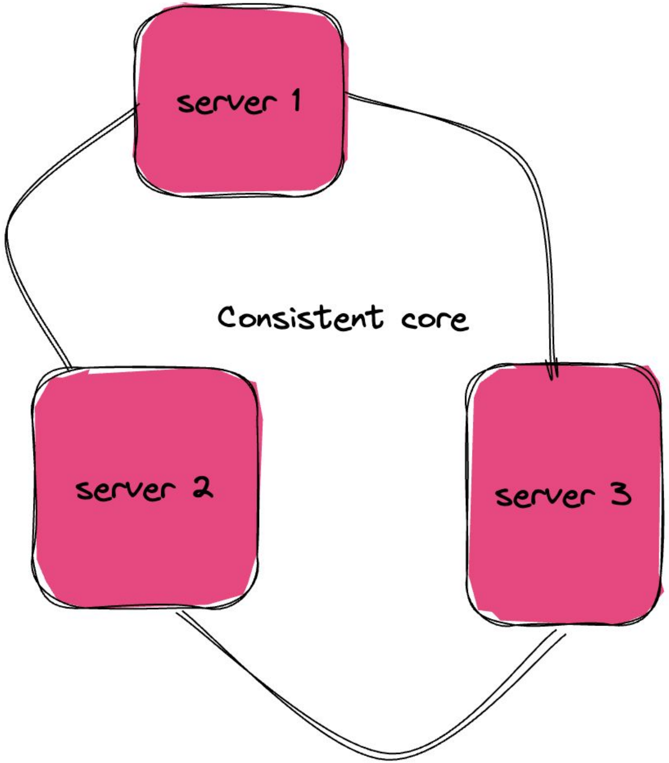
RAFT - Distributed consensus protocol

Distributed = Many

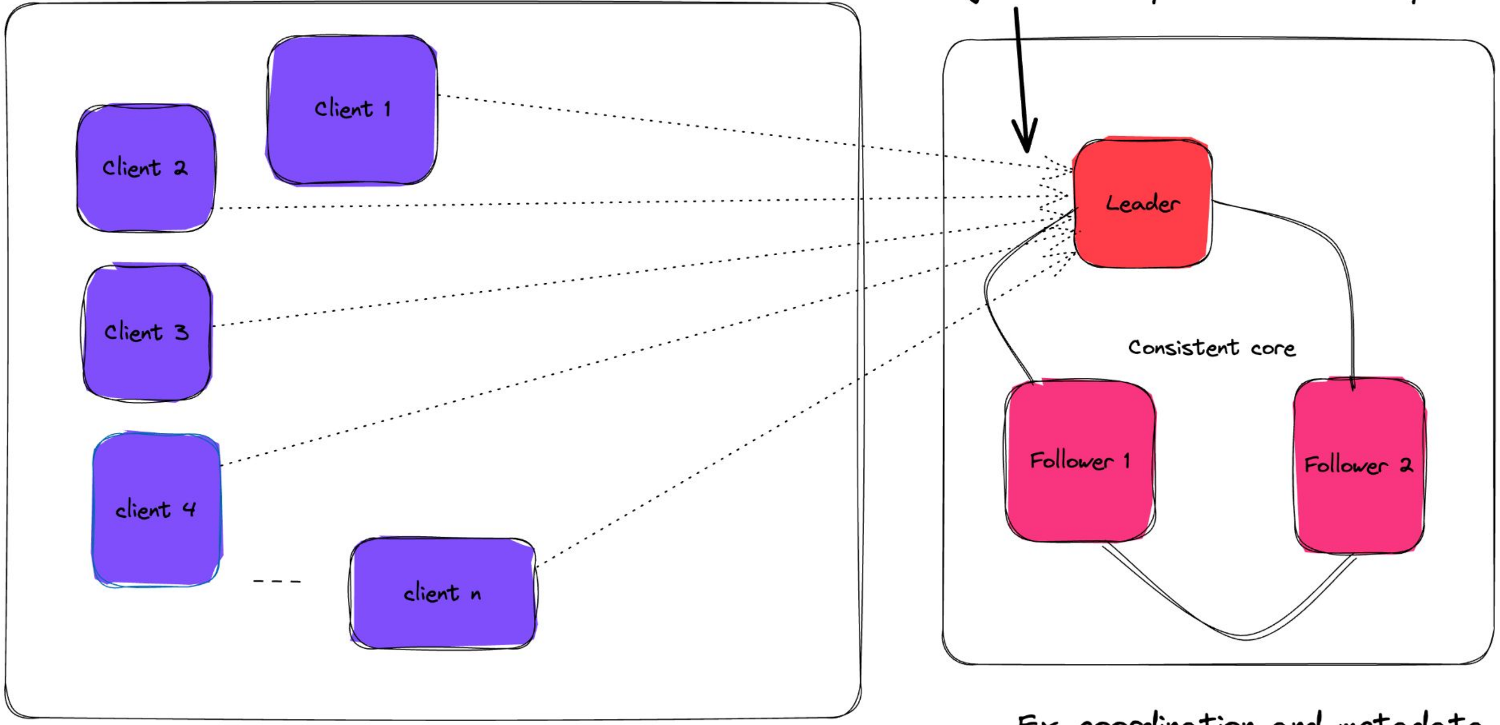
Consensus = Agreement

Consistent core



Linearizability

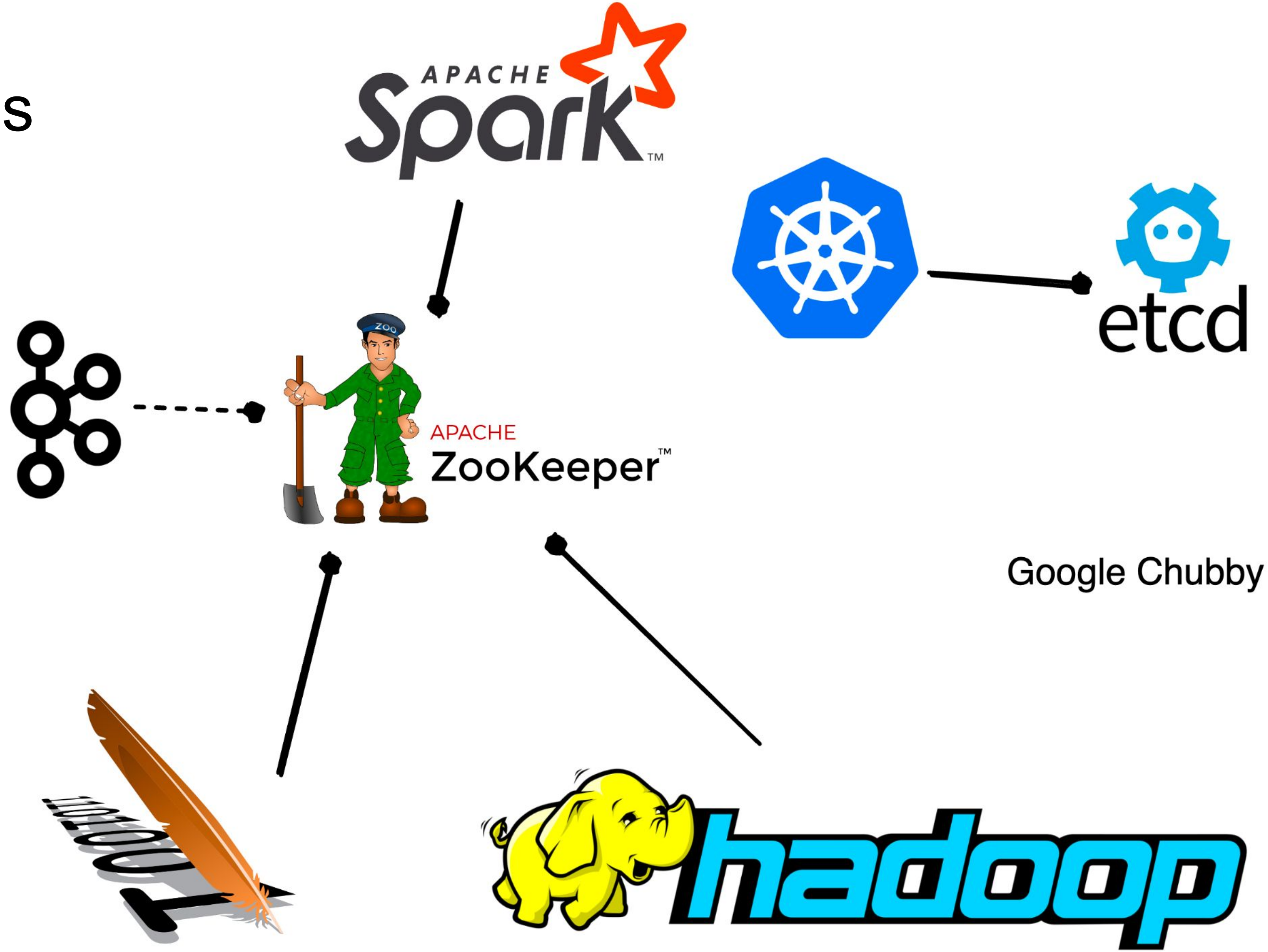
Strong consistency in distributed system



