SecretRoom:
An Anonymous Chat Client

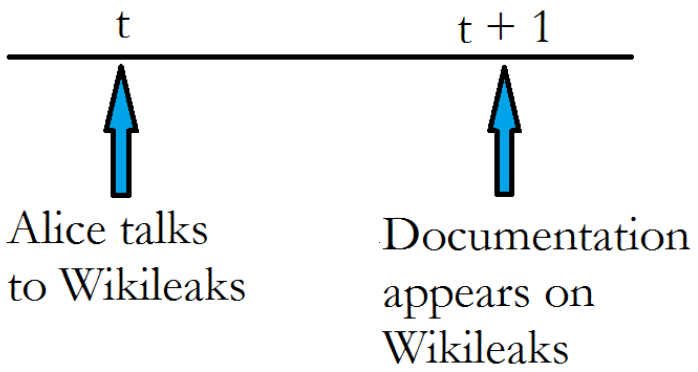
By: Cristian Gutu and Fengyao Ding
Computer Science
Mentor: Albert Kwon

Motivation

- Communicate anonymously
  - Wikileaks
  - Edward Snowden
- Encryption is easy
- Metadata is hard to hide

Hiding Metadata

- Hide ***who*** you talk to
  - Talking to Wikileaks may be enough to prosecute you
- Hide ***when*** you talk



Related Work: Tor

- Hides who is being talked to
- Does not hide when one talks
  - Especially with fewer people
- Anonymity is not guaranteed in presence of strong adversaries like ISPs and government.

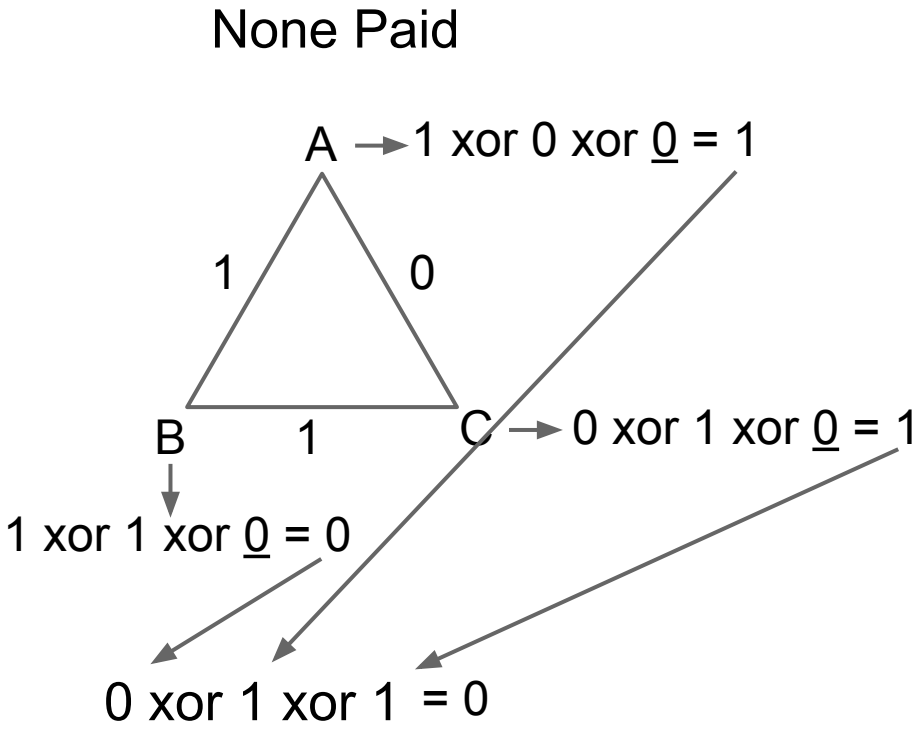
Dining Cryptographer Networks (DC-Nets)

