

BLOCKCHAIN

Rapport TP

Ce TP a pour objectif la création et la configuration d'une Blockchain
Privée Ethereum

Inès DERNONCOURT
A3 MSI



TABLE DES MATIERES

Introduction.....	2
Etape 1 : Installation d'Ethereum.....	2
Etape 2 : Création de comptes pour le réseau privé Ethereum	3
Etape 3 : Création du Genesis File.....	4
Etape 4 : Configuration du Bootnode	6
Etape 5 : Configure notre Ethereum et commencer à miner	7
Etape 6 : Démarrage des nœuds sur nos ordinateurs	8
Conclusion	9
Bonus	10

INTRODUCTION

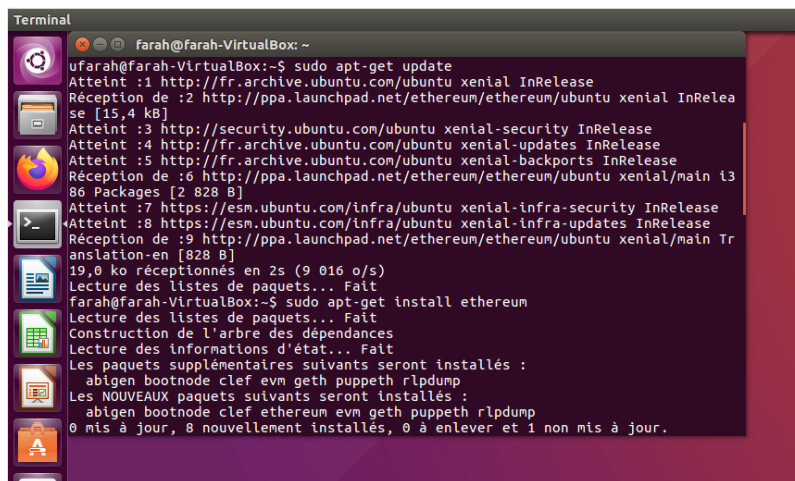
Je fais partie du groupe 3 qui a le serveur ESME3 dont l'adresse IP est 64.225.66.111. Ishika HOSSAIN se connecte au serveur.

Aussi pour information, ma Machine Virtuelle est sur Ubuntu 16 et avait été envoyée l'année dernière dans le cadre d'une autre matière par une camarade de classe Farah. Par soucis de place sur mon ordinateur je vais utiliser cette machine. C'est pour cela que toutes mes lignes de commandes commencent par « farah@farah ».

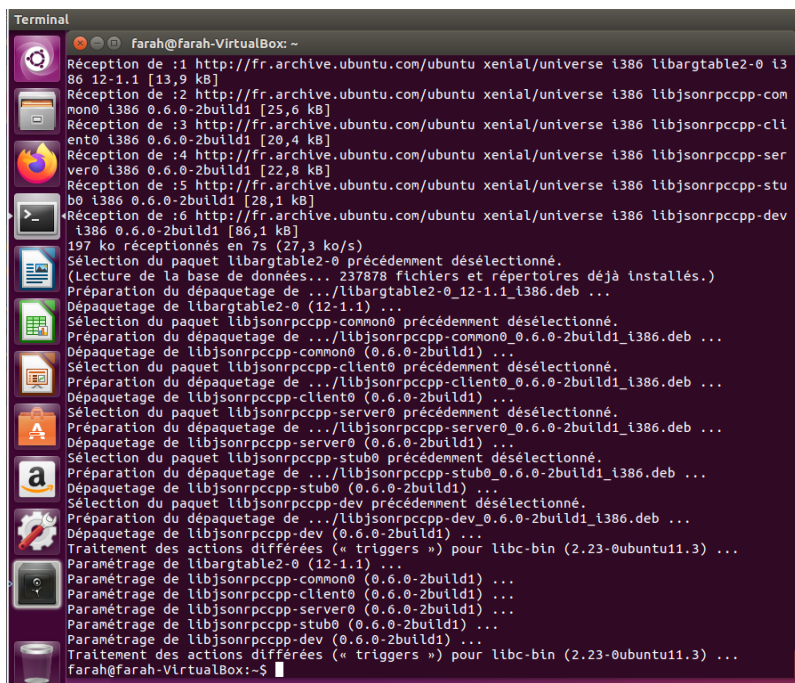
ETAPE 1 : INSTALLATION D'ETHEREUM

Dans cette étape, nous devons installer Ethereum sur nos VM.

