mtech-bitcoin-elective

lecture 1

course code: 20cy712
text: mastering blockchain by imran basheer yr 2017
⇒ understanding blockchain basics

syllabus

1) introduction
• who created blockchain, who did what etc
2) specific concepts
3) bitcoin
1- architecture
2- crytpocurrency
3- challenges
4- transfer method
5- mining
5) ethereum(30 to 40 %, hands on lab etc)
6) introduction to hyperledger and some projects hands on
7) consesus
1- what is it
2- types
3- challenges
8) different use cases of blockchain
1- present and future of blockchain use cases
9) challenges
1- present and future challenges
2- realtime
3- compliance
4- legal

10) supporting tools
1- improving costs, ux
2- helping to solve challenges easier

introduction

• bitcoin and blockchain are not the same
◇ bitcoin is the first successful commercial application of blockchain
• blockchain is defined in two ways
◇ blockchain ledger (datastructure)
■ blockchain is consisting of many blocks
- each block is having transactions
■ every block is linked to each other via crypto hash
◇ technology
what
■ decentralized computation(workers), distributed ledger (data which is worked on by workers available to all workers) platform
- decision is made by some handful of computers
why
■ to immutably store transactions in a verifiable manner
how the why
■ through a rational decision making process (consensus) by multiple parties in open and public system in a efficient manner
• apply blockchain in multiple parties usecase, not for less number of people.

• characteristics of blockchain
◇ transparent

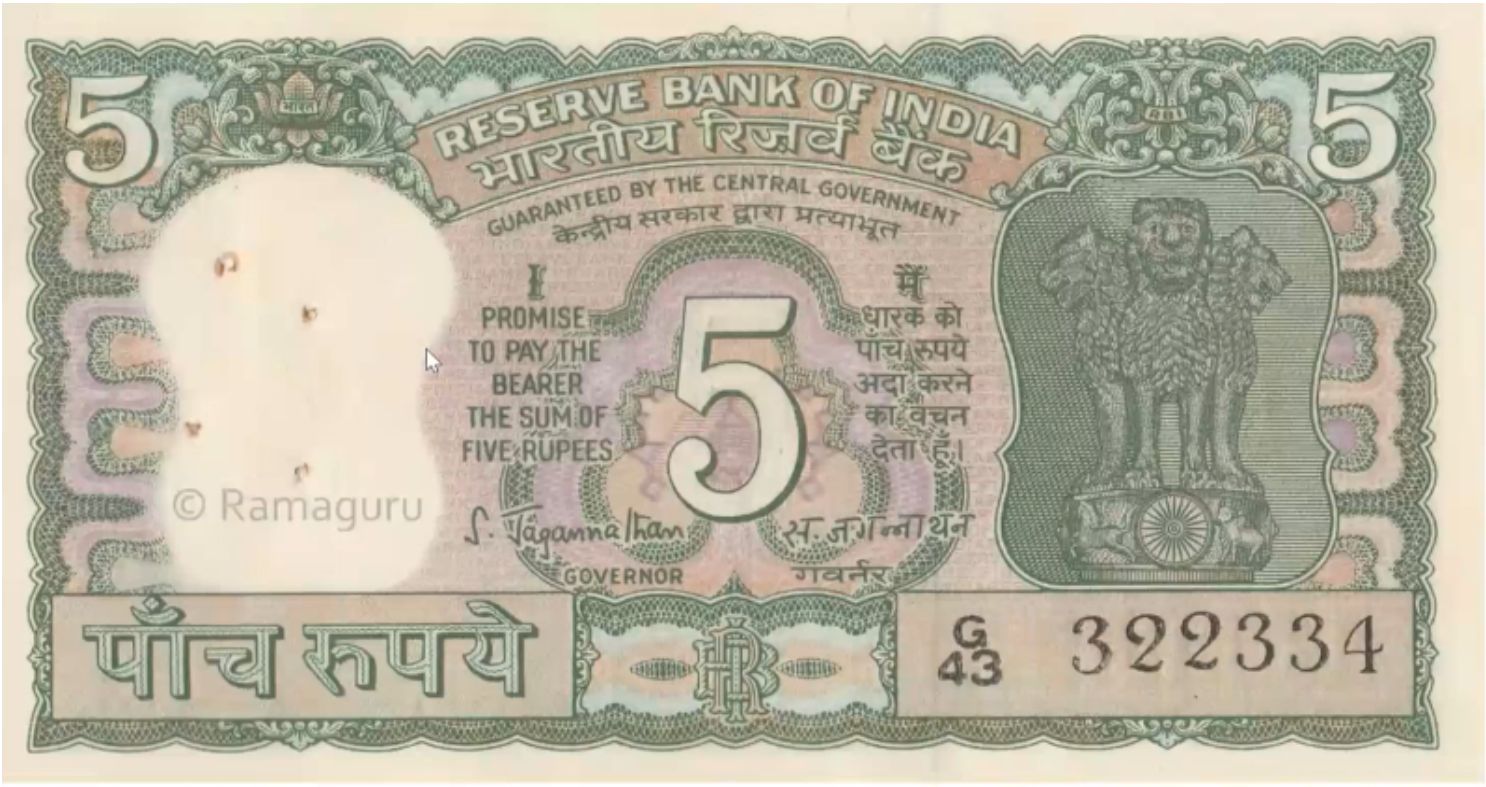
- distributed
- decentralized
- immutable
- consensus

lecture 2

why blockchain?

- there is no promissory note
- There is no reserve bank of india, there is only government of india
- Signed by finance secretary under ministry of finance
  - only note which is having no promissory note and signed by ministry of finance
- there is a coin image
- when ever 1 rupee note is issued, it is issued like this,

substitute for one rupee coin
  ◇ hence the coin image
  ◇ coins wont have promissory note
  ◇ rupee note issued to reduce metal consumption required to mint the actual coin



• “I promise to pay the bearer the sum of five rupees” → promissory note by Governor of RBI
  ◇ can only be given by RBI and only be guaranteed by central government
  ◇ basically saying RBI is viable to pay the owner of the note, gold/services worth Rs 5
• The paper having five rupees worth is guaranteed by central government
• The paper is issued by reserve bank of india

commodity money
• value is there because of material has intrinsic value
  ◇ ie laptop “worth” 60,000

◇ gold, bronze etc coins

fiat money
• worthless stuff guranteed by certain organisation to have some money
◇ it is currency not a note
◇ material has extrinsic value
◇ sodexo pass
◇ amazon gift voucher is valid at amazon not flipkart
◇ it is at the end of day worthless paper
◇ value can be revoked at any stage

transaction:
• product or service getting transferred from one entity to other entity
• transactions need not be always financial
• ie exchanging books, giving one pen away for free
• single ended transaction
◇ giving book to person
• double ended transaction
◇ selling book to person
◇ in return person gives money due to first transaction

problem with traditional transaction?
• for accountability and auditing and future settlement you need ledger system
• account keeping for traditional transaction is very difficult
◇ keeping unpaid money details in a physical ledger is difficult
■ one customer needs one page or pages(physical constraint)

    ■ needs to constantly update manually details could be mistakenly made or forged
  ◇ scalability is very difficult

transaction fee:

• utilizing a resource
  ◇ for transferring from account A to account B, if using google pay
    ■ google needs to ensure account A exists, has the money and account B exists
    ■ account details of A and B are in different banks, google needs to check with them individually before transaction is initiated
    ■ x amount of value from account A decreases and x amount of value from account B increases
    ■ this amounts to computational resource and storage resource being used which we pay with service fee or personal data being extorted etc :P
• profit
  ◇ google needs to make profit so something extra they will charge
  ◇ problem is unregulated banking services might charge really high profit charges
    ■ quarterly charge
    ■ maintenance charge

social justice

• present banking system does not guarantee financial inclusivity
• KYC norms becoming too unneeded
• 2014 -> Jan Dhan Yojana
  ◇ zero balance account can be started

◇ but requires aadhar etc which a large section of poor dont have

bitcoin blockchain

• removes banks as financial intermediaries
• charge less transaction fees
◇ records are distributed to all people to reduce the transaction fees
◇ decentralize the network
◇ apply public consensus
■ out of n nodes some nodes will perform the computation to see if transaction is valid
■ split the transaction among nodes validating transaction
• people have more financial freedom
◇ money is in control of the people not the government
• problem is if you are using real identity to participate you are not having privacy
◇ entire node is able to see transaction
◇ so annon the network

ram -> 50rs -> sam
sameul -> 100rs -> ram
jack -> 900 -> sam

this became

erer -> 50rs -> trtr
adad -> 100rs -> erer
sdsd -> 900rs -> trtr

◇ overtime if you use single anon identity patterns can

be established
    ■ so one anon user can have multiple accounts (as much as they want)
    ■ multiple accounts meaning multiple identities
    ■ bitcoin do not care about identity of user only validity of transaction

Value of bitcoin

• the more bitcoins you have in general the more expensive it is
  ◇ this is because it takes more time and effort to mine the same number of bitcoins today than 5 years ago
  ◇ bitcoins have a upper limit (21 million) this is will reach in 2140
    ■ but 18 million bitcoins are mined by 2021
      - ie process is more complex every year
      - ie finite resource to mine
  ◇ reward to create a new bitcoin block gets halved every four years
  ◇ more people will use the bitcoin blockchain but less bitcoins per person
  ◇ this added complexity and rarity makes bitcoin expensive

lecture 3

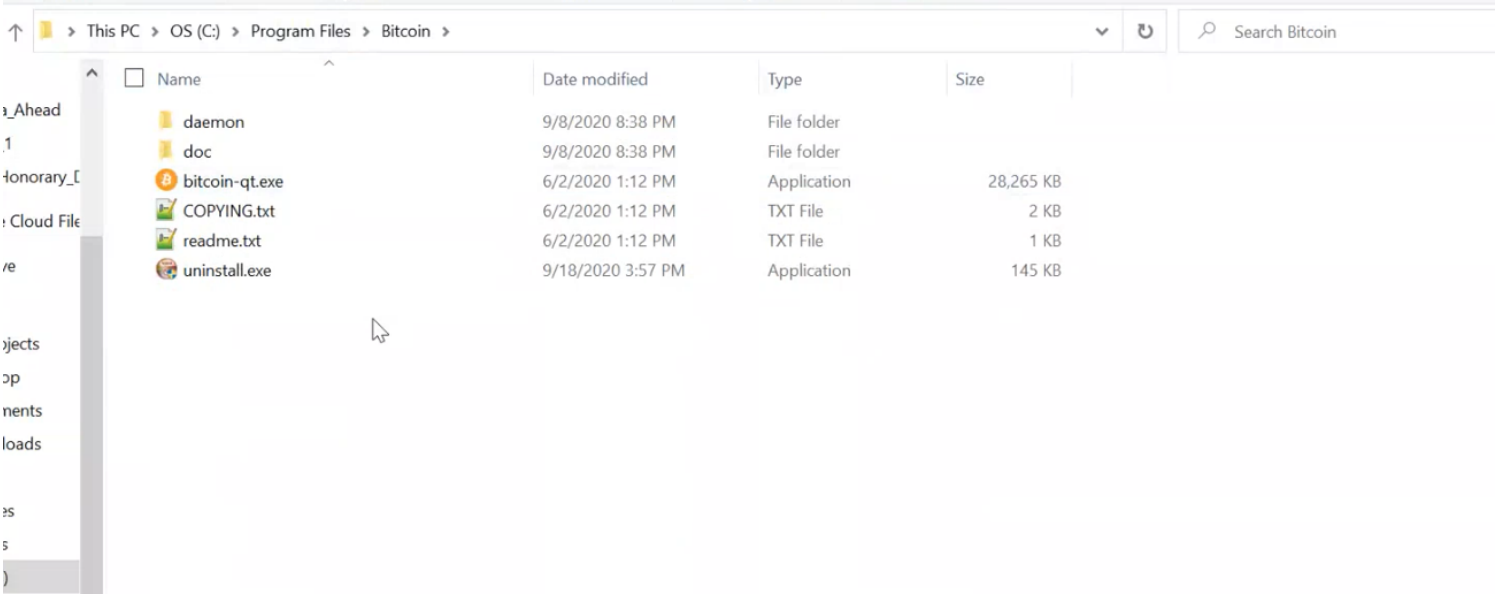
wallet

• container for money, documents etc
  ◇ physical wallet which can be carried with your self
  ◇ googlepay, paytm are digital wallet
    ■ whatever money u transfer from bank to digital wallet is there in ur digital wallet
    ■ if money is not there in wallet

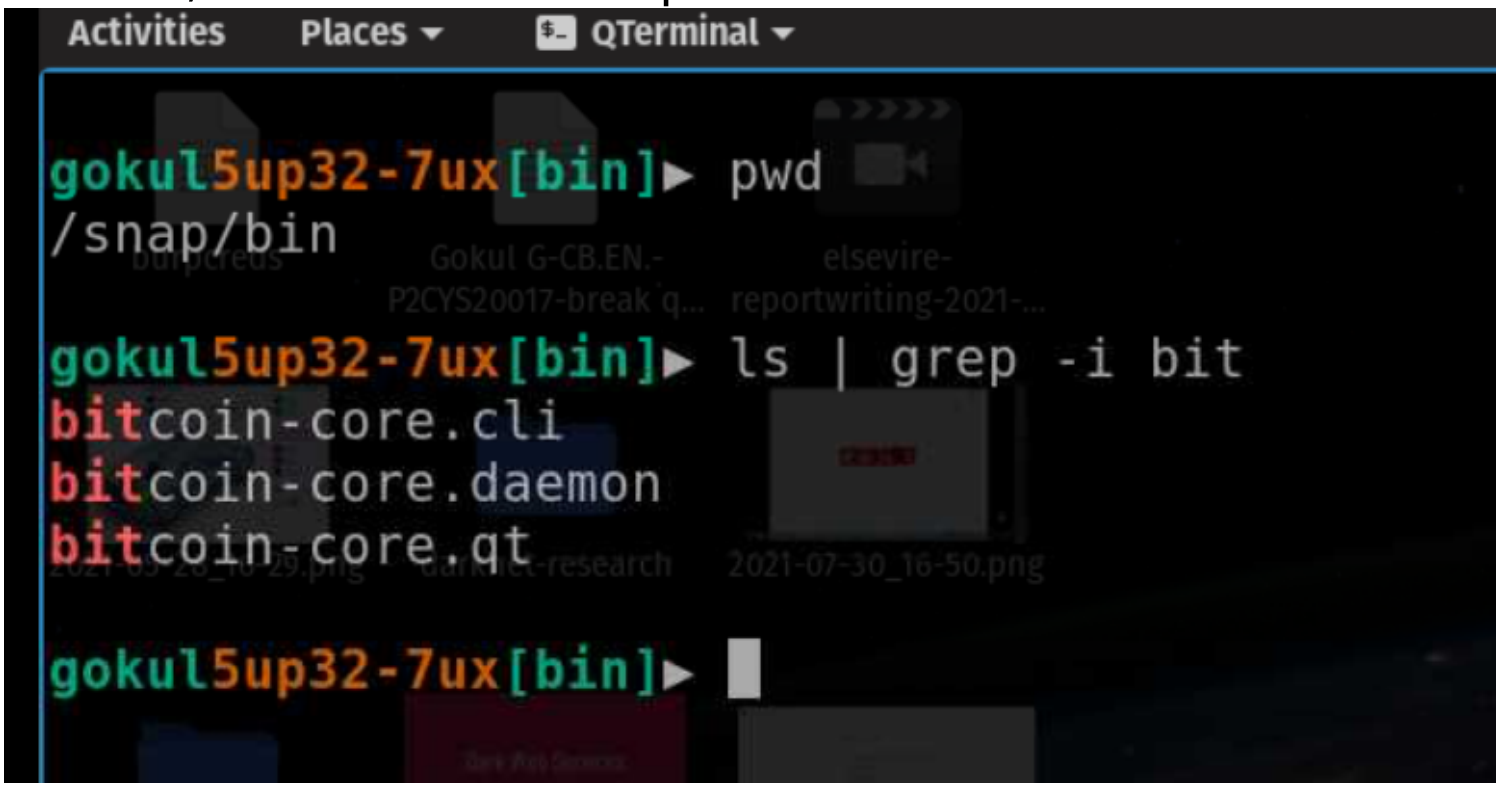
- then also we can do transactions
    - need not necessarily hold money
    - it is a place where we can have multiple accounts that we can use for digital transaction
    - it is a digital passbook, acting as payment gateway
    - mobile application or web mode
- crypto wallet
    - digital wallet for crypto assets
    - middle man which acts as passbook
    - multiple accounts possible for same or different blockchain
    - does not store the money as bitcoin etc are virtual
        - only have the private key and corresponding address used for signing the transaction
        - money model is UTXO → unspent transaction output (unspent bitcoin - value spend in tranasctions) and not a account balance model(data obtained from last transaction) like ethereum and this is a protocol token hence ethereum has crytpo called ether

bitcoin wallet

in windows:



in linux, installed with snap:



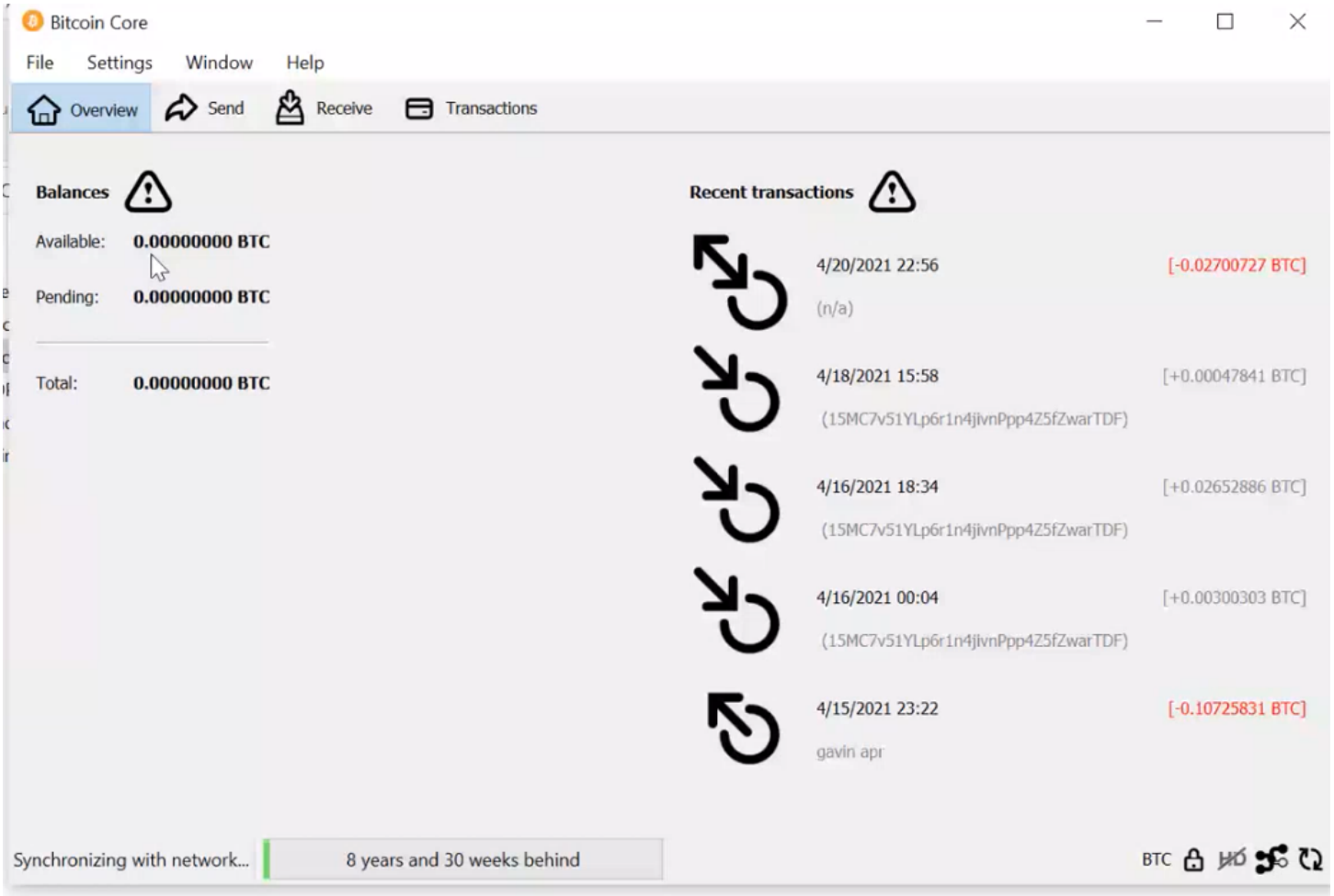
bitcoin-qt wallet is not only a wallet but also a bitcoin client

- meaning it is going to download the entire the bitcoin blockchain to the local node

◇ ie the ledger is going to be downloaded to your device

■ Client will download it because this is needed to participate in the blockchain

◇ if you run the bitcoin-qt, your computer is part of the blockchain ie a node



- we can have testing mode

◇ main net → everything is real, money and transaction is real

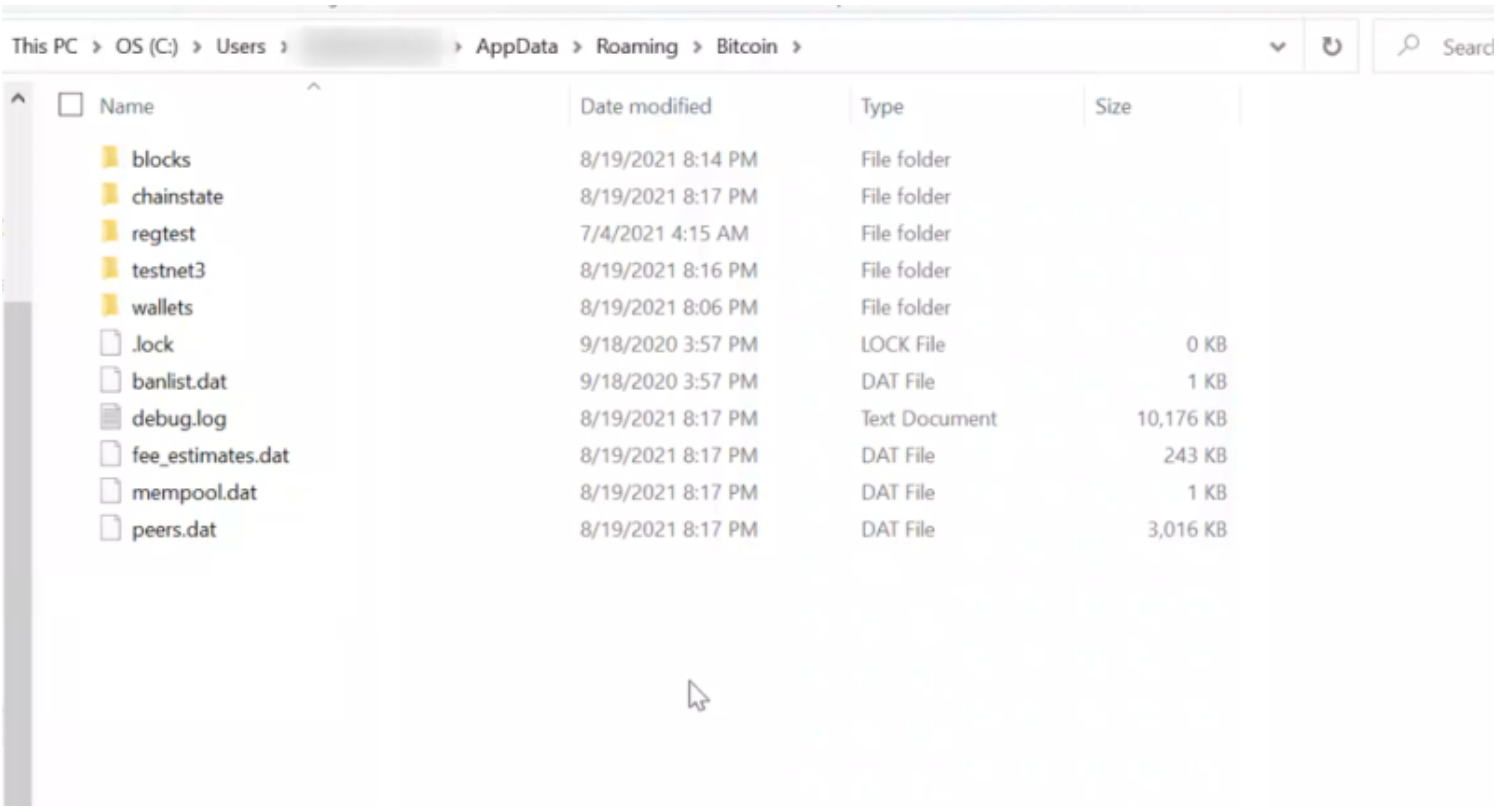
◇ test net → technology is real, money and transaction is dummy

■ we are gonna use only test network

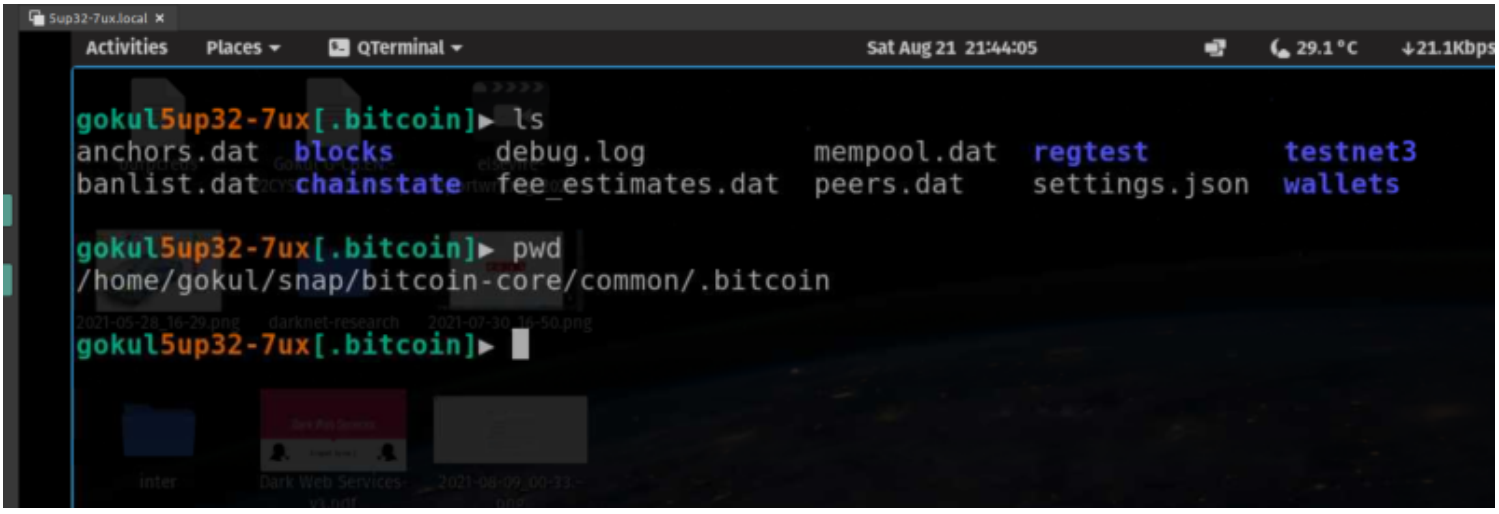
■ only less no of nodes when compared to main net

■ bitcoin.qt --testnet

in windows:



in linux:



blocks -> details about block stored as dat folder

- block details are stored in blkxxx.dat
  - ◇ one dat file has info about multiple blocks
  - ◇ max size is 128MB

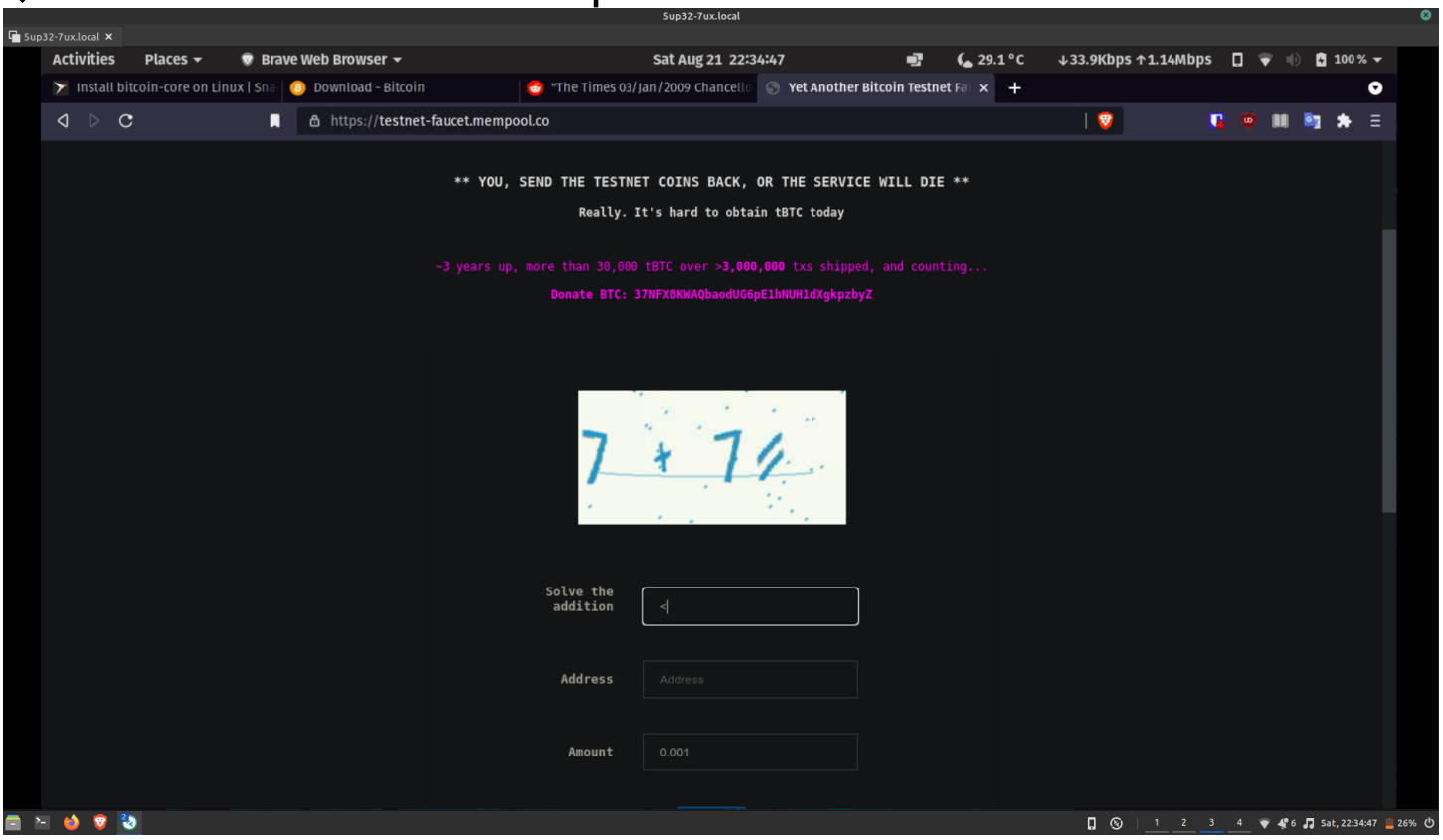
chainstate -> state of blockchain as database

wallets -> address and private key in a file called wallet.dat

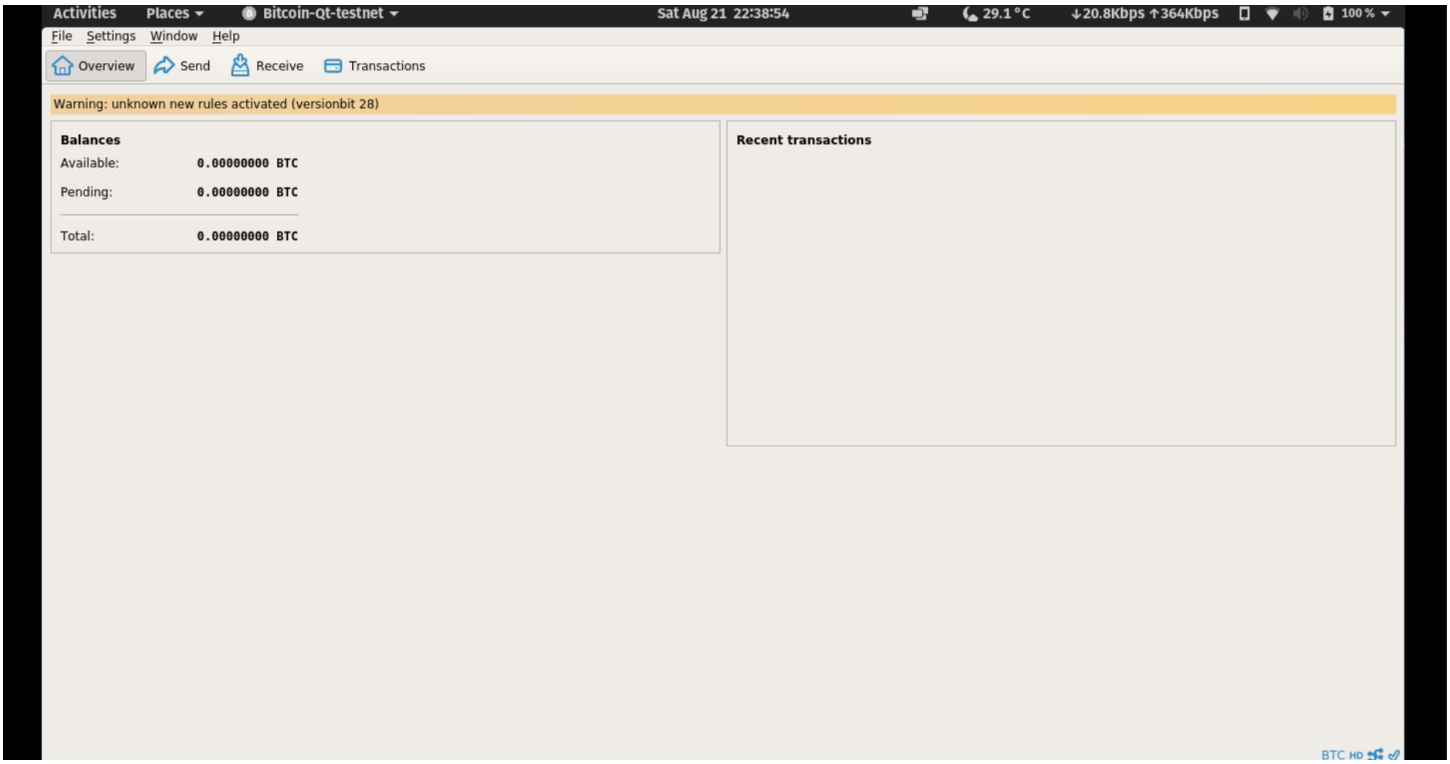
peers.dat -> connected peers
mempool.dat -> contain the transactions ur client has recieved to process
banlist -> block the contacts you dont want to communicate
feeestimate -> details regarding how transaction fees are kept
debug.log -> debugging info

getting test crypto to playwith

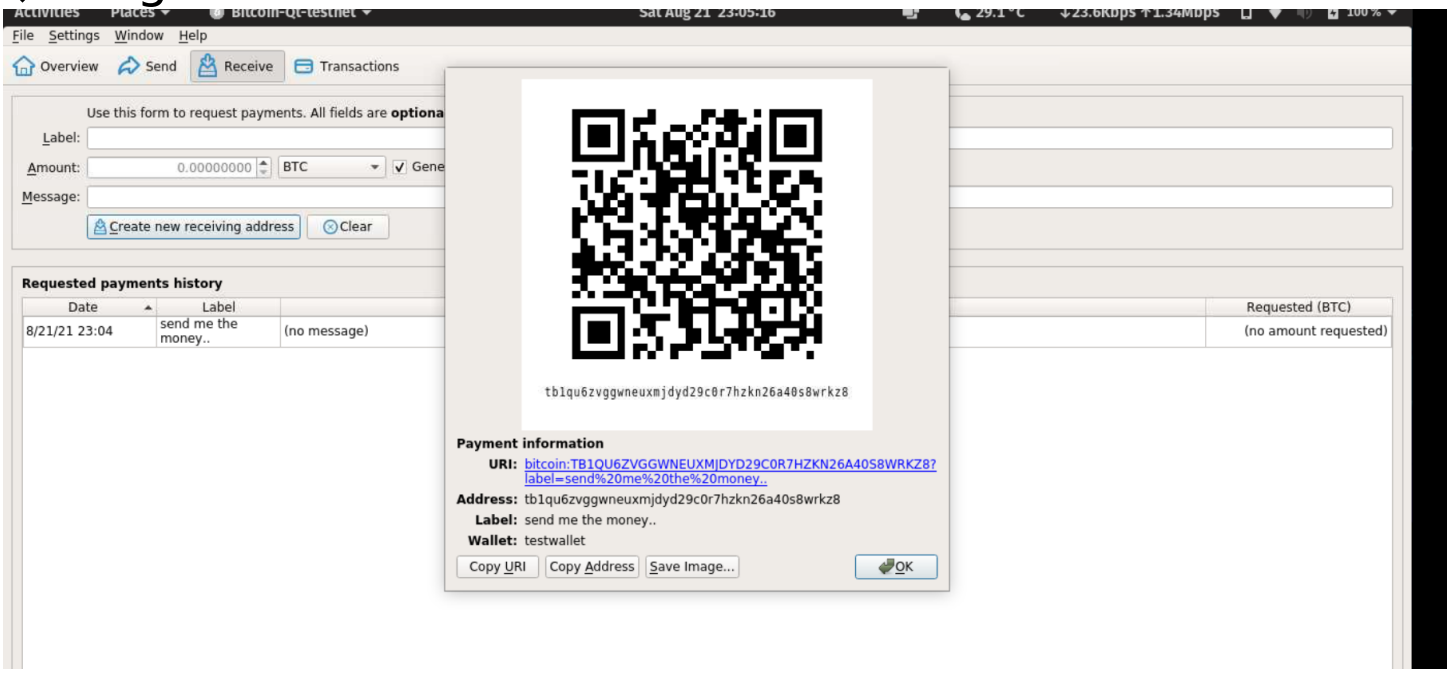
- use facuet
  - ◇ you can request test bitcoin to the site
  - ◇ this is not testnet but is an account with lot of “dummy coins”
  - ◇ only obligation is what ever you get you return it back
  - ◇ “testnet-faucet.mempool.co”



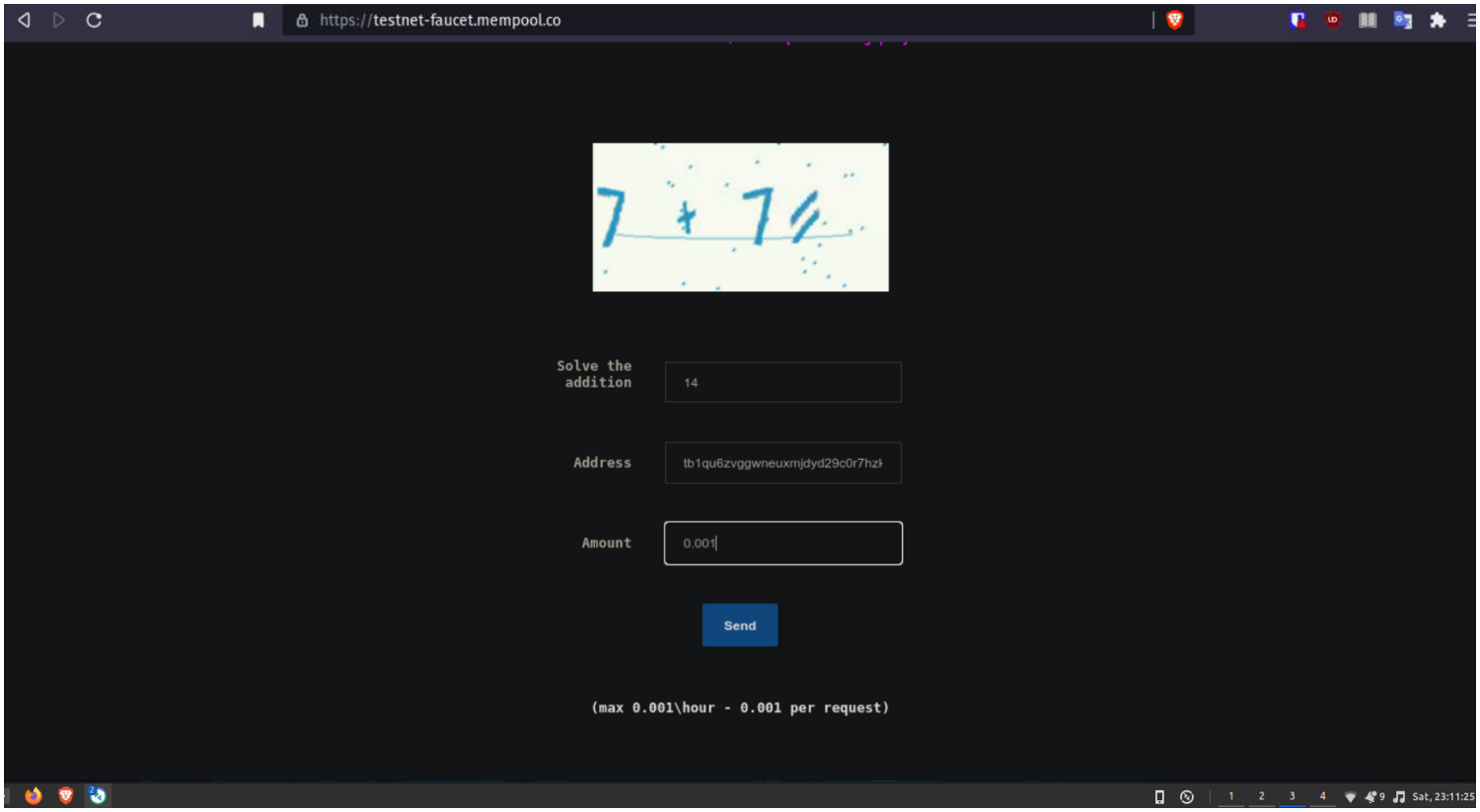
1) create a new wallet and address if you want

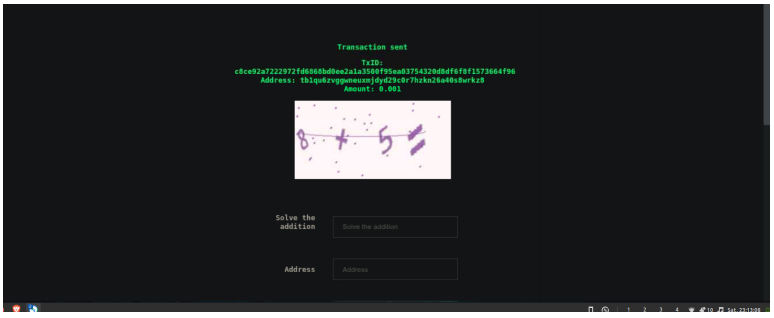


- now create a address
  ◇ length is 20-32 characters

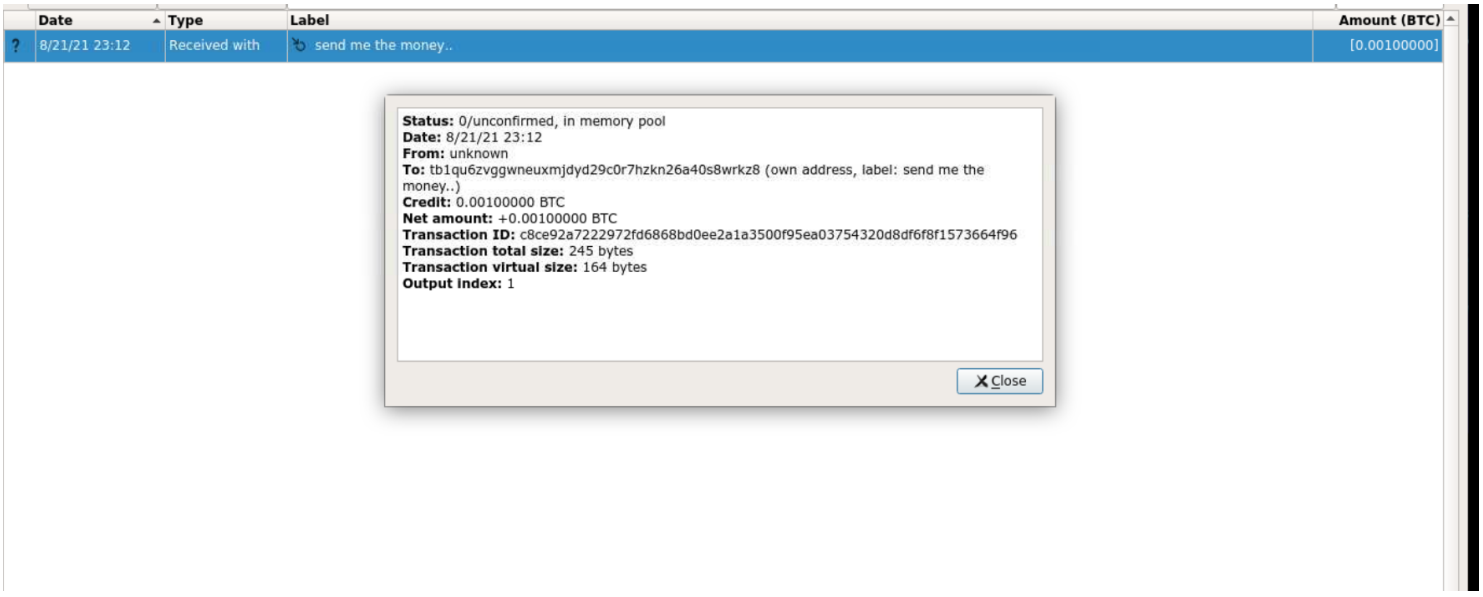


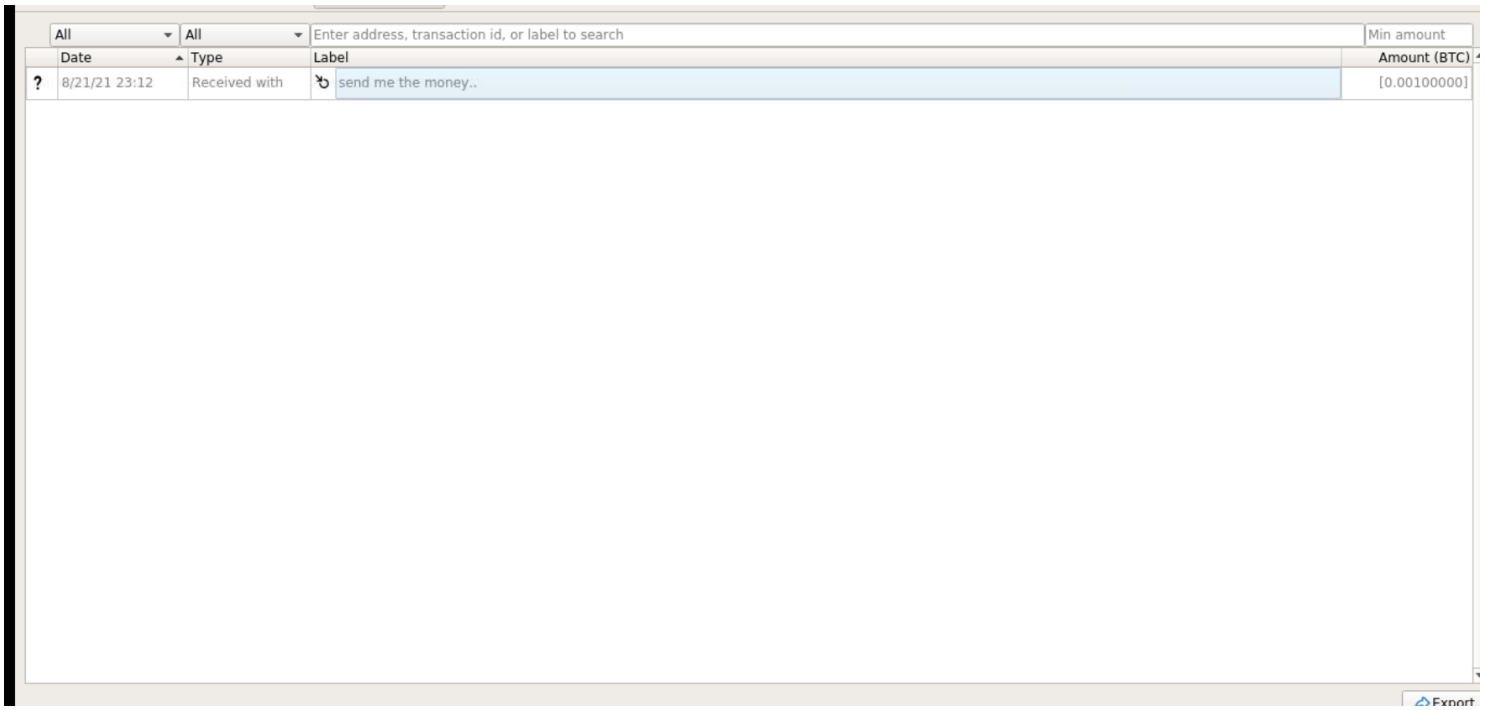
- copy the btc address and paste in faucet to get the money from it.



transaction sent to account 

status



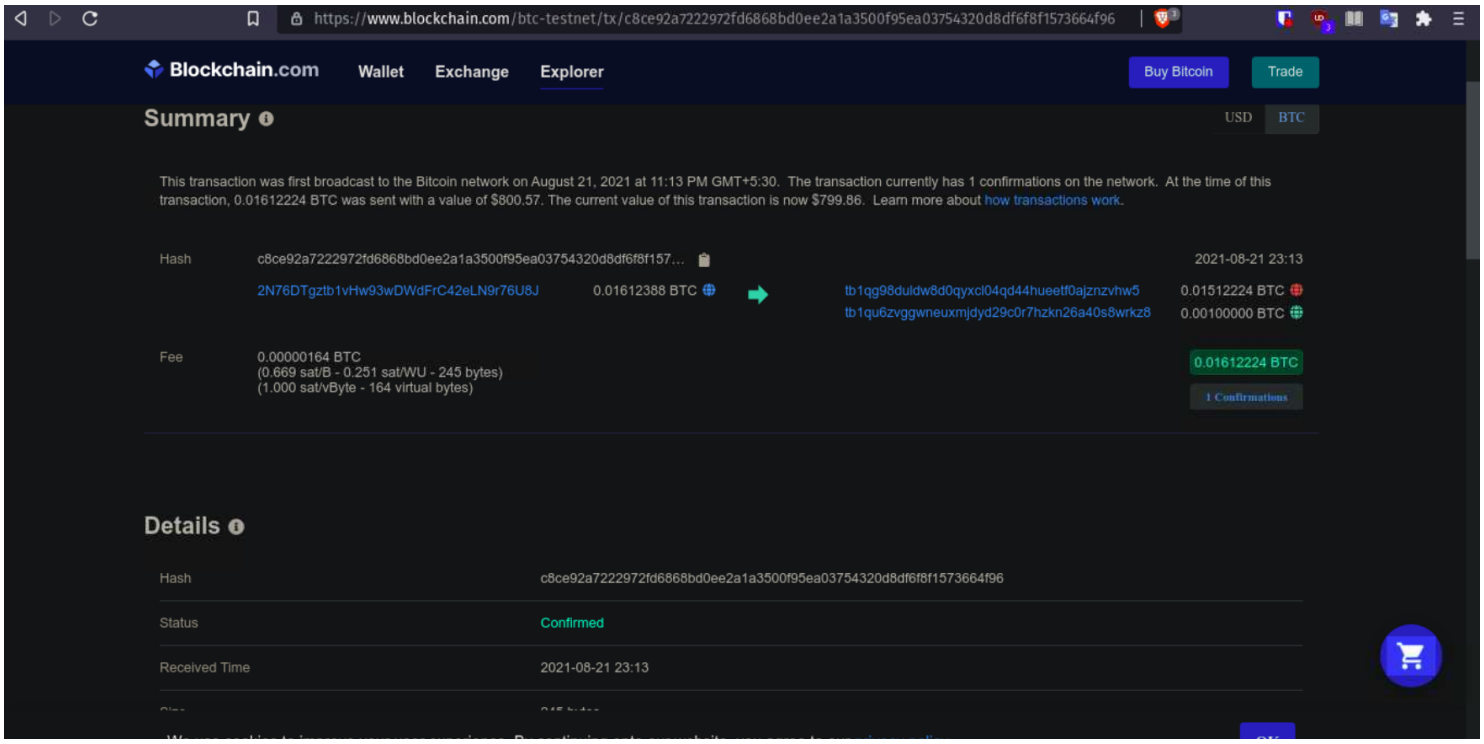


transaction is pending, because it is verified by the network ie miner

- recvier and sender account is valid?
- sender is having enough btc

but since it is peer to peer network, the moment the transaction is started, all clients will get the information about it loaded on its mempool.dat

- minimum 6 people need to verify that the transaction is valid(this is for mainnet)
- for testnet only one node needs to verify
  - ◇ process takes min 10 mins
- verified transactions gets written on the blockchain
- bitcoin is a digital and virtual currency ie it is just recorded in transaction and you cannot physically see it



lecture 04

- BTC, HD
  - ◇ HD -> hierachical determinsitc wallet
    - ■ hierarchical -> has well defined inheritance
    - ■ deterministic -> defined output - input pairs
    - ■ most wallets are HD wallet

how btc public-private key is generated:
when new wallet option is clicked:

private and public key points are generated on elliptic curve
pub key -> sdasda (in hex)
priv key -> dsadd (in hex)

operations performed on pub key to make it a bitcoin address:

1) pub key **-SHA256->** sha256_output1
2) sha256_output1 **-RIPEMD-160->** r160_output1

3) make a exact copy of r160_output1 ie r160_output1copy

4) r160_output1copy **-SHA256->** sha256_output2 **-SHA256->** sha256_output3

5) take first few bytes of sha256_output3 and append that to r160_output1 to get publicaddr_raw

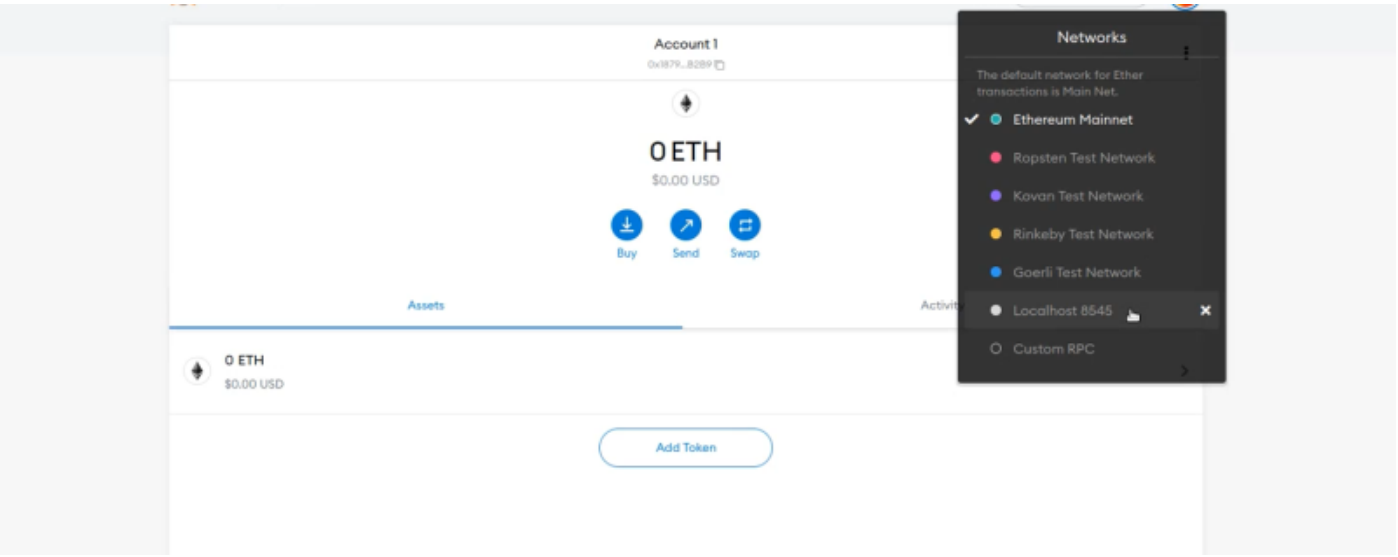
6) publicaddr_raw **-Base58CheckEncoding->** publicaddr_final (20-32 alphanumeric characters)
   1- before encoding, a hex value is prefixed which represents type of address (mainnet or testnet), compressed or uncompressed, private or public key etc

metamask - browser based extension simple node + mobile app

• we can use this to connect to ethereum blockchain and any decentralized web as well
   ◇ ex: binance smart chain (BSC mainet)
   ◇ mumbai test net etc
• we can either import or set up a new wallet
   ◇ if set up a new wallet create password and a secret code (for account recovery they will give 12 random english words)

    - aka secret phrase
    - aka mnemonic
    - this is for wallet password not blockchain
    - blockchain has no password only private key

• HD wallet
  ◇ for pseudo annoymity each user can have multiple addresses
    - for different types of ethererum nets in case of metamask
      - like ropsten
      - koven
      - rinkeby
    - we also local client or remote client etc
    - mainnet is there as well but easier to switch between nets



  ◇ each address is a private key
  ◇ when ever you want to regenerate addresses
    - addresses are stored in a hiearchical way
    - if you want to regenerate, plug in the secret code
    - all addresses are obtained in a deterministic way
      - “for these 12 english words for a particular currency” -> will always generate the hierarchical addresses

    ■ this is called BIP32 standard

• ether -> ETH
• the address you have in mainnet will be there in all the testnets
  ◇ but purpose is different
  ◇ same account number is there in multiple nets but different balances are there in diff nets based on transactions
    ■ no encoding is there in eth for addresses directly hexa they will use
    ■ in bitcoin we have for each net we have different account number
      - for purpose of distinction
depends on gas fees
      transaction fees is calculated
      more gas fees less time to complete

• for small transactions you need to pay higher transaction fees
  ◇ so for everyday paytm like usage ethereum is not good
  ◇ for size of data stored ie for smart contracts, NFT etc then alos storage fees is very high

faucet for metamask -> facuet.metamask.io
crypto wallet can also store crypto assets.

lecture 05

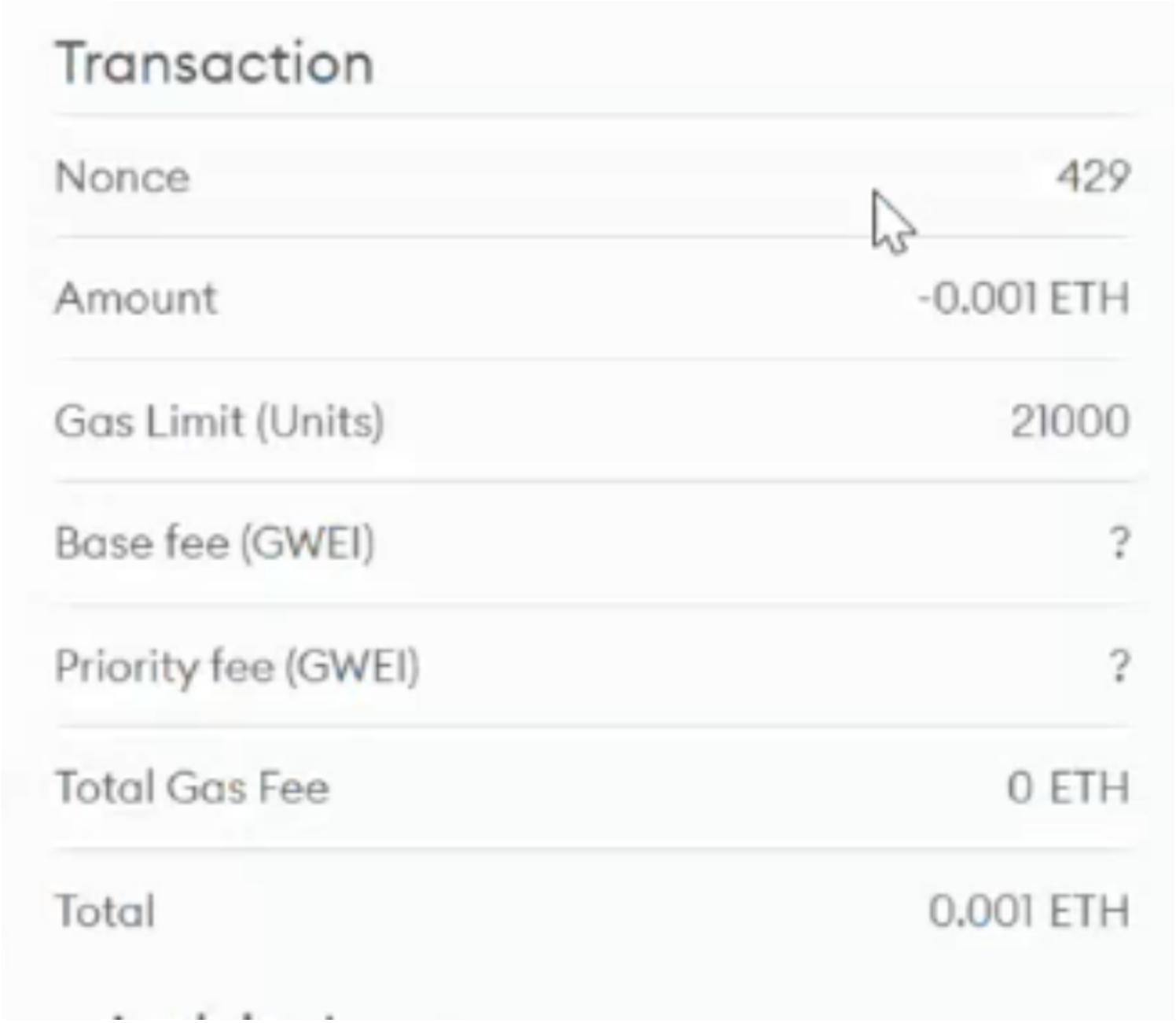
• prerequisite
  ◇ cryptography

    - ◇ mathematics
    - ◇ databases
    - ◇ how internet and distributed systems works
    - ◇ networking basics
    - ◇ programming
- public key
    - ◇ address you want to transact with
- private key
    - ◇ hash of the message your sigining with private key
        - ■ aka this is the signature
- cryptographic hash functions are one way mathematical functions used to provide immutablity to data
- properties of hash
    - ◇ preimage resistance
        - ■ if i have input very easy to find output
        - ■ if i have output practically difficult to find input
    - ◇ second preimage resistance
        - ■ for m1 input we have output H
        - ■ for m2 input we should not have same output H
    - ◇ collision resistance
        - ■ it is not possible to find any two different input which gives same output.
    - ◇ avalanche effect
        - ■ small change in input gives huge change in output
        - ■ ie deterministic attacks fails
        - ■ puzzle friendliness
    - ◇ MD (message digest)
        - ■ MD2(dead)
        - ■ MD3(dead)
        - ■ MD4(dead)
        - ■ MD5(dead)
    - ◇ SHA (secure hash algorithm)

- SHA1 160bytes
- SHA2 224,256,384,512bytes
  - blockchain uses SHA2-256
- SHA3 224,256,384,512 bytes
  - ethereum using keccack-256

◇ RIPEMD 160

• transaction ID → hash of transaction details



all the details are hashed and called as transaction ID

• computed by client and attach it to the transaction
• this details is send across all nodes
  ◇ transaction details are verified in public blockchain
  ◇ integrity of the detail is verified by digital signature
• as we have multiple transactions at the same time or similar time
  ◇ transactions in mempool are verified and are put in block
    ■ block is broadcasted to other nodes to be verified by consensus algo

minning:
• node will group all the transactions together add some data to it to form a block(including hash of previous block)
• block is verified and crypto puzzle is solved to add proof that he did the work correctly
  ◇ correct block means, the node gets the coins coming from new block as well as transaction fees
  ◇ transaction is verified and then money from ur account gets reduced only if transaction ends up on block ie transaction is complete
    ■ this is called finality of transaction
    ■ a block to be created requires 10 mins on avg in bitcoin
      - this does not mean ur transaction gets confirmed in 10 mins
      - depends on whether ur transaction is in the next block to be created or not
      - this consensus is bitcoin is called Proof of Work
        → to reduce congestion
        → to verify integrity of transaction
        → to scale difficulty based on no of people in blockchain and tech used

→ PoW is cryptohash
⇒ solution is complex
⇒ verification of solution is easy

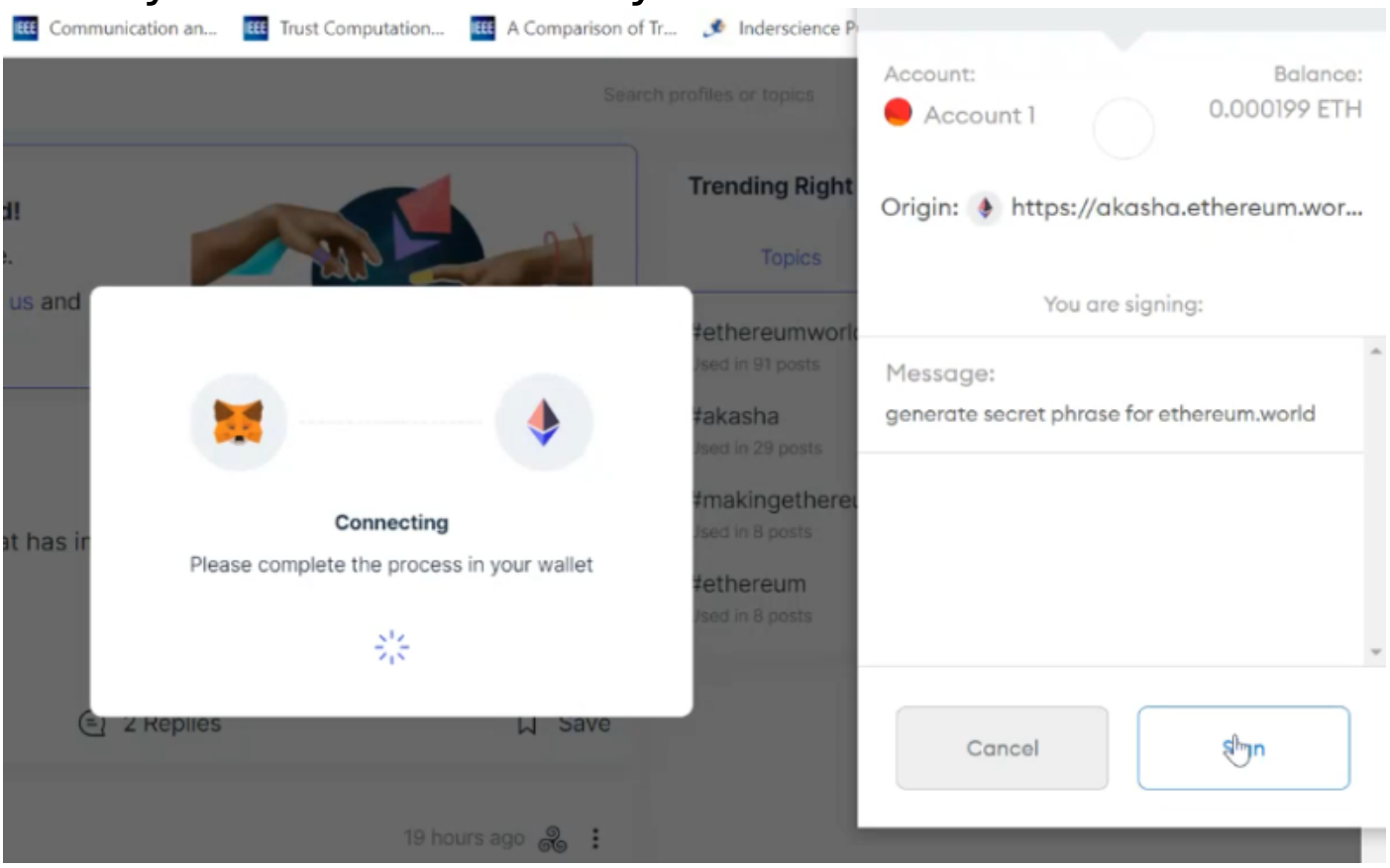
• difficulty parameter → measure of how difficult to mine a block
◇ reverse computing of nonce from hash value of block is difficult
◇ will say the first x characters of hash value is zero
◇ you have to bruteforce values starting with a nonce to figure out what is the nonce that causes this hashvalue
◇ the hashvalue, the number of x characters and difficulty parmater are all fixed by blockchain algorithm
◇ difficulty changes in bitcoin blockchain every 2016 blocks

• this nonce and the rest all are included in proposed block and send to others
• others will calculate the hash of the block and verify if the hash obtained is matching the difficulty to verify and confirm the block
• immutable in the sense all the blocks are connected via previous block hash

Lecture 06

• cryptokitty NFT
■ indivisble
- we can use “ethitem” to “fractionalize” NFT
→ wrap erc20, erc721 etc to mask NFT behavior and then fractionalize it
→ the main NFT in blockchain is burnt ie no one

