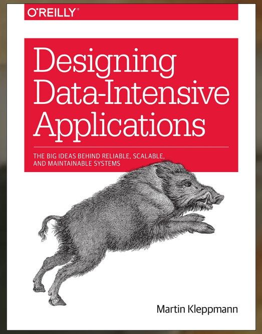
MSFT Sys Meetup



O'REILLY

Designing Data-Intensive Applications

THE BIG IDEAS BEHIND RELIABLE, SCALABLE, AND MAINTAINABLE SYSTEMS

Martin Kleppmann

http://

Policy Against Harassment at ACM Activities

https://www.acm.org/about-acm/policy-against-harassment

MSFT Sys Meetup wants to encourage and preserve this open exchange of ideas, which requires an environment that enables all to participate without fear of personal harassment. We define harassment to include specific unacceptable factors and behaviors listed in the ACM's policy against harassment. Unacceptable behavior will not be tolerated.

Freely accessible resources

Code

Zoom

Course

DDIA (O'Reilly)

Distributed System 3rd edition

Calendar: https://docs.google.com/spreadsheets/d/1RsbGpq1cwNSmYn5hcmT8Hv5O4qssl2HXsTcG82RHVQk/edit?usp=sharing

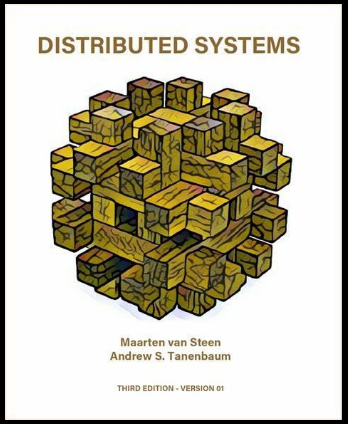
(Internal) Teams: g078pwd

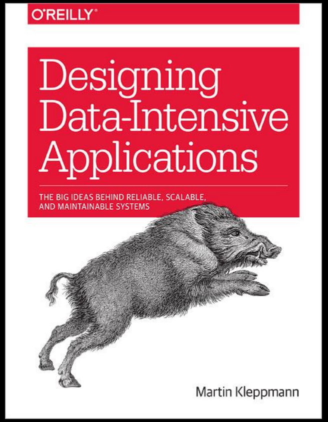
(Public) Discord

(Public) WeChat: add mossaka or Lin1991Wen

Notion: https://www.notion.so/invite/cd6df70a94e7f67f6d21f4c509783d3c9cfd0e69

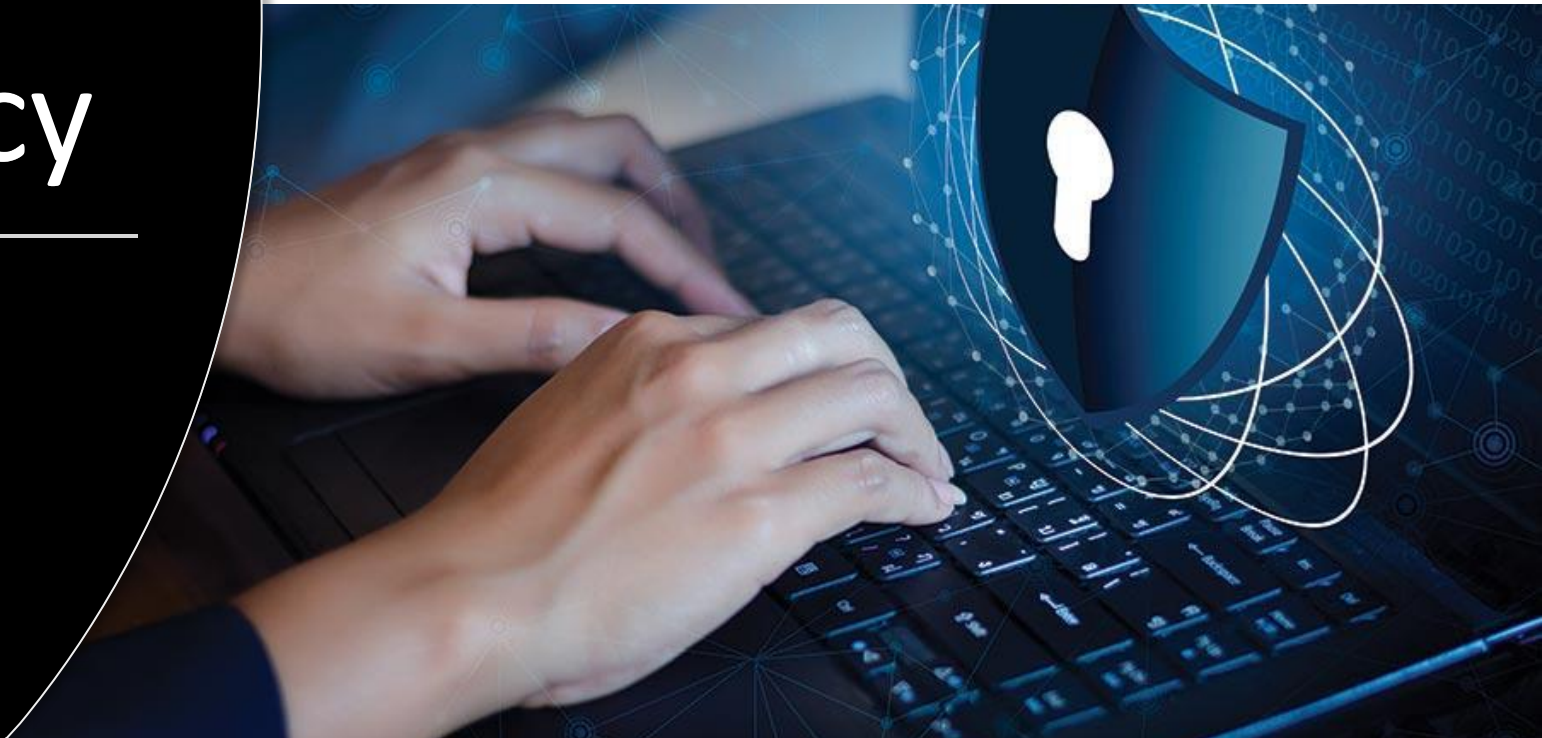
YouTube: https://www.youtube.com/playlist?list=PL1voNxn5MODMJxAZVvgFHZ0jZ-fuSut68




O'REILLY
Designing Data-Intensive Applications
THE BIG IDEAS BEHIND RELIABLE, SCALABLE, AND MAINTAINABLE SYSTEMS
Martin Kleppmann