Les captures d'écran ci-dessous de mon terminal montrent que l'installation a été réalisée avec succès :



```
Terminal
farah@farah-VirtualBox: ~
ufarah@farah-VirtualBox:~$ sudo apt-get update
Atteint :1 http://fr.archive.ubuntu.com/ubuntu xenial InRelease
Réception de :2 http://ppa.launchpad.net/ethereum/ethereum/ubuntu xenial InRelease [15,4 kB]
Atteint :3 http://security.ubuntu.com/ubuntu xenial-security InRelease
Atteint :4 http://fr.archive.ubuntu.com/ubuntu xenial-updates InRelease
Atteint :5 http://fr.archive.ubuntu.com/ubuntu xenial-backports InRelease
Réception de :6 http://ppa.launchpad.net/ethereum/ethereum/ubuntu xenial/main 13 86 Packages [2 828 B]
Atteint :7 https://esm.ubuntu.com/infra/ubuntu xenial-infra-security InRelease
Atteint :8 https://esm.ubuntu.com/infra/ubuntu xenial-infra-updates InRelease
Réception de :9 http://ppa.launchpad.net/ethereum/ethereum/ubuntu xenial/main Translation-en [828 B]
19,0 ko réceptionnés en 2s (9 016 o/s)
Lecture des listes de paquets... Fait
farah@farah-VirtualBox:~$ sudo apt-get install ethereum
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  abigen bootnode clef evm geth puppeth rlpdump
Les NOUVEAUX paquets suivants seront installés :
  abigen bootnode clef ethereum evm geth puppeth rlpdump
0 mis à jour, 8 nouvellement installés, 0 à enlever et 1 non mis à jour.
```

