Group messaging in the XRD private communication system

David Lu
Mentor: Albert Kwon

Acknowledgements

Thank you to Albert Kwon for mentoring me

Thank you to Prof. Devadas for PRIMES-CS

Thank you to Dr. Gerovitch for the PRIMES program

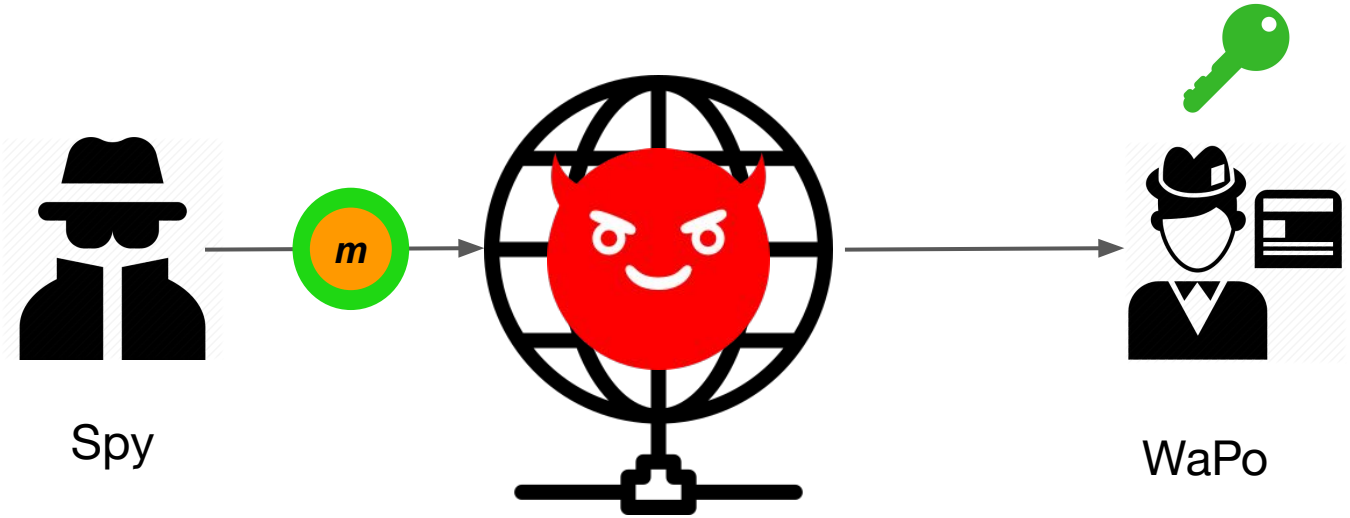
Thank you to my parents for their support

Motivation

Spy hides message *content* through encryption.

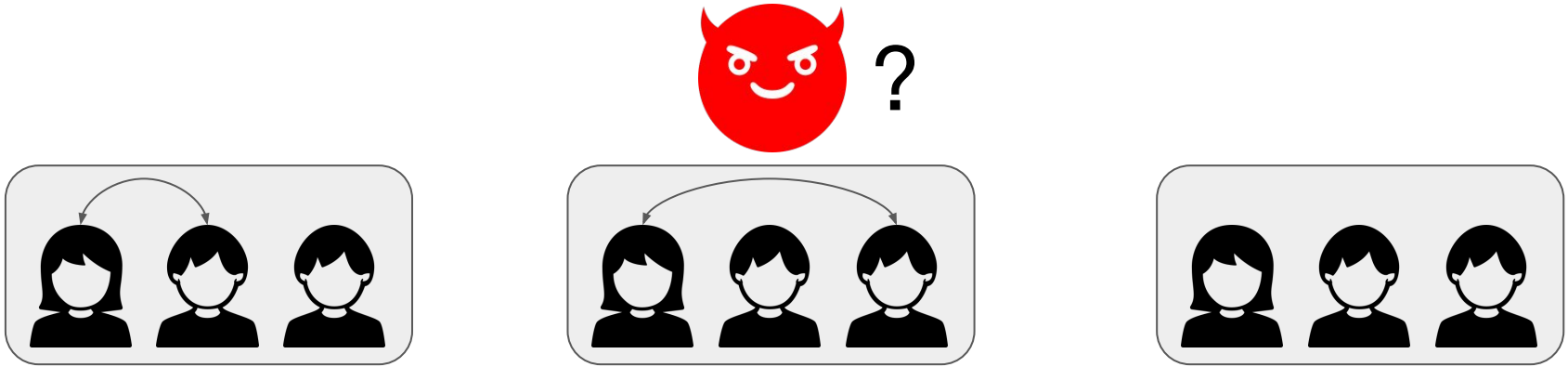
However, spy still leaks *metadata*:

- Identities
- Timing
- Size



Privacy guarantee

- Provide metadata private messaging against powerful adversaries

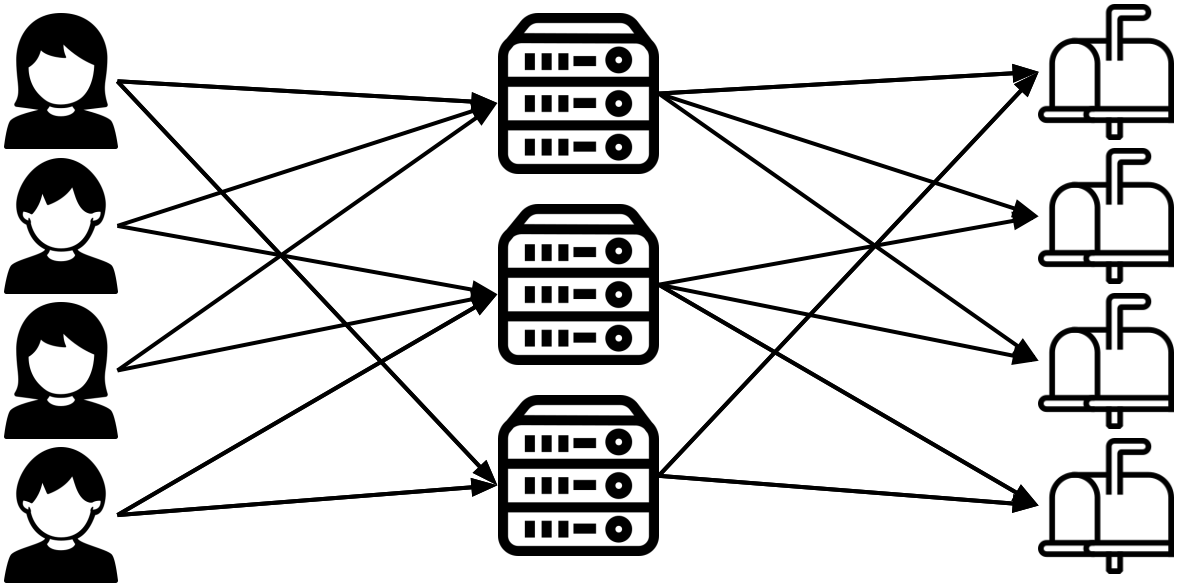


Deployment and threat model

- Global network adversary

XRD: basic design

$\ell = 2$

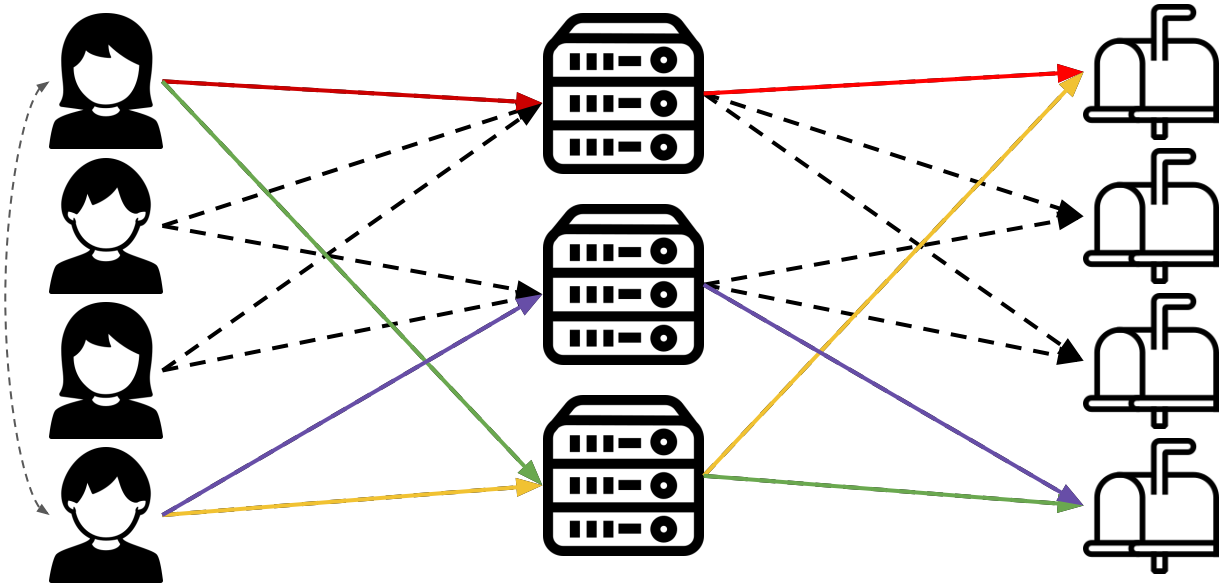


1. Send messages to ℓ servers

2. Forward messages to mailboxes

