Fiche Technique

Cryptographie Symétrique en PHP

Symétrique

Intro

Système de cryptographie symétrique en PHP

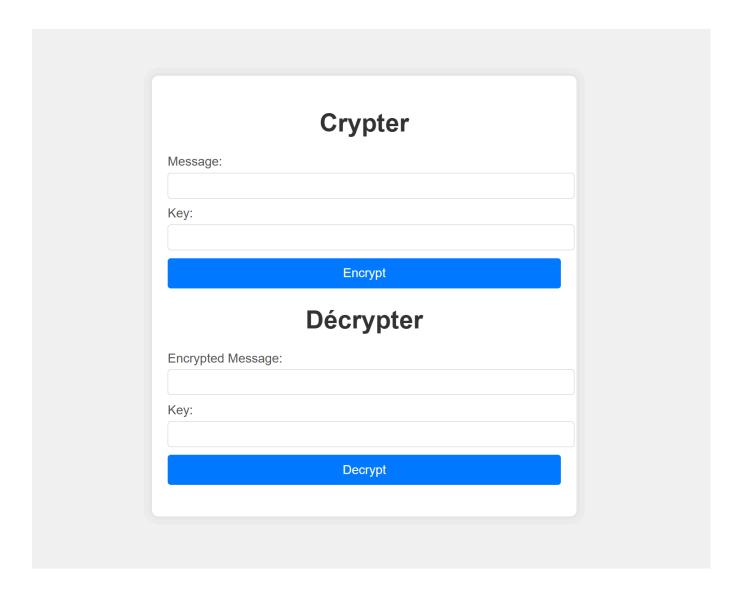
Algorithme : AES-256-CBC. Il permet de crypter et décrypter un message à l'aide d'une clé de 256 bits (32 octets).

Lancer le programme

- Ouvrir un terminal
- cd /chemin/vers/le/projet (php-encryption-project/Symétrique/src)
- php -S localhost:8000
- Key = EGX/sb/CA1SeeH79171XcaeFkb3OZ6Nhayw/jhawaRKlKgvJNOQWjAF+kg03FA6/
- Allez sur localhost:8000

Structure du code

- index.php : Interface utilisateur pour saisir et afficher les messages.
- encrypt.php : Fonction de chiffrement.
- decrypt.php : Fonction de déchiffrement.



Fonctionnement du code

Encrypte

```
function encryptMessage($plaintext, $key) {
$key = hash('sha256', $key, true);
```

Génère une clé de 32 octets à partir de la clé fournie.

Garantit une taille de clé compatible avec AES-256.

```
`$ivLength = openssl_cipher_iv_length('aes-256-cbc');`
```

Récupère la taille du vecteur d'initialisation (IV) requis.

Assure la cohérence avec l'algorithme utilisé.

```
`$iv = openssl_random_pseudo_bytes($ivLength); ``
```

Génère un IV aléatoire.

Ajoute de l'aléatoire pour chaque chiffrement, pour plus de sécurité.

```
`$encrypted = openssl_encrypt($plaintext, 'aes-256-cbc', $key, OPENSSL_RAW_DATA, $iv);`
```

Chiffre le message avec AES-256-CBC.

