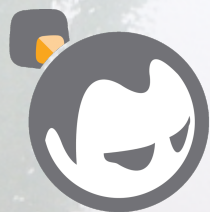


Things break.

 **riaik bends.**



Justin Sheehy
justin@basho.com



Sheehy

Sheehy

Perfection is Unattainable

A system cannot perform as well during a storm of component failure as it can on a sunny day.

Know How You Degrade

Plan it and understand it before your users do.

You might prevent whole system failure if you're lucky and good, but what happens during **partial failure?**

Know How You Degrade

Plan it and understand it before your users do.

You think you know
which parts will break.

Know How You Degrade

Plan it and understand it before your users do.

You think you know
which parts will break.

You are wrong.



Harvest and Yield

harvest: a fraction
data available / complete data

yield: a probability
queries completed / q's requested

in tension with each other:
(harvest * yield) ~ constant

goal: failures cause known linear reduction to one of these

Harvest and Yield

traditional design demands 100% **harvest**
but success of modern applications is
often measured in **yield**

plan ahead, know when you care!

Perfection is Unattainable

A system cannot perform as well during a storm of component failure as it can on a sunny day.

Perfection is Unattainable

failures will happen.

A system cannot perform as well during a storm of component failure as it can on a sunny day.

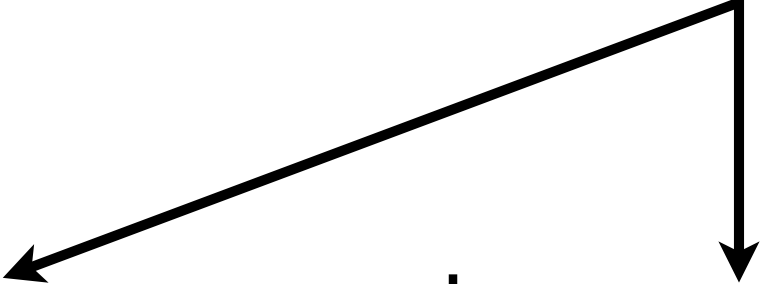
Resilience is Attainable

Assume that **failures will happen.**

Designing whole systems and components with individual failures in mind is a plan for **predictable success.**

Resilience is Attainable

Layered, multi-scale resilience is key!



Designing **whole systems** and **components** with individual failures in mind is a plan for predictable success.

Component Failure: reboot of live database

Worst case: whole DB corrupted!

Typical mitigation: write-ahead logging for repair

Component Failure: reboot of live database

Worst case: whole DB corrupted!

Typical mitigation: write-ahead logging for repair

Drawbacks: logging adds I/O, repair can be slow

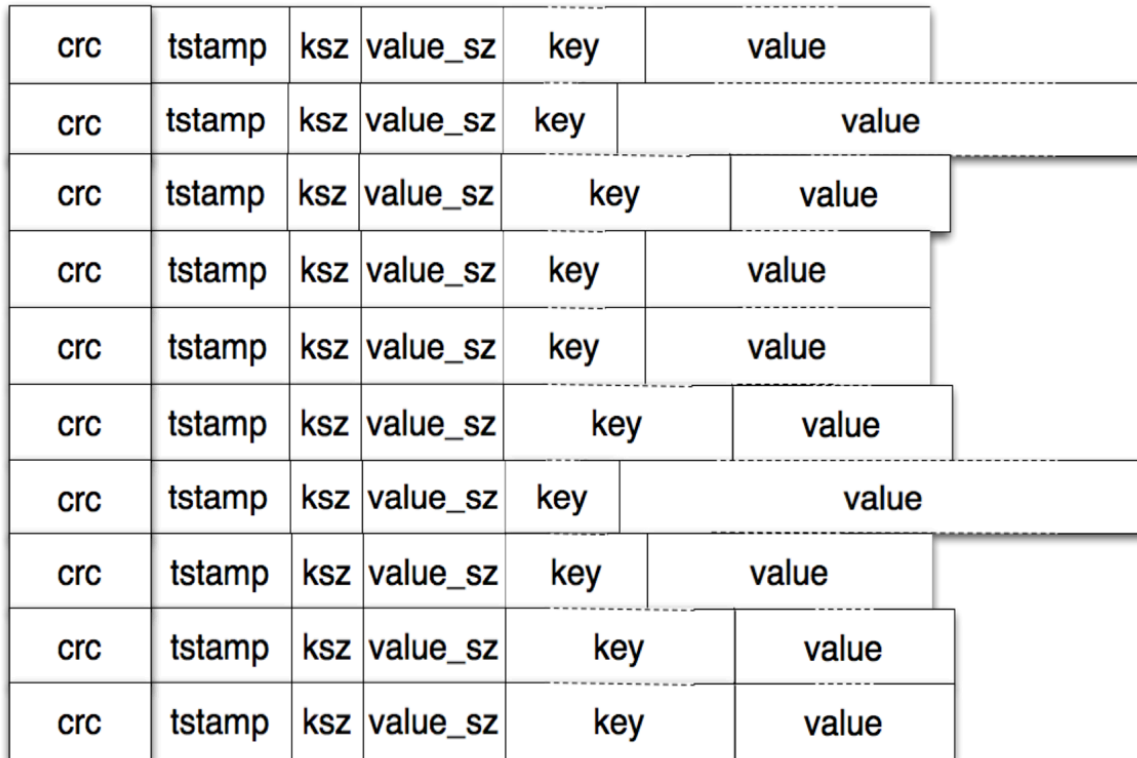
Component Failure: reboot of live database

Alternative: append-only main storage

"log-structured" databases

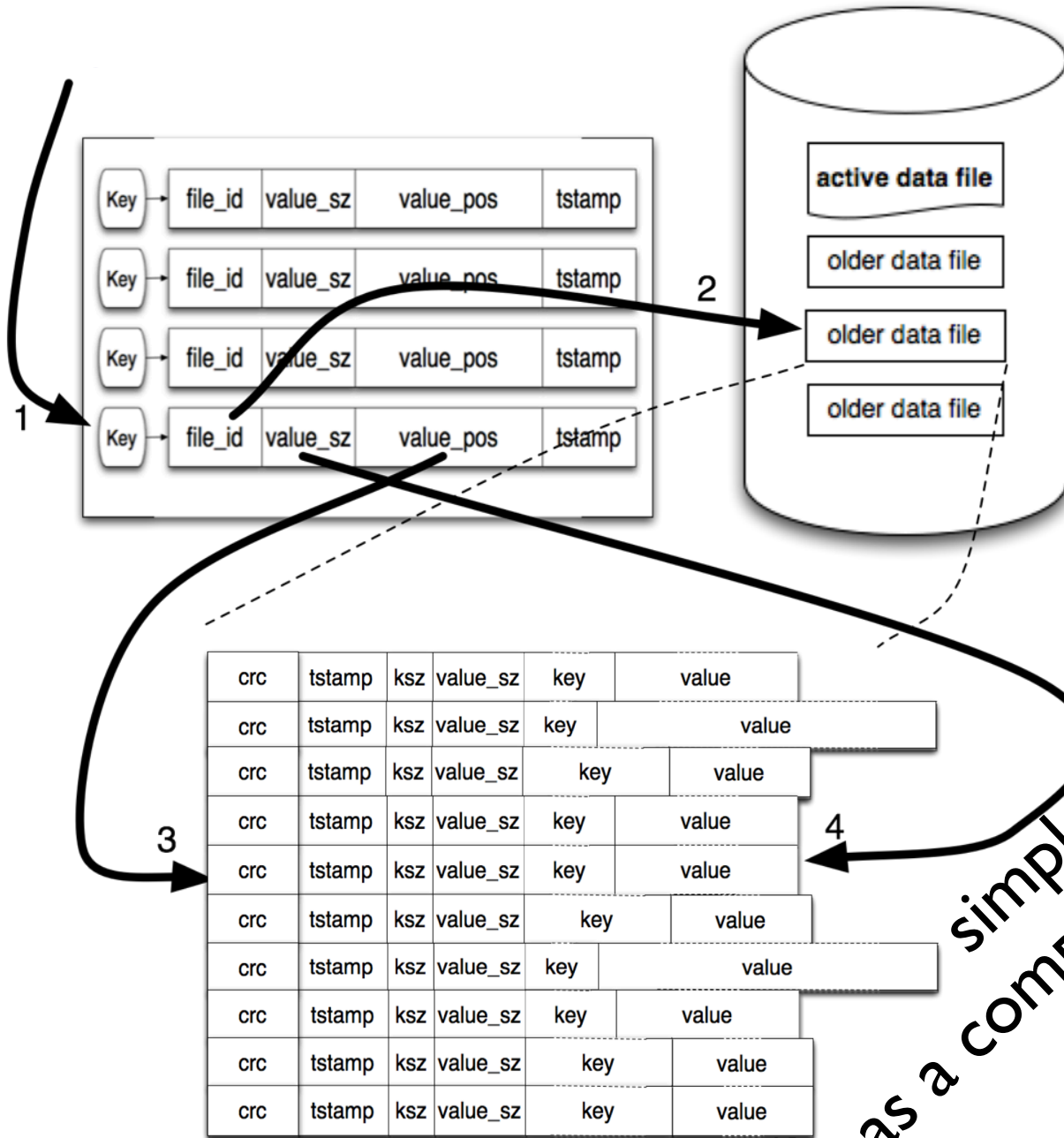
Example: **bitcask**

Bitcask



simple append-only file format

Bitcask



as a simple append-only file format
as a component of something bigger

Component Failure: reboot during record write

What about a half-written write?

Two problems: detection, minimization.

Component Failure: reboot during record write

What about a half-written write?

Two problems: **detection**, minimization.

minimum-length check, CRC-check per record

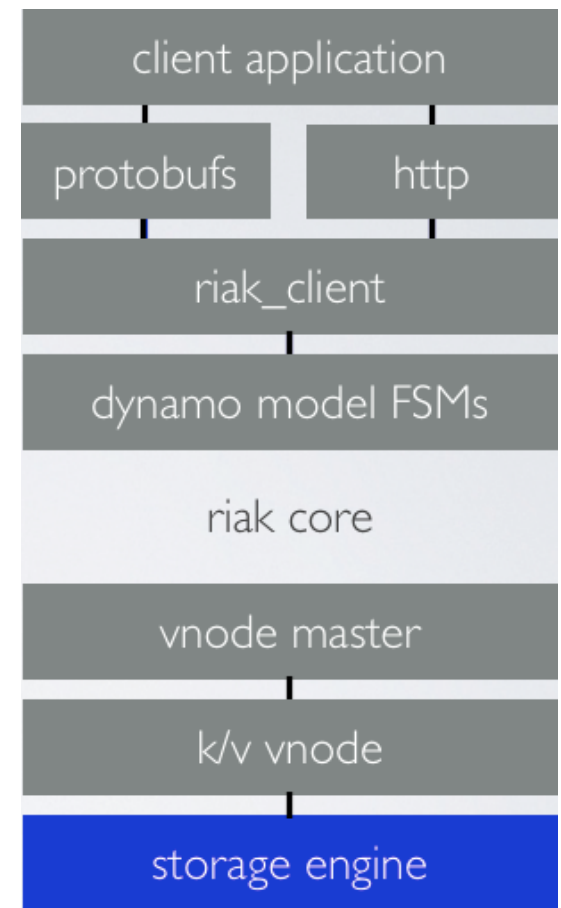
Component Failure: reboot during record write

What about a half-written write?

Two problems: detection, [minimization](#).

invalidate only the end-failed record, not the file

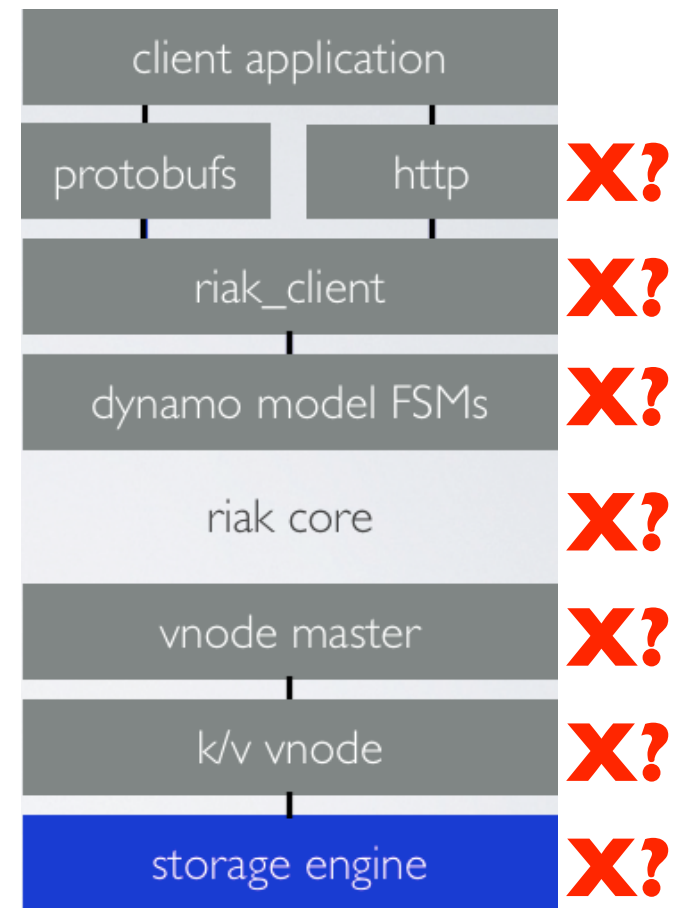
Zoom Out: Bitcask is one part of Riak



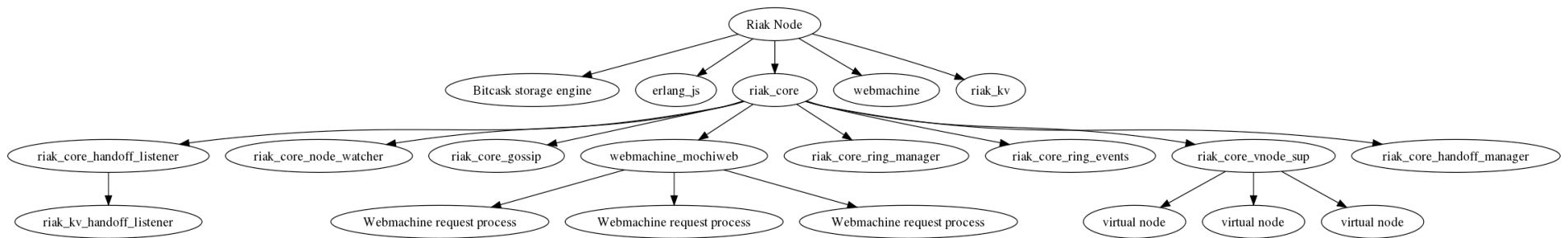
Component Failure: internal subsystem crash

Bugs can lurk anywhere.
Unpredictability, eek.

Typical mitigation:
complex exception-management

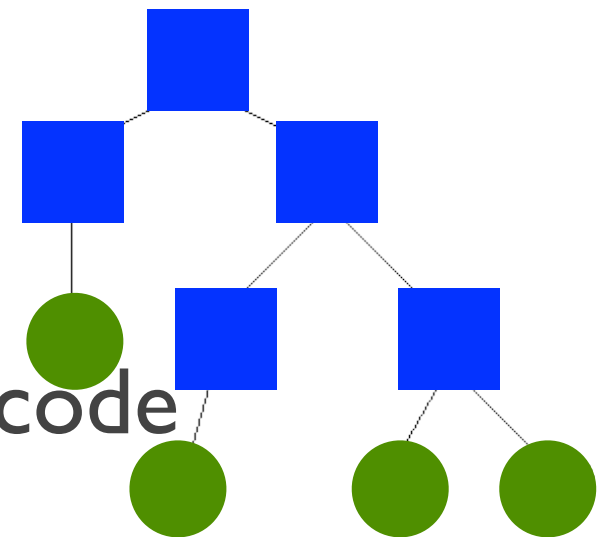


Component Failure: internal subsystem crash



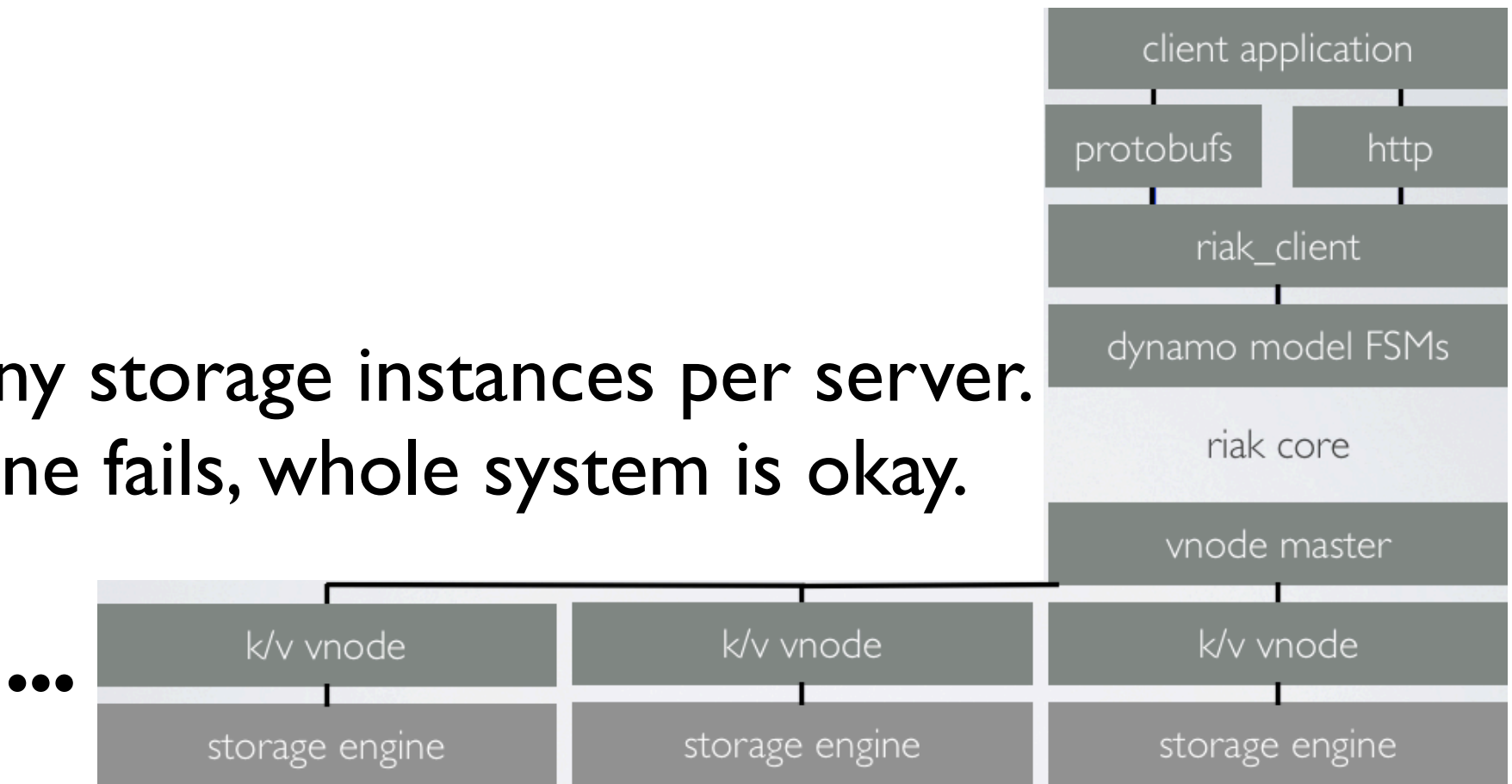
Stronger mitigation:
supervision trees and "let it crash"

