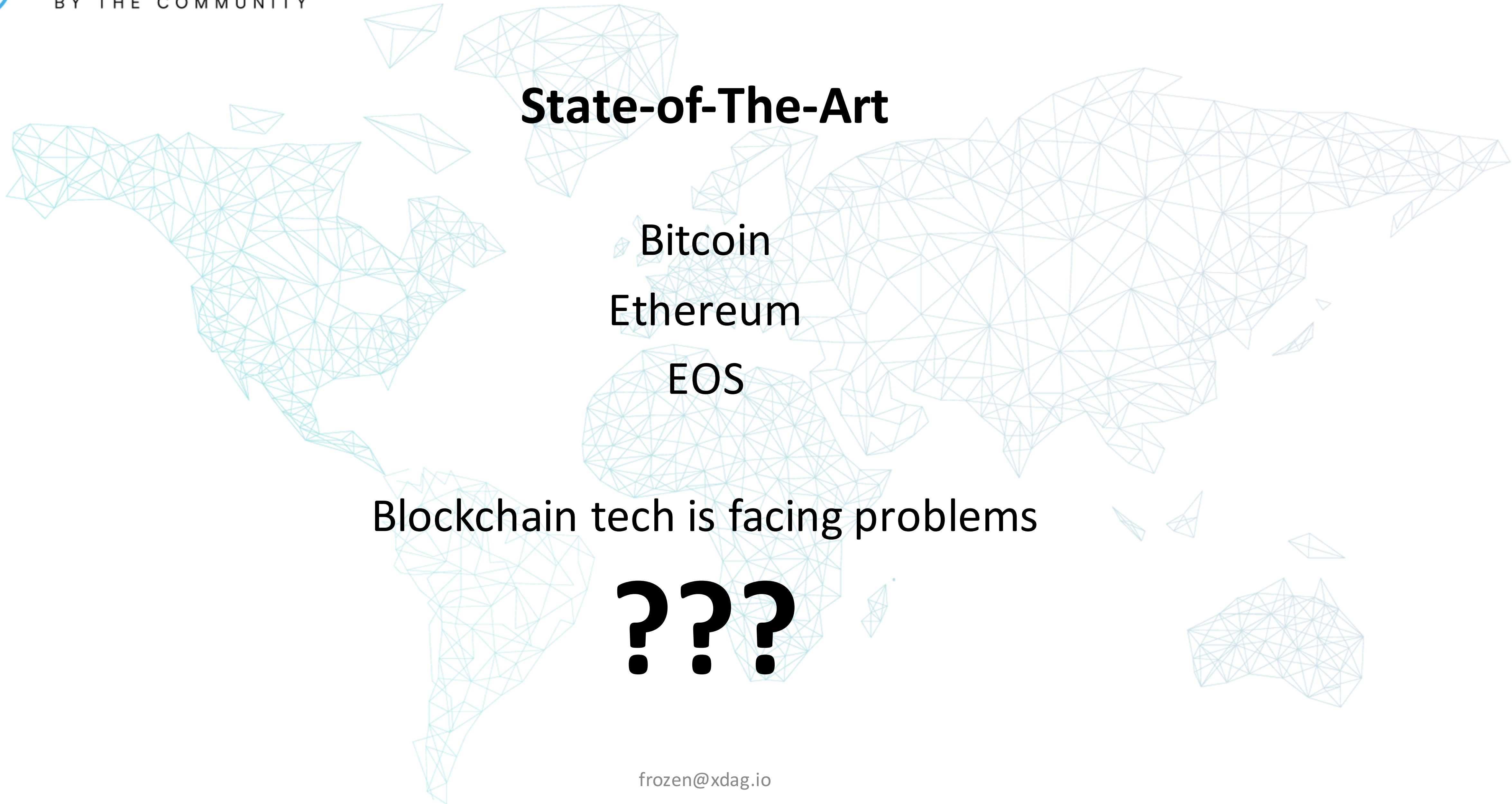
DAGGER
BY THE COMMUNITY

XDAG: PoW + DAG

frozen@xdag.io

https://github.com/xrdavies



State-of-The-Art

Bitcoin

Ethereum

EOS

Blockchain tech is facing problems

???

State-of-The-Art

Layer1

- Consensus PoS/DPoS
- Shading

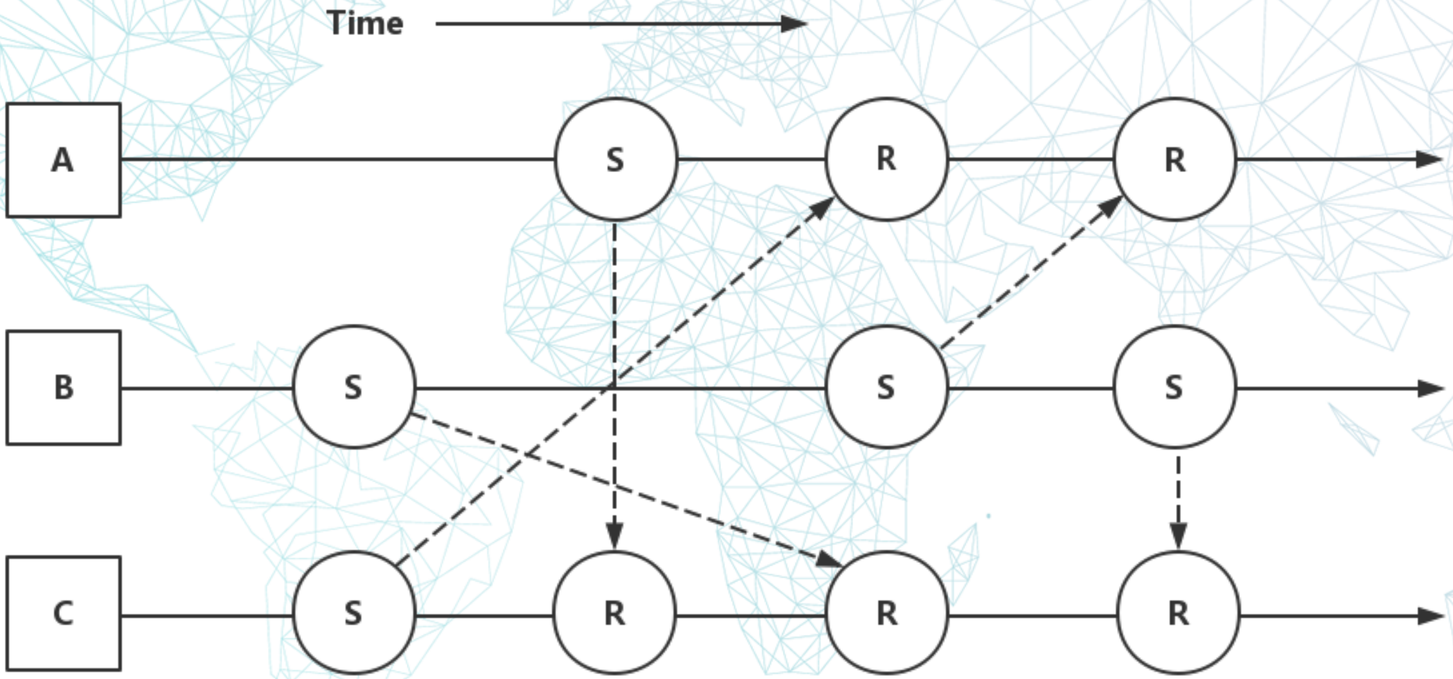
Layer2

- Side Chain
- State Channels
- Multi Chains

DAG

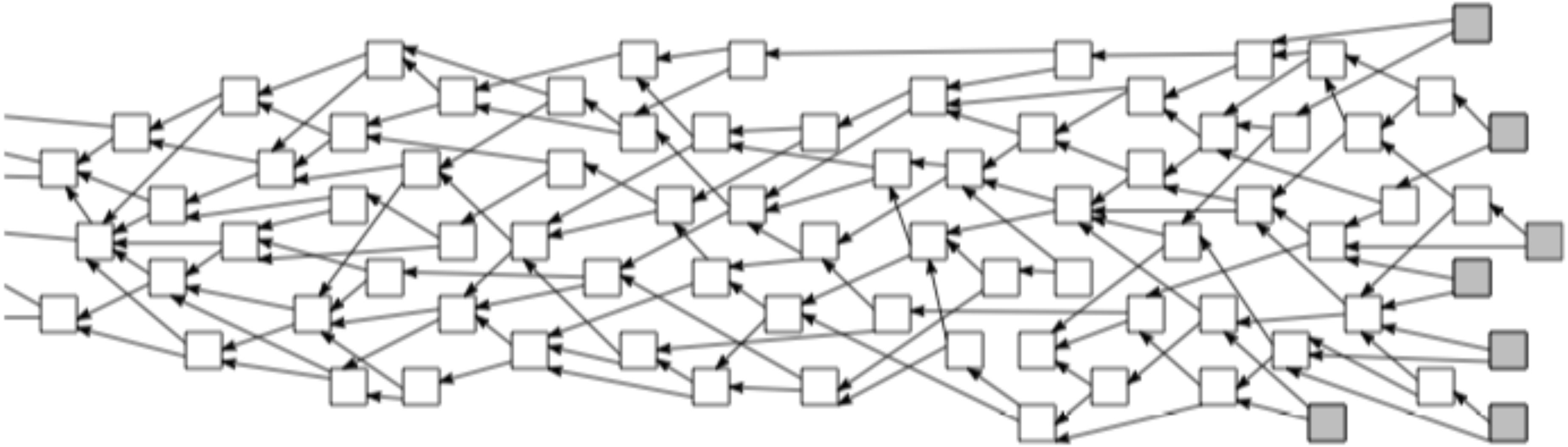
- Nano
- IOTA
- Byteball
- XDAG

NANO



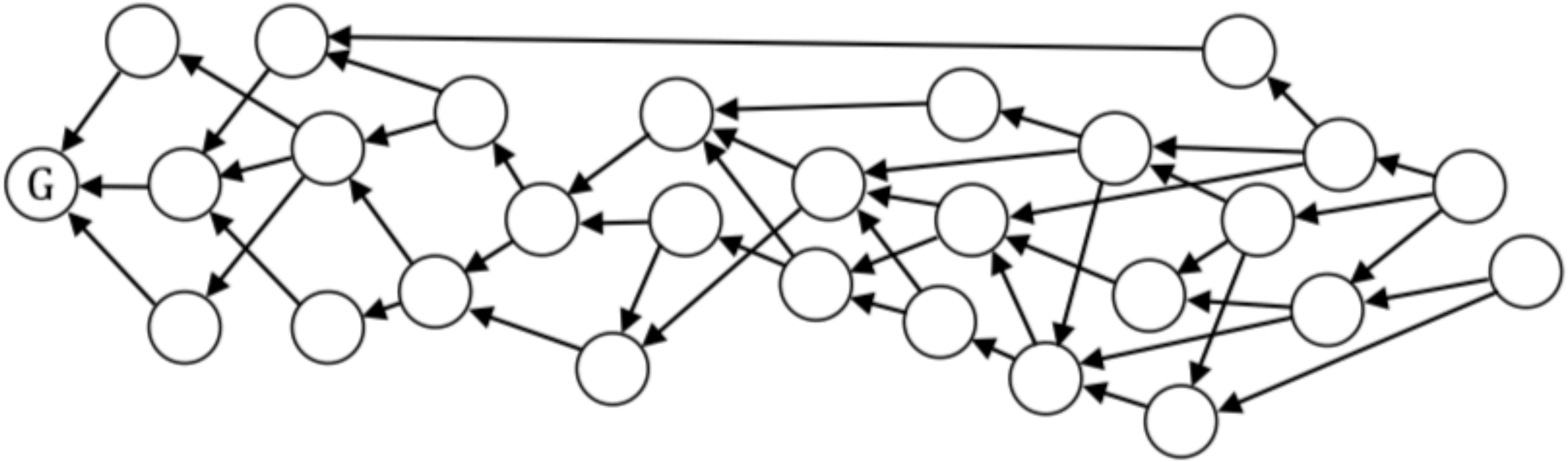
DAGGER
BY THE COMMUNITY

IoTA



frozen@xdag.io

Byteball



DAGGER
BY THE COMMUNITY

DAG – Directed Acyclic Graph

- XDAG is another innovation technology to solve problems

DAGGER
BY THE COMMUNITY

XDAG: A new DAG-based cryptocurrency

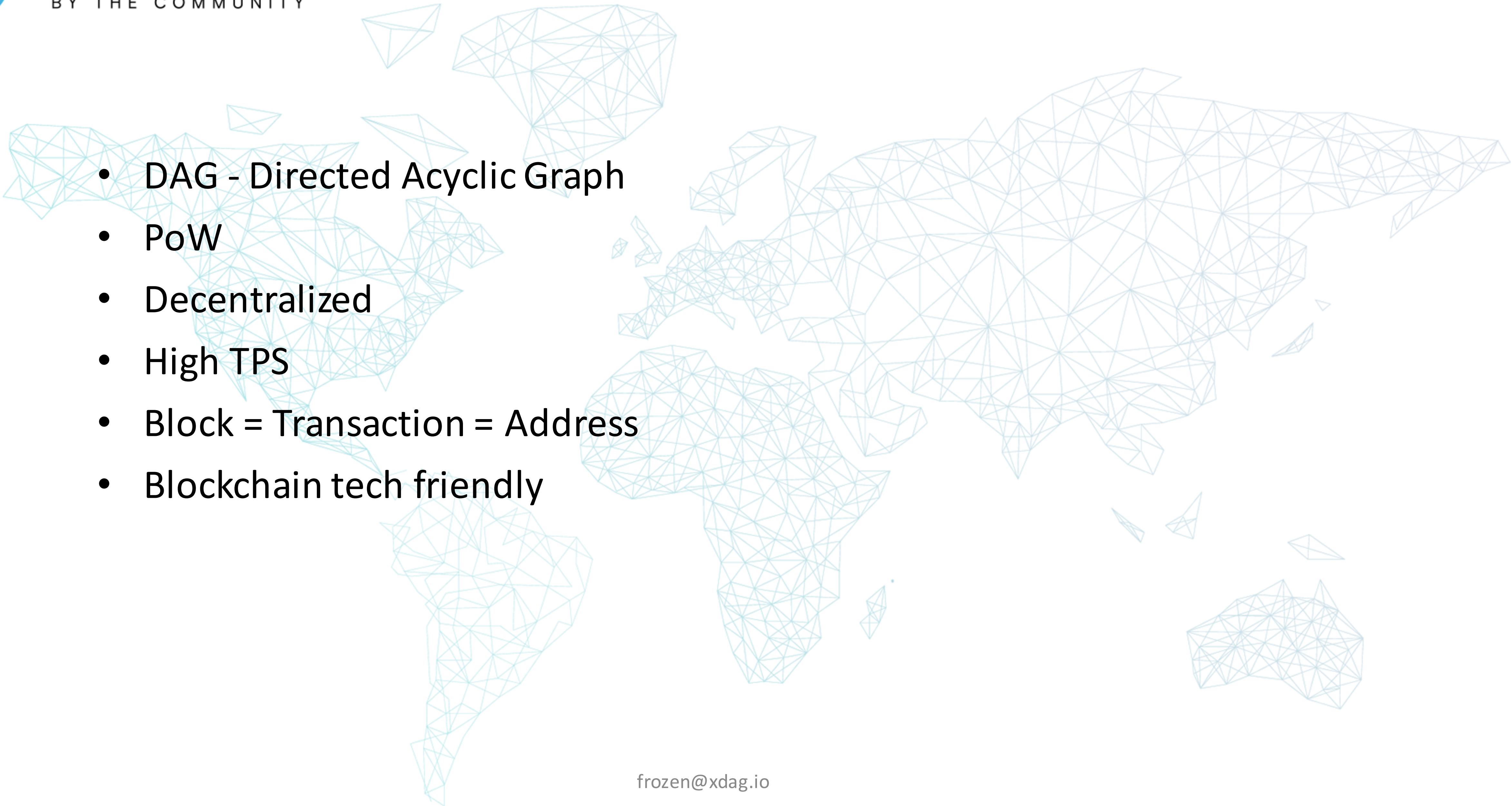
The first mineable DAG

No Pre-mine

No ICO

Community driven

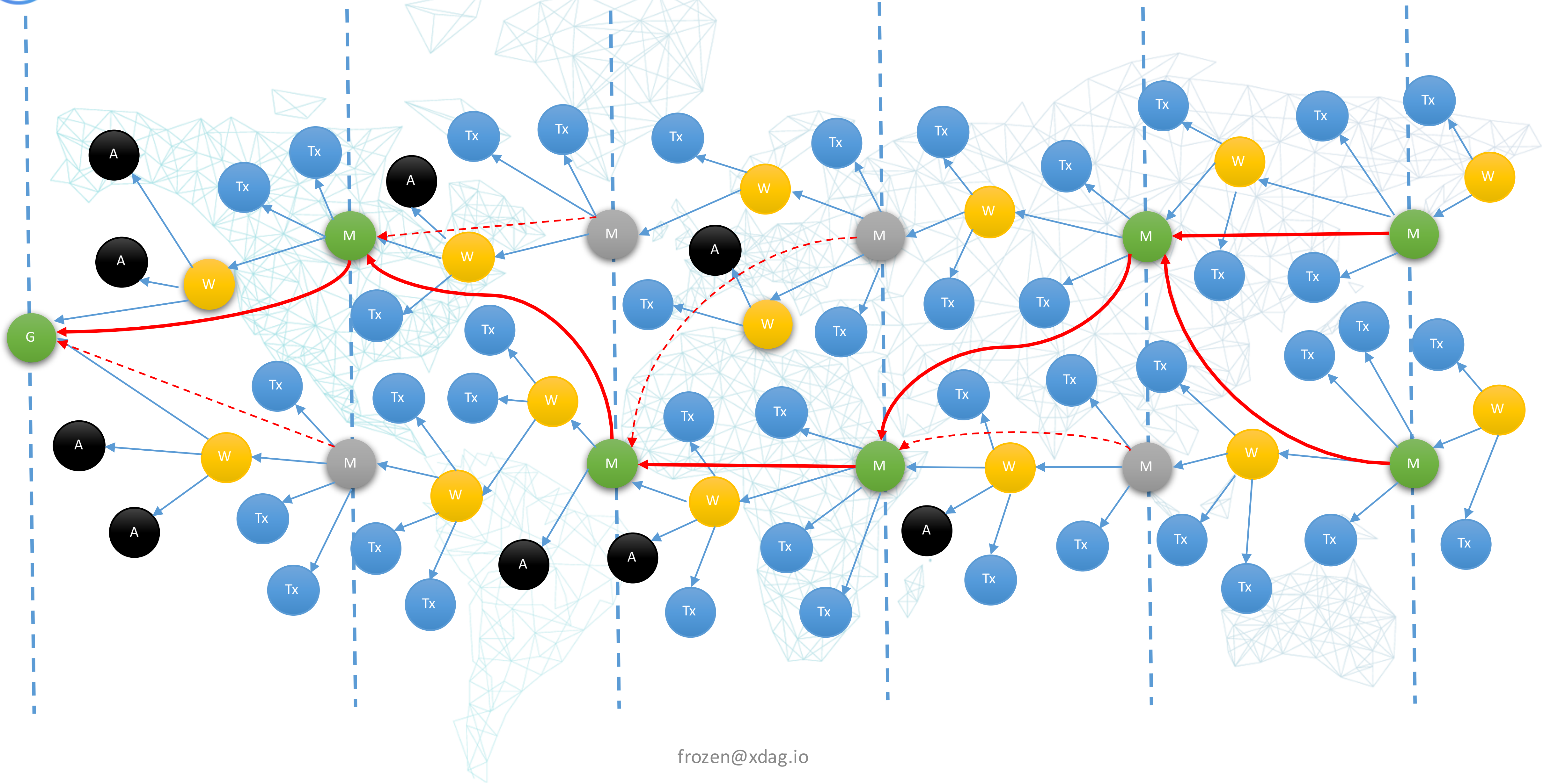
frozen@xdag.io



- DAG - Directed Acyclic Graph
- PoW
- Decentralized
- High TPS
- Block = Transaction = Address
- Blockchain tech friendly

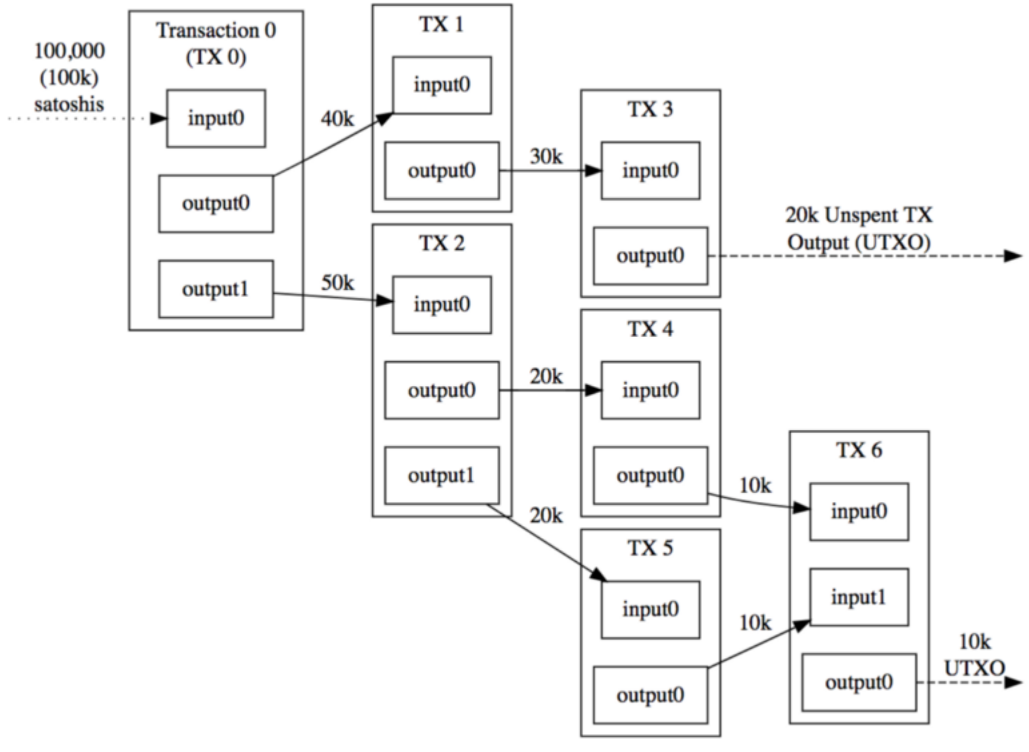
DAGGER

BY THE COMMUNITY



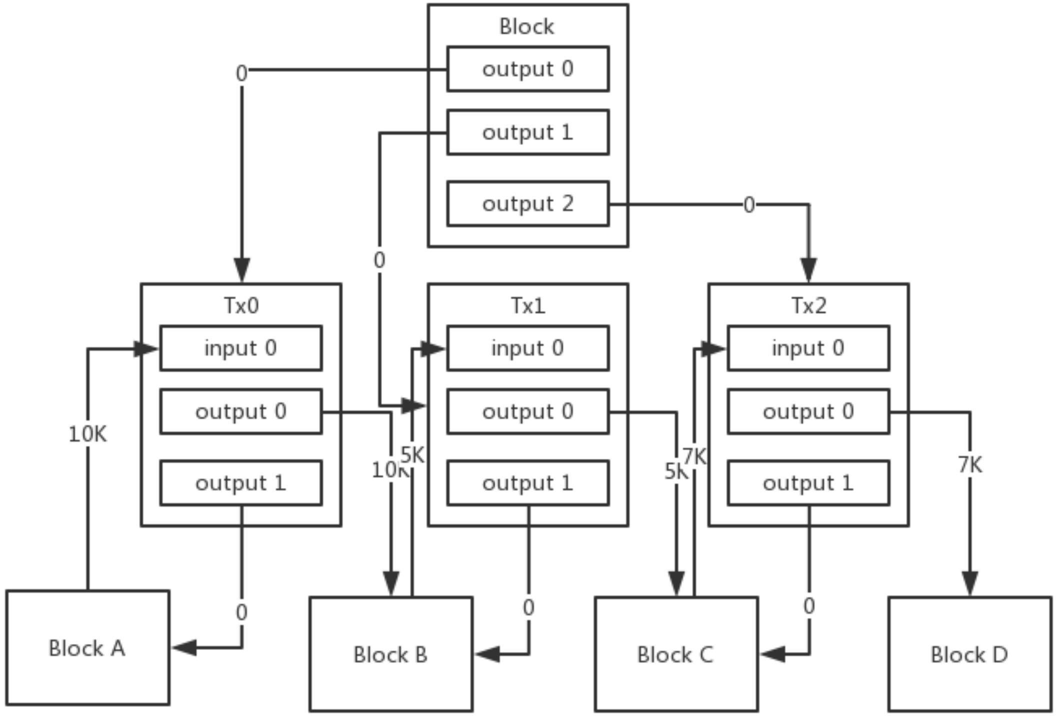
frozen@xdag.io

UTXO

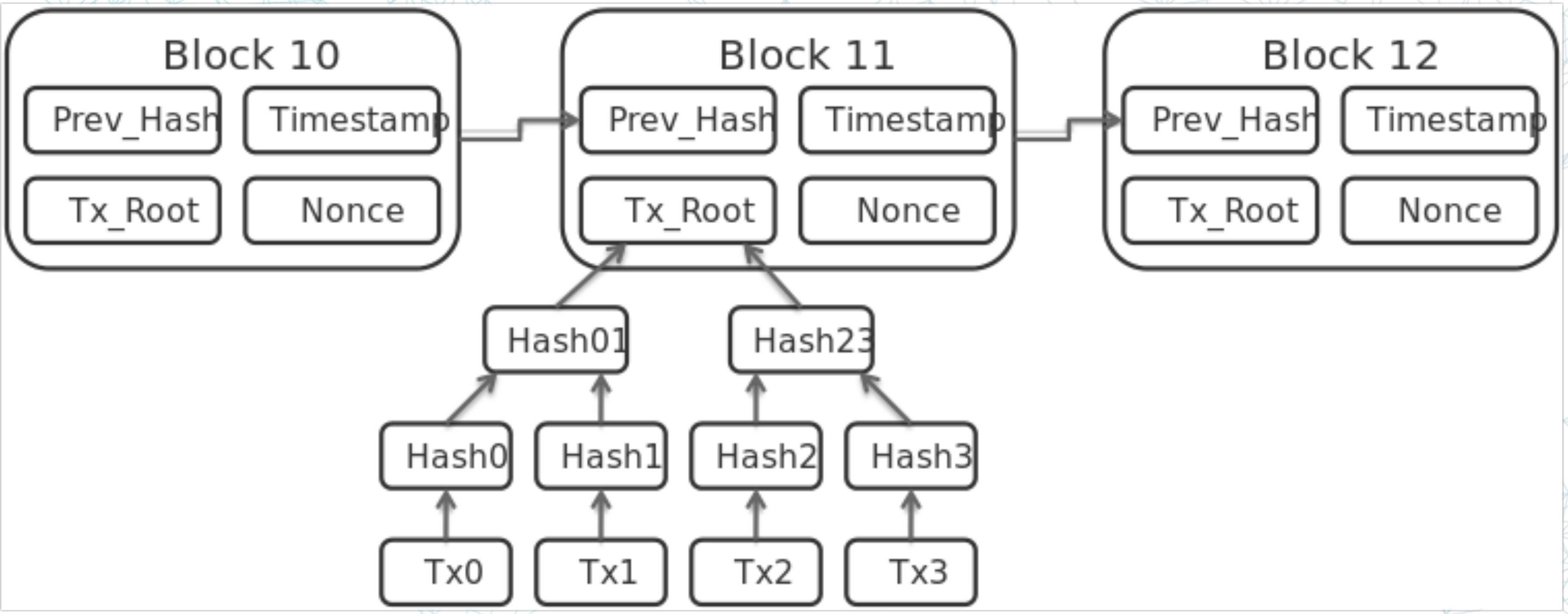


Triple-Entry Bookkeeping (Transaction-To-Transaction Payments) As Used By Bitcoin

XDAG UTXO

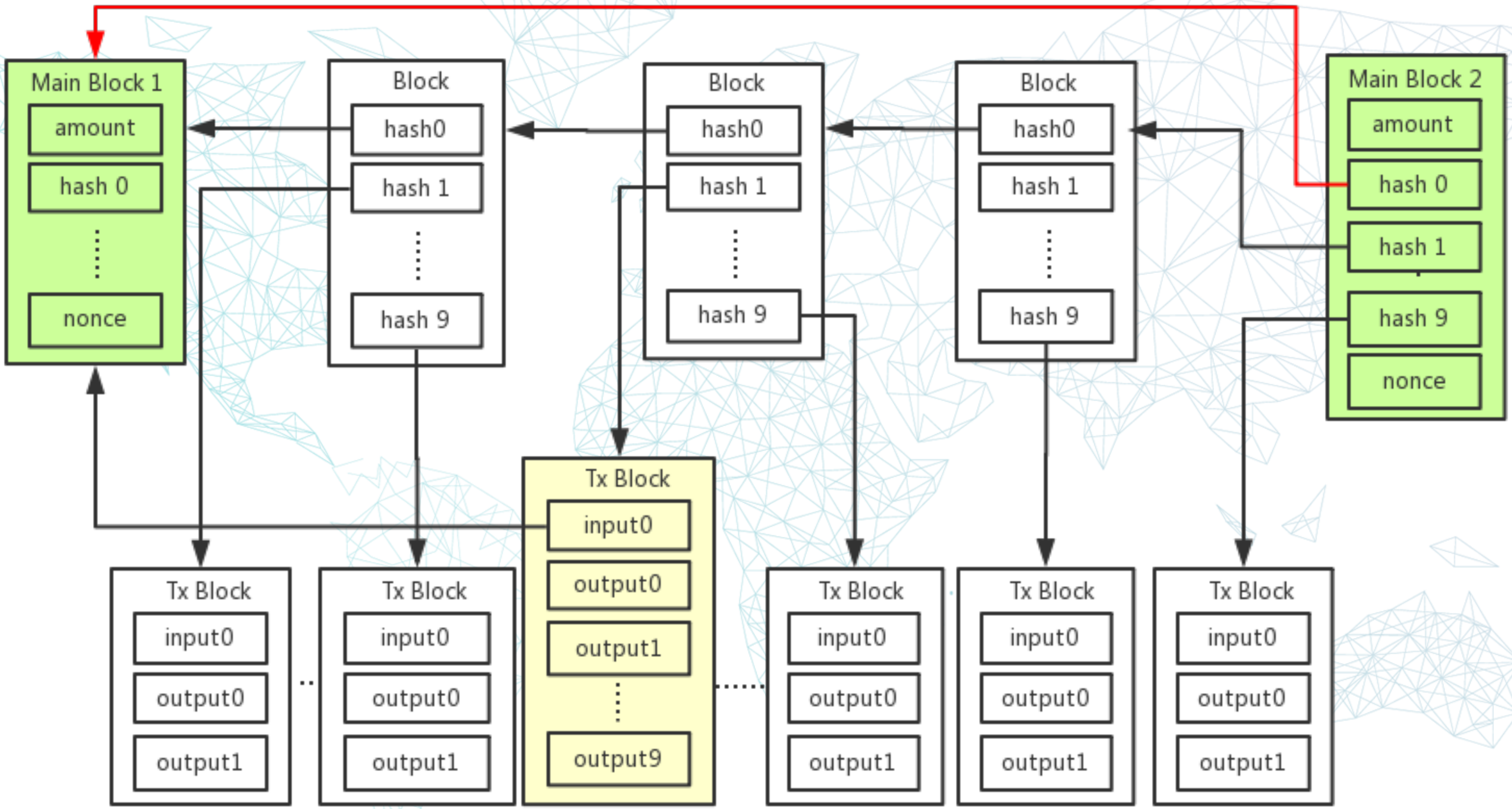


Blockchain


Block 10
Prev_Hash
Timestamp
Tx_Root
Nonce
Block 11
Prev_Hash
Timestamp
Tx_Root
Nonce
Block 12
Prev_Hash
Timestamp
Tx_Root
Nonce
Hash01
Hash23
Hash0
Hash1
Hash2
Hash3
Tx0
Tx1
Tx2
Tx3

DAGGER
BY THE COMMUNITY

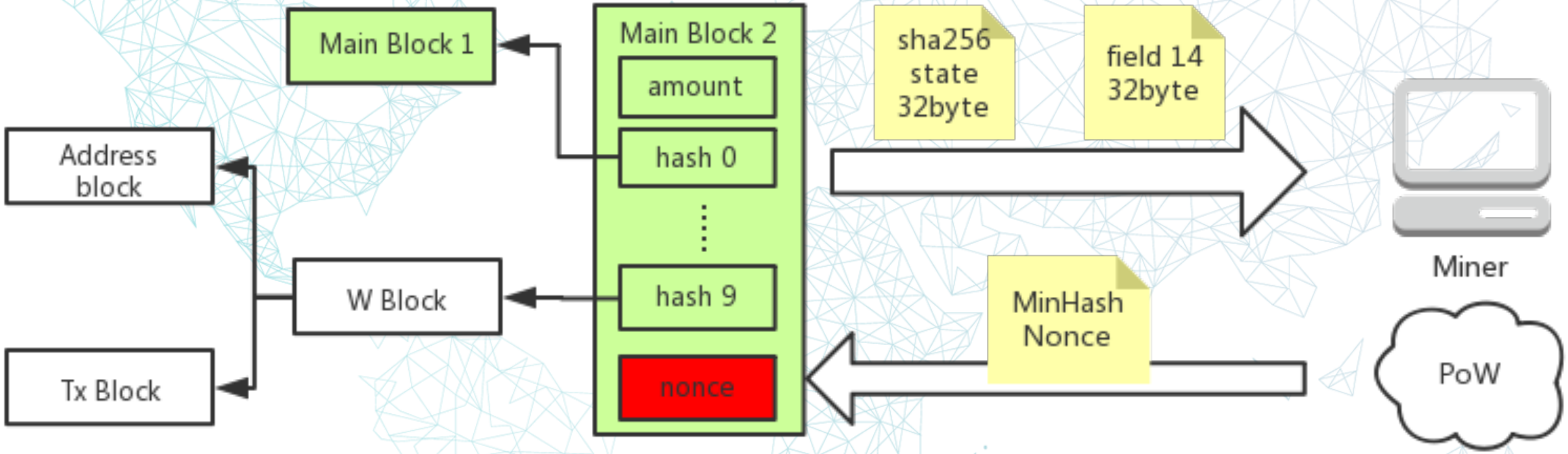
XDAG



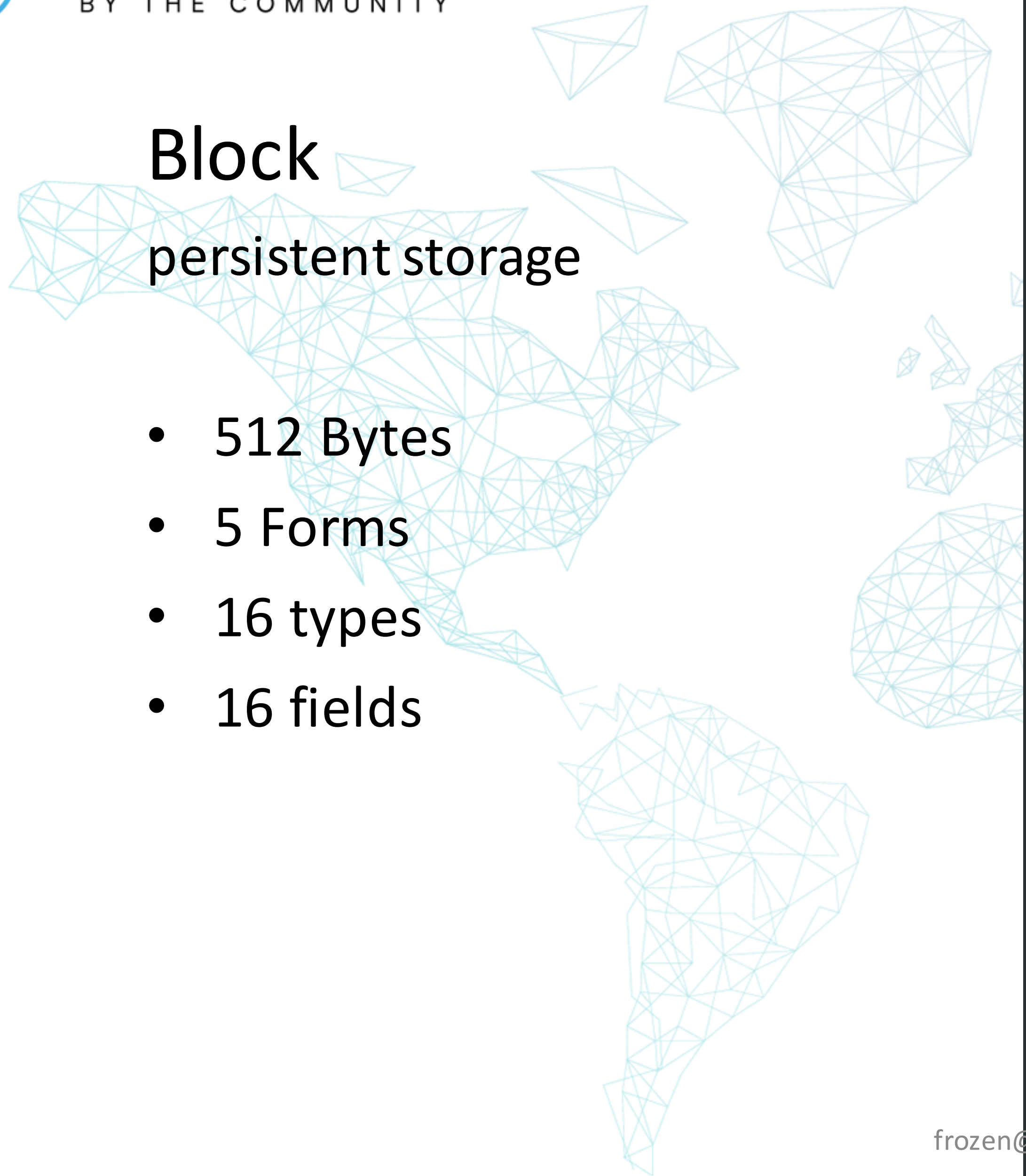
frozen@xdag.io

DAGGER
BY THE COMMUNITY

PoW: The Spirit of Blockchain



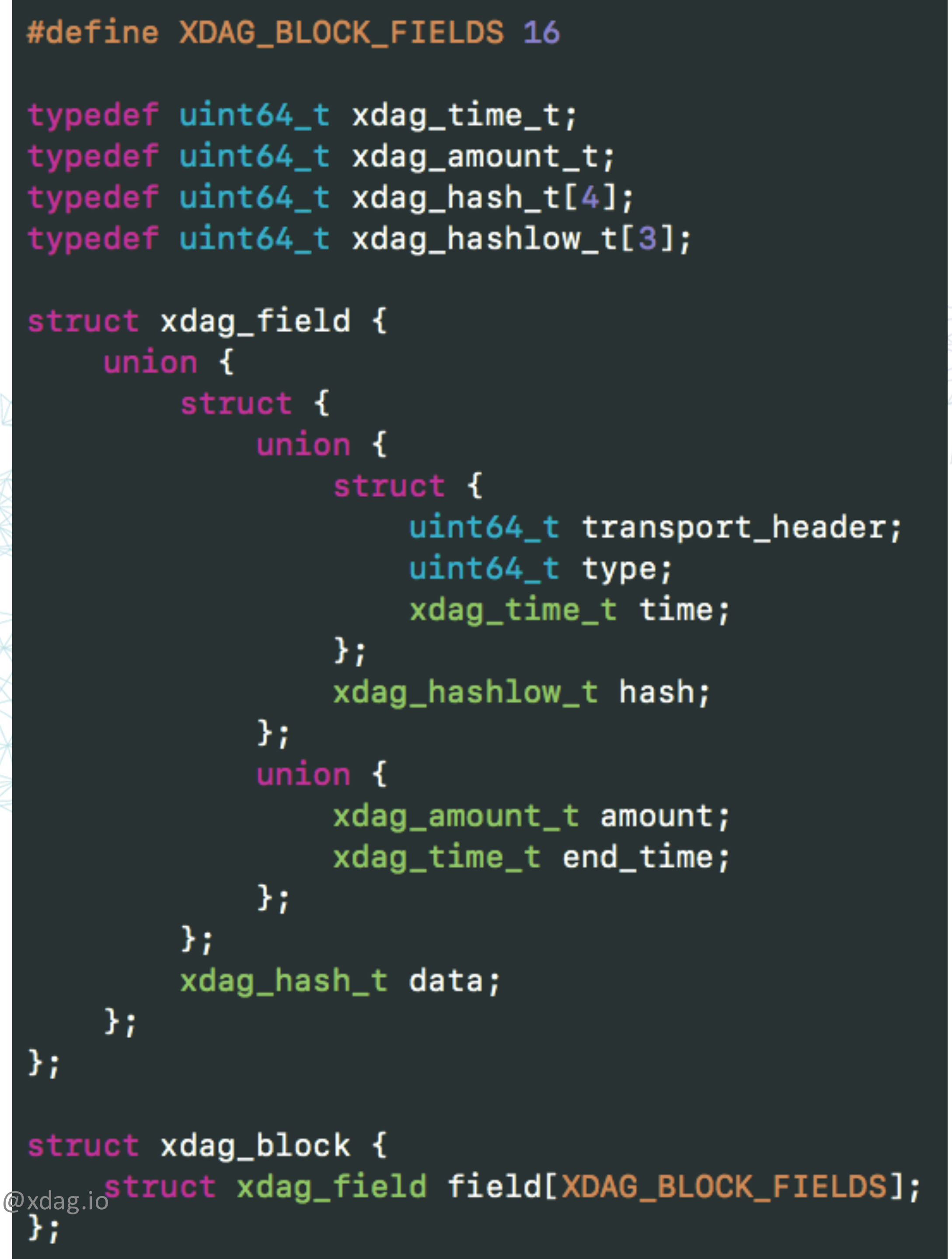
- Miner uses double sha256 to find minimal hash
- Node generates main block based on minimal hash every 64s
- Determine main chain based on difficulties of generated main blocks



Block

persistent storage

- 512 Bytes
- 5 Forms
- 16 types
- 16 fields



```c
#define XDAG_BLOCK_FIELDS 16

typedef uint64_t xdag_time_t;
typedef uint64_t xdag_amount_t;
typedef uint64_t xdag_hash_t[4];
typedef uint64_t xdag_hashlow_t[3];

struct xdag_field {
    union {
        struct {
            union {
                struct {
                    uint64_t transport_header;
                    uint64_t type;
                    xdag_time_t time;
                };
                xdag_hashlow_t hash;
            };
            union {
                xdag_amount_t amount;
                xdag_time_t end_time;
            };
        };
        xdag_hash_t data;
    };
};

struct xdag_block {
    struct xdag_field field[XDAG_BLOCK_FIELDS];
};
```

DAGGER
BY THE COMMUNITY

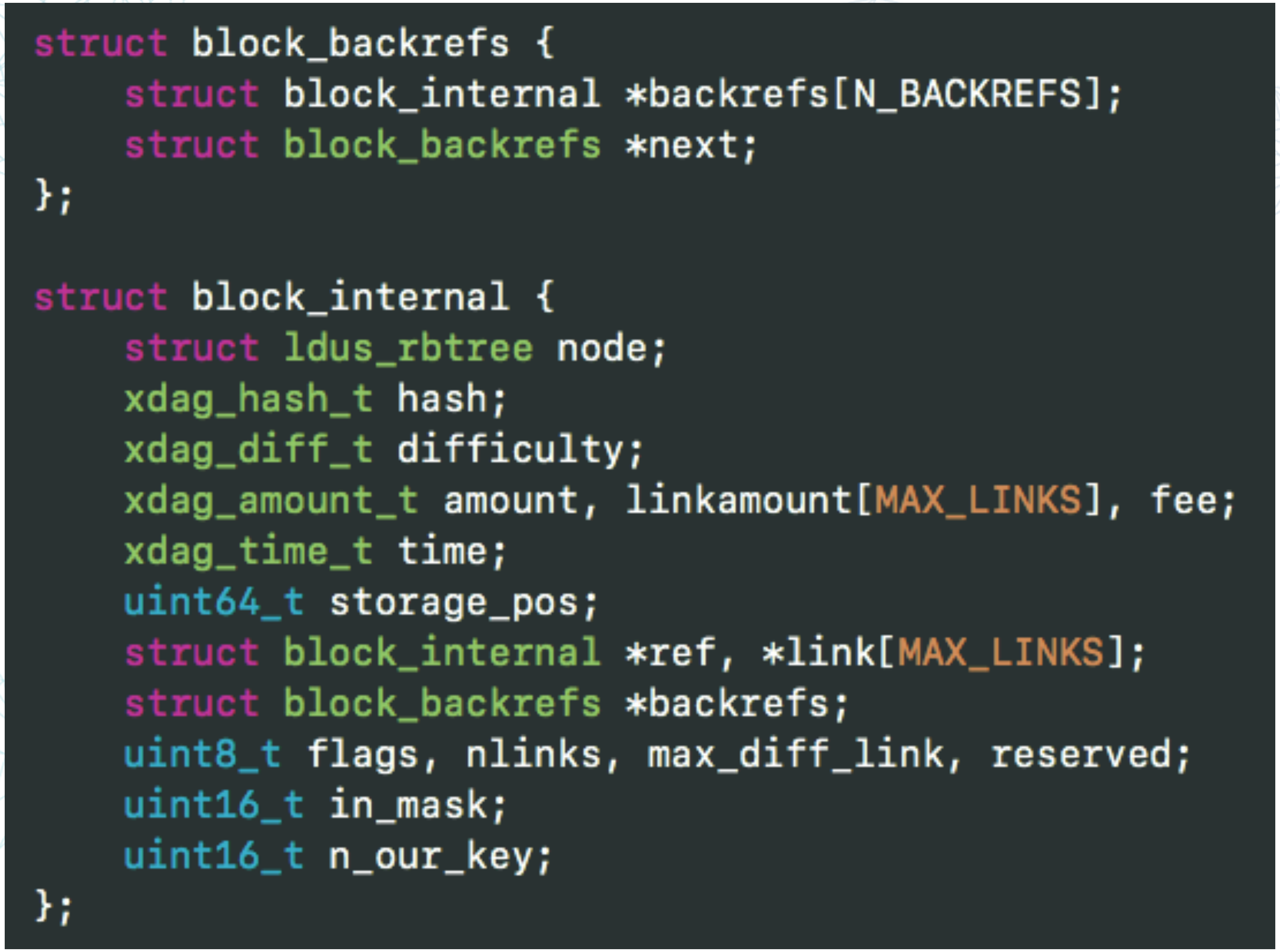
Block example

- 512 Bytes
- 1 header
- 15 fields
- Storage on disk
- max limit 10 transactions

8 Bytes	8 Bytes	8 Bytes	8 Bytes
transport header	type	time	amount
Output hash			amount
Output hash			amount
Input2 hash			amount
Input3 hash			amount
			amount
Output hash			amount
Output hash			amount
Output hash			amount
Public Key 1			
Output sign R 1			
Input sign S 1			
Public Key 2			
Input sign R 2			
Input sign S 2			

Internal Block

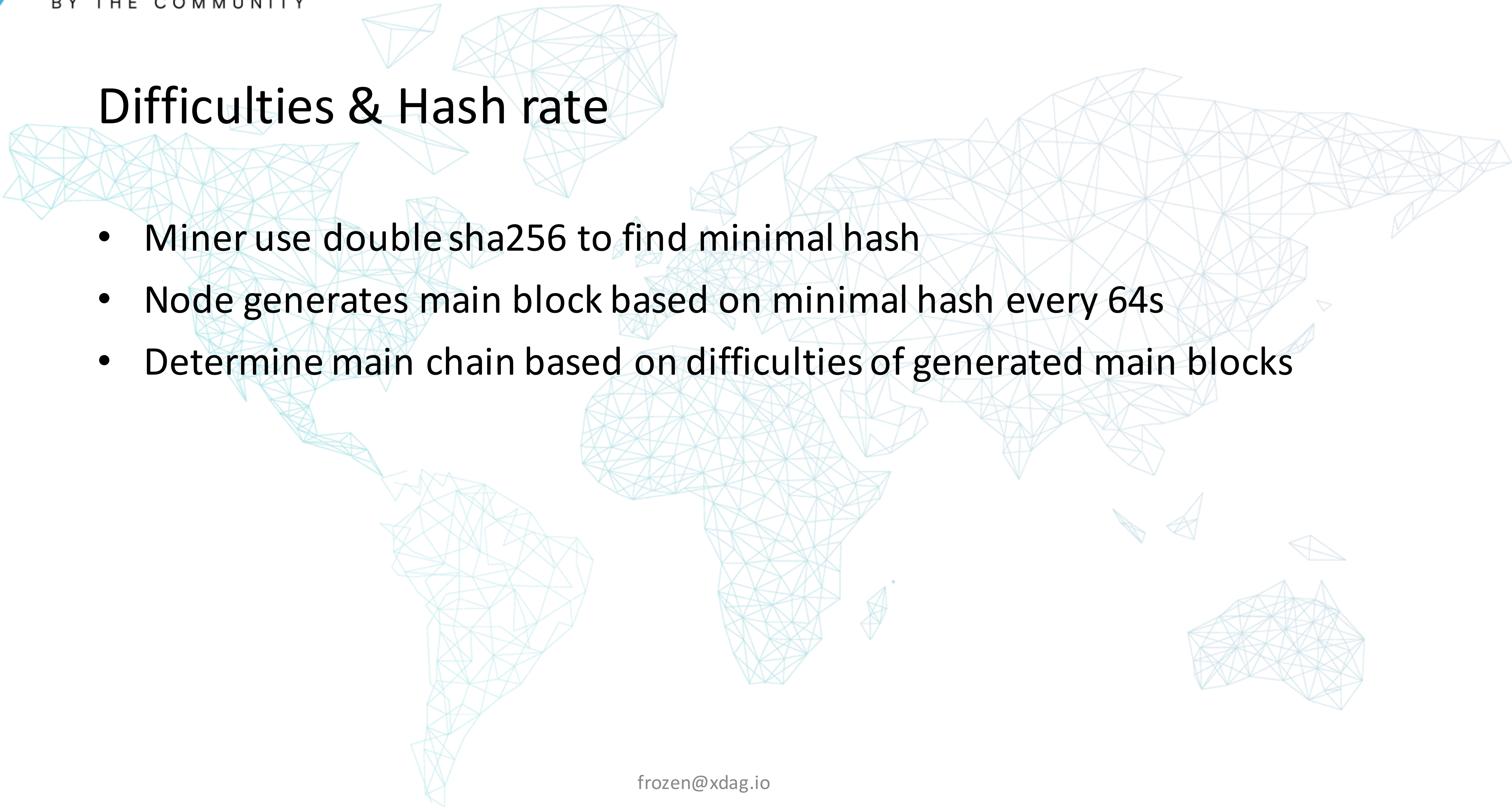
- Store DAG
- To construct account system



```
struct block_backrefs {
    struct block_internal *backrefs[N_BACKREFS];
    struct block_backrefs *next;
};

struct block_internal {
    struct ldus_rbtree node;
    xdag_hash_t hash;
    xdag_diff_t difficulty;
    xdag_amount_t amount, linkamount[MAX_LINKS], fee;
    xdag_time_t time;
    uint64_t storage_pos;
    struct block_internal *ref, *link[MAX_LINKS];
    struct block_backrefs *backrefs;
    uint8_t flags, nlinks, max_diff_link, reserved;
    uint16_t in_mask;
    uint16_t n_our_key;
};
```

