

Guide d'installation d'un serveur Debian 12

Galat Nathanaël

Table des matières

1. Installation de Debian 12.....	3
1.1. Préparation à l'installation.....	3
1.2. Installation.....	3
1.3. Déplacement de l'image disque.....	5
1.4. Vérification du serveur Debian.....	6
2. Installation Apache et PostgreSQL.....	7
2.1. Installation Apache.....	7
2.2. Installation PostgreSQL.....	8
2.3. Utilisation de PostgreSQL sur la machine virtuelle.....	9
2.4. Utilisation de PostgreSQL sur la machine Linux.....	10
2.5. Vérification de la base de données.....	11
3. Installation PHP et PhpPgAdmin.....	12
3.1. Installation PHP.....	12
3.2. Installation PhpPgAdmin.....	13
4. Sécurité.....	14
5. Résumé des commandes.....	15
6. Installation d'une machine virtuelle sur une clé USB (Annexe).....	16

1. Installation de Debian 12

1.1. Préparation à l'installation

À noter : les commandes avec un # devant signifient qu'elles doivent être effectuées avec l'utilisateur root.

Pour commencer il faut télécharger le fichier ISO de Debian 12. Cependant, pour gagner du temps et économiser de l'espace disque, le fichier ISO a déjà été téléchargé et mis à disposition.

Pour vérifier son intégrité, il faut vous rendre sur cette page web :

<https://cdimage.debian.org/cdimage/release/current/amd64/iso-cd/SHA512SUMS>

Puis exécuter la commande suivante :

« sha512sum debian-12.10.0-amd64-netinst.iso »

Une fois l'ISO vérifié (en comparant les deux codes du fichier: debian-12.10.0-amd64-netinst.iso), vous pouvez passer à l'installation.

1.2. Installation

Maintenant que vous avez vérifié l'ISO, vous pouvez lancer l'installation de Debian 12 grâce à la commande « S2.03-lance-installation ».

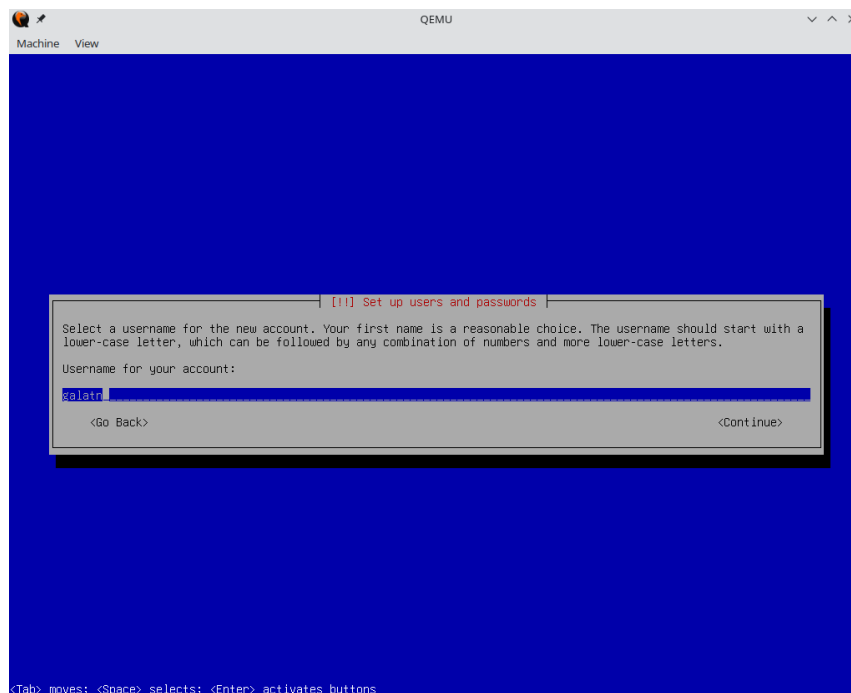
Dans le script «S2.03-commun», il y a les différents paramètres (options) permettant d'installer Debian 12. Je vais rapidement vous les expliquer ici :

- ➔ **qemu-system-x86_64** : Lance QEMU pour une architecture x86_64 (64 bits).
- ➔ **-machine q35** : Utilise le chipset Q35.
- ➔ **-cpu host** : Utilise le même modèle de CPU que l'hôte.
- ➔ **-m 4G** : Alloue **4 Go de RAM** à la machine virtuelle.
- ➔ **-enable-kvm** : Active l'accélération matérielle avec **KVM**.
- ➔ **-device VGA,xres=1024,yres=768** : Définit une carte graphique **VGA** avec une résolution de **1024x768**.
- ➔ **-display gtk,zoom-to-fit=off** : Affichage avec GTK de la VM sans zoom automatique.
- ➔ **-drive \$drive** : Spécifie un fichier image disque à utiliser.
- ➔ **-device e1000,netdev=net0** : Ajoute une carte réseau virtuelle **Intel e1000** attachée à l'interface **net0**.

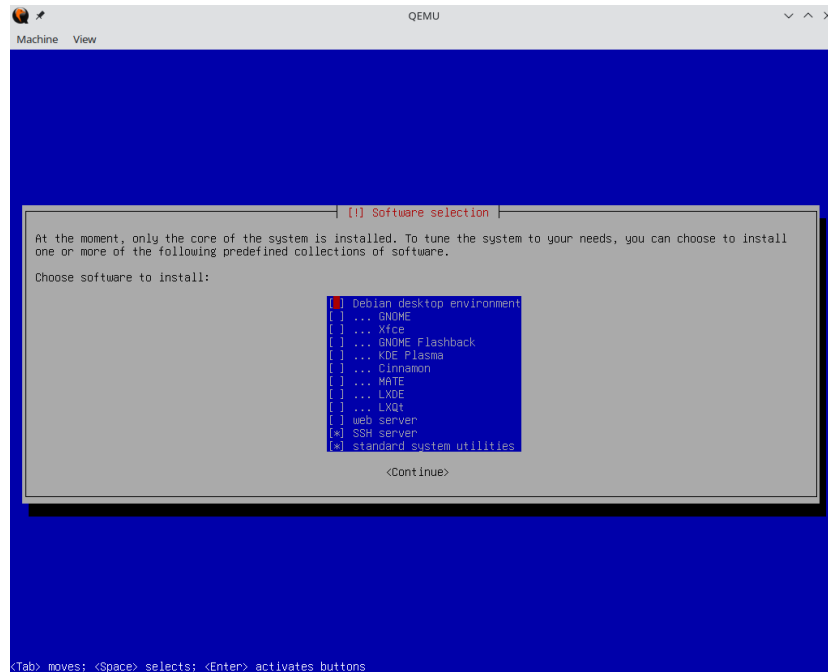
- ➔ **-netdev user,id=net0** : Active le mode utilisateur pour le réseau.
- ➔ **hostfwd=tcp::2222-:22** : Redirige le port 2222 de l'hôte vers le port SSH (22) de la VM.
- ➔ **hostfwd=tcp::4443-:443** : Redirige le port 4443 de l'hôte vers le port HTTPS (443) de la VM.
- ➔ **hostfwd=tcp::8080-:80** : Redirige le port 8080 de l'hôte vers le port HTTP (80) de la VM.
- ➔ **hostfwd=tcp::5432-:5432** : Redirige le port 5432 de l'hôte vers le port PostgreSQL (5432) de la VM.

Maintenant, vous pouvez passer à l'**installation**. Attention ! Certains passages de l'installation sont à modifier :

- **Location** : other/Europe/France
- **Keyboard** : French
- **Hostname** : utiliser server-"VOTRE_LOGIN_UGA"
- **Root Password** : un mot de passe simple est conseillé, par exemple "root".
- **User Account - Full Name** : votre nom complet
- **User Name** : saisir votre nom de login UGA (comme sur l'image ci-dessous)



- **User Password** : saisir un mot de passe simple, par exemple "etu".
- **Partition disks** : Yes
- **Software Selection** : vérifier que "Debian desktop" n'est pas coché et que "ssh server" est coché car nous ne voulons pas d'interface graphique (comme sur l'image ci-dessous)



- **Device for boot loader** : /dev/sda

1.3. Déplacement de l'image disque

Juste après l'installation, il faut éteindre la machine virtuelle proprement avec la commande suivante :

« # systemctl poweroff »

Cette commande est à effectuer avec le root, pour accéder au root vous pouvez utiliser la commande suivante :

« su - » ou « su »

Un mot de passe vous sera demandé, c'est celui que vous avez choisi lors de l'installation (ici root).

Lorsque la machine virtuelle est bien éteinte, vous pouvez utiliser la commande suivante pour mettre l'image de votre machine virtuelle sur le serveur erebus4 afin de vous faciliter l'accès par la suite.

« S2.03-déplace-image-disque-sur-erebus4 »

1.4. Vérification du serveur Debian

```
root@server-galatn:~# cat /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# systemd generates mount units based on this file, see systemd.mount(5).
# Please run 'systemctl daemon-reload' after making changes here.
#
# <file system> <mount point> <type> <options> <dump> <pass>
# / was on /dev/sda1 during installation
UUID=4994253d-6624-4d47-94ec-7bd72bb791d0 / ext4 errors=remount-ro 0 1
# swap was on /dev/sda5 during installation
UUID=54a8d2cd-a8ab-4232-962f-61cd95eaa495 none swap sw 0 0
/dev/sr0 /media/cdrom0 udf,iso9660 user,noauto 0 0
root@server-galatn:~#
```

Maintenant que l'image de votre machine virtuelle se trouve sur le serveur erebus4, vous pouvez utiliser la machine virtuelle avec la commande suivante :

« S2.03-lance-machine-virtuelle »

En vous connectant en root en faisant la commande suivante :

« cat /etc/fstab » (Le résultat de la commande est présent sur l'image ci-dessus)

Nous pouvons voir que la machine virtuelle est bien installée sur /dev/sda1 et qu'une partition swap est également installée sur /dev/sr0 .

Si vous effectuez la commande suivante :

« dpkg -l | grep xorg » puis « echo \$? » (Pour afficher le résultat de la première commande)

Vous pouvez constater que rien ne se renvoie, c'est normal car nous ne voulons pas d'interface graphique (Xorg) pour ce serveur Debian.

Pour vérifier que votre machine virtuelle peut atteindre d'extérieur, vous pouvez effectuer la commande suivante qui va mettre à jour apt :

« apt update »

Maintenant je vais vous donner quelques indications pour accéder à votre machine virtuelle depuis votre poste Linux (hôte) :

- En SSH, « ssh USER_NAME@localhost -p 2222 »
- En HTTP avec l'URL : http://localhost:8080/
- En HTTPS avec l'URL : https://localhost:4443/
- Avec PostgreSQL, « psql -h localhost -U postgres postgres » (A noter : Le premier postgres correspond à l'utilisateur et que le deuxième postgres correspond à la base de données)

2. Installation Apache et PostgreSQL

2.1. Installation Apache

Afin d'installer Apache, il faut déjà que vous soyez avec le root puis il vous faut exécuter la commande suivante :

« # apt install apache2 »

S'il y a un problème, voici le lien vers le manuel d'installation d'Apache :

<https://httpd.apache.org/docs/2.4/en/install.html>

Effectuer la commande :

« # systemctl status apache2 » pour vérifier qu'apache est démarré.

Si ce n'est pas le cas, vous pouvez effectuer la commande suivante afin de lancer Apache :

« # systemctl start apache2 »

Sans interface graphique, vous ne pouvez pas voir la page HTML. Cependant vous pouvez vous connecter au serveur Apache avec la commande suivante :

« telnet localhost 80 »

Un message devrait s'afficher puis vous devez écrire : « HEAD / HTTP/1.0 »

Le serveur devrait vous répondre "HTTP/1.1 200 OK" afin de confirmer le bon fonctionnement d'Apache.

Vous pouvez également accéder à la page par défaut d'Apache sur votre machine Linux. Dans le navigateur, aller à l'URL : <http://localhost:8080> (comme sur l'image ci-dessous)



2.2. Installation PostgreSQL

Maintenant, vous allez installer PostgreSQL afin de pouvoir posséder votre propre base de données sur votre serveur Debian.

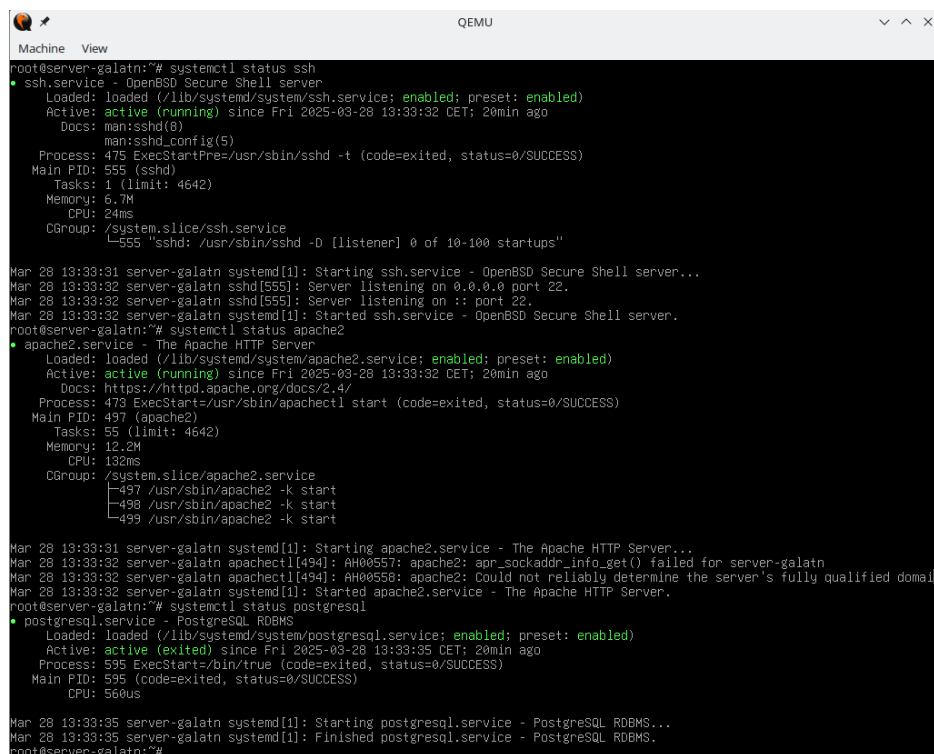
Commencer par installer le serveur et le client sur votre machine virtuelle avec la commande suivante :

