

19/06/2023



Automatisation « IaaS » du déploiement et du MCO de PostgreSQL pour la MAIF avec Ansible

1

**Introduction sur
l'objectif
de l'industrialisation**

2

**Architecture de
déploiement**

3

**Industrialisation
maintenue avec
Ansible**

4

**Focus sur notre
projet Ansible**

1

**Introduction sur
l'objectif
de l'industrialisation**

2

**Architecture de
déploiement**

3

**Industrialisation
maintenue avec
Ansible**

4

**Focus sur notre
projet Ansible**

1. Objectif de l'industrialisation

- Avoir des installations identiques
- N'oublier aucune action
- Tracer les actions
- Faciliter les opérations
- Pouvoir planifier des actions
- Pouvoir proposer du « Dbaas »

1

**Introduction sur
l'objectif
de l'industrialisation**

2

**Architecture de
déploiement**

3

**Industrialisation
maintenue avec
Ansible**

4

**Focus sur notre
projet Ansible**

2. Architecture de déploiement

Projet Ansible

- Choix d'une plateforme d'automatisation : Ansible
- Création d'un projet sous GitHub qui permet de versionner et de sauvegarder les évolutions
- Division du projet en rôles
- Via Ansible, livraison et utilisation de procédures d'un socle technique fourni par notre prestataire "Dalibo"

2. Architecture de déploiement

Base de données référentiel

Création d'une base interne « Référentiel »

Ajout de notions de bases :

- Serveur
- Nom des bases, port...
- Supervision
- Sauvegarde...

Evolution de la base pour y ajouter des informations complémentaires :

- Stockage de métriques
- Synchro CMDB
- Stats ...

