UNIVERSITY OF BAMBERG

Applied Computer Science Degree Program in the
Faculty of Information Systems and Applied Computer Sciences

Master's Thesis

Semi-automated Interactive Website Scanning for Comprehensive Privacy Analysis

BY

Martin Endreß

Supervisor:
Prof. Dr. Dominik Herrmann

Privacy and Security in Information Systems Group

September 2022

Abstract

The implementation of the GDPR and other privacy initiatives continues to be inadequate, and there exists a severe imbalance of power between website providers and end users. Recent studies in the field of web privacy have shown that many websites do not provide users with the necessary consistent and transparent consent controls, either by ignoring their choices or nudging users into unwanted behavior. Over the years, several measurement platforms have been proposed with the goal of monitoring the impact of these practices on user privacy. However, available privacy measurement methodologies do not take into account complex interactions with websites and often only measure the landing page.

In this master thesis, we address this shortcoming and propose a semi-automated interactive measurement platform that facilitates privacy analysis of complex interactions with websites. Users of the platform perform interactions on a remote browser running in an isolated and controlled environment. The behavior of the website is recorded in the background and stored for subsequent analysis. We demonstrate the merits of the approach through a study that analyzes the affirmative consent interaction with cookie banners on 300 German websites. The study shows that the requirement of freely given consent is often not implemented, as 79 percent of consent notices impede the user from interacting with the website. Further, 22 percent of websites contact third parties classified as trackers even before the positive consent was given. We also show that, compared to previous studies of website interactions, our semi-automated approach makes fewer assumptions regarding website selection and can therefore provide more representative results, albeit with the drawback of smaller scale.

We imagine that the proposed platform be used to study more complex interactions, include longitudinal measurements, and made more accessible to average web users. This would not only inform the public debate about privacy-invasive tracking practices, but it would also empower individual users with their limited choices.

Contents

1 Introduction

The choices of website providers on technologies and third party services have great influence on the privacy of their users. The most common way of monetizing websites is through the use of behavioral advertising. This technique involves websites using advertising networks that track the users' browsing behavior to serve targeted ads that users are likely to engage with. The practice has evolved over the years and new techniques are regularly developed. Historically, studies like the one by Krishnamurthy and Wills have noticed a steady increase in the extent websites use third parties during the mid 2000s [KW09]. A more recent study by Englehardt and Narayanan observed a trend towards monopolization of the third-party ecosystem [EN16]: They show that tracking is ubiquitous across the web and highly centralized, meaning that only a few tracking entities are present on the vast majority of websites. In addition to the extent of commercial web tracking, Englehardt et al. show the severe implications of widespread tracking associated with nation-state actors [Eng+15]: The authors demonstrate that the NSA is able to access and reconstruct a majority of a user's browsing history in the European Union (EU) using tracking cookies.[1]

1: Note that this analysis was conducted in 2015, three years before the GDPR came into force.

With the goal of improving personal data protection, the EU has introduced laws such as the ePrivacy Directive [Euro2; Euro9] and the General Data Protection Regulation (GDPR) [Eur16]. The ePrivacy Directive introduced legal requirements that websites have to implement when collecting personal information about its users. However, a study by Trevisan et al. shows that the ePrivacy Directive has failed to significantly improve user privacy [Tre+19]: The authors name inconsistent interpretation and enforcement by EU member states, lack of automatic tools that verify website's adherence, and the public's missing awareness of the threat to user privacy as the three main reasons for the failure of the ePrivacy Directive. In 2016, the EU introduced the GDPR as a replacement of the directive. It came into force two years later in 2018 and applied to all websites offering their services to EU citizens. The directive was aimed to provide more consistent and enforceable data protection mechanism which require strict conditions for processing of personal data [Eur16].

Motivation While the GDPR has improved data protection on the web to some extent [SK19; Urb+18], it still does not not solve all user privacy issues and falls short on fundamental topics: There is still an imbalance of power between websites and users [ABL15], and the techniques used by websites often render the desired transparency protocols and privacy settings ineffective. Most importantly,

- explicit control over privacy settings [BAL10],
- use of default options [GA05],
- dark patterns [Mat+19], and
- inconsistent or uninformed decision making by users [JPJ05]

negatively impact user privacy.

Relevant literature proves that the state of privacy for web users has worsened over the last two decades. The aforementioned imbalance of power between websites and users hinders progress towards privacy protection. Furthermore, several attempts of EU data protection regulation have failed to fundamentally change the state of affairs in favor of user privacy. Some authors go even further and argue that the concept of informed consent has to be questioned as a whole [BN14; Kre+20; Nis11].

Based on these observations, a number of researchers point to the importance of web privacy measurement tools in empowering and informing users [Bol+22; EN16; Maa+17]. For instance, Englehardt and Narayanan argue that web privacy measurement can 'play a key role in keeping online privacy incursions and power imbalances in check" [EN16]. They advocate that measurement tools like their `OpenWPM` platform be made widely available to the public rather than just within the research community. With this tool, so the authors argue, regular non-expert web users will have the ability to investigate the online tracking behavior of the websites they use. The measurement tool `CookieBlock` is also intended to inform and empower users [Bol+22]. It enforces GDPR compliant cookie consent by classifying cookies by their use categories and blocking or deleting cookies that the user does not accept. The authors of `CookieBlock` argue that this type of privacy measurement and classification serves as an intermediate step towards a transparent and privacy-friendly web. Another privacy measurement tool designed primarily for end users is `PrivacyScore` [Maa+17]. The platform by Maaß et al. compares privacy and security features of websites. It empowers individuals to identify privacy-invasive tracking practices and compare websites within user defined categories.

While the three exemplar approaches help users by informing them of existing tracking practices, they are not able to comprehensively model actual user behavior on the web. Bollinger et al. explicitly name the limited interaction with the website as a shortcoming of their tool [Bol+22]. Aqeel et al. also criticize that the majority of research literature on web privacy measurement does not take into account internal web pages [Aqe+20].[2] Instead, the proposed platforms often consider only one page per website, usually the landing page. The authors observe significant characteristic differences between the landing pages and internal pages with respect

2: Landing pages refer to the root document of a website, internal pages refer to all other pages.

to load times, number of unique third parties, and amount of content present. Urban et al. also encourage further investigation of internal pages, as these embed significantly more third parties and set more cookies [Urb+20]. Moreover, they stated that it is cumbersome even for website providers themselves to get an overview of all loaded third parties. This is especially relevant when these are not embedded by the first party directly, but instead loaded as fourth and fifth parties.

In this master's thesis, we address the deficit in measurement platforms that allow for user interaction with websites. We propose a semi-automated interactive web privacy measurement platform capable of modeling complex user interaction with websites. Any interactions and browsing behavior that a user performs on websites can be studied in this way. The platform is conceptualized as a semi-automated research tool to study possibly unobserved tracking techniques on internal web pages.

Research Questions This master's thesis addresses the following three research questions:

1. What would be a suitable architecture for a web privacy measurement platform that allows users to interactively scan websites for privacy-related behaviors?

The first research question is concerned with the software architecture and design decisions related to the web privacy measurement platform. It is supposed to enable users to scan websites interactively and with little overhead. In essence, the platform should not require prerequisites like installing plugins. During a website scan, user interaction and browser behavior are recorded and stored for later analysis. The answer to the first research question includes an analysis of the characteristics that an interactive web measurement platform should possess. This analysis refers to the proposed research prototype.

2. What model is suitable for analyzing website interactions for privacy-related properties and behavior?

The second research question deals with the model of an interactive website scan as well as its analysis. Answering it requires an analysis of the stored information about a scan, any missing information that is not modeled, and the representation of the analysis results. In this context, multiple scans of different websites or repeated scans of a single website also become relevant. This analysis also refers to the proposed research prototype.

3. How effective is an interactive privacy measurement platform in quantifying the tracking practices of a website as opposed to previous approaches?

The third research question evaluates the privacy measurement platform as a research tool. It is answered with the help of a case study on current tracking prevalence on a set of websites. The advantages and limitations of the research tool are discussed and compared with previous approaches.

Thesis Structure The rest of this master's thesis is structured as follows: In Chapter 2, we provide background information on tracking techniques, data protection legislation, and technologies used in the thesis. In Chapter 3, we discuss related work regarding the GDPR, privacy measurement, and characteristics of websites. In Chapter 4, we present and discuss our interactive measurement platform. This includes high-level idea and requirements, but also software architecture and corresponding design decisions. In Chapter 5, we report on a case study that was performed to demonstrate the merits of the approach. In Chapter 6, we discuss advantages and shortcomings of the approach and sketch future work. We conclude the thesis in Chapter 7.

2 Background

This chapter explains concepts and technologies relevant to the thesis. Whenever applicable, design decisions and advantages of these concepts are discussed. In addition, relevant terminology is defined.

2.1 ID Synchronization

Modern websites are complex and often embed a diverse set of third parties. As HTTP is a stateless protocol, a number of identifiers are stored in the browser during a website visit that enable first and third parties to re-identify the user on subsequent visits.[1] User actions are monitored and associated with the user, forming a unique profile. A crucial part of the tracking ecosystem is the communication and linkage of the user identifiers among third parties like data brokers and ad networks. In Chapter 5, we measure the prevalence of this process called ID synchronization using the techniques proposed in a study by Papadogiannakis et al. [Pap+21].

Figure 2.1 illustrates an example of a cookie synchronization operation between the third parties `tracker.com` and `advertiser.com`:[2] Assume that a user visits three websites in sequence. They first visit `site1.com`, which embeds a JavaScript resource from the third party `tracker.com`, instructing the client to set an identification cookie with the value 'user123'. The user then visits `site2.com`. It embeds an image resource from `advertiser.com` which sets a cookie storing another user id, namely 'userABC'. The third website visit to `site3.com` is where the cookie synchronization takes place: The website embeds JavaScript code from `tracker.com` that requests some resource, resulting in re-identification of the user as 'user123'. The webserver of `tracker.com` responds with a redirect to `advertiser.com` containing the identifier as a query parameter. The client then sends this GET request to `advertiser.com` alongside the cookie 'userABC' acquired on `site2.com`. As a result of the request, `advertiser.com` learns that both identifiers represent the same user and that this user visited `site3.com`.

1: For extended explanation and discussion of ID synchronization, we refer to the early analysis by Olejnik, Tran, and Castelluccia [OTC14] and a more recent study by Papadopoulos, Kourtellis, and Markatos [PKM19].

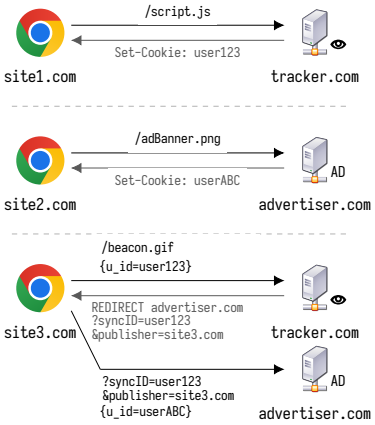


Figure 2.1: Example of cookie synchronization operation between `advertiser.com` and `tracker.com`. `{key=value}` denotes a cookie. (adjusted from Papadopoulos, Kourtellis, and Markatos [PKM19])

2: The example is adjusted from Papadopoulos, Kourtellis, and Markatos [PKM19].

2.2 Data Protection Legislation

Of the data protection laws enacted by the EU over the past several decades, this section focuses on the GDPR [Eur16]. It was introduced to the EU parliament in 2016 and came into force in May of 2018. The legislation was designed to improve on the ePrivacy Directive and establish a consistent and enforceable mechanism across the EU. While a comprehensive discussion and analysis of the GDPR is beyond the scope of this paper[3], this section addresses relevant aspects of the legislation with respect to tracking. The implications and adherence to the GDPR are discussed further in Section 3.1.

3: The GDPR also covers practices outside of the web.

Processing of Personal Data Articles 2 and 3 define the scope of the GDPR [Eur16, p. 32 sq.]: The regulation applies to all cases of data processing that is automated or part of a filing system. Geographically, the regulation applies to all instances of data collection of subjects located within the EU and the European Economic Area (EEA), regardless of the location of data processing.[4] Consequently, any service offered to EU users that collects personal data is subject to legislation.

4: In the interest of conciseness, the fact that the EEA is also covered will be omitted and we will write EU instead.

The legal basis for websites to process personal data about individual users is captured in Articles 6 and 7 [Eur16, p. 36 sq.]: According to Art. 6 Par. 1, data collectors are required to either (a) receive consent by the data subject, have a (b) contractual or (c) legal obligation, or verify the presence of either a (d) vital, (e) public, or (f) legitimate interest in data processing. If the website bases the data processing on the consent of the user, special conditions apply. Recital 32 states that

> "Consent should be given by a clear affirmative act establishing a freely given, specific, informed, and unambiguous indication of the data subject's agreement to the processing of personal data relating to him or her, such as by a written statement, including by electronic means, or an oral statement." [Eur16, p. 6]

Art. 7 Par. 4 deals with freely given consent and prohibits any interference with the performance of the contract based on consent or refusal [Eur16, p. 37]. The user's consent decision must therefore not be subject to any conditions like paid subscriptions or influenced through user impediment. This restriction is relevant to the appearance of consent notices.

