# README Addendum

## SSH without password

Before deployment, or before executing deploy.sh, you may need to configure the Deployment node to be able to SSH automatically and silently into other Bank nodes without prompting for SSH user and password. Below are the sample steps to do this:

[‘Credits’](https://www.tecmint.com/ssh-passwordless-login-using-ssh-keygen-in-5-easy-steps/)

From source computer, do this step:

(1) ssh-keygen -t rsa

Generating public/private rsa key pair.

Enter file in which to save the key (/home/sebtno/.ssh/id\_rsa):

Created directory '/home/sebtno/.ssh'.

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /home/sebtno/.ssh/id\_rsa.

Your public key has been saved in /home/sebtno/.ssh/id\_rsa.pub.

The key fingerprint is:

SHA256:RwHwvmogmUI5V03q4SK5/HWnqQQTaTpT0titnUadRBs sebtno@quorumnx01

The key's randomart image is:

+---[RSA 2048]----+

| =E... |

| + o.+o+ . |

| o.B.= +. . |

| +\*.B o. . |

|.\*oB \* S . |

|o O \* o |

| + . + ... |

| . o o.+ |

| . ooo |

+----[SHA256]-----+

For each destination, do the 4 steps FROM source computer:

(2) ssh sebtno@<Destination IP> mkdir -p .ssh

The authenticity of host '192.168.30.130 (192.168.30.130)' can't be established.

ECDSA key fingerprint is SHA256:Zg6gwGUFii4FOsyHwpYtVWC+qMx7pLiDfDk3a+xT268.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added '192.168.30.130' (ECDSA) to the list of known hosts.

sebtno@192.168.30.130's password:

(3) cat .ssh/id\_rsa.pub | ssh sebtno@<Destination IP> 'cat >> .ssh/authorized\_keys'

sebtno@192.168.30.130's password:

(4) ssh sebtno@<Destination IP> "chmod 700 .ssh; chmod 640 .ssh/authorized\_keys"

(5) ssh sebtno@<Destination IP>

## Backup your VMs

It is recommended to backup you VMs before each executing each section below.

## Configure coordinating node

### Configure coordinating node – Part 1

In readme’s step 2. Configure coordinating node,

change the IP to the static IP that other nodes can communicate

e.g from 115.66.31.158 to 192.168.9.136

prompt: localIpAddress: (115.66.31.158) 192.168.9.136

prompt: nodeName: nx01

Please select an option:

1) Raft

2) QuorumChain

5) Kill all geth and constellation

prompt: option: 5

ERROR: geth: no process found

constellation-node: no process found

geth: no process found

constellation-node: no process found

Please select an option:

1) Raft

2) QuorumChain

5) Kill all geth and constellation

prompt: option: 1

Please select an option below:

----- Option 1 and 2 are for the initial setup of a raft network -----

1) Start a node as the setup coordinator [Ideally there should only be one coordinator]

2) Start a node as a non-coordinator

----- Option 3 is for joining a raft network post initial setup -----

3) Join a raft network if you were not part of the initial setup

4) TODO: Start whisper services and attach to already running node

5) killall geth constellation-node

0) Quit

prompt: option: 1

Please select an option below:

1) Allow anyone to connect

2) [TODO] Allow only people with pre-auth tokens to connect

prompt: option: 1

Please select an option below:

1) Clear all files/configuration and start from scratch[WARNING: this clears everything]

2) Keep old files/configuration intact and start the node + whisper services

prompt: option: 1

[\*] Starting new node...

Generating node key

enode: enode://be28170a4761bbe181dd0b18c767c081d01ed3c481a15d01a08f1120f5ef1ad615126af437563b307780d0d1bac4f9c8ff3f5f18dd7614ea1dd22b41a80077ca@192.168.9.136:20000

[\*] Starting communication node...

[\*] RPC connection established, Node started

Please wait for others to join. Hit any key + enter once done.

prompt: done:

NOTE: You **hit any key and press enter** only **when ALL nodes are added**.

Failing this, you will encounter either the “node index.js” hangs in Participating nodes or the “node index.js” crashes in Coordinating node.

At this point, go to below section on Participating nodes (non-coordinator). Return here when Participating nodes (non-coordinator) section is complete.

### Configure coordinating node – Part 2

After ALL non-coordinating nodes are added and are waiting to join, you have to **hit any key and press enter**. (Below shows we are only added nx02 and nx03 for the whole network)

Please wait for others to join. Hit any key + enter once done.

prompt: done: nx02 has joined the network

nx03 has joined the network

p 🡨 hit p <enter>

Adding the following addresses to the genesis block: [ '0x6398dd29d801211d2492e9b6062c33b8aac74599',

'0x000000000000000000000000000000005a534c01',

'0x000000000000000000000000000000005a534c02',

'0x000000000000000000000000000000005a534c03',

'0x000000000000000000000000000000005a534c04',

'0x000000000000000000000000000000005a534c05',

'0x4a96d31bfe8d4e797ba60cb9994fb18220354263',

'0x49c40d934caf14114210161ab4bd6d59d12b53e3' ]

[\*] Creating genesis config...

[\*] Starting raft node...

[\*] RPC connection established, Node started

[\*] Done

Please select an option below:

----- Option 1 and 2 are for the initial setup of a raft network -----

1) Start a node as the setup coordinator [Ideally there should only be one coordinator]

2) Start a node as a non-coordinator

----- Option 3 is for joining a raft network post initial setup -----

3) Join a raft network if you were not part of the initial setup

4) TODO: Start whisper services and attach to already running node

5) killall geth constellation-node

0) Quit

prompt: option:

Go to Verify Coordinating Node section below.

Go to Verify Participating Node section below.

After you are done verifying, you can do the following

4) TODO: Start whisper services and attach to already running node

5) killall geth constellation-node

0) Quit

prompt: option:

You can enter “5” to kill all *geth* and *constellation-node* processes before you “0” to Quit.

If you leave the processes running, you can kill them later using

*ubin-quorum-setup/binaries/setup/ cleanup\_process.sh*

By now, all nodes would have the geth and quorum data-dirs populated.

You can re-start the same way again by executing the same steps, except that you will just select option 2 for the coordinator node and the rest of non- coordinator nodes.

2) Keep old files/configuration intact and start the node + whisper services

## Participating Nodes (non-coordinator)

Follow readme’s 3. Steps for Participating nodes (non-coordinator)

change the IP to the static IP that other nodes can communicate

e.g from 115.66.31.158 to 192.168.9.137

prompt: localIpAddress: (115.66.31.158) 192.168.9.137

prompt: nodeName: nx02

Please select an option:

1) Raft

2) QuorumChain

5) Kill all geth and constellation

prompt: option: 5

ERROR: geth: no process found

constellation-node: no process found

geth: no process found

constellation-node: no process found

Please select an option:

1) Raft

2) QuorumChain

5) Kill all geth and constellation

prompt: option: 1

Please select an option below:

----- Option 1 and 2 are for the initial setup of a raft network -----

1) Start a node as the setup coordinator [Ideally there should only be one coordinator]

2) Start a node as a non-coordinator

----- Option 3 is for joining a raft network post initial setup -----

3) Join a raft network if you were not part of the initial setup

4) TODO: Start whisper services and attach to already running node

5) killall geth constellation-node

0) Quit

prompt: option: 2

Please select an option below:

1) Clear all files/configuration and start from scratch[WARNING: this clears everything]

2) Keep old files/configuration intact and start the node + whisper services

prompt: option: 1

In order to join the network, please enter the ip address of the coordinating node

prompt: ipAddress: 192.168.9.136

[\*] Starting new node...