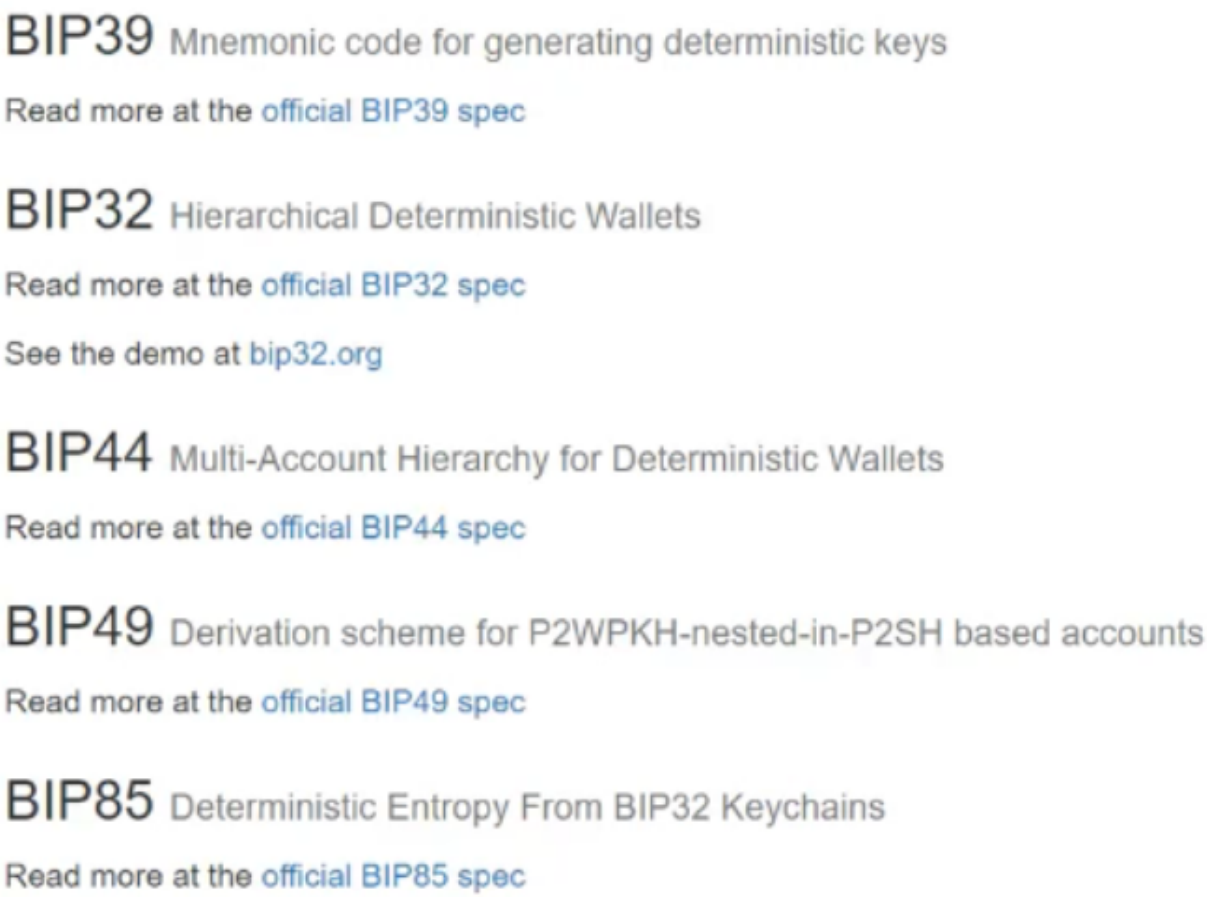
can spent or own the NFT
    - cannot be exchanged with other NFT of same chain
    - globally unique
  - ◇ first token based on ERC721 standard
  - ◇ value based on uniqueness and rarity
  - ◇ tokenization
    - digital conversion of asset and put that in blockchain
    - diginoor → fan movie clipping is billed as NFT
    - cryptopunks
    - mooncat rescue
    - pixelchain art
    - marble card → polygon matic
    - unlandme → virtual realestate
- decentralized social media applications
  - ◇ akasha.ethereum.world
    - preview ie alpha phase in rinkeyby test net
      - signin process
        ⇒ generate digital signature using user privatekey to validate identity



• Ad rewarding the user to click via the brave basic attention token issued on top of ethereum
  ◇ all the ppl who are using brave browser will get the ad
    ■ will reward the person who saw the ad with BAT
    ■ can also directly support the person who created the ad
  ◇ anyone can post ad
    ■ will get higher value to the one who posted the ad
      - as no middle men like fb ads or google ads or apple ads
    ■ so it is anti monopoly

Lecture 07

• HD wallet is proposed by bitcoin, this is BIP32,BIP44,BIP39 etc



◇ metamask using BIP44(EIP44) for multi account HD wallet
◇ BIP32 is the normal single account HD wallet
◇ secret phrase can be of any size(default 12) in any language
◇ so it is part of BIP (bitcoin improvement proposal)
◇ ie subsequent public private key pairs are dervied from single public-private key pair

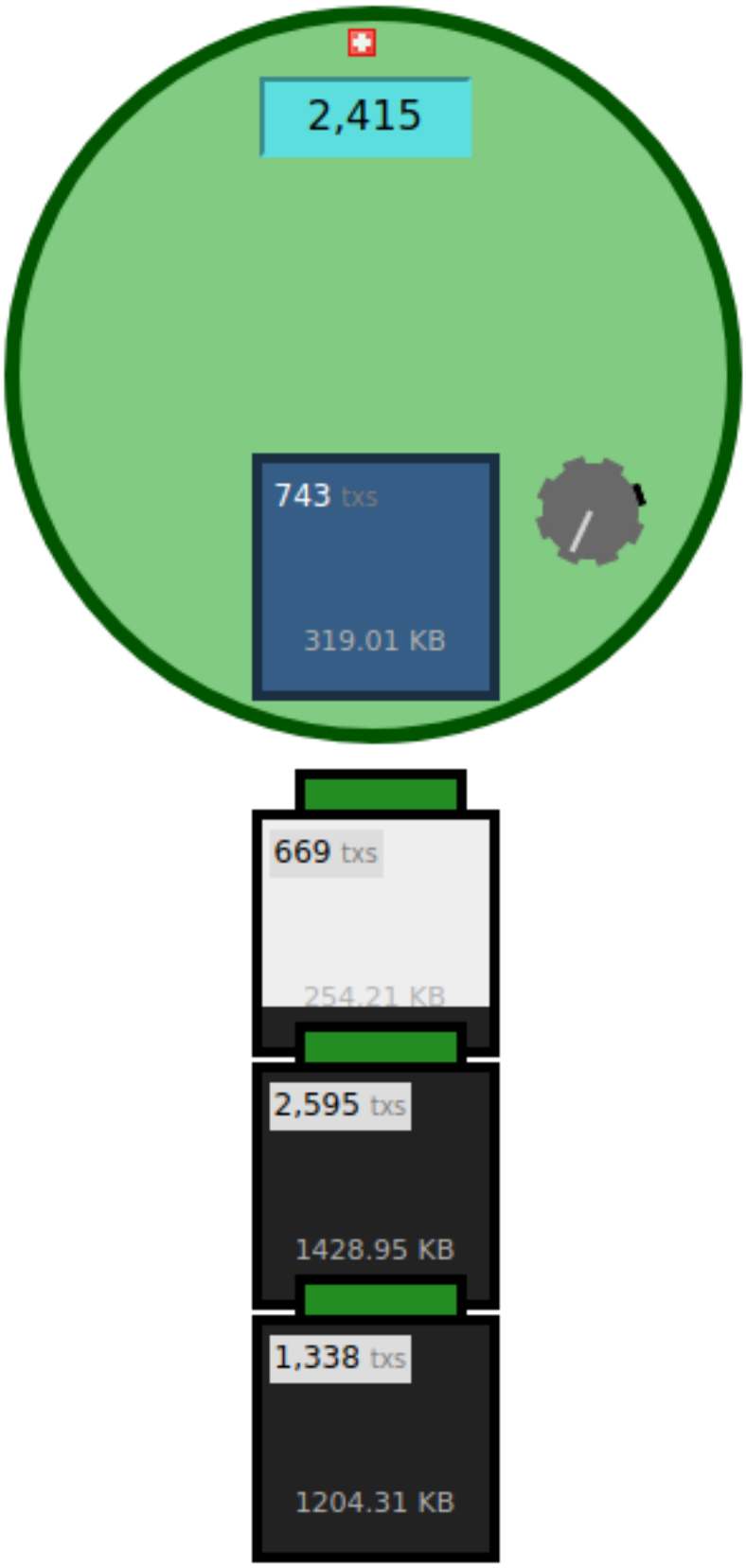
• fork
◇ hardfork ie create a new chain
■ from Bitcoin (BTC)
- Bitcoin cash (BCH)
- Bitcoin Cash SV (BCHSV)
- Bitcoin Gold (BTG)
■ from ethereum classic(ETC) (old version)
→ Ether (ETH) (new version of ethereum)
• mutable “blockchains”
◇ ie redactable blockchains where we can modify/-delete past transactions

Lecture 08

NODE

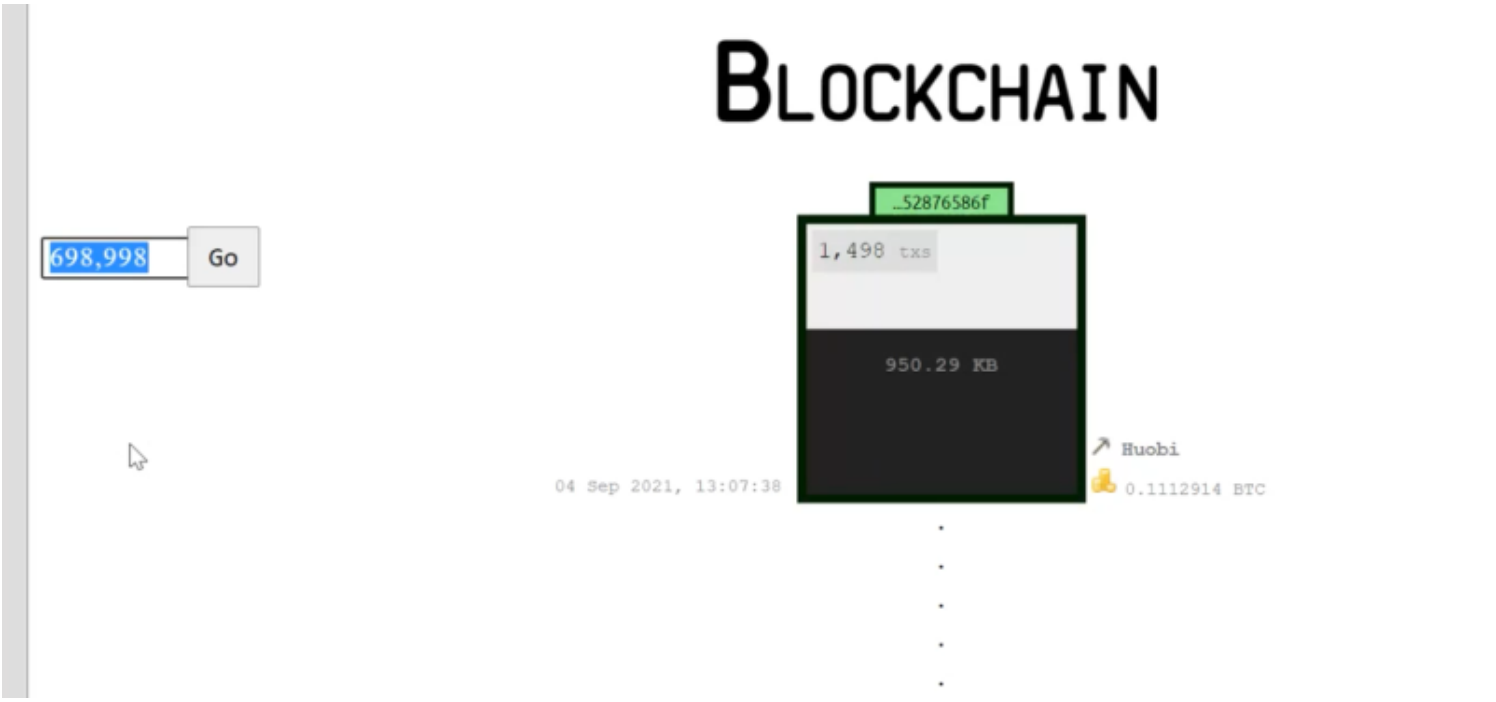
learn me

(show)

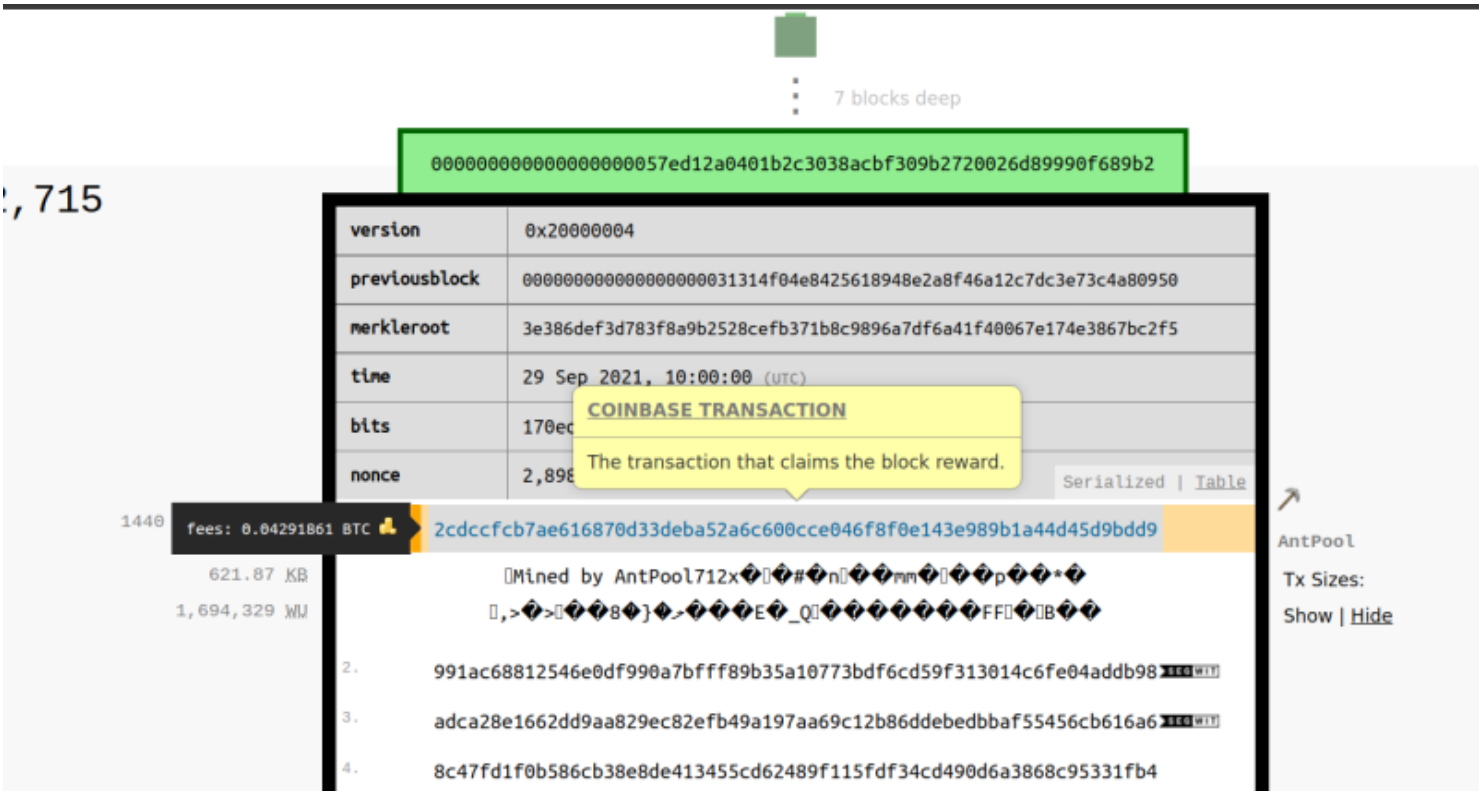


• 2415 confirmed transactions in mempool, with 743 in new proposed block by node, with gear reprsenting

difficulty
• new proposed block is the candidate block as other miners also will compete for bitcoins
  ◇ validity vs completeness or finality
• all confirmed blocks are having different number of transactions ie max 1 to 4MB like that
• block number is same as block height
  ◇ 698998 is blockheight
• hash of the block is not stored within the block as indicated by green color box and is only part of the next block
• the current block does not know its hash



block height 7027115 mined at 10 am, 29 sep 2021 by antpool got transaction fess of 1440 transactions plus of 04291861 BTC



the first transction is block is the coinbase transaction which is automatically generated and is the transaction reward of the miner.

the block hash is basically the hash of the block header.

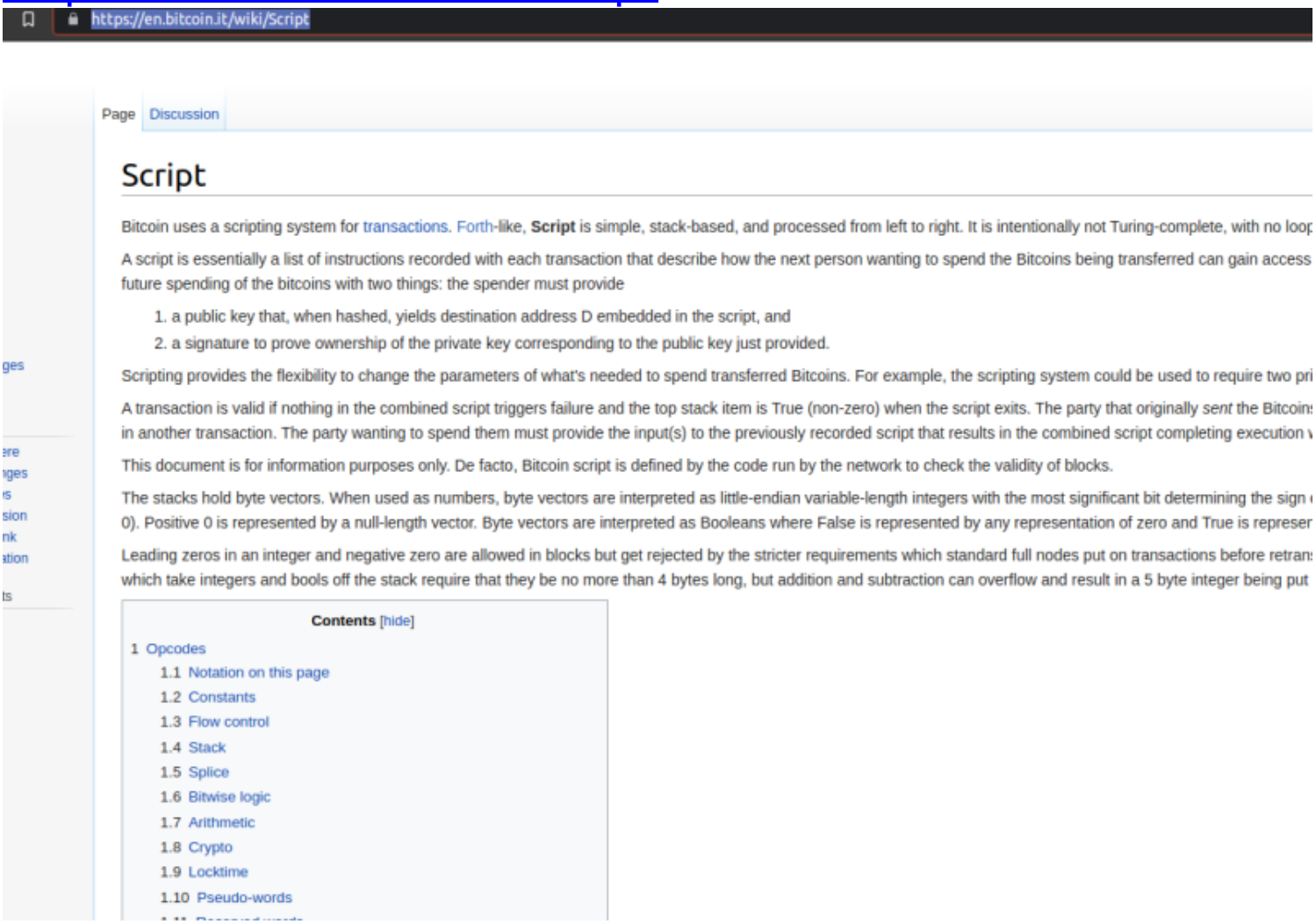
• transactions are secured via merkle tree
  ◇ we can which transaction got tampered by the merkle root hash
  ◇ the tampered transaction will affect all the parent hashes above it till the merkle root of the tree
  ◇ we can reconstruct the entire tree from non tampered hashes of other nodes to tell ok this transaction is wrong
  ◇ ethereum uses Patricia merkle tree which is an efficient form of merkle tree

public --> no sensitive data stored
private → sensitive data stored
permissoned → regardless of data stored you need permission to access

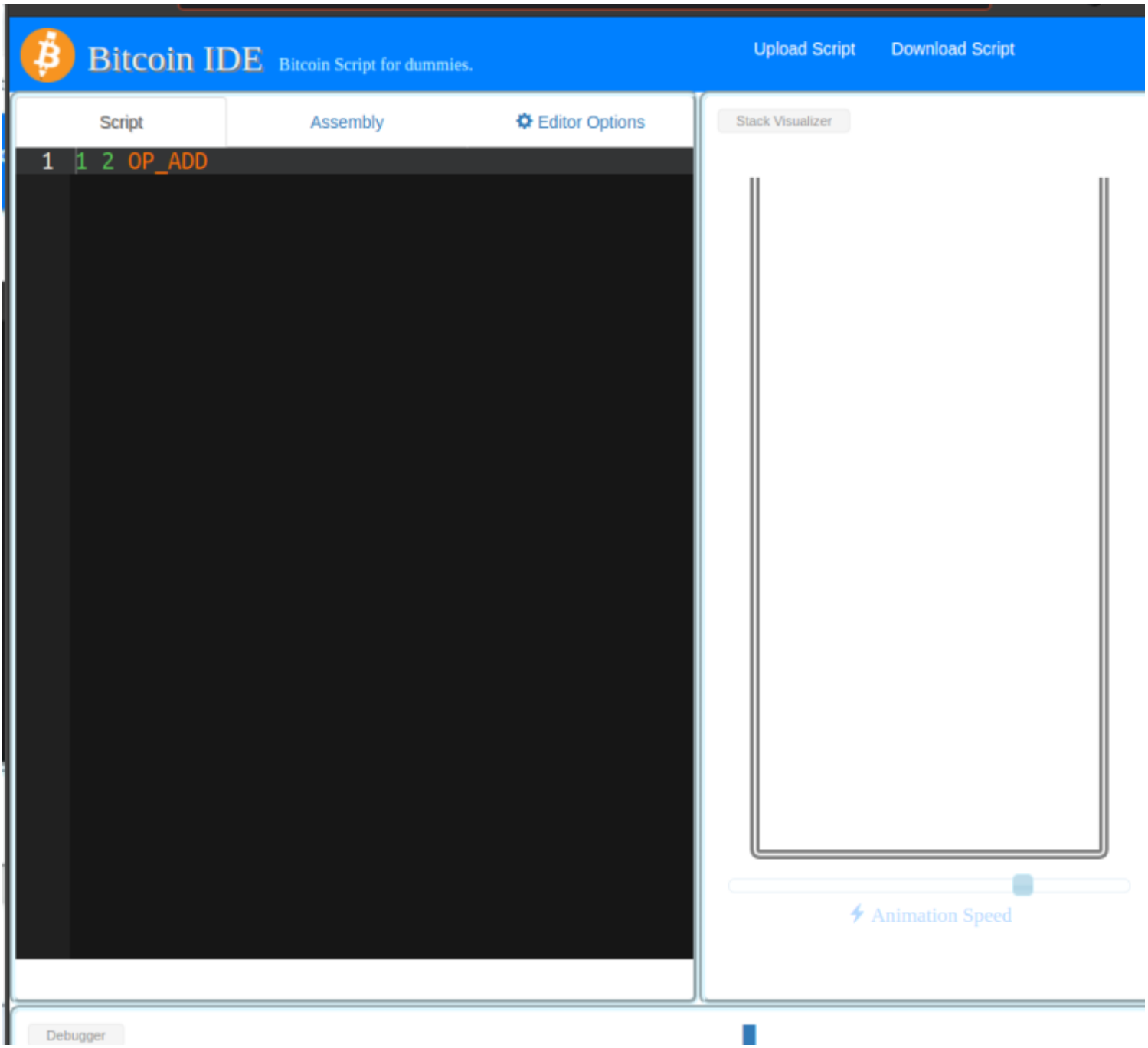
lecture 9

• in bitcoin, each address is like one bank account
• so we can have as many bank accounts as we want for all unique transactions, this aids in psuedo privacy
  ◇ this is part of UTXO
  ◇ we can have multiple accounts sending to single/-mutliple accounts or viceversa
• in ethereum, we are sending all stuff to single wallet address
  ◇ we can have as many wallets as we want
  ◇ this is acccount based model

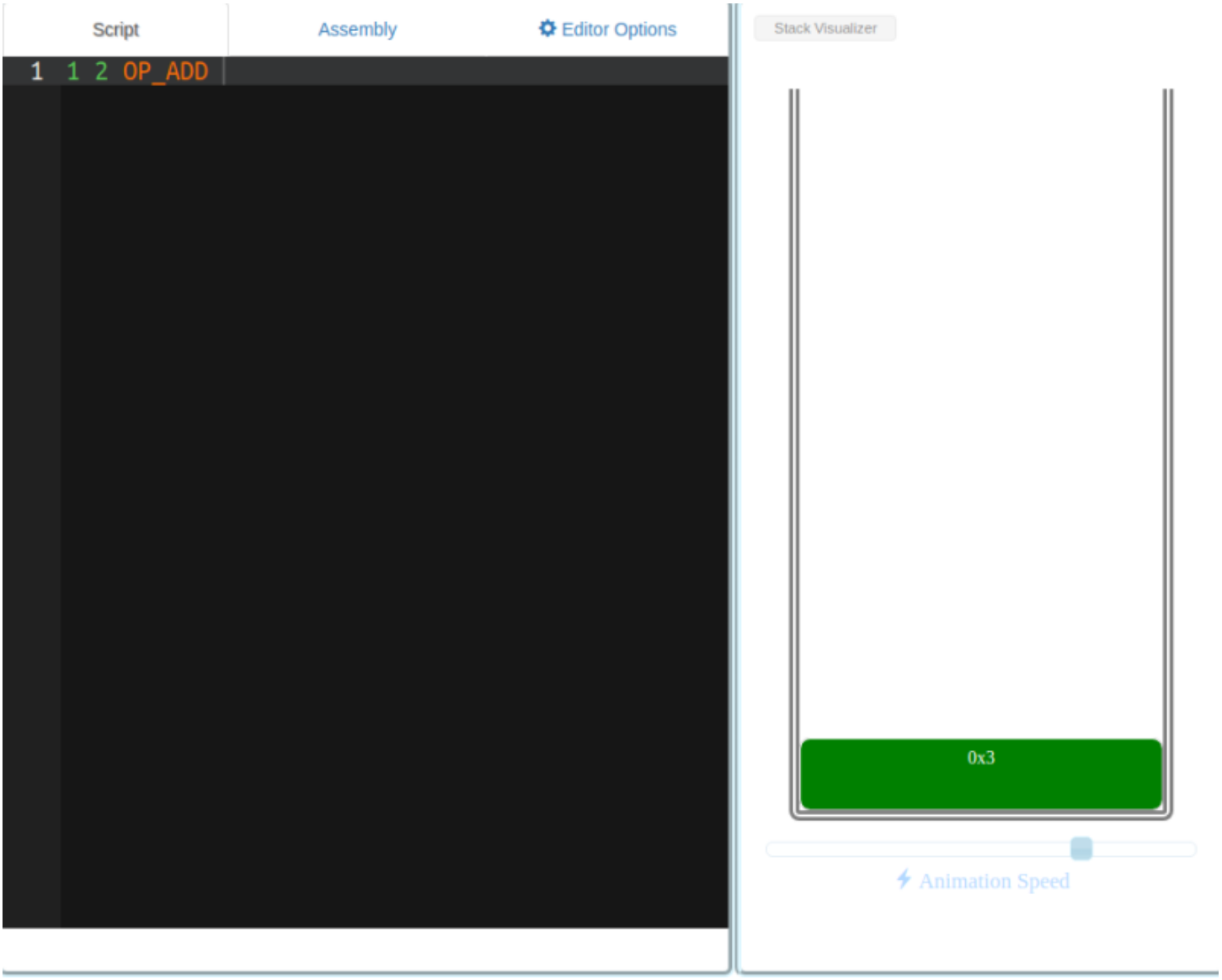
bitcoin script:
https://en.bitcoin.it/wiki/Script



https://siminchen.github.io/bitcoinIDE/build/editor.html



- similiar to assembly language code
  - ◇ stack based execution
    - ■ LIFO
    - ■ pop and push based operation
- pay this tx to this person after some time like this we can “code”
  - ◇ this is not as easy or feasible as smart contracts
  - ◇ this is one of the reasons for blockchain 2.0 and application based blockchain



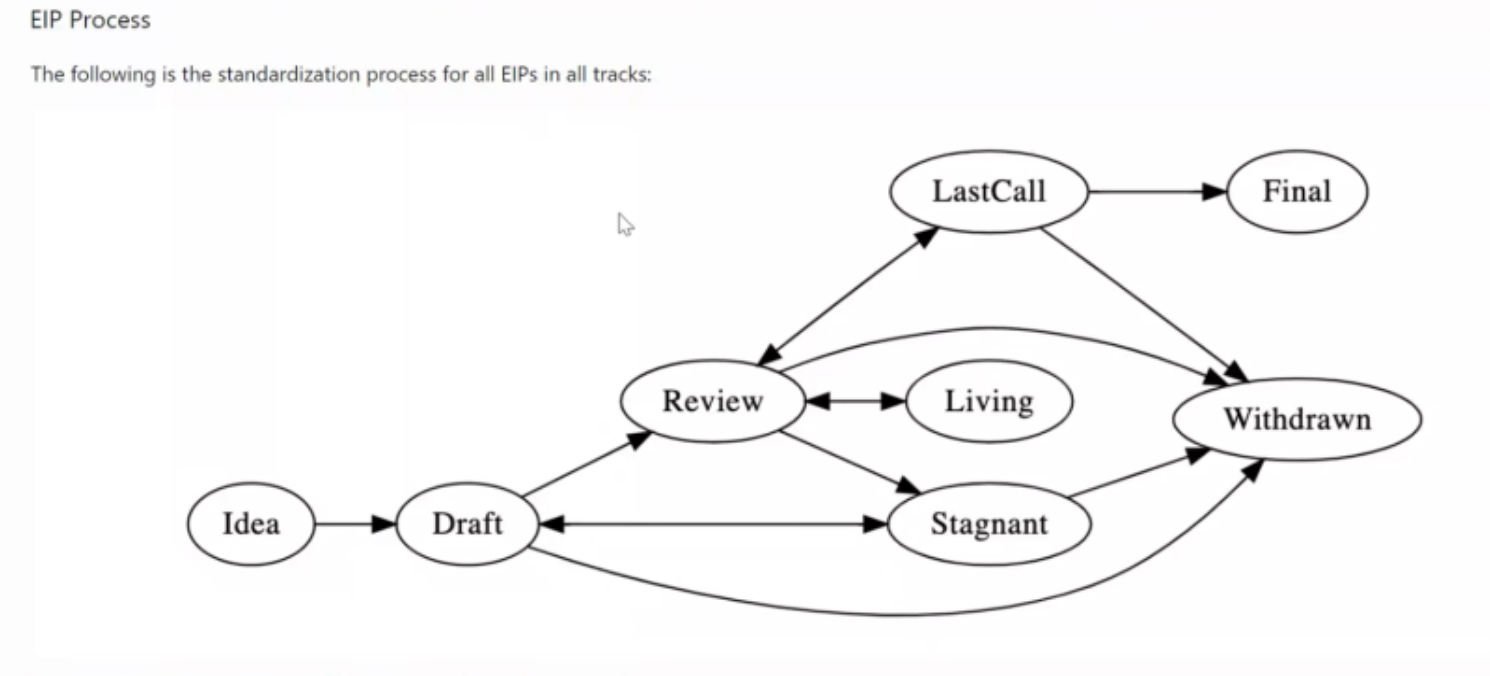
for : 1+2 = 3

	Script
	<sig> <pubKey> OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
	OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG

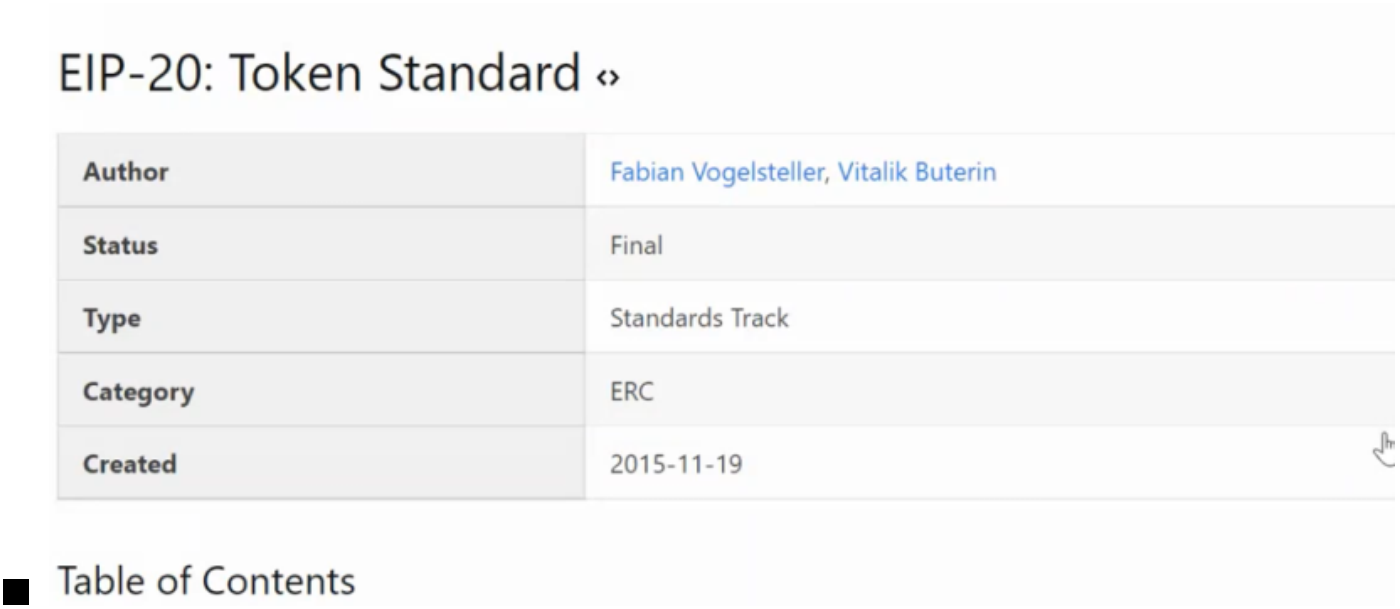
script to check if authorized user is only sending

lecture 10

- BIP
  - ◇ bitcoin improvement proposal
  - ◇ if it is implemented status is final
  - ◇ if it is replaced something, that something status is replace
  - ◇ even if it is implemented or dead, bip is always named as bip
  - ◇ bip-1 → bitcoin introduction and scope work etc
- EIP
  - ◇ EIP → ethereum improvement proposal



  - ◇ ie centralized decision are made as to how this is implemented
  - ◇ a idea which is finalized is called ERC
    - ■ ethereum request for comments
    - ■ EIP20 became ERC20 (for fungible tokens)
    - ■



block 478558 is the hard fork point from which BCH came from BTC
- uptill this point BTC and BCH blockchain are identical
- from BCH, we got BSV
- from BTC we got bitcoin Gold

lecture 11,12

why ethereum

- what i need is not provided by bitcoin
  - ◇ layer to write smart contract, dapps
  - ◇ easy scripting language integration
- instant payment not possible because of time and transaction payment issues
- what is in bitcoin I dont need
  - ◇ only fungible token support
    - ■ I need non fungible tokens
  - ◇ regardless of transaction everything is average of 10 mins for confirmation
    - ■ consensus for some time critical applications is not covered

