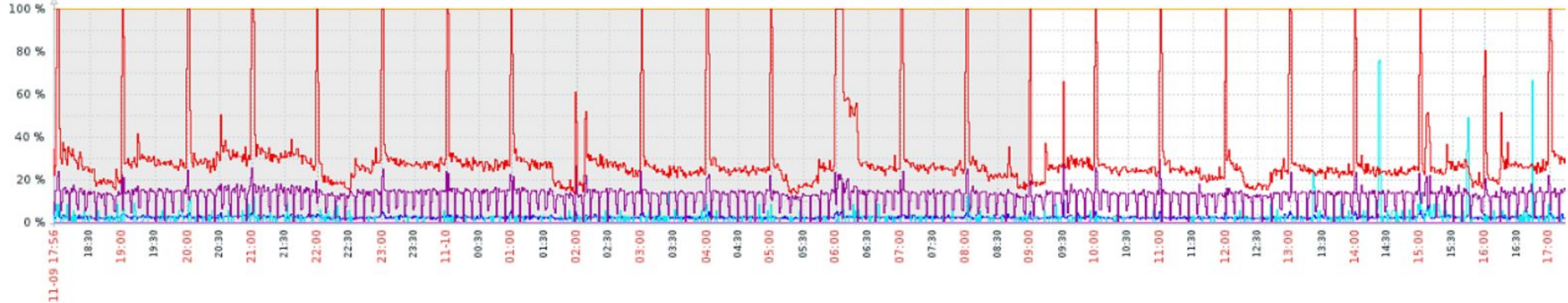


Zabbix API

Третий вызов производительности

Очистка истории = Zabbix Housekeeper Когда стоит **отключить** housekeeper?

Zabbix server: Zabbix internal process busy % (1d)



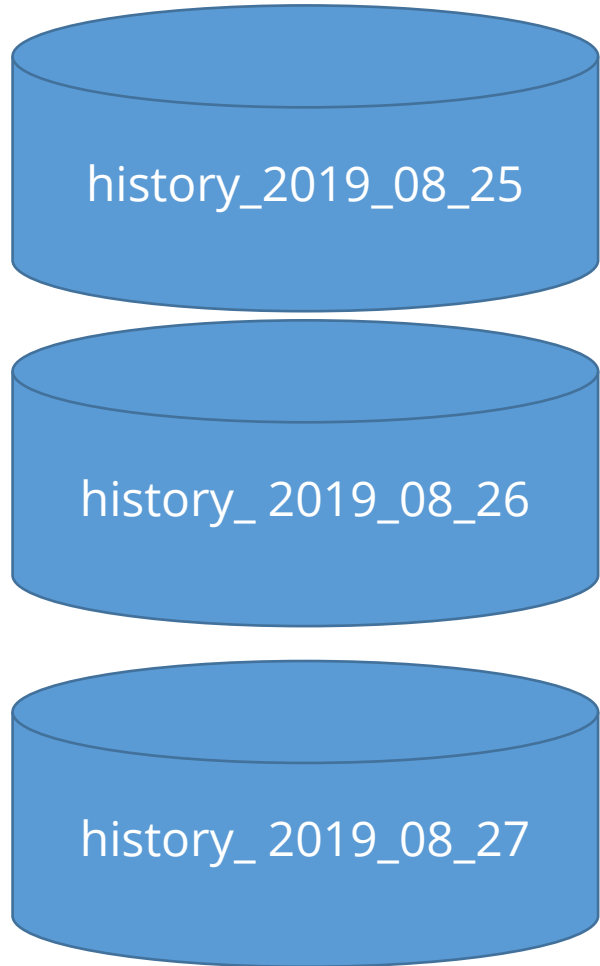
	last	min	avg	max
Zabbix busy timer processes, in %	[avg] 0 %	0 %	0.03 %	0.86 %
Zabbix busy escalator processes, in %	[avg] 2.76 %	1.25 %	2.53 %	10.74 %
Zabbix busy housekeeper processes, in %	[avg] 100 %	100 %	100 %	100 %
Zabbix busy alerter processes, in %	[avg] 0 %	0 %	1.43 %	76.09 %
Zabbix busy configuration syncer processes, in %	[avg] 15.06 %	0 %	12.55 %	48.82 %
Zabbix busy db watchdog processes, in %	[avg] 0.05 %	0 %	0.009626 %	0.12 %
Zabbix busy history syncer processes, in %	[avg] 33.49 %	13.42 %	29.14 %	100 %
Zabbix busy self-monitoring processes, in %	[avg] 0 %	0 %	0.002591 %	1.04 %

Partitioning

Секционирование или
партиционирование

history history_uint history_str history_text history_log	1 день	trends trends_uint	1 месяц
-----------------------------------------------------------------------	-----------	-----------------------	------------





Удаление данных – быстрое удаление (один файл/подтаблица, а не выборка строк на удаление)

Получение – заметно ускоряет выборку данных (использует одну или более секцию, а не всю таблицу) при правильном выборе индекса

Вставка данных – также ускоряется

Стандартный подход настройки партиционирования

PostgreSQL 9,10 - нужны процедуры и триггеры
+ скрипты для управления и наблюдения

```
# Create next month table for history, history_uint, history_str, history_log, history_text
sql="
CREATE TABLE IF NOT EXISTS public.history_y${next_year}m${next_month} PARTITION OF public.history
  FOR VALUES FROM ($start) TO ($end);
\ncREATE TABLE IF NOT EXISTS public.history_uint_y${next_year}m${next_month} PARTITION OF public.history_uint
  FOR VALUES FROM ($start) TO ($end);
\ncREATE TABLE IF NOT EXISTS public.history_str_y${next_year}m${next_month} PARTITION OF public.history_str
  FOR VALUES FROM ($start) TO ($end);
\ncREATE TABLE IF NOT EXISTS public.history_log_y${next_year}m${next_month} PARTITION OF public.history_log
  FOR VALUES FROM ($start) TO ($end);
\ncREATE TABLE IF NOT EXISTS public.history_text_y${next_year}m${next_month} PARTITION OF public.history_text
  FOR VALUES FROM ($start) TO ($end);
\ncREATE INDEX on public.history_y${next_year}m${next_month} USING btree (itemid, clock);
\ncREATE INDEX on public.history_uint_y${next_year}m${next_month} USING btree (itemid, clock);
\ncREATE INDEX on public.history_str_y${next_year}m${next_month} USING btree (itemid, clock);
\ncREATE INDEX on public.history_log_y${next_year}m${next_month} USING btree (itemid, clock);
\ncREATE INDEX on public.history_text_y${next_year}m${next_month} USING btree (itemid, clock);
"
```



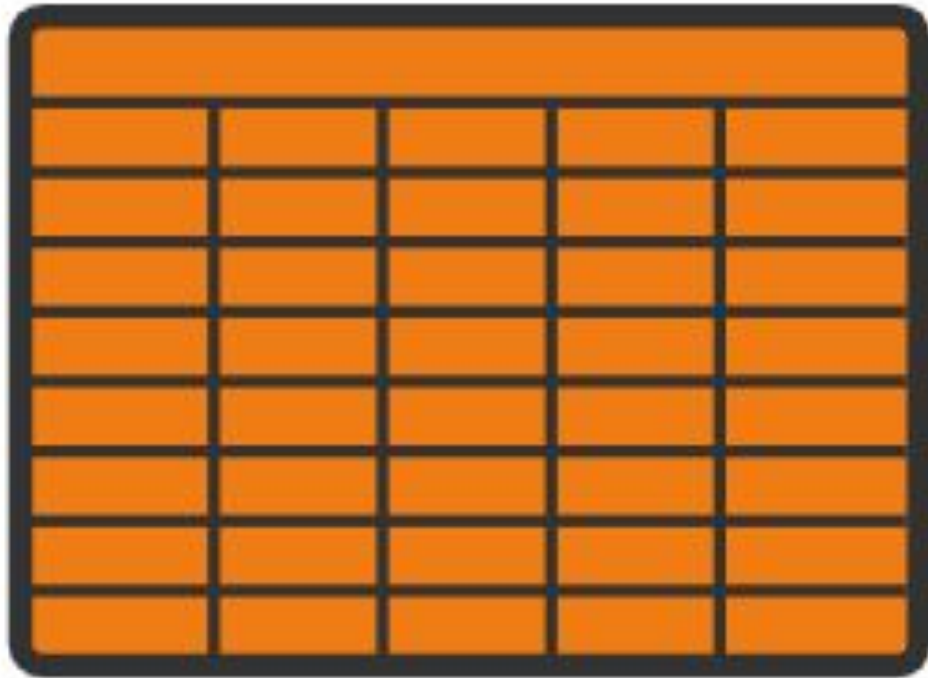


TimescaleDB

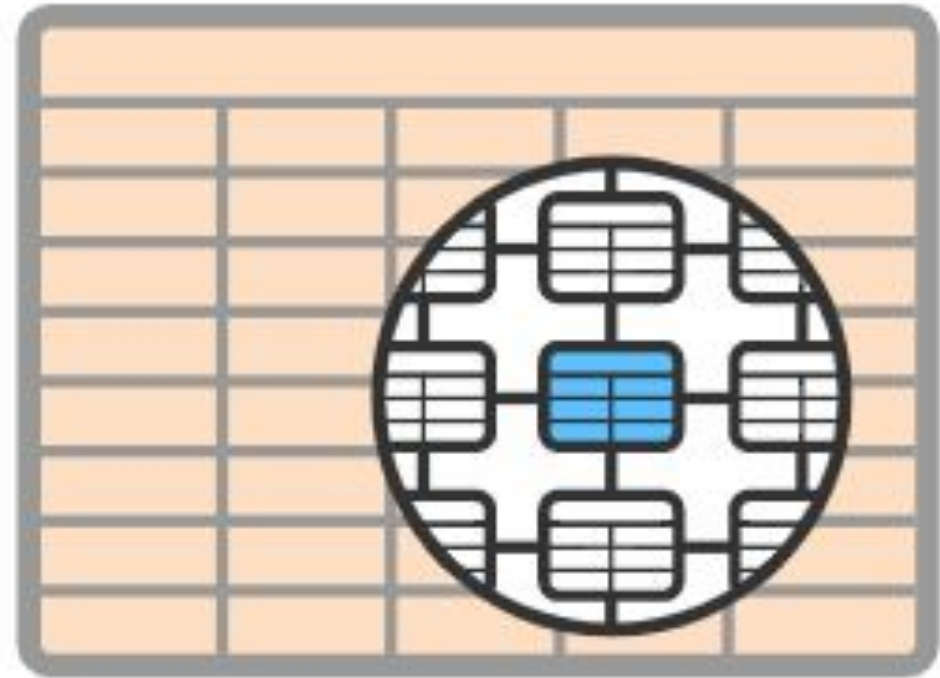
ZABBIX 2019
Conference
RUSSIA

TimescaleDB = PostgreSQL + Extension



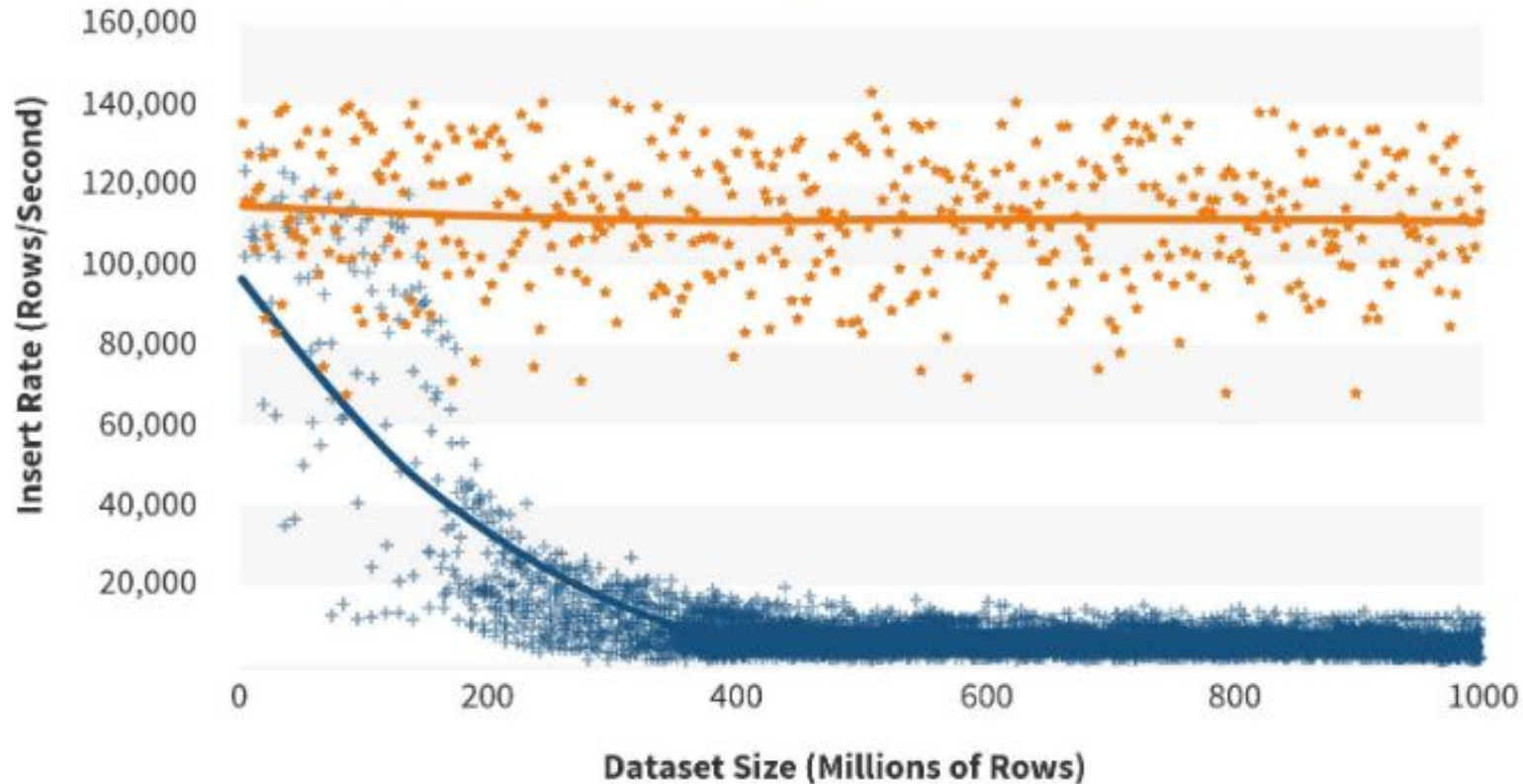


Hypertable



Chunk

Ingest Rate: PostgreSQL vs. Timescale



+ — PostgreSQL

Insert batch size: 10,000 Rows

Final avg. throughput: 5k (PG) vs. 111k (TS)

★ — TimescaleDB

Cache: 16 GB Memory

Time to load 1B rows: 37.9h (PG) vs. 2.6h (TS)

<https://docs.timescale.com/v1.3/getting-started>

Для базы данных с Zabbix выполняем команду

```
echo "CREATE EXTENSION IF NOT EXISTS timescaledb CASCADE;" | sudo -u postgres psql zabbix
```

Просто выполнить эти команды

```
SELECT create_hypertable('history', 'clock', chunk_time_interval => 86400, migrate_data => true);
SELECT create_hypertable('history_uint', 'clock', chunk_time_interval => 86400, migrate_data => true);
SELECT create_hypertable('history_log', 'clock', chunk_time_interval => 86400, migrate_data => true);
SELECT create_hypertable('history_text', 'clock', chunk_time_interval => 86400, migrate_data => true);
SELECT create_hypertable('history_str', 'clock', chunk_time_interval => 86400, migrate_data => true);
SELECT create_hypertable('trends', 'clock', chunk_time_interval => 86400, migrate_data => true);
SELECT create_hypertable('trends_uint', 'clock', chunk_time_interval => 86400, migrate_data => true);
UPDATE config SET db_extension='timescaledb',hk_history_global=1,hk_trends_global=1;
```



История

Активировать внутреннюю очистку истории

Переопределить период хранения истории элементов данных

* Период хранения данных

Динамика изменений

Активировать внутреннюю очистку истории

Переопределить период хранения динамики изменения элементов данных

* Период хранения данных


```
125:20190807:095228.409 zbx_setproctitle() title:'housekeeper [removing old history and trends]'  
