Let's talk gossip

am I qualified?

Rad(oslaw) Gruchalski

distributed computing
tools
programming languages

what is gossip

m2m communication protocol

epidemic, biased

dissemination protocols

events
background data

anti-entropy protocols

repair replicated data

aggregate protocols

calculate system-wide values

how does it work?

seeds and members

different role, no structural difference, membership

electing seeds

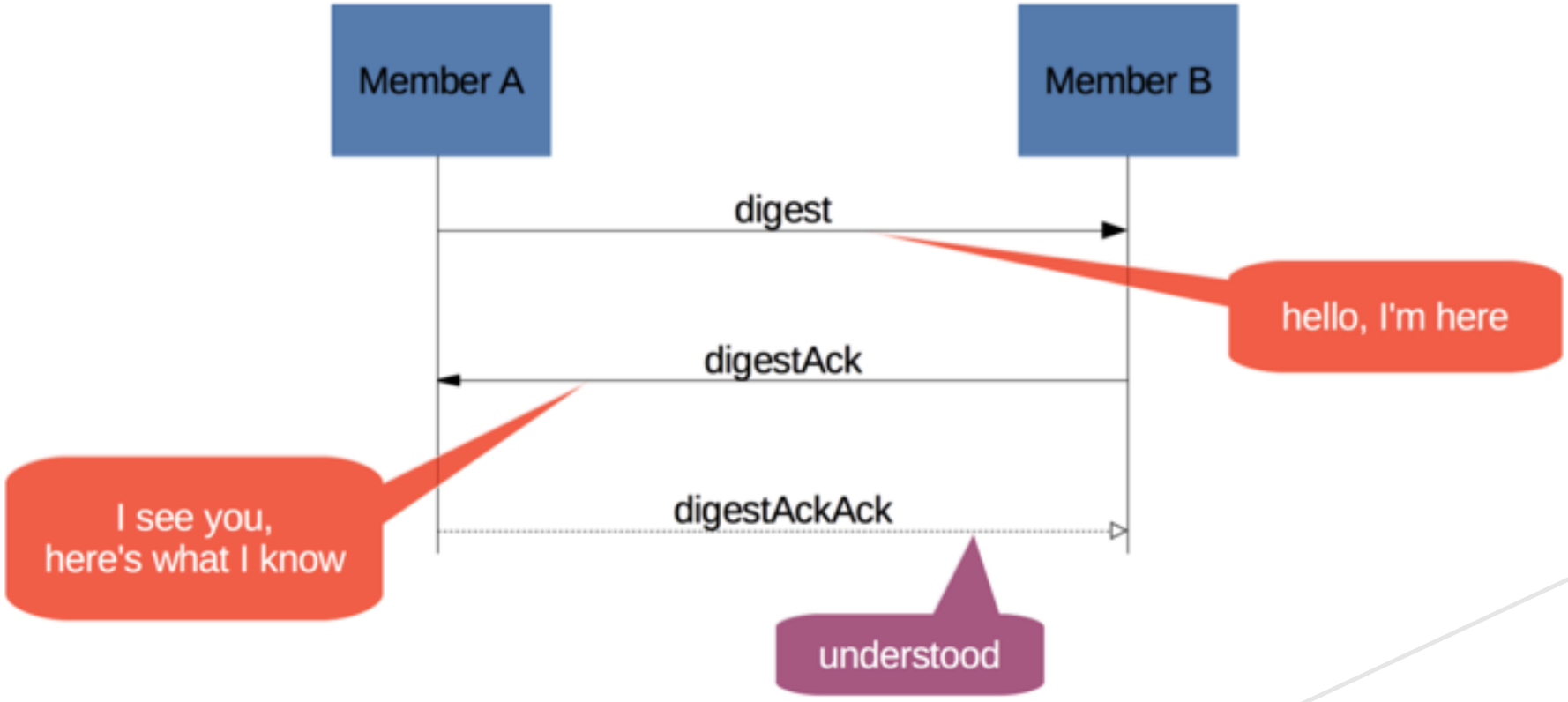
DNS, configuration management

UDP, usually

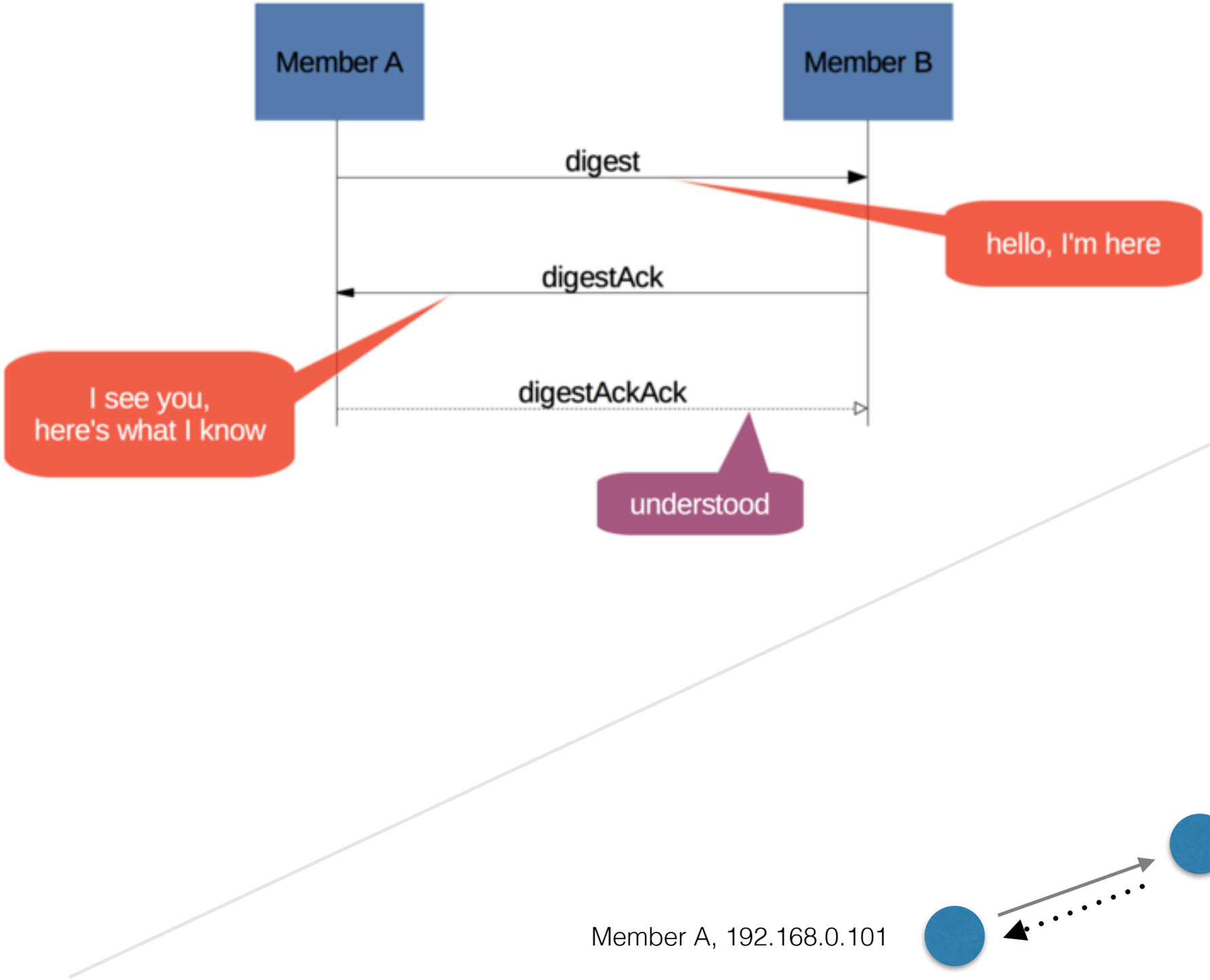
casual ordering
latency acceptable

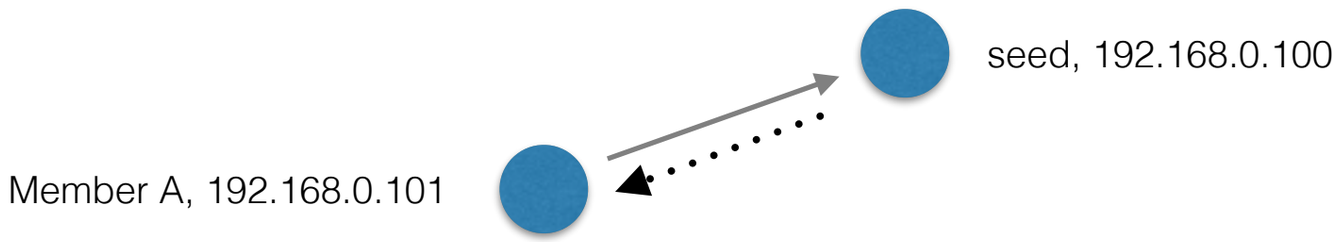
gossip communication

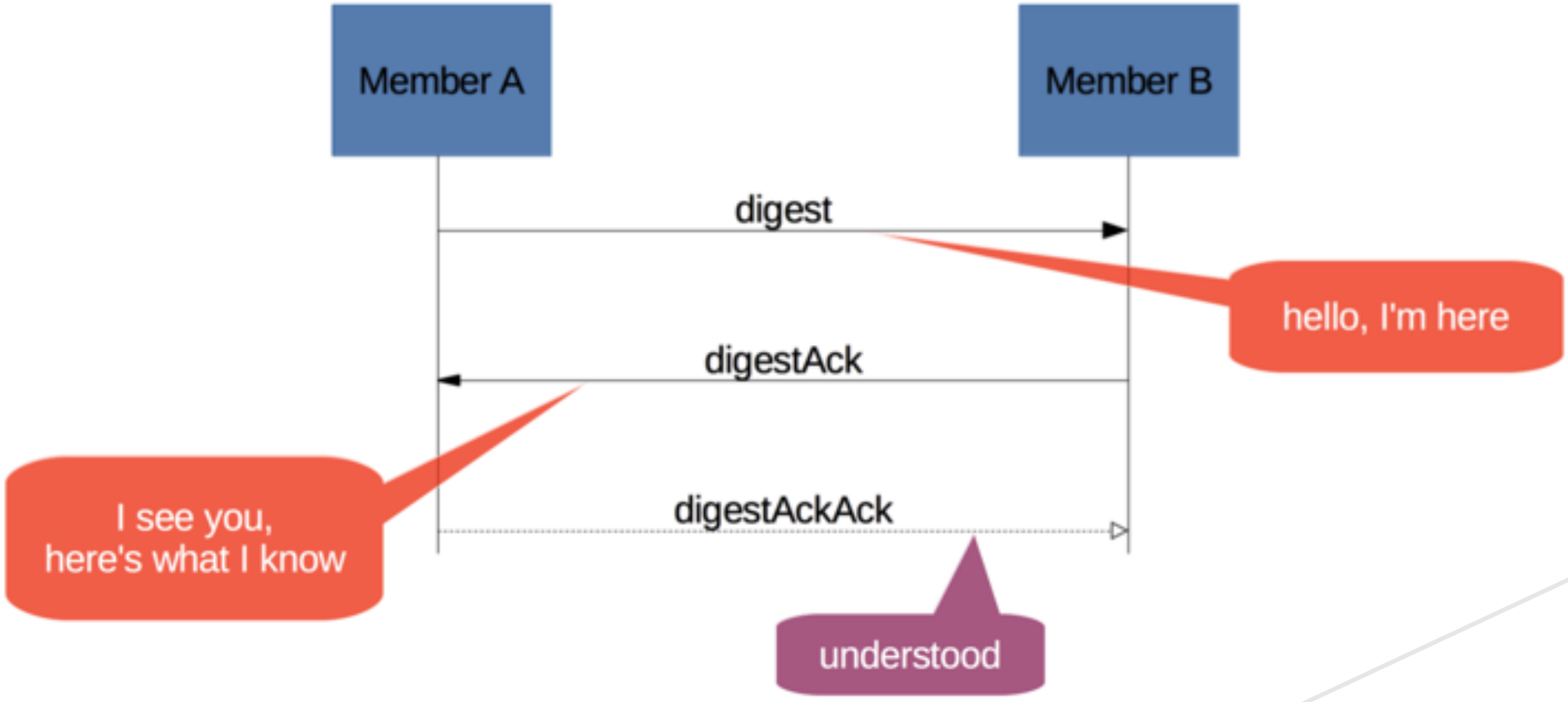
digest, digestAck(, digestAckAck)
membership
gossip round

