PostgreSQL Security: Defending Against External Attacks

Taras Kloba

Microsoft

Taras Kloba
Senior Solutions Architect,
Data & AI,
Microsoft

- 15+ years in data engineering
- 9 years working deeply with PostgreSQL
- Co-Leader of PostgreSQL Ukraine Community
- Winner of Ukrainian IT Awards 2019 – Best Software Architect in Ukraine
- Two-time Winner of TIDE NATO Hackathon
- Ex-Microsoft MVP specialized in Azure Databases for PostgreSQL
- Microsoft Certified Trainer (MCT)
- Certified Cloud Architect & Data Engineer on Microsoft Azure, Google Cloud Platform, and Amazon Web Services
- Ph.D.
- Father of three daughters

AGENDA

1 Introduction

2 Overview of PostgreSQL Security Threats

3 Securing PostgreSQL Deployments from External Threats

4 Authentication Security

5 Data Protection

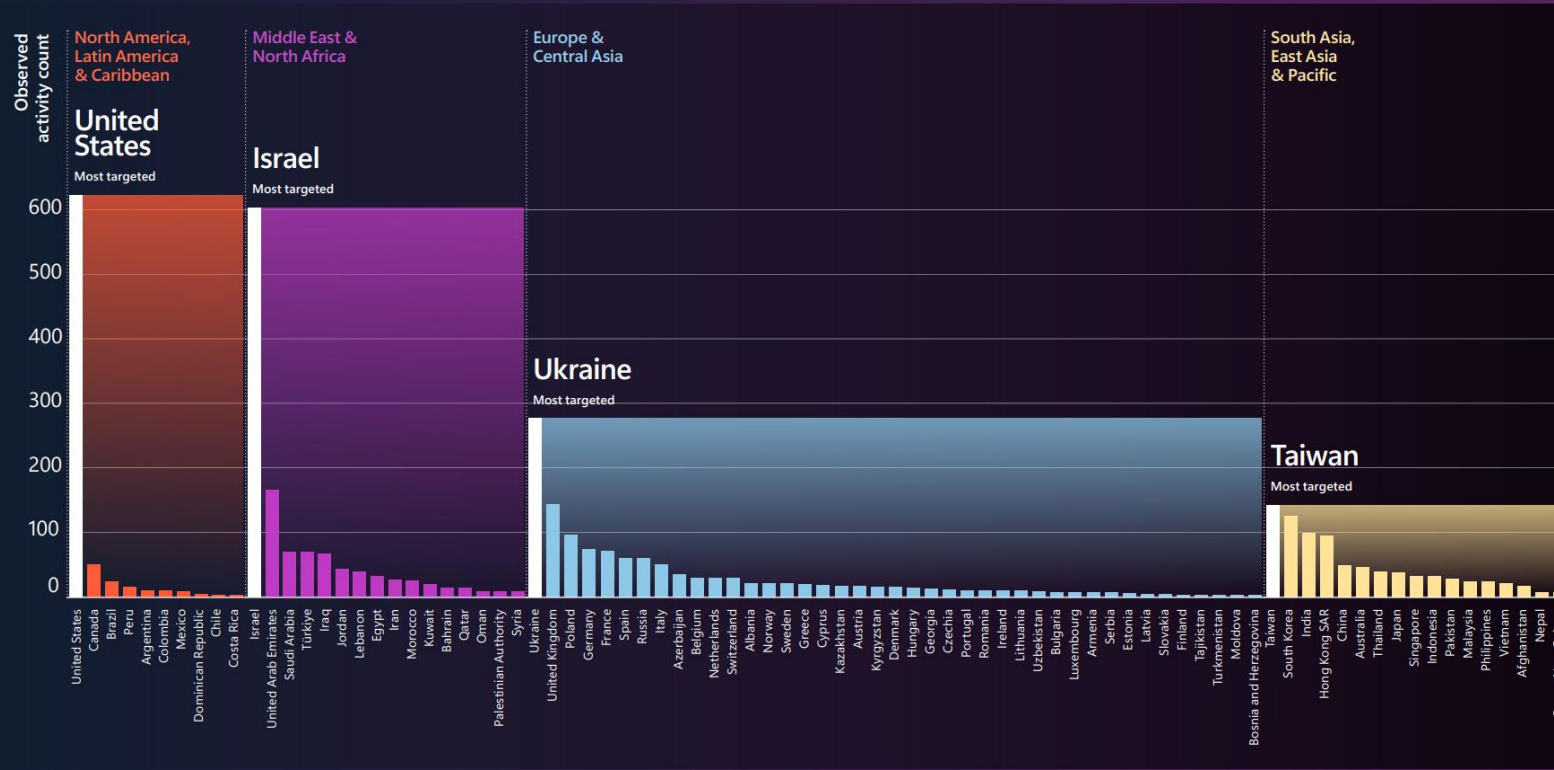
6 Privilege Escalation

7 Conclusion

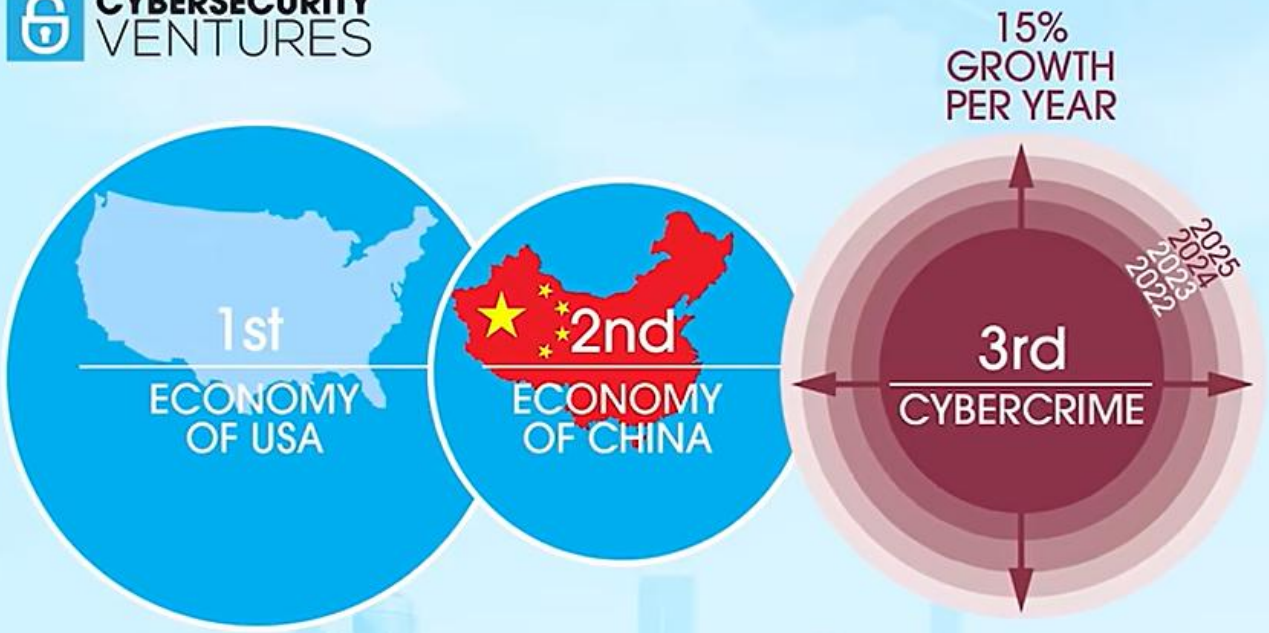
OVERVIEW OF POSTGRES SECURITY THREATS

Nation-state threat actor targeting

Regional sample of activity levels observed



CYBERSECURITY VENTURES



REAL-WORLD EXAMPLES OF SECURITY BREACHES

LINKEDIN DATA BREACH

CVE-2012-0866

Occurred in 2012 and exposed hashed passwords of **millions of users** due to attackers exploiting a vulnerability in a PostgreSQL-powered database.

THREE UK BREACHES

CVE-2013-1899

Occurred in 2016 and exposed the personal and billing information of **six million customers** due to attackers exploiting a known vulnerability in a PostgreSQL-powered customer database.

OPENTABLE DATA BREACH

CVE-2018-1058

Occurred in 2017 and affected over **800,000 customers** due to unauthorized access to a PostgreSQL database.