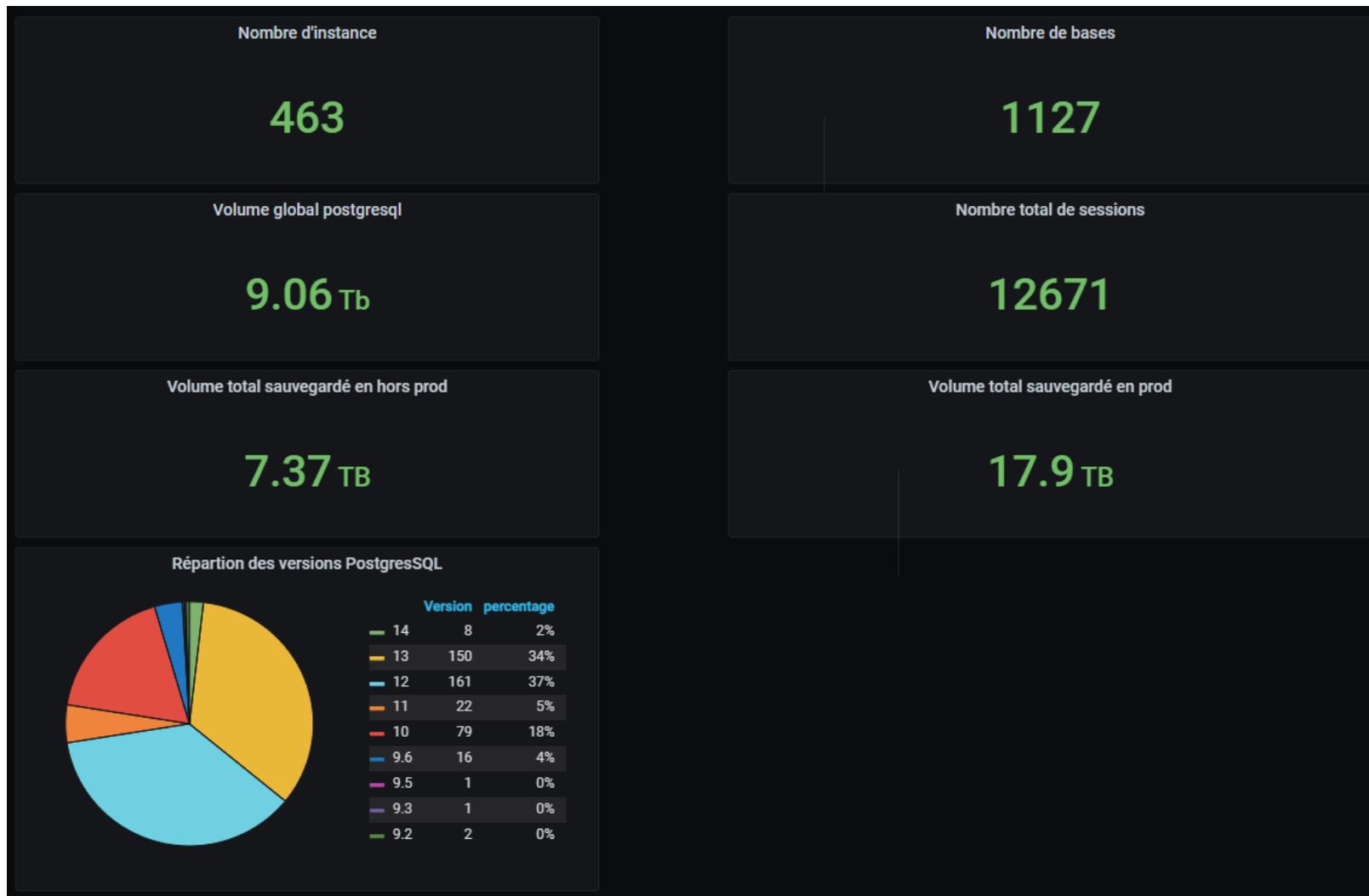
2. Architecture de déploiement

Base de données référentiel

nom_serveur	version	nom_instance	nom_base	si	nom_environnement	sauvegarde	socle	supervision	port	url_powa	vip	role_ha
s	12.4	main_5434	e	MICROSERVICE PAIEMENTS SINISTRES	DEV	aucune	1910	NON		powa-web off	NON	
s	12.4	main_5433	e	MICROSERVICE PAIEMENTS SINISTRES	DEV	aucune	1910	NON		powa-web off	NON	
s	9.6.22	main	e	JAHIA MAIF.FR	PROD	barman	1910	OUI		powa absent		master
s	9.6.22	main	e	JAHIA MAIF.FR	PROD	barman	1910	OUI		powa absent		replica
s	9.6.22	main	e	JAHIA MAIF.FR	PROD	barman	1910	OUI		powa absent		master
s	9.6.22	main	e	JAHIA MAIF.FR	PROD	barman	1910	OUI		powa absent		replica
s	9.3.15	main	a	DATALAKE	PROD	pitrery	172	OUI		powa absent	NON	
s	10.13	main	lt	ADMIN POSTGRESQL	LIV	barman	1111	OUI		powa-web off	NON	
s	12.3	main	lc	SIPI	LIV	barman	1111	OUI		http://s	NON	
s	12.3	main	l	SIPI	LIV	barman	1111	OUI		powa-web off	NON	
s	12.3	main	lt	FRONT OFFICE MAIF.FR (LEGACY)	LIV	barman	1111	OUI		powa-web off	NON	
s	13.3	main	rl	MICROSERVICE GUICHETS	RFU	barman	1111	OUI		powa-web off	NON	
s	13.3	main	m	MICROSERVICE GUICHETS	MCOR	barman	1111	OUI		powa-web off	NON	
s	13.3	main	te	MICROSERVICE GUICHETS	TCOR	barman	1111	OUI		powa-web off	NON	
s	13.3	main	td	MICROSERVICE GUICHETS	TCOR	barman	1111	OUI		powa-web off	NON	
s	14.4	main	c	KELFICHE (ST)	PPCOR	barman	1111	OUI		powa-web off	NON	
s	14.4	main	c	KELVALEUR D ACHAT (ST)	PPCOR	barman	1111	OUI		powa-web off	NON	
s	14.4	main	c	KELVALEUR D ACHAT (ST)	PPCOR	barman	1111	OUI		powa-web off	NON	

2. Architecture de déploiement

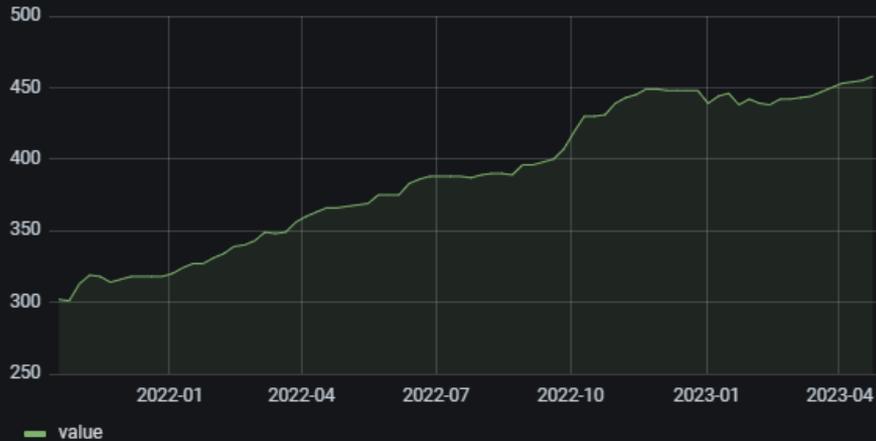
Base de données référentiel



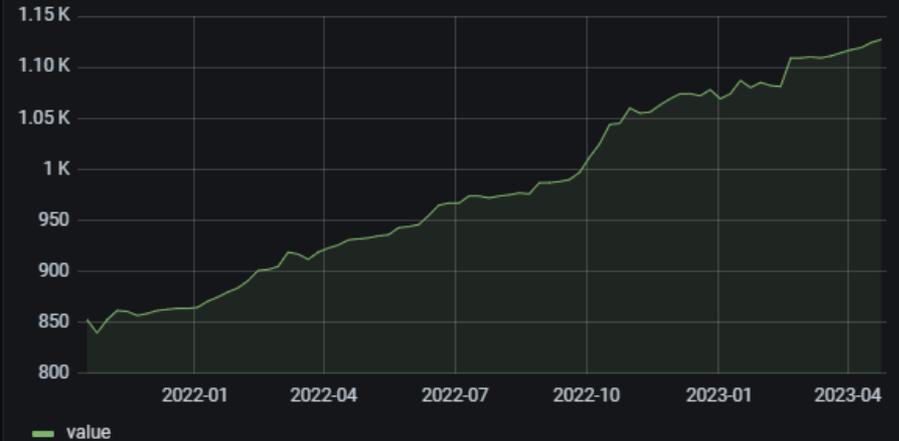
2. Architecture de déploiement

Base de données référentiel

Evolution du nombre d'instance



Evolution du nombre de bases



Evolution de la taille global des bases postgresql



Evolution du nombre total de session



2. Architecture de déploiement

Utilisation d'AWX

Utilisation d'AWX version communautaire de TOWER

This screenshot shows the 'Survey' step of an AWX configuration. The left sidebar lists the steps: 1 Inventory, 2 Other prompts, 3 Survey (selected), and 4 Preview. The main form area is titled 'Listeserveurs' and contains the following fields:

- Listeserveurs ***: An empty text input field.
- Version de postgres ***: A dropdown menu with '13' selected.
- Taille du Volume de donnees ***: A text input field with '20G'.
- Taille du volume des journaux d archivage ***: A text input field with '5G'.
- Port d'ecoute postgres ***: A text input field with '5432'.
- Nom de la base ***: An empty text input field.

At the bottom of the form are three buttons: 'Next' (highlighted in blue), 'Back', and 'Cancel'. A vertical navigation bar on the right side of the form shows the same four steps, with 'Survey' highlighted in blue.

This screenshot shows the 'Survey' step of an AWX configuration. The left sidebar lists the steps: 1 Inventory, 2 Other prompts, 3 Survey (selected), and 4 Preview. The main form area is titled 'Utilisateur de la base' and contains the following fields:

- Utilisateur de la base ***: An empty text input field.
- Mot de passe ***: An empty text input field.
- Environnement ***: An empty text input field.
- Si ***: An empty text input field.
- Vip ***: An empty text input field.
- Haute Disponibilité**: A dropdown menu with 'non' selected.

At the bottom of the form are three buttons: 'Next' (highlighted in blue), 'Back', and 'Cancel'. A vertical navigation bar on the right side of the form shows the same four steps, with 'Survey' highlighted in blue.

2. Architecture de déploiement

Portail MAIF

Ajout d'un portail pour rendre les utilisateurs plus autonomes.

Utilisation du portail « ASAP »

The screenshot displays the 'ASAP Portail Automation' interface. The header is blue with the ASAP logo and the text 'Automatisation des Services Agilité des Process'. The main content area shows a search for 'postgres' with 6 elements found. Below the search results, there are six service cards, each with a PostgreSQL icon, a title, a description, the group and service names, and a 'DEMANDER' button.

Service	Description	Groupe d'activité	Service
Ajout d'une base Postgresql	Ajout d'une base Postgresql sur une instance existante	BG-CLOUD	Services
Création d'une base Postgresql	Déploiement d'une base Postgresql	BG-CLOUD	Services
Création d'une base Postgresql POC	Création d'une base Postgresql pour du POC	BG-CLOUD-POC	Proof Of Concept
Désinstallation instance et base Postgresql	Désinstallation d'instance et de base Postgresql sur un serveur existant	BG-CLOUD	Services
Gestion base Postgresql	Prolongation, mise en service, suppression d'une base PostgreSql et du ou des serveurs associés	BG-CLOUD	Services
Installation instance et base Postgresql	Installation d'une instance Postgresql et création d'une base sur un serveur existant	BG-CLOUD	Services

2. Architecture de déploiement

The screenshot displays the 'Portail Automation' interface. At the top, there is a navigation bar with 'Catalogue', 'Déploiements', and 'Boîte de réception'. Below this, a search bar is present with the text 'Rechercher des éléments de catalogue par nom ou par description' and a filter icon. The main content area is a grid of 12 service cards, each with an icon, a title, a description, activity group, service name, and a 'DEMANDER' button.

Service	Description	Groupe d'activité	Service
Ajout d'une base Postgresql	Ajout d'une base Postgresql sur une instance existante	BG-CLOUD	Services
Création d'alias DNS	Permet de créer un alias DNS sur un hôte	BG-CLOUD	Services
Création d'une base Postgresql	Déploiement d'une base Postgresql	BG-CLOUD	Services
Création d'une base Postgresql POC	Création d'une base Postgresql pour du POC	BG-CLOUD-POC	Proof Of Concept
Demande de mise en production de serveurs	mise en production de serveur ainsi que recette conformité.	BG-CLOUD	Services
Demande d'environnement	Demande d'environnement pour construction de VM	BG-CLOUD	Services
Demande de serveurs physiques	Demande de serveurs physiques	BG-CLOUD	Services
Désinstallation instance et base Postgresql	Désinstallation d'instance et de base Postgresql sur un serveur existant	BG-CLOUD	Services
Devis Chiffrage Environnement(s)	Calcul coût infrastructure liée à une demande d'Environnement	BG-CLOUD	Construction de VM
Gestion base Postgresql	Prolongation, mise en service, suppression d'une base PostgreSql et du ou des serveurs associés	BG-CLOUD	Services
Installation instance et base Postgresql	Installation d'une instance Postgresql et création d'une base sur un serveur existant	BG-CLOUD	Services
Linux RedHat 8 POC	Déploiement de VM Linux RedHat dernière version (8.6) pour du POC	BG-CLOUD-POC	Proof Of Concept

2. Architecture de déploiement

Portail MAIF

Portail ASAP

Service Installation POSTGRESQL

Bonjour

La création de la base de données postgresql m_demo_pgday a été effectuée avec succès !

Votre base de données est prête à être utilisée ▶

Mode d'installation : **Standalone**
Version postgresql : **14**
Alias de connexion à la base de données : **mpgbd-demopgday.maif.local**
Nom de la base de données : **m_demo_pgday**
Utilisateur de la base de données : **usr_demo_pgday**
Mot de passe : **usr_demo_pgday**
Port d'accès à la base de données : **5432**
Volume pour les données : **20 Go**
Volume pour les logs : **10 Go**

1

**Introduction sur
l'objectif
de l'industrialisation**

2

**Architecture de
déploiement**

3

**Industrialisation
maintenue avec
Ansible**

4

**Focus sur notre
projet Ansible**

3. Industrialisation maintenue avec Ansible

VM Linux standardisées pour PostgreSQL

- Red Hat EL 7 et 8
- Paramétrage OS standard pour PostgreSQL
- CPU, mémoire et stockage en fonction des choix sur le portail "ASAP"
- Volumes groups root0fvg (système) et data0fvg (appli/données)
- Partitions dans data0fvg :
 - Partitions techniques
 - Deux partitions pour l'instance PostgreSQL :
 - /var/lib/pgsql/{version}/main/data (données)
 - /var/lib/pgsql/{version}/main/pg_xlog (WAL)

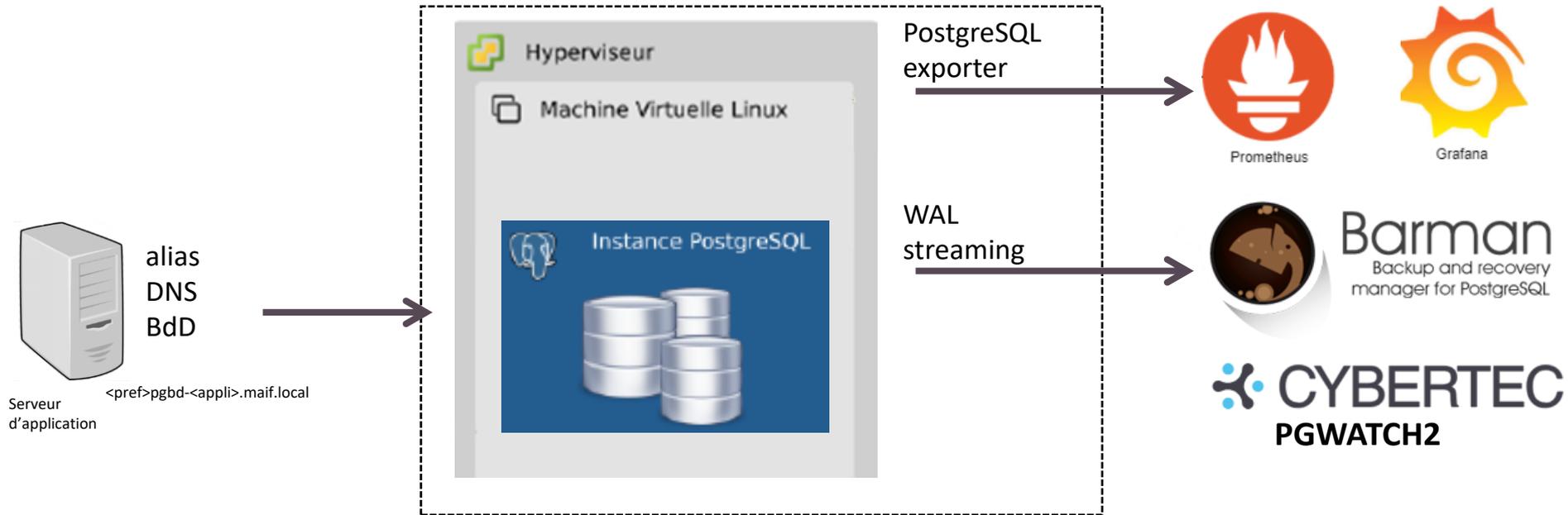
3. Industrialisation maintenue avec Ansible

Deux niveaux de service au catalogue C1 et C2

- En commun :
 - Plan de sauvegarde standard (barman avec WAL streaming en temps réel)
 - Objectif sans perte de données quel que soit le service
- Criticité 2 (C2) en standalone sur une seule VM
 - Disponibilité de service moindre
- Criticité 1 (C1) en cluster sur deux VM
 - Réplication "maitre-réplica" en WAL streaming géré par **patroni** :
 - **Protection élevée des données** grâce à la réplication
 - **Disponibilité de service maximale** avec le basculement automatique sur le réplica.

