Data Transparency Lab
Sharing data safely for privacy analysis purposes: The Aircloak trial for releasing data from Floodwatch

Summary

Sharing personal data has always been risky and difficult, even when it is done for a good purpose such as searching and auditing systems for leakage of private information. Traditional strong anonymization techniques like K-anonymity often distort the raw data too much, so typically data is shared after simple de-identification (removing personally identifying data like names and account numbers). However, the risk of re-identification remains.

To overcome these limitations, DTL is exploring new techniques that allow researchers to share datasets quickly, easily, and broadly with little risk of re-identification. In particular, a trial is currently underway in cooperation with two companies: The Office for Creative Research (OCR) and Aircloak, a spin-off of the Max Planck Institute for Software Systems.

The specific question the trial was intended to answer is whether we can use Aircloak technology to release datasets gathered by OCR in a way that is useful for privacy analysts but secure for users in view of cross-correlation and other de-anonymization attacks. This document reports the results and findings of such a trial.

The main conclusions of this report are two-fold. First, the trial shows that the Aircloak system (cloak) can accurately answer a magnitude of queries of different types. A cloak is a database that instead of exposing directly all the data it contains it just responds to specific questions about the amount of entries in the database that satisfy some criteria. Within the constraints imposed by the need for anonymization, the trial concluded that the cloak could answer all of the questions posed with adequate accuracy.

Second, our experience showed that the analyst must always keep in mind that the answers given by the cloak are necessarily missing certain information, and that in some cases this information may substantially distort answers. In the questions that we posed, this was not a problem so long as the analyst understands how to interpret the answers given. In all cases, we could identify roughly how much distortion exists. The distortion comes in two forms:

1. Unique data objects that don't aggregate.
2. Outlier users.

An example of a unique data object in this study is an advertisement that is sent only to one or two users. The majority of ads in the data set are of this type. An example of an outlier user is a user that receives far more ads than other users. These two distortions had a strong effect on queries that look at numbers of ads. As an analyst that does not

keep in mind these distortions can reach wrong conclusions without realizing it, the report describes how he could discover and accommodate these distortions

Description of the Trial

Aircloak

Aircloak is a spin-off of the Max Planck Institute for Software Systems that has implemented an analytics system with a different approach to anonymization. Aircloak system allows the raw data to be available in a database (a "cloak"). However, thanks to the technology used by Aircloak, none can view the raw data once is inserted in the cloak. The only way to access the information in the cloak is via a query language that enforces anonymity.

Floodwatch

Floodwatch is a browser application made by the Office for Creative Research (https://floodwatch.o-c-r.org/). Floodwatch records the ads that users see on their browsers, and provides a visual collage of the ads back to the user. OCR wishes to make this data available to researchers so as to better understand the online advertising ecosystem.

OCR released an initial version of Floodwatch, and gathered advertising data on thousands of users. Among other things, this data includes: a non-identifying user ID, an identifier for the ad's image, the page on which the ad was seen, and the time when the ad was viewed.

Target

The target of this trial is determining if an analyst could use Aircloak to retrieve useful information from the Floodwatch dataset in a secure way for users. In order to do so, this dataset was loaded into both the Aircloak system (Cloaks), and a (non-anonymized) relational database system. Aircloak ran a number of basic queries over the data on both systems, and compared the results.

Analysis

Data Format

The floodwatch data is split over 14 distinct SQL tables. Of these, we focused on tables that contained data about users and their activities. We reformatted two of the floodwatch tables into two tables and loaded these into cloaks, as shown in Figure 1. An important floodwatch table attribute that was not included in the cloak data is the image itself. Strictly speaking, the image table does not have to be in cloaks: the image IDs output by cloak queries could then be accessed from a database outside the cloak.

We separately loaded the floodwatch data into a standard SQL database. This allows us to query both the cloaked and un-cloaked databases and compare the results.