GDPR FINES: UP TO €20 MILLION OR 4% OF GLOBAL REVENUE

BBC

Home News US Election Sport Business Innovation Culture Arts Travel Earth Video Live

British Airways fined £20m over data breach

16 October 2020

Share Save

British Airways has been fined £20m ($26m) by the Information Commissioner's Office (ICO) for a data breach which affected more than 400,000 customers.

The breach took place in 2018 and affected both personal and credit card data.

BBC

Home News US Election Sport Business Innovation Culture Arts Travel Earth Video Live

Marriott Hotels fined £18.4m for data breach that hit millions

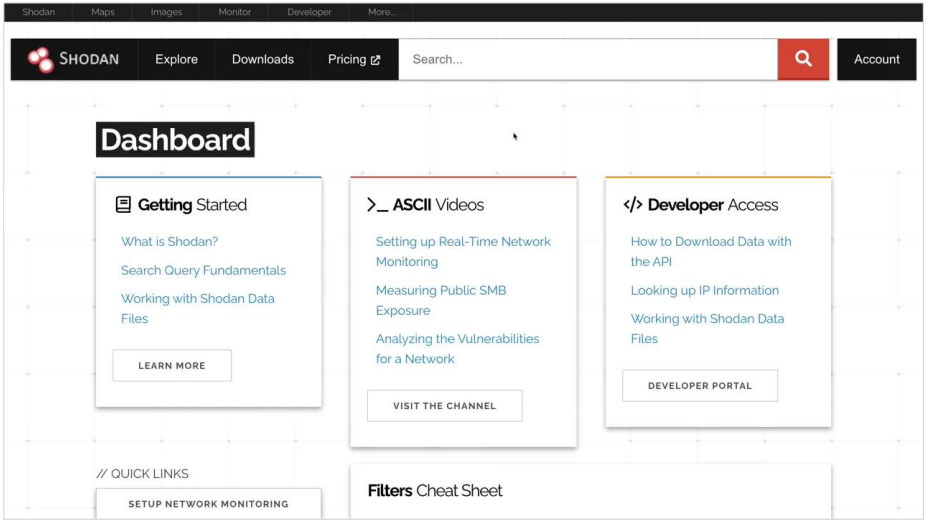
30 October 2020

Share Save

The UK's data privacy watchdog has fined the Marriott Hotels chain £18.4m for a major data breach that may have affected up to 339 million guests.

SECURING POSTGRES DEPLOYMENTS FROM EXTERNAL THREATS

IDENTIFYING PUBLICLY AVAILABLE POSTGRESQL DEPLOYMENTS



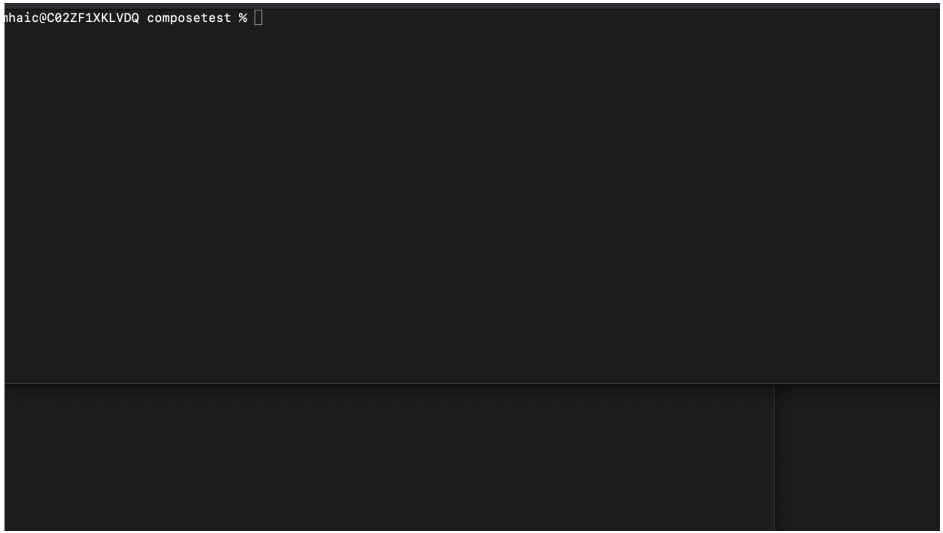
Such services as Shodan, Censys, BinaryEdge, and ShodanServer regularly scan the entire internet, making information about publicly available resources easily accessible for just a few dollars per month.

https://youtu.be/1Xwh6jVDFb0

LIMITING POSTGRESQL EXPOSURE TO THE INTERNET

Limit	Limit listen_addresses to safe networks
Restrict	Restrict access in pg_hba.conf
Utilize	Utilize a VPN or private network

PENETRATION TESTING POSTGRESQL DEPLOYMENTS USING HYDRA



Using a brute-force attack, any six-character password can be guessed in less than an hour. An attacker could also check up to five million the most common passwords within the same hour.

https://youtu.be/kR1wqUzO76w

THE DOMINO EFFECT OF A REUSED PASSWORD

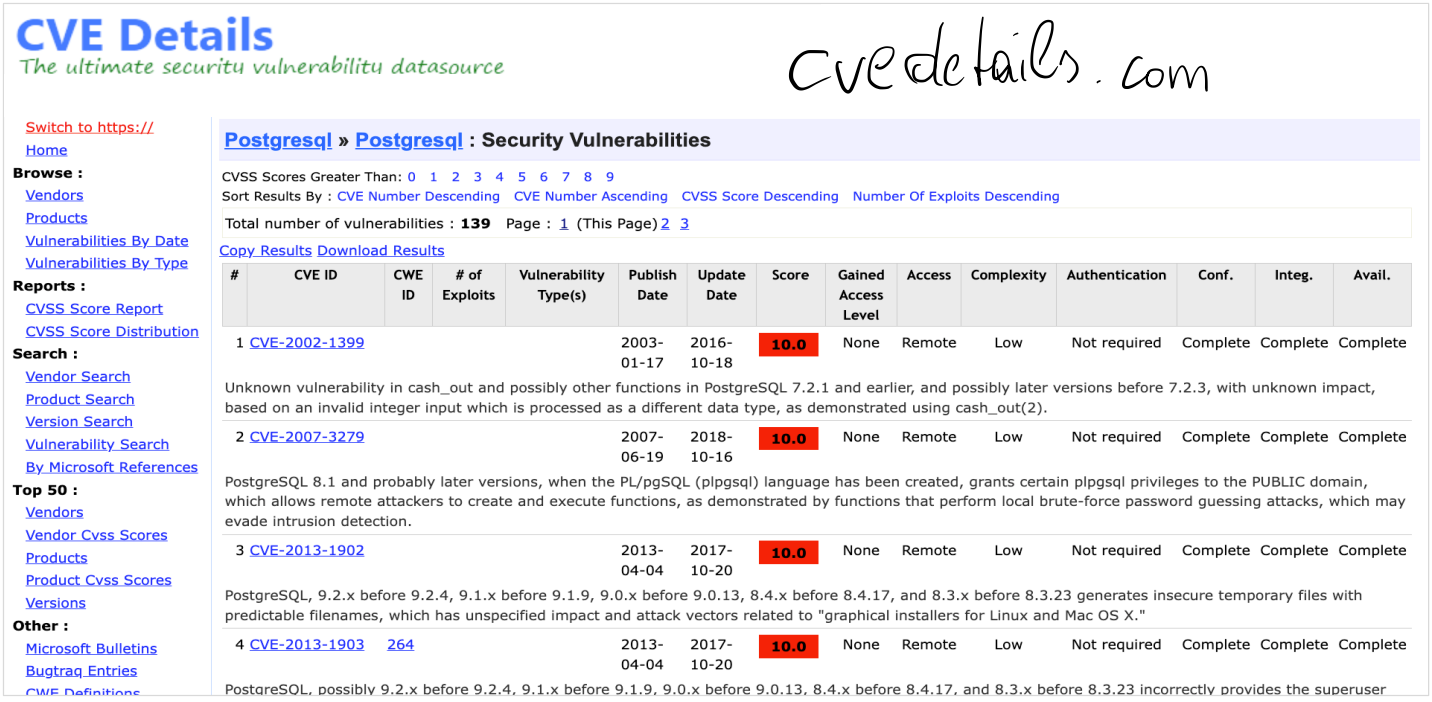
Check your email for breaches at
haveibeenpwned.com

14,170,129,036 pwned accounts

MITIGATION AGAINST BRUTE FORCE ATTACKS

Use	Use strong, long, unique, and random passwords
Integrate	Integrate with external authentication providers (e.g., Entra ID, Kerberos) to enforce password policies and enable 2FA
Enable	Enable the auth_delay extension to slow down brute-force attempts
Deploy	Deploy the Fail2Ban tool to block repeated failed login attempts

COMMON VULNERABILITIES AND EXPOSURES



CVE Details
The ultimate security vulnerability datasource

cvedetails.com

Switch to https://
Home
Browse :
Vendors
Products
Vulnerabilities By Date
Vulnerabilities By Type
Reports :
CVSS Score Report
CVSS Score Distribution
Search :
Vendor Search
Product Search
Version Search
Vulnerability Search
By Microsoft References
Top 50 :
Vendors
Vendor Cvss Scores
Products
Product Cvss Scores
Versions
Other :
Microsoft Bulletins
Bugtraq Entries
CWE Definitions

Postgresql » Postgresql : Security Vulnerabilities

CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9
Sort Results By : CVE Number Descending CVE Number Ascending CVSS Score Descending Number Of Exploits Descending

Total number of vulnerabilities : **139** Page : 1 (This Page) 2 3

Copy Results Download Results

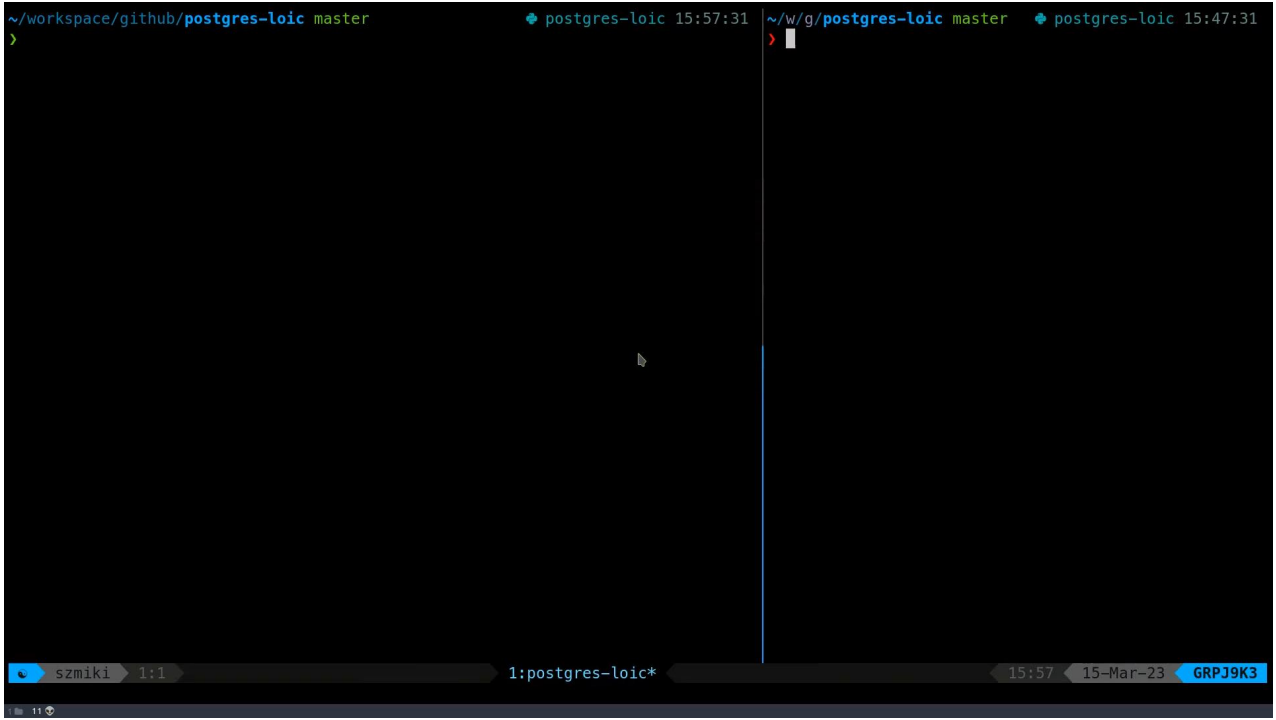
#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2002-1399				2003-01-17	2016-10-18	10.0	None	Remote	Low	Not required	Complete	Complete	Complete
	Unknown vulnerability in cash_out and possibly other functions in PostgreSQL 7.2.1 and earlier, and possibly later versions before 7.2.3, with unknown impact, based on an invalid integer input which is processed as a different data type, as demonstrated using cash_out(2).													
2	CVE-2007-3279				2007-06-19	2018-10-16	10.0	None	Remote	Low	Not required	Complete	Complete	Complete
	PostgreSQL 8.1 and probably later versions, when the PL/pgSQL (plpgsql) language has been created, grants certain plpgsql privileges to the PUBLIC domain, which allows remote attackers to create and execute functions, as demonstrated by functions that perform local brute-force password guessing attacks, which may evade intrusion detection.													
3	CVE-2013-1902				2013-04-04	2017-10-20	10.0	None	Remote	Low	Not required	Complete	Complete	Complete
	PostgreSQL, 9.2.x before 9.2.4, 9.1.x before 9.1.9, 9.0.x before 9.0.13, 8.4.x before 8.4.17, and 8.3.x before 8.3.23 generates insecure temporary files with predictable filenames, which has unspecified impact and attack vectors related to "graphical installers for Linux and Mac OS X."													
4	CVE-2013-1903	264			2013-04-04	2017-10-20	10.0	None	Remote	Low	Not required	Complete	Complete	Complete
	PostgreSQL, possibly 9.2.x before 9.2.4, 9.1.x before 9.1.9, 9.0.x before 9.0.13, 8.4.x before 8.4.17, and 8.3.x before 8.3.23 incorrectly provides the superuser													

Source: https://www.cvedetails.com/vulnerability-list.php?product_id=575

MITIGATION RECOMMENDATIONS FOR COMMON VULNERABILITIES

Update	Keep PostgreSQL updated with the latest patches.
Scan	Run security & vulnerability audits regularly.
Monitor	Watch logs for suspicious activity.
Disable	Turn off unused features to reduce risk.
Backup	Ensure encrypted backups and secure storage.

DDOS ATTACK



https://youtu.be/CP_eF8o7apw

DDOS ATTACK PREVENTION FOR POSTGRESQL

Pool	Use (for example) PgBouncer to manage connection pooling and limit the number of connections.
Proxy	Use Nginx as a reverse proxy with rate limiting for incoming connection attempts.
Monitor	Monitor server logs for potential attacks.
Update	Stay up-to-date on the latest security threats and trends.

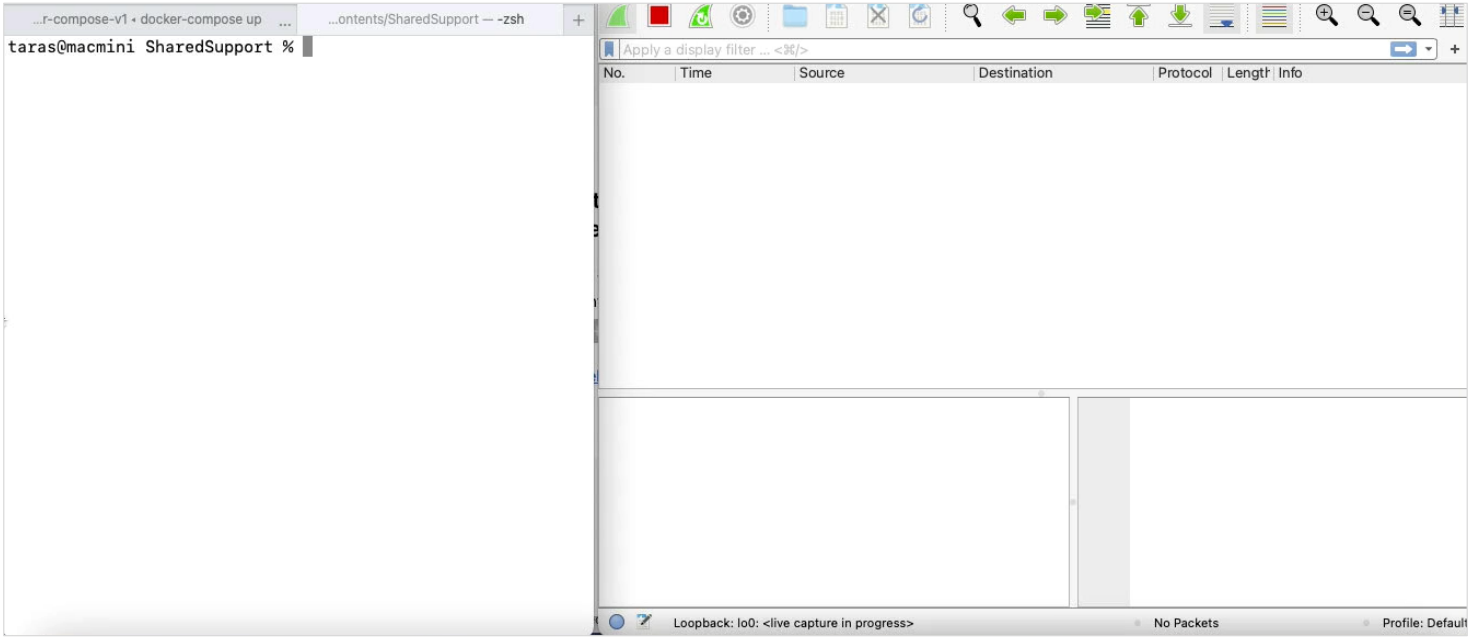
AUTHENTICATION SECURITY

"TRUST" AUTHENTICATION IN PG_HBA.CONF

```
# TYPE  DATABASE    USER        CIDR-ADDRESS          METHOD

# "local" is for Unix domain socket connections only
local   all         all                               trust
# IPv4 local connections:
host    all         all         127.0.0.1/32          trust
host    all         all         192.168.0.0/24        md5
host    all         all         68.145.52.251/32      md5
# IPv6 local connections:
-- INSERT --
```

