J'ai fait danser un troupeau d'éléphants

Patrick Francelle et Stéphane Schildknecht

Introduction

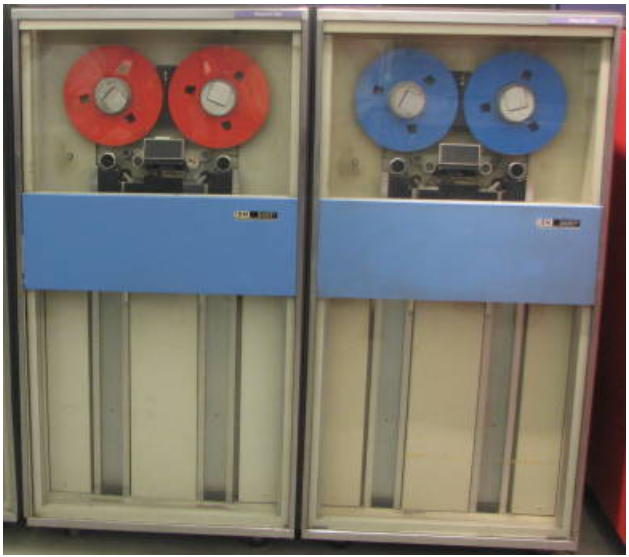
Mise en place de **PostgreSQL** et de **Slony**

- En remplacement d'un serveur IBM Zseries et d'une base VSAM
- Dans une entreprise de presse spécialisée

