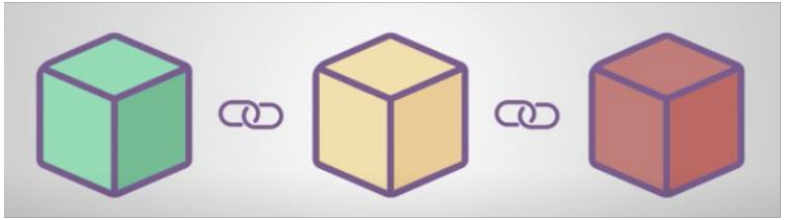
Elastico - A Secure Sharding Protocol for Open Blockchains

Published by Loi Luu, Viswesh Narayanan, Chaodong Zheng, Kunal Baweja, Seth Gilbert and Prateek Saxena

Presented by Manisha Sri Suresh, Pon Izhanathi Sundara Arumuganathan, Chalmuri Mayuri Naidu, Jaewan Yun and Saif Lakhani

UCDAVIS
COMPUTER SCIENCE

BLOCKCHAINS



Nakamoto Consensus, 2009

- Maintains distributed database in a decentralized network
- Blockchain agreement protocol

  - periodically agree on new block of data

bitcoin

ethereum

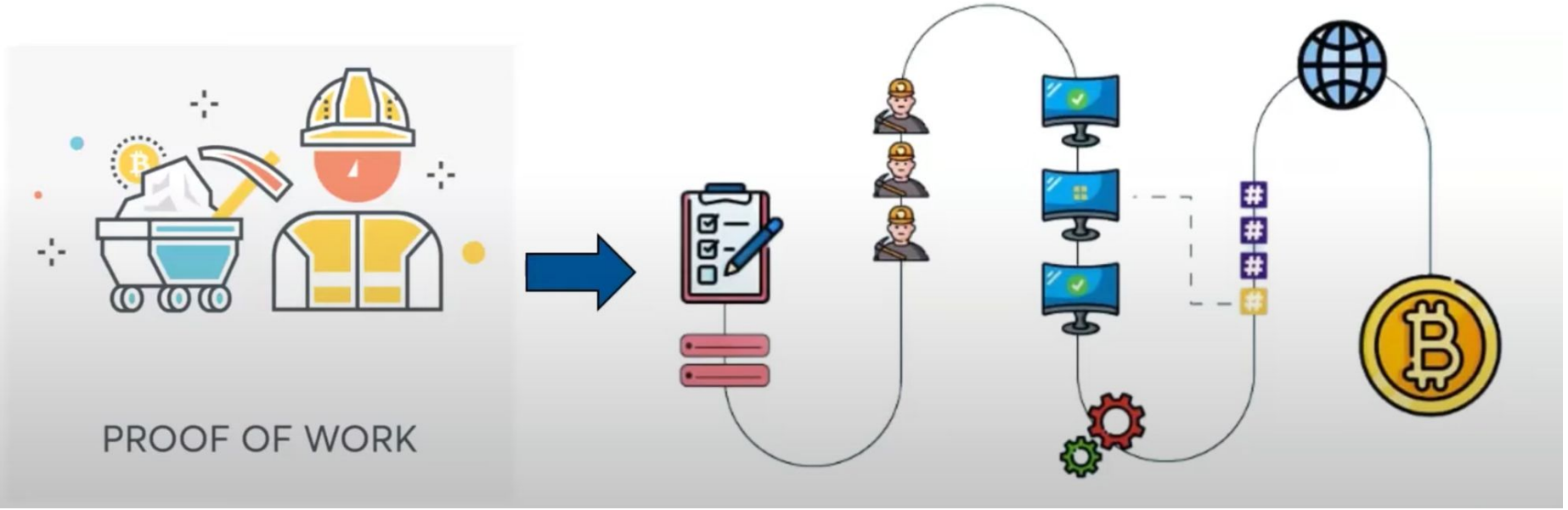
BLOCKCHAIN AGREEMENT PROBLEM :

- Large network of several processors to agree on the blockchain state
- No inherent identity
- No PKI

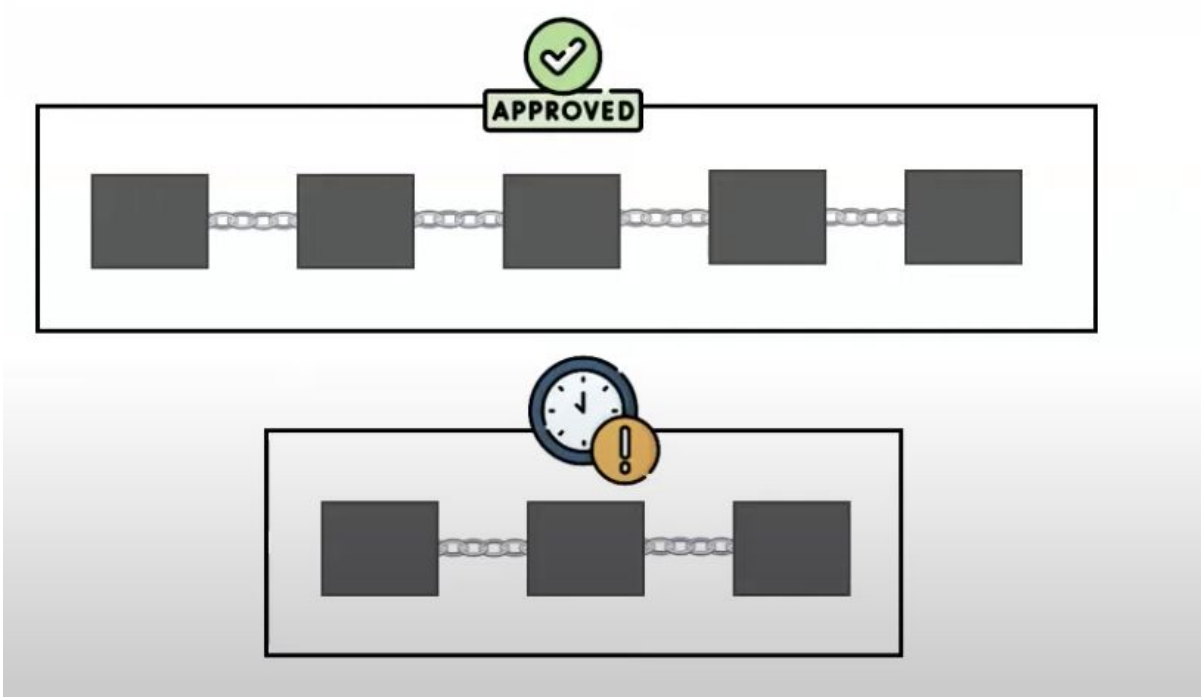
UCDAVIS COMPUTER SCIENCE

PROOF OF WORK (PoW)

Proof of work (PoW) is a decentralized consensus method that requires network participants to spend time-solving an arbitrary mathematical puzzle

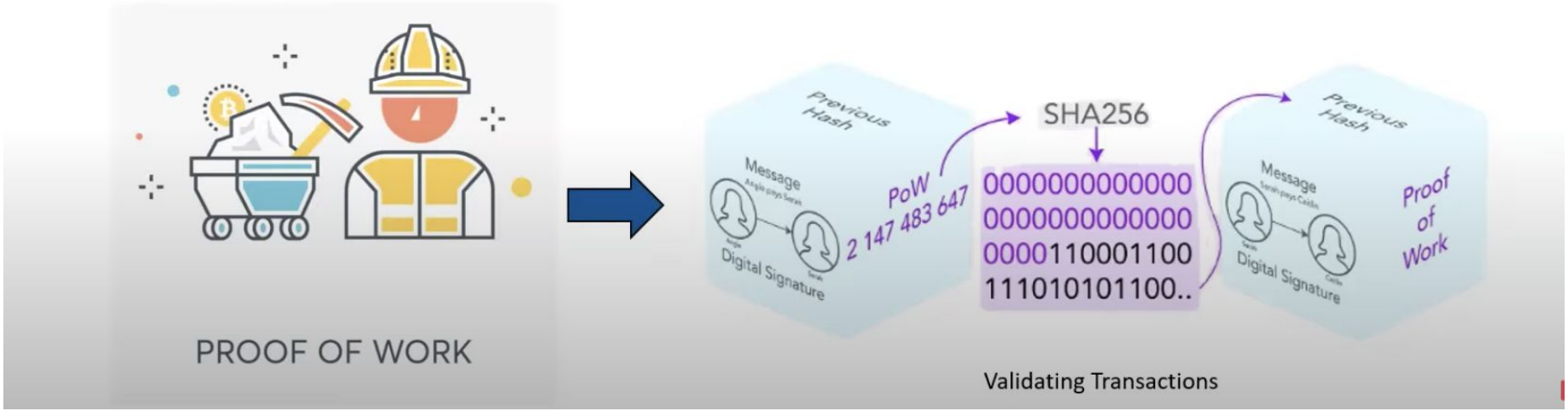


The protocol accepts the longest chain as valid



Benefits of PoW

Defense from malicious attacks



UCDAVIS COMPUTER SCIENCE

Issues with PoW

It has a negative impact on the environment.

The 51 percent attack can be used against it,

SCALABILITY ISSUE

3–7 TXs/second

12–30 TXs/second

Demand from practical application : 1,200 to 56,000 TXs/second

Classical Byzantine Consensus Protocols

Known identity set

- pre- established identities
- PKI

Bandwidth limited

- O(n^2) messages (e.g. PBFT)
- Work for a small network (e.g. n < 1000)

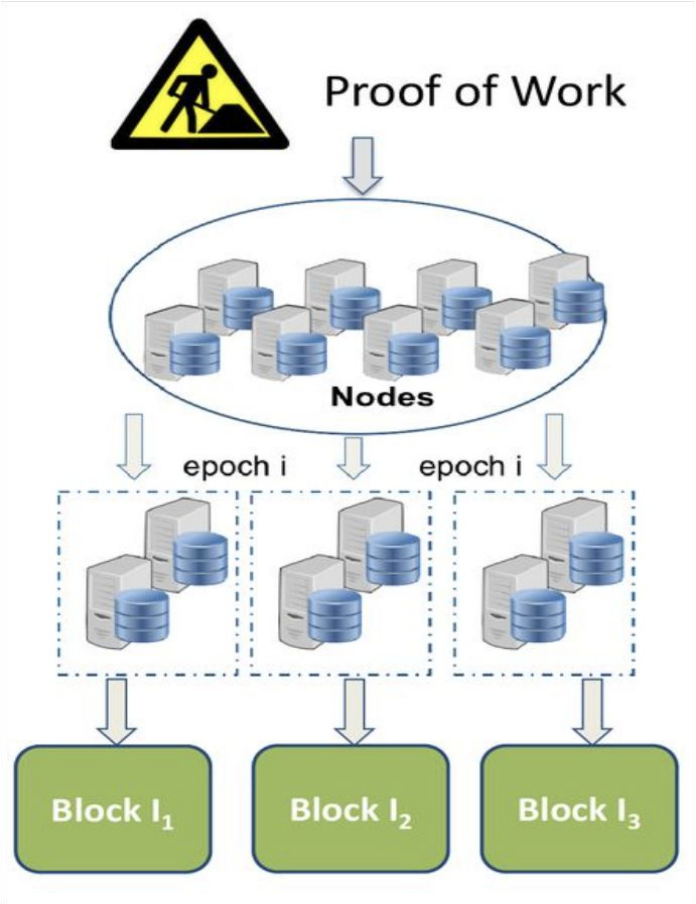
ELASTICO

- Sharding protocol
- No pre-established identity
- Throughput scales

SHARDING CONCEPT

•Partitioning of the set of transactions into so-called “shards”, where each shard is responsible for processing only part of the data stored in the network.

