Blockchain on AWS

TOORAJ HELMI

9/26/2018

Agenda

- INTRODUCTION TO BLOCKCHAIN
- PRINCIPLES OF DISTRUSTED TRUST
- BLOCKCHAIN PROJECTS
- SMART CONTRACTS
- INTRODUCTION TO ETHEREUM
- DEVELOPMENT TOOLS
- DEMO

What is Blockchain

Tech Guys

Distributed → Peer-to-Peer Network (RPLx)

+

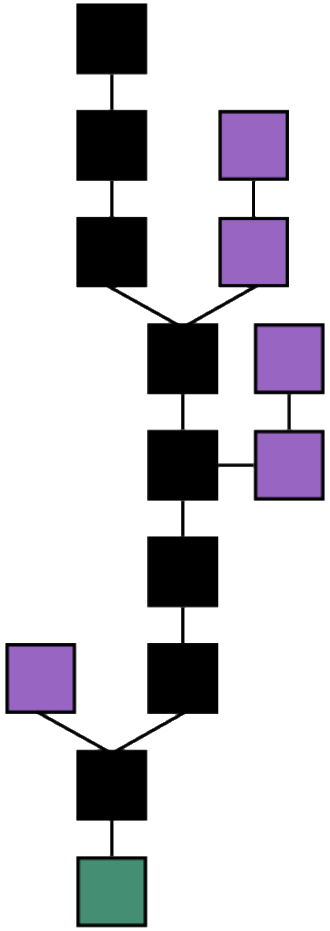
Immutable, Verifiable → Replicated Signed Storage & Code

+

Smart Contracts → Sandboxed Execution Environment

+

Mining → Automatic Transaction Approvers



Business Guy

- Democratized Bank

OR

- Distributed Economy

OR

- Distributed Immutable Records

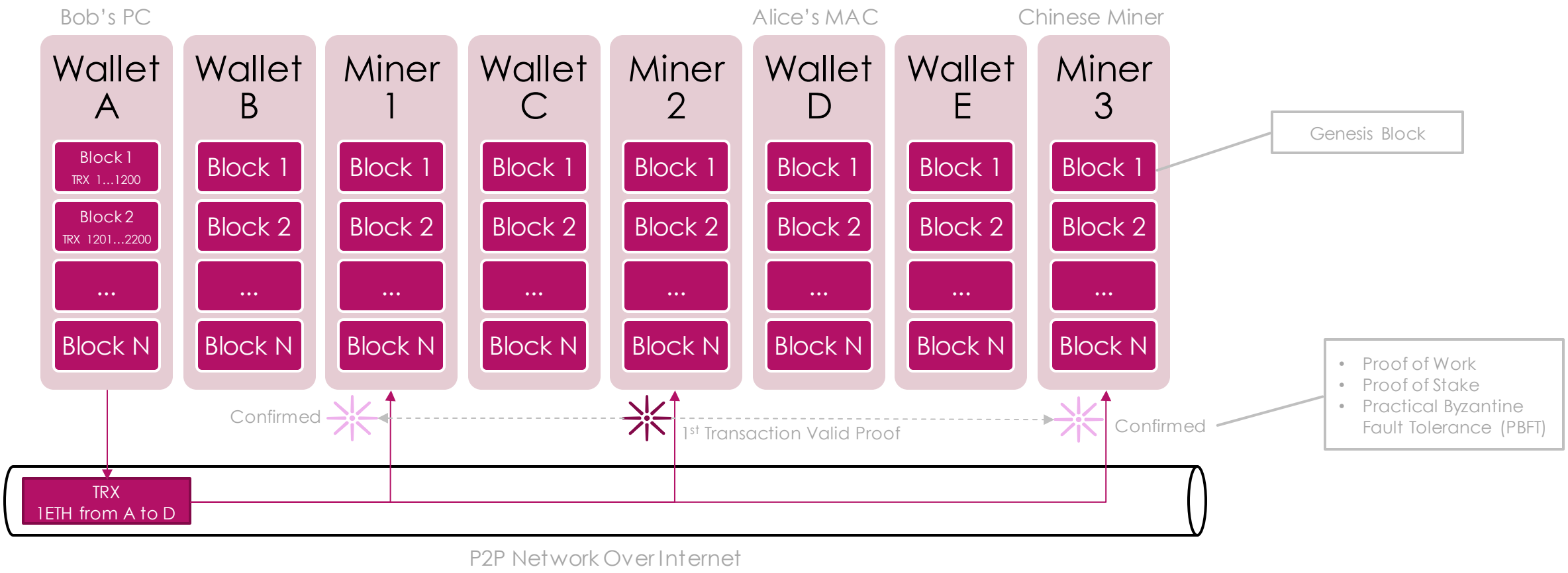
Why Blockchain

- Eliminates cost of trust

- Eliminates unbalanced level of ownership

- Bypassing stringent investing regulations

Blockchain Construction



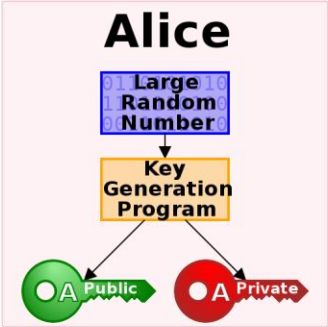
Principles of Distributed Trust

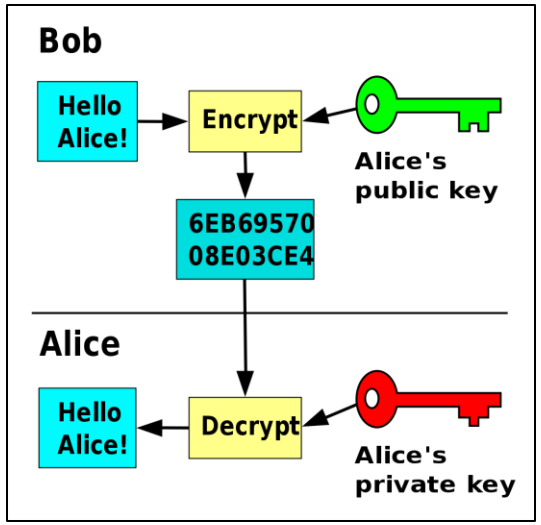
The major reason blockchain has become so popular is that it can be trusted to provide the truth – i.e. which transaction has been issued by who – and stay truthful without requiring a central authority. This is what we call Distributed Trust which is implemented based on two principles:

- VERIFYABILITY
- IMMUTABILITY

Principle 1: Verifiability

- Signing data using public (asymmetric) key cryptography





Singing Using	Visible To	Verifiable By	Use Case
Receiver's Public Key	Just the receiver	Just the receiver	Encrypted Email
Sender's Private Key	Everybody	Everybody	Blockchain

Sending a Transaction

- Transaction Message:

Sender Address	Receiver Address	Data	Nonce	Signature

- Addresses
  - Are public keys that are generated for an account when a new account is created
  - The private key is generated simultaneously that should be kept secured by the account owner
- To send a transaction
  - The sender assigns a randomly-generated integer - called nonce – to the message
  - Generate a digest from first four field
  - Signs the digest using his private key and includes it as the signature in the message

Verifying a Transaction, Positive Case

From	Bob PUB Key
To	Alice PUB Key
Data	5 Ξ
Nonce	2
Signature	34FA5 USIGN Bob Pri Key

DIGEST = 4A5E1

Bob

From	Bob PUB Key
To	Alice PUB Key
Data	5 Ξ
Nonce	2
Signature	34FA5

Calc Digest

DIGEST = 4A5E1

Decrypt Signature Using Bob's Pub Key = 4A5E1

Are They Equal?