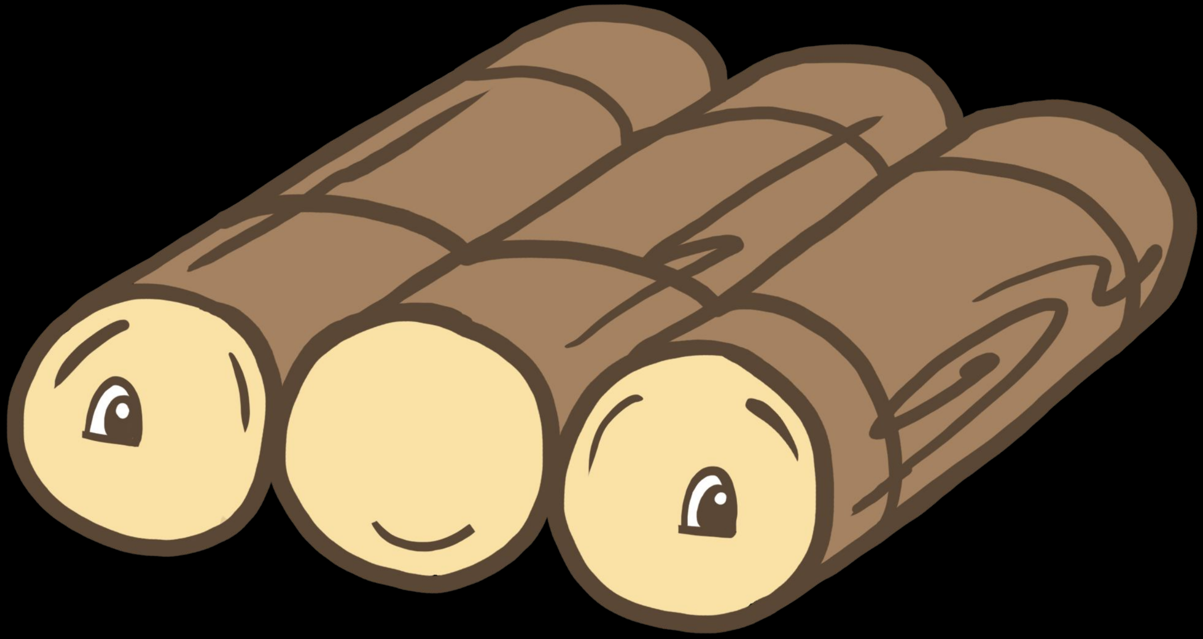
Company Privacy

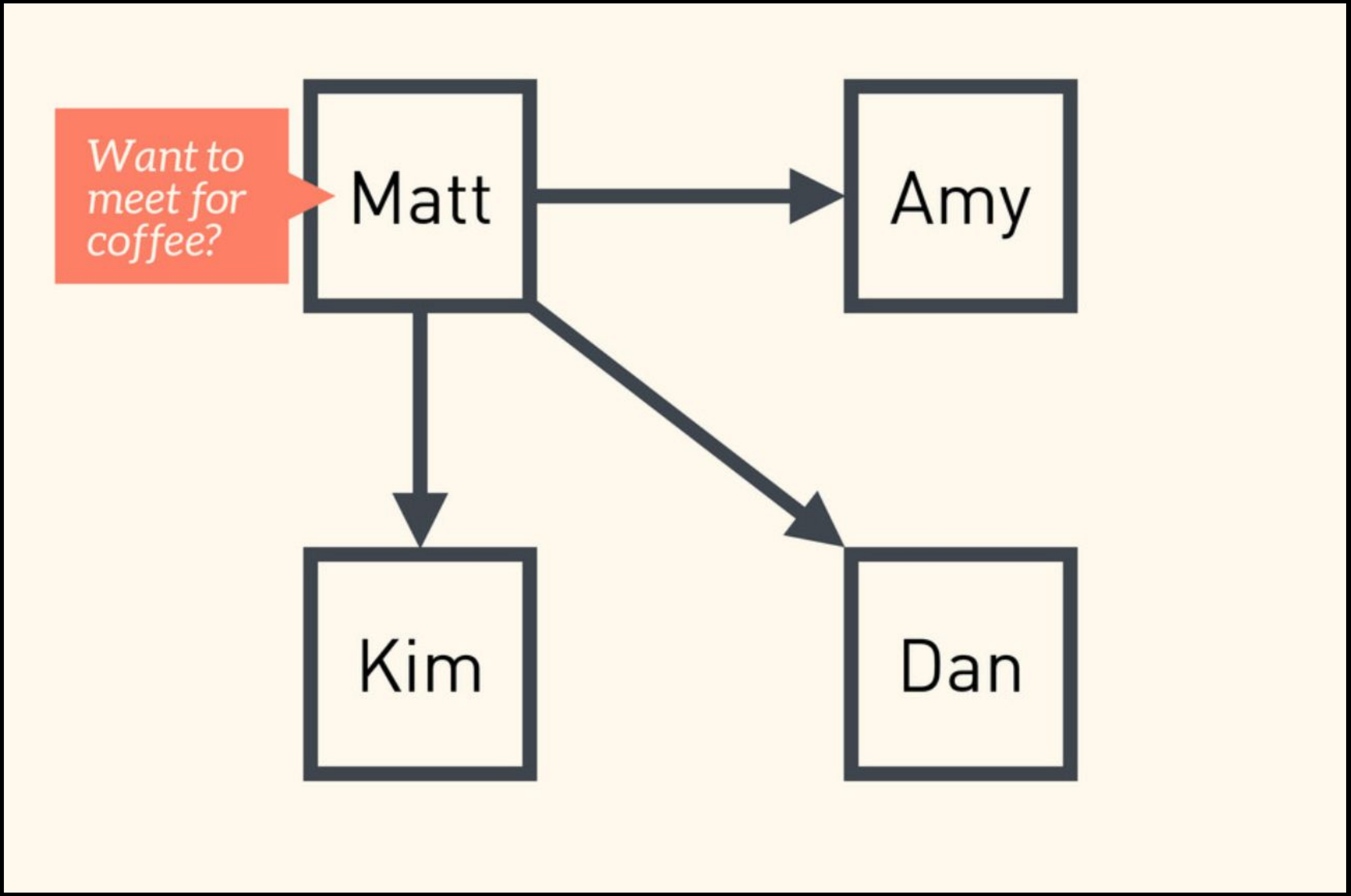
Microsoft

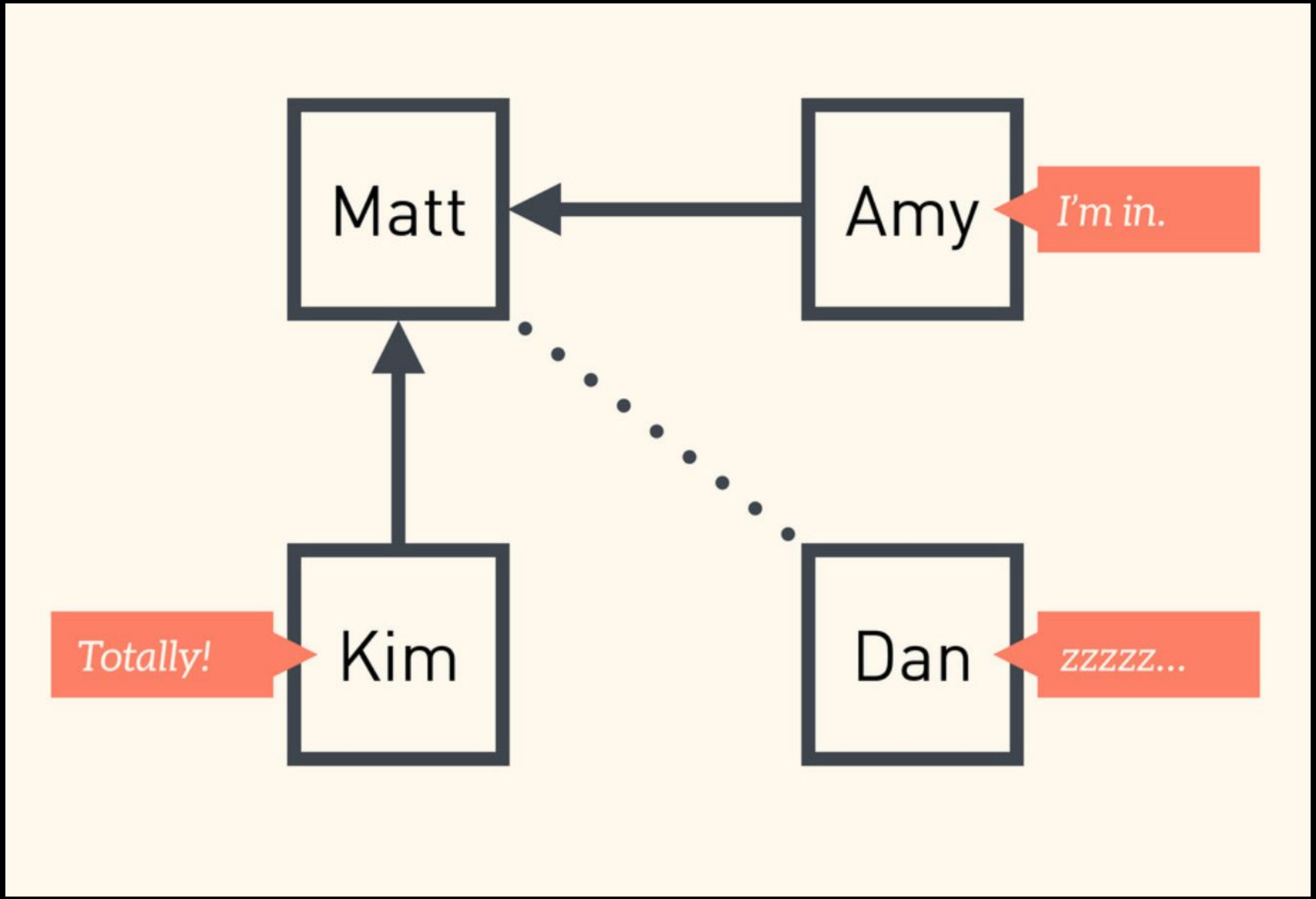


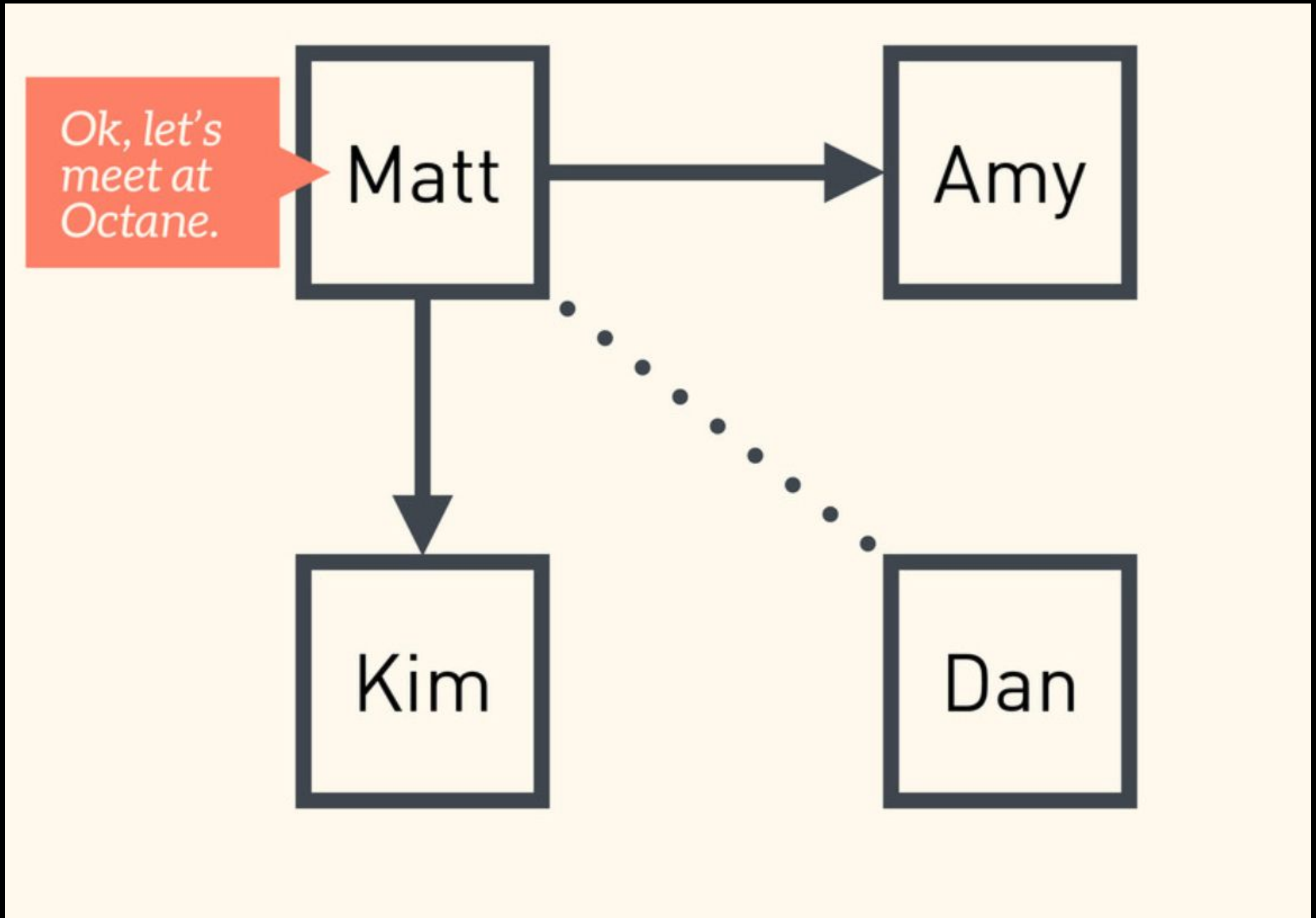
