DFINITY

INTRODUCING

DFINITY Crypto Techniques

V1 - 19th May 2017

DFINITY

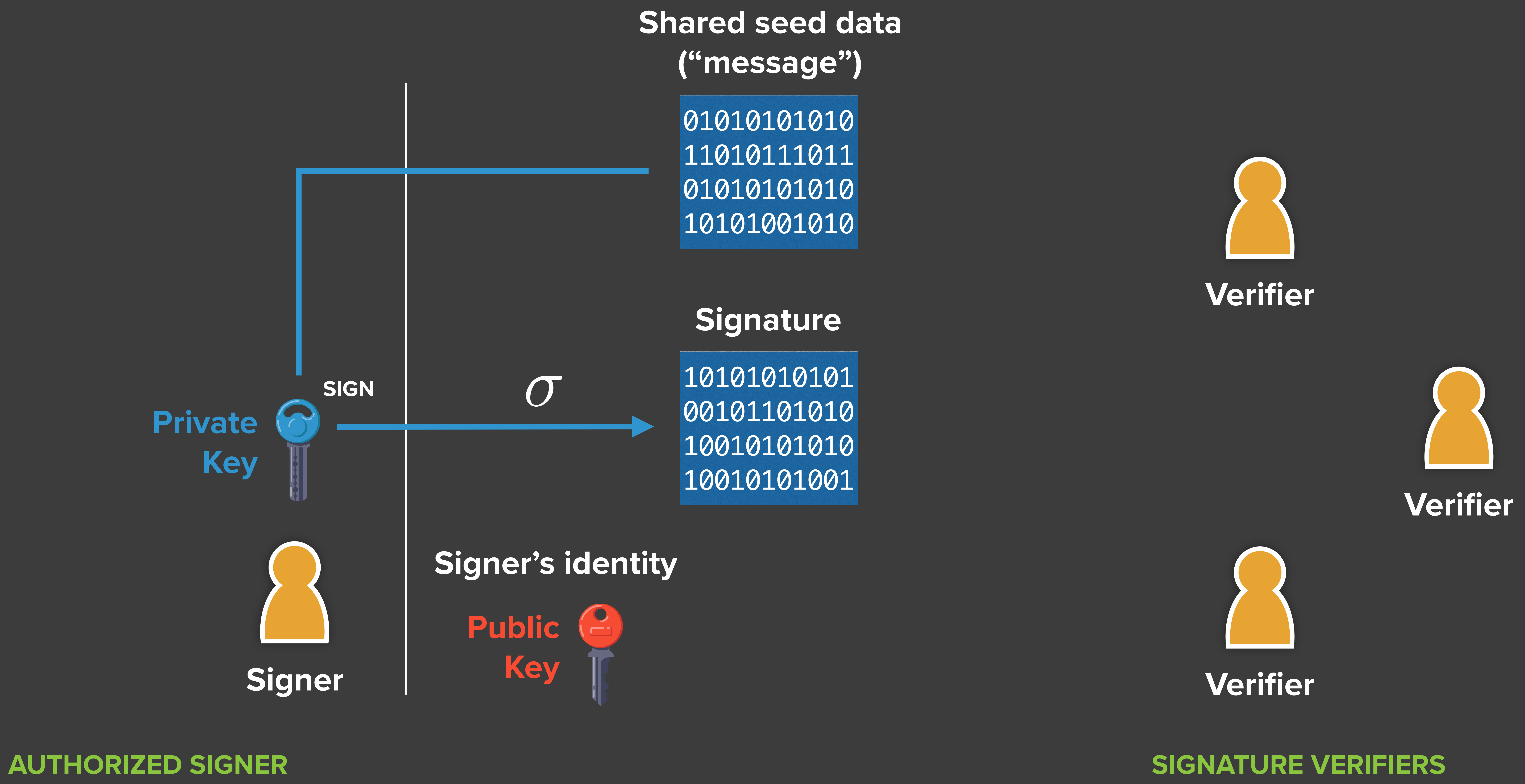
Threshold Relay

Produce randomness that is incorruptible, unmanipulable and unpredictable

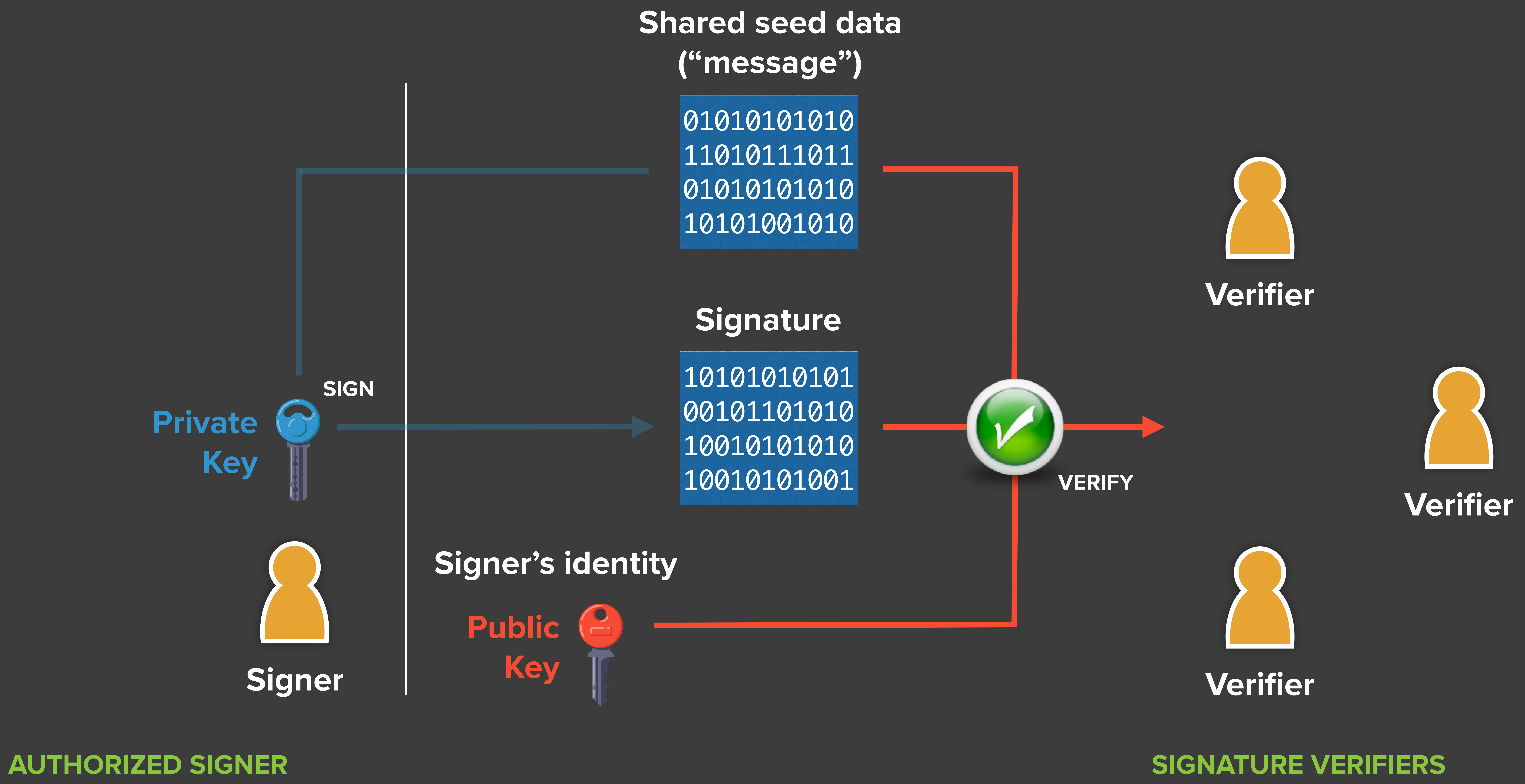
BACKGROUNDER

Explain “unique deterministic” threshold signatures...

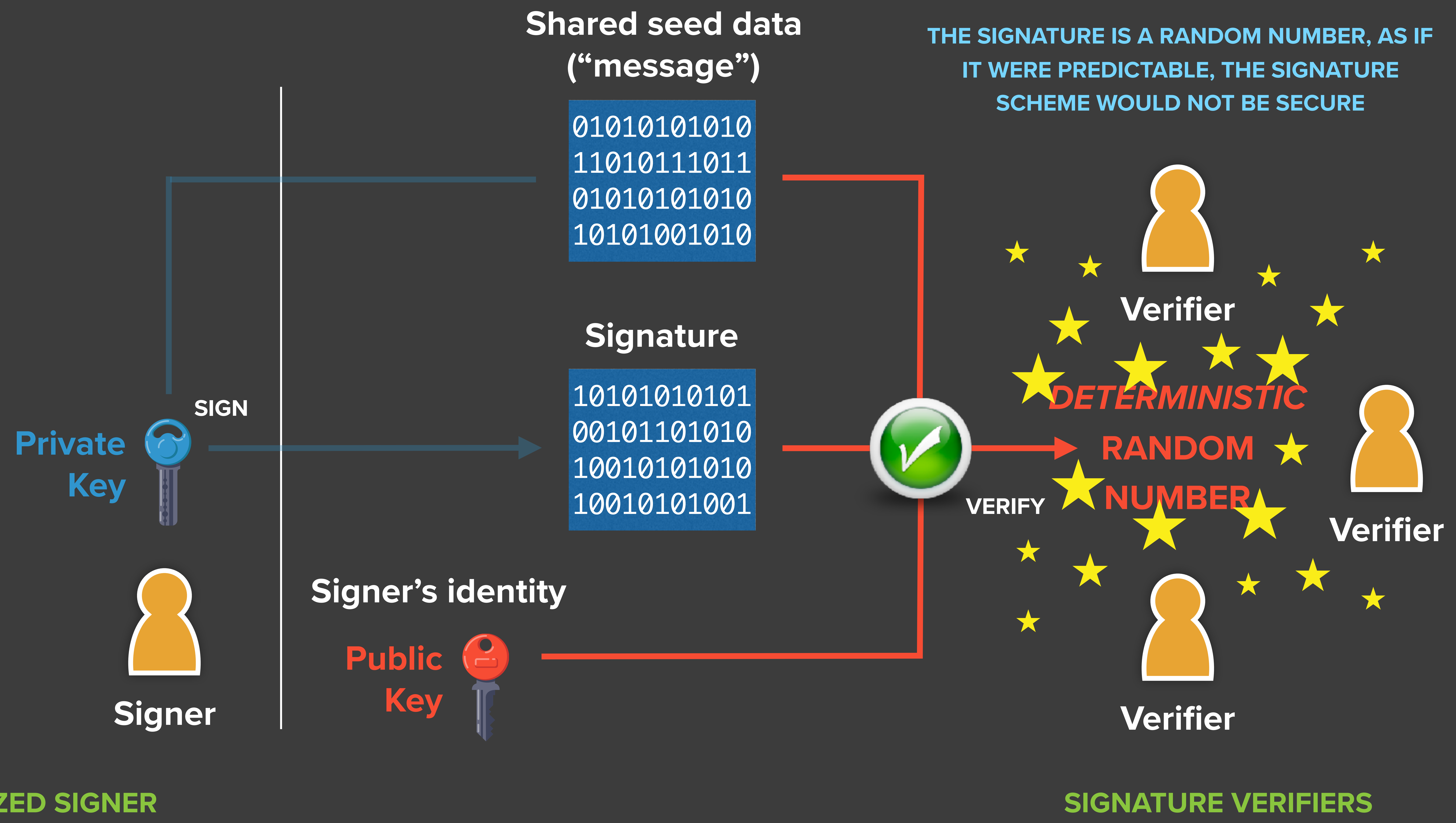
Usually a signer creates a signature on message data



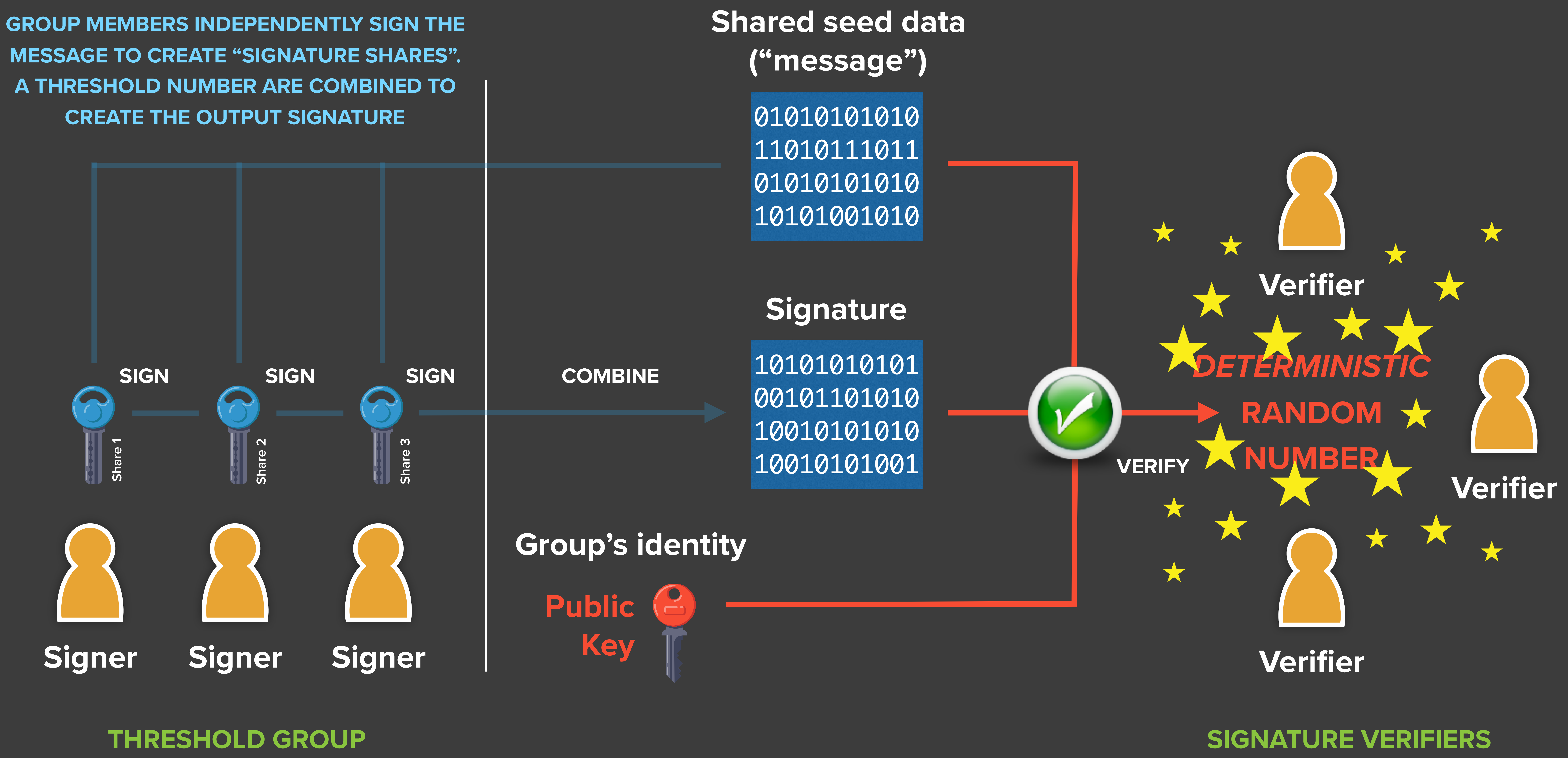
That can be verified using the signer’s public key



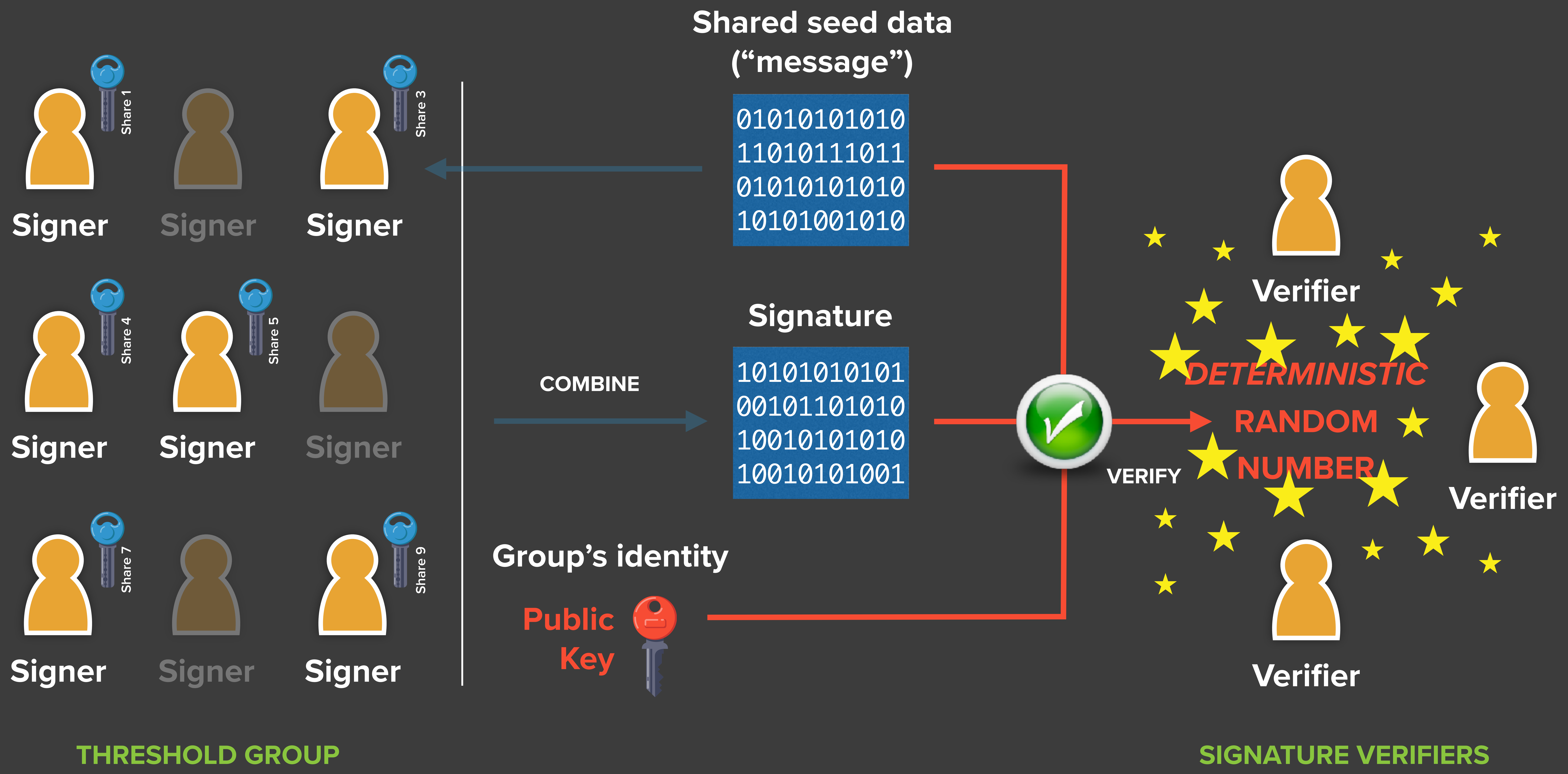
If scheme unique and deterministic then only 1 correct signature