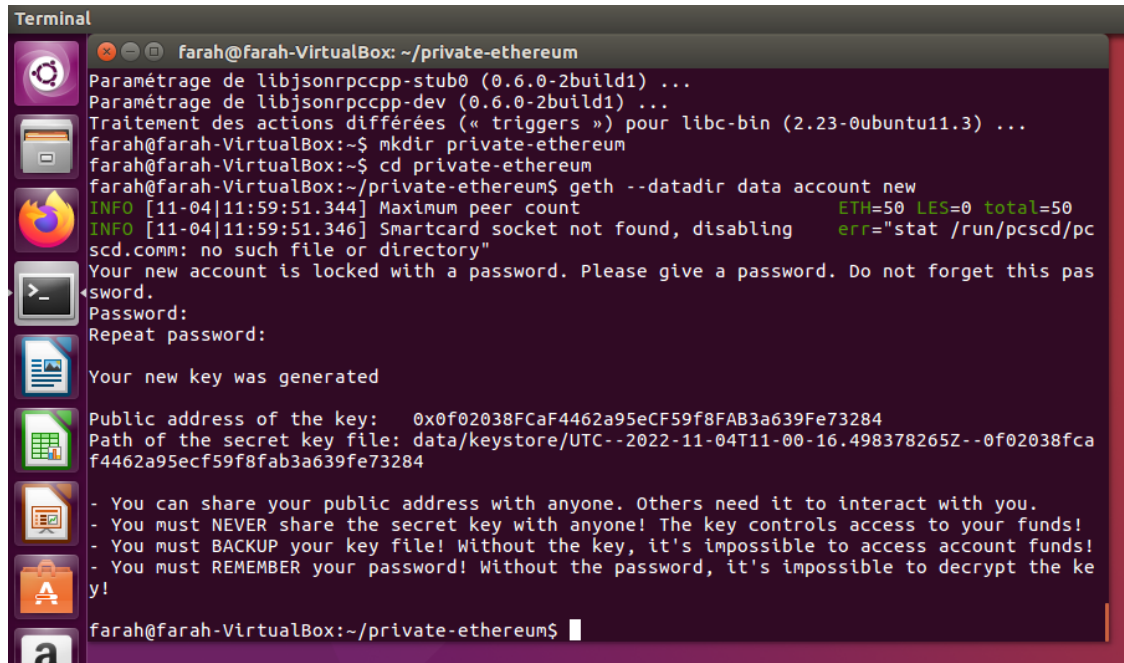


```
Terminal
farah@farah-VirtualBox: ~
ufarah@farah-VirtualBox:~$ sudo apt-get install ethereum
Réception de :1 http://fr.archive.ubuntu.com/ubuntu xenial/universe i386 libbargtable2-0 i3 86 12-1.1 [13,9 kB]
Réception de :2 http://fr.archive.ubuntu.com/ubuntu xenial/universe i386 libjsonrpcpp-com mon0 i386 0.6.0-2build1 [25,6 kB]
Réception de :3 http://fr.archive.ubuntu.com/ubuntu xenial/universe i386 libjsonrpcpp-cli ent0 i386 0.6.0-2build1 [20,4 kB]
Réception de :4 http://fr.archive.ubuntu.com/ubuntu xenial/universe i386 libjsonrpcpp-ser ver0 i386 0.6.0-2build1 [22,8 kB]
Réception de :5 http://fr.archive.ubuntu.com/ubuntu xenial/universe i386 libjsonrpcpp-stu b0 i386 0.6.0-2build1 [28,1 kB]
Réception de :6 http://fr.archive.ubuntu.com/ubuntu xenial/universe i386 libjsonrpcpp-dev i386 0.6.0-2build1 [86,1 kB]
197 ko réceptionnés en 7s (27,3 ko/s)
Sélection du paquet libbargtable2-0 précédemment désélectionné.
(Lecture de la base de données... 237878 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../libbargtable2-0_12-1.1_i386.deb ...
Dépaquetage de libbargtable2-0 (12-1.1) ...
Sélection du paquet libjsonrpcpp-common0 précédemment désélectionné.
Préparation du dépaquetage de .../libjsonrpcpp-common0_0.6.0-2build1_i386.deb ...
Dépaquetage de libjsonrpcpp-common0 (0.6.0-2build1) ...
Sélection du paquet libjsonrpcpp-client0 précédemment désélectionné.
Préparation du dépaquetage de .../libjsonrpcpp-client0_0.6.0-2build1_i386.deb ...
Dépaquetage de libjsonrpcpp-client0 (0.6.0-2build1) ...
Sélection du paquet libjsonrpcpp-server0 précédemment désélectionné.
Préparation du dépaquetage de .../libjsonrpcpp-server0_0.6.0-2build1_i386.deb ...
Dépaquetage de libjsonrpcpp-server0 (0.6.0-2build1) ...
Sélection du paquet libjsonrpcpp-stub0 précédemment désélectionné.
Préparation du dépaquetage de .../libjsonrpcpp-stub0_0.6.0-2build1_i386.deb ...
Dépaquetage de libjsonrpcpp-stub0 (0.6.0-2build1) ...
Sélection du paquet libjsonrpcpp-dev précédemment désélectionné.
Préparation du dépaquetage de .../libjsonrpcpp-dev_0.6.0-2build1_i386.deb ...
Dépaquetage de libjsonrpcpp-dev (0.6.0-2build1) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.23-0ubuntu11.3) ...
Paramétrage de libbargtable2-0 (12-1.1) ...
Paramétrage de libjsonrpcpp-common0 (0.6.0-2build1) ...
Paramétrage de libjsonrpcpp-client0 (0.6.0-2build1) ...
Paramétrage de libjsonrpcpp-server0 (0.6.0-2build1) ...
Paramétrage de libjsonrpcpp-stub0 (0.6.0-2build1) ...
Paramétrage de libjsonrpcpp-dev (0.6.0-2build1) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.23-0ubuntu11.3) ...
farah@farah-VirtualBox:~$
```

ETAPE 2 : CREATION DE COMPTES POUR LE RESEAU PRIVE ETHEREUM

Maintenant, nous créons 2 comptes :

- Le premier compte aura pour mot de passe : 94EsmeSudria



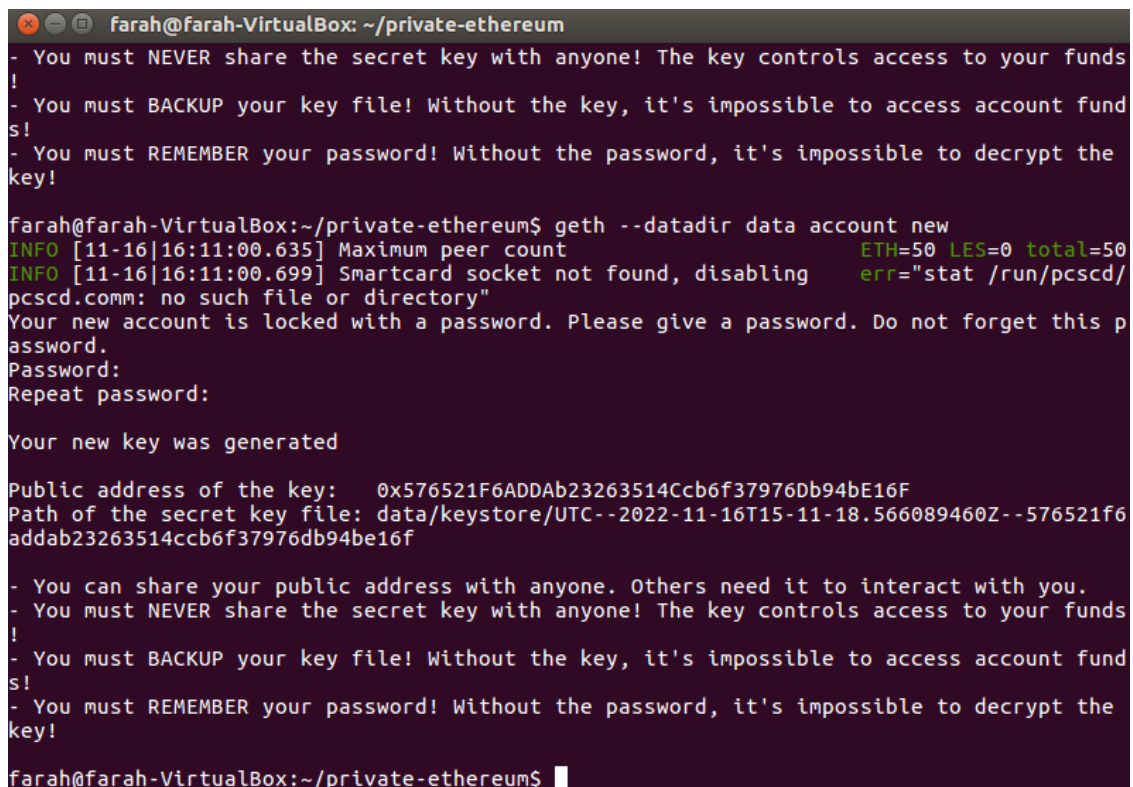
```
Terminal
farah@farah-VirtualBox: ~/private-ethereum
Paramétrage de libjsonrpcpp-stub0 (0.6.0-2build1) ...
Paramétrage de libjsonrpcpp-dev (0.6.0-2build1) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.23-0ubuntu11.3) ...
farah@farah-VirtualBox:~$ mkdir private-ethereum
farah@farah-VirtualBox:~$ cd private-ethereum
farah@farah-VirtualBox:~/private-ethereum$ geth --datadir data account new
INFO [11-04|11:59:51.344] Maximum peer count          ETH=50 LES=0 total=50
INFO [11-04|11:59:51.346] Smartcard socket not found, disabling  err="stat /run/pcscd/pcscd.comm: no such file or directory"
Your new account is locked with a password. Please give a password. Do not forget this password.
Password:
Repeat password:
Your new key was generated

Public address of the key: 0x0f02038FCaF4462a95eCF59f8FAB3a639Fe73284
Path of the secret key file: data/keystore/UTC--2022-11-04T11-00-16.498378265Z--0f02038fca4462a95ecf59f8fab3a639fe73284

- You can share your public address with anyone. Others need it to interact with you.
- You must NEVER share the secret key with anyone! The key controls access to your funds!
- You must BACKUP your key file! Without the key, it's impossible to access account funds!
- You must REMEMBER your password! Without the password, it's impossible to decrypt the key!

farah@farah-VirtualBox:~/private-ethereum$
```

- Le deuxième compte aura pour mot de passe : 94EsmeSudria1



```
farah@farah-VirtualBox: ~/private-ethereum
- You must NEVER share the secret key with anyone! The key controls access to your funds!
- You must BACKUP your key file! Without the key, it's impossible to access account funds!
- You must REMEMBER your password! Without the password, it's impossible to decrypt the key!

farah@farah-VirtualBox:~/private-ethereum$ geth --datadir data account new
INFO [11-16|16:11:00.635] Maximum peer count          ETH=50 LES=0 total=50
INFO [11-16|16:11:00.699] Smartcard socket not found, disabling  err="stat /run/pcscd/pcscd.comm: no such file or directory"
Your new account is locked with a password. Please give a password. Do not forget this password.
Password:
Repeat password:
Your new key was generated

Public address of the key: 0x576521F6ADDAB23263514Ccb6f37976Db94bE16F
Path of the secret key file: data/keystore/UTC--2022-11-16T15-11-18.566089460Z--576521f6addab23263514ccb6f37976db94be16f

- You can share your public address with anyone. Others need it to interact with you.
- You must NEVER share the secret key with anyone! The key controls access to your funds!
- You must BACKUP your key file! Without the key, it's impossible to access account funds!
- You must REMEMBER your password! Without the password, it's impossible to decrypt the key!

farah@farah-VirtualBox:~/private-ethereum$
```

La commande « *geth --datadir data account list* » va afficher la liste des comptes.

