Towards Understanding Privacy Implications of Adware and Potentially Unwanted Programs

Tobias Urban[1,2(✉)], Dennis Tatang[2], Thorsten Holz[2], and Norbert Pohlmann[1]

[1] Institute for Internet-Security, Gelsenkirchen, Germany
urban@internet-sicherheit.de
[2] Ruhr-University Bochum, Bochum, Germany

Abstract. Web advertisements are the primary financial source for many online services, but also for adversaries. Successful ad campaigns rely on good online profiles of their potential customers. The financial potentials of displaying ads have led to the rise of malicious software that injects or replaces ads on websites, in particular, so-called *adware*. This development leads to continuously further optimized and customized advertising. For these customization's, various tracking methods are used. However, only little work has gone into privacy issues emerging from adware.

In this paper, we investigate the tracking capabilities and related privacy implications of adware and potentially unwanted programs (PUPs). Therefore, we developed a framework that allows us to analyze any network communication of the Firefox browser on the application level to circumvent encryption like TLS. We use this framework to dynamically analyze the communication streams of over 16,000 adware or potentially unwanted programs samples that tamper with the users' browser session. Our results indicate that roughly 37% of the requests issued by the analyzed samples contain private information and are accordingly able to track users. Additionally, we analyze which tracking techniques and services are used by attackers.

Keywords: Adware · Potentially unwanted programs · Privacy

1 Introduction

Nowadays, browsers almost substitute application programs for particular tasks such as e-mail. They allow users to socially interact with others, work on projects, share ideas, or access a broad variety of multimedia content. The amount of private and critical data that browsers mediate continues to increase every year. Naturally, this leads to new risks in the scope of the browsers ecosystem (e.g., banking fraud, user tracking, spam, etc.) since it becomes an attractive target for adversaries. New threats are potentially unwanted programs (PUPs), adware, and malicious browser extensions which tamper with the user's browser session.

J. Lopez et al. (Eds.): ESORICS 2018, LNCS 11098, pp. 449–469, 2018.
https://doi.org/10.1007/978-3-319-99073-6_22

Injecting and replacing ads, as well as redirecting search queries, are popular ways of attackers to make profit.

Gathering, analyzing, and predicting user behavior using private information (e.g., clickstream data) has become a considerable spread phenomenon on the Internet [1]. It is well-known that tracking users and building user profiles is part of the business model of websites and other applications (e.g., mobile applications) [2–8]. However, privacy implications of malware are not well explored yet. In this work, we research privacy leakage by adware and PUPs—to the best of our knowledge, we are the first ones to report such implications on a larger scale.

As these topics are somehow related, technical differences and disparities in the motivation why users are being tracked exist. On the technical side, in contrast to websites and browser extensions, adware and PUPs are not installed with the users' consent or their knowledge, and therefore they do not know that they are being tracked. If it comes to tracking capabilities, websites and extensions are limited to the browser while adware and PUPs have richer access to the users' device and can thereby access more sensitive information (e.g., passwords). Especially malicious programs can track every step of a user by injecting tracking tools into every website a user visits. Thus, these programs can quickly create a comprehensive profile of a particular user which contains highly sensitive data and this is of potentially great value to ad companies.

However, on the motivational side, websites track users to monitor the users' behavior on their sites to improve their services (e.g., suggesting videos the users might like). Extensions might leak private information to third parties, or the extensions server, due to the service they offer (e.g., an extension that checks if a user visits a malicious URL might naturally send the URL to a third party). On the contrary, malware exfiltrates personal data in a purely malicious manner. As scamming money in classical Internet frauds (e.g., credit-card fraud) gets harder and harder, attackers search for new ways to maximize their monetizing efforts (e.g., ransomware or ad injection). Another, not well-explored way, to do that is to exfiltrate private data to build personalized online profiles e.g., the users' clickstreams which can be sold to third parties [9].

In this work, we show the scope of this unnoticed privacy breaches that emerge from adware and PUPs. We found that adware and PUPs heavily focus on the users' clickstream data which can give great detail about the users' personal life. Roughly 27% of all analyzed adware and around 30% of the PUP samples steal the full visited URLs of their victims. Furthermore, we show that data exfiltration is a central component of the malicious activities of adware and PUPs. Our results show that Asian tracking services are popular data sinks for the exfiltrated data. Given the high prevalence of adware and PUPs [10], this data exfiltration is a considerable threat to our modern society.

To sum up, we make the following contributions:

– We introduce a framework that allows us to capture traffic of software that tampers with the browser session on the application level (see Sect. 4) when visiting a predefined set of websites (see Sect. 4.2).

– We provide a detailed analysis of the negative privacy impact emerging from adware and PUPs. Our results show that more than 45% of all analyzed adware and PUPs samples exfiltrate personal data or track users (see Sect. 4.3). To the best of our knowledge, we are the first to report on data leakage and profiling by adware and PUPs on a large scale.
– Finally, we identified (1) the services used to track users, (2) the websites most commonly tracked, (3) and data that is predominantly exfiltrated by adware or PUPs (see Sect. 5.1).

2 Background

In this section, we explain the terms *adware*, *potentially unwanted programs*, and browser extensions. Further, we give a brief overview of the adware ecosystem and describe several tracking mechanisms.

2.1 Adware, Potentially Unwanted Programs, and Browser Extensions

In this work, we analyze two different types of software, namely adware and potentially unwanted programs (PUPs). We further analyze browser extensions to assess our results and to make them more comparable to other related work (e.g., [2,11–13]). In the following, we explain these types of software and discuss how we understand them in the scope of this work:

1. **Adware** is (malicious) software that generates revenue by displaying ads to users (e.g., by injecting or replacing ads on websites). Aside from the ad injection, adware often redirects search requests to advertising websites or collects private data of the users (e.g., clickstream data). Commonly, adware is considered to be malicious if the collection of data or ad-injection happens without adequately notifying the user and if it is installed like other malware (e.g., drive-by-downloads).
2. Potentially unwanted programs (**PUPs**), is a type of software that users might perceive undesirable, as it is installed along with software the user intends to install. The PUPs are bundled with popular benign software and are distributed by so-called pay-per-install services (PPI). PPI services get paid for installations of software (the installer bundle) on target hosts. PUPs could be software with any capability, malicious or benign. However, in the wild, this kind of software often shows similar behavior as adware [10] (e.g., ad-injection or user-tracking).
3. **Browser extensions** are programs that extend the functionality of a web browser (e.g., block advertisements). Extensions have generous access to many functions provided by the browser.