Added bonus: simpler and clearer code



Zoom Out: Virtual Nodes

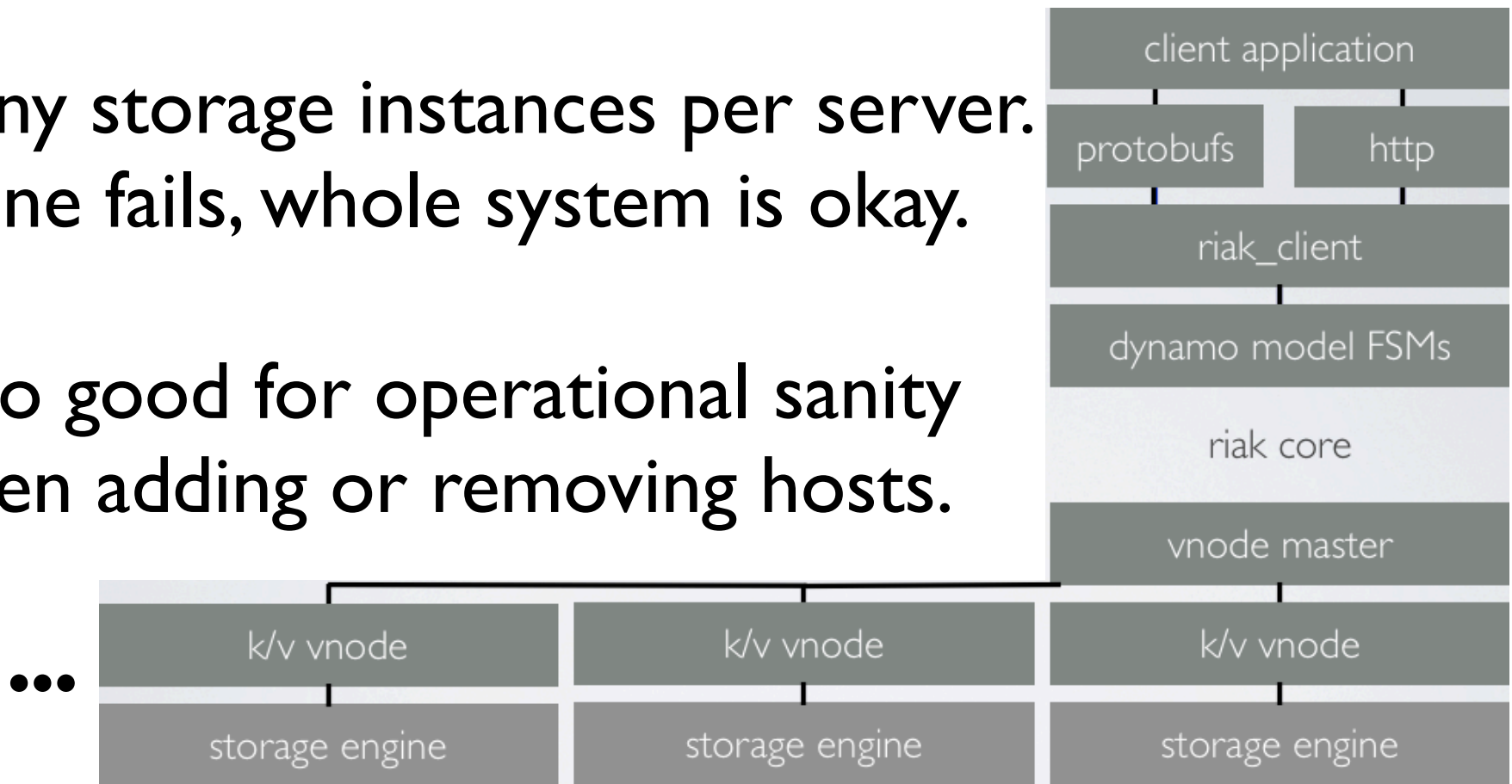
Many storage instances per server.
If one fails, whole system is okay.



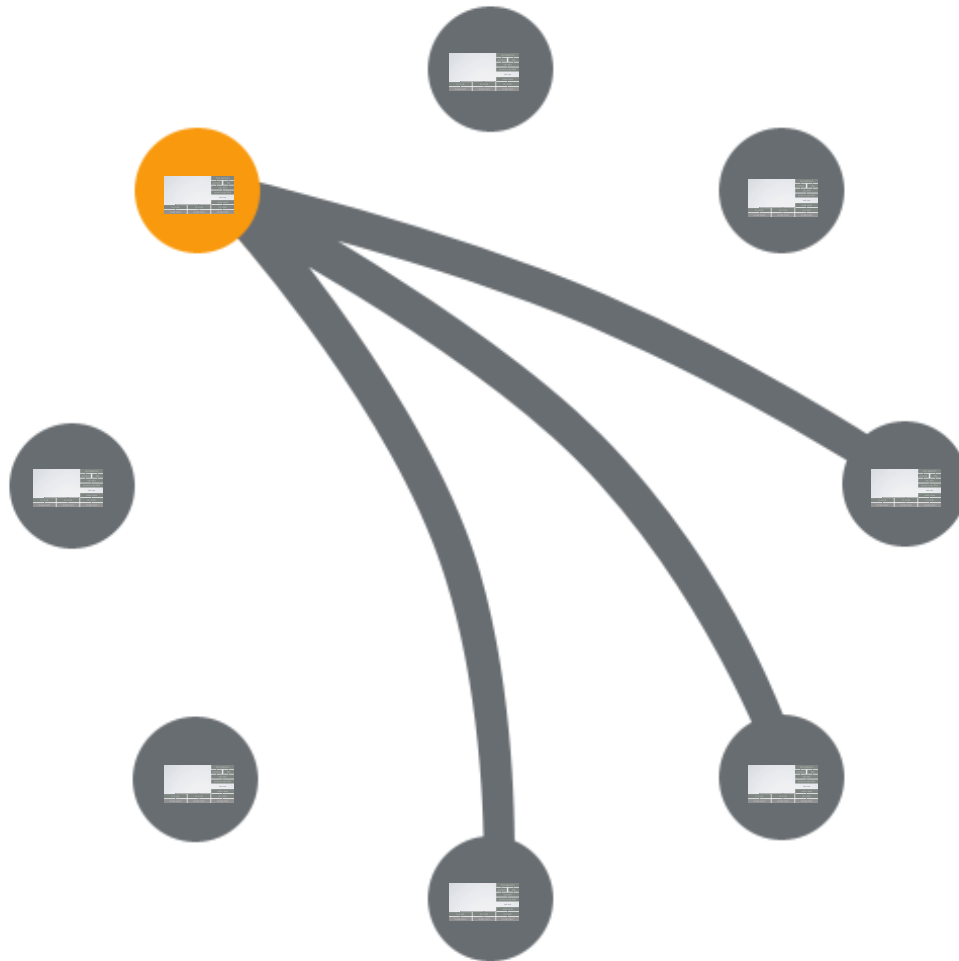
Zoom Out: Virtual Nodes

Many storage instances per server.
If one fails, whole system is okay.

Also good for operational sanity
when adding or removing hosts.



Zoom Out: Riak is a Distributed System



Component Failure: reboot during record write

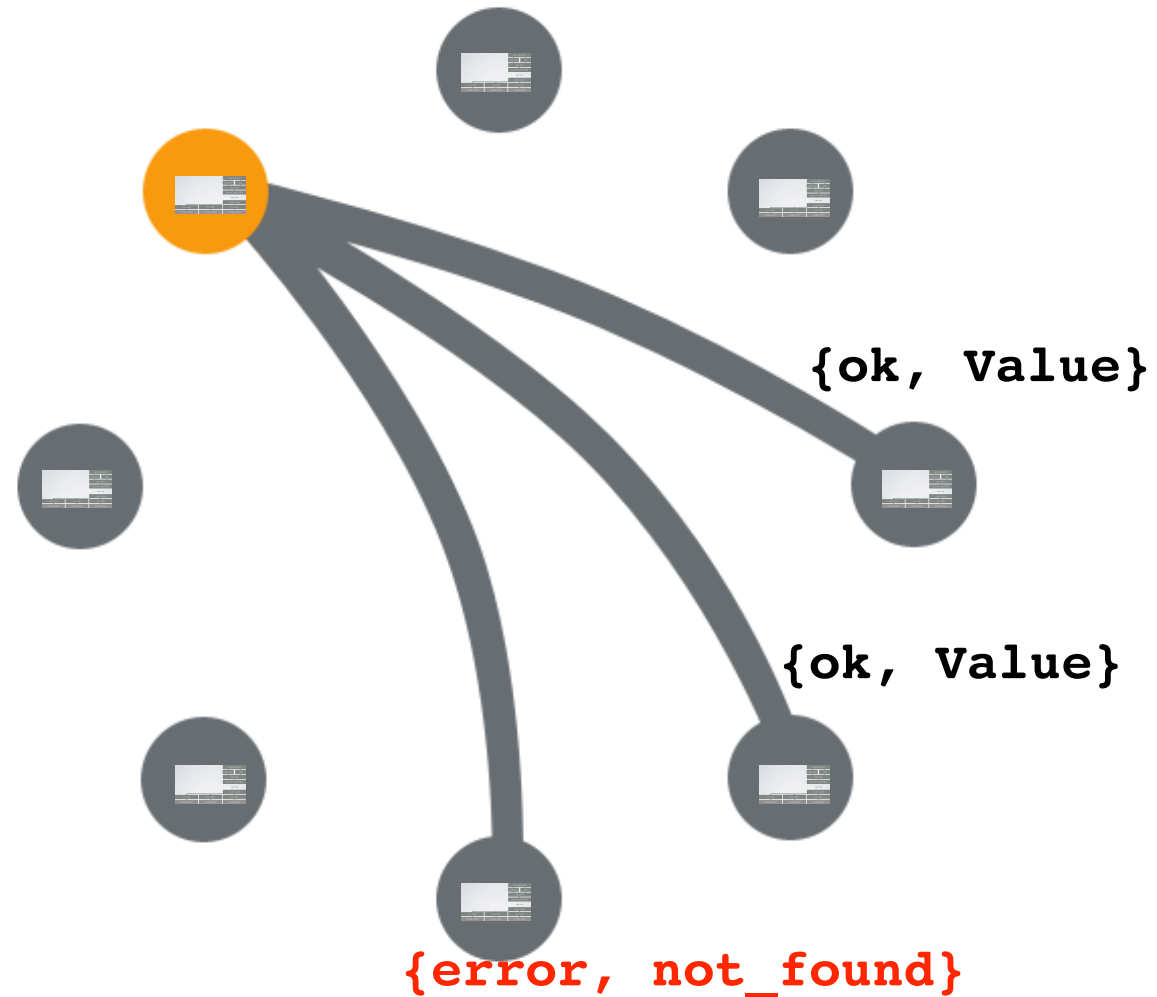
What about a half-written write?

Two problems: detection, **minimization**.

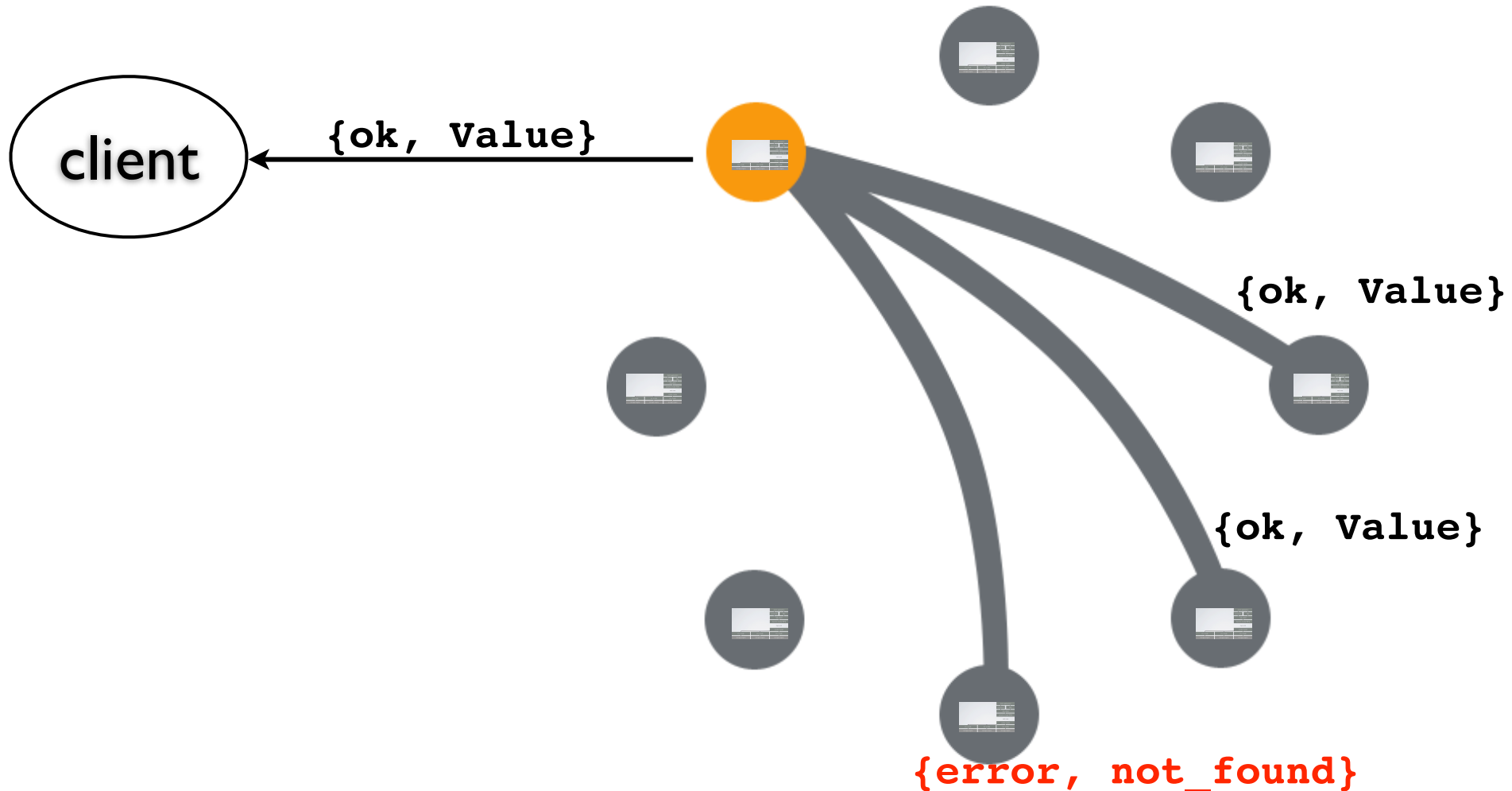
invalidate only the end-failed record, not the file

Isn't this still a busted record?