◇ PoW is good for ensuring no one can manipulate
■ but is slowly computionally infeasible to invest in a bitcoin operation nowadays

smart contract
• A consensus protocol for correct execution of a publicly specified computer program.
◇ executes terms based on some time/event based conditions
• the code is avaliable in all nodes of network via blockchain transaction
◇ when a smart contract is triggered, it gets triggered in all nodes of the blockchain
◇ once result is generated, it is publicly verified and agreed upon
■ the result is stored as a transaction

distributed applications
• set of smart contracts running on every node
• smart contract is not legally valid outside of ethereum
• ricardian contract
◇ form which has printable in a human readable format
■ for court solving , we can use this as this is english format
◇ using this as input a template code uses this and deploys as a smart contract
• ETH is 10^-18 divisible max
◇ more transactions are there
■ transactions in applications
■ applications
■ crytpo etc
◇ so we need to pump more money
◇ ethereum has no defined money upper limit

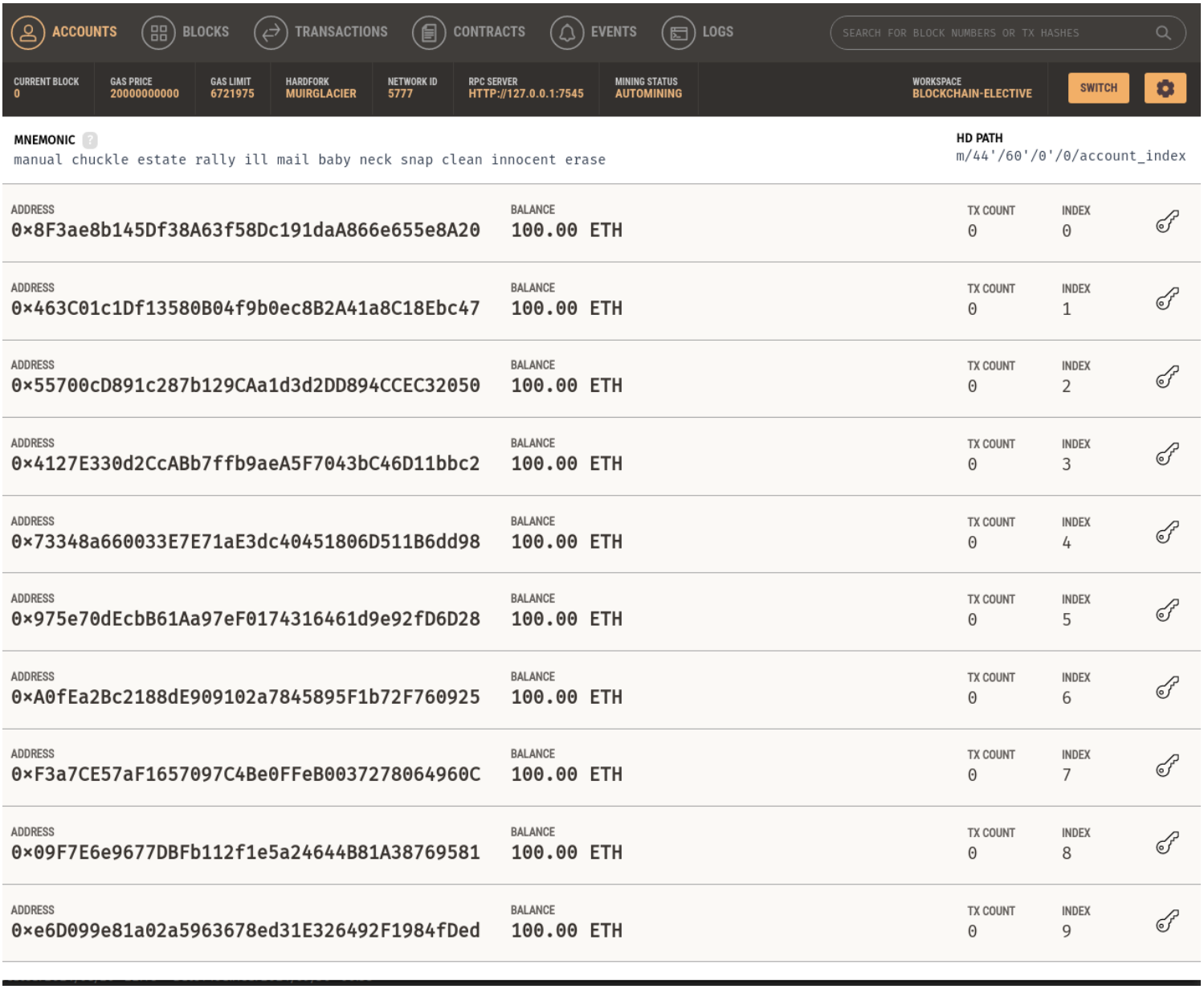
Unit	Wei Value	Wei
wei	1 wei	1
Kwei (babbage)	1e3 wei	1,000
Mwei (lovelace)	1e6 wei	1,000,000
Gwei (shannon)	1e9 wei	1,000,000,000
microether (szabo)	1e12 wei	1,000,000,000,000
milliether (finney)	1e15 wei	1,000,000,000,000,000
ether	1e18 wei	1,000,000,000,000,000,000

• we need some dev tools for writing and deploying smart contracts on blockchain
  ◇ IDE
    ■ compiler
    ■ deployer
    ■ helpful for debugging,performance etc for debugging
  ◇ local support for development
    ■ inbuilt wallet
    ■ inbuilt blockchain explorer
    ■ local blockchain
    ■ default is preloaded with currency
    ■ private blochain for development and testing
      - truffle framework(for smart contract developement) + ganache(private self contained blockchain)

→ ganaches use port number 7545,

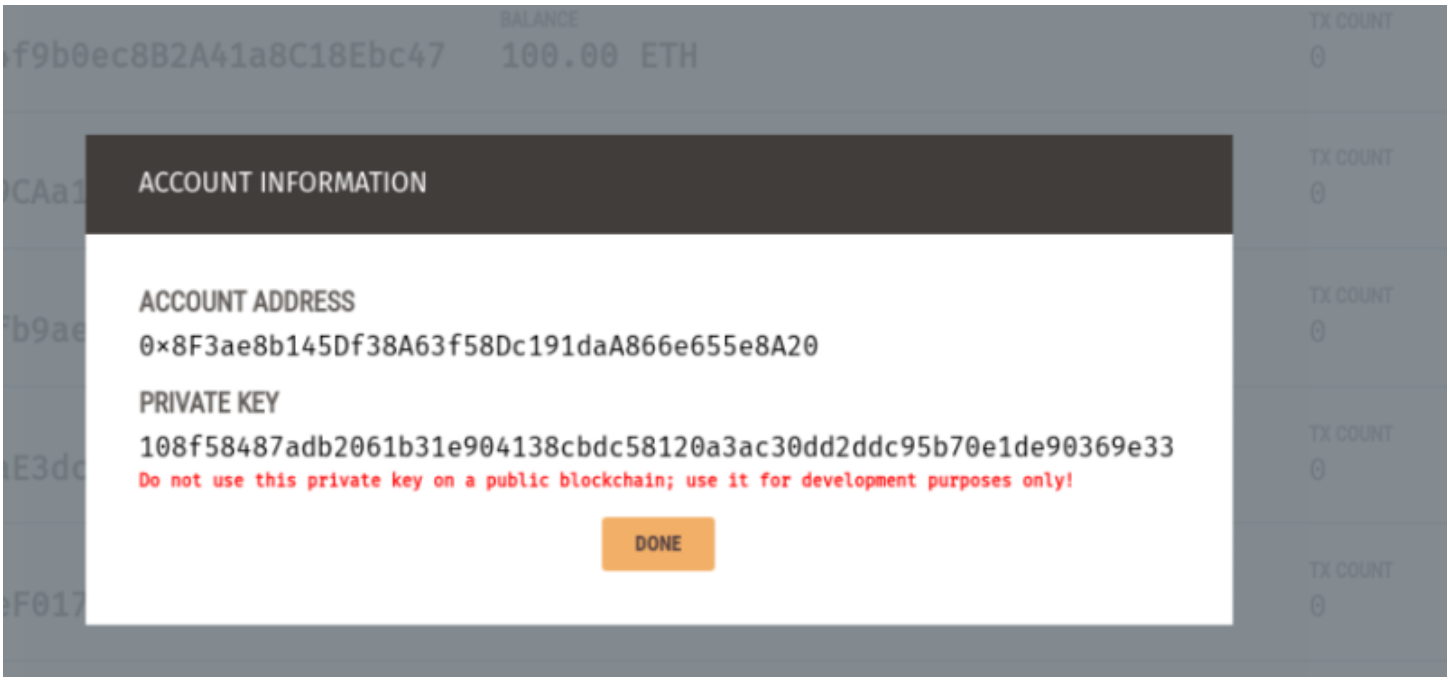
ganche:

click new workspace to save ur settings after u quit



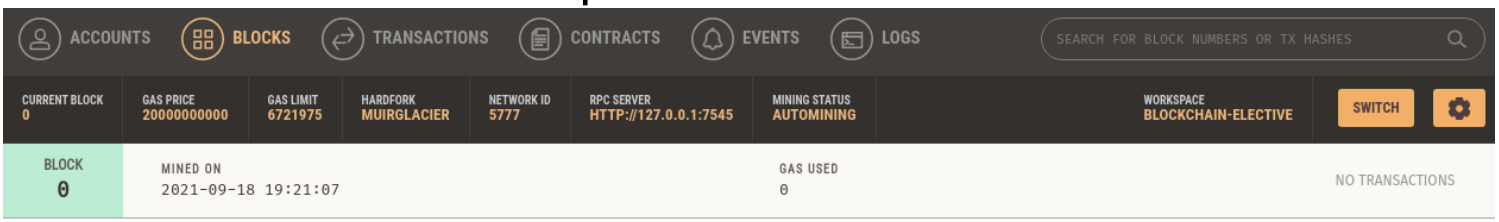
you have on opening

- 10 wallets with each 100 eth
  ◇ private keys are hidden behind the key icon next to the index



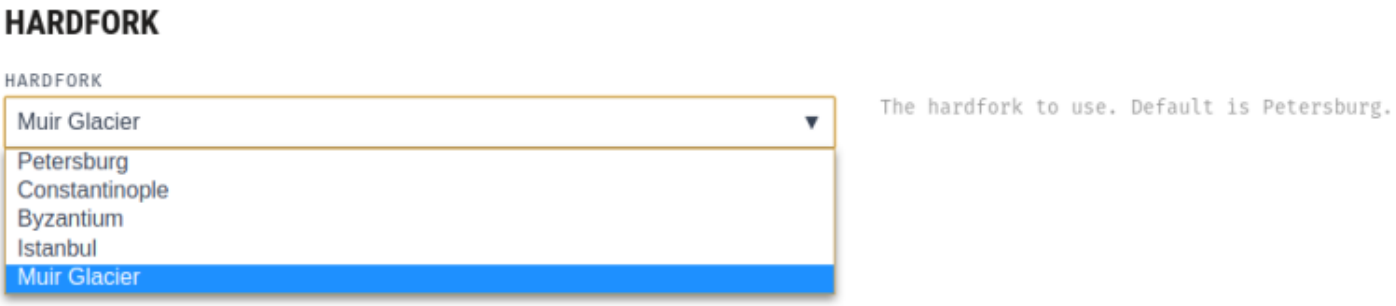
- HD wallet with eip44
- a miner which is on automine
- connection to remote process server

- in built blockchain explorer



•workflow
use truffle framework to write smart contract, compile on inbuilt solidity compiler and deploy it on ganache
- names of ethereum hard forks ie different incompatible

versions, latest hardfork is muir glacier



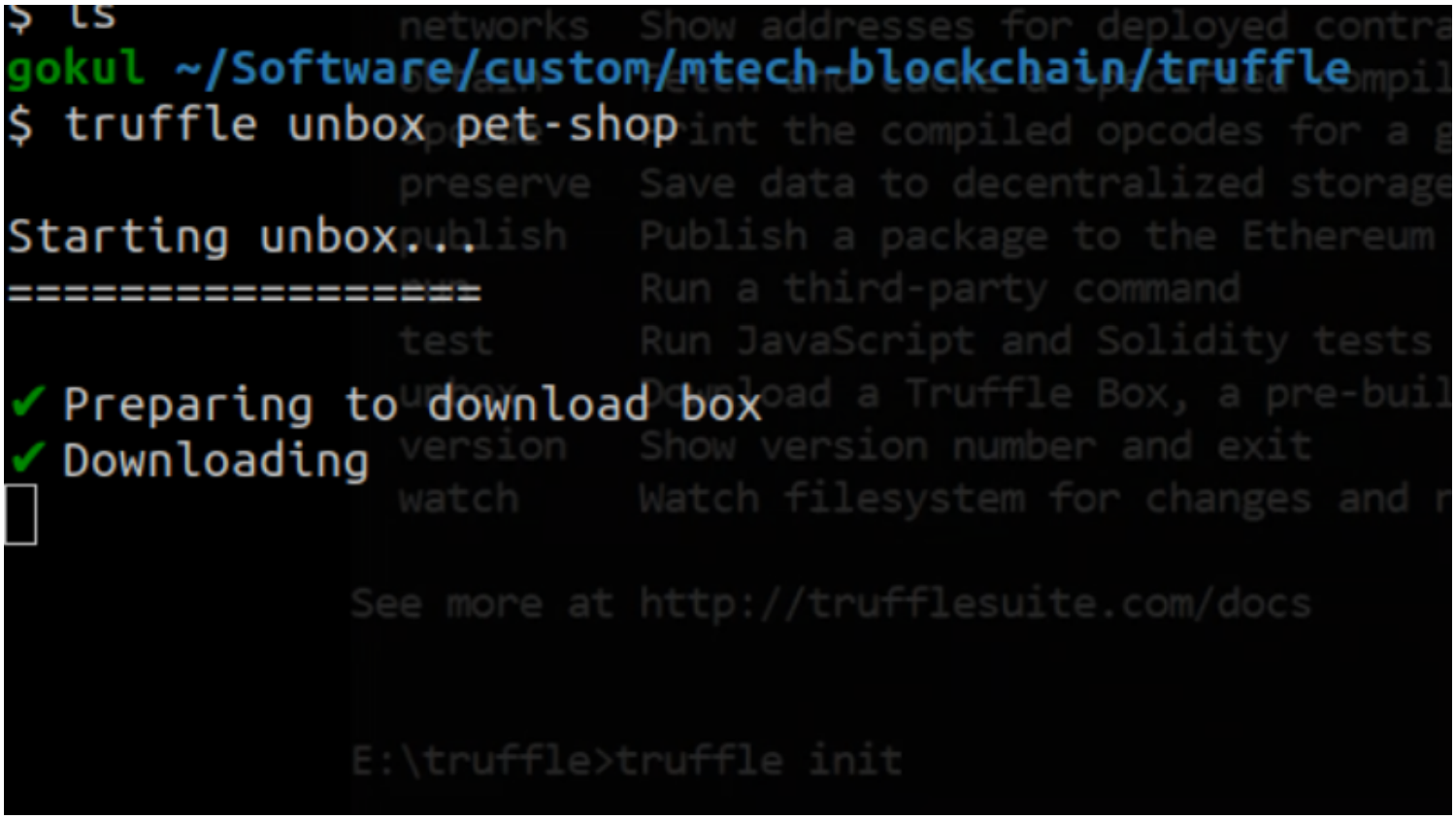
- has two transaction types
  - ◇ contract call
  - ◇ contract deployment

installing and working with truffle

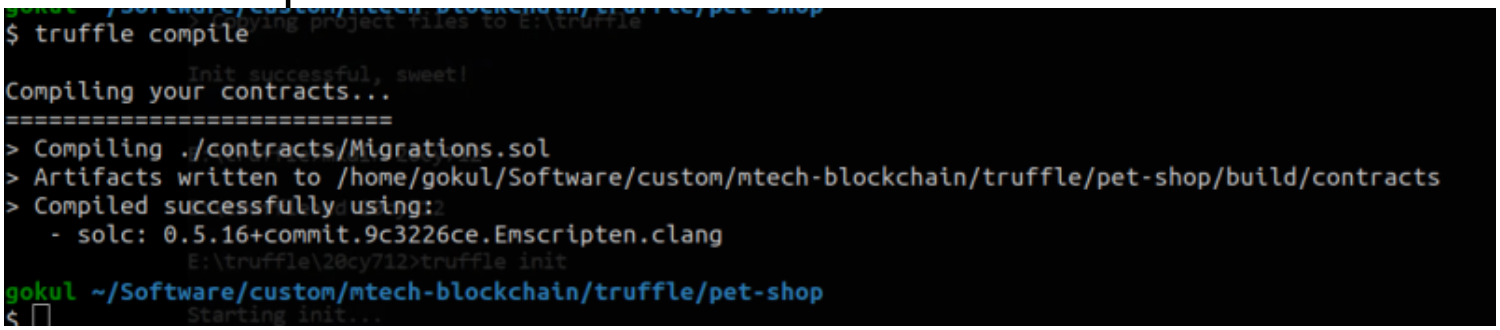
- install nvm, use it to update to latest stable version of npm and use that version
- then install truffle globally, npm install truffle --global
- then create a blank project
  - ◇ truffle init
  - ◇ in that truffe configjs file is there which helps to talk with ganache
  - ◇ test folder contains your custom test cases
  - ◇ migration folder is where you write code to deploy to ganache
  - ◇ contract folder is for smart contracts

we can download a preconfigured and working project called a box to test out application

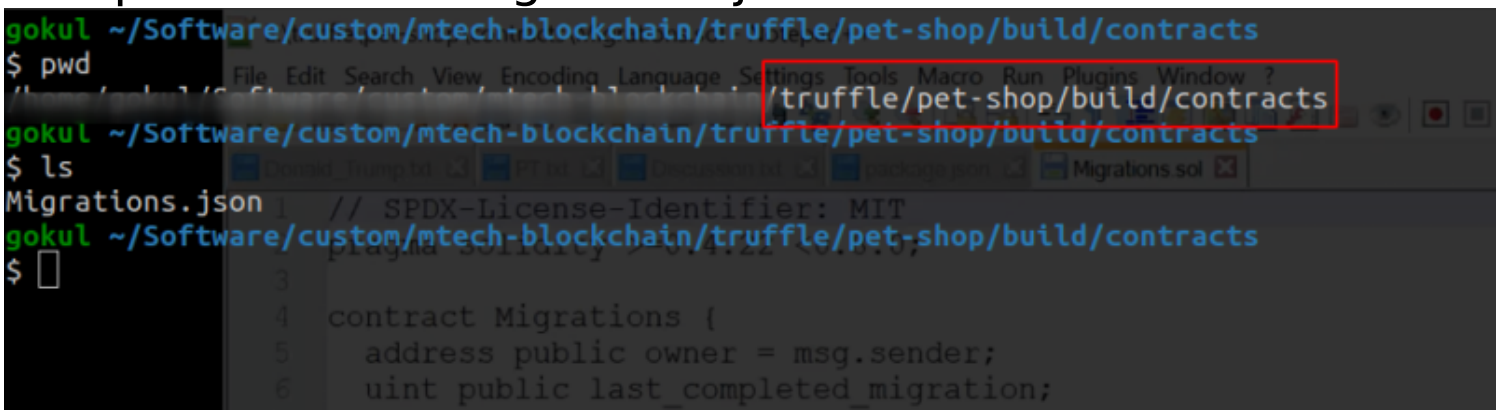
- the box we use is called pet-shop
- command truffle unbox pet-shop



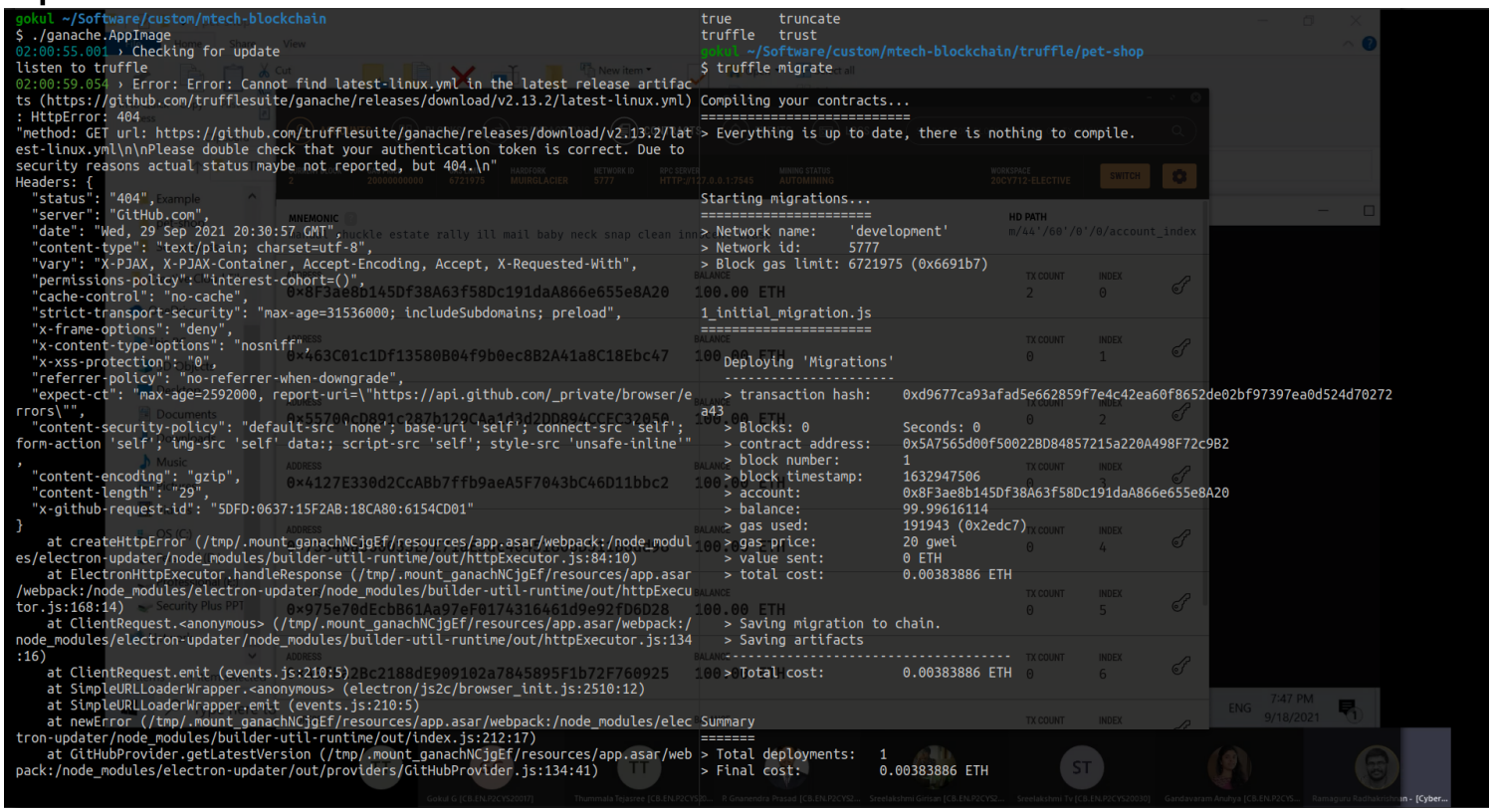
to complie all contracts within contract folder
truffle compile



and compiled output is written to build folder into name-of-contract.json, as we had only migrations.sol, the compiled code is migrations.json



now we can migrate/deploy the json file using truffle migrate, we should have ganache (mine is an appImage) open for this

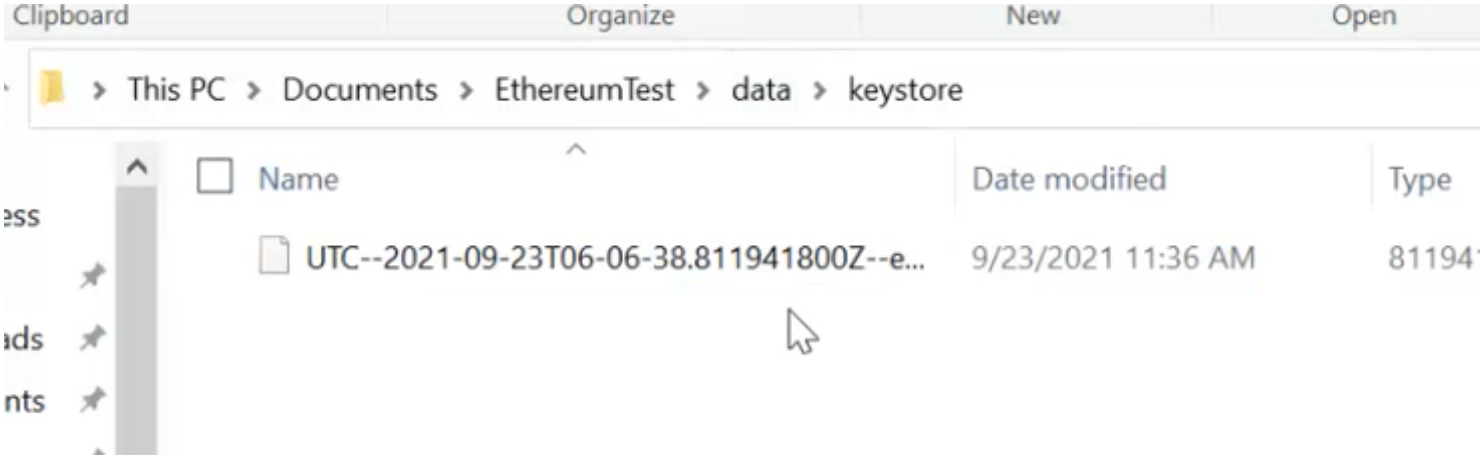


for unit testing
truffle test

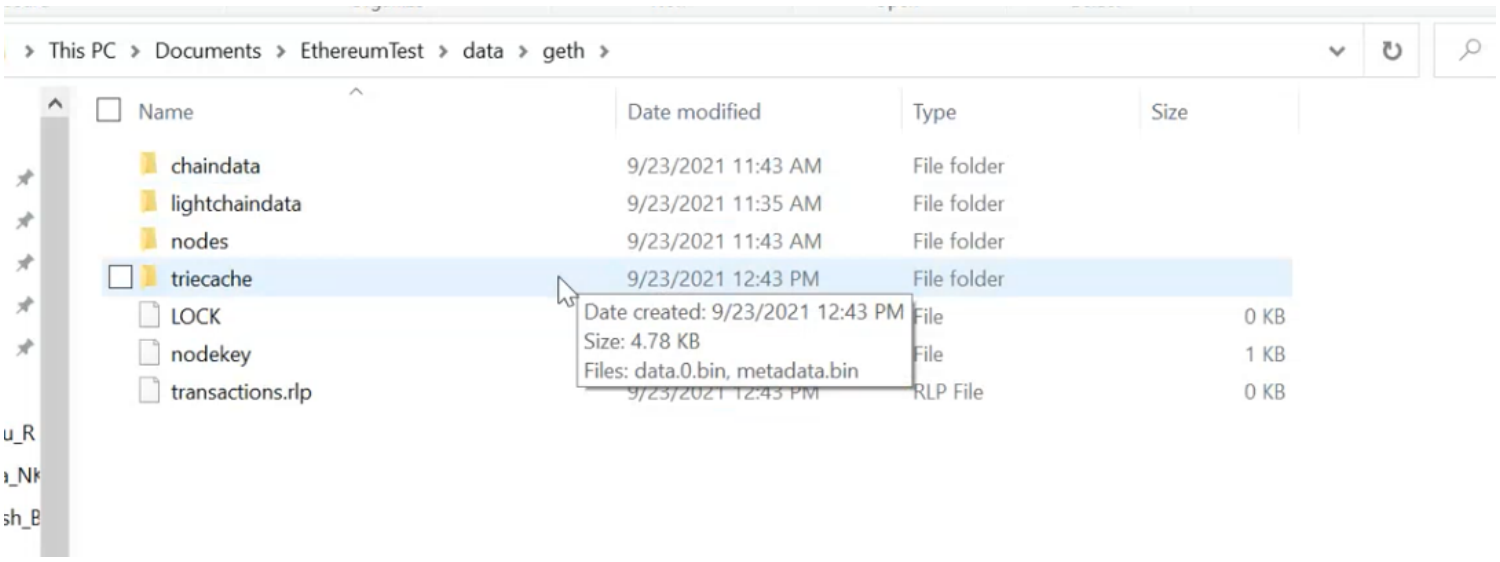
for deploying for testing locally:
npm install //for install all dependencies
npm run dev //running on local host

```
npm install -g truffle
truffle unbox
truffle init
truffle compile
truffle migrate (deploy)
truffle develop
truffle console
truffle debug
truffle test
truffle publish
```

lecture -13



keystore for account related data

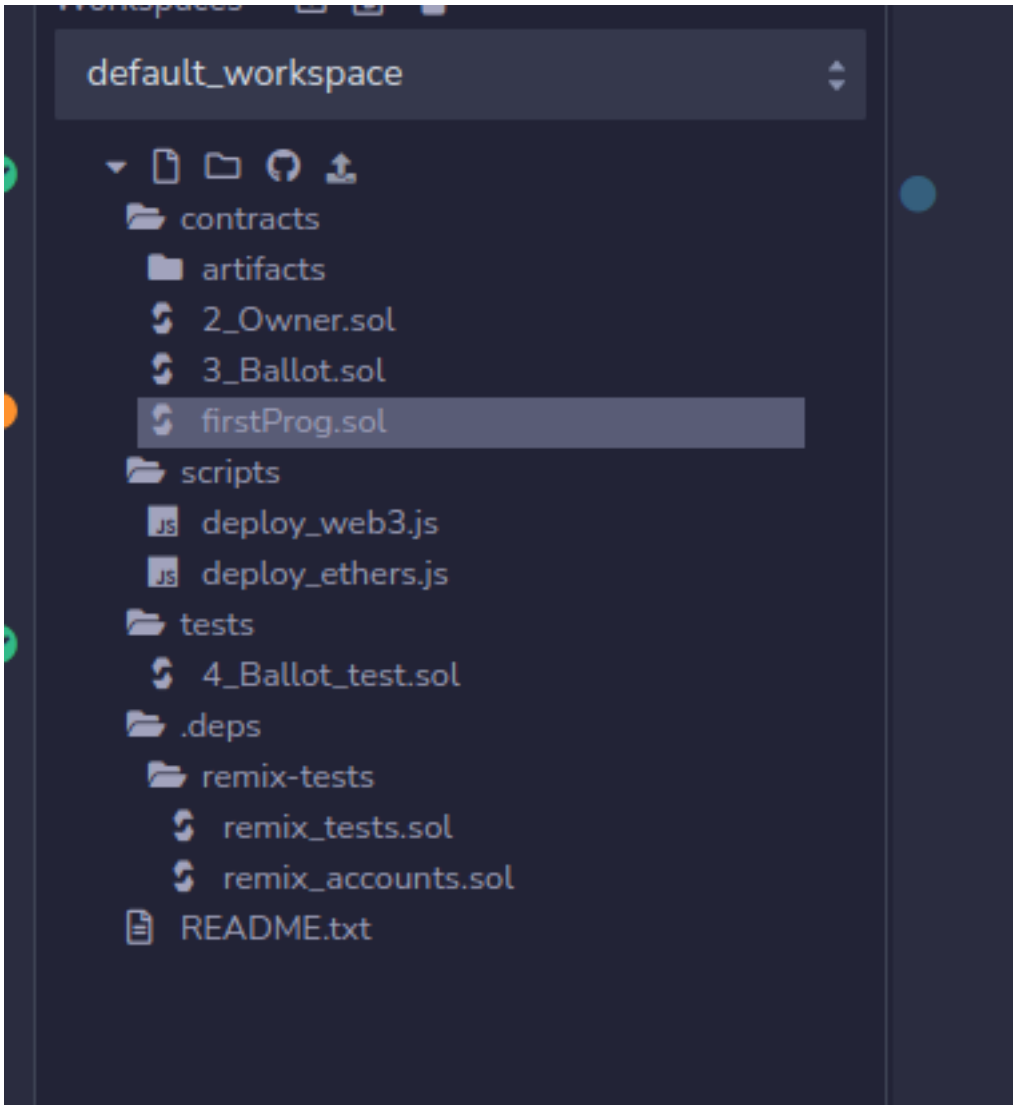


data folder for geth
<need to finish, not completed.>

lec-14 pending

lec-15

ganache: A private blockchain which has built in wallet and stuff
geth : A low level client to access ethereum blockchain
DApp: smart contract when it gets deployed on blockchain we call it DApp



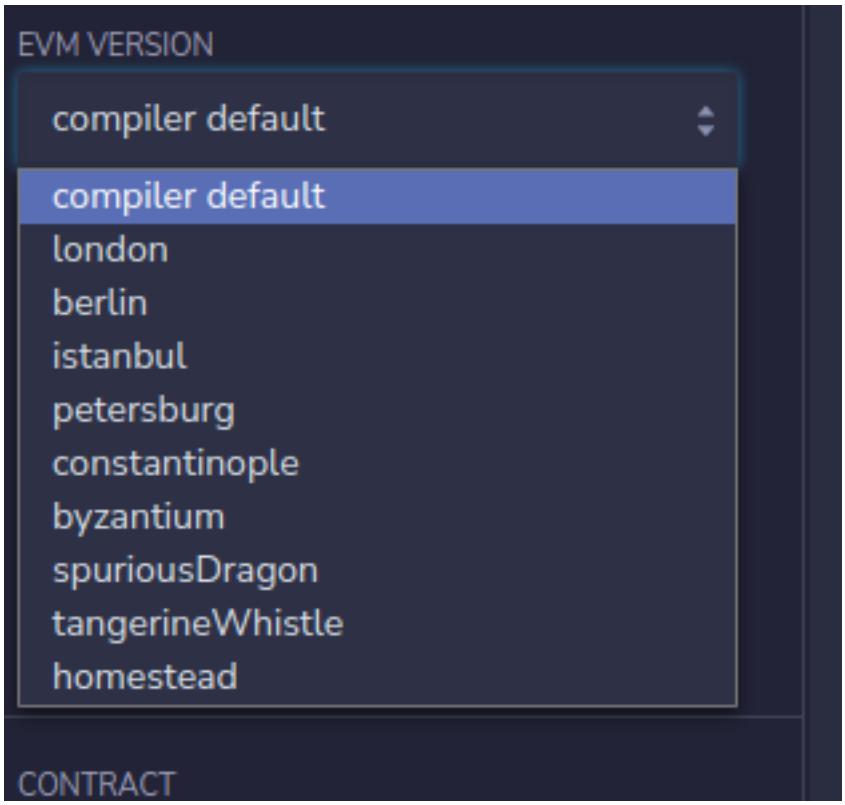
\

in remix ide, we always have three folders

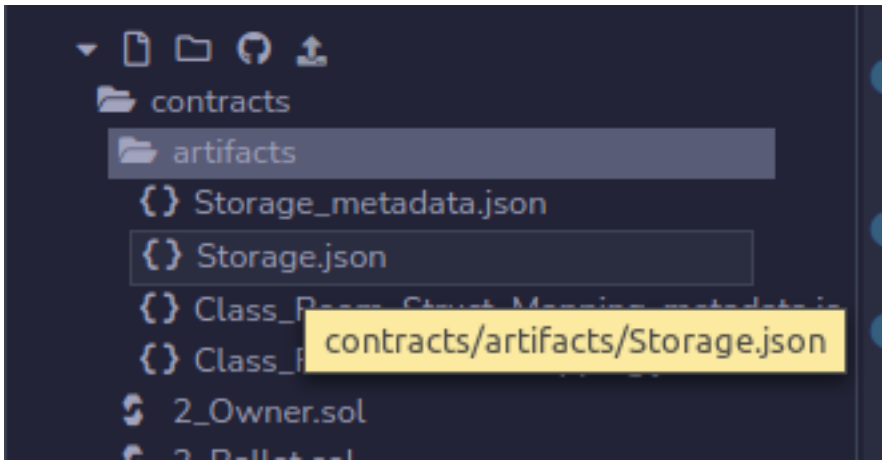
- contracts
  - ◇ write your SC here
- scripts
- tests

```
// SPDX-License-Identifier: GPL-3.0

pragma solidity >=0.7.0 <0.9.0;

/**
 * @author Ramaguru Radhakrishnan
 * @title Class Room with Structure & Mapping
 * @dev Store & retrieve a student details
 */
```

pragma solidity tells what lang version should the choose, ie between 7 and 9 anything is fine