Topic Covered

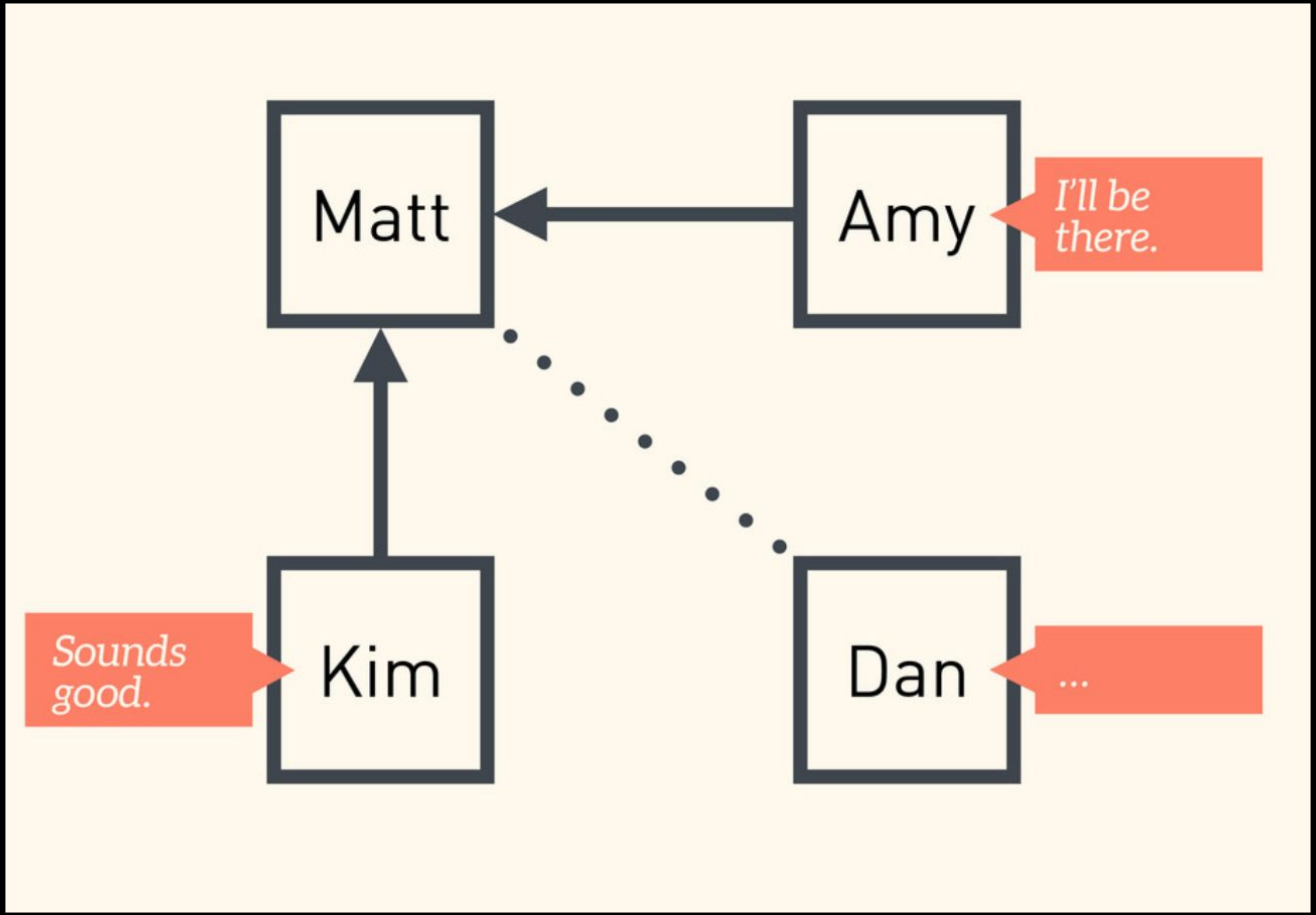
- What is raft
- Replicated state machines
- What is consensus
- Log structure
- Raft decomposition

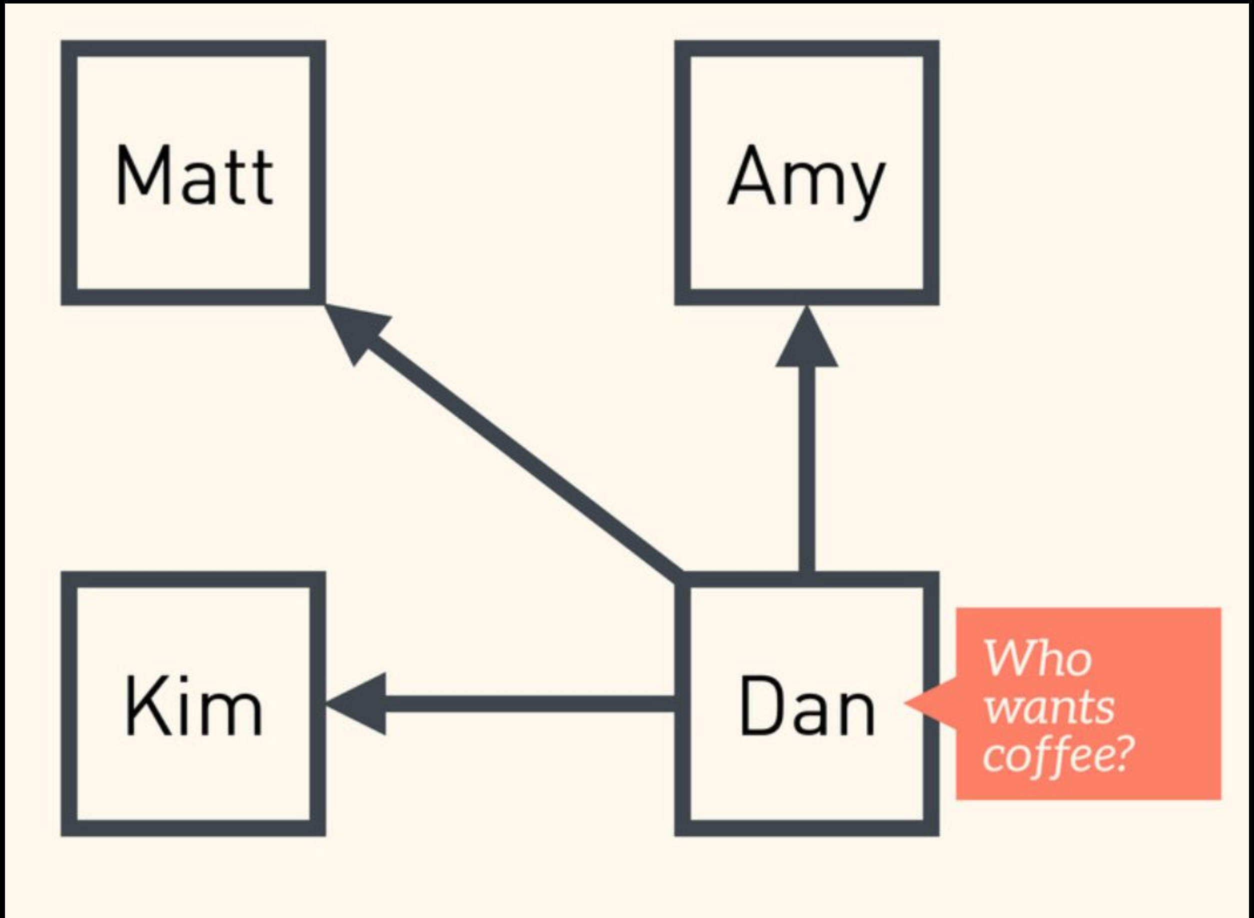


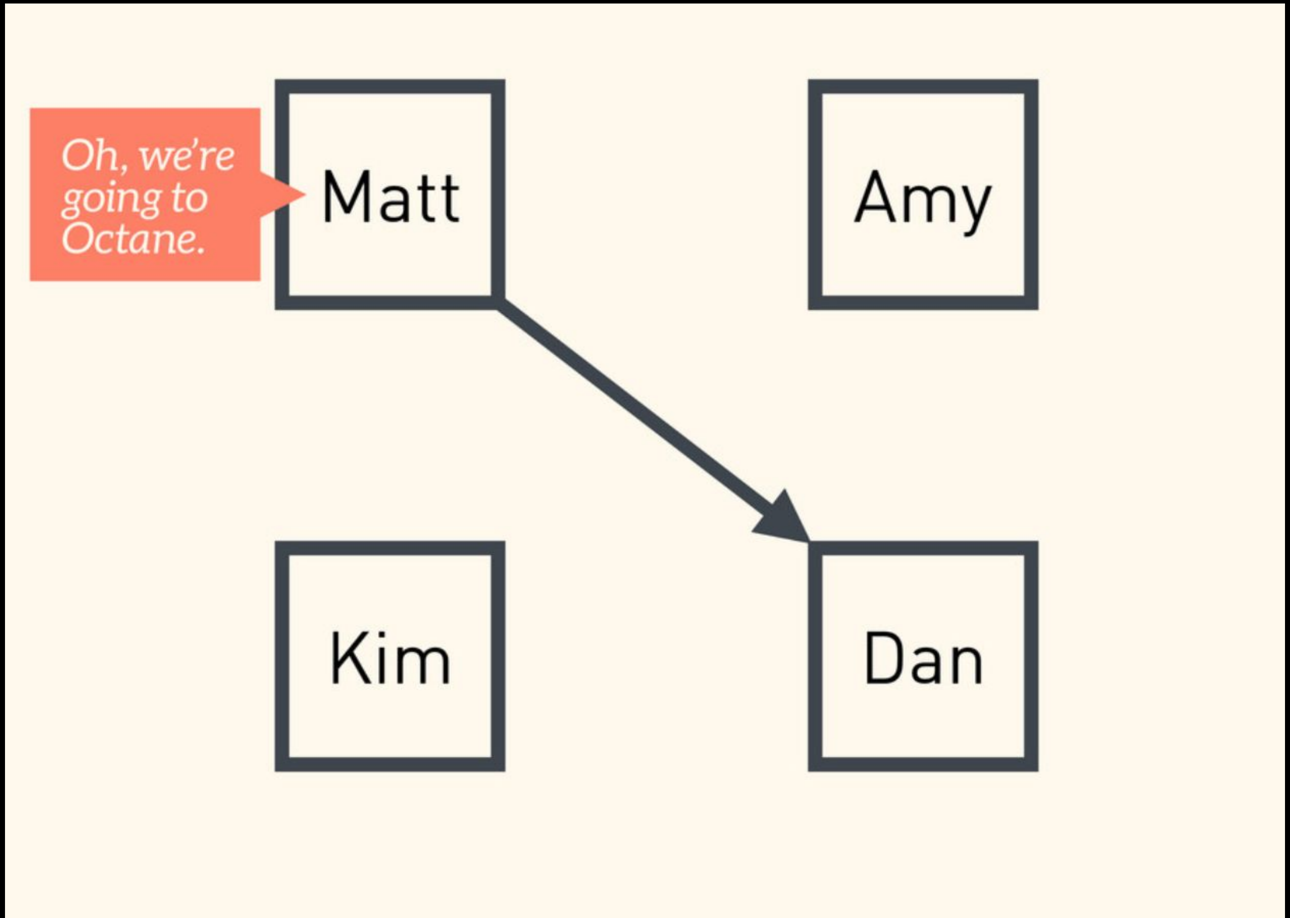

Want to meet for coffee?
Matt
Amy
Kim
Dan


Matt
Amy
I'm in.
Totally!
Kim
Dan
zzzzz...


Ok, let's meet at Octane.