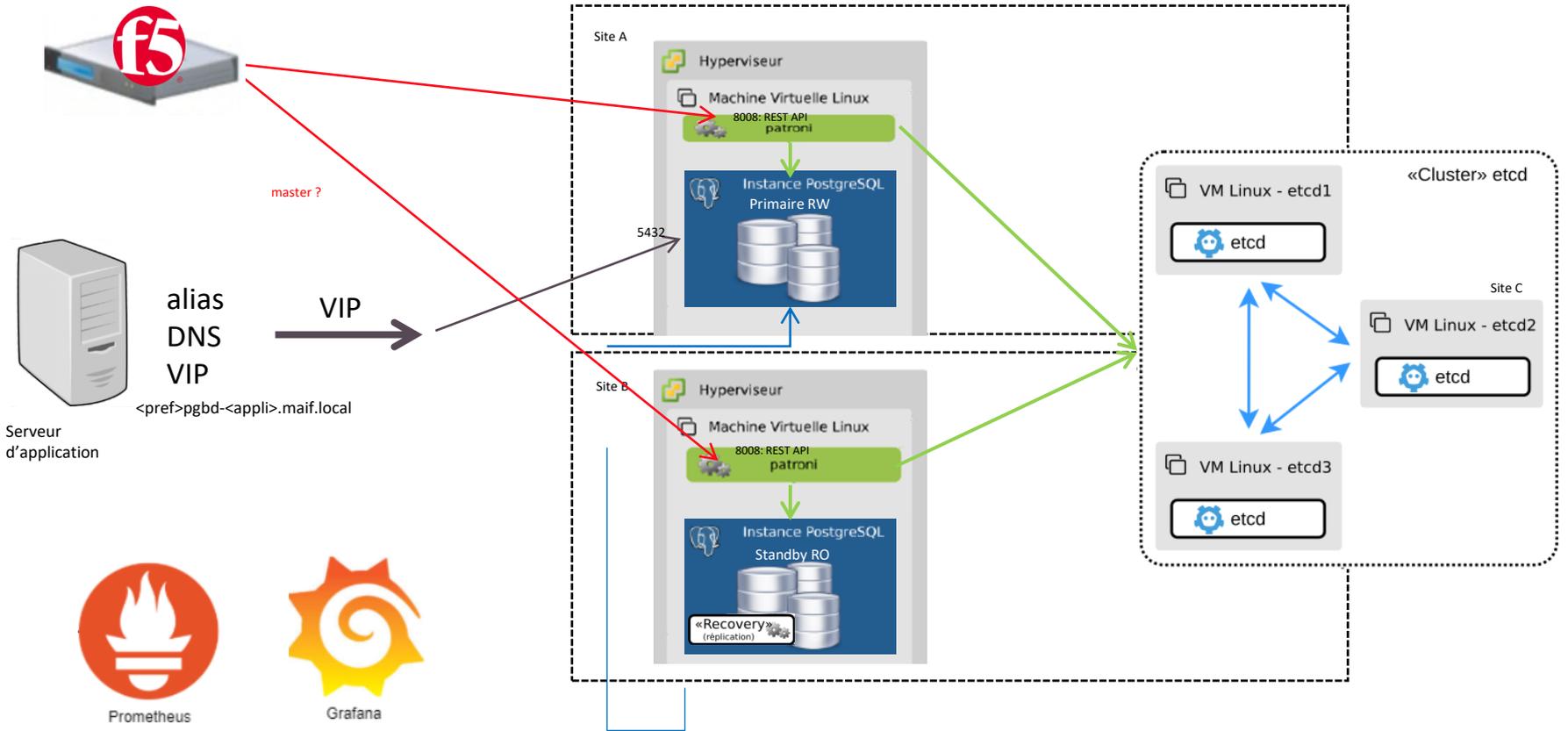
3 Industrialisation maintenue avec Ansible