Enforcement As specified in Article 83, a failure to comply with the GDPR may result in fines of up to 20 million Euros or 4 percent of the worldwide annual turnover [Eur16, p. 83]. In a legal analysis, Erickson highlighted the important role that regional data protection authoritys (DPAs) play in GDPR enforcement [Eri19, p. 885 sqq.]: She welcomes the rights given to individual users under the GDPR and argues that the DPA-based mechanism can be a role model for other countries like Brazil. Wolff and Atallah are less optimistic with respect to effective enforcement [WA21]: In their analysis of 261 cases during the first 24 months after the GDPR came into force, they found that fines were relatively low and comparable to verdicts prior to the GDPR.

Effective GDPR enforcement also requires capable and independent supervisory institutions. For this purpose, Article 51 mandates that every member state must establish an independent supervisory authority that is equipped with sufficient human, technical, and financial resources [Eur16, p. 66]. Section 3.1 discusses the extent to which the implementation of this article is sufficient.

Focus on Tracking Techniques Several parts of the GDPR are specifically concerned with online user tracking. For instance, Recital 30 focuses on tracking identifiers [Eur16, p. 6]: It specifies that any identifier provided by either a device, application, tool, or as a result of a protocol is subject to the legislation. This covers both stateless and stateful tracking techniques.[5] While terms like "stateless" or "fingerprinting" are not explicitly mentioned in the text, this abstract definition of personal information covers new and unobserved techniques implicitly by design.[6]

In the Planet49 case, the Court of Justice of the EU clarified that analytical trackers like `Google Analytics` must be explicitly agreed to and cannot be declared as necessary for operation by the website [Jus19]. Furthermore, the court ruled that pre-ticked checkboxes consent notices are insufficient as these practices do not constitute active user consent.

The consequences of the data protection legislation effort in the EU is discussed further in Section 3.1. As an answer to research question 1, we propose a measurement platform to analyze the legal compliance of websites with the GDPR and other data protection laws. While the proposed approach does not detect or classify practices of websites that violate the GDPR, it facilitates the process of finding possible violations. Decisions on violations can only be done by the courts.

5: We refer to Englehardt and Narayanan for a comprehensive explanation and analysis of stateful and stateless tracking techniques [EN16]

6: For an extensive discussion on GDPR with regard to fingerprinting see https://www.eff.org/de/deeplinks/2018/06/gdpr-and-browser-fingerprinting-how-it-changes-game-sneakiest-web-trackers (archived on 20.06.22)

2.3 Apache Guacamole

`Apache Guacamole` is a remote desktop gateway that runs in the browser.[7] The project offers access to remote desktop environments over VNC, RDP, and SSH connections without the need for client-side plugins or prior configuration.

An overview of the different Guacamole components and communication channels is shown in Fig. 2.2. The Guacamole server acts as a middleman between the `HTML5` web client and the set of remote desktops: The web client communicates with a Java web service using the proprietary Guacamole protocol. The remote viewport as well as user interaction is transmitted from and to the server in a protocol agnostic way. Both the web client and the Java servlet application do not know which remote desktop protocol is being used in the Guacamole backend. The Servlet container connects to a background process called `guacd`, again using the Guacamole protocol. This daemon establishes remote connections to respective desktop servers either through VNC, RDP, or SSH depending on the provided configuration.

7: Apache Guacamole: https://guacamole.apache.org/ (archived on 17.06.22)

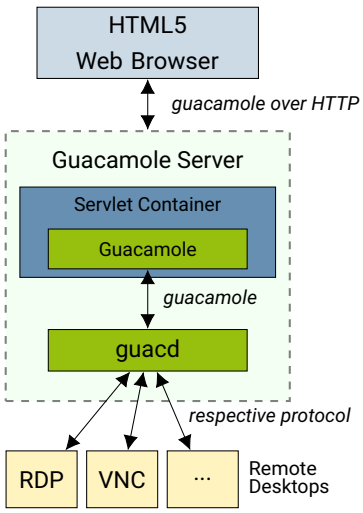
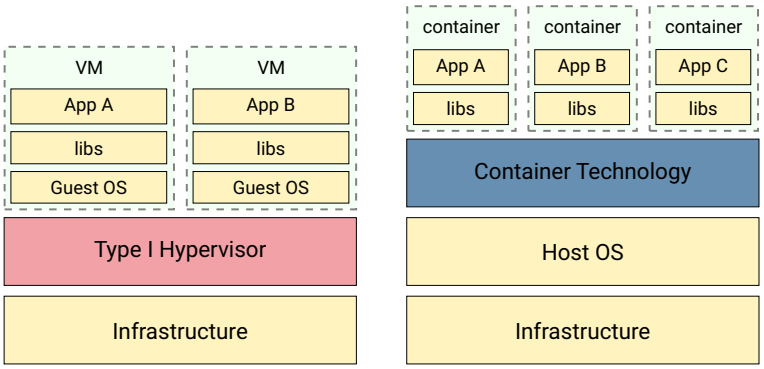


Figure 2.2: Guacamole components and communication channels. Adjusted from https://guacamole.apache.org/doc/gug/guacamole-architecture.html (archived on 17.06.22).



(a) Virtual Machine **(b)** Container

Figure 2.3: Architecture comparison between hypervisor-based virtual machines and container technology. Adjusted from [Fel+15; Sol+07].

The thesis makes use of Guacamole as part of the interactive measurement platform. Users of the platform can connect to a remote browser instance. The fact that Guacamole requires no prior installation lowers the barrier of entry for users of the measurement platform. Unlike with a local approach, here the platform has control over the browser.

2.4 Container Technology

Linux containers are a lightweight virtualization technology that, unlike traditional virtual machines (VMs), do not require hypervisor abstraction [Sol+07]. Since no hardware translation is used, the host operating system is shared among all containers. This core difference among the two technologies is illustrated in Fig. 2.3. The containers run on the host operating system and are isolated in terms of resource usage and dependencies. Resource isolation means that the CPU, memory, and disc I/O resources are explicitly assigned and monitored for each container. To achieve this isolation, Linux containers mainly use the three Linux kernel features cgroups, namespaces, and chroot.

There are a number of container runtimes and formats with different use cases and advantages, governed by the open container initiative (OCI).[8] While `Docker` is the most prominent among them, several other container engines like `systemd containerd`, `Podman`, or `LXC` have been proposed in recent years.

In the proposed privacy measurement platform, we make use of Podman as it offers several advantages over other Linux container technologies: Most importantly, due to the research effort by Red Hat, Podman integrates well with systemd services:[9] As a result, Podman containers can run as systemd services and systemd services can run inside Podman containers. Further advantages of Podman include that Podman is daemonless and provides a RESTful API for managing containers and images.

8: The Open Container Initiative proposes a governance structure for container formats: https://opencontainers.org/about/overview/ (archived on 17.06.22)

9: How to run systemd in a container: https://developers.redhat.com/blog/2019/04/24/how-to-run-systemd-in-a-container (archived on 17.06.22)

3 Related Work

In this chapter, we provide an overview of relevant literature and discuss their relation to the thesis. Previous approaches and results in privacy measurement are summarized and put into context. Another area of focus is the need for highly available, comprehensive privacy measurement methods. The end of each paragraph describes where in the paper the literature discussed was used or considered.

3.1 GDPR Compliance and Consequences

The impact of the GDPR and its adherence is being extensively researched, often times with a devastating perspective for user privacy. This section highlights interesting observations in recent literature and discusses how they relate to and possibly motivate aspects of privacy measurement. The selection of literature does not aim to be comprehensive. Instead, we examine a set of studies that are relevant to the capabilities of a measurement platform and the motivation thereof. We refer to the survey by Kretschmer, Pennekamp, and Wehrle for a comprehensive analysis of the impact of the GDPR [KPW21].

In a longitudinal study, Hils, Woods, and Böhme observed that the GDPR resulted in the emergence of consent management providers (CMPs), third party services that offer user consent collection [HWB20]. They found that the number of websites using CMPs increased from one percent in 2018 to 10 percent in 2020. While the authors note the privacy implications associated with further consolidation of information by private CMPs, they also emphasize the benefits, such as facilitated automation in measurement methods. This property was used by Bollinger et al. in a study on GDPR compliance of consent notices on websites using CMPs [Bol+22]: The authors use class labels and cookie categories provided by a selection of five popular CMPs to analyze respective websites for GDPR compliance. This reduced the number of analyzed websites from 6 million to 30 000. This methodology, in turn, allowed the authors to use the labeled data provided by the CMPs as the ground truth. They found that 95 percent of websites contained violations such as ignored or implied consent, assignment of the wrong cookie purpose, or undeclared categories [Bol+22], in line with previous research [MBS20; Meh20]. Bollinger et al. note that the 200-fold reduction of websites introduces a selection bias and see manual cookie collection as a possible

improvement over this methodology [Bol+22]. Furthermore, the authors name the limited interaction with the website, namely through account creation or login as reasons for distortions of the results. In our work, we abstract from this automated analysis approach: The proposed measurement platform is intended to be agnostic of website technologies and website selection. While this semi-automatic approach also restricts the number of websites that can be analyzed efficiently, we argue that the risks of selection biases and other side effects resulting from automation are reduced. Any interaction with websites that can be performed in a browser can also be performed in the proposed approach.

Recent literature reports on a large number of GDPR violations through privacy-invasive interface design choices that exploit human cognitive biases [Gra+18; Mat+19]. For instance, Mathur et al. study the extent of dark patterns on shopping websites through an automated measurement technique [Mat+19]. The authors found that many websites use visual interruptions to induce a user action in their interest, such as obtaining consent that was not freely given or activating a privacy-invasive feature. The authors propose an interactive measurement approach which simulates a shopping interaction. Our semi-automated approach can aid the effort of detecting novel and unseen privacy-invasive dark patterns which might fall under the radar when using automated tools.

A technical report by the Brave browser team from 2020 provides possible reasons for the lacking GDPR compliance in the EU: Ryan and Toner found that only 305 technology specialists are employed across the EU member states' 45 DPAs [RT20]. Almost a third of these are employed in Germany alone. Also, the Irish DPA as Europe's "lead authority"[1] is underfunded and hires do not keep up with the rise in GDPR complaints. From these observations, the authors conclude that the DPAs of EU member states are not equipped with sufficient resources to enforce the GDPR and therefore violate Article 51. The report suggests more funding for DPAs and strict enforcement of Article 51 as a possible solutions. We argue that our measurement approach can serve as an aid in the work by the DPAs.

1: The authors of the report attribute higher responsibility to Irish authorities as they are responsible for the supervision of tech giants like Google and Facebook.

The recent literature discussed shows a diverse, sophisticated, and an increasing number of violations of the GDPR. In particular, researchers observed violations as a result of a consolidation of the third-party and CMP ecosystem [Bol+22; HWB20] and an expansion of dark pattern use [Mat+19]. These findings depict a highly complex environment of competing interests and lacking governance by authorities. We also discussed the inability of federal authorities to enforce privacy laws [RT20]. Our work is motivated by the lack of measurement approaches that focus on complex user interactions without the need for restrictive assumptions.

3.2 Informed Consent

As a consequence of countless observations of GDPR violations in recent literature, a number of researchers have questioned the ethical and legal basis of informed consent as a whole.

Kreuter et al. found that users are willing to share extensive personal information about themselves and do not differentiate between data requests [Kre+20]. While the authors establish that, on a technical level, user consent was given for the collection of personal data and its purpose, they argue that the increased burden of obtaining and processing the relevant information impedes informed decision making.

Barocas and Nissenbaum go even further in their assessment of privacy with respect to informed consent [BN14, p. 57]: The authors' discussion addresses the complex flow of data among different parties with varying interests, just as can be observed on many websites that use tracking technologies. They argue that established techniques like plain text, simple-to-understand privacy policies and consent notices are subject to the "transparency paradox".[2] It states that these established techniques either overwhelm even experienced users because the parties and practices of websites are complex and change rapidly, or they do not convey data use in a way that is relevant to the decisions individuals must make. She argues that informed consent has the fundamental flaw that users are never able to grasp all facts and consequences relevant to the consent choice. She therefore concludes that informed consent in its current form must be overcome. In its place, she envisions well-integrated, privacy-friendly public policies that focus on website practices in relation to social norms themselves [Nis11].

2: The transparency paradox was initially proposed by Nissenbaum in 2011 [Nis11].

Barocas and Nissenbaum also underline the imbalance of power between websites and users [BN14, p. 63]. They heavily criticize big data advocates that see privacy as an unsustainable obstacle incompatible with technological advances in big data. These advocates include Mark Zuckerberg, who stated that users of social media "no longer have an expectation of privacy" and that Facebook must therefore change and reflect these new social norms.[3]

3: NYT author Kirkpatrick argues that Facebook is not reacting, but is instead itself an 'agent of social change'. Source: https://www.nytimes.com/external/readwriteweb/2010/01/10/10readwriteweb-facebooks-zuckerberg-says-the-age-of-privac-82963.html (archived on 14.06.22)

The criticism discussed in this section gives more context to the need for extensive privacy measurement approaches. For the reasons outlined in this section [BN14; Kre+20; Nis11], we argue that the decision of how much tracking is acceptable should not be the responsibility of the end user. Expanding comprehensive privacy measurement approaches can help inform the overall public debate about informed consent and privacy-invasive practices. We agree with Bollinger et al. that expanding privacy measurement centered around end users can only be an intermediate step on the road to a privacy-friendly web [Bol+22].

