

SAE 2.03

Debian(11) installation guide for Qemu/KVM
with Apache, PostgreSQL and PHP

Table of content

Introduction.....	4
Installing Debian.....	5
Iso image preparation and integrity:.....	5
Installing the Debian system on the virtual machine:.....	6
Optimisation of the disk image:.....	9
Verify the Debian server:.....	10
Port forwarding and SSH access:.....	12
Installing Packages.....	15
Apt:15	
Apache2.....	16
Installing apache2 package:.....	16
Verify Apache2 connection to the internet:.....	17
PostgreSQL.....	18
Installing postgresQL package:.....	18
Configure/setup postgresQL:.....	19
Setup accessibility to postgresQL:.....	21
PHP.....	24
Installing postgresQL package:.....	24
PhpPgAdmin.....	26
Installing PhpPgAdmin package:.....	26
Setting up PhpPgAdmin:.....	26
Additional PHP:.....	27
SSH Key access user [Login].....	28
SSH Key access root.....	29
To conclude.....	29
Security and evolutions:.....	29

Introduction

This guide will teach you how to install Debian 11 (Bullseye) software as a server on a Qemu/KVM virtual machine (VM).

Also, this guide includes instructions on how to install the Apache, PostgreSQL, and PHP (including PhpPgAdmin) packages inside the server.

Attribute summary:

- Software installed: Debian version 11.x, nicknamed (bullseye).
- For x86 64-bits processors.
- With ISO image of type "netinst".

This guide is in english but includes french parameters.

The abbreviation of virtual machine is VM.

You can find appendices (annexes) at the end of the guide for more specifications on certain commands.

Installing Debian

Iso image preparation and integrity:

The ISO image should already have been installed in:

```
$ /usr/local/images-ISO
```

Visualize this folder containing the iso image : debian-11.7.0-amd64-netinst.iso

```
kersuaar@transit:/usr/local/images-ISO$ ls -l
total 4372000
-rw-r--r-- 1 root staff 407896064 29 avril 15:36 debian-11.7.0-amd64-netinst.iso
-rw-r--r-- 1 root staff 4071903232 20 oct. 2022 ubuntu-22.10-desktop-amd64.iso
```

Figure 1: Result of `ls -l` command

To verify the integrity of the iso image copy this link below in your browser:

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

Then at the bottom of the page select the SHA-512UMS iso image.

Name	Last modified	Size
Parent Directory	-	-
SHA256SUMS	2023-04-29 20:22	302
SHA256SUMS.sig	2023-04-29 22:48	833
SHA512SUMS	2023-04-29 20:22	494
SHA512SUMS.sig	2023-04-29 22:48	833
debian-11.7.0-amd64-netinst.iso	2023-04-29 15:36	389M
debian-edu-11.7.0-amd64-netinst.iso	2023-04-29 15:36	450M
debian-mac-11.7.0-amd64-netinst.iso	2023-04-29 15:36	386M

Figure 2: Initial ISO image

!! if the iso image isn't installed you can still download it from the site at SHA512SUM. !!

Check the integrity of the image by visually comparing the two fingerprints. To do this, load the fingerprints in the terminal using the command below:

```
$ sha512sum /usr/local/image-ISO/debian-11.7.0-amd64-netinst.iso
```

You should see that the footprint of your iso-image corresponds to the footprint of the debian-11.7.0-amd64-netinst.iso as shown below.

```
4460ef6470f6d8ae193c268e213d33a6a5a0da90c2d30c1024784faa4e4473f0c9b546a41e2d34c43fbbd43542ae4fb93cfd5cb6ac9b88a476f1a6877c478674  debian-11.7.0-amd64-netinst.iso
images-ISO: bash — Konsole

kersuaar@pc-dg-027-10:/usr/local/images-ISO$ sha512sum debian-11.7.0-amd64-netinst.iso
4460ef6470f6d8ae193c268e213d33a6a5a0da90c2d30c1024784faa4e4473f0c9b546a41e2d34c43fbbd43542ae4fb93cfd5cb6ac9b88a476f1a6877c478674  debian-11.7.0-amd64-netinst.iso
```

Figure 3: Iso image comparison

Installing the Debian system on the virtual machine:

Firstly, launch a terminal and start your virtual machine(in this guide it will be):

```
$ S2.03-lance-installation
```

this command line launch QEMU with the following command inside:

qemu-system-x86_64	Specifies the use of QEMU to emulate an x86_64 system.
-M q35	Specifies the Q35 machine model to use.
-cpu host	Uses the same processor configuration as the host (the host machine on which the command is executed).
-m 4G	Allocates 4 GB of memory to the virtual machine. [Number of parallel coroutines for the conversion process].
-enable-kvm	Enables hardware virtualization to improve performance.
-device VGA,xres=1024,yres=768	Adds a VGA graphics card and sets its resolution to 1024x768.
-display gtk,zoom-to-fit=off	Uses GTK+ display for the virtual machine and disables the automatic zoom function.
-drive \$drive	Specifies the disk image file to use for the virtual machine (the variable \$drive must be defined).
-device e1000,netdev=net0	Adds a virtual E1000 network card and connects it to the virtual network "net0".
-netdev user, id=net0, hostfwd=tcp::2222-:22, hostfwd=tcp::4443-:443, hostfwd=tcp::8080-:80 , hostfwd=tcp::5432-:5432"	Sets the network (virtual network "net0") to be a user network with port forwarding for ports 22 (SSH), 443 (HTTPS), 80 (HTTP), and 5432 (PostgreSQL).

!! Do not attempt to resize the VM; it is not supposed to be resized during installation. !!

The table above provides an overview of the various options available when launching the QEMU virtual machine using the command.

Now, once the virtual machine is launched, here are the main steps you will need to follow:

[STEP 1] Choose the non-graphical installation:

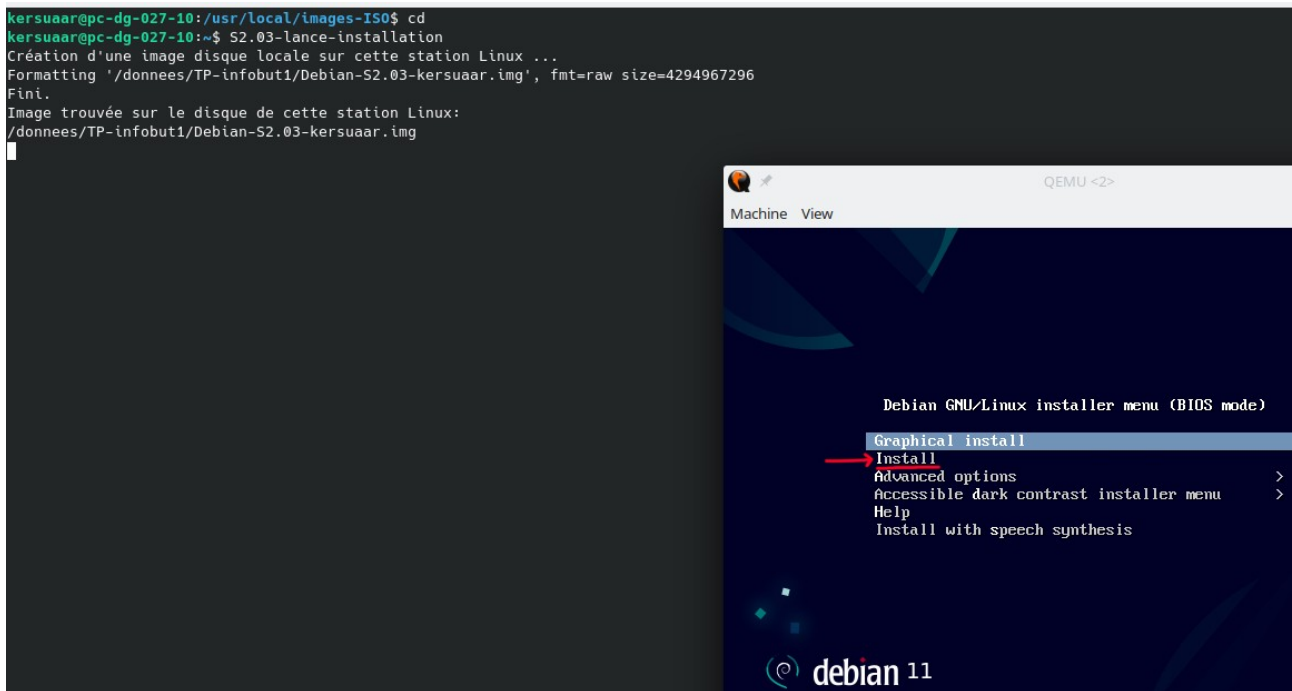


Figure 4: non graphical installation

[STEP 2] Follow these steps during the installation:

(If a step is not listed, choose the default option).

(Some images will be provided to help you visualize the result you should get).

Language: English

Location: Other/Europe/France

Locales: United States, en_US.UTF-8

Keyboard: French

Hostname: Use server-"YOUR_LOGIN_UGA"

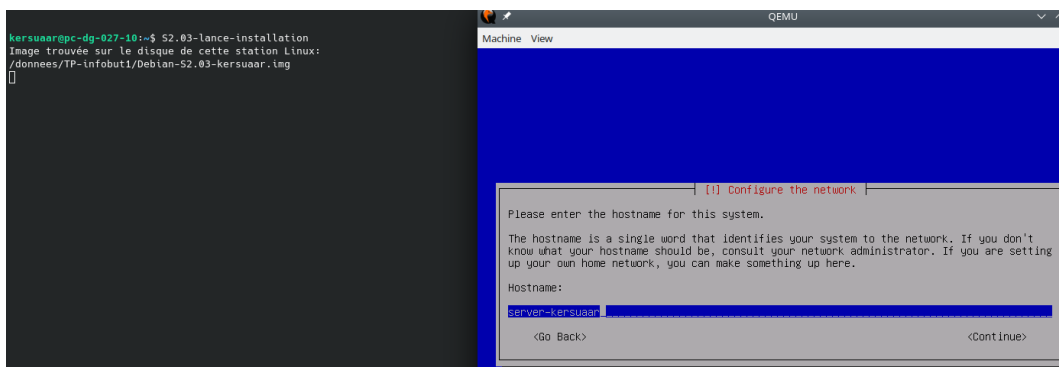


Figure 5: Hostname configuration

Root Password: It is recommended to use a simple password, such as "root". In this context, it does not pose a security issue. Check the "Show Password" box to ensure that the entered password is correct.

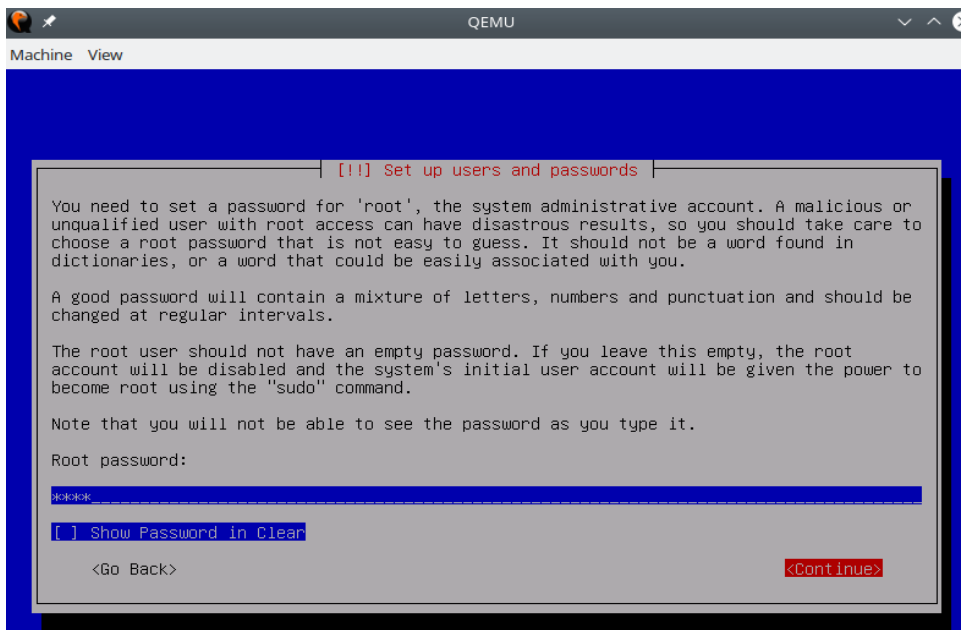


Figure 6: Root password configuration

User Account - Full Name: Your full name, for example "Jean Toto"

User Name: Enter your UGA login name

User Password: Enter a simple password, for example "etu". Check the "Show Password" box to ensure that the entered password is correct.

Partition Disks: Guided - use entire disk

Partition Disks: All files in one partition

Partition Disks: Yes

Software Selection: Verify that "Debian desktop" is not checked and "ssh server" is checked

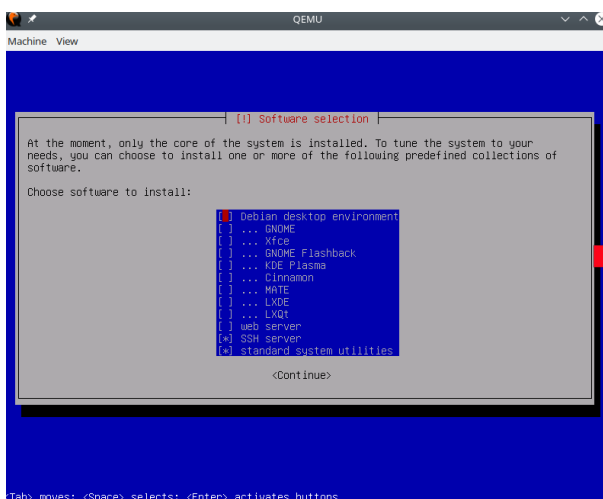


Figure 7: Software selection configuration

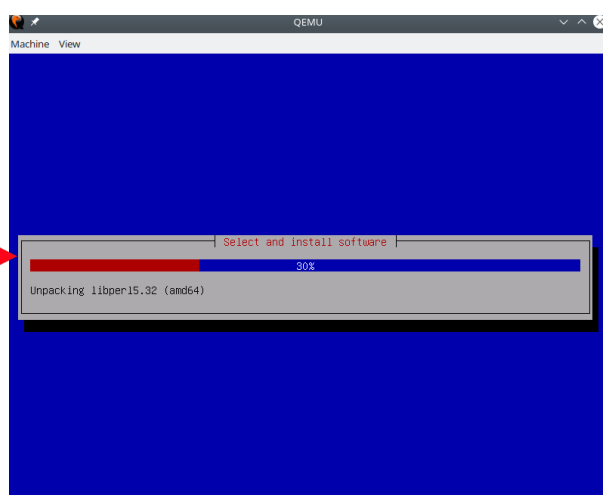


Figure 8: Installing software

Install GRUB: Yes

Device for Boot Loader: /dev/sda

Once the installation is complete, the virtual machine (VM) will restart.

To properly shut down your VM after each use, you will need to switch to the root account using [su -] or other commands (like typing root in the login):

```
server-kersuaar login: root
Password:
Linux server-kersuaar 5.10.0-22-amd64 #1 SMP Debian 5.10.178-3 (2023-04-22) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@server-kersuaar:~#
```

Figure 9: Accessing root user inside the virtual machine

and then type:

```
# poweroff
```

This command is useful for properly shutting down your virtual machine after each utilisation.

Optimisation of the disk image:

The disk image has been generated on the local disk of your Linux station. To enhance the utilization of the disk image for the virtual machine, you have the option to transfer this image to a USB key or a server (eberus4 in this guide).

In this guide, you can execute a script directly in your terminal to facilitate this process:

```
$ S2.03-déplace-image-disque-sur-erebus4
```

```
kersuaar@pc-dg-027-10:~$ S2.03-déplace-image-disque-sur-erebus4
Image trouvée sur le disque de cette station Linux:
/donnees/TP-infobut1/Debian-S2.03-kersuaar.img
Déplacement de l'image disque...
```

Figure 10: result of the moving command for the disk image

This will make it easier to use the virtual machine.

Verify the Debian server:

Now that the virtual machine (VM) is installed, proceed with executing it.

Next, we will verify if the Debian server can access the external network (internet) using its IP address.

To see your IP/Ethernet configuration type the following command:

```
$ ip addr
```

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast 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 86385sec preferred_lft 86385sec
    inet6 fec0::5054:ff:fe12:3456/64 scope site dynamic mngtmpaddr
        valid_lft 86385sec preferred_lft 14385sec
    inet6 fe80::5054:ff:fe12:3456/64 scope link
        valid_lft forever preferred_lft forever
```

Figure 11: result of the `ip addr` command

Alternatively you can achieve this by typing one of the following command lines, as shown below:

- `traceroute google.com`
- `traceroute 8.8.8.8`

```
$ traceroute [IP_address]
```

You should get:

```
kersuaar@server-kersuaar:~$ traceroute google.com
traceroute to google.com (142.250.179.110), 30 hops max, 60 byte packets
 1  10.0.2.2 (10.0.2.2)  0.128 ms  0.095 ms  0.090 ms  → Default gateway
 2  sw-dg-40d-1-tx.iut2.upmf-grenoble.fr (192.168.141.19)  6.742 ms  6.626 ms  6.853 ms  IP of a nearby router
 3  rt-wan.iut2.upmf-grenoble.fr (193.55.51.1)  1.502 ms  1.877 ms  2.129 ms
 4  r-viallet1.grenet.fr (193.54.184.185)  0.977 ms  0.933 ms  0.835 ms
 5  tigre1.grenet.fr (193.54.185.17)  12.373 ms  12.284 ms  12.494 ms
 6  te1-4-grenoble-rtr-021.noc.renater.fr (193.51.181.94)  1.382 ms  1.676 ms  1.918 ms
 7  te0-0-0-12-ren-nr-lyon2-rtr-091.noc.renater.fr (193.51.177.57)  6.953 ms  te-0-1-0-12-ren-nr-lyon2-rtr-091.noc.renater.fr (193.51.180.67)  7.262 ms  6.953 ms
 8  xe-1-0-1-marseille2-rtr-131.noc.renater.fr (193.51.177.196)  6.660 ms  xe-1-0-14-marseille2-rtr-131.noc.renater.fr (193.51.177.196)  6.829 ms  reserve-ip9-ren-nr-marseille2-rtr-091.noc.renater.fr (193.51.177.99)  6.722 ms
 9  72.14.218.132 (72.14.218.132)  6.680 ms  6.679 ms  6.648 ms
10  108.170.252.243 (108.170.252.243)  7.950 ms  74.125.244.227 (74.125.244.227)  7.580 ms  108.170.252.242 (108.170.252.242)  7.55 ms
11  216.239.35.209 (216.239.35.209)  13.489 ms  13.163 ms  64.233.175.243 (64.233.175.243)  13.970 ms
12  72.14.238.54 (72.14.238.54)  13.244 ms  13.868 ms  108.170.233.115 (108.170.233.115)  16.054 ms
13  108.170.245.1 (108.170.245.1)  15.487 ms  15.023 ms  108.170.244.193 (108.170.244.193)  14.536 ms
14  142.251.49.137 (142.251.49.137)  14.011 ms  13.461 ms  142.251.49.135 (142.251.49.135)  12.922 ms
15  par21s20-in-f14.1e100.net (142.250.179.110)  12.816 ms  12.948 ms  12.944 ms
kersuaar@server-kersuaar:~$ host 72.14.218.132
```

Figure 12: result of the `traceroute` command

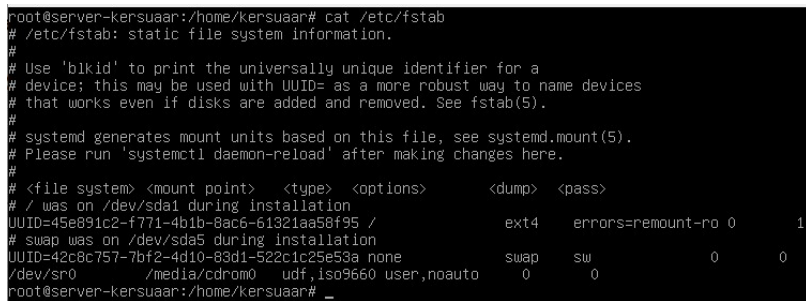
Now, verify the absence of the Xorg server with this command in multiple files:

```
$ dpkg -l | grep xorg
```

It is expected that no output will be returned because the Xorg server is not present. The absence of Xorg server is intentional as it is designed for machines with a graphical interface.

Alternatively, you can view your filesystem table, also known as fstab, by entering the following command:

```
# cat /etc/fstab
```



```
root@server-kersuaar:/home/kersuaar# 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=45e891c2-f771-4b1b-8ac6-61321aa58f95 / ext4 errors=remount-ro 0 1
# swap was on /dev/sda5 during installation
UUID=42c8c757-7bf2-4d10-83d1-522c1c25e53a none swap sw 0 0
/dev/sr0 /media/cdrom0 udf,iso9660 user,noauto 0 0
root@server-kersuaar:/home/kersuaar# _
```

Figure 13: result of the `/etc/fstab` command

The fstab is a configuration table designed to ease the burden of mounting and unmounting file systems to a machine (the VM in this guide).

Port forwarding and SSH access:

In order to facilitate connections to your server running in a virtual machine from clients on other stations, multiple port redirections have been implemented within the VM.

Here is the list of them:

Network service	Virtual machine port	Linux station port
SSH	22	2222
HTTP	80	8080
HTTPS	443	4443
PostgreSQL	5432	5432

Firstly check if your ssh software is running on your VM with the root user:

```
# systemctl status ssh
```

```
root@server-kersuaar:~# systemctl status ssh
• ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2023-05-10 13:31:27 CEST; 25min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
   Main PID: 423 (sshd)
     Tasks: 1 (limit: 4661)
    Memory: 3.8M
       CPU: 53ms
    CGroup: /system.slice/ssh.service
            └─423 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

May 10 13:31:27 server-kersuaar systemd[1]: Starting OpenBSD Secure Shell server...
May 10 13:31:27 server-kersuaar sshd[423]: Server listening on 0.0.0.0 port 22.
May 10 13:31:27 server-kersuaar sshd[423]: Server listening on :: port 22.
May 10 13:31:27 server-kersuaar systemd[1]: Started OpenBSD Secure Shell server.
root@server-kersuaar:~# _
```

Figure 14: status of ssh software

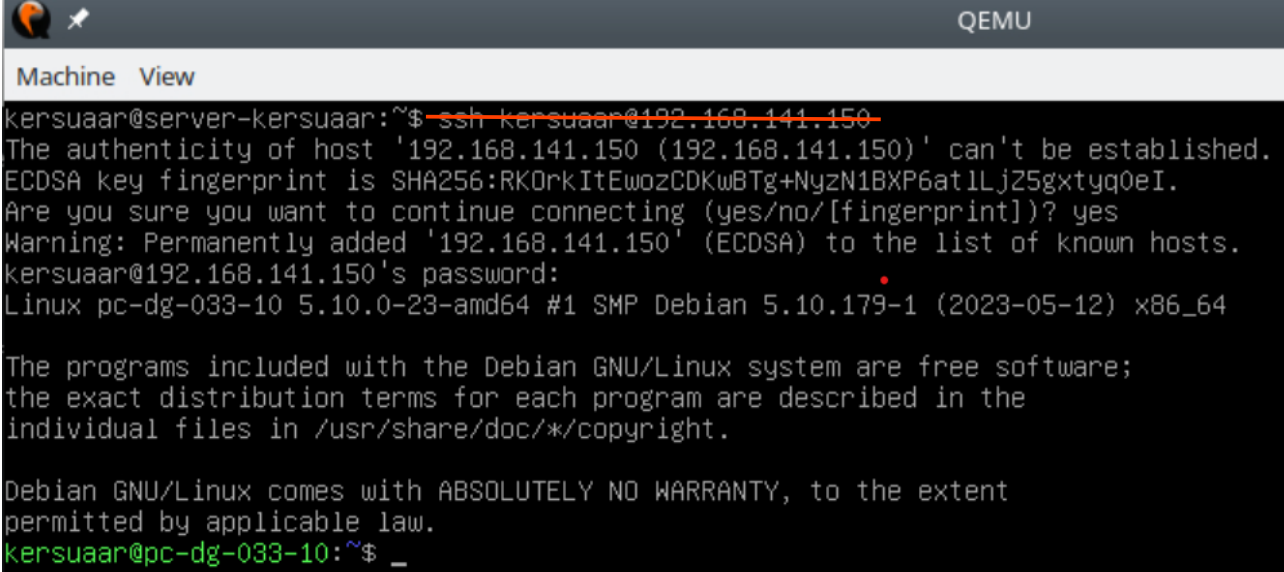
If you want to access your server from a terminal on your station, you can execute an SSH command line as shown below:

```
$ ssh Your_Login@localhost -p 2222
```

[**-p 2222**] stand for the port of your station for the VM.
[**localhost**] is the address of your server.

Alternatively, if you want to access a terminal of your station from a VM you can do it by getting the IP address of your station and run this command:

```
$ ssh Your_Login@Station_Ip_Address
```



The screenshot shows a QEMU terminal window with a title bar that includes a QEMU logo and the word "QEMU". Below the title bar is a tab labeled "Machine View". The terminal content shows a user at a server prompt running an SSH command to connect to 192.168.141.150. The connection is successful after accepting the host's ECDSA key fingerprint. The user is then prompted for a password and successfully logs in to a Debian system. The system information displayed includes the kernel version (5.10.0-23-amd64) and architecture (x86_64). The terminal also shows the standard Debian GNU/Linux disclaimer about warranty and copyright.

```
kersuaar@server-kersuaar:~$ ssh kersuaar@192.168.141.150
The authenticity of host '192.168.141.150 (192.168.141.150)' can't be established.
ECDSA key fingerprint is SHA256:RKOrkItEwozCDKwBTg+NyzN1BXP6at1LjZ5gxyq0eI.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.141.150' (ECDSA) to the list of known hosts.
kersuaar@192.168.141.150's password:
Linux pc-dg-033-10 5.10.0-23-amd64 #1 SMP Debian 5.10.179-1 (2023-05-12) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
kersuaar@pc-dg-033-10:~$ _
```

Figure 15: result of ssh from VM to Linux station

Finally you can connect to an http service by starting your server and running in your web browser URL this address:

```
http://localhost:8080/
```

You should see the page of the Apache2 server once we have installed it further in this guide.

When you access the HTTP URL, the request will be redirected to port 8080 of your Linux station, which will in turn be redirected to port 80 of your virtual machine.

To test if your server is functional, try installing a Debian package.

For example, you can install the text editor "micro" or "nano" if you don't already have it.

```
# apt install [micro / nano]
```

As shown in this screenshot:

```

root@server-kersuaar:/home/kersuaar# dpkg -s micro
Package: micro
Status: install ok installed
Priority: optional
Section: editors
Installed-Size: 14092
Maintainer: Utkarsh Gupta <utkarsh@debian.org>
Architecture: amd64
Source: micro (2.0.8-1)
Version: 2.0.8-1+b6
Recommends: xclip
Description: modern and intuitive terminal-based text editor
 micro is a terminal-based text editor that aims to be easy to use and
 intuitive, while also taking advantage of the full capabilities of
 modern terminals.
.
As the name indicates, micro aims to be somewhat of a successor to the
 nano editor by being easy to install and use.
Built-Using: golang-1.15 (= 1.15.9-6), golang-github-blang-semver (= 3.6.1-2), golang-github-dustin-go-humanize (= 1.0.0-2), golang-github-flynn-json5 (= 0.0~git20160717.7620272-2), golang-github-gdamore-encoding (= 1.0.0-2), golang-github-go-errors-errors (= 1.0.1-4), golang-github-kballard-go-shellquote (= 0.0~git20180428.95032a8-1), golang-github-lucasb-eyer-go-colorful (= 1.0.0-1), golang-github-mattn-go-isatty (= 0.0.12-1), golang-github-mattn-go-runewidth (= 0.0.9-1), golang-github-mitchellh-go-homedir (= 1.1.0-1), golang-github-sergi-go-diff (= 1.1.0-1), golang-github-xo-terminfo (= 0.0~git20210125.ca9a967-1), golang-github-yuin-gopher-lua (= 0.0~git20170915.0.eb1c729-4), golang-github-zyedidia-clipboard (= 1.0.3-1), golang-github-zyedidia-glob (= 0.0~git20170209.dd4023a-1.1), golang-github-zyedidia-pty (= 1.1.1+git20180126.3036466-3), golang-github-zyedidia-tcell (= 2.0.6-1), golang-github-zyedidia-terminal (= 0.0~git20180726.533c623-2), golang-golang-x-sys (= 0.0~git20210124.22da62e-1), golang-golang-x-text (= 0.3.6-1), golang-layeh-gopher-luar (= 1.0.4-1.1), golang-yaml.v2 (= 2.4.0-1)
Homepage: https://micro-editor.github.io
root@server-kersuaar:/home/kersuaar# _

```

Here the command [dpkg] is the package manager of Debian and Debian-based system.

\$ dpkg -s	Shows the status of an installed package.
------------	---

Installing Packages

Before using the apt software, we need to update it to prevent any disfunctionment during further installation.

Apt:

switch to the root user and type the command as shown below:

```
# apt update
```

this commande is not modifying any packages and update the internal database of available software packages and their versions.

You can upgrade the packages by typing:

```
# apt upgrade
```

but it will download and installs the latest versions of packages from the repositories. It will modifies the installed software applications.

Apache2

Next, we will proceed with the installation of Apache2 to have a web server to visualize web pages.

Installing apache2 package:

!/ Below is the documentation if you need further explanation !/

[Compiling and Installing - Apache HTTP Server Version 2.4](https://httpd.apache.org/docs/2.4/en/install.html)

- Direct URL :

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

To install Apache on your virtual machine you will need to use the root user in your VM.

Then proceed by running the command to download the package via apt:

```
# apt install apache2
```

And start your HTTP server (Apache2):

```
# systemctl restart apache2
```

Since your server is a non-graphical virtual machine you can verify if the Apache2 http server is running type the following command:

```
# systemctl status apache2.service
```

- [(.service) is not needed, it will be added automatically if you don't put it after the package name.]

```
root@server-kersuaar:~# service apache2 start
root@server-kersuaar:~# systemctl status apache2
• apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2023-05-10 13:37:05 CEST; 4min 51s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 933 (apache2)
    Tasks: 55 (limit: 4661)
   Memory: 9.1M
      CPU: 65ms
   CGroup: /system.slice/apache2.service
           └─ 933 /usr/sbin/apache2 -k start
              935 /usr/sbin/apache2 -k start
              936 /usr/sbin/apache2 -k start

May 10 13:37:05 server-kersuaar systemd[1]: Starting The Apache HTTP Server...
May 10 13:37:05 server-kersuaar apachectl[932]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, please see the Apache documentation for more details.
May 10 13:37:05 server-kersuaar systemd[1]: Started The Apache HTTP Server.
lines 1-16/16 (FND)
```

Figure 16: status of apache2 server

Alternatively since you can't display a HTML page. You can connect to the apache server with **telnet** on port 80(default port of web server).

Verify Apache2 connection to the internet:

Now request the HTTP header of some [Web page] by typing `HEAD/ HTTP/1.0` as shown below:

```
$ telnet localhost 80
(command running below)
Trying ::1...
Connected to localhost.
Escape character is '^]'.

(command you need to type)
HEAD / HTTP/1.0 (Press enter twice to confirm!!)

[press enter]
[press enter]

(If the connection succeeded the server will respond with)
HTTP/1.1 200 OK
[...]
```

However, you can only view the HTML page on the host machine as the VM does not have a graphical interface. To accomplish this, you will need to redirect the port from the host machine to port 80 (the default port for a web server) on the VM.

In this guide, this is achieved through the provided launch script. You can now open a web browser on your host machine and access the following URL:

http://localhost:8080

Ensure that you are redirected to the default Apache page after installing the `apache2` package in your VM.

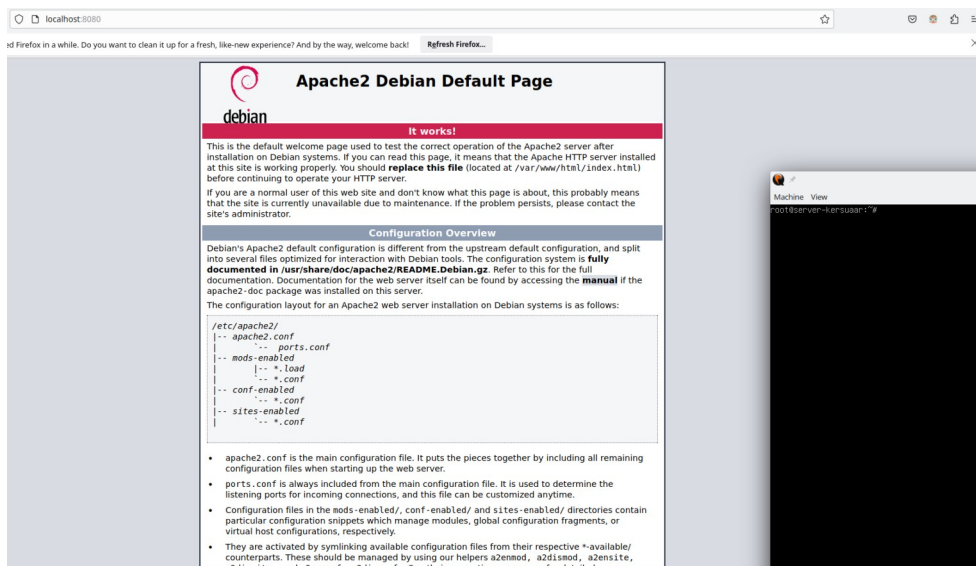


Figure 17: result of the url command for apache2

PostgreSQL

Next, we will proceed with the installation of PostgreSQL to enable the creation and usage of databases.

Installing postgresql package:

!! Below is the documentation if you need further explanation !!

<https://www.postgresql.org/>

1) First, install the package PostgreSQL (server and client) by running this command below in the root user:

```
# apt install postgresql
```

And verify that postgresql is running:

```
# systemctl status postgresql.service
```

```
root@server-kersuaar:~# systemctl status postgresql.service
• postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; vendor preset: enabled)
   Active: active (exited) since Wed 2023-05-10 13:50:13 CEST; 6min ago
   Main PID: 2492 (code=exited, status=0/SUCCESS)
     Tasks: 0 (limit: 4661)
    Memory: 0B
       CPU: 0
    CGroup: /system.slice/postgresql.service

May 10 13:50:13 server-kersuaar systemd[1]: Starting PostgreSQL RDBMS...
May 10 13:50:13 server-kersuaar systemd[1]: Finished PostgreSQL RDBMS.
root@server-kersuaar:~#
```

Configure/setup postgresQL:

1) To verify the installation, try connecting yourself to the user 'postgres' from the root user.

