h e p i a

Haute école du paysage, d'ingénierie
et d'architecture de Genève

PostgreSQL Meetup 23 Mai 2019

Industrialisation et Haute Disponibilité

PostgreSQL User Group Genève

h e p i a

Haute école du paysage, d'ingénierie
et d'architecture de Genève

dbi services

Dr Paul Albuquerque
Responsable de la filière Informatique à l'école HEPIA

OpenDB Appliance - **Pierre Sicot** – Dbi-Services

TeamBoard - **Pierre Giraud** - Dalibo

La réplication encore plus facile - **Pierre Boizot** - freelance

Déploiement automatisé d'un cluster PostgreSQL - **Fabrice Chapuis** - Banque Lombard Odier

PostgreSQL

(liberal Open Source license, similar to the BSD or MIT licenses)

https://www.postgresql.org/about/licence/

- Alternative aux logiciels propriétaires (maturité du produit)

- Communauté forte (nombreux contributeurs)

- Code ouvert

- Support (plusieurs sociétés d’experts)

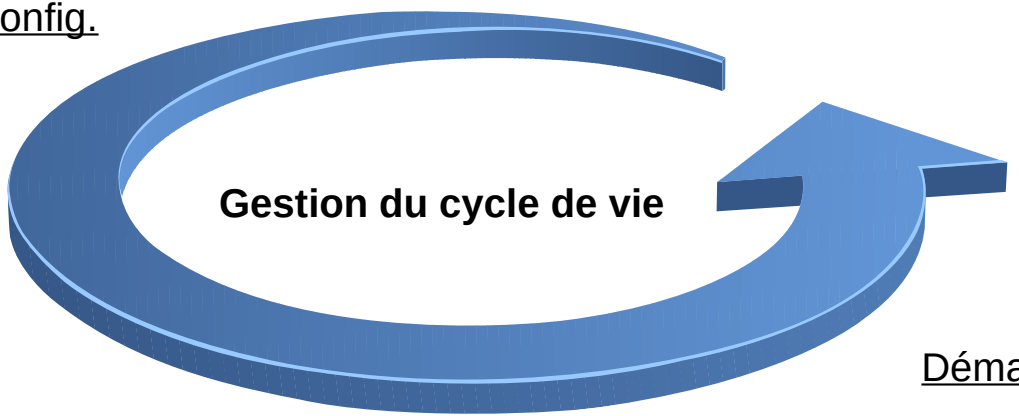
- Intégration avec d’autres logiciels libres

Industrialisation

- Taille de l’infrastructure
- Gestion des tâches opérationnelles
- Demandes de nouvelles installations
- Standardisation
- **DbaaS**

Déploiement

Mise à jour du schema
Copie
Modification de la config.



Démantèlement

Upgrade du SGBD
Patch

Migration

Outil de déploiement

quelle valeur ajoutée?

- **Modules** versus SHELL Scripts
- Langage déclaratif
- Eviter les erreurs (fiabilisation du déploiement)
- Consistance sur l’ensemble des serveurs
- Gain de temps
- **Idempotence**

SQL – idempotent?

Update, Delete, Insert

```
postgres=# CREATE TABLE sometable (a int primary key, b int, c int);
CREATE TABLE
postgres=# INSERT INTO sometable (a,b,c) values (1,3,2) ON CONFLICT (a)  DO NOTHING;
INSERT 0 1
postgres=# INSERT INTO sometable (a,b,c) values (1,3,2) ON CONFLICT (a)  DO NOTHING;
INSERT 0 0
postgres=# INSERT INTO sometable (a,b,c) values (1,2,3) ON CONFLICT (a)  DO NOTHING;
INSERT 0 0
postgres=#
```

	Companie	Method	Language	Licence	Written	Last release
	Puppet (Luke Kanies in 2005)	Pull	Declaratif	Apache from 2.7.0, GPL before	C++, Clojure and Ruby	V.6.4
	Chef	Pull	Decl / Imp	Apache License	Ruby (client) and Ruby / Erlang (server)	V.12.19
	Redhat Inc.	Push	Decl / Imp	GNU General Public License	Python	2.7*

* Python 2 (version 2.6 ou ultérieure) ou Python 3 (version 3.5 ou ultérieure).
V. 2.8 16.5.2019

https://en.wikipedia.org/wiki/Comparison_of_open-source_configuration_management_software

```
---
YAML:
    YAML Ain’t Markup Language:
        - Playbook:
        - Indentation avec des espaces: comme Python
        - Format compact: [‘facilement lisible’, ‘auto-documenté’]
        - Superset du JSON:  # Commentaires
        - Types: [‘mappings (hashes / dictionaries)’,’sequences (arrays / lists)’, ‘scalars
            (strings / numbers)’]
...
```

Inventaire (cibles)

INI-like (Ansible's defaults)

