Publius: A robust, tamper-evident, censorship-resistant web publishing system
Marc Waldman, Aviel D. Rubin, Lorrie Faith Cranor

- Pavan Kumar Panjam
Need for publishing anonymously...

- Political Dissent
- Radical Ideas
- I'm sure, you could think of some
Is Internet Censorship Resistant?

- Can you say “No one can force me to change the content of my website?”
- “The Net treats censorship as damage and routes around it“
  - John Gillmore
- “On the Internet, nobody knows you're a dog”
  - Peter Steiner's famous New Yorker cartoon caption
Reality...

- Church of Scientology
- DMCA
Publius Design Goals

- Censorship Resistant
- Tamper evident
- Source anonymous
- Updateble
- Deniable
- Fault tolerant
- Persistent
- Extensible
- Freely Available
Related- (Web anonymizing tools)

- Connection Based Anonymity
  - Hide the identity of the individual requesting a web page

- Location or Author Based Anonymity
  - Hide the identity of author or web server
Connection Based Anonymity

• Proxy for HTTP requests
  – www.Anonymizer.com

• Proxymate - Formerly known as LPWA
  – Http proxy pseudonym generation
Contd..

- **Onion Routing**
  - Mix Network
  - Http proxy developed
  - [www.onion-router.net](http://www.onion-router.net)  [www.torproject.org](http://www.torproject.org)

- **Crowds**
  - Http request via crowd
  - Dynamic Path Generation

- **Freedom** ([http://www.freedom.net](http://www.freedom.net))
  - Implemented at transport layer
  - Supports multiple protocols and allows the creation of Psuedonymns
Author/Location Based Anonymity

- Rewebber (aka Janus) : www.rewebber.de
  - Both connection and author based tool
  - URL rewrite
  - www.rewebber.com/surf – encrypted $E_k(U)$

- Anderson's Eternity Service
  - Server based storage medium  - resistant to DOS attacks
  - Fee based
  - Files cannot be removed
• Usenet Eternity
  – Usenet is a storage medium
  – Formatted using PGP, SHA-1
  – Send to alt.anonymous.messages
  – Limitations...

• Freenet
  – Adaptive Network

• Intermemory
  – Self replicating persistant RAM
  – Share RAM
Publius Overview:

- Publishers, Servers, Retriever
- Assume availability of a static list of servers
- Encrypt document with secret key 'k'
- Secret key split into (m, k) shares (Shamir)
- Store encrypted document and share on m servers
- Form URL cryptographically tied to a document
- Distribute Publius URL
  - http://!
    anon!/AH2LyMOBWrJrwv=GTEaS2GlNNE=NIBsZlvUQP4=sVfdKF7o/kl=EfUTWGQU7LX=OCk7tkhWTUe=
Publishing

- Alice Encodes content M with key K: \( \{M\}_K \)
- Splits K in to n shares s.t k can recover K
- For each share:
  - \( name_i = \text{wrap}(H(M \cdot \text{share}_i)) \)
  - \( Location_i = (name_i \mod m) + 1 \)
- If atleast 'd' unique locations are not found, start over with another K
Contd..

- ...d?
  - d is the number of publius servers that will end up hosting the content (k <= d <= m)
  - Coupon collectors problem (y ln y shares).
  - Therefore, choose d and set n = d ln d.

- Alice publishes \( \{M\}_k \), share\(_i\), and some other information in directory name\(_i\), on the server at location\(_i\).

- The URL that is produced contains at least \( d \) name\(_i\) values concatenated together.
Contd..

<table>
<thead>
<tr>
<th>Available Server Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>m</td>
</tr>
</tbody>
</table>

- $\text{name}_1 = 1e0995d6698$
- $\text{name}_2 = 620a8a3d63b$
- $\text{name}_n = \text{de26fe4fc8c6}$
- $\text{location}_1 = 4$
- $\text{location}_2 = 12$
- $\text{location}_n = 7$

Servers:
- Server 4: 201.18.24.5
  - /publius/1e0995d6698/[M]k
- Server 7: 210.183.28.4
  - /publius/de26fe4fc8c6/[M]k
- Server 8: 209.185.143.19
  - /publius/620a8a3d63b/[M]k
- Server 12: 209.185.143.19
  - /publius/620a8a3d63b/[M]k
Retrieve

- Bob wishes to view the content of publius URL:
- He parses the URL and gets the name_i's and computes location_i's.
- Chooses K servers arbitrarily, retrieves \( \{M\}_K \) from one and k shares from all those.
- Now Recover K, decrypt and check name_i's.
- Verify name_i values of corresponding shares by recomputing name_i.
- Something wrong, try different k shares.
Update and Delete

- Update: 'update' file, H(Server_domain_name · PW)
- Delete: H(Server_domain_name · PW)
Mutually linked documents

- Publius names are dependent on content, so dependencies can't be resolved directly
- Use the update option
Threats and Limitations

- Share deletion or corruption
- Update file deletion or corruption
- Denial of service attacks
- Threats to publisher anonymity
- Rubberhose cryptanalysis
Conclusion

- Visit the following website: