



Running a Private Blockchain

Austin Hester
University of Missouri – St. Louis
01/28/18



Software Requirements

- Linux OS
- go-1.8.x+
- go-ethereum-1.7.3-stable
- Nodejs-1.8.x+
 - npm
 - truffle
 - testrpc
- Solidity compiler
- [opt] Mist browser
- [opt] VSCode w/ Solidity ext.

```
// add-apt-repository ppa:gophers/archive
```

```
// add-apt-repository ppa:ethereum/ethereum
```



go-ethereum (geth)

- What is it?
 - Geth is a command-line interface for running nodes on the Ethereum network.
- What is it used for?
 - Geth is widely used for development of **Decentralized applications** (Dapps).
 - Also used for deploying development / testing blockchains.
 - It's a decent CPU miner.
- How can we use it?
 - Running a private blockchain with custom genesis block and testing Smart Contracts.



go-ethereum

▪ What does it look like?

Mining

```
austin@austin-ubuntu-pc: ~/private-chain
File Edit View Search Terminal Help
[INFO [01-27|18:16:24] ⚡ mined potential block          number=983 hash=59d1ce...7e7418
[INFO [01-27|18:16:24] Commit new mining work          number=984 txs=0
uncles=0 elapsed=141.61µs
[INFO [01-27|18:16:31] Successfully sealed new block      number=984 hash=575c94...2f8c39
[INFO [01-27|18:16:31] ⚡ mined potential block          number=984 hash=575c94...2f8c39
[INFO [01-27|18:16:31] Commit new mining work          number=985 txs=0
uncles=0 elapsed=139.581µs
[INFO [01-27|18:16:48] Imported new chain segment      blocks=1 txs=0 mgas=0.000 elapsed=16.061ms mgasps=0.000 number=985 hash=8eb50e...34f007
[INFO [01-27|18:16:48] Commit new mining work          number=986 txs=0
uncles=0 elapsed=115.254µs
[INFO [01-27|18:16:48] ⚡ block reached canonical chain  number=986 hash=0ba96a...628460
[INFO [01-27|18:16:52] Imported new chain segment      blocks=1 txs=0 mgas=0.000 elapsed=36.328ms mgasps=0.000 number=986 hash=6a457c...d3dfea
[INFO [01-27|18:16:52] Commit new mining work          number=987 txs=0
uncles=0 elapsed=156.573µs
[INFO [01-27|18:16:54] Imported new chain segment      blocks=1 txs=0 mgas=0.000 elapsed=29.603ms mgasps=0.000 number=987 hash=5cfdfb...090bb8
[INFO [01-27|18:16:54] Commit new mining work          number=988 txs=0
uncles=0 elapsed=164.392µs
[INFO [01-27|18:16:54] ⚡ block reached canonical chain  number=988 hash=8c10ce...55e126
```

Administration

[/go-ethereum/Management-API](https://github.com/ethereum/go-ethereum/wiki/Management-API)

```
austin@austin-ubuntu-pc: ~/private-chain
File Edit View Search Terminal Tabs Help
austin@austin-ubuntu-pc: ~/private-chain x austin@austin-ubuntu-pc: ~/private-chain x
  enode: "enode://136553d6605001d4c39a7a9b990fdb62c3621d0aa77a043c04f8267fa84a591941165ced5d27b149477b287b9fa9daee00c37b9112f8d1728f69c89567441205@88:39909",
  id: "136553d6605001d4c39a7a9b990fdb62c3621d0aa77a043c04f8267fa84a591941165ced5d27b149477b287b9fa9daee00c37b9112f8d1728f69c89567441205",
  ip: [REDACTED],
  listenAddr: "[::]:39909",
  name: "Geth/bootstrap/v1.7.3-stable/linux-amd64/go1.8.3",
  ports: {
    discovery: 39909,
    listener: 39909
  },
  protocols: {
    eth: {
      difficulty: 3583986794,
      genesis: "0x21d535a377b4fc878d6d7acaf64169fc9ad1c50f9b71a25d4b550e1e984ab944",
      head: "0x469c5550cdc12ba9ce0111de528d78260090a802cd88d4e7f03497d67fada823",
      network: 9111
    }
  },
  peers: [],
  addPeer: function(),
  exportChain: function(),
  getDatadir: function(callback),
  getNodeInfo: function(callback),
  getPeers: function(callback),
  importChain: function(),
  removePeer: function(),
  sleep: function github.com/ethereum/go-ethereum/console.(*bridge).Sleep-fm(),
  sleepBlocks: function github.com/ethereum/go-ethereum/console.(*bridge).SleepBlocks-fm(),
  startRPC: function(),
  startWS: function(),
  stopRPC: function(),
  stopWS: function()
}
> admin.
```

```

1 pragma solidity ^0.4.0;
2
3
4 contract Subcurrency {
5     address public minter;
6     mapping (address => uint) public balances;
7
8     // events allow light nodes to react to changes
9     event Sent(address from, address to, uint amount);
10
11     // Constructor, run on creation
12     function Subcurrency() public {
13         // msg.sender is the address of who called a function
14         minter = msg.sender;
15     }
16
17     function mint(address receiver, uint amount) public {
18         if (msg.sender != minter) return;
19         balances[receiver] += amount;
20     }
21
22     function send(address receiver, uint amount) public {
23         if (balances[msg.sender] < amount) return;
24         balances[msg.sender] -= amount;
25         balances[receiver] += amount;
26         Sent(msg.sender, receiver, amount);
27     }
28
29     function getBalance(address _account) public view returns (uint) {
30         return balances[_account];
31     }
32
33 }

```

Solidity

“Solidity is a contract-oriented, high-level language for implementing smart contracts. It was influenced by C++, Python and JavaScript and is designed to target the Ethereum Virtual Machine.”

- Contracts exist in **contract accounts**, while user accounts are called **external accounts**.
- Contract accounts are controlled by **code**, while external accounts are controlled by public-private **key pairs**.
- Ex: Simple contract which maps a balance to an address.
 - Event Sent for watching contract.
 - Minter can mint tokens to a receiver
 - Users can send tokens to others.
 - You can also getBalance of an address.



Truffle

- What is it?
 - Truffle is a development environment for Ethereum contracts.
- What is it used for?
 - Smart contract compilation / testing
 - Contract deployment
 - Interactive console for using the contract
 - Install packages via Ethereum package manager (EthPM).
- Using `testrpc` we can deploy contracts on a temporary blockchain.



Truffle

- How can we use it?

- `$ mkdir SmartToken && cd SmartToken`
- `$ truffle init` *// initialize*
- `$ truffle compile`
- `$ truffle migrate --reset` *// deploy contract on test network*
- `$ truffle console` *// start the console*

- `truffle> SmartToken.address`
- `truffle> JSON.stringify(SmartToken.abi)`

- The above two lines retrieve the information needed to interact with your contract on the Mist browser.



Mist Browser

- Mist browser is a GUI for monitoring and manipulating your private blockchain.
- You can link to a node by opening Mist using its rpc port.
- Mist browser doubles as an Ethereum wallet.
- You can:
 - Send / receive transactions.
 - Watch smart contract activity.

Mist Browser

You can watch a contract for events by giving it the address and JSON interface of the contract.

```
austin@austin-ubuntu-pc: ~/private-chain/Projects/SmartToken
File Edit View Search Terminal Tabs Help

austin@austin-ubuntu-p... x austin@austin-ubuntu-p... x austin@austin-ubuntu-p... x

n$ truffle version
Truffle v4.0.5 (core: 4.0.5)
Solidity v0.4.18 (solc-js)
austin@austin-ubuntu-pc:~/private-chain/Projects/SmartToken$ truffle console
truffle(development)> SmartToken.address
'0xb2aa049d00f4f10fd27eafda4e79188d2686e0ba'
truffle(development)> JSON.stringify(SmartToken.abi)
'[{{"constant":false,"inputs":[{"name":"recipient","type":"address"},{"name":"value","type":"uint256"}],"name":"depositToken","outputs":[{"name":"success","type":"bool"}],"payable":false,"stateMutability":"nonpayable","type":"function"},{"constant":true,"inputs":[{"name":"recipient","type":"address"}],"name":"getTokens","outputs":[{"name":"value","type":"uint256"}],"payable":false,"stateMutability":"view","type":"function"},{"constant":false,"inputs":[{"name":"recipient","type":"address"},{"name":"value","type":"uint256"}],"name":"withdrawToken","outputs":[{"name":"success","type":"bool"}],"payable":false,"stateMutability":"nonpayable","type":"function"},{"constant":true,"inputs":[{"name":"value","type":"uint256"}],"name":"add12","outputs":[{"name":"newvalue","type":"uint256"}],"payable":false,"stateMutability":"pure","type":"function"},{"anonymous":false,"inputs":[{"indexed":true,"name":"_from","type":"address"},{"indexed":false,"name":"_value","type":"uint256"}],"name":"OnValueChanged","type":"event"}]'
truffle(development)>
```

Mist Browser interface showing the "Watch contract" dialog box.

