Projet
Programmation par Composant

HMAC – SHA512

Auteurs :

- DOUR Marcellino
- PARDINI Raphaël

Historique des versions :

Version	Description	Date
1.0	Création du document	15/06/2022

I – DESCRIPTION

A. Contexte

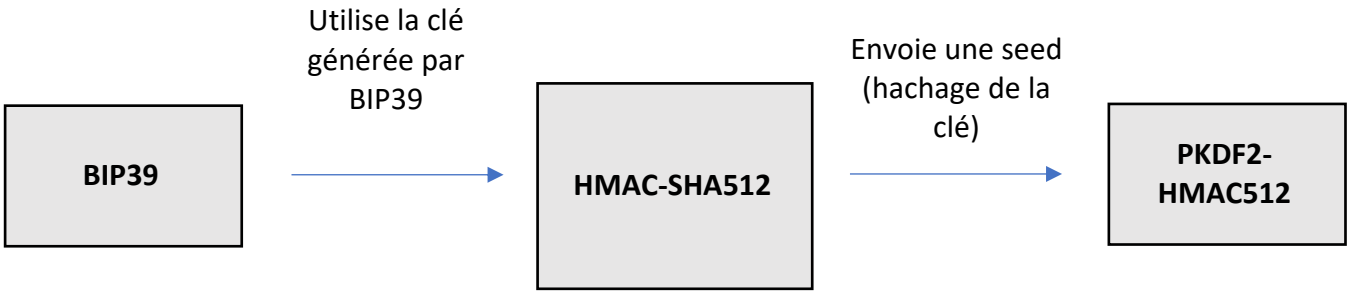
La création de ce composant intervient dans le cadre de notre cours « *Programmation par composant* », enseigné par Monsieur José Luu à l'Université Paris Dauphine. Le but du projet est de développer tous les composant d'une blockchain.

Dans notre groupe, nous allons implémenter la partie HMAC – SHA512.
C'est un type de code d'authentification de message, calculé en utilisant une fonction de hachage cryptographique en combinaison avec une clé secrète. Comme avec n'importe quel MAC, il peut être utilisé pour vérifier simultanément l'intégrité de données et l'authenticité d'un message. N'importe quelle fonction itérative de hachage, comme SHA-256 ou SHA-512, peut être utilisée dans le calcul d'un HMAC ; le nom de l'algorithme résultant est HMAC-SHA-256 ou HMAC-SHA-512. La qualité cryptographique du HMAC dépend de la qualité cryptographique de la fonction de hachage et de la taille et la qualité de la clé.

B. Interface et interaction avec chaque autre composant

Cette clé, composé de 24 mots, sera fourni par le composant 1 : décodeur / codeur BIP39.
Puis, la seed généré par la fonction de hachage sur ces mots sera utilisé par le composant 3 : PKDF2-HMAC512.

Voici un schéma représentant les différents composants connexes et leurs interactions :



C. Fonctions python

Dans notre composant, nous avons une fonction python d'interface. Elle est nommée **hmac_sha512()**.

Cette fonction prend en entrée une chaine de caractère, et renvoie en sortie une chaine de caractère.

N'importe quelle chaine de caractère peut être passée en entrée.

D. Cas d'erreurs

Notre fonction peut calculer un hachage quelques soit l'argument passé en entrée. Ainsi, nous n'avons pas de gestion d'erreur à gérer, car notre méthode ne peut pas rencontrer d'erreur d'exécution lié à un mauvais paramètre.

II – Tests

A. Plans de tests

Afin de tester notre composant, nous avons choisi d'utiliser le plan de test suivant :

- Nous allons générer une dizaine de clés via un codeur / décodeur BIP39.
- Puis, nous allons utiliser la fonction HMAC-SHA512 de notre composant, et comparer les résultats avec la fonction d'un programme en ligne.

Nous avons choisi les programmes en ligne suivant :

- BIP39 → https://iancoleman.io/bip39/#french
- HMAC-SHA512 → https://emn178.github.io/online-tools/sha512.html