« # apt install postgresql »

Vous pouvez vérifier l'installation avec la commande status utilisée précédemment :

« systemctl status postgresql »

A ce stade vous devriez avoir SSH, Apache et PostgreSQL (comme montré sur l'image ci-dessous)



```
Machine View
root@server-galatin:~# systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-03-28 13:33:32 CET; 20min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
  Process: 475 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
    Main PID: 555 (sshd)
      Tasks: 1 (limit: 4642)
     Memory: 6.7M
        CPU: 24ms
    CGroup: /system.slice/ssh.service
            └─555 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Mar 28 13:33:31 server-galatin systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Mar 28 13:33:32 server-galatin sshd[555]: Server listening on 0.0.0.0 port 22.
Mar 28 13:33:32 server-galatin sshd[555]: Server listening on :: port 22.
Mar 28 13:33:32 server-galatin systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
root@server-galatin:~# systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-03-28 13:33:32 CET; 20min ago
     Docs: https://httpd.apache.org/docs/2.4/
  Process: 473 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
    Main PID: 497 (apache2)
      Tasks: 55 (limit: 4642)
     Memory: 12.2M
        CPU: 132ms
    CGroup: /system.slice/apache2.service
            └─497 /usr/sbin/apache2 -k start
              └─498 /usr/sbin/apache2 -k start
                └─499 /usr/sbin/apache2 -k start

Mar 28 13:33:31 server-galatin systemd[1]: Starting apache2.service - The Apache HTTP Server...
Mar 28 13:33:32 server-galatin apachectl[494]: AH00557: apache2: apr_sockaddr_info_get() failed for server-galatin
Mar 28 13:33:32 server-galatin apachectl[494]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name.
Mar 28 13:33:32 server-galatin systemd[1]: Started apache2.service - The Apache HTTP Server.
root@server-galatin:~# systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; preset: enabled)
   Active: active (exited) since Fri 2025-03-28 13:33:35 CET; 20min ago
  Process: 595 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
    Main PID: 595 (code=exited, status=0/SUCCESS)
        CPU: 560us

Mar 28 13:33:35 server-galatin systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Mar 28 13:33:35 server-galatin systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@server-galatin:~#
```

Maintenant que PostgreSQL est installé, vous pouvez vous connecter avec postgres avec la commande suivante :

« # su - postgres » puis « psql -l » pour voir les différentes bases de données présentes.

2.3. Utilisation de PostgreSQL sur la machine virtuelle

Toujours avec postgres, connectez-vous à la base de données postgres avec :

« psql » ou « psql postgres »

Puis créer un nouvel utilisateur avec votre nom :

« CREATE USER [LOGIN] password 'etu' superuser; »

Puis créer une base avec :

« CREATE DATABASE ma_base with owner = [LOGIN]; »

Pensez bien à mettre votre login !

Vous pouvez maintenant vous connecter directement à votre base de données avec l'utilisateur qui a votre nom de login.

Maintenant, aller dans la base « ma_base » avec la commande suivante :

« \c ma_base »

Puis créer une table simple :

« CREATE TABLE ma_table (id integer, nom varchar); »

Puis ajouter y des quelques lignes de données :

« INSERT INTO ma_table (1,'lundi');

INSERT INTO ma_table (2,'mardi'); ... »

À ce stade vous devriez pouvoir interroger votre base de données avec l'utilisateur qui a votre nom de login comme sur l'image ci-dessous.

```
galatn@server-galatn:~$ psql ma_base
psql (15.12 (Debian 15.12-0+deb12u2))
Type "help" for help.

ma_base=# SELECT * FROM ma_table;
 id |  nom
----+-----
  1 | lundi
  2 | mardi
  3 | mercredi
  4 | jeudi
  5 | vendredi
  6 | samedi
  7 | dimanche
(7 rows)

ma_base=# _
```

2.4. Utilisation de PostgreSQL sur la machine Linux

Pour se connecter directement sur votre machine Linux à votre base de données, il faut modifier quelques fichiers afin d'y donner accès.