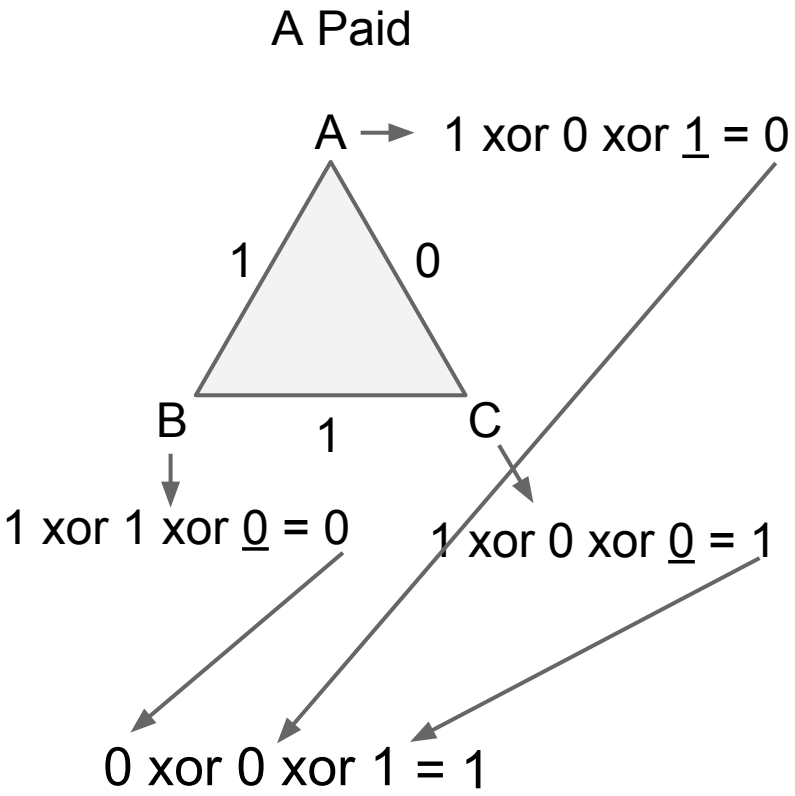
DC-Nets

- A truly anonymous communication protocol.
  - Hides who talked **and** when they talked.
- Based on boolean-XOR computations

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	0

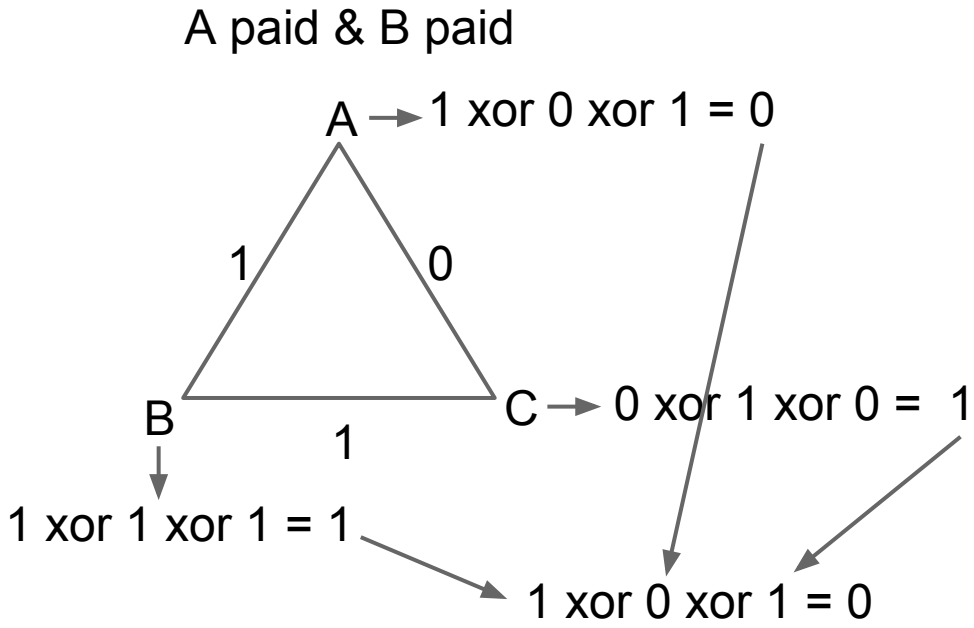
DC-Nets


None Paid
A
1 xor 0 xor 0 = 1
1
0
B
1
C
0 xor 1 xor 0 = 1
1 xor 1 xor 0 = 0
0 xor 1 xor 1 = 0


A Paid
A
1 xor 0 xor 1 = 0
1
0
B
1
C
1 xor 1 xor 0 = 0
1 xor 0 xor 0 = 1
0 xor 0 xor 1 = 1