Service C2 en standalone sur une VM



3 Industrialisation maintenue avec Ansible

Service C1 avec patroni



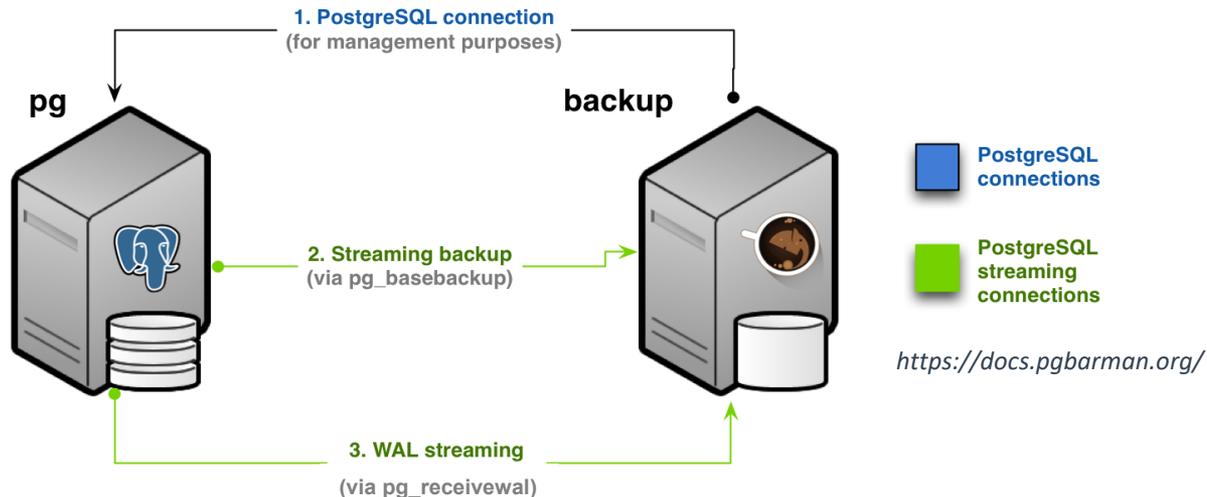
Barman
Backup and recovery manager for PostgreSQL



3. Industrialisation maintenue avec Ansible

Sauvegarde avec barman

- Sauvegarde centralisée sur 2 serveurs pour les réseaux de "BUILD" et "PROD"
 - Récupération "WAL streaming" temps réel des archives de transactions
 - Stratégie de sauvegarde locale sur NAS avec rétention de 8 jours
 - Copie sur l'infrastructure de sauvegarde NetBackup avec rétention de 2 mois
- ✓ Evolution prochaine vers PgBackRest avec deux destinations dont du stockage S3.



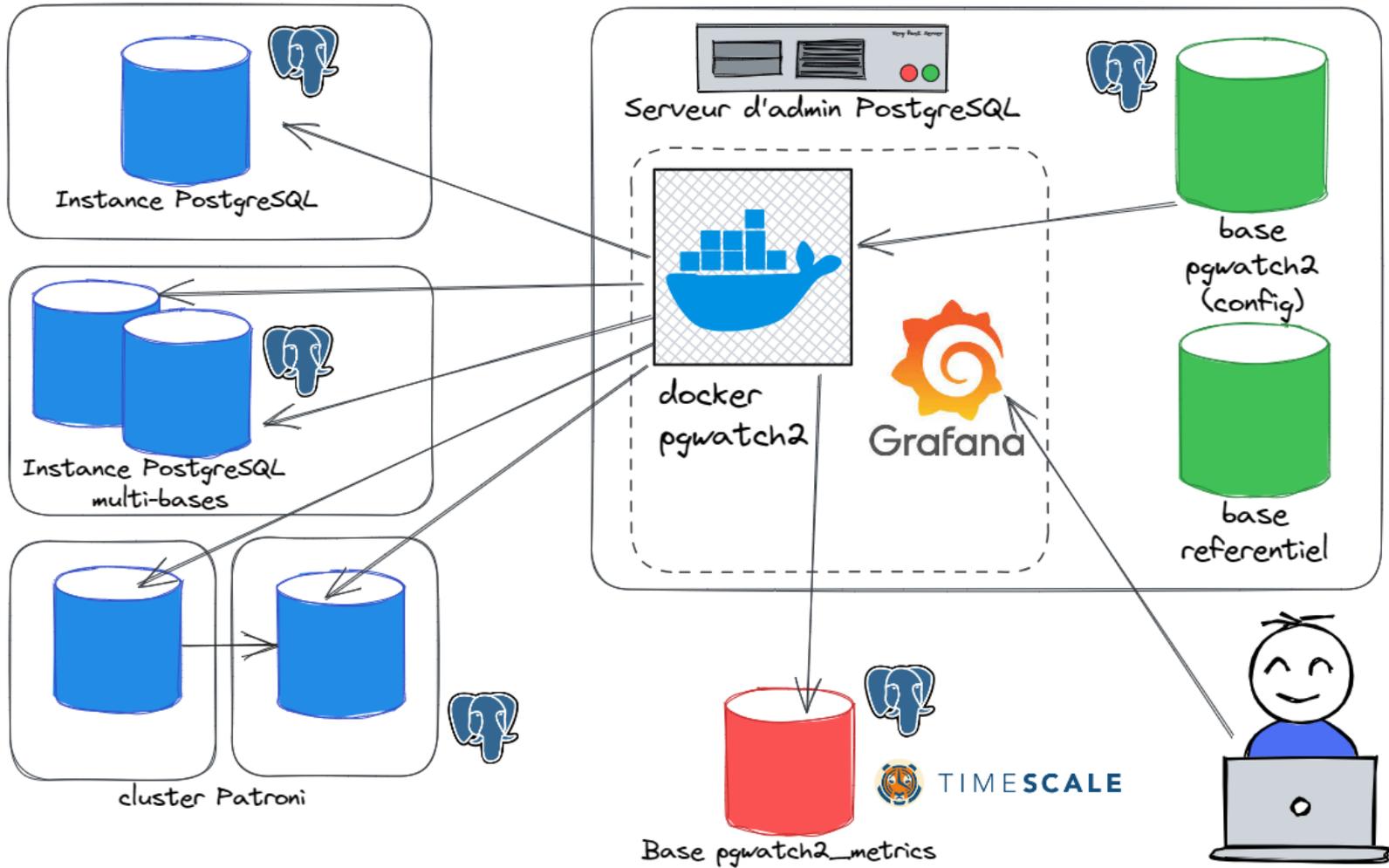
3. Industrialisation maintenue avec Ansible

