KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

Online Tracking Technologies and Protection Mechanisms

Asuman Şenol

Supervisors:
Prof. dr. Claudia Diaz
Asst. Prof. Güneş Acar

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Science (PhD): Electrical Engineering

June 2024

Online Tracking Technologies and Protection Mechanisms

Asuman ŞENOL

Examination committee:
Prof. dr. ir. Omer Van der Biest, chair
Prof. dr. Claudia Diaz, supervisor
Asst. Prof. Güneş Acar, supervisor
Prof. dr. ir. Bart Preneel
Prof. dr. ir. Frank Piessens
Dr. Igor Bilogrevic
(Google LLC. Staff Research Scientist)
Prof. dr. Veelasha Moonsamy
(Ruhr University Bochum)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Science (PhD): Electrical Engineering

June 2024

© 2024 KU Leuven – Faculty of Engineering Science
Uitgegeven in eigen beheer, Asuman Şenol, Kasteelpark Arenberg 10, bus 2452, B-3001 Leuven (Belgium)

Alle rechten voorbehouden. Niets uit deze uitgave mag worden vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotokopie, microfilm, elektronisch of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.

All rights reserved. No part of the publication may be reproduced in any form by print, photoprint, microfilm, electronic or any other means without written permission from the publisher.

Acknowledgements

First and foremost, I would like to express my gratitude to my supervisors, Güneş Acar and Claudia Diaz, for giving me the opportunity to pursue a PhD in an outstanding research environment. Despite the challenges posed by the epidemic, Güneş was always available to mentor me, even when I was conducting my research remotely, so I never felt alone. In addition to technical matters, I always felt his support in my personal life, and his guidance always points me in the right direction. I found that his way of thinking was the best guide. Claudia's insightful criticism and support had a significant impact on my research. None of this would have been possible without their dedication and leadership.

In addition, I would like to thank my assessors during my doctoral studies Prof. Frank Piessens and Prof. Bart Preneel, my chair Omer Van der Biest, as well as Prof. Veelasha Moonsamy and Dr. Igor Bilegrovic for accepting to be on my examination committee and for providing valuable feedback. Igor's support for a range of problems that I faced during my internship was super beneficial. I deeply appreciate the time, knowledge, and advice shared by all. Furthermore, I wish to express my appreciation to Nina Taft for providing me with the internship opportunity within her team, as well as for her support. I am also grateful for the friendship and technical discussions during lunch and coffee breaks I had with Hamza Harkous, Benoit Seguin, Enrico Bacis, and Dylan Cutler throughout my internship.

Additionally, I would like to thank all members of the CNIL commission, including Vincent Toubiana, Nataliia Bielova, and François Pellegrini, as well as all other employees, and Inria researchers for offering me the opportunity to present our work at the Privacy Research Day and for recognizing our efforts with the CNIL Privacy Protection Award.

I appreciate the opportunity to grow and learn with all of my collaborators, especially Zahra Moti, Frederik Zuiderveen Borgesius, and Mathias Humbert.

A special thank you to all the COSIC researchers: Lennart, Mahdi, Aysajan, Debo, Piyush, Younes, and Rafa. I am particularly appreciative of Iness's friendship and unwavering support. Having her in my life feels like having family abroad, especially when we're far from our own families. I count myself extremely fortunate to have her in my life. Thank you Iness for being with us.

I am incredibly appreciative of my friends for their support and encouragement. Merve, Lutfi, Sercan, Burcu, Kaan, and Busra, your presence and support have been such great pillars of strength.

And a special thanks to my family, who have supported me no matter what. I could not be as brave without them. Thank you, my dear parents, Gülay and Yıldırım Şenol, my brother Can, sister-in-law İrem, my niece Ela. And I am deeply grateful to my new parents Elif and Bektaş Kar.

My best friend, half of my life and my husband Kemal will receive the final words of appreciation. I may not say it enough, but I am thankful to have someone as unselfish, kind, and encouraging in my life as you. I'm looking forward to everything that lies ahead for us.

Asuman Şenol
Leuven, June 2024

Abstract

Businesses use sophisticated technologies and algorithms to monitor individuals' online behaviors, constructing detailed behavioral profiles that resemble digital fingerprints. These profiles include plenty of personal data, including political affiliations, medical histories, and even sexual orientation, to enable targeted advertising and personalized content delivery. Despite the financial benefits online tracking brings to publishers by boosting their revenues, there are a lot of privacy concerns, most of which are caused by a lack of user awareness. Users can not fully understand the scope of tracking mechanisms used because they frequently browse the web oblivious of the various tracking technologies in place.

It is crucial to increase societal awareness about online tracking and privacy violations resulting from misuse. Additionally, it is important for people to openly express their concerns about these issues. When individuals understand and care about these matters, it can compel policymakers to establish regulations to safeguard privacy. These regulations should be straightforward to comply with and effectively enforced, alleviating the need for regular people to grapple with understanding complex online tracking practices.

Furthermore, the lack of transparency regarding the misuse of tracking data adds to consumers' concerns as they are unaware of what happens to the data when it is transferred to third-party servers. The dynamic nature of the web adds to the complexity, making it difficult to track and prevent data collection efforts due to variations in resource loading throughout page views and A/B testing scenarios. In addition, trackers use newly available functionalities as a result of the constant evolution of web platforms, which leads to an endless game of cat and mouse between developers and trackers.

Real-world examples underscore the seriousness of online tracking, showing how businesses, marketers, and even governments use it to keep track of what people are doing online. Notably, the alarming case of targeted advertising

manipulation during the 2016 US presidential election underscores the potential for malicious exploitation of online tracking data. Another recent and significant concern revolves around TikTok's ownership and its data being viewed as a national security issue in the USA. The app's extensive data collection practices, along with its ownership by a Chinese company, as well as the potential for the injection of malicious software by China through TikTok downloads, have raised apprehensions about potential data misuse and security threats. This has prompted increased scrutiny from US authorities and calls for stricter regulation. Legislative efforts in the House of Representatives have drawn attention to the discussion of national security risks associated with TikTok in April 2024.

This thesis explores a variety of scenarios to better understand online tracking methods in response to these growing concerns. We performed numerous technical studies of tracking systems, shedding light on current privacy concerns arising from the prevalence of the digital landscape. It also examines mitigation techniques for cross-platform and cross-site tracking such as browser add-ons.

This thesis also makes practical contributions to the field. For instance, we developed an automated web crawler to identify that users' email addresses are exfiltrated to third-party domains from online forms before form submission and before giving consent. Analyzing forms on 2.8 million pages we found email addresses are exfiltrated to tracking related third-party domains on 1,844 websites when visited from the EU and 2,950 when visited from the US. Remarkably, our investigation unveiled instances where users' passwords were also exfiltrated on 52 websites by third-party session replay scripts. As a countermeasure to mitigate the collection of personal data from online forms, we developed a browser add-on, which is called LeakInspector, designed to audit and block personal data collection. The study received the CNIL Privacy Protection Award in 2023 because of the tangible impact of this research.

Moreover, we examined how Chrome's user-agent (UA) reduction effort affected tracking users online. Prior work on browser fingerprinting showed that the user-agent string can facilitate covert fingerprinting and tracking of users. In response to privacy concerns, browsers such as Chrome have recently minimized the information in the user-agent string to reduce identifiability. However, Chrome also introduced user-agent client hints (UA-CH) to provide access to browser properties that were previously redacted from the user-agent string. In this empirical study, we assessed the impact of these significant changes through large-scale web measurements on the top 100,000 websites. Our findings suggest that user-agent reduction efforts were effective in minimizing the passive collection of identifying browser features but tracking scripts still have unrestricted access on Chromium-based browsers.

We also analyzed tracking and targeted advertising practices on children's

websites. By using a multilingual classifier based on web page titles and descriptions, we identified two thousand child-directed websites from a dataset of over two million pages. Using an automated web crawler, we investigated the online tracking methods used by these sites from five different vantage points. Despite regulatory efforts, such as potential amendments to the Children's Online Privacy Protection Act (COPPA) in the US and the Digital Services Act (DSA) in the EU, our findings emphasize the urgent need for stricter measures and better enforcement from regulators and stakeholders. Notably, our study revealed that approximately 90% of child-directed websites use tracking mechanisms, with about 27% displaying targeted advertisements. Some of these ads contain improper content from sex toys and flirting chat services.

Finally, we shared our investigation into the usage of browser fingerprinting as an authentication factor, focusing on fingerprinting attempts made on login and signup pages. Our findings showed that approximately 9.2% of authentication pages overall implement fingerprinting techniques. However, this rate varies significantly depending on the popularity of the website. Notably, nearly 15% of login pages among the top 1,000 sites employ fingerprinting, while approximately 8% of login pages for sites ranked between 1,000 and 5,000 use this method.

