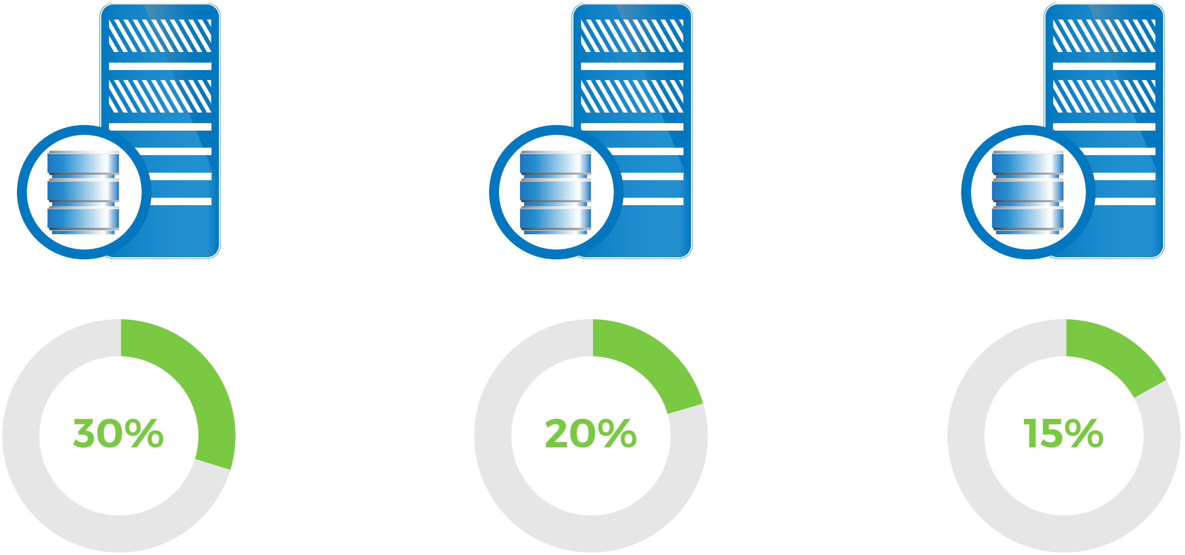
How choosing the Raft consensus algorithm saved us 3 months of development time

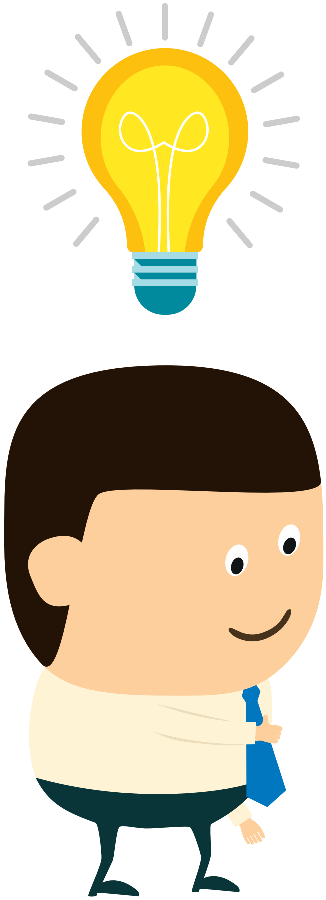
FOSDEM 16 .org

Brussels 30 & 31 January

What do I do with unused space on my servers?



Let’s build an S3 cluster!



Requirements:

- Fully S3 compatible
- Easy to maintain
- Fault tolerant

I found a great candidate:
SX + LibreS3

Bonuses:

- Block level deduplication
- Highly scalable
- Multiplatform

… but something was missing!

What about automatic failover?

Almost there!

- Fully distributed
- Data replication
- Cluster membership management

... but no support for detecting and kicking out dead nodes

How to deal with the failure?

- Some node has to make a decision
- Decisive node must not be faulty
- All the alive nodes should follow

There is a need for a consensus algorithm.

Choosing the algorithm

Paxos:

- Proven to work
- Very complicated
- Many variants and interpretations (ZooKeeper, …)

Raft:

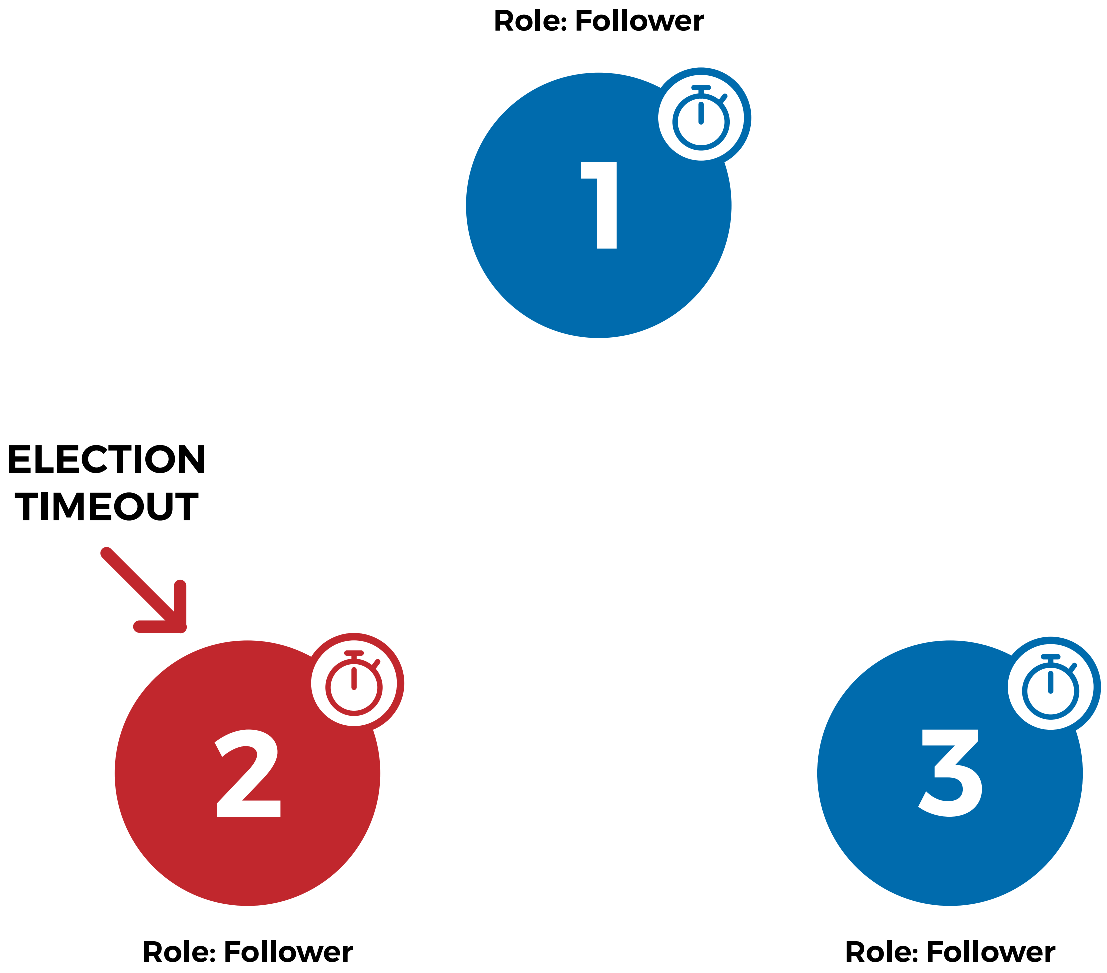
- Easy
- Straightforward implementation
- Accurate and comprehensive specs

And the winner is… Raft!

Raft

How does it work?

