BAR Gossip

Lorenzo Alvisi
UT Austin

MAD Services

- Nodes collaborate to provide service that benefits each node
- Service spans multiple administrative domains (MADs)
- Examples:

  Overlay routing, wireless mesh routing, content distribution, archival storage, ...

How MAD Services Fail

- Nodes can break
  - Fail-stop e.g., disk crash
  - Byzantine – arbitrary deviation

    Misconfigured, compromised by virus,
    operator error ("rm –rf *"), malicious user, …

How MAD Services Fail

- Nodes can break
  - Fail-stop e.g., disk crash
  - Byzantine – arbitrary deviation

    Misconfigured, compromised by virus,
    operator error ("rm –rf *"), malicious user, ...
- Nodes can be selfish
  - Minimize work and maximize gain

    e.g., in a cooperative backup service, store less than fair share of data

Byzantine Model [Lamport 1982,...]

- Tolerates arbitrary deviations from specification
- Can be practical

  [Castro and Liskov 1999, Rodrigues et al 2001, Yin et al 2003, Abd El-Malek et al 2005, Johansen et al 2006, Cowling et al 2006]

Byzantine Model [Lamport 1982,...]

- Tolerates arbitrary deviations from specification
- Can be practical

  [Castro and Liskov 1999, Rodrigues et al 2001, Yin et al 2003, Abd El-Malek et al 2005, Johansen et al 2006, Cowling et al 2006]

- Limits number f of faulty nodes
  - e.g. Agreement requires $f < n/3$
- Assumes all other nodes are correct

Inappropriate when all nodes may deviate when in their interest

Rational Model [Nash 1950,...]

- All nodes are rational, and rational nodes can deviate selfishly from their specification

  [Papadimitriou 2001, Cox and Noble 2003, Littlebridge et al 2003...]

Rational Model [Nash 1950,…]

- All nodes are rational, and rational nodes can deviate selfishly from their specification

  [Papadimitriou 2001, Cox and Noble 2003, Littlebridge et al 2003…]

- Does not tolerate Byzantine behavior
  - Broken nodes may violate assumptions
  - Malicious nodes may cause unbounded damage

Inappropriate when some node
may deviate against its interest

Three Challenges

1. To develop a model in which it is possible to <u>prove</u> properties about MAD services

2. To understand how to simplify the development of MAD services in the new model

3. To demonstrate that MAD services developed under the new model can be practical

Who's to blame


Jeff Napper


Allen Clement


Harry Li


Jean-Philippe Martin


Amit Aiyer


Edmund Wong


Lorenzo Alvisi


Mike Dahlin


Indrajit Roy

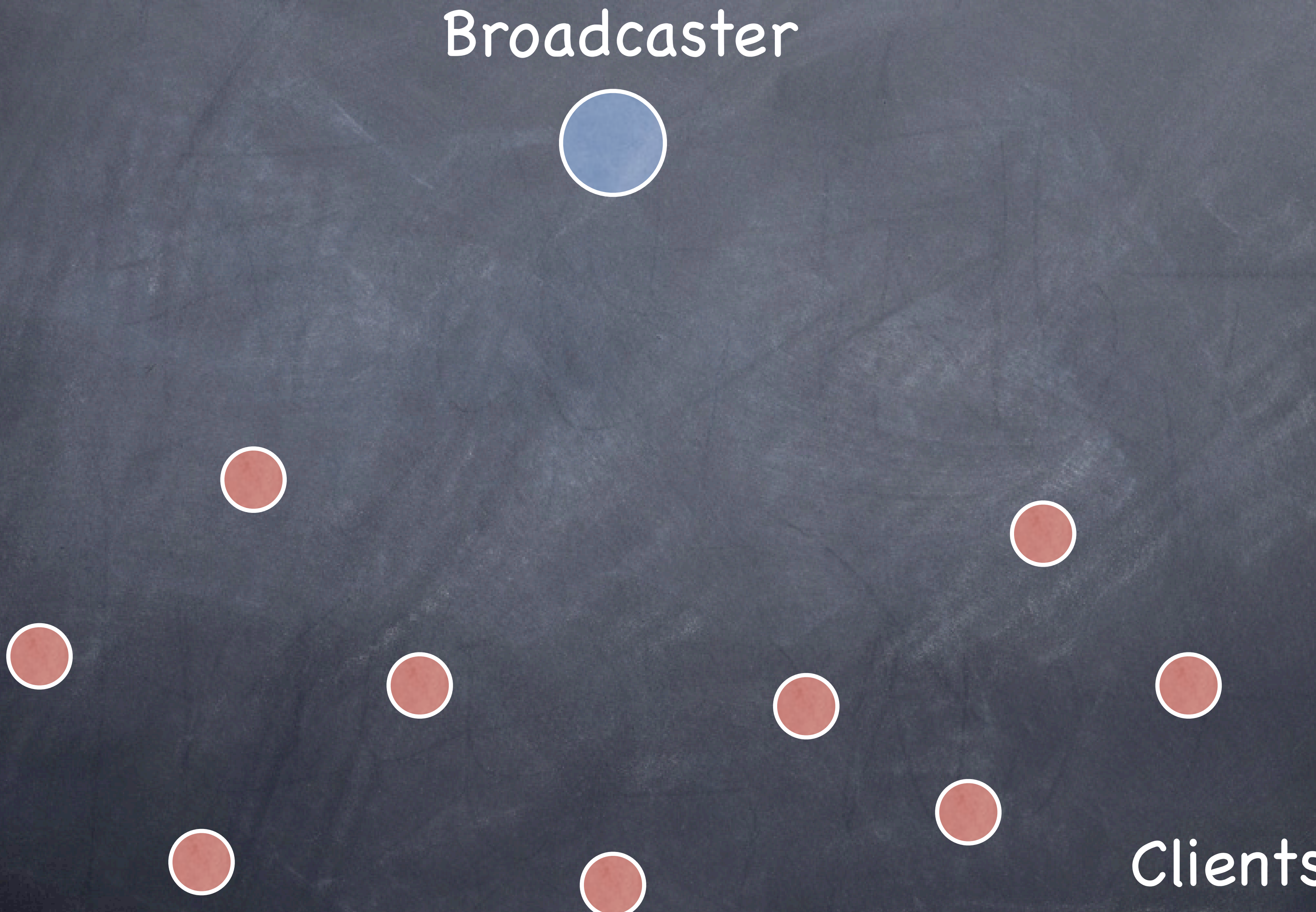
A First Foray

- BAR (Byzantine, Altruistic, Rational) Tolerance
  - no bound on rational nodes
  - utility functions add expectation of Byzantine behavior
- BAR-B, a BAR tolerant cooperative backup service (SOSP 05)
  - uses BAR-tolerant RSM to implement abstraction of Altruistic node on top of Rational and Byzantine ones
- FlightPath, a BAR tolerant data streaming application (OSDI 06)
  - uses BAR-tolerant gossip protocol to disseminate updates

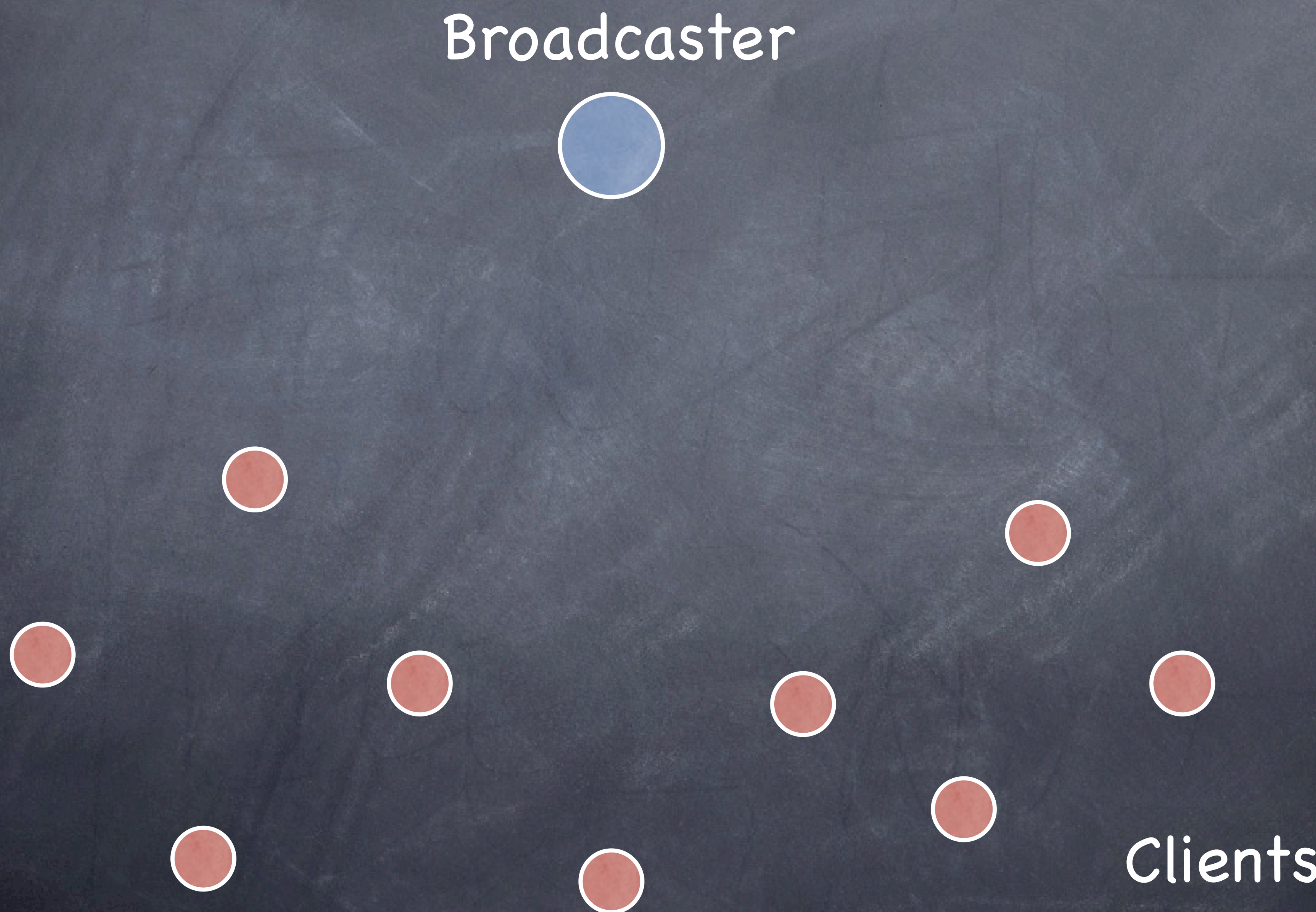
Live Streaming

- Examples: Internet radio, NCAA tournament, web concerts, Internet TV
- Practical challenges:
  - Reduce broadcaster's used bandwidth
  - Minimize latency
  - Increase reliability
  - Tolerate link and node failures

