Dissent in Numbers: Making Strong Anonymity Scale

David Isaac Wolinsky, Henry Corrigan-Gibbs, Bryan Ford, Aaron Johnson

Presented by Maciek Swiech
Motivation

- "Anonymous Communication Should Be Regarded as a Strong Human Right..."
  - Authoritarian governments, profit-based tracking systems want to access communication
- Anonymity systems depends on a large number of users working together
  - small strong systems can lead to less anonymity than large weak systems
Motivation - example

Meet tonight at 7 PM in the park for pizza and beer!
Motivation - example

What if we use strong anonymity to pass the message to our friends?

Meet tonight at 7 PM in the park for pizza and beer!

Bob, you’re going be spending some time in the slammer!
Motivation - example

Meet tonight at 7 PM in the park for pizza and beer!

Does it matter who sent the message if it is intercepted?
Motivation - example

But what if we send the message to lots of people...

Meet tonight at 7 PM in the park for pizza and beer!

All of you going to be spending time in the slammer!!!
Motivation - example

Meet tonight at 7 PM in the park for pizza and beer!

Ugh, we can’t put them all in Jail...

This party is over go home!!!
Goals

● Strong anonymity
● Scalability
● Churn tolerance
● Accountability

● Avoid prevailing tradeoff between scale and strength
Other approaches?

- Tor onion router
- Dining Cryptographers (DC-Net)
- Mix-Nets
Tor / Mix-nets

Bob connects to anonymity relays through multiple servers before reaching the public server.

Meet tonight at 7 PM in the park for pizza and beer!
DC-Net

- Pairwise 'coins' used to encode message
  - coin toss generated with shared PRNG
  - coins and message XORed to encode message
  - on decode XORed values cancel
  - unable to tell who sent the original message

XOR \{\text{shared values, cleartext}\} \rightarrow \text{ciphertext}
DC-Net

Cleartext message

Traffic analysis resistant since all member transmit equal length messages

Alice 1

Bob 1

Carol 1
Other approaches?

- **Tor onion router**
  - Scales well
  - Vulnerable to timing attacks

- **Dining Cryptographers (DC-Net)**
  - Not scalable
  - No accountability - vulnerable to DoS
  - No churn tolerance

- **Mix-Nets**
  - Not scalable
  - No accountability - vulnerable to DoS
Other approaches - shortcomings

<table>
<thead>
<tr>
<th>Feature</th>
<th>Mix-nets</th>
<th>Tor</th>
<th>DC-nets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong anonymity</td>
<td>✔️</td>
<td></td>
<td>✔️</td>
</tr>
<tr>
<td>Scalability</td>
<td></td>
<td>✔️</td>
<td>✔️¹</td>
</tr>
<tr>
<td>Churn tolerant</td>
<td>✔️</td>
<td>✔️</td>
<td></td>
</tr>
<tr>
<td>Accountability</td>
<td></td>
<td></td>
<td>✔️²</td>
</tr>
</tbody>
</table>
Dissent - key idea!

- Can improve both anonymity and scalability by mixing DC-Net and Mix-Net infrastructure
Dissent - architecture

- Clients
  - only need to communicate with one upstream server

- Servers
  - well provisioned machines
  - communicate with N clients and amongst each other
  - coordinate communication between client groups
Dissent - client algo

1. Scheduling
   a. client creates pseudonym key
   b. sends key to shuffle protocol -> encrypted ordering

2. Submission
   a. based on ordering, craft message bits
   b. create pseudo-random string for each server
   c. XOR strings and message, sign, send to server

3. Output
   a. wait for cleartext form any server, signed by all
   b. extract messages in cleartext
   c. prepare for next round
Dissent - server algo

1. Submission
   a. wait for enough ciphertexts to arrive

2. Inventory
   a. make list of all clients that sent messages
   b. send list to other servers

3. Commitment
   a. eliminate redundant entries amongst servers
   b. if not enough messages, drop and go to step 1
   c. create server ciphertext $s_j$, send commit hash to servers

$$s_j = (\bigoplus_{i \in l} s_{ij}) \oplus (\bigoplus_{i \in l_j} c_{ij})$$
4. Combining
   a. when all servers send commit hashes, share $s_j$

5. Certification
   a. verify server hashes
   b. decrypt message, sign, send signature to servers

6. Output
   a. collect all server signatures
   b. send cleartext to directly connected clients
Dissent - challenges

1. Computational Efficiency
2. Communication Efficiency
3. Churn Handling
4. Disruption Identification
5. Strong Anonymity
Computational Efficiency

- **DC-Net**
  - each client has a shared secret with every other client
  - computing all shared secrets with PRNG is $O(N^2)$

- **Dissent**
  - clients only need send messages to servers
  - shared secret computation reduced to $O(M*N)$
Communication Efficiency

- **DC-Net**
  - In order to distribute ciphertext and all necessary coded messages to every member, $O(N^2)$ messages are necessary.

- **Dissent**
  - Since clients only need to communicate with one server, and servers communicate with one another, can create pseudo-multicast trees.
  - Greatly reduce communication to $O(N + M^2)$.
Churn Handling

- **DC-Net**
  - client drops -> messages must be dropped and re-encoded
  - missing message -> one half of shared secret won't be canceled with XOR -> garbled message

- **Dissent**
  - servers wait for missing client to time out, don't include in encoding process
Disruption Identification

- DC-Net
  - cannot identify who is accountable for a malicious (or erroneous) disruption
  - susceptible to DoS attacks

- Dissent
  - use scheduling protocol and accusation to quickly and efficiently identify problem clients
Disruption Identification

- Parity bit check indicates disruption has occurred
- Identify a 'witness bit'
  - crypto message padding used
- Broadcast an accusation through the shuffle
- Servers verify that witness bit has been flipped
- Each server recomputes all its shared secrets and uses this to find mismatches
- Honest clients can 'rebut' false claims by sharing their shared secret with server
Strong Anonymity

- In DC-Nets, anonymity set is equivalent to all honest nodes
  - secret sharing graph guarantees anonymity
- In Dissent...
  - as long as one server is honest, anonymity set is maintained
  - secret sharing graph -> doesn't matter which server is honest!
Evaluation

- Dissent prototyped in C++
  - utilizing Qt framework
  - utilizing Crypto++ framework

- 2 orders of magnitude larger anonymity set
Evaluation - churn resistance

![Graph showing % client submission deadlines over time for different deadlines.]

- 120 seconds
- 1.1 * 95th percentile
- 1.2 * 95th percentile
- 2.0 * 95th percentile

Deadline cut-off
Evaluation - scalability

- Microblog
  - random 1% of clients share 128 byte message
- Data Sharing
  - one client shares 128K message each round
- 1000 clients -> 1 second delay
- Bandwidth dominates large messages
- Latency dominates small messages
Evaluation - server scalability

![Graph showing server scalability](image-url)
Evaluation - system overhead

- Key shuffle expensive
  - can be done rarely
- Blame shuffle expensive
  - ideally not done often
  - attack susceptibility?
Evaluation - WAN effects

- Fraction of responses received in fewer than X seconds
- Time to receive response (seconds)
- Size (bytes) of all index page content (HTML page, images, JS, CSS)
Evaluation - WAN effects

- Users who are willing to tolerate 35% reduction in speed can get benefits of both Tor and Dissent systems

<table>
<thead>
<tr>
<th>Method</th>
<th>No anonymity</th>
<th>Tor</th>
<th>Dissent</th>
<th>Tor + Dissent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average DL time</td>
<td>10s</td>
<td>40s</td>
<td>45s</td>
<td>55s</td>
</tr>
</tbody>
</table>
Limitations

● Server disruptions
  ○ servers going down causes large interruptions

● Internet-scale networks
  ○ Herbivore-like approach of breaking network into groups

● Intersection attacks
  ○ can correlate session information to deduce what clients participated in messages sent

● Group management and Server Selection
  ○ dynamic groups and trusting group membership
Limitations

● Formal security verification
  ○ based on work that has been proven, but no formal proof of Dissent

● Mobile Devices
  ○ no mobile deployment has been attempted
Conclusion

- Dissent provides:
  - Strong anonymity
  - Scalability (Computation and Communication)
  - Churn tolerance
  - Disruption identification
Questions / Discussion

● Authoritarian regimes
  ○ how many people is too many to throw into jail?
  ○ "slap-on-the-wrist" for using anonymity

● Upstream servers?
  ○ "clients communicate with only a single server..."
  ○ "...which i signs and transmits to one or more servers."