Leader election



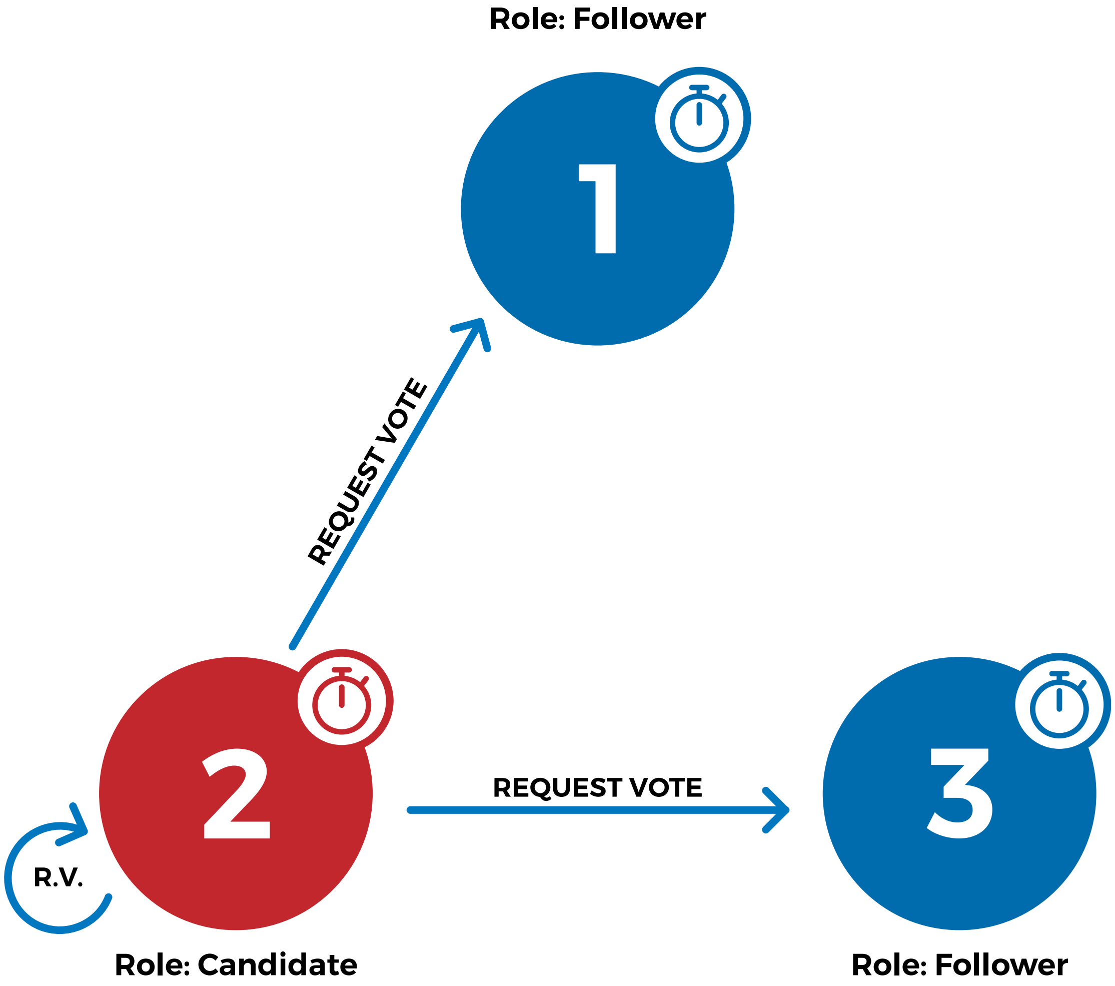
Role: Follower

ELECTION TIMEOUT

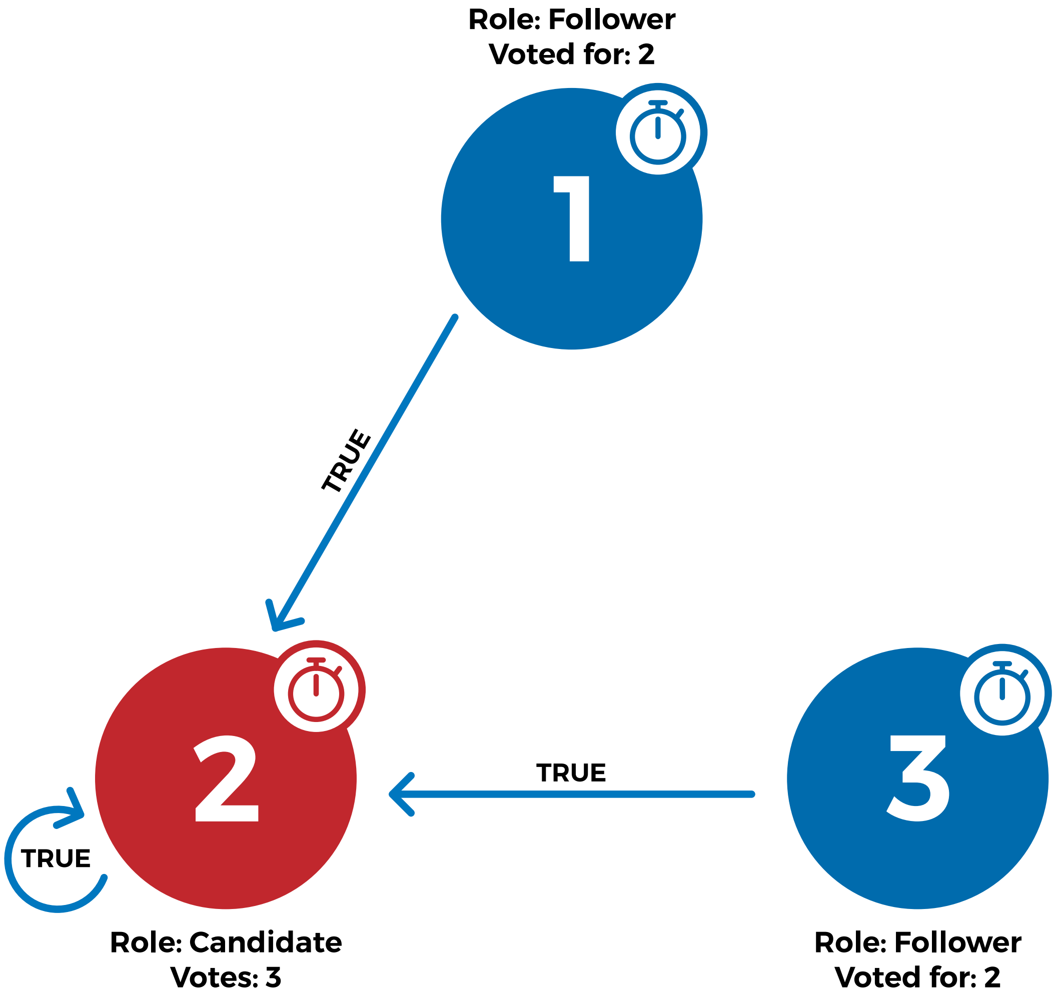
Role: Follower

Role: Follower

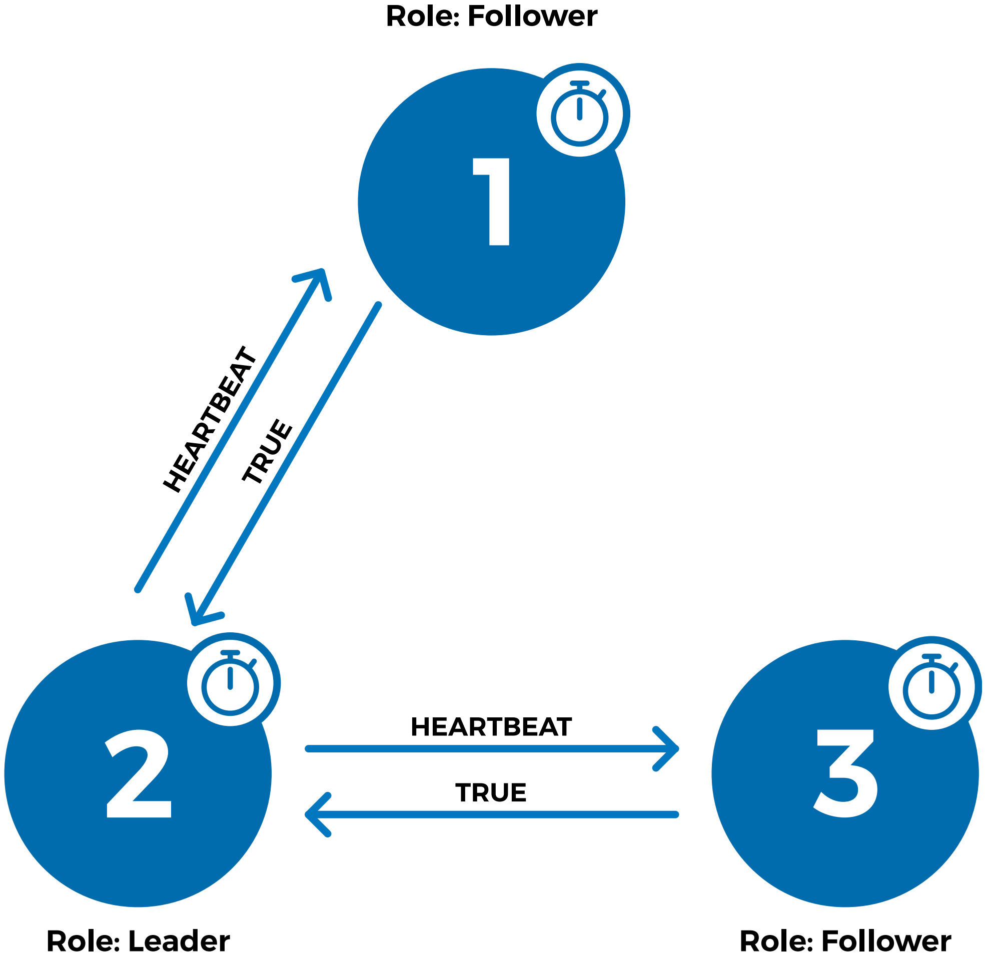
Leader election



Leader election



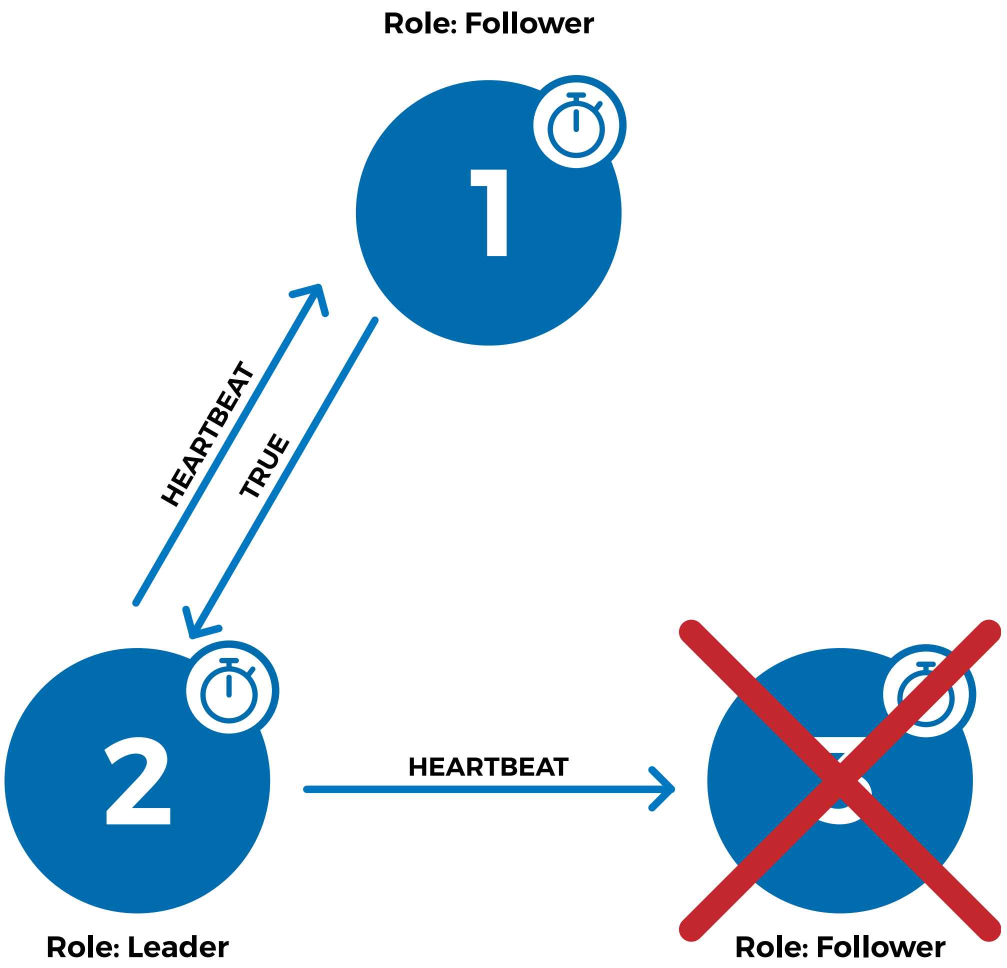
Leader election



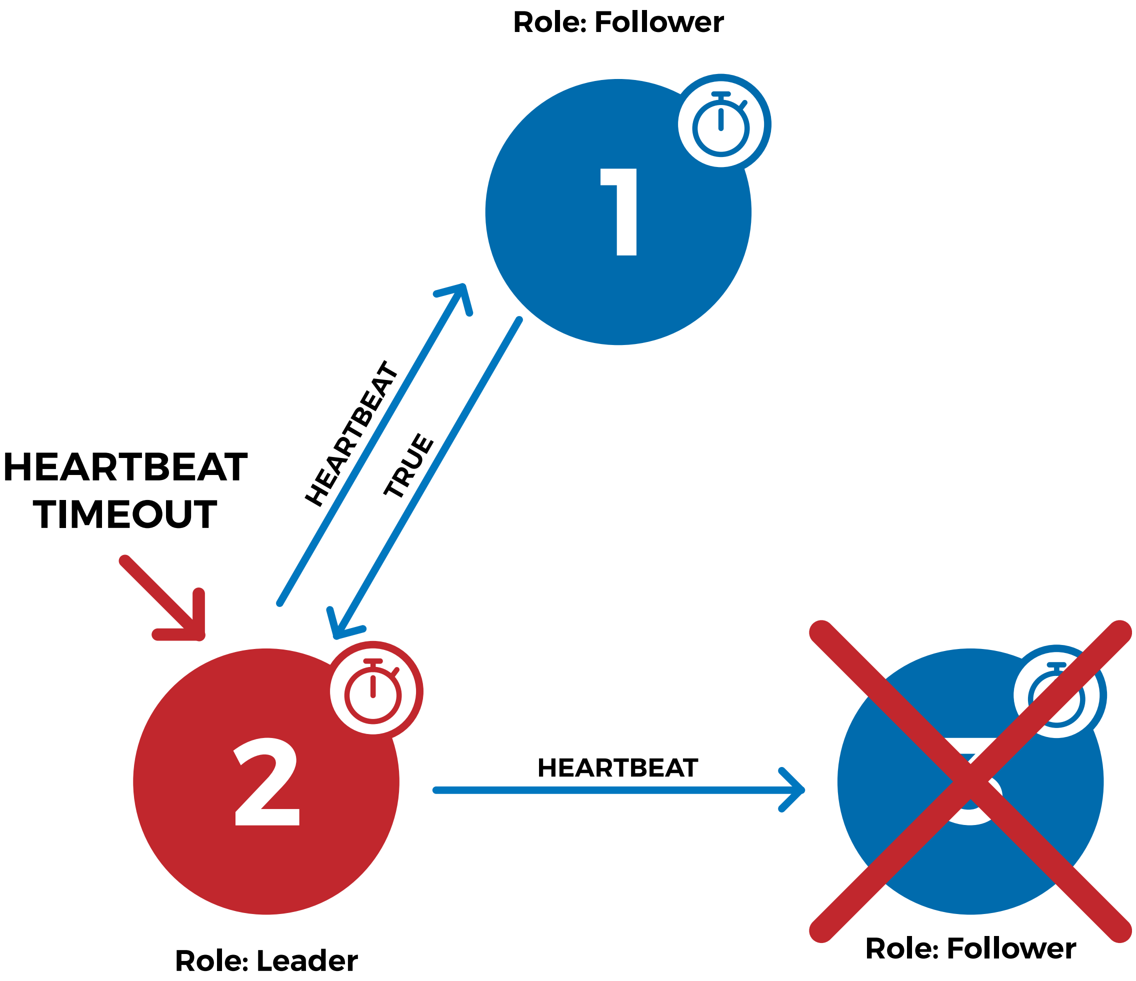
Raft

Node failure

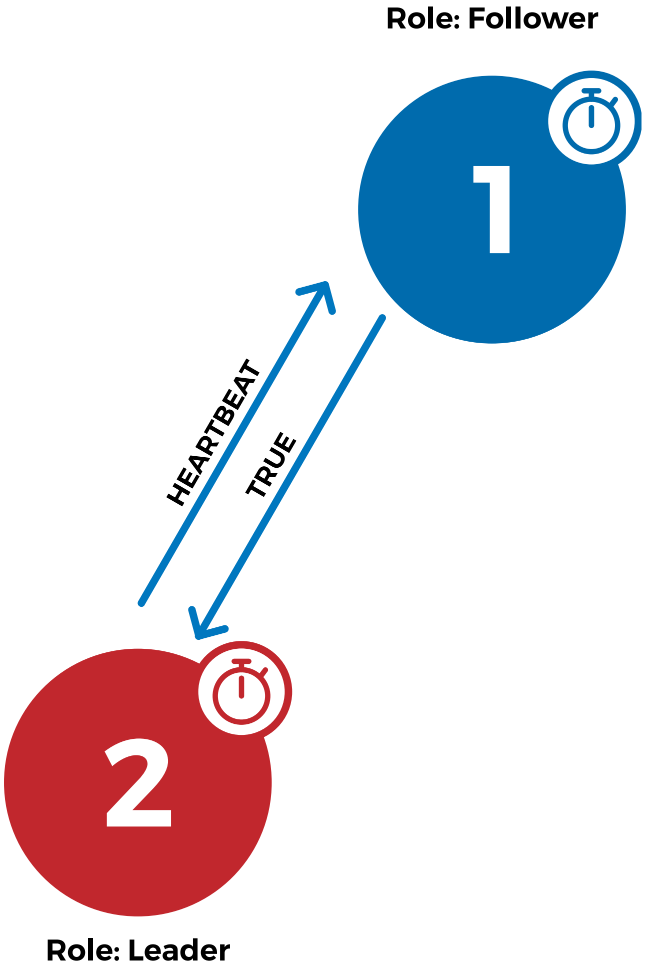
Dead node detection



Dead node detection



Dead node detection



How I implemented Raft in SX

Implementation details

- Heartbeats are sent via internal SX communication
- Membership changes are performed automatically
- Node failure detection relies on configurable timeouts
- Almost no impact on SX performance

How to enable Raft in SX?

Enable Raft node failure timeout:

```
$ sxadm cluster --set-param hb_deadtime=120 \
sx://admin@sx.foo.com
```

Kill one of the nodes and check its status:

```
$ sxadm cluster –I sx://admin@sx.foo.com
  * node 10…da: … status: follower, online: ** NO **
  * node bd…ad: … status: follower, online: yes
  * node c2…b7: … status: leader, online: yes
```

Wait for the node to be marked as faulty:

```
$ sxadm cluster –I sx://admin@sx.foo.com
  * node 10…da: … status: follower, online: ** FAULTY **
  * node bd…ad: … status: follower, online: yes
  * node c2…b7: … status: leader, online: yes
```

www.skylable.com

Robert Wojciechowski

follow @skylable

Stay tuned…

Coming up next: SXFS

FUSE based filesystem mapping for SX:

- Client-side encrypted
- Fully deniable
- Deduplication
- Fault tolerant

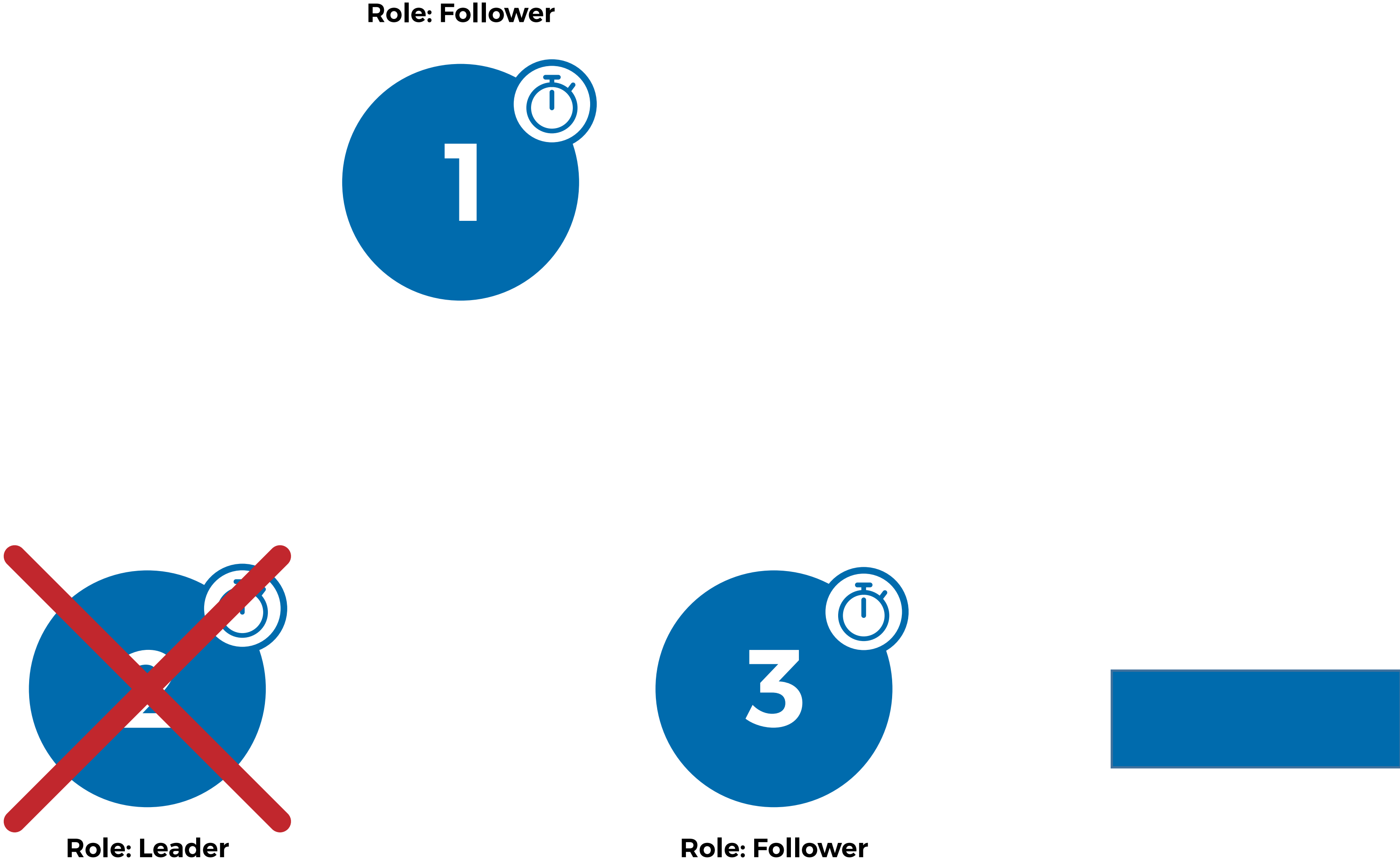
The election basics

- There is only one legitimate leader
- Each node chooses a timeout
- When timeout is reached a new election is started
- A candidate node votes for itself
- The candidate requests a vote
- In case the candidate received a majority of votes it becomes a new leader