Unique and deterministic threshold signature scheme possible



Whatever subset (threshold) of group sign still same signature



Important observations of powerful magic

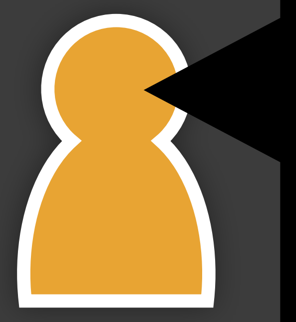
1. A group identified by its threshold public key can only produce a *single valid output signature* on given seed data

Important observations of powerful magic

1. A group identified by its threshold public key can only produce a *single valid output signature* on given seed data

2. A group is *fault tolerant* and *any subset of threshold size* can distribute signature shares for combination into the signature

Important observations of powerful magic



1. A group identified by its threshold public key can only produce a *single valid output signature* on given seed data
2. A group is *fault tolerant* and *any subset of threshold size* can distribute signature shares for combination into the signature
3. The resulting threshold signature can be *validated by anyone* who has the group's public key and the seed data

Important observations of powerful magic

1. A group identified by its threshold public key can only produce a *single valid output signature* on given seed data
2. A group is *fault tolerant* and *any subset of threshold size* can distribute signature shares for combination into the signature
3. The resulting threshold signature can be *validated by anyone* who has the group's public key and the seed data
4. The signature is a deterministically produced *random number*

Important observations of powerful magic

1. A group identified by its threshold public key can only produce a *single valid output signature* on given seed data

2. A group is *fault tolerant* and *any subset of threshold size* can distribute signature shares for combination into the signature

3. The resulting threshold signature can be *validated by anyone* who has the group's public key and the seed data

4. The signature is a deterministically produced *random number*

5. Given a group's public key and the input seed data the verifiers reach *immediate consensus* on the random number produced *without running a consensus protocol…*

A unique deterministic threshold signature scheme

Boneh-Lynn-Shacham signatures (BLS)

TIP 1 Ben Lynn is a full time member of the DFINITY team

TIP 2 You don’t need to understand this crypto to understand the remaining slides...

Key Generation

- Secret key: $x \bmod r$
- Public key: $P = xQ_2 \in G_2$

Signing

- Message hashed to $H(m) \in G_1$
- Signature: $s = xH(m) \in G_1$

Verification $e(s, Q_2) = e(H(m), P)$?

Parameters

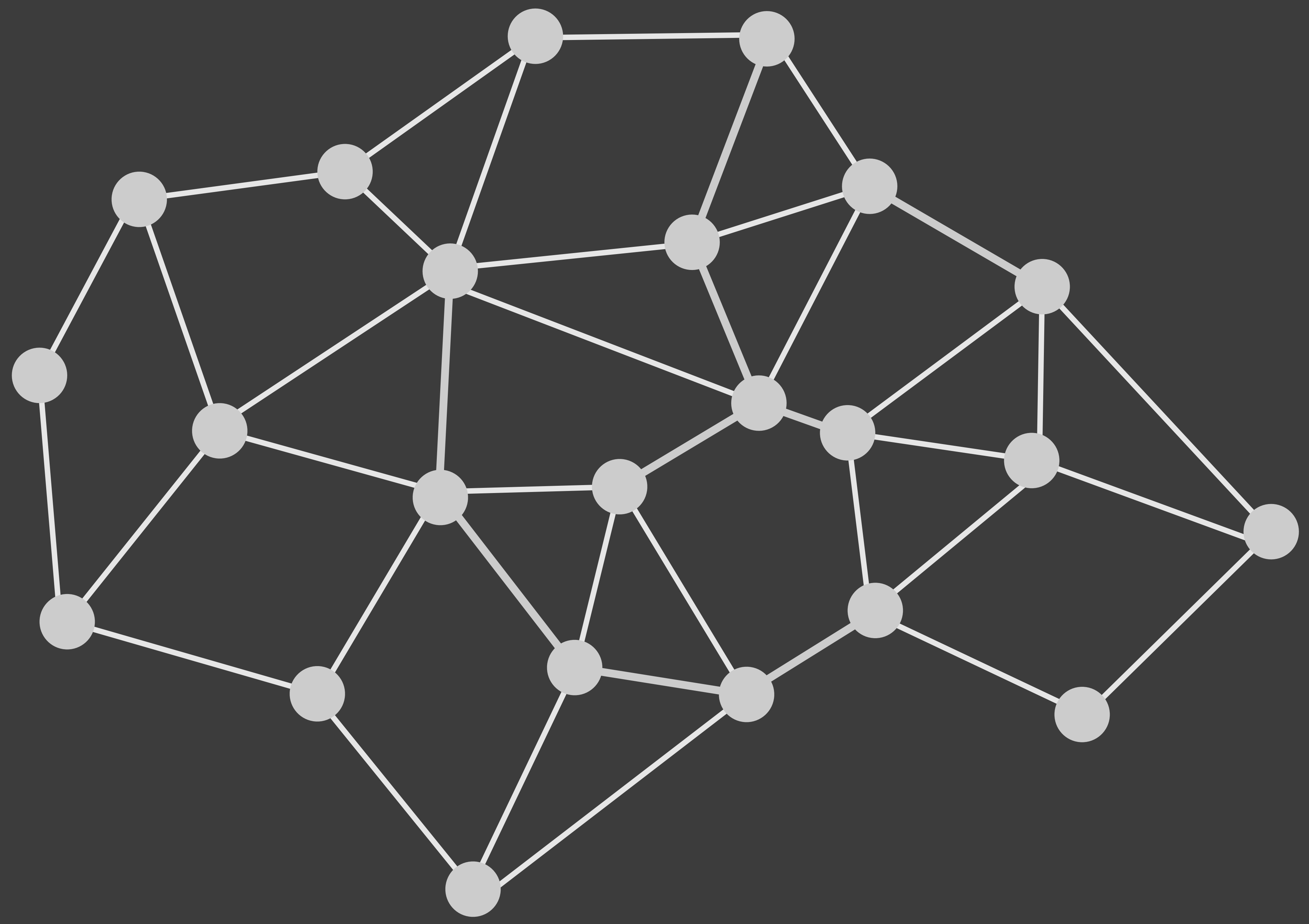
- Two groups G_1, G_2 of prime order r (on two elliptic curves)
- Generators $Q_1 \in G_1, Q_2 \in G_2$
- Bi-linear pairing $e : G_1 \times G_2 \mapsto G_T$

BLS, 2001 (Stanford University)

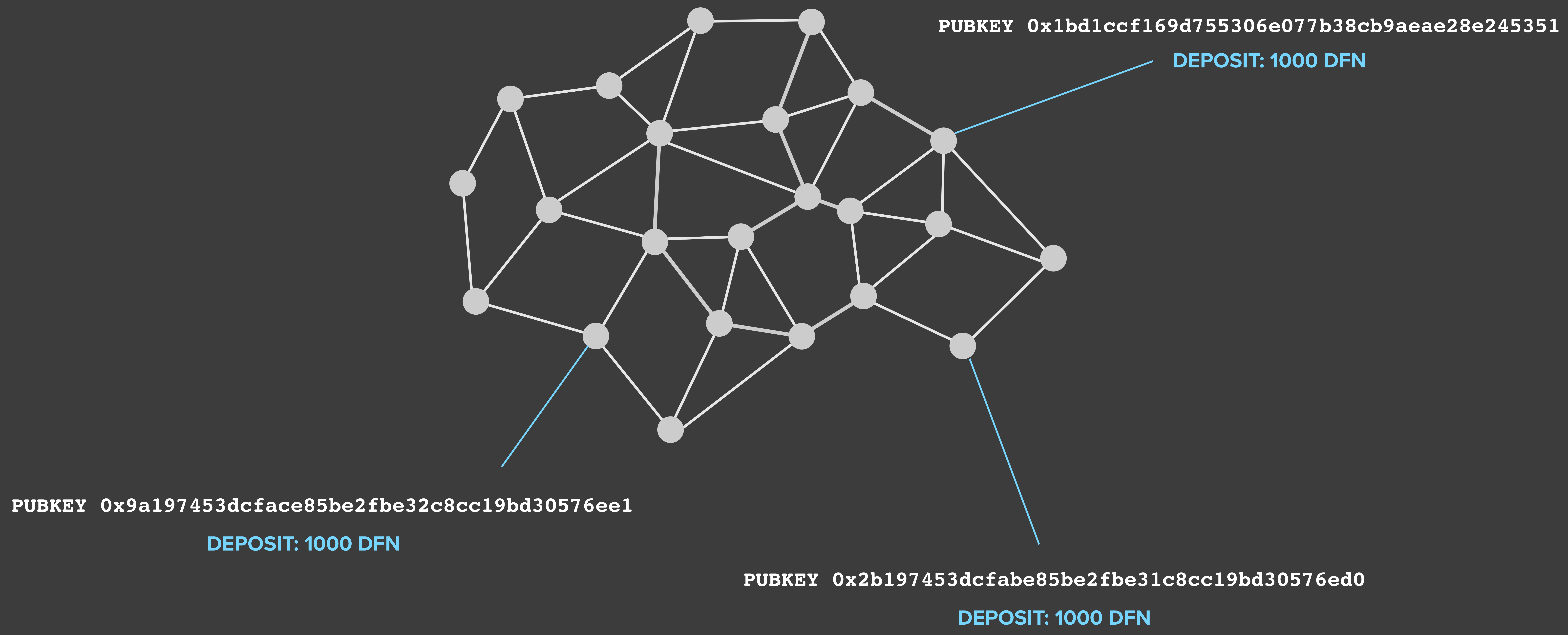
DECENTRALIZED VERIFIABLE RANDOM FUNCTION

Relay between groups to create a random sequence

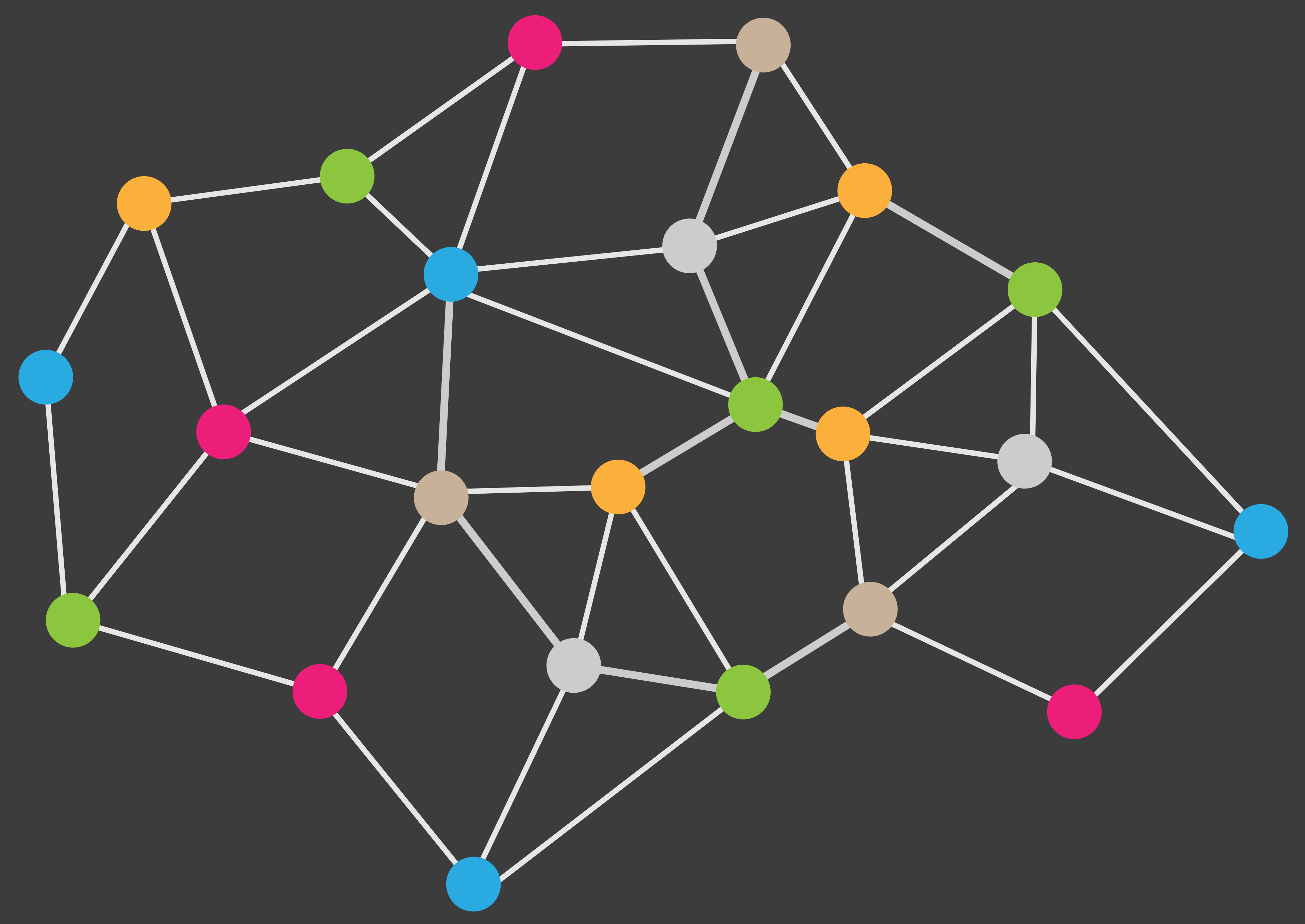
A vast peer-to-peer broadcast network of mining clients…



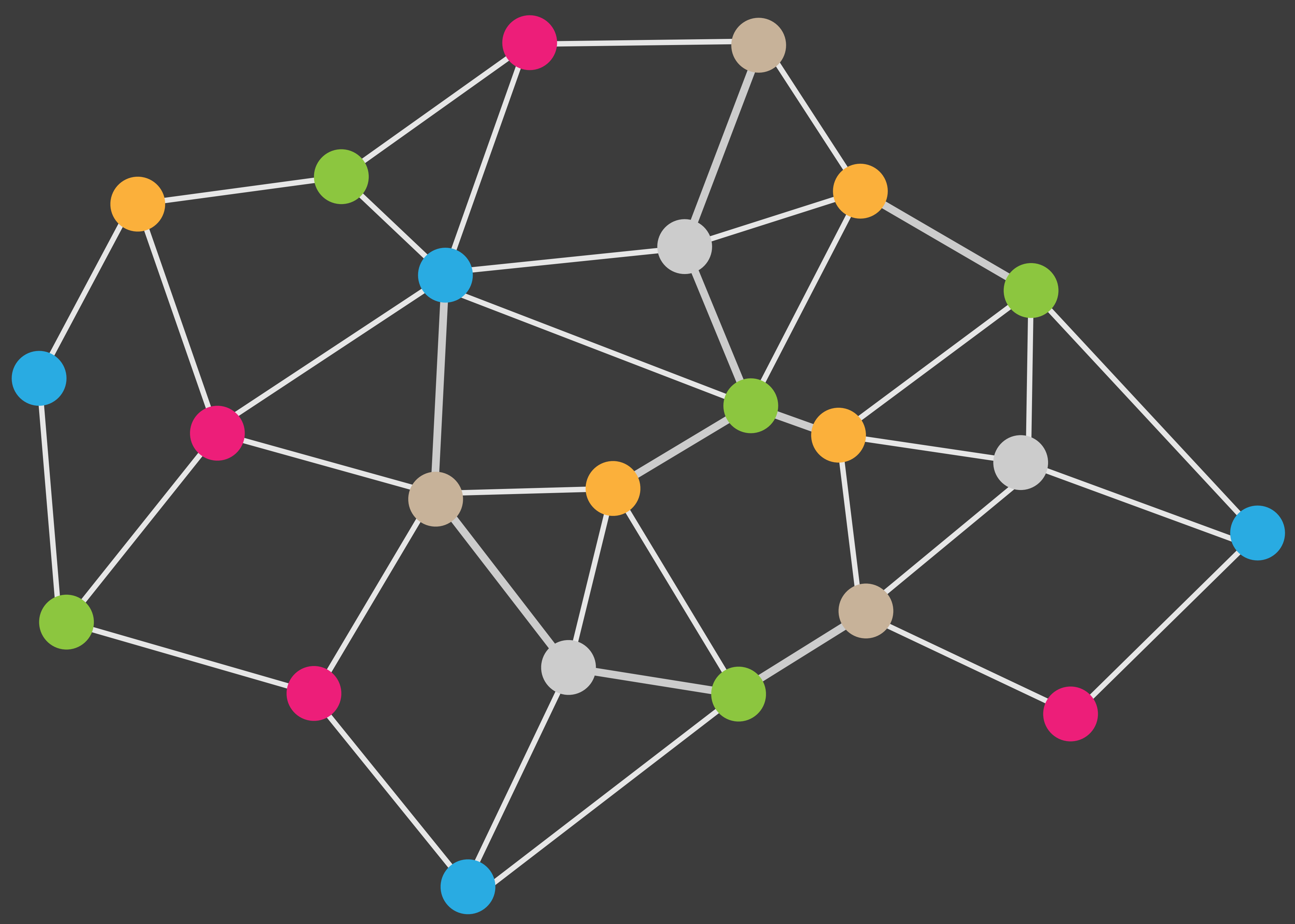
Whose public keys are registered on a supporting ledger



Each client (“process”) belongs to threshold groups



Whose public keys are also registered on the supporting ledger



GRP PUBKEY 0x7de4ac5...

