Member-only story

Postgres Security 101: Directory and File Permissions (2/8)

Oz · Following
4 min read · Oct 4, 2024

Listen Share More

One of the foundational aspects of securing PostgreSQL is ensuring that directory and file permissions are configured correctly. Misconfigured file permissions can expose sensitive database files, configurations, and logs to unauthorized users, putting the entire system at risk. In this article, we'll explore the importance of setting proper file and directory permissions, what critical files need attention, and best practices to minimize the chances of unauthorized access to your PostgreSQL environment.

2.1 Ensure the File Permissions Mask Is Correct

- Set the correct file permissions mask to prevent unauthorized access.


```
Currently it is set to 0022.
*/

su - postgres
unmask # check your defalt umask
#output
0022
echo 'umask 077' >> .bash_profile
source .bash_profile
umask
#output
0077

/*
        Common umask Settings
umask User Access Group Access Other
0000  all         all          all
0002  all         all          read, execute
0007  all         all          none
```

```
0022  all         read,execute read, execute
0027  all         read,execute none
0077  all         none         none
*/
```

2.2 Check Permissions of PGDATA

- Ensure the data directory permissions are set to restrict access appropriately. The data cluster Unix permissions must be 0700.

```
ls -ld /pg_data/data/

#output
drwx------ 19 postgres postgres 4096 May 30 00:00 /pg_data/data/

stat -c "%a" /pg_data/data/
#output
700
```

2.3 List Content of PGDATA to Check Unwanted Files and Symlinks

- Manually review the contents of the data directory for any unwanted files or symbolic links.

```
# The content of the PGDATA must be generated by PostgreSQL itself except custc
ls -ln /pg_data/data/

#Output
-rw-------  1 26 26   179 Mar 25 23:03 backup_label.old
drwx------ 63 26 26  4096 May 30 09:24 base
-rw-------  1 26 26    52 May 30 00:00 current_logfiles
drwx------  2 26 26  4096 May 30 11:31 global
drwx------  2 26 26    32 Feb  7 15:30 log
-rw-------  1 26 26  1896 May 27 15:53 patroni.dynamic.json
drwx------  2 26 26     6 Feb  7 15:30 pg_commit_ts
drwx------  2 26 26     6 Feb  7 15:30 pg_dynshmem
-rw-------  1 26 26  5996 May  6 14:51 pg_hba.conf
-rw-------  1 26 26  5996 May  7 11:33 pg_hba.conf.backup
-rw-------  1 26 26  1636 Mar 25 23:01 pg_ident.conf
-rw-------  1 26 26  1636 May  7 11:33 pg_ident.conf.backup
drwx------  4 26 26    84 May 30 14:58 pg_logical
drwx------  4 26 26    48 Feb  7 15:30 pg_multixact
drwx------  2 26 26     6 Feb  7 15:30 pg_notify
drwx------  4 26 26    50 May  8 13:43 pg_replslot
drwx------  2 26 26     6 Feb 12 14:42 pg_serial
```

```
drwx------  2 26 26     6 Feb 12 14:33 pg_snapshots
drwx------  2 26 26     6 May  7 11:33 pg_stat
drwx------  2 26 26  4096 May 30 15:14 pg_stat_tmp
drwx------  2 26 26    26 May 25 11:50 pg_subtrans
drwx------  2 26 26     6 Feb 27 15:42 pg_tblspc
drwx------  2 26 26     6 Feb 12 14:41 pg_twophase
-rw-------  1 26 26     3 Feb  7 15:30 PG_VERSION
lrwxrwxrwx  1 26 26     7 Feb  7 15:30 pg_wal -> /pg_wal
drwx------  2 26 26    26 Feb 12 14:37 pg_xact
-rw-------  1 26 26    88 Mar 25 23:03 postgresql.auto.conf
-rw-------  1 26 26 28098 Mar 25 23:03 postgresql.base.conf
-rw-------  1 26 26 28098 May  7 11:33 postgresql.base.conf.backup
-rw-r--r--  1 26 26  2475 May 27 15:53 postgresql.conf
-rw-r--r--  1 26 26  2475 May  7 11:33 postgresql.conf.backup
-rw-------  1 26 26   433 May  7 11:33 postmaster.opts
-rw-------  1 26 26    99 May  8 13:43 postmaster.pid
```

