UPGRADING ZABBIX TO A NEWER VERSION AND TIMESCALEDDB

Aleksandrs Petrovs-Gavrilovs
Technical Support Engineer
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
Zabbix LTS (Long Term Support) releases are supported for Zabbix customers during five (5) years
*Standard:* until next release
WHICH VERSION TO CHOOSE?

5.0 LTS
02

WHAT IS AND WHY TIMESCALEDB
BUT WHY?

Is this POSTGRESQL

TIMESCALEDB
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 TIMESCALEDDB?

- **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
WHY TIMESCALEDDB - PERFORMANCE

PostgreSQL with TSDB extension
WHY TIMESCALEDB - PERFORMANCE

Comparison

TimescaleDB vs PostgreSQL
new values per second

TimescaleDB avg 130K NVPS; PostgreSQL avg 90K NVPS
WHY TIMESCALEDB - COMPRESSION

Up to 90% storage savings

<table>
<thead>
<tr>
<th>Workload</th>
<th>Uncompressed</th>
<th>Compressed</th>
<th>Storage Savings</th>
</tr>
</thead>
<tbody>
<tr>
<td>IT metrics (from Telco beta tester)</td>
<td>1396 GB</td>
<td>77.0 GB</td>
<td>94% savings</td>
</tr>
<tr>
<td>Industrial IoT monitoring data (from beta tester)</td>
<td>1.445 GB</td>
<td>0.077 GB</td>
<td>95% savings</td>
</tr>
<tr>
<td>IT metrics (DevOps dataset from TSBS)</td>
<td>125 GB</td>
<td>5.5 GB</td>
<td>96% savings</td>
</tr>
<tr>
<td>IoT monitoring data (IoT dataset from TSBS)</td>
<td>251 GB</td>
<td>23.8 GB</td>
<td>91% savings</td>
</tr>
</tbody>
</table>
HOW COMPRESSION WORKS

Before compression

<table>
<thead>
<tr>
<th>time</th>
<th>device_id</th>
<th>cpu</th>
<th>disk_io</th>
<th>energy_consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>12:00:02</td>
<td>1</td>
<td>88.2</td>
<td>20</td>
<td>0.8</td>
</tr>
<tr>
<td>12:00:01</td>
<td>2</td>
<td>300.5</td>
<td>30</td>
<td>0.9</td>
</tr>
<tr>
<td>12:00:01</td>
<td>1</td>
<td>88.6</td>
<td>25</td>
<td>0.85</td>
</tr>
<tr>
<td>12:00:01</td>
<td>2</td>
<td>299.1</td>
<td>40</td>
<td>0.95</td>
</tr>
</tbody>
</table>

After compression

<table>
<thead>
<tr>
<th>time</th>
<th>device_id</th>
<th>cpu</th>
<th>disk_io</th>
<th>energy_consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>[12:00:02, 12:00:02, 12:00:01, 12:00:1]</td>
<td>[1, 2, 1, 2]</td>
<td>[88.2, 300.5, 88.6, 299.1]</td>
<td>[20, 30, 25, 40]</td>
<td>[0.8, 0.9, 0.85, 0.95]</td>
</tr>
</tbody>
</table>
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
WHAT TO UPGRADE?
IN SHORT

Everything.

(Almost)
COMPONENTS TO UPGRADE

☑ Zabbix server
☑ Database (depending on version)
☑ Web frontend
☑ Proxy
☑ Java Gateway
UPGRADE NOT REQUIRED

- Zabbix agent
- Database (depending on version)
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:
   ```bash
   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
   ```bash
   yum install postgresql12-server
   ```
3. Stop the current one (don't forget the server)
   ```bash
   systemctl stop postgresql-9.6
   ```
4. Initialize the new one
   ```bash
   sudo su postgres
   cd ~
   /usr/pqsql-12/bin/initdb -D /var/lib/pqsql/12/data/
   ```
5. Migrate the data
   ```bash
   /usr/pqsql-12/bin/pg_upgrade --old-datadir /var/lib/pqsql/9.6/data/
   --new-datadir /var/lib/pqsql/12/data/ --old-bindir /usr/pqsql-9.6/bin/
   --new-bindir /usr/pqsql-12/bin/
   ```
6. Start new instance
   ```bash
   systemctl start postgresql-12
   ```
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

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

Or back up the configuration files

```bash
$ 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.
   ```bash
   systemctl stop zabbix-server
   ```

2. Upgrade your current repository package
   ```bash
   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
   ```bash
   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**


   
   **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/
   repo_gpgcheck=1
   gpgcheck=3
   enabled=1
   gpgkey=https://packagecloud.io/timescale_timescaledb/gpgkey
   sslverify=1
   selcacert=/etc/pki/tls/certs/ca-bundle.crt
   metadata_expire=3600
   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 TIMESCALEDDB**

1. Start Zabbix server to update the DB schema
2. Stop Zabbix server
3. Enable TimescaleDB extension:
   ```bash
   echo "CREATE EXTENSION IF NOT EXISTS timescaledb CASCADE;" | sudo -u postgres psql zabbix
   ```
4. Then run the `timescaledb.sql` script located in `database/postgresql`
   ```bash
   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 TIMESCALEDDB**

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.

```sql
[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_hypetable
-------------------
(1,public,history,t)
(1 row)

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

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

create_hypetable
-------------------
(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_hypetable
```
PREPARE YOUR DB FOR TIMESCALEDDB

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

Update  Reset defaults
WHAT NEXT?

Enjoy monitoring!
05

RESULTS
SHOW OFF RESULTS
PERFORMANCE BEFORE

Graph

- Zabbix server: Number of processed values per second
PERFORMANCE AFTER

Graph

Zabbix server: Number of processed values per second
PERFORMANCE BEFORE
PERFORMANCE AFTER
DATABASE SIZE BEFORE

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

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