Account: 0x4a96d31bfe8d4e797ba60cb9994fb18220354263

Generating node key

enode: enode://63d0c71de52c59540150542b4e5db1353e4fe43ebfcc1c19fb7b4889ff75788842a410e389afcbb047e6ca7fcc7405df4737b9b58c7d89f57589c0810d6ce9b9@192.168.9.137:20000

[\*] Joining communication network...

[\*] RPC connection established, Node started

true

[\*] Communication network joined

[\*] Requesting network membership. This will block until the other node responds

[\*] Network membership: ACCEPTED

[\*] Requesting genesis block config. This will block until the other node is online

received genesis config

[\*] Requesting static nodes file. This will block until the other node is online

received static nodes file

[\*] RPC connection established, Node started

[\*] New node started

Please select an option below:

----- Option 1 and 2 are for the initial setup of a raft network -----

1) Start a node as the setup coordinator [Ideally there should only be one coordinator]

2) Start a node as a non-coordinator

----- Option 3 is for joining a raft network post initial setup -----

3) Join a raft network if you were not part of the initial setup

4) TODO: Start whisper services and attach to already running node

5) killall geth constellation-node

0) Quit

prompt:

Return to “Configure coordinating node – Part 2” above.

After you are done above, you can do the following:

4) TODO: Start whisper services and attach to already running node

5) killall geth constellation-node

0) Quit

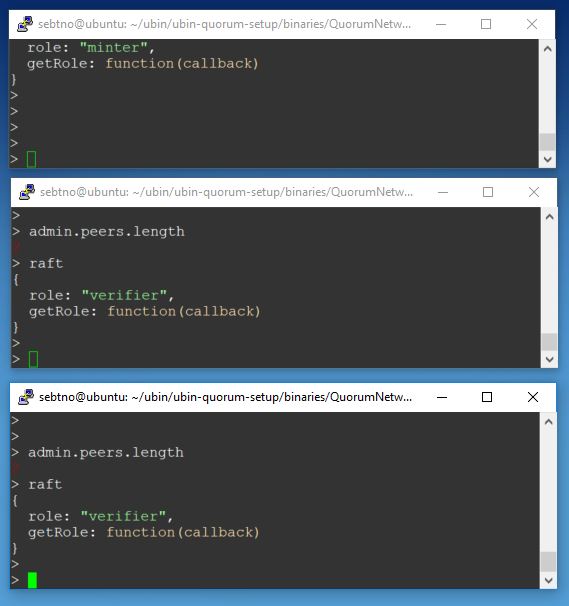
prompt:

You can enter “5” to kill all *geth* and *constellation-node* processes before you “0” to Quit.

If you leave the processes running, you can kill them later using

*ubin-quorum-setup/binaries/setup/ cleanup\_process.sh*

# 3 Quorum nodes up and running for now



## Verify Coordinating Node

If you are in screen session, Ctrl-A-D out to shell and attach to quorum geth node:

sebtno@ubuntu:~/ubin/ubin-quorum-setup/binaries/QuorumNetworkManager$ ./attachToLocalQuorumNode.sh

Welcome to the Geth JavaScript console!

instance: Geth/v1.5.0-unstable-f4adbc2e/linux/go1.7.3

coinbase: 0x178651dfebe721845ec7abf2b79003ba9a5dcb91

at block: 0 (Wed, 31 Dec 1969 16:00:00 PST)

datadir: /home/sebtno/ubin/ubin-quorum-setup/binaries/QuorumNetworkManager/Blockchain

modules: admin:1.0 debug:1.0 eth:1.0 net:1.0 personal:1.0 quorum:1.0 raft:1.0 rpc:1.0 shh:1.0 txpool:1.0 web3:1.0 zsl:1.0

> raft

{

role: "minter",

getRole: function(callback)

}

> eth.accounts

["0x178651dfebe721845ec7abf2b79003ba9a5dcb91"]

> eth.getBalance("0x178651dfebe721845ec7abf2b79003ba9a5dcb91")

1e+27

## Verify Participating Node

If you are in screen session, Ctrl-A-D out to shell and attach to quorum geth node:

sebtno@ubuntu:~/ubin/ubin-quorum-setup/binaries/QuorumNetworkManager$ ./attachToLocalQuorumNode.sh Welcome to the Geth JavaScript console!

instance: Geth/v1.5.0-unstable-f4adbc2e/linux/go1.7.3

coinbase: 0x96e9368ad97eebc5a821282a4dbff3ecf17620c6

at block: 0 (Wed, 31 Dec 1969 16:00:00 PST)

datadir: /home/sebtno/ubin/ubin-quorum-setup/binaries/QuorumNetworkManager/Blockchain

modules: admin:1.0 debug:1.0 eth:1.0 net:1.0 personal:1.0 quorum:1.0 raft:1.0 rpc:1.0 shh:1.0 txpool:1.0 web3:1.0 zsl:1.0

> raft

{

role: "verifier",

getRole: function(callback)

}

> ["0x96e9368ad97eebc5a821282a4dbff3ecf17620c6"]

["0x96e9368ad97eebc5a821282a4dbff3ecf17620c6"]

> eth.getBalance("0x96e9368ad97eebc5a821282a4dbff3ecf17620c6")

1e+27

> eth.blockNumber

0

> txpool

{

content: {

pending: {},

queued: {}

},

inspect: {

pending: {},

queued: {}

},

status: {

pending: 0,

queued: 0

},

getContent: function(callback),

getInspect: function(callback),

getStatus: function(callback)

}

>

## nx02 sends 123 to nx01

nx02

> eth.sendTransaction({from:eth.accounts[0], to:"0x178651dfebe721845ec7abf2b79003ba9a5dcb91", value:123})

"0x3b1330e22797a4bed376aa654f9976dd585653c5fcb8ceb30c8be0b7f6b26663"

> eth.getBalance(eth.accounts[0])

9.99999999999999999999999877e+26

>

nx01

> eth.accounts

["0x178651dfebe721845ec7abf2b79003ba9a5dcb91"]

> eth.getBalance("0x178651dfebe721845ec7abf2b79003ba9a5dcb91")

1e+27

> eth.getBalance(eth.accounts[0])

1.000000000000000000000000123e+27

>

# ReadMe section C. Contract Deployment

This section attempts to deploy SGDz contract to nx11. Our minimal setup has only 5 VMs running - nx01, nx02, nx03, nx04, nx05.

Instead of nx11, I shall test this section using nx05 as deployment node.

