Detecting Web Tracking at the Network Layer

Maximilian Wittig(✉) and Doğan Kesdoğan

University of Regensburg, Regensburg, Germany
{maximilian.wittig,dogan.kesdogan}@ur.de

Abstract. Third-party tracking allows companies to identify users and track their online activity across different websites or digital services. This paper presents a first experimental study to detect advertisements and tracker by inspecting fully encrypted network transactions at the TCP/IP network level associated with a website. The first results are encouraging and motivate to extend this first proof-of-concept study even further in the future. A classical application area in the future would be the use in areas where communication can only be accessed on encrypted TCP/IP level (keyword secure IoT environments) or the presented approach is used simply to enable a classical extension of the portfolio for tracker detection.

Keywords: Online tracking · Network Traffic Classification · Privacy

1 Introduction

Advertising is essential to a free market economy: it enables commerce by providing consumers with product and service information and encouraging competition. However, online marketing has emerged as a new independent business model, described in the literature as "surveillance capitalism" [32]. To protect themselves from advertising and online tracking, users rely on privacy-enhancing blocking tools, such as Adblock Plus or uBlock Origin. Many of them are based on manually created blacklists, which leads to scalability issues. Hence, the research community has applied machine learning to automate the creation of blacklists. To the best of our knowledge, these approaches solely leverage features at the application layer of the network stack [5,13,16,24]. Although the models perform well with an accuracy up to 98.1%, they assume access to the application layer. Consequently, related work proposes client-side tools that cannot be easily extended to the network-level as the trend shifts towards encrypted data transmission.

This raises the question whether the purpose of a communication (here advertising and tracking) can be inferred despite encryption. Regarding this question, there are many traffic classification problems that gain information by applying pattern recognition to encrypted traffic traces. For instance, in [3,12,18] the encrypted traffic is categorized into several application types, ranging from

N. Meyer and A. Grocholewska-Czuryło (Eds.): SEC 2023, IFIP AICT 679, pp. 131–148, 2024.
https://doi.org/10.1007/978-3-031-56326-3_10

chat application to streaming service. Another prominent example is website fingerprinting, which aims to determine the website a user has accessed via an anonymized channel [14,21,26,27]. These attacks have proven to be effective on HTTPS because traffic features, such as size, timing, and order of network packets, are unique to each website. To the best of our knowledge, we are the first to adapt traffic analysis to tracking detection. This use case entails additional challenges compared to the aforementioned traffic classification problems, since we intend to classify individual elements within a website in contrast to its cumulative traffic flow. Furthermore, modern websites contain various types of potentially harmful resources (images, JavaScript, etc.) which show diverse behavior patterns. Considering that the information about the resource type is not available in encrypted traffic, we face a binary classification problem involving heterogeneous target classes (tracking or non-tracking). Notably, our approach does not break encryption but exploits that privacy-invasive resources, or even communications with the serving host, follow an observable tracking protocol which is distinct from benign traffic traces. In fact, our model learns some meta-information to match observed patterns with known patterns. Traffic classification applied to the advertising and tracking problem potentially opens up network-wide detection and blocking of unwanted resources, which is becoming increasingly relevant as Internet of Things proliferate [20,30]. As a starting point, we seek to answer whether tracking resources are detectable in an encrypted traffic flow, and if so, what features best describe them.

In summary, our contributions are as follows:

– For our empirical study, we create a dataset of the top 1K most popular websites according to the Majestic list. Our dataset covers the entire network-stack from L1 to L7 and maximizes the number of trackers by automatically accepting the consent management platform.
– We train a classifier based on fully encrypted network transactions to identify tracking resources with an accuracy and F_1 score of about 90% and 88%, respectively. Given that the tracking portion of most domains is highly homogeneous, features related to the whole conversation exhibit high discriminatory power.
– We show that traffic features are particularly useful for detecting third-party tracker, as well as unwanted resource types such as documents (HTML), images, and scripts.

The remainder of the paper is structured as follows: Sect. 2 provides background information on cross-site online tracking and corresponding blocking measures. Section 3 presents our experimental design and Sect. 4 showcases our dataset. We show our results on feature importance and classifier performance in Sect. 5. Section 6 concludes our study.

2 Background and Related Work

2.1 Cross-Site Online Tracking

The beauty of the Web lies in the hyperlinking of objects, which allows third-party elements to be easily included into first-party websites. However, the analysis in [2] shows the picture of a hidden market whose actors collect personal data without the user's influence by exploiting third-party content. Today's online marketing is mainly driven by advertisers (demand for product placements), publishers (supply of advertising space) and advertising networks (intermediaries).

When a website is visited, the available advertising space is auctioned to the highest bidder by the ad network [31]. For the financial evaluation of an auction, bidders are provided with a detailed user profile containing information on the geography, demographics and preferences of the respective website visitor. This is achieved by using various tracking techniques that uniquely identify the user and expose information, such as the browsing history [7,19,23]. In collaboration with several publishers, third-party trackers scale across a variety of sites to create a comprehensive browsing profile. As more browser vendors discontinue support for third-party cookies, recent studies show a vast majority of websites are using first-party tracking techniques (e.g., first-party cookies or CNAME cloaking) and sharing them with other parties to circumvent blocking [4,8,9]. They conclude that tracking prevention should be extended to first parties as well.

2.2 Existing Blocking Techniques

Broadly speaking, several strategies exist to mitigate online tracking. One strategy might be to spoof web tracker by providing incorrect information. A prominent example is Brave's randomization to some fingerprinting endpoints [6]. However, blocking unwanted content according to known signatures is much more common. In contrast to spoofing, blocking requires the detection of trackers. This is done either by manually created blacklists or through machine learning to automate the signature creation process.

Blacklists. Popular browser extensions like uBlock Origin rely on blacklists. These lists are manually curated based on user feedback and suffer from scalability and robustness issues. First, blacklists struggle to keep up with the ever expanding advertising and tracking ecosystem. For example, the adoption of anti-adblocker rules took around 90 days [15]. To make matters worse, there are several evasion strategies for advertisers to avoid blacklists, such as changing domains or using the CNAME cloaking technique [10,29].

Machine Learning. To address both, scalability and robustness, researchers have applied machine learning (ML) for automated ad and tracker blocking. The first generation of ML approaches detects trackers based on a single dimension of the application layer. These approaches featurizing the content of either URLs, HTTP headers, or request and response payloads [5,13,24]. Since they mimic blacklists, they inherit their shortcomings (i.e. the presence of a specific keyword is susceptible to trivial evasion). Therefore, the second generation involves a number of graph-based approaches that capture the interactions among HTML elements, JavaScript, and HTTP requests [16,25]. They leverage rich cross-layer features and thus claim to be robust to evasion attempts. However, the aforementioned approaches require access to the application layer, which is usually encrypted. Thus, they cannot efficiently prevent tracking at the network-level, but only on a per-host basis. In contrast, we leverage traffic aspects of TCP/IP layer, which are hardly affected by encryption because the traffic shape is always available. Lastly, we argue that traffic flow statistics are robust to evasions. One could obfuscate such information by sending redundant traffic or delaying packets to manipulate the time series. However, this would have a negative impact on bandwidth, latency, and quality of service.

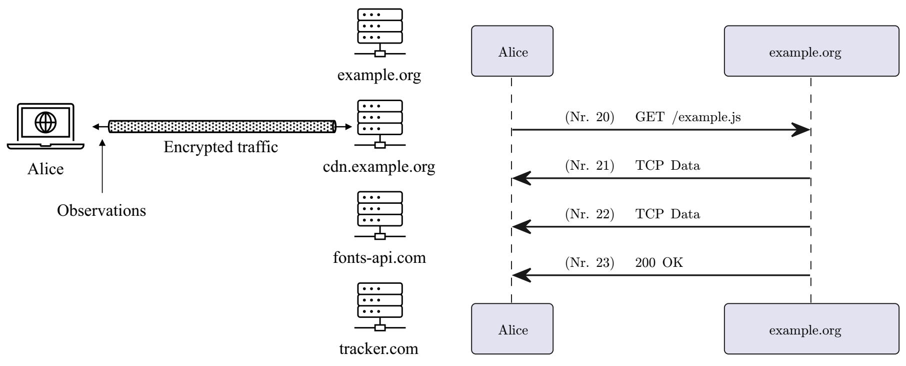
3 Experimental Design

This section provides an overview of our study design. First, we present our assumed attack scenario. Second, we report on our methodology for creating and preprocessing the dataset, as well as parameter selection and training of the classifiers and their evaluation. Last, we present our feature engineering process.

3.1 Scenario

Consider the following scenario (cf. Fig. 1). Alice wants to visit a website (here `example.org`). The initial response from the web server contains the web page, which links to new resources, leading to subsequent requests. These requests typically involve several hosts (third parties). The HTTP packets are then transmitted over Ethernet and are split into multiple TCP packets due to the maximum transmission unit (in our example from packet no. 20 to 23). In addition, all communication between Alice and the web servers is encrypted. However, for labeling purposes, we assume the network traffic in plain. To achieve this, we observe and experiment locally on Alice's host. This gives us also the opportunity to study the upper bound of tracking defenses based on network features with perfect information. One information that might be useful is the relative arrival time of packets since the site was accessed, or knowledge about individual resources. In our model, tracking is defined as a host providing tracking resources to Alice. While some web services are used exclusively for tracking purposes, others fall into a gray area because portions of the communication provide benefits. Therefore, a fine distinction of tracking is necessary. Since current tracking techniques follow fixed protocols, these may show different traffic characteristics than regular web services and thus differ in their communication patterns, even if encrypted. Furthermore, the question arises how well suited these structures are to determine the semantics of the data flow, in this case tracking. Towards answering these questions, we present a machine learning approach for the automated identification of privacy-intrusive services. We classify resources into two classes: (i) tracker and (ii) non-tracker (other). Here, the challenge is to assess the intent of a service with limited knowledge.

In our attack scenario, we run an ex-post traffic analysis for each website. Meaning, we expect Alice to visit each web page sequentially, observe all traffic,



(a) Traffic flows (b) Web resource

Fig. 1. Example scenario for the website call example.org.

and eventually estimate trackers independently of other pages. Our methodology prevents traffic features tailored to a host from being biased. For instance, extensive communication might result from a chatty host or because the host is present at many first parties. In a practical environment, a similar outcome is achieved by using a short time window. It is worth noting that isolating the network traffic of a website preserves the partyness of a resource, which is necessary for labeling the dataset. Furthermore, we discard signaling packets (e.g. ACK packets) since they provide no useful information for our classification task. A similar optimization is done in [21] with the usage of a minimum byte size filter.

3.2 Methodology

As depicted in Fig. 2, our study consists of three components: Web Crawler, Preprocessing and Evaluation. For our empirical study[1], we first build an appropriate dataset of network traces. For this purpose, we developed a tool based on Python-Selenium to automatically crawl websites and capture the entire network stack using Tcpdump. The crawler is containerized in Docker. This allows parallel browsing of websites while keeping the network traffic of each website isolated.

