Facade: High-Throughput, Deniable Censorship Circumvention Using Web Search

Ben Jones, Sam Burnett, Nick Feamster, Sean Donovan, Sarthak Grover, Sathya Gunasekaran, and Karim Habak

Georgia Institute of Technology

FOCI 2014
Censorship is a common problem
Difficult to hide that you are using Tor

Tor is supposed to hide you online. In this Harvard student’s case, it did the opposite.

Bob

Alice
HTTP circumvention tools are necessary or they soon will be. Bob
We need to target new points on performance/deniability curve

- **StegoTorus**
  - StegoTorus is vulnerable to attackers that can retain state or compute entropy of traffic.

- **Infranet**
  - Infranet has very low performance for random data.

- **Collage**

- **Facade**

*Note: graph drawn for emphasis, not to scale*
Research Problems

• How can we create deniable, HTTP covert channels?

• Can we get the deniability of Infranet with better performance for encrypted data uploads?
Our Solution

Facade Server

Tunnel through web search

• Everyone searches the web and search has dozens of bits of entropy
• Let’s use this entropy to hide information
Outline

• Motivation
• Facade protocol
• Evaluation
Threat Model

• Our target censor can:
  – Detect and block all protocols other than HTTP
  – Store some state (several HTTP request/response pairs)
    • Ex: detect that information in cookies has not been set by the server
  – Censor can operate in-path
    • Ex: create error conditions to fingerprint client or server
Facade Overview

• Facade encodes information in web search

• Real users browse and search at the same time so Facade encodes information in browsing and search

• Note: Facade server must have sufficient cover search traffic to maintain deniability
Encoding data in search

• Encode information in the path string with a dictionary encoding

• The dictionary is a mapping from data to English

• Example: http://www.example.com/?q=banana+law encodes the string “hello”
Making search deniable with OpenSearch

• What is it?
  – Specification for sending search requests

• How does it work?
  – Encodes query into a URL

• Why are we using it?
  – Widely deployed: Chrome, Baidu, Yandex, etc.
OpenSearch Example

Query: where to buy peanuts

URL: https://duckduckgo.com/?q=where+to+buy+peanuts
System Overview

0. User makes request for http://www.epochtimes.com

1. Facade breaks request into chunks for transmission

2. Facade client sends the first chunk "http://epochtimes.com"

3. Facade server decodes the first chunk

4. Facade server assembles the first chunks together

5. Facade server makes request for http://www.epochtimes.com and returns the content via an image encoding

8. Facade server makes request for http://www.epochtimes.com and returns the content via an image encoding

Encode “epochtimes.com” as http://example.com/q=panda+cookie

Encode “epochtimes.com” as http://example.com/q=shoe+coffee
Outline

• Motivation
• Facade protocol
• Evaluation
Performance Evaluation: Methods

- Evaluated entropy of a search request using AOL search corpus

- 20 million queries from 650k users

- Get the average information content (entropy) per query
## Performance Evaluation: Results

<table>
<thead>
<tr>
<th>Tool</th>
<th>Entropy Per Request (bits)</th>
<th>Request Rate to Equal 256kbps</th>
<th>Deniability Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facade</td>
<td>(1^{\sim}78)</td>
<td>3,300</td>
<td>HTTP+Search</td>
</tr>
<tr>
<td>Infranet</td>
<td>(2^{3})</td>
<td>85,300</td>
<td>HTTP+Browse</td>
</tr>
<tr>
<td>StegoTorus</td>
<td>(3^{12000})</td>
<td>21</td>
<td>HTTP</td>
</tr>
</tbody>
</table>

1 The paper contains an error wherein the entropy is reported with \(\log\) base e instead of 2
2 Infranet entropy computed with parameters from paper, i.e. 8 links per page, so \(\log_2(8)=3\) bits
3 StegoTorus entropy calculated based upon Base64 encoding 2000 characters per URL
Future Work: Tradeoffs

• Tune performance/deniability with dictionary choices
  – Per user/site dictionaries
  – Dictionaries with joint PDFs
Conclusion

• Facade: an HTTP covert channel that balances performance and deniability by improving upload performance

• Get Facade (in development) from https://github.com/ben-jones/facade

• Contact me: Ben Jones bjones99@gatech.edu