Difficulties & Hash rate

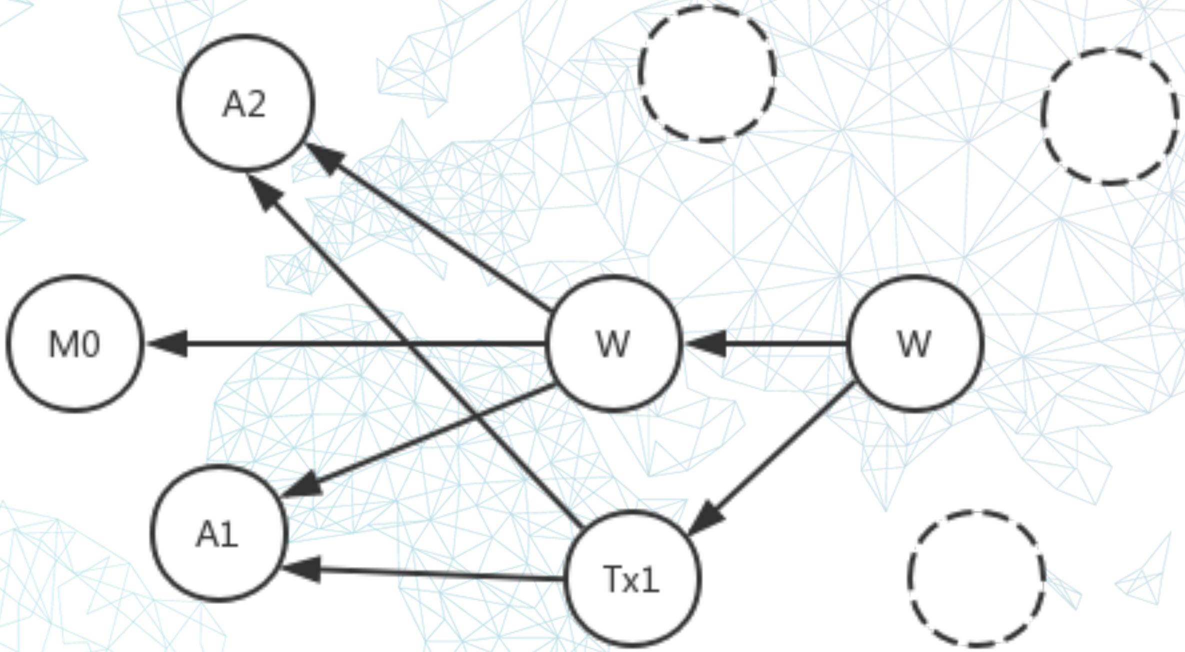


- Miner use double sha256 to find minimal hash
- Node generates main block based on minimal hash every 64s
- Determine main chain based on difficulties of generated main blocks

DAGGER
BY THE COMMUNITY

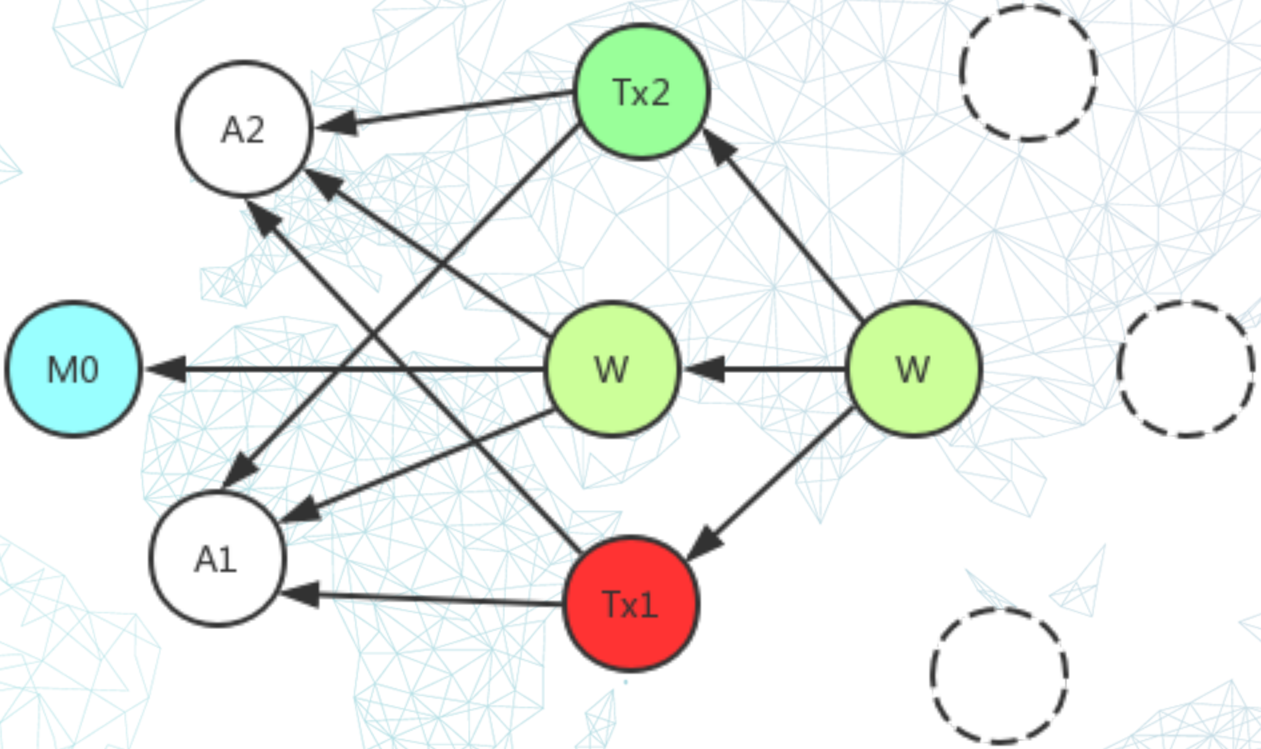
Simple Transaction case

- A1 A2 address
- M0 main block
- Tx1 transaction
- W witness block



Double spend 1

- double spend case on same node
- A1, A2 are wallet block
- Tx1, Tx2 are Txs from A1 to A2.
- Detected by ref orders directly.