Commencer par modifier le premier fichier « postgresql.conf » avec la commande :

« nano /etc/postgresql/15/main/postgresql.conf »

Rechercher dans la section CONNECTIONS AND AUTHENTICATION la ligne suivante à décommenter et mettre à jour :

« listen_addresses = '*' »

Rappel des commandes de base de nano :

Ctrl-w : rechercher

Ctrl-o : sauvegarder

Ctrl-x : sortir

Après avoir modifié ce fichier, vous pouvez modifier le deuxième fichier « pg_hba.conf » avec la commande suivante :

« nano /etc/postgresql/15/main/pg_hba.conf »

Modifier la partie « IPv4 local connections » par :

« host all all 0.0.0.0/0 scram-sha-256 »

Maintenant redémarrer le serveur postgresql avec :

« service postgresql restart »

Avec la commande « psql -h localhost ma_base » sur votre machine Linux, vous devriez pouvoir vous connecter à votre base de données et l'interroger comme l'image ci-dessous.

```
galatn@iut2-dg037-d13:~$ psql -h localhost ma_base
Password for user galatn:
psql (15.12 (Debian 15.12-0+deb12u2))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
Type "help" for help.

ma_base=# SELECT * FROM ma_table;
 id |  nom
----+-----
  1 | lundi
  2 | mardi
  3 | mercredi
  4 | jeudi
  5 | vendredi
  6 | samedi
  7 | dimanche
(7 rows)

ma_base=#
```

2.5. Vérification de la base de données

```
galatn@server-galatn:~$ psql -l
```

List of databases							
Name	Owner	Encoding	Collate	Ctype	ICU Locale	Locale Provider	Access privileges
ma_base	galatn	UTF8	en_US.UTF-8	en_US.UTF-8		libc	
postgres	postgres	UTF8	en_US.UTF-8	en_US.UTF-8		libc	
template0	postgres	UTF8	en_US.UTF-8	en_US.UTF-8		libc	=c/postgres +
template1	postgres	UTF8	en_US.UTF-8	en_US.UTF-8		libc	postgres=CtC/postgres +

```
(4 rows)
```

```
galatn@server-galatn:~$ _
```

Maintenant, vous devriez avoir les bases de données comme sur l'image ci-dessus. Avec comme propriétaire (Owner) de votre base (ici ma_base) l'utilisateur qui a votre nom de Login.

Vous pouvez voir le hachage des mots de passe dans la table « pg_shadow ».
Pour cela rien de plus compliqué, faites un Select de la table « pg_shadow ».

« SELECT * FROM pg_shadow; »

Vous devriez avoir un résultat similaire à l'image ci-dessous.

```
ma_base=# SELECT * from pg_shadow;
```

username	usesysid	usecreatedb	usesuper	use repl	usebypassrls	passwd	valuntil	useconfig
postgres	10	t	t	t	t	SCRAM-SHA-256\$4896:ckz8cflxV97dLv0k+M8F2Q==3UG9C46fbKnyvEd4YgmPiLLYw75LwPqIrQo130hptfE=:ZX08IPFMKpncPm1lCF/1mG6EfhXFPUGN6VjtGaqu98=		
galatn	16388	f	t	f	f			

```
(2 rows)
```

```
ma_base=#
```

L'utilisateur postgres est un superutilisateur avec tous les privilèges, notamment la création de bases de données, la réplication et la possibilité de contourner les politiques de sécurité (RLS).

En revanche, votre utilisateur a le droit de créer des bases mais n'a pas les privilèges d'administration ou de réplication. Son mot de passe est stocké sous forme chiffrée avec l'algorithme sécurisé SCRAM-SHA-256.

Ces informations sont sensibles et réservées aux superutilisateurs PostgreSQL.

3. Installation PHP et PhpPgAdmin

3.1. Installation PHP

Afin d'installer PHP, il faut déjà que vous soyez avec le root puis il vous faut exécuter la commande suivante :

```
« # apt install php-common libapache2-mod-php php-cli »
```

S'il y a un problème, voici le lien vers le manuel d'installation de PHP :

<https://www.php.net/manual/en/install.unix.php>

À partir d'ici vous pouvez mettre des fichiers PHP sur votre serveur et y accéder.

Le répertoire qui permet d'accéder à vos fichiers PHP en local : « /var/www/html/ »

Vous allez l'essayer avec le fichier « page_sae_S2.03.php » que vous allez copier à partir de transit.