The dialog box displays the following information:

- CONTRACT ADDRESS:** 0xb2aa049d00f4f10fd27eafda4e79188d2686e0ba
- CONTRACT NAME:** SmartToken
- JSON INTERFACE:** A JSON array of contract functions and events, including `depositToken`, `getTokens`, `withdrawToken`, `add12`, and `OnValueChanged`.

Buttons: CANCEL, OK

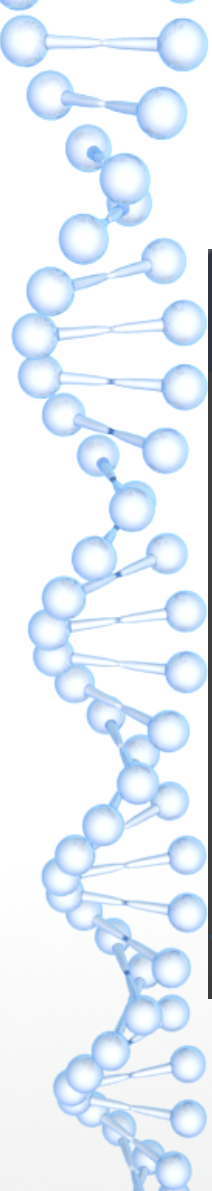
Footer: <https://github.com/ethereum/mist>

My Scripts for geth

- `generate_genesis.sh` // generates `genesis.json` for initialization (recommend changes)
- `initialize_chain.sh` // initialize blockchain with generated genesis block
- `create_account.sh` // create 2 accounts for each node
- `unlock_on_start.sh` // create `passwd.sec` in `datadirs`
- `get_enodes.sh` & `copy_static_nodes.sh` // retrieve `enodes`, required for syncing nodes
- `runbootstrap.sh` & `startminer[1,2].sh` // for starting each
- `attach_nodes.sh` // attach to node and open console
- `kill_chain.sh` // kill all instances of `geth`

```
austin@austin-ubuntu-pc: ~/private-chain
File Edit View Search Terminal Help
austin@austin-ubuntu-pc:~/private-chain$ ls
attach_nodes.sh      generate_genesis.sh  miner1      startminer1.sh
bootstrap            genesis.json         miner2      startminer2.sh
copy_static_nodes.sh get_enodes.sh        Projects    unlock_on_start.sh
create_account.sh    initialize_chain.sh  README.md
enode                kill_chain.sh        runbootstrap.sh
austin@austin-ubuntu-pc:~/private-chain$
```

runbootstrap.sh



```
austin@austin-ubuntu-pc: ~/private-chain
File Edit View Search Terminal Help
1 # !/bin/bash
2
3 CURR=`dirname $0`
4 if [ -d $CURR/bootstrap/geth ]; then
5 geth --identity "bootstrap" --networkid 9111 \
6     --datadir $CURR/bootstrap \
7     --ipcpath /home/$USER/.ethereum/geth.ipc \
8     --nodiscover --port "39909" --rpc --rpcport "42024" \
9     --rpcapi "db,eth,web3,net,personal,miner" \
10    --unlock 0 --password $CURR/bootstrap/passwd.sec
11 else
12     echo "Initialize first"
13 fi
~
"runbootstrap.sh" 13L, 394C 1,1 All
```

This script starts the `bootstrap` node. Avoids typing all this every time.



Raspberry Pi as Node

- I was successfully able to use a Raspberry Pi 3 as a light node, which syncs with the miners running on my PC.
- On the Pi:
 - Clone the repo: <https://github.com/ahester57/private-chain.git>
 - `$ sftp <to PC>; > get genesis.json`
 - `$ geth --datadir ./pinode0 init genesis.json`
 - Create accounts and run to get its "enode://..." address
 - Save "enode://..." to <file>
 - `$ sftp <to PC>; > put <file>`
 - Restart the Pi node, it will now wait until synced with our blockchain.



Raspberry Pi as Node

- On PC:

- `./attach_nodes.sh // choose 'b'`
- `> admin.addPeer("enode://..@<pi_ip>:<pi_port>");`
 - Should return `true`

- Wait a short while, and check it is connected using

- `> admin.peers` while attached to bootstrap node.
- The `geth` instance running on the Pi should start importing **chain segments** once connected to our blockchain.
- We can add its `enode` address to `static-nodes.json` to do this automatically on restart.



Resources

- <https://github.com/ethereum/wiki/wiki>
- <https://github.com/ethereum/go-ethereum>
- <https://github.com/ahester57/private-chain>
- <https://ethereum.org/token>
- <https://solidity.readthedocs.io/en/develop/>
- http://truffleframework.com/docs/getting_started/project
- <https://github.com/ethereum/mist/wiki>