Personal Internet Privacy and Surveillance

Implementation and evasion of user tracking

Juha Jokinen

Master’s thesis
May 2018
Technology, communication and transport
Master's Degree Programme in Information Technology

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Description

Author(s) Jokinen, Juha	Type of publication Master’s thesis	Date May 2018
		Language of publication: English
	Number of pages 90	Permission for web publication: x

Title of publication

Personal Internet Privacy and Surveillance

Implementation and evasion of user tracking

Degree programme

Master's Degree Programme in Information Technology

Supervisor(s)

Rantonen, Mika

Assigned by

JAMK University of Applied Sciences / JYVSECTEC

Abstract

The modern Internet employs a vast number of different methods in order to track the user across sites. Information can be collected to enhance functionality but also for financial gain, as user data is a hot commodity in modern society. Data can be sold not only to advertisers but also to law enforcement and government organisations for control over people.

This thesis researches the reasons behind data collection and especially user tracking. It focuses primarily on the different tracking methods a normal Internet user may encounter. Using the information as a basis, these methods are constructed on a closed environment and then tested against most common evasion methods.

Demonstrative cases are formed as a result of the research, enabling the assignor to utilize them for commercial training of clients on the subject and to provide a basis for further academic research in the JYVSECTEC’s environment.

The results show how efficient each of the methods is and how it is not very difficult to avoid the methods by using just a simple set of tools readily available. As a conclusion, it can also be seen how data collection has become very common and efficient in the Internet. Users have also become more conscious of their privacy and current events have made it impossible not to be involved as even the biggest companies are revealed to have implemented data collection in a very intrusive manner.

Keywords/tags

Cyber security, privacy, internet technologies, data collection, Big Data

Miscellaneous

Kuvailulehti

Tekijä(t) Jokinen, Juha	Julkaisun laji Opinnäytetyö, ylempi AMK	Päivämäärä Toukokuu 2018
	Sivumäärä 90	Julkaisun kieli Englanti
		Verkkojulkaisulupa myönnetty: x

Työn nimi

Personal Internet Privacy and Surveillance

Implementation and evasion of user tracking

Tutkinto-ohjelma

Master's Degree Programme in Information Technology

Työn ohjaaja(t)

Rantonen, Mika

Toimeksiantaja(t)

JAMK University of Applied Sciences / JYVSECTEC

Tiivistelmä

Nykyajan internetissä käytetään useita menetelmiä käyttäjien seuraamiseen. Tietoa voidaan kerätä puhtaasti toiminnallisuuden parantamiseksi, mutta myös taloudellisen hyödyn tavoittelemiseksi. Käyttäjätiedoista kertyvä ns. ”Big Data” on kaupallisesti merkittävässä asemassa nykyaikaisessa yhteiskunnassa. Lisäksi dataa ei kaupata vain mainostajille, vaan myös lainvalvonnan ja valtiollisten toimijoiden käyttöön.

Opinnäytetyössä tutkittiin datan keruun ja varsinkin käyttäjien seurannan syitä sekä menetelmiä. Pääpaino oli käyttäjien seurannassa ja yleisimmissä niitä toteuttavissa menetelmissä. Lähtötietojen perusteella menetelmiä sovellettiin suljettuun ympäristöön ja niiltä puolustautumista testattiin yleisillä vapaasti saatavissa olevillla välineillä.

Menetelmistä syntyi ympäristö, jota toimeksiantaja voi käyttää edelleen kaupallisessa koulutuksessa. Ympäristöä voidaan myös hyödyntää aiheesta tehtävän jatkotutkimuksen pohjana sekä soveltaa erilaisia menetelmiä sen jatkokehittämiseksi.

Työn tulokset osoittivat, kuinka tehokkaita menetelmät ovat ja kuinka niiltä suojautuminen ei ole lopulta kovin vaikeaa, kun käytetään olemassa olevia työkaluja. Loppuyhteenvetona voidaan nähdä myös, kuinka yleiseksi datankeruu on muodostunut ja miten siitä on tullut olennainen osa nykyajan Internet-sivustoja. Käyttäjistä on myös tullut entistä valveutuneempia varsinkin, kun otetaan huomioon työn aikana havaitut tietoturvapoikkeamat ja niiden vaikutus yksittäisten ihmisten mielipiteisiin ja Internet-käyttäytymiseen.

Avainsanat (asiasanat)

Kyberturvallisuus, yksityisyys, internet-teknologiat, Big Data

Muut tiedot (salassa pidettävät liitteet)

Contents

Figures

Tables

ACRONYMS

API	Application Programming Interface
ARP	Address Resolution Protocol
BAT	Basic Attention Token
CSS	Cascading Stylesheets
DHCP	Dynamic Host Configuration Protocol
DOM	Document Object Model
DNS	Domain Name System
DNT	Do Not Track
EFF	Electronic Frontier Foundation
ESR	Extended Support Release
EXIF	Exchangeable image file format
GIF	Graphics Interchange Format
GPS	Global Positioning System
HSTS	HTTP Strict Transport Security
HTML	Hypertext Markup Language
HTTP	Hypertext Transport Protocol
IMEI	International Mobile Equipment Identity
IP	Internet Protocol
ISP	Internet Service Provider
JSON	JavaScript Object Notation
JYVSECTEC	Jyväskylä Security Technology
OAuth	Authentication
NAT	Network Address Translation

NIST	National Institute of Standards and Technology
PHP	PHP: Hypertext Preprocessor (recursive acronym)
PNG	Portable Network Graphics
P3P	Platform for Privacy Preferences Project
PII	Personally Identifiable Information
RFC	Request for Comments
RGB	Red Green Blue
RGCE	Realistic Global Cyber Environment
SLAAC	IPv6 Stateless Address Autoconfiguration
TOR	The Onion Router
URL	Uniform Resource Locator
WWW	World Wide Web
W3C	World Wide Web Consortium

1 Introduction

Alan Westin describes privacy as follows: "...individuals, groups or institutions have the right to control, edit, manage and delete information about themselves and to decide when, how and to what extent that information is communicated to others" (Westin 1967). Although Westin has a broad view compared to modern standards, the basic value of privacy still holds. Privacy should be considered as a right and not as a privilege. The people requiring privacy are usually labeled as persons who want to hide something - radicals, criminals, terrorists, even though they have done nothing wrong and just want to uphold their civil rights.

People who discard the need for privacy usually use the excuse "But I have nothing to hide". However, this is not ultimately true, as every person has something they do not wish to publish about themselves. People make judgemental calls all the time about what they tell about ourselves to others and in what way. They have opinions and views they do not want to air to everyone, they have secrets and have the right to control who they share the information with. The loss of privacy by the means of surveillance prevents the normal transfer of this information to the parties that have been deemed worthy of it unless it is also presented to the surveilling party. The surveillance does not even need to be present as the mere knowledge of being watched over is enough. This will essentially make people shut up and keep the information to themselves more easily and the knowledge or illusion of being watched will affect their social behaviour as well. The phenomena is explained in a Ted Talk by reporter Glenn Greenwald (Greenwald 2014), who explains the reasons why people want to have private moments and why large-scale monitoring may affect them more than is understood.

The goal of this thesis was to explain the reasons behind internet surveillance and user tracking, find out the most common methods of how it is done and implement these in the JYVSECTEC Realistic Global Cyber Environment (RGCE) in a way that they can be further used for demonstrations and training exercises.

2 Research questions

2.1 Research objectives

The idea to this thesis is based on the writer's own interest to find out how individual persons are being tracked and monitored on the modern internet. Applying this information to JYVSECTEC Realistic Global Cyber Environment (RGCE) will allow the use of realistic cases for training and demo purposes. As it was quickly found out, documenting everything from the privacy point of view for tracking and monitoring became much too large a task for this thesis; hence the following main objectives were selected for the implementation and documentation:

- What are the motives behind tracking?
- What are the general methods of implementing user tracking on internet and how they work?
- How can these methods be implemented and used in the RGCE?
- What can the user do to evade these methods and how effective evasion is?

2.2 Research methods

As the subject is highly technical and relies heavily on technical implementation, qualitative research methods were selected for thesis. Research starts with documenting the different methods of web-based tracking and data collection. On the side, methods used in the operating system or network-level are also gathered. Most of this work is done with the help of the internet since there is no collective documentation elsewhere on how this is done.

Next, a set of evasion methods commonly used for the sake of maintaining privacy and removing or blocking tracking elements from web are selected for further study. These will probably be mostly web browser plugins based on whitelists, so an interest is taken on how these methods will cope in RGCE where the implementation differs somewhat from a real-world scenario.

Tracking methods are then implemented in the RGCE using readily available applications or code snippets as the author is no a programmer. Most of the methods are meant to be inserted into web pages so a subset of pages in RGCE is selected for testing.

Then, the different tracking methods are tested with the selected set of browsers and plugins, evaluating how easy or hard it is for the user to avoid them. The results will show how efficient different methods are in establishing an identity for the tracked user, and also how easy these methods are to implement in different sites. From the user's perspective it will be conducted what are the most efficient tools for avoiding these types of tracking methods and how easy or hard they are to use.

3 Privacy issues

3.1 Privacy and Security

Security and privacy are usually considered to coexist. Locking one's house makes one feel both secure and private, however, this is not necessarily true when using the Internet. Being secure in IT means one cannot be attacked with a malicious intent, and this may give the user the false feeling of being private, even though he/she can still be tracked in many ways not considered malicious by security vendors. Anti-viruses, firewalls, malware scanners and such do nothing for privacy, except maybe by removing some malicious tracking cookies or spyware. They do not prevent tracking cookies, browser fingerprinting or operating system telemetry just like locked doors do not prevent someone from eavesdropping or monitoring when a person is at home or not.

Security and privacy are not mutually exclusive, and it should not be debated on whether security or privacy is needed. Both are needded, however also liberty is needed to choose and control on how a person is being secured. (Schneier 2006a) Governmental organisations want to invade privacy with the pretext of hunting terrorists and criminals, however, this should not be used as an excuse to weaken one's rights for privacy. Even if the methods used by these organisations are secure and the data acquired is kept only as long as necessary, the user has lost the ability to control the data collected from. In addition, plenty of privacy infringements or data leaks happen not with malicious intent but out of the sheer thoughtlessness and carelessness when handling private or sensitive data. (Solove 2004)

3.2 Privacy and Anonymity

The Internet enables the individual to become anonymous or at least pseudo-anonymous with the help of nicknames and multiple accounts, which can be totally irrelevant to one’s real self. This makes it easy to voice opinions without being afraid of repercussions from friends, family, colleagues or authorities. However, being anonymous or having separate accounts does not provide privacy. A person’s actions can still be tracked page-by-page and new techniques make it possible to distinquish the user’s unique fingerprint from the thousands of similar anonymous accounts or page visits. Actions can be correlated to each other, accounts linked to each other and it only needs one of these accounts with links to the user’s real persona for the whole anonymity to lose its meaning. Of course, this requires vast resources and a heavy reason to be implemented; however having privacy through full anonymity is just an illusion.

Too much anonymity can eventually lead to a system failure. Wikipedia, eBay and many other sites use pseudo-anonymous accounts where the user does not have to publish personal information any more than it is necessary; however, this still makes the users accountable and reputable for their actions inside these systems. This also makes it easier to trust users when one can put a name or an identity on them, albeit using a pseudonym. Without the aliases and nicknames, anyone could sabotage a Wikipedia page or create scam purchases or transactions in eBay. "Privacy can only be won by trust, and trust requires persistent identity, if only pseudo-anonymously." (Kelly 2006)

3.3 Ephemeral Conversations

One huge concern for privacy on the Internet is the disability to forget things. There is no common and sure way to delete data once it has been on the Internet. One cannot have ephemeral conversations, where the conversation or data transfer only takse place once and is totally forgotten after that. Anything that can be actively monitored can also be recorded in some way, and this also applies to the Internet where conversations are essentially exchanges of packet data. Even if the data is protected with the best encryption scheme possible, it is still possible to record every packet in the conversation and store it for later times when maybe the encryption becomes obsolete and can be cracked. Even if the information given to a service is irrelevant to the user today or the service provider pledges to protect their user’s privacy, who guarantees this will always be the case? Big companies can change their privacy policy any time they want or the individual may later regret what they have said and wants to take it back, however, the data still remains the same. (Schneier 2006b)

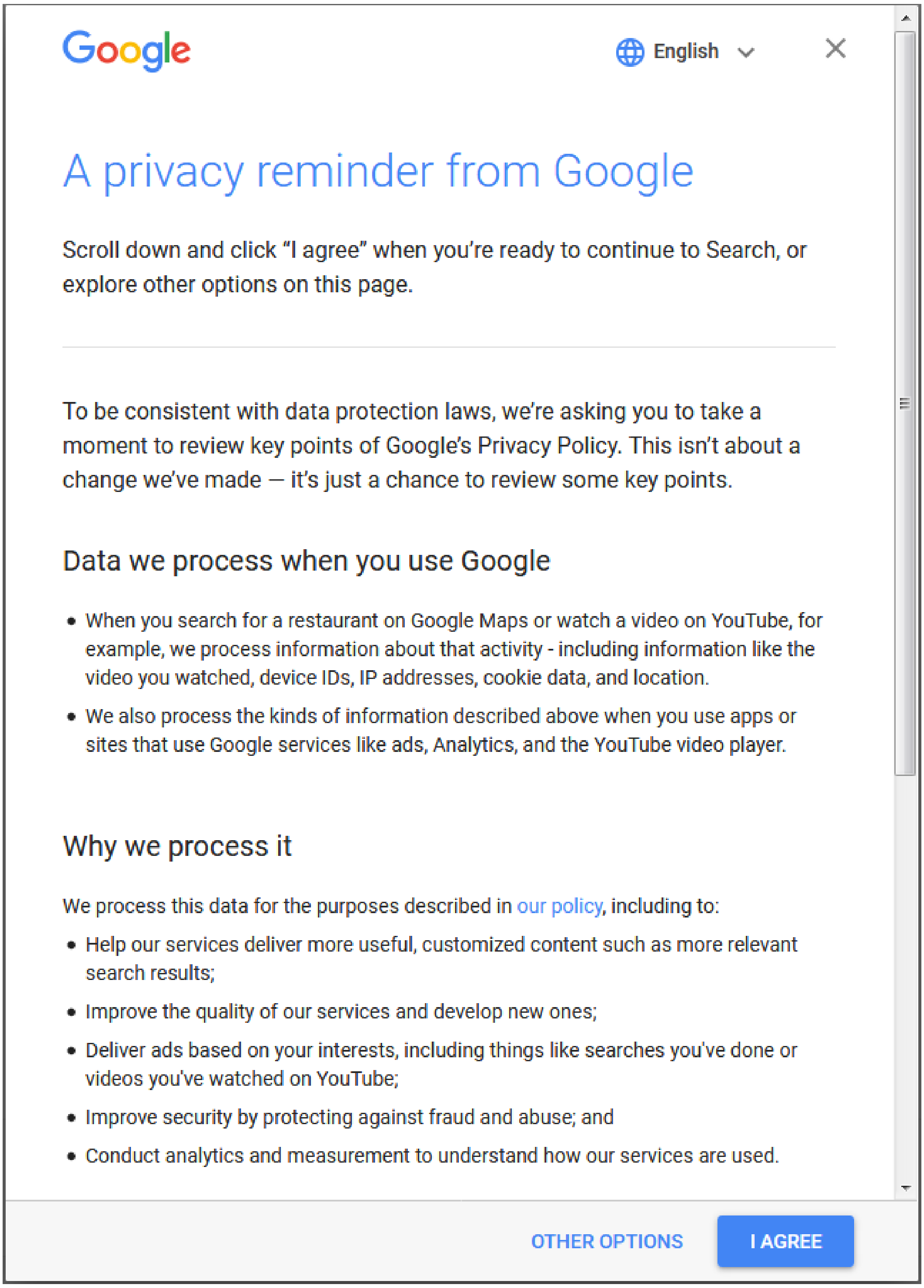
The automatic collection of conversations is in nature very impersonal, as it is not a human who collects and observes the information, but a machine or a complex system. This makes people accept it more easily as it is considered to be less invasive. The "I don't have any secrets" way of thinking can also be considered a wrong starting point to privacy. Daniel J. Solove (Solove 2004) calls this "security paradigm" and points out that privacy can still be infringed even if the information revealed is considered to be public knowledge.

3.4 Data collection and loss of control

The basic and most debated privacy issue with the Internet is the ability to collect data. Data can be collected both directly by analyzing conversations, images or even video files by applying e.g. keyword searches or facial recognition, or indirectly with the help of cookies, fingerprint techniques or metadata. It is trivial to record anything and everything the user does, as all of this can happen automatically and requires no intervention or a supervising user. The data collection can seem mostly harmless, such as analysis of purchase history through eBay for better marketing or analysis of browsing habits to tailor ads for the user. However, it can also include so-called Personally Identifiable Information (PII), which McCallister, Grance and Scarfone (2010) describe as "any information about an individual maintained by an agency, including (1) any information that can be used to distinguish or trace an individual‘s identity, such as name, social security number, date and place of birth, mother‘s maiden name, or biometric records; and (2) any other information that is linked or linkable to an individual, such as medical, educational, financial, and employment information."

This data collection happens in the background,is invisible to the user and usually there is no method of opting out. This violates the right to control how, when and what data is being collected from the user, the point of what Westin (Westin 1967) employed to describe privacy. As there also exists no method for the user to identify what data is recorded from his activity, there is also no way to distinguish if the data collected is relevant to the user’s interest or if the collecting party is acting with malicious intent. The user has effectively lost control of his/her data, who can access and use it and who it can be shared to. For the collecting party, the data is indexed, easily searchable and identifiable at any point.

The user may be told that data is collected. The European Union mandates that websites hosted in the member countries notify the user that the site uses cookies to track the user (European Parliament and the Council 2002). Later amendment (European Parliament and the Council 2009) also states that explicit consent from the user is required to use the cookies in the first place, as the previous version only required the user to be notified about them. In Finland, the legislation for PII data 523/1999 (Finlex 1999) requires the service providers to both ask for the users permission to collect sensitive information (Finlex 1999, section 8§) and to inform the user that they uphold a PII registry (Finlex 1999, section 10§). However, no universal legislation or directive for generic non-PII data collection exists. The terms of service usually include disclaimers that user activity may be collected and used to provide better services. Neither can the user opt-out some of these functions as that would prevent them from using the service. Tracking cookies are automatically accepted by using the website. After a while of using Google search, the user must agree to their privacy reminder (Figure 1).



A privacy reminder from Google

Scroll down and click "I agree" when you're ready to continue to Search, or explore other options on this page.

To be consistent with data protection laws, we're asking you to take a moment to review key points of Google's Privacy Policy. This isn't about a change we've made — it's just a chance to review some key points.

Data we process when you use Google

- When you search for a restaurant on Google Maps or watch a video on YouTube, for example, we process information about that activity - including information like the video you watched, device IDs, IP addresses, cookie data, and location.
- We also process the kinds of information described above when you use apps or sites that use Google services like ads, Analytics, and the YouTube video player.

Why we process it

We process this data for the purposes described in our policy, including to:

- Help our services deliver more useful, customized content such as more relevant search results;
- Improve the quality of our services and develop new ones;
- Deliver ads based on your interests, including things like searches you've done or videos you've watched on YouTube;
- Improve security by protecting against fraud and abuse; and
- Conduct analytics and measurement to understand how our services are used.

Figure 1. Google privacy reminder

A great deal of the control provided is also illusory. The user is given the option to limit access to their data or promises on data privacy for now, however, the service provider can dictate ultimately how and when to utilize data collection and who they share the data with. The terms of service can easily be changed, and most users will not notice any changes in their day-to-day activity. (Schneier 2006c)

3.5 Data brokers and reuse of data

Data brokers are the aggregators of collected personal data. They get feeds from industry customer systems, direct marketing, questionnaires, credit bureau information and governmental public records. The data is combined, correlated to dossiers and sold to marketers and ad networks. The data can consist of demographic data, lists of purchases or any kind of PII or non-PII data that can be combined together. (Schneier 2015)

Another issue with reusing second-hand data is that it was not collected for the application, and discrepancies may occur. For example, a data collected from a political forum can be used to predict the outcome of election; however, using the same dataset to rule out extremist criminal activity or terrorism will trigger plenty of false positives. The data corruption rate in marketing databases is very high since users can fill out feedback forms seemingly randomly, so the data is useless for deeper analytical purposes. (Schneier 2007)

Brokered data can ultimately also be used for governmental and law enforcement purposes. However, extra care should be taken when making decisions using a subset of personal information. Just as one’s person or personality is not the sum of his/her actions and thoughts, one’s digital person is not the sum of the information collected from him/her. The information can be hand-picked or sanitized in ways that will distort the digital image describing the persons and if this is then used for decisions affecting their lives, mistakes can easily be made. (Solove 2004)

3.6 Correlation

As data brokers can gather virtually endless amounts of data, they can also correlate them. The data gathered from simple demographic information combined with e.g. purchase history can be used to define the person’s sexual interest, risks of gambling or substance addiction or even to predict genetic diseases or mental issues before the person knows this him-/herself (Foster 2014).

Personally Identifiable Information can be used to distinguish specific individuals from any number of people. When this information is related to other PII data, it is considered linked or linkable. Linked data is considered to exist in the same system as the primary data, linkable data is remote; however, readily obtainable or public (McCallister, Grance & Scarfone 2010). This kind of linked data can be correlated and augment the information at hand about the individual.

Even if the data is anonymized or contains no PII entries, it can still be used in correlation with other data to distinguish individuals from each other. At the very lowest, a digital identity can be constructed using the correlation and separate pieces of user activity, possibly connecting several pseudonyms the user has on different services. On higher levels, the data can be used to identify a person by removing commons from the anonymous dataset and using so-called "micro-data" that is specific to the individual. Arvind Narayanan and Vitaly Shmatikov cross-referenced anonymous movie ratings of 500 000 Netflix subscribers to IMDb movie ratings and were able to connect the subscriber information to the corresponding IMDb user account by removing the top 500 watched movies from the equation. Similar methods can be employed on datasets where PII identifiers have been removed if the attacker has enough context and background information (Narayanan & Shmatikov 2008)

3.7 Termination of Authentication

When user wants to stop using the service, he/she can usually delete the account either via the account options or by creating a service request to the service provider. What happens to the data stored depends totally on the service provider and usually the user has little or no control on this. In Finland, the legislation 523/1999 subsection 26§ (Finlex 1999) allows anyone to request any organisation that maintains a PII database to present the dataset that has been collected from the individual person. The legislation subsection 29§ also allows the user to request a correction or deletion of the data. This legislation does not affect non-PII data, such as pseudonymous forums, social media or websites, or services provided from outside of Finland.

Another example of mandatory authentication is using webstores for purchases. Usually an account is required to do business, which includes the user’s name, address, contact and credit card information. Even if the user completes their purchase as a guest, this same data is still required to deliver the product and bill the user. However, when the customer has paid and received their product, there is no sure way to remove this information from the provider database. This is not only a security concern, where malicious parties can use this data for identity thefts, but also a privacy concern as a data leak from the provider can make the individuals purchase history public. (Schneier 2005)

4 Motivations behind data collection

4.1 Advertisers

The Internet is the most prominent place for advertising in the modern era. Ads can be placed not only on webpages but also on tablets and smartphone apps or other smart devices such as internet-enabled TVs. The competition on providing best ads is high, and advertisers have resorted to data collection on users to provide more relevant ads based on the browsing habits and interests of individual users. Several organisations exist only to provide ads and collect information from webpages.

DoubleClick is one of the oldest advertisement and data collection networks. Originally founded in 1996, it started as a small business providing primarily banner ads and evolved into a large-scale ad network until 2007, when Google acquired it (Citation needed). Many other networks have followed and the model has evolved to having *ad exchanges,* which can be used both for data agencies to send their customer data in and advertisers to send their ads and targets. This way the ad exchange or ad network generates targeted marketing. Targeted marketing relies heavily on user tracking (e.g. search patterns) to provide the most interesting ads to the target audience (Figure 2) (Soni 2017).

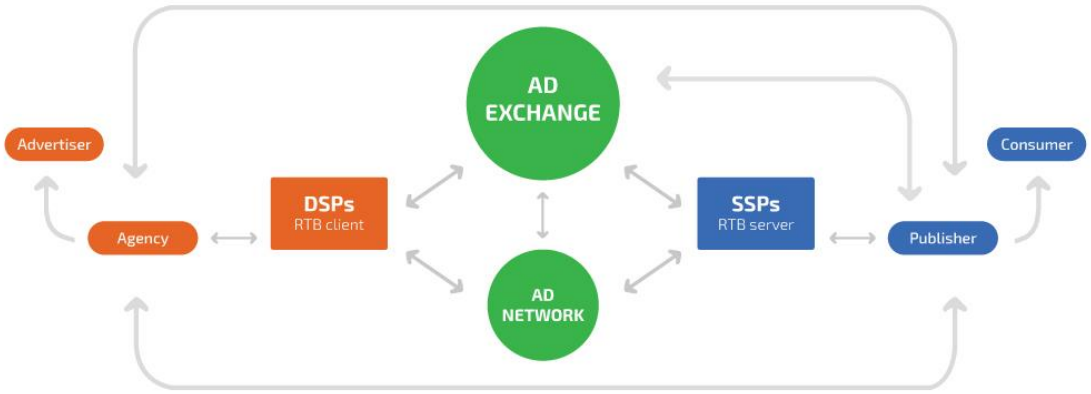


Figure 2. Digital advertising with ad exchanges/networks (Soni 2017)

Web stores can categorize clients as “high spenders” if they seem to be not interested in the prices of products, which allows the web store to raise the prices for these kind of people or offer coupons and sale prices to other, more price-sensitive clients. This is called “personalized pricing” or “price discrimination” and it can be easily done by tracking the user’s purchase and browsing habits in the e-store. This is generally considered in the least shady or even ethically wrong by people. (Borgesius 2017)

Other way to categorize is to use demographic data from data brokers to create marketable categories, e.g. “gullible seniors” list of persons could be used to push aggressive ads for medical care or living aids (Schneier 2015).

4.2 Organisation and workplace surveillance

Organisational data for workplaces is usually collected in the name of information security. It is deemed to be an important aspect of knowing what happens in the organisation’s network system, what data is sent and if company secrets are being leaked from the network. Some workplaces employ an invasive mindset of opening emails or logging the websites people use during the day. This might be rationalized under security or productivity.

Other reasons to collect data are mainly for advertising reasons as said earlier, however, the data can also be used for market research or betterment of customer services. Internet-facing webstores usually track the client’s purchases to provide better recommendations, coupons, sale vouchers and targeted ads to make and keep repeat customers. Loyalty programs track users heavily based on their purchase history and even other page visits if the data is available. (Schneier 2015)

4.3 Governments and public authorities

Governmental interest in data collection is usually for greater good. Law enforcement requires certain information on suspected crimes or terrorism. Either the government or authority itself can collect data using complex systems and public records but it can also be third-party data collected that has been mined. Financial data, medical records, religion and political interests, employee records can be used to profile persons of interest. ISPs are mandated to cooperate with law enforcement in many countries, and any organisation can be asked to release information in the event of a severe crime.

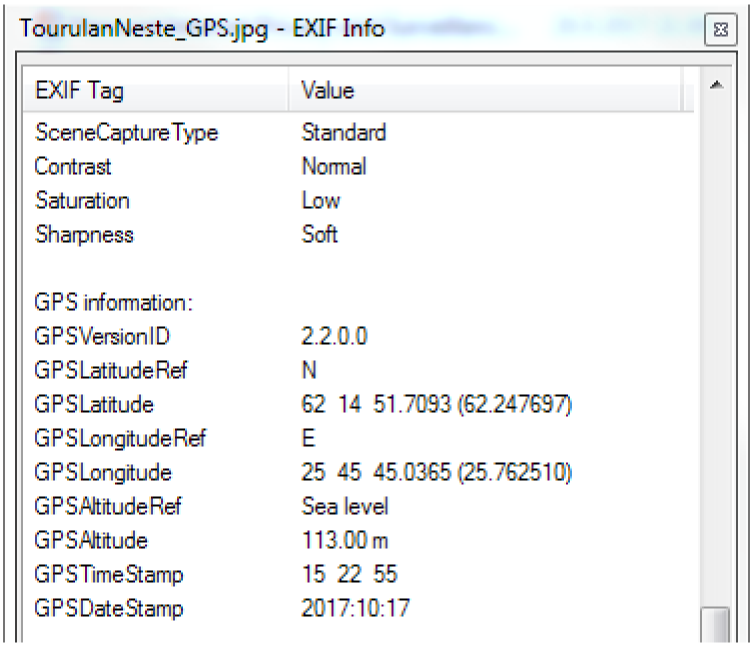
In 2012, the Motion Picture Association of America threatened to withdraw monetary support from the President of United States if they did not side with the Stop Online Piracy Act (Allen 2012). SOPA was considered one tool against Kim “Dotcom” Schmitz and his service Megaupload, which was officially a private file sharing site; however, it was mainly used to distribute pirated content (Couts 2012). In this case, one could say that implementing stronger data collection to weed out piracy benefits the government monetarily, even though the data could not be used for outright law enforcement purposes.

A large motivation for profiling people is so called "Social Sorting" where the service level offered to an individual is dictated by their identity derived from their social network or browsing activity. This kind of profiling has already been used e.g. in insurance business, where the insuree’s marital status and residence will affect the premium. (Lyon 2009) On the Internet, it is trivial to allocate better service effort for more eligible users, which may lead to a state of discrimination and privilege separation even if this was not the original intent. (Brown 2014)

The user profile information can also be used to mitigate risks, which can lead to false positives and unjustified detainment of suspected criminals (Lyon 2009). Public authorities can require social networks to hand over data from radical parties or suspected criminals. This data can be used to evaluate the further need to monitor the selected individual and maybe organise a wire-tap or some other, more direct method of surveillance. This is all rationalized with the pretext of fighting organised crime or terrorism, however, the results are inconclusive on how successful this kind of crime-prevention is in larger scale. (Schneier 2009)

4.4 Social networks

Social networks encourage the individual user to share as much information as possible. Facebook allows (and strongly suggests) the user to tag a friend in the picture or activity, allowing the social network to include this activity to the profiles of all participants. Tagging a place makes the network able to collect crude locations, however, including pictures with GPS coordinates will make the location more accurate. Many phone cameras allow the user to store the location where the pictures where taken in the image EXIF data as can be seen in Figure 1, and some models enforce this as the default setting.



TourulanNeste_GPS.jpg - EXIF Info

EXIF Tag	Value
SceneCaptureType	Standard
Contrast	Normal
Saturation	Low
Sharpness	Soft
GPS information:	
GPSVersionID	2.2.0.0
GPSLatitudeRef	N
GPSLatitude	62 14 51.7093 (62.247697)
GPSLongitudeRef	E
GPSLongitude	25 45 45.0365 (25.762510)
GPSAltitudeRef	Sea level
GPSAltitude	113.00 m
GPSTimeStamp	15 22 55
GPSDateStamp	2017:10:17

Figure 3. EXIF Location data in an image file

Networks aim to give individual stories maximum disclosure by making the users activity public by default. The user can limit the coverage using controls such as the Facebooks audience selector tool and friend groups (Figure 4) or Twitter protected tweets that are shown only to followers. However, these may not be easy to use and they do nothing to limit the access to the information by the social network itself.

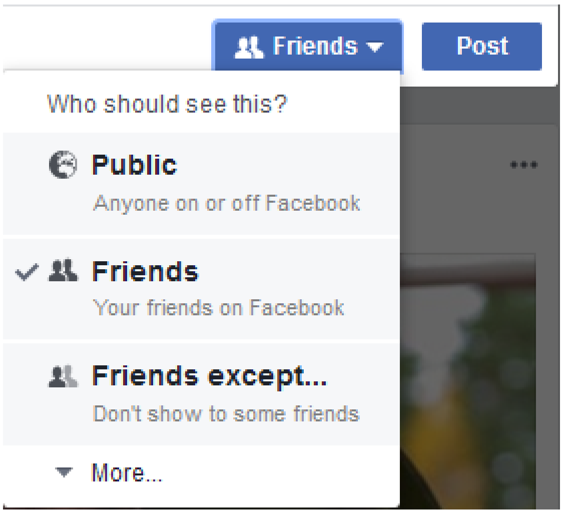


Figure 4. Facebook audience selector

Several sites provide social network authentication as the alternative way to use their services using OAuth. OAuth is an open protocol to allow authorization based on tokens received from the authorizing service, such as Facebook or Google. The user can connect their social media account to the service or app and continue using the service as long as they stay logged in to the social account as well. This creates a new option for the networks to gather information from the users, albeit limited to what services the user connects and when, however, the apps/services themselves can be overly intrusive and require information from the social account side. For example, Spotify by default will publish the user activity in Facebook Activity Log and the users Music page if connected via Facebook login. (Spotify Privacy Settings 2017)

4.5 The value of data

Data is currently the newest and probably the most valuable commodity on the market, and it is often compared to crude oil business in volume (The Economist 2017). The value of so-called Big Data lies in masses. Data from one individual user costs next to nothing in terms of collection, but when the data of thousands or millions of users is aggregated, it becomes an incredibly valuable asset in analytics. This in turn generates more revenue as more accurate marketing can be implemented. In the case of social media such as Facebook, the system feeds itself as more people want to join the system, when most of the people are already using it. (Schneier 2015).

5 User tracking methods

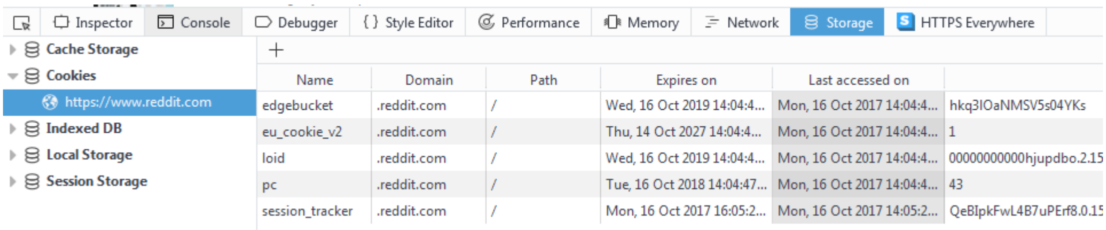
Data collection and user tracking are not mutually exclusive. Tracking the user's movements in or between webpages collects much information of his/her behavior on the Internet. Data collection can take place directly by using the data user has put up on the webpage (account information or purchase history for example) or indirectly and discreetly with the help of tracking components. As the goal of this thesis is to implement user-tracking components to JYVSECTECs RGCE, the main focus here will be on tracking methods and indirect information gathering.

5.1 Client-side and server-side tracking

This information collection or tracking can be implemented on either server or client-side. Server-side collection takes place on the server end, distinguishing users by IP addresses, user agents and other data that is naturally transmitted when the user requests a resource. This kind of tracking is harder to prevent; however, it is also easy to spoof and restrict what the server gets. Client-side data collection methods rely either on scripts and features set by the site or information exposed directly through the browser. This gives the tracking party vastly more information and some of the methods are very hard to avoid as blocking them may cause the browser or webpage to stop working correctly.

5.2 Cookies

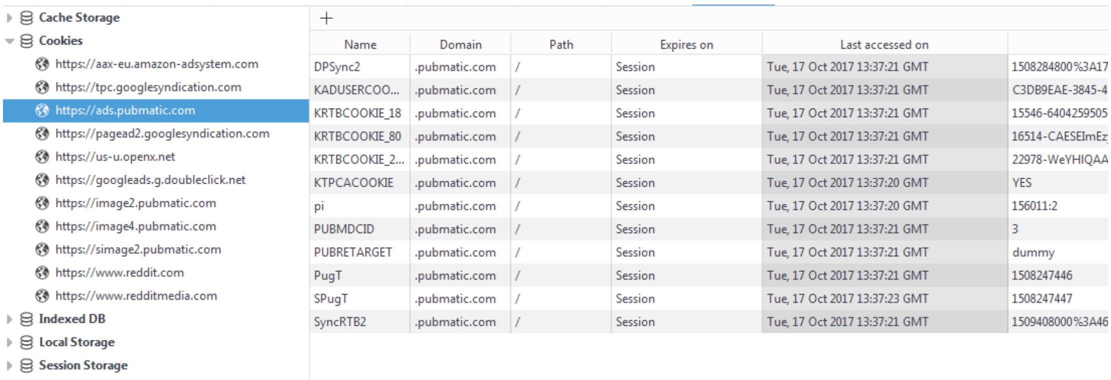
Cookies may be the oldest method to track user on the Internet, invented at Netscape in 1994. They work by storing site-specific variables in the browser cache of the user, making stateful handling possible on the client-side. When the user first accesses a page using cookies, the HTTP response includes a Set-Cookie header with the format variable=value. The information is then stored in the browser cache, either indefinitely or with a specific expiration date. The value stored in the cookie variable can be any kind of plaintext data, however, most often a session id is generated for the user. Subsequent visits to the page include the cookie data in the HTTP request header. The cookie can also be set and read using JavaScript API functions. The cookies are deleted when browser is closed or when the set expiration period is exceeded.



Name	Domain	Path	Expires on	Last accessed on	
edgebucket	.reddit.com	/	Wed, 16 Oct 2019 14:04:4...	Mon, 16 Oct 2017 14:04:4...	hkq3lOaNMSV5s04YKs
eu_cookie_v2	.reddit.com	/	Thu, 14 Oct 2027 14:04:4...	Mon, 16 Oct 2017 14:04:4...	1
loid	.reddit.com	/	Wed, 16 Oct 2019 14:04:4...	Mon, 16 Oct 2017 14:04:4...	00000000000hjupdbo.2.15
pc	.reddit.com	/	Tue, 16 Oct 2018 14:04:47...	Mon, 16 Oct 2017 14:04:4...	43
session_tracker	.reddit.com	/	Mon, 16 Oct 2017 16:05:2...	Mon, 16 Oct 2017 14:05:2...	QeBIpkFwL4B7uPErf8.0.15

Figure 5. Cookies set only for www.reddit.com in Private Browsing mode

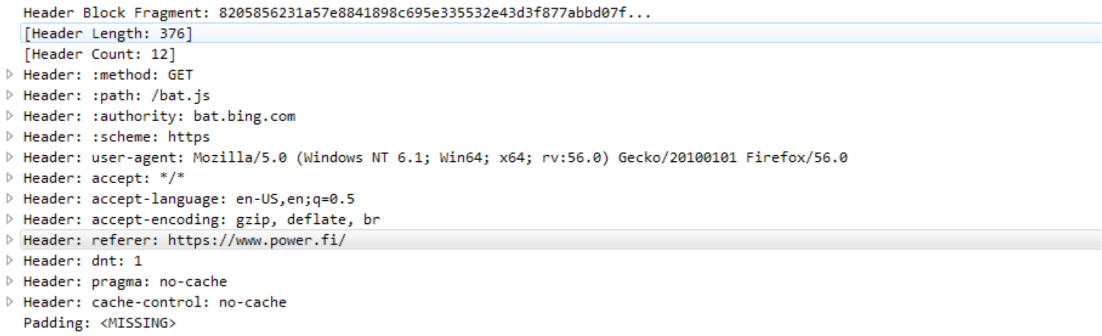
Cookies themselves are no problem to the individual privacy. They are used very often to retain site-specific settings such as the session information, shopping cart, language, forum theme or any other parameter that enhances the users browsing experience (Figure 5). However, some sites using advertisements, usage analytics or other reasons to track the user can use so-called third-party cookies. Third-party cookies are set in the HTTP headers when the advert or any other resource, e.g. the Facebook “like” button image, is requested from the third-party site. The variable is set to contain a unique id for each individual user. When the user browses to another site that uses the same third-party content, the cookie is then sent to the third-party site again, exposing this unique ID and user (Figure 6).


Cache Storage
Cookies
https://aax-eu.amazon-adsystem.com
https://tpc.googlesyndication.com
https://ads.pubmatic.com
https://pagead2.googlesyndication.com
https://us-u.openx.net
https://googleads.g.doubleclick.net
https://image2.pubmatic.com
https://image4.pubmatic.com
https://simage2.pubmatic.com
https://www.reddit.com
https://www.redditmedia.com
Indexed DB
Local Storage
Session Storage
+

Name	Domain	Path	Expires on	Last accessed on	
DPSync2	.pubmatic.com	/	Session	Tue, 17 Oct 2017 13:37:21 GMT	1508284800%3A17
KADUSERCOO...	.pubmatic.com	/	Session	Tue, 17 Oct 2017 13:37:21 GMT	C3DB9EAE-3845-4
KRTBCOOKIE_18	.pubmatic.com	/	Session	Tue, 17 Oct 2017 13:37:21 GMT	15546-6404259505
KRTBCOOKIE_80	.pubmatic.com	/	Session	Tue, 17 Oct 2017 13:37:21 GMT	16514-CAESEImEz
KRTBCOOKIE_2...	.pubmatic.com	/	Session	Tue, 17 Oct 2017 13:37:21 GMT	22978-WeYHIQAA
KTPCACOOKIE	.pubmatic.com	/	Session	Tue, 17 Oct 2017 13:37:20 GMT	YES
pi	.pubmatic.com	/	Session	Tue, 17 Oct 2017 13:37:20 GMT	156011:2
PUBMDCID	.pubmatic.com	/	Session	Tue, 17 Oct 2017 13:37:21 GMT	3
PUBRETARGET	.pubmatic.com	/	Session	Tue, 17 Oct 2017 13:37:21 GMT	dummy
PugT	.pubmatic.com	/	Session	Tue, 17 Oct 2017 13:37:21 GMT	1508247446
SPugT	.pubmatic.com	/	Session	Tue, 17 Oct 2017 13:37:23 GMT	1508247447
SyncRTB2	.pubmatic.com	/	Session	Tue, 17 Oct 2017 13:37:21 GMT	1509408000%3A46

Figure 6. Third-party cookies from different sources for www.reddit.com

The data collection may be augmented further with the use of referer information that exposes the original webpage URL to the tracking party. The original webpage can send referer HTTP header (Figure 7) when requesting resources or directly embed a HTTP GET parameter, e.g: <img src="image.jpg&referer=thissite.com">. This makes it trivial to collect information on browsing habits of the cookie holder, even if the user changes network for a portable device.

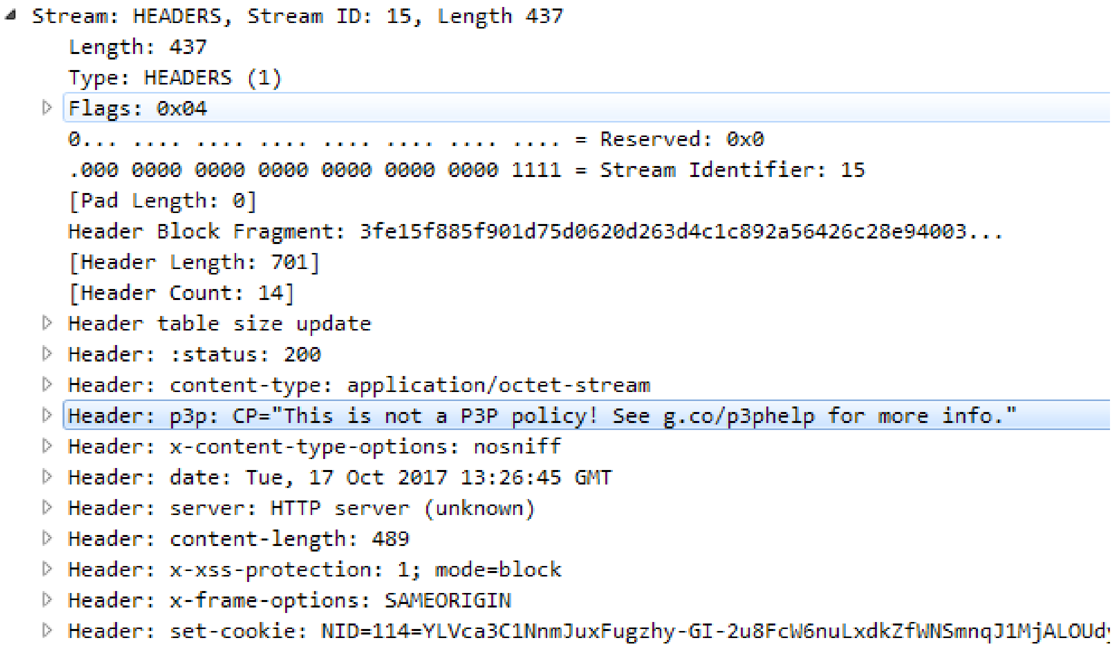


```
  Header Block Fragment: 8205856231a57e8841898c695e335532e43d3f877abbd07f...
  [Header Length: 376]
  [Header Count: 12]
▷ Header: :method: GET
▷ Header: :path: /bat.js
▷ Header: :authority: bat.bing.com
▷ Header: :scheme: https
▷ Header: user-agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:56.0) Gecko/20100101 Firefox/56.0
▷ Header: accept: */*
▷ Header: accept-language: en-US,en;q=0.5
▷ Header: accept-encoding: gzip, deflate, br
▷ Header: referer: https://www.power.fi/
▷ Header: dnt: 1
▷ Header: pragma: no-cache
▷ Header: cache-control: no-cache
  Padding: <MISSING>
```

Figure 7. HTTP Referer URL sent when requesting Microsoft Bing Advertisement tracking scripts from www.power.fi

One way to allow the user to opt-out of tracking is the use of AdChoices. AdChoices is a self-regulated program, originally established by the Digital Advertising Alliance (DAA). The program currently has over 200 participants and includes major organisations such as Facebook, Google and Microsoft. (YourAdChoices 2017) The user can visit http://youradchoices.com/ and set which ad networks he wishes to allow. This, however, requires the use of a third-party cookie, which makes it unusable when user sets his browser to deny third-party cookies. In addition, as the program is self-regulatory, nothing forces the advertisers to implement the AdChoices handling. (EFF 2017)

The WWW Consortium (W3C) initially tried to give the users the option to block the third-party cookies with Platform for Privacy Preferences Project (P3P), which is a protocol allowing the websites to send their privacy practices to the client browsers. The mechanism relied on user-agents and automation, making the user able to select his privacy settings once and automating the acceptance of the data collection mechanisms per-page, avoiding the need to read the privacy policy for all the visited sites. This kind of mechanism was already thought by Lawrence Lessig (2006), who describes a concept of negotiation between the user and the machine by pre-set rules of contract. The P3P project was suspended in 2006 as both browsers did not support it fully, and the web developers thought it was too complex to implement (Figure 8). (Platform for Privacy Preferences (P3P) Project 2007)



```
◢ Stream: HEADERS, Stream ID: 15, Length 437
    Length: 437
    Type: HEADERS (1)
  ▷ Flags: 0x04
    0... .... .... .... .... .... .... .... = Reserved: 0x0
    .000 0000 0000 0000 0000 0000 0000 1111 = Stream Identifier: 15
    [Pad Length: 0]
    Header Block Fragment: 3fe15f885f901d75d0620d263d4c1c892a56426c28e94003...
    [Header Length: 701]
    [Header Count: 14]
  ▷ Header table size update
  ▷ Header: :status: 200
  ▷ Header: content-type: application/octet-stream
  ▷ Header: p3p: CP="This is not a P3P policy! See g.co/p3phelp for more info."
  ▷ Header: x-content-type-options: nosniff
  ▷ Header: date: Tue, 17 Oct 2017 13:26:45 GMT
  ▷ Header: server: HTTP server (unknown)
  ▷ Header: content-length: 489
  ▷ Header: x-xss-protection: 1; mode=block
  ▷ Header: x-frame-options: SAMEORIGIN
  ▷ Header: set-cookie: NID=114=YLVca3C1NnmJuxFugzhy-GI-2u8FcW6nuLxdkZfWNSmnqJ1MjALOUd
```

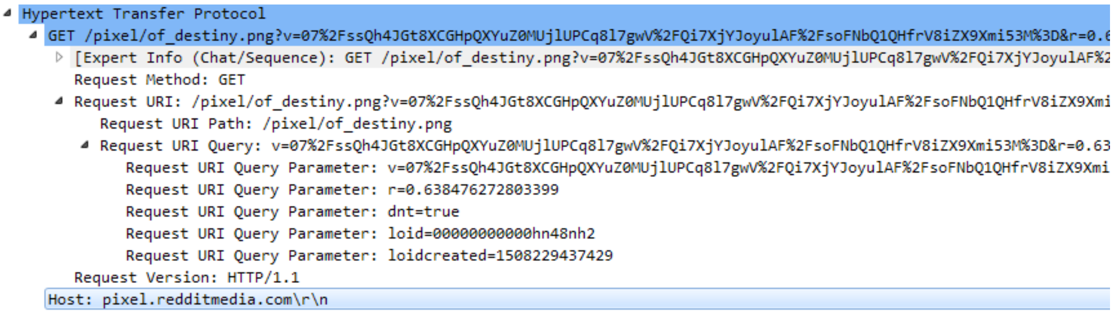
Figure 8. Google returns an informational message instead of a working P3P policy

After the P3P, the focus moved on to Do Not Track (DNT), which is a simple HTTP header message, implying the user’s wish they are not tracked on the webpage. The W3C established a working group for it in 2011, however, after a year the progress stalled and eventually the DAA pulled out of the project. The Electronic Frontier Foundation (EFF) has continued work on tightening the DNT policy and implemented their own browser add-on, Privacy Badger, for cases when the sites do not comply with the DNT header. DNT is implemented in all current browsers. (EFF 2017)

DNT has not been a success as there is no consensus on what kind of tracking it concerns. The header only implies if the user wants to disable tracking or not, it does not separate the methods used (such as cookies, advertising id or fingerprint) (Miners 2014). Advertisers have also raised controversy on the topic whether DNT or other tracking protection should be implemented by default. Microsoft tried to apply DNT as the default setting in Internet Explorer 10 from 2012 onwards, until they removed this default in IE10/IE11 2015 (Keizer 2015). During the time of writing this thesis, Apple has changed the behavior of the Safari browser to block third-party cookies by default, which again has raised controversy against them (Statt 2017). W3C is still working on the technique under the name Tracking Preference Expression, howeverm the work is still at recommendation level (W3C 2015).

5.3 Beacons

A web beacon or a web bug is a simple, possibly invisible resource placed on the webpage. Beacons can be transparent GIF or PNG images, scripts, HTTP IFRAME elements, or any other resource that is requested from the server. They can also be visible elements, such as advertisements or social media buttons (Facebook “like” button for example). When the requested beacon resource is downloaded from the server, it automatically leaves a log entry, which can contain the IP address, user-agent and referring URL from the webpage. Beacons can be used to track when a user opens a certain page or e-mail, as e-mails can be HTML encoded also. Many e-mail clients can be configured to block web content in order to disable beacon activity. Beacons can be coupled with HTTP cookies or URL queries to transmit more data when the resource is requested (Figure 9).



```
◢ Hypertext Transfer Protocol
  ◢ GET /pixel/of_destiny.png?v=07%2FssQh4JGt8XCGHpQXYuZ0MUjlUPCq8l7gwV%2FQi7XjYJoyulAF%2FsoFNbQ1QHfrV8iZX9Xmi53M%3D&r=0.6
    ▷ [Expert Info (Chat/Sequence): GET /pixel/of_destiny.png?v=07%2FssQh4JGt8XCGHpQXYuZ0MUjlUPCq8l7gwV%2FQi7XjYJoyulAF%2
      Request Method: GET
    ◢ Request URI: /pixel/of_destiny.png?v=07%2FssQh4JGt8XCGHpQXYuZ0MUjlUPCq8l7gwV%2FQi7XjYJoyulAF%2FsoFNbQ1QHfrV8iZX9Xmi
        Request URI Path: /pixel/of_destiny.png
      ◢ Request URI Query: v=07%2FssQh4JGt8XCGHpQXYuZ0MUjlUPCq8l7gwV%2FQi7XjYJoyulAF%2FsoFNbQ1QHfrV8iZX9Xmi53M%3D&r=0.63
          Request URI Query Parameter: v=07%2FssQh4JGt8XCGHpQXYuZ0MUjlUPCq8l7gwV%2FQi7XjYJoyulAF%2FsoFNbQ1QHfrV8iZX9Xmi
          Request URI Query Parameter: r=0.638476272803399
          Request URI Query Parameter: dnt=true
          Request URI Query Parameter: loid=00000000000hn48nh2
          Request URI Query Parameter: loidcreated=1508229437429
      Request Version: HTTP/1.1
    Host: pixel.redditmedia.com\r\n
```

Figure 9. Pixel beacon www.reddit.com, sending data with URL parameters.

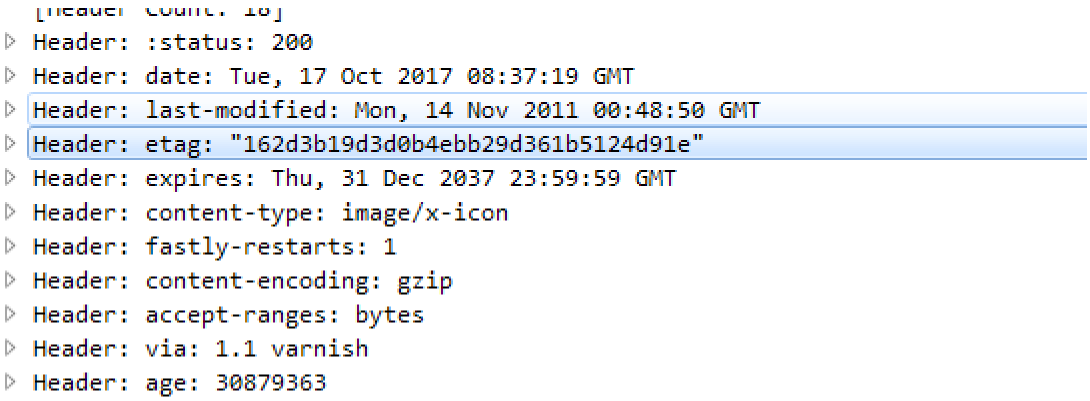
Mozilla Developers (MDN 2017a) have been experimenting in implementing a Beacon API, a common way of creating a beacon request to the server. This method uses a separate HTTP POST request and does not require any additional resource to be requested from the server. The API is supported in Google Chrome, Microsoft Edge, Mozilla Firefox and Opera browsers (MDN 2017a).

Facebook tried to implement its own beacon system in 2007, where user activity could be collected from the webpage and published automatically in the user’s activity feed. This promptly resulted in a class-action lawsuit as the users were not happy with the invasion in privacy, forcing Facebook to shut down their beacon implementation in 2009 (Telegraph 2009).

5.4 ETags

ETags (Entity tags) are a form of headers that can be used to differentiate a version of a certain resource. For example, an image file may have ETag associated to it at the first time it is requested from the server (Figure 10). Subsequent requests will include an HTTP Header "If-Match" or "If-None-Match" containing the ETag value. This way the server knows if the resource that is being requested has already been served. If the ETag value matches, server can return HTTP 304 Not Modified which makes the browser use the old resource from the cache. If the ETag does not match the one the server uses, the resource has been changed and will be sent again. (MDN 2017b)

ETags can also be used to track the user. The server can assign a pseudo-unique ETag to a resource served for the user, for example an image. Subsequent requests from the same user will include the same ETag and can be used to log the user. Instead of ETags, HTTP Last-Modified header value may also be used as it can hold any random string and does not necessarily need to be a date (Bujlow et al. 2015). Since these methods do not rely on cookies but the browser cache, they are impervious to the user clearing cookies on browser exit or blocking of third-party cookies. Full browser cache clearing, however, will remove the ETags or Last-Modified timestamps from the memory.



```
  [Header count: 18]
▷ Header: :status: 200
▷ Header: date: Tue, 17 Oct 2017 08:37:19 GMT
▷ Header: last-modified: Mon, 14 Nov 2011 00:48:50 GMT
▷ Header: etag: "162d3b19d3d0b4ebb29d361b5124d91e"
▷ Header: expires: Thu, 31 Dec 2037 23:59:59 GMT
▷ Header: content-type: image/x-icon
▷ Header: fastly-restarts: 1
▷ Header: content-encoding: gzip
▷ Header: accept-ranges: bytes
▷ Header: via: 1.1 varnish
▷ Header: age: 30879363
```

Figure 10. ETag and Last-Modified headers for the Reddit favicon file.

5.5 Other client-side mechanisms

Cookies are not the only method of placing tracking on client-side. Bujlow et al. (2015) describe multiple other mechanisms that can be used:

- JavaScript session variables can also be used to store up to 2Mb data.
- HTTP Basic Authentication. Based on the HTTP protocol, uses HTTP header information to send authentication parameters to server. Once user logs in, the basic authentication session stays in the browser cache until the browser is closed, making it possible to track the user. Limited to the same domain.
- URL query strings. HTTP GET parameters can be used to track the user even when switching domain (for example www.domain.com&trackingid=1234). Does not store information on the browser level, but using the same URL again will send the same parameters. Used mainly for referer tracking in search engines.
- Hidden form fields. A hidden form and fields may be injected into a webpage, transferring information in HTTP GET or POST request. If GET is used, works similar to URL query strings but when POST is used, data is sent in headers instead.
- Plugin storage. Both Adobe Flash and Microsoft Silverlight provide a way to store data in the client browser plugin cache. Flash Local Shared Objects are usually also called Flash Cookies, Supercookies or Zombie cookies. They can be recreated even after the browser deletes them and they are shared across browsers. Java JNLP Persistence-Service can also be used to store local data.
- Google Gears. Gears makes it possible to store data locally. User has to give his permission for Gears to store data so it is not well suited for data collection/tracking without the user knowing. Google Gears was discontinued in 2011
- window.name Javascript DOM property. All browsers support the commond Document Object Model (DOM) and this includes a property called window.name. This property can contain few megabytes of data. As the property is the same for all tabs in the same window, it can be used for tracking across various websites opened in separate tabs.
- HTML5 Local and Session storage. Local-storage can be used to store data permanently as it has no expiration set by default. Session storage works similarly but is emptied when the browser is closed.
- Web SQL Database and HTML5 IndexedDB. These are both mechanisms for creating a database in the client-side for easy structured data storage. The latter is a feature of HTML5 and has mainly replaced the former Web SQL.
- Internet Explorer userData storage can store data in XML format

It is also possible to use HTTP 301 or JavaScript redirection to make URL query string or third-party cookie usage easier. When the user accesses the page, he/she is redirected to another page first that sets the cookies or returns to the same page with the URL query string appended to the URL. As many browsers only deny setting but not reading third-party cookies, this only needs to be done once when the user first comes to the page. (Bujlow et al. 2015)

Since the web browser can render the page differently for different users, for example when the user is logged in, specially constructed timing attacks or pixel-stealing can be used to read the browser history or arbitrary graphics data from the webpage. (ibid., 2015)

5.6 Supercookies

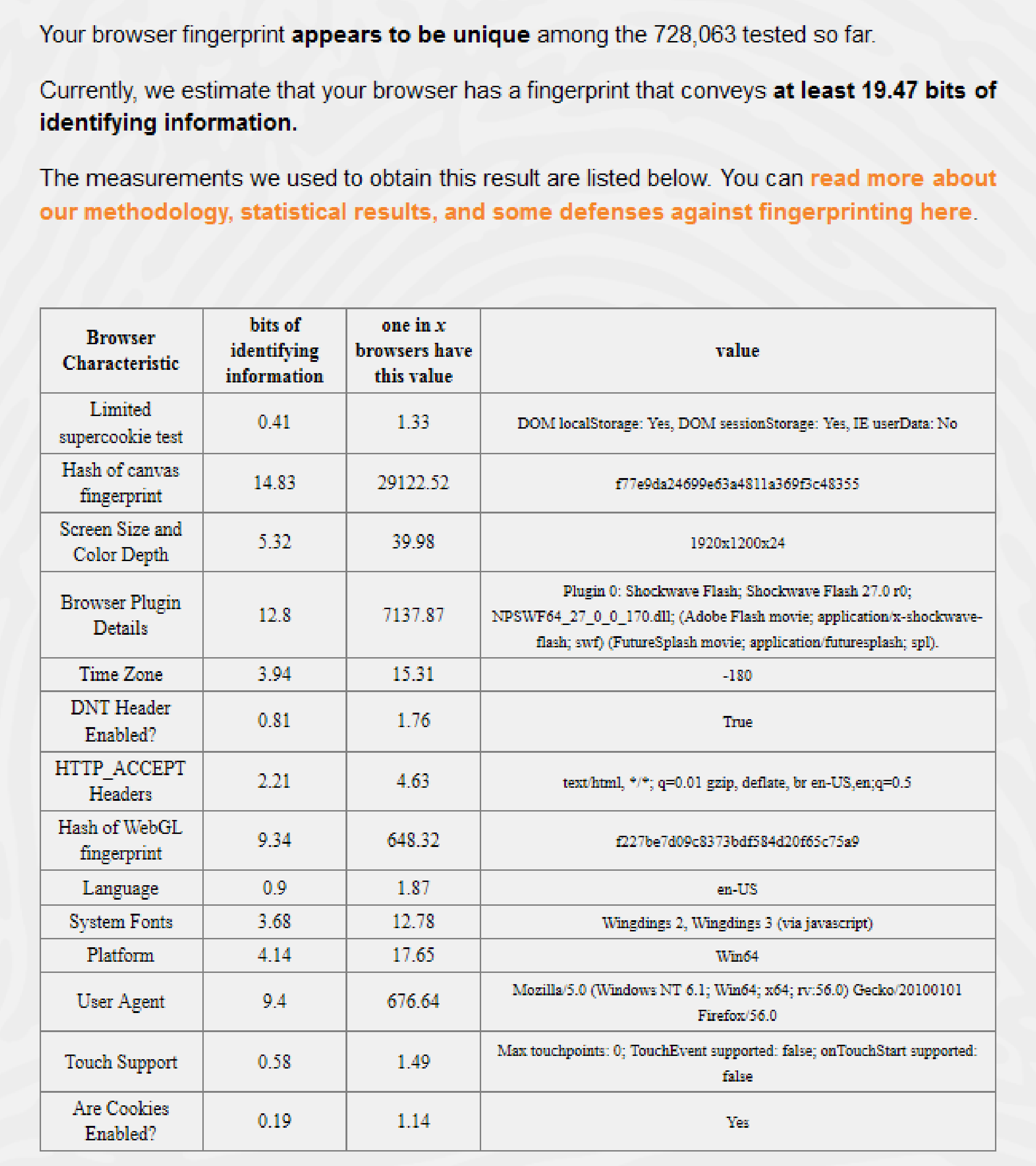
Since cookies are essentially tied to the domain, big corporations such as Microsoft and Google have a problem when they are using multiple different domains such as microsoft.com, live.com and bing.com. To solve this, they use Cookie Syncing, where the same cookie information is shared between multiple domains via so-called supercookies. Supercookies use some other mechanism to recreate the cookie information and pass it on to another domain. Microsoft creates supercookies with the use of unique identifiers for users embedded in ETags (Mayer 2011) and Google uses pixel beacons with Google User ID as the HTTP parameter (Google 2017a).

Evercookies, also called zombie cookies, are cookies that can be resurrected or recreated from scratch even after the cookie cache has been emptied. Multiple local storage types, such as HTML5 local and session storage, HTML5 IndexedDB, WebSQL, and ETags can be used to store data that would otherwise be lost. When using Flash cookies additionally to traditional cookies, the evercookies can even be rebuilt after change or reinstallation of a browser, since they are stored in the separate Flash Player plugin cache.

5.7 Fingerprinting

Fingerprinting is the method of collecting trivial bits of information and compiling these together to create a unique identifier or fingerprint that distinguishes the user from others. The information can be collected from many sources, such as network and geolocation information, device information such as operating system, screen size, available fonts or browser version and available plugins. None of these themselves are enough to identify a single user and as many web users will have similar devices (operating system and browser for example) there will be many very similar fingerprints generated. However, any single small change such as a different version of a plugin or device driver will be enough to make the fingerprint unique among millions.

The effect of fingerprinting can be easily demonstrated using the EFFs Panopticlick (Figure 11) which tests a base set of different ways to generate a fingerprint for the user (EFF 2018b). The test generated a unique fingerprint with ease for the authors PC, which is not a very standard issue and contains many plugins. Redoing the test with a company laptop yielded similar results.



Your browser fingerprint **appears to be unique** among the 728,063 tested so far.

Currently, we estimate that your browser has a fingerprint that conveys **at least 19.47 bits of identifying information.**

The measurements we used to obtain this result are listed below. You can **read more about our methodology, statistical results, and some defenses against fingerprinting here**.

Browser Characteristic	**bits of identifying information**	**one in *x* browsers have this value**	**value**
Limited supercookie test	0.41	1.33	DOM localStorage: Yes, DOM sessionStorage: Yes, IE userData: No
Hash of canvas fingerprint	14.83	29122.52	f77e9da24699e63a4811a369f3c48355
Screen Size and Color Depth	5.32	39.98	1920x1200x24
Browser Plugin Details	12.8	7137.87	Plugin 0: Shockwave Flash; Shockwave Flash 27.0 r0; NPSWF64_27_0_0_170.dll; (Adobe Flash movie; application/x-shockwave-flash; swf) (FutureSplash movie; application/futuresplash; spl).
Time Zone	3.94	15.31	-180
DNT Header Enabled?	0.81	1.76	True
HTTP_ACCEPT Headers	2.21	4.63	text/html, */*; q=0.01 gzip, deflate, br en-US,en;q=0.5
Hash of WebGL fingerprint	9.34	648.32	f227be7d09c8373bdf584d20f65c75a9
Language	0.9	1.87	en-US
System Fonts	3.68	12.78	Wingdings 2, Wingdings 3 (via javascript)
Platform	4.14	17.65	Win64
User Agent	9.4	676.64	Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:56.0) Gecko/20100101 Firefox/56.0
Touch Support	0.58	1.49	Max touchpoints: 0; TouchEvent supported: false; onTouchStart supported: false
Are Cookies Enabled?	0.19	1.14	Yes

Figure 11. Panopticlick results as tested at https://panopticlick.eff.org/

Another, even more accurate method to generate a unique fingerprint is canvas fingerprinting, which utilises the browsers drawable area, called canvas and JavaScript or some other script language. The canvas fingerprinting script draws invisible images containing a piece of text or WebGL imagery on the canvas and extracts the result image as a pixel data (Figure 12). This occurs in seconds or even milliseconds, without the user seeing anything on the webpage. As client machines can have differing fonts, browsers or graphic drivers, the resulting image can differ by mere pixels, making the fingerprint unique. (Acar et al. 2014)

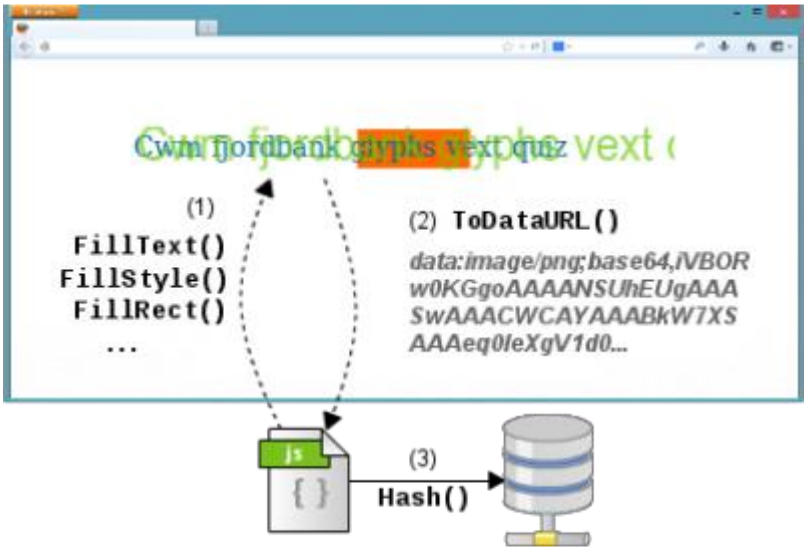


Figure 12. Canvas fingerprinting (Acar et al. 2014)

These mechanisms do not work very well for similar clients using the same base image such as organisation workers or schools. For home users they are hard to prevent as any number of plugins or changes in the browser will just add more noise. Canvas fingerprinting also uses legitimate API in the browser and disabling this will break many pages that use the feature. Acar et al. (2014) suggest that using Tor Browser is the most effective way to prevent canvas fingerprinting as other methods seem to be ineffective. Fingerprinting usually relies heavily on client-side scripts such as JavaScript but other methods such as pure CSS-based fingerprinting have been explored (Böhmer 2018).

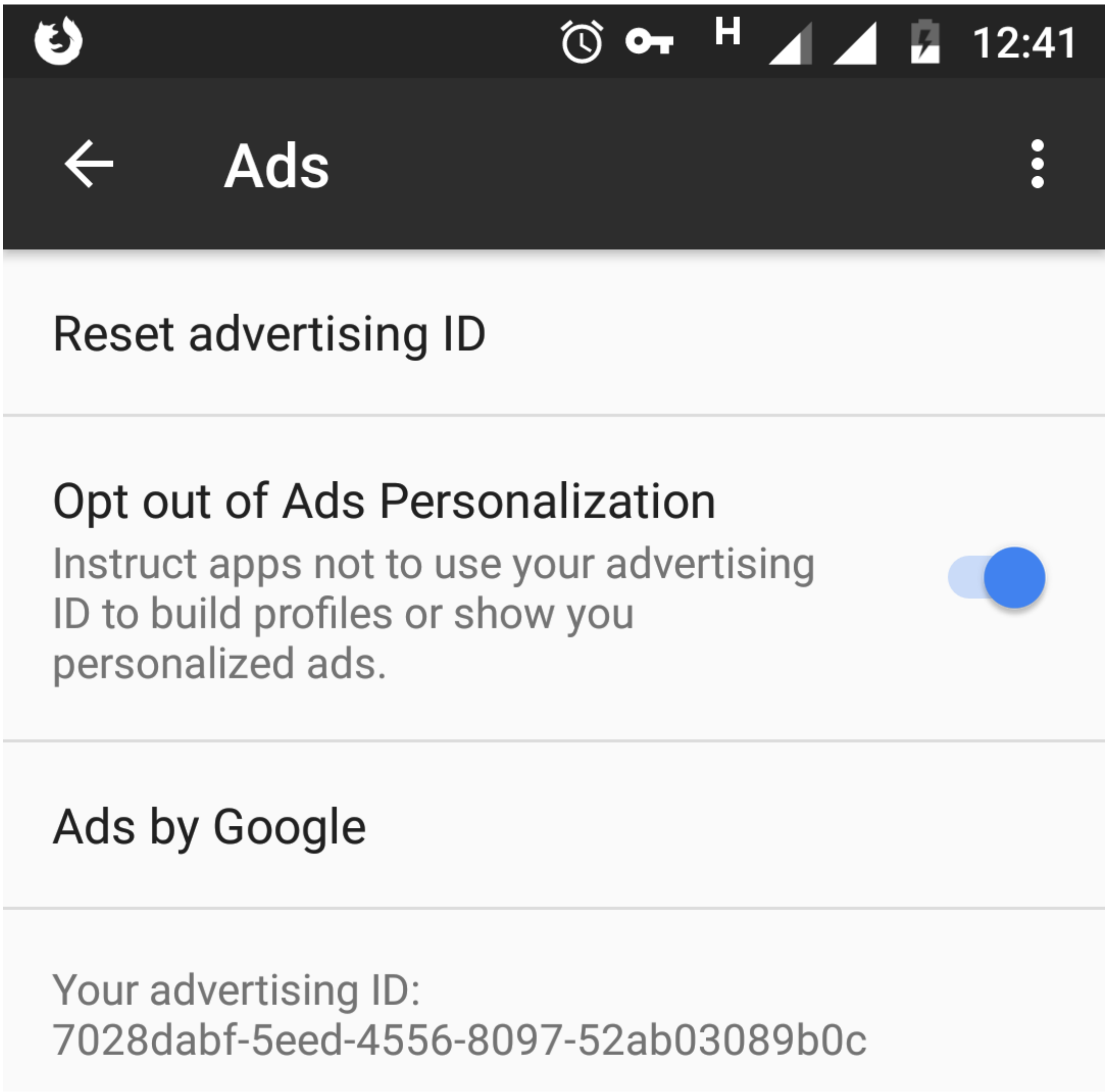
5.8 HSTS Fingerprint

When a client requests a resource using only the domain name, such as www.google.com or even just google.com, the default behavior of browsers is to use HTTP protocol and try http://www.google.com. This means that by default, plaintext HTTP protocol is used instead of secure HTTPS. Traditionally, this has been solved by redirecting the user to HTTPS connection, however, a man-in-the-middle attack made it possible to intercept data and pass it on via HTTP protocol. HTTP Strict Transport Security (HSTS) was created as a way for the server to signal the client that a resource can only be served using HTTPS. This way a man-in-the-middle attack cannot serve the record as plaintext HTTP as the browser would ignore the result. HSTS results are cached permanently in modern browsers unless the user clears the cache manually, which allows using HSTS for tracking. This allows the generation of an HSTS fingerprint. (Stockley 2015)

HSTS Fingerprint is generated by making the client request several benign files, such as pixel images or empty text files. The server side then sets HSTS reply for some of these, which are cached. Next time the user visits the page and loads these files, the ones that were replied with HSTS will be requested with HTTPS and others will be requested with HTTP. If the server handles the requests with HTTPS as binary one’s and HTTP as zeroes, the user can be handed a pseudo-unique identifier with N bits, where N is the number of files originally requested. For example, with just 16 different files 65536 different users can be distinguished. (Stockley 2015)

5.9 Ad identifiers

Google uses a user-specific identifier for offering targeted ads to the users in their Android OS and Play services. This id can then be directly employed in their respective operating systems, removing the need to use other, possibly more invasive tracking methods. The AdID resets every year or the user can reset it by himself in the account settings menu (Figure 13). The AdID is required for all new apps in place of other device identifiers and usage violations will trigger a warning for the developers (Google 2017b). Microsoft and Apple have followed this ideology and are implementing their own identifiers in their systems.



Reset advertising ID

Opt out of Ads Personalization

Instruct apps not to use your advertising ID to build profiles or show you personalized ads.

Ads by Google

Your advertising ID:
7028dabf-5eed-4556-8097-52ab03089b0c

Figure 13. Advertising settings in Android

5.10 Network-level tracking and metadata

The simplest method to track user in network level is to store the users IP address. This is not, however, feasible as due to limitations in IPv4 address space majority of individual users at home have a dynamic IP address that can change anytime. Some users such as organisation workers and users using mobile clients are also behind a firewall and NAT, where the user’s public IPv4 address may be shared between hundreds of other possible endpoints. IPv4 address blocks can, however, be used for coarse geolocational tracking with the use of Geo-IP databases.

IPv6 changes this behavior somewhat as the address space is vastly larger. Operators and ISPs can assign larger address spaces for customers and any endpoint can have a unique IPv6 address. IPv6 does not have or need NAT, all addresses are globally routable. Since Stateless Address Autoconfig (SLAAC) allows endpoints to assign IPv6 addresses from a network automatically using their hardware MAC address, they should also be indistinguishable. This has woken some interest to lobby the use of IPv6 with the agenda of better law enforcement (Jackson 2012) or ad serving (Gauss 2017). There has been concerns on using IPv6 unique addresses for cyberstalking or terrorism too (Groat et al. 2011).

However, the specifications for IPv6 Privacy Extensions for Stateless Address Autoconfig (RFC4941) and additional RFCs such as RFC3972 (RFC3972) and RFC7217 (RFC7217) make it possible to create interface-specific temporary addresses and allow the change of the IPv6 address for use in different transactions. The IPv6 Privacy Extensions are on for all modern operating systems by default, which will effectively prohibit the use of IPv6 addresses for pinpointing certain users with the use of their IPv6 address only. IPSec tunneling may also be used to encrypt and tunnel the payload in the IPv6 network (ibid., 2011).

Another way of tracking the user on network-level is adding metadata to packets when they leave the user’s network and enter the ISPs or Operators network. Jonathan Mayer (2014) explains in his blog how Verizon Wireless inserts a unique X-UIDH identifier to all HTTP headers at the network transit level. This header value is then used at the destination or third party website to direct requests to advertising exchange. If the user has not opted out of Verizon Selects, their targeted marketing, deep packet inspection can also be used to create a behavioral profile. AT&T similarly has been found to sell real-time data through their Mobile Identity API containing Personally Identifiable Information, such as users name and address information (Neustrom 2017).

Network-level data collection is considered more invasive to privacy because it cannot be blocked with cookie deletion or blocking, or any other means performed on the client level. A great deal of the data collected on network level is so-called metadata, data that is not directly relevant to the user’s actions, but linked to it. This kind of data can include e.g. the IP address, port/protocol numbers, user-agent, network type, device identifier, location or even the Personally Identifiable Information of the customer. Metadata is considered to be less sensitive information by the organisations and governments collecting it, making metadata collection acceptable in the context of many legislations. However, as metadata can be collected on network-level, it can be used to create a more detailed image of the user than using traditional data collection from the client-side. Compiling these individual images together gives organisations and governments the ability to create an overall view of total population, possibly even over continents. (Privacy International 2017)

5.11 Operating system telemetry

As the interests for information collection and user tracking have increased, modern operating systems have also evolved to provide data collection natively. This is also called telemetry. Before integrated telemetry, data collection had to be done with the help of spyware, i.e. specific pieces of software applications that were usually bundled together with legitimate applications or distributed via drive-by-downloads where the users were tricked into downloading the software by masquerading it to be an image or document file.

Telemetry works by collecting data on the operating system level using built-in commands and processes that are hard to separate from legitimate services or applications. After the introduction of Windows 10, Microsoft has constantly been accused of spying the users and collecting irrelevant information, with no universally working option to turn the telemetry off. Some of the telemetry functions have been confirmed to have been retrofitted to Windows 7 and 8 also. (Leonhard 2016)

Android makes it trivial for Google to collect data on their user’s activities in multiple ways when they are using their phones. Opt-out mechanisms exist, however, they need to be activated one-by-one. Bundled with aforementioned Google’s Advertisement id, targeted ads are very easy to push to the users.

Telemetry does not happen just on the operating system level. Geforce Experience, the control panel and Nvidia graphics driver companion application have collected crash reports and usage data for a long time, however, it was recently upgraded to include automatic telemetry option which seemingly cannot be turned off. The software also requires a mandatory login via either Google or Facebook account. nVidia Privacy Policy does state that they do not collect Personally Identifiable Information, however, there is also no method of opting out of the collection other than not using it. (Burke 2016)

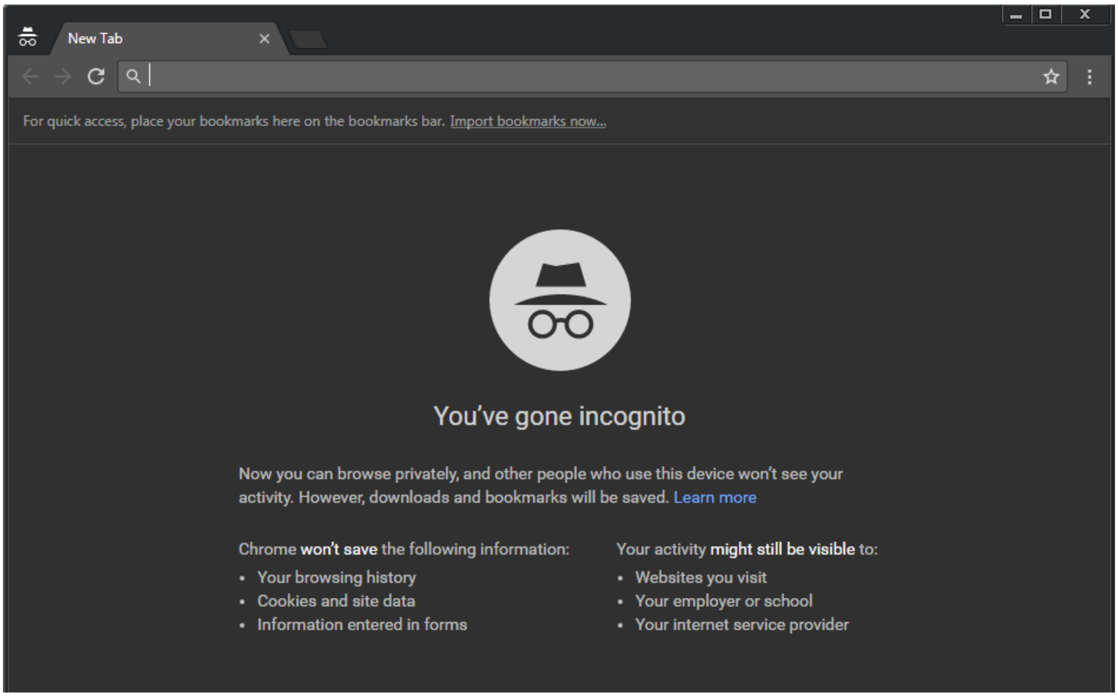
A totally new privacy issue is the spread of so-called smart devices, household appliances that are internet-enabled and run a specific operating system. These devices can contain unpatchable security vulnerabilities, which in turn can be used to monitor the user. They also include native ways to collect telemetry data from the user. One example is the Android application for Bose Bluetooth noise-cancelling headphones, Bose Connect, which was found to transmit music and audio data to third parties for data mining (Kyle vs Bose 2017)

6 Evasion tools

6.1 Browser integrated options

Most modern browsers include some options to enhance privacy and disable some tracking components. All major browsers currently allow the user to block third-party cookies and to delete cookies every time the browser is closed, even if their expiration is set to far in the future. These are, however, not default and deleting cookies every time the browser is closed will weaken user experience on some pages that rely heavily on cookies, making the users choose convenience over privacy. Third-party cookie blocking also does nothing for first-party cookies, which can be set and read by client-side or server-side script included in the first-party site.

Browsers also send Do Not Track requests either by default or for private browsing sessions. Private browsing sessions open up a new window that should lose all stored data when the private session is closed (Figure 14). The names vary by software (Private Browse, Incognito, InPrivate mode).



For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

You've gone incognito

Now you can browse privately, and other people who use this device won't see your activity. However, downloads and bookmarks will be saved. Learn more

Chrome **won't save** the following information:

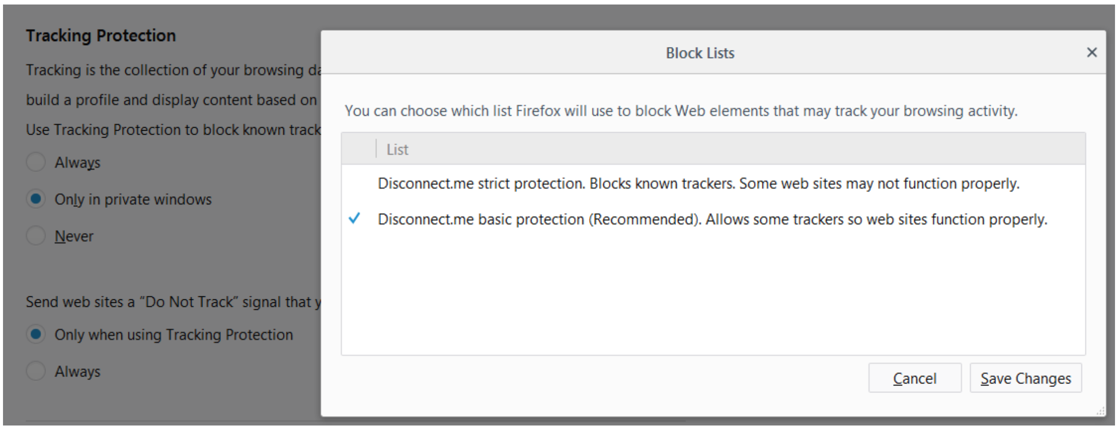
- Your browsing history
- Cookies and site data
- Information entered in forms

Your activity **might still be visible** to:

- Websites you visit
- Your employer or school
- Your internet service provider

Figure 14. Incognito mode in Google Chrome

Firefox has a Tracking Protection feature (Figure 15) since version 57, however, it is enabled by default on Private Browsing windows only. It uses ad and tracker-blocking services of Disconnect.me –service, which also has its own plugin for other software (Disconnect.me 2018). Private browsing also ignores the previously stored HSTS information so HSTS fingerprinting does not work.



Tracking Protection

Tracking is the collection of your browsing d

build a profile and display content based on

Use Tracking Protection to block known track

○ Always

● Only in private windows

○ Never

Send web sites a "Do Not Track" signal that y

● Only when using Tracking Protection

○ Always

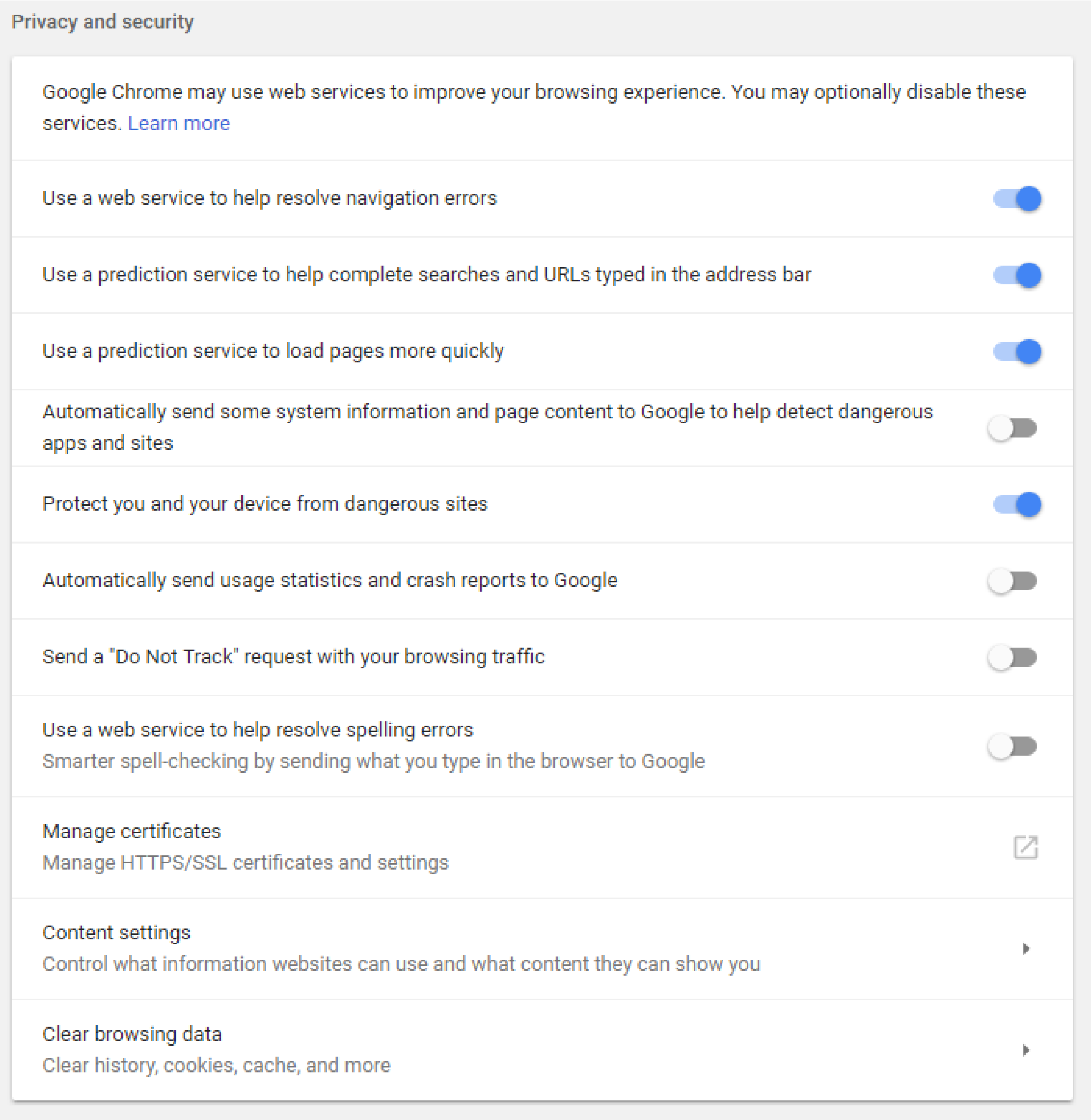
Block Lists

You can choose which list Firefox will use to block Web elements that may track your browsing activity.

	List
	Disconnect.me strict protection. Blocks known trackers. Some web sites may not function properly.
✓	Disconnect.me basic protection (Recommended). Allows some trackers so web sites function properly.

Figure 15. Tracking Protection settings

Some features in the browsers do not enhance but weaken the privacy by default. A good example is page prediction included in Google Chrome, which preloads and pre-renders some web content based on browsing history. This might load pages and tracking elements unbeknownst to the user in the background. Similar effect is on search prediction, which sends data to Google and suggests search patterns based on history. These features are on by default (Figure 16)



Privacy and security

Google Chrome may use web services to improve your browsing experience. You may optionally disable these services. Learn more

Use a web service to help resolve navigation errors ●

Use a prediction service to help complete searches and URLs typed in the address bar ●

Use a prediction service to load pages more quickly ●

Automatically send some system information and page content to Google to help detect dangerous apps and sites ○

Protect you and your device from dangerous sites ●

Automatically send usage statistics and crash reports to Google ○

Send a "Do Not Track" request with your browsing traffic ○

Use a web service to help resolve spelling errors
Smarter spell-checking by sending what you type in the browser to Google ○

Manage certificates
Manage HTTPS/SSL certificates and settings

Content settings
Control what information websites can use and what content they can show you

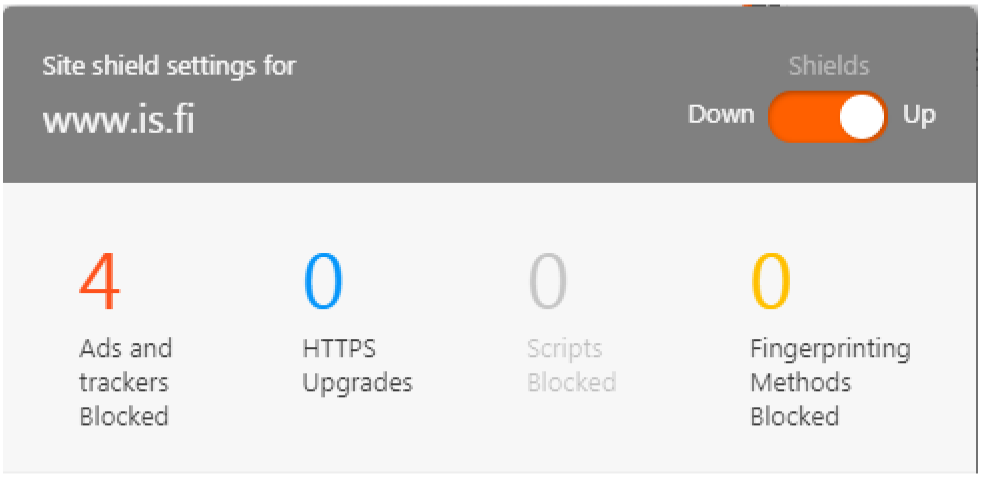
Clear browsing data
Clear history, cookies, cache, and more

Figure 16. Chrome Privacy settings

Microsoft Edge sends a great deal of Telemetry data by default using the Windows 10 native Telemetry collection. It also seems that it stores plenty of data locally as Artefacts, even if the user uses InPrivate mode (Muir 2015)

Apple Safari includes several options to enhance browsing privacy, for example third-party cookies are blocked by default. The newest version uses the Intelligent Tracking Prevention provided by the Webkit engine, which uses machine learning to remove cross-site tracking (Webkit.org 2018). This setting is on and has raised some controversy as advertisers feel this is considered “sabotaging the economic model for the Internet” (Macrumors 2018).

More privacy-oriented browsers such as the ones selected for the implementation tests include more built-in methods, and they are also enabled by default. For example, Brave browser includes a so-called 'Shields' –function, which provides the user a quick glance of what components can be and are currently blocked (Figure 17). These functions work mainly via the use of blocklists from AdBlock Easylist and Disconnect.me combined with hard-coded siteHacks (Hirahara 2017).



Site shield settings for

www.is.fi

Shields

Down Up

4 Ads and trackers Blocked

0 HTTPS Upgrades

0 Scripts Blocked

0 Fingerprinting Methods Blocked

Figure 17. Brave Shields

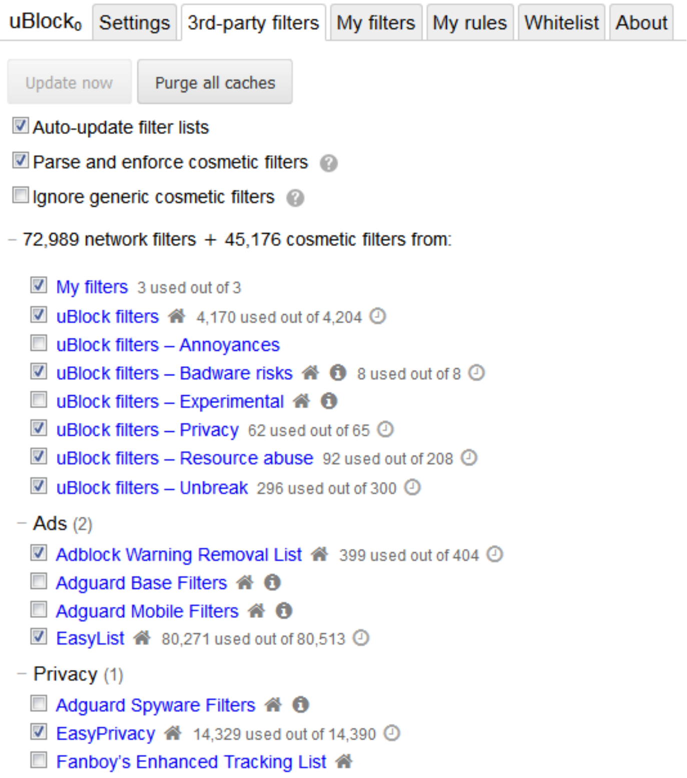
Brave has also the possibility to opt-in to ads that are considered harmless or non-intrusive. Brave Team's next step in the line is to replace blocked ads with selected Brave Ads, utilizing an anonymity protocol. The ultimate goal for the Team is to replace the current ad model with a blockchain-based token called Basic Attention Token (BAT). BAT provides micropayments to web browsing, recording the user attention on advertisements, which in turn earns them tokens. The tokens can then be used to acquire better content, leave or up-/downvote comments, or even convert the tokens to real money. (Brave 2018)

TorBrowser is the Tor Project's take on creating a privacy-enhanced browser. It is based on Firefox 52 Extended Support Release (ESR), uses the distributed TOR network by default and also employs various techniques to block tracking at the browser level. One key philosophy in Tor Browser is that it does not use filter- or blocklist-based addons, as "…these addons do not add any real privacy to a proper implementation of the above privacy requirements, and that development efforts should be focused on general solutions that prevent tracking by all third parties, rather than a list of specific URLs or hosts." (Perry et al. 2018)

Torbrowser has a specifically crafted Cross-Origin Indentifier Unlinkability feature (also called First Party Isolation by the Firefox team), which ties the identifier cookies and other browser state information to the domain only in the URL bar. This is combined with so-called “double-keying”, where every first party scope receives its own third party scope. This effectively renders third party cookie tracking across multiple sites useless; yet, still retains the functionality that requires third-party cookies. A similar feature, Cross-Origin Fingerprinting Unlinkability is included to prevent fingerprinting across domains. (Perry et al. 2018)

6.2 Ad blockers

Ad blockers were originally created as tools for blocking intrusive ads on pages. Banners, popups, or flash content can be blocked using simple plugins such as AdBlock, AdBlock Plus, uBlock Origin or Disconnect.me that match HTML elements or images based on class names or domains. Better ad blockers can block Google AdWord search results and YouTube ads also. Usually plugins like these are based on blocklists that are maintained and updated regularly. Combining several blocklists usually gives the best result (Figure 18).



☑ Auto-update filter lists
☑ Parse and enforce cosmetic filters
☐ Ignore generic cosmetic filters

– 72,989 network filters + 45,176 cosmetic filters from:

☑ My filters 3 used out of 3
☑ uBlock filters 4,170 used out of 4,204
☐ uBlock filters – Annoyances
☑ uBlock filters – Badware risks 8 used out of 8
☐ uBlock filters – Experimental
☑ uBlock filters – Privacy 62 used out of 65
☑ uBlock filters – Resource abuse 92 used out of 208
☑ uBlock filters – Unbreak 296 used out of 300

– Ads (2)

☑ Adblock Warning Removal List 399 used out of 404
☐ Adguard Base Filters
☐ Adguard Mobile Filters
☑ EasyList 80,271 used out of 80,513

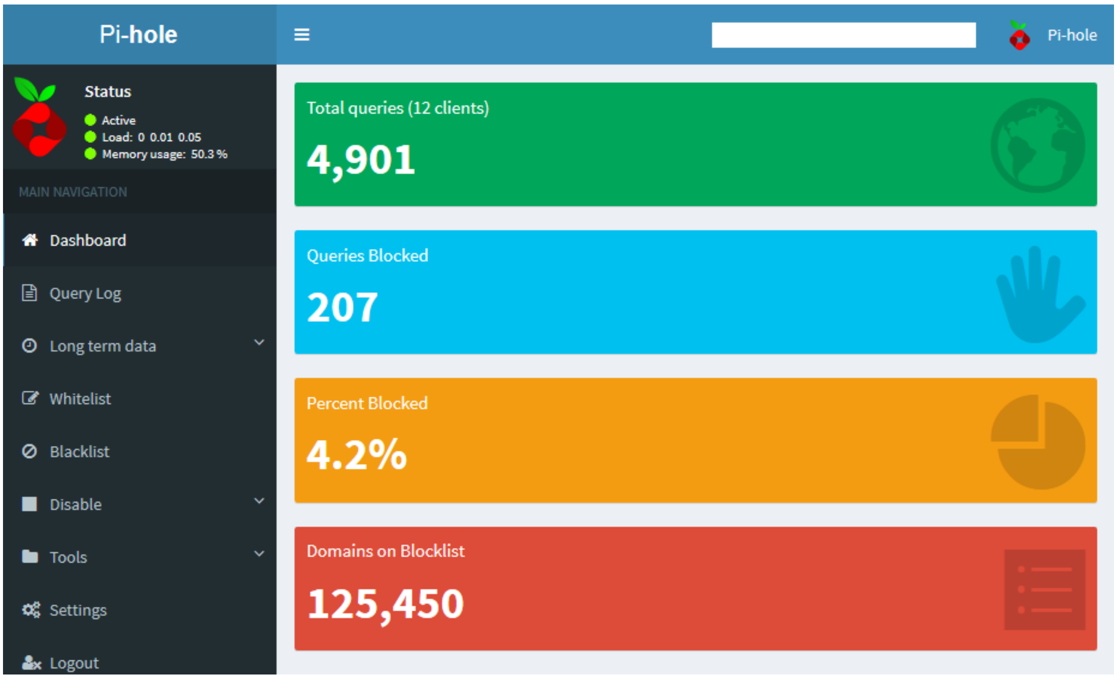
– Privacy (1)

☐ Adguard Spyware Filters
☑ EasyPrivacy 14,329 used out of 14,390
☐ Fanboy’s Enhanced Tracking List

Figure 18. Few of the blocklists used in uBlock Origin

These blocklists can be used to block tracking elements and domains, which are usually combined into ads anyway. However, as the blocklists usually contain high-level domains, big companies such as Google, Facebook or Amazon cannot be blocked as that would effectively deny all usage of their services.

Other way to block ads, trackers and spyware is to use network-level tools such as Pi-hole (Figure 19) that block the requests at Domain Name System (DNS) level. Every webpage visit or other request usually generates a DNS request as the name is *resolved* to an IP address. By circumventing DNS requests pointed to ad networks or tracking elements, these requests can be denied without them ever leaving the network. This does not, however, work on tracking elements that use hard-coded IP addresses such as Windows 10 Telemetry (Petri 2016).



Status
● Active
● Load: 0 0.01 0.05
● Memory usage: 50.3 %

Total queries (12 clients)
4,901

Queries Blocked
207

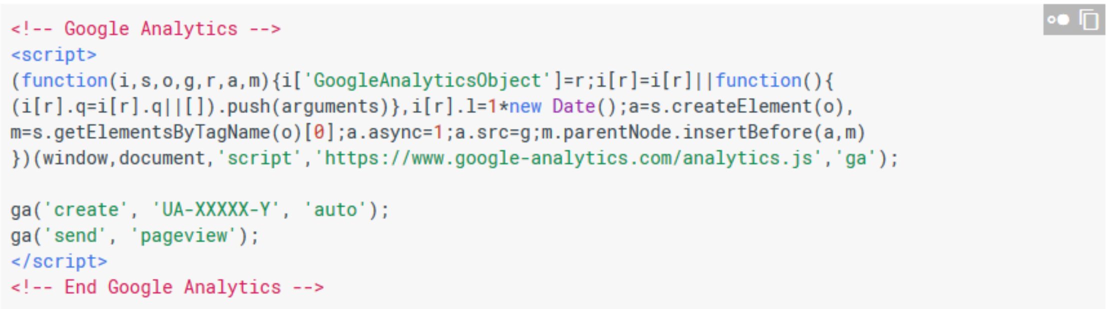
Percent Blocked
4.2%

Domains on Blocklist
125,450

Figure 19. Pi-hole dashboard

6.3 Script blockers

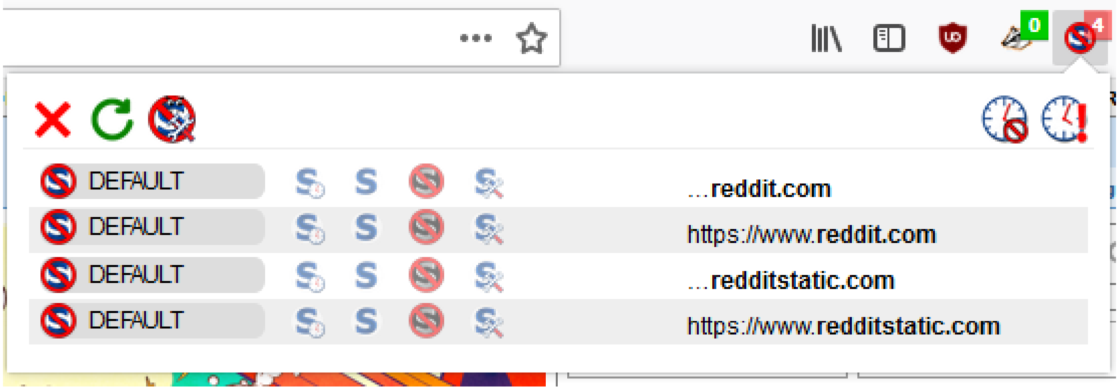
A great number of tracking elements are based on scripts embedded on the webpages. They get loaded with the page and are run on the client's browser, gathering information and sending it back to the tracking server. Figure 20 shows how Google Analytics is inserted on the page and how it calls a JavaScript file *analytics.js* from the Google servers. The script creates a tracking object with the unique identified *UA-XXXX-Y* and sends the command *pageview* to the analytics engine, ranking the currently loaded page up. (Google 2018)



```
<!-- Google Analytics -->
<script>
(function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
(i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
})(window,document,'script','https://www.google-analytics.com/analytics.js','ga');

ga('create', 'UA-XXXXX-Y', 'auto');
ga('send', 'pageview');
</script>
<!-- End Google Analytics -->
```

Figure 20. Google Analytics (Google 2018)

Fingerprinting (see section 5.7) is one tracking element that fully uses client-side scripts to collect as much information from the browser as it can. These kind of scripts can be blocked with the use of plugins such as NoScript or uMatrix, which use blocklists and user-generated rules to either allow or deny scripts on the webpage (Figure 21).



DEFAULT ...reddit.com
DEFAULT https://www.reddit.com
DEFAULT ...redditstatic.com
DEFAULT https://www.redditstatic.com

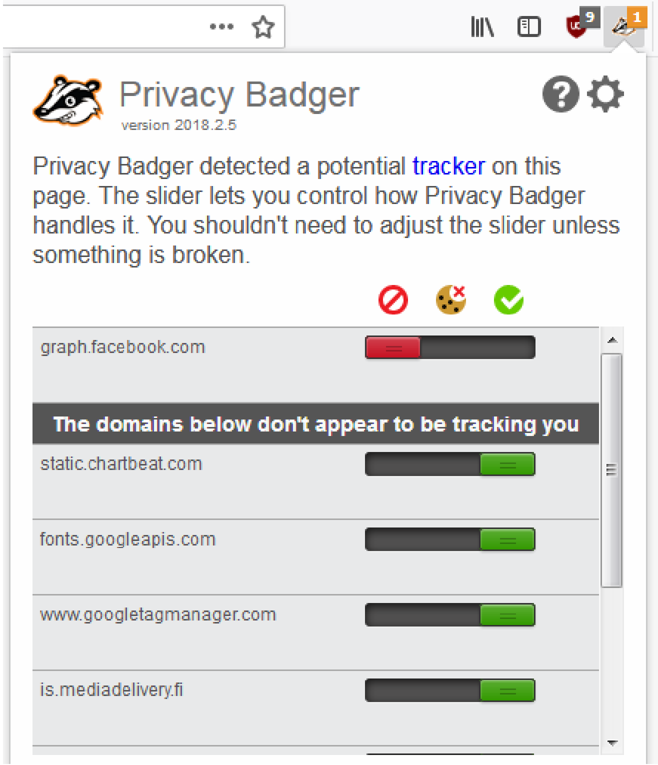
Figure 21. Scripts blocked for www.reddit.com by NoScript extension

The additional benefit is that many popups and minor annoyances on pages use similar scripts, however, the downside is that many dynamic web pages use scripts legitimately to render the page more efficiently. Hence, a great number of pages will either break or the user must create whitelist rules to allow those pages, which can be time-consuming.

6.4 Privacy Badger

As Do Not Track has not been widely adopted, Electronic Frontier Foundation (EFF) has created a seemingly on-for-all solution for denying tracking elements in the form of Privacy Badger. Privacy Badger is a browser plugin that analyzes the use of third-party content on websites and tries to block offending elements deemed to be tracking the user. The system does not use blocklists but instead cross-references cookie and script use on different websites with heuristics to find commons and checks them using algorithms and policies. The newer versions of Privacy Badger also feature blocking of supercookies that use local storage and canvas fingerprinting. (EFF 2015)

All third party elements are deemed “green” first to minimize false positives. When a third party element is found on multiple sites and is possibly trying to track the user, Privacy Badger moves it to “yellowlist”, where further analysis is necessary. If the element is considered to be tracking the user after a period of time, it is set as “red” and subsequently blocked (Figure 22). (EFF 2015)



Privacy Badger
version 2018.2.5

Privacy Badger detected a potential tracker on this page. The slider lets you control how Privacy Badger handles it. You shouldn't need to adjust the slider unless something is broken.

graph.facebook.com

The domains below don't appear to be tracking you

static.chartbeat.com

fonts.googleapis.com

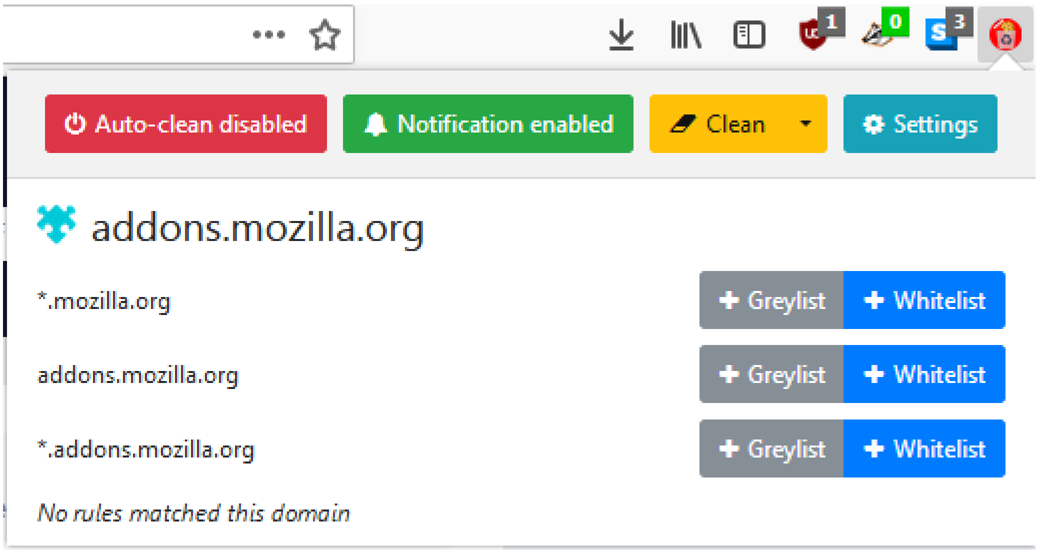
www.googletagmanager.com

is.mediadelivery.fi

Figure 22. Privacy Badger for a generic news site

6.5 Data removal

The previously explained plugins try to block all possible tracking. Another solution is to simply remove unique identifiers that have been stored in the browser cache or local storage, such as cookies or local storage objects. Plugins such as Cookie Autodelete, Self-destructing-cookies and Forget Me Not are aimed to remove local data selected by personal rules. By choosing to delete all cookies/local data by default and only store what is relevant, the user can essentially bypass many unique identifiers stored locally and still stay logged in to their most used services (Figure 23).


Auto-clean disabled
Notification enabled

addons.mozilla.org

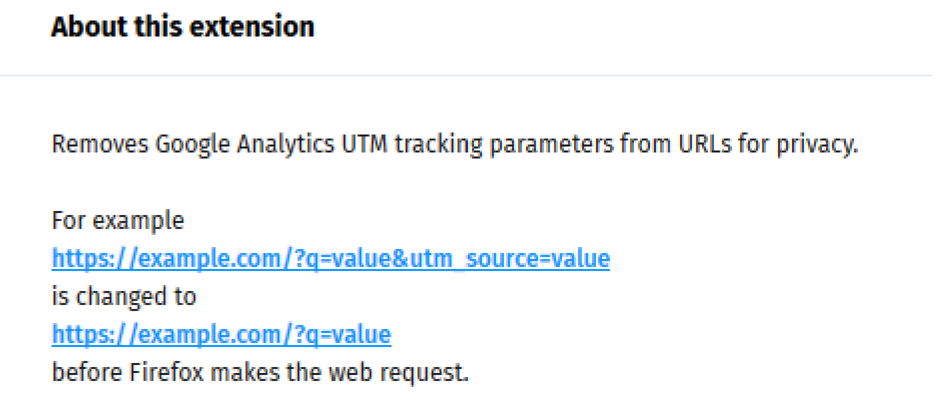
*.mozilla.org
addons.mozilla.org
*.addons.mozilla.org

No rules matched this domain

Figure 23. Settings for Cookie AutoDelete plugin

6.6 Link sanitizers

As it is possible to include tracking elements in the URL of a webpage using HTTP GET parameters, some plugins like UTM Remover and Neat URL have been created to sanitize links in URLs (Figure 24). Other plugins remove the tracking elements from Google or other search results as they usually contain referral information or HTTP redirects.



About this extension

Removes Google Analytics UTM tracking parameters from URLs for privacy.

For example
https://example.com/?q=value&utm_source=value
is changed to
https://example.com/?q=value
before Firefox makes the web request.

Figure 24. UTM Remover URL example

6.7 Noise generators

Noise generators such as AdNauseam, TrackMeNot and Canvas Defender have a completely different approach to disabling tracking on sites. They are based on the premise of obfuscation and making random noise, which will hide the user's relevant interests in the middle of huge data stream. The plugins work by generating false clicks on ads (AdNauseam) or by submitting random searches on Google at time intervals (TrackMeNot). Canvas Defender works by injecting random noise to the canvas fingerprint image, thus making the fingerprint different from the usual one. This way the signal-to-noise ratio of the tracking decreases and the tracked data becomes unusable. (Howe & Nissenbaum 2017)

The plugins have raised some controversy as AdNauseam was removed from the Google Chrome add-ons (AdNauseam 2017) and TrackMeNot was deemed as not worth the time by Bruce Schneier (2006d) as it only adds noise and does not anonymize the relevant searches in any way. This way the user is still tracked even though random queries are sent at the same time. Another downside is that the plugins generate extra bandwidth by constantly sending in queries, which on a metered connection will quickly add up. In addition, TrackMeNot enables the users to knowingly send in tracked keywords such as "child porn" or "bomb recipes", which can eventually lead to getting flagged on black lists by operators in some countries (Schneier 2006d).

7 Implementation and testing

The different tracking methods were implemented and tested by finding a readily available application or code that could be dropped in to the test environment. JYVSECTEC's Realistic Global Cyber Environment (RGCE) was used as the testing platfrom. RGCE is a closed environment or a cyber range that provides an isolated sandbox for cyber security activities, such as cyber exercises, training and research and development (JYVSECTEC 2017). RGCE includes a global network similar to the Internet, which enables realistic modeling of Internet-level networks, services and also cyber threats and campaigns. The environment includes a collection of news pages and an advertisement system, so these pages could be used for injecting the tracking code. A variety of pages was also created for the use of training sessions or demonstrations.

7.1 Test methodology

Windows 7 was selected as the base system of the tests mainly because RGCE uses Windows 7 virtual machines as base images. In addition, as all of the tested tracking methods were web-based, the operating system should not matter in the results. The base image was a clean install with Adobe Flash version 28.0.0.161 (NPAPI) and Java 8u161.

First, the idea was to test tracking with multiple browsers, however, as Microsoft Edge is not available for Windows 7 and Apple has stopped developing Windows version of Safari, these two became irrelevant. As EFF and multiple other sources seem to favor Firefox as a more privacy-oriented browser, it was selected as the main browser for testing. In addition, future demonstration and training cases will be heavily Firefox-based. Chrome and IE were used for comparison to see if there were major differences in browser behavior. Some new browsers such as Epic, Brave and Torbrowser have been created for maximum privacy, so these were also tested for reference purposes.

Torbrowser by default connects to every website via TOR, which caused minor issues. RGCE has a TOR implementation but at the moment of writing it was unavailable for testing and as the interest was mainly on browser-level methods it was deemed irrelevant. Turning TOR usage off required some manual intervention:

- Proxy via TOR was turned off (Options – Advanced – Network – Setttings – No Proxy)
- Local DNS was set to be used instead of proxied (about:config – network.proxy.socks_remote_dns = false)
- TOR services were turned off (Extensions: disable Torbutton and TorLauncher)

The full list of browsers used for testing as follows:

- Firefox 58.0.2
- Chrome 64.0.3282.186
- Internet Explorer 11.0.9600.17843
- Epic 62.0.3202.94
- Brave 0.21.18
- Torbrowser 52.6.0.6607

A selection of plugins was used for testing evasion techniques. These plugins were selected mainly by finding out what other people use by searching the privacy-oriented subreddit (r/privacy), guides for enhancing privacy and removing tracking components and plugin rankings on Firefox Add-ons page. This should represent a common view of what a privacy-oriented person would use for daily browsing. In addition, many of the alternative plugins use same or very similar implementation, for example, all the ad blockers rely on the same blocklists.

The selected plugins were:

- uBlock Origin 1.15.10
- Ghostery 8.0.9.7
- Disconnect 5.18.21
- NoScript 10.1.6.5
- Forget Me Not 0.8.8
- Privacy Badger 2018.2.5
- HTTPS Everywhere 2018.4.3 (For the HSTS case only)

Plugins were active one at a time and if needed, browser cache was flushed or the user profile deleted and re-created. All separate technologies were tested with clean copies of the VM so no residual data was present from previous methods. Private browsing mode and Firefox tracking protection was tested also on cases where it seemed to might have a difference.

7.2 Cookies

Cookies and beacons could be easily dropped to any existing web page in RGCE, however, a working background system for the tracking side was necessary. Mautic marketing automation tool was initially selected for this task as it had an open source community version and provided the support for both cookies and beacon images. The documentation was also top-notch so the software could easily be used for future demonstrations and training exercises.

Mautic uses marketing campaigns and they can be created in a flowchart-kind of tool where you can start tracking a user and then make decisions and actions based on the user input such as page visits and forms. A demo campaign for a fictional web shop Yalando.com was created (Figure 25) that included a raffle where the user can insert his e-mail address for further user identification.

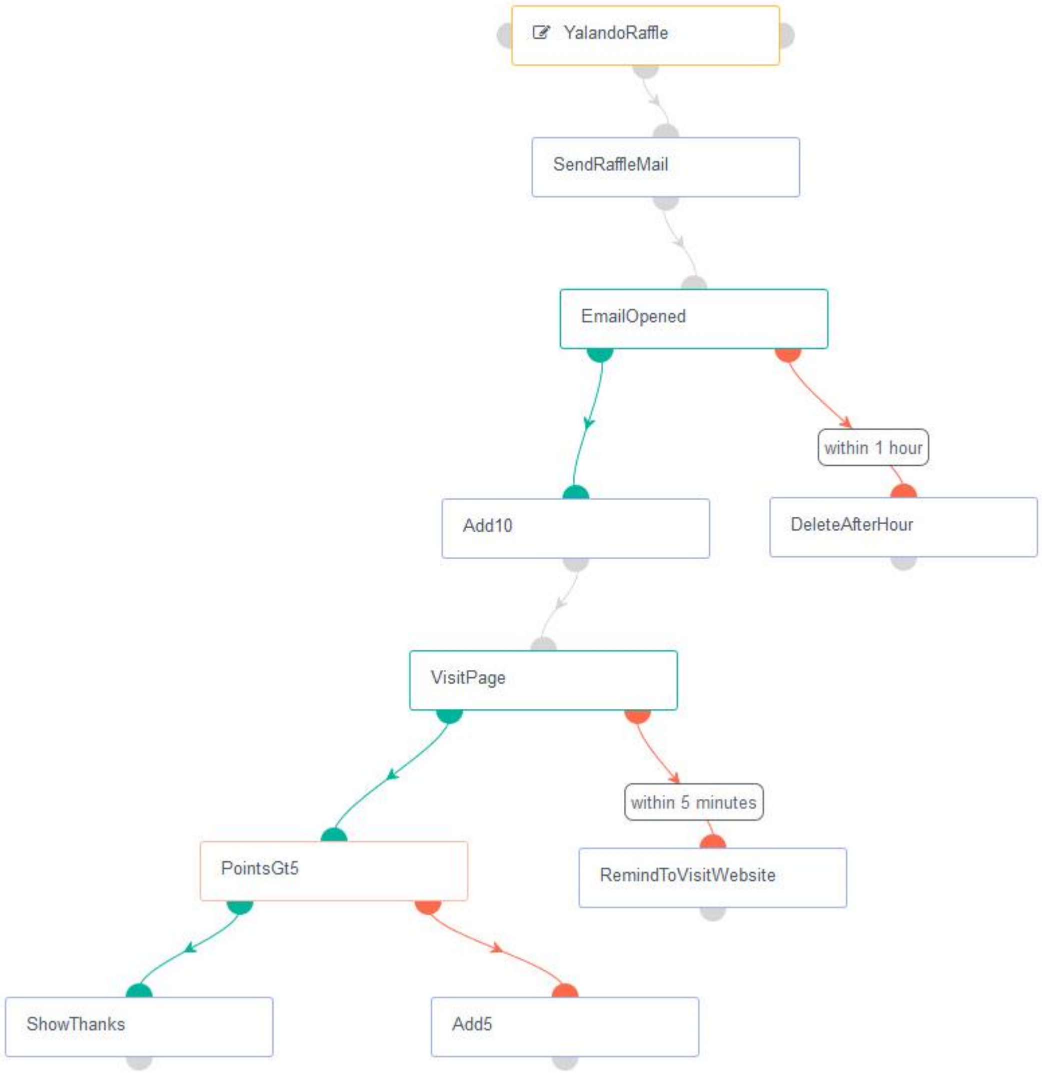
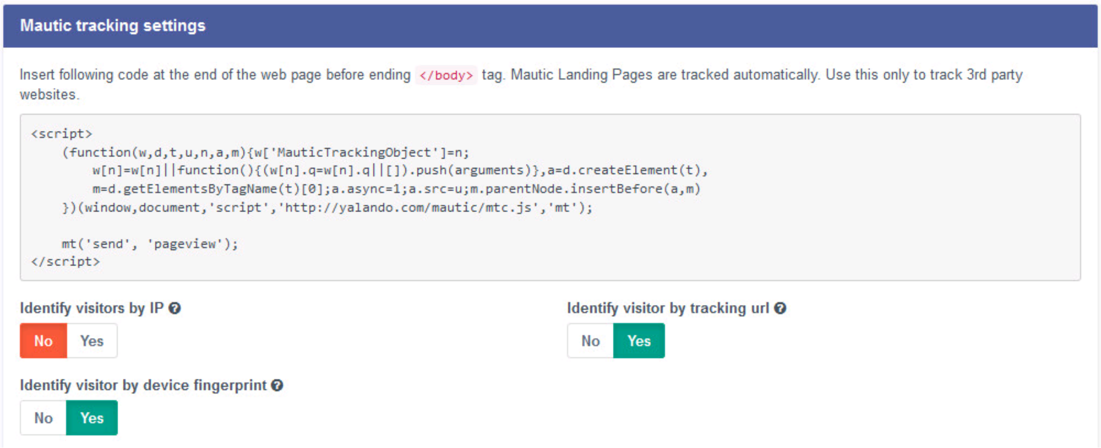


Figure 25. Campaign for testing with Yalando.com

The user can then be tracked on various web pages by including the Mautic tracking script which stores a cookie and loads a tracking beacon from the server. The user can be tracked with this script even if he/she skips the e-mail form. The tracking script was included in RGCE mockups of Iltalehti.fi and Ampparit.com. As the tests were conducted on the same workstation, Mautic recognized the client via IP address as the same user. IP tracking was turned off from Mautic settings, leaving only tracking via cookies and fingerprint active (Figure 26).



Mautic tracking settings

Insert following code at the end of the web page before ending `</body>` tag. Mautic Landing Pages are tracked automatically. Use this only to track 3rd party websites.

```
<script>
    (function(w,d,t,u,n,a,m){w['MauticTrackingObject']=n;
        w[n]=w[n]||function(){(w[n].q=w[n].q||[]).push(arguments)},a=d.createElement(t),
        m=d.getElementsByTagName(t)[0];a.async=1;a.src=u;m.parentNode.insertBefore(a,m)
    })(window,document,'script','http://yalando.com/mautic/mtc.js','mt');

    mt('send', 'pageview');
</script>
```

Identify visitors by IP: [✓ No] Yes

Identify visitor by tracking url: No [✓ Yes]

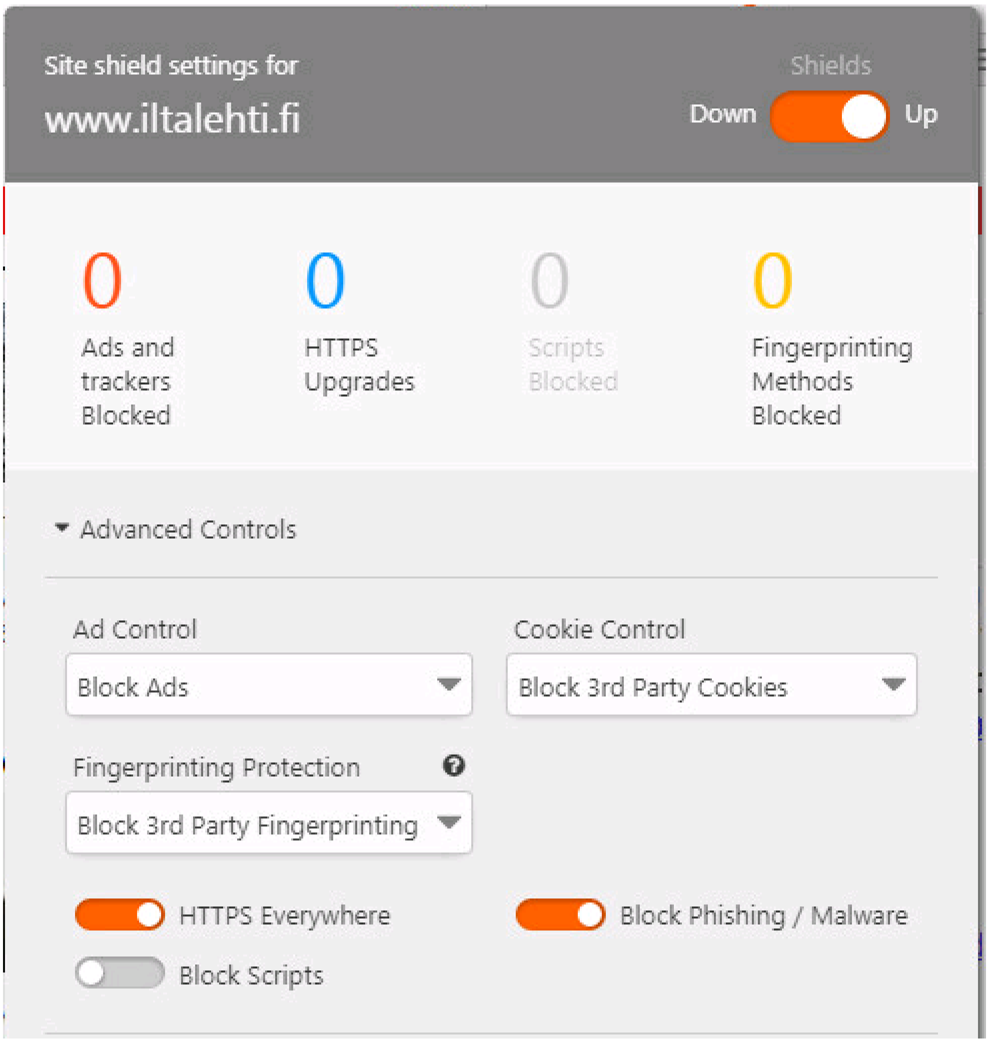
Identify visitor by device fingerprint: No [✓ Yes]

Figure 26. Mautic tracking script and methods

Next, cookies were tested by visiting Yalando.com and Iltalehti.fi/Ampparit.com to see that cookies are created correctly. Mautic Javascript tracking script (mtc.js) was loaded from the server correctly and the script created a cookie for the first party domain where the element was referred to. The cookies included a lead or user id, which can be used to distinguish the user directly, and a session id for the current session, which resets every 30 minutes according to documentation (Mautic 2018). If third-party cookies are blocked from the browser settings, no cookie gets sent with the request to yalando.com/mtc.js when browsing Iltalehti.fi. If the mtc.js is served from the site itself (iltalehti.fi/mtc.js), tracking works as intended. It seemed obvious that Mautic wants the tracking party to include the mtc.js from the first party domain in order to bypass third-party cookie blocking.

All general browsers (Firefox, Chrome and Internet Explorer) worked exactly alike. Private browse modes such as Incognito and InPrivate rejected the old cookies and new tracking identifiers were created for each private browse session. Cookies were forgotten automatically when the private browse session ended. Firefox Tracking Protection did nothing to evade cookies, as it relies on Do Not Track functionality and domain blocklists by Disconnect.me (Disconnect.me 2018).

The privacy-oriented browsers performed somewhat differently. Epic browser and Brave both blocked the tracking by default as no new page visits were registered in Mautic. Brave, however, did not show anything in its ‘Shields’ menu as blocked (Figure 27). As Brave also does not have a developer console like the other browsers, it was very hard to verify if the cookie was even set. However, allowing the use of third-party cookies (by default they are disabled) caused the page visits to register correctly in Mautic again.



Site shield settings for

www.iltalehti.fi

Shields

Down Up

0 Ads and trackers Blocked

0 HTTPS Upgrades

0 Scripts Blocked

0 Fingerprinting Methods Blocked

Advanced Controls

Ad Control: Block Ads

Cookie Control: Block 3rd Party Cookies

Fingerprinting Protection: Block 3rd Party Fingerprinting

HTTPS Everywhere

Block Phishing / Malware

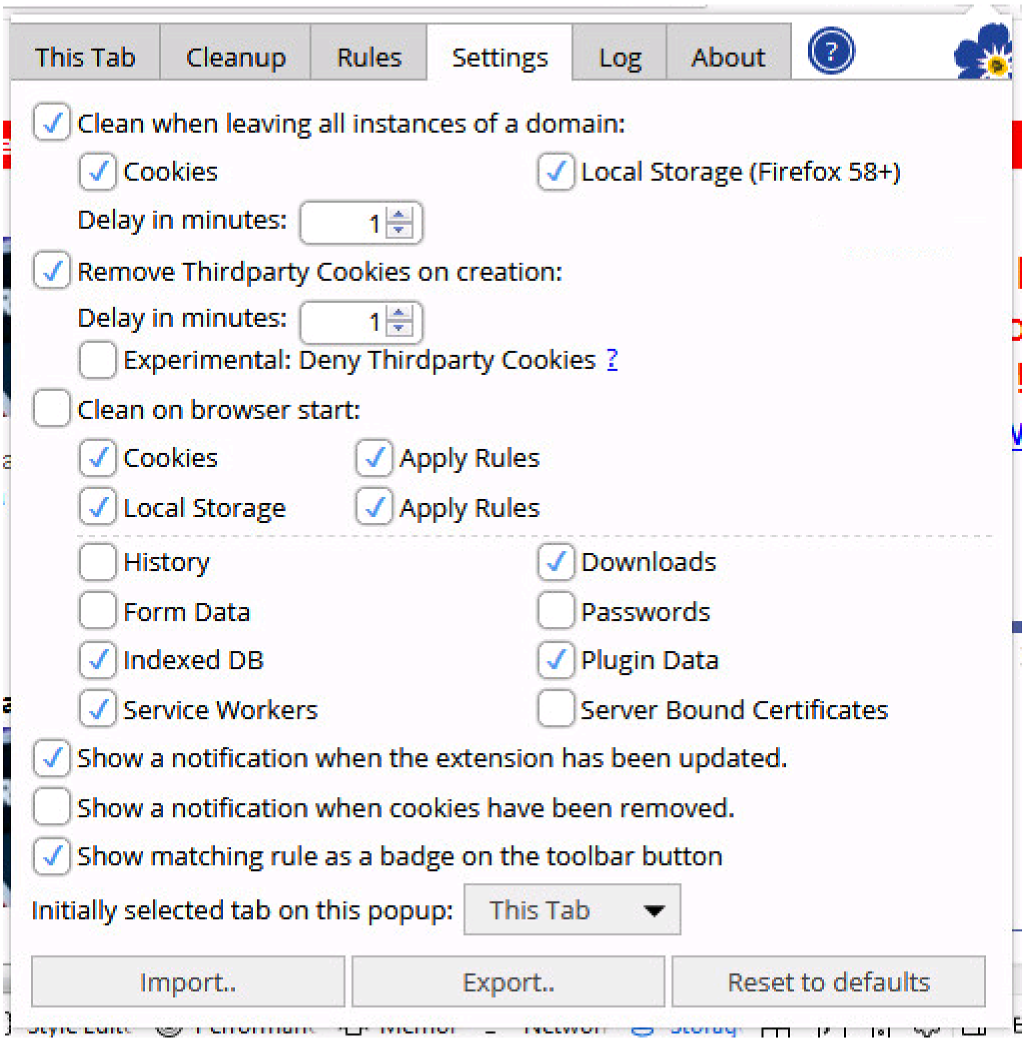
Block Scripts

Figure 27. Brave Shields –view did not register the tracking element

Torbrowser worked slightly differently as the mtc.js script did get run and the cookie did get set even though scripts were blocked from the settings. Subsequent visits to Iltalehti.fi generated new tracking ids to each and every component requested, which seemed to be the result of the Cross-Origin Identifier Unlinkability feature explained in chapter 6.1.

As the testing moved to plugins, Ghostery, uBlock Origin and Disconnect.me were found to rely only on blocklists, which by default does nothing to block either the mtc.js or the cookies set by it. Blacklisting yalando.com did of course deny the page to load the mtc.js script. However, first party server side cookies will not get blocked this way. Noscript otherwise did block the whole script altogether, which results in the cookie not been set ever. To further test this, a simple cookie example page was created with PHP and it was confirmed that with server-side first party scripts all of these extensions did nothing.

Forget Me Not also by default does not block the cookies, but instead deletes them after a set amount of time when the user leaves the sites (Figure 28). Third-party cookies can also be deleted after set minutes of creation. Both settings were confirmed to work as intended, however, the user must manually whitelist the pages that should retain their cookies such as logged in pages etc.



☑ Clean when leaving all instances of a domain:

☑ Cookies ☑ Local Storage (Firefox 58+)

Delay in minutes: 1

☑ Remove Thirdparty Cookies on creation:

Delay in minutes: 1

☐ Experimental: Deny Thirdparty Cookies ?

☐ Clean on browser start:

☑ Cookies ☑ Apply Rules

☑ Local Storage ☑ Apply Rules

☐ History ☑ Downloads

☐ Form Data ☐ Passwords

☑ Indexed DB ☑ Plugin Data

☑ Service Workers ☐ Server Bound Certificates

☑ Show a notification when the extension has been updated.

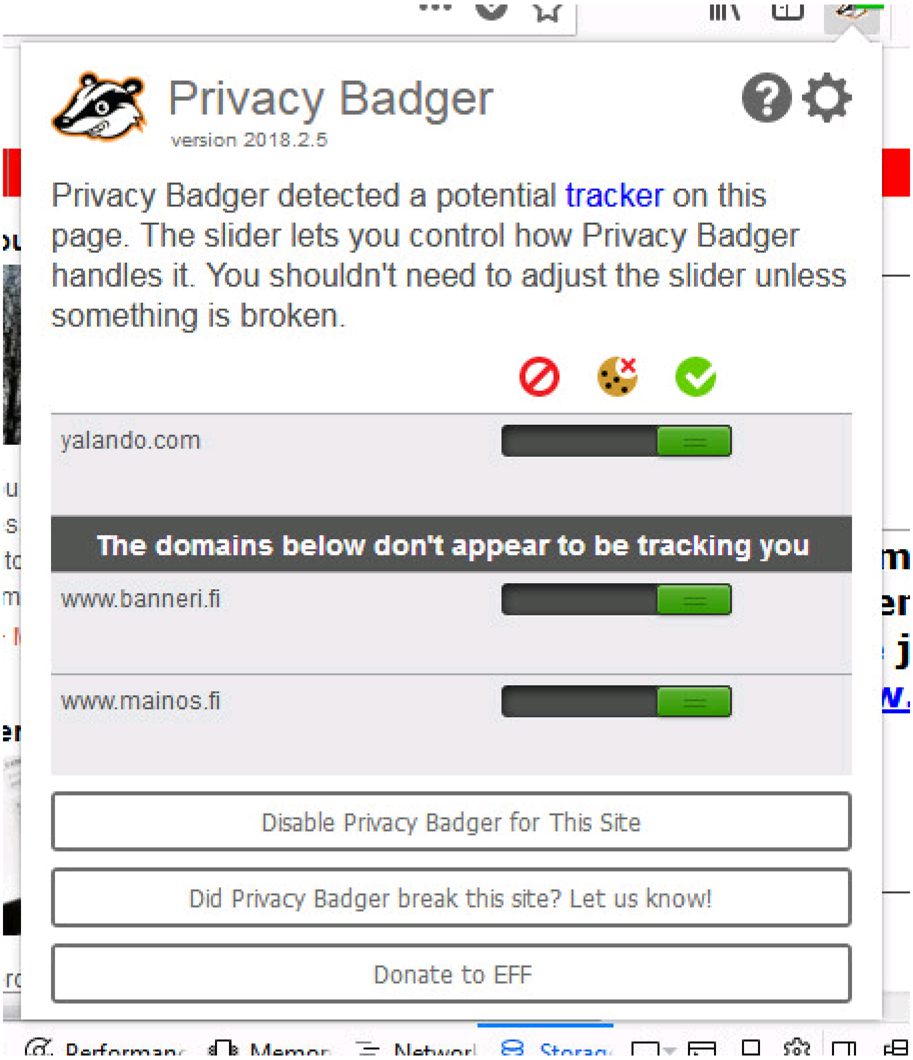
☐ Show a notification when cookies have been removed.

☑ Show matching rule as a badge on the toolbar button

Initially selected tab on this popup: This Tab

Figure 28. Forget Me Not automatic deletion of cookies

The last test was to use Privacy Badger, which should use heuristics to determine possible trackers on the web page. With Mautic tracking (Yalando, Iltalehti.fi and Ampparit.com), Privacy Badger categorized the third-party access as 'greenlisted' even with subsequent page visits (Figure 29). This is due to the fact that the cookie was set as first-party and Privacy Badger only considers third-party elements as tracking elements (Privacy Badger Source Code 2018).



Privacy Badger
version 2018.2.5

Privacy Badger detected a potential tracker on this page. The slider lets you control how Privacy Badger handles it. You shouldn't need to adjust the slider unless something is broken.

yalando.com

The domains below don't appear to be tracking you

www.banneri.fi

www.mainos.fi

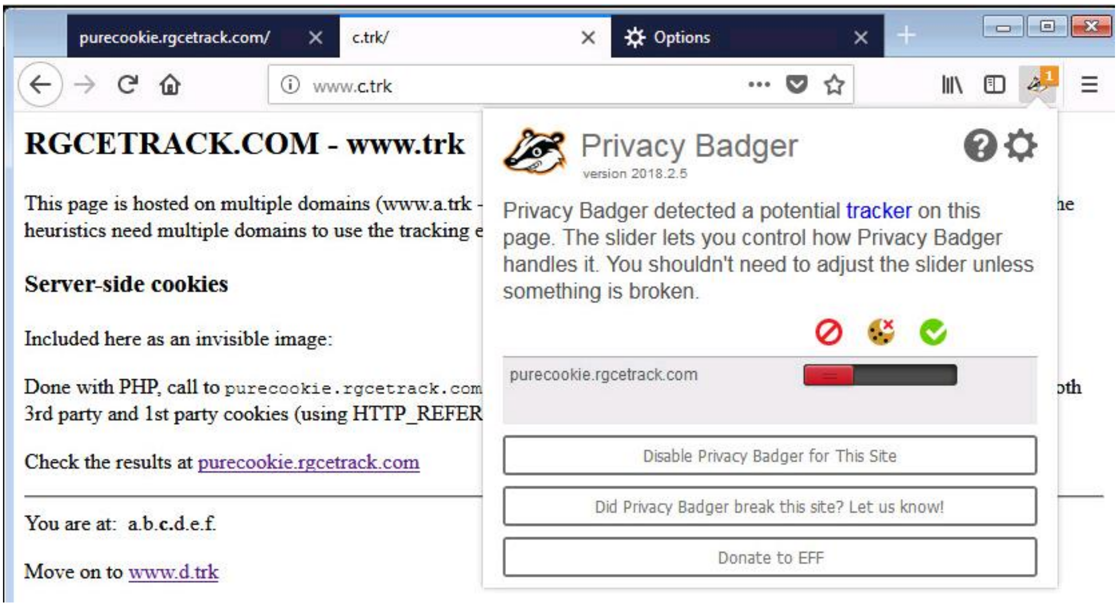
Disable Privacy Badger for This Site

Did Privacy Badger break this site? Let us know!

Donate to EFF

Figure 29. Privacy Badger has greenlisted yalando.com

To be able to demonstrate the heuristic behavior, a separate site set was created with different domains (www.a.trk – www.f.trk), which used the previously created third-party cookie site. This approach did not initially work as the system did only set a *sstrackid* cookie with an incrementing number for the clients and Privacy Badger requires somewhat entropy for the identifier (Privacy Badger Source Code 2018). By adding a second identifier, *sstrackmd5* which is the md5 hash of the identifier, enough entropy was found and Privacy Badger blocked the request on the third page of the site set (Figure 30).



RGCETRACK.COM - www.trk

This page is hosted on multiple domains (www.a.trk - he
heuristics need multiple domains to use the tracking e

Server-side cookies

Included here as an invisible image:

Done with PHP, call to purecookie.rgcetrack.com oth
3rd party and 1st party cookies (using HTTP_REFER

Check the results at purecookie.rgcetrack.com

You are at: a.b.c.d.e.f.

Move on to www.d.trk

Privacy Badger
version 2018.2.5

Privacy Badger detected a potential tracker on this page. The slider lets you control how Privacy Badger handles it. You shouldn't need to adjust the slider unless something is broken.

purecookie.rgcetrack.com

Disable Privacy Badger for This Site

Did Privacy Badger break this site? Let us know!

Donate to EFF

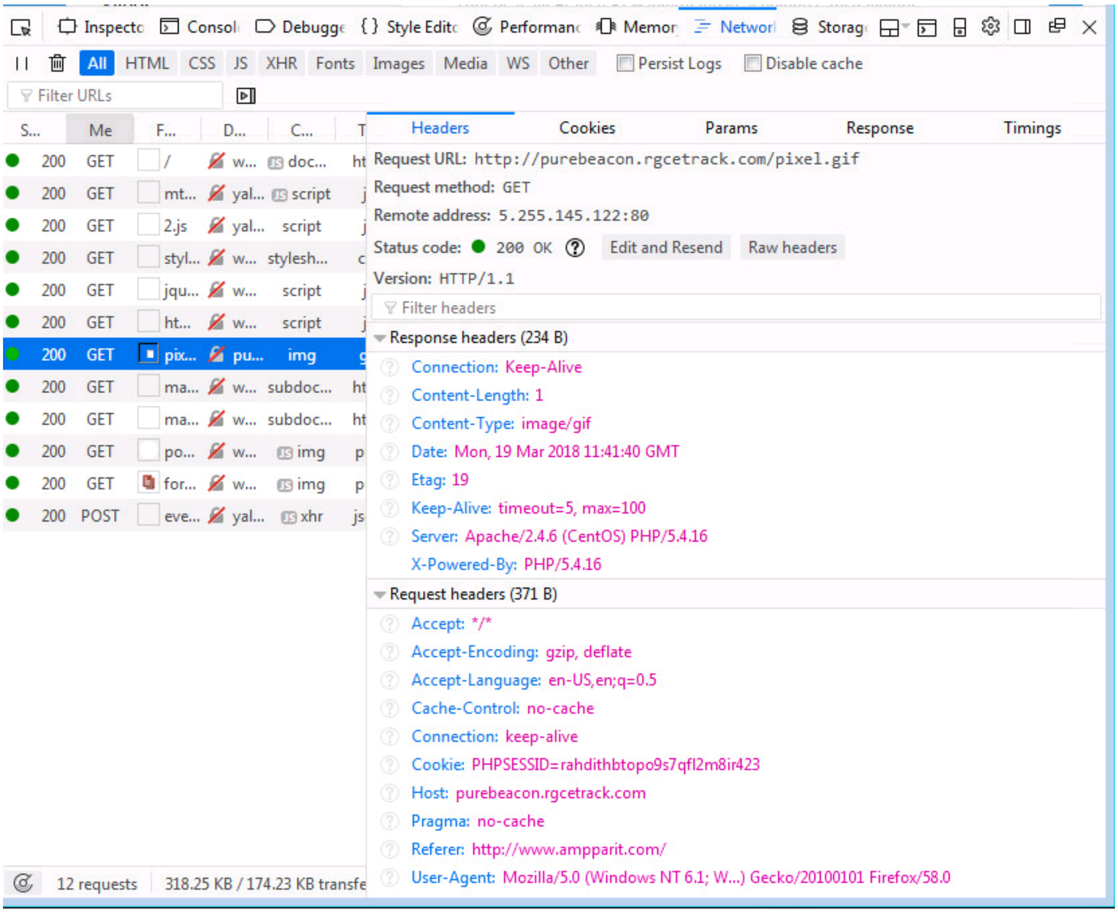
Figure 30. Privacy Badger and a redlisted third party cookie

Source code for the server-side scripts used is included in Appendix 1.

7.3 Beacons

Tracking beacon was already implemented in RGCE by Laboratory Engineer Kari Nurmi, with the help of ETags and PHP Session cookies. The site includes an invisible 1x1 GIF pixel that has an ETag value set according to the PHP Session cookie value created for the user. The ETag is formatted to look similar to a normal Apache webserver ETag value to further masquerade the tracking. PHP Sessions are used to hold user information in case the user forces the resource to refresh using Ctrl+Shift+R in the browser.

This system was used as a basis and a new site closely similar to the aforementioned server-side cookie page was created. ETag value was replaced by user id similarly to the cookie site for easier demonstration (Figure 31). The source code for the modified beacon system is included in Appendix 2. The beacon was then injected into few webpages in RGCE (ampparit.com and www.*.trk) for testing.



S...	Me	F...	D...	C...	T
● 200	GET	/	w...	doc...	ht
● 200	GET	mt...	yal...	script	j
● 200	GET	2.js	yal...	script	j
● 200	GET	styl...	w...	stylesh...	c
● 200	GET	jqu...	w...	script	j
● 200	GET	ht...	w...	script	j
● 200	GET	pix...	pu...	img	g
● 200	GET	ma...	w...	subdoc...	ht
● 200	GET	ma...	w...	subdoc...	ht
● 200	GET	po...	w...	img	p
● 200	GET	for...	w...	img	p
● 200	POST	eve...	yal...	xhr	js

12 requests 318.25 KB / 174.23 KB transfe

Request URL: http://purebeacon.rgcetrack.com/pixel.gif
Request method: GET
Remote address: 5.255.145.122:80
Status code: ● 200 OK
Version: HTTP/1.1

Response headers (234 B)
Connection: Keep-Alive
Content-Length: 1
Content-Type: image/gif
Date: Mon, 19 Mar 2018 11:41:40 GMT
Etag: 19
Keep-Alive: timeout=5, max=100
Server: Apache/2.4.6 (CentOS) PHP/5.4.16
X-Powered-By: PHP/5.4.16

Request headers (371 B)
Accept: */*
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.5
Cache-Control: no-cache
Connection: keep-alive
Cookie: PHPSESSID=rahdithbtopo9s7qfl2m8ir423
Host: purebeacon.rgcetrack.com
Pragma: no-cache
Referer: http://www.ampparit.com/
User-Agent: Mozilla/5.0 (Windows NT 6.1; W...) Gecko/20100101 Firefox/58.0

Figure 31. User id ETag value and PHP Session ID for the tracking pixel

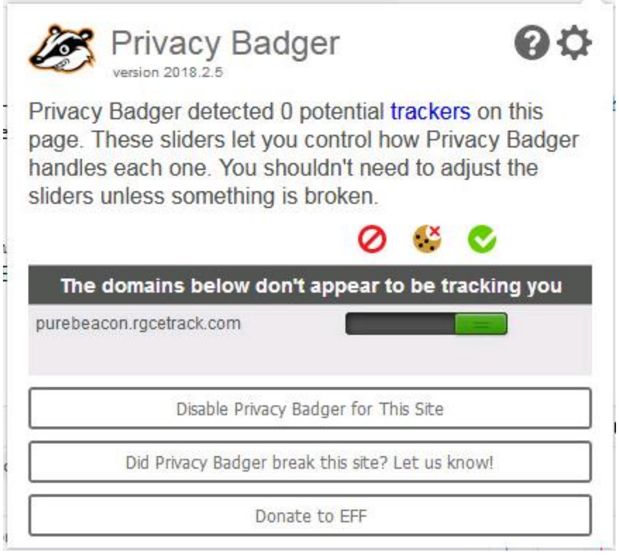
All general browsers worked the same, ETag was generated for the user and subsequent page visits returned HTTP 302 Not Modified for the tracking pixel. Forcing a hard refresh (Ctrl+Shift+R) resulted in HTTP 200 OK, however, as the session cookie was handled, user id stayed the same. Emptying just web cache or cookies did not reset the ID, yet, emptying both at the same time generated a new user id. Similar behavior was seen in private browsing. There was a small difference using Internet Explorer as the browser had to be closed in order for the cache data to really be deleted, other browsers removed the cache entries immediately after using the corresponding clear history –tool.

For the more privacy-oriented browsers, the results differed slightly. Brave did not set the HTTP Referer field correctly when sending the request to the tracking server, making all the requests seem to come from the tracking first-party. This was found out to be a feature in the third-party cookie blocking (Brave 2016). Brave Shields showed no tracking elements to be found and the cache was not automatically emptied on browser restart. Clean Reload or Ctrl+Shift+R did not work correctly, as the ETag-value still contained the same user id. Using the Session tab or Private tab feature did however generate a new id, so it seems that some cache separation is done on this level. The lack of developer console prevented further investigation on the issue.

Epic browser worked similarly to Chrome, however, but it has no private browsing. TorBrowser generated a new user id for every domain due to the Cross-Origin Identifier Unlinkability feature but ETag persisted under subdomains. As third-party cookies are denied by default, hard refresh did generate a new user id. Turning "Restrict third party cookies and other tracking data" feature off resulted in correctly set cookies and so ETag values were retained across domains.

Testing of plugins yielded similar results as with the cookie case, although no single clear method was found that would perform better than just clearing the browser cache and cookies. Ghostery, uBlock Origin and Disconnect.me did nothing unless the tracking domain was added manually to the blocklists. As both the ETag and cookie are set with server-side scripts, NoScript also did nothing. Forget Me Not did clear the cookie, but had no option to clear the cache after leaving the domain.

Privacy Badger did notice and deny the tracking, however, only due to the cookie being set. When tested with the cookie setting disabled, third-party content was recognized, yet it was not deemed as a tracking component (Figure 32). A GitHub issue (Privacy Badger 2017) was found, so blocking trackers using other methods than cookies is currently not implemented.



Privacy Badger
version 2018.2.5

Privacy Badger detected 0 potential trackers on this page. These sliders let you control how Privacy Badger handles each one. You shouldn't need to adjust the sliders unless something is broken.

The domains below don't appear to be tracking you

purebeacon.rgcetrack.com

Disable Privacy Badger for This Site
Did Privacy Badger break this site? Let us know!
Donate to EFF

Figure 32. Privacy badger did not detect ETag tracking

As the system did not implement reverse correlation from the ETag user id to the cookie, deleting cookies and requesting new ones created a new PHP Session and a new tracked user id. This could be easily implemented, however, it was left for future development.

7.4 Supercookies

Supercookies were implemented as a single page as the use case for this kind of tracking was mostly to demonstrate the different data storage options supercookies would use. Samy Kamkar (2010) has created a JavaScript project called evercookie, which uses multiple separate ways of storing data resiliently. Kamkar's evercookie script was used for the implementation as it was open source and required no further configuration except a working HTTP server. Some modifications were made to the script as it pointed to resources on the real Internet unavailable in the RGCE. The original code is also asynchronous, which required some changes on how the functions are called in order to demonstrate the evercookie in a better way. The index page for the demo site is included in Appendix 3.

Testing evercookie did not go as smoothly as planned. The browser APIs have changed a great deal since 2010 when Kamkar created the evercookie project and since it is not actively maintained, most of the features did not seem to work correctly. Some of the features were outright obsolete, such as Flash Local storage and Java as these plugins are no longer supported in current browsers. Microsoft Silverlight has also been obsoleted. This made cross-browser testing impossible and was subsequently dropped from the testing methodology.

Evercookie script uses the following storage methods for persistent cookie values:

- pngData: RGB values injected in PNG images using HTML5 similar to ETags
- etagData: ETag values
- cacheData: Using the HTTP request cache expiration as a cookie storage
- userData: Internet Explorer proprietary userData storage. Obsoleted in IE10.
- cookieData: Cookies
- globalData: Firefox-specific data storage, obsoleted from Firefox 9 onwards.
- localData and sessionData: HTML5 standard storage methods. globalData was
- windowData: Storing the window.name DOM element
- historyData: Injecting the tracking value into arbitrary HTTP requests and then reading the result using visited link status. Fixed back in 2010 in browsers.
- idbData: IndexedDB
- dbData: WebSQL database, obsolete and replaced by HTML IndexedDB
- lsoData: Flash Local Storage (not used in implementation)
- slData: Microsoft Silverlight (not used in implementation)

The demo site was constructed so that on the first visit it would try to request these storages the cookie value and if none of them would hold a value other than “unused”, it would then set the cookie in all possible storages. This way the first visit set the cookie and subsequent visits showed what storages were used. Persistence was tested with restarting the browser and clearing all history data.

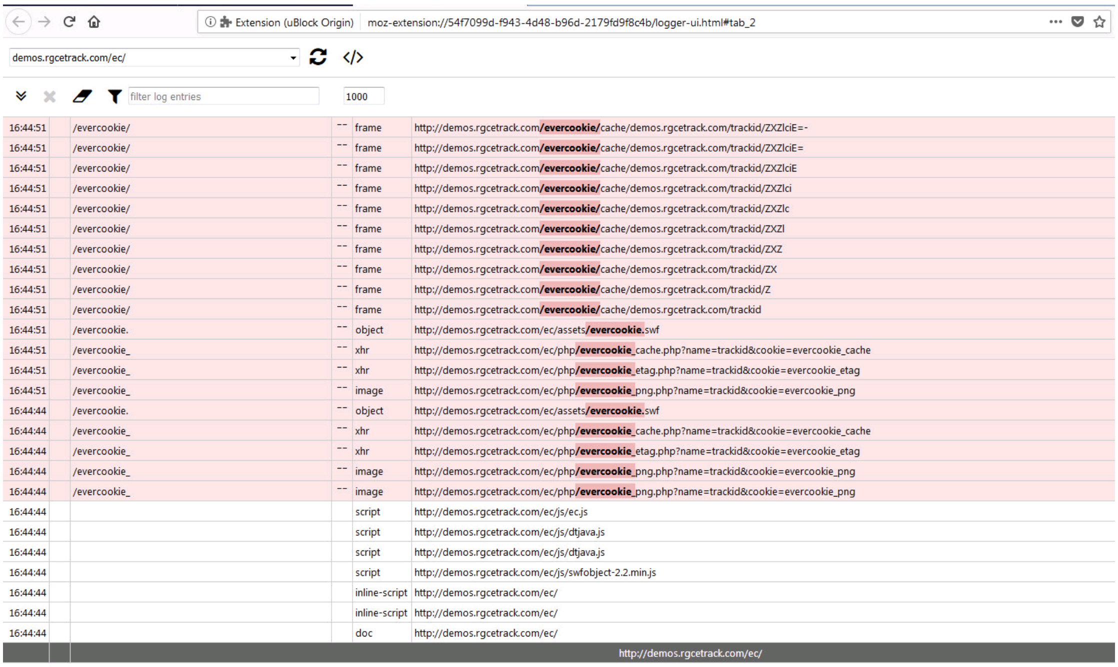
All browsers provided several different storage methods, shown in Table 1. All browsers also cleared every storage when clearing history data, which means the methods were not everlasting, although clearing just normal cache and cookies did leave residual storage objects behind.

Table 1. Working storage methods in different browsers tested

	Firefox	Chrome	IE	Brave	EPIC	Tor Browser
pngData		X		X		
etagData	X	X		X	X	X
cacheData		X	X	X	X	
userData						
cookieData	X	X	X	X	X	X
localData	X	X	X	X	X	X
globalData						
sessionData	X	X	X	X	X	X
windowData	X	X	X	X	X	
historyData						
idbData	X	X	X	X	X	
dbData		X		X	X	

As can be seen in the table, userData, globalData and historyData did not work on any of the browsers as they were either obsolete or fixed. dbData did still work on Chrome and Brave/EPIC which are based on Chrome source code even though it is obsolete. Chrome and Brave were the only ones that allowed RGB data caching using PNG images. Curiously, ETag cookies did not work in IE even though they did work when testing the Tracking Beacon ETags. TorBrowser was the only one that did not permit the script to read the value from window.name DOM data or IndexedDB.

Plugin results varied, Ghoster and Disconnect.me and Privacy Badger did nothing, however, Ublock Origin blocked the JavaScript file from loading due to the name containing “evercookie”, which is blacklisted in the built-in EasyPrivacy –list (Figure 33). Changing the script name and folder from evercookie to ec allowed the script to load.


Extension (uBlock Origin)
moz-extension://54f7099d-f943-4d48-b96d-2179fd9f8c4b/logger-ui.html#tab_2
demos.rgcetrack.com/ec/
filter log entries
1000

```
16:44:51  /evercookie/   --  frame         http://demos.rgcetrack.com/evercookie/cache/demos.rgcetrack.com/trackid/ZXZlciE=-
16:44:51  /evercookie/   --  frame         http://demos.rgcetrack.com/evercookie/cache/demos.rgcetrack.com/trackid/ZXZlciE=
16:44:51  /evercookie/   --  frame         http://demos.rgcetrack.com/evercookie/cache/demos.rgcetrack.com/trackid/ZXZlciE
16:44:51  /evercookie/   --  frame         http://demos.rgcetrack.com/evercookie/cache/demos.rgcetrack.com/trackid/ZXZlci
16:44:51  /evercookie/   --  frame         http://demos.rgcetrack.com/evercookie/cache/demos.rgcetrack.com/trackid/ZXZlc
16:44:51  /evercookie/   --  frame         http://demos.rgcetrack.com/evercookie/cache/demos.rgcetrack.com/trackid/ZXZl
16:44:51  /evercookie/   --  frame         http://demos.rgcetrack.com/evercookie/cache/demos.rgcetrack.com/trackid/ZXZ
16:44:51  /evercookie/   --  frame         http://demos.rgcetrack.com/evercookie/cache/demos.rgcetrack.com/trackid/ZX
16:44:51  /evercookie/   --  frame         http://demos.rgcetrack.com/evercookie/cache/demos.rgcetrack.com/trackid/Z
16:44:51  /evercookie/   --  frame         http://demos.rgcetrack.com/evercookie/cache/demos.rgcetrack.com/trackid
16:44:51  /evercookie.   --  object        http://demos.rgcetrack.com/ec/assets/evercookie.swf
16:44:51  /evercookie_   --  xhr           http://demos.rgcetrack.com/ec/php/evercookie_cache.php?name=trackid&cookie=evercookie_cache
16:44:51  /evercookie_   --  xhr           http://demos.rgcetrack.com/ec/php/evercookie_etag.php?name=trackid&cookie=evercookie_etag
16:44:51  /evercookie_   --  image         http://demos.rgcetrack.com/ec/php/evercookie_png.php?name=trackid&cookie=evercookie_png
16:44:44  /evercookie.   --  object        http://demos.rgcetrack.com/ec/assets/evercookie.swf
16:44:44  /evercookie_   --  xhr           http://demos.rgcetrack.com/ec/php/evercookie_cache.php?name=trackid&cookie=evercookie_cache
16:44:44  /evercookie_   --  xhr           http://demos.rgcetrack.com/ec/php/evercookie_etag.php?name=trackid&cookie=evercookie_etag
16:44:44  /evercookie_   --  image         http://demos.rgcetrack.com/ec/php/evercookie_png.php?name=trackid&cookie=evercookie_png
16:44:44  /evercookie_   --  image         http://demos.rgcetrack.com/ec/php/evercookie_png.php?name=trackid&cookie=evercookie_png
16:44:44                     script        http://demos.rgcetrack.com/ec/js/ec.js
16:44:44                     script        http://demos.rgcetrack.com/ec/js/dtjava.js
16:44:44                     script        http://demos.rgcetrack.com/ec/js/dtjava.js
16:44:44                     script        http://demos.rgcetrack.com/ec/js/swfobject-2.2.min.js
16:44:44                     inline-script http://demos.rgcetrack.com/ec/
16:44:44                     inline-script http://demos.rgcetrack.com/ec/
16:44:44                     doc           http://demos.rgcetrack.com/ec/
                                           http://demos.rgcetrack.com/ec/
```

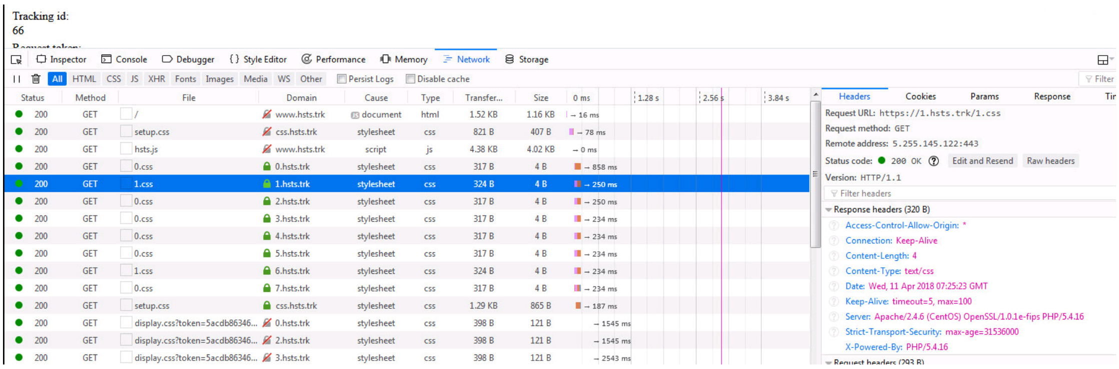
Figure 33. Ublock Origin blocks evercookie on keyword-basis

ForgetMeNot did not clear ETag storage as expected from testing with the Tracking Beacon, but it did clear all other storage values though. Still, using Firefox built-in option to delete all history data on browser closing works better.

To further test if evercookie could be used to demonstrate persistence older browsers were tested as there were base images in RGCE containing Firefox 36 and even Firefox 3. However, both versions yielded similar results and even with old versions of Flash enabled, no cross-browser persistence could be achieved. Cookie data was being set to Flash local storage, however, due to either browser security upgrades or code incompatibility, it could not be read from there. This effectively made the evercookie not viable for serious demonstrations unless further research is done on the subject.

7.5 HSTS

HTTP Strict-Transport-Security (HSTS) tracking was implemented both client- and server-side in the RGCE domain hsts.trk. Client-side demo uses Ben Friedlands proof-of-concept code from GitHub (Friedland 2016) and it is implemented using JavaScript. The JavaScript calls for several subdomain URLs, from which several set the HSTS header. This way, an N-bit identifier can be stored for the user. For server-side testing the code was created from scratch using methodology seen in the code of GitHub user "nevkontakte" (HSTS Super Cookie 2018), instead, it was implemented in PHP. The server-side demo serves the user dynamic CSS stylesheet files, which in turn import other CSS files under subdomains that either set or unset the HSTS max-age header (Figure 34). The main scripts for the server-side demo are included in Appendix 4.



Tracking id:
66
Request token:

Status	Method	File	Domain	Cause	Type	Transfer...	Size	0 ms
● 200	GET	/	www.hsts.trk	document	html	1.52 KB	1.16 KB	→ 16 ms
● 200	GET	setup.css	css.hsts.trk	stylesheet	css	821 B	407 B	→ 78 ms
● 200	GET	hsts.js	www.hsts.trk	script	js	4.38 KB	4.02 KB	→ 0 ms
● 200	GET	0.css	0.hsts.trk	stylesheet	css	317 B	4 B	→ 858 ms
● 200	GET	1.css	1.hsts.trk	stylesheet	css	324 B	4 B	→ 250 ms
● 200	GET	0.css	2.hsts.trk	stylesheet	css	317 B	4 B	→ 250 ms
● 200	GET	0.css	3.hsts.trk	stylesheet	css	317 B	4 B	→ 234 ms
● 200	GET	0.css	4.hsts.trk	stylesheet	css	317 B	4 B	→ 234 ms
● 200	GET	0.css	5.hsts.trk	stylesheet	css	317 B	4 B	→ 234 ms
● 200	GET	1.css	6.hsts.trk	stylesheet	css	324 B	4 B	→ 234 ms
● 200	GET	0.css	7.hsts.trk	stylesheet	css	317 B	4 B	→ 234 ms
● 200	GET	setup.css	css.hsts.trk	stylesheet	css	1.29 KB	865 B	→ 187 ms
● 200	GET	display.css?token=5acdb86346...	0.hsts.trk	stylesheet	css	398 B	121 B	→ 1545 ms
● 200	GET	display.css?token=5acdb86346...	2.hsts.trk	stylesheet	css	398 B	121 B	→ 1545 ms
● 200	GET	display.css?token=5acdb86346...	3.hsts.trk	stylesheet	css	398 B	121 B	→ 2543 ms

```
Request URL: https://1.hsts.trk/1.css
Request method: GET
Remote address: 5.255.145.122:443
Status code: ● 200 OK
Version: HTTP/1.1
Response headers (320 B)
    Access-Control-Allow-Origin: *
    Connection: Keep-Alive
    Content-Length: 4
    Content-Type: text/css
    Date: Wed, 11 Apr 2018 07:25:23 GMT
    Keep-Alive: timeout=5, max=100
    Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips PHP/5.4.16
    Strict-Transport-Security: max-age=31536000
    X-Powered-By: PHP/5.4.16
Request headers (293 B)
```

Figure 34. HSTS "bits" requested as css files

During testing, some variation was observed in the browser behaviour. Firefox saves the HSTS history in a profile-specific file (SiteSecurityServiceState.txt). Client-side JavaScript did not work at all in Internet Explorer 11, and the server-side method does not work in InPrivate session. Delete history works correctly in all the browsers. Brave and EPIC browsers will delete all history by default when exiting, yet, do not mitigate either of the methods during a session. TorBrowser was the only browser where neither client-side nor server-side method worked. Client-side JavaScript code generated a tracking ID for the user, however, it was different every time the page was refreshed. Server-side beacons did not load at all due to a security error when requested as third-party, which would point again to the Cross-Origin Identifier Unlinkability.

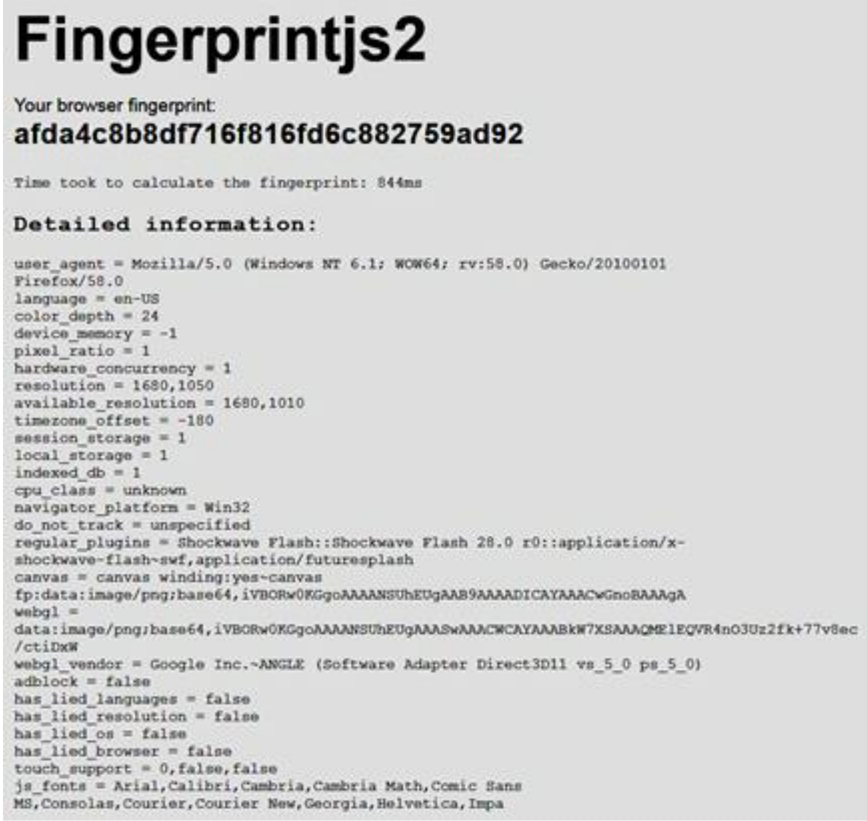
Not any of the plugins could be used to block the HSTS tracking from working, not even Forget Me Not, as it cannot remove the HSTS site data stored in Firefox profile. Privacy Badger did nothing even though the server-side code was included in multiple sites similar to previous methods as no traditional cookie is being set.

NoScript blocked the JavaScript but did nothing to the server-side method. A specific rule could be created for the tracking CSS, however, this is not a universal solution.

As HSTS relies on HTTP to HTTPS redirects, a further study was conducted in the effectiveness of a plugin called HTTPS Everywhere, made by Electronic Frontier Foundation (EFF). The plugin forces sites to be loaded via HTTPS, which could affect the way HSTS headers are handled, however, it was quickly found out that the plugin relies on a whitelist made by EFF, which again makes this not a universal solution (EFF 2018). Nor did the plugin include an option to modify the whitelist. Further analysis should be conducted to spoof the list for RGCE demonstrational purposes.

7.6 Fingerprinting

For the fingerprinting demonstration, the source code of Fingerprintjs2 was used (Vasilyev 2018). This library is commonly used for demonstrations and probably some actual user tracking also, as can be deducted from the uBlock Origin default filter list containing an entry pointing to this script. The library uses JavaScript and a clear enough index page for demonstration purposes, so it was used as is (Figure 35).



Fingerprintjs2

Your browser fingerprint:

afda4c8b8df716f816fd6c882759ad92

```
Time took to calculate the fingerprint: 844ms
```

Detailed information:

```
user_agent = Mozilla/5.0 (Windows NT 6.1; WOW64; rv:58.0) Gecko/20100101
Firefox/58.0
language = en-US
color_depth = 24
device_memory = -1
pixel_ratio = 1
hardware_concurrency = 1
resolution = 1680,1050
available_resolution = 1680,1010
timezone_offset = -180
session_storage = 1
local_storage = 1
indexed_db = 1
cpu_class = unknown
navigator_platform = Win32
do_not_track = unspecified
regular_plugins = Shockwave Flash::Shockwave Flash 28.0 r0::application/x-
shockwave-flash~swf,application/futuresplash
canvas = canvas winding:yes~canvas
fp:data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAB9AAAADICAYAAACwGnoBAAAgA
webgl =
data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAASwAAACWCAYAAABkW7XSAAAQME1EQVR4nO3Uz2fk+77v8ec
/ctiDxW
webgl_vendor = Google Inc.~ANGLE (Software Adapter Direct3D11 vs_5_0 ps_5_0)
adblock = false
has_lied_languages = false
has_lied_resolution = false
has_lied_os = false
has_lied_browser = false
touch_support = 0,false,false
js_fonts = Arial,Calibri,Cambria,Cambria Math,Comic Sans
MS,Consolas,Courier,Courier New,Georgia,Helvetica,Impa
```

Figure 35. Fingerprintjs2 initial results for Firefox

The script uses a number of data points provided by the browser, the most important being:

- Browser user agent
- Language and timezone
- Resolution, color depth and pixel ratio
- Plugin information
- Font information
- Canvas fingerprinting

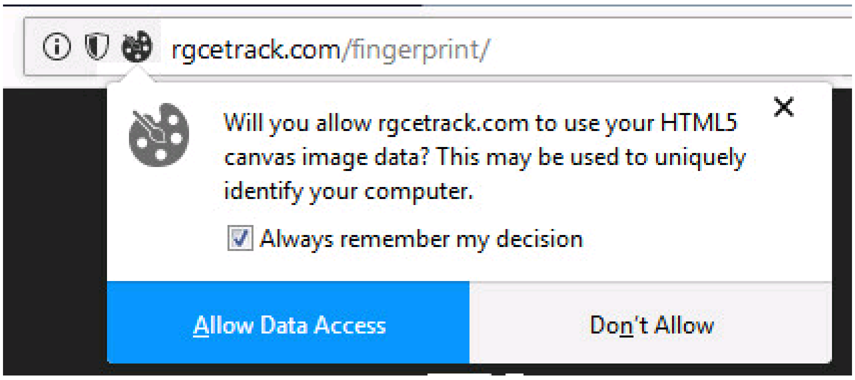
Ironically, the script also uses Do Not Track header as one of the tracking elements.

Testing was conducted differently from other methods, as the purpose of testing was not to avoid installation of tracking identifier to the client but to try to minimize the footprint seen from the client. All browsers generated a different identifier from each other as at least the user agent differed. Results in private browsing were different in Firefox and Internet Explorer, as the font information was not passed to the script in same way for some reason. Removing site-specific fonts from the browser settings resulted in the private browsing having the same fingerprint as in normal operation. As canvas fingerprinting relies on rendering, canvas data differed also on all browsers. It is hard to tell exactly what the difference was, as the data is binary-formatted, however, rendering engine is the main factor in the canvas fingerprinting.

Brave and TorBrowser managed to block some information from leaking to the fingerprint, however, Brave needed the option “Block all fingerprinting” to be set on instead of the default “Block third-party fingerprinting”, as this was not a third-party script. WebGL was blocked altogether and so no canvas fingerprint was generated. For Tor Browser, the timezone was seen as UTC and user agent the same as the original Firefox 52 ESR, which Tor Browser is based on. Tor Browsers user guide dictates that users should keep the window size as original (1000x900 pixels) in order to the fingerprint to stay the same. Interestingly, other browsers report the whole desktop resolution and not the viewport of the browser.

Turning plugins on in Firefox did not affect the fingerprint, which is because of Firefox preventing the enumeration of the plugins via JavaScript since version 28 (bugzilla 2012). Disconnect.me, Ghostery and Forget Me Not proved to be useless against fingerprinting, uBlock Origin blocked the resource initially due to "fingerprint2.js" being included in the default blocklist. Naming the script file to fp2.js instead was enough for the script to work again. NoScript disabled the whole script from being run by default. Privacy Badger has the option to block fingerprinting since version 1.0 (EFF 2015), but it did not recognize or block the fingerprinting attempt as it was seen as a first-party element. Including the fingerprinting code in multiple domains (www.*.trk from previous tests) as a third-party component resulted in Privacy Badger noticing the tracking element on the third try.

Lastly, as the Firefox includes an option to resist fingerprinting, it was also tested. The option *privacy.resistFingerprinting* was turned on from *about:config* and after this Firefox warned that the site was using canvas fingerprinting (Figure 36). The results also changed radically (Figure 37)

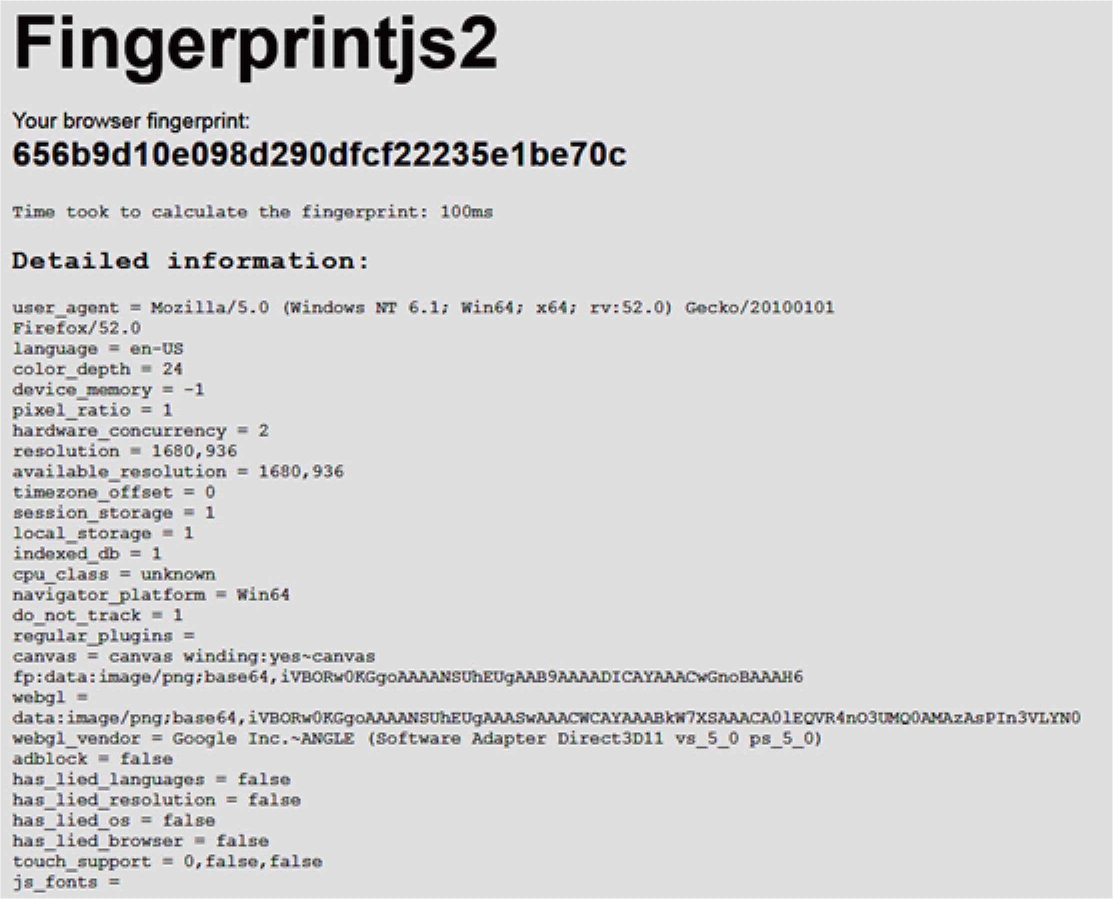


rgcetrack.com/fingerprint/

Will you allow rgcetrack.com to use your HTML5 canvas image data? This may be used to uniquely identify your computer.

☑ Always remember my decision

Figure 36. Firefox warns about canvas fingerprinting



Fingerprintjs2

Your browser fingerprint:

656b9d10e098d290dfcf22235e1be70c

```
Time took to calculate the fingerprint: 100ms

Detailed information:

user_agent = Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:52.0) Gecko/20100101
Firefox/52.0
language = en-US
color_depth = 24
device_memory = -1
pixel_ratio = 1
hardware_concurrency = 2
resolution = 1680,936
available_resolution = 1680,936
timezone_offset = 0
session_storage = 1
local_storage = 1
indexed_db = 1
cpu_class = unknown
navigator_platform = Win64
do_not_track = 1
regular_plugins =
canvas = canvas winding:yes~canvas
fp:data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAB9AAAADICAYAAACwGnoBAAAH6
webgl =
data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAASwAAACWCAYAAABkW7XSAAACA01EQVR4nO3UMQ0AMAzAsPIn3VLYN0
webgl_vendor = Google Inc.~ANGLE (Software Adapter Direct3D11 vs_5_0 ps_5_0)
adblock = false
has_lied_languages = false
has_lied_resolution = false
has_lied_os = false
has_lied_browser = false
touch_support = 0,false,false
js_fonts =
```

Figure 37. Fingerprintjs2 results for Firefox with resistFingerprinting on

As can be seen from the results, at least the following data was spoofed:

- User agent now points to Firefox 52, which is the same as in Tor Browser’s case
- Resolution now gives the viewport size instead of the display resolution
- Timezone offset is 0
- navigator_platform has changed from Win32 -> Win64
- regular_plugins value is empty
- canvas fingerprint is different (probably empty if the access request was denied)

This result is almost identical to the Tor Browser’s case, which is no surprise as Tor Browser uses the Firefox source code as a base. There is also a project called Tor Uplift that aims to implement Tor Browser patches on the mainline Firefox builds, so future Tor Browser versions can use newer Firefox source code (Mozilla 2018).

7.7 Network-level metadata injection

As all normal user tracking cases were covered, the focus turned onto network-level metadata injection like in the Verizon case (Mayer 2014). As RGCE already has a mockup of the Verizon network, a separate subnet was created for “Verizon Wireless” clients, which were to be part of this experiment. The topology is shown in Figure 38. Header injection was implemented with a Squid Transparent Forward Proxy (squid-cache wiki 2018) as the default gateway of the clients combined with a Python ICAP server created by Nikolay Ivanov (Ivanov 2017). ICAP stands for Internet Content Adaptation Protocol and can be used to rewrite portions of the HTTP requests passed to the ICAP server.

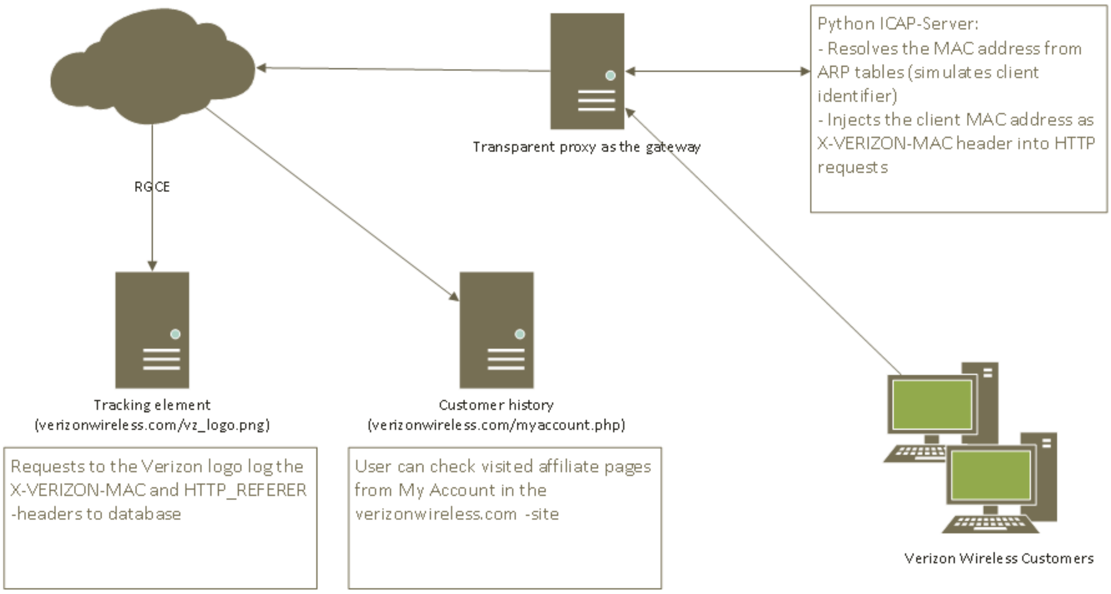
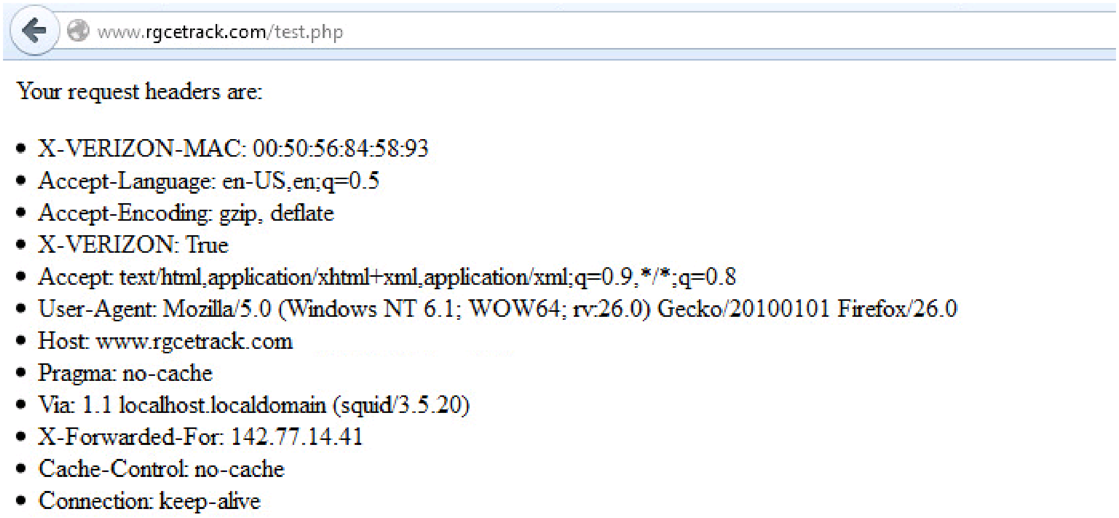

Transparent proxy as the gateway
Python ICAP-Server:
- Resolves the MAC address from ARP tables (simulates client identifier)
- Injects the client MAC address as X-VERIZON-MAC header into HTTP requests
RGCE
Tracking element
(verizonwireless.com/vz_logo.png)
Customer history
(verizonwireless.com/myaccount.php)
Requests to the Verizon logo log the X-VERIZON-MAC and HTTP_REFERER -headers to database
User can check visited affiliate pages from My Account in the verizonwireless.com -site
Verizon Wireless Customers

Figure 38. Metadata injection in RGCE Verizon Wireless network

The ICAP server was modified to inject the X-VERIZON and X-VERIZON-MAC header values into all HTTP requests. The first value was simply set as True and the latter one was resolved from the server’s ARP table, which contains the information on IP and MAC addresses. The MAC address here simulates a customer identifier, however, it could as easily be an IMEI, a DHCP unique identifier or even a customer number correlated from the ISP router information or customer database. The injected headers can be seen in Figure 39.

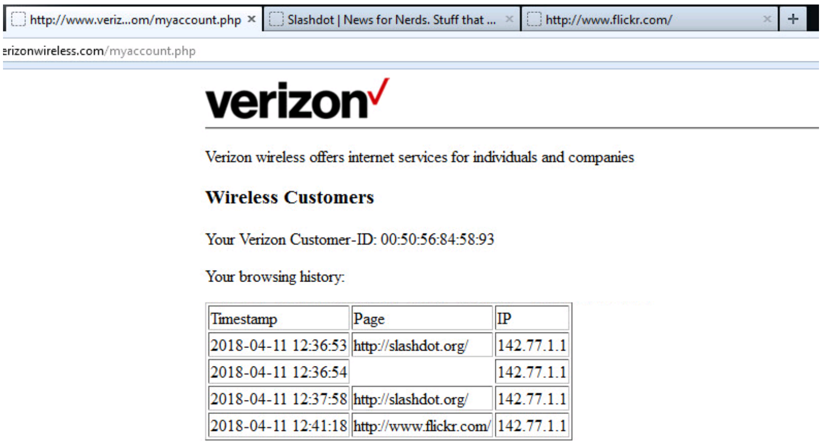


Your request headers are:

- X-VERIZON-MAC: 00:50:56:84:58:93
- Accept-Language: en-US,en;q=0.5
- Accept-Encoding: gzip, deflate
- X-VERIZON: True
- Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
- User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:26.0) Gecko/20100101 Firefox/26.0
- Host: www.rgcetrack.com
- Pragma: no-cache
- Via: 1.1 localhost.localdomain (squid/3.5.20)
- X-Forwarded-For: 142.77.14.41
- Cache-Control: no-cache
- Connection: keep-alive

Figure 39. Injected tracking headers in Verizon network

After confirming that headers were injected correctly, a tracking beacon was created similar to the one in the ETag case. This time the PHP code serving the requested image reads the X-VERIZON-MAC header value and uses that to log page visits. The tracking element can then be included in pages as a Verizon affiliate logo, and it should not be caught in any evasion method as no cookie, ETag or any other client-side tracking element is used. Lastly, an account information page was created (Figure 40) so the clients in RGCE Verizon Wireless network can check their logged pagevisits.


http://www.veriz...om/myaccount.php ×
Slashdot | News for Nerds. Stuff that ... ×
http://www.flickr.com/ ×
+
erizonwireless.com/myaccount.php
verizon✓

Verizon wireless offers internet services for individuals and companies

Wireless Customers

Your Verizon Customer-ID: 00:50:56:84:58:93

Your browsing history:

Timestamp	Page	IP
2018-04-11 12:36:53	http://slashdot.org/	142.77.1.1
2018-04-11 12:36:54		142.77.1.1
2018-04-11 12:37:58	http://slashdot.org/	142.77.1.1
2018-04-11 12:41:18	http://www.flickr.com/	142.77.1.1

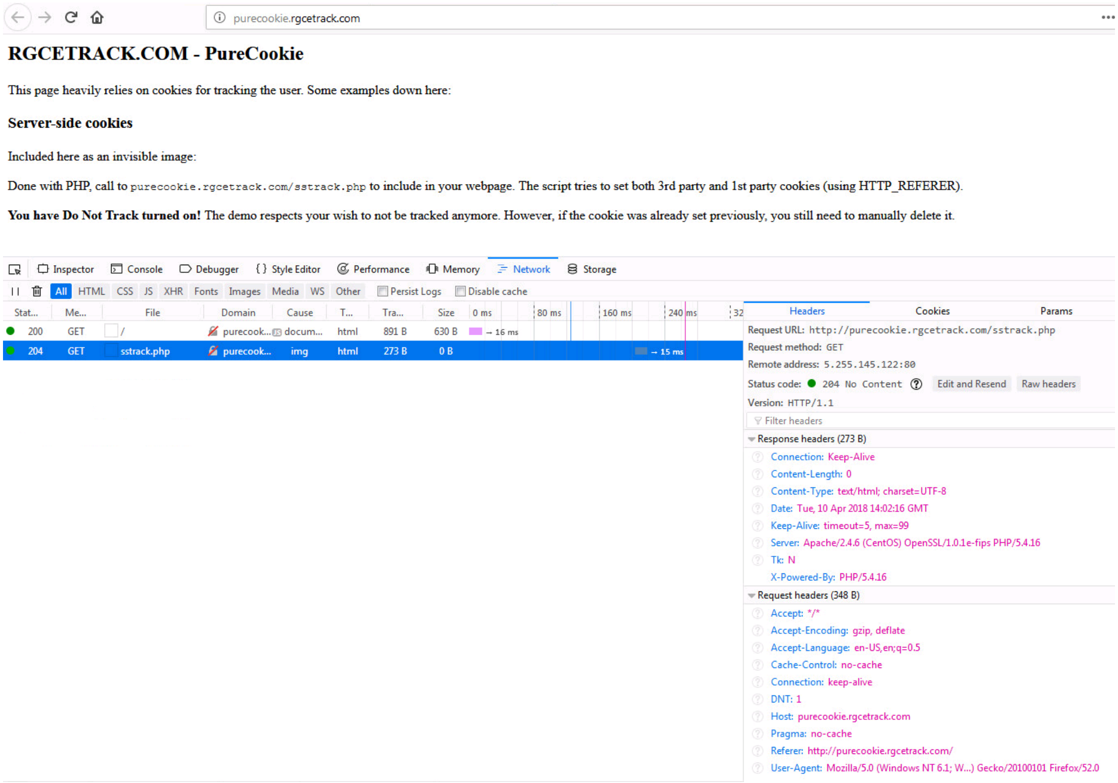
Figure 40. Account information page

7.8 Do Not Track demonstration

To prove that implementing Do Not Track is not very hard, server-side cookies from previous demonstrations were modified. As the DNT is just an HTTP Header value on a request, it was very simple to add handling for this header:

```
// If DNT is set, do not track
$req_headers = getallheaders();
if (isset($req_headers['DNT'])) {
            if ($req_headers['DNT'] == 1) {
                        // Set the Tk Header and no response code
                        http_response_code(204);
                        header('Tk: N');
                        die();
            }
}
// Set Tk header as tracked
header('Tk: T');
```

The code just checks if the “DNT” header value is set to 1, and then gives the result as HTTP Response code 204 No Content. Otherwise normal cookie operation resumes. Tk-header value is also set like described in W3C Tracking Preference Expression guide (W3C 2015). The result can be seen in Figure 41. Similar modifications can easily be implemented in any of the tracking elements.



RGCETRACK.COM - PureCookie

This page heavily relies on cookies for tracking the user. Some examples down here:

Server-side cookies

Included here as an invisible image:

Done with PHP, call to `purecookie.rgcetrack.com/sstrack.php` to include in your webpage. The script tries to set both 3rd party and 1st party cookies (using HTTP_REFERER).

You have Do Not Track turned on! The demo respects your wish to not be tracked anymore. However, if the cookie was already set previously, you still need to manually delete it.

Stat...	Me...	File	Domain	Cause	T...	Tra...	Size	
● 200	GET	/	purecook...	docum...	html	891 B	630 B	→ 16 ms
● 204	GET	sstrack.php	purecook...	img	html	273 B	0 B	→ 15 ms

Request URL: http://purecookie.rgcetrack.com/sstrack.php
Request method: GET
Remote address: 5.255.145.122:80
Status code: ● 204 No Content
Version: HTTP/1.1

Response headers (273 B)
Connection: Keep-Alive
Content-Length: 0
Content-Type: text/html; charset=UTF-8
Date: Tue, 10 Apr 2018 14:02:16 GMT
Keep-Alive: timeout=5, max=99
Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips PHP/5.4.16
Tk: N
X-Powered-By: PHP/5.4.16

Request headers (348 B)
Accept: */*
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.5
Cache-Control: no-cache
Connection: keep-alive
DNT: 1
Host: purecookie.rgcetrack.com
Pragma: no-cache
Referer: http://purecookie.rgcetrack.com/
User-Agent: Mozilla/5.0 (Windows NT 6.1; W...) Gecko/20100101 Firefox/52.0

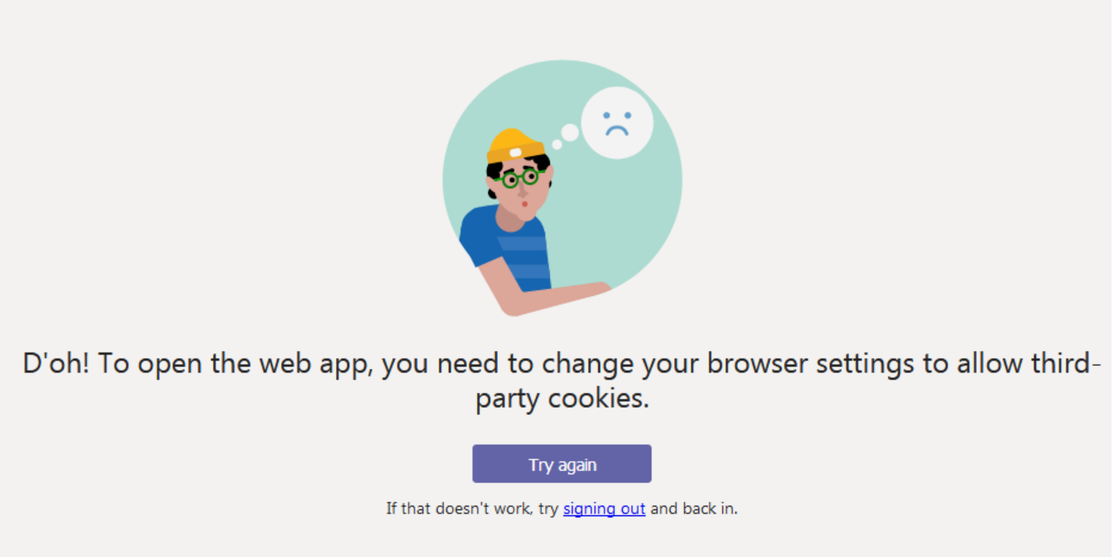
Figure 41. Cookie demo page respects Do Not Track requests

8 Results

8.1 Research results

The effectiveness of cookie blocking was somewhat inconclusive. First-party cookies can be used to skip most of the detection techniques and even though strict enough rules will block all cookies, the user is left with bad web experience or the need to always whitelist pages when needed. Blocklists cannot be used to block the first party cookies at all without breaking the functionality to the first-party site itself.

Third-party cookies, however, can be blocked effectively with any of the methods, however, for example the heuristics in Privacy Badger was a letdown as it requires enough entropy on the tracking id. As the entropy is checked against a hard-coded list (lowEntropyCookieValues) (Privacy Badger Source Code 2018), it should be possible to just divide the user identifier into smaller cookies and bypass this heuristic check. In addition, it still needs the cookie to be set as a third-party element. Additionally, many web applications, such as Microsoft Teams which is used in JAMK University of Applied Sciences, requires third-party cookies in order to function at all (Figure 42).



D'oh! To open the web app, you need to change your browser settings to allow third-party cookies.

If that doesn't work, try signing out and back in.

Figure 42. Third-party cookies required for Microsoft Teams

ETags, on the other hand, are simple to implement, and mimicking Apache ETag values can make them potentially invisible for the user. However, tracking the user purely with ETag values is not very robust and requires other methods (cookies, IP address) for correlation as hard refresh should always request a new value. Beacons can also use raw image data such as in Kamkar's evercookies (Kamkar 2010), which is a more robust way of storing data; however, as was tested, this is not an all-enduring solution. Further study should be made to see how forward proxies would handle Etags and if tracking could be implemented using various cache-control parameters.

HSTS Cookies were found to be a robust way of tracking the user, especially when using dynamically created CSS files that in turn load resources with HSTS bit set. However, as was pointed out in nevkontaktes implementation (HSTS Super Cookie 2018), it is a slow and very resource-intensive way to implement tracking as it requires an own HTTPS session for each bit of entropy. It also requires several different subdomains and possibly a wildcard certificate to be implemented.

In the fingerprinting case, it is almost impossible to verify the entropy of the user's fingerprints or to see how many other users have a similar fingerprint. Sites such as EFF's Panopticlick (EFF 2018b) can be helpful, however, ultimately the tracking organisations can use a different approach or alter the fingerprinting method slightly to provide different results. TorBrowser's approach to minimize the browser footprint does look promising though.

Metadata injection at the network level seemed to be the most intrusive, as there is almost no way for the user to block this or to see whether it is being done to them. Using HTTPS, VPN or TOR would be the only options here as no client-side data is retained. This kind of user tracking should anyway always be opt-in, as normal users may never even find out that they are being tracked.

8.2 Implementation results

All of the tracking methods were all relatively easy to implement even without comprehensive programming experience. All tracking methods were collected under the domain rgcetrack.com and separate documentation was provided for the assignor. The resulting tracking suite (Topology shown in Appendix 5) can be used for further teaching, researching and commercial training by the assignor.

As for the browser and evasion results, all of the browsers performed surprisingly similarly, and the selection of the browser is more dependent on user preference than anything else. If maximum privacy were be the goal, Tor Browser is clearly the choice, however, the other browsers would be enough for an average user when configured correctly. Clearing the history and caches often and using private browse sessions provides the easiest way to disable most of the user tracking. I would highly recommend following the development of Privacy Badger and the Tor Uplift project for future enhancements.

9 Quality analysis

Most of the resource material used for this thesis was obviously from the Internet, as technical aspects of web tracking can change very quickly. A great deal of the information for the methods of tracking and evasion was found by researching the source codes, which can be thought of as irrefutable, as the program cannot work in any other way. However, some information found on reasons for tracking (Gauss 2017) or hidden methods (Leonhard 2016) could not be verified even though the sources seemed reliable. Plenty of the information found was hearsay, conjecture or outright paranoia and was left out of this thesis. A great deal of the information was found on blogs and news articles about data breaches or vulnerabilities, such as the Verizon incident (Mayer 2014). These were cross-referenced with several other sites to verify the claims. Reddit was also a valuable tool to find out other people’s opinions on cases as such, although privacy-oriented subreddits can be difficult to traverse without absorbing too much paranoia.

The author found surprisingly little academic research on the subject, most of which consisted of already well-established methodologies (such as cookies) or how tracking affects humans subconsciously or hidden in the systems (e.g. price discrimination). Only a handful of actual research was found on evasion techniques and many of the whitepapers were very old related to the modern internet.

Some books were used to better understand the motives behind the user tracking and data collection, such as Schneier on Security (Schneier 2008) and Data and Goliath (Schneier 2015) by Bruce Schneier, which both include many of his blog posts used in this thesis as references. Schneier is considered to be one of the highly esteemed professionals in both computer security and privacy. One problem with Schneier's work, however, is that he concentrates a great deal on the issues in the United States, and many of the issues do not apply here in Europe or in third world countries or some, which concern more on the physical level such as air traffic safety.

The implementation and test methodology are no way perfect, as the writer is not a programmer nor a web developer. In many cases, more elegant solutions could have been used to make tracking methods more difficult to evade. Some evasion tools could also have been modified to implement features more suitable for RGCE and demonstration use, yet, they were left for further study. However, the methods depicted in this thesis should provide a good compilation on the most common ones for tracking and evasion.

10 Conclusions

The modern Internet has enabled us to express ourselves in more ways than ever. The use of social media is almost considered as a requirement for normal life. Voicing of opinions is easier than ever and so is the collection and monitoring of those opinions and the individuals behind them. However, one cannot stay private just by opting out of social media or by keeping to themselves anymore. By using the Internet, we publish a vast set of data about ourselves just by our actions themselves. Our browsing habits are recorded and monitored, our purchases can be tracked and our actions can be correlated to others, creating a profile or categorization for us. This profile or category can be resold and then used by a third party for any purpose they want, and the user has no control over it in any way. The only way to prevent this is to not use the Internet, however, this is futile as modern devices and operating systems track us also using telemetry hidden in their normal operations.

The evasion methods detailed in this thesis seem unfinished and have outright flaws. For example, the heuristic analysis of Privacy Badger was a huge letdown, as it required the cookie to be set as third party AND the entropy check is done via a hard-coded list (lowEntropyCookieValues). Brave seems more of a smoke-and-mirrors approach where ads and some third party tracking elements are blocked using outsourced lists, combined with hacks that are hastily collected and maintained in a not very organised way.

Clearly, an effort is being made to make browsers handle privacy better, which is good for the average user. Cross-Origin Identifier Unlinkability/First Party Isolation seems a very robust way to prevent tracking without breaking functionality that requires the use of third-party cookies. Firefox currently has this feature implemented, however, it is not active yet by default. Brave, Tor Browser and Firefox can block fingerprinting, which was once considered to be the ultimate tracking tool. The Tor Browser project is a fascinating effort to create an ultimate privacy-oriented browser and I like their approach the best, as they try to fix many of the grievances at the root of the problem without resorting to whitelisting or filters. Brave’s idea of replacing ads with a blockchain-based nanopayments seemed controversial, as purely from privacy standpoint the idea of replacing ads with other ads and implementing a tracking token to replace other ways of tracking does not seem like a benefit for the user, but the browser subsidiaries instead.

Some of the methods for tracking (such as using CSS and network-level metadata) and evasion (Safari hardening, First Party Isolation in Firefox) were implemented just in time during writing of this thesis. Several huge privacy incidents were also unearthed, such as the data collection from Facebook by Cambridge Analytica (Cadwalladr, Graham-Harrison 2018). As GDPR is just stepping into effect, privacy is currently a hot topic and both the way of tracking users and evading tracking and data collection methods will in no doubt change over the next few years. The biggest concern is that ultimately money runs the Internet, and as long as there is no collective organisation with vast resources behind privacy implementations, the focus will rest on providing more Big Data and user tracking, as it is better business.

References

Acar, G., Eubank, C., Englehardt, S., Juarez, M., Narayanan, A., Diaz, C. 2014. *The Web Never Forgets: Persistent Tracking Mechanisms in the Wild*. KU Leuven, ESAT/COSIC and iMinds, Leuven, Belgium and Princeton University.

AdNauseam. 2017. *AdNauseam banned from the Google Web Store.* Accessed on 1.3.2018. Retrieved from https://adnauseam.io/free-adnauseam.html

Allen, N. 2012. *Hollywood threatens to withdraw funding for Barack Obama over SOPA*. Article on The Telegraph. Accessed on 28.2.2018. Retrieved from https://www.telegraph.co.uk/news/worldnews/barackobama/9028611/Hollywood-threatens-to-withdraw-funding-for-Barack-Obama-over-SOPA.html

Borgesius, F. 2017. *Online price discrimination and EU data privacy law*. IViR Institute for Information Law, University of Amsterdam, The Netherlands. Journal of Consumer Policy, September 2017, Volume 40, Issue 3.

Brave. 2016. *Network requests from a script on a different domain have an incorrect referrer header.* Github source code issue. Accessed on 26.3.2017. Retrieved from https://github.com/brave/browser-laptop/issues/3067

Brave. 2018. What is Brave Ad Replacement. Accessed on 9.4.2018. Retrieved from https://www.brave.com/about-ad-replacement/

Brown, I. 2014. *Social media surveillance*. The International Encyclopedia of Digital Communication and Society.

bugzilla. 2012. Firefox bug report 757726: disallow enumeration of navigator.plugins. Accessed on 10.4.2018. Retrieved from https://bugzilla.mozilla.org/show_bug.cgi?id=757726

Bujlow, T., Carela-Español, V., Solé-Pareta, J., Barlet-Ros, P. 2015. *Web Tracking: Mechanisms, Implications, and Defenses.* Broadband Communications Research Group, Department of Computer Architecture, Universitat Politècnica de Catalunya, Barcelona, 08034, Spain.

