IDEO-LAB

main Setup
Admin Django architecture

Guillaume oneill

4th December 2025

Version 6.8

agenda

- ssh password activation
- execute base linux installation : install_script.sh
- Installation de virtualenv : install_venv.sh

Basic staging/Production Setup

- **Linux Ubuntu Basic System** :
  - Basis (apt-get Install)
  - Imaging
  - Memcache / RabbitMQ
- **Python and Django** :
  - Easy Install / PIP / VirtualEnvs (Wrapper)
  - Python and Django Addon
- **Nginx Installation** :
  - Nginx setup (enabled and available)
  - Gunicorn setup
- **Supervisor** :
- **PostGreSql** Server :
  - Create your Database
  - Dump / reload your Database
  - Create Grant Role
- **Java / Elastic Search** :
- **Your Core code** :
  - SSH key and GIT
  - Unfuddle SSH key register
  - Clone Core code
- **CronTab** (Cron Settings)

AWS EC2 SSH

Initial SSH AND PASSWORD SETTINGS

- USE special ssh given by aws connect with pem key
- cd /etc/ssh
- vi sshd_config :
  - change psw authentication allowed to yes
- cd /sshd_config.d
  - update 60-cloudimg-settings.conf
  - update authentication by password
- sudo /etc/init.d/ssh restart
- update user « ubuntu » password :
  - sudo passwd ubuntu

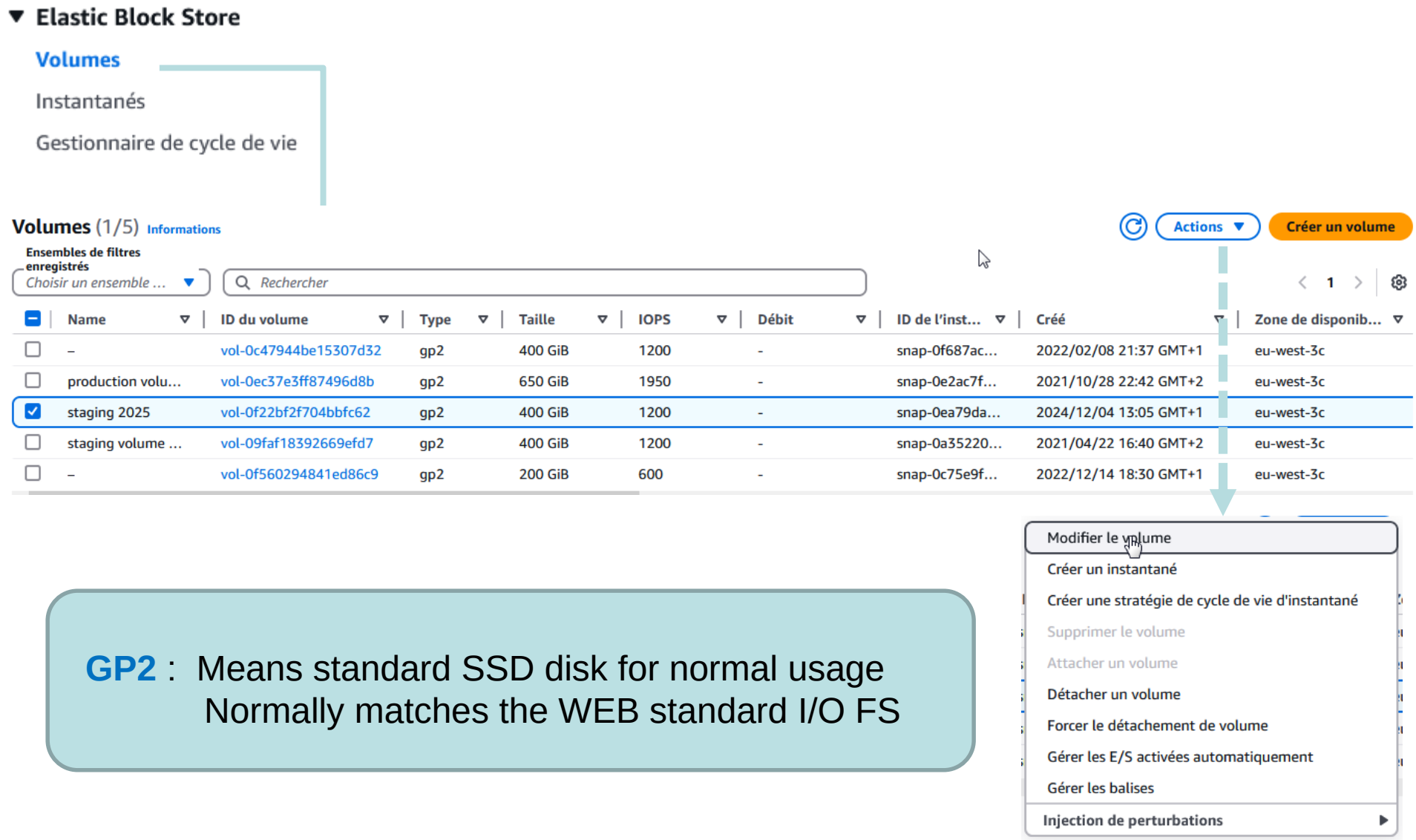
EXTEND EBS
WHEN REQUIRED
df –h : 100%

AWS EBS EXTEND RULES

- 1 REALIZE AN EBS SNAPSHOT
- 2 EBS VOLUMES EXTENDS

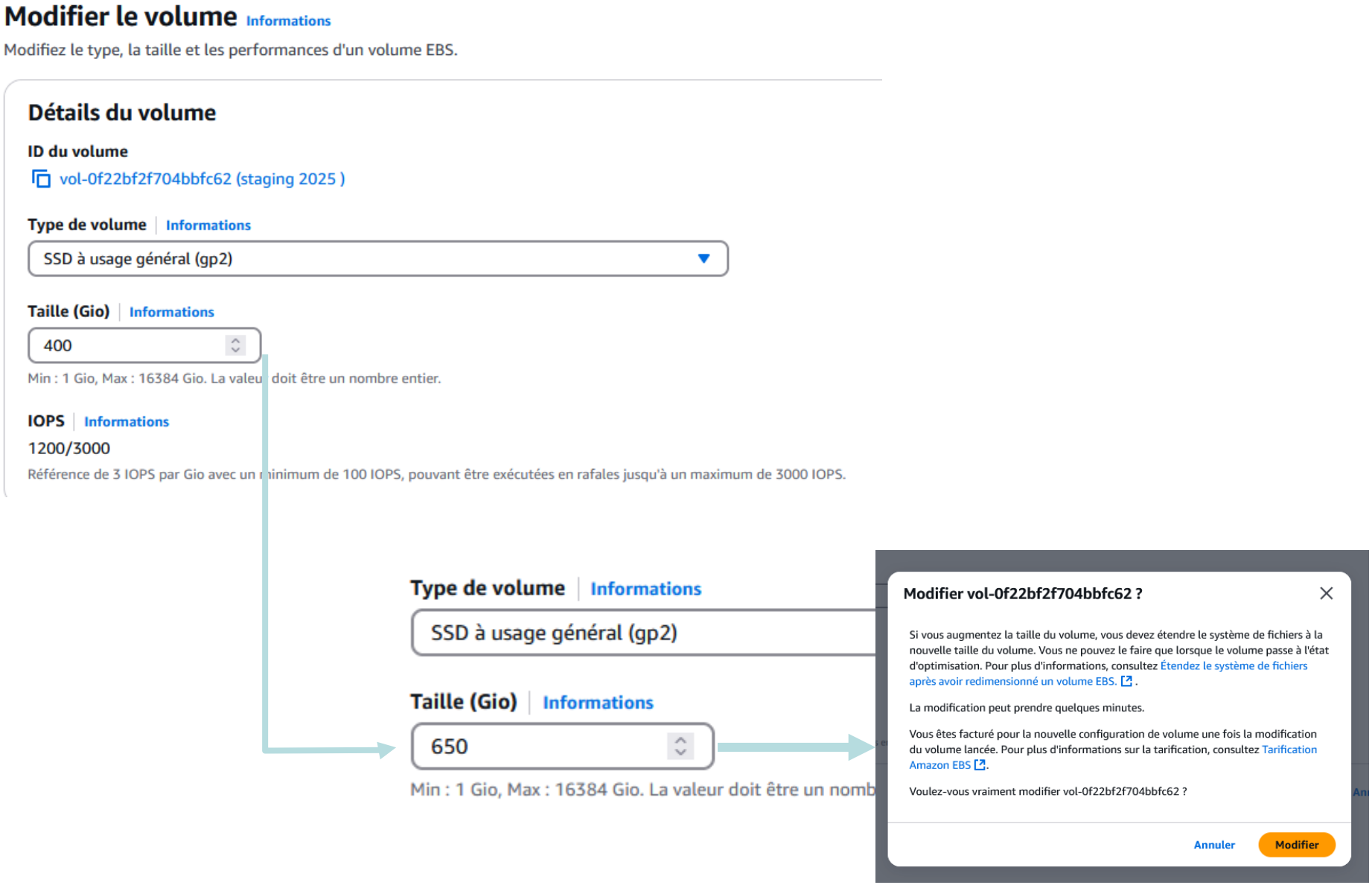
AWS EBS EXTEND

sample of extend from 400 to 650 Gb – gp2



GP2 : Means standard SSD disk for normal usage
Normally matches the WEB standard I/O FS

MODIFY VOLUME SIZE



EXTEND AWS EBS

- **STANDARD LINUX PROCEDURE** :
  - Check current root disk : df -h
  - Verify New attached Volume size
    - **lsblk**
  - Verify Partition (1) :
    - sudo fdisk -l
    - sudo **growpart** /dev/nvme0n1 **1** (*all entire volume*)
  - Extend current Filesystem:
    - df –hT
    - sudo resize2fs /etc/nvme0n1p1 (*Resize partition 1*)
    - df –h
    - **Enjoy new disk space** !

EXTEND PROCEDURES

fdisk -l

```
Disk /dev/nvme0n1: 400 GiB, 429496729600 bytes, 838860800 sectors
Disk model: Amazon Elastic Block Store
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: gpt
Disk identifier: 3A210580-6416-44A3-B0A2-BFE67042CD6F

Device           Start      End  Sectors  Size Type
/dev/nvme0n1p1  227328 16777182 16549855  7.9G Linux filesystem
/dev/nvme0n1p14   2048    10239     8192    4M BIOS boot
/dev/nvme0n1p15  10240   227327   217088  106M EFI System
```

growpart /dev/nvme0n1 1

```
ubuntu@ip-172-31-35-14:~$ sudo growpart /dev/nvme0n1 1
CHANGED: partition=1 start=227328 old: size=16549855 end=16777183 new: size=838633439 end=8388607
ubuntu@ip-172-31-35-14:~$ df -h
Filesystem       Size  Used Avail Use% Mounted on
/dev/root        7.6G  4.9G  2.7G  65% /
tmpfs             16G     0   16G   0% /dev/shm
tmpfs            6.2G  912K  6.2G   1% /run
tmpfs            5.0M     0  5.0M   0% /run/lock
efivarfs         128K  3.8K  120K   4% /sys/firmware/efi/efivars
/dev/nvme0n1p15  105M  6.1M   99M   6% /boot/efi
tmpfs            3.1G  4.0K  3.1G   1% /run/user/1000
```

resize2fs /dev/nvme0n1p1

```
ubuntu@ip-172-31-35-14:~$ sudo resize2fs  /dev/nvme0n1p1
resize2fs 1.46.5 (30-Dec-2021)
Filesystem at /dev/nvme0n1p1 is mounted on /; on-line resizing required
old_desc_blocks = 1, new_desc_blocks = 50
The filesystem on /dev/nvme0n1p1 is now 104829179 (4k) blocks long.
```

PRA

Fail Over Process

- **Original server** :
  - Every day:
    - AWS Snapshot
    - send to other Datacenter
  - Every Morning (8:00AM):
    - PG Dump (.tar) of database_name Database
    - SCP to Target server

Recover Process on Target server

- Target server :
  - Every Fail Over scenario:
    - PG Restore with last Database Dump
    - Update « django site » with FailOver Domain (*if required*)
    - Git pull –v , Master Branch
    - Restart Gunicorn, Nginx, PostGresql
  - Check :
    - Check CertBot Pem key validity (*SSL Key for Https*)
    - Check O/S Version compatibility (*22.04*)
    - Check server-sides (*Nginx, supervisor, Redis, Gunicorn, Postgresql, ..*)
    - Check Django & Python addon
    - Check free memory & TOP review

DAILY DUMP OF DATABASES

- PG DUMP
  - every Morning after Batch execution
  - send Dump to Datarecover Center
  - Restore Dump on Data recover center

Kernel Optimization

Base Optimization

- 1) Kernel compilation
- 2) Swappiness and virtual Memory
- 3) **Huge memory Pages**
- 4) Tune Kernel Memory Page
- 5) Monitor and Diagnostic
- 6) Application Optimization
- 7) Up to date Kernel and Patch

1. Kernel Compilation

Recompiling the Linux kernel with specific options tailored to your hardware and use case can significantly optimize memory usage.

- **Install Kernel Source:**

  ```bash
  sudo apt-get install build-essential libncurses-dev bison flex libssl-dev libelf-de
  sudo apt-get install linux-source
  ```

- **Configuration:** Use `make menuconfig` or `make nconfig` to configure the kernel. Key memory-related options:
  - Disable unnecessary modules and drivers.
  - Enable support for your specific hardware.
  - Adjust swap-related options like **SWAPNESS**.
  - Enable huge pages for memory-intensive applications.

  Example memory-related settings:

  - **CONFIG_SLAB_MERGE_DEFAULT**: Reduce memory fragmentation.
  - **CONFIG_TRANSPARENT_HUGEPAGE**: Useful for applications requiring large memory

2. Swappiness and Virtual Memory Tuning

- **Adjust Swappiness:** Swappiness controls how aggressively the kernel swaps memory to disk. Lower values favor RAM usage.

```bash
echo "vm.swappiness=10" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
```

- **Disable Overcommit:** Avoid out-of-memory (OOM) situations by controlling memory overcommitment.

```bash
echo "vm.overcommit_memory=2" | sudo tee -a /etc/sysctl.conf
echo "vm.overcommit_ratio=80" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
```

3. Huge Pages and Transparent Huge Pages

- Enable Huge Pages:

```bash
echo "vm.nr_hugepages=128" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
```

- Check Transparent Huge Pages (THP):

```bash
cat /sys/kernel/mm/transparent_hugepage/enabled
```

Enable THP for applications like databases:

```bash
echo "always" | sudo tee /sys/kernel/mm/transparent_hugepage/enabled
```

4. Kernel Parameters for Memory

- Modify `/etc/default/grub` to adjust kernel parameters:

```bash
sudo nano /etc/default/grub
```

Add parameters to `GRUB_CMDLINE_LINUX_DEFAULT`, such as:

- `mem=4G`: Limit the kernel to use only 4 GB of RAM.
- `noatime`: Reduce disk I/O by disabling access time updates.
- `zswap.enabled=1`: Enable Zswap for compressed caching.
- `vm.dirty_ratio` and `vm.dirty_background_ratio`: Adjust how much dirty memory is cached before writing to disk.

Update GRUB and reboot:

```bash
sudo update-grub
sudo reboot
```

5. Monitoring and Diagnostics

- Use tools to monitor memory performance and identify bottlenecks:
  - `top` / `htop` : Real-time monitoring.
  - `vmstat` : Analyze memory usage patterns.
  - `free -h` : View memory and swap usage.
  - `perf` : Kernel performance analysis.
- **Example: Memory Monitoring Script**

bash

Copier le code

```bash
while true; do
    free -m
    sleep 5
done
```

6. Specific Application Optimization

- For memory-intensive applications (e.g., databases, high-performance computing):
  - Use `numactl` for NUMA systems to bind memory to specific CPUs.
  - Allocate memory with `mmap` in user space to reduce kernel overhead.

---

7. Up-to-Date Kernel

Ensure your kernel is up-to-date for the latest performance improvements and bug fixes:

bash Copier le code

```
sudo apt update
sudo apt upgrade
sudo apt install linux-generic-hwe-$(lsb_release -rs)
```

By carefully tailoring these configurations, you can optimize your Ubuntu system's kernel for memory-intensive or low-resource scenarios.

High memory Optimization

What Is High Memory?

In a 32-bit Linux system:

- **Low Memory:** Directly mapped into the kernel address space, allowing quick access.
- **High Memory:** Not directly mapped; access requires additional mechanisms (like temporary kernel mappings or page table entries).

In 64-bit systems, the distinction between high and low memory is less relevant because the addressable memory space is much larger.

Memory Optimization

- 1. Configure Kernel for High Memory Access
- 2. Use High Memory Efficiently
- 3. Transparent Huge Pages (THP)
- 4. Huge Pages
- 5. Tune Kernel Memory Parameters
- 6. Optimize I/O Operations
- 7. Application-Level Optimizations
- 8. Monitor High Memory Usage
- 9. Advanced: Use ZSWAP and ZRAM

1. Configure Kernel for High Memory Access

The kernel includes features to optimize high memory usage:

- **Enable High Memory Support:** If you have more than 4 GB of RAM on a 32-bit system, ensure the kernel is configured to support high memory:

  bash Copier le code

  ```
  CONFIG_HIGHMEM=y
  CONFIG_HIGHMEM4G=y
  ```

  For systems with over 4 GB, enable:

  bash Copier le code

  ```
  CONFIG_HIGHMEM64G=y
  ```

- **Recompile Kernel for High Memory Optimization:**

  bash Copier le code

  ```
  make menuconfig
  ```

2. Use High Memory Efficiently

Accessing high memory requires temporary mappings, which add overhead. Optimize usage by:

- **Minimize High Memory Access:** Applications and kernel modules can be designed to keep frequently accessed data in low memory.
- **Kernel Bounce Buffers:** For high-memory I/O, use bounce buffers to temporarily hold data in low memory during operations.
- **Reduce Copying:** Use **DMA (Direct Memory Access)** for large data transfers, bypassing the CPU and reducing overhead.

---

3. Transparent Huge Pages (THP)

Transparent Huge Pages reduce overhead by allocating larger memory pages (e.g., 2 MB instead of 4 KB), which are especially beneficial in high-memory scenarios.

- **Check THP Status:**

```bash
cat /sys/kernel/mm/transparent_hugepage/enabled
```

4. Huge Pages

Huge Pages are manually managed and can optimize high memory usage for memory-intensive applications like databases or in-memory caches.

- **Enable Huge Pages:**

```bash
echo "vm.nr_hugepages=512" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
```

- **Allocate Huge Pages:** Pin huge pages in high memory to optimize large memory allocations:

```bash
sudo sysctl -w vm.hugetlb_shm_group=1000
```

- **Verify Huge Page Usage:**

```bash
cat /proc/meminfo | grep HugePages
```

5. Tune Kernel Memory Parameters

- **Reduce Page Cache Pressure:** Tune the kernel to retain more memory in the page cache:

```bash
echo "vm.vfs_cache_pressure=50" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
```

- **Balance Swap and RAM:** Optimize swapping behavior to prioritize RAM over high-memory swap areas:

```bash
echo "vm.swappiness=10" | sudo tee -a /etc/sysctl.conf
```

6. Optimize I/O Operations

High memory optimizations often coincide with I/O tuning:

- Use **Direct I/O** (O_DIRECT flag) to bypass the page cache for large sequential reads/writes.
- Enable asynchronous I/O (AIO) for overlapping I/O operations.

---

7. Application-Level Optimizations

- **Use High-Memory-Friendly Tools:** For databases or in-memory systems (e.g., Redis, MySQL), configure the memory allocator (like jemalloc or tcmalloc) to handle high memory efficiently.
- **Explicit Memory Pinning:** Use tools like `mlock` to pin critical memory regions in low memory:

```c
#include <sys/mman.h>
mlock(address, size);
```

8. Monitor High Memory Usage

Monitor high memory to ensure optimizations are effective:

- **Check High Memory Stats:**

  bash Copier le code

  ```
  cat /proc/meminfo | grep High
  ```

  Look at `HighTotal` and `HighFree` to assess usage.

- **Debug Kernel Memory Allocation:** Use `slabtop` to monitor slab allocation and adjust kernel memory management accordingly:

  bash Copier le code

  ```
  slabtop
  ```

9. Advanced: Use ZSWAP and ZRAM

- **ZSWAP**: Compress pages in RAM before swapping to disk, optimizing memory for high-memory systems:

bash Copier le code

```
echo 1 > /sys/module/zswap/parameters/enabled
```

- **ZRAM**: Create compressed RAM disks to reduce swapping:

bash Copier le code

```
sudo modprobe zram
echo 1G > /sys/block/zram0/disksize
mkswap /dev/zram0
swapon /dev/zram0
```

Docker

Docker Installation

sudo apt-get update & apt-get upgrade

sudo apt-get install python3-gi

sudo apt-get install pkg-config libcairo2-dev gcc python3-dev libgirepository1.0-dev

sudo apt-get install python3-setuptools python3-pip

pip install gobject PyGObject

sudo apt update

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable »

sudo apt update

sudo apt-cache policy docker-ce

sudo apt install docker-ce

sudo systemctl status docker

sudo docker build -t **project/ubuntu_base** .

Docker Components

- https://hub.docker.com/
- Create a new AMI common for any new server :
  - **Fixed and common Parts :**
    - **Main Linux components**
    - Nginx
    - supervisor
    - Virtualenvs
    - Python 3.12
    - PostGresql
    - Redis
    - Java
    - ElasticSearch
    - certbot
    - Common Python & Django addon
      - mandatory :
        - » gunicorn
        - » pg extra
        - » django 5
      - common addon for a Django project

Variable Components

- https://hub.docker.com/
- Create a new AMI common for any new server :
  - **Variable components :**
    - Restore PostGresql Database
    - GIT Clone
    - git checkout on xxxxBranchxxxx
    - Local settings.py
      - DataBase Name
      - Local country settings
      - Local server settings
      - Logging and Local base directory
      - Cache settings
    - Django Specific addon
      - See Addon Packager (poetry or other ....)

Linux Ubuntu 24.04

Update from Python to Python 3.12

- Python3 –V
- sudo apt-get install python3.12
- sudo update-alternatives --install /usr/bin/python3 python3 /usr/bin/python3.12
- sudo update-alternatives --install /usr/bin/python3 python3 /usr/bin/python3.12
- sudo update-alternatives --config python3
- Python3 –V

- sudo apt update
- sudo apt-get install python3.12-distutils
- sudo apt install software-properties-common
- sudo add-apt-repository ppa:deadsnakes/ppa
- sudo apt-get install python3.12 python3.12-dev
- mkvirtualenv -p /usr/bin/python3.12 venv_django5

Amazon EC2 Linux Basic System

- Apt-get update & apt-get upgrade
- **Ubuntu Setup for your own Django project Tasks** :
  - apt-get install python 3.12 (if missing)
  - apt-get install make
  - apt-get install gcc
  - apt-get install sudo
  - Apt-get install python3-pip
  - apt-get install python3-setuptools
  - apt-get install python-dev and python3-dev python3.8-dev
  - apt-get install build-essential
  - apt-get install yum curl
  - apt-get install libjpeg-progs libjpeg-dev
  - apt-get install libjpeg8 libjpeg8-dev
  - apt-get install zlib1g-dev
  - apt-get install binutils libproj-dev gdal-bin
  - apt-get install libxml2-dev
  - apt-get install libxslt1-dev
  - apt-get install libcurl3-dev
  - apt-get install python-crypto
  - Apt-get install python3-pil
  - apt-get install libgeoip1
  - apt-get install python3-gdal
  - Apt-get install yui-compressor
  - Apt-get install python3-pip
  - Apt-get install software-properties-common
  - apt-get install python3-full
  - apt-get install libdbd-pg-perl
  - apt-get install python3.10-venv
  - apt-get install python3.10-distutils

INSTALLATION automation SCRIPT

- Base Linux system Libraries
- gevent
- docker
- supervisor
- postgresql 18
- redis
- certbot
- memcached

Linux system Addon

Memcache, Rabbitmq and Redis

- apt-get install RabbitMQ-server
  - **RabbitMQ-server**

- Apt-get install **memcached**
  - vi /etc/memcached.conf
  - service memcached restart
  - memcached –d –m 2048 –u root
  - ps -eaf | grep memcached
  - service memcached restart

- **Redis server**:
  - see installation details on next page

Redis server on Ubuntu 24.04

```
$ sudo apt update
```

Installez ensuite Redis en saisissant ce qui suit :

```
$ sudo apt install redis-server
```

Copy

Cette opération permettra le téléchargement et l'installation de Redis et de ses dépendances. Suite à cela, vous devez effectuer un changement de configuration important dans le fichier de configuration Redis, qui a été automatiquement généré lors de l'installation.

Ouvrez ce nouveau fichier avec votre éditeur de texte préféré :

```
$ sudo nano /etc/redis/redis.conf
```

Dans le fichier, trouvez la directive `supervised`. Cette directive vous permet de déclarer un système init qui gérera Redis en tant que service et vous donnera plus de contrôle sur son fonctionnement. La directive `supervised` est configurée sur `no` par défaut. Etant donné vous exécutez Ubuntu, qui utilise le système init systemd, remplacez cette valeur par `systemd` :

```
$ sudo systemctl restart redis.service
```

Python and Django

Virtualenvs (Linux)

Virtualenv
virtualenvwrapper

Virtualenvwrapper setup

- https://virtualenvwrapper.readthedocs.io/en/latest/install.html#basic-installation
- https://virtualenvwrapper.readthedocs.io/en/latest/
- **Create required directory** structure :
  - Under /home
    - Mkdir web
    - Cd web / Mkdir virtualenvs / cd virtualenvs / Mkdir venv_2025
    - make it write enabled
- **Pip3 install –user** virtualenvwrapper (*only till python 3.10*)
- Apt-get **install** python3-**pip** (before)
- **Setup** .bashrc : (*chmod + locate the code in top of bashrc*)
  - HOMEW='/home/gon2000fr'
  - export WORKON_HOME=$HOMEW/.virtualenvs
  - export VIRTUALENVWRAPPER_SCRIPT=/home/ubuntu/.local/bin/virtualenvwrapper.sh
  - export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python3
  - export VIRTUALENVWRAPPER_VIRTUALENV=/home/ubuntu/.local/bin/virtualenv
  - export PIP_RESPECT_VIRTUALENV=true
  - export PIP_VIRTUAL_BASE=$WORKON_HOME
  - source /usr/local/bin/virtualenvwrapper_lazy.sh
- Quick start :
  - sudo -i
  - Source ~/.bashrc
  - Update .bash_profile (add source .bashrc)
  - Mkvirtualenv venv_2025 , or ….
  - Update POSTACTIVATE (under /venv_2025/bin)
  - Workon venv_2025

Virtualenvwrapper setup (last)

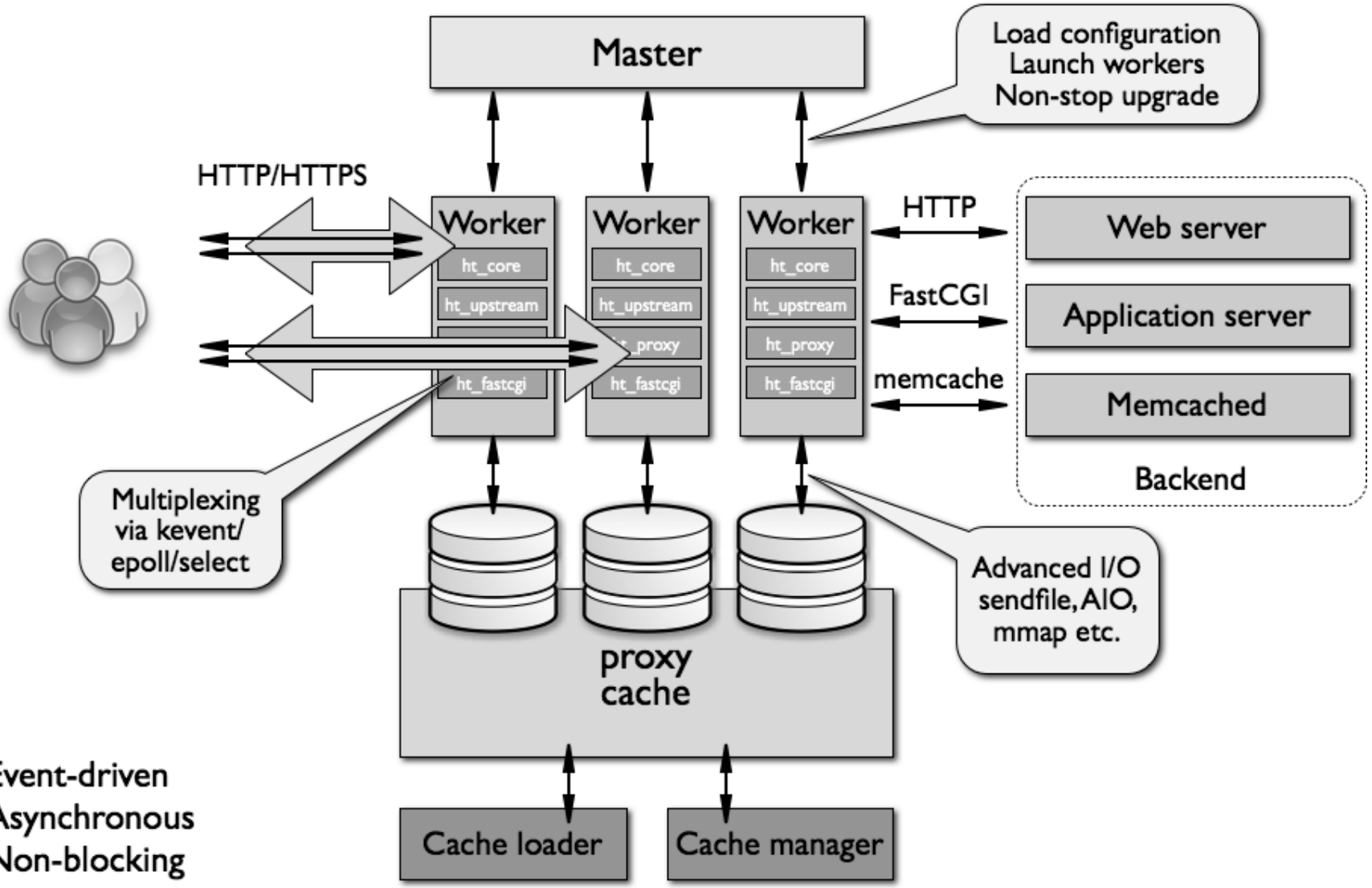
- **Create Under /home/.virtualenvs/your_venv/ :**
  - Mkdir logs
  - Mkdir config (required for Gunicorn and Nginx and Supervisor)
  - Mkdir temp
  - mkvirtualenv -p /usr/bin/python3.12 venv_django5
    - if a second venv is required with different version of Python

Admin Tools addon

- **We are missing some Css Layout** :
  - /admin_tools/css/menu.css et menu-ie.css
  - To be copied directly in Lib packages / admin_tools/dashboard
  - Move utils.js to from /static/js to /dashboard

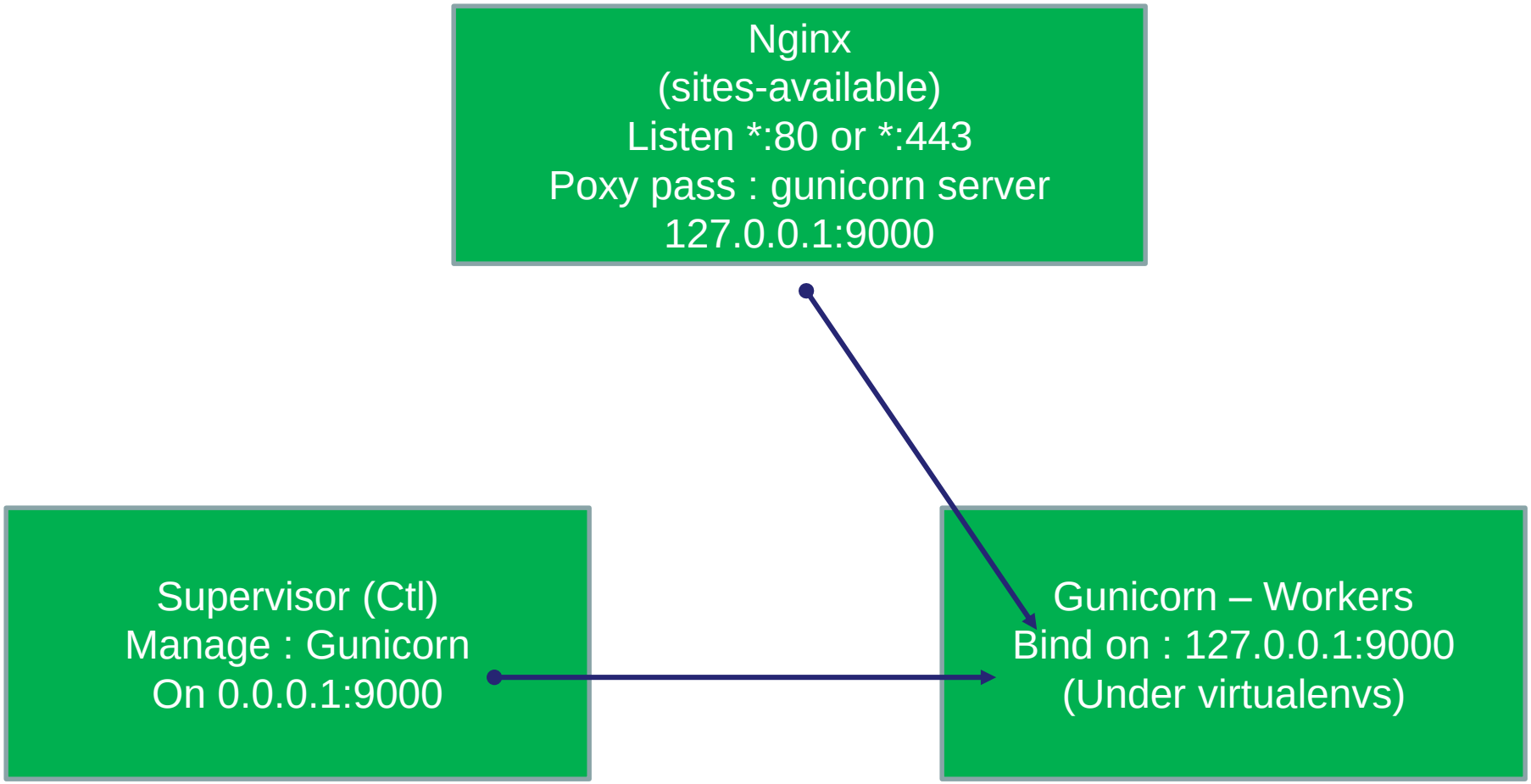
Nginx
and
Gunicorn

Nginx architecture



- Event-driven
- Asynchronous
- Non-blocking

Workflow Nginx/Gunicorn/Supervisor



http://senko.net/en/django-nginx-gunicorn/

Nginx

- Apt-get install nginx
- Apt-get install Gunicorn
- Apt-get install gunicorn3
- Pip3 install gevent

    - Configure nginx :
        - Create entry to sites-available
        - Create symlinks to sites-enabled