XRD: basic design

If 1 and 4 are talking to each other with $\ell = 2$

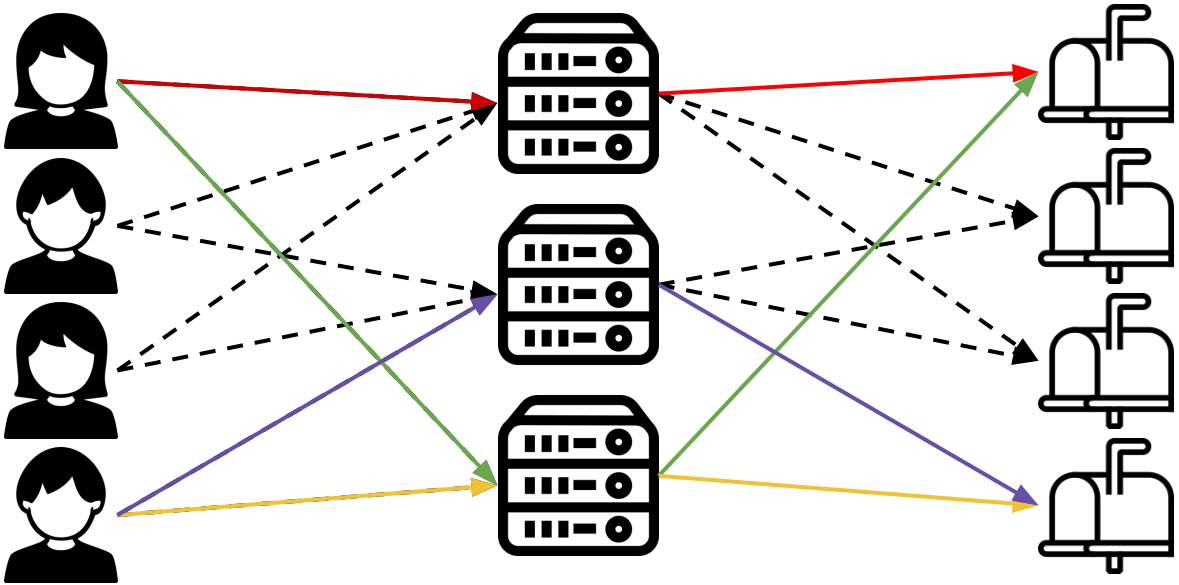


1. Send messages to ℓ servers

2. Forward messages to mailboxes

XRD: basic design

If 1 and 4 are *not* talking to each other with $\ell = 2$

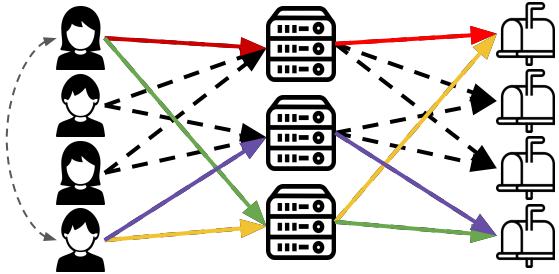


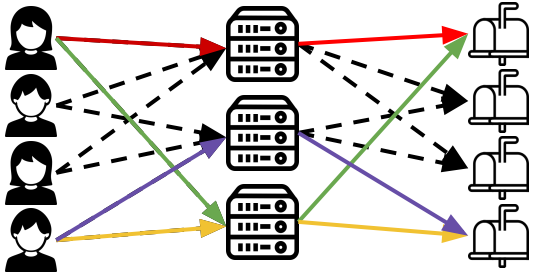
