The Anonymity Engine, Minimizing Quasi-Identifiers to Strengthen k-Anonymity

by

ERIC MATTHEW LOBATO

B.S., University of Colorado, 2015

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirement for the degree of

Master of Science

Interdisciplinary Telecommunications Department

2017

This thesis entitled:
The Anonymity Engine, Minimizing

Quasi-Identifiers to Strengthen k-Anonymity

written by Eric Matthew Lobato
has been approved for the
Interdisciplinary Telecommunications Department

_____

(Joe McManus)

_____

(David Reed)

_____

(Levi Perigo)

Date_____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

IRB protocol # _____

IACUC protocol # _____

Lobato, Eric Matthew (M.S, ITP)

The Anonymity Engine, Minimizing Quasi-Identifiers to Strengthen k-Anonymity

Thesis directed by Scholar in Residence Joe McManus

The k-anonymity model has become a standard for anonymizing data. However, almost all applications of k-anonymity are used to anonymize large data sets of personally identifiable information owned by a trusted third party before being given to analysists To further research in this this area, this study created a tool called the Anonymity Engine. This tool was built as a web browser plugin that analyzes headers on all web traffic exiting the system and builds a database of relevant quasi-identifier. Users are notified in real time if a data packet would compromise their identity and give the option to not send the data. This tool has also been used to generate data that shows that modifying data before implementing k-anonymity can impact the results. These modified results show that it can make some users more anonymous while reducing the level of privacy for other users depending on the traffic.

CONTENTS

Table

Figure

I. Introduction and Research Question

Imagine a user who logs onto a computer and visits the following websites:

- www.denverpost.com to check the news
- www.apple.com/support/products to look at warranty plans for their computer
- forecast.weather.gov to check the weather
- www.ebay.com to bid on a tie.

This may appear to be a normal browsing session, but to an advanced user with access to traffic sniffers could use this session to ascertain several identifying traits about the other user. First, that the person is from Colorado, identified from the zip code entered on the weather site and by the news site of choice. Second, that the person has recently broken their apple product, and finally that the person is shopping for neckwear. Perhaps the advanced user knows a friend in real life who matches these descriptions. If this was the case, they could easily learn more facts about their friend by following the traffic of this specific user and potentially learning things that their friend would wish to keep private.

As this example shows, maintaining personal privacy when using the internet has become one of the most important issues of the modern day. Unfortunately, nontechnical users struggle to understand how their browsing patterns could link information about themselves to the traffic they generate. Meanwhile, technical users might not always be convinced that they are truly anonymous when using privacy software such as VPNs, proxies or the Tor browser. This research aims to create a new tool that will give users a clear idea of what their web traffic says about them and whether or not they are anonymous over time. This will fill a niche in privacy tools that currently remains open as few privacy tools describe what actual network packets say about a user.

The tool that this research will produce is based on the existing model for privacy known as k-anonymity [1]. The k-anonymity model determines anonymity based on the idea that knowing a minimal amount of identifiers can link an individual in a data set. For a data set to meet k-anonymity it must meet the definition:

$$Let\ RT(A_i, \dots A_n)\ be\ a\ Table\ and\ QI_{RT}\ be\ the\ quasi-identifier\ associated\ with\ it.\ RT\ is\ said\ to\ satisfy\ k-anonymity\ if\ and\ only\ if\ each\ sequence\ of\ values\ in\ RT[QI_{RT}]appears\ with\ at\ least\ k\ occurances\ in\ RT[QI_{RT}]$$

Definition (1)

This definition leaves much to be desired for those who are not mathematicians. Thankfully, there are many examples which clearly show the intention of what k-anonymity is attempting to illustrate.

The classic example, which was first published with the definition of k-anonymity, was to think of a scenario where you wake up one morning to find an ambulance taking your neighbor, Bob, to a hospital. Let's say, for example, that you are the nosey type, and you investigate what happened by entering Bob's house. Conveniently, you find a sheet of hospital records on his counter that looks like Table I.

Table 1 Hospital Records

	ZIP Code	Age	Disease
1	80303	24	Flu
2	80211	25	Asthma
3	80303	28	Flu
4	80103	30	Heart Disease
5	80121	32	Cancer
6	80305	36	Cancer
7	80111	44	Flu
8	80112	57	Heart Disease
9	80111	68	Stroke

While the hospital data may look like it is anonymized due to the fact that it contains no names or obvious identifiers, you as Bob's neighbor can leverage some knowledge that you have about him to infer which patient is him. Let's say that you know that Bob is 30 years old because you attended his party last week, and you know his zip code because it is the same as yours. This pieces of information are Bob's quasi-identifiers; individually they mean nothing, but taken together they identify Bob. With just this background knowledge you can find that there is only one match on the records that fits this description and you realize that a heart attack caused by chest pains is what caused your dear neighbor to be sent to the hospital.

Now let's apply the theory of k-anonymity to the data set. We want to make sure that there are at least a k amount of rows that are the same. TABLE II does this for us by instating a k value of 3.

Table 2 Hospital Records with k of 3

	ZIP Code	Age	Disease
1	80***	2*	Flu
2	80***	2*	Asthma
3	80***	2*	Flu
4	80***	3*	Heart Disease
5	80***	3*	Cancer
6	80***	3*	Cancer
7	8011*	≥40	Flu
8	8011*	≥40	Heart Disease
9	8011*	≥40	Stroke

This means that, at a minimum, data has been anonymized so that there are 3 rows with matching data. Now, if you had found this sheet in Bob's house, you would be able to narrow down your choices but you would not be able to say for certain whether Bob was sent to the hospital due to a heart disease or cancer. This implementation of k-anonymity successfully prevented Bob's privacy from being breached.

When data such as this is used in the real world a trusted third party, such as the hospital record keepers, sanitize the data before it is shared with any relevant parties such as researchers. This system works well as long as the third party who is scrubbing the data is trustworthy themselves. One of the big differences with digital data is that unless the user is specifically trying to anonymize their data, unwanted third parties could potentially perform these linking attacks based on a user's browsing history. What's more is that as more and more technical users take steps to anonymize their traffic, it also becomes easier to link the data from users who are not taking steps to protect their data because they stand out in the crowd.

Returning to the original example of a hacker observing a person's browsing history we find that a similar tactic can be employed. Had the user chosen to view a weather website that was encrypted, and if they had instead gone to a more general news site as opposed to the Denver Post, the attacker would not have learned that the person was from Colorado and could not have guessed that he knew this user in real life.

This research is intending to find out what happens if the user chooses to take up the job of that trusted third party and only send data that would not violate a k-anonymity table. This would not be possible in the real world, after all, you wouldn't tell your doctor that age is greater than 20 if he asked you how old you were. Yet, in the computer world it is possible to mask yourself and still have a quality experience. Getting weather for your general area instead of through your GPS, or only using

search engines that encrypt your search would drastically improve your level of anonymity online. The problem is that many users do not know what they are sharing online when they browse the web and this research aims to fix that.

Formally, then, the question that this research seeks to answer is, “Does obfuscating quasi-identifiers before being added to a data set impact the anonymity of users in that data set in a significant or meaningful manner?”

To solve this problem, it has been broken down into two sub problems which, when solved, will produce an answer to this research question. The two sub problems are as follows, as well as the design requirements needed to guarantee a satisfactory answer:

a) Build a program that captures internet traffic and stores sensitive information.
   - Quasi-identifiers will be categorized by severity
   - Traffic will be captured at the application layer
   - Information will be securely stored
   - Users can easily view and understand their results

b) Testing the engine and analyzing results
   - Generate browsing histories which create data sets that are applicable to actual users.
   - Create two sets of data, one where users do not mask information, and one where they do.
   - Combine results and apply k-anonymity to study results.

.

II. Literature Review

One of the most heavily researched areas for privacy and anonymity is in the big data sector. Large data sets such as census data, medical records or voter registration databases need to be able to both provide enough information to be relevant to researchers, but obfuscated enough to not compromise the security of any individual whose information lies within the database. In 2002, Sweeny introduced the world to k-anonymity model, which remains the cornerstone for privacy research to today. The concept is simple, in any given schema the individual records can be classified as either explicit identifiers-such as name or social security number, quasi-identifiers-that can identify a person when combined or finally, sensitive-containing private information. The research shows how knowing a select few of the quasi-identifiers of a record in the database can enable an attacker to learn new private information related to that record. The classical example is how the author was able to link an entry in a census database to the Massachusetts Governor because of the six people who shared his birthdate, three were men and only one was in his zip code. k-Anonymity then is the end result of taking a schema and making each record indistinguishable with at least k-1 other records with respect to the quasi identifier.

After k-Anonymity was revealed, many other researchers sought to strengthen the model. It was shown that k-Anonymity is vulnerable to two types of attacks known as homogeneity attacks and background attacks. A homogeneity attack occurs when new information is revealed through the grouping of data. For example, an attacker could realize a patient has a heart condition if he were to find his target in a schema that grouped patients with heart disease together. Background attacks occur from creating deductions based on outside knowledge, such as an attacker ruling out entries on a schema because he knows that a disease only affects a certain ethnicity. Both of these attacks were sought to be mitigated by the introduction of *l*-diversity by Machanavajjhala in 2007 [2].

A data set is said to have l-diversity if there are at least l "well-represented" values for every equivalence class in the data set. The idea of a well-represented value is one that explains the information but may not convey helpful information to a third party. An example of this would be using proper medical terminology is patient information as opposed to a general "stomach problem" value. In response to Machanavajjhala, Li explains in his research how l-diversity can be useless in data sets where records could be limited to a positive or negative value. Furthermore, l-diversity would not prevent attribute disclosure in skewed situations where a participant would only want anonymity if they held a certain value (such as positive for cancer). Li's research built on l-diversity to create t-Closeness, a method for arranging sensitive data within data sets so that assumptions could not be made [3].

At this point, one of the obvious problems with k-anonymity is that the results are only correct and anonymous at the time of their release. One of the easiest ways to further strengthen the privacy offered by using a k-anonymity model is to create an algorithm that would reduce the risk of privacy disclosure in re-publication of data sets. This would effectively allow for a rolling privacy model as data sets grow throughout time. Xiao and Tao implemented two new approaches to mitigate this threat in 2007, calling the principles M-invariance and counterfeit generalization. The idea was to add some false records to a data set which would not compromise the readability for a valid user, but would render it impossible for an attacker to perform a background attack through the linking of tuples. [11]

Researchers continued to find new ways to attack and strengthen k-anonymity. In 2009, Sun introduced two new approaches to strengthen a popular implementation of the model, called p-sensitive k-anonymity. P-sensitivity is simple, in that its model simply requires that there be at least p unique values for any sensitive tuple in a schema. Sun and his team found this to still fall victim to several types of attack. Their new implementations were p+-sensitive k-anonymity and (p, α)-sensitive k-anonymity. [15] The p+-sensitive model works by categorizing groups of k-anonymous tuples to further obfuscate revealing data. The (p, α)-sensitive model works by defining ordinal weights for each category and then ensuring that categories that meet a certain weight never have more than α appearances within the schema. This is a very promising method of using k-anonymity in this research since the correlation variable that will be implemented in the anonymity engine is very similar to the α weight in the (p, α)-sensitive k-anonymity model.

In 2012 k-anonymity was strengthened again with Vel Kumar's model for l-diversity on k-anonymity with external databases [4]. This new model functions the same as the l-diversity model but splits the data across multiple databases in order to minimize the ability for an attacker to link information about an entry. It is interesting to note that this model ignores Li's t-closeness model and could provide an area for further research.

These refined k-anonymity models have found uses in other areas of privacy research particularly in the area of representing social networks through graphs. Xiangmin shows that the k-anonymity model can be applied to social network graphs to provide a stronger amount of privacy than simple identity masking [5]. Zhou continued the trend by showing that any social network graph could be fitted with k-automorphism by adding and deleting edge vertices to prevent background attacks [6]. Unfortunately, k-automorphism graphs are vulnerable to what are referred to as structural attacks. Cheng remedied these vulnerabilities with an algorithm called k-isomorphism that splits graphs into smaller subgraphs that are k-anonymous [7]. More recently, Liu strengthened the k-anonymity in social network graphs by assigning weights to individual nodes in order to represent key relationships for users [8].

However, k-anonymity doesn't just extend into the social media world. In fact, k-anonymity is employed in location based software through several methods such as location perturbation and obfuscation. However, these methods assume that the third party server parsing the location data is trusted, and that the server won't become targeted by an attacker. A proposed solution for this problem is to use dummy locations that generate realistic user data. Unfortunately, this solution is still lacking, as the data can fall victim to background attacks with obviously fake dummy data. Niu created a new algorithm that takes into account additional user information and an entropy metric to pick better dummies to pipe data through [9]. Niu then sought to strengthen the dummies themselves by proposing two new dummy location generation schemes: v-Circle and v-Grid. V-circle functions by generating temporary locations in a circle before blurring them into locations to be used by their privacy metric. V-Grid generates temporary locations based on a virtual grid before blurring them into proper positions that meet k-anonymity specifications [10].

One of the problems associated with using the k-anonymity model is finding the quasi-identifiers to anonymize within a given data set. Throughout the last decade several algorithms have been suggested and implemented which aim to ease the burden of picking out identifiers. In 2005, Bayardo proved that optimizing a k-anonymity algorithm would be NP-hard and made the problem easier to solve by reducing the possible combinations of tuples in a data set. Through the use of his algorithm, provably optimal k-anonymizations can be obtained for real census data under two representative cost metrics, and that the algorithm will complete within a time frame of a few minutes. [12]

Other researchers have introduced methods for finding quasi-identifiers as well. In 2016 Omer and Mohamad created two new algorithms to find the identifiers in a data set. These algorithms are called Selective and Decompose and while they do in fact limit the number of identifiers in a tuple set, the authors admit that their algorithms are still vulnerable to linking attacks. [13]

K-anonymity has also been extended to the cloud. Zhang produced an algorithm in 2012 for using quasi-identifiers as an index across cloud based services. This would allow for faster and simultaneously anonymous for data sets that reside across multiple nodes in the cloud. This new method is different than the current model, which is to re-anonymize the entire data set when a new user is entered or a privacy policy is updated. These methods scale poorly in the cloud and the new algorithm was shown to significantly improve performance in these sizable data sets [14]

K-anonymity has been used in a wide variety of applications in the privacy field, however all of these models propose methods of anonymizing the data after it has been received by a trusted third party. This research furthers the current state of the art by seeking to create a privacy model in which an

individual user would only release data that they know does not reduce their levels of privacy unless specifically authorized. This allows for a unique situation where a user can control what they are releasing on the open internet. Further research can also be done to implement the advances discussed in this review after the data has been collected. This would allow for significant new areas of study as every advancement to k-anonymity could potentially differ if applied on data that was modified by a user.