Web Crawler. The homepage of the 1000 most popular websites from the Majestic Million list is automatically visited and the web traffic to the destination port 80 and 443 is filtered. The Majestic 1K list ranks the top websites based on the number of backlinks and can be freely downloaded[2]. We use Google Chrome in headless mode to disable telemetry data. To evade bot detection, we obfuscate the user agent and set the language to en-Us. Furthermore, the homepage is

[1] The source code of our study is available at: https://github.com/wim50594/network-traffic-tracker-observer.

[2] https://majestic.com/reports/majestic-million.

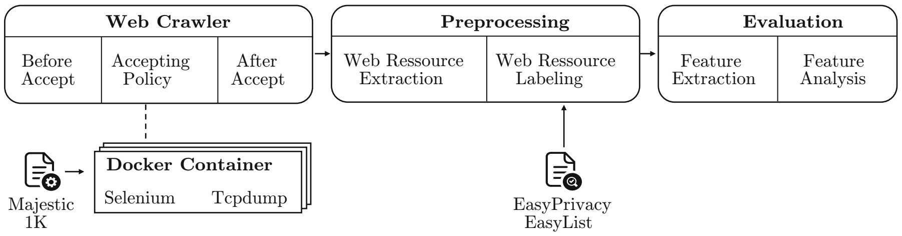


Fig. 2. Abstract overview of our experiment.

scrolled vertically to simulate organic user interaction and trigger additional requests. The environment variable `SSLKEYLOGFILE` is set to allow TLS data to be decrypted during preprocessing. The sslkeylog contains keying material of the TLS handshake. Each website is accessed up to three times for 20 s, with varying degree of cookie policy acceptance: before accept, accepting policy and after accept. Cookie consent is important because more and possibly different trackers may be loaded after acceptance. We only visit a website a third time if it offers a cookie policy and is discovered by us. To identify a cookie policy, we scan the Document Object Model for typical phrases to accept all cookies. We adopt the word list from preliminary work [17]. For all websites, we conduct a stateless crawl, meaning the cache is cleared after each visit.

Preprocessing. Since most requests are protected by HTTPS, the pcap files are decrypted using the sslkeylog. After that, we extract web resources from the packet flow via the tshark command line tool. During this process, an HTTP request and response, as well as their associated TCP segments, are reassembled into a resource (cf. Fig. 1b). Subsequently, each resource is labeled based on the blacklists EasyList (advertisement) and EasyPrivacy (tracking). Both lists[3] were originally developed for the browser extension Adblock and are used in numerous studies [11,13]. Although our main goal is to study tracker detection, we include EasyList in the ground truth because digital advertising is fundamentally linked to tracking and cannot be considered separately. In our case, the well-known advertising network `doubleclick.net` is only exposed by EasyList. To apply the filter rules to the web resources, we utilize Brave's adblock engine[4]. Last, we determine the party level of a resource based on eTLD+1 as well as the content type from the HTTP header.

Evaluation. First, we extract several network features with respect to IP address, size, direction, and timing. Next, we analyze the feature importance by plotting their distribution and measuring the classification performance of a decision tree and logistic regression. Both models were chosen because of their

[3] https://easylist.to/.
[4] https://github.com/brave/adblock-rust.

interpretability of the classification decision. The task of binary classification is to assign a web resource to one of the two categories (i) tracker or (ii) non-tracker. Both classifiers are trained in a supervised learning environment (ground truth is given by blacklists). For this, we use the Python library scikit-learn with default hyperparameters, except for a regularization term to prevent overfitting. These are configured by a grid search. The maximum depth of the decision tree is limited to $\lfloor \frac{\text{number of features}}{4} \rfloor$ with minimum samples at a leaf node set to 5, and L1 penalty term is set to $C = 10^{-4}$ for logistic regression. The performance is evaluated using 5-fold cross-validation and repeated 5 times unless otherwise stated. The split into training and test data is based on website visits, so communication to hosts remains compact. As the features are different in scale, they must be standardized before passed to the logistic regression.

3.3 Network Traffic Features

Based on observations during preliminary work regarding the prevalence and operation of trackers [1,8,23], we know that tracking services seldom deliver large responses (e.g., tracking pixel). At the same time, large amounts of data are supposed to flow to privacy-intrusive services. Hence, we expect the data flow to tracking parties to be characterized by small responses with low volume compared to large client requests. With these observations in mind, we develop statistical features for binary classification of web resources.

Table 1. Network traffic features for this study. The resource level refers to the associated packets of a resource. The conversation level comprises the entire packet flow to a host during a page visit.

