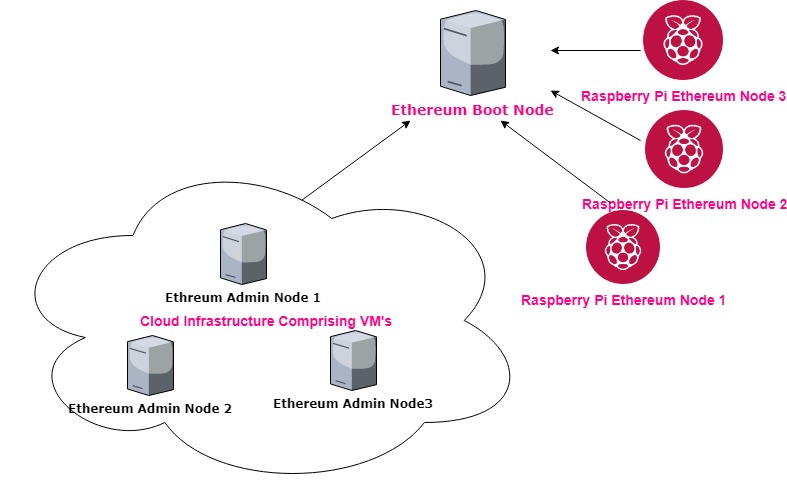
**VM Ware Setup for Ethereum:**



**Overall Setup for Ethereum Network is as Above excluding the middleware which will interact with each Node which will be explained in a later wiki file.**

**Step 1: Set Up of Ethereum Boot Node:**

Ethereum Boot Node is needed to setup for all the nodes in the Ethereum Network to undergo Node discovery automatically without having to automatically add each and every Node to other Node using its individual enode Address.

Reason for doing the boot Node Approach:

1. Automatised Node Discovery
2. Suitable for Scalable Ethereum Network
3. Facilitate easier communication (Ping & Pong) between Nodes

**How to Set Up Ethereum Boot Node?**

1. Choose a VM to configure as Boot Node:
2. Install Geth on the VM:

**sudo add-apt-repository -y ppa:ethereum/Ethereum**

**sudo apt-get update**

**sudo apt-get install Ethereum**

1. Then make directory of any desired place in VM
2. Generate the boot key which creates the enode address fopr the other Nodes to Communicate

**bootnode -genkey boot.key**

1. This creates a file **boot.key** which contains the HEX representation of the enode address
2. To get the representation of the enode address:

**Bootnode --nodekeyhex {Open the File Boot.key and get the Hex Representation} – writeaddress**

1. This generates the enode address

**--------------------------------------------------------------------------------------------**

**Eg:** To show the steps

1. **bootnode -genkey boot.key**

**{837c6e669ceabc7dd7efeeae83cfd4c8a2cc1de592ffb6d4bdbeec021586e523}**

1. Then get the ASCII representation:

**bootnode --nodekeyhex 837c6e669ceabc7dd7efeeae83cfd4c8a2cc1de592ffb6d4bdbeec021586e523 -writeaddress /// To Get ENODE ADDRESS**

1. **Here Enode Address which we obtained is:** 780636d36bd6d57b73cb8b0b5c659079df19e8195b889a5b2258dcdc1ada0ffaf24b82446279603b9d1fd65852375ef102fc2200e51bcd88c498e21bd3e6b291
2. This enode address is passed when starting each Node for automatic discovery along with its port and IP
3. Final Step will be starting of the Boot Node:

Here we start with the highest level of verbosity for log purpose and as well on port: 50006 which can be changed as choice.

**nohup bootnode --nodekey=boot.key -verbosity 9 -addr :50006 &**

**Step2:**

**Setup the Remaining Nodes to Produce the Private Network:**

1. First Install Geth in all the VMs which will be participating in the Network
2. **Remember**: For Raspberry Pi, we don’t install geth but just we place the binaries as listed in the step 2 of the previous part
3. Then we generate the Ethereum Address of each of the Nodes before hand as we need them to specify in the genesis.json to fund them with sufficient ethers in Genesis.json
4. To get the Ethereum Address for each VM, perform this step and Note it somewhere:

Here we specify the datadir of each of the Node to store the Keystore:

Datadir is specified here otherwise the default user profile location is chosen

Geth –datadir= { } account new

Eg:

1. **geth --datadir=/home/renault/Documents/trialether/node2 account new**
2. Then it prompts a password to secure the Keystore which has to be provided to secure the keystore in the datadir
3. Generated Address will be of type:

Eg:

**93f239df6bf0e68d2d614ec2abd4b9f53c887e9f**

1. Provide the password and Note the Address generated as well as the Password which will be used to access the Node later
2. Repeat this step for the remaining Nodes to get their corresponding addresses.

