pgAdmin4 in einem Ubuntu LXC-Container installieren

Einleitung

pgAdmin4 ist eine beliebte Open-Source-Webanwendung zur grafischen Verwaltung von PostgreSQL-Datenbanken. Sie bietet eine benutzerfreundliche Oberfläche zum Verwalten von Datenbanken, Abfragen von Daten, Anlegen von Tabellen und vielen weiteren administrativen Aufgaben.

Diese Anleitung beschreibt die Installation von pgAdmin4 in einem LXC-Container unter Ubuntu sowie die Anbindung eines PostgreSQL-Servers mit SSL-Verschlüsselung.

Voraussetzungen

- Ein LXC-Container mit Ubuntu 22.04 oder 24.04 (z. B. CTID 130, Hostname: `pgadmin`, IP: `192.168.137.130`)
- Internetzugang innerhalb des Containers
- Zugriff auf einen PostgreSQL-Server mit der IP `192.168.137.160`
- User `padl` in der Datenbank erstellt

Schritt 1: System aktualisieren

```
apt update && apt upgrade -y
```

Schritt 2: Notwendige Pakete installieren

```
apt install -y curl gnupg lsb-release wget ca-certificates
```

Warum ist dieser Schritt notwendig?

Diese Pakete sind auf einem frischen Ubuntu LXC-Container oft nicht vorinstalliert, werden aber für die folgenden Schritte zwingend benötigt:

Paket	Zweck
`curl`	Zum Herunterladen des GPG-Schlüssels für das pgAdmin4-Repository
`gnupg`	Um GPG-Schlüssel zu verarbeiten und Repositories zu verifizieren
`lsb-release`	Liefert z. B. die aktuelle Ubuntu-Version (`jammy`, `focal`) für den Repository-Eintrag
`wget`	Optionales Tool zum Herunterladen von Dateien – hilfreich bei Fehlersuche
`ca-certificates`	Stellt gültige Stammzertifikate bereit, damit HTTPS-Verbindungen (z. B. zu pgadmin.org) funktionieren

> **Hinweis** ⚠ Ohne diese Pakete könnten spätere Befehle fehlschlagen (z. B. „command not found“ oder „Repository kann nicht verifiziert werden“).

Man könnte prüfen, ob ein Paket schon installiert ist mit:

```
dpkg -l | grep <paketname>
# Beispiel:
dpkg -l | grep curl
```

Schritt 3: PostgreSQL-Repository und pgAdmin4-Schlüssel hinzufügen

```
curl https://www.pgadmin.org/static/packages_pgadmin_org.pub | gpg --dearmor -o
/usr/share/keyrings/pgadmin-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/pgadmin-keyring.gpg]
https://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/$(lsb_release -cs) pgadmin4
main" > /etc/apt/sources.list.d/pgadmin4.list

apt update
```

☑ Ziel von Schritt 3:

- Vertrauenswürdiger GPG-Schlüssel von pgAdmin4 wird importiert → damit apt die Quelle als sicher einstuft.
- Das pgAdmin4-Repository wird zu deiner sources.list.d hinzugefügt → damit apt install pgadmin4-web funktioniert.
- Anschließend wird ein apt update ausgeführt → damit apt die neuen Pakete findet.

🔍 Was passiert in den Befehlen?

```
curl https://www.pgadmin.org/static/packages_pgadmin_org.pub | gpg --dearmor -o
/usr/share/keyrings/pgadmin-keyring.gpg
```

- *Lädt den öffentlichen GPG-Schlüssel von der offiziellen pgAdmin-Website.*
- *Konvertiert ihn ins Format, das apt versteht, und speichert ihn sicher unter /usr/share/keyrings.*

```
echo "deb [signed-by=/usr/share/keyrings/pgadmin-keyring.gpg]
https://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/$(lsb_release -cs) pgadmin4
main" > /etc/apt/sources.list.d/pgadmin4.list
```

- *Erstellt eine neue Datei (pgadmin4.list) in /etc/apt/sources.list.d/*

- *Fügt das pgAdmin4-Repository hinzu – angepasst an die Ubuntu-Version ($(lsb_release -cs) → z. B. jammy)*

```
apt update
```

- *Lädt die Paketinformationen inklusive des neuen pgAdmin4-Repositories.*

Warum ist das notwendig?