Beknopte samenvatting

Bedrijven gebruiken geavanceerde technologieën en algoritmen om het onlinegedrag van individuen te monitoren en gedetailleerde gedragsprofielen op te stellen die op digitale vingerafdrukken lijken. Deze profielen bevatten veel persoonlijke gegevens, waaronder politieke voorkeuren, medische geschiedenis en zelfs seksuele geaardheid, om gerichte advertenties en gepersonaliseerde inhoudslevering mogelijk te maken. Ondanks de financiële voordelen die online tracking met zich meebrengt voor uitgevers door hun inkomsten te verhogen, zijn er veel zorgen over de privacy, waarvan de meeste worden veroorzaakt door een gebrek aan gebruikersbewustzijn. Gebruikers kunnen de reikwijdte van de gebruikte trackingmechanismen niet volledig begrijpen, omdat ze vaak op internet surfen en zich niet bewust zijn van de verschillende trackingtechnologieën die er zijn.

Het is van cruciaal belang om het maatschappelijk bewustzijn over online tracking en privacyschendingen als gevolg van misbruik te vergroten. Bovendien is het belangrijk dat mensen openlijk hun zorgen over deze kwesties uiten. Wanneer individuen deze zaken begrijpen en er zich om bekommeren, kan dit beleidsmakers dwingen regelgeving vast te stellen om de privacy te beschermen. Deze regelgeving moet eenvoudig na te leven zijn en effectief worden gehandhaafd, waardoor de noodzaak voor gewone mensen om te worstelen met het begrijpen van complexe online trackingpraktijken wordt verlicht.

Bovendien vergroot het gebrek aan transparantie met betrekking tot het misbruik van trackinggegevens de zorgen van consumenten, omdat zij niet weten wat er met de gegevens gebeurt wanneer deze worden overgedragen naar servers van derden. De dynamische aard van het web draagt bij aan de complexiteit, waardoor het moeilijk wordt om gegevensverzamelingsinspanningen te volgen en te voorkomen vanwege variaties in de belasting van bronnen door paginaweergaven en A/B-testscenario's. Daarnaast maken trackers gebruik van nieuw beschikbare functionaliteiten als gevolg van de constante evolutie van webplatforms, wat leidt tot een eindeloos kat-en-muisspel tussen ontwikkelaars

en trackers.

Voorbeelden uit de praktijk onderstrepen de ernst van online tracking en laten zien hoe bedrijven, marketeers en zelfs overheden het gebruiken om bij te houden wat mensen online doen. Met name het alarmerende geval van gerichte reclamemanipulatie tijdens de Amerikaanse presidentsverkiezingen van 2016 onderstreept het potentieel voor kwaadwillige exploitatie van online trackinggegevens. Een andere recente en belangrijke zorg betreft het eigendom van TikTok en het feit dat de gegevens ervan in de VS als een nationaal veiligheidsprobleem worden beschouwd. De uitgebreide gegevensverzamelingspraktijken van de app, samen met het feit dat het eigendom is van een Chinees bedrijf, evenals de mogelijkheid voor de injectie van kwaadaardige software door China via TikTok-downloads, hebben geleid tot bezorgdheid over mogelijk gegevensmisbruik en veiligheidsrisico's. Dit heeft geleid tot meer toezicht van de Amerikaanse autoriteiten en tot strengere regelgeving. Wetgevende inspanningen in de Tweede Kamer hebben de aandacht gevestigd op de discussie over de nationale veiligheidsrisico's die verband houden met TikTok in april 2024.

Dit proefschrift onderzoekt een verscheidenheid aan scenario's om online trackingmethoden beter te begrijpen als reactie op deze groeiende zorgen. We hebben talloze technische onderzoeken naar volgsystemen uitgevoerd, waarbij we licht werpen op de huidige privacyproblemen die voortkomen uit de prevalentie van het digitale landschap. Het onderzoekt ook mitigatietechnieken voor cross-platform en cross-site tracking, zoals browser-add-ons.

Dit proefschrift levert ook praktische bijdragen aan het vakgebied. We hebben bijvoorbeeld een geautomatiseerde webcrawler ontwikkeld om te identificeren dat de e-mailadressen van gebruikers vanuit onlineformulieren naar domeinen van derden worden geëxfiltreerd voordat formulieren worden ingediend en voordat ze toestemming geven. Bij het analyseren van formulieren op 2,8 miljoen pagina's ontdekten we dat e-mailadressen worden geëxfiltreerd om gerelateerde domeinen van derden te volgen op 1.844 websites wanneer ze vanuit de EU worden bezocht en op 2.950 wanneer ze vanuit de VS worden bezocht. Opmerkelijk genoeg heeft ons onderzoek gevallen aan het licht gebracht waarin de wachtwoorden van gebruikers ook op 52 websites werden gehackt door sessieherhalingsscripts van derden. Als tegenmaatregel om het verzamelen van persoonlijke gegevens uit onlineformulieren te beperken, hebben we een browser-add-on ontwikkeld, genaamd LeakInspector, ontworpen om de verzameling van persoonlijke gegevens te controleren en te blokkeren. Het onderzoek ontving in 2023 de CNIL Privacy Protection Award vanwege de tastbare impact van dit onderzoek.

Bovendien hebben we onderzocht hoe de inspanningen van Chrome om gebruikersagenten (UA) te verminderen, van invloed waren op het online volgen

van gebruikers. Uit eerder onderzoek op het gebied van browservingerafdrukken is gebleken dat de user-agent-string geheime vingerafdrukken en het volgen van gebruikers kan vergemakkelijken. Als reactie op privacyproblemen hebben browsers zoals Chrome onlangs de informatie in de user-agent-string geminimaliseerd om de identificeerbaarheid te verminderen. Chrome introduceerde echter ook user-agent client hints (UA-CH) om toegang te bieden tot browsereigenschappen die eerder waren geredigeerd uit de user-agent-string. In dit empirische onderzoek hebben we de impact van deze significante veranderingen beoordeeld door middel van grootschalige webmetingen op de top 100.000 websites. Onze bevindingen suggereren dat inspanningen om user-agents te verminderen effectief waren in het minimaliseren van de passieve verzameling van identificerende browserfuncties, maar dat trackingscripts nog steeds onbeperkte toegang hebben in Chromium-gebaseerde browsers.

We hebben ook tracking- en gerichte advertentiepraktijken op kinderwebsites geanalyseerd. Door een meertalige classificatie te gebruiken op basis van titels en beschrijvingen van webpagina's, hebben we tweeduizend op kinderen gerichte websites geïdentificeerd uit een dataset van meer dan twee miljoen pagina's. Met behulp van een geautomatiseerde webcrawler hebben we vanuit vijf verschillende gezichtspunten de online trackingmethoden onderzocht die door deze sites worden gebruikt. Ondanks regelgevende inspanningen, zoals mogelijke wijzigingen in de Children's Online Privacy Protection Act (COPPA) in de VS en de Digital Services Act (DSA) in de EU, benadrukken onze bevindingen de dringende behoefte aan strengere maatregelen en betere handhaving door toezichthouders en belanghebbenden. Uit ons onderzoek blijkt met name dat ongeveer 90% van de op kinderen gerichte websites trackingmechanismen gebruiken, terwijl ongeveer 27% gerichte advertenties weergeeft. Sommige van deze advertenties bevatten ongepaste inhoud van seksspeeltjes en flirtchats Diensten.

Ten slotte deelden we ons onderzoek naar het gebruik van browservingerafdrukken als authenticatiefactor, waarbij we ons concentreerden op vingerafdrukpogingen op inlog- en aanmeldingspagina's. Uit onze bevindingen blijkt dat in totaal ongeveer 9,2% van de authenticatiepagina's vingerafdruktechnieken implementeert. Dit tarief varieert echter aanzienlijk, afhankelijk van de populariteit van de website. Opvallend is dat bijna 15% van de inlogpagina's van de top 1.000 sites gebruik maakt van vingerafdrukken, terwijl ongeveer 8% van de inlogpagina's voor sites tussen de 1.000 en 5.000 deze methode gebruiken.

List of Abbreviations

API Application Programming Interface.

CAPTCHA Completely Automated Public Turing test to tell Computers and Humans Apart.

CHIPS Cookies Having Independent Partitioned State.

COPPA Children's Online Privacy Protection Act.

CRUX Chrome UX Report.

CSRF Cross-Site Request Forgery.

DAA Digital Advertising Alliance.

DSA Digital Services Act.

eTLD Effective Top-Level Domain.

ETP Enhanced Tracking Protection from Mozilla.

FP Fingerprint.

FTC Federal Trade Commission.

GDPR General Data Protection Regulation.

HTML Hyper Text Markup Language.