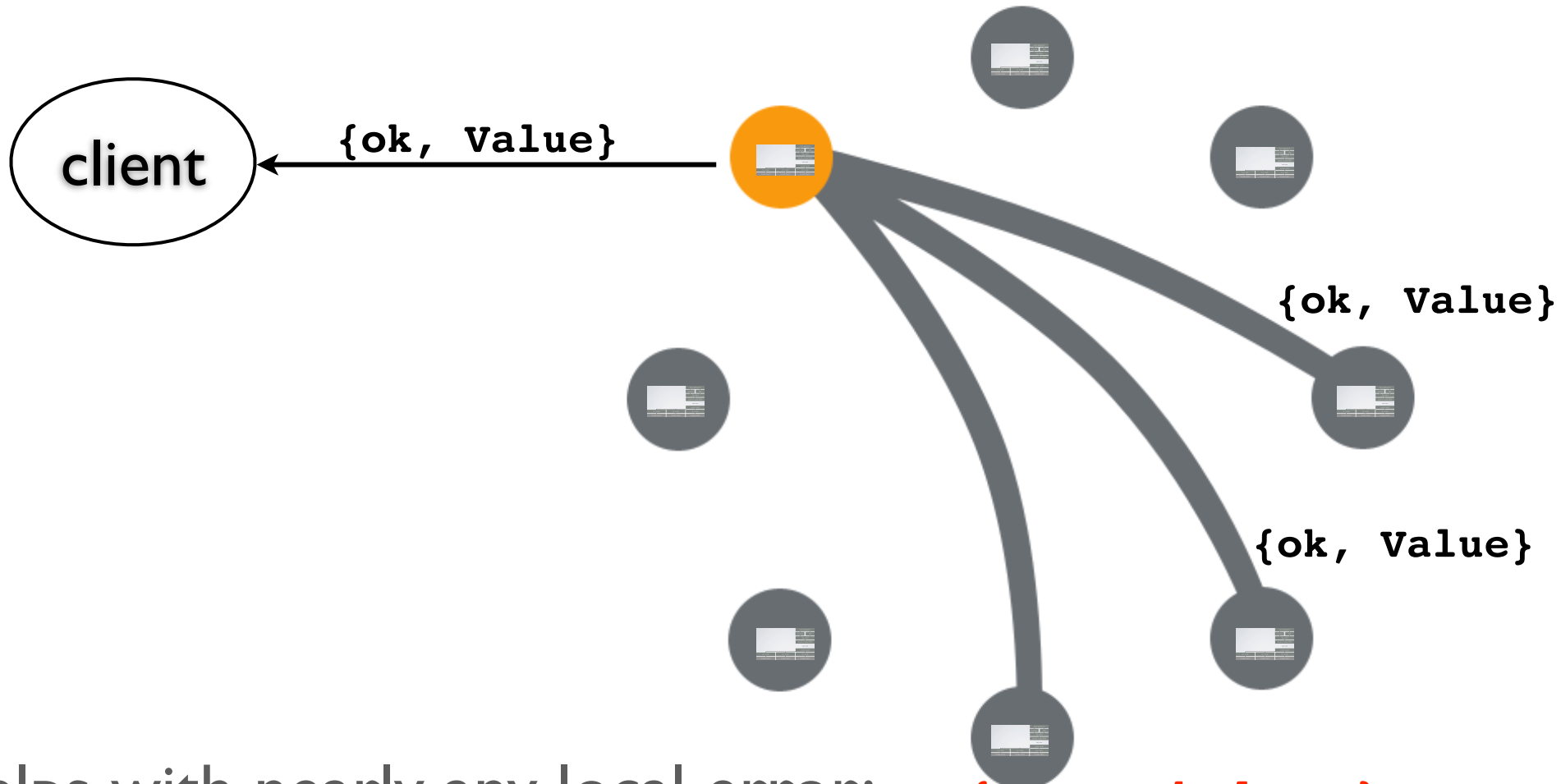
Mitigation: quorum reads



Mitigation: quorum reads



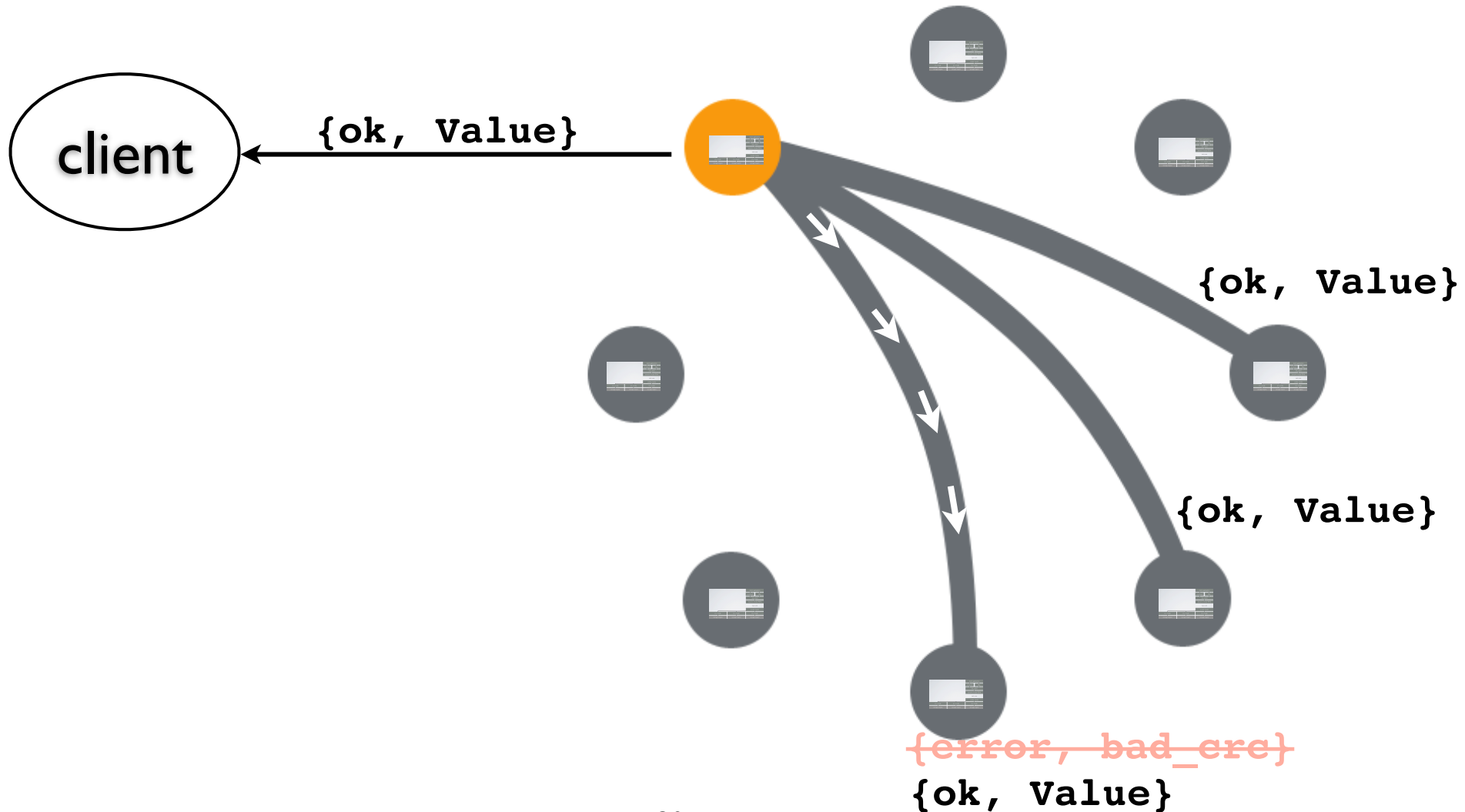
Mitigation: quorum reads



helps with nearly any local error:

`{error, bad_crc}`

Mitigation: read-repair



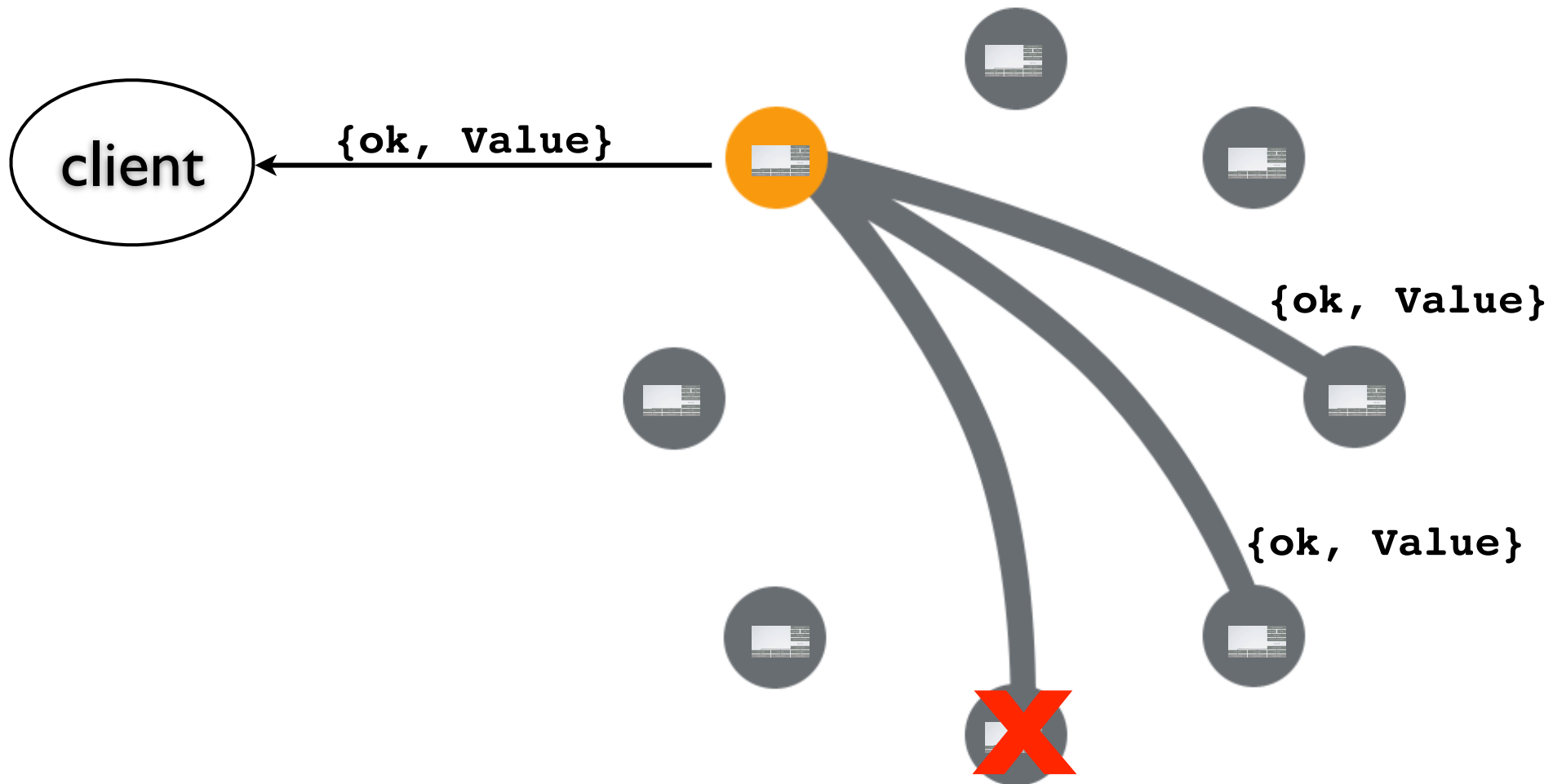
Component Failure: server down!

From a distributed system's point of view,
a whole server can be seen as "a component."

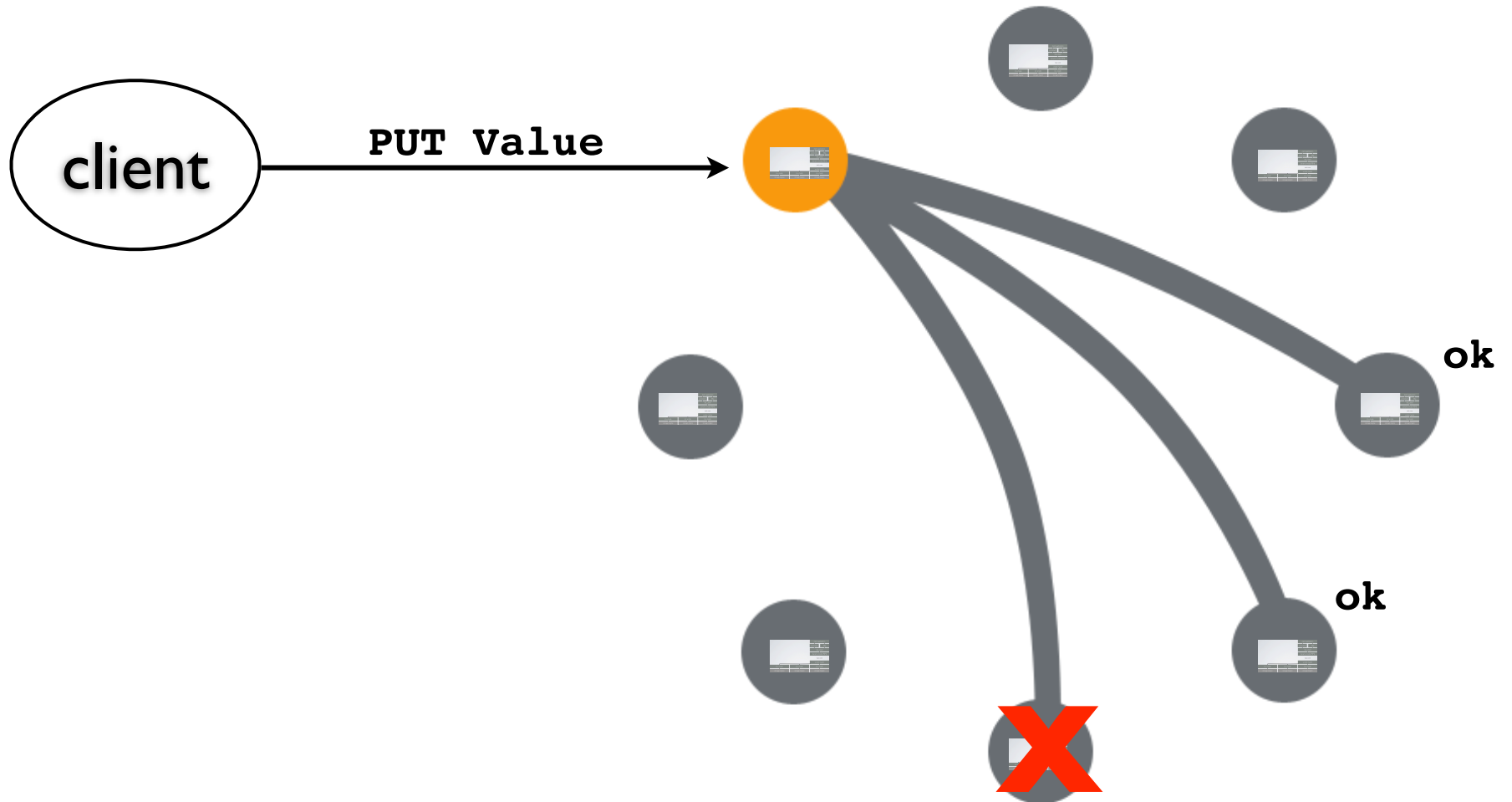
Computers fail **all the time**.