A: Resource level (packets associated with a resource)	
packet size	Packet sizes of a resource
in/out packet size	Directional packet sizes of a resource (from client's perspective)
relative time	Relative arrival time of the packets since website call
delta resource time	Delta time to the previous resource of a host
B: Conversation level (separated by IP address)	
packet size	Packet sizes of a conversation
in/out packet size	Directional packet sizes of a conversation (from client's perspective)
relative time	Relative arrival time of the packets since website call
prevalence(ip)	No. of observed IP address on first-party websites

The network features explored in our experiment are summarized in Table 1. We use two different granularity levels of features, namely the conversation level and the resource level. The former describes the communication behavior to a host during a page visit (e.g., in Fig. 1 the traffic to the IP host of `fonts-api.com`). At this level, we are only considering the packet flow without any resource information. However, since web services offer both tracking and non-tracking content (especially in the case of a first-party), we further explore the characteristics of individual resources. To build our features, we aggregate the network packets and compute the main descriptive statistics count, sum, min, max, mean, rsd, and span. The latter takes the difference between max and min. The relative standard deviation is standardized by std/mean (coefficient of variation). At both levels of granularity, we consider TCP packet size, TCP packet size of incoming and outgoing traffic, and the relative arrival time of packets since the site was visited. The split into in and out packets encodes directional features as shown to be important in previous work [26,27]. Since third-party trackers require a high penetration rate to create a comprehensive browsing profile, we expect them to be widely spread on first-party websites. Therefore, we calculate the frequencies of IP addresses on first-party websites within the training data. Unknown IP addresses within the test data are encoded as −1. Last, we measure the time intervals between resources (delta resource time).

Since we introduced several variants of features measuring similar properties, we expect that an automatic classifier will only need a subset of the proposed features. However, we examine all features to understand which specific features are most relevant for classifying tracking web services.

4 Dataset

We conducted our experiment on November 9, 2022, for which we obtained the most current state of the Majestic list and the two blacklists EasyList and EasyPrivacy. We ran our crawl at a German university. Accordingly, we were often redirected to a German version of the website due to IP geolocating.

Table 2. Characteristics of the Majestic-Million 1K dataset

Characteristic	
Websites successfully crawled	933
Cookie-acceptance rate	0.46
Count unique domains	first-party=827/third-party=2161
Average parties per domain	parties=25.52/tracking-parties=22.3
Count resources	first-party=83,993/third-party=182,001
Count tracking resources	is-tracker=105,586/not is-tracker=160,408
Count packets	in=3,258,443/out=320,468

The characteristics of the dataset are summarized in Table 2. In total, 933 websites were successfully visited, approximately 266 thousand resources were requested, and over 3.5 million packets were exchanged. Not all websites could be accessed because the Majestic list contains entries whose domain could not be resolved, or the connection was refused. Most websites (70%) fall under the categories of Computers & Technology, Education, News or Business. The number of first-parties is lower than successfully crawled websites, because some have multiple domains from which they are redirected to the main site. For example, `business.site` becomes `www.google.com`, which is already ranked 1. Likewise, typical properties of web traffic are observed. As expected, we find a significant preponderance of third-parties per domain on average, with 22.3 hosts sending at least one tracking-related resource. Consequently, only one third of the resources originates from the first-party. Furthermore, an imbalance exists between incoming packets to the client and outgoing packets to the web server by a factor of 10. On 46% of the websites, we accepted all cookies. Unfortunately, the ground truth of consent banner is unknown, but in a similar study a detection rate of 63.2% for European websites was reported [17]. Since our experiment crawls the top websites worldwide, with some providing no cookie policy (e.g., `apple.com` or `wikipedia.org`), the result looks reasonable. In 10% of the cases, a cookie banner could not be found within the time window of 20 s because the web pages consisted of extensive HTML elements. The five most common consent texts are: "accept all cookies", "accept all", "accept", "i accept", and "alle akzeptieren".

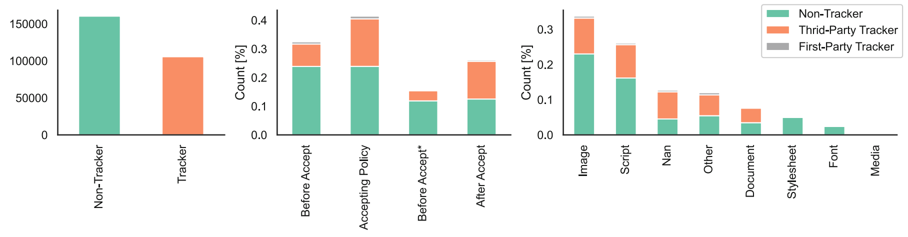


Fig. 3. Tracker distribution in Majestic 1K dataset.

Figure 3 presents a weak imbalance between our target classes with a distribution of about 60/40% harmless resources. However, only 6.7% of first-party resources are tracking-related, as most trackers are included by third parties (94.6%). When the cookie policy is accepted, we measure a remarkable increase of tracking resources. This indicates that the user's consent is only partially taken into consideration, since trackers are already loaded even without consent[5]. The number of resources after accepting is lower, because we visit web pages a third time only when a cookie policy is detected. For a better comparison, Before Accept* denotes a subset of websites whose cookie banners we have identified. The comparison with the samples of after accept shows a significant jump in resources as well as third-party trackers, while first-party trackers slightly increase. Most requested elements are images and scripts (59.9%). If no content type is provided in the HTTP header, it is labeled as "none". The resources marked as "other" include json files in most cases (65.3%).

[5] The interested reader can find an in-depth analysis in [22].

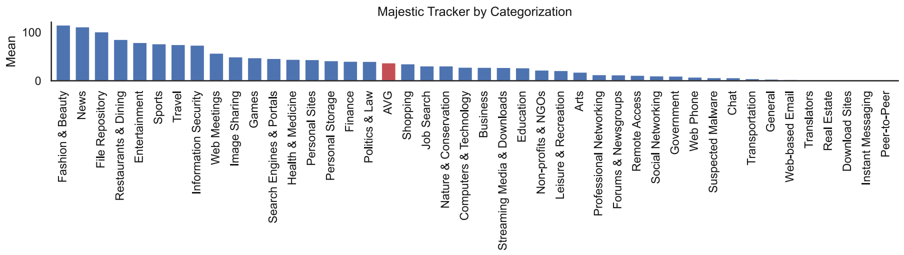


Fig. 4. Majestic 1K top tracked websites categorization by Cyren URL Lookup API[9]. Some were assigned to multiple categories and therefore counted repeatedly.

The amount of tracking varies considerably among different categories of websites. In Fig. 4 we show the average number of tracking resources across all three page visits for each web category. Websites at the lower end of the spectrum are mostly service providers that require authentication, as well as government and non-profit organizations. On average the web pages of the last 20 categories consist of 130.42 non-tracking resources vs. 171.93 overall. Meaning, they are simpler in design and require fewer resources. In contrast, monetization plays a major role for websites at the top of the spectrum, as they primarily offer editorial content with no external funding.

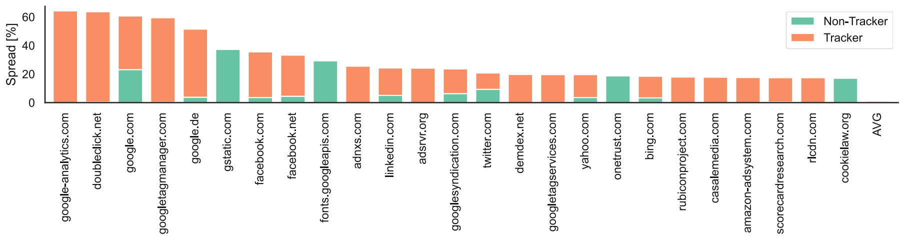


Fig. 5. Top 25 most widespread third parties.

The tracking ecosystem involves only a handful of third parties. Figure 5 shows the percentage of third parties on first-party sites, most of which are for tracking purposes. A regular user is exposed to at least one of the top 25 third parties with 84.89% chance. On average, third parties are only present on about 1% of the pages. The dominance of a few providers becomes even greater when you consider that many of the domains are owned by the same organization.

5 Analysis of Network Traffic

In this section, we analyze characteristic patterns in network flow to tracking or non-tracking services. To do this, we visualize the distribution of network features and then evaluate them using statistical methods. Lastly, we evaluate the classification power and how its performance depend on the website context and resource type.

5.1 Feature Distribution

Since a host may exchange a mix of tracking and non-tracking resources, we measure the purity of the communication channels by computing its service entropy. For our two classes, the entropy varies between 0 and 1. The overall entropy is 0.028 on average and 0.023 for third parties, indicating that communication is mostly pure. This can be explained by the fact that blacklists often only block at a coarse domain level (in our experiment 51.5% of the rules). However, for first-party communication, the mean entropy slightly increases to 0.072. While this suggests that detection at the granularity of communication is promising, the associated resources share the same conversational context. Therefore, it is necessary to distinguish individual resources more precisely to minimize the impact on functionality, which is especially important for the first-party.

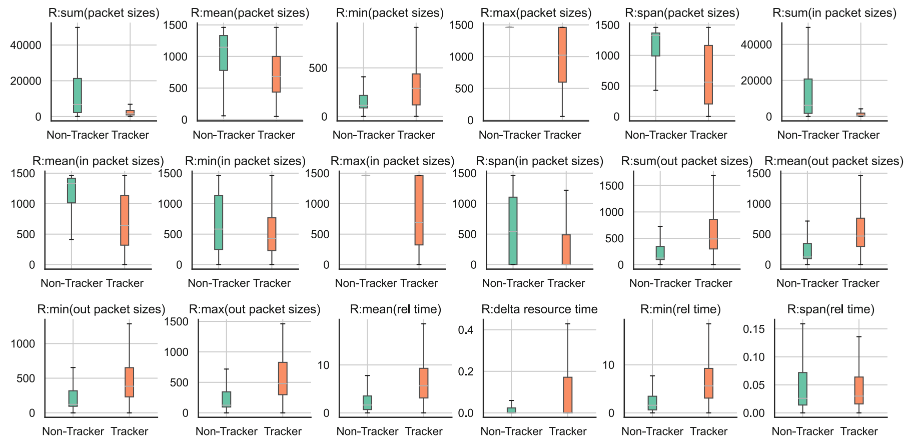


Fig. 6. Feature distribution of resources by size, direction, and time. As a rule of thumb, the less the boxes overlap, the more distinctive the feature.

This finding motivates us to look at the network properties of resources. For this purpose, we aggregate the packet attributes of resources by statistical location parameters (cf. Sect. 3.3). Figure 6 visualizes the discrimination power of the resource features. To achieve good classification results, the features should follow different distributions. For readability reasons, we only plot the relevant features without outliers. Overall, tracking resources tend to be smaller in size (sum) which also applies to individual packets (mean). A small span (max - min) represents equally sized packets. While the span of privacy-invasive resources is dispersed, non-trackers are more concentrated at the larger end. Harmless traffic is typically characterized by a small request and large responses. In contrast, a tracking pixel that only triggers requests to set or synchronize cookies, causes both short outbound and inbound traffic. The opposite case is primarily a result of tracking scripts that have small requests and large responses. Overall, trackers are significantly smaller than non-trackers in terms of packet size, except for `R:min(packet sizes)`, which varies substantially. This observation could possibly be explained due to the information leakage (e.g., via URL parameter [23]). Incoming traffic makes up the majority of packets (91%), so the feature distribution looks similar to the previous one. The opposite is true for outgoing packets. These are shifted further to the top due to the loss of information caused by either multiple parameters in the URL or POST data. The packet arrival time reveals that trackers are loaded later and with greater pauses between individual requests, while the total loading time of a resource (`R:span(rel time)`) is similarly short for both categories. Notably, the discrimination power of outgoing packet size and relative time is limited by a large interquartile range.

5.2 Feature Importance

Next, we look at the feature importance to discuss what features most accurately describe trackers. To begin with, we show in Fig. 7 a simplified version of the decision tree with a depth of 3, which still achieves an accuracy of 85.3% and a F_1 score of 0.823. It is an excellent tool for generalizing typical characteristics of tracking traffic. The tree consists exclusively of conversation features, as they are best suited to classify third-party trackers. Most resources (71%) can be described by two paths. Accordingly, a conversation is classified as tracker if it generates less than 53.8 kB of incoming traffic, it starts after 0.8 s, and its IP address has been observed over 3 times at a first-party. In contrast, harmless network traffic is characterized by at least 234.1 kB of incoming packets and an IP address prevalence of at most 29. While both paths lead to pure leaves, with Gini coefficients of 0.213 and 0.061, respectively, there are also minorities that do not yield good classification results. For example, if the incoming packets range from 53.8 KB to 234.1 kB and the conversation starts after 1.3 s, 48.6% of them are misclassified as trackers. Compared to most trackers, this class contains mostly advertisements, large JavaScript files, and twice as many requests to the host, resulting in a large communication volume.

To further understand which network features are most indicative of tracking, we compute the Gini importance for the decision tree and report the weights for the logistic regression in Fig. 8. For reference, we compare both scores with the point biserial correlation coefficient, which measures the individual predictive ability of each attribute. To improve readability, only significant features are shown.

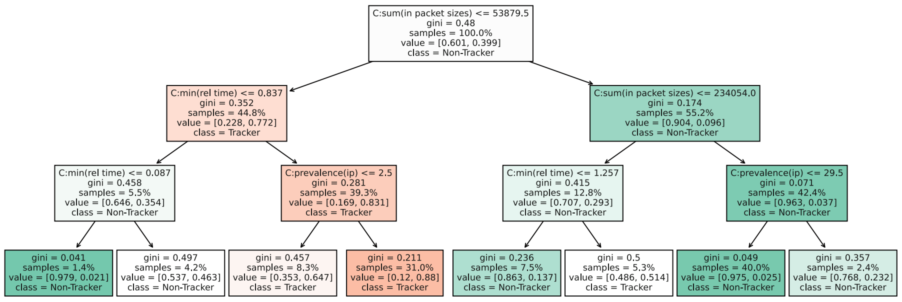


Fig. 7. Simplified decision tree with maximum depth of 3.

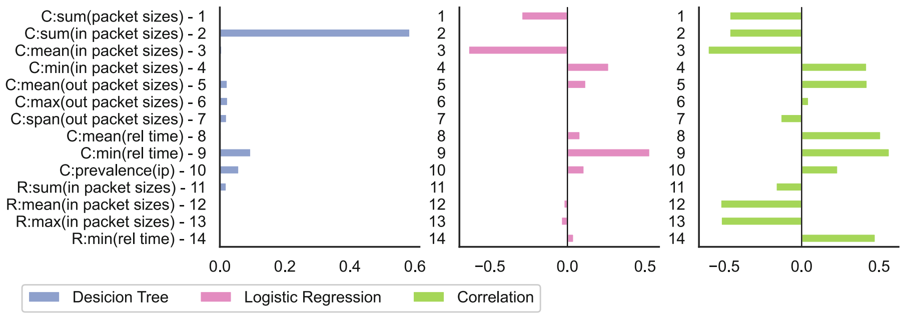


Fig. 8. Feature importance for different model.

Due to the regularization and the strong correlation between features, only a few features are selected. The most informative characteristics relate to the incoming packet size as well as the relative time. These results are in line with Fig. 6, as privacy-intrusive services generally do not provide useful web content ($r = -0.47$) and conversations with them tend to start after the main page has finished loading ($r = 0.57$), e.g., after an event has been triggered. In particular, the resource-specific features are less important, although the classification task involves the assessment of a resource. One explanation might be that web services provide very homogeneous content (either tracking or non-tracking) and therefore classification based on conversation is plausible. In addition, conversation features accumulate more information over the time horizon, leading to less variation compared to their resource counterpart. The importance of `C:prevalence(ip)` is consistent with our intuition that a higher frequency of a host on first-party websites is an indicator of tracking, since cross-site tracking requires widespread services ($r = 0.23$).

5.3 Classifier Performance

In this section, we evaluate the performance of our classifiers. To reduce variations, we test our classifier with 5-fold cross-validation and repeat this process

5 times, since results can vary depending on composition of training and test data. We are particularly interested in the distinctiveness of features at different levels of granularity, as perfect information may not be needed.

Table 3. Comparison of classification performance.

	All features		Conv. level		Res. level	
	Acc.	F1	Acc.	F1	Acc.	F1
Decision tree	0.899	0.875	0.882	0.860	0.853	0.824
Logistic regression	0.862	0.833	0.859	0.829	0.776	0.733

Overall Performance. The results in Table 3 show better overall performance for the decision tree than for the logistic regression. The decision tree can reproduce the blacklist labels 89.9% of the time. When looking at only a subset of features, conversational features perform better than resource-based features. As the communication flow contains more information, the conversational features exhibit more discriminative power. But the resource-specific features are still valuable, in detecting trackers for an impure communication with high service entropy. Therefore, combining both levels of granularity provides the best result.

First-Party vs. Third-Party. Even though the conversation features contribute the most to the classification, they are inadequate for detecting at first-party context. We observe different network characteristics for resources depending on the context of the party. For example, the average on `R:sum(out packet sizes)` is for first-party tracker 1.9 kB and for third-party tracker 0.9 kB, while `R:sum(in packet sizes)` is for non-tracker on average 27.5 and 21.7, respectively. Since first parties are only responsible for 5.6% of tracking, our classifiers are not able to generalize first-party trackers well. To compensate for the imbalance, we apply random undersampling to the majority classes. Each sample takes 30% of the dataset and is 5-fold cross-validated. This process is repeated 10 times, and we report the average in Table 4.

As the performance of first-party trackers is significantly lower than that of third-party trackers, correct detection in the first-party context remains a challenging task. While the decision tree based on resource features is best suited for first-party context, with a true positive rate of 78.2%, only 59.5% of the predicted positives are correct.

Table 4. Classification performance for first-party/third-party.

		First-Party		Third-Party	
		Precision	Recall	Precision	Recall
Decision Tree	All features	0.668	0.661	0.850	0.906
	Conv. level	0.688	0.657	0.811	0.926
	Res. level	0.595	0.782	0.802	0.882
Logistic Regression	All features	0.713	0.639	0.822	0.775
	Conv. level	0.715	0.638	0.823	0.775
	Res. level	0.464	0.745	0.768	0.752

Table 5. Classification performance for different resource types.

	Decision Tree			Logistic Regression		
	F1	Precision	Recall	F1	Precision	Recall
document	0.891	0.859	0.926	0.800	0.739	0.874
font	0.073	0.048	0.220	0.104	0.063	0.360
image	0.919	0.944	0.897	0.907	0.948	0.869
media	0.462	0.381	0.755	0.174	0.142	0.315
nan	0.876	0.842	0.914	0.835	0.761	0.925
other	0.838	0.811	0.868	0.777	0.731	0.830
script	0.847	0.881	0.816	0.524	0.857	0.378
stylesheet	0.302	0.221	0.525	0.147	0.103	0.272

Resource Types. To further uncover what our classifiers are able to detect, we present the classification results over various tracking object types in Table 5. Again, we apply undersampling due to heavy imbalance of types. Our approach performs equally well in distinguishing the different tracking types, except media (aka. video or audio), stylesheet, and font. However, they account for only 13.5% of the tracking and contribute little to the overall performance. The misclassifications are caused because the network flow of regular traffic is not sufficiently distinctive from tracking. The point biserial correlation coefficient is close to 0 for the three types for all network features.

6 Conclusion

In this work, we present preliminary experimental findings on identifying tracking resources based on TCP/IP features. Our classifier achieves a promising precision and recall of up to 85% and 90%, and is particularly suitable for third-party and image tracker detection. This shows that properties of tracking protocols are reflected on the network layer, which enables network-wide tracking protection without breaking encryption.

We conclude by discussing the limitations and outline future research directions. First, we assume a local attacker scenario with perfect information that offers a broad selection of features. However, our results suggest that analyzing the packet flow to a host is sufficient, as the performance improves marginally by including resource features. Besides the size of incoming packets, temporal features relative to the web page call provide useful information, but are usually only known to the client. Second, we only consider basic flow statistics (count, sum, mean, etc.) instead of conducting a time series analysis, though the latter may result in better performance [3,27]. Third, our experiment is limited by the low level of interaction on the homepage and focuses solely on web traffic. In future work, we plan to extend our analysis to mobile and IoT traffic.

References

1. 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, pp. 674–689 (2014). https://doi.org/10.1145/2660267.2660347
2. Agogo, D.: Invisible market for online personal data: an examination. Electron. Mark. **31**(4), 989–1010 (2020). https://doi.org/10.1007/s12525-020-00437-0
3. Akbari, I., et al.: A look behind the curtain: traffic classification in an increasingly encrypted web. In: Proceedings of the ACM on Measurement and Analysis of Computing Systems, vol. 5, no. 1 (2021). https://doi.org/10.1145/3447382
4. Bekos, P., Papadopoulos, P., Markatos, E.P., Kourtellis, N.: The Hitchhiker's guide to facebook web tracking with invisible pixels and click IDs. In: Proceedings of the ACM Web Conference 2023 (2023).https://doi.org/10.1145/3543507.3583311
5. Bhagavatula, S., Dunn, C., Kanich, C., Gupta, M., Ziebart, B.: Leveraging machine learning to improve unwanted resource filtering. In: Proceedings of the 2014 Workshop on Artificial Intelligent and Security Workshop, pp. 95–102 (2014). https://doi.org/10.1145/2666652.2666662
6. Brave: Fingerprint randomization (2020). https://brave.com/privacy-updates/3-fingerprint-randomization/
7. Bujlow, T., Carela-Español, V., Solé-Pareta, J., Barlet-Ros, P.: A survey on web tracking: mechanisms, implications, and defenses. Proc. IEEE **105**(8), 1476–1510 (2017). https://doi.org/10.1109/JPROC.2016.2637878
8. Chen, Q., Ilia, P., Polychronakis, M., Kapravelos, A.: Cookie swap party: abusing first-party cookies for web tracking. In: Proceedings of the Web Conference 2021, pp. 2117–2129 (2021). https://doi.org/10.1145/3442381.3449837
9. Demir, N., Theis, D., Urban, T., Pohlmann, N.: Towards understanding CNAME based first-party cookie tracking in the field. In: GI Sicherheit 2022 (2022)
10. Dimova, Y., Acar, G., Olejnik, L., Joosen, W., Van Goethem, T.: The CNAME of the game: large-scale analysis of DNS-based tracking evasion. Proc. Priv. Enhanc. Technol. **3**, 394–412 (2021). https://doi.org/10.2478/popets-2021-0053
11. 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, pp. 1388–1401 (2016).https://doi.org/10.1145/2976749.2978313