Environnement
Variables d'inventaire
(group_vars)

Playbook (tâches)
Rôles (local/externe)

https://www.ansible.com/

Modules

tps://docs.ansible.com/ansible/latest/user_guide/modules.html

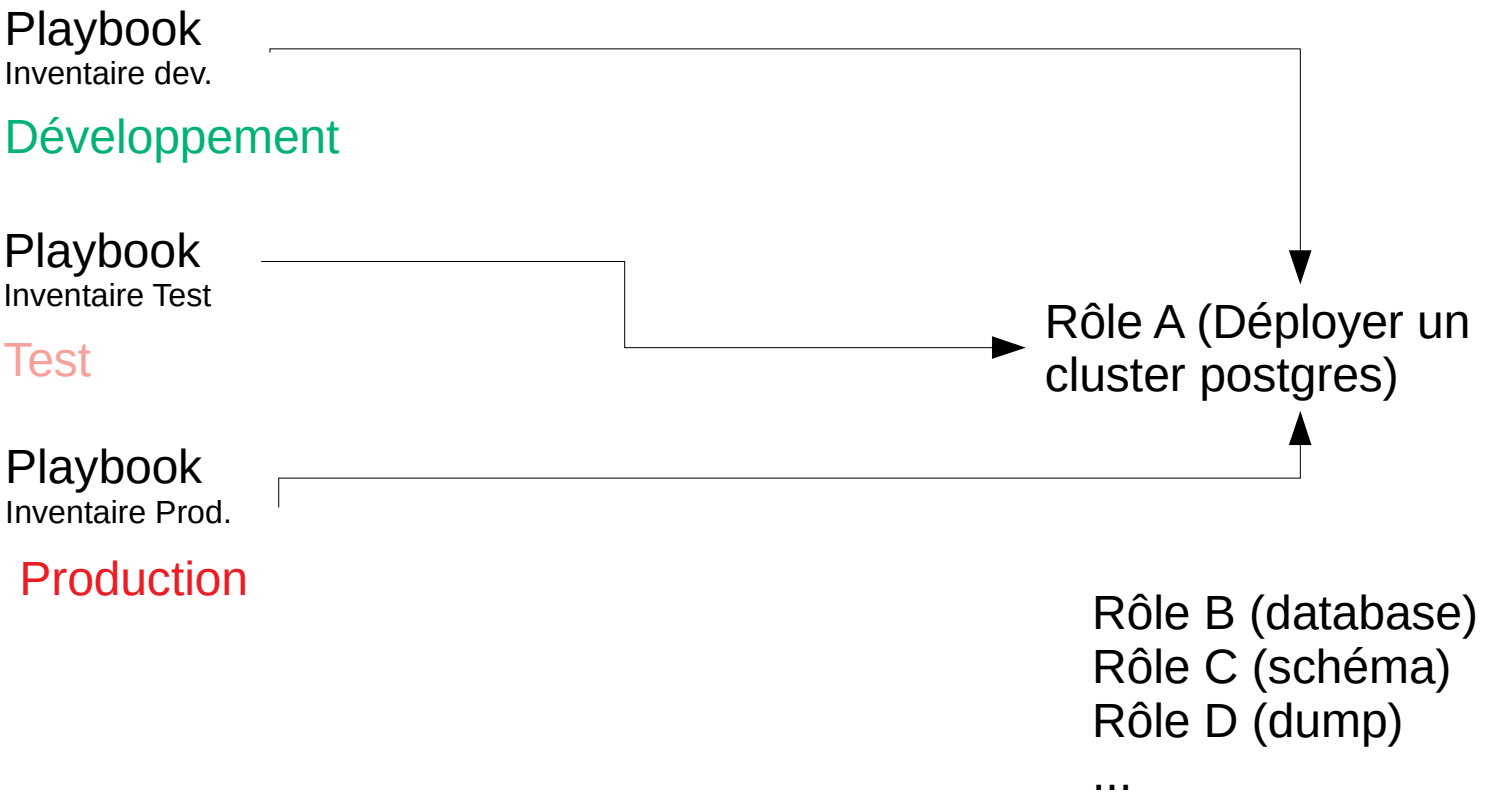
SSH

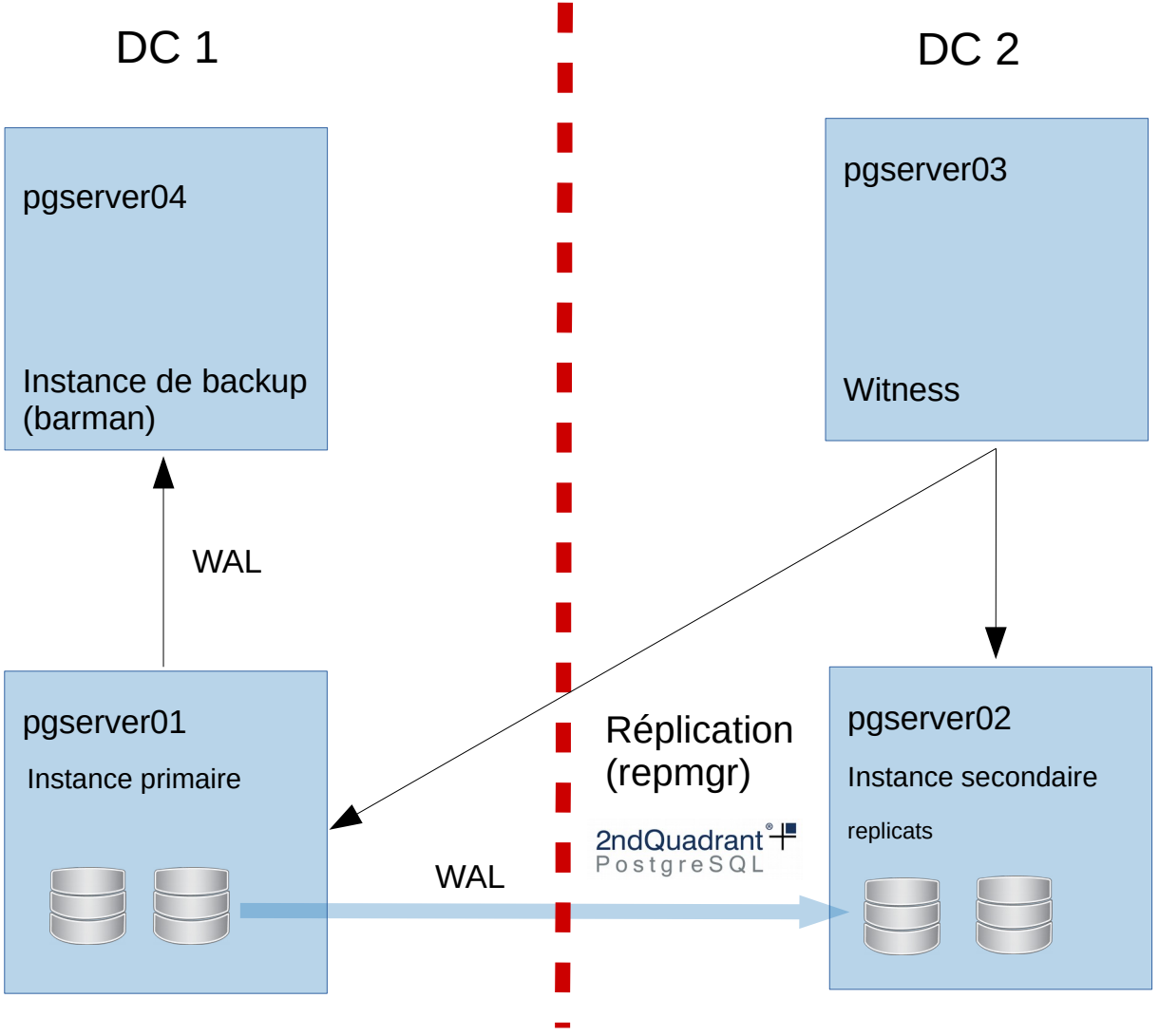
ansible -m ping <<hostname>>

Server 1
Primary

Server 2
Secondary

Server 3
Witness

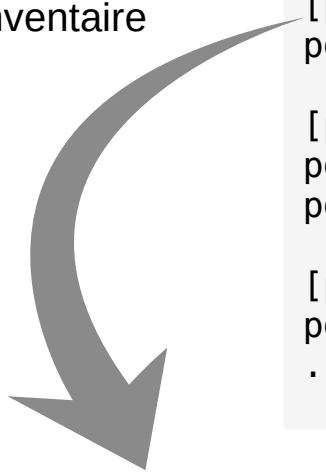

Playbook
Inventaire dev.
Développement
Playbook
Inventaire Test
Test
Playbook
Inventaire Prod.
Production
Rôle A (Déployer un cluster postgres)
Rôle B (database)
Rôle C (schéma)
Rôle D (dump)
...