•Significant reduction in processing time for blockchain scalability



PROBLEM DEFINITION

Protocol runs between the processor such that the following conditions hold :

- Agreement
- Validity
- Scalability
- Efficiency

UCDAVIS
COMPUTER SCIENCE

ASSUMPTIONS

- Synchronous network
    - Bounded delay from a node to all other nodes
- At most 1/4 computation power is controlled by adversary
- Processors have equal computation power

ELASTICO DESIGN

- Process Divided into 5 Steps:
  - Identity Establishment and Committee Formation
  - Overlay setup for Committees
  - Intra Committee consensus
  - Final Consensus Broadcast
  - Epoch Randomness Generation

Identity Setup and Committee Formation

n = total number of nodes in a network

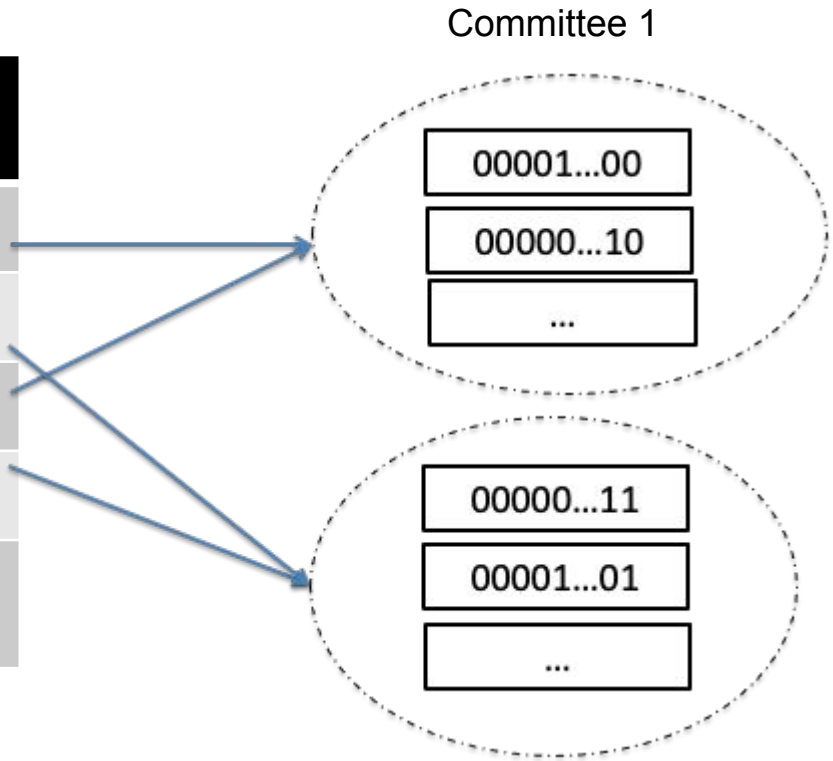
c = number of nodes in each committee

k = total number of committee in a network

s = no. of bits we consider from a POW solution for committee allocation

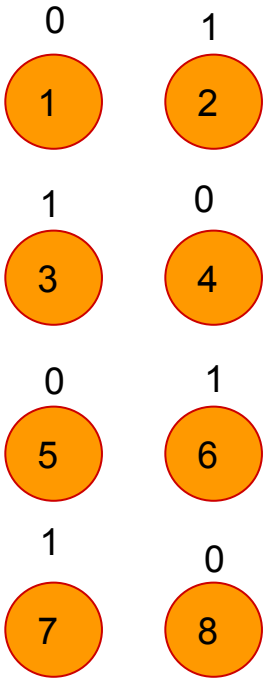
- Lets consider:
  - c = 4
  - k = 2 (2^s)
  - s = 1 bit (0 or 1)
  - n = c*k = 8 nodes

ID	PoW
1	00001…0
2	00000…1
3	00000…0
4	00001…1
..	…



UCDAVIS COMPUTER SCIENCE

Committee Formation



Solve Proof of Work

=

...1010011**0/1**

ID: 0/1

Directory Committee (DC)

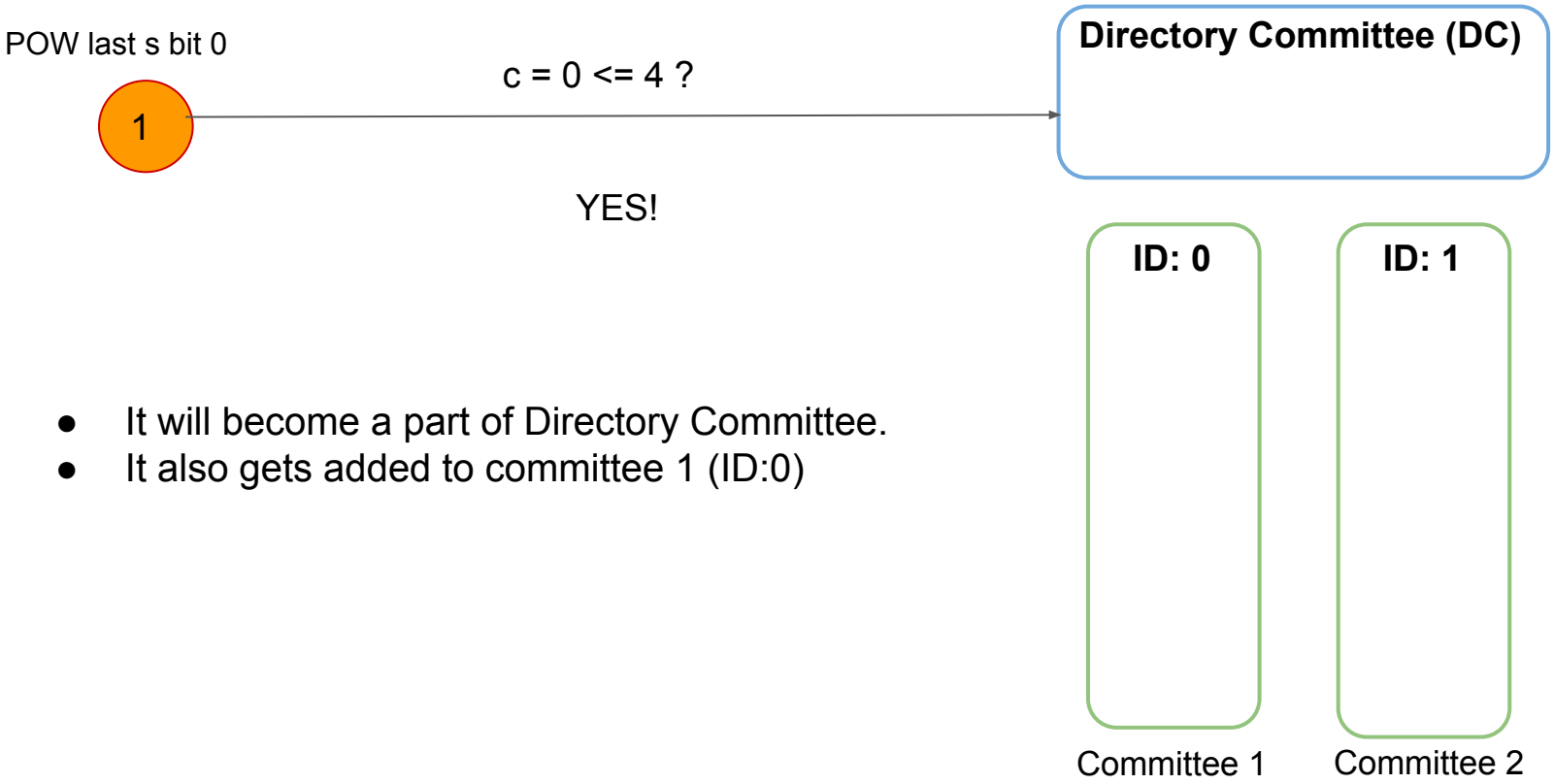
ID: 0

ID: 1

Committee 1

Committee 2

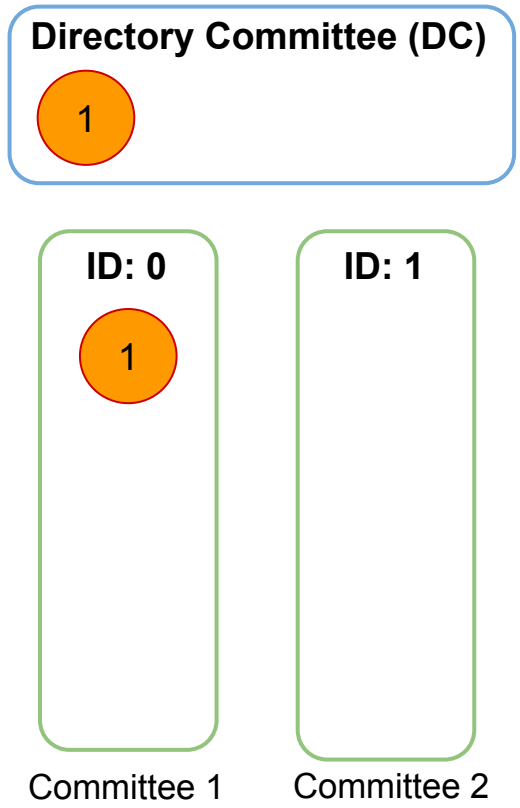
UCDAVIS COMPUTER SCIENCE



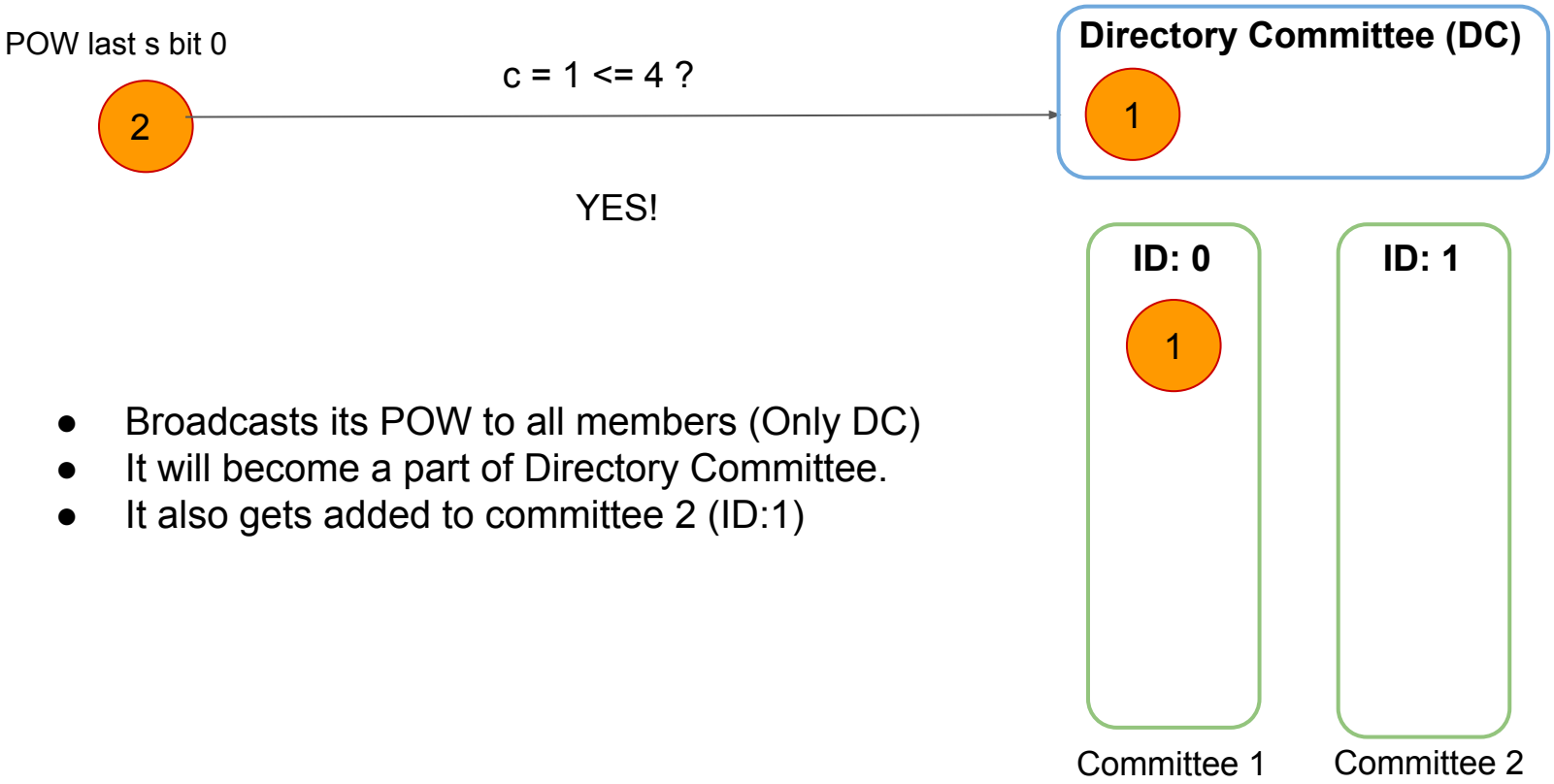
- It will become a part of Directory Committee.
- It also gets added to committee 1 (ID:0)