Add to chain

Miner

Good boy James

Verifying a Transaction, Negative Case

From	Bob PUB Key
To	Alice PUB Key
Data	5 Ξ
Nonce	2
Signature	34FA5 USIGN Bob Pri Key

DIGEST = 4A5E1

Bob

From	Bob PUB Key
To	James PUB Key
Data	5 Ξ
Nonce	2
Signature	34FA5

Malicious boy James

From	Bob PUB Key
To	James PUB Key
Data	5 Ξ
Nonce	2
Signature	34FA5 USIGN James Pri Key

Miner

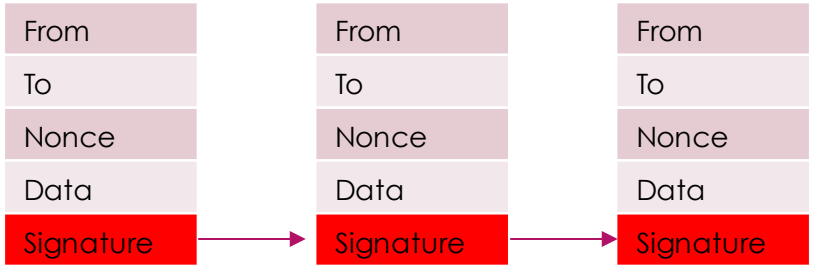
Calc Digest

DIGEST = 223FA

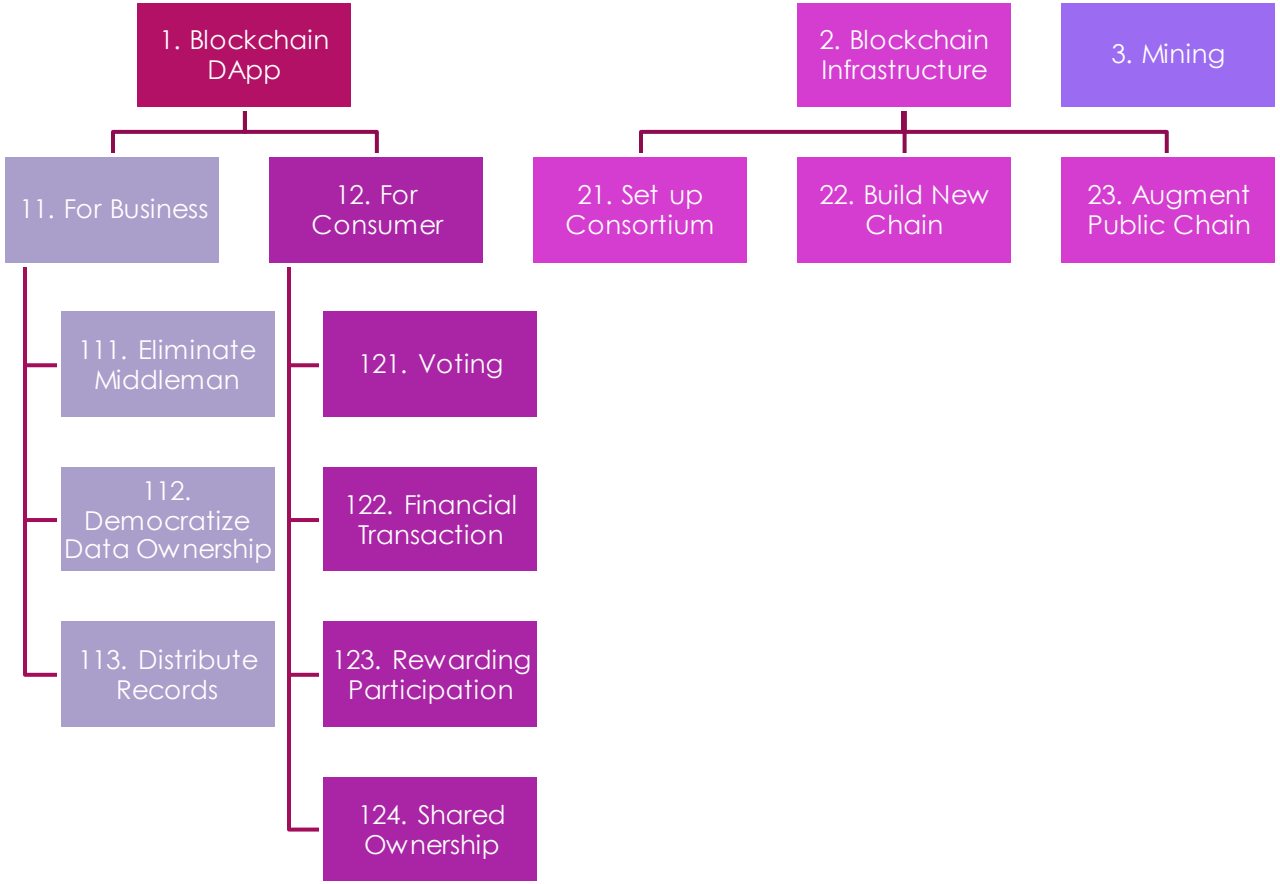
Decrypt Signature Using Bob's Pub Key = 4A5E1

Principle 2: Immutability

- Replication
  - Keep the entire replica of the blockchain on all miners nodes
- Chaining
  - Create an end-to-end dependency where changing any single piece of data invalidates the entire blockchain. This is done by including the signature of the previous block in calculating the digest of the next block.
- In order to manipulate a single transaction, an attacker has to change the entire blockchain on at least 50% of the miner nodes



Blockchain Application Classification



Notable Blockchain Projects

For Consumers

- Idex (Decentralized Exchange)
- Rentberry (Global rental platform)
- FairWin (Gambling platform)
- EtherSport (Sport betting)
- Tap Project (Gaming reward platform)
- Choon (Music streaming service)

For Enterprises

- Walmart tracking food source on IBM blockchain
- Toyota Research Institute: Car Sharing
- Icertis: Enterprise Contact Management
- UBS, Barclays, SIX, Credit Suisse, KBC: Massive Autonomous Distributed Reconciliation Platform' aka Madrec

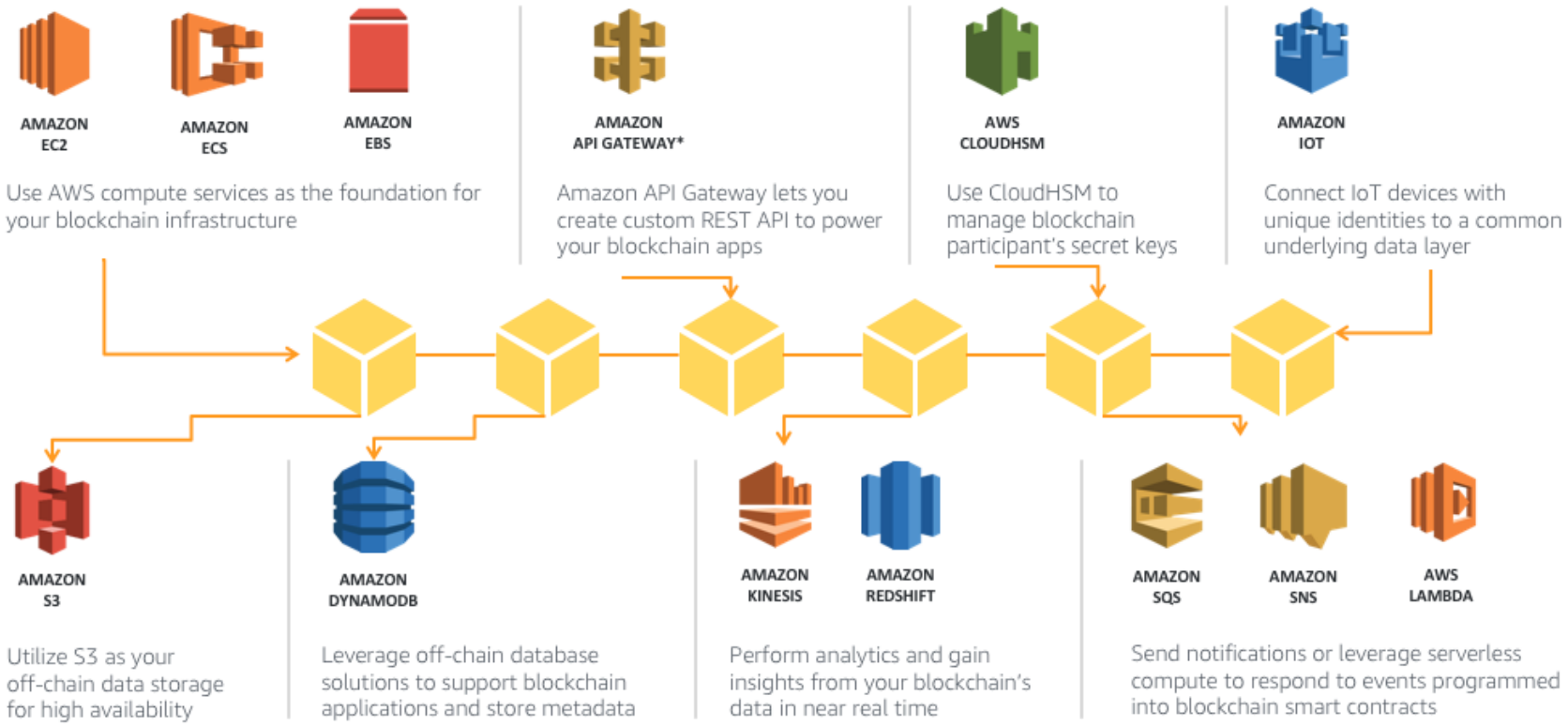
ICO: Utility vs Security Token

- Utility token is used to compensate for a service offered or product sold to the consumer
  - Filecoin–which raised an ICO-record $257 million–plans to provide a decentralized cloud storage service that will take advantage of unused computer hard drive space
  - Interchains.io: a sharable economy on personal computers.
- Security Token is solely used to raised funds
  - Less stringent than IPO to get on yet still taxable (SEV vs Howey).
    - Ethereum (ROI of 152,500%)
    - NEO (ROI of 118,000%)
    - Ark (ROI of 34,500%)

Blockchain on Cloud! Really?

- Remember the principle of immutability.
  - Blockchain distributed replica of blockchain on all participant nodes to minimize the risk of 51% attack.
- When blockchain in deployed on a cloud then the subscriber owner of the cloud has full control on the blockchain!
- So why would it make any sense to host a blockchain on the cloud?
  - Blockchain Provider: Auditing, Legal, …
  - Internal Audit System
  - Mining?

