FOAP: Fine-Grained Open-World Android App Fingerprinting

Jianfeng Li¹, Hao Zhou¹, Shuohan Wu¹, Xiapu Luo¹, Ting Wang², Xian Zhan¹, Xiaobo Ma³

¹The Hong Kong Polytechnic University
²Pennsylvania State University
³Xi’an Jiaotong University
Mobile apps reshape modern lifestyles from information retrieval to instant messaging, and from shopping to entertainment.

Despite the adoption of encrypted communication, mobile apps are susceptible to **app fingerprinting**.

App fingerprinting (AF) is a traffic analysis technique:

- Inferring user activity using features associated with *packet timing*, *packet direction*, and *packet size*.
- No need to inspect packet plaintext payload.

**Network administrator**
- *Censorship*
- *Malware detection*
- *User profiling*

**Attacker**
- *Reconnaissance attack*
- *User privacy inference*
Introduction

Limitation of existing AF

- **Closed-world assumption**: cannot handle apps unseen during model training
- **Coarse-grained identification**: recognizing apps, identifying selected user actions that need manual labeling

Our goal

- **Goal 1**: open-world app fingerprinting
  - Challenge: risk of false positives
- **Goal 2**: fine-grained user action identification
  - Challenge: automatic user action labeling

Threat model

The adversary sniffs network traffic on a wireless access point

The adversary *cannot* exploit

i) packet plaintext payload
ii) destination feature of network flows
Outline

Introduction

Design of FOAP

Evaluation
• Training data construction: network traffic labeling
• Open-world app traffic recognition: traffic segmentation, traffic filtering, flow recognition
• Fine-grained app fingerprinting: method-level user action identification
App-level labeling

Constructing coarse-grained training data

Extract flow features

123-dimentional feature vector for each flow

- General characteristic (8 features)
- Interactive pattern (20 features)
- Packet rate characteristic (5 features)
- Temporal characteristic (39 features)
- Packet size characteristic (51 features)
Local similarity (LS): a metric to quantify how a network flow is similar to network flows from the app of interest, say A, in terms of the intrinsic features from itself.

Observation:
- Network flows generated by A have relatively higher LS with A compared with network flows generated by other apps.
- There is obvious difference of LS between time periods dominated by A and those dominated by other apps.

Traffic segmentation:
- To locate all possible time periods when A is active, we conduct traffic segmentation in an unsupervised manner.
- Formulate traffic segmentation as a combinatorial optimization problem.
- Propose a divisive-agglomerative tree algorithm to solve it in a greedy fashion.
Open-World App Traffic Recognition

Traffic filtering

• If the app of interest is active during a traffic segment, we refer to it as a *relevant traffic segment* and otherwise an *irrelevant traffic segment*.

• Our goal is filtering out all irrelevant traffic segments to effectively reduce risk of false positives.

**Structural similarity (SS):** a metric to characterizes how network flows within a traffic segment are similar to network flows from the app of interest in various patterns.

• Mining flow patterns based on metric learning and clustering analysis.

• Filtering out irrelevant traffic segments by computing each traffic segment’s SS with the app of interest.

Flow recognition

• Constructing a bilevel recognition model to recognize all network flows generated by the app of interest from relevant traffic segments.
Method-level labeling

Constructing fine-grained training data

*Entrypoint (EP) method*—the component lifecycle or the UI callback
Method-Level User Action Identification

Flow Burstification

- **Persistent connection**: containing multiple packet bursts associated with sequentially invoked EP methods
- **Short connection**: containing only one packet burst associated with a single EP method

Spatial-Temporal Context Model

- **Spatial Context**: It reflects the contextual relationship between bursts within the same network flow
- **Temporal Context**: An EP method may trigger multiple network flows simultaneously, yielding concurrent in-flow bursts

Iterative Inference of EP Method

- Formulate contextual dependency of in-flow bursts based on conditional random field
- Identify EP Method based on a greedy inference algorithm
• **Open-World App Traffic Recognition**

**Dataset:** 1000 popular apps from Google Play Store

---

**Evaluating FOAP in open-world app traffic recognition (mean ± standard deviation)**

<table>
<thead>
<tr>
<th>$n_T$</th>
<th>AppScanner</th>
<th>AppScanner (extended)</th>
<th>FOAP (our approach)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
<td>Recall</td>
<td>F1-Score</td>
</tr>
<tr>
<td>5</td>
<td>0.315 ± 0.171</td>
<td>0.866 ± 0.105</td>
<td>0.439 ± 0.174</td>
</tr>
<tr>
<td>10</td>
<td>0.476 ± 0.183</td>
<td>0.814 ± 0.127</td>
<td>0.582 ± 0.161</td>
</tr>
<tr>
<td>15</td>
<td>0.568 ± 0.189</td>
<td>0.782 ± 0.137</td>
<td>0.641 ± 0.158</td>
</tr>
<tr>
<td>20</td>
<td>0.641 ± 0.180</td>
<td>0.755 ± 0.146</td>
<td>0.679 ± 0.151</td>
</tr>
</tbody>
</table>