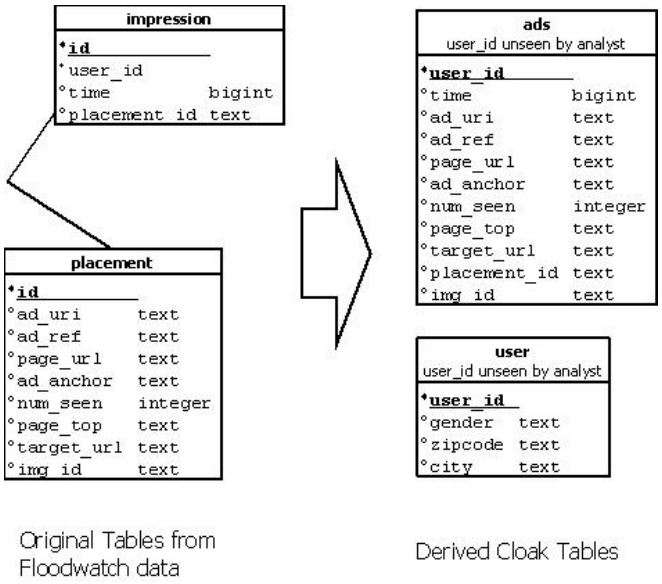

Figure 1: Floodwatch and Cloak Tables

Queries

In our simple analysis we queried for the following data (in three groups, focused on:

1a. Number of active users over time.

1b. Histogram and statistics for the number of ads received by each user (total ads, and distinct ads).

1c. Statistics for the number of distinct websites over which each user received ads.

2a. List of the ads seen by the most distinct users.

2b. List of the ads most frequently delivered.

3a. Number of ads delivered by each ad network.

If the cloak can properly answer these questions, a privacy analyst could check the type and amount of ads shown to different audiences based in geographical and ethnographical information (city, age, etc.).

The results are described in the following subsections.

General Workflow

Figure 2 shows the general workflow for analysts. All interactions with the cloak are via a REST API (HTTPS and JSON).

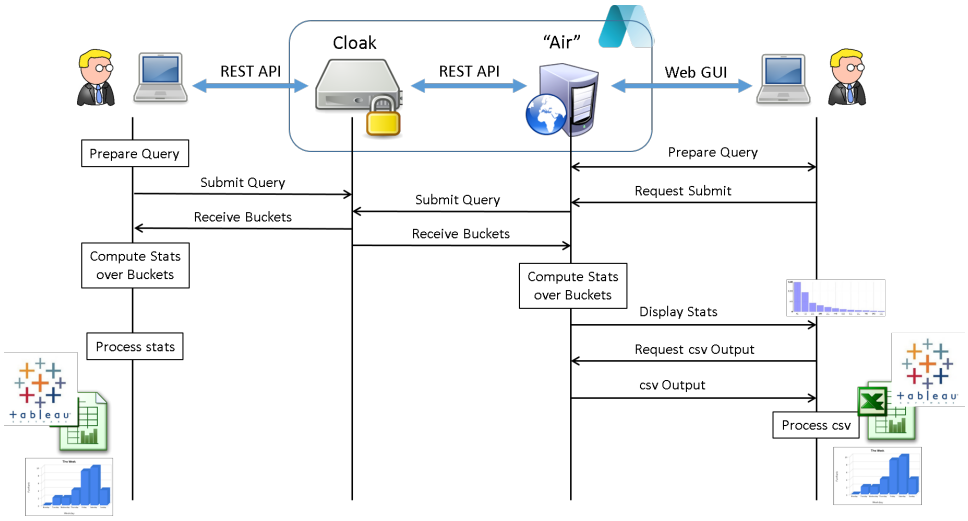

Figure 2: General Analyst workflow. Analyst may interact directly with cloaks, or via a Aircloak web service ("Air")

The analyst may interact directly with the cloak, for instance via analyst-written or Aircloak supplied scripts (Ruby). Alternatively, the analyst may interact with cloaks via an Aircloak-provided web service which we refer to as the "Air" component. This user-friendly interface lets the analyst compose queries, submit them, view the results which are stored in the Air, and export the results in csv format.

The output provided by cloaks is limited to "noisy counts of users", which we refer to as "buckets". Sometimes this output is of direct interest to the analyst, but often the analyst needs to post-process these buckets. One reason for post-processing is to smooth the data to reduce the effect of noise. Another is to convert a CDF (Cumulative Distribution Function) output into a histogram output. Another is to compute statistics such as average and median from the buckets.

The Air component can do much of this post-processing automatically for the analyst (see the "Compute Stats over Buckets" box of Figure 2). In so doing, the Air component displays a number of basic statistics as well as a simple histogram graph.

The analyst can also export the buckets from the Air in csv format. From here, the analyst can load the buckets into any number of business intelligence tools like Excel, Tableau, etc. to further process and display the data.