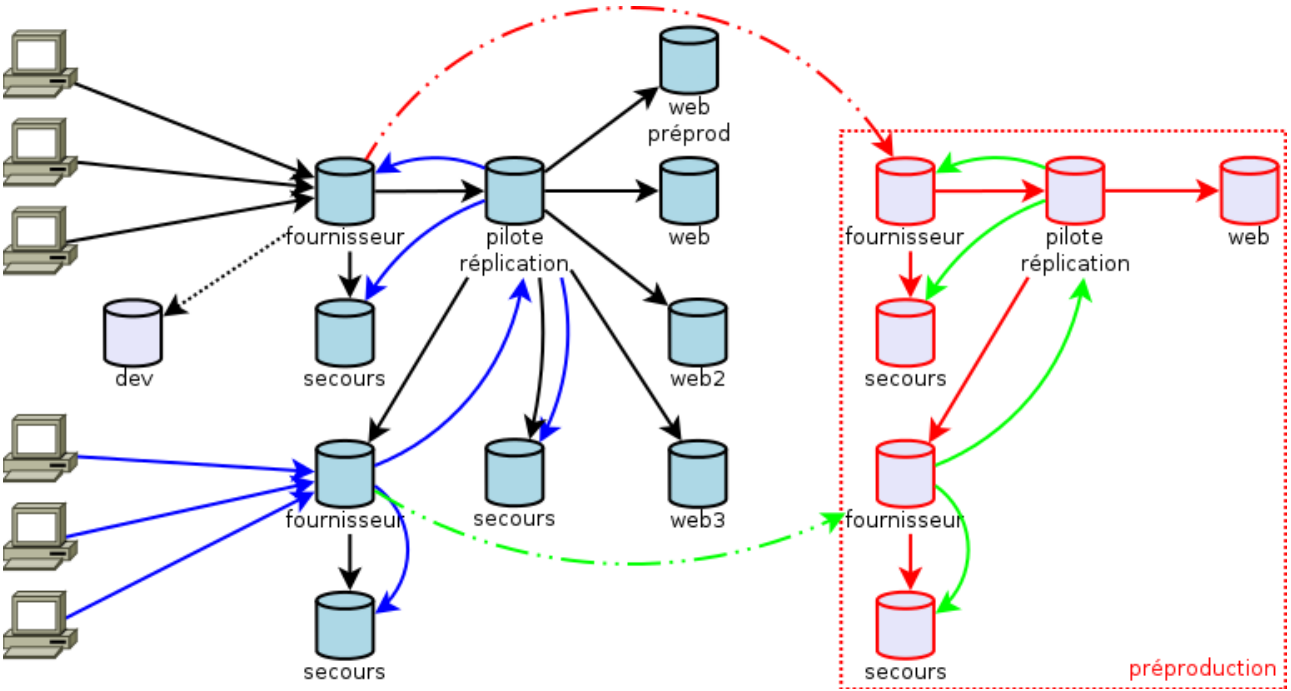
Évolution chronologique du SI

- Récit *crescendo*
- De 2007 à 2014

À l'origine



Aujourd'hui


web
préprod
fournisseur
pilote
réplication
web
dev
secours
web2
fournisseur
secours
web3
secours
fournisseur
pilote
réplication
web
secours
fournisseur
secours
préproduction

Migration ?

- Extraction des données de l'IBM
- Intégration dans PostgreSQL
- Fin de l'histoire

En résumé

Migration !

- La réalité est plus complexe
- Prise en compte
  - Des différentes applications
  - Des flux externes (entrants et sortants)
  - Des opérateurs de saisie
- Réécriture des clients lourds
- Intégration
  - Du travail des journalistes
  - De nouveaux flux
- Anticipation des évolutions

Au final, la migration s'est déroulée sur plus de 2 ans.

Point de départ

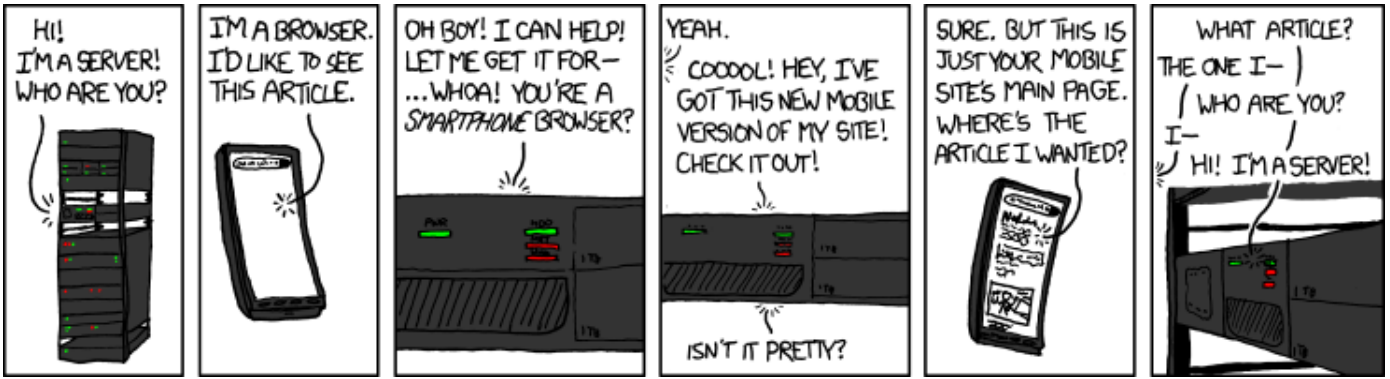
- Serveur IBM mainframe
  - Base *VSAM*
  - Consultation par applications en assembleur IBM
- Presse spécialisée
  - Applications métier non modifiables
  - Outils internes spécifiques
- Site web statique (1 MAJ par jour)

Objectifs

1. Mettre en place un site web dynamique
    - Mise à jour d'informations tout au long de la journée
    - Partie gratuite
    - Partie payante avec abonnement
2. Remplacer le serveur IBM et sa base de données
    - Performances insuffisantes
    - Évolutions limitées
    - Coûts de maintenance prohibitifs

Premier objectif

« Mettre en place un site web dynamique »



Contraintes métier

- Impossibilité de faire une bascule complète à court terme
  - Clients à réécrire
  - Données à migrer
- Mises à jour continuelle de la base IBM
  - Flux entrants
  - Opérateurs