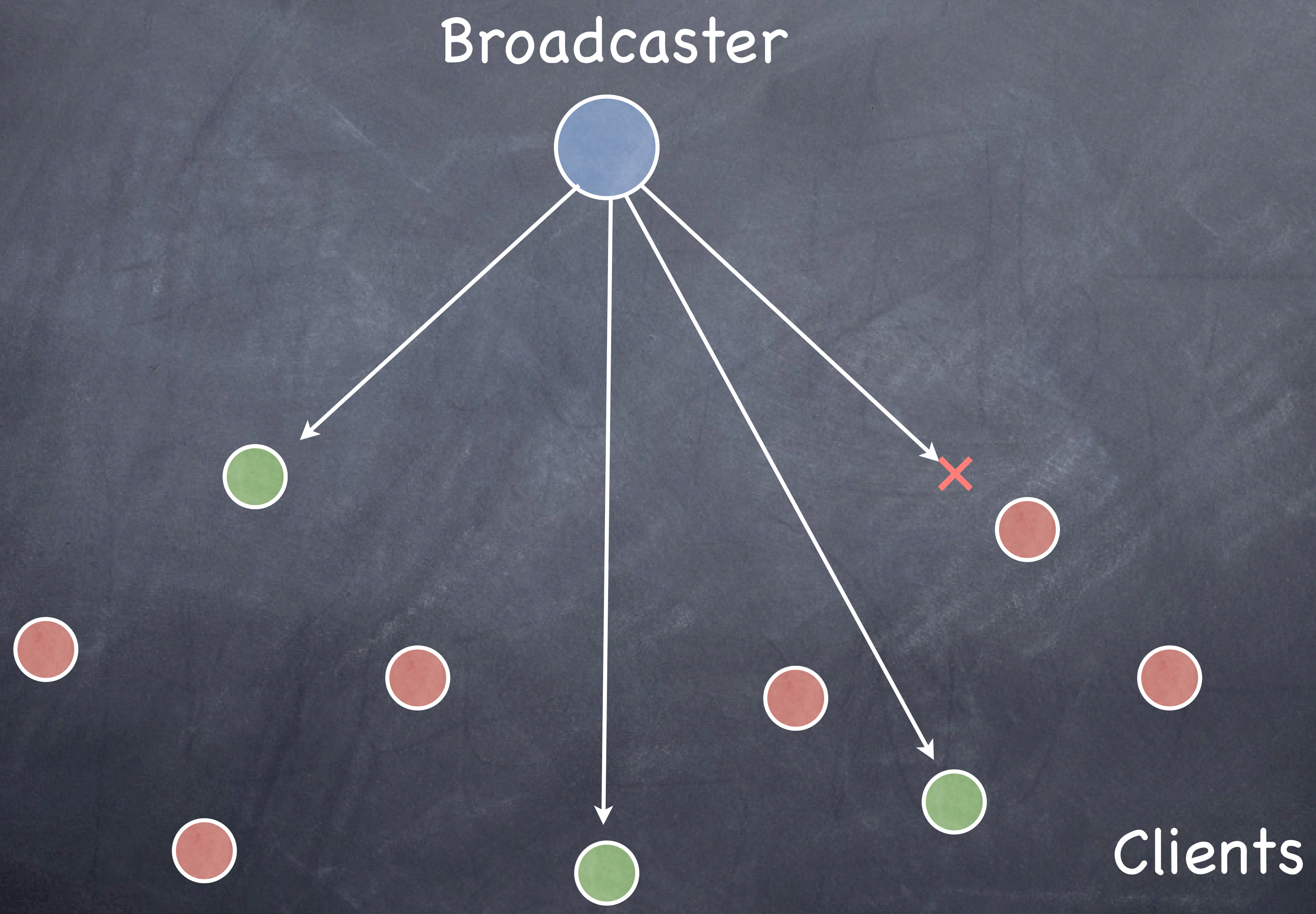
Live Streaming Setup



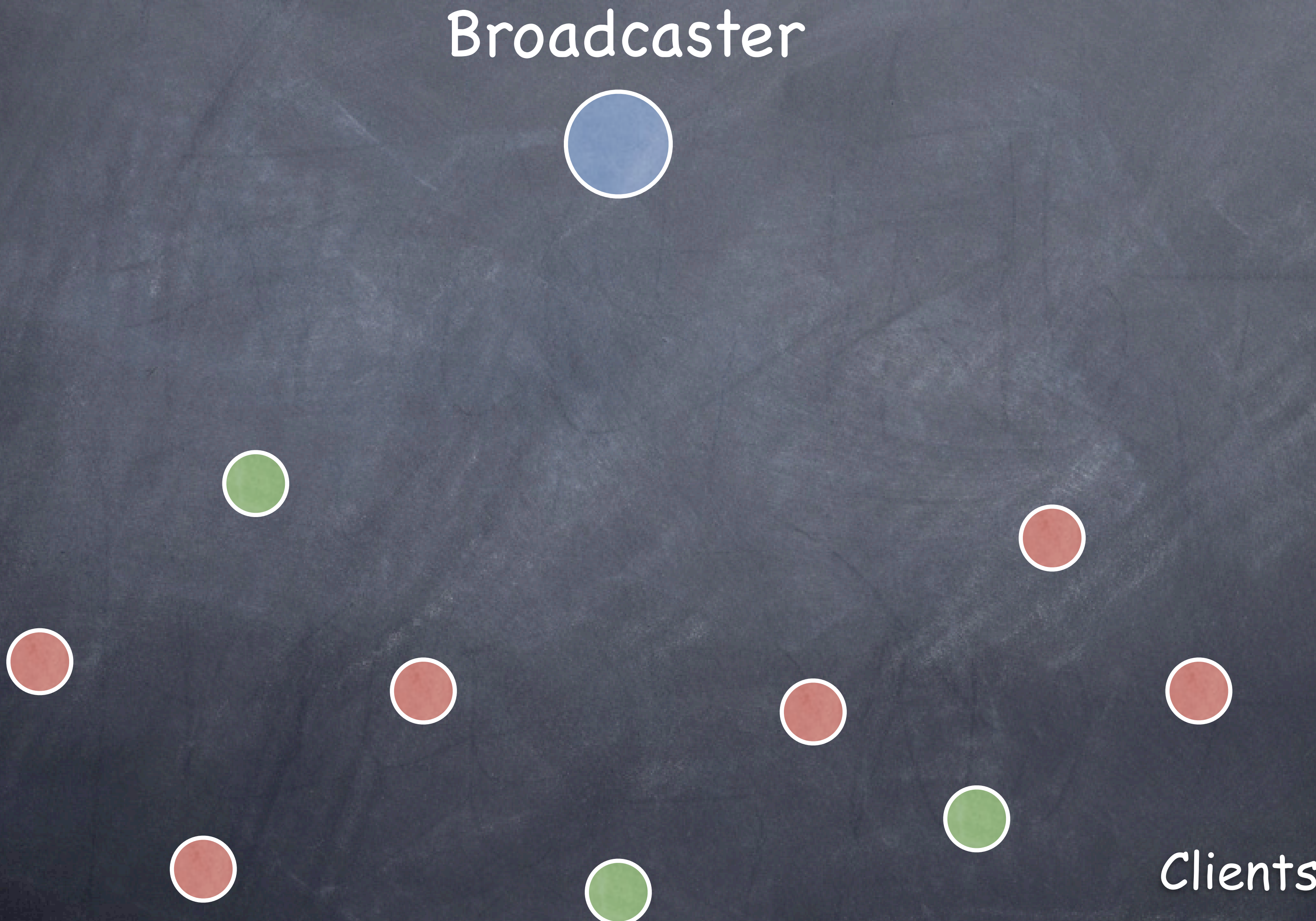
Live Streaming Setup



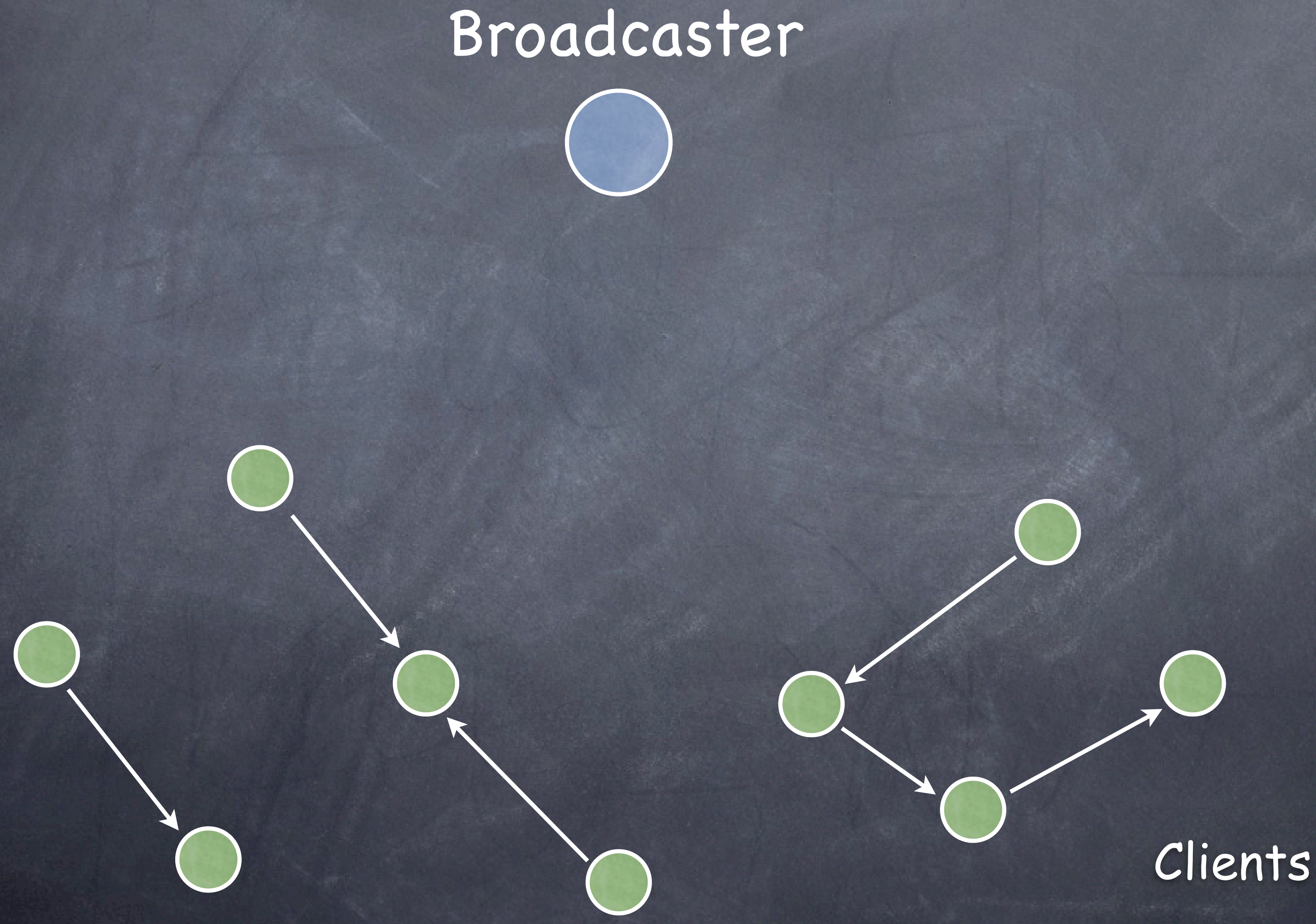
Live Streaming Setup



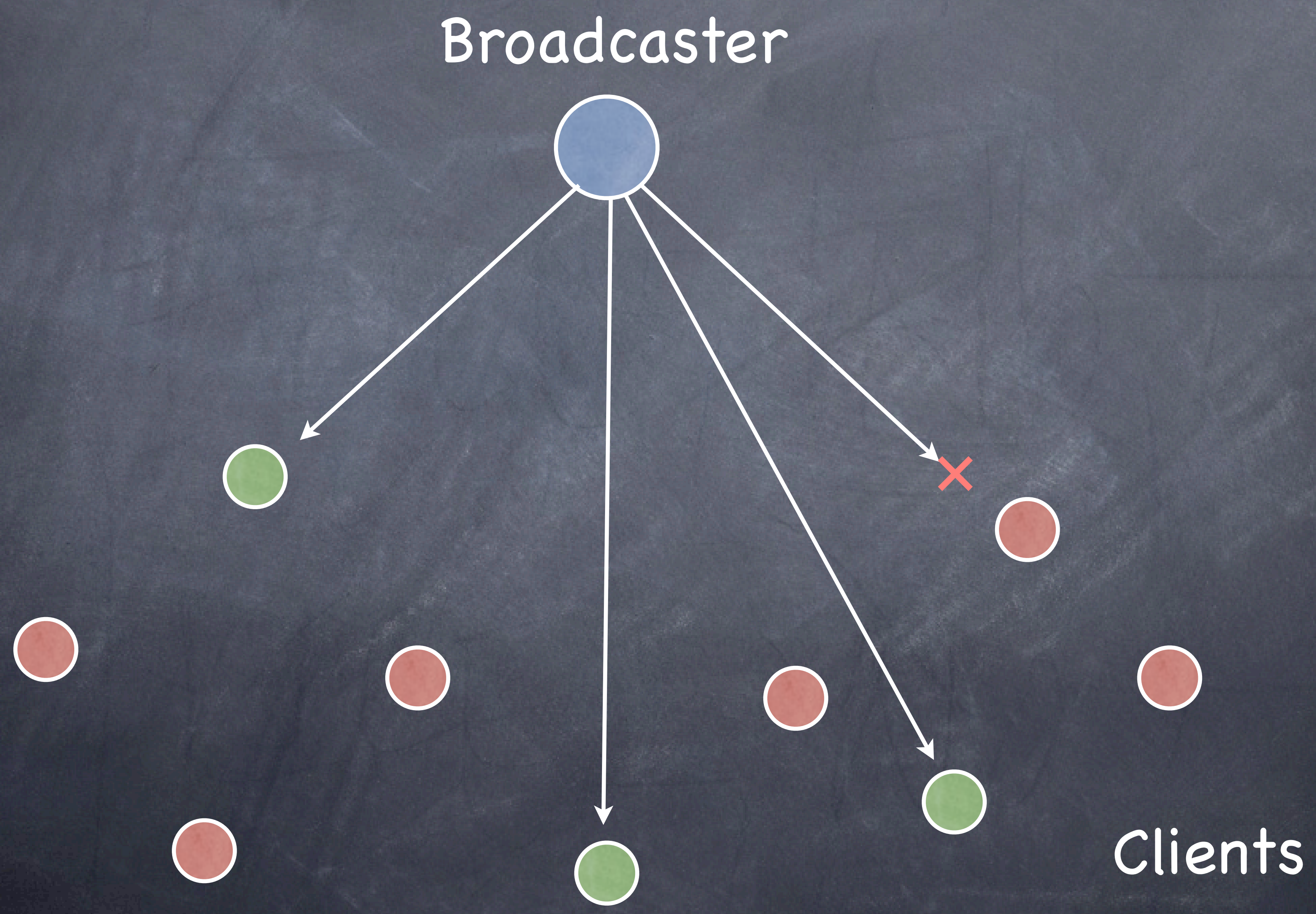
Live Streaming Setup



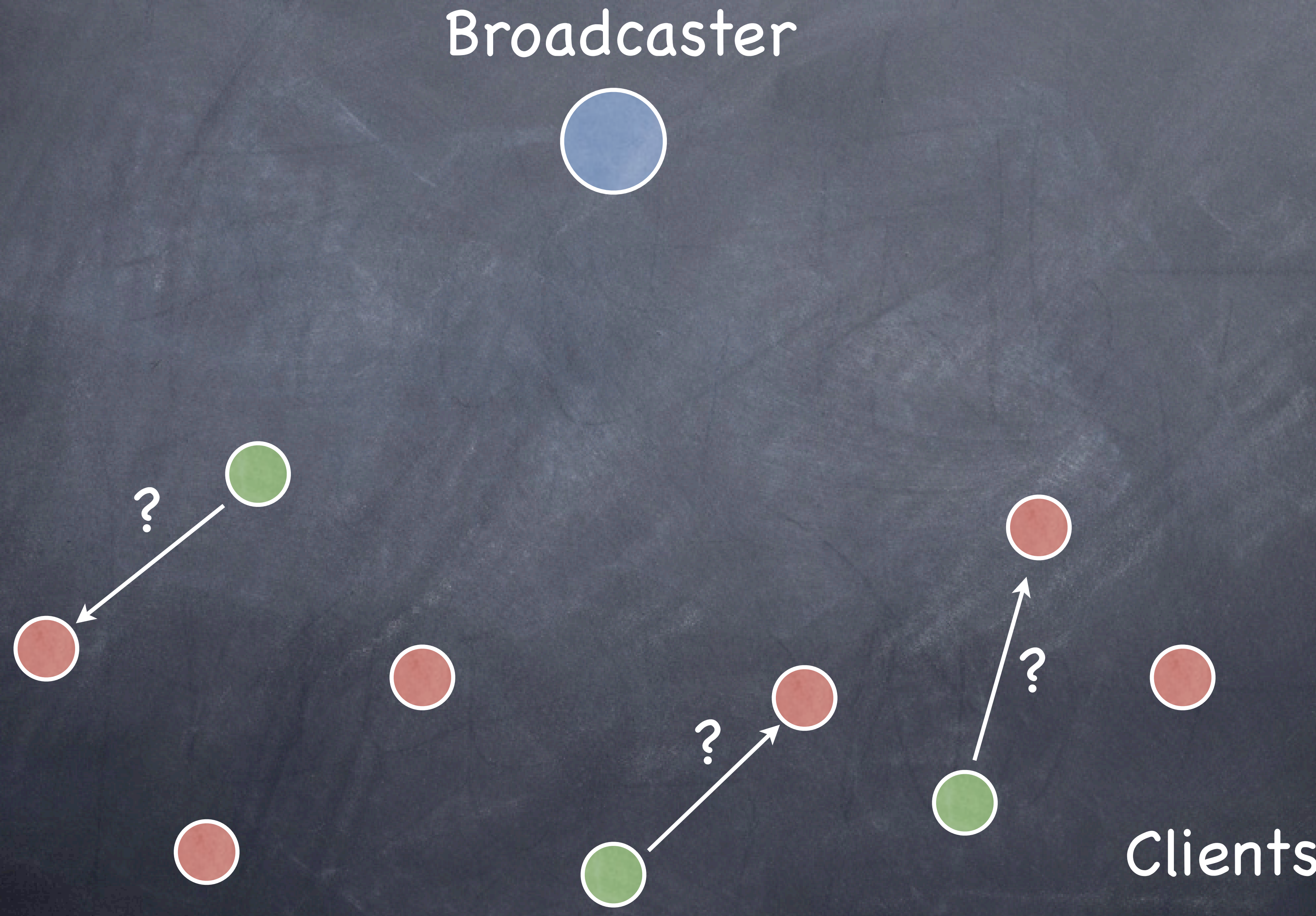
Live Streaming Setup



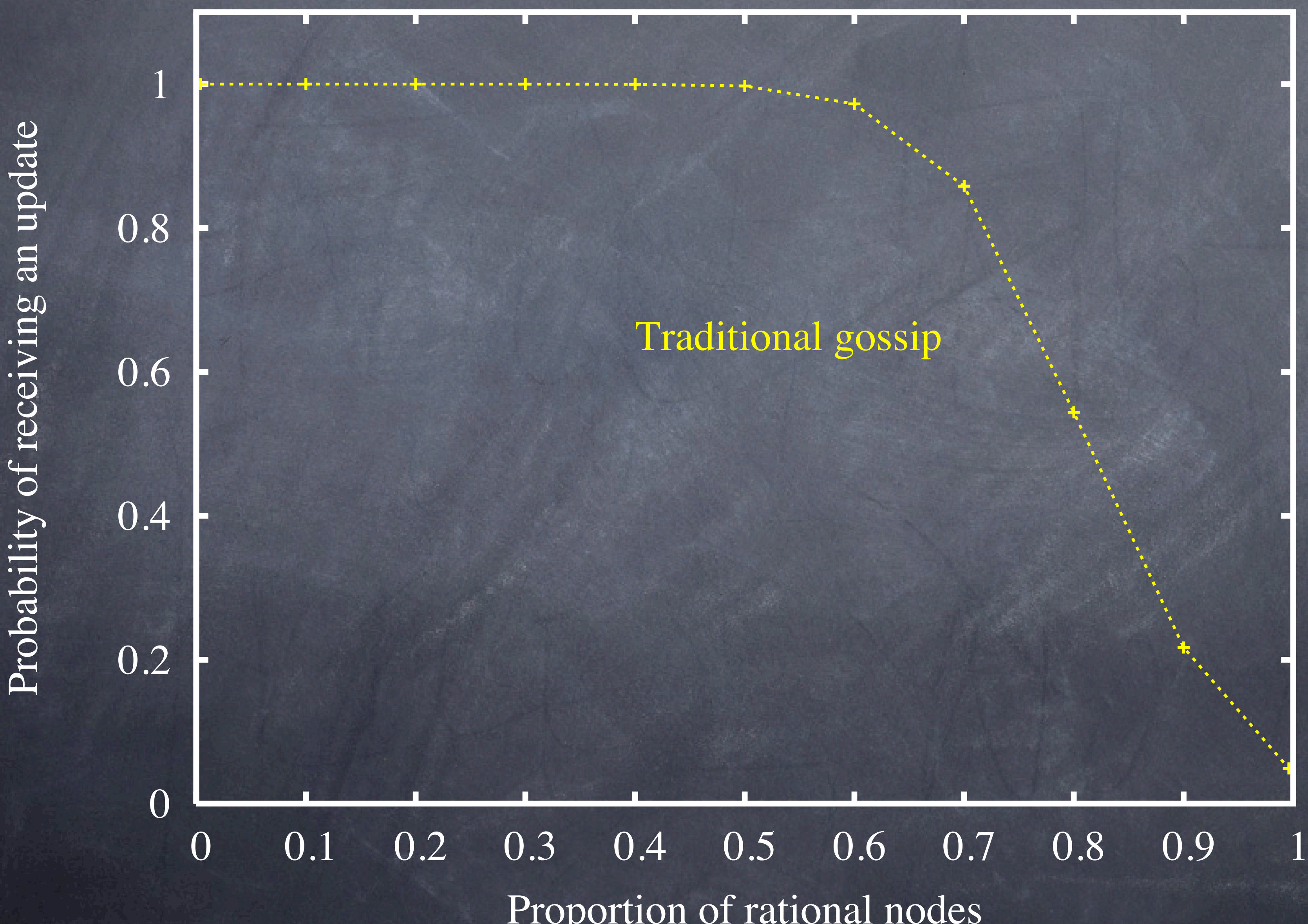
Rational Peers Don't Share!



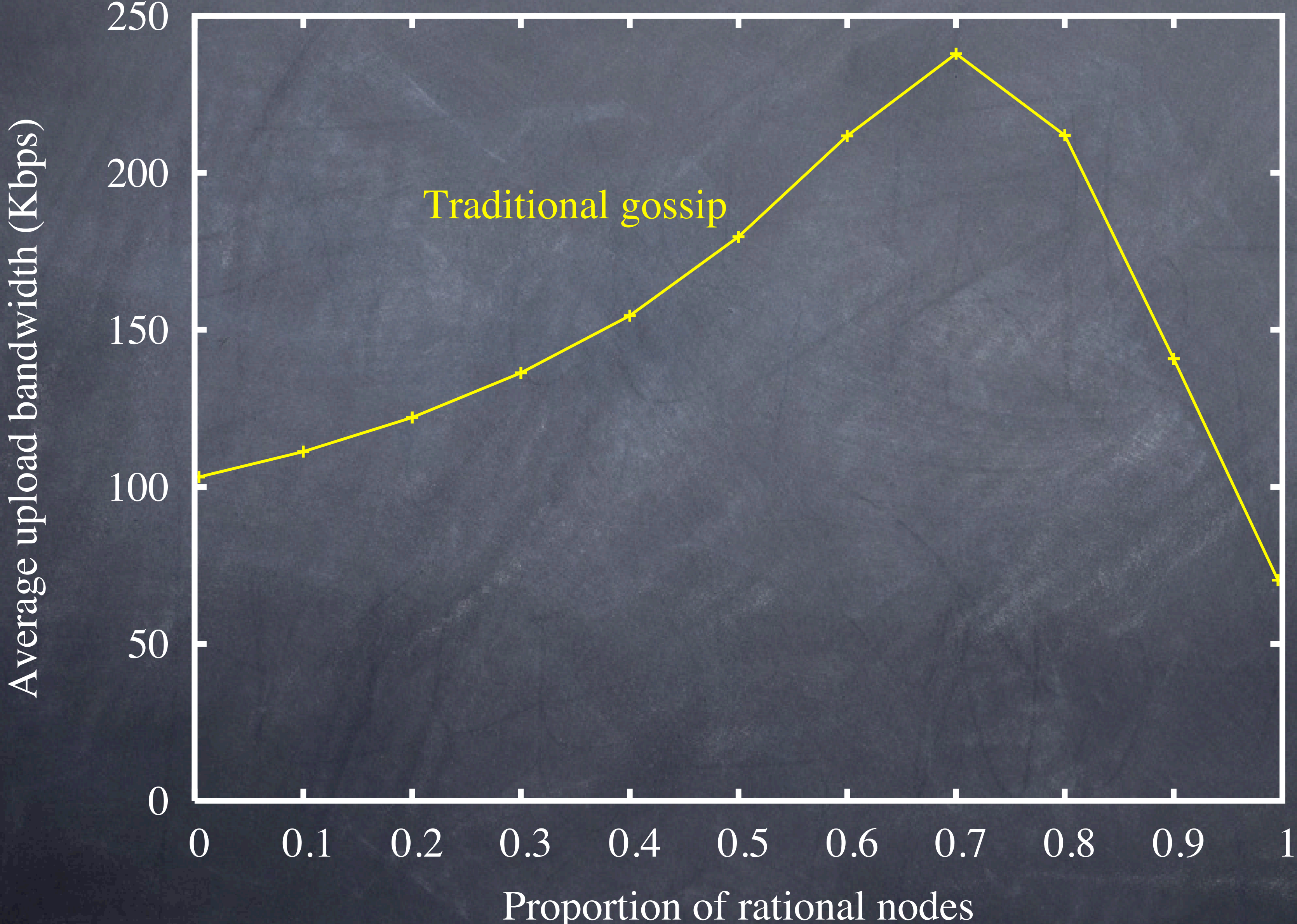
Rational Peers Don't Share!



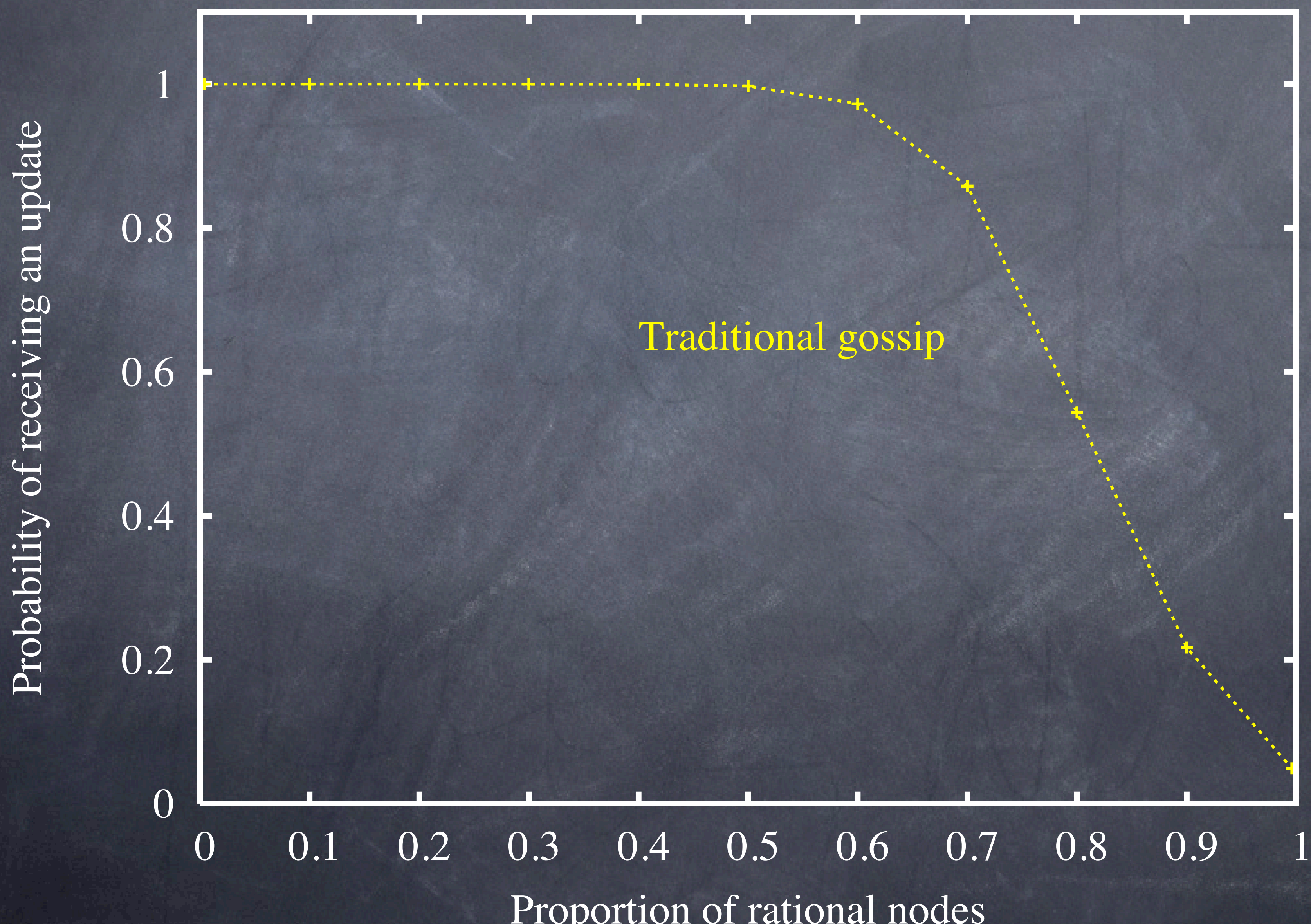
Reliability Degrades...



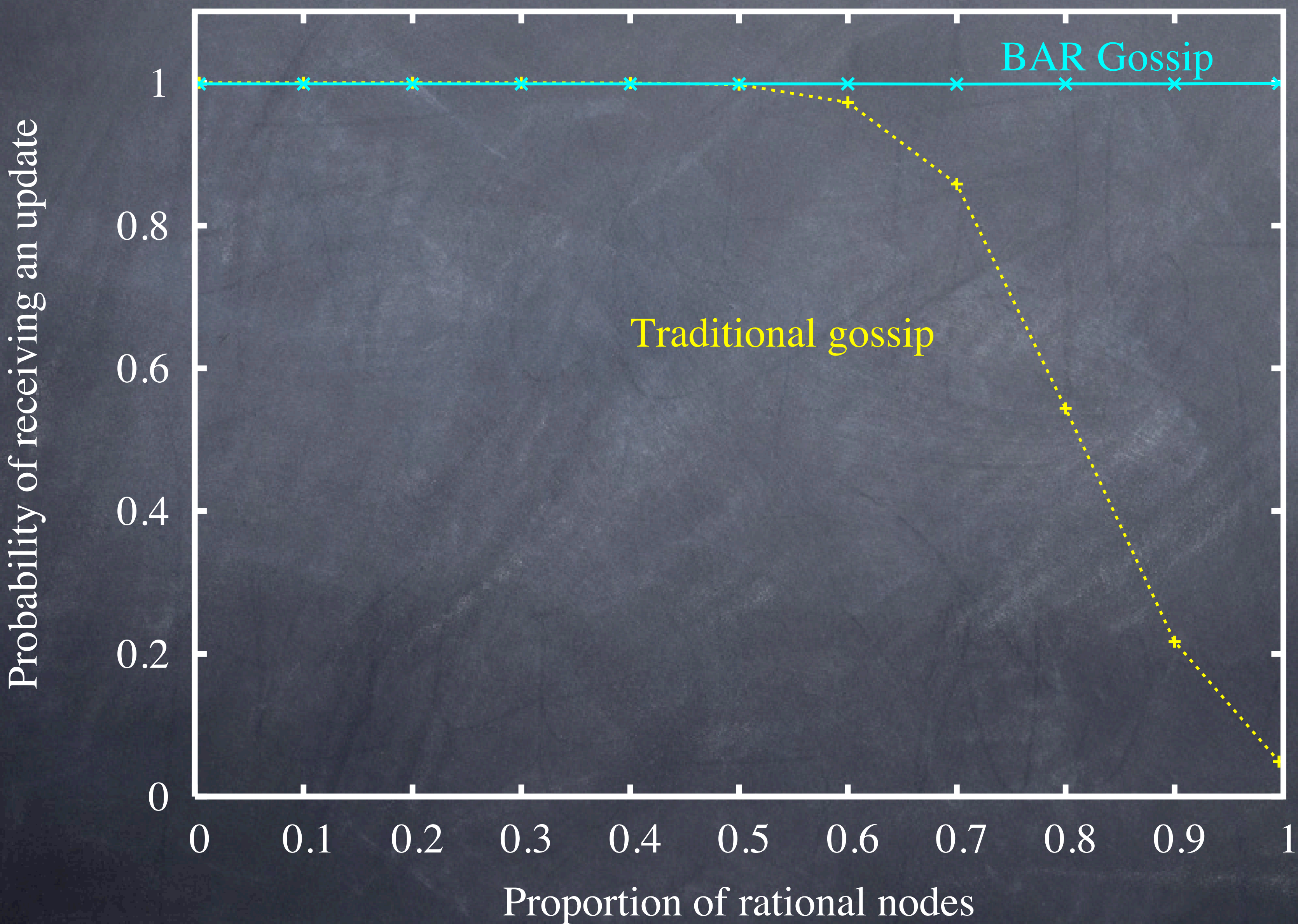
...and Altruistic nodes suffer



BAR Gossip

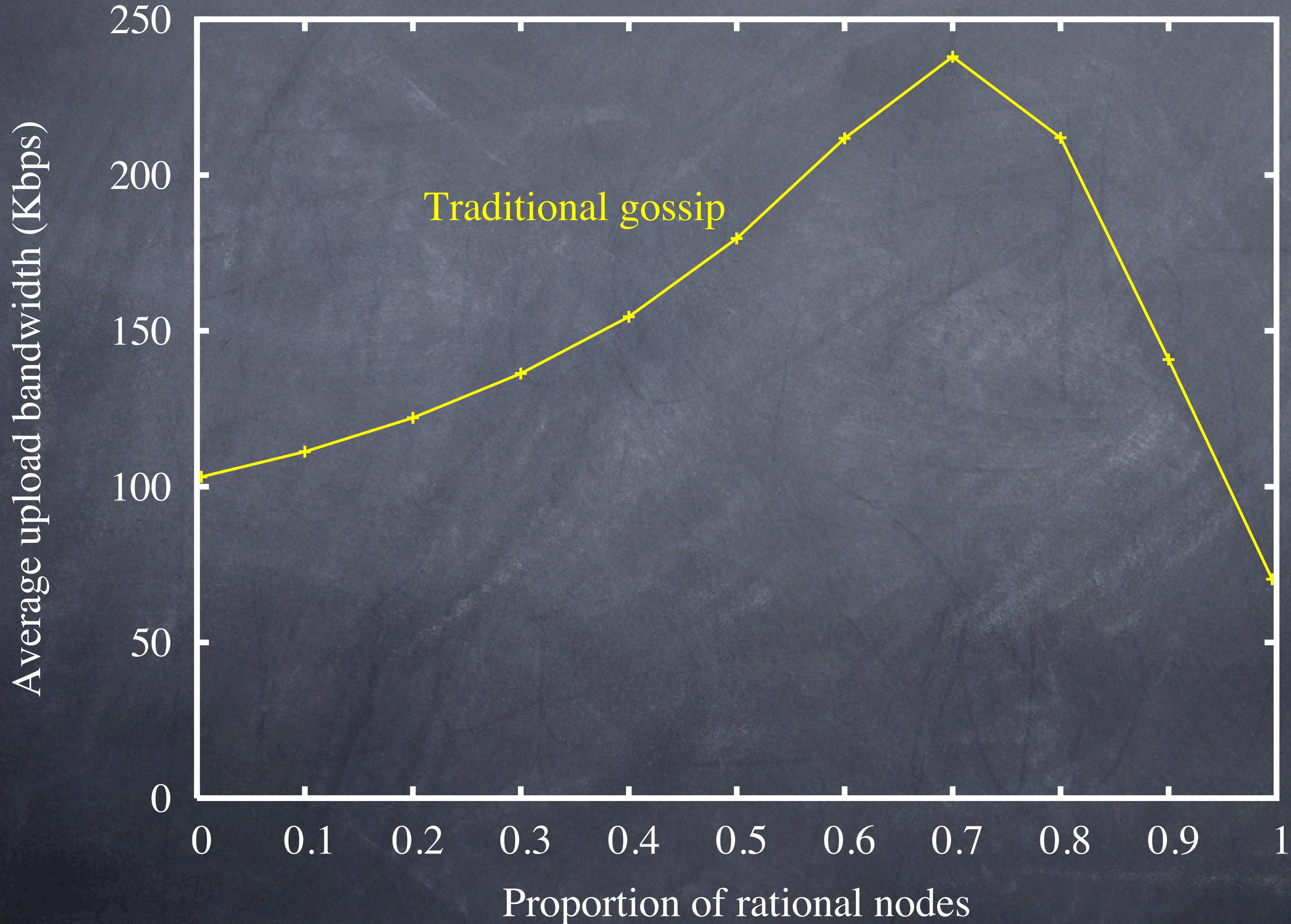

Traditional gossip
Probability of receiving an update
Proportion of rational nodes
0
0.2
0.4
0.6
0.8
1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

BAR Gossip



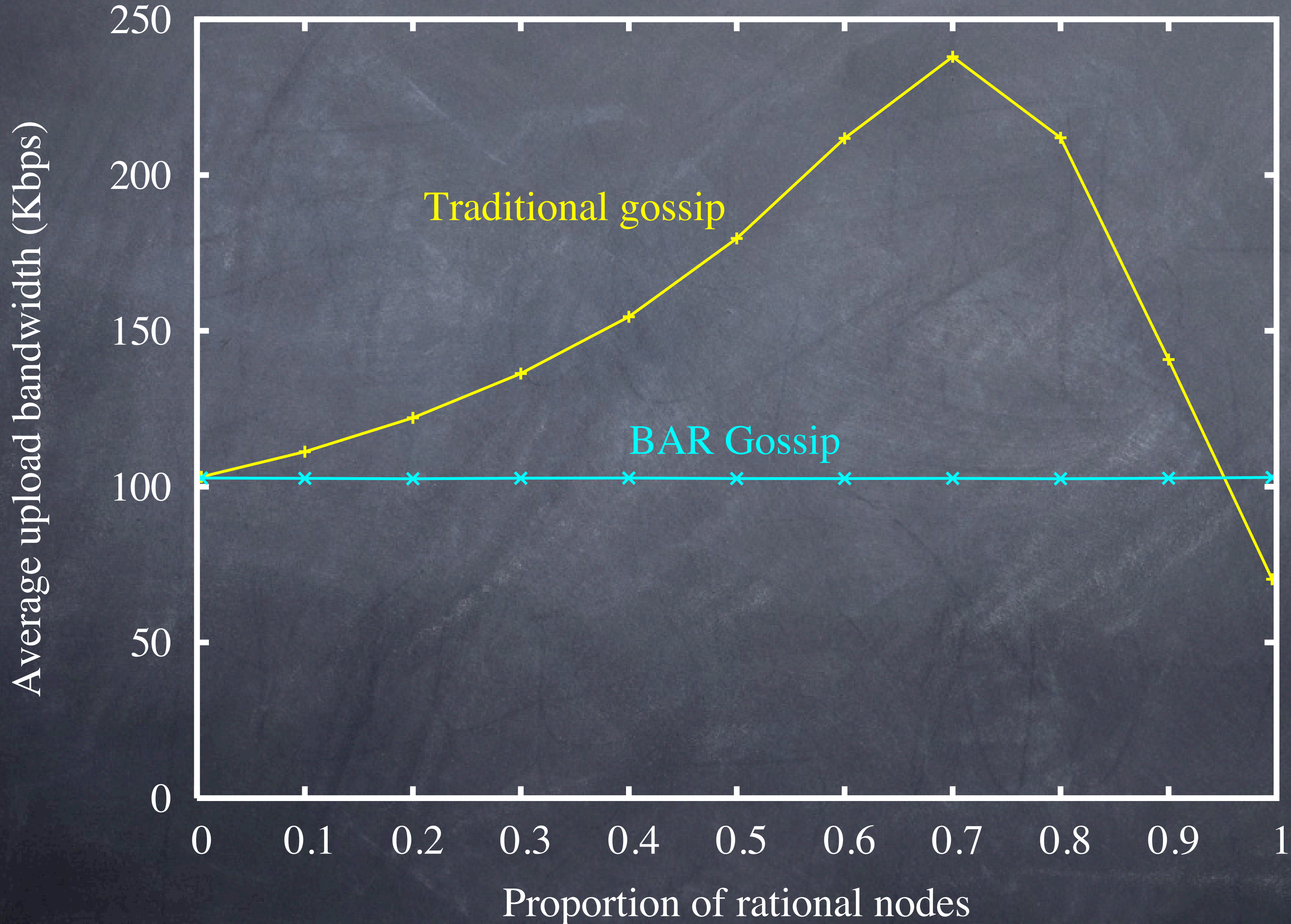
BAR Gossip
Traditional gossip
Probability of receiving an update
Proportion of rational nodes
0
0.2
0.4
0.6
0.8
1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

BAR Gossip



250
200
150
100
50
0
Average upload bandwidth (Kbps)
Traditional gossip
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Proportion of rational nodes

BAR Gossip



250
200
150
100
50
0
Average upload bandwidth (Kbps)
Traditional gossip
BAR Gossip
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Proportion of rational nodes

The Setup

Application

- Altruistic broadcaster
- BAR clients
- Static membership
- Full membership list
- Updates useful for finite time

The Setup

Application

- Altruistic broadcaster
- BAR clients
- Static membership
- Full membership list
- Updates useful for finite time

Crypto

- Public/Private key pairs
- Notation: $\langle M \rangle_A$

The Setup

Application

- Altruistic broadcaster
- BAR clients
- Static membership
- Full membership list
- Updates useful for finite time

Crypto

- Public/Private key pairs
- Notation: $\langle M \rangle_A$

Incentive Structure

- Benefit: playing updates
- Cost: bandwidth
- No long-term reputations

BAR Gossip Overview

Balanced Exchange

Optimistic Push

BAR Gossip Overview

Balanced Exchange

In each round:

- Select partner
- Exchange histories
- Trade equal number of updates

Optimistic Push

BAR Gossip Overview

Balanced Exchange

Optimistic Push

In each round:

- Select partner
- Exchange histories
- Trade equal number of updates

Little help to peers that fall behind

BAR Gossip Overview

Balanced Exchange

In each round:

- Select partner
- Exchange histories
- Trade equal number of updates

Little help to peers that fall behind

Optimistic Push

In each round:

- Select partners
- Exchange histories

BAR Gossip Overview

Balanced Exchange

In each round:

- Select partner
- Exchange histories
- Trade equal number of updates

Little help to peers that fall behind

Optimistic Push

In each round:

- Select partner
- Exchange histories
- Trade possibly unequal numbers of updates

Safety net for lagging peers

Balanced Exchange

In each round

- Select a partner
- Exchange histories
- Trade equal number of updates

Balanced Exchange

In each round

- Select a partner
- Exchange histories
- Trade equal number of updates

Balanced Exchange

In each round

- Select a partner
- Exchange histories
- Trade equal number of updates
  - fair exchange

Balanced Exchange

In each round

- Select a partner
- Exchange histories
- Trade equal number of updates
  - fair exchange is impossible without a trusted third party

B. Garbinato and I. Rickebusch. Impossibility results on fair exchange. Tech. Rep. DOP-20051122, Université de Lausanne, Distributed Object Programming Lab.

Balanced Exchange

In each round

- Select a partner
- Exchange histories
- Trade equal number of updates
  - fair exchange is <u>impossible</u> without a trusted third party
  - so we settle for fair enough!

Balanced Exchange

In each round

- Select a partner
- Exchange histories
- Trade equal number of updates
  - Exchange briefcases
  - Exchange keys

fair enough exchange

Design principles

Design principles

- Restrict choice

Design principles

- Restrict choice
  - Eliminate non-determinism

Design principles

- Restrict choice
  - Eliminate non-determinism
  - Evict provably deviant peers

Design principles

- Restrict choice
  - Eliminate non-determinism
  - Evict provably deviant peers
- Delay gratification

Design principles

- Restrict choice
  - Eliminate non-determinism
  - Evict provably deviant peers
- Delay gratification
  - Postpone payoff to keep rational peers engaged

The Intuition

- Restrict choice
    - Eliminate non-determinism
    - Evict provably deviant peers
- Delay gratification

Select D

Select C

Select B

The Intuition

Restrict choice
- Eliminate non-determinism
- Evict provably deviant peers

Delay gratification

Select D

Select C

Select B

The Intuition

Restrict choice

- Eliminate non-determinism
- Evict provably deviant peers

Delay gratification

Select D

Select C

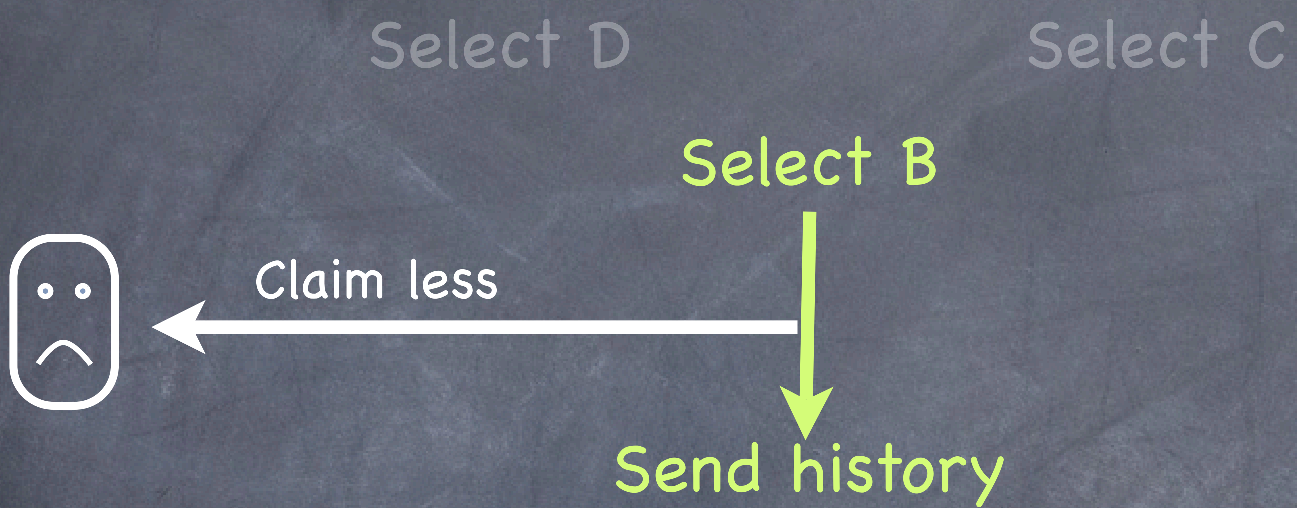
Select B

↓

Send history

The Intuition

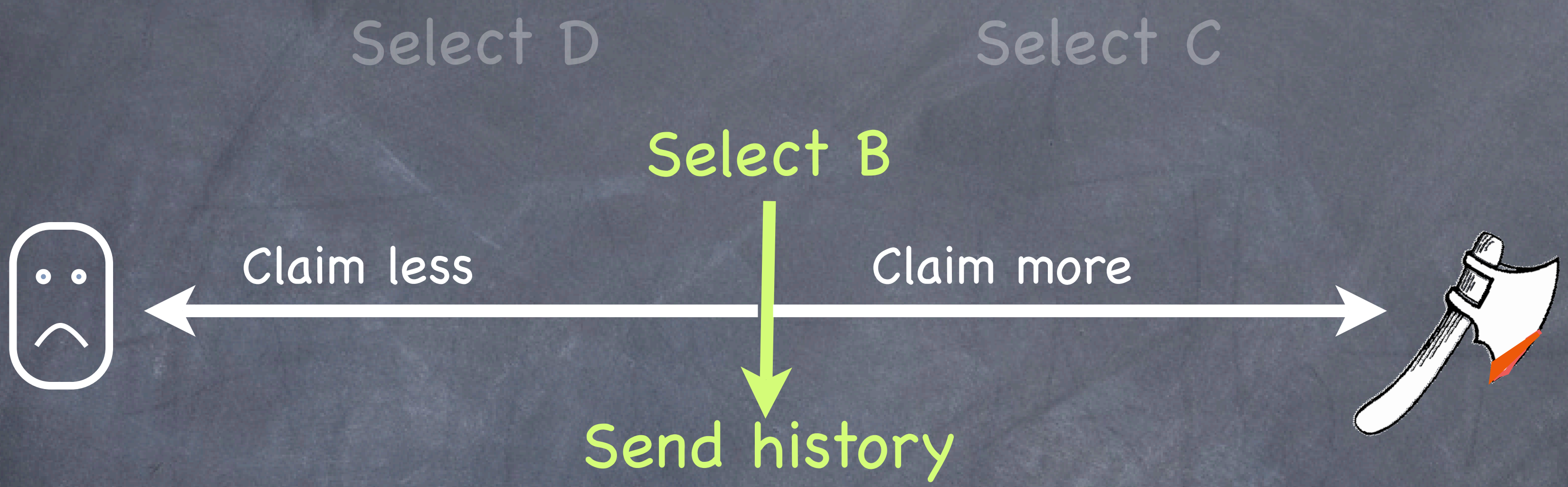
- Restrict choice
  - Eliminate non-determinism
  - Evict provably deviant peers
- Delay gratification



The Intuition

Restrict choice
- Eliminate non-determinism
- Evict provably deviant peers

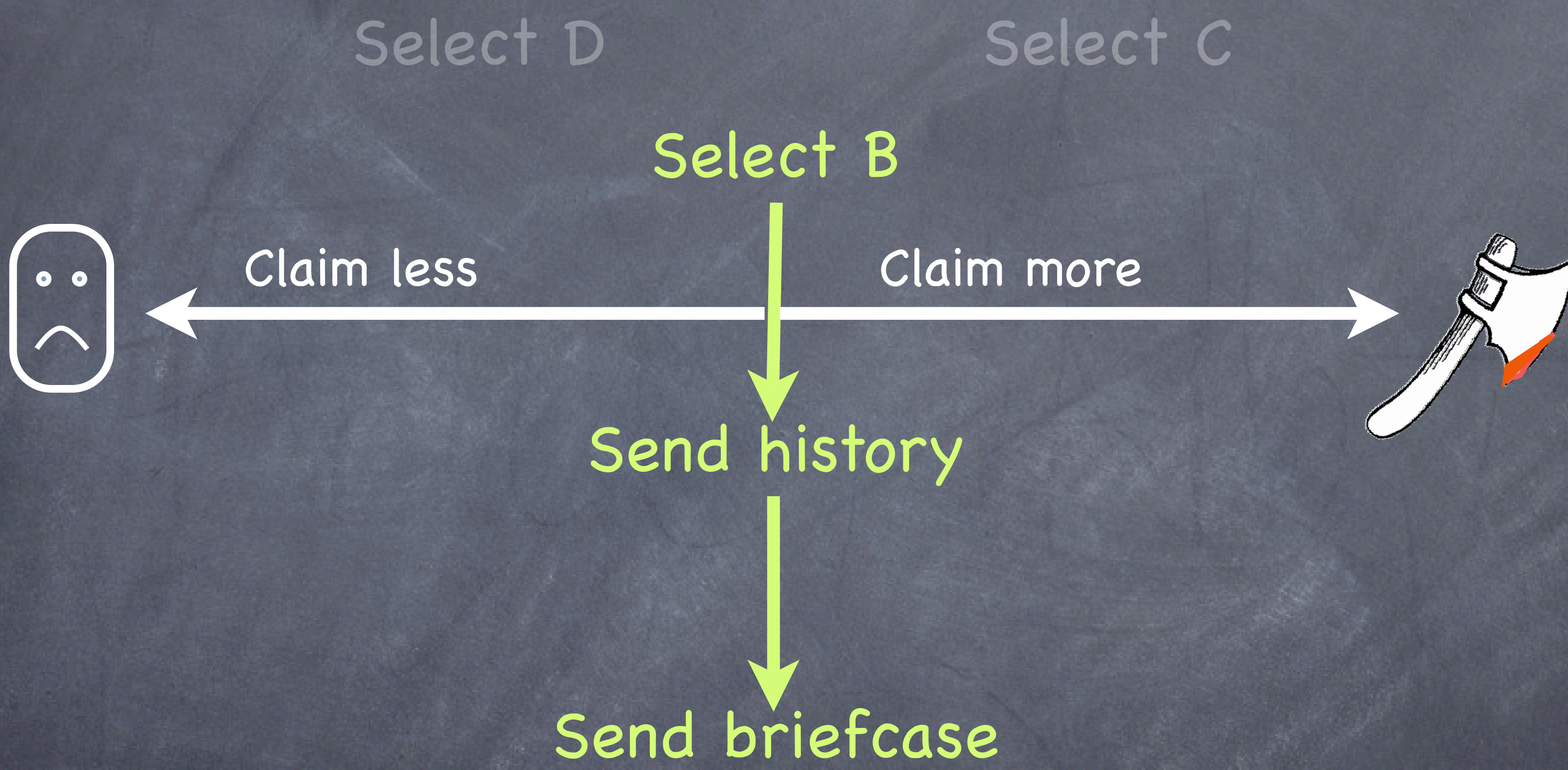
Delay gratification



The Intuition

Restrict choice
- Eliminate non-determinism
- Evict provably deviant peers

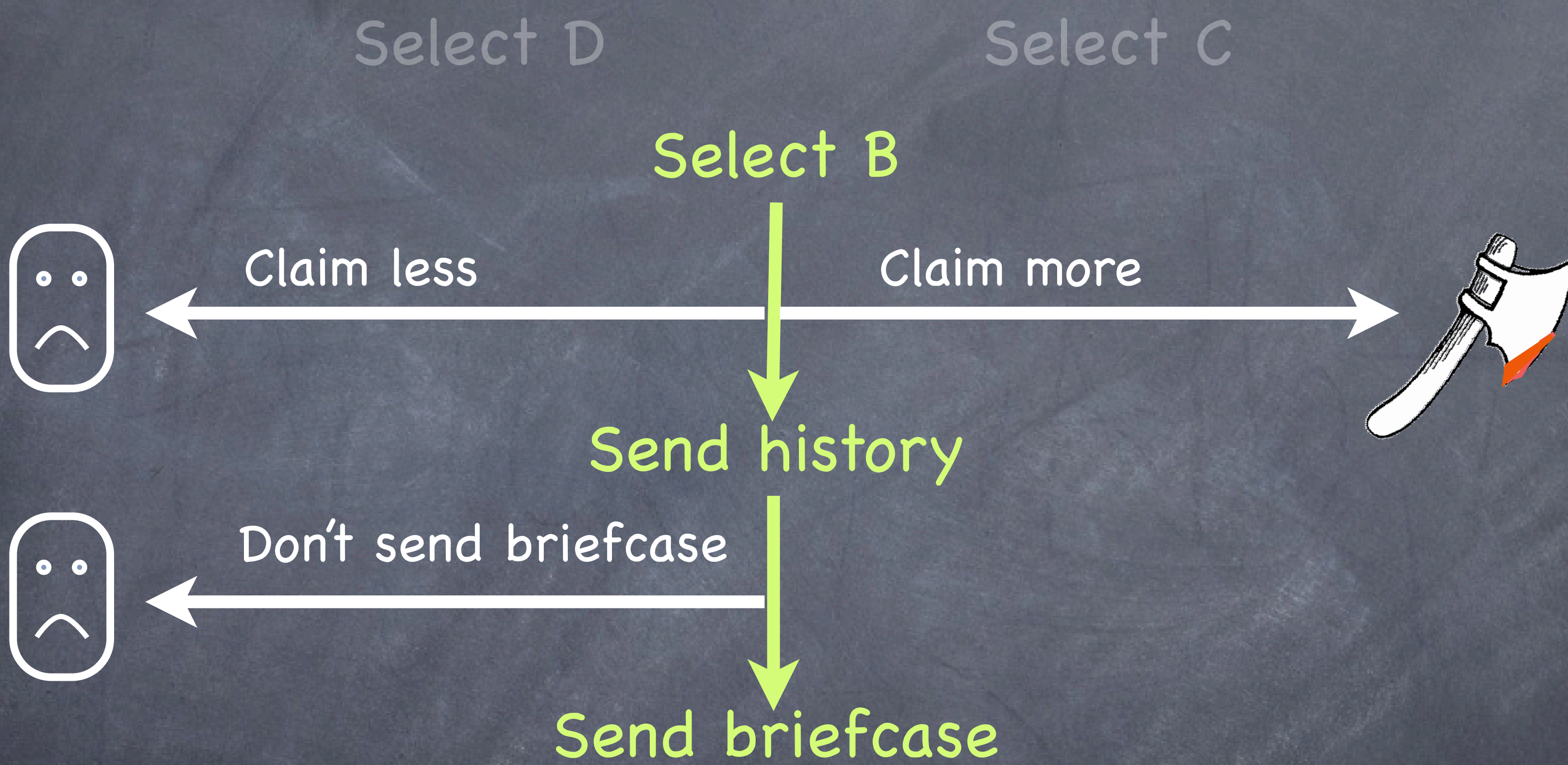
Delay gratification



The Intuition

Restrict choice
- Eliminate non-determinism
- Evict provably deviant peers

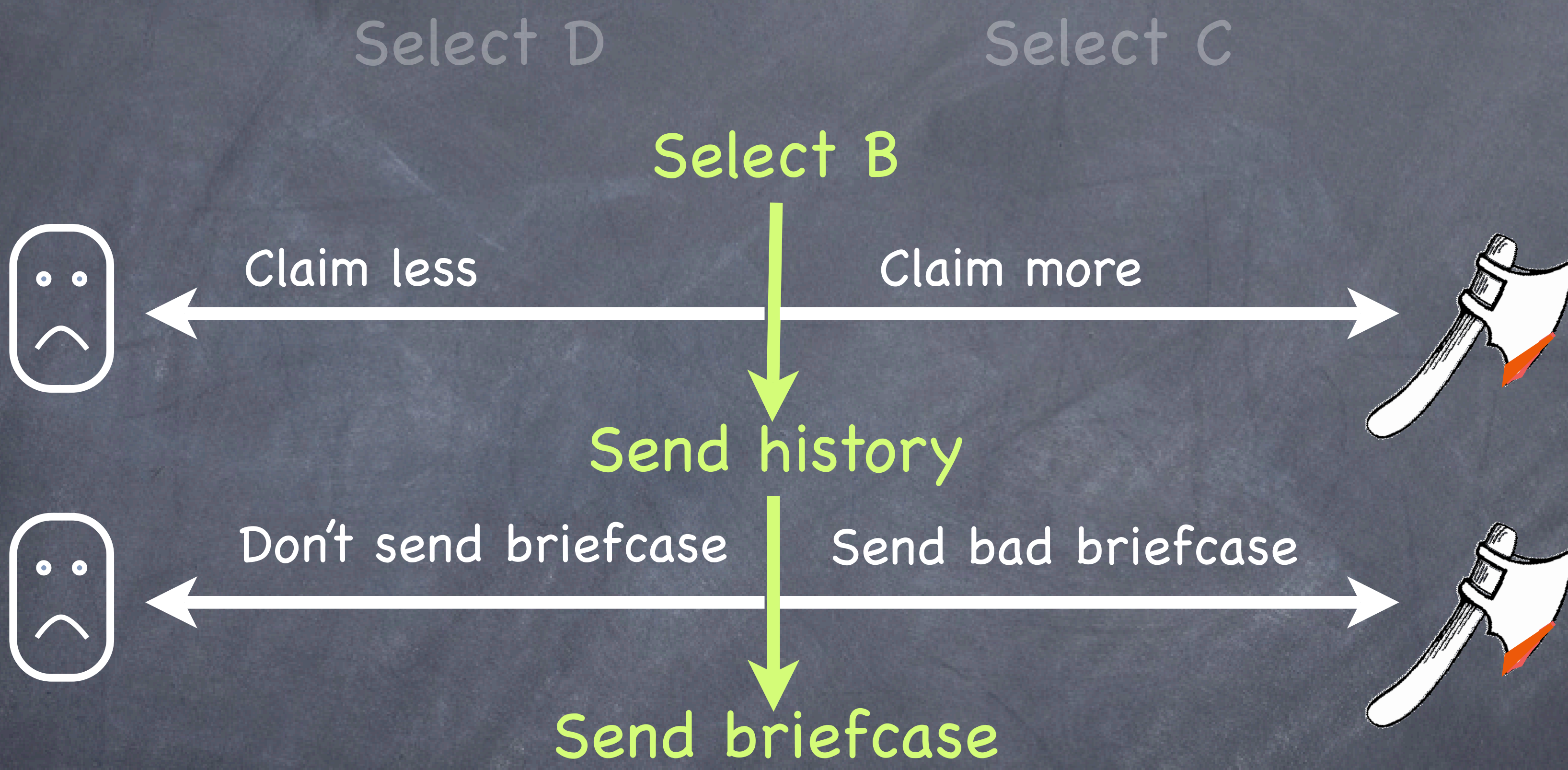
Delay gratification



The Intuition

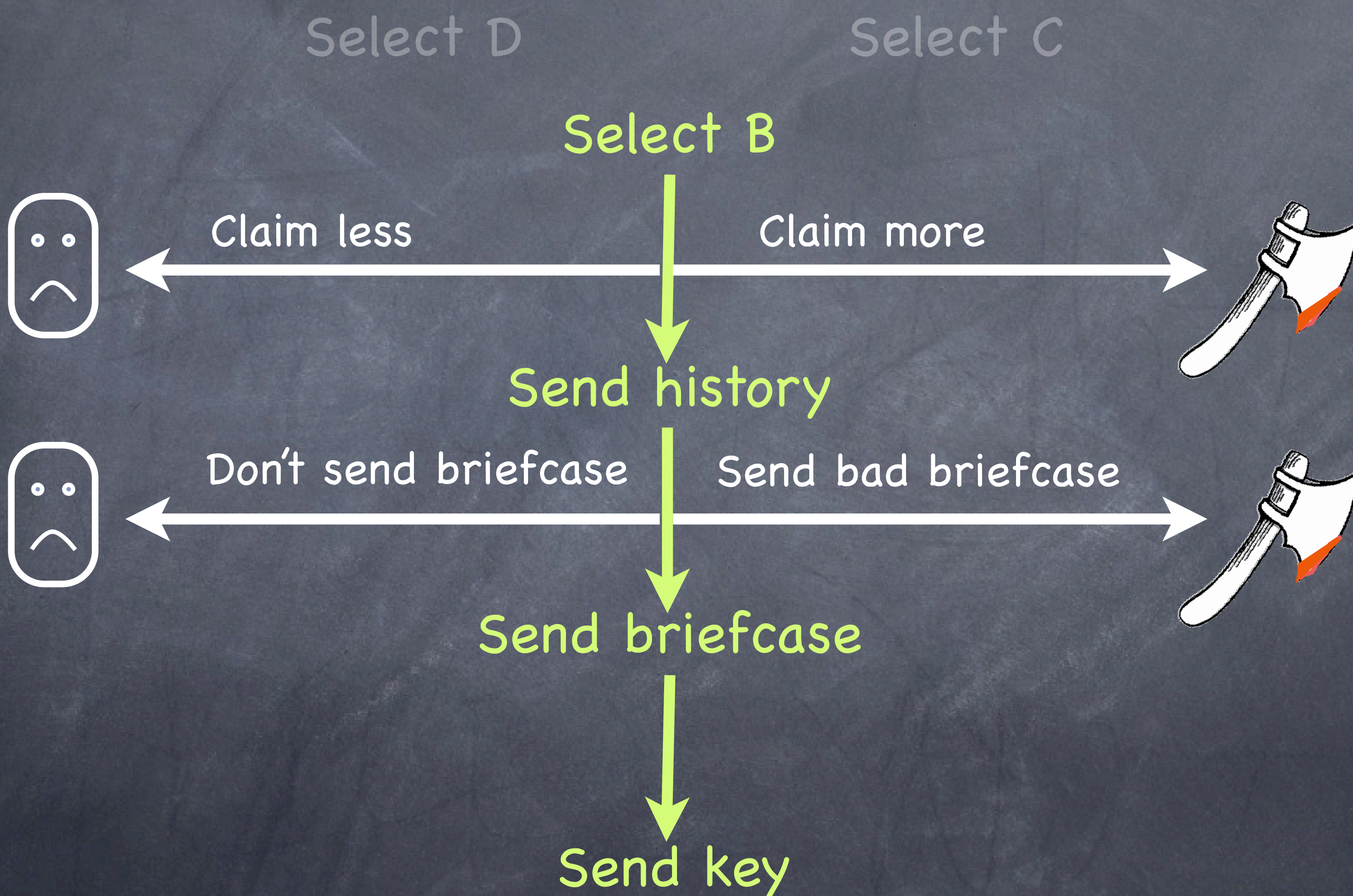
Restrict choice
- Eliminate non-determinism
- Evict provably deviant peers