Ohne diesen Schritt:

- ***Das System würde `pgAdmin4` nicht finden (da es nicht in den offiziellen Ubuntu-Repos enthalten ist).***
- ***Oder man würde eine Sicherheitswarnung erhalten, wenn man versuchst, ein nicht-verifiziertes Paket zu installieren.***

Schritt 4: pgAdmin4 Web-Version installieren

```
apt install -y pgadmin4-web
```

Ziel von Schritt 4:

- Installation der pgAdmin4 Webanwendung inkl. Webserver-Integration (Apache2)
- Vorbereitung für die Erstkonfiguration im nächsten Schritt

Was genau passiert?

Der Befehl `apt install -y pgadmin4-web`:

- Installiert pgAdmin4 (Web-Version)
- Installiert automatisch notwendige Abhängigkeiten wie:
  - Apache2 (als Webserver)
  - Python-Pakete (z. B. Flask, Gunicorn)
  - pgAdmin4-spezifische Dateien (z. B. Webinterface, statische Inhalte, Skripte)

Enthaltene Komponenten nach Installation:

Komponente	Funktion
`/usr/pgadmin4/`	Anwendungscode und Konfiguration
`/usr/pgadmin4/bin/setup-web.sh`	Script zur Initialkonfiguration der Web-Oberfläche
`apache2`	Webserver, auf dem die pgAdmin-Oberfläche gehostet wird

Voraussetzung:

Der vorherige Schritt (apt update + pgAdmin4-Repository) muss abgeschlossen sein, damit das Paket pgadmin4-web verfügbar ist.

Schritt 5: pgAdmin4 initial konfigurieren

```
/usr/pgadmin4/bin/setup-web.sh
```

Folgende Informationen werden abgefragt:

- E-Mail-Adresse für den Admin-Zugang (z. B. test@test.de)
- Passwort
- Webserver-Modus (wähle: Yes für Apache2-Integration)

☑ **Ziel von Schritt 5:**

- Erstellen eines Admin-Benutzers für die Weboberfläche
- Einrichten der Apache2-Webserver-Konfiguration
- Anlegen der nötigen Konfigurationsdateien und Pfade
- Aktivierung von pgAdmin4 als Webanwendung unter `/pgadmin4`

🔧 **Befehl:**

```
/usr/pgadmin4/bin/setup-web.sh
```

🔍 **Was genau passiert beim Ausführen?**

- Abfrage von Benutzerinformationen:
  - Admin-E-Mail-Adresse
  - Admin-Passwort
  - → Diese Zugangsdaten werden in einer SQLite-Datenbank gespeichert und sind notwendig zum Einloggen in die Weboberfläche.
- Erstellen der Konfiguration:
  - `config_local.py` wird generiert mit benutzerspezifischen Einstellungen
  - pgadmin4.db (Benutzerdatenbank) wird angelegt
- Apache2-Integration:
  - Apache2 wird automatisch konfiguriert (via WSGI-Modul)
  - Eine neue Site-Konfiguration für `/pgadmin4` wird aktiviert
  - Apache2 wird neu gestartet, damit die Weboberfläche sofort funktioniert

🌐 **Ergebnis:**

Nach erfolgreichem Setup ist pgAdmin4 erreichbar unter:

```
http://<container-ip>/pgadmin4
```

Beispiel:

```
http://192.168.137.130/pgadmin4
```

Dateien, die erstellt/angepasst werden:

Pfad	Beschreibung
`/var/lib/pgadmin/pgadmin4.db`	SQLite-Datenbank mit Benutzerkonten
`/var/lib/pgadmin/`	Arbeitsverzeichnis für pgAdmin
`/etc/apache2/conf-enabled/pgadmin4.conf`	Apache-Konfiguration für die Web-Oberfläche
`/usr/pgadmin4/web/config_local.py`	Lokale Konfiguration (z. B. Mailserver, Pfade, Zugriff)

Schritt 6: Firewall & Zugriff testen

Sicherstellen, dass der LXC-Container über den Browser erreichbar ist (z. B. `http://192.168.137.130/pgadmin4`). Falls UFW aktiviert ist:

```
ufw allow 80  //für HTTP Protokoll
ufw allow 443 //für HTTPS Protokoll
```

Schritt 7: PostgreSQL-Server mit SSL-Verbindung in pgAdmin4 einbinden

Nach erfolgreicher Einrichtung der SSL-Verschlüsselung auf dem PostgreSQL-Server (siehe Dokumentation [[2600 postgresql.md]]) kann der verschlüsselte Zugriff über pgAdmin4 erfolgen.