EVM machines:
homestead is oldest version
london is newest version
in truffle when we compile we have build folder
in remix we have artifact under contract folder



- byte code is basically the compiled output of solidity execution
  - ◇ byte code is machine independent
  - ◇ this is stored in blockchain as transaction
- ABI
  - ◇ this code is generated in *_metadata.json file under artifact folder

• in other programs, you have to compile, run test and debug then only can you deploy the program
• in solidity you have to deploy (in blockchain)first then you can run

how to write solidity programming:

this is a mandatory comment:
// SPDX-License-Identifier: GPL-3.0

pragma solidity to specify version

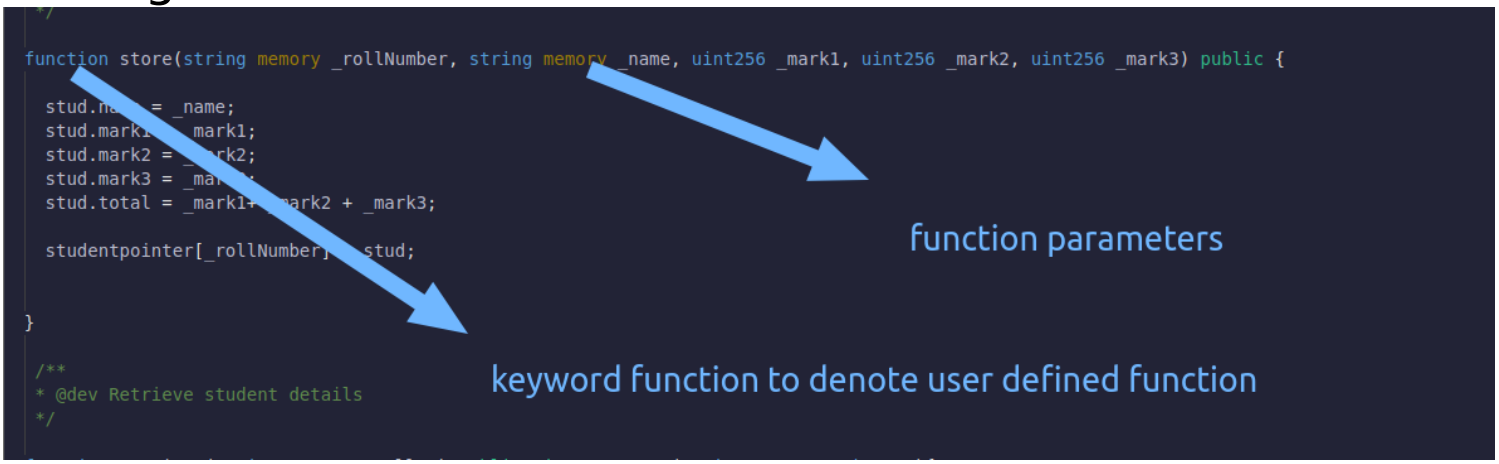
contract:
• equivalent of class in other OOPs languages
• this is the smart contract

all programs end with .sol

how to store variables in block chain?
• define a state variable in a contract and assign value to it
  ◇ assign value by putting direct equal to
    ■ string name = “gokul”
    ■ or write a dedicated function for it

```
 9    */
10   contract Class_Room_Struct_Mapping {
11
12       mapping(string=>student) studentpointer;
13       //Key - uint256 Roll No
14       //  studentpointer[_rollNumber] = stud;
15
16
17
18       struct student {
19       //State Variable
20       string name;
21       uint256 mark1;
22       uint256 mark2;
23       uint256 mark3;
24       uint256 total;
25
26       }
27
28       student stud;
29
```

writing functions

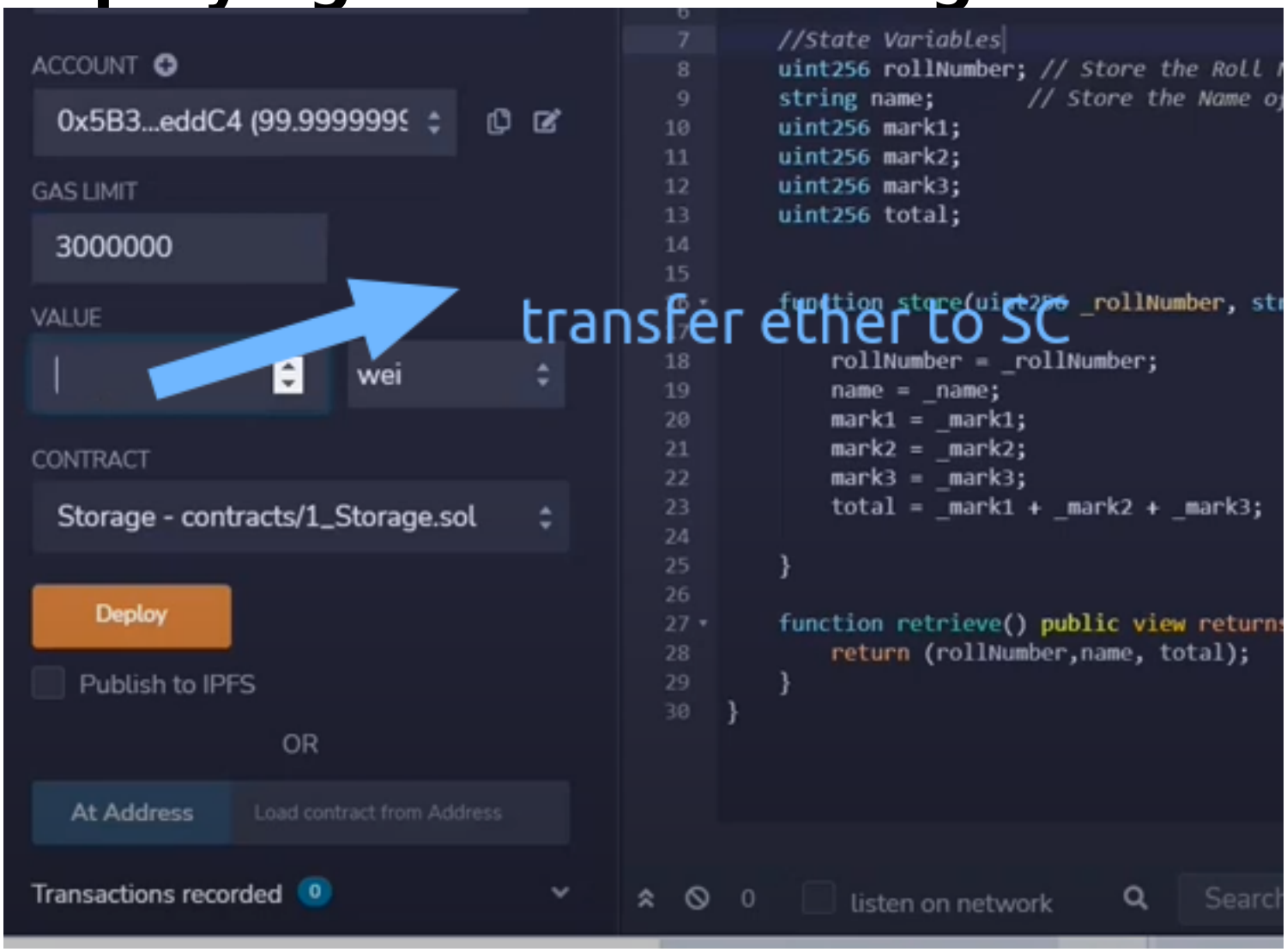



```
function retrieve(string memory _rollno) public view returns (string memory, uint256){
    student memory stud1 = studentpointer[_rollno];
    return (stud1.name, stud1.total);
}
```

view keyword is used so that end user can view the

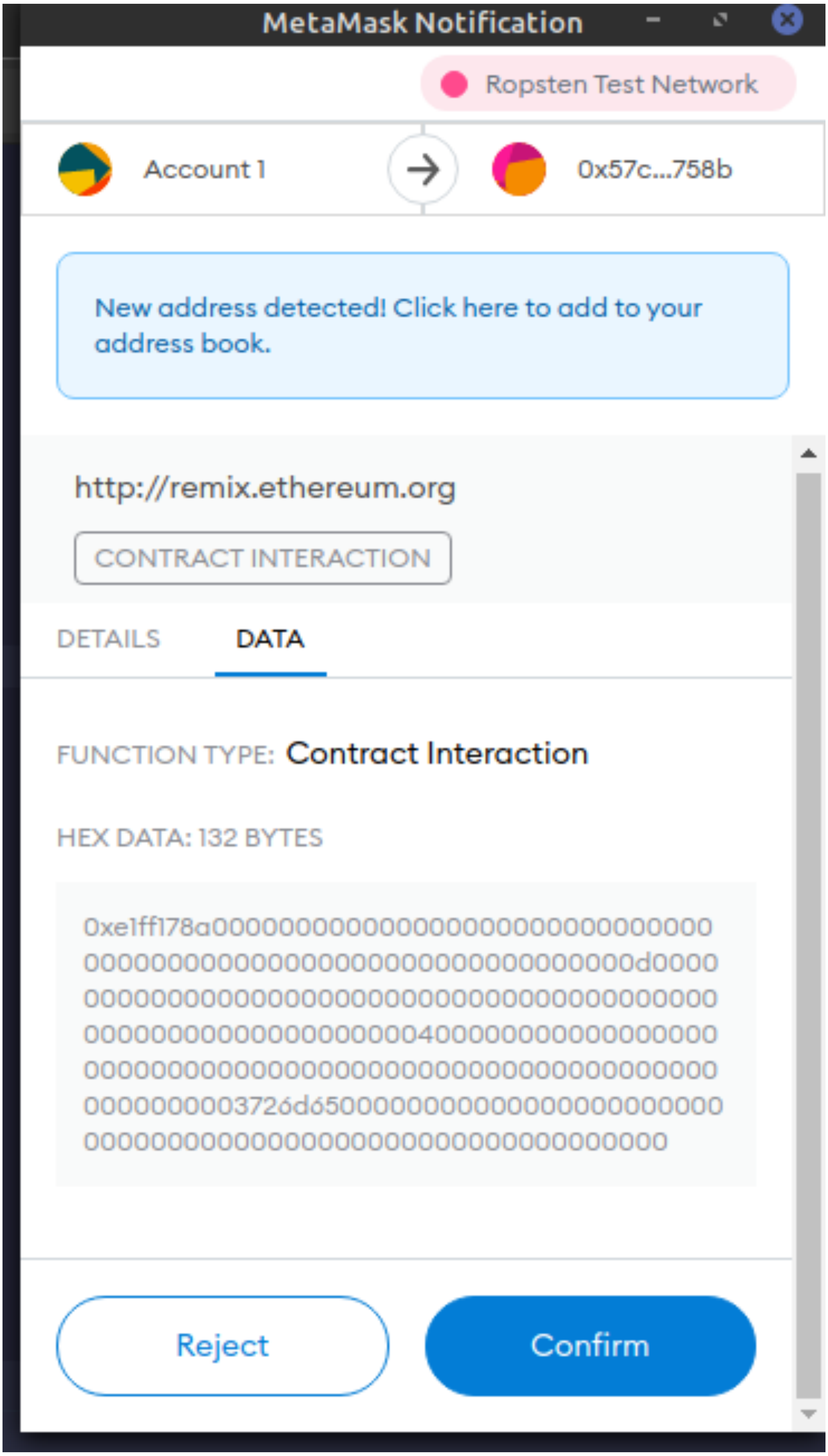
function result in Dapp
function maybe public ie can be interacted with
if u put private then nothing is visible
• other stuff we will see later are payable, internal and external
• keyword memory is used to mean vairables are only to be stored in program temp and not in blockchain
• you need not do that for integer datatypes

deploying in non EVM things



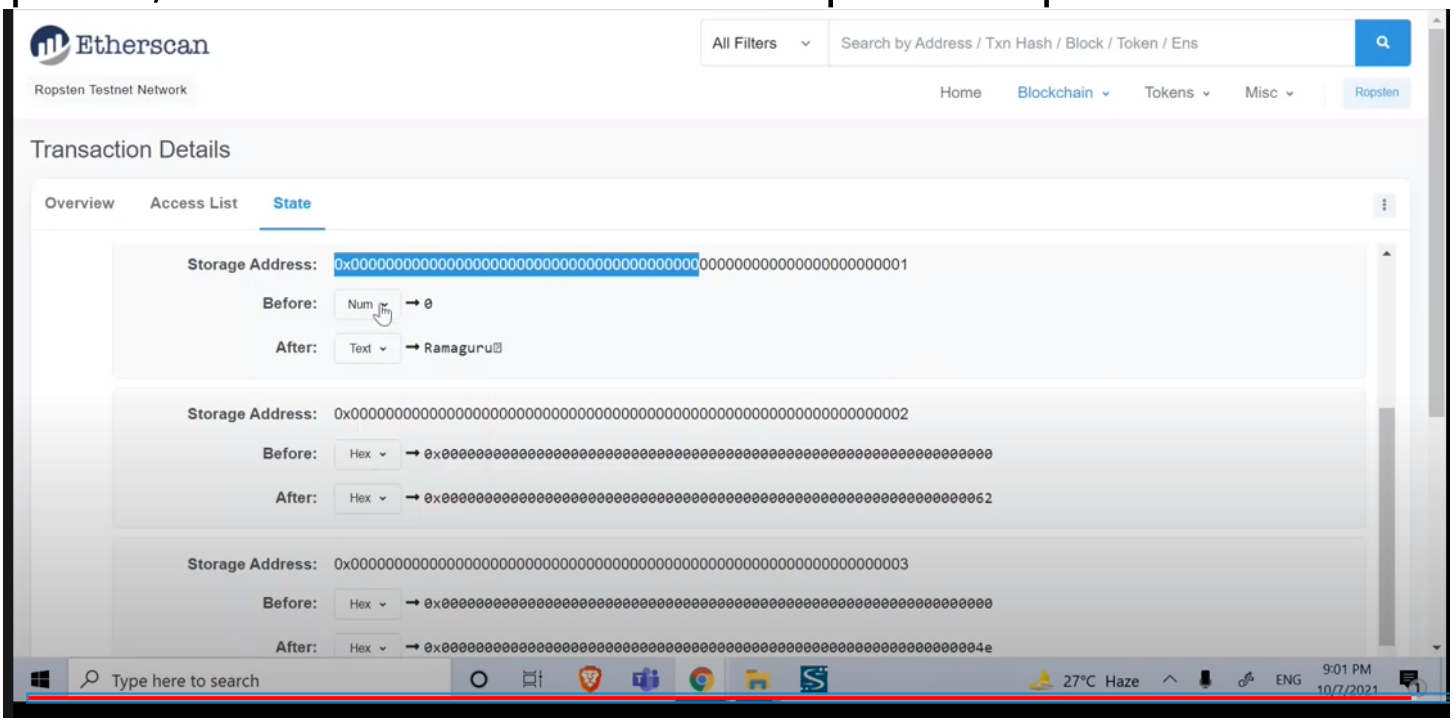
• java script vm is getting deployed in browser
• injected web3
  ◇ deploy it in testnet

■ automatically your metamask will open
■ when u deploy or interact with SC you have put to put in ether



• to store you are pefroming a consensus which requires money
◇ all members verify the transaction

◇ and store it among themselves
• while retrieving you are retrieving it from one of the nodes so no money required
• since this is a sc deployed in blockchain for example ropsten, this can be seen in the ropsten explorer
•

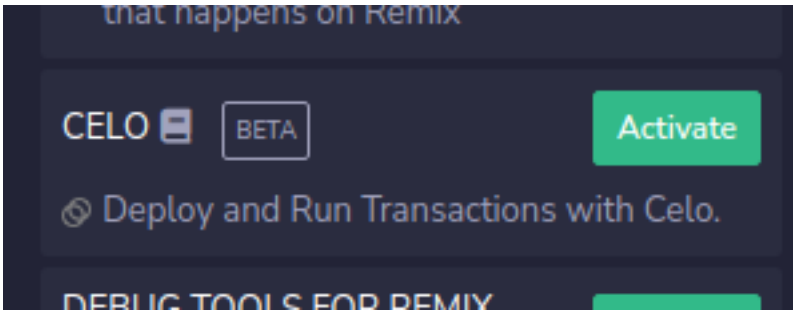


lec 16

ABI -> Application Programming Interface
• in order to talk to a SC deployed on blockchain you need ABI
• similiar to API in centralized web
• include this in ur front end and using this talk to smart contract

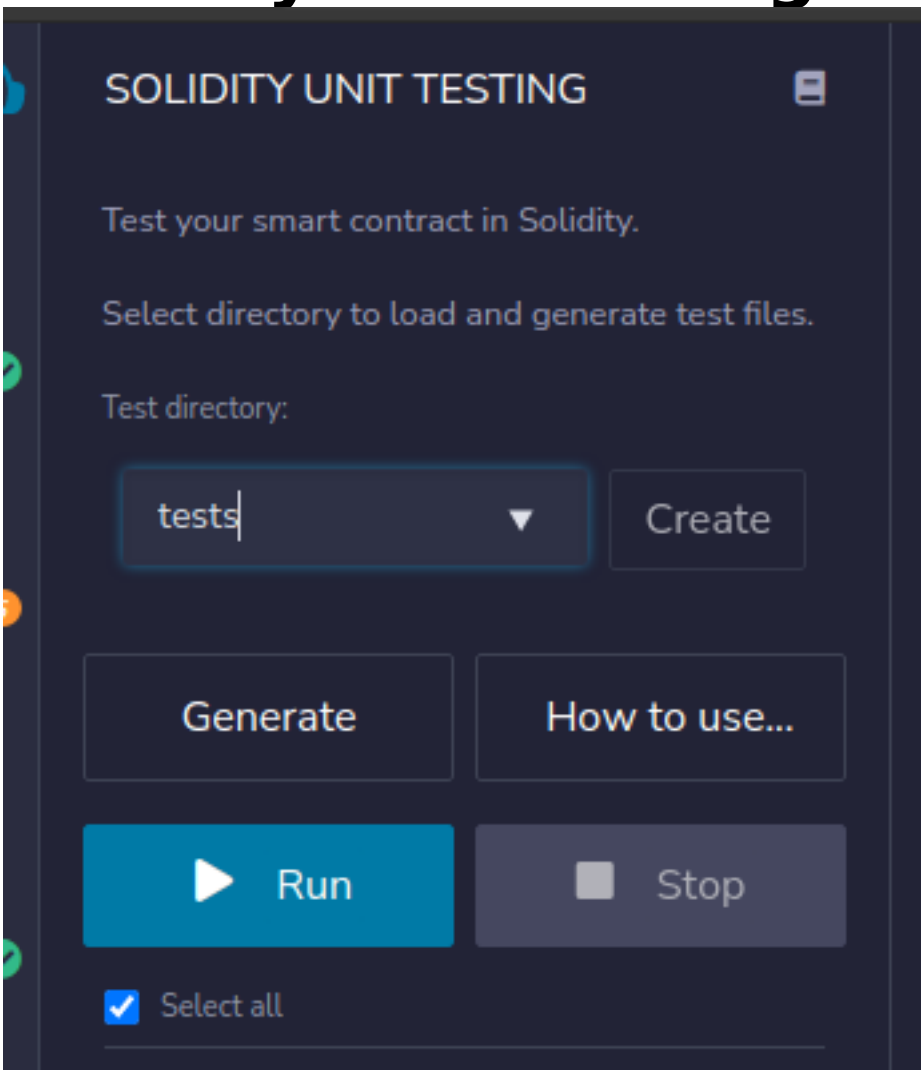
plugins

• celo
◇ hyperledger project
◇ write an SC and deploy on celo

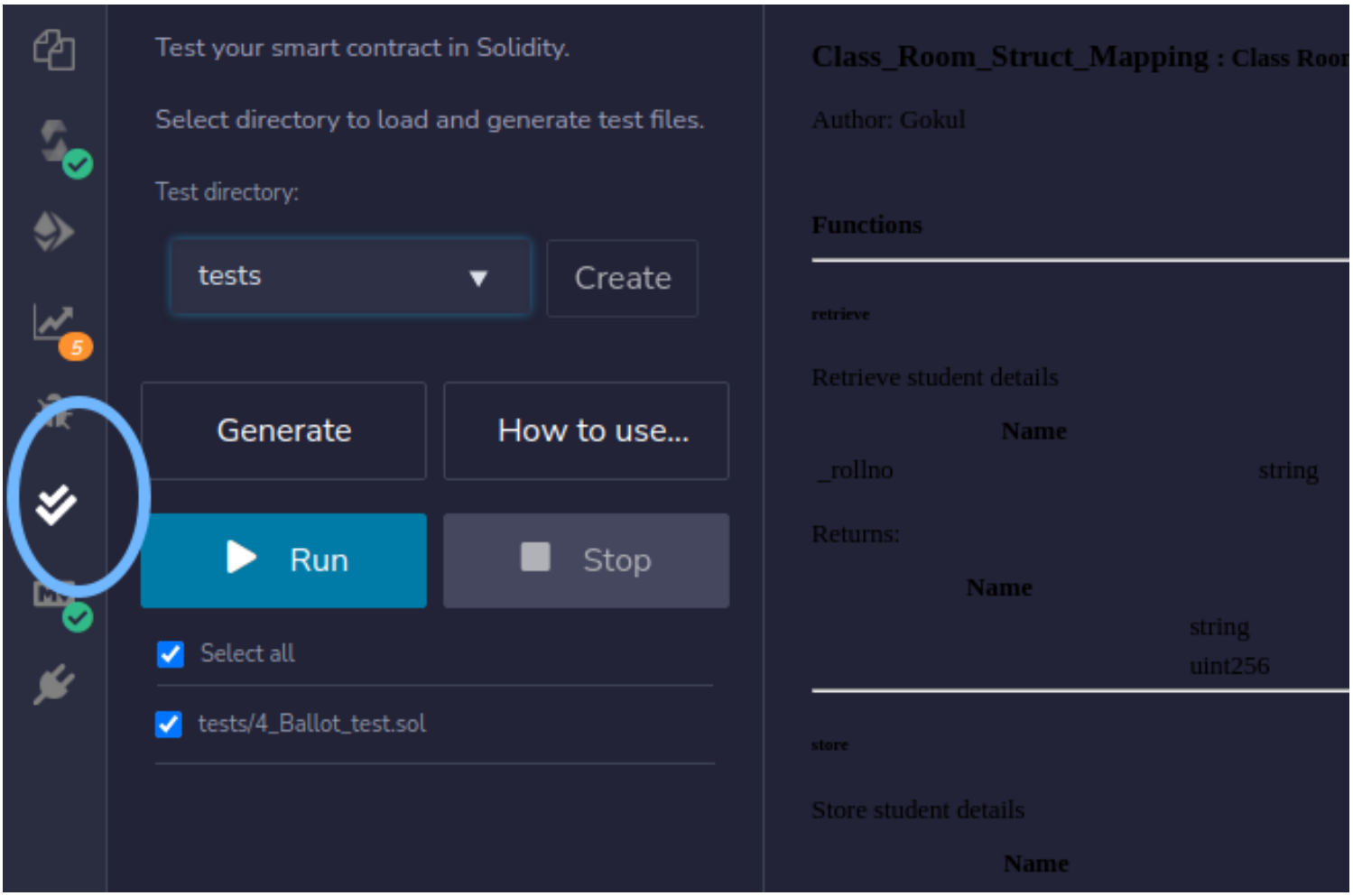


- ethdoc- documentation generator
  - ◇ generate documentation for your program
  - ◇ and publish to IPFS
- solidity to UML
  - ◇ generates UML class diagram for SC

Solidity unit testing

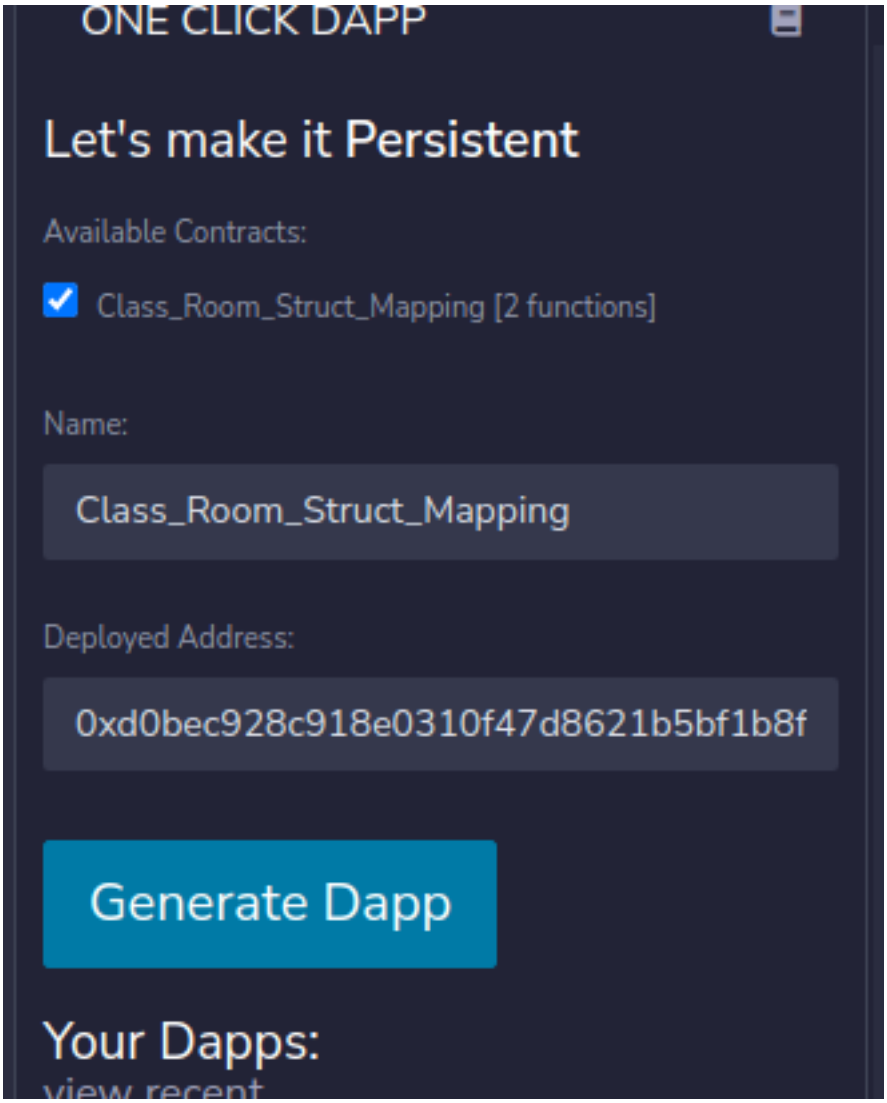


- you can run your test cases here by clicking run



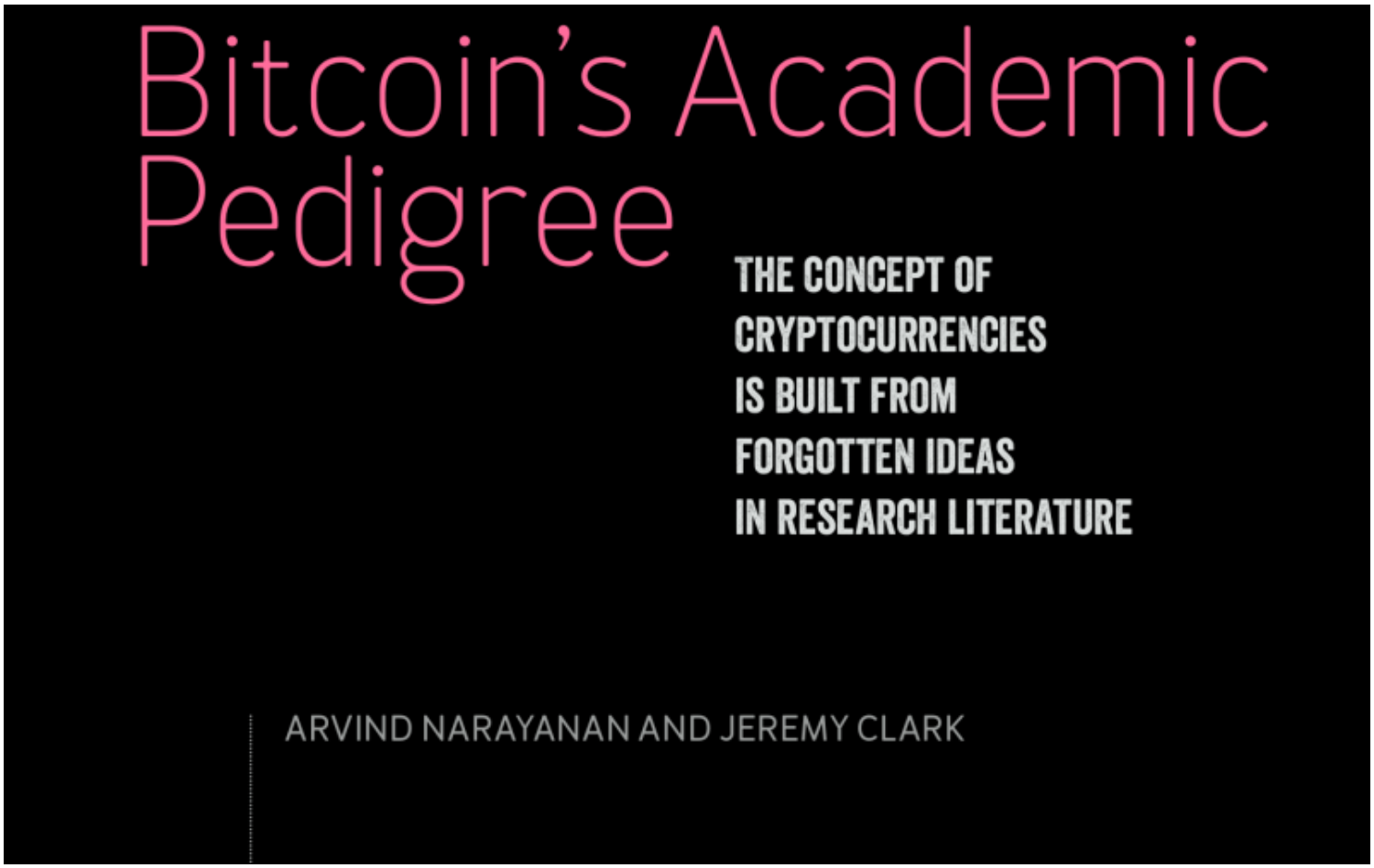
lec 17

- whenever you need to talk to a SC
  - ◇ you need smart contract address
  - ◇ ABI
- you can “publish” your code with minimal UI elements to IPFS using one click DApp
  - ◇ this is a plugin
  - ◇ deploy it on testnet/mainnet
  - ◇ copy the smart contract address, and paste



24:53

bitcoins-academic-pedigree



Primer

• misconception
  ◇ David Chaum tried to popularize anonymous digital payments via DigiCash, but didnt succeed
    ■ it required a centralized bank like server controlling the system and no bank volunteered
  ◇ Satoshi nakomoto invented bitcoin as a response to this
    ■ he was not in academia, and this was a sudden radical solution
• truth
  ◇ bitcoin was the result of smartly incorporating many fundamental ideas(which were many years old) of decentralization and blockchain to benefit mankind
  ◇ this explains why bitcoin took so long to create.

