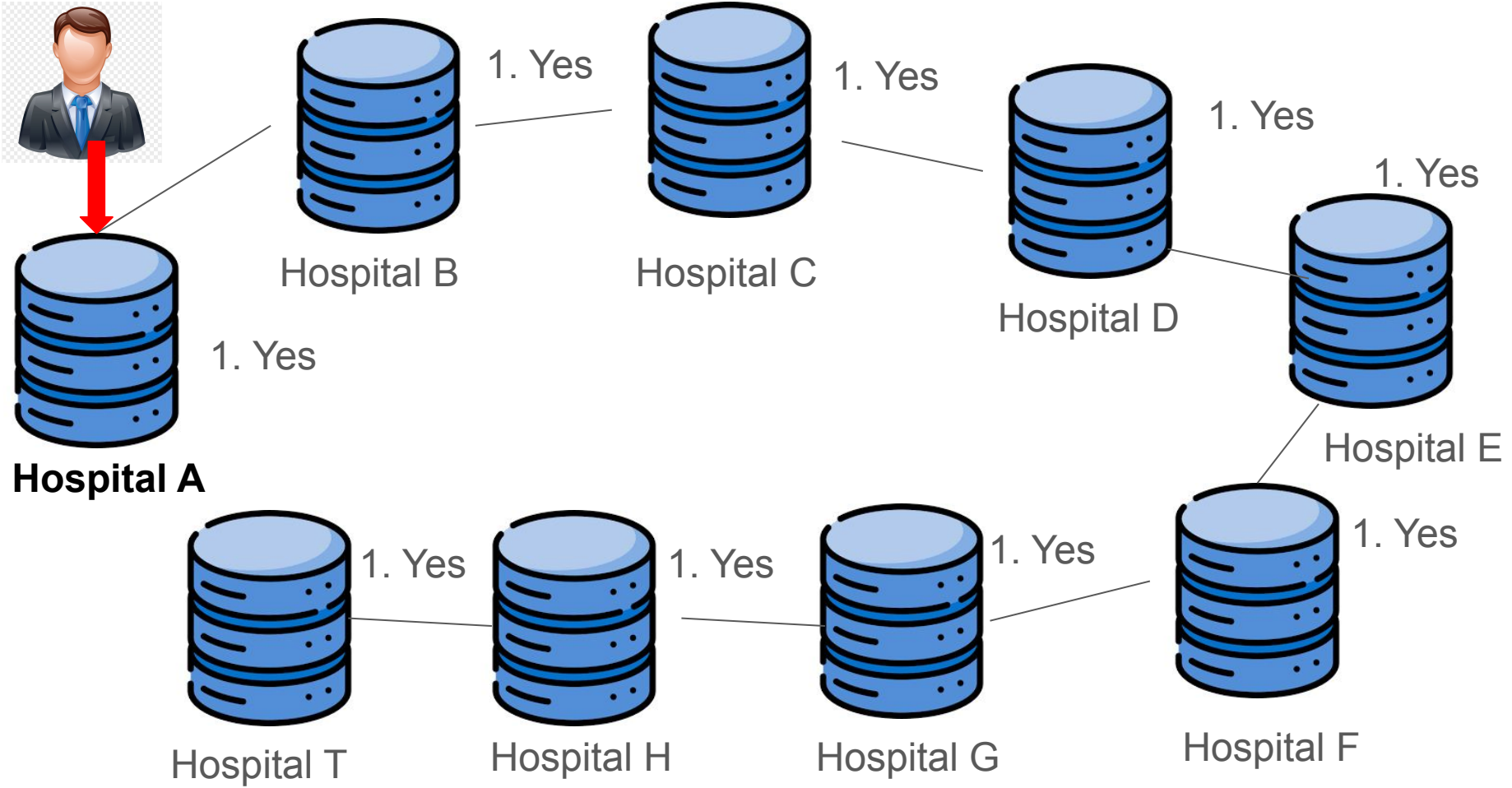
CSC 116 Blockchain


1. Yes
1. Yes
1. Yes
1. Yes
1. Yes
Hospital B
Hospital C
Hospital D
Hospital E
1. Yes
Hospital A
1. Yes
1. Yes
1. Yes
1. Yes
Hospital T
Hospital H
Hospital G
Hospital F