Update /etc/hosts and /etc/hostname as well so that these nodes know each other by **FQDN** (fully qualified domain name). E.g. all 5 nodes respectively.

|  |
| --- |
| sebtno@quorumnx05:~/ubin/ubin-quorum/build/contracts$ more /etc/hostname  quorumnx05  sebtno@quorumnx05:~/ubin/ubin-quorum/build/contracts$ more /etc/hosts  127.0.0.1 localhost  192.168.9.140 quorumnx01 quorumnx01.southeastasia.cloudapp.azure.com  192.168.9.137 quorumnx02 quorumnx02.southeastasia.cloudapp.azure.com  192.168.9.138 quorumnx03 quorumnx03.southeastasia.cloudapp.azure.com  192.168.9.139 quorumnx04 quorumnx04.southeastasia.cloudapp.azure.com  192.168.9.141 quorumnx05 quorumnx05.southeastasia.cloudapp.azure.com  # The following lines are desirable for IPv6 capable hosts  ::1 ip6-localhost ip6-loopback  fe00::0 ip6-localnet  ff00::0 ip6-mcastprefix  ff02::1 ip6-allnodes  ff02::2 ip6-allrouters |

All VMs are restarted after updates

## networkNodesInfo.json

The statement from README, “1. Copy the network configuration (networkNodeInfo.js)”, is wrong.

The file to copy should be “networkNodesInfo.json”.

## Git clone to nx05 (deployment node)

(This step is missing in this section)

git clone <https://github.com/project-ubin/ubin-quorum.git>

networkNodesInfo.json, convertConfig.js, deploy.sh can only be found in this repo.

## networkNodesInfo.json

Ascertain if the json data, especially “ipAddress” etc., are correct or should be updated.

Alternatively, as suggested by PHH (Prof Pieter Hartel), the convert script can be changed to suit your hostnames and domain names, such as the line from convertConfig.js:

Original:

"host" : "quorumnx"+nodeId+".southeastasia.cloudapp.azure.com”,

New:

"host" : "ubin"+parseInt( nodeId ) +".ewi.tudelft.nl”,

I manually removed the entry for “nx05”, the deployment node. My networkNodesInfo.json looks like this:

|  |
| --- |
| {  "4cb37375f73a56d17ba765c354f4ce9d37535f04dc7d1bfc3b0ee080bd467e450f6a6b2892dbdb68d9eb27cfe44072de4abd3ffec26816e56536d7188168f2cd":{"whisperId":"0x044e1e2f9c3a6e0ea458b816189d69286b7a56e85e0a5987c07884181d2cfa81661ea9f1daa640824243ccc04875fb5d71730915324a62535c2f70cd8305196e77",  "nodePubKey":"4cb37375f73a56d17ba765c354f4ce9d37535f04dc7d1bfc3b0ee080bd467e450f6a6b2892dbdb68d9eb27cfe44072de4abd3ffec26816e56536d7188168f2cd",  "ipAddress":"192.168.9.140",  "nodeName":"nx01",  "address":"0xe6ffc04d427d50a33bf82cf3c892ca725d3aede0",  "constellationPublicKey":"gYjkGtu94WzmRCP603ukUxT3BGU+gPmfgEaBeQknpWQ="},  "53bc6d11444549b88bed9b0021989f0e20a67f5b0093e2fa8eec91be7af02a9ff367f16a282396ff9559e882d1feb93cbb1f11a2456c3189df615992e2e63bab":{"whisperId":"0x046ddb413bf2bf0e411bea6ea4ff2de23346cb27c605497cf0e4f2239b5a371b905c4729f078bb5a6adcc33db7be05c61f36a40b98326134a8ea9627c9ce8b6d1d",  "nodePubKey":"53bc6d11444549b88bed9b0021989f0e20a67f5b0093e2fa8eec91be7af02a9ff367f16a282396ff9559e882d1feb93cbb1f11a2456c3189df615992e2e63bab",  "ipAddress":"192.168.9.137",  "nodeName":"nx02",  "address":"0x42450014b81060925fc204e5b2cbb0e1c05164b1",  "constellationPublicKey":"kq1FwdhHuoeW0mKc++sNMFgLnIpc84LFVpOb0Vi8am4="},  "9c066c9d5c0b34574f11b95d7827c7536112d9ce8781166082e3fbf634f42777166b72a5305d8441967251128a842dbfb6f098cb371186e0bd812fddca2156d8":{"whisperId":"0x04f7434908846abcb77d73eb274d43e6cbcb0bffcc1b4a415a8cfb8df35a1f197ee60a92a6e11d1cf1997a27dfe84353f5ff5de39f90fa7936204231ddea5caa16",  "nodePubKey":"9c066c9d5c0b34574f11b95d7827c7536112d9ce8781166082e3fbf634f42777166b72a5305d8441967251128a842dbfb6f098cb371186e0bd812fddca2156d8",  "ipAddress":"192.168.9.139",  "nodeName":"nx04",  "address":"0xec59c14d6bee54150b22123f1b81c34e088aead4",  "constellationPublicKey":"al96u8HR5TCzc8h8W+6fFGuCzX7P9Lsw1A6nqG1KLnc="},  "d252c2c264b7c33378147cdc7ce23f59fbbdb1ab58fe56c058141f818087e989919bfec4e1c8bd978f62b0df3a74971ca345599e22758c8865cf55bf7196e032":{"whisperId":"0x0447d51f3d976dfa2c977eeab50e7613b66c8838aff08e67d81cf36d1e598a93b81303719866b238dbf7f7c64060700cea2e63af1fcdc69504596b1fa49fc80bc0",  "nodePubKey":"d252c2c264b7c33378147cdc7ce23f59fbbdb1ab58fe56c058141f818087e989919bfec4e1c8bd978f62b0df3a74971ca345599e22758c8865cf55bf7196e032",  "ipAddress":"192.168.9.138",  "nodeName":"nx03",  "address":"0x9c38f9fecbc86f0c5535068896e4112b5e51a398",  "constellationPublicKey":"OxNSL7lXWp0xUsjfVswqM42UwnA+/A7Yxm/DPr73+zs="}  } |

## convertConfig.js

Before running convertConfig.js, modify the “stashNames” variable based on your own list of nodes and check networkNodesInfo.json contains the list of nodes (exclude Deployment node nx05).

My convertConfig.js looks like this:

|  |
| --- |
| /\* original list  var stashNames = {  "01" : "MASREGULATOR",  "02" : "MASGSGSG",  "03" : "BOFASG2X",  "04" : "CHASSGSG",  "05" : "CITISGSG",  "06" : "CSFBSGSX",  "07" : "DBSSSGSG",  "08" : "HSBCSGSG",  "09" : "MTBCSGSG",  "10" : "OCBCSGSG",  "12" : "SCBLSGSG",  "14" : "UOBVSGSG",  "15" : "XSIMSGSG"  };  \*/  // Seb: minimal list, "05" is the deployment node  var stashNames = {  "01" : "MASREGULATOR",  "02" : "MASGSGSG",  "03" : "BOFASG2X",  "04" : "CHASSGSG"  }; |

## deploy.sh – Pre-requisites

Look at what deploy.sh is doing:

|  |
| --- |
| rm -rf build  truffle deploy --network mas  cd solc-local  echo "[\*] deploying SGDz to node 01..."  sudo ./deploy\_multi.sh  cd ..  nohup npm run dev 2 >> ~/lite-server.log & |

deploy.sh invokes truffle to deploy to the network named ‘mas’ as configured in truffle.js. There is no mention of installing which version of truffle.