    - Nginx.conf
    - ssl sites-available modules

Supervisor

Supervisor

- apt-get install Supervisor
- Cd /etc/Supervisor
- Cd conf.d
  - **Insert a new Conf file** for your own Instance (/home/web/supervisord.conf) :
    - [program:staging_gunicorn]
    - command=/home/web/virtualenvs/venv_2025/bin/gunicorn_django -c /home/web/virtualenvs/django110/config/gunicorn/gunicorn_staging.conf.py
    - -b 0.0.0.0:9000
    - directory=/home/virtualenvs/venv_2025/your_django_project
    - autostart=true
    - autorestart=true
    - stdout_logfile=/home/web/.virtualenvs/test/logs/supervisord.stdout.log
    - stdout_logfile_maxbytes = 50MB
    - stdout_logfile_backups = 5
    - stderr_logfile=/home/web/.virtualenvs/test/logs/supervisord.stderr.log
    - stderr_logfile_maxbytes = 50MB
    - stderr_logfile_backups = 5
    - redirect_stderr=false

- **Reload Supervisor configuration** :
  - Sudo supervisorctl reload
  - Sudo supervisorctl restart staging
  - sudo supervisorctl restart staging
  - Check Logs under /home/web/virtualenvs/venv_2025/logs/

MariaDB

MariaDB Installation

- sudo apt install curl gnupg lsb-release –y
- curl -LsS https://mariadb.org/mariadb_release_signing_key.asc | sudo gpg --dearmor -o /usr/share/keyrings/mariadb-keyring.gpg
- echo "deb [signed-by=/usr/share/keyrings/mariadb-keyring.gpg] http://mirror.hetzner.com/mariadb/repo/11.4/ubuntu jammy main" | sudo tee /etc/apt/sources.list.d/mariadb.list
- sudo apt update
- sudo apt install mariadb-server mariadb-client -y

Setup MariaDb

- sudo tee /etc/mysql/mariadb.conf.d/99-django.cnf >/dev/null <<'EOF’
- [mysqld]
- # Encodage par défaut (Django recommande utf8mb4)
- character-set-server = utf8mb4
- collation-server = utf8mb4_unicode_ci
- # InnoDB (par défaut) + fichiers séparés
- innodb_file_per_table = 1
- innodb_flush_log_at_trx_commit = 1
- innodb_strict_mode = 1
- # SQL mode strict (évite les insert tronqués)
- sql_mode = STRICT_TRANS_TABLES,NO_ENGINE_SUBSTITUTION
- # Sécurité: n'écoute que localement (si serveur d’app web local)
- bind-address = 127.0.0.1
- EOF
- sudo systemctl restart mariadb

PostGreSQL 18

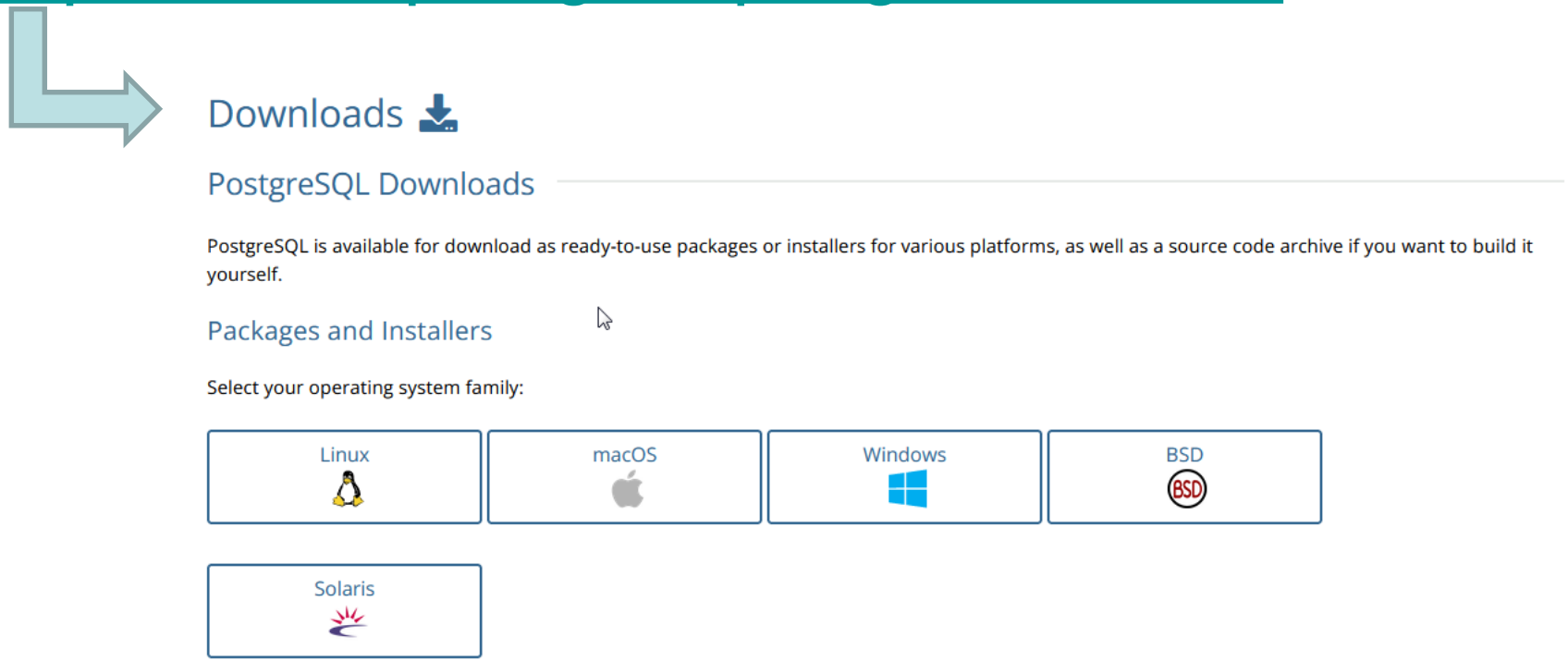
server

Database

python addon

Dowload

- https://www.postgresql.org/download/



PostgreSQL Installation

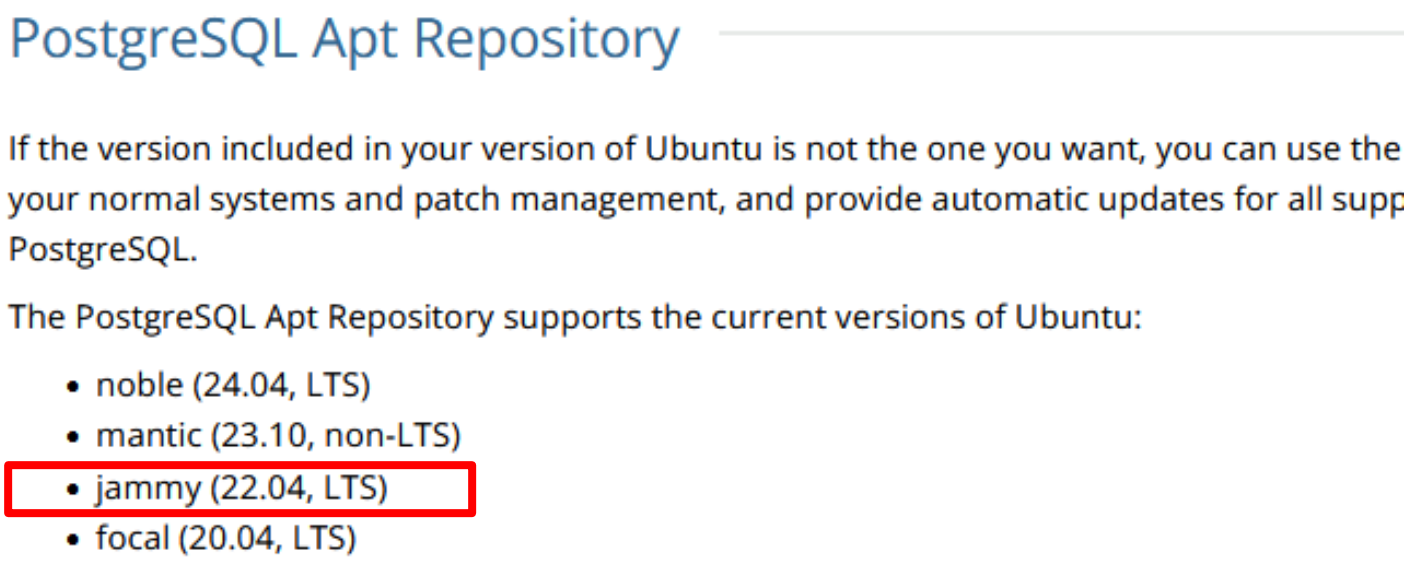
- sudo apt update
- sudo apt-get install **postgresql** postgresql-contrib postgresql-client
- sudo systemctl start postgresql.service
- sudo /etc/init.d/postgresql restart
- sudo –i –u postgres
- Psql
- Exit ➔ \q
- sudo apt-get install pgloader
- Vi /etc/postgresql/18/main/pg_hba.conf (main Postgres config)

- Under sudo mode :
  - Createuser –interactive –P
    - Pgloader_pg, django

- **CREATE DATABASE** database_name WITH TEMPLATE = template0 OWNER "postgres" ENCODING "UTF8" LC_COLLATE = "en_US.UTF-8";
- Under psql : \l (*will List all created DataBases*)

PostGreSQl 18

- sudo apt update
- sudo apt install postgresql-18
- https://dev.to/johndotowl/postgresql-18-installation-on-ubuntu-2404-51ia



PostGreSQL 18 -Installation

- sudo sh -c echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list
- curl -fsSL https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo gpg --dearmor -o /etc/apt/trusted.gpg.d/postgresql.gpg
- sudo apt update
- sudo apt install postgresql-18 postgresql-contrib-18
- **sudo systemctl start postgresql**
- **sudo systemctl enable postgresql**
- sudo vi /etc/postgresql/18/main/postgresql.conf

PostGreSQL – RDS and Remote access

- listen_addresses = ‘*’
- sudo sed -i '/^host/s/ident/md5/' /etc/postgresql/18/main/pg_hba.conf
- sudo sed -i '/^local/s/peer/trust/' /etc/postgresql/18/main/pg_hba.conf
- echo "host all all 0.0.0.0/0 md5" | sudo tee -a /etc/postgresql/18/main/pg_hba.conf
- sudo systemctl restart postgresql

PostGreSQL – Connection

- sudo –i –u postgres
- PSQL :
  - ALTER USER postgres PASSWORD ‘xxxxxxxxxxx';

PostGreSQL & Windows

- https://www.enterprisedb.com/downloads/postgres-postgresql-downloads

PostgreSQL Version	Linux x86-64	Linux x86-32	Mac OS X	Windows x86-64	Windows x86-32
16.2	postgresql.org	postgresql.org			Not supported
15.6	postgresql.org	postgresql.org			Not supported
14.11	postgresql.org	postgresql.org			Not supported

PostGreSQL & Python

sudo apt install

python3-pip
python3-dev
libpq-dev
postgresql-18
postgresql-contrib-18

pip install

psycopg2
django-postgres-extra

PostgreSQL Migration

- Createuser –interactive –P
- Create sudo user postgres or add it to sudo group
- create ROLE root WITH SUPERUSER REPLICATION BYPASSRLS LOGIN CREATEDB CREATEROLE;
- sudo pgloader mysql://login:password@localhost:3306/ideolab postgresql:///ideolab
- Cron Migration has been released : clone_postgresql.py
- recursive extract of FK and M2M fields

Migrations known issues

- Company Data model has corrupted FK on country
  - we have to convert Data Model FK (wrong option)
  - we have to convert Varchar country_id to Integer(11)
  - default to france (Id=77)
- City has an enclosed Manager which block migration
- M2M Migrations will be done separatly
- Case sensitive Issues (*Big show stopper*)
  - example with parameters !!

PSQL – Main commands

- \q : quit
- \l : List DataBase
- \password root : change Password user
- \du : List all Users
- \conninfo : connexion information
- \c database : connect to a DataBase
- \dt : List of Tables under the DataBase
- \dt+ : List of tables with more Information
- \d : List of relations
- \di : List all Indexes of DataBase

PostGreSQL 18 & Linux 24.04 LTS

- sudo apt-get install postgresql
- sudo apt-get install postgresql-client
- sudo apt install postgresql-doc
- sudo systemctl restart postgresql

- https://ubuntu.com/server/docs/install-and-configure-postgresql
- https://www.postgresql.org/docs/current/admin.html

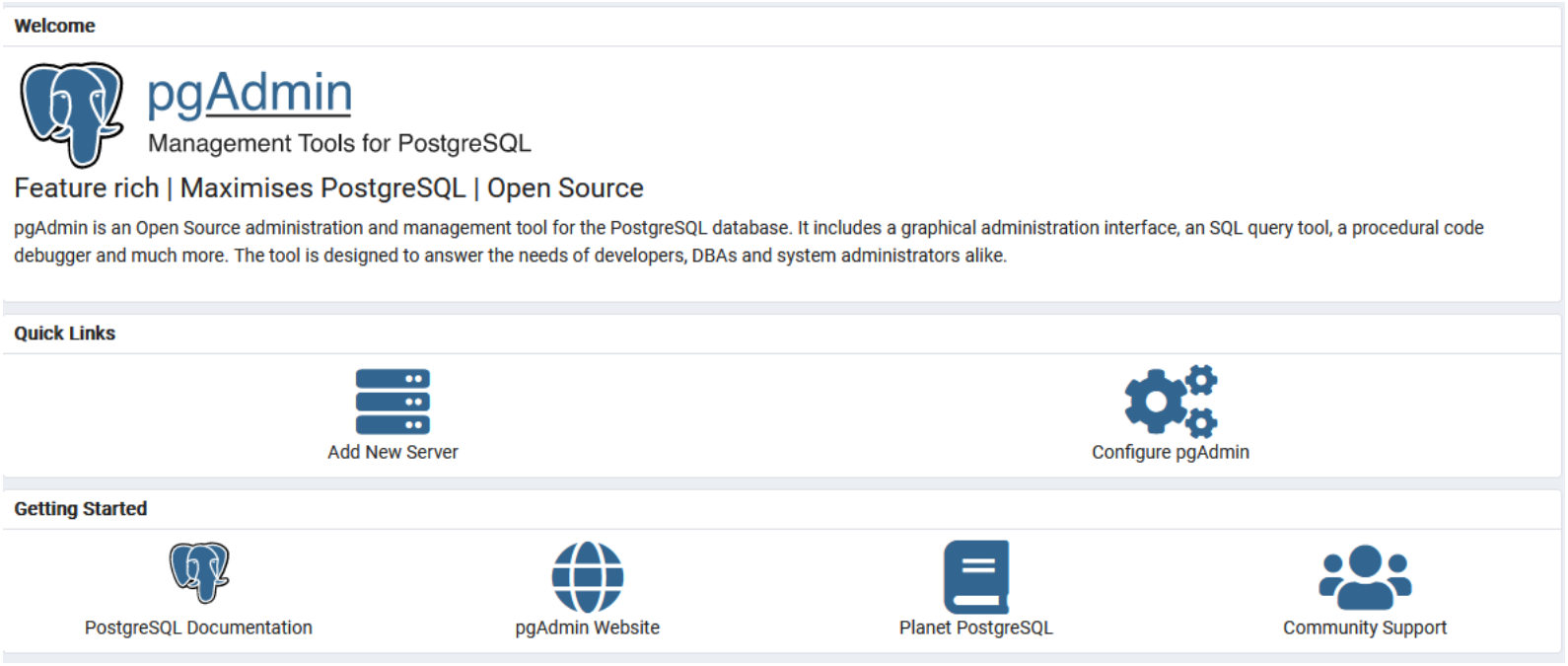
Django and PostgreSQL

- Engine : django.db.backends.postgresql_psycopg2
- Engine : django.db.backends.postgresql
- Pip3 install psycopg2-binary

