SOLVING PROBLEMS

Blockchain at Sabay

The problem

- Sabay is collecting money for digital services
- Payment landscape in Cambodia is scattered
- No cooperation, many different system
- Many problems with Payment API's
- Issues with dispute resolution about payments
- 2017 Sabay introducing for new product which depend on micro payments (SOYO), very high workload for our teams

The problem

- 2018 Sabay Group is collecting money for many different micro services and group partners
- cooperate with 24 different payment gateways to collect
- settle several hundred thousand micro payments per month
- New product launch planed for 2020 adding 10x of transactions to the system
- by 2020 predicted collapse of current infrastructure ☹

What do we want

- Connect all payment options to our network
- Allow payment providers and merchants to work together
- Allow payment providers and merchants to make their own agreements
- Fast payments, 2-3 second to confirm
- Allow others to offer services
- Use the network for more then just payments

What we **don’t** want

- To issue a currency
- having anyone trust us (we should be able to prove it)
- Settle payments for others
- Be in competition
- Holding money in our accounts
- License to run a payment network

Blockchain to the rescue

- Simple proposition, move to blockchain !
- Many blockchain projects can process high volume of transactions securely
- Blockchain project are secure, decentralized and distributed

THE PROMISE OF BLOCKCHAIN

What can it do

The promise of blockchain

- a Blockchain is a ledger (book) with digitally signed entries
- The author of the entry is certain (**authenticity**)
- Ensure the entry was not altered (**integrity**)
- The author can not deny the entry (**non-repudiation**)

The promise of blockchain

- Each block (page) of the book is linked to the previous block (**continuity**)
- The content of a page once added can not be altered (**immutable**)
- No single entity is in control of what entries are added to the ledger (**decentralized**)
- The ledger is distributed across several systems (**distributed**)
- The system is able to achieve **consensus** as to what version of the ledger is correct

How can this be achieved ?

Entries

- Authenticity
- Integrity
- Non-repudiation

Block / Page

- Continuity
- Immutable

System

- Decentralized
- Distributed
- Consensus

MESSAGE INTEGRITY

Hash functions

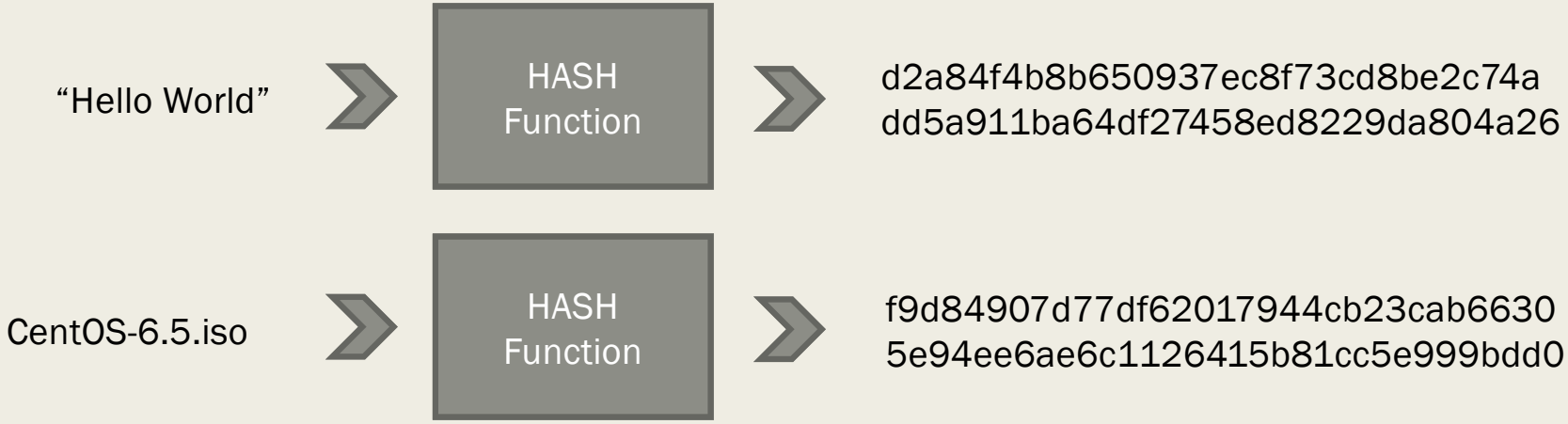
Digital signatures

Verifying a message/ ledger entry

1. Ensure the message was not changed
2. Ensure we know who send it
3. Ensure only the authorized person could have created it
4. Doing this in a way which can not be faked