DC-Nets: Problems

- $O(n^2)$ communication
- Collision when multiple people talk
  - Need a “schedule”

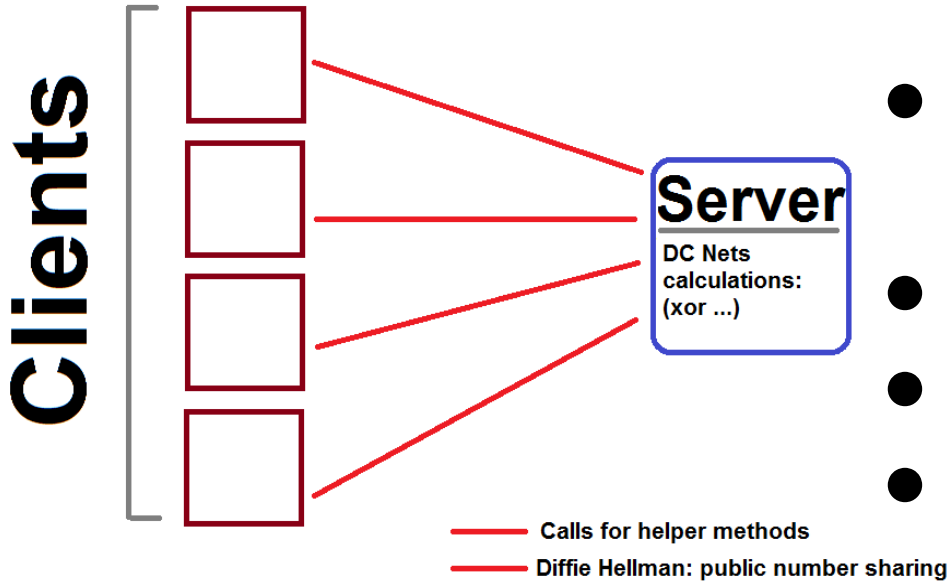


SecretRoom Design

SecretRoom Design

- Dynamic desktop application that relies on DC-Nets for a truly anonymous workflow.
- All client actions are obfuscated.
  - Network activity is strictly indistinguishable.

SecretRoom Model



- client-client → client-server
- server can be malicious
- k malicious clients
- n-k honest clients

Chat Room Functionality

- Chat room is pre-established per topic of interest
- Minimum of 3 clients per chat room
  - DC-Nets protocol requires at least 3 clients for anonymity

SecretRoom Protocol

- Build pairwise secrets
- Build chat schedule
- Communicate anonymously via DC-Nets

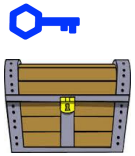
Building Pairwise Secrets

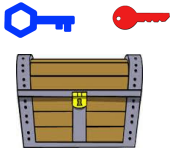
- Diffie Hellman Protocol
  - Allows 2 or more people to arrive at a common secret number, without revealing the secret

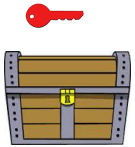
Diffie-Hellman

- Alice and Bob want to share a secret

Alice







Bob

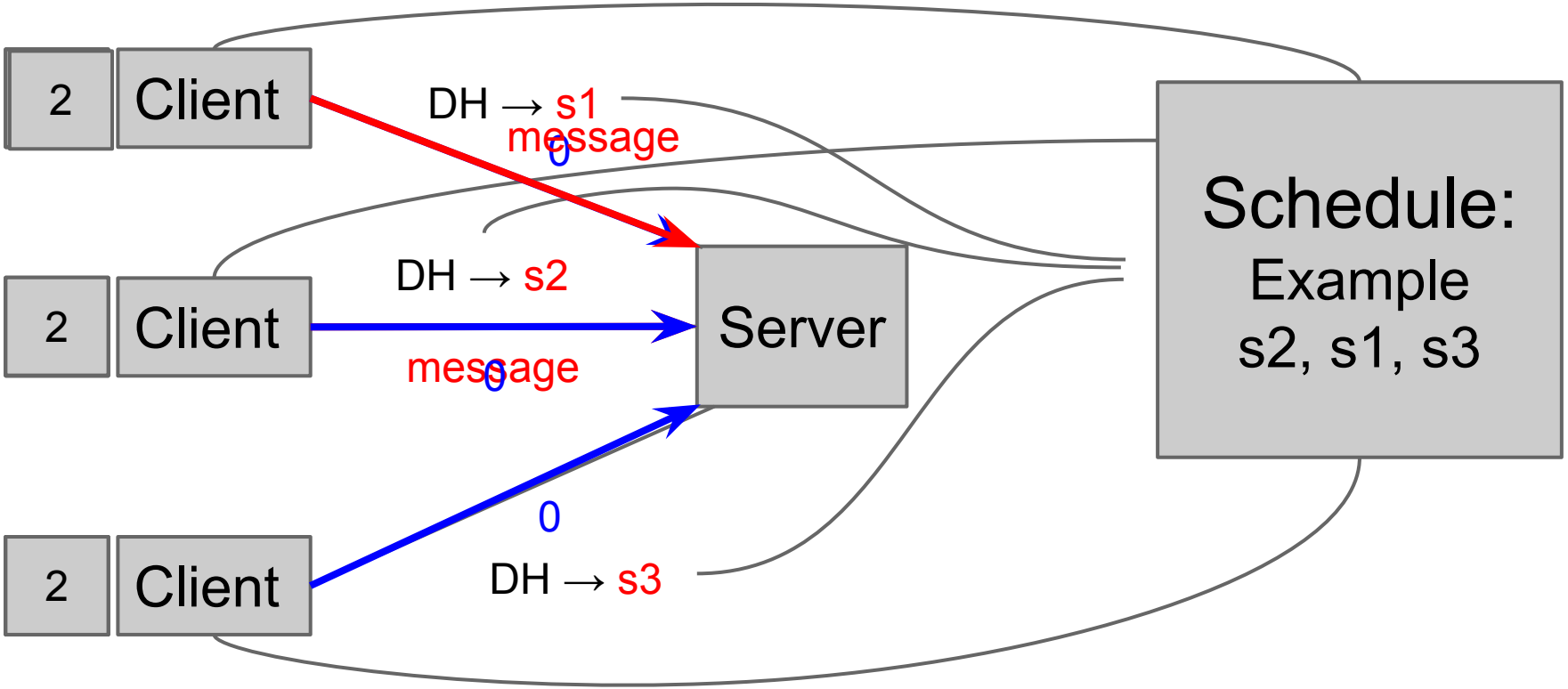
Diffie Hellman

- Given *p*, *g*, and a secret number *a*: compute g^a mod p
  - If g^a mod p = z and we know z, g, p *a* is hard to find (especially with large numbers)

Improving Secret Sharing

- Using naive DH exchange, a client need to talk to all other clients
  - $O(n^2)$ communication for n clients
- Upload/download public values to/from the server
  - Only need to communicate with the server
  - O(n) communication
  - Server does not learn secret values