1. Yes
2.

Hospital B

1. Yes
2.

Hospital C

1. Yes
2.

Hospital D

1. Yes
2.

Hospital E

1. Yes
2.

Hospital A

1. Yes
2.

Hospital T

1. Yes
2.

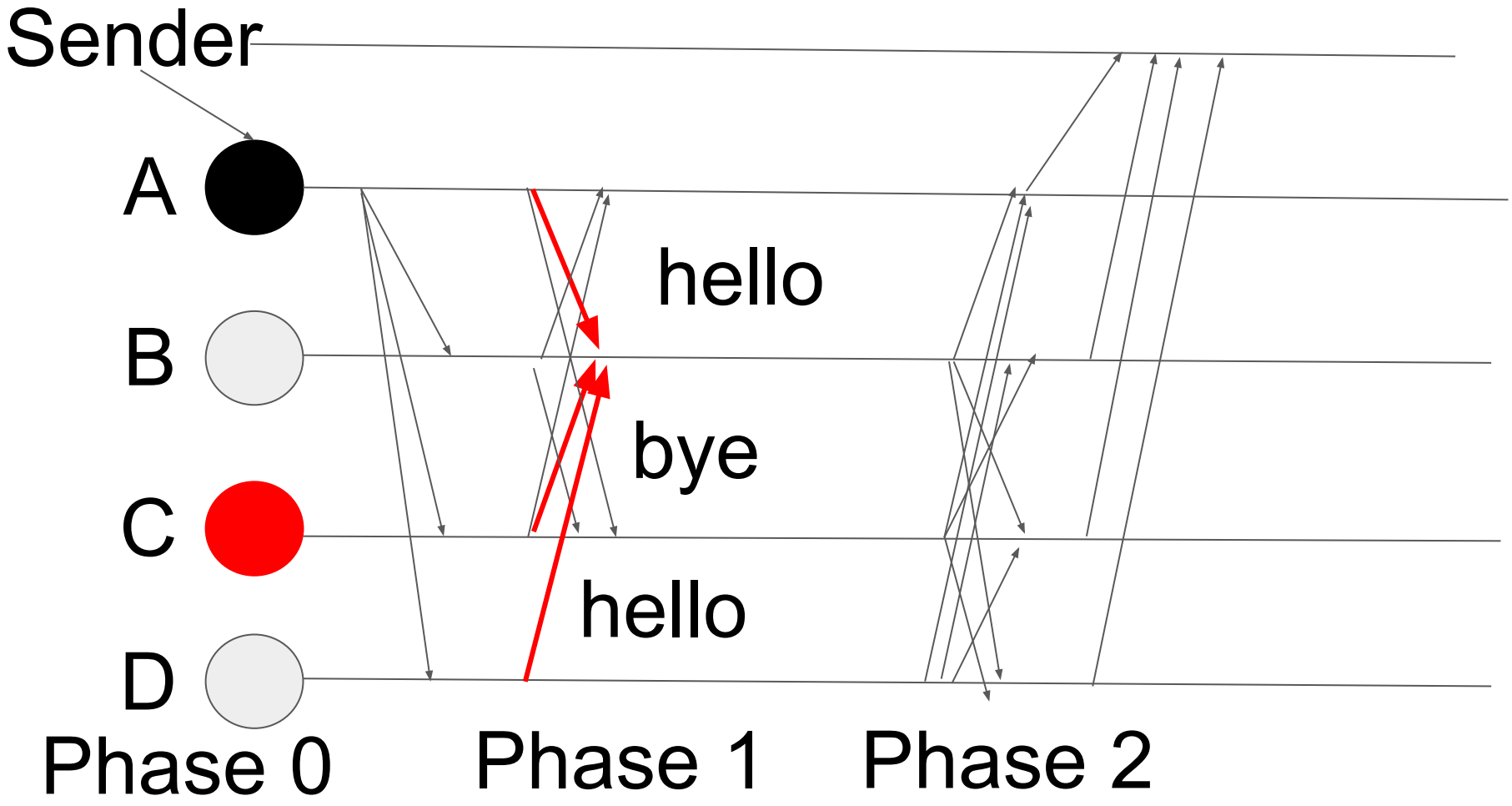
Hospital H

1. Yes
2.

Hospital G

1. Yes
2.

Hospital F


Sender
A
B
C
D
hello
bye
hello
Phase 0
Phase 1
Phase 2

Phase 0: send messages to everyone

Phase 1: ensuring that your messages that you received are correct: You need to select the majority ones.

Phase 2: after you decided, you need to tell your final decision to everyone, so everyone will store the data in their local database, and reply to sender. So sender will send a new message (sender is waiting, leader is waiting your reply as well)

Why we need BFT?

To make agreement with all the nodes. So all the nodes will do the same order.

