Quadratic worst-case message complexity for State Machine Replication in the partial synchrony model

ANDREW LEWIS-PYE, London School of Economics, UK

Presented by:

Devavrat Singh Bisht
Kunjal Agrawal
Priyadharshini Ganeshkumar
Purva Khadke
Vrushali Harane

Important terms

1. **Consistency :** Consistency ensures that all correct (non-faulty) nodes in the system agree on the same state or value.

2. **Liveness :** Liveness ensures that the system continues to make progress and eventually reaches a decision. It guarantees that the system does not "stall" or become stuck indefinitely.

Important terms

3. **Epoch:** An epoch is a higher-level structure that consists of multiple views. In each epoch, the leader role rotates across f+1 replicas to ensure that at least one correct leader will be chosen within the epoch.

4. **View:** A view is a phase of the protocol where one replica (node) acts as the leader and proposes blocks for the other replicas to validate and confirm. If the leader is faulty or no progress is made, the system moves to the next view with a different leader.

Communication Models

There are 3 standard communication models in distributed systems :

1. **Synchronous** :
   - There is a fixed upper bound Δ on time it takes to for any message to be delivered between the nodes.
   - All replicas know the maximum time a message can take to reach its destination.
2. **Partial Synchrony :**
   - The upper bound Δ on message delivery only applies after an unknown Global Synchronization Time (GST).
   - Before GST, message delivery can be unpredictable and take arbitrary amounts of time. After GST, message delays are bounded by Δ.

Communication Models

2. **Partial Synchrony** :

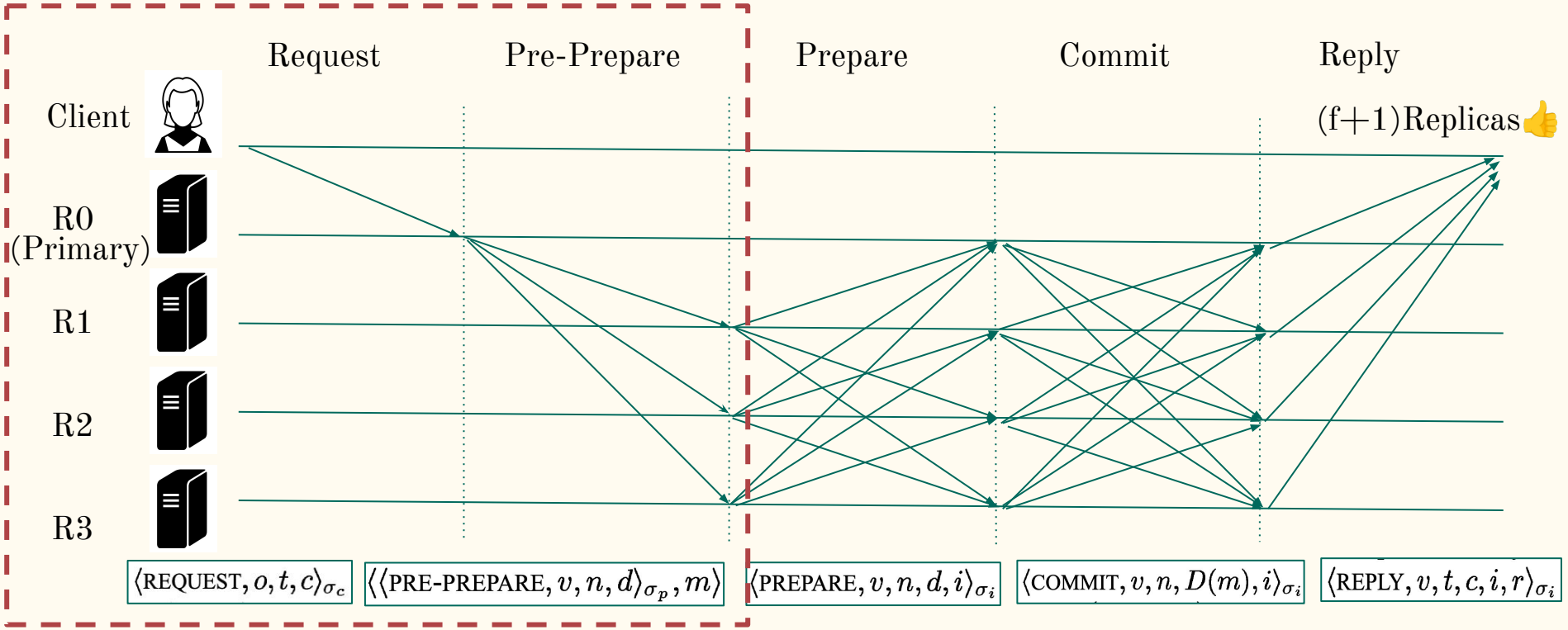
- Before GST, the network behaves unpredictably, and messages may take arbitrarily long to arrive.
- After GST, messages are delivered within a known bound Δ.
- Nodes do not start the protocol simultaneously, but all correct nodes are assumed to join before GST

3. **Asynchronous :**

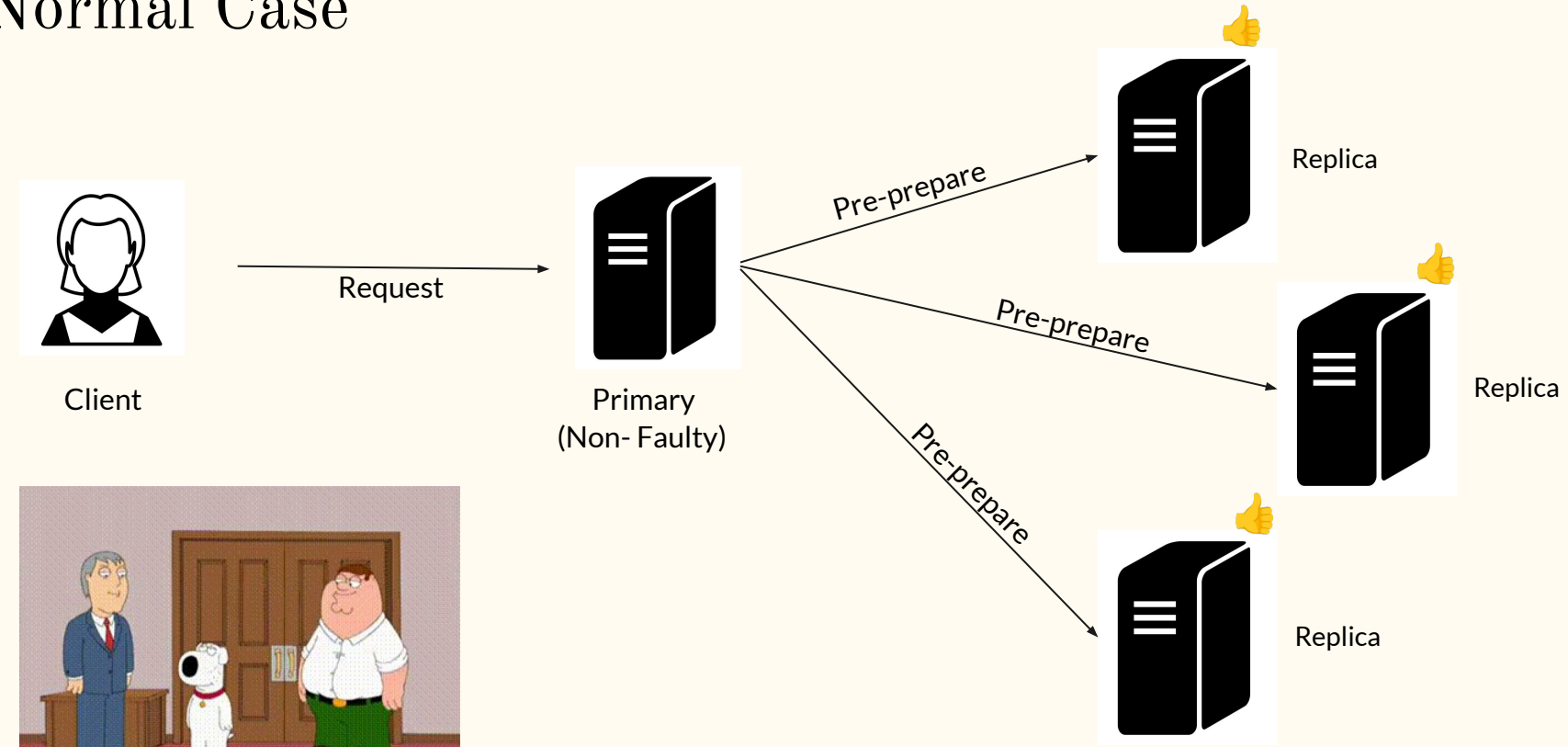
- In the asynchronous model, there is no bound on the time it takes for messages to be delivered.
- Messages can take any finite amount of time to arrive, and there is no guarantee about how long this will be.
- There is no timing information or synchronization between replicas.

Let’s revisit PBFT

PBFT : 3 Phases and Quadratic Message Transfer



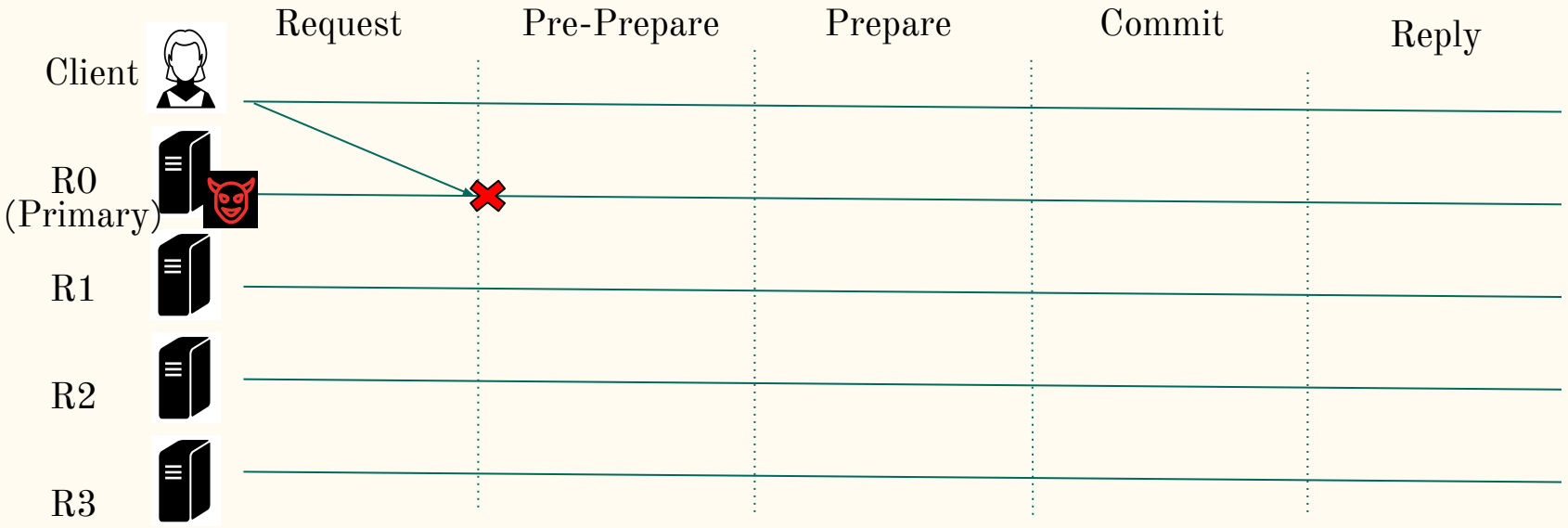
Normal Case


Client
Request
Primary
(Non- Faulty)
Pre-prepare
Pre-prepare
Pre-prepare
Replica
Replica
Replica

EVERYBODY HAPPY!

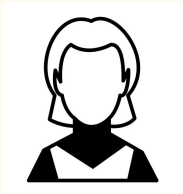
What happens when Primary is Faulty?

- Time to change the Primary - View Change



View Change in PBFT

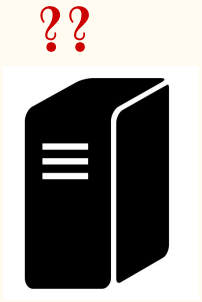
Faulty Primary : View Change



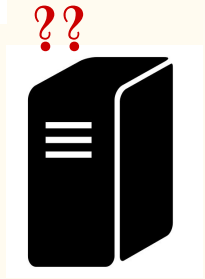
Client

Request

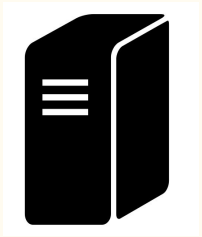
Primary



Replica



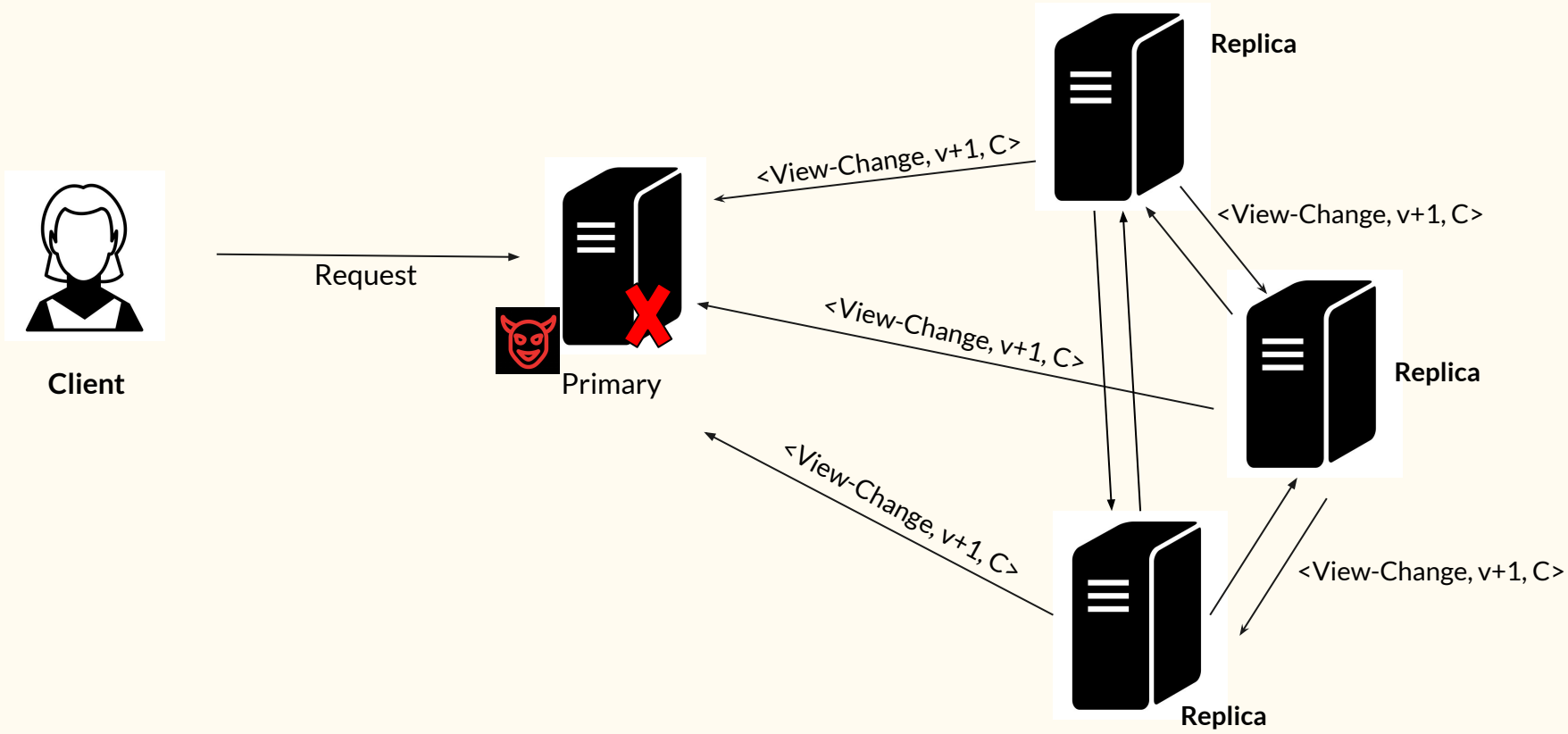
Replica



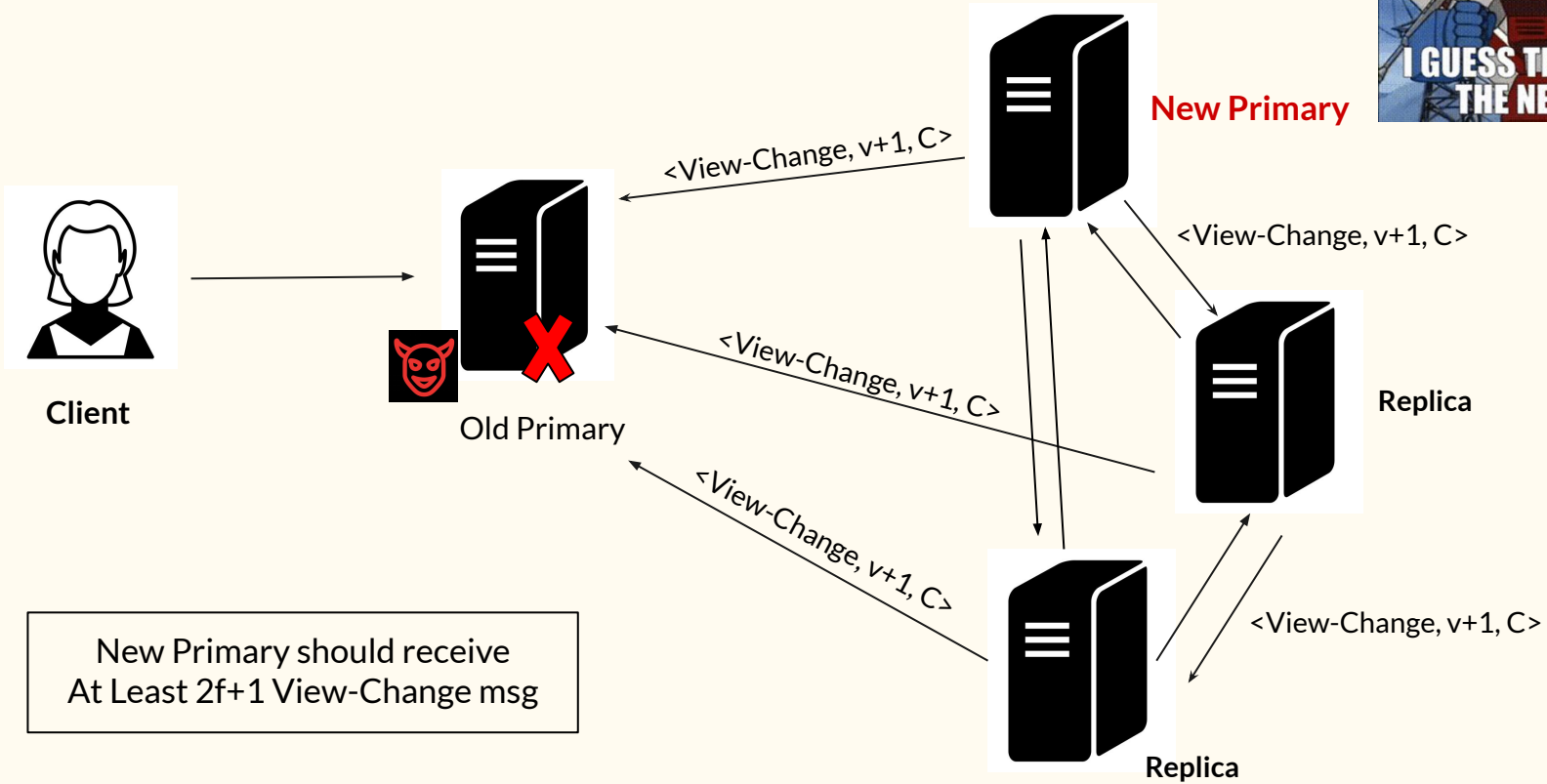
Replica

Timeout!!

Faulty Primary : View Change

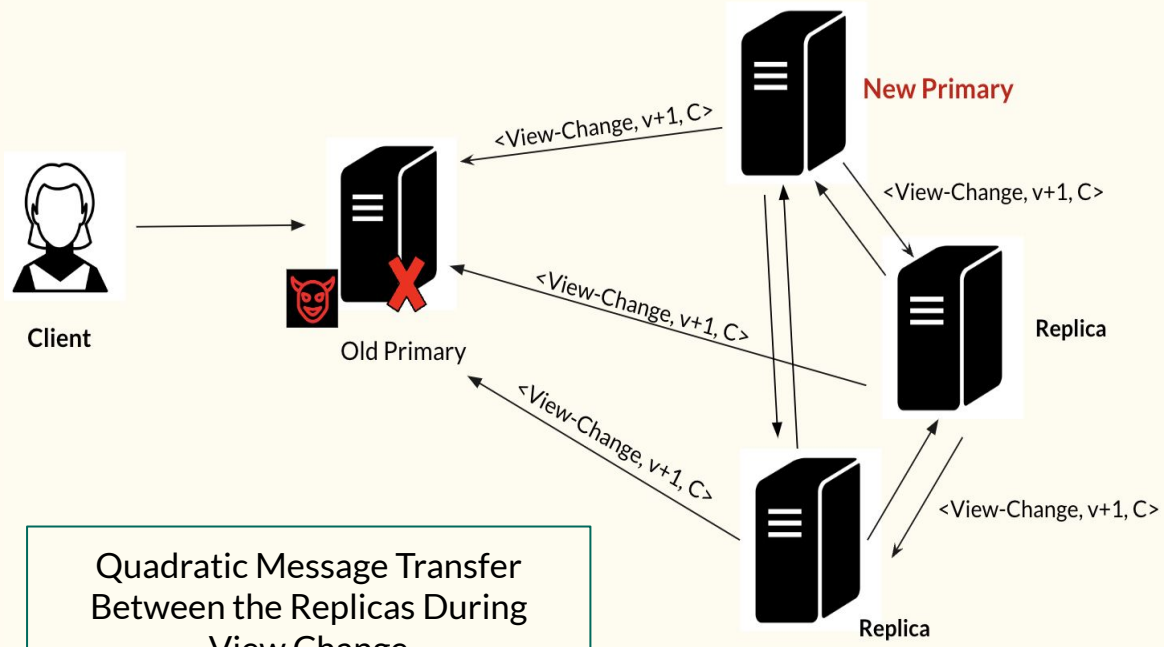


Faulty Primary : View Change



New Primary should receive
At Least 2f+1 View-Change msg

View Change Worst Case Message Complexity : **O(n^3)**



Quadratic Message Transfer Between the Replicas During View Change

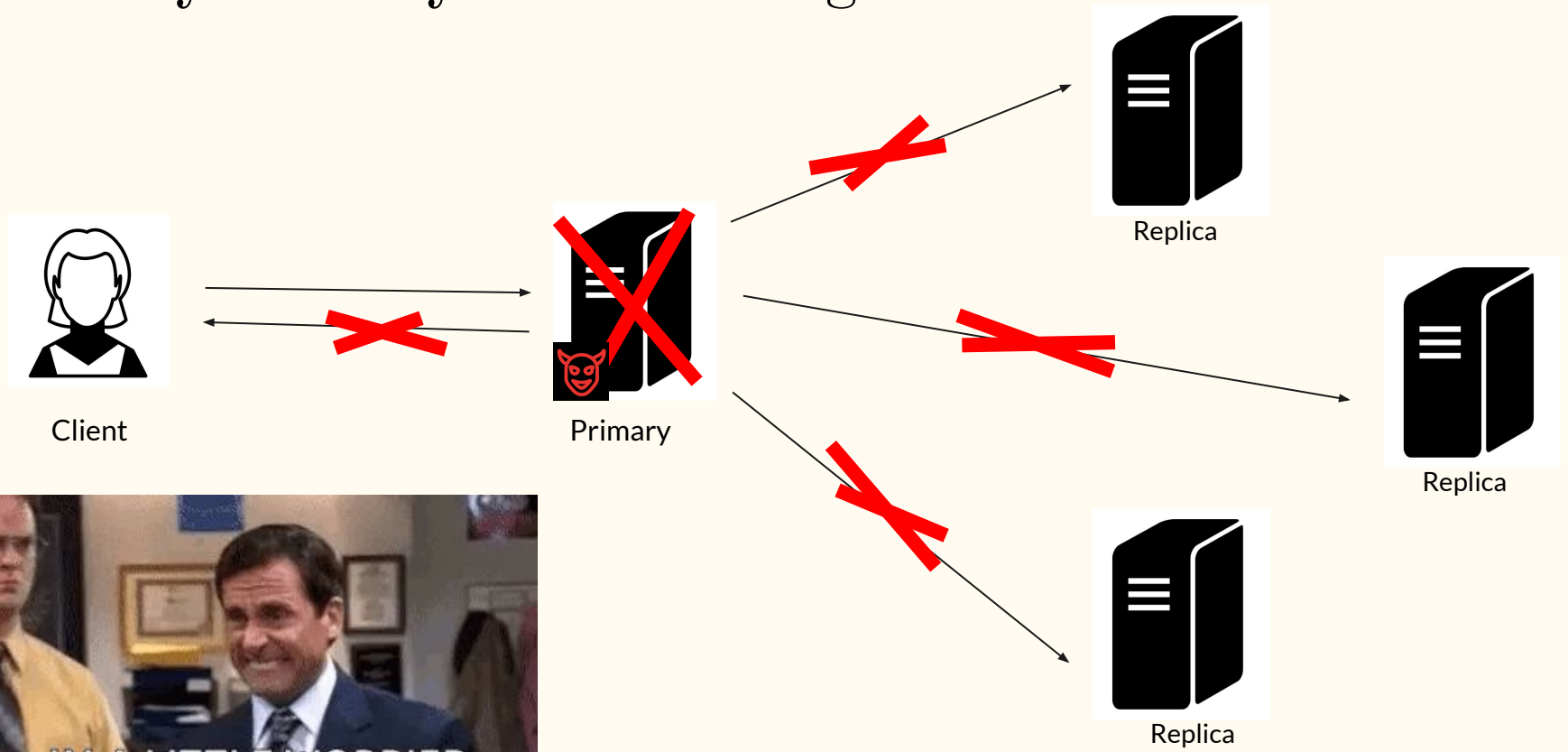
What if the new Primary is also faulty?