AWS Blockchain Ecosystem



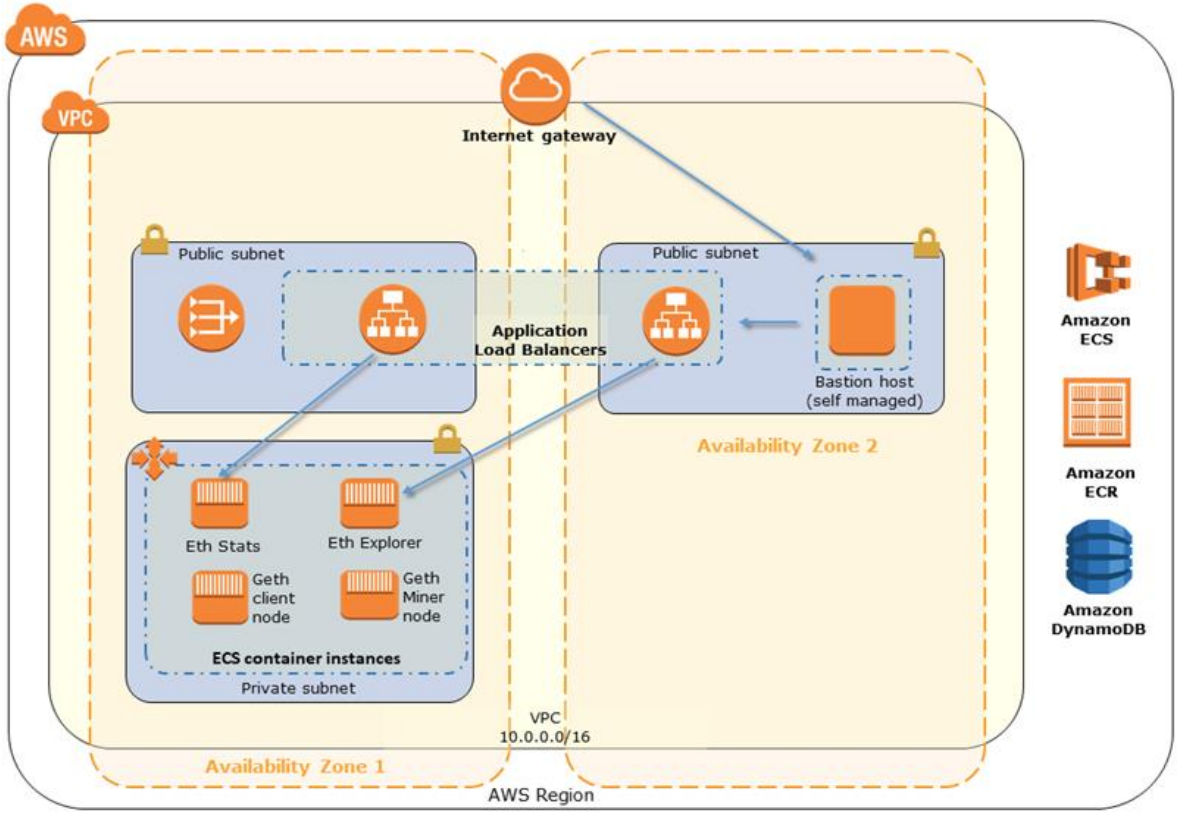
What is Ethereum

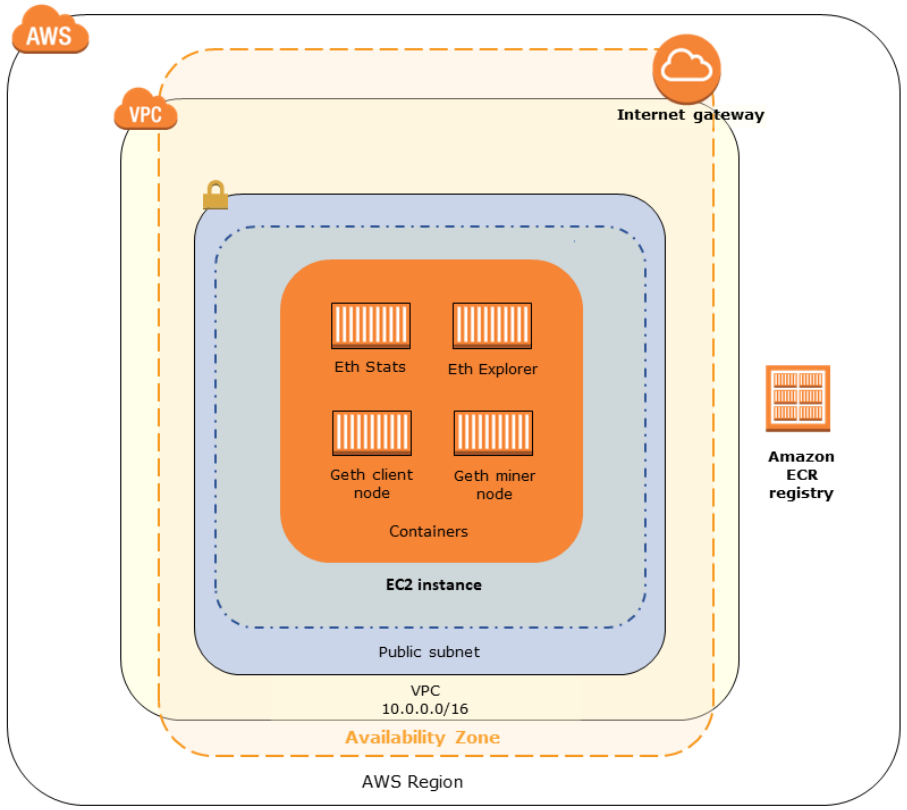
- Open source
- Public blockchain
- Featuring smart contract functionality
- It supports a modified version of Satoshi Nakmoto (PoW)

What is Hyperledger Fabric?

- Private, opensource blockchain supported by IBM.
- Allows components, such as consensus and membership services, to be plug-and play.
- It leverages container technology to host smart contracts called "chaincode" that contain the business rules of the system.
- It uses certificate authority (CA) to provide a permissioned access to the ledger.
- It uses models, participants, and transactions for programming smart contracts.

DEMO 1 – Setup Ethereum on AWS Network Architecture





DEMO 1 – Setup Ethereum on AWS Steps

1: Wei
!0^6: MWei
10^9: GWei
10^12: Szabo
10^15: Finney
10^18: Ether (Ξ)

- Pre-req
  - Create a VPC and Subnets
  - Create Security Groups
  - Create an IAM Role for Amazon ECS and an EC2 Instance Profile
  - Create a Bastion Host
- Create a Private Blockchain
  - Use this template
- Post Deployment
  - Add browser proxy

Field	Values
Fees	Target Block Gas Limit = 8MWEI
Accounts	0x0ADfCCa4B2a1132F82488546AcA086D7E24EA324, 0x0bd5EebDC3E53973dDF236D43906C776a5fE3784, 0x9537cb86f5a03C8CCB52c44b49757861eCA0004b, 0x1Fbc353788338F902630E5494aD7FaC7dF8dBb29, 0x5ccBe3B9B15eFB62bB2696051091Ee7C1Eb4c7E6
Initial Account Balance	1000 ETH, What does this mean?
Miner Account Address	0x0ADfCCa4B2a1132F82488546AcA086D7E24EA324

DEMO 1 – Setup Ethereum on AWS Completion Result

aws Services Resource Groups toorajhelmi Ohio Support

CloudFormation Stacks › Stack Detail

MessageBankNet

Other Actions Update Stack

Stack name: MessageBankNet

Stack ID: arn:aws:cloudformation:us-east-2:713774468243:stack/MessageBankNet/789f0c20-b7ab-11e8-a3a9-0658482cfd4e

Status: CREATE_COMPLETE

Status reason:

Termination protection: Disabled

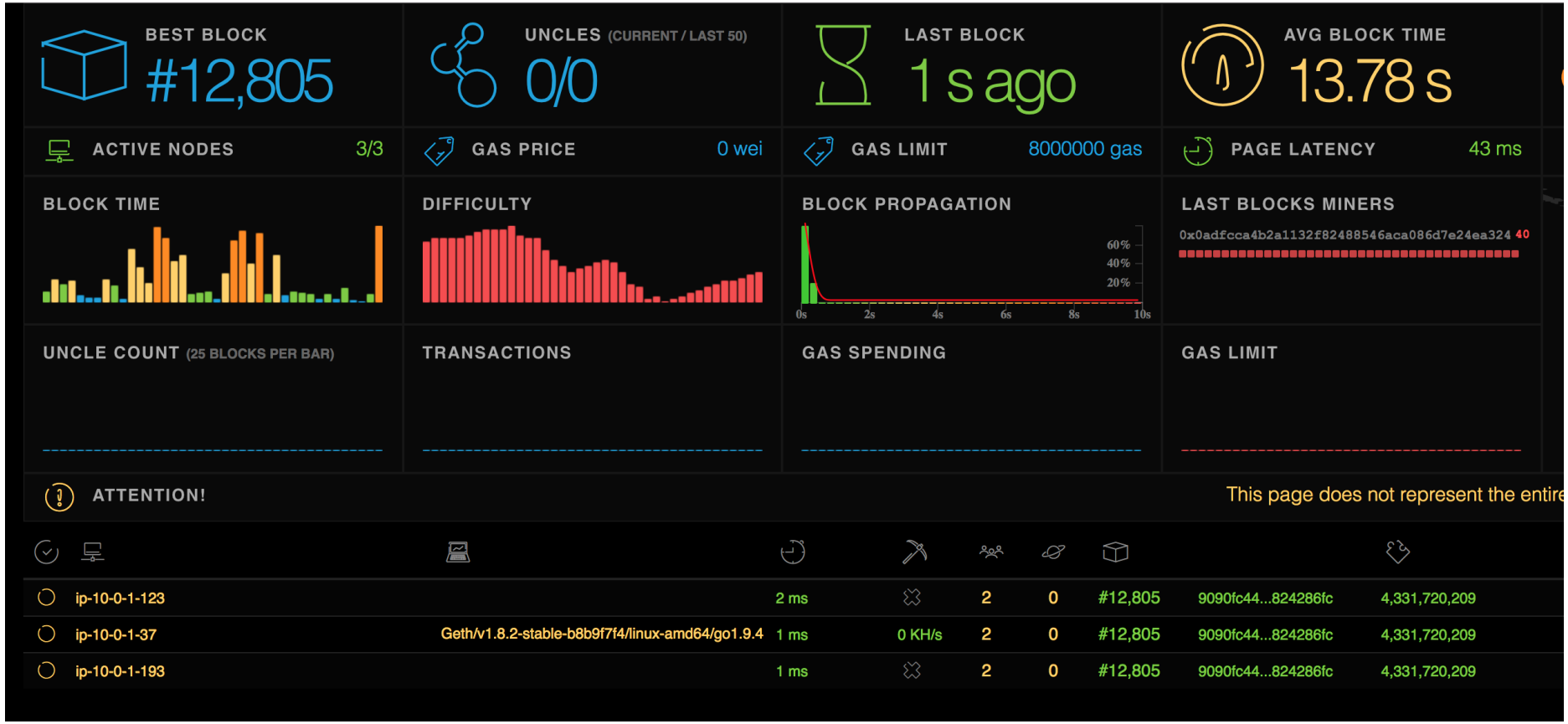
IAM role:

Description This template creates an Ethereum network on an Amazon ECS cluster or Amazon EC2 instance. It also includes the Ethereum Network Stats and Ethereum Explorer services. Additional components may be deployed depending on the container platform chosen. License: Apache 2.0 (Please do not remove) Apr 4, 2018 (bt-f5kb4kx9v)

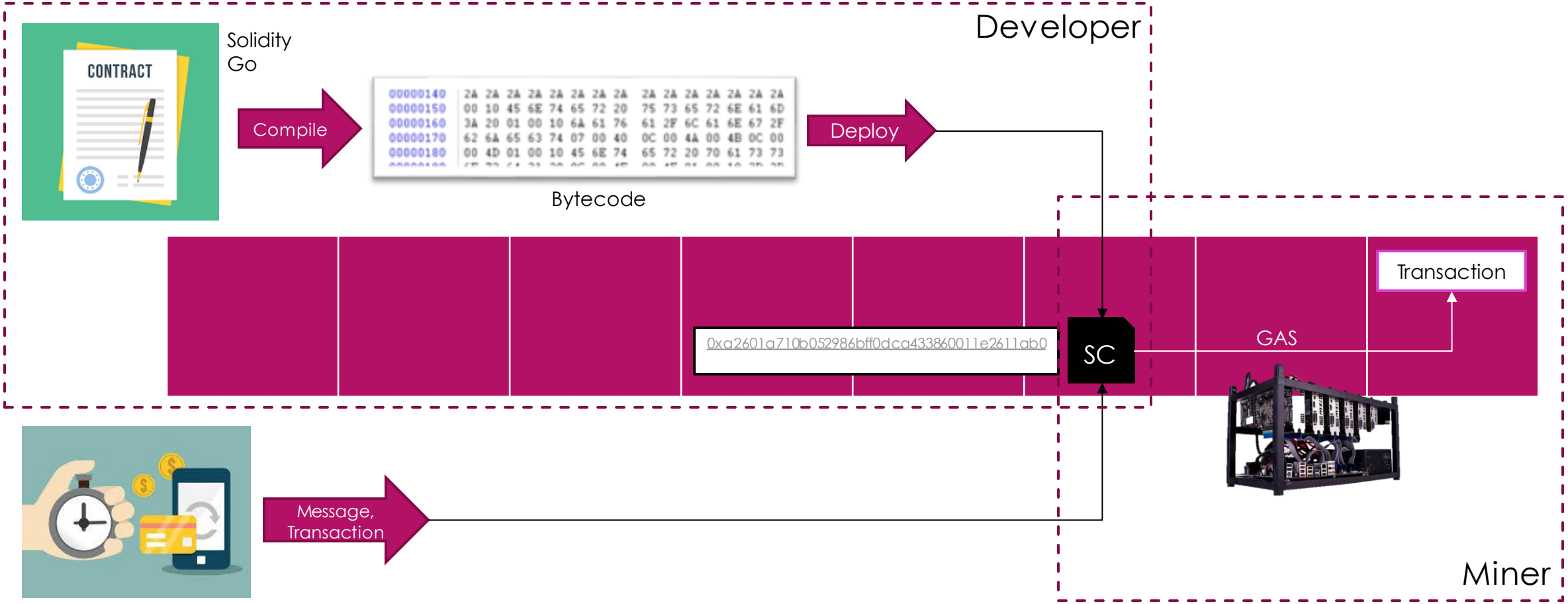
Outputs

Key	Value	Description	Export Name
EthStatsURL	http://internal-Messa-LoadB-1HYBHCHIBVG97-1662950222.us-east-2.elb.amazonaws.com	Visit this URL to see the status of your Ethereum...	
EthExplorerURL	http://internal-Messa-LoadB-1HYBHCHIBVG97-1662950222.us-east-2.elb.amazonaws.com:8080	Visit this URL to view transactions on your Ether...	
EthJsonRPCURL	http://internal-Messa-LoadB-1HYBHCHIBVG97-1662950222.us-east-2.elb.amazonaws.com:8545	Use this URL to access the Geth JSON RPC of ...	

DEMO 1 – Setup Ethereum on AWS Eth Stat



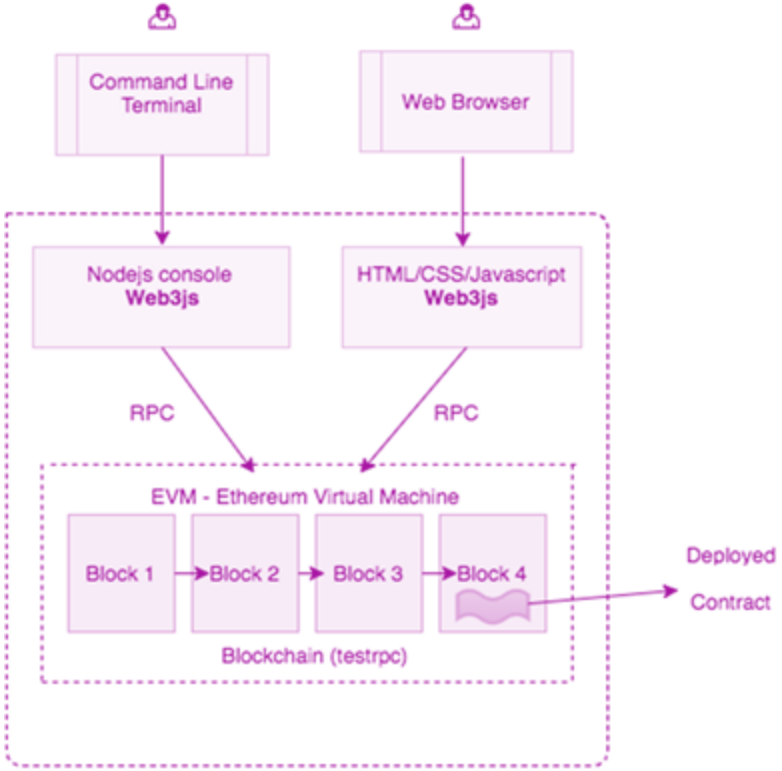
Smart Contracts



How Ethereum Works

https://ethereum.github.io/yellowpaper/paper.pdf

- Ethereum Virtual Machine (EVM)
  - AKA Wallet is installed on each mining node
  - Implements a complete deterministic turning machine
  - Runs the contracts in response to transactions using its state machine
  - Holds storage per account (256B), stack (8K), and memory
- Accounts
  - Every account has a balance, a nonce, bytecode, and the root hash of a storage tree
  - External Accounts vs Contract Accounts
- State Machine
  - Updates account balances and nonces
  - Handles gas and gas refunds
  - Executes EVM byte code (which can cause account balances and storage values to change)
  - Pays miners for mining blocks