How can the overall system continue to perform?

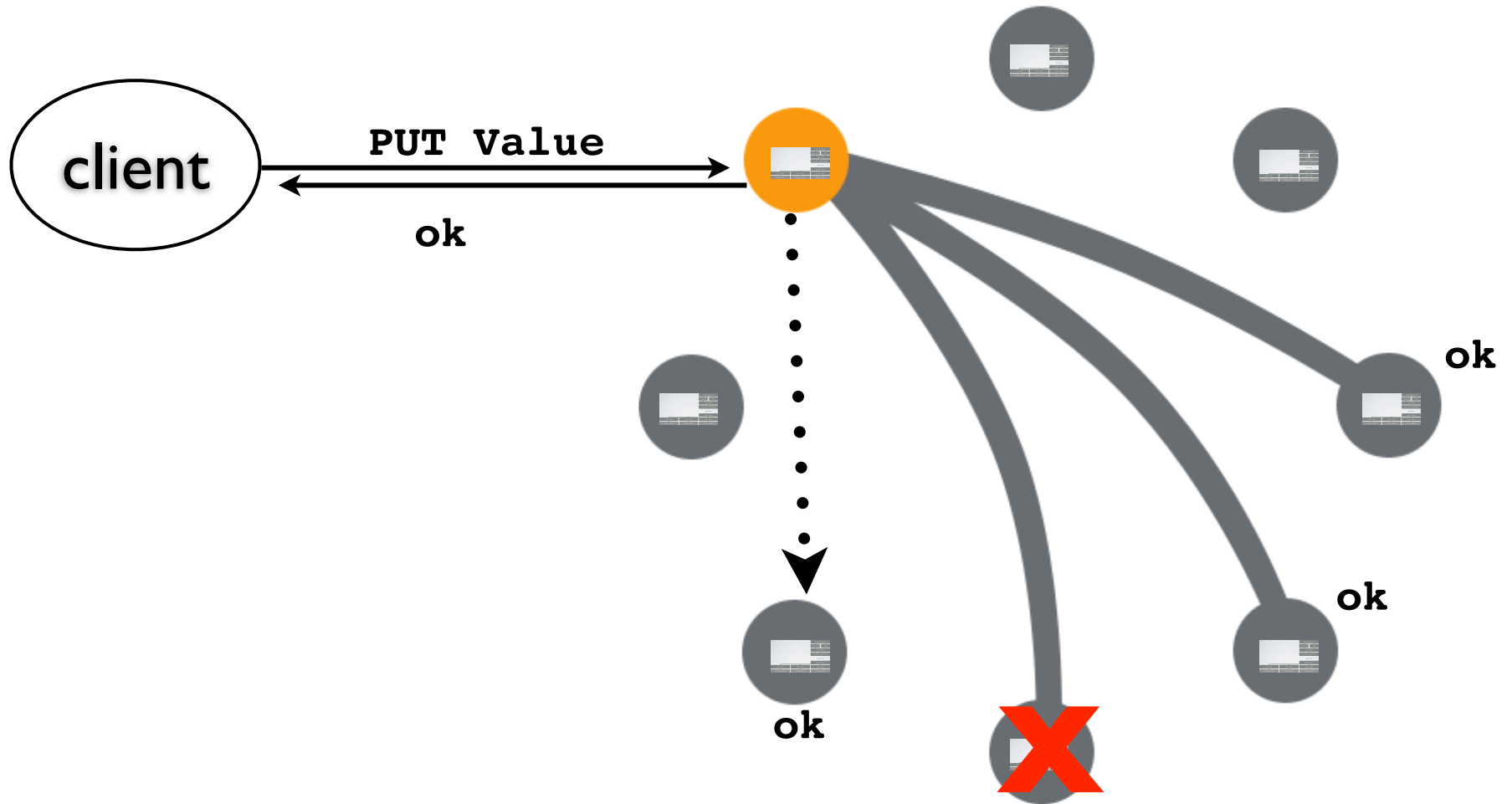
Mitigation: quorum reads



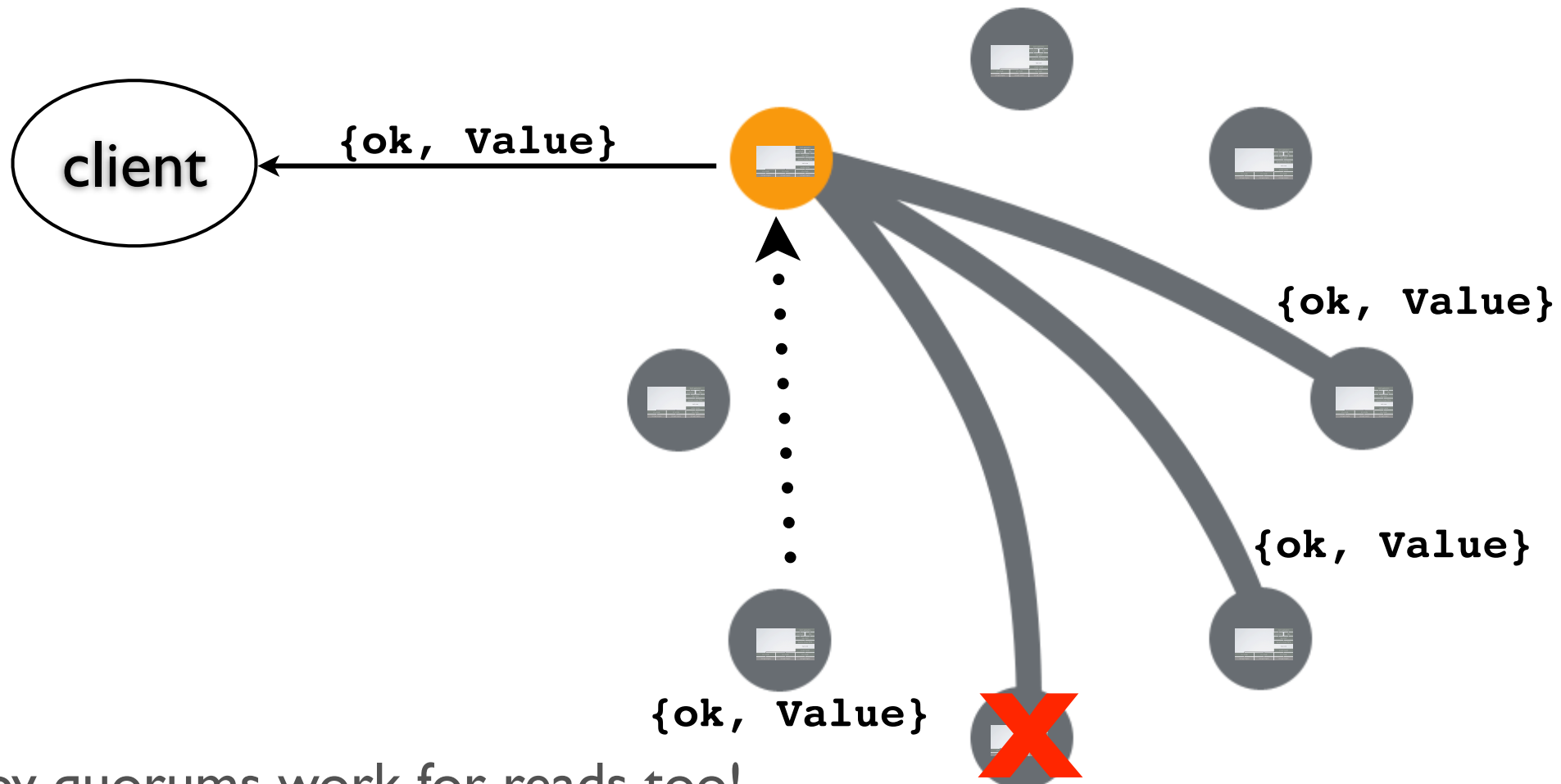
What about writes?



