Data Leakage: Collusion Graph for determining usage of Beacons and their providers

Priya Goyal (prgoyal@iitk.ac.in), Shilpa Sharma (shilpash@yahoo-inc.com), Chethan Prakash (chethanp@yahoo-inc.com)
Trust & Safety, Advertising and Data Platform

Abstract

Data Leakage in the digital advertising world occurs when third parties obtain information about an online publisher's audience without compensation or even the knowledge of the publisher via usage of beacons/cookies. If you are a publisher who depends on advertising dollars to fund your operations, data leakage is a critical threat to your bottom line. Data means audience, and audiences are what advertisers pay to reach. If they can reach them without buying expensive content adjacency, they will. By allowing advertisers to cookie pool their users, advertisers can retarget the publisher's audience through cheaper channels. Besides, it leads to user's personal data theft, decay in the value of publisher's audience as it is now accessible through cheaper channels, huge cost on user's experience through page latency, reduction in the number of clients and ad network partners from publisher's value chain and many more. Publishers without a way to secure their data are pretty much asking to have their audience filtered away from them. They need a way to determine the usage of beacons/cookies and their providers and who this information is being made available to so that they can decide whether to allow/deny a vendor from putting beacon on their property.

In this paper, we look at the usage of beacons; how they are implemented; the malpractices associated with them and then come up with the solution to this problem by developing a tool that allows checking the usage of beacons and their providers along with the frequency of calls.

1. Introduction

A web bug is an object that is embedded in a web page or e-mail and is usually invisible to the user but allows checking that a user has viewed the page or e-mail. One common use is in e-mail tracking. Alternative names are ***web beacon***, ***tracking bug***, and ***tag*** or ***page tag***. Common names for web bugs implemented through an embedded image include ***tracking pixel***, ***pixel tag***, ***1×1 gif***, and ***clear/transparent gif***. To work effectively across sessions, Web Bug requires a unique id to be associated with each call, which can identify the user uniquely. Web technologies allow multiple ways of creating such persistent unique ids, which can be relayed back to Web Bug servers. Some of them are:

A. Cookies

On first visit, Web Bug server sets a unique cookie. When the user browses the same website in the future, the data stored in the cookie along with the referrer information is relayed back. (From where the call was made)

B. ETags

Browser uses a concept of ETag, for cache controlling. Every resource in a web server is associated with a unique Etag id, unless the content changes. This Etag gets stored into the browser, next time when this resource is to be loaded by the browser. The same Etag is sent back to the server, thus relaying back the same unique id.

C. Flash Cookies

Flash allows storing data in a local storage called, Local Storage Objects (LSO's). This storage objects are universal to a machine, not even browsers. So the same information (unique id in our case), can be accessed and relayed back from multiple browsers installed on the same machine.

D. HTML5 Storage

In HTML 5, new persistence storage api was provided for each domain. It acts similar to a Cookie, but allows more structured and more amount of data to be stored.

E. *Evercookie*

An open source API, which allows to create cookies on a machine, which are very difficult to get rid of.

The ***observed properties of Web Bugs*** are:

A. Light in payload:

This is because they want to capture the information as fast as possible, without affecting the overall page performance where it is hosted.

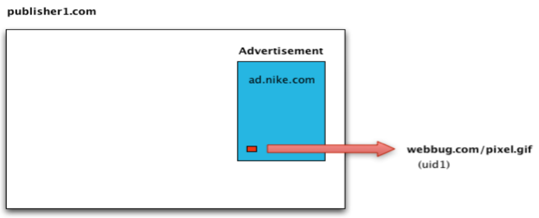
B. Invisibility:

Generally small 1x1 sized pixels are used, so that they are not visible to the user.
Note: These are just desired properties and ***not*** necessarily have to be followed.

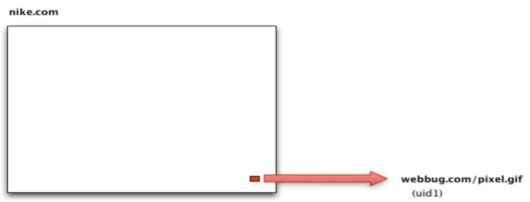
2. How are users Tracked?

This section explains how web bugs can be used to track users across advertisements and web pages.

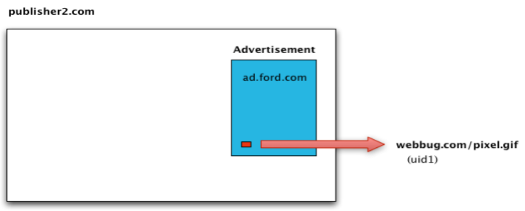
A. The user visits a publisher page ***publisher1.com*** showing an ad from ***ad.nike.com***, the ***web bug*** is notified about this call.



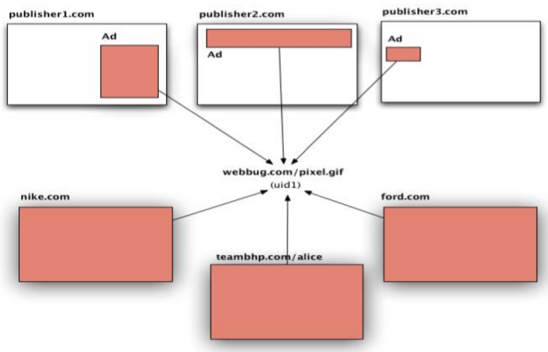
B. The user visits ***nike.com*** sometime later, which also contains a web bug. (Of course ***nike.com*** has added it, as they want to track their users)



C. Later the same user visits ***publisher2.com***, loading an ad from ***ad.ford.com***.



D. Slowly ***webbug.com*** gathers all the information attached to the specific ***uid***



3. What Information can be tracked?

A Web Bug along with it can carry following set of information about the user.

1. IP Address of the computer where it was accessed from. (Using Web Bug server logs, which can capture IP address of http calls)

2. URL of the page that the Web Bug is located on. (Using the ***Referrer*** header sent along with the HTTP call)

3. ***URL of the Web Bug image*** which can be constructed by the javascript running in the page, which captures extra information like, plugins used.

4. Time when page was viewed (Using Web Bug server logs, which can capture timestamps of http calls)

5. Browser used by the user (Using the ***User-Agent*** string sent along with the HTTP call)

6. A previously set cookie/ETag (A cookie/Etag which was set during the previous visits)

7. ***Machine properties***
(A javascript running on the web page can detect following properties, and relay it back to Web Bug

- *Screen Resolution*
- *Plugins available*
- *OS used*

4. Usage of Beacons

The intent of web bug is very difficult to determine. It can be used as follows:

4.1 Good Intent:

- Advertisers need to know how many times their ad has been shown on the publisher's page. Beacons/Cookies allow the 3rd party Ad serving website to collect this information.

- To track, from which demographical location a particular campaign is viewed. (Anonymized up to City, State our Country level).

- To track, how many users viewing an Ad Campaign, also visits the Landing page. This helps both the ad serving company and the advertiser determine if the particular ad campaign produced the desired results.

- Cookies/ Beacons can be used as a tool of measurement like counting the number of web page hits, track web site usage- page by linked page.

4.2 Bad Intent

- Cookies/ Beacons can be used to build a users interest profile (surfing habits by tracking the user across different websites) and sell it to potential advertisers, to retarget ad.

- To collect what kind of advertisements are actually targeted to a user through a Good Exchange, and utilize this fact to resale those positions.

Although the usage of beacons in online ad industry is legit practice, but there are concerns on what information is being tracked and how is it used. There are many instances of illegal use of information available via beacons/cookies: **[1]** Labor party tracks Ken Livingstone supporters with hidden web bugs accessible by USA tracking companies and US Government, [2] Ad network at the center of third flash cookie lawsuit and many more.

5. Solution to the problem

Although usage of beacons/cookies are legit means to track impressions, conversions, etc. - it is impossible to control the amount of information that is made available to the tracker as it is an inherent behavior of the protocol. As a result, this now falls upon the advertisers and publishers to control who this information is made available to, how much of that information is stored and what use it is being finally put to so that they can become actionable. This will allow them to decide whether to allow/deny some vendors from placing beacons on their property.

However, in order to accomplish this, **advertisers and publishers must be provided with tools in order to track the usage of such beacons and their providers in order to determine whether this information is being made available to an approved / authorized vendor**.

Ad Quality Systems at Yahoo! Plans to aid these advertisers and publishers in 3 phased manner:

1. **Discover and detect beacons**:
   a. As part of this initiative, all beacons/cookies/1x1 pixels being used in an ad tag would be detected and resolved to the corresponding vendor.
   b. Basic Protection: Block certain known offenders (and in-effect block creatives that serve beacons from such offenders) from polluting the ecosystem
2. **Notify seat-holders**: In order to make this feature usable and scalable, provide the seat-holders with a summary of beacons that have served on their pages. This in turn will provide the seat-holders with necessary insights wrt to beacons serving on their properties.
3. **Controls**: Provide seat-holders with the ability to selectively approve or ban creatives that serve beacons from certain known vendors.

As a result of this, publishers and networks can not only control the vendors who have access to their user's personal data but also protect themselves from data theft.

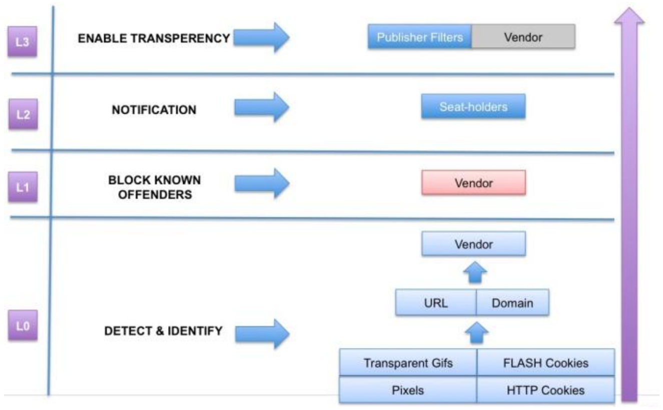


Figure 1

5.1 Detect and Identify

Given a url, we need to identify its potential tracking calls and associate them to appropriate vendors. For this we pass the creatives to the Creative Tester (CT) which is the internal tool of Trust and Safety team, RMX that analyses the creative and assigns them various violations tags. We begin with the files (***Filtered URL***) that are downloaded by CT as part of creative component and have size less than 100 bytes.

We use the ***Ghostery feed*** which is a full list of URL patterns identifying them as Trackers or Ad calls and maps them to the corresponding vendor. This feed looks like:

```
...
  ,{
    "type": "tracker",
    "aid": "2",
    "cid": "145",
    "pattern": "stat    ic\\.scribefire\\.com\\/ads\\.js",
    "name": "ScribeFire QuickAds",
```

"id": "33",
"affiliation": ""
},
…

5.1.1 Some terminologies/tools used

- ***Ghostery:*** Ghostery is a free privacy browser extension for Internet Explorer, Opera, Mozilla Firefox, Apple Safari, and Google Chrome that enables its users to easily detect and control tags, web bugs, pixels and beacons that have the potential to collect data on their browsing habits. Ghostery also has a privacy team that creates profiles of page elements and companies for educational purposes.
- The Ghostery extension downloads a database of 743 patterns identifying them as Trackers or Ad calls.
- ***Trackers:*** They are the parts of a website like ads that are loaded from other websites on visiting the webpage. Their types are:
  - **Advertiser** (AD): a tracker that delivers advertisements
  - **Analytics** (AN): a tracker that provides research/analytics for website publishers
  - **Beacon** (B): a tracker that exists only to track user behavior
  - **Widget** (W): a tracker that provides some kind of page function (comment forms, "Like" buttons, ...)
  - **Privacy** (P): a tracker that discloses data practices involved in delivering an ad.

Following are the statistics for the Web Bug tracker data from **28th June to 1st July**:

Total requests processed: 3901058
Tracker type numbers:

- ***Beacon:*** 7875597
- ***flash_cookie:*** 22349
- ***http_cookie:*** 2566812
- ***privacy:*** 1458803
- ***analytics:*** 3061779
- ***advertising:*** 29237848
- ***widget:*** 29237848

We pass the urls through the **Tracker Identification service** which classifies it as Tracker and Vendor using the Ghostery feed.

5.2 Collusion graph for beacon analysis

Collusion is an agreement between two or more parties, sometimes illegal and therefore secretive, to limit open competition by deceiving, misleading, or defrauding others of their legal rights, or to obtain an objective forbidden by law typically by defrauding or gaining an unfair advantage. In order to track the usage of beacons and their provider to determine whether the information is being made available to the authorized/ approved vendor, we need to look at the hierarchy of beacons. By using the mapped xml files for each creative (approx. 30,000), we construct this hierarchy graph. This beacon hierarchy graph will help observe the patterns of which vendors are big players and create a black-list, white-list of vendors.

In order to achieve this, we use the graph database (Neo4j) instead of relational database (MySQL). RDBMS are largely used for an aggregated data but the graph database can be used for highly connected data.

5.2.1 Neo4j Graph Database

A graph database stores data in a graph, the most generic of data structures, with the capability of elegantly representing any kind of data in a highly accessible way.

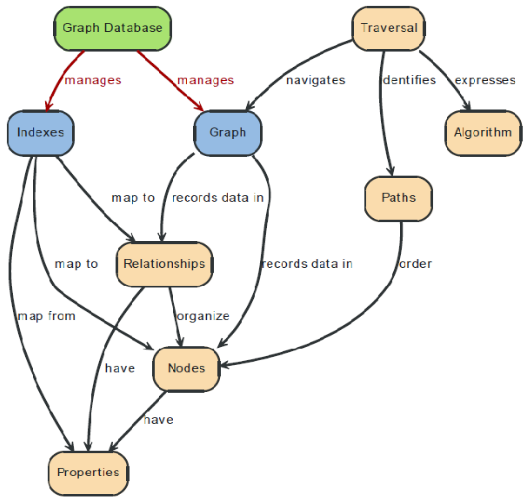


Figure 2: Overview of graph Database

The above figure gives the overview of graph database. The database constructs a special type of graph called *property graph* where the nodes and the relationships between the nodes can have properties i.e. key-value pairs. Any node and relationship can have any number of properties.

We chose graphDb over RDBMS because in RDBMS we use index stack to keep the relationship between two entities and have to do index look-up whenever searching for child/parent nodes. But for a highly connected data, this process becomes cumbersome and not scalable, less effective. The graphDb allows keeping the relationship between two nodes directly instead of keeping the indexes and also the relationships can have properties which may help in modeling the graph better. Apart from this, Neo4j allows performing all the operation of RDBMS also.

Other benefits: The data stored in graph database is very easily accessible. We can write various traversals for the graph, filter the traversal output as desired e.g. in social network graph like Facebook, we can easily query "the friends of friends who also like oranges and have a pet". Graph database allows querying this very easily.

For constructing the beacon hierarchy graph, we first construct the hierarchy graph of which url is calling whom and then by effectively using the graph database, we extract the beacon graph from this url hierarchy graph.

5.2.3 Gephi Visualization tool

Gephi is Open Source Visualization Platform build on top of the NetBeans platform. It is written in Java so you can run it on various Operating Systems including Windows, Linux, Mac OS. It supports many interesting graph analysis capabilities including:

- Real-time visualization
- Metrics
- Dynamic network analysis
- Cartography clustering and hierarchical graphs
- Dynamic filtering

For visualizing the graphs, we use this open source tool. This tool has integration with the Neo4j graph database and allows the full import of database. We can write various filters for visualizing the graph and this tool has a python console for this purpose. The tool has a data laboratory which allows visualizing the complete graph in tables and make edits to the node and relationship properties. We can write various traversals also for the graph and there are many plugins available for this. We can also export the graph database in various formats so that it can be reused.

5.2.4 Resulting Collusion Graph

We obtained the graph for the 30,000 xml files with 502 beacon nodes and 1939 edges and used the Gephi tool for visualizing this graph. We can use this graph to navigate from parent to child or vice-versa to see what vendors are called by whom. This graph also shows the total number of times a beacon was called, the number of times a beacon is calling another beacon, the parent beacons (blue line), child beacons (red line). Nodes have size according to the total number of times they were called and they have been distinguished in color according to the vendor name.

Also, we can customize the graph by writing various types of traversal queries.

The following figure show the navigation in the graph for vendor *[x+1]* with tracker type *beacon*.

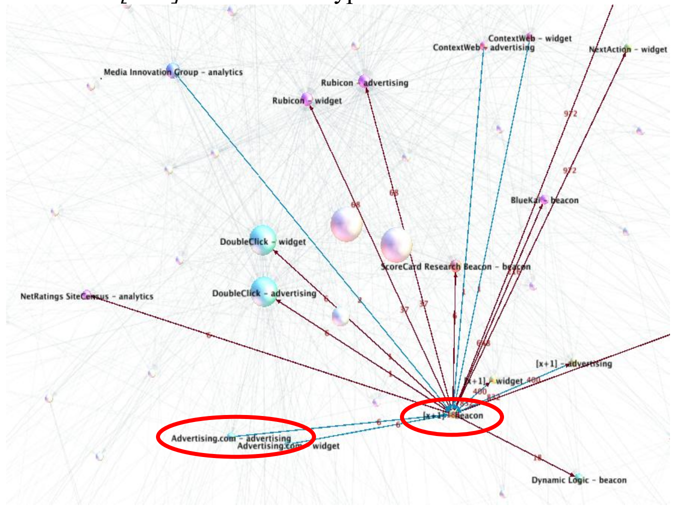


Figure 3

The next figure shows the navigation to its child node with vendor *Advertising.com* and tracker type *advertising*.

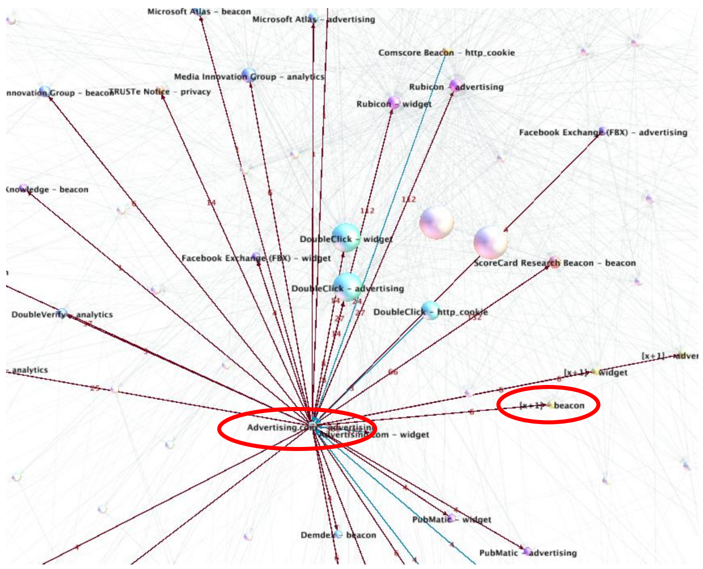

Microsoft Atlas – beacon
Microsoft Atlas – advertising
Comscore Beacon – http_cookie
Media Innovation Group – analytics
TRUSTe Notice – privacy
Rubicon – advertising
Rubicon – widget
Facebook Exchange (FBX) – advertising
DoubleClick – widget
Facebook Exchange (FBX) – widget
ScoreCard Research Beacon – beacon
DoubleClick – advertising
DoubleVerify – analytics
DoubleClick – http_cookie
[x+1] – widget
[x+1] – beacon
Advertising.com – advertising
Advertising.com – widget
PubMatic – widget
Demdex – beacon
PubMatic – advertising

Figure 4

In figure 3, [x+1] is calling many beacons and is being called by many beacons. By looking at which beacon is calling it maximum times or which beacon is called by [x+1] maximum time (figure 4), the graph can be traversed easily with greater insights, which will help in analyzing and making decisions about banning a creative or not.

5.2.5 Features of the Collusion graph

The collusion graph constructed has many features.

- We can adjust the size of the nodes according to the number of times it has been called and we can distinguish the nodes according to the vendor type, tracker type by using the UI facilities.
- We can see total how many times a vendor has been called.
- We can see what beacons call a particular beacon (a particular tracker type) and what beacons is this beacon calling.
- We can see how many times a node has been called by another node and vice versa.

Any other desired features/queries can be easily added in the graph database. We can also customize the UI by writing various filters. This will allow analyzing the vendor and its providers and determine the big players so as to be actionable.

6. Usability

The tool constructed can be used in many ways and can be helpful to a lot of people. This tool can be used to create the beacon hierarchy graph for a single creative as well as for thousands of creatives together.

- Earlier for navigating through the data, people would use pi chart, xml files, filters but these provided data visualization in just two dimensions. With this data visualization tool, detection and resolution of these beacons and their providers is easy to observe with the multi-dimensional view available.

- Whenever a blacklisted vendor is detected on a creative, the ad is completely banned and not served on the publisher's website but banning the ad results in the loss of revenue. In terms of business, we can't draw a sharp line between revenue and risk. Some analysis is always needed before taking an action i.e. banning an ad. So by using this tool, ad exchange will be able to analyze if it is okay to ban particular creative by looking at the vendors it is calling and deciding other parameters. Also for creating the black-list and white-list of vendors, this tool will help in deciding which vendors to focus attention on: instead of focusing on the small vendors, the attention can be given to the big players.

- This tool will be helpful for publishers to do their analysis about the beacon hierarchy, create a personalized list of blacklist and white-list of vendors and make a decision of allowing/denying those vendors to put creatives on website.

REFERENCES

[1]https://p10.secure.hostingprod.com/@spyblog.org.uk/ssl/spyblog/2011/08/28/labour-party-tracks-ken-livingstone-supporters-with-hidden-web-bugs-accessible-b.html

[2]http://www.zdnet.com/blog/btl/ad-network-at-center-of-third-flash-cookie-lawsuit/38346

[3]http://twiki.corp.yahoo.com/view/AdQuality/DataLeakagePRD

[4]http://www.adopsinsider.com/online-ad-measurement-tracking/a-primer-on-data-leakage-for-digital-publishers/

[5]http://twiki.corp.yahoo.com/view/AdQuality/DataLeakagePRD

[6]http://twiki.corp.yahoo.com/view/AdQuality/CTWebBug

[7] http://purplebox.ghostery.com/

[8] http://knowyourelements.com/

[9]http://www.ted.com/talks/gary_kovacs_tracking_the_trackers.html

[10] http://en.wikipedia.org/wiki/Web_bug