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
        'NAME': 'your_db_name',
        'USER': 'postgres',
        'PASSWORD': 'your_db_password',
        'HOST': '127.0.0.1',
        'PORT': '5432',
    }
}
```

PG Admin 4

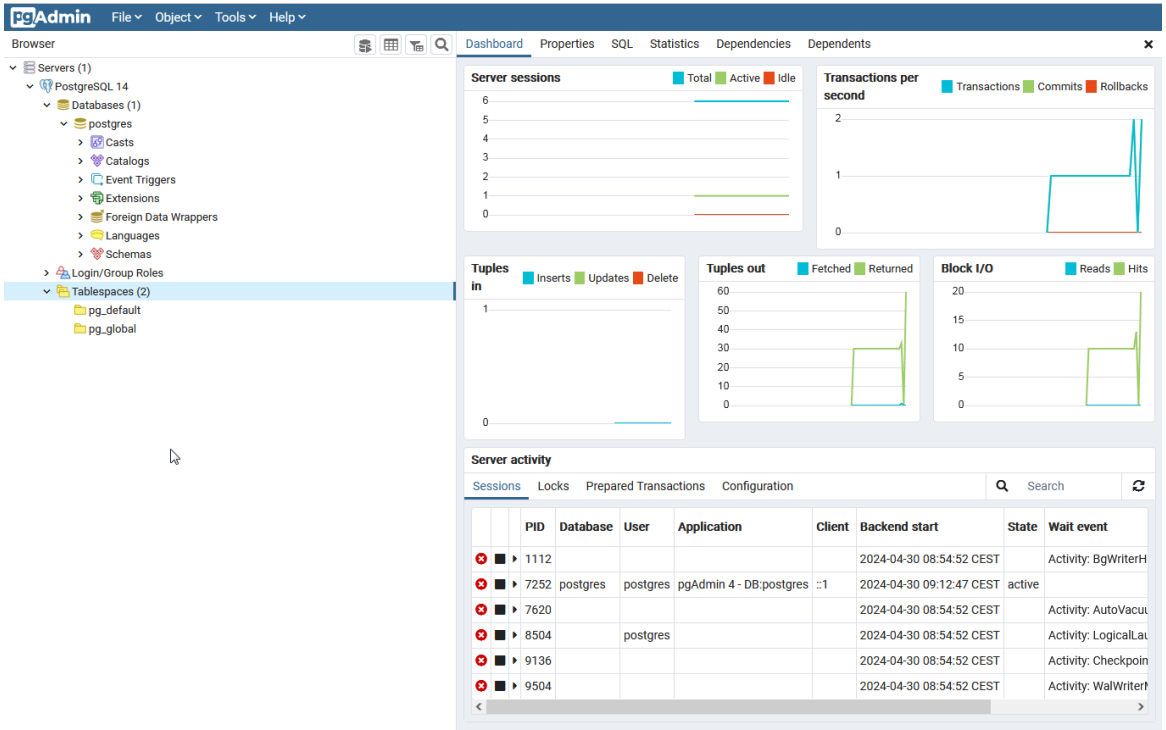
- https://www.how2shout.com/linux/install-postgresql-pgadmin-4-on-ubuntu-24-04-lts-jammy-linux/
- https://www.pgadmin.org/
- https://www.webhi.com/how-to/how-to-install-pgadmin4-on-ubuntu-18-04-20-04-22-04/
-



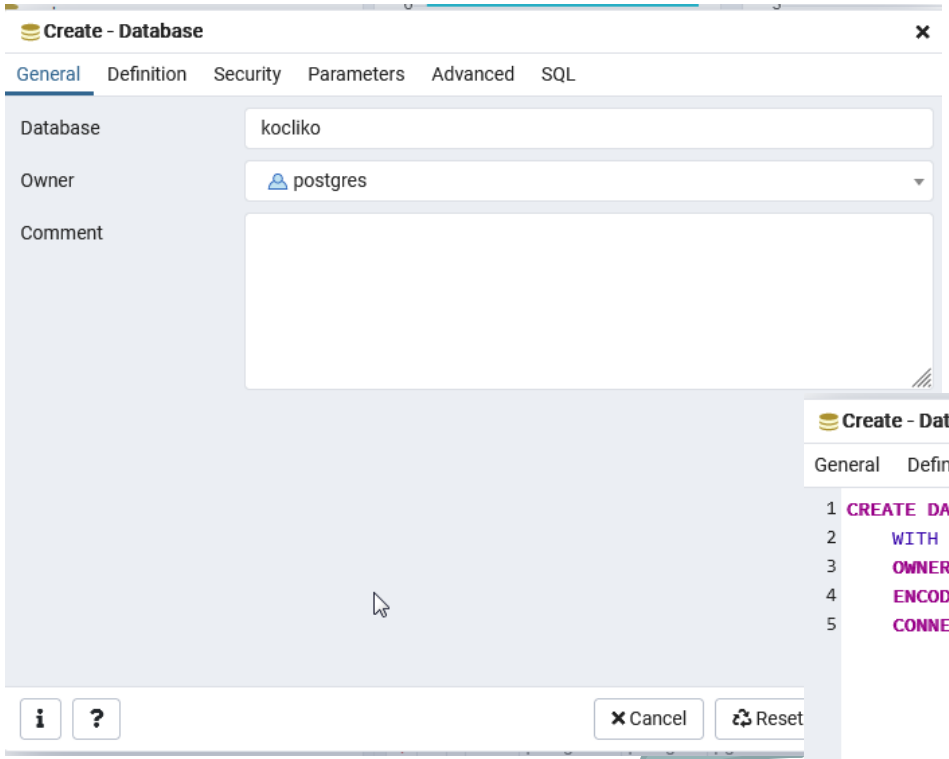
PgAdmin 4 - installation

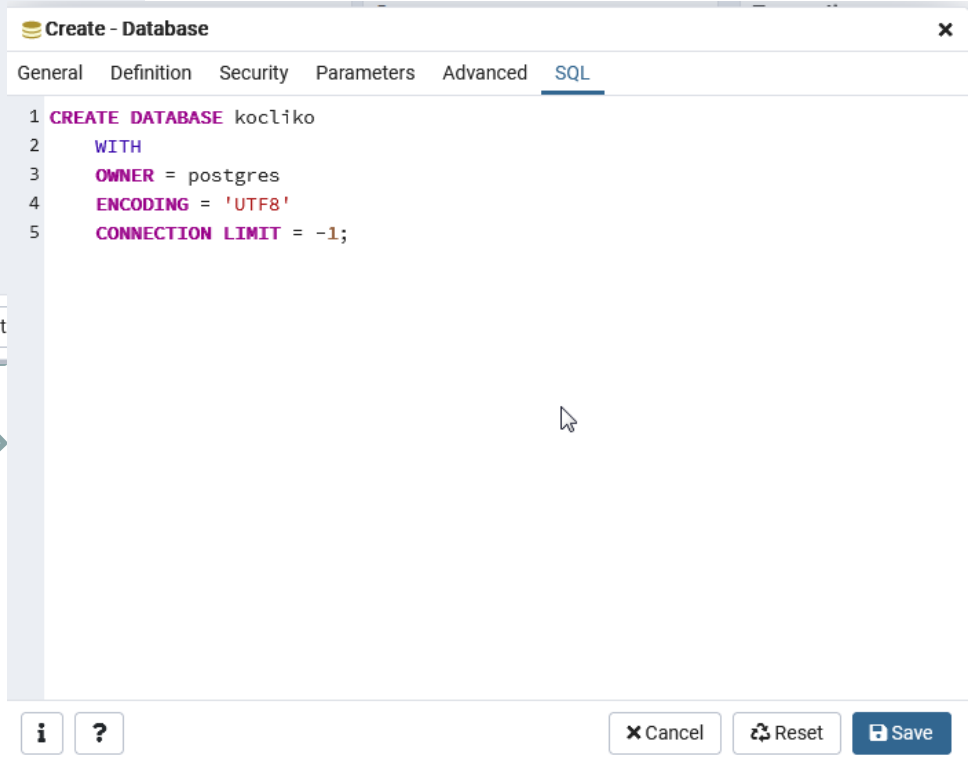
- sudo apt update && sudo apt upgrade –y
- sudo curl https://www.pgadmin.org/static/packages_pgadmin_org.pub | sudo apt-key add
- echo "deb https://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/$(lsb_release -cs)
- pgadmin4 main" | sudo tee /etc/apt/sources.list.d/pgadmin4.list
- sudo apt update
- sudo apt install pgadmin4

PgAdmin 4


pgAdmin
File
Object
Tools
Help
Browser
Servers (1)
PostgreSQL 14
Databases (1)
postgres
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Schemas
Login/Group Roles
Tablespaces (2)
pg_default
pg_global
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Server sessions
Total
Active
Idle
Transactions per second
Transactions
Commits
Rollbacks
Tuples in
Inserts
Updates
Delete
Tuples out
Fetched
Returned
Block I/O
Reads
Hits
Server activity
Sessions
Locks
Prepared Transactions
Configuration
Search
PID
Database
User
Application
Client
Backend start
State
Wait event
1112
2024-04-30 08:54:52 CEST
Activity: BgWriterH
7252
postgres
postgres
pgAdmin 4 - DB:postgres
::1
2024-04-30 09:12:47 CEST
active
7620
2024-04-30 08:54:52 CEST
Activity: AutoVacu
8504
postgres
2024-04-30 08:54:52 CEST
Activity: LogicalLa
9136
2024-04-30 08:54:52 CEST
Activity: Checkpoin
9504
2024-04-30 08:54:52 CEST
Activity: WalWriter

Create your Database





dump and restore

- https://www.dbvis.com/thetable/a-complete-guide-to-pg_dump-with-examples-tips-and-tricks/
- https://www.postgresql.org/docs/current/app-pgdump.html
- https://www.timescale.com/learn/a-guide-to-pg_restore-and-pg_restore-example
- https://docs.postgresql.fr/16/app-pgdump.html

- pg_dump --dbname=database_name --file=dump_postgresql_2806.sql
- pg_dump --format=custom --file dump_postgresql_2806.sql database_name

---

- **pg_dump --clean -F c database_name > dump_postgresql_2806.tar**
- **pg_restore –U postgres –W –C –c –if-exists -d database_name dump.tar**
- **pg_restore -vFd -d database_name dump.tar**
- **pg_restore --jobs=8 --dbname=database_name --verbose dump_postgresql_1407.tar**

vacuum concept

- In PostgreSQL, when a row or tuple is updated or deleted, the record is not actually physically deleted or altered. This leaves obsolete records on the disk, which consume disk space and also negatively affect query performance.

- To fix this, PostgreSQL provides a neat feature called Vacuum that lets me easily clear out such records from the disk and gain back the space, improving query performance too.

- **Vacuum analyze** is a handy way to combine reorg + stat analyzer. could be executed daily

MAIN SERVER SETTINGS

- shared_buffers = 45% of total DRAM
- temp_buffers = 64 to 256 MB
- work_mem = 256 MB
- maintenance_work_mem = 4GB
- fsync = off
- commit_delay = 1000 (micro second)
- checkpoint_timeout = 10min

- ***show shared_buffers;***
- ***alter system set shared_buffers = '32GB';***

MAIN SERVER SETTINGS

- random_page_cost = 2 (For better Index scan)
- effective_cache_size = 48GB
- synchronous_commit = off
- commit_siblings = 10 - 20

WHEN DYNAMIC ALTER OF A SETTINGS IT'S LOCATED HERE :
/var/lib/postgresql/16/main
postgresql.auto.conf

ERROR LOG IS HERE :
/VAR/LOG/POSTGRESQL

WAL Settings

- wal_sync_method = open_datasync
- full_page_writes = off
- wal_buffers = 3% of shared buffer
- max_wal_size = 2GB

Index Monitoring

- pg_stat_user_indexes()
- pg_stat_activity()
- pg_relation_size()
- idx_scan
- EXPLAIN_ANALYZE

Tuning guidelines

- https://www.postgresql.org/docs/current/performance-tips.html
- https://wiki.postgresql.org/wiki/Tuning_Your_PostgreSQL_Server
- https://www.postgresql.org/docs/16/populate.html#POPULATE-MAX-WAL-SIZE
- https://amperecomputing.com/tuning-guides/postgreSQL-tuning-guide
- https://medium.com/@talhakhalid101/tuning-tips-to-maximize-your-postgresql-performance-2a78996ee666
- https://www.postgresql.org/docs/current/runtime-config-wal.html#GUC-COMMIT-SIBLINGS
- https://www.cybertec-postgresql.com/en/bulk-load-performance-in-postgresql/