Contraintes techniques

- Complexité du métier
  - Abstraction nécessaire pour développement externe
- Hébergement externe du site web
- Haute-disponibilité requise

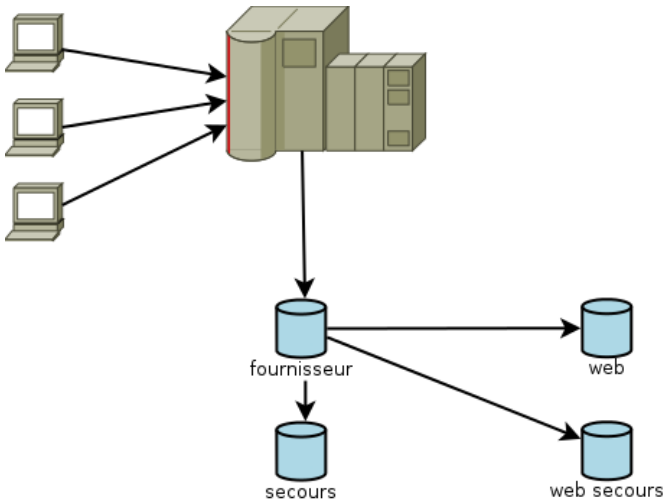
Solutions

- Pas d'abandon immédiat de la base IBM
- Ruissellement des données
  - Extractions régulières IBM
  - Insertions dans la base de données
- Développement de procédures stockées
- Réplication
- Choix de PostgreSQL

Pourquoi PostgreSQL ?

- Comparaison de PostgreSQL, Oracle, Sybase et Mysql
- PostgreSQL vainqueur
  - Fiabilité des données
  - Performances
  - Open source
  - Disponibilité des pilotes et API
  - Procédures stockées
  - Coût de possession faible

Architecture retenue



Réplication

- Nécessaire car :
  - Écritures en local
  - Lectures distantes
  - Copie de secours locale
  - Copie de secours distante

Slony

- Outil externe à PostgreSQL
- Réplication asynchrone et asymétrique
- Tables et séquences regroupées en *sets*
- Un fournisseur par *set*
- *Triggers* sur les tables
- Possibilité de réplication en cascade

Slony1-ctl

- Slony
  - Langage interne
  - Ordre précis des scripts et opérations
  - Complexité exponentielle des scripts
- Slony1-ctl
  - Scripts shell (bash)
  - Génération automatisée des scripts slony
  - Autonomie des adminsys

Slony vs Slony1-ctl

```
"/usr/bin/slonik" << SEOF
include </usr/local/slony1-ctl/etc/slclust.preamble>
try{
  init cluster (ID=1, COMMENT = 'slclust : 1 - mabase - pg');
}
on error {
  echo 'Cluster 1 : Failed!!!'; exit 1;
}
on success {
  echo 'Cluster 1 : Success';
}
try{
  store node (ID = 2, EVENT NODE = 1, comment = 'DB mabase - Host backup - Cluster slclust');
}
on error {
  echo 'Store node 2 : Failed !!!'; exit 1;
}
on success {
  echo 'Store node 2 : Success';
}
  store path (SERVER = 1, CLIENT = 2, CONNINFO='dbname=mabase host=pg port=5432 user=slony');
  store path (SERVER = 2, CLIENT = 1, CONNINFO='dbname=mabase host=backup port=5433 user=slony');

try{
  create set (ID=1, ORIGIN=1, COMMENT='Tables in mabase');
  # Tables
  set add table (SET ID=1, ORIGIN=1, ID=1, FULLY QUALIFIED NAME = 'public.group', COMMENT='table public.group');
  set add table (SET ID=1, ORIGIN=1, ID=2, FULLY QUALIFIED NAME = 'public.group_permissions', COMMENT='table public
.group_permissions');
### ...
# Sequences
  set add sequence (SET ID=1, ORIGIN=1, ID=1, FULLY QUALIFIED NAME = 'public.group_id_seq', COMMENT='sequence publi
c.group_id_seq');
  set add sequence (SET ID=1, ORIGIN=1, ID=2, FULLY QUALIFIED NAME = 'public.group_permissions_id_seq', COMMENT='se
quence public.group_permissions_id_seq');
### ...
}
on error {
  echo 'Creation Set : Failed !!!'; exit 1;
}
on success {
  echo 'Creation Set : Success';
}
SEOF
if [ "$?" -ne 0 ]; then
  echo "Error slonik while initialisation"; exit 1
fi
exit 0

"/usr/bin/slon" -f "/usr/local/slony1-ctl/etc/slon.cfg" "slclust" "dbname=mabase host=pg port=5432 user=slony" >"/u
sr/local/slony1-ctl/logs/slclust_mabase_1.log" 2>&1 &
"/usr/bin/slon" -f "/usr/local/slony1-ctl/etc/slon.cfg" "slclust" "dbname=mabase host=backup port=5433 user=slony"
>"/usr/local/slony1-ctl/logs/slclust_mabase_2.log" 2>&1 &

"/usr/bin/slonik" << SEOF
include </usr/local/slony1-ctl/etc/slclust.preamble>
try{
  subscribe set ( ID = 1, PROVIDER = 1, RECEIVER = 2, FORWARD = YES );
}
on error {
  echo 'Subscription Failed !!!'; exit 1;
}
on success {
```

```
postgres@barman91:~$ ./01_create_init.sh -c slclust && ./02_exec_init.sh -c slclust
```