Most of the analytics done for this report was via the Air web GUI, additionally using Excel and a few scripts on the csv output.

1a. Number of active users over time

Figure 3 shows the daily counts for both the true and cloaked queries (there are roughly 8000 users in total). The time frame (late May to late October) represents the complete data set. Figure 3 shows that the counts are very similar. Figure 4 shows the absolute difference between the cloaked and true answers. Here we see that the noise is always within ±10.

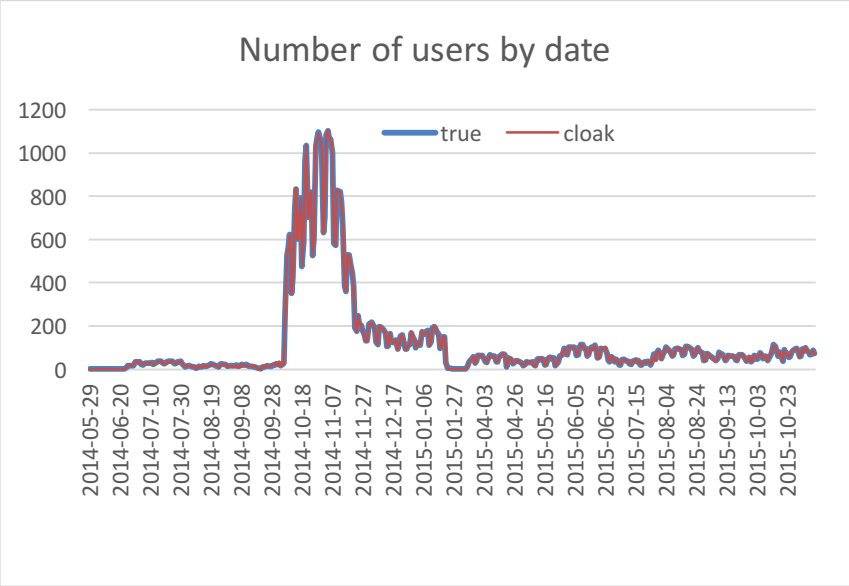

Figure 3: Daily number of users for both true and cloaked queries

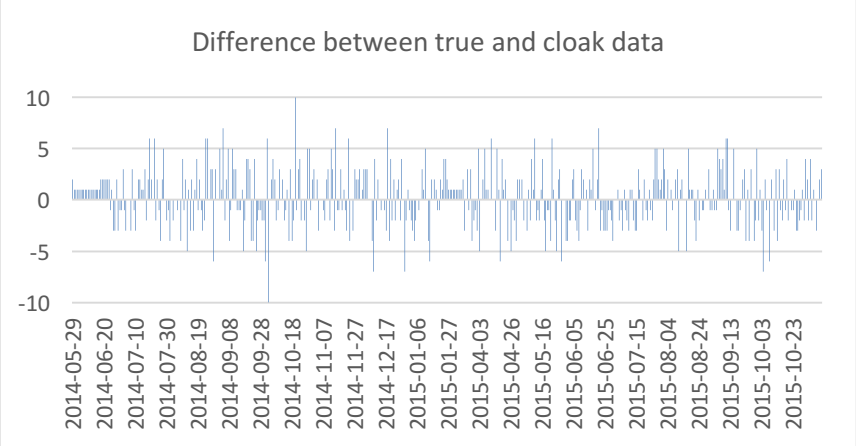

Figure 4: Difference between true and cloaked data for daily number of users

These graphs were produced using the .csv output of the air component, and then modifying the csv either within Excel, or with analyst-written scripts that run over the data.

The cloak "task" for computing the number of users is as follows:

```
for row in user_table("ads") do
  report_property("time-day", math.floor(row.time / (60*60*24)))
  report_property("time-week", math.floor(row.time / (60*60*24*7)))
end
```

This simply counts the user in daily and weekly "buckets" for each ad time that fits within the corresponding day or week. `report_property()` issues the count. Since a user can only be counted once per bucket, repeated calls to `report_property()` for a given day or week will not increase the count for the user.

Rather than SQL, we used a simple python script to compute the equivalent results for the true data, here read from the csv file. The script is as follows:

```
#!/usr/bin/env python

# You need to have the real data and users-over-time.json in the
# current working directory to make this work...

import sys
import json
import time

cloak_data = {}
def read_buckets(file):
  buckets = json.loads(open(file, "r").read())["results"][0]["buckets"]
  for bucket in buckets:
    if bucket["label"] == "time-day":
      cloak_data[int(bucket["value"])*60*60*24] = bucket["count"]

read_buckets("users-over-time.json")


for line in open("users-over-time-day-real.csv", "r").readlines():
  splitted = line.strip().split(",")
  date = int(splitted[0])
  ts = time.gmtime(date)
  date_string = time.strftime('%Y-%m-%d', ts)
  if cloak_data.has_key(date):
    print "%s,%s,%d" % (date_string, splitted[1], cloak_data[date])
  else:
    print "%s,%s,0" % (date_string, splitted[1])
```

1b. Histogram and statistics for the number of ads received by each user (total ads, and distinct ads).

Table 1 shows various statistics (Min, Max, Average, Median, and Std. Dev.) for the number of ads received for each user, including both total ads and distinct ads.

Table 1: Number of ads received by each user (the max range of the quantize_property() function was set for 25000 and 10000 for total ads and distinct ads respectively).

	Total Ads		Distinct Ads	
	True	Cloak	True	Cloak
Min	1	<0	1	<0
Max	284295	>5000	73668	>1960
Average	751.19	639.92	258.08	232.75
Median	220	225	104	102.5
Std. Dev.	4608.88	1033.27	1198.70	322.27

The cloak task that produced Table 1 is as follows:

```
local num = 0
local distinct = 0
local ads = {}

for row in user_table("ads") do
  if ads[row.img_id] == nil then
    ads[row.img_id] = true
    distinct = distinct + 1
  end
  num = num + 1
end

Aircloak.Distributions.quantize_property("ads received", num, {min=0,
max=25000, step=10})
Aircloak.Distributions.quantize_property("distinct ads received",
distinct, {min=0, max=10000, step=5})
```

Most of the work here is done by the Aircloak-supplied function `Aircloak.Distributions.quantize_property()`. This function builds a Cumulative Distribution Function (CDF) using the specified range and step size. The CDF buckets are reported by the cloak and the post-processed outside of the cloak. The cloak output is 1) smoothed, 2) converted into the Probability Density Function (PDF), and 3) used to compute the statistics in Table 1 (min, max, median, std. dev., and average). The output is also binned into no more than 20 bins, and displayed on hello.aircloak.com. The reason that `quantize_property()` builds a CDF and not a PDF is so that very small buckets produced by the cloak are not filtered out and suppressed by the cloak: the "100%" point in the CDF contains all the data.

A question is, "how should the parameters in the `quantize_property()` function be set? The `quantize_property()` function can be used to explore where the bulk of the data lies. By initially setting the max value in `quantize_property()` quite high, the analyst can learn where the high-end (or low end) of the bulk of the data lies from the max (or min) values returned by the function. This is because the statistics computation computes min and max values by a heuristic that detects when the CDF flattens out on each end.

We can see this from Table 1. Although the max for total ads received was set at 25000, the max reported by the cloak is 5000. This is not a true max, because individual outliers are invisible due to the cloak's anonymization. This is by design, since outliers can reveal information about individuals (i.e. a very rich person's salary). As we have access to the real data, we know that this is exactly the case as there is a value that goes beyond that limit (284K received ads). This max (and possibly a few other large values not reported by the cloak) pulls the average and standard deviation up.

The values of the same parameters calculated using directly the raw data via pure SQL are quite similar with the exception of the standard deviation because of the removal of the outliers by the cloak.

The analyst can produce a more reliable average and standard deviation by limiting the `quantize_property()` function to the range encompassing the bulk of the data. This

can be seen in Table 2. In this case, we ignore ad counts above 5000 and 1960 for total and distinct ads respectively. For the cloak task, we do this by setting the max parameter in the `quantize_property()` function to 5000 and 1960 respectively. For the SQL query on the true data, we use the following for distinct ads. Total ads was produced by remove the DISTINCT term, and increasing the WHERE count to 5000.

```
SELECT stddev(count) FROM
  (SELECT i.user_id AS user_id,
  COUNT(DISTINCT p.img_id) as count
  FROM placement AS p
  INNER JOIN impression AS i ON p.id = i.placement_id
  GROUP BY i.user_id)
AS foo
WHERE count <= 1960;
```

With this approach, the average and standard deviation are quite close to their true counterparts *for data within the specified range*. This means that the cloaked data can be quite accurate if the analyst is careful enough (and notes that outliers are not considered by the cloak).

Often an analyst might anyway "throw out" outliers, but in this case it is not so clear. 284K ads over 5 months represents about 4 ads per minute assuming 8 hours of browsing per day. That is extreme, but feasible. One might well want to take closer look at the outlier user to see what caused it, obviously, with the cloak this is not possible, however Floodwatch believes the outlier(s) might be the result of system testing that they did early on, so removing them from statistical purposes seems the right thing to do.

Table 2: Number of ads received by each user (the max range of the quantize_property() function was set for 5000 and 1960 for total ads and distinct ads respectively). The true data also limited to 5000 and 1960 total and distinct ads respectively.

	Total Ads		Distinct Ads	
	True	Cloak	True	Cloak
Min	1	<0	1	<0
Max	4936	>4890	1959	>1320
Average	554.77	557.17	219.29	214.41
Median	209	215	102	110
Std. Dev.	808.47	811.24	289.30	266.59

Figure 5 shows a histogram of the total number of ads received by each user. This histogram, as well as the numbers in Table 1, are all generated by the web service from the output of the `quantize_property()` function (hello.aircloak.com).

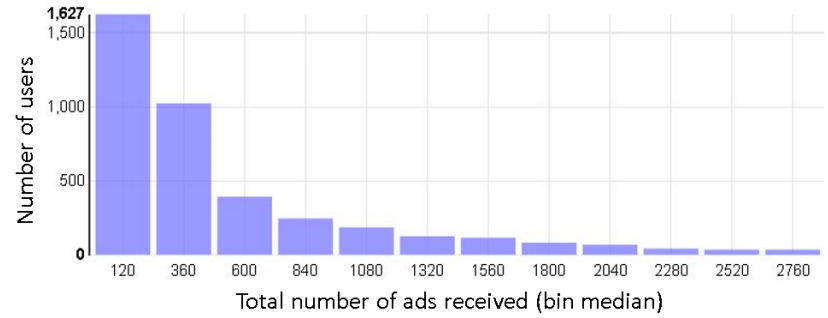

Figure 5: Histogram total ads received by each user (cloak)

Figure 6 shows a histogram of the number of distinct ads received by each user.

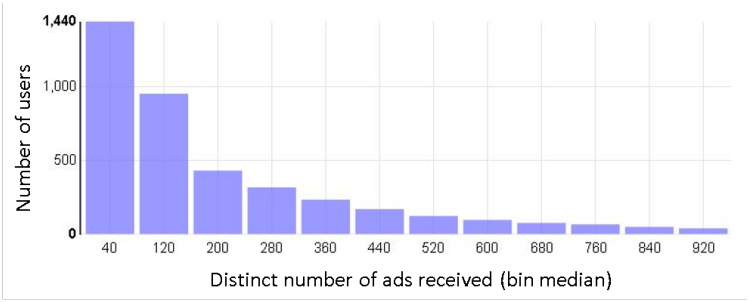

Figure 6: Histogram of distinct ads received by each user (cloak)

As an exercise, we looked more closely at the very low end of number of ads received. Figure 7 shows the number of users that received between 1 and 17 ads in total. The "true" line is the true count from the un-cloaked database. The "cloak-raw" line is the direct output from the cloak (as PDF histogram). The "cloak-smooth" line is the smoothed output derived from the cloak's CDF output as described above. Comparing the true and cloak-raw lines, we see that the noisy data tracks the true data pretty well. Even at counts of less than 100, the cloak provides meaningful results. The smoothed line is less accurate, but this would be the case whether or not the true or noisy data were being smoothed.