Ainsi, nous remarquons bien que les 2 comptes ont été créés « Account #0 » et « Account #1 » :

```
farah@farah-VirtualBox:~/private-ethereum2$ geth --datadir data account list
INFO [11-16|17:14:39.734] Maximum peer count           ETH=50 LES=0
total=50
INFO [11-16|17:14:39.735] Smartcard socket not found, disabling err="stat /run/pcscd/pcscd.comm: no such file or directory"
WARN [11-16|17:14:39.748] Sanitizing cache to Go's GC limits provided=1024
updated=333
INFO [11-16|17:14:39.749] Set global gas cap           cap=50,000,00
0
Account #0: {63f82cb6ae216c3a9e397db8b26a3e7c00801b27} keystore:///home/farah/private-ethereum2/data/keystore/UTC--2022-11-16T16-13-44.361191292Z--63f82cb6ae216c3a9e397db8b26a3e7c00801b27
Account #1: {4a05225b93b3090be548dfc567dc7e93d42c1d31} keystore:///home/farah/private-ethereum2/data/keystore/UTC--2022-11-16T16-14-14.643857938Z--4a05225b93b3090be548dfc567dc7e93d42c1d31
farah@farah-VirtualBox:~/private-ethereum2$
```

ETAPE 3 : CREATION DU GENESIS FILE

Maintenant, nous allons créer et remplir notre fichier genesis.json à l'identique avec celui qu'Ishika a créé sur le serveur :

```
farah@farah-VirtualBox: ~/private-ethereum
GNU nano 2.5.3          Fichier : genesis.json          Modifié
{
"config": {
"chainId": 12345678,
"homesteadBlock": 0,
"eip150Block": 0,
"eip155Block": 0,
"eip158Block": 0,
"byzantiumBlock": 0,
"constantinopleBlock": 0,
"petersburgBlock": 0,
"ethash": {}
},
"difficulty": "1",
"gasLimit": "8000000",
"alloc": {
  "8bf6f8c459db71707bb60fd5ea86bd2895c60e00": { "balance": "3000000000000000000000"},
  "5c7ec9bd1bbebe8af2aed7ebb7484f7e42d329d": { "balance": "4000000000000000000000"}
}
}
```

^G Aide ^O Écrire ^W Chercher ^K Couper ^J Justifier ^C Pos. cur.
^X Quitter ^R Lire fich. ^\ Remplacer ^U Coller ^T Orthograp. ^_ Aller lig.

Détaillons ensemble les paramètres de ce document :

- « *config* » : Il indique que les paramètres suivants concernent la configuration de notre blockchain
- « *chainID* » : Il permet de donner un identifiant à notre blockchain en l'occurrence ici 12345678 et est utilisé pour la protection contre la relecture. Il s'agit ici d'une valeur unique pour notre blockchain privée.
- « *homesteadBlock* » : Homestead est la deuxième version de la plateforme Ethereum. Notre blockchain ne subira pas le passage à Homestead donc nous laissons ce paramètre à 0.
- « *eip150Block* » : Il s'agit de modifications de la version homestead dans le protocole. Notre blockchain ne sera pas dure pour ces changements donc nous laissons ce paramètre à 0.
- « *eip155Block* » : Il s'agit de modifications de la version homestead dans le protocole. Notre blockchain ne sera pas dure pour ces changements donc nous laissons ce paramètre à 0.
- « *eip158Block* » : Il s'agit de modifications de la version homestead dans le protocole. Notre blockchain ne sera pas dure pour ces changements donc nous laissons ce paramètre à 0.
- « *byzantiumBlock* » : Ce paramètre indique que Byzantium est activé dans notre réseau privé.
- « *constantinopleBlock* » : Ce paramètre indique que Constantinople est activé dans notre réseau privé.
- « *petersburgBlock* » : Ce paramètre indique que Petersburg est activé dans notre réseau privé.
- « *ethash* » : Il s'agit de l'algorithme qui rend possible le fonctionnement du minage.
- « *difficulty* » : Il s'agit de la difficulté de notre blockchain.
- « *gasLimit* » : Cela correspond à la valeur de dépense de gaz par block. Nous mettons une valeur élevée pour ne pas être bloqués pendant la phase de test.
- « *alloc* » : Il permet de définir une liste de wallets pré-remplis. Il s'agit d'une fonctionnalité spécifique à Ethereum. Nous y mettons les adresses des 2 comptes créés par Ishika sur le serveur.