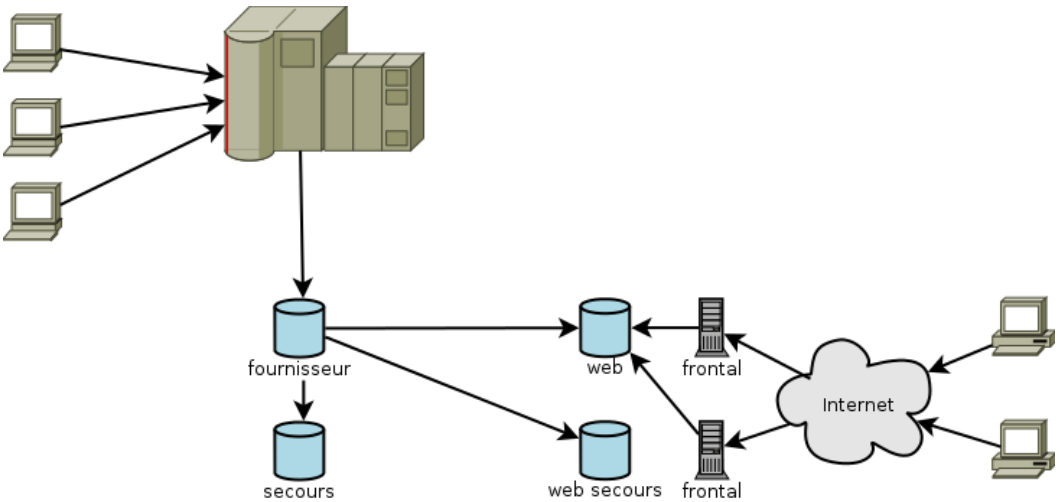
Difficultés rencontrées

- Nouveauté de la solution
  - Modèle relationnel
  - SQL
  - Procédures stockées
  - Réplication
- Complexité apparente de l'architecture
- Travail collaboratif
- Résistance au changement

Solutions

- « Toute résistance est inutile »
- *MCD*
- Formations
- Assistance
- Mise en place d'outils
    - d'administration
    - de versionnage
    - documentaires

Premier objectif atteint



Second objectif

« Abandonner le serveur IBM »

Pré-requis

- Migration des applications clientes
  => Environnement de développement
- Pas d'impact sur les évolutions parallèles du SI
  => Anticipation des changements