- https://blog.gitguardian.com/10-tips-to-optimize-postgresql-queries-in-your-django-project/

SERVER CONFIGURATION

- https://docs.postgresql.fr/16/runtime-config.html
- https://data.sigea.educagri.fr/download/sigea/supports/PostGIS/distance/perfectionnement/M06_Bonnes_pratiques/co/20_Parametrage_spatial.html
- https://www.postgresql.org/docs/current/runtime-config-query.html

Purge Old version of Postgresql

- dpkg -l | grep postgresql
- sudo systemctl disable postgresql
- sudo apt-get purge postgresql-18
- sudo apt-get remove --purge postgresql-18
- sudo apt-get autoremove
- sudo rm -rf /etc/postgresql
- sudo rm -rf /var/lib/postgresql
- sudo find / -iname "*postgres*" -exec rm -rf {} \;
- dpkg -l | grep postgresql
- sudo dpkg --remove postgresql-18

Connecting to localhost:5433 database database_name as user 'postgres'...

OS information

[INFO] OS: linux Version: 4.19.0 Arch: x86_64-linux-gnu-thread-multi
[INFO] OS total memory: **45.93 GB**
[BAD] Memory overcommitment is allowed on the system. This may lead the OOM Killer to kill at least one PostgreSQL process, DANGER!
[INFO] sysctl vm.overcommit_ratio=50
[BAD] vm.overcommit_ratio is too low, you will not be able to use more than (50/100)*RAM+SWAP for applications
[WARN] Use the --ssd arg if the VM only uses SSD storage

General instance informations

----- PostgreSQL version -----

[OK] You are using the latest PostreSQL major version (**16.3** (Ubuntu 16.3-1.pgdg20.04+1))

----- Extensions -----

[INFO] Number of activated extensions: 1
[INFO] Activated extensions: plpgsql
[WARN] Extension pg_stat_statements is disabled in database database_name

----- Connection information -----

[INFO] max_connections: 500
[INFO] Current used connections: 6 (1.20%)
[INFO] 3 connections are reserved for super user (0.60%)
[INFO] Average connection age: 21m 40s

----- Memory usage -----

[INFO] Configured work_mem : 300.00 MB
[INFO] Using an average ratio of : work_mem buffers by connection of 150% (use --wmp to change it)
[INFO] Total work_mem (per connection) : 450.00 MB
[INFO] shared_buffers : **28.00 GB**
[INFO] Track activity reserved size : 0.00 B
[INFO] maintenance_work_mem= : 4.00 GB
[INFO] Max memory usage:
shared_buffers (28.00 GB)

+ max_connections * work_mem * average_work_mem_buffers_per_connection (500 * 300.00 MB * 150 / 100 = 219.73 GB)
+ autovacuum_max_workers * maintenance_work_mem (3 * 4.00 GB = 12.00 GB)
+ track activity size (0.00 B)
= 259.73 GB

QUERY OPTIMIZING

- USE only() : limit use of columns
- pay attention to : use of upper()
- dont use : « __iexact » (Django query)
- explicit Index : for upper value (class Meta)
- select_related : Not best match
- large list : use iterator()
- repeated queries : use of « Prepare »

PostGreSQL
&
Partitioning

Base Rules

- Django's ORM doesn't have built-in support for partitioned tables,

- We need for using partitions in our IdeoLab application, little extra work.

- Another option is to use a package called [django-postgres-extra]

Performances Improvement

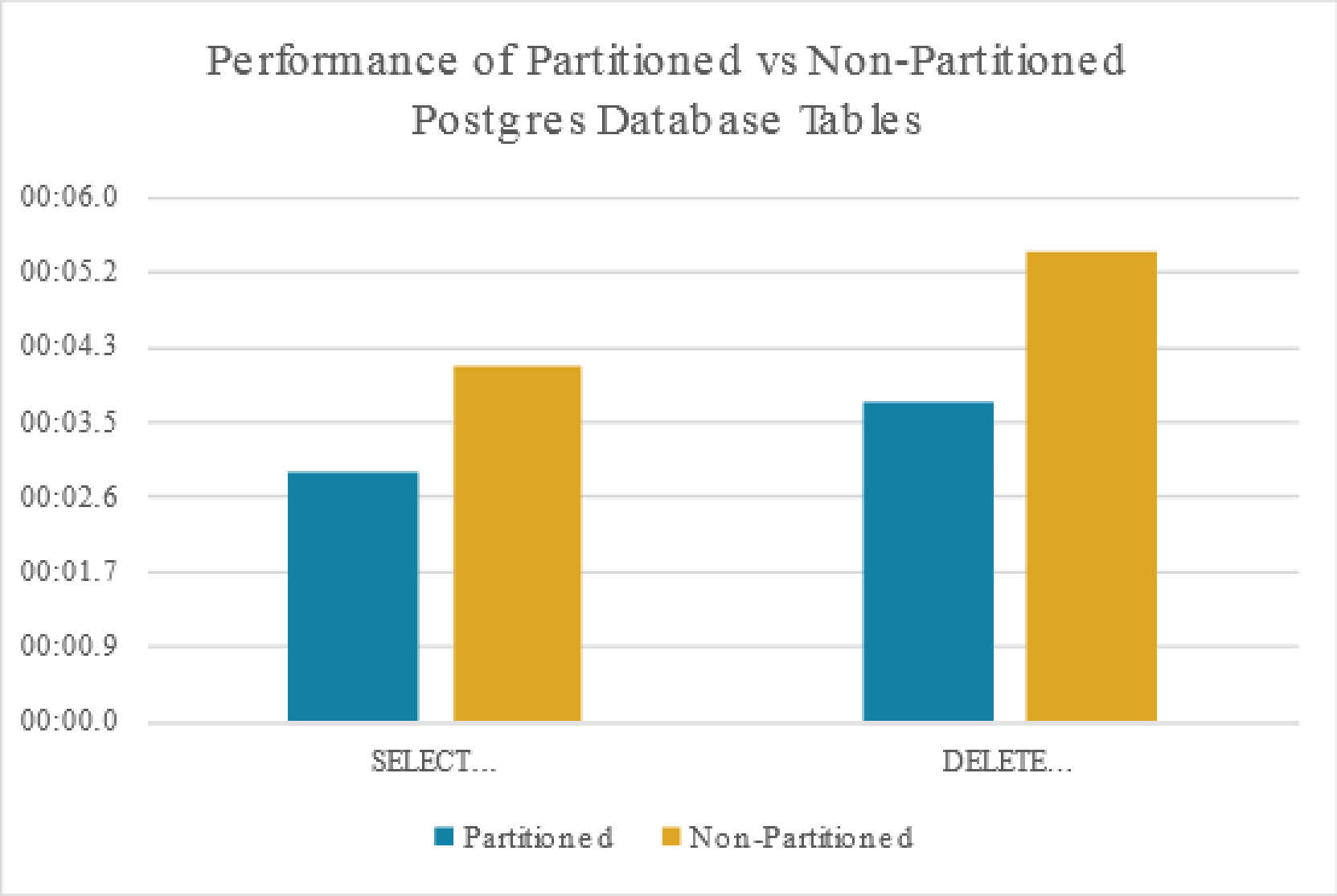


Table Partitioning

- **Table Partitioning:** When you deal with large volumes of data, consider using PostgreSQL's partitioning functionality. This involves splitting large tables into smaller parts, which can significantly speed up queries and reduce maintenance overhead.

- https://www.postgresql.org/docs/current/ddl-partitioning.html
- https://rodoq.medium.com/partition-an-existing-table-on-postgresql-480b84582e8d

- Candidates SQL Tables :
  - **AccomodationConsumption**
  - **AccomodationTemperature**
  - **AccomodationHumidity**
  - **apiDataSensor**
  - **Various logging and Messages Tables**
  - **Sensor + Sensor Status**

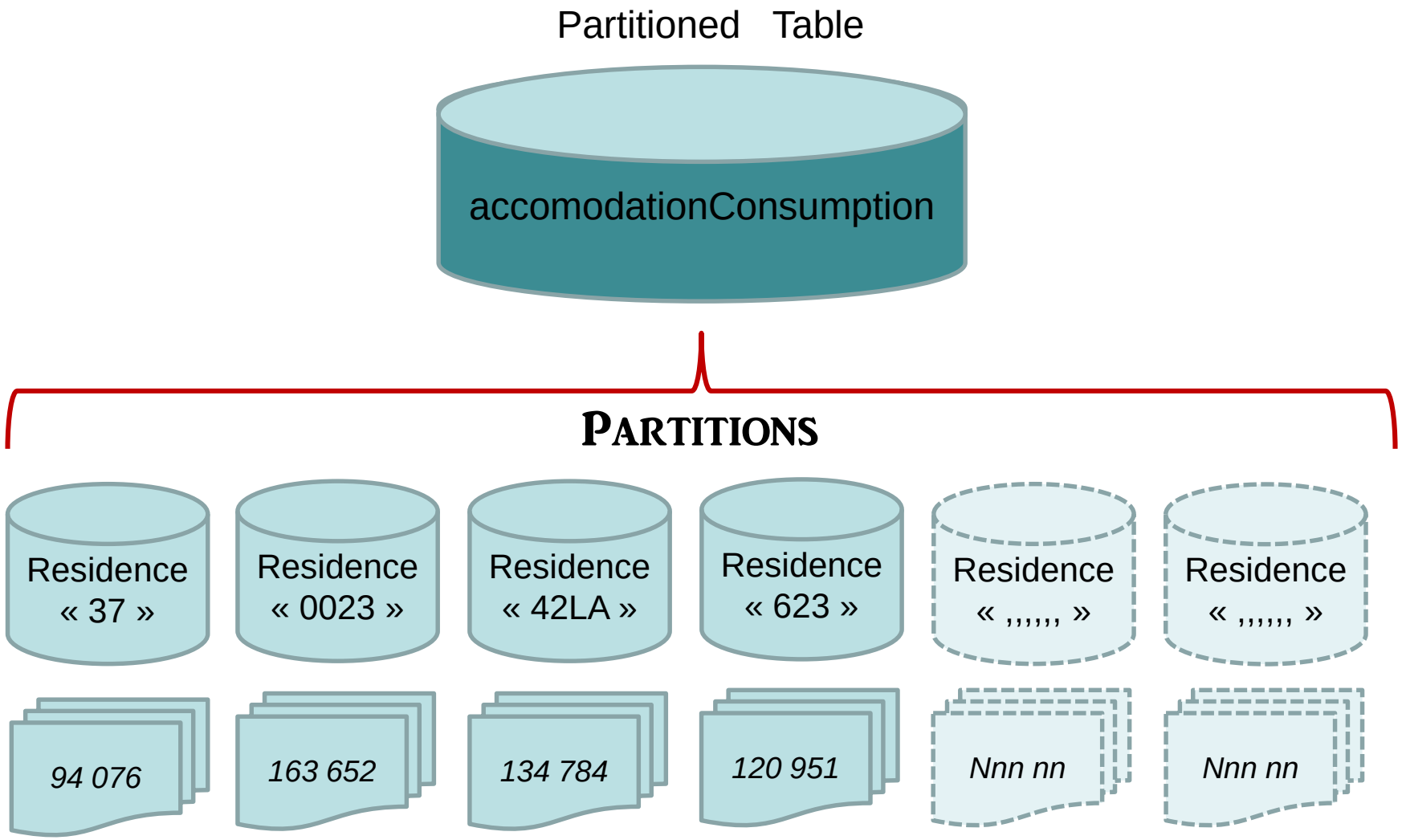
Tablespace Partition Key

- Several Tracks on the way :
  - By Year
  - By Year – Month
  - By Residence
- The Partition key should be scalable
- Building a Partition simulator

How it works

- The partitioned accomodation table - the table we will split into smaller tables by “residence key”-
- It doesn't hold any data. It exists as a parent to the partitions and a blueprint for the table schema.
- One important thing to understand about a partitioned table is that the partitions themselves are also tables
- They are created individually, and we can query them separately

Partitioned Table



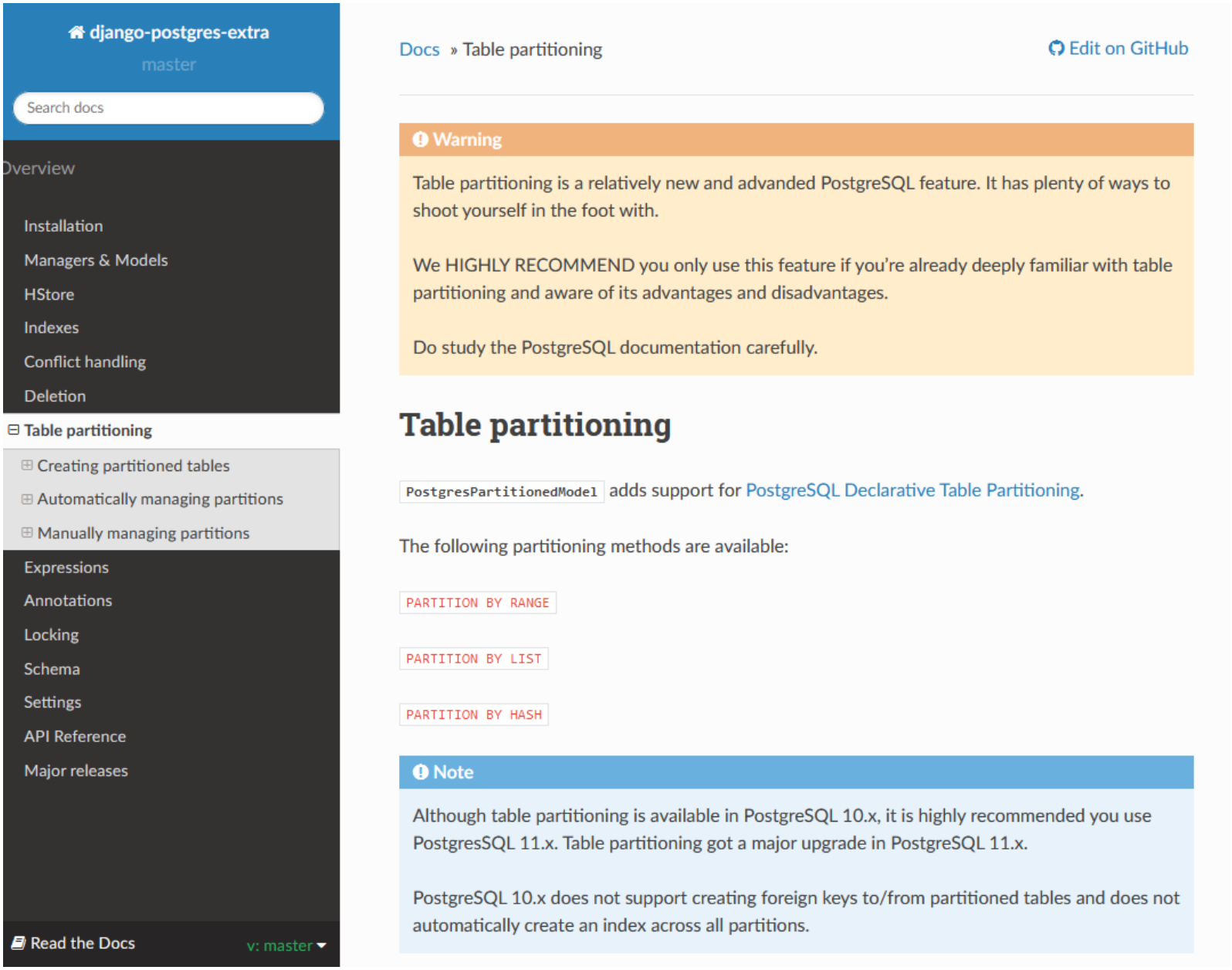
Tablespace Partitioning

- https://pganalyze.com/blog/postgresql-partitioning-django
- https://django-postgres-extra.readthedocs.io/en/master/table_partitioning.html

Creating a partitioned Table