3.3 Privacy Measurement

In this section, we present and discuss proposed privacy measurement platforms and techniques. We differentiate between automated and semi-automated measurement techniques and note what conclusions we drew from the literature study.

Automated Measurement A number of automated measurement platforms have been proposed in recent years. An automated privacy measurement platform usually consists of an instrumented browser instance simulating a user in an automated fashion and a data collection system.

A study by Papadogiannakis et al. analyzes the tracking behavior of websites in the "post-cookie" era [Pap+21]: The authors anticipate that the reliance on cookies for tracking purposes will decrease and stateless techniques will increase instead.[4] Using the consent automation tool `Consent-O-Matic`, they measure the extent of first-party ID leaking, ID synchronization and browser fingerprinting before and after the consent interaction.[5] As summarized in Table 3.1, websites make extensive use of privacy-invasive tracking techniques even prior to user consent. Note that these values represent lower bounds to tracking prevalence as the detection heuristics are limited by nature. For instance, fingerprinting is detected through the comparison of JavaScript stack traces with a dictionary of fingerprinting function names. As a validation of our measurement approach, we replicate and extend this study using the proposed measurement platform. The interactive approach proposed in this thesis abstracts from the dependence on technologies such as Consent-O-Matic which only work on a selection of websites. This limitation becomes clear when looking at the website selection of the study: Only three percent of the initial set of 850 000 websites were examined as these require corresponding CMPs for consent automation.[6]

`OpenWPM` is a web privacy measurement platform proposed by Englehardt and Narayanan [EN16]. The platform instruments the Firefox browser and simulates user behavior in a controlled way. During the automated website visits, it records the browser behavior including responses, cookies, JavaScript execution data, and other information. Using this platform, researchers can perform extensive studies that analyze websites for stateful and stateless tracking prevalence. In addition, the platform models the location of third parties and is able to work with browser profiles allowing for diverse measurement methodologies. A crucial design goal of OpenWPM is to enable a robust and human-like measurement environment. To this end, the authors chose the fully-fledged Firefox browser over lightweight browsers like PhantomJS. Even with corresponding `UserAgent` strings, the authors observed that on a crawl of the top 100 websites, PhantomJS requests 10, 15, and 30 percent fewer html, text, and stream resources, respectively. From this observation, we conclude that the choice of browser is a crucial prerequisite for accurate measurement. OpenWPM offers a custom headless mode that runs the measurement instance in a container and draws the graphical interface to a virtual frame

4: This prediction is consistent with observations from other studies [Dab+19; EN16].

5: Consent-O-Matic browser extension: https://github.com/cavi-au/Consent-O-Matic (archived on 05.07.22)

Table 3.1: Fraction of websites that exhibit first-party user ID leaking, third-party ID synchronization, and fingerprinting. Measurements are taken before and after user consent. (n=27 180, adjusted from Papadogiannakis et al. [Pap+21])

Technique	before c.	after c.
ID leak	52.4%	65.4%
ID sync	24.0%	29.6%
fingerprinting	1.0%	1.2%

6: 26 percent of websites were filtered out due to unavailability, robots.txt specification, or pornographic material.

buffer using the display server `Xvfb`. The measurement browser instance proposed in this thesis is derived from this headless mode. Instead of discarding the output of the display server, it is made available to the user using Guacamole.

Maaß et al. developed the web privacy and security measurement platform `PrivacyScore` [Maa+17]. It measures privacy and security characteristics of websites and presents them in a way that allows for comprehensive comparison and ranking. The website scan analyzes privacy characteristics like the presence of third party trackers, fingerprinting scripts, or the server location. The platform is conceptualized as a tool to facilitate the comparison of privacy characteristics for end users, researchers, and government entities. The authors also note the importance of ease of use for users to create awareness for privacy-invasive behavior by websites. We conclude from their platform evaluation that a transparent and easy-to-use approach is essential in empowering and informing users. We also use their proposed tracker detection methodology in our case study.

Semi-automated Interactive Measurement Several approaches proposed in recent literature rely on semi-automated privacy measurement approaches. Often times, semi-automated techniques complement automated approaches. Here, we highlight a few approaches and describe what we learn from them.

Matte, Bielova, and Santos compare the practices of consent notices and analyze their GDPR compliance on EU websites [MBS20]. In particular, their analysis focuses on the effectiveness of the Transparency and Consent Framework (TCF), an industry standard on consent collection. The authors first automatically search for websites that use the TCF. They then perform a semi-automated crawl on a sample of 560 websites. In this measurement, the authors provide affirmative consent and record if consent notices exhibit a 'pre-selected choice' or have 'no way to opt out'. Each of the scenarios constitute a GDPR violation. The authors in addition detect 'non-respect of choice'. As a result, the analysis revealed numerous of these three violations: The authors measure that 12 percent of studied websites register affirmative consent without user interaction and five percent do not respect the user's choice at all. Collectively, 54 percent of websites using the TCF exhibit at least one such violation. As a solution, the authors believe the European authorities have a responsibility to provide more guidance or better enforce the regulations. The study discussed here is a good example of combining automated techniques with semi-automated website crawls. We argue that our measurement approach facilitates these combined approaches.

Hils, Woods, and Böhme conducted a user study on the impedance that privacy aware users face when interacting with the consent notice of websites embedding CMPs [HWB20]. The authors measured user preference based on the various options available as well as the differences in interaction time depending on the user's action. For websites with options to accept or reject all cookies, the authors measured that a median user needed 3.2 seconds to accept and 3.6 seconds to decline all cookies. The median time users need to deny consent doubles to 6.7 seconds for

websites that do not show a reject button. At the same time, the rate of affirmative consent increases from 83 percent to 90 percent. This shows that the design choices of websites greatly influence user interaction of privacy-aware users. We learn from this observation that user impedance and timing measurement is crucial to privacy measurement.

3.4 Landing Pages versus Internal Pages

This section discusses recent literature on the differences between landing pages and internal pages of websites. Landing pages refer to the root document of a website, internal pages refer to all other pages.

Aqeel et al. study the difference between landing and internal web pages with regard to their performance [Aqe+20]. They criticize that the vast majority of web measurements focus exclusively on the landing page and argue that the characteristics observed cannot be generalized to internal pages. With the help of a literature survey and replication study, the authors show that two thirds of the studies examining characteristics of a set of websites in a top list need to be revised if internal sites are also considered. In addition, the authors find that for the top 1000 websites, internal pages embedded a median of 18 more third-party domains than the corresponding landing pages.

Urban et al. have similar findings [Urb+20]: The authors show that studies which only focus on the landing page of a website miss a substantial amount of the website traffic to third parties. For example, they find that internal pages set 36 percent more cookies than landing pages across all types of websites. In a replication study, the authors are able to show that three out of the nine studies examined lead to different website behavior when a larger number of internal pages were taken into account. While the new observations did not invalidate previous findings, the replication study shows that the examination of landing pages alone fails to comprehensively identify relevant behavior and characteristics of a website.

Recent literature shows that landing pages exhibit significantly different properties compared to internal pages [Aqe+20; Urb+20]. This implies that observations made for landing pages cannot be generalized to internal pages and motivates further research on more complex website interaction. We conclude from these studies that internal pages are highly relevant to privacy measurement and argue that the proposed approach in this thesis facilitates a comprehensive this kind of analysis.

4 Semi-automated Interactive Website Scanning

In this chapter, we describe and discuss the proposed privacy measurement platform. The platform gives users as well as researchers the ability to comprehensively analyze tracking practices on websites. The requirements of the web application that allows users to interact with a remote browser and scan website's behavior are stated in Section 4.1. Next, Section 4.2 defines the activity of an interactive scan. We then answer the first and second research questions by examining the architecture and design features of the interactive web measurement platform in Section 4.3. The result model is presented in Section 4.4. Finally, we discuss the partly implemented record and replay feature in Section 4.5.

4.1 Requirements

The goal of the measurement platform is to provide the following core functionality:

- The user of the platform can interact with a website under investigation in a controlled way, without the need for prior installation of plugins or software.
- The interaction of the user and resulting activity in the browser is recorded and stored for later analysis.
- During the interaction with the website, the user can mark control points between which the recorded data is partitioned.
- Result scan data is made available to the user in an appropriate format.

Apart from the core functionalities listed here, further design requirements of the platform have been analyzed. These include capabilities like deleting cookies during a scan, taking screenshots of the current viewport, or recording an interaction and replaying it at a later point in time. These minor features and design decisions are discussed in Sections 4.3 and 4.5.

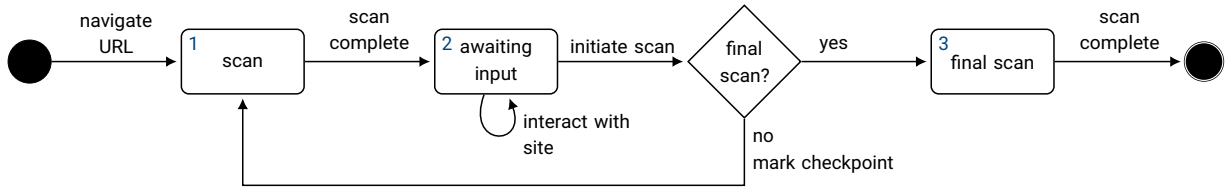


Figure 4.1: Activity diagram of an interactive website scan.

4.2 Interactive Scan

An interactive website scan consists of a number of steps that are performed by either the measurement platform or the user. The activity diagram depicted in Fig. 4.1 provides an overview of the high-level activities during a single website scan. It consists of the following steps which correspond closely with the states and transitions in the activity diagram:[1]

1: In the interest of conciseness, we exclude all error cases in this diagram.

1. The user enters the URL of a website to be investigated and starts the scan. In the background, a browser instance is initiated and made available to the user through a remote desktop connection. (⟶1)
2. The behavior of the website is recorded through the debug interface of the browser in an initial scan. A website scan captures the network traffic, cookies, and the JavaScript function call profile. User interaction is blocked during all scans. (1)
3. Upon scan completion, the user controls are available again. (1⟶2)
4. The user can now interact with the website. The website behavior as well as user inputs[2] are recorded in the background. At this point, the user can also take screenshots of the current viewport. (2⟶2)
5. The user can either mark a checkpoint by initiating an intermediate scan or they can finish the scan.
   (a) When the user decides to perform an intermediate scan, the browser behavior backlog and state is stored as a new object in the list of iterations. By means of the intermediate scan, the user can differentiate among individual actions while visiting the website. Alternatively to initiating a scan after an interaction, the user can delete all browser cookies, which also triggers a scan. (2⟶1)
   (b) Otherwise, the final scan is initiated. After the final scan, the browser instance is shut down and the scan results are stored. The user may now start another scan or analyze the results. (2⟶3, 3)

2: Recording and replaying website scans is discussed in Section 4.5.

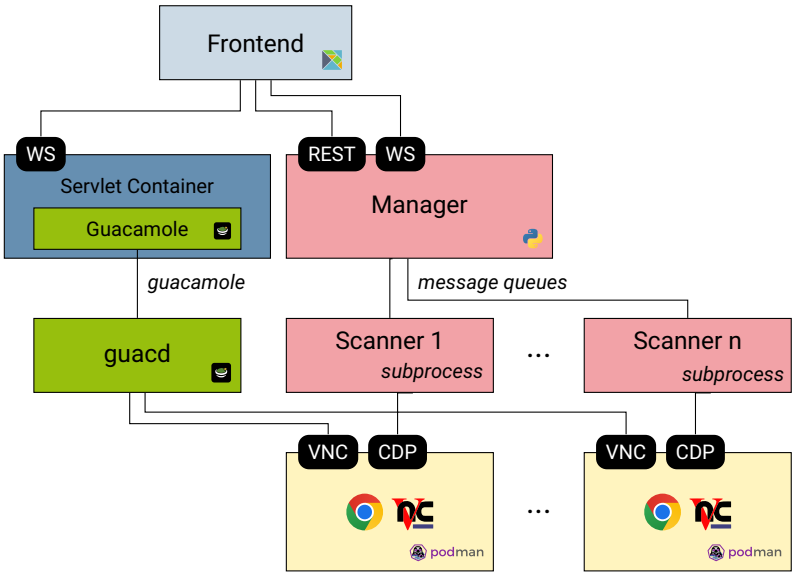


Figure 4.2: High level overview of the complete software architecture with communication channels. The communication channels use the protocol denoted at respective ports. Exceptions are guacamole and the message queues, which are stated verbatim.

4.3 Software Architecture

In this section, we present and discuss the software architecture of the research prototype. We first give an overview of the high-level architecture in Section 4.3.1.[3] From there, we describe the individual subsystems and explain design decisions in Sections 4.3.2 to 4.3.4. The Guacamole subsystem is not discussed explicitly, as it was described comprehensively in section 2.3. Section 4.3.5 provides relevant details with respect to deployment and maintenance.

3: The high-level description can also be read after the discussion of the individual subsystems, depending on the reader's preference.

4.3.1 High-Level Architecture

In this section, we describe the high-level software architecture of the research prototype. We sketch the roles of individual subsystems and describe the interaction between the interlinked components. An overview of the architecture is depicted in Fig. 4.2. The figure demonstrates the general case of *n* interactive scans running in parallel.

