Atom

Horizontally Scaling Strong Anonymity

Albert Kwon
MIT
Srinivas Devadas
MIT

Henry Corrigan-Gibbs
Stanford
Bryan Ford
EPFL

10/30/17, SOSP’17
Motivation

Anonymous bulletin board (broadcast) in the face of global adversary

Protest at 4 p.m.!
Anonymous communication networks

Anonymity provider (set of servers)
Existing systems vs. Atom

<table>
<thead>
<tr>
<th>Properties</th>
<th>Tor [USENIX Sec’04]</th>
<th>Riposte [Oakland’15]</th>
<th>Atom</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scaling</td>
<td>Horizontal</td>
<td>Vertical</td>
<td>Horizontal</td>
</tr>
<tr>
<td>Latency (1 million users)</td>
<td>&lt; 10s</td>
<td>11 hrs</td>
<td>28min</td>
</tr>
<tr>
<td>Anonymity against global adversaries</td>
<td>Vulnerable</td>
<td>Secure</td>
<td>Secure</td>
</tr>
</tbody>
</table>
Deployment and threat model

- Global network adversary
- A large number of users are malicious
- Constant fraction of the servers are malicious
  - 20%
Atom overview
Atom overview

Unknown random permutation of all inputs
Horizontally scalability

Width

More servers => Larger width

Depth

Fixed (Independent of the width)
Challenges

1. Guaranteeing anytrust property
Challenges

1. Guaranteeing anytrust property
2. Group mixing and routing protocol
Challenges

1. Guaranteeing anytrust property
2. Group mixing and routing protocol
3. Active adversaries
Active attacks
Challenges

1. Guaranteeing anytrust property
2. Group mixing and routing protocol
3. Active adversaries
4. Tolerating server churn
Challenges

1. Guaranteeing anytrust property
2. Group mixing and routing protocol
3. Active adversaries
4. Tolerating server churn
Generating anytrust groups

20% malicious

Randomly select $k$ servers

Public randomness

$Pr[\text{group is fully malicious}] = 0.2^k$

$Pr[\text{any group is fully malicious}] < (# \text{ of groups}) \cdot 0.2^k < 2^{-64}$
Handling actively malicious servers

Trusted third party

Idea: use verifiable trap messages
Send trap and real messages in a random order

Trusted third party

& $ # @

: encrypted for TTP
TTP checks for the traps

: encrypted for TTP

Trusted third party

& $ # @

@ $ 3 & 1 # 2 4
What happens when a trap message is dropped?

: encrypted for TTP

Trusted third party

& $ # @

@ $ 3 & 1 # 2 4
What happens when a real message is dropped?

0: encrypted for TTP

Trusted third party

0 @ $ # @

0 3 @ & # 2 4
Improving the trap messages

- Distributing the trust in the third party
- Distributing the trap verification and decryption
Properties of trap-based defense

- If the adversary tampers with any trap, then no plaintext revealed
- Can remove 1 message with probability $\frac{1}{2}$
  - Remove $t$ messages with probability $2^{-t}$
  - Realistically remove $< \sim 64$ msgs
- Reactive
Two modes of operation

<table>
<thead>
<tr>
<th>Idea</th>
<th>Trap messages</th>
<th>Zero-knowledge Proof</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verify untamperable traps</td>
<td></td>
<td>Verify protocol with ZKP</td>
</tr>
<tr>
<td>Anonymity set size</td>
<td>N - t</td>
<td>N</td>
</tr>
<tr>
<td>Defense type</td>
<td>Reactive</td>
<td>Proactive</td>
</tr>
<tr>
<td>Latency</td>
<td>1x</td>
<td>4x</td>
</tr>
</tbody>
</table>
Implementation

- ~4000 lines of Go
- Both trap and ZKP based defenses
- Code available at github.com/kwonalbert/atom
Evaluation setup

- Heterogenous set of 1024 EC2 servers
  - 80% of the servers were 4-core machines
- 20% malicious servers
- Trap messages
- 160-byte msgs

Depth = 10

32 server group
Latency is inversely proportional to the number of servers.

- For 128 servers, the latency is approximately 200 minutes.
- For 256 servers, the latency is approximately 150 minutes.
- For 512 servers, the latency is approximately 100 minutes.
- For 1024 servers, the latency is approximately 50 minutes.

Comparing Riposte with Atom (~1 million users):
- Riposte has a latency of 680 minutes.
- Atom has a latency of 50 minutes, which is 23x better than Riposte.
Latency scales linearly with the number of users
Limitations

- Medium to high latency
- Denial-of-service
Related work

● Strong anonymity but vertically scaling
  ○ Dissent[OSDI’12], Riffle [PETS’16], Riposte [Oakland’15], …

● Horizontally scaling systems but weaker anonymity
  ○ Crowds [ACM’99], Mixminion [Oakland’03], Tor [USENIX Sec’04], Aqua [SIGCOMM’13], Loopix [USENIX Sec’17], …

● Distributed mixing
  ○ Parallel mix-net [CCS’04], matrix shuffling [Håstad’06], random switching networks [SODA’99, CRYPTO’15], …

● Private point-to-point messaging
  ○ Vuvuzela [SOSP’15], Pung [OSDI’16], Stadium [SOSP’17]
Conclusion

- Atom provides horizontally-scaling strong anonymity
  - Global anonymity set
  - Latency is inversely proportional to the number of servers
- Supports 1 million users with 160 byte msgs in 28min

github.com/kwonalbert/atom
These icons were acquired from thenounproject.com, and are under CC BY 3.0 US

Created by H Alberto Gongora
Created by H Alberto Gongora
Created by Anil

Created by Andre Luiz Gollo
Created by Creative Stall