ZABBIX 5.0

UPGRADING ZABBIX TO A NEWER VERSION AND TIMESCALEDB

Thomas Oftring, Rhein-Main Solutions GmbH
Zabbix Certified Trainer

RHEIN-MAIN SOLUTIONS

01

WHY UPGRADE?

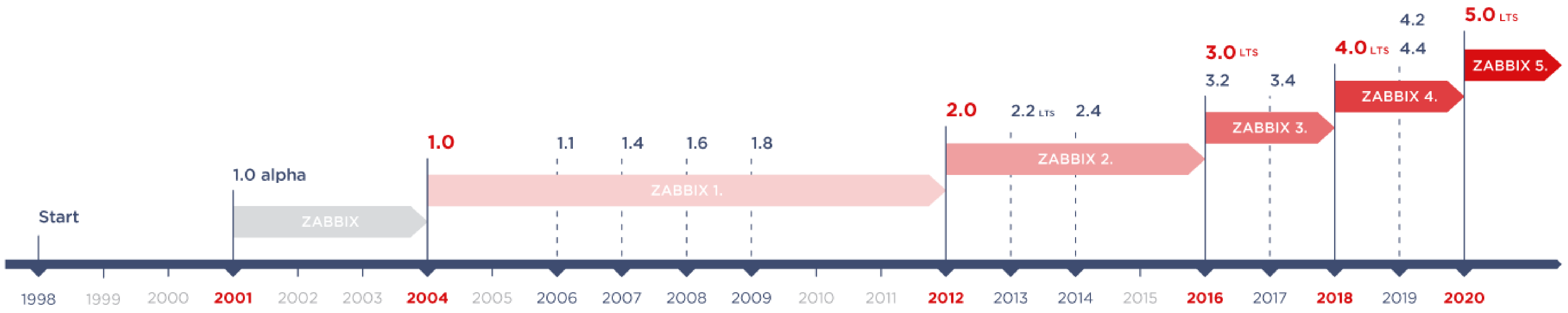
IN SHORT

SLAPS SERVER
ZABBIX
THIS BAD BOY CAN COLLECT SO MUCH DATA WHILE BEING SECURE AND EFFICIENT

COMPONENTS TO UPGRADE

- Improved stability
- Improved performance
- Improved security
- New feature support

AND OF COURSE LIFE CYCLE



Zabbix LTS (Long Term Support) releases are supported for Zabbix customers during five (5) years
Standard: until next release

RHEIN-MAIN SOLUTIONS

WHICH VERSION TO CHOOSE?

5.0 S

02

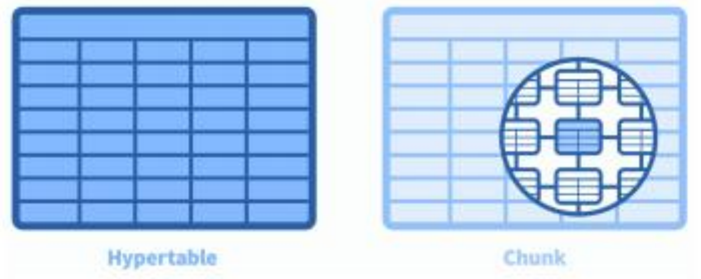
WHAT IS AND WHY TIMESCALEDB

BUT WHY?

WHAT IS TIMESCALEDB?

TimescaleDB is a PostgreSQL extension, which adds time-series based performance and data management optimizations to a regular PostgreSQL database. Based on architectural solutions like:

- Hypertables
  - Abstraction layer and primary point of interaction with your data used for creating tables and indexes, altering tables, inserting data, selecting data
- Chunks
  - Hypertables are automatically split into chunks; each chunk corresponds to a specific time interval and a region of the partition key's space

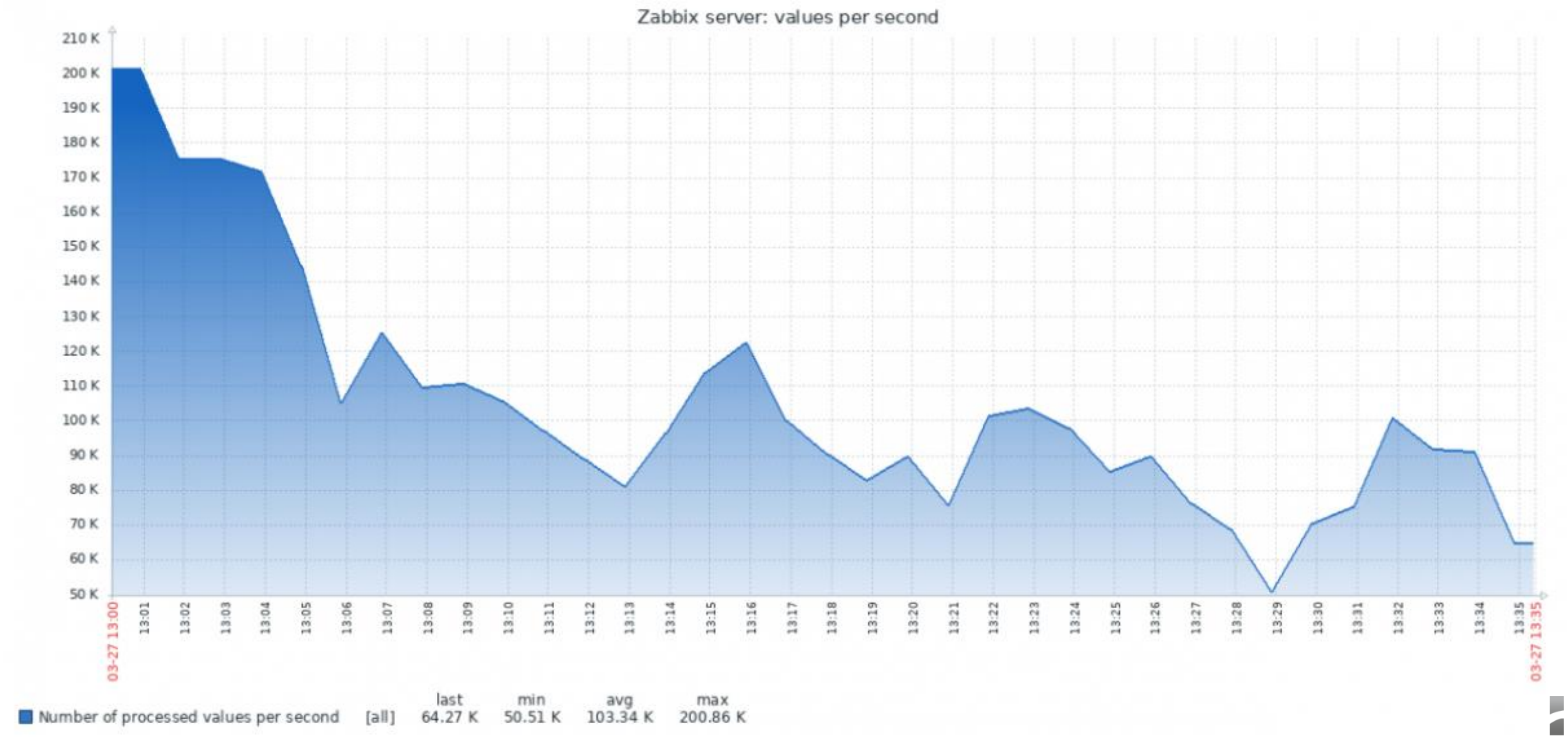


WHY TIMESCALEDB?

- It’s an extension
  - It doesn’t require extra hardware, virtual machines or any other infrastructure changes, you can continue to use your PostgreSQL tools of choice and SQL operations and queries
- Intact code
  - It lets you keep virtually all database-related code in Zabbix intact.
- Performance
  - Considerable performance improvements for Zabbix history syncer and housekeeper.