HTTP Hypertext Transfer Protocol.

HTTPS Hypertext Transfer Protocol Secure.

IT Information Technology.

ITP Intelligent Tracking Prevention from Safari.

ML Machine Learning.

OBA Online Behavioral Advertising.

RTB Real-time bidding.

SOP Same Origin Policy.

TRC Tracker Radar Collector.

UA User Agent.

UA-HC User Agent Client Hints.

URL Uniform Resource Locator.

VPN Virtual Private Network.

Web RTC Web Real-Time Communication.

Web USB Web Universal Serial Bus.

WebGL Web Graphics Library.

Contents

Abstract iii

Beknopte samenvatting vii

List of Abbreviations xii

Contents xiii

List of Figures xix

List of Tables xxiii

I Online Tracking Technologies and Protection Mechanisms 1

1 Introduction 3
- 1.1 Contributions . . . 5
  - 1.1.1 Detection of Personal Data Collection From Online Forms Before Form Submission . . . 5
  - 1.1.2 Investigation of the Recent Changes in Browser Behavior Related to User-Agent (UA) String . . . 6
  - 1.1.3 Large-scale, Automated Study on Tracking and Advertising on Children's Websites . . . 7
  - 1.1.4 Identifying Authentication Pages and Their Fingerprinting Behavior . . . 7
- 1.2 Structure of the Thesis . . . 8

2 Online Tracking Mechanisms 11
- 2.1 Tracking with HTTP Cookies . . . 11
- 2.2 Cookie Syncing . . . 14

2.3 Online Behavioral Advertising 15
2.3.1 Real-Time Bidding 15
2.3.2 Regulations for Online Behavioral Advertising 17
2.3.3 Our Contributions 19
2.4 Phasing out Third-Party Cookies 20
2.4.1 Initiatives by Major Web Browsers 20
2.5 The Economical Impact of Third-Party Cookie Deprecation . . 21
2.6 Tracking via Email Addresses 22
2.6.1 Our Contributions 23
2.7 Browser Fingerprinting 24
2.7.1 Active and Passive Browser Fingerprinting 26
2.7.2 Our Contributions 27
2.7.3 Other Uses of Browser Fingerprinting 28
2.7.4 Our Contributions 29

3 Countermeasures Against Online Tracking 31
3.1 Defenses Developed by Major Browser Vendors 31
3.1.1 Google 31
3.1.2 Apple 33
3.1.3 Mozilla 34
3.2 Browser Add-ons 35
3.2.1 Using a Blocklist to Block Known Trackers 35
3.2.2 Using Algorithms to Identify Trackers 36
3.2.3 Our Contributions 36

4 Conclusion and Future Research Directions 39
4.1 Conclusion 39
4.2 Future Research Directions 40

Bibliography 54

II Publications 55

List of Publications 57

Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission 59
1 Introduction 62
2 Background and Related Work 63
2.1 Background 63
2.2 Related Work 64
3 Methods 67

3.1 Extending Tracker Radar Collector . . . 67
3.2 Discovering Inner Pages . . . 67
3.3 Identifying Email and Password Fields . . . 68
3.4 Filling Email and Password Fields . . . 68
3.5 Interaction with Consent Management Dialogs . . . 69
3.6 Measurement Configuration . . . 70
3.7 Email and Password Leak Detection . . . 70
3.8 Determining Tracker-Related Leaks . . . 72
3.9 Dataset . . . 72
4 Measurement Results . . . 73
4.1 Email Leaks . . . 73
4.2 Password Leaks . . . 76
4.3 Vantage Points: EU vs. US . . . 77
4.4 The Effect of Consent . . . 78
4.5 Mobile . . . 78
4.6 Emails Received on the Filled Addresses . . . 79
5 Does Email Exfiltration Comply With the GDPR? . . . 80
6 Security Disclosures, GDPR Requests, and Leak Notifications . 82
7 Countermeasures . . . 85
8 Limitations . . . 87
9 Ethics Considerations . . . 88
10 Conclusion . . . 88
References . . . 89

Unveiling the Impact of User-Agent Reduction and Client Hints: A Measurement Study 99
1 Introduction . . . 103
2 Background and Related Work . . . 106
2.1 What Changed and How? . . . 107
2.2 User-Agent Client Hints . . . 107
2.3 Delegating Hints to Third-Parties . . . 112
2.4 User-Agent Client Hint Opt-in and Delegation via HTML 114
2.5 Statuses and Positions of Other Browser Vendors . . . 114
2.6 Related Work . . . 115
3 Methodology . . . 117
3.1 Extending Tracker Radar Collector . . . 117
3.2 Detecting Fingerprinting Attempts . . . 118
3.3 Interaction with Consent Dialogs . . . 119
3.4 Detection of High-Entropy Value Exfiltrations . . . 119
3.5 Identifying Tracking-Related Requests . . . 120
3.6 The Crawl . . . 120
4 Measurement Results . . . 121
4.1 `getHighEntropyValues` Calls and Exfiltrations . . . 121

4.2 The Collection of User-Agent Client Hint HTTP Headers 124
4.3 User-Agent Client Hint Opt-in and Delegation via HTML 127
4.4 Reduction in High-Entropy User-Agent Exposure 127
5 Discussion 129
5.1 Limitations 130
6 Conclusion 131
References 131

Targeted and Troublesome: Tracking and Advertising on Children's Websites 147
1 Introduction 151
2 Related Work 153
2.1 Web Tracking Measurements 153
2.2 Tracking & Ads On Children's Media 154
2.3 Improper and Malicious Ads 154
2.4 Ad Transparency 155
2.5 Website Categorization 155
3 Building a List of Child-Directed Websites 156
3.1 Labeled Data for ML Classifier 157
3.2 Building the ML Classifier 158
3.3 The List of 2K Children's Websites 159
4 Web Tracking and Advertising Measurements 160
4.1 Identifying Fingerprinting Attempts 160
4.2 Identifying Rracking-related Requests 161
4.3 Discovering Inner Pages 161
4.4 Interacting with Consent Dialogs 162
4.5 Video Screen Captures 162
4.6 Identifying Ads and Ad Targeting Criteria 162
4.7 List of Crawls 166
5 Measurement Results 167
5.1 Ad Targeting and Content Analysis 168
5.2 Tracking and Fingerprinting Analysis 173
6 Discussion 177
6.1 Legal Implications 177
6.2 Research Ethics 178
6.3 Limitations 179
7 Conclusion 180
8 Acknowledgments 181
References 181

The Double Edged Sword: Identifying Authentication Pages and their Fingerprinting Behavior 199
1 Introduction 203

2 Background and Related Work 204
2.1 Browser Fingerprinting & Mitigations 204
2.2 Fingerprinting for Web Authentication 205
2.3 Login/Sign-Up Page Detection 206
3 Methodology 207
3.1 Existing Login/Sign-Up Page Detection Methods 207
3.2 Our Login/Sign-up Page Classifier 209
4 Measurement Results 214
4.1 Comparison of Login/Sign-Up Detection Techniques . . 214
4.2 Fingerprinting by Page Type 215
4.3 Comparison to Prior Results 217
4.4 Potential Usage for Anti-Fraud 219
5 Limitations 219
6 Conclusion 220
References 221

Curriculum **235**

List of Figures

2.1 How a website uses third-party cookies to place targeted ads. 3rd-party.example sets a new cookie with a unique identifier via the Set-Cookie HTTP response header. Each time the user accesses one of the websites, the browser will include any existing cookies in the Cookie HTTP header sent to their respective domains.[1] Consequently, 3rd-party.example can retrieve the previously stored cookie, enabling easy matching of visits across the two sites. 12
2.2 advertiser.com and trackerdomain.com synchronize their cookieIDs. Even without any code on Website C, advertiser.com finds out that: (i) cookieIDs *user_x* and *user_y* are the same, and (ii) *user_x* recently visited the website. Finally, both domains can match user data directly between their servers. 14
2.3 Ad Exchange model. 16
2.4 The AdChoices icon. The AdChoices icon represents an industry standard for disclosing online behavioral advertising practices. The figure is taken from [1]. 18
2.5 Google's ad disclosure pages differentiate between targeted and non-targeted ads. The upper figure corresponds to a targeted ad, denoted by *Google's estimation of your interests*, whereas the lower one pertains to a non-targeted ad, identified by *Ad personalization is turned off*. 18

3.1 The LeakInspector user interface (UI) functions as follows: When a tracker script scans ("sniffs") an input field containing personal information, LeakInspector identifies and highlights the input field. It then proceeds to block any leaky requests containing this personal data. Within the UI, users can view recent sniff and leak attempts, along with details such as the tracker domain, company, and tracker category. 37

Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission **59**

1 Components of our crawler. We integrate Firefox Relay's Fathom-based email field classifier[2] and Consent-O-Matic[3] to Tracker Radar Collector (TRC)—a web privacy measurement crawler developed by DuckDuckGo[4]. TRC is based on Puppeteer, which uses Chrome Devtools Protocol to interact with the underlying browser. We modify TRC to efficiently discover inner pages, and fill email and password fields. 64

Unveiling the Impact of User-Agent Reduction and Client Hints: A Measurement Study **99**

1 User-agent reduction in desktop and mobile Chrome browsers. 103

2 The `Accept-CH` response header mechanism used by servers to request high-entropy user-agent client hints from clients. After receiving an `Accept-CH` header during a secure top-level navigation request, the browser appends the requested client hint headers in subsequent requests. The image is adapted from Google's documentation.[5] 109

3 Requesting high-entropy CHs initially using `Critical-CH` header: When the server asks for a `Critical-CH`, the client will retry the initial webpage request with it. The image is adapted from Google's documentation.[6] 110

4 Delegating client hints to Third-Party Origins by `Permissions-Policy`. The image is based on a figure in Google's documentation.[7] 113

5 Comparing distinct 3rd-party domains that *can* run scripts and thus *can* access high-entropy UA-CHs and those that *accessed* high-entropy UA-CHs via the JavaScript API. 128

6 User-agent string format change for desktop and mobile Chrome browsers after UA reduction. 144

Targeted and Troublesome: Tracking and Advertising on Children's Websites **147**

1 A sample of improper ads found on child-directed websites in our crawls. 152

2 Pipeline for building a list of child-directed websites. 156

3 Google's ad disclosure pages indicating whether an ad is targeted or not. The top figure belongs to a targeted ad (indicated by *Google's estimation of your interests*), while the bottom one is for a non-targeted ad (indicated by *Ad personalization is turned off*.) 164

4 Overview of the ad content analysis pipeline. 167

5 Tranco rank (x-axis) distribution of sites that use targeted vs. non-targeted ads. Popular websites (below) appear to be more prone to disabling ad targeting. 169
6 Comparison of the average number of third-party and tracker domains/entities on websites containing ads vs. not containing ads. 176

The Double Edged Sword: Identifying Authentication Pages and their Fingerprinting Behavior **199**

1 Example web forms: ⓐ is a login form without a password field; ⓑ is a sign-up form without a password field; ⓒ is a login form without an email field; and ⓓ is a newsletter sign-up form that resembles an account creation form. 206
2 ML pipeline for classifying login/sign-up pages. 209
3 (1) Our crawler extends the Tracker Radar Collector. We collect login and sign-up signals, fingerprinting signals, and JavaScript execution traces. (2) We then compare all authentication page detection methods and detect fingerprinting scripts. 211
4 For login, sign-up, and homepages that perform fingerprinting, we plot the percent of pages that call each fingerprinting API. . 217
5 The Hepsiburada login page fails to load when the extension is active, and the extension has disabled fingerprinting scripts. . . 230
6 Effect of fingerprinting script blocking on Deezer's login page. Deezer's login form behaves abnormally when JShelter is activated and blocks fingerprinting scripts ⓒ, causing disruptions in the functionality of login buttons. 231

List of Tables

2.1 Summary of large-scale studies conducted on browser fingerprinting over the years. 26
2.2 Shannon's entropy for all attributes from Panopticlick, AmIUnique, and Hiding in the Crowd studies. H_M is the worst-case scenario where the entropy is maximum and all values of an attribute are unique. 26

Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission **59**

1 Desktop crawl statistics based on servers located in the EU and the US. no-action, accept-all, reject-all indicate consent modes. Crawled pages also include inner pages that we visited. 71
2 Top tracker domains and associated entities that emails or passwords are exfiltrated to in desktop crawls using the no-action mode which was conducted in May'21. *: Third-party domain is not among the request initiators; that means the leak could have been triggered by another party. Prominence (Prom.) values have been multiplied by 1,000 for readability. 73
3 The number of distinct websites where email and passwords are sent to first-party domains vs. third-party domains in desktop crawls using the no-action mode. 74
4 Top ten websites where the filled email was collected by a tracker before form submission in desktop crawls using the no-action mode. *: Not reproducible anymore as of February 2022. . . . 74
5 Per-category number of websites we crawled, filled an email field, and observed an email leak to a tracker domain (based on desktop crawls using the no-action mode). The percentage under the Leaky sites column is based on total websites where we could fill an email field (i.e. 100 * Num. of leaky sites / Num. of filled sites). 77

6 The number of distinct websites where emails were leaked and a CMP was detected in desktop crawls using the no-action mode. 79
7 The number of sites leaking emails or passwords to trackers, compared to the number of sites where we could fill an email address in desktop and mobile crawls using the no-action mode. 79

Unveiling the Impact of User-Agent Reduction and Client Hints: A Measurement Study 99

1 User-agent client hint headers along with their descriptions, sample values, and entropy levels. 105
2 The number of distinct websites where high-entropy UA-CHs were retrieved by a script and exfiltrated. On over 91% of the websites, scripts exfiltrate high-entropy UA-CHs that they retrieve via the `getHighEntropyValues` method. 121
3 Most common categories of third-party scripts calling `getHighEntropyValues` method. The right-hand column shows the number of distinct websites where we detected at least one call to `getHighEntropyValues` from the respective category. The categorization is based on DuckDuckGo's Tracker Radar dataset,[8] where a script is assigned one or more categories based on its domain. 122
4 Top tracker domains that call getHighEntropyValues (left) and exfiltrate (right) high-entropy values. 123
5 `getHighEntropyValues` arguments sorted by the number of distinct sites they were observed on. The rightmost column indicates the distinct script domains that used the particular hint. 123
6 The number of distinct sites where CH headers were observed on requests made to all parties, third parties, and tracker domains. 125
7 The number of distinct sites where UA-CHs were opted-in via the `Accept-CH` header. 126
8 The number of distinct sites delegating high-entropy UA-CHs via the `Permissions Policy` header. Most websites use '*' instead of specific origins to delegate to all possible origins. These cases are indicated in the 'Allow All' column. 127
9 The number of distinct sites where delegation is accomplished through HTML. 128
10 The distribution of CrUX websites that call getHighEntropyValues. Results are sliced by site rank. and based on successful visits. 144

11 The top third-party domain categories along with the respective counts of websites where we observed at least one third-party domain exfiltrates high-entropy UA values to their endpoints. The categorization is based on Tracker Radar's domains,[9]acknowledging the possibility of a website belonging to multiple categories simultaneously. 145
12 We found 2,190 distinct third-party fingerprinting scripts. 12.5% of them also call the `getHighEntropyValues` method. 145
13 The top third-party domain categories and the corresponding number of websites where at least one high-entropy UA-CHs was sent to a third party. 146

Targeted and Troublesome: Tracking and Advertising on Children's Websites 147
1 Classification results before and after applying threshold (with 10-fold cross-validation). 159
2 Crawl statistics based on different vantage points. *: Avg./Sum of all sites visited across all crawls. 168
3 Number of visits and scraped ads, along with percentages of ads/targeted ads per crawl. *: Percentage of targeted ads is only based on ads with disclosures. In the rightmost two columns, we include a site if we scraped at least one ad/targeted ad from one of its pages. 169
4 Top ten advertisers by the number of ads across all crawls. 171
5 Number of improper ads identified for each crawl. 171
6 Average number of third-party and tracker domains, and the prevalence of tracking and fingerprinting on child-directed websites from five vantage points. 173
7 Prevalence of tracker entities in terms of number of distinct websites in Frankfurt and NYC desktop crawls. 174
8 Websites with the most distinct tracker entities. The table shows the websites' distinct third-party tracker entities, the number of requests, cookies, and Tranco rank. 175

The Double Edged Sword: Identifying Authentication Pages and their Fingerprinting Behavior 199
1 Classifier performance on test dataset. 210
2 Assessment of login and sign-up detection methods through the analysis of a randomly sampled set of 998 websites. 213
3 For each detection technique (explained in §3), we list the number of distinct login and sign-up pages it identifies as well as the number of domains (i.e. number of CrUX list entries) it can identify a login and sign-up page for. 213

4 The list of primary fingerprinting domains and related entities where at least one fingerprinting attempt was detected during a crawl conducted in August 2023. *Some entities may have multiple associated scripts. 215
5 The overall count of unique web pages where fingerprinting attempts were observed. Inner pages include login and sign-up pages. 215
6 Percent of websites that perform fingerprinting on various page categories, grouped by popularity according to the CrUX dataset. 218

Part I

Online Tracking Technologies and Protection Mechanisms

Chapter 1

Introduction

Online tracking has become a common practice in the digital age, changing privacy norms and user experiences. Businesses create detailed behavioral profiles of individuals by monitoring their online behaviors through complex networks and algorithms. To create these profiles, a variety of personal data, such as political and religious affiliations, medical histories, and sexual orientation are collected for targeted advertising and customized content delivery. Even though online advertising provides revenue for many publishers, it poses serious concerns about freedom and privacy.