```
CREATE TABLE accomodationConsumption_partitioned (
  id                    BIGSERIAL,
  uid                   text NOT NULL,
  residence_id          integer  NOT NULL,
  PRIMARY KEY (id, residence_id)
) PARTITION BY RANGE (residence_id);
```

Django and Partitioning



Django Partitioning - Get's started

```
from psqlextra.types import PostgresPartitioningMethod

from psqlextra.models import PostgresPartitionedModel

class accomodationConsumption_partitioned(PostgresPartitionedModel):
    class PartitioningMeta:
        method = PostgresPartitioningMethod.RANGE
        key      = ['residence_id']

    uid                 = models.TextField()
    residence_id    = models.integerField()
    nnnnnn.....       = models.----------
    .........
```

Temporary Solution

- Custom Partitioning
- Partitioning by Year:
  - create one SQL per Year
  - create a common Partition selector :
    - getPartitionModel()

Elasticsearch and Java

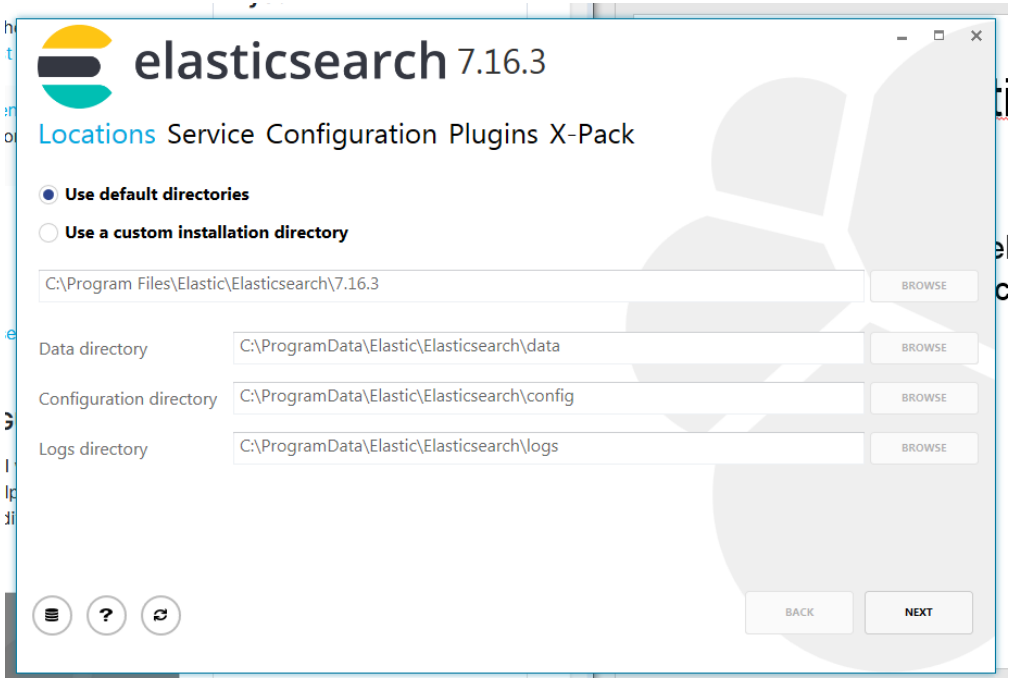
Elasticsearch 7 on Ubuntu 24.04

- sudo apt search openjdk
- sudo apt-get install openjdk-8-jdk
- wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
- sudo apt-get install apt-transport-https
- echo "deb https://artifacts.elastic.co/packages/5.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-5.x.list
- sudo apt-get update
- sudo apt-get install elasticsearch
- sudo /bin/systemctl daemon-reload
- sudo /bin/systemctl enable elasticsearch.service
- sudo systemctl start elasticsearch.service
- sudo ls -la /var/log/elasticsearch/
- sudo cat /var/log/elasticsearch/elasticsearch.log
- curl -XGET 'localhost:9200/?pretty’

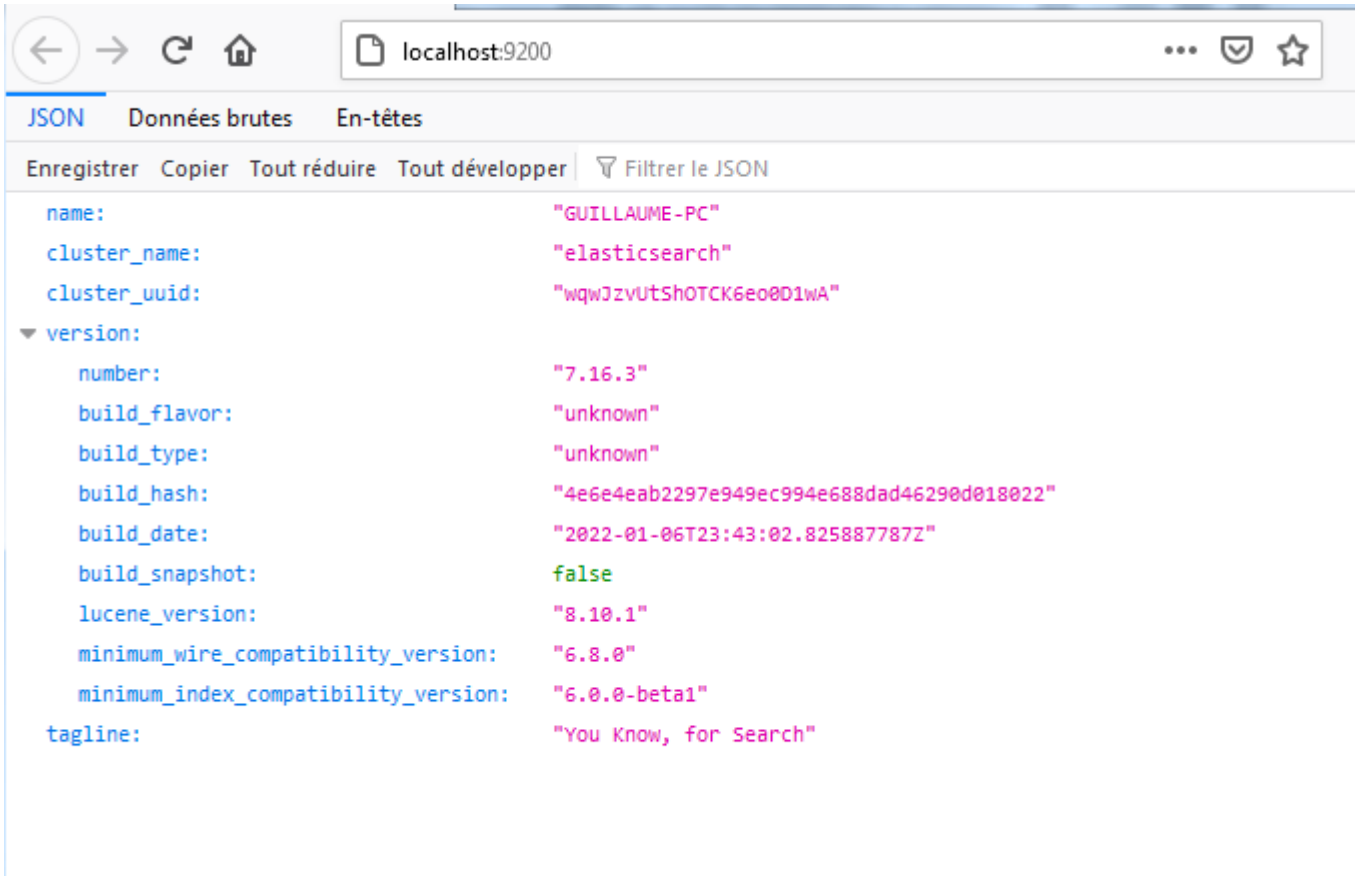
- https://www.willandskill.se/en/install-elasticsearch-5-x-on-ubuntu-16-04-lts-in-less-than-3-minutes/

ElasticSearch 7

- On windows :
  - https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-7.16.3.msi



Test Access


localhost:9200
JSON
Données brutes
En-têtes
Enregistrer
Copier
Tout réduire
Tout développer
Filtrer le JSON
name: "GUILLAUME-PC"
cluster_name: "elasticsearch"
cluster_uuid: "wqwJzvUtShOTCK6eo0D1wA"
version:
number: "7.16.3"
build_flavor: "unknown"
build_type: "unknown"
build_hash: "4e6e4eab2297e949ec994e688dad46290d018022"
build_date: "2022-01-06T23:43:02.825887787Z"
build_snapshot: false
lucene_version: "8.10.1"
minimum_wire_compatibility_version: "6.8.0"
minimum_index_compatibility_version: "6.0.0-beta1"
tagline: "You Know, for Search"

Installing on Linux

wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-7.5.0-amd64.deb

wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-7.5.0-amd64.deb.sha512

sudo shasum -a 512 -c elasticsearch-7.5.0-amd64.deb.sha512

sudo dpkg -i elasticsearch-7.5.0-amd64.deb

sudo /bin/systemctl daemon-reload
sudo /bin/systemctl enable elasticsearch.service

sudo systemctl start elasticsearch.service

DJANGO 5
and
Python

Integrating with Django 5

- pip install django==4.1.4
- pip install celery==5.2.7
- pip install django-environ==0.9.0
- pip install django-cors-headers==3.13.0
- pip install django-admin-tools==0.9.2
- pip install django-online-users==0.3
- pip install django-ckeditor==6.5.1
- pip install django-user-agents==0.4.0
- pip install django-adminplus==0.5
- pip install django-private-chat==0.3.0
- Pip install django-rest-framework==0.1.0
- Pip install django-filter==22.1
- Pip install django-rest-passwordreset==1.3.0
- Pip install django-oauth-toolkit==2.2.0
- Pip install django-seed==0.3.1
- Pip install django-import-export==3.0.2
- Pip install rules==3.3
- Pip install django-extensions==3.2.1
- Pip install django-watson==1.6.2
- Pip install django-admin-list-filter-dropdown==1.0.3
- Pip install django-compressor==4.1
- Pip install babel==2.11.0
- Pip install geoip2==4.6.0
- Pip install simplejson==3.18.0
- Pip install sorl-thumbnail==12.9.0
- pip install psycopg2-binary
- pip install **django-multiple-authentication**
- pip install django-admin-autocomplete-filter
- pip install djangorestframework-jsonapi==4.1.0
- pip install django-multiple-authentication
- pip install gunicorn
- Pip install jsonfield
- Pip install django-mysql==4.8.0
- Pip install django-bootstrap-datepicker-plus==4.0.0
- Pip install django-redis==5.2.0
- Pip install django-jsonview==2.0.0
- Pip install googletrans==3.0.0
- Pip install drf-extensions==0.7.1
- Pip install djangorestframework-jsonapi==4.1.0
- Pip install pandas==2.0.3
- Pip install Pillow=9.3.0
- Pip install django-htmlmin==0.11.0
- Pip install whitenoise==6.2.0
- Pip install google-api-core==2.11.0
- Pip install google-api-python-client==2.70.0
- Pip install google-auth==2.15.0
- Pip install google-auth-httplib2==0.1.0
- Pip install google-auth-oauthlib==0.8.0
- Pip install google-cloud-core==2.3.2
- Pip install google-cloud-translate==3.9.0
- Pip install googleapis-common-protos==1.57.0
- Pip install googletrans==3.0.0
- pip install grpcio
- pip install gevent
- pip install psycopg2
- pip install pip-audit
- pip install gitpython
- pip install django-mobile-detect
- pip install django-session-timeout==0.1.0

Integrating with Django 5.1

- **pip install django==5.1 (*to be installed in first*)**
- **pip install celery==5.4.0**
- **pip install django-environ==0.11.2**
- **pip install django-cors-headers==4.4.0**
- **pip install django-admin-tools==0.9.3**
- **pip install django-online-users==0.3**
- **pip install django-ckeditor==6.7.1**
- **pip install django-user-agents==0.4.0**
- **pip install django-adminplus==0.6**
- **pip install django-private-chat==0.3.0**
- **Pip install django-rest-framework==0.1.0**
- **Pip install django-filter==24.3**
- **Pip install django-rest-passwordreset==1.4.1**
- **Pip install django-oauth-toolkit==3.0.1**
- **Pip install django-seed==0.3.1**
- Pip install django-import-export==4.1.1
- Pip install rules==3.5
- Pip install django-extensions==3.2.3
- Pip install django-watson==1.6.3
- Pip install django-admin-list-filter-dropdown==1.0.3
- Pip install django-compressor==4.5.1
- Pip install babel==2.16.0
- Pip install geoip2==4.8.0
- Pip install simplejson==3.19.3
- Pip install sorl-thumbnail==12.11.0
- pip install psycopg2-binary==2.9.9
- pip install django-multiple-authentication==2.0.5
- pip install django-admin-autocomplete-filter==0.7.1
- pip install gunicorn==23.0.0
- pip install django-postgres-extra==2.0.8
- pip install gitpython==3.1.43

- Pip install jsonfield==3.1.0
- Pip install django-bootstrap-datepicker-plus==5.0.5
- Pip install django-redis==5.4.0
- Pip install django-jsonview==2.0.0
- Pip install googletrans==3.0.0
- Pip install drf-extensions==0.7.1
- Pip install djangorestframework-jsonapi==7.0.2
- Pip install pandas==2.2.3
- Pip install Pillow==10.4.0
- Pip install django-htmlmin==0.11.0
- Pip install whitenoise==6.7.0
- Pip install google-api-core==2.20.0
- Pip install google-api-python-client==2.147.0
- Pip install google-auth==2.35.0
- Pip install google-auth-httplib2==0.2.0
- Pip install google-auth-oauthlib==1.2.1
- Pip install google-cloud-core==2.4.1
- Pip install google-cloud-translate==3.16.0
- Pip install googleapis-common-protos==1.65.0
- Pip install googletrans==3.0.0
- pip install grpcio==1.66.2
- pip install gevent==24.2.1
- pip install psycopg2==2.9.9
- pip install xlrd==2.0.1
- pip install xlwt==1.3.0
- pip install openpyxl==3.1.5
- pip install pyqt5==5.15.11
- pip install setuptools==75.1.0
- munch==4.0.0
- lockfile==0.12.2
- email-validator
- psutil

Django 4 and MySQL

- sudo apt-get install python3-mysqldb
- sudo apt-get install default-libmysqlclient-dev
- Pip3 install mysqlclient

Django 5 and MySQL

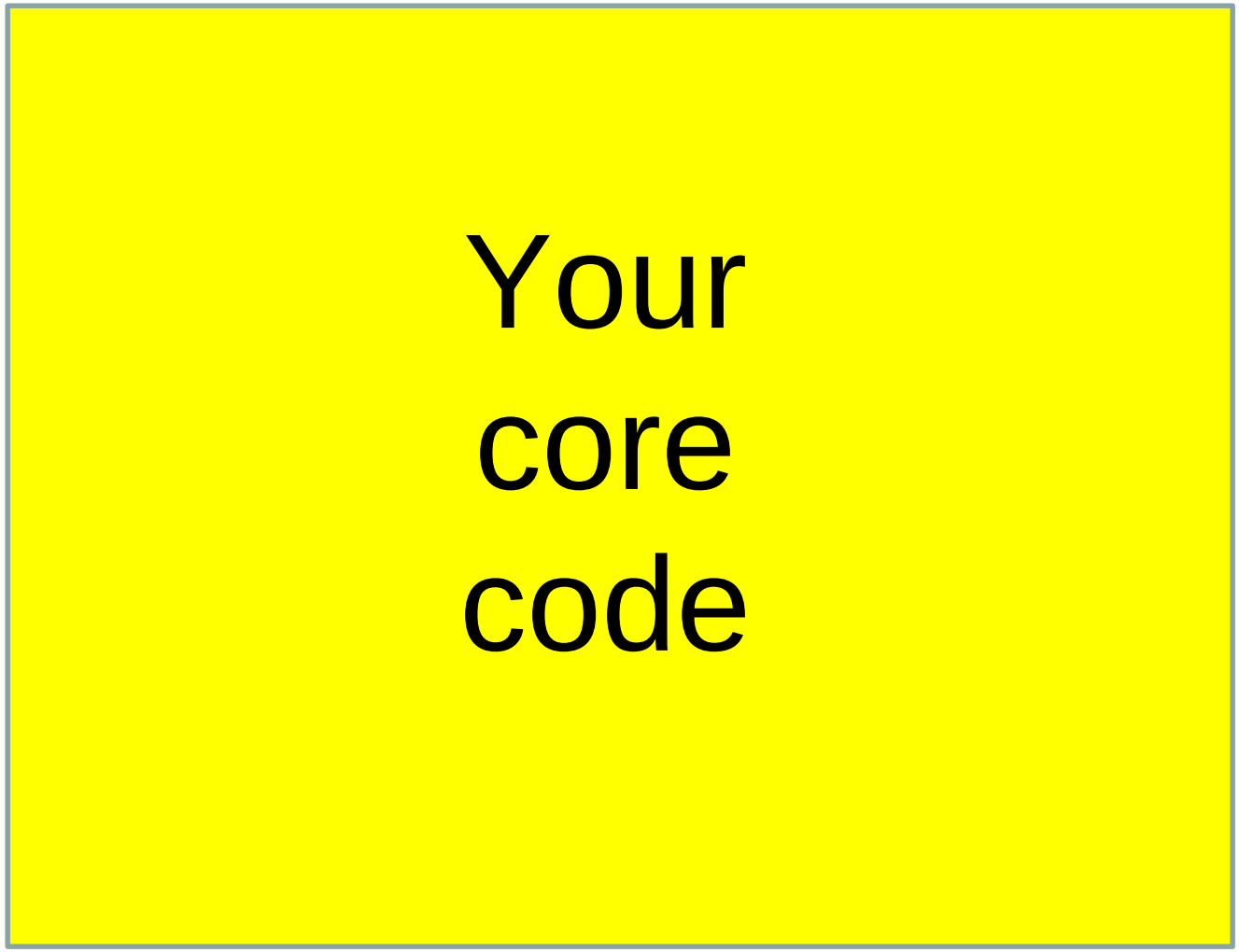
- sudo apt-get install python3-mysqldb
- sudo apt-get install default-libmysqlclient-dev
- Pip3 install mysqlclient

Django 4 and PostGresql

- Engine : django.db.backends.postgresql_psycopg2
- Engine : django.db.backends.postgresql
- Pip3 install psycopg2-binary

Django 5 and PostGresql

- Engine : django.db.backends.postgresql_psycopg3 ???
- Engine : django.db.backends.postgresql
- Pip3 install psycopg3-binary ???


Your
core
code

Settings git repo on windows

- Git clone from unfuddle
- Create_netrc on c:/windows/user/xxxxx
- Add credential code :
- Git add *
- Git commit –am « initial commit »
- Git push origin master
- https://unfuddle.com/stack/docs/help/git-access-ssh/ or BitBucket, GitHub

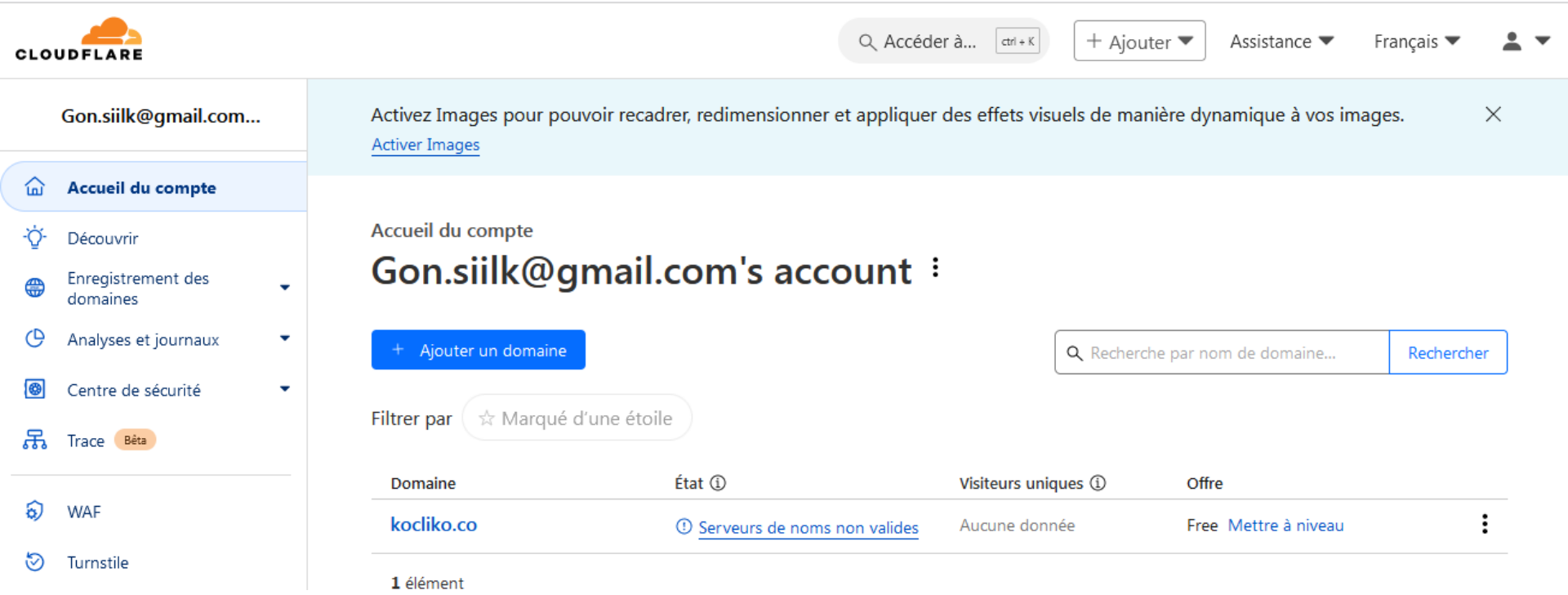
SSH key and GIT

- Pip install virtualenvwrapper
- Apt-get install git
- ssh-keygen -t rsa -C email@gmail.com
- Setup into .bashrc (see previous page for details)
- Mkvirtualenv venv_2025
  - /home/web/virtualenvs/venv_2025:
    - Bin
    - Include
    - Lib
    - ideolab-mailer (Our repo under Virtualenvs)
    - Src (not used)
- Pip install gunicorn
- Cd /home/web/virtualenvs/venv_2025
- Create.netrc on /home
  - **machine unfuddle.io or any Https linked to your Rep**
  - **login email@gmail.com**
  - **password Password+2025 (your own password)**
  - protocol https

SECURITY

CloudFlare

- Under DNS Registrar & Reniew Payment



Firewall

Base IP Tables rules

- 1. **Limiter les connexions par IP**
- 2. **Limiter le taux de requêtes**
- 3. **Bloquer les IP malveillantes (Blacklist manuelle)**
- 4. **Protection contre le SYN Flood**
- 5. **Limiter les ping (ICMP Echo Requests)**
- 6. **Bloquer les paquets invalides**
- 7. **Protection contre les scans de port**
- 8. **Activer le mode syn-cookies**
- 9. **Journalisation des attaques**
- 10. **Sauvegarder les règles**
- **11 Drop Invalid Packets**
- **12 BLOCK COMMON ATTACK PATTERNS**

Base IPTables commands

- iptables -A INPUT -p tcp --syn --dport 80/443 -m connlimit --connlimit-above 50 -j DROP
- iptables -A INPUT -p tcp --dport 80/443 -m limit --limit 25/second --limit-burst 50 -j ACCEPTiptables -A INPUT -p tcp --dport 80/443 -j DROP
- iptables -A INPUT -s 192.168.1.100 -j DROP
- iptables -A INPUT -p tcp --syn -m limit --limit 10/second --limit-burst 20 -j ACCEPTiptables -A INPUT -p tcp --syn -j DROP
- iptables -A INPUT -p icmp --icmp-type echo-request -m limit --limit 1/second -j ACCEPTiptables -A INPUT -p icmp --icmp-type echo-request -j DROP
- iptables -A INPUT -m state --state INVALID -j DROP
- iptables -A INPUT -p tcp --dport 0:442 -m recent --name portscan --rcheck --seconds 86400 -j DROPiptables -A INPUT -p tcp --dport 0:65535 -m recent --name portscan --set -j DROP
- net.ipv4.tcp_syncookies = 1
- iptables -A INPUT -m limit --limit 5/min -j LOG --log-prefix "DDoS Attempt: " --log-level 7
- iptables-save > /etc/iptables/rules.v4
- iptables -t mangle -A PREROUTING -m conntrack --ctstate INVALID -j DROP
- iptables -A INPUT -p tcp -m connlimit --connlimit-above 50 --connlimit-mask 32 -j DROP
- iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP
- iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

Drop Packet from Bogus IP Address

- *iptables -t mangle -A PREROUTING -s 224.0.0.0/3 -j DROP*
- *iptables -t mangle -A PREROUTING -s 169.254.0.0/16 -j DROP*
- *iptables -t mangle -A PREROUTING -s 172.16.0.0/12 -j DROP*
- *iptables -t mangle -A PREROUTING -s 192.0.2.0/24 -j DROP*
- *iptables -t mangle -A PREROUTING -s 192.168.0.0/16 -j DROP*
- *iptables -t mangle -A PREROUTING -s 10.0.0.0/8 -j DROP*
- *iptables -t mangle -A PREROUTING -s 0.0.0.0/8 -j DROP*
- *iptables -t mangle -A PREROUTING -s 240.0.0.0/5 -j DROP*

ITS JUST A SAMPLE TO BE AUTOMATED BY A PERL SCRIPT

Rules against port scanning & Rst packets

- iptables -N port-scanning
- iptables -A port-scanning -p tcp --tcp-flags SYN,ACK,FIN,RST RST -m limit --limit 1/s --limit-burst 2 -j RETURN
- iptables -A port-scanning -j DROP

- iptables -A INPUT -p tcp --tcp-flags RST RST -m limit --limit 2/s --limit-burst 2 -j ACCEPT iptables -A INPUT -p tcp --tcp-flags RST RST -j DROP

iptables summary –Prod server