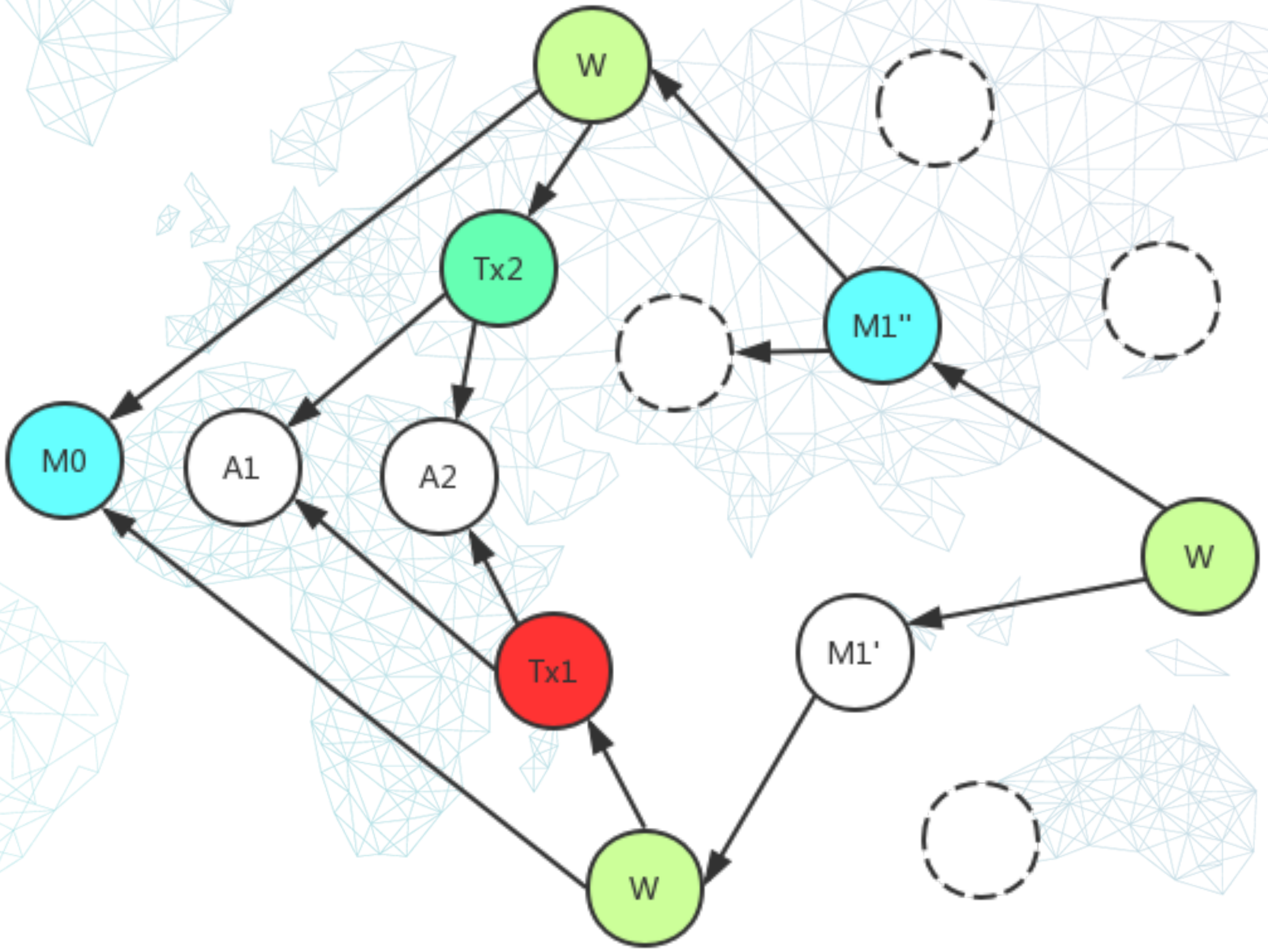


DAGGER

BY THE COMMUNITY

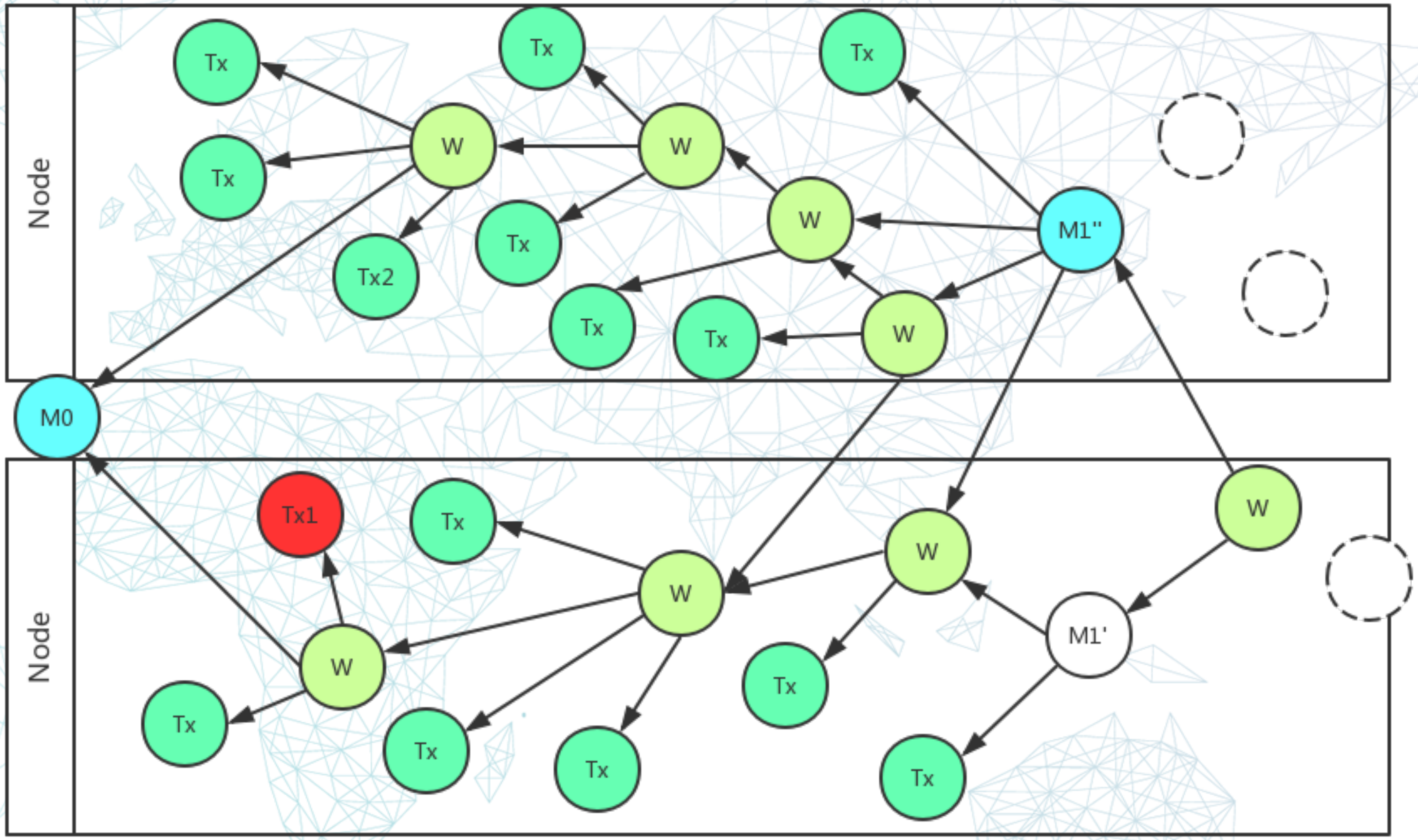
Double spend 1

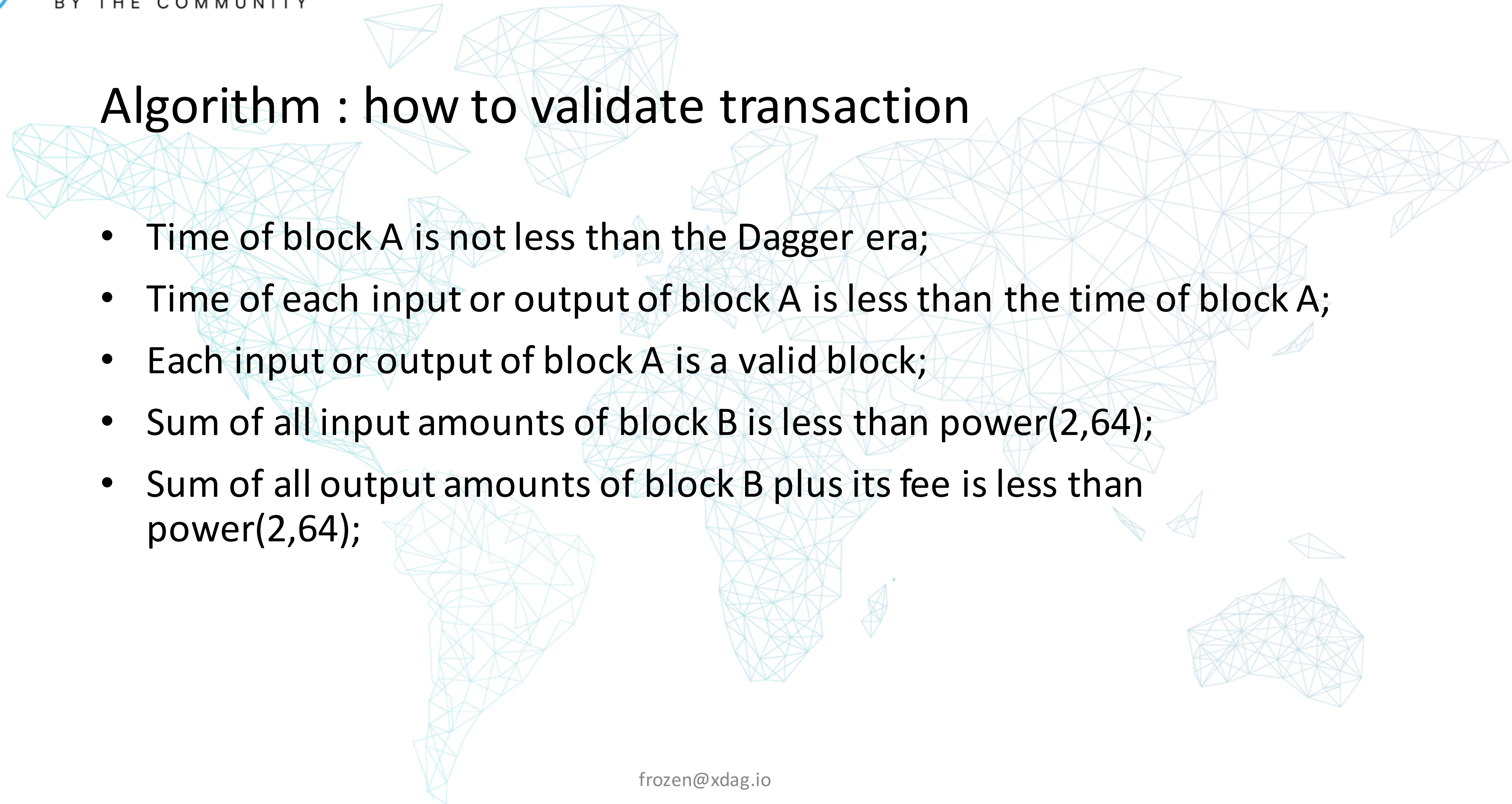
- double spend case on different nodes
- A1, A2 are wallet block
- Tx1, Tx2 are Txs from A1 to A2.
- Detected by ref orders and block difficulties.



High TPS

- each node process its transactions and construct self sub-DAG. Then merge them together through mined main blocks.





