Spécifications pour le Composant de Récupération de la Clé Publique d'une Signature ECDSA

Auteurs : Giovanni AMOUSSOU et Yufei LIU

Historique des versions

Version 1.0 (28 mai 2023) : Première version du document

Version 2.0 (28 juin 2023) : Mise à jour avec des informations complémentaires.

Version 3.0 (30 juin 2023) : Dernière mise à jour du document de spécifications.

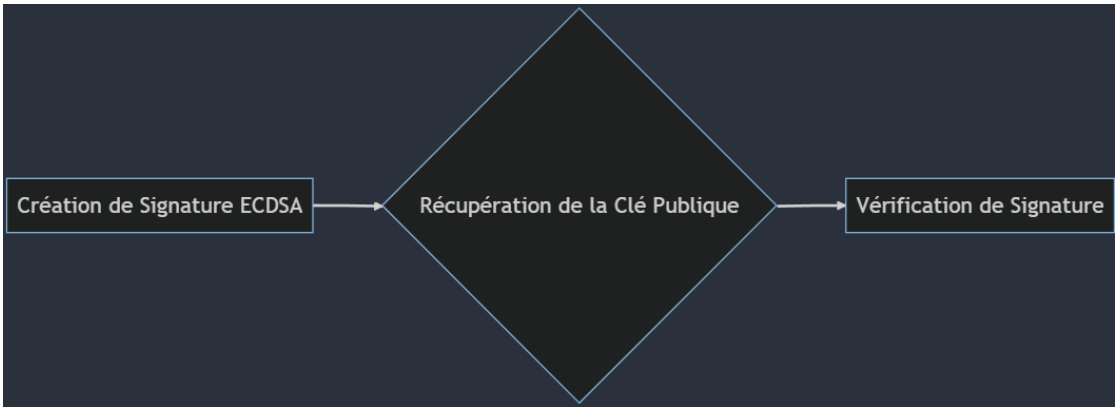
Table des matières

I. Description

1. Contexte :

La récupération de la clé publique d'une signature ECDSA est un processus crucial dans la vérification de l'intégrité et de l'authenticité d'un message numérique. Ce composant facilite cette tâche en fournissant une interface simple pour extraire la clé publique à partir d'une signature ECDSA donnée.

2. Schéma bloc incluant les composants connexes :



3. Interface et interaction avec chaque autre composant :

Le composant reçoit une signature ECDSA sous forme hexadécimale du module de création de signatures.

Le composant fournit la clé publique extraite, représentée en tant que chaîne, au module de vérification de signatures.

```
def recover_public_key(signature: str) -> str:
    """
    Récupère la clé publique à partir d'une signature ECDSA.

    :param signature: La signature ECDSA en hexadécimal (par exemple, '0x1a2b3c...').
    :return: La clé publique en format de chaîne.
    :raises ValueError: Si la signature est invalide ou n'est pas un hexadécimal correct.
    """
    pass
```

4. Cas d'erreurs :

Si la signature fournie n'est pas en format hexadécimal, le composant lève une exception **ValueError** avec un message spécifiant que la signature doit être une chaîne hexadécimale.

Si la signature fournie est invalide ou ne correspond à aucune clé publique connue, le composant lève également une exception **ValueError**.

Le code de la fonction pourrait ressembler à ceci :

```python
def recover_public_key(signature: 'hexadécimale (chaîne)') -> str:
    """
    Récupère la clé publique à partir d'une signature ECDSA.

    :param signature: La signature ECDSA en hexadécimal (par exemple, '0x1a2b3c...').
    :return: La clé publique en format de chaîne.
    :raises ValueError: Si la signature n'est pas un hexadécimal correct ou si la signature est invalide.
    """
    # Vérifie que la signature est un hexadécimal correct
    try:
        int(signature, 16)
    except ValueError:
        raise ValueError("La signature doit être une chaîne hexadécimale")

    # Autre code pour récupérer la clé publique...
    pass
```

II. Test

1. Plan de test :

Le composant sera testé en utilisant un ensemble de signatures ECDSA et de clés publiques connues. Pour chaque paire de clés, le composant doit récupérer avec succès la clé publique à partir de la signature. Le plan de test consiste donc à :

1. Générer une paire de clés ECDSA (clé privée et clé publique).

2. Utiliser la clé privée pour créer une signature ECDSA d'un message donné.

3. Passer cette signature (sous forme hexadécimale) à la fonction ***`recover_public_key()`***.

4. Vérifier que la clé publique retournée par la fonction ***`recover_public_key()`*** correspond à la clé publique originale générée à l'étape 1.

La correspondance entre les deux clés publiques est l'élément clé du test. Si elles correspondent, cela signifie que la fonction ***`recover_public_key()`*** a correctement extrait la clé publique de la signature ECDSA. Sinon, si elles ne correspondent pas, cela indique un problème avec la fonction ***`recover_public_key()`.***

2. Mode d'emploi :

Pour exécuter le programme de test, lancez simplement la fonction ***`test_recover_public_key()`.*** Si la fonction s'exécute sans lever d'exception, cela signifie que le composant fonctionne correctement. Sinon, si une exception est levée, le composant a échoué au test.

Pour en savoir plus vous pouvez vous référé au README présent sur notre lien github :