One of the main concerns is the absence of user awareness. Users frequently browse the web without realizing the extent of tracking and what data is collected, which makes it difficult for them to identify the techniques used for monitoring and leaves them with little understanding of the trackers in use [75, 92, 103]. It is crucial for users to understand these technologies and privacy risks because their voices, collectively representing societal concerns, can drive policymakers to enact laws against privacy violations.

Another concern is the difficulty in auditing the misuse of tracking data. Web users have no idea what happens after their data is transferred to third-party servers. Websites' privacy policies rarely provide information about which third parties are incorporated into the website. Researchers, authorities, and media pressure were successful in persuading some large data collectors to develop transparency tools. However, the transparency interfaces were found to be misleading and incomplete in prior research by Andreou et al. and Le Pochat et al. [8, 63].

The third concern is the difficulties related to the web's dynamism. In each page

load, different sets of resources can be loaded to the same website. Additionally, there are examples where a web page might simultaneously have two versions for A/B testing purposes [55]. This complexity can increase the difficulty of measuring and preventing data collection [35].

Finally, trackers take advantage of the evolving web platform's sophistication as browser developers regularly introduce new features and APIs such as WebRTC, and WebUSB. Trackers tend to use these new features first, as indicated by previous research highlighting fingerprinters as early adopters of uncharted web features [14]. This creates a constant cat-and-mouse situation, where trackers discover uncharted features.

Past research has shown that various online tracking methods are used by companies, advertisers, or even governments to monitor users' online activities and behaviors. According to documents made public by Edward Snowden, intelligence services track user location and network traffic using advertising identifiers [108].

In recent years, targeted advertising has become a concerning issue for online manipulation, exemplified by the Cambridge Analytica scandal during the 2016 US presidential election. Russian entities actively targeted American voters with highly controversial Facebook advertisements [98]. Using Facebook's ad targeting categories, advertisers targeted those who would be interested in controversial subjects like "gun rights." Online tracking was used as a starting point for developing some of these profiles [11].

Online tracking and profiling can also be used for determining which products are shown (price steering) or customizing the prices of products (price discrimination) on e-commerce sites. Both researchers and Internet users have uncovered evidence of personalization on e-commerce sites through price steering and discrimination, which are based on users' click and purchase history, their location, and the device they use to access the service, among other factors [51, 76]. For instance, the travel website Orbitz was found to be customizing the results of hotel searches [25]. Orbitz "steered" Mac OS X users towards more expensive hotels.

This thesis aims to shed light on various aspects of these practices to inform the public and relevant authorities, paving the way for actions such as implementing regulations, enforcing them, and fostering public pressure for change. It conducts a deep technical analysis of tracking technologies and their uses. Furthermore, it sheds light on contemporary privacy issues, emphasizing their effects on personal privacy and data security. The thesis also investigates current countermeasures used by web browsers to minimize online tracking.

1.1. Contributions

The purpose of this dissertation is to advance the understanding of online tracking and strategies for mitigating it. To that end, we have uncovered previously unknown privacy issues on millions of web pages. To identify these issues, we developed several crawlers and created web extensions aimed at reducing privacy risks. Furthermore, we shared our findings with websites where privacy violations were identified, as well as with relevant third parties responsible for these violations. As a result of our outreach efforts, some websites removed tracker domains, demonstrating the real-world impact of these studies. Additionally, our research gained public attention through multiple media outlet articles and helped to raise awareness on the subject. Our contributions can be summarized as follows:

1.1.1 Detection of Personal Data Collection From Online Forms Before Form Submission

Users often fill their email addresses into online forms for various purposes, such as authentication (along with their password) or subscribing to a newsletter. It is not uncommon for users to decide to abandon a form without submitting it. According to a survey by The Manifest, 81% of the 502 respondents have abandoned forms at least once, and 59% abandoned a form in the last month [111].[1] To identify such cases, we investigated the extent to which third-party trackers collect email addresses, and sometimes passwords, even if the user does not submit any form.

We developed an instrumented web crawler that can detect and fill email and password fields on web pages. It also records requests, responses, cookies, and specific JavaScript API calls. Further, we analyzed the most popular 100,000 websites and discovered that users' email addresses are exfiltrated to third parties before form submission on 1,844 websites when visited from the EU and 2,950 when visited from the US. Some of the websites removed third parties that exfiltrate users' emails from their pages when we disclosed our findings about this data collection. Furthermore, we found that 52 websites had accidentally collected passwords because of third-party session replay scripts; these were fixed after we shared our findings.

[1]The Manifest surveyed 502 Americans who filled out online forms recently. Most (67%) did so in the past week. The respondents included both genders and all age groups, with a majority (56%) aged 55 or older. They represented all U.S. regions.

LeakInspector: We developed a proof of concept browser add-on to audit unauthorized data collection. This browser extension blocks requests that contain personal data obtained from web forms. Moreover, it logs technical details of detected data sniffing and leakage attempts for further analysis and technical audits.

This study received the CNIL Privacy Protection Award in 2023 [19]. Additionally, it gained notable attention from the press with coverage in various publications including WIRED [67], Le Monde [65], Fortune [64] and Nos.nl [66].

This contribution was published in:

- Asuman Senol, Gunes Acar, Mathias Humbert, and Frederik Zuiderveen Borgesius. Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission. In 31st USENIX Security Symposium (USENIX Security 22), pages 1813–1830, 2022.

1.1.2 Investigation of the Recent Changes in Browser Behavior Related to User-Agent (UA) String

With this empirical study, we investigated recent changes in browser behavior regarding user-agent strings. While browsers such as Chrome have reduced identifying user-agent client hints (UA-CH) to enhance user privacy, they have also introduced high-entropy user-agent client hints (UA-CH) and a new JavaScript API to access specific browser details [114]. Using an instrumented crawler, we assessed the impact of these changes across the top 100,000 websites, measuring access to high-entropy browser features via UA-CH HTTP headers and the JavaScript API. Our findings revealed that high-entropy UA-CHs were accessed by one or more scripts on approximately 59.2% of the sites through JavaScript. Additionally, on about 91.6% of the sites where high-entropy client hints were accessed via the JavaScript API, these hints were transmitted by a tracker script to a third-party server. Notably, the usage of UA-CH HTTP headers was minimal for both first-party opt-in (1.3% of sites) and delegation to third parties (0.4% of sites). With this empirical study, we demonstrated that while efforts to minimize UA information seem to have reduced the passive fingerprinting surface, tracking scripts continue to enjoy unrestricted access by using new JavaScript API on Chromium-based browsers.

This contribution was published in:

- Asuman Senol and Gunes Acar. Unveiling the Impact of User-Agent Reduction and Client Hints: A Measurement Study. In Proceedings of the 22nd Workshop on Privacy in the Electronic Society, pages 91–106, 2023.

1.1.3 Large-scale, Automated Study on Tracking and Advertising on Children's Websites

This study addresses the lack of comprehensive research focusing on tracking and advertising practices on websites targeted at children. Initially, a classifier was developed to compile a list of child-directed websites based on web page titles and descriptions. By crawling these sites from five vantage points, the study assesses the prevalence of trackers, fingerprinting scripts, and advertisements on child-directed sites. Results indicate that approximately 90% of child-directed websites contain one or more trackers, and about 27% display targeted advertisements, potentially requiring verifiable parental consent. The research further identifies inappropriate ads through a machine-learning pipeline that analyzes both images and text extracted from ads. This process detects ads promoting dating services, weight loss products, mental health services, sex toys, and flirting chat services, some featuring repulsive and sexually explicit imagery. Furthermore, we contacted five companies displaying inappropriate ads and they stated that they immediately commenced an internal investigation and shared preliminary findings with the European Data Protection Agency (DPA).

This contribution was published in:

- Z. Moti, A. Senol, H. Bostani, F. Zuiderveen Borgesius, V. Moonsamy, A. Mathur, and G. Acar. Targeted and Troublesome: Tracking and Advertising on Children's Websites. In 2024 IEEE Symposium on Security and Privacy (SP), pages 117–117, Los Alamitos, CA, USA, may 2024. IEEE Computer Society.

1.1.4 Identifying Authentication Pages and Their Fingerprinting Behavior

This study focuses on identifying authentication pages and their behavior regarding fingerprinting which is commonly linked with cross-site user tracking. However, it can also be used to improve online security by adding an extra

layer against service abuses such as unauthorized login attempts during user authentication.