Raft
Paxos
Zab

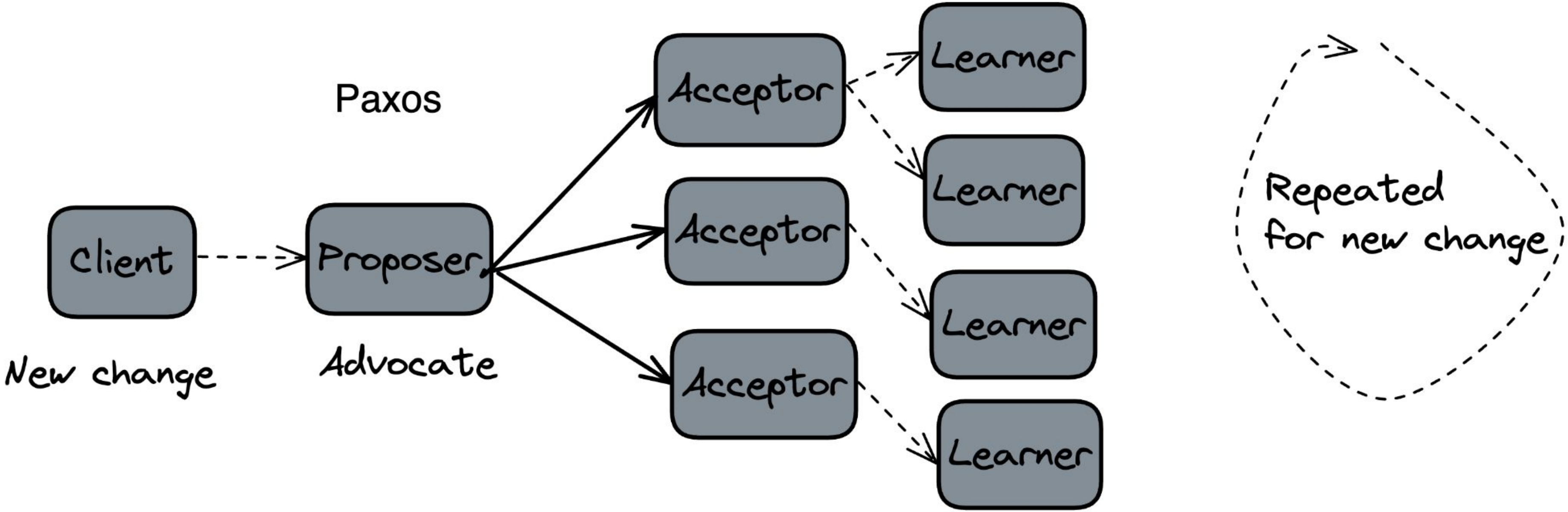
Ex. coordination and metadata storage

Examples



Google Chubby

Paxos

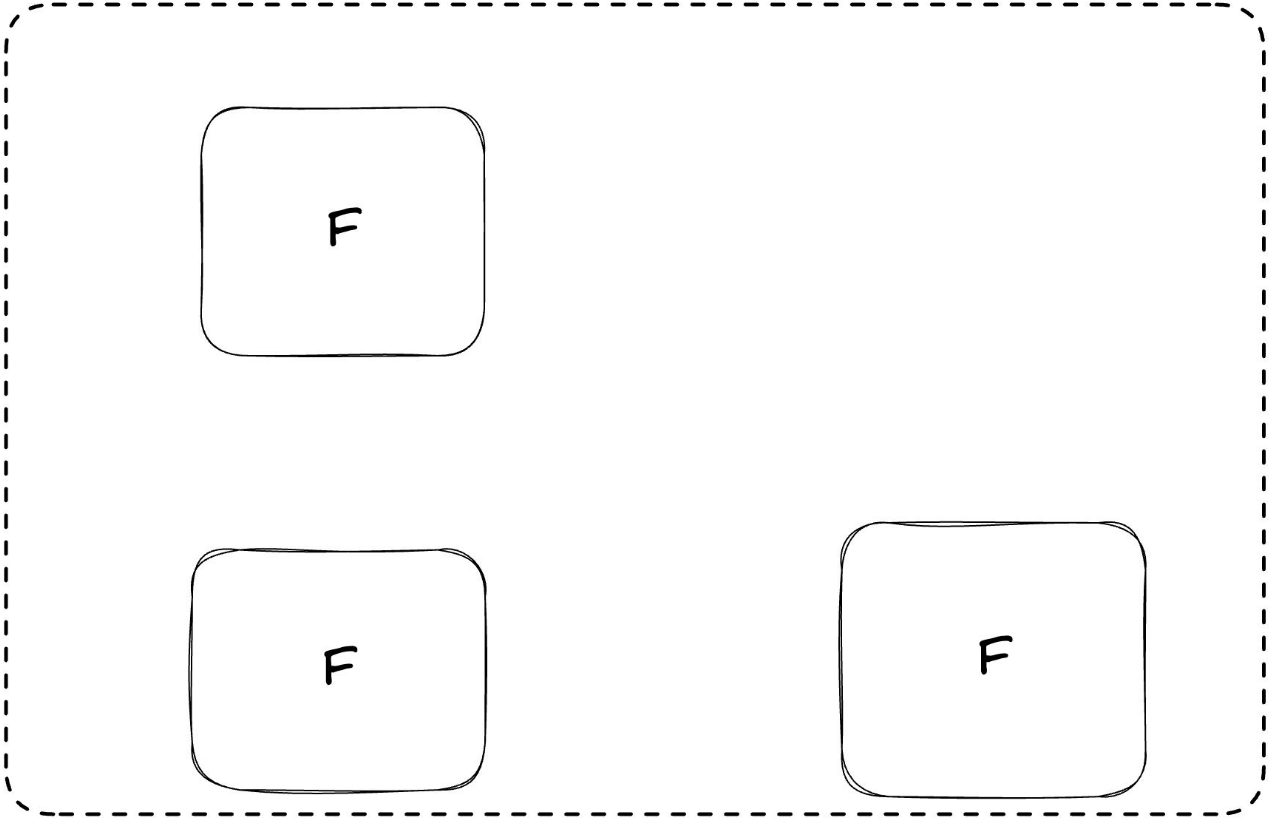

Paxos
Client
New change
Proposer
Advocate
Acceptor
Acceptor
Acceptor
Learner
Learner
Learner
Learner
Repeated for new change

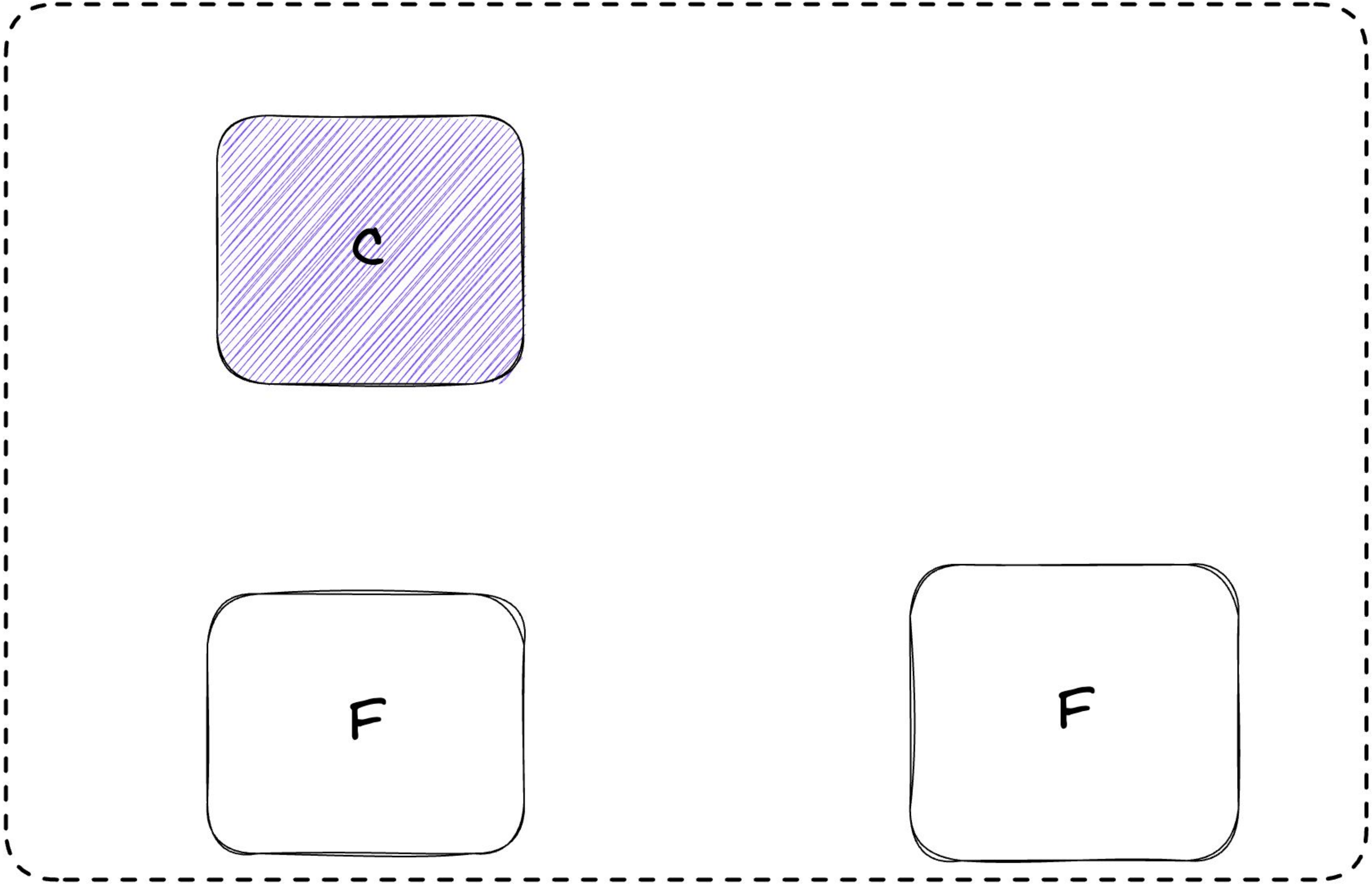
Roles in Raft

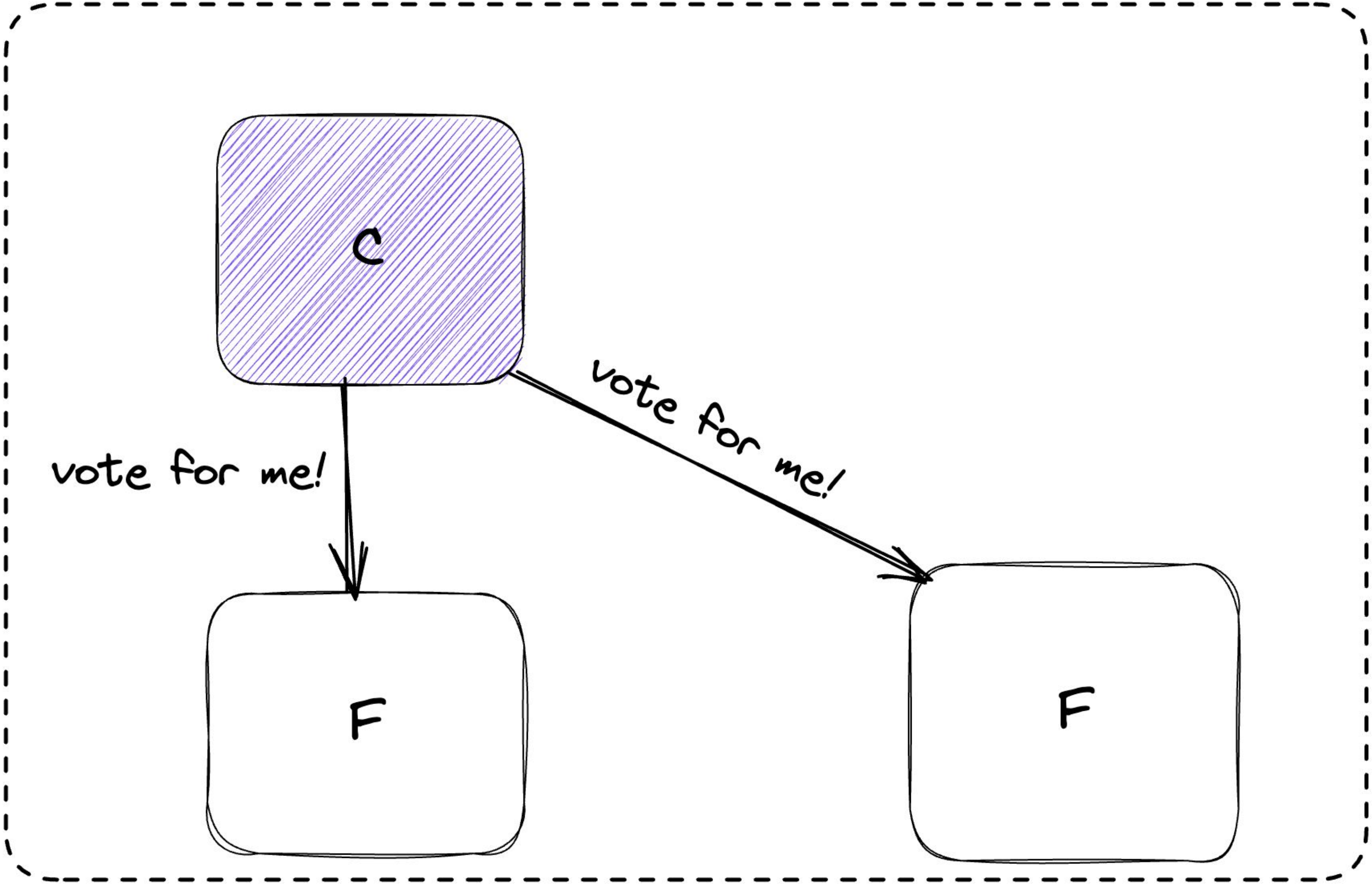
It has 3 roles

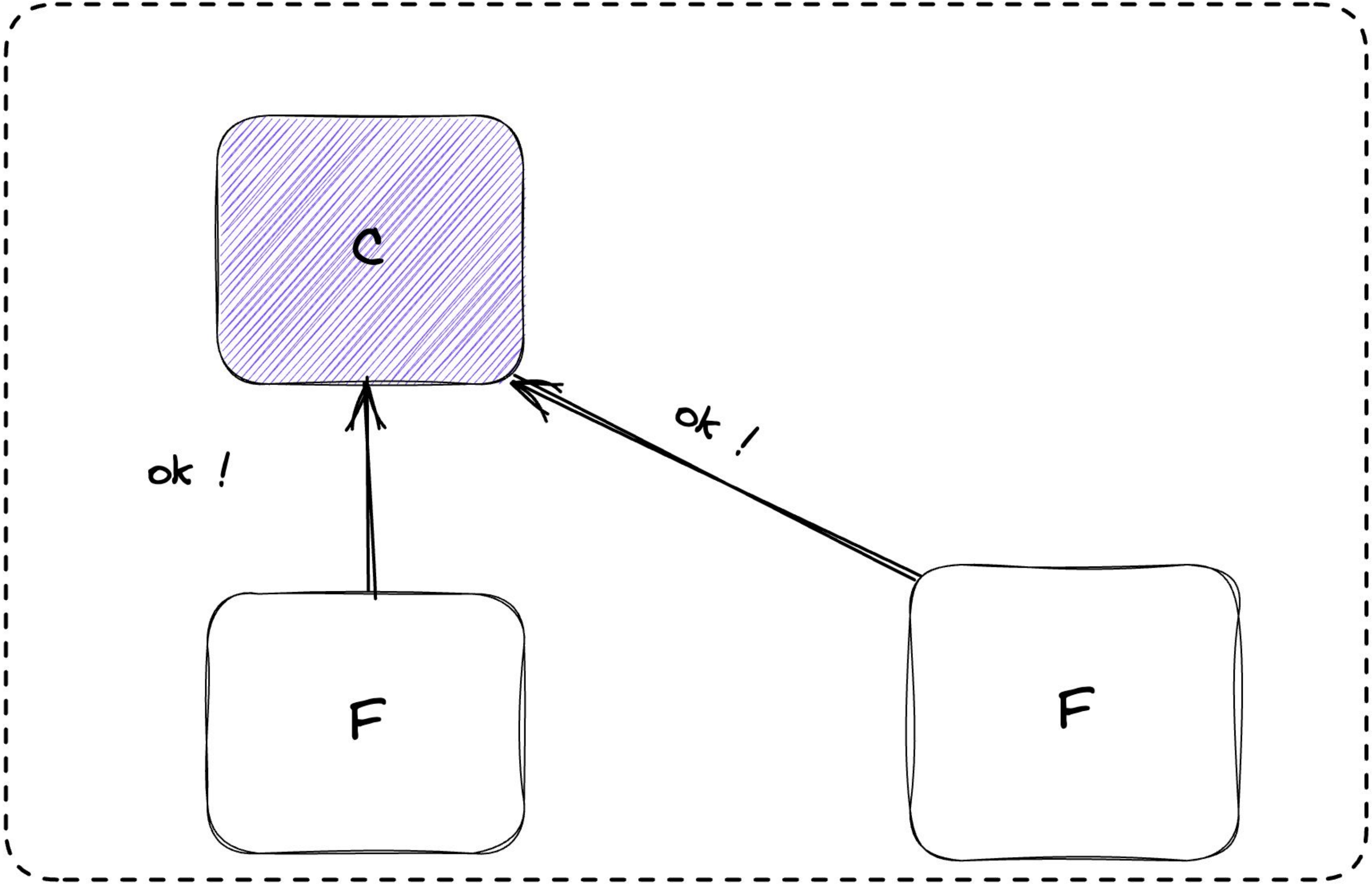