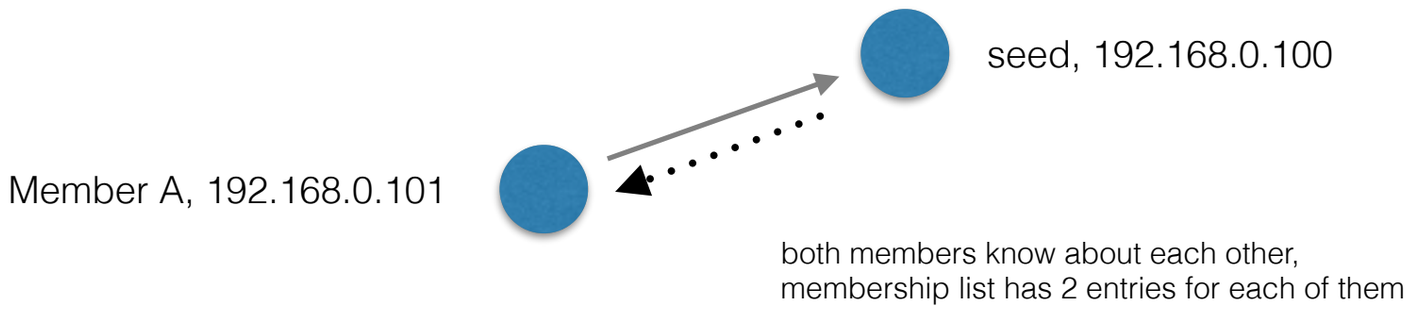


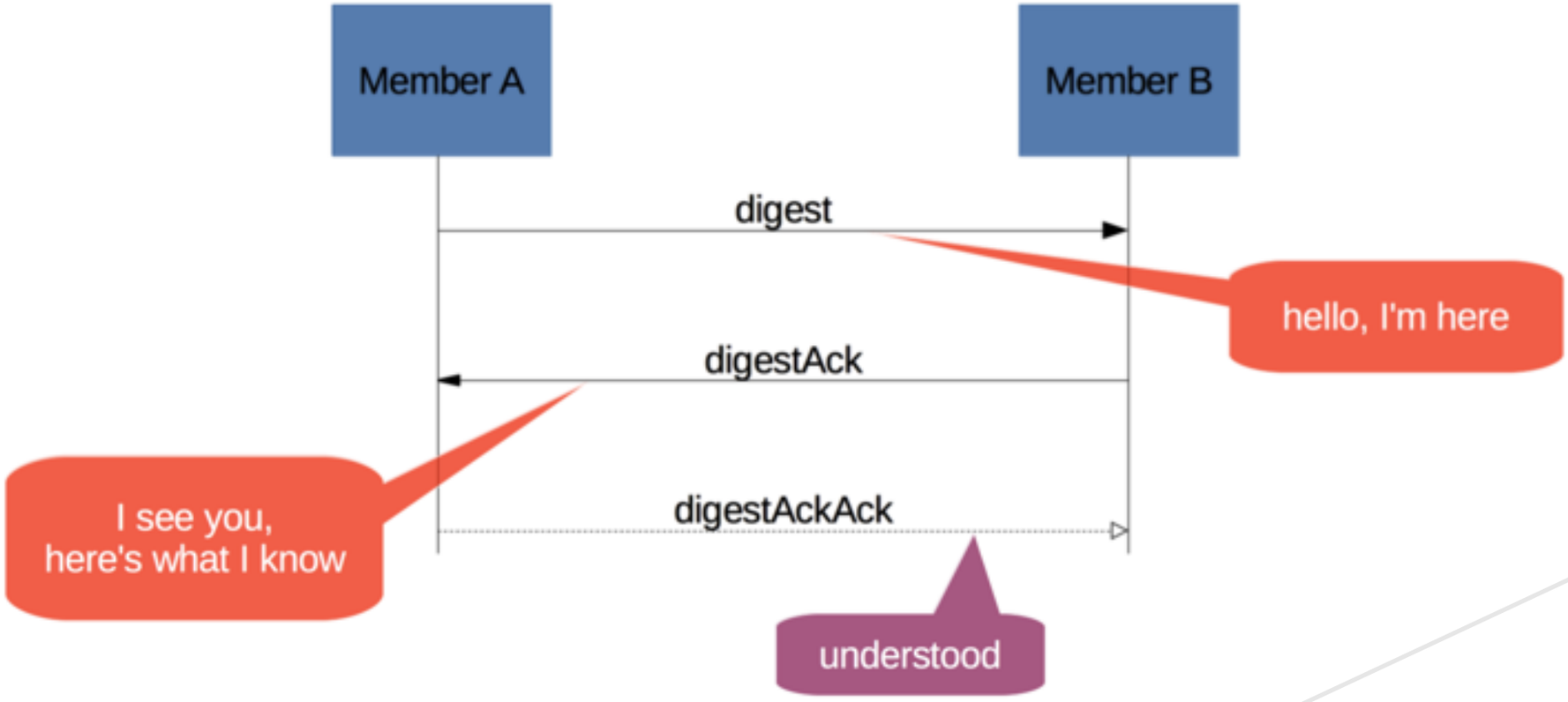
seed, 192.168.0.100

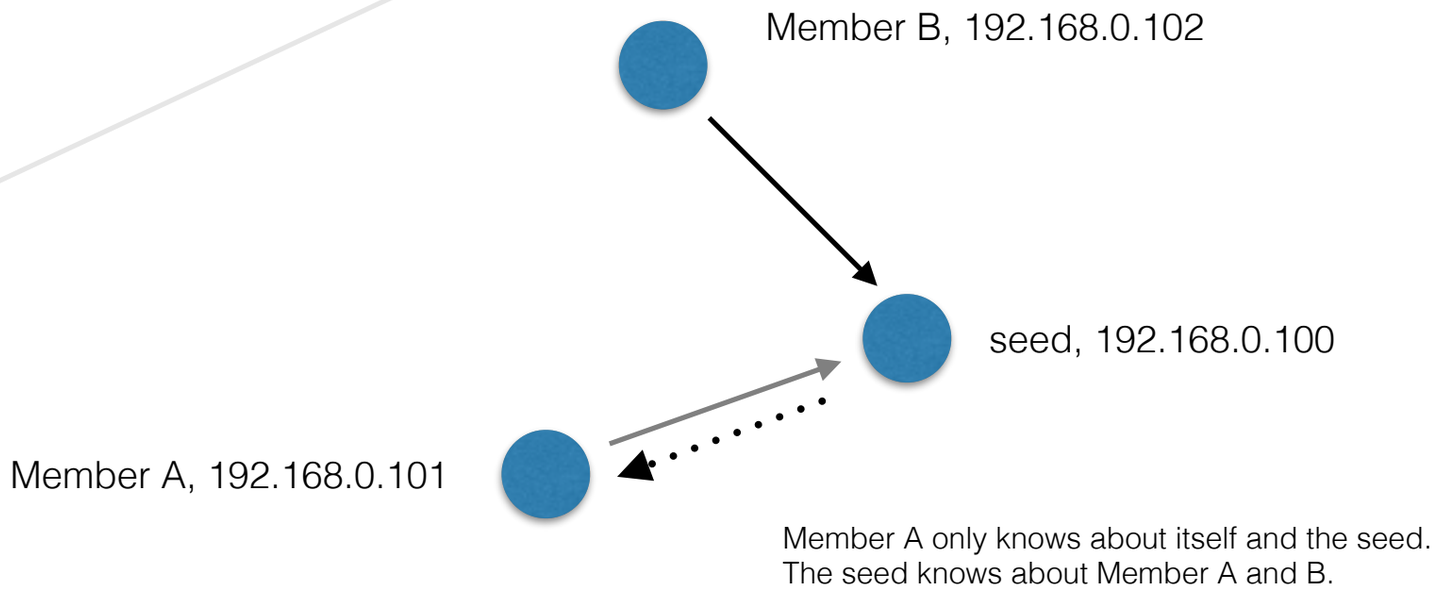

Member A
Member B
digest
hello, I'm here
digestAck
I see you,
here's what I know
digestAckAck
understood

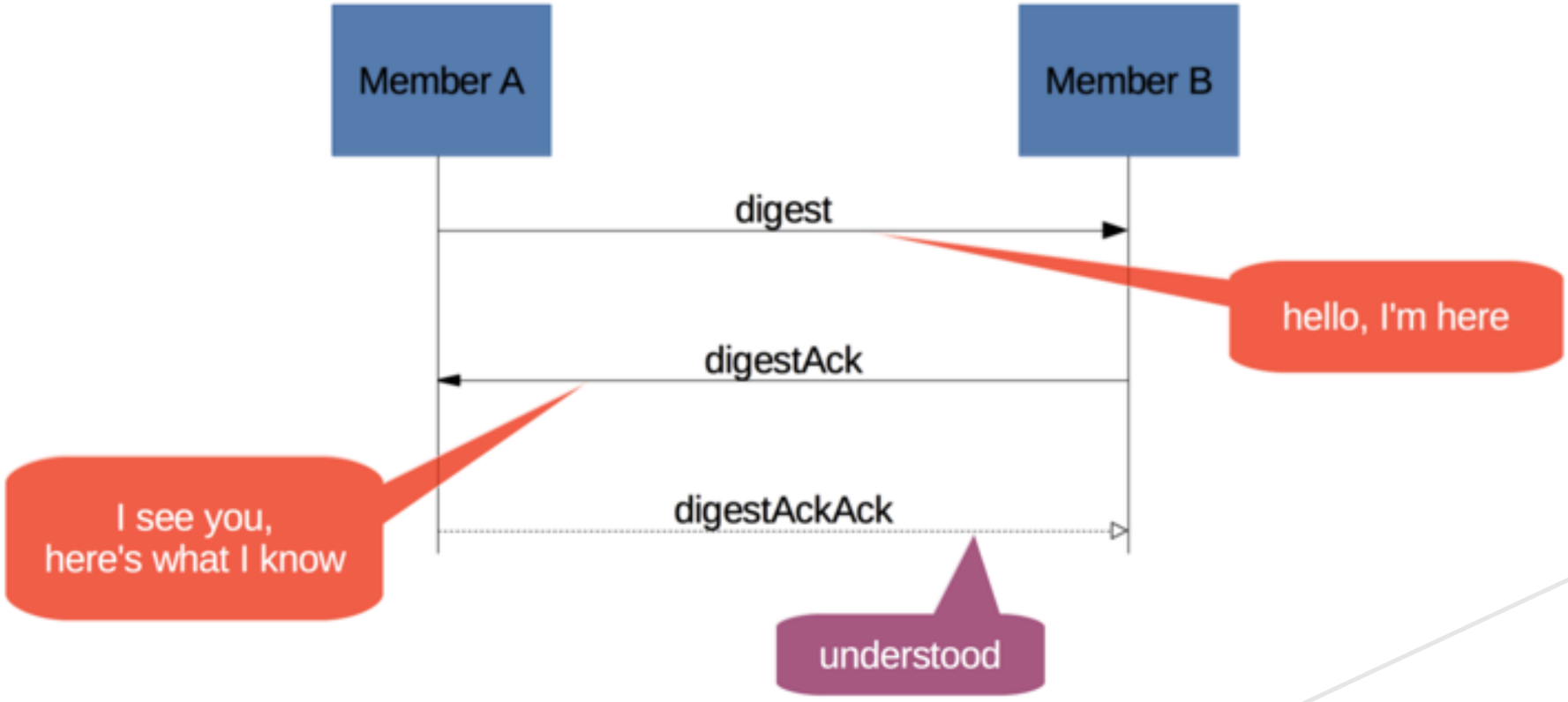

seed, 192.168.0.100
Member A, 192.168.0.101

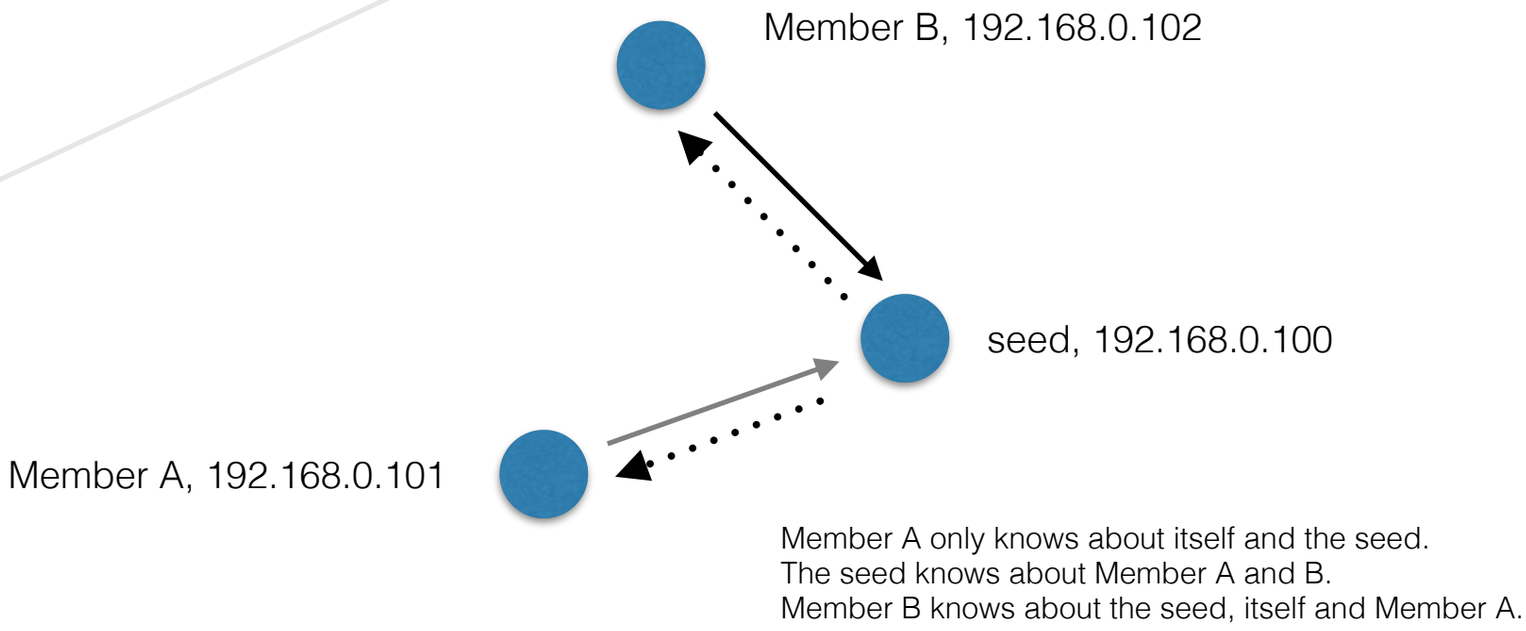

Member A
Member B
digest
hello, I'm here
digestAck
I see you,
here's what I know
digestAckAck
understood

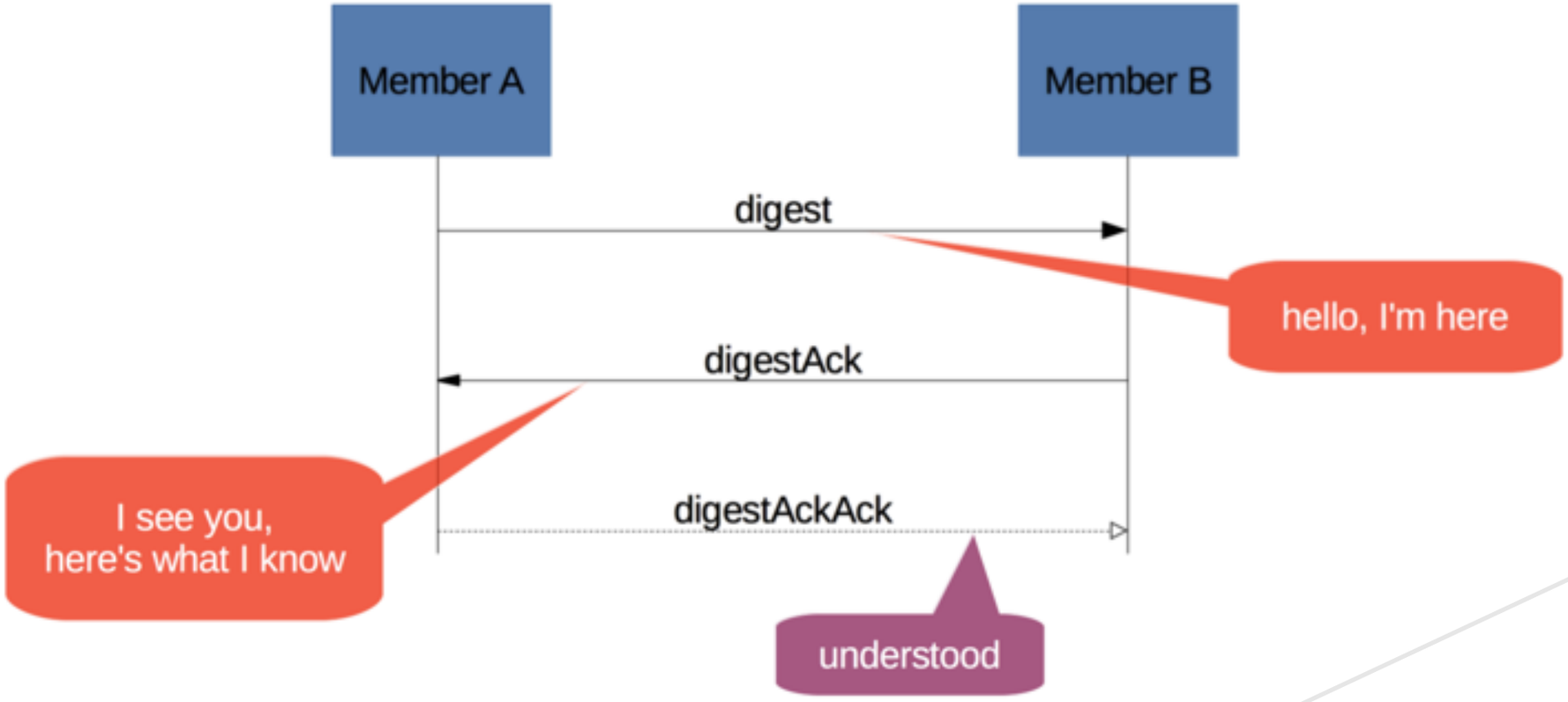

seed, 192.168.0.100
Member A, 192.168.0.101
both members know about each other,
membership list has 2 entries for each of them

