Website Fingerprinting Attacks and Defenses in the Wild

Marc Juarez
imec-COSIC KU Leuven

TU Wien - 1st December 2017, Vienna, Austria
About Me

• PhD student at COSIC, KU Leuven

• Research interests: machine learning + privacy and security
  - Private web search, browser fingerprinting, website fingerprinting
Outline

1. Introduction to Website Fingerprinting (WF)
2. Critical Analysis of WF attacks
3. Defenses against WF
4. ‘Fingerprintability’ analysis
Website Fingerprinting
Adversary Model

Tor network

Examples:
- Autonomous System (AS)
- Internet Service Provider (ISP)
- Local network admin
Website Fingerprinting

Tor network

Entry

Middle

Exit

Web

Web page models
Website Fingerprinting: Training

- Number of Packets
- Average Packet Size
- % of Incoming Packets
- Timing of Packets...

Tor network

Web
Website Fingerprinting: Testing

Tor network

Entry  Middle  Exit

Web
State-of-the-Art Attacks

- k-NN (Wang et al., 2015)
- CUMUL (Panchenko et al., 2016)
- k-Fingerprinting (Hayes and Danezis, 2016)
kNN (Wang et al., 2015)

- Features
  - 3,000
  - total size, total time, number of packets, packet ordering
  - traffic bursts

- Classifier
  - $k$-Nearest Neighbors (k-NN)

- Accuracy
  - From 90% to 93% (100 sites)
k-Fingerprinting (Hayes et al, 2016)

- Features
  - 175
  - **Timing** and Size based

- Classifier
  - Random Forest (RF) + k-NN

- Accuracy
  - From 90% to 95% (100 sites)
CUMUL (Panchenko et al, 2016)

• Features
  – 100 interpolation points of the cumulative sum of packet lengths
  – Size
• Classifier
  – Support Vector Machine (SVM)
• Accuracy
  – From 90% to 95% (100 sites)
Why Do We Care?

• Tor is the most popular anonymity network and aims to protect against such adversaries

• It threatens HTTPS, fingerprinting interactions within a single site

• Evaluations show attack achieves over 90% accuracy

  –but… how concerned should we be in practice?
A Critical Analysis of Website Fingerprinting Attacks

Marc Juarez\textsuperscript{1}    Sadia Afroz\textsuperscript{2,3}    Gunes Acar\textsuperscript{1}
Claudia Diaz\textsuperscript{1}    Rachel Greenstadt\textsuperscript{3}

\textsuperscript{1}imec-COSIC KU Leuven
\textsuperscript{2}Berkeley University
\textsuperscript{3}Drexel University

Presented in CCS 2014, Scottsdale, AZ, USA
Unrealistic assumptions
Closed vs Open World

Closed world

Open world
Comparative Experiments

- Control

- Test
Comparative Experiments

- Control
  - Train: on data with default value
  - Evaluate: on data with **default value**

- Test
Comparative Experiments

- Control
  - Train: on data with default value
  - Evaluate: on data with **value of interest**

- Test
Unrealistic assumptions

Client settings: e.g., browsing behaviour
Client settings: multitab

- FF user uses an average of two to three tabs
Unrealistic assumptions

Adversary: e.g., replicability
Replicability: Network Conditions

VM New York
VM Leuven
VM Singapore

66.95%
9.33%

Control (LVN)
Test (SI)
Replicability: Network Conditions

VM New York
VM Leuven
VM Singapore

76.40% 68.53%

Control (SI) Test (NY)
Unrealistic assumptions

Web: e.g., staleness
Staleness and Personalization

Total trace size

13 Feb 2013

14 Feb 2013
Result Summary

Accuracy (%)

<table>
<thead>
<tr>
<th>Test setting</th>
<th>Control</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Staleness (9d)</td>
<td>80</td>
<td>40</td>
</tr>
<tr>
<td>Multitab (0.5s)</td>
<td>70</td>
<td>30</td>
</tr>
<tr>
<td>Multitab (3s)</td>
<td>70</td>
<td>30</td>
</tr>
<tr>
<td>Multitab (5s)</td>
<td>70</td>
<td>30</td>
</tr>
<tr>
<td>TBB version (Worst)</td>
<td>60</td>
<td>20</td>
</tr>
<tr>
<td>TBB version (Avg)</td>
<td>70</td>
<td>40</td>
</tr>
<tr>
<td>Network (Worst)</td>
<td>60</td>
<td>20</td>
</tr>
<tr>
<td>Network (Avg)</td>
<td>70</td>
<td>40</td>
</tr>
</tbody>
</table>
The Base Rate Fallacy

• Breathalyzer test:
  ○ 0.88 identifies truly drunk drivers (True Positives)
  ○ 0.05 sober drivers as drunk (False Positives)

• Alice gives positive in the test
  ○ What is the probability that she is indeed drunk?
  ○ Is it 0.95? Is it 0.88? Something in between?
The Base Rate Fallacy

- Breathalyzer test:
  - 0.88 identifies truly drunk drivers (True Positives)
  - 0.05 sober drivers as drunk (False Positives)

- Alice gives positive in the test
  - What is the probability that she is indeed drunk?
  - Is it 0.95? Is it somewhere in between?

Only 0.1!
The Base Rate Fallacy

- Circumference represents the world of drivers.
- Each dot represents a driver.
The Base Rate Fallacy

- 1% of drivers are driving drunk (base rate or prior).
The Base Rate Fallacy

- From drunk people 88% are identified as drunk by the test
From the sober people, 5% are erroneously identified as drunk
The Base Rate Fallacy

- Alice must be within the black circumference
- Ratio of red dots within the black circumference:

$$BDR = \frac{7}{70} = 0.1!$$
The Base Rate Fallacy

- Base rate must be taken into account
- In WF:
  - Blue: webpages
  - Red: monitored
  - Base rate?
Take aways

- WF attacks fail in realistic conditions
- We do not completely dismiss the attack
  - Targeted attacks may still be highly effective
  - Attack can be enhanced at a greater cost
- Defenses might be cheaper in practice
Discussion

1. What is the probability of visiting a *monitored* web page?

2. How confident is the adversary for each guess?

3. Can the adversary mitigate the effect of these real-world factors?
Website Fingerprinting Defenses at the Application Layer

Giovanni Cherubin\textsuperscript{1}  Jamie Hayes\textsuperscript{2}  Marc Juarez\textsuperscript{3}

\textsuperscript{1}Royal Holloway University of London
\textsuperscript{2}University College London
\textsuperscript{3}imec-COSIC KU Leuven

Presented in PETS 2017, Minneapolis, MN, USA
Tor Hidden Services (HSees)

User visits *xyz.onion* without resolving it to an IP
Website Fingerprinting on HSes

• The WF adversary can distinguish HSes from regular sites

• WF in HSes is more threatening:
  - **Fewer sites** makes HSes more identifiable (~ closed world)
  - HS users are more vulnerable because content is **sensitive**
The SecureDrop Case

- A whistleblowing platform as a HS
- By Freedom of the Press Foundation
- Vulnerable to website fingerprinting (?)
The New York Times is Now Available as a Tor Onion Service
Website Fingerprinting Defenses

WF Defenses:
- BuFLO
- Tamaraw
- CS-BuFLO
- WTF-PAD
- Walkie-Talkie

These are TCP packets

Entry

Middle

Tor network

Dummy

Real

44
Application-layer Defenses

- Existing defenses are designed at the network layer.

**Key observation:** identifying info originates at app layer!

- 'Latent' features: $F_1, \ldots, F_n$
- Observed features: $O_1, \ldots, O_n$
- Web content
  - HTTP(S)
  - Tor
  - TLS
  - TCP
  - ...

Identifying info

Last layer of encryption

Adversary
Pros and Cons of app-layer Defenses

The main advantage is that they are easier to implement:

- do not depend on Tor to be implemented

Disadvantages:

- padding runs end-to-end
- may require server collaboration… but HSeS have incentives!
LLaMA

• Client-side (FF add-on)
• Applied on website requests
• More latency overhead

ALPaCA

• Server-side (first one)
• Applied on hosted content
• More bandwidth overhead

(two different solutions, not a client-server solution)
ALPaCA

- Abstract web pages as **num objects** and **object sizes**: pad them to match a target page

- Does not impact user experience:
  
e.g., comments in HTML/JS, images’ metadata, “display: none” styles
Example: protect a SecureDrop page

- Strategy I: target page is Facebook
ALPaCA strategies (2)

- Strategy II: pad to an “anonymity set” target page

Defines num objects and object sizes by:

- Deterministic: next multiple of $\lambda, \delta$
- Probabilistic: sampled from empirical distribution
LLaMA

- Client-side (FF add-on)
- Applied on website requests
- More latency overhead

ALPaCA

- Server-side (first one)
- Applied on hosted content
- More bandwidth overhead

(two different solutions, **not** a client-server solution)
LLaMA

- Inspired by Randomized Pipelining
  Goal: randomize HTTP requests
- Same goal from a FF add-on:
  - Random delays ($\delta$)
  - Repeat previous requests ($C_1$)
Evaluation: methodology

- Collect **with** and **without** defense: 100 HSes (cached)
  - Security: *accuracy* of attacks
    - $kNN$, $k$-Fingerprinting ($kFP$), CUMUL
  - Performance: overheads
    - *latency* (extra delay)
    - *bandwidth* (extra padding/time)
LLaMA: results

- Accuracy drops between 20% and 30%
- Less than 10% latency and bandwidth overheads
ALPaCA: Results

- From 40% to 60% decrease in accuracy
- 50% latency and 85% bandwidth overheads
Take aways

- WF defenses at the app layer are **easier to implement**
- **HSes have incentives** to support server-side defenses:
  
  SecureDrop has implemented a prototype of ALPaCA

- ALPaCA is running on a HS: [3tmaadslguc72xc2.onion](http://3tmaadslguc72xc2.onion)

- Source code: [github.com/camelids](http://github.com/camelids)
Discussion

1. Do you think these defenses have any cost in anonymity? Why?

2. Would our results still apply if all hidden services install ALPaCA?

3. Are all the sites as easy to protect?

4. Does the protection vary over time?
How Unique is Your Onion? An Analysis of the Fingerprintability of Tor Onion Services

Rebekah Overdorf\textsuperscript{1}  Marc Juarez\textsuperscript{2}  Gunes Acar\textsuperscript{2}
Rachel Greenstadt\textsuperscript{1}  Claudia Diaz\textsuperscript{2}

\textsuperscript{1}Drexel University
\textsuperscript{2}imec-COSIC KU Leuven

Presented in CCS 2017, Dallas, TX, USA
Disparate impact

- WF studies report average success
- But...
  - Are certain websites more susceptible to website fingerprinting attacks than others?
  - What makes some sites more vulnerable to the attack than others?
Data Collection and Preprocessing

- 70 visits for each to 482 Tor Onion Services
  - total of 33740 visits
- pcap files
- packet details
- page source
- HTTP requests and responses
- screenshots
## Dataset and Results

<table>
<thead>
<tr>
<th>Classifier accuracy</th>
<th>k-NN</th>
<th>k-FP</th>
<th>CUMUL</th>
</tr>
</thead>
<tbody>
<tr>
<td>69.97%</td>
<td>77.71%</td>
<td>80.73%</td>
<td></td>
</tr>
</tbody>
</table>

- 70 visits to 482 Tor Onion Services
Fingerprintability
Fingerprintability
Fingerprintability

\[ \text{recall} = \frac{TP}{TP + FN} = 0.8 \]

How well am I identified as me?
Fingerprintability
Fingerprintability

$\text{precision} = \frac{TP}{TP + FP} = 0.5$

How likely is it that a positive classification is me?
Fingerprintability

\[ F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = 0.65 \]
Misclassifications

Visits that are “safe” from an attack
Misclassifications

Visits that are only misclassified by kNN

Visits that are misclassified by all 3
Misclassifications

Some sites are hidden from all attacks!
Misclassifications: Coinciding Predictions

Percentage of sites that are misclassified as the same site.
Misclassifications: Coinciding Predictions

Edges that are only in kNN confusion graph

Edges that are in all 3 graphs
Misclassifications: Coinciding Predictions

The attacks fail in different ways
DDG was mistaken for Secure Drop
Feature analysis

Two important predictors:

- Size (most features are a proxy!)

- Static-ness
  
  • Features that do not change between visits
What makes classifiers fail?

Web Page Size vs Fingerprintability
We don’t make traces, we make websites

Trace Features:
- Number of Packets
- Average Packet Size
- % of Packets that are incoming
- etc.

Site-Level Features:
- Resources
- Fonts
- Scripts
- etc.

- Can we determine what characteristics of a website affect its fingerprintability?
Site-level features

Number of HTTP requests
Number of HTTP responses
Has advertisement
Has tracking/analytics
HTML source size
Page load time
Made with Django
Made with Dokuwiki
Made with Drupal
Made with Joomla
Made with MediaWiki
Made with OnionMail
Made with phpSQLiteCMS
Made with vBulletin
Made with WooCommerce
Made with Wordpress

Made with CMS
Number of audio
Number of domains
Number of redirections
Number of empty content
Number of fonts
Number of HTML resources
Number of images
Number of other content
Number of scripts
Number of stylesheets
Number of videos
Number of waterfall phases
Screenshot size
Page weight
Total request size
Meta-Learner
(throw more machine learning at it)
Meta-Learner
(throw more machine learning at it)
Results

[Bar chart showing information gain for different metrics such as med screenshot size, med num HTML, med total HTTP download, etc., for all sites, smallest 10% of sites, and largest 10% of sites.]
Results

![Bar chart showing information gain for different metrics. The chart compares 'All Sites', 'Smallest 10% of Sites', and 'Largest 10% of Sites' for metrics such as median screenshot size, median number of HTML files, median number of fonts, median number of videos, median number of waterfall phases, median HTML src size, standard deviation screenshot size, median number of HTTP requests, and median number of HTTP responses.](chart.png)
Results
The SecureDrop site (Project On Gov’t Oversight’) in our dataset had an F1-Score of **99%** with CUMUL

– As compared to the 80% average accuracy

Multiple (duplicates) exist, but that doesn’t offer much protection
Take aways

• WF attacks are successful on Tor Hidden Services

• Certain websites more susceptible to WF attacks than others

• To avoid WF: make your HS **small** and **dynamic**

• Source code and data for fingerprintability analysis:
  
  cosic.esat.kuleuven.be/fingerprintability
Conclusion

• We have seen several research biases: overestimated adversary, base rate fallacy, unrealistic implementation, reporting averages only

• They may have an impact in the perception of the problem (alarming the community) and may steer research in wrong directions

• How to avoid them: keep human factor and practical considerations in mind when doing threat modeling

Credit for this presentation to:
Gunes Acar, Claudia Diaz, Tariq Elahi and Rebekah Overdorf
The HS world

- Exploratory crawl: 5,000 HSes (from Ahmia.fi)
- Stats for the HS world (from intercepted HTTP headers)
  - Distribution of types, sizes and number of resources
    - Most HSes are small compared to an average website
- Few HSes have any JS or 3rd-party content
  - JS: less than 13% ⇒ Assumption: no JS
  - 3rd party content: less than 20% ⇒ Assumption: no 3rd parties
Limitations and Future Work

- ALPaCA can only make sites bigger, but not smaller

- What’s the optimal padding at the app layer? Lack of a thorough feature analysis.

- How do the distributions change over time? How do we update our defenses accordingly?
  - How does the strategy need be adapted as HSes adopt our defense(s)?