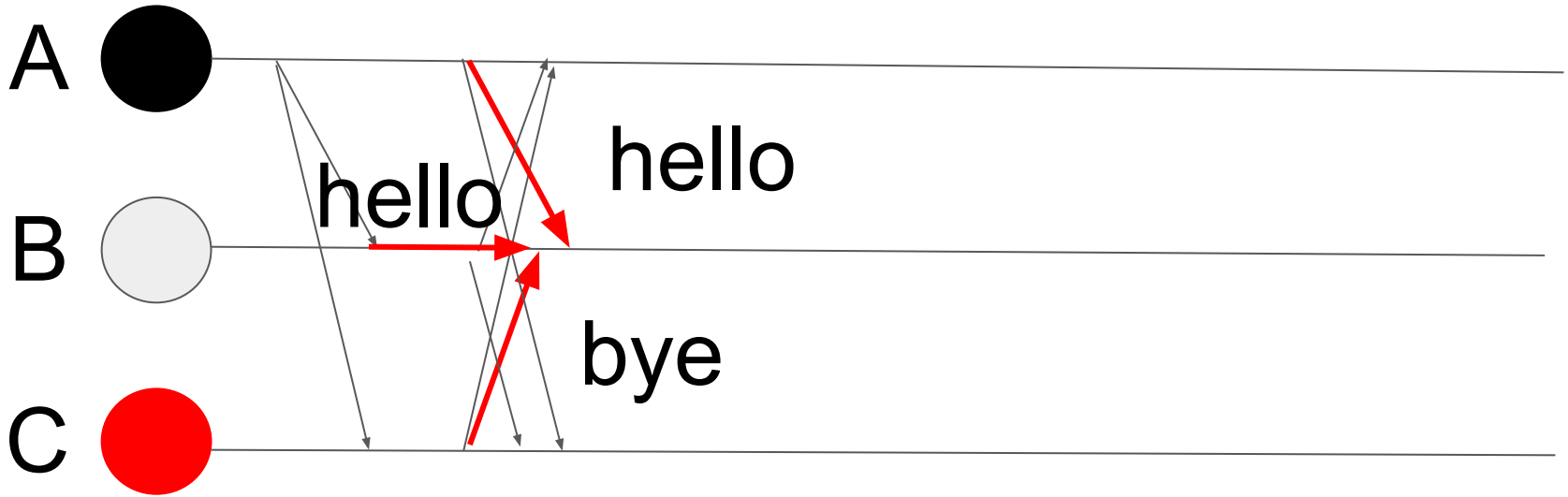
Which means all the nodes will store the messages in same order (total order & totality).

Why need leader? Why can not leader just broadcast messages in a strict order to all of you?

Network delay:
You will receive order 2 first, and then order 1 because of the network delay or attack.