Delay gratification



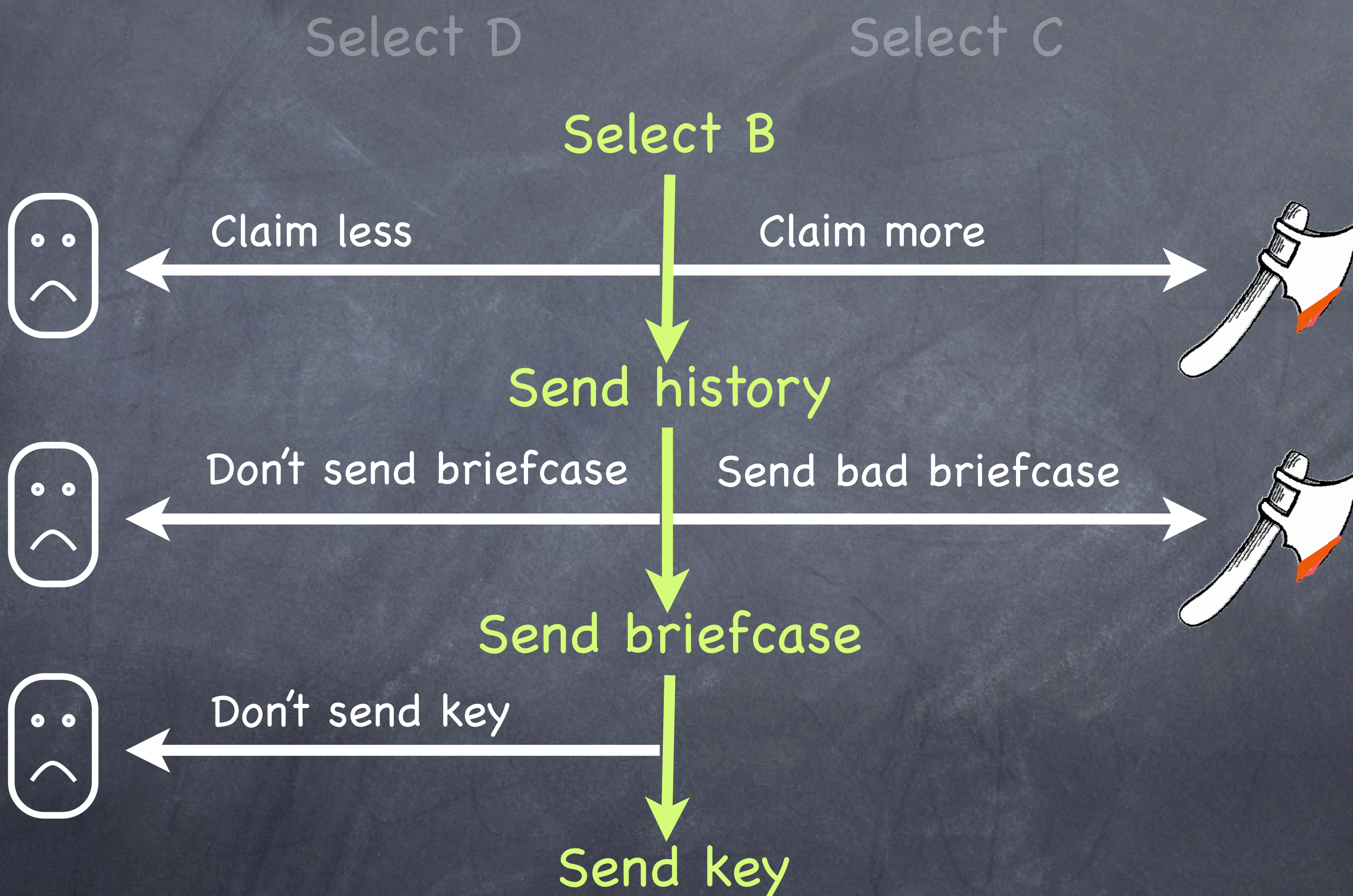
The Intuition

- Restrict choice
  - Eliminate non-determinism
  - Evict provably deviant peers
- Delay gratification



The Intuition

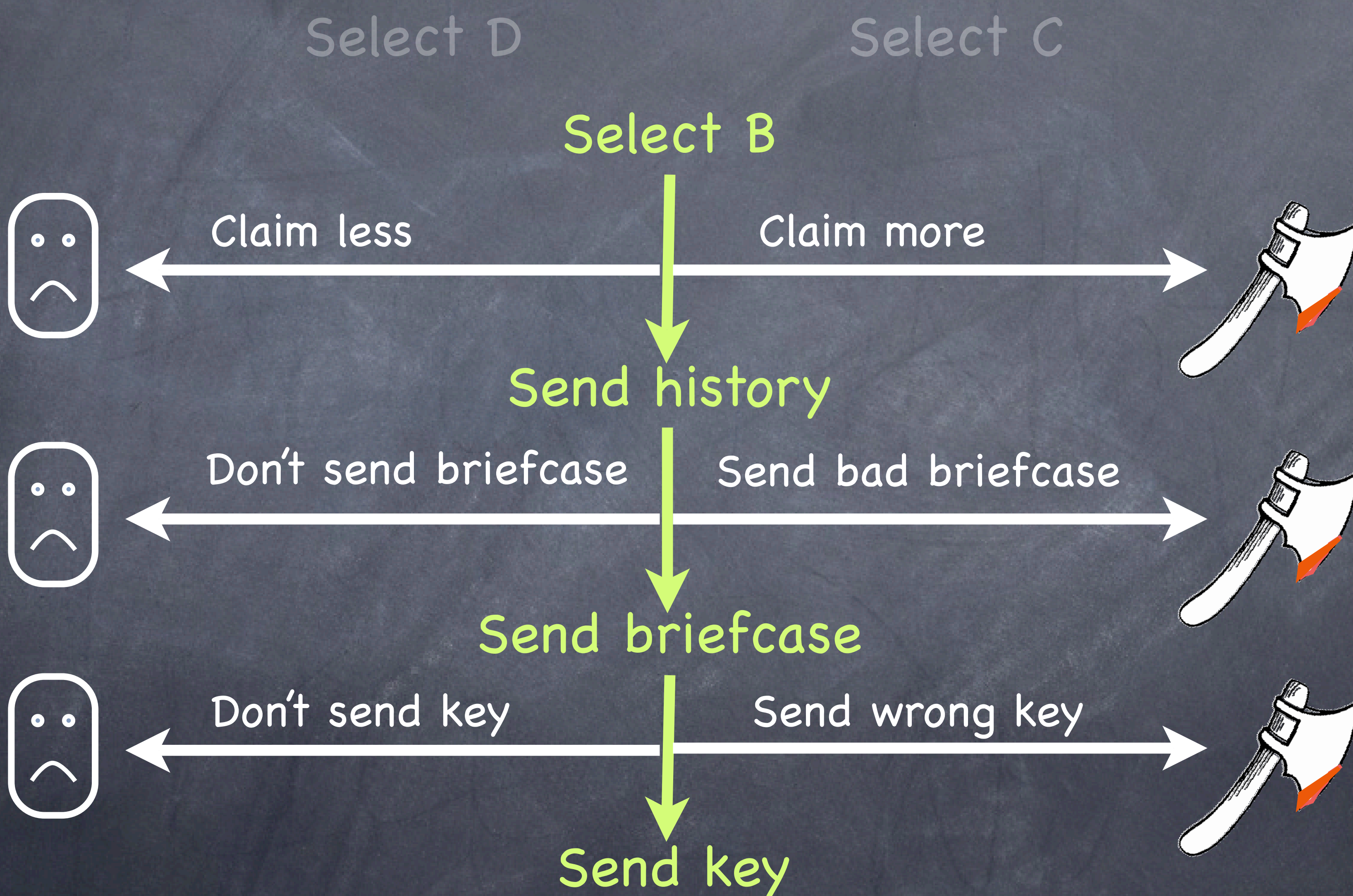
- Restrict choice
  - Eliminate non-determinism
  - Evict provably deviant peers
- Delay gratification



The Intuition

Restrict choice
- Eliminate non-determinism
- Evict provably deviant peers

Delay gratification



Balanced Exchange is a Nash Equilibrium

Theorem: A balanced exchange is incentive compatible for strategies that maximize the number of useful updates received in that exchange

Balanced Exchange is a Nash Equilibrium

Theorem: A balanced exchange is incentive compatible for strategies that maximize the number of useful updates received in that exchange

- Partner selection
- History exchange
- Briefcase exchange
- Key exchange

Balanced Exchange is a Nash Equilibrium

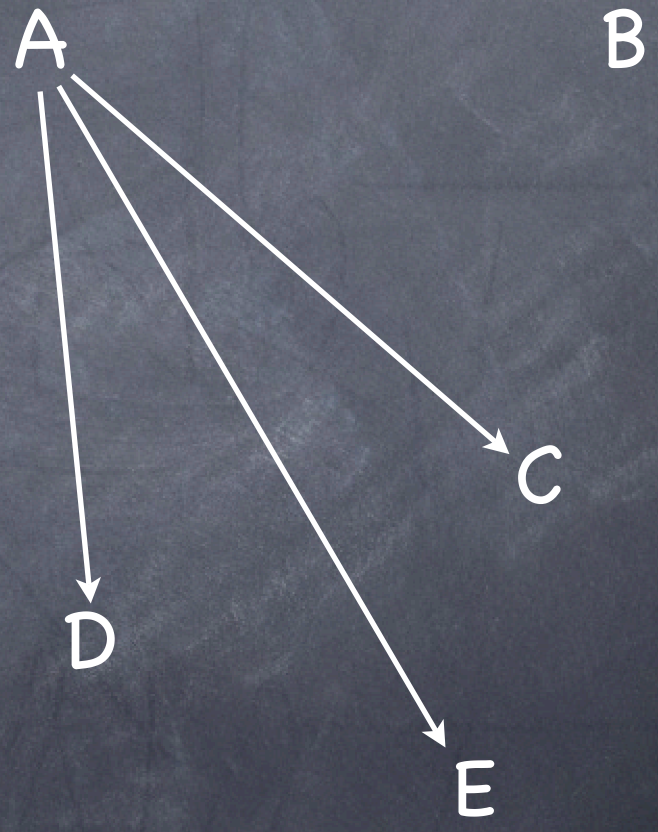
Theorem: A balanced exchange is incentive compatible for strategies that maximize the number of useful updates received in that exchange

- Partner selection
- History exchange
- Briefcase exchange
- Key exchange

Incentive compatible

Partner Selection

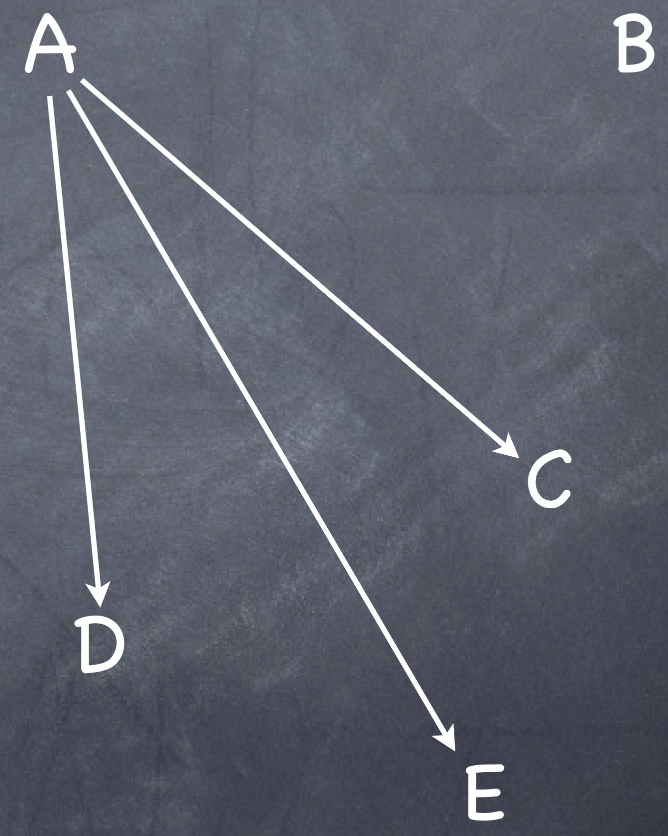
Q: How do we limit a peer to one uniformly selected partner per round?



Partner Selection

Q: How do we limit a peer to one uniformly selected partner per round?

A: Verifiable pseudo-random partner selection

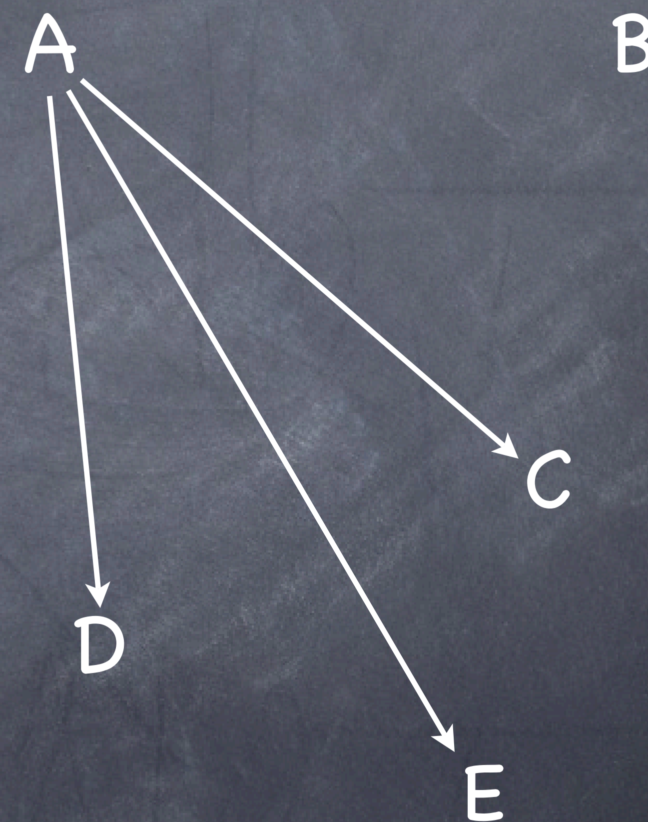


Partner Selection

Q: How do we limit a peer to one uniformly selected partner per round?

A: Verifiable pseudo-random partner selection

- A's PRNG seed in round $r : \langle r \rangle_A$

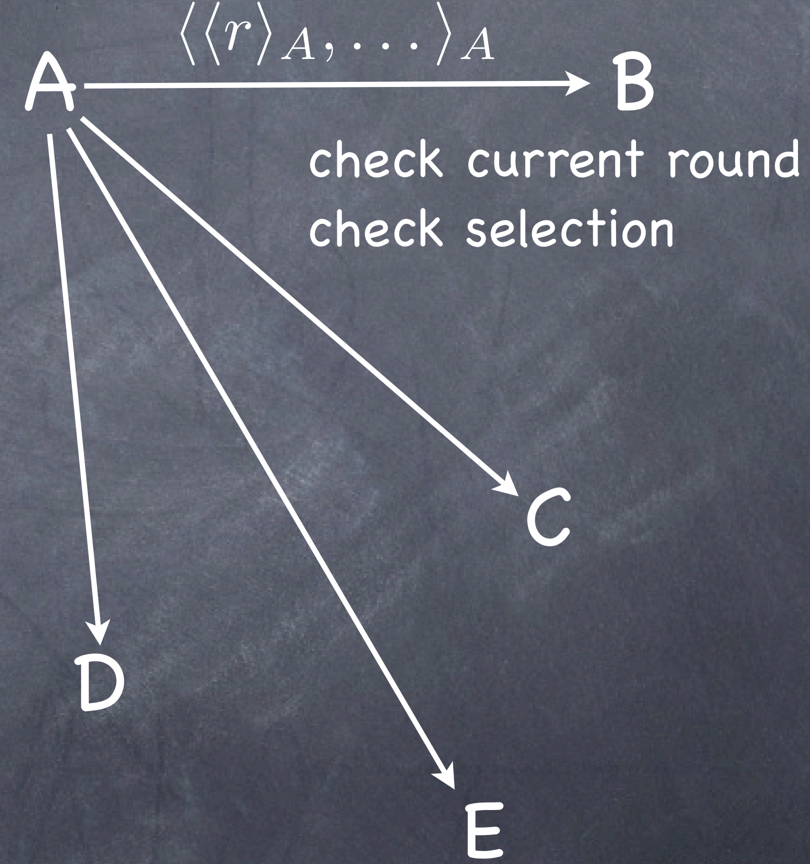


Partner Selection

Q: How do we limit a peer to one uniformly selected partner per round?

A: Verifiable pseudo-random partner selection

- A's PRNG seed in round $r : \langle r \rangle_A$

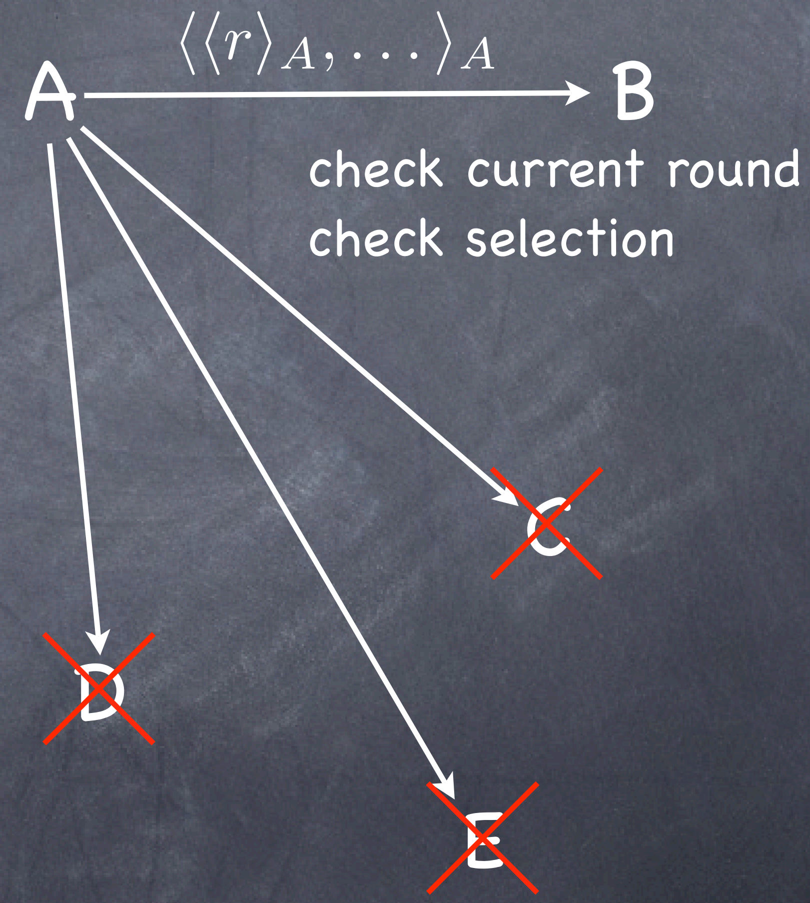


Partner Selection

Q: How do we limit a peer to one uniformly selected partner per round?

A: Verifiable pseudo-random partner selection

- A's PRNG seed in round r : $\langle r \rangle_A$

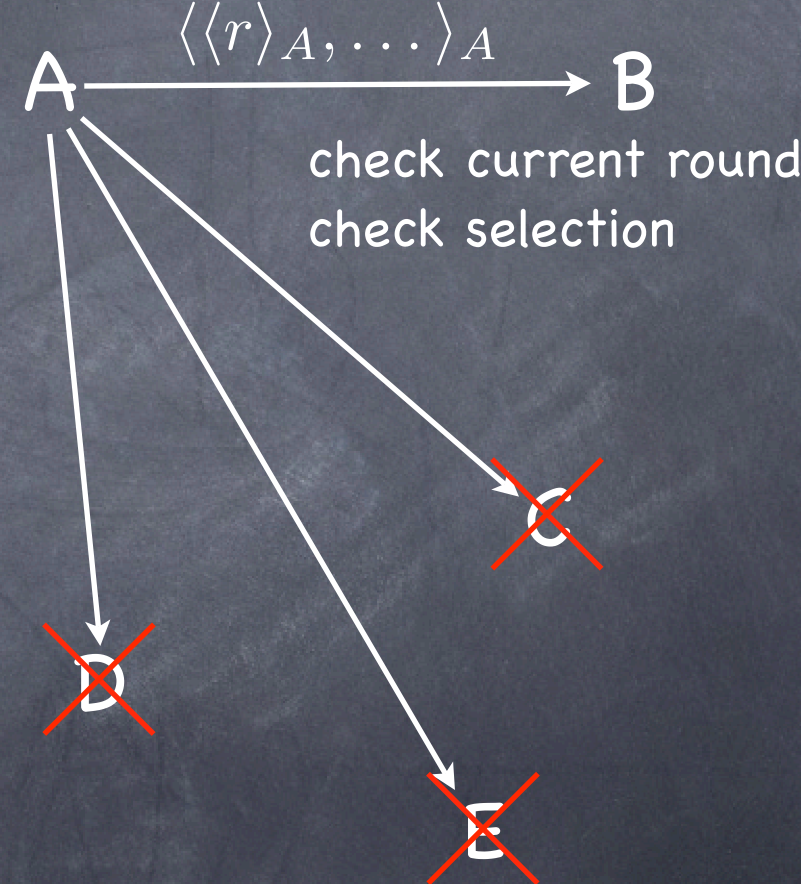


Partner Selection

Q: How do we limit a peer to one uniformly selected partner per round?

A: Verifiable pseudo-random partner selection

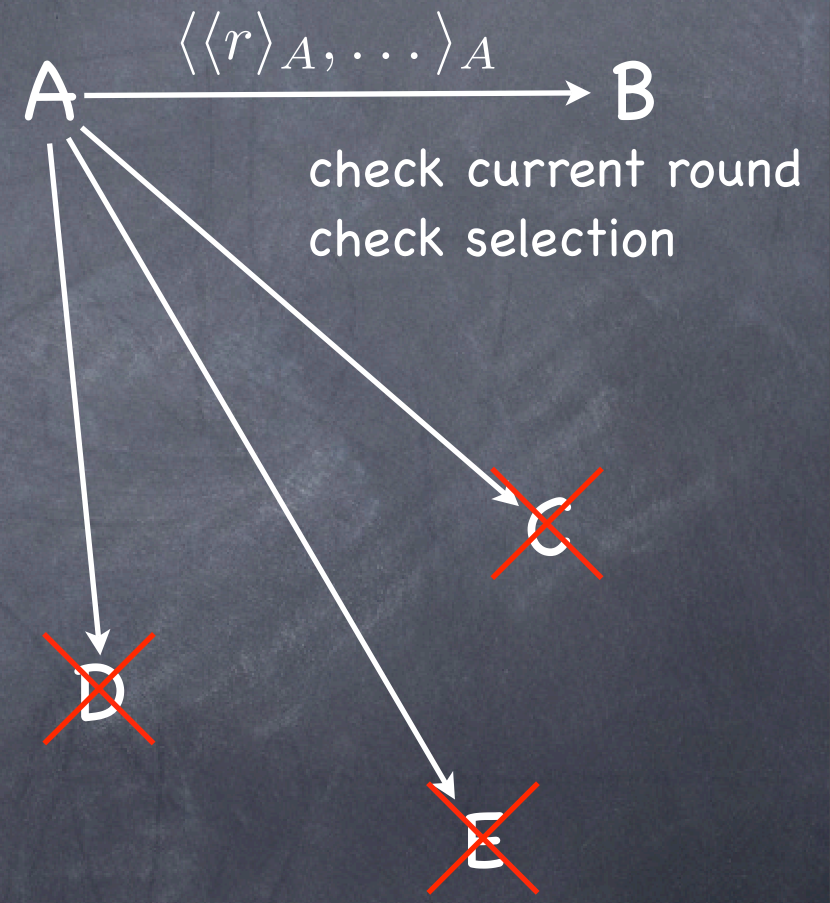
- A's PRNG seed in round $r : \langle r \rangle_A$
  - Eliminates non-determinism



Partner Selection

Q: How do we limit a peer to one uniformly selected partner per round?

A: Verifiable pseudo-random partner selection



- A's PRNG seed in round r : $\langle r\rangle_A$
  - Eliminates non-determinism
  - Retains strength of randomness:
    - ✓ uniform selection of partners
    - ✓ unpredictability

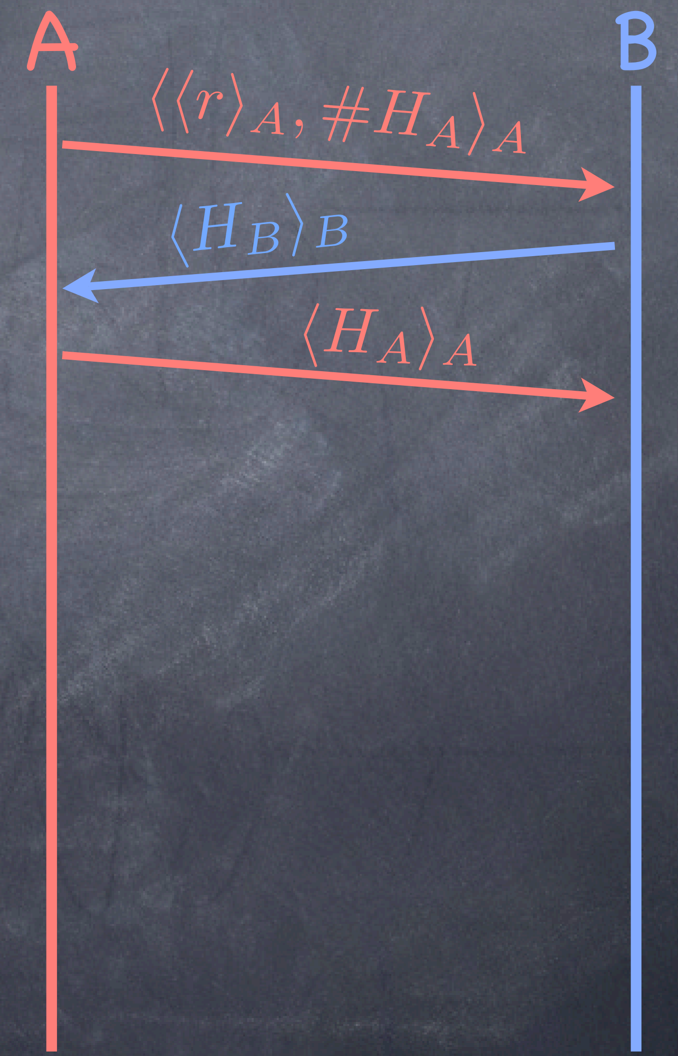
History Exchange

Q: How do we handle a client lying about its history?