$$n >= 3f + 1$$

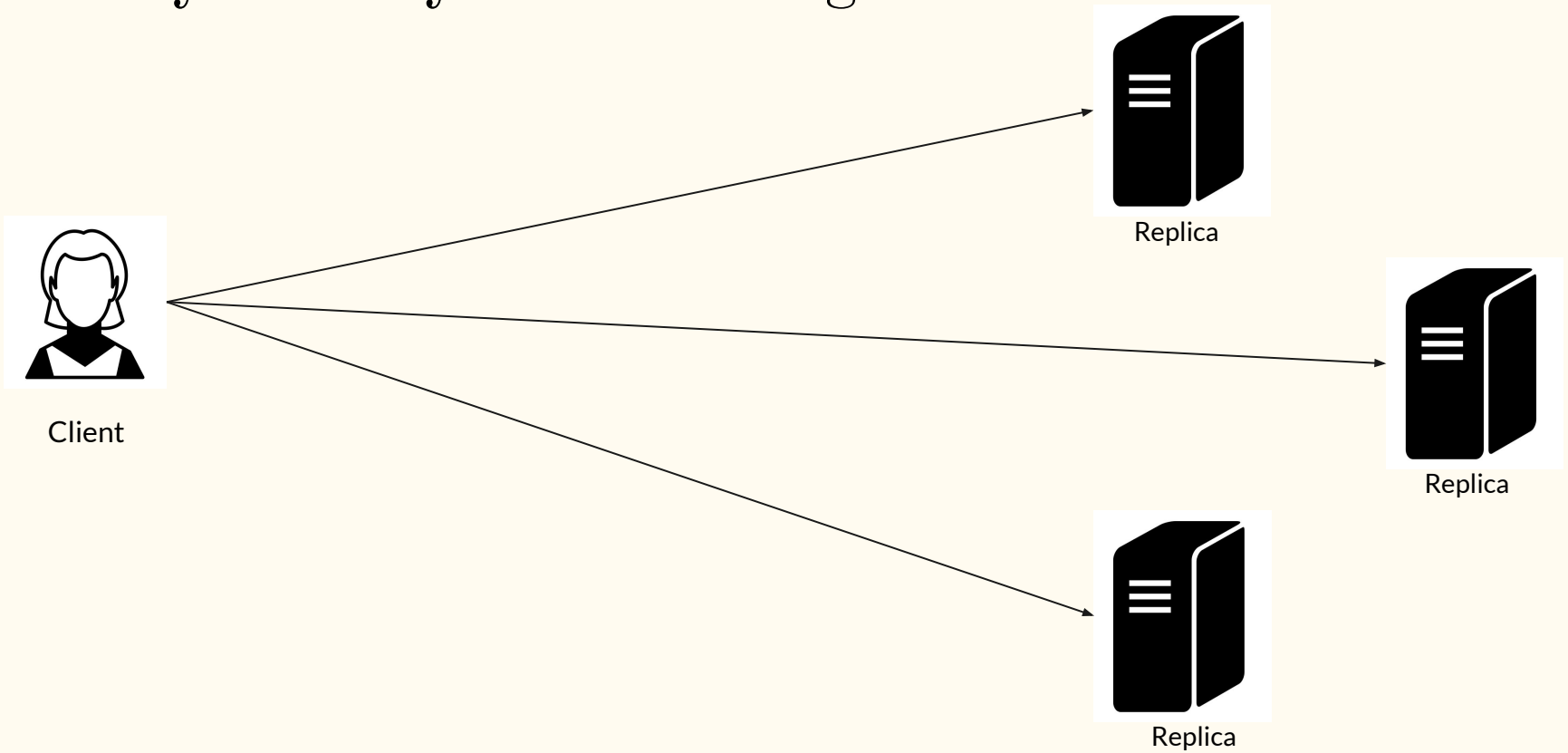
$$f <= (n-1)/3$$

O(n^2) * O(f+1)

Faulty Primary : View Change



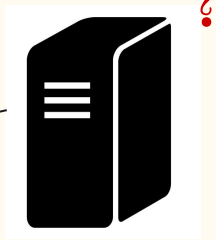
Faulty Primary : View Change



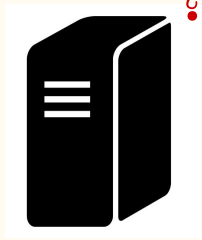
Faulty Primary : View Change

Client

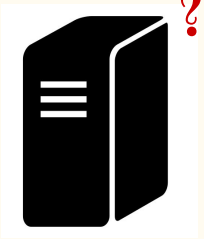
Primary



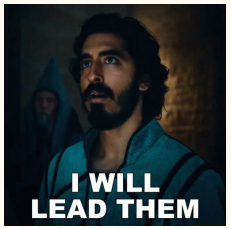
New Leader



Replica



Replica

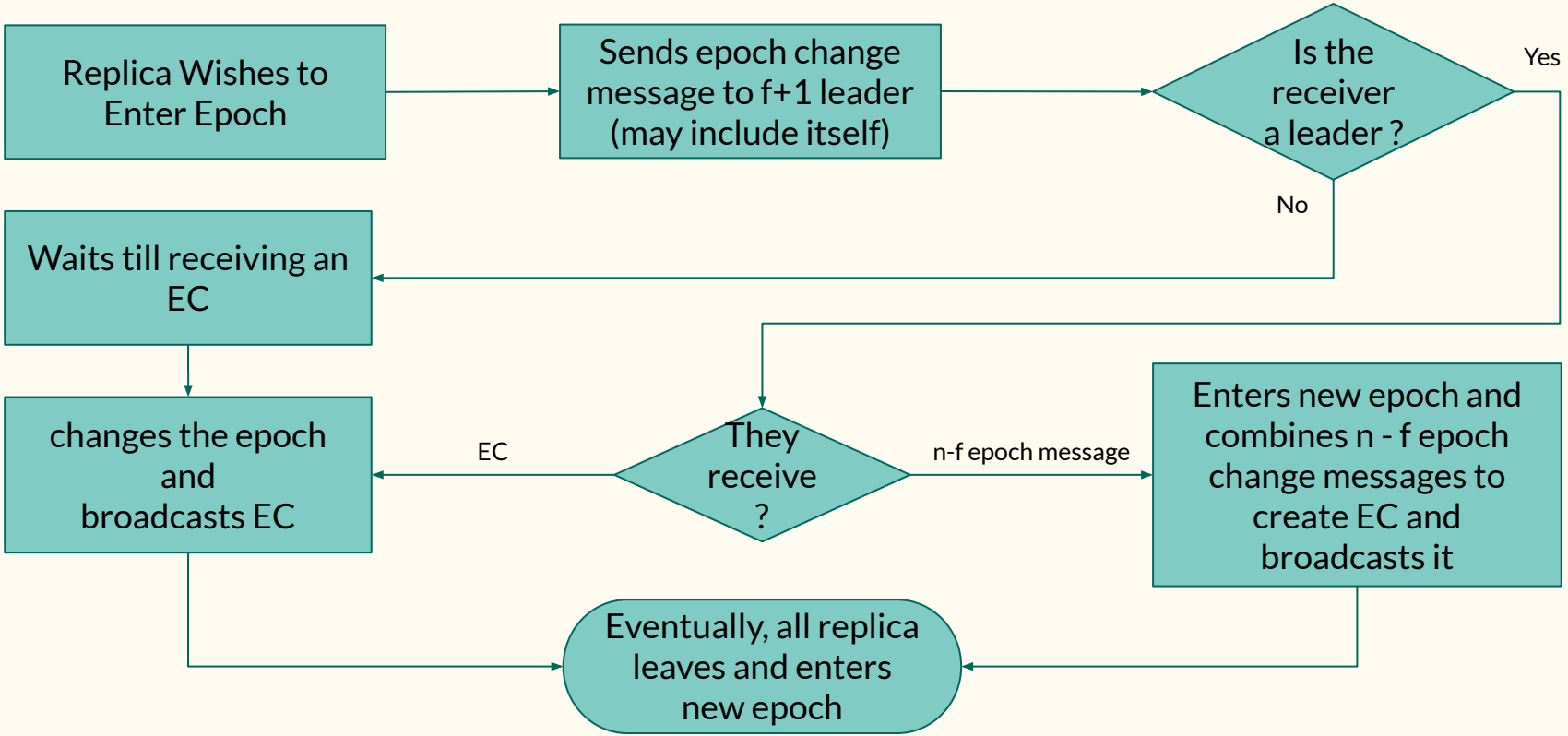


Epoch Change

When and why Epoch Changes ?

- An epoch changes only after all the views in the epoch has been completed.
- In ideal scenario a view lasts for a time of 12Δ.
- An epoch has $f+1$ leaders, thus in ideal case it lasts for a time of $12(f+1)\Delta$.
- Depending on the blockchain, the epoch might be used for:
  - Distributing the rewards
  - Take snapshots (balances, transactions, etc) for checkpointing and recovery in case of issues.
  - Network parameter adjustments, eg. the complexity of solving the problem increases in Bitcoin
  - Updating validator nodes.

Flowchart for Epoch Change



Steps for Epoch Change

- When any replica wishes to enter epoch e, it sends an “epoch e” message to each of the $f + 1$ leaders of epoch e (potentially including itself).

Instructions for Epoch Change : For Leader

- If a replica is one of the f +1 leaders for epoch e, they enter epoch e if they are presently in a lower epoch and either:
    - (a) They receive n — f epoch e messages (from different replicas). Then the leader combines the n — f epoch e messages into a message with a single threshold signature, which we call an EC for epoch e, and the leader then broadcasts that EC.

OR

    - (b) They receive an EC for epoch e. Then the leader simply broadcasts the EC.

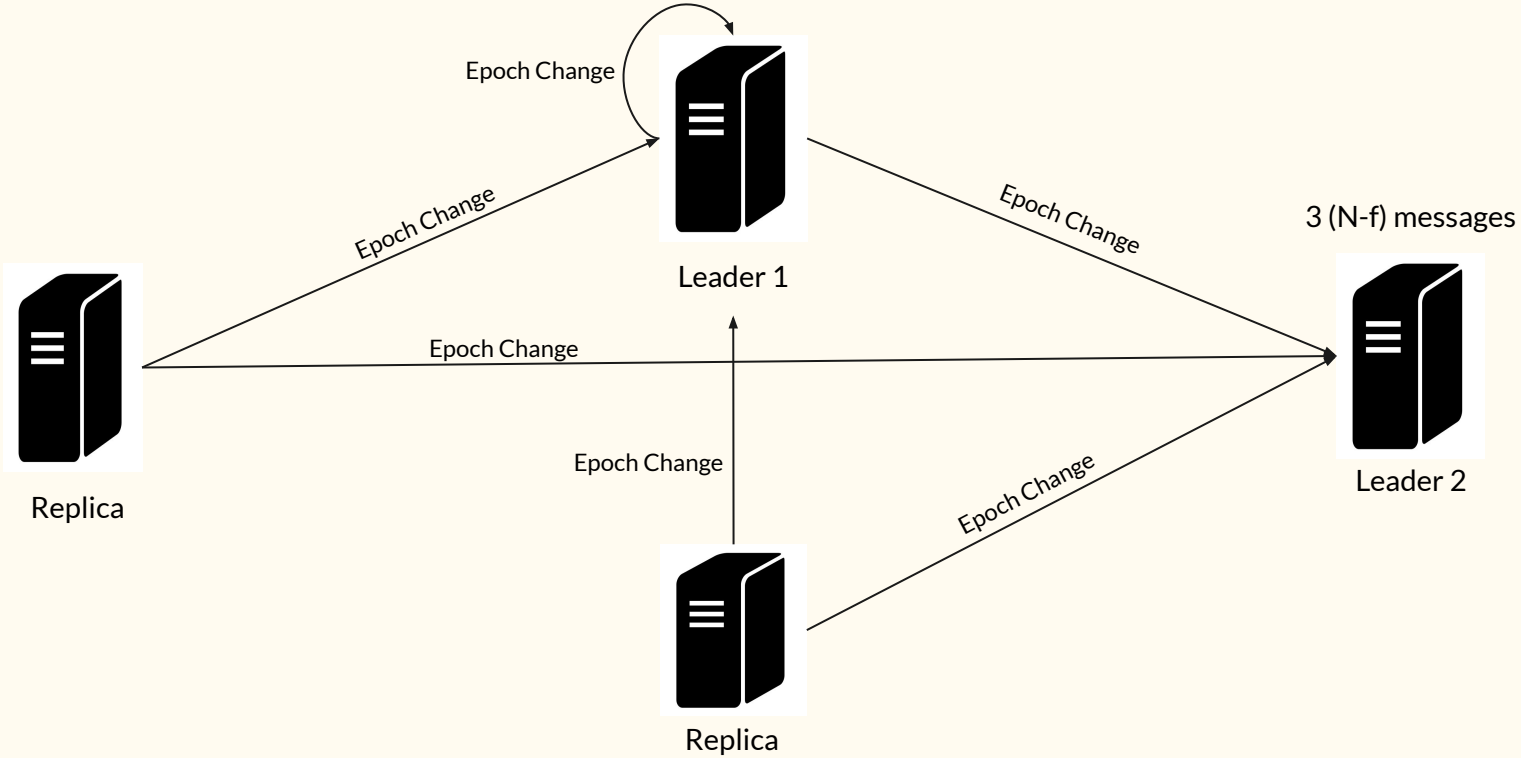
Instructions for Epoch Change : For non-Leader

