Tracing Information Flows Between Ad Exchanges Using Retargeted Ads

Muhammad Ahmad Bashir, Sajjad Arshad, William Robertson, Christo Wilson

Northeastern University
Your Privacy Footprint
Your Privacy Footprint
Your Privacy Footprint

PubMatic → CNN → Facebook

OpenX → Facebook

doubleclick → Facebook
Your Privacy Footprint

- Chrome
- CNN
- Amazon
- PubMatic
- OpenX
- Criteo
- AppNexus
- Facebook
- Rubicon
- DoubleClick
Your Privacy Footprint

- PubMatic
- CNN
- Amazon
- BBC
- Rubicon Project
- OpenX
- Criteo
- AppNexus
- DoubleClick
Your Privacy Footprint

Diagram showing connections between various entities including CNN, Amazon, BBC, PubMatic, OpenX, criteo, AppNexus, and rightmedia.
Real Time Bidding
Real Time Bidding

• RTB brings more flexibility in the ad ecosystem.
  • Ad request managed by an Ad Exchange which holds an auction.
  • Advertisers bid on each ad impression.
Real Time Bidding

- RTB brings more flexibility in the ad ecosystem.
  - Ad request managed by an Ad Exchange which holds an auction.
  - Advertisers bid on each ad impression.

Cookie matching is a prerequisite.

Exchange ← Advertiser
Real Time Bidding

• RTB brings more flexibility in the ad ecosystem.
  • Ad request managed by an Ad Exchange which holds an auction.
  • Advertisers bid on each ad impression.
  
• RTB spending to cross $20B by 2017\textsuperscript{[1]}.
  • 49% annual growth.
  • Will account for 80% of US Display Ad spending by 2022.

User → Publisher → Ad Exchange → Advertisers

GET, CNN’s Cookie
User → Publisher

GET, CNN’s Cookie

Publisher → User

GET, CNN’s Cookie

User → Ad Exchange

Solicit bids, DoubleClick’s Cookie

Ad Exchange → Advertisers

Bid

Advertisers → Ad Exchange

Solicit bids, DoubleClick’s Cookie
Real Time Bidding (RTB)

User

Publisher

GET, CNN’s Cookie

GET, DoubleClick’s Cookie

Ad Exchange

Solicit bids, DoubleClick’s Cookie

Bid

Advertisers
Real Time Bidding (RTB)

User

Publisher

Ad Exchange

Advertisers

GET, CNN’s Cookie

Solicit bids, DoubleClick’s Cookie

Bid

GET, CNN’s Cookie

GET, DoubleClick’s Cookie

GET, RightMedia’s Cookie

Advertisement
Real Time Bidding (RTB)

User

Publisher

Ad Exchange

Advertisers

GET, CNN’s Cookie

GET, CNN’s Cookie

GET, DoubleClick’s Cookie

Solicit bids, DoubleClick’s Cookie

Bid

GET, RightMedia’s Cookie

Advertisement

Advertisers cannot read their cookie!
Cookie Matching

Key problem: Advertisers cannot read their cookies in the RTB auction
  • How can they submit reasonable bids if they cannot identify the user?

Solution: **cookie matching**
  • Also known as cookie synching
  • Process of linking the identifiers used by two ad exchanges
Cookie Matching

Key problem: Advertisers cannot read their cookies in the RTB auction
  • How can they submit reasonable bids if they cannot identify the user?

Solution: cookie matching
  • Also known as cookie synching
  • Process of linking the identifiers used by two ad exchanges
Cookie Matching

Key problem: Advertisers cannot read their cookies in the RTB auction
  • How can they submit reasonable bids if they cannot identify the user?

Solution: cookie matching
  • Also known as cookie synching
  • Process of linking the identifiers used by two ad exchanges
Cookie Matching

Key problem: Advertisers cannot read their cookies in the RTB auction
  • How can they submit reasonable bids if they cannot identify the user?

Solution: cookie matching
  • Also known as cookie synching
  • Process of linking the identifiers used by two ad exchanges

GET, Cookie=12345

301 Redirect, Location=http://criteo.com/?dblclk_id=12345

GET ?dblclk_id=12345, Cookie=ABCDE
Cookie Matching

Key problem: Advertisers cannot read their cookies in the RTB auction
  • How can they submit reasonable bids if they cannot identify the user?

Solution: cookie matching
  • Also known as cookie synching
  • Process of linking the identifiers used by two ad exchanges
Cookie Matching

Key problem: Advertisers cannot read their cookies in the RTB auction
  • How can they submit reasonable bids if they cannot identify the user?

Solution: cookie matching
  • Also known as cookie synching
  • Process of linking the identifiers used by two ad exchanges

```
GET, Cookie=12345

301 Redirect, Location=http://criteo.com/?dblclk_id=12345

GET ?dblclk_id=12345, Cookie=ABCDE
```
Prior Work

• Several studies have examined cookie matching
  • Acar et al. found hundreds of domains passing identifiers to each other
  • Olejnik et al. found 125 exchanges matching cookies
  • Falahrastegar et al. analyzed clusters of exchanges that share the exact same cookies
Prior Work

• Several studies have examined cookie matching
  • Acar et al. found hundreds of domains passing identifiers to each other
  • Olejnik et al. found 125 exchanges matching cookies
  • Falahrastegar et al. analyzed clusters of exchanges that share the exact same cookies

• These studies rely on studying HTTP requests/responses.
Challenge 1: Server Side Matching
Challenge 1: Server Side Matching

1) Criteo observes the user.
   (IP: 207.91.160.7)
Challenge 1: Server Side Matching

1) Criteo observes the user. (IP: 207.91.160.7)