Security of Smart Contract

- Re-Entrancy (How DAO attack happened)

```
pragma solidity ^0.4.0;
contract Fund
{
mapping(address => uint) shares;
function withdraw() public
{
  If (msg.sender.send(shares[msg.sender]))
     shares[msg.sender] = 0;
}
```

```
contract Fund
{
contract. mapping(address => uint) shares;
function withdraw() public
{
  var share = shares[msg.sender];
  shares[msg.sender] = 0;
  msg.sender.transfer(share);
}
```

- Private Information
- Call stack Depth (1024 instrcutions)

Development Tools

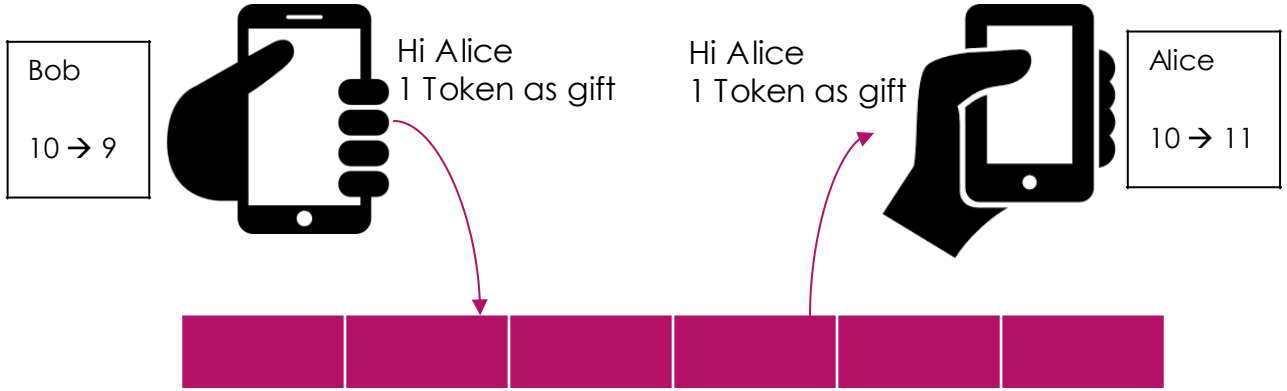
- Tools, libraries, and Simulators
  - VS Code, Remix
  - TESTRPC to simulate local Ethereum network
  - Truffle sets of tools to compile, unit test, and deploy Solidity smart contract
  - Metamask: Chrome Extension letting HTML DApp talk to Ethereum
  - Zeppelin: Provides implementations of standards like ERC20 and ERC721
  - Infura: Rest API

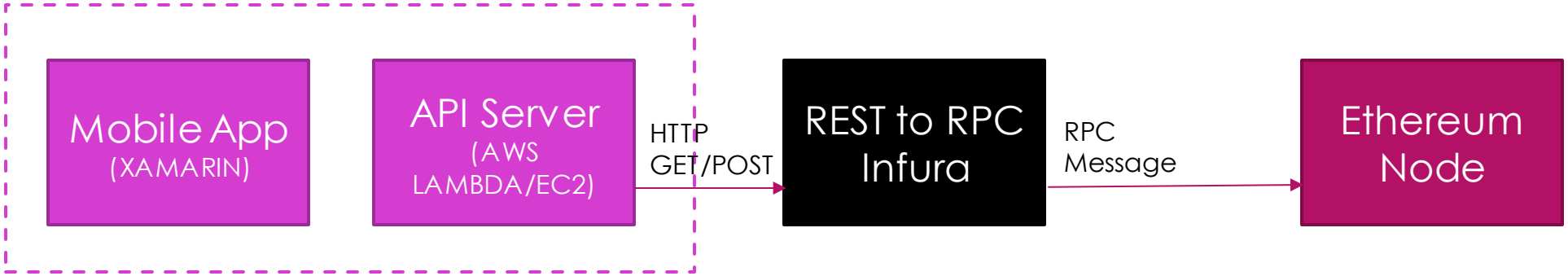
- Minimum Setup to develop locally
  - Install VS Code
  - Install Node.js
  - Install testrpc: npm install -g ethereumjs-testrpc
  - Install Truffle: npm install -g truffle