View Of Node 1

Node	Committee
1	1, DC



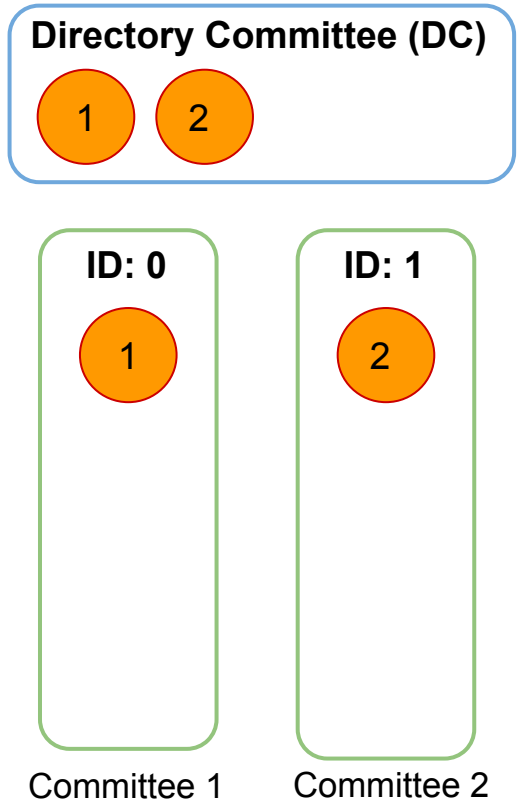
UCDAVIS
COMPUTER SCIENCE

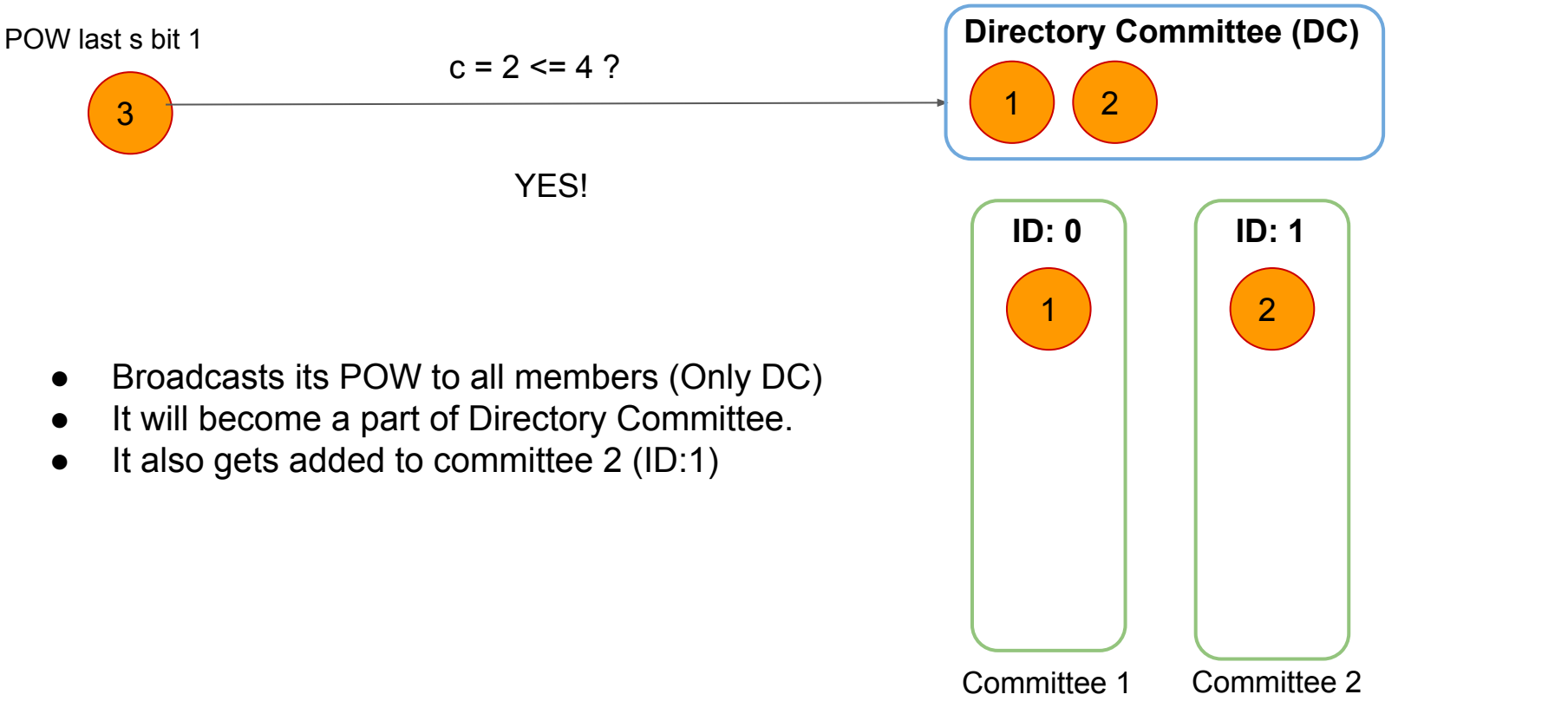


- Broadcasts its POW to all members (Only DC)
- It will become a part of Directory Committee.
- It also gets added to committee 2 (ID:1)

View Of Node 1	
Node	Committee
1	1, DC
2	2, DC

View Of Node 2	
Node	Committee
1	1, DC
2	2, DC





- Broadcasts its POW to all members (Only DC)
- It will become a part of Directory Committee.
- It also gets added to committee 2 (ID:1)

View Of Node 1

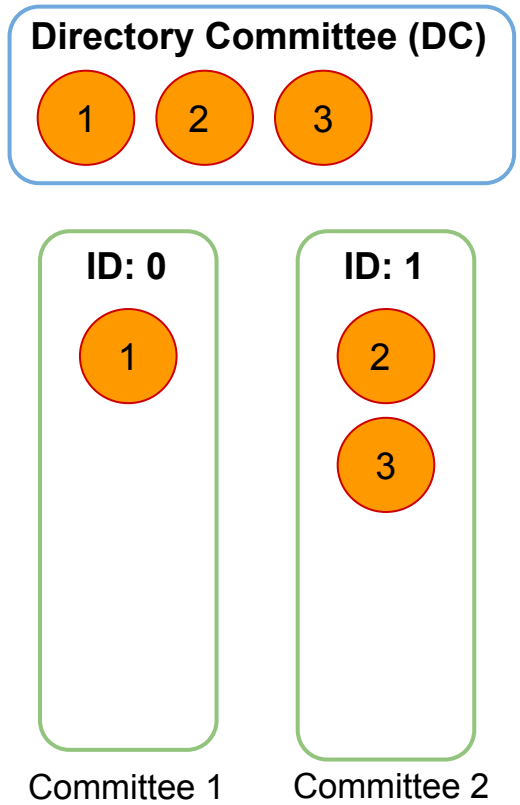
Node	Committee
1	1, DC
2	2, DC
3	2, DC

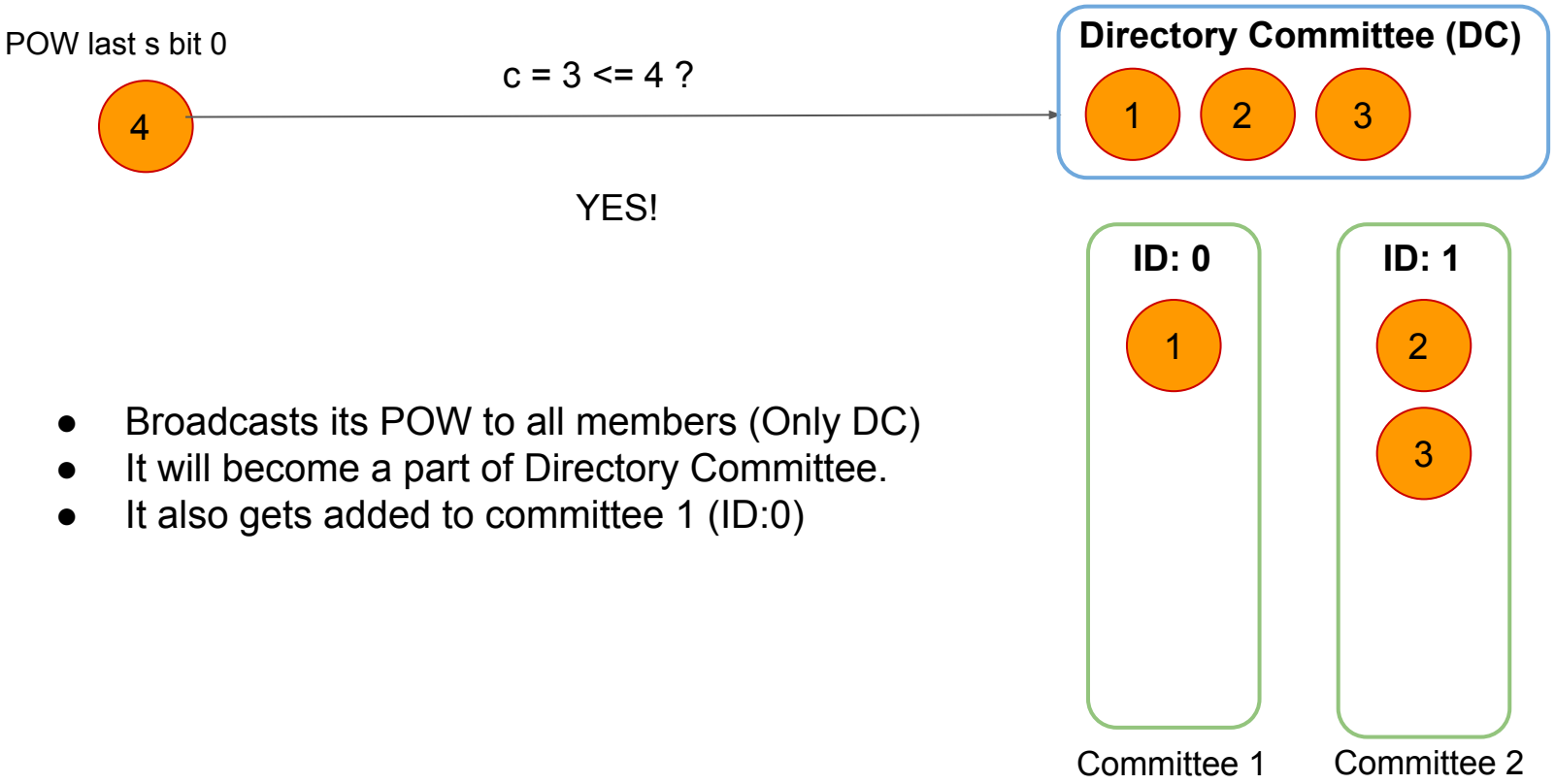
View Of Node 2

Node	Committee
1	1, DC
2	2, DC
3	2, DC

View Of Node 3

Node	Committee
1	1, DC
2	2, DC
3	2, DC





- Broadcasts its POW to all members (Only DC)
- It will become a part of Directory Committee.
- It also gets added to committee 1 (ID:0)

View Of Node 1

Replica	Committee
1	1, DC
2	2, DC
3	2, DC
4	1, DC

View Of Node 2

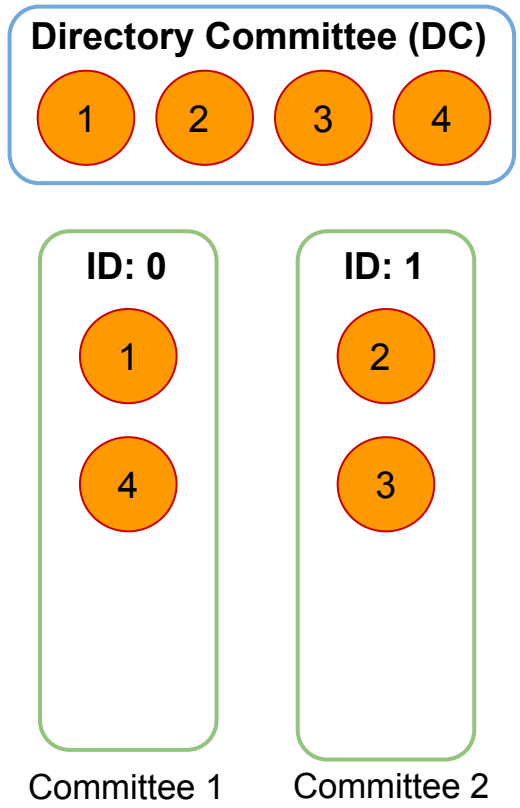
Replica	Committee
1	1, DC
2	2, DC
3	2, DC
4	1, DC

