System models

Permissioned vs unpermissioned and failure models

Leander Jehl

System models

Permissioned vs unpermissioned

Unpermissioned: A system that anyone can join, usually anonymously.
Example: Bitcoin, Ethereum, PoW or PoStake blockchain

Permissioned: Need permission to join a network. System comprised of nodes with known identity.

Example:

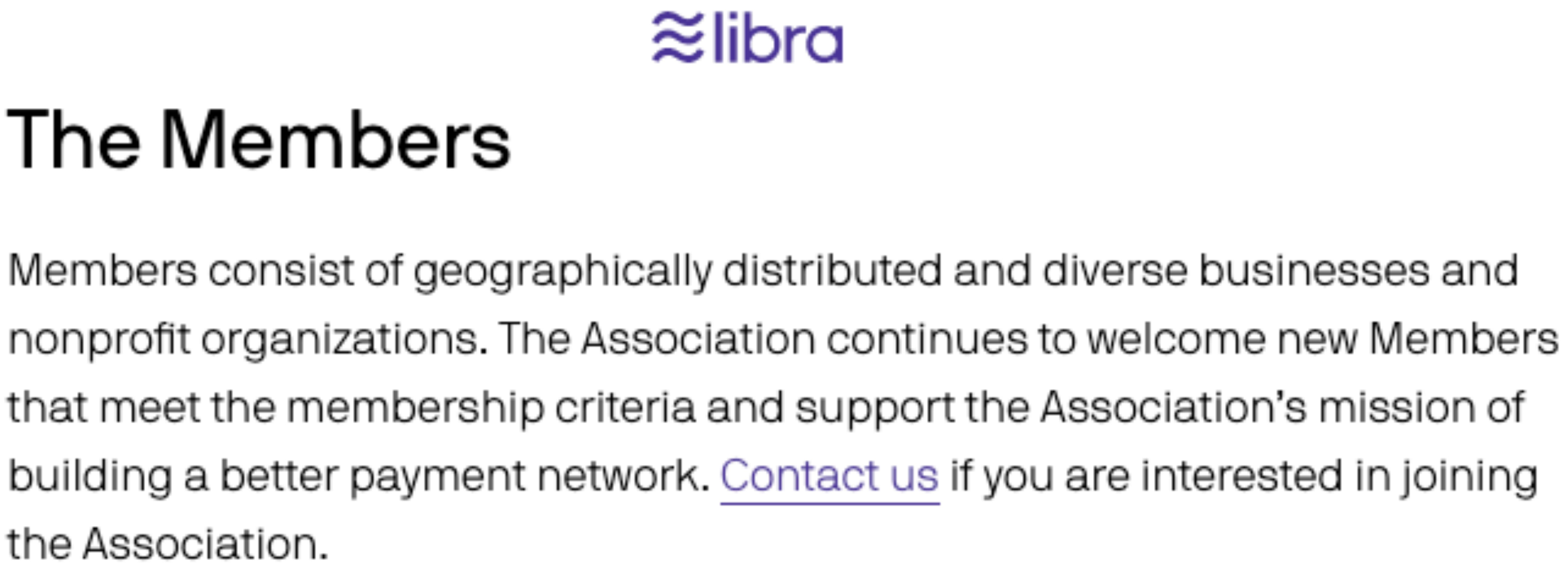
- Several organisations running a system together, each running one servers.

Permissioned systems

Libra

Example:

- Several organisations running a system together, each running one server



The Members

Members consist of geographically distributed and diverse businesses and nonprofit organizations. The Association continues to welcome new Members that meet the membership criteria and support the Association's mission of building a better payment network. Contact us if you are interested in joining the Association.

Permissioned systems

Paxos - BFT

Example:

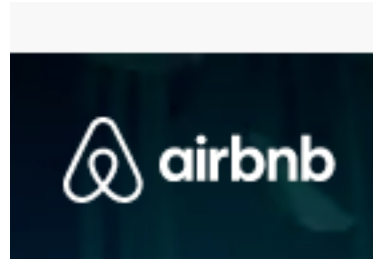
- Multiple servers, replicating one system for fault tolerance.

Permissioned systems

Identity based system

Example:

- Each peer is uniquely identified as person by passport, Bank-ID, Social-media profile



Permissioned systems

Properties

Permissioned: Need permission to join a network. System comprised of nodes with known identity.

- List of participants available
- Participants addressable, e.g. by public key and IP

Failure Models

Failure Models

Crash

Crash failure: Assumes nodes may stop responding at arbitrary points. But nodes do not behave against the protocol.

- Cannot use a single coordinator since coordinator may fail.
- Typically system is functional if only a fraction of the nodes fail.
- Example: Paxos (DAT520), less than half of the nodes may fail.
- Possible to have services that are correct (safe), but possibly not functional (live) even if all but one node fail.

Failure Models

Byzantine

Byzantine failures: Assume a fraction of the nodes may fail arbitrarily, i.e. they may stop or even disobey/attack the protocol.

- Typically up to 1/3 or 1/2 of the nodes may fail.
- If failure threshhold is violated, bad things may happen.
- Example Bitcoin

Failure Models

Byzantine fault tolerance (BFT)

Byzantine failures: Assume a fraction of the nodes may fail arbitrarily, i.e. they may stop or even disobey/attack the protocol.

- Typically up to 1/3 or 1/2 of the nodes may fail.
- If failure threshhold is violated, bad things may happen.
- Example Bitcoin
- If attacker has 1/3 of mining power he can do selfish mining, but
  - cannot steal bitcoin.

Failure Models

Rational

Rational failures: Nodes has a well defined utility function and will deviate from the protocol, if it increases their utility.

- Different from BFT here all nodes may fail at once.
- Utility functions based on
  - Reward
  - Access to functioning service
  - Cost (computation, networking, …)

Failure Models

Rational

Rational failures: Nodes has a well defined utility function and will deviate from the protocol, if it increases their utility.

- Different from BFT here all nodes may fail at once.
- Utility functions based on
  - Reward
  - Access to functioning service
  - Cost (computation, networking, …)
- Protocol may be game theoretic equilibrium

Nash-Equilibrium

Rational failures

Example: Prisoners dilemma

To prisoners are interrogated separately. Both can either choose to confess or diny charges.

- *If both diny, both get one year in jail.*
- *If both confess, both get 2 years in jail.*
- *If only one confesses, he goes free and the other one gets 3 years in jail.*

Without knowing what the other one will do, a prisoner can improve his uility, by confessing.

Nash-Equilibrium

Rational failures

A **Nash-Equilibrium** is a set of strategies strategy s.t. if all other players follow this strategy, a single player cannot improve his utility by deviation.

In the prisoners dilemma, the strategy where both prisoners confess is a nash-equilibrium.

Failure Models

BAR Fault tolerance

In **BAR fault tolerance**, we assume that a large fraction of nodes is rational, few nodes are honest (do not fail) and some nodes are byzantine.

In a game theoretic view, a byzantine player is one that has a unknown or different utility function.

Open permissioned system example

Stellar (not curriculum)

- In Stellar, not all nodes are equal. Each node defines his requirements.
- Slides: https://sosp19.rcs.uwaterloo.ca/slides/mazieres.pdf
- Video: https://sosp19.rcs.uwaterloo.ca/videos/D1-S2-P2.mp4