Breaking it down – Hash functions

A hash function converts data of any size into a string of a fixed size

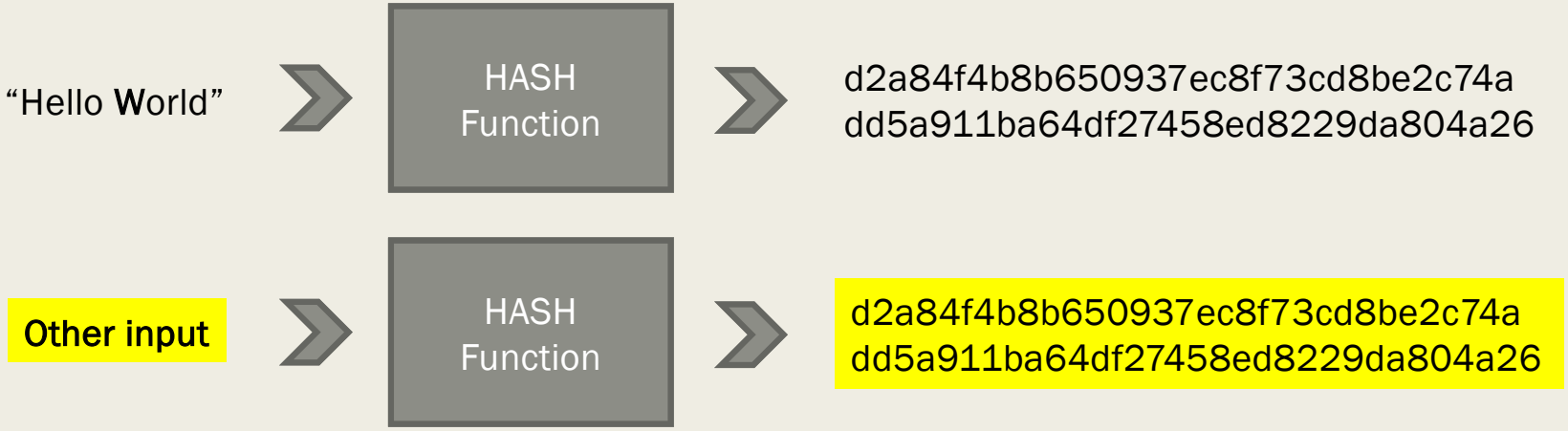


Hash functions

A small change in the input will result in a totally different hash

Hash functions

When 2 different inputs produce the same hash,
we get a **hash collision** and the hash function becomes obsolete



Hash functions

A hash function is consider “collision resistant” when it is **very hard***
to find two inputs that hash to the same output

“Hello World” → HASH Function → d2a84f4b8b650937ec8f73cd8be2c74a
dd5a911ba64df27458ed8229da804a26

~~Other input~~ → HASH Function → d2a84f4b8b650937ec8f73cd8be2c74a
dd5a911ba64df27458ed8229da804a26

*e.g. needing petaflops of power

Hash functions

Hash function	Hash length	Secure	Published
MD5	128 bits (32 characters)	No (not collision resistant)	1992 (2013)
SHA1	160 bits (40 characters)	No (not collision resistant)	1995 (2017)
SHA2 (256)	256 bits (64 characters)	Yes	2001
SHA2 (512)	512 bits (128 characters)	Yes	2001
SHA3 (224)	224 bits (56 characters)	Yes	2015
SHA3 (512)	512 bits (128 characters)	Yes	2015

Digital signature

- Digital signature is equivalent of a handwritten signature, insofar that there are:
  - *know samples of the signature (which can be verified) and*
  - *there is only one person who can produce the signature (non-repudiation)*
- But we all know handwritten signatures can be faked
- So can we also fake digital signatures?

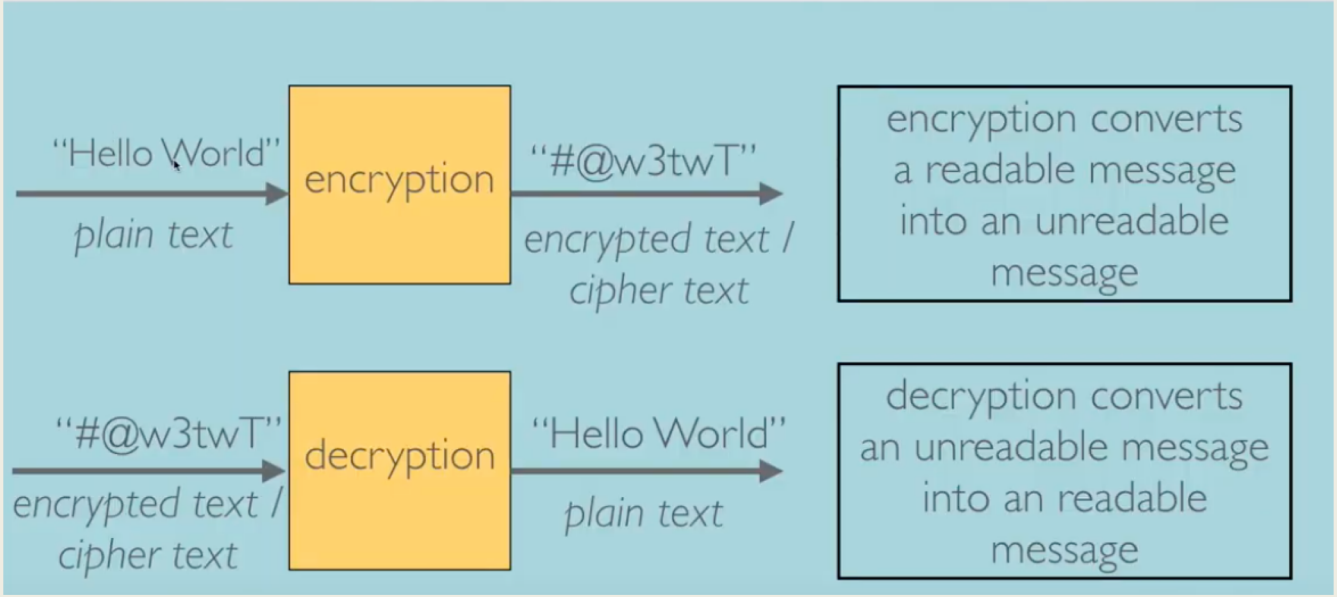
Cryptography, encrypt, decrypt

Cryptography converts data into a particular form

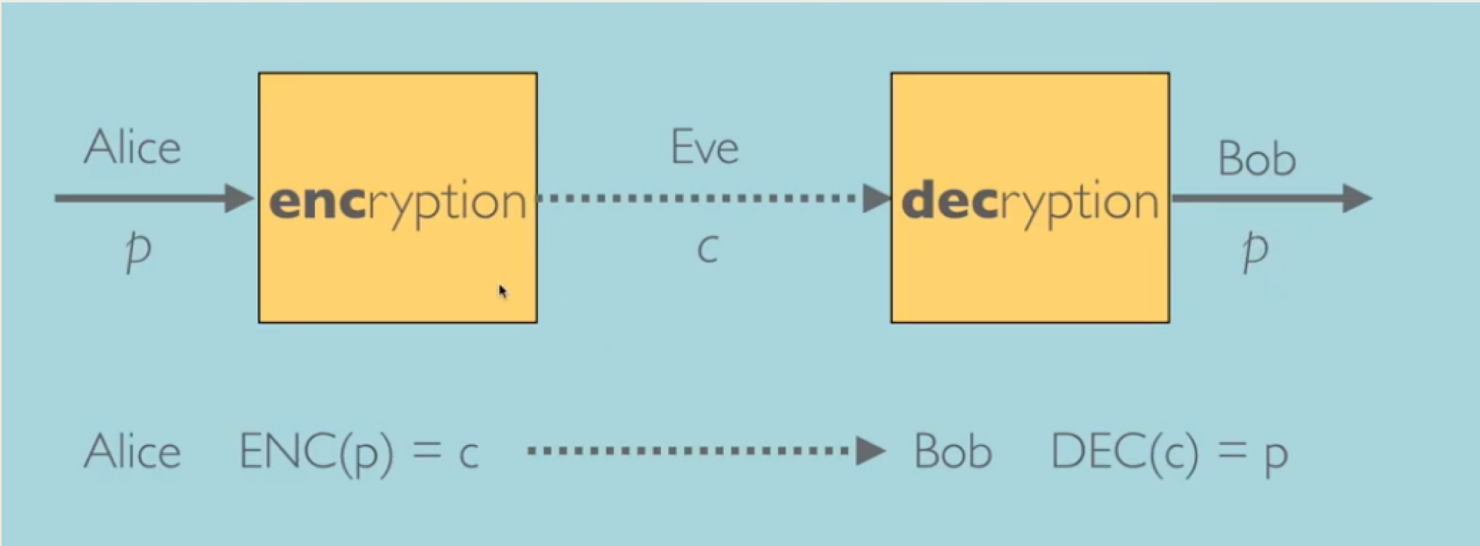
so that only those for whom it is intended can read it

The converted data is unreadable for unauthorized user.

Encryption / decryption



Encryption / decryption



Encrypt it? How?

- In order to encrypt and decrypt text we need some sort of a key
- Two basic techniques to encrypt data symmetric and asymmetric
- Symmetric key algorithm require the **same key** for both encryption and decryption
- But both sender and receive must know the key, and any 3rd party learning the key can also decrypt the message
- Symmetric key algorithms are AES (Advanced Encryption Standard), Triple DES (Data Encryption Standard)
- Symmetric key encryption can not be used for signatures

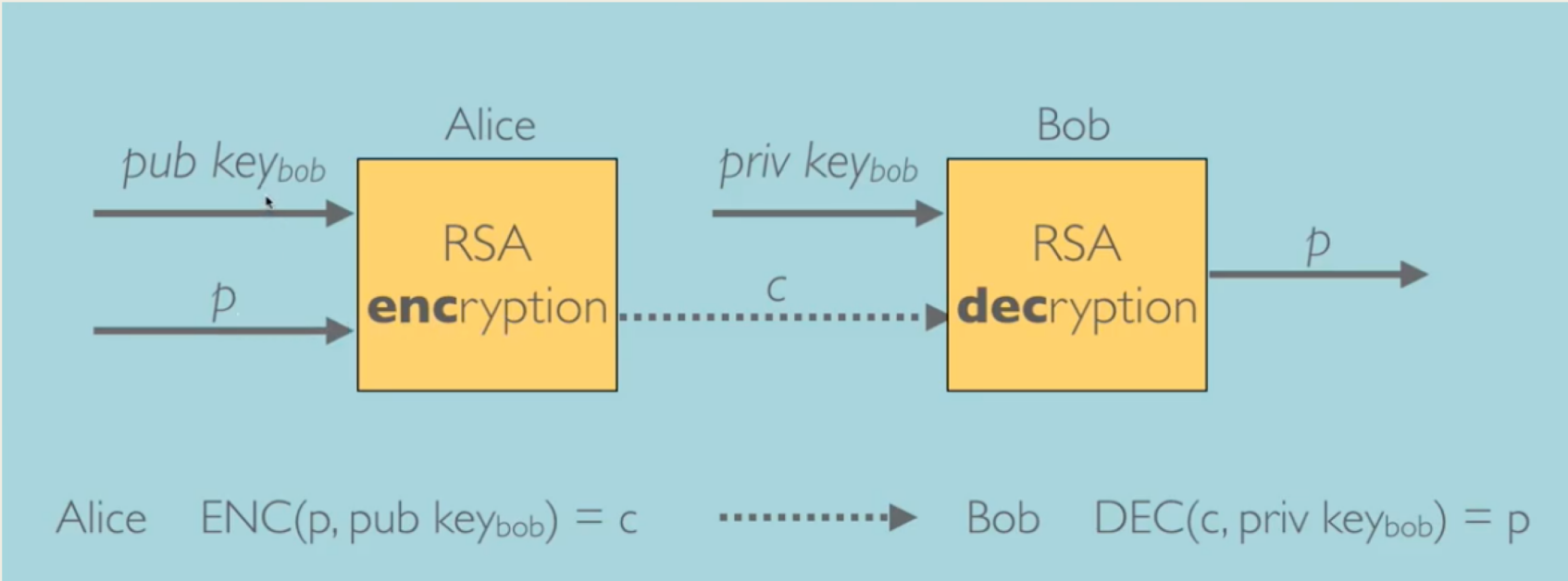
Asymmetric key algorithm

- Asymmetric key algorithm, require two keys called a public and a private key
- The public key is used for encryption
- The private key is used for decryption
- Public key and private key are mathematically interconnected
- **One public key only has one corresponding private key**
- Anyone is allowed to know the public key

Asymmetric key algorithm

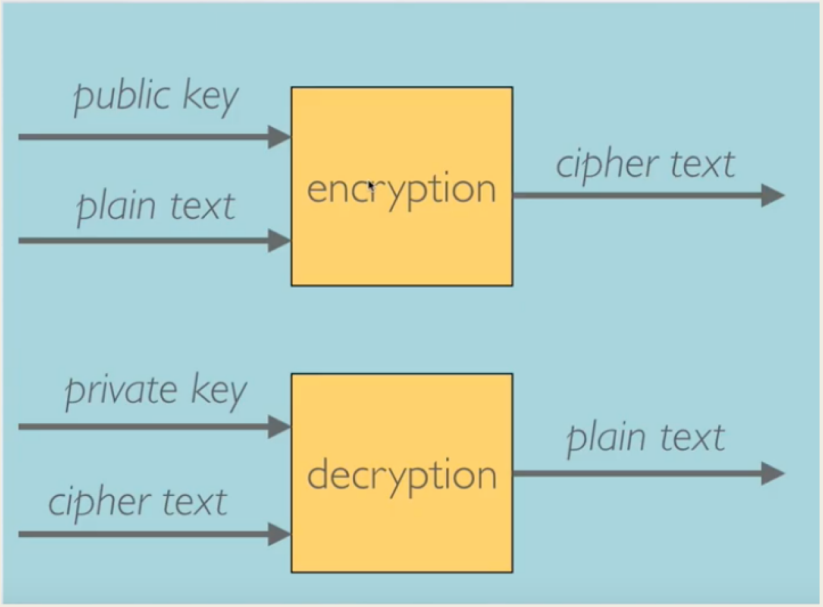
- Any sender can encrypt a message for the receivers public key
- but only the receiver can decrypt the cipher text using the private key
- Asymmetric key algorithms are:
  - *RAS (Rivest Shamir Adleman) PGP is based on this*
  - *ECDSA (Elliptic Curve Digital Signature Algorithm)*
- Most blockchains use ECDSA

Asymmetric key algorithm

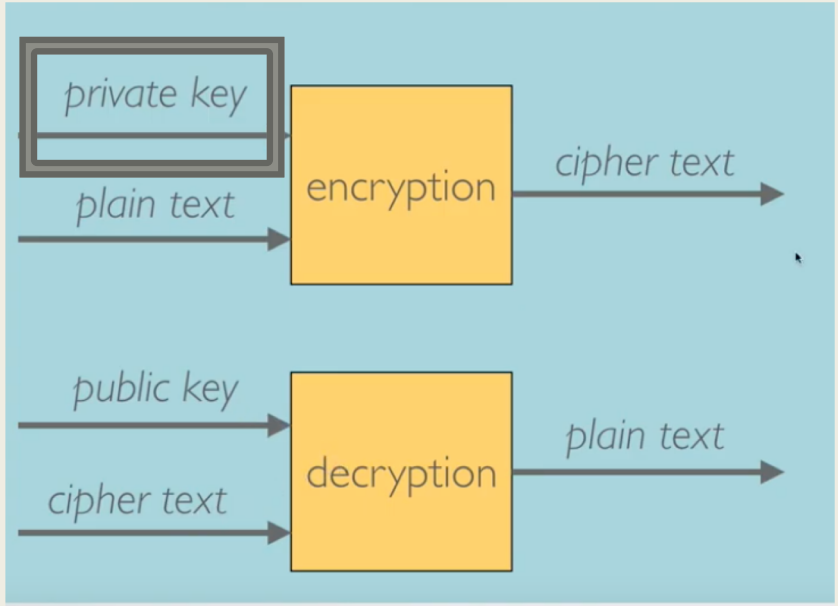


Digital Signatures

Encryption



Signature



Digital Signatures

- **A public key; is a account address on the blockchain**
- A ledger entry; has a message, a hash of the message and a signature
- The digital signature are the cipher of the message hash encrypted with the secret key corresponding to the public key*
- The signature can be decrypted with corresponding public key
- The decrypted signature must match the hash of the original message
- This allows anyone to verify the authenticity and integrity of the message

**YES! that's why we need collision free hash functions*

THE LEDGER

The book of truth!

