MAdFraud: Investigating Ad Fraud in Android Applications

Jonathan Crussell, Ryan Stevens, Hao Chen

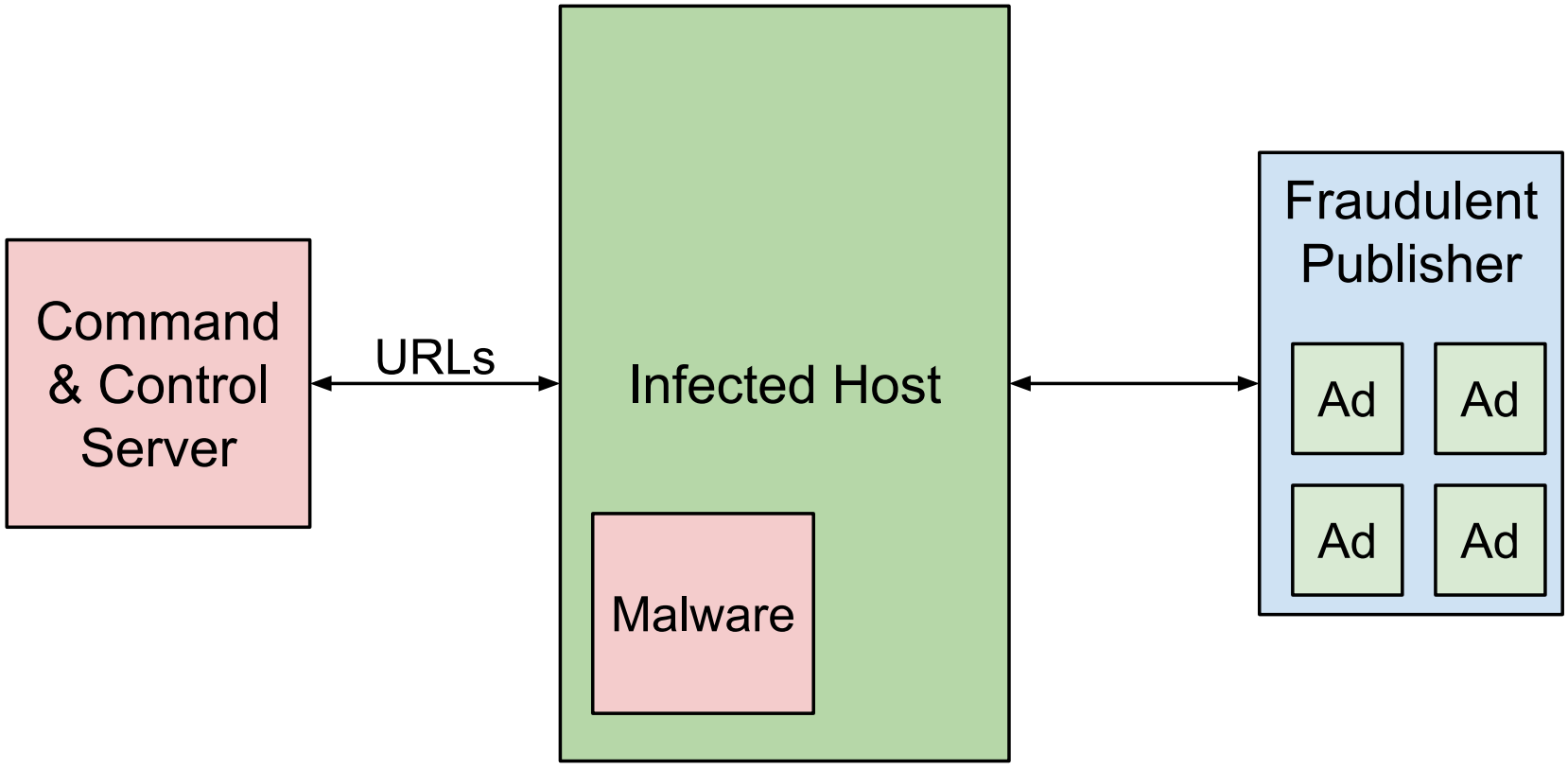
UC Davis Computer Security Lab

June 17th, 2014

Free but ad supported

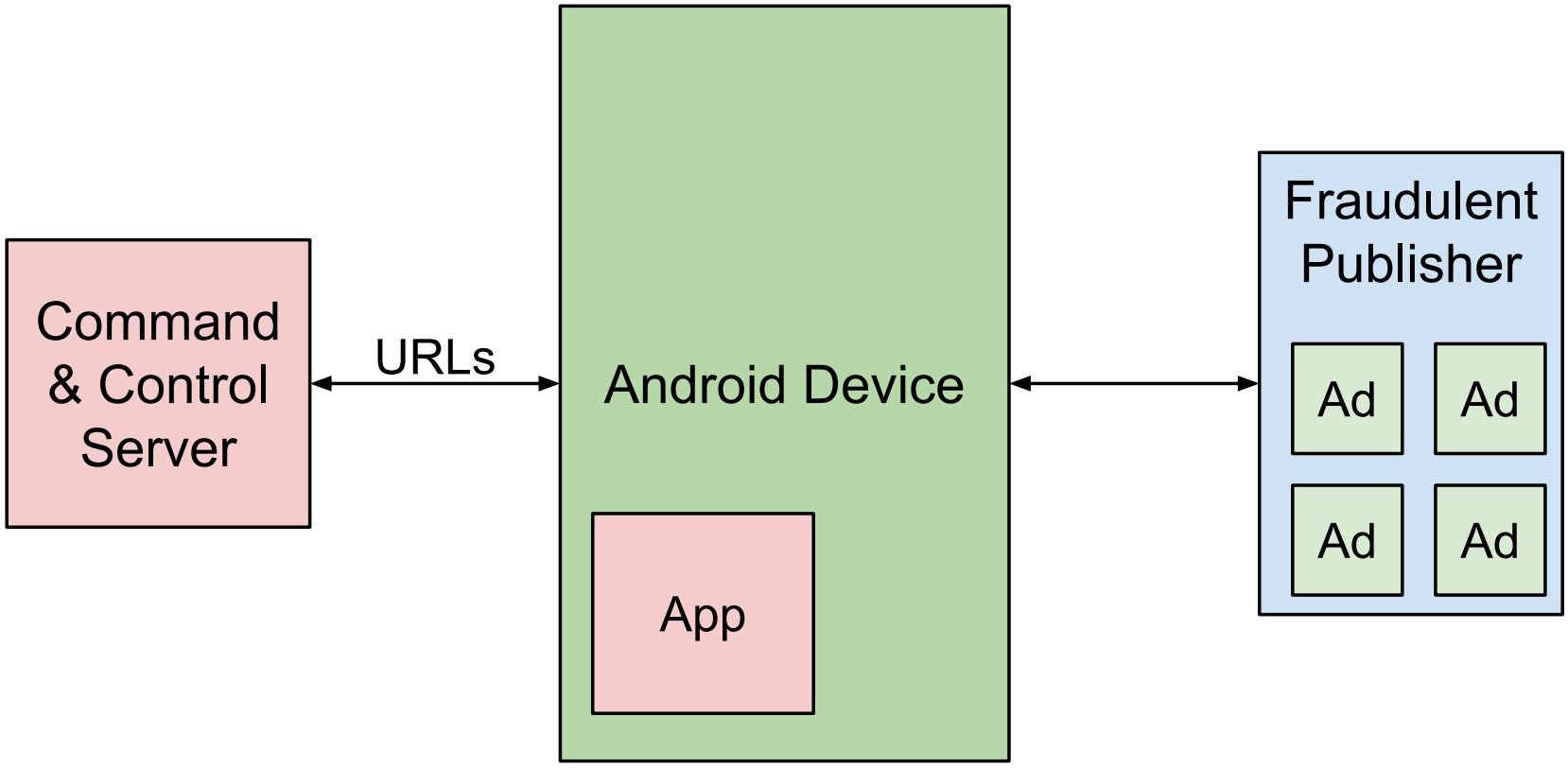
Web Ad Fraud

Programs that automatically "view" ads and "click" on them

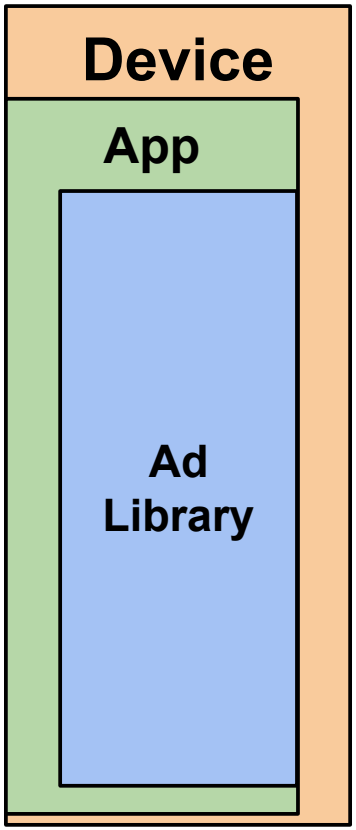


Mobile Ad Fraud

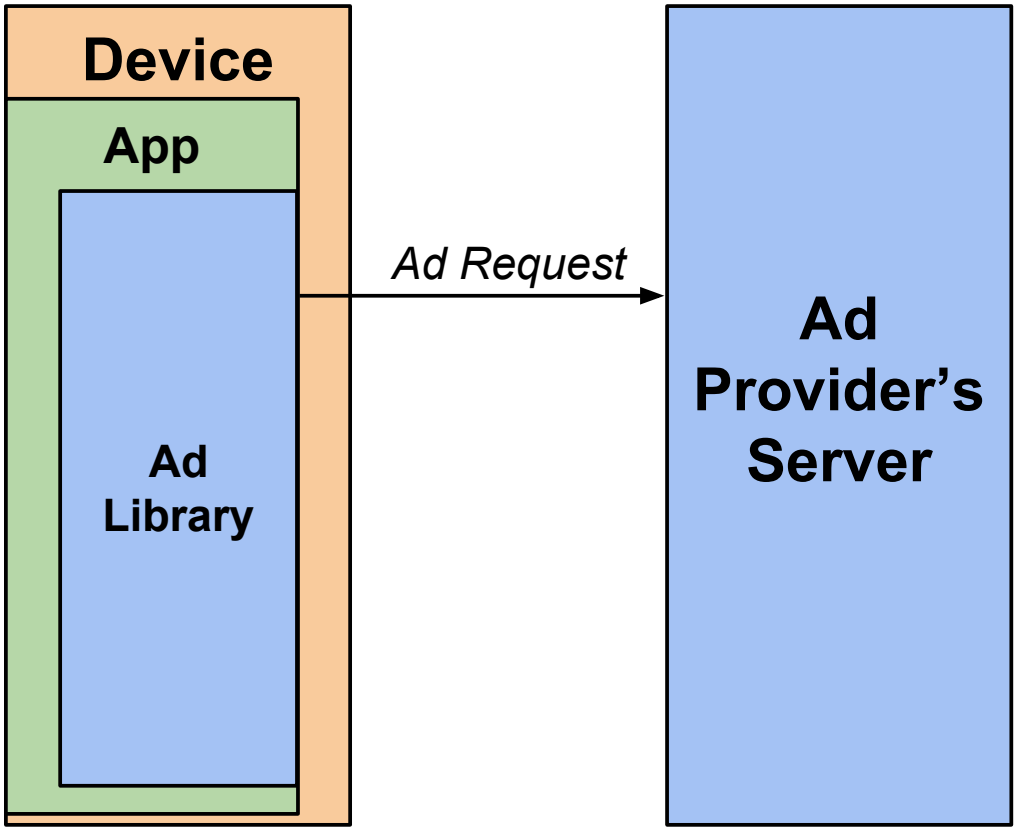
Apps that automatically "view" ads and "click" on them



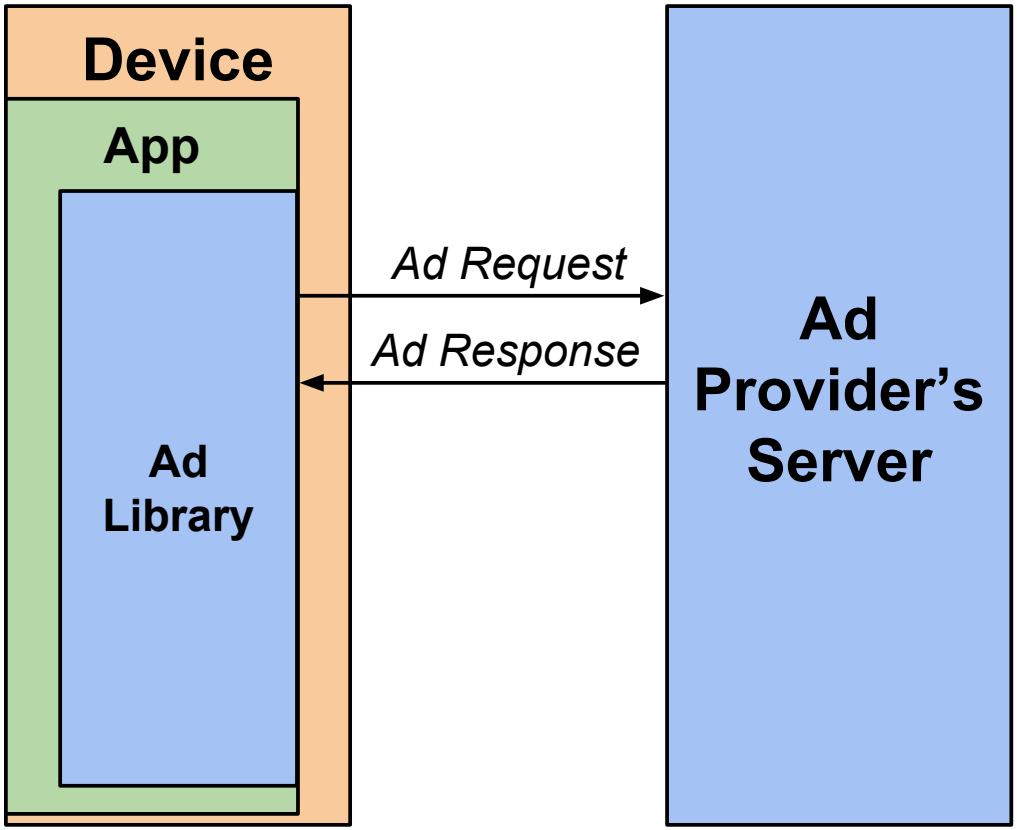
Anatomy of Mobile Ad Traffic



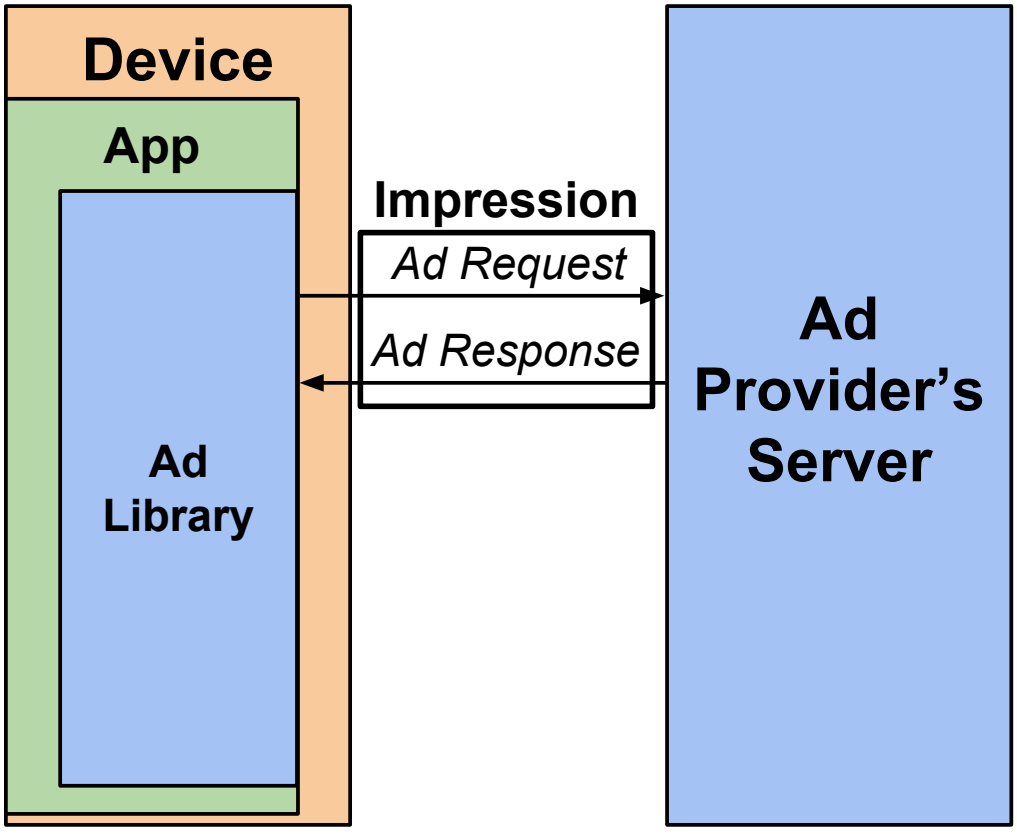
Anatomy of Mobile Ad Traffic



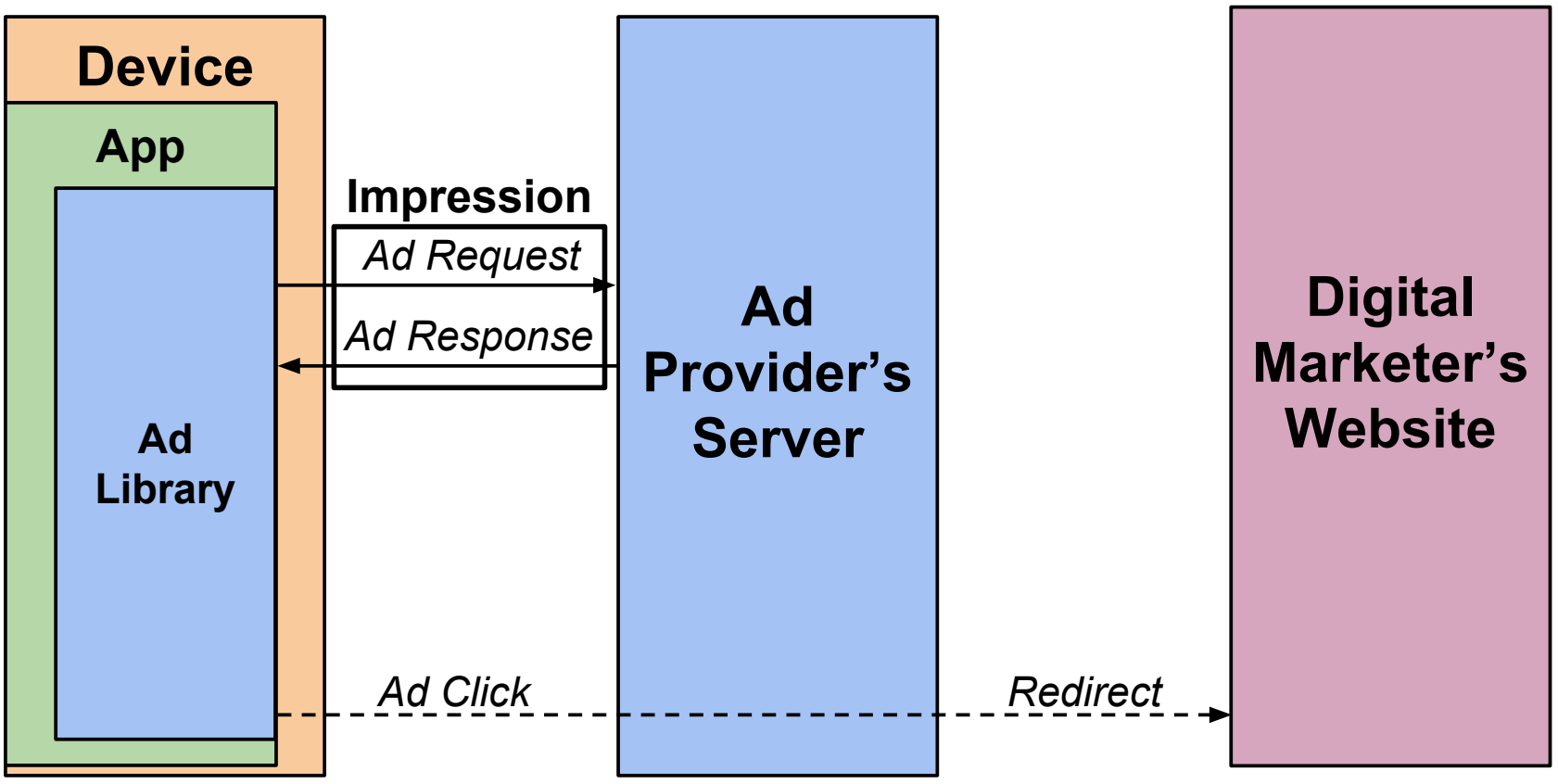
Anatomy of Mobile Ad Traffic



Anatomy of Mobile Ad Traffic



Anatomy of Mobile Ad Traffic



MAdFraud

Goals:

- Design system for automatically detecting ad traffic
- Use system to detect fraud and other undesirable ad behavior

Definition: Ad fraud

Apps that:

- Request ads when in the background (impression fraud)
- Click on ads without user interaction (click fraud)

Definition: Ad fraud

Apps that:

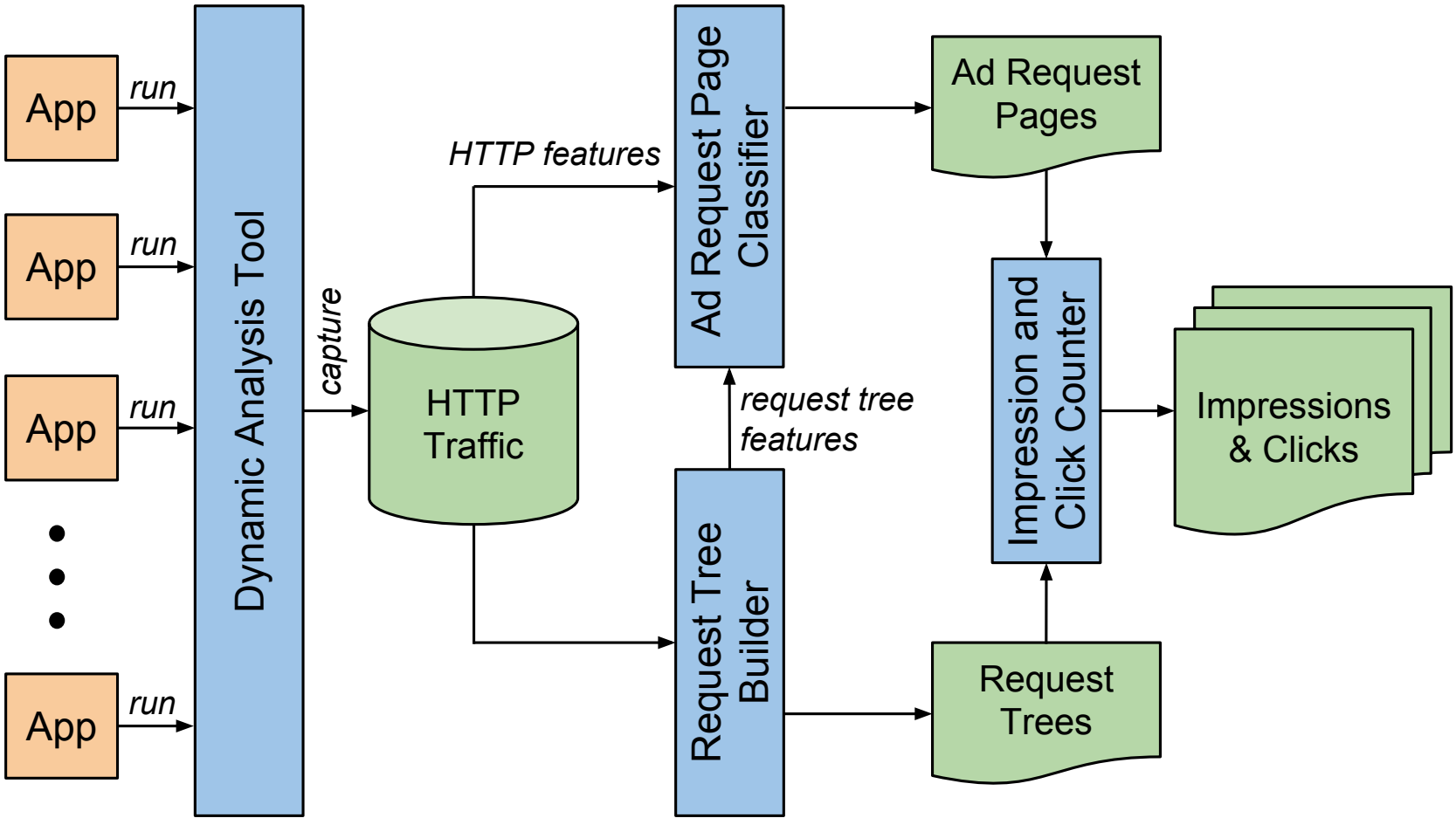
- Request ads when in the background (impression fraud)
- Click on ads without user interaction (click fraud)

Studied in related work:

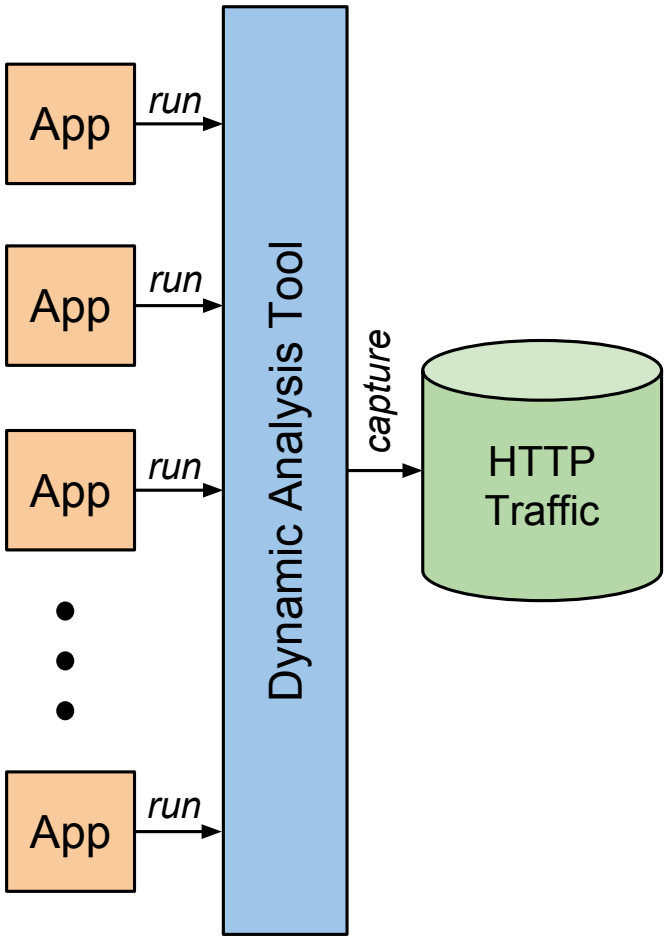
- Apps that obscure or obstruct ads (display fraud) [1]
- Clickjacking

[1] Liu et al., NSDI 2014

MAdFraud


App
App
App
App
run
run
run
run
Dynamic Analysis Tool
capture
HTTP Traffic
HTTP features
Ad Request Page Classifier
request tree features
Request Tree Builder
Ad Request Pages
Request Trees
Impression and Click Counter
Impressions & Clicks

MAdFraud


App
run
App
run
App
run
App
run
Dynamic Analysis Tool
capture
HTTP
Traffic

Dynamic Analysis

For each app:

- Create an emulator and install the app
- Run app in the foreground for 1 minute
- Use an Intent to move the app into the background
- Run app in the background for 1 minute
- Capture all network traffic throughout

Dynamic Analysis

For each app:

- Create an emulator and install the app
- Run app in the foreground for 1 minute
- Use an Intent to move the app into the background
- Run app in the background for 1 minute
- Capture all network traffic throughout

Never interact with app to ensure all detected clicks are fraudulent

Identifying Impressions and Clicks

Simple approach:

- Reverse engineer ad libraries to find ad request and click URLs
- Use URLs to count the number of impressions and clicks

Identifying Impressions and Clicks

Simple approach:

- Reverse engineer ad libraries to find ad request and click URLs
- Use URLs to count the number of impressions and clicks

Problems:

- Time intensive for the many ad libraries (80+)
- Lack context:
  - Was the ad resold?
  - Was the click proceeded by an impression?

Identifying Impressions and Clicks

Simple approach:

- Reverse engineer ad libraries to find ad request and click URLs
- Use URLs to count the number of impressions and clicks

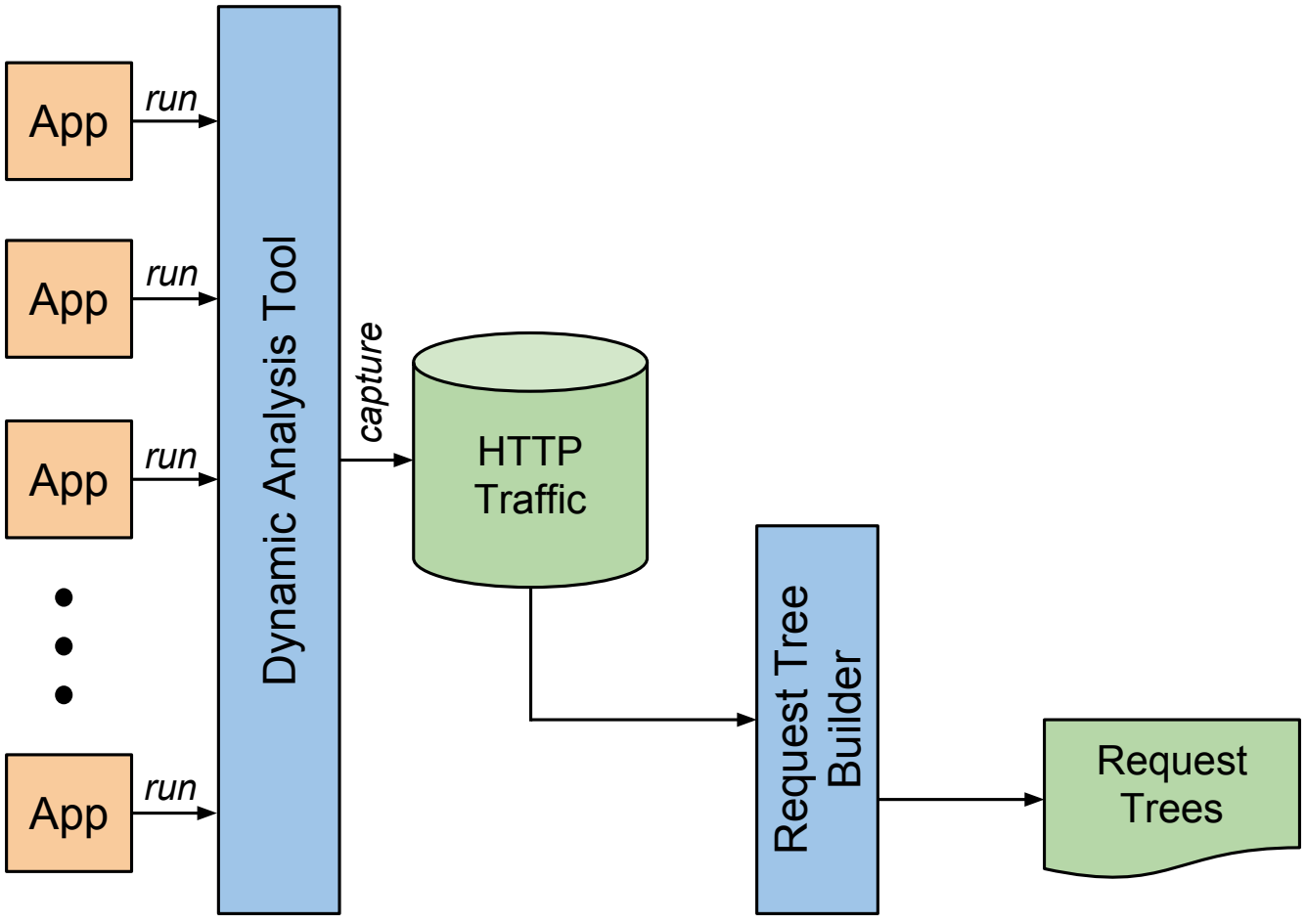
Problems:

- Time intensive for the many ad libraries (80+)
- Lack context:
  - Was the ad resold?
  - Was the click proceeded by an impression?

Need an automated approach with context:

- Machine learning to identify impressions
- Request trees to identify clicks

MAdFraud


App
App
App
App
run
run
run
run
Dynamic Analysis Tool
capture
HTTP Traffic
Request Tree Builder
Request Trees

Request Tree Builder

Build trees of HTTP requests using:

Request Tree Builder

Build trees of HTTP requests using:

- Referrer Request Header

```
GET /connect/ping?... HTTP/1.1
Host: www.facebook.com
Referer: http://www.cnn.com
```

Request Tree Builder

Build trees of HTTP requests using:

- Referrer Request Header

```
GET /connect/ping?... HTTP/1.1
Host: www.facebook.com
Referer: http://www.cnn.com
```

- Location Response Header

```
HTTP/1.1 302 Found
Location: http://static...com/...
```

Request Tree Builder

Build trees of HTTP requests using:

- Referrer Request Header

```
GET /connect/ping?... HTTP/1.1
Host: www.facebook.com
Referer: http://www.cnn.com
```

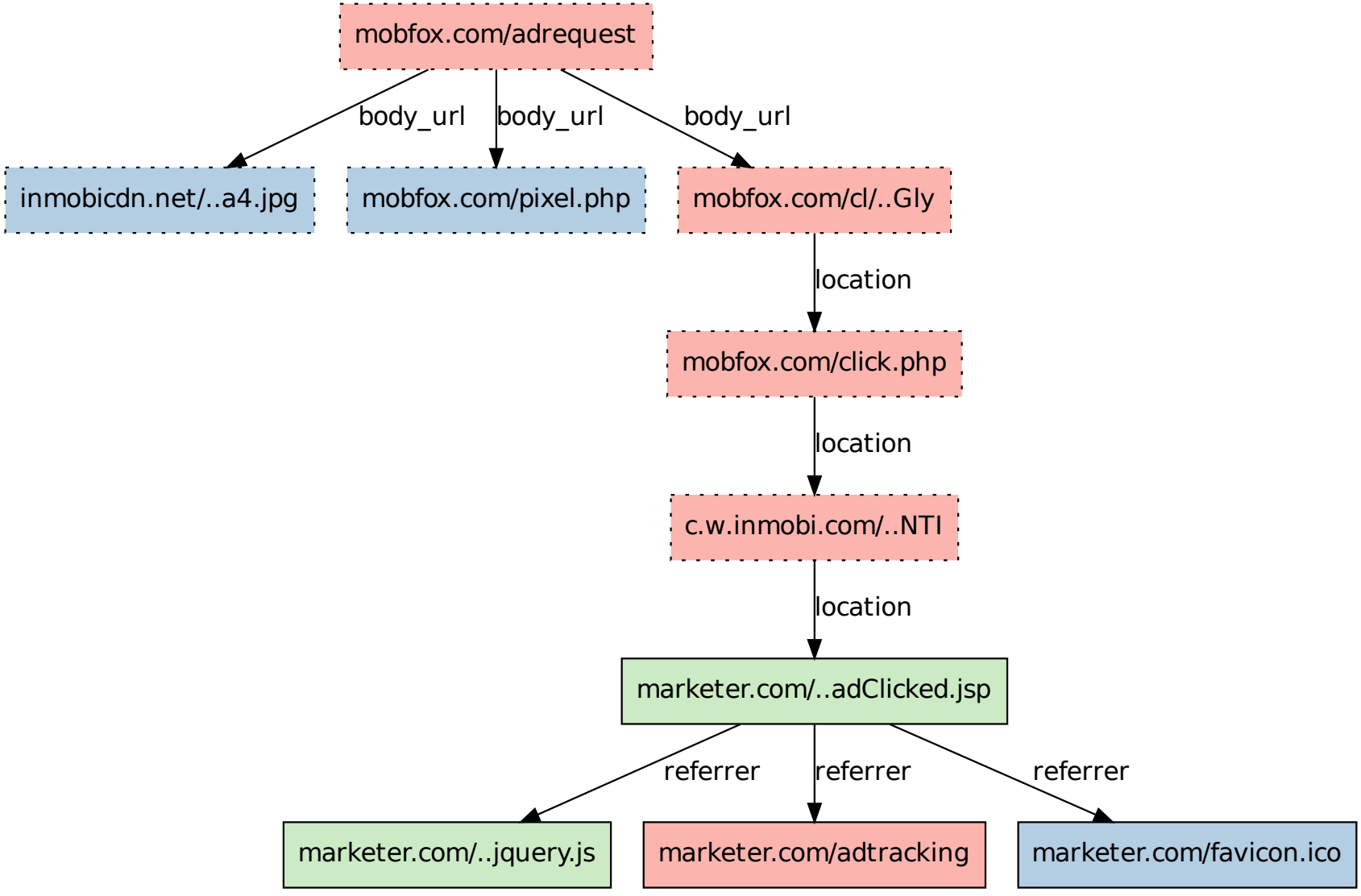
- Location Response Header

```
HTTP/1.1 302 Found
Location: http://static...com/...
```

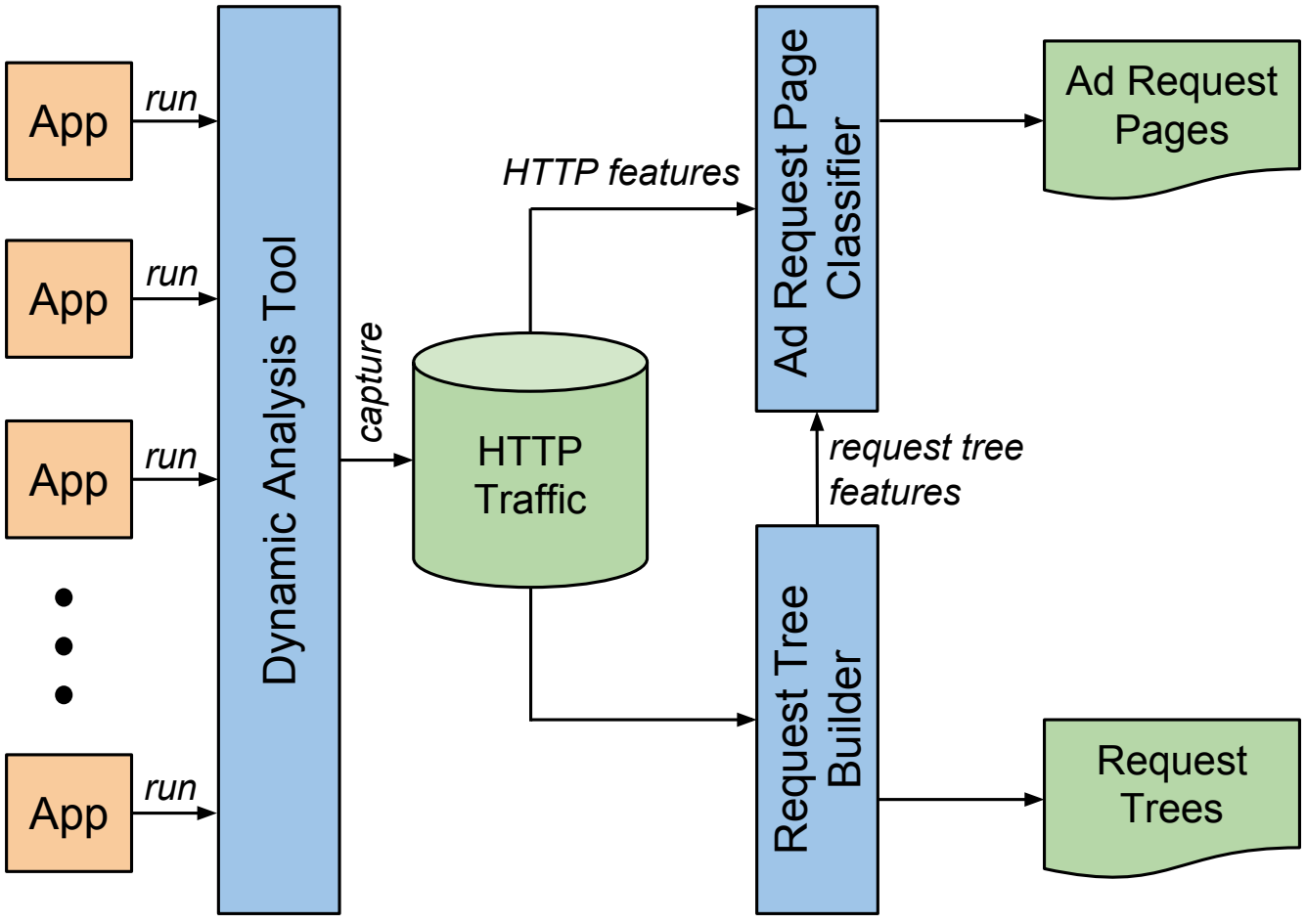
- URLs in Response Body

```
<Ad id="..." AdRequestId="..."
    BannerUrl="http://cdn...com/..."
    ClickUrl="http://tracking...com/..." />
```

Request Tree For an Ad Request



MAdFraud


App
App
App
App
run
run
run
run
Dynamic Analysis Tool
capture
HTTP Traffic
HTTP features
Ad Request Page Classifier
Ad Request Pages
request tree features
Request Tree Builder
Request Trees

Ad Requests

Ad requests have characteristic features:

- Many query parameters for ad targeting
- Several *GUIDs* in query parameters
- At the top of a request tree

Ad Requests

Ad requests have characteristic features:

- Many query parameters for ad targeting
- Several *GUIDs* in query parameters
- At the top of a request tree

Group individual requests into pages:

`http://domain.com/path/to/page.php?k1=v1&k2=v2`

Ad Request Page Classifier

Classify pages as for ad requests (ARQ) or not ($NARQ$)

Ad Request Page Classifier

Classify pages as for ad requests (ARQ) or not ($NARQ$)

Methodology:

- Aggregate all the requests to a given page
- Build features based on:
  - Query parameters (16)
  - Request trees (14)
  - HTTP properties (8)

Ad Request Page Classifier

Classify pages as for ad requests (ARQ) or not ($NARQ$)

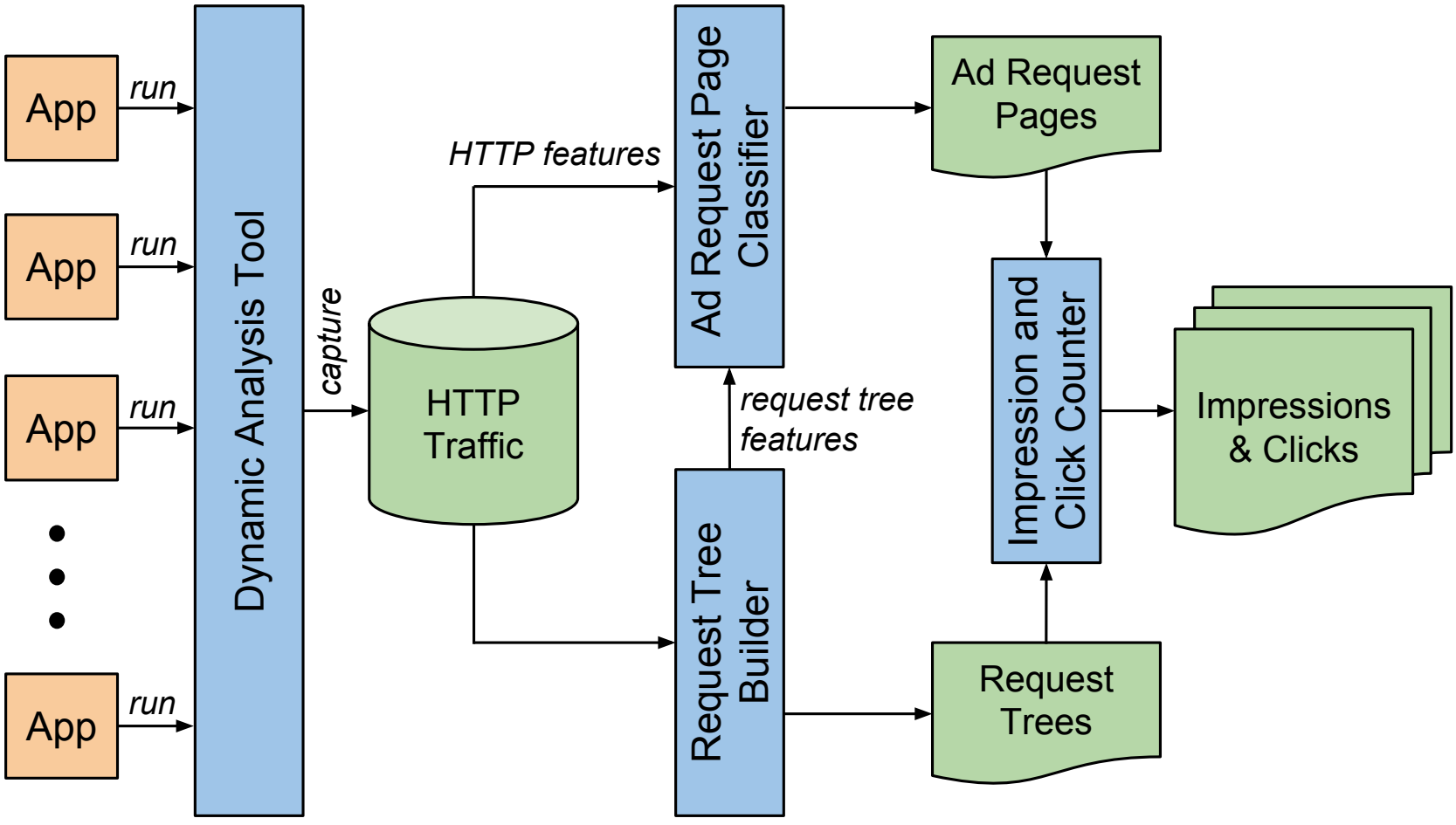
Methodology:

- Aggregate all the requests to a given page
- Build features based on:
  - Query parameters (16)
  - Request trees (14)
  - HTTP properties (8)

Evaluation on ground truth from popular ad providers:

- Class-weighted accuracy of 85.9%
- Query parameters are the most predictive

MAdFraud


App
App
App
App
run
run
run
run
Dynamic Analysis Tool
capture
HTTP Traffic
HTTP features
Ad Request Page Classifier
request tree features
Request Tree Builder
Ad Request Pages
Request Trees
Impression and Click Counter
Impressions & Clicks

Counting Impressions and Clicks

Combine:

- 229 *ARQ* pages from 77 ad providers identified by classifier
- Request trees built from all HTTP traffic

Counting Impressions and Clicks

Combine:

- 229 *ARQ* pages from 77 ad providers identified by classifier
- Request trees built from all HTTP traffic

Methodology:

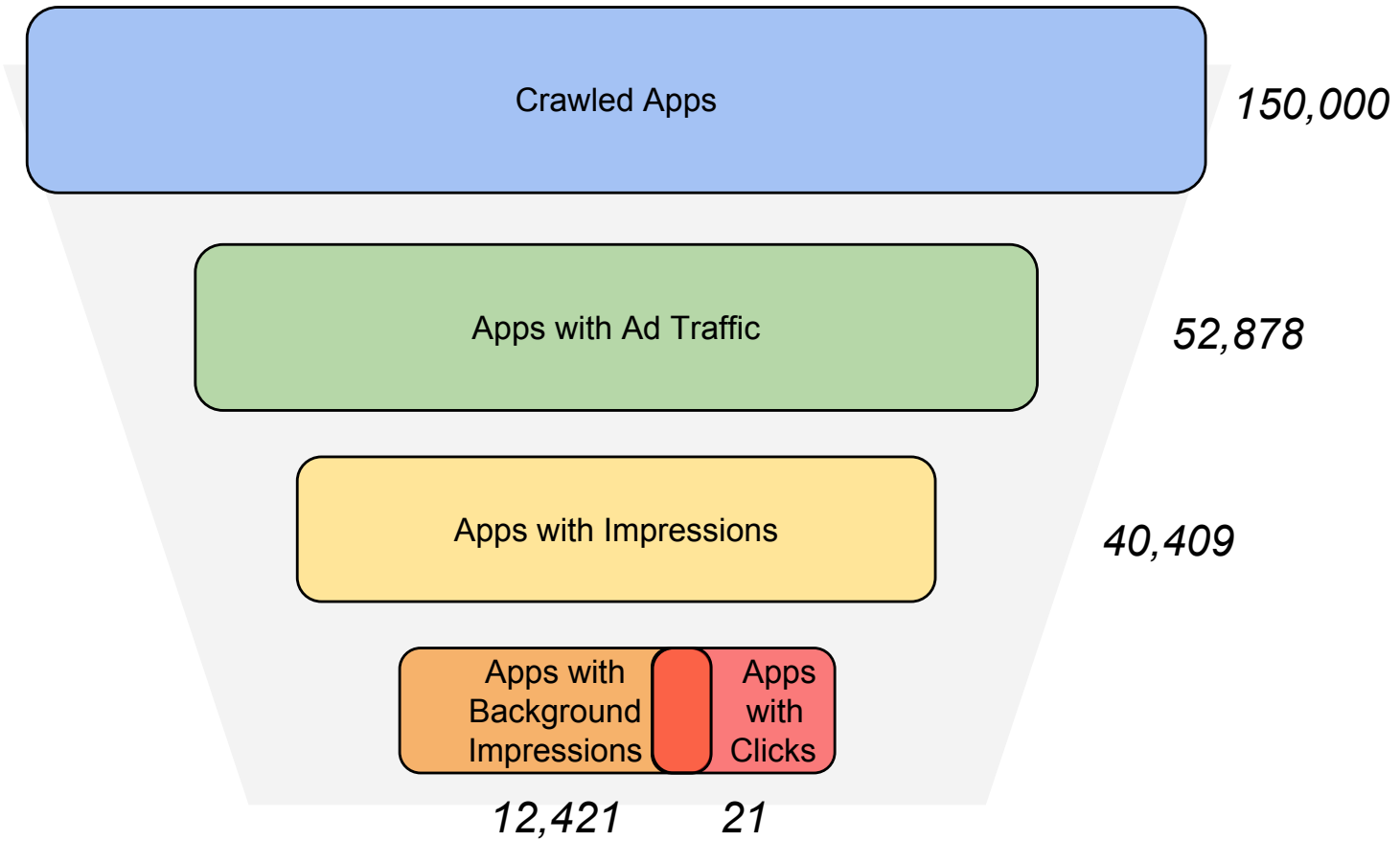
- Find all trees with an *ARQ* page at the root
- Traverse trees from root to find clicks:
  - Redirect to an HTML page
  - Redirect to non-HTTP schema (e.g. market://)
  - Restriction: landing page must be for non-ad provider domain

Dataset

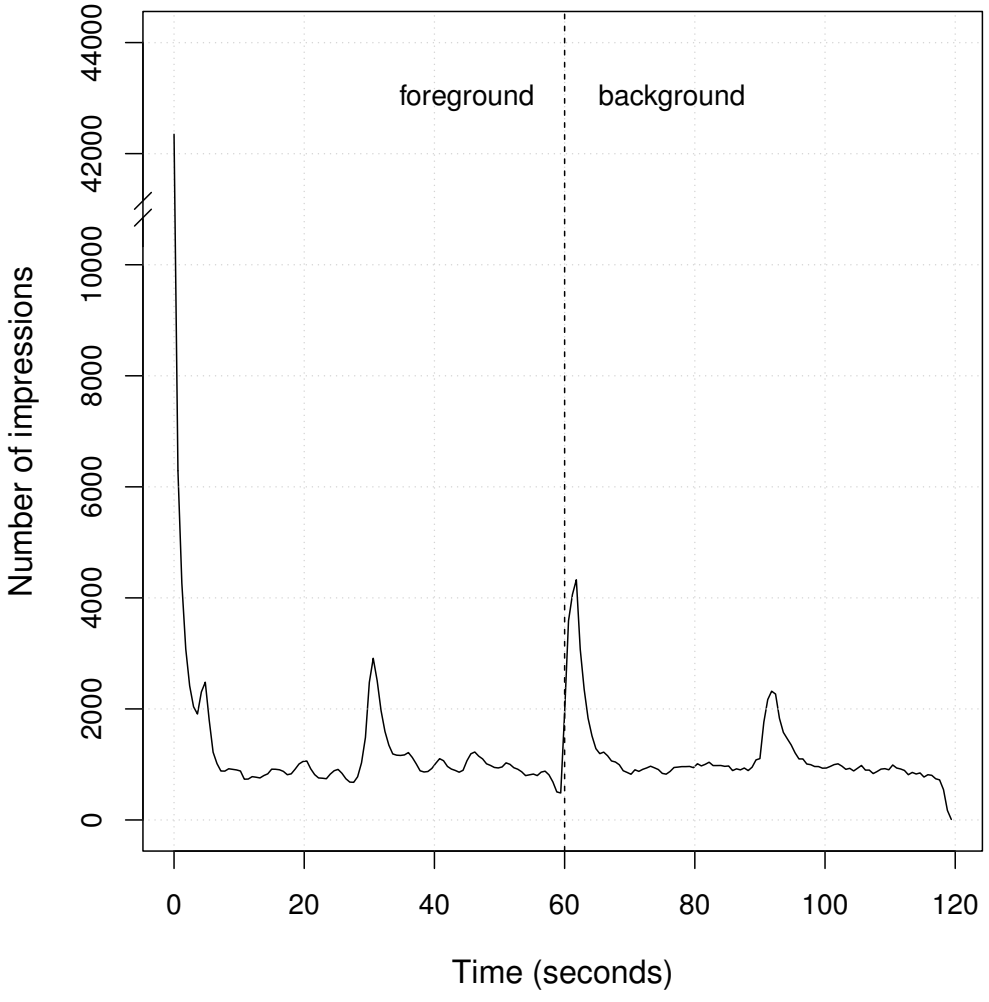
713,173 apps crawled from 19 markets

- Only evaluate on a subset of 150,000 apps

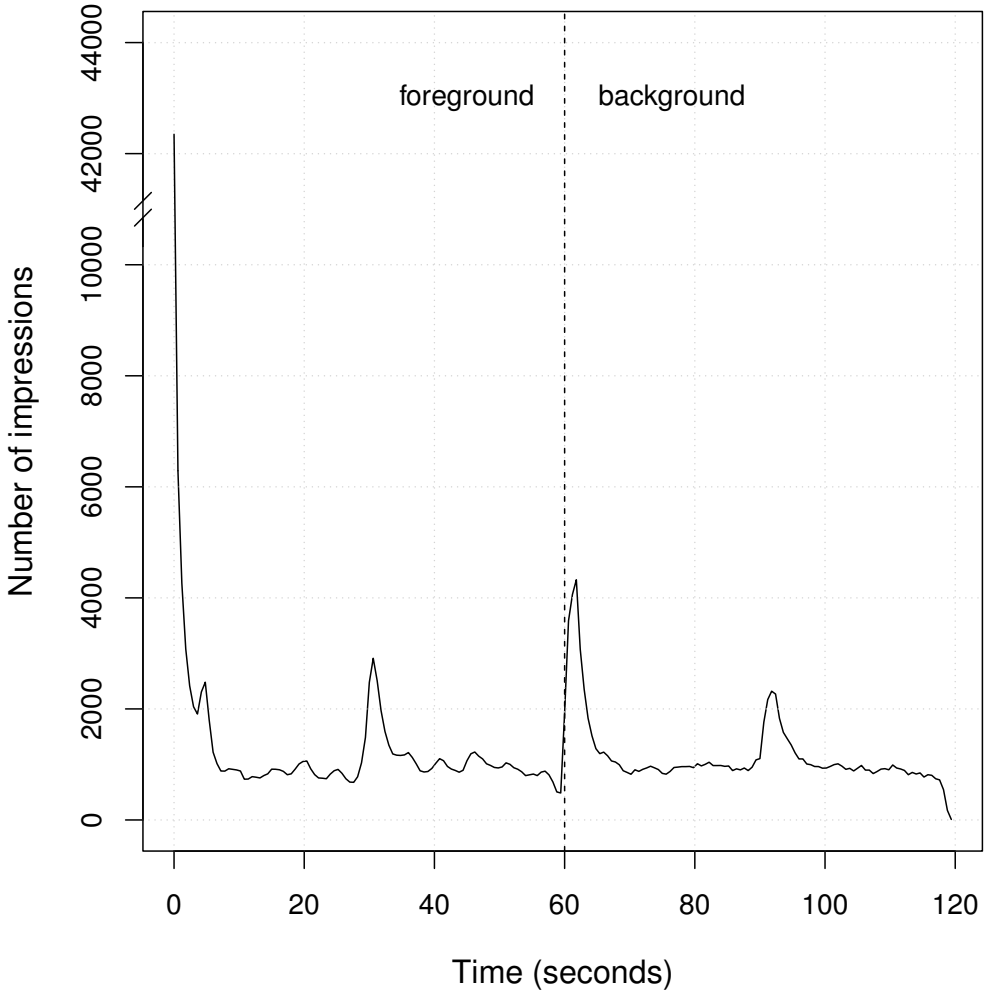
Results Overview



Finding: Background Impressions



Finding: Background Impressions



12,421 apps make impressions in the background

Finding: Click Fraud

App	# Impressions	# Clicks	Click Interval (s)	Ad providers
79b85a	63	18	1,935.6	MobFox
9e5b41	63	18	1,925.7	MobFox
f56bda	63	17	2,049.3	MobFox
5a6fc0	54	6	5,795.6	MobFox
63fd85	54	6	5,798.7	MobFox
76a7dc	54	6	5,797.3	MobFox
86cd17	54	5	6,956.4	MobFox
94bfa8	4,366	5	8,353.1	Migital
7bb12f	4,392	4	6,717.8	Migital
a3d816	4,381	3	10,000	Migital
807a0a	98	2	0	AppsGeyser
d9162a	4,385	2	16,919.2	Migital
57b67c	56	1	N/A	AppsGeyser
b611ea	4,374	1	N/A	Migital
c7681c	4,416	1	N/A	Migital
d55ece	4,384	1	N/A	Migital
Total	31,257	96		

Conclusion

Contributions:

- Developed a system, MAdFraud, to analyze mobile ad traffic
- Novel approach to identify impressions and clicks
- Discovered and analyzed fraudulent ad behavior

Dataset:

- `http://cancer.cs.ucdavis.edu/~jcrussell/madfraud-dataset/`

Questions/Comments?

Presenter: Jonathan Crussell
jcrussell@ucdavis.edu

Finding: Click Fraud

App	Source	# Installs	# Clicks
79b85a	Opera	236	18
9e5b41	SlideME	915	18
f56bda	Google Play	1,000	17
5a6fc0	Opera	117	6
63fd85	Opera	255	6
76a7dc	Opera	125	6
86cd17	SlideME	915	5
94bfa8	Google Play*	500	5
7bb12f	1Mobile	N/A	4
a3d816	?	N/A	3
807a0a	BrotherSoft	N/A	2
d9162a	BrotherSoft	N/A	2
57b67c	Google Play*	10,000	1
b611ea	?	N/A	1
c7681c	?	N/A	1
d55ece	?	N/A	1
Total		14,063	96

Limitations

- Do not capture HTTPS traffic
- Apps run on emulator instead of real device
- We do not interact with apps so may miss some fraud
- We cannot detect display fraud

Cross validation

		Prediction		
		ARQ	*NARQ*	Recall
Truth	*ARQ*	28	11	71.8%
	NARQ	9	11,475	99.9%
	Precision	75.7%	99.9%	

Confusion matrix of our ad request page classifier, computed using 3 fold cross-validation on a ground truth dataset of known ad request pages.