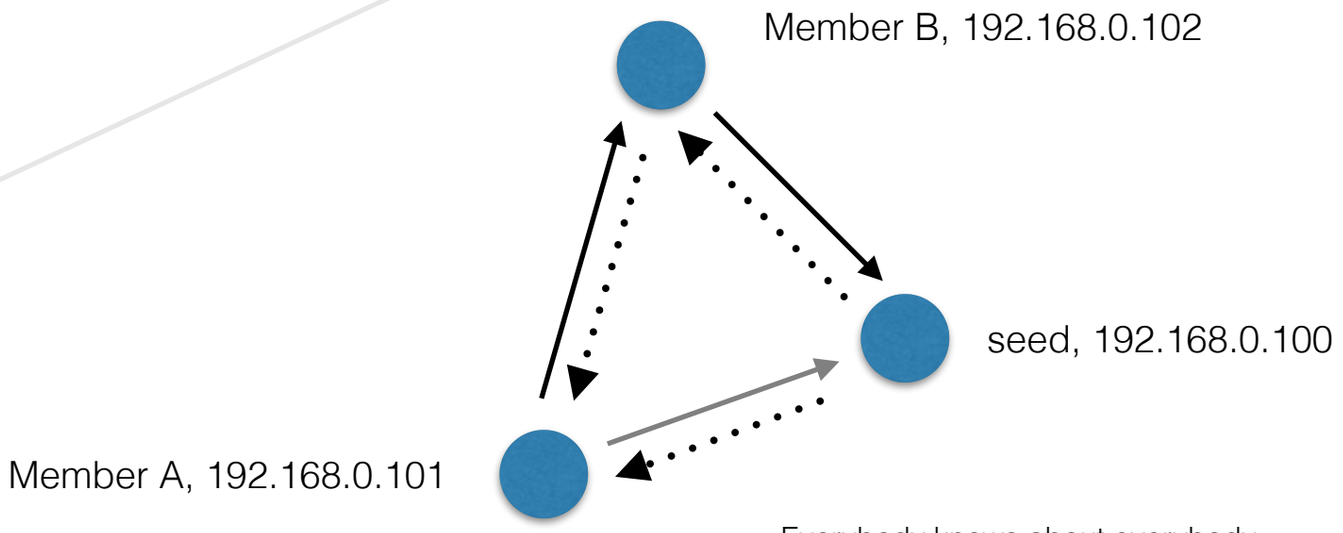

Member A
Member B
digest
hello, I'm here
digestAck
I see you,
here's what I know
digestAckAck
understood


Member B, 192.168.0.102
seed, 192.168.0.100
Member A, 192.168.0.101
Member A only knows about itself and the seed.
The seed knows about Member A and B.