Matt
Amy
Kim
Dan

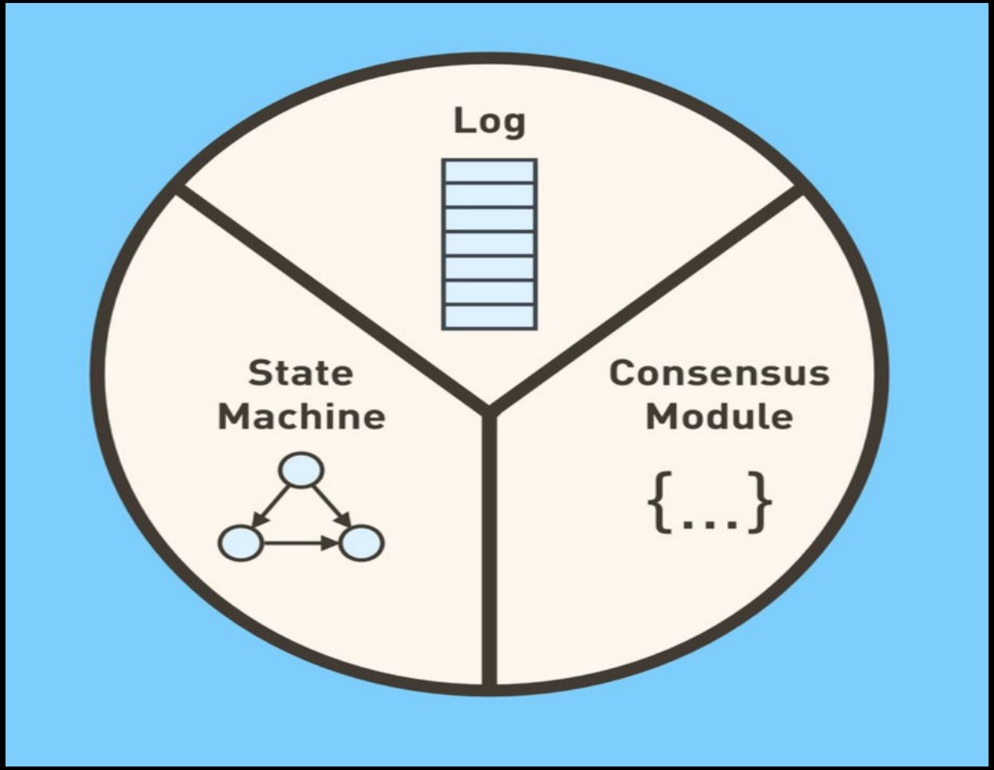

Matt
Amy
I'll be there.
Kim
Sounds good.
Dan
...

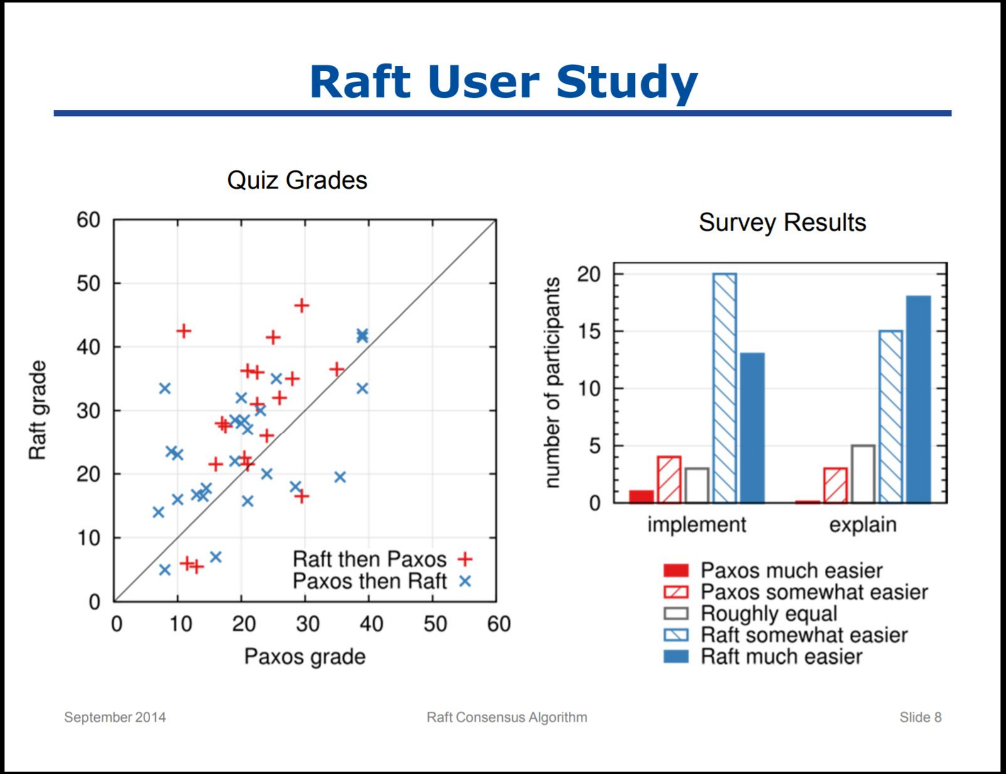

Matt
Amy
Kim
Dan
Who wants coffee?