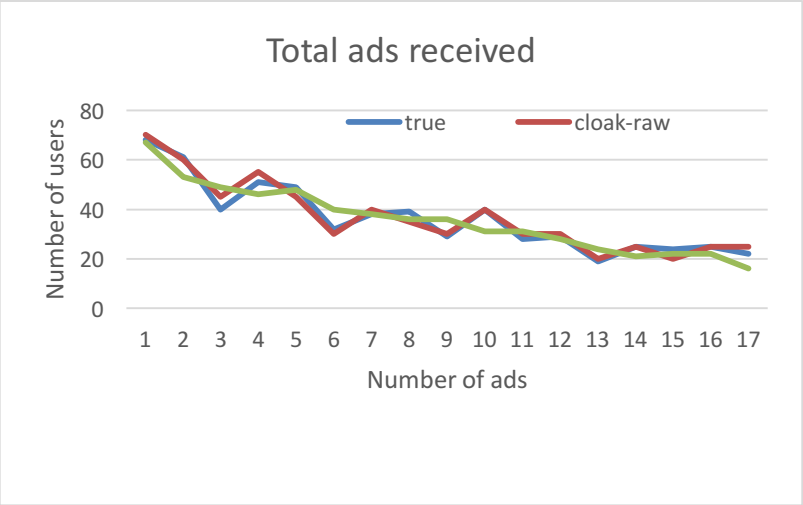

Figure 7: Number or users receiving between 1 and 17 total ads, for 1) the raw data, 2) the direct output from the cloak (cloak-raw), and 3) the smoothed cloak output as displayed on the web GUI (cloak-smooth)

1c. Statistics for the number of distinct websites over which each user received ads

In this query, we look at the number of distinct websites that each user accesses by looking at the `page_top` column. This column is described by Floodwatch as "url of top level page, could be different from page url if this was detected in an iframe".

Table 3 gives the statistics for the true data and the cloaked data as derived from the noisy cloak output. In this, case, there is a single outlier in the data, which received ads over 11,588 distinct websites. All other entries were below 1000.

Table 3: Statistics for number of distinct websites over which each user received ads

	True	Cloak
Min	0	<0
Max	11588	>5000
Average	66.25	63.74
Median	26	27.5
Std. Dev.	200.17	84.24

Figure 8 shows the histogram of distinct websites per user in the range of 0 to 200 websites. Our post-processing of the cloak-output CDF estimates that this data covers roughly 90% of the total ads recorded.

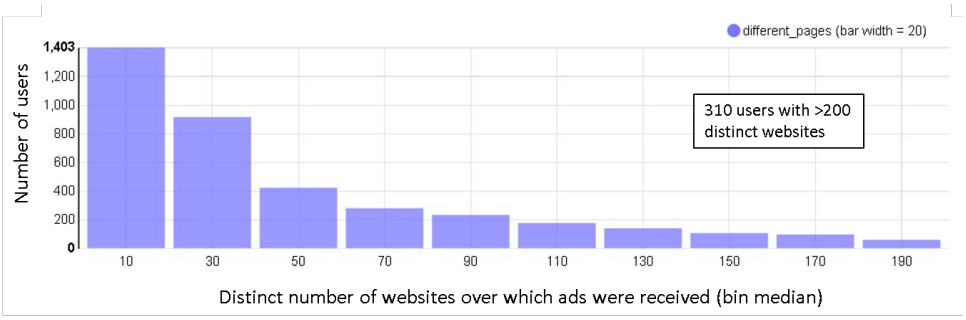

Figure 8: Distinct number of websites over which ads were received, derived from cloak CDF and smoothed

2a. List of ads seen by the most distinct users

Table 4 lists the 20 ads (image id) seen by the most users, as computed from the real data and from the cloak. Only the first few of 32 hex digits are shown.

Table 4: List of ads seen by the most distinct users, for real and cloak queries. "real count" and "cloak count" refer to the number of distinct users for real and cloak queries respectively. Yellow image ID does not appear in the cloak list.

real img	real count	cloak img	cloak count
b342af56fa	1294	b342af56fa	1295
d87085aa44	1136	d87085aa44	1130
cebb49a73e	456	cebb49a73e	455
96d6fb808c	367	96d6fb808c	370
8cd387817e	358	8cd387817e	360
50036beae6	351	50036beae6	355
69a587a8a5	349	69a587a8a5	350
66203d2144	343	66203d2144	345
6f328a96cd	337	6f328a96cd	335
f579fa1a42	335	f579fa1a42	330
aa109200ea	303	aa109200ea	305
d1833eff00	294	d1833eff00	295
aec6e9058a	292	aec6e9058a	290
f7d6139a1b	282	17d33b5198	285
17d33b5198	282	4a8770c167	280
4a8770c167	281	a7b9ad9c9c	280
6cd266167a	280	c26f0c7bab	280
c26f0c7bab	279	f7d6139a1b	280
a7b9ad9c9c	278	1f52f6a073	275

The first 13 entries list the same image IDs, with counts differing slightly due to noise. 19 of the 20 entries are present in both lists: `6cd2…` is missing from the cloak list (yellow highlighted), and `1f52…` is missing from the real list. The reason is that the cloak gave `6cd2…` a count of 275 whereas the real count was 280, thus pushing it off of the top-20 list.