---

**Effect of different modules of FOAP**

**False positive rate for open-world network traffic recognition (lower is better)**

<table>
<thead>
<tr>
<th>Open-World Traffic</th>
<th>AppScanner</th>
<th>AppScanner (extended)</th>
<th>FOAP (our method)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile Web</td>
<td>$2.41 \times 10^{-2}$</td>
<td>$2.18 \times 10^{-2}$</td>
<td><strong>$9.93 \times 10^{-4}$</strong></td>
</tr>
<tr>
<td>IoT</td>
<td>$3.39 \times 10^{-2}$</td>
<td>$2.85 \times 10^{-2}$</td>
<td><strong>$4.36 \times 10^{-5}$</strong></td>
</tr>
<tr>
<td>PC</td>
<td>$1.21 \times 10^{-2}$</td>
<td>$2.02 \times 10^{-2}$</td>
<td><strong>$3.54 \times 10^{-4}$</strong></td>
</tr>
<tr>
<td>SC-App</td>
<td>$3.24 \times 10^{-2}$</td>
<td>$3.09 \times 10^{-2}$</td>
<td><strong>$1.45 \times 10^{-2}$</strong></td>
</tr>
<tr>
<td>DC-App</td>
<td>$2.12 \times 10^{-2}$</td>
<td>$1.99 \times 10^{-2}$</td>
<td><strong>$2.64 \times 10^{-3}$</strong></td>
</tr>
</tbody>
</table>
**Evaluation**

- **EP Method Identification**

  Evaluating FOAP in EP method identification (mean ± standard deviation)

<table>
<thead>
<tr>
<th>ε</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.888 ± 0.102</td>
<td>0.883 ± 0.105</td>
<td>0.885 ± 0.101</td>
</tr>
<tr>
<td>1</td>
<td>0.886 ± 0.104</td>
<td>0.868 ± 0.114</td>
<td>0.876 ± 0.106</td>
</tr>
<tr>
<td>2</td>
<td>0.885 ± 0.111</td>
<td>0.853 ± 0.126</td>
<td>0.866 ± 0.114</td>
</tr>
<tr>
<td>5</td>
<td><strong>0.892 ± 0.105</strong></td>
<td>0.828 ± 0.137</td>
<td>0.856 ± 0.116</td>
</tr>
</tbody>
</table>

- **Cross-Dataset Evaluation**

  Evaluating FOAP in cross-dataset experimental settings (mean ± standard deviation)

<table>
<thead>
<tr>
<th>Transfer Setting</th>
<th>AppScanner</th>
<th>AppScanner (extended)</th>
<th>FOAP (our approach)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
<td>Recall</td>
<td>Precision</td>
</tr>
<tr>
<td>M → M</td>
<td>0.625 ± 0.167</td>
<td>0.775 ± 0.120</td>
<td>0.681 ± 0.135</td>
</tr>
<tr>
<td>M → H</td>
<td>0.506 ± 0.222</td>
<td>0.521 ± 0.195</td>
<td>0.481 ± 0.185</td>
</tr>
<tr>
<td>EM → H</td>
<td>0.514 ± 0.213</td>
<td>0.649 ± 0.162</td>
<td>0.547 ± 0.180</td>
</tr>
</tbody>
</table>

  Evaluating EP method identification in cross-dataset experimental settings (mean ± standard deviation)

<table>
<thead>
<tr>
<th>Transfer Setting</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>M → M</td>
<td>0.879 ± 0.098</td>
<td>0.872 ± 0.093</td>
<td>0.874 ± 0.093</td>
</tr>
<tr>
<td>M → H</td>
<td>0.771 ± 0.143</td>
<td>0.801 ± 0.146</td>
<td>0.783 ± 0.139</td>
</tr>
<tr>
<td>EM → H</td>
<td>0.830 ± 0.145</td>
<td>0.834 ± 0.150</td>
<td>0.831 ± 0.143</td>
</tr>
</tbody>
</table>
Evaluation

- Fine-Grained User Activity Inference

An example of inferring Twitter user activities

- User Privacy Analysis

Identifying positive reporting in a COVID-19 contact tracing app (TraceTogether)
Evaluation

- **Defense**: packet padding and dummy packet injection

Design of our AProxy countermeasure

Evaluating AProxy against FOAP

(a) Open-world app traffic recognition.

(b) EP method identification.
Summary

• FOAP is the first approach to identifying method-level fine-grained user action of Android apps in the open-world setting

• FOAP effectively reduces the false positive risk in open-world app recognition through adaptive traffic filtering

• FOAP carries out method-level user action identification via synthesizing traffic and binary analysis

• FOAP can be applied in user activity inference and user privacy analysis
Thank You

Email: jfli.xjtu@gmail.com