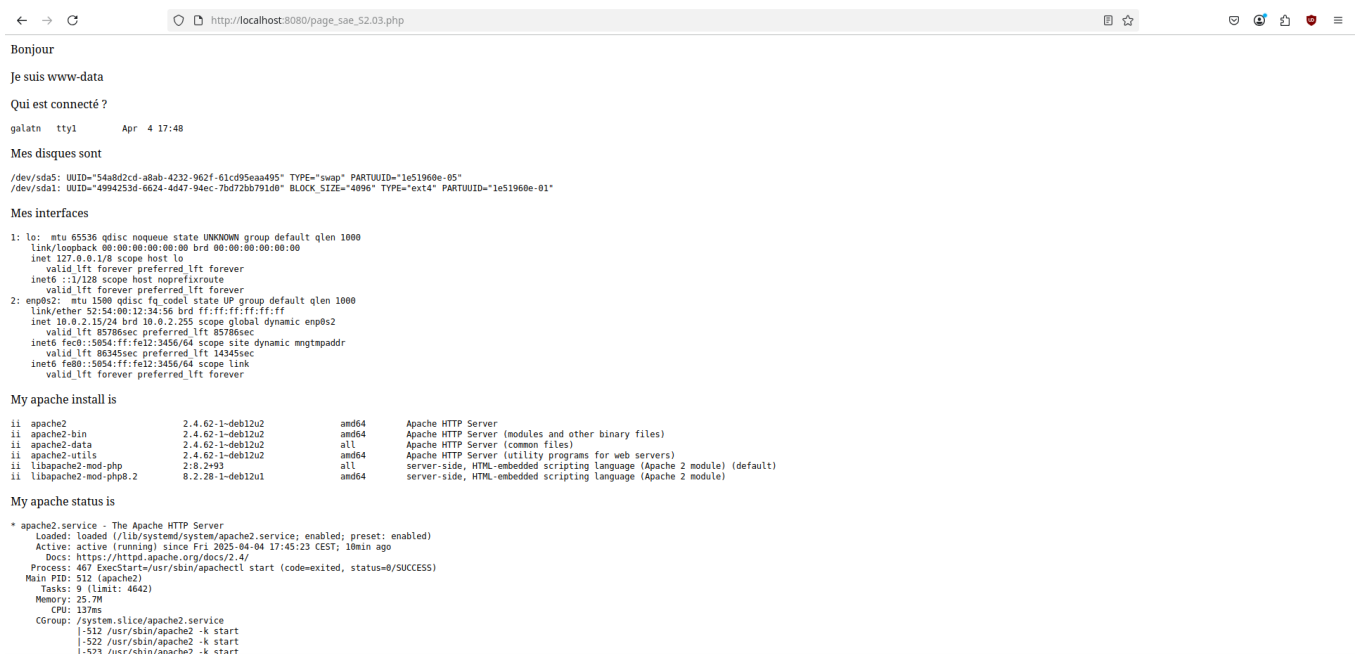
Pour cela utiliser la commande suivante :

```
« scp LOGIN@transit.iut2.univ-grenoble-alpes.fr:/users/info/www/intranet/enseignements/S2.03/page_sae_S2.03.php /var/www/html/ »
```

Vous devez taper yes puis saisir votre mot de passe transit.

Si vous allez sur le lien suivant : http://localhost:8080/page_sae_S2.03.php

Vous devriez accéder à la page comme sur l'image ci-dessous.



```
Bonjour

Je suis www-data

Qui est connecté ?

galatn  tty1      Apr  4 17:48

Mes disques sont

/dev/sda5: UUID="54a8d2cd-a8ab-4232-962f-61cd95eaa495" TYPE="swap" PARTUUID="1e51960e-05"
/dev/sda1: UUID="4994253d-6624-4647-94ec-7bd72bb79108" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="1e51960e-01"

Mes interfaces

1: lo: mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 08:00:00:00:00:00 brd 08:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s2: mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 52:54:00:12:34:56 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s2
        valid_lft 85786sec preferred_lft 85786sec
    inet6 fe80::5054:ff:fe12:3456/64 scope site dynamic mngtmpaddr
        valid_lft 86345sec preferred_lft 14345sec
    inet6 fe80::5054:ff:fe12:3456/64 scope link
        valid_lft forever preferred_lft forever

My apache install is

ii apache2                2.4.62-1-deb12u2      amd64      Apache HTTP Server
ii apache2-bin            2.4.62-1-deb12u2      amd64      Apache HTTP Server (modules and other binary files)
ii apache2-data            2.4.62-1-deb12u2      all        Apache HTTP Server (common files)
ii apache2-utils           2.4.62-1-deb12u2      amd64      Apache HTTP Server (utility programs for web servers)
ii libapache2-mod-php      2:8.2+93              all        server-side, HTML-embedded scripting language (Apache 2 module) (default)
ii libapache2-mod-php8.2   8.2.28-1-deb12u1      amd64      server-side, HTML-embedded scripting language (Apache 2 module)

My apache status is

* apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-04-04 17:45:23 CEST; 10min ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 407 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
  Main PID: 512 (apache2)
    Tasks: 9 (limit: 4642)
     Memory: 25.7M
        CPU: 137ms
   CGroup: /system.slice/apache2.service
           └─512 /usr/sbin/apache2 -k start
             └─522 /usr/sbin/apache2 -k start
             └─523 /usr/sbin/apache2 -k start
```