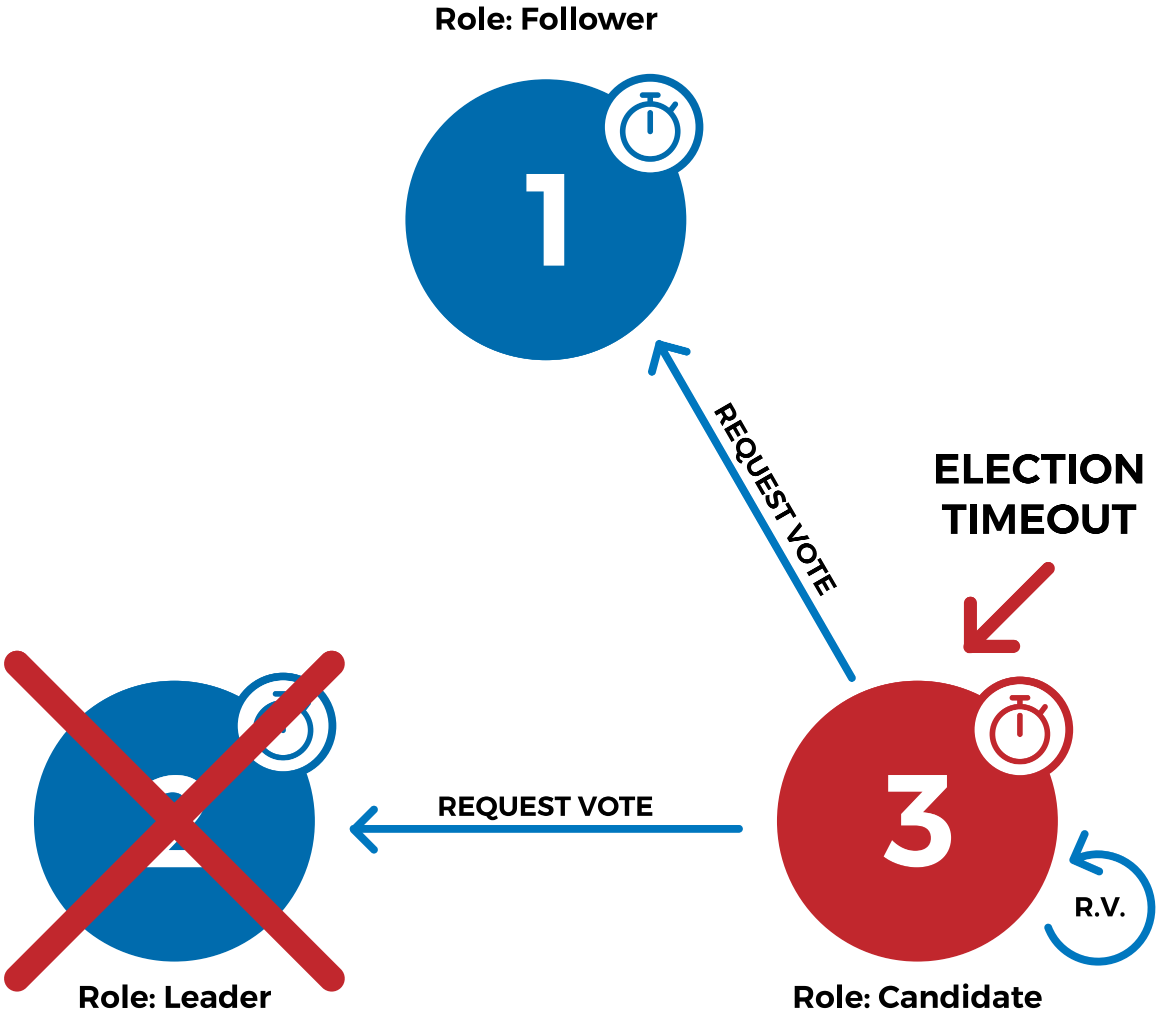
Corner cases

Leader failure

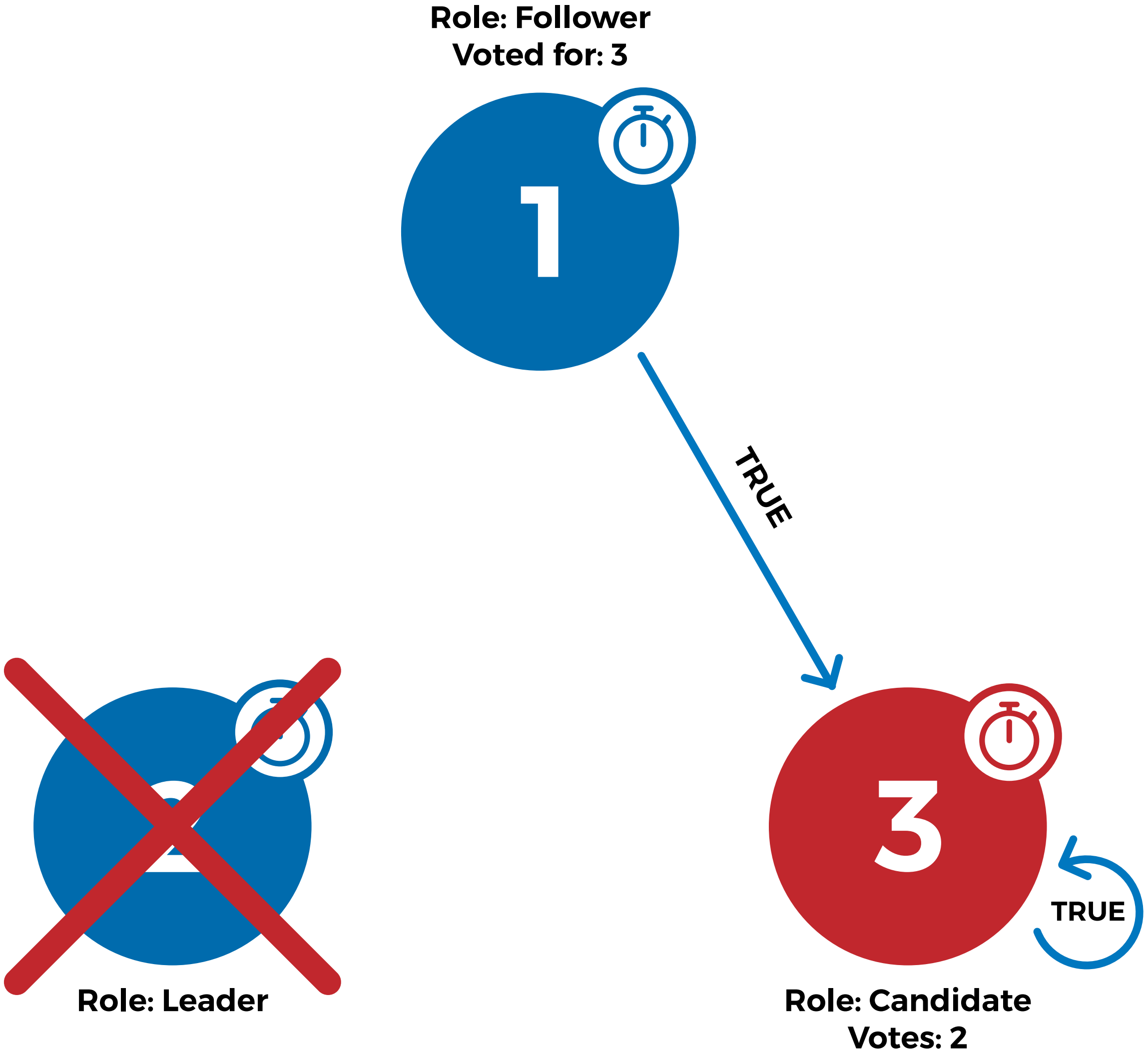
Leader node failure



Leader node failure



Leader node failure

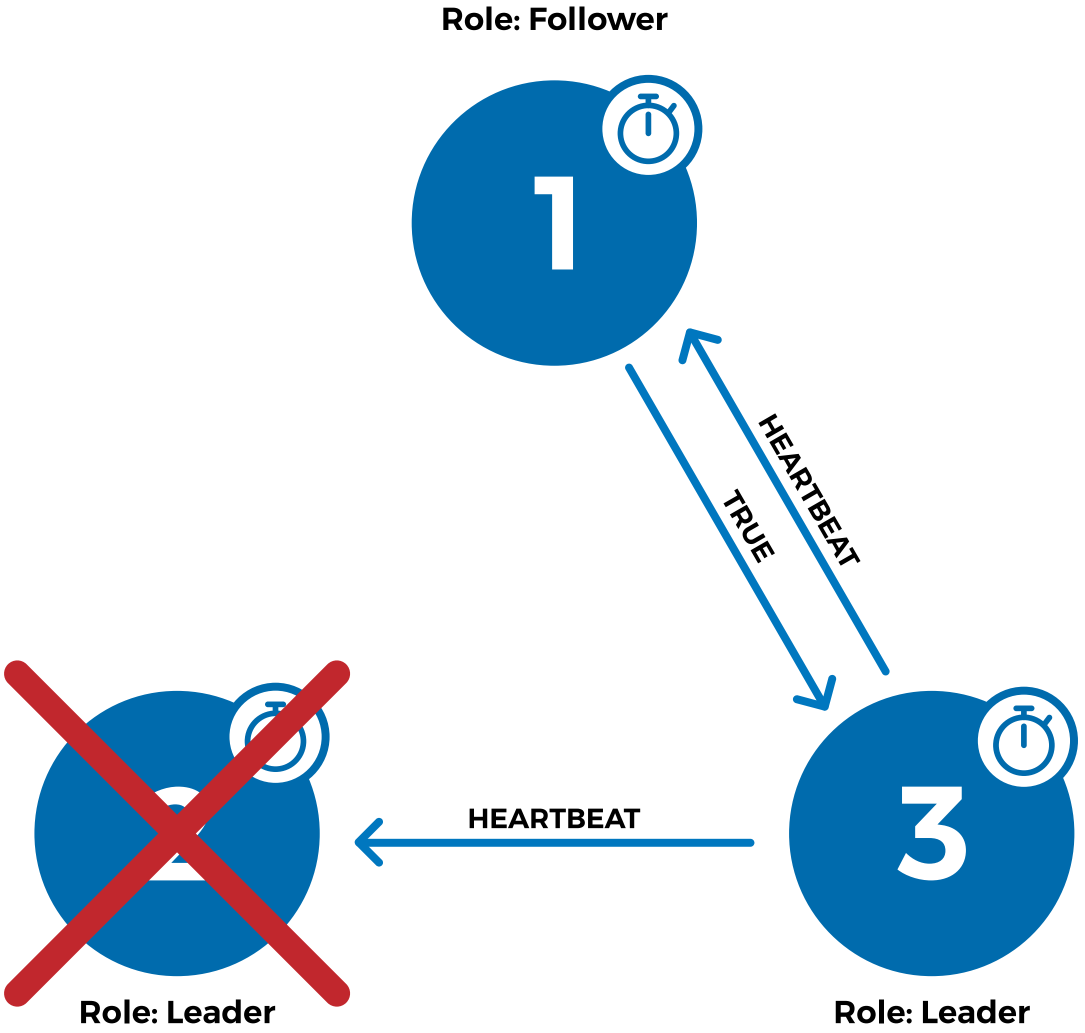


Role: Follower
Voted for: 3

Role: Leader

Role: Candidate
Votes: 2

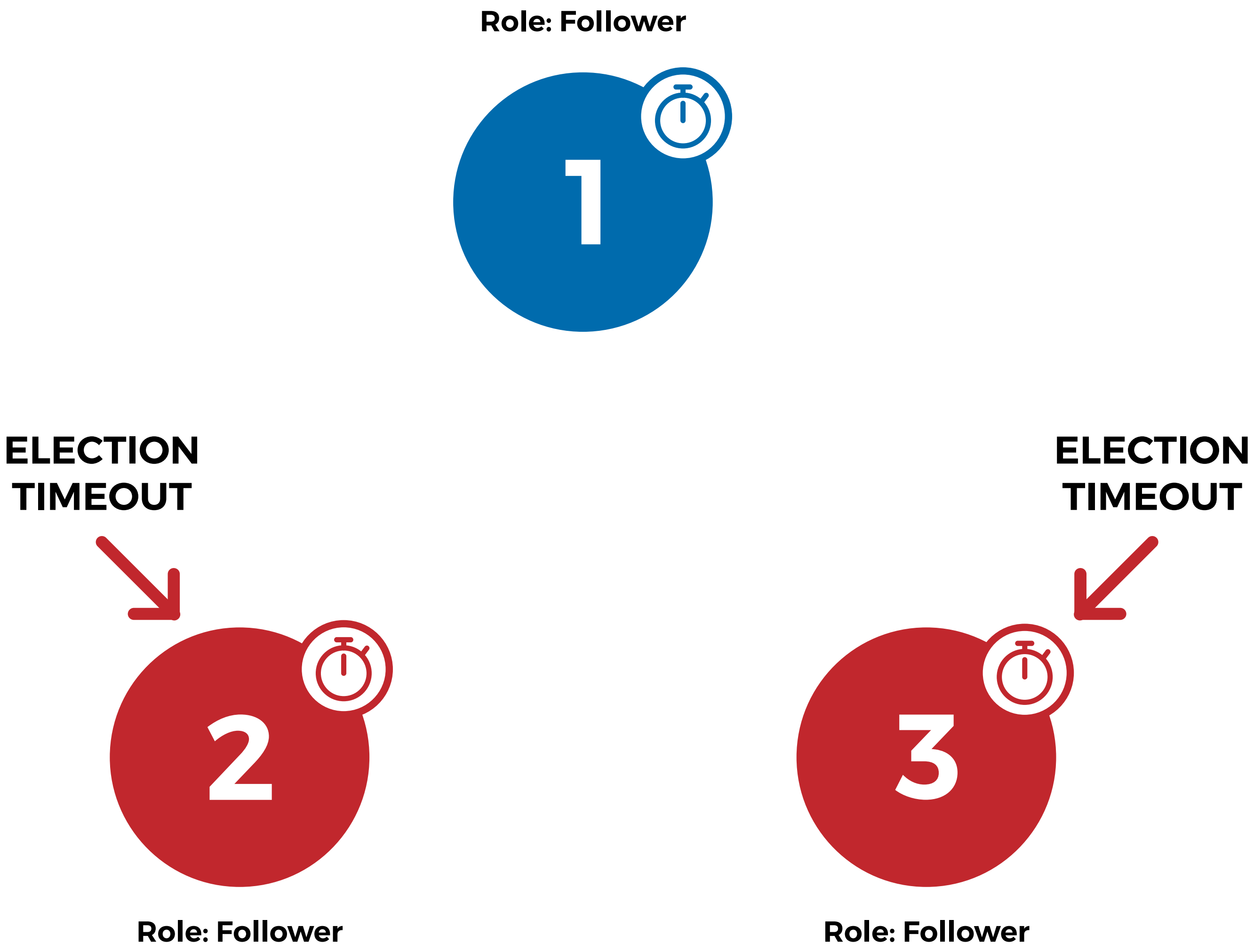
Leader node failure



Corner cases