WHY TIMESCALEDB - PERFORMANCE

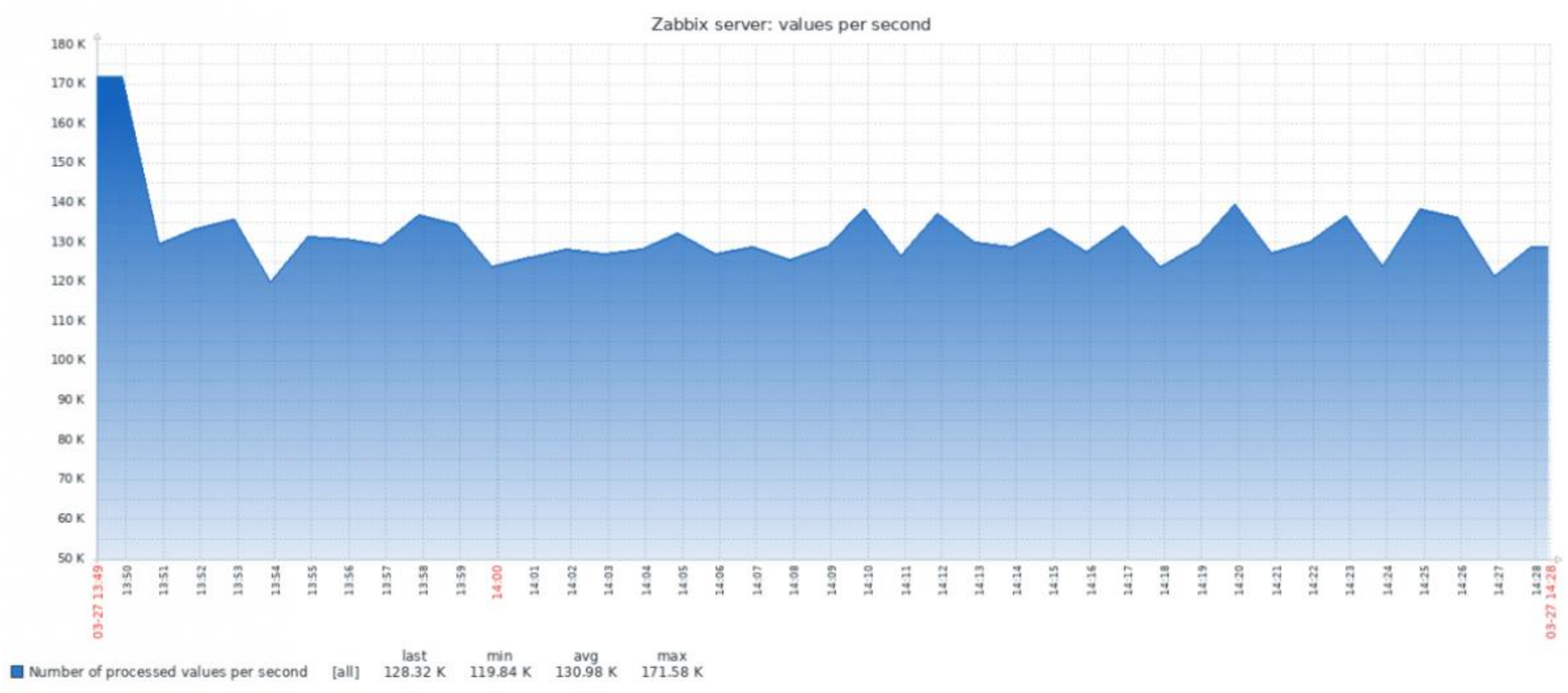
PostgreSQL without TSDB extension



		last	min	avg	max
🟦 Number of processed values per second	[all]	64.27 K	50.51 K	103.34 K	200.86 K

WHY TIMESCALEDB - PERFORMANCE

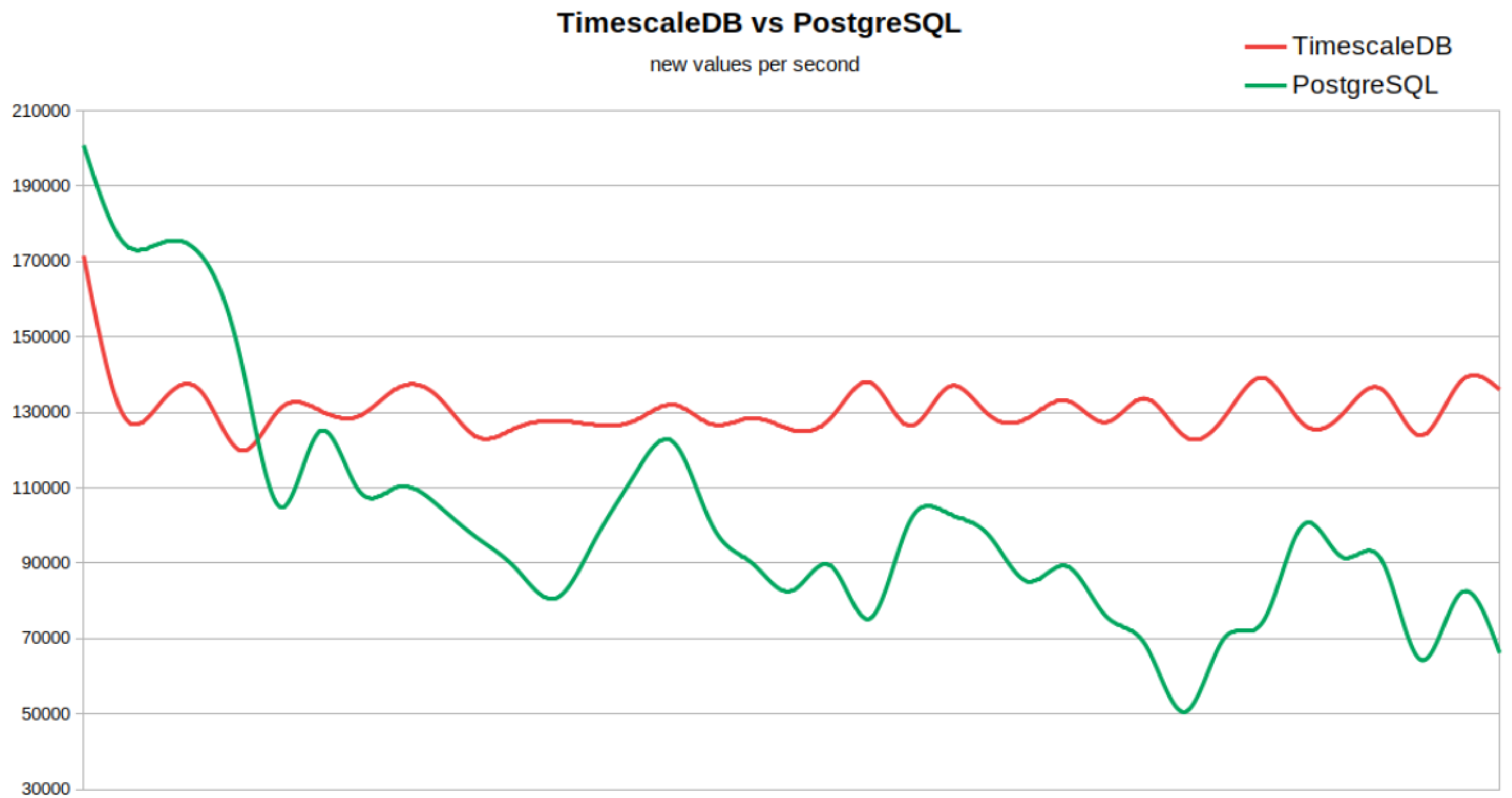
PostgreSQL with TSDB extension



		last	min	avg	max
🟦 Number of processed values per second	[all]	128.32 K	119.84 K	130.98 K	171.58 K

WHY TIMESCALEDB - PERFORMANCE

Comparison

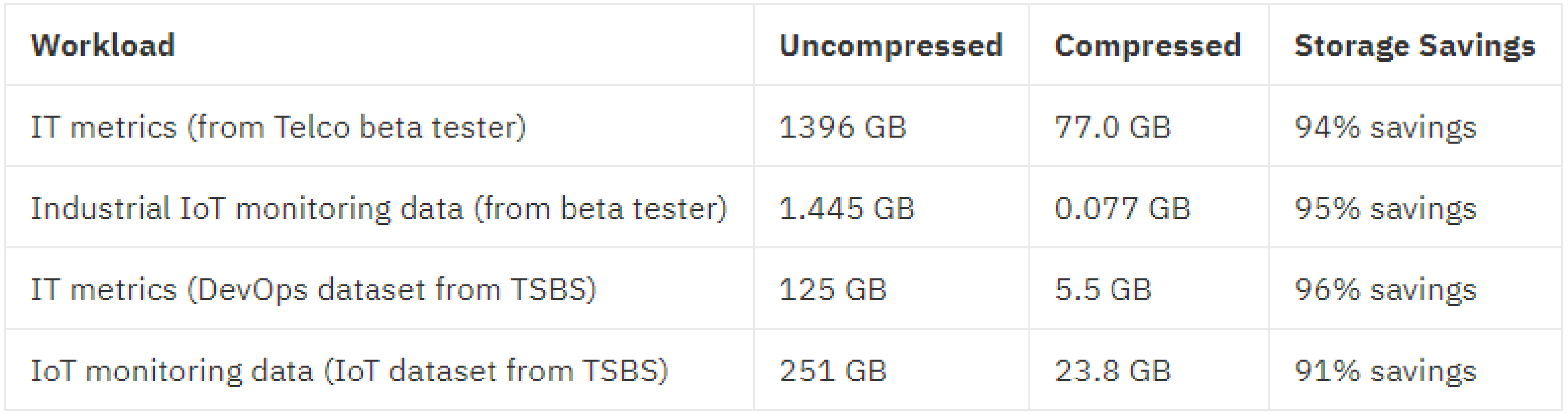


TimescaleDB avg 130K NVPS; PostgreSQL avg 90K NVPS

RHEIN-MAIN SOLUTIONS

WHY TIMESCALEDB - COMPRESSION

Up to 90% storage savings



Workload	Uncompressed	Compressed	Storage Savings
IT metrics (from Telco beta tester)	1396 GB	77.0 GB	94% savings
Industrial IoT monitoring data (from beta tester)	1.445 GB	0.077 GB	95% savings
IT metrics (DevOps dataset from TSBS)	125 GB	5.5 GB	96% savings
IoT monitoring data (IoT dataset from TSBS)	251 GB	23.8 GB	91% savings

HOW COMPRESSION WORKS

Before compression

time	device_id	cpu	disk_io	energy_consumption
12:00:02	1	88.2	20	0.8
12:00:01	2	300.5	30	0.9
12:00:01	1	88.6	25	0.85
12:00:01	2	299.1	40	0.95

After compression

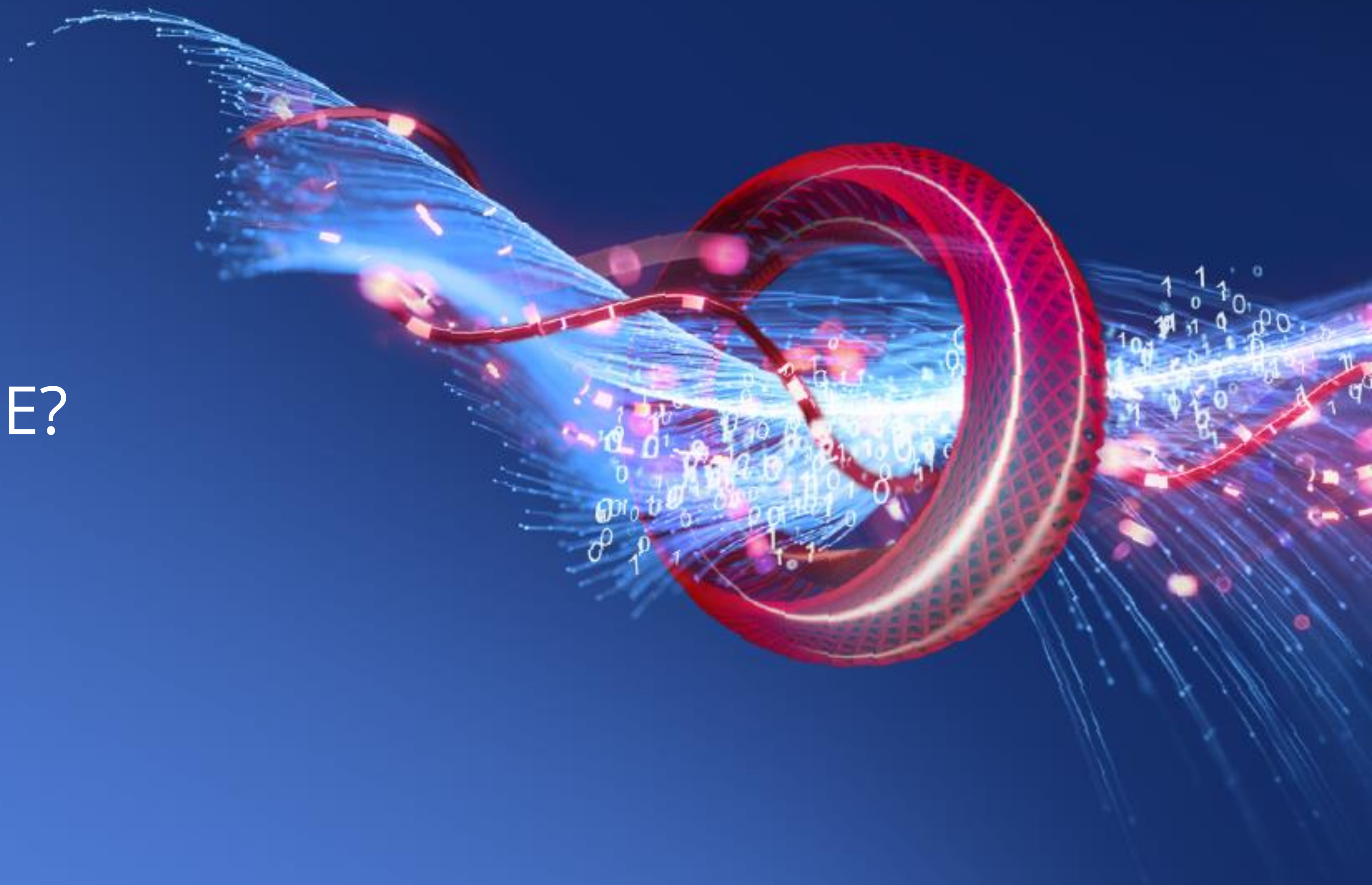
time	device_id	cpu	disk_io	energy_consumption
[12:00:02, 12:00:02, 12:00:01, 12:00:1]	[1, 2, 1, 2]	[88.2, 300.5, 88.6, 299.1]	[20, 30, 25, 40]	[0.8, 0.9, 0.85, 0.95]

WHY TIMESCALEDB - IT IS STILL SQL

- Supports all SQL operations and queries
- Compatible with existing PostgreSQL ecosystem and tooling
- Transparent time/space partitioning for both scaling up (single node)
- It is still quite accessible for newcomers

03

WHAT TO UPGRADE?



IN SHORT

(Almost)

COMPONENTS TO UPGRADE

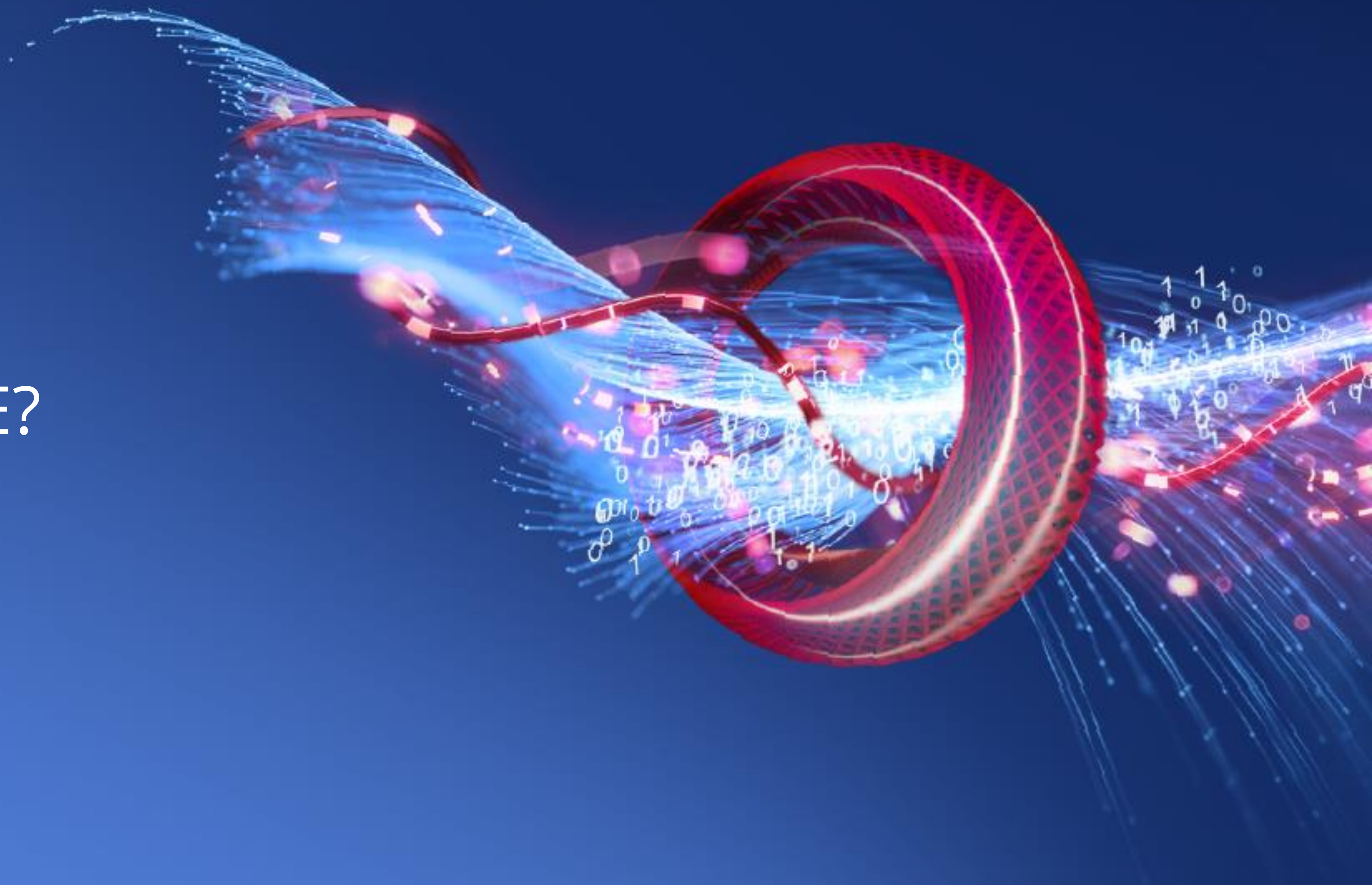
- Zabbix server
- Database (depending on version)
- Web frontend
- Proxy
- Java Gateway

UPGRADE NOT REQUIRED

- Zabbix agent
- Database (depending on version)

04

HOW TO UPGRADE?



HOW TO UPGRADE

SETUP

- CentOS 7
- PostgreSQL 9.6
- Zabbix server 4.0
- Your average hardware

UPGRADING POSTGRESQL

Using an older PostgreSQL version, but want to use the TimescaleDB in full, including compression?

Make sure you upgrade your DB to at least PostgreSQL version 10.2 or higher. How to upgrade PostgreSQL?

1. Install the repo RPM for version you will be using:

```
yum install https://download.postgresql.org/pub/repos/yum/reporpms/EL-7-x86_64/pgdg-redhat-repo-latest.noarch.rpm
```

2. Install new PostgreSQL version

```
yum install postgresql12-server
```

3. Stop the current one (don’t forget the server)

```
systemctl stop postgresql-9.6
```

4. Initialize the new one

```
sudo su postgres
cd ~/
/usr/pgsql-12/bin/initdb -D /var/lib/pgsql/12/data/
```

5. Migrate the data

```
/usr/pgsql-12/bin/pg_upgrade --old-datadir /var/lib/pgsql/9.6/data/
--new-datadir /var/lib/pgsql/12/data/ --old-bindir /usr/pgsql-9.6/bin/
--new-bindir /usr/pgsql-12/bin/
```

6. Start new instance

```
systemctl start postgresql-12
```

BACK UP

- Read the release notes and take note of the important changes:

https://www.zabbix.com/documentation/5.0/manual/installation/upgrade/packages/rhel_centos

- Make a Zabbix database back up

```
pg_dump dbname > dbname.bak
```

- Or back up the configuration files

```
$ cp /etc/zabbix/zabbix_server.conf /<backup directory>/      # Zabbix server config file
$ cp /etc/zabbix/zabbix_agentd.conf /<backup directory>/      # Zabbix agent config file
$ cp /usr/share/zabbix/alertscripts/* /<backup directory>/    # Alert scripts
$ cp /usr/share/zabbix/externalscripts/* /<backup directory>/ # External scripts
$ cp -R /usr/share/zabbix/ /<backup directory>/               # Web frontend PHP files
$ cp /etc/httpd/conf/httpd.conf /<backup directory>/          # Apache config. files
$ cp /etc/httpd/conf.d/zabbix.conf /<backup directory>/       # Zabbix/PHP parameters
$ cp /etc/zabbix/web/zabbix.conf.php /<backup directory>/     # Zabbix frontend parameters.
```

START THE UPGRADE

1. Stop Zabbix server to make sure that no new data is inserted into database.

```
systemctl stop zabbix-server
```

2. Upgrade your current repository package

```
rpm -Uvh https://repo.zabbix.com/zabbix/5.0/rhel/7/x86_64/zabbix-release-5.0-1.el7.noarch.rpm
```

3. Upgrade Zabbix components

```
yum upgrade zabbix-server-pgsql zabbix-agent
```

UPGRADE FRONTEND

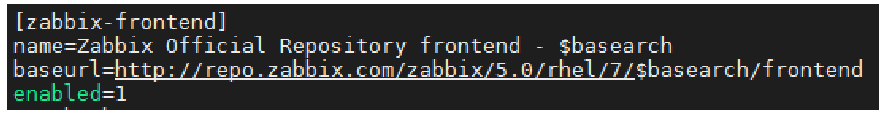
1. Remove old frontend

```
yum remove zabbix-web-*
```

2. Install SCL repository

```
yum install centos-release-scl
```

3. Edit /etc/yum.repos.d/zabbix.repo file



```
[zabbix-frontend]
name=Zabbix Official Repository frontend - $basearch
baseurl=http://repo.zabbix.com/zabbix/5.0/rhel/7/$basearch/frontend
enabled=1
```

4. Install new frontend packages

```
yum install zabbix-web-pgsql-scl
```

5. Update timezone in /etc/opt/rh/rh-php72/php-fpm.d/zabbix.conf file.

6. Start and enable php-fpm service.

```
systemctl start rh-php72-php-fpm
```

7. Restart Apache

```
systemctl restart httpd
```

INSTALL TIMESCALEDB

1. Add TimescaleDB repo:

https://docs.timescale.com/latest/getting-started/installation/rhel-centos/installation-yum

```
sudo tee /etc/yum.repos.d/timescale_timescaledb.repo <<EOL
[timescale_timescaledb]
name=timescale_timescaledb
baseurl=https://packagecloud.io/timescale/timescaledb/el/7/\$basearch
repo_gpgcheck=1
gpgcheck=0
enabled=1
gpgkey=https://packagecloud.io/timescale/timescaledb/gpgkey
sslverify=1
sslcacert=/etc/pki/tls/certs/ca-bundle.crt
metadata_expire=300
EOL
sudo yum update -y
```

2. Install appropriate package for PG version

```
yum install -y timescaledb-postgresql-12
```

3. Configure your PostgreSQL

```
timescaledb-tune --pg-config=/usr/pgsql-12/bin/pg_config
```

PREPARE YOUR DB FOR TIMESCALEDB

1. Start Zabbix server to update the DB schema
2. Stop Zabbix server
3. Enable TimescaleDB extension:

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

4. Then run the timescaledb.sql script located in database/postgresql

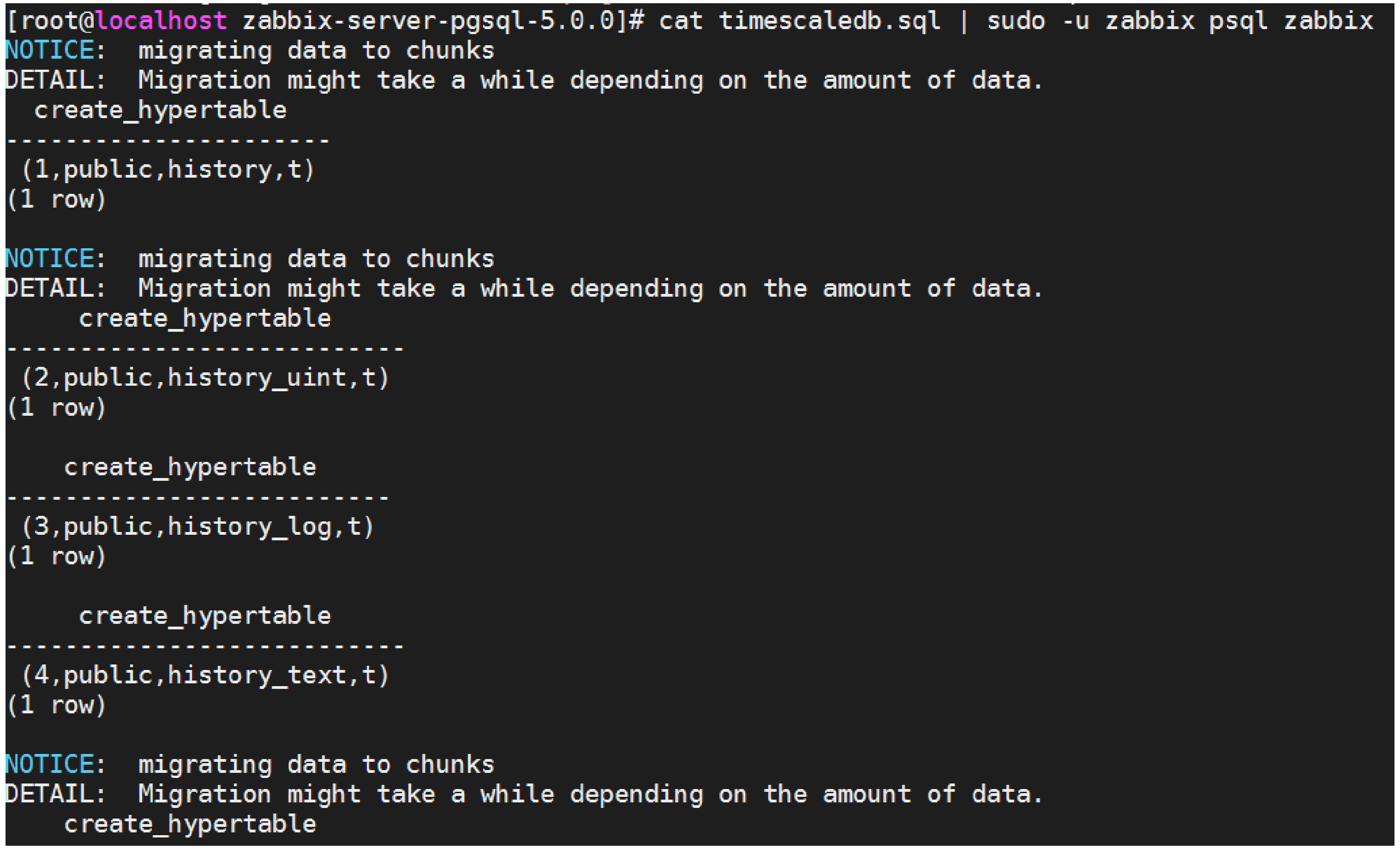
```
cd /usr/share/doc/zabbix-server-pgsql-5.0.0/
cat timescaledb.sql | sudo -u zabbix psql zabbix
```

5. Wait. The migration of existing history and trend data may take a lot of time. Zabbix server and frontend must be down for the period of migration.

6. When finished – start Zabbix server

PREPARE YOUR DB FOR TIMESCALEDB

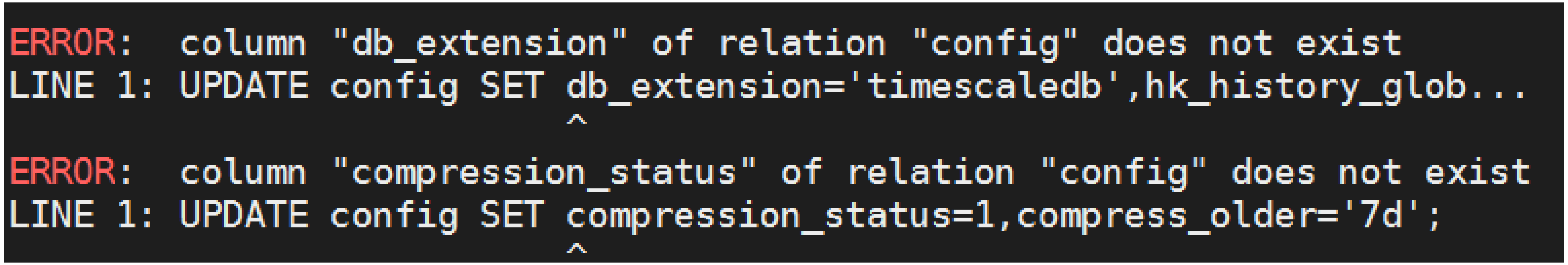
1. Wait. The migration of existing history and trend data may take a lot of time. Zabbix server and frontend must be down for the period of migration.



```
[root@localhost zabbix-server-pgsql-5.0.0]# cat timescaledb.sql | sudo -u zabbix psql zabbix
NOTICE:  migrating data to chunks
DETAIL:  Migration might take a while depending on the amount of data.
  create_hypertable
----------------------
 (1,public,history,t)
(1 row)

NOTICE:  migrating data to chunks
DETAIL:  Migration might take a while depending on the amount of data.
     create_hypertable
---------------------------
 (2,public,history_uint,t)
(1 row)

    create_hypertable
--------------------------
 (3,public,history_log,t)
(1 row)

     create_hypertable
---------------------------
 (4,public,history_text,t)
(1 row)

NOTICE:  migrating data to chunks
DETAIL:  Migration might take a while depending on the amount of data.
    create_hypertable
```

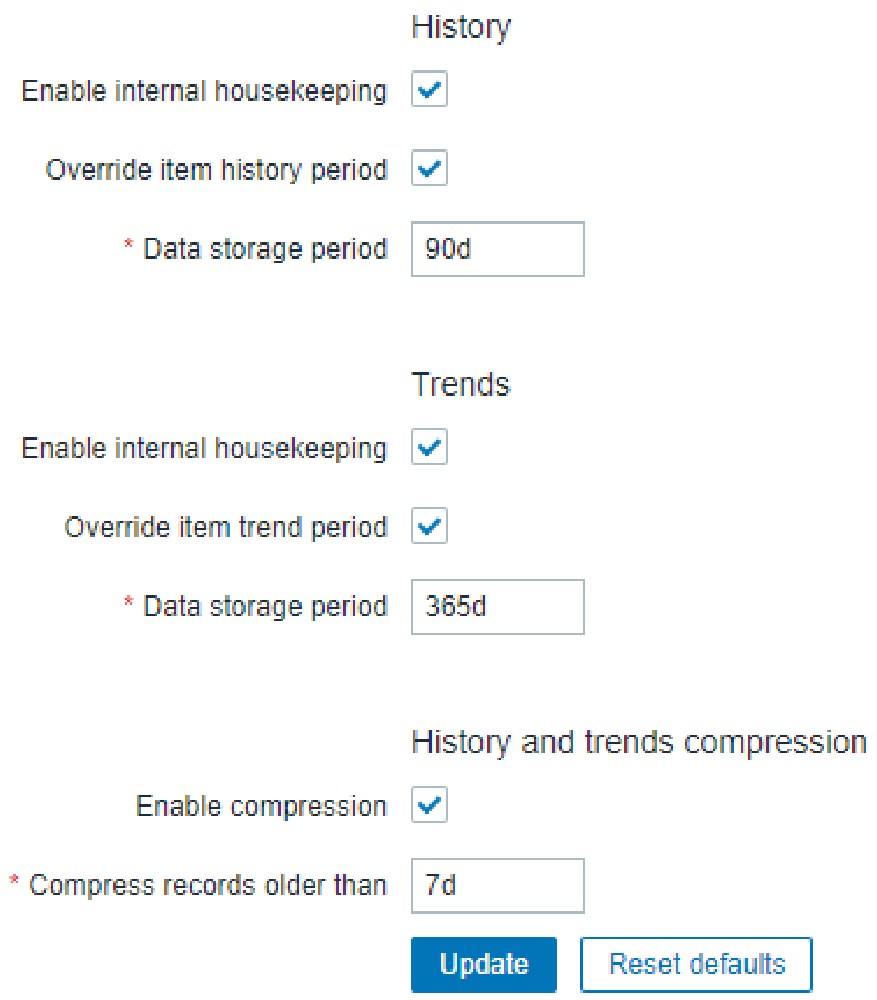
PREPARE YOUR DB FOR TIMESCALEDB

1. Make sure the output is not



```
ERROR:  column "db_extension" of relation "config" does not exist
LINE 1: UPDATE config SET db_extension='timescaledb',hk_history_glob...
                          ^
ERROR:  column "compression_status" of relation "config" does not exist
LINE 1: UPDATE config SET compression_status=1,compress_older='7d';
                          ^
```

CHECK HOUSEKEEPING SETTINGS



History

Enable internal housekeeping	✓
Override item history period	✓
* Data storage period	90d

Trends

Enable internal housekeeping	✓
Override item trend period	✓
* Data storage period	365d

History and trends compression

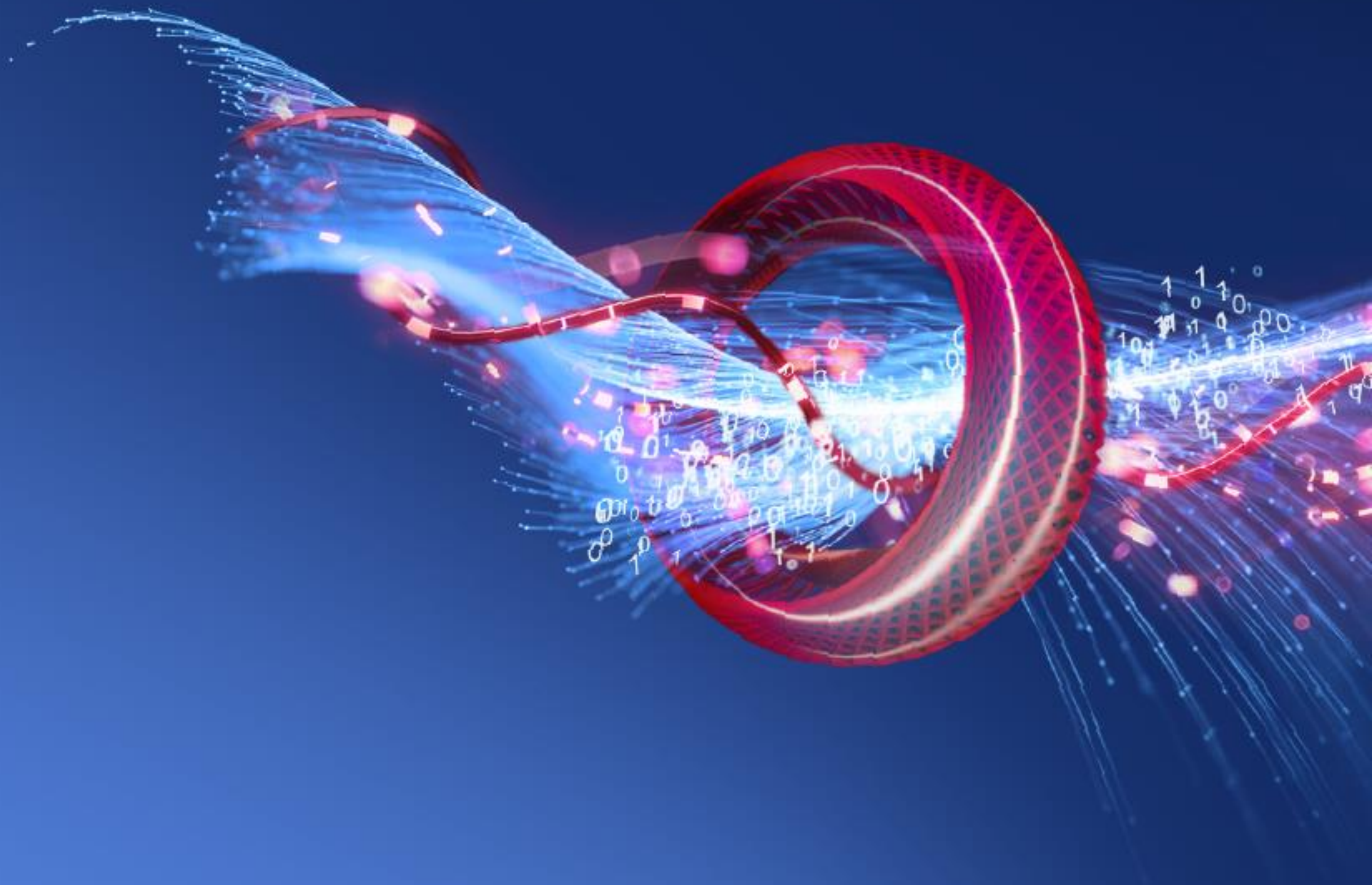
Enable compression	✓
* Compress records older than	7d

WHAT NEXT?

Enjoy monitoring!

05

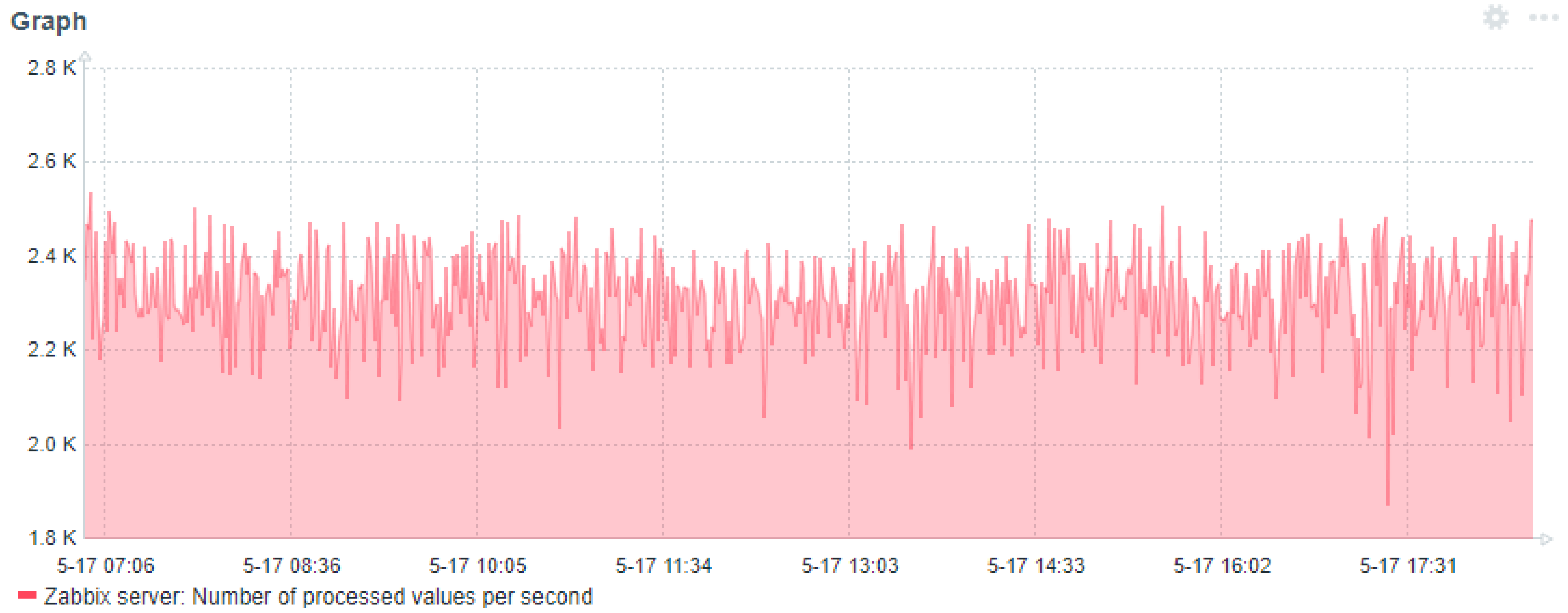
RESULTS



RHEIN-MAIN SOLUTIONS

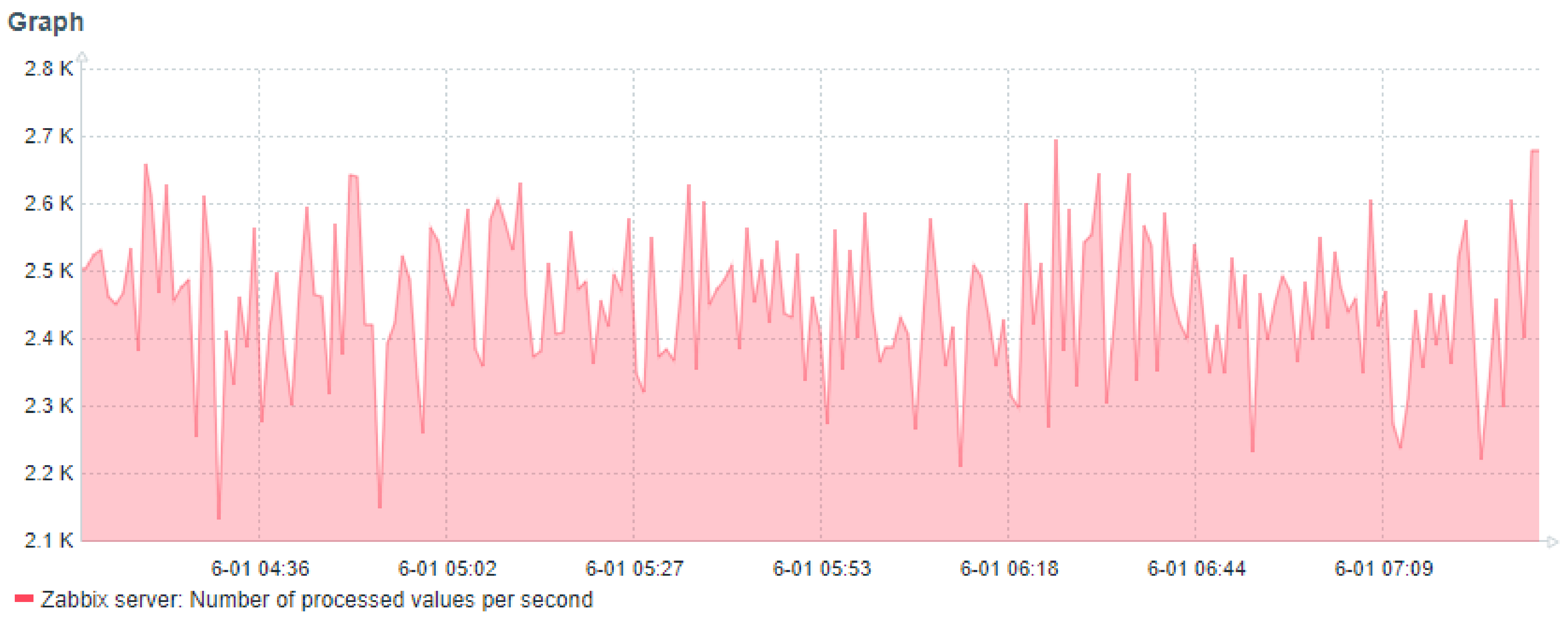
SHOW OFF RESULTS

PERFORMANCE BEFORE



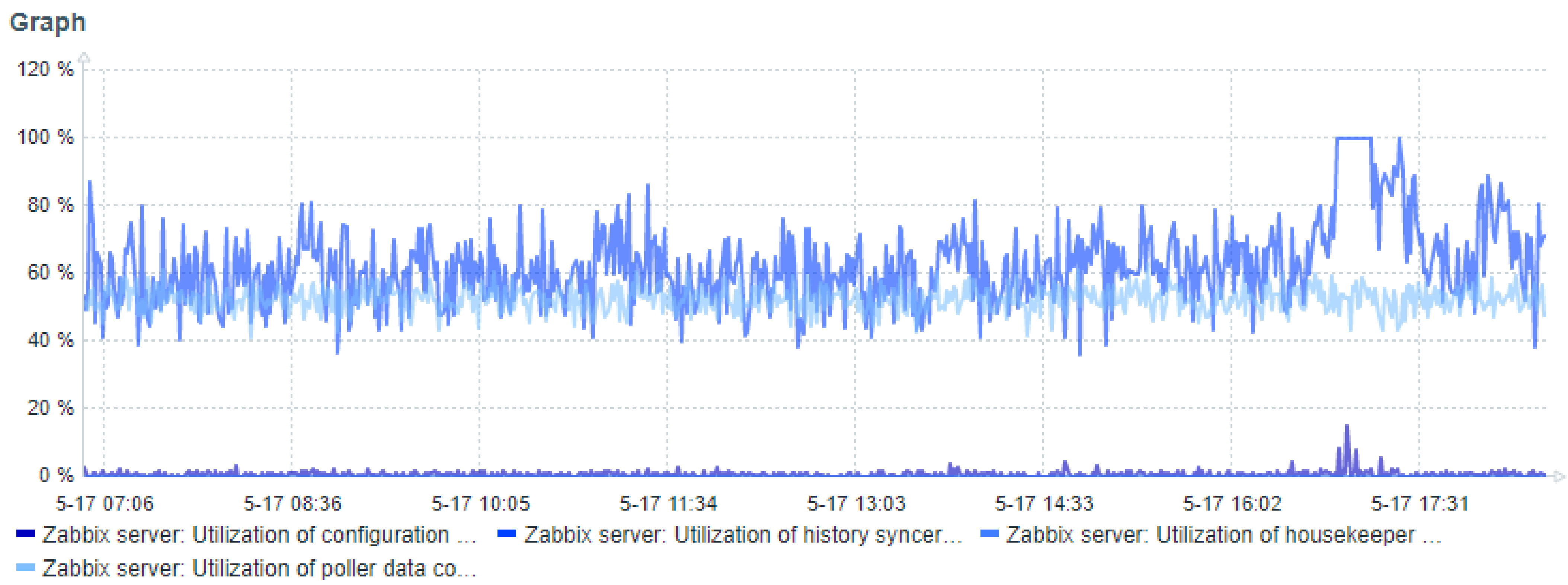
RHEIN-MAIN SOLUTIONS

PERFORMANCE AFTER

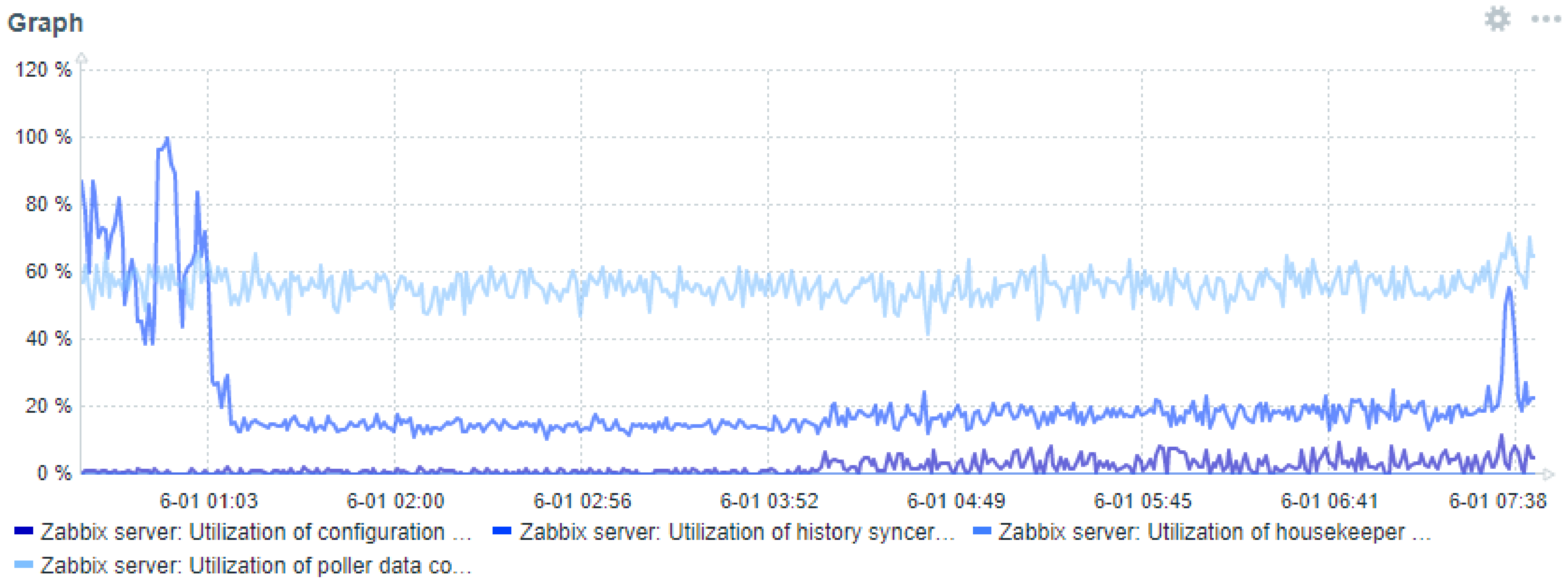


RHEIN-MAIN SOLUTIONS

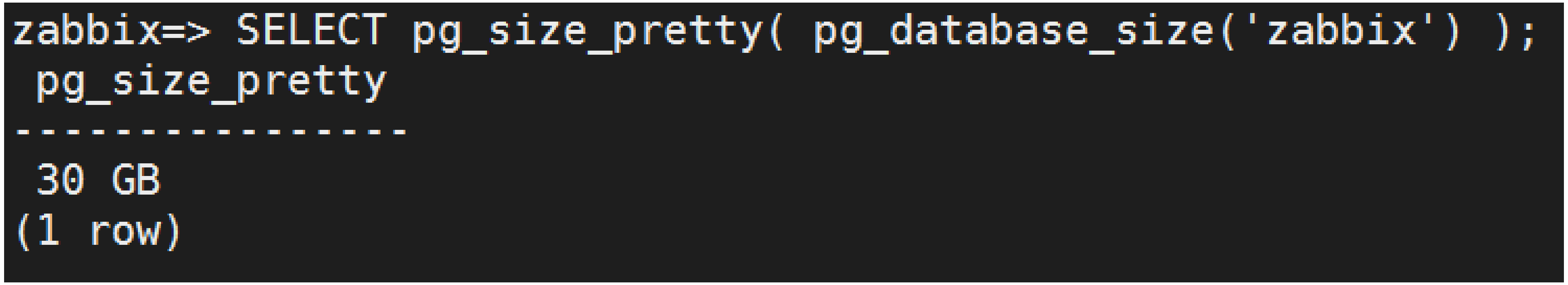
PERFORMANCE BEFORE



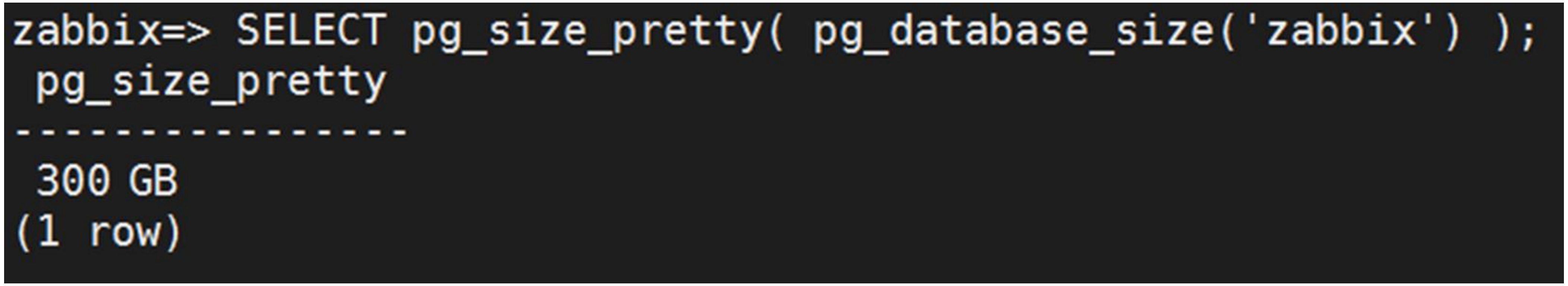
PERFORMANCE AFTER



DATABASE SIZE BEFORE



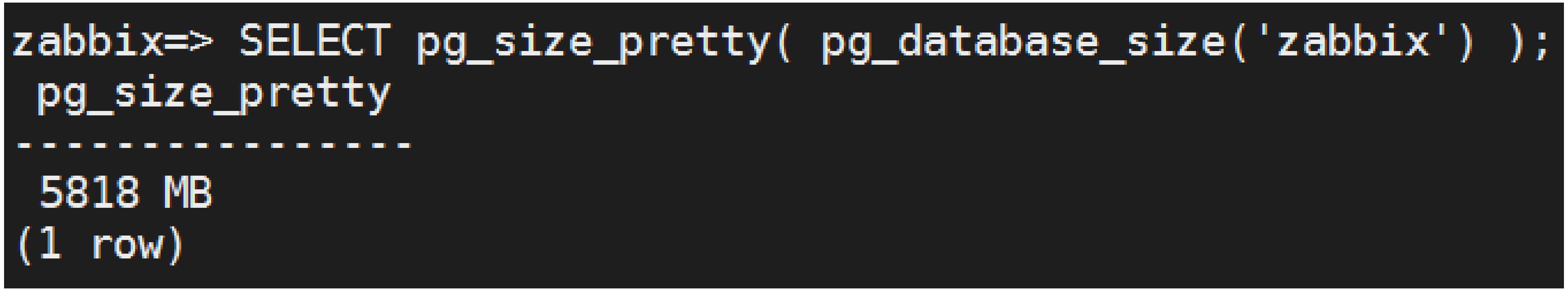
```
zabbix=> SELECT pg_size_pretty( pg_database_size('zabbix') );
 pg_size_pretty
----------------
 30 GB
(1 row)
```



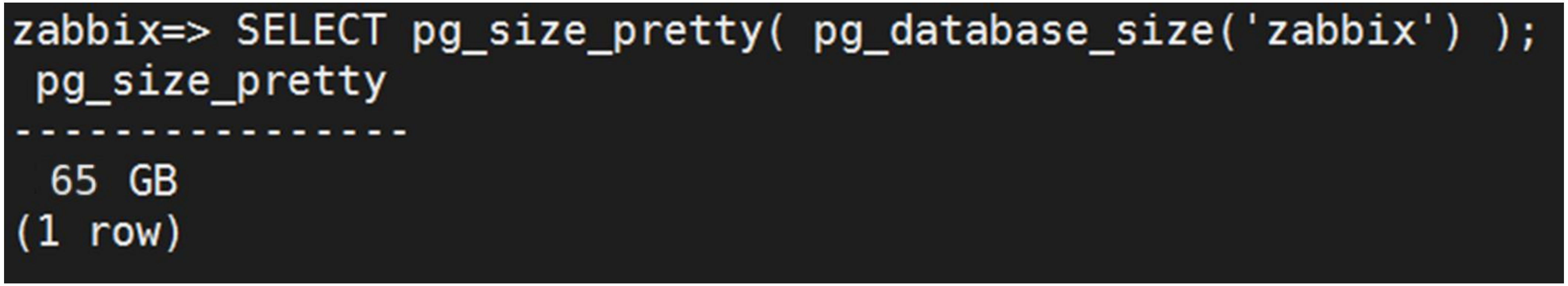
```