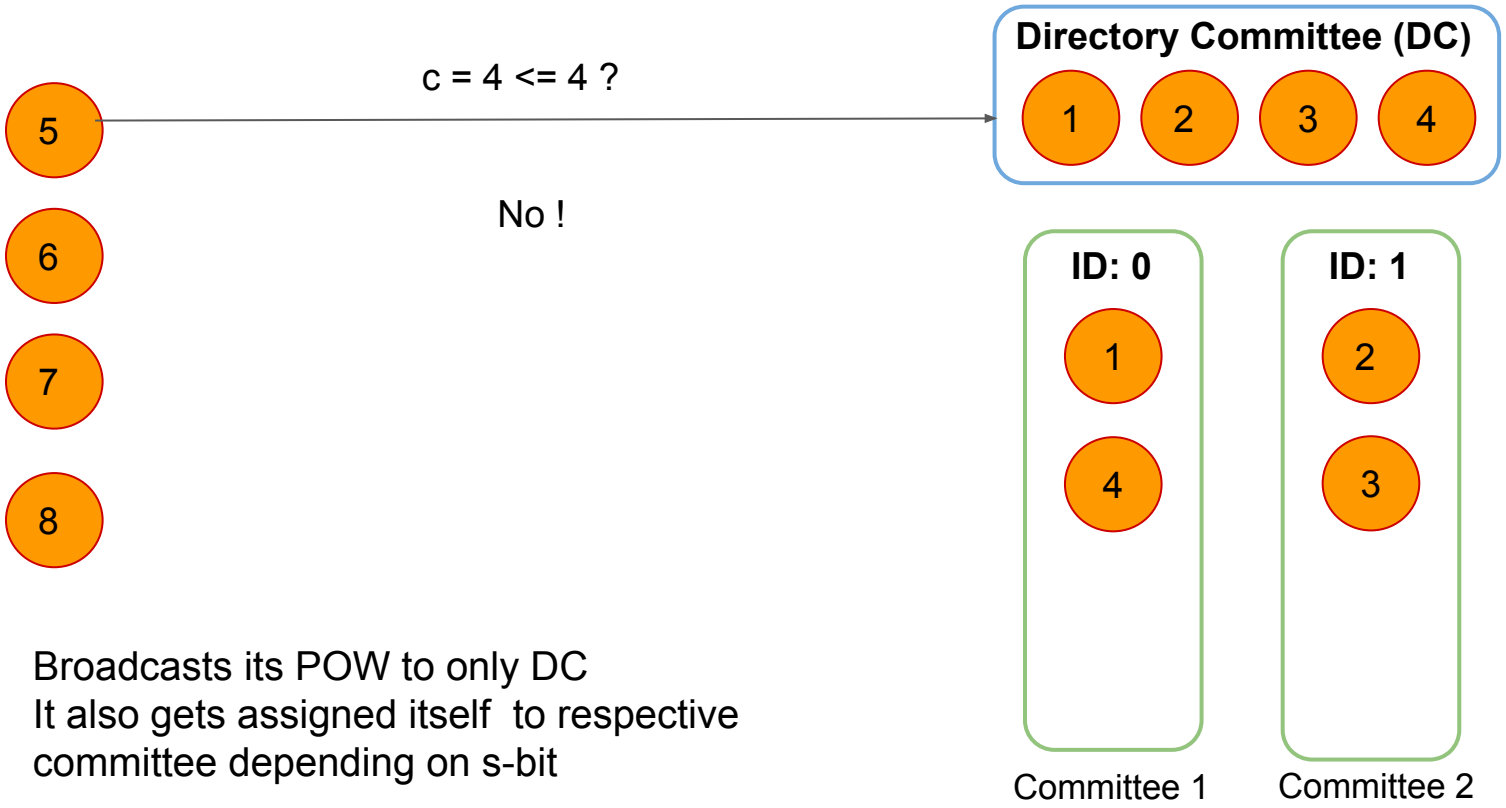
View Of Node 3

Replica	Committee
1	1, DC
2	2, DC
3	2, DC
4	1, DC

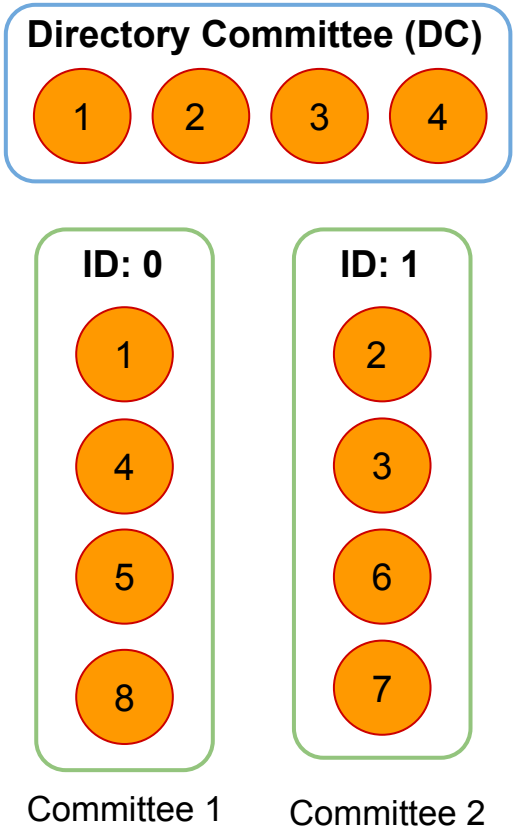
View Of Node 4

Replica	Committee
1	1, DC
2	2, DC
3	2, DC
4	1, DC



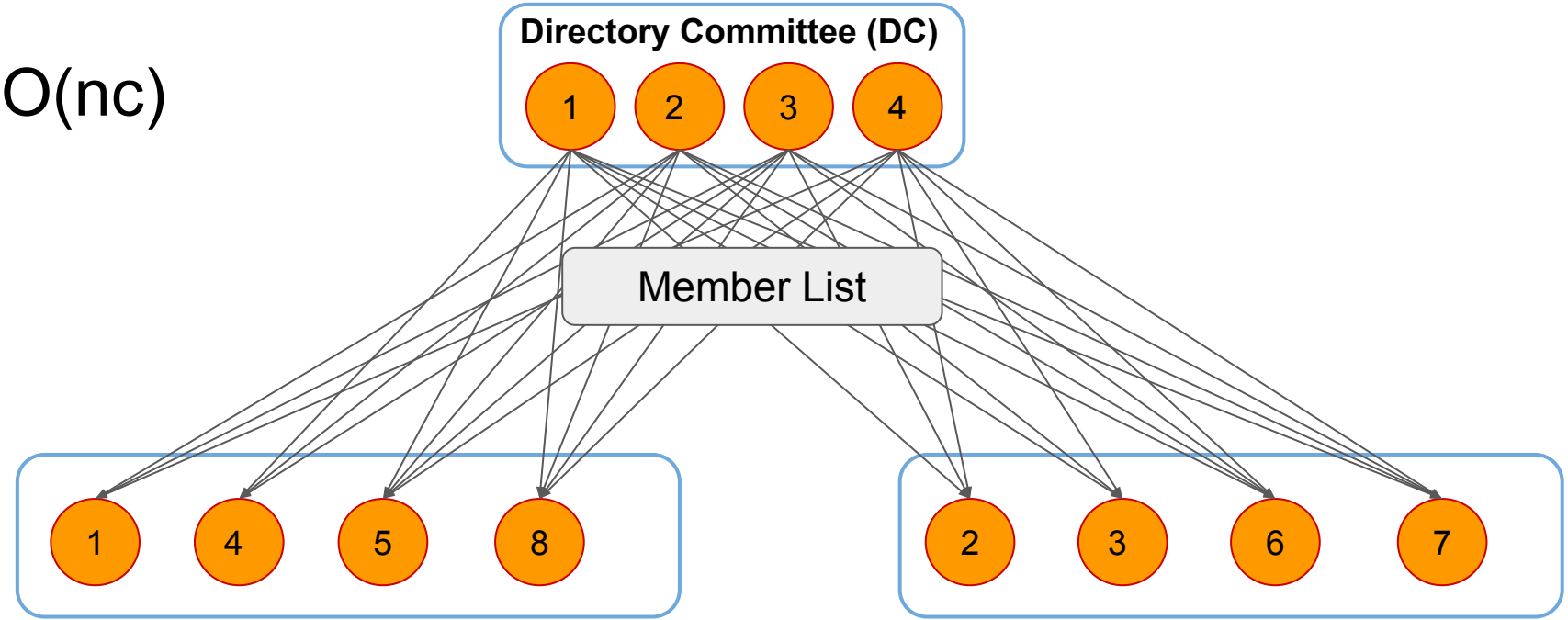


- Broadcasts its POW to only DC
- It also gets assigned itself to respective committee depending on s-bit



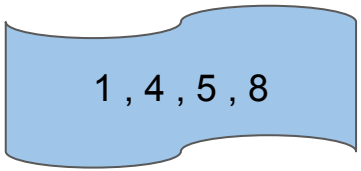
View Of Node 1		View Of Node 2		View Of Node 3		View Of Node 4	
Node	Committee	Node	Committee	Node	Committee	Node	Committee
1	1, DC	1	1, DC	1	1, DC	1	1, DC
2	2, DC	2	2, DC	2	2, DC	2	2, DC
3	2, DC	3	2, DC	3	2, DC	3	2, DC
4	1, DC	4	1, DC	4	1, DC	4	1, DC
5	1	5	1	5	1	5	1
6	2	6	2	6	2	6	2
7	2	7	2	7	2	7	2
8	1	8	1	8	1	8	1

O(nc)



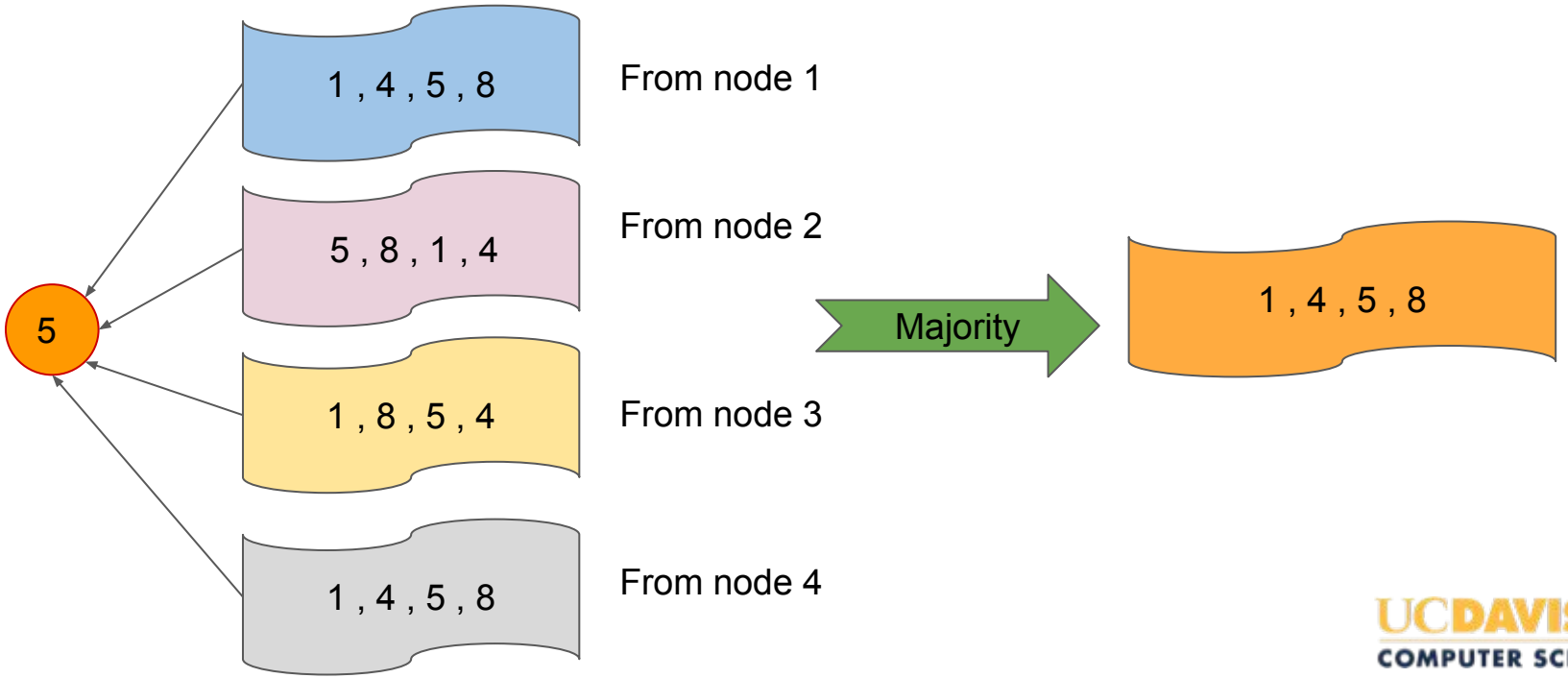
Committee 1

Committee 2



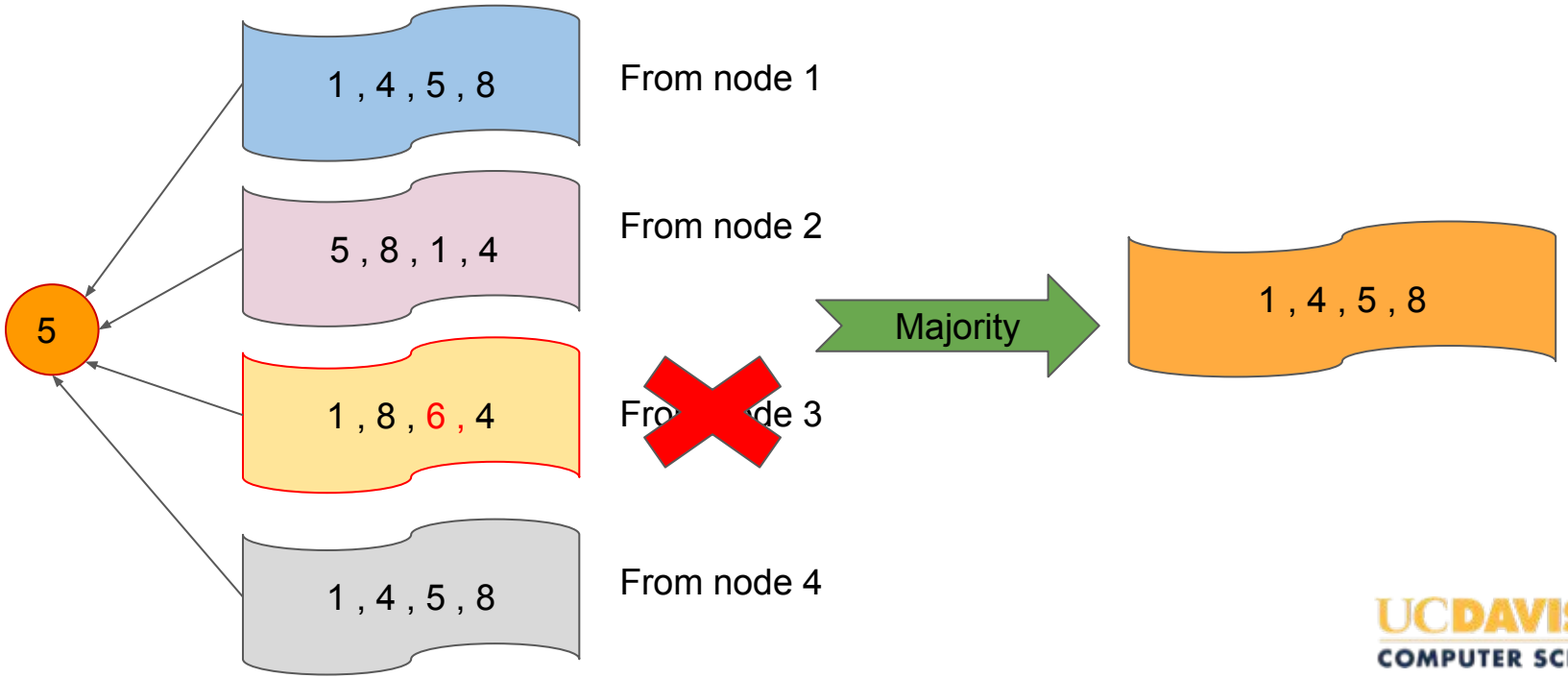
Identifying own committee members

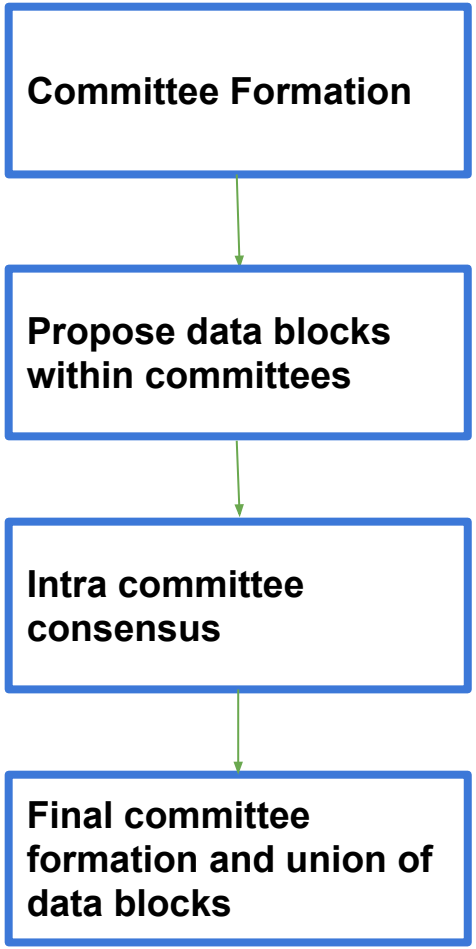
Case 1: No byzantine behaviours from DC



Identifying own committee members

Case 2: Some byzantine behaviours from DC





Propose a block within a committee

Transaction set included in data blocks are disjoint -

- Each data block consists of set of transactions with prefix similar to the block hash of the committee

When s-bits is 1, we have 2^s, i.e. $2^1 = 2$ committees

Block	Transaction IDs
Data Block 1	**0**100100….
Data Block 2	**1**101110….