- Leader
- Follower
- Candidate

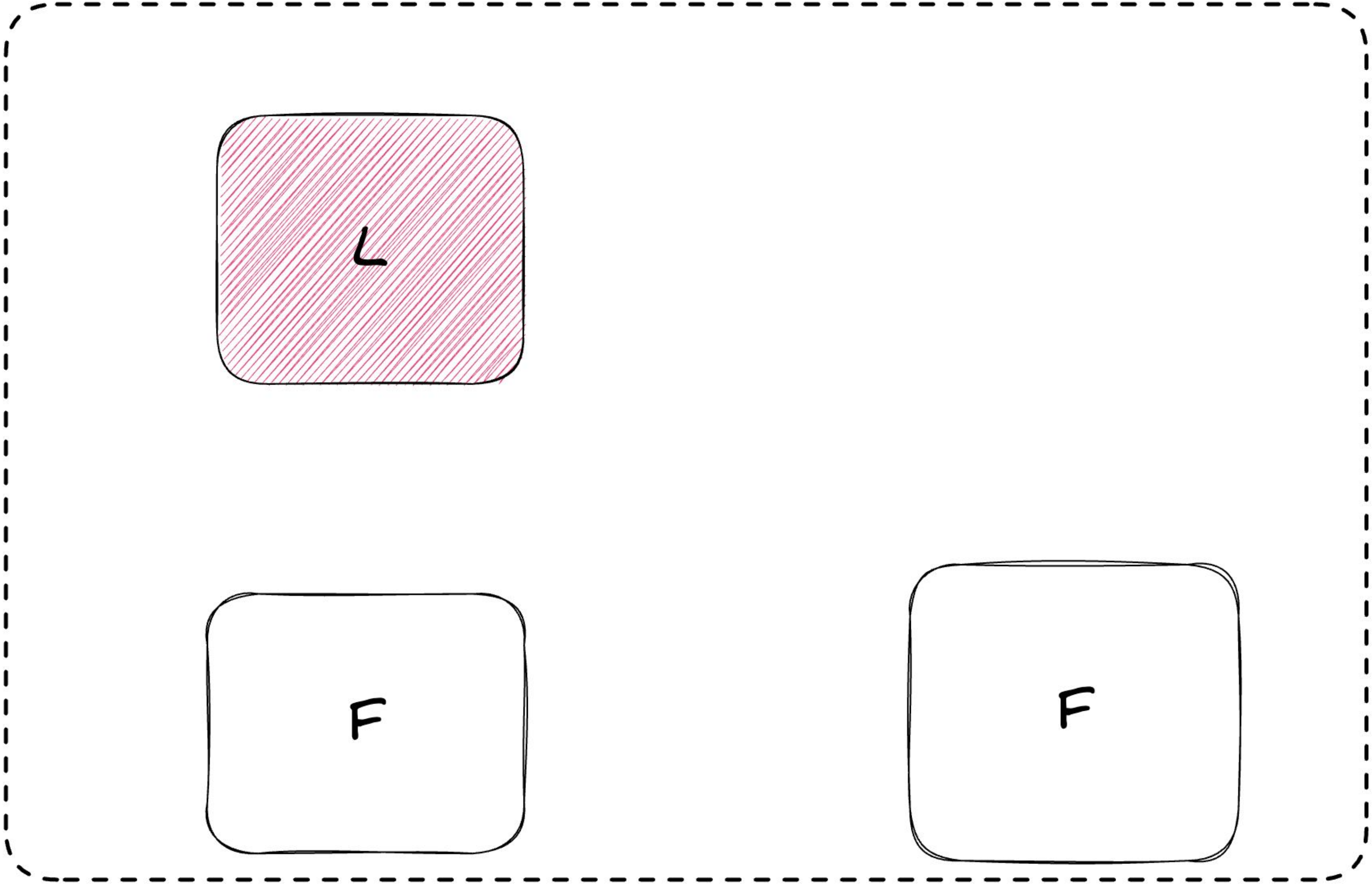
Example

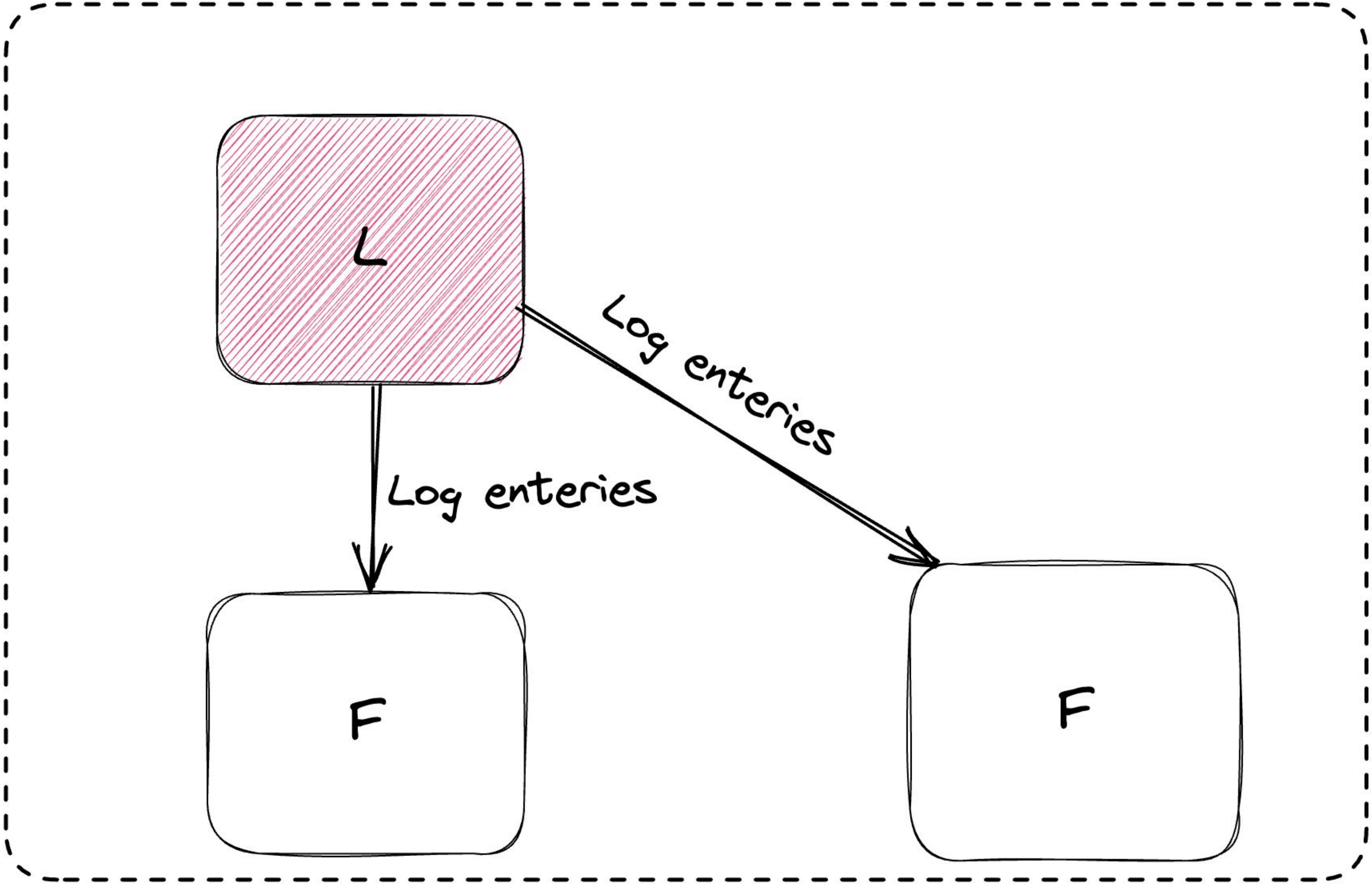


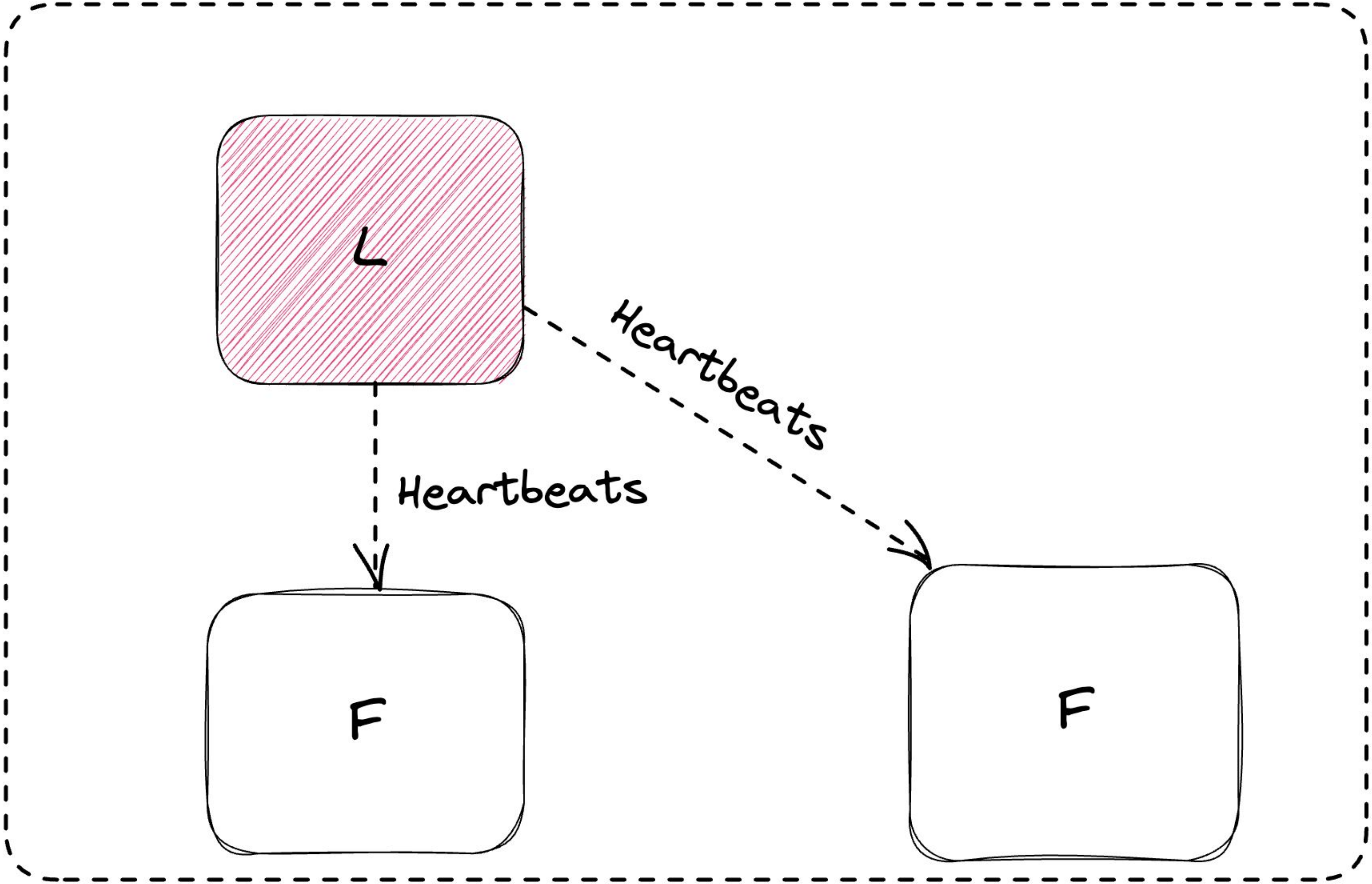