The pre-requisite steps are described below before running deploy.sh

### Install truffle

Anyhow, just install truffle as below:

sudo npm install -g truffle

### npm install

From ubin-quorum directory where package.json is found, run “npm install” so that required js modules can be resolved when deploy.sh executes js programs later.

### Refresh your memory on truffle commands

<http://truffleframework.com/docs/advanced/commands>

See what is inside the directories ubin-quorum/migrations and ubin-quorum/contracts

Update ubin-quorum/truffle.js before any “truffle deploy”

Except for the deployment node, ensure the hosts are reachable and their respective geth processes are running.

Check the ports the nx01 is listening on, e.g:

|  |
| --- |
| sebtno@quorumnx01:~$ ps -ax | grep geth  2728 pts/18 Sl+ 1:58 geth --datadir CommunicationNode --networkid 91350 --shh --rpc --rpcaddr 0.0.0.0 --rpcapi admin,db,eth,debug,miner,net,shh,txpool,personal,web3,quorum --nodiscover --rpcport 50010 --port 50000  2768 pts/18 Sl+ 9:51 geth --datadir Blockchain --shh --port 20000 --unlock 0 --password passwords.txt --raft --rpc --rpcaddr 0.0.0.0 --rpcapi admin,db,eth,debug,miner,net,shh,txpool,personal,web3,quorum,raft,zsl --rpcport 20010 |

NOTE: --password means the coinbase account is automatically unlocked for transactions. passwords.txt is also empty. In production, this can never be.

### truffle.js

My truffle.js looks like this:

|  |
| --- |
| module.exports = {  networks: {  development: {  host: "localhost", // Seb: quorumnx05.southeastasia.cloudapp.azure.com is the deployment node  port: 8545,  network\_id: "\*", // Match any network id  },  mas: {  host: "quorumnx01.southeastasia.cloudapp.azure.com",  port: 20010,  network\_id: "\*", // Match any network id  from: "0xe6ffc04d427d50a33bf82cf3c892ca725d3aede0", // Seb: nx01's coinbase  gas: 200000000  },  cb: {  host: "quorumnx02.southeastasia.cloudapp.azure.com", // Seb: central bank  port: 20010,  network\_id: "\*" // Match any network id  },  a: {  host: "quorumnx03.southeastasia.cloudapp.azure.com",  port: 20010,  network\_id: "\*" // Match any network id  },  b: {  host: "quorumnx04.southeastasia.cloudapp.azure.com",  port: 20010,  network\_id: "\*" // Match any network id  }  }  }; |

If deploy.sh hangs at “waiting to be mined”,

|  |
| --- |
| sebtno@quorumnx05:~/ubin/ubin-quorum/solc-local$ sudo ./deploy\_multi.sh  [\*] deploying contract...Contract transaction send: TransactionHash: 0x14a147274ec0c43bda7a19d5ac9496e301c1c4caca11d540833c4dd6b4a064d9 waiting to be mined... |

Kill geth and quorum processes in nx05. (geth and quorum processes refers to the Coordinator and Non-coordinator nodes as described in prior sections above)

Only nx01, nx02, nx03, nx04 should be running geth and quorum processes during deploy.sh (1 minter, 3 verifier, may this for Raft protocol)

## deploy.sh – Execution

Finally, deploy.sh should execute successfully like this:

|  |
| --- |
| sebtno@quorumnx05:~/ubin/ubin-quorum$ ./deploy.sh  Compiling ./contracts/Migrations.sol...  Compiling ./contracts/Owned.sol...  Compiling ./contracts/PaymentAgent.sol...  Compiling ./contracts/SGDz.sol...  Compiling ./contracts/Stash.sol...  Compiling ./contracts/ZSLPrecompile.sol...  Compilation warnings encountered:  /home/sebtno/ubin/ubin-quorum/contracts/Migrations.sol:11:3: Warning: Defining constructors as functions with the same name as the contract is deprecated. Use "constructor(...) { ... }" instead.  function Migrations() {  ^ (Relevant source part starts here and spans across multiple lines).  ,/home/sebtno/ubin/ubin-quorum/contracts/Owned.sol:6:3: Warning: Defining constructors as functions with the same name as the contract is deprecated. Use "constructor(...) { ... }" instead.  function Owned() {  ^ (Relevant source part starts here and spans across multiple lines).  ,/home/sebtno/ubin/ubin-quorum/contracts/Owned.sol:11:27: Warning: "throw" is deprecated in favour of "revert()", "require()" and "assert()".  if(msg.sender!=owner) throw; \_;  ^---^  ,/home/sebtno/ubin/ubin-quorum/contracts/Stash.sol:15:3: Warning: Defining constructors as functions with the same name as the contract is deprecated. Use "constructor(...) { ... }" instead.  function Stash(bytes32 \_bankName) {  ^ (Relevant source part starts here and spans across multiple lines).  ,/home/sebtno/ubin/ubin-quorum/contracts/Stash.sol:28:26: Warning: "throw" is deprecated in favour of "revert()", "require()" and "assert()".  if (\_dAmt > balance) throw;  ^---^  ... many  ... … warnings  … … … from latest solidity compiler...  Writing artifacts to ./build/contracts  Using network 'mas'.  Running migration: 1\_initial\_migration.js  Deploying Migrations...  ... 0x27382a7d0fbe100a9ab01272ea0c388f4c33de82c08ef71b1bb9ced230fb42f1  Migrations: 0x5fb6d7436d73b389c24f4e93134eaae246e8d132  Saving successful migration to network...  ... 0x612fe16949f797f287353e51b71f15fcfd8f7a73cc06b02b3981df4ccaddb165  Saving artifacts...  Running migration: 2\_deploy\_contracts.js  Deploying SGDz...  ... 0x927083bebe826858c85029292fbcb078f7bade964d25413444aa6cd14180f20a  SGDz: 0xfb8c9174e65844d343eaeafe64d7daa66ce16a80  Deploying PaymentAgent...  ... 0x20abea38235f16a11d55715111d9801cc42d523e1f30ffcee681129421a72ec7  PaymentAgent: 0xd4dd942bec54d85a6928a060595deeaebe92f40c  Saving successful migration to network...  ... 0xd66d311a37dbbe1b9e108cb29452634a36ad1e2d89568a88aedb0356de93a3f1  Saving artifacts...  [\*] deploying SGDz to node 01...  [\*] deploying contract...  deploy\_multi.js... 🡨 Seb added this log  Contract transaction send: TransactionHash: 0x7e8e572011f2e8d00b36c1af1e4e995d116a681b0ff8402d54134ff44926eef2 waiting to be mined...  Contract mined! Address: 0x3b334f3378e155db8eb2e0df4bc5e2c33f947c6c  [\*] hardwiring z-contract address...  [\*] starting lite-server... 🡨 Seb added this log  sebtno@quorumnx05:~/ubin/ubin-quorum$ nohup: redirecting stderr to stdout |

## initStash.sh

Again, look at what initStash.sh is doing.

