

Proof of Authority Development Chain

For this assignment, you will take on the role of a new developer at a small bank.

Your mission, should you choose to accept it, will be to set up a testnet blockchain for your organization.

To do this, you will create and submit four deliverables:

- Set up your custom testnet blockchain.
- Send a test transaction.
- Create a repository.
- Write instructions on how to use the chain for the rest of your team.

Background

You have just landed a new job at ZBank, a small, innovative bank that is interested in exploring what blockchain technology can do for them and their customers.

Your first project at the company is to set up a private testnet that you and your team of developers can use to explore potentials for blockchain at ZBank.

You have decided on setting up a testnet because:

There is no real money involved, which will give your team of developers the freedom to experiment.

Testnets allows for offline development.

In order to set up a testnet, you will need to use the following skills/tools we learned in class:

- Puppeth, to generate your genesis block.
- Geth, a command-line tool, to create keys, initialize nodes, and connect the nodes together.
- The Clique Proof of Authority algorithm.

Tokens inherently have no value here, so we will provide pre-configured accounts and nodes for easy setup.

After creating the custom development chain, create documentation for others on how to start it using the pre-configured nodes and accounts. You can name the network anything you want, have fun with it!

Be sure to include any preliminary setup information, such as installing dependencies and environment configuration.

Instructions

Setup the custom out-of-the-box blockchain

- Create a new project directory for your new network. Call it whatever you want!
- Create a "Screenshots" folder inside of the project directory.

This directory was created as Blockchain Tools and contains all the needed commands to create and run the network.

- Create accounts for two (or more) nodes for the network with a separate `datadir` for each using `geth`.

Two nodes were created using `geth` and were called `node1` and `node2`. They

- Run `puppeth`, name your network, and select the option to configure a new genesis block.
- Choose the `Clique (Proof of Authority)` consensus algorithm.

This is what was done for the proof of authority development chain running the `puppeth` command.

```
MINGW64/c/Users/Anand/Blockchain-Tools
> zach

Sweet, you can set this via --network=zach next time!

INFO [09-04|15:34:16.768] Administering Ethereum network      name=zach
WARN [09-04|15:34:17.054] No previous configurations found      path=C:\\User
s\\Anand\\.puppeth\\zach

What would you like to do? (default = stats)
1. Show network stats
2. Configure new genesis
3. Track new remote server
4. Deploy network components
> 2

What would you like to do? (default = create)
1. Create new genesis from scratch
2. Import already existing genesis
> 1

Which consensus engine to use? (default = clique)
1. Ethash - proof-of-work
2. Clique - proof-of-authority
> 2

How many seconds should blocks take? (default = 15)
> 5

Which accounts are allowed to seal? (mandatory at least one)
> 0x5A4eD3836076A25687847286d18d39c89c0ff060
> 0x

Which accounts should be pre-funded? (advisable at least one)
> 0x5A4eD3836076A25687847286d18d39c89c0ff060
> 0x

Should the precompile-addresses (0x1 .. 0xff) be pre-funded with 1 wei? (advisable yes)
> no

Specify your chain/network ID if you want an explicit one (default = random)
> 333
INFO [09-04|15:42:55.336] Configured new genesis block

What would you like to do? (default = stats)
1. Show network stats
2. Manage existing genesis
3. Track new remote server
4. Deploy network components
> |
```

- Paste both account addresses from the first step one at a time into the list of accounts to seal.
- Paste them again in the list of accounts to pre-fund. There are no block rewards in PoA, so you'll need to pre-fund.
- You can choose no for pre-funding the pre-compiled accounts (0x1 .. 0xff) with wei. This keeps the genesis cleaner.
- Complete the rest of the prompts, and when you are back at the main menu, choose the "Manage existing genesis" option.

