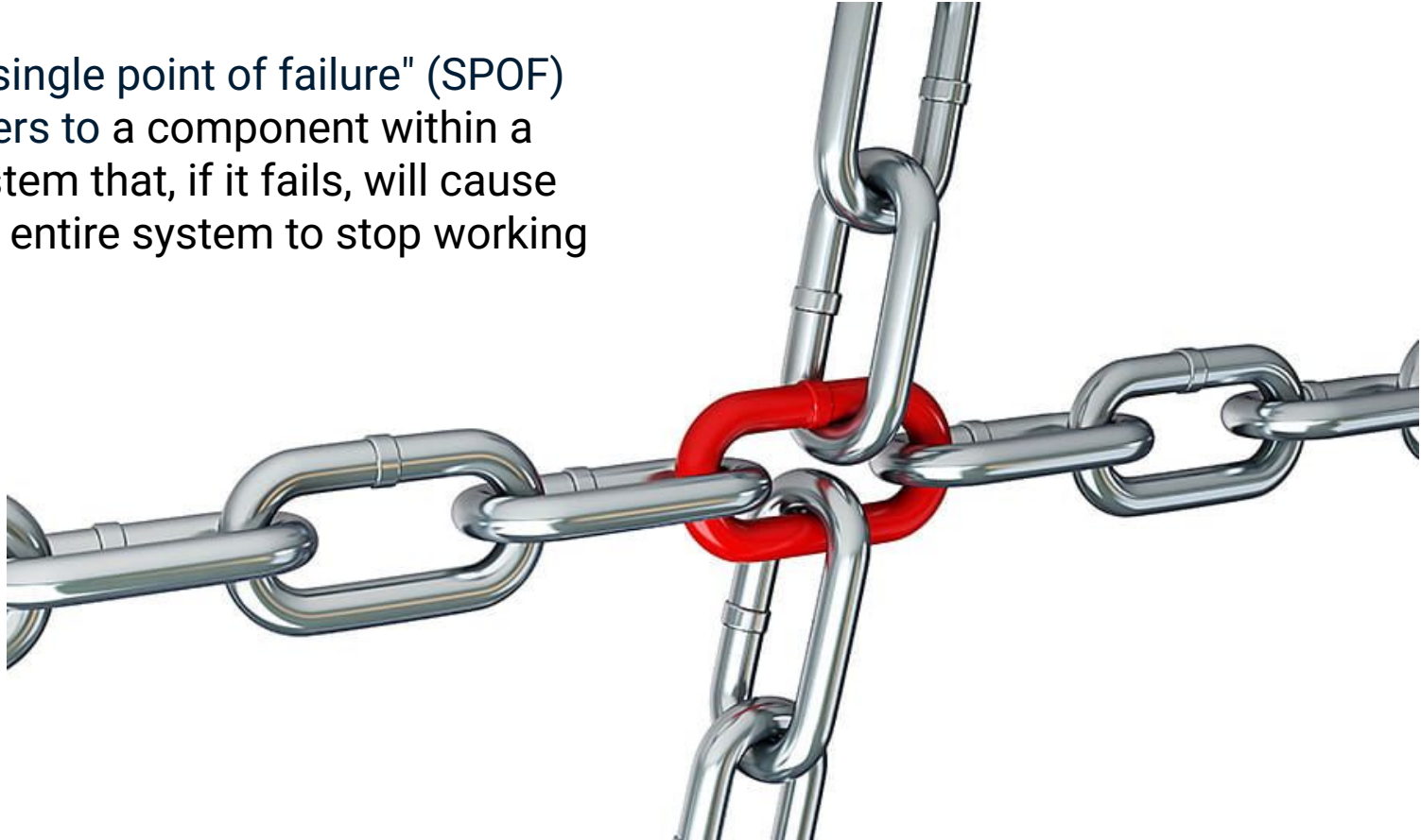
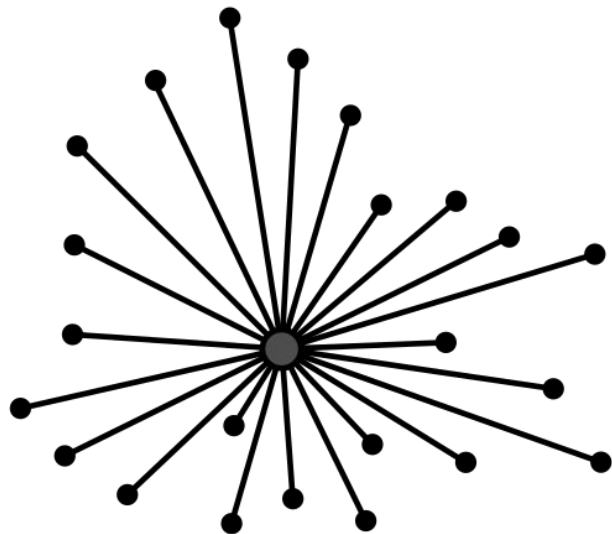


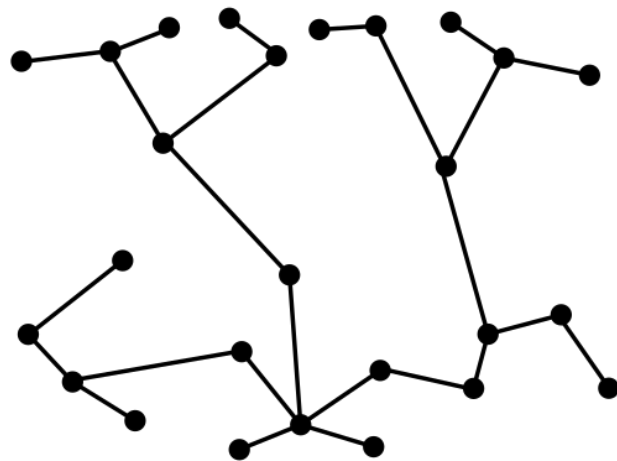
# **CSC 116 Single Point of Failure & BFT**

A "single point of failure" (SPOF) refers to a component within a system that, if it fails, will cause the entire system to stop working





CENTRALIZED

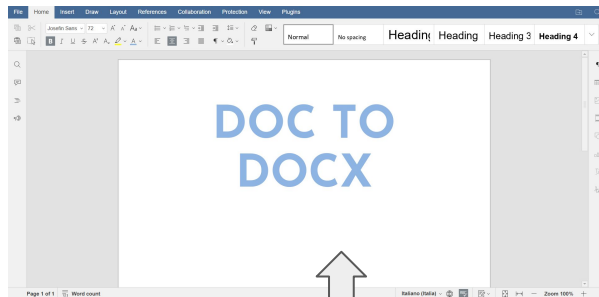
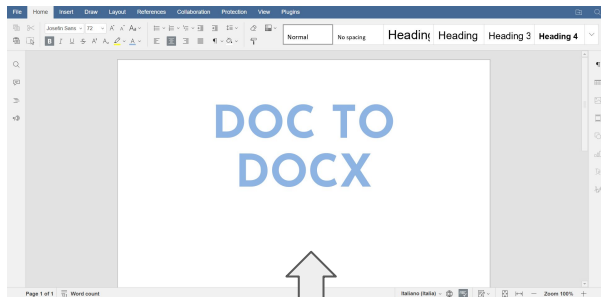
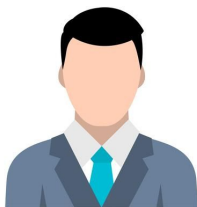
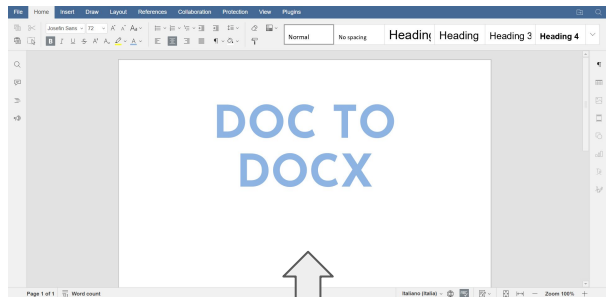