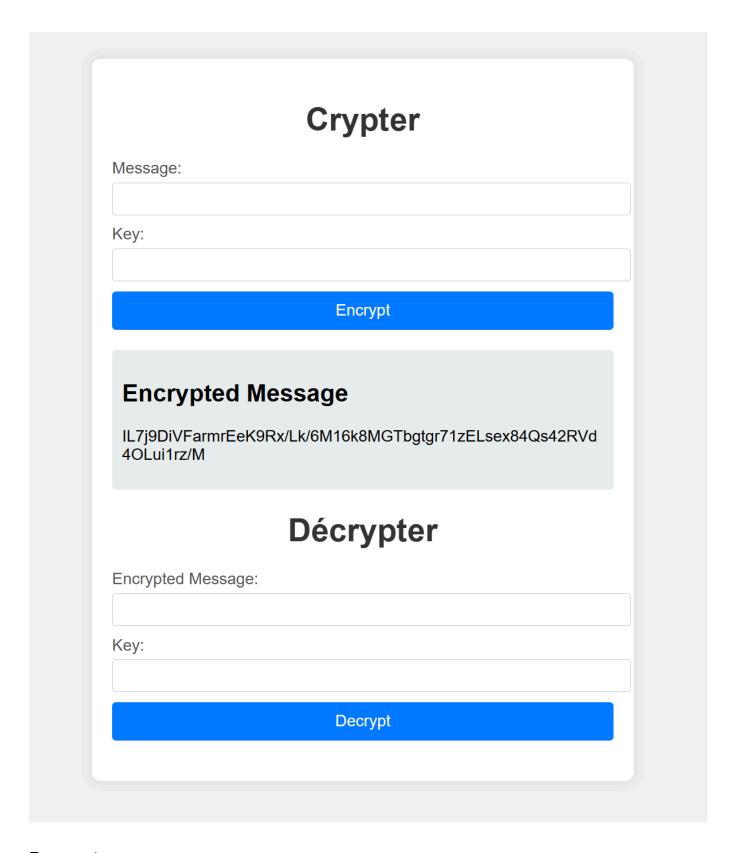
Transforme le texte clair en un texte illisible.

```
`return base64_encode($iv . $encrypted);`
```

Combine IV et message chiffré en base64.

Facilite le stockage et le transport.

Crypter
Message:
Salomon vous êtes Juif!
<ey:< td=""></ey:<>
EGX/sb/CA1SeeH79171XcaeFkb3OZ6Nhayw/jhawaRKIKgvJNOQWjAF+kg03
Encrypt
Décrypter
Encrypted Message:
Key:
Decrypt



Decrypte

function decryptMessage(\$encryptedMessage, \$key) { \$key = hash('sha256', \$key,
true);

Génère la même clé de 32 octets utilisée pour le chiffrement.

Cela assure la cohérence avec le processus inverse.

```
`$data = base64_decode($encryptedMessage);
```

Décode le message base64 en binaire.

Cela permet d'extraire l'IV et le message chiffré.

```
`$ivLength = openssl_cipher_iv_length('aes-256-cbc');
```

Récupère la taille de l'IV utilisée.

Nécessaire pour séparer correctement les données.

```
`$iv = substr($data, 0, $ivLength);
```

Extraie l'IV du message.

Utilisé pour déchiffrer le reste des données.

```
`$encryptedMessage = substr($data, $ivLength);
```

Récupère uniquement le message chiffré.

À décrypter avec la clé et l'IV.

```
return openssl_decrypt($encryptedMessage, 'aes-256-cbc', $key,
OPENSSL_RAW_DATA, $iv);
```

Déchiffre le message chiffré avec AES-256-CBC.

Retourne le message original.

Crypter			
Message:			
Key:			
Encrypt			
Encrypted Message			
IL7j9DiVFarmrEeK9Rx/Lk/6M16k8MGTbgtgr71zELsex84Qs42RVd 4OLui1rz/M			
Décrypter			
Encrypted Message:			
IL7j9DiVFarmrEeK9Rx/Lk/6M16k8MGTbgtgr71zELsex84Qs42RVd4OLui1rz/M			
Key:			
EGX/sb/CA1SeeH79171XcaeFkb3OZ6Nhayw/jhawaRKIKgvJNOQWjAF+kg03FA6/			
Decrypt			

	Crypter
Message:	
Key:	
	Encrypt
Encrypted M	aceana
4OLui1rz/M	Rx/Lk/6M16k8MGTbgtgr71zELsex84Qs42RVd
	Décrypter
Encrypted Message:	
Key:	
	Decrypt
Decrypted M	essage
Salomon vous êtes	

Interface Utilisateur (index.php)

- **Deux formulaires** : un pour le chiffrement et un pour le déchiffrement.
- Entrées : message et clé.
- **Sorties** : message chiffré ou déchiffré.