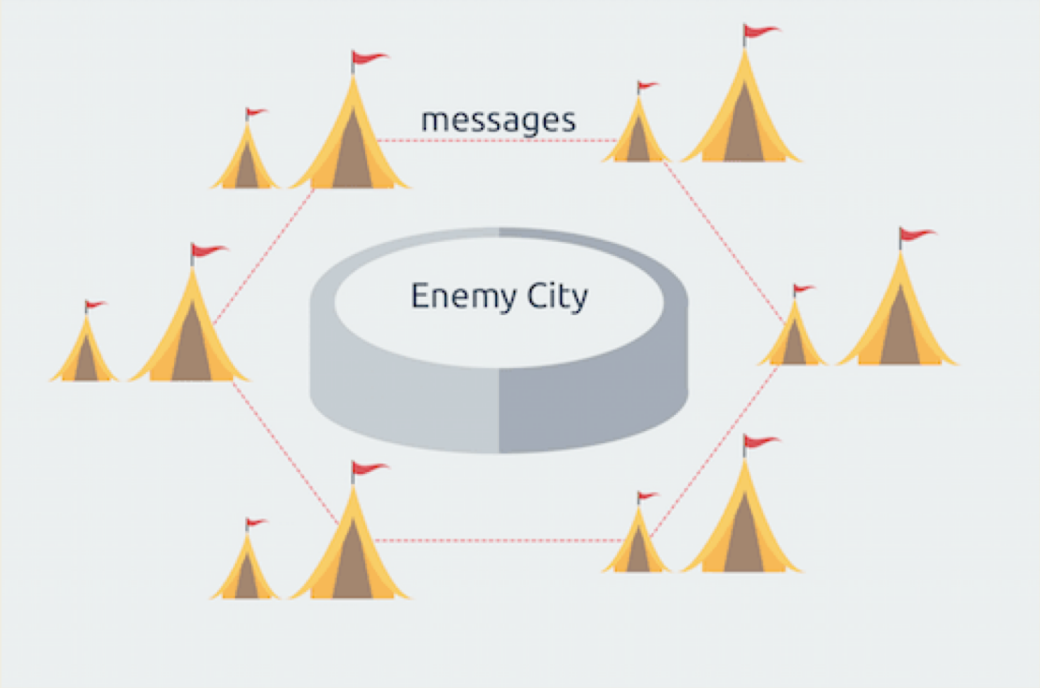
Blockchain ledger

- A kind of database where confirmed/validated transactions are recorded
- Blockchains do not have a central ledger (decentralization)
- Each node has a full copy of the ledger (distributed)
- Ledgers may only contain balance information (bitcoin) or may also contain other information (Ethereum)
- Each ledger contains the hash of the previous ledger (continuity)
- Each ledger contains its own **hash** proving the integrity of the ledger

What is in the ledger?

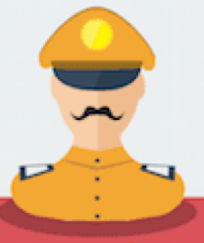
- In any distributed system without a central authority in order to reach consensus we need to solve the Byzantine Generals Problem
- Meaning we have to deal with arbitrary failures of the participants:
  - *including lying,*
  - *fabrication of messages,*
  - *collusion with other participants,*
  - *selective non-participation*

Byzantine Generals Problem



Byzantine Generals Problem

- For instance, if nine generals are voting,
- four of whom support attacking - while four others are in favour of retreat,
- the ninth general may send a vote of retreat to those generals in favour of retreat, and a vote of attack to the rest !!!



	The Situation	
Agree on a Strategy	Objective	Agree on Valid Transactions
Separated Camps	Spacial Distribution	Distributed Nodes in the Network
Loyal Troop and Loyal Generals	The Good Ones	Truthful Nodes
Traitors	The Bad Ones	Evil Nodes
Corrupt a Message	The Attack	Add an Invalid Transaction to the Blockchain
How to Know which Message is True	The Problem	How to know which Transaction is Valid
Don't Have	A Solution	Proof of Work
Don't Have	Consensus	Blockchain with More Combined Difficulty

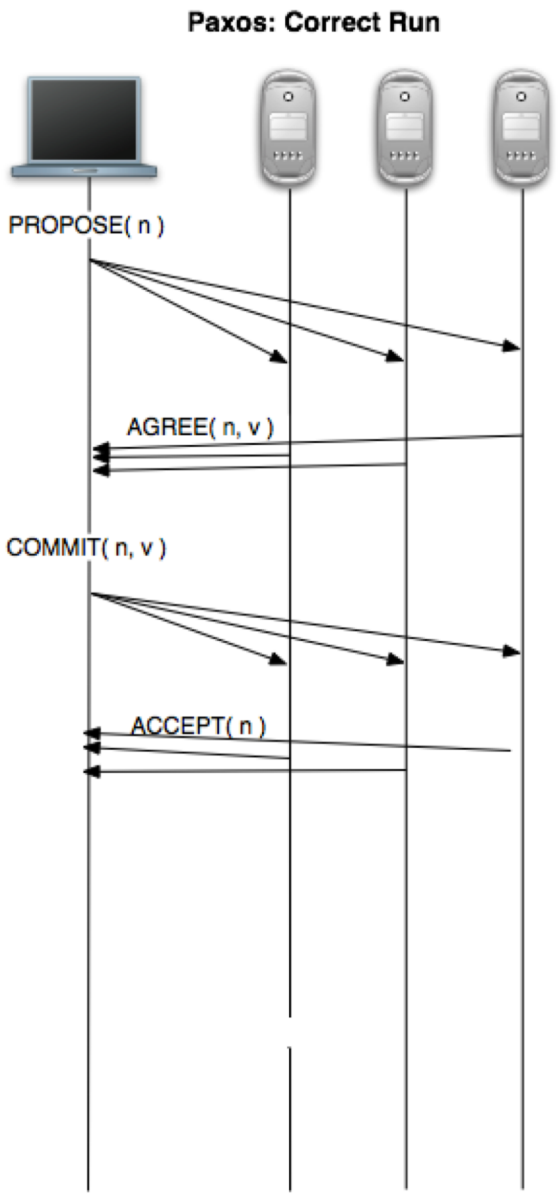
BFT was solved in 1982 and the are several modern software solutions