- Documentation:
  - Ethereum Concepts, Homestead
  - Solidity

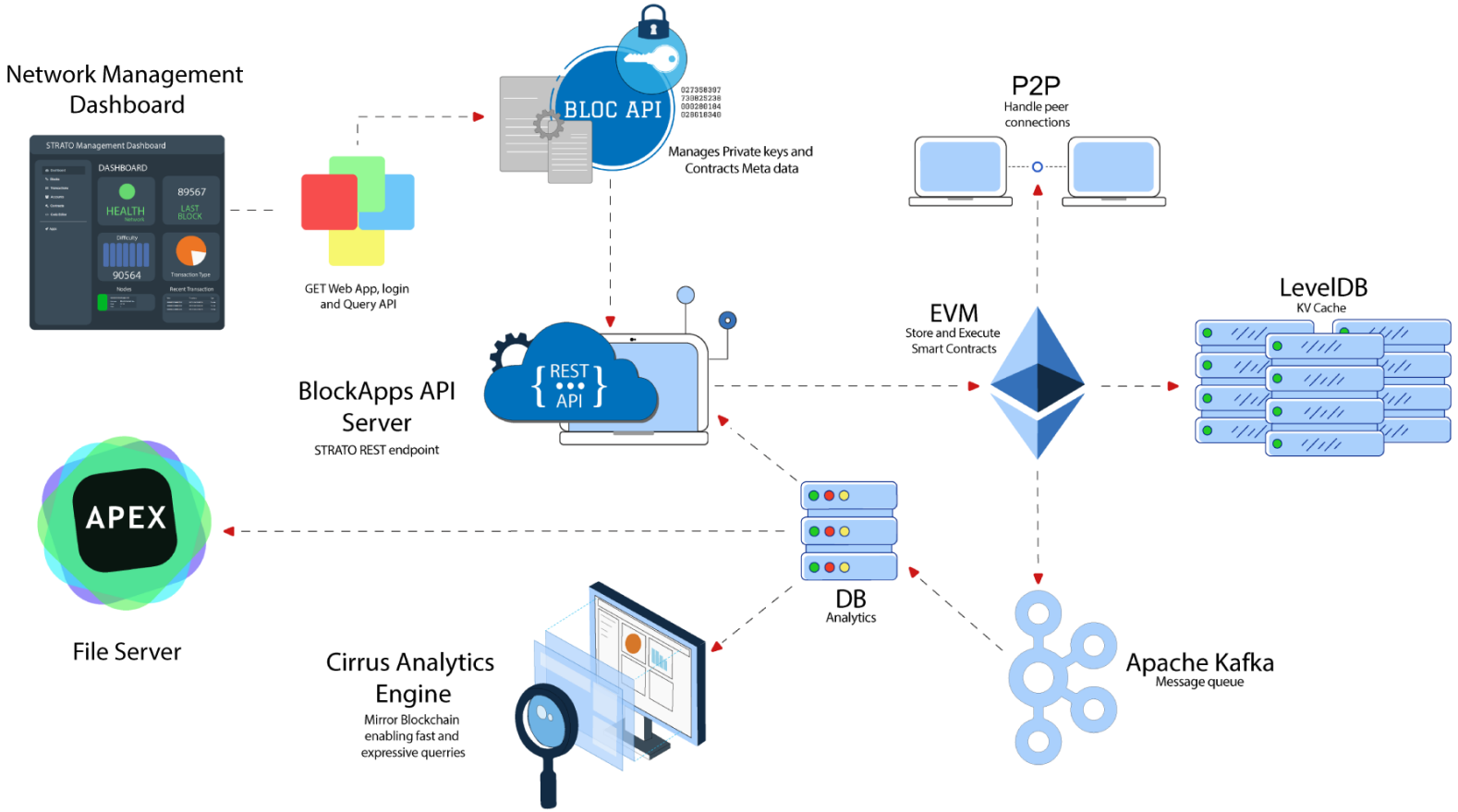
DEMO 2 – Develop a DAPP

- Send tokens with messages
  - UI: Cross platform mobile application
  - Backend: Ethereum Blockchain





Another Option: Block Apps STRATO



MBN Demo Step 1: Setup Ethereum

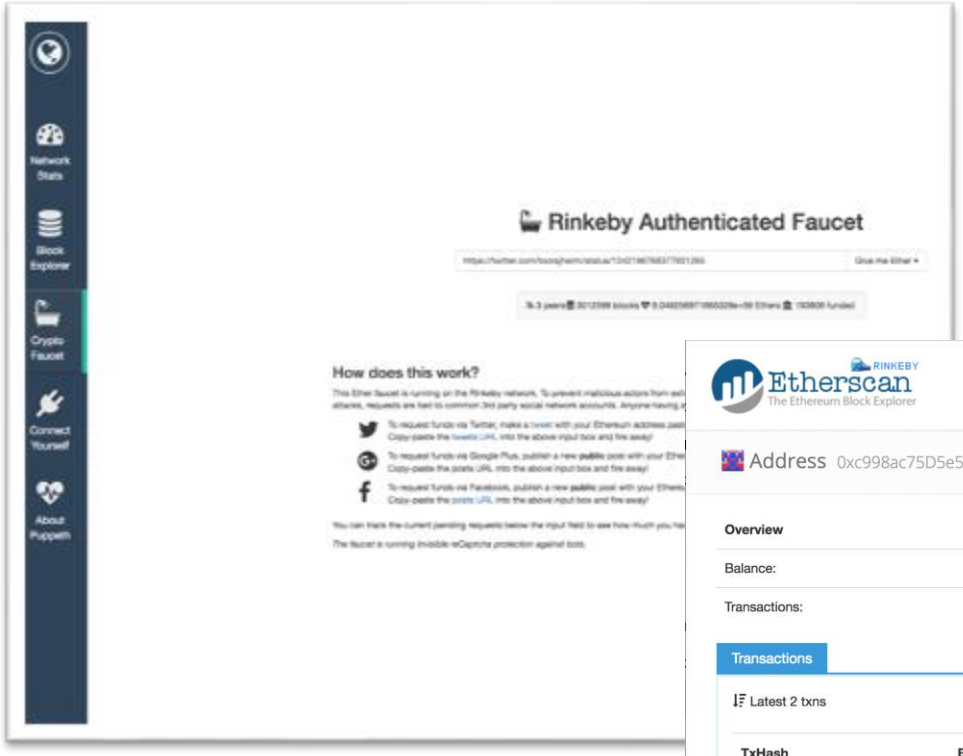
- Install Ethereum locally (rinkeby test net)

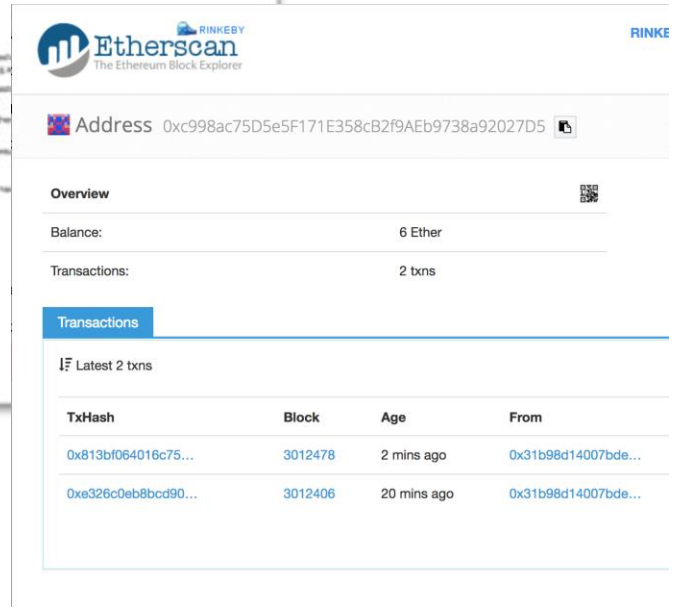
  brew install Ethereum

  brew install geth

  geth --rinkeby

  - Attach to rinkeby geth --datadir=$HOME/.rinkeby --syncmode=light attach ipc:$HOME/Library/Ethereum/rinkeby/geth.ipc console
  - Create new account:
    - personal.newAccount()
    - personal, eth.coinbase, eth.getBalance(eth.coinbase)
    - Keystore: /Users/ToorajHelmi/Library/Ethereum/rinkeby
  - Add ETH: https://www.rinkeby.io/#faucet
  - Check balance: https://rinkeby.etherscan.io/address/0xc998ac75d5e5f171e358cb2f9aeb9738a92027d5

