Blockchain Principles and Applications

Amir Mahdi Sadeghzadeh, PhD

Data and Network Security Lab (DNSL)
Trustworthy and Secure AI Lab (TSAIL)

Most slides have been adapted from ECE/COS 470, Pramod Viswanath, Princeton 2024 and CS251, Dan Boneh, Stanford 2023.

Recap

Networking Requirements

No centralized server (single point of failure, censorship)

Key Primitive

Broadcast blocks and transactions to all nodes

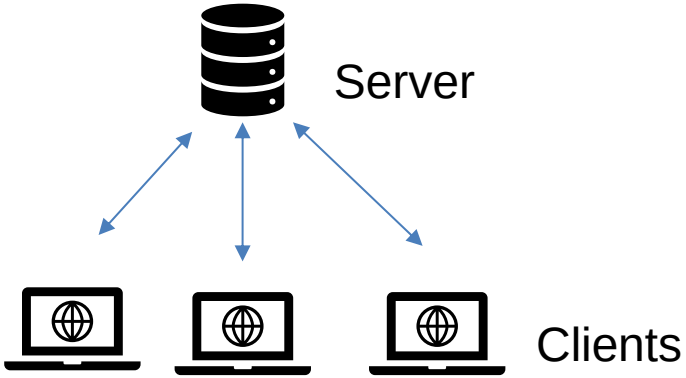
Robustness

some nodes go offline

new nodes join

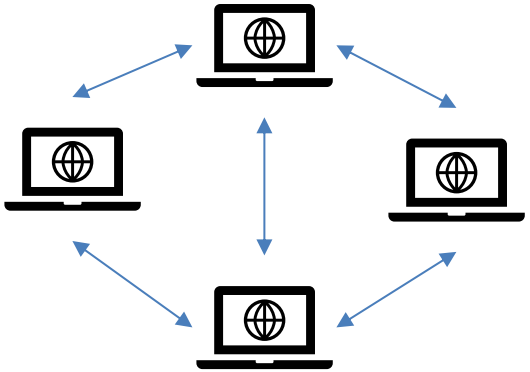
Types of Network Architecture

Client server



Server stores most of the data

Peer to Peer

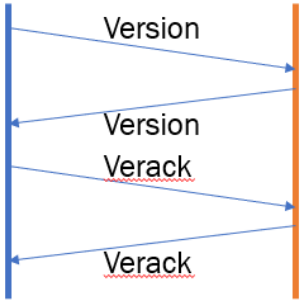


Each node acts as a client and a server

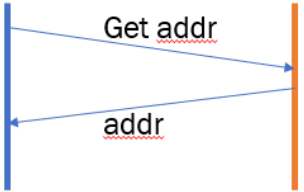
BitTorrent, Napster

Peer discovery

- DNS seed nodes (Hard coded in the codebase)
- Easy to be compromised, do not trust one seed node exclusively
- Hardcoded peers (fallback)
- Ask connected peers for additional peers



Connecting to a peer



Gathering additional peers
Addr: contains list of up to 1000 nodes

Bitcoin network

- TCP connection with peers
- At most 8 outbound TCP connections
- May accept up to 117 inbound TCP connections
- Maintains a large list of nodes (IP, port) on the bitcoin network
- Establishes connection to a subset of the stored nodes

Peer-2-Peer Networking (Continue)