Oh, we're going to Octane.
Matt
Amy
Kim
Dan

What is Raft?

- Raft is a consensus algorithm for managing a replicated log
- It's equivalent to Paxos in fault-tolerance and performance





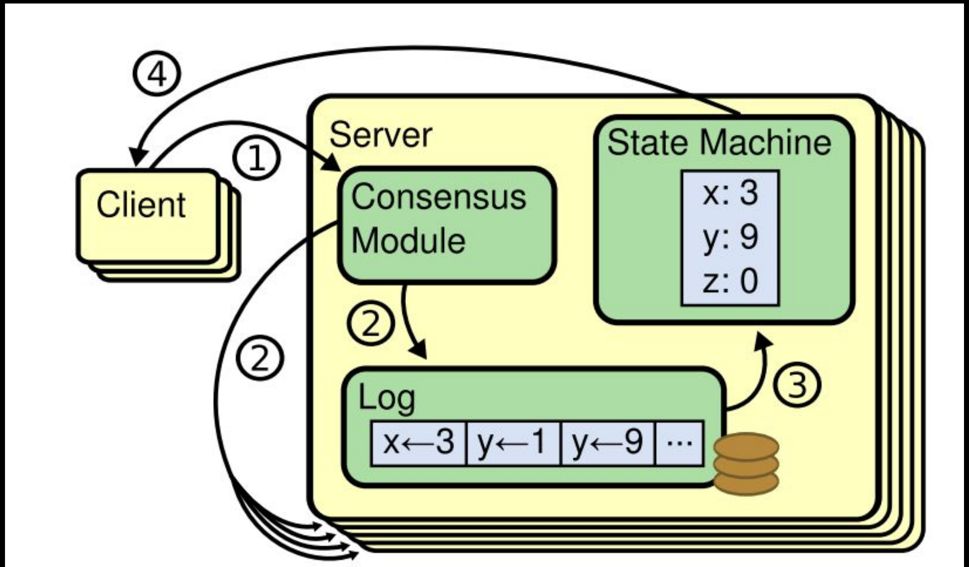
Raft User Study

September 2014 Raft Consensus Algorithm Slide 8

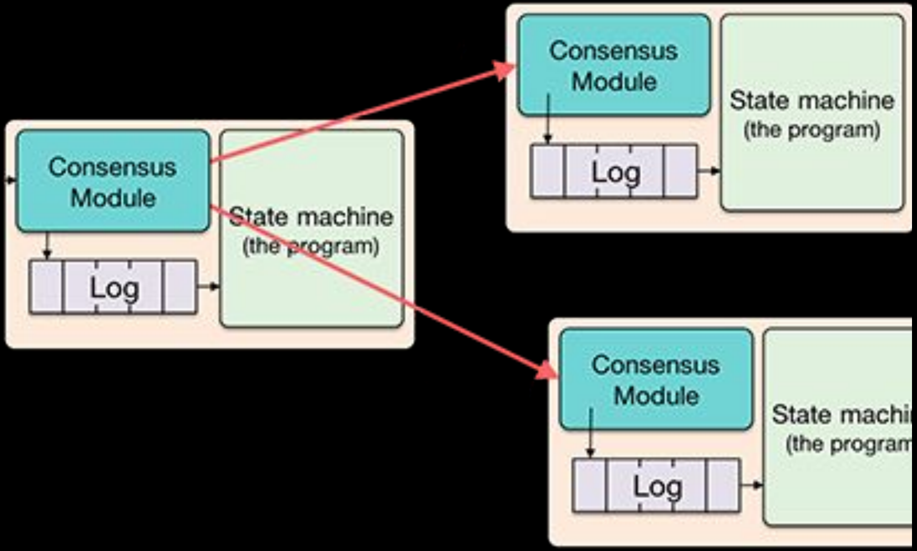
- “The dirty little secret of the NSDI community is that at most five people really, truly understand every part of Paxos ;-).” – NSDI reviewer

- “There are significant gaps between the description of the Paxos algorithm and the needs of a real-world system…the final system will be based on an unproven protocol.” – Chubby authors

Replicated state machines

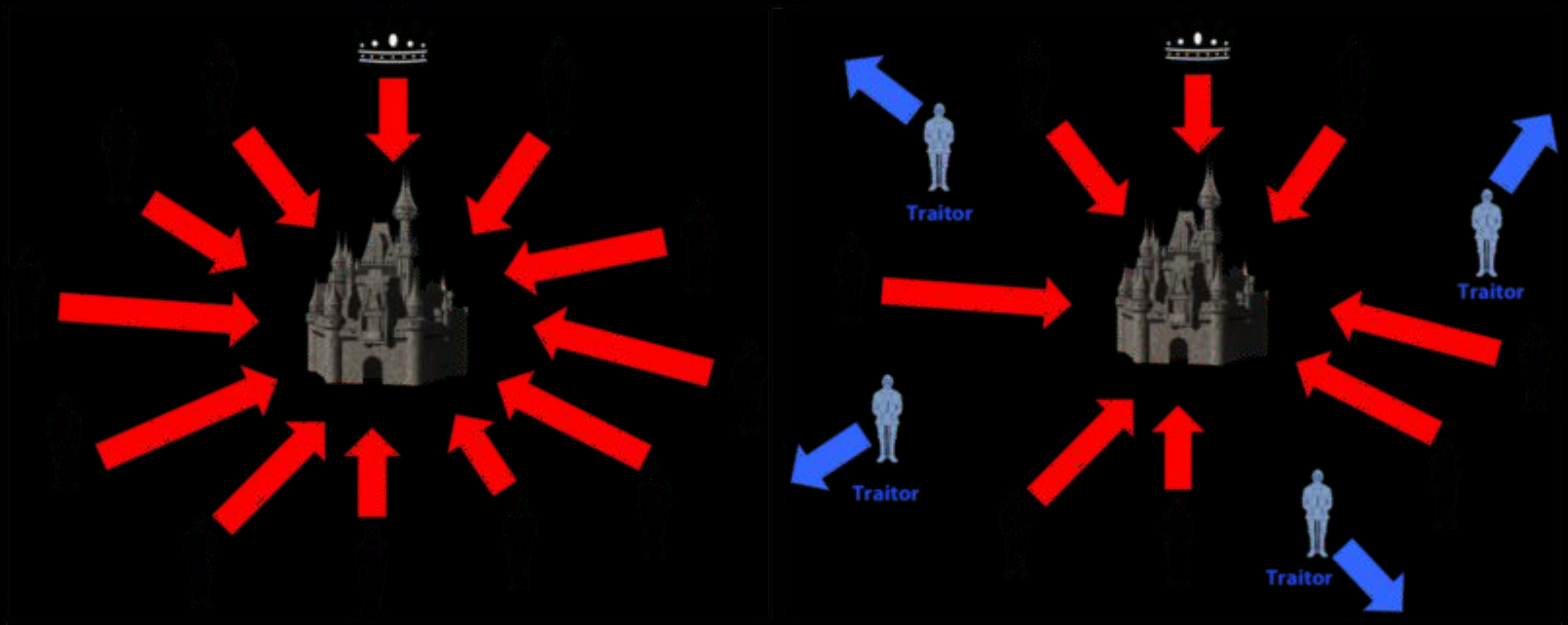


- The consensus algorithm manages a replicated log containing state machine commands from clients.
- The state machines process identical sequences of commands from the logs, so they produce the same outputs