L'instanciation est bien réussie car nous obtenons le résultat suivant après le *geth init* :

```
farah@farah-VirtualBox: ~/private-ethereum
farah@farah-VirtualBox:~/private-ethereum$ nano genesis.json
farah@farah-VirtualBox:~/private-ethereum$ geth init --datadir data genesis.json
INFO [11-16|14:35:11.890] Maximum peer count           ETH=50 LES=0 total=50
INFO [11-16|14:35:11.897] Smartcard socket not found, disabling err="stat /run/pcscd/pc
scd.comm: no such file or directory"
WARN [11-16|14:35:11.908] Sanitizing cache to Go's GC limits provided=1024 updated=3
33
INFO [11-16|14:35:11.912] Set global gas cap           cap=50,000,000
INFO [11-16|14:35:11.919] Allocated cache and file handles database=/home/farah/pr
ivate-ethereum/data/geth/chaindata cache=16.00MiB handles=16
INFO [11-16|14:35:11.962] Opened ancient database      database=/home/farah/pr
ivate-ethereum/data/geth/chaindata/ancient/chain readonly=false
INFO [11-16|14:35:11.964] Writing custom genesis block
INFO [11-16|14:35:11.978] Persisted trie from memory database nodes=3 size=409.00B ti
me="301.129µs" gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [11-16|14:35:11.980] Successfully wrote genesis state database=chaindata hash
=c3e65d..a2fc9c
INFO [11-16|14:35:11.980] Allocated cache and file handles database=/home/farah/pr
ivate-ethereum/data/geth/lightchaindata cache=16.00MiB handles=16
INFO [11-16|14:35:12.023] Opened ancient database      database=/home/farah/pr
ivate-ethereum/data/geth/lightchaindata/ancient/chain readonly=false
INFO [11-16|14:35:12.024] Writing custom genesis block
INFO [11-16|14:35:12.026] Persisted trie from memory database nodes=3 size=409.00B ti
me="141.467µs" gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [11-16|14:35:12.027] Successfully wrote genesis state database=lightchaindata
hash=c3e65d..a2fc9c
farah@farah-VirtualBox:~/private-ethereum$
```

ETAPE 4 : CONFIGURATION DU BOOTNODE

Initialisons notre bootnode et récupérons en sortie notre enode :

```
farah@farah-VirtualBox:~/private-ethereum2$ bootnode
Fatal: Use -nodekey or -nodekeyhex to specify a private key
farah@farah-VirtualBox:~/private-ethereum2$ bootnode --genkey=boot.key
farah@farah-VirtualBox:~/private-ethereum2$ bootnode --nodekey=boot.key
enode://49b99f7594c3deca1632b1ac8ebe873924c68d46b1ec1663c6773848206c3be996d57230
e53f5480b776bd3a50541d14a4c0d81affe5cc8a6b13bf7ae8d5c58f@127.0.0.1:0?discport=30
1
Note: you're using cmd/bootnode, a developer tool.
We recommend using a regular node as bootstrap node for production deployments.
INFO [11-16|17:16:47.284] New local node record          seq=1,668,615
,407,242 id=abfada5b185e3603 ip=<nil> udp=0 tcp=0
```

Vérifions maintenant que nous possédons la même adresse comme identifiant :

```
farah@farah-VirtualBox:~/private-ethereum2$ bootnode --nodekey=boot.key --writea
ddress
49b99f7594c3deca1632b1ac8ebe873924c68d46b1ec1663c6773848206c3be996d57230e53f5480
b776bd3a50541d14a4c0d81affe5cc8a6b13bf7ae8d5c58f
farah@farah-VirtualBox:~/private-ethereum2$
```


ETAPE 5 : CONFIGURE NOTRE ETHEREUM ET COMMENCER A MINER

Cette étape a bien été faite par Ishika sur le server afin de configurer notre blockchain privée Ethereum pour que par la suite, dans l'étape 6, nous puissions nous connecter et récupérer les nœuds qu'elle a minés.