We will go through each subsystem, indicated by the different colors, starting from the bottom and moving upwards. For every active scan, a Podman container is run, each comprising a browser and a display server. The containers respectively provide two ports for the backend subsystems to connect to, namely VNC and a browser debugging interface. On the left side of Figure 4.2, the guacd service uses VNC to make the remote desktops available to the servlet container. On the right side, the manager subsystem connects to the browser interface through sub-processes, one per container. The frontend of the prototype is where the user accesses the website under investigation and controls the scan. Communication is done through three connections with the two backend subsystems: The

frontend has access to the remote desktop over a websocket connection to the servlet container. At the same time, the user can control the scan via the RESTful API of the manager. The frontend also establishes a websocket connection for bidirectional communication for scan status updates.

With the exception of the frontend, every subsystem is initiated as a systemd service. Systemd facilitates monitoring, logging and failure recovery. An additional systemd service not depicted in Figure 4.2 is the webserver `nginx`. We use nginx to combine the different web services into a single web application accessible under one domain. The webserver acts as a proxy for the Guacamole and manager backend subsystems. It is also responsible for serving the static HTML frontend and other static files.

4.3.2 Chrome Container

In the measurement platform, we make use of container technologies to run the browser instance in a controlled way. The decision to utilize container technologies is motivated primarily by the arguments made by Boettiger, who discusses their advantages and limitations in systems research with respect to reproducibility [Boe15]. The authors name, among others, the "dependency hell" and barriers to adoption and reuse as technical challenges of current approaches in systems research. They argue that container technology solves these issues with the help of an isolated and reproducible setup.

The three main design goals for the browser instance are

1. to enable fast and consistent startups and shutdowns,
2. isolation among multiple instances, and
3. a high level of reproducibility.

The first design goal aims to ensure a usability of the browser. The second design goal of isolation involves releasing resources after they have been used and not interfering with other instances during execution. With the third design goal of reproducibility, we want to increase trust in the measurement platform and the results. This also facilitates possible extensions of the approach. Reproducibility ensures that the result is consistent and is not influenced by the operating system, browser profile, or location. Thus, the user should not only be able to perform interactive scans themselves, but also host the scanner on their own machine with little effort. By doing so, the scanner can also act as a development platform for unreleased websites.

During development, we explored and implemented a number of OCI container-based approaches before deciding on a Podman container architecture. In the first approach, we implemented a Docker-based container instance. The approach was later rejected because Docker does not offer multiple system processes through, e. g., `systemd`. The increased complexity imposed by the Docker ecosystem posed further challenges. As an alternative approach, we implemented the browser as a systemd

Listing 4.1: Sketch of the Chrome container Dockerfile configuration.

```
FROM fedora
RUN dnf -y install chromium socat x11vnc xorg-x11-server-Xvfb
# ... < load config files >
USER chrome
RUN systemctl enable browser cdp_proxy vnc
EXPOSE 5900 9000
CMD [ "/sbin/init" ]
```

container using `systemd-nspawn` and `machinectl`.[4] This approach was also rejected because of the configuration complexity of provisioning, network setup, and slow startup times.

We then decided on using Podman as the container manager as it fulfilled all three design goals. As depicted in Figure 4.3, the Podman container instance mainly consists of two parts: One is the Chrome browser itself, which provides a debugging interface for programmatic access through the Chrome DevTools Protocol (CDP). The other is a remote desktop server that makes the browser available to the user through a frame buffer, exposing a VNC port. A sketch of the container image definition is shown in Listing 4.1. The image consists of the three systemd services `browser`, `cdp_proxy`, and `vnc`.[5] The browser service manages the Chrome instance that is started with a selection of initialization flags. Important flags include `enable-automation`, which disables functions not suitable for automation, `disable-background-networking` for reduced noise in network traffic, and `remote-debugging-port` to enable CDP. The reverse proxy service is necessary to make the Chrome API available outside the container, which would otherwise only listens on `localhost`. The VNC service publishes the virtual display of Chrome using the display server `Xvfb`.[6]

4: Documentation of systemd-nspawn and machinectl: https://www.freedesktop.org/software/systemd/man/ (archived at 27.06.22)

Figure 4.3: Remote Chrome Container with two ports `VNC` and `CDP`.

5: Running Containers as systemd Services: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux_atomic_host/7/html/managing_containers/running_containers_as_systemd_services_with_podman (archived on 17.06.22)

6: Xvfb docs: https://www.x.org/archive/X11R7.6/doc/man/man1/Xvfb.1.xhtml (archived on 27.06.22)

4.3.3 Backend

This section describes and discusses the backend implementation of the scanner prototype written in Python. Figure 4.4 presents an overview of its software architecture. It consists of a manager process and a set of sub-processes. Each sub-process is responsible for controlling the running container instances and performing the website scans.

Manager Process The manager process is responsible for performing provisioning tasks and establishing communication channels to the sub-processes. As depicted in Fig. 4.4, it provides both a REST and a websocket interface. The REST interface is made available to the frontend and provides the ability to initiate and control website scans. The websocket provides an asynchronous communication channel with the frontend, e. g., for issuing status updates and error messages.

We will go through the steps required for performing an interactive scan from the perspective of the manager. As a first step, the frontend registers itself over a websocket. Afterwards, the frontend may initiate a scan through the REST interface. Upon request of the frontend, which contains the URL of the website under investigation, the manager performs a

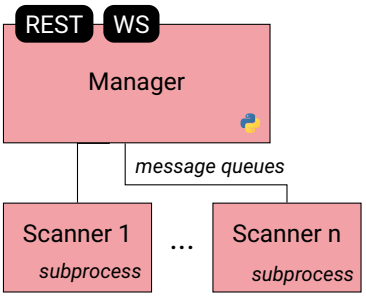


Figure 4.4: Manager process.

Listing 4.2: Function `start_scanner` with command message loop. Note that some details such as error handling have been omitted for conciceness.

```
async def start_scanner(self):
    async with Browser(self.debugging_port) as browser:
        while True:
            msg = await self.queue.get()
            try:
                rec_poison_pill = await self.process_message(msg)
                if rec_poison_pill:
                    break
    PODMAN_API.stop_container(self.container_id)
    self.send_socket_msg("ScanComplete")
```

series of actions: It creates and starts a new container instance. Additionally, it creates a sub-process which is configured with the website URL, the Chrome debugging port, and the client websocket connection. The websocket connection is used for status updates and error messages. Note that, in the interest of conciseness, Fig. 4.2 is not perfectly accurate as this connection from the frontend to each subsystem is omitted. Additionally, a command message queue is initialized for further communication with the scanner sub-process. This message queue is independent of the websocket and is used for a number of control commands, which will be discussed later. The sub-process is then started and the scan initialization message is put into the message queue. This process is detailed in the next paragraph. Subsequent requests to the REST interface are converted to command messages and put in the respective message queue. When the user decides to stop scanning, the sub-process instructs the container to shut down and then also terminates. The manager then stores the results of the scan and provides an interface for them for the frontend.

Chrome Handler Sub-Process The manager creates one handler sub-process per Chrome container. Each container is responsible for handling the communication with the browser, including data collection. Here, we will discuss the tasks of a sub-process.

Listing 4.2 shows the function `start_scanner` responsible for receiving messages in the message queue.[7] When the function is called, the sub-process connects to the debugging interface of Chrome using the assigned debugging port. It then runs a while loop reading the command messages from the queue. Received messages are passed to the `process_message` function, which handles the respective tasks. It returns a boolean poison pill indicating whether the command message loop should be terminated. The function expects one of the following messages:

7: In the interest of conciseness, we omitted some code details such as type conversions, error handling, and logging.

`StartScan` This command message is by design the first message placed in the command message queue. First, relevant background tasks are initiated, including setting up callbacks and starting the `JavaScript` profiler. The browser is then instructed to disable user inputs and navigate to the URL provided by the user. After the DOM content is loaded and the network is inactive for a certain time, the behavior of the browser is recorded in an initial scan. This includes network requests and responses, cookies, and the stack traces of JavaScript

recorded up to this point. Afterwards, the browser is instructed to accept user input again. A success message is sent to the frontend via the websocket. No poison pill is returned.[8]

8: Only the `StopScan` sends a `True` poison pill.

`RegisterInteraction` This message is sent when the user decides to mark a checkpoint and initiate a manual interaction. During all scans, user input is disabled. The behavior of the browser since the last scan is captured and stored.

`ClearCookies` This command instructs the browser to deletes all cookies. Similar to a manual scan, a new scan iteration is performed.

`TakeScreenshot` Upon receiving this command, the browser is instructed to takes a screenshot of the current viewport. No scan is performed here.

`PerformUserInteraction` This command message is special and not the result of a user's request. Instead, it is part of the record and replay feature discussed extensively in Section 4.5. The command handler replays a user interaction that was recorded in a previous scan. This includes searching for the DOM element using the given selector and performing the corresponding action.

`StopScan` A final scan is performed and appended to the list. The poison pill `True` is returned to the calling function.

When the poison pill is received as a result of processing the final message, the command message loop terminates. Subsequently, the sub-process shuts down the container and sends a status feedback to the frontend through the websocket. The websocket is also used to send error messages to the frontend and inform the user of failures during the scan.

Browser Instrumentation As mentioned, the Chrome browser offers a debugging API through the CDP.[9] While direct low-level websocket implementation of this protocol is possible, this entails practical limitations due to limited documentation[10] and error-prone implementation. Instead, it is recommended to choose an alternative from a range of high-level wrappers. These protocols differ in the capabilities and design goals such as debugging, testing, automation, or combinations thereof.[11] We therefore chose the high-level Playwright framework which combines the low level websocket functionality with a number of high level features. These include

- automatically waiting for DOM or network events,
- HTTP Archive (HAR) recording,
- DOM navigation and search, and
- JavaScript function evaluation.

As the low-level protocol is still available through Playwright's API, Playwright is a superset of CDP.

9: Chrome devtools protocol docs: https://chromedevtools.github.io/devtools-protocol/ (archived on 27.06.22)

10: Documentation of CDP: https://chromedevtools.github.io/devtools-protocol/tot/ (archived on 20.06.22)

11: Comprehensive list of browser APIs and tools: https://github.com/dhamaniasad/HeadlessBrowsers (archived on 27.06.22)

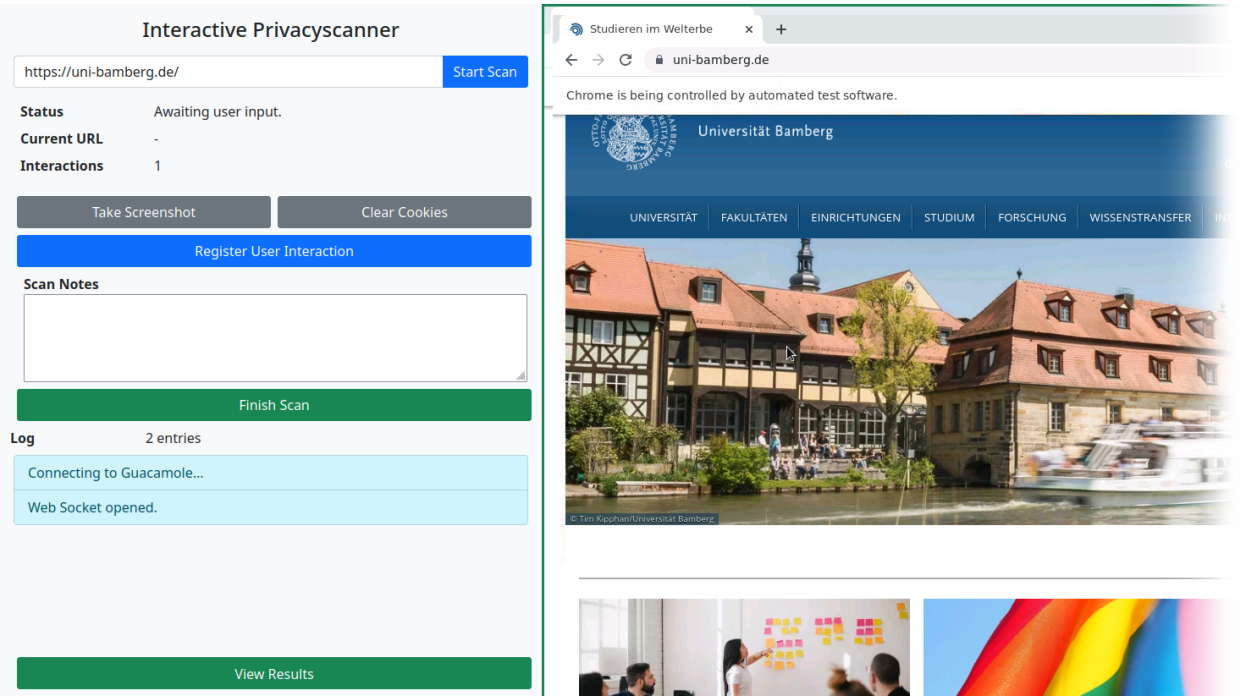


Figure 4.5: Frontend of the measurement platform during an interactive website scan of https://uni-bamberg.de/. The user monitors and controls the scan using the control panel on the left. They can interact with the remote browser on the right (cut off for conciseness).