98 of the top 100 appear in both lists.

2b. List of the ads most frequently delivered

Table 5 shows the top 20 most frequently delivered ads (including repeats to the same user) for real and cloaked queries. Here, 8 of the real entries do not appear in the cloak list (yellow highlighted). Overall, the cloak under-estimates the frequency of ads. Of the top 100 for both real and cloak, there are 17 mismatches (ads that appear in one list but not the other) for each list.

Table 5: List of top 20 most frequently delivered ads, for real and cloak queries. "real count" and "cloak count" refer to the number of distinct users for real and cloak queries respectively. Yellow image IDs are those that appear in the real list only

real img_id	real count	cloak img_id	cloak count
d87085aa440	10662	50036beae629	9715
50036beae62	10100	d87085aa4401	9080
b342af56fa86	7528	b342af56fa86e	6655
f579fa1a42a6	6808	f579fa1a42a6f	6480
4a8770c167b	6148	4a8770c167ba	5970
e0c0c2d021b1	4681	e0c0c2d021b1	4520
9edb404720a	4502	9edb404720a3	4085
a7b9ad9c9d3	4385	f71563b678d0	3220
f71563b678d0	4027	a7b9ad9c9d3e	2925
66203d2144b	3791	6f328a96cdd6	2850
e8f0bafc8a2e	3563	66203d2144bf	2780
c0925cf6d5bd	3408	cebb49a73e68	2535
bdf0ed4496fe	3309	39698b83b1fe	2195
e1c700d57ed1	3236	7a94d930a5c7	2150
d4dc83bbca7e	3146	27e5261a2218	2120
2a8c7588471	3018	249c6b785d6e	2100
6f328a96cdd6	2992	6df68a00e507	2100
be111e8c4ea4	2952	fe6fd1720772	2100
cebb49a73e68	2749	157ac2cb3854	2090
7b10057e2c51	2708	adbd0a150964	2090

The inaccuracy occurs here because many ads appear frequently at only a small number of users. As an example, we know from the real data that the 11th-ranked ad (e8f0ba…) appears 3563 times for one user only. Because this ad is unique to one user, by design the cloak will never reveal it. In short, the yellow-highlighted ads in Table 5 are all potential privacy violations.

Figure 9 shows the actual images of the top 11 ads as reported by the real data. One of these was suppressed by the cloak. This is an image of a real person, so it has been pixelated as it represents a potential privacy violation.

Figure 9: Top 11 most frequently delivered ad images. One of these was suppressed by Aircloak, because it appeared at only one user so it has been pixelated

To explore this further, we generated a query that generates a CDF of the maximum number of repeat ads that each user has seen. The results are shown in Figure 10. From this we see that, while most users didn't see many repeat ads, many users saw 10s of repeats, and a small number of users saw hundreds of repeats, a few more than 2000.

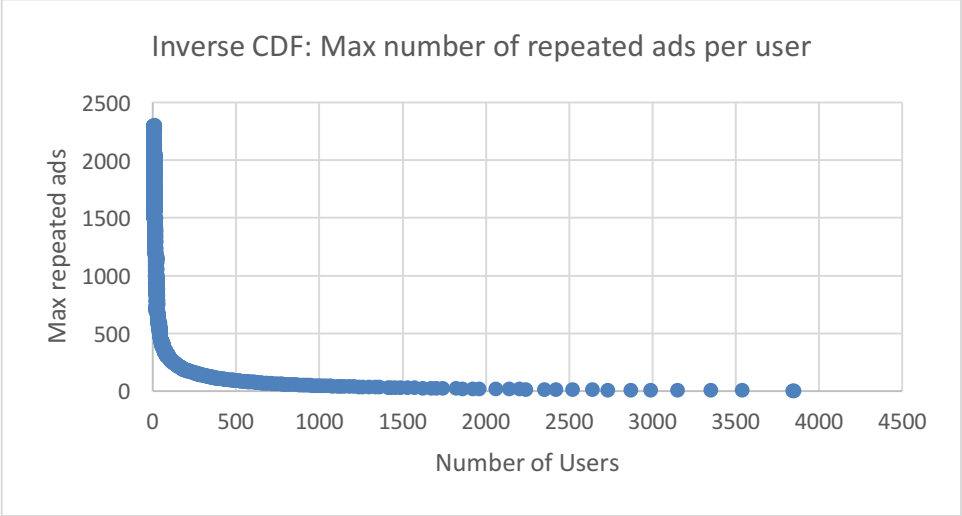

Figure 10: CDF of the maximum number of repeated ads seen by users (cloak)