Ledger

- Ledger is a way of recording transactions
  - ◇ one application is to simplify digital payments
  - ◇ Alice send Bob $100 by paypal
    - ■ Paypal has a ledger which says $100 taken from alice account and deposit in bob account
    - ■ absence of single ledger between banks complicates the process if directly we do bank-bank transfer
- Bitcoin has a publically verifiable open ledger
  - ◇ trusted by all users
  - ◇ bitcoin uses this system for recording payments to generate currency
- Bitcoin ledger needs to be once written never rewritten
  - ◇ ie immutable
- should be able to obtain a hash of the blockchain at any time
  - ◇ ie any user can verify the integrity of the chain
  - ◇ ledger is global data structure collectively maintained by a group of untrusting nodes
- another approach is to decentralize the ledger by making the entire participants download the entire ledger locally and update and maintain it.
  - ◇ it is up to the user querying this set of ledger to resolve any conflicts, not the rest of the nodes

Linked time stamping

- bitcoin blockchain data structure is borrowed from haber and scott
  - ◇ reference paper was for making a digital notary service

◇ for a system to say with authority that the document was created at certain time, and no later
• methodology of paper
◇ documents are created and at the same time broadcasted to all
◇ author of document generates time stamp of document
◇ the document, the time stamp and the previously broadcasted document is signed together by author.
◇ the previous document and the document created after this is also having a similar digital signature
◇ this creates a document chain with signature pointers asserting integrity of all its contents
◇ the creator cannot assert a falsified document to be the previous document as the same would be reflected in the future documents of the chain
◇ no outsider can alter the content or timestamp of any document as he would have to redo the entire chain
■ he would also have to find the secret key of the creator
◇ thus if the timestamping service is trustable and non-biased the entire chain up to that point is
■ immutable
■ temporally ordered
■ if system rejects documents with incorrect time stamps then validity of time stamp also preserved
• paper 2.0
■ instead of digital signature, hash was used
- called hash pointers
■ instead of using single documents, multiple documents with the approx same timestamp was used
- called blocks
■ instead of having a linear chain inside of block,

binary tree Datastructure is used
- called merkle tree
• satoshi took these ideas and introduced the proof of work and changed its security properties

Merkle trees

• in bitcoin, transactions are used in place of documents
• leaf nodes are transactions, internal nodes are having two pointers
• has two properties
◇ hash of the latest block acts as digest for verifying integrity of the entire block
■ a change to any content of the block's leaf node would mean we have to rewrite the whole block which means rewrite the whole blockchain up to that point
■ if you know the hash of the latest block you can use it to verify the ledger if you got a copy of it from untrusted source
◇ we can prove the existence of particular transaction in the ledger
■ send small number of nodes in that transaction's block
■ send some information regarding all the blocks that come after the transaction's block

Byzantine fault tolerance

• blockchain needs to be fault tolerant and be aware of intentional/unintentional forks
• Byzantine fault tolerance assures blockchain can operate as intended for “non-fault-stop” failures
■ failures which is not serious enough to pause the

system
- known as “traitor nodes” sending wrong messages
  - out of sync/malfunctioning nodes due to network latency or programming errors
  - malicious nodes

##

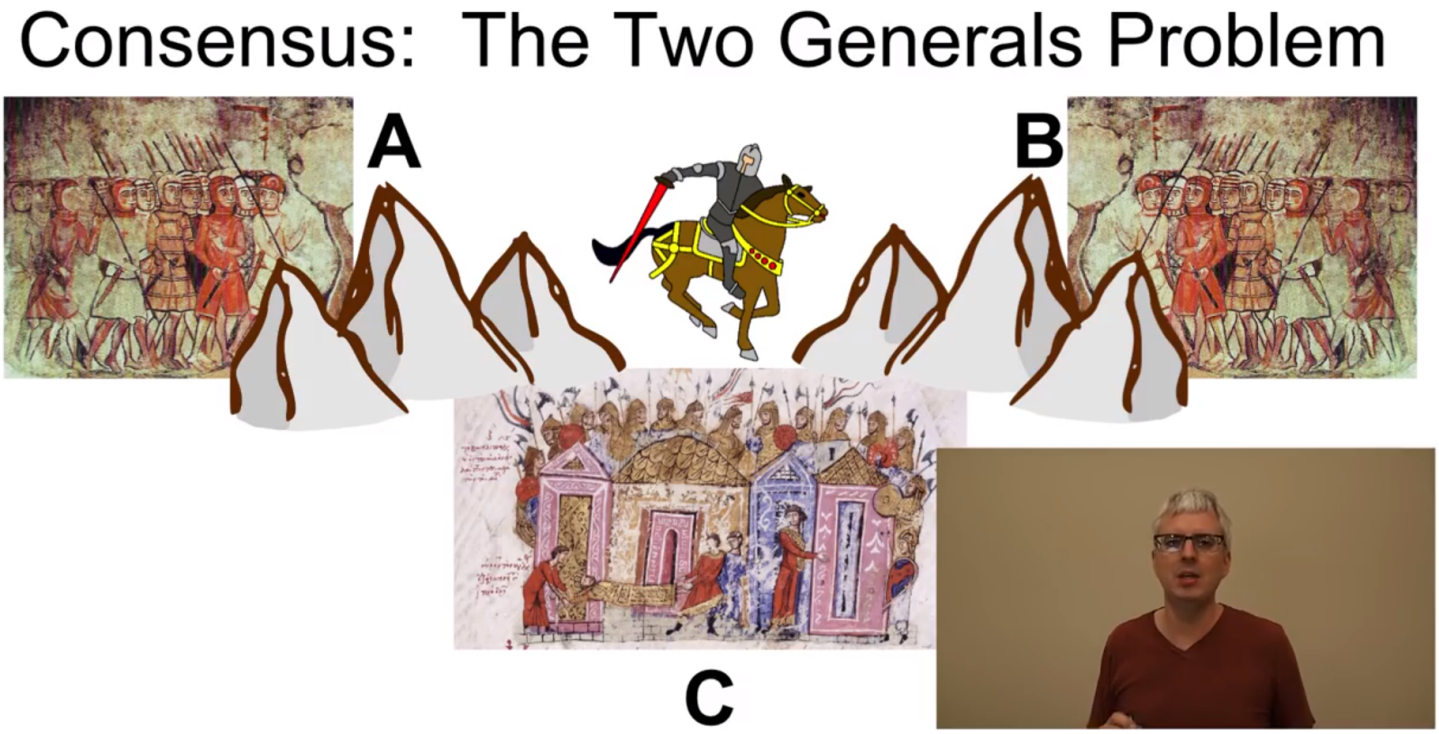
youtube video notes:

Assumptions/questions before designing byzantine systems

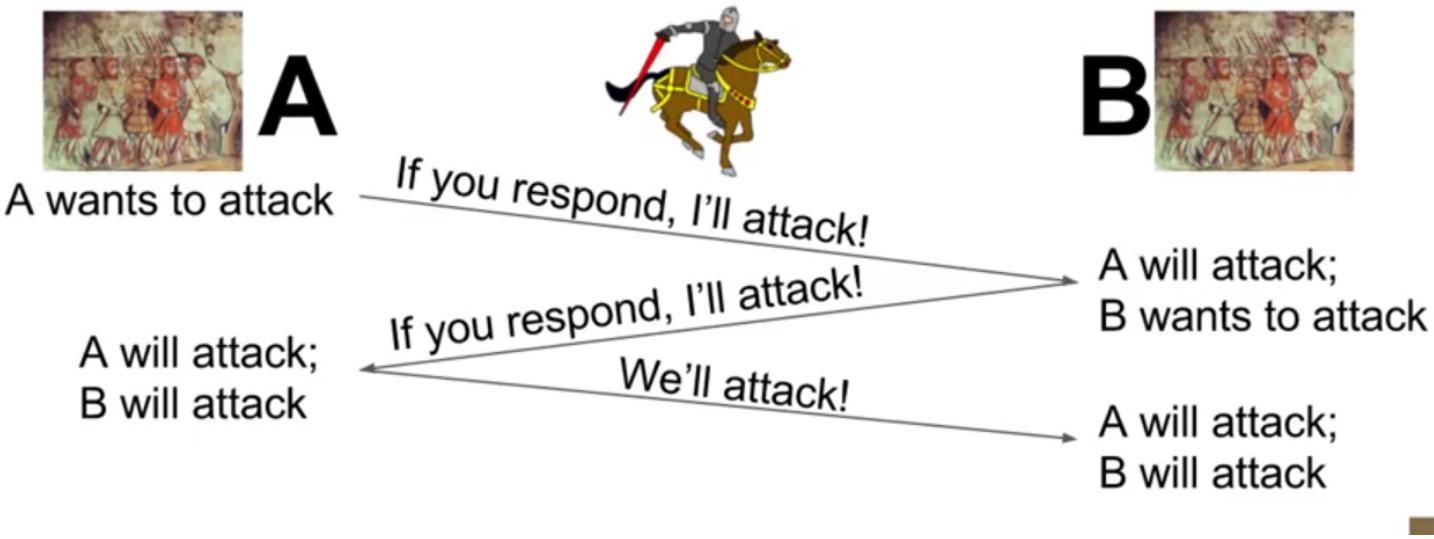
1) are all/some nodes seeing messages you send
   1- are nodes broadcasting info or selectively doing unicast
2) do your nodes and network fail?
   1- how reliable is your nodes and network to attacks/-failures
3) adversary capability
   1- how much computational resource (one laptop to an entire super computer cluster) your attacker have to attack your network
4) static or dynamic adversary
   1- is your attacker having only one tactic
   2- is your attacker having capacity to adjust to your defensive techniques and create attacks on the fly
5) bounded communication time
   1- are messages between your nodes time stamped?
6) fully connected network?
   1- are everything connected to everything?
   2- are you using relay nodes?
7) randomized algo
   1- is you or your adversary using randomized algo?
8) adversary using quantum computer or binary

computer?

fixed scenario - the byzantine two general problem



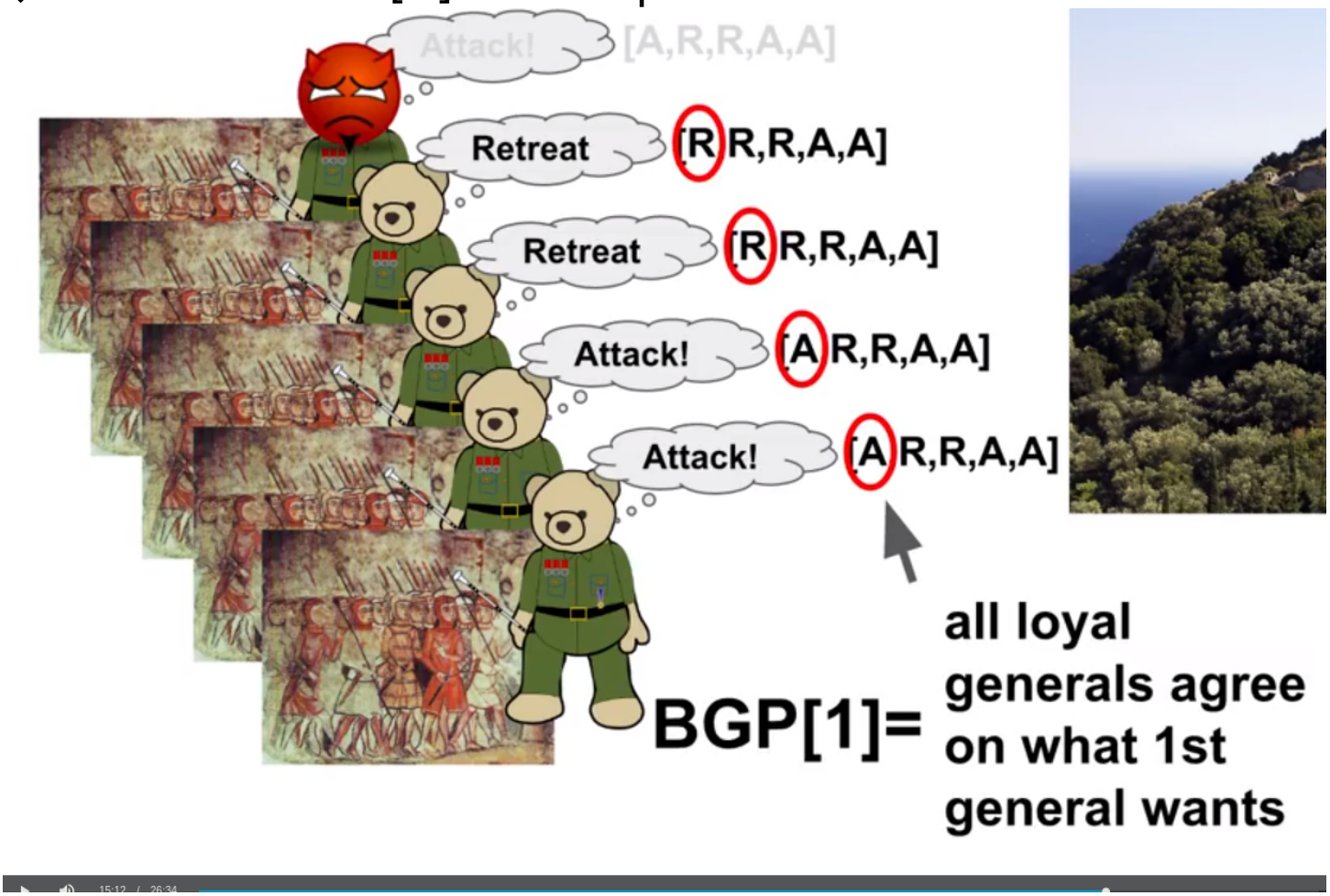
• army A and army B are lead by two generals
• both are in either side of battlefield , this field is controlled by army C
• the two generals want to make a common decision
  ◇ army A and army B attacks army C => army A, army B wins
  ◇ army A and army B dont attack and retreat
  ◇ if either one only attacks, army C will win
• the two armies can only communicate to each other via a horseman, who has to travel over the battefield and may be killed by army C
• the solution to byzantine problem is find a way for army A general to get the message that tomorrow we both attack or dont attack and have a way to verify that army B got the message



• in this case if “we'll attack!” gets lost, A will attack for sure and B wont, as it is waiting for A's response
• we can add as many buffers of “we'll attack” but still the same problem
• ie no perfect solution
• for a network
  ◇ army A and army B generals are two “innocent nodes” who want to communicate
  ◇ army C is the network which is fully under control of the hacker or untrustworthy
• Practical solution is to assume
  ◇ traitor node is not perfectly “byzantine"
    ■ ie statically it wont attack 100% of the time
    ■ ie one solution is to flood the network with “we'll attack” to let atleast one go through

fixed scenario Byzantine General's problem
• army A and army B generals are innocent nodes
• army C general is the malicious node(s)
• situation is how many byzantine nodes (traitor nodes)

can a network survive?
  ◇ tells one way of building of doing it
  ◇ doesnt answer “is the proposed way of doing it” worth it.
• multiple generals want to storm a fort
• they put votes
  ◇ all of them know each other's votes(either Attack -> A or Retreat ->R)
  ◇ majority wins
• if a malicious general is there and wants to fail the attack, he might say attack but might do retreat or vice-versa
• what to do if we want all loyal generals to agree on one specific general
  ◇ this is the BGP[1] subset problem



• questions
  ◇ how many traitors can we tolerate before decision is

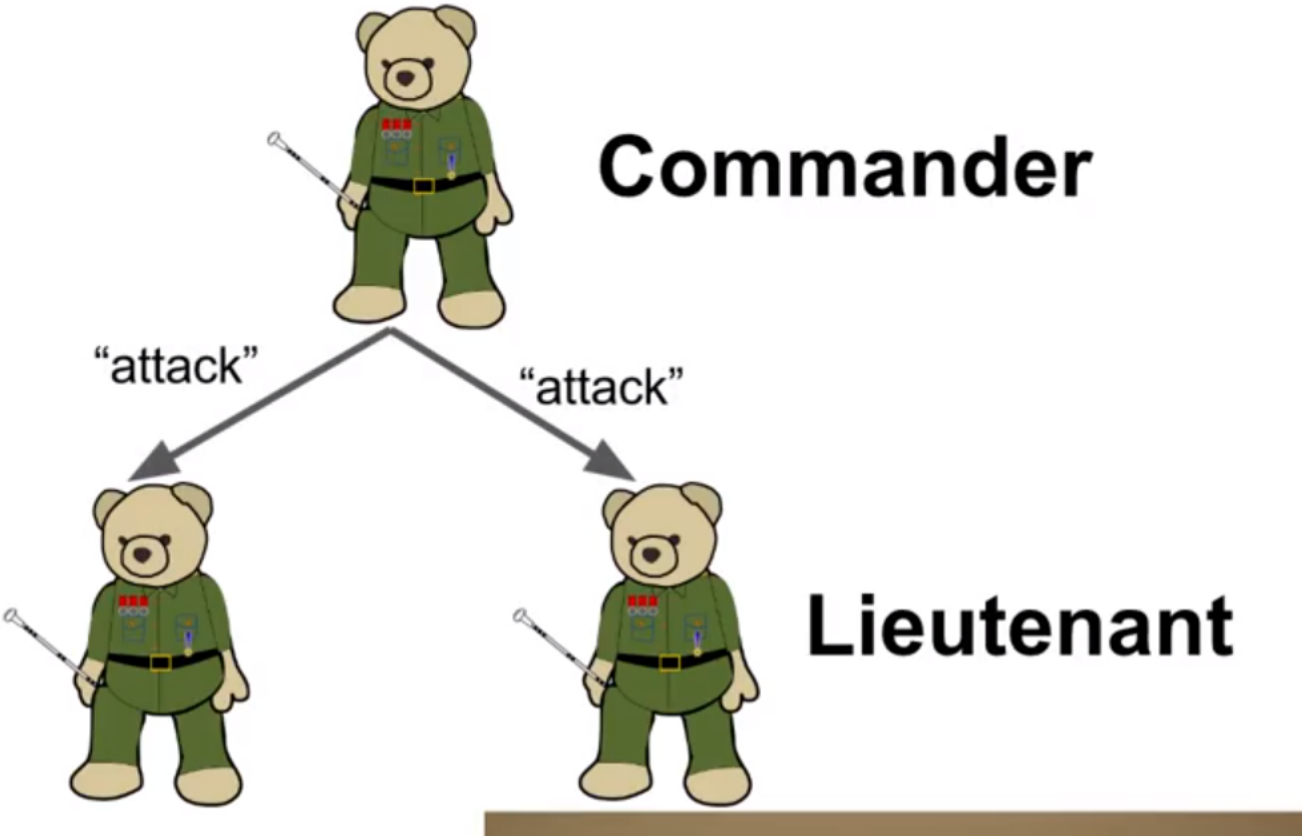
affected?

- ■ is it 1 general, 2 general etc
- ■ if there are n-1 traitors for a n node network consesus is not possible

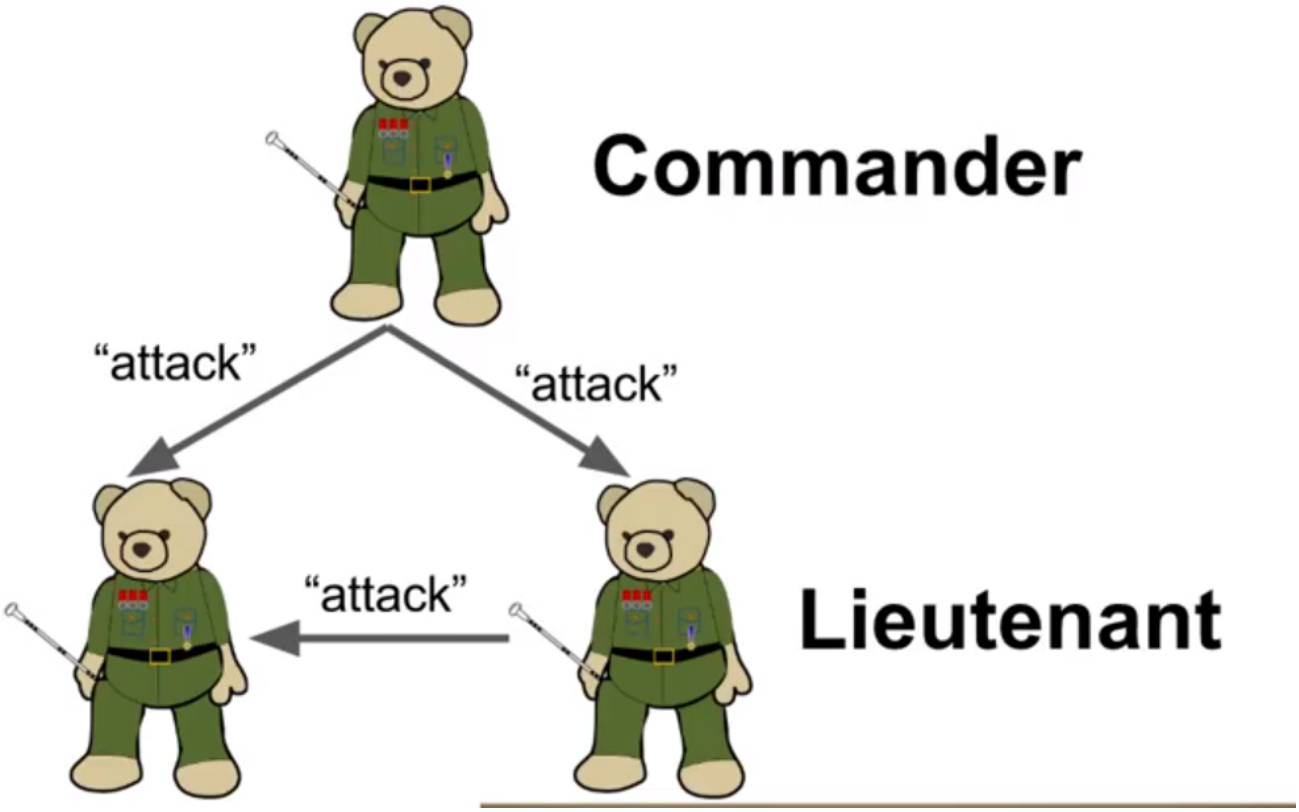
◇ assuming point to point communications
◇ no crypto

• trivial cases
◇ n=1 => no consesus possible
◇ n = 2 => no consesus possible
◇ n = 3

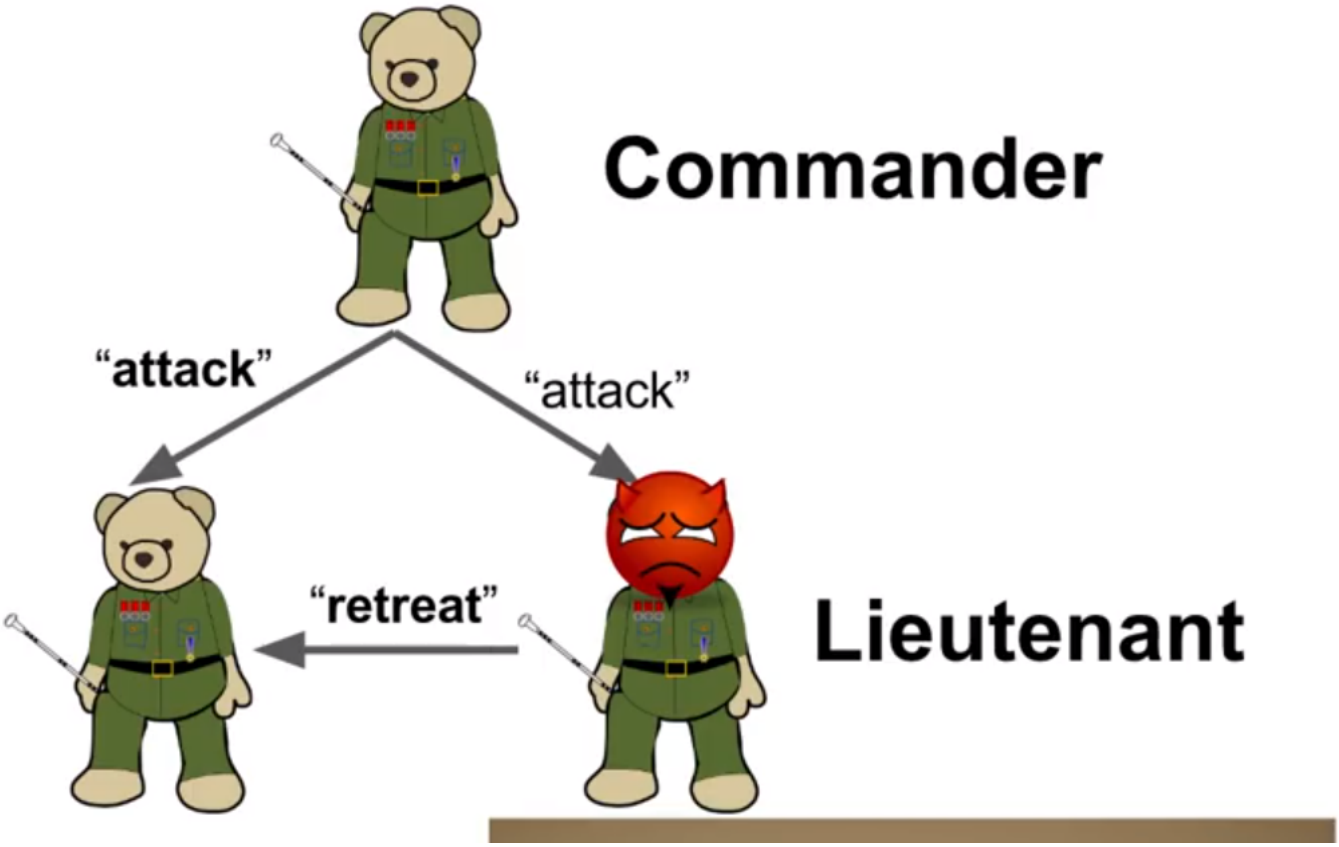
- ■ one commander main message
- ■ two lieutenant trying to agree on whether it is attack or retreat



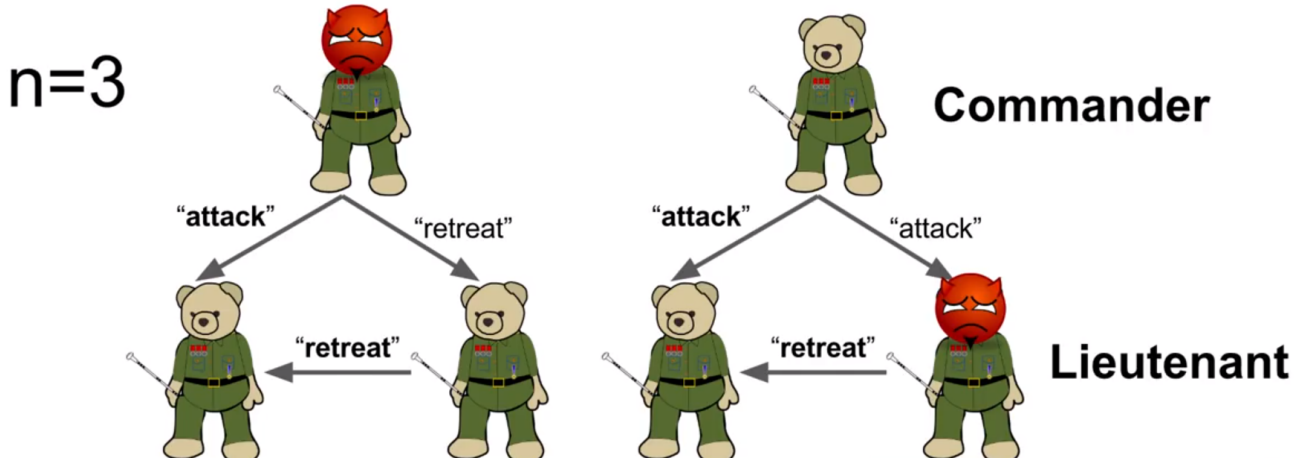
- ■ case1: commander said attack, left node asks right node if commander said attack, right node says attack and all three have reached consensus



■ case 2: if you are not the traitor, and your peer is the traitor and he flips the decision of the commander when he says it to you



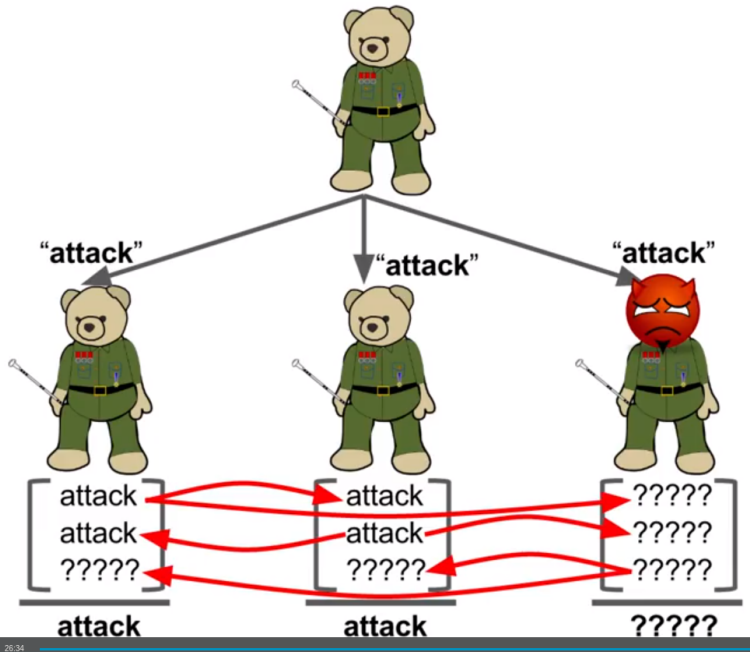
• you have to figure out whether actual decision is attack or retreat
  ◇ two cases
    ■ either commander is the traitor and gave conflicting information
    ■ or peer is traitor
    ■ in both cases im getting the same information from commander and peer so cant decide
    ■ ie no solution for 3 generals 1 traitor



  "No solution for 3m+1 generals with >m traitors"
• ie if greater than or equal to 1/3rd of generals are traitors you cannot reach correct consensus

if less than 1/3rd is there then what is the solution?
• one solution is oral messages for <= m traitors OM(m) for all <=m

How this works with m=1



OM(m), m>0:

C: Sends order.

L:
1. Records if received.
2. Use OM(m-1) to tell others.
3. Follows majority() order.

m	Messages Sent
0	O(n)
1	O(n^2)
2	O(n^3)
3	O(n^4)

ie for m traitors there is O(n^m) complexity which is absurdly long ie oral message will work but is not fast
• if you want to solve BGP
  ◇ dont solve on your own
  ◇ use tried and tested method
  ◇ keep n and m low

##
• for crypto, similarly it doesnt matter which specific generals or nodes are malicious it needs to arriving at specific correct decision and all nodes must agree on that decision.
• the communication is set of balances
• paxos
  ◇ communication channels not reliable and nodes going unreliable due to natural causes not mailcious
• there is a paper which has practical byzantine fault tolerance(pbft)
  ◇ solves both byzantine two generals and byzantine general's problem for any malicious or natural cause
• satoshi's protocol has consensus mechanism which he/-she/them claims is in accordance with limitation set by byzantine general's problem
  ◇ tolerant to attackers
  ◇ as well as unintentional faults from nodes

Proof of work

• in a peer-peer network which is (byzantine fault tolerant) bft compliant assumes decisions are not tampered if a “strict majority”(half) or “super majority”(two-thirds) are honest

• one attacker can influence the decision by creating sock nodes called sybils (one guy pretending to be many) to generate the majority he/she needs.
  ◇ this is called sybil attack(formalized name given by John in 2002)
  ◇ mitigation is proof of work(pow)

origins

• pow first proposal in 1992
  ◇ goal to detect spam
  ◇ spam, sybil attack, DoS are similar where enemy amplifies influence in network compared to regular users
  ◇ pow is defense in all the cases
  ◇ receiver would only open the mail if sender attached some proof that it did computational work, hence proof of work
  ◇ for a regular computers each proof would take few seconds(ie low computational cost), for one guy pretending to many to launch all mails it would take longer
  ◇ pow instance aka puzzle needs to be specific to the mail as well as the recipient, else for the pow computation cost of one message
    ■ spammer can send same message to lot of people
    ■ or different messages to same person
  ◇ regardless of how hard it is to generate the pow for sender, for verifying at the receiver it should pose low computation power
  ◇ has backdoor for central authorities to send messages w/o solving the puzzle
  ◇ researchers gave three such puzzles

hashcash

• made by Adam Back in 1997, part of cypherpunk community
  ◇ cypherpunks are activists who wanted to create social and political change through cryptography and opposed power of governments and centralized banks etc
• simpler than 1992's pow concept
  ◇ no central authority, no backdoor
  ◇ uses hash function instead of digital signature
  ◇ principle is hash function may be modelled as random function and only way to figure out the input for the particular output is to brute force inputs.
    ■ this can be extended into only way to find single input that hashes into random set of outputs is to bruteforce all sets of possible inputs
• hashcash is not cash as it doesnt prevent double spending
• Adam viewed bitcoin as straightforward extension of hashcash

Sybil-resistant networks

• John who coined Sybil attacks, said
  ◇ if a node is acting like N nodes then it would be unable to solve all N puzzles in time and the fake identities can be purged
  ◇ a malicious node however can have “numerical advantage” over innocent node, because of numerical superiority
  ◇ so regardless of intent, all nodes should can claim as many identities as possible
  ◇ BFT protocol changed from

“at most a fraction f of nodes are faulty”

to

“at most f is the total computational power controlled by faulty nodes”

• it became no longer important to validate identities
  ◇ ie you can be anonymous and run BFT protocol
• satoshi motivated all nodes to perform computationally expensive pow (earlier pow was not so expensive that it took a single computer minutes to solve)
  ◇ motivation was bitcoin

pow and digital cash: a catch 22

• anti spam measure did not succeed because spammers purchased better hardware to decrease time lag
• we created better crypto puzzles they bought better hardware
  ◇ this better hardware needs money to be bought
  ◇ token currency had limits and was strictly regulated at that time
  ◇ so spammers cannot buy hardware using token currency for prolonged periods of time
  ◇ this makes creating unregulated digital cash a problem
    ■ spammers can use digital cash to buy better hardware
    ■ anti spam people may/may not buy better pow solutions using digital money
      - either way they lose money/time in fighting spam
    ■ so for pow to work you should not create digital cash
      - this is the catch 22 , because for bitcoin to work, we need pow and digital cash to coexist
• one solution is to make the digital cash pow itself as

with hashcash, b-money and bit gold
  ◇ b-money and bit gold are timestamping services that sign on the creation of money through pow process
  ◇ but if disagreement occurs then majority wins
    ■ if normal nodes arent allowed to assume as many identities as possible, or a gate keeper is there to control entry into the network, it is vulnerable to Sybil attack

Putting it all together

• puzzle is not money, used for intergrity of blockchain
  ◇ puzzle solved by miners
  ◇ transactions checked by miners
  ◇ puzzle solution and transaction result send to blockchain for global verification, then if approved miner rewarded else no reward
• correctly solved means miner gets transaction fees+newly minted coins and adds the transactions to new block
  ◇ blocks are timestamped so no cheating
• puzzle has no economic value
  ◇ difficulty of solving depends on size of whole network
    ■ difficulty remains roughly the same regardless of miner's computing power
  ◇ number of bitcoins from a new block halves every four years
  ◇ monetary incentive is provided by pow to preserve the integrity of blockchain
• without all these features 51%(majority wins) percent attacks are very feasible

Public keys as identities

• transaction: string which is a statement encoding that A

wish to pay B X value signed by A
◇ this string is confirmed and finalized in a block by pow
◇ value is transferred from A's public anon id to B's public anon id
◇ you can create as many public id as you want (public-private key pair)
■ in dencental and anon way
■ ie this is a did (decentralized id)
■ but there is no way to route the message to the right computer if all you know is the unlinkable public address
- trade off made for security reasons

The blockchain

• blockchain has no standard technical definition but is a loose term for systems that somewhat is similar to bitcoin and its ledger

some examples

1) database for different banks to hold their transactions at a common place

• transactions are audited at end of day and accounts are settled by a main/central bank
◇ all parties involved are know to each other and trusted hence bitcoin pow not needed
◇ crypto currency not needed because account is holding INR or USD etc
◇ linked time stamping is useful to ensure global ordering of transactions
◇ state replication is useful for extra availability
■ less processing fees etc

2) asset management system

- tracks ownership of financial security, real estate etc
- requirement is : secure, global registry of documents, low barrier of entry and public participation.
- application:
  - ◇ electronic voting
  - ◇ timestamping services
  - ◇ business dealings with blockchain
    - ■ payment done only if asset is transferred
    - ■ done using smart contract
      - encodes
        - → complex business logic
        - → assets
        - → prices etc

smart contracts

- consensus protocol for correct execution of a publicly specified program
- end user can invoke these functions in their programs
  - ◇ miners will execute them and blockchain arrives at common consensus

◇ end user can trust this output
■ ie we can have a script that tells to move money from account to another and this is executed by miners and validated by the blockchain
• smart contracts can be used to
◇ handle, own and transfer money -> decentralized version of gofundme or patreon campaigns

permissioned blockchains

• blockchains which are
◇ selective in whom gets to participate
■ write transactions, mine block etc
■ so PoW is dropped and we can use simple BFT
◇ could use hash trees instead of merkle binary trees to simplify stuff
• should be concerned (these things dont matter in a public blockchain as nothing overtly secret will be put there anyway)
■ about confidentiality of data
■ identity and public key infra
■ access control

Much needed skepticism

• many banking related blockchains dont use nakomoto

consensus (pow)
  ◇ use only ledger data structure and byzantine agreement
    ■ relatively very old and not so called “revolutionary”
• blockchain is presented as more secure that traditional centralized registers
  ◇ not always true
  ◇ this claim is said wrt to decentralizing aspect
    ■ therefore endpoint security is now at hands of end users so they have more control and hence the system is more secure if users are careful
  ◇ but unlike traditional systems, blockchain transactions are
    ■ near instant
    ■ irreversible
    ■ in public blockchains and anonymous by design
    ■ if private key is compromised, everything is lost
  ◇ at least 6% of bitcoins are stolen at least once
  ◇ due to carelessness of bitcoin exchangers like Mt Gox, bitcoins are also destroyed

bitcoin-wallet-blogpost

crypto wallet

types of wallets

it is one of the following general types
• device
• physical medium
• program
• service

detailed classification

1) connectivity to internet

1- hot wallet
1> wallet connected to internet
2> not recommended because crypto and or private keys may be stolen by exploiting web based vulnerabilties

2- cold wallet
1> connected to internet/blockchain only when transaction is done
2> safer
3> hardware wallet
1. stores private key in a safe way like that of a protected micro controller
1- ex:
1> ledger nano
2> trezor
4> paper wallet
1. used in initial days of bitcoin
2. private key encoded in a piece of paper
3. not inuse anymore because of how stupidly easy to lose

2) functionality
  1- full node
    1> aka minning node
    2> stores the entire blockchain ledger on the system
    3> only use this if you intend to mine crypto
    4> helps in validating transactions and blocks and relay them to other nodes in network
    5> requires absurd levels of hardware and computation power to be a feasible long term full node
    6> ex:
      1. bitcoin core wallet

  2- simple node
    1> does not mine and does not store any data from blockchain
    2> only does transaction , no validation
    3> ex:
      1. electrum -> desktop wallet made on electron platform for bitcoin'
      2. metmask -> multi platform wallet for ethereum
3) storage
  1- desktop wallet
    5> most common type of wallet
    6> either full node or simple node
    7> EX:
      3. bitcoin core -> full node
      4. electrum -> simple node
      5. greenaddress -> simple node
  2- web wallet
    8> web based simple node
    9> runs from either remote URL or browser extension
    10> EX:
      6. bitgo -> bitcoin

7. metamask -> ethereum
8. spectro -> bitcoin, ether, NEM, dash

3- mobile wallet

11> simple node for phones

12> may have desktop or web wallet counterparts

13> EX:

9. electrum - bitcoin , desktop wallet is there
10. metamask - ethereum ,web wallet is there
11. samourai - bitcoin
12. spectrocoin -> bitcoin, ether, NEM, dash, web wallet is there

what can it do

• track ownership of assets and transactions
• receive and send crypto within a distributed network
• similiar to how gpay, paytm works but not dealing with fiat currencies but with virtual currency like
  ◇ bitcoin
  ◇ litecoin
  ◇ ether
• wallets can u give connection to “test nets”
  ◇ for development and research purposes
  ◇ can perform dummy transactions that is free of cost

binance-smart-chain

binance smart chain:
it is a clone of ethereum chain sorta
so
lang: solidity

tools:
• remix
• metamask, binance wallet

seminar-topic-proof-of-authority

Presentation Contents:
☐ Define the Consensus Algorithm
☐ Need/History of the Consensus Algorithm
☐ Working of Consensus Algorithm
☐ Adv/Dis. Adv
☐ Examples and Real-Time Use cases

Outline:
• proof of * algorithms
• proof of authority
  ◇ why PoA
  ◇ why in ethereum
• Types of PoA
  ◇ Authority Round (AuRa)
  ◇ Clique

Intro to consensus algorithms
• a method of creating a single source of truth/arriving at agreement
  ◇ ex: everyone knows that I have X ETH

- Lots of algorithms exist for establishing this
- Challenges
  - ◇ bad actors
  - ◇ network delays
    - ■ distributed networks introduce delay
- Proof of work
  - ◇ computers racing with each other to solve a simple math problem
  - ◇ problem not easy to solve
    - ■ finding what nonce gives this particular type of hash
  - ◇ incentive: cost of computational effort put in >> profit of forged reward
  - ◇ winner proposes next block and gets block reward
  - ◇ problem
    - ■ computationally expensive
      - - GPU, ASICS
      - - electricity, water and rent costs
      - - environmental effects
    - ■ cannot produce blocks at a reliable fixed interval
      - - depends on difficulty level, hash power
    - ■ only used to solve crypto puzzle
- Proof of stake
  - ◇ deposit some funds and depending on amount of money you get more voting power
  - ◇ if you lie/misbehave your fund (stake) gets taken away
  - ◇ incentive: stake >> profit of forged reward
  - ◇ better than PoW
    - ■ not much computationally expensive
    - ■ no need to pollute environment
  - ◇ where PoS can improve
    - ■ assumes that bigger the numerical value of stake, more concerned you are about integrity of network

    ■ But Regardless of actual value invested, a person who as invested 20% of his/her stake will be more concerned than the person who has invested 1%

Proof of authority

• how PoA is better
  ◇ removes monetary incentives all together
  ◇ instead of staking fund, validators stake identity and credibilty
  ◇ though not suited for public blockchains, can have high throughput in permissioned blockchains with less no of validators
• trusted set of nodes as validators
  ◇ no bad actors
    ■ all relevant actors are trusted
• produce blocks at reliable fixed interval
  ◇ deterministic calculation of when next block available created
  ◇ know the state of authorities
• higher performance
  ◇ depending on network conditions

PoA in ethereum

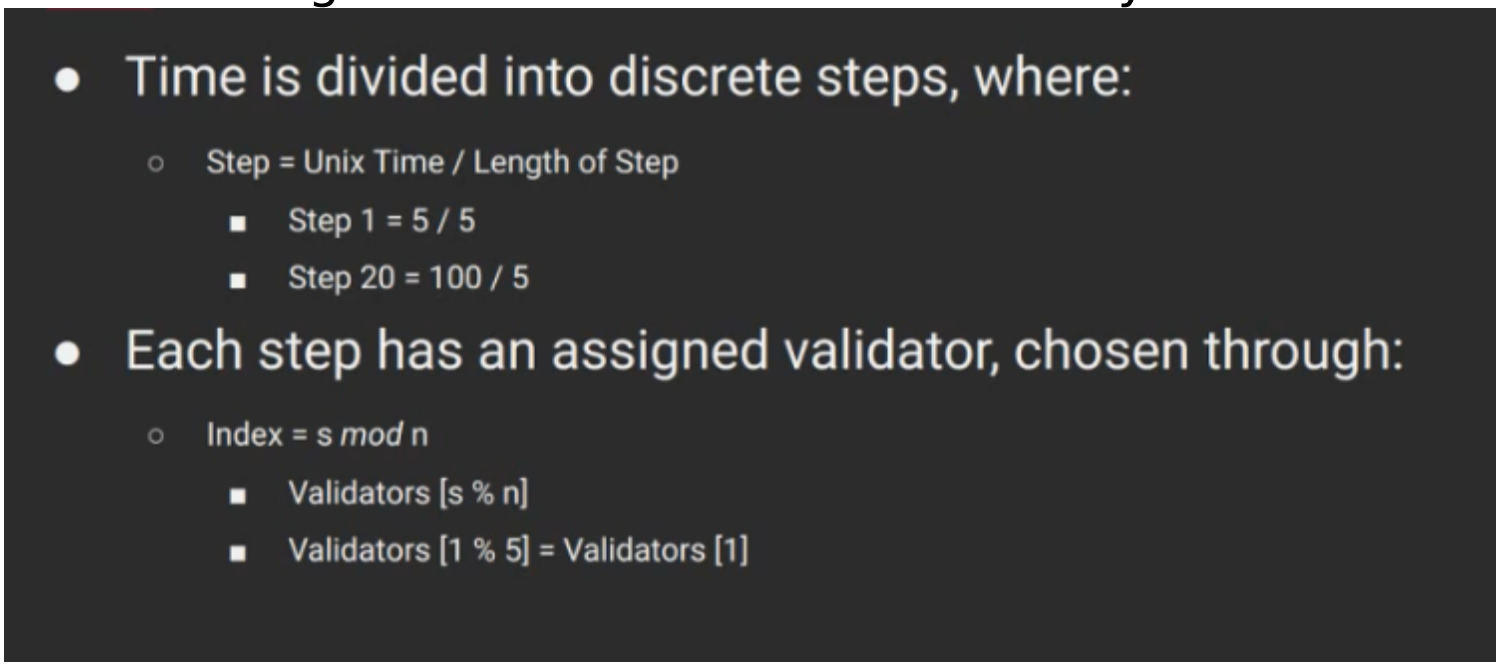
• ropsten is a PoW testnet
• no incentive to mine for crypto in a testnet
  ◇ ie low hash rate
  ◇ lead to majority attack in 2017
    ■ raised the block gas price
    ■ spam the network with transaction
    ■ made the network unusable
• introduced two more testnets incase attack is repeated
  ◇ Kovan

- ■ Authority Round
  - developed by parity
- ◇ Rinkeby
  - ■ Clique
    - running clique engine developed by Geth

Authority round (Aura)

Block Proposal

• round robin based algorithm
• editable via smart contract
• time is divided into discrete slots
• each validator is given a slot to propose block
• • assuming validators are listed in an array



finality in Aura

• block is finalized when more than half of validators have confirmed blocks on top of it
• bitcoin and ethereum have probabilistic finality
• higher the amount, higher the blockchain confirmation required
  ◇ busier the network longer time for transaction confirmation

- to avoid dropped transaction due to bitcoin softforks
  - ◇ wait for 6 blocks minimum(1 hour)
- similiar condtion of ETH
  - ◇ 1 minute for roughly one confirmation
  - ◇ need 50 confirmation for major exchanges

validator set

- we can choose certain nodes to be validators
  - ◇ chain specification (chain.json)
    - ■ hard code them into it and is there on every client
    - ■ if we want to change then hard fork the chain
  - ◇ smart contract
    - ■ decide easily when some node can be validator
    - ■ one of the original authorities also knows the validator smart contract

from 0-10 block have the validator of this static list
from 10-20 have another validator
from 20 onwards let the smart contract choose

- epoch
  - ◇ time where we have one set of validator
    - ■ ie we have 3 epochs

Clique

- defined in EIP-225
- goal: standardize PoA for ethereum clients
- used in Rinkeby and Goerli testnets

block proposals

- validators are either INTURN or NOTURN
- INTURN → more priority or advantage or weight in proposing a block
- NOTURN → less weight but still can propose a block
- if you are INTURN propose block ASAP
  - ◇ if you are NOTURN wait a random amount for INTURN validator to propose a block
- after signing a block, validators are not allowed to sign until the next

  floor(SIGNER_COUNT/2) + 1 block

choosing the longer chain

- resolving soft forks
  - ◇ prefer chains having the most difficulty
  - ◇ Difficulty of INTURN = 2
  - ◇ Diffculty of NOTURN = 1
- ie gives preference to chains built with INTURN validators

validator set changes

- instead of smart contract, as a validator you can vote to change the validator set at every block
- one vote per block proposal, majority of validators to vote on a certain action for to be enacted
  - ◇ ie majority consensues

Examples/Real life use cases:
1) in testnets to deter DoS attacks
2) In hyperleger Besu (Besu is a Java-based Ethereum client that implements the Enterprise Ethereum Alliance (EEA) specification and can be run on the public network or on private networks, as well as on a number of testnets.)

iov-proj

Lit survey

1) Blockchain for the Internet of Vehicles

Word mapping:
AI → Artificial Intelligence
CAN → Controller Area Network
DB → database
DDTN → Delay and Disruption Tolerant Network [custom]
DID → Distributed Identifier
DoS → Denial of Service
DSRC → Dedicated Short-Range Communication
DTC → Diagnostic Trouble Code
ECU → Electronic Control Unit (inside vehicles)
IoT → Internet of Things
IoV → Internet of Vehicles, subclass of IoT
IPFS → InterPlanetary File System

LAN → Local Area Network
LTE → Long Term Evolution
malnode → malicious node in the network[custom]
MITM → Man/Woman/Group In The Middle
mobapps → applications installed in apple or android mobile devices
OBD → On Board Diagnostics
OEM → Original Equipment Manufacturer
OS → Operating Systems
OTA → Over The Air
p2p → peer to peer
PVSS → Publicly Verifiable Secret Sharing
QBA → Quantum Byzantine Fault Tolerance
RA → Register Authority
RBFT → Redundant Byzantine Fault Tolerance
regnum → vehicle Registration Number [custom]
RTBC → RealTime Blockchain
things → according to context may refer to “vehicle, smart object near road, people, internet" as whole [custom]
V2V → Vehicle To Vehicle
V2X → Vehicle to things communication
VANET → Vehicular Adhoc NETwork
VSP → Vehicle Service Provider [custom]
WAN → Wide Area Network
WiMaX → World-wide Interoperability for Microwave Access
ι → Identity chain
τ → Transaction chain

0) Abstract

• advances in IoT, internet => traditional VANET to change to IoV

◇ IoV vehicles uses V2X tech to make decision and alert user if need be

• security, privacy at risk due to connected nature

◇ challenge of IoV

■ high connectivity among and between vehicles leads to sensitive information leak with unknown/-untrusted vehicles

■ malicious node compromising nodes by falsifying information for making critical decisions

■ remote control of vehicle via the internet

• problem: ensure authentication, provide secure communication between nodes

◇ solution: use RTBC for IoV

■ keep track of communication => accountability

■ adds features

- automatic toll payment
- vehicle service slot booking + payment
- fuel payment
- vehicle insurance renewal

■ native cryptocurrency for intranetwork payment

1) Introduction

• current problem

◇ vehicles → computers on wheels communicating via IoT

◇ VANET

■ every vehicle made to a wireless node to talk to other such nodes creating a short term WAN

■ cannot provide global and sustainable services to end users

- problem because

→ “things” are temporary, random, unstable and

range of usage is localized and "on/off"

1.1) Internet of Vehicles

• IoV defined as
  ◇ a large scale distributed wireless communication and information exchange for V2X
  ◇ for the purpose of
    ■ tight integration of human-vehicle-thing-environment
    ■ reduce cost
    ■ improve efficiency of transport
    ■ increase service level of city
  ◇ using
    ■ vehicle's intelligence
      - combine driver and network to function as one being to make the entire environment intelligent
    ■ vehicle's networking
      - VANET
      - vehicle telematics
      - mobile internet
  ◇ according to agreed communication protocols and data interaction standards
    ■ IEEE 802.11p standard
    ■ cellular tech

• Features
  ◇ uses IoT for intelligent transport system
  ◇ open and integrated network
  ◇ with high manageability, controllability, operationalization, credibility
  ◇ composed of multiple users.vehicles, things and networks
• integration of three networks

◇ inter-vehicle
◇ intra-vehicle
◇ vehicular mobile network (dynamic inter-vehicle ↔ things with DDTN)

1.2) Blockchain

• defined as
◇ decentralized computation and distributed information sharing platform
◇ enables multiple authoritative domains who do not trust each other
◇ to cooperate, coordinate and collaborate
◇ in a rational decision making process
• features
◇ immutability
◇ consensus
◇ smart contracts
◇ shared information ledger
◇ decentralization
◇ hence best tech for IoV to answer challenges and problems

2) Security and privacy issues

2.1) Connected Vehicle Security

a) CAN bus
• protocol allowing direct ECU-ECU communication
• features
◇ data integrity
◇ data consistency
◇ error detection
• problem

  ◇ no network isolation
  ◇ no encryption
  ◇ no authentication and access control
• causing
  ◇ message spoofing
  ◇ DoS

b) OBD interface
• allows to access status of subsystems of vehicle
• entry point for the in-vehicle network
• problem
  ◇ no authentication
  ◇ no features to detect malicious code

c) infotainment system
• combination of hardware and software which provides entertainment/information to vehicle occupants
• has added smart phone to list of stuff to connect with
  ◇ usually sensitive vehicle information is sent to user's smartphone

d)OTA
• update pushed from OEM to vehicle wirelessly
• problem
  ◇ no software authentication by OEM

2.2) Intelligent Device Security

• OS of vehicle network and mobile device getting connected
• problem
  ◇ resulting vehicular computational intelligence causes
    ■ safety issues in system
    ■ via vulnerabilities in OS and mobapps

2.3) V2X communication security

- many ways to communicate
  - ◇ DSRC
  - ◇ LTE
  - ◇ WiMax
  - ◇ Bluetooth
  - ◇ ZigBee (IEEE 803.15.4)
- equals many ways of risk of compromise of short term communications
  - ◇ malicious nodes can do
    - ■ protocol cracking
    - ■ MITM

2.4) Data security

- large volumes and variety of incoming traffic means challenge in processing data efficiently and safely
- exposing PII is a risk
- harm road infrastructure and road users

3) Related Work

- three level “client-connection-cloud” model using blockchain
  - ◇ for secure data communication between nodes
  - ◇ addresses security and privacy issues in IoV
- authenticity and accountability provided by RA
  - ◇ maintains record of every vehicle prevents entry of malnode
- confidentiality
  - ◇ uses asymmetric encryption in communication

- CUBE Auto Blockchain

◇ multilayer protection method for a secure automotive ecosystem
◇ using
■ blockchain
- solves traditional blockchain issues by
→ peer-peer hypermedia protocol
→ asymmetric encryption
■ endpoint protection
- generates endpoints at all external connection points
- does hash analysis using know bad 300 million attacks with 10MB storage requriement
■ cloudbased intelligence with deep learning
- unknown malware is send to transnational transcorporational DB
- unknown attacks sent to sandbox for quarantine analysis
→ result uploaded in DB shared to nodes
→ native tokens used
⇒ nodes get token when sharing user data
⇒ nodes pay tokens if data is used for decision making, vehicle service etc
■ quantum hash encryption
- securing connected car environment

3.1) Shortcomings of existing solutions

• no privacy
◇ data stored in public blockchain
• use of heavy encryption
◇ introduces latency in network
◇ against principles of DDTN
◇ no explanation of how vehicle communicate securely with non-vehicles

• no accountability for nodes

4) Proposed Work

• VAAHAN-Namchain
  ◇ RTBC with AI
  ◇ end-end solution
  ◇ for
    ■ IoV ecosystem's security, safety and privacy
  ◇ why real time
    ■ safety and life critical in automotive environment

4.1) Blockchain architecture

• Application layer
  ◇ bridge between user and blockchain
  ◇ application is web or mobile based
    ■ interact with smart contracts, tokens , consensus
  ◇ uses
    ■ supports consensus algorithms
      - RBFT
      - QBA
    ■ has
      - PVSS which allows anyone including participants to verify anyone's share
• Storage layer
  ◇ BigchainDB
    ■ blockchain with traditional DB properties
    ■ use MongoDB for storage
    ■ tendermint protocols for inter-node communication
    ■ stores vehicle details and v2v communication transactions
  ◇ IPFS
    ■ p2p, content addressing method for storing files

    - Vehicle log files are stored in IPFS
    - hash of content address is in BigchainDB
- Network layer
  ◇ uses Libp2p for p2p communication between vehicles in blockchain
  ◇ for IoT
  ◇ transport protocol agnostic
  ◇ uses multiaddress - a self describing addressing format

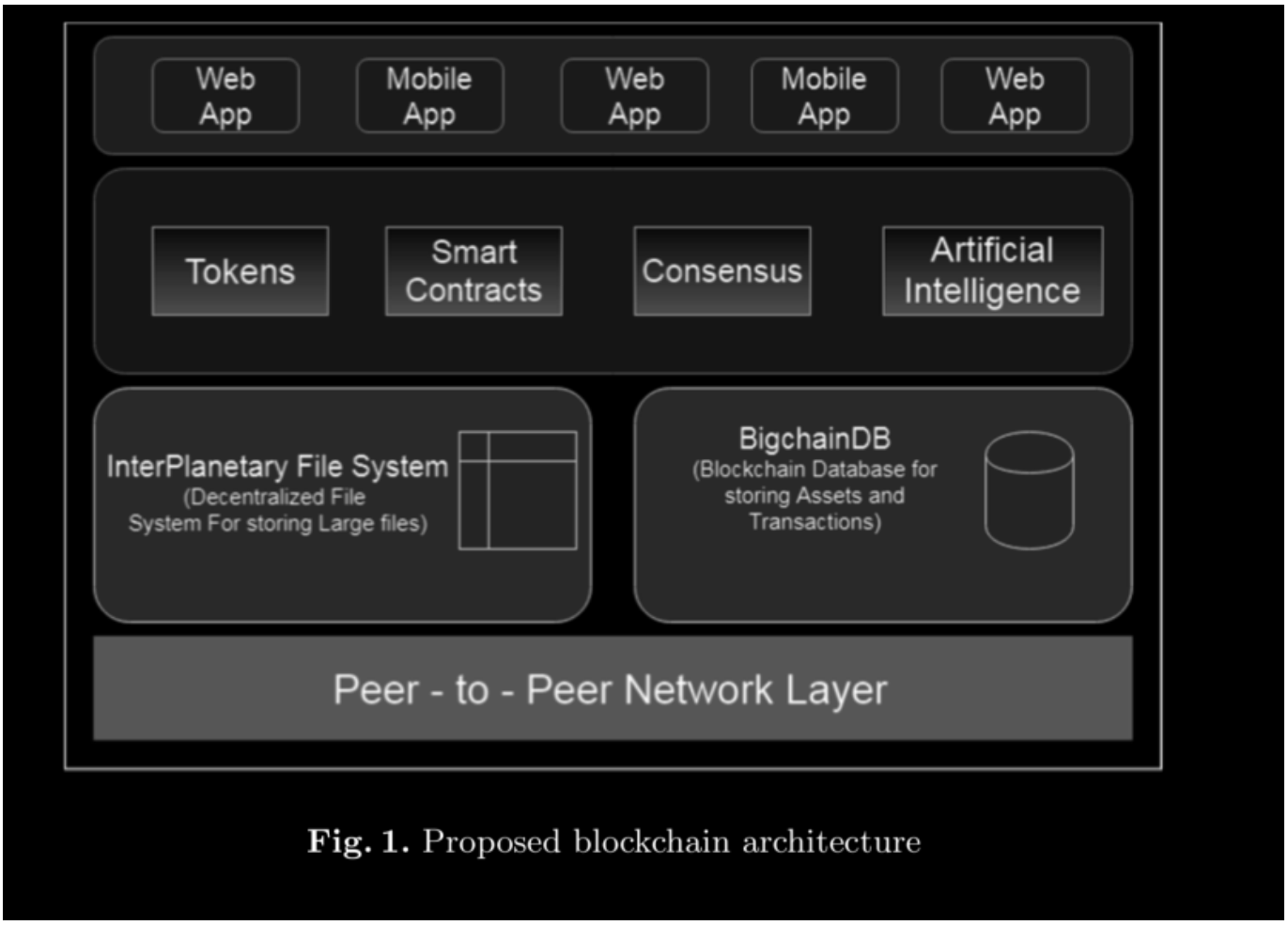


Fig. 1. Proposed blockchain architecture

4.2 Actors and components of blockchain

- Actors
  ◇ RA
    - central government transport authority

    ■ verifies new vehicles and gives regnum
    ■ RA relevant only during initial phase of registration
  ◇ vehicle owner
    ■ any person with valid ID proof of residence and owns a vehicle
    ■ vehicles are registered against owner
  ◇ VSP
    ■ provide services like gas pumps, vehicle insurance and other automotive services
• Component
  ◇ ι
    ■ identity chain which stores the transaction record about registered vehicles
  ◇ τ
    ■ transaction chain stores communication between various actors of the system
  [both chains
    ■ transaction stored in JSON
    ■ uses SHA3-256 in multihash format]
    ■ nothing related to registration transaction except DID is stored here
  ◇ AI
    ■ to learn and predict passenger, driver and vehicle behavior
    ■ can help IoV to encourage good driving and vehicle maintenance
    ■ help detect malicious activity and trigger actions using smart contracts
  ◇ smart contracts
    ■ governement and OEM to deploy
      - insurance renewal, fuel refill payment, service payment etc

DID

• prove identity of node that is publicly verifiable and accessible
• makes user pseudo-anonymous
• DID is from ι
  ◇ no other info is accessed

4.3 Workflow

Registration Workflow

1. The user submits the unique identification details (Aadhar Number) to the RTO via showroom coordinator and also present in person.
2. The vehicle showroom coordinator submits the vehicle along with the vehicle details like Engine number, Chassis number, Model number, Insurance details, etc. and the user information like Name, Address, Contact details, Nominee. This constitutes **a transaction** in the network.
3. The Registration Authority verifies the information provided and approves the transaction. This creates a block in the ι which needs to be validated.
4. The transaction is validated by the user, vehicle showroom coordinator, Nominee of the user, Transport Authority belonging to the different region, coordinator from vehicle manufacturer.
5. Once the transaction is validated, it is then added to the ι and a Distributed Identifier (DID) is given to the vehicle showroom coordinator.
6. Upon the receipt of this DID, the vehicle showroom coordinator, configures the vehicle with this DID.
7. This DID would be then used by the vehicle for communicating to the vehicles, roadside infrastructure, internet and any other IoT components part of the ecosystem.

Transaction Workflow

OBD detects something is wrong and a corresponding DTC is generated

1. Vehicle sets the DTC inside the vehicle and sends the DTC code with the vehicle's DID, state information as a transaction to the τ.

2. The diagnostic service smart contract running would be invoked based on this transaction. Based on the critical nature, two possible actions could be taken:
   - smart contract would ignore this transaction, if the DTC is non-critical.
   - or smart contract would add this transaction as a valid block in τ and alert the vehicle service center.
3. On receiving this, now the vehicle's service center technician can also perform two actions
   - perform a remote diagnostics to rectify the identified problem.
   - or alert the vehicle owner for immediate service, which is also logged as a transaction in τ
4. If the vehicle owner ignores this notification and if the vehicle or vehicle owner is affected because of the identified issue. Then Insurance claim for vehicle parts (incase of vehicle part failure) or free replacement of parts or insurance claim for loss of life shall not be provided, since he/she did not turn up to service center as alerted by service center technician.

4.4) Cryptocurrency

- native crypto cqlled “Naanayam” (coin in tamil)
  - ◇ similar to ERC-731 tokens(this is NFT)
    - ■ cryptokitties are represented in ERC731
- use cases
  - ◇ reward for good driving practices
    - ■ AI used for learning, predicting user behavior
    - ■ calculate driver score based on some metrics
    - ■ reward of Naanayam tokens done via reward smart contract
      - for example rewarding good driving behavior
  - ◇ vehicle service booking and payment
    - ■automatic service scheduling by smart vehicles
    - ■ slot booking via smart contract with user consent
    - ■ once service is done smart contract automatically deduct corresponding charge from user wallet

◇ automatic toll payment

Table 1. Comparison with other blockchains

Features	BigchainDB	Indy [24]	CUBE	VAAHAN
Native tokens			✓	✓
Support for native tokens	✓		✓	✓
ZKP		✓		✓
Identity management	✓	✓		✓
Database query support	✓			✓
BFT	✓	✓		✓
File storage				✓
Smart contracts		✓		✓

presentations

paper-understanding-blockchain-for-the-internet-of-vehicles

Understanding of “Blockchain for the Internet

# of Vehicles” conf. paper

Word mapping:

AI → Artificial Intelligence
CAN → Controller Area Network
CID → Content Identifier
DB → database
DDTN → Delay and Disruption Tolerant Network [custom]
DID → Decentralized Identifier/Identity
DoS → Denial of Service
DSRC → Dedicated Short-Range Communication
DTC → Diagnostic Trouble Code
ECU → Electronic Control Unit (inside vehicles)
IoT → Internet of Things
IoV → Internet of Vehicles, subclass of IoT
IPFS → InterPlanetary File System
LAN → Local Area Network
LTE → Long Term Evolution
malnode → malicious node in the network[custom]
MITM → Man/Woman/Group In The Middle
mobapps → applications installed in apple or android mobile devices
OBD → On Board Diagnostics
OEM → Original Equipment Manufacturer
OS → Operating Systems
OTA → Over The Air
p2p → peer to peer
PVSS → Publicly Verifiable Secret Sharing
QBA → Quantum Byzantine Fault Tolerance
RA → Register Authority
RBFT → Redundant Byzantine Fault Tolerance
regnum → vehicle Registration Number [custom]

RTBC → RealTime Blockchain
things → according to context may refer to “vehicle, smart object near road, people, internet" as whole [custom]
V2V → Vehicle To Vehicle
V2X → Vehicle to things communication
VANET → Vehicular Adhoc NETwork
VSP → Vehicle Service Provider [custom]
WAN → Wide Area Network
WiMaX → World-wide Interoperability for Microwave Access
ι → Identity chain
τ → Transaction chain

Current Scenario

- Vehicle communication moving from VANET to IoV
- VANET
  - ◇ every vehicle made into to a wireless node to talk to other such nodes creating a short term WAN
- what is the problem with VANET?
  - ◇ cannot provide global and sustainable services to end users
    - ■ why because
      - → entities are temporary, random, unstable and range of usage is localized
- why IoV
  - ◇ Features
    - ■ subclass of IoT
    - ■ open and integrated network
    - ■ composed of multiple users,vehicles, things and

networks
  ◇ integration of three networks
    ■ inter-vehicle
    ■ intra-vehicle
    ■ vehicular mobile network (dynamic inter-vehicle ↔ things with DDTN)

IoV defined as
  ◇ a large scale distributed wireless communication and information exchange for V2X
  ◇ for the purpose of
    ■ tight integration of human-vehicle-thing-environment
    ■ reduce cost
    ■ improve efficiency of transport
    ■ increase service level of city
  ◇ using
    ■ vehicle's intelligence
      - combine driver and network to function as one being to make the entire environment intelligent
    ■ vehicle's networking
      - VANET
      - vehicle telematics
      - mobile internet
  ◇ according to agreed communication protocols and data interaction standards
    ■ IEEE 802.11p standard
    ■ cellular tech

Problem with IoV

Connected Vehicle Security

a) CAN bus

- protocol allowing direct ECU-ECU communication
- features
  - ◇ data integrity
  - ◇ data consistency
  - ◇ error detection
- problem
  - ◇ no network isolation
  - ◇ no encryption
  - ◇ no authentication and access control
- causing
  - ◇ message spoofing
  - ◇ DoS

b) OBD interface
- allows to access status of subsystems of vehicle
- entry point for the in-vehicle network
- problem
  - ◇ no authentication
  - ◇ no features to detect malicious code

c) infotainment system
- combination of hardware and software which provides entertainment/information to vehicle occupants
- has added smart phone to list of stuff to connect with
  - ◇ usually sensitive vehicle information is sent to user's smartphone

d)OTA
- update pushed from OEM to vehicle wirelessly
- problem
  - ◇ no software authentication by OEM

Intelligent Device Security

- OS of vehicle network and mobile device interacting
- problem
  - ◇ resulting vehicular computational intelligence causes
    - ■ safety issues in system
    - ■ via vulnerabilities in OS and mobapps

V2X communication security

- many ways to communicate
  - ◇ DSRC
  - ◇ LTE
  - ◇ WiMax
  - ◇ Bluetooth
  - ◇ ZigBee (IEEE 803.15.4)
- equals many ways of risk of compromise of short term communications
  - ◇ malicious nodes can do
    - ■ protocol cracking
    - ■ MITM

Data security

- large volumes and variety of incoming traffic means challenge in processing data efficiently and safely
- exposing PII is a risk

Existing Solution

- three level “client-connection-cloud” model using blockchain
  - ◇ for secure data communication between nodes
  - ◇ addresses security and privacy issues in IoV
- authenticity and accountability provided by RA

◇ maintains record of every vehicle prevents entry of malnode

• confidentiality

◇ uses asymmetric encryption in communication

• CUBE Auto Blockchain

◇ multi layer protection method for a secure automotive ecosystem

◇

■ blockchain

- solves *conventional blockchain* issues by

→ peer-peer hypermedia protocol

→ asymmetric encryption

■ endpoint protection

- generates endpoints at all external connection points

- does hash analysis using know bad 300 million attacks with 10MB storage requirement

■ cloud based intelligence with deep learning

- unknown malware is send to transnational trans-corporational DB

- unknown attacks sent to sandbox for quarantine analysis

→ result uploaded in DB shared to nodes

→ native tokens used

⇒ nodes get token when sharing user data

⇒ nodes pay tokens if data is used for decision making, vehicle service etc

■ quantum hash encryption

- securing connected car environment

Problem With Existing Solution

• no privacy

◇ data stored in public blockchain
• use of heavy encryption
◇ introduces latency in network
◇ against principles of DDTN
◇ no explanation of how vehicle communicate securely with non-vehicles
• no accountability for nodes

Proposed Work

• VAAHAN-Namchain
◇ RTBC with AI
■ why real time?
- safety and life critical in automotive environment
■ why AI?
- helps with user/vehicle prediction, attack mitigation
◇ end-end solution
■ for IoV ecosystem's security, safety and privacy

Architecture of proposed work

• **Application layer**
◇ bridge between user and blockchain
◇ application is web or mobile based
■ interact with smart contracts, tokens , consensus
◇ it has
■ consensus algorithms
- RBFT
- QBA
■ secret sharing
- PVSS which allows anyone including participants to verify anyone's share
• **Storage layer**

◇ BigchainDB
■ use MongoDB for storage
■ tendermint protocols for inter-node communication
■ stores vehicle details and v2v communication transactions
◇ IPFS
■ p2p, content addressing method for storing files
■ Vehicle log files are stored in IPFS
■ hash of content address is in BigchainDB

• **Network layer**
◇ uses Libp2p for p2p communication between vehicles in blockchain
◇ for IoT
◇ transport protocol agnostic
◇ uses multiaddress - a self describing addressing format

Actors and components of blockchain

• **Actors**
◇ RA
■ central government transport authority
■ verifies new vehicles and gives regnum
■ RA relevant only during initial phase of registration
◇ vehicle owner
■ any person with valid ID proof of residence and owns a vehicle
- what is DID?
→ prove identity of node that is publicly verifiable

and accessible
→ makes user pseudo-anonymous
→ DID is from ι
⇒ no other information from ι is accessed by VSPs
■ vehicles are registered against owner
◇ VSP
■ provide services like gas pumps, vehicle insurance and other automotive services
• **Component**
◇ ι
■ identity chain which stores the transaction record about registered vehicles
◇ τ
■ transaction chain stores only communication between various actors of the system

◇ In both chains
■ transaction stored in JSON
■ uses SHA3-256 in multihash format

◇ AI
■ to learn and predict passenger, driver and vehicle behavior
■ can help IoV to encourage good driving and vehicle maintenance
■ help detect malicious activity and trigger actions using smart contracts
◇ smart contracts
■ government, VSP and OEM to deploy
- insurance renewal, fuel refill payment, service payment etc

Workflow

Registration Workflow

Transaction Workflow

OBD detects something is wrong and a corresponding DTC is generated

Cryptocurrency

- native cryptocurrency called “Naanayam”
  - ◇ similar to ERC-721 tokens(this is NFT)
    - ■ cryptokitties are represented in ERC-721
- use cases
  - ◇ reward for good driving practices
    - ■ AI used for learning, predicting user behavior
    - ■ calculate driver score based on some metrics
    - ■ reward of Naanayam tokens done via reward smart contract
      - for example rewarding good driving behavior
  - ◇ vehicle service booking and payment
    - ■automatic service scheduling by smart vehicles
    - ■ slot booking via smart contract with user consent
    - ■ once service is done smart contract automatically deduct corresponding charge from user wallet

◇ automatic toll payment

Proposed solution OK?

- **Authenticity**
  ◇ Uses RA to ensure authenticity of vehicle and user
- **Accountability**
  ◇ smart contracts employed for services using native cryptocurrency
  ◇ Transactions recorded in blockchain
- **Privacy**
  ◇ Uses DID to provide pseudo-anonymity
  ◇ Only authorized people can access PII from ι
    ■ Compartmentalization of data
      - only service relevant details accessible to VSP via τ

- **System security**
  ◇ Threats identified and triggers relevant smart contracts via AI
  ◇ Has BFT measures so resiliant against fault, compromised decision maker,DoS etc
  ◇ OTA may be send via consensus using tendermint protocol?[inference]
- **Availability and usability of data/services**
  ◇ data
    ■ relevant data like vehicle logs stored in IPFS, with corresponding ID in separate DB to ensure availability of data
    ■ faster consensus and block creation using tendermint

◇ services
■ monetized via native crytpocurrency
■ libp2p is used to ensure network communication is modular, future-proof
■ IoV technology used for scalable networks
■ User incentivized to keep road/vehicle/people/other-assets safe via reward schemes built with smart contracts

scenario-applicablity

Two groups:
group 1
1) normal vehicle with driver
2) v2v communication ie smart vehicle
3) autonomous driving and IoV

group 2
a)life cycle (production to auto junkyard)
b) only on road (safety of other vehicles and assets on the road)

Task:
what technology and what is applicable for each scenario based on googling and reading papers...

Results:
1a)
• store relevant car related information locally, share information about the car by external parties and distribute it in secure and privacy preserving way
◇ **hyperledger fabric** → doesnt require mining nodes

◇ decentralized app **Dapp** to interface with car and blockchain platform

◇ **smart contract called repository** → holds public key, identity of car ,address for the smart contract for other relevant entities within network

• Anti theft

◇ Ethereum Smart contract using **Remix-Ethereum IDE** , contract coded in **Solidity** language

• Supply chain management

◇ Blockchain collects data from assembly line machines, sensors (IOT) ensures integrity of information and used in tracing parts in a car

◇ **MQTT** used for IOT communication

◇ car parts identified by RFID

1b)

• not applicable as car wont interact with other vehicles or assets.

2a)

• vehicular forensics framework for hit and run cases

◇ Use a permissioned blockchain to manage collected vehicle related data for example as obtained from **Event Data Recorder** (EDR)

■ if it has **Road Side Units** (RSU) and other non vehicle things it is in 3a

◇ Integrate **Vehicular Public Key Infrastructure** (VPKI) for adding new nodes and keeping privacy

◇ Design of fragmented ledger to store to store vehicle related data, diagnosis , maintenance report etc

2b)

• blockchain based communication scheme
  ◇ A methodology to secure inter communication by
    ■ **ring signature based scheme** for identity verification
    ■ Information shared among vehicles verified using **multi party smart contracts**
    ■ Network is **3GPP Rel.14 (LTE V2X)**
    ■ considers inside and outside threats
      - unauthorized participation
      - replay attacks
      - **sybill attacks**
• **Regional Blockchain** to prevent 51% attack for v2v networks
  ◇ Deriving a condition for blockchain using PoW which gives low probability of 51% attack
    ■ number of good and bad nodes
    ■ message delivery time
    ■ puzzle computation time

3a)
• securing OTA update process
  ◇ let OEMs participate in making FOTA authentic and code untampered using **consortium blockchain**
  ◇ **credit reputation** for each vendor as incentive for making correct decision
  ◇ **Zero-knowledge proof protocol** used to exchange update with **proof of distribution** from AV
  ◇ **Attribute based encryption** to let only authorized vehicle only get the info

3b)
• platooning

◇ **path matching** scheme
◇ **Platoon Head**(PH) leading **Platoon Members**(PM)
■ **Reputation value based PH rotational selection**
■ **Smart contract enabled payment** for motivating AV to become PH and for deterring malicious activity

work:
* subject to implemenation
1 paper me first author
2 paper me second author(2 target)

target areas:
1) OTA
1- multiple OEM
2- for one download only one OEM
2) car's critical decision is recorded in blockchain
1- location of blockchain(applicable to all use cases)
1> inside or outside or both
3) v2x communication (i,p,v)
4) SCM , on road (fixed)

paper-understanding-integration-of-blockchain-and-IoV-in-car-supplychain

acronyms:

poc: proof of concept
dlt : distributed ledger technology
iot: internet of things
isa: information system architecture
scm: supply chain management
qa: quality assured
pow: proof of work

aim:

◇ design a blockchain based system in which an information
flow parallel to the real product across organizations is possible
and to provide the end customer with information about manufactured product
◇ blockchain collects data from machines and sensors
◇ create a system in which the following are digitally connected to product
■ production information
■ sub-productions
■ transportation
■ provenance
■ quality
◇ data should be stored to in a way to make undetected changes impossible
◇ should enable traceability of product

• keywords:
◇ hyperledger
◇ blockchain
◇ scm
◇ design science research

◇prototype

introduction:

• need of efficient scm
◇ size of data generated during production and its subsequent accessibility
■ not all data is same and not all data is stored and accessible after production
■ transparency to customers
- information about
→ suppliers
→ geo-location of materials
→ incase of food scm
⇒ processing
⇒ transfer and distribution
⇒ date of sale
⇒ feeding
- if these data is available to customers, they pay extra as it is qa
- helpful in after sales of auto parts
■ increased operational efficiency of traceability system
- less cost during predictive maintenance and recalls
• what is traceability
◇ “the history of a product in terms of the direct properties of that
product and/or properties that are associated with that product
once these products have been subject to particular value-
adding processes using associated production means and

in
associated environmental conditions"
  ◇ traceability system is where traceability information of product is stored and processed.
• why blockchain in scm
  ◇ dlt stores information and ensures unauthorized persons cant change it.
  ◇ tech is lightweight and efficient so complete history of product is stored
  ◇ data stored becomes more valuable
    ■ enables trust in the data
    ■ high availability of data
  ◇ collaboration between various suppliers for a particular OEM possible
• challenges
  ◇ maintain the relation between detailed information retained by supplier and aggregate information transferred to dlt (ie how much data is too much data and is IP violation)
  ◇ assigning data generated to correct product (ie correctness of data fed into the blockchain)

dlt

• central aspects of bitcoin protocol
  ◇ hash algorithms
  ◇ signatures
  ◇ time stamps
  ◇ merkle trees
  ◇ network protocol
  ◇ pow
  ◇ mining

There is a "trust transfer" from the intermediaries to the technical implementation of the Bitcoin protocol: In the Bitcoin protocol, transactions are signed by a participant and published to the network. These transactions are verified by miners and combined into a block of validated transactions. The hash value of the block is calculated with a cryptographic hash function and compared to a specified value, called target. The block is only valid if the hash is below the target. A nonce that is held in the block header is increased until the desired hash value is calculated. The Bitcoin protocol dynamically adjusts the target so the time between blocks is 10 minutes on average. Once a miner has generated a valid block, the miner can append the block to the blockchain and publish it to the network. For each block, the hash value of the predecessor block is contained in the header, which means that later changes and thus a violation of the integrity are only possible with enormous computational effort. The Bitcoin technology illustrates that organizations and individuals trust distributed technologies to an extent to use it as currency.

- hyperledger project
  - ◇ modular structure of five blockchain frameworks
    - ■ fabric
    - ■ sawtooth
    - ■ iroha
    - ■ burrow
    - ■ indy
  - ◇ integrated tools
    - ■ caliper
    - ■ cello
    - ■ composer
    - ■ explorer
    - ■ quilt
  - ◇ consensus algorithm, network participants and other

core elements are interchangable
◇ ledger
■ world state: current state of asset
■ transaction log: entire history of transactions done with asset
◇ hyperledger fabric is permissioned blockchain for enterprise
■ uses chaincode -> similiar to smartcontract
- client side execution
- result is broadcasted to rest of the network
- go, JS, java
◇ PKI
■ for granular role based control using Membership Service Provider(MSP)
- entities who want to enroll in a network would have to ask the MSP
◇ use of channels
■ sidechains which prevent leaking of information from one sidechain to another
■ for privacy in blockchain
■ this is kind of like subnets in networking

the use case automotive manufacturing

• system has product of vehicle with three organizations and end user(owner of vehicle)
• the car is identified via RFID through the sc and different organizations
• steps
1) roof and shell is made in automobile maker(org1)
2) QA control is carried out and roof-shell is sent to transport company(org2)

3) QA report is generated and stored in database
4) associated hash value of generated document with ID is assigned to car with a transaction and stored in blockchain
5) sensors like humidity, position of component, temperature, location tracking(hash of several route points) are measured along transport
6) at end of transport, transport report with inputs from the sensors is made and stored in database
7) car given to car seller(org3)
8) customer at car seller side can see
1- origin of parts
2- quality of parts
3- transport report

methodology

1) problem identification and motivation
1- large amount of information is made during production process
2- this is valuable for participants
3- hyperledger based systems
1> capture information and ensure integrity
2> give means to collaborate
2) objectives of solution
1- refer to aim

3) demonstration of solution

We simulated several runs of the supply chain defined in the scenario. Two microcontrollers were equipped with RFID readers. In addition, toy cars and their individual parts were equipped with RFID tags. The sensors were also connected at the same time. Data on locations, origin, transport routes and quality assurance were simulated within the environment. We have also set up a web server to allow users access through a frontend interface to the blockchain network.

solution architecture

a) software architecture

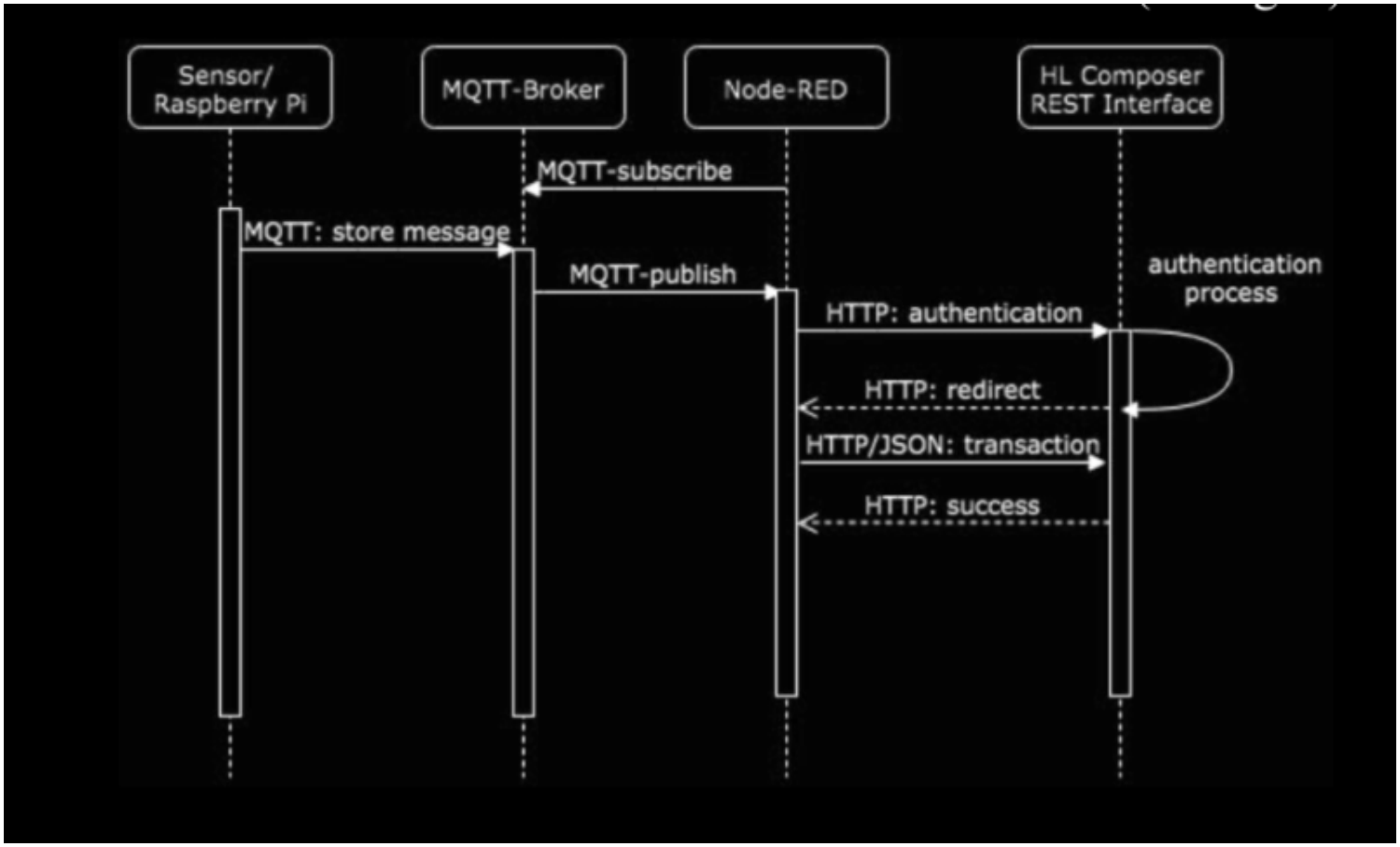
- blockchain for value addition of data
  - ◇ **hyperledger fabric**
  - ◇ for enterprise apps
  - ◇ **Hyperledger composer tool**
    - ■ simplifies creation of apps by hiding certain technical details
- tokenization of assets
  - ◇ connect real world assets to digital data using IoT integration
  - ◇ use **MQTT**
    - ■ Message Queuing Telemetry Transport
    - ■ lightweight and for sending and receiving data
    - ■ uses publish-subscribe pattern
      - sender transmits message to a middle man or broker
      - the message is intended to a recipient but the identity of which is not known to sender
      - broker forwards it to receiver
- allowing user to see the data

◇ use **REST API** to query data
◇ create a **user friendly gui** app
• Integration of all aforementioned via **Node-RED**
◇ open source flow based programming tool enabling rapid prototyping
◇ we can model and program the information flows among users, blockchain and IoT devices

b) hardware architecture
• **NTAG215 NFC stickers** for identifying physical products
• **MFRC-522 modules** for reading the information transmitted from NFC stickers
◇ attached to **raspberry pi B3** via GPIO
• physical products have **Ruuvi sensors** for identifying temperature,humidity etc
◇ these sensors transmit information to the raspberry Pi B3 via bluetooth
• could have several of the data collecting raspberry Pis over short distances connected to internet via 2.4/5G Hz wireless lan

c) communication architecture

1) sensor -> blockchain
• unidirectional connection of IoT to blockchain



• ruvi sends info to rapsberry pi
• pi publishes information labelled with MQTT broker topicsr with predefined python scripts to Node-RED server
  ◇ we can have multiple Node-RED servers for redundancy
• server listens to different MQTT broker topics and send HTTP POST request to transfer infomation to Hyperledger Composer REST interface

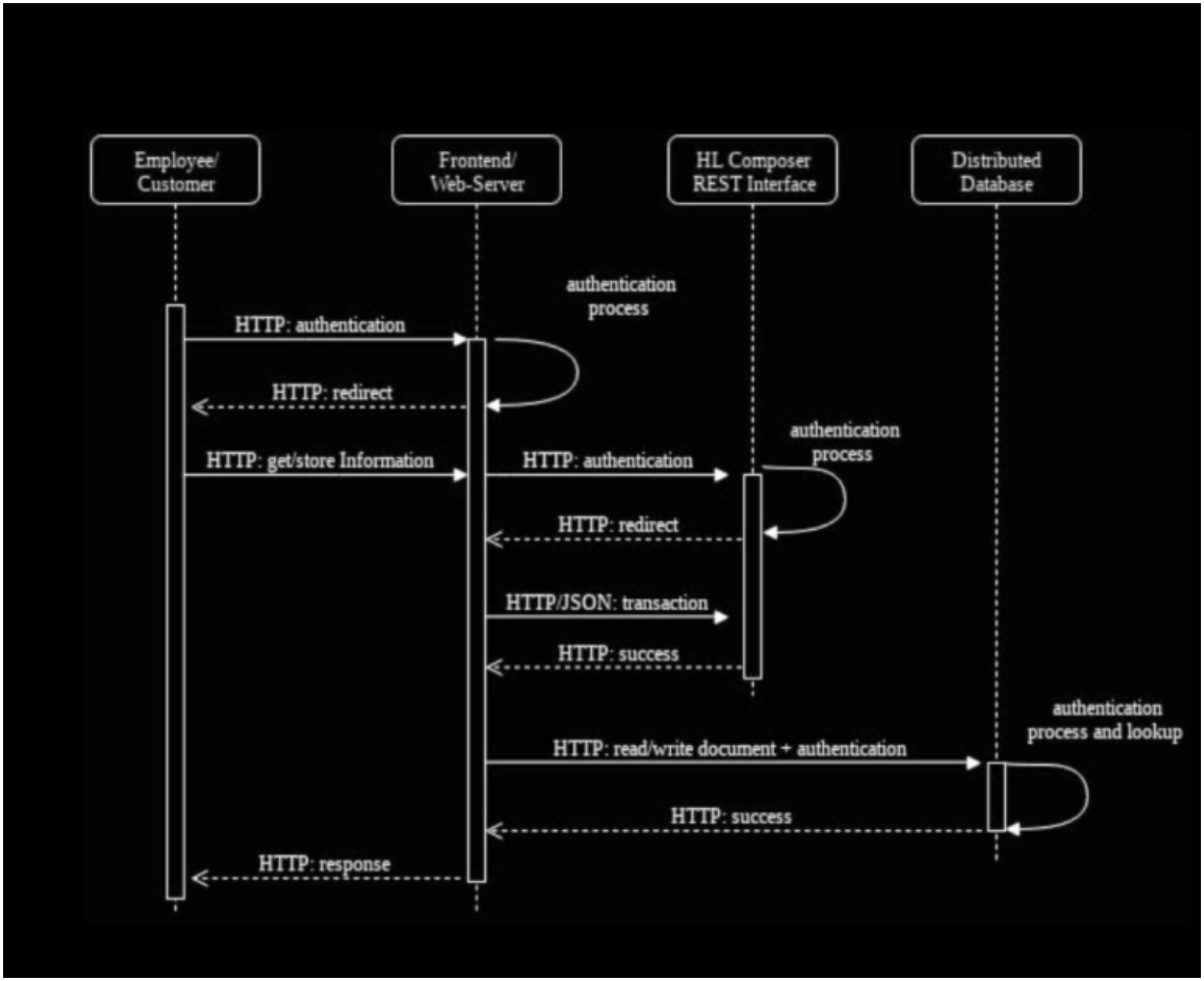
2) employee ↔ blockchain
• employee interact with blockchain with bidirectional connection
  ◇ trigger transactions
  ◇ upload documents via python frontend
• use NoSQL DB called **CouchDB** which exposes HTTP/-JSON API to store documents
  ◇ for interoperability and scalability
3) customer -> blockchain

1- single directional connection between customer (owner of vehicle and blockchain)

2- send HTTP request to webserver for user authentication

1> if authenticated, server forwards request to Hyperledger Composer REST interface via HTTP GET request and obtains data

2> server forwards the data back to customer in HTML format



database architecture

couch DB

- document database
- content is JSON for scalability
- uses REST API so no need to find new ways to connect

client server chain
  ◇ DB entries is stored and queried using HTTP requests
• different DB used
  ◇ one DB for transaction storage
  ◇ one DB as state DB in hyperledger fabric
• **other alternatives to couchDB is RiakKV or MongoDB**
• DB stores documents which are referenced in blockchain by unique ID
• hash of the document is in the blockchain
• ID from blockchain is retrieved and using that the webserver queries DB to get the file and compares calcualted hash with that of the hash in the blockchain to prevent tampering

GUI
• for authenticating with server
• to help employee manually enter information or for uploading transport reports

future work:
• create different channels for every organization
• multiple ways clients to connect to atleast one peer which connects to RED Node
• optimize DB to handle BLOB files to support images or other media objects

supporting-documents

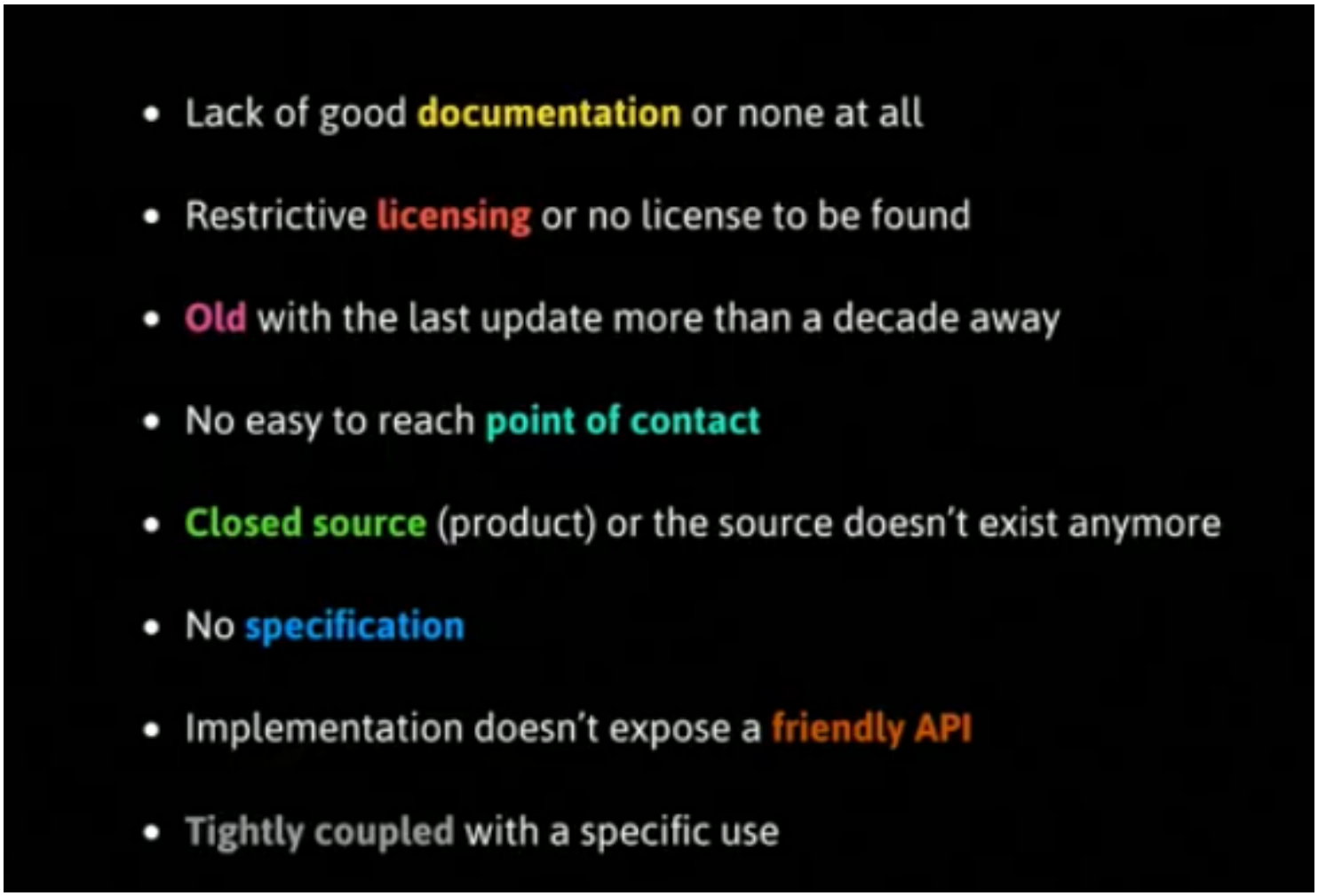
libp2p

word mapping

CAM → Content Address Model
LAM → Location Address Model
IPFS → Inter Planetary File System
p2p --> peer to peer
PAM → Process Addressing Model

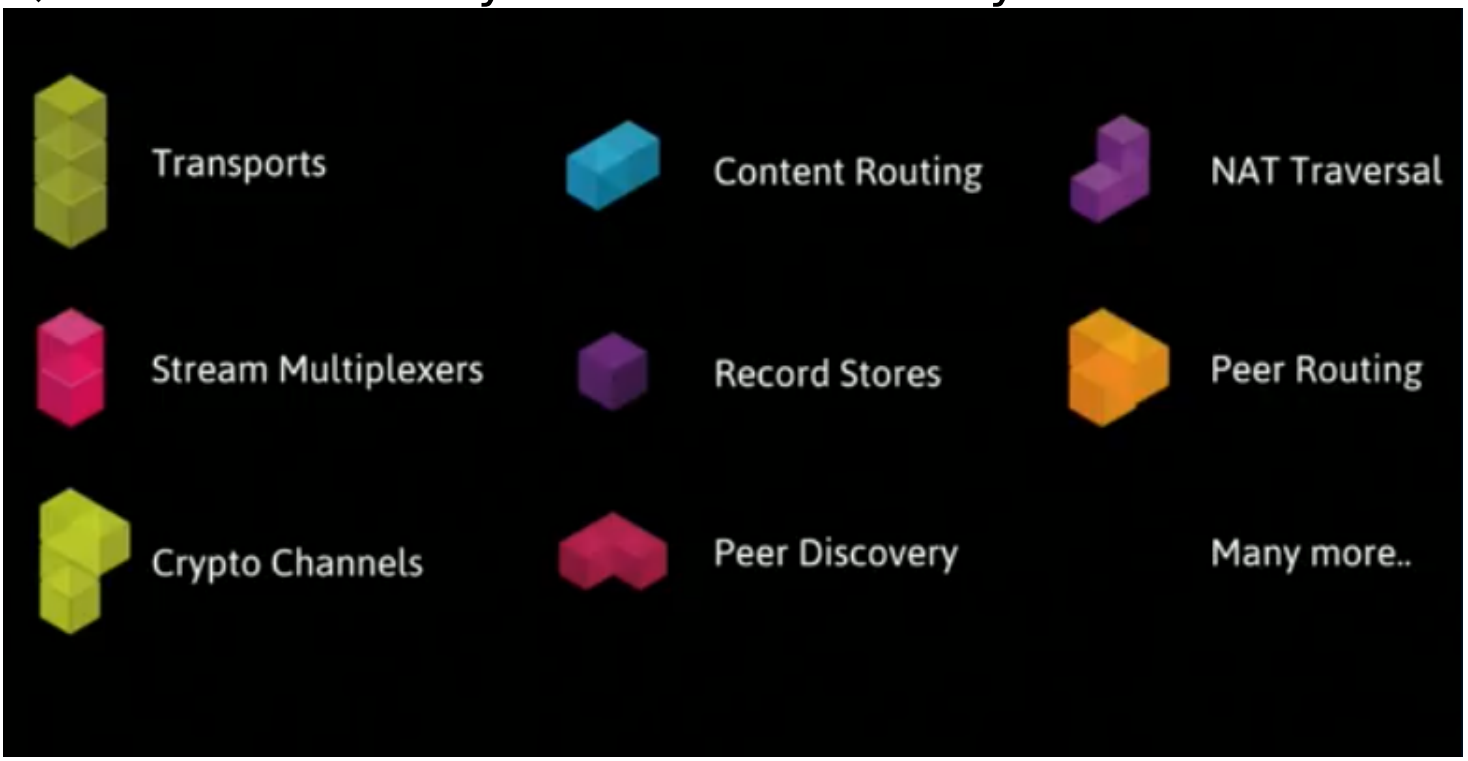
- move from LAM to CAM and PAM
  - ◇ has a lot of problems including reworking networking stack
- problems to IPFS
  - ◇ firewall,NAT → how to keep this from interfering p2p communication
  - ◇ High latency networks
  - ◇ Reliability
    - ■ network speed
    - ■ downtime
  - ◇ roaming
    - ■ how to ensure my processes are found regardless of what machine im using or whether im mobile
  - ◇ censorship
  - ◇ runtimes with different properties
    - ■ different machines/standards have different protocols/rules to transport information
    - ■ we need to link for example IoT with Android with Windows

already existing solutions cannot be resued for various reasons:



• instead of proposing one more different solution to the same problem, libp2p is platform for integrating all existing and future solutions
  ◇ one solution being IPFS
• libp2p is PAM
  ◇ find, connect and authenticate processes, independent of network hops

- very modular
  ◇ use one or many of these to create your own stuff



- all modules are having friendly API for easy integration
- platform agnostic

◇ with minimal modification you can get the same code to run on nodejs or rust or go on different platforms and all communicate with each other

tendermint-protocol

word mapping:

foss → free and open source
pos → proof of stake
poa → proof of authority
pow → proof of work
pbft → practically byzantine fault tolerant

• tendermint is foss consensus mechanism/software using pos or poa instead of pow
• has pbft

working

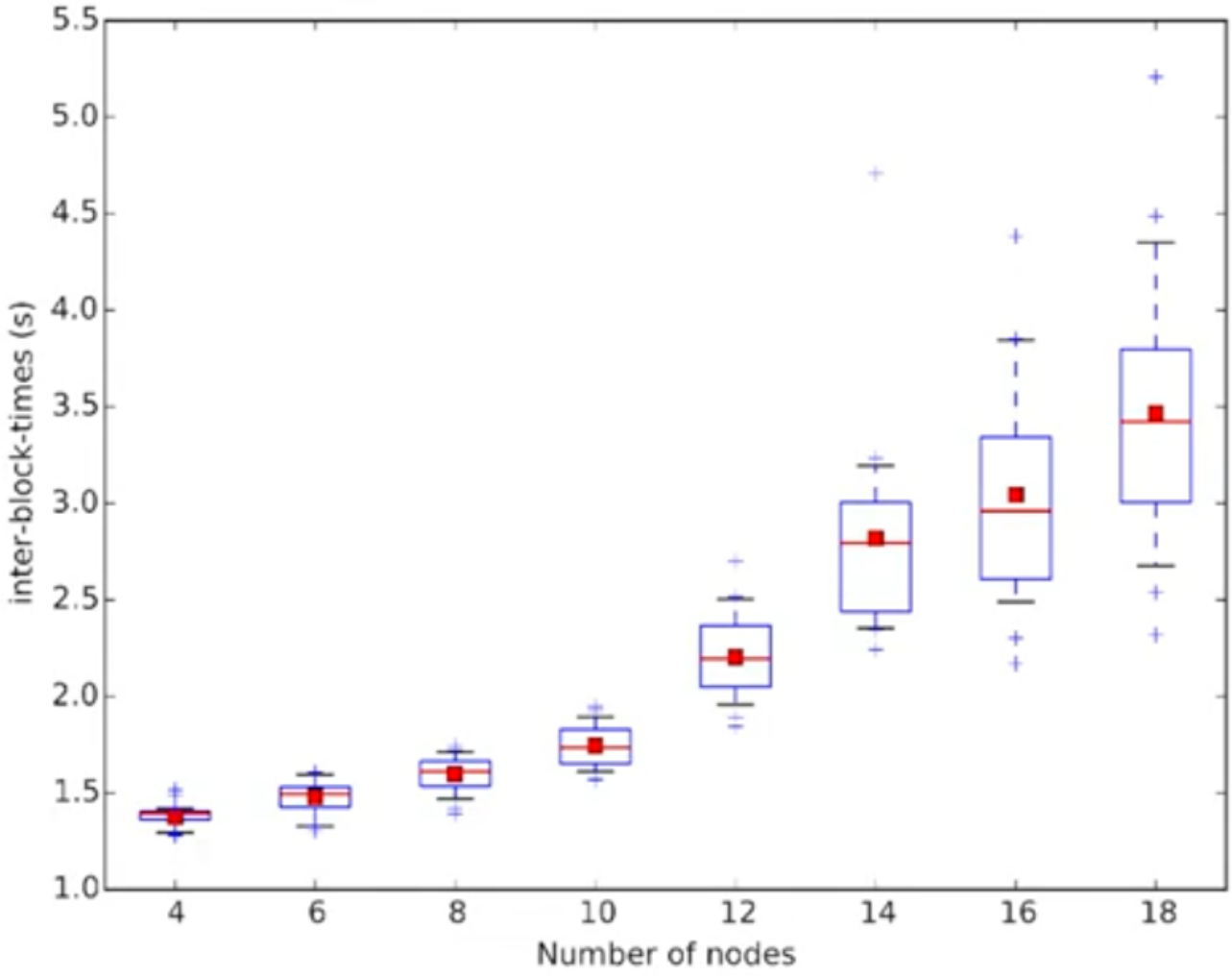
• has always authoritative validator nodes
  ◇ 1-50
  ◇ usually 7 is good, 16-19 is upper end (for strict majority)
    ■ one is proposer, 3 are validators for 2/3rd majority min req
  ◇ problem
    ■ does not scale very well
    ■ for nth node communicate with n-1 nodes(in mesh)
      - grows in quadratic complexity
      - not very scalable

    ■ requires huge bandwidth
  ◇ deployed on different machines
    ■ on different participants, locations, clouds
    ■ done to prevent bias/dominance
  ◇ interconnected
    ■ ie depending on purpose
      - ring
        → needs one or more nodes as relays to get from A to B
      - full mesh
        → ie all validators know all others
        → fastest way of data communication/validation
• Each validator node gets chance to propose a new block via round robin
  ◇ validator can pass the offer to became a proposer (fault tolerant)
  ◇ if proposer fails while in middle of process, transaction is in mempool and timeout for the next block occurs and that block is skipped
    ■ for next block next validator becomes proposer and include not yet processed transaction. (fault tolerant against byzantine failure as long it is minority)
• for each proposal
  ◇ proposer collects all transaction for the last block time, puts in a block for other validators to approve/reject
    ■ block status → approve/reject/pending is known to validator/non-validator
  ◇ if 2/3th majority then go to next phase, enters commit phase
    ■ all validators sent precommit message to all other validators
    ■ if validator get atleast 2/3rd approve message
      - enters into commit state

◇ if precommit does not result in majority of nodes getting precommit then either it will wait till block timeout
◇ after commit all validators will have same block status
◇ result of block is pushed to relevant applications by validators in a couple of 10 milli seconds
• consensus or lack thereof happens in a few 100 milli seconds
• also has non-validator nodes
◇ receives result of consensus which is a new block

• entire block created and available to all nodes in under few secs if number of validators are kept reasonable

Findings from doing a Blockchain Load Test...



PONTON

distributed-id

- question
  - ◇ proving your identity to the internet while keeping your identity(credentials, PII) safe
  - ◇ security vs usability tradeoff
- drawback of traditional systems
  - ◇ username + password
    - ■ hard to remember
    - ■ too many

    ■ desirable target for crackers
  ◇ social identity (SSO)
    ■ dont feel like using social account to login to banking info
    ■ behavior tracking by providers of SSO like microsoft, facebook, google
• DID
  ◇ think of it like if pgp became decentralized
  ◇ convenience
  ◇ direct connection, no SID
  ◇ via decentralization
• Use a wallet application
  ◇ has your private key to prove it is you
  ◇ public key is encoded into a transaction in blockchain
    ■ sovrin
    ■ bitcoin
    ■ ethereum
  ◇ the chain returns a did identifier(derived from public key)
    ■ unique id generated by the ledger to prove my identity
      - only attack is private key being stolen
    ■ only i can say this did is invalid or whatever using my private key
      - ie user in control
    ■ decentrailized so easy integration and visibility
  ◇ key recovery
    ■ follow what metamask is doing using 10-20 random words

how this works?

1) assert to www.example.com that you are did:12345

and sign it with private key
2) www.example.com will visit the blockchain, find the transaction corresponding to did:12345,locate the public key and verify the signature and authenticate user

- link non specific data
  - ◇ like shoe size, favorite color etc to my did
  - ◇ associate any claims/possesions with your did

Parameters

Assembly unit

Manufacturing unit

1) Tranmission
   1- Manual
   2- Automatic
   3- Continuously Variable Transmission (CVT)
   4- Dual Clutch Transmission (DCT)
   5- Semi Automatic
2) assembly type
   1- classic
      1> early days of car assembly
      2> modern day assembly of luxury cars like rolls royce
   3- automated
   4- intermittent
   5- lean
4) Brand
   6- Ford

7- Toyota
8- Jaguar
5) Make (Model year)
6) Mode of fuel
9- Petrol
10- Diesel
11- Electric
7) Geolocation
12- hall sensor
13- GPS sensor -TK103B
8) Autonomy level
14- L0 (full manual)
15- L1 (light driver assistance)
16- L2 (partial automation)
17- L3 (ADAS)
18- L4 (full automatic with manual override)
19- L5 (full automatic)
9) Part manufacturer ID (Bosch, Valeo, Siemens etc)
20- A2C8139490080 -> siemens
21- 0445020175 -> Bosch
10) ruvi sensor time stamped logs from assembly unit
22- temp
23- tilt
24- etc

Final car (NFT)

1) VIN
2) Chasis no
3) Proof for customization/rarity
1- limited edition certification
2- custom paint job
4) results of test run after production
5) easy to read document for QA derived from logs

6) child NFT

ruvi sensor feeding data to Manufacturing unit as time stamped logs, understandable processed info will be made available to driver as a easy to understand document

how-to-write-problem-statement

- problem statement should
  ◇ put the problem in context (what we already know)
  ◇ describe the precise issue(