125:20190807:095228.409 In housekeeping_history_and_trends() now:1565171548  
125:20190807:095228.410 In hk_history_delete_queue_prepare_all()  
125:20190807:095228.410 End of hk_history_delete_queue_prepare_all()  
125:20190807:095228.410 In hk_drop_partition_for_rule() now:1565171548  
125:20190807:095228.410 query [txnlev:0] [SELECT drop_chunks(1557395548,'history')]  
125:20190807:095228.411 End of hk_drop_partition_for_rule()  
125:20190807:095228.411 In hk_drop_partition_for_rule() now:1565171548  
125:20190807:095228.412 query [txnlev:0] [SELECT drop_chunks(1557395548,'history_str')]  
125:20190807:095228.412 End of hk_drop_partition_for_rule()  
125:20190807:095228.412 In hk_drop_partition_for_rule() now:1565171548  
125:20190807:095228.412 query [txnlev:0] [SELECT drop_chunks(1557395548,'history_log')]  
125:20190807:095228.412 End of hk_drop_partition_for_rule()  
125:20190807:095228.412 In hk_drop_partition_for_rule() now:1565171548  
125:20190807:095228.413 query [txnlev:0] [SELECT drop_chunks(1557395548,'history_uint')]  
125:20190807:095228.413 End of hk_drop_partition_for_rule()  
125:20190807:095228.413 In hk_drop_partition_for_rule() now:1565171548  
125:20190807:095228.413 query [txnlev:0] [SELECT drop_chunks(1557395548,'history_text')]  
125:20190807:095228.413 End of hk_drop_partition_for_rule()  
125:20190807:095228.413 In hk_drop_partition_for_rule() now:1565171548  
125:20190807:095228.413 query [txnlev:0] [SELECT drop_chunks(1533635548,'trends')]  
125:20190807:095228.414 End of hk_drop_partition_for_rule()  
125:20190807:095228.414 In hk_drop_partition_for_rule() now:1565171548  
125:20190807:095228.414 query [txnlev:0] [SELECT drop_chunks(1533635548,'trends_uint')]  
125:20190807:095228.414 End of hk_drop_partition_for_rule()
```

Zabbix + TimescaleDB

20 CPU Intel(R) Xeon(R) CPU E5-2630 2.20GHz

16G RAM

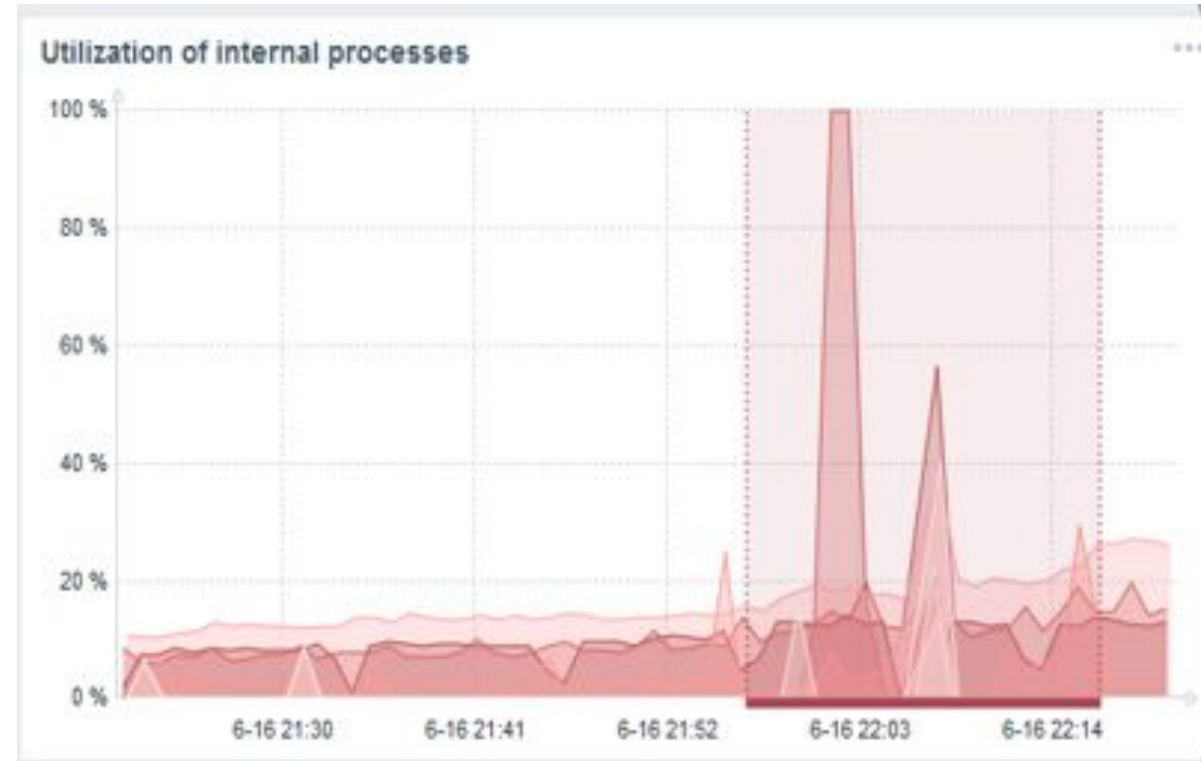
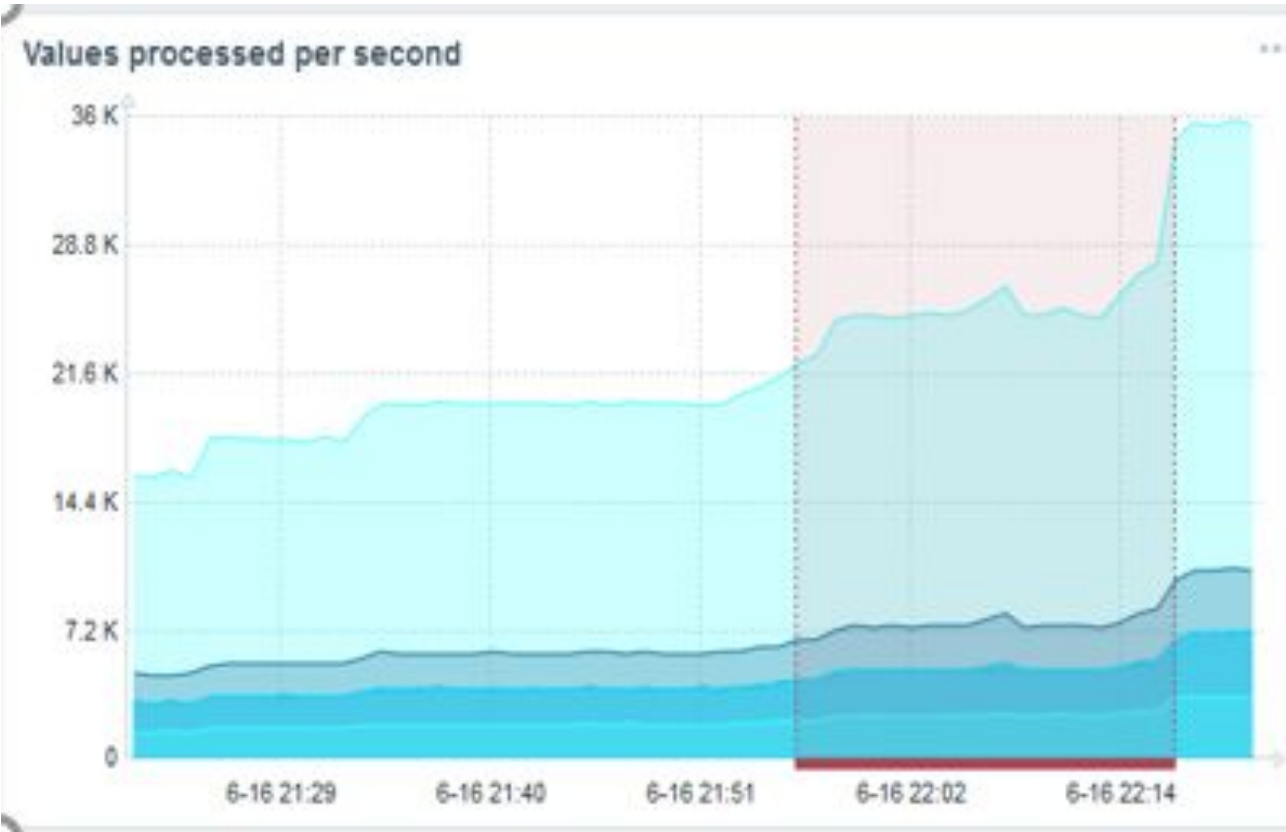
200G SSD disk

Vmware guest



36K NVPS PostgreSQL

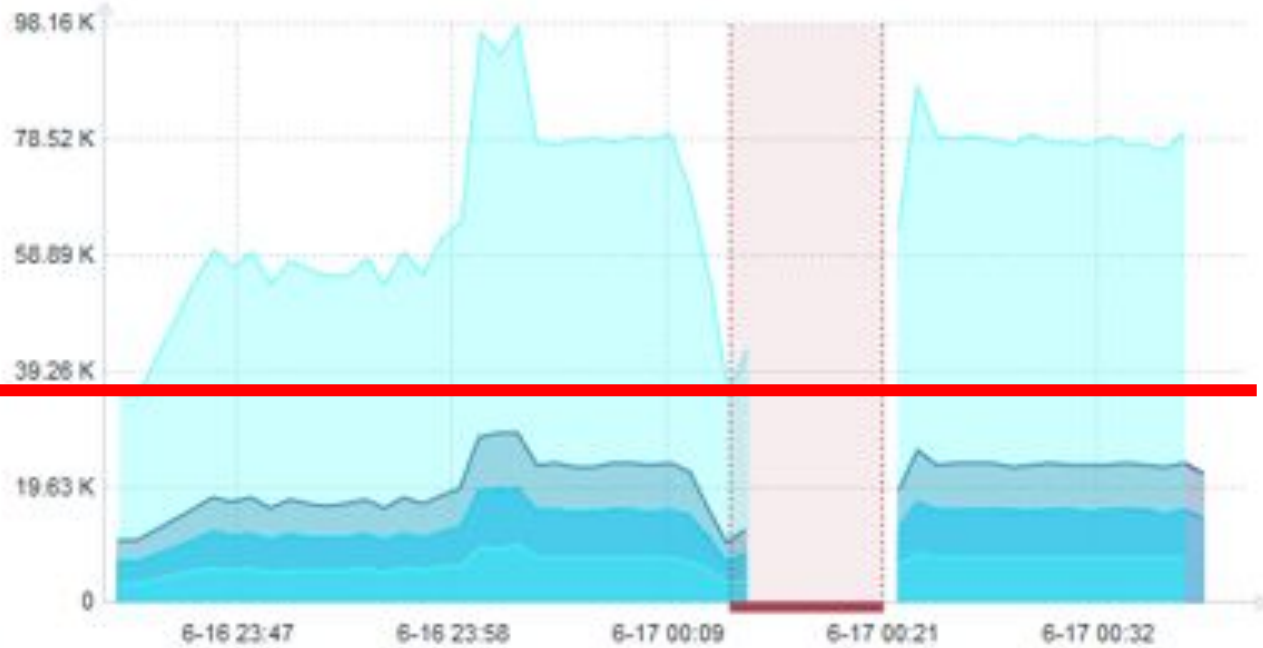
250K элементов данных, 175K триггеров



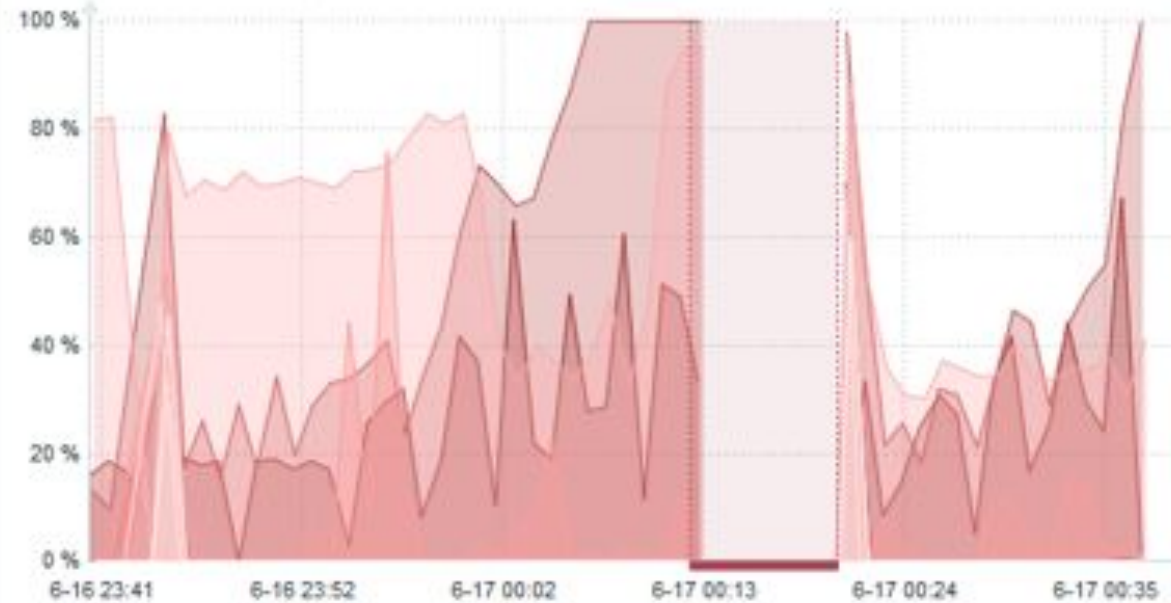
80K NVPS PostgreSQL

400K элементов данных, 280K триггеров

Values processed per second



Utilization of internal processes



80K NVPS PostgreSQL

Недостаточная производительность диска

```
02:45:19 PM      DEV      tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz   await   svctm   %util
02:45:20 PM      sda       14.00    128.00    294.00    30.14     0.12     8.57    2.86    4.00
02:45:20 PM      sdb    4448.00    5696.00  659990.00   149.66    325.00    72.96    0.23  100.40

02:45:20 PM      proc/s   cswch/s
02:45:21 PM      12.00  76047.00

02:45:20 PM      DEV      tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz   await   svctm   %util
02:45:21 PM      sda       8.00     34.00     56.00    11.25     0.02     2.50    2.50    2.00
02:45:21 PM      sdb    3587.00    7840.00  575677.00   162.68    321.66    75.86    0.28  100.00

02:45:21 PM      proc/s   cswch/s
02:45:22 PM      12.00  48602.00

02:45:21 PM      DEV      tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz   await   svctm   %util
02:45:22 PM      sda       1.00    136.00     0.00   136.00     0.00     4.00    4.00    0.40
02:45:22 PM      sdb    8175.00    6544.00  649720.00    80.28    189.02    33.03    0.12  100.00

02:45:22 PM      proc/s   cswch/s
02:45:23 PM      9.00  43769.00

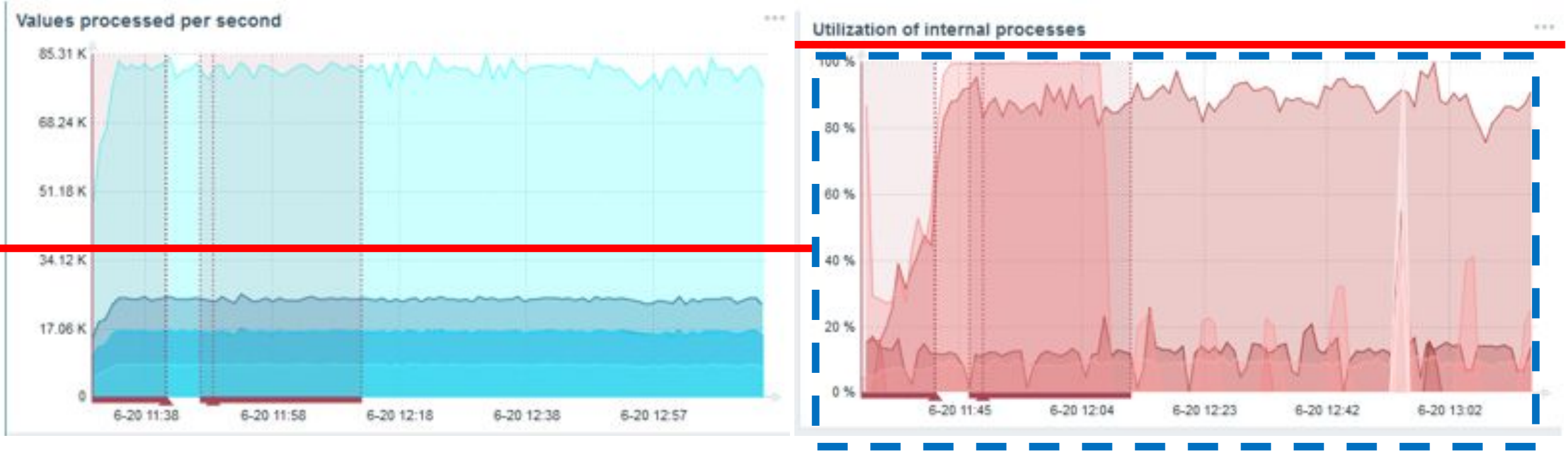
02:45:22 PM      DEV      tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz   await   svctm   %util
02:45:23 PM      sda       0.00     0.00     0.00     0.00     0.00     0.00    0.00    0.00
02:45:23 PM      sdb    8380.00    6496.00  598151.00    72.15    141.41    17.05    0.12  99.60
^C

Average:      proc/s   cswch/s
Average:      11.50  52161.50

Average:      DEV      tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz   await   svctm   %util
Average:      sda       5.75     74.50     87.50    28.17     0.04     6.26    2.78    1.60
Average:      sdb    6147.50    6644.00  620884.50   102.08    244.27    41.05    0.16  100.00
root@perftest-debian-9:~#
```