The primary contribution of this study lies in the detection of login and signup pages through the development of a machine-learning model that achieves 96-98% precision and recall. Each year, numerous studies analyze various aspects of authentication pages, including security and privacy. Previous approaches, mostly relying on heuristics, have not been as successful as our ML model, as demonstrated in our paper. We believe that our ML model can significantly help researchers by quickly identifying authentication pages with high accuracy, thereby making their investigations easier.

The second contribution of this paper involves quantifying the prevalence of fingerprinting scripts found on login and sign-up pages. While the rates of fingerprinting are similar to landing pages, disparities exist between home pages and authentication pages regarding the inclusion of third-party scripts. Fingerprinting attempts were detected on 9.2% of login pages overall; however, the rate varies depending on the website's popularity. Approximately 15% of the login pages for the top 1,000 websites use fingerprinting, compared to 8% of the login pages for the 1K–5K ranks.

This contribution was published in:

- Asuman Senol, Alisha Ukani, Dylan Cutler, and Igor Bilogrevic. The Double Edged Sword: Identifying Authentication Pages and their Fingerprinting Behavior. In The Web Conference (WWW), 2024.

1.2. Structure of the Thesis

This thesis is divided into two main parts. The first part begins with an **Introduction (Chapter I)** that outlines the objectives, contributions, and motivation of the thesis. **Chapter II**, titled "Online Tracking Mechanisms" presents an overview of the state-of-the-art in online tracking studies. In **Chapter III**, we elaborate on the protective measures implemented by major browser vendors, as well as existing browser add-ons aimed at preventing online tracking/personal data collections. Part I concludes by summarizing the findings and proposing potential future directions for further exploration.

Part II contains four publications:

1. Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission [103]

2. Unveiling the Impact of User-Agent Reduction and Client Hints: A Measurement Study [102]
3. Targeted and Troublesome: Tracking and Advertising on Children's Websites [77]
4. The Double Edged Sword: Identifying Authentication Pages and their Fingerprinting Behavior [104]

These articles are included exactly as they appeared in their peer-reviewed versions with the exception of editorial adjustments to provide for a uniform single-column layout throughout this thesis.

Chapter 2

Online Tracking Mechanisms

In this chapter, we provide the background of the many online tracking technologies, which are necessary to situate the contributions. Additionally, we explore countermeasures aimed at mitigating privacy risks on the web. Each state-of-the-art overview presented in this chapter also includes our contributions.

2.1. Tracking with HTTP Cookies

Cookies are small bits of data produced by a web server while a user browses a website; they are stored on the user's browser. They serve valuable and sometimes essential functions online. These functions include enabling web servers to maintain stateful information, like items added to a shopping cart on an e-commerce site, or monitoring the user's browsing behavior, including specific button clicks, login actions, or previously visited pages. Cookies can also retain information inputted by the user into form fields, such as names, addresses, passwords, and payment card details, for future reference. Categorically, cookies can be classified based on their lifespan and origins.

Cookies are classified into two types based on their lifespans: session cookies and persistent cookies. Session cookies are temporary and functional only during a user's active browsing session. They maintain session data, such as login information or items in a shopping cart. These cookies are deleted once the browser is closed. Conversely, persistent cookies persist beyond the browsing session. They remain stored on the user's device, holding long-term information

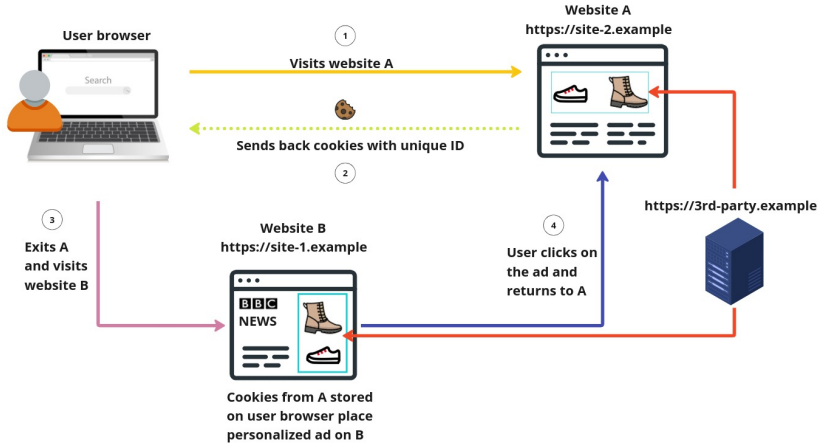


Figure 2.1: How a website uses third-party cookies to place targeted ads. 3rd-party.example sets a new cookie with a unique identifier via the Set-Cookie HTTP response header. Each time the user accesses one of the websites, the browser will include any existing cookies in the Cookie HTTP header sent to their respective domains.[1] Consequently, 3rd-party.example can retrieve the previously stored cookie, enabling easy matching of visits across the two sites.

such as user preferences or login details. Persistent cookies facilitate personalized content delivery and targeted advertising.

When discussing the origins of cookies, they can be classified as first-party or third-party depending on the source of data collection. First-party cookies originate from the website directly visited by the user. These cookies are used by the website itself to improve the user experience by preserving preferences and optimizing site functionality. Conversely, third-party cookies originate from domains other than the website being visited. Commonly they are used for advertising and tracking user behavior across various websites, which has led to significant privacy concerns due to their tracking capabilities [107, 99, 86, 82].

Cookies are associated with a specific website domain and are kept separate from cookies belonging to other domains. This separation is maintained through a security feature called the Same Origin Policy (SOP), which is implemented in web browsers. The Same Origin Policy is a crucial security measure that helps to prevent malicious attacks between different websites. It works by restricting scripts running on one website (origin) from accessing data from

[1] Other cookie attributes such as the SameSite flag affect the determination of when the cookie will be sent.

another website (origin). This limitation ensures that a script from one origin cannot tamper with or access sensitive information from another origin. An origin is defined by three components: the protocol (such as HTTP or HTTPS), the domain name (e.g. www.example.com), and the port number (if specified). If these three elements match between two URLs, they are considered to have the same origin. For example, the URLs `https://www.example.com` and `https://shop.example.com` share the same origin because they have the same protocol (HTTPS), domain (`https://www.example.com`), and port number (typically omitted, but assumed to be the default ports for HTTPS, which is port 443). As a result, cookies set by scripts running on one of these URLs can be accessed by scripts running on the other URL within the same origin.

Despite the isolation between origins facilitated by SOP, cookies may still be used for cross-site tracking, establishing connections among a user's interactions across different websites. For example, consider an e-commerce platform that has a third party, `https://3rd-party.example`, which sets a third-party cookie on the user's device as shown in Figure 2.1. Due to website B including a tracking script from 3rd-party.example as the e-commerce site, the same user can be recognized via the unique ID stored by third-party cookies. As a remark, for cookies to be used for cross-site tracking they must have "SameSite=None". The "SameSite" [105] attribute governs whether a cookie is transmitted with cross-site requests, thereby offering some defense against CSRF (Cross-Site Request Forgery) attacks. The potential attribute values are as follows:

Strict: This value ensures that the browser sends the cookie only for requests originating from the same site that set the cookie.

Lax: In this scenario, the cookie is not transmitted for cross-site requests like those to load images or frames. However, it is sent when a user navigates to the origin site from an external source, such as following a link. This represents the default behavior if the SameSite attribute isn't explicitly specified.

None: This value dictates that the browser sends the cookie for both cross-site and same-site requests. It is crucial to note that the Secure flag must also be added when setting this value, in the format SameSite=None; Secure. A secure cookie can only be transmitted over an encrypted connection, specifically HTTPS. They cannot be sent over unencrypted connections, like HTTP. This characteristic enhances the security of the cookie, significantly reducing the risk of it being compromised through eavesdropping or cookie theft.

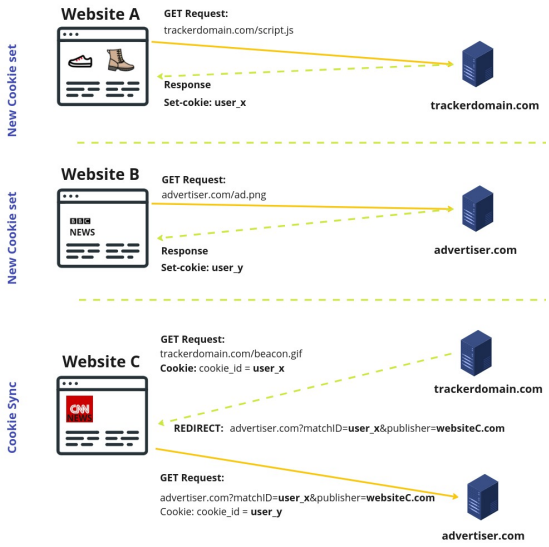


Figure 2.2: advertiser.com and trackerdomain.com synchronize their cookieIDs. Even without any code on Website C, advertiser.com finds out that: (i) cookieIDs $user_x$ and $user_y$ are the same, and (ii) $user_x$ recently visited the website. Finally, both domains can match user data directly between their servers.