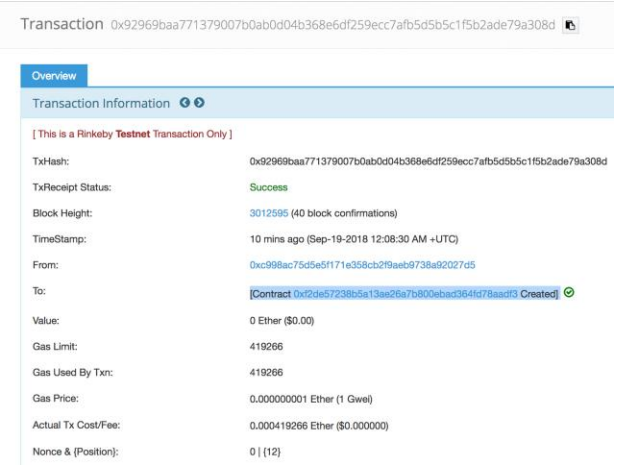




MBN Demo Step 2: Write, Compile & Deploy Smart Contract

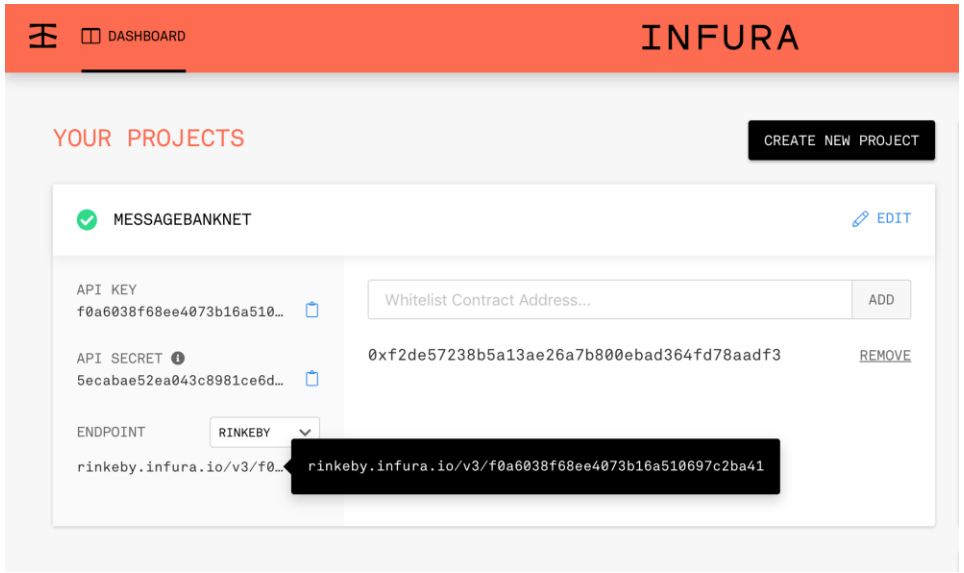
```
pragma solidity ^0.4.21;

contract MessageBankNet {
    address public minter;
    mapping (address => uint) public balances;
    mapping (address => string) public messages;
    event Sent(address from, address to, uint amount, string message);

    constructor() public {
        minter = msg.sender;
    }
    function mint(address receiver, uint amount) public {
        if(msg.sender != minter) return;

        balances[receiver] += amount;
    }
    function send(address receiver, uint amount, string message)
public {
        if(balances[msg.sender] < amount) return;

        balances[msg.sender] -= amount;
        balances[receiver] += amount;
        messages[receiver] = message;
        emit Sent(msg.sender, receiver, amount, message);
    }
}
```

- Compile in Remix
- Deploy to Rinkeby Test Network
  - Confirm in Metamask
- Get contract address from Metamask



MBN Demo Step 3: Connect to Mobile App

- Add REST API endpoint in Infura
- Build the API server using Node
  - npm init (package name should be app.js_
  - npm install -save express body-parser cjson web3
  - API Routing: token-route.js
  - API Wrapper: token.js
  - Use https://ethtools.com/mainnet/wallet/load/ to get private key used in token.js



Node api GET/POST → W3.Contract → Infura REST endpoint → Blockchain

MBN Demo Step 4: Test the Contract

- Test locally: node app
- Use POSTMAN to test:
  - mintToken: http://localhost:8080/token/mintToken


```
{
        "address": "0xc998ac75d5e5f171e358cb2f9aeb9738a92027d5",
        "tokens": 1,
}
```


  - Get Balance: http://localhost:8080/token/getBalance?address=0xc998ac75d5e5f171e358cb2f9aeb9738a92027d5


```
{
        "Ether Balance": "8.999536992",
        "Token Balance": "1"
}
```


- sendToken: http://localhost:8080/token/sendToken


```
{
        "address": "0xf9880966388914467dc58e95d69e265e7586d4d5",
        "tokens": 1,
        "message": "Hello"
}
```


- Get Balance: 0xc998ac75d5e5f171e358cb2f9aeb9738a92027d5


```
{
        "Ether Balance": "8.999499054",
        "Token Balance": "0"
}
```


- Get Balance: 0xf9880966388914467dc58e95d69e265e7586d4d5


```
{
  "Ether Balance": "0",
  "Token Balance": "1"
}
```

MBN Demo - Step 5: Deploy to AWS

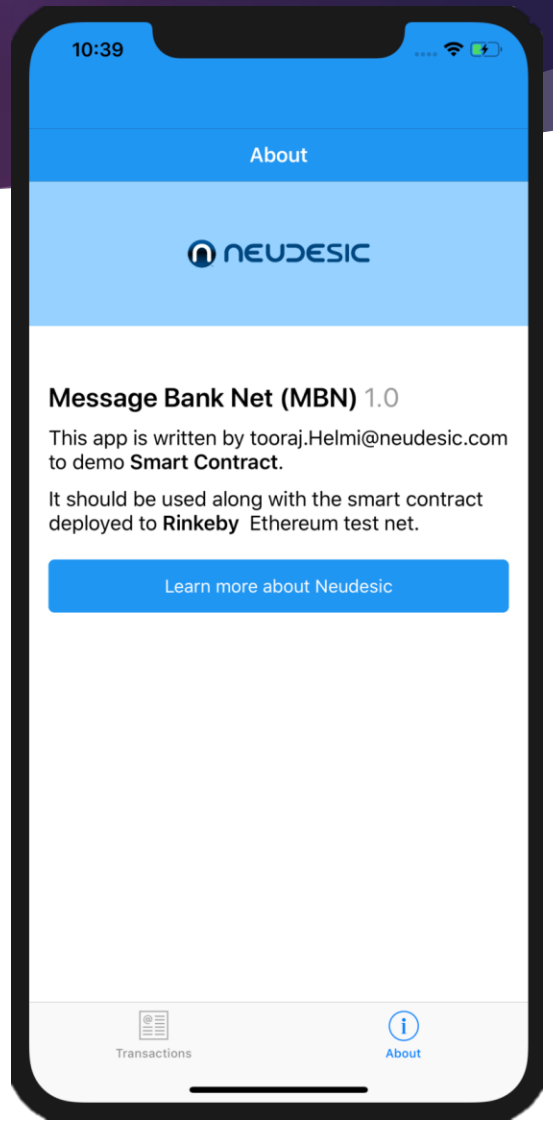
- Set up EC2:
  - Create a Linux instance
  - Install node and python
  - Make sure to open 8080 from any source in SG
  - Run node app

Python:
1. Install GCC: #yum install gcc
2. Download Python: cd /usr/src
   # wget https://www.python.org/ftp/python/2.7.10/Python-2.7.10.tgz
3. Extract Archive and Compile
   # tar xzf Python-2.7.10.tgz
   # cd Python-2.7.10
   # ./configure
   # make altinstall
4. Check Version: # python2.7 –V

Node:
1. Install Updates: sudo yum update –y
2. curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh | bash
3. nvm install 8.10
4. Install node-gyp globally (npm install -g node-gyp
5. Install development tools (sudo yum groupinstall "Development Tools" (Needed by scrypt)

MBN Demo - Step 6: Connect the Mobile App

- Run the node app on EC2
- Use a HttpClient to send GET and POST requests to the API



Advanced Topics

- Cross-blockchain communication
- Oracles
- Versioning
- Distributed smart contracts
- Resiliency vs performance tradeoff
- Transferring ETH

“Tell me and I forget.
Teach me and I remember.
Involve me and I learn.”

Benjamin Franklin

Please ask your questions or provide your feedback.

Thanks!