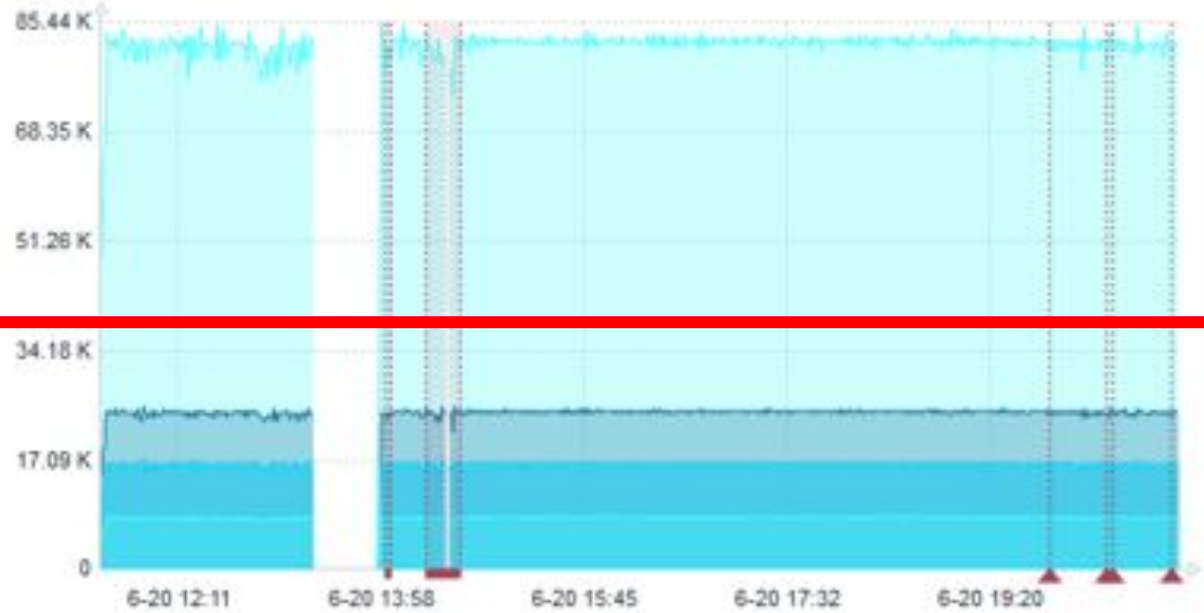
80K NVPS PostgreSQL

Другой сервер 48CPU, 128G RAM, 1.5T

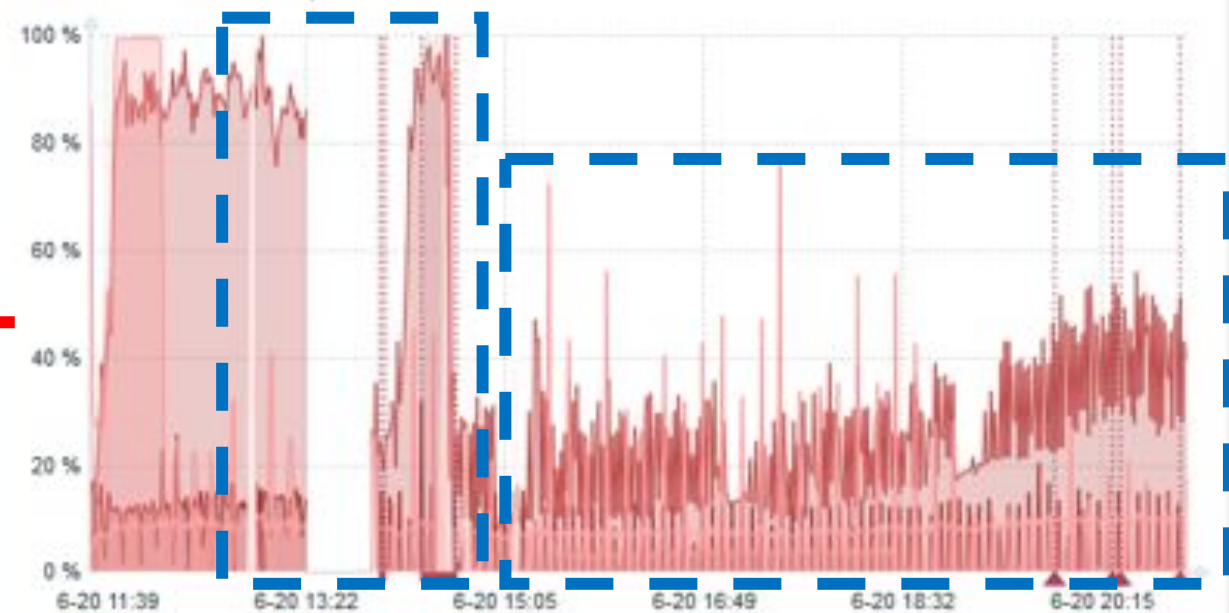


80K NVPS PostgreSQL vs TimescaleDB

Values processed per second



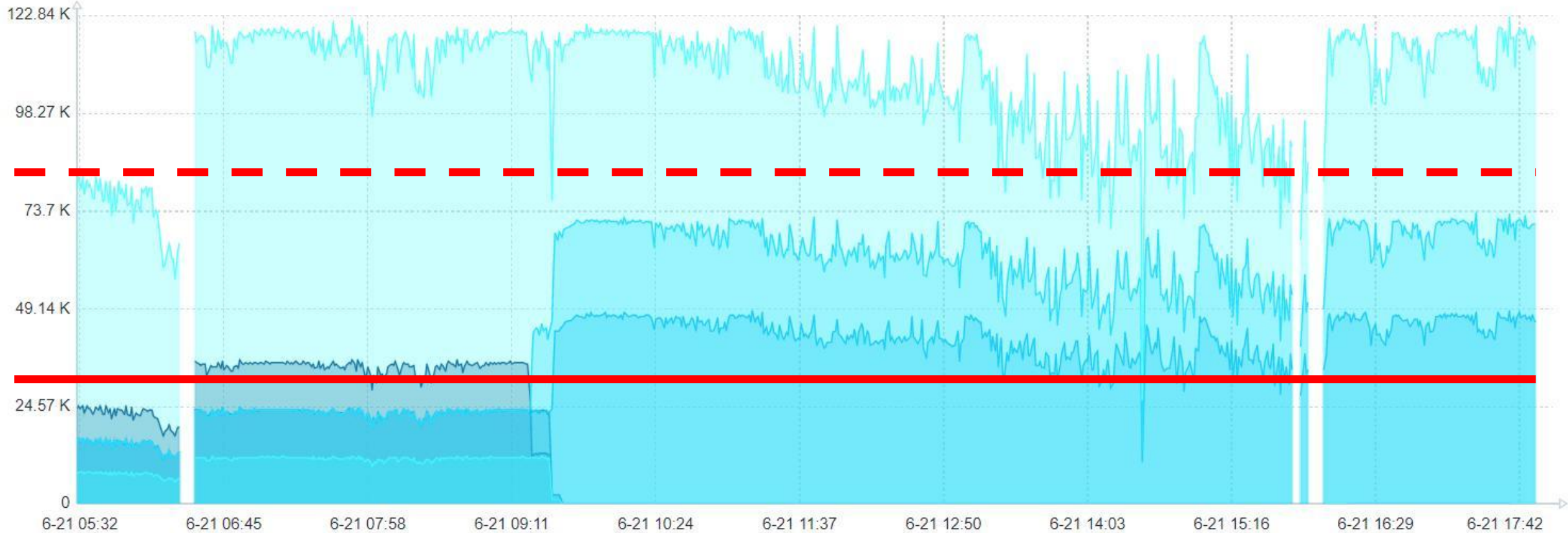
Utilization of internal processes



120K NVPS TimescaleDB

500K элементов данных, 350K триггеров

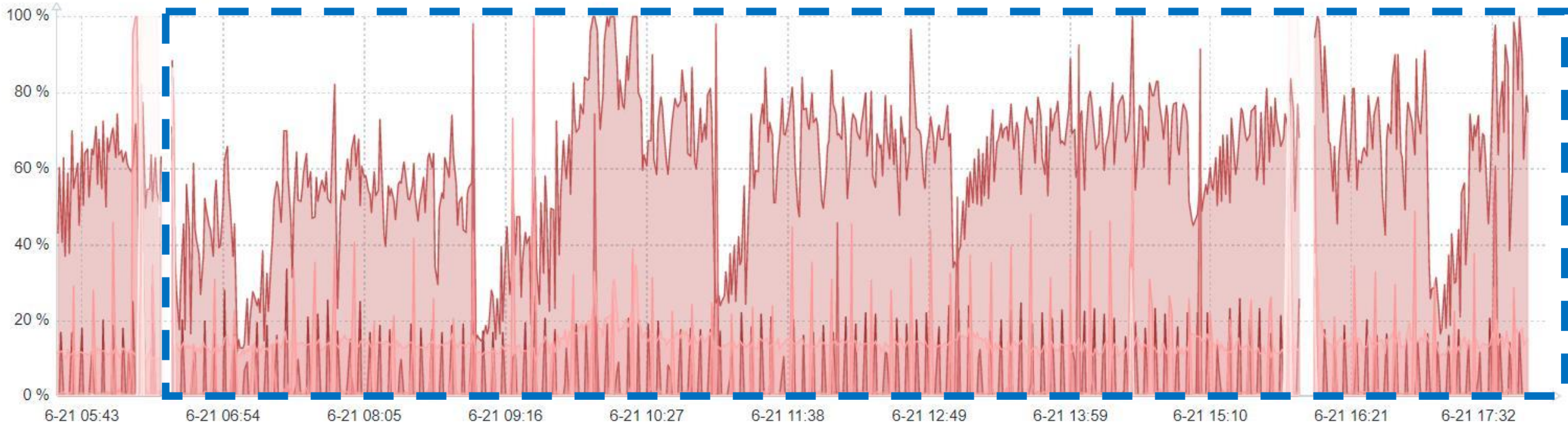
Values processed per second



120K NVPS TimescaleDB

500K элементов данных, 350K триггеров

Utilization of internal processes