2) RightMedia observes the user. (IP: 207.91.160.7)
Challenge 1: Server Side Matching

1) Criteo observes the user. (IP: 207.91.160.7)

2) RightMedia observes the user. (IP: 207.91.160.7)

Behind the scene, RightMedia and Criteo sync up.
Challenge 2: Obfuscation
Challenge 2: Obfuscation
Challenge 2: Obfuscation

GET, Cookie=ABCDE
Challenge 2: Obfuscation

amazon.com

GET, Cookie=ABCDEF

GET, Cookie=12345
Challenge 2: Obfuscation

GET %^$ck#&93#&, Cookie=XYZYX
GET %^$ck#&93#&, Cookie=XYZYX
GET, Cookie=12345
GET, Cookie=ABCDE
Challenge 2: Obfuscation

amazon.com

dbclk.js

GET, Cookie=ABCDE

GET, Cookie=12345

GET, Cookie=XYZYX
Challenge 2: Obfuscation

amazon.com

dbclk.js

GET, Cookie=ABCDE

GET, Cookie=12345

GET %^$ck#&93#& Cookie=XYZYX
Challenge 2: Obfuscation

GET %^$ck#&93#$& Cookie=XYZYX

GET, Cookie=12345

GET, Cookie=ABCDE

amazon.com
dbclk.js
criteo

doubleclick by Google
Challenge 2: Obfuscation

GET %^$ck#$&93#$& Cookie=XYZYX
GET, Cookie=12345
GET, Cookie=ABCDE

amazon.com
dbclk.js
criteo
Goal

Develop a method to identify information flows (cookie matching) between ad exchanges

• Mechanism agnostic: resilient to obfuscation
• Platform agnostic: detect sharing on the client- and server-side
Goal

Develop a method to identify information flows (cookie matching) between ad exchanges

• Mechanism agnostic: resilient to obfuscation
• Platform agnostic: detect sharing on the client- and server-side
Key Insight: Use Retargeted Ads

Retargeted ads are the most highly targeted form of online ads.
Key Insight: Use Retargeted Ads

Retargeted ads are the most highly targeted form of online ads

Cisco-Linksys AE1000 High-Performance Wireless-N Adapter
by Linksys

Price: $15.99 Prime

Only 1 left in stock.
Want it Tuesday, June 14? Order within 33 hrs 50 mins and choose One-Day Shipping at checkout. Details
Sold by Home Sweet Home Direct and Fulfilled by Amazon.
Eligible for AmazonSmile donation.

Want to hire a computer technician?
Buy professional computer technician services directly on Amazon. Backed by our Happiness Guarantee.
Learn more

**Key Insight: Use Retargeted Ads**

Retargeted ads are the most highly targeted form of online ads.

Key insight: because retargets are so specific, they can be used to conduct controlled experiments.

- Information **must be** shared between ad exchanges to serve retargeted ads.
Contributions

1. Novel methodology for identifying information flows between ad exchanges

2. Demonstrate the impact of ad network obfuscation in practice
   • 31% of cookie matching partners cannot be identified using heuristics

3. Develop a method to categorize information sharing relationships

4. Use graph analysis to infer the roles of actors in the ad ecosystem
Contributions

1. Novel methodology for identifying information flows between ad exchanges

2. Demonstrate the impact of ad network obfuscation in practice
   • 31% of cookie matching partners cannot be identified using heuristics

3. Develop a method to categorize information sharing relationships

4. Use graph analysis to infer the roles of actors in the ad ecosystem
Data Collection
Classifying Ad Network Flows
Results
Using Retargets as an Experimental Tool

Key observation: retargets are only served under very specific circumstances

1) Chrome, Amazon, Criteo
Using Retargets as an Experimental Tool

Key observation: retargets are only served under very specific circumstances

1) Advertiser observes the user at a shop
Using Retargets as an Experimental Tool

Key observation: retargets are only served under very specific circumstances

1) Advertiser observes the user at a shop

2)
Using Retargets as an Experimental Tool

Key observation: retargets are only served under very specific circumstances

1) Advertiser observes the user at a shop

2) Advertiser and the exchange must have matched cookies
Using Retargets as an Experimental Tool

Key observation: retargets are only served under very specific circumstances

1) Advertiser observes the user at a shop
   Advertiser and the exchange must have matched cookies

2) This implies a causal flow of information from Exchange → Advertiser
Data Collection Overview
Data Collection Overview

Visit Persona

Single Persona
10 websites/persona
10 products/website
Data Collection Overview

Visit Persona

Single Persona
10 websites/persona
10 products/website

Visit Publishers

150 Publishers
15 pages/publisher
Data Collection Overview

Visit Persona

Single Persona
10 websites/persona
10 products/website

Visit Publishers

150 Publishers
15 pages/publisher

Store Images,
Inclusion Chains,
HTTP requests/
responses

571,636
Images
Data Collection Overview

Visit Persona

Single Persona
10 websites/persona
10 products/website

Visit Publishers

150 Publishers
15 pages/publisher

Store Images,
Inclusion Chains,
HTTP requests/
responses

571,636
Images
Data Collection Overview

90 Personas

Visit Persona

Single Persona
10 websites/persona
10 products/website

Visit Publishers

150 Publishers
15 pages/publisher

Store Images,
Inclusion Chains,
HTTP requests/
responses

571,636 Images
Data Collection Overview

- 90 Personas
- Visit Persona
  - Single Persona
    - 10 websites/persona
    - 10 products/website
  - Visit Publishers
    - 150 Publishers
      - 15 pages/publisher

- Store Images, Inclusion Chains, HTTP requests/responses
- Ad Detection
  - Filter Images which appeared in > 1 persona
- Potential Targeted Ads
  - 31,850

- 571,636 Images
Data Collection Overview

90 Personas

Visit Persona

Single Persona
10 websites/persona
10 products/website

Visit Publishers

150 Publishers
15 pages/publisher

Store Images,
Inclusion Chains,
HTTP requests/
responses

Crowd Sourcing

Isolated Retargeted Ads

Potential Targeted Ads
31,850

Filter Images which appeared in > 1 persona

Ad Detection

571,636 Images
Crowd Sourcing

We used Amazon Mechanical Turk (AMT) to label 31,850 ads.
Crowd Sourcing

We used Amazon Mechanical Turk (AMT) to label 31,850 ads.