4.3.4 Frontend

In this section, we present and discuss the frontend of the measurement platform, developed in `Elm` and JavaScript. Elm is a functional frontend language that compiles to HTML, JavaScript and CSS.[12] We chose Elm because it is a functional language with strict typing, making it free from runtime exceptions. This property facilitates rapid development and refactoring as the burden of testing and integration is greatly reduced. We also make use of Elm's interoperability with JavaScript when interfacing with Guacamole and the backend websocket. Here, we will abstract from the implementation of the prototype and instead only focus on the design decisions which lead to the resulting prototype.

12: Elm guide: https://guide.elm-lang.org/ (archived on 27.06.22)

Figure 4.5 shows a screenshot of the frontend view, which is comprised of two parts: The control panel is located on the left and features all control options for the scan as well as information about the current scan status. All actions performed on the control panel are communicated over the REST API of the manager backend. The user can start a scan by providing the URL of the website under investigation. The screenshot shown in Fig. 4.5 is the result of this action for the website of the University of Bamberg. Only the control options at the current state is highlighted.[13]

13: The 'Start Scan' button is normally disabled at this point and only made available for demonstration purposes.

After initialization of the browser instance and the remote desktop connection, the user can instruct the backend to perform various actions, corresponding to the messages described in the previous section. They can take a screenshot of the viewport, clear browser cookies, register a user interaction or finish the scan. Users can also add miscellaneous notes which are stored when the scan is finished. For instance, a note might

be added containing information about the appearance of the website that are not captured otherwise. Below the interface that enables control over the scan, a log informs the user about the scan status and possible errors. Log messages are created by the manager, respective sub-processes, and by Guacamole and sent to the frontend via websockets and differ by color based on severity. The log serves as a central place where events of different parts of the platform inform the user of changes and errors. The results page accessible at the bottom of the control panel allows the user to download and investigate scan results.

Because nginx serves as a proxy for the backend servers, the websocket communication is cached by the backend nginx service. As a result, establishing the connection sometimes takes up to 10 seconds.[14]

14: This is an implementation flaw. A fix of this problem is reserved for future work. Possible solution can be found in the discussion of caching delay problems on StackOverflow: https://serverfault.com/questions/796735/nginx-reverse-proxy-is-slow (archived on 08.09.22)

The access to the remote browser provided by Guacamole is located on the right side of Fig. 4.5. A border around the display indicates the three relevant states with regard to its focus and scan state. The border color is set to

- green if user input is awaited and the browser is in focus,
- blue if user input is awaited and the browser is not in focus, and
- yellow if a scan is in progress irrespective of focus.[15]

15: While we did consider splitting the 'scan in progress' state to capture all four possible states, we rejected the idea because of the small semantic difference between these two states. A user survey can conclude if this decision was justified.

The border was created to transparently communicate to the user which of the two nested browsers is in focus and whether or not interaction with the remote browser is possible.

4.3.5 Maintenance and Deployment

This section provides some details regarding the maintenance and deployment of the prototype implementation. The source code is published as a git repository on Github.[16] The repository contains all source documents which constitute the platform software architecture. The interested reader is referred to the readme file for the documentation on the installation of the platform. Additionally, the repository stores all scripts that were used in the analysis. These include scripts for website selection, computation of statistics, and for plotting results. A copy of the repository release 'submission', the raw result files and figures of the case study from Chapter 5, and a copy of the master's thesis report is stored in the supplementary material.

16: Interactive Privacyscanner: https://github.com/martin-endress/interactive_privacyscanner

Listing 4.3: Informal JSON Schema for the result format.

```
{ "url": <scan URL>,
  "interaction": [
  { "event": "initial" | "interaction" | "final" | "clear cookies",
    "cookies": <cookies>,
    "requests": <requests>,
    "responses": <responses>,
    "profile": <JS function calls>,
    "screenshots": <image paths>,
    "url": <current url>,
    "timestamp": <timestamp>,
    "user interaction": [{"event": e.g. "click", "element": <selector> }]
  }],
  "note": <user note>
}
```

4.4 Result Model

In this section we briefly summarize the result format of an interactive website scan. An informal scheme of the scan result JSON file is depicted in Listing 4.3.[17] We model the interaction of a scan as a list of interaction objects. Each object contains the event interaction that triggered the interaction as well as the data recorded during a scan. An interaction item is created during the initial scan, as a result of a user-initiated scan, as a final scan, or as a result of clearing the cookies. The recorded scan model associated to an interaction consists of cookies, requests, responses, the JavaScript function profile, and paths to recorded screenshots. In addition, the information about the recorded user interaction events are captured. The record and replay functionality is discussed in Section 4.5. For potential timing analyses, the timestamps of scan initialization is recorded. Finally, the current URL is saved. The manual user note and the starting URL are modeled at the top level of the JSON file.

17: While we are aware of the JSON schema standard, we chose to present the schema in a concise format.

Several other files are stored aside from our custom result JSON file. The measurement platform stores the recorded HTTP network traffic in a HAR file. This file is provided by the Playwright API and facilitates qualitative analysis methodologies. It is independent of our interaction model, meaning that the recorded data is not partitioned among the interactions. Manual partitioning of the HAR file using the timestamps might be possible, but is reserved for future work. The screenshots taken during the scan are also stored along side the result file.

4.5 Record Interaction

This section discusses the feature of recording and replaying interactive website scans. In essence, the feature includes recording the sequence of user inputs so that the interaction can be replayed at a later point in time, possibly using different settings. The capability extends the platform in a meaningful way and might facilitate further research methodologies. We see following three advantages of this capability: Primarily, the feature allows for longitudinal studies that investigate changes of web ecosystems and practices. The record and replay feature also facilitates more complex scan methodologies, e. g., using prepared user profiles or observations from different locations. Finally, we argue that a record and replay feature improves the reproducibility of any research methodology, as researchers and students can rerun scans on their own hardware.

We now describe and discuss the implementation of the record and replay feature. The approach is inspired by Playwright's test generator feature which offers automated interactive integration tests on websites.[18] The program records user actions and stores them as a list of instrumentation instructions with associated DOM element selectors.[19] The test generator also offers the capability to emulate different user scenarios including profiles, devices, or user location.

18: Playwright test generator documentation: https://playwright.dev/docs/codegen (archived on 25.08.22)

19: Note that Playwright uses the term locator. For consistency, we stick to the term 'selector'.

As the instructions and model are specific to the closed source Playwright project, we derive a custom methodology: In our approach, user actions are recorded by injected JavaScript code using DOM event handlers. They are modeled as tuples of actions with associated HTML elements. For testing purposes, we initially implement only the click event. We construct a selector for the HTML element associated with the event fired. For this, we use a simple transformation script that recursively goes through the parents of the selected element and stores the tag and index or the id of each element.[20] The index is omitted if it is the first element with the corresponding tag. A result selector XPath might look like the following:

20: How to generate unique CSS selector for DOM element? https://stackoverflow.com/a/49663134 (archived on 25.08.22)

```
html > body > div > div:nth-child(3) > #article-1337 > p
```

Instructions specific to the interactive scan such as taking a screenshot are already captured in the result model, namely the event.[21] Using the DOM event and selector tuple and the event information captured in the result model, the interaction can be replayed.

21: This is not the DOM event, see section 4.4.

When an interactive scan of a website is replayed, the operation is reversed and the sequence of recorded actions is performed in a new scan. We iterate over the sequence of actions and construct a list of commands for the Chrome handler. In case a user action is to be applied, the command handler is instructed to search for the DOM element using the selector and perform the corresponding action. This the event information is passed with the `PerformUserInteraction` command message. In all other cases, the normal command messages constructed and issued. In the event of a failure, e. g., when the element could not be located the scan is aborted and the error message is stored.

During testing of the record and replay feature, we observed that the approach has multiple flaws. Many of the replayed scans did not succeed and reported that no element was found for the given selector. We conclude that this was the result of two problems: First, the approach fails to perform actions inside iframes as they construct new documents with own runtimes. In privacy measurement, this flaw greatly limits the research methodologies as iframes are often used for consent management or third party tracking [SN18]. Second, the selector generation approach is not very robust to website changes. The only information used for generating the selector is the tag and order of elements, which can change quickly when new content is published.

Due to the complexity of the record and replay feature and the described shortcomings of the prototype, we did not conduct a comprehensive analysis or study. Possible solutions to these problems and applications are discussed in Section 6.3.

5 Case Study

In this chapter, we report on a case study that was conducted using the measurement platform prototype. The analysis serves as an example of the many use cases the platform can be applied to. It also answers the third research question by evaluating the tool's capability as a research tool. We define the methodology of the study and subsequently present and discuss its findings.

5.1 Methodology

In this section, we describe the analysis methodology of third-party tracking prevalence upon a cookie consent interaction. Using this methodology, we aim to quantify tracking practices before and after affirmative cookie notice consent. An affirmative cookie consent interaction means selecting and accepting all cookies in a consent notice. We chose an affirmative cookie consent as the type of interaction under investigation because of its high relevance and easy interpretation. We consider the consent interaction as relevant because of the requirements established by the GDPR and other legislation. The ability to interpret the interaction is given because clear preconditions are defined for it. As explained in Section 2.2, no personal information may be collected before consent has been given.

More specifically, the study analyzes

1. user impediment as a result of consent notices,
2. third-party tracking prevalence,
3. cookie syncing activity, and
4. fingerprinting presence.

While the first analysis is done independently of the consent interaction, the analysis items two to four consider the behavior before and after the affirmative consent interaction. The methodology for the second analysis has been proposed by Maaß et al. [Maa+17]. The methodology for the third and fourth analyses has been proposed by Papadogiannakis et al. [Pap+21].

5.1.1 Website Selection

Here, we describe the set of analyzed websites. We chose a random sample of 300 `.de` domains from the Tranco list.[1] We opted for a random sample instead of the top entries in order to examine both popular and infrequently visited websites and thus allow for a more comprehensive overview. This decision was motivated by the measurement study conducted by Bollinger et al., who argue that that a randomized approach provides more representative results [Bol+22].[2] The authors argue that this approach alleviates the risk of introducing a selection bias.

We utilize the Tranco list proposed by Pochat et al. for a number of reasons [Poc+19]. It combines several lists like Alexa into a single transparent list.[3] Advantages include high reproducibility and robustness against manipulation and rapid fluctuations.

Table 5.1 provides an overview of the number of websites analyzed. The sample of 300 `.de` domains was transformed to accessible URLs using a methodology proposed by Hils, Woods, and Böhme [HWB20]: We iterated over the list of domains and attempted four connections to URLs using the combinations of TLS and no TLS, and with and without the `www` subdomain. The URLs of the first successful connection was used for the analysis, respectively. If all four connections fail after a given timeout, the domain was filtered out. This was the case for 44 domains. Hils, Woods, and Böhme repeated this process several times over multiple weeks to ensure temporal robustness [HWB20]. However, we did not do so because of time constraints.[4] Out of the 256 remaining website URLs, 251 scans were performed and analyzed successfully. For the remaining five URLs, the measurements failed due to website crashes or timeouts.

Table 5.1: Summary of analyzed websites.

	Websites	
Description	*n*	%
initial set	300	100%
filtered not available	44	14.7%
filtered in analysis	5	1.7%
successfully scanned	251	83.7%

5.1.2 Data Collection

Measurements took place on July 08, 2022. We visited the 256 selected website URLs using the proposed privacy measurement platform and performed an affirmative cookie consent interaction if possible. The measurement platform recorded the website behavior in the background and subsequently stored the measurements. The average website scan took 8.3 seconds.

The activity diagram depicted in Fig. 5.1 describes the steps that were performed in one measurement. It was provided to six data collectors who helped with the measurement survey. The actions performed are summarized in the following: As a first step, the data collector inputs the URL of the website to be scanned. If a connection to the websocket is established, they start the scan. Some states in the diagram are specific to the measurement platform. For instance, the data collector has to wait until a websocket connection is established due to the caching problems discussed earlier. If an error occurs at this point, the measurement is restarted or aborted depending on its severity. As summarized in Table 5.1, severe errors on five websites resulted in failed scans. These errors are the result of browser security features and insufficient permissions.

1: Tranco List used in the study: https://tranco-list.eu/list/JX8KY/full (archived on 05.07.22)

2: The authors additionally use a logarithmic distribution to reflect Zipf's law with respect to website popularity.

3: Note that at the time of writing, the Alexa list is deprecated and will only be available through December 15, 2022. Pochat et al. continuously look for alternatives. Source: https://support.alexa.com/hc/en-us/articles/4410503838999 and https://tranco-list.eu/ (archived on 07.09.22)

4: Hils, Woods, and Böhme use the top 10 000 domains of the Tranco list.