III. Research Design and Methodology

A. The Anonymity Engine

The program, referred to as an Anonymity Engine, functions by capturing web traffic. It runs on all major operating systems (Windows, OSX and Linux). The Engine was created as a Chrome browser extension in order to capture traffic at the software level as opposed to the network level. The browser extension has been built from the ground up to leverage the Chrome API as much as possible and there are no plans to port the extension to Firefox. The plugin intercepts all headers that are being sent and received from Chrome and stores potential quasi-identifiers in a Sqlite database that resides in-memory. The extension will be able to function in incognito mode and in offline mode. The identifiers that the software will be storing will be categorized as:

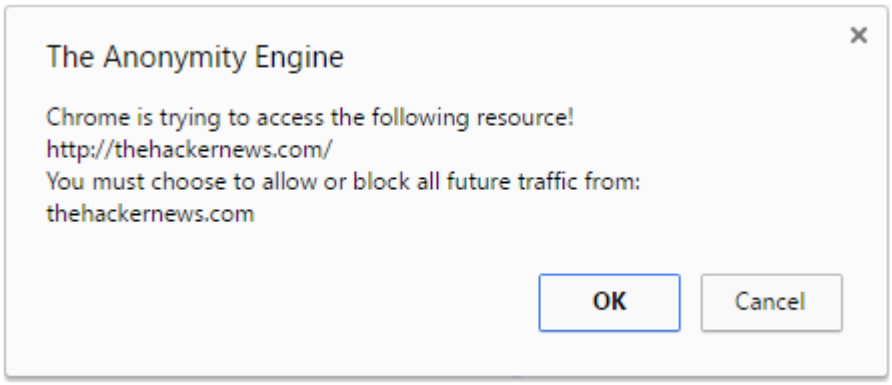
- Destination IP addresses
- Browser Fingerprint Information (user-agent)
- Operating systems
- Search Queries
- Sensitive Information (as POST data)
- POST forms

The Anonymity Engine has two display modes. The first is Basic mode, which aims to condense the information gathered as succinctly as possible. All URLS and POST data shown in this view were sent using HTTP. The decision to cull the data to only plaintext is further explained in the Design section. When a user clicks on the extension button in their browser bar, a popup webpage with information about what identifiers they have sent will be shown. The first section on the display page is information interpreted from the users-agent header string. The user-agent string contains information such as operating system, browser version and languages accepted. Websites utilize this information to ensure that the data they display will render correctly on a user's screen. However, this information consists entirely of quasi-identifiers, since it is quite possible for a specific operating system or browser version to isolate a user in a group. The version of the Anonymity Engine used in this research does not block or modify the headers. The first reason for this is to reduce the amount of variables in the analysis. The second reason is that some web pages will not display correctly if this information is omitted. Finally, the target audience for this application are users who would not know what it means to modify this data.

The next section of the display popup shows a table titled "URLS Visited(plaintext,HTTP)." This table contains all unsafe URLS logged as well as a count of how many times they've been accessed. As a user navigates around the internet, all the URLS they visit will be logged internally by the application, however, if a user visits a page which does not use HTTPS, the application intervenes. For every request

made to a HTTP page an alert will be displayed to the user. An example of this popup can be seen in Figure X.

Figure 1 Anonymity Engine GET Blocking Example

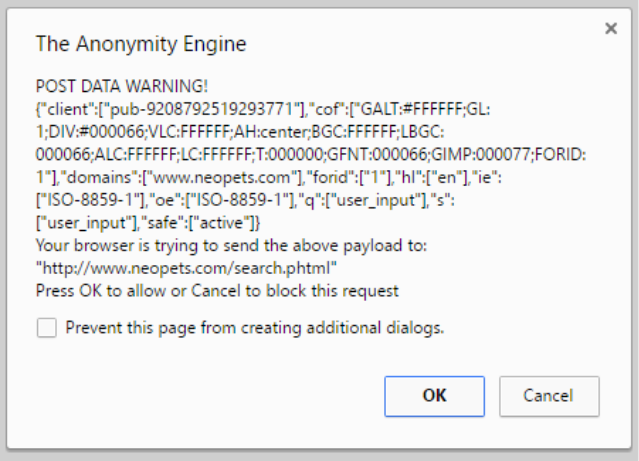


This alert displays the resource that Chrome is trying to access as well as the domain of URLS which will remain blocked or allowed with the user's permission. If the user clicks "OK", the root URL address is whitelisted and the user will not see a popup display the next time they visit this site. The Engine will still be logging that the user went to an insecure site and will display that information within the popup window. Alternatively, if a user clicks "cancel" the root URL is blacklisted and no traffic will be sent there. A user can view which URLS they have blacklisted or whitelisted under the advanced view.

This information, as well as everything else the application collects, can be deleted by pressing the "delete database button" in the popup window. Furthermore, any information that is stored in the extension is wiped on exiting chrome. This is done both to ensure user confidentiality as well as an effort to keep the memory size of the application reasonable.

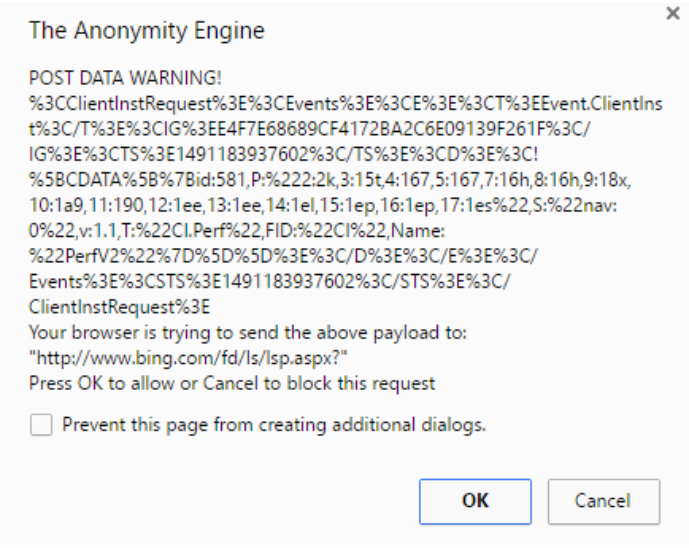
The next section on the basic view page is a table titled "Data you've Sent(plaintext,HTTP)." This table is populated when a chrome makes a POST request to a website that uses HTTP. This information can be categorized as sensitive since many POST requests are sent because a user has entered information onto a web page. When this event occurs a popup alert window will display and the user will be asked if they wish to send that data to the server. An example of this can be seen in table X. In this example, a user typed "user_input" into a search box on a website and was asked if they wished to submit it, since they can now view the other data they were unknowingly sending as well.

Figure 2 Anonymity Engine POST Blocking Example(form data)



In this situation, selecting “OK” or “Cancel” will only allow or block the single request. If there are multiple POST requests each one must be clicked through manually. This design choice was implemented because a user will only ever have several POST requests to allow or block as opposed to the potentially hundreds of GET requests on a content heavy page. If the POST data is encoded in UTF-8, Chrome can natively parse it and display it in the popup window as before. However, if the data is not in a UTF-8 format, the raw binary is attempted to be converted and displayed. An example of this can be seen in Table X. There is also a chance that there is no payload. In this situation, the data is simply listed as “empty payload.”

Figure 3 Anonymity Engine POST Blocking Example(binary data)



The next table is titled “Things you’ve searched for.” This table is a condensed version of the URL table on the advanced mode which only shows entries that were sent using HTTP and include either the keyword “search” or “q=”. This table catches any query using the common method of delimitating

searches on the internet. However, any URLs that are missed will still be visible in the URL table as mentioned before. This table is simply meant as a quality of life enhancement and its contents should supplement data found in the other tables.

The basic view also has a button titled “download database.” This button will allow the user to download a local copy of their data. The data is stored as a normal sqlite database and the schema for the database can be seen in Table X. The database consists of five tables. The first table is DIP, for destination IP addresses, this table is populated with the IPv4 addresses of any web resource that is accessed during a browsing session. This table is not shown in any way in the basic mode and is shown in the advanced mode to provide additional insight to technical users. The next table, URL, contains the full location of any request sent by the browser. This table can grow to be large if a user is accessing sites with many images or scripts because the address of all of those assets will be saved here. The next table is the Agent table. This table contains the user’s agent-header string along with a count of how many times it has been sent. If a user was to spoof or modify their header, it would also log the new header. Finally, the POST table contains all form data or decoded raw bytes that a user has posted along with the address of the website that the payload was sent to.

Table 3 Database Schema

name	sql
DIP	CREATE TABLE DIP(address text)
URL	CREATE TABLE URL(address text)
AGENT	CREATE TABLE AGENT(agent text, counter int)
POST	CREATE TABLE POST(data text, url text)

Finally, the last feature of the basic mode is the checkbox titled “Block all insecure traffic.” If this box is checked, any requests made to a resource that is HTTP is blocked. This includes GET and POST requests.

The second display mode is the advanced mode, which shows all the information collected, regardless of the protocol used to send the data. This view can take longer to load after long browsing sessions due to the amount of time the queries take to resolve. Navigation buttons have been added to the top to quickly navigate to the data set that interests the user. The first table this mode shows is the User Agent String and the number of times it has been sent. This string is what is parsed to get the Operating system, architecture and Chrome version in the basic view. The next table shows the destination IP addresses of every resource accessed as well as a count of how many times it has been requested. The next table is the POST Data table. This table shows the payloads and destination URLS for every POST request the user

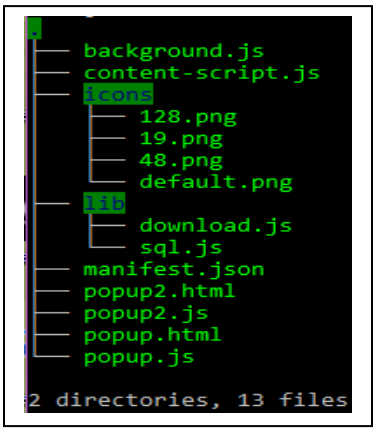
has sent. Similarly, the Search Data table again contains all queries that involve the words ‘search’ or ‘q=.’ The last two tables on the advanced view page show which sites the user has blacklisted and whitelisted.

B. Design

Since the Anonymity Engine is viewing data at the Application layer, it will see all traffic that the browser sends. This means the application will be logging both plaintext data sent using HTTP as well as encrypted HTTPS data. Furthermore, any headers that use HTTPS will be visible in plaintext to the end user. This is a feature of using the Chrome API to log the headers before they are sent to a server. However, it now means that while the complete dataset can paint a perfect picture of a user’s history, a real attacker would not have access to this complete dataset. If a malicious user was using a device such as a network tap to capture traffic on a network from another user, they would only be able to view data that is sent using HTTP. Any information that goes over the wire using HTTPS would be encrypted to the attacker and would not contribute to identifying the victim. To simulate this, and to show users a realistic data set, the Basic mode only logs HTTP traffic. Additionally, the testing and data analysis portion of this research will also ignore any encrypted data. The assumption is that HTTPS data is “safe” and would not expose a user if all of their traffic was encrypted.

The Anonymity engine is a complex tool and in the hopes of reproducibility, this section will explain how the application works in detail at the code level. The code is published on GitHub for reference. The folder structure of the application can be seen in Table 4.

Table 4 Application Structure



All Chrome extensions are required to have a manifest file to dictate what permissions the app will have, what files will be used during run-time and versioning information. The permissions that are given in the anonymity engine are: “webRequest”, “webRequestBlocking” and “<all_urls>.” The webRequest permission allows the application to use the webRequest API. The webRequest Blocking

permission allows the application to block traffic before it leaves the system. The "<all_urls> permission tells the program to run on every possible page.

The manifest.json file also contains content security policy(csp) rules for the application. In order to have critical program functionality, one additional override had to be called. The unsafe-eval permission allows the JavaScript eval command to be executed. This functionality must be allowed in order to use the sql.js library. This is a minimal security risk because Chrome sandboxes its extensions and will not allow them to execute code that is not specifically allowed in the "manifest.json" file. Allowing the unsafe-eval option will only allow eval to be called in the files that the application needs to be run. There is no other use of the command outside of the sql.js library.

The icon folder is where the applications icons are stored. The "default.png" file is what will be displayed on the user's browser bar alongside other installed extensions. The "48.png" and "128.png" icons are named for their pixel size and are used on the main chrome extension page and when shown in other menus. The download.js and sql.js files are third-party libraries that are required for the application to function. Both libraries are licensed under the MIT license and the authors have been cited in the code and this document. Download.js allows the database to be downloaded onto a user's machine, while sql.js contains all of the database functionality that the application will leverage.

The popup.html and popup.js files control what will be displayed when a user clicks on the icon button in their browser. Currently, the pages draw tables of data that contain the results of Sqlite queries which have been performed in the background. Popup1.html renders the basic view for the application and Popup2.html render the advanced view. Their respective JavaScript files control what is drawn in the html divisions.

The background.js file is where most of the work happens. When the program begins, it builds a sqlite database using a version of sqlite that has been compiled for javascript. Then the program sets up listeners for every step of a web request's life cycle. As chrome builds, sends and receives headers the program catches relevant data and stores it in the database. These listener functions are invoked from Chrome's webrequest API. For example, the onBeforeRequest listener is where POST data and GET data get written, so if the request is a post request, the form data is stored before the request is sent, provided the user agreed to send the data. The other stages of a request lifecycle are onBeforeSendHeader, onSendHeaders, onResponseStarted, and onCompleted. OnBeforeSendHeaders captures the user-agent string and the user-locale string from the headers. OnSendHeaders captures the

URL of a website that traffic is being sent to. OnResponseStarted captures the IP address of the URL. Finally, During the OnCompleted step no data is being sent or received, so the listener is set to log to the console when a request finishes.

This file also contains many functions that are essential to functionality. The function "WriteDB()" is for exporting the database so that it can be safely download. "DeleteDB()" reverts all of the tables and global variables back to their original settings. GetBasicData() and GetAdvancedData() consist of the sqlite queries that are called when the popup is opened which populate the tables for the user to view. The "GetQuasiIdentifiers()" table is where the agent header string is parsed for the basic view and it returns the proper operating system and version that the user is browsing on. "getRootUrl()" is a function that does a regular expression replacement that will take in a URL and return only the root of that URL. For example, if a user has browsed to "www.google.com/images/123.jpg", this function will return www.google.com. Finally, the "toggleBlocker()" function flips Booleans in order to make the "block all unsafe traffic" feature work.

The application also makes use of Chrome's console logging features. Experienced users can manually view the log through their settings tab and see all of the data caught in real time. There is no real advantage to this method as the popup boxes also show the data in real-time in addition to being sorted. Still, the feature is left as a development mode for future expansion. Normal users are expected to only be viewing the human readable quasi-identifier tables in their browser by clicking on the extension's icon.