Mitigation: sloppy quorum

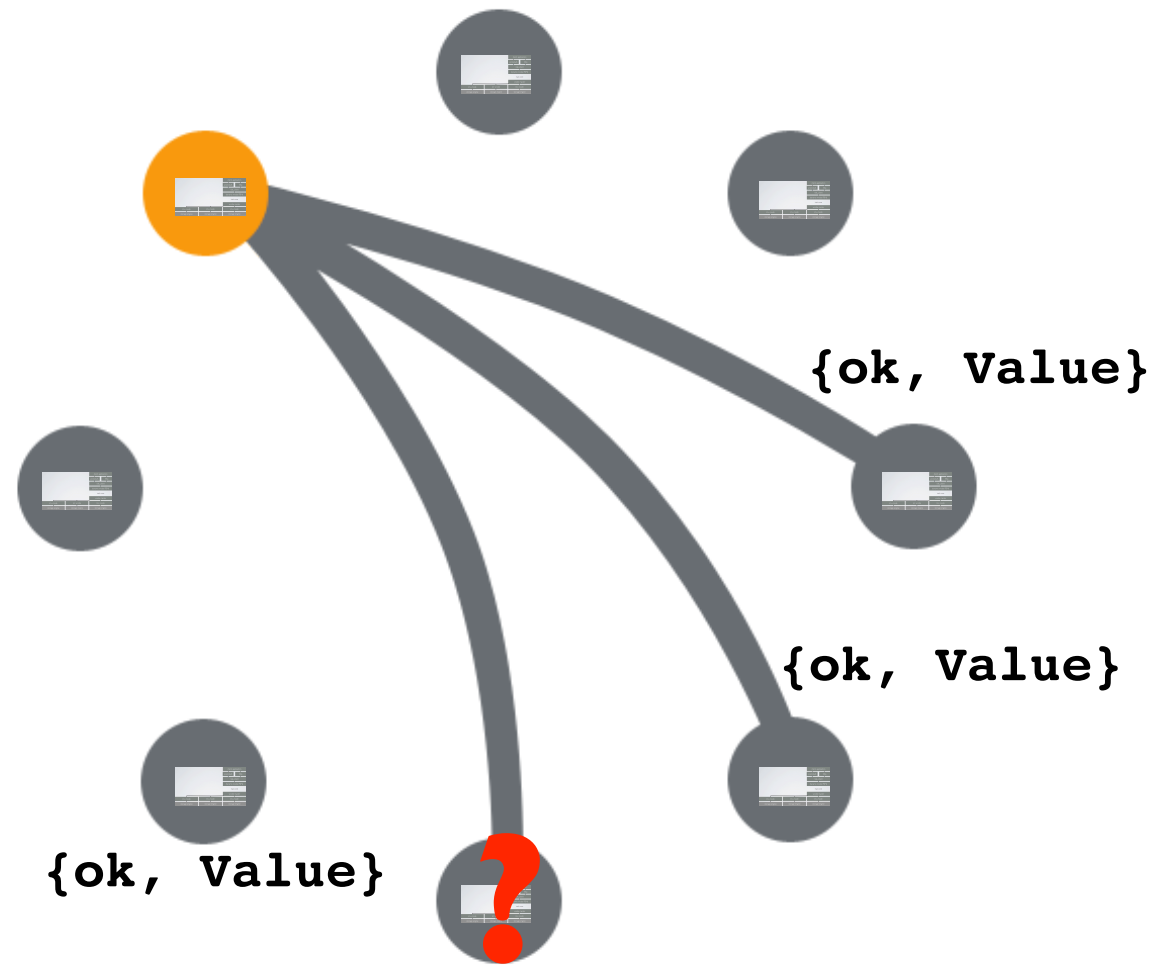


Mitigation: sloppy quorum

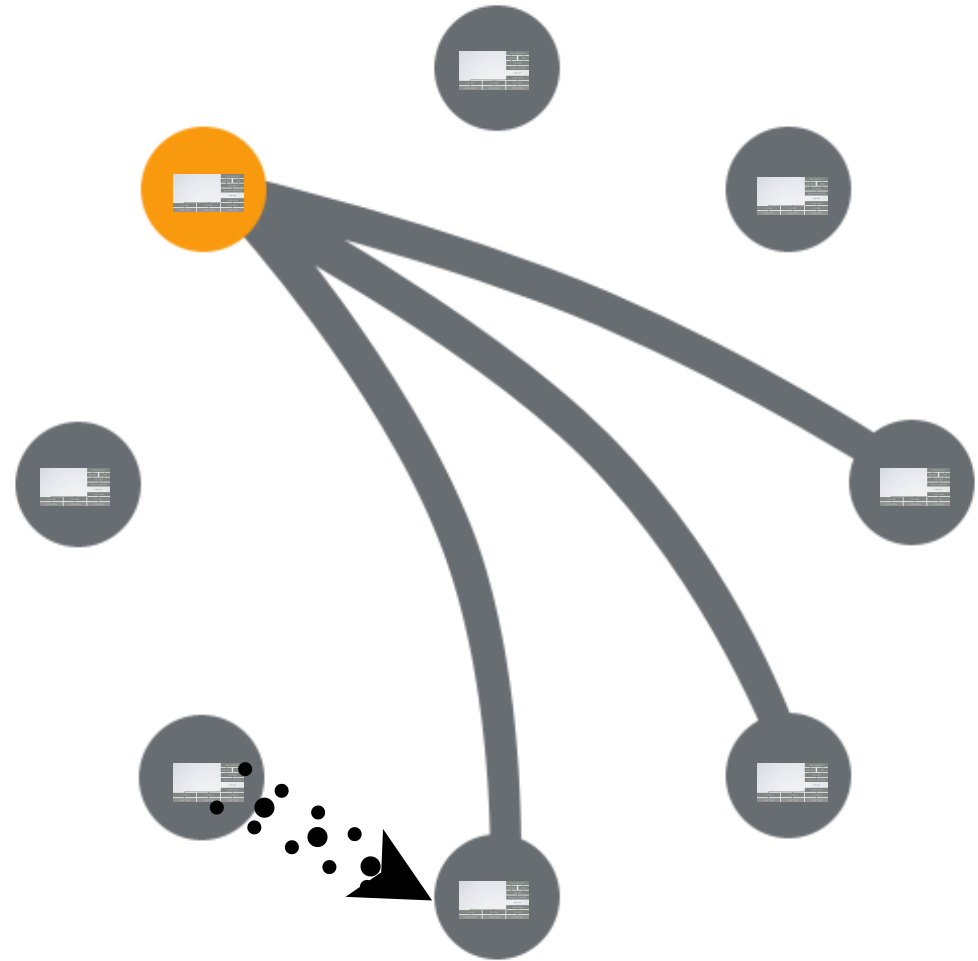


sloppy quorums work for reads too!