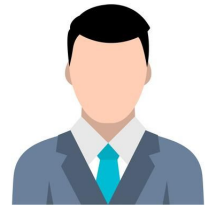
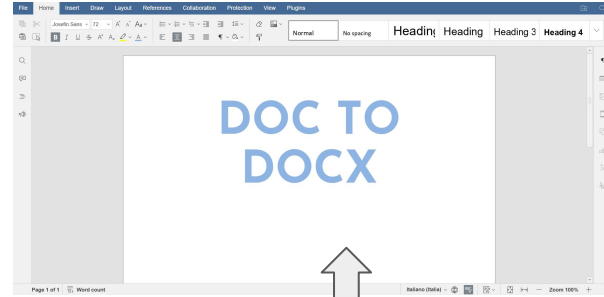
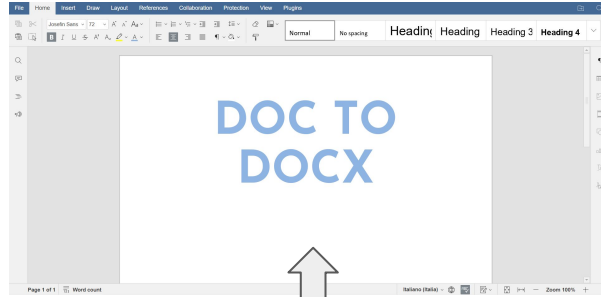
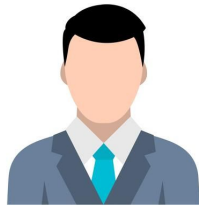
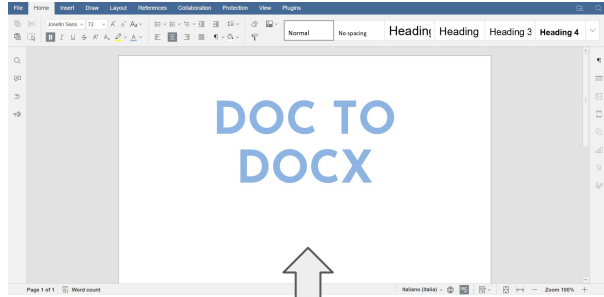
DECENTRALIZED

**How to make sure that all the  
datasets in different servers  
are the same datasets?**



Google Docs





# **BFT**

(Byzantine Fault Tolerance)



It's an algorithm!



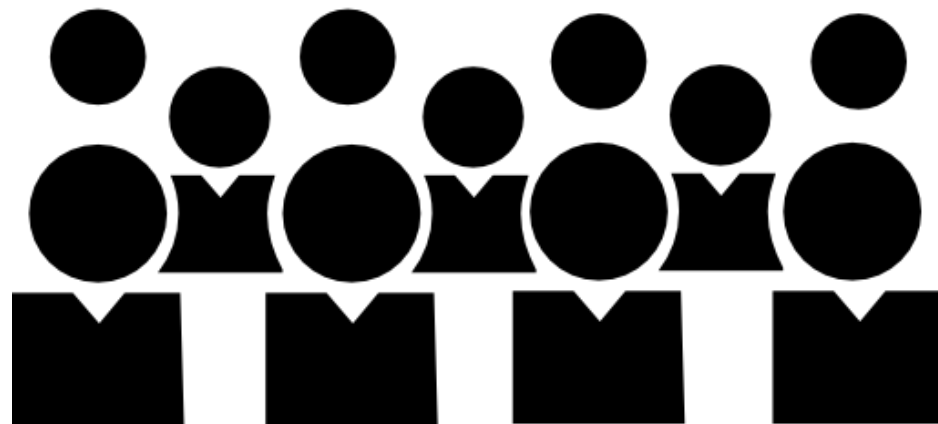
Fake news!

You don't need to complete the assignment 2,  
I will give you 10% for free.



Yusen

Command





1. Yusen is  
a global unique  
hash code!



2. You are all real AI models.  
Your brains tell you: 1). Yusen  
is a real person and he is the  
instructor of this course. 2) He  
is in the class, we are face to  
face, 3). I clearly hear what he  
said in the classroom.



All of you in this classroom will  
trust that this is a real command.  
But the students not here may  
not trust. It should be a joke!!