In this work, we examine the negative privacy implications of adware and PUPs and compare these findings to extension downloaded from the Firefox

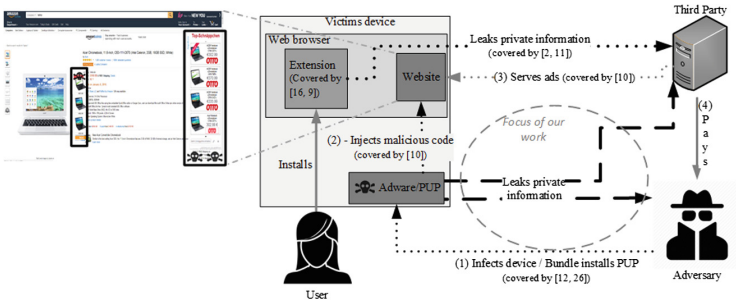

Fig. 1. Overview of the adware ecosystem. The adversary infects the victim's device with malicious software which insert ads into a visited website. After displaying the ads, or a click on the ad by the user, the adversary gets paid typically by a an ad network.

Add-On repository [14]. In the past, adware or PUPs could come in form of an extension but due to policy changes of Firefox one can only install extensions present in their repository. This is probably why none of the analyzed samples successfully installed an extension. We focus on the negative privacy impact of adware and PUPs but also give hints regarding the "ad injection" and "search query redirection" capabilities of the analyzed samples (see Sect. 5).

As just defined, adware and PUPs have similar capabilities, and therefore it is reasonable to analyze both and compare them to each other. In order to make our results more comparable to previous work, we additionally analyzed browser extensions which are well explored regarding their (malicious) behavior. Of course, adware has more access to the operating system and could, therefore, come along with many other malicious capabilities than browser extensions. Therefore, we analyze the outbound network traffic that is not emerging from the browser ("second channel") to examine privacy breaches on that channel, too.

2.2 Adware Ecosystem

The focus of this work lies in the analysis of privacy implications of adware and PUPs. The adware ecosystem is presented in Fig. 1: (1) The user's system is infected with software (i.e., adware, PUPs, or extensions) that tampers with the browser session. (2) The extensions, PUPs, and adware inject their (malicious) objects (e.g., JavaScript code, or images) into the visited website. These objects might be used to load some content from a third party (e.g., ads), or might exfiltrate private information about the user.

Many parts of the ecosystems are already well explored (dotted lines). In this work, we analyze the privacy implications of adware an PUPs for users (dashed lines). To the best of our knowledge, there has been no research analyzing this part systematically on a large scale.

The main monetization technique of adware (as the name hints) is injecting ads into websites and getting paid based on the payment model of the ad-network (e.g., pay-per-view) (3). Nevertheless, authors of adware, PUPs, or malicious extensions might also sell private data they exfiltrate from their victims [15] (4).

2.3 Tracking Mechanisms

Tracking mechanisms can be subdivided into *stateful* and *stateless* tracking methods. Stateful tracking identifies users through a unique identifier chosen by the tracker. On the contrary, stateless tracking tries to determine users through properties of the users' device or browser (e.g., installed fonts or drivers).

Two exemplary *stateful* tracking techniques are explained in the following:

- A *web beacon* (sometimes called *tracking pixel* or *web bug*) is often not larger than 1×1 pixel and usually a transparent graphic image, which is placed on a website for monitoring the user behavior [16]. It is often used with cookies as an additional tracking mechanism. Software that tampers with the user's browser session, like browser extensions, can insert such web beacons on every visited website.
- `Third party cookies` are a popular way to track users across different servers. In contrast to first-party cookies, which are set by the currently visited website, third party cookies are set, e.g., by content loaded from the third party by the visited website. However, third-party cookies are set for the same reason than standard first-party cookies so that a visited website can identify a user later on.

Two examples of *stateless* tracking are browser and canvas fingerprints:

- `Browser fingerprinting` enables website providers to recognize and identify a user's system by unique properties of each browser. Eckersley demonstrates that a combination of browser and device features can almost uniquely identify most users on the web [17]. Web-based browser fingerprinting is, therefore, a conventional technique that has been investigated by several other researchers [17–20]. This technique can further be abused for customization of displayed products, e.g., recently Hupperich et al. showed that the location plays a role in the price offered for hotel bookings [21].
- `Canvas fingerprinting` is possible by abusing the HTML canvas element, that was introduced in HTML5, to draw graphics onto websites. Mowery and Shacham demonstrate that it is feasible to use for user tracking [7].

3 Related Work

Adware and Malicious Add-Ons. Jagpal et al. [22] present WebEval, a system that identifies malicious extensions for the Google Chrome web browser. The authors identify different types of malicious extensions. The two most common types are Facebook session hijackers and ad-injectors (adware). Similar to

our work, they perform a dynamic analysis of each extension and log how it interacts with the browser and operating system. Jagpal et al. do that by performing everyday tasks like querying search engines, visiting social media, and browsing favorite news sites. Aside from their dynamic approach they also conduct a static code analysis to decide if an extension is malicious or not.

Hulk [11] is another framework that is used to identify malicious browser extensions. Hulk employs so-called *HoneyPages* and a technique called "event handler fuzzing". HoneyPages are empty HTML pages. If an extension queries for a tag on a website (e.g., `getElementById ("foo")`) this tag is automatically inserted into the HoneyPage. Thus, the extension assumes the element is present on the website and interacts with it. Using *event handler fuzzing*, `Hulk` pretends to visit all websites on the Alexa Top 1M [23] but just presents a HoneyPage to the extension.

Thomas et al. [12] combine Hulk and WebEval to measure the effect of malicious extensions on the websites google.com, amazon.com, and walmart.com. They report that 5% of the daily unique IP addresses visiting `google.com` are infected with malware that injects ads into websites.

OriginTracer [8] is a tool developed by Arshad et al., which allows tracking the provenance of web content injected into websites by web extensions. They evaluate the usability and performance of the introduced tool and show that such a tool is of great value for users to identify content that was injected into websites by third parties.

Neither Hulk, WebEval nor OriginTracer target privacy implications but focus on identifying malicious browser extensions. We measured and analyzed the negative privacy impact for users that are infected by adware or PUPs.

Analysis About Fingerprinting on the Web. In a large-scale study, Acar et al. examine three advanced web tracking mechanisms (canvas fingerprinting, evercookies, and cookie syncing) [3]. According to their study, 5% of the top 100k websites use canvas fingerprints to identify users.