When s-bits is 2, we have 2^s, i.e. $2^2 = 4$ committees

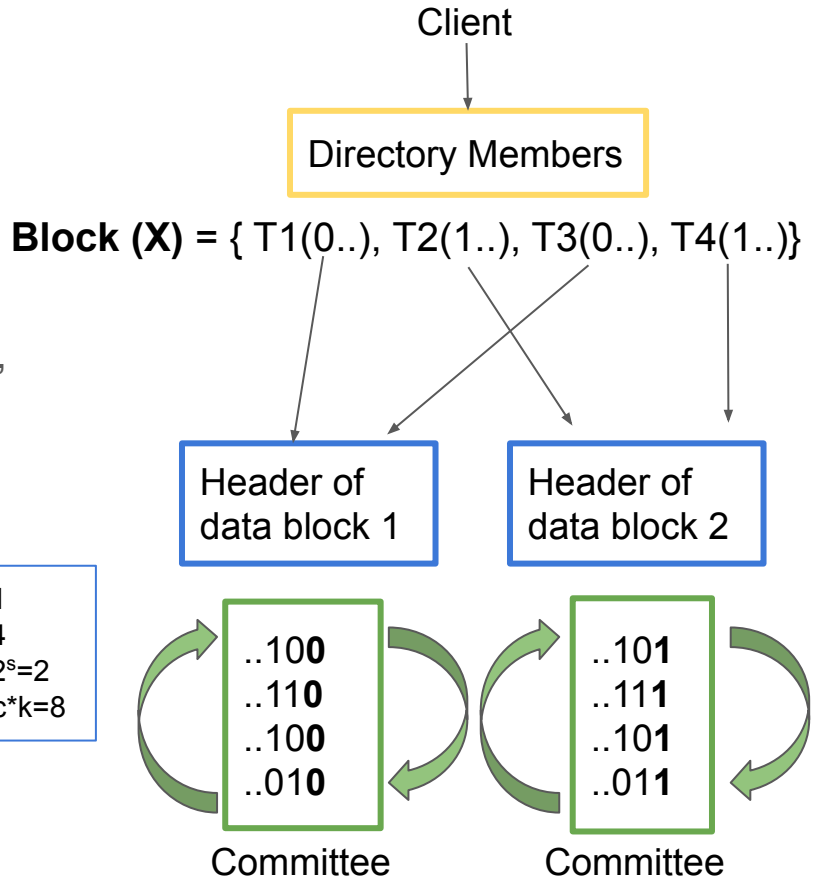
Block	Transaction IDs
Data Block 1	**00**100100….
Data Block 2	**01**101110….
Data Block 3	**10**100100….
Data Block 4	**11**101110….

UCDAVIS COMPUTER SCIENCE

Intra committee consensus

Run a classical Byzantine consensus protocol

- Header of each block contains block hash, committee members and signatures
- Members in a committee agree and sign on one valid data block
- No of messages exchanged ~ $o(c^2)$



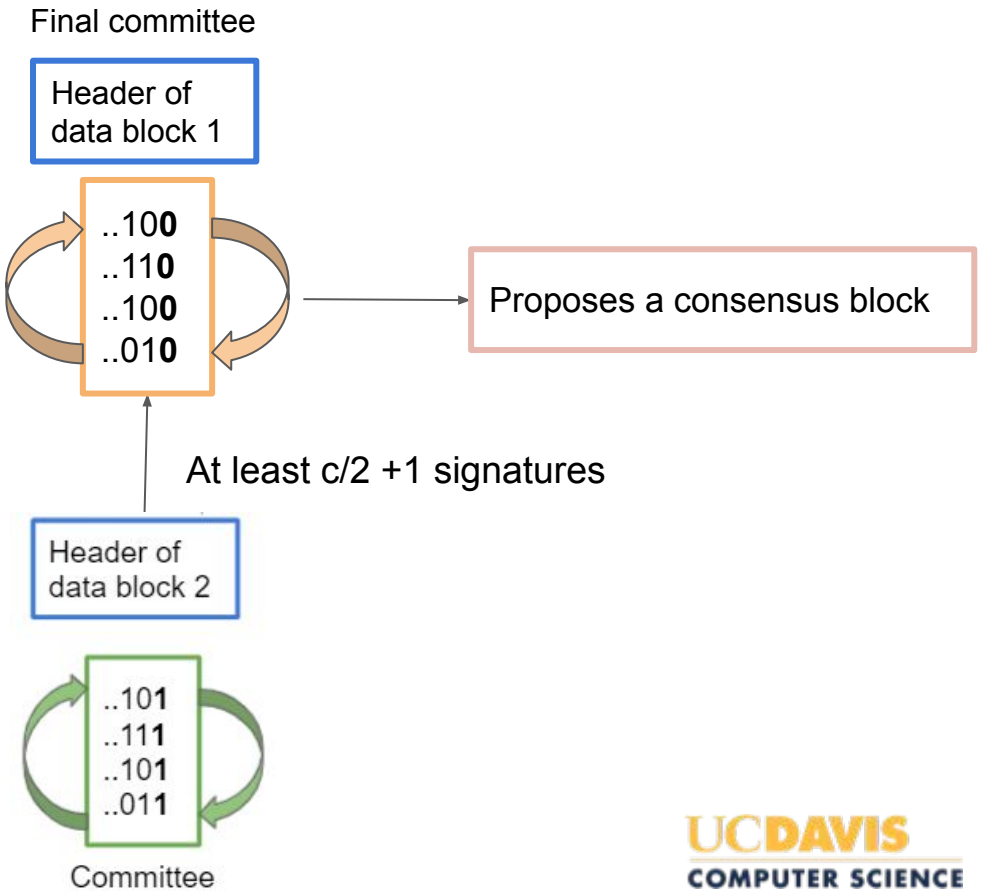
s=1
c=4
$k=2^s=2$
$n=c*k=8$

HEADER

Block hash	Committee members	Signatues

Final Committee unions all shard txns (Final consensus broadcast)

- Final committee is one of the existing committee, we assume it to be the first committee
- Creates union of transactions from all committees -> Runs a classical Byzantine consensus protocol



UCDAVIS COMPUTER SCIENCE

Transaction processing with 2 ‘s’ bits

Block (X) = { T1(00..), T2(01..), T3(00..), T4(01..), T5(10..), T6(11..), T7(10..), T8(11..) }

s=2
c=4
k=2^s=4
n=c*k=16

Header of data block **1** ‘block hash 00’

Header of data block **2** ‘block hash 01’

Header of data block **3** ‘block hash 10”

Header of data block **4** ‘block hash 11’

..001**00**
..010**00**
..101**00**
..011**00**

Committee

..001**01**
..010**01**
..101**01**
..011**01**

Committee

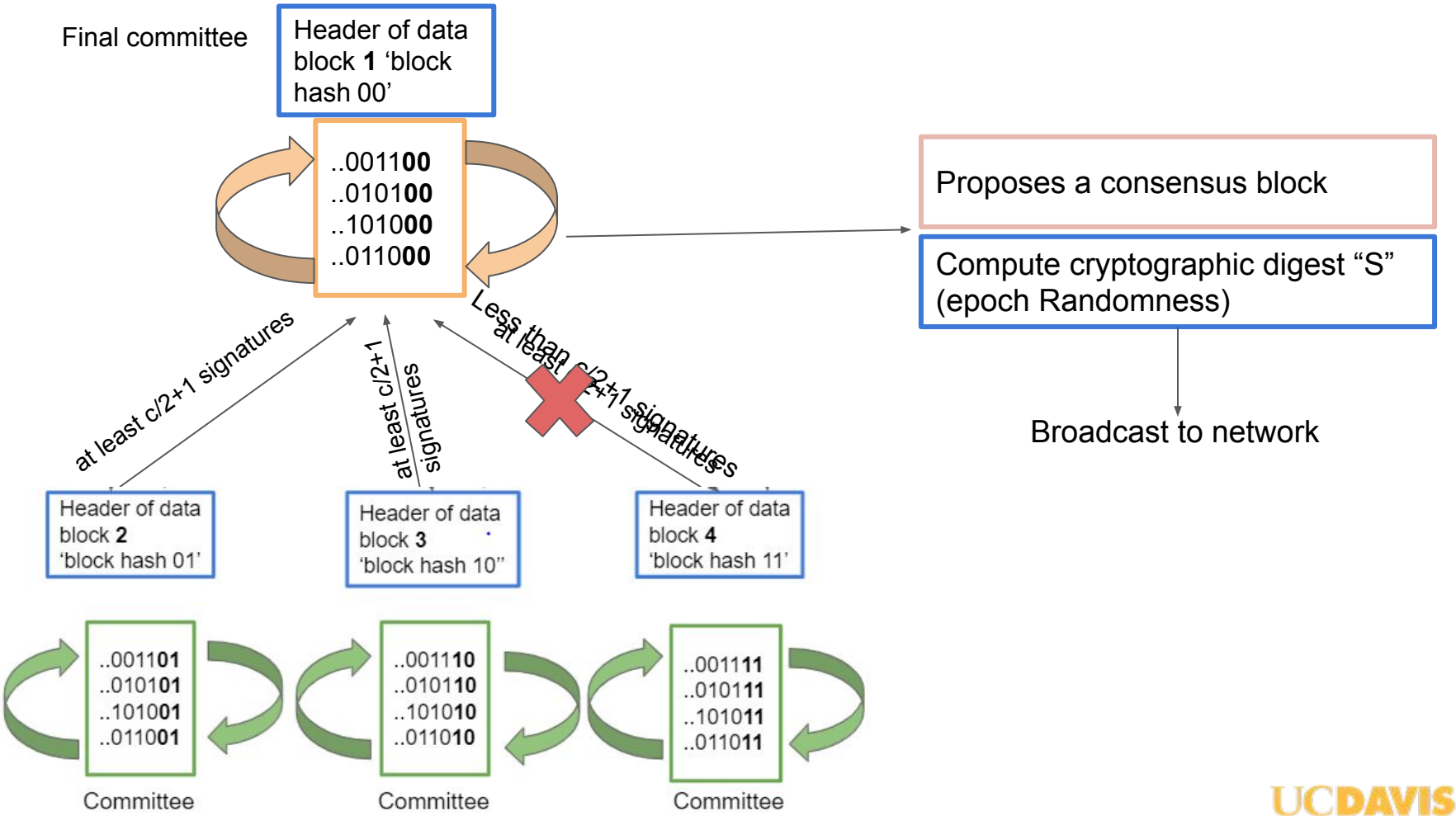
..001**10**
..010**10**
..101**10**
..011**10**

