initMAX

Running Zabbix with PostgreSQL as backend DB and migrating from MySQL

Prague PostgreSQL Developer Day 2023

1

About initMAX s.r.o.


Zabbix
Monitoring
LOG Management
& Wazuh
OpenSource
IDM
PostgreSQL DBA
& Consulting
DevOps
& Automation

Zabbix
Monitoring

LOG Management
& Wazuh

OpenSource
IDM

PostgreSQL DBA
& Consulting

DevOps
& Automation


LOG Management
& Wazuh
Zabbix
Monitoring
OpenSource
IDM
PostgreSQL DBA
& Consulting
DevOps
& Automation


LOG Management
& Wazuh
Zabbix
Monitoring
OpenSource
IDM
PostgreSQL DBA
& Consulting
DevOps
& Automation


LOG Management
& Wazuh
Zabbix
Monitoring
OpenSource
IDM
PostgreSQL DBA
& Consulting
DevOps
& Automation


Zabbix
Monitoring
LOG Management
& Wazuh
OpenSource
IDM
PostgreSQL DBA
& Consulting
DevOps
& Automation

initMAX

About us

Jakub

DBA & TECHNICAL CONSULTANT

Our Team

Tomáš

CEO & CERTIFIED ZABBIX TRAINER

Alois

TECHNICAL CONSULTANT

Miroslav

DBA & TECHNICAL CONSULTANT

Tomáš

DEVELOPER

Vojta

TECHNICAL CONSULTANT

Petr

DEVELOPER

initMAX

Our customers

initMAX

Our PostgreSQL customers

- The largest Czech bank. Part of the Erste Group

- An insurance company with a 30-year tradition. Part of Vienna Insurance Group

- One of the largest online consumer electronics retailers in Central Europe

2

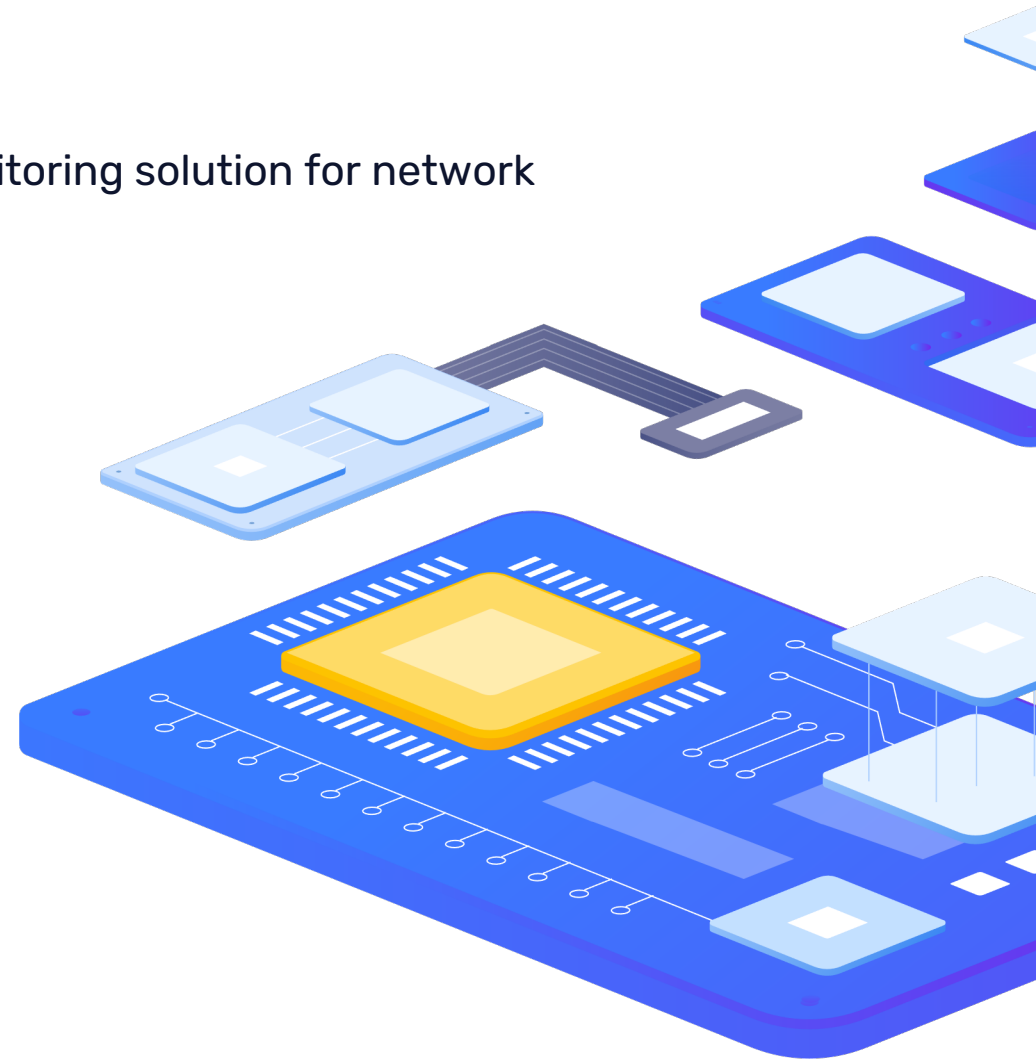
What is Zabbix and which DB engines uses

initMAX

What Zabbix is

- Zabbix is a mature and effortless enterprise-class open source monitoring solution for network monitoring and application monitoring of millions of metrics.

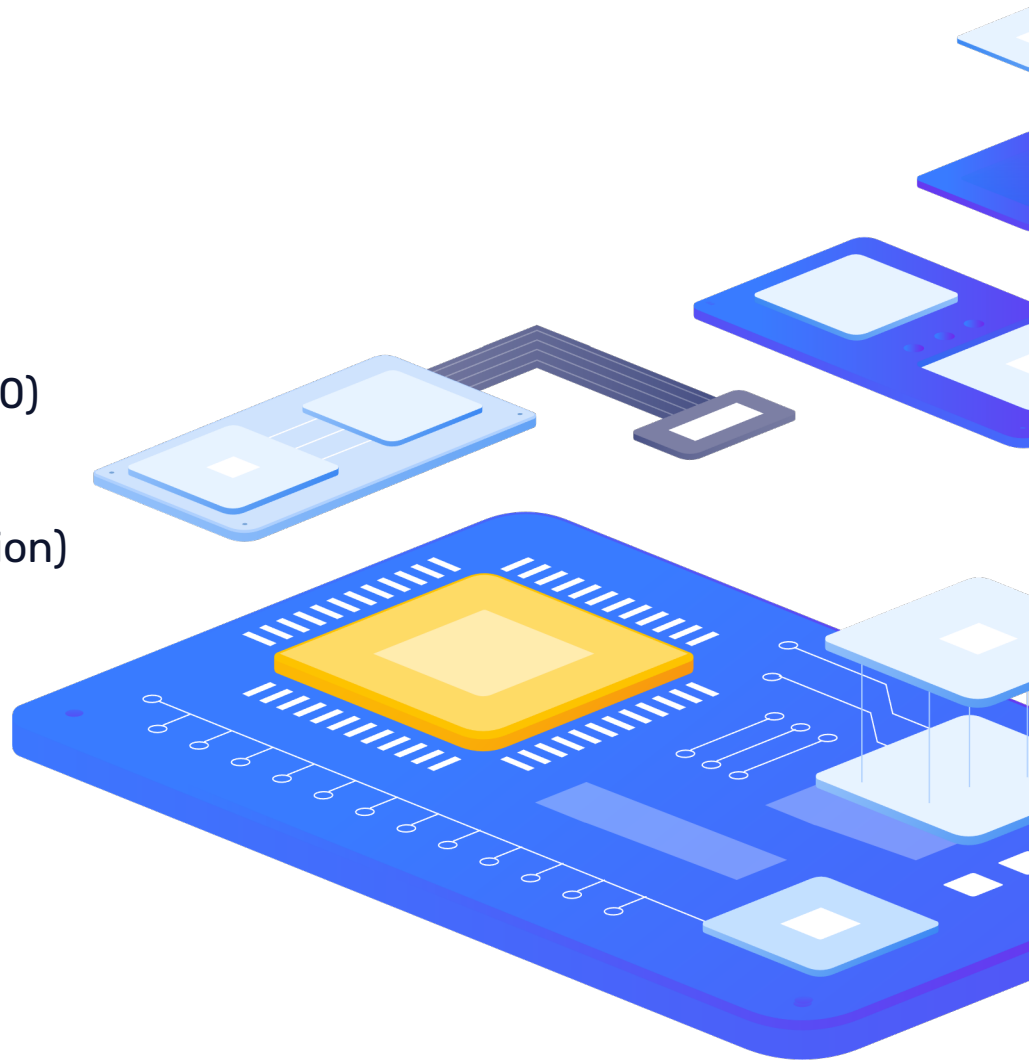
- Zabbix offers solutions for many different use cases and areas
  - Hardware monitoring
  - OS monitoring
  - Virtual machine monitoring (VMWare, Docker, Podman etc.)
  - Cloud monitoring (Azure, AWS etc.)
  - Application monitoring
  - Web monitoring
  - Visualizace
  - Business services/SLA monitoring



initMAX

Zabbix history

- Zabbix has been available since 2001 and is licensed under GPL2
- Supports PostgreSQL as a backend DB from the beginning
- The oldest officially supported version of Zabbix is 4.0 LTS
  - The oldest supported version of PostgreSQL is 8.1 (for Zabbix 4.0)
- Since Zabbix 4.2 TimescaleDB is experimentally supported
- Since version 4.4 TimescaleDB is fully supported (without compression)
- Since Zabbix 5.0.26 support for TimescaleDB compression
  - At customer site first deployment since Zabbix 5.0.28
    - PostgreSQL 10.x
    - TimescaleDB 1.7.x