GRP PUBKEY 0x8fb251b...

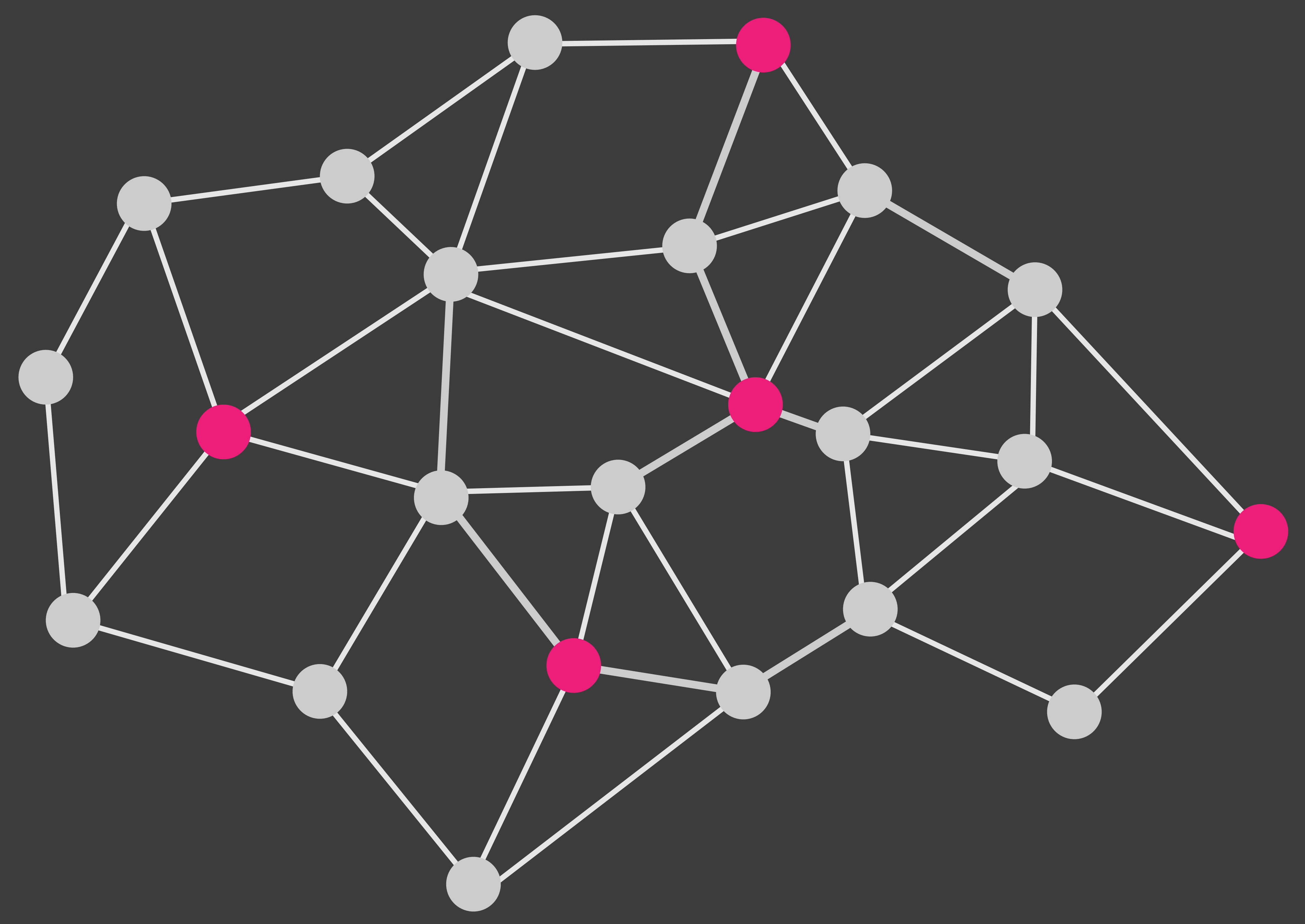
GRP PUBKEY 0x1a7234e...

GRP PUBKEY 0x2b197453...

GRP PUBKEY 0xb6e1a33...

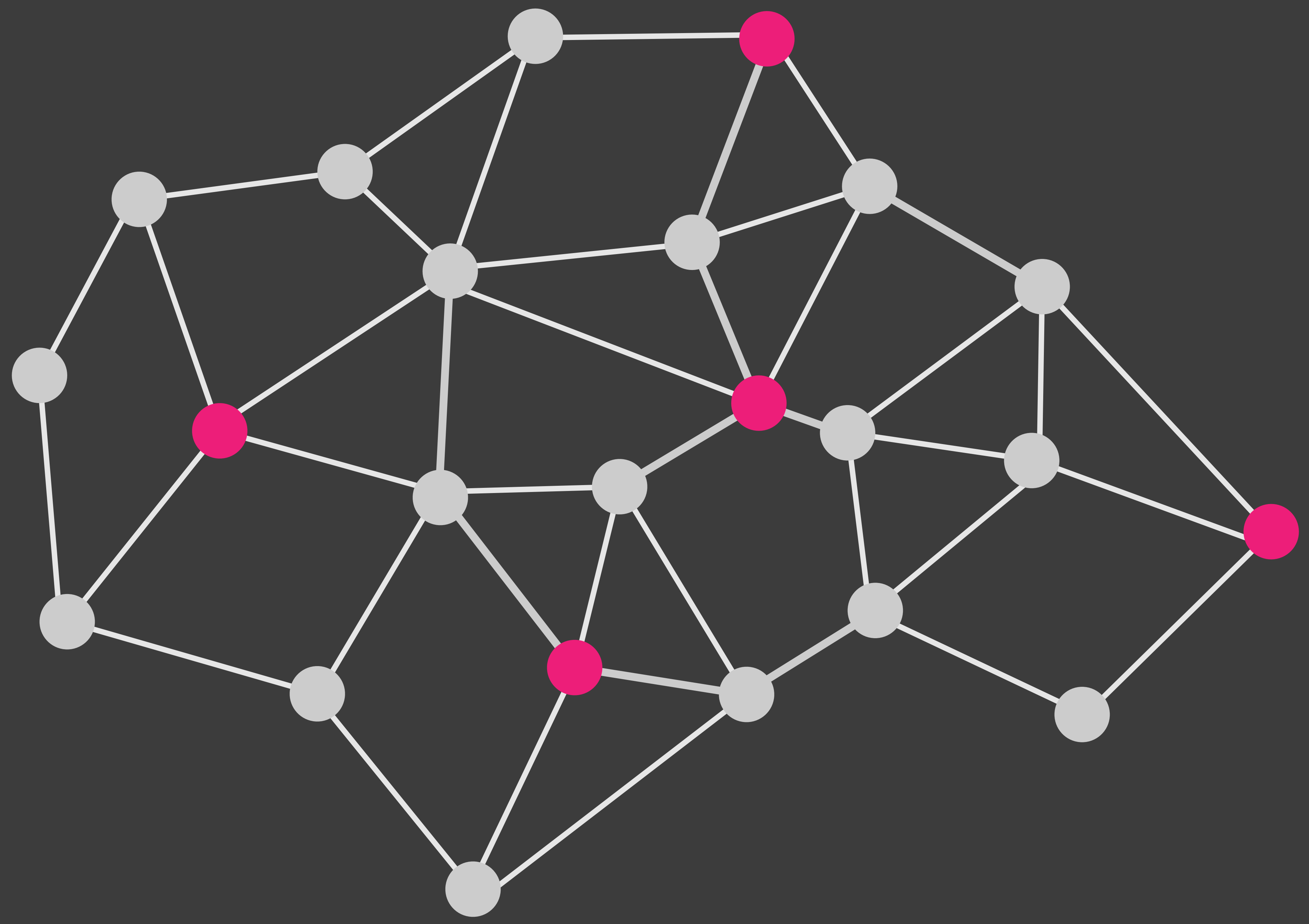
...

At each height in the sequence, there is a current group...



h

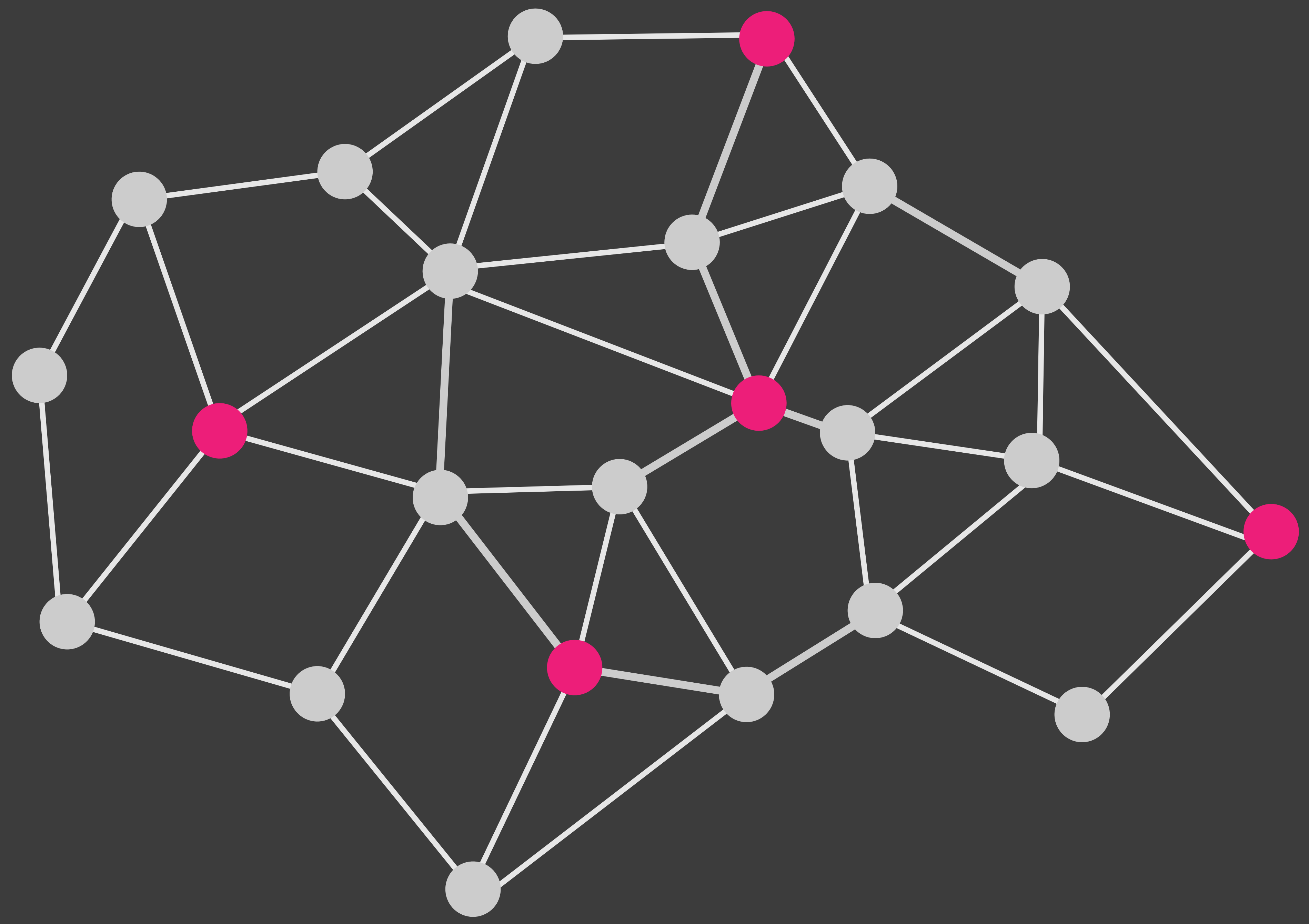
That signs the previous group's signature...



$$e(\sigma, g) = e(H(m), g^x)$$

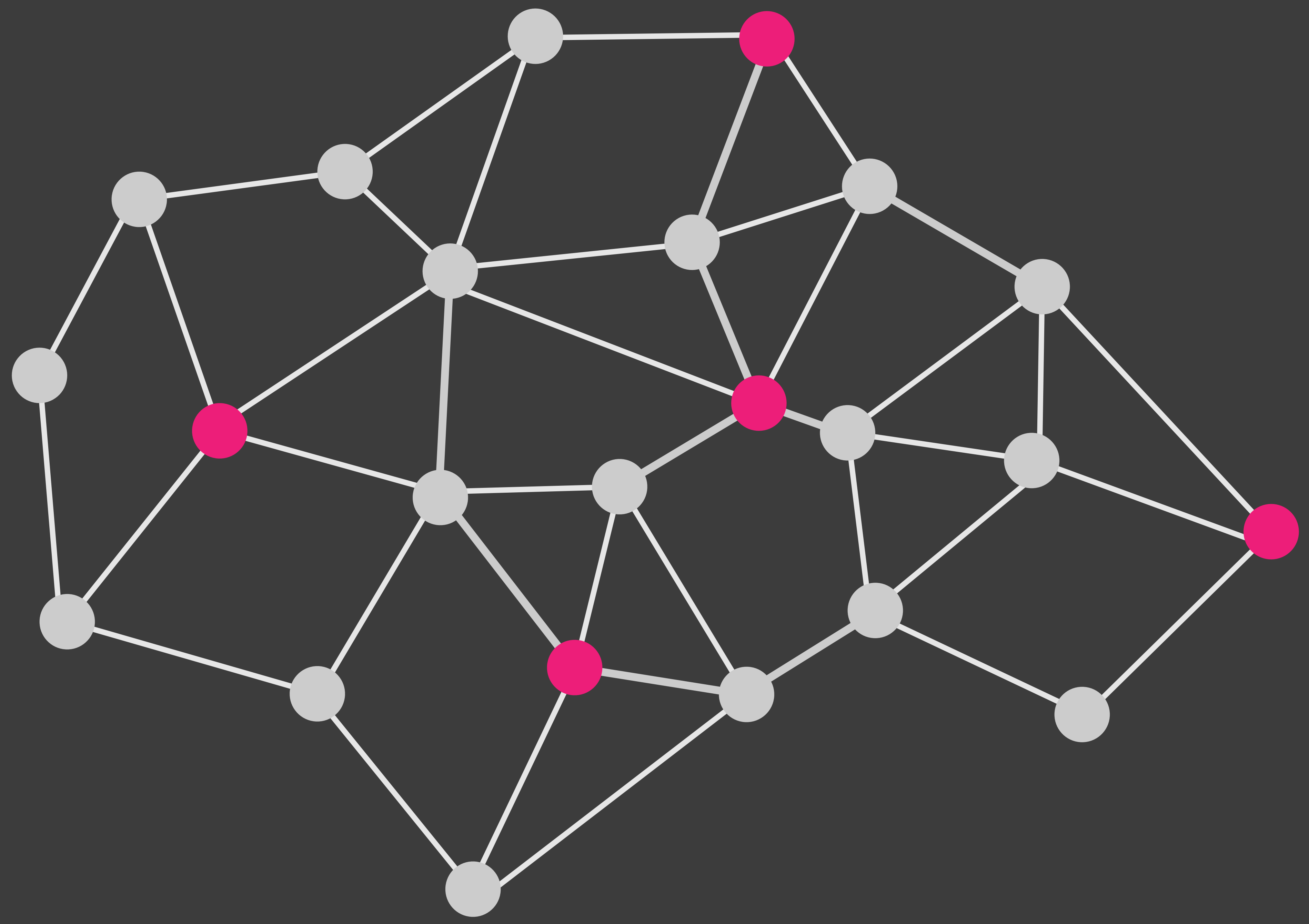
BLS Signature Scheme

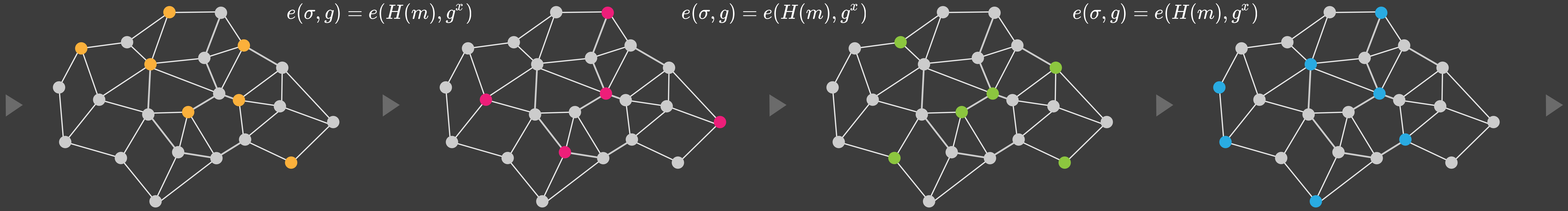
Their random number selects the next group (the “relay”)