### config.json

The js programs executed by initStash.sh uses test-scripts/config/config.json as input.

This config.json was previously generated by convertConfig.js. The nodes in this config.json file should be correct if convertConfig.js (described above) was executed based on your correct nodes information.

My config.json looks like this:

|  |
| --- |
| [  {"nodeId":1,  "host":"quorumnx01.southeastasia.cloudapp.azure.com",  "port":"20010",  "accountNumber":0,  "ethKey":"0xe6ffc04d427d50a33bf82cf3c892ca725d3aede0",  "constKey":"gYjkGtu94WzmRCP603ukUxT3BGU+gPmfgEaBeQknpWQ=",  "stashName":"MASREGULATOR",  "enode":"4cb37375f73a56d17ba765c354f4ce9d37535f04dc7d1bfc3b0ee080bd467e450f6a6b2892dbdb68d9eb27cfe44072de4abd3ffec26816e56536d7188168f2cd",  "centralBank":false,  "regulator":true,  "localport":3000},  {"nodeId":2,  "host":"quorumnx02.southeastasia.cloudapp.azure.com",  "port":"20010",  "accountNumber":0,  "ethKey":"0x42450014b81060925fc204e5b2cbb0e1c05164b1",  "constKey":"kq1FwdhHuoeW0mKc++sNMFgLnIpc84LFVpOb0Vi8am4=",  "stashName":"MASGSGSG",  "enode":"53bc6d11444549b88bed9b0021989f0e20a67f5b0093e2fa8eec91be7af02a9ff367f16a282396ff9559e882d1feb93cbb1f11a2456c3189df615992e2e63bab",  "centralBank":true,  "regulator":false,  "localport":3000},  {"nodeId":3,  "host":"quorumnx03.southeastasia.cloudapp.azure.com",  "port":"20010",  "accountNumber":0,  "ethKey":"0x9c38f9fecbc86f0c5535068896e4112b5e51a398",  "constKey":"OxNSL7lXWp0xUsjfVswqM42UwnA+/A7Yxm/DPr73+zs=",  "stashName":"BOFASG2X",  "enode":"d252c2c264b7c33378147cdc7ce23f59fbbdb1ab58fe56c058141f818087e989919bfec4e1c8bd978f62b0df3a74971ca345599e22758c8865cf55bf7196e032",  "centralBank":false,  "regulator":false,  "localport":3000},  {"nodeId":4,  "host":"quorumnx04.southeastasia.cloudapp.azure.com",  "port":"20010",  "accountNumber":0,  "ethKey":"0xec59c14d6bee54150b22123f1b81c34e088aead4",  "constKey":"al96u8HR5TCzc8h8W+6fFGuCzX7P9Lsw1A6nqG1KLnc=",  "stashName":"CHASSGSG",  "enode":"9c066c9d5c0b34574f11b95d7827c7536112d9ce8781166082e3fbf634f42777166b72a5305d8441967251128a842dbfb6f098cb371186e0bd812fddca2156d8",  "centralBank":false,  "regulator":false,  "localport":3000}  ] |

### Fix bugs in pledge.js and setShieldedBalance.js

See Bugs section.

### initStash.sh – 4 nodes only

Execute initStash.sh passing in the balances for the 4 nodes only:

|  |
| --- |
| sebtno@quorumnx05:~/ubin/ubin-quorum/test-scripts$ ./initStash.sh 10 2000 3000 4000  [\*] linking z-contract...  Using network 'mas'.  mined!, block: 98, tx hash: 0x242425baef1f740d7b3add92dbe12089ca1b70a493c8afb148fbeed95df8eb6b  [\*] initializing states and initial funds...  Using network 'mas'.  constellationKeys kq1FwdhHuoeW0mKc++sNMFgLnIpc84LFVpOb0Vi8am4=,OxNSL7lXWp0xUsjfVswqM42UwnA+/A7Yxm/DPr73+zs=,al96u8HR5TCzc8h8W+6fFGuCzX7P9Lsw1A6nqG1KLnc=  cbConstKey kq1FwdhHuoeW0mKc++sNMFgLnIpc84LFVpOb0Vi8am4=  Querying stashes...  Creating stash for MASGSGSG  mined!, block: 99, tx hash: 0xa921cc8ef27f1d9a4ab330524a92ad64675e3eb6d41aeff9827158ae9d3c91b9  Registering stash for MASGSGSG  mined!, block: 100, tx hash: 0x4dafb514664f7bc374f84a97eddeaca19a5990be33727b2d3bf5da1e2396eebc  Marking stash for MASGSGSG  mined!, block: 101, tx hash: 0x8c1ea2ad77b08c7fba30a290d498796b8997948169371da5a20a348d100631f0  Setting stash MASGSGSG as central bank  mined!, block: 102, tx hash: 0x71b874723fc683e943cd6b5f2040f374ed85da97e85798e2d9f423a4af3fa39c  Creating stash for BOFASG2X  mined!, block: 103, tx hash: 0x8b605eb236e9adcde7025fc2fdfed3ef410ce4abcff297b7d3d23989dd91e2f0  Registering stash for BOFASG2X  mined!, block: 104, tx hash: 0x5f47247cfd4100cbe95657335d9447989f654a3408333d3382cee1c7082b69e1  Marking stash for BOFASG2X  mined!, block: 105, tx hash: 0x6c16edba6388360be73f0d38713ae7b89f1b4314ad6b249b141ece31fb3bd909  Creating stash for CHASSGSG  mined!, block: 106, tx hash: 0x905ae9f0569c046e5aebe8ad320958edb45adee94b82322934249ec34d529266  Registering stash for CHASSGSG  mined!, block: 107, tx hash: 0xef097953aba06ff5fb7b436056ab34db35861dff09094e303e43414e76cb624f  Marking stash for CHASSGSG  mined!, block: 108, tx hash: 0x4f06bfabb6ca98584243db06fdb6be434b82f260ffaf2bd3ab5358c7468eae9d  Using network 'mas'.  Setting gridlock resolution trigger to 10  Threshold: 10  Using network 'cb'.  constKey:  [ 'gYjkGtu94WzmRCP603ukUxT3BGU+gPmfgEaBeQknpWQ=',  'OxNSL7lXWp0xUsjfVswqM42UwnA+/A7Yxm/DPr73+zs=',  'al96u8HR5TCzc8h8W+6fFGuCzX7P9Lsw1A6nqG1KLnc=' ]  Setting MASGSGSG's stash balance to 2000...  mined!, block: 110, tx hash: 0x9592c89b8dede8e6a2b9516f3265f502a6a2850c64c449dedd7ff24d18dffb74  Using network 'mas'.  consKey: kq1FwdhHuoeW0mKc++sNMFgLnIpc84LFVpOb0Vi8am4=  Setting MASGSGSG's shielded balance to 2000...  mined!, block: 111  setting currentSalt...  salt: e83e6dff923acedc4c829b3a305e378a  mined!, block: 112  Using network 'cb'.  constKey:  [ 'OxNSL7lXWp0xUsjfVswqM42UwnA+/A7Yxm/DPr73+zs=' ]  Setting BOFASG2X's stash balance to 3000...  mined!, block: 113, tx hash: 0xc65e77a05c25455bc4495659c46fcf1137117983d12fefe8783fd3f59dd00477  Using network 'mas'.  consKey: OxNSL7lXWp0xUsjfVswqM42UwnA+/A7Yxm/DPr73+zs=  Setting BOFASG2X's shielded balance to 3000...  mined!, block: 114  setting currentSalt...  salt: 5260dde2fd42323031d0d6f0351ddf62  mined!, block: 115  Using network 'cb'.  constKey:  [ 'al96u8HR5TCzc8h8W+6fFGuCzX7P9Lsw1A6nqG1KLnc=' ]  Setting CHASSGSG's stash balance to 4000...  mined!, block: 116, tx hash: 0x7384bc7c32e0f0ffc7a584cbc3c17027af06f80e9d3e54fcbb5fbfea1e3beb01  Using network 'mas'.  consKey: al96u8HR5TCzc8h8W+6fFGuCzX7P9Lsw1A6nqG1KLnc=  Setting CHASSGSG's shielded balance to 4000...  mined!, block: 117  setting currentSalt...  salt: b734a190c8fa436e8877723f957565cf  mined!, block: 118  sebtno@quorumnx05:~/ubin/ubin-quorum/test-scripts$ |

# ReadMe section D. DApp Setup

As mentioned in ReadMe, perform “npm install” in all nodes except the deployment node.

## start-api.sh from Deployment Node

This script must be perform in deployment node only. For my case, nx05.

It copies build output from deployment nodes to bank nodes.

It copies config from deployment nodes to bank nodes.

Finally, it starts the api server on each bank node, see *package.json* "start".

See also server/index.js, server/utils/config.js

Ensure the current logged-in user from deployment node is able to *ssh* to order nodes via FQDN without prompting for password. Check the /etc/hosts file for the list of IPs and FQDNs that deployment node needs to *ssh* to. Refer to prior section “SSH without password” for details.

See what start-api.sh is doing, and modify its arguments according to your environment. E.g.

|  |
| --- |
| # Seb:  # modified for own ubin home and user  API\_PATH='/home/sebtno/ubin/ubin-quorum'  HOME\_DIR='/home/sebtno/ubin/ubin-quorum'  USER='sebtno' |

The script also remotes into other nodes and attempts “*sudo*”. This may result in *tty* error if executing user is not logged in as root. Remove the “*sudo*” in the script since the currently logged in user is executing npm daemon and owning the directories.

My start-api.sh looks like this:

|  |
| --- |
| #!/bin/bash  # Seb:  # ====================================================  # !!! IMPORTANT !!!  # This script should be executed from deployment node  # only.  # Ensure deployment node USER can ssh to other nodes,  # check your /etc/hosts file for the list of IPs.  # ====================================================  echo "###############################################"  echo " Starting UBIN Quorum API Layer";  echo "###############################################"  NETWORK\_CONFIG\_PATH='/server/config/network.json'  # Seb:  # modified for own ubin home and user  API\_PATH='/home/sebtno/ubin/ubin-quorum'  HOME\_DIR='/home/sebtno/ubin/ubin-quorum'  USER='sebtno'  # Seb:  # jq is yet another functional language for json  # Installs jq if it is not already installed  if hash jq 2>/dev/null; then  echo "JQ installed.. Proceeding...";  else  echo "JQ is not installed.. Installing JQ.."  sudo apt install jq  fi  echo "###############################################"  # Seb:  # Copies build output from deployment nodes to bank nodes  # Copies config from deployment nodes to bank nodes  # Finally, it starts the api server on each bank node,  # see package.json "start", see also server/index.js, server/utils/config.js  jq -c '.[] | { host, stashName}' server/config/network.json | while read i; do  HOST=`echo $i | jq -r .host`  STASHNAME=`echo $i | jq -r .stashName`  echo "Copying contract to $HOST ..."  # error: ssh -n sebtno@quorumnx01.southeastasia.cloudapp.azure.com "sudo rm -r /home/sebtno/ubin/ubin-quorum/build/contracts/\*json"  # sudo: no tty present and no askpass program specified  # There is no need for sudo, otherwise shell expects su password input from terminal  # ssh -n $USER@$HOST "sudo rm -r $HOME\_DIR/build/contracts/\*json"  # error: ssh -n sebtno@quorumnx01.southeastasia.cloudapp.azure.com "cd /home/sebtno/ubin/ubin-quorum && sudo mkdir -p build/contracts && sudo chmod -R 777 build"  # ssh -n $USER@$HOST "cd $HOME\_DIR && sudo mkdir -p build/contracts && sudo chmod -R 777 build"  ssh -n $USER@$HOST "rm -r $HOME\_DIR/build/contracts/\*json"  ssh -n $USER@$HOST "cd $HOME\_DIR && mkdir -p build/contracts && chmod -R 777 build"  scp -r build/contracts/\*.json $USER@$HOST:$HOME\_DIR/build/contracts    echo "Copying network config to $HOST ..."  # ssh -n $USER@$HOST "sudo rm -r $HOME\_DIR/server/config/network.json"  ssh -n $USER@$HOST "rm -r $HOME\_DIR/server/config/network.json"  scp -r server/config/network.json $USER@$HOST:$HOME\_DIR/server/config/  echo "Starting API Server for $STASHNAME ..."  ssh -n $USER@$HOST bash -c "'  pkill -9 node  cd $API\_PATH  nohup npm start $STASHNAME > ~/api.log &  '"  echo "API Server for $STASHNAME is running"  done |

Note that when stopping the DApp using “pkill -9 node”, the “node index.js” will also be killed. You can kill the PID for DApp by checking with “ps” command.

# Running truffle tests

These tests are run from Deployment node

## run\_test\_cancelPayment.sh

This test involves 1 bank.

