Postgres Security 101: Replication (7/8)

Oz · Following

3 min read · Oct 4, 2024

Replication plays a critical role in maintaining data availability, redundancy, and security within your PostgreSQL environment. In this seventh installment of the **Postgres Security 101** series, we will dive into replication best practices, configuration settings, and monitoring techniques to ensure a secure and reliable PostgreSQL replication setup. By the end of this article, you will understand how to set up a dedicated replication user, configure proper logging, and maintain backups and WAL archiving for optimal security.

7.1 Ensure a Replication-Only User Is Created and Used for Streaming Replication

For security purposes, it is essential to create a dedicated replication user with limited privileges. This ensures that replication processes do not have access to unnecessary features or data.

To create a replication-only user:

```
CREATE ROLE replication_user WITH REPLICATION PASSWORD 'test123';
```

This user is granted only the required replication permissions without unnecessary access to the database, helping to reduce potential attack vectors.

7.2 Ensure Logging of Replication Commands Is Configured

Logging replication-related commands and activities is essential for auditing and troubleshooting. Here's how to set up comprehensive logging to monitor replication activities:

```
log_autovacuum_min_duration: 5000
log_checkpoints: true
log_connections: true
logging_collector: true
log_directory: /pg_log/log
log_disconnections: true
log_filename: postgresql-%Y-%m-%d_%H%M%S.log
log_line_prefix: '%t [%p]: [%l-1] user=%u,db=%d,host=%h,app=%a '
log_rotation_age: 1d
log_lock_waits: true
log_min_duration_statement: 5000 # Logs statements with duration longer than 50
log_statement: ddl
log_temp_files: 0
```

By setting these parameters, you will ensure that any replication-related commands are logged, helping you detect potential issues early and maintain a robust auditing process.

7.3 Ensure Base Backups Are Configured and Functional

Base backups are a cornerstone of any disaster recovery strategy. Regularly performing and verifying base backups helps you quickly recover your data in case of failure or corruption. A reliable tool for this is `pgBackRest`, which simplifies backup and restore operations for both standalone and replication setups.

To set up a base backup:

- Install and configure `pgBackRest`.
- Test your backup process to ensure it functions as expected.
- Regularly verify your backups by performing restores on non-production environments.

For more information, you can read **Taking Backups Using pgBackRest on a Replication Server**.

7.4 Ensure WAL Archiving Is Configured and Functional

Write-Ahead Logging (WAL) archiving is essential for point-in-time recovery (PITR). It ensures that you can restore the database to a specific point in time by replaying WAL files. To set up WAL archiving:

```
archive_command: pgbackrest --stanza=cbs_backup archive-push %p
archive_mode: true
archive_timeout: 60
```

In this configuration:

- `archive_command` pushes archived WAL files to a backup directory using `pgBackRest`.
- `archive_mode` enables archiving.
- `archive_timeout` ensures that WAL files are archived regularly, even when there is low activity.

7.5 Ensure Streaming Replication Parameters Are Configured Correctly

Streaming replication must be configured properly to ensure data consistency between the primary and standby nodes. The following parameters should be set in your `postgresql.conf` to facilitate a secure and reliable replication process:

- `wal_level` : Set to `replica` to enable replication.
- `max_wal_senders` : Specify the number of connections allowed for replication.
- `wal_keep_size` : Set the size of WAL files to retain for replication purposes.
- `synchronous_commit` : Set to `on` for synchronous replication, ensuring the standby receives changes before committing them.

For more details on setting up streaming replication, you can refer to ***Setting Up Streaming Replication in PostgreSQL on RHEL 9***.

Conclusion

Proper replication configuration ensures that your PostgreSQL environment remains secure and reliable, even in the face of unexpected events. By following these guidelines — creating a replication-only user, configuring comprehensive logging, setting up reliable backups, maintaining WAL archiving, and ensuring correct streaming replication settings — you can protect your data and minimize the risk of disruptions. In the next and final article of this series, ***Postgres Security 101: Special Configuration Considerations (8/8)***, we'll cover advanced security

configurations to further enhance your PostgreSQL deployment. If you have any questions or need further assistance, feel free to reach out in the comments below and directly.

Database Security | Postgres Security | Security | Cybersecurity | Technology

Following

Written by Oz

149 Followers · 13 Following

Database Administrator 🐘

No responses yet

Gvadakte

What are your thoughts?