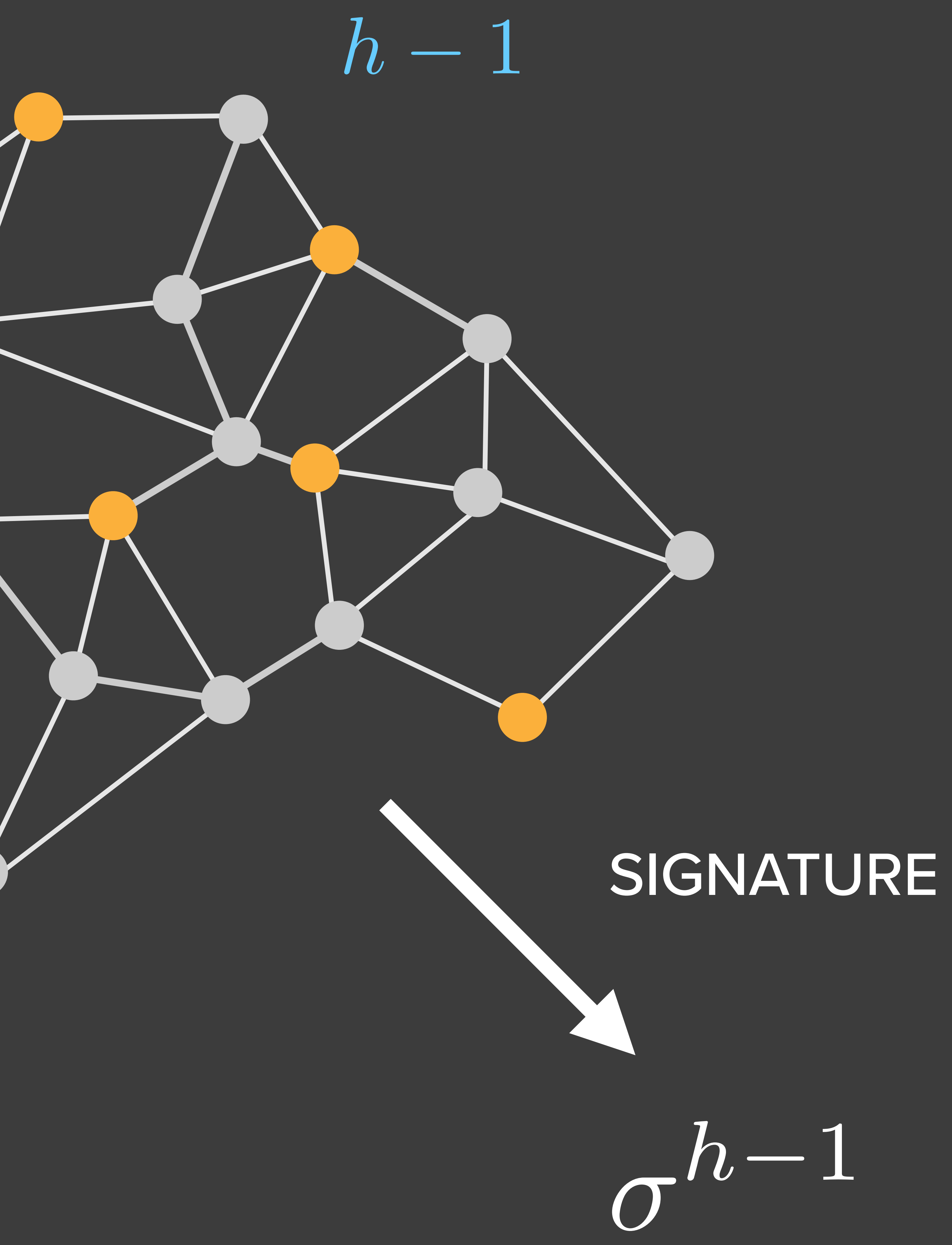
$$G^{h+1} = \mathcal{G}[\sigma^h \bmod |\mathcal{G}|]$$

The relaying between groups is unmanipulable and infinite

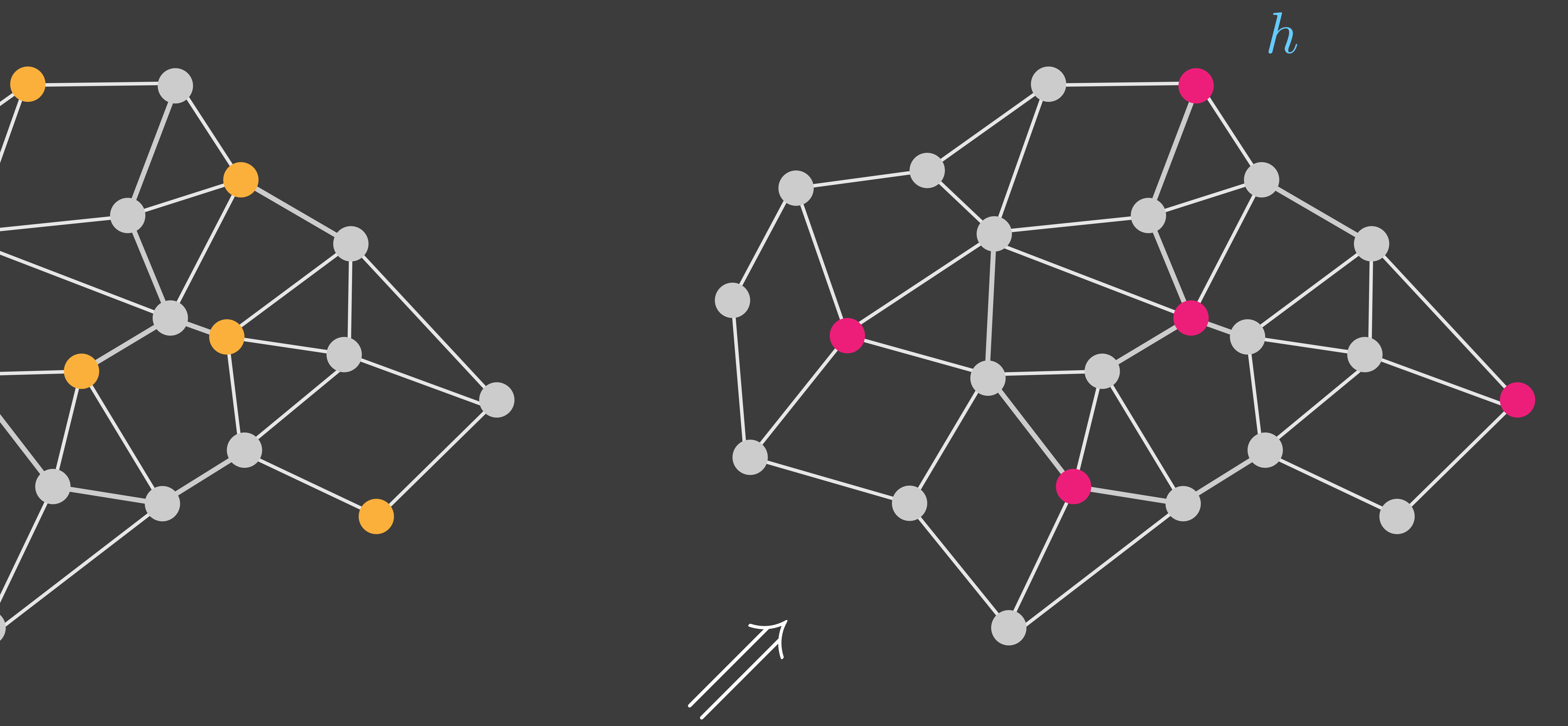




This is what Threshold Relay looks like

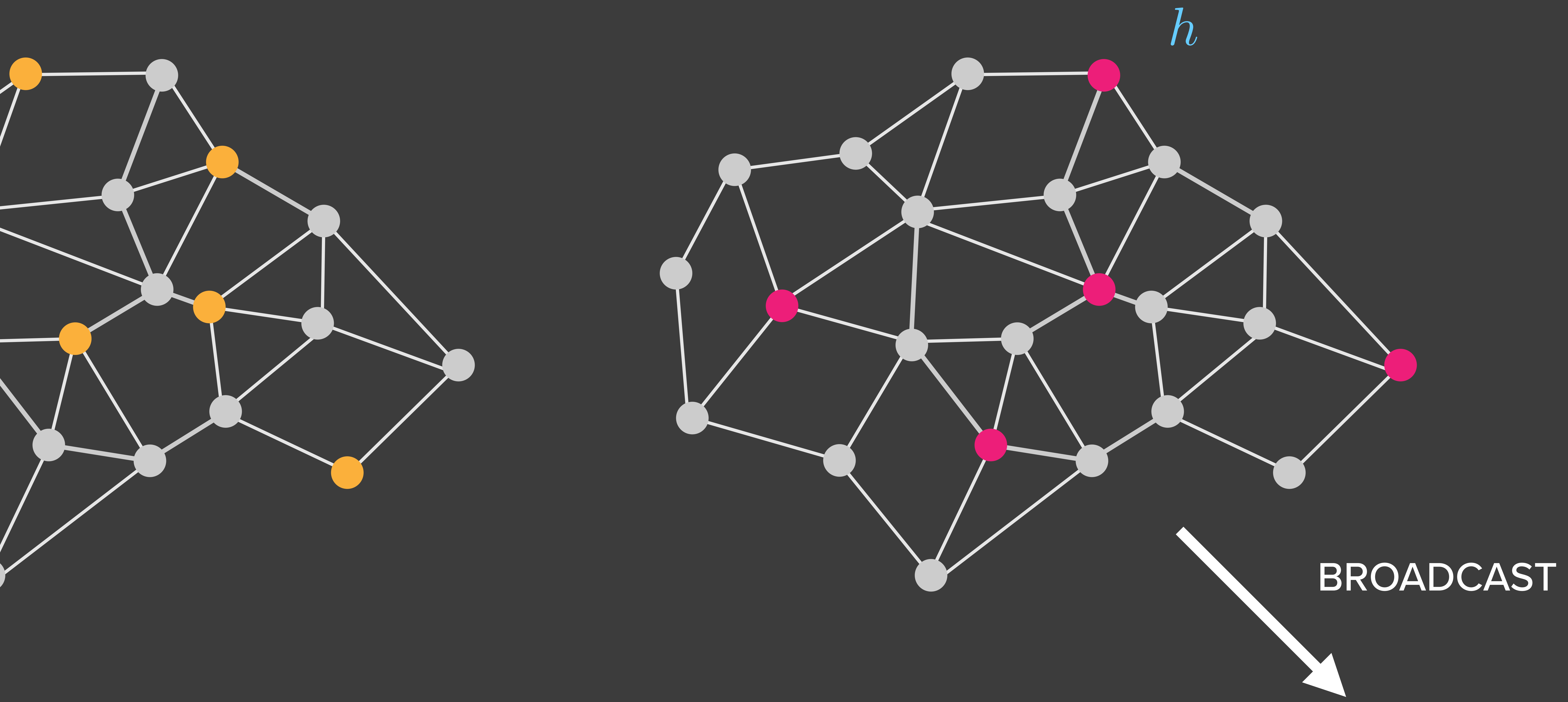


The signature created at *h-1* selects the group at *h*



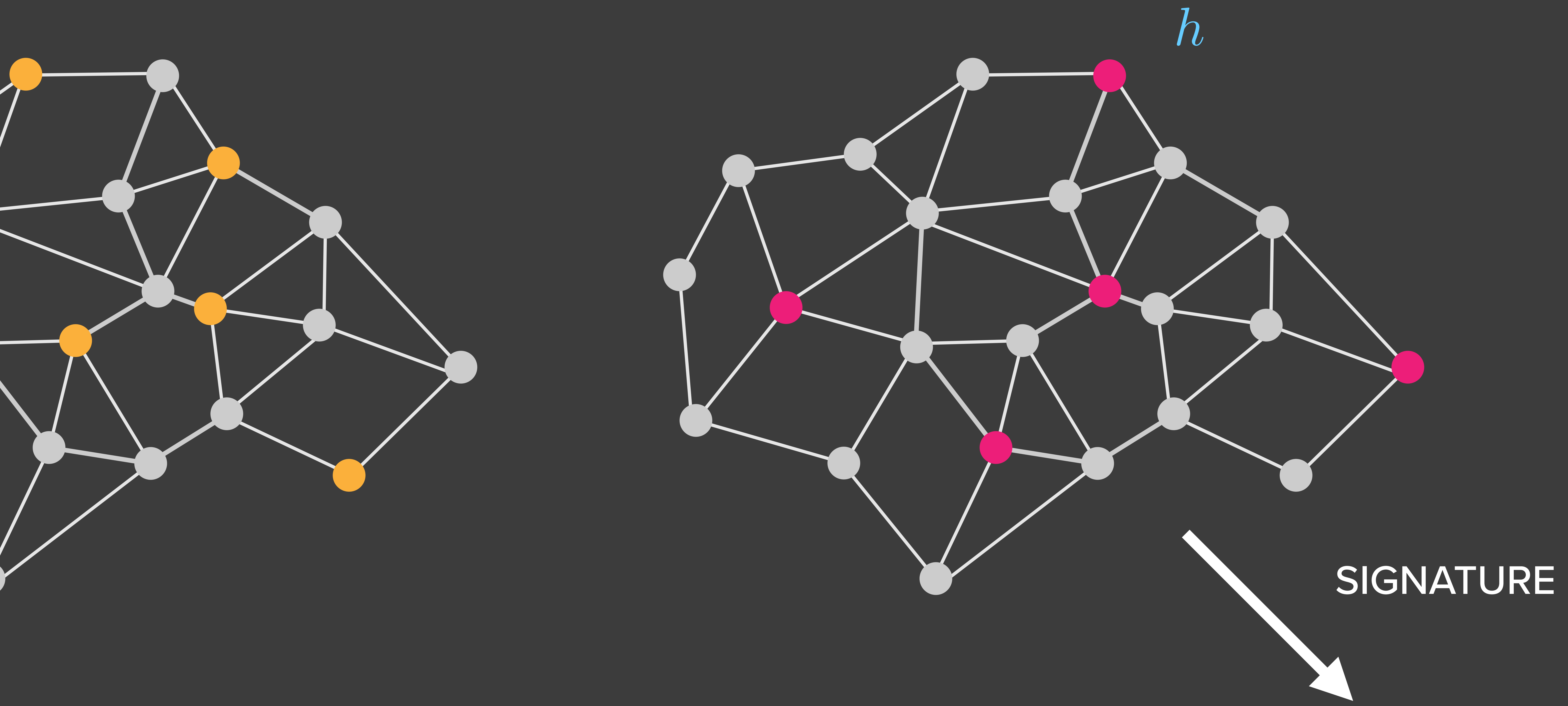
$$G^h = \mathcal{G}[\sigma^{h-1} \bmod |\mathcal{G}|]$$

Group members at *h* broadcast signature shares



$$\{\sigma_p^h, p \in G^h\}$$

Collect threshold of shares & create unique group signature...



$$\sigma^h = bls(\{\sigma_p^h, p \in G^h\})$$

That selects the next group, ad infinitum

$$G^{h+1} = \mathcal{G}[\sigma^h \bmod |\mathcal{G}|]$$

Producing a decentralized Verifiable Random Function (VRF)

$$\sigma^{h-7}, \quad \sigma^{h-6}, \quad \sigma^{h-5}, \quad \sigma^{h-4}, \quad \sigma^{h-3}, \quad \sigma^{h-2}, \quad \sigma^{h-1}, \quad \sigma^{h} \implies$$

Random number sequence is

Deterministic • Verifiable • Unmanipulable

Next value released on agreement a threshold of the current group...

Unpredictable

No consensus protocol is necessary!

“Random numbers should not be generated with a method chosen at random

- Donald Knuth

TLDR; such unmanipulable randomness is powerful...

Decentralized Protocols
for “Scaling Out”

DFINITY

COMING UP...

PSP Blockchain Designs

Validation Towers

Validation Trees

USCIDs

Lottery Charging

Lazy Validation

Decentralized Applications
with advanced features

ϕ

E.g. PHI autonomous loan issuance and crypto “fiat”

Validate anything...

Fair financial exchanges...

Fault Tolerance Example

NETWORK METRICS

Processes	10,000
Faulty	3,000
(Correct)	7,000
Group Size	400
Threshold	201

Note: in practice the probability 30% of professionally run mining processes "just stop" is very low. Miners will generally deregister IDs to retrieve deposits when exiting.

$$\frac{P(Faulty \geq 200)}{1e{-}17}$$

Probability that a sufficient proportion of the group are faulty that it cannot produce a signature

Calculated using hypergeometric probability e.g. http://www.geneprof.org/GeneProf/tools/hypergeometric.jsp

Note: groups should expire to thwart "adaptive" adversaries

Communications Overhead **Example**

MESSAGE FORMAT

Process ID	20 bytes
Signature share	32 bytes
Signature on comms	32 bytes
Total	84 bytes

In order for a group to produce a threshold signature, its members must broadcast "signature shares" on the message that can be combined. Here is a typical packet carrying a signature share.