C. Testing Plan

While the program is intended to be used as a standalone application, it will also be used to generate data for analysis. There will be two tests conducted in order to both gauge the effectiveness of the Anonymity Engine and to study the effects of changing browsing history before it has a level of k-anonymity applied. The tests will simulate the browsing sessions of five hypothetical users. The tests will be done on the same machine to mitigate different agent-header stings. Additionally, the anonymity engine will be the only extension running on the test machine during the tests. This will prevent other extensions such as an advertising blockers to interfere with the results of the data. For the first test, each user will navigate to ten websites and view one web pages. Each user will have a different amount of HTTPS sites that they view. User 1 will visit zero HTTPS sites, user 2 will visit 20% HTTPS sites, User 3 will visit 50%, User 4 will visit 80% and User 5 will only visit HTTPS sites. A list of websites has also been chosen for each of the users to navigate to. Search data will not be

included in the analysis, as all major search engines, such as Google or Duckduckgo, only work over HTTPS and the data would be encrypted. Additionally, each of the users has also been given a trait that inspires their browsing session. This was done to give a hypothetical attacker an advantage in identifying which traffic belongs to a user. Finally, each user will have the anonymity engine active during the entirety of their browsing session.

The variable that will change between the first and second test is how often the users choose to allow traffic to an unsafe site. For the first session, no user will allow the anonymity engine to interfere and will allow access to any site. For the second test, the users will again navigate through their sites, however they will only allow one insecure request per site. This will allow them to visit their website while exposing as little as possible. After all users have completed a test session, the data will be aggregated and a k-anonymity algorithm applied. Should the anonymity engine impact the privacy of the users in any way, the difference between the two tests will display the significance. The table of users and sites they will visit can be seen in Table 5 below.

Table 5 Experiment Users, URLs and Traits

ID	**User1**	**User2**	**User 3**
Title	**Bodybuilder (0% https)**	**Businessman (20% https)**	**CS Student (50%https)**
1	www.ebay.com	https://www.google.com	www.thedenverpost.com
2	imgur.com	https://www.reddit.com	https://www.reddit.com
3	ideafit.com	www.fitnessmagazine.com	www.thehackernews.com
4	www.cnn.com	www.imdb.com	https://threatpost.com
5	www.fitnessmagazine.com	www.thedenverpost.com	www.ebay.com
6	www.menshealth.com	www.ebay.com	https://www.amazon.com
7	www.imdb.com	www.cnn.com	www.stackoverflow.com
8	www.apple.com	www.time.com/money	https://github.com
9	www.planetfitness.com	www.nasdaq.com	www.colorado.edu
10	www.muscleandfitness.com	www.marketwatch.com	https://w3schools.com

User 4	**User 5**
General user (80% https)	**Paranoid user (100% https)**
https://www.google.com	https://amazon.com
https://www.reddit.com	https://www.reddit.com
https://www.facebook.com	https://www.youtube.com
https://www.amazon.com	https://www.wunderground.com
https://www.wikipedia.com	https://www.wikipedia.com
https://www.linkedin.com	https://www.github.com
https://www.dropbox.com	https://www.duckduckgo.com
https://www.yahoo.com	https://www.dropbox.com
www.imgur.com	https://twitter.com
www.thedenverpost.com	https://www.pbs.org

In order to make the data compact, as well as incorporating a degree of l-diversity, the sites visited will be categorized into six genres. All traffic will fall into one of the categories of entertainment, health, news, business, social media, learning resource or other. Websites that are classified as entertainment are marked with an “E” in the raw data, and represent sites such as Imgur, Reddit or IMDB. Health websites relate to fitness or medical data and are marked with an “H” in the raw data. The News category accounts for specialized news sites and news aggregators, this category is marked with an “N” in the data. Business sites involve finance or commerce. Social Media sites are any web resource that involves multiple users interacting such as Facebook, and including Twitter. These are denoted with an “M”. Next, Learning Resource sites are ones that academically oriented. Sites ending with “.edu” as well as search engines and academic forums will be in this category. Websites that users can make purchases on will be put in the shopping category with a “S”. Any advertisement, analytics or metrics will be categorized as “Ads/Analytics” with an “A”. Finally, if a website does not meet one of the above requirements, such as an API call or script resource, it will be categorized as “Other” with an “O”.

When gathering data, it was observed that some of the trials were getting more data than could reasonably be parsed. In order to make the data set more manageable, all URLs accessed were stripped to their root components and requests were then counted. After all of the data was collected, it was than normalized and any traffic entry which consisted of less than .001% was removed. This only had a noticeable effect for the first two users in experiment 1, where entries with a count of one or two were removed. For example, User One in Experiment One was reduced from two hundred sixty-five entries down to one hundred and seventy. The reduction is noted in the Raw Data section in Appendix B.

.In addition to URLs visited, the table also shows URLS which were sent POST data as a quasi-identifier. This data should be considered highly sensitive because some POST data is entered by the user. Therefore, having a higher than normal count in one of the categories in this column can be indicative of user’s trait. Finally, the agent-header information that the anonymity engine stores in shown in the tables for completeness. Due to the fact that having different operating systems or browser versions would completely identify a user in such a small sample set, these have been set as constants.

IV. Results and Conclusions

A. Results

This section will compile the data that the tests produced. The first table shows each user with their quasi-identifiers as portrayed in a method similar to the table in the introduction. Table 6 categorizes all the URLS visited but does not obfuscate the number of times the website has been accessed. This is to act as a baseline for observations when observing the following two tables. The goal will be to try to determine the plausibility of identifying which user generated each data set. In order to make things simple, the order of the users remains the same as in Table 5. User 1 is still a "bodybuilder" so we can expect to see data that skews favorably towards websites in the "Health" category. The second user is a "businessman" so he should logically have more traffic in that category. The computer science student as user 3 should likewise trend towards Learning Resource websites. The fourth user, a general user, has no defining traits and uses a mix of sites that the other would have also used. The last user, a "paranoid" type, can be expected to generate the least amount of results since they only visit encrypted HTTPS web sites.

Based on the raw counts of requests to each type of website, it would not be impossible to guess who owns each piece of browsing history in the table below. The first user has the most websites in the Health category by a generous margin. It would be safe to assume that the 'bodybuilder" type would be exposed from this data. Similarly, the second user is the only user with any traffic in the Business category. The third user is the only user with any websites visited in the Learning Resource category. The fourth user has similar numbers in all categories that they generated traffic in. Finally, the paranoid user is revealed through his lack of traffic! The URLs that were sent POST data should also reflect these identifiers. However, the POST data reveal very little extra to identify any of the users. This is likely reflective of the data generating process which did not spend long browsing each website. Longer browsing sessions performed by real users would likely add enough requests to make an impact on the data.

Table 6 Experiment 1 Data with Raw Request Counts

USER	AGENT HEADERS	URLS Visited	URLS POSTED
User 1	**Operating System: Windows 10.0** **Architecture: 64-bit** **Chrome Version: 56.0.2924.87**	**Website Type** / **Count** Entertainment (E) 155 Health (H) 166 News (N) 53 Business (B) 0 Shopping (S) 173 Ads/Analytics (A) 211 Learning Resource (L) 0 Other (O) 639 Social Media (M) 8 Total: 1405	**Website Type** / **Count** Entertainment (E) 2 Health (H) 0 News (N) 0 Business (B) 0 Shopping (S) 4 Ads/Analytics (A) 9 Learning Resource (0 Other (O) 21 Social Media (M) 0 Total: 36
User 2	**Operating System: Windows 10.0** **Architecture: 64-bit** **Chrome Version: 56.0.2924.87**	**Website Type** / **Count** Entertainment (E) 85 Health (H) 65 News (N) 179 Business (B) 67 Shopping (S) 231 Ads/Analytics (A) 374 Learning Resource (L) 0 Other (O) 671 Social Media (M) 0 Total: 1672	**Website Type** / **Count** Entertainment (E) 4 Health (H) 0 News (N) 0 Business (B) 1 Shopping (S) 4 Ads/Analytics (A) 2 Learning Resource (L) 0 Other (O) 31 Social Media (M) 0 Total: 42
User 3	**Operating System: Windows 10.0** **Architecture: 64-bit** **Chrome Version: 56.0.2924.87**	**Website Type** / **Count** Entertainment (E) 10 Health (H) 0 News (N) 28 Business (B) 0 Shopping (S) 163 Ads/Analytics (A) 79 Learning Resource (L) 37 Other (O) 138 Social Media (M) 2 Total: 457	**Website Type** / **Count** Entertainment (E) 0 Health (H) 0 News (N) 0 Business (B) 0 Shopping (S) 3 Ads/Analytics (A) 0 Learning Resource (L) 1 Other (O) 0 Social Media (M) 0 Total: 4
User 4	**Operating System: Windows 10.0** **Architecture: 64-bit** **Chrome Version: 56.0.2924.87**	**Website Type** / **Count** Entertainment (E) 85 Health (H) 0 News (N) 20 Business (B) 0 Shopping (S) 0 Ads/Analytics (A) 44 Learning Resource (L) 0 Other (O) 53 Social Media (M) 3 Total: 205	**Website Type** / **Count** Entertainment (E) 0 Health (H) 0 News (N) 0 Business (B) 0 Shopping (S) 0 Ads/Analytics (A) 0 Learning Resource (L) 0 Other (O) 0 Social Media (M) 0 Total: 0
User 5	**Operating System: Windows 10.0** **Architecture: 64-bit** **Chrome Version: 56.0.2924.87**	**Website Type** / **Count** Entertainment (E) 0 Health (H) 0 News (N) 0 Business (B) 0 Shopping (S) 0 Ads/Analytics (A) 0 Learning Resource (L) 0 Other (O) 0 Social Media (M) 0 Total: 0	**Website Type** / **Count** Entertainment (E) 0 Health (H) 0 News (N) 0 Business (B) 0 Shopping (S) 0 Ads/Analytics (A) 0 Learning Resource (L) 0 Other (O) 0 Social Media (M) 0 Total: 0

Table 7 is the same as the previous, only k-anonymity has been applied to each sub-table of URLs. For this study, a value of 3 has been selected for each table. Again, this means that for each list of URL's visited, there are at least three rows that have the same value. We can see immediately that k-anonymity is working as intended as there is now some confusion as to which data belongs to which user. Trying to identify the "bodybuilder" type would lead to either of the first two users as possible solutions. Since an outsider would not be able to know how much traffic a single visit to a "health" related site generates, it would be feasible to consider either User One's traffic with "more than 50 requests" or User Two's "up to sixty-six" browsing history.

The same scenario occurs again when we try to find the businessman type user. Comparing the traffic from User One's "less than one-hundred" requests verses User Two's "greater than sixty-six" makes it hard to say for certain which user generated each piece of traffic. This also happens when attempting to compare users Three, Four and Five to find the student type. Since all have similar values across all the categories, it is harder to conclude the owner of each data set. It would be quite reasonable for an attacker to think that User Five, the paranoid type, was the student type due to the high ceiling on the "learning resource" category.

In the above scenarios, only the URLs visited were being considered, but the URLs which were sent POST requests can also affect how the data is interpreted. We know from the plaintext data that this POST data reveals nothing noteworthy. However, once k-anonymity is applied, the values can appear to have an impact to an outsider. For example, User Two now appears to have a stronger chance of being the bodybuilder type due to the higher threshold value on the Health category in their data. Furthermore, with the data having the same anonymized values for users Three, Four, and Five it is much harder to pick out the "general" user.

These results show how effective the k-anonymity model is at anonymizing identifiers in a way that still keeps the overall data intact for interested parties. Table 7 is a good example of what an internet service provider might share with a research group who is studying trends in what user's view on the internet.

Table 7 Experiment 1 Data with k values of 3

USER	AGENT HEADERS	URLS Visited	URLS POSTED
User 1	Operating System: Windows 10.0 Architecture: 64-bit Chrome Version: 56.0.2924.87	**Website Type** / **Count** Entertainment (E) >50 Health (H) >50 News (N) >50 Business (B) <100 Social Media (M) <100 Learning Resource (L) <100 Shopping (S) >170 Ads/Analytics (A) >170 Other (O) >170	**Website Type** / **Count** Entertainment (E) <3 Social Media (M) <3 Health (H) <3 News (N) <5 Learning Resource (L) <5 Business (B) <5 Shopping (S) >3 Ads/Analytics (A) >3 Other (O) >3
User 2	Operating System: Windows 10.0 Architecture: 64-bit Chrome Version: 56.0.2924.87	**Website Type** / **Count** Entertainment (E) >66 News (N) >66 Business (B) >66 Learning Resource (L) <=65 Health (H) <=65 Social Media (M) <=65 Shopping (S) >200 Ads/Analytics (A) >200 Other (O) >200	**Website Type** / **Count** Entertainment (E) >=4 Shopping (S) >=4 Other (O) >=4 Business (B) <5 Ads/Analytics (A) <5 Learning Resource (L) <5 Health (H) <8 News (N) <8 Social Media (M) <8
User 3	Operating System: Windows 10.0 Architecture: 64-bit Chrome Version: 56.0.2924.87	**Website Type** / **Count** Social Media (M) <5 Health (H) <5 Business (B) <5 Entertainment (E) >=10 News (N) >=10 Learning Resource (L) >=10 Shopping (S) >75 Ads/Analytics (A) >75 Other (O) >75	**Website Type** / **Count** Entertainment (E) <10 Health (H) <10 News (N) <10 Business (B) <=3 Shopping (S) <=3 Ads/Analytics (A) <=3 Learning Resource (L) <5 Other (O) <5 Social Media (M) <5
User 4	Operating System: Windows 10.0 Architecture: 64-bit Chrome Version: 56.0.2924.87	**Website Type** / **Count** Health (H) <=20 News (N) <=20 Business (B) <=20 Shopping (S) <10 Learning Resource (L) <10 Social Media (M) <10 Entertainment (E) >30 Ads/Analytics (A) >30 Other (O) >30	**Website Type** / **Count** Entertainment (E) <=5 Health (H) <=5 News (N) <=5 Business (B) <10 Shopping (S) <10 Ads/Analytics (A) <10 Learning Resource (L) <30 Other (O) <30 Social Media (M) <30
User 5	Operating System: Windows 10.0 Architecture: 64-bit Chrome Version: 56.0.2924.87	**Website Type** / **Count** Entertainment (E) <10 Health (H) <10 News (N) <10 Business (B) <30 Shopping (S) <30 Ads/Analytics (A) <30 Learning Resource (L) <50 Other (O) <50 Social Media (M) <50	**Website Type** / **Count** Entertainment (E) <10 Health (H) <10 News (N) <10 Business (B) <=25 Shopping (S) <=25 Ads/Analytics (A) <=25 Learning Resource (L) <30 Other (O) <30 Social Media (M) <30

Table 8 Experiment 2 Data with k values of 3

USER	AGENT HEADERS	URLS Visited	URLS POSTED
User 1	**Operating System: Windows 10.0** **Architecture: 64-bit** **Chrome Version: 56.0.2924.87**	**Website Type** / **Count** Entertainment (E) <=80 Health (H) <=80 News (N) <=80 Business (B) <25 Shopping (S) <25 Ads/Analytics (A) <25 Learning Resource (L) <10 Other (O) <10 Social Media (M) <10	**Website Type** / **Count** Entertainment (E) <10 Health (H) <10 News (N) <10 Business (B) <=25 Shopping (S) <=25 Ads/Analytics (A) <=25 Learning Resource (L) <30 Other (O) <30 Social Media (M) <30
User 2	**Operating System: Windows 10.0** **Architecture: 64-bit** **Chrome Version: 56.0.2924.87**	**Website Type** / **Count** Health (H) <80 News (N) <80 Business (B) <80 Entertainment (E) <5 Shopping (S) <5 Ads/Analytics (A) <5 Learning Resource (L) <3 Other (O) <3 Social Media (M) <3	**Website Type** / **Count** Entertainment (E) <10 Health (H) <10 News (N) <10 Business (B) <=25 Shopping (S) <=25 Ads/Analytics (A) <=25 Learning Resource (L) <30 Other (O) <30 Social Media (M) <30
User 3	**Operating System: Windows 10.0** **Architecture: 64-bit** **Chrome Version: 56.0.2924.87**	**Website Type** / **Count** Entertainment (E) <20 Health (H) <20 Business (B) <20 Learning Resource (L) <45 News (N) <45 Shopping (S) <45 Ads/Analytics (A) <10 Other (O) <10 Social Media (M) <10	**Website Type** / **Count** Entertainment (E) <10 Health (H) <10 News (N) <10 Business (B) <=25 Shopping (S) <=25 Ads/Analytics (A) <=25 Learning Resource (L) <30 Other (O) <30 Social Media (M) <30
User 4	**Operating System: Windows 10.0** **Architecture: 64-bit** **Chrome Version: 56.0.2924.87**	**Website Type** / **Count** Entertainment (E) <10 Health (H) <10 Ads/Analytics (A) <10 Business (B) <15 Shopping (S) <15 News (N) <15 Learning Resource (L) <5 Other (O) <5 Social Media (M) <5	**Website Type** / **Count** Entertainment (E) <10 Health (H) <10 News (N) <10 Business (B) <=25 Shopping (S) <=25 Ads/Analytics (A) <=25 Learning Resource (L) <30 Other (O) <30 Social Media (M) <30
User 5	**Operating System: Windows 10.0** **Architecture: 64-bit** **Chrome Version: 56.0.2924.87**	**Website Type** / **Count** Entertainment (E) <10 Health (H) <10 News (N) <10 Business (B) <30 Shopping (S) <30 Ads/Analytics (A) <30 Learning Resource (L) <50 Other (O) <50 Social Media (M) <50	**Website Type** / **Count** Entertainment (E) <10 Health (H) <10 News (N) <10 Business (B) <=25 Shopping (S) <=25 Ads/Analytics (A) <=25 Learning Resource (L) <30 Other (O) <30 Social Media (M) <30

The final table, Table 8, shows the k-anonymized data from Experiment 2. In short, this is the data set that shows how a k-anonymity table would look when the user specifically chooses to minimize the data that they send in HTTP. This led to an interesting result once all the data had been compiled. It became apparent that the act of blocking a majority of the unsafe content appeared to highlight what little unsafe content was sent. This, in turn, lead to the anonymized data favoring the categories that would identify each user.

The high threshold values for User One and User Two seem to highlight that their favorite sites to browse must be in the Business, News, Health or Entertainment categories. Unfortunately, there is no overlap on the categories that would identify them, and it appears obvious that User One must be the "bodybuilder" type. If an attacker assumes this is the case, then User Two must be the "businessman" type since it must have a high value given the other entries in the table. The other users are harder to identify. However, the remaining users gain a slight bit of protection. As the data shows, the "student" type could be User Three or User Five since the values for Learning Resources are similar across those users. Nothing unique can be observed from Four, would could also potentially incriminate it given that it represents a general user, but again an attacker would be guessing between the last two users.

The identifiability of users One and Two could be lowered by raising the threshold values for the other entries in the URLS visited column. This fix would mitigate the problem above but the raw data itself is still worthy of analysis. As a user reduces the number of insecure sites that they access, the sites that they do access are much more obvious. This is different to the data from the first experiment where the extra sites that a user was accessing acted as a shield obscuring the sites they intended to visit. There are several situations in experiment 1 where a site such as stumbleupon (an entertainment website) which was not intentionally browsed to, had more requests than a site which was intentionally browsed to. These extraneous requests can help pad out request numbers and overinflate the k-anonymity values. This makes it harder to predict which user is which after k-anonymity is applied.

However, despite the apparent benefits in this situation it is a much better solution to simply switch to browsing more HTTPS sites rather than trusting the sheer amount of advertising and metrics sites to anonymize one's real browsing history. This method of obfuscating data is not recommended in the real world. While adding extraneous sites to one's browsing history may make it harder for an attacker to pinpoint what they've visited in a list, there is a real risk that these unsafe HTTP sites could have their own malware or identity stealing implements. The anonymity engine protects against these types of threats by showing the user when their browser is trying to access one of these sites. As Experiment 1 shows, browsing to a mere ten sites in HTTP can cause a web browser to connect to over one thousand other web sites in plaintext in order to load all the assets for that single site.

It is worthwhile to consider a third Experiment that could have been conducted where the users blocked every piece of HTTP traffic and only browsed using encrypted sites. This would lead the first four users to have the exact same apparent traffic as the fifth user, that is to say, all users would have tables with only zero entries. In this situation, nothing could be gleaned from any user, and the k - anonymity thresholds would be completely suggestive.

B. Conclusions

The biggest factor in determining a user's identifiability are the decisions they make when browsing the internet. The Anonymity Engine shines in this regard as it exposes the choice to connect insecurely, a decision which most users do not even realize that they have. Showing users that a site that they are connecting to could give them away will probably not affect them on a single site basis. However, if a user switches to using more secure sites as a result, their overall identifiability is reduced. This can increase over time until a user browses in a completely secure fashion. This can be seen in User Four in both experiments. With only a 20% HTTP exposure rate, it is very difficult to draw any meaningful conclusions from their data, especially if they are only browsing popular websites. If User Four used the Anonymity Engine daily, they would probably grow to a 90% or 100% HTTPS exposure rate as they learn what resources are insecure, increasing the user's personal privacy over time.

Returning to the research question, the second conclusion is that modifying data before it undergoes k-anonymity can, in fact, have a meaningful impact on the data after k-anonymity has been applied. However, this is not always a positive impact nor is it true in all cases. The data from Experiment 2 shows that reducing the amount of data significantly can cause what is visible to be overrepresented in a k-anonymity chart. This is evidenced by the first few users in the experiment. The degree to which this effect occurs is largely dependent on the value chosen for k when applying the algorithm. The last three users were actually made more difficult to identify in this experiment after using a significant amount of HTTPS traffic and minimizing their identifiers. There is also a direct correlation between the number of users using HTTP and their identifiability in a k-anonymity chart. Both experiments show that is harder to place users to a trait as the HTTPS usage rate rise. Therefore, as users browse more secure sites it becomes harder to identify any of them in a crowd.

Finally, there is also the case to be made for anonymizing data so that the group constructing the k-anonymized chart cannot identify a user. In the real world, a k-anonymity chart would be constructed by a third-party group such as an internet service provider. There is a strong argument to be made for wanting to anonymize data before it is visible by one of these groups. Using a tool such as the anonymity engine to build wariness against unsafe sites will help protect a user from giving away data. Learning how much data is being sent in an unsafe fashion will allow users to make more informed decisions when they

browse the internet. In a world where single page loads can generate hundreds of requests, knowing what isn't safe could be the difference between being identified in a crowd and staying safely hidden in the noise.

V. Bibliography

[1] L. Sweeney, “k-Anonymity: A Model for Protecting Privacy.,” *International Journal of Uncertainty, Fuzziness & Knowledge-Based Systems*, vol. 10, no. 5, p. 557, Oct. 2002.
[2] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, “L-diversity: Privacy beyond K-anonymity,” *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, p. 3–es, Mar. 2007.
[3] N. Li, T. Li, and S. Venkatasubramanian, “t-Closeness: Privacy Beyond k-Anonymity and L-Diversity,” in *2007 IEEE 23rd International Conference on Data Engineering*, 2007, pp. 106–115.
[4] P. M. Vel Kumar and M. Karthikeyan, “L-diversity on K-anonymity with External Database for Improving Privacy Preserving Data Publishing,” *International Journal of Computer Applications*, vol. 54, no. 14, 2012.
[5] X. M. Ren, B. X. Jia, K. C. Wang, and J. Cheng, “Research on k-Anonymity Privacy Protection of Social Network,” *Applied Mechanics and Materials*, vol. 530–531, pp. 701–704, Feb. 2014.
[6] L. Zou, L. Chen, and M. T. Özsu, “K-Automorphism: A general framework for privacy preserving network publication,” *Proceedings of the VLDB Endowment*, vol. 2, no. 1, pp. 946–957, 2009.
[7] J. Cheng, A. W. Fu, and J. Liu, “K-isomorphism: privacy preserving network publication against structural attacks,” in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, 2010, pp. 459–470.
[8] C.-G. Liu, I.-H. Liu, W.-S. Yao, and J.-S. Li, “K-Anonymity against neighborhood attacks in weighted social networks,” *Security and Communication Networks*, vol. 8, no. 18, pp. 3864–3882, Dec. 2015.
[9] B. Niu, Q. Li, X. Zhu, G. Cao, and H. Li, “Achieving k-anonymity in privacy-aware location-based services,” in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, 2014, pp. 754–762.
[10] B. Niu, Z. Zhang, X. Li, and H. Li, “Privacy-area aware dummy generation algorithms for Location-Based Services,” in *2014 IEEE International Conference on Communications (ICC)*, 2014, pp. 957–962.
[11] X. Xiao and Y. Tao, “M-invariance: towards privacy preserving re-publication of dynamic datasets,” in *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, 2007, pp. 689–700.
[12] R. J. Bayardo and R. Agrawal, “Data privacy through optimal k-anonymization,” in *21st International Conference on Data Engineering (ICDE’05)*, 2005, pp. 217–228.
[13] A. M. Omer and M. M. B. Mohamad, “SIMPLE AND EFFECTIVE METHOD FOR SELECTING QUASI-IDENTIFIER,” *Journal of Theoretical and Applied Information Technology*, vol. 89, no. 2, p. 512, 2016.
[14] X. Zhang, C. Liu, S. Nepal, and J. Chen, “An efficient quasi-identifier index based approach for privacy preservation over incremental data sets on cloud,” *Journal of Computer and System Sciences*, vol. 79, no. 5, pp. 542–555, Aug. 2013.
[15]X. Sun, L. Sun, and H. Wang, “Extended k-anonymity models against sensitive attribute disclosure,” Computer Communications, vol. 34, no. 4, pp. 526–535, Apr. 2011