2.2. Cookie Syncing

Cookie Synchronization is a method that effectively bypasses the same-origin policy, enabling web entities to match (synchronize) cookies and correlate the various IDs they assign to the same user across their web browsing activity [21]. To illustrate, let's imagine a scenario where a user is browsing different domains like websiteA.com and websiteB.com, each hosting third-party services like trackerdomain.com and advertiser.com as shown in Figure 2.2. These third parties can set their own cookies on the user's browser to recognize them later. For instance, trackerdomain.com knows the user as $user_x$, while advertiser.com identifies the same user as $user_y$.

Now, suppose the user visits a website, let's say websiteC.com, which only includes JavaScript code from trackerdomain.com and not advertiser.com. This means advertiser.com doesn't have visibility into visits to websiteC.com. However, when the trackerdomain.com code runs, it triggers a series of requests: first to trackerdomain.com (step 1), then a redirect to advertiserdomain.com (step 2), which prompts the user's browser to make another request to

advertiserdomain.com with specific information (step 3):

```
GET advertiser.com?syncID=user_x&publisher=websiteC.com
Cookie: {cookie_ID=user_y}
```

Upon receiving this request with the *user_y* cookie, advertiser.com learns that *user_y* visited websiteC.com. What's more concerning is that advertiser.com also deduces that *user_y* and *user_x* are actually the same user. This synchronization is known as cookie syncing and allows advertiser.com and trackerdomain.com to collaborate and achieve two goals: (i) identify users visiting websiteC.com and (ii) merge two different user identities (cookies) into one on the web. This matching process carries significant consequences, as it enables server-to-server data merges, potentially resulting in a more detailed behavioral profile of the user, as described in Section 2.3.1.

2.3. Online Behavioral Advertising

Online Behavioral Advertising (OBA), also known as interest-based advertising, is a form of marketing that targets advertisements with users' interests. Third-party cookies serve as the primary mechanism for enabling OBA, allowing companies to develop profiles based on individuals' interests [58]. Several sophisticated methods have been developed, often relying on complex analysis of users' data, to make advertising more effective; one such method is Real-Time Bidding (RTB), which is discussed in the following section.

2.3.1 Real-Time Bidding

RTB [93] represents a modern approach to ad serving, aiming to increase monetization in the online advertising market. When a user visits a website showing advertisements through RTB, the ad request is forwarded to an Ad Exchange. The Ad Exchange then disseminates this request, along with user data, to ad buyers, initiating an auction. These buyers participate in the auction, and the winning bidder gains the opportunity to serve ads to the user. To exchange user data between Ad Exchanges and buyers, cookie matching is used, which enables the correlation of cookies belonging to the same user across different domains.

RTB operates like a stock exchange [120], but instead of trading stocks, it deals in audiences for online advertisements. This mechanism benefits publishers by enabling them to sell their ad slots at the most competitive price, while also

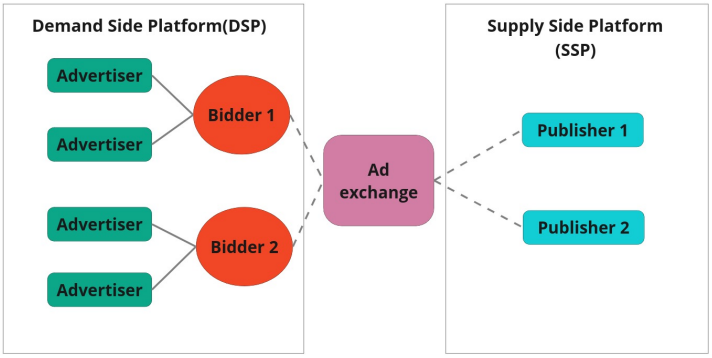


Figure 2.3: Ad Exchange model.

empowering bidders to dynamically adjust their buying strategies in real-time. The following is an explanation of DoubleClick's Ad Exchange [96]. In Real-Time Bidding (RTB), four key entities are involved as visualized in Figure 2.3:

1. Publishers (e.g., cnn.com): These entities own websites where ads are displayed.
2. Ad Exchanges (e.g., DoubleClick): Platforms facilitating advertising transactions between publishers, who sell ad slots, and bidders, who purchase these slots, utilizing real-time auctions.
3. Bidders (e.g., ads2.bid): These are typically large advertising agencies representing small and medium advertisers, participating in RTB auctions to secure ad spaces.
4. Advertisers (e.g., skyscanner.com): Entities seeking to promote and sell their products or services.

Whenever an ad appears on a website that a user visits, the Real-Time Bidding (RTB) process operates in the following manner:

1. When a user accesses a publisher's website within an Ad Exchange's advertising network, an HTTP request is sent to the Ad Exchange.
2. The Ad Exchange forwards bid requests for this ad impression to its pool of bidders.
3. Bidders analyze the impression and submit bid responses, containing the prices they're willing to pay and their ad snippets.

4. Bidders evaluate the impression and submit bid responses, including the prices they are prepared to pay and their ad snippets.

5. After a successful transaction, the Ad Exchange bills the winning bidder and compensates the publisher, deducting a commission before remitting payment.

Bid requests sent from Ad Exchanges to bidders typically contain details such as the Ad Exchange's user's cookie (or user's ID) and the visiting context of the user. This context encompasses information like the URL of the website being visited, the site's categories, the first three bytes of the user's IP address, and assorted browser data [94, 93].

Upon receipt of a bid request, the bidder identifies the user, employing Cookie Matching if this protocol has been previously executed. Subsequently, it establishes the bid price by considering the user's profile it holds and the user's context provided by the Ad Exchange. Bidders might also place bids on new users for whom they lack prior information.

2.3.2 Regulations for Online Behavioral Advertising

For advertisers, OBA is a crucial avenue for targeting specific audiences effectively. The online advertising sector is experiencing rapid growth, consistently achieving record-breaking revenues each year [52]. The industry argues that OBA enhances ad relevance and efficiency, therefore amplifying advertising effectiveness [15, 22]. However, the practice entails the collecting, using, and sharing of personal data, consequently raising concerns regarding consumer privacy. Consequently, OBA has gathered considerable attention from regulatory parties such as the U.S. Federal Trade Commission [43] and European Data Protection Authorities [12], as well as consumer advocacy groups. In response, industry coalitions such as the Digital Advertising Alliance[2] (DAA) in the U.S. and the European Interactive Digital Advertising Alliance[3] have established self-regulatory initiatives aimed at safeguarding consumer privacy and outlining protocols for informing consumers about data collection and usage.

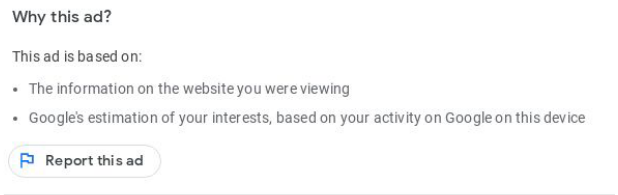
As part of its self-regulatory measures, the DAA introduced the AdChoices icon for websites in 2010 [5]. The AdChoices initiative aims to provide users with greater transparency and control over advertisements. Recognizable by its icon (Figure 2.4), which typically appears in the top right corner of ads from

[2]https://digitaladvertisingalliance.org/
[3]https://edaa.eu/

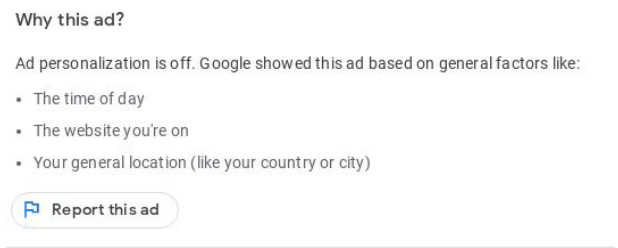


Figure 2.4: The AdChoices icon. The AdChoices icon represents an industry standard for disclosing online behavioral advertising practices. The figure is taken from [77].

participating companies, it serves as a link to additional information about the ad or the website's data collection practices (Figure 2.5). However, the Adchocies icon is found to be very ineffective, and it provides little control [69, 23].



(a) Targeted ad



(b) Non-targeted ad

Figure 2.5: Google's ad disclosure pages differentiate between targeted and non-targeted ads. The upper figure corresponds to a targeted ad, denoted by *Google's estimation of your interests*, whereas the lower one pertains to a non-targeted ad, identified by *Ad personalization is turned off*.