Race condition

Election race condition



Role: Follower

1

ELECTION TIMEOUT

ELECTION TIMEOUT

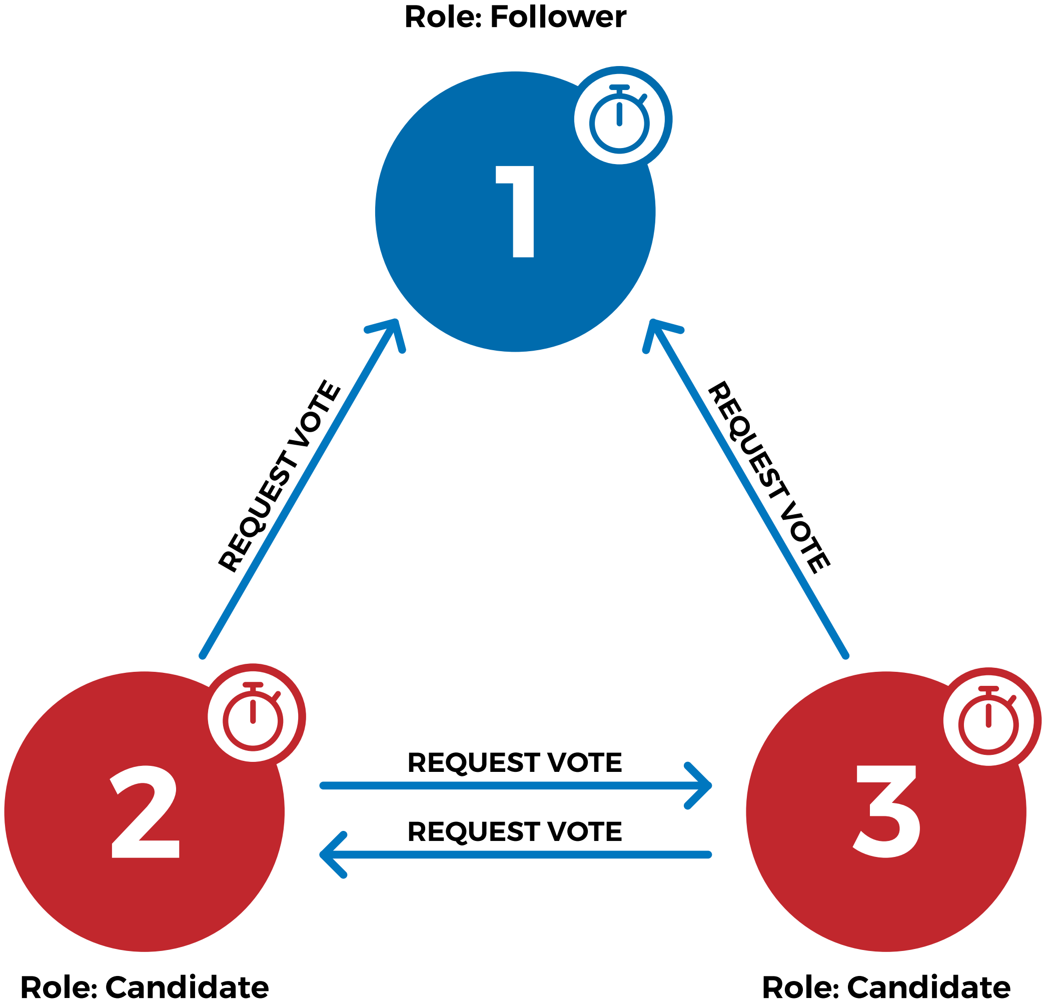
2

3

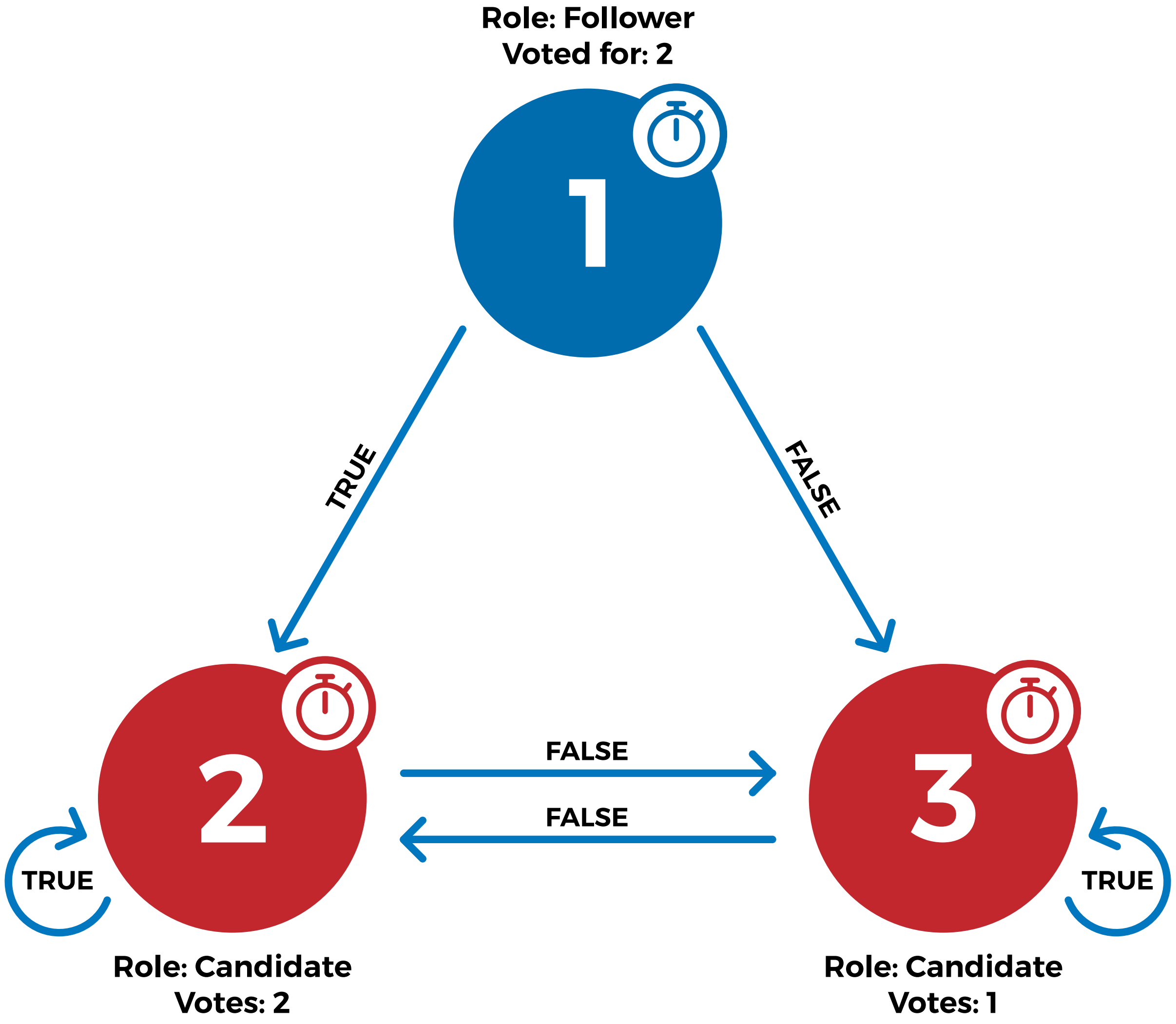
Role: Follower

Role: Follower

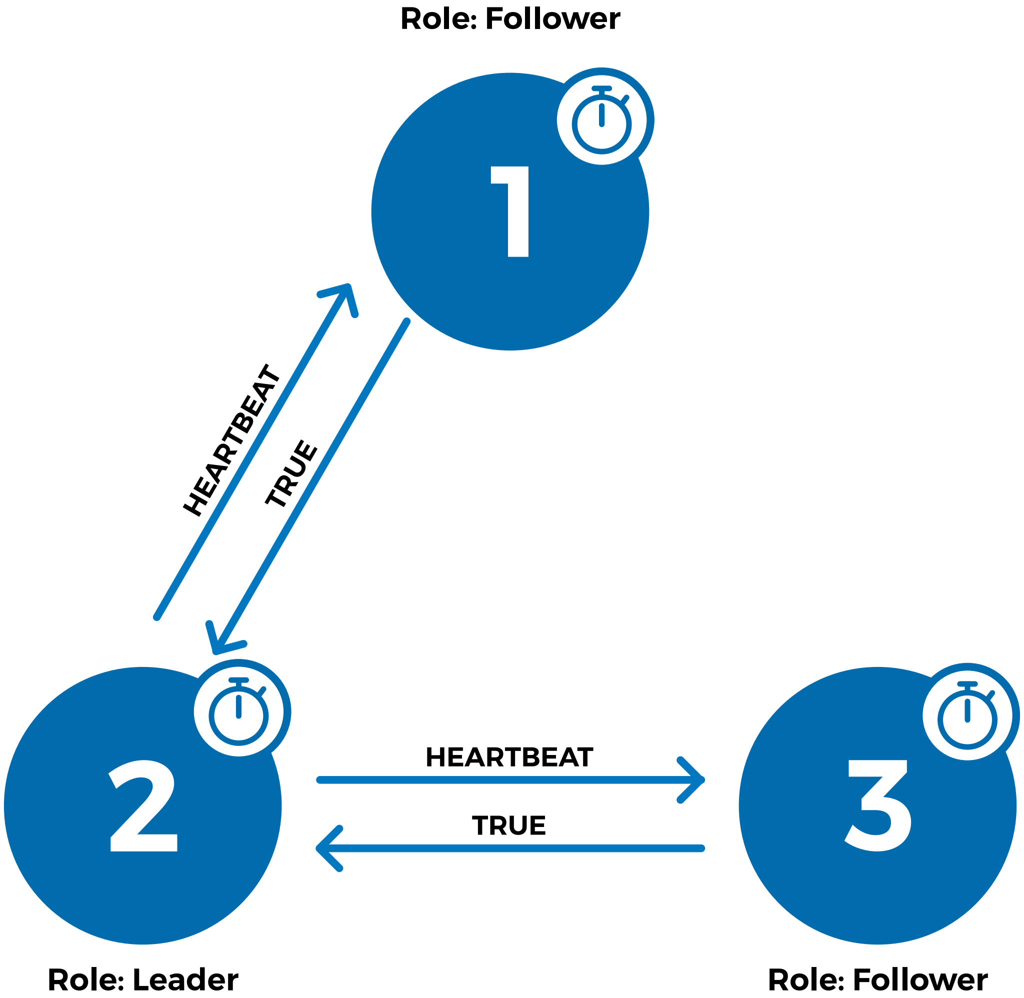
Election race condition



Election race condition



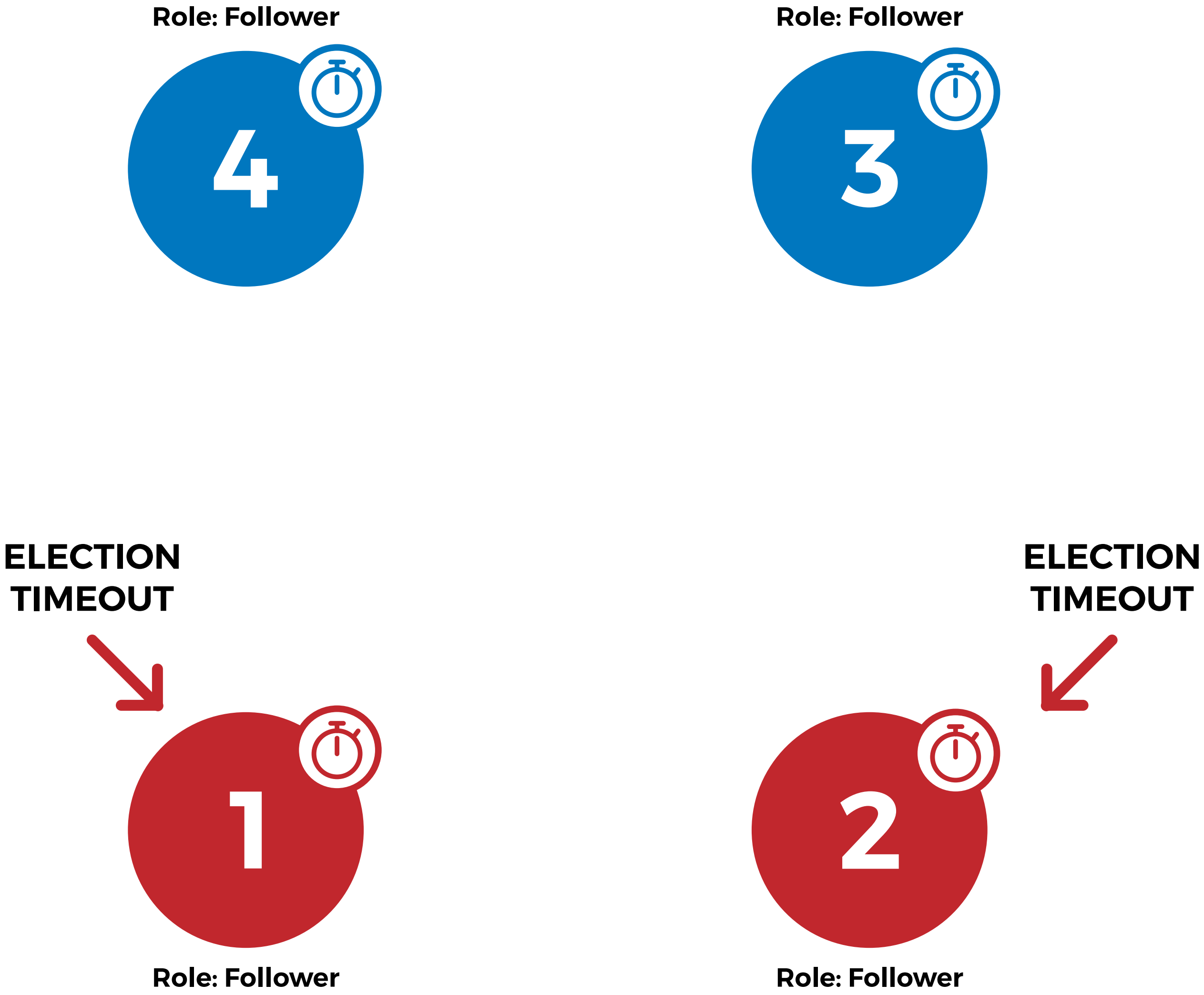
Election race condition