Supervision et métrologie

- Supervision avec un agent "postgres exporter" sur toutes VM et l'agent "patroni exporter" sur les clusters.
- PoWA et ses extensions dans une base "powa"
- Service powa-web stoppé par défaut
- PgWatch2 sur notre serveur d'exploitabilité :
 - PgWatch2 dans un docker avec son serveur Grafana
 - Sa base de données de configuration
 - Sa base de données des mesures collectées sur une VM "PostgreSQL+TimescaleDB"

3. Industrialisation maintenue avec Ansible

Architecture PgWatch2



1

**Introduction sur
l'objectif
de l'industrialisation**

2

**Architecture de
déploiement**

3

**Industrialisation
maintenue avec
Ansible**

4

**Focus sur notre
projet Ansible**

4. Focus sur notre projet Ansible Playbooks avec appels de rôles pour la modularité

Utilisation de git pour le stockage du code



1 projet contenant :

- › Des *playbooks* → Utilisés en « *Templates AWX* »
- › Des *roles* → Utilisés par les *playbooks*

```
PGSQL_INST_PLAYBOOK
├── defaults
├── files
├── group_vars
├── images
├── roles
├── ! add_barman.yml
├── ! add_certificate.yml
├── ! add_ldap.yml
├── ! add_user_certificate.yml
├── ! ajout_fonctions_communes.yml
├── ! alter_user_pwd.yml
├── ! change_pwd.yml
├── ! check_barman.yml
├── ! check_certificate.yml
├── ! cmdb_gen_flux_json.yml
├── ≡ ectd_inventory
├── ! get_liste_schemas.yml
├── ! installation_barman.yml
├── ! installation_client.yml
├── ! installation_etcd.yml
├── ! installation_ha.yml
├── ! installationpg_background.yml
├── ≡ inv_pg_background
├── ≡ inventory_postgresql
├── ! maj_fonction_cmdb_gen_flux_json.yml
├── ! mco_donnees_stockage.yml
├── ! postgresql_add_db.yml
├── ! postgresql_add_exporter.yml
├── ! postgresql_add_oracle_fdw.yml
├── ! postgresql_add_patroni_exporter.yml
├── ! postgresql_add_pgwatch2.yml
├── ! postgresql_add_postgres.yml
```

4. Focus sur notre projet Ansible

Playbooks avec appels de rôles pour la modularité

Utilisation de git pour le stockage du code

1 projet contenant :

- › Des *playbooks* → Utilisés en « *Templates AWX* »
- › Des *roles* → Utilisés par les *playbooks*

```
▼ roles
  > add_agent_kibana
  > add_atu
  > add_barman
  > add_certificate
  > add_cmdb_patroni
  > add_common_fcn
  > add_db
  > add_etcd
  > add_exporter
  > add_front
  > add_ha
  > add_ldap
  > add_oracle_fdw
  > add_patroni_exporter
  > add_pgwatch2
  > add_powa
  > add_preload_libraries
  > add_temboard_agent
  > add_user
  > add_user_certificate
  > alter_user_pwd
  > change_pwd
  > change_si
  > check_barman
  > check_certificate
  > cp_multi_instances
  > deploy_barman
  > fs_pg_wal
  > installation_barman
  > installation_client
  > installation_socle
  > liste_schemas
```

4. Focus sur notre projet Ansible

Playbooks avec appels de rôles pour la modularité

Exemple de playbook `postgresql_upgrade.yml`:

```
---  
  
roles:  
- { role: pre_upgrade_ha, tags: pre_upgrade_ha, when: ha == "oui" }  
- { role: pre_upgrade_instance, tags: pre_upgrade }  
- { role: upgrade_instance, tags: upgrade }  
- { role: post_upgrade_instance, tags: post_upgrade }  
- { role: add_powa, tags: add_powa }  
- { role: add_ha, tags: add_ha, when: ha == "oui" }  
- { role: post_upgrade_barman, tags: post_upgrade_barman }  
- { role: add_cmdb_patroni, when: ha == "oui" }  
  
- fail:  
  msg: "version_socle < 191 qui est non compatible avec ce playbook. Mettre a jour le socle"  
  when: version_socle |int < 191
```