**Step 3:**

**Formation of Genesis.json**

**Genesis.json is needed to form the Genesis Block of the blockchain which will help all the Nodes to communicate on the same network id.**

**Standard genesis.json to be setup which include the latest version of Ethereum:**

**Although this Genesis.json can be followed to initate each Node**

**RED Highlighted can be changed for new chainId, although this should not be specified or conflict any Ethereum TestNet id or Main Net**

**Difficulty can be changed as well ,here we set to low value**

{

"config": {

**"chainId": 26,**

"homesteadBlock": 0,

"eip155Block": 0,

"eip158Block": 0,

"ByzantiumBlock": 0

},

"nonce": "0x0000000000000033",

"timestamp": "0x0",

"parentHash": "0x0000000000000000000000000000000000000000000000000000000000000000",

"gasLimit": "0x8000000",

**"difficulty": "0x10",**

"mixhash": "0x0000000000000000000000000000000000000000000000000000000000000000",

"alloc": {

**"0xaa43515b48c36a51f158cc8b941900b8aa6fb885": {**

"balance": "99999999999999999999999999999999"

},

**"0x22a8c699ff1081c31e32ef0ec9266694e415fcd5": {**

"balance": "99999999999999999999999999999999"

},

**"0x420a6e75d4c68e5123de6c998a753e66f81db3ed": {**

"balance": "99999999999999999999999999999999"

},

**"0x9d8a6e7068faf675d933a872195b4ab44b3a1793": {**

"balance": "99999999999999999999999999999999"

},

**"0xe7c2f755c8230c5e8296c3f435d1d25e32c8c7b6": {**

"balance": "99999999999999999999999999999999"

},

**"0x881878dcb53fef876f0bfd3f5548862762bef4b6": {**

"balance": "99999999999999999999999999999999"

}

}

}

**Green Highlight correspond to the Address of each VM Nodes which will be participating in the Network, which you have to replace**

Also, you can include the address of Raspberry Pi Ethereum Nodes which can be generated as in similar manner to VM

**Go the Path of Binary in the Raspberry Pi and then execute the following**

**./geth --datadir=/home/renault/Documents/trialether/node2 account new**

**Step 4: Initialise each Node both VM as well as Raspberry Pi with same Genesis.json**

Genesis.json is the blueprint of the Network and all Nodes need to possess the same Genesis.json

1. Initalise each Node with Genesis.json in a particular directory where you want to store the Blockchain Data

**geth --datadir=/home/renault/Documents/trialether/node1 init genesis.json**

1. **Do the same for all the VM Nodes as well as Raspberry Nodes**
2. **This step actually generates the first block of the blockchain data needed for syncing amon all the Nodes on the Network**

**Step 5: Start the Node on VM as well as Raspberry Pi pointing to the bootnode**

**In VM ware: we start the Geth Nodes with mining of threads 2**

**Here we start**

1. **In the back ground**
2. **Specify the same datadir where we initalised genesis.json**
3. **Same Network ID specified in genesis.json**
4. **Enable RPC Calls**
5. **Enable couple of other API’s**
6. **Enable RPC communication on port 2000 (can be changed)**
7. **Point to boot node address which we created in the first step running at enode address, IP, PORT**
8. **Accept any RP Connection from any IP**
9. **Unlock the Account od the Ethereum Node for the Node to start**
10. **Here we create a password file named “pass.txt” which contain the password of the Ethereum Node which was specified while creating the Ethereum Node address**
11. **Enable Mining**
12. **Enable Mining with threads 2 for VM and for Raspberry Pi with thread 1**

**On VM:**

**nohup geth --datadir=/home/lab/opt/blkchain/ether --networkid 26 --port 2001 --rpc --rpcapi web3,eth,personal,miner,net,txpool,admin,debug,miner --rpcport 2000 -bootnodes 'enode://afa26826ebd14f9ef631085aa85b3f927ade7f58d7fea246834cc9e9e8955fede13341596d31d33e5dce11f4a66e857a5dbb7f1e086d3107942787fe6d159bbc@10.214.222.244:50006' --rpccorsdomain "\*" --rpcaddr "0.0.0.0" --unlock '0x22a8c699ff1081c31e32ef0ec9266694e415fcd5' --password 'pass.txt' --mine --miner.threads 2 &**

**On Raspberry Pi:**

**nohup ./../../soft/geth-linux-arm64-1.8.27-4bcc0a37/geth --datadir=/home/renault/Documents/blkchain/ether --networkid 26 --port 2001 --rpc --rpcapi web3,eth,personal,miner,net,txpool,admin,debug,miner --rpcport 2000 -bootnodes 'enode://afa26826ebd14f9ef631085aa85b3f927ade7f58d7fea246834cc9e9e8955fede13341596d31d33e5dce11f4a66e857a5dbb7f1e086d3107942787fe6d159bbc@10.214.222.244:50006' --rpccorsdomain "\*" --rpcaddr "0.0.0.0" --unlock '0x420a6e75d4c68e5123de6c998a753e66f81db3ed' --password 'pass.txt' --mine --miner.threads 1 &**

**Step 6: Checklist:**

**If all of them communicate well can be checked with the following**

First Attach IPC Console on any of the Node whether in VM or Raspberry Pin asl all are peers and there is no Admin or Peer Node distinction here

**Attach the console:**

**This geth.ipc is found the directory of the Blockchain directory as specified while starting the Node**

1. **geth attach ipc:geth.ipc**
2. Then you’ll get the console
3. Then perform: **admin.peers** which should list you all the peers of the VM or Raspberry Pi Nodes which proves that you are connected to all the peers

This completes the basic setup of the Network.