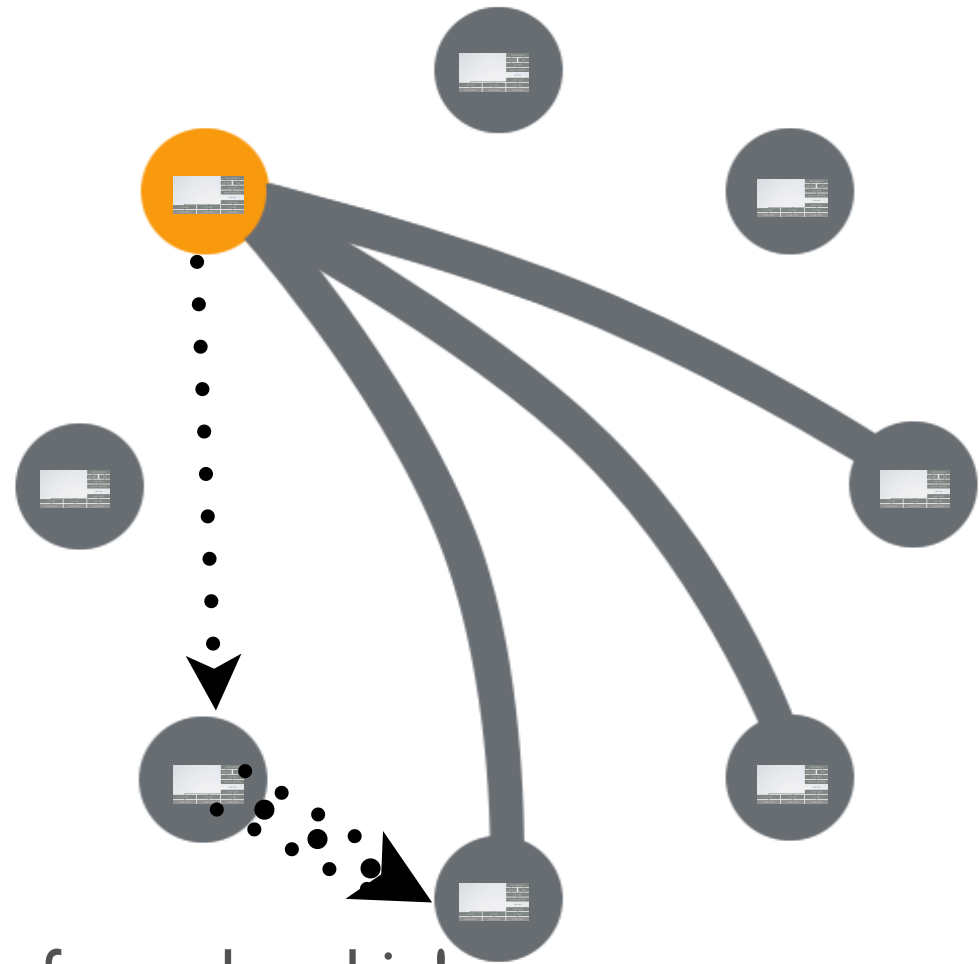
sloppy quorums are sloppy



Mitigation: hinted handoff

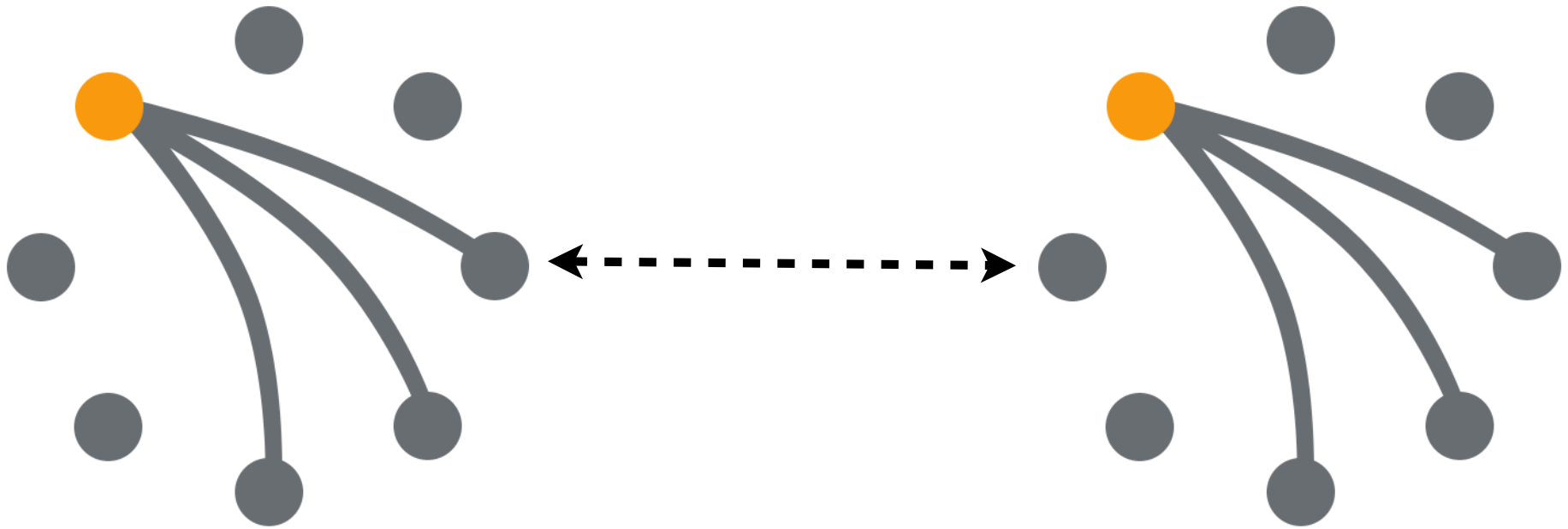


Mitigation: hinted handoff

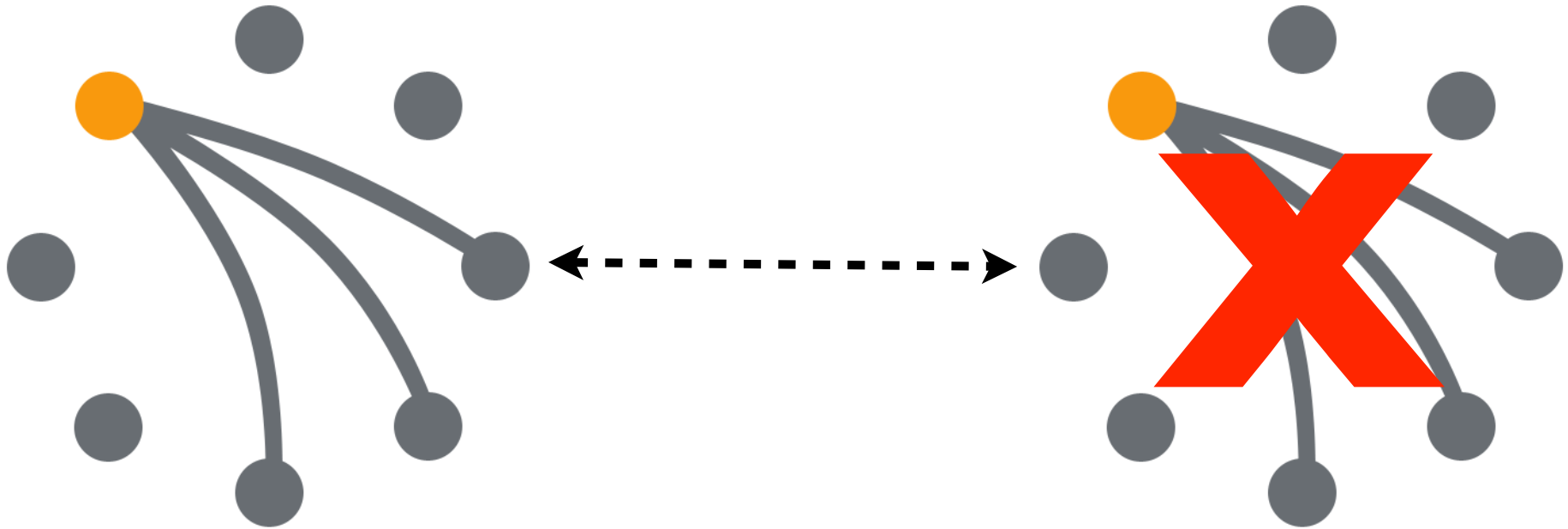


also a fix for inconsistent view of membership!

Zoom out: multiple clusters



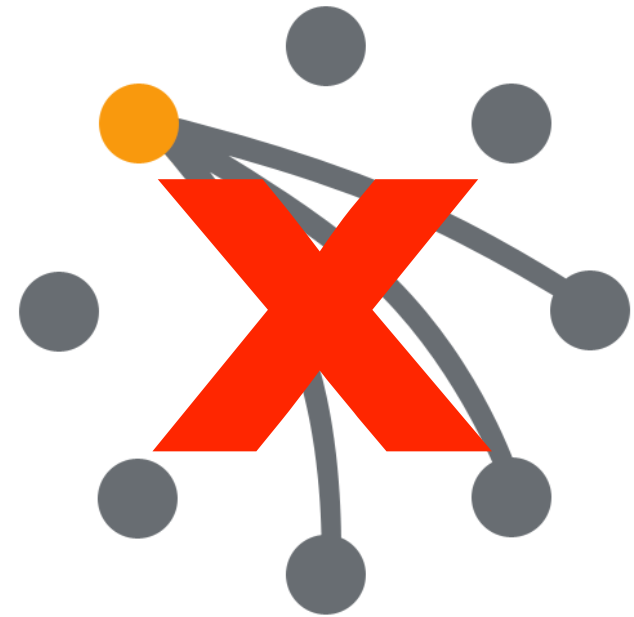
Component Failure: datacenter-level outage



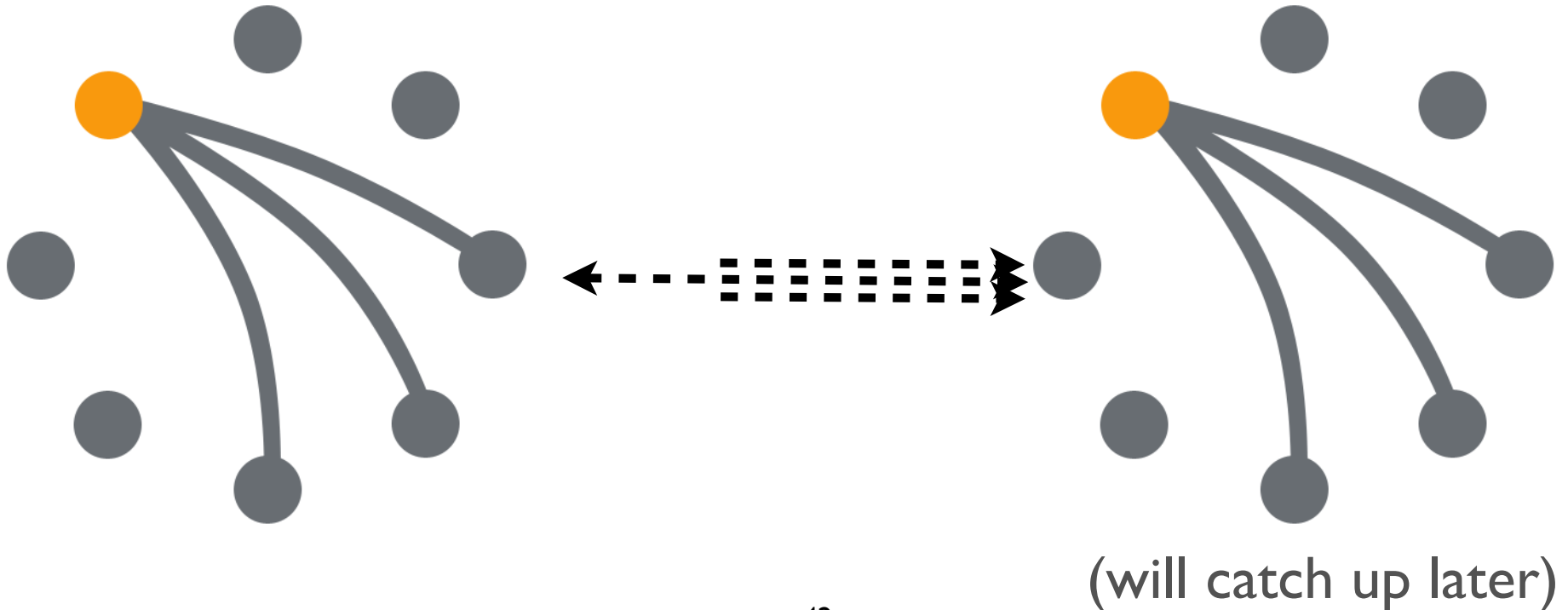
Mitigation: masterless replication



still live!



Mitigation: masterless replication



Things break.

 **riak bends.**



Justin Sheehy
justin@basho.com