120K+ NVPS TimescaleDB

Добавил 3 прокси и распределил часть проверок

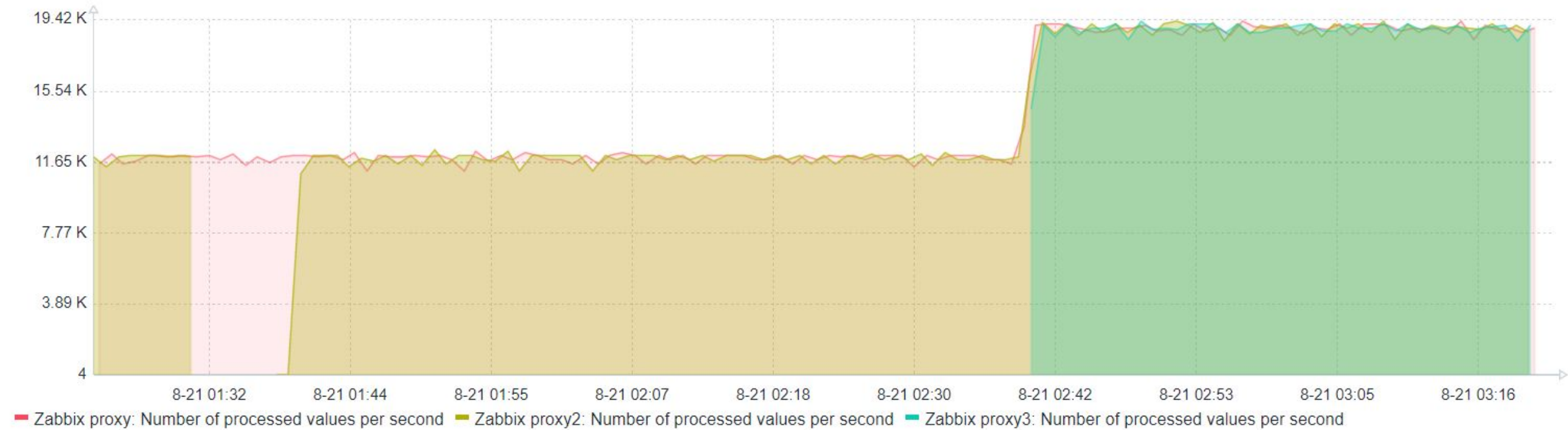
System information

Parameter	Value	Details
Zabbix server is running	Yes	192.168.3.63:10051
Number of hosts (enabled/disabled/templates)	152	66 / 0 / 86
Number of items (enabled/disabled/not supported)	721122	721093 / 0 / 29
Number of triggers (enabled/disabled [problem/ok])	432498	432498 / 0 [152468 / 280030]
Number of users (online)	2	1
Required server performance, new values per second	144051.23	