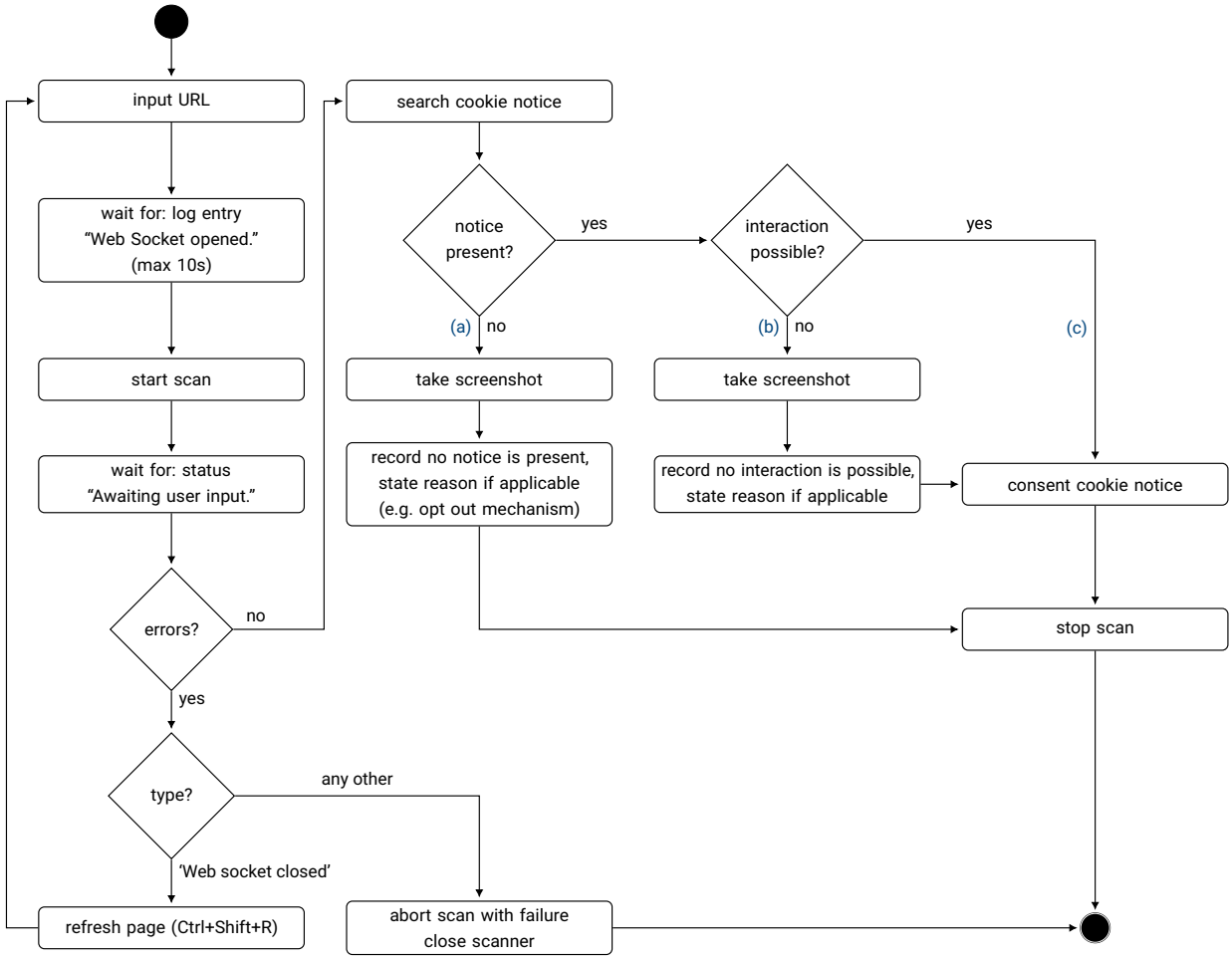


Figure 5.1: Activity Diagram of the study survey. The study participants controlling the measurement platform follow this diagram to generate measurement data. Consent notices are classified as (a) not present, (b) modal, and (c) non modal.

If no errors occurred, the consent notice is searched for. We classify the presence and type of consent notice in three categories: If a consent notice is (a) not present, we manually note its absence in the user note, take a screenshot of the website, and stop the scan. If a consent notice is (b) present and interaction is not possible, we state its presence, take a screenshot of the website, and proceed with the interaction. We classify as modal all consent notices that impede the user due to their size or modal property to prevent user interactions. If a consent notice is (c) present and does not impede user interaction, we classify it as such and also proceed with the interaction. We now consent to all cookies and possibly other website features listed in the consent notice. Subsequently, we terminate the scan.

5.1.3 Resource Use

Prior to data collection, a benchmark was performed to ensure sufficient resources for at least six parallel measurements. In the benchmark, we accessed popular and resource intensive websites such as web shops or media websites. We observed container memory usage of about 100 MB in idle and up to 700 MB on resource intensive websites and low CPU usage. As a result, we determined that one virtual machine with eight GB of memory is sufficient for six parallel scans. However, during data collection, the browser container instances experienced severe degradation due to insufficient memory. The browser slowed down and interaction with the website was severely limited. Upon inspection of resource use, we observed that more than twelve containers were running simultaneously at certain times. We assume that this is the result of multiple failed scans resulting in detached container instances. These detached containers are usually terminated by a watchdog process at a fixed interval, but continue to consume memory and CPU resources before the timeout. As a temporary solution, we reduced the watchdog timeout and repeated the measurement on affected websites. The detection of detached containers and automatic restart capability is reserved for future work.

5.1.4 Data Analysis

After data collection, the results were transformed into a single CSV file for subsequent analysis. Here we describe the construction of this file.

User Impediment Through Modals The classification of user impediment resulting from the consent notice is captured in the user note. The manual classification into three categories is derived from the note: If no cookie consent notice is present, we classify the website as 'no banner'. We differentiate between two cookie notice types. If the banner is perceived as an impediment to website use, either due to its size or being modal, we classify it as 'modal'. Otherwise, we classify it as a normal 'banner'. This classification is derived from the paths depicted in Fig. 5.1.

Third-Party Tracking For the second analysis on third-party tracking prevalence, we analyze the request log and examine third-party requests. We compare the domains of all third-party requests with the tracking detection rule sets provided by `EasyList`.[5] The EasyList rule set identifies trackers that are used for advertising purposes, whereas the `EasyPrivacy` rule set identifies trackers for other purposes. Here, we use the classification methodology proposed by Maaß et al. [Maa+17]:[6] If the hostname of the third party matches any rule of either the EasyList, EasyPrivacy, or Annoyance rule set, we classify the domain as a tracker. We use the `adblockeval` library to check if the hostnames match any of the rule sets.[7] The number of third-party cookies and requests to trackers is calculated for the period before and after the cookie consent interaction, respectively. We also capture these statistics from the perspective of the third party: For each third party and tracker, we count the number of websites that make at least one request to the server.

5: EasyList rule sets: https://easylist.to/ (archived on 06.08.22)

6: The EasyList and EasyPrivacy rule sets has been used in various other studies, e. g., by Englehardt and Narayanan [EN16].

7: Github page of adblockeval: https://github.com/hprid/adblockeval (archived on 06.08.22)

Third-Party Cookie Synchronization In the third analysis, we investigate first-party ID leaks and cases of cookie synchronization. The analysis uses the detection methodology proposed by Papadogiannakis et al. [Pap+21]. To perform the analysis, we convert the result file to the format used by the authors and run their analysis script[8], again partitioned into the periods before and after the cookie consent interaction. The analysis script searches for unique IDs in network requests and compares them with cookie values to detect possible ID leaks. An ID leak is considered to be present if an ID assigned to a user through a cookie value is sent to a different party in a network request. For this purpose, cookie values are compared with IDs in the path and parameter list for GET requests, in the body for POST requests, or in the referrer header, respectively. The detection script only takes into account strings that can be used as identifiers and filters out common keywords and consent information.[9] An ID leak is considered to be an event of cookie synchronization if the request domain differs from the first party. Note that this methodology does not precisely reflect the definition of ID synchronization from Section 2.1, as the detection approach does not consider a combination of multiple IDs set by different parties. However, the detection approach chosen by the authors still produces good results without adding too much computational complexity.

8: Gitlab Repository with analysis scripts: https://gitlab.com/papamano/consent-guard (archived on 08.09.22)

9: We refer to the original paper for a more detailed description of these filters [Pap+21].

Fingerprinting The fourth analysis focuses on the prevalence of device fingerprinting.[10] In this analysis, we abstract from the individual techniques and use the fact that all of them use client side JavaScript functions to determine a fingerprint of the device or browser. Papadogiannakis et al. noticed that recently proposed methods on fingerprinting detection which use JavaScript profiling analysis introduce unnecessary false positives in the results [Pap+21]. They argue that such approaches were not able to differentiate between genuine uses of browser functionality like the canvas element and malicious use for fingerprinting purposes. Therefore, the analysis proposed by the authors abstracts from low-level browser functions and follows a blacklist approach: Function names of the recorded JavaScript execution profile are compared with a dictionary of known fingerprinting library functions. The dictionary is constructed through profiling of popular fingerprinting libraries such as FingerprintJS. While this approach effectively alleviates false positive results, the authors acknowledge that it misses many minified or obfuscated fingerprinting scripts, resulting in false negative results.

10: We assume that the reader knows about device fingerprinting concepts. We refer to the study by Englehardt and Narayanan for a comprehensive analysis on the vast range of different fingerprinting techniques [EN16].

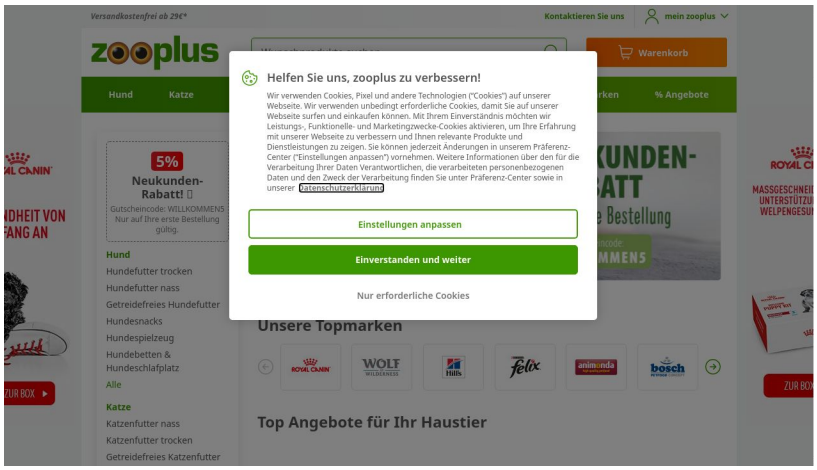


Figure 5.2: Screenshot of https://www.zooplus.de/. The cookie consent notice impedes the user from interacting with the website. User consent can therefore not be given freely as required by the GDPR [Eur16, p. 6].

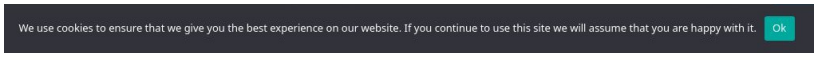


Figure 5.3: Screenshot of the consent notice of website https://www.zum.de/. The website uses an opt-out method.

5.2 Results

In this section, we report on the measurement results and analyze the behavior of websites before and after the interaction with the cookie consent notice.

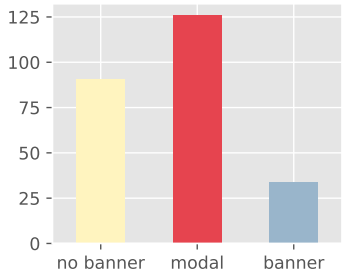
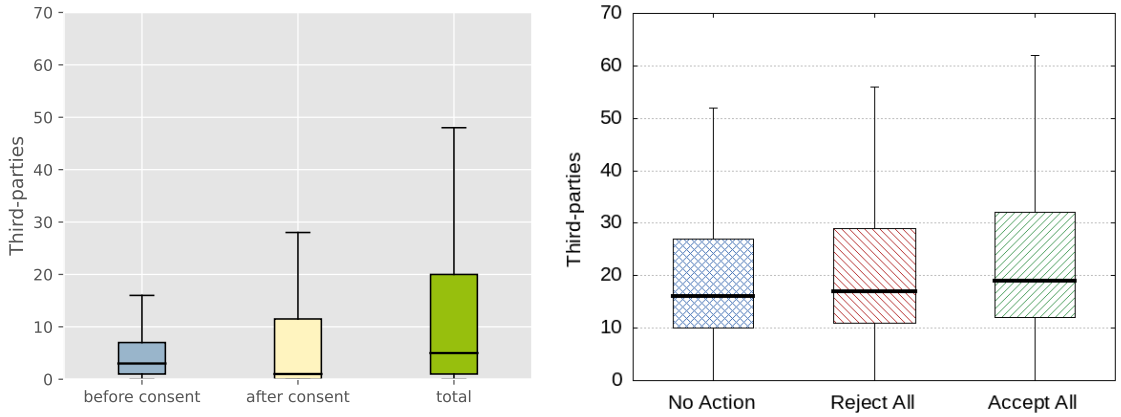


Figure 5.4: Presence and type of cookie banners. (n=251) A modal banner impedes user interaction with the website. The categories refer to observations a), b), and c) from Figure 5.1.

User Impediment Through Modals Figure 5.4 reports on user impediment through consent notices. Of the 160 websites that use a consent notice, 126, or 78.8 percent, display the notice in a way that impedes the user from interacting with the website. For this purpose, either a large part of the viewport is reserved for the consent notice or the element is displayed as a modal. This impedes or even prevents the usage of the website without interacting with the consent notice.

An example of a common modal consent notice is shown in Fig. 5.2. The website uses a modal which covers the majority of the screen, making interaction impossible. Recent literature states that these practices are not compliant with the GDPR [Bol+22]. Authors argue that user impediment through modals imply that any choice by the user is not freely, as required by Recital 32 [Eur16, p. 6]. A detailed legal analysis is beyond the scope of this study and can in the end only be made through court decisions. Another common potential GDPR violation is captured in Fig. 5.3: Here, the website does not ask the user for consent, but instead uses an opt-out mechanism.