"TRUST" AUTHENTICATION IN PG_HBA.CONF

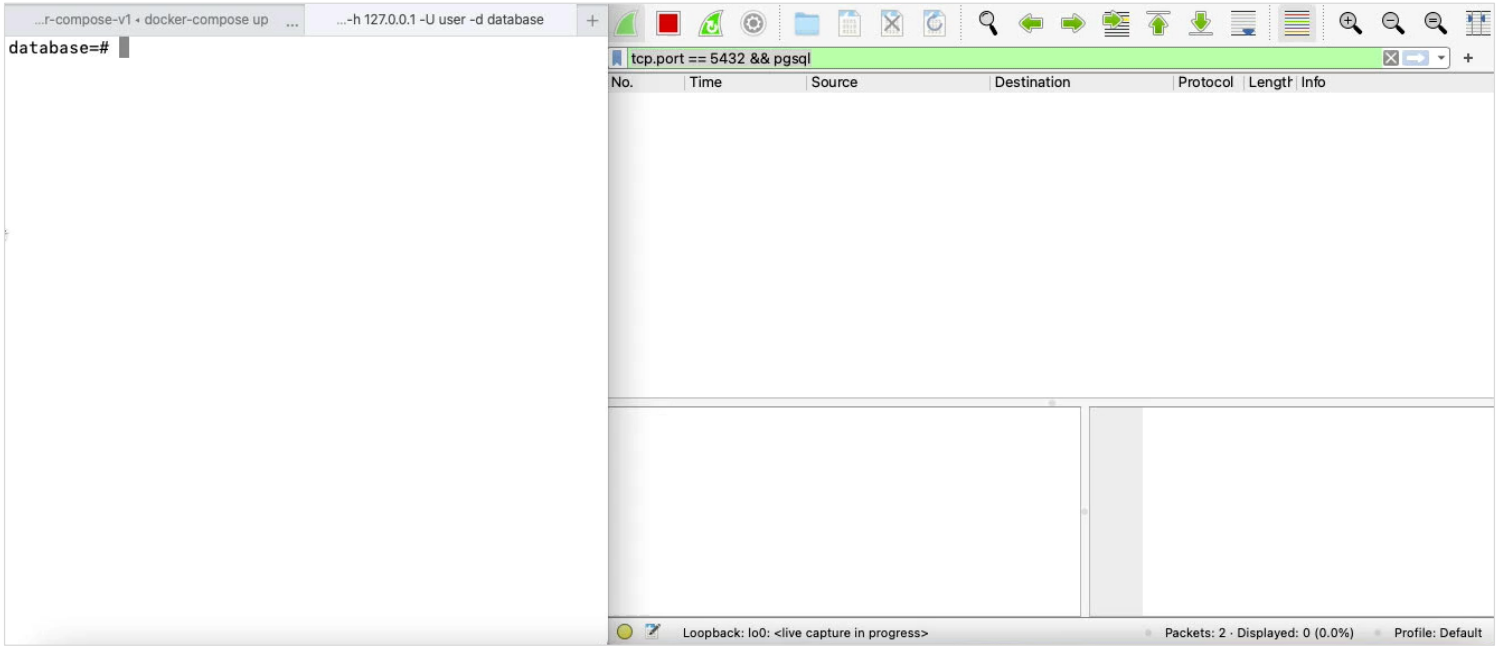


https://youtu.be/Dd_4bX3WJIQ

AVOID USING "TRUST" AUTHENTICATION IN PG_HBA.CONF

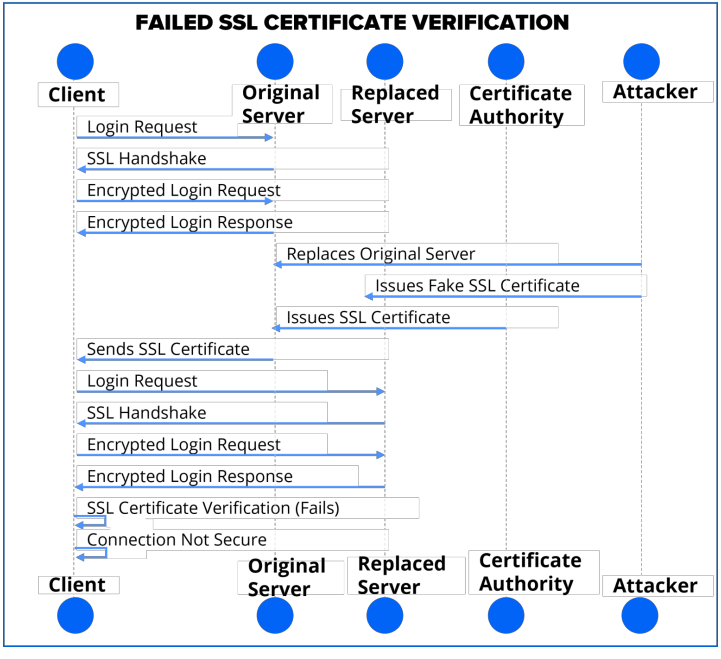
Avoid	Avoid using "trust" authentication in production environments.
Use	Use more secure authentication methods like Kerberos, LDAP, or Entra ID.
Audit	Regularly audit and review `pg_hba.conf` to ensure proper authentication settings are in place.