12. Fathi-Kazerooni, S., Kaymak, Y., Rojas-Cessa, R.: Tracking user application activity by using machine learning techniques on network traffic. In: 2019 International Conference on Artificial Intelligence in Information and Communication, pp. 405–410 (2019). https://doi.org/10.1109/ICAIIC.2019.8669040
13. Gugelmann, D., Happe, M., Ager, B., Lenders, V.: An automated approach for complementing ad blockers' blacklists. In: Proceedings on Privacy Enhancing Technologies, pp. 282–298. No. 2 (2015). https://doi.org/10.1515/popets-2015-0018
14. Hayes, J., Danezis, G.: K-Fingerprinting: a robust scalable website fingerprinting technique. In: Proceedings of the 25th USENIX Conference on Security Symposium, pp. 1187–1203 (2016)
15. Iqbal, U., Shafiq, Z., Qian, Z.: The ad wars: retrospective measurement and analysis of anti-adblock filter lists. In: Proceedings of the 2017 Internet Measurement Conference, pp. 171–183 (2017). https://doi.org/10.1145/3131365.3131387
16. Iqbal, U., Snyder, P., Zhu, S., Livshits, B., Qian, Z., Shafiq, Z.: AdGraph: a graph-based approach to ad and tracker blocking. In: 2020 IEEE Symposium on Security and Privacy, pp. 763–776 (2020). https://doi.org/10.1109/SP40000.2020.00005
17. Jha, N., Trevisan, M., Vassio, L., Mellia, M.: The internet with privacy policies: measuring the web upon consent. ACM Trans. Web **16**(3) (2022). https://doi.org/10.1145/3555352
18. Ma, Q., Huang, W., Jin, Y., Mao, J.: Encrypted traffic classification based on traffic reconstruction. In: 2021 4th International Conference on Artificial Intelligence and Big Data, pp. 572–576 (2021).https://doi.org/10.1109/ICAIBD51990.2021.9459072
19. Mayer, J.R., Mitchell, J.C.: Third-party web tracking: policy and technology. In: 2012 IEEE Symposium on Security and Privacy, pp. 413–427 (2012). https://doi.org/10.1109/SP.2012.47
20. Mohajeri Moghaddam, H., et al.: Watching you watch: the tracking ecosystem of over-the-top TV streaming devices. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pp. 131–147 (2019). https://doi.org/10.1145/3319535.3354198
21. Panchenko, A., Niessen, L., Zinnen, A., Engel, T.: Website fingerprinting in onion routing based anonymization networks. In: Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society, pp. 103–114 (2011). https://doi.org/10.1145/2046556.2046570
22. Papadogiannakis, E., Papadopoulos, P., Kourtellis, N., Markatos, E.P.: User tracking in the post-cookie era: how websites bypass GDPR consent to track users. In: Proceedings of the Web Conference 2021, pp. 2130–2141 (2021). https://doi.org/10.1145/3442381.3450056
23. Roesner, F., Kohno, T., Wetherall, D.: Detecting and defending against third-party tracking on the web. In: Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation, pp. 155–168 (2012)
24. Shuba, A., Markopoulou, A., Shafiq, Z.: NoMoAds: effective and efficient cross-app mobile ad-blocking. Proc. Priv. Enhanc. Technol. (4) (2018)
25. Siby, S., Iqbal, U., Englehardt, S., Shafiq, Z., Troncoso, C.: WebGraph: capturing advertising and tracking information flows for robust blocking. In: 31st USENIX Security Symposium, pp. 2875–2892 (2022)
26. Siby, S., Juarez, M., Diaz, C., Vallina-Rodriguez, N., Troncoso, C.: Encrypted DNS $->$ privacy? A traffic analysis perspective. In: Network and Distributed System Security Symposium (2020)

27. Sirinam, P., Juarez, M., Imani, M., Wright, M.: Deep fingerprinting: undermining website fingerprinting defenses with deep learning. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (2018). https://doi.org/10.1145/3243734.3243768
28. Sjösten, A., Snyder, P., Pastor, A., Papadopoulos, P., Livshits, B.: Filter list generation for underserved regions. In: Proceedings of The Web Conference 2020, pp. 1682–1692 (2020). https://doi.org/10.1145/3366423.3380239
29. Snyder, P., Vastel, A., Livshits, B.: Who filters the filters: understanding the growth, usefulness and efficiency of crowdsourced ad blocking. SIGMETRICS (2020). https://doi.org/10.1145/3392144
30. Varmarken, J., Le, H., Shuba, A., Markopoulou, A., Shafiq, Z.: The TV is smart and full of trackers: measuring smart TV advertising and tracking. Proc. Priv. Enhanc. Technol. **2**, 129–154 (2020). https://doi.org/10.2478/popets-2020-0021
31. Yuan, S., Abidin, A.Z., Sloan, M., Wang, J.: Internet advertising: an interplay among advertisers, online publishers, ad exchanges and web users. arXiv: 1206.1754 (2012)
32. Zuboff, S.: The Age of Surveillance Capitalism: The Fight for a Human Future at the New Frontier of Power. PublicAffairs (2018)