APPENDIX A

CODE SUBMISSION

1. Background.js

```
////////////////////////////
//Globals and imports
////////////////////////////
const WEB_REQUEST = chrome.webRequest;  // Acess the Chrome API
var last_ip;                            // stores the last ip accessed.
Prevents duplicate calls to db
var last_Headers;                       // stores the last headers, prevents
duplicate calls to db
var last_Agent;                         // stores the last agent string,
reduces database writes since this rarely changes
var sql = window.SQL                    // imports the sql.js file
var db = new sql.Database();
////////////////////////////
//Create Tables
sqlstr = "CREATE TABLE DIP(address text);"
sqlstr += "CREATE TABLE URL(address text);"
sqlstr += "CREATE TABLE AGENT(agent text, counter int);"
sqlstr += "CREATE TABLE POST(data text, url text);"
db.run(sqlstr); // Run the query without returning anything

////////////////////////////
//Global vars containing Tables.
//Popup reads these.
////////////////////////////
var binaryArray;        // database contents (for download)
var iptable;            // table for DIPS
var urltable;           // all urls accessed
var agnttable;          // stores user agent information
var httpsposttable;     // contains all post requests
var httpposttable;      // only contains http post requests
var searchtable;        // contains urls that have the word 'search'
var basicsearchtable;   // modified values from basicurl table
var basicurltable;      // only shows http urls
var block_all=false;    // controlled from popup.html, blocks all http when
true
var allowed_urls=[];    // list of whitelisted urls
var blocked_urls=[];    // list of blacklisted urls
var post_urls=[];       // list of urls browser posted to
////////////////////////////
//Asynchronous Listener functions
////////////////////////////
WEB_REQUEST.onBeforeRequest.addListener(
    function(details) {
      console.log("Preparing Headers----------------------------")
      ////////////////////////////////// GET Requests
      if(details.method == "GET"){
      var root_url=getRootUrl(details.url); //Get the root URL to block or
allow
      if(details.url.includes('http:')){
        if(block_all == false){
          if (blocked_urls.includes(root_url)){
```

```
            return {cancel:true};
          }
          if (! allowed_urls.includes(root_url)){ //see if this url has been
accessed before(
            var blocker=window.confirm("Chrome is trying to access the
following resource!\n"
                                       +details.url
                                       +"\nYou must choose to allow or block
all future traffic from:\n"
                                       +root_url);
            if (blocker == false){
              blocked_urls.push(root_url);
              return {cancel:true};
            }
          else{
            allowed_urls.push(root_url);
          }
        }
      }
        else{return{cancel:true};}}
    }
    ////////////////////////////////////PUT Requests
    else if(details.method == "PUT"){
      var putdata= JSON.stringify(details.requestBody.UploadData);
      if (typeof putdata == "undefined"){
        putdata = "empty payload";
      }
      var url=JSON.stringify(details.url);
      if(url.includes('http:')){
        if(block_all == false){
          if (url.includes('http:')){
            var blocker=window.confirm("PUT DATA WARNING!\n"
                                       +putdata
                                       +"\nYour browser is trying to send the
above payload to:\n"
                                       +url+
                                       "\nPress OK to allow or Cancel to block
this request");
            if (blocker == false){
              return {cancel:true};
            }
            else{
              allowed_urls.push(url);
            }
          }
        }
      }
    ////////////////////////////////////// POST Requests
      else if(details.method == "POST"){
        if(details.requestBody){
          try{
            var postdata= JSON.stringify(details.requestBody.formData);
            if (typeof postdata == "undefined"){
              if (typeof details.requestBody.raw[0] == "undefined")
                {postdata = "empty payload";}
              else{ //we have non utf data!!
```

```
                postdata =
decodeURIComponent(String.fromCharCode.apply(null,new
Uint8Array(details.requestBody.raw[0].bytes)));
                postdata=encodeURI(postdata);
              }
            }
            var url=JSON.stringify(details.url);
            if(url.includes('http:')){
              if(block_all == false){
                  var blocker=window.confirm("POST DATA WARNING!\n"
                                             +postdata
                                             +"\nYour browser is trying to
send the above payload to:\n"
                                             +url+
                                             "\nPress OK to allow or Cancel to
block this request");
                  if (blocker == false){
                    return {cancel:true};
                  }
                  else{
                    if(!allowed_urls.includes(url))
                    {allowed_urls.push(url);}
                  }
              }
            }
            console.log("POST value:"+ postdata)
            db.run("INSERT INTO POST VALUES (?,?)",[postdata,url]);
          }
          catch(err){
              console.log("Attempted to decode raw POST data bytes. Data is
not unicode string or no bytes sent")
          }
        }
      }
    },
    {urls: ["<all_urls>"]},
    ["blocking", "requestBody"]
);

chrome.webRequest.onBeforeSendHeaders.addListener(
  function(details) {
    if(JSON.stringify(details.requestHeaders) != last_Headers){
      for (var i=0; i < details.requestHeaders.length; i++){
        var tmp = JSON.stringify(details.requestHeaders[i]);
        var value= JSON.stringify(details.requestHeaders[i].value);
        console.log("header value:"+ i + " " + tmp)
        if (details.requestHeaders[i].name === 'User-Agent') {
          if (last_Agent == value){
            db.run("UPDATE AGENT SET counter=(SELECT counter FROM AGENT WHERE
agent = ?)+1 WHERE agent=?", [value,value]);
          }
          else{
            db.run("INSERT INTO AGENT VALUES
(?,1)",[JSON.stringify(details.requestHeaders[i].value)]);
          }
          last_Agent=JSON.stringify(details.requestHeaders[i].value);
        }
```

```
      }
    last_Headers=JSON.stringify(details.requestHeaders);
    }
    return {requestHeaders: details.requestHeaders};
  },
  {urls: ["<all_urls>"]},
  ["blocking", "requestHeaders",]
);

chrome.webRequest.onSendHeaders.addListener(
  function(details){
    console.log("Sending Headers----------------------------")
    console.log("method:" + details.method)
    console.log("site:" + details.url)
    db.run("INSERT INTO URL VALUES (?)",[details.url]);
  },
  {urls: ["<all_urls>"]},
  ["requestHeaders"]
);

chrome.webRequest.onResponseStarted.addListener(
  function(details){
    if(JSON.stringify(details.ip) != last_ip){
    console.log("IP:" + JSON.stringify(details.ip))}
    last_ip=JSON.stringify(details.ip);
    db.run("INSERT INTO DIP VALUES (?)",[last_ip]);
  },
  {urls: ["<all_urls>"]},
  ["responseHeaders"]
);

chrome.webRequest.onCompleted.addListener(
  function(details){
    console.log("Request Completed----------------------------")
  },
  {urls: ["<all_urls>"]},
   ["responseHeaders"]
);
////////////////////////////
//Helper Functions
////////////////////////////
function WriteDB(){
  binaryArray = db.export();
}

function DeleteDB(){
  console.log("Deleting Tables...")
  db.exec("DELETE FROM URL;");
  db.exec("DELETE FROM POST;");
  db.exec("DELETE FROM AGENT;");
  db.exec("DELETE FROM DIP;");
  last_ip=null;
  last_Headers=null;
  last_Agent=null;
  allowed_urls=[];
  blocked_urls=[];
  post_urls=[];
```

```
  console.log("Tables Deleted.")
}

function GetAdvancedData(){
    console.log("Updating DB...")
    var res2 = JSON.stringify(db.exec("select address, count(address) from
DIP WHERE address is not NULL group by address;"));
    iptable = res2;
    res2.free;
    var res3 = JSON.stringify(db.exec("select address, count(address) from
URL WHERE address is not NULL group by address;"));
    urltable = res3;
    res3.free;
    var res4 = JSON.stringify(db.exec("select agent, counter from AGENT;"));
    agnttable = res4;
    res4.free;
    console.log(res4)
    var res5 = JSON.stringify(db.exec("select * from POST WHERE data is not
NULL;"));
    httpsposttable = res5;
    res5.free;
    var res6 = JSON.stringify(db.exec("SELECT * from URL WHERE address LIKE
'%search?%';"));
    searchtable = res6;
    res6.free;
}

function GetBasicData(){
    var res7 = JSON.stringify(db.exec("select distinct
replace(substr(substr(address, instr(address,
'//')+2),0,instr(substr(address, instr(address, '//')+2),'/')),'','') AS url,
count(replace(substr(substr(address, instr(address,
'//')+2),0,instr(substr(address, instr(address, '//')+2),'/')),'','')) AS
count from URL WHERE address like 'http:%' group by url HAVING
COUNT(*)>0;"));
    basicurltable = res7;
    //basicurltable2=basicurltable.concat(allowed_urls);
    res7.free;
    var res8 = JSON.stringify(db.exec("SELECT * from POST WHERE data is not
null and url like '%http:%';"));
    httpposttable = res8;
    res8.free;
    var res9=JSON.stringify(db.exec("SELECT * FROM URL WHERE (address is not
null) and (address like '%http:%') and (address like '%search?%' or address
like '%q=%');"));
    basicsearchtable = res9;
    res9.free;
}

function GetQuasiIdentifiers(){
  var res4 = JSON.stringify(db.exec("select agent from AGENT;"));
  var OS= res4.match(/\([^\)]+\)/g);
  var version=res4.match(/Chrome\/.* /g);
  var version=version[0].split("/");
  if (OS[0].includes("Windows") == true){
    var result=OS[0].split(" ");
    switch(result[2].replace(';','')){
```

```
      case "10.0":
        result="10.0";
        break;
      case "6.1":
        result="7";
        break;
      case "6.3":
        result="8.1";
        break;
      case "6.2":
        result="8"
      case "6.0":
        result="Vista";
        break;
      case "5.2":
        result="XP";
        break;
      case "5.1":
        result="XP";
        break;
      case "5.0":
        result="2000";
        break;
    }
    var result="Windows "+ result;
    if(OS[0].includes("x64")){
      var arch="64-bit";}
    else{
      var arch="32-bit";
    }
  }
  else if(OS[0].match("Mac")== true){
    var result="Mac OS";
  }
  else if(OS[0].match("Linux")==true){
    var result="Linux";
  }

  return [result, arch, version[1]];
}

function getRootUrl(url) {
  return
url.toString().replace(/^(.*\/\/[^\/?#]*).*$/,"$1").replace('http://','');
}

function toggleBlocker(){
  block_all=!block_all;
  console.log(block_all)
}
```

2. Manifest.json

```
{
  "name": "The Anonymity Engine",
  "version": "1.0",
  "description": "Notifies you when you're about to access a resource that
uses HTTP",
  "icons": {  "48" : "icons/48.png",
              "128" : "icons/128.png"},
  "permissions": ["webRequest", "webRequestBlocking","<all_urls>"],
  "content_security_policy": "script-src 'self' 'unsafe-eval'; object-src
'self'",
  "background": {
    "scripts": ["lib/sql.js", "lib/download.js", "popup.js",
"popup2.js","background.js"]
  },
  "browser_action": {
    "default_icon": "icons/default.png",
    "default_title": "The Anonymity Engine",
    "default_popup": "popup.html"
  },
  "manifest_version": 2
}
```

3. Popup.html

```
<!--
HTML for Anonymity Engine. Basic information page
-->
<html>
<head></head>
<style>
table {
    border: 1px solid black;
    border-collapse: collapse;

}
tr, td {
    border: 1px solid black;
    padding: 5px;
    min-width: 100px;
    max-width:250px;
    word-wrap: break-word;
}
</style>
    <div id="top"><a href="top"></a></div>
    <div id='button'></div>
    <div id='button2'></div>
    Block All insecure traffic<div id='radio1'></div>
    <a href="#bottom">Go To Bottom</a>
    <a href="popup2.html">Advanced Mode</a>

    <p> <b>What your Browsing History Says About You:</b>

    <div id = "identifers"></div>

    <b>URLS Visited(plaintext,HTTP):</b>
    <div id ="display5"></div>
        <table id="myTable5"></table>
```

```
    </div>

    <b>Data You've Sent(plaintext,HTTP):</b>
    <div id ="display6"></div>
        <table id="myTable6"></table>
    </div>

    <b>Things you've searched for(plaintext,HTTP,best guess):</b>
    <div id ="display7"></div>
        <table id="myTable7"></table>
    </div>

<div id="bottom"><a href="bottom"></a></div>
<a href="#top">Back to Top</a>
  <script src="popup.js"></script>
    </body>
</html>
```

4. Popup.js

```
////////////////////////////
//js for popup window
////////////////////////////
var myVar;
var background = chrome.extension.getBackgroundPage(); //gets access to the
background page
//////////
    //Buttons
    myButton = document.createElement("input");
    myButton.type = "button";
    myButton.value = "Download Database";
    placeHolder = document.getElementById("button");
    placeHolder.appendChild(myButton);
    document.getElementById("button").onclick = function()
{DownloadFunction()};
  ////////////////////
    myButton2 = document.createElement("input");
    myButton2.type = "button";
    myButton2.value = "Delete Database";
    placeHolder2 = document.getElementById("button2");
    placeHolder2.appendChild(myButton2);
    document.getElementById("button2").onclick = function()
{DeleteFunction()};
  ////////////////////
    myButton3 = document.createElement("input");
    myButton3.setAttribute("id", "jingles");
    myButton3.type = "checkbox";
    placeHolder3 = document.getElementById("radio1");
    placeHolder3.appendChild(myButton3);
    if (background.block_all == true){
      document.getElementById("jingles").checked = true;}
    else{
      document.getElementById("jingles").checked = false;
    }
    document.getElementById("radio1").onclick = function()
{background.toggleBlocker()};
    ////////////////////
```

```
background.GetBasicData();

window.onload = function(){
  try{
    ////////////////////////
    //Get Identifiers from agent headers
    try{
      var identifiers=background.GetQuasiIdentifiers();
      document.getElementById("identifers").innerHTML = "<b> Operating
System: "+identifiers[0]+
                                                        "<p> Architecture: "+
identifiers[1]+
                                                        "<p> Chrome Version:
" +identifiers[2];}
    catch(err){}
    ////////////////////////
    //live data display
    var U = setInterval(function() {
        document.getElementById("display5").innerHTML = "<table
id='myTable5'></table>";
        var data = JSON.parse(background.basicurltable); // Display Var
        if (typeof data[0] !== "undefined"){
          TableMaker(data[0],"myTable5");}
        else{
         document.getElementById("display5").innerHTML = "<table
id='myTable5'><col width='50'><col
width='50'><tr><td></td><td></td></tr></table>";
        }
    }, 500);
    var V = setInterval(function() {
        document.getElementById("display6").innerHTML = "<table
id='myTable6'></table>";
        var data = JSON.parse(background.httpposttable); // Display Var
        if (typeof data[0] !== "undefined"){
          TableMaker(data[0],"myTable6");}
        else{
         document.getElementById("display6").innerHTML = "<table
id='myTable6'><col width='50'> <col
width='50'><tr><td></td><td></td></tr></table>";
        }
    }, 500);
    var W = setInterval(function() {
        document.getElementById("display7").innerHTML = "<table
id='myTable7'></table>";
        var data = JSON.parse(background.basicsearchtable); // Display Var
        if (typeof data[0] !== "undefined"){
          TableMaker(data[0],"myTable7");}
        else{
         document.getElementById("display7").innerHTML = "<table
id='myTable7'><col width='50'> <col
width='50'><tr><td></td><td></td></tr></table>";
        }
    }, 500);
}
  catch(err){
    console.log("Warning: Cannot Print Database that is empty.")
  }
```

```
};

function TableMaker(stuff,id) {
  var content = stuff.columns[0];
  var table = document.getElementById(id);
  var header = table.createTHead();
  var headerRow = header.insertRow(0);
  var numCols = stuff.columns.length;

  for(var i=0; i < numCols; i++)
  {
      var cell = headerRow.insertCell(i);
      cell.innerHTML = "<b>" + stuff.columns[i] + "</b>";
  }

  var numRows = stuff.values.length;

  for(var j=0; j < numRows; j++)
  {
      var row = table.insertRow();
      for(var i=0; i < numCols; i++)
          {
          var cell1 = row.insertCell(-1);
          cell1.innerHTML = stuff.values[j][i];
        }
  }
}

function MakeColumn(basic_array, id) {
   var table = document.getElementById(id);
     for(var i=0; i < basic_array.length; i++)
     {
          var row = table.insertRow();
          var cell1 = row.insertCell(-1);
          cell1.innerHTML = basic_array[i];
     }
}

function DownloadFunction(){
    background.WriteDB();
    download(background.binaryArray, "sql.db", "text/plain");
}

function DeleteFunction(){
    background.DeleteDB();
}

//download.js v4.2, by dandavis; 2008-2017.
//[MIT] see http://danml.com/download.html for tests/usage
function(r,l){"function"==typeof
define&&define.amd?define([],l):"object"==typeof
exports?module.exports=l():r.download=l()})(this,function(){return function
l(a,e,k){function q(a){var h=a.split(/[:;,]/);a=h[1];var
h=("base64"==h[2]?atob:decodeURIComponent)(h.pop()),d=h.length,b=0,c=new
Uint8Array(d);for(b;b<d;++b)c[b]=h.charCodeAt(b);return new
f([c],{type:a})}function m(a,b){if("download"in d)return
d.href=a,d.setAttribute("download",n),d.className="download-js-
```

```
link",d.innerHTML="downloading...",d.style.display="none",document.body.appen
dChild(d),setTimeout(function(){d.click(),document.body.removeChild(d),!0===b
&&setTimeout(function(){g.URL.revokeObjectURL(d.href)},250)},66),!0;if(/(Vers
ion)\/(\d+)\.(\d+)(?:\.(\d+))?.*Safari\//.test(navigator.userAgent))return/^d
ata:/.test(a)&&(a="data:"+a.replace(/^data:([\w\/\-\+]+)/,"application/octet-
stream")),!window.open(a)&&confirm("Displaying New Document\n\nUse Save As...
to download, then click back to return to this
page.")&&(location.href=a),!0;var
c=document.createElement("iframe");document.body.appendChild(c),!b&&/^data:/.
test(a)&&(a="data:"+a.replace(/^data:([\w\/\-\+]+)/,"application/octet-
stream")),c.src=a,setTimeout(function(){document.body.removeChild(c)},333)}va
r g=window,b=k||"application/octet-
stream",c=!e&&!k&&a,d=document.createElement("a");k=function(a){return
String(a)};var
f=g.Blob||g.MozBlob||g.WebKitBlob||k,n=e||"download",f=f.call?f.bind(g):Blob;
"true"===String(this)&&(a=[a,b],b=a[0],a=a[1]);if(c&&2048>c.length&&(n=c.spli
t("/").pop().split("?")[0],d.href=c,-1!==d.href.indexOf(c))){var p=new
XMLHttpRequest;return
p.open("GET",c,!0),p.responseType="blob",p.onload=function(a){l(a.target.resp
onse,n,"application/octet-
stream")},setTimeout(function(){p.send()},0),p}if(/^data:([\w+-]+\/[\w+.-
]+)?[,;]/.test(a)){if(!(2096103.424<a.length&&f!==k))return
navigator.msSaveBlob?navigator.msSaveBlob(q(a),n):m(a);a=q(a),b=a.type||"appl
ication/octet-stream"}else if(/([\x80-\xff])/.test(a)){e=0;var c=new
Uint8Array(a.length),t=c.length;for(e;e<t;++e)c[e]=a.charCodeAt(e);a=new
f([c],{type:b})}a=a instanceof f?a:new
f([a],{type:b});if(navigator.msSaveBlob)return
navigator.msSaveBlob(a,n);if(g.URL)m(g.URL.createObjectURL(a),!0);else{if("st
ring"==typeof a||a.constructor===k)try{return
m("data:"+b+";base64,"+g.btoa(a))}catch(h){return
m("data:"+b+","+encodeURIComponent(a))}b=new
FileReader,b.onload=function(a){m(this.result)},b.readAsDataURL(a)}return!0}}
);
```

5. Popup2.html

```
<!--
HTML for Anonymity Engine. Advanced information page
-->
<html>
<head></head>
<style>
table {
    border: 1px solid black;
    border-collapse: collapse;
}
tr, td {
    border: 1px solid black;
    padding: 5px;
    min-width: 100px;
    max-width:250px;
    word-wrap: break-word;
}
</style>
    <body>
    <div id="top"><a href="top"></a></div>
    <div id='button'></div>
```

```
    <div id='button2'></div>
    <a href="#bottom">Go To Bottom</a>
    <a href="#display2">Go To URL Data</a>
    <a href="#display3">Go To POST Data</a>
    <a href="#display4">Go To SEARCH Data</a>
    <a href="#display8">Go To Allowed URLS</a>
    <a href="popup.html">Basic View</a>

        <p> <b>User Agent String:</b>
    <div id ="display0"></div>
        <table id="myTable0"></table>
    </div>
            <p> <b>GET DIP Addresses:</b>
    <div id ="display"></div>
        <table id="myTable"></table>
    </div>
        <p> <b>ALL URLS Requested(HTTP/HTTPS):</b>
    <div id ="display2"></div>
        <table id="myTable2"></table>
    </div>
        <p> <b>POST Data:</b>
    <div id ="display3"></div>
        <table id="myTable3"></table>
    </div>
        <p> <b>(probable) Search Data:</b>
    <div id ="display4"></div>
        <table id="myTable4"></table>
    </div>
    <b>URLs you've allowed:</b>
    <div id ="display8"></div>
        <table id="myTable8"></table>
    </div>
    <b>URLs you've blocked:</b>
    <div id ="display9"></div>
        <table id="myTable9"></table>
    </div>
<div id="bottom"><a href="bottom"></a></div>
<a href="#top">Back to Top</a>
  <script src="popup2.js"></script>
    </body>
</html>
```

6. Popup2.js

```
////////////////////////////
//js for popup window
////////////////////////////
var myVar;
var background = chrome.extension.getBackgroundPage(); //do this in global
scope for popup.js
//////////
    //Buttons
    myButton = document.createElement("input");
    myButton.type = "button";
    myButton.value = "Download Database";
    placeHolder = document.getElementById("button");
    placeHolder.appendChild(myButton);
```

```
    document.getElementById("button").onclick = function()
{DownloadFunction()};
  ////////////////////

    myButton2 = document.createElement("input");
    myButton2.type = "button";
    myButton2.value = "Delete Database";
    placeHolder2 = document.getElementById("button2");
    placeHolder2.appendChild(myButton2);
    document.getElementById("button2").onclick = function()
{DeleteFunction()};
  ////////////////////

background.GetAdvancedData();

window.onload = function(){
  try{
    try{
    var identifiers=background.GetQuasiIdentifiers();
    document.getElementById("identifers").innerHTML = "<b> Operating System:
"+identifiers[0]+
                                                       "<p> Architecture: "+
identifiers[1]+
                                                       "<p> Chrome Version: "
+identifiers[2];}
    catch(err){}

    ////////////////////
    //live data display
    var t = setInterval(function() {
      document.getElementById("display").innerHTML = "<table
id='myTable'></table>";
      var data = JSON.parse(background.iptable); // Display Var
      if (typeof data[0] !== "undefined"){
        TableMaker(data[0],"myTable");
      }
      else{
        document.getElementById("display").innerHTML = "<table
id='myTable'><col width='50'><col
width='50'><tr><td></td><td></td></tr></table>";
      }
    }, 500);
    //
    var Q = setInterval(function() {
      document.getElementById("display2").innerHTML = "<table
id='myTable2'></table>";
      var data = JSON.parse(background.urltable); // Display Var
      if (typeof data[0] !== "undefined"){
        TableMaker(data[0],"myTable2");
      }
      else{
        document.getElementById("display2").innerHTML = "<table
id='myTable2'><col width='50'><col
width='50'><tr><td></td><td></td></tr></table>";
      }
    }, 500);
    //
```

```
    var R = setInterval(function() {
      document.getElementById("display0").innerHTML = "<table
id='myTable0'></table>";
      var data = JSON.parse(background.agnttable); // Display Var
      if (typeof data[0] !== "undefined"){
        TableMaker(data[0],"myTable0");
      }
      else{
        document.getElementById("display0").innerHTML = "<table
id='myTable0'><col width='50'><col
width='50'><tr><td></td><td></td></tr></table>";
      }
    }, 500);
    //
    var S = setInterval(function() {
      document.getElementById("display3").innerHTML = "<table
id='myTable3'></table>";
      var data = JSON.parse(background.httpsposttable); // Display Var
      if (typeof data[0] !== "undefined"){
        TableMaker(data[0],"myTable3");
      }
      else{
        document.getElementById("display3").innerHTML = "<table
id='myTable3'><col width='50'><col
width='50'><tr><td></td><td></td></tr></table>";
      }
    }, 500);
    var T = setInterval(function() {
      document.getElementById("display4").innerHTML = "<table
id='myTable4'></table>";
      var data = JSON.parse(background.searchtable); // Display Var
      if (typeof data[0] !== "undefined"){
        TableMaker(data[0],"myTable4");
      }
      else{
        document.getElementById("display4").innerHTML = "<table
id='myTable4'><col width='50'><col
width='50'><tr><td></td><td></td></tr></table>";
      }
    }, 500);
    var X = setInterval(function() {
        document.getElementById("display8").innerHTML = "<table
id='myTable8'></table>";
        var data = background.allowed_urls; // Display Var
        if (typeof data[0] !== "undefined"){
          MakeColumn(data,"myTable8");}
        else{
         document.getElementById("display8").innerHTML = "<table
id='myTable8'><col width='50'> <col
width='50'><tr><td></td><td></td></tr></table>";
        }
    }, 500);
    var Y = setInterval(function() {
        document.getElementById("display9").innerHTML = "<table
id='myTable9'></table>";
        var data = background.blocked_urls; // Display Var
        if (typeof data[0] !== "undefined"){
```

```
          console.log("woppah")
          MakeColumn(data,"myTable9");}
        else{
         document.getElementById("display9").innerHTML = "<table
id='myTable9'><col width='50'> <col
width='50'><tr><td></td><td></td></tr></table>";
        }
    }, 500);
  }
  catch(err){
    console.log("Warning: Cannot Print Database that is empty.")
  }
};

function TableMaker(stuff,id) {
  var content = stuff.columns[0];
  var table = document.getElementById(id);
  var header = table.createTHead();
  var headerRow = header.insertRow(0);
  var numCols = stuff.columns.length;

  for(var i=0; i < numCols; i++)
  {
      var cell = headerRow.insertCell(i);
      cell.innerHTML = "<b>" + stuff.columns[i] + "</b>";
  }

  var numRows = stuff.values.length;

  for(var j=0; j < numRows; j++)
  {
      var row = table.insertRow();
      for(var i=0; i < numCols; i++)
          {
          var cell1 = row.insertCell(-1);
          cell1.innerHTML = stuff.values[j][i];
        }
  }
}

function MakeColumn(basic_array, id) {
   var table = document.getElementById(id);
     for(var i=0; i < basic_array.length; i++)
     {
          var row = table.insertRow();
          var cell1 = row.insertCell(-1);
          cell1.innerHTML = basic_array[i];
     }
}

function DownloadFunction(){
    background.WriteDB();
    download(background.binaryArray, "sql.db", "text/plain");
}

function DeleteFunction(){
    background.DeleteDB();
```

```
}

//download.js v4.2, by dandavis; 2008-2017.
//[MIT] see http://danml.com/download.html for tests/usage
;(function(r,l){"function"==typeof
define&&define.amd?define([],l):"object"==typeof
exports?module.exports=l():r.download=l()})(this,function(){return function
l(a,e,k){function q(a){var h=a.split(/[:;,]/);a=h[1];var
h=("base64"==h[2]?atob:decodeURIComponent)(h.pop()),d=h.length,b=0,c=new
Uint8Array(d);for(b;b<d;++b)c[b]=h.charCodeAt(b);return new
f([c],{type:a})}function m(a,b){if("download"in d)return
d.href=a,d.setAttribute("download",n),d.className="download-js-
link",d.innerHTML="downloading...",d.style.display="none",document.body.appen
dChild(d),setTimeout(function(){d.click(),document.body.removeChild(d),!0===b
&&setTimeout(function(){g.URL.revokeObjectURL(d.href)},250)},66),!0;if(/(Vers
ion)\/(\d+)\.(\d+)(?:\.(\d+))?.*Safari\//.test(navigator.userAgent))return/^d
ata:/.test(a)&&(a="data:"+a.replace(/^data:([\w\/\-\+]+)/,"application/octet-
stream")),!window.open(a)&&confirm("Displaying New Document\n\nUse Save As...
to download, then click back to return to this
page.")&&(location.href=a),!0;var
c=document.createElement("iframe");document.body.appendChild(c),!b&&/^data:/.
test(a)&&(a="data:"+a.replace(/^data:([\w\/\-\+]+)/,"application/octet-
stream")),c.src=a,setTimeout(function(){document.body.removeChild(c)},333)}va
r g=window,b=k||"application/octet-
stream",c=!e&&!k&&a,d=document.createElement("a");k=function(a){return
String(a)};var
f=g.Blob||g.MozBlob||g.WebKitBlob||k,n=e||"download",f=f.call?f.bind(g):Blob;
"true"===String(this)&&(a=[a,b],b=a[0],a=a[1]);if(c&&2048>c.length&&(n=c.spli
t("/").pop().split("?")[0],d.href=c,-1!==d.href.indexOf(c))){var p=new
XMLHttpRequest;return
p.open("GET",c,!0),p.responseType="blob",p.onload=function(a){l(a.target.resp
onse,n,"application/octet-
stream")},setTimeout(function(){p.send()},0),p}if(/^data:([\w+-]+\/[\w+.-
]+)?[,;]/.test(a)){if(!(2096103.424<a.length&&f!==k))return
navigator.msSaveBlob?navigator.msSaveBlob(q(a),n):m(a);a=q(a),b=a.type||"appl
ication/octet-stream"}else if(/([\x80-\xff])/.test(a)){e=0;var c=new
Uint8Array(a.length),t=c.length;for(e;e<t;++e)c[e]=a.charCodeAt(e);a=new
f([c],{type:b})}a=a instanceof f?a:new
f([a],{type:b});if(navigator.msSaveBlob)return
navigator.msSaveBlob(a,n);if(g.URL)m(g.URL.createObjectURL(a),!0);else{if("st
ring"==typeof a||a.constructor===k)try{return
m("data:"+b+";base64,"+g.btoa(a))}catch(h){return
m("data:"+b+","+encodeURIComponent(a))}b=new
FileReader,b.onload=function(a){m(this.result)},b.readAsDataURL(a)}return!0}}
);
```

APPENDIX B

RAW DATA

Experiment 1 User1-GET URLS

url	count	normed	Class
a.rfihub.com	2	0.001333	O
a.scorecardresearch.com	3	0.002	A
aax.amazon-adsystem.com	5	0.003333	A
ad.afy11.net	3	0.002	A
ads.pubmatic.com	6	0.004	A
ads.rubiconproject.com	5	0.003333	A
ads.yahoo.com	2	0.001333	A
adserver-us.adtech.advertising.com	11	0.007333	A
ag.innovid.com	3	0.002	O
ajax.googleapis.com	3	0.002	O
ami-d.openx.net	4	0.002667	O
amplifypixel.outbrain.com	2	0.001333	O
api.bounceexchange.com	4	0.002667	O
api.ideafit.com	2	0.001333	H
api.sele.co	2	0.001333	O
api.viglink.com	6	0.004	O
as.casalemedia.com	18	0.012	A
asset.pagefair.com	3	0.002	O
assets.adobedtm.com	4	0.002667	O
assets.bounceexchange.com	9	0.006	O
b.monetate.net	3	0.002	O
b.scorecardresearch.com	54	0.036	A
b.sharethrough.com	3	0.002	O
bat.bing.com	6	0.004	O
beacon.krxd.net	31	0.020667	O
bid.intentiq.com	2	0.001333	O
brightcove.vo.llnwd.net	2	0.001333	O
c.amazon-adsystem.com	2	0.001333	A
c.betrad.com	4	0.002667	A
c.brightcove.com	2	0.001333	O
c5x8i7c7.ssl.hwcdn.net	10	0.006667	O
cdn-maf0.heartyhosting.com	12	0.008	O
cdn-maf1.heartyhosting.com	11	0.007333	O
cdn-maf2.heartyhosting.com	16	0.010667	O
cdn-maf3.heartyhosting.com	19	0.012667	O

cdn.doubleverify.com	2	0.001333	A
cdn.gigya.com	4	0.002667	O
cdn.ideafit.com	16	0.010667	H
cdn.krxd.net	23	0.015333	O
cdn.livefyre.com	3	0.002	O
cdn.viglink.com	3	0.002	O
cdn.yldbt.com	3	0.002	O
cdn01.smartling.com	3	0.002	O
cdn4.ideafit.com	4	0.002667	H
cdn5.ideafit.com	2	0.001333	H
cedexis-video-embed.ora.tv	17	0.011333	O
cm.g.doubleclick.net	9	0.006	A
cti.w55c.net	2	0.001333	O
d1v3n981s5f4uj.cloudfront.net	3	0.002	O
datacloud-us-east-1.tealiumiq.com	2	0.001333	O
datacloud.tealiumiq.com	3	0.002	O
dpm.demdex.net	2	0.001333	O
e.nexac.com	40	0.026667	O
eb2.3lift.com	4	0.002667	O
ebayinc.demdex.net	4	0.002667	S
edge.quantserve.com	3	0.002	O
entitlements.jwplayer.com	5	0.003333	O
events.bounceexchange.com	4	0.002667	O
f.monetate.net	3	0.002	O
f.ora.tv	10	0.006667	O
fastlane.rubiconproject.com	14	0.009333	O
fitnessmagazine.mdpcdn.com	24	0.016	H
fls-na.amazon.com	26	0.017333	S
fonts.googleapis.com	3	0.002	O
geo-um.btrll.com	2	0.001333	O
gha.ebay.com	4	0.002667	S
googleads.g.doubleclick.net	10	0.006667	A
gscounters.us1.gigya.com	3	0.002	O
gum.criteo.com	2	0.001333	O
hpr.outbrain.com	4	0.002667	O
i.cdn.cnn.com	7	0.004667	N
i.cdn.turner.com	6	0.004	O
i.ebayimg.com	15	0.01	S
i.imgur.com	66	0.044	E
i.jsrdn.com	5	0.003333	O
i.w55c.net	2	0.001333	O
i.yldbt.com	6	0.004	O
i1.ebayimg.com	13	0.008667	S