Améliorations Possibles

- Stocker les clés de manière sécurisée hors du code source, par exemple dans un fichier .env ou un coffre-fort de clés.
- Implémenter un mécanisme de journalisation (logs) pour suivre les opérations de chiffrement/déchiffrement.
- Ajouter une interface utilisateur plus intuitive avec des messages d'erreur clairs pour améliorer l'expérience utilisateur.

Asymétrique

Intro

Système de cryptographie Asymétrique en PHP

Algorithme : Il permet de crypter et décrypter un message à l'aide d'une clé privé et d'une clé privé.

Lancer le programme

- Ouvrir un terminal
- cd /chemin/vers/le/projet (php-encryption-project/Asymétrique/src)
- php -S localhost:8000
- Allez sur localhost:8000

Structure du code

- index_As.php :
- encrypt_As.php:
- decrypt_As.php:

Fonctionnement du code

Index

<?php

```
`require_once 'encrypt_As.php';`
```

```
`require_once 'decrypt_As.php';`
```

require_once : inclut chaque fichier une seule fois pour éviter les doublons. Les trois variables sont initialisées à vide pour éviter les erreurs.

```
`$message = '';`
`$encryptedMessage = '';`
`$decryptedMessage = '';`

`if ($_SERVER['REQUEST_METHOD'] == 'POST') {`
```

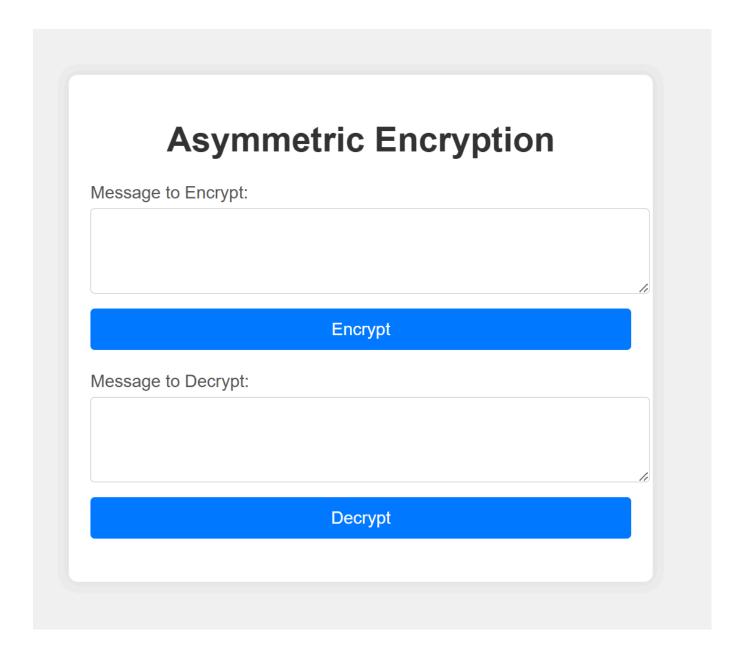
Vérifie si une requête POST a été envoyée

```
if (isset($_POST['encrypt'])) {
$message = $_POST['message'];
$encryptedMessage = encryptMessageAsymmetric($message,
'keys/public_key.pem');
Si le formulaire de chiffrement est soumis

} elseif (isset($_POST['decrypt'])) {
$encryptedMessage = $_POST['encrypted_message'];
$decryptedMessage = decryptMessageAsymmetric($encryptedMessage,
'keys/private_key.pem');
}
```

Si le formulaire de déchiffrement est soumis

 Ce fichier gère la logique principale en vérifiant le type de requête et en appelant les fonctions de chiffrement/déchiffrement. C'est comme une page de redirection qui envoie les visiteurs vers le bon service, chiffrement ou déchiffrement.