Structure: ($PGDATA,log)
Configuration
(postgresql.conf,
pg_hba.conf)

Backup
(pg_basebackup,
pg_dump)
Purge des archives
SSL
Logrotate

Inventaire



```
# PostgreSQL targets
[postgresql:children]
pg-cluster

[pg-cluster]
pgserver01  # Primary
pgserver02  # Secondary

[primary-pg-cluster]
pgserver01
...
```

Variables d’inventaire

```
postgresql:
  cluster_name: 'somecluster'
  cluster_port: 5432
  shared_buffers: 512MB
  huge_pages: try
  work_mem: 8MB
  max_connections: 100
  archive_mode: 'on'
  archive_command: 'cp %p archives/%f'
  archive_timeout_in_seconds: 3600
  database_name: 'somedatabase'
  database_user: 'bob'
  ...
```

Playbook Ansible
(cluster)

```
---
- name: PostgreSQL - deploy cluster
  hosts: pg-cluster
  gather_facts: no

  vars:
  ...
  roles:
    - role: deploy-cluster
    vars:
      postgresql_cluster_name: "{{ postgresql.cluster_name }}"
      postgresql_cluster_port: "{{ postgresql.cluster_port }}"
      postgresql_shared_buffers: "{{ postgresql.shared_buffers }}"
...
```

Playbook Ansible
(database)

```
---
- name: PostgreSQL - deploy database
  hosts: pg-cluster:&primary-pg-cluster
  gather_facts: no

  vars:
  roles:
    - role: deploy-database
      vars:
        postgresql_cluster_name:"{{ postgresql.cluster_name }}"
        postgresql_cluster_port: "{{ postgresql.cluster_port }}"
        postgresql_database_name: "{{ postgresql.database_name }}"
        postgresql_user_name: "{{ postgresql.database_user }}"
...
```

Rôle deploy-cluster (fichier main.yml)

```
- import_tasks: preflight.yml
  tags:
    - preflight

 - import_tasks: install.yml
   tags:
     - install

- import_tasks: configure.yml
  tags:
    - configure

- import_tasks: postflight_primary.yml
  tags:
    - postflight_primary

- import_tasks: postflight_secondary.yml
  tags:
    - postflight_secondary
```

check input parameters

Install packages

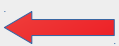
create structure / install fichiers de config.

Tâches spécifiques pour l'instance primaire: initdb + check

Tâches spécifiques pour l'instance secondaire: clone db + check

Rôle deploy-cluster (primaire)

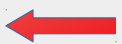
```
- name: "{{ postgresql_cluster.name }} - initialize cluster"
  shell: "{{ postgresql_paths.program.path }}/bin/initdb -D {{ postgresql_paths.cluster.path }}
--data-checksums --locale={{ postgresql_locale }} --encoding={{ postgresql_encoding }}"
  args:
    creates: "{{ postgresql_paths.cluster.path }}/PG_VERSION"
  become: true
  become_user: postgres
  register: out_init_cluster
  when:
   - primary
```



primary: "{{ inventory_hostname in groups['postgresql-primary'] }}" (preflight)

Rôle deploy-cluster (standby)

```
- name: "{{ postgresql_cluster.name }} - clone primary database"
  shell: /usr/pgsql-10/bin/repmgr -h "{{ master_node }}" -p "{{ postgresql_cluster_port }}" -U
     repmgr -d repmgr -f /etc/{{ postgresql_cluster_name }}/repmgr.conf -F standby clone
  args:
    creates: "{{ postgresql_paths.cluster.path }}/PG_VERSION"
  become: true
  become_user: postgres
  become_method: sudo
  register: out_clone_primary
  when:
     - not primary
```



Rôle deploy-database (main.yml)

```
---
  import_tasks: preflight.yml
  tags:
    - preflight

- import_tasks: deploy-user.yml
  tags:
    - deploy-user

- import_tasks: deploy-database.yml
  tags:
    - deploy-database

-  import_tasks: postflight.yml
   tags:
    - postflight
```

check input parameters

Création du db owner

Création de la db + grants

Contrôle que les tâches se sont bien exécutées (test des accès à la base)

Rôle deploy-database

```
- name: "{{ postgresql_cluster.name }} - create dbuser"
  postgresql_user:
    login_user: postgres
    port: "{{ postgresql_cluster_port }}"
    name: "{{ postgresql_database_user }}"
    password:
    role_attr_flags:NOSUPERUSER,NOCREATEDB,NOCREATEROLE
    state: present
    become: true
    become_user: postgres
    become_method: sudo
```

Modules Ansible

```
- name: "{{ postgresql_database_name }} - create database"
  postgresql_db:
    login_user: postgres
    name: "{{ postgresql_database_name }}"
    port: "{{ postgresql_cluster_port }}"
    owner: "{{ postgresql_database_user }}”
    state: present
    become: true
    become_user: postgres
```

Moteur de templating http://jinja.pocoo.org/docs/2.10/

jinja2 format (.j2)
Postgresql.conf.j2

Variabilisé (gérer plusieurs environnements)

```
#------------------------------------------------------------------
# RESOURCE USAGE (except WAL)
#------------------------------------------------------------------
# - Memory -
shared_buffers = {{ postgresql_shared_buffers }}

huge_pages = {{ postgresql_huge_pages }}

#temp_buffers = 8MB

#max_prepared_transactions = 0
```

```
- template:
  src: "postgresql.conf.j2"
  dest: "${PGDATA}/postgresql.conf"
  owner: postgres
  group: postgres
  rights: u=rw
```