Further steps on the puppet command to create the network and manage the existing genesis.

```
MINGW64~/c/Users/Anand/Blockchain-Tools
> zach

Sweet, you can set this via --network=zach next time!

INFO [09-04|15:34:16.768] Administering Ethereum network      name=zach
WARN [09-04|15:34:17.054] No previous configurations found      path=C:\User
s\Anand\puppeth\zach

What would you like to do? (default = stats)
1. Show network stats
2. Configure new genesis
3. Track new remote server
4. Deploy network components
> 2

What would you like to do? (default = create)
1. Create new genesis from scratch
2. Import already existing genesis
> 1

Which consensus engine to use? (default = clique)
1. Ethash - proof-of-work
2. Clique - proof-of-authority
> 2

How many seconds should blocks take? (default = 15)
> 5

Which accounts are allowed to seal? (mandatory at least one)
> 0x5A4eD3836076A25687847286d18d39c89c0ff060
> 0x

Which accounts should be pre-funded? (advisable at least one)
> 0x5A4eD3836076A25687847286d18d39c89c0ff060
> 0x

Should the precompile-addresses (0x1 .. 0xff) be pre-funded with 1 wei? (advisab
le yes)
> no

Specify your chain/network ID if you want an explicit one (default = random)
> 333
INFO [09-04|15:42:55.336] Configured new genesis block

What would you like to do? (default = stats)
1. Show network stats
2. Manage existing genesis
3. Track new remote server
4. Deploy network components
> |
```

Activate Windows
Go to Settings to activate Windows.

- Export genesis configurations. This will fail to create two of the files, but you only need `networkname.json`.
- You can delete the `networkname-harmony.json` file.
- Screenshot the `puppeth` configuration once complete and save it to the Screenshots folder.

Network was named zach after our instructor:

```
MINGW64/c/Users/Anand/Blockchain-Tools
How many seconds should blocks take? (default = 15)
> 5

Which accounts are allowed to seal? (mandatory at least one)
> 0x5A4eD3836076A25687847286d18d39c89c0ff060
> 0x

Which accounts should be pre-funded? (advisable at least one)
> 0x5A4eD3836076A25687847286d18d39c89c0ff060
> 0x

Should the precompile-addresses (0x1 .. 0xff) be pre-funded with 1 wei? (advisable yes)
> no

Specify your chain/network ID if you want an explicit one (default = random)
> 333
INFO [09-04|15:42:55.336] Configured new genesis block

What would you like to do? (default = stats)
1. Show network stats
2. Manage existing genesis
3. Track new remote server
4. Deploy network components
> 2

1. Modify existing configurations
2. Export genesis configurations
3. Remove genesis configuration
> 2

Which folder to save the genesis specs into? (default = current)
Will create zach.json, zach-aleth.json, zach-harmony.json, zach-parity.json
> C:\Users\Anand\Blockchain-Tools
INFO [09-04|15:46:17.637] Saved native genesis chain spec path=C:\Users\Anand\Blockchain-Tools\zach.json
ERROR[09-04|15:46:17.713] Failed to create Aleth chain spec err="unsupported consensus engine"
ERROR[09-04|15:46:17.714] Failed to create Parity chain spec err="unsupported consensus engine"
INFO [09-04|15:46:17.839] Saved genesis chain spec client=harmony path=C:\Users\Anand\Blockchain-Tools\zach-harmony.json

What would you like to do? (default = stats)
1. Show network stats
2. Manage existing genesis
3. Track new remote server
4. Deploy network components
> |
```

- Initialize each node with the new `networkname.json` with `geth`.
- Run the first node, unlock the account, enable mining, and the RPC flag. Only one node needs RPC enabled.
- Set a different peer port for the second node and use the first node's `enode` address as the `bootnode` flag.
- Be sure to unlock the account and enable mining on the second node!
- You should now see both nodes producing new blocks, congratulations!