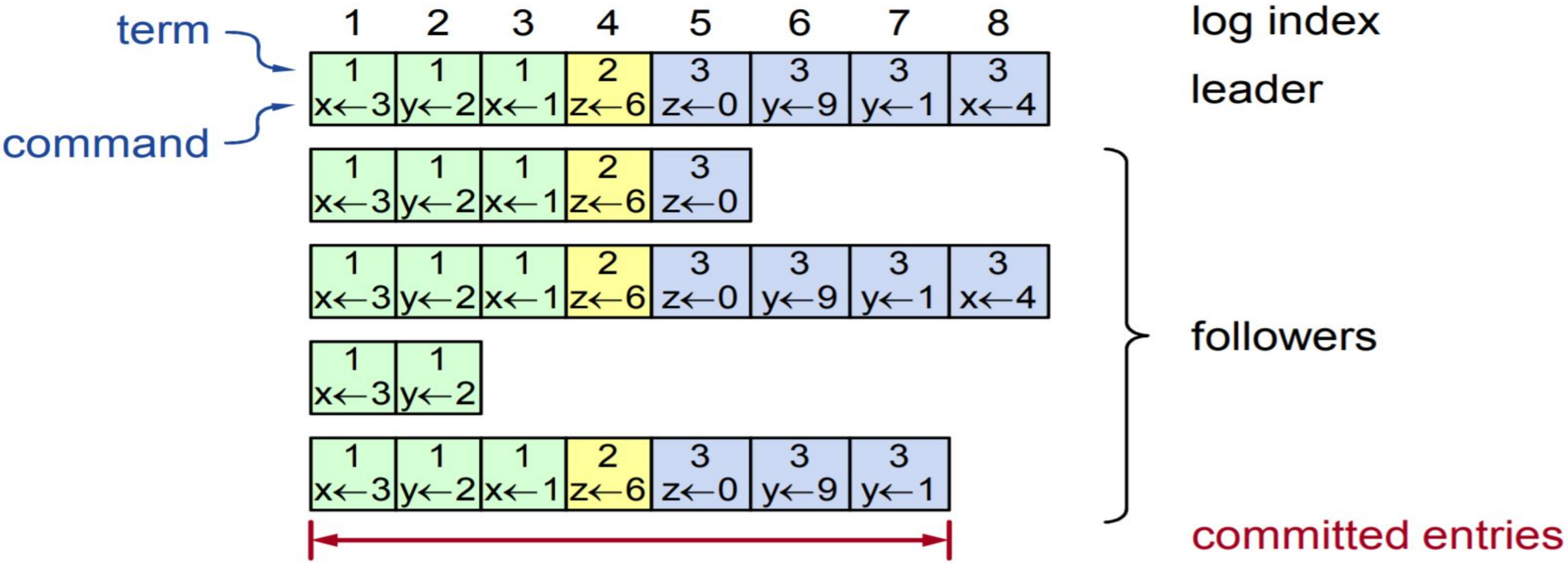


What is consensus?

- Consensus involves multiple servers agreeing on values
- Once they reach a decision on a value, that decision is final
- Make progress when any majority of their servers is available



Log Structure



term

command

1	2	3	4	5	6	7	8	log index
1 x←3	1 y←2	1 x←1	2 z←6	3 z←0	3 y←9	3 y←1	3 x←4	leader
1 x←3	1 y←2	1 x←1	2 z←6	3 z←0				followers
1 x←3	1 y←2	1 x←1	2 z←6	3 z←0	3 y←9	3 y←1	3 x←4	
1 x←3	1 y←2							
1 x←3	1 y←2	1 x←1	2 z←6	3 z←0	3 y←9	3 y←1		

committed entries

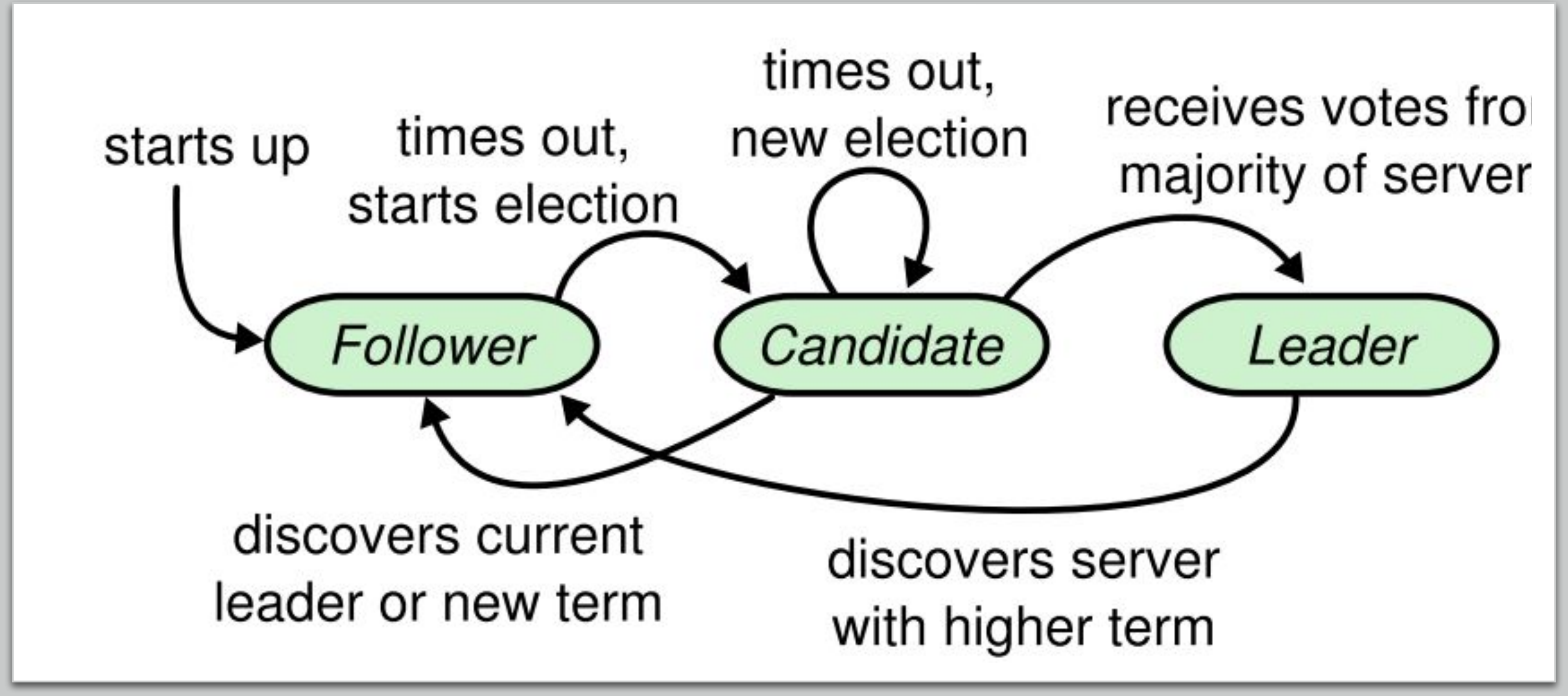
- **Log entry = index, term, command**
- **Log stored on stable storage (disk); survives crashes**

Raft Decomposition

1. **Leader election:**
   - Select one server to act as leader
   - Detect crashes, choose new leader
2. **Log replication (normal operation)**
   - Leader accepts commands from clients, appends to its log
   - Leader replicates its log to other servers (overwrites inconsistencies)
3. **Safety**
   - Keep logs consistent
   - Only servers with up-to-date logs can become leader

Server state

- **Leader**. The leader handles all client requests and responses. There is only one leader at a time.
- **Candidate**. A server may become a candidate during the election phase. One leader will be chosen from one or more candidates. Those not selected will become followers.
- **Follower**. The follower does not talk to clients. It responds to requests from leaders and candidates.