Committee

..001**11**
..010**11**
..101**11**
..011**11**

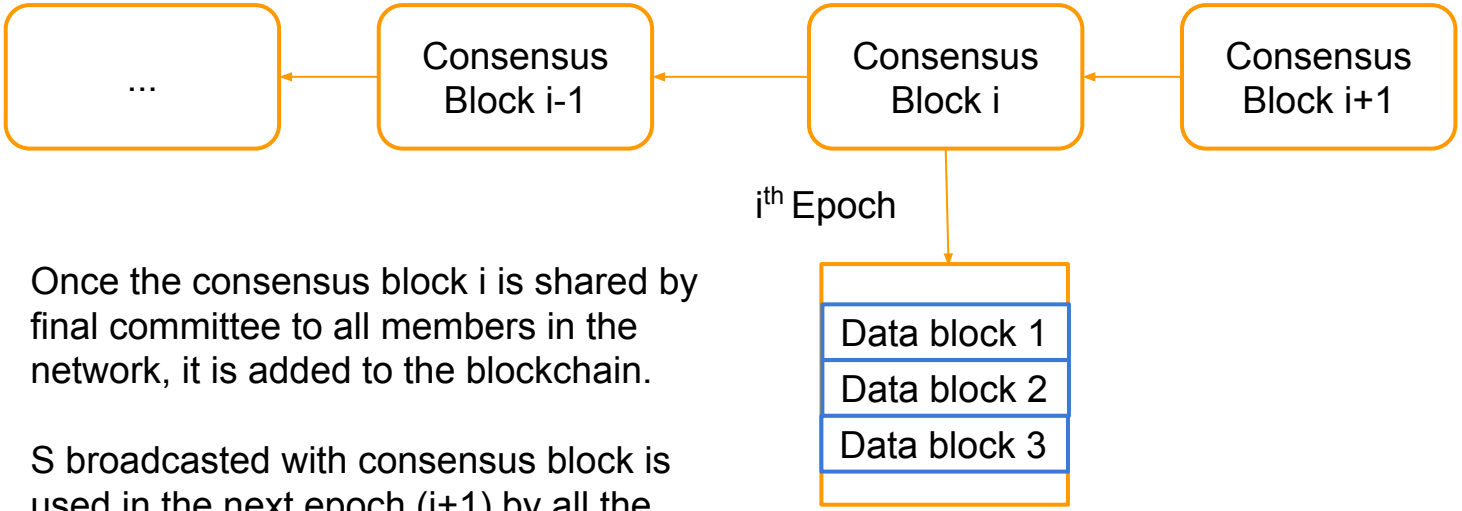
Committee

UCDAVIS COMPUTER SCIENCE



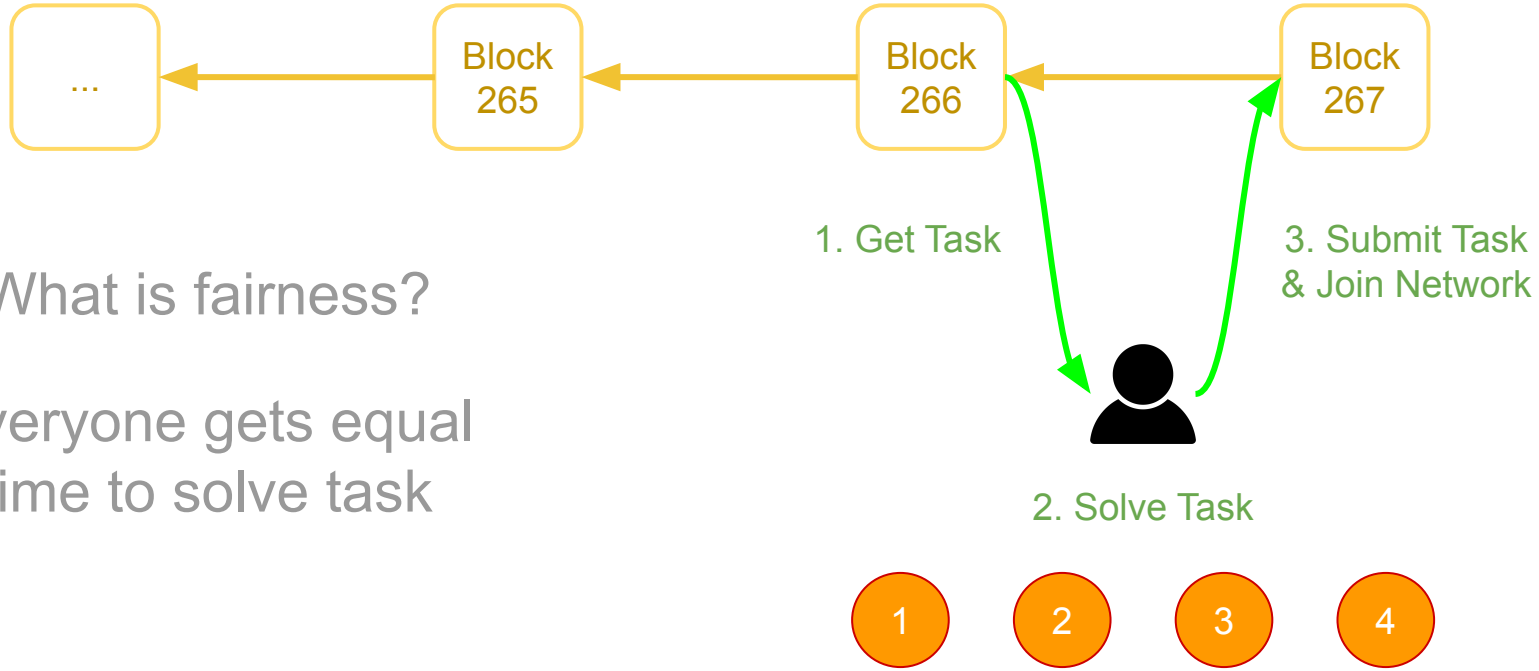
UCDAVIS COMPUTER SCIENCE

Consensus block added to the blockchain



- Once the consensus block i is shared by final committee to all members in the network, it is added to the blockchain.
- S broadcasted with consensus block is used in the next epoch (i+1) by all the members to verify identities of other nodes.
- Entire 5 step process repeats in the next epoch (i+1).

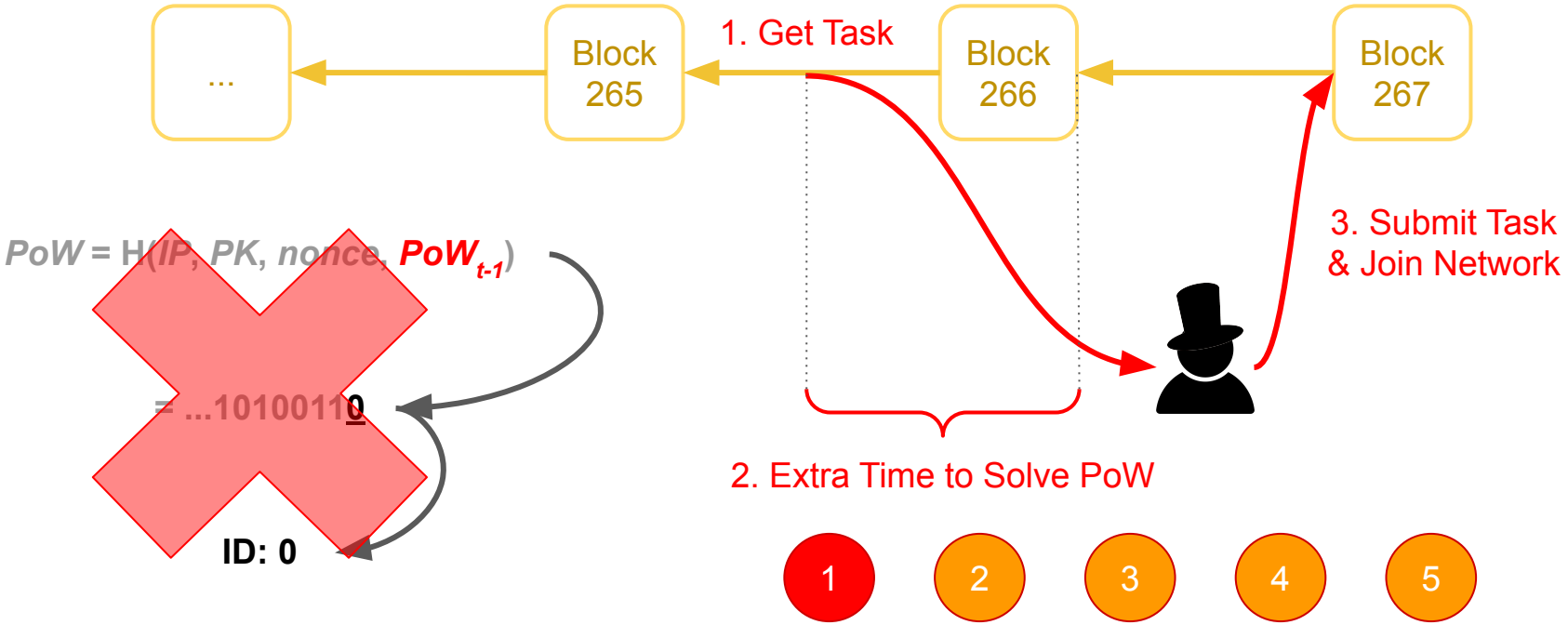
Enforcing Fair Acceptance



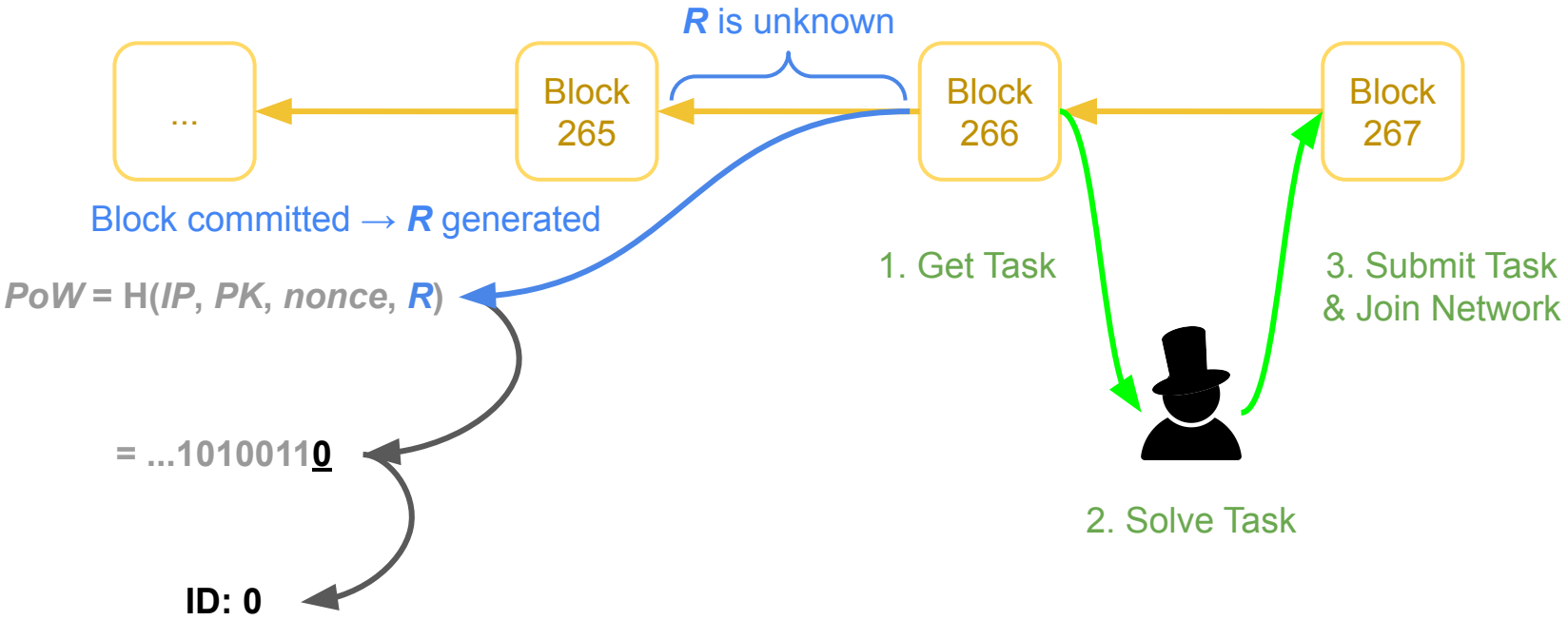
What is fairness?

Everyone gets equal time to solve task

Problem: Cheaters Precompute the PoW Task



Solution: Synchronize the Reveal of PoW Task



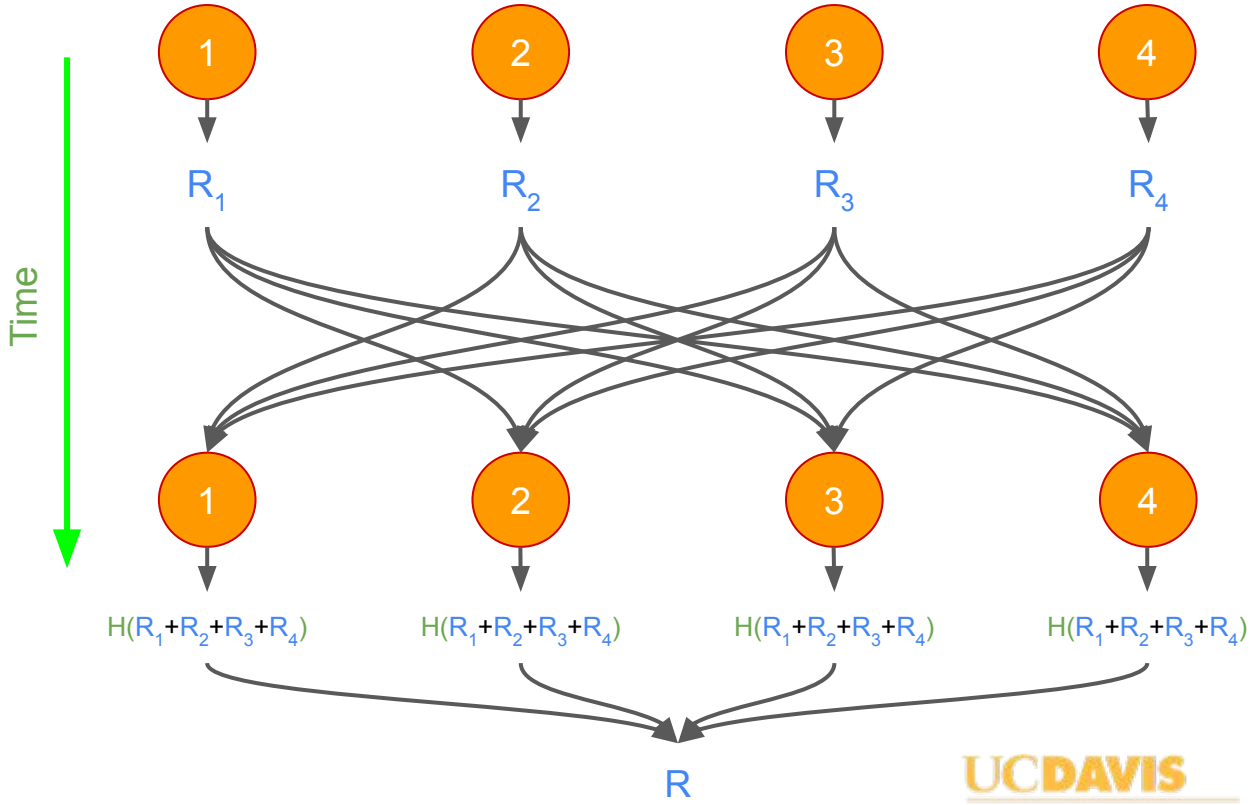
Generating Epoch Randomness *R*

How to robustly generate verifiable random numbers in a dynamic peer-to-peer system?