Algorithm : how to validate transaction

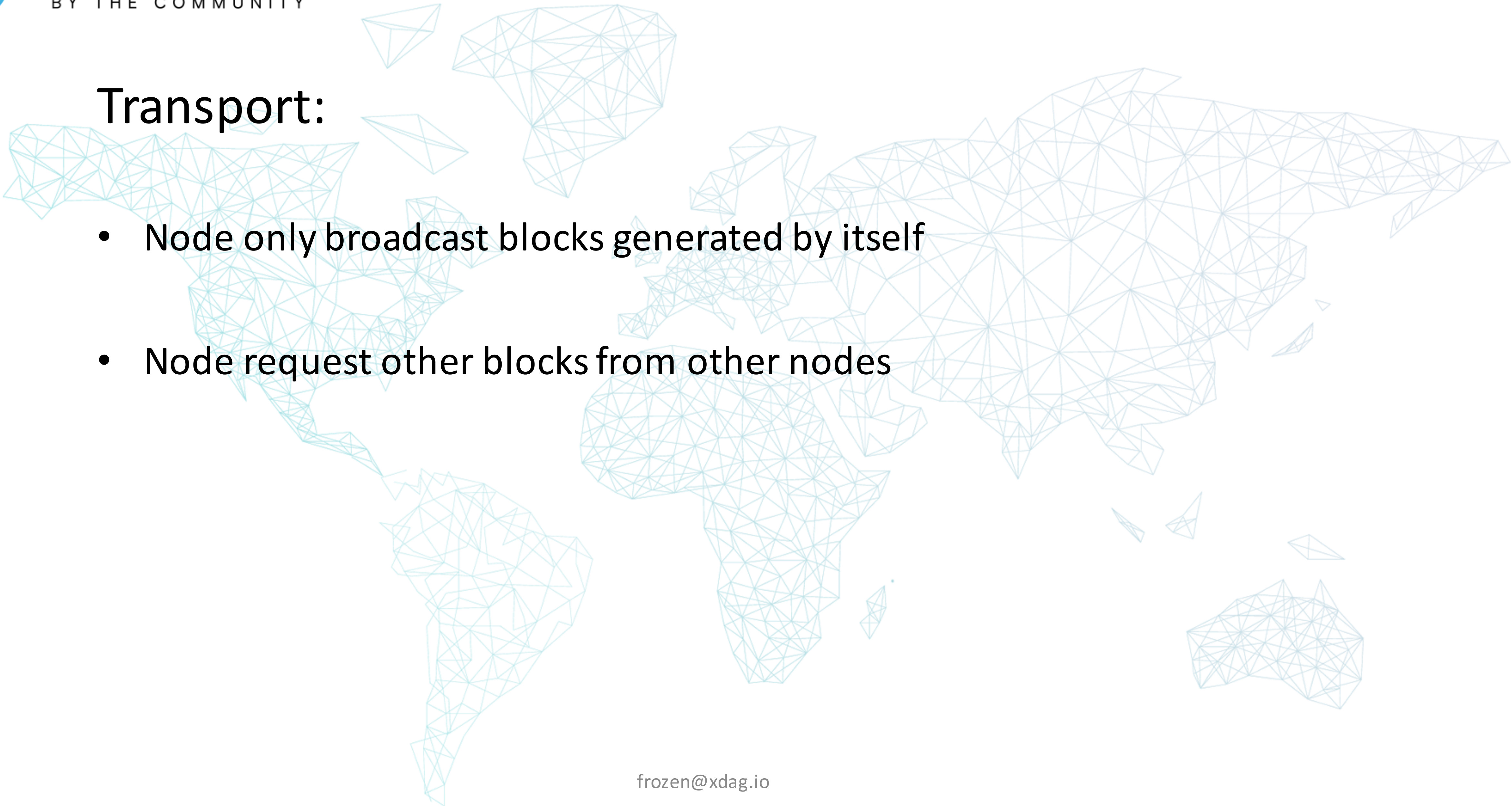
- Time of block A is not less than the Dagger era;
- Time of each input or output of block A is less than the time of block A;
- Each input or output of block A is a valid block;
- Sum of all input amounts of block B is less than power(2,64);
- Sum of all output amounts of block B plus its fee is less than power(2,64);

Algorithm : how to validate transaction

- If there is at least one input and sum of all inputs must be not less than sum of all outputs plus fee; otherwise sum of all outputs must be zero;
- For each input B of the block A there are public key K and input or output signature S in the block A and output signature T in the block B such that signature S is obtained from block A using key K and signature T is obtained from block B using the same key K (informal description: only owner of block B can withdraw money from it).
- Number of output signature fields must be even instead of number of input signature fields may be odd; in this case the last input signature field may be used as nonce which can be altered without rebuilding any signatures.

Algorithm : how to sort transactions

- Block referenced by a main block is ahead of block not referenced
- The smaller i-referenced block to the same common block is ahead
- The referenced block is ahead of linked block



Transport:

- Node only broadcast blocks generated by itself
- Node request other blocks from other nodes

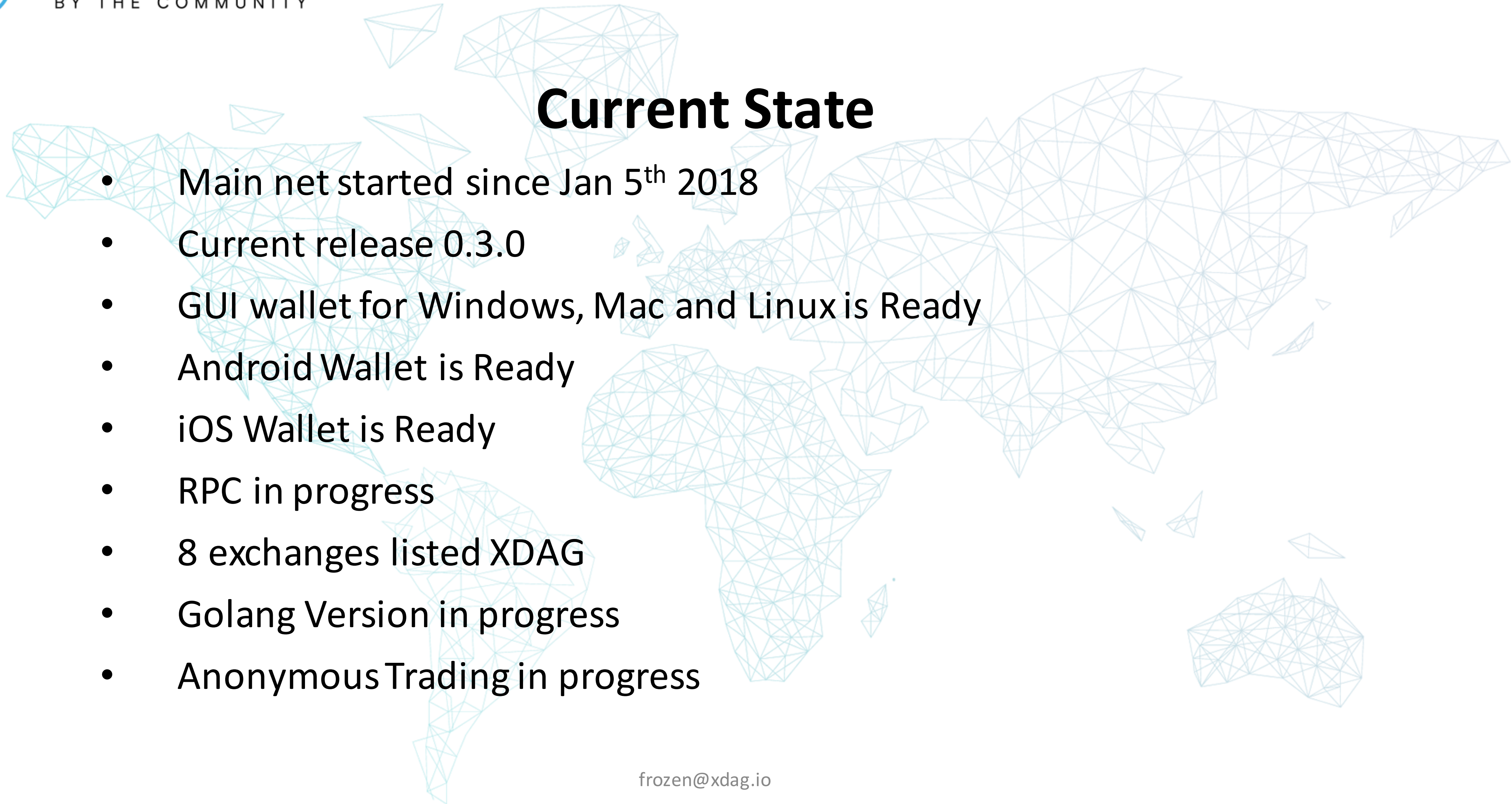
Security

- ECDSA secp256k1 for signing

- Semi-symmetric for transport

Community Driven

- No ICO, No Pre-mining, No investment, No capitals
- Community members are from different social channels
- Current developers are from different countries who never met each other



Current State

- Main net started since Jan 5th 2018
- Current release 0.3.0
- GUI wallet for Windows, Mac and Linux is Ready
- Android Wallet is Ready
- iOS Wallet is Ready
- RPC in progress
- 8 exchanges listed XDAG
- Golang Version in progress
- Anonymous Trading in progress

The Future

- **Hash algorithm adjustment** – Q1 2019
- **Anonymous Trading** – Q2 2019
- **Smart Contract** – Q3 2019

- Full Wallet – Q1 2019

How to Join & Help Community

Everyone related to XDAG is part of community

- Spread XDAG
- Discuss proposal
- Report issues
- Translation
- Contribute code

Thank you!

Thanks to all developers!

Evgeniy, Frozen, sgaragagghu, AnythingTechPro

Bill, Solar, ssyijiu, trueserve, Toneyisnow, czslience, kbs1

rubencm, Jimmy, mathsw, Wendy

Thanks to all Miners, Pool Owners, Community Members
and
other Contributors

frozen@xdag.io

DAGGER
BY THE COMMUNITY

Dev QQ Group

WeChat Official Account

frozen@xdag.io