ENFORCING SSL CONNECTIONS IN POSTGRESQL: A NECESSITY, NOT A CHOICE

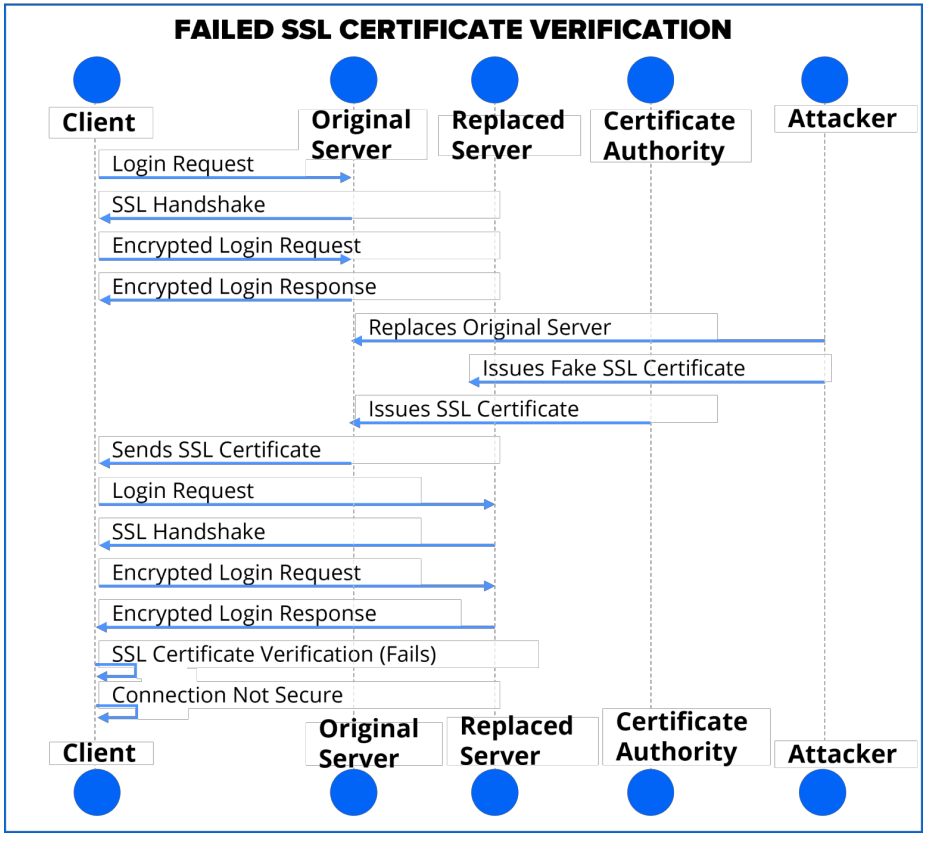


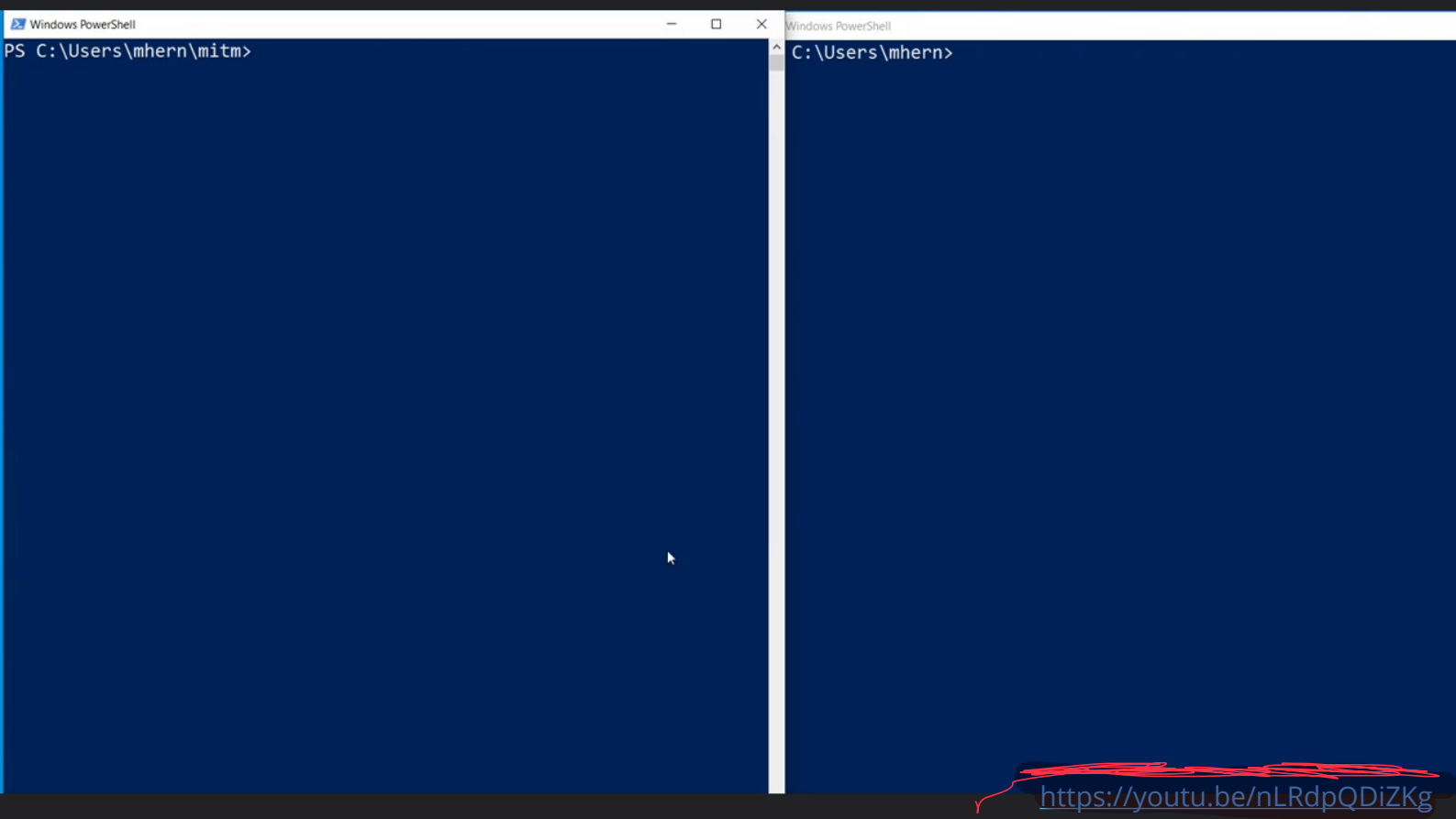
https://youtu.be/MzCBmZIQK7A

USING CERTIFICATES FOR HIGH CONFIDENCE CONNECTIONS IN POSTGRESQL



This diagram illustrates a man-in-the-middle attack where an attacker replaces the original server with a fake server and issues a fake SSL certificate to the client, causing a failed connection. It emphasizes the significance of SSL certificate verification in preventing such attacks and ensuring secure communication between clients and servers.


FAILED SSL CERTIFICATE VERIFICATION
Client
Original Server
Replaced Server
Certificate Authority
Attacker
Login Request
SSL Handshake
Encrypted Login Request
Encrypted Login Response
Replaces Original Server
Issues Fake SSL Certificate
Issues SSL Certificate
Sends SSL Certificate
Login Request
SSL Handshake
Encrypted Login Request
Encrypted Login Response
SSL Certificate Verification (Fails)
Connection Not Secure
Client
Original Server
Replaced Server
Certificate Authority
Attacker

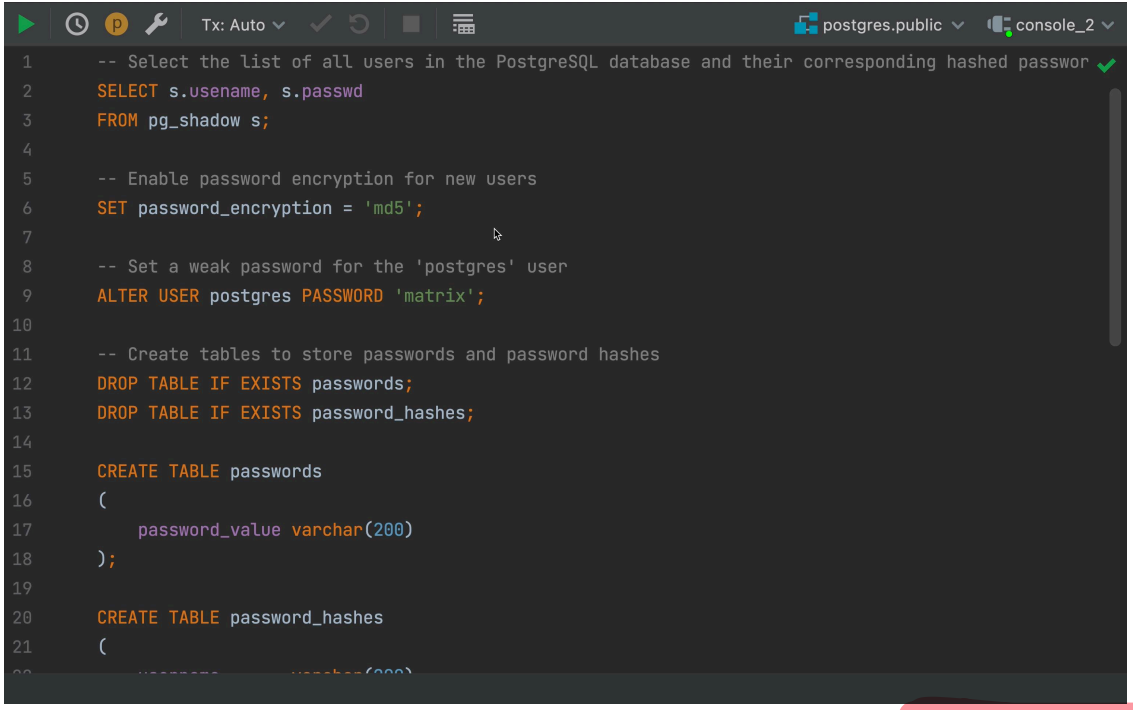


https://youtu.be/nLRdpQDiZKg

USING CERTIFICATES FOR HIGH CONFIDENCE CONNECTIONS

Use	Use client and server certificates for secure, authenticated communication.
Implement	Use a signed server certificate with the common name matching the server name.
Enforce	Set `sslmode=verify-full` on the client to prevent man-in-the-middle attacks.
Rotate	Regularly rotate certificates to maintain security over time.

REPLACING MD5 WITH SCRAM-SHA-256 FOR IMPROVED PASSWORD SECURITY



https://youtu.be/p7lfhC3qCgM

WHY STRONG PASSWORDS MATTER

Number of Characters	Numbers Only	Lowercase Letters	Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters, Symbols
4	Instantly	Instantly	Instantly	Instantly	Instantly
5	Instantly	Instantly	Instantly	Instantly	Instantly
6	Instantly	Instantly	Instantly	Instantly	Instantly
7	Instantly	Instantly	2 secs	7 secs	31 secs
8	Instantly	Instantly	2 mins	7 mins	39 mins
9	Instantly	10 secs	1 hour	7 hours	2 days
10	Instantly	4 mins	3 days	3 weeks	5 months
11	Instantly	2 hours	5 months	3 years	34 years
12	2 secs	2 days	24 years	200 years	3k years
13	19 secs	2 months	1k years	12k years	202k years
14	3 mins	4 years	64k years	750k years	16m years
15	32 mins	100 years	3m years	46m years	1bn years
16	5 hours	3k years	173m years	3bn years	92bn years
17	2 days	69k years	9bn years	179bn years	7tn years
18	3 weeks	2m years	467bn years	11tn years	438tn years

GPUs drastically speed up password cracking with tools like **Hashcat** and **gpuhash.me**, highlighting the need for strong, unique passwords.

https://www.reddit.com/r/Infographics/comments/tb59mh/oc_you_asked_for_it_the_animated_comparison_of/

USING CERTIFICATES FOR HIGH CONFIDENCE CONNECTIONS

Enable	Set `password_encryption = scram-sha-256` in `postgresql.conf` for better internal password hashing.
Migrate	Rehash existing passwords to upgrade them to SCRAM-SHA-256.
Integrate	Use external authentication providers like LDAP or Entra ID to enforce strong password policies and 2FA.
Warn	Even with SCRAM-SHA-256, weak passwords remain vulnerable to attacks, so enforcing password complexity is critical.

DATA PROTECTION

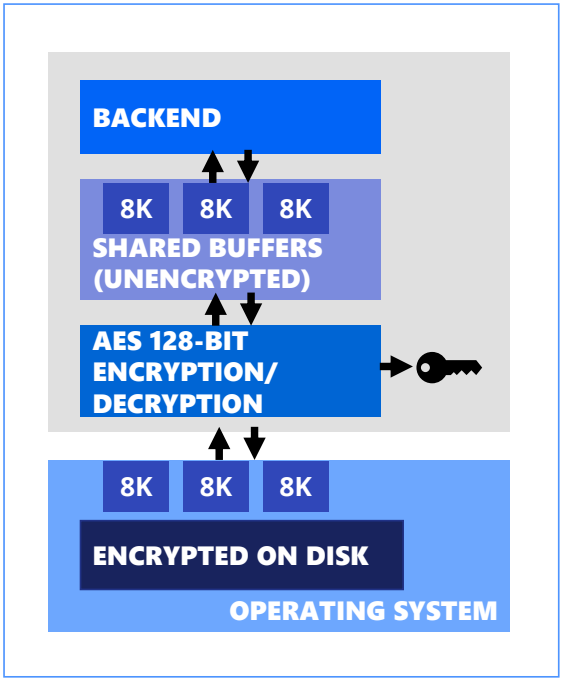
EXPLORING THE CONTENTS OF POSTGRESQL DATA FOLDER

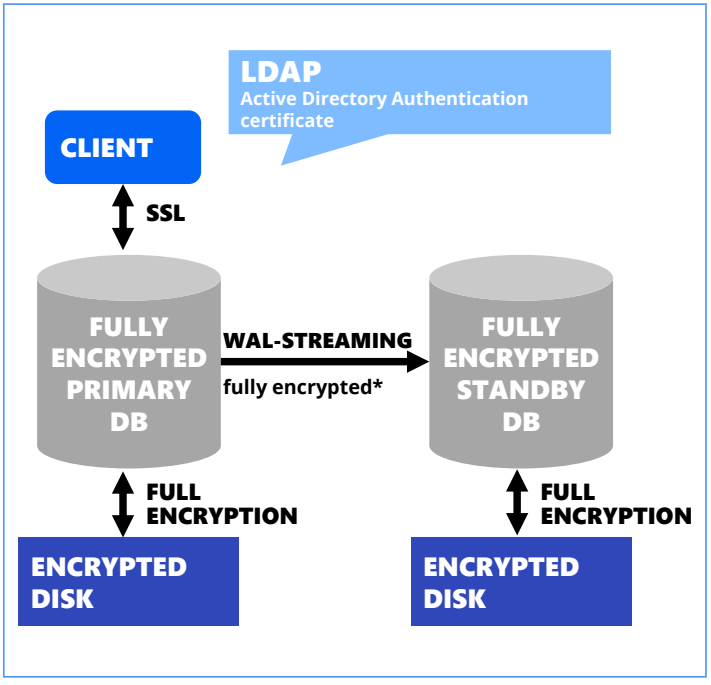
```
test=#
```

```
taras@macmini data %
```

https://youtu.be/OPL-eu4aWG0

POSTGRESQL TRANSPARENT DATA ENCRYPTION





Source: https://www.cybertec-postgresql.com/en/products/postgresql-transparent-data-encryption/