UCDAVIS
COMPUTER SCIENCE

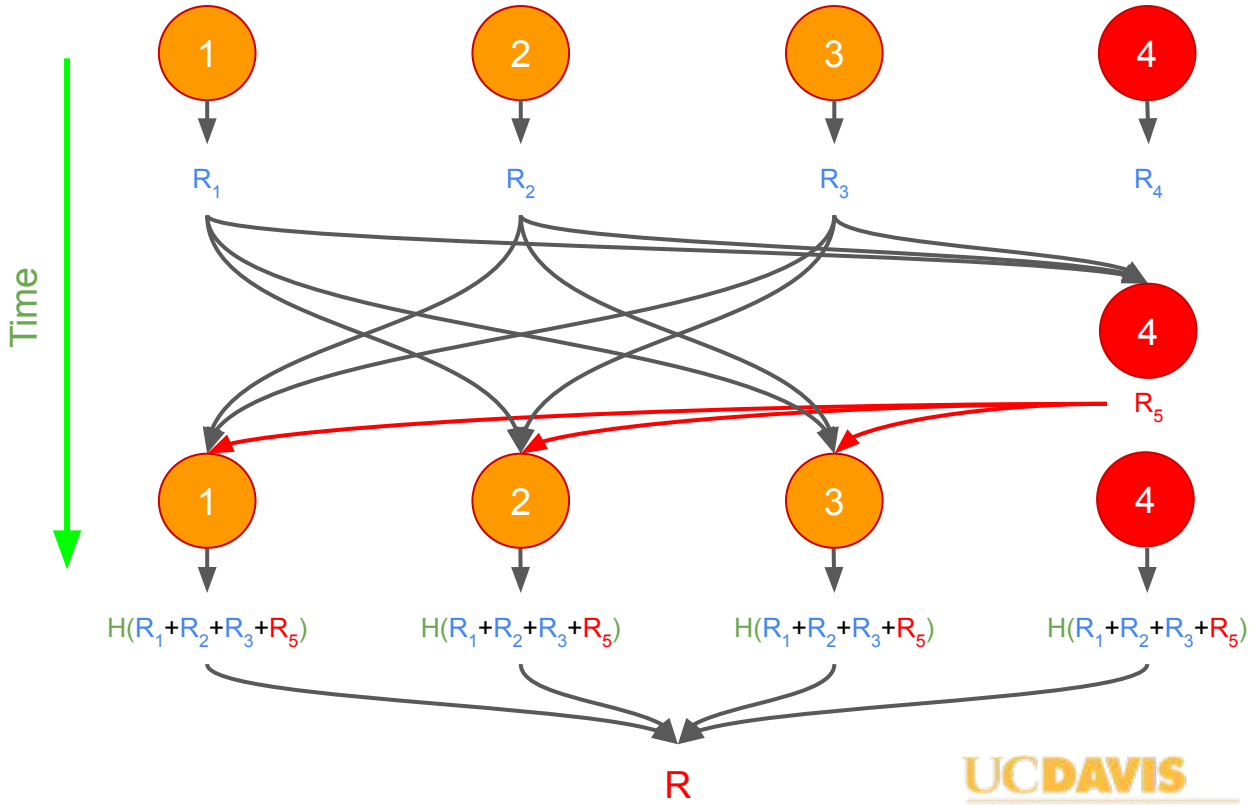
Problem: Decentrally Generating a Random String

Does this work?



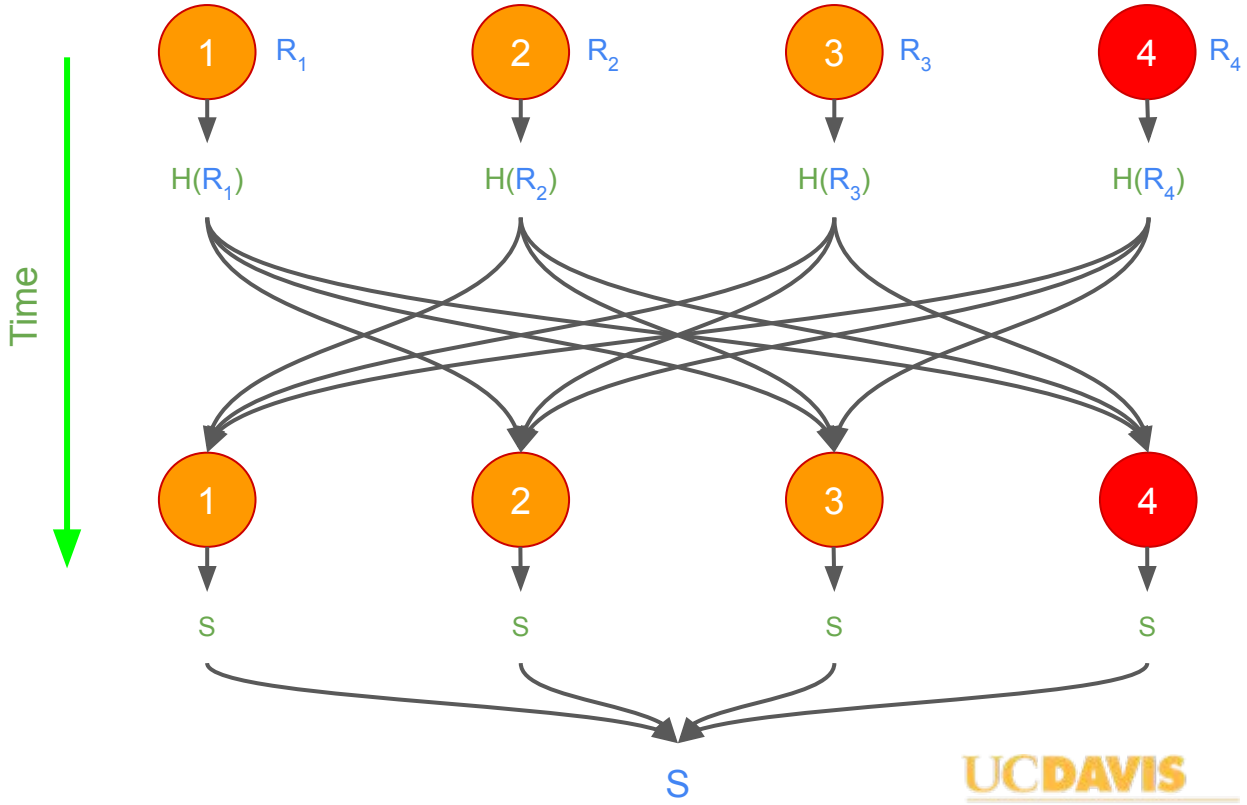
Problem: Decentrally Generating a Random String