Encrypt

```
function encryptMessageAsymmetric($plaintext, $publicKeyPath) {
    $publicKey = file_get_contents($publicKeyPath);
    if (!$publicKey) {
    return 'Failed to load public key';
}

    $encrypted = '';

Vérifie si la clé publique est correctement chargée

    if (!openssl_public_encrypt($plaintext, $encrypted, $publicKey)) {
        return 'Failed to encrypt message';
    }
}
```

```
}
```

Chiffre le message avec la clé publique

return base64_encode(\$encrypted);

Retourne le message chiffré encodé en base64

 Cette fonction utilise la clé publique pour chiffrer les données avec OpenSSL. C'est comme mettre un secret dans une boîte fermée avec un cadenas public.

Asymmetric Encryption Message to Encrypt:	
Encrypt	/.
Encrypted Message Z+9MlM4zGl9pH6lRQN9dm3tGPrRYRIRNsUlQiGlr1a1V1H56Gfo1 uQNiV6W8xu17nETHqBzLaQ4HjmE4khuUn0wld1mC1Rem6rWiw5 S1F92CKO/tljSbLQRjVQfDUL+wqgCJUgPheZXHHnX2n9Z74VTtK m7/vj+Fh1NA5K4pb5pkgXHJprlX3R735imTrnolLKu2+y2PYixT1Fp0 REVFfFqyKVLrTQwGKLm62KtvK5eE1TaDhgL5NxPsiWDt3YJsBlFf iWMzcADCBYUEz1RRj/vTtRi7vCwXizLco50MCm84H+ANbbWAzn GznVOC2MtFUg7zgCT8M/kSJTTTL0ZKPQ==	
Message to Decrypt:	
Decrypt	

Decrypt

<?php

function decryptMessageAsymmetric(\$encryptedMessage, \$privateKeyPath) {
 \$privateKey = file_get_contents(\$privateKeyPath);
}

Définit une fonction en PHP qui prend deux paramètres le message chiffré à déchiffrer et le chemin du fichier contenant la clé privée.

```
if (!$privateKey) {
  return 'Failed to load private key';``}
  $encryptedMessage = base64_decode($encryptedMessage);
Vérifie si la clé privée est correctement chargée

if (!$encryptedMessage) {
  return 'Failed to decode base64';
}

$decrypted = '';
Vérifie si le message encodé est correctement décodé
  if (!openssl_private_decrypt($encryptedMessage, $decrypted, $privateKey))
{ return 'Failed to decrypt message'; }`
  return $decrypted;
```

Déchiffre le message avec la clé privée

• Cette fonction utilise la clé privée pour déchiffrer les données encodées. On utilise une clé spéciale (privé) pour ouvrir une boîte verrouillée.

Asymmetric Encryption

| Message to Encrypt: | |
|--|---|
| | |
| | / |
| Encrypt | |
| | |
| Encrypted Message | |
| Z+9MIM4zGI9pH6IRQN9dm3tGPrRYRIRNsUIQiGIr1a1V1H56Gfo1 uQNiV6W8xu17nETHqBzLaQ4HjmE4khuUn0wld1mC1Rem6rWiw5 S1F92CKO/tIjSbLQRjVQfDUL+wqgCJUgPheZXHHnX2n9Z74VTtK m7/vj+Fh1NA5K4pb5pkgXHJprIX3R735imTrnolLKu2+y2PYixT1Fp0 REVFfFqyKVLrTQwGKLm62KtvK5eE1TaDhgL5NxPsiWDt3YJsBIFf iWMzcADCBYUEz1RRj/vTtRi7vCwXizLco50MCm84H+ANbbWAzn GznVOC2MtFUg7zgCT8M/kSJTTTL0ZKPQ== | |
| Message to Decrypt: | |
| | |
| | / |
| Decrypt | |
| | |
| Decrypted Message | |
| Be or not to be | |

Interface Utilisateur (index_As.php)

• **Deux formulaires** : un pour le chiffrement et un pour le déchiffrement.

• Entrées : message

Sorties : message déchiffré.