4. Focus sur notre projet Ansible

Playbooks avec appels de rôles pour la modularité

Exemple de rôle :

```
- name: Récupération du ou des serveurs a traiter
  postgresql_query: <6 keys>
  register: list_serveurs
  delegate_to: "{{serveur_admin}}"

- name: Creation du répertoire de travail
  file: <5 keys>
  delegate_to: "{{serveur_admin}}"

- name: Generation des templates ldap2pg
  template: <5 keys>
  with_items: "{{ list_serveurs.query_result }}"
  delegate_to: "{{serveur_admin}}"

- name: copy du ldaprc
  template: <5 keys>
  delegate_to: "{{serveur_admin}}"

- name: synchronisation AD
  become: true
  become_user: root
  shell: "docker run --rm -v {{ ldap2pg_dir }}/ldap2pg_{{item.nom_serveur}}.yaml:/workspace/ldap2pg.yaml -v {{ ldap2pg_dir }}:/ldaprc:/workspace/ldaprc dalibo/ldap2pg {{Dryrun_mode}}"
  with_items: "{{ list_serveurs.query_result }}"
  no_log: true
  delegate_to: "{{serveur_admin}}"

- name: suppression des templates
  file: <2 keys>
  with_items: "{{ list_serveurs.query_result }}"
  delegate_to: "{{serveur_admin}}"
```

4. Focus sur notre projet Ansible

Utilisation des méthodes Ansible pour PostgreSQL

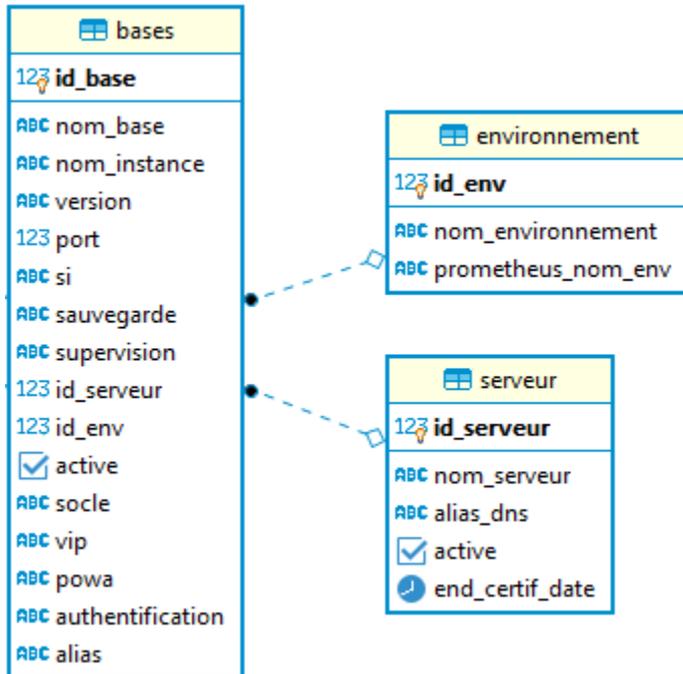
Collection Ansible utile :

Community.Postgresql : une vingtaine de *modules*

- › *postgresql_db* – Ajouter/Supprimer des databases
 - › “_ext - des extensions
 - › “_idx – des indexes
 - › “_privs – des privileges
- › *postgresql_info* – Récupérer des informations sur une instance Pg : databases, mode recovery, slots de réplication, rôles, settings, version...
- › *postgresql_query* – Jouer des requêtes SQL sur une base

4. Focus sur notre projet Ansible fonction centrale de notre référentiel

Une base « référentiel » comme point de centralisation des informations



Utilisation de « *postgresql_query* » pour mettre à jour le référentiel via les playbooks.

4. Focus sur notre projet Ansible

Chiffrement des mots de passe avec Vault

```
! all.yml M ● ⓘ README.md
group_vars > all > ! all.yml
79 #####
80 # Variables passwords vaultés #
81 #####
82
83 # DB user "postgres"
84 postgres_postgres_pwd: !vault |
85     $ANSIBLE_VAULT;1.1;AES256
86     373563303537616664653862393064643965313033633132323138383139636237326431616666562
87     3665383861363763383231663238623131623935616531340a633830626135666464316238623433
88     39633961343264396637646238326261663039356537386333313966343932383562356234383762
89     6465353032643965640a656532363733643764343966393836616664626437303234373263333731
90     3364
91
92 # DB user "powa"
93 > postgres_powa_pwd: !vault | ...
100
101 # DB user "postgres_exporter"
102 > postgres_postgres_exporter_pwd: !vault | ...
109
110 # DB user "replication"
111 > postgres_replication_pwd: !vault | ...
118
119 # DB user "backup"
120 > postgres_backup_pwd: !vault | ...
```

4. Focus sur notre projet Ansible

Gestion des certificats pour l'authentification TLS

Rôles Ansible :

- › Demande de certificat à la PKI
- › Envoi du certificat + clé privée sur destination
- › Configuration postgresql.conf (ssl=on, ssl_cert_file, ssl_key_file) + pg_hba.conf