Handlers

```
- name: restart postgresql
  Systemd:
    Name:"{{ postgresql_cluster.daemon }}"
    daemon_reload: true
    state: restarted
    enabled: true
    become: true
```

Service postgres (Rhel7)

Fonctions liées à a sécurité

- Encryption:
  - **Ansible Vault**: AES
  - Encrypter un fichier
  - Encrypter une chaîne de caractères

- Sécurisation des accès:
  - Compte technique: Postgres
  - Comptes admin nominatifs (console d’administration)

Ansible Tower (enterprise)

- Console graphique
- **API rest**
- Gestion des droits avec des groupes
- Centralisation des logs
- Ansible Tower Version 3.4.2; March 06, 2019

- J’ai terminé: Merci pour votre attention
  Questions?

h e p i a

Haute école du paysage, d'ingénierie
et d'architecture de Genève

PostgreSQL Meetup 23 Mai 2019

Industrialisation et Haute Disponibilité

PostgreSQL User Group Genève

h e p i a
Haute école du paysage, d'ingénierie
et d'architecture de Genève

dbi services

Dr Paul Albuquerque
Responsable de la filière Informatique à l'école HEPIA

OpenDB Appliance - **Pierre Sicot** – Dbi-Services

TeamBoard - **Pierre Giraud** - Dalibo

La réplication encore plus facile - **Pierre Boizot** - freelance

Déploiement automatisé d'un cluster PostgreSQL - **Fabrice Chapuis** - Banque Lombard Odier

PostgreSQL

(liberal Open Source license, similar to the BSD or MIT licenses)

https://www.postgresql.org/about/licence/

- Alternative aux logiciels propriétaires (maturité du produit)

- Communauté forte (nombreux contributeurs)

- Code ouvert

- Support (plusieurs sociétés d’experts)

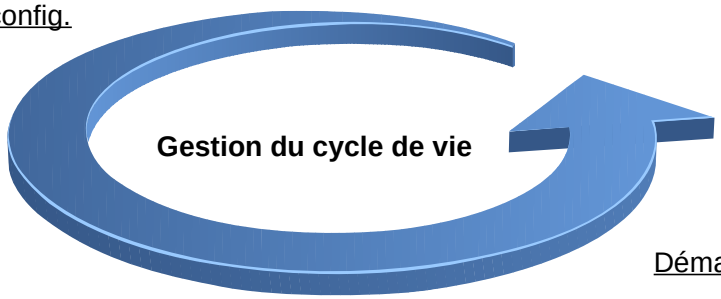
- Intégration avec d’autres logiciels libres

Industrialisation

- Taille de l’infrastructure
- Gestion des tâches opérationnelles
- Demandes de nouvelles installations
- Standardisation
- **DbaaS**

Déploiement

Mise à jour du schema
Copie
Modification de la config.



Démantèlement

Upgrade du SGBD
Patch

Migration

Outil de déploiement

quelle valeur ajoutée?

- **Modules** versus SHELL Scripts
- Langage déclaratif
- Eviter les erreurs (fiabilisation du déploiement)
- Consistance sur l’ensemble des serveurs
- Gain de temps
- **Idempotence**

SQL – idempotent?

Update, Delete, Insert

```
postgres=# CREATE TABLE sometable (a int primary key, b int, c int);
CREATE TABLE
postgres=# INSERT INTO sometable (a,b,c) values (1,3,2) ON CONFLICT (a)  DO NOTHING;
INSERT 0 1
postgres=# INSERT INTO sometable (a,b,c) values (1,3,2) ON CONFLICT (a)  DO NOTHING;
INSERT 0 0
postgres=# INSERT INTO sometable (a,b,c) values (1,2,3) ON CONFLICT (a)  DO NOTHING;
INSERT 0 0
postgres=#
```