zabbix=> SELECT pg_size_pretty( pg_database_size('zabbix') );
 pg_size_pretty
----------------
 300 GB
(1 row)
```

RHEIN-MAIN SOLUTIONS

DATABASE SIZE AFTER



```
zabbix=> SELECT pg_size_pretty( pg_database_size('zabbix') );
 pg_size_pretty
----------------
 5818 MB
(1 row)
```

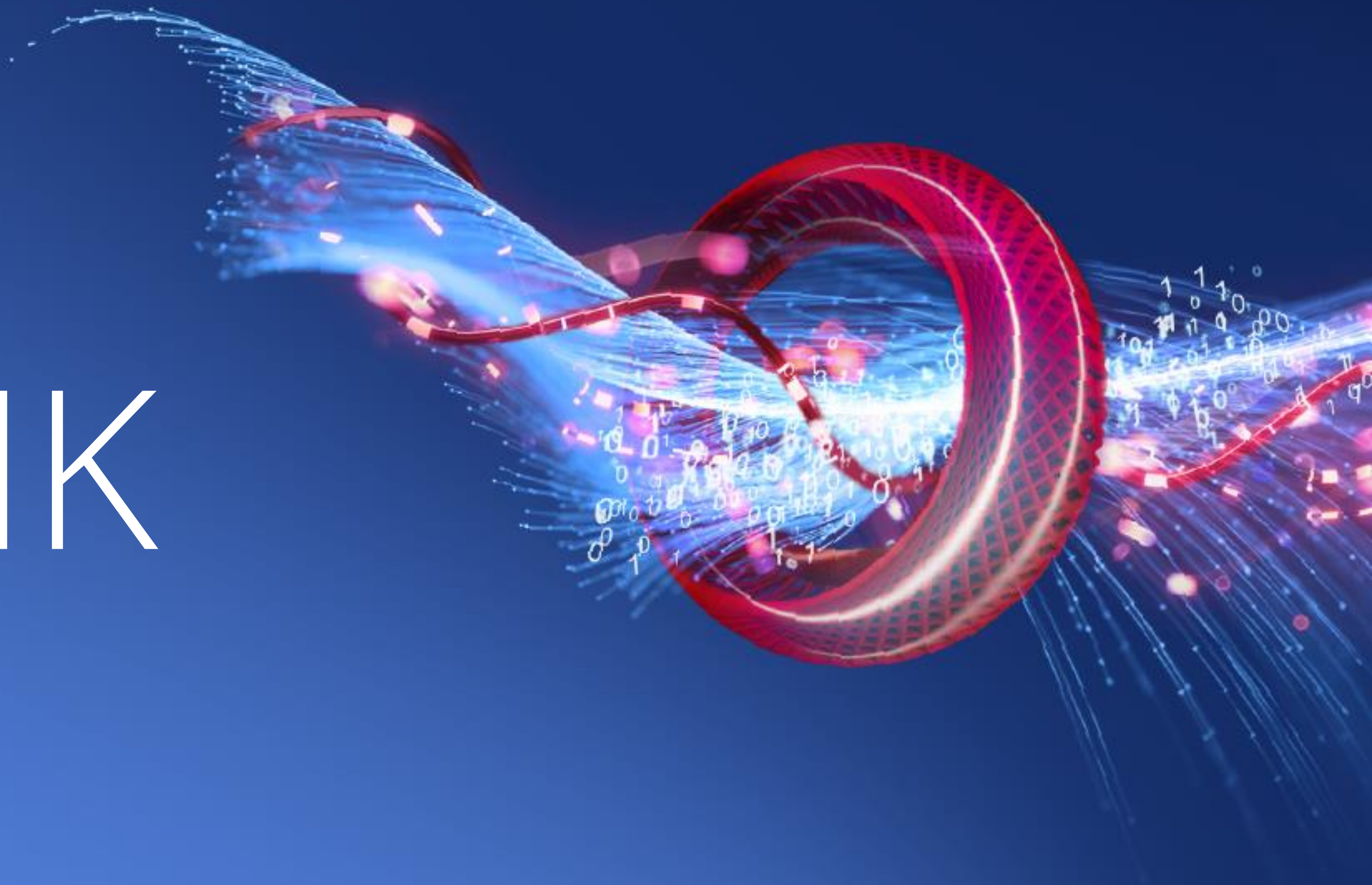


```
zabbix=> SELECT pg_size_pretty( pg_database_size('zabbix') );
 pg_size_pretty
----------------
 65 GB
(1 row)
```

FOREWORD (A BIT OF COOLING OFF)

- **It’s still experimental**
- **Compressed chunk modifications (inserts, deletes, updates) are not allowed**
- **What about other databases? (IBM DB2 got dropped)**
- **Your average hardware**

THANK YOU!



RHEIN-MAIN SOLUTIONS