3.2. Installation PhpPgAdmin

Pour installer PhpPgAdmin, il faut le package qui se trouve sur :

<http://ftp.de.debian.org/debian>

Pour accéder à ce package, il vous faut modifier les sources apt présent dans le fichier « sources.list »

Vous allez donc utiliser la commande suivante :

« nano /etc/apt/sources.list »

Puis ajouter la ligne suivante : « deb http://ftp.de.debian.org/debian bookworm-backports main »

Lorsque cela est fait, sauvegarder (Ctrl-s) et quitter (Ctrl-x) puis mettez à jour apt avec :

« apt update »

Maintenant, vous pouvez télécharger PhpPgAdmin avec la commande suivante :

« # apt install phppgadmin »

Maintenant il va falloir permettre la connexion en modifiant le fichier « Connection.php »

Donc exécuter la commande : « nano /usr/share/phppgadmin/classes/database/Connection.php »

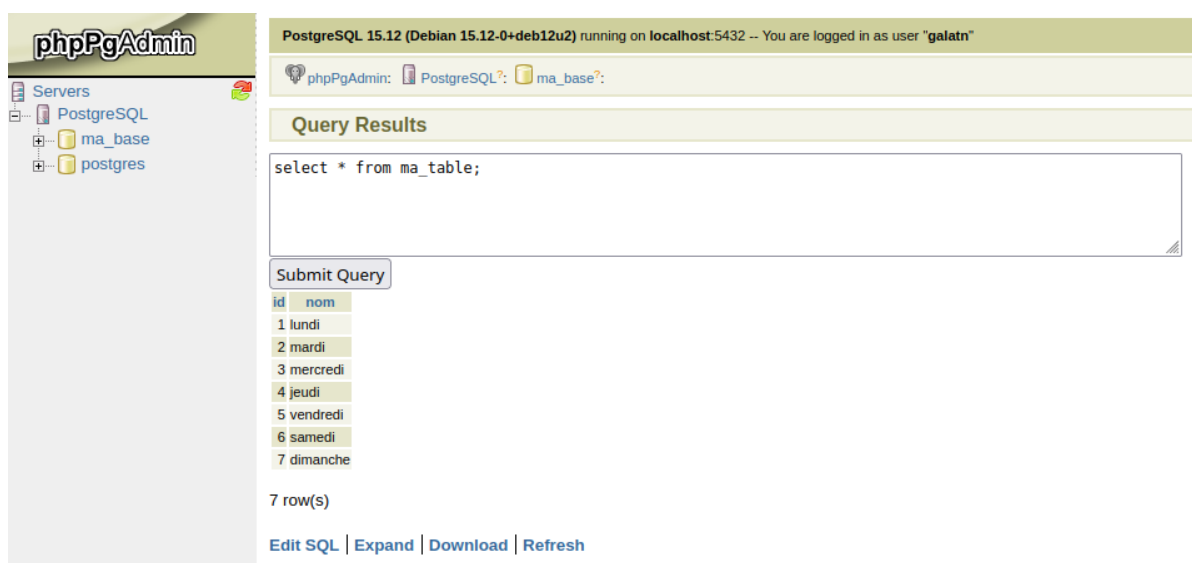
Et modifier la ligne : « case '14': return 'Postgres'; break; »

par « case '15': return 'Postgres'; break; »

Vous pouvez utiliser PhpPgAdmin en allant sur le lien suivant :

<http://localhost:8080/phppgadmin/>

Puis vous pouvez interroger votre base de données comme sur l'image ci-dessous.



4. Sécurité

A ce stade, vous devriez avoir un espace de stockage similaire à l'image ci-dessous.

Essayez avec la commande : « df -h »

```
root@server-galatn:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            1.9G   0    1.9G   0% /dev
tmpfs           392M  480K  392M   1% /run
/dev/sda1       3.0G  1.6G  1.3G  56% /
tmpfs           2.0G  1.1M  2.0G   1% /dev/shm
tmpfs           5.0M   0    5.0M   0% /run/lock
tmpfs           392M   0    392M   0% /run/user/1000
root@server-galatn:~#
```

Cependant, des failles de sécurité sont régulièrement découvertes dans les paquets Debian, la solution est de régulièrement mettre à jour le système et ses paquets avec :

« # apt update && apt upgrade -y »

Vous pouvez également supprimer les logiciels inutiles comme par exemple :

« # systemctl disable bluetooth »

« # systemctl disable avahi-daemon »

Pour renforcer la sécurité de votre système, vous pouvez télécharger un logiciel qui empêchera les brute-force tels que Fail2ban par exemple.

