Dissent in Numbers: Making Strong Anonymity Scale

David Wolinsky¹, Henry Corrigan-Gibbs¹, Bryan Ford¹, and Aaron Johnson²
¹Yale University, ²US Naval Research Laboratory
Motivation – Strength in Numbers

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

Bob, you’re going be spending some time in the slammer!
Motivation – Strength in Numbers

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

All of you going to be spending time in the slammer!!!
Motivation – Strength in Numbers
Meet tonight at 7 PM in the park for pizza and beer!

Motivation – Strength in Numbers

Ugh, we can’t put them all in Jail...

This party is over go home!!!
Making Strong Anonymity Scale?

• Challenge – tradeoff between scale and strength in anonymity systems favoring scale

• Goals
  • Strong anonymity (timing analysis resistant)
  • Scalability (100s to 1,000s of active participants)
  • Churn tolerant (unannounced member departures)
  • Accountability

Achieved in Dissent!
Organization

• Motivation
• Existing Approaches
• Dissent – Strong, Scalable Anonymity
  • Computational efficiency
  • Communication efficiency
  • Churn tolerant
  • Anonymity
  • Accountability
• Evaluation
• Conclusions
Organization

• Motivation

• **Existing Approaches**
  • Dissent – Strong, Scalable Anonymity
    • Computational efficiency
    • Communication efficiency
    • Churn tolerant
    • Anonymity
    • Accountability

• Evaluation

• Conclusions
Tor – The Onion Router

Anonymizing Relays

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

Tor is scalable, supports more than 400,000 clients with 1,000 clients per server
Bob Tor – The Onion Router

Anonymizing Relays

Not timing analysis resistant!

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

Aha! Got you!

State-run ISP
DC-net

Cleartext message

Traffic analysis resistant since all members transmit equal length messages

Alice

Carol

Bob
DC-net

Cleartext message

Traffic analysis resistant since all member transmit equal length messages

Alice

Bob

Carol
Practical Considerations

<table>
<thead>
<tr>
<th></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>

- **Mix-nets / Shuffles** – Chaum, Neff, Wikstrom
- **Onion Routing** – Tor and I2P
- **DC-nets** – \(^1\)Herbivore and \(^2\)Dissent v1
  - Herbivore supported many concurrent users but distributed amongst many parallel DC-nets thus lacks the “Strength in Numbers” or large anonymity set sizes
Organization

- Motivation
- Existing Approaches
- **Dissent – Strong, Scalable Anonymity**
  - Computational efficiency
  - Communication efficiency
  - Churn tolerant
  - Anonymity
  - Accountability
- Evaluation
- Conclusions
Key Insight
Key Insight

Use DC-net style anonymity within the Mix-net topology to obtain scalability!
Making Strong Anonymity Scale!

- Challenge – tradeoff between scale and strength in anonymity systems favoring scale
- Dissent’s solution
  - Improving Computation Efficiency
  - Improving Communication Efficiency
  - Handling Churn
  - Identifying Disruptions
  - Maintaining Strong Anonymity
Improving Computational Efficiency
Computational Overhead

Computation overhead due to $O(N^2)$ secret shares
Computational Overhead

Computation overhead due to $O(N^2)$ secret shares

$N = 100$, 4950 shared secrets, 9900 RNG operations
5.5 ms/peer
Computational Improvement

Each server has $N$ secrets with $M \ll N$

With $M$ servers and $N$ clients...

$O(M*N)$ shared secrets with $M \ll N$

$N = 100$ and $M = 5$

500 shared secrets, 1000 RNG operations

RNG reduction: $1000 \ll 9900$

Each client has $M$ secrets

Each server has $N$ secrets
Improving Communication Efficiency
Bandwidth Overhead

Computation overhead due to $O(N^2)$ secret shares

Bandwidth overhead due to $O(N^2)$ communication

$N = 100$, Ciphertexts exchanged in DC-nets: 9900

- Crystal
- Ben
- Anna
- Bob
- Brett
- Carol
- Amy
- Alice

Cleartext

N = 100, Ciphertexts exchanged in DC-nets: 9900

- Crystal
- Ben
- Anna
- Bob
- Brett
- Carol
- Amy
- Alice

Cleartext
We can construct a DC-net aware multicast tree!

Earlier DC-nets had $O(N^2)$ communication cost.
Bandwidth Efficiency

Earlier DC-nets had $O(N^2)$ communication cost

We can construct a DC-net aware multicast tree!

