



# Bibliothèque PHP pour sécuriser l'utilisation de Memcached

## SecureMemcachedLibrary

Pierre BLONDEAU

Département Informatique UCBN,  
Laboratoire GREYC, CRNS UMR 6072, UCBN, ENSICAEN  
[pierre.blondeau@unicaen.fr](mailto:pierre.blondeau@unicaen.fr)

12 Décembre 2013





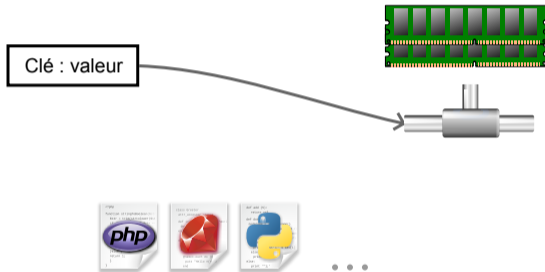
- 1 Memcached
  - Présentation
  - Contexte d'utilisation
  - Problème
- 2 SecureMemcachedLibrary
  - Fonctions
  - Gestion des clés
- 3 Démonstration / Exemple
  - Signature
  - Chiffrement



- Réseaux et machines de plus en plus performants
- Temps de réponse d'une page web de plus en plus important
- Une solution : Stockage des données statiques ( > 5 min ) en RAM



- Stockage d'un couple clé/valeur dans la RAM
- De nombreuses bibliothèques d'utilisation dans de nombreux langages



- Stockage d'un résultat lors du premier appel d'une fonction
- Ou enrichissement de ces résultats par une tâche planifiée
- Récupération du résultat lors des prochains appels



Architecture la plus courante :

- 1 serveur
- 1 serveur web ( 1 VHOST )
- 1 serveur de base de données ( 1 base de données )
- 1 serveur Memcached

Mise en place de pages web pour nos 600 utilisateurs ( enseignants / chercheurs et étudiants )

- 2 serveurs
- 2 serveurs web ( N VHOST )
- 2 serveurs de base de données ( N base de données )
- 2 serveurs Memcached



### Accès réseau au serveur Memcached public

- Confidentialité des données
  - Lister toutes les clés
  - Lire les valeurs correspondantes
- Injection de code malveillant
  - Modifier la valeur d'une clé

Depuis la version 1.4.3 il est possible d'activer une authentification SASL

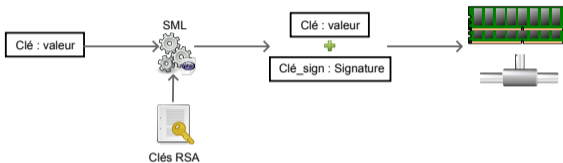
- limite l'accès à Memcached à des personnes autorisées
- Mais une fois connecté :
  - Lister toutes les clés de tous les utilisateurs
  - Lire et modifier les valeurs



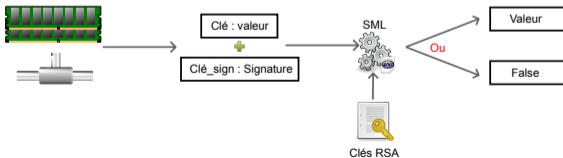
- Bibliothèque PHP
- Utilise les fonctions d'OpenSSL
- Basées sur RSA
  
- Signature
  - Assure l'intégrité des données
  - Mais elles restent publiques
  
- Chiffrement
  - Chiffre les données
  - Signe les données



### Signature :

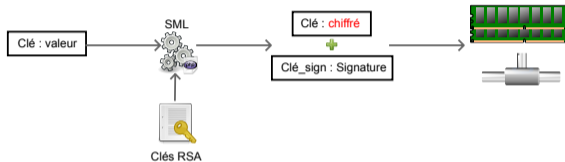


### Vérification :

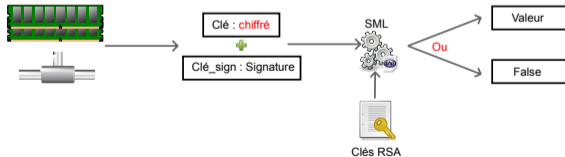




### Chiffrement :



### Déchiffrement et Vérification :





Toute la sécurité de la bibliothèque repose sur la sécurité de ces clés !

## 1 Stockage en mémoire partagée

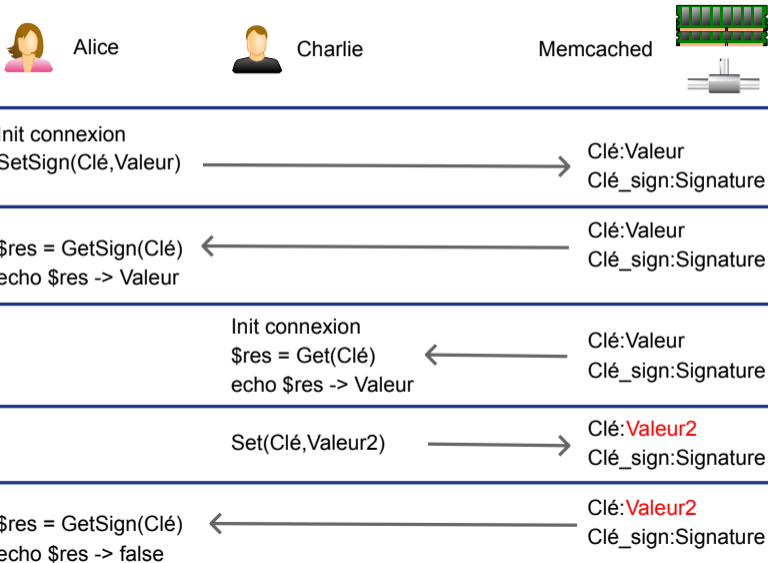
- Apache ITK
- 1 VHOST = 1 utilisateur ( UID ) et 1 groupe ( GID )
- Accès sécurisé ( chmod 600 ) et TRÈS rapide

## 2 Stockage sur le système de fichiers

- Autres cas
- Stocké dans un simple fichier
- Attention à la lisibilité de ce fichier ( utilisateur www-data, configuration serveur web, lsof, etc ... , etc ... )



# Démonstration / Exemple Signature





# Démonstration / Exemple

## Chiffrement



Alice



Charlie

Memcached



Init connexion

SetEncrypt(Clé,Valeur)



Clé:Chiffré

Clé\_sign:Signature

\$res = GetDecrypt(Clé)

echo \$res -> Valeur



Clé:Chiffré

Clé\_sign:Signature

Init connexion

\$res = Get(Clé)

echo \$res -> Chiffré



Clé:Chiffré

Clé\_sign:Signature

SetEncrypt(Clé,Valeur2)



Clé:Chiffré2

Clé\_sign:Signature

\$res = GetDecrypt(Clé)

echo \$res -> false



Clé:Chiffré2

Clé\_sign:Signature





- Sécurisation des transactions avec Memcached
- Amélioration des temps de réponse
- Légère perte de performance par rapport à l'utilisation sans SML
- Augmentation de l'occupation mémoire

Avez vous des questions ?

## Site du projet

- <https://forge.greyc.fr/projects/sml>