```
# su - postgres
```

1) Now connect to PostgreSQL from the postgres user:

```
$ psql
```

2) And create a new user with your UGA login for the name.
For e.g:

```
$ CREATE USER Your_UGA_Login password 'temporary_Password';
```

3) Now create a data base with your user as the owner,
(name it after your login)
For e.g:

```
$ CREATE DATABASE data_base_name WITH OWNER = user_name;
```

Now, execute the provided command and verify if the database with your user as the owner has been successfully created, as illustrated below:

```
$ psql -l
```

```
postgres@server-kersuaar:~$ psql -l
                                List of databases
  Name      | Owner   | Encoding | Collate  | Ctype    | Access privileges
-----+-----+-----+-----+-----+-----
 kersuaar_base | kersuaar | UTF8      | en_US.UTF-8 | en_US.UTF-8 |
 postgres     | postgres | UTF8      | en_US.UTF-8 | en_US.UTF-8 |
 template0    | postgres | UTF8      | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +
              |          |           |             |             | postgres=CTc/postgres
 template1    | postgres | UTF8      | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +
              |          |           |             |             | postgres=CTc/postgres
(4 rows)
```

Figure 18: List of databases present in the VM

4) And create a simple table inside your database:

```
$ CREATE TABLE table_name ([name] [varchar/Date/...], ...)
```

5) Insert some row into this table:

```
$ INSERT INTO table_name VALUES ('value1', ...)
```

To verify the data entered in the table, you can interrogate the table and see:

```
kersuaar_base=# select * from simple_table_test1;
 login
-----
 colinev
 vivarat
 scheerc
(3 rows)
```

Figure 19: result of a sql request from the virtual machine

Setup accessibility to postgresQL:

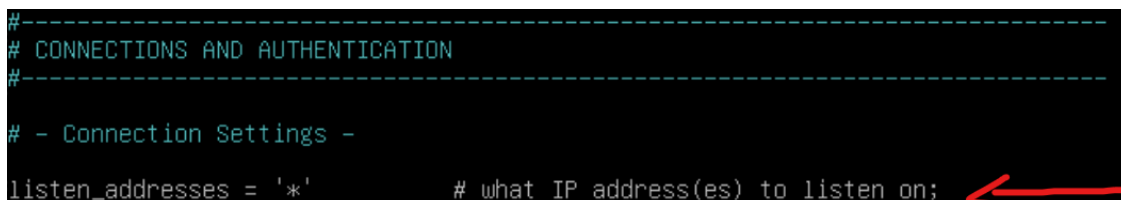
Now configure PostgreSQL to be accessible from your station.
You will need to modify 2 configurations files and restart your PostgreSQL server.

Firstly place yourself in the postgresql folder:

```
# cd /etc/postgresql/13/main/
```

And search for `postgresql.conf` and change the line 'listen_addresses' to:
(If it is currently commented, please uncomment it.)

```
listen_addresses = '*'
```

A screenshot of a terminal window showing the configuration file `postgresql.conf`. The file is dark-themed with green and white text. It shows the section `# CONNECTIONS AND AUTHENTICATION` and the line `listen_addresses = '*'` which is commented out with a `#`. A red arrow points to the `*` in the `listen_addresses` line.

```
#-----  
# CONNECTIONS AND AUTHENTICATION  
#-----  
# - Connection Settings -  
listen_addresses = '*'          # what IP address(es) to listen on;
```

Figure 20: `postgresql.conf` file

Additionally, please uncomment the line and update "password_encryption" from "md5" to "scram-sha-256".

!! Quick reminder to save before exiting the file !!

```
password_encryption = scram-sha-256
```

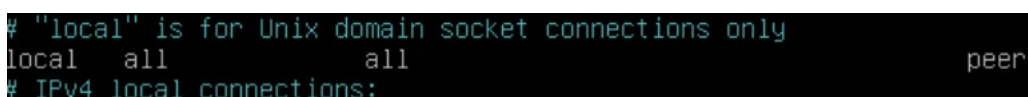
A screenshot of a terminal window showing the `password_encryption` line in `postgresql.conf`. The line is `password_encryption = scram-sha-256` and is commented out with a `#`. The comment `# md5 or scram-sha-256` is visible to the right of the line.

```
password_encryption = scram-sha-256          # md5 or scram-sha-256
```

Figure 21: `password_encryption` line from `postgresql.conf`

And now search for `pg_hba.conf` add a new entry for the remote connection at **#IPv4 local connections:**

```
host all all 0.0.0.0/0 scram-sha-256
```

A screenshot of a terminal window showing the `pg_hba.conf` file. It shows the section `# "local" is for Unix domain socket connections only` and the line `local all all peer`. Below that, it shows the section `# IPv4 local connections:`.

```
# "local" is for Unix domain socket connections only  
local all all peer  
# IPv4 local connections:
```

Figure 22: `pg_hba.conf` file

Make sure to change the method name from "md5" to "scram-sha-256" wherever it is used since we use SCRAM-SHA-256 authentication.

The next step is to define your new password to hash it with the scram-sha-256 encryption:

```
$ ALTER USER Your_UGA_Login password 'new_temporary_Password' ;
```

And check if your password is properly encrypted.

You can verify it by listing the system table and checking the encryption applied to it by typing:

```
# select username, passwd from pg_shadow;
```

```
postgres=# select username,passwd from pg_shadow;
username |                                passwd
-----+-----
postgres |
kersuaar | SCRAM-SHA-256$4096:BRUiJnPrbuY6C7qtGB9jSQ==$LN70F5Kk1XFIKUPnY+zGH1Qxf8sIWXgbbD1utCsvgXP0=:SAK2eVKC6YEzS0hddJX/ztcOnT8
Yu+U1yHo8uGWEU20=
(2 rows)

(END)_
```

Figure 23: result of the request of pg_shadow table

If you also want to see all the permissions for each user, you can type:

```
# select * from pg_shadow;
```

```
postgres=# select * from pg_shadow;
username | usesysid | usecreatedb | usesuper | userepl | usebypassrls |          passwd          | valuntil | useconfig
-----+-----+-----+-----+-----+-----+-----+-----+-----
postgres |      10 | t           | t        | t       | t           |          |          |
kersuaar |    16384 | t           | t        | f       | f           | SCRAM-SHA-256$4096:BRUiJnPrbuY6C7qtGB9jSQ==$LN70F5Kk1XFIKUPnY+zGH1Qxf8sIWXgbbD1utCsvgXP0=:SAK2eVKC6YEzS0hddJX/ztcOnT8Yu+U1yHo8uGWEU20= |          |
(2 rows)

(END)_
```

Figure 24: result of the request * of pg_shadow table

Verify that you can connect to postgresQL from your Linux station:

```
$ psql -h localhost postgres
```

And make a permanent password to secure the access with this command:

```
# \password username
Enter new password: <permanent-password>
```

The last step is to access the data from your Linux station:

Once we can access postgresql from another station you can execute the same command.