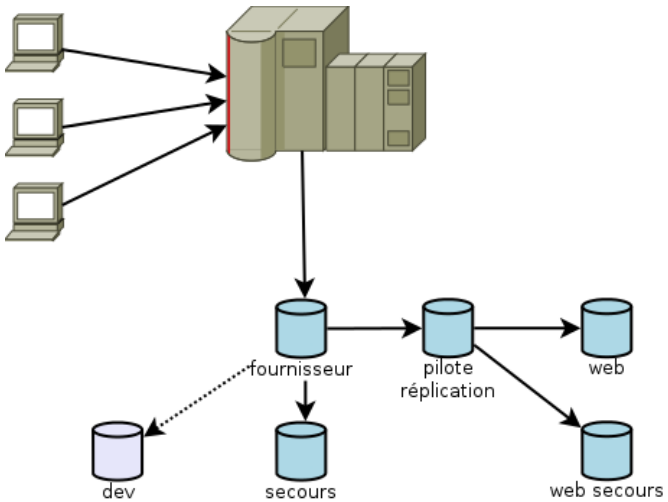
Environnement de développement

- Développement collaboratif
- Référentiel versionné
- Outil de publication en (pré)production
- Serveur de base de données de test
    - Dump quotidien des bases de production
    - Restauration des procédures stockées de dev

Anticipation des changements

- Prévision de projets parallèles
  - Nouveaux nœuds de réplication ?
  - Charge sur le fournisseur principal
- Mise en place d'un pilote de réplication
  - Gestion centralisée
  - Facilité d'intégration de nouveaux nœuds
  - Sous contrôle des adminsys

Anticipation des changements

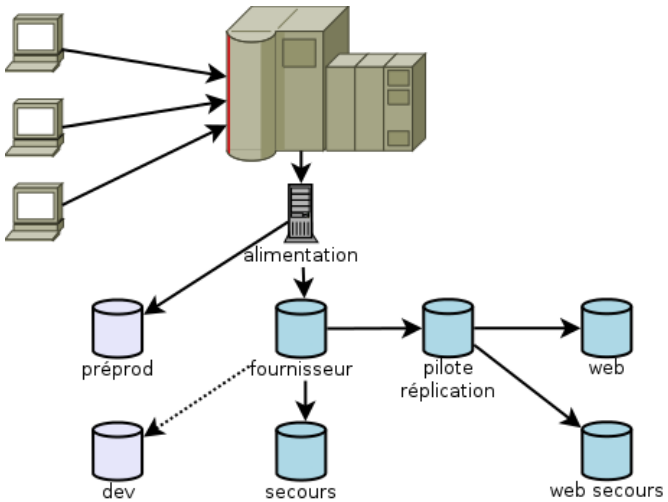


Préparation de la bascule

- Déport des fonctions d'alimentation
- Alimentation possible de plusieurs fournisseurs principaux
    - Création d'une pré-production temps réel
- Possibilité de supprimer l'IBM !