2.4 Check Permissions of pg_hba.conf

- Verify that the `pg_hba.conf` file permissions restrict access to authorized users only. The pg_hba.conf UNIX permission must be 0640 or 0600, especially when it is stored outside the PGDATA

```
# The pg_hba.conf UNIX permission must be 0640 or 0600, especially when it is s
ls -ld /pg_data/data/pg_hba.conf
#output
-rw------- 1 postgres postgres 5996 May  6 14:51 /pg_data/data/pg_hba.conf

stat -c "%a" /pg_data/data/pg_hba.conf
#output
600
```

2.5 Check Permissions on Unix Socket

- Ensure the Unix socket permissions are correctly set to secure local connections. The default permissions are 0777, meaning anyone can connect. Reasonable alternatives are 0770 (only user and group, see also unix_socket_group) and 0700 (only user).

```
psql -c 'SHOW unix_socket_directories;'
#output
unix_socket_directories
```

```
---------------------------
 /var/run/postgresql, /tmp

ls -ld /var/run/postgresql
#output
drwxr-xr-x 2 postgres postgres 80 May  7 11:33 /var/run/postgresql
stat -c "%a" /var/run/postgresql
#output
755
```

2.6 Disable PostgreSQL Command History

- On Linux/UNIX, the PostgreSQL client logs most interactive statements to a history file. The default PostgreSQL history file is named .psql_history in the user's home directory.

```
#Remove .psql_history if it exists.

rm -f ~<user>/.psql_history || true
#Use either of the techniques below to prevent it from being created again:
#Set the HISTFILE variable to /dev/null in ~<user>/.psqlrc
cat << EOF >> ~<user>/.psqlrc
\set HISTFILE /dev/null
EOF
#Create ~<user>/.psql_history as a symbolic to /dev/null.
ln -s /dev/null $HOME/.psql_history
#Set the PSQL_HISTORY variable for all users:
sudo echo 'PSQL_HISTORY=/dev/null' >> /etc/environment
```

Securing PostgreSQL is not just about internal settings; it also requires vigilant control over the file system where the database resides. By applying the correct directory and file permissions, you can significantly reduce the risk of unauthorized access to crucial data and configurations. This essential step is a key part of maintaining a secure and reliable PostgreSQL setup. To continue enhancing your database security knowledge, I recommend reading my next article: "**Postgres Security 101: Logging and Auditing (3/8)**", where we dive into PostgreSQL's logging features and how they play a critical role in monitoring and securing your database environment.For more detailed and technical articles like this, keep following our blog on Medium. 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?