Figure 11 focuses on the users that received the most repeated ads (the left part of Figure 10). Here we see that there were a handful of users that received many hundreds of ads.

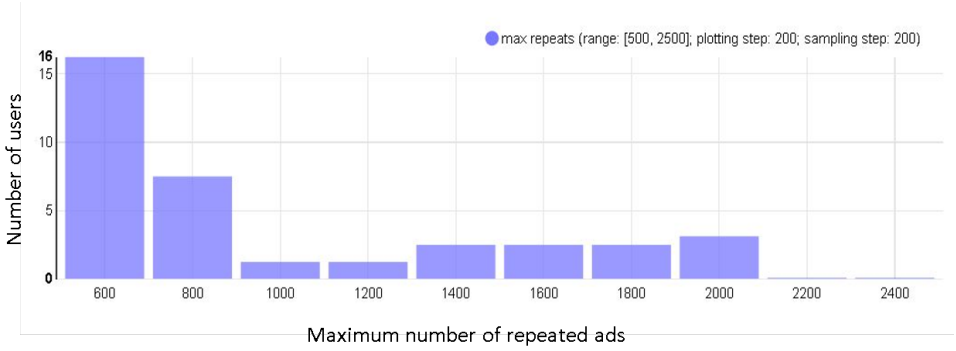

Figure 11: Histogram focusing on users that recevied the most repeat ads (cloak)

This suggests to the analyst that the cause of the missing ads are ads that are repeated very frequently for one or very few users. Specifically, what is missing are users that are outliers in-so-far-as the number of times they have received a given ad. This effect manifests itself in two ways. First, if an ad appears at only 2 or 3 or fewer users, then the ad is never exposed by the cloak at all. Second, if more than this many users see the ad a few times, but one user sees the ad 100s of times, then the cloak will expose the ad, but under-count the number of times the ad was delivered (because the one outlier user is suppressed).

To repeat, this is by design. Any anonymizing system must necessarily hide outlier users.

3a. Number of ads delivered by each ad network

We used the column `ad_anchor` to determine the names of ad networks used to deliver ads. This column is described by Floodwatch as being "unresolved outgoing url of the ad." After some initial queries, we learned that the `ad_anchor` data is pretty dirty, for two reasons. First, many of the entries are not ad networks, but rather the URL of a website (i.e. www.nytimes.com or www.cnn.com)[1]. Second, some of the entries are not proper URLs (i.e. "https:" alone). Rather than go through the effort of cleaning up the data, we chose to focus on just two well-known ad networks and gather data about those networks.

	googleadservices.com		doubleclick.net	
	True	Cloak	True	Cloak
Min	1	<0	1	<0
Max	19155	>446	95192	>1430
Average	79.27	61.93	321.68	217.65
Median	24	24.5	94	81.5
Std. Dev.	480.95	129.98	1801.1	315

Again, the divergences on the results are because of the outliers so this could be adjusted by removing them as done in section 1b.

Conclusions

We have conducted a trial to test a new approach to let researchers extract insights from datasets that contain Personal Information. The datasets are injected in a special type of database (cloaked database) that doesn't let anyone access directly the information but just via a query language (so that the Personal Information is protected).

We have shown that even if the absence of direct access to the data, privacy analysts can extract valuable statistical conclusions and insights. In order to do so, the analyst must be careful and understand that because of the cloaked nature of the database, additional considerations should be taken when defining the queries. We have compared the conclusions achieved via interacting with the cloaked database with the ones that could have been achieved by accessing directly the data and we have concluded that there are no significant differences between them.

We have also shown that the anonymization techniques used by the cloak (hide items with a small count, add noise, etc.) block the return of information that could be used to re-identify end-users (e.g. outliers).

[1] It should be noted that those websites are themselves also ad networks, and so these URLs are legitimate.