Phase 0 Phase 1

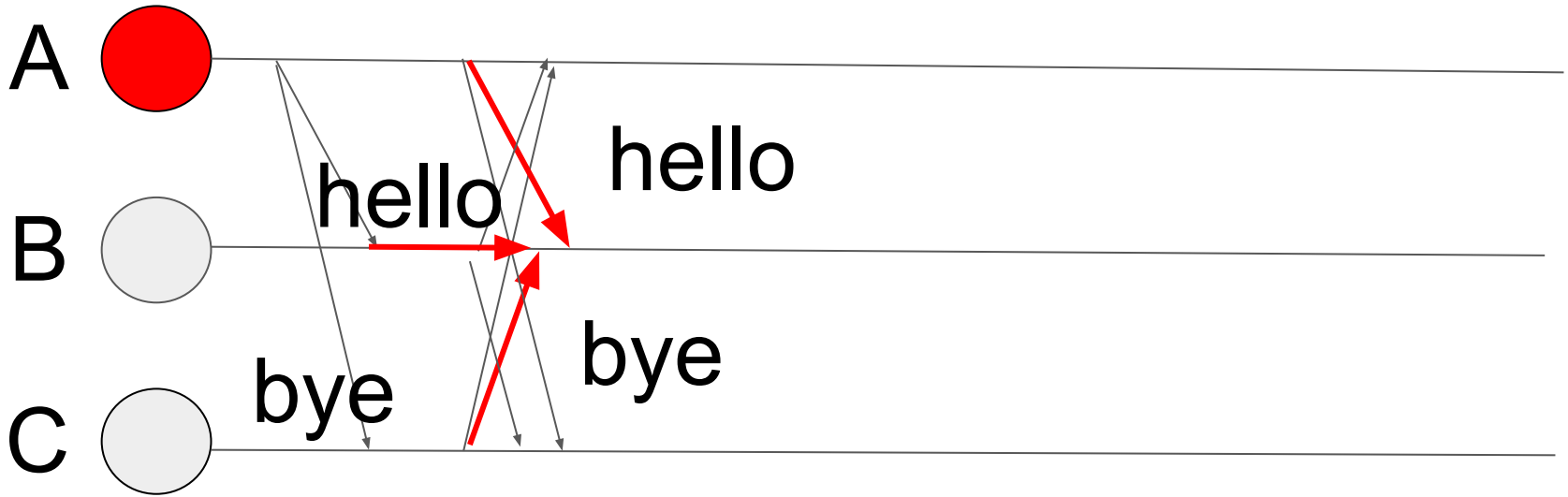
Malicious students



B: (hello), hello, bye

Phase 0 Phase 1

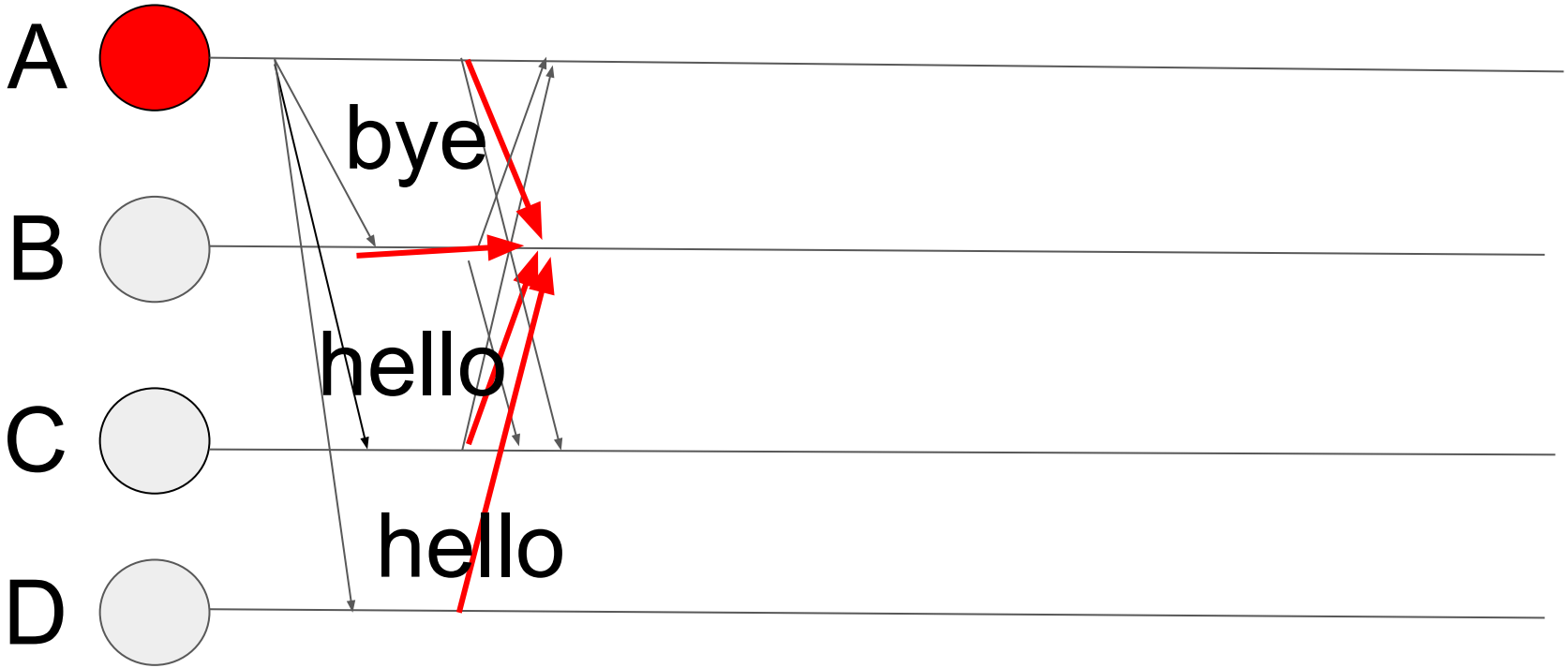
Malicious leader



B: (hello), hello, bye

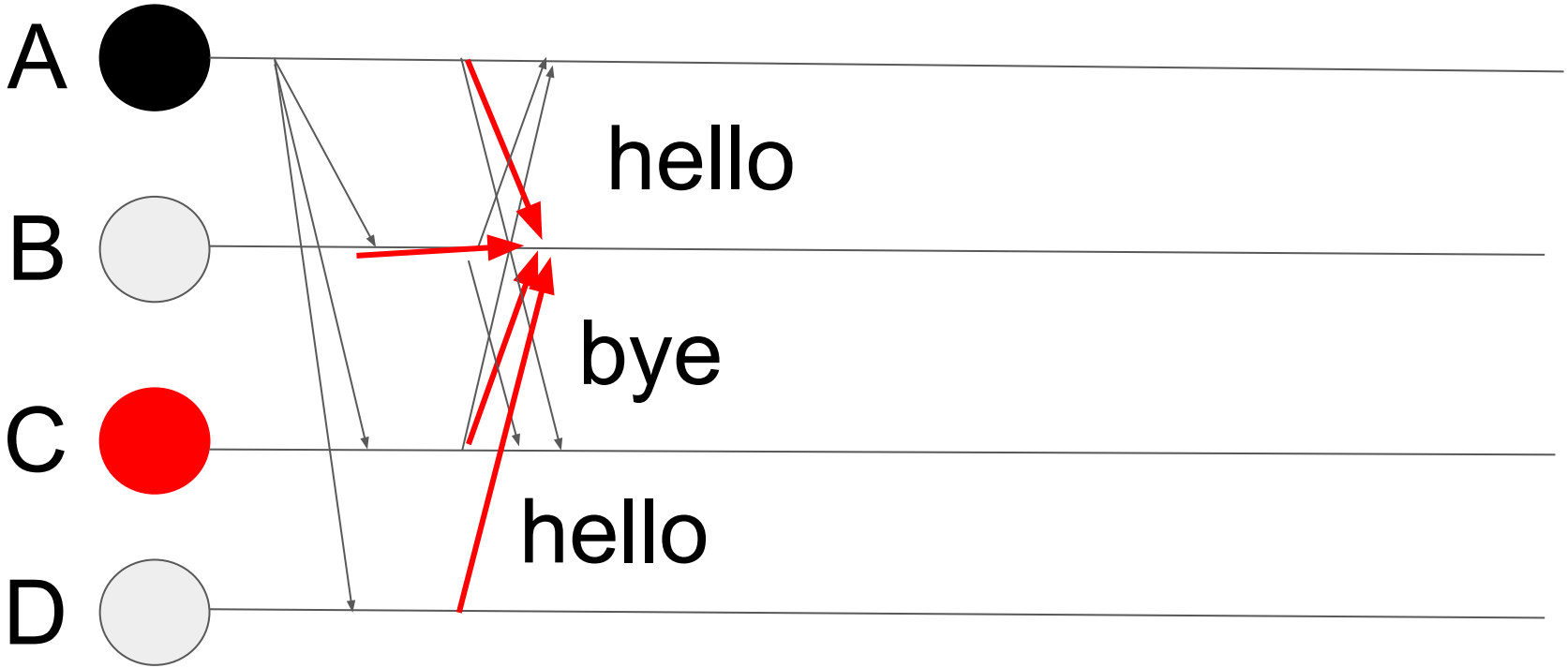
C: (bye), bye, hello

Phase 0 Phase 1



B: (bye), bye, hello, hello

Phase 0 Phase 1



B: (hello), hello, hello, bye

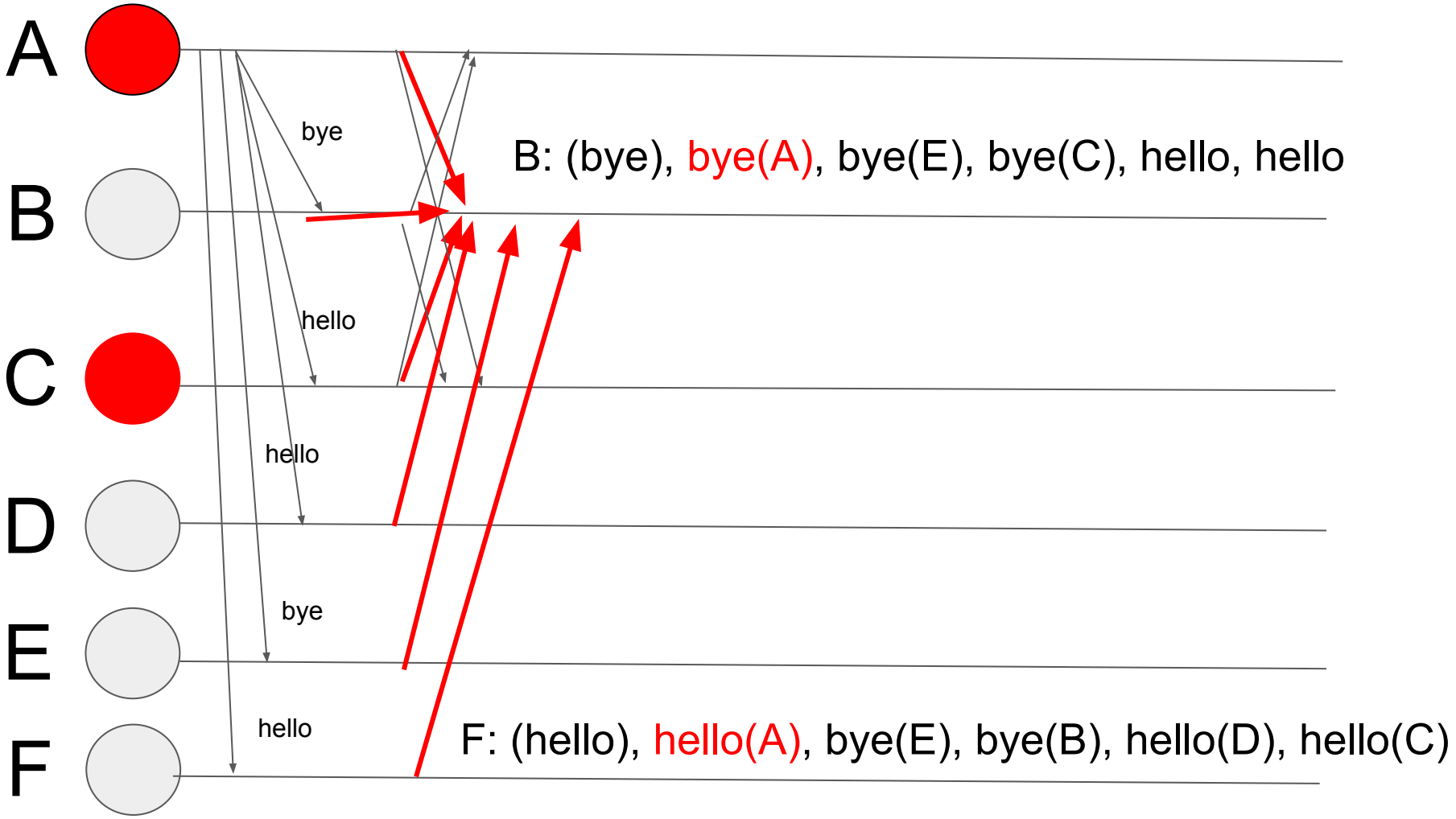
n: total nodes

f: total number of malicious nodes

n - f > f : the number of correct students needs to large than the number of malicious students.

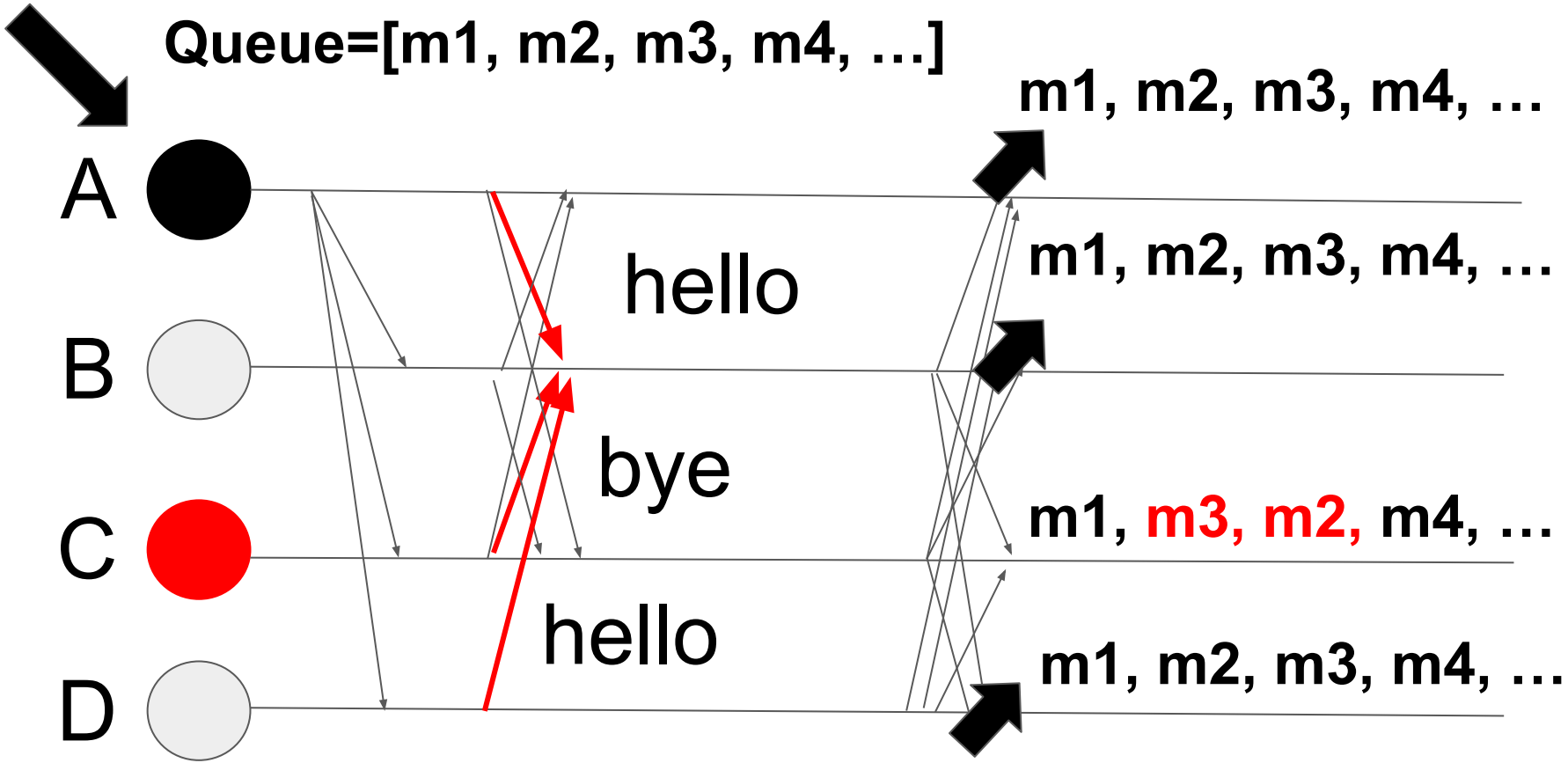
$$n > 2f$$

$$n >= 2f + 1$$

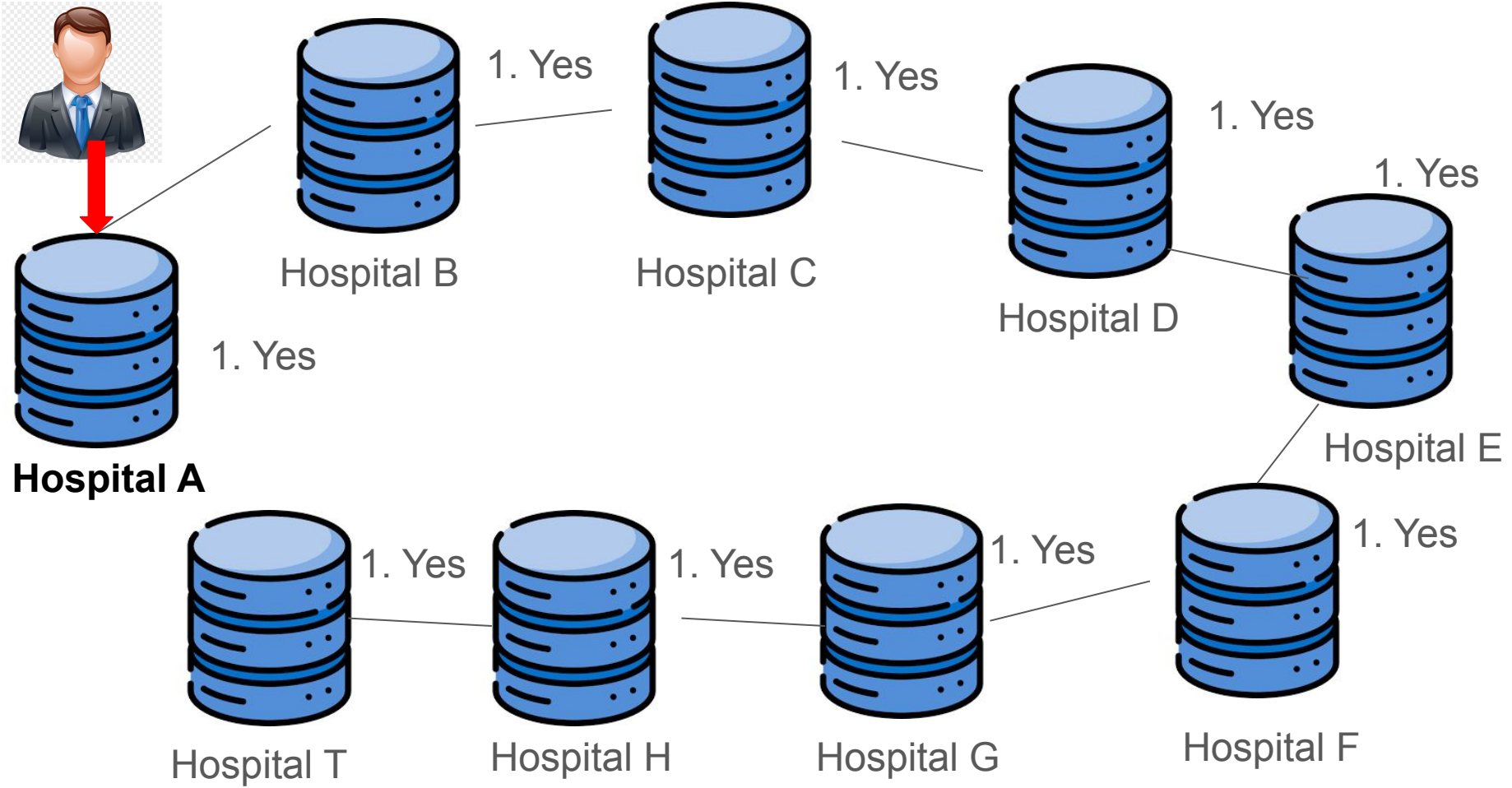

A
B
C
D
E
F
bye
hello
hello
bye
hello
B: (bye), bye(A), bye(E), bye(C), hello, hello
F: (hello), hello(A), bye(E), bye(B), hello(D), hello(C)

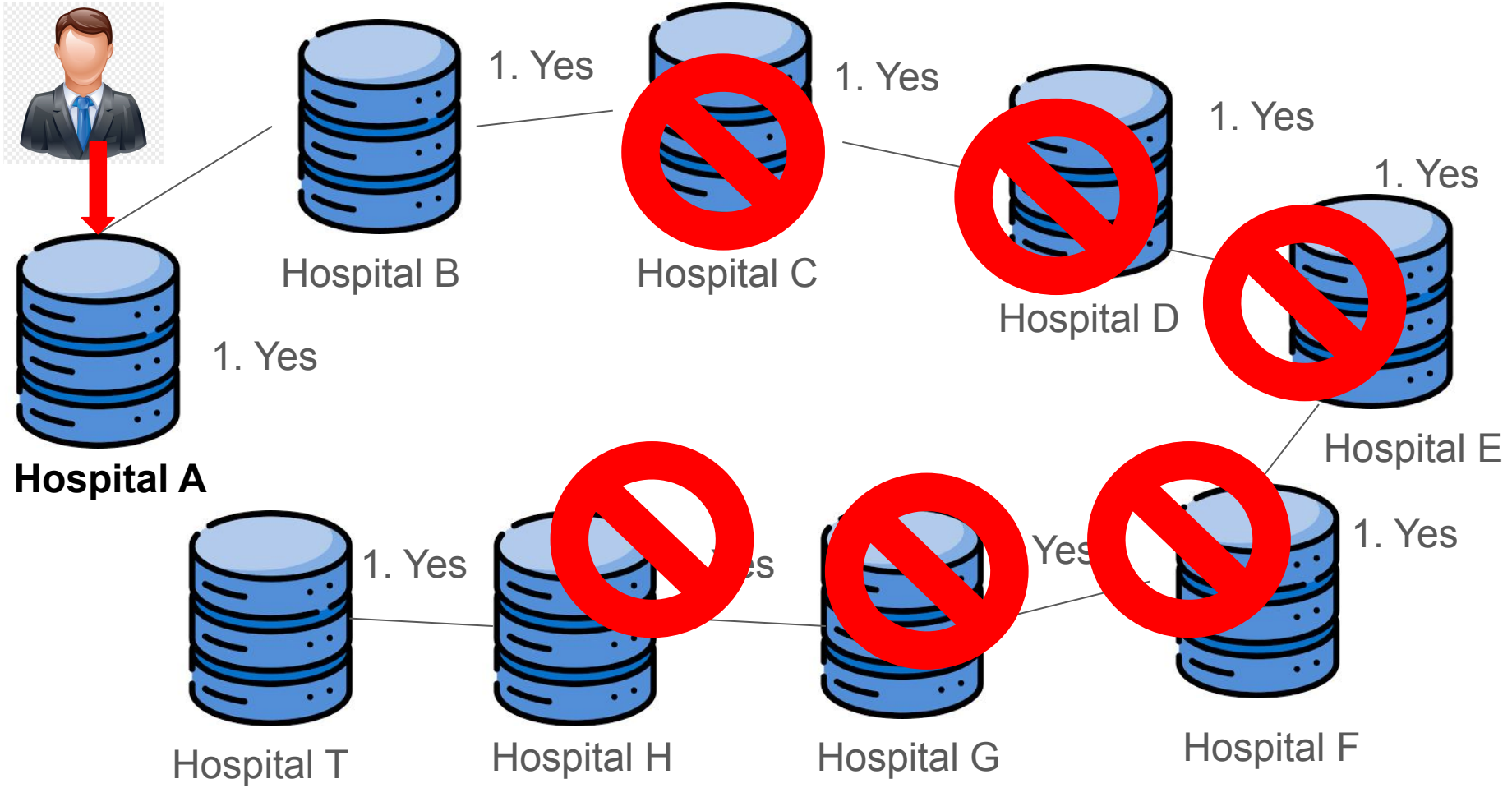
4 nodes can tolerate 1
5 nodes can tolerate 1
6 nodes can tolerate 1
7 nodes can tolerate 2
8 nodes can tolerate 2
9 nodes can tolerate 2
10 nodes can tolerate 3

$$n >= 3f + 1$$


Queue=[m1, m2, m3, m4, …]
A
B
C
D
hello
bye
hello
m1, m2, m3, m4, …
m1, m2, m3, m4, …
m1, m3, m2, m4, …
m1, m2, m3, m4, …

51% Attackers in Blockchain


1. Yes
1. Yes
1. Yes
1. Yes
1. Yes
Hospital B
Hospital C
Hospital D
Hospital E
1. Yes
Hospital A
1. Yes
1. Yes
1. Yes
1. Yes
Hospital T
Hospital H
Hospital G
Hospital F


1. Yes
1. Yes
1. Yes
1. Yes
1. Yes
1. Yes
Hospital A
Hospital B
Hospital C
Hospital D
Hospital E
Hospital F
Hospital G
Hospital H
Hospital T

1. Yes

Hospital B

1. NO

Hospital C

1. No

Hospital D

1. No

Hospital E

1. Yes

Hospital A

Attackers can not control 51% of nodes

1. Yes

Hospital T

1. NO

Hospital H

1. NO

Hospital G

1. No

Hospital F

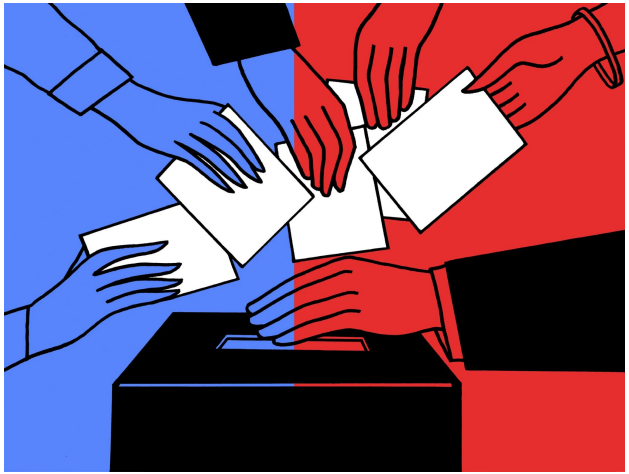
Why use BFT Consensus in Blockchain?

BFT can be used in Blockchain to make sure that the messages in every nodes are in the same order.

There are so many blockchain nodes: Immutability

The Blockchain has a feature: Immutability. Because the data in the blockchain has so many copies, it is hard to controlled 51% of nodes.

Proof of Work and BFT



Aspect	Proof of Work (PoW)	Byzantine Fault Tolerance (BFT)
Consensus Method	Competition-based mining	Voting-based agreement
Block Production	Through solving cryptographic puzzles	Through leader proposal and voting
Node Identity	Anonymous, anyone can join	Known participants
Communication	Minimal between nodes	Heavy message exchange
Fault Tolerance	~51% honest computing power needed	Typically handles up to 33% Byzantine nodes
Performance	Lower throughput, higher latency	Higher throughput, lower latency
Security	Hard to attack	Mathematical proof of safety
Network Load	Lower network overhead	Higher network overhead
Incentives	Block rewards and fees	Usually no direct incentives