- Each replica which is not a leader for epoch e enters the epoch if they are presently in a lower epoch and if they see an EC for epoch e. Upon entering epoch e, they broadcast that EC for epoch e to all replicas.
- A replica leaves an epoch when it enters a higher epoch.

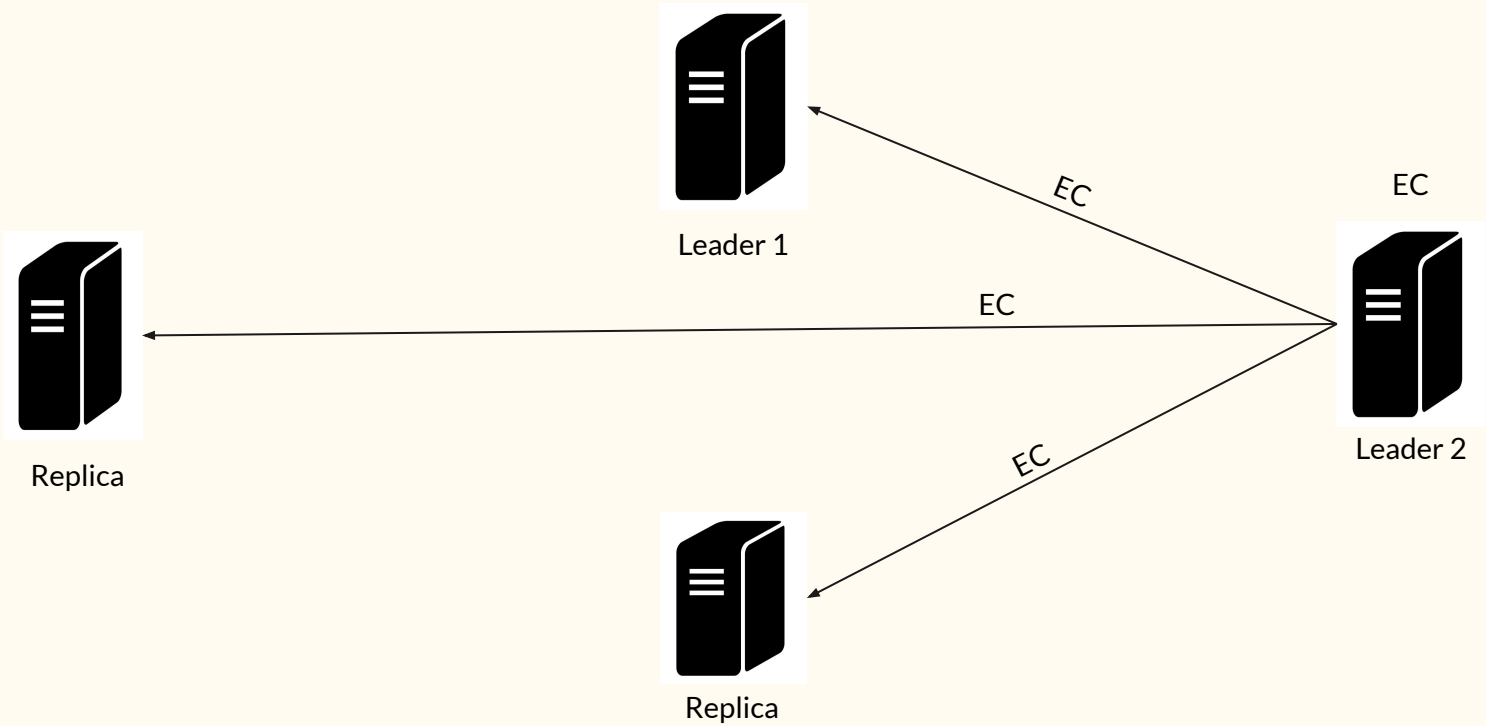
Instructions for Epoch Change Ensures:

1. If a correct replica i enters epoch e at time t, then all correct replicas will have entered an epoch $e' \geq e$ by time $\max\{t, GST\} + \Delta$, since they will see the EC sent by i by that time.
2. Each epoch change has message complexity $O(n2)$. Since each replica sends O(f) messages upon wishing to enter epoch e and $O(n)$ messages upon entering epoch e.

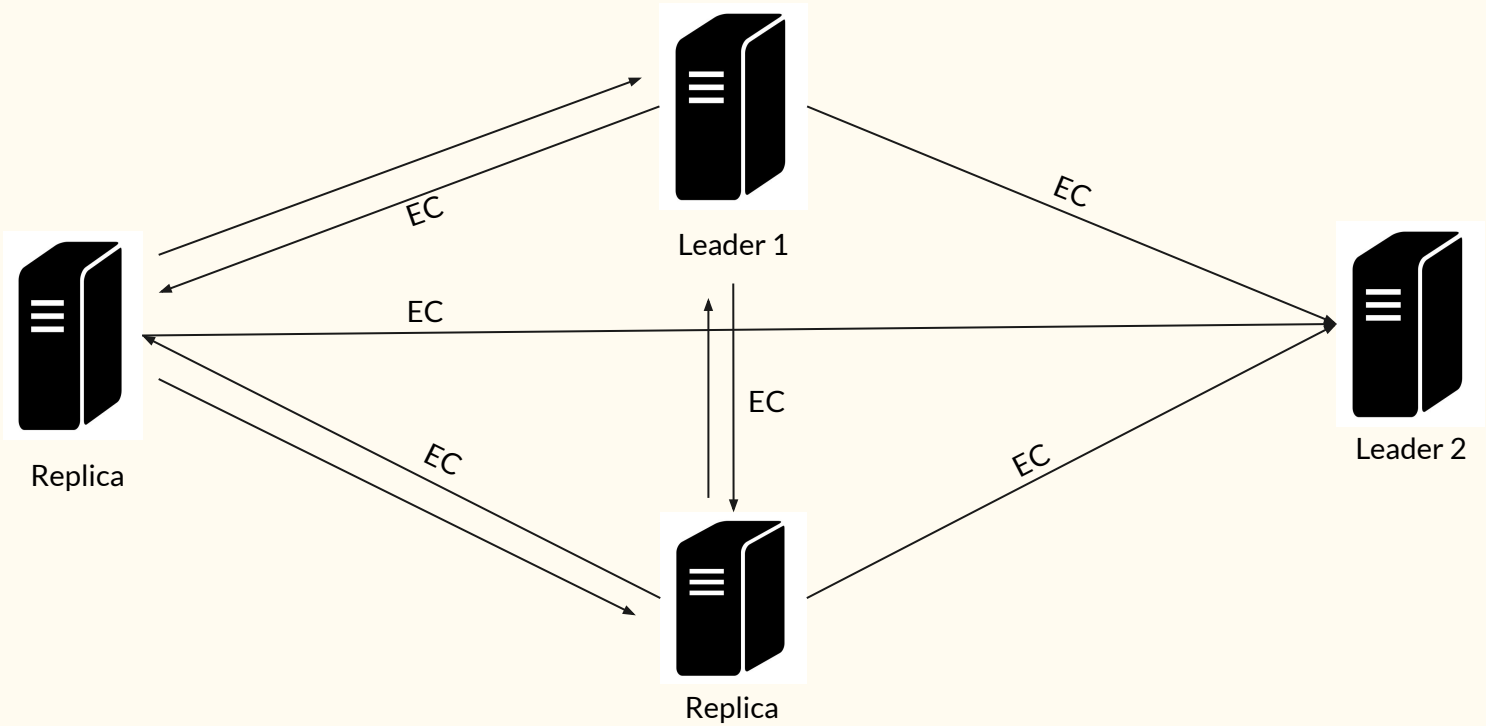
Example of Epoch Change



Example of Epoch Change



Example of Epoch Change

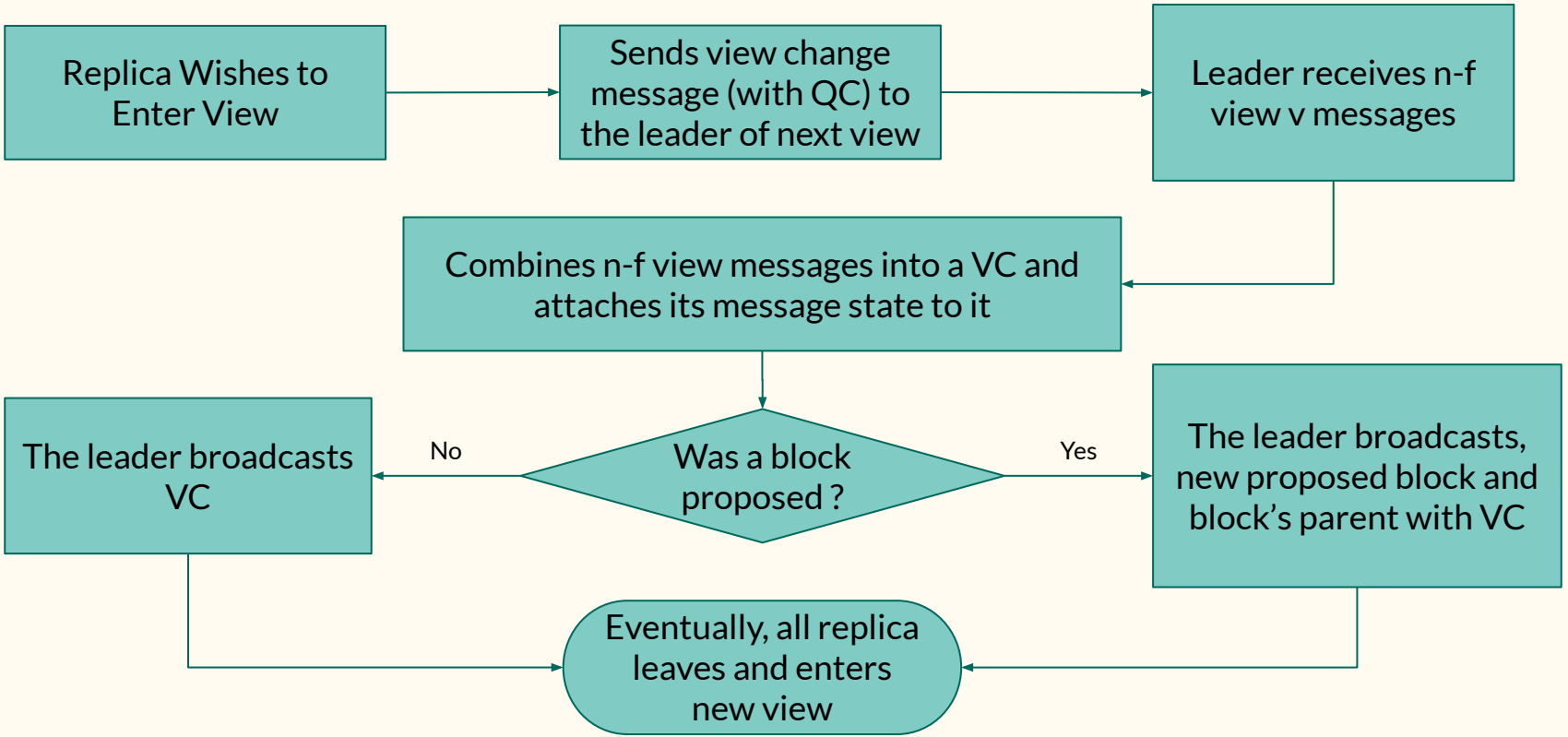


View Change

When and why View Changes ?

