SV53: Sécurité système et réseaux

Rapport de projet

Sujet :

Blockchain

Ecrit par:

AZINCOURT Vincent

GUEYE Fatou

Informations complémentaires :

Département Informatique : SV53-A21

Abdeljalil Abbas-Turki

Étape 1 : Étude la blockchain

1. Après l'analyse du code on détermine trois parties dans la structure de notre blockchain :

```python
# fichier main.py
"""
une Blockchain basique sur Python
"""

import hashlib
class GeekCoinBlock:

    def __init__(self, previous_block_hash, transaction_list):

        self.previous_block_hash = previous_block_hash
        self.transaction_list = transaction_list

        self.block_data = f"{' - '.join(transaction_list)} - {previous_block_hash}"
        self.block_hash = hashlib.sha256(self.block_data.encode()).hexdigest()

class Blockchain:
    def __init__(self):
        self.chain = []
        self.generate_genesis_block()

    def generate_genesis_block(self):
        self.chain.append(GeekCoinBlock("0", ['Genesis Block']))

    def create_block_from_transaction(self, transaction_list):
        previous_block_hash = self.last_block.block_hash
        self.chain.append(GeekCoinBlock(previous_block_hash, transaction_list))
```

On y trouve donc trois parties:

1. L'empreinte numérique du bloc précédent
2. La donnée du bloc : transaction
3. L'empreinte numérique du bloc actuel

En exécutant le code, on obtient ceci :

Data 1: Genesis Block - 0
Hash 1: 39331a6a2ea1cf31a5014b2a7c9e8dfad82df0b0666e81ce04cf8173cc5aed3e

Data 2: J'ai dépensé 5 € chez amazone - J'ai dépensé 70 € à Total - 39331a6a2ea1cf31a5014b2a7c9e8dfad82df0b0666e81ce04cf8173cc5aed3e
Hash 2: d58a4ce7eb1d27c32330f56ac99222635c6df05bc4115796c3f063e677202bdf

Data 3: J'ai reçu 2100€ de salaire - J'ai fait 100€ de course à Auchan - d58a4ce7eb1d27c32330f56ac99222635c6df05bc4115796c3f063e677202bdf
Hash 3: 60ea3eb8f2729da1719ea7203837ad399286188f23fb1c8d536556e245835854

Data 4: J'ai payé 110€ de facture d'électricité - J'ai payé 30€ de facture de téléphone - 60ea3eb8f2729da1719ea7203837ad399286188f23fb1c8d536556e245835854
Hash 4: 6867487ab729aefc001dbb1b1c619a30ad59e7d28043553800b66f06cdaee072

Message de la transaction (en rouge)

Empreinte numérique du bloc précédent (en jaune)
Nouvelle empreinte du bloc (actuel) (en bleu)

2. Les parties manquantes de la blockchain sont :

- ☐ La signature de la transaction qui peut être réalisé en générant un clé publique et une clé privé par un procédé de courbes elliptiques
- ☐ et la preuve de travail qui est une phrase ajouté

3. Se référer au code main.py pour l'ajout de la signature. Celle çi est renvoyée en sous forme d'entier et est unique pour chaque message.

Si l'on décortique chaque phase de la signature, on construit tout d'abord la clé privée (SigningKey.generate())

Clee privee = c3a904a9d03fdee1adcbf1b2d098a4409508c268360f4e22

Puis, à partir de celle-ci, on fait la clé publique (SigningKey.generate().verifying_key)

Cle publique = 02972eee7eb9aedff6e1670c7f9d5d27c0ab2563f1fcc2185d

Puis on construit le message chiffré (byte)

Message chiffre =
d58a4ce7eb1d27c32330f56ac99222635c6df05bc4115796c3f063e677202bdf

et enfin on génère signe le bloc à partir des informations précédentes, il s'agit également de byte que l'on peut décoder avec cette commande : int.from_bytes(sign, byteorder = 'big')

Signature =
24229493651710385956821903059040117788200554809533901668604240026255665522377512330058713622570201764147679961458807

4. Étant donné l'absence de preuve de travail, cette blockchain présente des risques d'être dupliqué et dans notre cas de créer de fausses transactions.

En mettant en oeuvre la signature et la preuve de travail, on obtient ainsi ce résultat :

La Blockchain est valide.

Data 1: Genesis Block - 0
Signature 1: 0
Hash 1: 39331a6a2ea1cf31a5014b2a7c9e8dfad82df0b0666e81ce04cf8173cc5aed3e
Proof of work 1: 1

Data 2: J'ai dépensé 5 € chez amazone - J'ai dépensé 70 € à Total -
39331a6a2ea1cf31a5014b2a7c9e8dfad82df0b0666e81ce04cf8173cc5aed3e
Signature 2:
220311725171891691667455873061256294007187015577414537019272307278625817907369939266548817309492629928154367780396 64
Hash 2: d58a4ce7eb1d27c32330f56ac99222635c6df05bc4115796c3f063e677202bdf
Proof of work 2: 632238

Data 3: J'ai reçu 2100€ de salaire - J'ai fait 100€ de course à Auchan -
d58a4ce7eb1d27c32330f56ac99222635c6df05bc4115796c3f063e677202bdf
Signature 3:
294556777366529732157913468566533266153650492354408437492665451863841289003669654135876454755676543754744092895603 34
Hash 3: 60ea3eb8f2729da1719ea7203837ad399286188f23fb1c8d536556e245835854
Proof of work 3: 403091

Data 4: J'ai payé 110€ de facture d'électricité - J'ai payé 30€ de facture de téléphone -
60ea3eb8f2729da1719ea7203837ad399286188f23fb1c8d536556e245835854
Signature 4:
904658462402148337779573052566367836684837261977096444168109245875755812024172892735289711648854185942548763630206 2
Hash 4: 6867487ab729aefc001dbb1b1c619a30ad59e7d28043553800b66f06cdaee072
Proof of work 4: 714736

La première étape (en violet) permet de vérifier la validité du bloc via la réponse avec succès à la preuve de travail. Pour rendre l'exploitation minière difficile, la preuve de travail doit être suffisamment dure pour être exploitée.
Après avoir extrait le bloc avec succès, le bloc sera alors ajouté à la chaîne.
Après avoir miné plusieurs blocs, la validité de la chaîne (en gris) doit être vérifiée afin d'éviter toute sorte de falsification de la blockchain.
La signature vue à l'étape précédente, la signature du bloc (en vert), à également été ajoutée à chaque bloc de la chaîne.