Efficient Networking

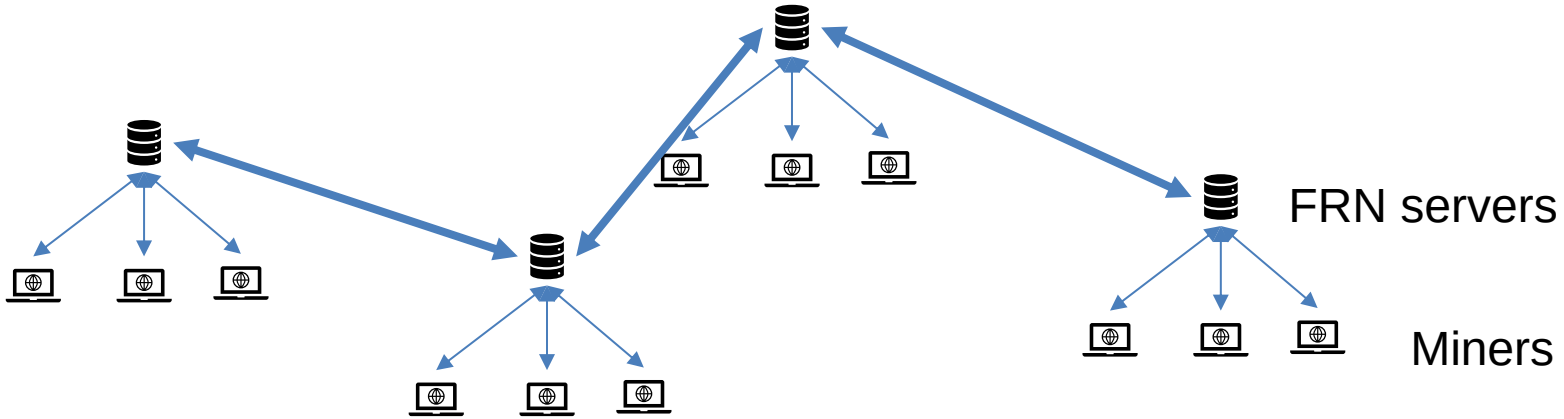
- **Trusted networks**

- Privacy
  - Can link transaction source to IP address

- Security
  - Plausible deniability for forking
  - Eclipse attacks

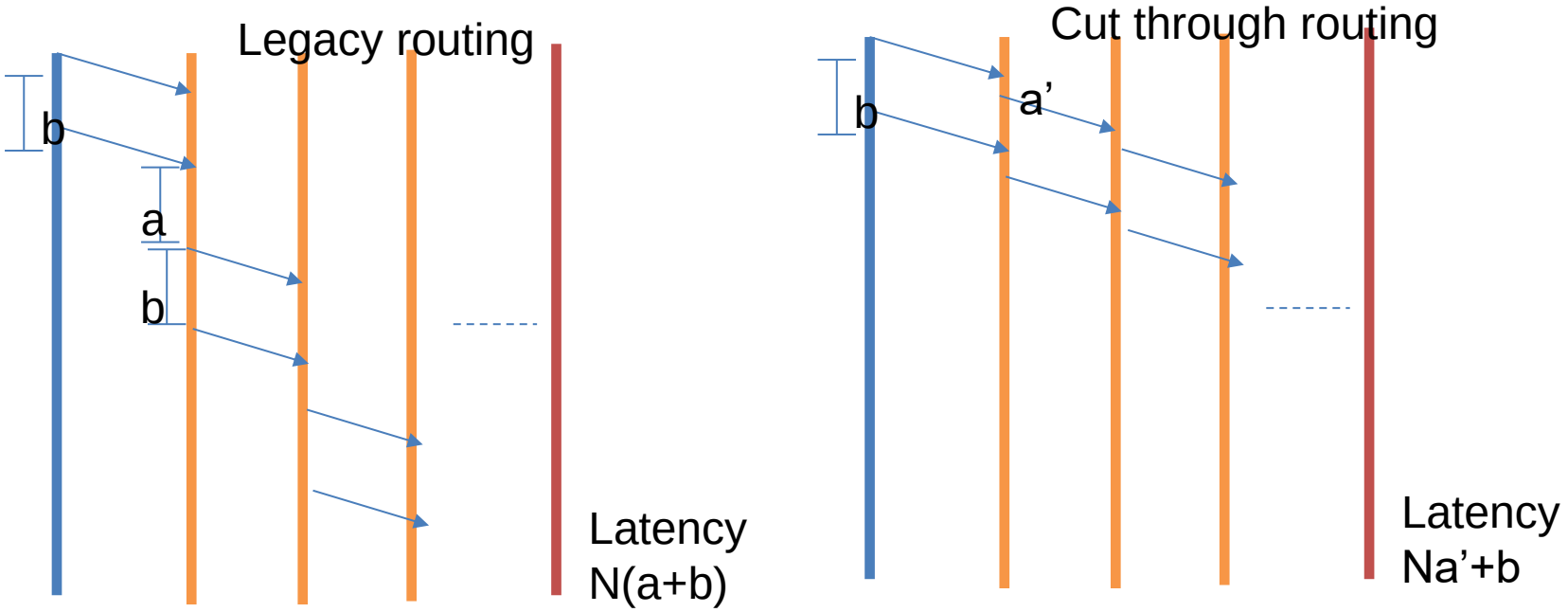
Trusted Networks

- FRN (Fast relay network): Hub and spoke model, trusted servers, servers are fast

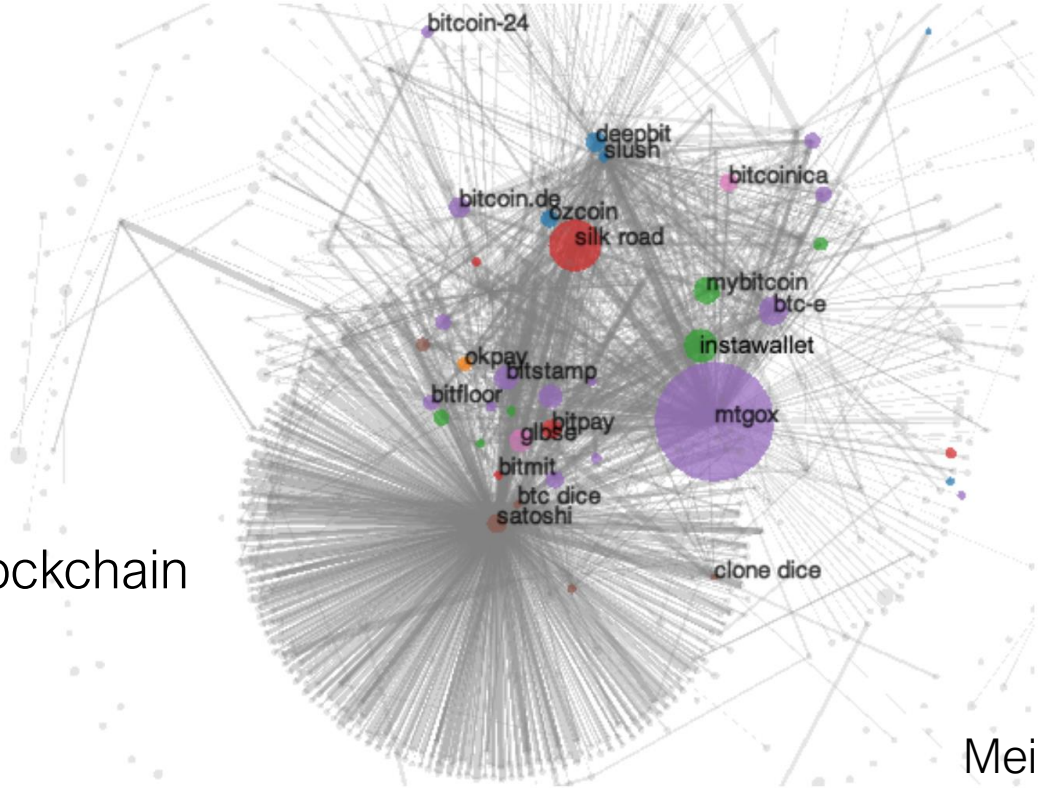


Trusted Networks: Falcon

- Cut through routing for servers, only verify headers before forwarding



How can users be deanonymized?



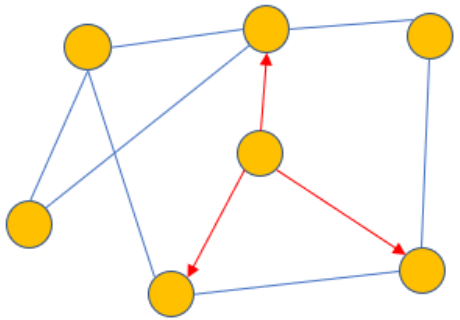
Blockchain

Meiklejohn et al., 2013

What about the peer-to-peer network?

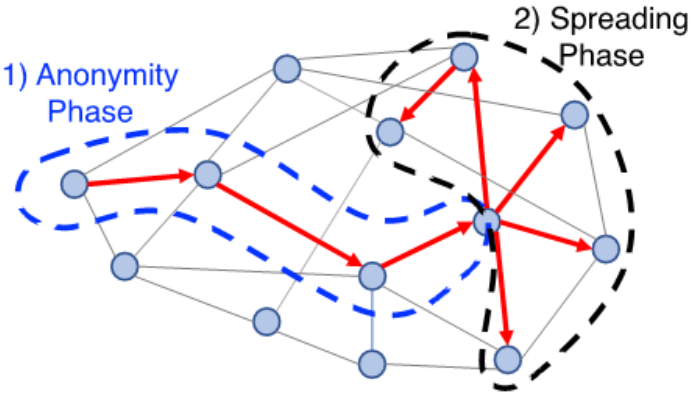
Dandelion

Deanonymization Analysis



Pr(detection)

