CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Cookie Paywall Discrepancies

Assessing differences between operating systems, web browsers and geographic locations

Master's thesis in Computer science and engineering

Andreas Stenwreth
Simon Täng

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's thesis 2024

Cookie Paywall Discrepancies

Assessing differences between operating systems, web browsers and geographic locations

Andreas Stenwreth
Simon Täng

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

Cookie Paywall Discrepancies
Andreas Stenwreth
Simon Täng

Supervisor: Victor Morel, Computer Science and Engineering
Examiner: Gerardo Schneider, Computer Science and Engineering

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2024

Abstract

The cookie paywall allows visitors to access the content of a website only after making a choice between paying a fee or accepting tracking. The practice has been studied in previous research in regard to its prevalence and legal standing, but the effects of the client's device and geographic location remain unexplored. To address these questions, this study explores the effects of the client's browser, device type (desktop or mobile), and geographic location on the presence and behaviour of cookie paywalls and the handling of users' data. Using an automatic crawler on a dataset with 804 websites that present a cookie paywall, we observed that the presence of a cookie paywall was most affected by the geographic location of the user. We further showed that both the behaviour of a cookie paywall and the processing of user data are affected by all three factors, but no patterns of significance could be found. Finally, an additional type of paywall was discovered to be used on approximately 11% of the studied websites, namely the "double paywall", which consists of a cookie paywall that is complemented by a hard or soft paywall once tracking is accepted.

Keywords: cookie paywall, cookie, paywall, pay-or-okay, pay-or-tracking, accept-or-pay, double paywall

Acknowledgements

We want to thank Victor Morel, our supervisor, for providing the ideas for this project, pushing us to do better and providing insightful advice along the way. We would also like to extend our gratitude to Gerardo Schneider for providing thoughtful review and valuable suggestions that have raised the quality of our work.

Andreas Stenwreth & Simon Täng, Gothenburg, 2024-09-02

Contents

List of Figures

List of Tables

1

Introduction

When a user browses a website, their behaviour and personal choices are often recorded through the use of cookies [9]. A common use case is to track what websites users visit to present them with targeted advertising, which in turn generates revenue. Collecting data for the sake of targeted advertising requires consent in the EU as regulated by the General Data Protection Regulation (GDPR) and the ePrivacy Directive (ePD) [27]. Therefore, several methods have been introduced to generate revenue through blocking the website content either until the user consents to tracking cookies or until a payment is made.

The common denominator of these methods is that they "wall off" the user from the content until revenue can be created from said user. Some of these methods are the *cookie wall*, which requires acceptance of cookies and trackers before accessing the website, the *hard paywall*, in which users must pay a fee in order to gain access to content on a site, and the *soft paywall*, that allows limited, free-of-charge viewing before payment is expected [33]. Combining a cookie wall and a hard paywall, the cookie paywall is a novel type of paywall that has recently gained interest in academia [33, 32, 52, 44].

The cookie paywall, sometimes denoted "pay-or-tracking wall", "accept-or-pay cookie banner" or "pay-or-okay banner", allows visitors to access the content of a website only after making a choice between paying a fee or accepting tracking [33]. Thus, this practice widens the choice of payment method by making it possible to either pay through a subscription or to "pay with personal data". An example of such a paywall can be found in Figure 1.1.

Even though cookie paywalls have triggered interest from academia and regulators working on privacy, the technical aspects remain understudied. Notably, the studies that have been performed on the technical aspects of cookie paywalls have been limited in scope in regard to the plurality of browsers and environments (e.g. desktop or mobile) [33, 32, 52]. Nevertheless, discrepancies in the behaviour have been observed between browsers on some websites [33], raising the possibility that more websites use different practices depending on the browser or device used to access them. Performing a large-scale factorial experiment on the websites previously found to be using cookie paywalls could therefore provide valuable insights into how different devices may affect a user's experience and privacy online.

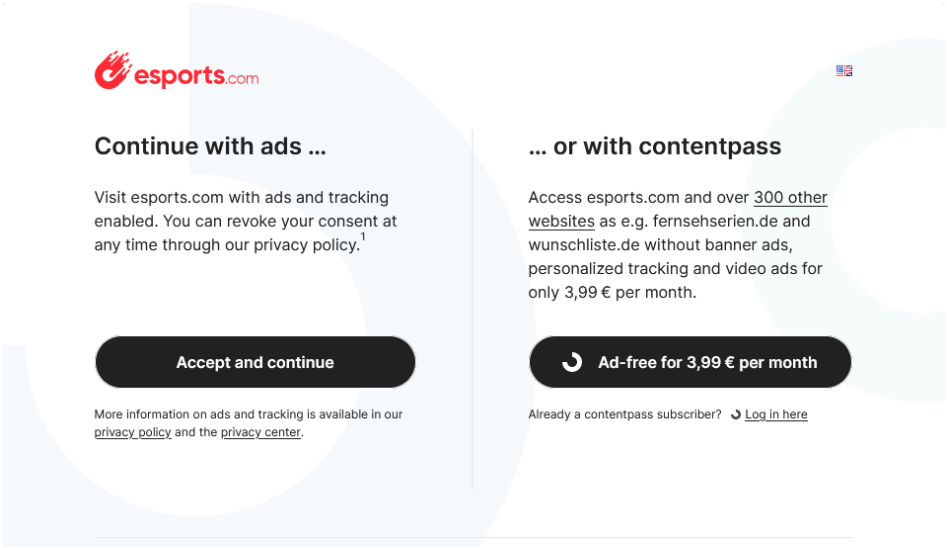

Figure 1.1: *A cookie paywall found at* `https://www.esports.com/en`.

Furthermore, during the exploratory stage of this thesis, an additional type of paywall was discovered. This "double paywall" initially presents a cookie paywall, which is later complemented by a soft or hard paywall after cookies have been accepted. The second paywall is usually only presented on certain parts of a website, such as a "subscription only" article, thus restricting access to these parts. Documenting the reach of double paywalls may give more insight into this newly discovered practice.

1.1 Objective & Research Questions

To address the previously identified research gap regarding cookie paywalls, the aim of this thesis is to survey cookie paywalls for discrepancies resulting from the choice of browser, type of device (desktop and mobile) and geographic location. More specifically, in what ways do these factors impact the presence and behaviour of cookie paywalls and the handling of users' data? Furthermore, this thesis aims to study the prevalence of the new practice of double paywalls. These issues are addressed through the following research questions:

RQ1. How is the prevalence of cookie paywalls affected by the browser, device type and geographic location of the user?

RQ2. Do the browser, device type and geographic location affect how much of a website can be accessed before a cookie paywall appears?

RQ3. Do the browser, device type and geographic location affect how and by whom user data is processed?

RQ4. How widespread is the occurrence of double paywalls?

Research questions 1-3 are addressed through a factorial experiment with the factors: desktop and mobile devices (desktop running Windows, Mac OS or Linux and mobile devices running Android or iOS), the top 4 most used web browsers (Google Chrome, Safari, Microsoft Edge and Mozilla Firefox) [63] and different geographical

locations (inside/outside the EU). This experiment is done using a dataset, described in Chapter 3, consisting of websites previously found to be using cookie paywalls. Research question 4 is explored using the same dataset as research questions 1-3, but only using the Chrome browser on Linux from inside the EU.

1.2 Scope

The resources allotted for this thesis require that the scope be set appropriately in terms of the number of factors to be examined. What this entails is a limited set of browsers, operating systems, and geographic locations. By restricting the number of factors used in the experiment, the expected time needed for data collection and the complexity of the following analysis are reduced.

The operating systems and web browsers considered are those presented in Section 1.1. Though not exhaustive, these operating systems and web browsers cover a large share of the mobile and non-mobile market [63, 64]. Furthermore, due to the plurality of browsers considered in the study, browser specific evasion techniques to avoid bot detection are not explored.

To examine if the geographic location (specifically being inside the EU) has an impact on whether a website may present a cookie paywall or not, one EU and one non-EU geographic location is used, namely Sweden and the USA. Sweden is chosen as the EU location since it is the physical location of the study. The USA, more specifically New York state, is chosen due to the use of the English language.

Finally, this thesis does not directly examine the compliance of cookie paywalls with regard to the GDPR and the ePD. The goal of this thesis is to examine the technical aspects of cookie paywalls and their implementations; therefore, the legality of such implementations is out of scope for this thesis.

1.3 Report Outline

Chapter 2 of this report proceeds to explain the technical and legal background needed for this project and describes some of the concepts and tools used in the project as well as the legal landscape in the EU. Chapter 2 also covers previous studies in the area of cookie banners and cookie paywalls. The crawler program developed for this thesis is presented in Chapter 3 and the results from the data collected using this program are presented in Chapter 4. A discussion of these results can be found in Chapter 5, followed by a conclusion of the thesis in Chapter 6.

2

Background and Related Work

Cookie paywalls can be viewed both from a technical and legal standpoint. From a technical perspective, it is important to understand the underlying web technologies and the tools needed for large scale analysis of cookie paywalls. From a legal perspective, any cookie paywall found on a European website falls under European privacy regulations as the aim of a cookie paywall is to collect personal data for processing. Some of the technical terms and concepts are presented in Section 2.1, Section 2.2, Section 2.3 and Section 2.4. The legal background and the technical frameworks used to facilitate the work to ensure compliance with European privacy regulations are covered in Section 2.5 and Section 2.6. Finally, Section 2.7 presents previous research in the area of paywalls, cookie banners and cookie paywalls.

2.1 Web Technologies

When connecting to a web server, there is an exchange of information between the web server and the web browser using the Hypertext Transfer Protocol (HTTP) [38]. HTTP follows a client-server model where a client, in this case the web browser, sends an HTTP request to the web server, expecting an HTTP response.

Along with every HTTP request made to a server, browsers include a self-identifying User-Agent HTTP header known as a user agent string [40]. A user agent is a program that represents a person. This can be a browser, a bot, a download manager, or some other app accessing the web. The user agent string lets the server identify the application, operating system, vendor and version of the requesting user agent.

A web server responds to a HTTP request with a HTTP response containing a status code, a webpage and optionally creates and includes cookies [9]. HTTP is a stateless protocol and, as such, does not retain any information about a user's session. Cookies can, however, preserve the state and can therefore be used to associate website activity with a user, personalise the user experience, and track what websites a user visits.

The webpage sent from the web server to the web browser is constructed using three core languages: HTML, CSS and JavaScript [37]. HTML is the code used to structure the webpage and its contents, CSS is used to style web contents and JavaScript is used to add interactivity. To allow programs to change the document structure, style and content, the webpage is represented by the Document Object

Model (DOM), which is loaded when accessing the page in a web browser [36].

The DOM represents the webpage as a node tree, where each node represents a part of the page [36]. The nodes are primarily created by HTML, in which case they are denoted as elements, but some nodes can be created using CSS and are then referred to as pseudo-elements [39]. The DOM representation allows the webpage to be manipulated and modified with a scripting language, such as JavaScript. The DOM defines the structure of elements and pseudo-elements, as well as properties and events for all HTML elements [54]. It also defines methods for accessing them. However, some elements create encapsulated environments which limits the interaction between the regular DOM and the internals of these elements. Two such elements are the inline frame (iframe) and the shadow DOM.

An iframe is an HTML element that embeds another full HTML document in the current document [34]. Because an iframe embeds an independent HTML document that has its own browsing context, it is isolated from the JavaScript and CSS of its parent object in the DOM. Some use cases of iframes include presenting cookie banners, embedding videos and displaying advertisements.

Shadow DOM is a functionality that lets you attach a DOM tree to an element in the regular DOM tree [41]. The internals of the attached tree are hidden from JavaScript and CSS running on the webpage and none of the internal elements can affect anything outside the shadow DOM. Thus, internal elements are not directly accessible using Javascript calls or element selectors [28].

2.2 Web Tracking

Web tracking is the act of collecting, storing, and sharing data about uniquely identifiable users on websites. Several web tracking techniques are available and can be divided into stateful and stateless techniques [31]. Stateful tracking stores the data required for user identification on the client side [55], most commonly as a cookie or as an entry in the web browser's local or session storage. Entries in local storage do not expire, whereas entries in session storage are cleared when the page session ends [42]. With stateless tracking, data used for identification consists of information collected about the user's browser, OS and hardware [5]. One technique for stateless tracking is through browser fingerprinting, in which HTTP header data, JavaScript and available APIs are leveraged to fingerprint a user's device [29].

2.2.1 Stateful Web Tracking

Cookies may be used to store data for web tracking purposes [9]. Cookies are small files generated by a web server and sent to a web browser. Web browsers store the cookies they receive and attach the relevant cookies to any future request that is made to the web server. Commonly, cookies are classified as either session cookies or persistent cookies depending on the lifetime of the cookie. Session cookies are deleted after a user ends their session, for example by signing out of their account, exiting the website or closing the web browser. Persistent cookies remain in a user's

web browser for a predetermined length of time. This time could range from seconds to years.

Cookies are further categorized as either first or third-party cookies depending on the domain the cookie belongs to and the domain in which the cookie is delivered to the user [6]. First-party cookies are placed by the domain that is directly visited. The information in this type of cookie is only available for the domain that created the cookie, and it is often used to keep track of the state of a user on a website, for example, for shopping cart persistence and remembering login details.

Third-party cookies are placed by other domains than the one currently visited. This is done through third-party code embedded in the visited domain, which allows for cross-site data transfer. Third-party cookies can be accessed on any website that loads the third-party code and are often used for data brokerage[1], tracking and advertising.

When used for tracking, cookies are placed in the user's browser to collect data about their online activities [1]. This data includes geographic location, device specifications and actions taken on a website. To differentiate users from each other, a unique identifier is assigned to each user by the cookie. Thus, a profile can be created for each user, containing information about their interests based on their actions on the internet. When visiting a website that carries the script of the domain that created the cookie, the collected information is sent to the origin domain [9].

2.2.2 Stateless Web Tracking

Browser fingerprinting, also simply known as fingerprinting, is a set of techniques used for tracking and identifying devices without requiring the device to locally store information [67]. Instead, only the unique characteristics of the device are used, such as the hardware, the OS, and the browser and its configuration [29]. This unique fingerprint, when connected to a user, can then be used to track a user's actions on the web. Fingerprinting can be achieved through passive and active techniques.