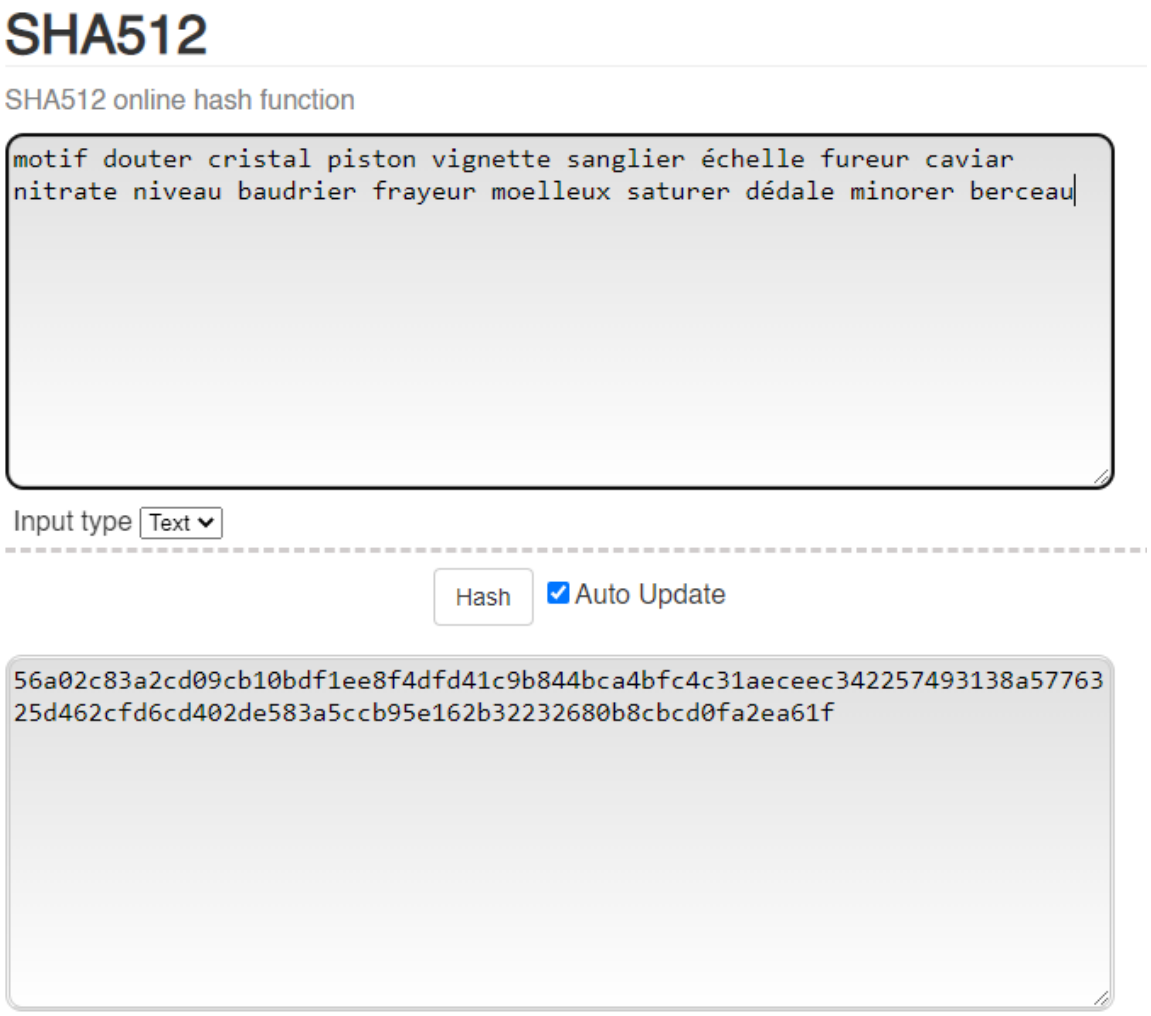
B. Programme de tests

Dans notre programme de test, nous allons stocker une liste de couple clé/seed. Puis, nous allons appliquer notre fonction sur chacune de ces clés, et vérifier que la seed obtenu est correct.

De plus, nous vérifierons que lorsque la clé en entré est incorrect, le programme renvoie bien une exception.

Voici des captures d'écran de l'application de ces tests :

Capture d'écran de deux clés et de deux résultats via un programme en ligne



Capture d'écran des résultats de notre programme avec les mêmes clés passées en paramètre

```
>>> import hmac_sha512_component
>>> data = "motif douter cristal piston vignette sanglier échelle fureur
 caviar nitrate niveau baudrier frayeur moelleux saturer dédale minorer
berceau"
>>> hash = hmac_sha512_component.hmac_sha512(data)
>>> print(hash)
56a02c83a2cd09cb10bdf1ee8f4dfd41c9b844bca4bfc4c31aeceec342257493138a5776
325d462cfd6cd402de583a5ccb95e162b32232680b8cbcd0fa2ea61f
```