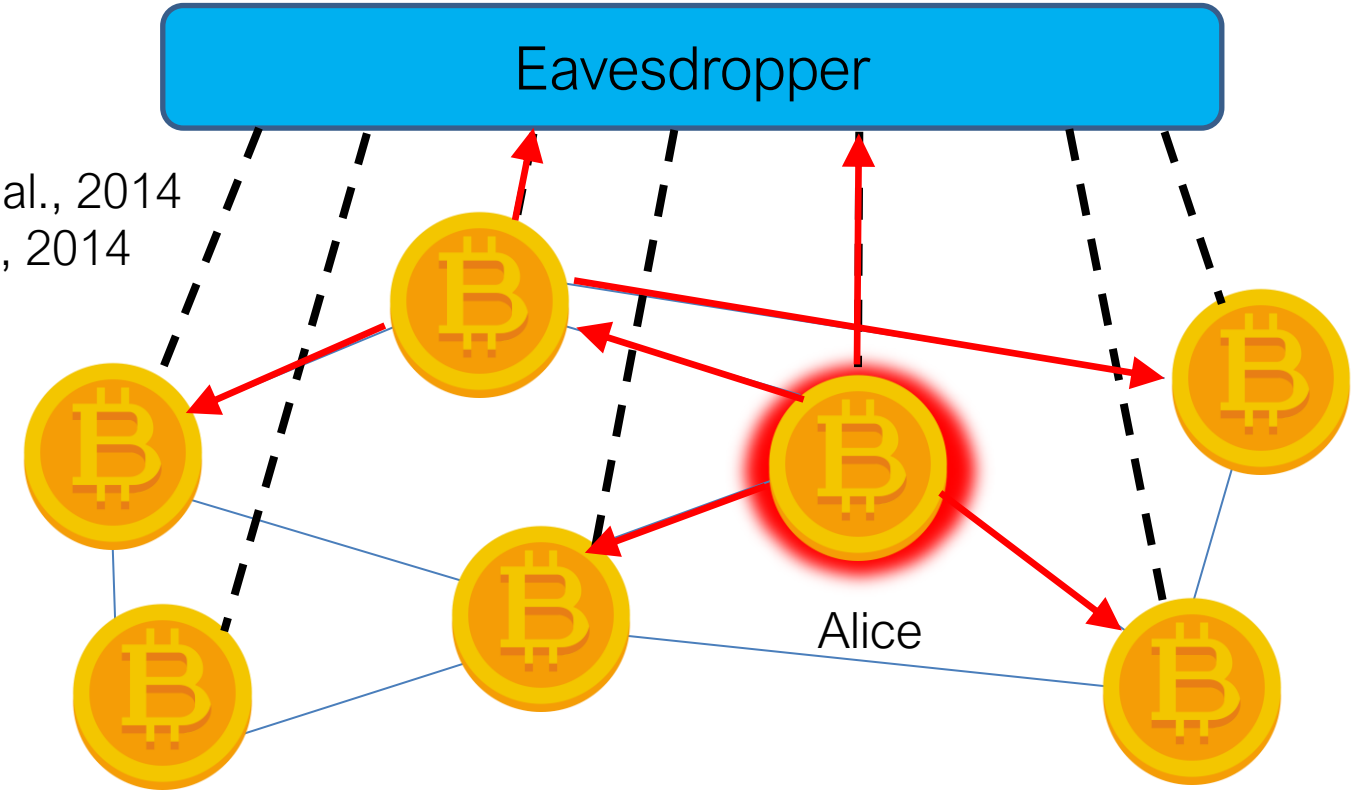
Redesign for Anonymity



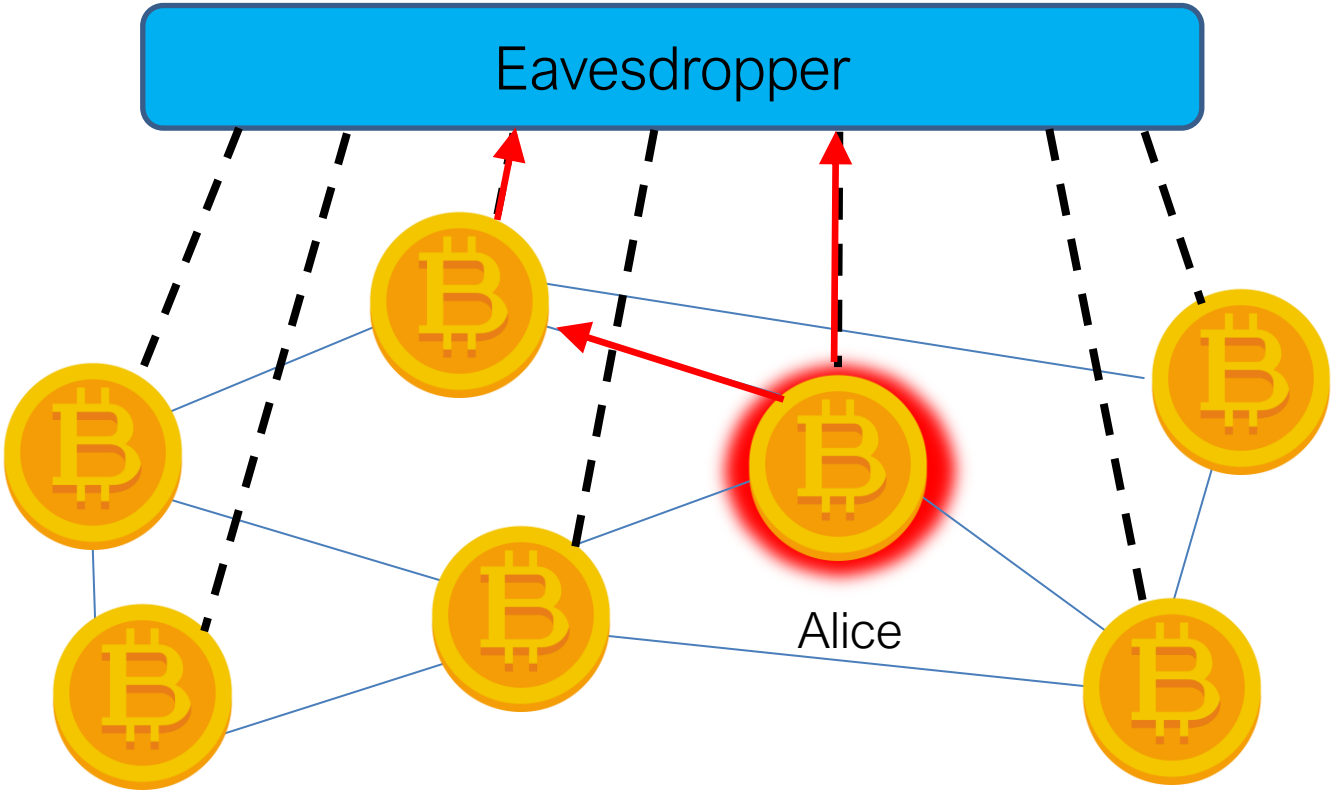
Dandelion

Attacks on the Network Layer

Biryukov et al., 2014
Koshy et al., 2014

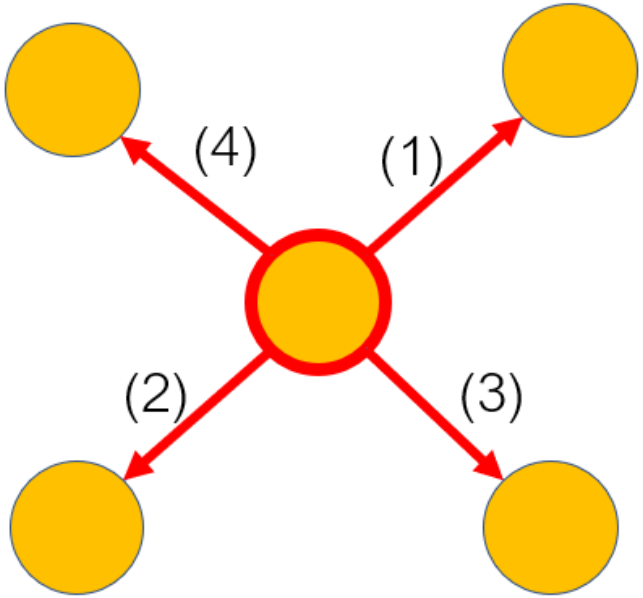


What can go wrong?