N = 100 and M = 5
Ciphertexts exchanged in DC-nets: 9900, Dissent: 205

Clients submit their ciphertext upstream to one server

Servers XOR these messages together and share with each other

Servers XOR these messages to compute the cleartext and distribute it to their downstream clients
Creating Churn Tolerance
Churn Intolerance

- Computation overhead due to $O(N^2)$ secret shares
- Bandwidth overhead due to $O(N^2)$ communication
- What if Alice left without transferring?

The resulting cleartext is garbage due to the dependency on Alice’s secret shares.
Tolerating Churn

Server$_1$ will timeout on Alex

The protocol continues uninterrupted, since the servers have yet to compute their ciphertext.
Handling Disruptions via Accountability...
DC-net

Easily disrupted

Alice

Bob

Carol

1

0

1

0

1

0

0
How can we prove Bob transmitted the wrong ciphertext without losing anonymity?
Scheduling

Anonymizing shuffle produces random permutation and hence the schedule

How do many members share the DC-net without disrupting each other?

Create a transmission schedule!
DC-net

Integrity check (parity bit)

Integrity check failed!
To determine the disruptor Alice needs to anonymously specify a bit that the disruptor "flipped" from 0 -> 1
Safely Deanonymize a Bit

Alice

\{\text{Bit}_1\}_{\text{Alice}}

0

0

Bob

Shuffle

\{\text{Bit}_1\}_{\text{Alice}}

Carol

0

0
DC-net

In practice, this is a bit more complicated though the details are in the paper.
In practice, this is a bit more complicated though the details are in the paper.

If Carol reveals the shared secret, Alice can confirm that Bob disrupted the previous round.

1 with Bob
1 with Carol

1 with Alice
0 with Carol

1 with Alice
1 with Bob
Progress!

- We have gained
  - Improvements in computation and communication
  - Ability to tolerate churn
  - Identify disruptors
- How does this impact strong anonymity?
DC-net – Anonymity Set

Anonymity set size: 4
(Honest participants)
Dissent retains this feature...
Dissent – Anonymity Set

Anonymity set size: 11 (Honest participants)

Secret sharing graph prevents the clients upstream server from deanonymizing it

Anonymity set remains equal as long as there is 1 honest server
Organization

• Motivation
• Existing Approaches
• Dissent – Strong, Scalable Anonymity
  • Computational efficiency
  • Communication efficiency
  • Churn resistance
  • Anonymity
  • Accountability

• Evaluation
• Conclusions
Dissent – Prototype

- Written in C++
  - Qt from networking, serialization, and events processing
  - Crypto++ as the crypto library
Related Work

- **Herbivore TR’03**: Evaluated only up to 40 members.
- **Dissent CCS’10**: Graph showing time (in minutes) vs. number of nodes.
Scaling to Thousands of Clients

- Bandwidth limitations
- CPU Overheads
- Latency limited

1,000 clients: ~1 second
> 5,000 concurrent clients!!
CPU Time

![Graph showing CPU Time per exchange (seconds) vs. Number of clients. The graph has two lines: one for Servers and one for Clients. The Servers line slopes upward with increasing number of clients, suggesting an increase in CPU Time. The Clients line remains relatively flat, indicating minimal change in CPU Time with increasing clients.]
Comparison to Shuffles

Dissent keeps up! Verifiable shuffles do not
Churn Resilience

Nearly 99% complete in less than 1 second

Nearly 50% complete in less than 400 ms
Protocol Breakdown

- "Fast" DC-net
- Slow Key Shuffle
- Really slow blame shuffle
- Efficient disruption analysis
- "Fast" DC-net round
- Key shuffle

Graph showing time in seconds versus number of clients.
Organization

• Motivation
• Existing Approaches
• Dissent – Strong, Scalable Anonymity
  • Computational efficiency
  • Communication efficiency
  • Churn resistance
  • Anonymity
  • Accountability
• Evaluation
• Conclusions
Key Take Aways

• We can construct strong and scalable anonymous communication systems
  • $O(N^2)$ communication cost to $O(N)$
  • Churn tolerance
  • Provides an effective means to identify disruptors
• Two orders of magnitude larger anonymity sets than previous DC-net approaches
• Maintains strong anonymity properties from DC-nets
Future Work

• Further bandwidth and computation optimizations
• Slot length scheduling policies
• Better ways to anonymously distribute blame
• Handling long term intersection attacks
• Formal security analysis
• Making available for real applications and real users
Finished!

Thanks, questions?

Dissent – Strong, scalable accountable anonymity
Find out more at
http://dedis.cs.yale.edu/2010/anon/
We’ll be at the poster session tonight!
Extra slides
Servers might be run within a single cloud but owned by different “anonymity providers”.

100 Mbit/sec LAN with 10 msec delay

100 Mbit/sec shared upstream link with 50 msec delay

8 – 16 servers
1 – 320 clients per server
24 – 5120 clients
Scaling to Thousands of Clients

![Graph showing the relationship between the number of clients and time per round in seconds, with markers indicating CPU Overheads and Bandwidth limitations.]

- 128K message - Server processing (DeterLab)
- 128K message - Client submission (DeterLab)
- 1% submit - Server processing (PlanetLab)
- 1% submit - Client submission (PlanetLab)
- 1% submit - Server processing (DeterLab)
- 1% submit - Client submission (DeterLab)
Each server performs in serial expensive decryption operations.
Dissent

Clients connect to a single upstream server

Servers connect with each other

Alice

Carol

Bob
Dissent

Clients have a shared secret with each server

Diffie-Hellman public keys exchanged during registration

Secrets:
- $\text{Secret}_{A0}$
- $\text{Secret}_{A1}$
- $\text{Secret}_{A2}$

Servers:
- Server$_0$
- Server$_1$
- Server$_2$
Dissent

Ciphertext_{A} = blame, nonce, next slot length, msg, hash
Ciphertext_{A0} = RNG(Secret_{A0}, length)
Ciphertext_{A} = Ciphertext_{A0} XOR Ciphertext_{A1} XOR Ciphertext_{A2} XOR (0, ..., 0, Cleartext_{A}, 0, ..., 0)
Dissent

Server_0

Server_1

Server_2

Alice

Ciphertext_A = blame, nonce, next slot length, msg, hash
Ciphertext_{A0} = RNG(Secret_{A0}, length)
Ciphertext_{A} = Ciphertext_{A0} XOR Ciphertext_{A1} XOR
Ciphertext_{A2} XOR (0, ..., 0, Cleartext_A, 0, ..., 0)
Dissent

Server_0

Server_1

Server_2

Ciphertext_A

Ciphertext_C

Ciphertext_B

Alice

Carol

Bob
Dissent

Client list exchange

Server 0

Server 1

Server 2

[Carol] [Alice] [Carol]

[Alice] [Bob] [Alice]
Server_0 knows that Alice, Bob, and Carol submitted:
Ciphertext_0 = Ciphertext_A XOR Ciphertext_{A0} XOR Ciphertext_{B0} XOR Ciphertext_{C0}

Commit_0 = Hash(Ciphertext_0)
Server$_0$ knows that Alice, Bob, and Carol submitted:
Ciphertext$_0 = \text{Ciphertext}_A \text{ XOR Ciphertext}_A^0 \text{ XOR Ciphertext}_B^0 \text{ XOR Ciphertext}_C^0$

Commit$_0 = \text{Hash}(\text{Ciphertext}_0)$
Server_0 knows that Alice, Bob, and Carol submitted:
Ciphertext_0 = Ciphertext_A XOR Ciphertext_{A0} XOR Ciphertext_{B0} XOR Ciphertext_{C0}

Commit_0 = Hash(Ciphertext_0)

Cleartext = Ciphertext_0 XOR Ciphertext_1 XOR Ciphertext_2

Signature_0 = \{Cleartext\}_{Key0}
Dissent

Client list exchange
Ciphertext evaluation
Ciphertext commit
Ciphertext exchange

Cleartext evaluation
Cleartext commit
Cleartext distribution
Dissent – Blame

Cleartext_A = blame, nonce, next slot length, msg, hash
Ciphertext_{A0} = RNG(Secret_{A0}, length)
Ciphertext_A = Ciphertext_{A0} XOR Ciphertext_{A1} XOR Ciphertext_{A2} XOR (0, ..., 0, Cleartext_A, 0, ..., 0)
Identifying Disruptors
Dissent – Blame

\[\text{Ciphertext}_{A} = \text{blame, nonce, next slot length, msg, hash}\]

\[\text{Ciphertext}_{A0} = \text{RNG(Secret}_{A0}, \text{ length)}\]

\[\text{Ciphertext}_{A} = \text{Ciphertext}_{A0} \text{ XOR Ciphertext}_{A1} \text{ XOR Ciphertext}_{A2} \text{ XOR (0, ..., 0, Cleartext}_{A}, 0, ..., 0)\]
Round will complete as normal, but everyone will see the blame flag set, resulting in a blame shuffle.