Using `geth`, I ran the two nodes. Here is the command to run node1:

```
MINGW64/c/Users/Anand/Blockchain-Tools
Downloads/
Favorites/
Links/
'Local Settings'@
Music/
'My Documents'@
NTUSER.DAT
NTUSER.DAT{53b39e87-18c4-11ea-a811-000d3aa4692b}.TxR.0.regtrans-ms
NTUSER.DAT{53b39e87-18c4-11ea-a811-000d3aa4692b}.TxR.1.regtrans-ms
NTUSER.DAT{53b39e87-18c4-11ea-a811-000d3aa4692b}.TxR.2.regtrans-ms
NTUSER.DAT{53b39e87-18c4-11ea-a811-000d3aa4692b}.TxR.b1f
NTUSER.DAT{53b39e88-18c4-11ea-a811-000d3aa4692b}.TM.b1f
NTUSER.DAT{53b39e88-18c4-11ea-a811-000d3aa4692b}.TMContainer000000000000000000
1.regtrans-ms
NTUSER.DAT{53b39e88-18c4-11ea-a811-000d3aa4692b}.TMContainer000000000000000000
2.regtrans-ms
NetHood@
OneDrive/
Pictures/
PrintHood@
Recent@
'Saved Games'/
Searches/
SendTo@
'Start Menu'@
Templates@
Untitled.ipynb
Videos/
anaconda3/
ntuser.dat.LOG1
ntuser.dat.LOG2
ntuser.ini
wallet/
(base)
Anand@DESKTOP-V26681V MINGW64 ~
$ cd wallet/
(base)
Anand@DESKTOP-V26681V MINGW64 ~/wallet
$ ls
Multi-Blockchain-Wallet.ipynb  constants.py  hd-wallet-derive/
__pycache__/                 derive@      wallet.py
(base)
Anand@DESKTOP-V26681V MINGW64 ~/wallet
$ cd ..
(base)
Anand@DESKTOP-V26681V MINGW64 ~
$ cd Blockchain-Tools/
(base)
Anand@DESKTOP-V26681V MINGW64 ~/Blockchain-Tools
$ ./geth --datadir node1 --unlock 0x293fa33066893E8B3cFBF36B2864749D70c2eF5c --mine --rpc --allow-insecure-unlock
```

This is the result of the command. I put in the testnet password to unlock the node.

```
MINGW64/c/Users/Anand/Blockchain-Tools
Anand@DESKTOP-V26681V MINGW64 ~/wallet
$ ls
Multi-Blockchain-Wallet.ipynb  constants.py  hd-wallet-derive/
_pycache/                    derive@      wallet.py
(base)
Anand@DESKTOP-V26681V MINGW64 ~/wallet
$ cd ..
(base)
Anand@DESKTOP-V26681V MINGW64 ~
$ cd Blockchain-Tools/
(base)
Anand@DESKTOP-V26681V MINGW64 ~/Blockchain-Tools
$ ./geth --datadir node1 --unlock 0x293fa33066893EBB3cF8F36B2864749D70c2eF5c --mine --rpc --allow-insecure-unlock
INFO [09-12|09:21:26.518] Bumping default cache on mainnet      provided=1024 updated=4096
WARN [09-12|09:21:26.520] Sanitizing cache to Go's GC limits    provided=4096 updated=682
INFO [09-12|09:21:26.523] Maximum peer count                    ETH=50 LES=0 total=50
INFO [09-12|09:21:26.875] Starting peer-to-peer node            instance=Geth/v1.9.7-stable-a718daa6/windows-amd64/go1.13.4
INFO [09-12|09:21:26.875] Allocated trie memory caches          clean=170.00MiB dirty=170.00MiB
INFO [09-12|09:21:26.875] Allocated cache and file handles      database=C:\Users\Anand\Blockchain-Tools\node1\geth\chaindata cache=34
1.00MiB handles=8192
