Nathan Leclercq

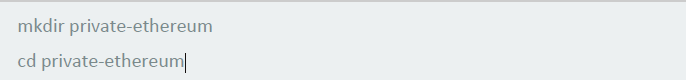
**Rapport TP blockchain Groupe 9**

Nous sommes le Gr9 et l’adresse IP de notre serveur fournit est 64.227.65.50 ainsi J’y accède directement depuis ma machine grâce à Putty, je choisis L’identifiant d’ESME9, puis je me connecte en tant que root avec le mot de passe donné (94EsmeSudria).

**Création de comptes pour le réseau privé Ethereum**

Dans un premier temps, j’ai créé le repertoire private-ethereum sur lequel je vais travailler et effectuer l’ensemble des manipulations pour le réseau.

Voici les 2 lignes de commandes utilisées, la première pour créer le répertoire et la deuxième pour aller sur le répertoire.



Suite à cela, j’ai dû créer 2 comptes ( wallet ) avec la commande suivante :

 Une image contenant texte

Description générée automatiquement

Nous pouvons voir ci-dessus les différentes adresses des deux comptes grâce à la commande geth –datadir data account list qui permet d’afficher l’ensemble des comptes éxistants.

**Création du Genesis File uniquement sur le serveur fourni dans un premier temps**

Un fichier Genesis contient les propriétés qui définissent la Blockchain. Un fichier Genesis est le point de départ de la Blockchain et il est donc obligatoire de créer le fichier Genesis pour créer une Blockchain.

Cette commande permet de créer le fichier genesis.json et d’écrire le code.



Une image contenant texte

Description générée automatiquement

Voici le code de notre fichier génésis.

Pour les paramètre, nous avons remplacé le chainID par le notre ( 904112022 ).

**HomesteadBlock** : version d’Etherum, généralement fixé à 0 dans le fichier genesis.json

**Eip150Block … petersburgBlock** : Ceux sont des protocols servant à protéger notre blockchain. Ils sont souvent configurés à 0 dans le fichier genesis.json

**Difficulty** : Permet de définir la difficulté de minage, ici 1 signifie que le minage sera très facile.

**Alloc** : Permet d’attribuer des ether aux adresses ( ici les 2 comptes que nous avons créé ) inidiquées

Initiation du genesis.json, on peut voir que tout a bien été initialisé.

Une image contenant texte

Description générée automatiquement

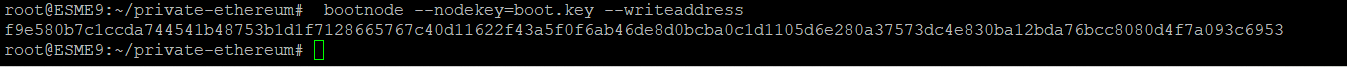
**Configuration du Bootnode uniquement sur le serveur dans un premier temps**

Grâce à ces nœuds de démarrage, un nœud peut rejoindre le réseau et trouver d'autres nœuds. Dans le cas d'un cluster privé, ces nœuds de démarrage prédéfinis ne sont pas d'une grande utilité. Par conséquent, go-ethereum propose une implémentation de bootnode qui peut être configurée et exécutée dans votre réseau privé.

Une image contenant texte

Description générée automatiquement

Après vérification de l’adresse du bootnode, nous pouvons voir que les 2 adresses d’identifiants sont bien identiques.



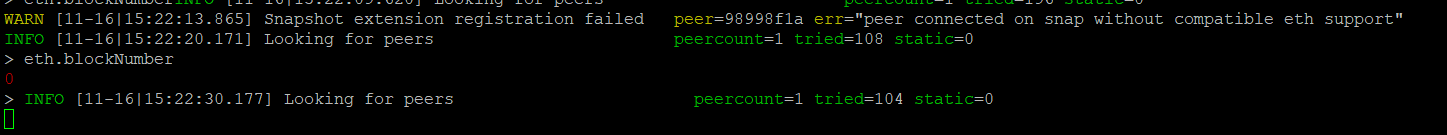
**Configurez votre Ethereum Private Blockchain et commencer à miner uniquement sur le serveur dans un premier temps**

****

Nous avons lancé la commande ci-dessus en remplaçant les données surlignées en jaune ou en verte par nos paramètres ainsi que le networkid, ce qui donne :

geth --mine --networkid 904112022 --http.port 30303 --nat extip:64.227.65.50 --miner.etherbase=0x8100df3fb57b7232ca236552e3439feefef4df18 --datadir data console

Une fois cela fait, nous devons vérifier l’état de la Blockchain ainsi que le numéro de Block qui doit se trouver à 0.



Nous vérifions aussi que notre compte possède bien les ether alloués grâce au fichier genesis.json et nous devons vérifier les Genesis Block ( 0 ). ( pas de scree mais tout était bon )

Comme tout est bon, nous lançons ensuite le minage. Voici le minage, voici le block 1. Une image contenant texte

Description générée automatiquement

Nous lançons ensuite la commande suivante afin d’affecter notre IP à notre nœud.

bootnode --nodekey=boot.key --addr 64.227.65.50.30303

Voici l’URL enode que les autres nœuds pourront utiliser afin de se connecter et d’échanger des informations.

self=enode://0ea118c53fcde460dee9e57025320c73b643be0cdf86ceaa18469a6a365a3d3c3a8e939619ba764b347f61d2f3387e619aafb1228cfcf5b3cbdbeb8089a8af2e@64.227.65.50:30303

**Démarrage de vos noeuds sur les ordinateurs de vos camarades**

J’ai lancé les deux commandes suivantes en remplaçant le networkid et l’IP par les informations de notre serveur.

La commande net.listening me retournait « true ».

Une image contenant texte

Description générée automatiquement

Une fois cela fait, la synchronisation avait bien commencé mais le serveur et le client n’avaient pas les mêmes URL enode, nous ne pouvions donc pas continuer le TP car cela signifie que le client n’était pas synchronisé avec le serveur. Nous avions pourtant le même fichier genesis.json ( espace, retour à la lignes … ), tout était identique.

Une image contenant texte

Description générée automatiquement