A website performing passive fingerprinting collects the data that is available in the web requests sent by the user's device, including data such as the HTTP header and the IP header [67]. Some of the important data contained in the HTTP header include the user agent string, content encoding and content language. The IP header contains the IP address of the device. IP addresses are assigned to organisations which are associated with geographic locations; thus the geographic location of the IP address can be determined through an IP-to-location database [68].

Active fingerprinting complements the passive collection of device characteristics by using JavaScript and potentially other available APIs [29]. The use of JavaScript enables the collection of information such as screen resolution, time zone, platform, the list of supported fonts and the list of installed plugins.

[1]The business of collecting and selling personal data, or data about people, as a commodity.

2.3 Device Impersonation Techniques

Both HTTP headers and responses to JavaScript commands can be altered to imitate other devices with different characteristics [29]. This can be done both with malicious intent, such as avoiding bot detection, and for research purposes, such as device emulation. Actively altering the fingerprint of a device through the HTTP header or the responses to JavaScript commands does, however, pose the risk of providing contradictory or inconsistent values [29]. Such inconsistencies may itself be a unique identifier that can be used for tracking. It may additionally be detected by websites or other service providers and raise red flags, potentially blocking access to the website.

Similar to altering a device's responses, a device can also alter its IP address as well as its apparent geographic location by connecting to the Internet using a Virtual Private Network (VPN) [10]. This is done by creating a secure "tunnel" between the device and a VPN provider located at another geographical location, for example outside of the EU. The user then shares the IP address with other users connected to the same tunnel endpoint.

The use of a VPN may grant access to content and websites not available from the geographic location of the user, but it can also lead to refused access to certain websites [10]. If the reputation of the IP address of the VPN endpoint is poor, a website can block access based on a list of untrustworthy IP addresses [25].

2.4 Web Crawling

A web bot is a software program that operates on the Internet and performs repetitive tasks [12]. One use case of web bots is web crawling, that is, automatically accessing websites and obtaining data [11]. Such a bot is commonly referred to as a web crawler and is most often used by search engines to index the web. Typically, a web crawler starts with a set of URLs and adds to its list the hyperlinks found at the visited websites; alternatively, the whole set of URLs can be provided from the start.

Crawling activities can be regulated on a website by deploying the Robots Exclusion Protocol [56] in a file called robots.txt. This file includes a list of pages on the website that a crawler is discouraged, but not legally or technically disallowed, from visiting. Although this is not an enforcement standard, many ethical robots will follow the rules stated in this file.

Web scraping is a more targeted form of web crawling [11]. Web scraping is the method of downloading and storing *specific* content from a website for later retrieval or analysis [70]. This is generally done automatically by a software program by sending HTTP requests and then saving the information in the reply. The reply can optionally be parsed, storing only elements of interest. Sophisticated web scrapers use browser automation programs and APIs to interact with websites as if they were a human user.

2.4.1 Selenium

Selenium is a suite of tools that supports automation of web-browsers in a way that reflects a human user experience [60]. It is primarily used for automated testing of web applications, but supports any use case of browser automation, such as web scraping [58]. Automation is achieved using a Selenium Webdriver, that delegates requests and responses between Selenium and the browser and can be accessed through official libraries in various programming languages [60]. Selenium Webdriver has over the years, together with the W3C standards group, turned into an official web browser standard, called the WebDriver specification [65, 46]

Selenium supports several web browsers through different WebDriver implementations, called drivers [61]. These browsers include Google Chrome, Microsoft Edge, Mozilla Firefox and Safari. Each browser has custom capabilities and options, such as running in headless mode (no user interface) or browsing in private mode. WebDriver also provides a multitude of functionalities, most notably functions for finding web elements, running JavaScript and interacting with the web page. Finding web elements is achieved using the XPATH query language [43] or using CSS selectors [35], which specify the query used when searching the DOM for elements. The ability to run JavaScript directly on a web page makes it possible to fetch data or alter the web page. The web page may also be interacted with, for example, by clicking buttons, scrolling, taking screenshots and saving cookies [62].

Selenium grid allows the execution of a WebDriver script on remote machines running different operating systems and browsers [59]. A Selenium grid runs as its own server in a hub and node configuration where the hub acts as a single entry point to run, for example, a web crawler [57]. When a new node is connected to the hub, the node reports the platform and web browsers that are available on it. Selenium nodes provide the option of running different web browsers on Windows, Linux or MacOS. Each node also has the ability to run several browsers on the same machine as well as multiple instances of the same web browser. Selenium nodes can also act as a relay between the Selenium hub and an Appium instance, allowing Android and iOS devices to be connected to the grid [45].

2.4.2 Appium

Appium is an open-source project which supports UI-automation of mobile platforms, including Android and iOS [47]. Similar to Selenium, Appium follows the WebDriver specification and supports drivers for Chrome, Firefox, and Safari, which are programmatically controlled through Python and other supported programming languages. As a result of Appium following the WebDriver specification, it is directly compatible with web crawlers developed for Selenium.

Web scraping on mobile devices using Appium can be done either through an emulator or through physical hardware. When web scraping on physical hardware using Appium, Appium connects the web scraper to a proxy, which forwards instructions to the mobile device. For Android devices the Android Debugging Bridge (adb) is used, which provides access to a Unix shell that can be used to control the web browser

on the device [2]. For iOS devices, the connection is established through Apple's integrated development environment, Xcode [3]. Xcode uses a device connectivity stack, which provides an interface for installing and running apps on iOS devices.

2.5 GDPR and ePD

Cookies and cookie paywalls handle and collect the personal data of EU citizens and are therefore covered by the GDPR [20]. According to the GDPR, users must be notified of any tracking and be able to act, opt out, or leave before any tracking is initialized. This requires the elicitation of consent. GDPR defines consent as:

> *"...any freely given, specific, informed and unambiguous indication of the data subject's wishes by which he or she, by a statement or by a clear affirmative action, signifies agreement to the processing of personal data relating to him or her"* [18, p. 6].

User consent is, however, only one of six legal bases for the processing of personal data [18, p. 36]. The other five bases are: necessary for a contract involving the data subject, legal obligations, vital interests of the data subject, public interest, and legitimate interest of the data controller. When processing data, only one legal basis needs to be chosen, but once chosen, it cannot be changed [4]. It is, however, possible to identify multiple bases.

The ePD acts as a complement to the GDPR, sometimes overriding it, by specifying how electronic communication providers should handle personal data [27]. According to Koch [27], in order to comply with both GDPR and ePD, entities must:

- Receive users' consent before using any cookies except strictly necessary cookies.
- Provide accurate and specific information about the data each cookie tracks and its purpose in plain language before consent is received.
- Document and store consent received from users.
- Make it as easy for users to withdraw their consent as it was for them to give their consent in the first place.
- Allow users to access your service even if they refuse to allow the use of certain cookies

To ensure compliance with the GDPR and the ePD, the European Data Protection Board (EDPB) provides guidelines on the collection of consent [17]. In studying whether the guidelines are being followed, the Cookie Banner Taskforce [13], created by the EDPB, found and reported several different types of practices being used on websites. Additionally, the report states the consensus view among the Supervisory Authorities in the EU on whether identified practices are in accordance with the privacy regulations.

2.6 Transparency and Consent Framework

The Transparency and Consent Framework (TCF), created by IAB Europe, is an industry standard and tool designed to create a standardised experience when making privacy choices on websites [22]. The overarching goal of the TCF is to apply the principles and requirements from the GDPR and the ePD in the handling of user consent [22]. The standard specifically aims to provide accurate and specific information to the user about the data that is stored and the purpose for which it is stored. This is implemented through consent notices, such as cookie banners, provided by Consent Management Platforms (CMPs).

CMPs provide functions including presenting a consent banner to ask for user consent and logging the provided response of a user [51]. After consent is given, the CMP regulates the activation of cookies and other technologies based on the given consent in line with the EU data protection regulations. Additionally, CMPs package user preferences in a standardized payload called the Transparency and Consent String (TC String).

A TC String is an encoded HTTP-transferable string that enables communication of transparency and consent information [23]. The TC String contains information such as the number of vendors to which data is conveyed based on user consent or legitimate interest, as well as the purposes for which consent and legitimate interest are used as a basis for data processing. A list of these purposes can be found in Appendix A. The information is passed to all relevant parties including the data subject, the publisher and third-party companies known as vendors. A TC String is stored in the user's web browser as either a persistent cookie or as an entry in the browser's local storage.

Together with presenting a consent banner, many CMPs also provide integration with cross-site subscription-based models as a way of offering additional revenue streams for websites. Companies such as Content Pass GmbH [19] and Traffective GmbH [66] offer such subscription-based models in the form of contenpass and Freechoice, respectively. These products provide the option of paying a monthly fee to gain access to all partnered websites without personalised advertisement [49]. Most CMPs offer integration options for contentpass, whereas Freechoice offers its services only to the websites in its own CMP network. The websites using the contentpass and Freechoice subscription models receive compensation for the traffic generated by subscribed users.

2.7 Related Work

After the GDPR came into effect in 2018, there has been an increase in the number of websites in Europe displaying a cookie consent notice [15]. Since then, studies have examined the prevalence of cookie notices and paywalls, and later also cookie paywalls. Even though only a subset of these studies focus on cookie paywalls, the methods presented for studying other types of cookie notices and paywalls can be applicable when studying this specific type of paywall.

This section begins with covering exploratory studies of cookie notices and paywalls in Section 2.7.1, followed by empirical and exploratory studies of cookie paywalls in Section 2.7.2. Finally, Section 2.7.3 presents studies discussing challenges with mobile emulation.

2.7.1 Cookie Notices and Paywalls

Rasaii et al. [53] researched cookie banners and tracking cookies on the Tranco top-10k [30] websites in both EU and non-EU countries as well as on a desktop and emulated mobile device. In the study, cookie banners were found on approximately 47% of the websites when inside the EU and on less than 30% when in non-EU regions. It was also discovered that more tracking cookies were sent both before and after accepting or rejecting cookies when accessing websites from a vantage point outside of the EU. The number of third-party and tracking cookies was further shown to differ when using an emulated mobile device compared to a desktop browser. The mobile device was emulated by changing the user agent and screen size of a desktop browser and resulted in a difference on 14.6% of the analysed websites. The prevalence of cookie banners did not change significantly between the two setups.

Similar to Rasaii et al, van Eijk et al. [16] published a study on the impact of user location on cookie notices. In the study, a VPN provider with several vantage points in Canada, America, Switzerland and the EU was used to simulate users in different countries. To detect cookie banners, CSS element names were obtained from a crowd-sourced list with element names that are typically used for cookie banners. The authors found that, with the exception of .com websites, websites seemed to follow the cookie notice requirements based on the expected location of their audience rather than the location of individual users. That is, the country code Top Level Domain (ccTLD) instead of the VPN. Thus, websites based in the EU tended to have a higher prevalence of cookie notices. The .com websites, without a ccTLD, displayed an increase of cookie banners by 102% when using a vantage point in the EU.

The first study to investigate the privacy impacts of paywalls was conducted by Papadopoulos et al. [48]. In the study, an automated approach using machine learning was used to research the prevalence and types of paywalls. It was observed that 7.6% of websites used paywalls in 2019 and that the number of paywalls seemed to double every 6 months, showing continued rapid growth until the end of the study in 2019. When paying the fee of these paywalls the tracking of users was not significantly reduced. Additionally, based on how restrictive paywalls are, the authors introduced two broad categories; (i) *hard paywalls*, where users must pay a fee in order gain access to the site and (ii) *soft paywalls* that allow limited, free-of-charge viewing for a specific amount of time or number of visits.

2.7.2 Cookie Paywalls

In 2024, Müller-Tribbensee et al. [44] reported that the first known cookie paywall was introduced in 2018 in the online version of the Austrian newspaper Der Standard.

Using the Wayback Machine [24], the authors further studied the top 50 publishers in 21 European countries. Doing this, they showed that the widespread adoption of cookie paywalls started in 2021 and that there has been a continued growth since.

The term "cookie paywall" was coined in 2022 by Morel et al. [33] in a study where cookie paywalls were manually identified. The study examined 2800 websites from 13 different European countries, of which 61 websites used paywalls and 13 websites implemented cookie paywalls. It was further discovered that 11 out of the 13 websites implementing cookie paywalls also used the TCF and that none of these websites stored any trackers before consent was given. Morel et al. also found that, on some websites, the user was presented with different consent notices depending on the web browser that was used.

The detection process was later automated by Morel et al. [32] and Rasaii et al. [52]. Both studies developed a web crawler using Selenium on the Mozilla Firefox web browser. However, neither of the studies considered the use of different browsers nor examined the role of mobile devices.

In the study conducted by Morel et al. [32], further examples were shown of the TCF being used on websites implementing cookie paywalls. The study used a web crawler on the top 1 million URLs rated by Tranco [30] in order to find and classify websites using cookie paywalls. In the study, 431 websites were found to use cookie paywalls, all of which used the TCF. Additionally, 14 of the websites were discovered to still collect personal data by default after paying for a subscription, meaning a user had to manually decline cookies.

In the study by Rasaii et al. [52], the top 10k URLs of eight different vantage points, as rated by Google Chrome User Experience Report [8], were crawled, resulting in 45 222 websites. The study found that 280 of the examined websites used a cookie paywall, many of them being in the top 1000 domains in the country of the vantage point. In general, the sites using cookie paywalls sent more cookies on average compared to sites without cookie paywalls. In contrast to the results of Morel et al. [32], none of the websites using cookie paywalls were found to send any tracking cookies after paying their fee.

Both the study by Morel et al. [32] and Rasaii et al. [52] used language-based approaches to identify whether a website uses a cookie paywall. Morel et al. first searched for relevant web elements and then performed natural language processing on the text found in these elements. Rasaii et al. instead searched the website for cookie paywall-related keywords and then filtered out the smallest element that encompasses all keywords. The current thesis draws inspiration from both of these approaches by first searching for relevant elements and then for whole keywords and phrases.

2.7.3 Emulation Discrepancies

When studying fingerprinting on mobile devices, Yang et al. [69] found that websites were served differently depending on the type of device that accessed them. More specifically, whether the website was accessed using a mobile browser or a desktop

browser with a changed user-agent string and screen resolution to emulate a mobile browser. The DOM structure of the websites differed for some of the tested websites depending on whether they were accessed using the "real" smartphone or the emulated mobile browser. The results indicate that some websites use other methods than the user-agent string and the screen resolution to detect the type of a browser, and that these websites may not return their mobile version to an emulated mobile device. Yang et al. further point out that, even though tools have been built to easier emulate mobile browsers, they require manual fine tuning for each new device.

Cassel et al. [7] further researched the differences between emulated mobile browsers and browsers run on actual mobile hardware. In addition to user-agent string and screen resolution, other browser properties and sensor API calls were spoofed to create an almost identical fingerprint to that of mobile Firefox. Despite this, it was determined that the distribution of first and third-party cookies differed significantly between the emulated and the mobile browser.

3

Methodology

To answer the research questions in Section 1.1, a crawler program was developed to collect data on websites implementing cookie paywalls. The crawler needed to be able to detect cookie paywalls at different stages, collect TC String data and traverse websites to search for additional paywalls. These functionalities further had to be performed using different combinations of web browsers, operating systems and geographic locations.

The design and implementation of this crawler were determined during the exploratory stage of the thesis. During this stage, the crawlers developed by Morel et al. [32] and Rasaii et al. [52] were examined, and the dataset used for the study was collated. Additionally, the websites in the dataset were explored to find common characteristics in the implementation of cookie paywalls.

In this chapter, the execution of the crawler program has been divided into three phases: a manual setup phase, an initialisation phase and a crawling phase. The setup phase creates the Selenium grid, the initialisation phase configures the program according to the arguments provided by the user, and the crawling phase performs necessary actions to collect data.

3.1 Setup Phase

The aim of the manual setup phase is to create a Selenium grid that supports the desired web browsers, operating systems and geographic locations for the crawl. The Selenium grid allows a client to run a web crawler program on one operating system but use a WebDriver on another. For example, a Linux client may run an instance of the crawler program which requires a Firefox WebDriver running on MacOS. The client queries the Selenium hub, which dynamically assigns a driver to the client from its inventory in which it keeps track of what WebDriver instances are available on what platforms. The client can then use the provided remote driver in the crawler program as if the driver was available on the local device.

The inventory of WebDrivers that the Selenium hub keeps is created through the addition of Selenium nodes. Figure 3.1 provides an overview of nodes connected to the hub, in this case one node for each operating system that the crawler program can use, as outlined in Section 1.1. As can be seen in Figure 3.1, mobile devices are connected to the Selenium grid through a node acting as a relay between Selenium

hub and Appium. The relay is configured to report the capabilities of the underlying device to the hub, adding these capabilities to the inventory of the hub.

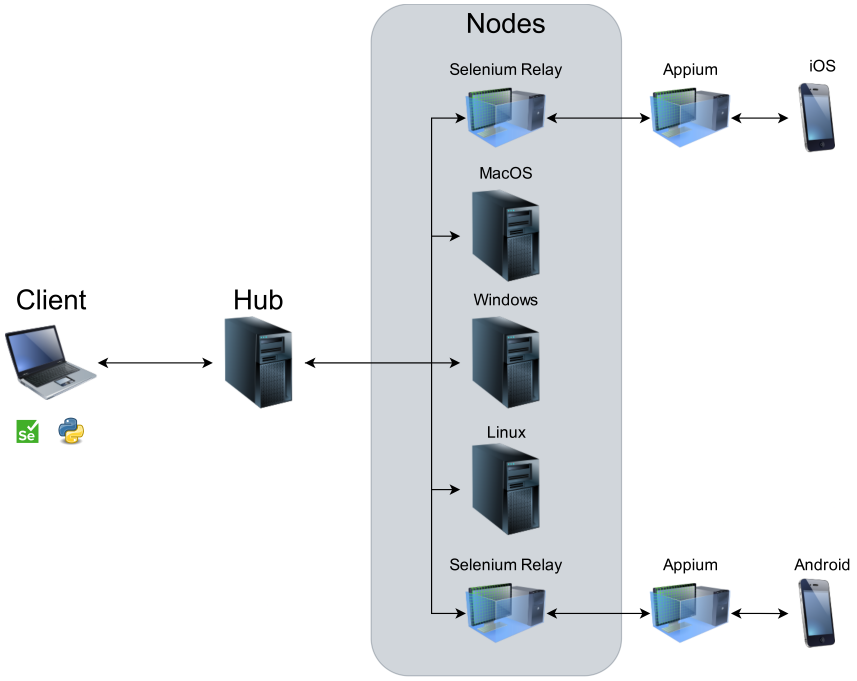

Figure 3.1: *Selenium grid architecture.*

Adding a node that emulates a user accessing websites from different geographic locations requires the node to be set up with a VPN as a proxy between the node and a website. This is done prior to starting the crawl. The VPN provider chosen for this is Proton VPN [50], as they provide native apps for all the operating systems used in this thesis and allow for control over what server to connect to. Websites may, however, implement protection mechanisms based on the IP address of the client. IP addresses linked to VPN services may trigger such protection mechanisms due to historical malicious use of such services. To limit the potential impact of this behaviour on the data collection, the VPN is only used for one vantage point, namely, the USA.

3.2 Initialisation Phase

The initialisation phase begins with the parsing of command-line arguments that establish the settings for the crawler program. Such settings include whether screenshots should be taken during the crawl, what file or directory the output should be saved to, and whether the crawler should search for double paywalls or not. Only three arguments are strictly necessary: the operating system to use, the browser to run and the websites to crawl.

Once the arguments are parsed, the program loads the URLs to be crawled from either a list of URLs or a CSV file provided as a command line argument. The dataset consists of the 431 websites found to be using cookie paywalls in Morel et

al. [32], the 280 websites found in Rasaii et al. [52], plus an additional 441 and 215 websites found on the contentpass and Freechoice websites, respectively. After removing duplicates and websites that did not present cookie paywalls when accessed using Chrome on Linux from Sweden on June 3, 2024, these combine into a final dataset of 804 *websites*.

3.3 Crawling Phase

The crawler program runs on the given dataset using drivers, running in privacy mode, that are made available through the Selenium grid. The crawler examines and interacts with each website in the dataset and when doing so, may add undesirable data to the browser's storage, which could potentially affect the behaviour of other websites. Because of this, each website is crawled by first creating a new driver, examining and interacting with the website, and finally, discarding the driver.

The examination and interaction with each website in the dataset is performed by first trying to detect the presence of a cookie paywall. If a cookie paywall is detected, the TC String is collected and cookies are accepted. Once cookies are accepted, the crawler may optionally search for double paywalls by traversing the now interactable website and search for additional paywalls.

3.3.1 Cookie Paywall Detection Stages

What a cookie paywall looks like and when it appears differs between websites. During the exploratory stage of the project, cookie paywalls were found to appear either instantly, after some time had passed, after moving the mouse cursor, or after scrolling. Furthermore, some cookie paywalls were discovered to be designed as normal cookie notices that only show a paywall once cookies had been rejected. Because of this, the crawler program has to be able to perform several types of actions to determine both whether a website uses a cookie paywall and how much of a website is accessible before a cookie paywall appears.

To accommodate for these different types of implementations, the detection algorithm is run several times, with one action performed between each run. This program flow is illustrated in Figure 3.2. If the detection algorithm detects a cookie paywall, the action required for the cookie paywall to appear is saved and no further actions are performed. If no cookie paywall is located by the algorithm, the program performs the next action and runs the detection algorithm again. A website gets flagged as not using a cookie paywall only if moving the mouse, scrolling and trying to click a reject button did not result in any cookie paywall being detected. If no cookie paywall is detected after a timeout period of 600 seconds has passed, its absence is recorded and the next website is examined. If the program times out or crashes on a website, the website is manually examined to complement the automatic data collection.

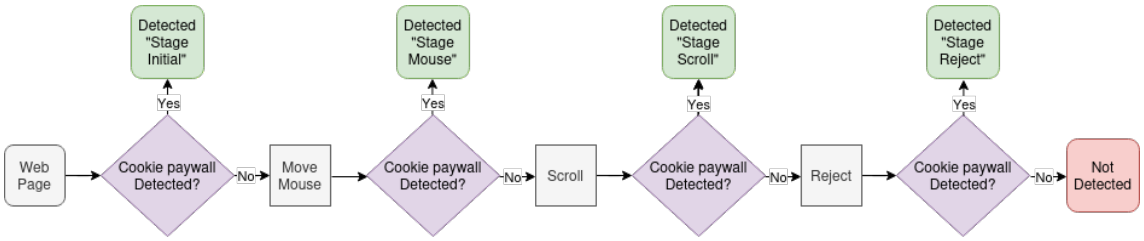

Figure 3.2: *The program flow for detecting cookie paywalls in different stages.*

3.3.2 Detection Algorithm

The detection algorithm parses the DOM of a website in search for web elements and text related to cookie paywalls. These elements are often located in the regular DOM but have also been found in shadow DOM environments and iframes. Neither shadow DOMs nor iframes are accessible using primitive scraping methods and can therefore not be modified or inspected directly by Selenium in the regular DOM. Selenium can, however, check for the *shadow_root* property in an element. Once an element with the *shadow_root* property has been located, its *shadow_root* can be used to access and interact with the shadow DOM that is attached to the element. To access the contents of an iframe, the driver first has to switch its browsing context to the HTML document in the iframe. After switching browsing context, the content of the regular HTML page cannot be accessed before switching back. The crawler program uses both of these methods to search for cookie paywalls in addition to searching the regular DOM environment.

A large portion of a website is not related to a cookie paywall, and as such, not restricting the search is inefficient and could potentially produce false positives. To circumvent this, the crawler program first searches the DOM for elements containing HTML IDs and class names known to be associated with cookie paywalls. These IDs and class names were found during the initial exploration of the dataset used for the study. A subset of the HTML attributes used by the four most prevalent CMPs in the dataset can be found in Table 3.1.

Table 3.1: *IDs and class names associated to cookie paywalls.*

Class/Id	Attribute	Associated CMP	Location
Class	cmp_paywall	Traffective GmbH	Shadow DOM
	didomi-popup-container	Didomi	Body
Id	sp_message_iframe	Sourcepoint Technologies	Iframe
	cmpwrapper	Consentmanager	Shadow DOM

Because CMPs provide other types of cookie notices than cookie paywalls, these attributes alone cannot be used to determine whether a cookie paywall has been detected. The program therefore uses a corpus of syntagms (sets of words with a sequential relationship to one another) to detect whether an element is a cookie paywall. Syntagms are used instead of keywords, as more contextual information than individual words may reduce the risk of misinterpreting a body of text.

The corpus of syntagms was created by exploring the dataset of cookie paywalls during the early stages of the thesis. During this exploration of the dataset, it was also found that websites tend to use the same terminology regardless of the web browser or platform used, and that most websites use standardised syntagms. Thus, the expected difference in the implementation of a cookie paywall on a website is not the syntagms used, but rather whether any cookie paywall is shown at all.

Examples of the syntagm combinations that are used are shown in Table 3.2. As can be seen in the table, two syntagms were, in general, considered sufficient to identify a cookie paywall. This is due to the two-choice nature of a cookie paywall, containing a consent alternative and a subscription alternative. Some syntagms are, however, valued higher than others. For example, the word "contentpass" does not need to be combined with another syntagm, since the company Content Pass GmbH exclusively deals with cookie paywalls. Conversely, some combinations consist of three syntagms to reduce the risk of false positives. If any of the syntagm combinations are found, the website is flagged as using a cookie paywall.

Table 3.2: *Syntagm combinations used for detecting cookie paywalls.*

Language	Consent Syntagm	Subscribe Syntagm
English	"continue to use with cookies" "with advertising and tracking	"sign up for a paid subscription" "read ad-free"
German	"akzeptieren und weiter" "weiter mit werbung"	"jetzt abonnieren" "bereits abonnent"
French	"j'accepte" "accepter et continuer"	"je m'abonne" "s'abonner"
*	"contentpass" "freechoice"	

The list of HTML IDs and class names is not complete and is therefore not sufficient to find all elements that may contain a cookie paywall. If no such cookie paywall specific elements are found, or no cookie paywall is found within the elements, the crawler program searches for the syntagms in the whole DOM. The search is, however, still restricted by only examining elements with a text length of at least 20 characters and not more than 5000 characters. This ensures that no inspected body of text is shorter than any of the combinations of syntagms or a great deal larger than the largest cookie paywall in the dataset.

3.3.3 TC String Collection

Once a cookie paywall has been found, the crawler program retrieves the TC String, containing the user consent information. The TC String is collected at two separate instances, before and after all cookies are accepted. If no cookie paywall is detected, no TC String is collected due to the inability of the crawler to accept cookies.

Cookies are accepted by first finding and then clicking buttons inside elements designated as cookie paywalls. The button elements that are searched for cannot have more than 30 characters of text and have to contain keywords indicating

acceptance, such as *accept*, *consent*, and *confirm*. These keywords are translated into all 12 supported languages [1] based on the translation provided by Rasaii et al. [53]. For each button found, the program attempts to click it. If it does not succeed in finding or clicking a button, the program records this and fetches the next website in the dataset. If it does succeed, the collection of the second TC String can continue.

The crawler program's collection procedure is performed in three steps, where the success of one step ends the procedure. First, the TCF API is leveraged by sending a JavaScript command to the webpage requesting the TC String. The second and third step directly search the information in the cookie storage and the browser's local storage, respectively. The TC string is stored as a key-value pair, where the key differs between websites, but all valid TC String values start with the character C followed by consent-specific data. Some common keys associated with valid TC Strings can be viewed in Table 3.3.

Table 3.3: *Cookie and local storage keys for storing the TC String.*

Location	Key
Cookie	euconsentv2 cmpconsent gdpr_consent
Local storage	euconsent tcString consentstring

The direct search for the TC string in the cookie storage is done by retrieving all cookies and searching for entries containing the TC String. Similarly, the direct search in local storage is done using JavaScript to search for keys that are commonly associated with TC Strings. During these direct searches, the found TC String is verified to start with "C", signaling a valid TC String.