Bitcoin used Byzantine Fault Tolerance (BFT) and Prove of Work to overcome failure

No single actor can dictate the content of the ledger

What is in the ledger?

- Sabay decided not to use a PoW blockchain, slow transaction speed, high power consumption
- Stellar implemented a Federated Byzantine Agreement System (FBAS) based on PAXOS,
- FBAS allows the network to reach consensus by defining trust pools and extending trust to 3rd parties on agreement



BUILDING A NETWORK BUILDING SSN

Working together

SSN a open, distributed

ledger

to records

payment authorization

- Open
- Neutral
- Fast
- Scalable
- Secure
- Trustless
- Immutable

Building infrastructure to verify payments

https://ssn.digital

Building a network – tokens (IOU)

- On the Stellar blockchain any account can issue tokens (similar to ERC20 Ethereum tokens)
- Other accounts must trust the issuer in order to hold the token
- issuer must accept the trust request
- When the token is returned to the issuer, the token is destroyed

Building a network - tokens

- Payment Providers issue tokens on SSN
- Tokens symbolize the payment has been authorized on the payment providers platform **by the customer, for the merchant**
- Payment provider signs token creation with a digital signature
- Merchant will see the transaction in his account on the blockchain
- Merchant can be 100% sure the payment is been made

Tokens mean what?

- If the **merchant receives** a token, it means the customer has paid and the payment provider owns then merchants is owned the amount
- The transaction contains a memo with the payment details needed by the merchant

- **Sending the tokens back** to the issuer, can then be interpreted as a request to payout, to settle the amount own by the payment provider to the merchant
- The transaction memo will then contain information to which account to settle the amount

But …

- Not every merchant has an account with every payment provider to settle tokens!
- SSN has a build in decentralized exchange which allows
  - *Trading of tokens*
  - *Path payment , atomic swap of token during a payment*
- The trading function of the network allows any merchant to “sell” unwanted tokens to anyone on the network who also trusts the token issuer

Token trading

- If the merchant has tokens from payment provider A and B but only wants to settle with provider A
- Merchant will offer token from provider B in exchange for token from payment provider A
- However due to demand there maybe is no market to swap tokens from provider B
- If provider A is very (very) popular for payouts, and payment provider B is very popular with customers for payment, there will be a imbalance of tokens and special providers could offer a exchange service, for a commission

Path payments

- Suppose a merchant has payment provider A tokens but he want to cash out with provider C
- Stellar has a automated system to find any trading exchange to get from A to C
- On the network there are two trades open; from
  - *A to B*
  - *B to C*
- During a path payment, the system will automatically evaluate those offers and make the payment

Token Settlement

- Using the build in network function for exchanging tokens we can build a advanced settlement network, based on the need of the merchants
- Money will stay in escrow accounts with payment providers and only move during settlement to the merchant accounts
- No need to compare records between merchant and provider
- Settlement providers can offer service to help merchants to settle with their providers

REAL WORLD IMPACT?

Going digital

The network effect

- Using the simple mechanisms to create and destroy tokens and attaching specific meaning to it; allows us to construct a network where merchants can access payment providers as needed
- Every new payment provider joining the network, adds to the networks growth
- Every new merchant, can with one connection access existing connections, rather then having to build every connection themselves.
- Payment providers and merchants are free to work with each other (or not)

Knowing that …

- Both parties agree digitally to provide and accept payments
- Providers and merchants “own” their account key
- Only account owners (the one with the key) can issue transactions
- SSN can not change any records*

*So don’t lose your keys!

MAKING IT WORK

@Sabay

The work at Sabay

- Documenting all flows and developing guidelines for use cases
- Developing API's to support blockchain functions, e.g. building transactions
- Developing system for merchants to tell payment providers about the details of a payment request
- Developing guidelines for account security and multi signature transactions
- Specifying account configuration to allow KYC for merchants

Technical Regulations

technical regulations are procedures that must be followed when participating on SSN.