GROUP SIZE

Group size	400
Threshold	201

COMMUNICATION OVERHEAD

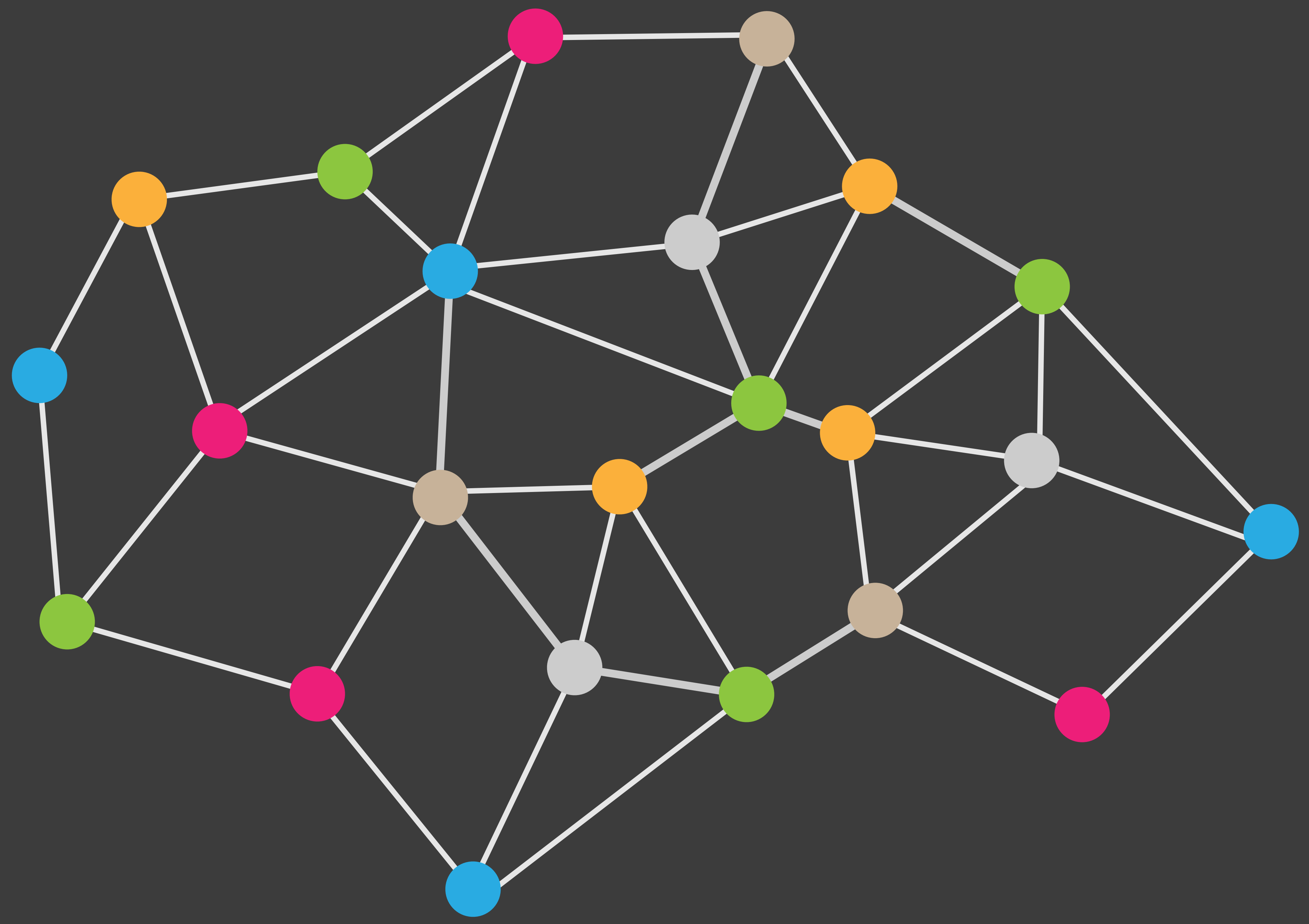
Expected	22 KB

400 messages involve 34 KB of data transfer. However, only 17 KB (half the messages) are required to construct the signature. Thereafter signature shares are not relayed, so a more typical overhead is 22 KB.

BACKGROUNDER

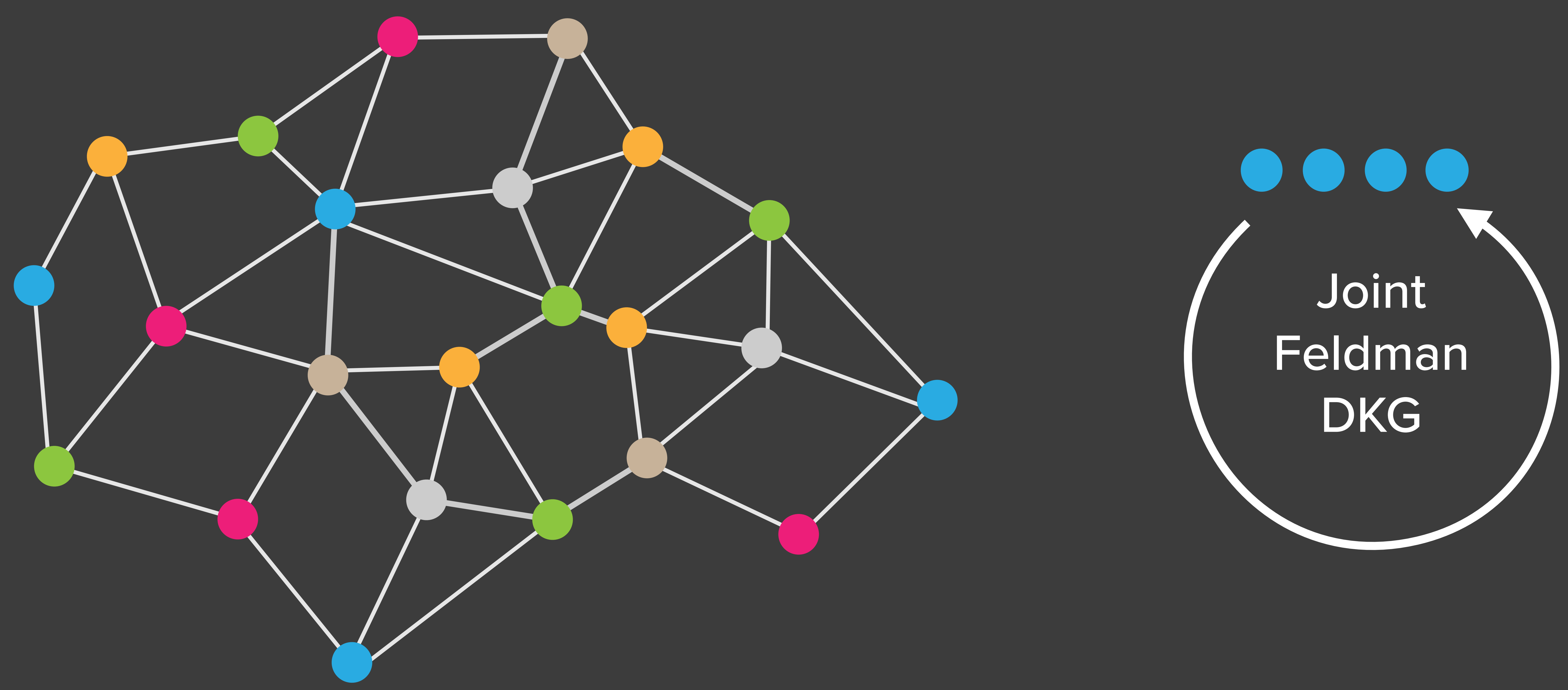
How to setup groups…

Clients randomly assigned to groups by randomness (VRF)



GRP PUBKEY | GRP PUBKEY | GRP PUBKEY | GRP PUBKEY | GRP PUBKEY | •••

Need setup threshold scheme within 1000 blocks using DKG...



GRP PUBKEY

GRP PUBKEY

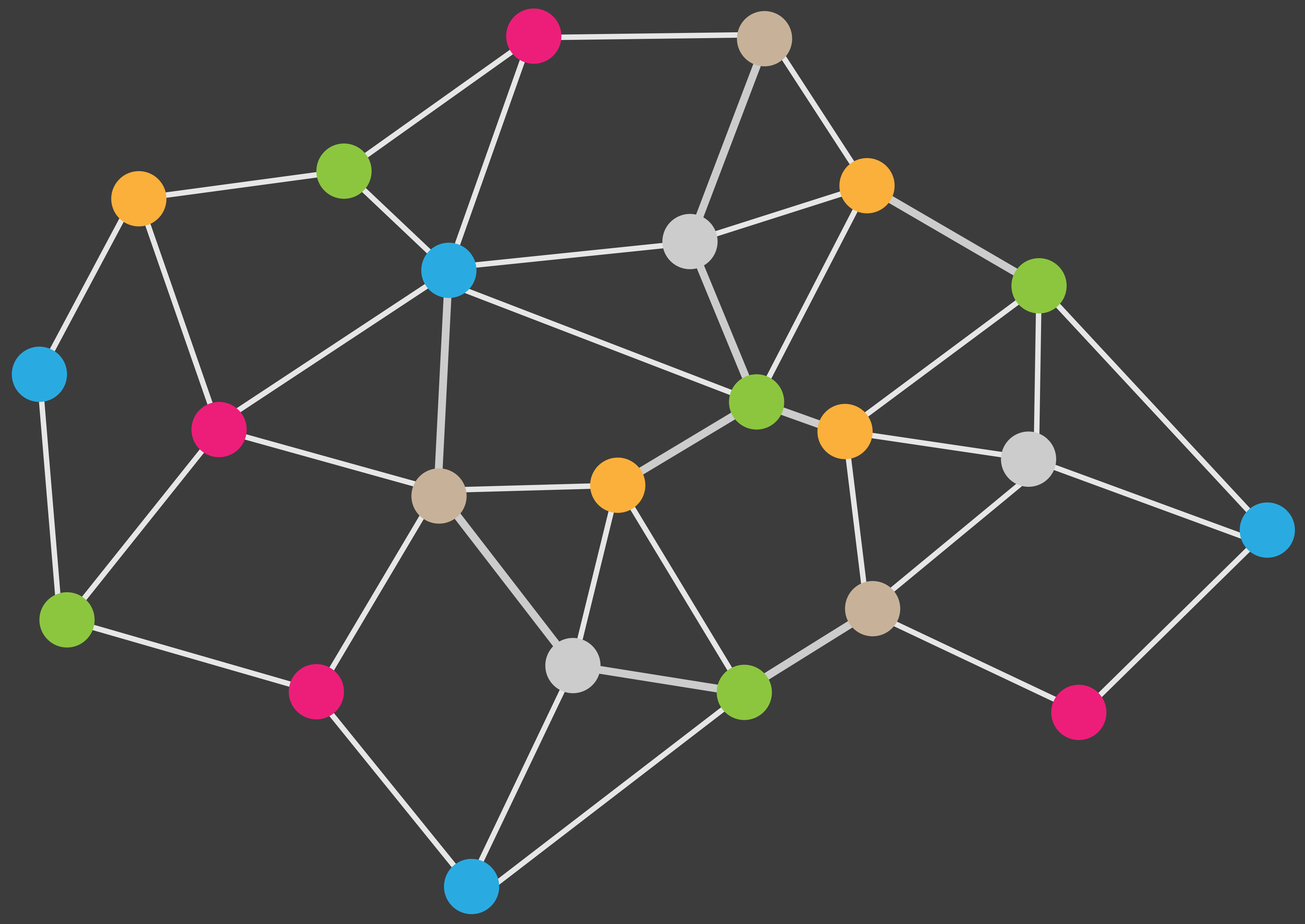
GRP PUBKEY

GRP PUBKEY

GRP PUBKEY

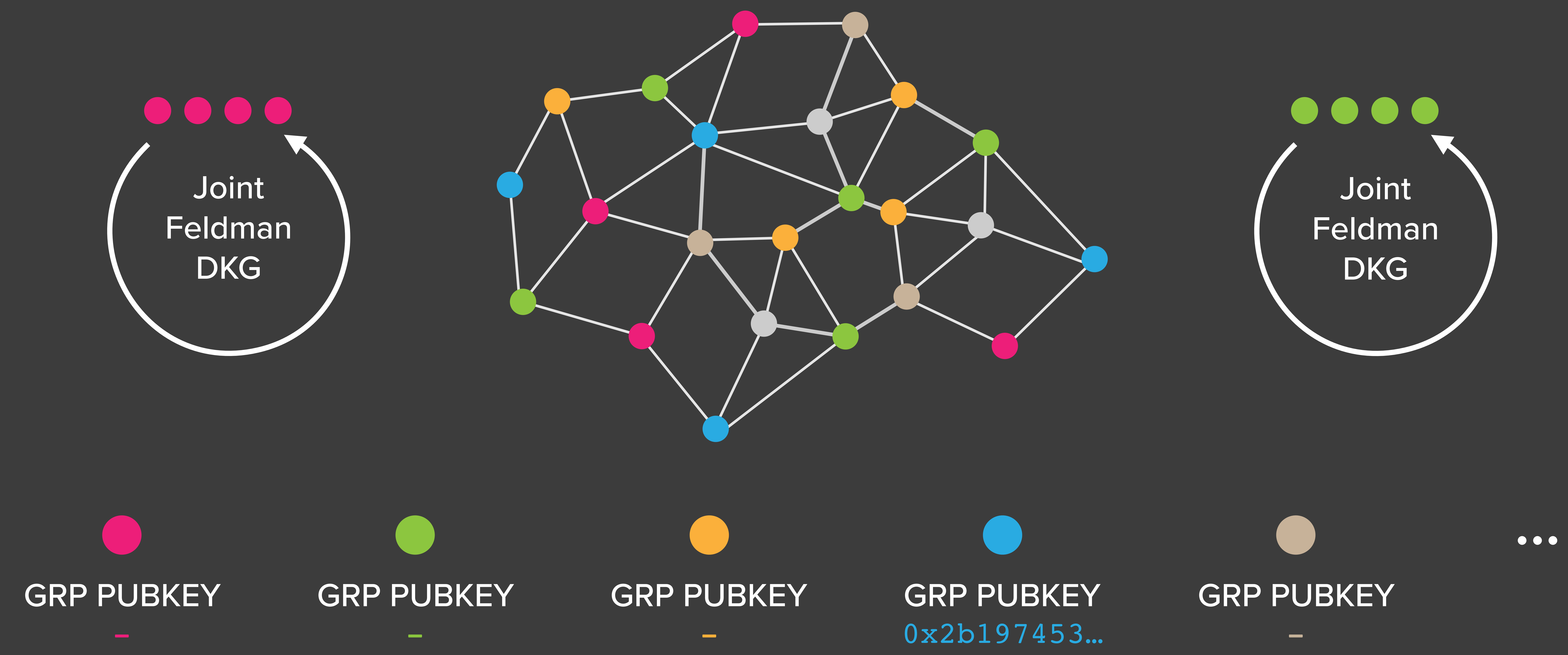
...

Successful groups register their Public Key on the ledger

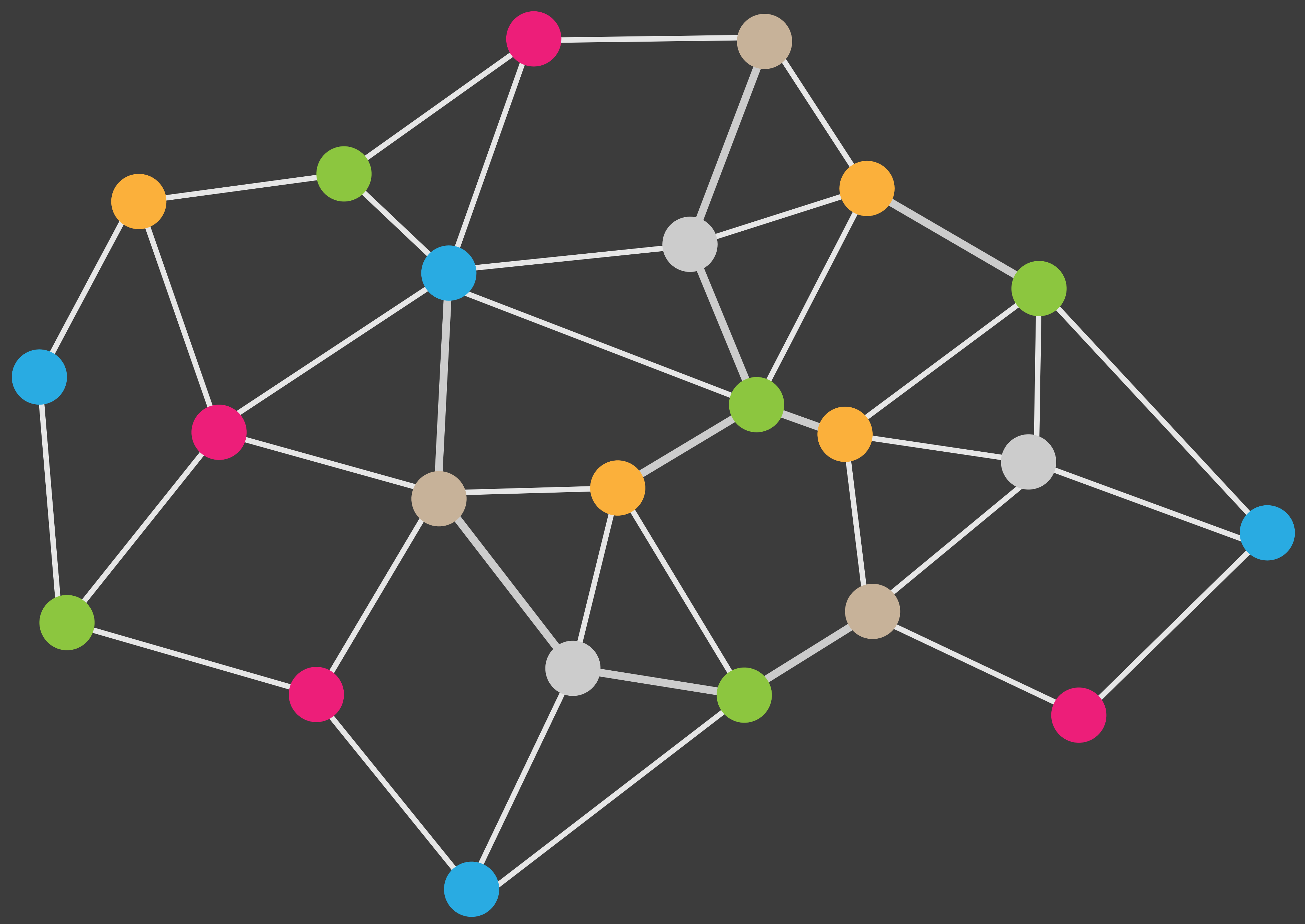


GRP PUBKEY	GRP PUBKEY	GRP PUBKEY	GRP PUBKEY	GRP PUBKEY	...
–	–	–	0x2b197453…	–	

Setup is independent of blockchain progression...