|  |
| --- |
| sebtno@quorumnx05:~/ubin/ubin-quorum/test-scripts$ sed -i -e 's/\r$//' run\_test\_cancelPayment.sh  sebtno@quorumnx05:~/ubin/ubin-quorum/test-scripts$ vi run\_test\_cancelPayment.sh  sebtno@quorumnx05:~/ubin/ubin-quorum/test-scripts$ ./run\_test\_cancelPayment.sh  [\*] initializing states and initial funds...  Using network 'mas'.  WIPEOUT!!!  mined!, block: 119, tx hash: 0xa55628646edd2a771ebf0ff4f13eab73e91a19376f59ea3fa37169bb9817d05c  []  Using network 'mas'.  Querying stashes...  Creating stash for MASGSGSG  mined!, block: 120, tx hash: 0x1cf383c5cd0b7957fba0e4990c4230ef2401c81560f59681d65483b65d4d0a61  Marking stash for MASGSGSG  mined!, block: 121, tx hash: 0x2e15959c50582ee1d9a22d4792f1b8eb79621cda5a650d9174f0ff772c988488  Creating stash for BOFASG2X  mined!, block: 122, tx hash: 0x6d3f2124ee7e49845e834a88037945d0c6ea8a6e289b1f48c802ae448d658efd  Marking stash for BOFASG2X  mined!, block: 123, tx hash: 0x498d3ca451d3b71faeadac21328e8366a92f3d2853aac1cb4efab9b659e53954  Creating stash for CHASSGSG  mined!, block: 124, tx hash: 0x64d566093e289628282eaf230857711e25011ec66a8be327f66207b67dcecc88  Marking stash for CHASSGSG  mined!, block: 125, tx hash: 0xedd5b85a93536441c7d5698e9a0e384ccf5716b62ba5803f9d73f2e6f5687756  Using network 'mas'.  Setting MASGSGSG's stash balance to 100...  Error: Invalid number of arguments to Solidity function  at Object.InvalidNumberOfSolidityArgs (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/errors.js:25:1)  at SolidityFunction.validateArgs (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/function.js:74:1)  at SolidityFunction.toPayload (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/function.js:90:1)  at SolidityFunction.sendTransaction (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/function.js:163:1)  at SolidityFunction.execute (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/function.js:256:1)  at /usr/lib/node\_modules/truffle/build/webpack:/~/truffle-contract/contract.js:204:1  at new Promise (<anonymous>)  at /usr/lib/node\_modules/truffle/build/webpack:/~/truffle-contract/contract.js:155:1  at <anonymous>  Using network 'mas'.  Setting BOFASG2X's stash balance to 200...  Error: Invalid number of arguments to Solidity function  at Object.InvalidNumberOfSolidityArgs (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/errors.js:25:1)  at SolidityFunction.validateArgs (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/function.js:74:1)  at SolidityFunction.toPayload (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/function.js:90:1)  at SolidityFunction.sendTransaction (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/function.js:163:1)  at SolidityFunction.execute (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/function.js:256:1)  at /usr/lib/node\_modules/truffle/build/webpack:/~/truffle-contract/contract.js:204:1  at new Promise (<anonymous>)  at /usr/lib/node\_modules/truffle/build/webpack:/~/truffle-contract/contract.js:155:1  at <anonymous>  Using network 'mas'.  Setting CHASSGSG's stash balance to 300...  Error: Invalid number of arguments to Solidity function  at Object.InvalidNumberOfSolidityArgs (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/errors.js:25:1)  at SolidityFunction.validateArgs (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/function.js:74:1)  at SolidityFunction.toPayload (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/function.js:90:1)  at SolidityFunction.sendTransaction (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/function.js:163:1)  at SolidityFunction.execute (/usr/lib/node\_modules/truffle/build/webpack:/~/web3/lib/web3/function.js:256:1)  at /usr/lib/node\_modules/truffle/build/webpack:/~/truffle-contract/contract.js:204:1  at new Promise (<anonymous>)  at /usr/lib/node\_modules/truffle/build/webpack:/~/truffle-contract/contract.js:155:1  at <anonymous>  [\*] submitting payment for regression testing  Using network 'a'.  module.js:549  throw err;  ^  Error: Cannot find module 'js-sha256'  at Function.Module.\_resolveFilename (module.js:547:15)  at Function.Module.\_load (module.js:474:25)  at Module.require (module.js:596:17)  at require (internal/module.js:11:18)  at require (/usr/lib/node\_modules/truffle/build/cli.bundled.js:121295:20)  at /home/sebtno/ubin/ubin-quorum/test-scripts/createTrx.js:9:14  at ContextifyScript.Script.runInContext (vm.js:59:29)  at ContextifyScript.Script.runInNewContext (vm.js:65:15)  at /usr/lib/node\_modules/truffle/build/cli.bundled.js:121314:14  at FSReqWrap.readFileAfterClose [as oncomplete] (fs.js:511:3)  Using network 'a'.  Querying gridlock queue depth...  depth: 0  Querying gridlock queue...  [\*] gridlock queue should be zero  [\*] setting up a queue  Using network 'a'.  module.js:549  throw err;  ^  Error: Cannot find module 'js-sha256'  at Function.Module.\_resolveFilename (module.js:547:15)  at Function.Module.\_load (module.js:474:25)  at Module.require (module.js:596:17)  at require (internal/module.js:11:18)  at require (/usr/lib/node\_modules/truffle/build/cli.bundled.js:121295:20)  at /home/sebtno/ubin/ubin-quorum/test-scripts/createTrx.js:9:14  at ContextifyScript.Script.runInContext (vm.js:59:29)  at ContextifyScript.Script.runInNewContext (vm.js:65:15)  at /usr/lib/node\_modules/truffle/build/cli.bundled.js:121314:14  at FSReqWrap.readFileAfterClose [as oncomplete] (fs.js:511:3)  [\*] cancelling the payment  Using network 'a'.  searching for transaction R0000002...  Trx R0000002 does not exist  [\*] checking payment status  Using network 'a'.  Trx R0000002 does not exist  [\*] submitting another payment  Using network 'a'.  module.js:549  throw err;  ^  Error: Cannot find module 'js-sha256'  at Function.Module.\_resolveFilename (module.js:547:15)  at Function.Module.\_load (module.js:474:25)  at Module.require (module.js:596:17)  at require (internal/module.js:11:18)  at require (/usr/lib/node\_modules/truffle/build/cli.bundled.js:121295:20)  at /home/sebtno/ubin/ubin-quorum/test-scripts/createTrx.js:9:14  at ContextifyScript.Script.runInContext (vm.js:59:29)  at ContextifyScript.Script.runInNewContext (vm.js:65:15)  at /usr/lib/node\_modules/truffle/build/cli.bundled.js:121314:14  at FSReqWrap.readFileAfterClose [as oncomplete] (fs.js:511:3)  Using network 'a'.  Querying gridlock queue depth...  depth: 0  Querying gridlock queue...  [\*] gridlock queue should not contain R0000003 |

# Ubin UI

Allocate a machine to do this, as this will be your web server. Ensure your Quorum nodes are reachable from this machine.

