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

1. Send messages to $l$ servers
2. Forward messages to mailboxes

$l = 2$
XRD: basic design

1. Send messages to $l$ servers
2. Forward messages to mailboxes

If 1 and 4 are talking to each other with $l = 2$
XRD: basic design

1. Send messages to $l$ servers
2. Forward messages to mailboxes

If 1 and 4 are *not* talking to each other with $l = 2$
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 chat succeeds!*
Group messaging

- Multiple 1-1 conversations

Group chat fails!
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 \((\ell/x+1)\) groups
- \(\ell = \sqrt{2nx + x^2/4} - x/2\)
- minimizes load per server (for n servers)

6 servers, \(x = 2\)
3 groups, \(\ell = 4\)
Greedy algorithm #2

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

6 servers, $x = 2, 3$ groups
Greedy algorithm #2

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

6 servers, $x = 2, 3$ groups
Results

Max number of messages per server vs. group size (100 servers, 1M users)

- Greedy #1
- Greedy #2
- Baseline (original algorithm)
- Theoretical optimal

![Graph showing the max number of messages per server vs. group size with different algorithms and theoretical optimal.](image-url)
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

Success rate of group chat vs. group size (100 servers, 1M users)
Scalability properties of original system

For \( m \) users and \( n \) chains,

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