Serververbindung in pgAdmin4 anlegen

1. Öffne die Weboberfläche von pgAdmin4 im Browser:

   `http://192.168.137.130/pgadmin4`

2. Melde dich mit dem zuvor erstellten Admin-Konto an.

3. Klicke im linken Bereich mit der rechten Maustaste auf **"Servers"** und wähle **"Create"** → **"Server..."**

Reiter **General**

Feld	Eingabe

Feld	Eingabe
Name	`PostgreSQL-SSL` (beliebig wählbar)

Reiter **Connection**

Feld	Eingabe
Host name/address	`192.168.137.160`
Port	`5432`
Maintenance database	`postgres` (Standard)
Username	z. B. `pdal`
Password	[Passwort eintragen]
Save Password?	☑ aktivieren

> ⚠ Hinweis: Damit die Verbindung funktioniert, muss der PostgreSQL-Server die IP des pgAdmin4-Containers zulassen (siehe `pg_hba.conf`) und auf SSL konfiguriert sein (`postgresql.conf`).

Reiter **SSL**

Feld	Eingabe
SSL mode	`require`, `verify-ca` oder `verify-full` (je nach Sicherheitsanforderung und Serverkonfiguration)
Root Certificate	Nur bei `verify-ca` oder `verify-full`: z. B. `/usr/local/share/ca-certificates/postgres.crt`
Client Certificate	*optional*, nur wenn mTLS (Client-Zertifikate) aktiv
Client Certificate Key	*optional*

> ⚠ Wenn kein Root-Zertifikat angegeben wird, kann `require` verwendet werden, um eine verschlüsselte, aber nicht verifizierte Verbindung herzustellen.

Verbindung testen und speichern

- Klicke auf **"Save"**
- Die Verbindung erscheint nun links unter „Servers“.
- Bei erfolgreicher Konfiguration wird die Datenbankstruktur geladen.

⚠ Hinweis

Wenn die Verbindung fehlschlägt:

- **Fehlermeldung prüfen**: z. B. Zertifikatsfehler oder „Falsche Authentifizierung“.
- **SSL Mode testweise auf `disable` setzen**, um zu prüfen, ob das Problem SSL-bedingt ist.
- **Logdateien auf dem PostgreSQL-Server einsehen** (`/var/log/postgresql/`), um Verbindungsversuche und Fehler zu analysieren.

Die Oberfläche von pgAdmin4

Auf der linken Seite findet man unter dem vergebenen Namen "Datenbanken". Hier sollte die schon erstellte Datenbank erscheinen. Unter "Schemas" -> "Tabellen" sollte die Beispielsdatenbank sichtbar sein. Hier kann man sehr komfortabel Datenbanken oder Tabellen erstellen. Auch das Testen von Query`s ist hier sehr einfach.

Unter "Anmeldungs-/Gruppenrollen" sollten die von uns erstellten Rollen sichtbar sein. Man kann hier auch weitere Rollen erstellen.

Oben rechts unter, der Email-Adresse, findet man die Benutzerverwaltung. Hier sollte man sich einen Admin-User für pgAdmin4 erstellen. Alle weiteren User können auch hier erstellt werden.

Quellen

- „pgAdmin4 — pgAdmin4 9.5 documentation“. Zugegriffen: 22. Juli 2025. [Online]. Verfügbar unter: pgadmin4 Doc
- „18.9. Secure TCP/IP Connections with SSL“, PostgreSQL Documentation. Zugegriffen: 22. Juli 2025. [Online]. Verfügbar unter: PostgreSQL Doc

---

Lizenz