« # apt install fail2ban »

« # systemctl enable --now fail2ban »

Vous pouvez également mettre en place un pare-feu avec UFW :

« # ufw enable »

« # ufw allow OpenSSH »

« # ufw allow 'Apache Full' »

En ce qui concerne Apache, vous pouvez commencer par masquer la version en modifiant le fichier « security.conf » avec la commande suivante :

« nano /etc/apache2/conf-available/security.conf » et en mettant les lignes suivantes :

ServerTokens Prod

ServerSignature Off

Vous pouvez également utiliser HTTPS avec Let's Encrypt par exemple :

« # apt install certbot python3-certbot-apache »

« # certbot --apache »

Toutes ces mesures vous permettront de mieux sécuriser votre serveur Debian.

5. Résumé des commandes

COMMANDES :	DESCRIPTION
sha512sum [FICHER]	Calcule le hachage SHA-512 du fichier pour vérifier son intégrité
S2.03-lance-installation	Lance la procédure d'installation de Debian
S2.03-déplace-image-disque-sur-erebus4	Déplace l'image disque vers la machine "erebus4"
S2.03-lance-machine-virtuelle	Lance la machine virtuelle
# systemctl poweroff	Éteint proprement la machine virtuelle
su -	Passe à l'utilisateur root
cat [CHEMIN]	Affiche le contenu d'un fichier
apt update	Met à jour la liste des paquets disponibles.
ssh USER_NAME@localhost -p 2222	Se connecte en SSH à sa propre machine via le port 2222
psql -h localhost -U [USER] [BD]	Se connecte à une base de données PostgreSQL locale avec l'utilisateur donné.
# apt install [LOGICIEL]	Installe un logiciel avec apt
# systemctl status [LOGICIEL]	Affiche le statut d'un logiciel
# systemctl start [LOGICIEL]	Démarre un logiciel
# su - postgres	Bascule vers l'utilisateur système postgres
psql -l	Liste toutes les bases de données PostgreSQL
psql [USER]	Ouvre une base de données avec l'utilisateur choisi
service postgresql restart	Redémarre le service PostgreSQL.
nano [CHEMIN]	Ouvre un fichier avec l'éditeur Nano
scp [CHEMIN_SOURCE] [CHEMIN_DEST]	Copie un fichier à distance via SSH
df -h	Affiche l'espace disponible et utilisé.
SQL :	-----
CREATE USER [LOGIN] [OPTION];	Crée un nouvel utilisateur PostgreSQL
CREATE DATABASE [BASE] WITH owner = [USER];	Crée une nouvelle base de données nommée et lui ajoute un propriétaire
SELECT * FROM pg_shadow;	Affiche la table système pg_shadow
\c [BD]	Change de base de données dans PostgreSQL

6. Installation d'une machine virtuelle sur une clé USB (Annexe)

Pour créer une machine virtuelle sur clé USB, il faut d'abord préparer la clé USB en la formatant en exFAT (ou ext4) afin de pouvoir créer le disque sans problème.

Une fois la clé formatée, vous devez créer le disque dur virtuel de votre serveur Debian directement sur votre clé USB. Vous pouvez utiliser la commande suivante (ici, 20 Go sont alloués) :

```
« qemu-img create -f qcow2 /media/$USER/NOM_DE_LA_CLÉ/debian-vm.qcow2 20G »
```

Une fois cette commande effectuée, vous devez télécharger et placer l'ISO Debian sur la clé dans un répertoire iso (iso/debian-12.10.0-amd64-netinst.iso)

Vous pouvez ici aussi vérifier l'ISO avec SHA (cf : 1.1)

Ensuite, il vous faut réutiliser le script « S2.03-commun » en modifiant et ajoutant certaines choses :

```
qemu-system-x86_64 \  
-machine q35 \  
-cpu host \  
-m 4G \  
-enable-kvm \  
-device VGA,xres=1024,yres=768 \  
-display gtk,zoom-to-fit=off \  
-cdrom /media/usb/iso/debian-12.iso \                               (ajout)  
-boot d \                                                            (ajout)  
-drive file=/media/usb/debian-vm.qcow2,format=qcow2 \              (modifié)  
-device e1000,netdev=net0 \  
-netdev user, id=net0,hostfwd=tcp::2222-:22, hostfwd=tcp::4443-:443, hostfwd=tcp::8080-:80,  
hostfwd=tcp::5432-:5432
```

Votre serveur s'installera sur votre clé USB (Attention à bien vérifier les chemins!)

Vous pouvez mettre ce script (sans les lignes cdrom et boot) dans un fichier .sh (ex : demarrer_vm.sh) afin de ne pas remettre le script à chaque démarrage et de juste exécuter :
« # ./demarrer_vm.sh »

L'installation des différents logiciels reste cependant similaire car la machine virtuelle Debian fonctionne de la même manière, même si elle est hébergée sur clé USB.