- A replica sees a block corresponding to view *v-1* confirmed.
- Its personal timer reaches *12Δ* .
- A leader failure happens.

Flowchart for View Change



The instructions for view changes

- When each replica starts epoch e, they reset their personal timer to 0 and begin by wishing to enter view 0.
- A replica wishes to enter view $v > 0$ of epoch e at the first point at which it is in a lower view (for this epoch) and either:
  - (a) It sees a block corresponding to view $v - 1$ confirmed,

    OR

  - (b) Its personal timer reaches $12v\Delta$ (the idea to have in mind here is that each view will last for at most 12Δ after GST when leaders are correct). We'll refer to $12v\Delta$ as the trigger time for view v.

The instructions for view changes

- When any replica wishes to enter view v, it sends a view v message to the leader of view v. It attaches to that message its present message state (which includes the highest QC it has seen).
- The leader of view v of epoch e enters the view if they are in epoch e and both:
  - (a) $i = 0$ or they are presently in a lower view

    And

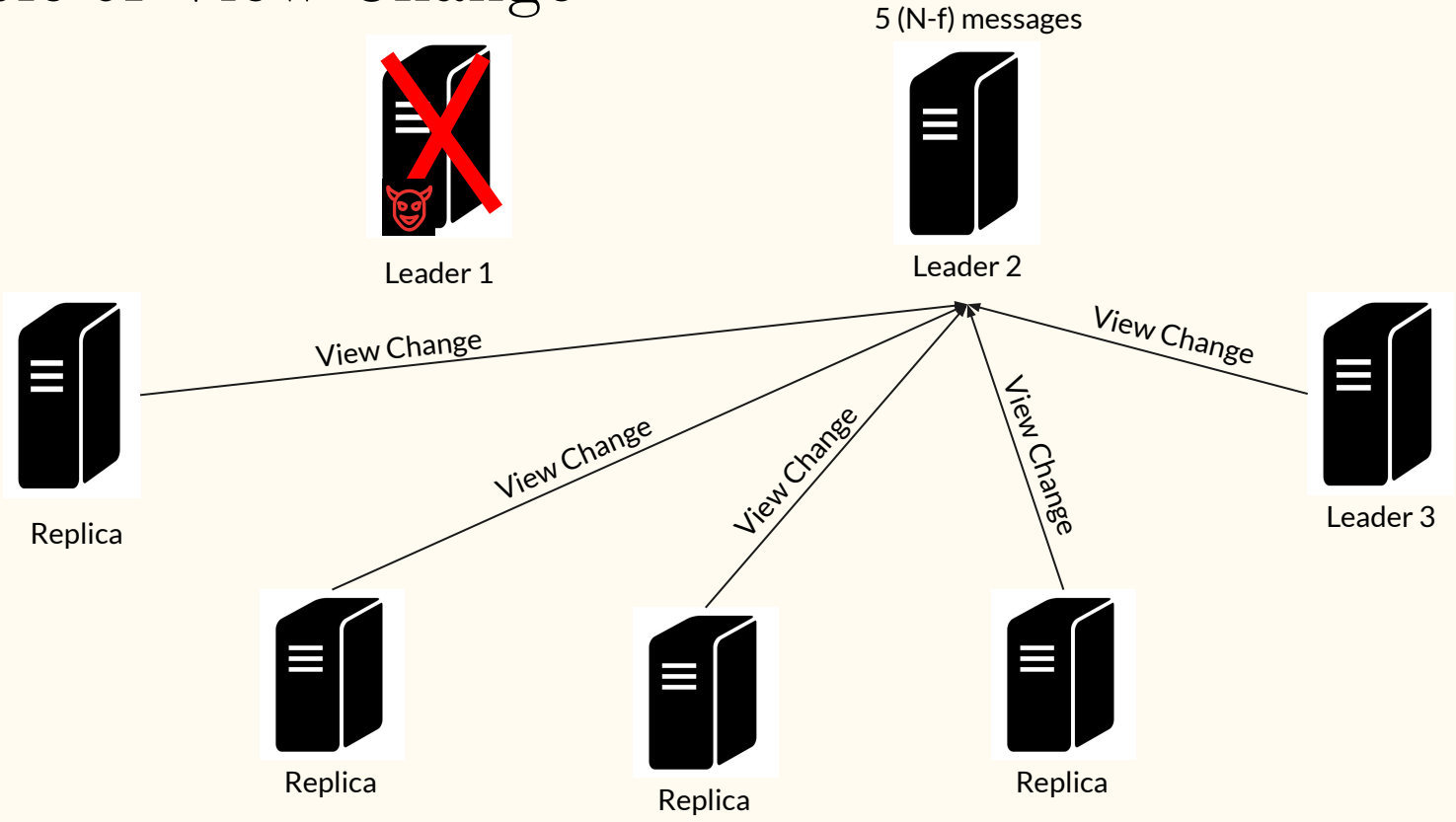
  - (b) They receive $n - f$ view v messages (from different replicas). When this occurs, the leader combines the $n - f$ view v messages into a message with a single threshold signature, which we call a VC for view v. The leader considers the highest QC it has seen, which corresponds to a block B' (say). The leader then broadcasts a new proposed block B with parent B', together with the newly produced VC, and attaches to that message its present message state.

The instructions for view changes

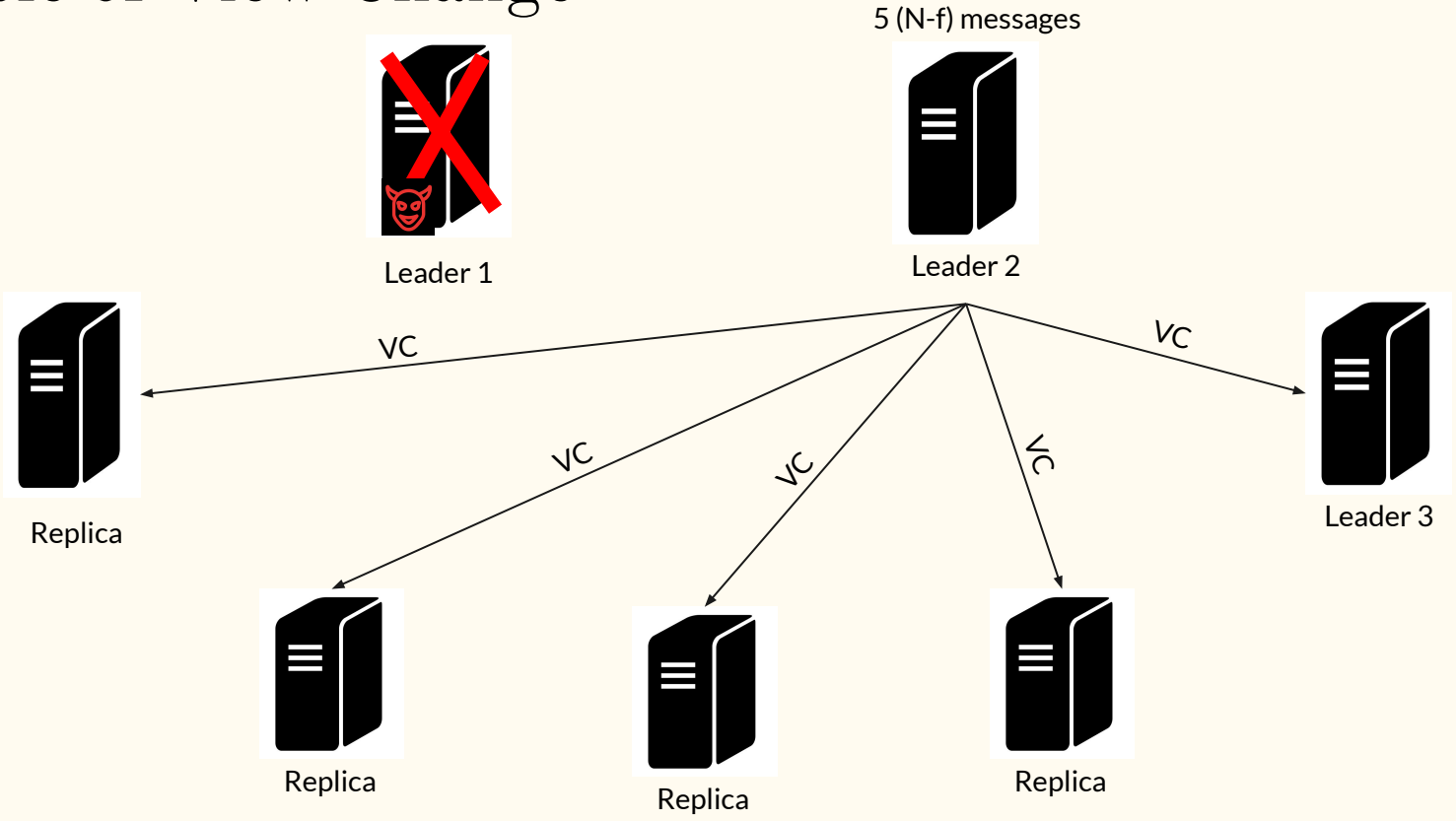
- Each replica which is not the leader of view v of epoch e enters the view if they are in epoch e and both:
  - (a) $v = 0$ or they are presently in a lower view

    And
  - (b) they see a VC for view v.