```
kersuaar@pc-dg-025-03:~$ psql -h localhost kersuaar_base
Password for user kersuaar:
psql (13.10 (Debian 13.10-0+deb11u1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.

kersuaar_base=# select * from simple_table_test1 ;
 login
-----
 colinev
 vivarat
 scheerc
(3 rows)

kersuaar_base=#
```

Figure 25: result of a sql request from a station

Now restart your PostgreSQL server to finish the process.

```
# systemctl restart postgresql
```

And check his status:

```
# systemctl status postgresql
```

PHP

we will proceed with the installation of PHP to enable the visualization and development of web pages from your Linux station.

Installing postgresQL package:

!/ Below is the documentation if you need further explanation !/

[PHP: Installation on Unix systems - Manual](#)

[PHP tutorial thread](#)

To begin with, install the PHP package (for Apache 2) by executing the command below as the root user:

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

2) Restart apache2 once php is installed:

```
# systemctl restart apache2
```

3) To verify if the installation went well place yourself at /var/www/html:

```
# cd /var/www/html
```

Subsequently, create a new file named "info.php" and write a script in it:

```
# cat > info.php
(new text)
<?php
phpinfo();
phpinfo(INFO_MODULES);
?>
```

Save it and in your Linux station open the following page on your browser:

```
http://localhost:8080/info.php
```


You should arrive on this page:

localhost:8080/info.php

efox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back!

Refresh Firefox...

PHP Version 7.4.33



System	Linux server-kersuaar 5.10.0-22-amd64 #1 SMP Debian 5.10.178-3 (2023-04-22) x86_64
Build Date	Feb 22 2023 20:07:47
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/apache2
Loaded Configuration File	/etc/php/7.4/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/apache2/conf.d
Additional .ini files parsed	/etc/php/7.4/apache2/conf.d/10-opcache.ini, /etc/php/7.4/apache2/conf.d/10-pdo.ini, /etc/php/7.4/apache2/conf.d/20-calendar.ini, /etc/php/7.4/apache2/conf.d/20-ctype.ini, /etc/php/7.4/apache2/conf.d/20-exif.ini, /etc/php/7.4/apache2/conf.d/20-ffi.ini, /etc/php/7.4/apache2/conf.d/20-fileinfo.ini, /etc/php/7.4/apache2/conf.d/20-ftp.ini, /etc/php/7.4/apache2/conf.d/20-gettext.ini, /etc/php/7.4/apache2/conf.d/20-iconv.ini, /etc/php/7.4/apache2/conf.d/20-ldap.ini, /etc/php/7.4/apache2/conf.d/20-ldap20-openssl.ini, /etc/php/7.4/apache2/conf.d/20-phar.ini, /etc/php/7.4/apache2/conf.d/20-posix.ini, /etc/php/7.4/apache2/conf.d/20-readline.ini, /etc/php/7.4/apache2/conf.d/20-shmop.ini, /etc/php/7.4/apache2/conf.d/20-sockets.ini, /etc/php/7.4/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.4/apache2/conf.d/20-sysvsem.ini, /etc/php/7.4/apache2/conf.d/20-sysvshm.ini, /etc/php/7.4/apache2/conf.d/20-tokenizer.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902.NTS
PHP Extension Build	API20190902.NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	available, disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, convert.iconv.*

This program makes use of the Zend Scripting Language Engine:
Zend Engine v3.4.0, Copyright (c) Zend Technologies
with Zend OPcache v7.4.33, Copyright (c), by Zend Technologies

zend engine

Machine View

GNU nano 5.4

<?php
phpinfo();
?>

root@server-kersuaar: /var/www/html#

Figure 26: landing page of info.php file

PhpPgAdmin

Now, we will install phpPgAdmin to facilitate the usage of your PostgreSQL databases through a web interface.

Installing PhpPgAdmin package:

Switch back to the root user of your virtual machine and proceed with the installation of the PhpPgAdmin package:

```
# apt install phpPgAdmin
```

Setting up PhpPgAdmin:

To access the web site of PhpPgAdmin, we will need to edit the phppgadmin.file:

```
# nano /etc/apache2/conf-enabled/phppgadmin.conf
```

And remove the “require local host” line.

!! Quick reminder to save before exiting the file !!

Now try to access the web page from your Linux station:

<http://localhost:8080/phppgadmin>

Now connect to postgres or your username (Login) and select your database
Then perform a request such as SELECT (...):

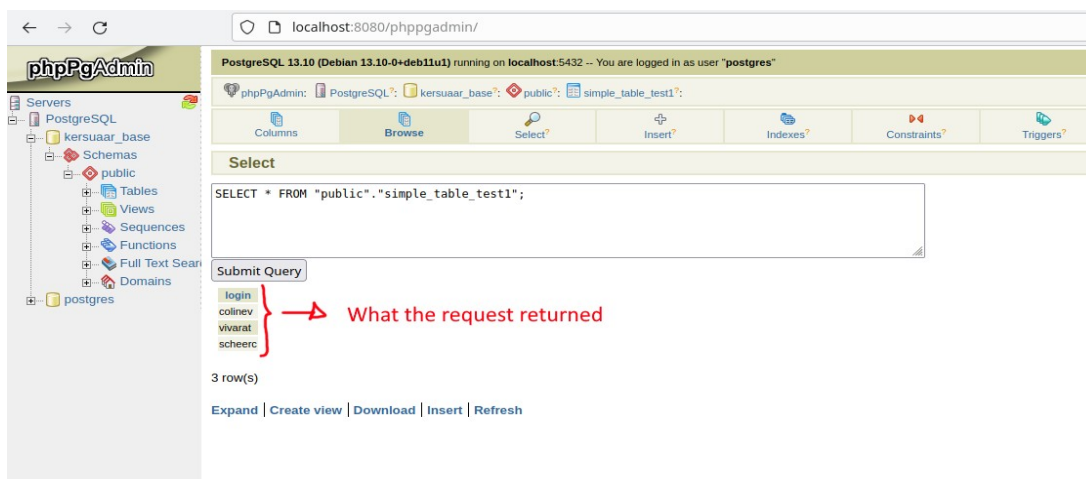


Figure 27: PhpPgAdmin request on database

Additional PHP:

We will now visualize a file on your Linux station browser to see the characteristics of your virtual machine and the status of your apache2, postgresSQL, php services.

To transfer the file "page_sae_S2.03.php" to your virtual machine, type the following command in your root user:

```
# scp -p 2222 -- /users/info/www/intranet/enseignements/S2.03/page_sae_S2.03.php /var/www/html
```

The '--' command is to separate the source and destination file/folder names.

Prior to accessing the page, execute the provided command.