Burke, S. 2016. *Analyzing GeForce Experience Data Transfers with Packet Monitoring*. Accessed on 18.10.2017. Retrieved from https://www.gamersnexus.net/industry/2672-geforce-experience-data-transfer-analysis

Böhmer, J. 2018. *Crooked Style Sheets*. Accessed on 1.3.2018. Retrieved from https://github.com/jbtronics/CrookedStyleSheets

Cadwalladr, C. Graham-Harrison, E. 2018. *Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach*. Accessed on 9.4.2018. Retrieved from https://www.theguardian.com/news/2018/mar/17/cambridge-analytica-facebook-influence-us-election

Couts, A. 2012. *Megaupload shut down by feds: Why do we need SOPA?* Article on Digitaltrends.com. Accessed on 28.2.2018. Retrieved from https://www.digitaltrends.com/web/megaupload-shut-down-by-feds-why-do-we-need-sopa/

Disconnect.me. 2018. *Disconnect*. Accessed on 28.2.2018. Retrieved from https://disconnect.me/disconnect

Economist, The. 2017. *The world’s most valuable resource is no longer oil, but data* Accessed on 9.4.2018. Retrieved from https://www.economist.com/news/leaders/21721656-data-economy-demands-new-approach-antitrust-rules-worlds-most-valuable-resource

EFF. 2015. *Privacy Badger*. Accessed on 1.3.2018. Retrieved from https://www.eff.org/privacybadger

EFF. 2017. *Do Not Track*. Accessed on 16.10.2017. Retrieved from https://www.eff.org/issues/do-not-track

EFF. 2018. *HTTPS Everywhere*. Accessed on 9.4.2018. Retrieved from https://www.eff.org/https-everywhere

EFF. 2018b. *Panopticlick*. Fingerprinting test site. Accessed on 10.4.2018. Retrieved from https://panopticlick.eff.org/

European Parliament and the Council. 2002. Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications). Accessed on 24.9.2017. Retrieved from http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32002L0058:EN:HTML

European Parliament and the Council. 2009. DIRECTIVE 2009/136/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 25 November 2009 amending Directive 2002/22/EC on universal service and users' rights relating to electronic communications networks and services, Directive 2002/58/EC concerning the processing of personal data and the protection of privacy in the electronic communications sector and Regulation (EC) No 2006/2004 on cooperation between national authorities responsible for the enforcement of consumer protection laws. Accessed on 16.10.2017. Retrieved from http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2009:337:0011:0036:en:PDF

Finlex. 1999. *Henkilötietolaki [Law on personal information]*. Accessed on 24.9.2017. Retrieved from http://www.finlex.fi/fi/laki/ajantasa/1999/19990523

Foster, M. 2014. *CBS - 60 Minutes: The Data Brokers (Miners)*. Documentary on CBS, Published by Red Team Cyber Security on Youtube. Retrieved from: https://www.youtube.com/watch?v=wVZffBzwq90

Friedland, B. 2016. *HSTS Super Cookie*. Source code on GitHub. Accessed on 9.4.2018. Retrieved from https://github.com/ben174/hsts-cookie

Gauss, R. 2017. *IPv6 and Ad Serving*. Blogpost on Aerserv advertising. Accessed on 17.10.2017. Retrieved from https://www.aerserv.com/ipv6-and-ad-serving/

Google. 2017a. *Cookie Matching*. Accessed on 17.10.2017. Retrieved from https://developers.google.com/ad-exchange/rtb/cookie-guide

Google. 2017b. *Advertising ID*. Accessed on 18.10.2017. Retrieved from https://support.google.com/googleplay/android-developer/answer/6048248?hl=en

Google. 2018. Adding analytics.js to Your Site. Accessed on 1.3.2018. Retrieved from https://developers.google.com/analytics/devguides/collection/analyticsjs/

Greenwald, G. 2014. *Why privacy matters*. TED Talk. Accessed on 9.9.2017. Retrieved from https://www.ted.com/talks/glenn_greenwald_why_privacy_matters#t-949479

Groat, S., Dunlop, M., Marchany, R., Tront, J. 2011. *IPv6: Nowhere to Run, Nowhere to Hide*. Bradley Department of Electrical and Computer Engineering, Virginia Tech Information Technology Security Office, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA

Hirahara, S. 2017. All About Ad Blocking. Documentation about Brave Shields functionality. Accessed on 14.3.2018. Retrieved from https://community.brave.com/t/all-about-ad-blocking/10004

Howe, D., Nissenbaum, H. 2017. *Engineering Privacy and Protest: a Case Study of AdNauseam*. School of Creative Media City University, Hong Kong. Cornell Tech New York University.

HSTS Super Cookie. 2018. *Javascript-less HSTS super-cookie PoC*. Accessed on 9.4.2018. Retrieved from http://hsts.nevkontakte.com/

Ivanov, N. 2017. *icapserver*. Github source code. Accessed on 9.4.2017. Retrieved from https://github.com/Peoplecantfly/icapserver

Jackson, W. 2012. *Why the FBI wants IPv6: It's better for tracking criminals*. Article at GCN. Accessed on 17.10.2017. Retrieved from https://gcn.com/articles/2012/06/07/fbi-wants-ipv6-hard-to-track-ipv4-with-nat.aspx

JYVSECTEC - Jyväskylä Security Technology. 2017. *JYVSECTEC CYBER RANGE. RGCE and solutions.* Accessed on 25.4.2018. Retrieved from https://jyvsectec.fi/wp-content/uploads/2017/02/JYVSECTEC-cyber-range.pdf

Kamkar, S. 2010. *evercookie*. Accessed on 17.10.2017. Retrieved from: https://samy.pl/evercookie/

Keizer, G. 2015. *Microsoft rolls back commitment to Do Not Track*. Computerworld news article. Accessed on 16.10.2017. Retrieved from https://www.computerworld.com/article/2905551/microsoft-rolls-back-commitment-to-do-not-track.html

Kelly, K. 2006. *More anonymity is good*. Accessed on 9.9.2017. Retrieved from https://www.edge.org/q2006/q06_4.html

Kyle Z v. Bose Corp. 2017. CLASS ACTION COMPLAINT AND DEMAND FOR JURY TRIAL. Case No. 17-cv-2928. UNITED STATES DISTRICT COURT FOR THE NORTHERN DISTRICT OF ILLINOIS, EASTERN DIVISION.

Leonhard, W. 2016. *Microsoft previews telemetry push with new Win7/8.1 patches KB 3192403, 3192404*. Computerworld news article. Accessed on 28.2.2018. Retrieved from https://www.computerworld.com/article/3132377/microsoft-windows/microsoft-previews-telemetry-push-with-new-win781-patches-kb-3192403-3192404.html

Lessig, L. 2004. *CODE version 2*. New York: Basic Books

Lyon, D. 2009. *Surveillance, Power and Everyday Life*. Oxford Handbook of Information and Communication Technologies

Macrumors. 2018. *Ad Firms Hit Hard by Apple's Intelligent Tracking Prevention Feature in Safari*. Article in the Macrumors blog. Accessed on 28.2.2018. Retrieved from https://www.macrumors.com/2018/01/09/ad-firms-hit-hard-by-safari-tracking-prevention/

Mautic. 2018. *Manipulating contacts.* Mautic documentation. Accessed on 14.3.2018. Retrieved from https://github.com/mautic/developer-documentation/blob/master/source/includes/_plugin_manipulating_contacts.md

Mayer, J. 2011. *Tracking the trackers: Microsoft Advertising*. Article in The Center for Internet and Society at Stanford Law School. Accessed on 17.10.2017. Retrieved from http://cyberlaw.stanford.edu/blog/2011/08/tracking-trackers-microsoft-advertising

Mayer, J. 2014. How Verizon's Advertising Header Works. Blogpost. Accessed on 18.10.2017. Retrieved from http://webpolicy.org/2014/10/24/how-verizons-advertising-header-works/

McCallister, E., Grance, T., Scarfone, E. 2010. *Guide to Protecting the Confidentiality of Personally Identifiable Information (PII)*. Recommendations of the National Institute of Standards and Technology (NIST)

MDN. 2017a. *Mozilla Developers Network - Beacon API*. Accessed on 17.10.2017. Retrieved from https://developer.mozilla.org/en-US/docs/Web/API/Beacon_API

MDN. 2017b. *Mozilla Developers Network - ETag*. Accessed on 17.10.2017. Retrieved from https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/ETag

Miners, Z. 2014. *Internet 'Do Not Track' system is in shatters*. Computerworld news article. Accessed on 16.10.2017. Retrieved from https://www.computerworld.com/article/2489727/data-privacy/internet--do-not-track--system-is-in-shatters.html

Mozilla. 2018. *Tor Uplift Project*. Accessed on 10.4.2018. Retrieved from https://wiki.mozilla.org/Security/Tor_Uplift

Muir, B. 2015. *Windows 10 – Microsoft Edge Browser Forensics*. Accessed on 28.2.2018. Retrieved from https://bsmuir.kinja.com/windows-10-microsoft-edge-browser-forensics-1733533818

Narayanan, A., Shmatikov, V. 2008. *Robust De-anonymization of Large Datasets*. The University of Texas at Austin.

Neustrom, P. 2017. *Want to see something crazy?.* Blog post at Medium. Accessed on 17.10.2017. Retrieved from https://medium.com/@philipn/want-to-see-something-crazy-open-this-link-on-your-phone-with-wifi-turned-off-9e0adb00d024

Perry, M., Clark, E., Murdoch, S., Koppen, G. 2018. *The Design and Implementation of the Tor Browser [DRAFT]*. Accessed on 14.3.2018. Retrieved from https://www.torproject.org/projects/torbrowser/design

Petri, D. 2016. Windows 10 Ignoring the Hosts File for Specific Name Resolution. Accessed on 1.3.2018. Retrieved from https://www.petri.com/windows-10-ignoring-hosts-file-specific-name-resolution

Platform for Privacy Preferences (P3P) Project. 2007. Accessed on 16.10.2017. Retrieved from https://www.w3.org/P3P/

Privacy Badger. 2017. *Investigate blocking evercookie-like tracking.* Github source code issue. Accessed on 26.3.2018. Retrieved from https://github.com/EFForg/privacybadger/issues/1558

Privacy Badger Source Code. 2018. Source code of Privacy Badger for Pale Moon. Accessed on 14.3.2018. Retrieved from https://github.com/JustOff/privacy-badger-me

Privacy International. 2017. *Metadata*. Accessed on 17.10.2017. Retrieved from https://www.privacyinternational.org/node/53

RFC3972. 2005. *Cryptographically Generated Addresses (CGA)*. IETF Request for Comments.

RFC4941. 2007. *Privacy Extensions for Stateless Address Autoconfiguration in IPv6*. IETF Request for Comments.

RFC7217. 2014. *A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC).* IETF Request for Comments.

Schneier, B. 2005. *Authentication and Expiration*. IEEE Security & Privacy.

Schneier, B. 2006a. *The Eternal Value of Privacy*. Accessed on 9.9.2017. Retrieved from https://www.wired.com/2006/05/the-eternal-value-of-privacy/

Schneier, B. 2006b. *Casual Conversations, R.I.P.* Accessed on 9.9.2017. Retrieved from https://www.forbes.com/2006/10/18/nsa-im-foley-tech-security-cx_bs_1018security.html

Schneier, B. 2006c. *Facebook and Data Control.* Accessed on 24.9.2017. Retrieved from https://www.schneier.com/blog/archives/2006/09/facebook_and_da.html

Schneier, B. 2006d. *TrackMeNot*. Accessed on 1.3.2018. Retrieved from https://www.schneier.com/blog/archives/2006/08/trackmenot_1.html

Schneier, B. 2007. *Risks of Data Reuse*. Accessed on 24.9.2017. Retrieved from https://www.schneier.com/blog/archives/2007/06/risks_of_data_r.html

Schneier, B. 2008. Schneier on Security. Wiley and Sons.

Schneier, B. 2009. *Beyond Security Theater*. Accessed on 28.2.2018. Retrieved from https://www.schneier.com/blog/archives/2009/11/beyond_security.html

Schneier, B. 2015. Data and Goliath. New York: W. W. Norton & Company

Solove, D. 2004. *The Digital Person - Technology and privacy in the information age*. New York: New York University Press

Soni, D. 2017. *What is the Difference Between SSP & Ad Exchange?* Accessed on 28.2.2018. Retrieved from http://www.dheerajsoni.com/difference-ssp-ad-exchange/

squid-cache wiki. 2018. *Linux traffic Interception using DNAT*. Accessed on 9.4.2018. Retrieved from https://wiki.squid-cache.org/ConfigExamples/Intercept/LinuxDnat

Statt, N. 2017. *Advertisers are furious with Apple for new tracking restrictions in Safari 11*. The Verge news article. Accessed on 16.10.2017. Retrieved from https://www.theverge.com/2017/9/14/16308138/apple-safari-11-advertiser-groups-cookie-tracking-letter

Stockley, M. 2015. *Anatomy of a browser dilemma – how HSTS 'supercookies' make you choose between privacy or security.* Accessed on 1.3.2018. Retrieved from https://nakedsecurity.sophos.com/2015/02/02/anatomy-of-a-browser-dilemma-how-hsts-supercookies-make-you-choose-between-privacy-or-security/

Spotify Privacy Settings. 2017. Accessed on 23.9.2017. Retrieved from https://support.spotify.com/lt/article/privacy-settings/

Telegraph. 2009*. Facebook shuts down Beacon*. News article. Accessed on 17.10.2017. Retrieved from http://www.telegraph.co.uk/technology/facebook/6214370/Facebook-shuts-down-Beacon.html

Vasilyev, V. 2018. *Fingerprintjs2*. Github source code. Accessed on 10.4.2018. Retreived from https://github.com/Valve/fingerprintjs2

W3C. 2015. *Tracking Preference Expression (DNT).* Accessed on 16.10.2017. Retrieved from https://www.w3.org/TR/tracking-dnt/

Webkit.org. 2018. *Intelligent Tracking Prevention*. Accessed on 28.2.2018. Retrieved from https://webkit.org/blog/7675/intelligent-tracking-prevention/

Westin, A. 1967. *Privacy and Freedom.* New York: Atheneum

YourAdChoices. 2017. *DAA Participating Companies & Organizations*. Accessed on 16.10.2017. Retrieved from http://youradchoices.com/

X

Appendices

Appendix 1. Source code for the server-side third party cookie example

sstrack.php

```
<?php

include "../sql_settings.php";

// Open SQL connection

$conn = new mysqli($sql_host, $sql_user, $sql_pass, $sql_db);
if ($conn->connect_error) {
            die("Could not connect to database - ". $conn->connect_error);
}

// Server-side tracking by PHP cookies

// Set cookie
if (! isset($_COOKIE["sstrackid"] )) {
            // Create new id and save to sql
            $useragent = $_SERVER['HTTP_USER_AGENT'];
            $sql = "INSERT INTO user (useragent) VALUES ('$useragent');";
            $result = $conn->query($sql);
            $trackid = $conn->insert_id;
            // Set both 1st and 3rd party cookie for maximum resilience
            // 1st party can be used later for the JS example
            setcookie("sstrackid", $trackid, time() +
60*60*24*365,"rgcetrack.com");
            setcookie("sstrackid", $trackid, time() +
60*60*24*365,$_SERVER["HTTP_REFERER"]);

            // To demonstrate Privacy badger heuristics, we need more entropy
            // So create another cookie that has an md5 hash of the value
            setcookie("sstrackmd5", md5($trackid), time()
+60*60*24*365,"rgcetrack.com");
            setcookie("sstrackmd5", md5($trackid), time()
+60*60*24*365,$_SERVER["HTTP_REFERER"]);

} else {
            $trackid = $_COOKIE['sstrackid'];
}

$ip = $_SERVER['REMOTE_ADDR'];
$referer = $_SERVER['HTTP_REFERER'];

date_default_timezone_set('UTC');
$phptime = new DateTime("now");
$phptime->setTimezone(new DateTimeZone('UTC'));
$mysqldate = $phptime->format('Y-m-d H:i:s');

// Then add the pagevisit to seen
$sql = "INSERT INTO seen (uid,timestamp,ipaddress,referer) VALUES
($trackid,'$mysqldate','$ip','$referer');";
$result = $conn->query($sql);

$conn->close();

?>
```

Inclusion of the script as a non-visible image in a page:

```
<img src="sstrack.php" style="display: none;">
```

SQL clauses for the tracking database:

```
drop table if exists seen;
drop table if exists user;

create table user (id int not null auto_increment primary key, useragent
varchar(255));
create table seen (id int not null auto_increment primary key, uid int null,
timestamp datetime, ipaddress varchar(32), referer varchar(255),
            foreign key (uid) references user(id));
```

Appendix 2. Source code for the tracking beacon

pixel.gif: (Apache should have AddType application/x-httpd-php .gif in order to run the PHP code when this “image” is requested)

```
<?php

include "../sql_settings.php";

// Open SQL connection

$conn = new mysqli($sql_host, $sql_user, $sql_pass, $sql_db);
if ($conn->connect_error) {
            die("Could not connect to database - ". $conn->connect_error);
}

// Tracking via ETag or sessions

// Start the session using cache_limiter - Disables pragma: no-cache which
will break the ETag
session_cache_limiter('private_no_expire:');
session_start();
if (isset($_SESSION['trackid'])) {
            $trackid = $_SESSION['trackid'];
}

// Old visits should have the trackid set in the ETag (HTTP_IF_NONE_MATCH set
in the request headers)
if (isset($_SERVER['HTTP_IF_NONE_MATCH'])) {
            $trackid = $_SERVER['HTTP_IF_NONE_MATCH'];
} else {
            // Check the session data first to avoid duplicates on
Ctrl+Shift+R
            if (!isset($trackid)) {
                        $useragent = $_SERVER['HTTP_USER_AGENT'];
                        // Create new id and save to sql
                        $sql = "INSERT INTO user (useragent) VALUES
('$useragent');";
                        $result = $conn->query($sql);
                        // Set the Etag to include the id - very crude but
good for demos
                        $trackid = $conn->insert_id;
            }
            // Otherwise the trackid is already set in the session - no
further action required
}

$ip = $_SERVER['REMOTE_ADDR'];
$referer = $_SERVER['HTTP_REFERER'];

date_default_timezone_set('UTC');
$phptime = new DateTime("now");
$phptime->setTimezone(new DateTimeZone('UTC'));
$mysqldate = $phptime->format('Y-m-d H:i:s');

// Then add the pagevisit to seen
```

```
$sql = "INSERT INTO seen (uid,timestamp,ipaddress,referer,method) VALUES
($trackid,'$mysqldate','$ip','$referer','ETAG_PIXEL');";
$result = $conn->query($sql);

$conn->close();

// Finally, return the ETag and the real pixel image
//echo $trackid;
//header('Etag: "' . $trackid . '"');
//header('Etag: ' . $_SESSION['trackid'] );
$_SESSION['trackid'] = $trackid;
header('Etag: ' . $trackid );
//if (isset($_SERVER['HTTP_IF_NONE_MATCH']) && ($_SESSION['trackid'] ==
$_SERVER['HTTP_IF_NONE_MATCH'])) {
if (isset($_SERVER['HTTP_IF_NONE_MATCH']) && ($trackid ==
$_SERVER['HTTP_IF_NONE_MATCH'])) {
            http_response_code(304);
} else {
            header("Content-Type: image/gif");
            echo file_get_contents("realpixel.gif");
}


?>
```

Appendix 3. Index page for the evercookie demonstration page

```
<script type="text/javascript" src="js/swfobject-2.2.min.js"></script>
<script type="text/javascript" src="js/dtjava.js"></script>
<script type="text/javascript" src="js/evercookie.js"></script>

<body>
<h3>RGCETrack.com - Evercookie</h3>

<p>Current Cookie value: </p><p id="cookieval">Please wait...</p>
<p id="description"></p>

<script>

// callback function for the ec.get
function checkCookie(best_candidate, all_candidates) {
            var best_candidate;
            // Since the script seems to run asynchronously, change the HTML
in a page p-element instead of a variable
            //document.getElementById("cookieval").innerHTML = value;
            if (!(best_candidate == "ever!")) {
                        ec.set("trackid", "ever!");
                        document.getElementById("cookieval").innerHTML =
"ever! (Just set)";
                        document.getElementById("description").innerHTML = "A
new cookie has been set. Please refresh the page to see results.";
            } else {
                        // Show stuff
                        document.getElementById("cookieval").innerHTML =
best_candidate;
                        for (var item in all_candidates)

            document.getElementById("description").innerHTML += "Storage
mechanism " + item +
                                                " returned: " +
all_candidates[item] + "<br/>\n";
            };

}

// Return all possible storages
function getCookie(best_candidate, all_candidates)
{
            alert("The retrieved cookie is: " + best_candidate + "\n"+
                        "all_candidates object has been iterated and will be
shown next in the page, explaining all possible storage types.");

}

// IndexedDB breaks something?
// Also something causes a loop?
var ec = new evercookie({
            baseurl: '/evercookie',
            asseturi: '/assets',
            phpuri: '/php',
            java: false
//          idb: true,
//          hsts: true,
//          db: true,
//          silverlight: false,
//          lso: false,
//          java: false,
//          history: true



            /* Options */
});

// Check the cookie and act accordingly
ec.get("trackid", checkCookie);

</script>

<button onClick='ec.set("trackid","undefined")'>Remove cookie</button>

</body>
```

Appendix 4. HSTS Server-side source code

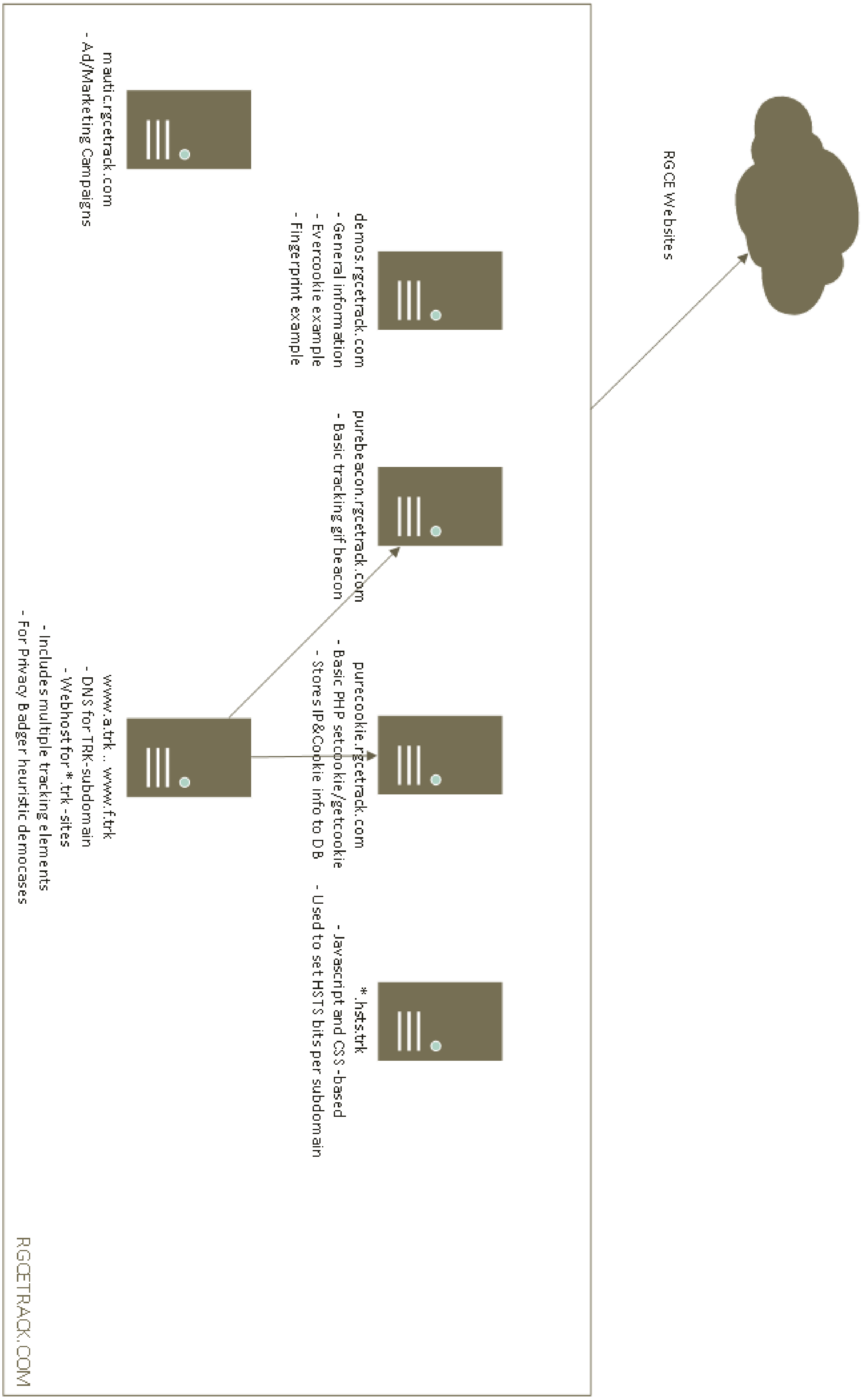
setup.css: (Similar to Appendix 2, Apache should have AddType application/x-httpd-php .css in order to run the PHP code when this "image" is requested. Also, as seen in the code, Content-type headers need to be set to text/css)

```
<?php

// Spoof header, disable caching
header("Content-type: text/css");
header("Cache-control: no-store, no-cache, must-revalidate, max-age=0");
header("Cache-control: post-check=0, pre-check=0", false);
header("Pragma: no-cache");

$bitlength= 8;

// This script generates the ID for a new visitor

include "../sql_settings.php";

// Open SQL connection

$conn = new mysqli($sql_host, $sql_user, $sql_pass, $sql_db);
if ($conn->connect_error) {
            die("Could not connect to database - ". $conn->connect_error);
}

// Check if the request was HTTPS
if (isset($_SERVER["HTTPS"]) || $_SERVER["SERVER_PORT"] == 443 ) {

            // Return a HSTS header - Forces subsequent visits to be HTTPS
            header('Strict-Transport-Security: max-age=31536000');

            // We have already been here, generate a token and request
            $token = uniqid();
            for ($i = 0; $i <$bitlength; $i++)
            {
                        // Print the corresponding bit as include definition
                        echo "@import
url(\"http://$i.hsts.trk/display.css?token=$token\") all;\n";
            }
            //echo "@import \"test.css\" all;\n";

} else {
            // We have not been here, create cookie content

            // Generate new ID
            $useragent = $_SERVER['HTTP_USER_AGENT'];
            // Create new id and save to sql
            $sql = "INSERT INTO user (useragent) VALUES ('$useragent');";
            $result = $conn->query($sql);
            // Set the trackid to include the last id - very crude but good
for demos
            $trackid = $conn->insert_id;

            $conn->close();

            // Split the ID down to bits
            $bits = decbin($trackid);
            $bits = str_pad ($bits, $bitlength, "0", STR_PAD_LEFT);
            for ($i = 0; $i < strlen($bits); $i++)
            {
                        // Print the corresponding bit as include definition
                        $r=$bits[strlen($bits)-$i-1];
                        echo "@import url(\"https://$i.hsts.trk/$r.css\")
all;\n";
            }

            // Import ourselves again
            echo "@import \"https://css.hsts.trk/setup.css\" all;\n";


}

?>
```

display.css:

```
<?php

header("Content-type: text/css");

$bitlength= 8;

// This script generates some css formatting for demonstration
// It also logs token requests to SQL

// Check what our bit sequence is
$domain = explode(".",$_SERVER['HTTP_HOST']);
$bit = $domain[0];
$token = $_GET['token'];

// Fore testing:
//$bit="7";
//$token="asdf";

// Check if we were requested via HTTP or HTTPS

// Check if the request was HTTPS
if (isset($_SERVER["HTTPS"]) || $_SERVER["SERVER_PORT"] == 443 ) {

            echo ".b$bit::after{display:inline;content:'1';}";
            $setbit = 1;
} else {

            echo ".b$bit::after { \n display: inline; \n content: '0'; \n
}\n";
            $setbit=0;

}
echo "\n.token::after { \n display: inline; \n content: '$token'; \n }\n";

// Log the request to hsts_tokens
include "../sql_settings.php";

// Open SQL connection

$conn = new mysqli($sql_host, $sql_user, $sql_pass, $sql_db);
if ($conn->connect_error) {
            die("Could not connect to database - ". $conn->connect_error);
}

$sql = "INSERT INTO hsts_token (token,bit$bit) VALUES ('$token',$setbit) ON
DUPLICATE KEY UPDATE bit$bit=$setbit;";
$result = $conn->query($sql);

// Find out if we have all bits set
$sql = "SELECT * FROM hsts_token WHERE ";
for ($i = 0; $i <$bitlength; $i++)
{
            $sql = $sql . "bit$i IS NOT NULL AND ";
}
$sql = $sql . "token='$token';";
$result = $conn->query($sql);

if ($row = $result->fetch_assoc()) {
            // We have all bits, get bitmask
            $bits=str_pad("",$bitlength," ");
            for ($i = 0; $i <$bitlength; $i++)
            {
                        $bits = substr_replace($bits, $row["bit$i"],
$bitlength-$i, 1);
                        //$bits[$bitlength-$i-1] = $row["bit$i"];
            }
            $trackid = bindec($bits);
            echo "\n.trackid::after { \n display: inline; \n content:
'$trackid'; \n }\n";

            $ip = $_SERVER['REMOTE_ADDR'];
            $referer = $_SERVER['HTTP_REFERER'];

            date_default_timezone_set('UTC');
            $phptime = new DateTime("now");
            $phptime->setTimezone(new DateTimeZone('UTC'));
            $mysqldate = $phptime->format('Y-m-d H:i:s');


            // Then add the pagevisit to seen
            $sql3 = "INSERT INTO seen
(uid,timestamp,ipaddress,referer,method) VALUES
($trackid,'$mysqldate','$ip','$referer','HSTS_TOKEN:$token');";
            $result = $conn->query($sql3);

}

// Set the trackid to include the last id - very crude but good for demos
$trackid = $conn->insert_id;

$conn->close();

?>
```

Appendix 5. Topology for the RGCETrack demonstrational suite


mautic.rgcetrack.com
- Ad/Marketing Campaigns
demos.rgcetrack.com
- General information
- Evercookie example
- Fingerprint example
purebeacon.rgcetrack.com
- Basic tracking gif beacon
purecookie.rgcetrack.com
- Basic PHP setcookie/getcookie
- Stores IP&Cookie info to DB
*.hsts.trk
- Javascript and CSS -based
- Used to set HSTS bits per subdomain
www.a.trk .. www.f.trk
- DNS for TRK-subdomain
-Webhost for *.trk-sites
- Includes multiple tracking elements
- For Privacy Badger heuristic democases
RGCETRACK.COM
RGCE Websites