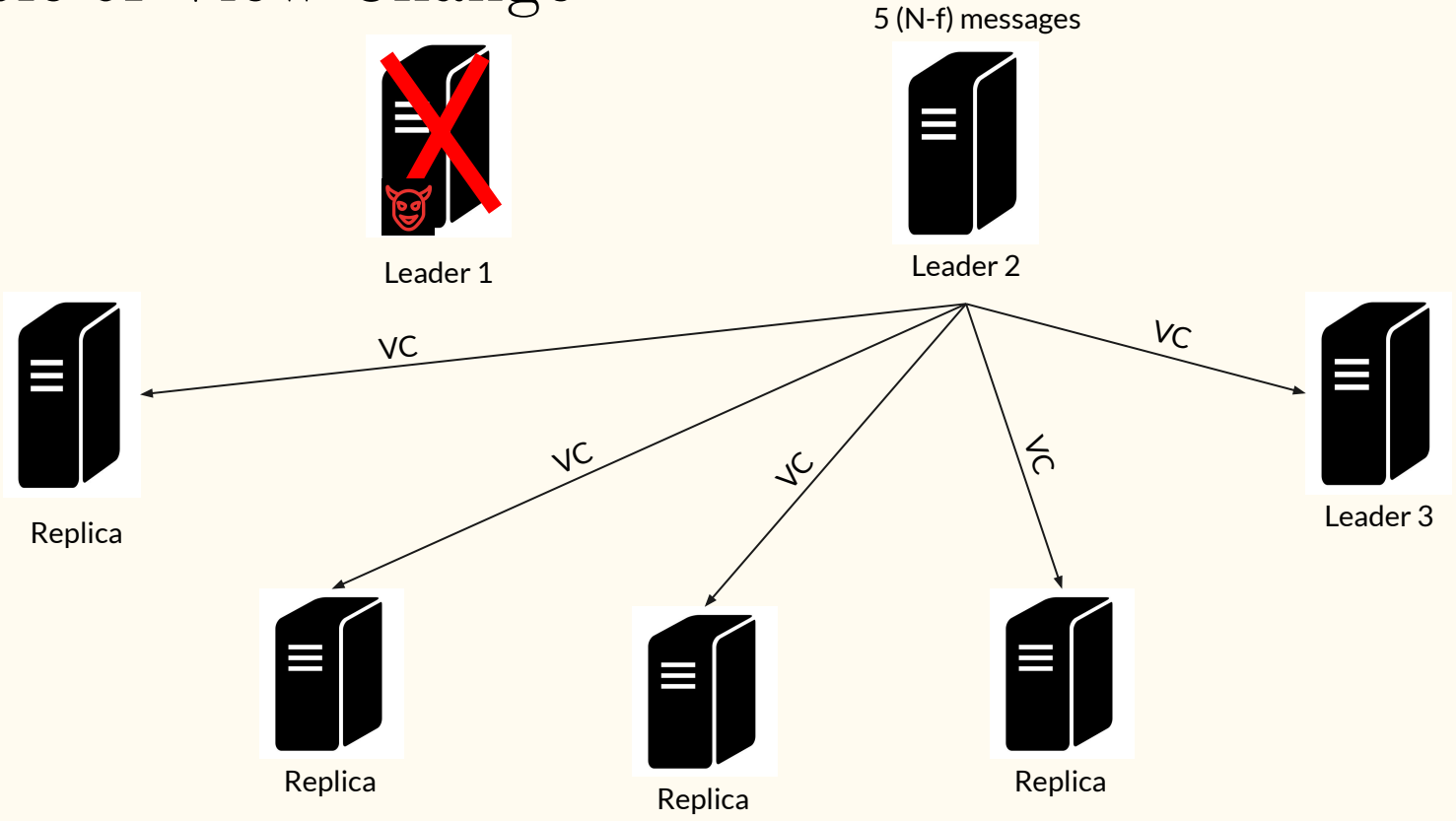
Example of View Change



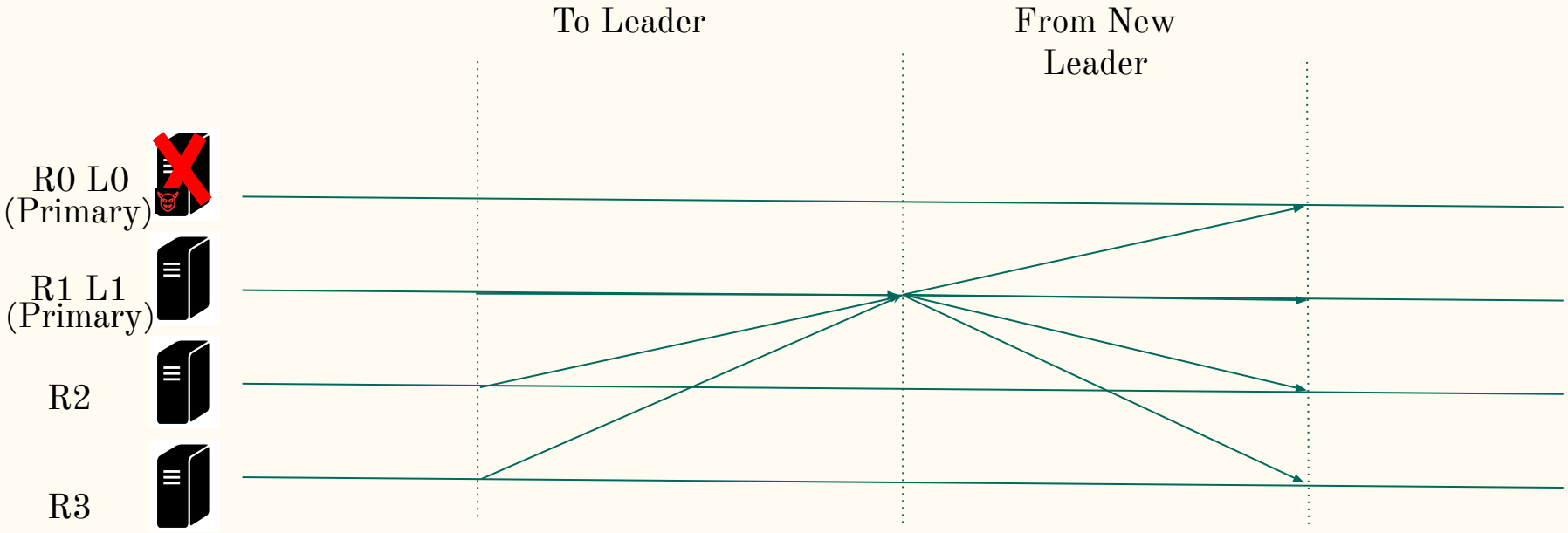
Example of View Change



Example of View Change



Message Diagram



Complexity O(n) + O(n) = O(n)

Worst case complexity for view change

- There are a total of *f+1* views an epoch
- A maximum of *f* faults can be tolerated.
- If all of the elected leaders would be faulty, we would need to loop through all of them.
- That means the worst case complexity = message complexity x number of leaders
- Thus, the worst case message complexity would is $O(n * (f+1)) = O(n^2)$

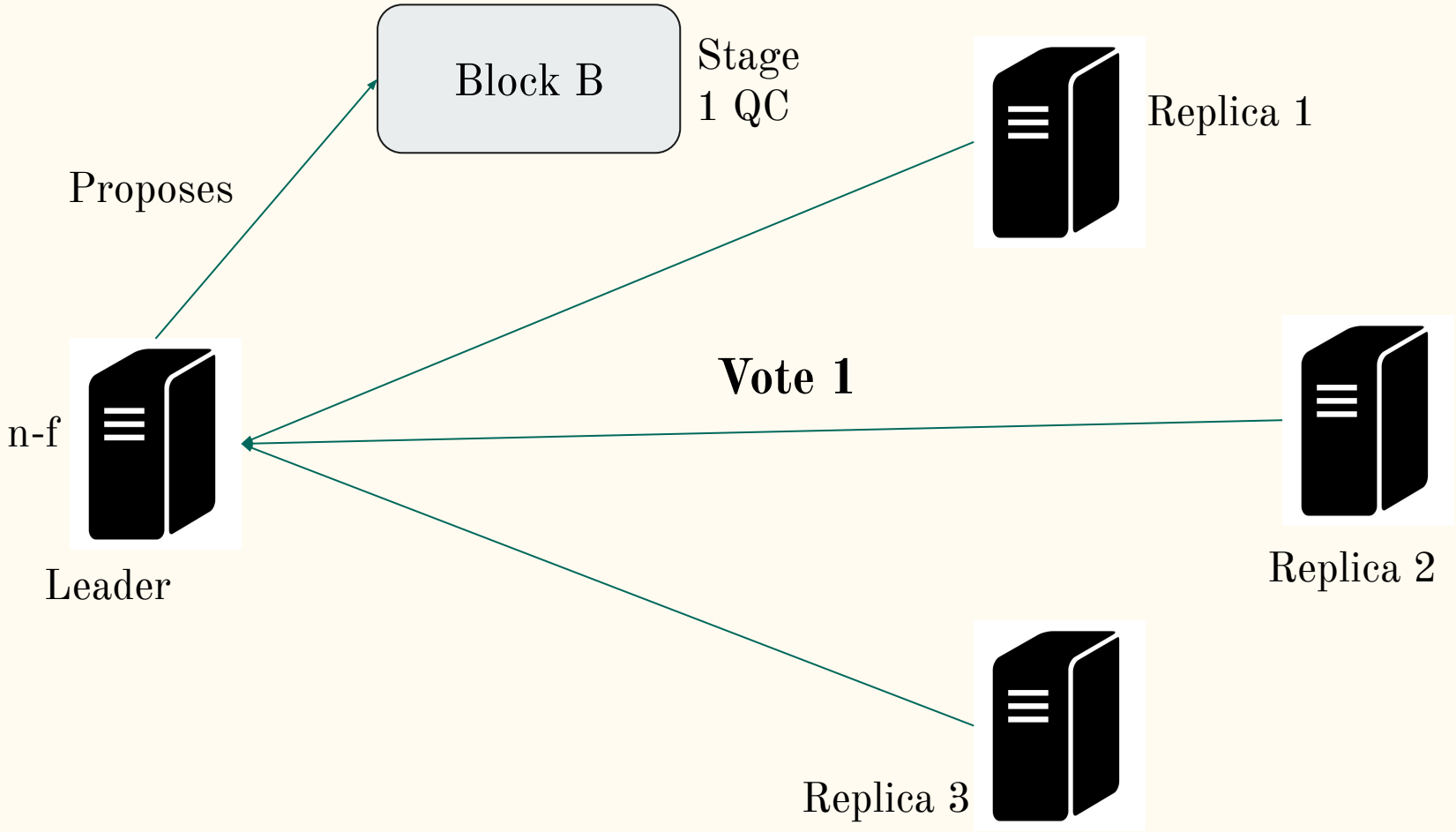
Worst case complexity for view change in PBFT

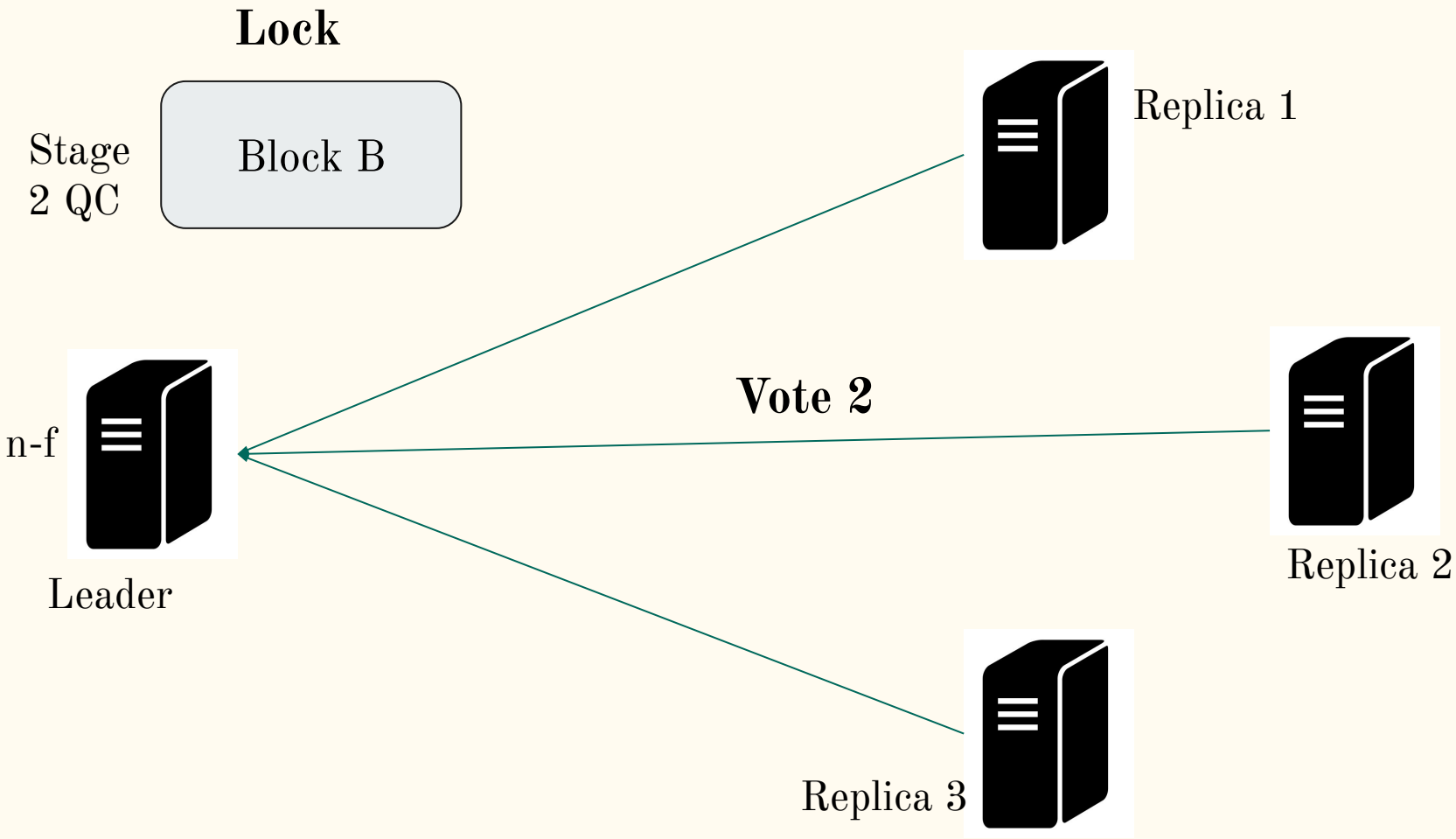
- While PBFT does not explicitly talk about epoch, we can use similar concept.
- That means the worst case complexity = message complexity x number of leaders
- Thus, the worst case message complexity would is $O(n^2 * (f+1)) = O(n^3)$

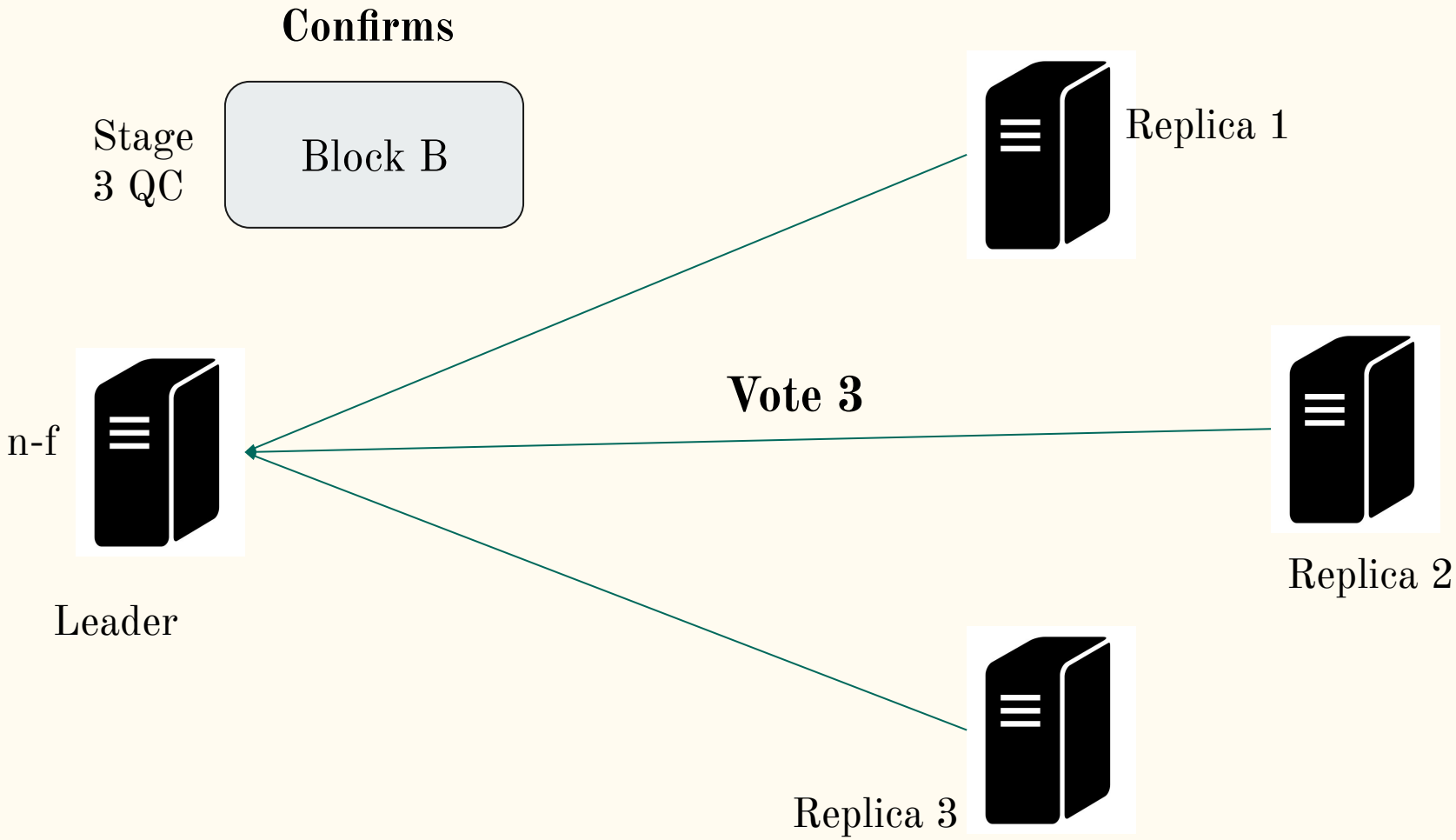
Comparison of message complexities

	PBFT	Quadratic Worst Case
One View Change	$O(n^2)$	$O(n)$
Worst Case	$O(n^3)$	$O(n^2)$