Messages

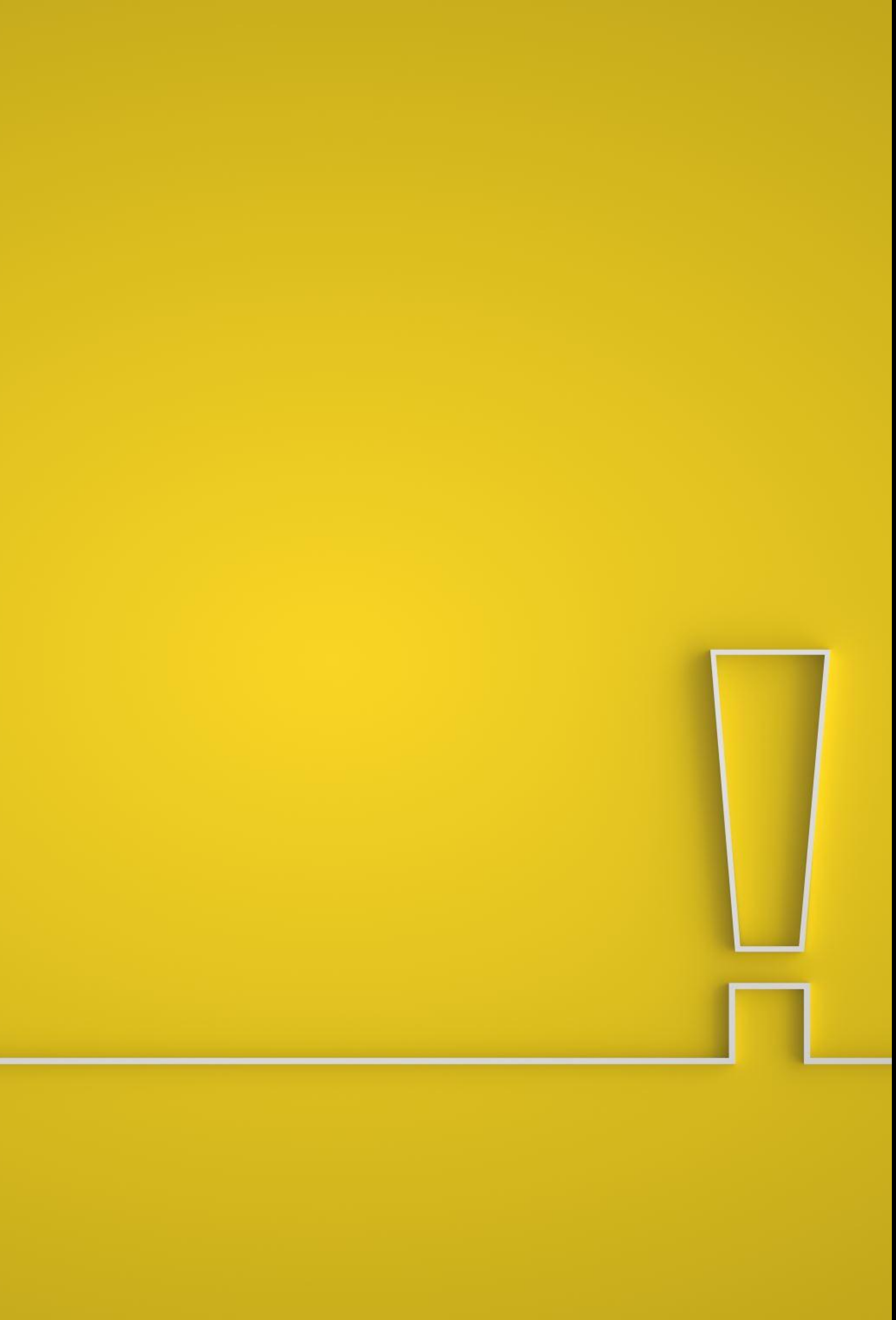
- **RequestVotes**
  - Used by a candidate during elections to try to get a majority vote.
- **AppendEntries**
  - Used by leaders to communicate with followers to:Send log entries (data from clients) to replicas.
  - Send commit messages to replicas. That is, inform a follower that a majority of followers received the message.
  - Send heartbeat messages. This is simply an empty message to indicate that the leader is still alive.

Term

- Act as a logic clock
- Increase monotonically
- Each server stores current term number
- Current terms are exchanged whenever servers communicate
- Server rejects the request with a stale term
- If a candidate or leader discovers its term is out of date, it will revert to follower

Election timeout

- Split brain
- Randomized election timeout
- Alternative: ranking system



Safety

- If a server receives a *RequestVotes* message and the candidate has an earlier term then the server will reject the vote.
- If the term numbers are the same but the log length of a candidate is shorter than that of the server that receives the message, the server will reject the vote.

Questions?

Freely accessible resources

Code

Zoom

Course

DDIA (O’Reilly)

Distributed System 3rd edition

Calendar: https://docs.google.com/spreadsheets/d/1RsbGpq1cwNSmYn5hcmT8Hv5O4qssl2HXsTcG82RHVQk/edit?usp=sharing

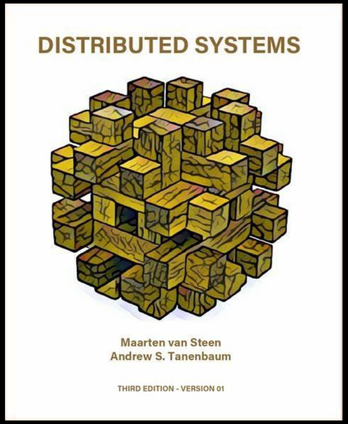
(Internal) Teams: g078pwd

(Public) Discord

(Public) WeChat: add mossaka or Lin1991Wen

Notion: https://www.notion.so/invite/cd6df70a94e7f67f6d21f4c509783d3c9cfd0e69

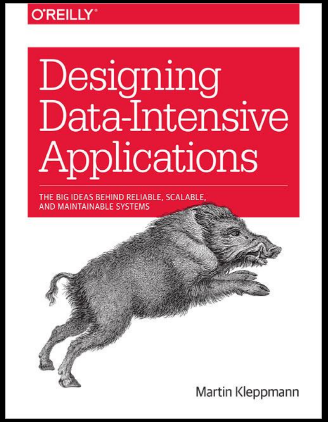
YouTube: https://www.youtube.com/playlist?list=PL1voNxn5MODMJxAZVvgFHZ0jZ-fuSut68



DISTRIBUTED SYSTEMS

Maarten van Steen
Andrew S. Tanenbaum

THIRD EDITION - VERSION 01



O'REILLY

Designing Data-Intensive Applications

THE BIG IDEAS BEHIND RELIABLE, SCALABLE, AND MAINTAINABLE SYSTEMS

Martin Kleppmann