i2.cdn.cnn.com	18	0.012	N
i2.ebayimg.com	15	0.01	S
i3.ebayimg.com	8	0.005333	S
ia.media-imdb.com	24	0.016	E
ib.3lift.com	3	0.002	O
ib.adnxs.com	7	0.004667	O
image2.pubmatic.com	4	0.002667	O
image6.pubmatic.com	2	0.001333	O
images.fitnessmagazine.mdpcdn.com	37	0.024667	H
images.outbrain.com	20	0.013333	O
imasdk.googleapis.com	14	0.009333	O
img1.zergnet.com	2	0.001333	E
img2.zergnet.com	5	0.003333	E
img4.zergnet.com	2	0.001333	E
imgur.com	2	0.001333	E
ir.ebaystatic.com	67	0.044667	S
jadserve.postrelease.com	2	0.001333	O
js.moatads.com	4	0.002667	A
jslog.krxd.net	2	0.001333	O
jwpltx.com	8	0.005333	O
kr.ixiaa.com	2	0.001333	O
l.ooyala.com	2	0.001333	O
log.dmtry.com	3	0.002	O
log.outbrain.com	10	0.006667	O
match.adsrvr.org	5	0.003333	A
metrics.brightcove.com	5	0.003333	A
ml314.com	2	0.001333	O
native.sharethrough.com	5	0.003333	O
odb.outbrain.com	4	0.002667	O
optimized-by.rubiconproject.com	2	0.001333	O
p.imgur.com	4	0.002667	E
p.rfihub.com	3	0.002	O
pagead2.googlesyndication.com	43	0.028667	A
ping.chartbeat.net	6	0.004	O
pixel.mathtag.com	2	0.001333	O
pixel.quantserve.com	6	0.004	O
pixel.rubiconproject.com	2	0.001333	O
platform.twitter.com	8	0.005333	M
player.ooyala.com	9	0.006	O
players.brightcove.net	4	0.002667	O
polo.feathr.co	3	0.002	O
pubads.g.doubleclick.net	5	0.003333	A
revee.outbrain.com	6	0.004	O

rover.ebay.com	7	0.004667	S
rtd.tubemogul.com	3	0.002	O
rules.quantcount.com	3	0.002	O
rva.outbrain.com	2	0.001333	O
s.imgur.com	26	0.017333	E
s0.2mdn.net	7	0.004667	O
s3-us-west-2.amazonaws.com	7	0.004667	O
secureir.ebaystatic.com	2	0.001333	S
showads.pubmatic.com	3	0.002	A
srv.main.ebayrtm.com	2	0.001333	S
srx.main.ebayrtm.com	2	0.001333	S
ssl.p.jwpcdn.com	30	0.02	O
static.chartbeat.com	4	0.002667	O
stats.aws.rubiconproject.com	8	0.005333	O
storage.outbrain.com	3	0.002	O
sync.graph.bluecava.com	5	0.003333	O
sync.tidaltv.com	3	0.002	O
tacoda.at.atwola.com	3	0.002	O
tag.bounceexchange.com	2	0.001333	O
tags.bluekai.com	5	0.003333	O
tags.tiqcdn.com	3	0.002	O
thumbs.ebaystatic.com	4	0.002667	S
tpc.googlesyndication.com	9	0.006	O
tps11007.doubleverify.com	10	0.006667	A
trk-sp.sele.co	2	0.001333	O
tru.am	2	0.001333	O
us-u.openx.net	11	0.007333	O
video.ami-admin.com	2	0.001333	O
vidthm.ora.tv	4	0.002667	O
w.sharethis.com	5	0.003333	O
widget-cdn.rpxnow.com	2	0.001333	O
widgets.outbrain.com	9	0.006	O
www.cnn.com	10	0.006667	N
www.ebay.com	4	0.002667	S
www.fitnessmagazine.com	4	0.002667	H
www.googleadservices.com	2	0.001333	A
www.googletagmanager.com	3	0.002	O
www.googletagservices.com	5	0.003333	O
www.i.cdn.cnn.com	18	0.012	N
www.ideafit.com	9	0.006	H
www.imdb.com	4	0.002667	E
www.imgur.com	2	0.001333	E
www.menshealth.com	43	0.028667	H

www.ora.tv	15	0.01	O
www.planetfitness.com	25	0.016667	H
www.stumbleupon.com	14	0.009333	E
www.zergnet.com	4	0.002667	E
x.bidswitch.net	3	0.002	O
z.cdn.turner.com	3	0.002	O

Website Type	**Count**	**Count**
Entertainment (E)	>50	155
Health (H)	>50	166
News (N)	>50	53
Business (B)	<100	0
Social Media (M)	<100	8
Learning Resource (L)	<100	0
Shopping (S)	>170	173
Ads/Analytics (A)	>170	211
Other (O)	>170	639
Total:		1405

Reduction:265->170
Removed traffic >.001

Experiment 1-User1-POST URLS

URL	Count	Class
http://ebayinc.demdex.net	4	S
http://stats.aws.rubiconproject.com	8	A
http://revee.outbrain.com	3	O
ttp://fastlane-adv.rubiconproject.com	1	A
http://tps11007.doubleverify.com	9	O
http://sync.graph.bluecava.com	1	O
http://www.imdb.com	2	E
http://datacloud.tealiumiq.com	1	O
http://datacloud-us-east-1.tealiumiq.com	2	O
http://api.viglink.com	5	O

Website Type	**Count**	**Count**
Entertainment (E)	<3	2
Social Media (M)	<3	0
Health (H)	<3	0
News (N)	<5	0
Learning Resource (L)	<5	0
Business (B)	<5	0

Shopping (S)		>3	4
Ads/Analytics (A)		>3	9
Other (O)		>3	21
	Total:	36	

Experiment1 User2-GET URLS

url	count	normed	Class
0914.global.ssl.fastly.net	17	0.008487	O
aa.agkn.com	4	0.001997	O
aax.amazon-adsystem.com	8	0.003994	A
acdn.adnxs.com	3	0.001498	A
ad.crwdcntrl.net	4	0.001997	A
ad.doubleclick.net	21	0.010484	A
ad.wsod.com	21	0.010484	A
ad.wsodcdn.com	8	0.003994	A
ads.pubmatic.com	10	0.004993	A
ads.rubiconproject.com	4	0.001997	A
ajax.googleapis.com	3	0.001498	O
aktrack.pubmatic.com	3	0.001498	O
ap.lijit.com	4	0.001997	O
api.content-ad.net	4	0.001997	A
as.casalemedia.com	8	0.003994	A
asset.pagefair.com	3	0.001498	O
assets.adobedtm.com	4	0.001997	O
assets.digitalfirstmedia.com	10	0.004993	O
b.monetate.net	3	0.001498	O
b.scorecardresearch.com	23	0.011483	A
beacon.krxd.net	20	0.009985	A
c.betrad.com	3	0.001498	A
c.newsinc.com	8	0.003994	O
cdn.doubleverify.com	6	0.002996	O
cdn.dynamicyield.com	3	0.001498	O
cdn.gigya.com	3	0.001498	O
cdn.krxd.net	13	0.00649	O
cdn.livefyre.com	3	0.001498	O
cdn.newsapi.com.au	4	0.001997	O
cdn.optimizely.com	3	0.001498	O
cdn.vidible.tv	8	0.003994	O
choices-or.truste.com	7	0.003495	O
choices.truste.com	4	0.001997	O

cloudfront-labs.amazonaws.com	4	0.001997	O
cm.g.doubleclick.net	9	0.004493	A
code.webyclip.com	7	0.003495	O
content.nasdaq.com	14	0.00699	B
cs.lkqd.net	3	0.001498	O
csi.gstatic.com	8	0.003994	O
csp.azureedge.net	8	0.003994	O
d1eoo1tco6rr5e.cloudfront.net	4	0.001997	O
d2lv4zbk7v5f93.cloudfront.net	4	0.001997	O
digitalfirst-d.openx.net	12	0.005991	O
dsum.casalemedia.com	13	0.00649	A
dt.adsafeprotected.com	3	0.001498	A
eb2.3lift.com	13	0.00649	O
ebayinc.demdex.net	4	0.001997	S
edge.quantserve.com	5	0.002496	O
f.monetate.net	3	0.001498	A
fastlane.rubiconproject.com	5	0.002496	O
fitnessmagazine.mdpcdn.com	24	0.011982	H
fls-na.amazon.com	37	0.018472	S
fonts.googleapis.com	9	0.004493	O
fonts.gstatic.com	9	0.004493	O
gha.ebay.com	8	0.003994	S
googleads.g.doubleclick.net	10	0.004993	A
googleads4.g.doubleclick.net	3	0.001498	A
hpr.outbrain.com	4	0.001997	O
i.cdn.cnn.com	7	0.003495	N
i.cdn.turner.com	6	0.002996	O
i.ebayimg.com	36	0.017973	S
i1.ebayimg.com	8	0.003994	S
i2.cdn.cnn.com	15	0.007489	N
i2.ebayimg.com	6	0.002996	S
ia.media-imdb.com	56	0.027958	E
ib.3lift.com	3	0.001498	O
ib.adnxs.com	29	0.014478	O
idsync.rlcdn.com	6	0.002996	O
image2.pubmatic.com	5	0.002496	O
images.fitnessmagazine.mdpcdn.com	37	0.018472	H
images.outbrain.com	10	0.004993	O
imedia-d.openx.net	6	0.002996	O
insight.adsrvr.org	12	0.005991	A
ir.ebaystatic.com	94	0.04693	S
jadserve.postrelease.com	18	0.008987	O
js.matheranalytics.com	14	0.00699	A

js.moatads.com	6	0.002996	A
launch.newsinc.com	4	0.001997	O
log.dmtry.com	9	0.004493	O
log.outbrain.com	7	0.003495	O
match.adsrvr.org	9	0.004493	A
nexus.ensighten.com	12	0.005991	O
ntvcld-a.akamaihd.net	6	0.002996	O
optimized-by.rubiconproject.com	3	0.001498	O
p.rfihub.com	3	0.001498	O
pagead2.googlesyndication.com	91	0.045432	A
ping.chartbeat.net	10	0.004993	O
pix04.revsci.net	3	0.001498	O
pixel.mathtag.com	4	0.001997	O
pixel.quantserve.com	9	0.004493	O
pixel.rubiconproject.com	8	0.003994	O
pixel.wp.com	8	0.003994	O
platform.twitter.com	6	0.002996	S
player.ooyala.com	9	0.004493	E
pubads.g.doubleclick.net	11	0.005492	A
px.dynamicyield.com	6	0.002996	O
revee.outbrain.com	4	0.001997	O
rover.ebay.com	16	0.007988	S
rtb-csync.smartadserver.com	3	0.001498	A
rtb.districtm.io	3	0.001498	O
rtd.tubemogul.com	5	0.002496	O
s.gravatar.com	5	0.002496	O
s.marketwatch.com	10	0.004993	B
s.ntv.io	3	0.001498	O
s0.2mdn.net	17	0.008487	O
secure-dcr.imrworldwide.com	6	0.002996	O
showads.pubmatic.com	5	0.002496	A
srv-2017-04-03-08.pixel.parsely.com	3	0.001498	O
ssum.casalemedia.com	3	0.001498	A
static.chartbeat.com	7	0.003495	O
stats.aws.rubiconproject.com	3	0.001498	O
stats.wp.com	3	0.001498	O
sts3.wsj.net	7	0.003495	O
subscription-assets.timeinc.com	4	0.001997	O
sync.mathtag.com	4	0.001997	O
t.brand-server.com	5	0.002496	O
tags.bluekai.com	10	0.004993	O
tags.tiqcdn.com	26	0.012981	O
theknow.denverpost.com	100	0.049925	N

thumbs.ebaystatic.com	6	0.002996	S
tiads.timeinc.net	3	0.001498	B
tpc.googlesyndication.com	37	0.018472	O
tps11017.doubleverify.com	12	0.005991	O
tps11027.doubleverify.com	8	0.003994	O
trk-sp.sele.co	4	0.001997	O
trk.vidible.tv	20	0.009985	O
uat-net.technoratimedia.com	5	0.002496	O
us-u.openx.net	24	0.011982	O
video.adaptv.advertising.com	5	0.002496	A
webservices.webspectator.com	9	0.004493	O
wfpscripts.webspectator.com	5	0.002496	O
widgets.outbrain.com	8	0.003994	O
www.bankrate.com	3	0.001498	B
www.cnn.com	9	0.004493	N
www.decenthat.com	6	0.002996	O
www.denverpost.com	31	0.015477	N
www.dianomi.com	6	0.002996	O
www.ebay.com	7	0.003495	S
www.fitnessmagazine.com	4	0.001997	H
www.fortune.idmanagedsolutions.com	7	0.003495	O
www.googletagmanager.com	3	0.001498	O
www.googletagservices.com	9	0.004493	O
www.i.cdn.cnn.com	17	0.008487	N
www.i.matheranalytics.com	27	0.01348	A
www.imdb.com	6	0.002996	E
www.nasdaq.com	37	0.018472	B
www.stumbleupon.com	14	0.00699	E
www.wtp101.com	6	0.002996	O
x.bidswitch.net	10	0.004993	A
z-ecx.images-amazon.com	3	0.001498	S
z.cdn.turner.com	4	0.001997	O

Website Type	**Count**	**Count**
Entertainment (E)	>66	85
News (N)	>66	179
Business (B)	>66	67
Learning Resource (L)	<=65	0
Health (H)	<=65	65
Social Media (M)	<=65	0
Shopping (S)	>200	231
Ads/Analytics (A)	>200	374
Other (O)	>200	671

Total:	1672	

395->151

Removed traffic >.001

Experiment 1 User2-POST URLS

URL	Count	Class
http://csi.gstatic.com	8	O
http://ebayinc.demdex.net	4	S
ttp://fastlane-adv.rubiconproject.com	1	A
http://www.imdb.com	2	E
http://px.dynamicyield.com	1	O
http://revee.outbrain.com	2	O
http://stats.aws.rubiconproject.com	1	A
http://tps11017.doubleverify.com	20	O
http://www.imdb.com	2	E
http://www.nasdaq.com	1	B

Website Type	**Count**	**Count**
Entertainment (E)	>=4	4
Shopping (S)	>=4	4
Other (O)	>=4	31
Business (B)	<5	1
Ads/Analytics (A)	<5	2
Learning Resource (L)	<5	0
Health (H)	<8	0
News (N)	<8	0
Social Media (M)	<8	0
Total:	42	