```
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@ip-172-31-36-52:~# iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP
root@ip-172-31-36-52:~# iptables -A INPUT -p tcp --tcp-flags ALL ALL -j DROP
root@ip-172-31-36-52:~# iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
root@ip-172-31-36-52:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       tcp  --  anywhere             anywhere             tcp dpt:http flags:FIN,SYN,RST,ACK/SYN #conn src
/32 > 100
DROP       tcp  --  anywhere             anywhere             tcp dpt:https flags:FIN,SYN,RST,ACK/SYN #conn sr
c/32 > 100
DROP       tcp  --  anywhere             anywhere             tcp dpt:7755 flags:FIN,SYN,RST,ACK/SYN #conn src
/32 > 100
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http limit: avg 25/sec burst 50
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https limit: avg 50/sec burst 100
ACCEPT     tcp  --  anywhere             anywhere             tcp flags:FIN,SYN,RST,ACK/SYN limit: avg 50/sec
burst 100
DROP       tcp  --  anywhere             anywhere             tcp dpt:http
DROP       tcp  --  anywhere             anywhere             tcp dpt:http
DROP       tcp  --  anywhere             anywhere             tcp flags:FIN,SYN,RST,ACK/SYN
ACCEPT     icmp --  anywhere             anywhere             icmp echo-request limit: avg 3/sec burst 5
DROP       icmp --  anywhere             anywhere             icmp echo-request
DROP       all  --  anywhere             anywhere             state INVALID
DROP       tcp  --  anywhere             anywhere             tcp dpts:0:finger recent: CHECK seconds: 86400 n
ame: portscan side: source mask: 255.255.255.255
LOG        all  --  anywhere             anywhere             limit: avg 5/min burst 5 LOG level debug prefix
"DDoS Attempt: "
DROP       tcp  --  anywhere             anywhere             #conn src/32 > 50
DROP       tcp  --  anywhere             anywhere             tcp flags:FIN,SYN,RST,PSH,ACK,URG/NONE
DROP       tcp  --  anywhere             anywhere             tcp flags:FIN,SYN,RST,PSH,ACK,URG/FIN,SYN,RST,PS
H,ACK,URG
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@ip-172-31-36-52:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       tcp  --  anywhere             anywhere             tcp dpt:http flags:FIN,SYN,RST,ACK/SYN #conn src/32 > 100
DROP       tcp  --  anywhere             anywhere             tcp dpt:https flags:FIN,SYN,RST,ACK/SYN #conn src/32 > 100
DROP       tcp  --  anywhere             anywhere             tcp dpt:7755 flags:FIN,SYN,RST,ACK/SYN #conn src/32 > 100
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http limit: avg 25/sec burst 50
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https limit: avg 50/sec burst 100
ACCEPT     tcp  --  anywhere             anywhere             tcp flags:FIN,SYN,RST,ACK/SYN limit: avg 50/sec burst 100
DROP       tcp  --  anywhere             anywhere             tcp dpt:http
DROP       tcp  --  anywhere             anywhere             tcp dpt:http
DROP       tcp  --  anywhere             anywhere             tcp flags:FIN,SYN,RST,ACK/SYN
ACCEPT     icmp --  anywhere             anywhere             icmp echo-request limit: avg 3/sec burst 5
DROP       icmp --  anywhere             anywhere             icmp echo-request
DROP       all  --  anywhere             anywhere             state INVALID
DROP       tcp  --  anywhere             anywhere             tcp dpts:0:finger recent: CHECK seconds: 86400 name: portscan side: source mask: 255.255.255.255
LOG        all  --  anywhere             anywhere             limit: avg 5/min burst 5 LOG level debug prefix "DDoS Attempt: "
DROP       tcp  --  anywhere             anywhere             #conn src/32 > 50
DROP       tcp  --  anywhere             anywhere             tcp flags:FIN,SYN,RST,PSH,ACK,URG/NONE
DROP       tcp  --  anywhere             anywhere             tcp flags:FIN,SYN,RST,PSH,ACK,URG/FIN,SYN,RST,PSH,ACK,URG
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
```

Install detection tools

- Apt-get install chkrootkit
- Apt-get install tiger
- Apt-get install rkhunter (rkhunter –c –sk)
- Apt-get install packagesearch
- Apt-get install logwatch
- Apt-get install tcptrack
- https://www.cyberciti.biz/faq/block-entier-country-using-iptables/
- *ifconfig*
- *tcptrack –i ens5*

Ddos attach prevention

iptables -t mangle -A PREROUTING -m conntrack --ctstate INVALID -j DROP

iptables -t mangle -A PREROUTING -p tcp ! --syn -m conntrack --ctstate NEW -j DROP

iptables -t mangle -A PREROUTING -p tcp -m conntrack --ctstate NEW -m tcpmss ! --mss 536:65535 -j DROP

iptables -t mangle -A PREROUTING -p icmp -j DROP

https://javapipe.com/blog/iptables-ddos-protection/

https://javapipe.com/blog/iptables-ddos-protection/

iptables -A INPUT -p tcp -m connlimit --connlimit-above 80 -j REJECT --reject-with tcp-reset

iptables -A INPUT -p tcp -m conntrack --ctstate NEW -m limit --limit 60/s --limit-burst 20 -j ACCEPT

iptables -A INPUT -p tcp -m conntrack --ctstate NEW -j DROP

iptables -t mangle -A PREROUTING -f -j DROP

Doos attack iptables last rules

- ### 5: Block spoofed packets ###
- /sbin/iptables -t mangle -A PREROUTING -s 224.0.0.0/3 -j DROP
- /sbin/iptables -t mangle -A PREROUTING -s 169.254.0.0/16 -j DROP
- /sbin/iptables -t mangle -A PREROUTING -s 172.16.0.0/12 -j DROP
- /sbin/iptables -t mangle -A PREROUTING -s 192.0.2.0/24 -j DROP
- /sbin/iptables -t mangle -A PREROUTING -s 192.168.0.0/16 -j DROP
- /sbin/iptables -t mangle -A PREROUTING -s 10.0.0.0/8 -j DROP
- /sbin/iptables -t mangle -A PREROUTING -s 0.0.0.0/8 -j DROP
- /sbin/iptables -t mangle -A PREROUTING -s 240.0.0.0/5 -j DROP
- /sbin/iptables -t mangle -A PREROUTING -s 127.0.0.0/8 ! -i lo -j DROP
-
- ### 6: Drop ICMP (you usually don't need this protocol) ###
- /sbin/iptables -t mangle -A PREROUTING -p icmp -j DROP
-
- ### 7: Drop fragments in all chains ###
- /sbin/iptables -t mangle -A PREROUTING -f -j DROP
-
- ### 8: Limit connections per source IP ###
- /sbin/iptables -A INPUT -p tcp -m connlimit --connlimit-above 111 -j REJECT --reject-with tcp-reset
-
- ### 9: Limit RST packets ###
- /sbin/iptables -A INPUT -p tcp --tcp-flags RST RST -m limit --limit 2/s --limit-burst 2 -j ACCEPT
- /sbin/iptables -A INPUT -p tcp --tcp-flags RST RST -j DROP
-
- ### 10: Limit new TCP connections per second per source IP ###
- /sbin/iptables -A INPUT -p tcp -m conntrack --ctstate NEW -m limit --limit 60/s --limit-burst 20 -j ACCEPT
- /sbin/iptables -A INPUT -p tcp -m conntrack --ctstate NEW -j DROP
-

Install a Firewall

- Apt-get install iptables
- Apt-get install portsentry
- Apt-get install nmap
- Apt-get install fail2ban

- Limit Flood or Service deny
- Limit scan ports
- Limit connections per sec/IP
- Denny Ip address
- Server Intrusion (portsentry/Iptables)
  - Portsentry –audp
  - Portsentry -atcp
- Scan Currently Opened Ports (nmap)
  - Both servers PX90 and PX120
- Set in crontab an antivirus scanner
  - Virus « IptabLex » and « IptabLes »

```
#!/bin/sh

# Réinitialise les règles
sudo iptables -t filter -F
sudo iptables -t filter -X

# Bloque tout le trafic
sudo iptables -t filter -P INPUT DROP
sudo iptables -t filter -P FORWARD DROP
sudo iptables -t filter -P OUTPUT DROP

# Autorise les connexions déjà établies et localhost
sudo iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
sudo iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
sudo iptables -t filter -A INPUT -i lo -j ACCEPT
sudo iptables -t filter -A OUTPUT -o lo -j ACCEPT

# ICMP (Ping)
sudo iptables -t filter -A INPUT -p icmp -j ACCEPT
sudo iptables -t filter -A OUTPUT -p icmp -j ACCEPT

# SSH
sudo iptables -t filter -A INPUT -p tcp --dport 22 -j ACCEPT
sudo iptables -t filter -A OUTPUT -p tcp --dport 22 -j ACCEPT

# DNS
sudo iptables -t filter -A OUTPUT -p tcp --dport 53 -j ACCEPT
sudo iptables -t filter -A OUTPUT -p udp --dport 53 -j ACCEPT
sudo iptables -t filter -A INPUT -p tcp --dport 53 -j ACCEPT
sudo iptables -t filter -A INPUT -p udp --dport 53 -j ACCEPT

# HTTP
sudo iptables -t filter -A OUTPUT -p tcp --dport 80 -j ACCEPT
sudo iptables -t filter -A INPUT -p tcp --dport 80 -j ACCEPT

# FTP
sudo iptables -t filter -A OUTPUT -p tcp --dport 20:21 -j ACCEPT
sudo iptables -t filter -A INPUT -p tcp --dport 20:21 -j ACCEPT

# Mail SMTP
iptables -t filter -A INPUT -p tcp --dport 25 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 25 -j ACCEPT

# Mail POP3
iptables -t filter -A INPUT -p tcp --dport 110 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 110 -j ACCEPT

# Mail IMAP
iptables -t filter -A INPUT -p tcp --dport 143 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 143 -j ACCEPT

# NTP (horloge du serveur)
sudo iptables -t filter -A OUTPUT -p udp --dport 123 -j ACCEPT
```

How to Check overloading server

- *#!/bin/bash*
- ip="148.251.238.138"
- port="80"
- **for** i **in {**1..100**}**
- **do** # ***do*** *nothing just connect and exit*
  echo "exit" | nc ${ip} ${port};
- **done**

New Iptables rules by country

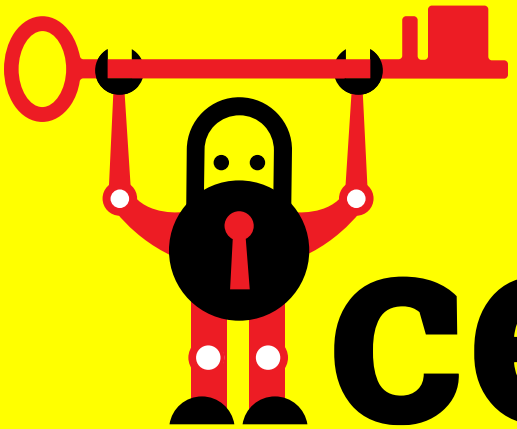
```
# ----------------------------------------
# ---- Block all Chineses IP address   ---
# ----------------------------------------
counter=1
ISO="cn ru kr"
IPT=/sbin/iptables
WGET=/usr/bin/wget
EGREP=/bin/egrep
SPAMLIST="countrydrop"
ZONEROOT="/root/iptables"
DLROOT="http://www.ipdeny.com/ipblocks/data/countries"
$IPT -F $SPAMLIST
[ ! -d $ZONEROOT ] && /bin/mkdir -p $ZONEROOT
$IPT -N $SPAMLIST
for c  in $ISO
do
        tDB=$ZONEROOT/$c.zone
        #$WGET -O $tDB $DLROOT/$c.zone
        SPAMDROPMSG="$c Country Drop"
        BADIPS=$(egrep -v "^#|^$" $tDB)
        for ipblock in $BADIPS
        do
          $IPT -A $SPAMLIST -s $ipblock -j DROP
          counter=`expr $counter + 1`
        done
done
$IPT -I INPUT -j $SPAMLIST
$IPT -I OUTPUT -j $SPAMLIST
$IPT -I FORWARD -j $SPAMLIST
iptables -L -n
echo "ADDED " $counter " IP Address Blocks"
```

TO DO LIST

- AUTOMATE LOGWATCH ANALYZE
- KERNEL PARAMETERS TUNING
- REGULAR MONITORING (TOOLS)
- CLOUDFLARE DNS REGISTRAR

SSL HTTPS

ssl Https



certbot

Certbot and Let's encrypt

- sudo apt-get update
- sudo apt-get install software-properties-common
- sudo add-apt-repository ppa:certbot/certbot
- sudo apt-get update
- sudo apt-get install certbot

- ********** IF ERROR ****
- **python3 -m pip install cffi**
- **python3 -m pip install certbot-nginx**

- If already installed, good to desinstall and reinstall again

sudo certbot –nginx

sudo certbot --nginx certonly –d www.ideo-lab.com -d www.staging.ideo-lab.com

NGINX Settings

```
# --------------------------------
# --------------------------------
# --- SSL Settings             ---
# --------------------------------
# --------------------------------
ssl_enable = False
SECURE_SSL_REDIRECT = False
SESSION_COOKIE_SECURE = False
CSRF_COOKIE_SECURE = False

SSL_URLS = (
    '^/login/',
        # ...
    )

SSL_IGNORE_URLS = (
    '^/i18n_js$',
    '^/static/',
        # ...
    )

if ssl_enable:
    SECURE_SSL_REDIRECT = True
    SESSION_COOKIE_SECURE = True
    CSRF_COOKIE_SECURE = True
    SECURE_HSTS_SECONDS = 31536000
    SECURE_PROXY_SSL_HEADER = ('HTTP_X_FORWARDED_PROTO', 'https')
    SECURE_REDIRECT_EXEMPT = [
    ]
```

NGINX Settings

```
# ------------------------------------------------
   # ---  S S L   Settings for nginx                ---
   # ------------------------------------------------
   ssl on;
   ssl_certificate  /etc/letsencrypt/live/www.ideo-lab.com/fullchain.pem;
   ssl_certificate_key /etc/letsencrypt/live/www.ideo-lab.com/privkey.pem;
   ssl_trusted_certificate  /etc/letsencrypt/live/www.ideo-lab.com/chain.pem;
   include /etc/letsencrypt/options-ssl-nginx.conf;
   ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;

   ssl_stapling on;  .
   ssl_stapling_verify on;
   ssl_protocols    SSLv3  TLSv1  TLSV1.1  TLSv1.2;
   ssl_ciphers EECDH+AESGCM:EECDH+CHACHA20:EECDH+AES;
   ssl_session_cache builtin:1000  shared:SSL:10m;
   ssl_ecdh_curve secp384r1;
   ssl_prefer_server_ciphers on;

   ## TLS parameters
   ssl_session_cache shared:SSL:10m;
   ssl_session_timeout 5m;
   ssl_session_tickets off;
```

Google API

Google Python addon

- Pip install google-api-core==2.11.0
- Pip install google-api-python-client==2.70.0
- Pip install google-auth==2.15.0
- Pip install google-auth-httplib2==0.1.0
- Pip install google-auth-oauthlib==0.8.0
- Pip install google-cloud-core==2.3.2
- Pip install google-cloud-translate==3.9.0
- Pip install googleapis-common-protos==1.57.0
- Pip install googletrans==3.0.0

Google translate Installation

https://googleapis.dev/python/translation/latest/index.html

https://google-auth.readthedocs.io/en/latest/user-guide.html#service-account-private-key-files

https://console.cloud.google.com/apis/credentials?showWizardSurvey=true&project=servicei-ranbir&hl=fr

https://console.cloud.google.com/billing/0030D7-73F8F0-D0405C/settings
https://cloud.google.com/translate/docs/basic/translating-text#translating_input_strings
https://cloud.google.com/translate/docs/advanced/translating-text-v3#translating_text
==========
https://developers.google.com/places/web-service/autocomplete?hl=fr
==========
in Google Cloud access Setup :

1) https://cloud.google.com/translate/docs/setup
2) https://console.cloud.google.com/apis/credentials/serviceaccountkey
3) Download Json API Key

"""

Google translate Token

```
def googleApiToken(version=settings.GOOGLE_API_VERSION):
    """
    pip install --upgrade google-api-python-client
    pip install --upgrade google-auth-oauthlib google-auth-httplib2
    https://cloud.google.com/translate/docs/basic/translating-text
    """

    from google.cloud import translate_v2 as translate
    from google.cloud import translate_v3 as translate3
    from google.oauth2 import service_account

    googleKey           = settings.GOOGLE_API_JSON_KEY
    credentials         = service_account.Credentials.from_service_account_file(googleKey)
    scoped_credentials  = credentials.with_scopes(['https://www.googleapis.com/auth/cloud-platform'])
    translator          = translate.Client(credentials=scoped_credentials)

    return translator
```

EMAILS

POSTFIX

INSTALL POSTFIX

- sudo apt update && sudo apt install postfix mailutils
- sudo apt install postfix
- sudo apt install mailutils

1. Installer `mailutils` (s'il n'est pas déjà là)

bash Copier Modifier

```
sudo apt update
sudo apt install mailutils
```

CONFIG POSTFIX

3. Configurer Postfix pour envoi

Modifie `/etc/postfix/main.cf` :

ini — Copier — Modifier

```ini
myhostname = mail.mon-domaine.com
myorigin = /etc/mailname
mydestination = localhost
relayhost =
inet_interfaces = all
inet_protocols = ipv4
```

Crée `/etc/mailname` :

bash — Copier — Modifier

```bash
echo "mon-domaine.com" | sudo tee /etc/mailname
```

Puis :

bash — Copier — Modifier

```bash
sudo systemctl restart postfix
```

🧱 Étapes à suivre (dans l'ordre)

1. **Prérequis**

- ✅ Un serveur Linux (Ubuntu 20.04+ ou Debian)
- ✅ Un nom de domaine (ex : `mon-domaine.com`)
- ✅ Accès DNS chez ton registrar (GoDaddy, OVH, etc.)
- ✅ Une IP **fixe et propre** (pas blacklistée)

---

2. **Choix du serveur SMTP simple :** `Postfix`

Tu veux **envoyer uniquement**, donc on installe `Postfix` en mode "send-only".

bash Copier Modifier

```
sudo apt update && sudo apt install postfix mailutils
```

À l'écran de config :

- Choisir `Internet Site`
- Nom du mail : `mon-domaine.com` (ou `mail.mon-domaine.com`)

---

3. Configurer Postfix pour envoi

Modifie `/etc/postfix/main.cf` :

ini — Copier — Modifier

```
myhostname = mail.mon-domaine.com
myorigin = /etc/mailname
mydestination = localhost
relayhost =
inet_interfaces = all
inet_protocols = ipv4
```

Crée `/etc/mailname` :

bash — Copier — Modifier

```
echo "mon-domaine.com" | sudo tee /etc/mailname
```

Puis :

bash — Copier — Modifier

```
sudo systemctl restart postfix
```

4. Configurer le reverse DNS (PTR)

- Aller dans ton panneau serveur / hébergeur (AWS, OVH, etc.)
- Associer ton IP publique à `mail.mon-domaine.com`
- Obligatoire pour que Gmail/Outlook ne bloquent pas

---

5. Configurer les enregistrements DNS du domaine

Chez GoDaddy ou autre :

🔹 **SPF :**

txt Copier Modifier