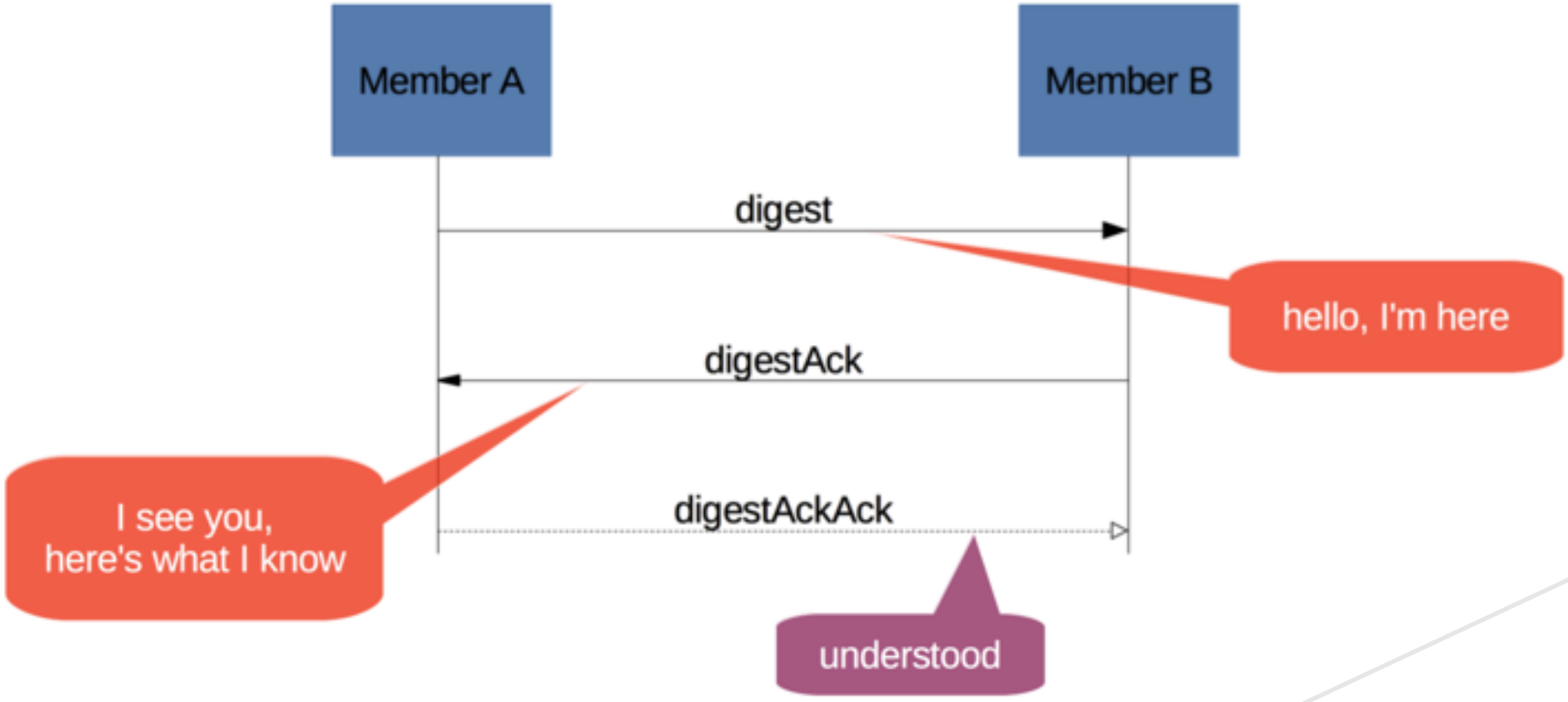

Member A
Member B
digest
hello, I'm here
digestAck
I see you,
here's what I know
digestAckAck
understood

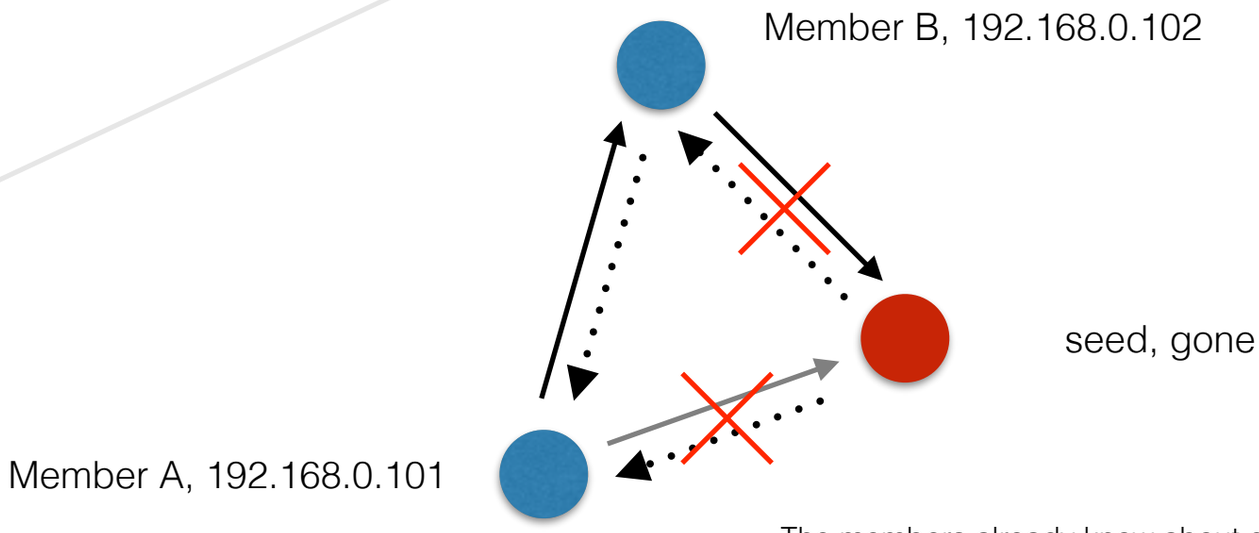

Member B, 192.168.0.102
seed, 192.168.0.100
Member A, 192.168.0.101
Member A only knows about itself and the seed.
The seed knows about Member A and B.
Member B knows about the seed, itself and Member A.





Everybody knows about everybody.
But what if the seed blows up?





The members already know about each other.
They are unaffected and can still communicate.

No new members can join!

security

no formal specification
implementation ideas in an example

real-world products

- Apache Cassandra
- Basho Riak
- Netflix Dynomite

gossiperl

gossip daemon written in Erlang
content indenpendent
Apache Thrift

reasoning

gossip a service for the host, not a library

features

language agnostic, multiple overlays, double security layer, managed via REST API, multicast overlays, IPv6 support, simple pub/sub, multi-datacenter support *****

pub/sub demo

clients

Erlang, Java, Scala with Akka, Ruby, JavaScript with Chrome support, C#, more to come

use cases

service discovery
distributed state

little demo

gossiperl on RPi with multicast

the code

- https://github.com/gossiperl/gossiperl
- https://gossiperl/gossiperl
- http://gossiperl.com

questions?

mentioned products

- Apache Cassandra: https://github.com/apache/cassandra
- Basho Riak: https://github.com/basho/riak_core
- Netflix Dynomite: https://github.com/netflix/dynomite

(some) relevant resources

- Exploiting Gossip for Self-Management in Scalable Event Notification Systems:
  Ken Birman, Anne-Marie Kermarrec, Krzystof Ostrowski, Marin Bertier, Danny Dolev, Robbert Van Renesse; Cornell University, Ithaca; INRIA/ IRISA and IRISA/INSA, Rennes; Hebrew University, Jerusalem
- A Gossip-Style Failure Detection Service:
  Robbert van Renesse, Yaron Minsky, and Mark Hayden*; Dept. of Computer Science, Cornell University; 4118 Upson Hall, Ithaca, NY 14853
- A Middleware for Gossip Protocols:
  Michael Chow, Robbert van Renesse; Cornell University
- Astrolabe: A Robust and Scalable Technology For Distributed System Monitoring, Management, and Data Mining:
  Robert van Renesse, Kenneth P. Birman, and Werner Vogels; Department of Computer Science; Cornell University, Ithaca, NY 14853