• Total 1,142 Tasks.
• 30 ads / Task.
• 27 unlabeled.
• 3 labeled by us.
• 2 workers per ad.
• $415 spent.
Crowd Sourcing

We used Amazon Mechanical Turk (AMT) to label 31,850 ads.

- Total 1,142 Tasks.
- 30 ads / Task.
- 27 unlabeled.
- 3 labeled by us.
- 2 workers per ad.
- $415 spent.
We used Amazon Mechanical Turk (AMT) to label 31,850 ads.

- Total 1,142 Tasks.
- 30 ads / Task.
- 27 unlabeled.
- 3 labeled by us.
- 2 workers per ad.
- $415 spent.
Crowd Sourcing

We used Amazon Mechanical Turk (AMT) to label 31,850 ads.

- Total 1,142 Tasks.
- 30 ads / Task.
- 27 unlabeled.
- 3 labeled by us.
- 2 workers per ad.
- $415 spent.
Final Dataset

5,102 unique retargeted ads
  • From 281 distinct online retailers

35,448 publisher-side chains that served the retargets
  • We observed some retargets multiple times
Data Collection

Classifying Ad Network Flows

Results
A look at Publisher Chains
A look at Publisher Chains

Publisher-side chain
A look at Publisher Chains

Example

Shopper-side chain

Publisher-side chain
A look at Publisher Chains

Example

- How does Criteo know to serve ad on BBC?
A look at Publisher Chains

Example

**Shopper-side chain**

![Shopper-side chain diagram]

**Publisher-side chain**

![Publisher-side chain diagram]

- How does Criteo know to serve ad on BBC?
  - In this case it is pretty trivial.
  - Criteo observed us on the shopper.
A look at Publisher Chains

**Example**

**Shopper-side chain**

- How does Criteo know to serve ad on BBC?
  - In this case it is pretty trivial.
  - Criteo observed us on the shopper.

**Publisher-side chain**

- Can we classify all such publisher-side chains?
What is a Chain?
What is a Chain?
What is a Chain?
What is a Chain?
What is a Chain?
What is a Chain?
What is a Chain?
What is a Chain?

[Diagram showing network flow with logos for CNN, rubicon project, criteo, and doubleclick by Google]
What is a Chain?
What is a Chain?
What is a Chain?
What is a Chain?

Diagram showing the flow of data from CNN, Rubicon Project, Criteo, and Amazon.com.
What is a Chain?
Four Classifications

Four possible ways for a retargeted ad to be served

1. Direct (Trivial) Matching
2. Cookie Matching
3. Indirect Matching
4. Latent (Server-side) Matching
Four Classifications

Four possible ways for a retargeted ad to be served

1. Direct (Trivial) Matching
2. Cookie Matching
3. Indirect Matching
4. Latent (Server-side) Matching
1) Direct (Trivial) Matching

Example:

- Shopper-side: Amazon → criteo
- Publisher-side: BBC → criteo
1) Direct (Trivial) Matching

**Example**

- **Shopper-side**
  - Icon: Amazon (Amazon)
  - Arrow: criteo

- **Publisher-side**
  - Icon: BBC
  - Arrow: criteo

**Rule**

- **Shopper-side**
  - `^shop.*a.*$`

- **Publisher-side**
  - `^pub a$`
1) Direct (Trivial) Matching

**Example**

![Example Diagram](image)

**Rule**

\[ ^{\text{shop}} \rightarrow \text{.} \rightarrow a \rightarrow \text{.}\$ \]

\[ ^{\text{pub}} \rightarrow a\$ \]

\text{a is the advertiser that serves the retarget}
1) Direct (Trivial) Matching

**Shopper-side**

Example:

- `^shop.*a.*$`  
  - `a` must appear on the shopper-side...

**Publisher-side**

Example:

- `^pub a$`  
  - `a` is the advertiser that serves the retarget

... but other trackers may also appear
2) Cookie Matching

Example:

Shopper-side:

Publisher-side:

Rule:
2) Cookie Matching

**Shopper-side**

Example

![Amazon](a) → criteo

Rule

^shop.*a.*$

**Publisher-side**

Example

![BBC](BBC) → doubleclick → criteo

Rule

^pub.*e.a$
2) Cookie Matching

**Example**

Shopper-side

```
^shop.*a.*$
```

Publisher-side

```
^pub.*e.*a$
```

e precedes a, which implies an RTB auction
2) Cookie Matching

**Example**

- **Shopper-side**
  - ![Amazon Logo](image1)
  - ![Criteo Logo](image2)

- **Publisher-side**
  - ![BBC Logo](image3)
  - ![DoubleClick Logo](image4)
  - ![Criteo Logo](image2)

**Rule**

- **Shopper-side**
  - `^shop.*a.*$`
  - *a* must appear on the shopper-side

- **Publisher-side**
  - `^pub.*e.a$`
  - *e* precedes *a*, which implies an RTB auction
2) Cookie Matching

**Shopper-side**

Example:

```
^shop.*a.*$
```

**Publisher-side**

Example:

```
^pub.*e a$
```

**Anywhere**

Example:

```
.*doubleclick
```

**Rule**

- `^shop.*a.*$` must appear on the shopper-side.
- `^.*e a.*$` transition `e→a` is where cookie match occurs.
- `^.*e a$` `e` precedes `a`, which implies an RTB auction.
3) Latent (Server-side) Matching

Example

Shopper-side

Publisher-side

Rule

Amazon ➔ Google

BBC ➔ Doubleclick ➔ 2mdn
3) Latent (Server-side) Matching

**Example**

- **Shopper-side**
  - ^shop
  - [^ea]$

- **Publisher-side**
  - ^pub
  - .*
  - e
  - a$

Example of the process: Amazon (shop) sends data to Google (ea) on the shopper-side. The BBC (pub) receives data from Google (ea) and sends it to DoubleClick (doubleclick), which forwards it to 2mdn.
3) Latent (Server-side) Matching

### Example

**Shopper-side**

![Amazon](a) \(\rightarrow\) ![Google](g)

**Publisher-side**

![BBC](bbc) \(\rightarrow\) ![doubleclick](doubleclick) \(\rightarrow\) 2mdn

### Rule

- **Shopper-side**: \(^{\text{^shop}}\) [^ea]$  
  - Neither e nor a appears on the shopper-side

- **Publisher-side**: ^pub.*e.a$
3) Latent (Server-side) Matching

**Shopper-side**

Example

```
^shop -> [^ea]$
```

Neither e nor a appears on the shopper-side

**Publisher-side**

Rule

```
^pub .* e a$
```

a must receive information from some shopper-side tracker
3) Latent (Server-side) Matching

**Example**

Shopper-side: Amazon (a) -> Google (G)

Publisher-side: BBC (B) -> Doubleclick (D) -> 2mdn

**Rule**

- `^shop` -> `[^ea]$
  - Neither e nor a appears on the shopper-side

- `^pub` -> `.*` -> e -> a$
  - a must receive information from some shopper-side tracker

We find latent matches in practice!
Data Collection
Classifying Ad Network Flows
Results
Categorizing Chains

<table>
<thead>
<tr>
<th>Type</th>
<th>Chains</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct (Trivial) Match</td>
<td>1770</td>
<td>5</td>
</tr>
<tr>
<td>Cookie Match</td>
<td>25049</td>
<td>71</td>
</tr>
<tr>
<td>Latent (Server-side) Match</td>
<td>5362</td>
<td>15</td>
</tr>
<tr>
<td>No Match</td>
<td>775</td>
<td>2</td>
</tr>
</tbody>
</table>
Categorizing Chains

<table>
<thead>
<tr>
<th>Type</th>
<th>Chains</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct (Trivial) Match</td>
<td>1770</td>
<td>5</td>
</tr>
<tr>
<td>Cookie Match</td>
<td>25049</td>
<td>71</td>
</tr>
<tr>
<td>Latent (Server-side) Match</td>
<td>5362</td>
<td>15</td>
</tr>
<tr>
<td>No Match</td>
<td>775</td>
<td>2</td>
</tr>
</tbody>
</table>

Take away:
Categorizing Chains

<table>
<thead>
<tr>
<th>Type</th>
<th>Chains</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct (Trivial) Match</td>
<td>1770</td>
<td>5</td>
</tr>
<tr>
<td>Cookie Match</td>
<td>25049</td>
<td>71</td>
</tr>
<tr>
<td>Latent (Server-side) Match</td>
<td>5362</td>
<td>15</td>
</tr>
<tr>
<td>No Match</td>
<td>775</td>
<td>2</td>
</tr>
</tbody>
</table>

Take away:

1- As expected, most retargets are due to cookie matching
Categorizing Chains

<table>
<thead>
<tr>
<th>Type</th>
<th>Chains</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct (Trivial) Match</td>
<td>1770</td>
<td>5</td>
</tr>
<tr>
<td>Cookie Match</td>
<td>25049</td>
<td>71</td>
</tr>
<tr>
<td>Latent (Server-side) Match</td>
<td>5362</td>
<td>15</td>
</tr>
<tr>
<td>No Match</td>
<td>775</td>
<td>2</td>
</tr>
</tbody>
</table>

**Take away:**

1- As expected, most retargets are due to cookie matching
2- Very small number of chains that cannot be categorized
   - Suggests low false positive rate of AMT image labeling task
## Categorizing Chains

### Raw Chains

<table>
<thead>
<tr>
<th>Type</th>
<th>Chains</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct (Trivial) Match</td>
<td>1770</td>
<td>5</td>
</tr>
<tr>
<td>Cookie Match</td>
<td>25049</td>
<td>71</td>
</tr>
<tr>
<td>Latent (Server-side) Match</td>
<td>5362</td>
<td>15</td>
</tr>
<tr>
<td>No Match</td>
<td>775</td>
<td>2</td>
</tr>
</tbody>
</table>

### Take away:

1- As expected, most retargets are due to cookie matching
2- Very small number of chains that cannot be categorized
   • Suggests low false positive rate of AMT image labeling task
3- Surprisingly large amount latent matches...
## Categorizing Chains

<table>
<thead>
<tr>
<th>Type</th>
<th>Raw Chains</th>
<th>%</th>
<th>Clustered Chains</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct (Trivial) Match</td>
<td>1770</td>
<td>5</td>
<td>8449</td>
<td>24</td>
</tr>
<tr>
<td>Cookie Match</td>
<td>25049</td>
<td>71</td>
<td>25873</td>
<td>73</td>
</tr>
<tr>
<td>Latent (Server-side) Match</td>
<td>5362</td>
<td>15</td>
<td>343</td>
<td>1</td>
</tr>
<tr>
<td>No Match</td>
<td>775</td>
<td>2</td>
<td>183</td>
<td>1</td>
</tr>
</tbody>
</table>

Cluster together domains by “owner”

- E.g. google.com, doubleclick.com, googlesyndication.com
Categorizing Chains

<table>
<thead>
<tr>
<th>Type</th>
<th>Raw Chains</th>
<th></th>
<th></th>
<th>Clustered Chains</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Chains</td>
<td>%</td>
<td></td>
<td>Chains</td>
<td>%</td>
</tr>
<tr>
<td>Direct (Trivial) Match</td>
<td>1770</td>
<td>5</td>
<td></td>
<td>8449</td>
<td>24</td>
</tr>
<tr>
<td>Cookie Match</td>
<td>25049</td>
<td>71</td>
<td></td>
<td>25873</td>
<td>73</td>
</tr>
<tr>
<td>Latent (Server-side) Match</td>
<td>5362</td>
<td>15</td>
<td></td>
<td>343</td>
<td>1</td>
</tr>
<tr>
<td>No Match</td>
<td>775</td>
<td>2</td>
<td></td>
<td>183</td>
<td>1</td>
</tr>
</tbody>
</table>

Cluster together domains by “owner”
- E.g. google.com, doubleclick.com, googlesyndication.com

Latent matches essentially disappear
- The vast majority of these chains involve Google
- Suggests that Google shares tracking data across their services
Who is Cookie Matching?

<table>
<thead>
<tr>
<th>Participant 1</th>
<th>Participant 2</th>
<th>Chains</th>
<th>Ads</th>
<th>Heuristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteo</td>
<td>googlesyndication</td>
<td>9090</td>
<td>1887</td>
<td>↔️ P</td>
</tr>
<tr>
<td>criteo</td>
<td>doubleclick</td>
<td>3610</td>
<td>1144</td>
<td>→ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>adnxs</td>
<td>3263</td>
<td>1066</td>
<td>↔️ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>rubiconproject</td>
<td>1586</td>
<td>749</td>
<td>↔️ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>servedbyopenx</td>
<td>707</td>
<td>460</td>
<td>↔️ P</td>
</tr>
<tr>
<td>doubleclick</td>
<td>steelhousemedia</td>
<td>362</td>
<td>27</td>
<td>→ P ↔️ E, P</td>
</tr>
<tr>
<td>mathtag</td>
<td>mediaforge</td>
<td>360</td>
<td>124</td>
<td>↔️ E, P</td>
</tr>
<tr>
<td>netmng</td>
<td>scene7</td>
<td>267</td>
<td>119</td>
<td>→ E ↔️ ?</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>adsrvr</td>
<td>107</td>
<td>29</td>
<td>↔️ P</td>
</tr>
<tr>
<td>rubiconproject</td>
<td>steelhousemedia</td>
<td>86</td>
<td>30</td>
<td>↔️ E</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>steelhousemedia</td>
<td>47</td>
<td>22</td>
<td>?</td>
</tr>
<tr>
<td>adtechus</td>
<td>adacado</td>
<td>36</td>
<td>18</td>
<td>?</td>
</tr>
<tr>
<td>atwola</td>
<td>adacado</td>
<td>32</td>
<td>6</td>
<td>?</td>
</tr>
<tr>
<td>adroll</td>
<td>adnxs</td>
<td>31</td>
<td>8</td>
<td>?</td>
</tr>
</tbody>
</table>

**Heuristics Key**
(used by prior work)

- **E** – share exact cookies
- **P** – special URL parameters
- **DC** – DoubleClick URL parameters
- **?** – Unknown sharing method
## Who is Cookie Matching?

<table>
<thead>
<tr>
<th>Participant 1</th>
<th>Participant 2</th>
<th>Chains</th>
<th>Ads</th>
<th>Heuristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteo</td>
<td>googlesyndication</td>
<td>9090</td>
<td>1887</td>
<td>⇔ P</td>
</tr>
<tr>
<td>criteo</td>
<td>doubleclick</td>
<td>3610</td>
<td>1144</td>
<td>⇔ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>adnxs</td>
<td>3263</td>
<td>1066</td>
<td>⇔ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>rubiconproject</td>
<td>1586</td>
<td>749</td>
<td>⇔ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>servedbyopenx</td>
<td>707</td>
<td>460</td>
<td>⇔ P</td>
</tr>
<tr>
<td>doubleclick</td>
<td>steelhousemedia</td>
<td>362</td>
<td>27</td>
<td>⇔ P, ⇔ E, P</td>
</tr>
<tr>
<td>mathtag</td>
<td>mediaforge</td>
<td>360</td>
<td>124</td>
<td>⇔ E, P</td>
</tr>
<tr>
<td>netmng</td>
<td>scene7</td>
<td>267</td>
<td>119</td>
<td>⇔ E, ⇔ ?</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>googlesyndication</td>
<td>107</td>
<td>29</td>
<td>⇔ P</td>
</tr>
<tr>
<td>rubiconproject</td>
<td>steelhousemedia</td>
<td>86</td>
<td>30</td>
<td>⇔ E</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>steelhousemedia</td>
<td>47</td>
<td>22</td>
<td>⇔ ?</td>
</tr>
<tr>
<td>adtechus</td>
<td>adacado</td>
<td>36</td>
<td>18</td>
<td>⇔ ?</td>
</tr>
<tr>
<td>atwola</td>
<td>adacado</td>
<td>32</td>
<td>6</td>
<td>⇔ ?</td>
</tr>
<tr>
<td>adroll</td>
<td>adnxs</td>
<td>31</td>
<td>8</td>
<td>⇔ ?</td>
</tr>
</tbody>
</table>

### Heuristics Key (used by prior work)

- **E** – share exact cookies
- **P** – special URL parameters
- **DC** – DoubleClick URL parameters
- **?** – Unknown sharing method
## Who is Cookie Matching?

### Heuristics Key
- **E** – share exact cookies
- **P** – special URL parameters
- **DC** – DoubleClick URL parameters
- **?** – Unknown sharing method

<table>
<thead>
<tr>
<th>Participant 1</th>
<th>Participant 2</th>
<th>Chains</th>
<th>Ads</th>
<th>Heuristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteo</td>
<td>googlesyndication</td>
<td>9090</td>
<td>1887</td>
<td>↞ P</td>
</tr>
<tr>
<td>criteo</td>
<td>doubleclick</td>
<td>3610</td>
<td>1144</td>
<td>E, P ↞ DC, P</td>
</tr>
<tr>
<td>criteo</td>
<td>adnxs</td>
<td>3263</td>
<td>1066</td>
<td>↞ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>rubiconproject</td>
<td>1586</td>
<td>749</td>
<td>↞ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>servedbyopenx</td>
<td>707</td>
<td>460</td>
<td>↞ P</td>
</tr>
<tr>
<td>doubleclick</td>
<td>steelhousemedia</td>
<td>362</td>
<td>27</td>
<td>P ↞ E, P</td>
</tr>
<tr>
<td>mathtag</td>
<td>mediaforge</td>
<td>360</td>
<td>124</td>
<td>↞ E, P</td>
</tr>
<tr>
<td>netmng</td>
<td>scene7</td>
<td>267</td>
<td>119</td>
<td>E ↞ ?</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>adsrvr</td>
<td>107</td>
<td>29</td>
<td>↞ P</td>
</tr>
<tr>
<td>rubiconproject</td>
<td>steelhousemedia</td>
<td>86</td>
<td>30</td>
<td>↞ E</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>steelhousemedia</td>
<td>47</td>
<td>22</td>
<td>?</td>
</tr>
<tr>
<td>adtechus</td>
<td>adacado</td>
<td>36</td>
<td>18</td>
<td>?</td>
</tr>
<tr>
<td>atwola</td>
<td>adacado</td>
<td>32</td>
<td>6</td>
<td>?</td>
</tr>
<tr>
<td>adroll</td>
<td>adnxs</td>
<td>31</td>
<td>8</td>
<td>?</td>
</tr>
</tbody>
</table>
Who is Cookie Matching?

<table>
<thead>
<tr>
<th>Participant 1</th>
<th>Participant 2</th>
<th>Chains</th>
<th>Ads</th>
<th>Heuristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteo</td>
<td>googlesyndication</td>
<td>9090</td>
<td>1887</td>
<td>↔️ P</td>
</tr>
<tr>
<td>criteo</td>
<td>doubleclick</td>
<td>3610</td>
<td>1144</td>
<td>E, P, ↔️ DC, P</td>
</tr>
<tr>
<td>criteo</td>
<td>adnxs</td>
<td>3263</td>
<td>1066</td>
<td>↔️ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>rubiconproject</td>
<td>1586</td>
<td>749</td>
<td>↔️ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>servedbyopenx</td>
<td>707</td>
<td>460</td>
<td>↔️ P</td>
</tr>
<tr>
<td>doubleclick</td>
<td>steelhousemedia</td>
<td>362</td>
<td>27</td>
<td>P, ↔️ E, P</td>
</tr>
<tr>
<td>mathtag</td>
<td>mediaforge</td>
<td>360</td>
<td>124</td>
<td>↔️ E, P</td>
</tr>
<tr>
<td>netmng</td>
<td>scene7</td>
<td>267</td>
<td>119</td>
<td>E, ↔️ ?</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>adsrvr</td>
<td>107</td>
<td>29</td>
<td>↔️ P</td>
</tr>
<tr>
<td>rubiconproject</td>
<td>steelhousemedia</td>
<td>86</td>
<td>30</td>
<td>↔️ E</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>steelhousemedia</td>
<td>47</td>
<td>22</td>
<td>?</td>
</tr>
<tr>
<td>adtechus</td>
<td>adacado</td>
<td>36</td>
<td>18</td>
<td>?</td>
</tr>
<tr>
<td>atwola</td>
<td>adacado</td>
<td>32</td>
<td>6</td>
<td>?</td>
</tr>
<tr>
<td>adroll</td>
<td>adnxs</td>
<td>31</td>
<td>8</td>
<td>?</td>
</tr>
</tbody>
</table>

**Heuristics Key**

(used by prior work)

- **E** – share exact cookies
- **P** – special URL parameters
- **DC** – DoubleClick URL parameters
- **?** – Unknown sharing method
### Who is Cookie Matching?

**Heuristics Key**
- **E** – share exact cookies
- **P** – special URL parameters
- **DC** – DoubleClick URL parameters
- **?”** – Unknown sharing method

<table>
<thead>
<tr>
<th>Participant 1</th>
<th>Participant 2</th>
<th>Chains</th>
<th>Ads</th>
<th>Heuristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteo</td>
<td>googlesyndication</td>
<td>9090</td>
<td>1887</td>
<td>↔ P</td>
</tr>
<tr>
<td>criteo</td>
<td>doubleclick</td>
<td>3610</td>
<td>1144</td>
<td>→ E, P, ↔ DC, P</td>
</tr>
<tr>
<td>criteo</td>
<td>adnxs</td>
<td>3263</td>
<td>1066</td>
<td>↔ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>rubiconproject</td>
<td>1586</td>
<td>749</td>
<td>↔ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>servedbyopenx</td>
<td>707</td>
<td>460</td>
<td>↔ P</td>
</tr>
<tr>
<td>doubleclick</td>
<td>steelhousemedia</td>
<td>362</td>
<td>27</td>
<td>→ P, ↔ E, P</td>
</tr>
<tr>
<td>mathtag</td>
<td>mediaforge</td>
<td>360</td>
<td>124</td>
<td>↔ E, P</td>
</tr>
<tr>
<td>netmng</td>
<td>scene7</td>
<td>267</td>
<td>119</td>
<td>→ E, ↔ ?</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>adsrvr</td>
<td>107</td>
<td>29</td>
<td>↔ P</td>
</tr>
<tr>
<td>rubiconproject</td>
<td>steelhousemedia</td>
<td>86</td>
<td>30</td>
<td>↔ E</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>steelhousemedia</td>
<td>47</td>
<td>22</td>
<td>?</td>
</tr>
<tr>
<td>adechus</td>
<td>adacado</td>
<td>36</td>
<td>18</td>
<td>?</td>
</tr>
<tr>
<td>atwola</td>
<td>adacado</td>
<td>32</td>
<td>6</td>
<td>?</td>
</tr>
<tr>
<td>adroll</td>
<td>adnxs</td>
<td>31</td>
<td>8</td>
<td>?</td>
</tr>
</tbody>
</table>
### Who is Cookie Matching?

<table>
<thead>
<tr>
<th>Participant 1</th>
<th>Participant 2</th>
<th>Chains</th>
<th>Ads</th>
<th>Heuristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteo</td>
<td>googlesyndication</td>
<td>9090</td>
<td>1887</td>
<td>⟷ P</td>
</tr>
<tr>
<td>criteo</td>
<td>doubleclick</td>
<td>3610</td>
<td>1144</td>
<td>E, P ⟷ DC, P</td>
</tr>
<tr>
<td>criteo</td>
<td>adnxs</td>
<td>3263</td>
<td>1066</td>
<td>⟷ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>rubiconproject</td>
<td>1586</td>
<td>749</td>
<td>⟷ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>servedbyopenx</td>
<td>707</td>
<td>460</td>
<td>⟷ P</td>
</tr>
<tr>
<td>doubleclick</td>
<td>steelhousemedia</td>
<td>362</td>
<td>27</td>
<td>P ⟷ E, P</td>
</tr>
<tr>
<td>mathtag</td>
<td>mediaforge</td>
<td>360</td>
<td>124</td>
<td>⟷ E, P</td>
</tr>
<tr>
<td>netmng</td>
<td>scene7</td>
<td>267</td>
<td>119</td>
<td>E ⟷ ?</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>adsrvr</td>
<td>107</td>
<td>29</td>
<td>⟷ P</td>
</tr>
<tr>
<td>rubiconproject</td>
<td>steelhousemedia</td>
<td>86</td>
<td>30</td>
<td>⟷ E</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>steelhousemedia</td>
<td>47</td>
<td>27</td>
<td>?</td>
</tr>
<tr>
<td>adtechus</td>
<td>adacado</td>
<td>36</td>
<td>18</td>
<td>?</td>
</tr>
<tr>
<td>atwola</td>
<td>adacado</td>
<td>32</td>
<td>6</td>
<td>?</td>
</tr>
<tr>
<td>adroll</td>
<td>adnxs</td>
<td>31</td>
<td>8</td>
<td>?</td>
</tr>
</tbody>
</table>

**Heuristics Key**

- **E** – share exact cookies
- **P** – special URL parameters
- **DC** – DoubleClick URL parameters
- **?** – Unknown sharing method
## Who is Cookie Matching?

<table>
<thead>
<tr>
<th>Participant 1</th>
<th>Participant 2</th>
<th>Chains</th>
<th>Ads</th>
<th>Heuristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteo</td>
<td>googlesyndication</td>
<td>9090</td>
<td>1887</td>
<td>←→ P</td>
</tr>
<tr>
<td>criteo</td>
<td>doubleclick</td>
<td>3610</td>
<td>1144</td>
<td>→ E, P ← DC, P</td>
</tr>
<tr>
<td>criteo</td>
<td>adnxs</td>
<td>3263</td>
<td>1066</td>
<td>←→ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>rubiconproject</td>
<td>1586</td>
<td>749</td>
<td>←→ E, P</td>
</tr>
<tr>
<td>criteo</td>
<td>servedbyopenx</td>
<td>707</td>
<td>460</td>
<td>←→ E, P</td>
</tr>
<tr>
<td>doubleclick</td>
<td>steelhousemedia</td>
<td>362</td>
<td>27</td>
<td>→ P ← E, P</td>
</tr>
<tr>
<td>mathtag</td>
<td>mediaforge</td>
<td>360</td>
<td>124</td>
<td>←→ E, P</td>
</tr>
<tr>
<td>netmng</td>
<td>scene7</td>
<td>267</td>
<td>119</td>
<td>→ E ← ?</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>adsrvr</td>
<td>107</td>
<td>29</td>
<td>←→ P</td>
</tr>
<tr>
<td>rubiconproject</td>
<td>steelhousemedia</td>
<td>86</td>
<td>30</td>
<td>←→ E</td>
</tr>
<tr>
<td>googlesyndication</td>
<td>steelhousemedia</td>
<td>47</td>
<td>22</td>
<td>?</td>
</tr>
<tr>
<td>adtechus</td>
<td>adacado</td>
<td>36</td>
<td>18</td>
<td>?</td>
</tr>
<tr>
<td>atwola</td>
<td>adacado</td>
<td>32</td>
<td>8</td>
<td>?</td>
</tr>
<tr>
<td>adroll</td>
<td>adnxs</td>
<td>31</td>
<td>8</td>
<td>?</td>
</tr>
</tbody>
</table>