Third-Party Tracking Next, we we report on the analysis on third party tracking during the consent interaction. We plot the number of unique third parties before and after the user interaction in Fig. 5.5a. The



(a) Results from this study. Plots 'before consent' and 'total' refer to 'No Action' and 'Accept All' as denoted by color.

(b) Results from original study by Papadogiannakis et al. The 'Reject All' action is not covered in our study [Pap+21].

Figure 5.5: Number of third parties for different types of website interactions.

methodology used here is taken from the study by Papadogiannakis et al., results of which are redrawn in Fig. 5.5b [Pap+21]. While we perform an action and measure the behavior before and after this action, the authors of the original study differentiate between the three cases 'No Action', 'Reject All', and 'Accept All'. The number of third parties before the consent interaction closely correspond to the 'No Action' case, whereas the total number of third parties correspond the the 'Accept All' case. This is also denoted by color. The third party requests recorded after the consent interaction are not captured in the original study, whereas we do not perform the 'Reject All' action. A two sided KS-test proves that the distributions of distinct third parties after consent significantly differ from the distribution before consent.[11] The number of third parties greatly increases as a result of the consent interaction.

11: KS-test: $D_{(before,\,after)} = 0.22; p < 10^{-5}$

Although a direct comparison of the results is not feasible as the study conducted here is not a replication, we nevertheless argue that the two approaches are comparable to some extent. The main difference between the two approaches is the selection of websites and test setup. We still can make interesting anecdotal observations that motivate further research. The large difference in the average number of third parties is of particular interest. We assume that it is mainly caused by two factors: The selection of websites in the original study is reduced to four percent of the accessible websites as CMP usage was required in the methodology.[12] Matte, Bielova, and Santos demonstrate that websites using CMPs exhibit highly different behavior with respect to third parties [MBS20]. Therefore, the 23-fold reduction in website selection solely as a result of the CMPs requirement leads to a selection bias, as CMP use is not an independent variable.

12: From 850 thousand websites studied, 628 thousand were pre-selected as available, and only 27 thousand ended up being used in the measurement.

The overall reduction in third parties can also be a result of the consolidation of the third-party ecosystem, as reported by Englehardt and Narayanan [EN16]. Note that further analysis is needed to fully determine the reasons resulting in the different results.

Table 5.2: Top six third parties classified as trackers embedded in websites. Additional trackers are only seen after consent, total refer to all trackers across the interaction.

#	before consent	after consent (additional)	after consent (total)
1)	googletagmanager.com	google.com	googletagmanager.com
	22.7%	23.5%	34.3%
2)	google-analytics.com	google-analytics.com	google-analytics.com
	10.4%	19.5%	29.9%
3)	**fonts.gstatic.com**	google.de	google.com
	10.0%	19.5%	28.3%
4)	app.usercentrics.eu	googleads.g.doubleclick.net	google.de
	9.2%	17.5%	22.3%
5)	fonts.googleapis.com	stats.g.doubleclick.net	googleads.g.doubleclick.net
	9.2%	17.5%	21.9%
6)	api.usercentrics.eu	adservice.google.com	stats.g.doubleclick.net
	9.2%	16.3%	21.5%

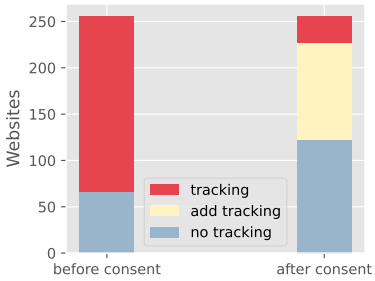


Figure 5.6: Number of websites that use trackers as detected by EasyList, partitioned in before and after the user consent interaction.

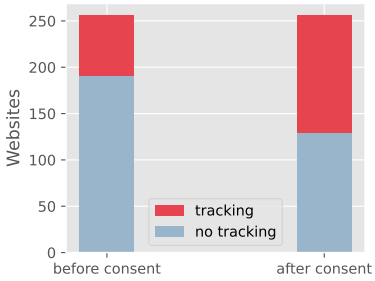


Figure 5.7: Number of websites that set at least one cookie by a tracker detected by EasyList, partitioned in before and after the user consent interaction.

Next we analyze which of these third parties are used as trackers. Using the methodology proposed by Maaß et al., we find that 191 out of 256 websites under investigation request at least one third party detected as a tracker before any interaction with the consent notice was performed [Maa+17]. The average number of distinct trackers contacted before any user interaction is 5.2. Figure 5.6 summarizes the results of the analysis. Websites that contact at least one unseen tracker as a result of the consent notice interaction are denoted as 'add tracking'. Interestingly, 56 websites that requested at least one tracker before the consent interaction made no further tracking related requests after the affirmative consent was given. This observation indicates that the user choice is not respected by these websites. 104 websites requested additional trackers after the consent was given.

We summarize the observations from the cookie tracking analysis. Figure 5.7 shows the number of websites that store at least cookie from a third party classified as a tracker. The average number of distinct third parties that store at least one cookie is 0.6 before the interaction and 5.1 after the interaction.

We now take a look at which third parties requested by websites. Table 5.2 lists the top six third parties that are classified as trackers before and after the consent interaction. In this kind of analysis, many set operations are possible and can reveal interesting information about the behavior of websites. For this analysis, we differentiate between trackers before the interaction, additional unseen third parties that were requested after the interaction, and all third parties seen throughout the interaction. While legal analysis can only be done by the courts as well as legal experts, recent court decisions allow some level of interpretation. For instance, a German court recently ruled that accessing Google's Fonts API without prior consent violates the GDPR.[13] The court based its decision on the fact that a request to the Google API constitutes unlawful data processing because the conditions established in Article 6 are not met [Eur16, p. 36]. This means that at least 10 percent of the websites studied are in violation of the GDPR because they request fonts before they have obtained consent.

13: LG München I, Endurteil vom 20.01.2022 - 3 O 17493/20: https://openjur.de/u/2384915.html (archived on 02.09.22)

Third Party Cookie Synchronization Next we analyze ID leaking activity using the methodology proposed by Papadogiannakis et al. [Pap+21]. Figure 5.8 summarizes the results of the study. It counts the number of websites that exhibit ID leaks, or even cookie synchronization events, partitioned into the periods before and after the cookie consent interaction. Unsurprisingly, the number of ID leaking activity increased as a result of the interaction. A detailed analysis on the specific third parties which are involved in the practice was not conducted.

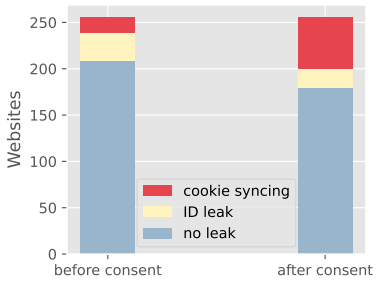


Figure 5.8: Number of websites that exhibit ID leak and cookie synchronization behavior, partitioned in before and after the user consent interaction.

Fingerprinting Finally, we describe the results of the device fingerprinting analysis. Using the classification methodology proposed by Papadogiannakis et al., we detected no fingerprinting activity both before and after the interaction [Pap+21]. We advise that this observation is to be taken with caution as the methodology has crucial limitations: As the detection methodology is based on the function names, minified or compressed JavaScript code cannot be detected in the analysis. Further, the analysis is based on a blacklist approach, which cannot capture novel techniques and functions.

In this section, we have demonstrated using a case study how our measurement platform can be used in for a number of research methodologies. Through adoption of previous study methodologies, we have demonstrated that our approach allows for easy replication and extension of previous studies [Maa+17; Pap+21].

6 Discussion

In this chapter, we discuss the proposed privacy measurement approach. We highlight its advantages over previous proposals and list possible analysis methodologies that are enabled through the approach. Furthermore, we list and discuss its limitations. We finally outline future work which is primarily motivated by the limitations.

6.1 Contribution

This section summarizes the high level advantages of our privacy measurement approach. Here, we list the key contributions and, where appropriate, compare them with previous methodologies from recent literature.

The privacy measurement approach is capable of modeling realistic user interactions. We have shown that, by means of the fully-fledged Chrome browser, the platform allows a privacy analysis of realistic and complex user interactions with any website. Compared to previous studies, there are fewer constraints on possible interactions which stem from the capabilities of instrumentation technologies. As demonstrated by Aqeel et al., many studies do not include internal pages in their analysis [Aqe+20]. We argue that our approach facilitates studies on internal pages which can be conducted complementary to automated methodologies that executes random interaction with websites. For instance, automated measurement can serve as a pre-study on websites which are then subject to a more comprehensive qualitative assessment using our platform.

The set of user interactions that can be modeled using the proposed platform is free from assumptions. Papadogiannakis et al. needed to filter the websites under investigation to enable controlled interaction with the consent notice [Pap+21]. In essence, the websites studied had to use one of several CMPs in order to automate the website interaction. In line with research by Matte, Bielova, and Santos [MBS20], we argue that this restriction of website selection to about four percent of selected and available websites leads to biases which our approach avoids.

Finally, we argue that our proposed privacy measurement platform has the potential to empower users by informing them about privacy invasive website practices. The idea follows the principles outlined by Barocas and Nissenbaum in helping to inform users about website practices and helping users make informed decisions [BN14].

6.2 Limitations

This section highlights the limitations of the privacy measurement approach as well as the case study. While some of the limitations are inherent to the approach itself and cannot be avoided, others were unaddressed challenges that were beyond the scope of this work.

The scope of privacy measurement studies using our semi-automated platform is very limited. The website scans need to be conducted manually and therefore do not scale well with the number of websites under investigation. While recent studies have measured up to a million websites, our approach realistically allows analysis of websites in the hundreds or few thousands. Careful attention must be paid to select a representative sample of websites. This process involves choosing a distribution to accurately reflect properties of the web like Zipf's law or pre-filtering the set of websites for interesting properties using automated tools.

The user experience of the measurement platform during a website scan fails to fully replicate the experience of a regular browser session. Apart from minor deviations like a missing window size adjustment or a lower resolution, the most severe problem with the user experience is the introduced latency when interacting with a website. This latency is primarily caused by network delays as well as the frame buffer of the display server. User experience degradation increases the effort required during studies and can even lead to human error. This can possibly negatively influence the measurement results.

The platform has not been tested against services which assess the uniqueness of the client in terms of device fingerprints. Theoretically, a website could behave differently when accessed by measurement tools. This shortcoming could be solved through appropriate browser profile setup and stealth functionality[1] provided by Playwright.

Some minor implementation issues were unaddressed or partially addressed as they were considered out of scope: One suggestion is a more transparent user feedback during website scans. This would improve usability but also make the platform more accessible for end users. The capability of signing the result data can also be an interesting addition. When the platform is hosted by an independent institution which signs result data, more trust can be established in the results because of the ensured data provenience. Further problems closely related to the implementation are listed as issues in the Github repository.[2]

1: Playwright Stealth: https://github.com/reanalytics-databoutique/webscraping-open-project/blob/main/Pages/Tools/Playwright_stealth.md (archived on 25.07.22)

2: Github issues of the prototype implementation: https://github.com/martin-endress/interactive_privacyscanner/issues (archived on 02.09.22)

6.3 Future Work

In this section, we discuss future work that would extend the measurement platform's capabilities, mitigate current limitations, or use the platform for relevant studies.

An interesting addition to the platform is the ability to measure websites using user profiles. Several privacy measurement studies have demonstrated that the geographic location, device properties, and the browser profile can have significant impact on the behavior of websites [Dab+19; Eij+21; Eng+15; Meh20; Wes+21]. Dabrowski et al. predict that the heterogeneity of website behavior observed from different locations will further increase due to additional privacy protection legislation initiatives [Dab+19]. While Eijk et al. have shown that websites often align their privacy practices with the jurisdictions of their expected audience, they also note that the question of how many measurement locations to use for comprehensive privacy measurement remains unsolved [Eij+21]. The authors advocate that all measurement studies should have at least one measurement each inside and outside the EU. In a cross-platform case study on the top 116 websites and respective Android apps, Mehrnezhad analyze the differences of tracking and cookie notice practices [Meh20]. Apart from finding numerous GDPR violations, she observes that the cookie notices among different platforms are inconsistent in terms of presentation and behavior. This observation motivates a differentiated approach which is capable of measuring different platforms including mobile clients and browser profiles. We imagine the platform being extended in such a way that an interaction is recorded once and then executed with the combinations of multiple locations, browser profiles, and time intervals.

The record and replay feature discussed in Section 4.5 can be an interesting topic for future work. The robustness of the recording functionality can be improved using the ROBULA+ methodology proposed by Leotta et al. [Leo+16]. The robust selectors proposed by the authors overcome test code fragility by using ad-hoc heuristics. Further approaches to be considered include the similarity based approach by Nass et al. which incorporates multiple selector parameters [Nas+22].

Future work should improve the user experience of the measurement platform caused by VNC. Other technologies such as an ssh X tunnel or a MITM proxy might be considered to improve latency. We argue that this makes the platform less error-prone and more friendly to end users, who will only use it if the effort is minimal.

In its current form, the measurement platform does not offer easy interpretation of the scan results. The offline privacy analysis is performed independently of the measurement using the downloaded raw data. Live feedback or direct analysis of key results would be helpful especially for end users who are interested in privacy practices of frequented websites.

We argue that the platform can be used for several relevant studies and be extended with other measurement techniques. Possible studies might investigate login and registration interactions, interactions with the privacy policy, or opt-out attempts. Interactions like these have received little attention in recent literature. The platform can also be extended to include detection methods for 'leaky forms' [Sen+22]. This practice includes form fields whose content is transmitted to third parties before submission, e. g., when the respective element loses its focus.

7 Conclusion

In this master's thesis, we presented a new approach for measuring privacy-related website behavior during user interactions, we implemented the approach in a reference prototype, and we examined the merits of the approach by analyzing a cookie consent interaction on 300 German websites.

The main motivation behind the project was the lack of studies that focus on internal pages and complex interactions with websites. We were also motivated to empower end users by making privacy practices more transparent to facilitate informed decisions.

The privacy measurement approach is designed as a web platform that allows users to control a remote Chrome browser. Users interact with the website under investigation, while at the same time the platform monitors and controls the browser. The behavior of the website is recorded in the background through the browser's debugging interface. During a website scan, users can mark interaction checkpoints between which the recorded website behavior is partitioned. The website behavior itself includes the network traffic, cookies, and the JavaScript function call profile. Additional notes and screenshots can capture relevant information that is not modeled by the platform itself. User actions on the website are also recorded. This feature has the potential to facilitate longitudinal analyses through repeated scans using different profiles.

After the website scan, offline analysis can be performed to link user interactions with website behavior. We demonstrated the merits of our approach in a case study that analyzed the affirmative consent interaction with the cookie notice on 300 German websites. The study provided showed a high level of user impediment through modals and high third-party tracking prevalence before the consent interaction. We also showed that our approach differs from previous studies in that it does not require assumptions or restrictions to be made in the selection of sites.

References

[ABL15] A Acquisti, L Brandimarte, and G Loewenstein. Privacy and Human Behavior in the Age of Information. In: *Science*:(2015) (page 2).

[Aqe+20] W Aqeel et al. On Landing and Internal Web Pages. In: *Proceedings of the ACM Internet Measurement Conference*. ACM, Oct. 2020 (pages 2, 14, 37).

[BN14] S Barocas and H Nissenbaum. Big Data's End Run around Procedural Privacy Protections. In: *Commun. ACM* 57(11):(2014), 31–33 (pages 2, 11, 37).

[Boe15] C Boettiger. An Introduction to Docker for Reproducible Research. In: *ACM SIGOPS Oper. Syst. Rev.* 49(1):(2015), 71–79 (page 18).

[Bol+22] D Bollinger et al. Automating Cookie Consent and GDPR Violation Detection. In: *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022. https://www.usenix.org/conference/usenixsecurity22/presentation/bollinger (pages 2, 9–11, 28, 32).

[BAL10] L Brandimarte, A Acquisti, and G Loewenstein. Misplaced Confidences: Privacy and the Control Paradox. In: *9th Annual Workshop on the Economics of Information Security, WEIS 2010, Harvard University, Cambridge, MA, USA, June 7-8, 2010*. 2010. http://weis2010.econinfosec.org/papers/session2/weis2010_brandimarte.pdf (page 2).

[Dab+19] A Dabrowski et al. Measuring Cookies and Web Privacy in a Post-GDPR World. In: *Passive and Active Measurement*. Springer International Publishing, 2019, pp. 258–270 (pages 12, 39).

[Eij+21] R van Eijk et al. The Impact of User Location on Cookie Notices (inside and Outside of the European Union). In: *CoRR* abs/2110.09832:(2021). arXiv: 2110.09832 (page 39).

[EN16] S Englehardt and A Narayanan. Online Tracking: A 1-million-site Measurement and Analysis. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*. Ed. by ER Weippl et al. ACM, 2016, pp. 1388–1401 (pages 1, 2, 7, 12, 30, 31, 33).

[Eng+15] S Englehardt et al. Cookies That Give You Away: The Surveillance Implications of Web Tracking. In: *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*. Ed. by A Gangemi, S Leonardi, and A Panconesi. ACM, 2015, pp. 289–299 (pages 1, 39).

[Eri19] A Erickson. Comparative Analysis of the Eu's Gdpr and Brazil's Lgpd: Enforcement Challenges with the Lgpd. In: *Brook. J. Int'l L.* 44:(2019), 859 (page 6).

[Eur02] C of European Union. Directive 2002/58/ec of the European Parliament and of the Council. https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX:32002L0058. 2002 (page 1).

[Eur09] C of European Union. Directive 2009/136/ec of the European Parliament and of the Council. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32009L0136. 2009 (page 1).

[Eur16] C of European Union. Regulation 2016/679 of the European Parliament and of the Council. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32016R0679. 2016 (pages 1, 6, 7, 32, 34).

[Fel+15] W Felter et al. An Updated Performance Comparison of Virtual Machines and Linux Containers. In: *2015 IEEE International Symposium on Performance Analysis of Systems and Software, ISPASS 2015, Philadelphia, PA, USA, March 29-31, 2015*. IEEE Computer Society, 2015, pp. 171–172 (page 8).

[Gra+18] CM Gray et al. The Dark (patterns) Side of UX Design. In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI 2018, Montreal, QC, Canada, April 21-26, 2018*. Ed. by RL Mandryk et al. ACM, 2018, p. 534 (page 10).

[GA05] R Gross and A Acquisti. Information Revelation and Privacy in Online Social Networks. In: *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society, WPES 2005, Alexandria,*

VA, USA, November 7, 2005. Ed. by V Atluri, SDC di Vimercati, and R Dingledine. ACM, 2005, pp. 71–80 (page 2).

[HWB20] M Hils, DW Woods, and R Böhme. Measuring the Emergence of Consent Management on the Web. In: *IMC '20: ACM Internet Measurement Conference, Virtual Event, USA, October 27-29, 2020*. ACM, 2020, pp. 317–332 (pages 9, 10, 13, 28).

[JPJ05] C Jensen, C Potts, and CS Jensen. Privacy Practices of Internet Users: Self-reports Versus Observed Behavior. In: *Int. J. Hum. Comput. Stud.* 63(1-2):(2005), 203–227 (page 2).

[Jus19] C of Justice of the European Union. Press Release No 125/19: Storing Cookies Requires Internet Users' Active Consent. https://curia.europa.eu/jcms/upload/docs/application/pdf/2019-10/cp190125en.pdf. 2019 (page 7).

[KPW21] M Kretschmer, J Pennekamp, and K Wehrle. Cookie Banners and Privacy Policies: Measuring the Impact of the GDPR on the Web. In: *ACM Trans. Web* 15(4):(2021), 20:1–20:42 (page 9).

[Kre+20] F Kreuter et al. Collecting Survey and Smartphone Sensor Data with an App: Opportunities and Challenges around Privacy and Informed Consent. In: *Soc Sci Comput Rev* 38(5):(2020), 533–549 (pages 2, 11).

[KW09] B Krishnamurthy and CE Wills. Privacy Diffusion on the Web: A Longitudinal Perspective. In: *Proceedings of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain, April 20-24, 2009*. Ed. by J Quemada et al. ACM, 2009, pp. 541–550 (page 1).

[Leo+16] M Leotta et al. Robula+: An Algorithm for Generating Robust Xpath Locators for Web Testing. In: *J. Softw. Evol. Process.* 28(3):(2016), 177–204 (page 39).

[Maa+17] M Maaß et al. Privacyscore: Improving Privacy and Security Via Crowd-sourced Benchmarks of Websites. In: *Privacy Technologies and Policy - 5th Annual Privacy Forum, APF 2017, Vienna, Austria, June 7-8, 2017, Revised Selected Papers*. Ed. by E Schweighofer et al. Vol. 10518. Lecture Notes in Computer Science. Springer, 2017, pp. 178–191 (pages 2, 13, 27, 30, 34, 35).

[Mat+19] A Mathur et al. Dark Patterns at Scale: Findings from a Crawl of 11k Shopping Websites. In: *Proc. ACM Hum. Comput. Interact.* 3(CSCW):(2019) (pages 2, 10).

[MBS20] C Matte, N Bielova, and C Santos. Do Cookie Banners Respect My Choice? : Measuring Legal Compliance of Banners from IAB Europe's Transparency and Consent Framework. In: *2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020*. IEEE, 2020, pp. 791–809 (pages 9, 13, 33, 37).

[Meh20] M Mehrnezhad. A Cross-platform Evaluation of Privacy Notices and Tracking Practices. In: *IEEE European Symposium on Security and Privacy Workshops, EuroS&P Workshops 2020, Genoa, Italy, September 7-11, 2020*. IEEE, 2020, pp. 97–106 (pages 9, 39).

[Nas+22] M Nass et al. Similarity-based Web Element Localization for Robust Test Automation. In: *CoRR* abs/2208.00677:(2022). arXiv: 2208.00677 (page 39).

[Nis11] H Nissenbaum. A Contextual Approach to Privacy Online. In: *Daedalus* 140(4):(2011), 32–48 (pages 2, 11).

[OTC14] L Olejnik, M Tran, and C Castelluccia. Selling off User Privacy at Auction. In: *21st Annual Network and Distributed System Security Symposium, NDSS 2014, San Diego, California, USA, February 23-26, 2014*. The Internet Society, 2014. https://www.ndss-symposium.org/ndss2014/selling-privacy-auction (page 5).

[Pap+21] E Papadogiannakis et al. User Tracking in the Post-cookie Era: How Websites Bypass GDPR Consent to Track Users. In: *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*. Ed. by J Leskovec et al. ACM, 2021, pp. 2130–2141 (pages 5, 12, 27, 31, 33, 35, 37).

[PKM19] P Papadopoulos, N Kourtellis, and EP Markatos. Cookie Synchronization: Everything You Always Wanted to Know but Were Afraid to Ask. In: *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*. Ed. by L Liu et al. ACM, 2019, pp. 1432–1442 (page 5).

[Poc+19] VL Pochat et al. Tranco: A Research-oriented Top Sites Ranking Hardened against Manipulation. In: *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society, 2019. https://www.ndss-symposium.org/ndss-paper/tranco-a-research-oriented-top-sites-ranking-hardened-against-manipulation/ (page 28).

[RT20] J Ryan and A Toner. Europe's Governments Are Failing the Gdpr: Brave's 2020 Report on the Enforcement Capacity of Data Protection Authorities. https://brave.com/static-assets/files/Brave-2020-DPA-Report.pdf (archived on 20.06.22). 2020 (page 10).

[Sen+22] A Senol et al. Leaky Forms: A Study of Email and Password Exfiltration before Form Submission. In: *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022. https://www.usenix.org/conference/usenixsecurity22/presentation/senol (page 40).

[Sol+07] S Soltesz et al. Container-based Operating System Virtualization: A Scalable, High-performance Alternative to Hypervisors. In: *Proceedings of the 2007 EuroSys Conference, Lisbon, Portugal, March 21-23, 2007*. Ed. by P Ferreira, TR Gross, and L Veiga. ACM, 2007, pp. 275–287 (page 8).

[SK19] JK Sørensen and S Kosta. Before and After GDPR: the Changes in Third Party Presence at Public and Private European Websites. In: *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*. Ed. by L Liu et al. ACM, 2019, pp. 1590–1600 (page 2).

[SN18] O Starov and N Nikiforakis. Privacymeter: Designing and Developing a Privacy-preserving Browser Extension. In: *Engineering Secure Software and Systems - 10th International Symposium, ESSoS 2018, Paris, France, June 26-27, 2018, Proceedings*. Ed. by M Payer, A Rashid, and JM Such. Vol. 10953. Lecture Notes in Computer Science. Springer, 2018, pp. 77–95. https://doi.org/10.1007/978-3-319-94496-8%5C_6 (page 26).

[Tre+19] M Trevisan et al. 4 Years of EU Cookie Law: Results and Lessons Learned. In: *Proc. Priv. Enhancing Technol.* 2019(2):(2019), 126–145 (page 1).

[Urb+18] T Urban et al. The Unwanted Sharing Economy: An Analysis of Cookie Syncing and User Transparency under GDPR. In: *CoRR* abs/1811.08660:(2018). arXiv: 1811.08660 (page 2).

[Urb+20] T Urban et al. Beyond the Front Page:measuring Third Party Dynamics in the Field. In: *Proceedings of The Web Conference 2020*. ACM, Apr. 2020 (pages 3, 14).

[Wes+21] V Wesselkamp et al. In-depth Technical and Legal Analysis of Tracking on Health Related Websites with ERNIE Extension. In: *WPES '21: Proceedings of the 20th Workshop on Workshop on Privacy in the Electronic Society, Virtual Event, Korea, 15 November 2021*. ACM, 2021, pp. 151–166 (page 39).

[WA21] J Wolff and N Atallah. Early Gdpr Penalties: Analysis of Implementation and Fines through May 2020. In: *Journal of Information Policy* 11:(2021), 63–103 (page 6).

Declaration of Authorship

Ich erkläre hiermit gemäß § 9 Abs. 12 APO, dass ich die vorstehende Masterarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Des Weiteren erkläre ich, dass die digitale Fassung der gedruckten Ausfertigung der Masterarbeit ausnahmslos in Inhalt und Wortlaut entspricht und zur Kenntnis genommen wurde, dass diese digitale Fassung einer durch Software unterstützten, anonymisierten Prüfung auf Plagiate unterzogen werden kann.

Bamberg, den _____ _____

Martin Endreß