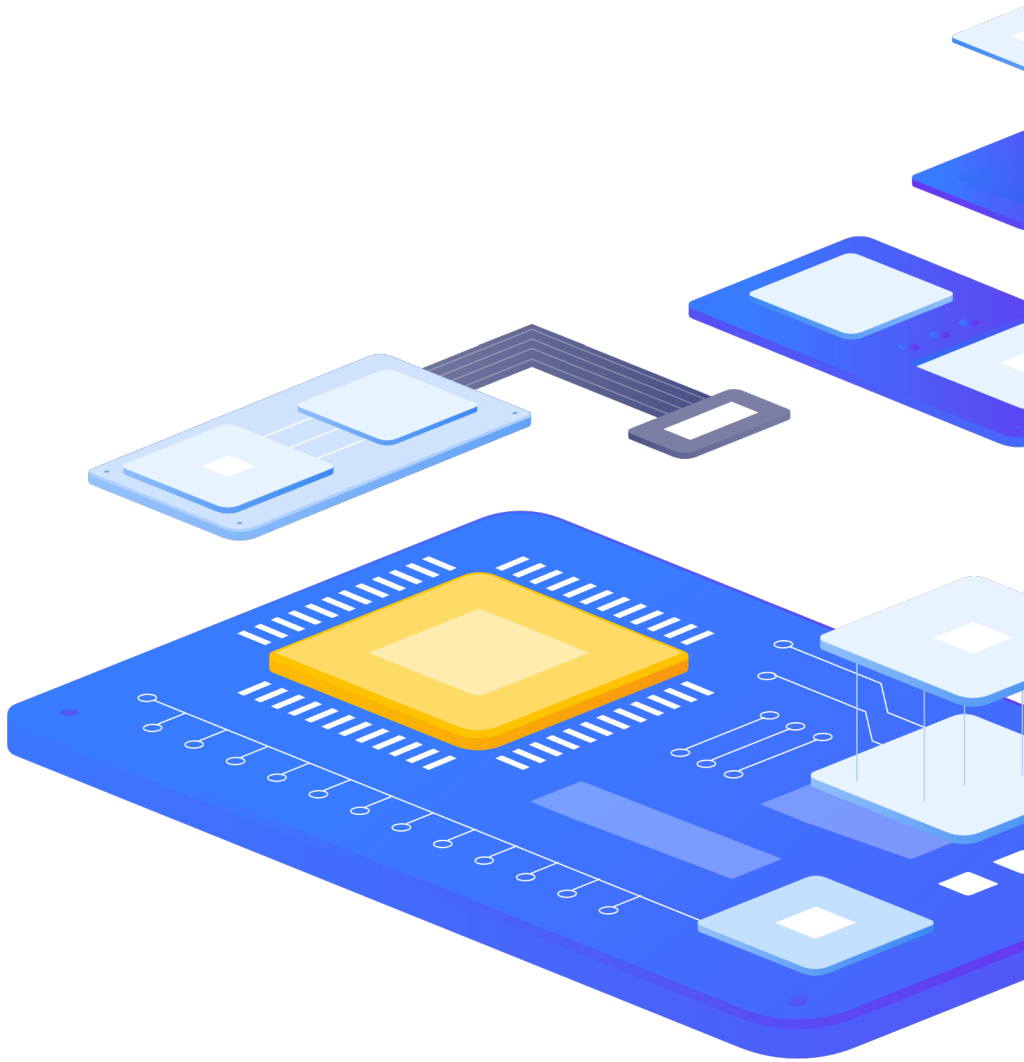


initMAX

Zabbix architecture

- **Server**
  - Programmed in C language
  - Breathtaking change in Zabbix version 6.0 better DB connection optimalization
  - Zabbix offers a native high-availability
  - For high-availability DB we use Patroni

- **Front-end**
  - Programmed in PHP language (7.4.0 or later)
  - PgBouncer for connection time optimalization



initMAX

Which DB engines Zabbix uses in LTS version 6.0

- **Following database engines are supported:**
  - MySQL / Percona (8.0.x)
  - MariaDB (10.5.00-10.10.x)
  - Oracle (19c - 21c)
  - **PostgreSQL** (13.0-15.x)
  - **TimescaleDB for PostgreSQL** (2.0.1-2.9.x)
- **MySQL DBs was usualy first choice**
  - Performance issues in larger environments
  - This led to the idea of migrating to PostgreSQL
  - TimescaleDB
- **Our problems**
  - We found memory leak in TimescaleDB version 2.7.1 (fixed in 3 days)

3

Why did we decide to migrate to PostgreSQL

initMAX

Advantages of PostgreSQL

- Open Source
  - Community driven
- Low cost
  - No licence fees
  - No contract problems
- Great extensibility
  - Compression (TimescaleDB Community Edition)
  - More… we will discuss it later
- Stability and durability
- Scalability
- We are able to use Zabbix Frontend for managing partitioning and compression

initMAX

Advantages of PostgreSQL

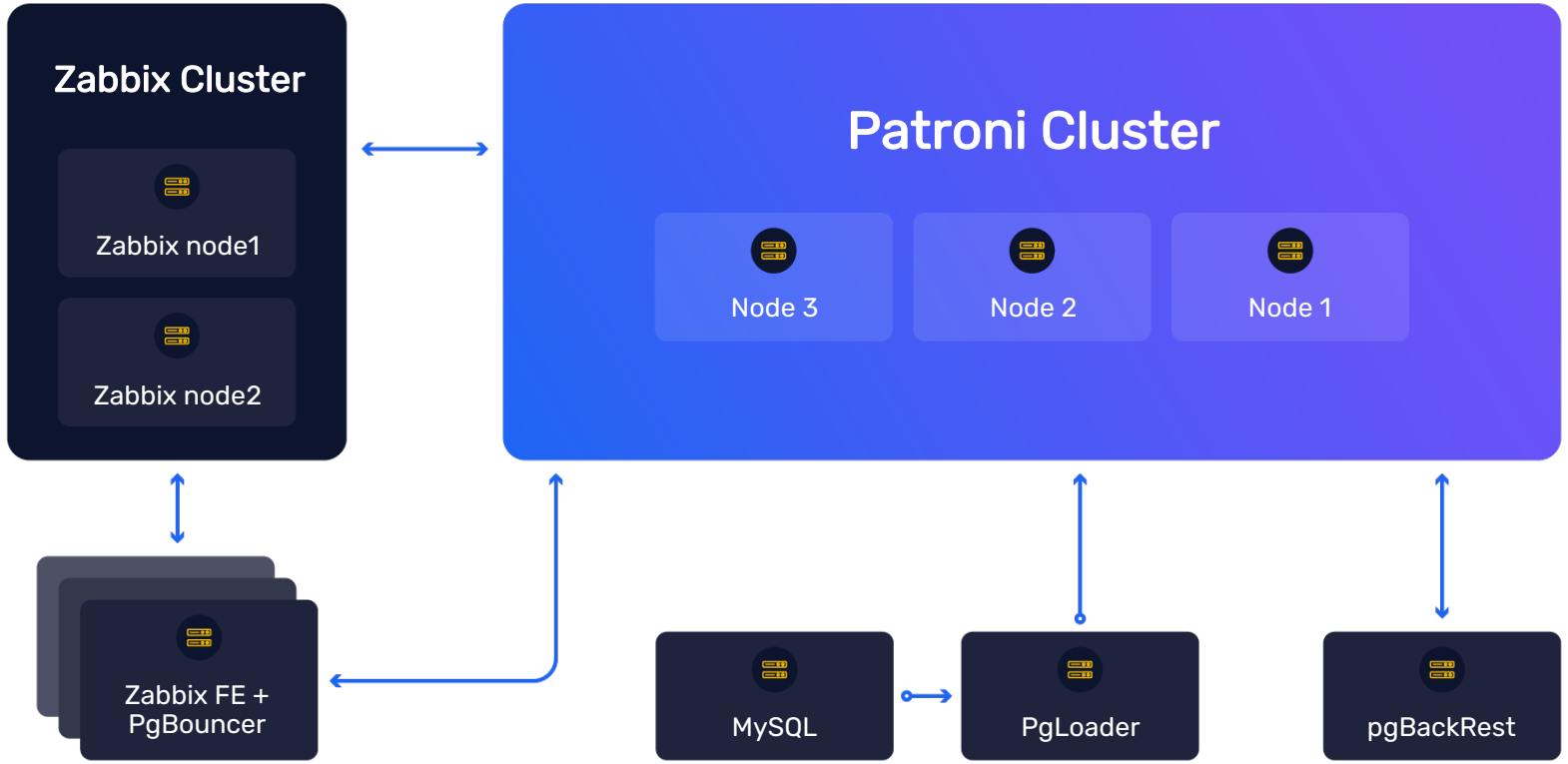
- Zabbix DB load example (hosts 4.5k, 3.5k NVPS, 600k items)
  - TPS 2-3 k
  - QPS 5-8 k
  - Avg. WAL rate 1-3 MB/s
  - Select queries 20%, 4-6 k wps
  - Touples read vs. write ratio 95%
- DB size, 500GB-800GB in MySQL would take about 1TB–2 TB
- Where MySQL consumed almost 100% CPU, PostgreSQL consumed less than 15%
- Slow GUI on MySQL, rocket on PostgreSQL with TimescaleDB

4

Hands-on experience with PostgreSQL

initMAX

Environment example



Prague PostgreSQL Developer Day 2023

initMAX

Basic PostgreSQL Tuning and optimalization

- Handy tools for basic tuning
  - https://pgtune.leopard.in.ua/
  - https://pgconfigurator.cybertec.at/
- TimescaleDB tuning tool (`timescaledb-tune`)
  - Handles setting the most common parameters to good values based on your system. It accounts for memory, CPU, and PostgreSQL version
- RAM, WAL, `parallel workers, max_connections, fill_factor`
- Kernel tuning (swap, mem overcommit, fs...)
- Security (ssl, scram, rbac, restrict public permissions)
- Regular base vacuum
- pg_stat_statements and other usefull tools
- Logging
- Ansible for instalation and optimalization

initMAX

Monitoring

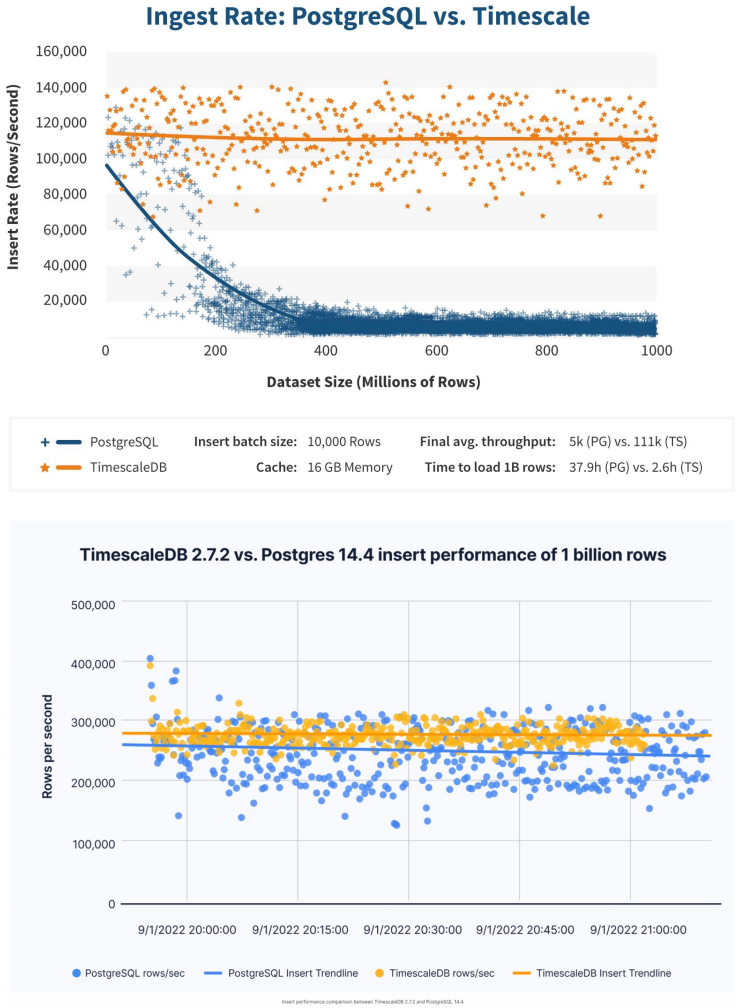
- Zabbix agent2 – you can join our webinar
- Alternatively pgWatch2
- What do we mainly watch?
  - CPU
  - Shared buffers HIT ratio > 95%
  - TPS
  - Idle in transaction sessions
  - Active sessions
  - Blocked sessions
  - Checkpoint requested
  - WAL rate
  - WAL archiving
  - Sequential scan
  - Buffers cleanup
  - Temp bytes
  - Row fetched vs returned

initMAX

Extensions we can't live without

- **TimescaleDB**
  - TimescaleDB is an open-source relational database for time-series data
  - Better performance at scale
  - Lower storage costs
  - Automatic partitioning (reduces bloat)
  - 1000x faster for some queries
  - Be carefull with licensing at the begining (Apache/Community)
  - Read carefully documentation and release notes when upgrading

- **pg_stat_statements**
  - Provides a means for tracking planning and execution statistics of all SQL statements executed by a server
  - Allows you to quickly identify problematic queries
  - Providing instant visibility into your database performance



https://www.timescale.com/blog/postgresql-timescaledb-1000x-faster-queries-90-data-compression-and-much-more/

initMAX

Extensions

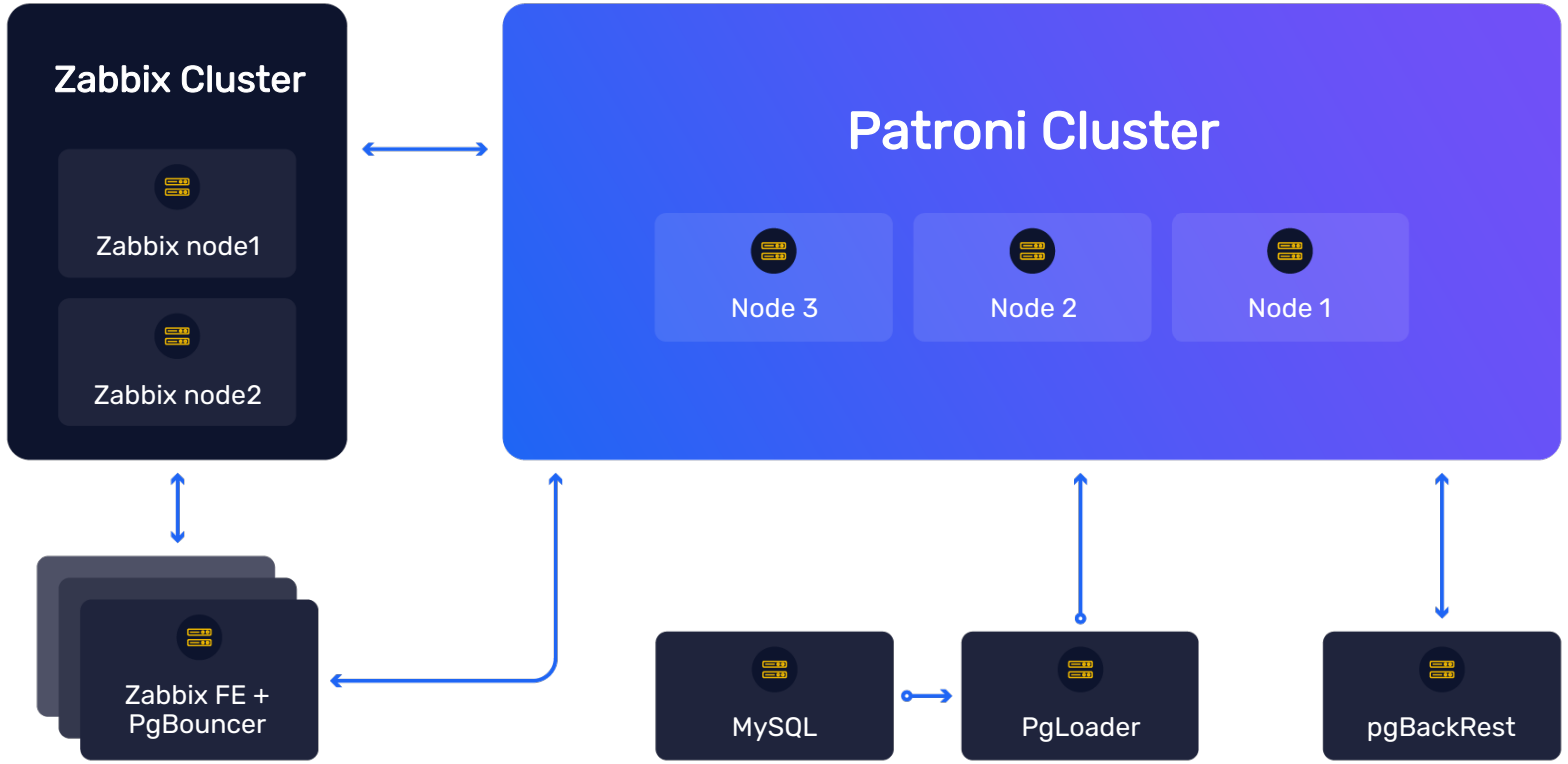
- **pg_repack**
  - This extension lets you remove bloat from tables and indexes
  - Works online, without holding an exclusive lock on the processed tables during processing
  - Usually not used (we do vacuum regularly)

- **pg_cron**
  - Simple job scheduler that runs inside the database
  - Uses the same syntax as regular cron
  - Parsing and scheduling comes directly from the cron source code by Paul Vixie

- **Plpython3u**
  - Extension which allows PostgreSQL functions and procedures to be written in the Python language
  - Handy for system monitoring and Data Science

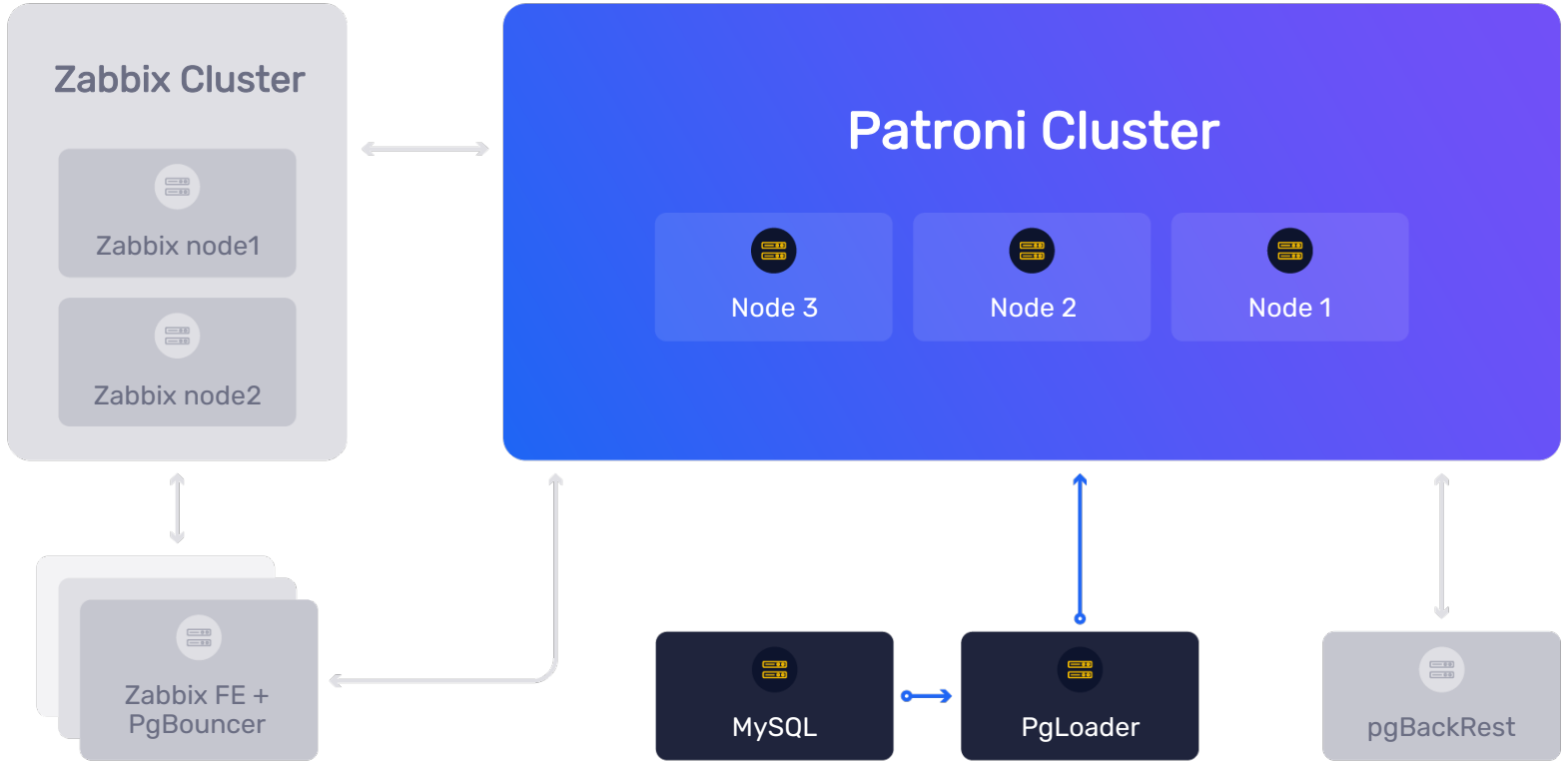
initMAX

PgLoader - Location in the environment



initMAX

PgLoader - Location in the environment



initMAX

PgLoader - Migration from MySQL

- Loads data from various sources into PostgreSQL
- Many source formats supported
- On the fly data transformation
- Reading files from an archive
- On error stop/On error resume next
- Pre/Post SQL commands

- **Installation**
  - Deb - from apt.postgresql.org
  - RPM - from yum.postgresql.org

initMAX

PgLoader - Migration from MySQL

- How to run pgloader

```
pgloader --root-dir /tmp/pgloader --logfile /tmp/pgloader/pgloader.log /tmp/pgloader/loader.conf
```

- Sample configuration

```
LOAD DATABASE
FROM mysql://user:pass@localhost/zabbix
INTO postgresql://user:pass@localhost/zabbix?sslmode=require
WITH include no drop,
truncate,
create no tables,
create no indexes,
no foreign keys,
reset sequences,
data only
ALTER SCHEMA 'zabbix' RENAME TO 'public';
```

initMAX

PgLoader - Migration from MySQL

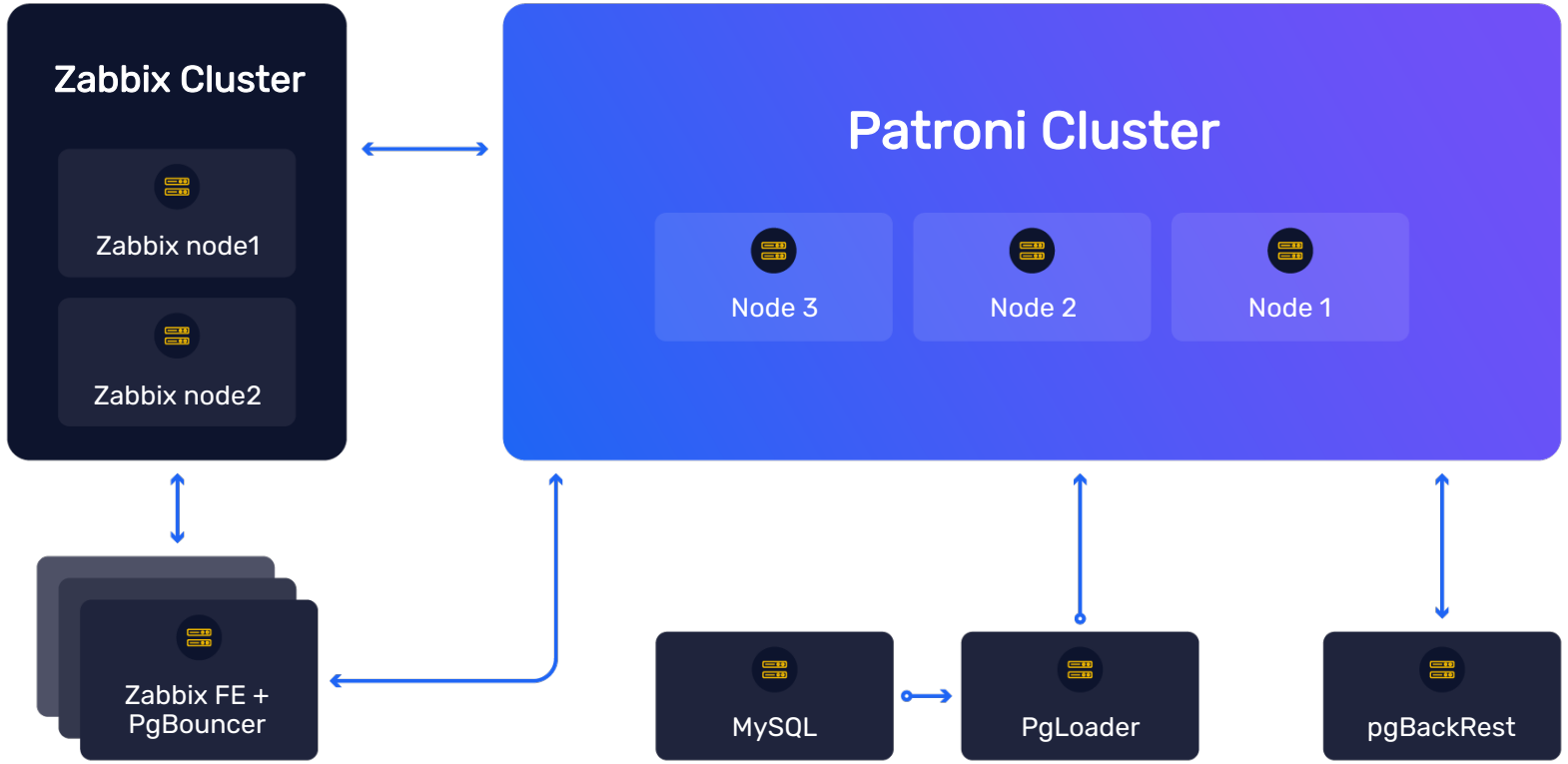
- **700GB MySQL database took 10hours to migrate**
  - Migration scenarios

- **Migration process**
  - Downloads and extracts Zabbix sources
  - Creates DB and user in PostgreSQL
  - Creates extension TimescaleDB
  - Creates database schema without constraints and indexes (from Zabbix sources)
  - Configures PgLoader
  - Runs PgLoader
  - Creates constraints and indexes (from Zabbix sources)
  - Creates hypertables (from Zabbix sources)
  - Enables compression

- **Migration process is different between of zabbix versions**

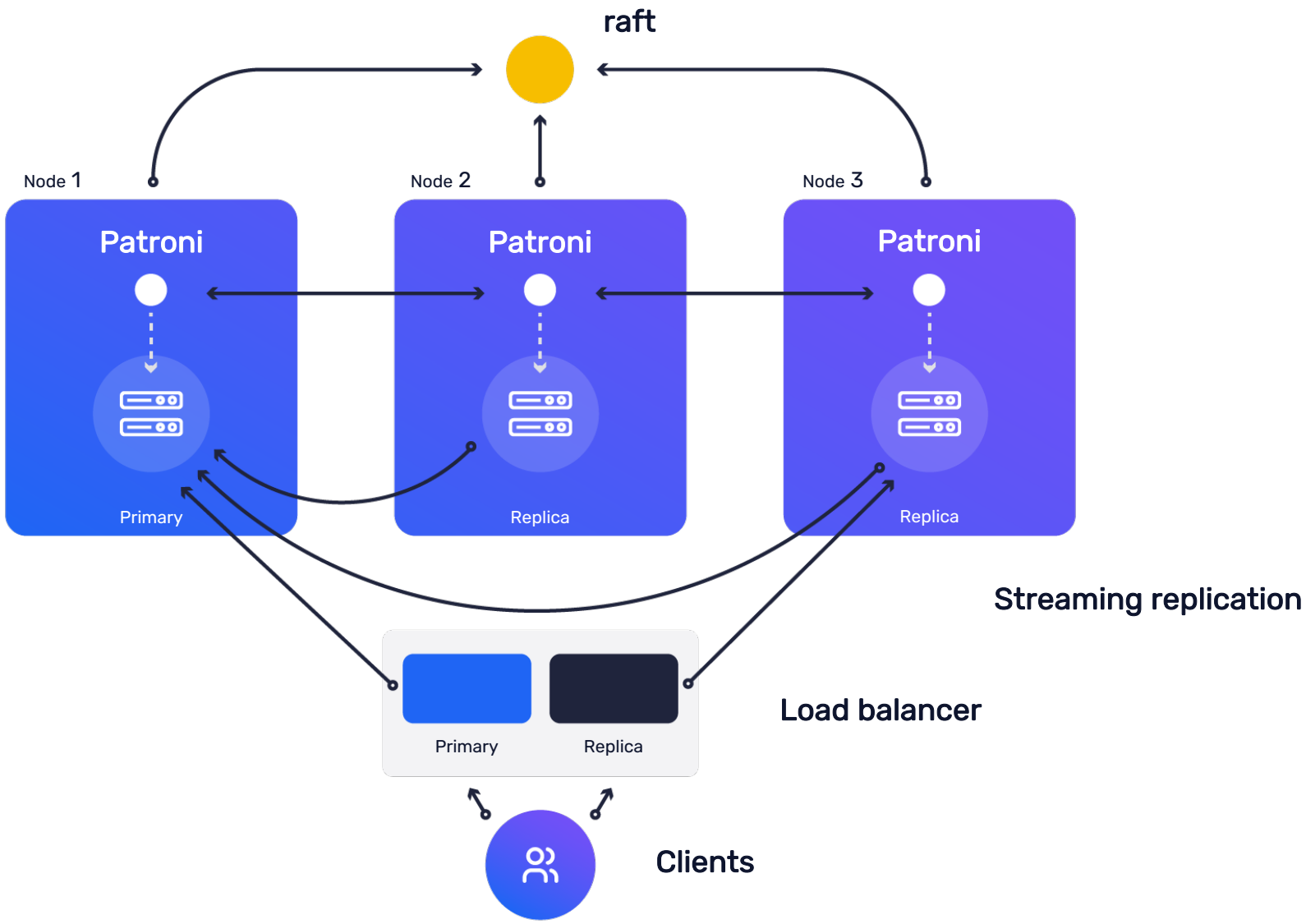
initMAX

Patroni



initMAX

Patroni

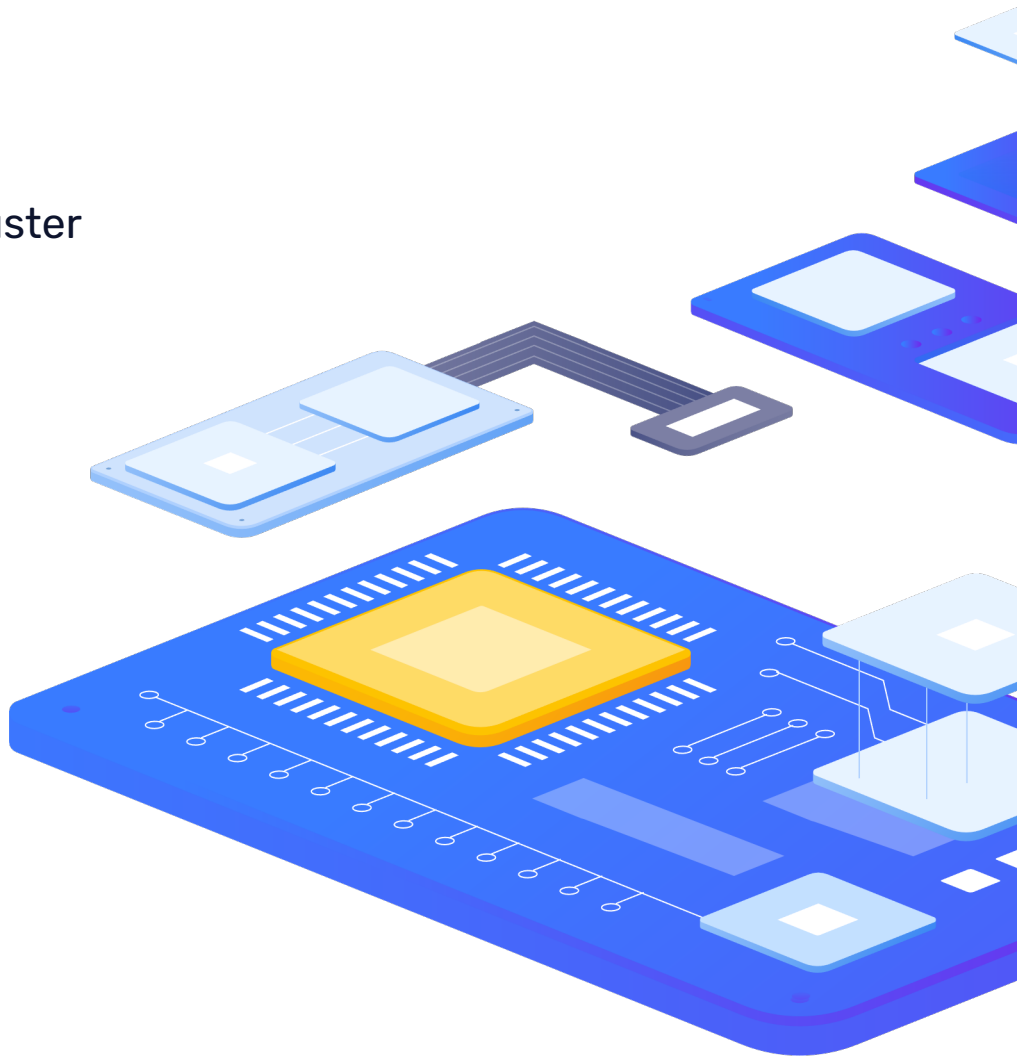


initMAX

Patroni

- Written in python (psycopg2 or psycopg 3.0)
- Provides a template for configuring a highly available PostgreSQL cluster
- Using PostgreSQL streaming replication
- Asynchronous/Synchronous replication
- Using distributed configuration stores (Etcd, Consul, pure Raft, ...)
- Cluster management via CLI or API
- Watchdog support for additional split-brain prevention
- Custom scripts to clone a new replica (pgBackRest, Wal-G, ...)
- Major upgrade of PostgreSQL version is a bit tricky
- Proven HA solution, very durable and reliable

- **Installation**
  - We recommend installation via pip



initMAX

Patroni

- We don't use external raft (etcd, Consul...)
  - We run Patroni on pure Raft (PySyncObj)
  - Situation has changed after new version 3.0 was released
- libpg - from version 10 multihost connection string
- LoadBalancer or VIP - Zabbix does not support multi host connection string
- Keepalived for VIP handling (can't use vip-manager)
- HA-Proxy

```
…
listen primary
    bind *:5000
    option httpchk OPTIONS /primary
    http-check expect status 200
    default-server inter 3s fall 3 rise 2 on-marked-down shutdown-sessions
    server patroni01.initmax.cz patroni01.initmax.cz:5432 maxconn 160 port 8008 check
    server patroni02.initmax.cz patroni02.initmax.cz:5432 maxconn 160 port 8008 check
    server patroni03.initmax.cz patroni03.initmax.cz:5432 maxconn 160 port 8008 check

listen replicas
    bind *:5001
    option httpchk OPTIONS /replica
…
```

initMAX

Patroni

- Management using CLI or API

```
patronictl list my-ha-cluster -e
+ Cluster: my-ha-cluster (7070430412368999376) ---------+----+-----------+-----------------+-------------------+------+
| Member          | Host             | Role    | State   | TL | Lag in MB | Pending restart | Scheduled restart | Tags |
+-----------------+------------------+---------+---------+----+-----------+-----------------+-------------------+------+
|    patroni01    | 192.168.240.131 | Replica | running | 15 |         0 |                 |                   |      |
|    patroni02    | 192.168.240.132 | Replica | running | 15 |         0 |                 |                   |      |
|    patroni03    | 192.168.240.133 | Leader  | running | 15 |           |                 |                   |      |
+-----------------+------------------+---------+---------+----+-----------+-----------------+-------------------+------+
```

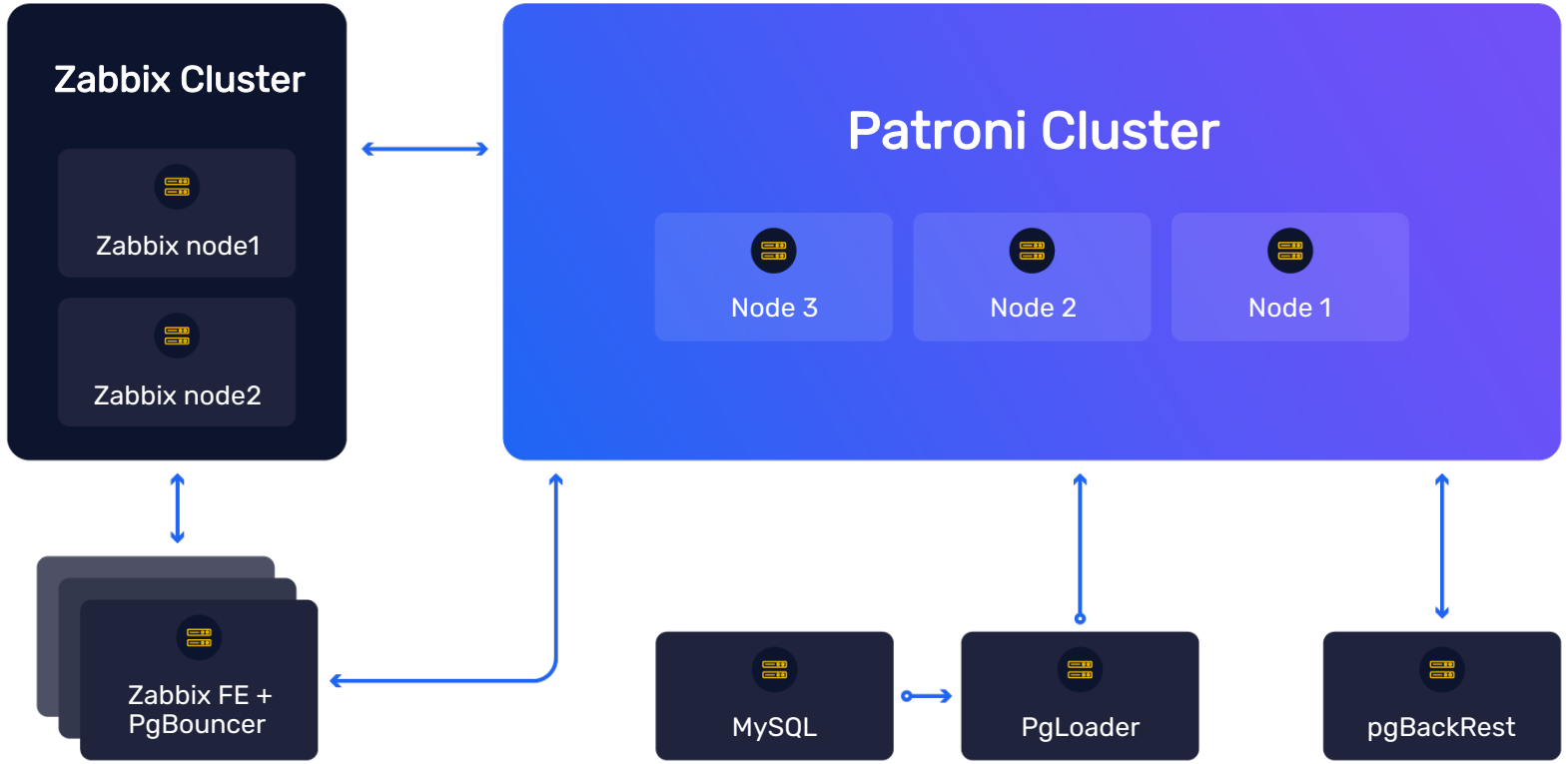
- API endpoint for leader and replica node Discovery

```
curl -skI -XGET https://localhost:8008/replica
HTTP/1.0 503 Service Unavailable
```

```
curl -skI -XGET https://localhost:8008/primary
HTTP/1.0 200 OK
```

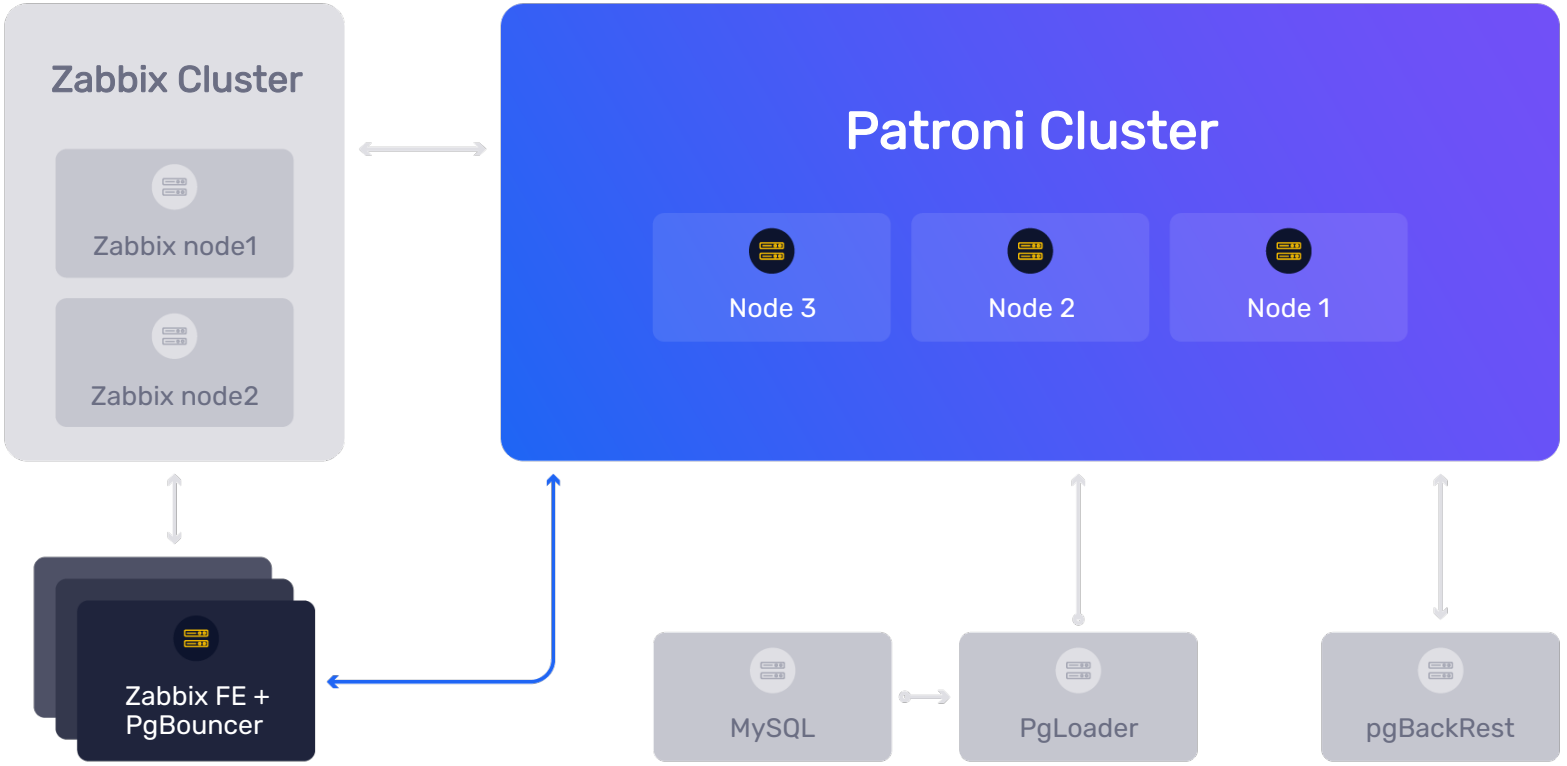
initMAX

PgBouncer



initMAX

PgBouncer

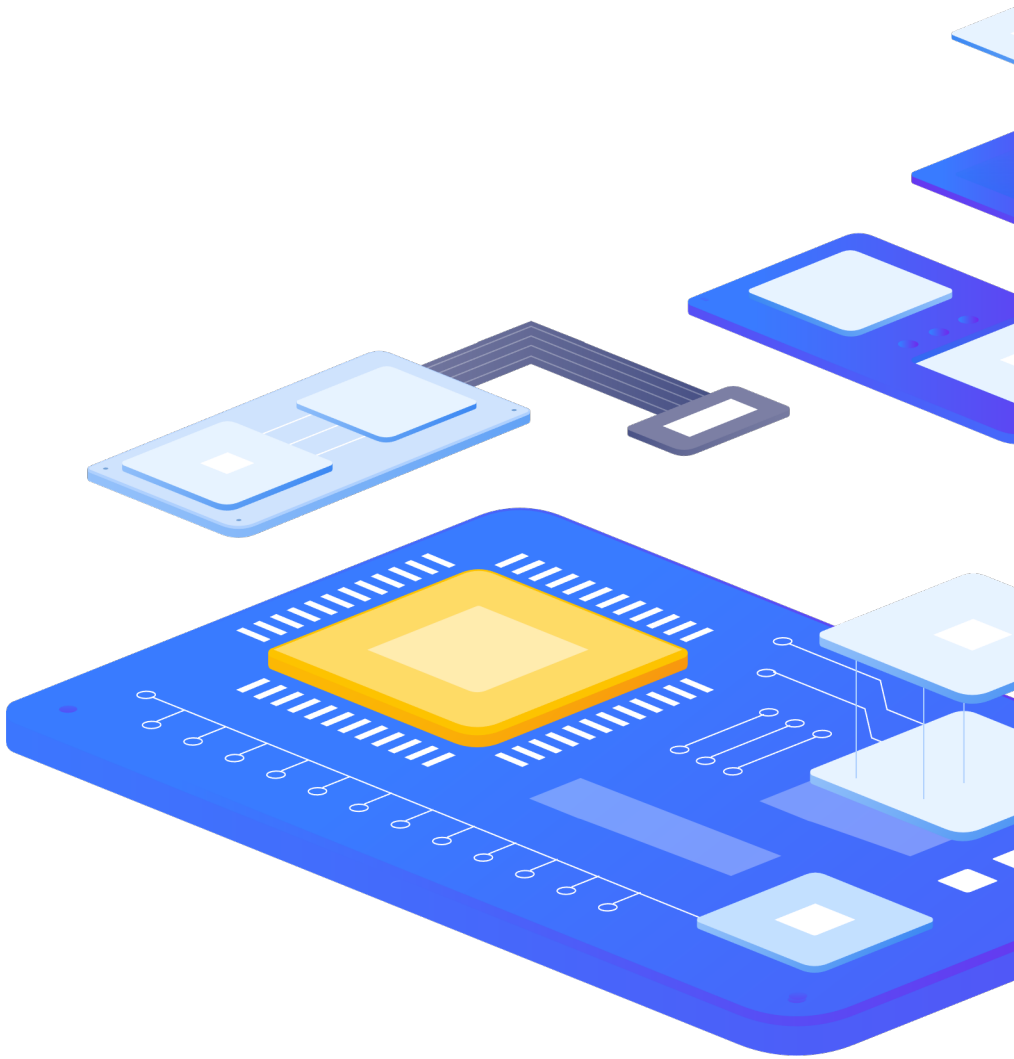


initMAX

PgBouncer

- Open Source, lightweight connection pooler for PostgreSQL
- Effective way to improve the performance of apps and decrease the load on PostgreSQL servers (40% - 60%)
- High CPU load (system time) is the indicator
- `pool_mode = statement | transaction | session`
- Prepared statements only in session pool mode
- Programmed in C, libevent
- Single core
- Scaling is handled externally
  - Multiple instances
  - Since version 1.12 `SO_REUSEPORT` option

- **Installation**
  - Deb - from apt.postgresql.org
  - RPM - from yum.postgresql.org



initMAX

PgBouncer

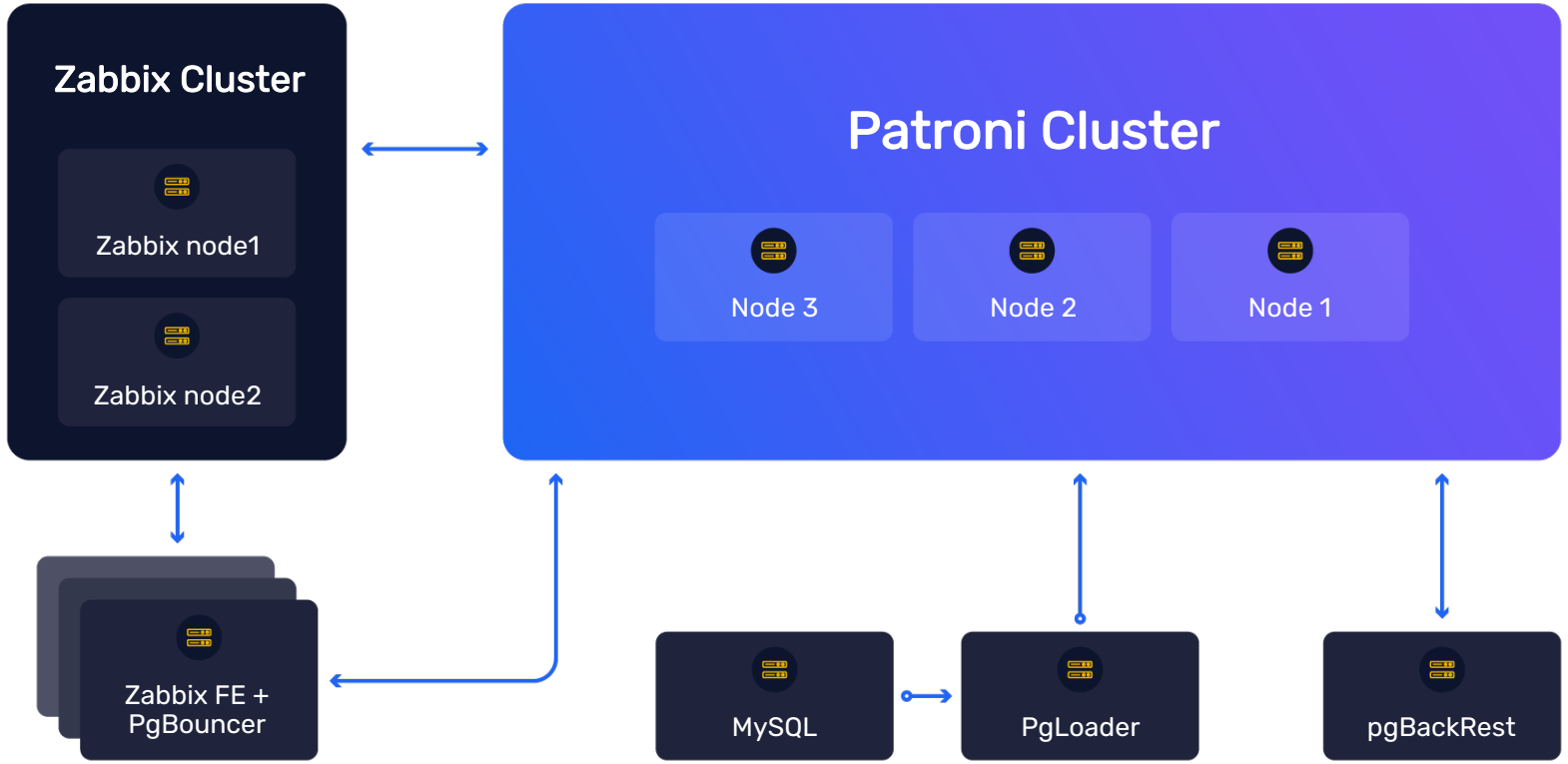
- In case of multiple APP servers installation on the DB server
- Otherwise installed on APP server
- AUTH_QUERY (template1)

```
auth_query = SELECT uname, phash FROM pgbouncer.user_lookup($1)
```

```
CREATE OR REPLACE FUNCTION pgbouncer.user_lookup(
in i_username text,
out uname text,
out phash text
) RETURNS record AS $$
BEGIN
SELECT usename, passwd FROM pg_catalog.pg_shadow WHERE usename = i_username INTO uname, phash;
RETURN;
END;
$$ LANGUAGE plpgsql SECURITY DEFINER;
```

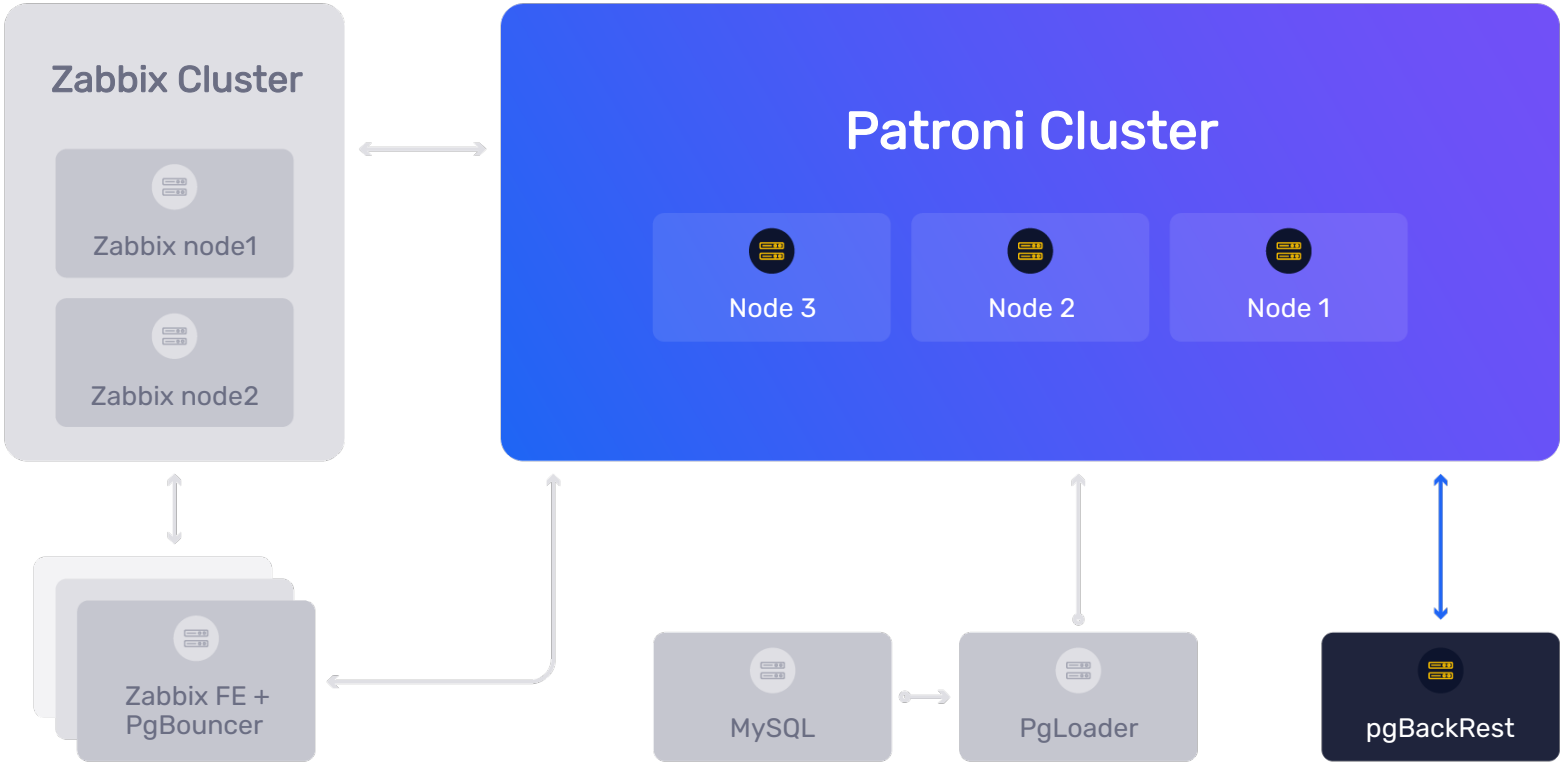
initMAX

pgBackRest



initMAX

pgBackRest

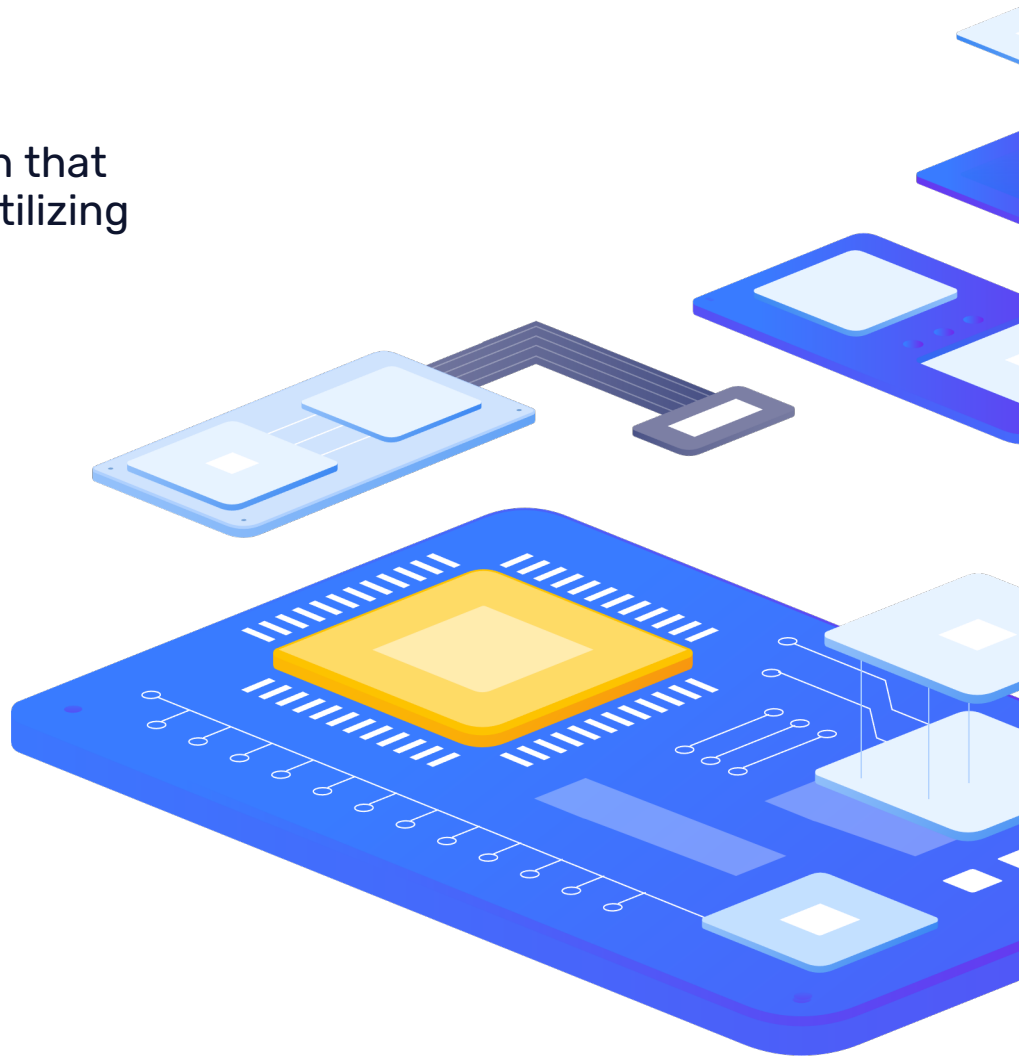


Prague PostgreSQL Developer Day 2023

initMAX

pgBackRest

- Powerfull tool specialized to easy-to-use backup and restore solution that can seamlessly scale up to the largest databases and workloads by utilizing algorithms that are optimized for database-specific requirements.
- Parallel Backup & Restore
- Local or Remote Operation
- Multiple Repositories
- Full, Incremental, Differential Backups
- Point in Time Recovery
- Backup Rotation & Archive Expiration
- Backup Integrity
- Backup Resume
- Streaming Compression & Checksums
- S3, Azure, and GCS Compatible Object Store Support
- Encryption
- Compatibility with PostgreSQL >= 9.0



initMAX

pgBackRest

- Backup examples

```
pgbackrest --stanza=zabbixdb --type=full backup
pgbackrest --stanza=zabbixdb --type=diff backup
```

- Restore examples

```
pgbackrest --stanza=zabbixdb --delta restore
pgbackrest --stanza=zabbixdb --type=time --target="2023-01-25 08:55:50" --delta restore
```

- Info about backups

```
pgbackrest --stanza=zabbixdb info
```

- Numbers as an illustration of performance – on 1 GigE 1TB DB/hr (backup)
- 700G full backup 40min

initMAX

pgBackRest - info

```
stanza: zabbixdb
    status: ok
    cipher: none

    db (current)
        wal archive min/max (13): 000000140000598700000071/0000001400005A2F00000008

        full backup: 20230115-050005F
            timestamp start/stop: 2023-01-15 05:00:05 / 2023-01-15 05:39:04
            wal start/stop: 000000140000598700000071 / 0000001400005988000000B5
            database size: 681.6GB, database backup size: 681.6GB
            repo1: backup set size: 258.3GB, backup size: 258.3GB

        full backup: 20230118-050005F
            timestamp start/stop: 2023-01-18 05:00:05 / 2023-01-18 05:39:40
            wal start/stop: 0000001400005A0500000024 / 0000001400005A0600000045
            database size: 690.0GB, database backup size: 690.0GB
            repo1: backup set size: 260.7GB, backup size: 260.7GB

        diff backup: 20230118-050005F_20230119-050004D
            timestamp start/stop: 2023-01-19 05:00:04 / 2023-01-19 05:08:12
            wal start/stop: 0000001400005A2B000000D2 / 0000001400005A2C00000043
            database size: 692.5GB, database backup size: 143.6GB
            repo1: backup set size: 261.5GB, backup size: 40.0GB
            backup reference list: 20230118-050005F
```

5

Links

initMAX

Useful links

Sources of information

- https://planet.postgresql.org/
- https://commitfest.postgresql.org/
- https://www.zabbix.com/documentation/

Tools and Extensions

- https://www.timescale.com/
- https://www.postgresql.org/docs/current/pgstatstatements.html
- https://github.com/reorg/pg_repack
- https://github.com/citusdata/pg_cron
- https://pgbackrest.org/
- https://www.pgbouncer.org/
- https://github.com/dimitri/pgloader
- https://github.com/zalando/patroni
- https://github.com/cybertec-postgresql/vip-manager

6

Visit our booth.

www.initmax.cz

info@initmax.cz

Thank you for your patience