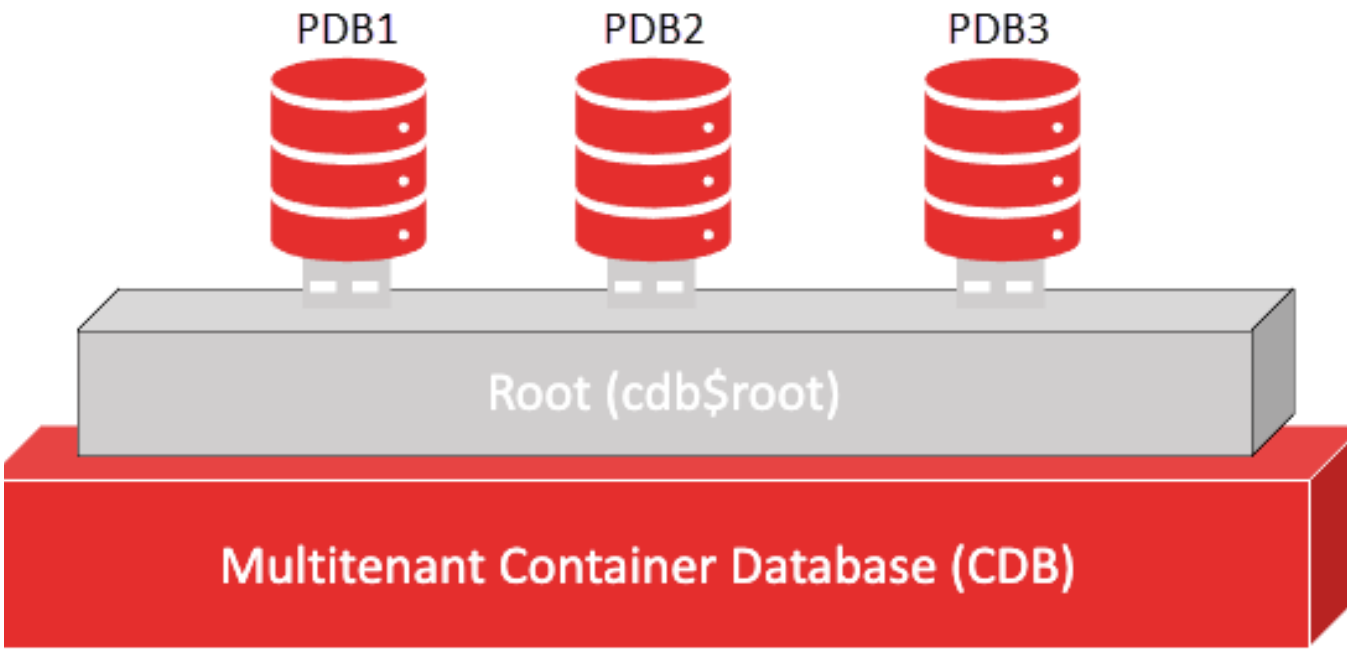
More from Oz

Oz

Installing Percona Monitoring & Management (PMM) with Postgres

Introduction:

Sep 26, 2024 54 1

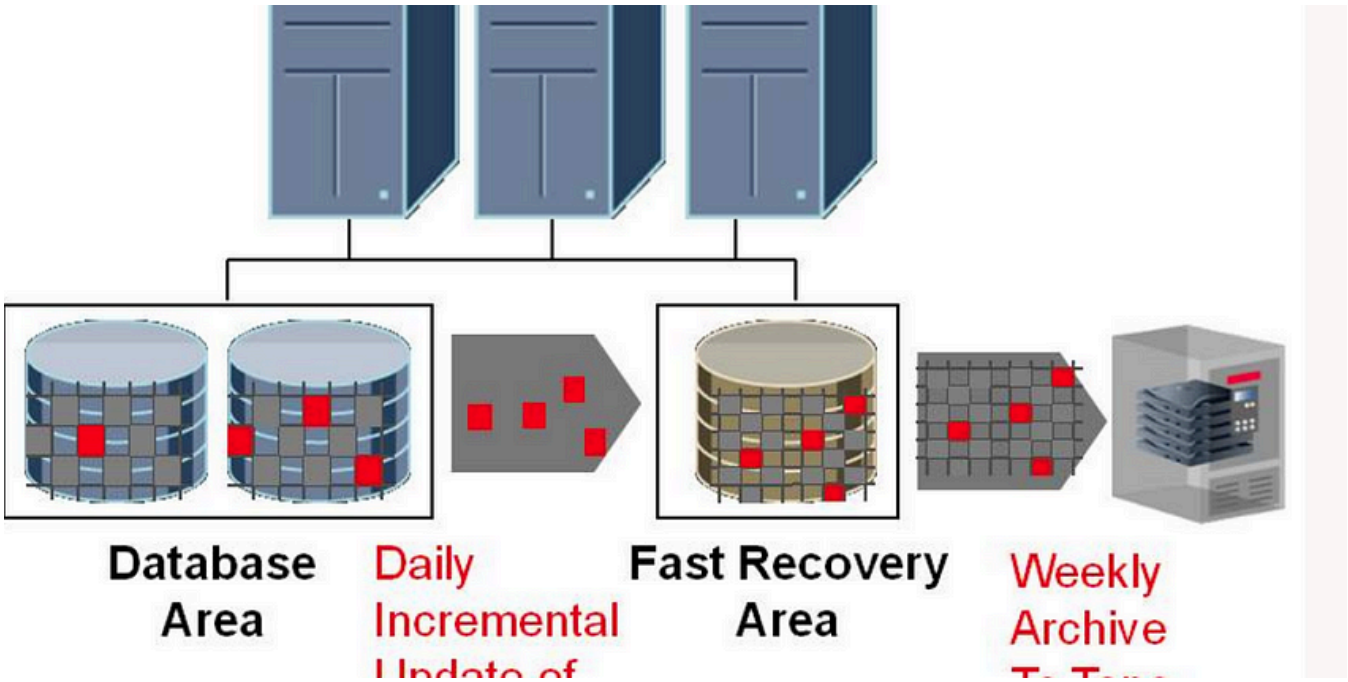


Oz

Pluggable Database Command

———————————— -———————————————— - create pluggable database pdb1 admin user root identified by test123; alter pluggable database...

May 12, 2023



Oz

RMAN Backup Basic Commands

rman target / rman target sys/password@YDKTST; backup database; backup database format ‘/backup/path/%d_%t_%s.rman’; backup tablespace...

May 11, 2023 1

Oz

delete jobs

May 8, 2023

See all from Oz

Recommended from Medium

Tihomir Manushev

Vector Search with pgvector in PostgreSQL

Simple AI-powered similarity search

Mar 9

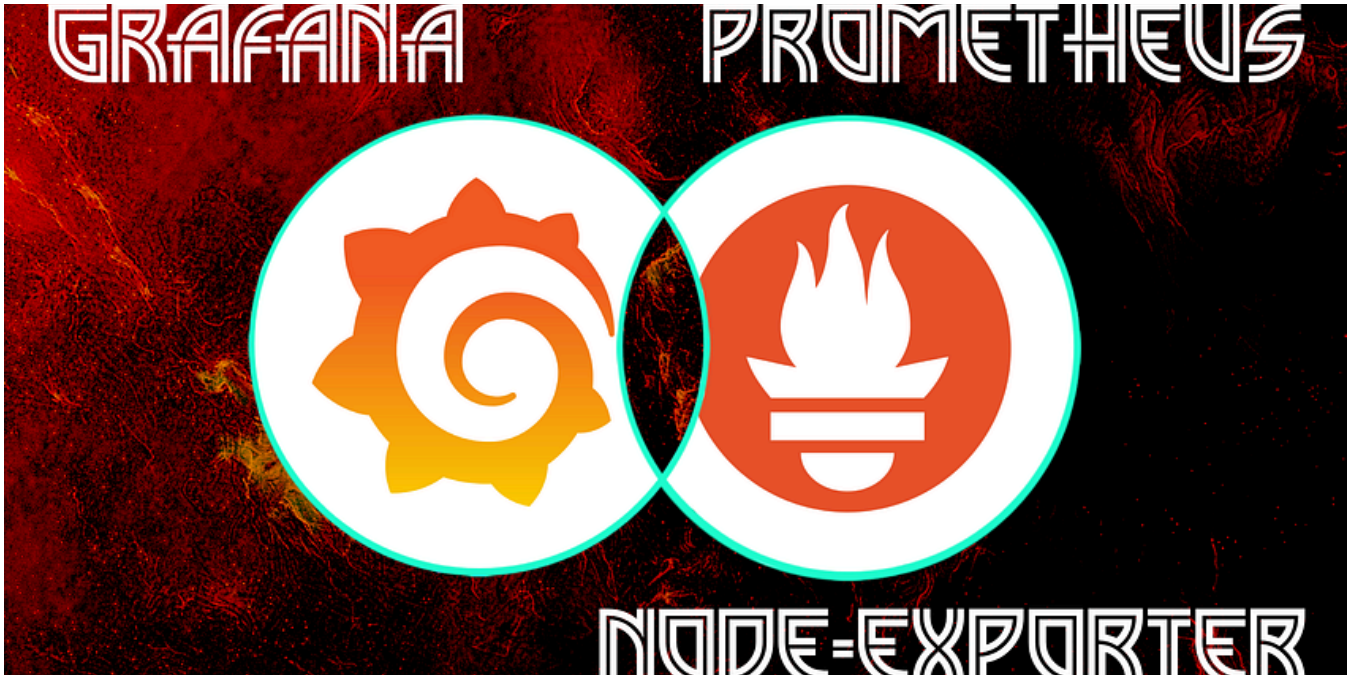
at it Means	Best Used For
e directly in the row	Simple data like INT
e in the row (unless large)	Larger types, but tri
npress + store out-of-row	Long texts, large ob
e out-of-row, no compression	When compression

Udbhav Singh

Storages in PostgreSQL

What Are Storage Types in PostgreSQL — And Why Should You Care?