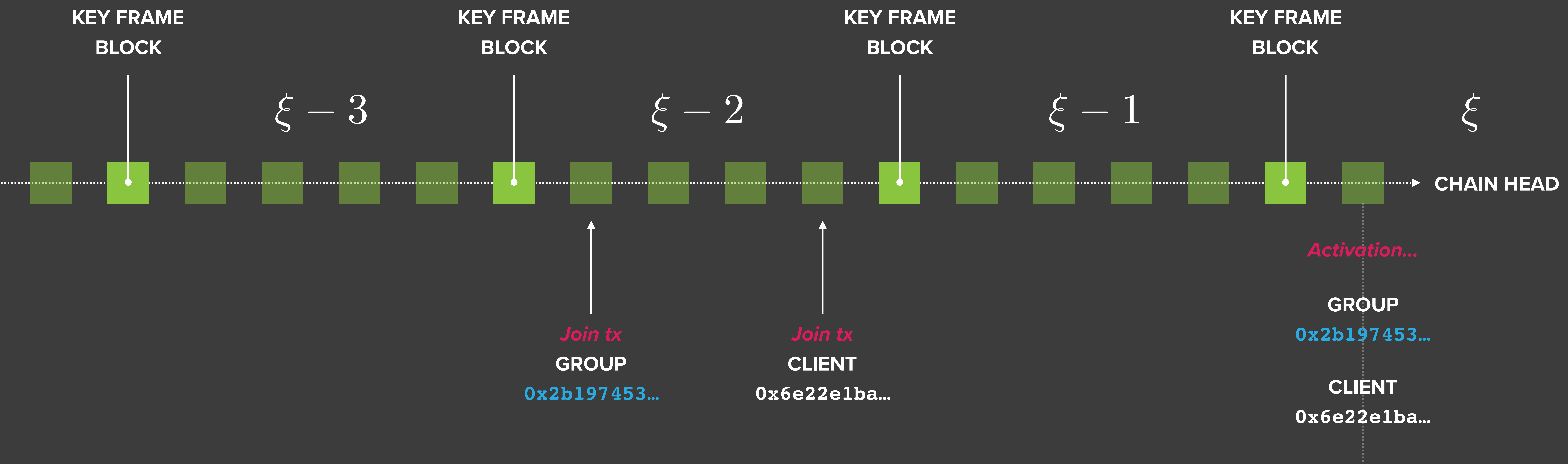


And occurs asynchronously



GRP PUBKEY	GRP PUBKEY	GRP PUBKEY	GRP PUBKEY	GRP PUBKEY	...
0x7de4ac5...	0x8fb251b...	–	0x2b197453...	–	

New clients and groups activated in CURRENT_EPOCH + 2

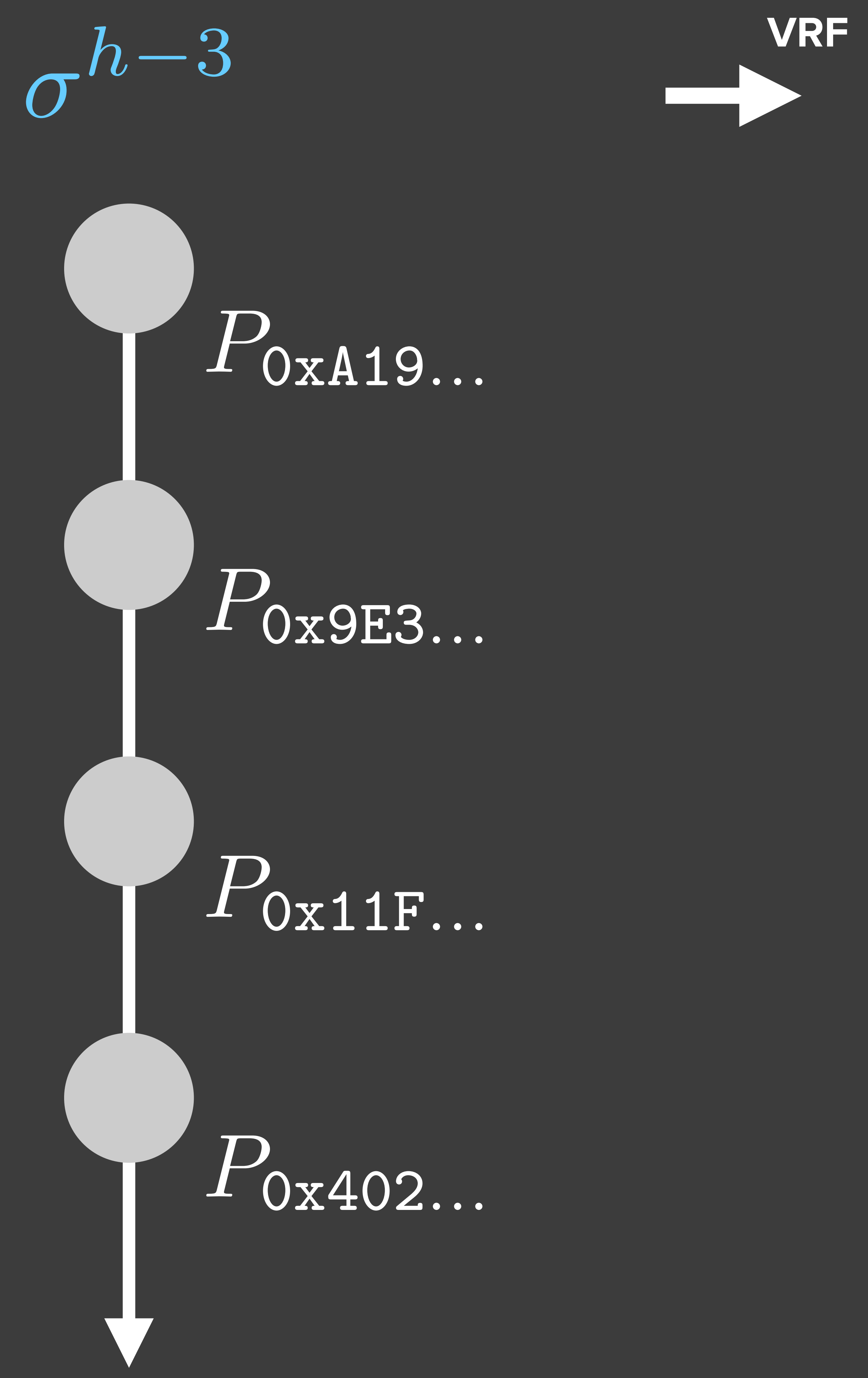


In choosing the epoch length there are a number of considerations. For correctness, an epoch must minimally contain more blocks than may ever be present in a chain fork. However, since light clients only require key frame header copies, for reasons of efficiency, epochs may be much longer e.g. one week

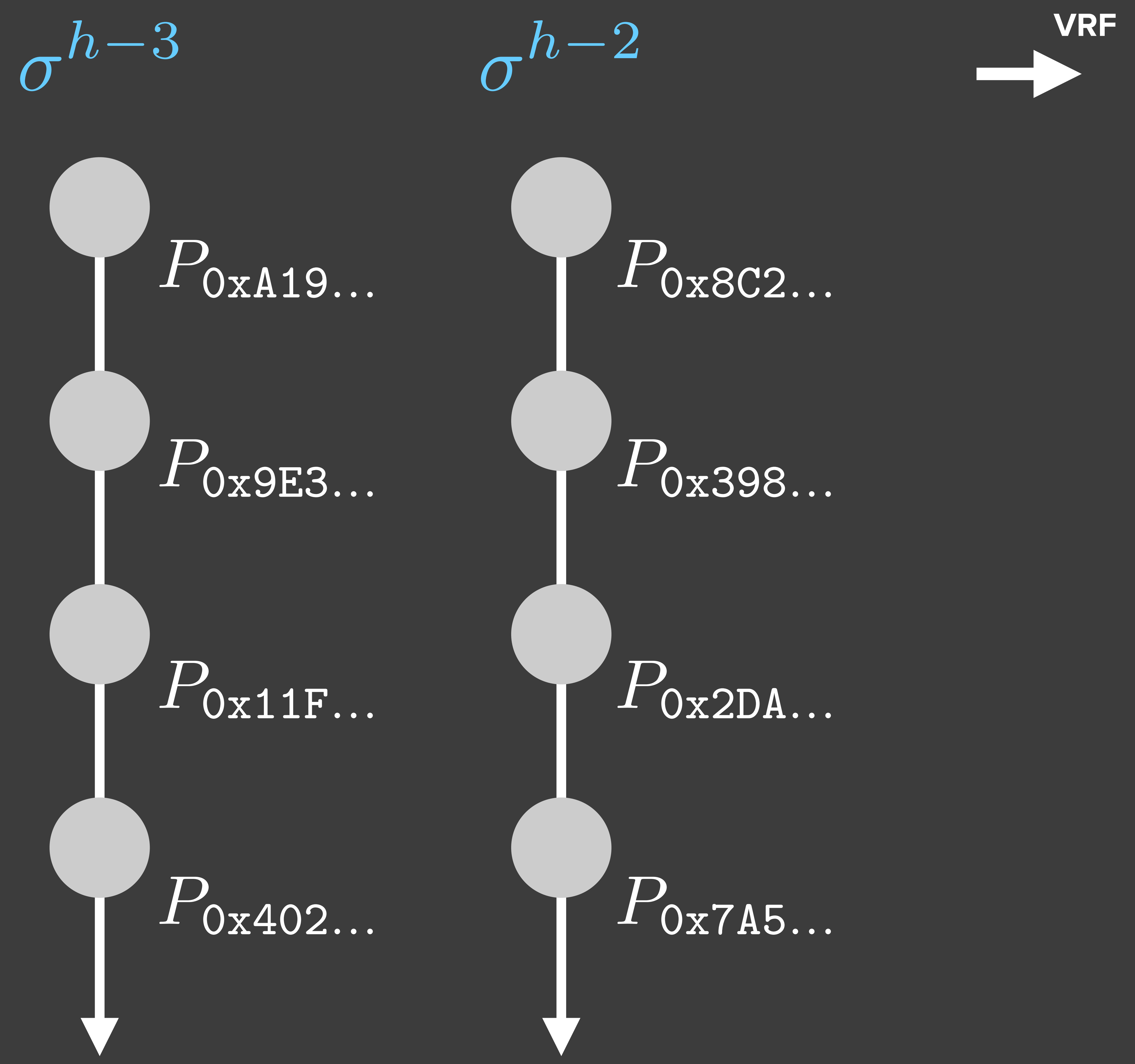
Probabilistic Slot Protocol

Extend the Threshold Relay system to produce a more secure and faster (50X faster than Ethereum) blockchain

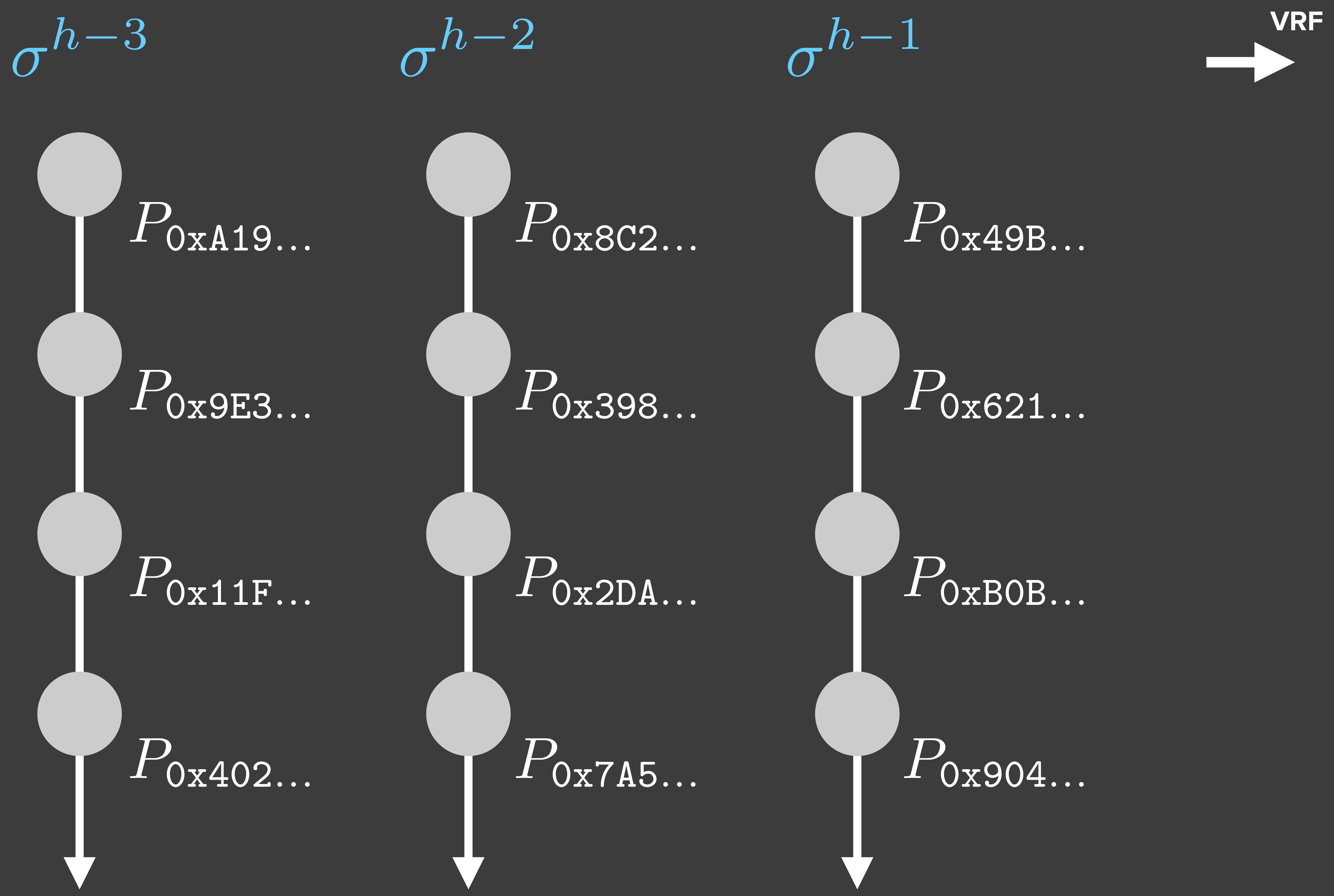
At each height, the randomness orders the processes...



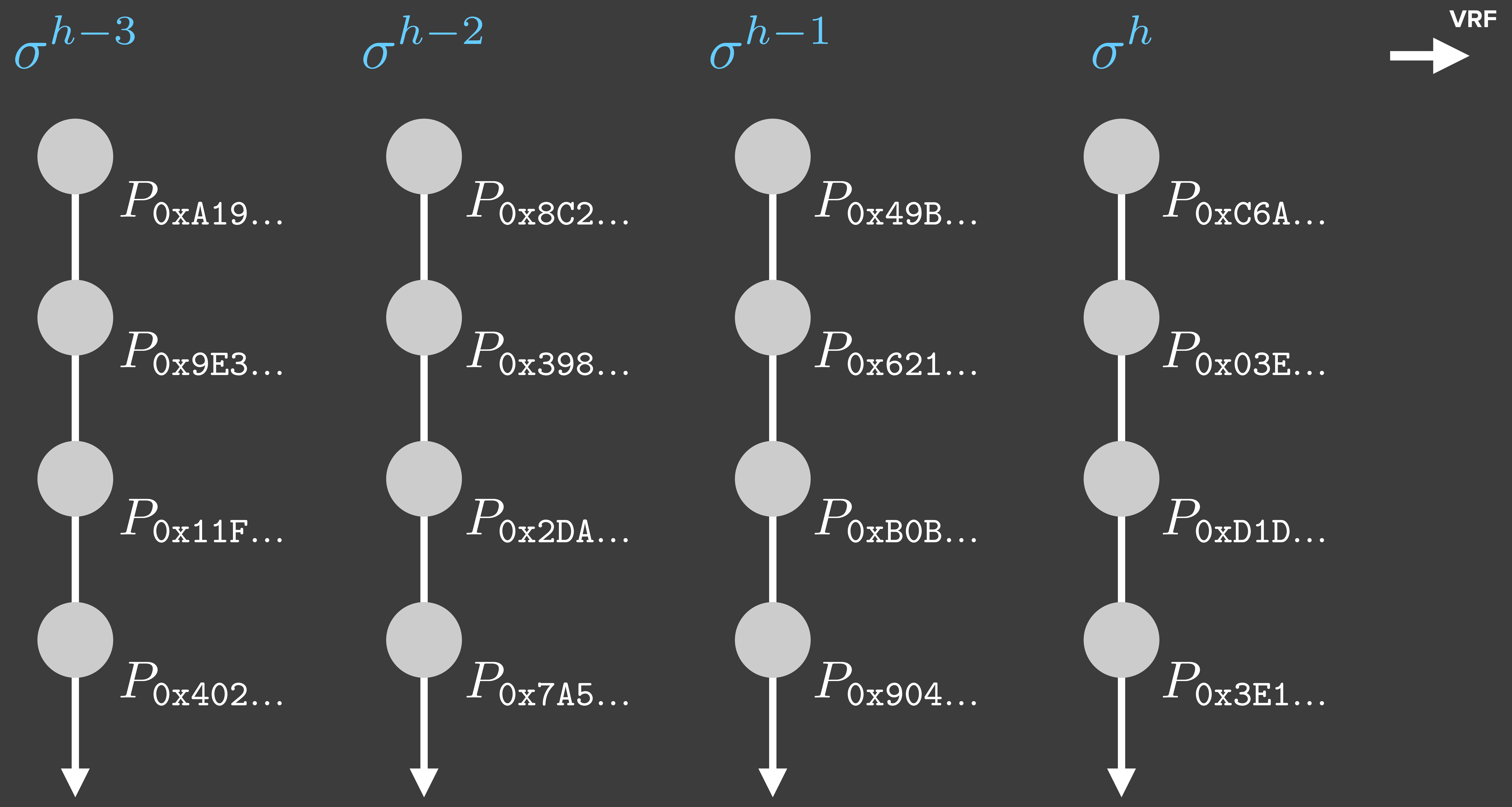
At each height, the randomness orders the processes...