Do

git clone https://github.com/project-ubin/ubin-ui.git

Follow the instructions from,

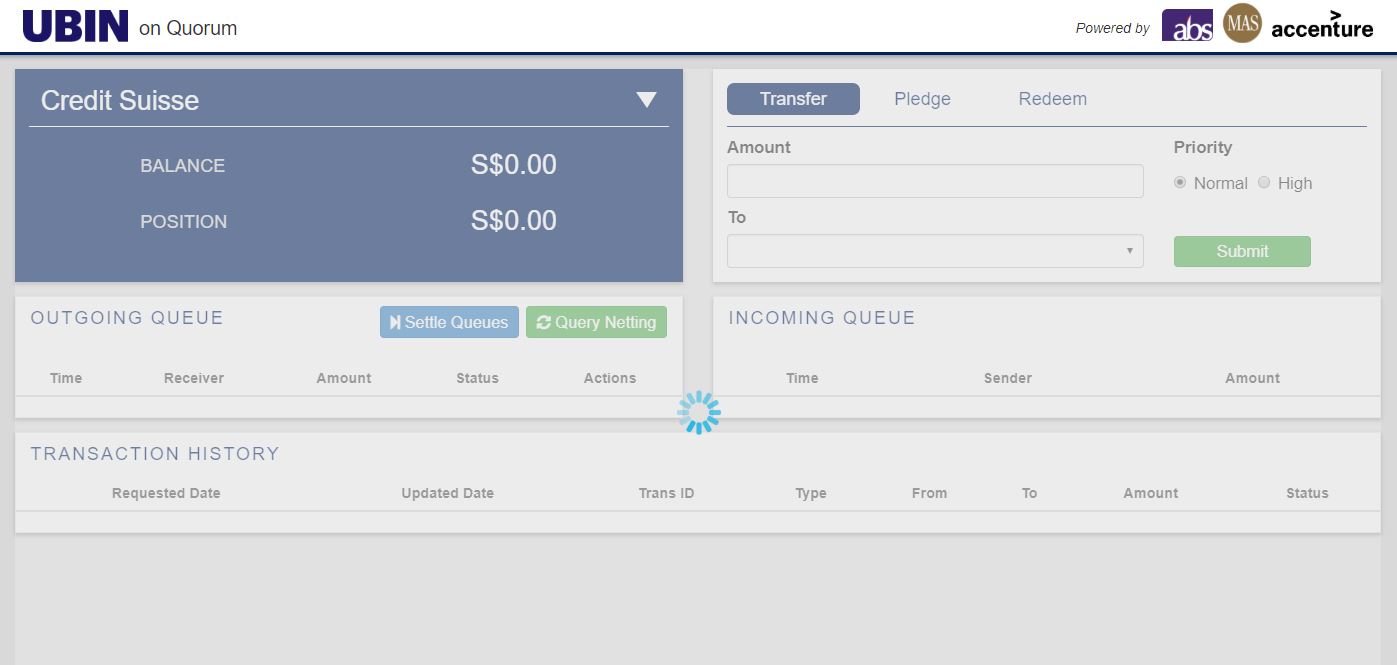
<https://github.com/project-ubin/ubin-ui>

You have to update app/scripts/services/Constants.js according to the nodes and banks you have running.

E.g. of my Constants.js

|  |
| --- |
| // Seb changed  // Endpoints are Linux VMs  // IP addresses used instead because I have not modified by hosts file in my web server machine  var quorumConstants = {  //"mepsEndpoint": "http://quorumnx02.southeastasia.cloudapp.azure.com:9001",  "mepsEndpoint": "http://192.168.9.137:9001",  "bankNodes": {  // "MASREGULATOR": {  // "host": "quorumnx01.southeastasia.cloudapp.azure.com",  // "port": "3000",  // "bankName": "Monetary Authority of Singapore Regulator",  // "centralBank": false,  // "regulator": true  // },  "MASGSGSG": {  //"host": "quorumnx02.southeastasia.cloudapp.azure.com",  "host": "192.168.9.137",  "port": "3000",  "bankName": "Monetary Authority of Singapore Central Bank",  "centralBank": true,  "regulator": false  },  "BOFASG2X": {  //"host": "quorumnx03.southeastasia.cloudapp.azure.com",  "host": "192.168.9.138",  "port": "3000",  "bankName": "Bank of America Merrill Lynch",  "centralBank": false,  "regulator": false  },  "CHASSGSG": {  //"host": "quorumnx04.southeastasia.cloudapp.azure.com",  "host": "192.168.9.139",  "port": "3000",  "bankName": "J.P. Morgan Chase",  "centralBank": false,  "regulator": false  }  …  //constants.defaultBankLoc = "CSFBSGSX"; // Seb changed  constants.defaultBankLoc = "BOFASG2X"; |

Below shows the screen shot of Project Ubin UI using Quorum implementation.



# Bugs

When using screen sessions, you may not see exception stack or error trace when node.js throws or crashes. Detach from all screens, re-produce your error from normal ssh shell sessions.

## Bug 1 - TypeError: peers.contains is not a function

prompt: option: nx03 has joined the network

/home/sebtno/ubin/ubin-quorum-setup/binaries/QuorumNetworkManager/Communication/networkMembership.js:197

if(!peers.contains(msg.from) && message && message.indexOf(request) >= 0){

^

TypeError: peers.contains is not a function

at /home/sebtno/ubin/ubin-quorum-setup/binaries/QuorumNetworkManager/Communication/networkMembership.js:197:15

at /home/sebtno/ubin/ubin-quorum-setup/binaries/QuorumNetworkManager/node\_modules/web3/lib/web3/filter.js:120:21

at Array.forEach (<anonymous>)

at /home/sebtno/ubin/ubin-quorum-setup/binaries/QuorumNetworkManager/node\_modules/web3/lib/web3/filter.js:119:32

at Array.forEach (<anonymous>)

at Object.onMessage [as callback] (/home/sebtno/ubin/ubin-quorum-setup/binaries/QuorumNetworkManager/node\_modules/web3/lib/web3/filter.js:117:22)

at /home/sebtno/ubin/ubin-quorum-setup/binaries/QuorumNetworkManager/node\_modules/web3/lib/web3/requestmanager.js:259:20

at Array.forEach (<anonymous>)

at /home/sebtno/ubin/ubin-quorum-setup/binaries/QuorumNetworkManager/node\_modules/web3/lib/web3/requestmanager.js:258:12

at XMLHttpRequest.request.onreadystatechange (/home/sebtno/ubin/ubin-quorum-setup/binaries/QuorumNetworkManager/node\_modules/web3/lib/web3/httpprovider.js:118:13)

## Fix – use array includes

//if(!peers.contains(msg.from) && message && message.indexOf(request) >= 0){

// contains() is not a function of array

if(!peers.includes(msg.from) && message && message.indexOf(request) >= 0){else {

console.log("bye")

}

## Alternative Fix – add array prototype to the js file

Array.prototype.contains = function ( item ) {

for (i in this) {

if (this[i] == item) return true;

}

return false;

}

## Bug 2 - Cannot find module 'js-sha256'

### Fix for initStash.sh only

When running initStash.sh, you may encounter error Cannot find module 'js-sha256'.

Since sha256 is not used by the 2 js programs, just remove or comment out the “require”:

Open test-scripts/pledge.js, comment out or remove the line “var sha256 = require('js-sha256').sha256;”

Open test-scripts/setShieldedBalance.js, comment out or remove the line “var sha256 = require('js-sha256').sha256;”

### Fix for truffle unit tests

To fix the truffle errors from ubin-quorum/test-scripts that require 'js-sha256', add “js-sha256” module dependency in package.json, then re-run “npm install”.

|  |
| --- |
| "devDependencies": {  "babel-cli": "^6.10.1",  "babel-loader": "^6.2.4",  "babel-preset-env": "^1.6.0",  "eslint": "^4.8.0",  "eslint-config-airbnb-base": "^12.0.1",  "eslint-plugin-import": "^2.7.0",  "nodemon": "^1.12.1",  "solidity-sha3": "^0.4.1",  "solc": "^0.3.6",  "js-sha3": "^0.6.1",  "js-sha256": "^0.9.0",  "chalk": "^2.1.0"  }, |