1. Send messages to ℓ servers

2. Forward messages to mailboxes

Security argument

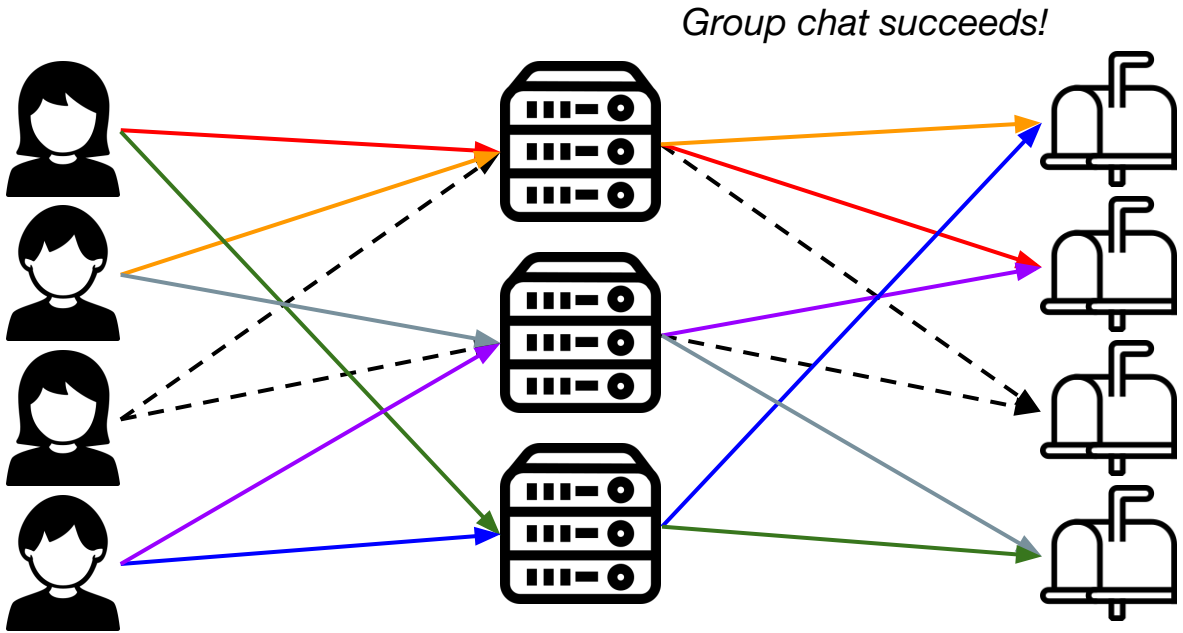
- Every mailbox gets exactly ℓ messages
  - Mailboxes are identical
- Every pair of users intersects
  - Hides which users are talking with each other
- The origin of the message is hidden by the server
  - Hides swap-or-not