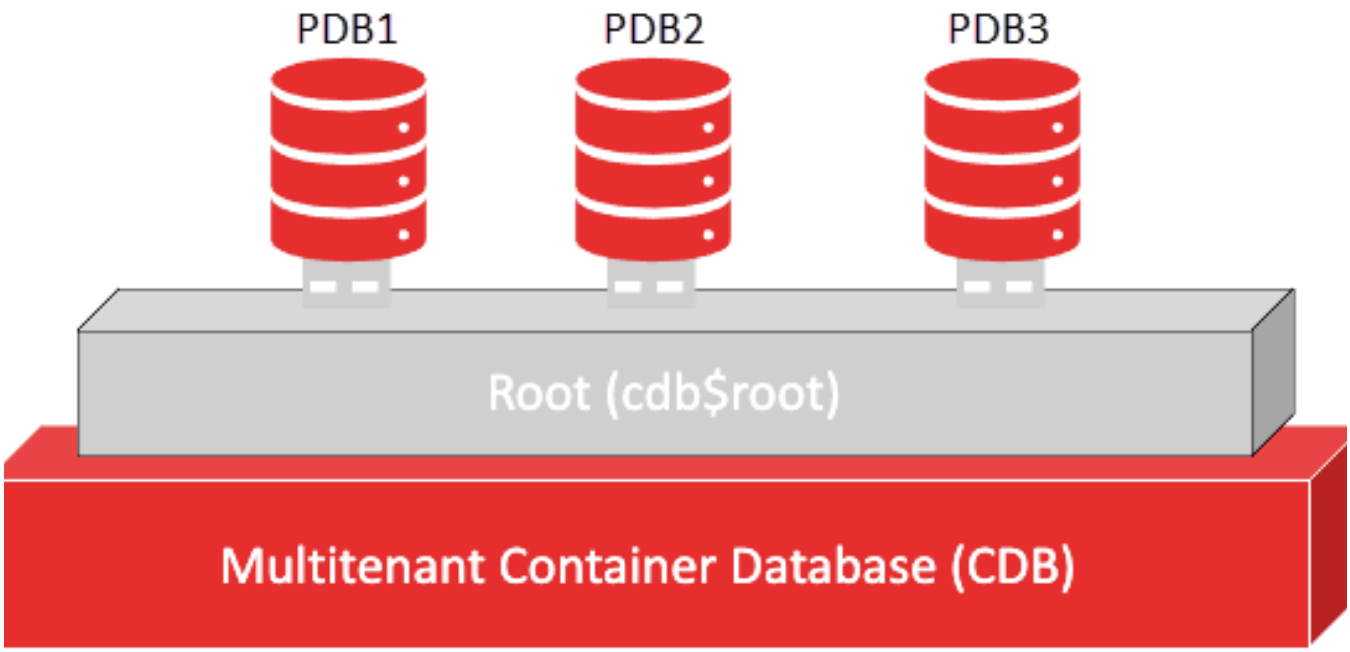
More from Oz

Oz

Installing Percona Monitoring & Management (PMM) with Postgres

Introduction:

Sep 26, 2024 54 1

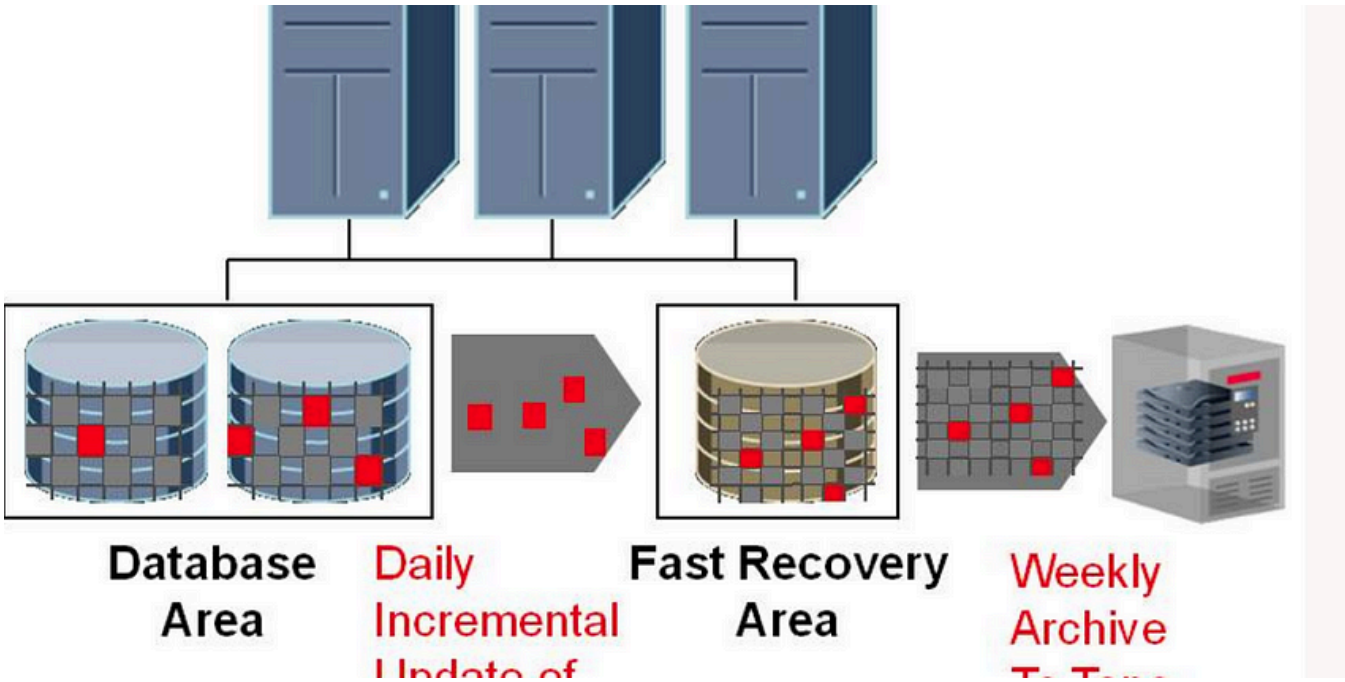


Oz

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

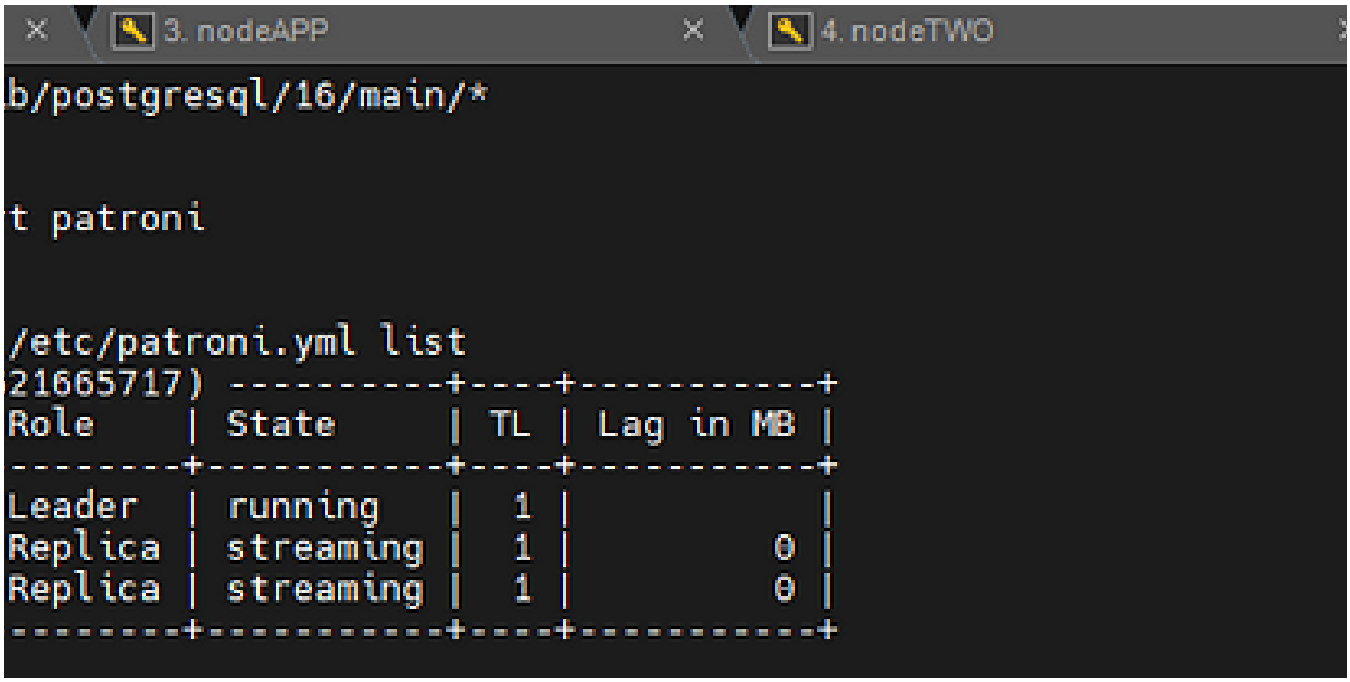
Recommended from Medium

Tihomir Manushev

Vector Search with pgvector in PostgreSQL

Simple AI-powered similarity search

Mar 9

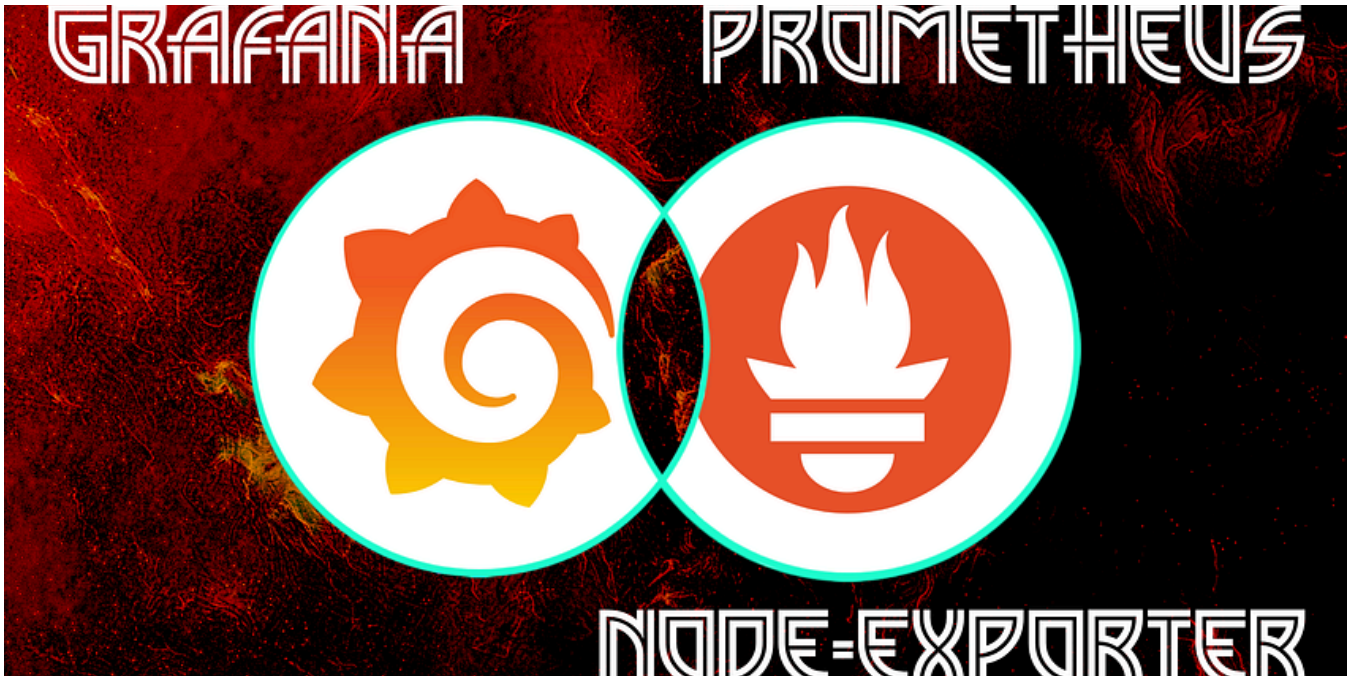


Dickson Gathima

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