If no TC String is found after all three steps, this is noted, and the crawler proceeds with the next operation. Depending on whether this is the first or second time the TC String is collected, this operation is either to accept cookies, or to search for additional paywalls on the website.

3.3.4 Double Paywalls

After accepting cookies, the website can be interacted with and traversed in order to search for additional paywalls. These paywalls were only found on "premium" parts of websites, such as subscription-only articles, when exploring the dataset. Because of this, the program searches the webpage for element attributes that indicate the existence of premium content. For example, some websites indicate that an article is for subscription members only by including a "premium icon" with the class name *premium-icon*. If no elements with premium indicators are found, the program

[1] English, German, French, Spanish, Dutch, Italian, Danish, Polish, Russian, Japanese, Turkish, and Portuguese

searches for any article, and as a last resort, any link that does not lead off the website. Once an element has been selected, it is clicked to enter the webpage.

The detection of an additional paywall is done by searching the webpage for visible elements that either have a class name or id containing *paywall*, *paid-barrier* or *offer*, or have an attribute *allow="payment"*. When manually inspecting websites, these attributes were found to indicate possible subscription alternatives, for both paywalls and other services. Therefore, if such an element is found, the program takes a screenshot of the element and saves the URL for manual inspection. If no potential paywalls are found on a webpage, the program returns to the previous page and searches for a new element to click on.

This detection procedure is performed on up to three separate webpages on each website, where each webpage is examined only once. If a double paywall is detected or the maximum number of webpages are searched, the website has been successfully crawled and the program proceeds to the next website in the dataset.

3.4 Validation

The data produced by the crawler program is validated to provide a degree of confidence for subsequent conclusions. Due to the size of the produced body of data it would be infeasible to manually validate all of it. As such, the validation is carried out by manually inspecting a random subset of 10% of the websites for each configuration of the crawler. This inspection is conducted by examining screenshots taken by the crawler after any potential cookie paywall is detected but before trying to accept cookies.

To verify that the crawler program has correctly marked websites that present a double paywall, the screenshots taken by the program are inspected. If a screenshot is inconclusive, the URL to the webpage that contained the extra paywall is manually examined. If the webpage does not present a paywall, the entire website is manually examined by traversing different webpages.

In addition to the dataset used by the crawler program, a second dataset of 200 websites is used to ensure the syntagms used in the detection algorithm are unique enough to avoid false positives. The websites in this dataset have been manually verified on the 14 of June 2024 to not use cookie paywalls when accessing them using Chrome on Linux from Sweden. Only one configuration of web browser, operating system, and geographic location is used, as the purpose of the dataset is only to verify that the detection algorithm does not produce false positives when processing text on an arbitrary website.

4

Results

This chapter presents the data collected in June 2024 using all configurations of the crawler program, that is, combinations of browser, operating system and geographic location. First, the validation of the results is discussed in Section 4.1. The later sections present the data corresponding to each of the research questions described in Section 1.1. The results regarding the prevalence of cookie paywalls are presented in Section 4.2. Data on what actions were required before a cookie paywall appeared is provided in Section 4.3. Section 4.4 presents the data collected from the gathered TC Strings. Finally, the results regarding the the occurrence of double paywalls are presented in Section 4.5.

4.1 Validation of the Crawler Program

A manual inspection of the output from the validation set showed that the crawler program produced false negatives for six out of 2080 instances of websites, where an instance is a website accessed using one configuration of the crawler program. All of these false negatives were found on Android from the USA on three separate websites. Thus, for the validation set, the program had an accuracy of 99.7%, precision of 100% and recall of 99.7%. Furthermore, no false positives nor false negatives were produced in the second dataset, containing 200 websites with no cookie paywall.

Outside of the validation set, the crawler program failed to detect cookie paywalls on four websites in all configurations, namely *0180.info*, *teltarif.de*, *filext.com* and *leo.org*. Two of these websites, *0180.info* and *teltarif.de*, were able to detect that the web browser was controlled by Selenium and did not present the cookie paywall when Selenium was used. The website *filext.com* only presented a cookie paywall on some webpages, not including the landing page. Finally, *leo.org* implemented its cookie paywall using a number of nested shadow DOMs, splitting syntagms over several DOM trees.

4.2 Prevalence of Cookie Paywalls

Figure 4.1 presents the number of cookie paywalls detected in each of the configurations. The highest number of cookie paywalls found in any run was 795, using Firefox on Android from Sweden, and the lowest was 514 when using Edge on Linux from the USA. On average, 780 cookie paywalls were detected per run when accessing

the websites from Sweden, and 581 when accessed from the USA. Additionally, the number of detected cookie paywalls in Sweden was higher than that of the USA across all configurations of the program. Box plots showing the distribution of the number of cookie paywalls detected, aggregated over browser, operating system and geographic location, respectively, can be found in Appendix B.

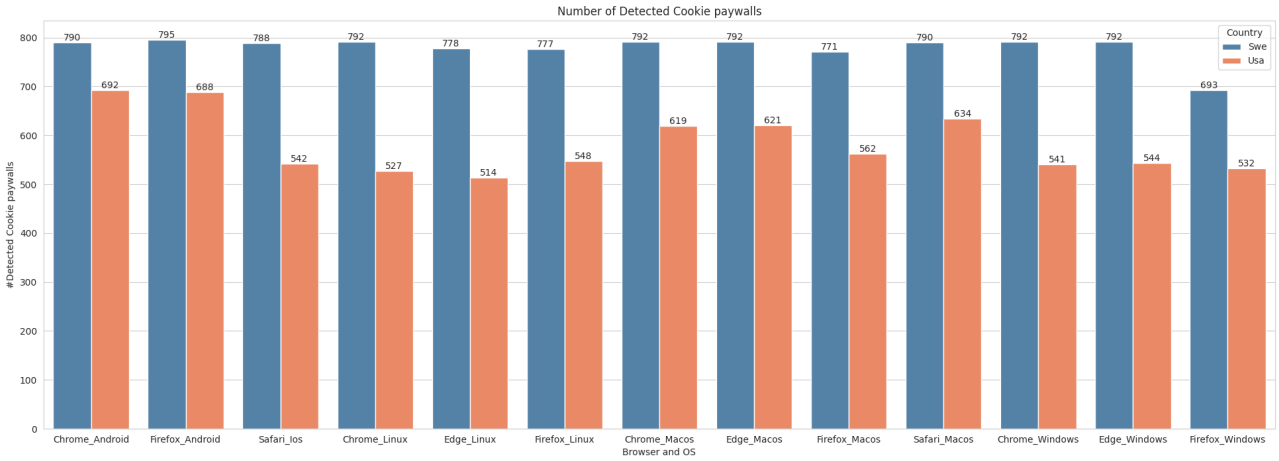

Figure 4.1: *Number of cookie paywalls detected for each combination of the studied factors.*

When considering the CMPs used by the websites, it was found that the predominant CMP among the websites presenting a cookie paywall in Sweden but not in the USA was *Traffective GmbH*. Figure 4.2 shows that this CMP was used by approximately 70% of these websites for all browser and operating system combinations except when using Android. When using an Android device, the CMP *Traffective GmbH* was the second most prevalent CMP at 20%. Figures showing the full distribution of CMPs on websites not presenting a cookie paywall can be found in Appendix C. Furthermore, a table presenting the complete list of CMPs, and the number of websites by which were used, can be found in Appendix D.

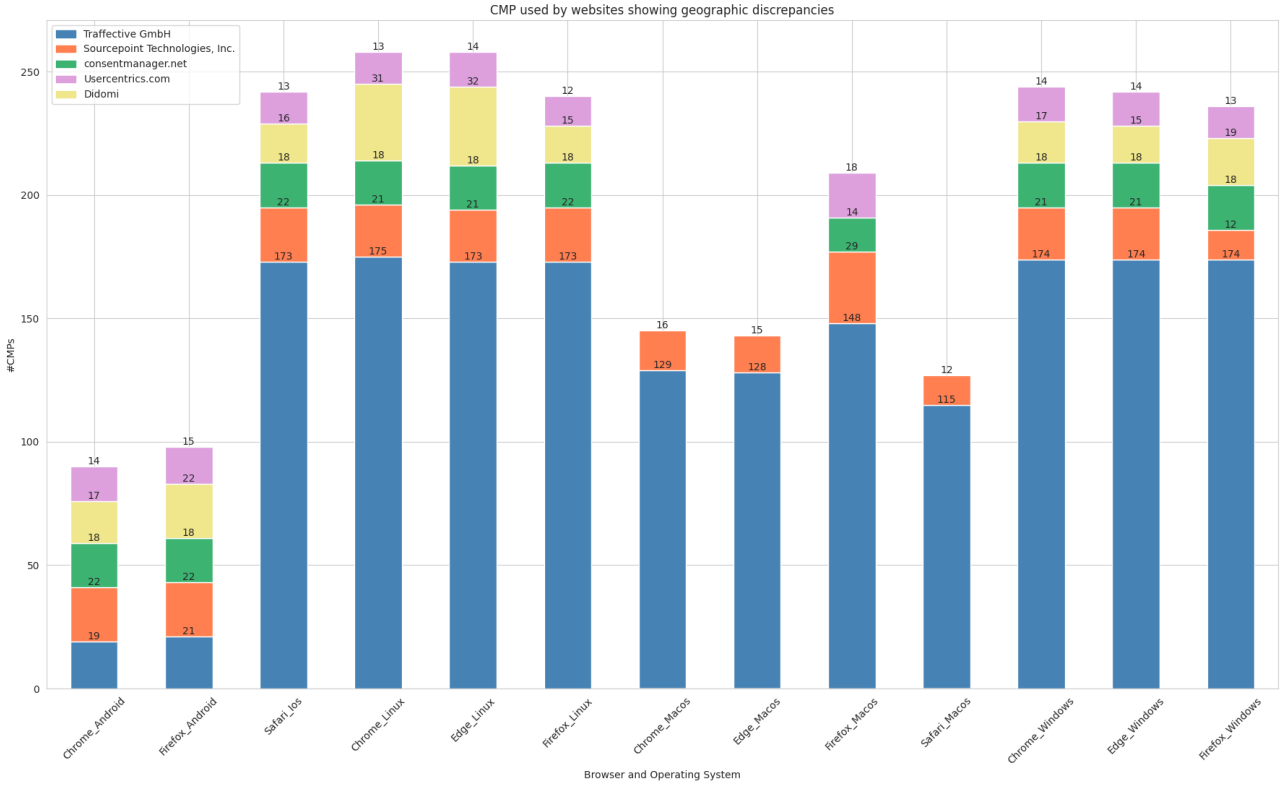

Figure 4.2: *Distribution of CMPs used by websites that present a cookie paywall when accessed from Sweden by not the USA. Only CMPs used on more than 10 websites are considered.*

4.3 Actions Required to Display a Cookie Paywall

Out of the 800 websites where a cookie paywall was detected, the crawler program produced complete data, that is, a cookie paywall was found in every configuration, for 380 websites. A frequency table for the whole dataset can be found in Appendix E. To properly be able to compare the different combinations of the studied factors, the remainder of this section focuses on these 380 websites.

The distribution of instances where a cookie paywall was presented immediately, after moving the mouse and after scrolling are presented in Table 4.1, Table 4.2 and Table 4.3. Table 4.1 shows the distribution over each browser, Table 4.2 shows the distribution over each operating system and Table 4.3 shows the distribution over each geographic location.

All three tables show that the vast majority of websites presented a cookie paywall immediately after entering the website, regardless of the browser, operating system and geographic location. The tables also show that scrolling is not necessary for any website when accessed from Sweden, when browsing using the Safari browser or when using iOS, MacOS or Windows. Across all combinations, only 17 websites

showed any kind of discrepancy in what action was required, and for each of these websites the cookie paywall appeared immediately in at least one configuration.

Table 4.1: *Distribution of required actions, aggregated over each browser.*

Browser	Initial	Mouse	Scroll
Chrome	3020 (99.34%)	19 (0.63%)	1 (0.03%)
Edge	2261 (99.17%)	18 (0.79%)	1 (0.04%)
Firefox	3014 (99.14%)	22 (0.72%)	4 (0.13%)
Safari	1516 (99.74%)	4 (0.26%)	0 (0%)

Table 4.2: *Distribution of required actions, aggregated over each operating system.*

Operating System	Initial	Mouse	Scroll
Android	1513 (99.54%)	2 (0.13%)	5 (0.33%)
iOS	758 (99.74%)	2 (0.26%)	0 (0%)
Linux	2248 (98.60%)	31 (1.36%)	1 (0.04%)
MacOS	3034 (99.80%)	6 (0.20%)	0 (0%)
Windows	2258 (99.04%)	22 (0.96%)	0 (0%)

Table 4.3: *Distribution of required actions, aggregated over each geographic location.*

Country	Initial	Mouse	Scroll
Sweden	4925 (99.70%)	15 (0.30%)	0 (0%)
USA	4886 (98.91%)	48 (0.97%)	6 (0.12%)

4.4 Processing of User Data Communicated Through the TC String

As outlined in Section 3.3.3, TC Strings were collected both before and after cookies were accepted, but only after a cookie paywall was detected. The TC Strings were then decoded using the website *iabtcf.com* and the number of vendors to which data was conveyed based on both user consent and legitimate interest were extracted. Additionally, the decoder extracted the purposes for which user consent and legitimate interest were used as bases for data processing.

The crawler program produced complete data (a valid TC String was collected in every configuration) for 335 websites after accepting cookies and 238 websites before accepting cookies. The following results in this section are derived from data collected on these 335 and 238 websites with complete data.

4.4.1 Number of Vendors

The number of websites that showed any discrepancies in the number of registered vendors after cookies were accepted were 124 and 23 for user consent and legitimate

interest, respectively. Before accepting cookies, no websites showed any discrepancies in the number of vendors using user consent and 16 websites showed discrepancies for vendors using legitimate interest.

Table 4.4, Table 4.5 and Table 4.6 show the average number of vendors that were registered on the basis of user consent after cookies were accepted as well as the percentual difference between the mean of each configuration and the mean over all configurations. Table 4.4 shows the results aggregated over each browser, Table 4.5 shows the results aggregated over each operating system and Table 4.6 shows the results aggregated over each geographic location. Tables presenting the same information for legitimate interest, both before and after accepting cookies, can be found in Appendix F. No difference was shown on any website in the number of vendors using user consent as legal basis.

For both user consent and legitimate interest, both before and after accepting cookies, a slightly higher number of vendors were registered in the USA than in Sweden. The same was true for when using Edge and Safari compared to the other web browsers and when using an Apple operating system (MacOS or iOS) compared to the other studied operating systems.

Table 4.4: *Statistics on the number of vendors using user consent as a legal basis after accepting cookies, aggregated over each browser.*

Browser	Mean	Standard Deviation	Diff Mean
Firefox	280.082	274.633	-0.054%
Chrome	280.225	274.582	-0.003%
Edge	280.248	274.606	0.005%
Safari	280.528	274.653	0.105%

Table 4.5: *Statistics on the number of vendors using user consent as a legal basis after accepting cookies, aggregated over each operating system.*

Operating System	Mean	Standard Deviation	Diff Mean
Linux	280.063	274.589	-0.061%
Windows	280.080	274.612	-0.055%
MacOS	280.429	274.639	0.070%
iOS	280.634	274.729	0.143%
Android	280.127	274.652	-0.038%

Table 4.6: *Statistics on the number of vendors using user consent as a legal basis after accepting cookies, aggregated over each geographic location.*

Country	Mean	Standard Deviation	Diff Mean
Swe	280.068	274.493	-0.059%
Usa	280.399	274.673	0.059%

When instead examining the difference between configurations on a per website basis, it was found that if there was a difference between configurations, it tended to be

small. For example, the max difference in the number of vendors using user consent as a basis for data processing was 26[1]. However, the average max difference over all websites was 1.21, with a standard deviation of 2.53. The same information for user consent and legitimate interest before and after cookies were accepted can be found in Table 4.7.

Table 4.7: *Statistics over the max difference in the number of vendors on a per website basis.*

Cookies Accepted	Legal Basis	Maximal Max Diff	Mean Max Diff	Standard Deviation
No	User Consent	0	0	0
	Legitimate Interest	3	0.12	0.43
Yes	User Consent	26	1.21	2.53
	Legitimate Interest	6	0.30	0.88

4.4.2 Purposes for Data Processing

All but one website showed no difference in what purposes user consent was used as a basis for processing data after cookies were accepted. On this website, fewer purposes were registered for all combinations involving Firefox. No website registered any purposes based on user consent before cookies were accepted. For legitimate interest, 16 and 37 websites differed between configurations before and after accepting cookies, respectively. In all cases where some discrepancy was found, the difference consisted of a varying number of purposes registered.

Table 4.8, Table 4.9 and Table 4.10 show the average number of allowed purposes for a vendor using legitimate interest, both before and after cookies were accepted. Table 4.8 shows the results aggregated over each browser, Table 4.9 shows the results aggregated over each operating system and Table 4.10 shows the results aggregated over each geographic location.

Table 4.8: *Statistics on the number of purposes allowed for vendors using legitimate interest as a basis, aggregated over each browser.*

Browser	Before		After	
	Mean	Diff Mean	Mean	Diff Mean
Firefox	0.849	-2.088%	3.028	-0.544%
Chrome	0.870	0.336%	3.050	0.167%
Edge	0.871	0.497%	3.046	0.040%
Safari	0.891	2.759%	3.066	0.694%

When accessing websites from Sweden, the average number of purposes registered was lower than when accessing from the USA. Furthermore, it was found that 20 websites used legitimate interest as a basis to select or create a profile for personalised

[1]Found on the website *as.com* with 806 vendors on Firefox on Windows from the USA and 780 on Chrome on Linux from Sweden.

Table 4.9: *Statistics on the number of purposes allowed for vendors using legitimate interest as a basis, aggregated over each operating system.*

Operating System	Before		After	
	Mean	Diff Mean	Mean	Diff Mean
Linux	0.854	-1.442%	3.043	-0.058%
Windows	0.854	-1.442%	3.043	-0.058%
MacOS	0.882	1.790%	3.031	-0.434%
iOS	0.899	3.729%	3.101	1.870%
Android	0.857	-1.119%	3.048	0.106%

Table 4.10: *Statistics on the number of purposes allowed for vendors using legitimate interest as a basis, aggregated over each geographic location.*

Country	Before		After	
	Mean	Diff Mean	Mean	Diff Mean
Sweden	0.860	-0.746%	2.945	-3.258%
USA	0.873	0.746%	3.144	3.258%

ads or content in the USA, whereas no websites did this in Sweden. When including the websites where complete data could not be obtained, this number increases to 39. All of these websites used *consentmanager.net* as their CMP.

Devices using Firefox did, on average, register fewer purposes for vendors using legitimate interest as their basis. Additionally, devices using iOS or MacOS registered more purposes before cookies were accepted. After cookies were accepted, mobile devices (Android and iOS) registered more purposes than desktop devices.

4.5 Prevalence of Double paywalls

During the execution of the crawler program, 104 websites were flagged to present a double paywall, of which 93 *websites* had actual double paywalls. Out of these, 73 websites were classified as news websites according to Cyren's URL Category Checker [14], followed by the categories *Computers & Technology* and *Business*. A more detailed categorisation can be found in Figure 4.4, in which each category is presented with the number of websites classified as said category.

Out of the detected double paywalls, the majority of websites were hosted in Germany, as can be seen in Figure 4.3. Additionally, all double paywalls were hosted in one of five European countries that are members of the EU, with Austria and France having the second and third highest number of double paywalls. The distribution countries for all 804 websites in the dataset can be found in Appendix G.

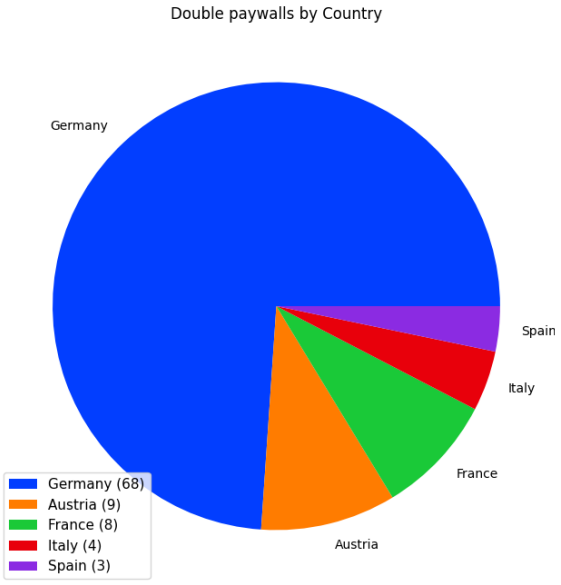

Figure 4.3 *Double paywalls by Country.*

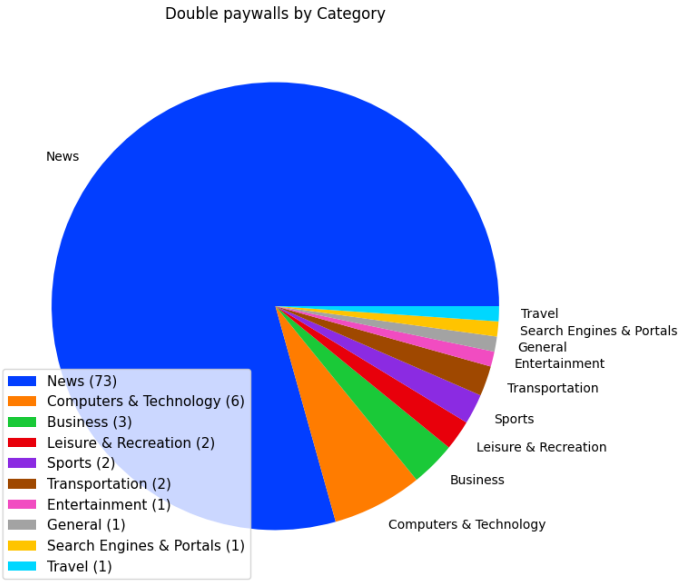

Figure 4.4 *Double paywalls by Category.*

5

Discussion

This chapter discusses the results presented in Chapter 4 followed by suggestions for future work and a short reflection on ethical considerations. The results regarding the prevalence of cookie paywalls are discussed in Section 5.1. Results on what actions were required before a cookie paywall appears is discussed in Section 5.2. Section 5.3 reviews the data collected from the gathered TC Strings. Finally, the results regarding the occurrence of double paywalls are discussed in Section 5.4. Limitations of this study are presented in Section 5.5 and potential avenues for future work are covered in Section 5.6 . Finally, an outline of some ethical considerations in Section 5.7.

5.1 Prevalence of Cookie Paywalls

Out of the studied factors, the geographic location seems to have the largest impact on whether a website presents a cookie paywall or not. For all combinations of browser and operating system, fewer cookie paywalls were detected when accessing websites from the USA than when accessing them from Sweden. It was found that, in all but two combinations, the majority of websites showing this type of behaviour used the CMP *Traffective GmbH*. Thus, it is possible that this CMP examines the location of the user and systematically chooses to not present a cookie paywall if the user is located outside of the EU. However, six websites using this CMP did not exhibit this behaviour in any of the combinations, indicating the presence of other, possibly website-related, factors. Additionally, the CMP was only used by 20% of the websites displaying this behaviour on the two combinations using Android, which would suggest that the choice of browser and operating system is also of some importance.

The type of device (mobile or desktop) accessing a website does not seem to be directly correlated to the prevalence of cookie paywalls. There were larger differences in the number of detected cookie paywalls between the two mobile devices than the difference between the iOS device and any of the desktop operating systems. However, when studying the results with higher granularity, by studying differences between all operating systems, one can see that using Android, especially from the USA, resulted in more cookie paywalls being encountered. Thus, the role of the operating system seems to be of greater importance than the type of device used. Furthermore, the operating system seems to have a larger role when accessing

websites from the USA than when accessing from Sweden.

In general, the choice of web browser did not seem to impact the prevalence of cookie paywalls on its own. In the majority of cases, fewer cookie paywalls were encountered when using Firefox compared to the other browsers, but the average number of cookie paywalls did not differ by more than 25 websites. **However, one configuration of the crawler program produced an outlier, namely when using Firefox on Windows from Sweden.** This combination presented 78 fewer cookie paywalls than the second lowest encounter rate in Sweden and only one more cookie paywall than the highest encounter rate in the USA. The reason for this outlier is unknown but may partly be a result of the CMP *Sourcepoint Technologies*, as this CMP was used by approximately 67% of the websites with no cookie paywall in this configuration.

5.2 Actions Required to Display a Cookie Paywall

The vast majority of cookie paywalls appeared immediately, and few websites required that the crawler program moved the mouse before the cookie paywall was displayed. However, when manually exploring the dataset, it was discovered that some websites displayed their cookie paywall after a delay of 15-30 seconds. Thus, it is possible that a subset of the websites for which the crawler program registered that a mouse movement was necessary would have presented a cookie paywall without any action being required if the program waited long enough. Because of numerous factors, such as network delays and varying load times, it was not possible to use time as a metric. However, if a different methodology is used, this might be a feasible metric for searching for discrepancies.

Only 17 websites, approximately 4% of the websites with complete data, displayed differences in the required action, or lack thereof, when accessed from different browsers, operating systems and geographic locations. This small number of websites showing any kind of discrepancy makes it infeasible to determine if these differences depend on the studied factors or inconsistencies in the behaviour of the crawler program, network latency or some other factor. Consequently, it is plausible that these discrepancies are website specific rather than a result of the different configurations used.

5.3 Processing of User Data Communicated Through the TC String

The analysis of the number of vendors and purposes for the different legal bases was solely conducted on websites that provided this information for every combination. This choice enabled the creation of comparable results between configurations but may also have led to potential patterns existing in the entire dataset being missed.

Discrepancies in the number of vendors between different combinations of browser, operating system and geographic location were found, but

the differences were too small to make any concrete connection to a specific factor. On a per website basis, some differences were discovered between the combinations of the studied factors, but these differences were, in general, small. Furthermore, when comparing the averages aggregated over each factor, one can see that no single factor resulted in a significant deviation from the total mean, with the largest deviation being 0.34%. These deviations were a result of a small subset of websites that on average differed by one vendor.

Similar to the number of vendors, the number of registered purposes varied between different combinations of the studied factors, but too few websites showed any discrepancies to distinguish a pattern. When examining the purposes for data processing, it was found that only one website showed any kind of discrepancy when user consent was used as a legal basis. When instead focusing on legitimate interest, a few websites (16 pre-accept, 37 post-accept) showed some kind of discrepancy in what purposes were registered. In all of these cases, the difference was constituted by the extension of the list of allowed purposes. Out of the studied factors, the geographic location seems to make the largest impact, registering more purposes in the USA than in Sweden, but because of the low number of websites presenting any type of discrepancy this pattern does not seem to be part of a larger trend.

When accessed from the USA, 39 websites used legitimate interest as a basis to select or create a profile for personalised ads and content, but no websites did so when accessed from Sweden. The use of this legal basis for these purposes is forbidden in the current version of the TCF framework (TCF v2.2) after a decision by the Belgian Data Protection Authority in 2022 [21]. This indicates that websites using *consentmanager.net* as its CMP may, based on the geographic location, disregard the rules of the IAB Europe TCF and change how the collected user data is processed.

5.4 Prevalence of Double paywalls

Approximately 11.6% of websites used a double paywall. The majority of these websites (78%) were categorised as news websites, including several large news sites such as *zeit.de*, *lemonde.fr* and *abc.es*. This is significantly higher than the proportion of news sites among websites using cookie paywalls found in previous research [32]. A possible explanation could be that this practice lends itself well to news websites as it may allow a website to entice users to subscribe to the newspaper through providing a subset of its content, whilst still being able to monetise this audience through targeted advertising. However, the predominance of news websites may instead partially be an effect of a bias in the methodology. The detection of double paywalls was automated, using a limited number of indicators that a double paywall was used. Thus, extending the set of indicators could provide a less biased search and potentially a wider variety of website categories.

The majority of double paywalls were found on websites based in Germany (73%), followed by Austria (9.7%) and France (8.6%). The high prevalence of German

websites can be explained by the majority (79%) of cookie paywalls being found on German websites (one condition for a double paywall is that there is a cookie paywall). However, the distribution of double paywalls was not proportionate to the distribution of cookie paywalls. For example, a relatively large proportion of cookie paywalls in Austria were double paywalls (35%) compared to Germany (11%) and France (21%). A possible explanation for this may be the disproportionate number of German websites in the dataset compared to any other country.

5.5 Limitations

Our study provides insight into what discrepancies can be found in cookie paywalls between different browsers, operating systems and geographic locations. However, it is important to consider certain limitations: first, an automated approach was used to collect data for the analysis of the study. A subset of this data was manually examined to provide a degree of confidence for the conclusions, but this manual verification might not guarantee complete accuracy for all the collected data. Second, websites have previously been proven to be able to detect Selenium-driven web browsers and consequently alter their behaviour [7]. It has also been shown that websites may deliver different content to detected web bots than to normal users [26]. Because of this, the crawler program may not fully represent the regular website behaviour of an actual user. Finally, the data for this study was collected over several weeks. Thus, there exists a possibility that some websites would have changed their behaviour during the time of the data collection.

5.6 Future Work

This thesis has produced a dataset of 804 websites with confirmed cookie paywalls as of June 2024. Based on this dataset and the data collected from the crawls, future research could perform a larger scale data collection and a deeper analysis of the studied websites. This includes extending the search for cookie paywalls by not limiting the search to landing pages as well as extracting more data from each website. Furthermore, the dataset may be updated to reflect the current state of cookie paywalls so that it may be possible to track and study the trend of the prevalence and behaviour cookie paywalls.

Future crawlers may want to use a larger set of geographic locations and a wider range of browsers on common operating systems. A larger set of vantage points, both located inside and outside the EU, would give deeper insights into whether the trends found in this thesis apply solely to individual countries or extend to larger regions. Studying additional browsers, especially browsers such as Chrome and Firefox on iOS can widen the basis for comparison between mobile and desktop devices.

A natural continuation of this thesis would be to explore not only the presence of cookie paywalls, but also what, if anything, replaces a cookie paywall on a website if it appears in some configurations but not in others. Additionally, a comparison of the tracking conducted by such websites could provide an understanding of how the

collection of user consent affects how user data is collected and used.

5.7 Ethical Considerations

Websites publicly accessible on the internet without any type of required authentication should not present personal information that may harm individuals or disclose information that may potentially harm organizations or businesses. Despite this, the crawler program is purposefully limited to only collecting TC Strings generated by the websites and recording the presence of cookie and double paywalls.

Some websites do not want crawler programs to access or limit the degree of access on their websites, which should be respected. The crawler program developed for this thesis respects the disallow list and any rate limiting requirements presented in the robots.txt of a website. Additionally, the crawler ensures that it does not leave the website when searching for double paywalls. Furthermore, no attempts were made to circumvent any bot protection techniques used by the websites in the dataset.

6

Conclusion

This thesis studied the effects of the web browser, device type and geographic location on the presence and behaviour of cookie paywalls and the handling of users' data. To do this, an automated crawler was built to collect data from 804 websites with confirmed cookie paywalls. Using this data, it was shown that all factors affected the cookie paywall to some degree and that changing the combination of the factors predominantly affected the presence of the cookie paywall. Through further examining the research questions, described in Section 1.1 , the following conclusions were made:

RQ1: How is the prevalence of cookie paywalls affected by the browser, device type and geographic location of the user? The prevalence of cookie paywalls, presented in Section 4.2, was most affected by the geographic location used to access them, with more cookie paywalls being displayed from Sweden than the USA. The browser had little effect on the number of cookie paywalls encountered, but some combinations of browser and operating system produced outliers. The device type did not seem to directly correlate to the prevalence of cookie paywalls, but the operating system of the device did.

RQ2: Do the browser, device type and geographic location affect how much of a website can be accessed before a cookie paywall appears? As shown in Section 4.3, 98% of displayed cookie paywalls appeared immediately and only 17 websites required any type of action for the cookie paywall to appear. Due to the low number of websites displaying discrepancies, the individual effect of the studied factors remains unclear and the possibility that this was a website-specific behaviour cannot be ruled out.

RQ3: Do the browser, device type and geographic location affect how and by whom user data is processed? Discrepancies were found in the number of vendors user data was shared with and the purposes for which these vendors processed data, but the differences were small and were only found on a subset of the websites. On these websites a slight tendency for more purposes was displayed when accessed from the USA. The results used for answering this research question can be found in Section 4.4.

RQ4: How widespread is the occurrence of double paywalls? The use of double paywalls is fairly widespread, as shown in Section 4.5, with 93 websites using one. The majority of these were classified as news websites and found in Germany.

Bibliography

[1] Aleksandr. *What are tracking cookies and how do they work?* Cookiebot. 28th Dec. 2023. URL: https://www.cookiebot.com/en/tracking-cookies/ (visited on 18/04/2024).

[2] Android Developers. *Android Debug Bridge (adb) | Android Studio*. Android Developers. 9th Feb. 2024. URL: https://developer.android.com/tools/adb (visited on 24/04/2024).

[3] Apple Inc. *Xcode updates*. Apple Developer Documentation. June 2023. URL: https://developer.apple.com/documentation/Updates/Xcode (visited on 08/05/2024).

[4] Ben Wolford. *What are the GDPR consent requirements? - GDPR.eu*. Mar. 2024. URL: https://gdpr.eu/gdpr-consent-requirements/ (visited on 18/03/2024).

[5] Frederic Besson, Nataliia Bielova and Thomas Jensen. 'Hybrid Information Flow Monitoring against Web Tracking'. In: *2013 IEEE 26th Computer Security Foundations Symposium*. 2013 IEEE 26th Computer Security Foundations Symposium. ISSN: 2377-5459. June 2013, pp. 240–254. DOI: 10.1109/CSF.2013.23. URL: https://ieeexplore.ieee.org/document/6595832 (visited on 18/04/2024).

[6] Aaron Cahn et al. 'An Empirical Study of Web Cookies'. In: *Proceedings of the 25th International Conference on World Wide Web*. WWW '16. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 11th Apr. 2016, pp. 891–901. ISBN: 978-1-4503-4143-1. DOI: 10.1145/2872427.2882991. URL: https://dl.acm.org/doi/10.1145/2872427.2882991 (visited on 17/04/2024).

[7] Darion Cassel et al. 'OmniCrawl: Comprehensive Measurement of Web Tracking With Real Desktop and Mobile Browsers'. In: *Proceedings on Privacy Enhancing Technologies* 2022.1 (1st Jan. 2022), pp. 227–252. ISSN: 2299-0984. DOI: 10.2478/popets-2022-0012. URL: https://petsymposium.org/popets/2022/popets-2022-0012.php (visited on 26/04/2024).

[8] Chrome User Experience Report contributors. *Chrome User Experience Report*. 2023. URL: https://developer.chrome.com/docs/crux/ (visited on 15/03/2024).

[9] Cloudflare Inc. *What are cookies? | Cookies definition*. 2024. URL: https://www.cloudflare.com/en-gb/learning/privacy/what-are-cookies/ (visited on 13/03/2024).

[10] Cloudflare Inc. *What is a VPN?* What is a VPN? 2024. URL: https://www.cloudflare.com/learning/access-management/what-is-a-vpn/ (visited on 19/04/2024).

[11] Cloudflare Inc. *What is a web crawler? | How web spiders work*. 2024. URL: https://www.cloudflare.com/en-gb/learning/bots/what-is-a-web-crawler/ (visited on 13/03/2024).

[12] Cloudflare, Inc. *What is a bot? | Bot definition*. URL: https://www.cloudflare.com/learning/bots/what-is-a-bot/ (visited on 06/05/2024).

[13] Cookie Banner Taskforce. *Report of the work undertaken by the Cookie Banner Taskforce*. 18th Jan. 2023. URL: https://www.edpb.europa.eu/system/files/2023-01/edpb_20230118_report_cookie_banner_taskforce_en.pdf (visited on 18/03/2024).

[14] Cyren. *Website URL Category Check*. Cyren Website URL Category Checker. 2024. URL: https://data443.com/cyren-url-category-check-gate/ (visited on 18/06/2024).

[15] Martin Degeling et al. 'We Value Your Privacy ... Now Take Some Cookies: Measuring the GDPR's Impact on Web Privacy'. In: *Proceedings 2019 Network and Distributed System Security Symposium*. 2019. DOI: 10.14722/ndss.2019.23378. arXiv: 1808.05096[cs]. URL: http://arxiv.org/abs/1808.05096 (visited on 03/05/2024).

[16] Rob van Eijk et al. 'The Impact of User Location on Cookie Notices (Inside and Outside of the European Union)'. en. In: arXiv:2110.09832 [cs]. arXiv, Oct. 2021. URL: http://arxiv.org/abs/2110.09832 (visited on 03/05/2024).

[17] European Data Protection Board. *Guidelines 05/2020 on consent under Regulation 2016/679*. 5th Apr. 2020. URL: https://www.edpb.europa.eu/sites/default/files/files/file1/edpb_guidelines_202005_consent_en.pdf (visited on 18/03/2024).

[18] European Parliament and Council of the European Union. *Regulation (EU) 2016/679 of the European Parliament and of the Council*. of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). 4th May 2016. URL: https://data.europa.eu/eli/reg/2016/679/oj (visited on 13/04/2023).

[19] Content Pass GmbH. *contentpass*. en. 2024. URL: https://www.contentpass.net/ (visited on 07/05/2024).

[20] Lu Yu He Li and Wu He. 'The Impact of GDPR on Global Technology Development'. In: *Journal of Global Information Technology Management* 22.1 (2019). Publisher: Routledge, pp. 1–6. DOI: 10.1080/1097198X.2019.1569186. URL: https://doi.org/10.1080/1097198X.2019.1569186 (visited on 13/03/2024).

[21] IAB Europe. *FAQ: APD DECISION ON IAB EUROPE AND TCF - Updated February 2023*. Feb. 2023. URL: https://iabeurope.eu/wp-content/uploads/FAQ_-APD-DECISION-ON-IAB-EUROPE-AND-TCF-Updated-Febraury-2023.docx.pdf (visited on 13/03/2024).

[22] IAB Europe. *The Transparency & Consent Framework (TCF) v2.2*. 2023. URL: https://iabeurope.eu/transparency-consent-framework/ (visited on 13/03/2024).

[23] IAB Tech Lab. *Transparency and Consent String with Global Vendor & CMP List Formats*. 2023. URL: https://github.com/InteractiveAdvertisingBureau/GDPR-Transparency-and-Consent-Framework/blob/master/TCFv2/IAB%20Tech%20Lab%20-%20Consent%20string%20and%20vendor%20list%20formats%20v2.md (visited on 13/03/2024).

[24] Internet Archive. *Wayback Machine*. 2024. URL: https://web.archive.org/ (visited on 02/05/2024).

[25] IPQualityScore LLC. *Proxy Detection Database | Identify VPNs, Bots, & Tor Connections*. 2024. URL: https://www.ipqualityscore.com/proxy-detection-database (visited on 17/06/2024).

[26] Hugo Jonker, Benjamin Krumnow and Gabry Vlot. 'Fingerprint Surface-Based Detection of Web Bot Detectors'. In: *Computer Security – ESORICS 2019*. Ed. by Kazue Sako, Steve Schneider and Peter Y. A. Ryan. Cham: Springer International Publishing, 2019, pp. 586–605. ISBN: 978-3-030-29962-0. DOI: 10.1007/978-3-030-29962-0_28.

[27] Richie Koch. *Cookies, the GDPR, and the ePrivacy Directive*. 2023. URL: https://gdpr.eu/cookies/ (visited on 13/03/2024).

[28] Jörg Krause. 'Shadow DOM'. In: *Developing Web Components with TypeScript: Native Web Development Using Thin Libraries*. Berkeley, CA: Apress, 2021, pp. 43–52. ISBN: 978-1-4842-6840-7. DOI: 10.1007/978-1-4842-6840-7_3. URL: https://doi.org/10.1007/978-1-4842-6840-7_3 (visited on 13/03/2024).

[29] Pierre Laperdrix et al. *Browser Fingerprinting: A survey*. 4th Nov. 2019. DOI: 10.48550/arXiv.1905.01051. arXiv: 1905.01051[cs]. URL: http://arxiv.org/abs/1905.01051 (visited on 17/04/2024).

[30] Victor Le Pochat et al. 'Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation'. In: *Proceedings 2019 Network and Distributed System Security Symposium*. Network and Distributed System Security Symposium. San Diego, CA: Internet Society, 2019. ISBN: 978-1-891562-55-6. DOI: 10.14722/ndss.2019.23386. URL: https://www.ndss-symposium.org/wp-content/uploads/2019/02/ndss2019_01B-3_LePochat_paper.pdf (visited on 15/03/2024).

[31] Jonathan R. Mayer and John C. Mitchell. 'Third-Party Web Tracking: Policy and Technology'. In: *2012 IEEE Symposium on Security and Privacy*. 2012 IEEE Symposium on Security and Privacy. ISSN: 2375-1207. May 2012, pp. 413–427. DOI: 10.1109/SP.2012.47. URL: https://ieeexplore.ieee.org/document/6234427 (visited on 18/04/2024).

[32] Victor Morel et al. 'Legitimate Interest is the New Consent - Large-Scale Measurement and Legal Compliance of IAB Europe TCF Paywalls'. In: *Proceedings of the 22nd Workshop on Privacy in the Electronic Society*. CCS '23. ACM, Nov. 2023. DOI: 10.1145/3603216.3624966. URL: http://dx.doi.org/10.1145/3603216.3624966 (visited on 13/03/2024).

[33] Victor Morel et al. 'Your Consent Is Worth 75 Euros A Year - Measurement and Lawfulness of Cookie Paywalls'. In: *Proceedings of the 21st Workshop on Privacy in the Electronic Society*. WPES'22. event-place: Los Angeles, CA, USA. New York, NY, USA: Association for Computing Machinery, 2022, pp. 213–218. ISBN: 978-1-4503-9873-2. DOI: 10.1145/3559613.3563205. URL: https://doi.org/10.1145/3559613.3563205 (visited on 13/03/2024).

[34] Mozilla Foundation. *<iframe>: The Inline Frame element - HTML: HyperText Markup Language | MDN*. 28th Feb. 2024. URL: https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe (visited on 13/03/2024).

[35] Mozilla Foundation. *CSS selectors - Learn web development | MDN*. In collab. with chrisdavidmills et al. 1st Jan. 2024. URL: https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Selectors (visited on 18/04/2024).

[36] Mozilla Foundation. *DOM (Document Object Model)*. 2023. URL: https://developer.mozilla.org/en-US/docs/Glossary/DOM (visited on 13/03/2024).

[37] Mozilla Foundation. *Getting started with the web - Learn web development | MDN*. In collab. with SphinxKnight et al. 4th Feb. 2024. URL: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web (visited on 15/03/2024).

[38] Mozilla Foundation. *HTTP | MDN*. 2024. URL: https://developer.mozilla.org/en-US/docs/Web/HTTP (visited on 15/03/2024).

[39] Mozilla Foundation. *Pseudo-classes and pseudo-elements - Learn web development | MDN*. In collab. with chrisdavidmills et al. 1st Jan. 2024. URL: https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Selectors/Pseudo-classes_and_pseudo-elements (visited on 17/04/2024).

[40] Mozilla Foundation. *User agent - MDN Web Docs Glossary: Definitions of Web-related terms | MDN*. URL: https://developer.mozilla.org/en-US/docs/Glossary/User_agent (visited on 14/03/2024).

[41] Mozilla Foundation. *Using shadow DOM*. 2024. URL: https://developer.mozilla.org/en-US/docs/Web/API/Web_components/Using_shadow_DOM (visited on 13/03/2024).

[42] Mozilla Foundation. *Window: sessionStorage property - Web APIs | MDN*. In collab. with chrisdavidmills et al. 8th Apr. 2023. URL: https://developer.mozilla.org/en-US/docs/Web/API/Window/sessionStorage (visited on 17/06/2024).

[43] Mozilla Foundation. *XPath | MDN*. In collab. with SphinxKnight et al. 10th July 2023. URL: https://developer.mozilla.org/en-US/docs/Web/XPath (visited on 18/04/2024).

[44] Timo Mueller-Tribbensee, Klaus M. Miller and Bernd Skiera. *Paying for Privacy: Pay-or-Tracking Walls*. 6th Mar. 2024. DOI: 10.48550/arXiv.2403.03610. arXiv: 2403.03610[econ,q-fin]. URL: http://arxiv.org/abs/2403.03610 (visited on 04/04/2024).

[45] OpenJS Foundation. *Appium and Selenium Grid - Appium Documentation*. Appium and Selenium Grid. 2023. URL: https://appium.io/docs/en/2.5/guides/grid/# (visited on 26/04/2024).

[46] OpenJS Foundation. *How Does Appium Work? - Appium Documentation*. 2023. URL: https://appium.io/docs/en/latest/intro/appium/ (visited on 24/04/2024).

[47] OpenJS Foundation. *Welcome - Appium Documentation*. 2023. URL: https://appium.io/docs/en/latest/ (visited on 24/04/2024).

[48] Panagiotis Papadopoulos et al. *Keeping out the Masses: Understanding the Popularity and Implications of Internet Paywalls*. _eprint: 1903.01406. 2020. URL: https://arxiv.org/pdf/1903.01406.pdf (visited on 13/03/2024).

[49] David Pfau. *PUR models Status quo on the European market*. In collab. with Lab Consent Management of the Data Economy department. Oct. 2023. (Visited on 07/05/2024).

[50] Proton AG. *Proton VPN: Secure, fast VPN service in 90+ countries*. en. 2024. URL: https://protonvpn.com (visited on 06/05/2024).

[51] Publift. *What Is a Consent Management Platform?* 2024. URL: https://www.publift.com/blog/what-is-consent-management-platform (visited on 13/03/2024).

[52] Ali Rasaii, Devashish Gosain and Oliver Gasser. 'Thou Shalt Not Reject: Analyzing Accept-Or-Pay Cookie Banners on the Web'. In: *Proceedings of the 2023 ACM on Internet Measurement Conference*. IMC '23: ACM Internet Measurement Conference. Montreal QC Canada: ACM, 24th Oct. 2023, pp. 154–161. DOI: 10.1145/3618257.3624846. URL: https://dl.acm.org/doi/10.1145/3618257.3624846 (visited on 05/03/2024).

[53] Ali Rasaii et al. 'Exploring the Cookieverse: A Multi-Perspective Analysis of Web Cookies'. In: arXiv:2302.05353. arXiv, 10th Feb. 2023. DOI: 10.48550/arXiv.2302.05353. arXiv: 2302.05353[cs]. URL: http://arxiv.org/abs/2302.05353 (visited on 13/03/2024).

[54] Refsnes Data. *The HTML DOM (Document Object Model)*. 2024. URL: https://www.w3schools.com/js/js_htmldom.asp (visited on 13/03/2024).

[55] Iskander Sanchez-Rola et al. 'The web is watching you: A comprehensive review of web-tracking techniques and countermeasures'. In: *Logic Journal of the IGPL* 25.1 (Feb. 2017). Conference Name: Logic Journal of the IGPL, pp. 18–29. ISSN: 1368-9894. DOI: 10.1093/jigpal/jzw041. URL: https://ieeexplore.ieee.org/document/8142531 (visited on 18/04/2024).

[56] A. Sellars. 'Twenty Years of Web Scraping and the Computer Fraud and Abuse Act'. In: 28th July 2018. URL: https://www.semanticscholar.org/paper/Twenty-Years-of-Web-Scraping-and-the-Computer-Fraud-Sellars/c3c9b0a6ee180dd6c0bfa72d869eb08f45ef0b06 (visited on 29/04/2024).

[57] Software Freedom Conservancy. *Getting started with Selenium Grid*. Selenium. Section: documentation. 25th May 2023. URL: https://www.selenium.dev/documentation/grid/getting_started/ (visited on 26/04/2024).

[58] Software Freedom Conservancy. *Organizing and Executing Selenium Code*. 2024. URL: https://www.selenium.dev/documentation/webdriver/getting_started/using_selenium/#web-scraping (visited on 13/03/2024).

[59] Software Freedom Conservancy. *Selenium Grid*. Selenium. 6th Feb. 2024. URL: https://www.selenium.dev/documentation/grid/ (visited on 26/04/2024).

[60] Software Freedom Conservancy. *Selenium WebDriver - Getting started*. 2022. URL: https://www.selenium.dev/documentation/webdriver/getting_started/ (visited on 13/03/2024).

[61] Software Freedom Conservancy. *Supported Browsers*. Selenium. 2022. URL: https://www.selenium.dev/documentation/webdriver/browsers/ (visited on 15/03/2024).

[62] Software Freedom Conservancy. *WebDriver | Selenium*. 2024. URL: https://www.selenium.dev/documentation/webdriver/ (visited on 13/03/2024).

[63] StatCounter. *Browser Market Share Worldwide*. 2023. URL: https://gs.statcounter.com/browser-market-share (visited on 13/03/2024).

[64] StatCounter. *Operating System Market Share Worldwide*. 2023. URL: https://gs.statcounter.com/os-market-share (visited on 24/04/2024).

[65] Stewart, Simon and Burns, David. *WebDriver*. Webdriver. 16th Apr. 2024. URL: https://w3c.github.io/webdriver/ (visited on 24/04/2024).

[66] Traffective GmbH. *Freechoice | Deine Wahl!* de-DE. 2024. URL: https://freechoice.club/ (visited on 07/05/2024).

[67] Alisha Ukani. *Characterizing Browser Fingerprinting and its Mitigations*. 12th Oct. 2023. DOI: 10.48550/arXiv.2311.12197. arXiv: 2311.12197[cs]. URL: http://arxiv.org/abs/2311.12197 (visited on 17/04/2024).

[68] Whois.com. *Whois.com - Free Whois Lookup*. 2024. URL: https://www.whois.com/whois/ (visited on 25/04/2024).

[69] Zhiju Yang and Chuan Yue. 'A Comparative Measurement Study of Web Tracking on Mobile and Desktop Environments'. en. In: vol. 2020. 2. Apr. 2020. DOI: 10.2478/popets-2020-0016. URL: https://par.nsf.gov/biblio/10175641-comparative-measurement-study-web-tracking-mobile-desktop-environments (visited on 25/04/2024).

[70] Bo Zhao. 'Web Scraping'. In: *Encyclopedia of Big Data*. Ed. by Laurie A. Schintler and Connie L. McNeely. Cham: Springer International Publishing, 2017, pp. 1–3. ISBN: 978-3-319-32001-4. DOI: 10.1007/978-3-319-32001-4_483-1. URL: http://link.springer.com/10.1007/978-3-319-32001-4_483-1 (visited on 18/04/2024).

A

Purposes for Processing User Data

- Purpose 1: Store and/or access information on a device
- Purpose 2: Use limited data to select advertising
- Purpose 3: Create profiles for personalised advertising
- Purpose 4: Use profiles to select personalised advertising
- Purpose 5: Create profiles to personalise content
- Purpose 6: Use profiles to select personalised content
- Purpose 7: Measure advertising performance
- Purpose 8: Measure content performance
- Purpose 9: Understand audiences through statistics or combinations of data from different sources
- Purpose 10: Develop and improve services
- Purpose 11: Use limited data to select content

B

Box Plots over the Number of Detected Cookie Paywalls

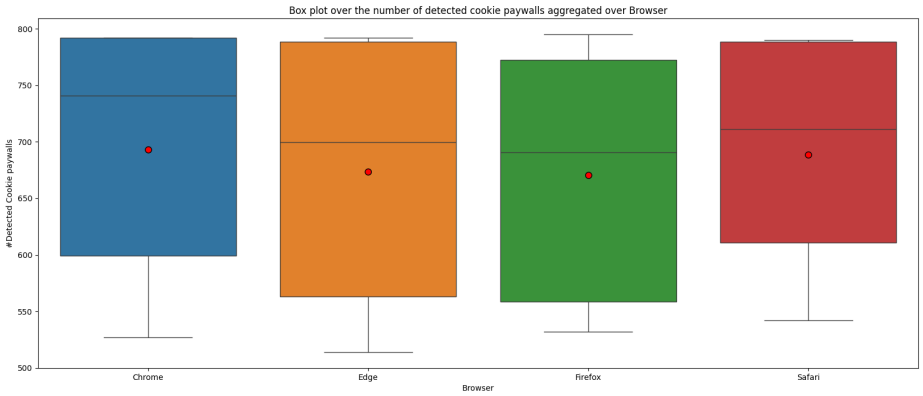

Figure B.1: *Box plots over the number of detected cookie paywalls, aggregated over each browser.*

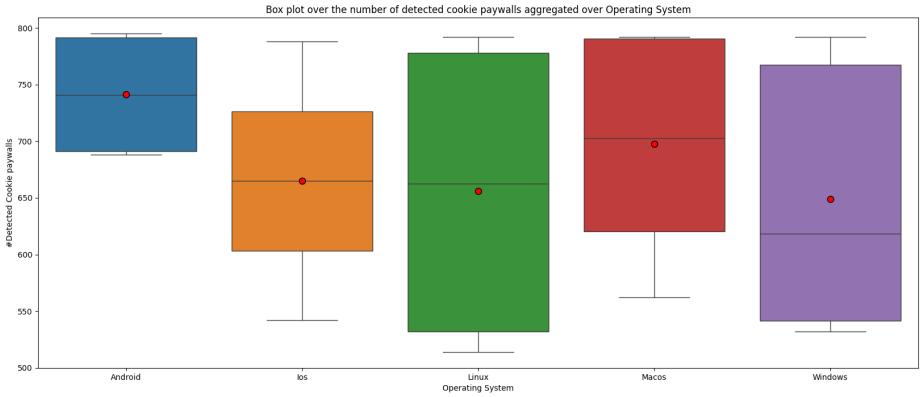

Figure B.2: *Box plots over the number of detected cookie paywalls, aggregated over each operating system.*

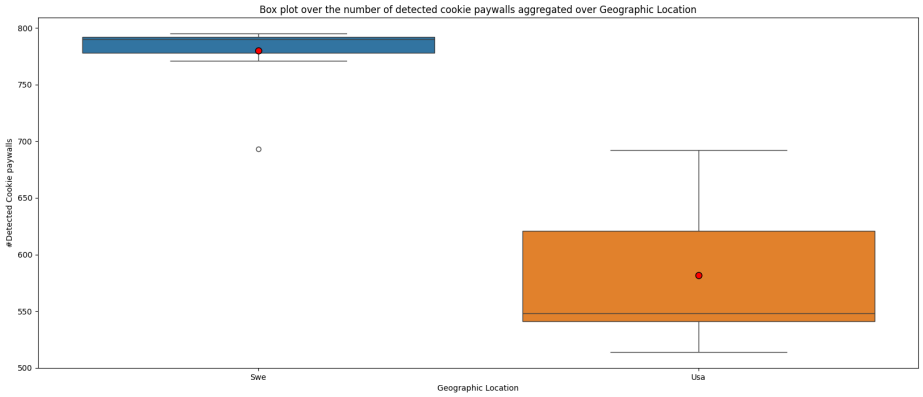

Figure B.3: *Box plots over the number of detected cookie paywalls, aggregated over each geographic location.*

C

Distribution of CMP on Websites not Presenting a Cookie Paywall

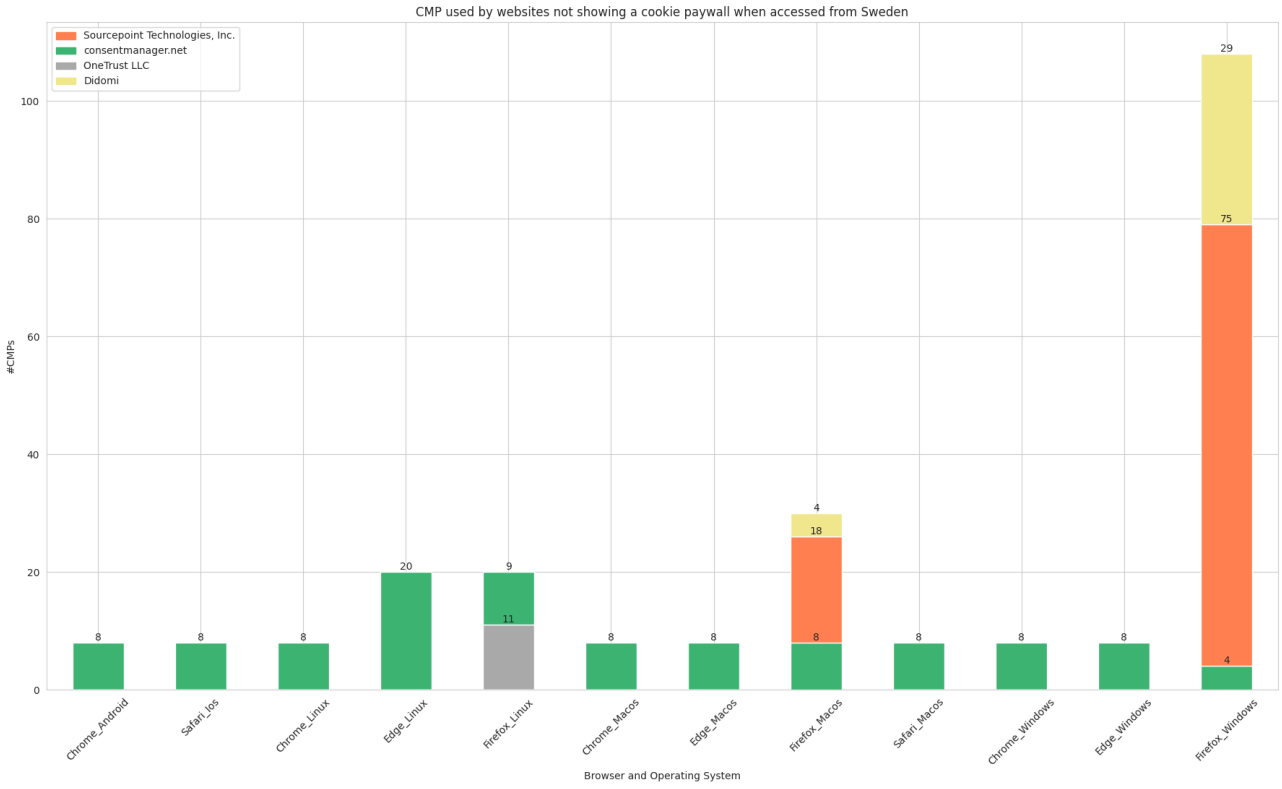

Figure C.1: *Distribution of CMPs used by websites that do not present a cookie paywall when accessed from Sweden. Only CMPs used by more than 3 websites are considered.*

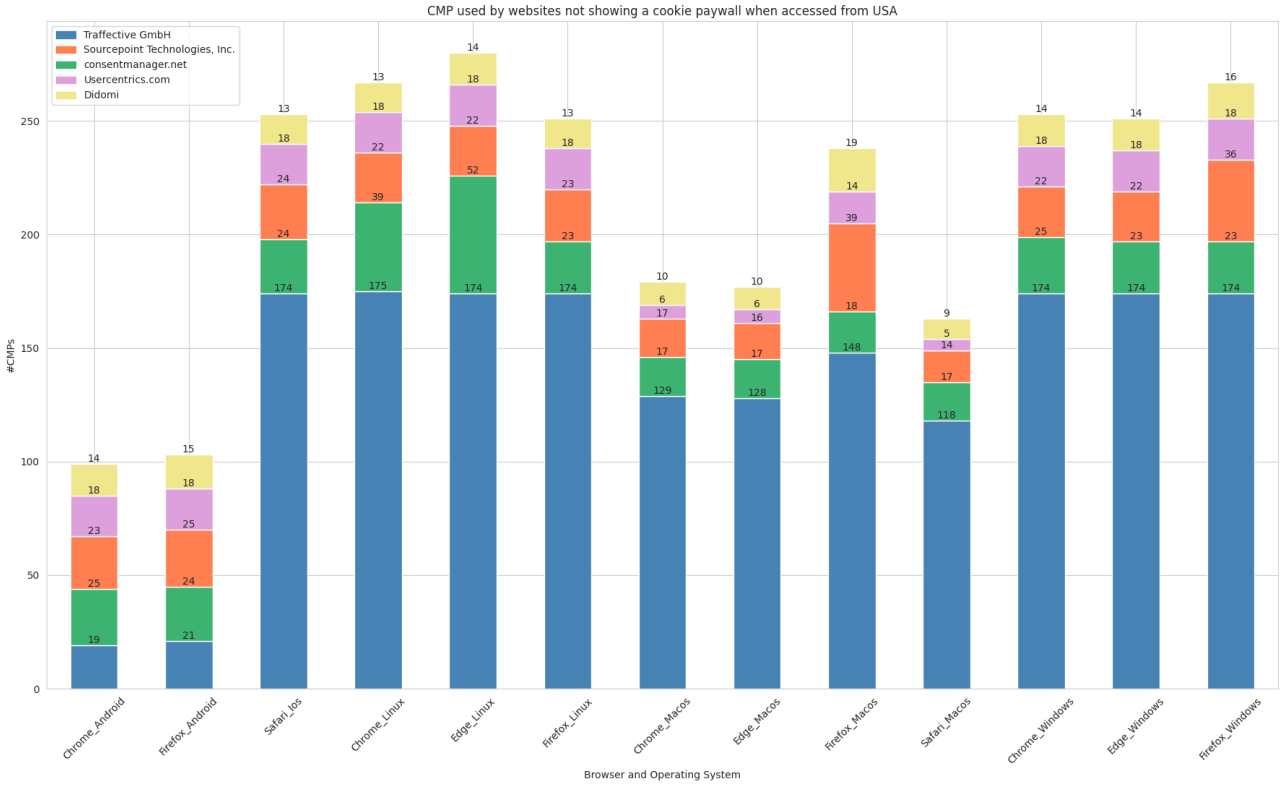

Figure C.2: *Distribution of CMPs used by websites that do not present a cookie paywall when accessed from the USA. Only CMPs used by more than 3 websites are considered.*

D

CMPs & SMPs

Table D.1: *CMPs found in the dataset.*

CMP Name	Amount
Sourcepoint Technologies, Inc.	219
consentmanager.net	195
Traffective GmbH	181
Didomi	65
Papoo Software & Media GmbH	34
OneTrust LLC	33
Usercentrics.com	23
RCS MediaGroup S.p.A.	18
Axel Springer Deutschland GmbH	10
iubenda	9
1&1 Mail & Media GmbH'	2
Société Éditrice du Monde	2
Ezoic	1
Seven.One Entertainment Group GmbH	1
Google LLC	1
Cookiebot	1
AppConsent by SFBX®	1
wetter.com GmbH	1
devowl.io GmbH	1
ALZ Software Ltd (trading as Clickio)	1
InMobi PTE Ltd	1

Table D.2: *Known SMPs found in the dataset.*

SMP Name	Amount
contentpass	366
freechoice	175

E

Frequency Table for the Actions Required

Table E.1: *Frequency table of the action required before a cookie paywall is displayed.*

Country	Operating System	Browser	Immediate	Mouse	Scroll
Sweden	Android	Chrome	760	27	1
		Firefox	764	28	1
	iOS	Safari	759	27	1
	Linux	Chrome	785	3	2
		Edge	771	4	1
		Firefox	745	29	1
	MacOS	Chrome	786	2	1
		Edge	783	5	1
		Firefox	739	27	2
		Safari	758	12	18
	Windows	Chrome	789	0	1
		Edge	785	4	1
		Firefox	662	29	0
USA	Android	Chrome	683	0	9
		Firefox	677	0	11
	iOS	Safari	539	2	1
	Linux	Chrome	515	10	1
		Edge	503	8	2
		Firefox	538	7	1
	MacOS	Chrome	611	6	1
		Edge	610	9	1
		Firefox	552	8	1
		Safari	615	6	13
	Windows	Chrome	528	11	1
		Edge	532	10	1
		Firefox	526	5	0

F

Statistics on the Number of Vendors using Legitimate Interest.

Table F.1: *Statistics on the number of vendors using legitimate interest as a legal basis after accepting cookies, aggregated over each browser.*

Browser	Mean	Standard Deviation	Diff Mean
Firefox	37.508	70.503	-0.118%
Chrome	37.547	70.504	-0.014%
Edge	37.556	70.526	0.010%
Safari	37.646	70.752	0.248%

Table F.2: *Statistics on the number of vendors using legitimate interest as a legal basis after accepting cookies, aggregated over each operating system.*

Operating System	Mean	Standard Deviation	Diff Mean
Linux	37.502	70.446	-0.134%
Windows	37.501	70.445	-0.137%
MacOS	37.613	70.656	0.162%
iOS	37.681	70.872	0.341%
Android	37.519	70.497	-0.088%

Table F.3: *Statistics on the number of vendors using legitimate interest as a legal basis after accepting cookies, aggregated over each geographic location.*

Country	Mean	Standard Deviation	Diff Mean
Sweden	37.513	70.433	-0.105%
USA	37.592	70.645	0.105%

Table F.4: *Statistics on the number of vendors using legitimate interest as a legal basis before accepting cookies, aggregated over each browser.*

Browser	Mean	Standard Deviation	Diff Mean
Firefox	6.514	20.950	-0.379%
Chrome	6.547	20.945	0.127%
Edge	6.550	20.948	0.165%
Safari	6.556	20.949	0.256%

Table F.5: *Statistics on the number of vendors using legitimate interest as a legal basis before accepting cookies, aggregated over each operating system.*

Operating System	Mean	Standard Deviation	Diff Mean
Linux	6.529	20.948	-0.157%
Windows	6.530	20.951	-0.135%
MacOS	6.553	20.945	0.216%
iOS	6.559	20.958	0.304%
Android	6.529	20.954	-0.146%

Table F.6: *Statistics on the number of vendors using legitimate interest as a legal basis before accepting cookies, aggregated over each geographic location.*

Country	Mean	Standard Deviation	Diff Mean
Sweden	6.537	20.945	-0.027%
USA	6.541	20.944	0.027%

G

Cookie paywalls by Country

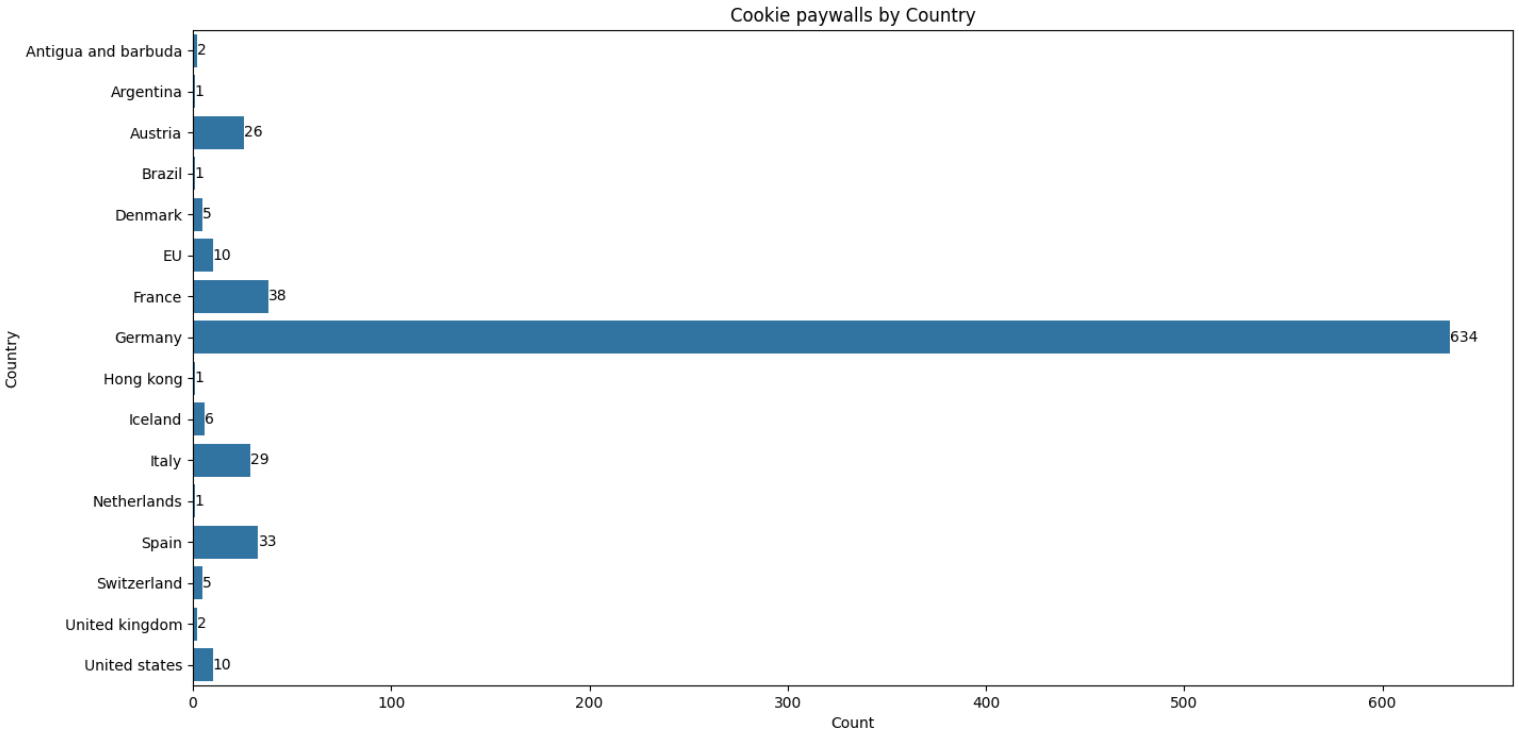

Figure G.1: *Distribution of cookie paywalls by country.*