INFO [09-12|09:21:27.984] Opened ancient database               database=C:\Users\Anand\Blockchain-Tools\node1\geth\chaindata\ancient
INFO [09-12|09:21:27.984] Initialised chain configuration        config="{ChainID: 333 Homestead: 0 DAO: <nil> DAOsupport: false EIP150: 0 EI
P155: 0 EIP158: 0 Byzantium: 0 Constantinople: 0 Petersburg: 0 Istanbul: 0 Engine: clique}"
INFO [09-12|09:21:27.984] Initialising Ethereum protocol        versions="[64 63]" network=1 dbversion=7
INFO [09-12|09:21:28.231] Loaded most recent local header        number=0 hash=8bc06d_b861b1 td=1 age=1w20h58m
INFO [09-12|09:21:28.232] Loaded most recent local full block    number=0 hash=8bc06d_b861b1 td=1 age=1w20h58m
INFO [09-12|09:21:28.232] Loaded most recent local fast block    number=0 hash=8bc06d_b861b1 td=1 age=1w20h58m
INFO [09-12|09:21:28.464] Loaded local transaction journal       transactions=0 dropped=0
INFO [09-12|09:21:28.466] Regenerated local transaction journal   transactions=0 accounts=0
INFO [09-12|09:21:28.490] Allocated fast sync bloom              size=340.00MiB
INFO [09-12|09:21:28.504] Initialized fast sync bloom             items=354 errorrate=0.000 elapsed=7.796ms
INFO [09-12|09:21:30.856] New local node record                  seq=11 id=d6c2185e2857e0fb ip=127.0.0.1 udp=30303 tcp=30303
INFO [09-12|09:21:31.435] Started P2P networking                 self=enode://b3b1eda6b74648982f79337e1bbd8542d6af02c2051709031aa40bb2247173e
657c7e2dc8ccadd4bcd510207b2dd8edacc96585c89fc7bd04c411a1cba8b47ed8127.0.0.1:30303
INFO [09-12|09:21:31.601] IPC endpoint opened                    url=\\\\.\\pipe\\geth.ipc
INFO [09-12|09:21:32.325] HTTP endpoint opened                   url=http://127.0.0.1:8545 cors= vhosts=localhost
Unlocking account 0x293fa33066893EBB3cF8F36B2864749D70c2eF5c | Attempt 1/3
!! Unsupported terminal, password will be echoed.
Password: testnetINFO [09-12|09:21:35.780] New local node record                  seq=12 id=d6c2185e2857e0fb ip=68.231.172.155 udp=65370 tcp=
30303
INFO [09-12|09:21:50.061] Unlocked account                      address=0x293fa33066893EBB3cF8F36B2864749D70c2eF5c
INFO [09-12|09:21:50.567] Transaction pool price threshold updated price=1000000000
INFO [09-12|09:21:50.686] Transaction pool price threshold updated price=1000000000
INFO [09-12|09:21:50.686] Etherbase automatically configured    address=0x293fa33066893EBB3cF8F36B2864749D70c2eF5c
WARN [09-12|09:21:51.960] Block sealing failed                   err="unauthorized signer"
INFO [09-12|09:21:51.968] Commit new mining work                 number=1 sealhash=674ca6..4210c7 uncles=0 txs=0 gas=0 fees=0 elapsed=826.079m
s
```

Then I opened up another terminal and ran the command for node2:

```
MINGW64/c/Users/Anand/Blockchain-Tools
(base)
Anand@DESKTOP-V26681V MINGW64 ~
$ cd Blockchain-Tools/
(base)
Anand@DESKTOP-V26681V MINGW64 ~/Blockchain-Tools
$ ./geth --datadir node2 --unlock 0x0c2cd784E23E2D67F1aa087E61af8e19bB986757 --mine --port 30304 --bootnodes "enode://b3b1eda6b74648982f79337e1bbd8542d6af02c2051709031aa40bb2247173e657c7e2dc8ccadd4bcd510207b2dd8edacc96585c89fc7bd04c411a1cba8b47ed8127.0.0.1:30303" --ipcdisable --allow-insecure-unlock247173e657c7e2dc8ccadd4bcd510207b2dd8edacc96585c89fc7bd04c411a1cba
```

This is the result along with the password to unlock the node:


```
MINGW64/c:/Users/Anand/Blockchain-Tools
(base)
Anand@DESKTOP-V26681V MINGW64 ~
$ cd Blockchain-Tools/
(base)
Anand@DESKTOP-V26681V MINGW64 ~/Blockchain-Tools
$ ./geth --datadir node2 --unlock 0x0c2cd784E23E2D67F1aa087E61af8e19bB986757 --mine --port 30304 --bootnodes "enode://b3b1eda6b74648982f79337e1
bbd8542d6af02c2051709031aa40bb2247173e657c7e2dc8ccadd4bcd510207b2dd8edacc96585c89fc7bd04c411a1cba8b47ed8127.0.0.1:30303" --ipcdisable --allow-i
nsecure-unlock247173e657c7e2dc8ccadd4bcd510207b2dd8edacc96585c89fc7bd04c411a1cha
INFO [09-12|09:23:55.929] Bumping default cache on mainnet      provided=1024 updated=4096
WARN [09-12|09:23:55.942] Sanitizing cache to Go's GC limits      provided=4096 updated=682
INFO [09-12|09:23:55.944] Maximum peer count                      ETH=50 LES=0 total=50
INFO [09-12|09:23:56.181] Starting peer-to-peer node             instance=Geth/v1.9.7-stable-a718daa6/windows-amd64/go1.13.4
INFO [09-12|09:23:56.181] Allocated trie memory caches           clean=170.00MiB dirty=170.00MiB
INFO [09-12|09:23:56.181] Allocated cache and file handles       database=C:\Users\Anand\Blockchain-Tools\node2\geth\chaindata cache=34
1.00MiB handles=8192
INFO [09-12|09:23:56.595] Opened ancient database                 database=C:\Users\Anand\Blockchain-Tools\node2\geth\chaindata\ancient
INFO [09-12|09:23:56.683] Initialised chain configuration         config="{ChainID: 333 Homestead: 0 DAO: <nil> DAOsupport: false EIP150: 0 EI
P155: 0 EIP158: 0 Byzantium: 0 Constantinople: 0 Petersburg: 0 Istanbul: 0 Engine: clique}"
INFO [09-12|09:23:56.684] Initialising Ethereum protocol         versions="[64 63]" network=1 dbversion=7
INFO [09-12|09:23:57.254] Loaded most recent local header        number=0 hash=8bc06d_b861b1 td=1 age=1w21h1m
INFO [09-12|09:23:57.265] Loaded most recent local full block    number=0 hash=8bc06d_b861b1 td=1 age=1w21h1m
INFO [09-12|09:23:57.269] Loaded most recent local fast block    number=0 hash=8bc06d_b861b1 td=1 age=1w21h1m
INFO [09-12|09:23:58.190] Loaded local transaction journal       transactions=0 dropped=0
INFO [09-12|09:23:58.360] Regenerated local transaction journal   transactions=0 accounts=0
INFO [09-12|09:23:58.376] Allocated fast sync bloom             size=340.00MiB
INFO [09-12|09:23:58.610] Initialized fast sync bloom            items=354 errorrate=0.000 elapsed=12.728ms
INFO [09-12|09:24:01.391] New local node record                  seq=8 id=18e1b8c670f4b843 ip=127.0.0.1 udp=30304 tcp=30304
INFO [09-12|09:24:02.275] Started P2P networking                 self=enode://f86ebf7e643c2244c0730247f152275128d3f66e71de20248d6753d873f0156
dc9c41e0a378a305c1cd860a29622c042cd1e544c84ca24379f86c6542f364a8127.0.0.1:30304
Unlocking account 0x0c2cd784E23E2D67F1aa087E61af8e19bB986757 | Attempt 1/3
!! Unsupported terminal, password will be echoed.
Password: tINFO [09-12|09:24:04.994] New local node record      seq=9 id=18e1b8c670f4b843 ip=68.231.172.155 udp=52523 tcp=30304
estnet
INFO [09-12|09:24:14.169] Unlocked account                      address=0x0c2cd784E23E2D67F1aa087E61af8e19bB986757
INFO [09-12|09:24:14.286] Transaction pool price threshold updated price=1000000000
INFO [09-12|09:24:14.360] Transaction pool price threshold updated price=1000000000
INFO [09-12|09:24:14.366] Etherbase automatically configured    address=0x0c2cd784E23E2D67F1aa087E61af8e19bB986757
WARN [09-12|09:24:14.493] Block sealing failed                  err="unauthorized signer"
INFO [09-12|09:24:14.494] Commit new mining work                 number=1 sealhash=c95fcf...74a430 uncles=0 txs=0 gas=0 fees=0 elapsed=66.641ms
```

The nodes were both running. This is where I ran into problems. I tried connecting to the network and was not able to. I checked the firewall and the settings, but all of those seemed to be ok. I tried running it in a different machine and got the same result. Here is one of several attempts to connect to the zach network including creating a new zach1 network. None of these attempts bore fruit. For future work, I will look into this and maybe use some software other than mycrypto to see if it will connect:

