FlowPrint: Semi-Supervised Mobile-App Fingerprinting on Encrypted Network Traffic

Thijs van Ede, Riccardo Bortolameotti, Andrea Continella, Jingjing Ren, Daniel J. Dubois, Martina Lindorfer, David Choffnes, Maarten van Steen and Andreas Peter

Contact: t.s.vanede@utwente.nl
Monitoring network traffic

- Apps communicate with the internet
Monitoring network traffic

- Apps communicate with the internet
- Can we infer mobile app usage from network traffic?
Monitoring network traffic

- Apps communicate with the internet
- Can we infer mobile app usage from network traffic?
- Traffic is encrypted
Monitoring network traffic

- Apps communicate with the internet
- Can we infer mobile app usage from network traffic?
- Traffic is encrypted
- Apps consist of modules
Monitoring network traffic

- Apps communicate with the internet
- Can we infer mobile app usage from network traffic?
- Traffic is encrypted
- Apps consist of modules
- Modules are shared by apps, leading to *homogeneous* traffic
Monitoring network traffic

- Apps communicate with the internet
- Can we infer mobile app usage from network traffic?
- Traffic is encrypted
- Apps consist of modules
- Modules are shared by apps, leading to *homogeneous traffic*
- Generated traffic depends on *dynamic* user input
Monitoring network traffic

- Apps communicate with the internet
- Can we infer mobile app usage from network traffic?
- Traffic is encrypted
- Apps consist of modules
- Modules are shared by apps, leading to homogeneous traffic
- Generated traffic depends on dynamic user input
- Apps on the device evolve over time
Monitoring network traffic

- Apps communicate with the internet
- Can we infer mobile app usage from network traffic?
- Traffic is encrypted
- Apps consist of modules
- Modules are shared by apps, leading to homogeneous traffic
- Generated traffic depends on dynamic user input
- Apps on the device evolve over time
  - Removal
Monitoring network traffic

- Apps communicate with the internet
- Can we infer mobile app usage from network traffic?
- Traffic is encrypted
- Apps consist of modules
- Modules are shared by apps, leading to homogeneous traffic
- Generated traffic depends on dynamic user input
- Apps on the device evolve over time
  - Removal
  - Installation
Monitoring network traffic

- Apps communicate with the internet
- Can we infer mobile app usage from network traffic?
- Traffic is encrypted
- Apps consist of modules
- Modules are shared by apps, leading to heterogeneous traffic
- Generated traffic depends on dynamic user input
- Apps on the device evolve over time
  - Removal
  - Installation
  - Update
Monitoring network traffic

- Apps communicate with the internet
- Can we infer mobile app usage from network traffic?
- Traffic is encrypted
- Apps consist of modules
- Modules are shared by apps, leading to homogeneous traffic
- Generated traffic depends on dynamic user input
- Apps on the device evolve over time
  - Removal
  - Installation
  - Update

Can we infer mobile app usage from network traffic **without prior knowledge** of installed apps?
Intuition

Apps are composed of a unique set of modules that each communicate with a relatively invariable set of network destinations.
Intuition

Apps are composed of a unique set of modules that each communicate with a relatively invariable set of network destinations.

![Diagram showing modules of App X and App Y](image-url)
Intuition

Apps are composed of a unique set of modules that each communicate with a relatively invariable set of network destinations.
Intuition

Apps are composed of a unique set of modules that each communicate with a relatively invariable set of network destinations.
Apps are composed of a unique set of modules that each communicate with a relatively invariable set of network destinations.
Intuition

Apps are composed of a unique set of modules that each communicate with a relatively invariable set of network destinations.
Intuition

Apps are composed of a unique set of modules that each communicate with a relatively invariable set of network destinations.
Intuition

Apps are composed of a unique set of modules that each communicate with a relatively invariable set of network destinations. How do we **extract** these **patterns** without prior knowledge of the apps?
FlowPrint - Overview

A. Feature extraction

B. Clustering

C. Browser isolation

D. Cross-correlation

E. Fingerprinting

F. Matching / updating

.pcap files

Fingerprints

shared
FlowPrint - Feature extraction

For each flow in the network, we extract

- Originating device
- Destination (IP, port)-tuple
- TLS certificate
- Timestamps
FlowPrint - Clustering

In 5 minute batches, we cluster flows by network destination:

- Destination (IP, port)-tuple or
- TLS certificate
FlowPrint - Clustering

In 5 minute batches, we cluster flows by network destination:

- Destination (IP, port)-tuple or
- TLS certificate
FlowPrint - Clustering

In 5 minute batches, we cluster flows by network destination:
- Destination (IP, port)-tuple or
- TLS certificate

- Some of these clusters are shared
FlowPrint - Cross-correlation

- Network destinations that are active together likely belong to the same app
FlowPrint - Cross-correlation

- Network destinations that are active together likely belong to the same app
- Compute correlation based on activity

\[(c_i \star c_j) = \sum_{t=0}^{T} c_i[t] \cdot c_j[t]\]
FlowPrint - Cross-correlation

- Network destinations that are active together likely belong to the same app
- Compute correlation based on activity

\[(c_i * c_j) = \sum_{t=0}^{T} c_i[t] \cdot c_j[t]\]
FlowPrint - Cross-correlation

- Network destinations that are active together likely belong to the same app
- Compute correlation based on activity

\[
(c_i \ast c_j) = \sum_{t=0}^{T} c_i[t] \cdot c_j[t]
\]
FlowPrint - Fingerprinting

- Remove weak correlations in graph
FlowPrint - Fingerprinting

- Remove weak correlations in graph
- Find cliques of strongly correlated clusters
FlowPrint - Fingerprinting

- Remove weak correlations in graph
- Find cliques of strongly correlated clusters
- Extract fingerprints as the set of destinations
  - Destination (IP, port)-tuple
  - TLS certificate
FlowPrint - Fingerprint matching

- Fingerprints are a set of destinations
  - Destination (IP, port)-tuple
  - TLS certificate
FlowPrint - Fingerprint matching

- Fingerprints are a set of destinations
  - Destination (IP, port)-tuple
  - TLS certificate
- Compare using the Jaccard similarity

\[ J(F_a, F_b) = \frac{|F_a \cap F_b|}{|F_a \cup F_b|} \]
Evaluation

- How well does our approach work?
Evaluation

● How well does our approach work?
  ○ Recognizing known apps
Evaluation

- How well does our approach work?
  - Recognizing known apps
  - Detecting previously unseen apps
Evaluation

● How well does our approach work?
  ○ Recognizing known apps
  ○ Detecting previously unseen apps

● Datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Encrypted</th>
<th>Homogeneous</th>
<th>Dynamic</th>
<th>Evolving</th>
<th>Malicious</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cross Platform</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ReCon</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Andrubis</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td></td>
<td>✔</td>
</tr>
</tbody>
</table>
Evaluation - Recognizing known apps

- Stable performance if number of apps increase
Evaluation - Recognizing known apps

- Stable performance if number of apps increase
- Compared FlowPrint with supervised approach AppScanner
  - F1-score of $0.89$ vs $0.58$
  - Precision of $0.92$ vs $0.88$
  - Recall of $0.89$ vs $0.50$
Evaluation - Detecting previously unknown apps

- Good performance in detecting and isolating previously unseen apps
Evaluation - Detecting previously unknown apps

- Good performance in detecting and isolating previously unseen apps
- Low number of flows gives worse performance
  - Low code coverage
Evaluation - Detecting previously unknown apps

- Good performance in detecting and isolating previously unseen apps
- Low number of flows gives worse performance
  - Low code coverage
- No observable difference between benign and malicious apps
Conclusion

FlowPrint isolates apps within encrypted network traffic **without** requiring **prior knowledge**

- Performs **better** than **supervised detectors**
- Requires **no training**
- **Recognizes** known apps
- **Isolates and detects** previously **unseen apps**

https://github.com/Thijsvanede/FlowPrint
Questions?

FlowPrint isolates apps within encrypted network traffic without requiring prior knowledge

- Performs better than supervised detectors
- Requires no training
- Recognizes known apps
- Isolates and detects previously unseen apps

https://github.com/Thijsvanede/FlowPrint

Thijs van Ede
✉ t.s.vanede@utwente.nl
🐦 @EdeThijs
FlowPrint - Browser Isolation

- Browser shows fewer repeatable patterns
- Each website has its own fingerprint
- Isolate browser using Random Forest
  - Relative change in active clusters
  - Relative change in bytes uploaded
  - Relative change in bytes downloaded
  - Relative change in upload/download ratio
Different app versions

![Graph showing the fraction of matching fingerprints over version differences between training and testing. The graph includes data for ReCon, ReCon Extended, and the average. Significant differences are observed over time.]
Changing features
Fingerprints per app
Execution time

- **Fingerprinting**
- **Clustering**
- **Batch time**

- **Find closest fingerprint**
- **Check for unseen apps**