Let's make some conclusions:  
**What you find in this game?**

1 Your eyes record all the students, they are all evidences

## **Consensus**

2 You heard the leader's command, and the leader is trusted (I am a real instructor)

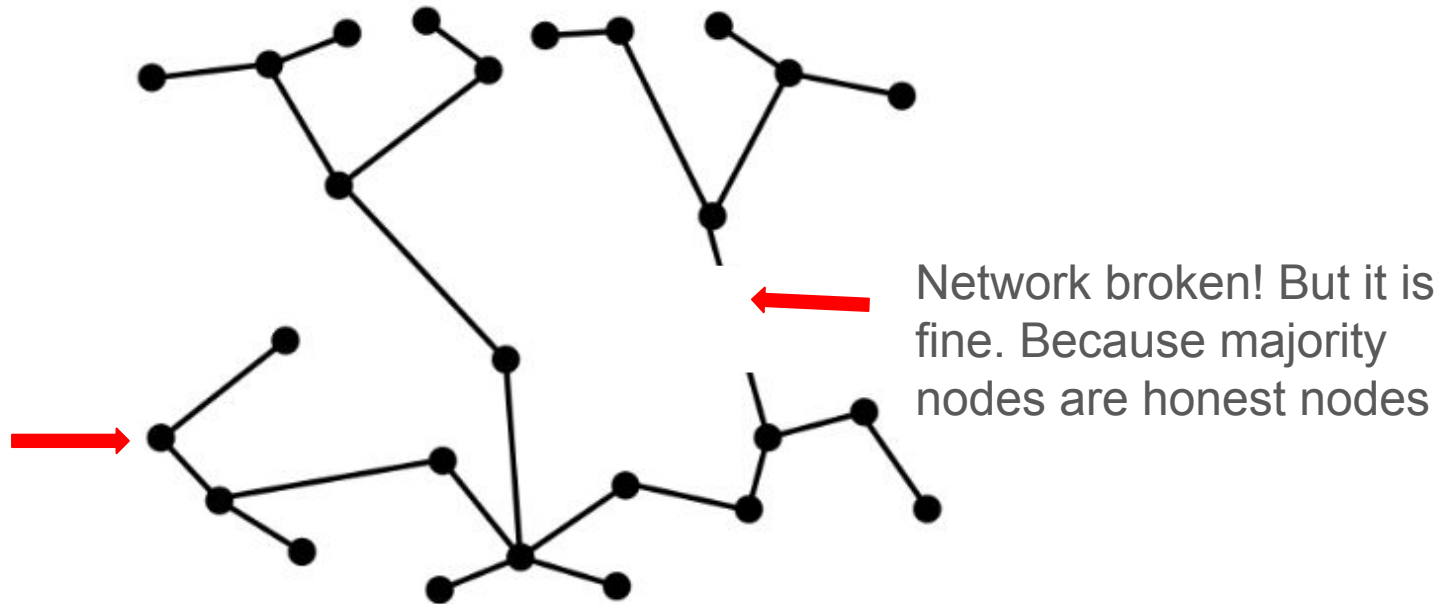
3 Your brain tells you it is true

**Other  
Students  
May not  
Trust you**

1 You are not a leader, it sounds not real

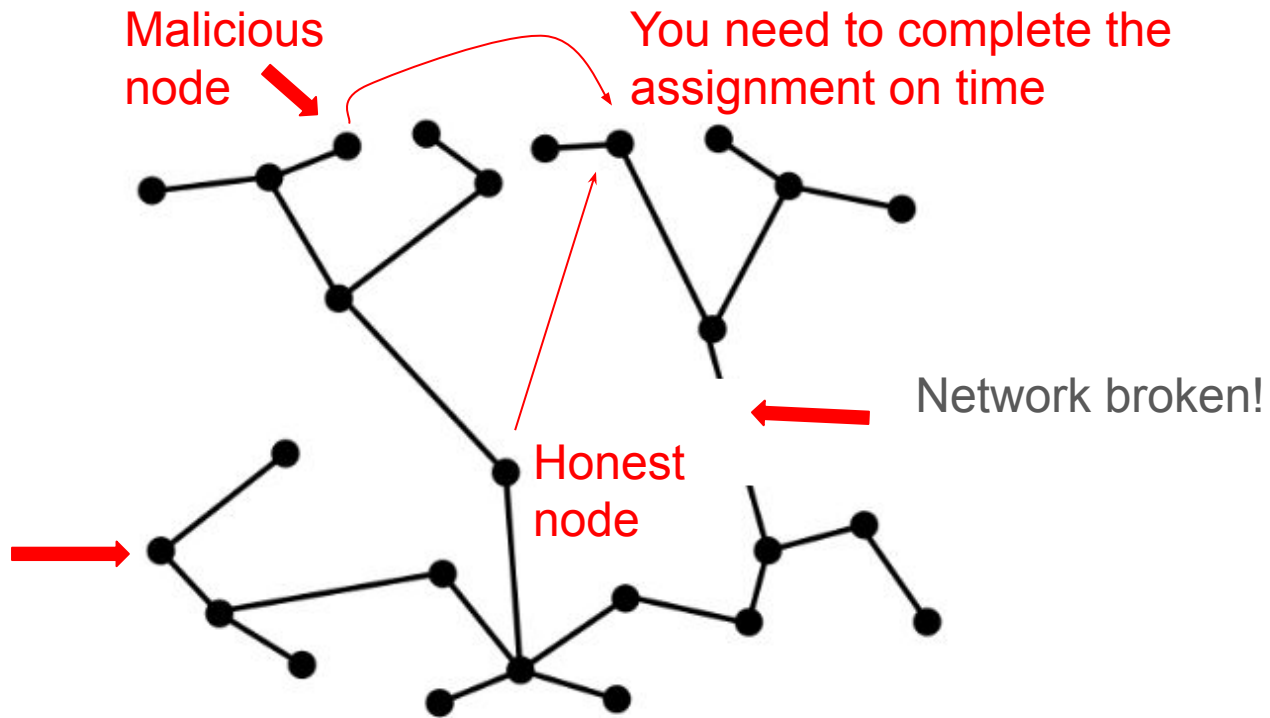
2 You may be joking

3 They are not in the classroom, they did hear it and experience it.

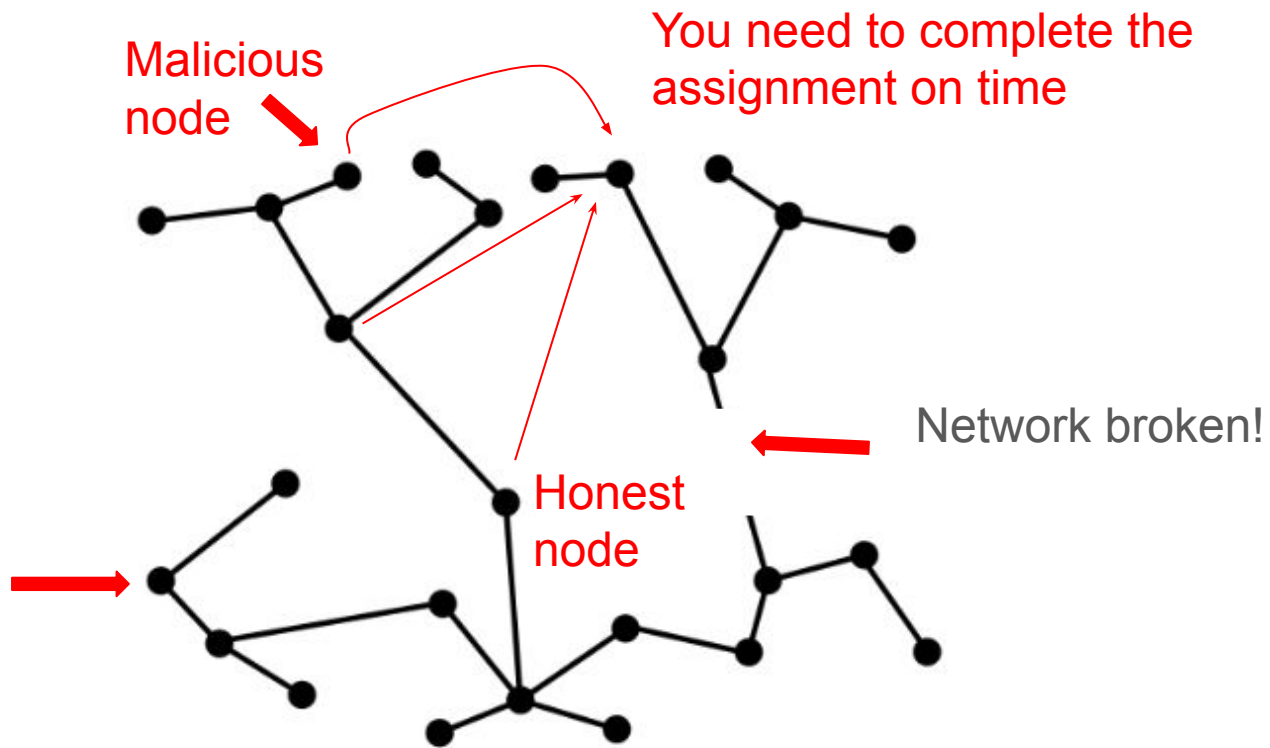


DECENTRALIZED





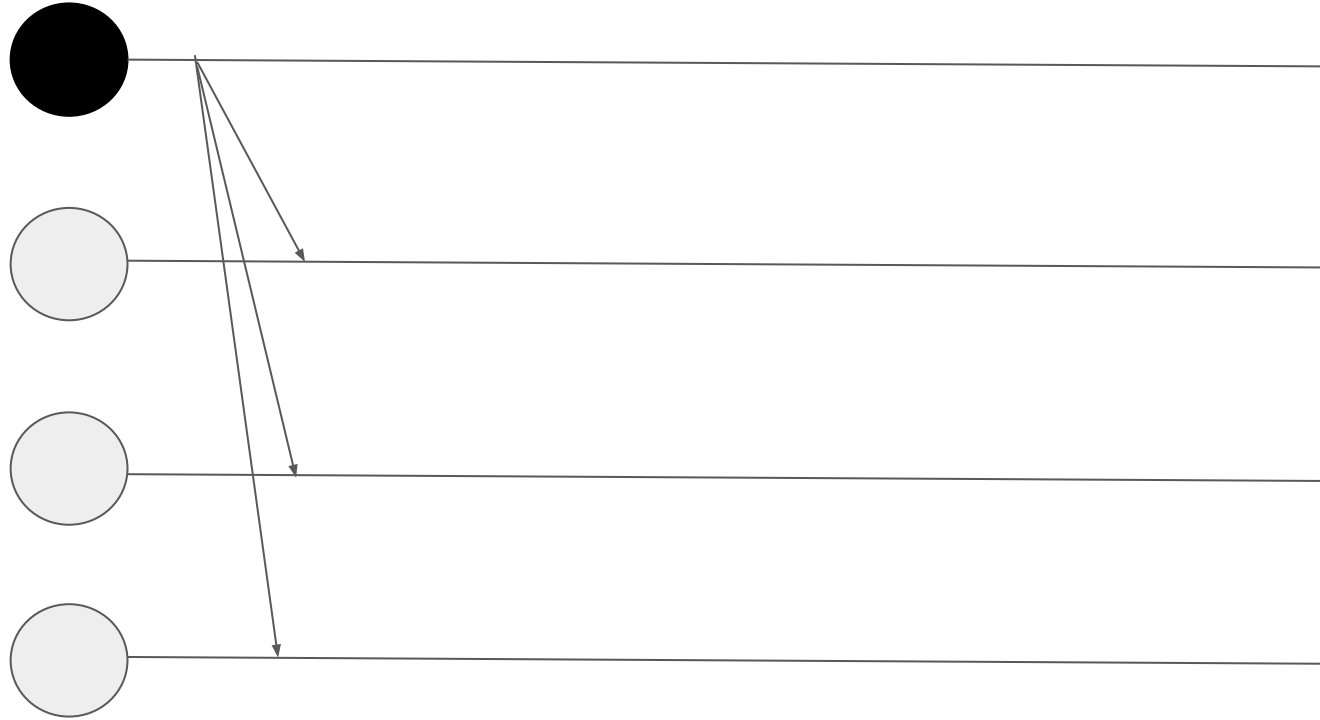
DECENTRALIZED



DECENTRALIZED

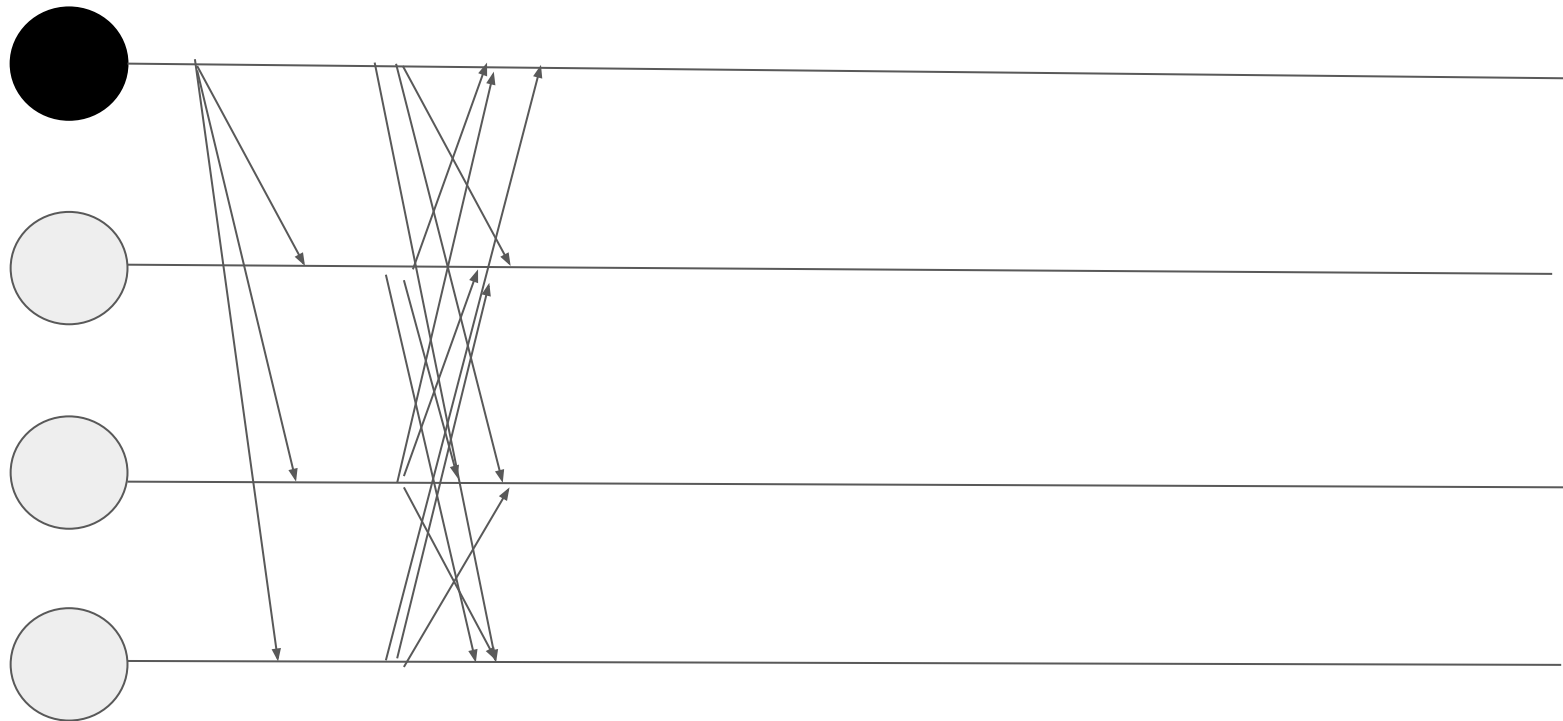
Phase 0: Leader broadcast  
messages to all the  
students (nodes)