View Instructions

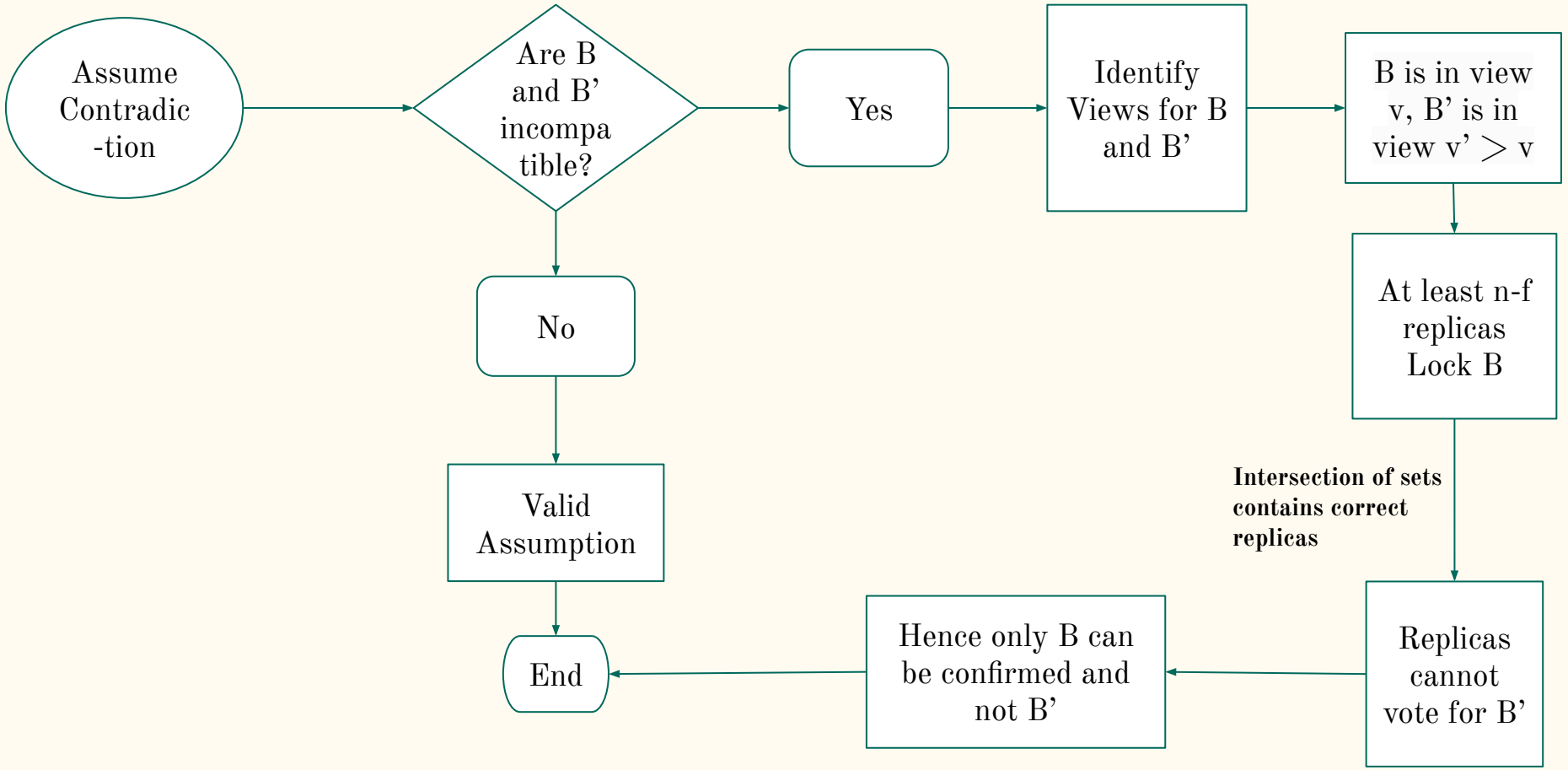

Block B
Stage
1 QC
Proposes
Replica 1
n-f
Vote 1
Replica 2
Leader
Replica 3


Lock
Stage
2 QC
Block B
Replica 1
n-f
Vote 2
Replica 2
Leader
Replica 3

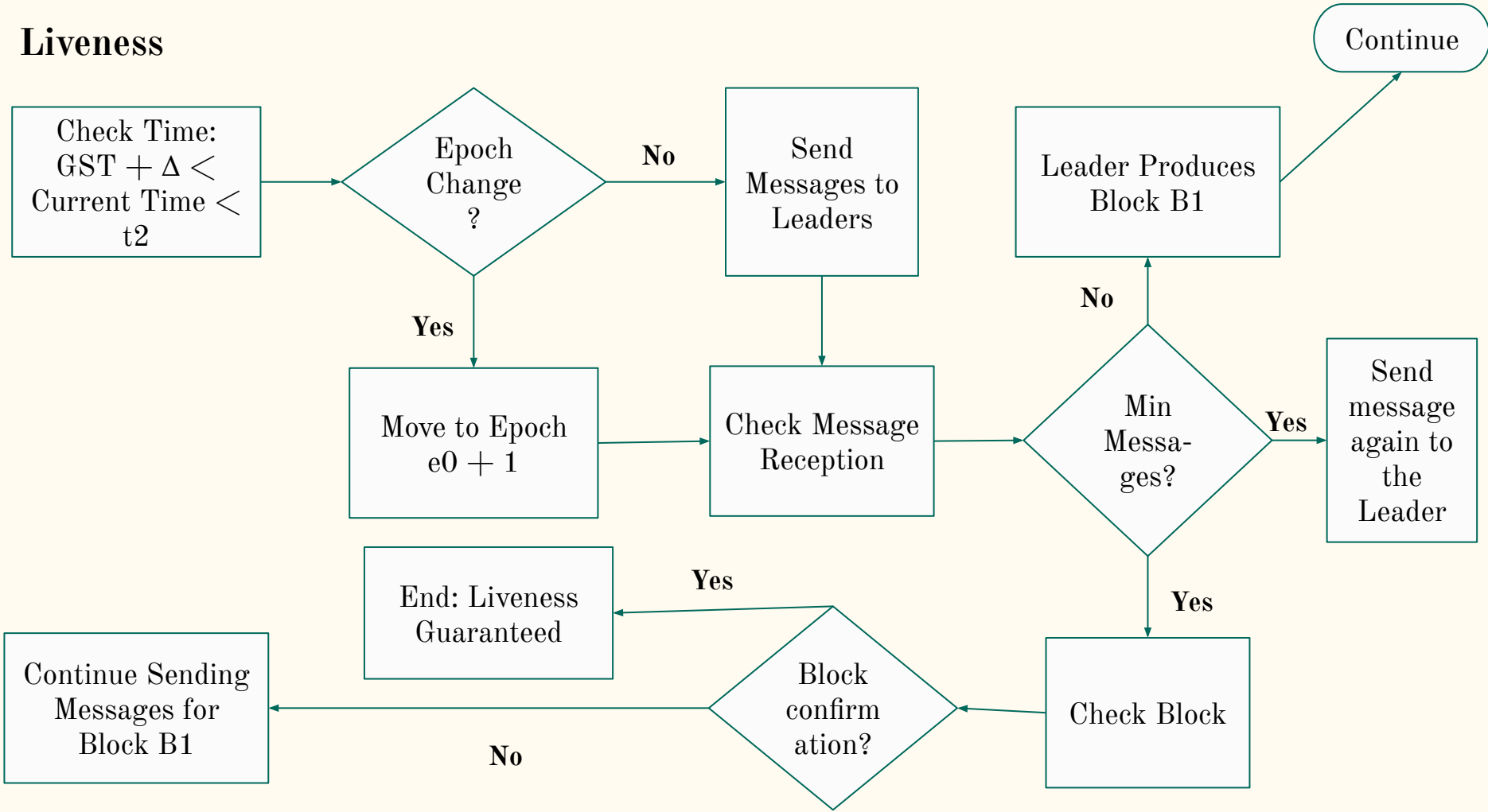

Confirms
Stage
3 QC
Block B
Replica 1
n-f
Vote 3
Replica 2
Leader
Replica 3

Message Complexity

Consistency



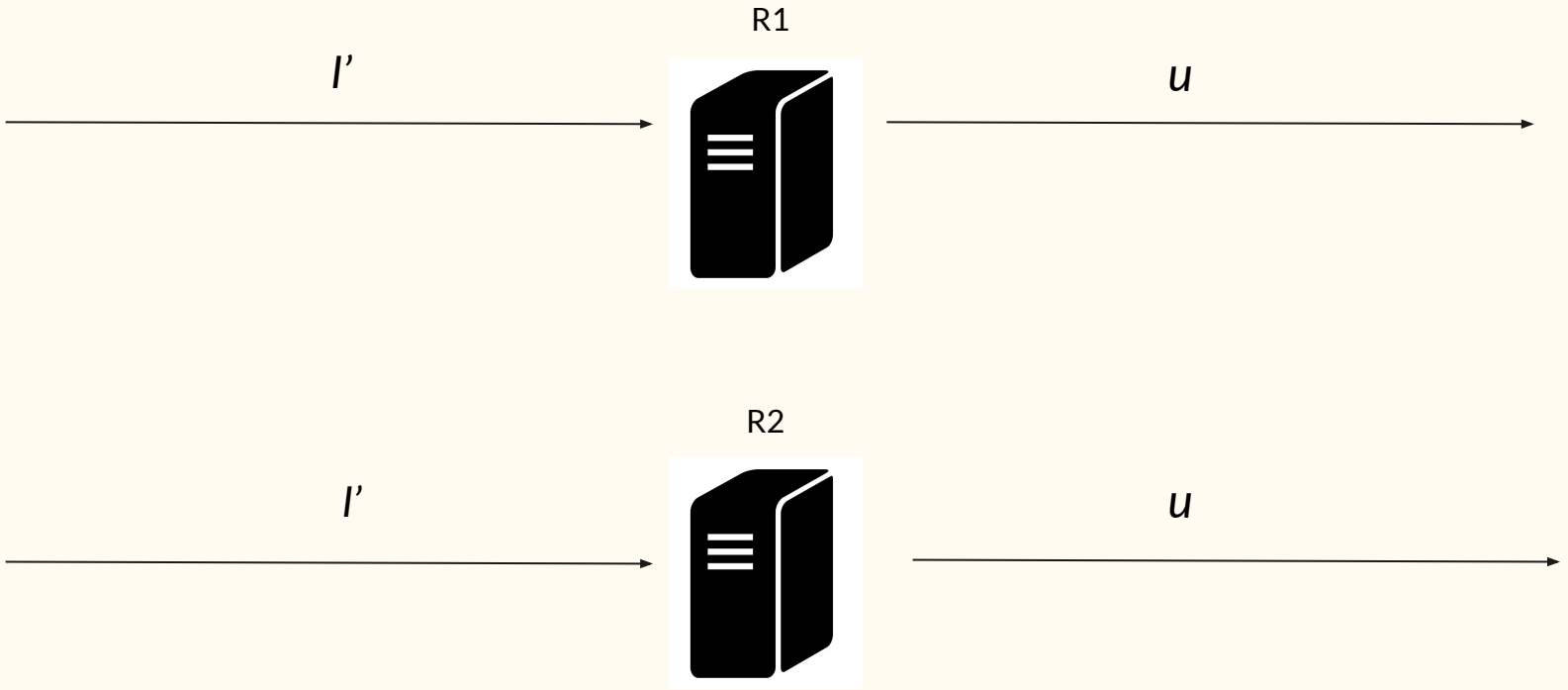
Liveness


Check Time: GST + Δ < Current Time < t2
Epoch Change ?
No
Send Messages to Leaders
Yes
Move to Epoch e0 + 1
Check Message Reception
Min Messa-ges?
Yes
Send message again to the Leader
No
Leader Produces Block B1
Continue
Yes
Check Block
Block confirm ation?
Yes
End: Liveness Guaranteed
No
Continue Sending Messages for Block B1

Communication Complexity

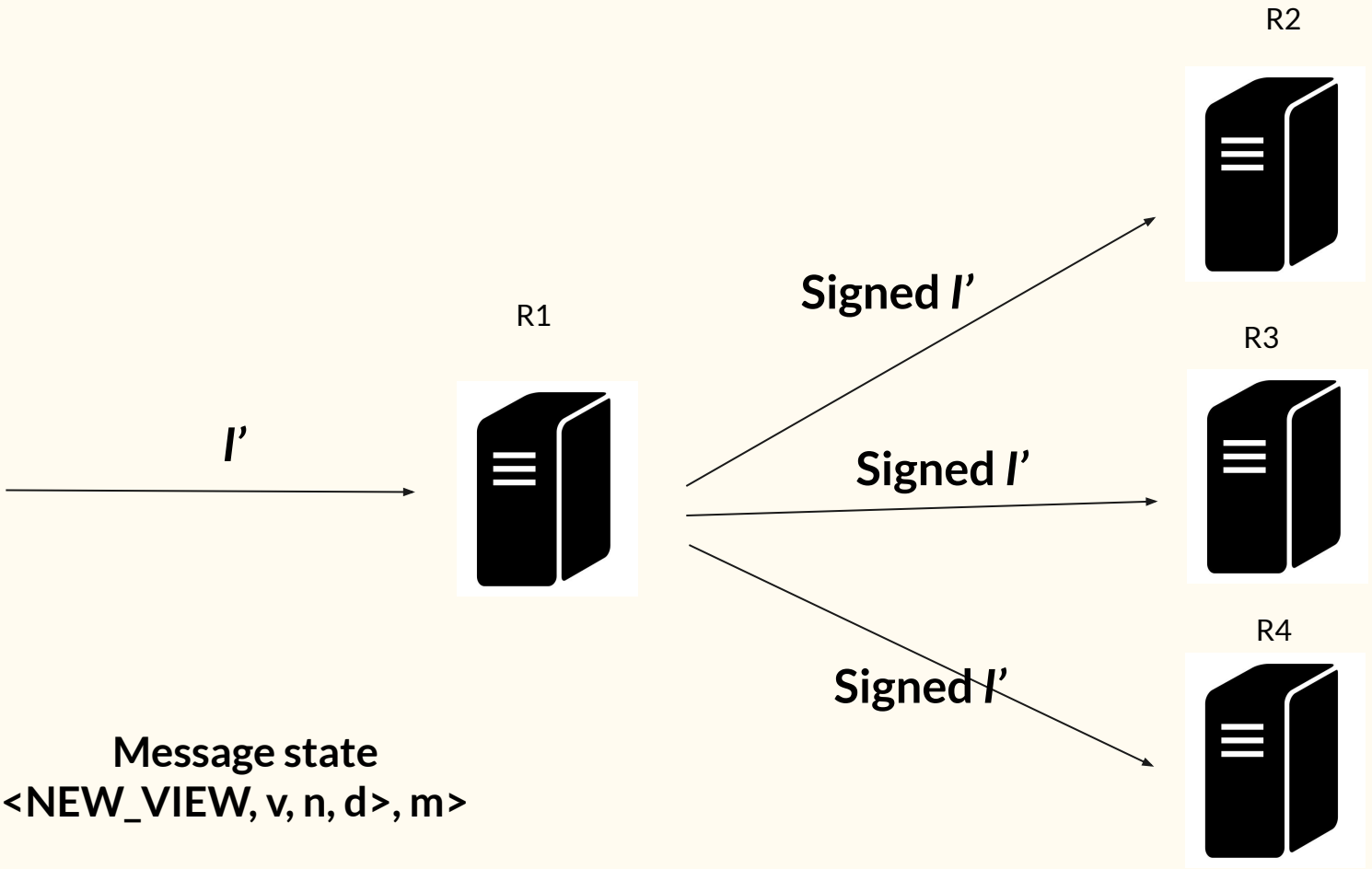
Why?

For all correct replica same input should result same output

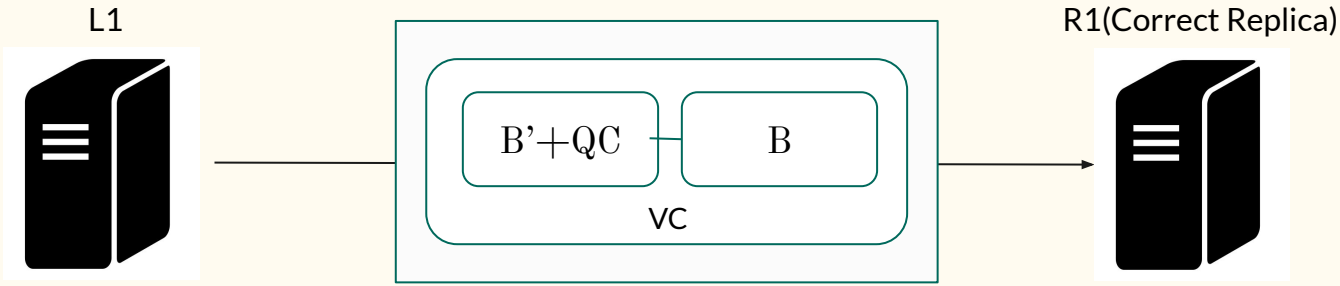


1. Replicas want sufficient information **to build a blockchain.**

2. Replicas want sufficient information **to execute the requests.**

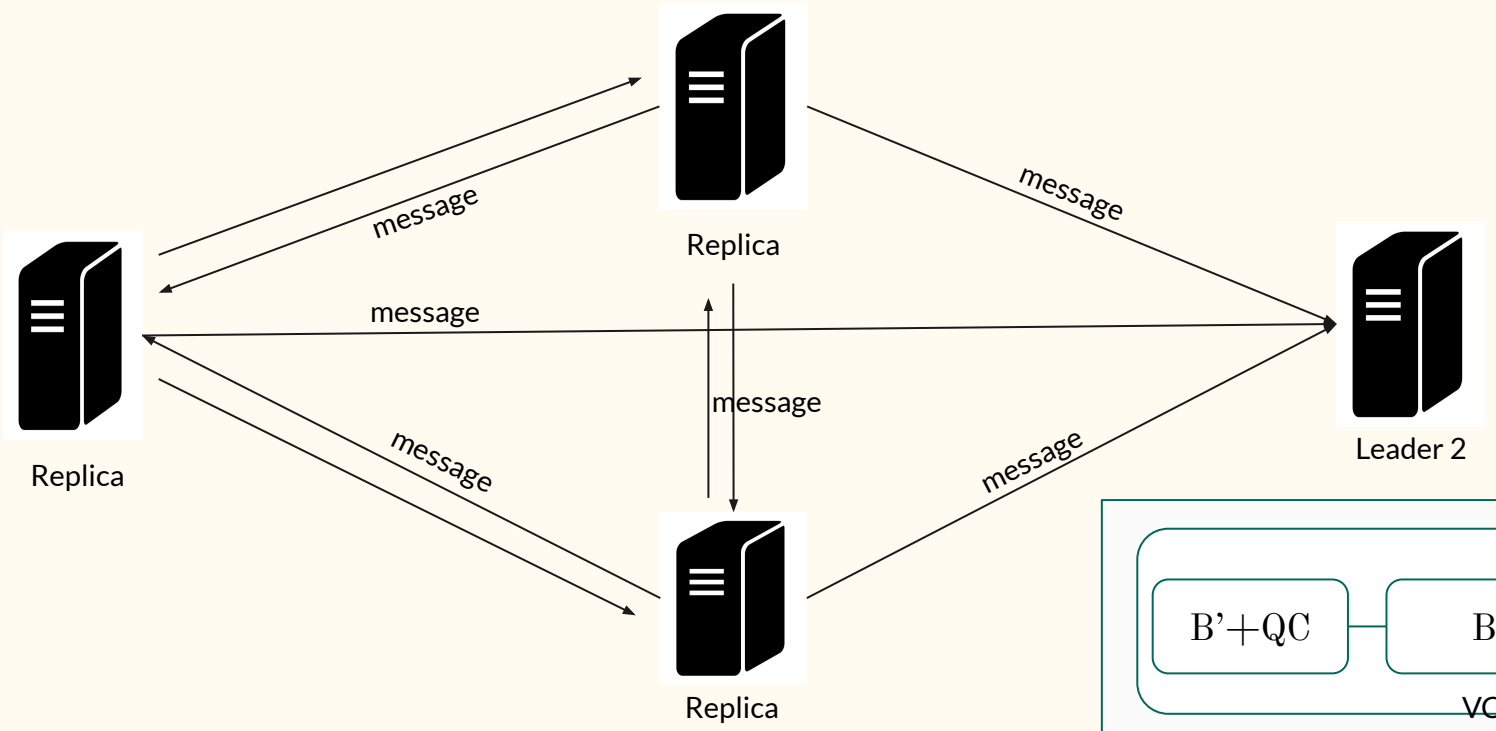

R2
R1
R3
R4
I'
Signed I'
Signed I'
Signed I'
Message state
<<NEW_VIEW, v, n, d>, m>

Correct Replica verifies the confirmed block



1. h(B) == h(B')
2. dv(B) == dv(B')

Communication Complexity :O(k*n^2)



1. The number of bits sent by correct replicas (combined) in the time interval (GST + Δ, t1] is (n^2).
2. A block corresponding to epoch 0 will be confirmed and seen by a correct replica by time t1.

GST + **Δ = time taken send the message in the system**

Δ = time taken to receive the message by the replica

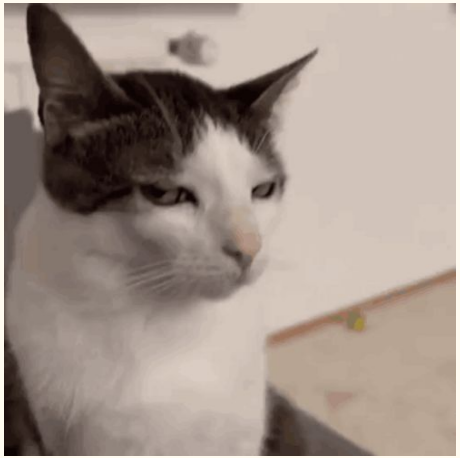
12Δ= trigger time for a view to complete

f+1 = no of views in an epoch

$$t1 \leqslant GST + \Delta + \Delta + 12(f+1)\Delta$$

Future Scope

Can we achieve mean communication complexity as O(kn)?



Can we only validate blockchain by just sending certificates of the current request?