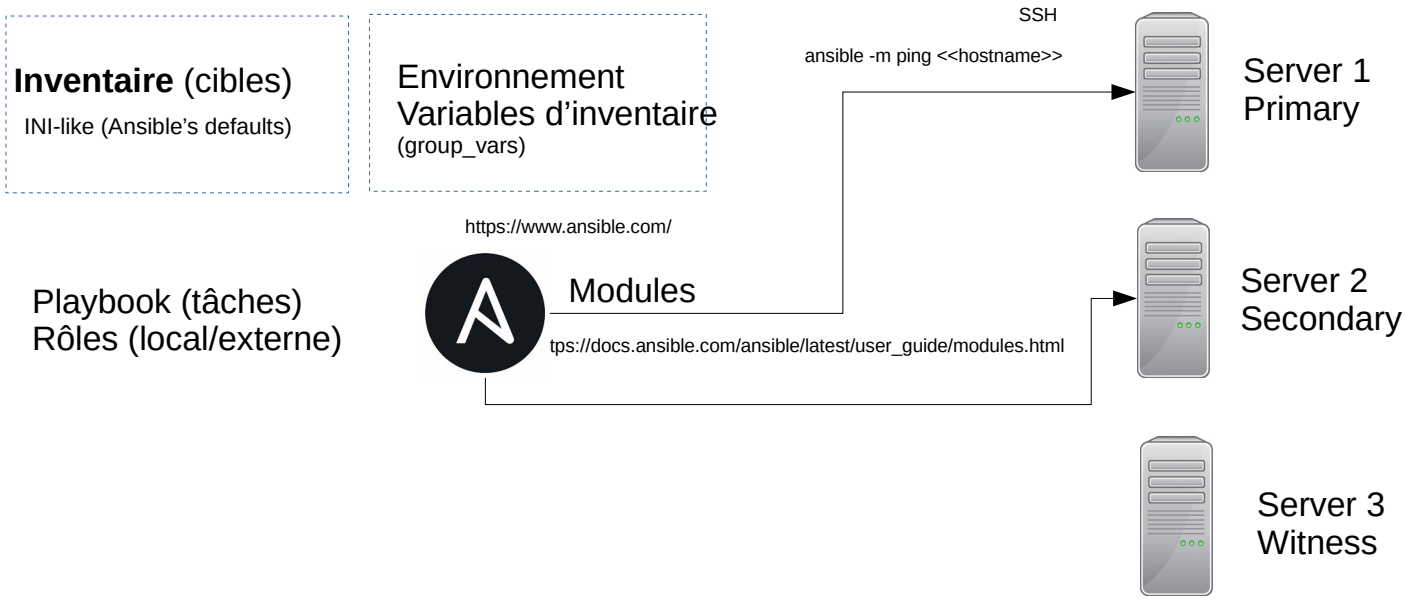
	Companie	Method	Language	Licence	Written	Last release
	Puppet (Luke Kanies in 2005)	Pull	Declaratif	Apache from 2.7.0, GPL before	C++, Clojure and Ruby	V.6.4
	Chef	Pull	Decl / Imp	Apache License	Ruby (client) and Ruby / Erlang (server)	V.12.19
	Redhat Inc.	Push	Decl / Imp	GNU General Public License	Python	2.7*

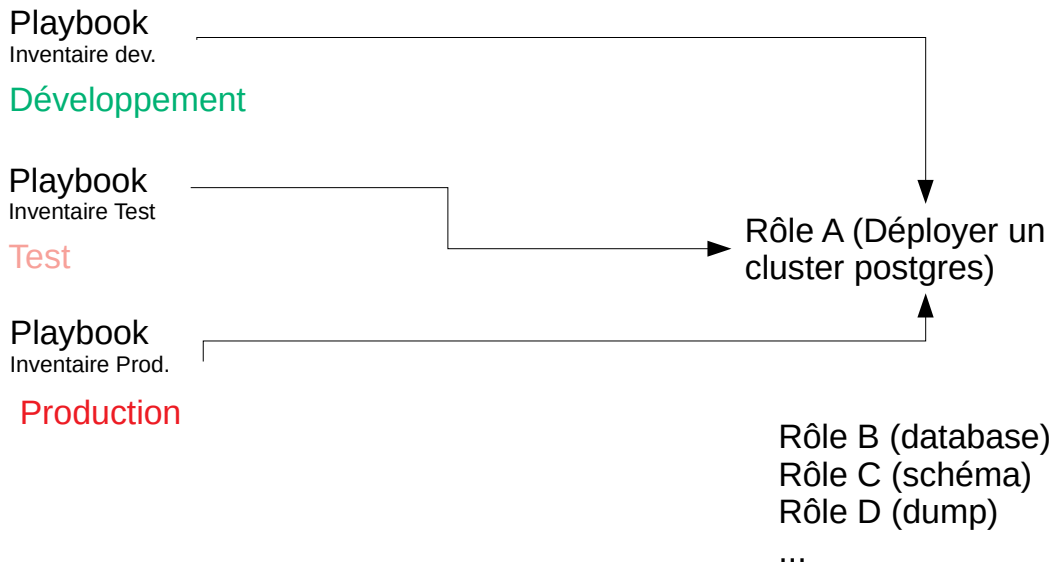
* Python 2 (version 2.6 ou ultérieure) ou Python 3 (version 3.5 ou ultérieure).
V. 2.8 16.5.2019