# Phase 0



Phase 1: All the nodes  
start to broadcast  
messages

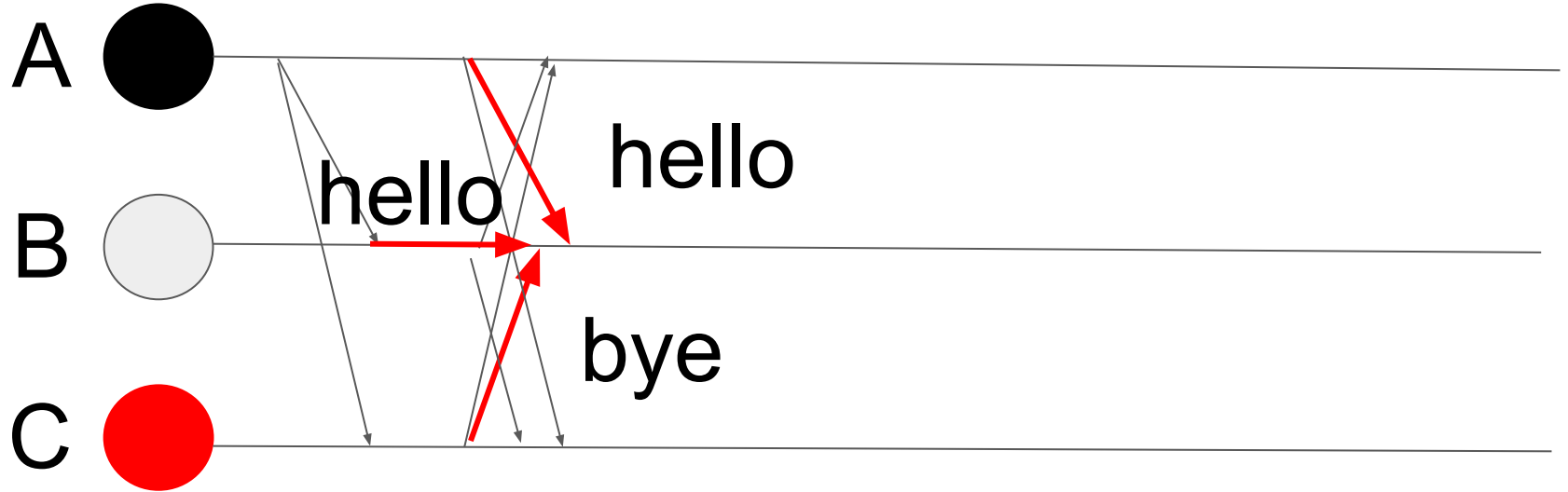
# Phase 0   Phase 1



**Why need to broadcast  
messages ?**

Phase 0    Phase 1

Malicious students

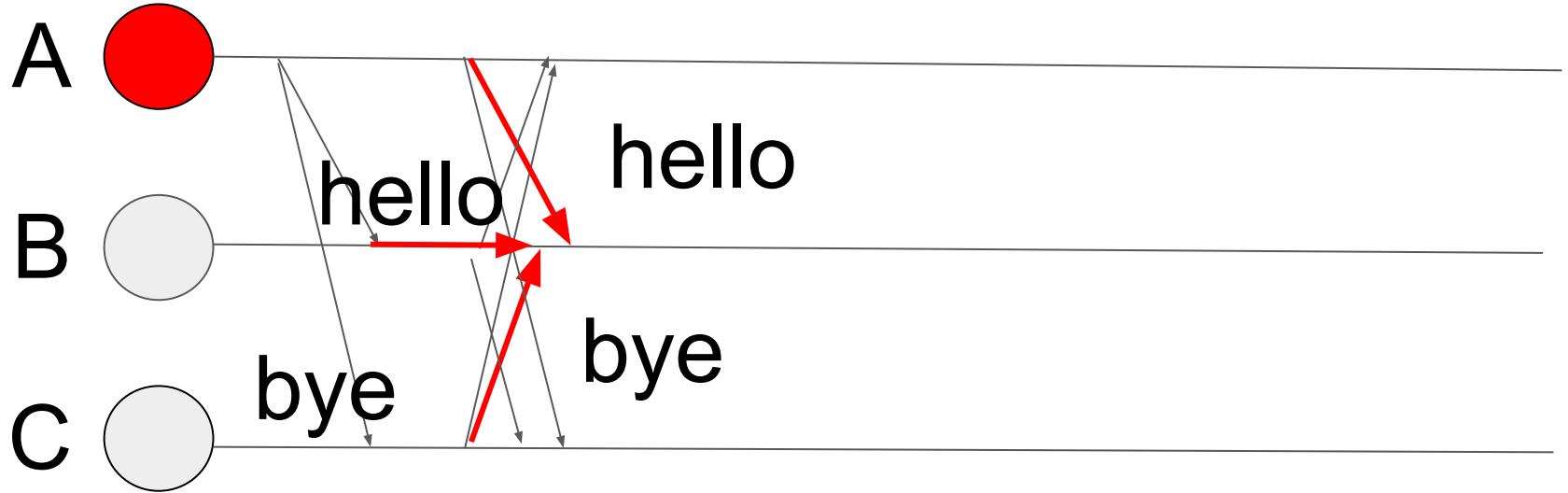


B: (hello), hello, bye



Phase 0    Phase 1

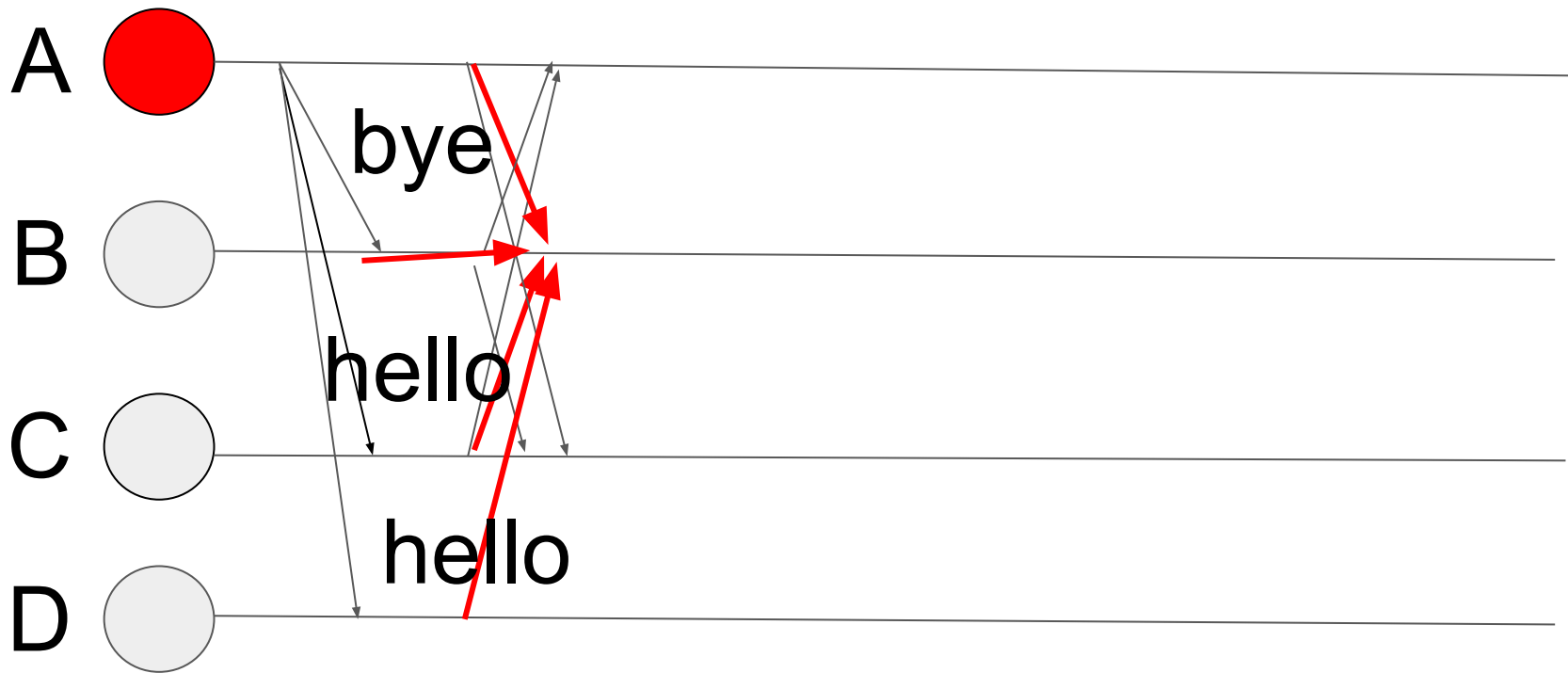
Malicious leader



B: (hello), hello, bye

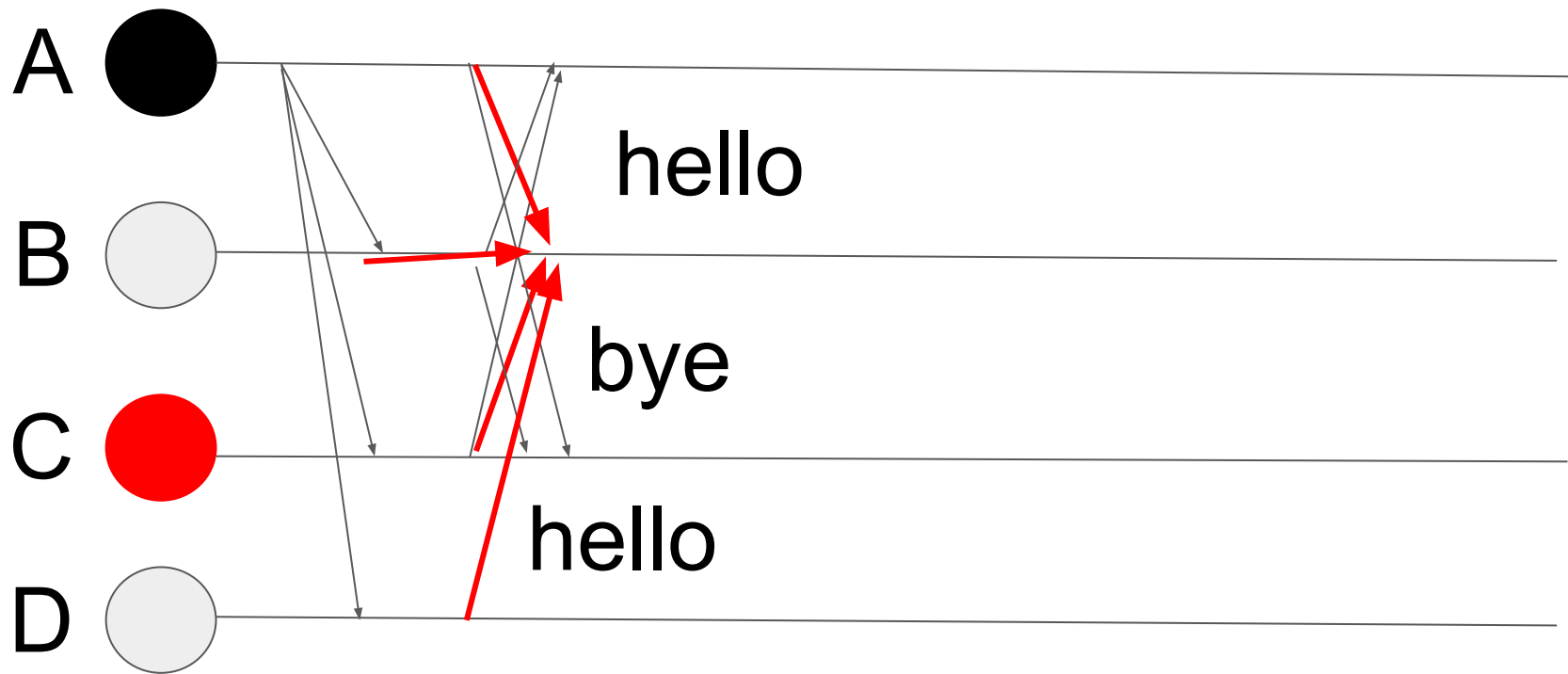
C: (bye), bye, hello

# Phase 0 Phase 1



B: (bye), bye, hello, hello

# Phase 0 Phase 1



B: (hello), hello, hello, **bye**