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

Anubhav Bhardwaj

PostgreSQL Monitoring Script with Email Alerts

Overview

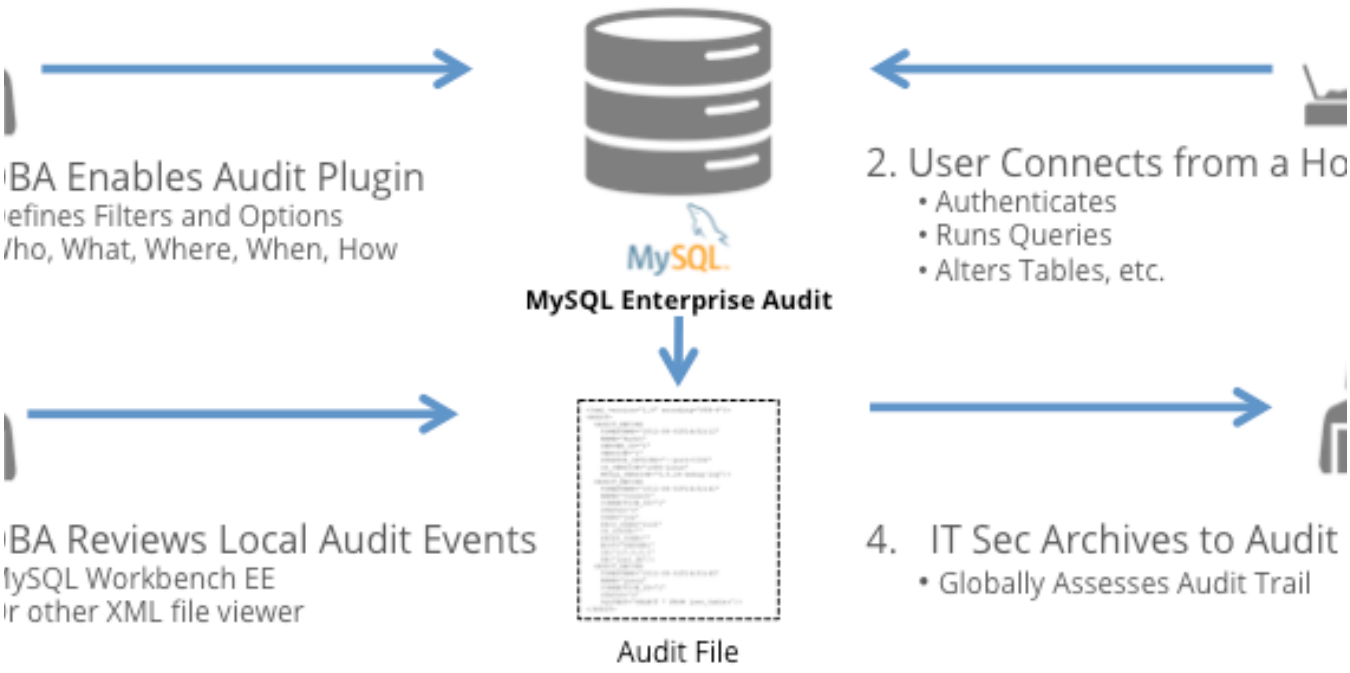
Jan 14 5

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



Hari

How to Enable MYSQL DB audit logs

Step 1: Install the MySQL Enterprise Audit Plugin

Nov 12, 2024 53

See more recommendations