Corner cases

Split votes

Split votes



Role: Follower

4

Role: Follower

3

ELECTION TIMEOUT

ELECTION TIMEOUT

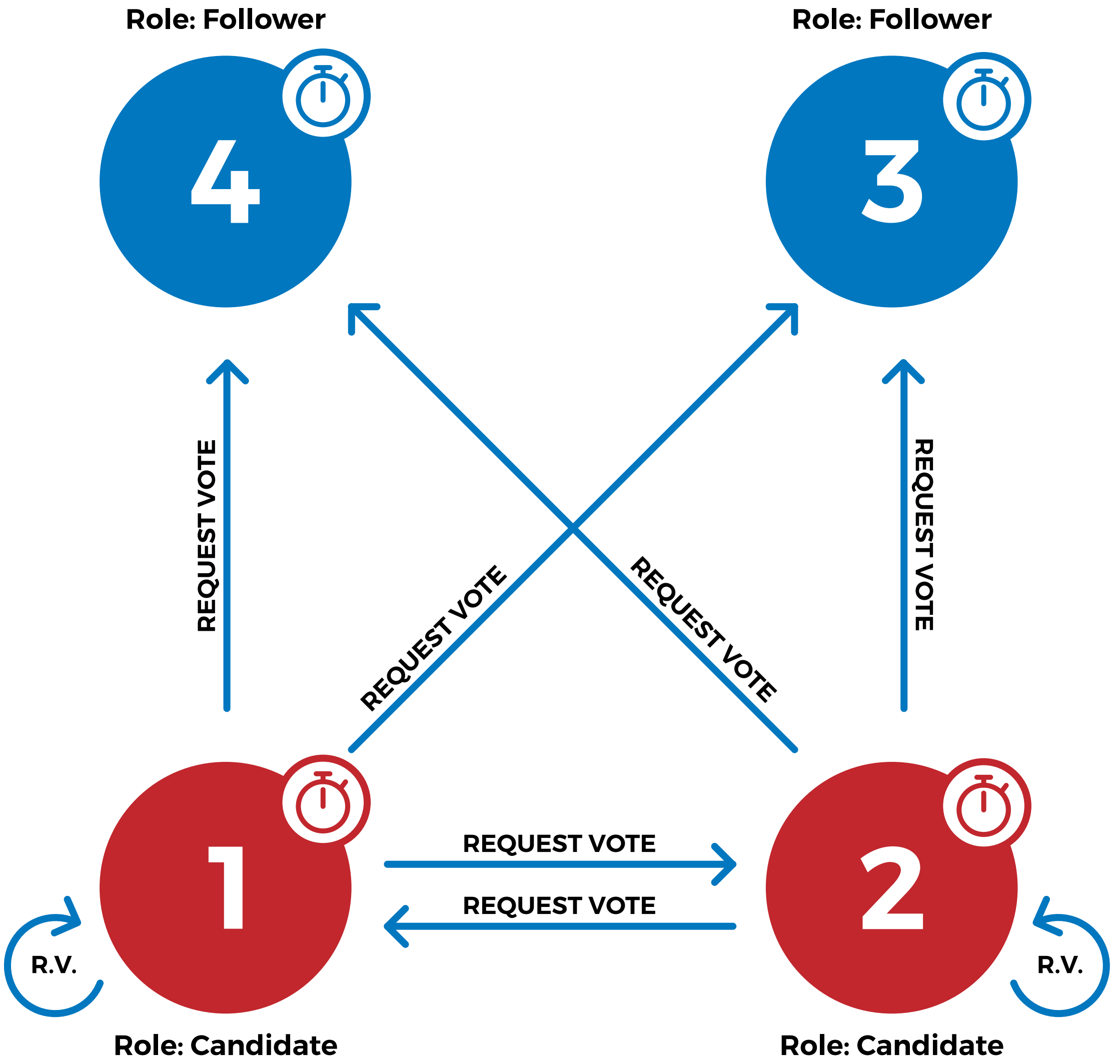
1

Role: Follower

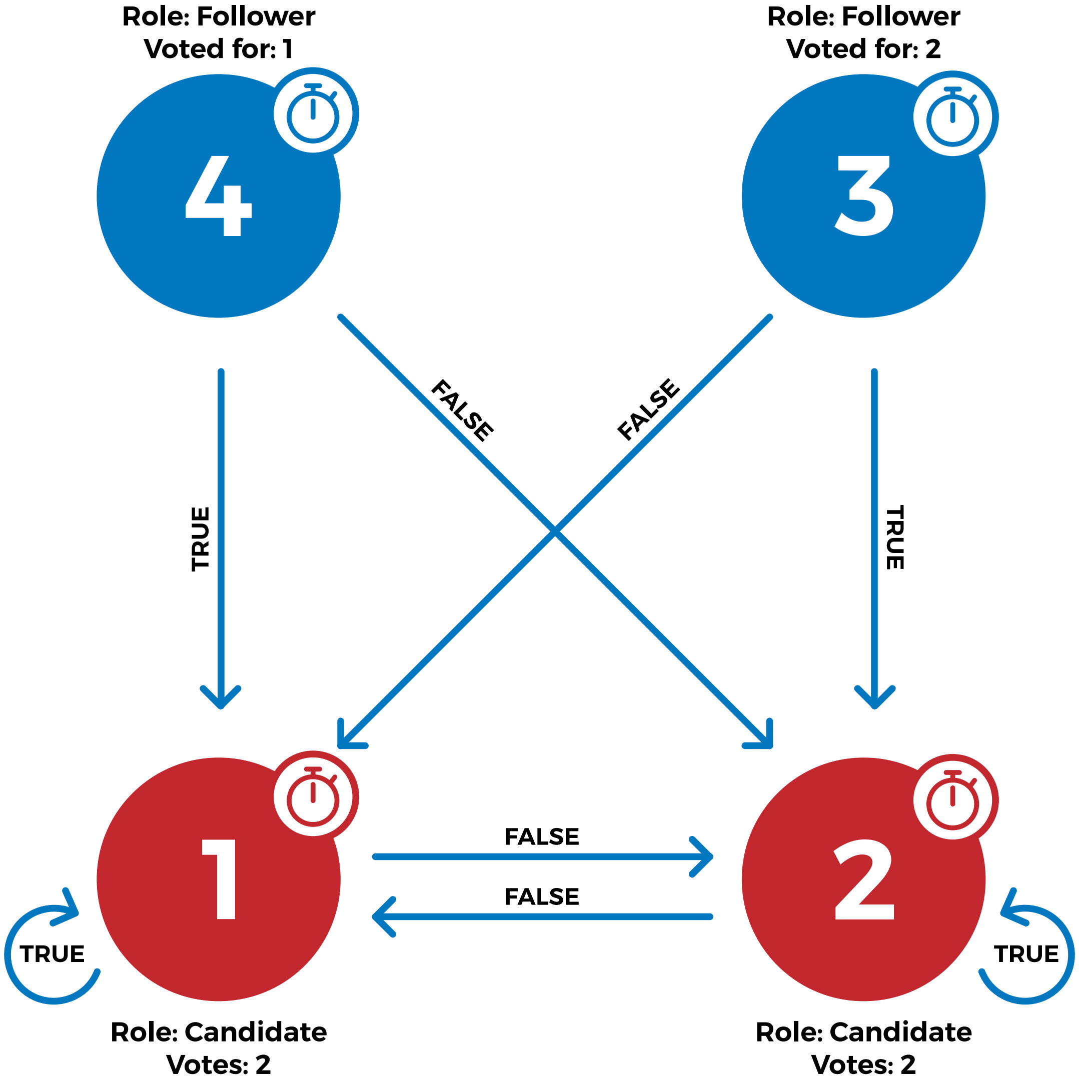
2

Role: Follower

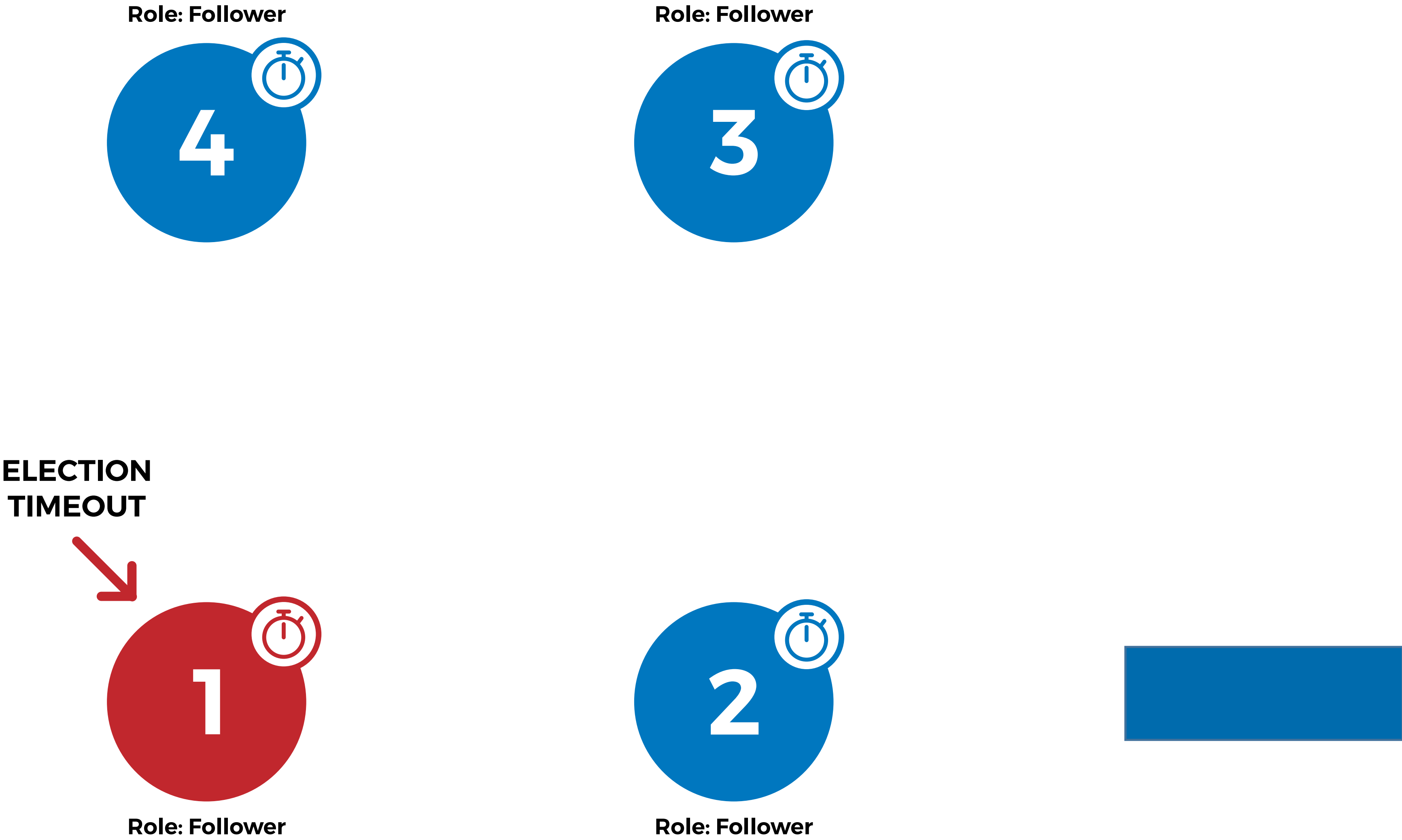
Split votes


Role: Follower
Role: Follower
4
3
REQUEST VOTE
REQUEST VOTE
REQUEST VOTE
REQUEST VOTE
1
2
REQUEST VOTE
REQUEST VOTE
R.V.
R.V.
Role: Candidate
Role: Candidate

Split votes


Role: Follower
Voted for: 1
4
Role: Follower
Voted for: 2
3
FALSE
FALSE
TRUE
TRUE
FALSE
FALSE
TRUE
1
2
TRUE
Role: Candidate
Votes: 2
Role: Candidate
Votes: 2

Split votes



Role: Follower

4

Role: Follower

3

ELECTION TIMEOUT

1

Role: Follower

2

Role: Follower