Experiment 1 User3-GET URLS

url	count	normed	class
0914.global.ssl.fastly.net	3	0.006565	O
aax.amazon-adsystem.com	1	0.002188	A
ad.crwdcntrl.net	1	0.002188	A
ad.doubleclick.net	2	0.004376	A
ads.rubiconproject.com	1	0.002188	A
adserver-us.adtech.advertising.com	2	0.004376	A
ae649e70f5e2adfc986ba6c3ffaa12ac4.profile.atl-m.cloudfront.net	1	0.002188	O
ajax.cloudflare.com	1	0.002188	O
aktrack.pubmatic.com	1	0.002188	O
as.casalemedia.com	1	0.002188	A

assets.adobedtm.com	4	0.008753	O
assets.digitalfirstmedia.com	5	0.010941	O
b.scorecardresearch.com	5	0.010941	A
bid.contextweb.com	1	0.002188	O
c.newsinc.com	2	0.004376	O
cdn.optimizely.com	1	0.002188	O
cdnjs.cloudflare.com	2	0.004376	O
cloudfront-labs.amazonaws.com	1	0.002188	O
cm.g.doubleclick.net	1	0.002188	A
count-server.sharethis.com	31	0.067834	O
d1eoo1tco6rr5e.cloudfront.net	1	0.002188	O
d2lv4zbk7v5f93.cloudfront.net	1	0.002188	O
deals.ebay.com	1	0.002188	S
digitalfirst-d.openx.net	7	0.015317	O
disqus.com	1	0.002188	M
dpm.demdex.net	1	0.002188	O
ebay-u.openx.net	1	0.002188	S
ebayinc.demdex.net	4	0.008753	S
ebayus-d.openx.net	1	0.002188	S
edge.quantserve.com	2	0.004376	O
edge.sharethis.com	1	0.002188	S
engine.adzerk.net	2	0.004376	A
fast.ebayinc.demdex.net	1	0.002188	S
fast.fonts.net	8	0.017505	O
fonts.googleapis.com	1	0.002188	O
fonts.gstatic.com	4	0.008753	O
gha.ebay.com	5	0.010941	S
googleads.g.doubleclick.net	8	0.017505	A
i.ebayimg.com	24	0.052516	S
i.stack.imgur.com	10	0.021882	E
i1.ebayimg.com	8	0.017505	S
i2.ebayimg.com	6	0.013129	S
i3.ebayimg.com	2	0.004376	S
ib.adnxs.com	2	0.004376	A
include.ebaystatic.com	1	0.002188	S
insight.adsrvr.org	3	0.006565	A
ir.ebaystatic.com	81	0.177243	S
jadserve.postrelease.com	11	0.02407	A
js.matheranalytics.com	2	0.004376	A
l.sharethis.com	1	0.002188	O
match.adsrvr.org	5	0.010941	A
nexus.ensighten.com	3	0.006565	O
ntvcld-a.akamaihd.net	3	0.006565	O

optimized-by.rubiconproject.com	2	0.004376	O
p.adsymptotic.com	1	0.002188	A
p.ebaystatic.com	2	0.004376	S
p.univide.com	1	0.002188	O
pagead2.googlesyndication.com	10	0.021882	A
pages.ebay.com	1	0.002188	S
ping.chartbeat.net	2	0.004376	O
pixel.quantserve.com	2	0.004376	O
pixel.wp.com	1	0.002188	O
platform.twitter.com	1	0.002188	M
pm.w55c.net	1	0.002188	O
pubads.g.doubleclick.net	2	0.004376	A
r.turn.com	1	0.002188	O
rover.ebay.com	10	0.021882	S
rtb.gumgum.com	1	0.002188	O
rtd.tubemogul.com	1	0.002188	O
s.gravatar.com	3	0.006565	O
s.ntv.io	1	0.002188	O
secureir.ebaystatic.com	2	0.004376	S
showads.pubmatic.com	2	0.004376	A
sp.adbrn.com	1	0.002188	A
srv.main.ebayrtm.com	2	0.004376	S
srx.main.ebayrtm.com	2	0.004376	S
stackoverflow.com	3	0.006565	L
static.ads-twitter.com	1	0.002188	A
static.adzerk.net	3	0.006565	A
static.chartbeat.com	1	0.002188	O
sync.1rx.io	2	0.004376	O
sync.srv.stackadapt.com	1	0.002188	O
sync.tidaltv.com	1	0.002188	O
t.co	1	0.002188	O
t.mookie1.com	1	0.002188	O
t.sharethis.com	1	0.002188	O
tags.bluekai.com	1	0.002188	O
tapestry.tapad.com	1	0.002188	O
thehackernews.com	8	0.017505	N
thumbs.ebaystatic.com	4	0.008753	S
tlg.mookie1.com	1	0.002188	O
tpc.googlesyndication.com	5	0.010941	O
us-u.openx.net	9	0.019694	O
uw.evm1.stackadapt.com	1	0.002188	O
w.sharethis.com	8	0.017505	O
www.colorado.edu	34	0.074398	L

www.decenthat.com	1	0.002188	O
www.denverpost.com	19	0.041575	N
www.ebay.com	4	0.008753	S
www.googletagservices.com	2	0.004376	A
www.i.matheranalytics.com	9	0.019694	A
www.thedenverpost.com	1	0.002188	N
x.bidswitch.net	2	0.004376	O
z.moatads.com	1	0.002188	A

Website Type	**Count**	**Count**
Social Media (M)	<5	2
Health (H)	<5	0
Business (B)	<5	0
Entertainment (E)	>=10	10
News (N)	>=10	28
Learning Resource (L)	>=10	37
Shopping (S)	>75	163
Ads/Analytics (A)	>75	79
Other (O)	>75	138
Total:		457

Reduction:104->104
Removed traffic >.001

Experiment 1 User3-POST URLS

URL	Count	Class
http://ebayinc.demdex.net	3	S
http://stackoverflow.com/	1	L

Experiment 1 User4-GET URLS

url	count	normed	Class
0914.global.ssl.fastly.net	3	0.014634	O
aa76b95f96e908ffbcef32a4c7c0e71b3.profile.lax-m.cloudfront.net	1	0.004878	O
aax.amazon-adsystem.com	2	0.009756	A
ad.crwdcntrl.net	1	0.004878	A
ad.doubleclick.net	2	0.009756	A
ads.yahoo.com	1	0.004878	A
ajax.googleapis.com	1	0.004878	O
assets.digitalfirstmedia.com	5	0.02439	O
b.scorecardresearch.com	4	0.019512	A
c.amazon-adsystem.com	1	0.004878	A

c.newsinc.com	2	0.009756	O
cdn.optimizely.com	1	0.004878	O
cloudfront-labs.amazonaws.com	1	0.004878	O
cm.g.doubleclick.net	1	0.004878	A
cmap.ox.ace.advertising.com	1	0.004878	A
cms.quantserve.com	1	0.004878	O
d1eoo1tco6rr5e.cloudfront.net	1	0.004878	O
d2lv4zbk7v5f93.cloudfront.net	1	0.004878	O
digitalfirst-d.openx.net	6	0.029268	O
disqus.com	1	0.004878	M
edge.quantserve.com	2	0.009756	O
i.imgur.com	66	0.321951	E
imgur.com	1	0.004878	E
insight.adsrvr.org	3	0.014634	A
jadserve.postrelease.com	12	0.058537	A
js.matheranalytics.com	2	0.009756	A
match.adsrvr.org	1	0.004878	A
nexus.ensighten.com	3	0.014634	O
ntvcld-a.akamaihd.net	3	0.014634	O
p.imgur.com	2	0.009756	E
p.rfihub.com	1	0.004878	O
pagead2.googlesyndication.com	5	0.02439	A
ping.chartbeat.net	1	0.004878	O
pixel.quantserve.com	2	0.009756	O
pixel.wp.com	1	0.004878	O
platform.twitter.com	2	0.009756	M
rules.quantcount.com	1	0.004878	O
s.gravatar.com	3	0.014634	O
s.imgur.com	15	0.073171	E
s.ntv.io	1	0.004878	O
static.chartbeat.com	1	0.004878	O
sync.mathtag.com	1	0.004878	O
sync.srv.stackadapt.com	1	0.004878	O
tpc.googlesyndication.com	1	0.004878	O
us-u.openx.net	6	0.029268	O
uw.evm1.stackadapt.com	1	0.004878	O
www.decenthat.com	1	0.004878	O
www.denverpost.com	19	0.092683	N
www.i.matheranalytics.com	7	0.034146	A
www.imgur.com	1	0.004878	E
www.thedenverpost.com	1	0.004878	N
z.moatads.com	1	0.004878	A

Website Type		Count	Count
Health (H)		<=20	0
News (N)		<=20	20
Business (B)		<=20	0
Shopping (S)		<10	0
Learning Resource (L)		<10	0
Social Media (M)		<10	3
Entertainment (E)		>30	85
Ads/Analytics (A)		>30	44
Other (O)		>30	53
	Total:	0	

Reduction:52->52
Removed traffic >.001
total requests: 205

Experiment 1 User4-POST URLS

URL	Count	Class

Website Type	Count	Count
Entertainment (E)	<=5	0
Health (H)	<=5	0
News (N)	<=5	0
Business (B)	<10	0
Shopping (S)	<10	0
Ads/Analytics (A)	<10	0
Learning Resource (L)	<30	0
Other (O)	<30	0
Social Media (M)	<30	0
Total:		0

Experiment 1 User5-GET URLS

url	count	normed	Class

Website Type	Count
Entertainment (E)	<10
Health (H)	<10
News (N)	<10
Business (B)	<30

Shopping (S)	<30
Ads/Analytics (A)	<30
Learning Resource (L)	<50
Other (O)	<50
Social Media (M)	<50
Total:	0

Experiment 1 User5-POST URLS

URL	Count	Class

Website Type	**Count**
Entertainment (E)	<10
Health (H)	<10
News (N)	<10
Business (B)	<=25
Shopping (S)	<=25
Ads/Analytics (A)	<=25
Learning Resource (L)	<30
Other (O)	<30
Social Media (M)	<30
Total:	0

Experiment 2-User1-GET URLS

url	count	normed	Class
imgur.com	1	0.008772	E
www.apple.com	22	0.192982	E
www.cnn.com	8	0.070175	N
www.ebay.com	1	0.008772	S
www.fitnessmagazine.com	4	0.035088	H
www.ideafit.com	9	0.078947	H
www.imdb.com	2	0.017544	E
www.menshealth.com	41	0.359649	H
www.muscleandfitness.com	1	0.008772	H
www.planetfitness.com	25	0.219298	H

Website Type	**Count**	**Count**
Entertainment (E)	<=80	25

Health (H)	<=80	80
News (N)	<=80	8
Business (B)	<25	0
Shopping (S)	<25	1
Ads/Analytics (A)	<25	0
Learning Resource (L)	<10	0
Other (O)	<10	0
Social Media (M)	<10	0
Total:		114

Reduction:10->10
Removed traffic >.001

Experiment 2 User1-POST URLS

url	count	normed	Class

Website Type	**Count**	**Count**
Health (H)	0	0
News (N)	0	0
Business (B)	0	0
Entertainment (E)	0	0
Shopping (S)	0	0
Ads/Analytics (A)	0	0
Learning Resource (L)	0	0
Other (O)	0	0
Social Media (M)	0	0
Total:		0

Experiment 2 User2-GET URLS

url	count	normed	Class
time.com	2	0.016807	B
www.cnn.com	8	0.067227	N
www.denverpost.com	27	0.226891	N
www.ebay.com	2	0.016807	S
www.imdb.com	2	0.016807	E
www.marketwatch.com	1	0.008403	B
www.nasdaq.com	69	0.579832	B
www.reuters.com	5	0.042017	B
www.thedenverpost.com	2	0.016807	N
www.time.com	1	0.008403	B

Website Type	Count	Count
Health (H)	<80	0
News (N)	<80	37
Business (B)	<80	78
Entertainment (E)	<5	2
Shopping (S)	<5	2
Ads/Analytics (A)	<5	0
Learning Resource (L)	<3	0
Other (O)	<3	0
Social Media (M)	<3	0
Total:		119

Reduction:10->10
Removed traffic >.001

Experiment 2 User2-POST URLS

url	count	normed	Class

Website Type	Count	Count
Health (H)	0	0
News (N)	0	0
Business (B)	0	0
Entertainment (E)	0	0
Shopping (S)	0	0
Ads/Analytics (A)	0	0
Learning Resource (L)	0	0
Other (O)	0	0
Social Media (M)	0	0
Total:		0

Experiment 2 User3-GET URLS

url	count	normed	Class
thehackernews.com	1	0.012658228	N
www.colorado.edu	35	0.443037975	L
www.denverpost.com	37	0.46835443	N
www.ebay.com	2	0.025316456	S
www.stackoverflow.com	1	0.012658228	L
www.thedenverpost.com	3	0.037974684	N

Website Type	Count	Count

Entertainment (E)	<20	0
Health (H)	<20	0
Business (B)	<20	0
Learning Resource (L)	<45	36
News (N)	<45	41
Shopping (S)	<45	2
Ads/Analytics (A)	<10	0
Other (O)	<10	0
Social Media (M)	<10	0
Total:		79

Reduction:6->6

Removed traffic >.001

Experiment 2 User3 POST URLS

url	count	normed	Class

Website Type	**Count**	**Count**
Health (H)	0	0
News (N)	0	0
Business (B)	0	0
Entertainment (E)	0	0
Shopping (S)	0	0
Ads/Analytics (A)	0	0
Learning Resource (L)	0	0
Other (O)	0	0
Social Media (M)	0	0
Total:		0

Experiment 2 User4-GET URLS

url	count	normed	class
imgur.com	1	0.058824	E
www.denverpost.com	14	0.823529	N
www.imgur.com	1	0.058824	E
www.thedenverpost.com	1	0.058824	N

Website Type	**Count**	
Entertainment (E)	<10	2
Health (H)	<10	0
Ads/Analytics (A)	<10	0
Business (B)	<15	0

Shopping (S)	<15	0
News (N)	<15	15
Learning Resource (L)	<5	0
Other (O)	<5	0
Social Media (M)	<5	0
Total:		17

Reduction:4->4
Removed traffic >.001

Experiment 2 User4-POST URLS

url	count	normed	Class

Website Type	**Count**	**Count**
Health (H)	0	0
News (N)	0	0
Business (B)	0	0
Entertainment (E)	0	0
Shopping (S)	0	0
Ads/Analytics (A)	0	0
Learning Resource (L)	0	0
Other (O)	0	0
Social Media (M)	0	0
Total:		0

Experiment 2 User5-GET URLS

url	count	normed	class

Website Type	**Count**	**Count**
Entertainment (E)	0	0
Health (H)	0	0
Ads/Analytics (A)	0	0
Business (B)	0	0
Shopping (S)	0	0
News (N)	0	0
Learning Resource (L)	0	0
Other (O)	0	0
Social Media (M)	0	0

Total: 0

Experiment 2 User5-POST URLS

url	count	normed	Class

Website Type	**Count**	**Count**
Health (H)	0	0
News (N)	0	0
Business (B)	0	0
Entertainment (E)	0	0
Shopping (S)	0	0
Ads/Analytics (A)	0	0
Learning Resource (L)	0	0
Other (O)	0	0
Social Media (M)	0	0
Total:		0