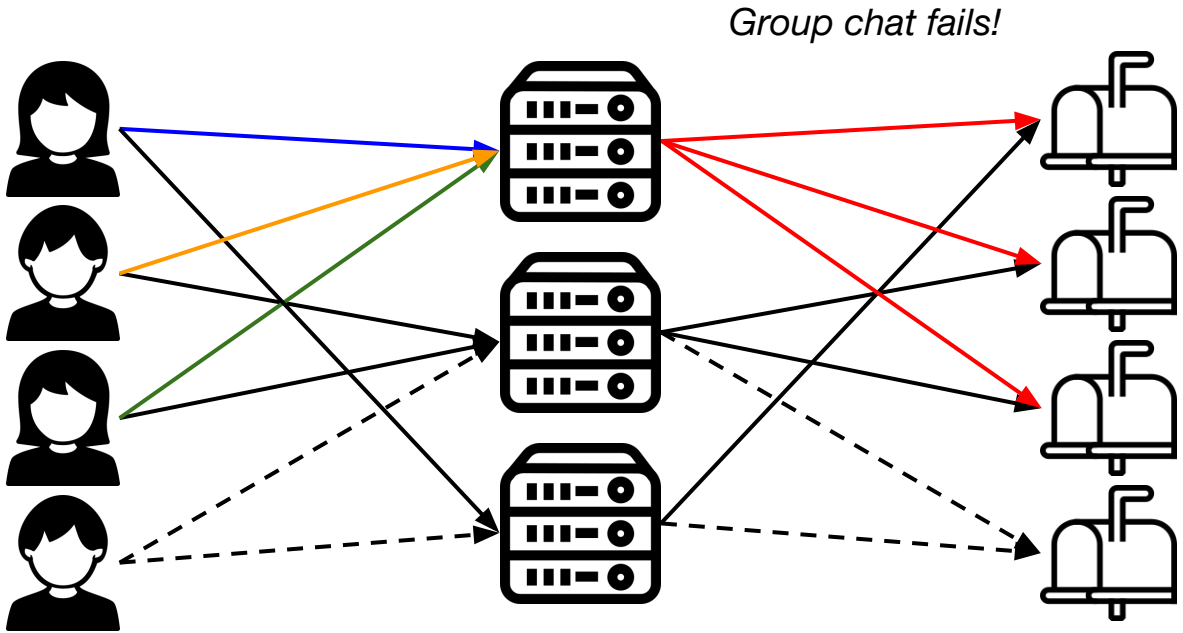
Group messaging

- Multiple 1-1 conversations



Group messaging

- Multiple 1-1 conversations



Goal

- Enable group conversations of any size
- Minimize the max number of messages that a server gets
  - This determines overall latency of XRD

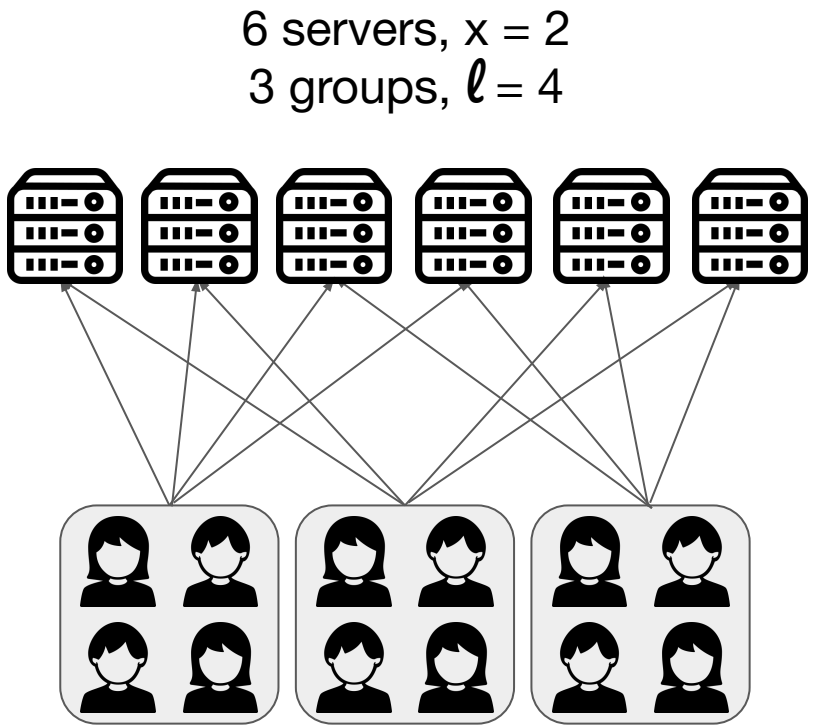
Approaches

- Randomized algorithm
- Parallel algorithm
- Linear programming
- Mixed integer programming
- Greedy algorithms (this talk)

Greedy algorithm #1

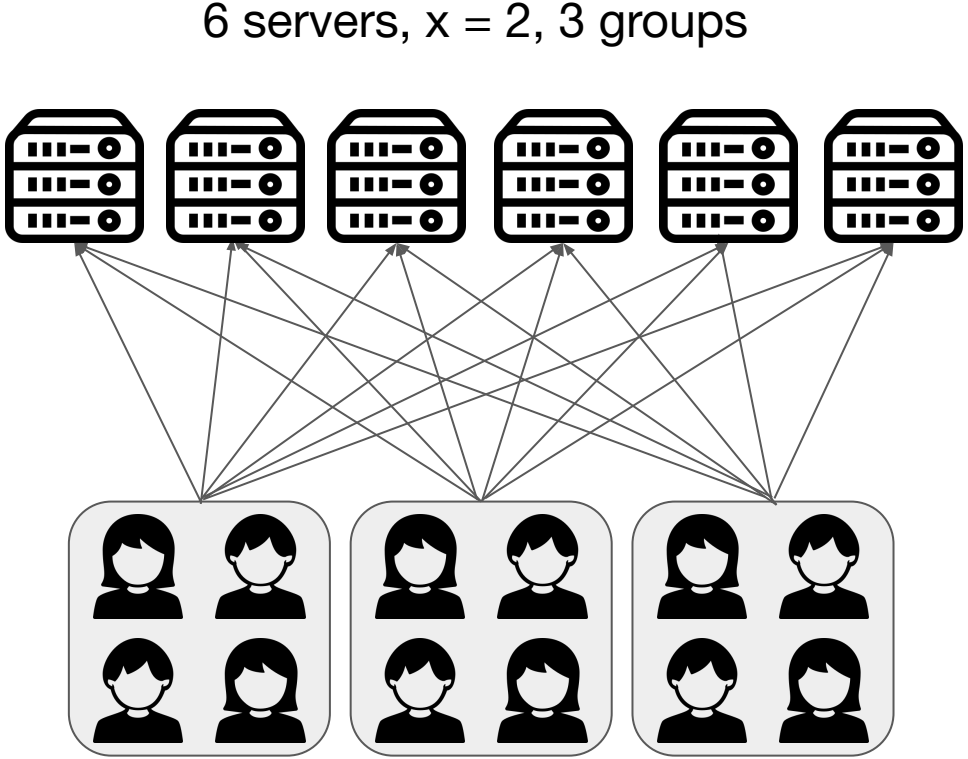
- **Goal:** Every pair of users intersects at least x times

- Divide into (ℓ/x+1) groups
- $\ell = \sqrt{(2nx + x^2/4)} - x/2$ minimizes load per server (for n servers)



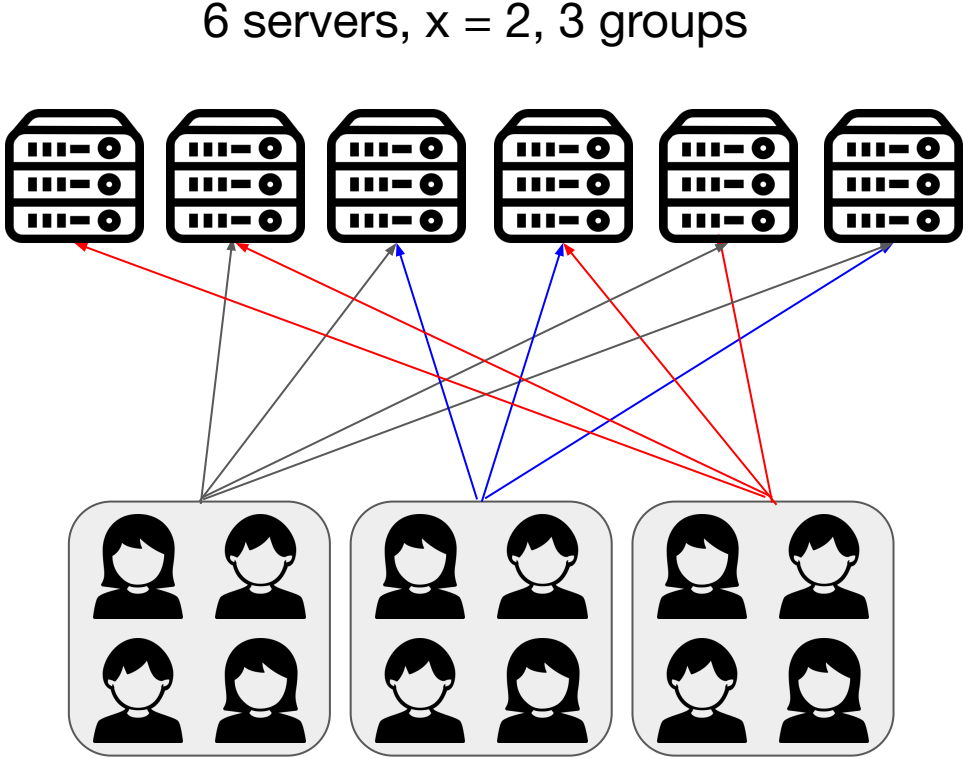
Greedy algorithm #2

- Start with every user sending message to every server
- Remove messages that least impact intersections between users

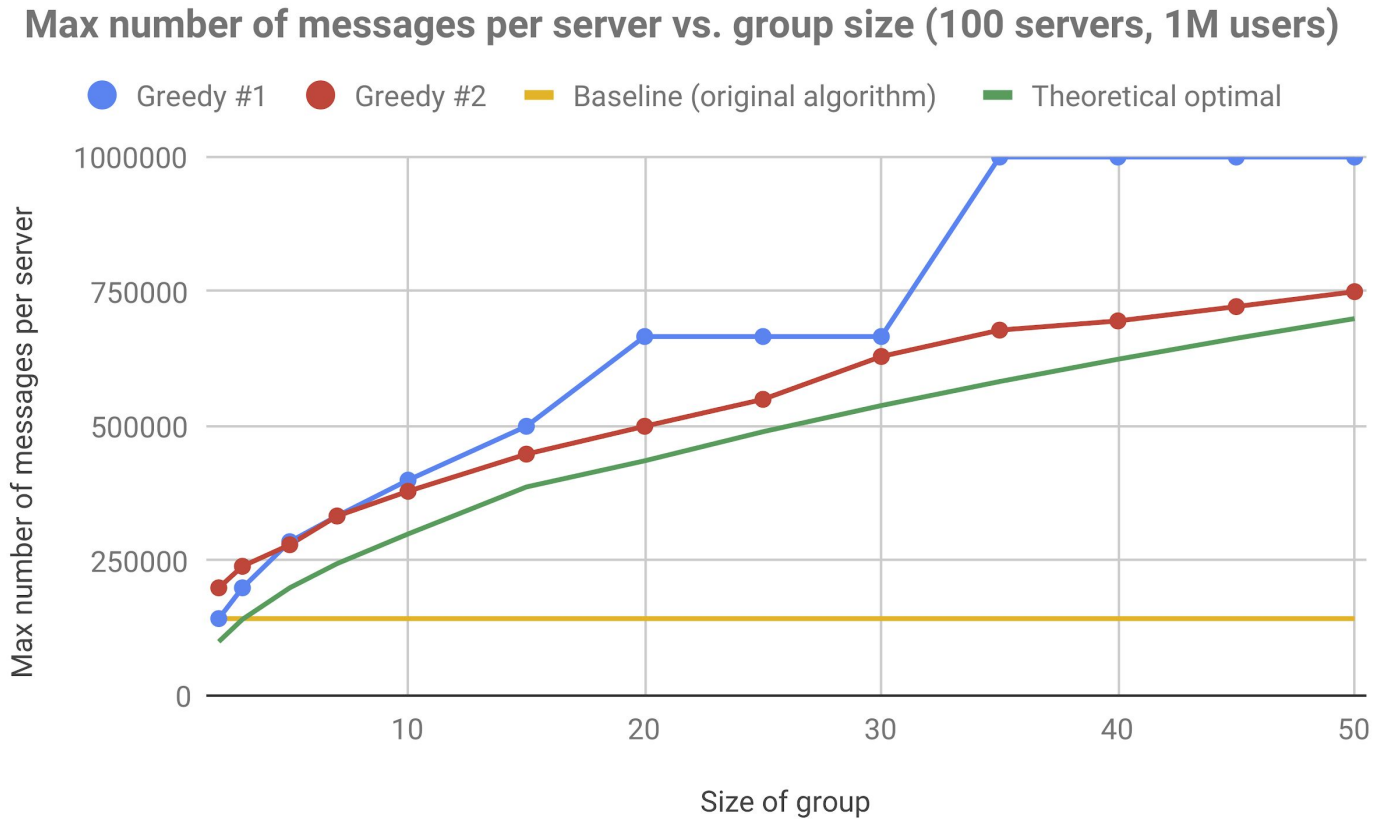


Greedy algorithm #2

- Start with every user sending message to every server
- Remove messages that least impact intersections between users



Results

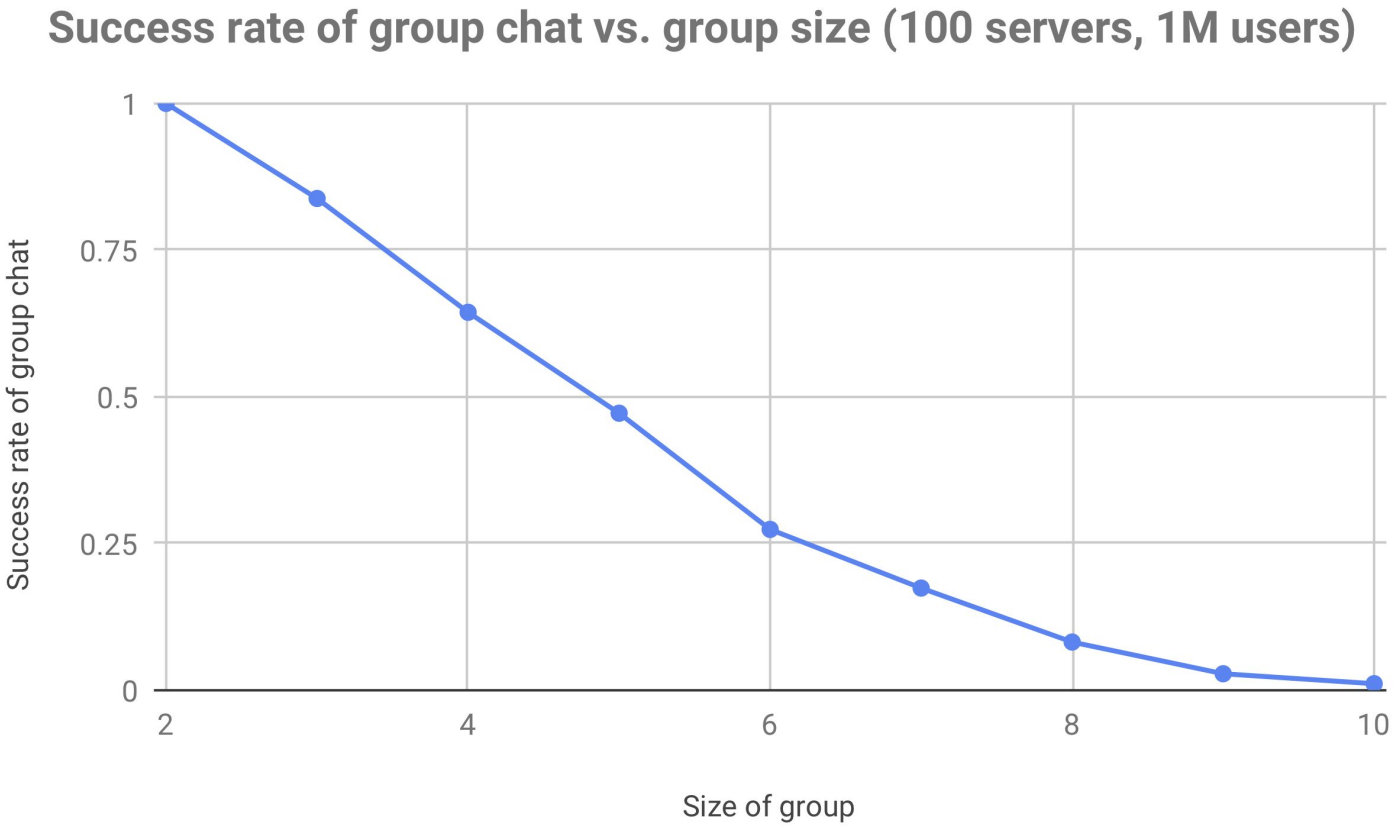

Max number of messages per server vs. group size (100 servers, 1M users)
Greedy #1
Greedy #2
Baseline (original algorithm)
Theoretical optimal
1000000
750000
500000
250000
0
Max number of messages per server
10
20
30
40
50
Size of group

Conclusion

- Two greedy algorithms to enable group messaging in XRD
- For 100 servers and 1M users:
  - Groups of up to 5 users with 96% latency increase
  - Groups of up to 20 users with 250% latency increase

Backup

Success rate of current algorithm



Scalability properties of original system

For m users and n chains,

- We can make sure all users intersect with $\ell = \sqrt{(2n)}$
- Each chain handles $m^*\ell/n = (\sqrt{2})^*m/(\sqrt{n})$ messages
  - If you increase n, the load per chain goes down (scalable)