In 2010, Ashkan et al. conducted a study on the use of Flash cookies [24]. 50% of the websites in their set (Alexa top 100 sites [23]) use this kind of cookie mostly without disclosing this in their privacy policies. Note that since May 2011, all EU countries adopted a directive which says amongst others that websites have to display a "warning" to users if they use cookies [25].

FPDetective, a framework to analyze and detect web-based fingerprints, is introduced by Acar et al. [26]. They used their framework to crawl the most popular websites and analyze if the JavaScript code that is transmitted is used to create fingerprints. In their work, the authors show that fingerprinting is a growing problem and significantly more attractive than previous work suggested.

Englehardt and Narayanan [5] present the most recent study on online tracking. They introduce the open-source measurement tool OpenWPM, which they used to crawl and analyze the top one million websites on the internet. They measure several stateful and stateless tracking techniques and discover some methods that have not been noticed in the wild before (e.g., audio fingerprinting).

The introduced work measures the tracking capabilities and other privacy implications of modern websites. In this work, we analyze the exfiltration of private data and user tracking by malware, i.e., adware and PUPs.

Prevalence of Potentially Unwanted Programs. The prevalence and distribution of PUPs are examined by Kotzias et al. [10]. By analyzing AV telemetry, Kotzias et al. show that around 54% of 3.9 million analyzed hosts have PUPs installed. Furthermore, they found that the top PUP publisher ranks 15 among all software publisher (benign or not). They analyze the PUP-malware relationship and conclude that PUP and malware distribution is independent from another.

The pay-per-install (PPI) ecosystem is analyzed by Thomas et al. [27]. The authors show that PPIs sell access to the users' systems for prices ranging from 0.10$ to 1.50$ per installation. Furthermore, they show that PPI services take a considerable part in distributing PUPs. Based on Google Safe Browsing telemetry, they show that PUPs are downloaded three times more often than classical malware. Both works show the massive prevalence of PUPs but do not investigate the influence this type of software has on the users' privacy.

Privacy Implications of Browser Extensions. The privacy diffusion enabled by browser extensions is examined by Starov and Nikiforakis [2]. They dynamically analyze the privacy leakage of extensions available for the Google Chrome browser. They find that a non-negligible amount (6.3%) of the top 10,000 extensions leak privacy-sensitive data. To counter the leakage, they design BrowsingFog a tool to conceal the user's actual interest on the web. The tool pretends to visit different websites on the internet ("fog") which makes it arguably harder to distinguish between intended and non-intended page visits.

The most recent work in this field of research is written by Weissbacher et al. [13]. The authors present a prototype implementation called Ex-Ray that can identify the privacy-violating behavior of browser extensions. In their work, they use an unsupervised learning approach to identify those extensions. The proposed experimental setup is comparable to our setup but only captures traffic on the network level. Thus, they cannot access and analyze the data, if they are transferred over a TLS secured channel.

The work of Starov and Nikiforakis is to some extent comparable to our work but, due to the nature of their analysis framework, does not cover tracking capabilities of extensions and does not look for exfiltrated metadata (e.g., user-agents or passwords). In [2] the software is analyzed that might need some personal information to successfully run their service (e.g., to identify malicious URLs). In contrast, we focus on malware that exfiltrates data in a purely malicious manner which foreshadows that there is a clear distinction between these two types of software. On a technical level we extend the findings of [2] by (1) identifying all exfiltrated data, (2) showing that there is a significant difference in type and amount of exfiltrated data, (3) identify websites to which visits are

primary tracked, (4) analyzing the tracking behavior of malware, (5) determining the tracking services used by different malware families, and (6) identifying the used tracking techniques.

4 Approach

In this section, we introduce our framework, describe its working principles, inform about our analyzed data set, and give an overview of the investigated samples. Note that in contrast to most related work, due to the application-level monitoring, our system can even inspect HTTPS traffic, can find private data in encoded and deflated content, and allows a stateful analysis.

4.1 Framework

We developed a framework (see Fig. 2) that allows us to (1) perform a *stateful* analysis of each sample, (2) *capture*, if needed *decrypt*, *decode* and *analyze* HTTP(s) communication on application level, and further (3) collect and analyze all network traffic not emerging from the browser.

The general workflow of a single analysis run goes as follows. The analysis slave pulls and installs an adware sample, PUP sample or extension from the server (1). Afterward, the slave visits a predefined set of websites (2a) and logs the resulting communication. To do so, we developed a browser extension that captures all network traffic on the application level. Since we save the traffic on the application level, we can inspect all requests and responses before or after they are encrypted or decrypted, by the TLS layer. After visiting a website, we wait for 30 seconds so it can finish loading and the analyzed software has time to inject content into the site. Additionally, we record all traffic on network level that is originated from aside the browser (2b). We cannot decrypt the traffic apart from the browser. Thus in our analysis, we are limited to the unencrypted traffic. At the end of the analysis run, the plain HTTP(s) traffic and the further communication is sent to the server for review. Before the analysis we—if needed and possible—inflate (e.g., `gzip`) and decode (e.g., `BASE64`) all data (see also Sect. 4.3).

In this work, we perform a *stateful* analysis which means that the used browser has properties that a mock browser or a default state would typically not show (e.g., a browsing history or cookies). If one wants to analyze the tracking capabilities of the software, it is inevitable to perform a *stateful* analysis because resetting the state of the browser during the investigation of a sample might disable some mechanisms that are used for tracking (e.g., cookies). The clean installation state of our slaves—that is recovered after each restart—has a browsing history, several cookies set, passwords in the browser's password vault, and other properties that are usually set when using a browser. Note that most prior work performs a stateless analysis of ad-injectors or browser extensions [11,12,26]. Only OpenWPM performs a stateful analysis [5].

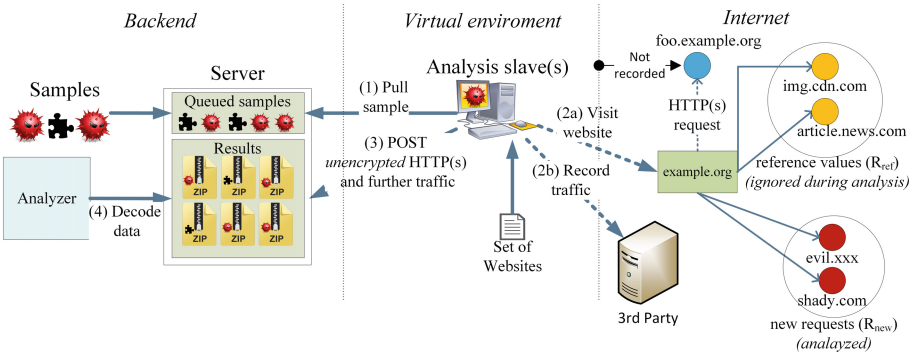

Fig. 2. Overview of our developed framework for the dynamic traffic analysis of adware, PUPs and browser extensions.

To conduct a representative analysis, we need to learn the regular communication of a website to distinguish between requests regularly issued by the site and requests issued by an object injected by the adware, PUP, or extension. We collect the non-malicious regular communication of a website for our analysis by visiting all sites with an analysis slave— but without installed sample or browser extension.

Since websites tend to load dynamic content from various and often changing sources, each slave collects new reference values after analyzing two samples. All collected reverence values are combined to one reference set R_{ref}. In our analysis, we consider requests that target domains (TLD+1) that are not part of R_{ref} for a given site. We call that set R_{new} Example (see also the right side of Fig. 2): Let's assume that R_{ref} for `example.org` contains requests to `cdn.com` and `news.com`. However, if an infected client visits `example.com` the websites issues requests to `evil.xxx`, and `shady.com`. In our study, we only consider requests `evil.xxx`, and `shady.com` because they are not in R_{new}.

4.2 Dataset

We used the global Alexa Top 100 [23] (as of 01/15/2017) as the basis for our set of websites which are visited by the analysis slaves. We restricted our analysis to unique hostnames from this list (e.g., we only analyze `google.com` even if `google.co.uk` is on the list as well) because we assume that the communication would be similar.

After filtering the sets consists of 57 domains. We added five popular e-commerce domains (e.g., `bestbuy.com`) because we expect the adware or PUPs to be more active on e-commerce websites, which turned out to be true for PUPs but not necessarily for adware (see Sect. 5. For each of those domains, we chose two subsites either randomly by visiting the domain and selecting two links, or if possible by selecting the most popular subsites for this site (e.g., products).

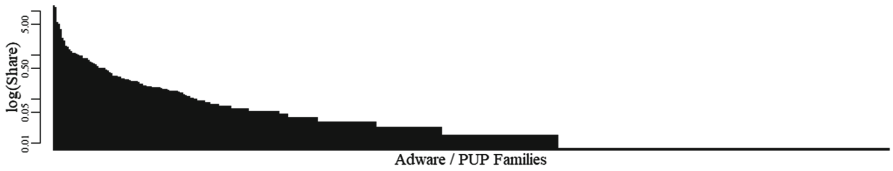

Fig. 3. Distribution, on a logarithmic scale, of the analyzed malware sample families. One adware family (`Dealply`) is dominant in our set while the rest is more or less balanced - which allows us to generalize our results.

A more detailed overview of the set can be found in Appendix A. In total, the analysis of each sample takes around 70 min (including booting, infection, visiting the 128 websites, waiting 30 s, etc.). Previous work either visited a broad set of websites once to conduct their analysis (e.g., [5]), used some mock pages to analyze the injected content (e.g., [11]), or did not disclose how many sites they visit (e.g., [22]).

For our analysis, we used 8,536 distinct adware samples (referred to as S_{AD}) and 8,109 distinct PUP samples (S_{PUP}) (different regarding SHA256 hashes). The samples in $S_{AD} \cup S_{PUP}$ come from 484 different malware families (AV labels). Less than 12% of the samples belong to the most common adware family (`DealPly`), and 5% belong to the most common PUP family (`InstallCore`). The full distribution—on a logarithmic scale—of malware families is displayed in Fig. 3. The distribution of samples across malware families shows that the data set is balanced and allows to generalize our results.

We used samples that were submitted to VirusTotal [28] between 01/01/2017 and 12/20/2017. VirusTotal shut down their API in August and ever since then provides a data set for researchers on Google drive that is updated monthly. The used samples are either identified to be a potentially unwanted program (PUP) or adware by the anti-malware engines used by VirusTotal. We used samples with these labels because we expect that those samples will primarily exfiltrate private data and inject content into websites. To better assess our findings regarding adware and PUPs and to make our work more comparable with previous work, we analyzed the top 5,500 Firefox extensions (S_{ext}) available in the Firefox add-on repository [14]. According to the number of users, we took from the add-on repository, the top 5,500 extension cover 97.2% off all Firefox extension installations. Previous work focused on Chrome extensions, and therefore our analysis also complements these results.

4.3 Analysis

In the following, we focus on analyzing the communication of adware and PUPs. More specifically, we analyze the used tracking services, exfiltrated information, and tracked websites. Additionally, we compare these findings to privacy leakage of the browser extensions we analyzed and with results of previous work.

A website can implement a Content Security Policy (CSP) as a defense mechanism to mitigate certain types of attacks like cross-site scripting or data injection attacks. During our analysis, we found that only 17 subsites use CSPs.

Exfiltrated Personal Information. In this work, we consider information to be private if it holds: (1) data that can be used to identify the client (e.g., IP-addresses), (2) can be used to create a user profile (e.g., visited URLs), or (3) contain sensitive data stored on the computer (e.g., passwords). We consider a website to be a tracker (or tracking service) if it gathers data that can be used to identify users or create profiles about them.

We identified the exfiltrated data by analyzing the transferred cookie, or data sent via the HTTP body. Individual headers can be used to gather personal information about the user (e.g., the user agent or user's preferred language), but these headers are commonly set by default. Hence, we cannot measure if the analyzed sample utilizes these fields. Before analyzing the fields we, if possible, deflate (e.g., `gzip/deflate`) and decode (e.g., `BSAE64`) them. If possible, we repeat this process in case fields are encoded or inflated multiple times, as observed by Starov et al. [2] (e.g., `base64_enc(base64_enc(url_enc(<data>)))`).

After the inflating and decoding, we perform a keyword matching to determine whether a request is used to leak private information. We identified the keywords by manual inspection of several requests issued by the different analyzed samples. We used 13 keyword categories that on the one hand are commonly used to identify or track users (e.g., screen resolution or installed fonts) and on the other hand information that is specific for our analysis setup (e.g., IP addresses or passwords). Some categories are identified by multiple keywords others just by one (e.g., the password is equal for all machines all the time while the user agent varies from sample to sample). We found 15,462 keywords in the analyzed requests. A manual inspection of a sample of the requests we identified a small (less than ten requests) to be false negatives (e.g., a keyword in a seemingly random string - *AR5* ***WIN7SP1*** *UFB2RI3*). A list of the most relevant keywords (based on their occurrence) is given in Table 2. Furthermore, we check if script code that is sent to the client within the response might be used to track users. If possible, we implemented several metrics provided in [5,26] to identify JavaScript that is used to track users.

To summarize, we consider a request to have negative privacy implication if and only if (1) it is part of R_{new}, and (2) it is used for tracking or contains private information.

5 Results

In this section, we provide an overview of the results of our analysis. Throughout this section, if not stated otherwise, we only consider requests used to track users or leak personal data to third parties.

Table 1. Websites that were actively tracked by the analyzed samples (Alexa Ranks as off 11/30/2017).

Adware				PUPs				Extensions			
%-Sam.	Website	Cat.	Rank	%-Sam.	Website	Cat.	Rank	%-Sam.	Website	Cat.	Rank
15.94	tmall.com	shopping	14	17.02	tmall.com	shopping	14	19.74	tmall.com	shopping	14
6.54	msn.com	misc	49	6.65	cnn.com	news	106	10.05	instagram.com	image	17
5.40	cnn.com	news	106	6.07	asos.com	shopping	360	9.40	youtube.com	video	2
5.28	youtube.com	video	2	5.96	ebay.com	shopping	38	7.11	microsoft.com	shopping	50
4.93	asos.com	shopping	360	5.90	target.com	shopping	283	6.13	cnn.com	news	106

In total, we analyzed 16,645 malicious software samples (8,536 adware samples and 8,109 PUPs) and 5,500 Firefox extensions. We analyzed about 850GB (compressed JSON data) of generated adware/PUP traffic. 45% of the adware samples, 40% of the PUP samples, and 45% of the Firefox extensions inject content into a website that issued requests to domains not present in R_{ref}. Our results, if not stated otherwise, only take these samples into account.

We found that the adware and PUP samples issued 21,429 requests to domains not present in our reference dataset, an increase of 10%. 61 of the adware samples changed the home page of the browser, and 221 changed the browser's standard search engine or redirected search queries. In contrast, only 6 PUPs changed the home page, but still, 180 replaced the default search engine. Due to Firefox policies, Firefox extensions cannot change these attributes.

5.1 Privacy Aspects

In this subsection, we present the results of the analysis of the HTTP(s) traffic emerging from the browser. Remember that our framework allows to (1) analyze all traffic in plain text—no matter if HTTPs was used or not—and (2) tries to deflate and decode all data before the analysis (e.g., `HTTP GET` parameters).

Tracked Websites. Table 1 displays the top websites to which visits were actively tracked by the analyzed samples. We consider a website to be tracked if the analyzed sample injects content that can be used for tracking (e.g., a web beacon), or if an observed outgoing request contains any personal information. In our set of websites, each site is tracked by at least 1.5% of the adware and PUP samples. These samples circumvent the CSPs used by websites.

It is notable that the extensions and adware focus on popular websites (e.g., `Youtube` or `Instagram`) from different categories while PUPs predominantly focuses on shopping sites. This indicates that PUPs try to understand what a user plans to buy while adware is gathering information that gives a broader overview of the users habits since they track more general websites as well as shopping sites. Accordingly, this allows providing targeted ads for individual persons, making these kinds of information valuable for ad-companies. Overall, way fewer extensions exfiltrate personal information (31.64%) compared to adware and PUPs (46.41%).

Our results show that user tracking is a significant part of the malicious behavior of adware and PUPs. Almost 40% of the request issued by the adware samples, and 35% of the requests issued by PUPs contain personal information or may be used to track users (e.g., they include the visited URL: *shady.com/?url=google.com%2Fiphone%2B6*). In contrast, only 28% of the requests are used by the extensions for those purposes.

Leaked Personal Information. To measure the privacy impact, we first identify the transferred personal information triggered by the tested samples. We analyze the transferred cookie, and data sent in the HTTP body requests. Furthermore, we inspect if a response contains JavaScript that is used for stateless tracking or if the answer includes a web beacon.

As described in Sect. 4.3, after deflating and decoding, we perform a keyword matching to determine whether a request leaks personal information usable for tracking mechanisms or not. Table 2 shows the results of that matching.

Figure 4 displays the third parties receiving the personal information. Note, if a request contains multiple keywords, we count the request numerous times.

In general, compared to PUPs, extensions and adware focus on meta information (e.g., language, time, IP address, etc.). The visited domain is exfiltrated by all analyzed software types alike (32%) while PUPs and adware predominately exfiltrate the full request URL (domain and GET parameters). However, one can argue that some extensions transfer this information as part of their service (e.g., an extension that checks if the users visit a malicious website will naturally send the current URL to a third party). In contrast, adware or PUPs leak personal data in a malicious manner or because the used ad services requires the current URL. In either way, the user's privacy is undermined unnoticed and without the user's consent. Table 2 shows that PUPs and adware, in contrast to extensions, focuses on the user's clickstream (i.e., browsing history). This is a more significant threat to the user privacy due to the detailed information leaked users' personal life (e.g., habits).

We can *not* identify any privacy-related information in about 6.9% of the requests issued by adware and PUPs (e.g., `cdn.gigya.com/JS/gigya.js?apiKey=3_GL3L[...]`) and 56% of the requests did *not* contain any data we analyzed (e.g., `code.jquery.com/jquery-2.2.4.min.js`).

To the best of our knowledge, there has not been any report on privacy breaches of adware and PUPs. Our measurements show that a significant part, more than ⅓, of the adware's and PUPs communication leaks personal information of users or tracks them. If one takes into account that the majority of the leaked data is the user's browsing history (Domain and URL in Table 2) this kind of leakage is way more severe than the extension leaks.

Starov and Nikiforakis observed that several Chrome extensions, 6.3% of the top 10k, 'unintentionally' leak the HTTP referrer header to third parties (e.g., by embedding objects on every website) [2]. We observed a comparable leakage by 6.55% of the analyzed Firefox extensions and by 6.91% of the analyzed adware.

Table 2. Most commonly leaked personal information

Information	Adware			PUP			Extensions		
	%-S.	Median	Max	%-S.	Median	Max	%-S.	Median	Max
IP address	0.92	3	3	0.69	2	3	0.85	6	30
Operating sys.	5.49	2	5	5.54	2	5	6.21	2	30
User-Agent	5.41	2	2	4.77	2	3	5.35	14	60
Desktop res.	7.35	3	20	6.32	2	7	7.19	2	9
Domain	32.16	2	27	35.12	2	26	32.77	2	126
Full URL	27.18	2	13	29.52	2	10	15.56	2	66
Referrer leak	6.91	0	19	3.31	3	23	6.55	0	20

We did not further investigate this unintentional leakage because the header provides only little utility for the adversary and there are several other ways for her to access this information (e.g., by merely reading the visited URL) and furthermore we cannot measure if the header is utilized. Naturally, the third party receiving the referrer header could use this information. Thus, this kind of leakage still poses a threat to the user's privacy.

Tracking Services. Figure 4 displays the tracking services used by the different malware families. To increase readability, we only listed services used at least seven times by any family and the top 16 malware families individually and combined all other families to *Others*. `Agent`, `Dealply`, the most common adware families in our dataset, and `InstallCore`, the most common PUP family in our dataset, are using a broad variety of tracking services One can see that `TaboTabo` and `MMStat` are overall the most common services used to track users. `taobao.com` is operated by `Zhejiang Taobao Network Ltd.`, while `mmstat.com` is operated by `Alibaba Co., Ltd.` Both two big Chinese players in the Internet landscape. The third most common observed tracker, `GoogleVideo`, is a content delivery network—which is also a known tracker—used to host video or sound files. An overall overview of the most commonly used tracking services and the personal data used by these services is given in Appendix B.

Along with the findings that ad-injection targets users in South Asia, and South East Asia [12] our results indicate that adware and PUPs use services based in Asia. The usage of these services is understandable because access to big American tracking services (e.g., Facebook or Google) is not possible since they are blocked in China [29].

Tracking Techniques. Table 3 presents the tracking techniques utilized by the analyzed samples—only requests are listed that are used for a specific tracking technique. Previous work shows that stateless tracking is becoming more common on popular websites [5]. However, the analyzed adware samples and PUPs do not utilize stateless tracking techniques. This behavior is comprehensible since

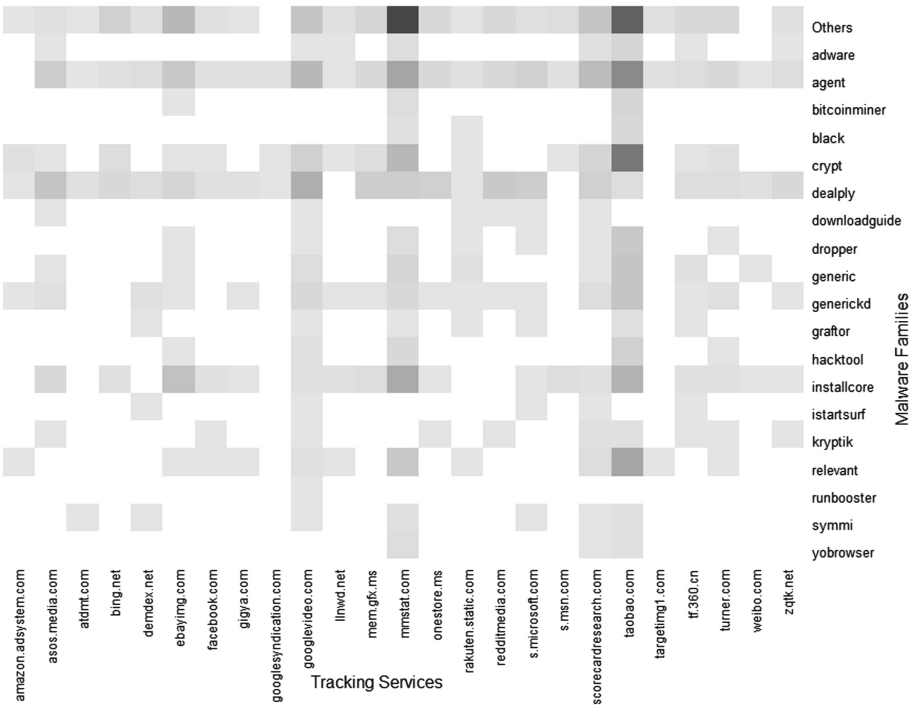

Fig. 4. Top tracking services used by the analyzed adware (A) and PUP (P) families.

the samples can manipulate every website the user visits and therefore can inject a stateful tracking object into each site. Thus, they do not have to rely on more complex and error-prone stateless tracking techniques.

Our analysis shows that web beacons are the most common tracking method among all analyzed samples (adware, PUPs and browser extensions). This result is reasonable since they are easy to implement and are not as easy to block as third-party cookies. It is notable that extensions do not as often use web beacons but utilize 3rd party cookies more commonly.

The results indicate that user tracking is less critical to adware and PUP authors than exfiltrating personal data. But one can argue that exfiltrating the visited URL or domain is also a form of tracking. Requests that contain personal information but do not follow a specific tracking scheme are not considered (e.g., A request contains personal information and loads a picture bigger than a typical web beacon is not counted). The vast majority (around 88%) of requests that impact the users' privacy leak personal information.

Non-browser Emitted Communication. The analysis in this section includes all malware and PUP samples even if they did not insert any object into a website. Similar to the analysis of the traffic emitted by the browser, we used the communication of R_{ref} as reference values for non-malicious communication

Table 3. Tracking techniques used by the analyzed adware and extensions. The vast majority tracks the users in a different way (e.g., by leaking the URL to a third party).

		Cookies	Web beacon	Stateless	Data leakage
Adware	(%-Sam.)	0.03 %	17.36 %	0.02 %	88.55 %
PUPs	(%-Sam.)	0.02 %	16.93 %	1.07 %	87.42 %
Extensions	(%-Sam.)	4.47 %	9.83 %	0.09 %	89.32 %

(e.g., connections issued by the operating system). The analysis in this chapter *excludes* all local traffic and traffic on the browser level.

We used our identified keywords to check if any personal information is sent to any of these IP addresses (malicious or non-malicious). To do that we match the identified keywords against the payload of each unencrypted packet.

Less than 0.5% of the packets contain meta information (e.g., operation system, or used language), and no packet included clickstream data. The small amount of exfiltrated data shows that adware and PUPs do not leak unencrypted personal information on network level which makes this kind of leakage hard to detect. Due to the low amount of identified exfiltrated data, we did not further investigate this communication, but this analysis could be part of future work on this topic.

6 Discussion

In the following, we discuss ethical considerations and limitations of our work.

6.1 Ethical Considerations

Running live malware samples always comes with some ethical issues. On the one hand, one wants to understand how malware works in a realistic environment but on the other hand, running malware might result in harming individuals not involved in the analysis process (e.g., via credit card fraud). Since we run malware that generates revenue by displaying ads and stealing private information we eventually created some income for the malware authors during our analysis. We implemented measures to decrease the potential harm a sample can cause (e.g., by limiting the upload bandwidth to minimize their participation in a possible DDoS attack).

6.2 Limitations

Our developed framework allows the dynamic analysis of software that tampers with the users' browser session. However, it comes, like most dynamic approaches, with some limitations. Using a predefined set of websites leaves the risk that the analyzed software does not get active on the visited websites

(e.g., banking-malware might only get active on specific subsites of a particular banking site). However, previous work has shown that the top-ranked pages trigger a lot of malware samples and extensions [2,11,12,26]. Also, some samples might only inject content into websites only if certain search words appear, as shown in [12]. Since we use a predefined set of websites and therefore predefined keywords, we will not see injections related to other keywords.

Currently, our analysis slaves do not interact with the websites in a way a real user might (e.g., scrolling, or clicking on links). Some malware samples might only trigger if an event occurs, if the user interacts with a website we missed this kind of behaviour.

Since we are using a virtual environment to execute the malware, some samples might recognize that they are being analyzed. We took several measures to hide that the malware is executed on a virtual machine (e.g., changing CPU information and some registry keys). However, a malware sample might still detect that it is being analyzed and show a different behavior.

7 Conclusion

Our results show that not only websites and browser extensions but also—on a massive scale—adware and PUPs negatively impact the user's privacy. We analyzed over 16,000 adware and PUP samples towards their privacy implications to the user. Our results illustrate that these kinds of software excessively leak private data (e.g., IP addresses or clickstream data). More than 37% of all requests issued by malware or PUPs is used for one of these two purposes. Adware and PUPs mainly focus on the user's clickstream which holds sensitive personal information and may give great detail of the user's life ranging from e.g., habits, personal preferences to political views. Thus, adware is a not negligible threat to the user's privacy especially because the leakage happens without consent or knowledge of the user. Regarding the tracking behavior PUPs and adware are quite similar and, since they heavily focus on the users' clickstream, pose a far worse threat to the users' privacy than extensions do.

We could show that while there are—regarding the privacy influence—similarities between extensions and adware/PUPs there are also apparent differences. Adware and PUPs mainly focus on the users' clickstream and can, therefore, create comprehensive profiles of users' which are valuable to different companies (e.g., ad-networks). Furthermore, our results show that adware and PUPs do not adopt state of the art tracking techniques.

Acknowledgment. This work was partially supported by the Ministry of Culture and Science of the German State of North Rhine-Westphalia (MKW grant 005-1703-0021 "MEwM") and partially supported by the German Federal Ministry of Education and Research (BMBF grants 16KIS0395 "secUnity" and 01IS14009B "BD-Sec"). We would like to thank the anonymous reviewers for their valuable feedback.

A Set of Websites

The websites used in our analysis are listed in Table 4. We used the Alexa top 100 as the basis for the set. The set of the websites is described in detail in Sect. 4.2.

The set consists of ten search engines, 20 social media sites, 11 online-shops, 5 domains hosting adult content, and 16 domains that do not fit in any of these categories (e.g., `github.com` or `cnn.com`). 34 of the domains are hosted in the United States of America, 14 are hosted in the People's Republic of China, four in the Russian Federation, three in the Kingdom of the Netherlands, two in the Republic of Ireland, and five sites are hosted in different countries in Asia (ROK, SVR, JPN, HKG, and TWN).

Table 4. Set of websites used in our analysis.

search.rakuten.co.jp	instagram.com	coccocqc.com
movie.youku.com	search.naver.com	forums.craigslist.org
www.baidu.com	everysinglewordspoken.tumblr.com	www.ebay.com
health.china.com.cn	foodwishes.blogspot.com	coccoc.com
www.flipkart.com	edition.cnn.com	news.xinhuanet.com
www.zalando.de	www.google.com	hyperboleandahalf.blogspot.com
channel.pixnet.net	www.youtube.com	history.gmw.cn
vk.com	www.bing.com	sd.360.cn
marketplace.asos.com	stock.sohu.com	2kindsofpeople.tumblr.com
imgur.com	github.com	www.xvideos.com
zy.youku.com	xhamster.com	www.pixnet.net
military.china.com.cn	news.gmw.cn	www.alibaba.com
finance.qq.com	en.bongacams.com	world.taobao.com
mall.360.com	stackoverflow.com	www.microsoftstore.com
www.reddit.com	www.asos.com	bbs.tianya.cn
www.twitch.tv	www.so.com	www.apple.com
world.tmall.com	en.wikipedia.org	news.mail.ru
www.quora.com	www.aliexpress.com	news.youth.cn
ok.ru	news.naver.com	www.xinhuanet.com
www.groupon.com	www.sogou.com	auto.mail.ru
www.pornhub.com	www.facebook.com	twitter.com
yandex.ru	cbachina.sports.sohu.com	www.msn.com
www.linkedin.com	www.amazon.com	de.pinterest.com
newyork.craigslist.org	intl.target.com	www.imdb.com
ent.qq.com	www.hao123.com	www.microsoft.com
www.walmart.com	v.youth.cn	

B Tracking Services

Table 5 displays the most common services to which privacy-related information are leaked or which provide tracking tools (e.g., web beacons). Only one service

gathers additional information about the client's system aside from the domain. All, but one, tracking services are operated by "big players" based in China. The analyzed extensions tend to use tracking services operated by American companies (e.g., Google or Facebook). Our results show that the services used by Firefox extensions are comparable to Google Chrome extensions [2].

In total only 151 different trackers were used. 60 trackers are used by just three or fewer samples. Adware and PUP authors tend to rely on existing infrastructure rather than setting up their own. Among the observed tracking services, there is no indication for any preferred service. The top 20 services are used on average by 7.48% (±0.78%) of the adware and PUP samples. This result indicates that the used services do not differentiate among each other regarding the utility of the adware or PUP.

Table 5. Tracking services used by the analyzed adware, leaked information, and the most common communication path

Service	%-S.	Information	Company
taobao.com	10.04	URL, time, language, os.	Taobao Network
mmstat.com	8.47	URL, time, language, os, browser, screen res.	Alibaba
sogoucdn.com	8.36	domain	Sogou Info. Service
ebaystatic.com	8.28	domain	eBay
ykimg.com	8.23	domain	Nexperian Holding

References

1. Bucklin, R.E., Sismeiro, C.: A model of web site browsing behavior estimated on clickstream data. J. Mark. Res. **40**(3), 249–267 (2003)
2. Starov, O., Nikiforakis, N.: Extended tracking powers: measuring the privacy diffusion enabled by browser extensions. In: Proceedings of the 26th International Conference on World Wide Web, WWW 2017, pp. 1481–1490. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva (2017)
3. Acar, G., Eubank, C., Englehardt, S., Juarez, M., Narayanan, A., Diaz, C.: The web never forgets: persistent tracking mechanisms in the wild. In: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, CCS 2014, pp. 674–689. ACM, New York (2014)
4. Boda, K., Földes, Á.M., Gulyás, G.G., Imre, S.: User tracking on the web via cross-browser fingerprinting. In: Laud, P. (ed.) NordSec 2011. LNCS, vol. 7161, pp. 31–46. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29615-4_4
5. Englehardt, S., Narayanan, A.: Online tracking: a 1-million-site measurement and analysis. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS 2016, pp. 1388–1401. ACM, New York (2016)

6. Olejnik, Ł., Acar, G., Castelluccia, C., Diaz, C.: The leaking battery. In: Garcia-Alfaro, J., Navarro-Arribas, G., Aldini, A., Martinelli, F., Suri, N. (eds.) DPM/QASA -2015. LNCS, vol. 9481, pp. 254–263. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-29883-2_18
7. Mowery, K., Shacham, H.: Pixel perfect: fingerprinting canvas in HTML5. In: Fredriksonn, M. (ed.) Proceedings of the Web 2.0 Security and Privacy Workshop (W2SP), pp. 1–12. IEEE Computer Society, New York, May 2012
8. Arshad, S., Kharraz, A., Robertson, W.: Identifying extension-based ad injection via fine-grained web content provenance. In: Monrose, F., Dacier, M., Blanc, G., Garcia-Alfaro, J. (eds.) RAID 2016. LNCS, vol. 9854, pp. 415–436. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45719-2_19
9. LLC, WS: WOT API—WOT (Web of Trust) (2017). https://www.mywot.com/en/api. Accessed 31 Oct 2017
10. Kotzias, P., Bilge, L., Caballero, J.: Measuring PUP prevalence and PUP distribution through pay-per-install services. In: 25th USENIX Security Symposium (USENIX Security 16), pp. 739–756. USENIX Association, Austin (2016)
11. Kapravelos, A., Grier, C., Chachra, N., Kruegel, C., Vigna, G., Paxson, V.: Hulk: eliciting malicious behavior in browser extensions. In: Proceedings of the 23rd USENIX Conference on Security Symposium, SEC 2014, pp. 641–654. USENIX Association, Berkeley (2014)
12. Thomas, K., et al.: Ad injection at scale: assessing deceptive advertisement modifications. In: Proceedings of the 2015 IEEE Symposium on Security and Privacy, SP 2015, pp. 151–167. IEEE Computer Society, Washington (2015)
13. Weissbacher, M., Mariconti, E., Suarez-Tangil, G., Stringhini, G., Robertson, W., Kirda, E.: Ex-Ray: detection of history-leaking browser extensions. In: Proceedings of the 33rd Annual Computer Security Applications Conference, pp. 1–13. ACM, New York (2017)
14. Mozilla Foundation: Add-ons for Firefox (2017). https://addons.mozilla.org/. Accessed 05 July 2017
15. Bonderud, D.: WOT privacy breach: trust tanks as browser add-on caught selling user data (2017). https://securityintelligence.com/news/wot-privacy-breach-trust-tanks-as-browser-add-on-caught-selling-user-data. Accessed 31 Oct 2017
16. Smith, R.M.: The web bug faq. Nov **11**, 4 (1999)
17. Eckersley, P.: How unique is your web browser? In: Atallah, M.J., Hopper, N.J. (eds.) PETS 2010. LNCS, vol. 6205, pp. 1–18. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14527-8_1
18. Hupperich, T., Maiorca, D., Kührer, M., Holz, T., Giacinto, G.: On the robustness of mobile device fingerprinting: can mobile users escape modern web-tracking mechanisms? In: Proceedings of the 31st Annual Computer Security Applications Conference, ACSAC 2015, pp. 191–200. ACM, New York (2015)
19. Kurtz, A., Gascon, H., Becker, T., Rieck, K., Freiling, F.C.: Fingerprinting mobile devices using personalized configurations. Proc. Priv. Enhanc. Technol. (PoPETs) **2016**(1), 4–19 (2016)
20. Nikiforakis, N., Kapravelos, A., Joosen, W., Kruegel, C., Piessens, F., Vigna, G.: Cookieless monster: exploring the ecosystem of web-based device fingerprinting. In: Proceedings of the 2013 IEEE Symposium on Security and Privacy, SP 2013, pp. 541–555. IEEE Computer Society, Washington (2013)
21. Hupperich, T., Tatang, D., Wilkop, N., Holz, T.: An empirical study on online price differentiation. In: Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy, CODASPY 2018, pp. 76–83. ACM, New York (2018)

22. Jagpal, N., et al.: Trends and lessons from three years fighting malicious extensions. In: Proceedings of the 24th USENIX Conference on Security Symposium, SEC 2015, pp. 579–593. USENIX Association, Berkeley (2015)
23. Alexa Internet: Top 500 global sites (2017). http://www.alexa.com/topsites
24. Soltani, A., Canty, S., Mayo, Q., Thomas, L., Hoofnagle, C.J.: Flash cookies and privacy. In: AAAI Spring Symposium: Intelligent Information Privacy Management, pp. 1–6. Association for the Advancement of Artificial Intelligence, Palo Alto (2010)
25. European Parliament: The Council: Directive 2009/136/ec (2009)
26. Acar, G., et al.: FPDetective: dusting the web for fingerprinters. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security, CCS 2013, pp. 1129–1140. ACM, New York (2013)
27. Thomas, K., et al.: Investigating commercial pay-per-install and the distribution of unwanted software. In: 25th USENIX Security Symposium (USENIX Security 16), pp. 721–739. USENIX Association, Austin (2016)
28. VirusTotal: Free online virus, malware and url scanner (2017). https://virustotal.com/. Accessed 24 July 2017
29. GreatFire: Blocked sites in China - bringing transparency to the great firewall of China (2017). https://en.greatfire.org/search/blocked