History Exchange

Q: How do we handle a client lying about its history?

A: Client commits to a history before discovering partner's history

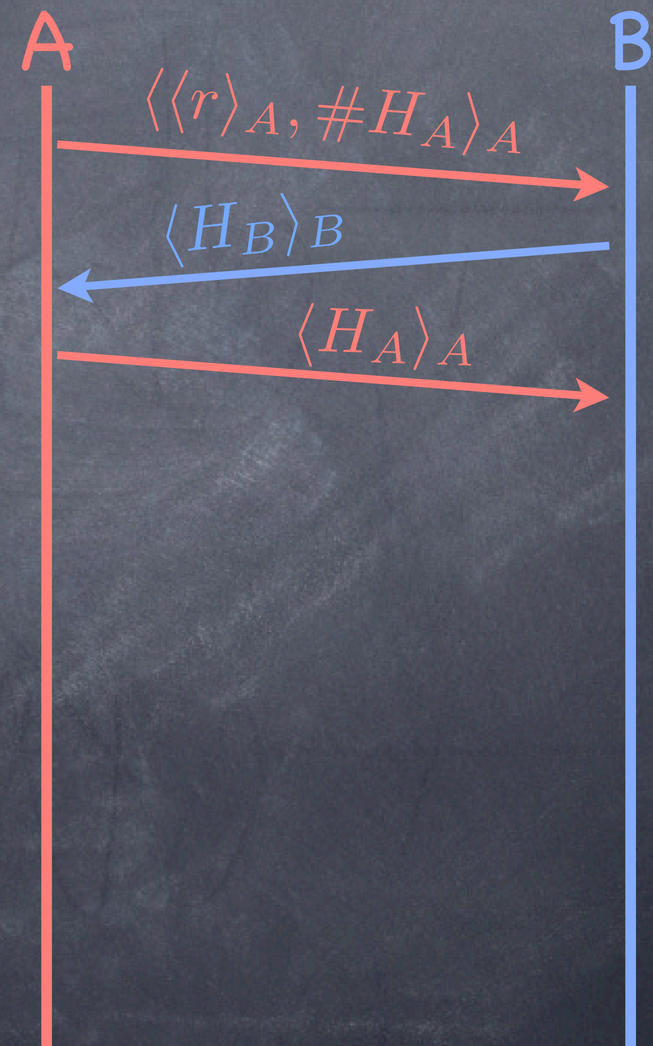


History Exchange

Q: How do we handle a client lying about its history?

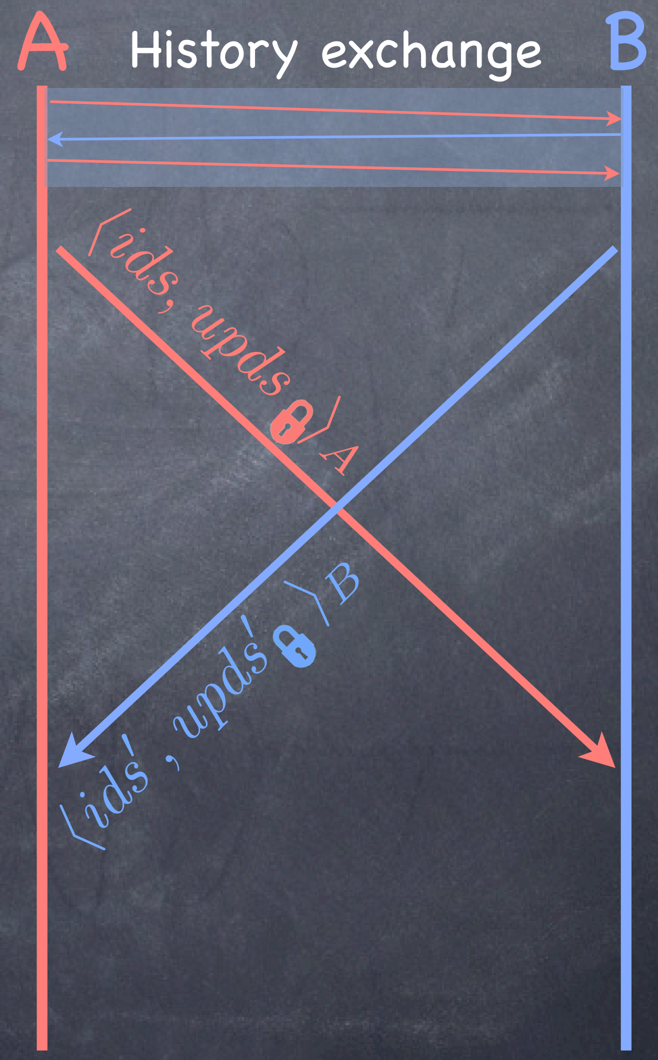
A: Client commits to a history before discovering partner's history

- Under-reporting decreases number of useful updates exchanged
- Over-reporting risks eviction



Briefcase Exchange

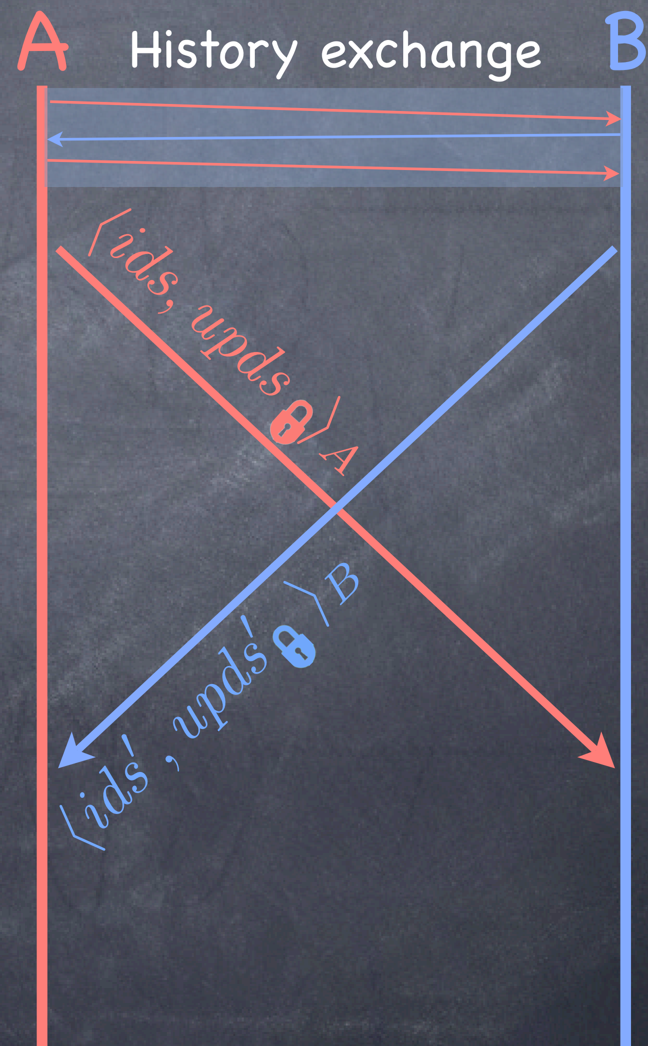
Q: How do we encourage a rational client to send a briefcase?



Briefcase Exchange

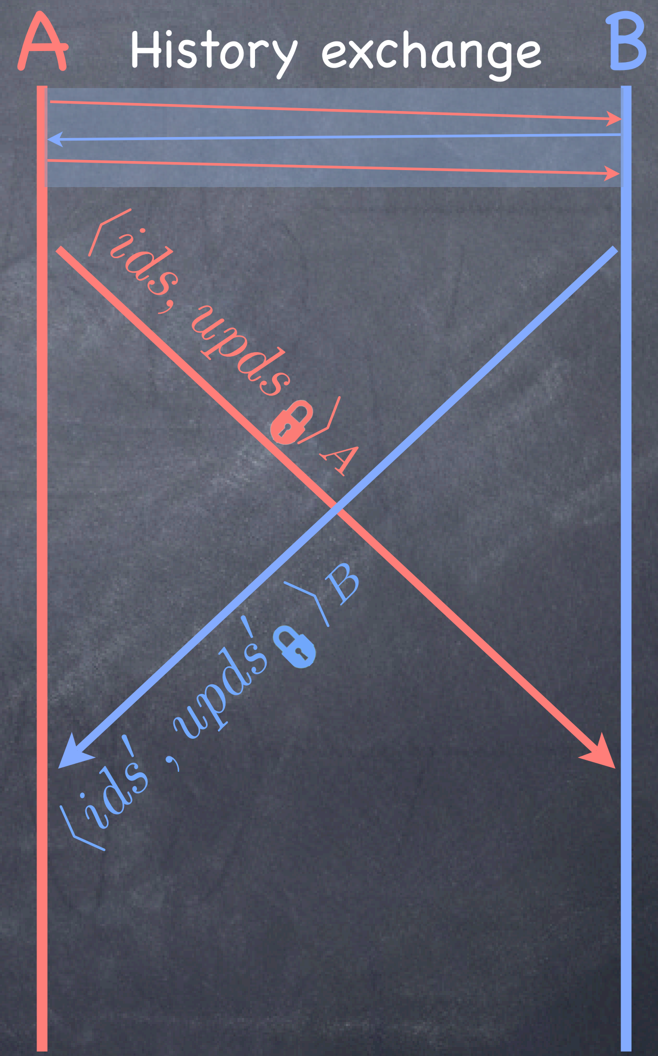
Q: How do we encourage a rational client to send a briefcase?

A: Client gives key only after swapping briefcases



Valid Briefcase Exchange

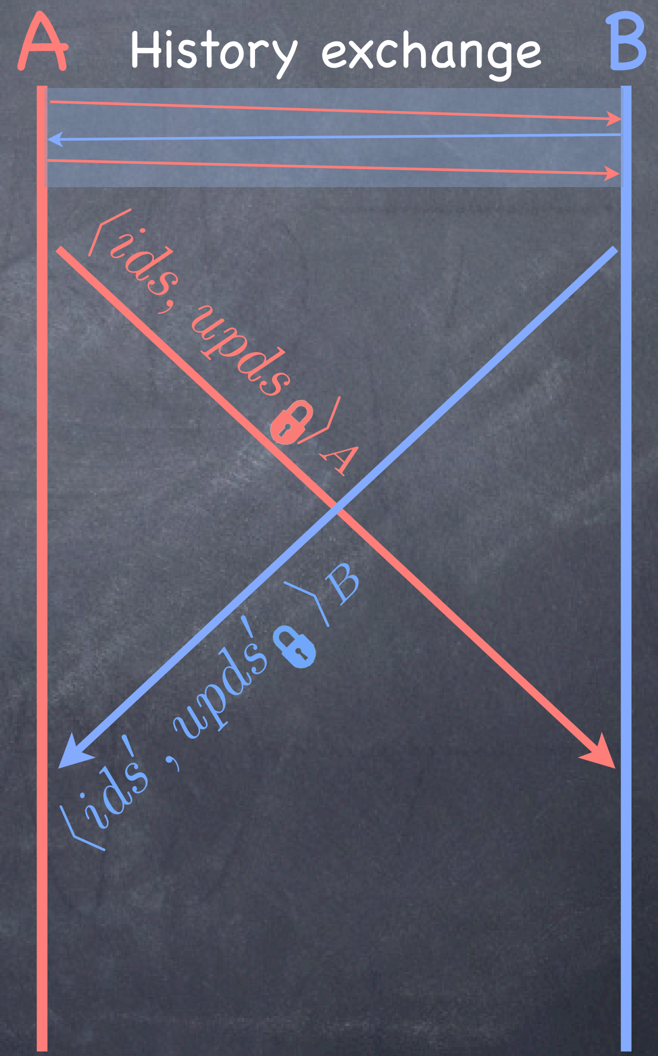
Q: How do we encourage a rational client to send only appropriate briefcases?



Valid Briefcase Exchange

Q: How do we encourage a rational client to send only appropriate briefcases?

A: Hold client accountable for contents of briefcase



Valid Briefcase Exchange

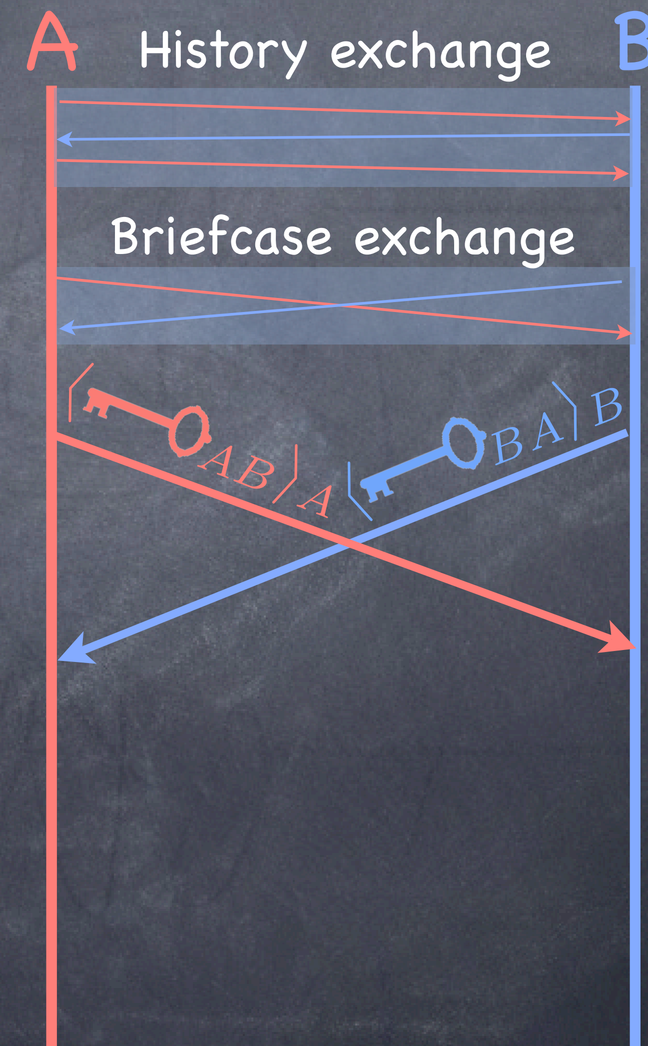
Q: How do we encourage a rational client to send only appropriate briefcases?

A: Hold client accountable for contents of briefcase

- Briefcase contains encrypted updates and ids of updates
- Inconsistencies risk eviction
- Decryption key is reproducible by broadcaster

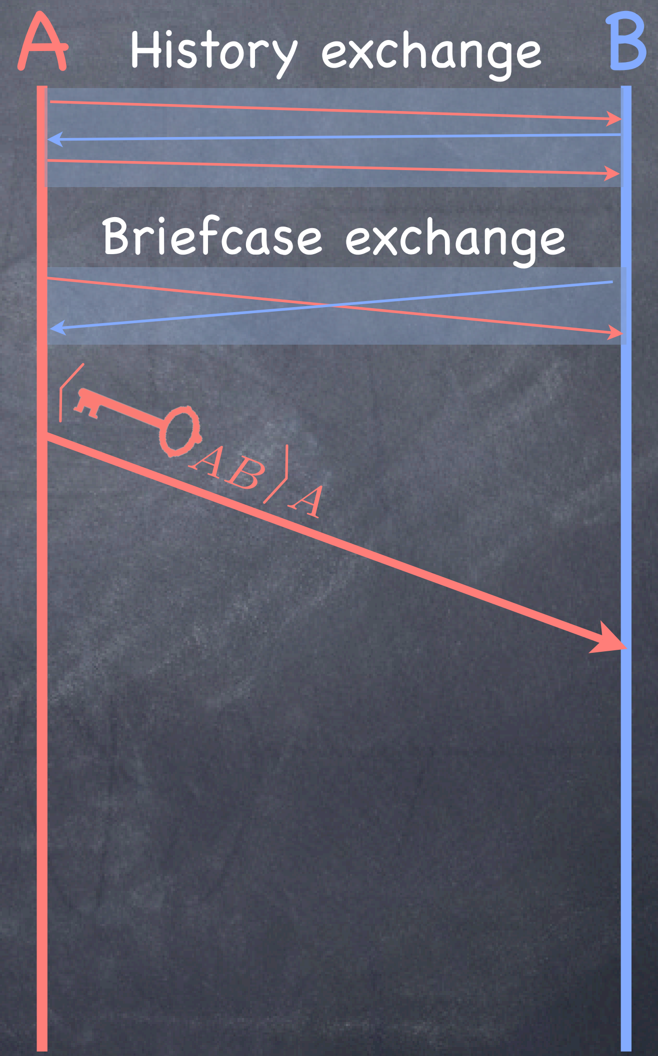
Key Exchange

Q: How do we encourage a rational client to send the appropriate key?



Key Exchange

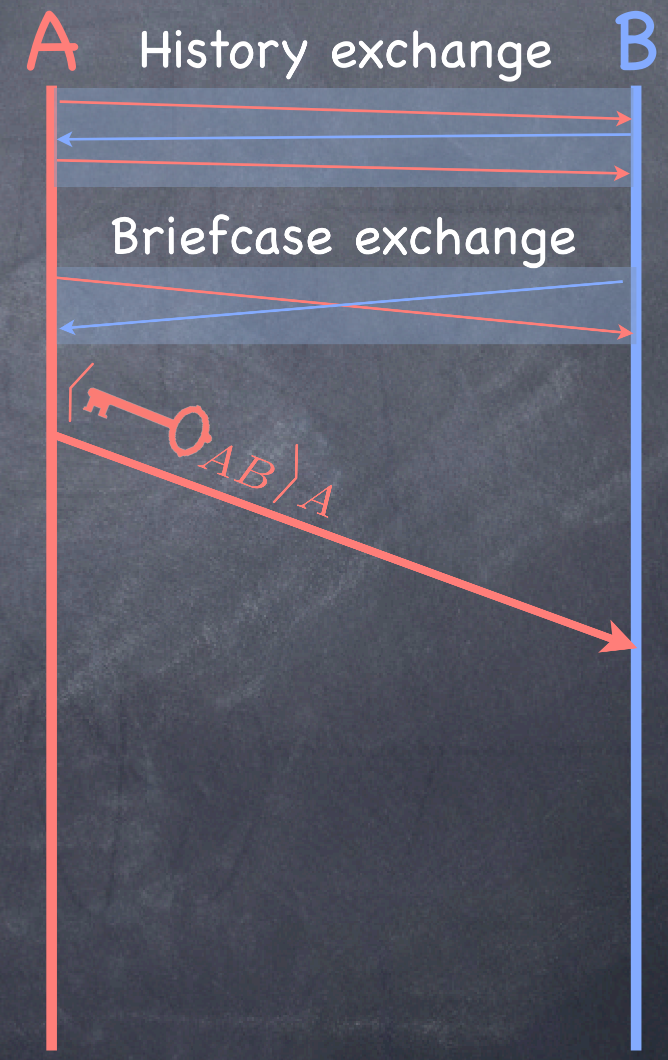
Q: How do we encourage a rational client to send the appropriate key?



Key Exchange

Q: How do we encourage a rational client to send the appropriate key?

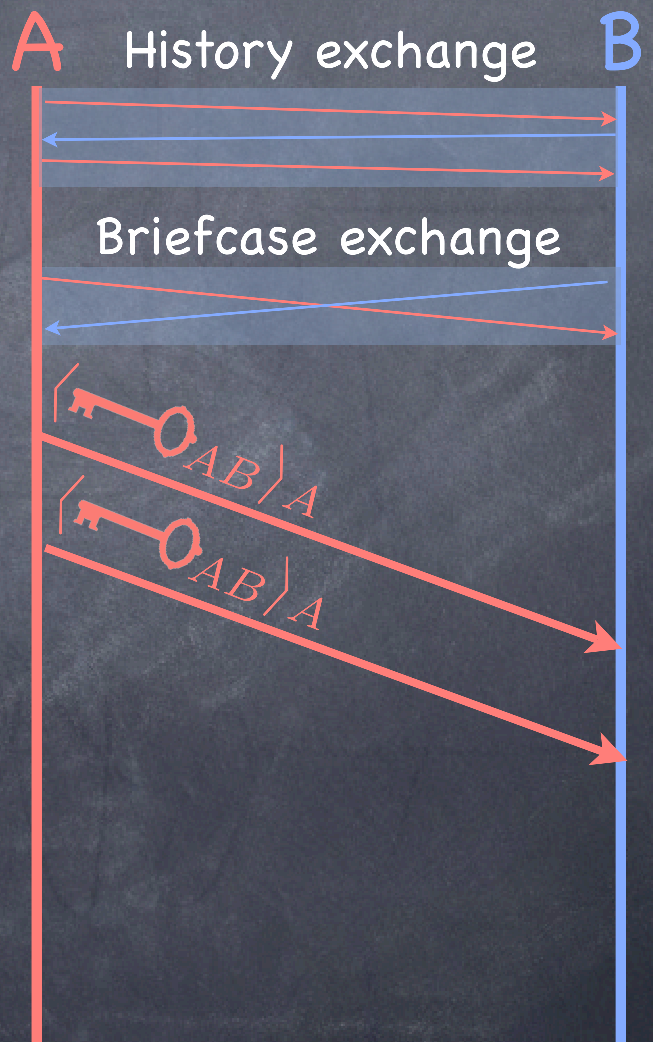
A: Repeated Key Requests



Key Exchange

Q: How do we encourage a rational client to send the appropriate key?

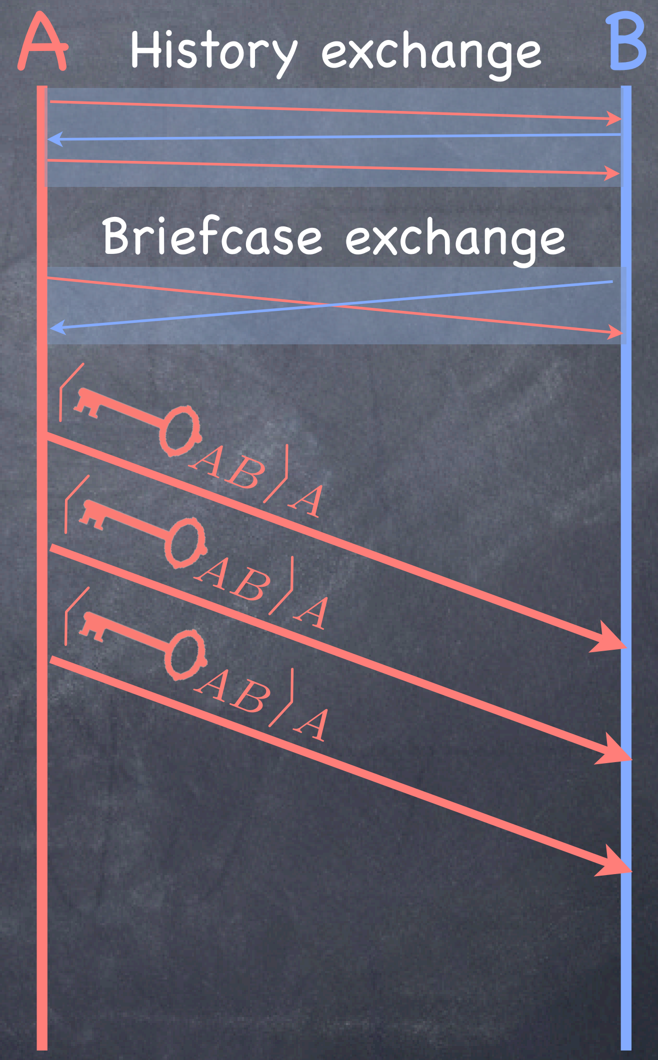
A: Repeated Key Requests



Key Exchange

Q: How do we encourage a rational client to send the appropriate key?

A: Repeated Key Requests

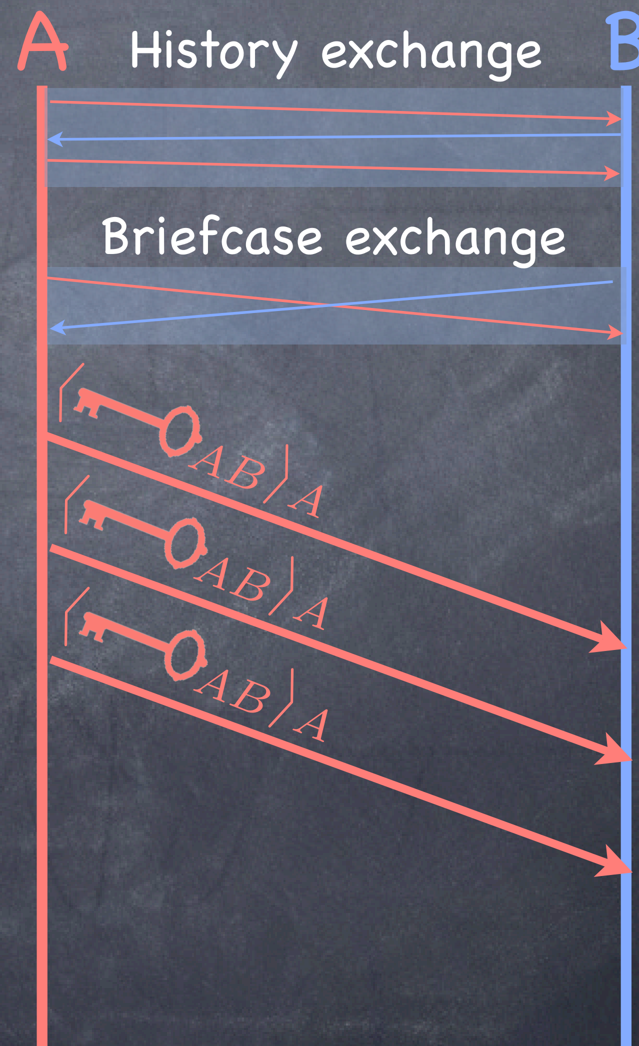


Key Exchange

Q: How do we encourage a rational client to send the appropriate key?

A: Repeated Key Requests

- Rational client minimizes cost by sending key

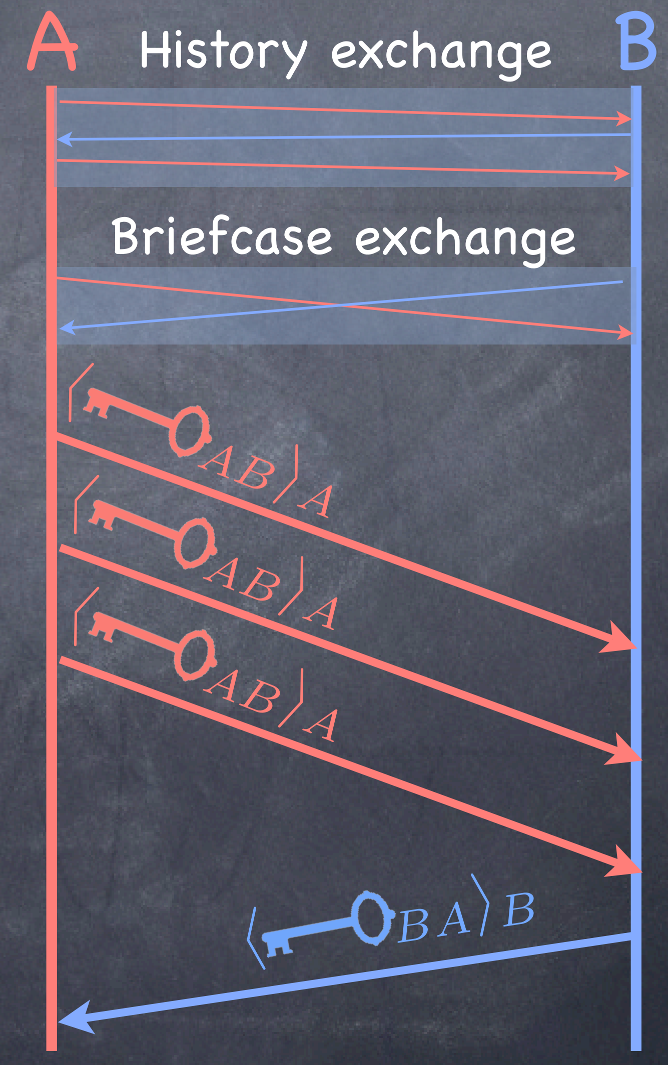


Key Exchange

Q: How do we encourage a rational client to send the appropriate key?

A: Repeated Key Requests

- Rational client minimizes cost by sending key

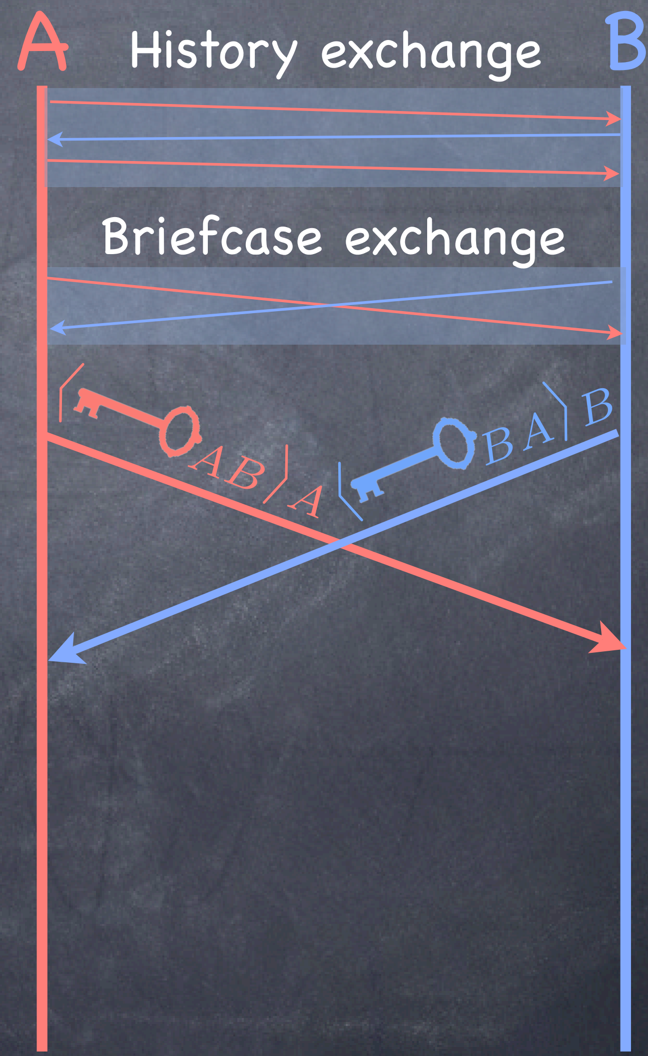


Key Exchange

Q: How do we encourage a rational client to send the appropriate key?

A: Repeated Key Requests

- Rational client minimizes cost by sending key
- Rational client proactively sends key

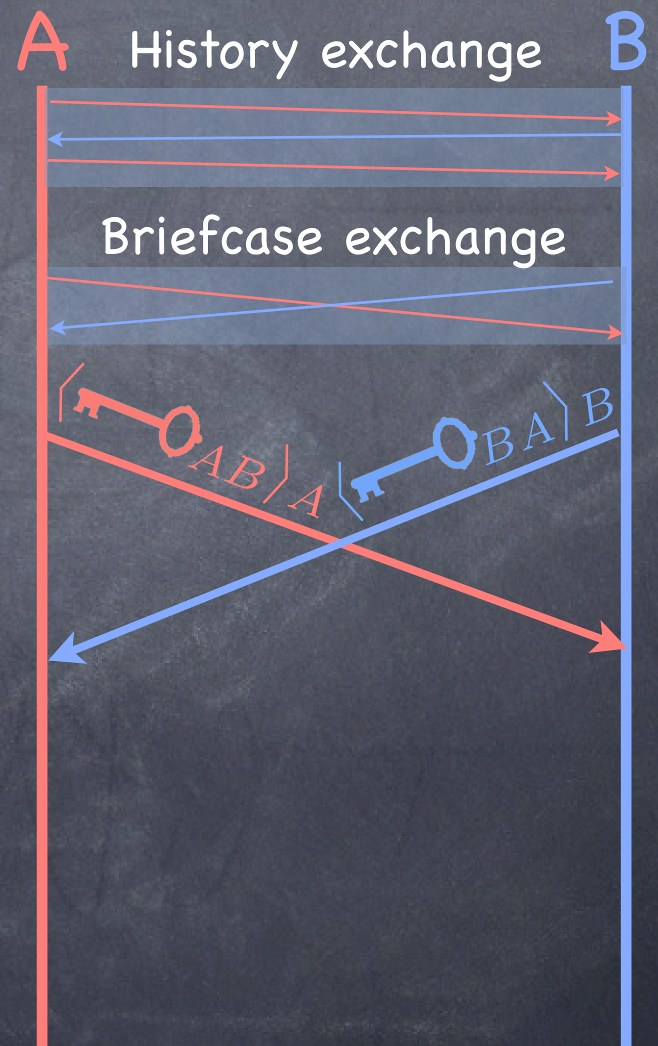


Key Exchange

Q: How do we encourage a rational client to send the appropriate key?

A: Repeated Key Requests

- Rational client minimizes cost by sending key
- Rational client proactively sends key
- Hold client accountable for key responses



BAR Gossip Overview

Balanced Exchange

In each round:

- Select partner
- Exchange histories
- Trade equal number of updates

Incentive compatible!

Optimistic Push

In each round:

- Select partners
- Exchange histories
- Trade possibly unequal numbers of updates

Safety net for lagging peers

BAR Gossip Overview

Balanced Exchange

In each round:

- Select partner
- Exchange histories
- Trade equal number of updates

Incentive compatible!

Optimistic Push

In each round:

- Select partner
- Exchange histories
- Trade possibly unequal numbers of updates

Safety net for lagging peers

BAR Gossip Overview

Balanced Exchange

In each round:

- Select partner
- Exchange histories
- Trade equal number of updates

Incentive compatible!

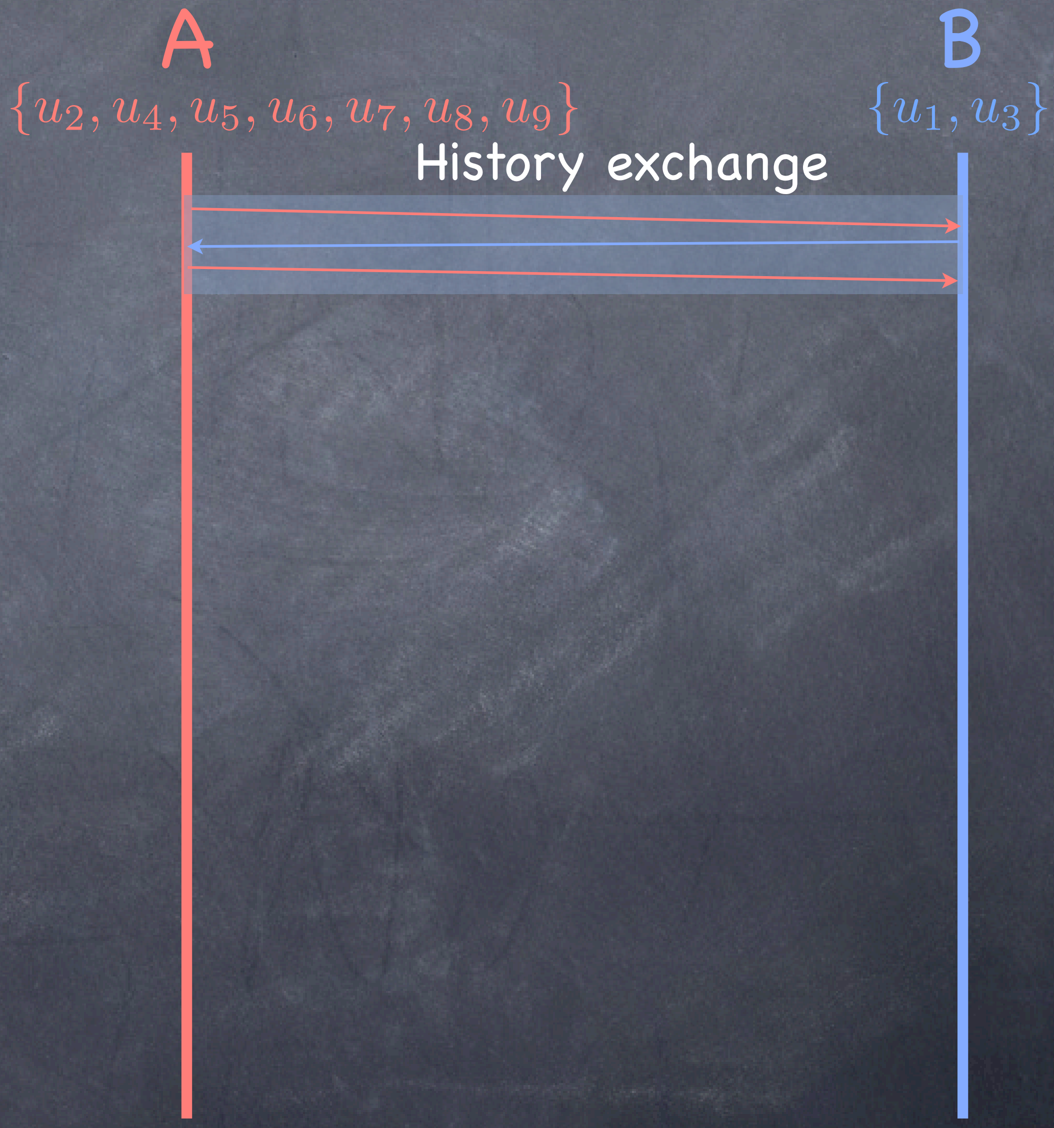
Optimistic Push

In each round:

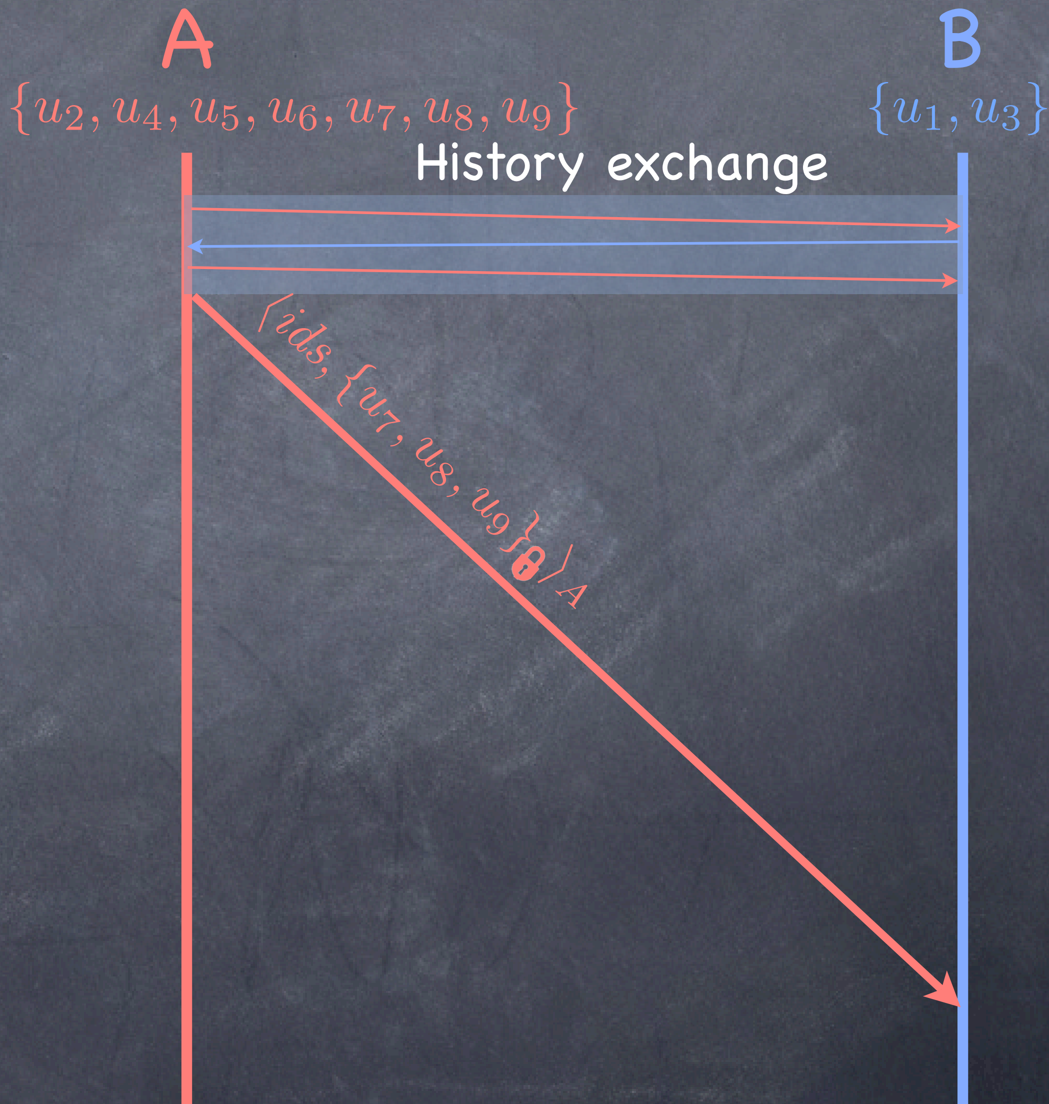
- Select partner
- Exchange histories
- Trade possibly unequal numbers of updates

Safety net for lagging peers

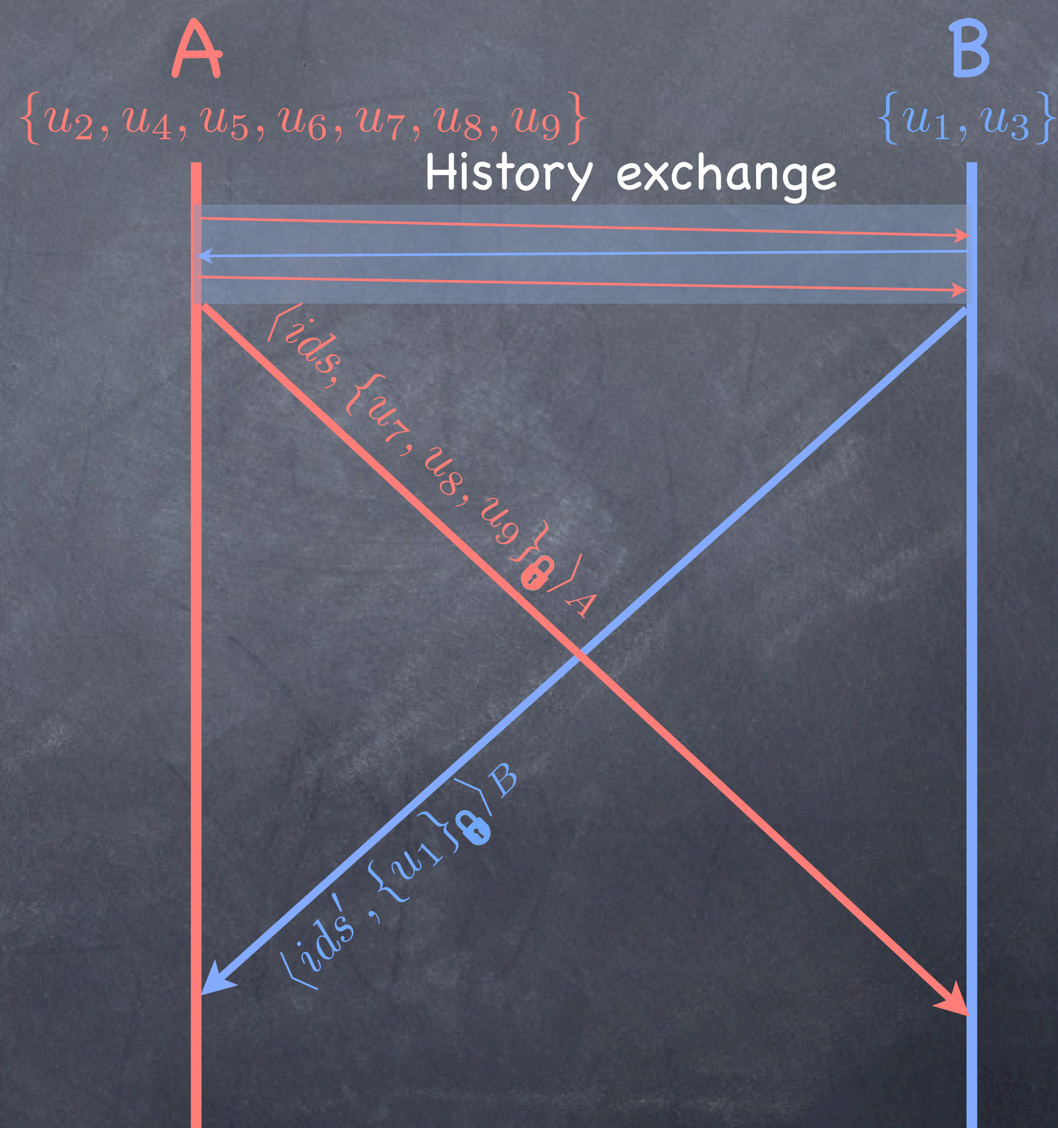
Optimistic Push



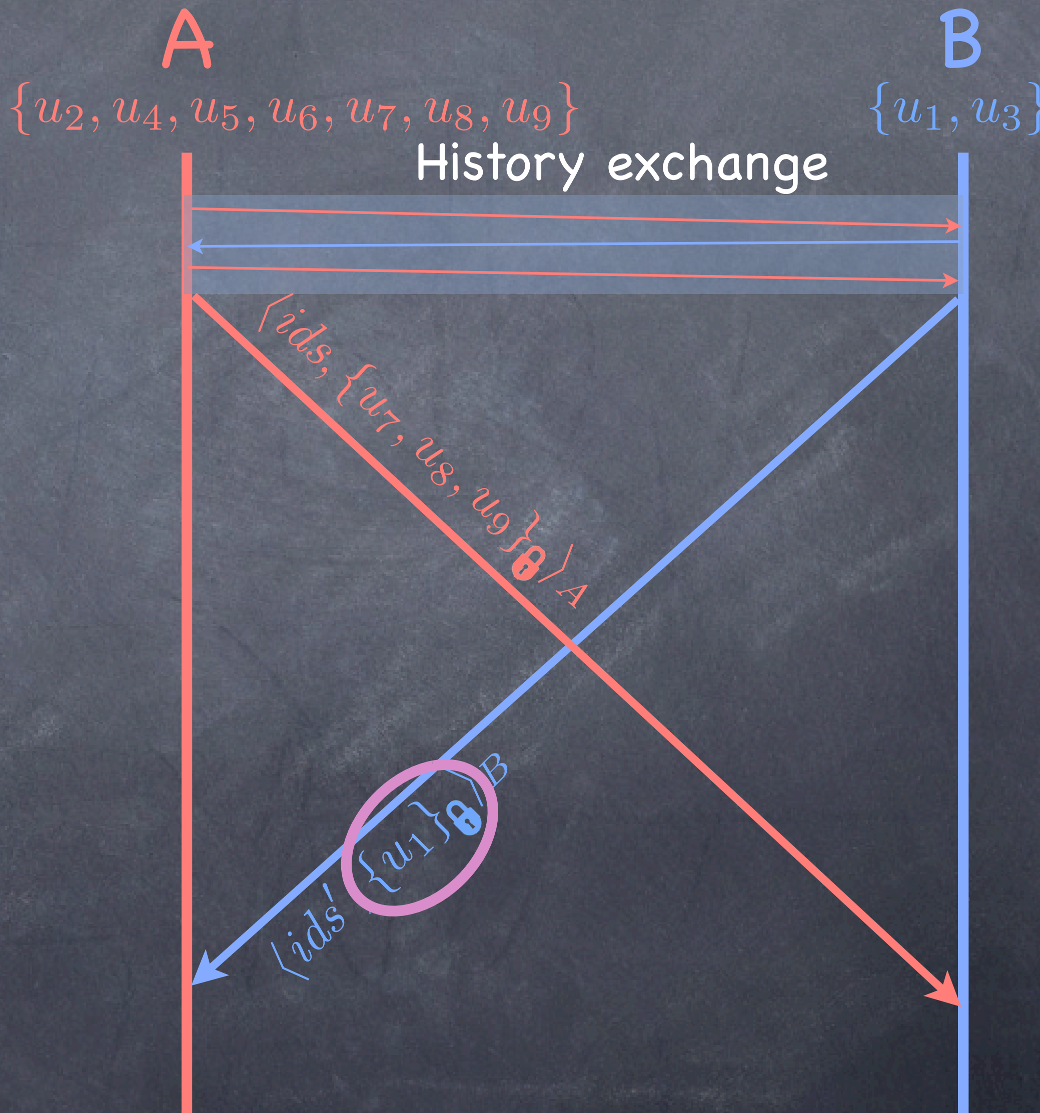
Optimistic Push



Optimistic Push

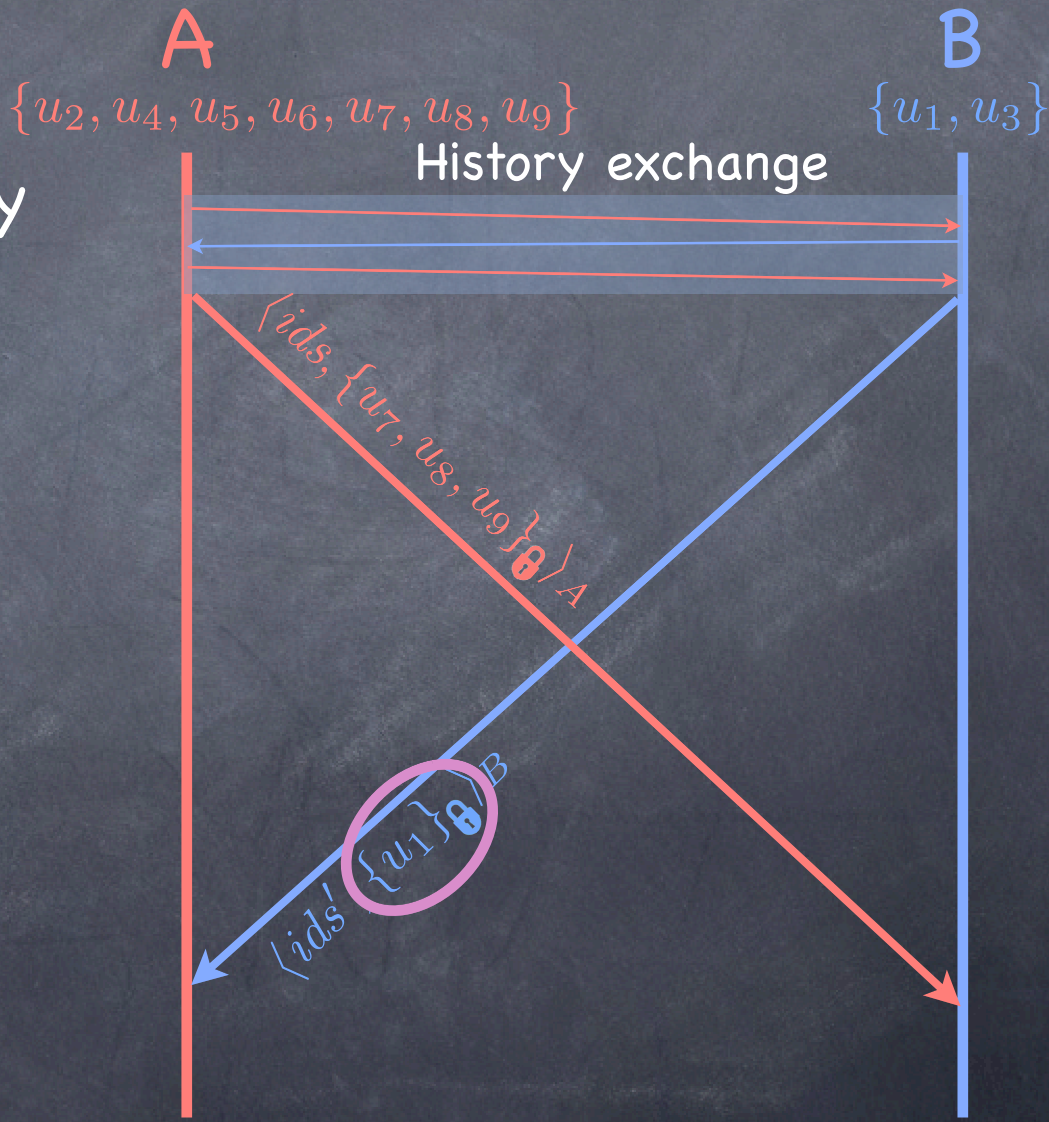


Optimistic Push



Optimistic Push

Q: How do we encourage a lagging client to send as many updates as possible?

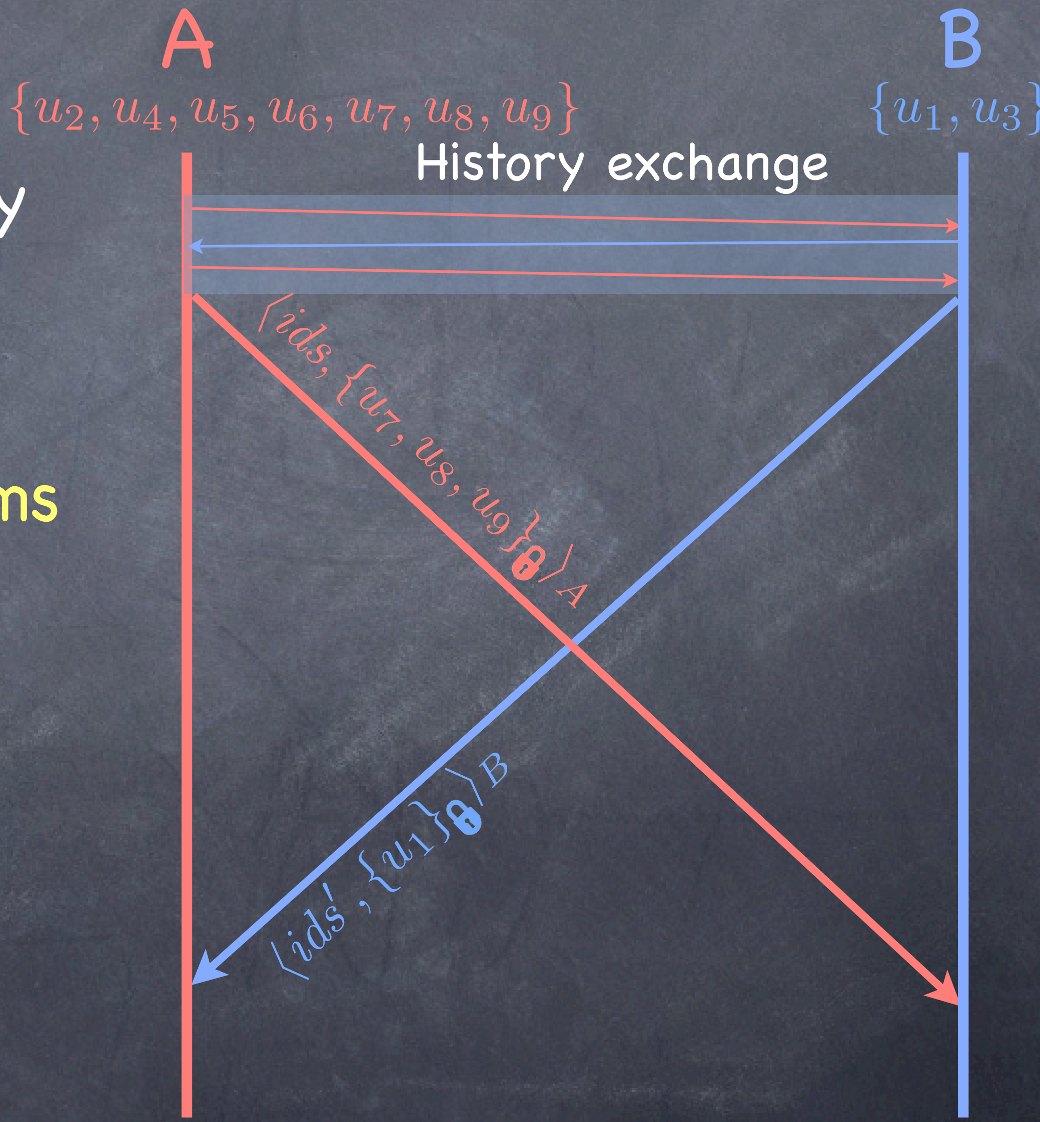


Optimistic Push

Q: How do we encourage a lagging client to send as many updates as possible?

A: Require both briefcases to have the same number of items

- If necessary, include junk

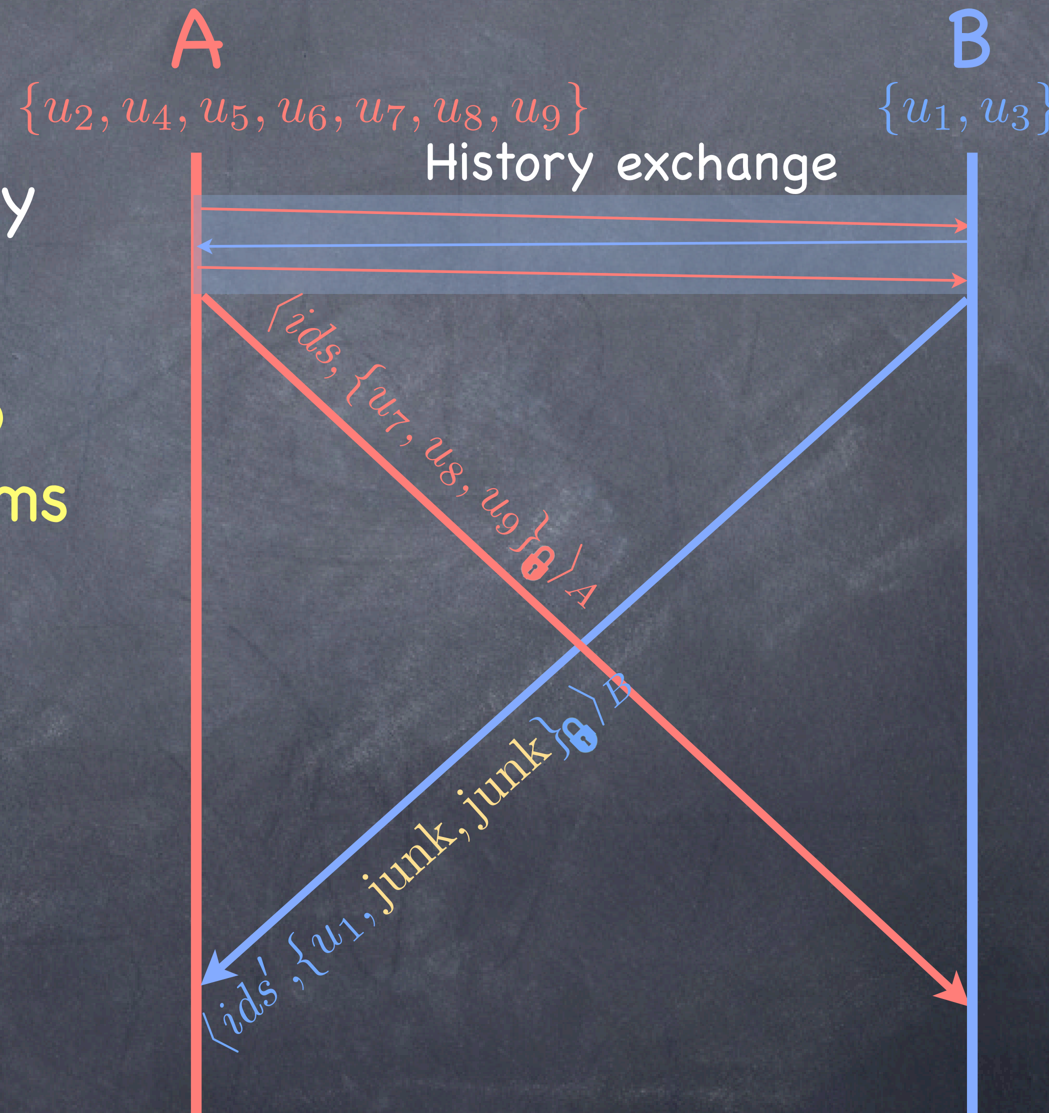


Optimistic Push

Q: How do we encourage a lagging client to send as many updates as possible?

A: Require both briefcases to have the same number of items

- If necessary, include junk

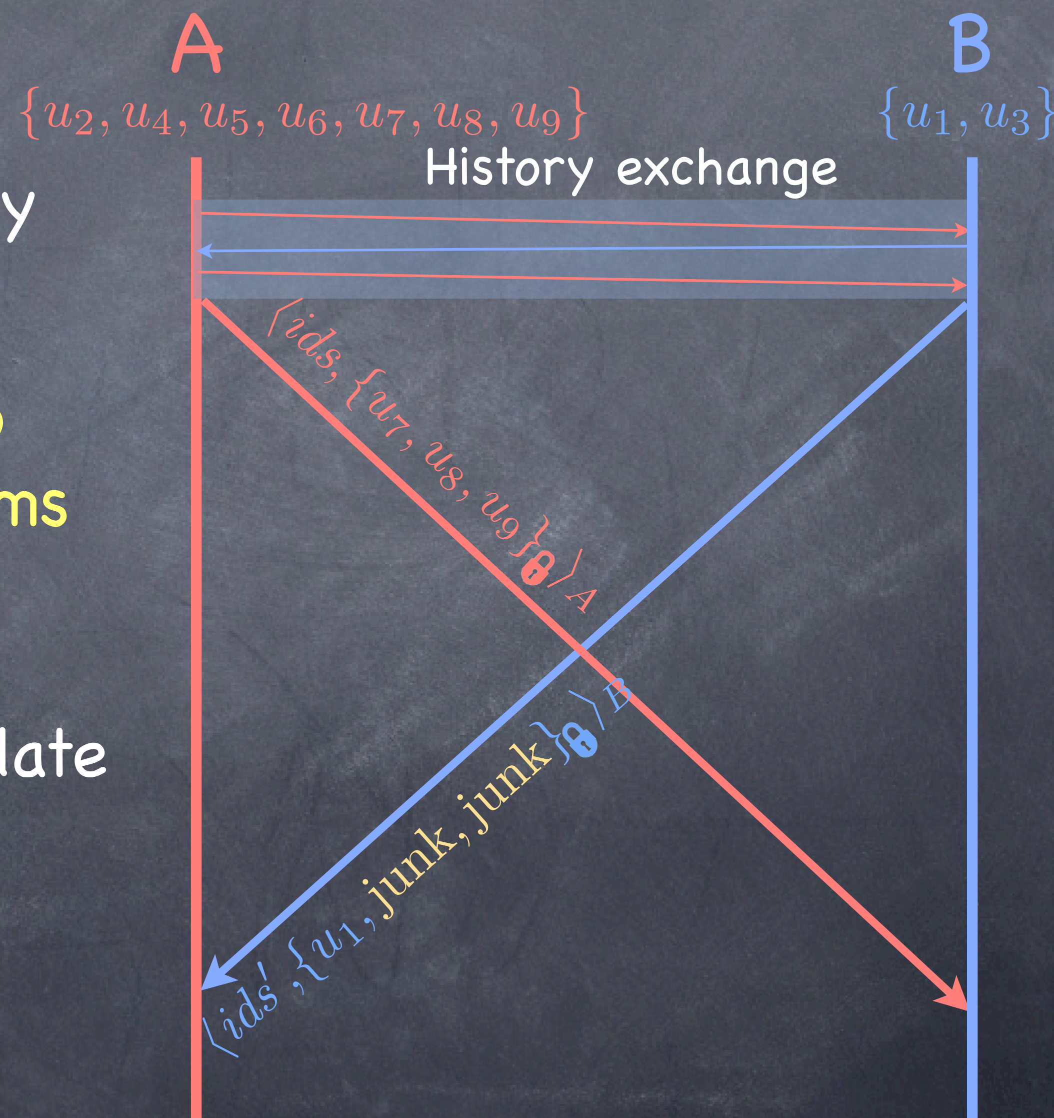


Optimistic Push

Q: How do we encourage a lagging client to send as many updates as possible?

A: Require both briefcases to have the same number of items

- If necessary, include junk
- Junk is larger than an update

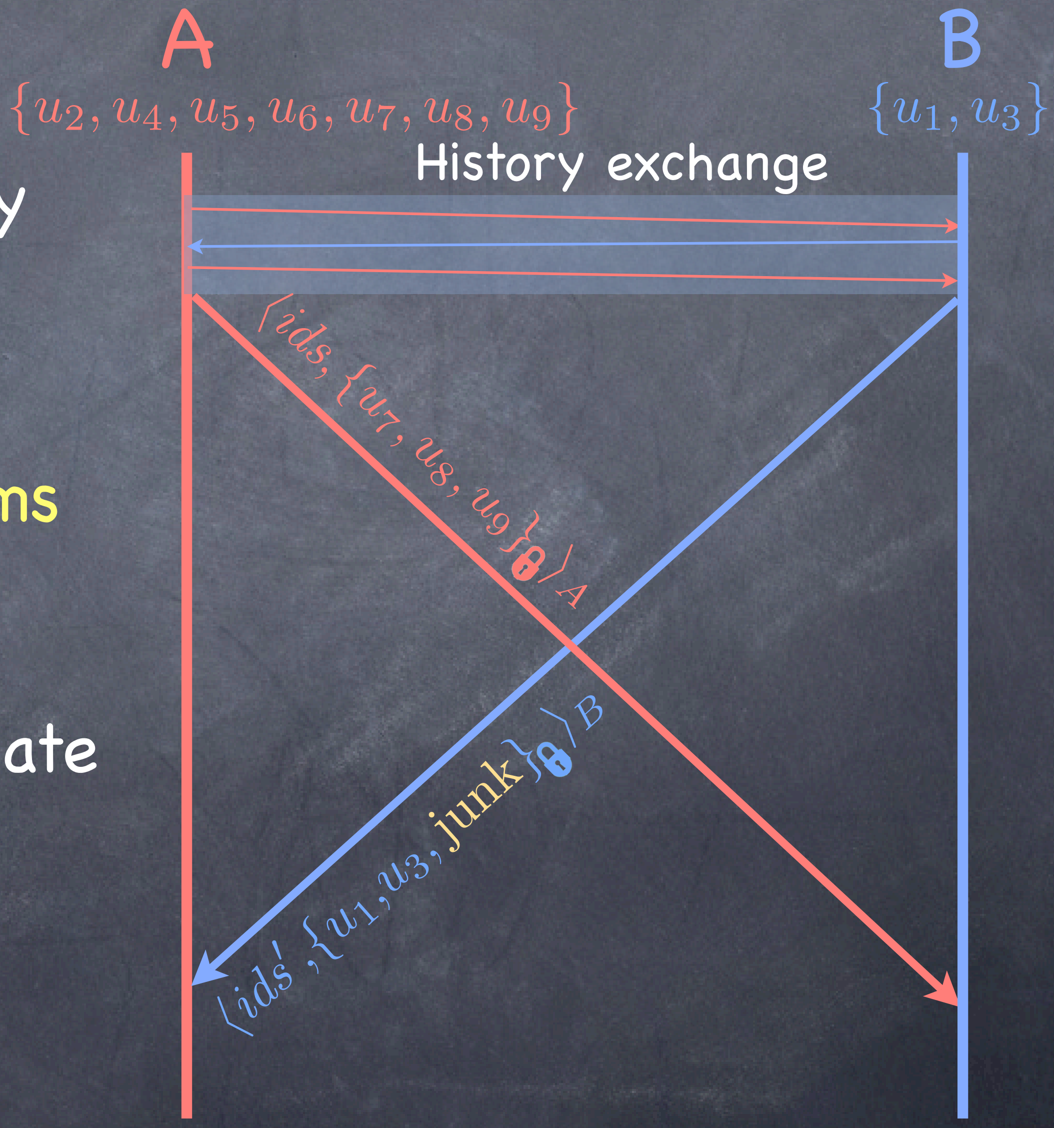


Optimistic Push

Q: How do we encourage a lagging client to send as many updates as possible?

A: Require both briefcases to have the same number of items

- If necessary, include junk
- Junk is larger than an update



BAR Gossip Recap

Balanced Exchange

In each round:

- Select partner
- Exchange histories
- Trade equal number of updates

Incentive compatible!

Optimistic Push

In each round:

- Select partner
- Exchange histories
- Trade possibly unequal numbers of updates

Explore strategy space experimentally

FlightPath Experiments

- Setup: 45 Emulab clients, each update multicast to random 3 clients
- Goal: evaluate Optimistic Push strategy space
  - Which strategies are attractive?
  - Which strategies are attractive with failures?

Alternate Strategies in Optimistic Push

	Responds with updates	Responds with junk	Doesn't respond
Initiates Pushes			
Does not initiate pushes			

Alternate Strategies in Optimistic Push

	Responds with updates	Responds with junk	Doesn't respond
Initiates Pushes	Follow Protocol		
Does not initiate pushes			

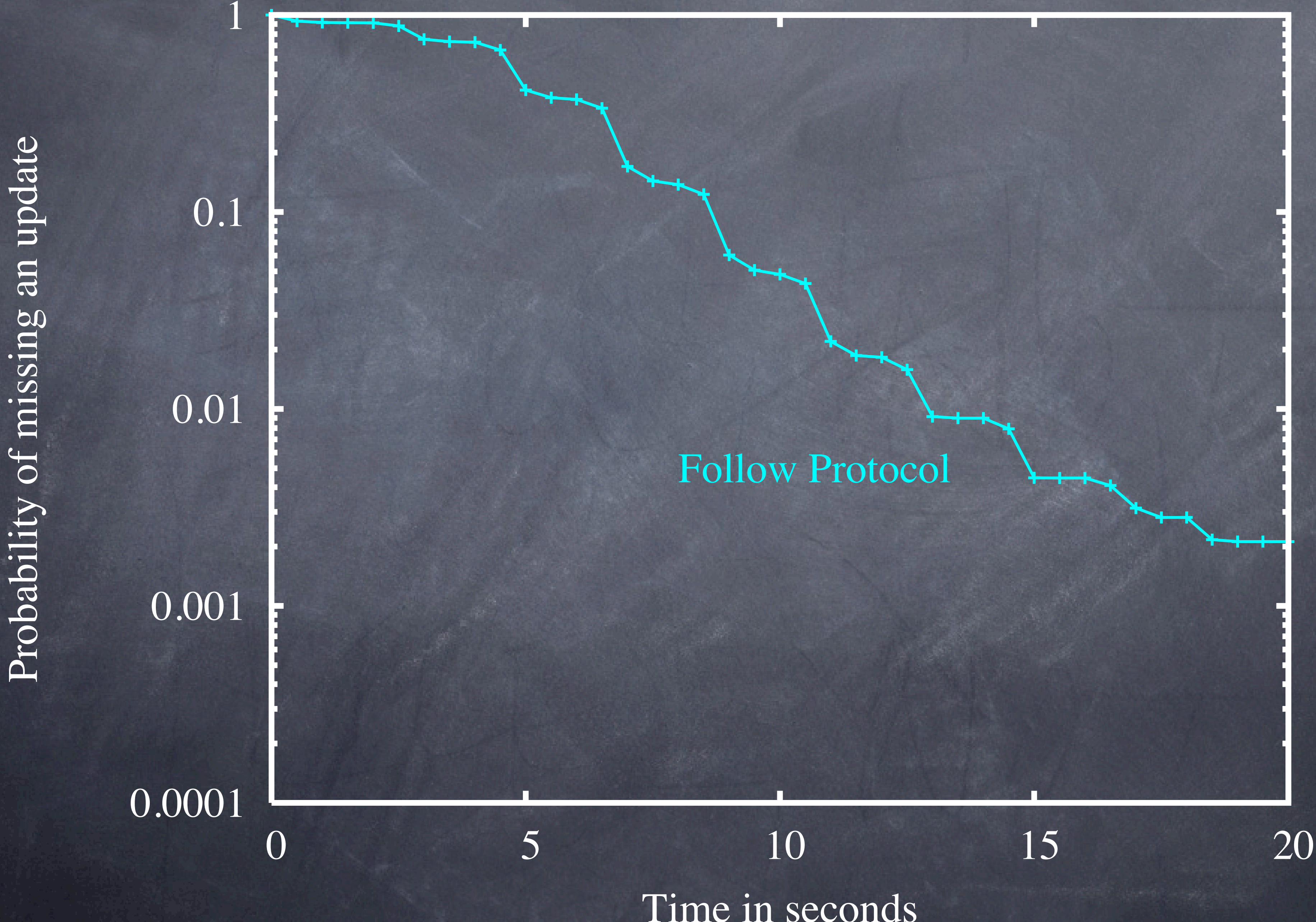
Alternate Strategies in Optimistic Push

	Responds with updates	Responds with junk	Doesn't respond
Initiates Pushes	Follow Protocol	Wasteful Strategy	
Does not initiate pushes			

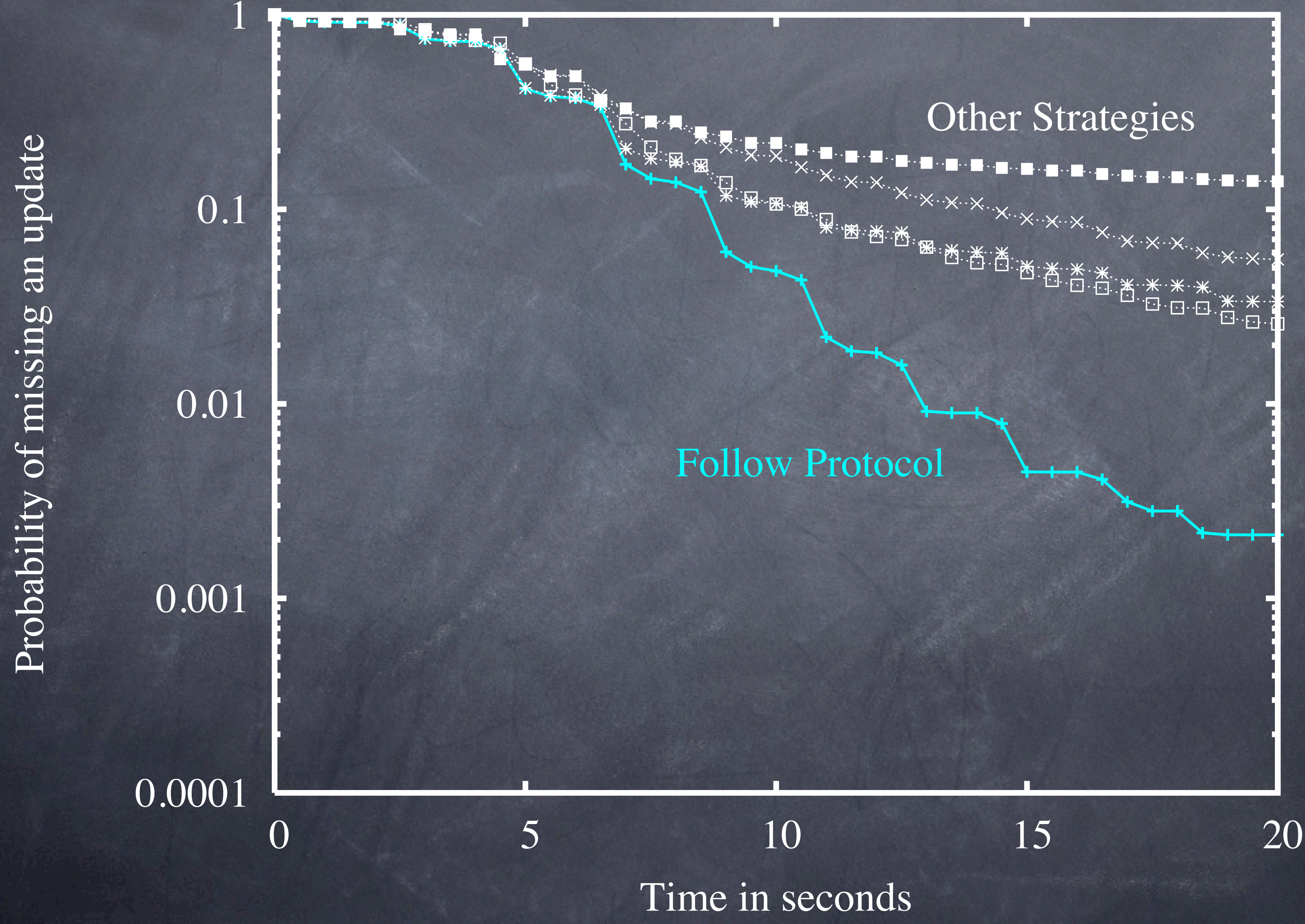
Alternate Strategies in Optimistic Push

	Responds with updates	Responds with junk	Doesn't respond
Initiates Pushes	Follow Protocol	Wasteful Strategy	
Does not initiate pushes	Other Strategies		

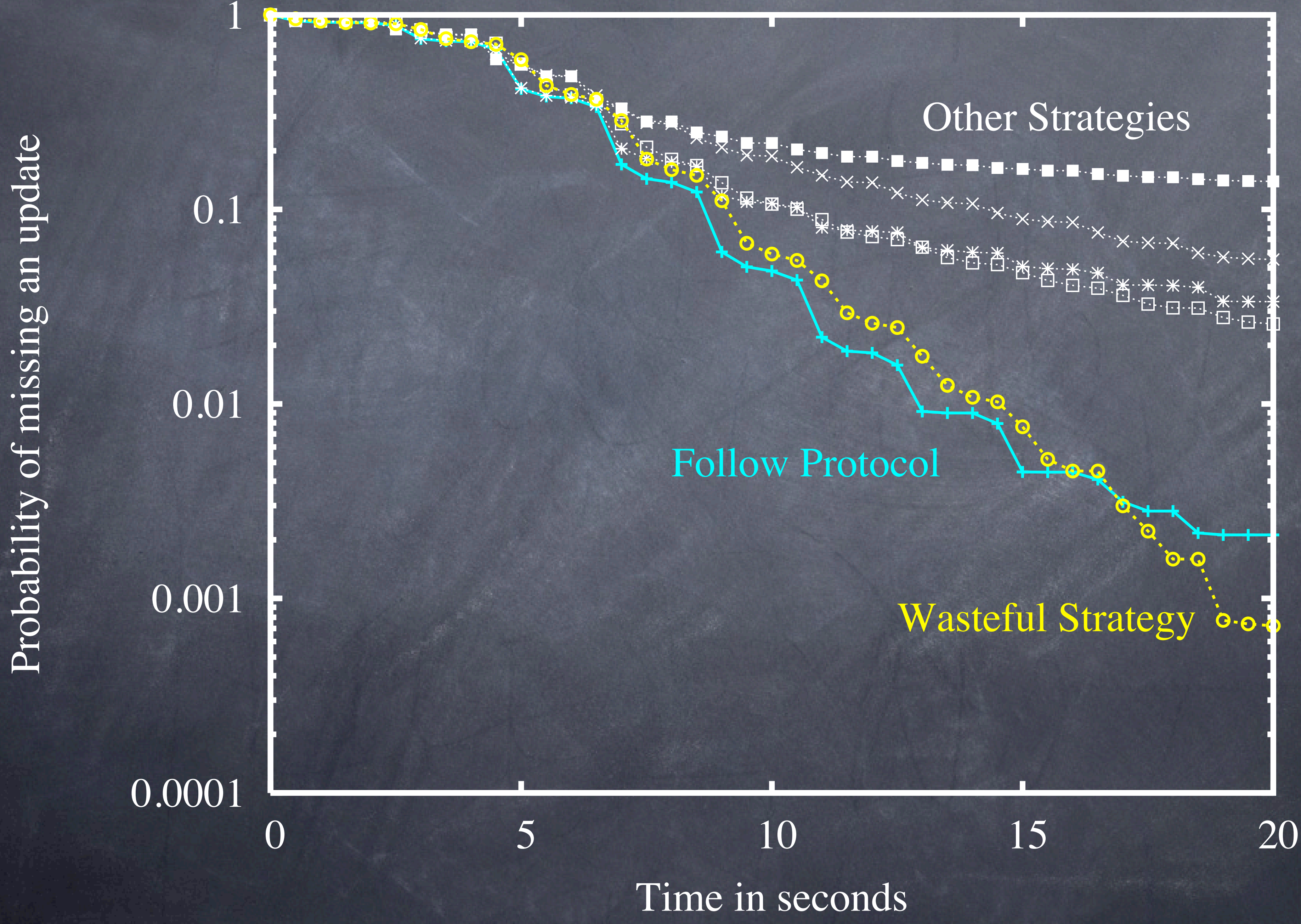
Convergence Graph



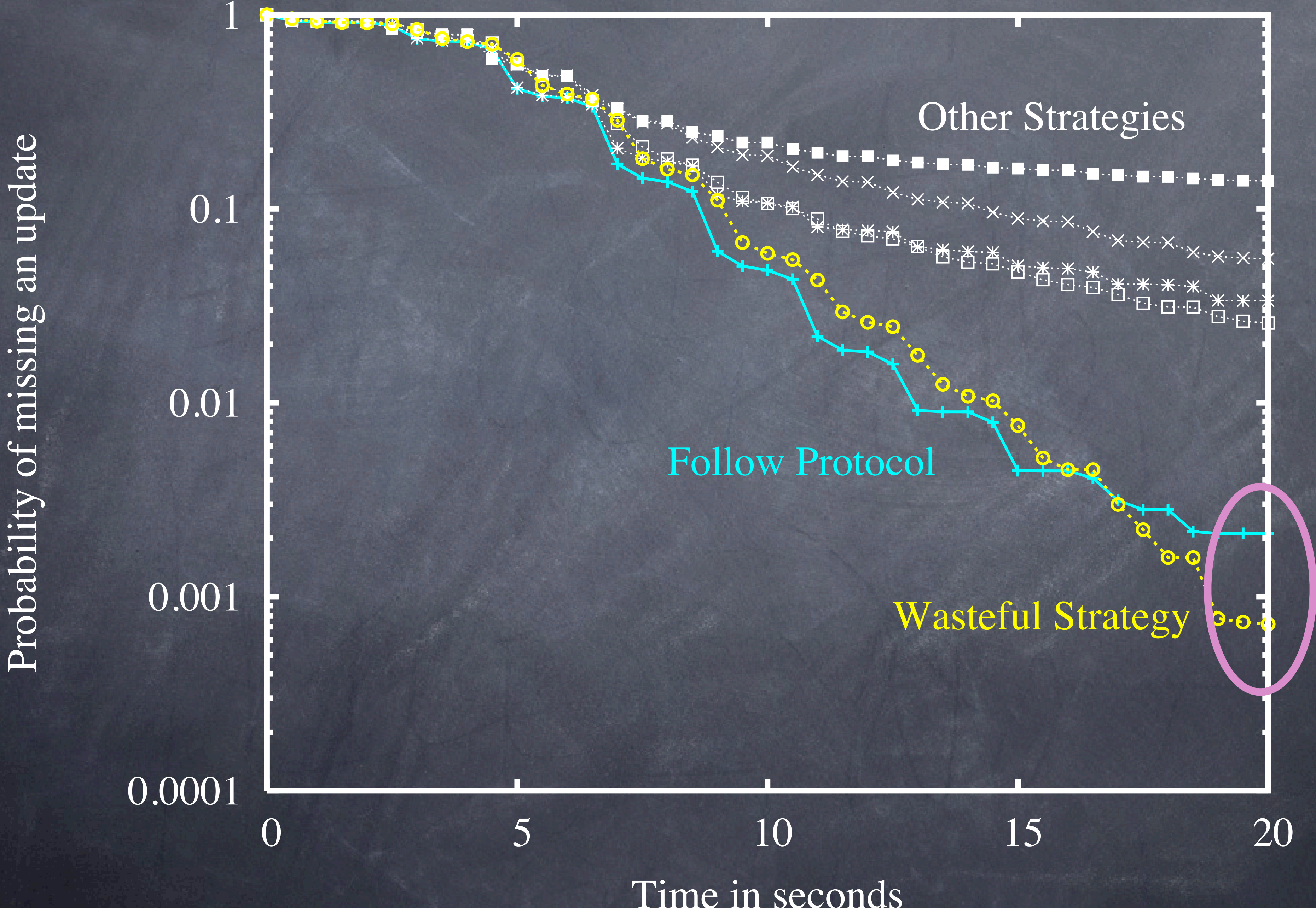
Convergence Graph



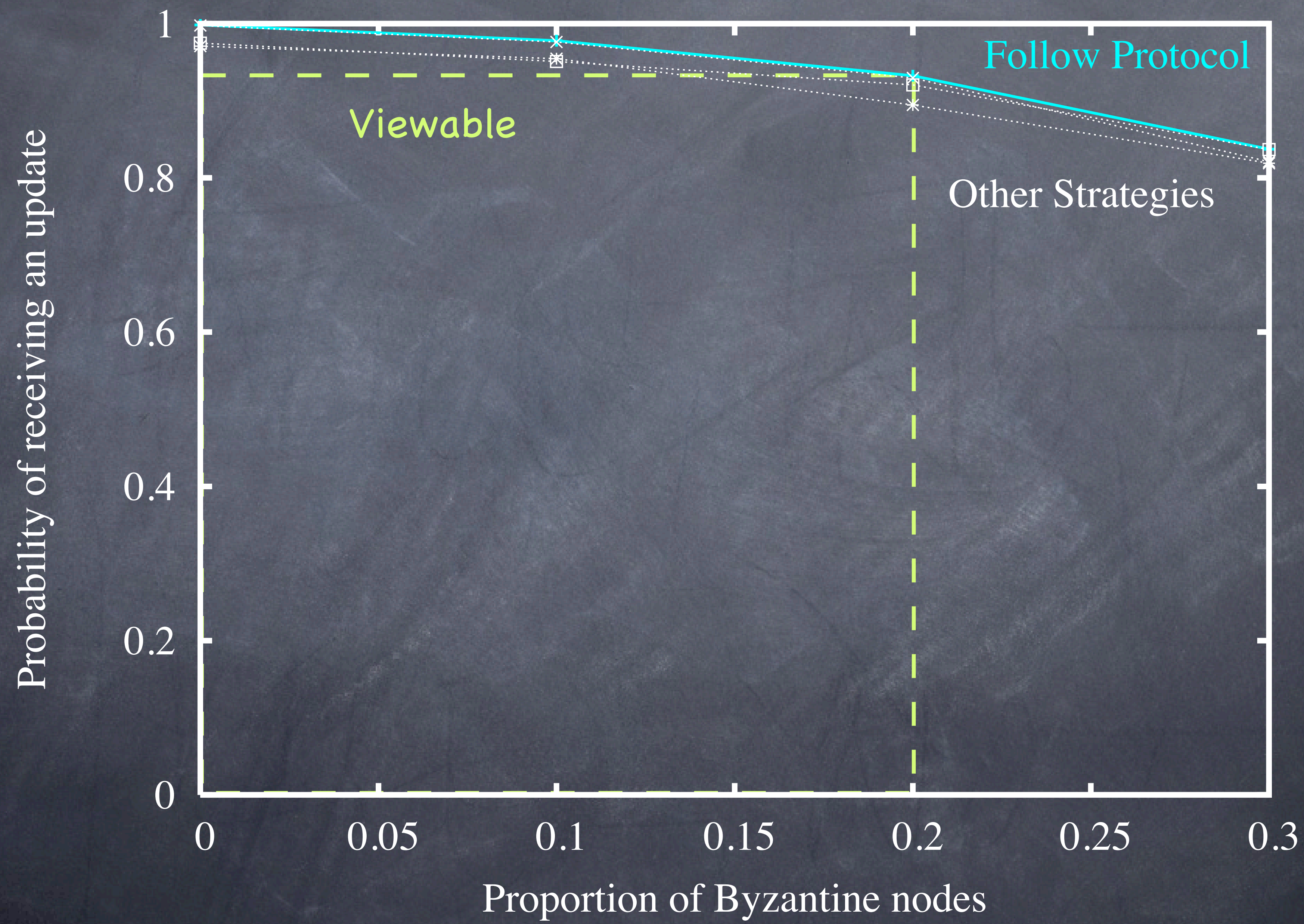
Convergence Graph



Convergence Graph



Reliability with Byzantine

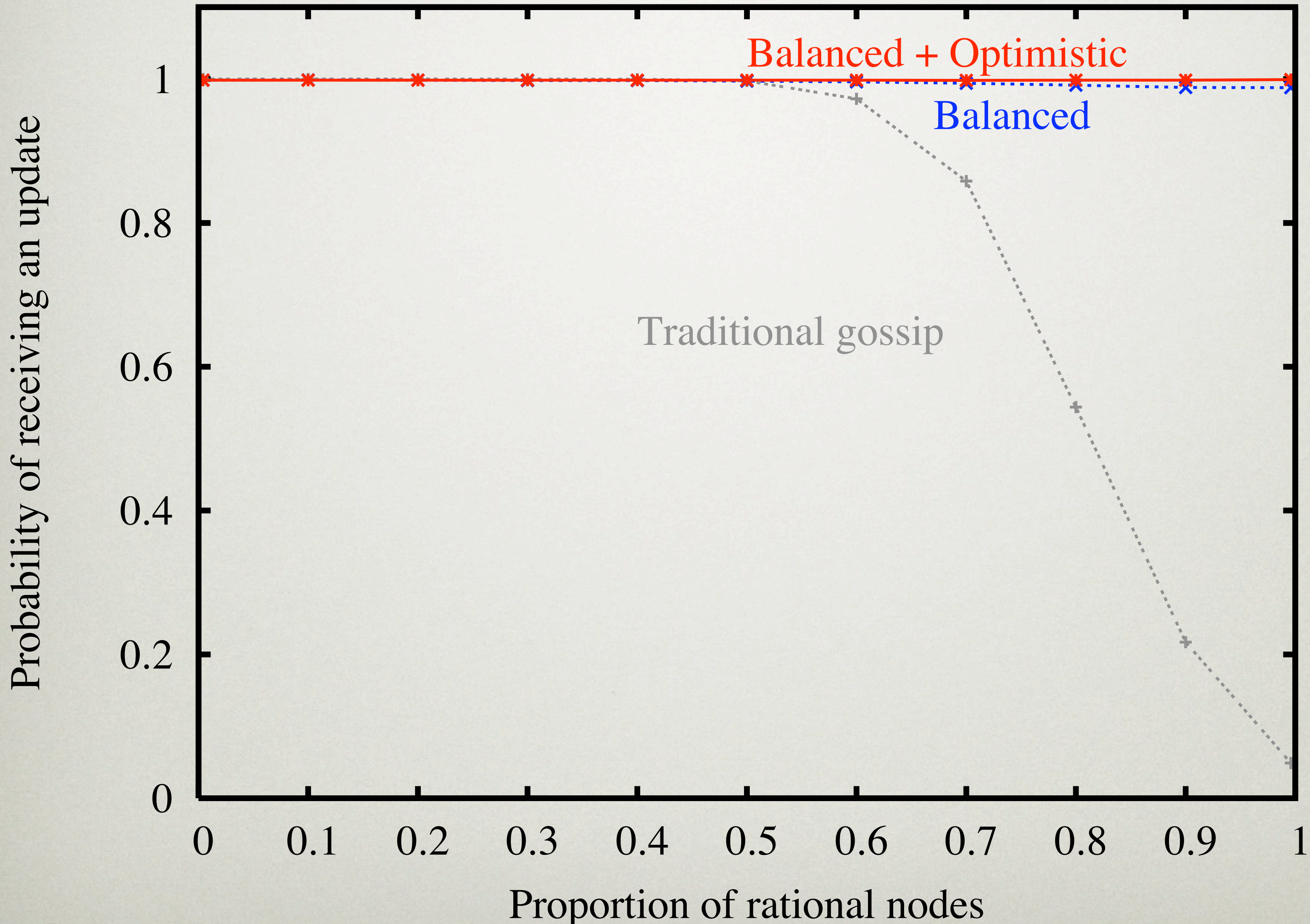


Conclusions

- BAR Gossip:
  - Balanced Exchange: provable, ~98%
  - Optimistic Push: ~99.9%
- Two key ideas:
  - Verifiable partner selection
  - Fair enough exchange
- Currently working on:
  - Dynamic membership
  - Partial membership
  - Network awareness

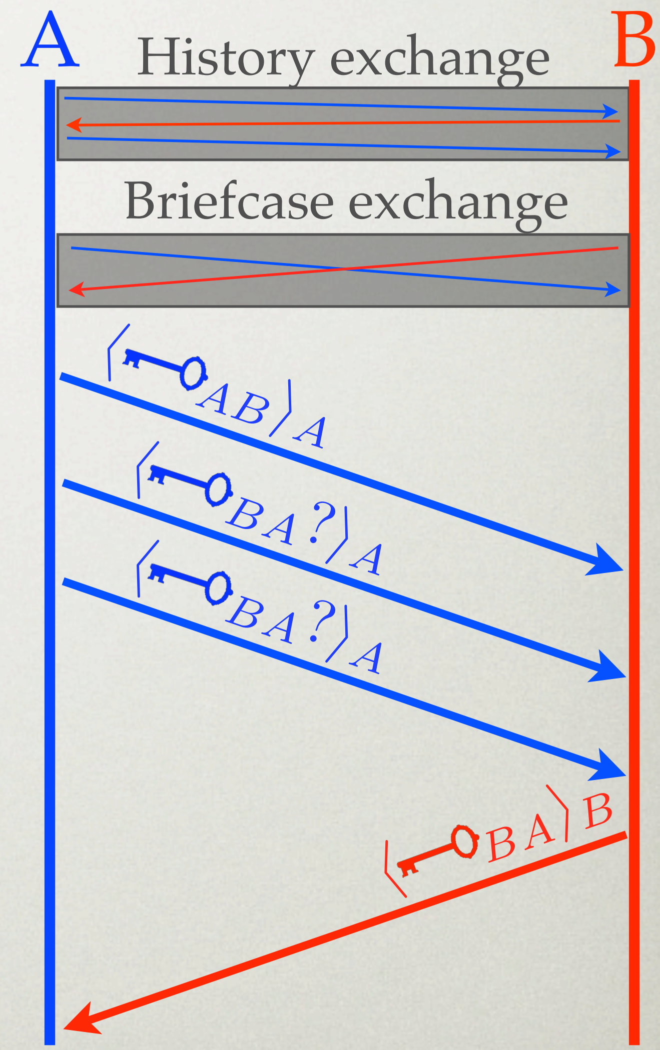
Backup Slides

Optimistic Push's Effect



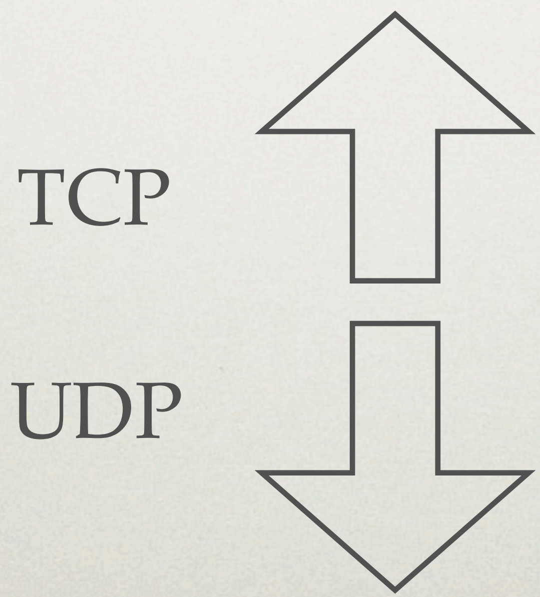
Why Resend Key Requests?

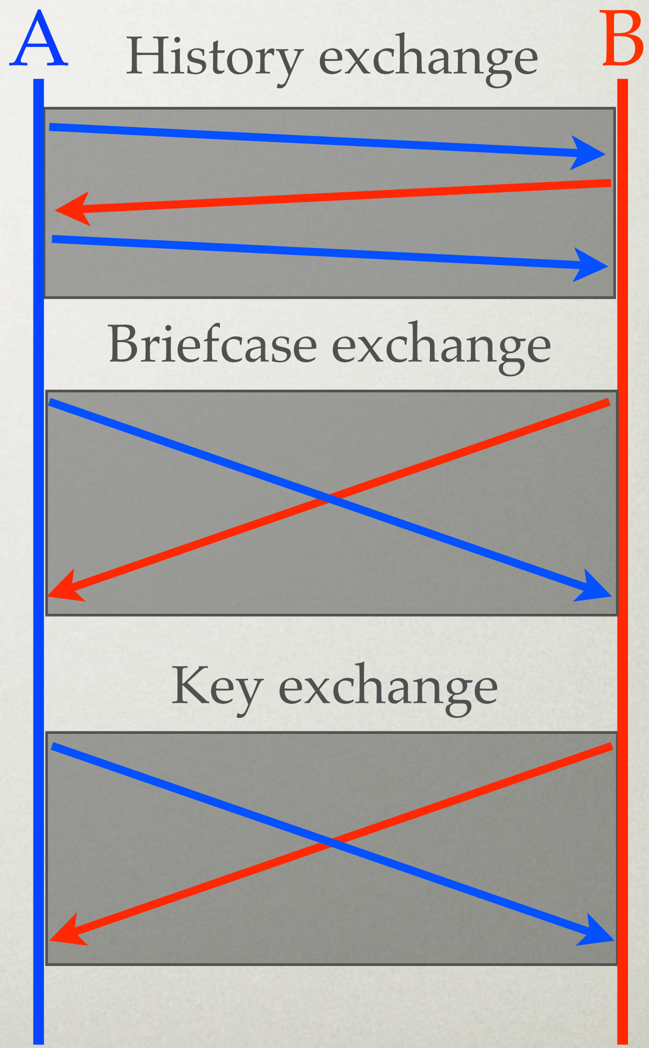
- Cost to A is small compared to big benefit of unlocking briefcase
- Cost to B is large compared to small benefit of not sending key



TCP and UDP

UDP necessary so that each peer *believes* its partner will send key requests





Why Reject?

- Peer terminates an exchange if that peer expects nothing useful from its partner
- Peer expects something useful only if it believes in fair enough exchange
- Fair enough exchange mechanism relies on mutual fear of eviction

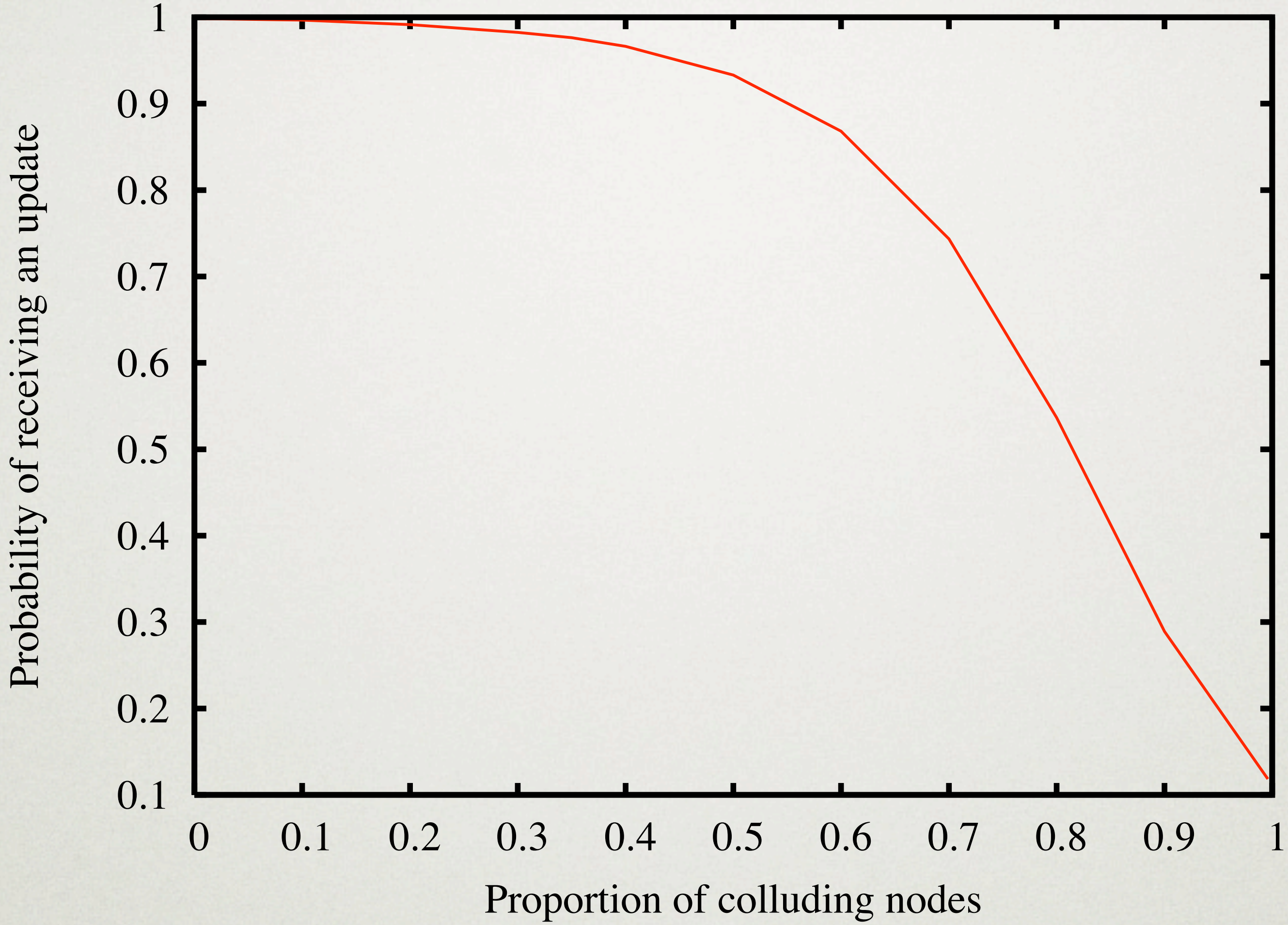
How Does Eviction Work?

- Broadcaster evicts clients by attaching eviction notices onto updates
- Broadcaster periodically asks clients to testify against their peers
- Clients testify because they expect nothing useful from future exchanges with those peers

End-to-End Metric

Strategy	Jitter	Std. Dev.
Follow Protocol	0.48%	1.16%
Wasteful Strategy	0.32%	0.78%
Initiate OP, Decline OP	11.59%	6.22%
Respond to OP with useful	18.10%	6.08%
Respond to OP with junk	14.76%	9.44%
Never run OP	47.94%	7.52%

Collusion



- Colluding nodes use unrealistic protocol
- BAR Gossip still robust for small colluding groups
- For large groups, colluding nodes may not trust each other

Denial-of-Service

DoS Resistant Unforgeable Multicast (DRUM)

- Resource bounding
- Random port hopping

Gal Badishi , Idit Keidar , Amir Sasson. Exposing and Eliminating Vulnerabilities to Denial of Service Attacks in Secure Gossip-Based Multicast, In Proceedings of DSN, 2004.