```
# /sbin/blkid
```

```
root@server-kersuaar:~# /sbin/blkid
/dev/sda1: UUID="45e891c2-f771-4b1b-8ac6-61321aa58f95" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="593fdeeb-01"
/dev/sda5: UUID="42c8c757-7bf2-4d10-83d1-522c1c25e53a" TYPE="swap" PARTUUID="593fdeeb-05"
```

Figure 28: result of /sbin/blkid command

And you should get this page showing the characteristics of your VM:

```
Bonjour
Je suis www-data
Qui est connecté ?
kersuaar tty1      May 31 11:15
Mes disques sont
Mes interfaces
1: lo: mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid lft forever preferred lft forever
    inet6 ::1/128 scope host
        valid lft forever preferred lft forever
2: enp0s2: mtu 1500 qdisc pfifo fast 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 85391sec preferred lft 85391sec
    inet6 fe80::5054:fff:fe12:3456/64 scope site dynamic mngtppadr
        valid lft 86389sec preferred lft 14389sec
    inet6 fe80::5054:fff:fe12:3456/64 scope link
        valid lft forever preferred lft forever

My apache install is
ii apache2                2.4.56-1-deb11u2      amd64      Apache HTTP Server
ii apache2-bin             2.4.56-1-deb11u2      amd64
ii apache2-data            2.4.56-1-deb11u2      all
ii apache2-utils           2.4.56-1-deb11u2      amd64
ii libapache2-mod-php      2:7.4+76              all
ii libapache2-mod-php7.4  7.4.33-1-deb11u3      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; vendor preset: enabled)
   Active: active (running) since Wed 2023-05-31 11:15:29 CEST; 16min ago
     Docs: https://httpd.apache.org/docs/2.4/
    Main PID: 460 (apache2)
      Tasks: 13 (limit: 4661)
     Memory: 49.0M
        CPU: 73ms
    CGroup: /system.slice/apache2.service
            ┌─460 /usr/sbin/apache2 -k start
            ├─455 /usr/sbin/apache2 -k start
            ├─456 /usr/sbin/apache2 -k start
            ├─457 /usr/sbin/apache2 -k start
            ├─459 /usr/sbin/apache2 -k start
            ├─459 /usr/sbin/apache2 -k start
            ├─459 /usr/sbin/apache2 -k start
            ├─461 /usr/sbin/apache2 -k start
            ├─468 /usr/sbin/apache2 -k start
            ├─468 /usr/sbin/apache2 -k start
            ├─461 /usr/sbin/apache2 -k start
            └─462 /usr/sbin/apache2 -k start
```

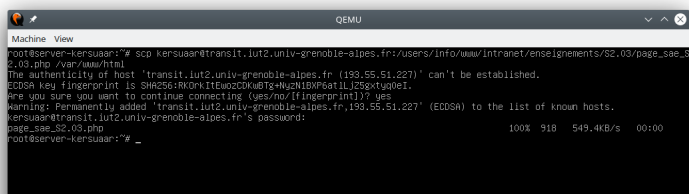


Figure 29: Characteristics of the virtual machine

SSH Key access user [Login]

If you are interested in further exploring how to enhance the security and simplify access to your VM server, you can follow these additional steps.

First, generate on your Linux station a ssh key with this command:

```
$ ssh-keygen
```

Press enter for the passphrase if you don't want to have one (it's better for security purpose).

```
kersuaar@transit:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/users/info/etu-1a/kersuaar/.ssh/id_rsa):
/users/info/etu-1a/kersuaar/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /users/info/etu-1a/kersuaar/.ssh/id_rsa
Your public key has been saved in /users/info/etu-1a/kersuaar/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:+YKT0BTh88FVEde7HqLC1qsIljREBXI0EMNRfxZgqRw kersuaar@transit
The key's randomart image is:
+---[RSA 3072]-----+
|  .**B==0...+0.. |
|  .E++ .. . . |
|  ..*.00 . |
|  .= 000 . |
|  .0. S . |
|  ..00 . . 0 |
|  ++ 0 0 . 0 . |
|  . ...= 0 . |
|  ...0.. |
+-----[SHA256]-----+
```

Figure 30: result of ssh-keygen

Then copy the id of the public key on your virtual machine to setup authorized_keys on it:

```
$ ssh-copy-id -p 2222 UGA_login@localhost
```

Ensure that the virtual machine has the same SSH key in the ".ssh/authorized_keys" file.

Otherwise, you will be unable to establish a connection.

And you can now establish a connection to your virtual machine without requiring a password.

SSH Key access root

Alternatively, you can also enable passwordless access for the root user of your virtual machine by copying the key from the 'authorized_keys' file of the Uga_login user to the 'authorized_keys' of the root user.

```
# cp ~Uga_login/.ssh/authorized_keys .ssh/authorized_keys
```

```
cp ~kersuaar/.ssh/authorized_keys .ssh/authorized_keys
```

Figure 31: copying command from root user

Verify if the ssh key is the same as the Login_user one:

```
root@server-kersuaar:~# cat .ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCAQCWqNH00y8YE06QbXuqtqR6d2rFmJ43bskRjVSPk7XN/013VzhHsmtkLMzjCgRai10+ui8Gy2MVsXrhKGHC4ApML9Im
yAHJn0sQczTbvTKFrDBq2Brx13VfMI87MyEU0c7MPR2Ww/greAdS7BYc70Nruh1/vS5gy82DkfUnR3jAysUgqx1NmVw89Z3vt6Mzw748dL28Cf3CZVaUj4SHVA51B0Jx
pJHpumamU4FbAuztUbsUrDtU3vEs8pC2bGZCfGf01ILm1WxcawMvJkDwh29g1tGJp08T9a1Cr0TTIV1uVcsUQVh4gtcT0cLUy4fCwJPd5v1SQqbwIqXu6vUB4sReqt
fTITuMoxVNAV8bt5Lr7t6xW/z3xhIAMk/wpcdD4tDi7u0X7fXE1vF+iJxCw1LSj7ugz7+CngX2cHcFeS7CgQHuhgmQz2758L5doo06B7cc70BXqvpnyEUh/20L0AQ31KH
dM1tJhktC0Bmq2DqM1gnjCcq53jinYNO+pkRXqvt0vx7wL02JaMzII52sD0716Nz/7UkL34uDDTzzIbCsvRtmBhRrYt0W0/61yo2Ka3x93jk71Iv26SS5PIFcE39oF9
ML6CqSh7N6K9WtWRCgXGiULYvFt7AT7SPXrgr8u0Ck56NtEN20pbLJQpoDEgu24urk3+BjQtX3S3v4i4Dw== kersuaar@pc-dg-025-15
```

Figure 32: ssh key of the login_user

Lastly, use the following command to check the available storage space:

```
# df -h
```

```
root@server-kersuaar:~# df -H
Filesystem      Size  Used Avail Use% Mounted on
udev            2.1G   0    2.1G   0% /dev
tmpfs           412M   500k  411M   1% /run
/dev/sda1       3.2G   1.5G   1.5G  50% /
tmpfs           2.1G   17k   2.1G   1% /dev/shm
tmpfs           5.3M   0    5.3M   0% /run/lock
tmpfs           412M   0    412M   0% /run/user/1000
```

Figure 33: Storage characteristics

To conclude

Security and evolutions:

This guide provides a basic understanding of setting up a virtual machine, but there are many additional tasks to consider in order to enhance security and control access.

For instance, configuring user privileges on databases and defining access permissions for specific users.

Additionally, it is advisable to add a passphrase to your SSH keys to enhance security in case the keys are compromised or accessed by unauthorized individuals.

Thank you very much for reading this guide, hope it proves to be helpful to you.