Par ailleurs, nous devons affecter notre adresse IP à notre nœud bootnode précédemment instancié :

```
farah@farah-VirtualBox: ~/private-ethereum2
vate-ethereum2/data/geth/chaindata cache=16.00MiB handles=16
INFO [11-16|17:34:50.883] Opened ancient database           database=/home/farah/pri
vate-ethereum2/data/geth/chaindata/ancient/chain readonly=false
INFO [11-16|17:34:50.884] Writing custom genesis block
INFO [11-16|17:34:50.889] Persisted trie from memory database   nodes=3 size=409.00B tim
e="193.742µs" gcnodes=0 gcsiz=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [11-16|17:34:50.891] Successfully wrote genesis state     database=chaindata hash=
a054c8..1677ab
INFO [11-16|17:34:50.891] Allocated cache and file handles     database=/home/farah/pri
vate-ethereum2/data/geth/lightchaindata cache=16.00MiB handles=16
INFO [11-16|17:34:50.927] Opened ancient database             database=/home/farah/pri
vate-ethereum2/data/geth/lightchaindata/ancient/chain readonly=false
INFO [11-16|17:34:50.927] Writing custom genesis block
INFO [11-16|17:34:50.929] Persisted trie from memory database   nodes=3 size=409.00B tim
e="117.659µs" gcnodes=0 gcsiz=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [11-16|17:34:50.929] Successfully wrote genesis state     database=lightchaindata
hash=a054c8..1677ab
farah@farah-VirtualBox:~/private-ethereum2$ bootnode --nodekey=boot.key --addr 192.168.66.1
28:30301
enode://49b99f7594c3deca1632b1ac8ebe873924c68d46b1ec1663c6773848206c3be996d57230e53f5480b77
6bd3a50541d14a4c0d81affe5cc8a6b13bf7ae8d5c58f@192.168.66.128:0?discport=30301
Note: you're using cmd/bootnode, a developer tool.
We recommend using a regular node as bootstrap node for production deployments.
INFO [11-16|17:36:04.245] New local node record               seq=1,668,616,564,237 id
=abfada5b185e3603 ip=<nil> udp=0 tcp=0
```


ETAPE 6 : DEMARRAGE DES NŒUDS SUR NOS ORDINATEURS

Maintenant, connectons-nous à notre blockchain privée. Nous remplacerons par notre chainID préalablement défini dans le fichier genesis.json et nous prendrons l'enode généré par le serveur.

Sur la capture ci-dessous, nous pouvons voir que la connexion a réussie puisque nous sommes synchronisés avec le serveur et que nous importons ses paquets :

```

farah@farah-VirtualBox: ~/private-ethereum2
INFO [11-16|17:38:41.652] Looking for peers                    peercount=1 tried=123 st
atic=0
INFO [11-16|17:38:42.753] Generating ethash verification cache epoch=0 percentage=40 el
apsed=3.002s
INFO [11-16|17:38:45.754] Generating ethash verification cache epoch=0 percentage=82 el
apsed=6.004s
WARN [11-16|17:38:46.955] Snapshot extension registration failed peer=020ad183 err="peer
connected on snap without compatible eth support"
INFO [11-16|17:38:46.964] Generated ethash verification cache epoch=0 elapsed=7.213s
INFO [11-16|17:38:47.231] Imported new block headers count=50 elapsed=7.492s
number=50 hash=420aa3..3eb810 age=1w3d22h
INFO [11-16|17:38:47.237] Downloader queue stats receiptTasks=0 blockTask
s=0 itemSize=549.03B throttle=8192
INFO [11-16|17:38:47.485] Imported new chain segment blocks=50 txs=0 mgas=0.0
00 elapsed=246.086ms mgasps=0.000 number=50 hash=420aa3..3eb810 age=1w3d22h dirty=21.00KiB
INFO [11-16|17:38:48.850] Imported new block headers count=2 elapsed=18.638m
s number=52 hash=84f992..be4616 age=1w3d22h
INFO [11-16|17:38:48.865] Imported new chain segment blocks=2 txs=0 mgas=0.0
00 elapsed=13.912ms mgasps=0.000 number=52 hash=84f992..be4616 age=1w3d22h dirty=21.84KiB
INFO [11-16|17:38:48.866] Snap sync complete, auto disabling

```

Vérifions que les 2 premiers blocks sont identiques à ceux sur le serveur.

Voici les blocks côté serveur :

[illegible]

Pour le premier block côté client nous obtenons le même hash que côté serveur :

[illegible]

De même, pour le deuxième block :

[illegible]

Nous voyons bien que les blocks sont identiques.

CONCLUSION

Nous avons donc réussi l'objectif de ce TP : nous avons créé, configuré et synchronisé notre blockchain.