120K+ NVPS TimescaleDB

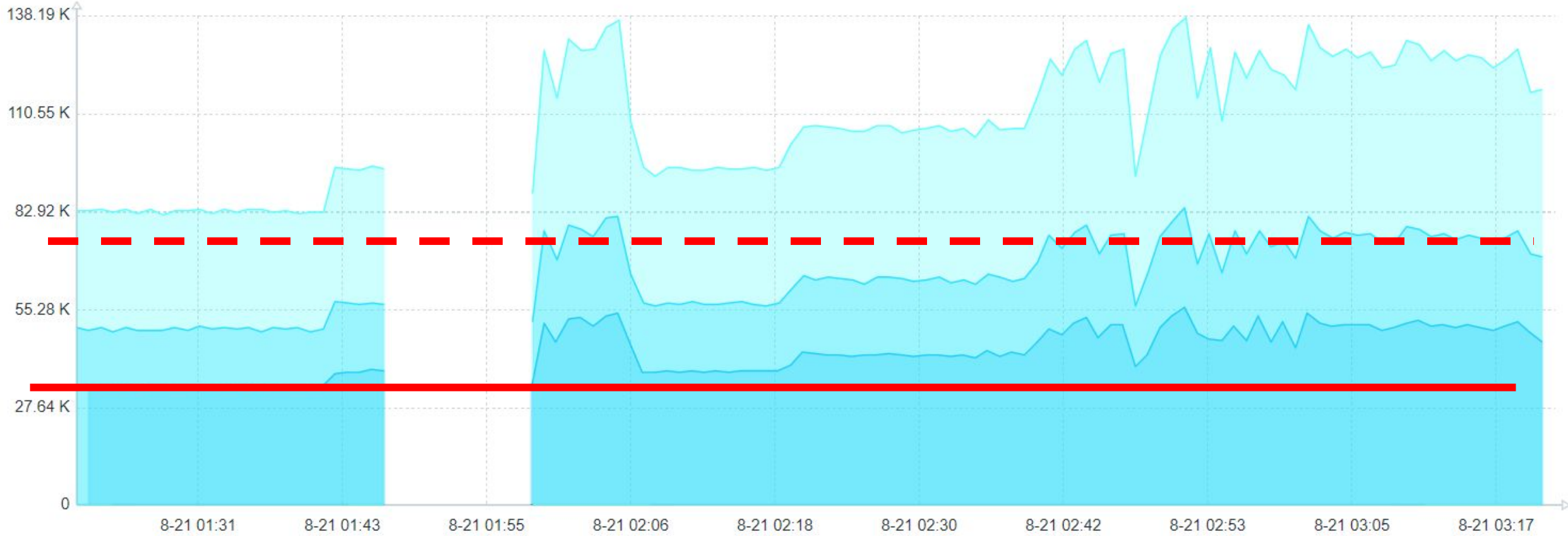
10-20K через прокси с PostgreSQL

Proxies values per second



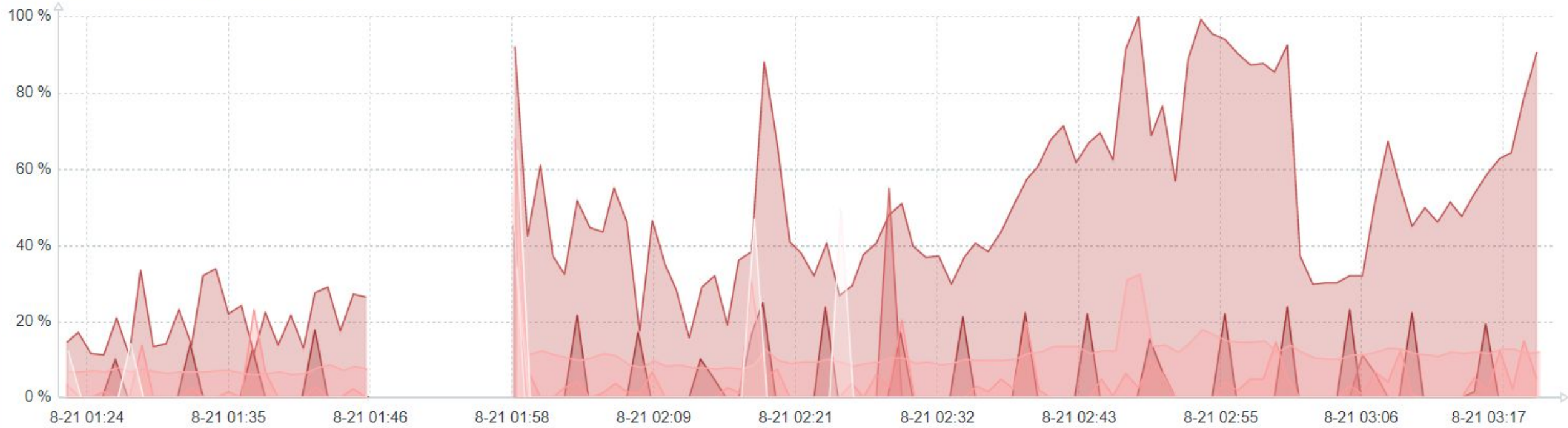
130K+ NVPS TimescaleDB

Values processed per second



130K+ NVPS TimescaleDB

Utilization of internal processes



130K+ NVPS TimescaleDB

```
z42=# SELECT * FROM hypertable_relation_size_pretty("");
```

Table_name	table_size		index_size		toast_size		total_size
History	225 GB		342 GB				567 GB
history_uint	183 GB		288 GB				471 GB
history_log	22 GB		21 GB		24 kB		43 GB
history_str	20 GB		33 GB				53 GB
trends	370 MB		520 MB				890 MB
trends_uint	440 MB		520 MB				960 MB

SUM > 1.2 T

130K+ NVPS TimescaleDB

```
z42=# SELECT * FROM hypertable_approximate_row_count();
 schema_name | table_name | row_estimate
-----+-----+-----
 public      | history    | 1803701120
 public      | history_log | 267764352
 public      | history_str | 387992448
 public      | history_text | 133875024
 public      | history_uint | 2095128064
 public      | trends     | 3234145
 public      | trends_uint | 4425204
(7 rows)
```

130K+ NVPS TimescaleDB

13916:20190821:021722.515 slow query: 25.018634 sec, "select min(clock) from proxy_history"

13916:20190821:021905.737 slow query: 103.218789 sec, "delete from proxy_history where id<64980232 and (clock<1566339417 or (id<=64676112 and clock<1566343017))"

13916:20190821:021905.773 housekeeper [deleted **43028145** records in 128.279393 sec, idle for 1 hour(s)]

Выводы

- 120K max (дальше нужно изменять код)
- если не производительные диски
- если у вас стандартный PostgreSQL и вы на пороге партиционирования
- подходит как стартовая точка для любой инсталляции
- Используйте прокси. Они быстрые и эффективные!

Сообщество и События

Каналы Telegram:

[@ZabbixPro](https://t.me/ZabbixPro)

@ZabbixJunior

Eng: @ZabbixTech

Форум

IRC #zabbix

ZABBIX SUMMIT 2019

Riga, Latvia

11 - 12 Oct, 2019

ZABBIX '19
SUMMIT

СПАСИБО! ВОПРОСЫ?



Гуцин Андрей Николаевич
ZABBIX Инженер технической поддержки

ZABBIX 2019
Conference
RUSSIA



twitter.com/zabbix_ru



habr.com/ru/company/zabbix/