### Heuristics Key
(used by prior work)
- **E** – share exact cookies
- **P** – special URL parameters
- **DC** – DoubleClick URL parameters
- **?** – Unknown sharing method

31% of cookie matching partners would be missed.
Summary

We develop a novel methodology to detect information flows between ad exchanges

- Controlled methodology enables causal inference
- Defeats obfuscation attempts
- Detects client- and server-side flows

Dataset gives a better picture of ad ecosystem

- Reveals which ad exchanges are linking information about users
- Allows us to reason about how information is being transferred
Questions?
Muhammad Ahmad Bashir
ahmad@ccs.neu.edu
Inclusion Chains

• Instrumented Chromium binary that records the provenance of page elements
  • Uses Information Flow Analysis techniques (IFA)
  • Handles Flash, `exec()`, `setTimeout()`, cross-frame, inline scripts, etc.
Inclusion Chains

• Instrumented Chromium binary that records the provenance of page elements
  • Uses Information Flow Analysis techniques (IFA)
  • Handles Flash, `exec()`, `setTimeout()`, cross-frame, inline scripts, etc.

**DOM: a.com/index.html**

```html
<html>
<body>
  <script src="b.com/adlib.js"></script>
  <iframe src="c.net/adbox.html">
    <html>
      <script src="code.js"></script>
      <object data="d.org/flash.swf"></object>
    </html>
  </iframe>
</body>
</html>
```
Inclusion Chains

- Instrumented Chromium binary that records the provenance of page elements
  - Uses Information Flow Analysis techniques (IFA)
  - Handles Flash, `exec()`, `setTimeout()`, cross-frame, inline scripts, etc.

**DOM: a.com/index.html**

```html
<html>
<body>
<script src="b.com/adlib.js"></script>
<iframe src="c.net/adbox.html">
  <html>
    <script src="code.js"></script>
    <object data="d.org/flash.swf">
    </object>
  </html>
</iframe>
</body>
</html>
```
3) Indirect Matching

Example

Shopper-side

Publisher-side

Rule
3) Indirect Matching

**Shopper-side**

Example: Amazon (shop) pointing to DoubleClick (pub).

Rule: `^shop[^a]e[^a]$`

**Publisher-side**

Example: BBC (pub) pointing to DoubleClick then Criteo.

Rule: `^pub.*e a$`
3) Indirect Matching

**Example**

- **Shopper-side**
  - Rule: `^shop[^a]e[^a]$`
  - Example: Amazon (shop) → doubleclick

- **Publisher-side**
  - Rule: `^pub.*e[a]$`
  - Example: BBC (pub) → doubleclick → criteo.

Only the exchange e appears on the shopper-side...
3) Indirect Matching

**Example**

![Amazon logo](image) → doubleclick

**Rule**

```
^shop -> [^a] e -> [^a]$ 
```

- Only the exchange e appears on the shopper-side...

**Publisher-side**

![BBC logo](image) → doubleclick → criteol.

**Rule**

```
^pub -> .* e -> a$ 
```

- e must pass browsing history data to participants in the auction, thus no cookie matching is necessary
3) Indirect Matching

**Example**

**Shopper-side**

![Amazon](a) → doubleclick

![BBC](BBC) → doubleclick → criteo

**Rule**

\[ ^\text{shop} \rightarrow [^a] \rightarrow e \rightarrow [^a]$ \]

Only the exchange e appears on the shopper-side...

\[ ^\text{pub} \rightarrow .* \rightarrow e \rightarrow a$ \]

e must pass browsing history data to participants in the auction, thus no cookie matching is necessary.

**We do not expect to find indirect matches in the data.**
References


Filtering Images

<table>
<thead>
<tr>
<th>Filter</th>
<th>Total Unique Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>All images from the crawlers</td>
<td>571,636</td>
</tr>
</tbody>
</table>

Use EasyList to identify advertisements.
Remove ads that are shown to >1 persona.
## Filtering Images

<table>
<thead>
<tr>
<th>Filter</th>
<th>Total Unique Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>All images from the crawlers</td>
<td>571,636</td>
</tr>
<tr>
<td>Use EasyList to identify advertisements</td>
<td>93,726</td>
</tr>
</tbody>
</table>
Filtering Images

<table>
<thead>
<tr>
<th>Filter</th>
<th>Total Unique Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>All images from the crawlers</td>
<td>571,636</td>
</tr>
<tr>
<td>Use EasyList to identify advertisements</td>
<td>93,726</td>
</tr>
<tr>
<td>Remove ads that are shown to &gt;1 persona</td>
<td>31,850</td>
</tr>
</tbody>
</table>

• Personas visited non-overlapping retailers
  • By definition, retargets should only be shown to a single persona
Filtering Images

<table>
<thead>
<tr>
<th>Filter</th>
<th>Total Unique Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>All images from the crawlers</td>
<td>571,636</td>
</tr>
<tr>
<td>Use EasyList to identify advertisements</td>
<td>93,726</td>
</tr>
<tr>
<td>Remove ads that are shown to &gt;1 persona</td>
<td>31,850</td>
</tr>
<tr>
<td>Use crowdsourcing to locate retargets</td>
<td>5,102</td>
</tr>
</tbody>
</table>

- Personas visited non-overlapping retailers
  - By definition, retargets should only be shown to a single persona
- Spent $415 uploading 1,142 HITs to Amazon Mechanical Turk
  - Each HIT asked the worker to label 30 ad images
  - 27 were unlabeled, 3 were known retargets (control images)
  - All ads were labeled by 2 workers
  - Any ad identified as targeted was also manually inspected by us