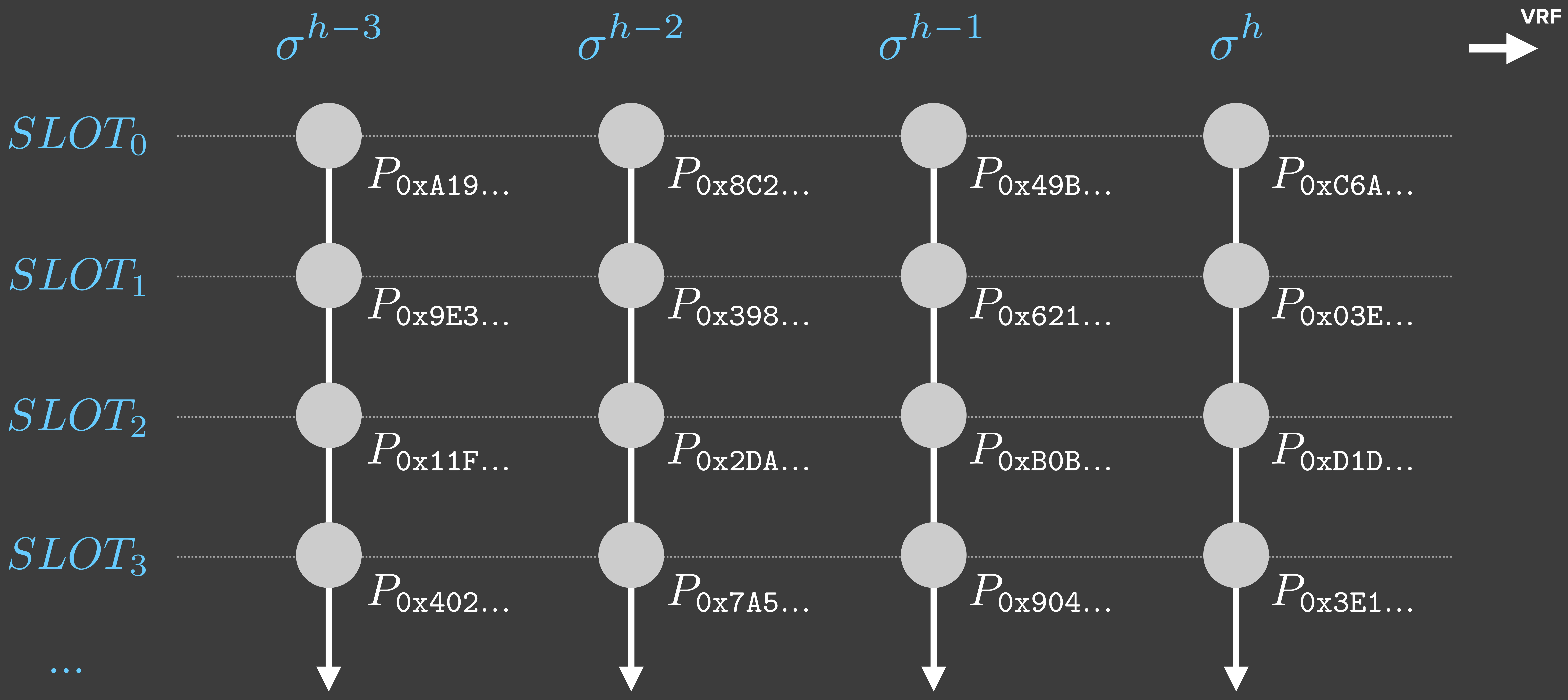
At each height, the randomness orders the processes...



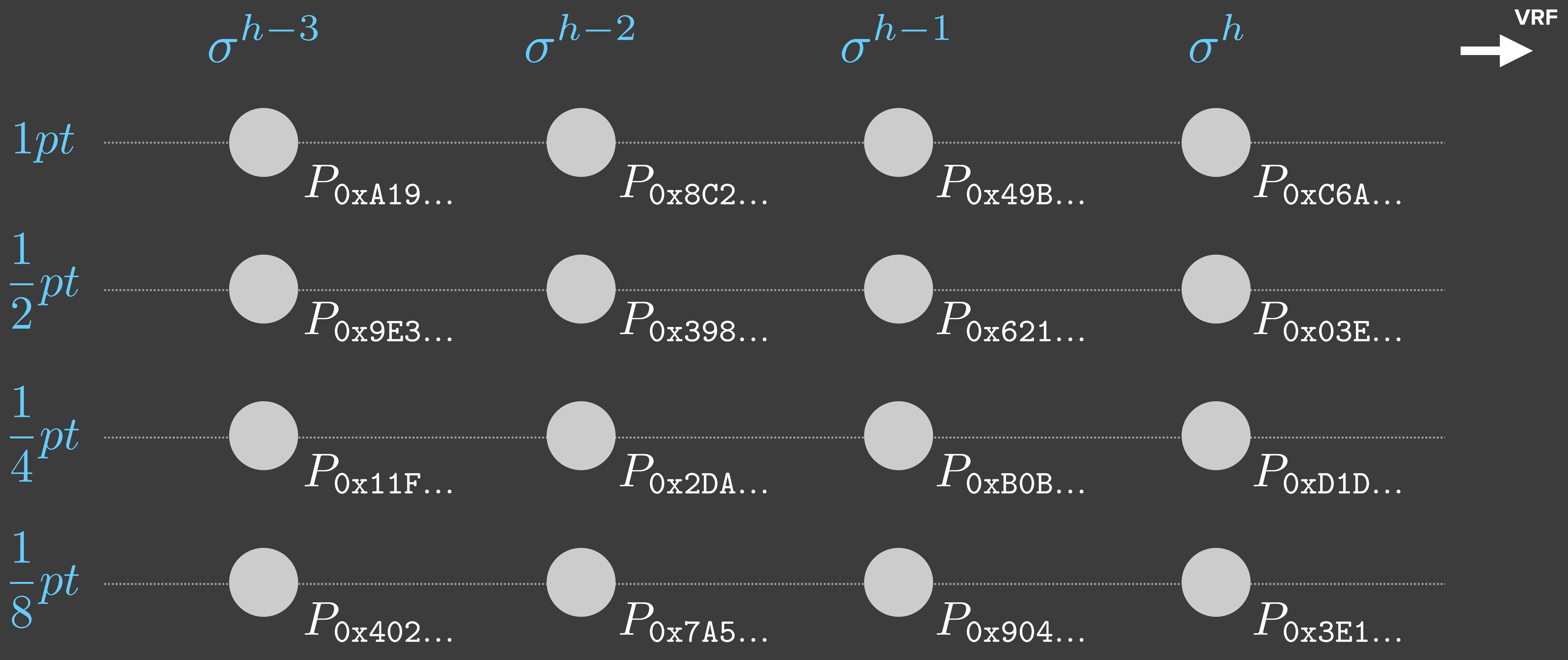
At each height, the randomness orders the processes...



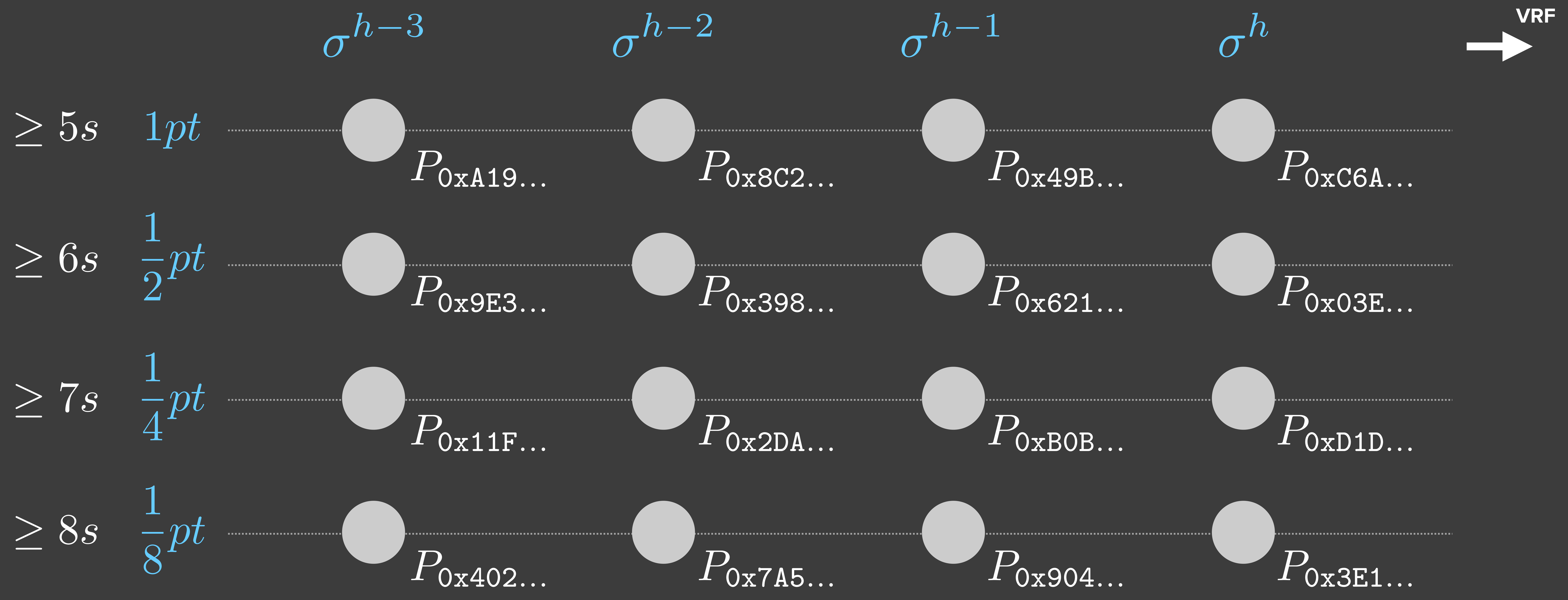
Indexes are priority “slots” for forging (zero highest)



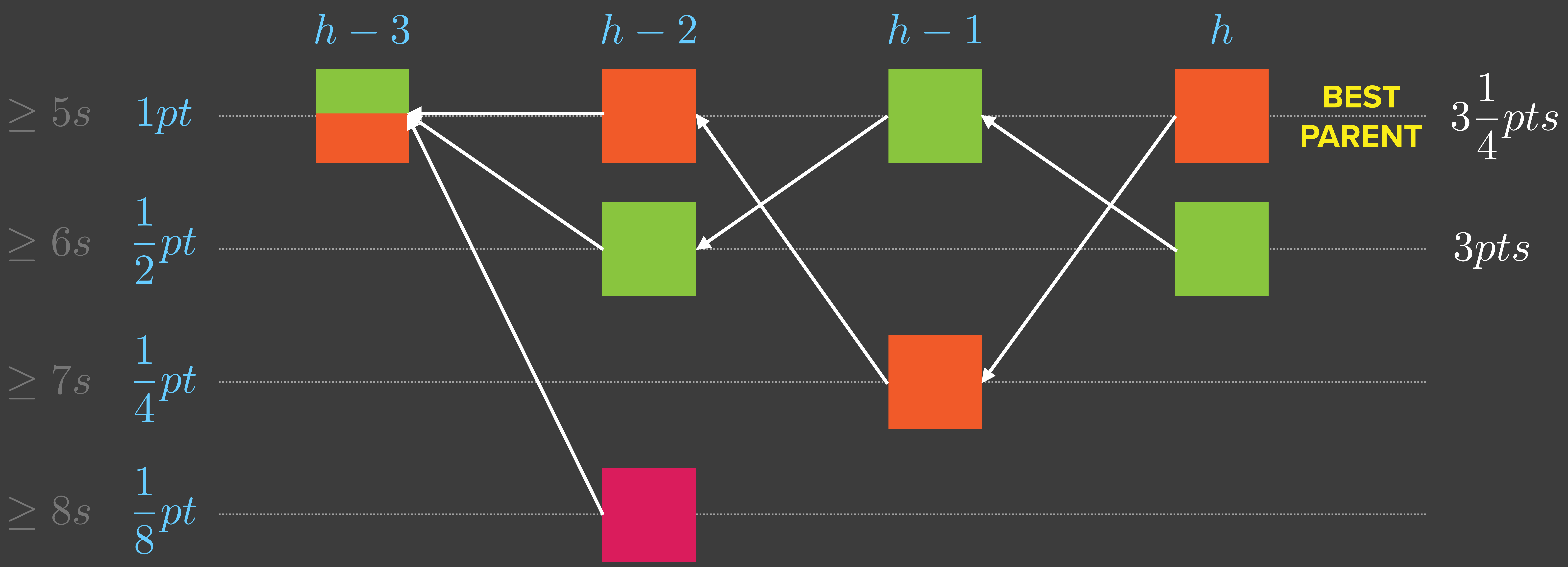
Value of candidate blocks scored by author's slot...



Can also introduce block relay rules, e.g. delays



We can create & score blockchains that converge



Very nice. But usual limitations. O no...

SELFISH MINING ATTACKS

The adversary can withhold blocks to gain an advantage over honest processes.

Selfish mining attacks increase the confirmations necessary for finality.

NOTHING AT STAKE

The adversary can go back in time and create forks from below *h* to Double Spend.

He only needs to be lucky and be granted a sequence of zero slots.

Solution?

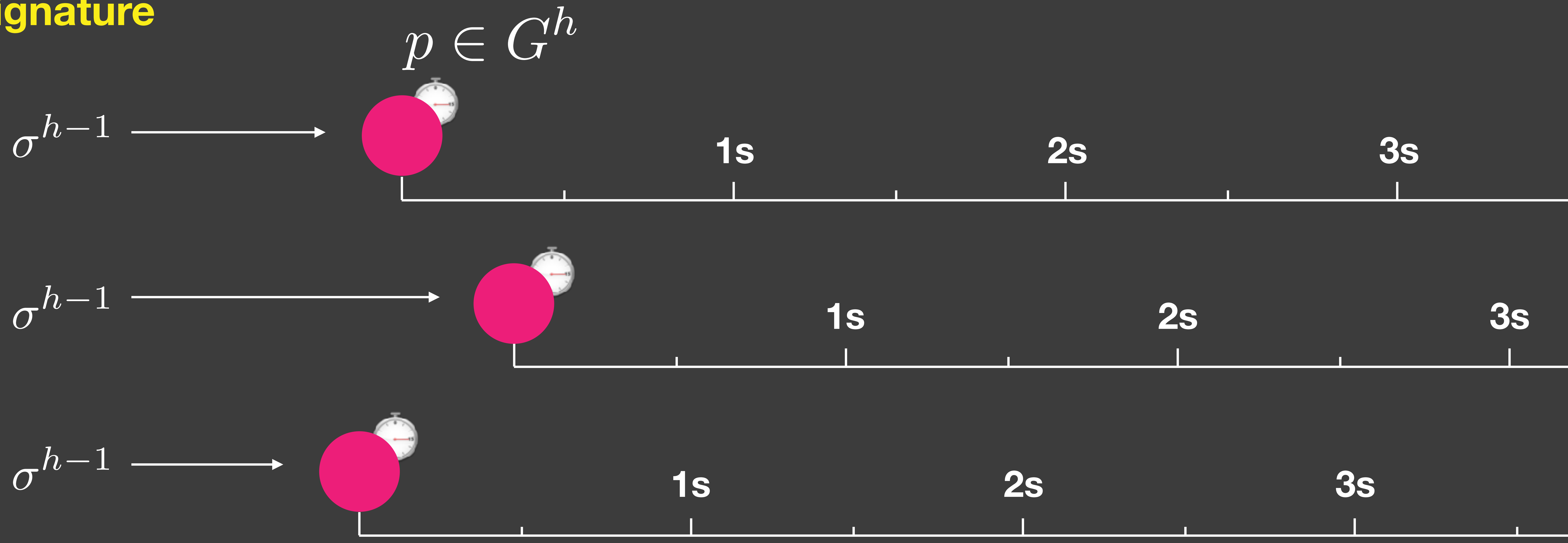
Threshold groups “notarize” (sign) at least one block at their height before relaying...

