Présentation du cahier des charges technique

GULDNER - HURIER- LACAVE - SOILIHI

Sommaire

- Conception logicielle
- Architecture système
- Mécanismes de protection
- Politique de sécurité

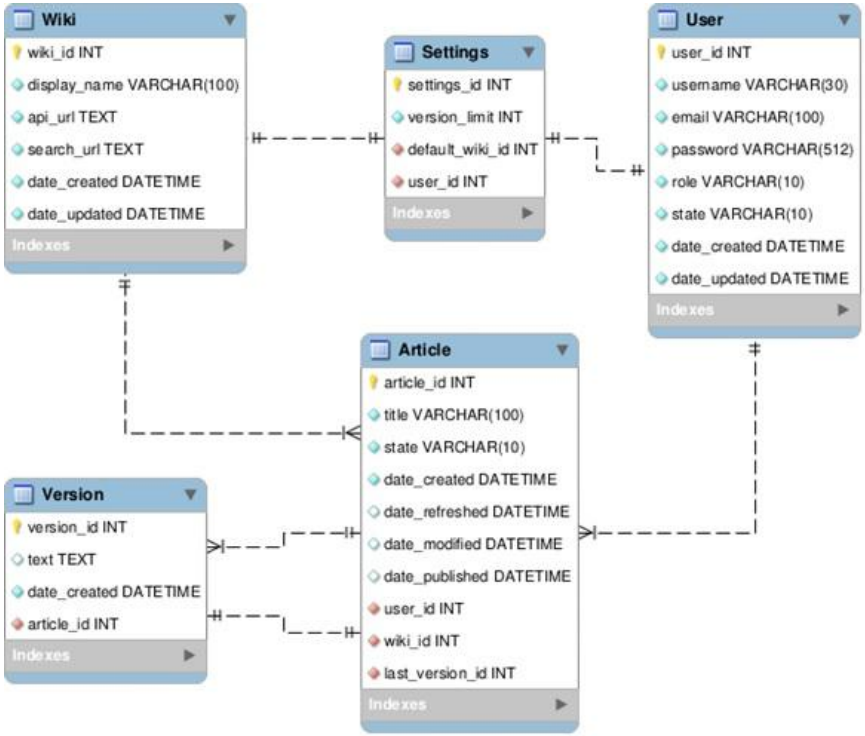
Conception logicielle

Schéma de la BDD

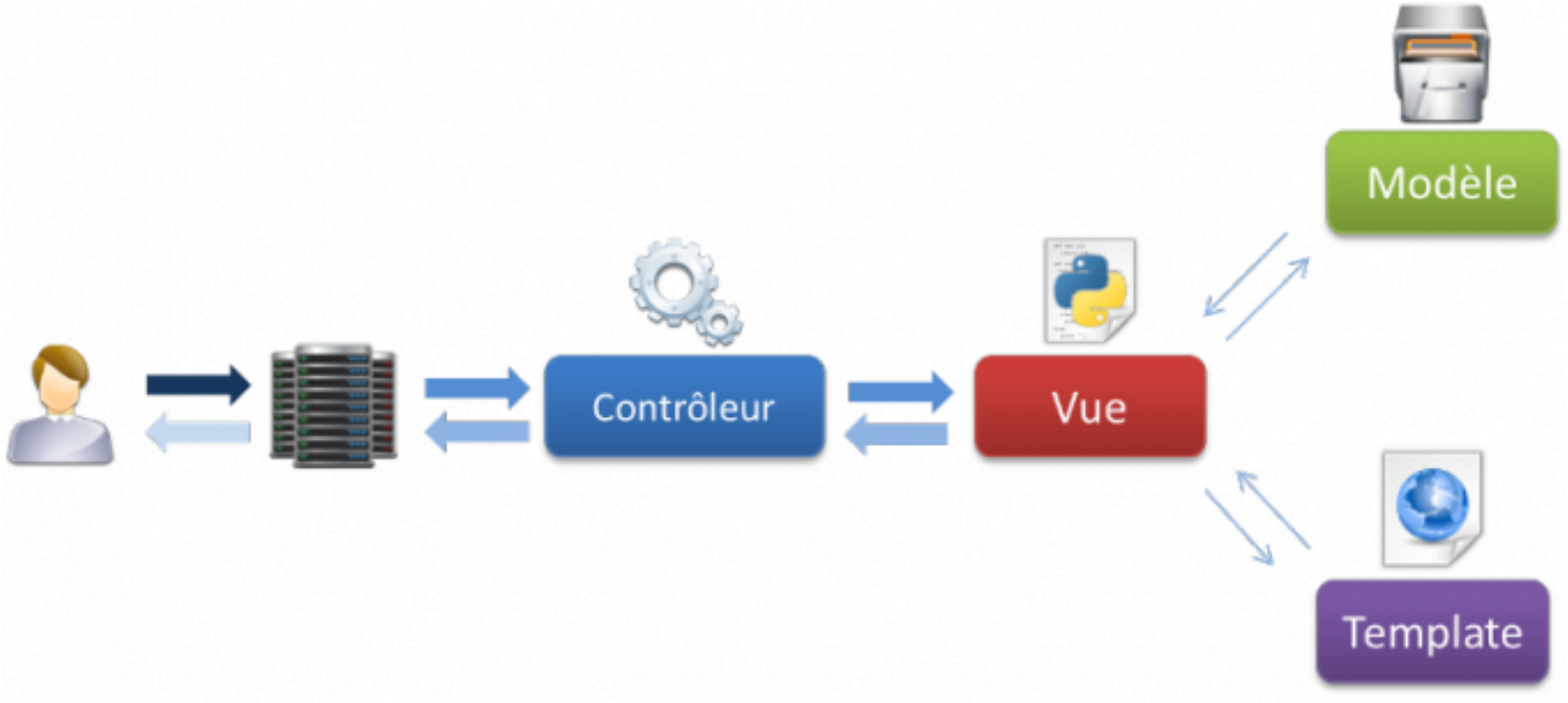
- **information critique: User**
- chiffrement: PBKDF2 + SHA256
  - méthode du key stretching
  - hash(key+password+salt)

Contraintes communes:

- encodage: UTF-8
- moteur: InnoDB
  - gère les clés étrangères
- modifications en cascade



“Design”

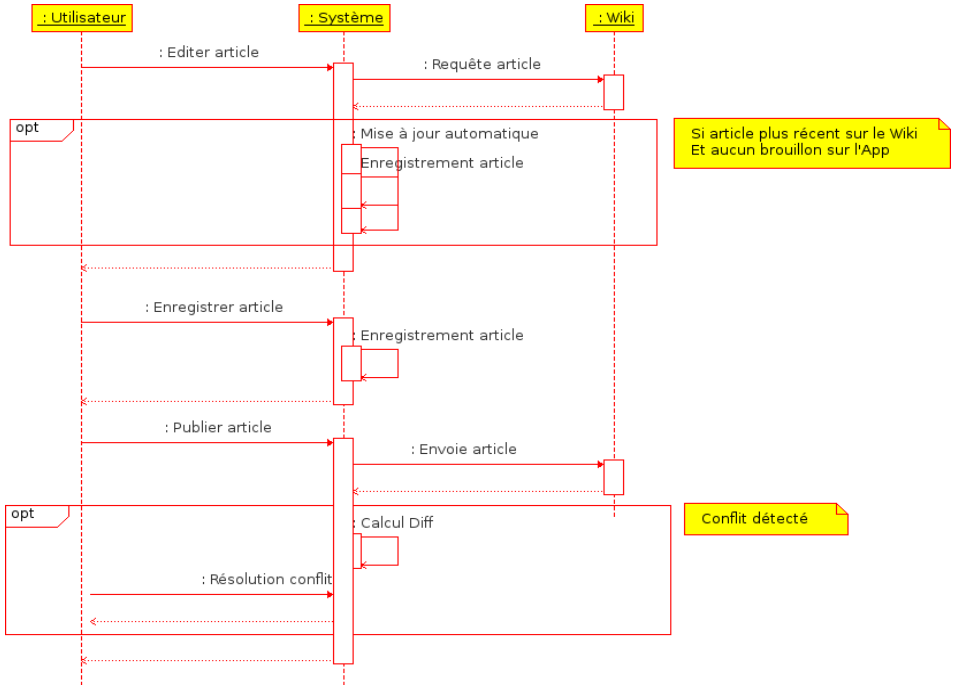


Synchronisation

- nouvelle version !
  - regroupe 4 traitements
  - plus souple et économe

Modifications:

- MAJ automatique
- gestion “lazy” des conflits
- bibliothèque de calcul diff
  - lxml.html.diff



API tiers

Authentification unique

- utilise le principe de SSO
  - single-sign on
- plugin Mediawiki: SocialLogin
  - intégration Facebook et Google
- plugin Django: SocialAuth
  - OAuth1, OAuth2, OpenID

API Wiki

- utilise une API HTTP Rest
  - Representational State Transfer
- Envoie de requête GET HTTP
- Retour au format JSON, WDDX, XML, YAML et PHP

- Bibliothèque XML : ElementTree
- Bibliothèque HTTP: http.client

Architecture système

Environnement sys.

Nom	Catégorie	Description	Arguments
Ubuntu Server	Système d' exploitation	Système Linux basé sur Debian Testing	libre, gratuit, nombreux outils, facile à administrer
OpenSSH	Accès à distance	Shell sécurisé en ligne de commande	authentification par mot de passé et clé asymétrique
MySQL	Serveur de base de données	Base relationnelle en mode client/serveur	populaire, bien documenté, performant à faible volume
Nginx	Serveur Web	Sert du contenu HTTP avec modèle asynchrone	performances très élevées, modulaire, CK10 problem

Environnement dév.

Nom	Catégorie	Description	Arguments
Python	Langage de programmation	Langage haut-niveau, objet et dynamique	Utilisé par de grands groupes (Google, Nasa)
WikiWikiWeb (Django)	Framework web	Outil libre et complet basé sur MVC	Nombreux outils de validation et productivité
Mediawiki	Logiciel de Wiki	Gestion du savoir moteur de Wikipédia	Spécification client
PhpMyAdmin	Administration SQL	Accès depuis le Web écrit en HTML/PHP	Simple, complet, évite d’ ouvrir un port MySQL

Instance applicative

Site en production

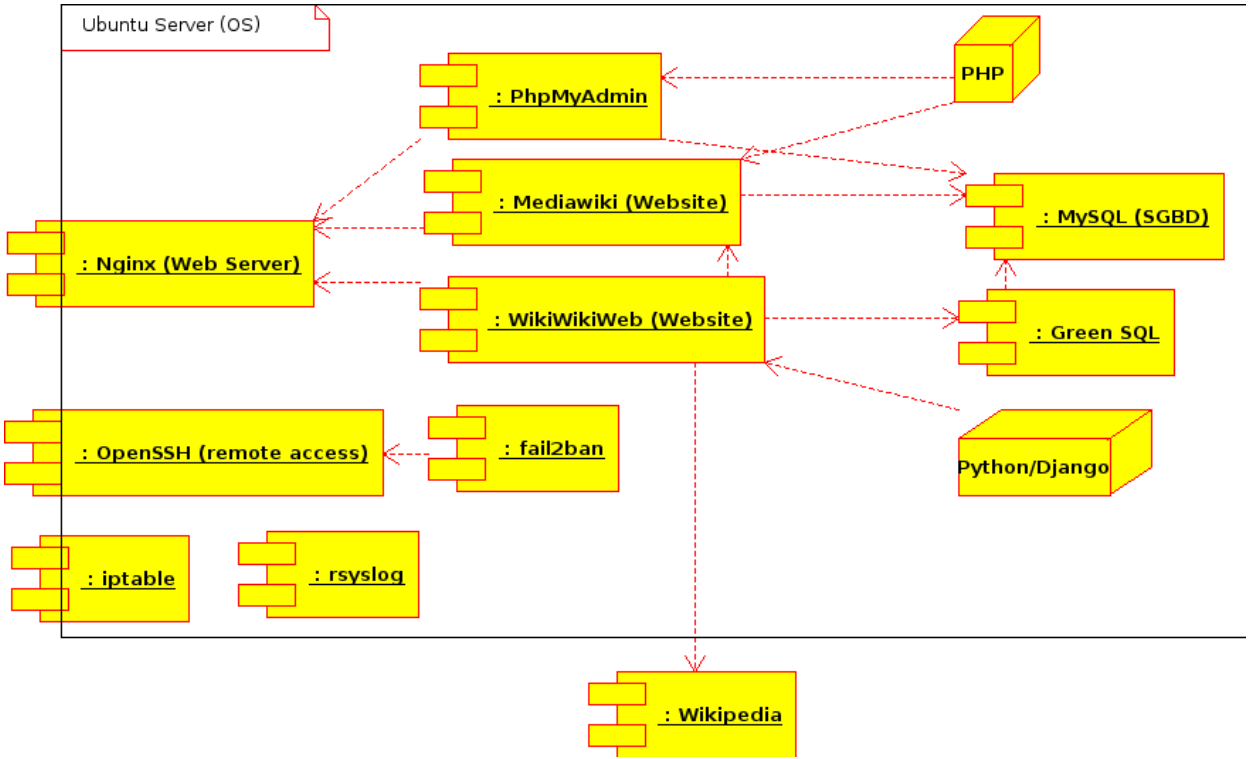
- contraintes importantes!
  - disponibilité, intégrité
  - performance, 24/24 et 7/7
- déployé à partir du 10 déc.

Site en développement

- plus souple
  - MySQL => SQLite
  - Nginx => Django Server
- doit rester confidentiel

1 semaine prévue le passage en production et les tests

Diagramme de déploiement



Mécanismes de protection

Accès à distance

- accès en ligne de commande via SSH2 (OpenSSH)
  - phrase de passe + clé asymétrique = double sécurité !
  - utilise l'algorithme DSA pour l'authentification

- que faire contre les attaques par force brute/dictionnaire ?
  - **fail2ban**: surveille les tentatives de connexion et blocage dynamique
  - fonctionne comme service (daemon)

- comment rendre la vie plus difficile pour nos attaquants?
  - **brouillage de port:** utiliser des ports non standards pour les services
  - PS: la sécurité par l'obscurité a ses limites

Règles de filtrage

Iptables

- Configuration du pare-feu
  - gestion des règles
  - gestion des chaînes
- Règles strictes: deny => allow
- protection contre les attaques prévisibles (NUL, XMAS, SYN)

Snort

- IPS: Intrusion Prevention System
  - bloque les scans
  - bloque les prises d’empreinte
  - bloque les buffer overflows

“Greatest open source software of all time” - InfoWorld's Open Source Hall of Fame 2009

Validations

Django Validation

- validation des formulaires
- validation des modèles
- validation des templates
- validateurs personnalisés

Avantage d’utiliser un framework

très bonne documentation

Green SQL

- Injection SQL: ennemi web n°1 !
  - A1 sur TOP 10 OWASP
- pare-feu SQL pour filtrer les tentatives d’injection

Configurations

A5: Security Misconfiguration (regroupement récent)

A10: Using Components with Known Vulnerabilities

Pour s’en prémunir:

- suivre les recommandations:
  - Django, Mediawiki, Nginx, PHP …
- utiliser des mécanismes de cloisonnement
  - utilisateurs, permissions d’accès, pas virtualisation

Politique de sécurité

Sauvegardes

- Technologie: Script RSync + Cron
- Permissions d’accès via ACL
  - principle of least privilege
- Sauvegarde complète et hebdomaire
  - format zip avec mot de passe

Plan de reprise

En cas de sinistre:

- accès distant
- base de données
- applications web
- système de log
- système de sécurité

- faire attention:
  - corruption des données
  - vol de certificat
  - backdoors

regarder les logs

et comprendre

Conclusion

Bilan

“La sécurité d'un système d'informations est égale à celle de son maillon le plus faible”

Nous n’avons rien négligé !

Notre valeur ajoutée

- doublement des sécurités
  - ex: Django + Green SQL, SSH2, Iptables+Snort
- s’intéresser aux recommandations
- un objectif et une méthode professionnel
  - mais sans les moyens :)

Faille de sécurité !

- Register globals
  - configuration du php.ini
- déclaration des variables REQUEST en global
- compétence en retro-ingénierie et web

Merci pour votre attention

Questions ?