Projet : Blockchain

1. Introduction

Ce projet à consisté en la création d'une blockchain simple afin d'en illustrer les mécanismes. Il s'agit d'une blockchain centralisée avec un serveur principal qui peut avoir de multiples clients. Le processus de consensus est une preuve de travail basé(PoW) sur du SHA-256.

Toutes les communications se font via des messages en udp et le projet peut marcher en ligne si les pare-feu laissent passer les messages (il faudra cependant rentrer les bonnes adresses dans le code). Par défaut le projet se lance en localhost (127.0.0.1)

2. Lancer le projet

Pour lancer le projet il suffit d'exécuter les 2 fichiers python sur 2 terminaux différents. Il faut cependant que les ports 8888,8887 et 5000 soient libres afin que les connexions puissent se faire.

Pour exécuter les serveur :

python3 serveur.py

Pour exécuter le client :

python3 client.py

Il n'y a pas d'argument ni pour le client ni pour le serveur. Il n'y pas aussi d'ordre pour les lancer. En revanche, il faut qu'un serveur soit lancé pour que le client puisse faire ses actions.

3. La structure des blocs

```
block = {
    "index": index,
    "timestamp": timestamp,
    "transactions": transactions,
    "previous_hash": previous_hash,
    "nonce": nonce
}
```

Les blocs contiennent 5 champs différents :

- index : (int) numéro du bloc
- timestamp : (str) date et heure de la création du bloc
- transactions : (list[str]) liste des transactions dans le bloc
- previous_hash : (str) hash du bloc précédent
- nonce : (int) nonce qui permet d'arriver à la target

Ils sont sous la forme d'un dictionnaire et on peut les avoir sous une forme écrite dans le fichier ***blockchain.json***. Le hash du bloc est la valeur hashé du bloc sous la forme de chaîne de caractère.

4. Le serveur

Le serveur est la partie qui va gérer la blockchain. Il est en charge de tout ce qui est création, chargement si le fichier ***blockchain.json*** existe et ajout de block. Le serveur au démarrage va aussi vérifier l'intégrité de la blockchain afin de vérifier si elle est cohérente.

Le serveur lorsqu'il reçoit un message d'ajout de block (reçu sur le port 8888) il va envoyer en broadcast sur le port 5000 un message pour demander aux mineurs de chercher un nonce correcte. Ce message est envoyé toutes les 3 secondes en broadcast. Lors de la réception d'un nonce (port 8887) il va vérifier s'il est correct puis ajouter le bloc dans sa mémoire et dans le fichier de sauvegarde. Il envoie aussi un message de fin de recherche en broadcast sur le port 5000 pour annoncer la fin de la proof of work.

L'autre requête qu'il peut recevoir sur le port 8888 et une demande d'envoie de la blockchain. Le serveur va alors répondre au client qui a envoyé le requête la blockchain sous une forme de string convertit en bit.

5. Le client

Le client lui à de multiple possibilité d'action :

```
billy@BillyTheChild:/mnt/c/Informatique$ /bin/python3 /mnt/c/Informatique/M1/S2/Crypto/projet/client.py
Que souhaitez-vous faire ?
1. Ajouter une transaction
2. Obtenir la blockchain complète
3. Devenir un mineur de block
4. Verifier l'intégrité
5. Quittez
6. Devenir un mineur de block en partant d'un nonce x
```

1. L'ajout de transaction permet à l'utilisateur de rentrer toute une chaîne de caractère et envoie ensuite au serveur la transaction qu'il ajoutera dans le prochain bloc.

2. L’obtention de la blockchain permet de recevoir la version du serveur et l’affiche dans le client.

3. Le client se met en attente d’un message de la part du client pour commencer une PoW. Une fois le message reçut le client va afficher le nonce et le hash correspondant. A chaque passage il va aussi lire sur le port 5000 pour savoir s'il doit interrompre sa recherche. Une fois trouvé un bon nonce il va le transmettre au serveur via le port 8887.

4. Le client demande la blockchain du serveur et va ensuite refaire le hash des blocs et vérifier si la blockchain est cohérente avec les hash précédents.

5. Éteint le client

6. Identique à l'étape 3. sauf que le client peut préciser où commencer la recherche nonce. Cela permet dans le cas de plusieurs clients d'accélérer la recherche.

6. Conclusion

Ce projet permet donc la mise en place d’une simple blockchain en proof of work. Pour être plus proche des blockchains bien connues et utilisées dans le monde il faudrait aussi qu’il y ait un système de récompense pour les mineurs. Une vérification dans le serveur des transactions serait nécessaire aussi ainsi que l'instauration de droits en fonction des différents clients.

Chez le client il serait intéressant de mettre en multithreading la recherche de nonce dans la proof of work pour accélérer le processus.

Malgré cette simplicité, cette blockchain permet une bonne description de ce qu’est une blockchain d’un point de vue pratique et permet une bonne initiation du concept auprès de néophyte.