  ...dès que les Devs seront prêts

Préparation de la bascule

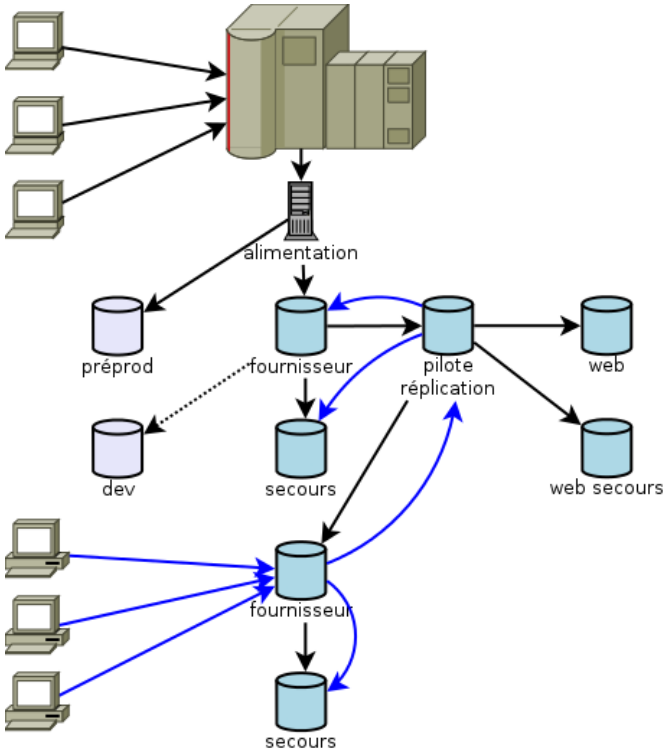


Une réception inattendue

Évolution de l'entreprise

- Intégration d'un second site
  - Données spécifiques à ce site
- Réplications avec idée d'un PRA
  - Réplication descendante vers le second site
  - Réplication montante vers le premier site
- Impacts importants sur slony
  - Réplications croisées

Intégration d'un second site



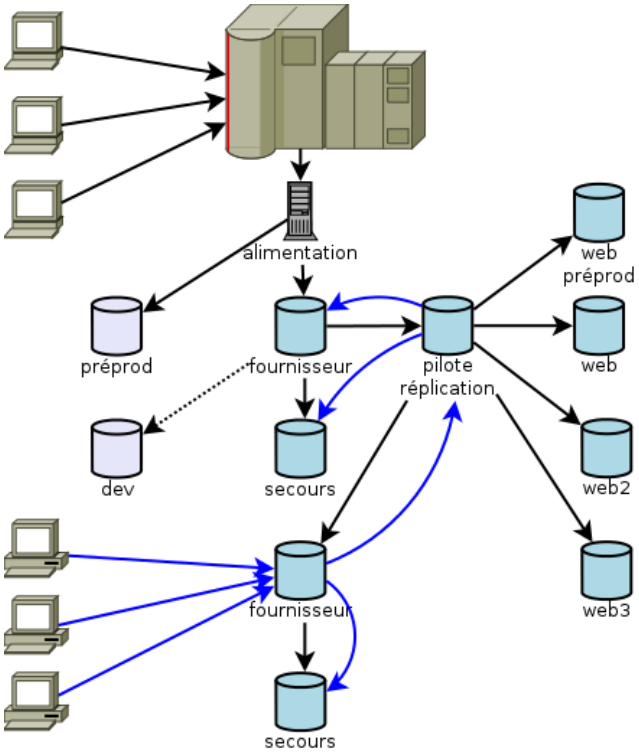
Difficultés rencontrées

- Faiblesses réseau
  - Lenteurs d'initialisation
  - Difficultés de réplication
  - Déconnexions
- Méconnaissance de Slony par les nouvelles équipes
  - Modifications de schémas « sauvages »
  - Actualisations massives de données

Solutions apportées

- Réseau
  - Mobilisation des équipes réseau
  - Changement de fournisseur d'accès
  - Outil de relance automatique des connexions
- Connaissance Slony
  - Formation des équipes
  - Mise en place de bonnes pratiques

Autres ajouts

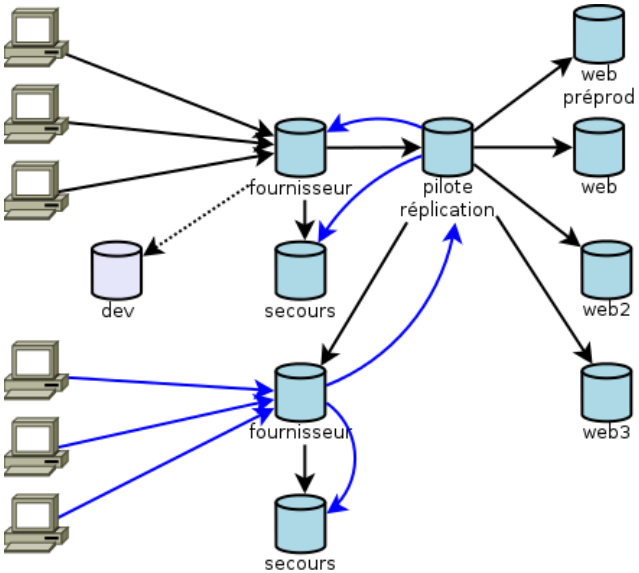


Second objectif (suite)