SECURING POSTGRESQL DATA AND BACKUPS

Encrypt	Encrypt backups and store them in a secure, offsite location.
Secure	Use strong and unique passwords for backup files to prevent unauthorized access.
Test	Test backups regularly to ensure their integrity and restore capability.
Control	Implement access controls to restrict who can access or manage backups.
Encrypt at Rest	Use PostgreSQL Transparent Data Encryption (TDE) to secure data at rest.
Protect Keys	Store encryption keys securely, separate from the backup files.
Monitor	Continuously monitor for unauthorized access to encryption keys and backup files.

PRIVILEGE ESCALATION

DEMONSTRATING PRIVILEGE ESCALATION RISKS

```
CREATE OR REPLACE FUNCTION GetUserID(user_name TEXT)
RETURNS INTEGER
LANGUAGE plpgsql
SECURITY DEFINER
AS $$
DECLARE
    user_id INTEGER;
BEGIN
    EXECUTE user_name;

    SELECT id INTO user_id
    FROM public.users
    WHERE username = user_name;

    RETURN user_id;
END;
$$;
```

Since it runs under **SECURITY DEFINER**, an attacker can inject malicious SQL, such as dropping tables, modifying data, or granting elevated privileges, all with the function owner's privileges.

```
SELECT GetUserID('DROP TABLE public.users;');
```

PRIVILEGE ESCALATION VIA COPY FROM PROGRAM

```
COPY (SELECT '') TO PROGRAM 'psql -c "ALTER ROLE attacker_user WITH SUPERUSER;"';
```

This command exploits the COPY FROM PROGRAM feature in PostgreSQL to execute a system command. It uses psql to alter the attacker_user role and grant it **superuser** privileges. The command runs on the server host, leveraging misconfigured trust-based authentication (or similar passwordless access), allowing an attacker to escalate privileges without proper authentication.

TRY TO PREVENT PRIVILEGE ESCALATION

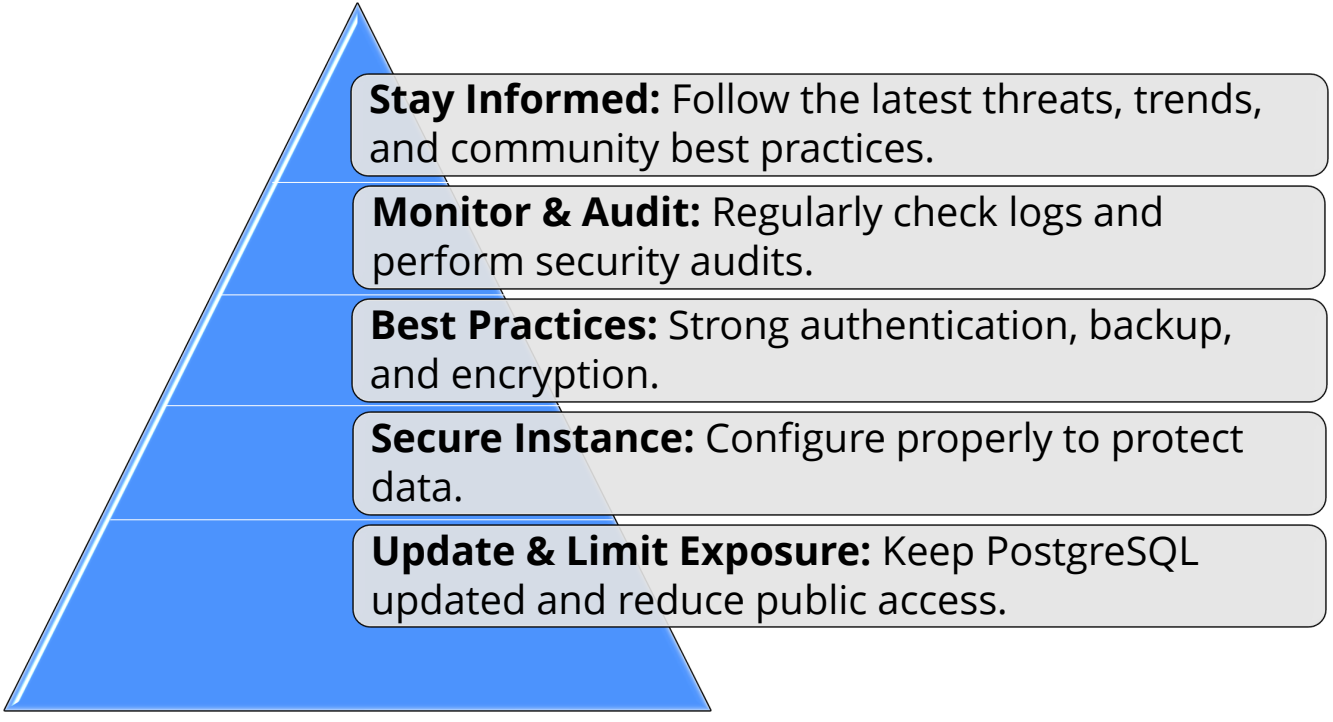
Restrict	Limit `SECURITY DEFINER` usage, validate input, and set `search_path` to a safe value (e.g., `pg_catalog`).
Disable	Allow only superusers to use `COPY FROM PROGRAM`.
Enforce	Avoid `trust` or `peer` in `pg_hba.conf`; use strong authentication like `md5` or `scram-sha-256`.

CONCLUSION

- don't expose DBs to the internet
- Vault instead of passwords on disk

DEFENDING YOUR POSTGRESQL DEPLOYMENT IS CRITICAL TO PREVENT LOSSES, AS CYBERCRIMINALS INCREASINGLY TARGET DATABASES AND EXPLOIT SOFTWARE VULNERABILITIES

SUMMARY


Stay Informed: Follow the latest threats, trends, and community best practices.
Monitor & Audit: Regularly check logs and perform security audits.
Best Practices: Strong authentication, backup, and encryption.
Secure Instance: Configure properly to protect data.
Update & Limit Exposure: Keep PostgreSQL updated and reduce public access.

CONNECT ON SOCIAL MEDIA

TARAS KLOBA
Senior Solutions Architect,
Data & AI,
Microsoft

aka.ms/Taras