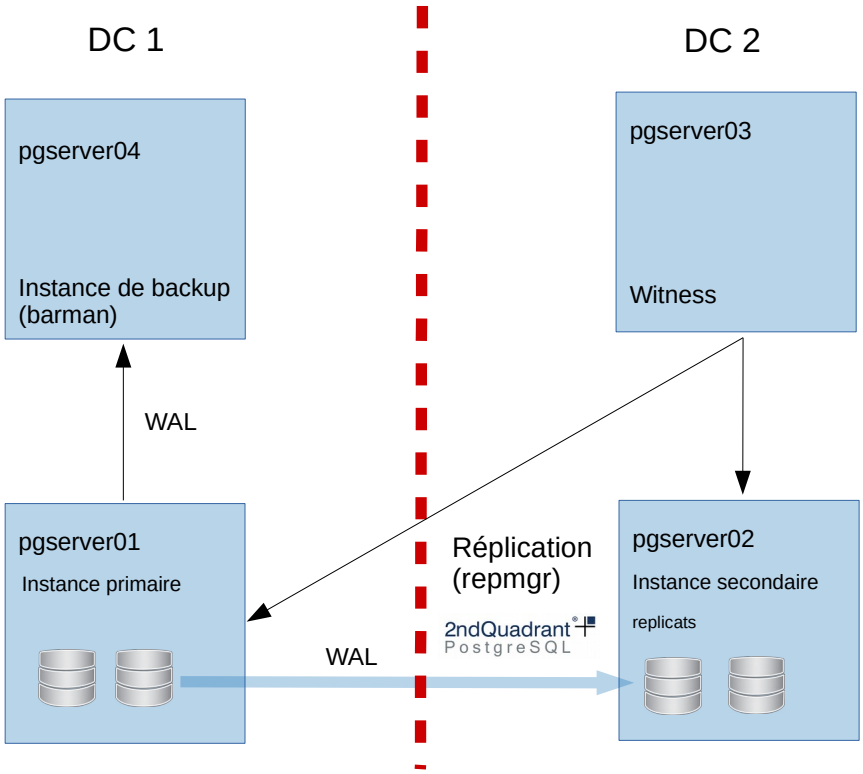
https://en.wikipedia.org/wiki/Comparison_of_open-source_configuration_management_software

---
YAML:
YAML Ain’t Markup Language:
- Playbook:
- Indentation avec des espaces: comme Python
- Format compact: [‘facilement lisible’, ‘auto-documenté’]
- Superset du JSON: # Commentaires
- Types: [‘mappings (hashes / dictionaries)’,’sequences (arrays / lists)’, ‘scalars (strings / numbers)’]

...


Inventaire (cibles)
INI-like (Ansible's defaults)
Environnement
Variables d'inventaire
(group_vars)
SSH
ansible -m ping <<hostname>>
Server 1
Primary
https://www.ansible.com/
Playbook (tâches)
Rôles (local/externe)
Modules
tps://docs.ansible.com/ansible/latest/user_guide/modules.html
Server 2
Secondary
Server 3
Witness

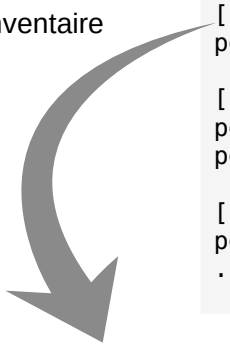

Playbook
Inventaire dev.
Développement
Playbook
Inventaire Test
Test
Playbook
Inventaire Prod.
Production
Rôle A (Déployer un cluster postgres)
Rôle B (database)
Rôle C (schéma)
Rôle D (dump)
...



Structure: ($PGDATA,log)
Configuration
(postgresql.conf,
pg_hba.conf)

Backup
(pg_basebackup,
pg_dump)
Purge des archives
SSL
Logrotate

Inventaire



```
# PostgreSQL targets
[postgresql:children]
pg-cluster

[pg-cluster]
pgserver01  # Primary
pgserver02  # Secondary

[primary-pg-cluster]
pgserver01
...
```

Variables d’inventaire

```
postgresql:
  cluster_name: 'somecluster'
  cluster_port: 5432
  shared_buffers: 512MB
  huge_pages: try
  work_mem: 8MB
  max_connections: 100
  archive_mode: 'on'
  archive_command: 'cp %p archives/%f'
  archive_timeout_in_seconds: 3600
  database_name: 'somedatabase'
  database_user: 'bob'
  ...
```

Playbook Ansible
(cluster)

```
---
- name: PostgreSQL - deploy cluster
  hosts: pg-cluster
  gather_facts: no

  vars:
  ...
  roles:
    - role: deploy-cluster
    vars:
      postgresql_cluster_name: "{{ postgresql.cluster_name }}"
      postgresql_cluster_port: "{{ postgresql.cluster_port }}"
      postgresql_shared_buffers: "{{ postgresql.shared_buffers }}"
...
```

