Running an Ethereum Network Consisting of a Two nodes

We will setup two nodes on the same machine, creating a peer-to-peer network on our localhost. In addition to the two nodes, a bootnode (discovery service) will also be setup.

Follow the **Steps 1 - 6** from **Running an Ethereum Network Consisting of a Single node** section and create two node directories, which will result in a folder structure like this:

```
qwerty$ tree -L 2
.
├── accounts.txt
├── boot.key
├── genesis.json
├── node1
│   ├── geth
│   ├── keystore
│   └── password.txt
└── node2
    ├── geth
    ├── keystore
    └── password.txt
```

All except the **boot.key** file 😉

Now let's go setup a boot node.

Step 1: Create a bootnode

A bootnode only purpose is to helping nodes discovering each other's (remember, the Ethereum blockchain is a peer-to-peer network). Nodes could have dynamic IP, being turned off, and on again. The bootnode is usually ran on a static IP and thus acts like a pub where nodes know they will find their mates.

Initialize the bootnode:

```
> bootnode -genkey boot.key
```

This creates a value called the **enode** uniquely identifying your bootnode (more on this soon) and we store this enode in the boot.key file.

Step 2: Start the bootnode service

```
> bootnode -nodekey boot.key -verbosity 9 -addr :30310
```

This will display the enode address of your boot node on the console, something like this:

```
enode://3ec4fef2d726c2c01f16f0a0030f15dd5a81e274067af2b2157cafbf76aa79fa9c
0be52c6664e80cc5b08162ede53279bd70ee10d024fe86613b0b09e1106c40@[::]:30310
```

Note: In case the enode address is not shown use the below command to get the enode address.

bootnode -nodekeyhex
ac54865a750cf6e357ba23cf74ec1246902ed6222f07f38ee81fe31cace89e27 -writeaddress

Note: Your node key can be found with in the **boot.key** file.

Step 3: Starting your nodes

Big time! Finally (but usually here the troubles arrive too).

Everything in one huge command! I am going to cover some options but please do your homework and refer to the doc.

Starting node 1:

```
geth --datadir node1/ --syncmode 'full' --port 30311 --rpc --rpcaddr
'localhost' --rpcport 8545 --rpcapi
'personal,db,eth,net,web3,txpool,miner' --bootnodes
'enode://9ef15816d8bb9566c0324d7373d961687d4406f10c0556be6919c10239a52cc20
212028ff16f141073fd50dbdf1177a8aaaba4c4a672d15289a2b49c1ec8050b@127.0.0.1:
30310' --ipcpath '~/.ethereum/geth.ipc' --rpccorsdomain '*' --networkid
1515 --unlock 'account address' --gasprice '1' --password (path of
password file)
```

Starting node 2:

```
geth --datadir node2/ --syncmode 'full' --port 30312 --rpc --rpcaddr
'localhost' --rpcport 8502 --rpcapi
'personal,db,eth,net,web3,txpool,miner' --bootnodes
'enode://9ef15816d8bb9566c0324d7373d961687d4406f10c0556be6919c10239a52cc20
212028ff16f141073fd50dbdf1177a8aaaba4c4a672d15289a2b49c1ec8050b@127.0.0.1:
30310' --ipcpath '~/.ethereum/geth.ipc' --rpccorsdomain '*' --networkid
1515 --gasprice '1' --unlock 'account adress' --password (path of the
password file)
```

Let's see what these new parameters are:

- **--syncmode 'full'** helps preventing the error Discarded Bad Propagated Block.
- **--port 30311** is the enode port for node1 and must be different from the bootnode port (that is 30310 if you followed my command) because we are on a localhost. On a real network (one node per machine), use the same port.
- **--rpcapi** allows the listed modules to be used over RPC calls (see section 3.3 for an example). See the Geth Management APIs for more info. Be mindful about hacks as everyone can call your RPC methods if no firewall is protecting your node.
- **--bootnodes** tells your node at what address to find your bootnode. Replace [::] with the bootnode IP. No domain name is allowed! Only IPs. Check enode URL format.
- **--networkId** as defined in the **genesis.json** file. Please use the same id!
- **--gasprice '1'** I don't like to pay on my own network 😉 be careful with gas price. If your transactions are not being broadcasted to the network but only the node receiving the transactions is processing them, this means you sent a transaction with a gas price that is not accepted (too low) by the other nodes on the network. No error will be return. If you have two nodes, only one will be processing the transactions. This is sneaky and reduces your network throughput by a factor 2.
- **--unlock --password --mine** tell the node to unlock this account, with the password in that file and to start mining.

Now let's do this on different computers

Create a hotspot on one of the PC using the following command:

```
sudo service network-manager restart

nmcli device wifi hotspot con-name my-hotspot ssid my-hotspot band
bg password 12345678
```

Now let's follow the same steps as above and do change the IP from local host to target peer IP.

Note: Trusted nodes are used as pre-configured connections which are always allowed to connect, even above the peer limit.