- TR001 Account Security settings
- TR002 Payment address and payment details
- TR003 SSN QR Code Technical Details Based On EMV Co Standard
- TR004 Merchant Settlement process (cash out)
- TR005 Network fees and native tokens
- TR006 Account KYC settings

Technical Guides and work flows

Technical guides describe services and solutions offered on SSN and how to implement them

- TG001 Implementation guide for payment providers / cashiers
- TG002 Mobile top up over SSN (as provided by Sabay Digital)
- TG004 Implementation of a payment address resolver for merchants

SSN 3rd party API 1.0.0 OAS3

/api/api-ssn-digital.yaml

This is the API for 3rd parties services for SSN

Contact the developer

Servers

https://api.ssn.digital/v1 - Production server

verify access to network and account configuration

POST `/verify/trust` verify if an account accepts an asset

POST `/verify/signature` verify a hash signature

The work at Sabay …

- Developing merchant dashboard to support trust request to and settlement with - payment providers
- Developing tools for payment providers to accept merchants
- Changing all of Sabay’s payment receivers to accept payment on the block chain
- Integrating all of Sabay’s business units and teaching them about the new system

A FEW EXAMPLES IN PYTHON

If you want to play on our test network drop as request in telegram and we will seed your account

Creating accounts

```python
from stellar_base.keypair import Keypair
from stellar_base.operation import CreateAccount
from stellar_base.transaction import Transaction
from stellar_base.transaction_envelope import TransactionEnvelope as Te
from stellar_base.horizon import Horizon

# new secret key https://lab.testing.ssn.digital
ALICE_SK = "SBMXAA26CONB26WH54ZSWRR3H3IA3SCTXNV7CMKIJVI7KJC5DWD2YF25"
ALICE_KP = Keypair.from_seed(ALICE_SK)

# who has money
SEED_PK = GBWDQ3I5DAKB3L0YKT4K0BZMBISQ0JR5FAQWYYSIKXQ3ECI3C53IGAKQ
SEED_SK = 'no no no no no'

horizon  = Horizon( "https://horizon.testing.ssn.digital" )
tx = Transaction(
    source=SEED_PK,
    sequence= horizon.account( SEED_PK ).get('sequence'),
    fee=10000000,
    operations= [ CreateAccount( destination=ALICE_KP.address().decode(), starting_balance='10') ]
)

# Build a transaction envelope, sign it, hand it to stellar
envelope = Te( tx=tx, network_id= 'ssn_testing_network' )
envelope.sign( SEC_SK )
response = horizon.submit( envelope.xdr() )
```

Checking Balances

```python
from stellar_base.address import Address
import json

# new secret key https://lab.testing.ssn.digital
ALICE_SK = "SBMXAA26CONB26WH54ZSWRR3H3IA3SCTXNV7CMKIJVI7KJC5DWD2YF25"
ALICE_KP = Keypair.from_seed(ALICE_SK)

address = Address(a
    address     = ALICE_KP.address().decode(),
    network     = 'ssn_testing_network',
    horizon_uri = 'https://horizon.testing.ssn.digital')
address.get()

print('Balances: {}'.format( json.dumps(address.balances ,indent=4, sort_keys=False)))
```

Making payments

```python
from stellar_base.builder import Builder

ALICE_SK = "SBMXAA26CONB26WH54ZSWRR3H3IA3SCTXNV7CMKIJVI7KJC5DWD2YF25"
BOB_PK   = 'GA7YNBW5CBTJZ3ZZOWX3ZNBKD6OE7A7IHUQVWMY62W2ZBG2SGZVOOPVH'

builder = Builder(
    secret     = ALICE_SK,
    network    = 'ssn_testing_network',
    horizon_uri = 'https://horizon.testing.ssn.digital')

builder.append_payment_op(
    destination=BOB_PK,
    amount='2.25', asset_code='XLM'
    ).add_text_memo("Thanks for the Coffee")

builder.sign()
response = builder.submit()
print(response)
```

Learn more … ask us

Website

https://ssn.digital

Telegram:

https://t.me/ssn_digital (or scan QR)

Documentation

https://github.com/sabay-digital/org.ssn.doc.public

APIs

https://api-reference.ssn.digital

Testing lab

https://lab.testing.ssn.digital

SCAN HERE TO APPLY
INNOVATION
AND
INSPIRATION
@SABAY
"WHAT WILL
YOU CREATE?"
sabay.com

sabay
BE PART
OF SOMETHING
AWESOME
SCAN HERE
TO APPLY
"JOIN THE
SABAY TEAM!"