BONUS

Aujourd'hui, resynchronisons-nous au serveur. L'énodé est resté le même depuis hier :

```
farah@farah-VirtualBox: ~/private-ethereum2
farah@farah-VirtualBox:~$ cd private-ethereum2
farah@farah-VirtualBox:~/private-ethereum2$ geth --networkid 12345678 --datadir data --bootnode
des enode://f0c175d4ddf4432e3e915e4d5d1137f5073853a3d5ceeffe627a781a5dfff1dc4f97412ba71dc1578e
d29e2ee77aaa629c0eb33d65d0a970fabaa92c3691d19d8@64.225.66.111:30303 console
INFO [11-16|18:20:38.119] Maximum peer count                      ETH=50 LES=0 total=50
INFO [11-16|18:20:38.123] Smartcard socket not found, disabling  err="stat /run/pcscd/pcscd
.comm: no such file or directory"
WARN [11-16|18:20:38.136] Sanitizing cache to Go's GC limits      provided=1024 updated=333
INFO [11-16|18:20:38.137] Set global gas cap                     cap=50,000,000
INFO [11-16|18:20:38.145] Allocated trie memory caches           clean=49.00MiB dirty=83.00
MiB
INFO [11-16|18:20:38.151] Allocated cache and file handles       database=/home/farah/priva
te-ethereum2/data/geth/chaindata cache=165.00MiB handles=524,288
INFO [11-16|18:20:38.221] Opened ancient database                database=/home/farah/priva
te-ethereum2/data/geth/chaindata/ancient/chain readonly=false
INFO [11-16|18:20:38.242] -----
INFO [11-16|18:20:38.243] Chain ID: 12345678 (unknown)
INFO [11-16|18:20:38.243] Consensus: Ethash (proof-of-work)
INFO [11-16|18:20:38.243] Pre-Merge hard forks:
INFO [11-16|18:20:38.243]   - Homestead: 0 (https://github.com/ethere
um/execution-specs/blob/master/network-upgrades/mainnet-upgrades/homestead.md)
INFO [11-16|18:20:38.243]   - Tangerine Whistle (EIP 150): 0 (https://github.com/ethere
um/execution-specs/blob/master/network-upgrades/mainnet-upgrades/tangerine-whistle.md)
INFO [11-16|18:20:38.243]   - Spurious Dragon/1 (EIP 155): 0 (https://github.com/ethere
um/execution-specs/blob/master/network-upgrades/mainnet-upgrades/spurious-dragon.md)
INFO [11-16|18:20:38.243]   - Spurious Dragon/2 (EIP 158): 0 (https://github.com/ethere
um/execution-specs/blob/master/network-upgrades/mainnet-upgrades/spurious-dragon.md)
INFO [11-16|18:20:38.244]   - Byzantium: 0 (https://github.com/ethere
um/execution-specs/blob/master/network-upgrades/mainnet-upgrades/byzantium.md)
INFO [11-16|18:20:38.244]   - Constantinople: 0 (https://github.com/ethere
um/execution-specs/blob/master/network-upgrades/mainnet-upgrades/constantinople.md)
```

Lançons la commande « *net.listening* » :

```
To exit, press ctrl-d or type exit
> net.listening
true
```

Maintenant récupérons les blocks pour vérifier qu'ils sont bien identiques à ceux d'hier, cela signifiera que nous sommes bien connectés au serveur :

[illegible][illegible]

Les hash sont les bons. Nous sommes donc bien connectés au serveur et récupérons les bons blocks.

Ishika a lancé un nouveau minage de block. Nous l'avons récupéré et nous avons bien le même :

[illegible]

Nous allons maintenant créer un compte sur notre blockchain. Les adresses renvoyées correspondent aux adresses des comptes que nous avons créés au début du TP :

```
> eth.accounts
["0x63f82cb6ae216c3a9e397db8b26a3e7c00801b27", "0x4a05225b93b3090be548dfc567dc7e93d42c1d31"]
```

Maintenant, nous allons nous envoyer une transaction avec les commandes :

- Sur le serveur donc sur le PC d'Ishika :
 - o Ishika va miner un nouveau block en partant du dernier block dans lequel se trouvera la transaction qu'elle nous envoie
 - o « *personal.unlockAccount(eth.accounts[0])* »
 - o « *eth.sendTransaction({to : « chaîne de caractère correspondant au compte destinataire », from : eth.accounts[0], value : 25000})* »
- Sur le client donc sur mon PC :
 - o « *eth.getBalance(« chaîne de caractère correspondant au compte destinataire »)* »

Faisons un *getBalance* avant qu'Ishika nous envoie une transaction pour comparer l'avant après.

Voici avant :

```
> eth.getBalance("0x63f82cb6ae216c3a9e397db8b26a3e7c00801b27")
0
```

Après la commande, nous voyons bien que le résultat n'est plus de 0 mais du montant de la transaction.