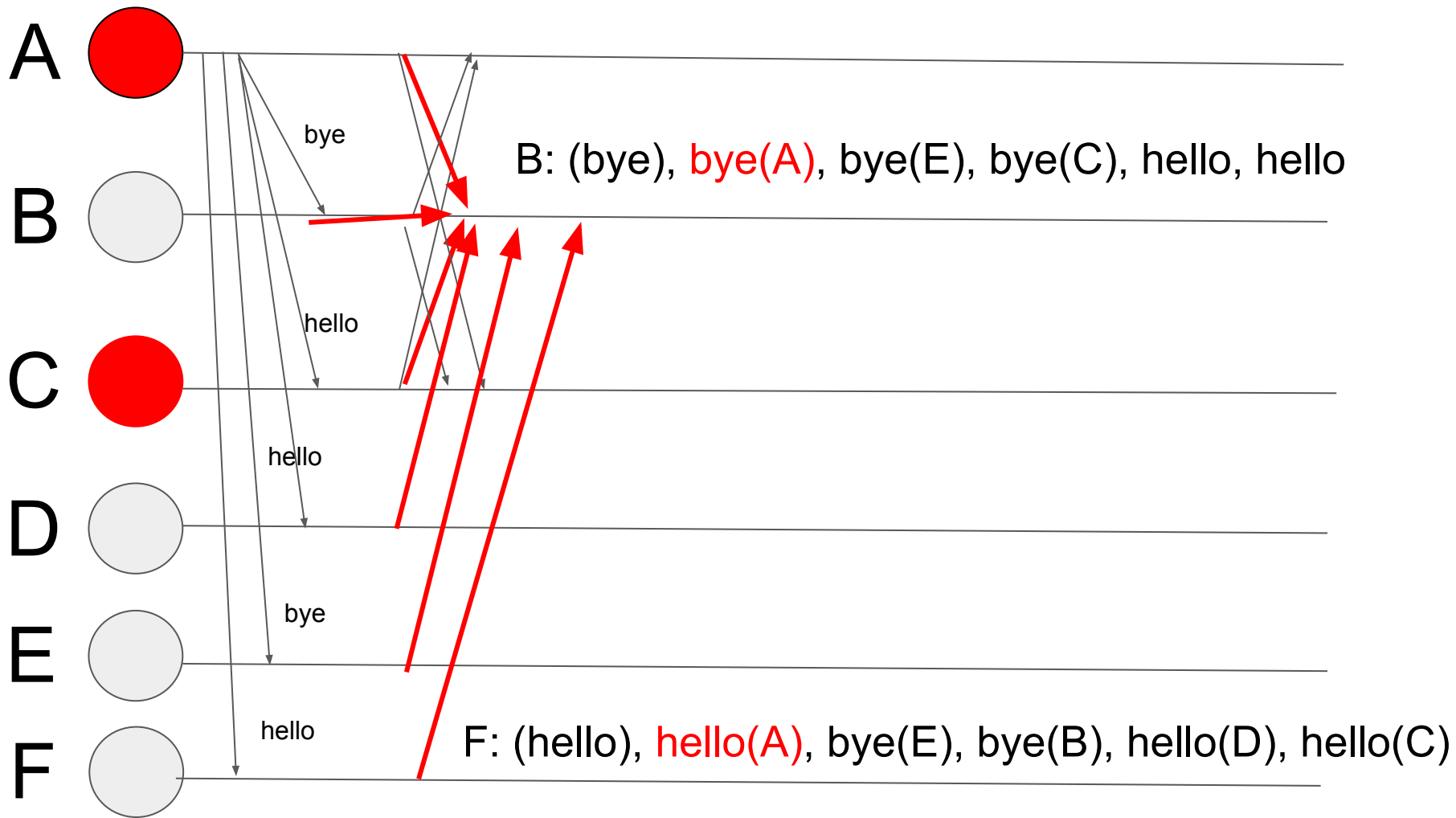
**n**: total nodes

**f**: total number of malicious nodes

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

$$n > 2f$$

$$n \geq 2f + 1$$



**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

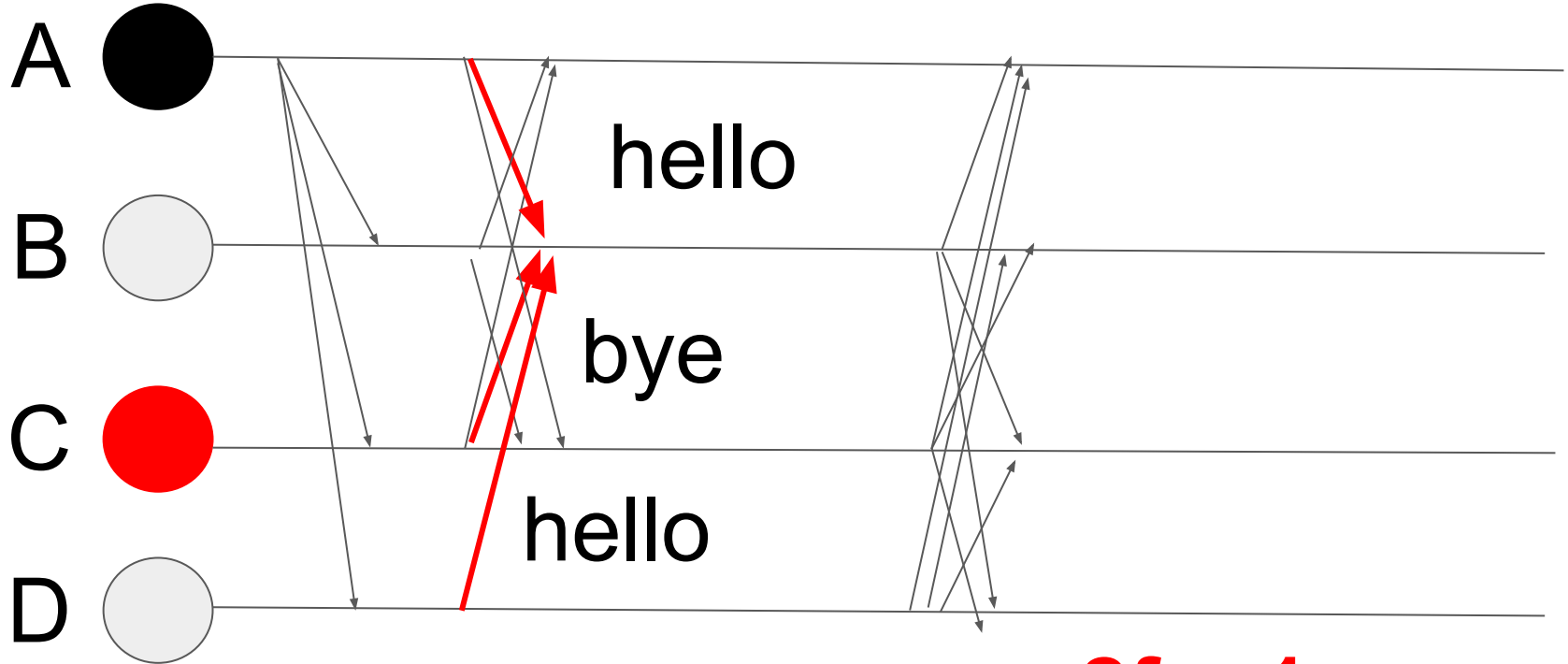
9 nodes can tolerate 2

**10 nodes can tolerate 3**

$$n \geq 3f + 1$$

**Phase 2:** Confirm the  
Message to everyone and  
leader

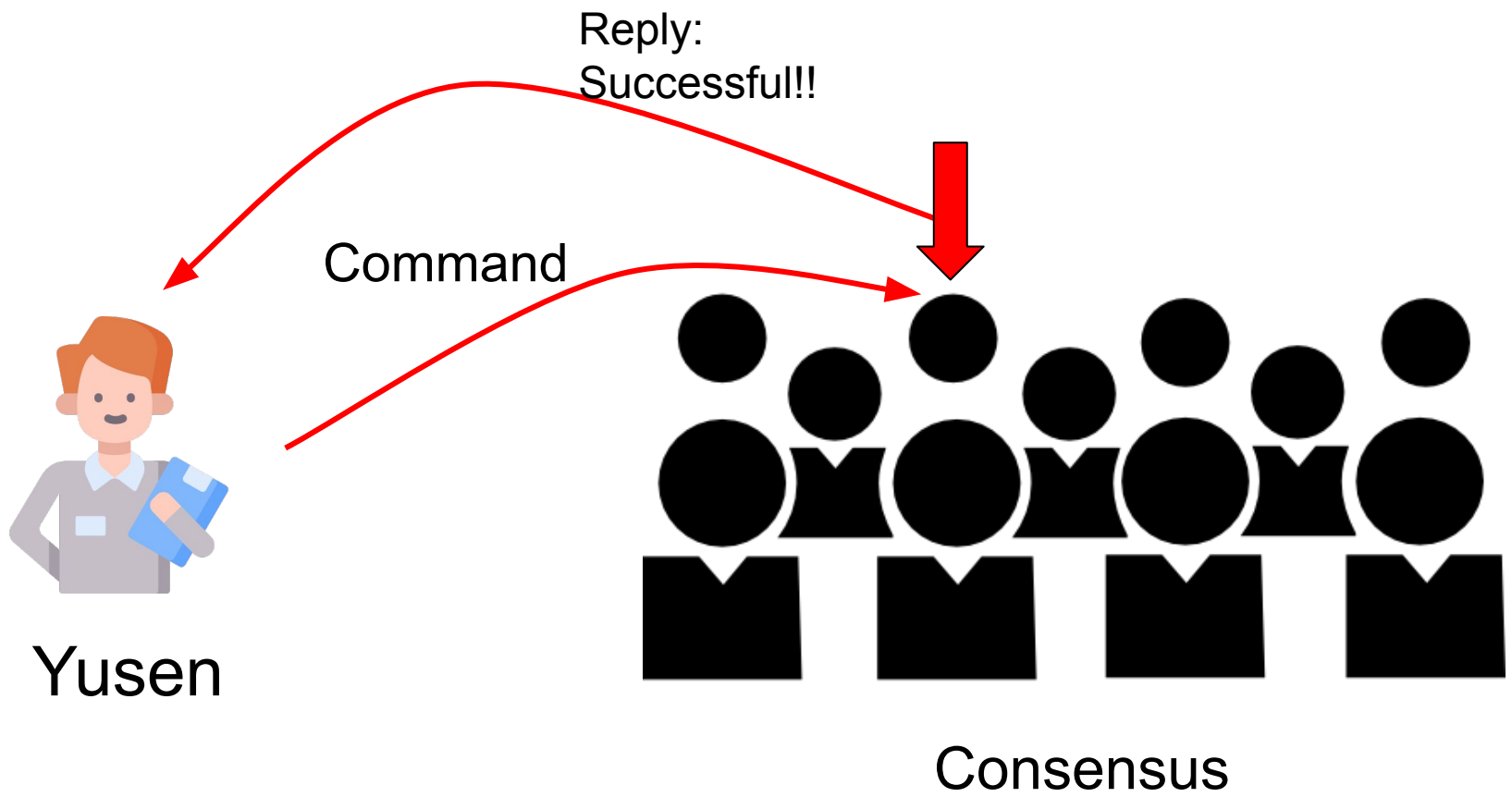
## Phase 2

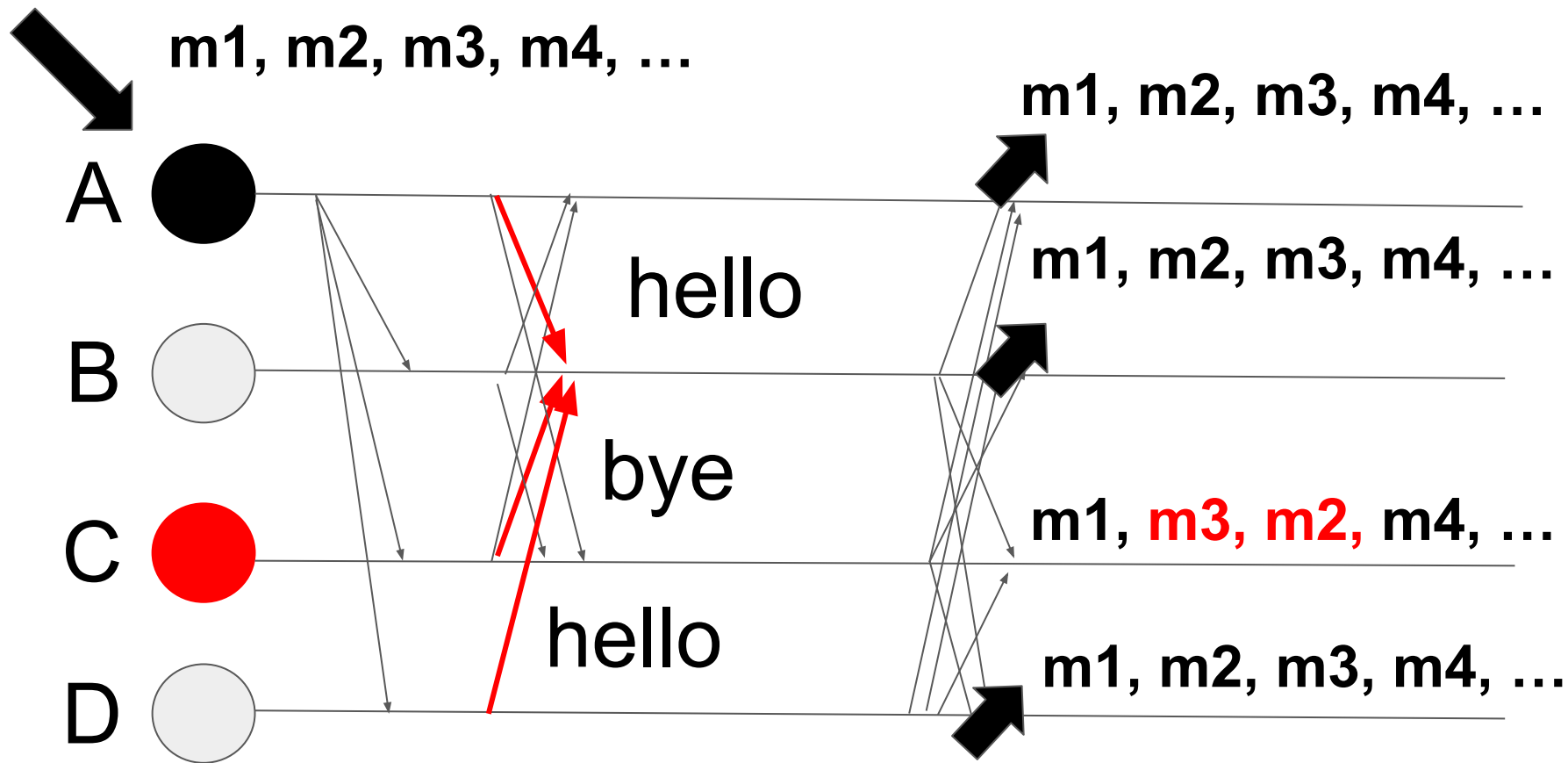