Bad actor 4 chooses its random string to create an *R* it wants



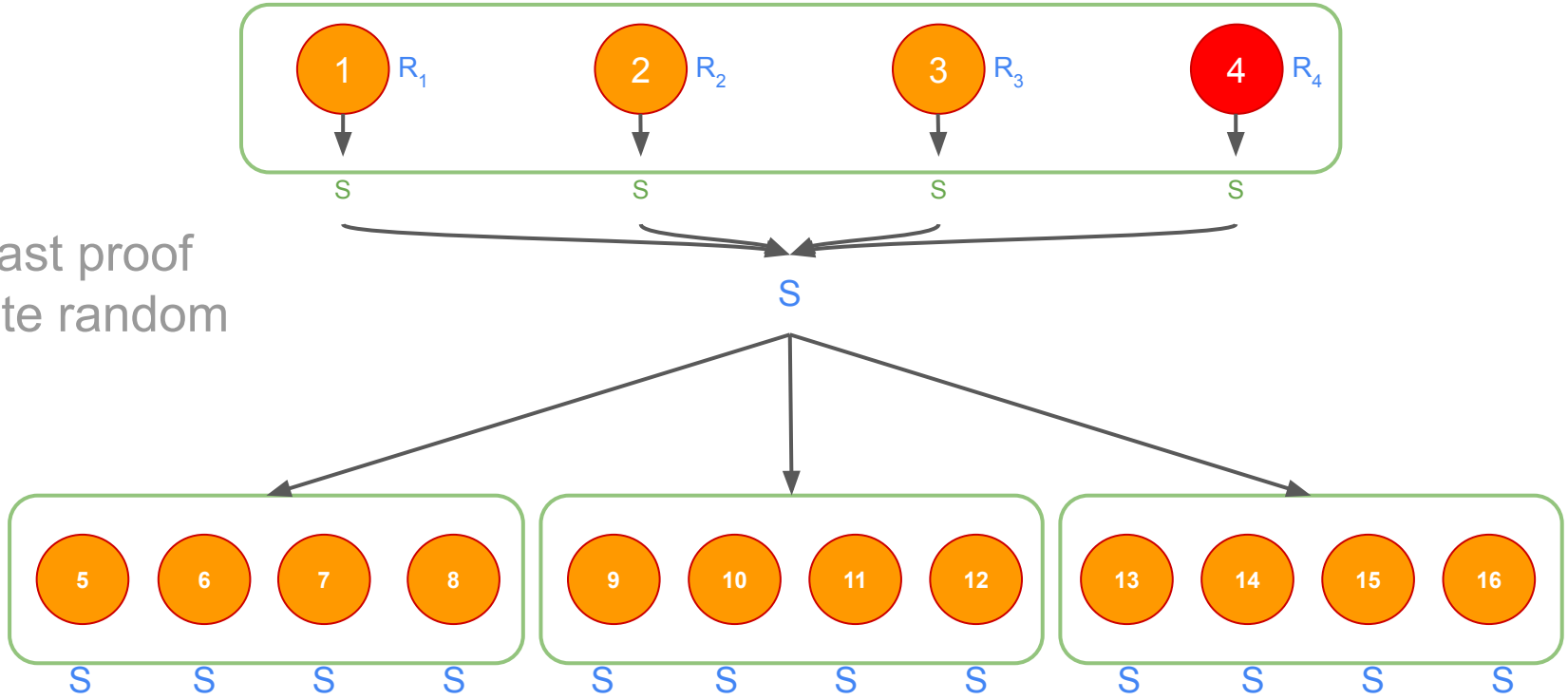
Solution: Commit Private Random Strings

"Lock-in" private random strings before revealing

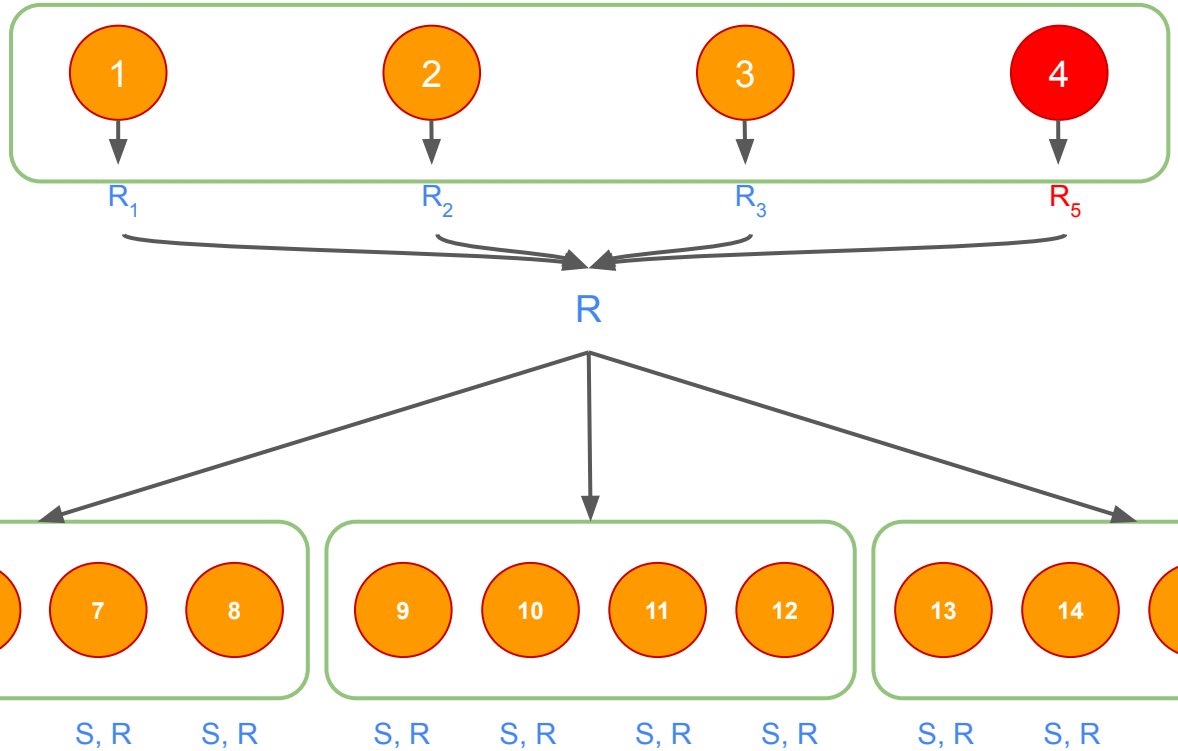


Generating Epoch Randomness: Phase 1

Broadcast proof of private random strings



Generating Epoch Randomness: Phase 2

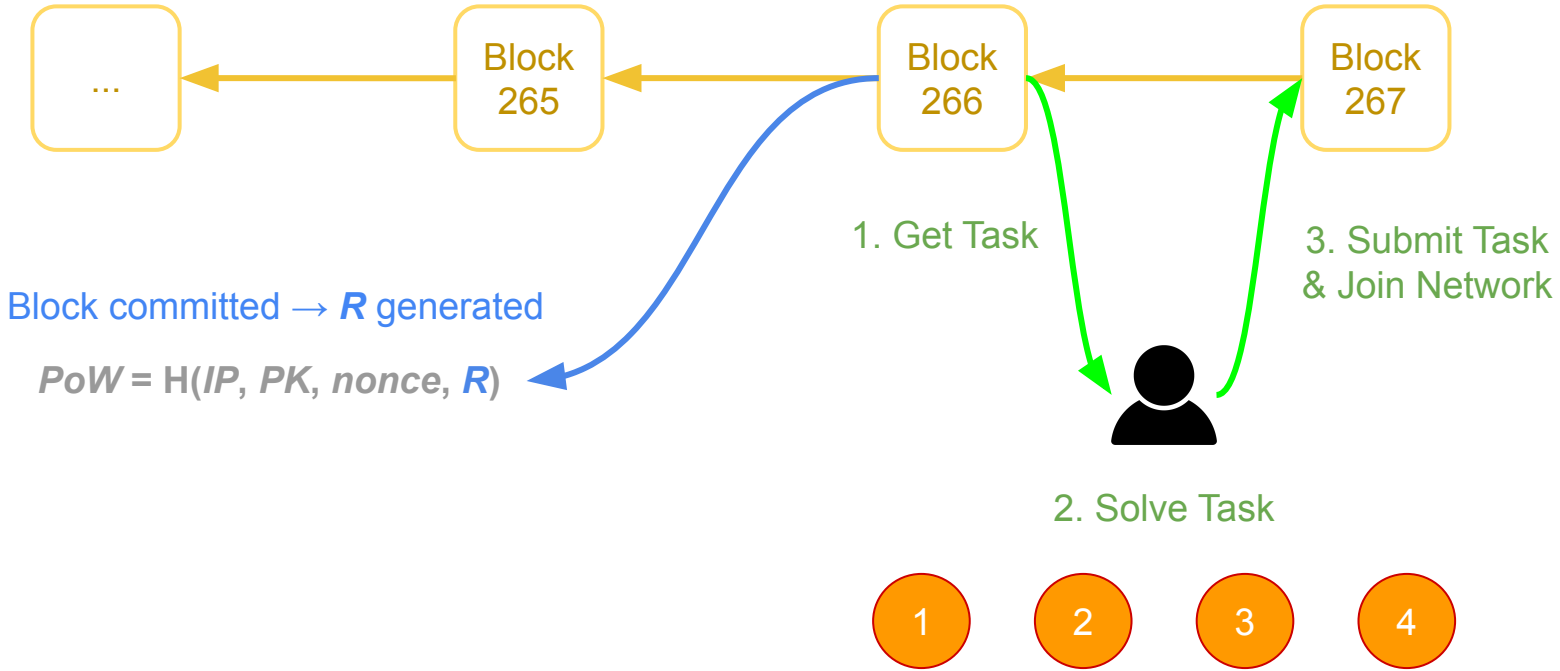


Broadcast private random strings

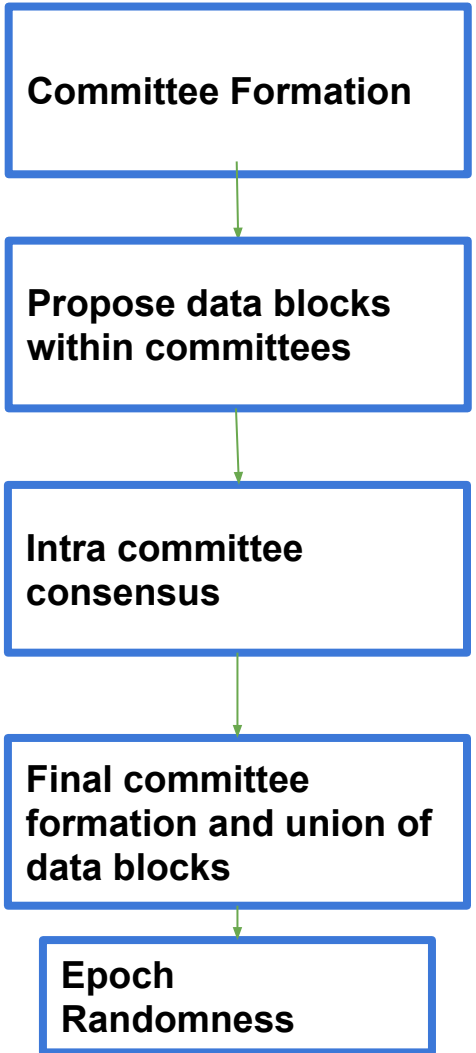
Verify(S, R_1)
Verify(S, R_2)
Verify(S, R_3)
~~Verify(S, R_5)~~

UCDAVIS COMPUTER SCIENCE

Enforcing Fair Acceptance



RECAP



Elastico Security Analysis

Good Randomness

- Every user has a publicly random string of r bits, verifiably generated in the previous epoch;
- No user has access to such a verifiable random string more than δt prior to the beginning of the epoch;
- Malicious users can bias the randomness with negligible probability.

Good Majority

- In every epoch with good randomness, for every sufficiently large integer $n' \geq n_0$: among the first n_0 identities created, at most $n_0/3 - 1$ are controlled by the adversary w.h.p.

Elastico Security Analysis

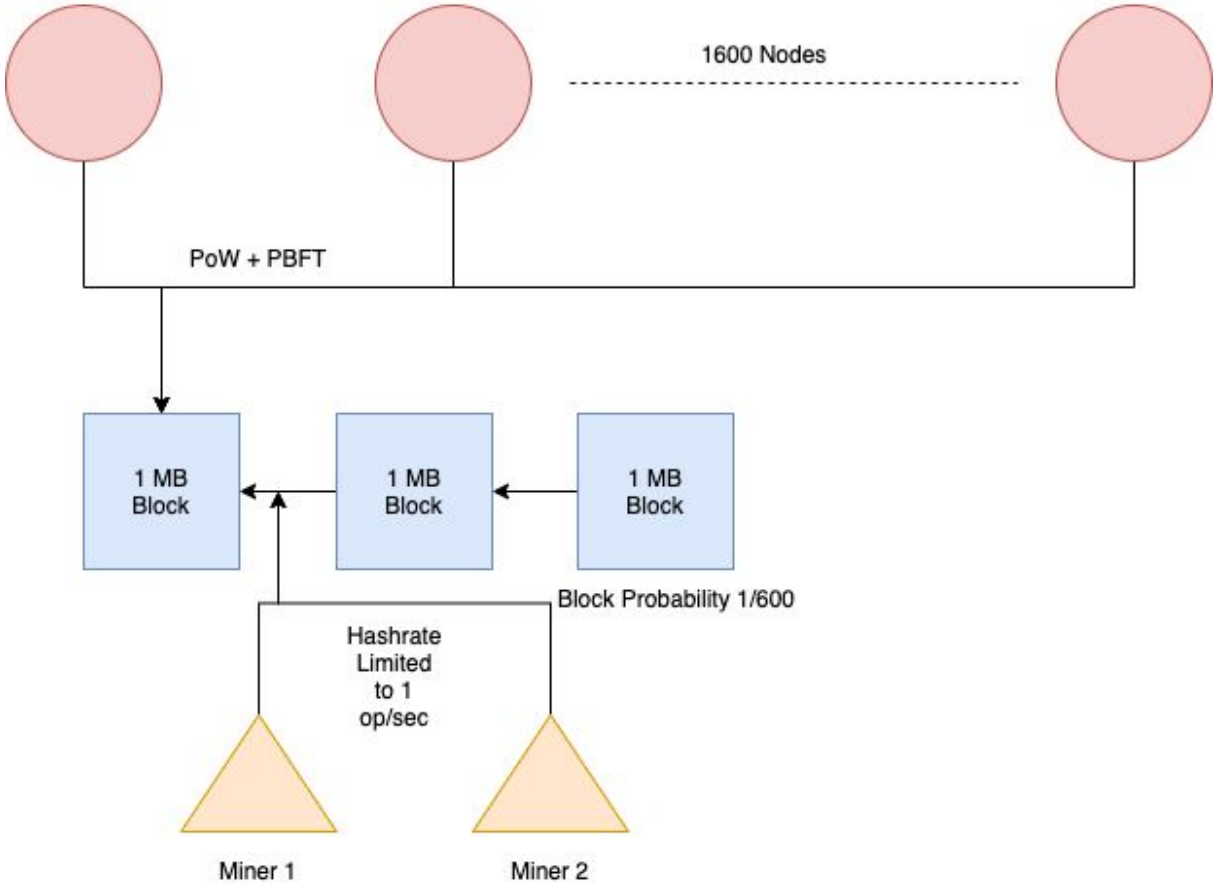
Good views with bounded inconsistency

- Each member has their own view of who are in the committee. Two views of two honest members differ by at most 1/3 of the committee size
- All honest members have identities of other honest members in their views
- The total number of unique identities in all views is at most 3c/2 of which less than 1/3 fraction are malicious.

Consensus

- In every epoch with good randomness, the honest members agree on a unique set X_i with at least c/2+1 signatures, with high probability.

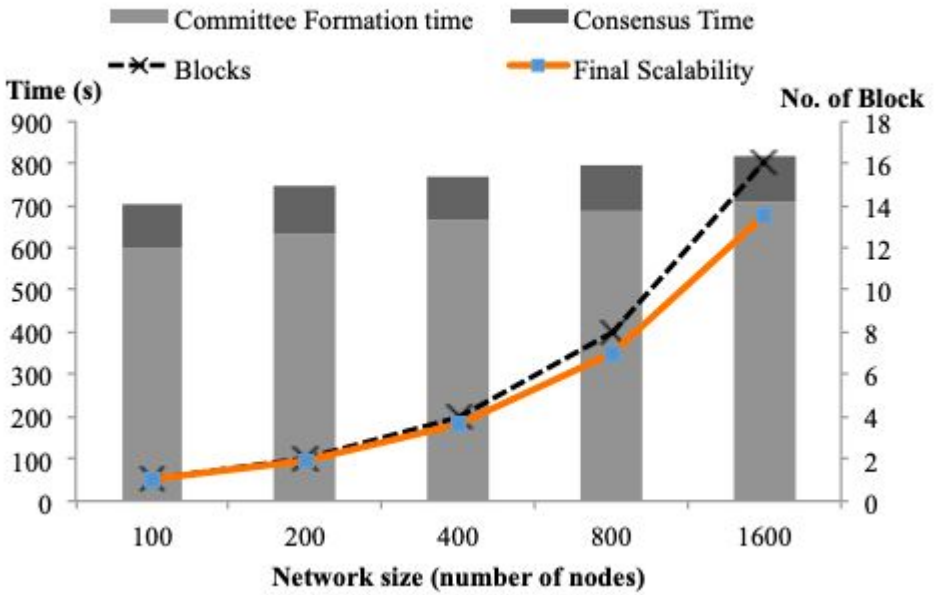
Elastico Consensus



Implementation Details

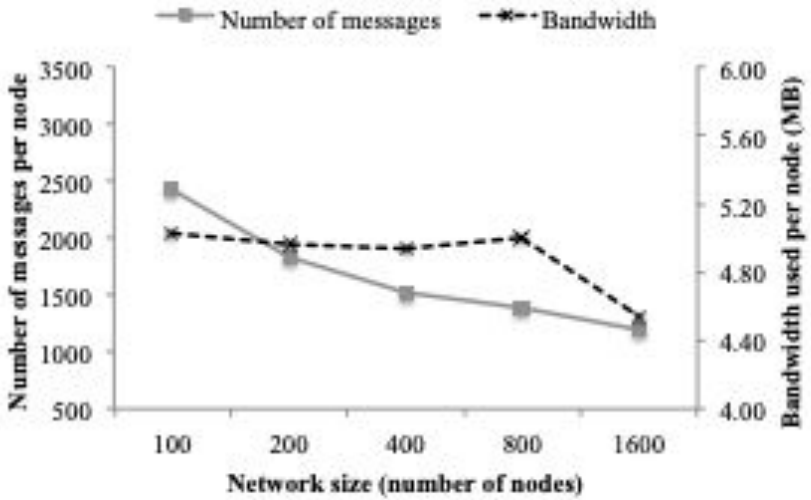
- Elastico was implemented on 800 AWS EC2 Instances (2 vcpus)

1600 Nodes, each running 1 vcpu



Spot the error

Bandwidth Utilization



400 Nodes	4.80 MB
800 Nodes	5.01 MB

Relatively Unchanged?

Why Elastico?

	BFT	Nakamoto	Quorum-BFT	Two-phase commit	ELASTICO
Candidate	Tendermint [35] IBM Blockchain [36] Chain OS [19] DigitalAsset [37]	Bitcoin [1] Ethereum [38] BitcoinNG [9] IntelLedger [18]	Ripple [22] Stellar [21]	Spanner [15] RSCoin [39] Databases	This work
Decentralized	Yes ✓	Yes ✓	No	No	Yes ✓
Identity-less	No	Yes ✓	No	No	Yes ✓
Bandwidth (per node)	$O(n^2)$	Constant ✓	$O(n)$	Constant ✓	Constant ✓
Scalability	No	No	No	Yes ✓	Yes ✓

Can you tell us why “not” elastico?

Conclusion

- The traditional PoW infrastructure has scalability and performance issues.
- We introduce elastico, which is a revolutionary permissionless sharded protocol.
- Elastico allows to reduce the number of message communications from n^2 to nc
- First the directory committee is formed, and then the network is populated.
- When a client proposes a transaction, the directory committee splits the data, assigns it to shards, and then the final committee verifies them, and tells every node to commit.
- The “epochRandomness” is generated, and using this, the new round begins.

References

- https://slideplayer.com/slide/13077927/
- https://www.slideshare.net/YongraeJo/scp-157559777
- https://loiluu.com/talks/scp.pptx
- https://dl.acm.org/doi/pdf/10.1145/2976749.2978389

Thank You

UCDAVIS
COMPUTER SCIENCE