6d ago 18

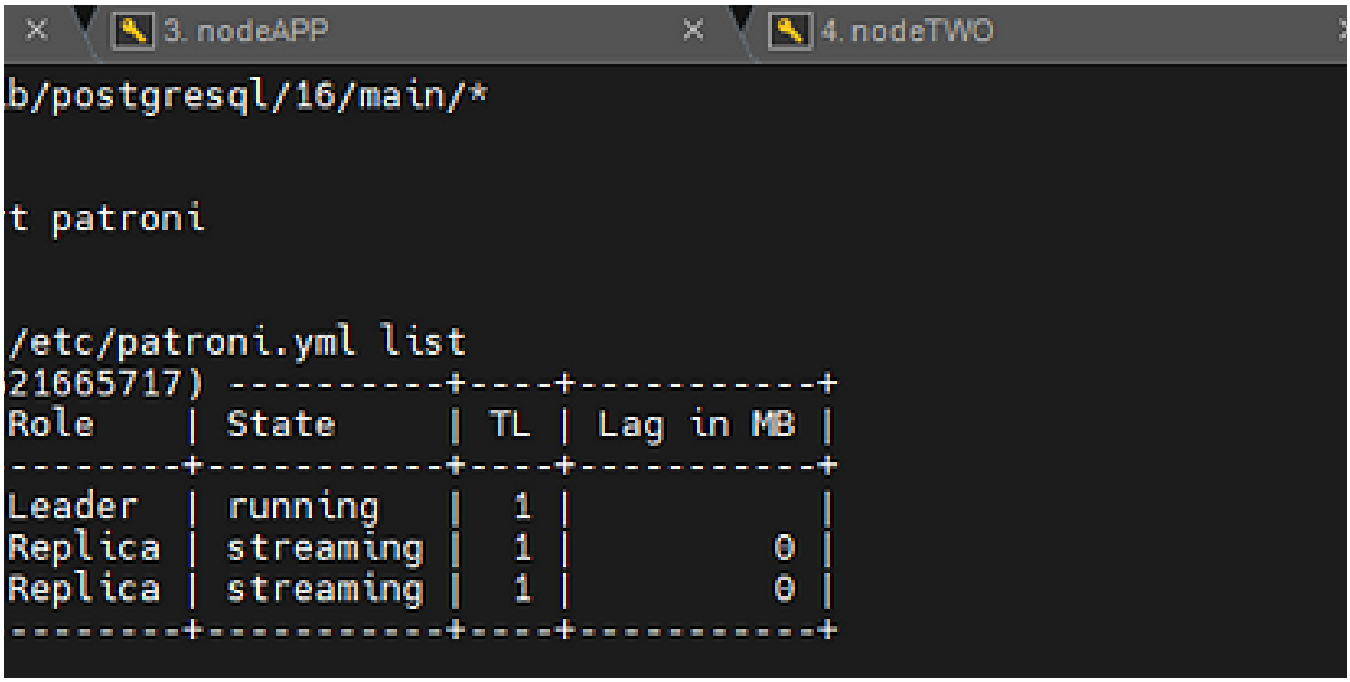


crptcpchk

Grafana, Prometheus & Node-Exporter | Setup Guide

Grafana open source software enables you to query, visualize, alert on, and explore your metrics, logs, and traces wherever they are stored...

Oct 30, 2024 8 1



Dickson Gathima

Building a Highly Available PostgreSQL Cluster with Patroni, etcd, and HAProxy

Achieving high availability in PostgreSQL requires the right combination of tools and architecture.

Mar 14 4

In Databases by Sergey Egorenkov

Making SQL query 40x faster for 10 million rows table

Make your SQL query really fast using this approach

Mar 17 8

In Towards Dev by Nakul Mitra

PostgreSQL Performance Optimization—Cleaning Dead Tuples & Reindexing

Performance optimization is crucial in PostgreSQL to ensure efficient query execution and minimal resource consumption.

Mar 28 1

See more recommendations