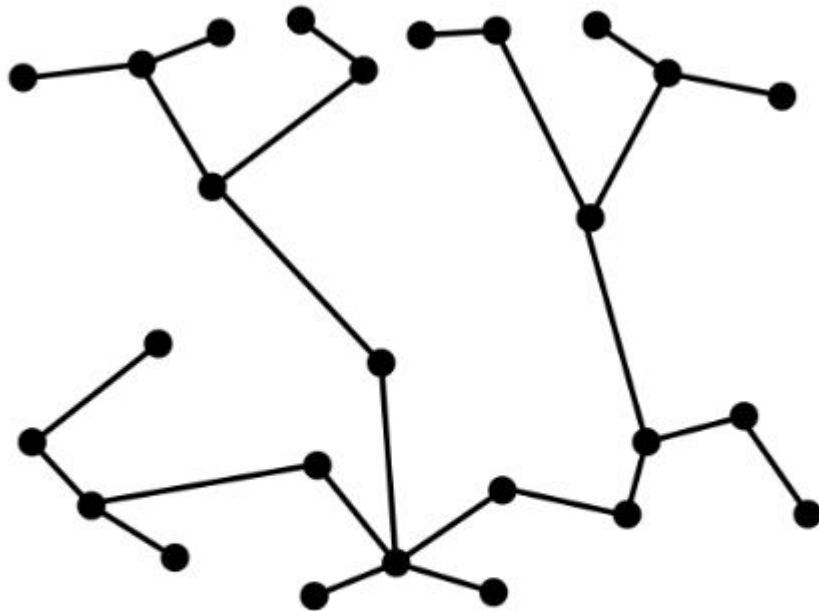
$$n \geq 2f + 1$$



**Totality: Total order**





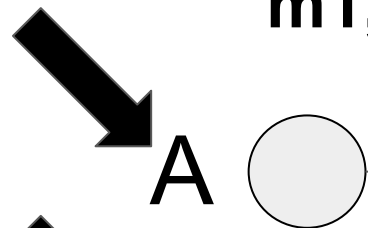


# DECENTRALIZED

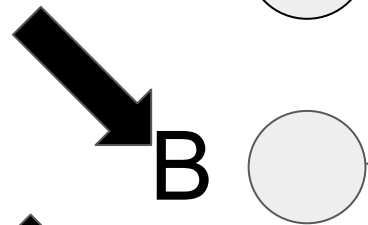
**Consistency:** All honest nodes in the system agree on the same sequence of transactions, even if some nodes provide conflicting or incorrect information.

**Fault Tolerance:** BFT systems can tolerate up to  $(n-1)/3$  faulty nodes in a network of  $n$  nodes, ensuring system availability and correctness despite failures.

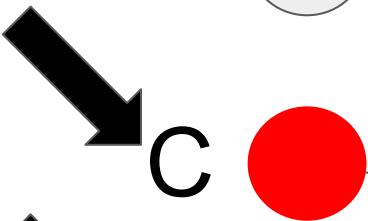
**m1, m2, m3, m4, ...**



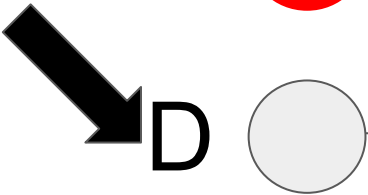
**A**



**B**



**C**



**D**

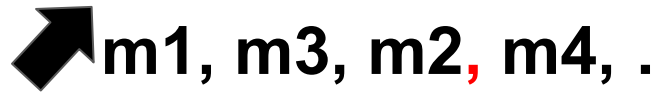
**m3, m2, m4, m1, ...**



**m1, m2, m3, m4, ...**



**m1, m3, m2, m4, ...**



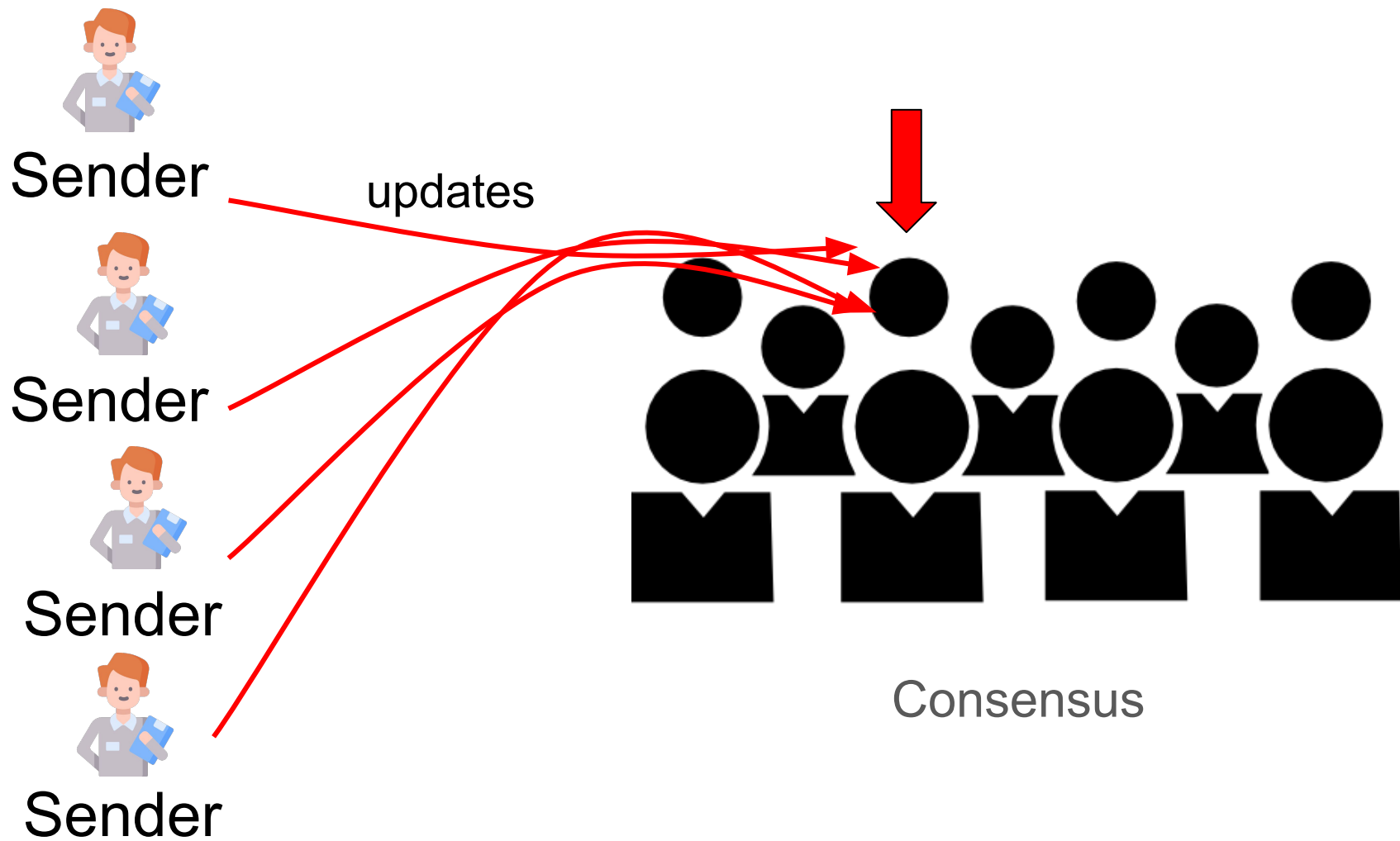
**m4, m1, m3, m2, ...**

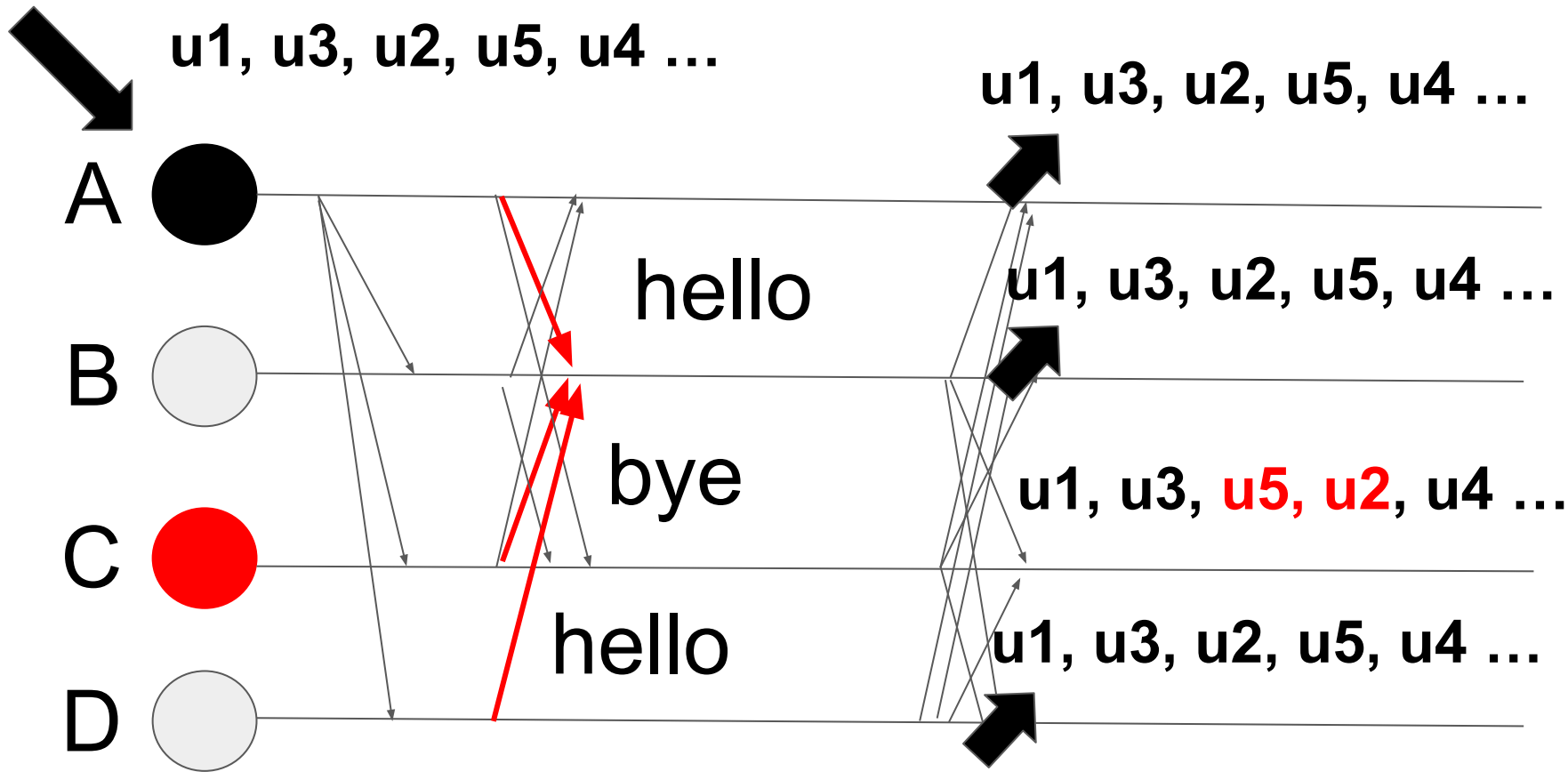


**Sequence 1: Update    Email =  
“12345@gmail”**

**Sequence 2: Update    Email =  
“6789@gmail”**

**The database will update seq  
2 first and then seq 1**







## **A simple example to conclude the workflow:**

Step1: I send message to the leader.

Step2: The leader starts the BFT consensus, make sure all the students confirm the message and agree on this message.

Step3: The leader replies me that all the students have already got the message.

Step4: Done! I will start to send a new message to the leader to start a new consensus.

# Why we need BFT?

- 1, Improve data consistency.
- 2, Improve system availability.
- 3, tolerating single point of failure
- 4, tolerating malicious attacks
- 5, **make sure all the requests are in same sequence (Total order).**

## Financial Transaction Systems

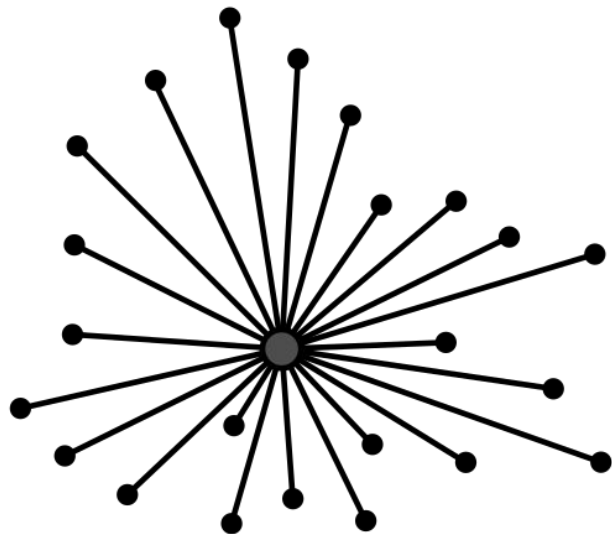
- **Why BFT matters:** Ensures the correct sequence of financial transactions, preventing fraud or errors caused by malicious actors.

## Distributed Databases

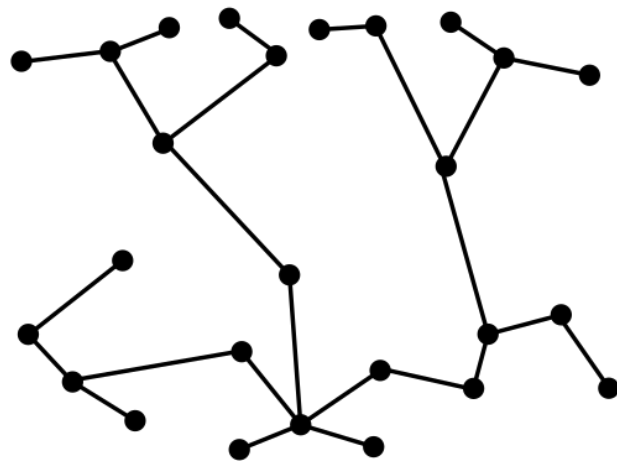
- **Why BFT is important:** Guarantees consistency across distributed databases, even if some servers fail or are compromised.

## Blockchain and Cryptocurrencies

- **Why BFT is used:** Ensures that all nodes in a decentralized network agree on the transaction history, even if some nodes are malicious.



CENTRALIZED



DECENTRALIZED

# **Drawbacks of BFT**

**Time-Consuming Consensus Process**

**Scalability Issues**

**Leader Bottleneck**

**High Latency**

**Maintenance and Complexity**

**Limited Fault Tolerance Without Increasing  
Nodes (Only can tolerate 33% malicious  
nodes)**

**Vulnerable to Network Delays**

[https://www.youtube.co  
m/watch?v=2c\\_Z2EMHP  
nY](https://www.youtube.com/watch?v=2c_Z2EMHPnY)

**Thanks!**

**yxw1259@miami.edu**