Se débarrasser de la base IBM

- Changement des applis internes spécifiques
- Développement d'interfaces de saisie
- Développement de procédures stockées
- Formation des équipes
- Bascule « big bang » périlleuse
    - Retour en arrière ?
- Perte de la pré-production slony

Se débarrasser de la base IBM



Mission accomplie !

Mission accomplie ?

Oui !

- Les 2 objectifs sont atteints

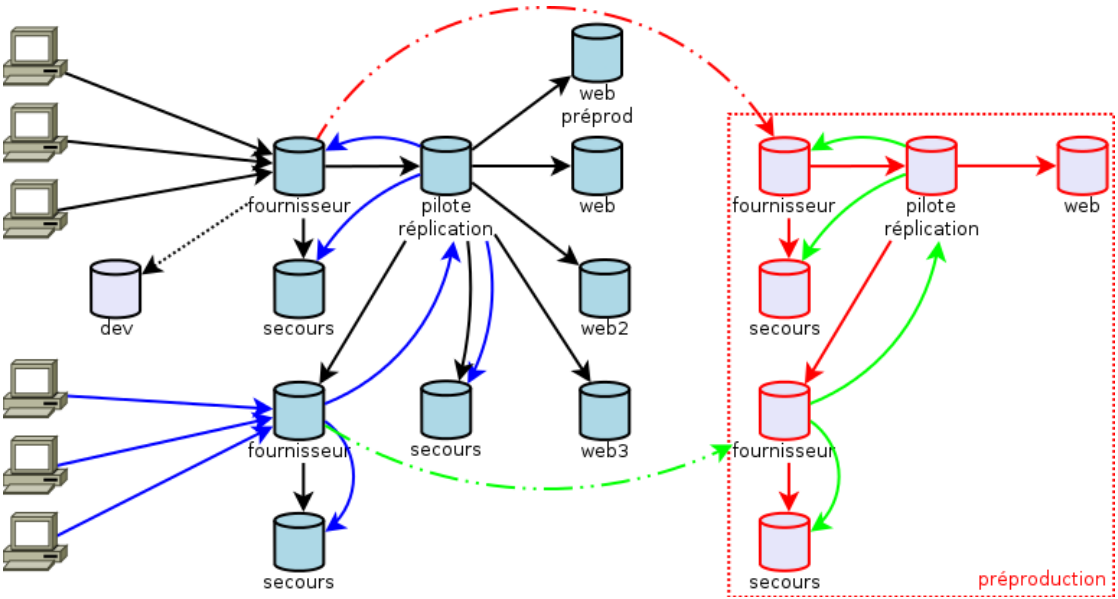
Mais

- Mises à jour nécessaires

Tester l'architecture

- Création d'une nouvelle pré-production slony
- Enregistrement de toutes les requêtes
    - Ne faites pas ça chez vous !
- Filtres, reconstitution et réexécution des requêtes
- Possibilité de tester toutes les requêtes

Tester l'architecture

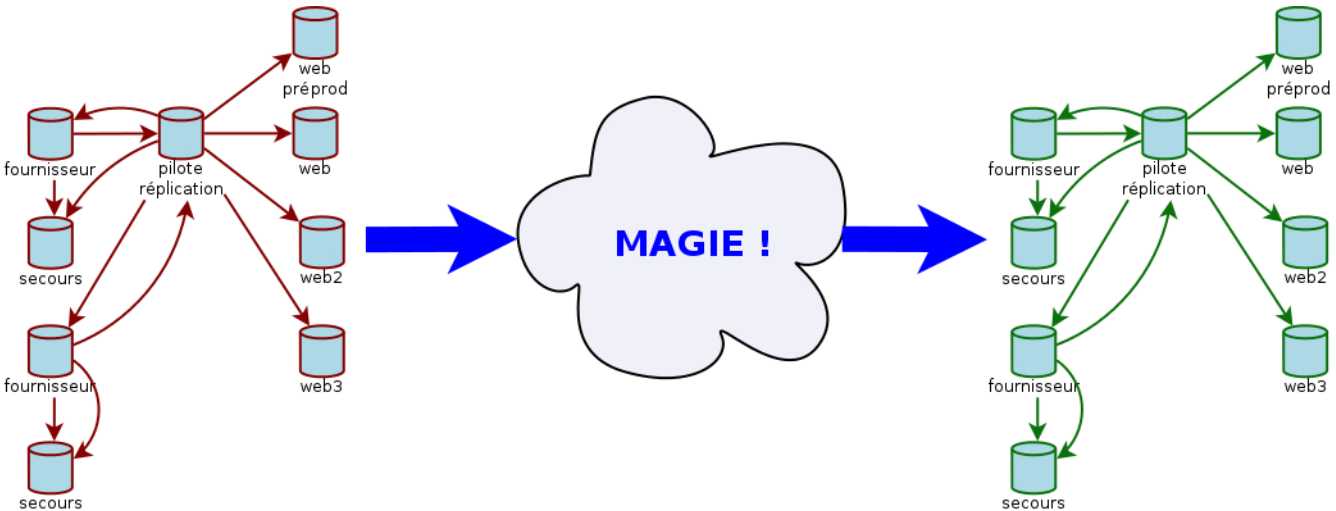


La dernière étape

Objectifs

- Migration PostgreSQL 8.2 vers 9.2
- Migration Slony 1.2 vers 2.1
- Mise à jour du système

Déroulement ?



Contraintes

- Slony 1.2 : jusqu'à PostgreSQL 9.0
- Slony 2.1 : à partir de PostgreSQL 8.3

  => Migration en 2 temps
- « Régressions » entre PostgreSQL 8.2 et 8.3

  => Revoir toutes les requêtes et procédures stockées

Déroulement !

1. Instanciation et réplication de nouvelles bases PostgreSQL en version 9.0
2. Suppression des anciennes bases au fur et à mesure des installations
3. Suppression de Slony 1.2
4. Déploiement de Slony 2.1
5. Migration des bases PostgreSQL en version 9.2

Difficultés rencontrées

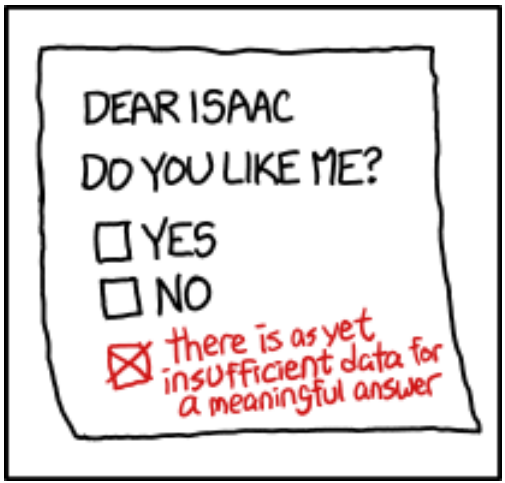
Étonnamment peu

- Incompatibilité des versions anticipée
- Bonne option de Slony 2.x
- Efficacité de *pg_upgrade*
- Robustesse des outils

Conclusion

- Besoin originel simple
- Ajout progressif de fonctionnalités
- Apparition de nouveaux usages avec les outils
- Slony était le bon choix
  - Impossible (à l'époque) de faire la même chose avec un autre outil
- L'architecture continue d'évoluer
  - Montées de version PostgreSQL et Slony
- Du temps gagné si démarrage retardé
  - PostgreSQL 8.3

Questions ?


DEAR ISAAC
DO YOU LIKE ME?
YES
NO
there is as yet insufficient data for a meaningful answer