C
F
F

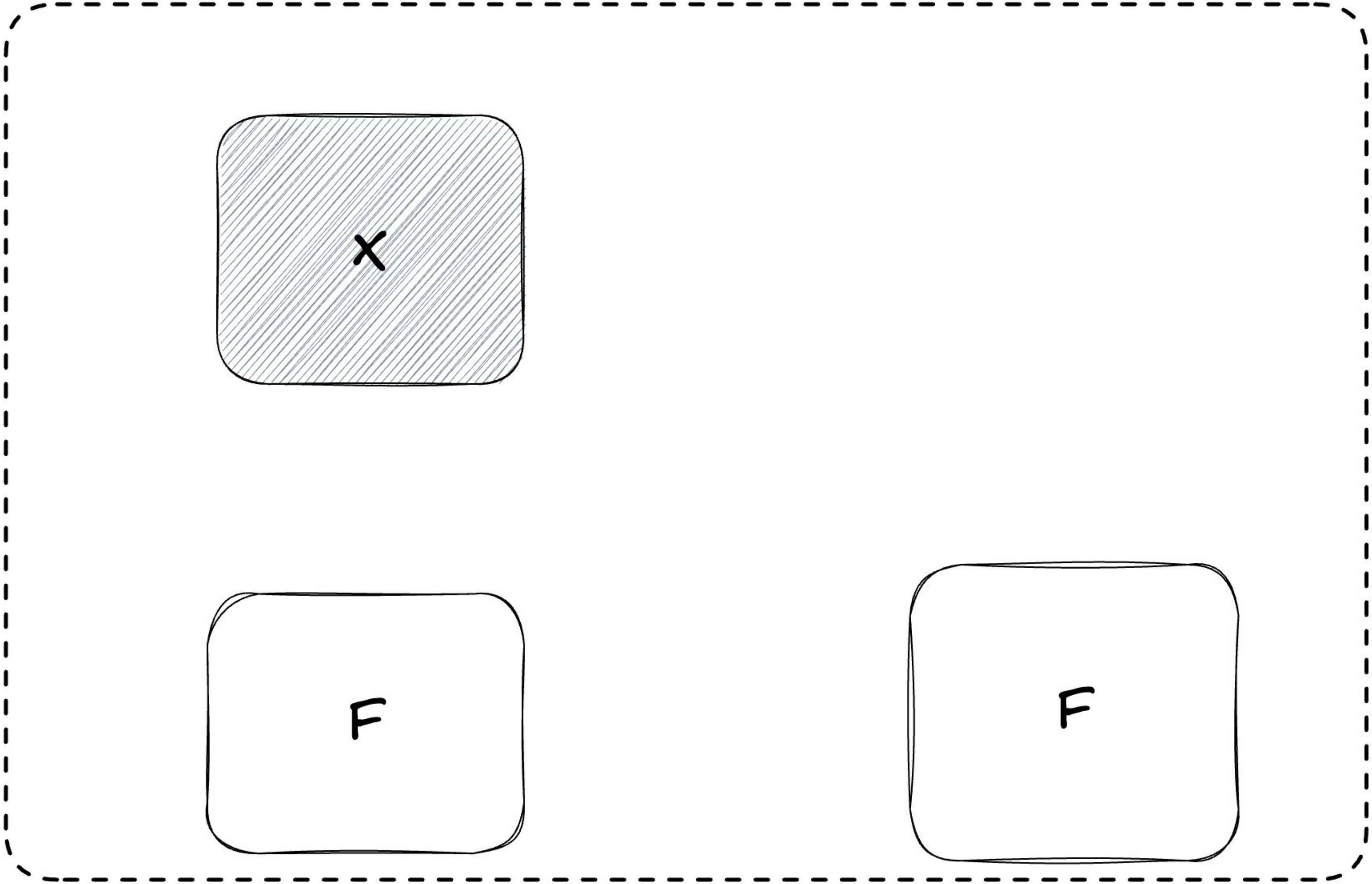

C
vote for me!
vote for me!
F
F

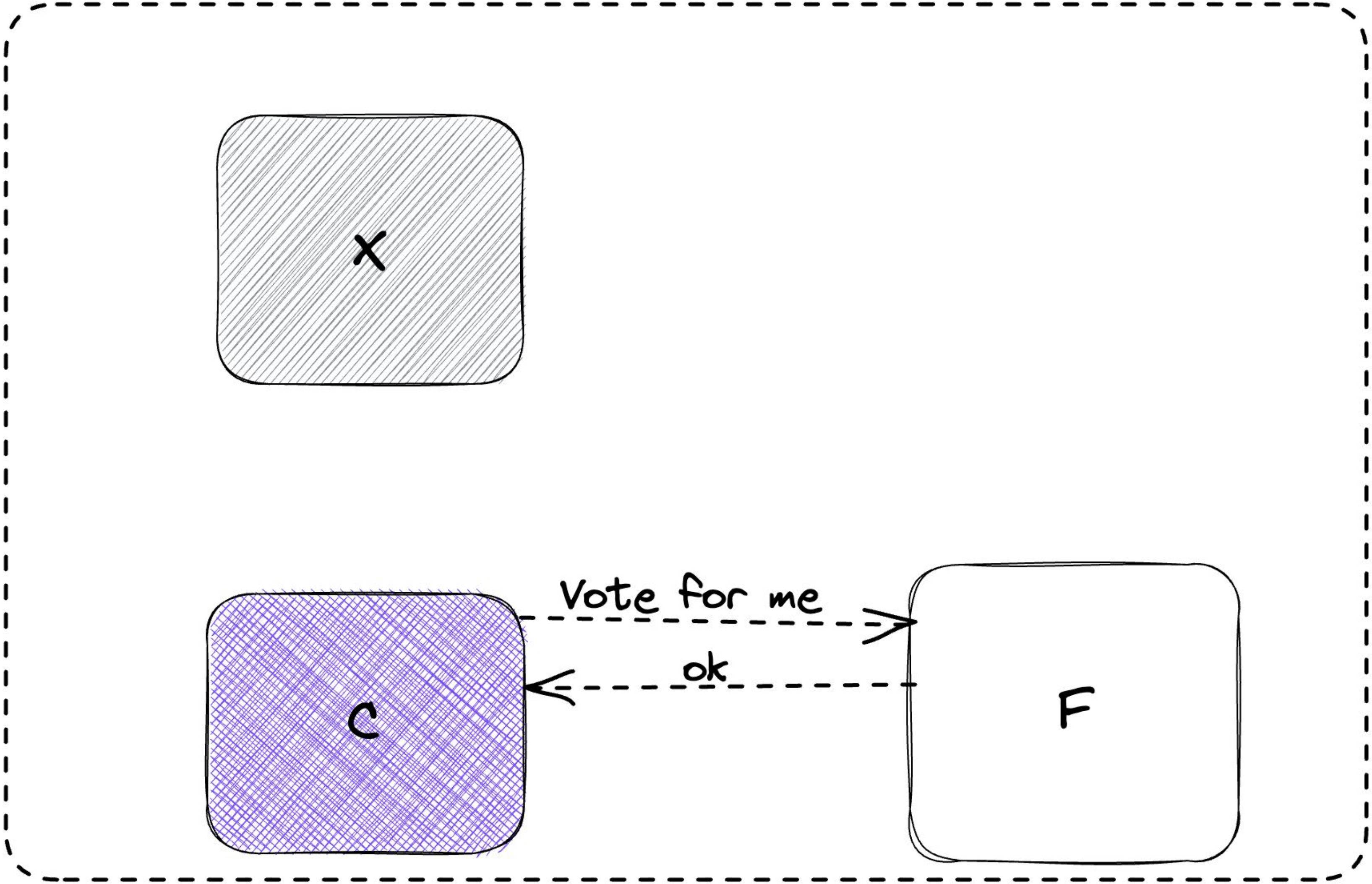

C
ok !
ok !
F