Playbook Ansible
(database)

```
---
- name: PostgreSQL - deploy database
  hosts: pg-cluster:&primary-pg-cluster
  gather_facts: no

  vars:
  roles:
    - role: deploy-database
      vars:
        postgresql_cluster_name:"{{ postgresql.cluster_name }}"
        postgresql_cluster_port: "{{ postgresql.cluster_port }}"
        postgresql_database_name: "{{ postgresql.database_name }}"
        postgresql_user_name: "{{ postgresql.database_user }}"
...
```

Rôle deploy-cluster (fichier main.yml)

```
- import_tasks: preflight.yml
  tags:
    - preflight

 - import_tasks: install.yml
   tags:
     - install

- import_tasks: configure.yml
  tags:
    - configure

- import_tasks: postflight_primary.yml
  tags:
    - postflight_primary

- import_tasks: postflight_secondary.yml
  tags:
    - postflight_secondary
```

check input parameters

Install packages

create structure / install fichiers de config.

Tâches spécifiques pour l'instance primaire: initdb + check

Tâches spécifiques pour l'instance secondaire: clone db + check

Rôle deploy-cluster (primaire)

```
- name: "{{ postgresql_cluster.name }} - initialize cluster"
  shell: "{{ postgresql_paths.program.path }}/bin/initdb -D {{ postgresql_paths.cluster.path }}
--data-checksums --locale={{ postgresql_locale }} --encoding={{ postgresql_encoding }}"
  args:
    creates: "{{ postgresql_paths.cluster.path }}/PG_VERSION"
  become: true
  become_user: postgres
  register: out_init_cluster
  when:
   - primary
```

primary: "{{ inventory_hostname in groups['postgresql-primary'] }}" (preflight)

Rôle deploy-cluster (standby)

```
- name: "{{ postgresql_cluster.name }} - clone primary database"
  shell: /usr/pgsql-10/bin/repmgr -h "{{ master_node }}" -p "{{ postgresql_cluster_port }}" -U
     repmgr -d repmgr -f /etc/{{ postgresql_cluster_name }}/repmgr.conf -F standby clone
  args:
    creates: "{{ postgresql_paths.cluster.path }}/PG_VERSION"
  become: true
  become_user: postgres
  become_method: sudo
  register: out_clone_primary
  when:
     - not primary
```

Rôle deploy-database (main.yml)

```
---
  import_tasks: preflight.yml
  tags:
    - preflight

- import_tasks: deploy-user.yml
  tags:
    - deploy-user

- import_tasks: deploy-database.yml
  tags:
    - deploy-database

-  import_tasks: postflight.yml
   tags:
    - postflight
```

check input parameters

Création du db owner

Création de la db + grants

Contrôle que les tâches se sont bien exécutées (test des accès à la base)

Rôle deploy-database

```
- name: "{{ postgresql_cluster.name }} - create dbuser"
  postgresql_user:
    login_user: postgres
    port: "{{ postgresql_cluster_port }}"
    name: "{{ postgresql_database_user }}"
    password:
    role_attr_flags:NOSUPERUSER,NOCREATEDB,NOCREATEROLE
    state: present
    become: true
    become_user: postgres
    become_method: sudo
```

Modules Ansible

```
- name: "{{ postgresql_database_name }} - create database"
  postgresql_db:
    login_user: postgres
    name: "{{ postgresql_database_name }}"
    port: "{{ postgresql_cluster_port }}"
    owner: "{{ postgresql_database_user }}”
    state: present
    become: true
    become_user: postgres
```

Moteur de templating http://jinja.pocoo.org/docs/2.10/

jinja2 format (.j2)
Postgresql.conf.j2

Variabilisé (gérer plusieurs environnements)

```
#------------------------------------------------------------------
# RESOURCE USAGE (except WAL)
#------------------------------------------------------------------
# - Memory -
shared_buffers = {{ postgresql_shared_buffers }}

huge_pages = {{ postgresql_huge_pages }}

#temp_buffers = 8MB

#max_prepared_transactions = 0
```

```
- template:
  src: "postgresql.conf.j2"
  dest: "${PGDATA}/postgresql.conf"
  owner: postgres
  group: postgres
  rights: u=rw
```

Handlers

```
- name: restart postgresql
  Systemd:
    Name:"{{ postgresql_cluster.daemon }}"
    daemon_reload: true
    state: restarted
    enabled: true
    become: true
```

Service postgres (Rhel7)

Fonctions liées à a sécurité

- Encryption:
  - **Ansible Vault**: AES
  - Encrypter un fichier
  - Encrypter une chaîne de caractères

- Sécurisation des accès:
  - Compte technique: Postgres
  - Comptes admin nominatifs (console d’administration)

Ansible Tower (enterprise)

- Console graphique
- **API rest**
- Gestion des droits avec des groupes
- Centralisation des logs
- Ansible Tower Version 3.4.2; March 06, 2019

- J’ai terminé: Merci pour votre attention
  Questions?