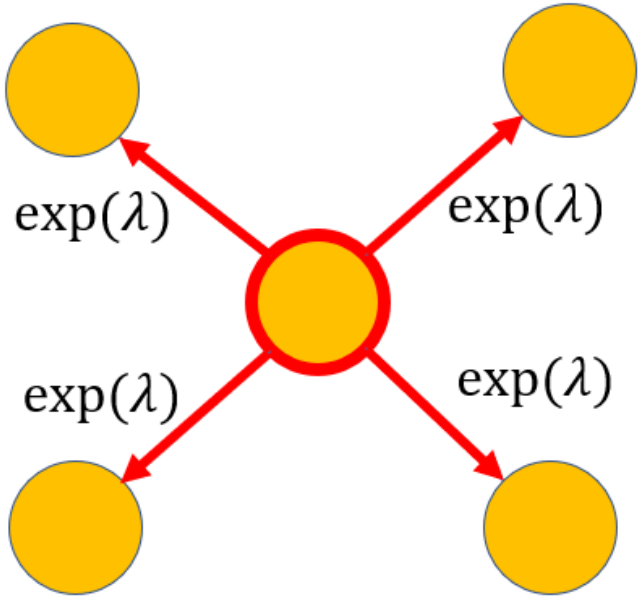


Flooding Protocols

Trickle (pre-2015)



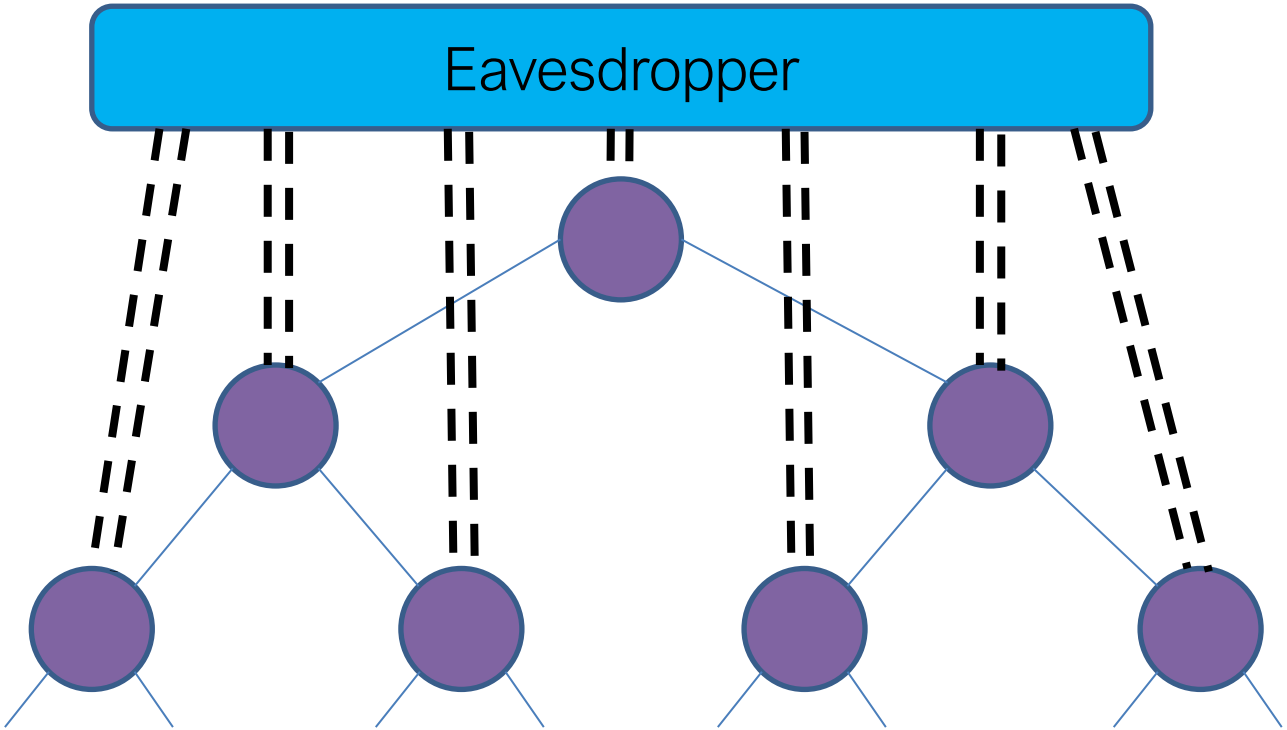
Diffusion (post-2015)



d-regular trees

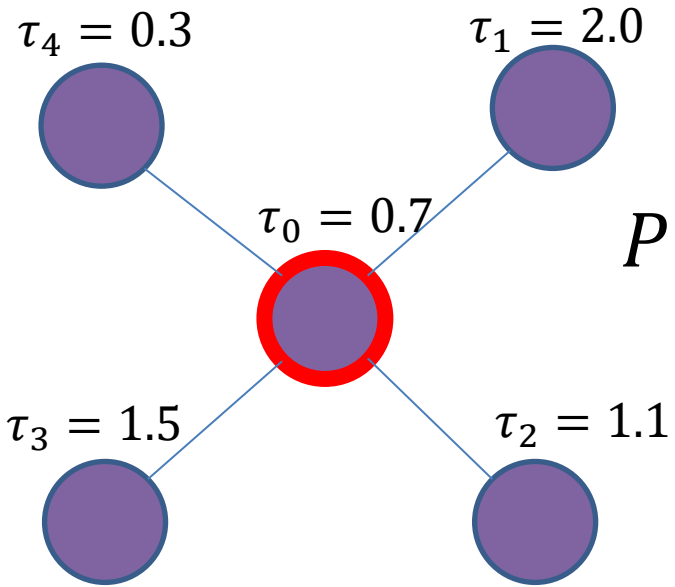
Fraction of spies $p = 0$

Arbitrary number of connections θ



Anonymity Metric

$$\boldsymbol{\tau} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \dots \\ \tau_n \end{bmatrix}$$



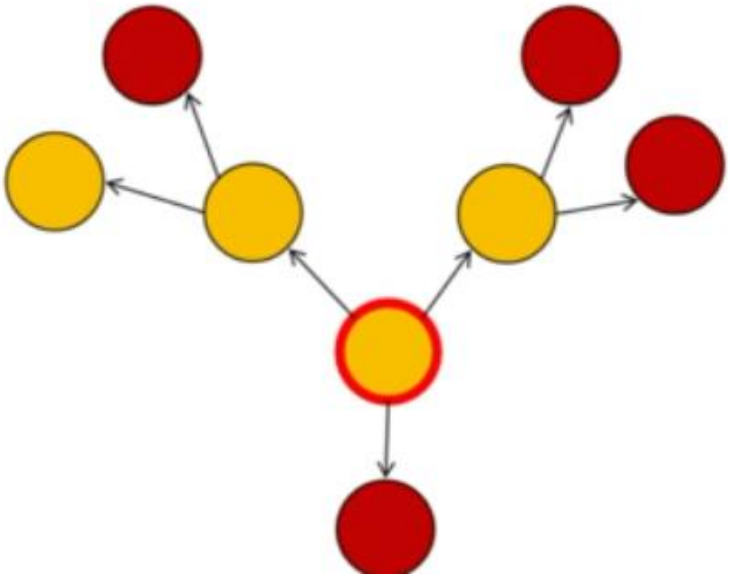
timestamps

$$P(\text{detection}|\boldsymbol{\tau}, G)$$

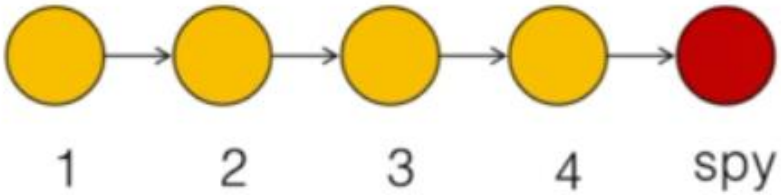
graph

What are we looking for?

Asymmetry



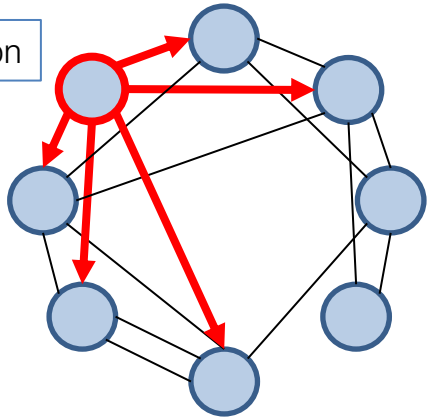
Mixing



What can we control?

Spreading Protocol

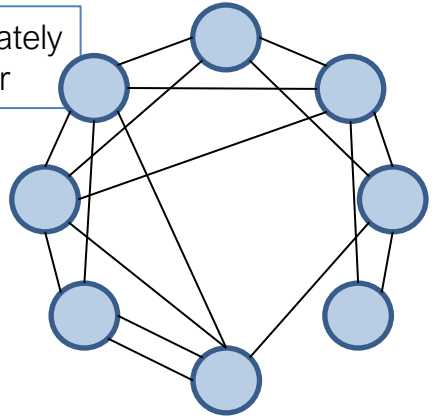
Diffusion



Given a graph, how do we spread content?

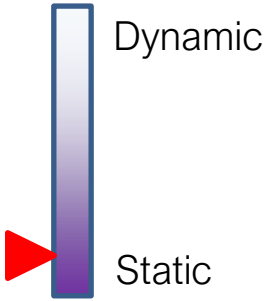
Topology

Approximately regular



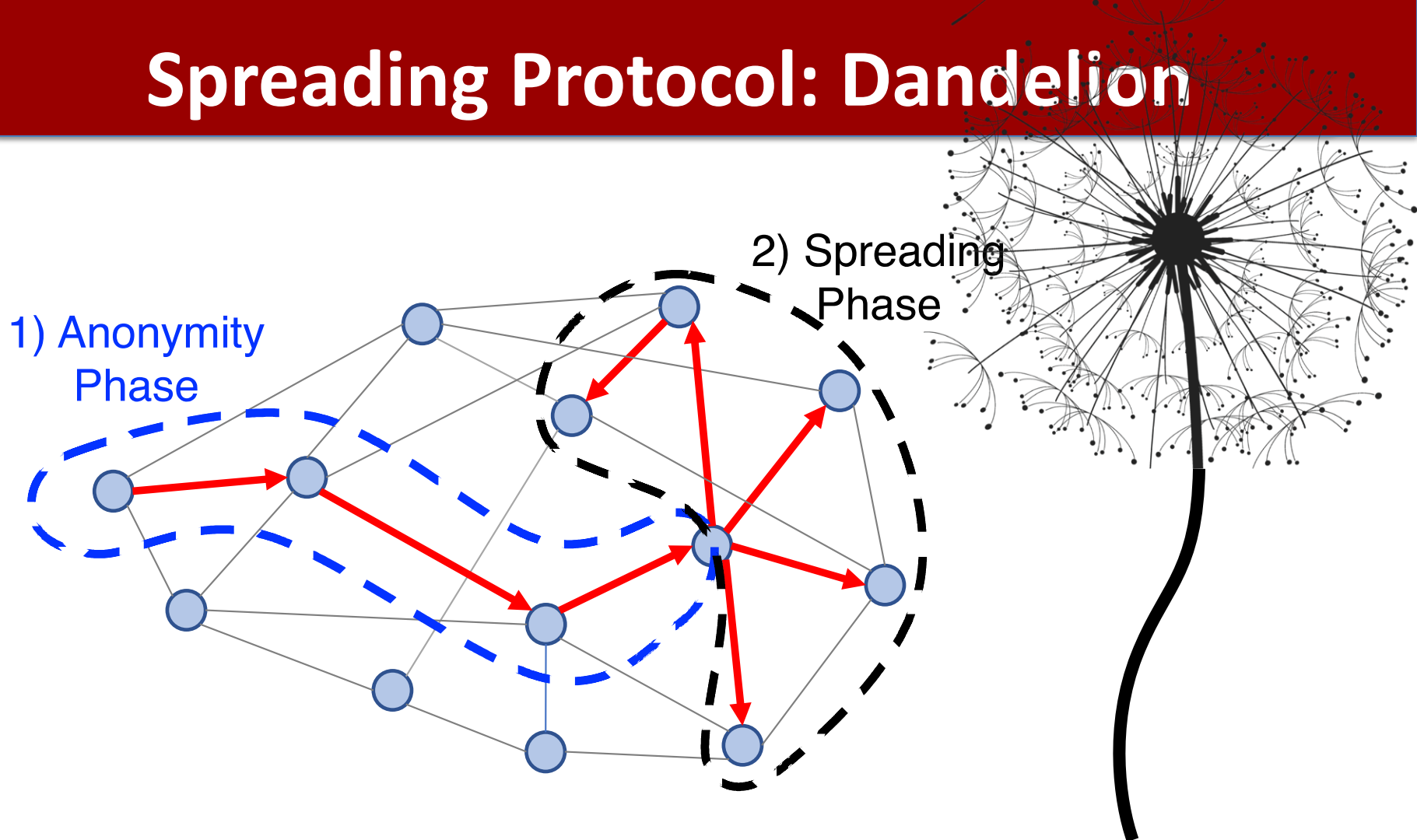
What is the underlying graph topology?

Dynamicity



How often does the graph change?

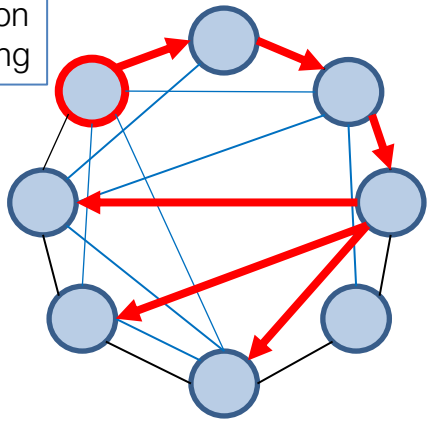
Spreading Protocol: Dandelion



Dandelion Network Policy

Spreading Protocol

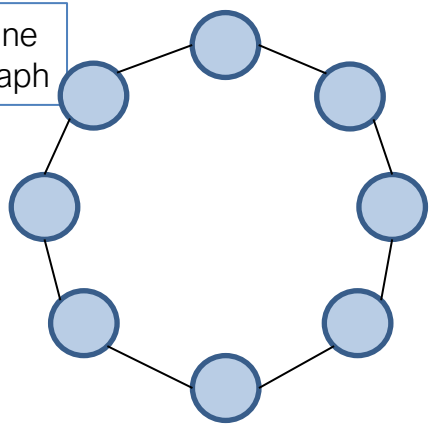
Dandelion Spreading



Given a graph, how do we spread content?

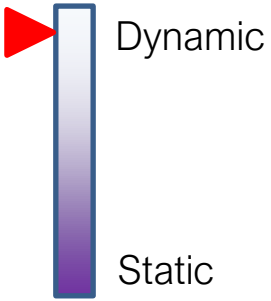
Topology

Line graph



What is the anonymity graph topology?

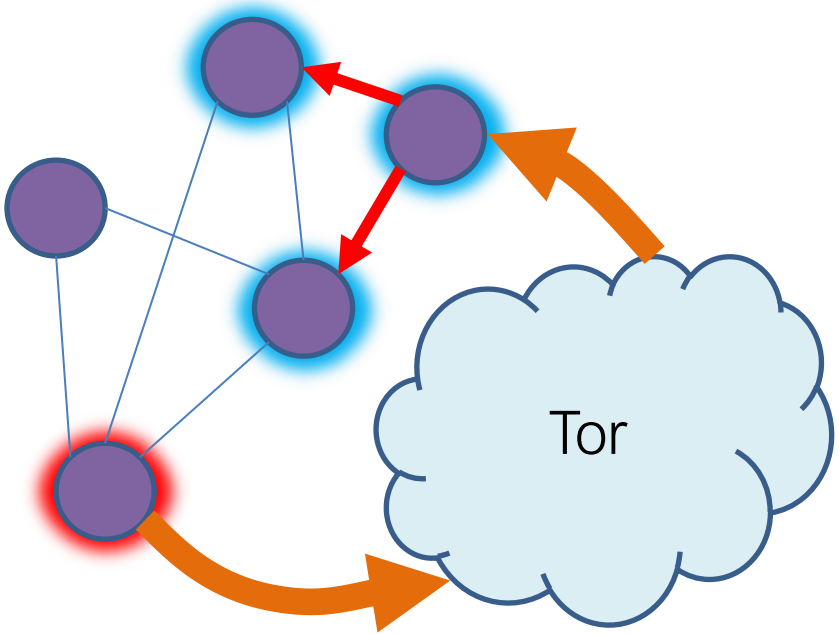
Dynamicity



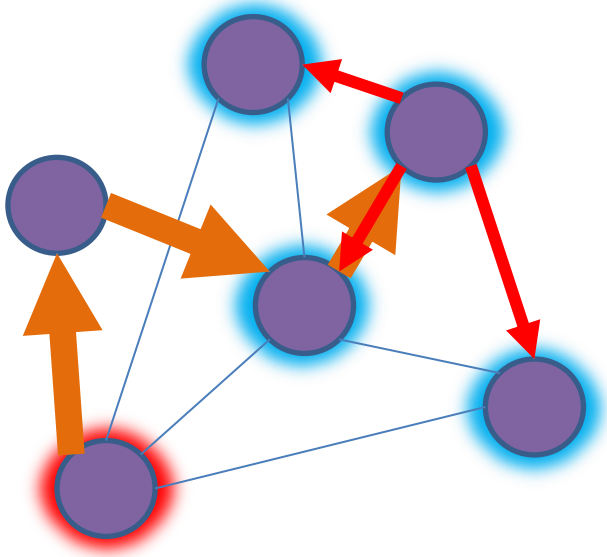
How often does the graph change?

Alternative solutions

Connect through Tor

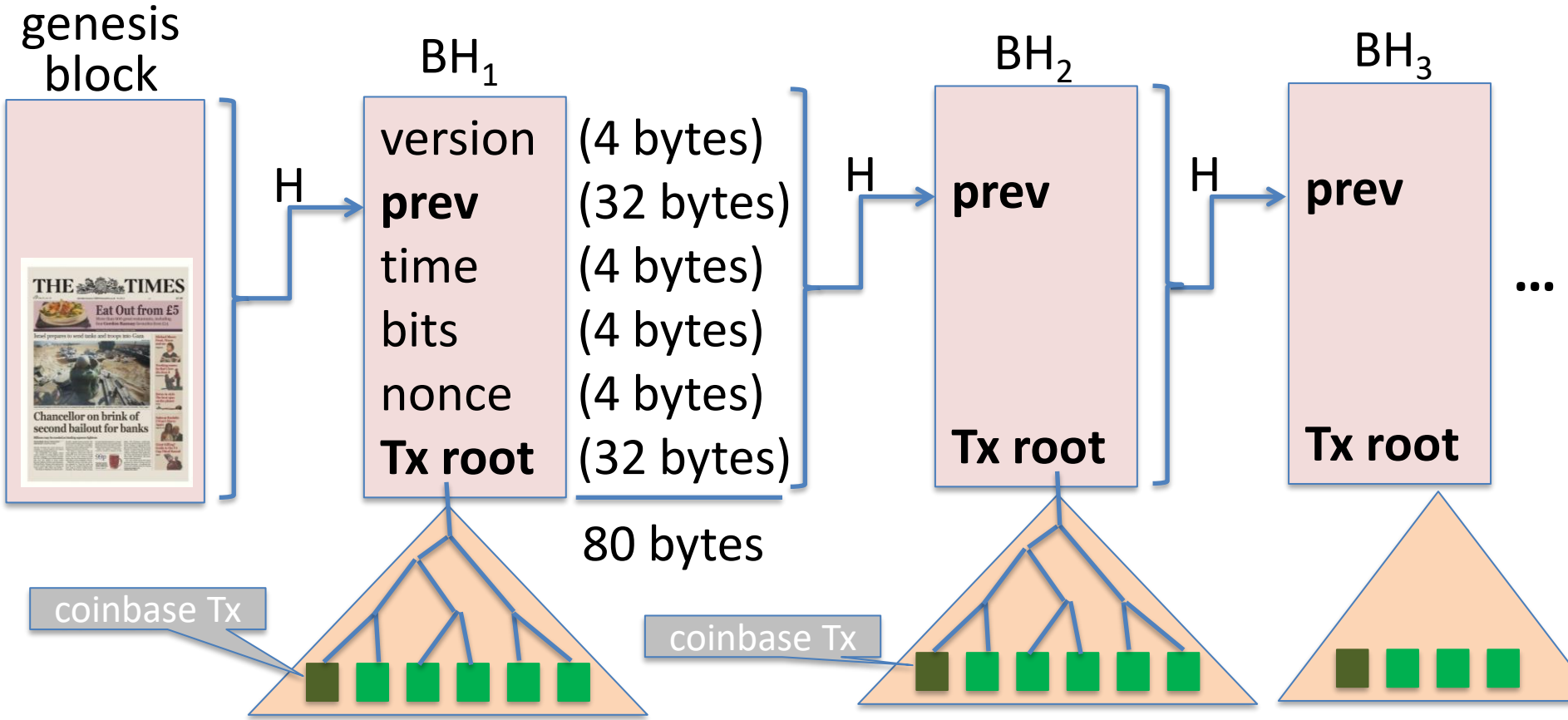


I2P Integration (e.g. Monero)



Transactions

Bitcoin blockchain: a sequence of block headers, 80 bytes each



Bitcoin blockchain: a sequence of block headers, 80 bytes each

time: time miner assembled the block. Self reported.
(block rejected if too far in past or future)

bits: proof of work difficulty
nonce: proof of work solution
} for choosing a proposer

Merkle tree: payer can give a short proof that Tx is in the block

new block every ≈10 minutes.

An example

Tx data

Height	Mined	Miner	Size	#Tx
648494	17 minutes	Unknown	1,308,663 bytes	1855
648493	20 minutes	SlushPool	1,317,436 bytes	2826
648492	59 minutes	Unknown	1,186,609 bytes	1128
648491	1 hour	Unknown	1,310,554 bytes	2774
648490	1 hour	Unknown	1,145,491 bytes	2075
648489	1 hour	Poolin	1,359,224 bytes	2622

Block 648493

Timestamp	2020-09-15 17:25
Height	648493
Miner	SlushPool
Number of Transactions	2,826
Difficulty	17,345,997,805,929.09
Merkle root	350cbb917c918774c93e945b960a2b3ac1c8d448c2e67839223bbcf595baff89
Transaction Volume	11256.14250596 BTC
Block Reward	6.25000000 BTC
Fee Reward	0.89047154 BTC

(from coinbase Tx)

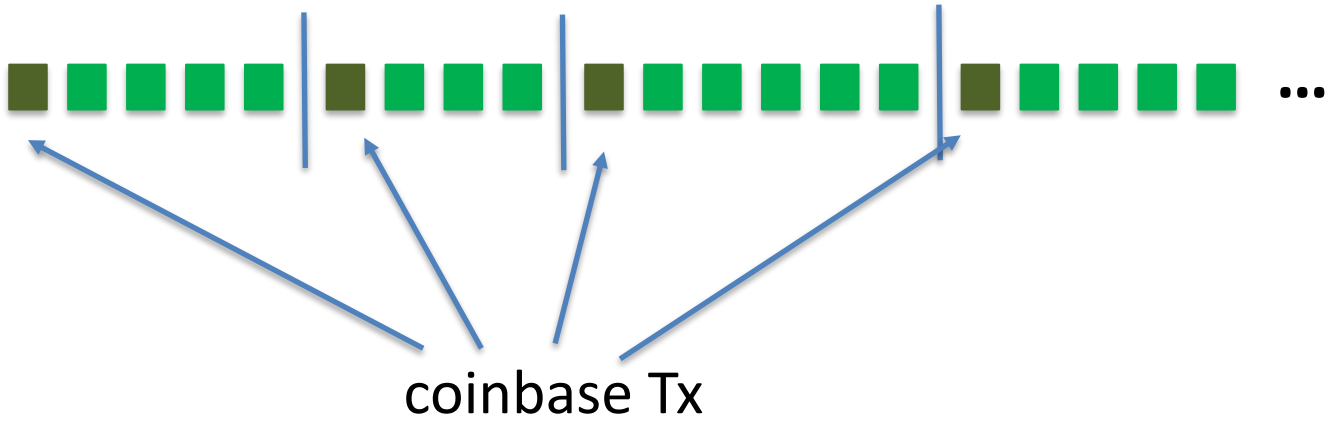
(D)

(adjusts every two weeks)

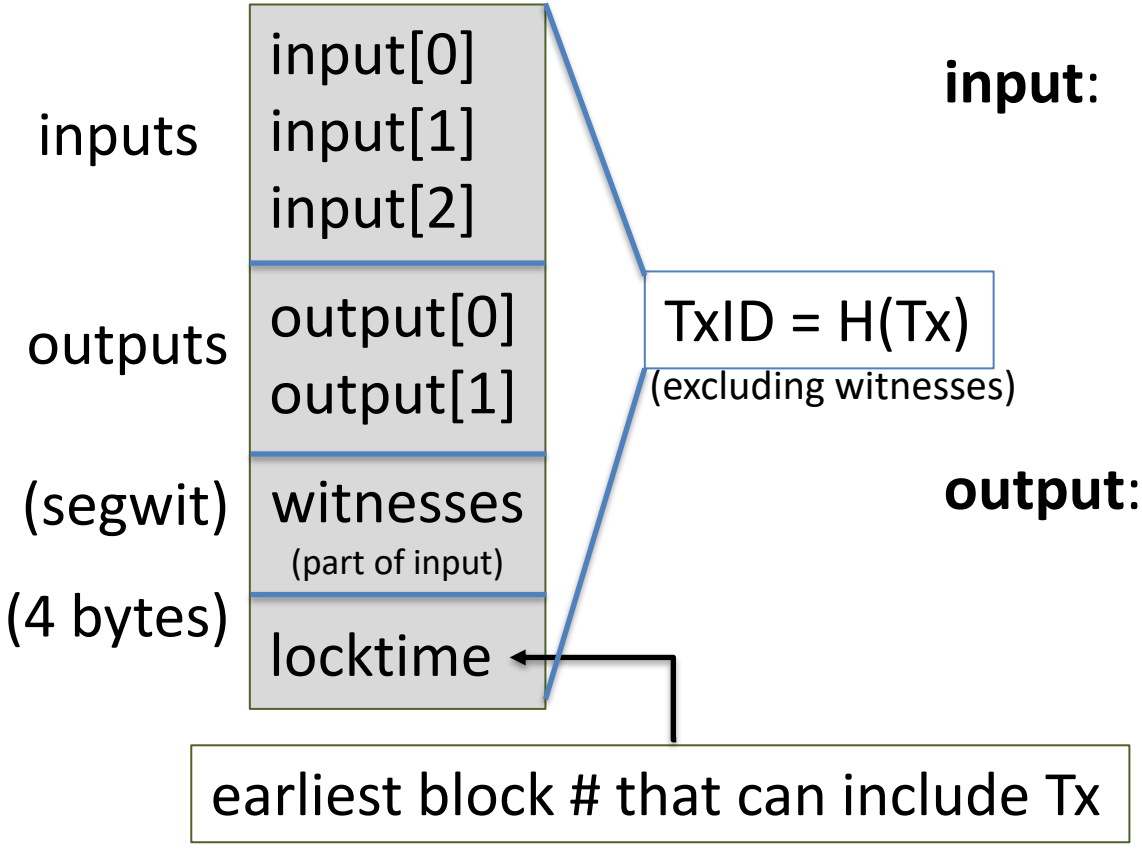
(Tx fees given to miner in coinbase Tx)

This lecture

View the blockchain as a sequence of Tx (append-only)



Tx structure (non-coinbase)



input:

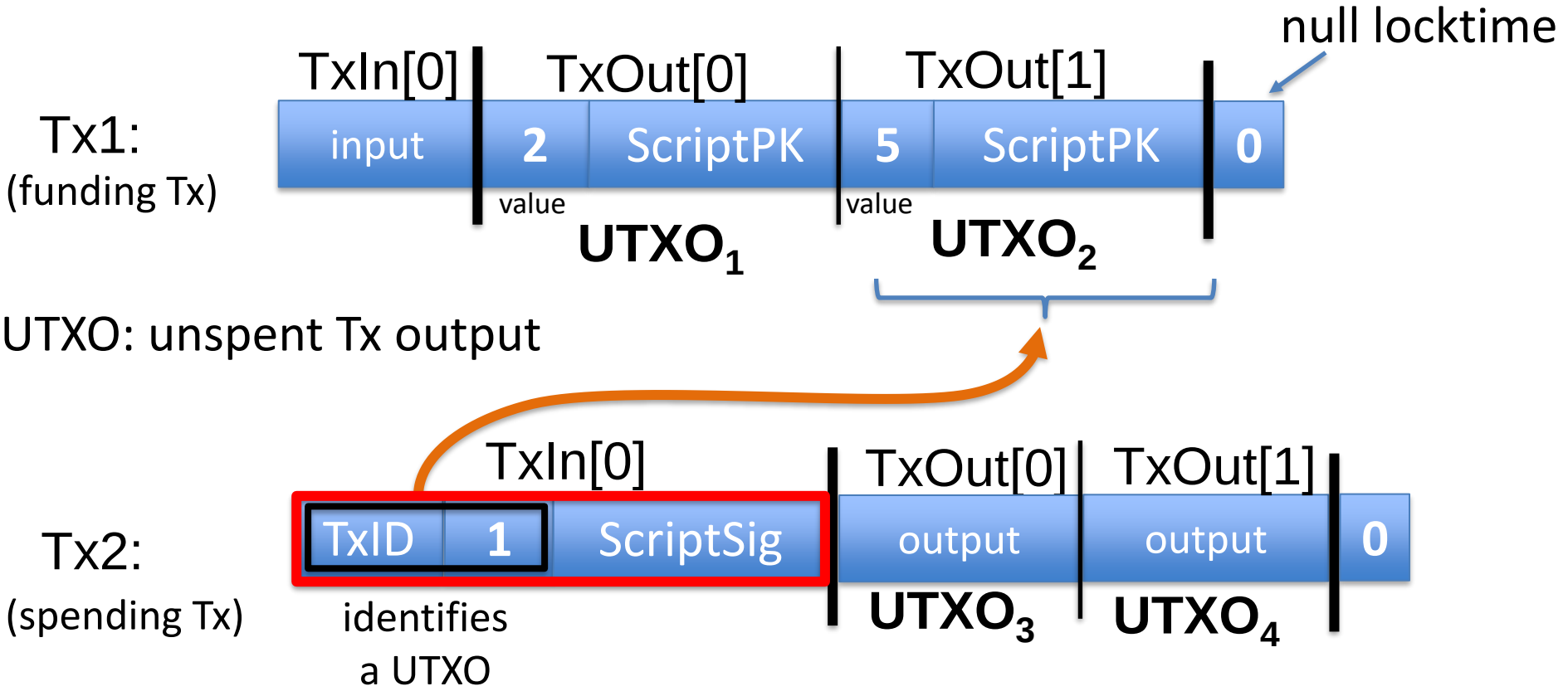
TxID	32 byte hash
out-index	4 byte index
ScriptSig	program
seq	ignore

output:

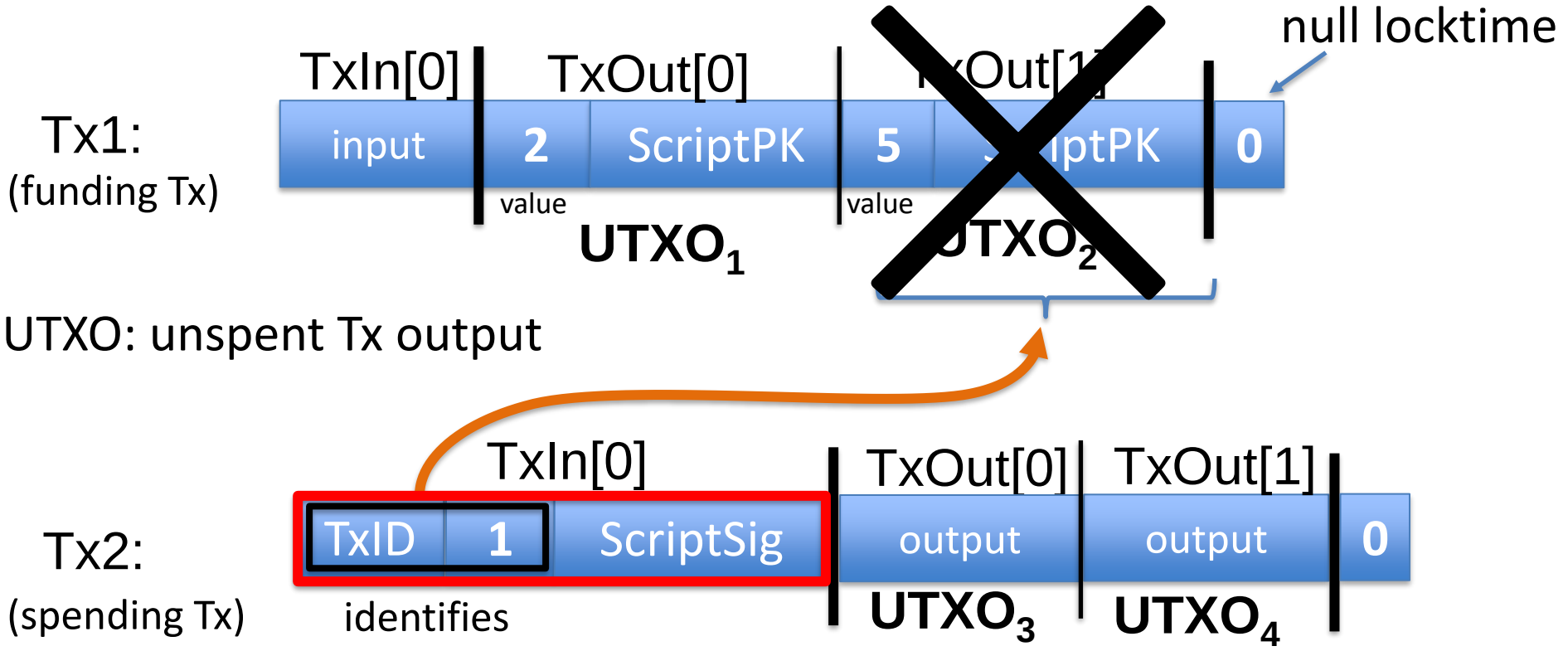
value	8 bytes
ScriptPK	program

#BTC = value/10^8

Example


Tx1:
(funding Tx)
TxIn[0]
input
TxOut[0]
2
value
ScriptPK
UTXO1
TxOut[1]
5
value
ScriptPK
UTXO2
0
null locktime
UTXO: unspent Tx output
Tx2:
(spending Tx)
TxIn[0]
TxID
1
ScriptSig
identifies
a UTXO
TxOut[0]
output
UTXO3
TxOut[1]
output
UTXO4
0

Example


Tx1:
(funding Tx)
TxIn[0]
TxOut[0]
TxOut[1]
null locktime
input
2
ScriptPK
5
ScriptPK
0
value
value
UTXO1
UTXO2
UTXO: unspent Tx output
Tx2:
(spending Tx)
TxIn[0]
TxOut[0]
TxOut[1]
TxID
1
ScriptSig
output
output
0
identifies
a UTXO
UTXO3
UTXO4

Validating Tx2

Miners check (for each input):

program from funding Tx: under what conditions can UTXO be spent

1. The program ScriptSig | ScriptPK returns true
2. TxID | index is in the current UTXO set
3. sum input values ≥ sum output values

After Tx2 is posted, miners remove $UTXO_2$ from UTXO set

Resources

- ECE/COS 470, Pramod Viswanath, Princeton 2024
- CS251, Dan Boneh, Stanford 2023