F

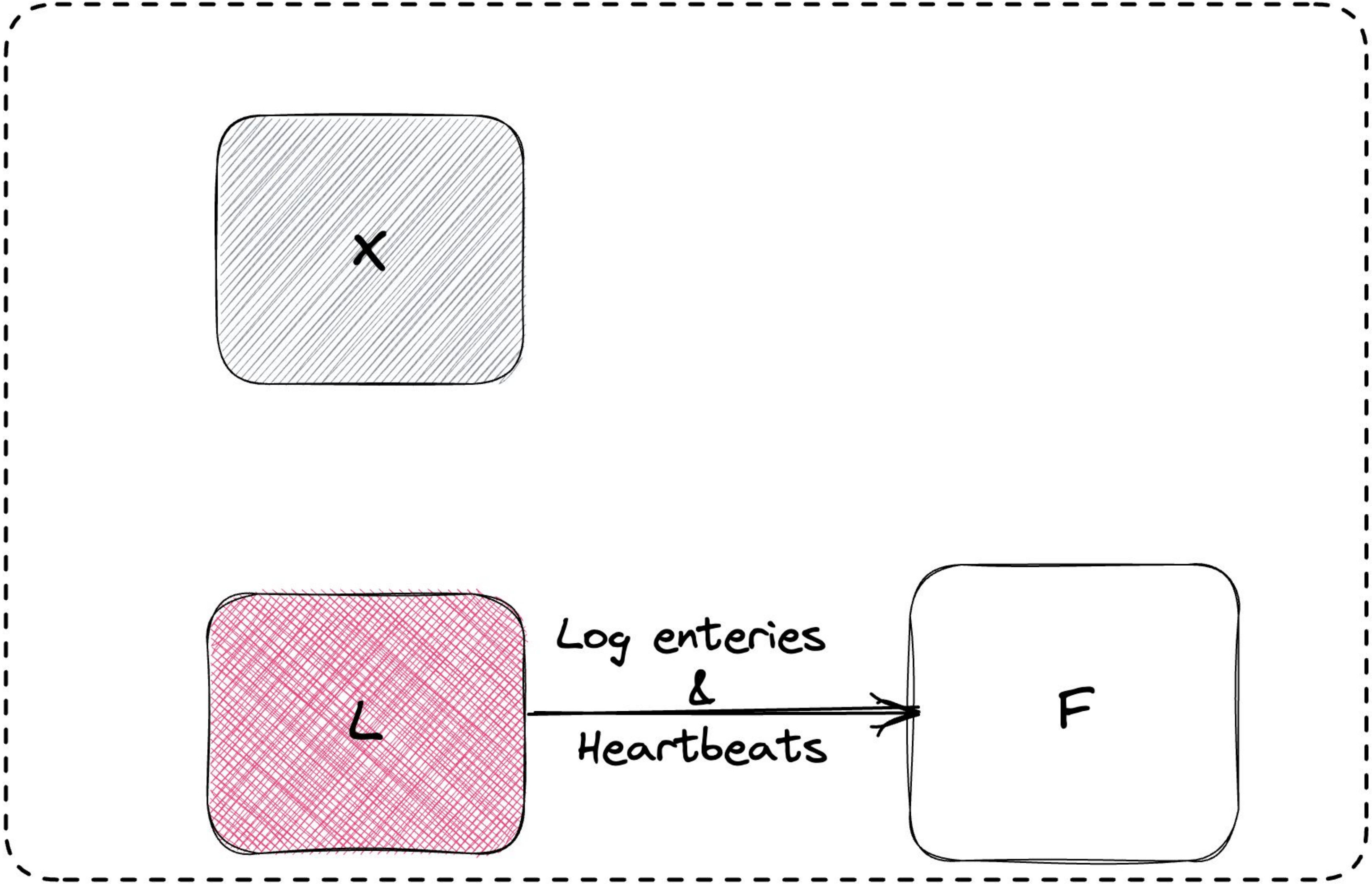

L
F
F

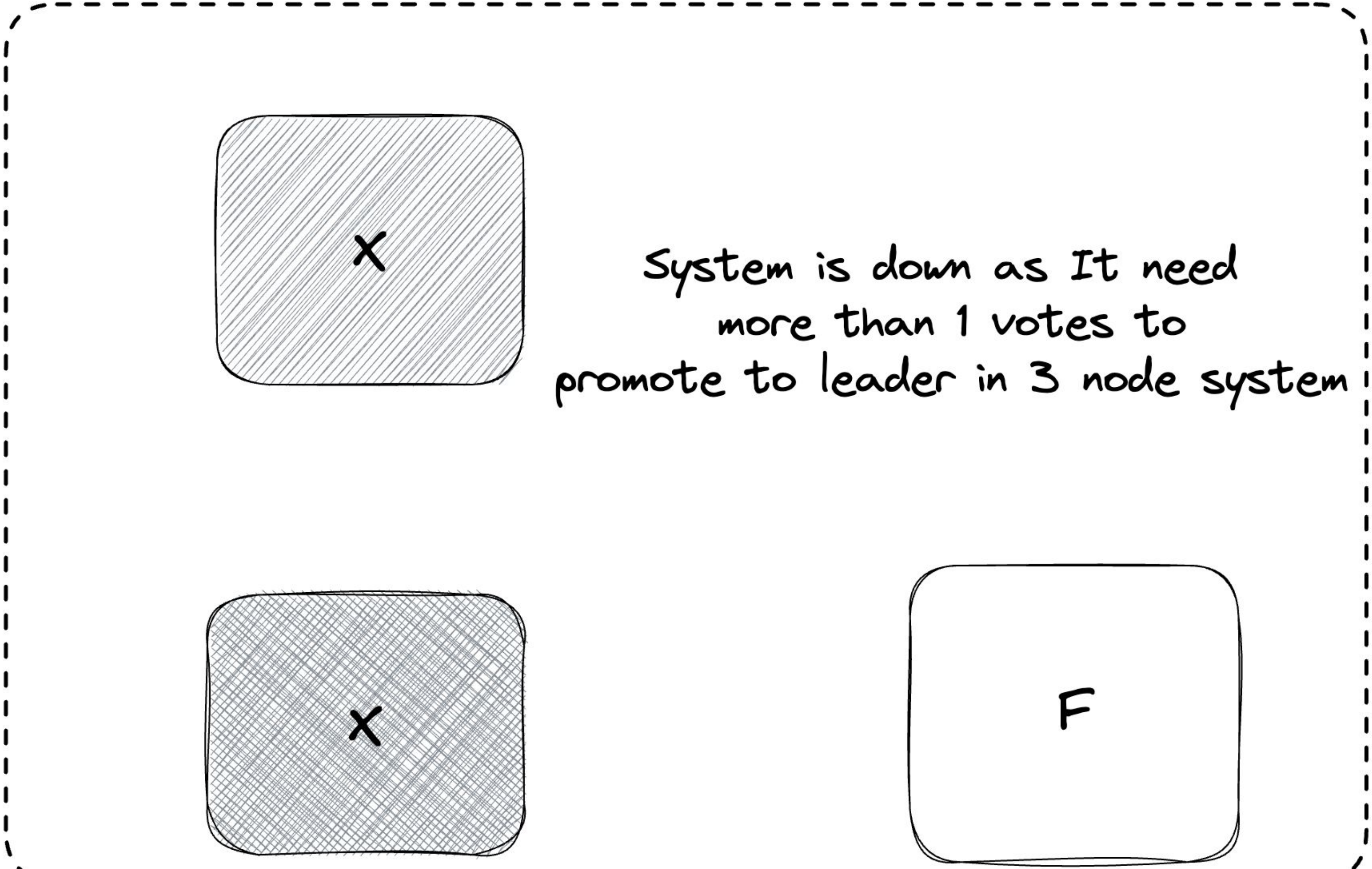

L
Log enteries
Log enteries
F
F


L
Heartbeats
Heartbeats
F
F


X
F
F


X
Vote for me
ok
C
F


X
L
Log enteries
&
Heartbeats
F



System is down as It need more than 1 votes to promote to leader in 3 node system

Leader Election (Best case scenario)

Refer diagram attached separately

Leader Election (Split Vote)

Refer diagram attached separately

Log Replication

Refer diagram attached separately

Log Replication (with network partitions)

Refer diagram attached separately

A Byzantine failure

a condition of a distributed computing systems, where **components may fail and there is imperfect information** on whether a component has failed.

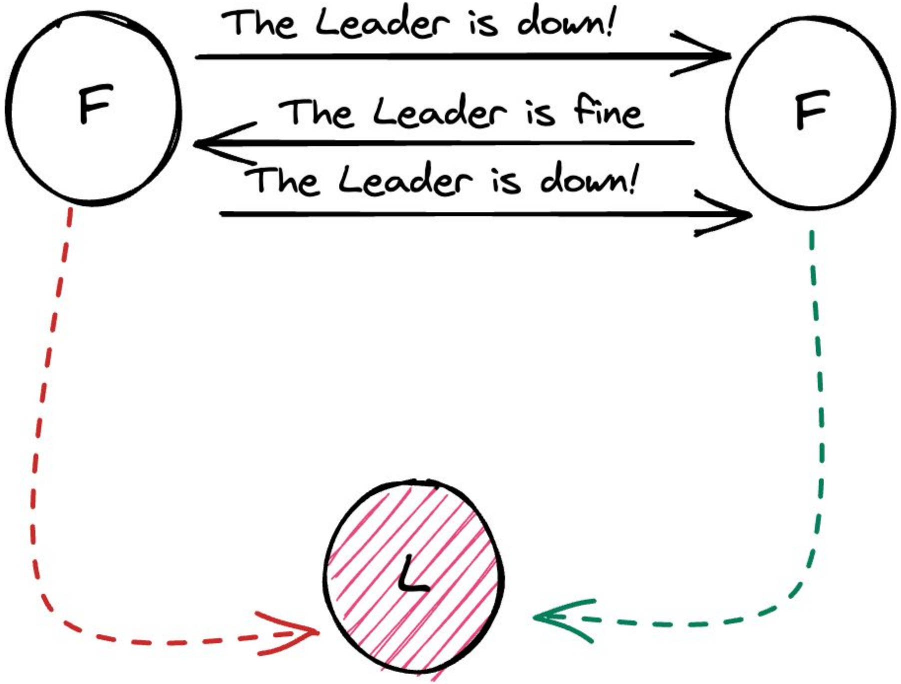
a component such as a server can inconsistently **appear both failed and functioning** to failure-detection systems, presenting different symptoms to different observers.

etcd - A Byzantine failure is real

- When cluster leader fails, **etcd uses RAFT protocol** to maintain consistency and establish consensus to promote a new leader.
- In the RAFT protocol, cluster members are assumed to be either **available or unavailable**, and to provide **accurate information or none at all**.
- This works fine when a machine crashes, but is not always able to handle situations where different members of the cluster have **conflicting information**.

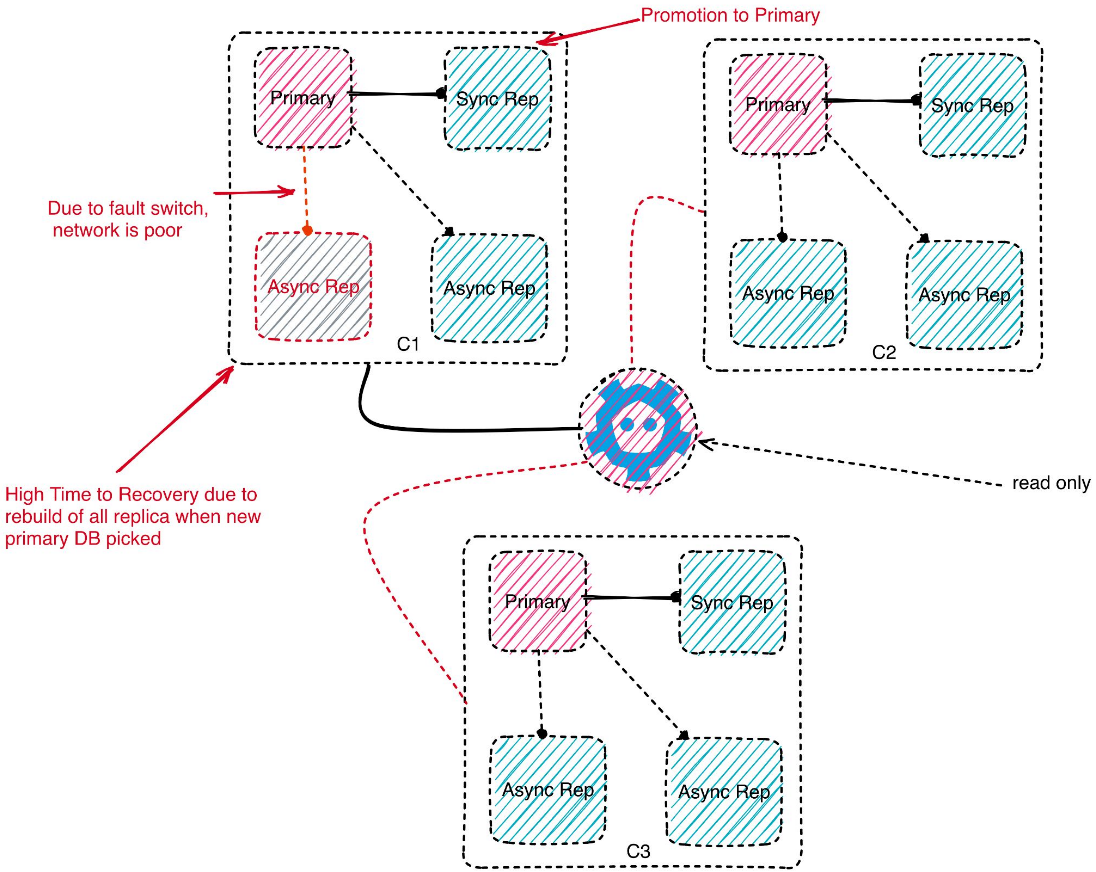
RAFT - A Byzantine failure is real

This caused cluster members to have conflicting views of reality, known in distributed systems theory as a Byzantine fault.



In 2020, Problem faced by

CLOUDFLARE



Promotion to Primary

Due to fault switch,
network is poor

High Time to Recovery due to
rebuild of all replica when new
primary DB picked

read only

Impact of Byzantine failure in cloudflare

- API success rate **dipped to 75%**.
- Redundant cluster nodes were unreachable due to **limbo in leader election**.
- **Read only** operation permitted to database.
- More **pressure to single cluster** for serving traffic.
- Downtime of around **6 hrs 33 min**.
- Dashboard api response **80 times slower** compared to normal.

Thank You

Happy

Learning