A valid block proposed at *h* must reference a block that was notarized at *h-1*

Thus, blocks must be published in good time or have no chance of notarization

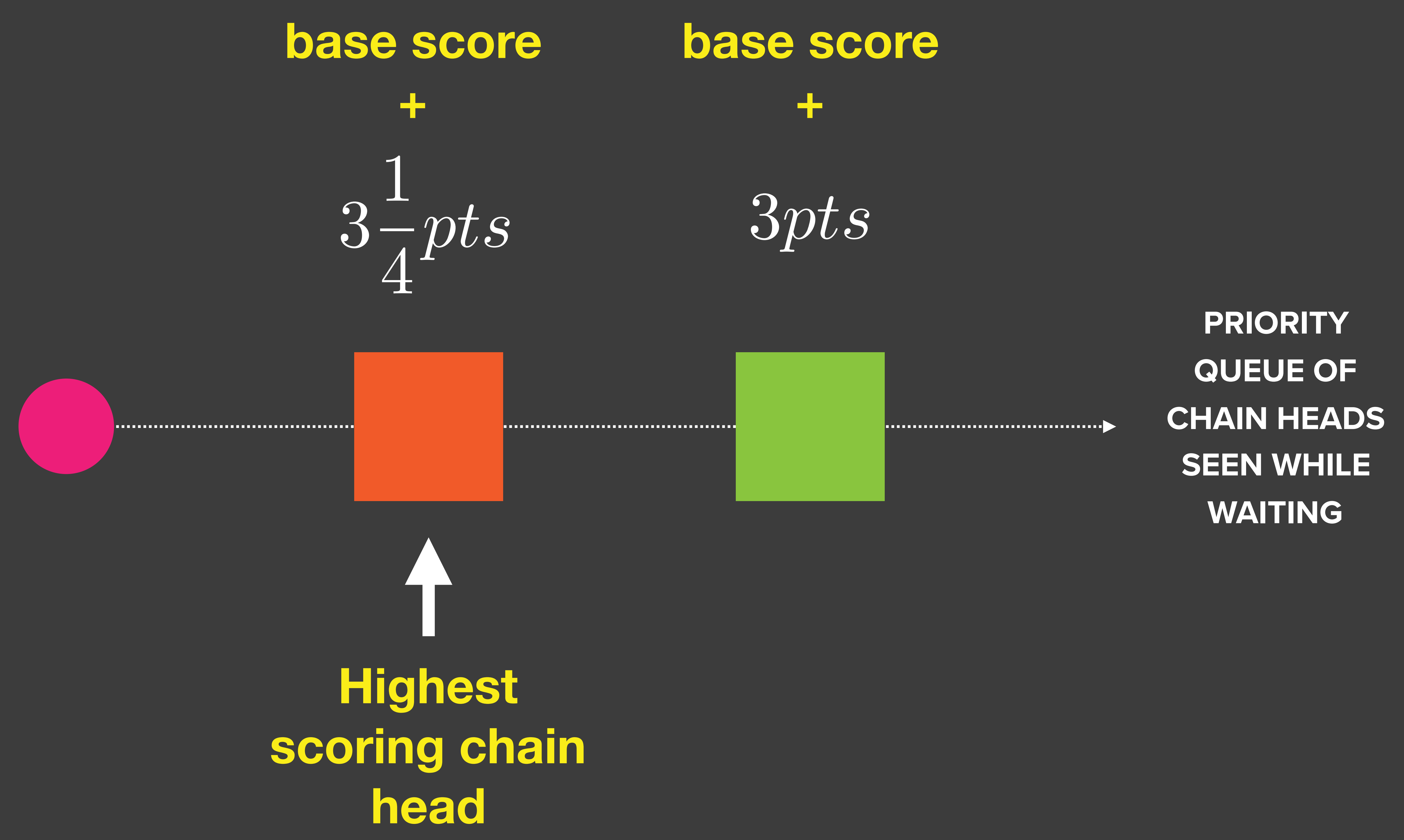
When group selected, its members start their timers...

Triggered by propagation threshold signature



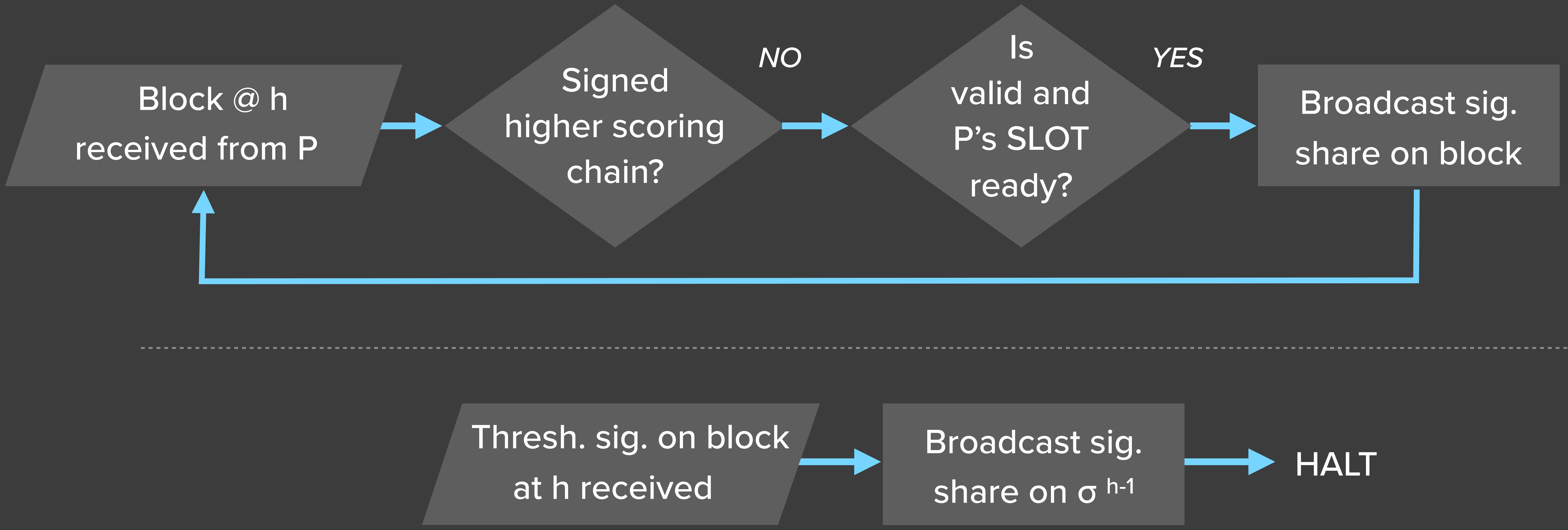
Members start processing blocks after expiry BLOCK_TIME. Clocks will be slightly out-of-sync, but that's OK!

Queue blocks score order while waiting BLOCK_TIME



When BLOCK_TIME expires, witness by notarizing...

Group members sign until ≥1 blocks receive threshold signature



Sign the best blocks seen

Stop signing, relay and halt

An important observation

In normal operation, if BLOCK_TIME is sufficiently large considering network synchrony, each group member will remove from its queue and process the highest scoring chain head first...

Consequently, the group will ONLY witness (notarize/sign) the block representing the highest scoring chain head

This prevents and immediately collapses forks in normal operation driving extremely high consistency and rapid finality

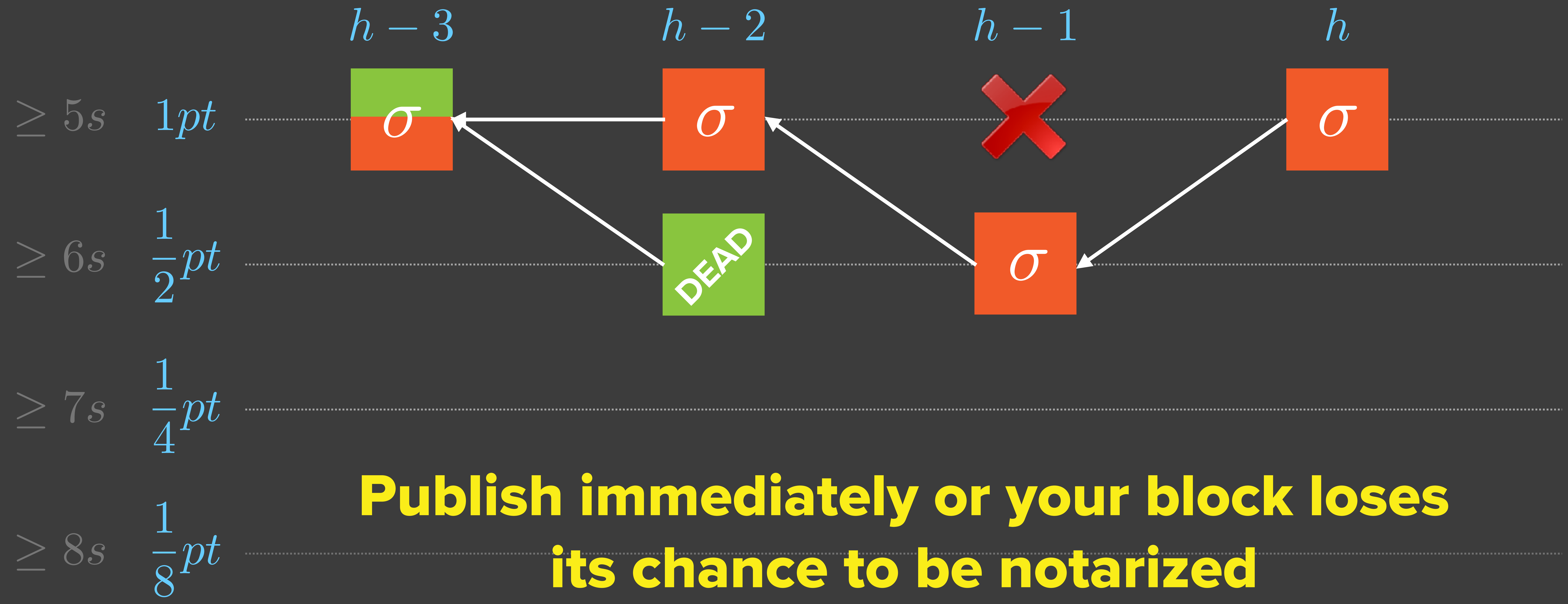
TLDR; tweaking to address the threat of equivocation

A faulty process in SLOT 0 controlled by an adversary might wish to broadcast vast numbers different versions of its block to DOS…

Of course, this faulty process will later be expelled for its provably Byzantine actions, but why provide room for misbehavior…

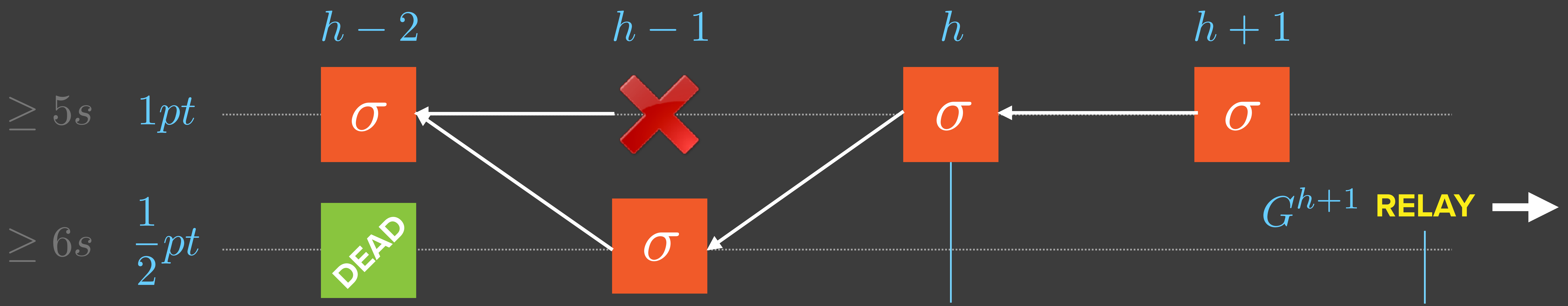
SOLUTION if process sees equivocated highest scoring block(s), only forward to peers that haven't detected equivocation yet. If group member sees equivocated highest scoring block, don't sign it, and instead start signing next highest scoring block seen when from a *different* slot

Fair mining, super high consistency and rapid finality



Publish immediately or your block loses its chance to be notarized and included....

Optimal case. Overwhelming finality in 2 blocks + relay



No alternative chain head
or even partially signed chain head is visible. Yet, for a viable chain head to exist, it must have been shared with some correct processes to collect signatures, and they would have propagated (broadcast) it…

The trap shuts!
Now group ***h+1*** has relayed it will not notarize/sign any more blocks. Too late for any alternative chain head at ***h*** to "appear" and get notarized…

Gains from Notarization

Fast Optimal Avg. Finality

$$BLOCK\_TIME = 5s$$

$$\Longrightarrow$$

$$7.5s$$

Addresses Key Challenges

- Selfish Mining
- Nothing At Stake
- Equivocation

Quantifiable finality

Hooks make possible calculate probabilities more meaningfully

SPV

Light client needs only Merkle root of groups

Relative Performance Copper Release

	Bitcoin	ethereum	DFINITY
Block Time	Average 10 mins *varies wildly*	Average 20 secs *varies wildly*	Average 5 secs *low variance*
“TX finality” (speed)	6 confirmations avg. 1 hr	37 confirmations avg. 10 mins	2 confirmations+relay ***avg. 7.5 secs*** ***Optimal case normal operation***
Gas available	- - -	Low due to Poisson distribution	**50X+ Ethereum** ***Unlimited scale-out achieved by applying randomness in following techniques…***

Miscellanea

Death By Poisson Process

The Simplest Flaws Are The Worst…

50% of Ethereum blocks are empty !

Miners prefer to build on empty blocks
since no need validate/delay
= more profitable

An empty block has more chance being confirmed….

Duh !

Bitcoin Could Consume as Much Electricity as Denmark by 2020, Motherboard 3/29/2016

Separate and decouple concerns

Proof-of-Work Blockchain

Sybil resistance
Validation
State storage
Consensus

DFINITY

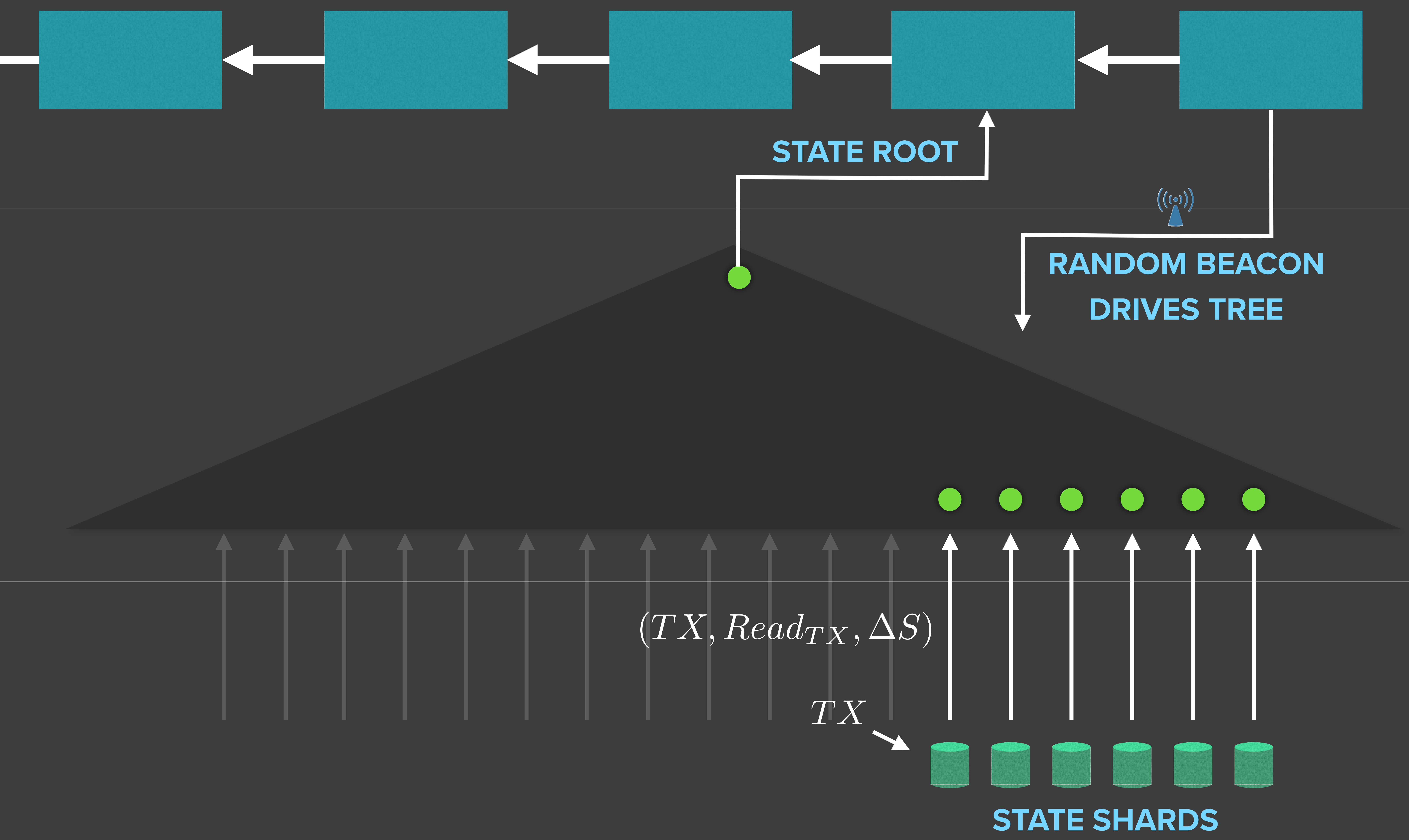
Consensus	Sybil resistance
Validation	
State storage	

Computer Science should not go out of fashion

TCP/IP

Application
Transport
Internet
Network Access

“Scale-out” using 3-layer architecture



CONSENSUS

Threshold Relay chain generates randomness and records network metadata and Validation Tree “state root”.

VALIDATION

Asynchronous “Validation Tree” composed “Validation Towers”. Does for state validation what Merkle tree does for data.

STORAGE

State and updates to state stored on shards. State transitions passed to Validation Tree.

Near Term Client Releases

1 COPPER

- Threshold Relay + PSP
- Blockchain Nervous System (BNS)
- Security deposits
- State-root-only-chain (transaction logging not necessary)

2 ZINC

- Special features enabling creation robust *and* high performance private networks using unlimited host computers
- Single *atomic* call from smart contract on private cloud into smart contract on public cloud network

3 TUNGSTEN

- State sharding (basic)
- Validation Towers (basic)
- Asynchronous model for cross-shard programming
- USCIDs (Unique State Copy IDs)
- Advancements in BNS

President/Chief Scientist DFINITY Stiftung

President/CTO String Labs

http:// twitter.com /dominic_williams

http:// linkedin.com /in/thedwilliams/