```
mon-domaine.com. IN TXT "v=spf1 mx a ip4:TON.IP.PUBLIQUE -all"
```

🔹 **DKIM :**

→ Optionnel au début, on peut l'ajouter ensuite via `opendkim`.

🔹 **DMARC :**

txt Copier Modifier

```
_dmarc.mon-domaine.com. IN TXT "v=DM ↓ ; p=none"
```

Postfix "send-only"

/etc/postfix/main.cf

🗂️ Résumé de la config modifiée (version clean)

ini

Copier Modifier

```
myhostname = mail.tondomaine.com
myorigin = tondomaine.com
mydestination = localhost
relayhost =
mynetworks = 127.0.0.0/8 [::1]/128
inet_interfaces = loopback-only
inet_protocols = all
smtpd_relay_restrictions = permit_mynetworks permit_sasl_authenticated defer_u
compatibility_level = 3.6
disable_vrfy_command = yes
```

✅ Étapes suivantes

1. **Redémarrer Postfix**

bash — Copier — Modifier

```
sudo systemctl restart postfix
```

2. **Tester l'envoi**

bash — Copier — Modifier

```
echo "Test mail" | mail -s "Hello" ton.email@gmail.com
```

3. **Surveiller les logs**

bash — Copier — Modifier

```
tail -f /var/log/mail.log
```

4. **Configurer SPF dans tes DNS**

txt — Copier — Modifier

```
mon-domaine.com. IN TXT "v=spf1 a mx ip4:TON.IP.PUBLIQUE -all"
```

create a Mail domain

⚡ Pour un usage professionnel / technique

5. Zoho Mail

- 👍 **Avantages :**
  - Solution email pro sans pub, gratuite pour les petits domaines (même pour un domaine personnalisé).
  - Interface moderne.
- 👎 **Inconvénients :**
  - Interface un peu chargée pour un utilisateur non-tech.
  - Configuration de domaine parfois complexe.

4. **Amazon SES (Simple Email Service)** 💪

- 👍 **Avantages :**
  - Ultra économique : **0,10 $ / 1000 mails**.
  - Haute délivrabilité.
  - Parfait pour serveurs sur AWS EC2.
- 👎 **Inconvénients :**
  - **Activation lente et compliquée** (sortir du sandbox, validation DKIM/SPF...).
  - Console AWS pas très intuitive.
- **Activation :** 1h à 24h avec demande manuelle.

👉 Idéal pour usage massif ou serveurs déjà sur AWS, mais pas recommandé pour tester en 5 minutes.

1. **Mailersend** ***(filiale de MailerLite)***

- 🌟 **Avantages :**
  - Interface claire, activation en <10 min.
  - **API** + **SMTP** très faciles à intégrer (curl, Python, PHP...).
  - 3 000 mails/mois **gratuits**, sans carte bancaire.
- 🔧 **Activation :**
  - Ajouter un domaine + valider DKIM/SPF (guidé et clair).
  - Pas de forcing avec CB ou validation à rallonge.
- 💬 **Mon avis :** Très bon rapport simplicité / qualité.
  - https://www.mailersend.com

🔧 Comparatif express

Service	Gratuit	Activation	Facilité	Délivrabilité	Idéal pour
Mailjet	✅ 6k/mois	Très facile	★★★★	★★★	🇫🇷 Startups / tests
SendGrid	✅ 100/jr	Moyen	★★	★★★★	Volume pro
Brevo	✅ 300/jr	Facile	★★★★	★★★	Sites simples
Amazon SES	❌	Compliquée	★	★★★★	Usage massif
Postmark	🔸 100 test	Moyen	★★★	★★★★	Emails critiques

🎯 Conclusion

Service	Délivrabilité	Activation	Gratuité	Idéal pour
Mailersend	★★★★	Très facile	✅	Envois propres & faciles
Resend	★★★★	Ultra rapide	✅	Projet moderne, rapide
Postmark	★★★★★	Sérieux	100 essais	Emails critiques
Elastic	★★★	Simple	✅	Emails basiques

Les gestionnnaires Email

⚠️ Pourquoi c’est devenu un enfer :

Problème	Explication
Blocage à l'inscription	Adresse IP considérée "risquée", compte désactivé sans explication.
Carte bancaire requise	Même pour le plan gratuit, ou pour passer en mode "production"
Rejet automatique	Si domaine, IP ou nom n'inspirent pas confiance à leurs filtres (parfois abusifs).
Validation manuelle lente	Parfois 48h, voire sans réponse.

😡 Pourquoi c’est devenu invivable :

- Tous ces services ont été **submergés de spammeurs** → maintenant ils filtrent tout.
- Si tu n’as pas un domaine + site web + IP propre + CB + bonne tête, ils **te jettent**.
- Même des devs expérimentés passent **1h pour configurer une ligne SPF**.
- **Godaddy = encore pire** → propagation DNS lente, console bugguée.

Alternative SMTP Relay

```python
import smtplib
from email.mime.text import MIMEText

# Gmail : smtp.gmail.com / port 587
# OVH : ssl0.ovh.net / port 587
# Outlook : smtp.office365.com / port 587

smtp_server = "smtp.gmail.com"
smtp_port = 587
username = "ton.email@gmail.com"
password = "motdepasse_app"  # Mot de passe d'application, pas ton mot de pass

msg = MIMEText("Voici un test SMTP depuis mon serveur")
msg["Subject"] = "Test SMTP"
msg["From"] = username
msg["To"] = "destinataire@exemple.com"

with smtplib.SMTP(smtp_server, smtp_port) as server:
    server.starttls()
    server.login(username, password)
    server.send_message(msg)

print("✅ Email envoyé avec Gmail")
```

Test Emails

python

Copier Modifier

```python
import smtplib
from email.mime.text import MIMEText

smtp_host = "smtp.mailersend.net"  # ou Resend, Postmark, etc.
smtp_port = 587
smtp_user = "your_api_user@example.com"
smtp_pass = "your_real_api_key_or_password"

msg = MIMEText("Bonjour depuis mon serveur !")
msg["Subject"] = "Test SMTP"
msg["From"] = "you@yourdomain.com"
msg["To"] = "destinataire@example.com"

with smtplib.SMTP(smtp_host, smtp_port) as server:
    server.starttls()
    server.login(smtp_user, smtp_pass)
    server.send_message(msg)

print("✅ Email envoyé")
```

AWS SES

✅ Option 1 — Amazon SES (recommandé avec EC2)

Avantages :

- Parfaitement intégré à AWS
- Pas besoin de serveur SMTP externe
- Ultra fiable pour les emails transactionnels
- Peut envoyer via **SMTP ou API**

À faire :

1. Crée une identité (adresse ou domaine)
2. Vérifie-la (via lien ou DNS)
3. Demande à sortir du "sandbox" (formulaire rapide)
4. Configure Postfix pour utiliser `email-smtp.us-east-1.amazonaws.com`

🧠 Je peux te faire le fichier de config exact si tu me donnes ton domaine ou ton email d'envoi.

✅ ÉTAPE 1 – Activation de SES

1. Va sur la console AWS SES :

   👉 https://console.aws.amazon.com/ses/home

2. Choisis une **région SES** (ex : `us-east-1` ou `eu-west-1`) — à noter pour plus tard.

3. Clique sur "**Verified Identities**", puis :

   - **Adresse email** : si tu veux envoyer depuis `contact@exemple.com`
   - **Domaine complet** : si tu veux envoyer depuis plusieurs adresses @exemple.com (meilleure option !)

4. Suis les instructions pour :

   - Vérifier ton domaine via les **3 entrées DNS** (SPF, DKIM, vérification)
   - Une fois propagées (via ton registrar), SES affiche "Verified".

✅ ÉTAPE 2 – Demande de sortie du “sandbox”

Par défaut, SES t’empêche d’envoyer vers des emails non vérifiés (mode test).

1. Va ici :

   👉 https://console.aws.amazon.com/ses/home → onglet "Account dashboard"

2. Clique sur : **“Request production access”**

3. Dans le formulaire :

   - Use case = "Transactional emails from a Django/Flask/EC2 server"
   - Volume estimé = 200–500/jour
   - Pas de spam, double opt-in, suppression immédiate si blacklist, etc.

Réponse en 24h max.

✅ ÉTAPE 3 – Générer des identifiants SMTP

1. Dans SES → menu **SMTP Settings**
2. Clique : **"Create SMTP credentials"**
3. Crée un **nouvel utilisateur IAM SES SMTP**
4. Tu obtiens :
    - Un **nom d'utilisateur SMTP (IAM style)**
    - Un **mot de passe SMTP (base64 encoded)**

⚠️ **Note-les**, tu ne les reverras pas après.

✅ ÉTAPE 4 – Configurer Postfix sur EC2

Voici un extrait de configuration pour `/etc/postfix/main.cf` :

ini — Copier — Modifier

```ini
relayhost = [email-smtp.us-east-1.amazonaws.com]:587
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
smtp_sasl_security_options = noanonymous
smtp_use_tls = yes
smtp_tls_CAfile = /etc/ssl/certs/ca-certificates.crt
```

Et dans `/etc/postfix/sasl_passwd` :

markdown — Copier — Modifier

```
[email-smtp.us-east-1.amazonaws.com]:587 YOUR_SMTP_USERNAME:YOUR_SMTP_PASSWORD
```

Puis :

bash — Copier — Modifier

```bash
sudo postmap /etc/postfix/sasl_passwd
sudo chown root:root /etc/postfix/sasl_passwd*
sudo chmod 600 /etc/postfix/sasl_passwd*
sudo systemctl restart postfix
```

✅ ÉTAPE 5 – Test envoi depuis EC2

Utilise un test rapide :

bash Copier Modifier

```bash
echo "Test SES" | mail -s "Hello from EC2" you@example.com
```

Vérifie dans les logs si besoin :

bash Copier Modifier

```bash
tail -f /var/log/mail.log
```

SES EN PAS A PAS

https://eu-central-1.console.aws.amazon.com/ses/home?region=eu-central-1#/**homepage**

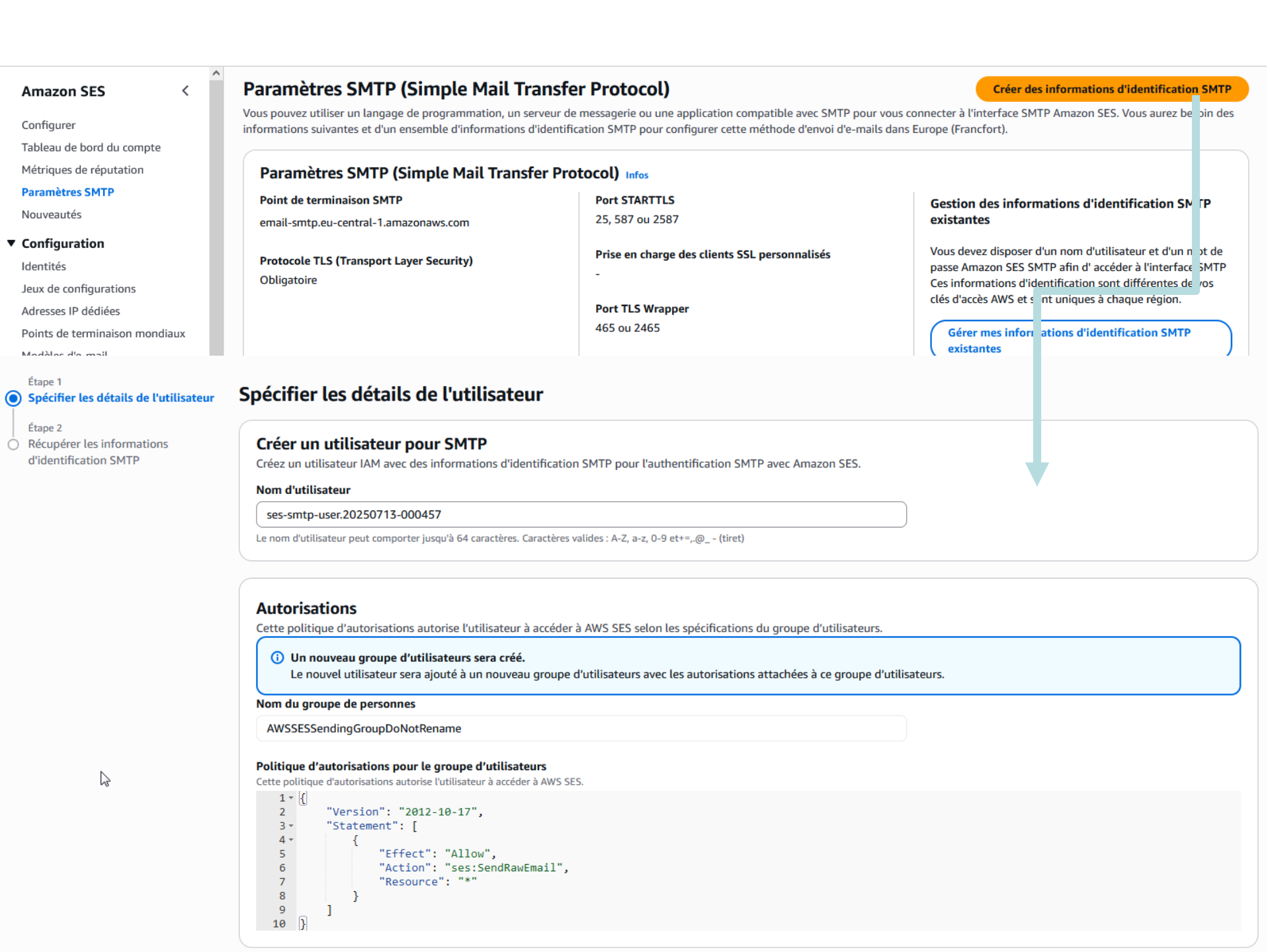
https://eu-central-1.console.aws.amazon.com/ses/home?region=eu-central-1#/**get-set-up**

CREATE SES ID

✳️ [A] Génération du bon identifiant SMTP dans AWS

1. Va dans AWS Console > **SES (Simple Email Service)**.
2. Clique sur **Paramètres SMTP** > puis **Créer des identifiants SMTP**.
3. Cela va :
    - Générer un identifiant de type `AKIA...` (comme un user IAM)
    - Mais **le mot de passe est un mot de passe SMTP transformé, spécialement calculé.**
4. Copie **le SMTP Username** et **le SMTP Password** générés.

> Le mot de passe SMTP ne doit jamais être celui d'un user IAM, sinon → erreur 535.


Amazon SES
Configurer
Tableau de bord du compte
Métriques de réputation
Paramètres SMTP
Nouveautés
Configuration
Identités
Jeux de configurations
Adresses IP dédiées
Points de terminaison mondiaux
Paramètres SMTP (Simple Mail Transfer Protocol)
Créer des informations d'identification SMTP
Vous pouvez utiliser un langage de programmation, un serveur de messagerie ou une application compatible avec SMTP pour vous connecter à l'interface SMTP Amazon SES. Vous aurez besoin des informations suivantes et d'un ensemble d'informations d'identification SMTP pour configurer cette méthode d'envoi d'e-mails dans Europe (Francfort).
Paramètres SMTP (Simple Mail Transfer Protocol) Infos
Point de terminaison SMTP
email-smtp.eu-central-1.amazonaws.com
Protocole TLS (Transport Layer Security)
Obligatoire
Port STARTTLS
25, 587 ou 2587
Prise en charge des clients SSL personnalisés
-
Port TLS Wrapper
465 ou 2465
Gestion des informations d'identification SMTP existantes
Vous devez disposer d'un nom d'utilisateur et d'un mot de passe Amazon SES SMTP afin d' accéder à l'interface SMTP Ces informations d'identification sont différentes de vos clés d'accès AWS et sont uniques à chaque région.
Gérer mes informations d'identification SMTP existantes
Étape 1
Spécifier les détails de l'utilisateur
Étape 2
Récupérer les informations d'identification SMTP
Spécifier les détails de l'utilisateur
Créer un utilisateur pour SMTP
Créez un utilisateur IAM avec des informations d'identification SMTP pour l'authentification SMTP avec Amazon SES.
Nom d'utilisateur
ses-smtp-user.20250713-000457
Le nom d'utilisateur peut comporter jusqu'à 64 caractères. Caractères valides : A-Z, a-z, 0-9 et+=,.@_ - (tiret)
Autorisations
Cette politique d'autorisations autorise l'utilisateur à accéder à AWS SES selon les spécifications du groupe d'utilisateurs.
Un nouveau groupe d'utilisateurs sera créé.
Le nouvel utilisateur sera ajouté à un nouveau groupe d'utilisateurs avec les autorisations attachées à ce groupe d'utilisateurs.
Nom du groupe de personnes
AWSSESSendingGroupDoNotRename
Politique d'autorisations pour le groupe d'utilisateurs
Cette politique d'autorisations autorise l'utilisateur à accéder à AWS SES.
{
"Version": "2012-10-17",
"Statement": [
{
"Effect": "Allow",
"Action": "ses:SendRawEmail",
"Resource": "*"
}
]
}

Étape 1
Spécifier les détails de l'utilisateur

Étape 2
Récupérer les informations d'identification SMTP

Récupérer les informations d'identification SMTP

C'est la seule fois que vous pouvez voir et télécharger ces informations d'identification de sécurité SMTP. Ne communiquez jamais vos informations d'identification SMTP à quiconque.

Informations d'identification SMTP

Nom utilisateur IAM
ses-smtp-user.20250713-000457

Nom d'utilisateur SMTP
AKIAW3KEEWHTUROSBWPM

Mot de passe SMTP
BDgY33JJdBAOKgwknklYK66JUPqSbUOwbMA7Muda9V/s Masquer

Annuler | Télécharger le fichier .csv | Revenir à la console SES

Informations d'identification SMTP

Nom utilisateur IAM

ses-smtp-user.20250713-000457

Nom d'utilisateur SMTP

AKIAW3KEEWHTUROSBWPM

Mot de passe SMTP

BDgY33JJdBAOKgwknklYK66JUPqSbUOwbMA7Muda9V/s **Masquer**