Rapport :

Rappel du projet :

Création d'un réseau gérant une bloc chaine.
Le réseau est constitué de nœud blocs qui portent un nombre n de thread représentant des participants.
Les participants génèrent des transactions à intervalles réguliers ou non.
Les nœuds blocs génèrent des blocs validant les transactions des participants.
Lorsqu'un nœud bloc génère un bloc il distribue des points blocs entre les participants en fonction de leurs contributions. La contribution d'un participant dépend du nombre d'autre participants rattachés au nœud et du nombre de transaction qu'il aura réalisé par rapport aux autres. Une transaction correspond à un don d'une partie des points blocs d'un participant vers un autre. Chaque participant débute avec 1 point bloc.

Langage utilisé :

Le projet est réalisé en python 3.6 en utilisant l'API RPYC (3.4.4).
Rpyc est une API toujours en développement mais très intéressante sur son principe permettant l'appel distant de fonction d'un serveur à un autre.
Une fois connecté à un serveur il est possible d'accéder aux fonctions custom portant un nom spécifique : exposed_xxx()
Ainsi après avoir créé et stocké l'objet permettant la connexion à un serveur on peut accéder de façon distante à ces fonctions :

Serveur local :

```
c = rpyc.connect(ip,port)
c.root.xxx()
```

Serveur distant :

la fonction exposed_xxx() est appelée.

Via ce procéder il est possible d'agir via les fonctions exposées sur le serveur distant ou de passé certain type d'argument à cette dernière, ces derniers sont toutefois restreint aux types naturels de python : int, char, string, … Malheureusement comme dit plus haut l'API est toujours en développement et certains artifices ont dû être mis en place pour se servir efficacement de l'API notamment l'utilisation d'une variable global pour placer certaines ressources critiques dans l'espace de nom de tous les acteurs.

Présentation des différents messages et de la bloc chaine :

Un bloc est généré lorsque qu'un hash contenant au moins 5 zéros successifs est trouvé. Pour ce faire le nœud hash la bloc chaine précédente réduite à une string en y ajoutant un nonce. Il est obtenu en tirant 32 caractères d'un octet soit 255 valeurs moins les 5 caractères suivant '\n', ' ' , ' :' , ' ;' et '\0' (pour une gestion plus simple une fois la bloc chaine sous forme de string).
Un bloc est composé de sa profondeur, son hash, son nonce, son créateur, du nombre d'opérations et la liste de ces opérations.
Une bloc chaine est représenté sous forme décroissante : c'est le bloc le plus profond qui est en tête de chaine.

Il existe deux types de messages : les missives et les opérations
Les opérations sont générées par les participants lors de leurs actions : connexion (C), déconnexion (D), erreur (E) et transaction (T).
Format en string :

C : 'C : numéro d'opération :id participant :id nœud'
D : 'D : numéro d'opération :id participant :id nœud'
E : 'E : numéro d'opération : id participant : id nœud : nombre de testes'

T :'T : numéro d'opération : id participant : id nœud : id participant cible : id nœud cible : valeur transaction'

Il existe également les opérations G émise par le nœud lui-même lors de la création d'un bloc. Elles sont constituées de nœud d'origines, du hash généré, du nonce généré et de la profondeur.

Les missives sont les messages transitant sur le réseau. Il sont de 4 types : S, R , Q ou P.
Les types Q et P ne sont utilisés uniquement pour la création de log pour savoir si une opération ou bloc chaine a été acceptée ou non.
Les missives S transportent des opérations, les R transportent les bloc chaines.
Pour transiter sur les réseaux les missives sont réduites en string via le procéder suivant : chaque champ de la missive et chaque bloc d'une bloc chaine sont séparés par ' ', chaque champ d'un bloc est séparé par '\n', chaque opération d'un bloc par ' ;', chaque champ d'une opération par ' :'.

Diffusion et acceptation des missives entrantes :

Lorsqu'une nouvelle opération arrive depuis le réseau on vérifie que cette dernière est connue ou non. Si l'on a déjà vu cette opération cette dernière n'est pas traitée. Dans le cas contraire elle est ajoutée aux opérations courante et mise en attente de bloc. S'il s'agit d'une connexion ou d'une déconnection cette dernière prend effet immédiatement. L'opération est alors diffusée dans le réseau vers tous les voisins sauf l'émetteur de l'opération.
Si l'opération est généré par les participants du nœud elle est directement ajoutée aux opérations courantes puis diffusée à tous les voisins.

Validation d’une opération et validation d’un bloc :

Les opérations C et D sont considérés comme valide dès leur réception. Les E sont pris en compte uniquement par le nœud les émettant. Les T ne sont validés que lorsqu’ils sont présents dans un bloc à une profondeur égale à la profondeur max moins 3.
Les opérations G sont validées par le nœud dans le cas où la profondeur du bloc généré est strictement supérieure à la profondeur courante. A ce moment-là les opérations courantes sont ajoutées à ce nouveau bloc et la nouvelle est bloc chaine est diffusée sur le réseau.
Lors de l’arrivé d’une nouvelle bloc chaine celle-ci n’est accepté que si sa profondeur est strictement supérieur à celle courante. Dans les cas où la nouvelle bloc chaine est acceptée on retire de la liste courante la totalité des opérations présentent dans la nouvelle bloc chaine.

Remarques et réflexions :

De par la nature des serveurs RPYC les réseaux formables sont comparables à des graphes orientés ou non.
Il arrive que lors de terminaison le dernier serveur RPYC raise une erreur, sans conséquence toute fois sur les logs (apparition de l’erreur lors de la déconnection du serveur).
Un nœud bloc s’arrête lorsqu’il n’a plus aucune destination atteignable et plus aucune opération courante.
Pour simplifier le lancement des testes il faut utiliser la commandes : python3 launch.py file file est remplacé par l’un des fichiers .ini présent à la racine. Pour réaliser un test custom il suffit de créer un fichier ini avec autant de lignes que de nœud sous le format :
‘nombre de participant max simultané’ ‘nombre totale de participant’ ‘voisin 1’….’voisin n’