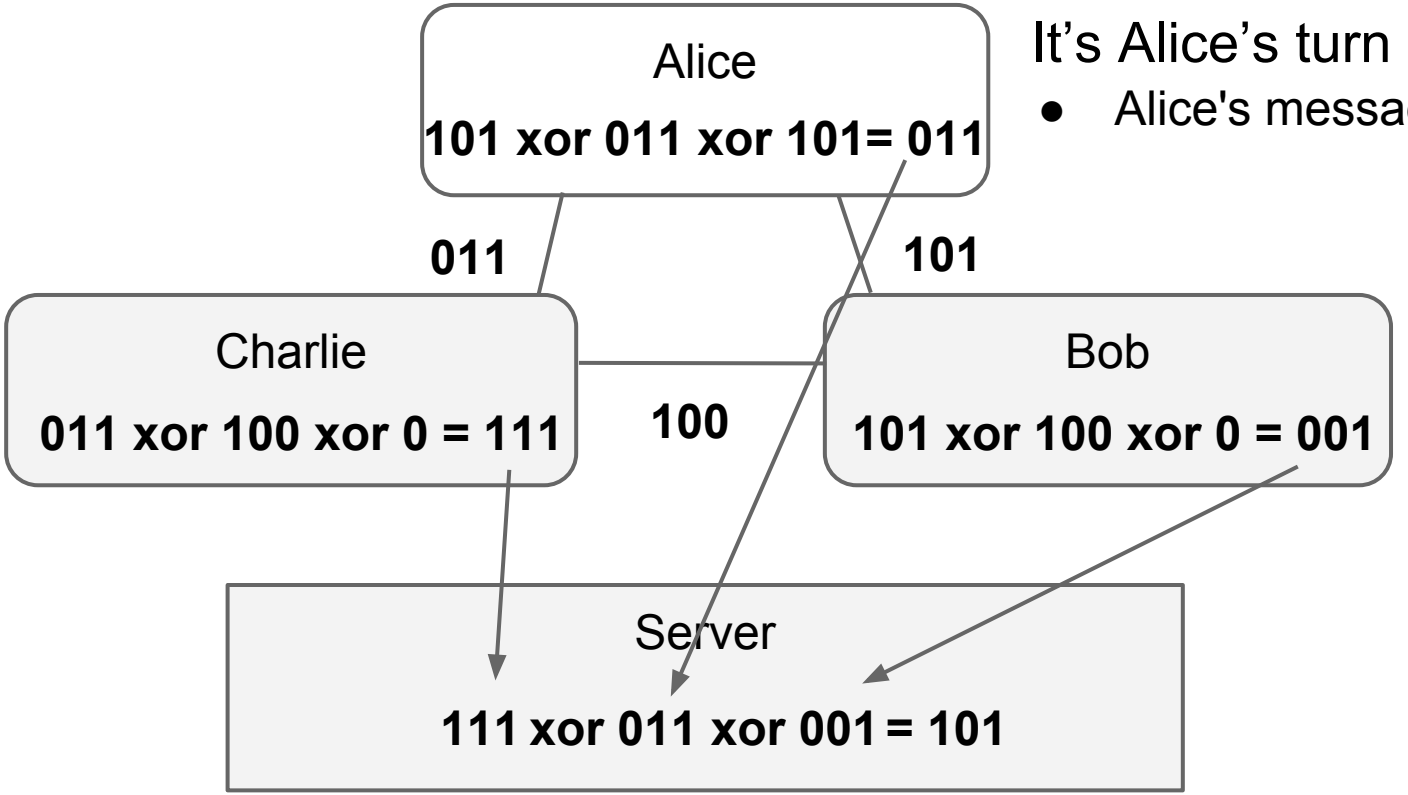
Scheduling (1/2)



Scheduling (2/2)

- Send message when not user’s turn:
  - All messages added to waiting queue, sent when turn comes
- What if user is stalling?
  - (Small) time limit passed → next user given priority. (High turn circulation speed → process seems fluid)

DC-Net Communication



It’s Alice’s turn to send message:

- Alice's message is 101 (in binary)

Implementation

Implementation

- Language: Python
- Networking Framework: xmlrpc python
- Anonymity primitive: DC-Nets
- Shared Key Algorithm: Diffie Hellman
- Libraries:
  - Socket Server
  - Queue
  - SimpleXMLRPCServer and xmlrpclib
  - sys
  - randint

Current Status

- Functional chat room with a fixed amount of clients
  - DC-Nets implemented
  - Diffie Hellman implemented
- Guaranteed anonymity with a trusted server
  - Server knows the schedule

Future Work

- Secure schedule creation
  - No single party learns the schedule
- Detect disruption
  - Detect senseless XOR
- Dynamic chat room
- “Fake” clients
  - Increase the anonymity set size
- Scalability study

Conclusion

- SecretRoom provides strong anonymity
  - Attacker cannot distinguish between someone who is chatting & someone who is not.
  - Network activity is strictly indistinguishable.
- Baseline implementation done

Thank you!

Any questions?

Reducing Diffie Hellman Secret Sharing

- Use Pseudo-Random Number Generator
- Using same generation “seed” clients can make new shared secret keys with a PRNG.
  - Faster
  - Less network communication

Diffie-Hellman

- Choose prime $p = 23$ and generator $g = 5$
- Alice chooses a secret $a = 6$, then sends

  $A = 5^6 \bmod 23 = 8$ to Bob
- Bob chooses a secret $b = 15$, then sends

  $B = 5^{15} \bmod 23 = 19$ to Alice
- Alice computes $s = 19^6 \bmod 23 = 2$
- Bob computes $s = 8^{15} \bmod 23 = 2$
- Alice and Bob now share a secret (the number **2**).