In the blame shuffle, the slot owner will specify a bit to deanonymize which will reveal the offending client / server.

The message in the shuffle is signed with the slot owner’s anonymous meaning it is safe to deanonymize.
Related Work – Herbivore

With 40 members, communication delays between .6 and 1.2 seconds.
Related Work – Earlier Dissent

With 44 members, communication delays for the DC-net were 2 minutes.
Future Work in Dissent

- Disruption resistance is online, requires additional steps after the protocol has completed.
- Practical use in real environments – Such as using WiFi enabled smart phones.
- Anonymity boxes – isolated environments running within a virtual machine isolating the user’s private information from the anonymity network.
- Participation limits to prevent Sybil attacks.
Dissent Disruption Resistance

• A malicious bit flip resulting from a 0 -> 1 in the cleartext can be used to generate an accusation
  • In a DC-net, client requests accusation shuffle
  • In shuffle, client specifies the flipped bit
• Servers share bits for this bit index, finding either
  • A server sent bits that do not match his ciphertext – thus he is guilty of the disruption
  • A client’s ciphertext does not match the accumulation of the server’s bits
• Clients rebut by sharing with servers the shared secret of the offending server, accepting blame, or remaining suspect
## Analytical Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>DC-Nets</th>
<th>Herbivore</th>
<th>Dissent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messages</td>
<td>$O(N^2)$</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>Secrets</td>
<td>$O(N^2)$</td>
<td>$O(N^2)$</td>
<td>$O(N*M)$</td>
</tr>
<tr>
<td>Anon</td>
<td>$O(K)$</td>
<td>$O(K)$</td>
<td>$O(K)$, assuming 1 honest server</td>
</tr>
</tbody>
</table>

$N = $ Members (clients)  
$M = $ Servers  
$K = $ honest members
Server Count Effects

![Graph showing the effects of server count on time per round in seconds]

- 128K message - Server processing
- 128K message - Client submission
- 1% submit - Server processing
- 1% submit - Client submission

- 6.25 second
- 9 seconds
- 5.5 second
- 220 ms
## Analytical Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>Dissent</th>
<th>D3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shuffle</td>
<td>Comm O(N) serial steps</td>
<td>O(1)</td>
</tr>
<tr>
<td></td>
<td>Anon O(K), K = honest members</td>
<td>O(K), K = honest members, assuming 1 honest server</td>
</tr>
<tr>
<td>DC-net</td>
<td>Comm O(N^2) messages</td>
<td>O(N) messages</td>
</tr>
<tr>
<td></td>
<td>O(N^2) shared secrets</td>
<td>O(N) shared secrets</td>
</tr>
<tr>
<td></td>
<td>Anon O(K), K = honest members</td>
<td>O(K), K = honest members, assuming 1 honest server</td>
</tr>
</tbody>
</table>
Client/Server Trust Models

• Trust all servers
  • Unrealistic in the real world
• Trust no servers – SUNDR
  • Ideal but complicated due to lack of knowledge and message time constraints
• Trust at least one server – Anytrust
  • With one honest server, anonymity set is equal to the set of all honest members (clients)
  • No need to know which server to trust
  • (Used in Mix-nets)
DC-Nets Generalized

- Members share secrets with each other
  - Such as Diffie-Hellman exchanges
  - Can be used to generate variable length string
- Each member constructs a ciphertext
  - XOR in the string generated by each shared secret
  - Optionally, XOR secret message
- Positions inside a DC-net can be assigned via randomness (Ethernet style backoff) or a Mix-Net
- After obtaining a copy of each ciphertext
  - XOR each ciphertext together
  - Effectively, cancelling out generated strings
  - Revealing secret messages
## Existing Approaches

<table>
<thead>
<tr>
<th>Method</th>
<th>Weakness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mix-Nets, Tor</td>
<td>Traffic analysis attacks</td>
</tr>
<tr>
<td>Group / Ring Signatures</td>
<td>Traffic analysis attacks</td>
</tr>
<tr>
<td>Voting Protocols</td>
<td>Fixed-length messages</td>
</tr>
<tr>
<td>DC Nets</td>
<td>Anonymous DoS attacks</td>
</tr>
<tr>
<td>Dissent</td>
<td>Intolerant to churn / long delays</td>
</tr>
<tr>
<td></td>
<td>between msgs</td>
</tr>
<tr>
<td>Herbivore</td>
<td>Small anonymity set</td>
</tr>
</tbody>
</table>