Despite AdChoices' partial coverage in terms of geography, participating companies, and types of advertising, past research indicates that over 60% of ads in a sample of 183 ads from leading news websites are covered by AdChoices according to a study by Storey et al. [109].

2.3.3 Our Contributions[4]

As explained in Section 2 digital advertising heavily relies on user data and consequently online businesses such as advertisers and marketers aggressively collect data to construct detailed user profiles. While online tracking and targeted advertising pose risks to users of all ages, children are particularly vulnerable. Children do not understand the implications of online tracking [122], yet they wield significant influence over their parents' purchasing decisions [62]. As a result, they become attractive targets for advertisers and marketers [62, 59], who exploit their susceptibility to persuasive advertising and exposure to potentially harmful content [100]. Despite the significant impact of online tracking on children, there has been limited research conducted in this area. To address this gap, we presented a measurement of tracking and (targeted) advertising on websites directed at children.

The lack of a comprehensive and updated list of children's websites poses a challenge, so we first built a multilingual classifier based on web page titles and descriptions. Applying this classifier to over two million pages from the Common Crawl dataset [20], we compiled a list of two thousand child-directed websites. By crawling these sites from multiple vantage points (Frankfurt, Amsterdam, London, San Francisco, and New York City), we measured the prevalence of trackers, fingerprinting scripts, and advertisements. Our crawler also detects ads on child-directed websites and determines if ad targeting is enabled by scraping ad disclosure pages by clicking the AdChoices button whenever available.

Our findings reveal that approximately 90% of child-directed websites embed one or more trackers, with around 27% containing targeted advertisements. Furthermore, we identify inappropriate ads on these websites by developing an ML pipeline that processes both images and text extracted from ads. This pipeline enables us to conduct semantic similarity queries for various search terms, exposing ads promoting services related to dating, weight loss, and mental health, as well as ads for sex toys and flirting chat services. Some of these ads feature repulsive, sexually explicit, and highly inappropriate imagery.

We publicly shared the source code of the crawler[5] that we used to collect the required data from child-directed websites.

[4]This section reuses text from our publication [77].

[5]`https://github.com/targeted-and-troublesome/targeted-and-troublesome-crawler`

2.4. Phasing out Third-Party Cookies

In response to rising privacy concerns and evolving regulations, there has been a collective effort within the tech industry to phase out third-party cookies. Several key stakeholders, including major web browsers, have begun taking action to reduce or completely stop using third-party cookies. The goal of eliminating these cookies is to improve user privacy, restrict cross-site tracking, and develop a more private online environment.

2.4.1 Initiatives by Major Web Browsers

Major web browser vendors have taken proactive measures to phase out or restrict the functionality of third-party cookies.

Google Chrome: Chrome, as part of its Privacy Sandbox [91] initiative, is gradually discontinuing third-party cookie support. This effort aims to enrich privacy-centric alternatives that enable personalized advertisements without cross-site tracking. The phase-out started in Q1 2024 and initially deactivated third-party cookies for 1% of users to make comprehensive testing easier [1].

In preparation for phasing out third-party cookies, Chrome introduced the Partitioned attribute as a component of Cookies Having an Independent Partitioned State (CHIPS) [17]. This feature makes cross-site access easier by using a unique cookie for every website. Chrome offers a new attribute within the Set-Cookie HTTP header known as “Partitioned” as shown below. When enabled, this attribute allows site owners to voluntarily adopt CHIPS for their websites.

Typically, the operation of third-party cookies is as follows, as shown in Figure 2.1: When a user accesses https://site-1.example, containing embedded content from 3rd-party.example, a cookie is set on the user’s device by 3rd-party.example. Subsequently, when the user visits https://site-2.example, which also includes content from 3rd-party.example, this second instance of 3rd-party.example retains access to the previously set cookie due to the traditional storage method, wherein cookies are associated with a key derived from the host or domain name of the originating site, known as the host key. In the described scenario, the cookie is stored with a key (“3rd-party.example”). However, with the introduction of the Partitioned attribute, third-party cookies are stored using two keys: the host key and a new partition key. The partition key is determined by the scheme and eTLD+1 (effective top-level domain plus one additional domain level) of the top-level URL. Upon setting the Partitioned attribute, if a user visits https://site-1.example, which embeds content from https://3rd-

party.example, and a cookie is set by https://3rd-party.example, it implies that the site owner has chosen to enable CHIPS. In this context, the storage key for the cookie becomes ("https://site-1.example"), ("3rd-party.example"). Consequently, when the user subsequently visits https://site-2.example, which also embeds content from https://3rd-party.example, this new embedded instance is unable to access the cookie since the partition key no longer matches.

```
Set-Cookie: __Host-example=34d8g; SameSite=None; Secure; Path=/;
    Partitioned;
```

Mozilla Firefox: Firefox conducted tests on third-party cookie blocking in 2013 [37, 39]. However, default-on third-party cookie blocking was not included in the release until 2018 [38], with the introduction of Enhanced Tracking Protection (ETP). Enhanced Tracking Protection restricts the functionality of third-party cookies by default by using Disconnect blocklist [31]. In 2022, Firefox introduced Total Cookie Protection (TCP). TCP functions by creating a distinct "cookie jar" for each website users visit. Instead of allowing trackers to correlate users' behavior across multiple sites, they only observe behavior on individual sites. Whenever a website or third-party content embedded within it places a cookie in the browser, that cookie is confined to the designated cookie jar for that specific website. Other websites cannot access cookies from jars that do not belong to them, thus granting users freedom from intrusive ads and reducing the amount of information collected about them by companies [42].

Apple Safari: Safari limits third-party cookies and cross-site tracking through its Intelligent Tracking Prevention (ITP) feature [119]. Pioneering the initiative in 2017, Safari became the first mainstream browser to default to blocking all third-party cookies via ITP. This feature uses machine learning to automatically identify third-party trackers [44].

2.5. The Economical Impact of Third-Party Cookie Deprecation

The deprecation of third-party cookies in Google Chrome, which holds a 64.73% market share [49], is pivotal, influencing the ad industry's trajectory. Google's Privacy Sandbox initiative, outlined in Section 3, proposes standards for replacing third-party cookies. GDPR has accelerated browsers' shift away from third-party cookies, although they can still be used with explicit user consent. Rather than simply complying with regulations, Google and other browsers are proactively phasing out third-party cookies.

A comprehensive 2023 study by Adobe [110] reveals that in the UK, 70% of marketing and customer experience leaders heavily rely on third-party cookies. Additionally, 41% allocate at least half of their marketing budgets to cookie-based tracking, with 59% planning to increase such spending in 2023. The anticipated end of third-party cookies is expected to adversely affect businesses, with 71% of business owners expressing concern. Among them, 10% fear devastating impacts, 21% anticipate significant harm, and 39% predict moderate negative consequences.

Changes in the digital advertising ecosystem aimed at safeguarding user data privacy could offer significant benefits to internet users. With the ability to opt in or opt out of behavioral tracking for mobile use, users gain more control over their data. Mandated disclosure requirements for personal data processing and use also enhance transparency.

The absence of targeted and retargeted advertising may result in less personalized ads as users traverse different websites. Instead of relevant ads targeted to their interests, users may encounter more generic or contextual advertising tied to the visited page rather than individual preferences.

Additionally, the discontinuation of third-party cookies raises concerns about the future availability of many free online content and service sites. These platforms often rely on targeted advertising for revenue, and without third-party cookies, their ability to offer free access may be compromised.

2.6. Tracking via Email Addresses

Protections against third-party cookies introduced by major browser vendors resulted in a growing demand for alternative tracking techniques. One such substitute is email-based tracking, which has been extensively used in a variety of online activities such as account registration, login, and newsletter subscriptions. In addition to their extensive usage of email addresses, they are very useful tracking identifiers, because they are unique and persistent.

It is interesting to note that Data brokers [40] and advertisers [6] have already taken advantage of the power of email addresses by using email hashes to identify users, monitor their activity on various devices, and link their offline and online activities.

2.6.1 Our Contributions

As we described in Section 2.6, email addresses became a new attractive target due to their extensive usage in webpages such as creating accounts, and subscribing to newsletters. We found that users' email addresses are exfiltrated to tracking, marketing, and analytics domains prior to form submission and without their consent. We crawled the top 100,000 websites from two different vantage points; EU and US. We showed that users' emails are exfiltrated to tracking related domains on 1,844 websites in the EU crawl and 2,950 websites in the US crawl. We additionally discovered incidental password collection on 52 websites due to session replay scripts. We also showed that Meta (formerly Facebook) and TikTok collect hashed personal information such as email and zip code from web forms, even if the user neither submits the form nor provides consent.

Even though most email addresses are exfiltrated to well-known tracking domains, we also found 41 tracker domains that are not included in any of the widely used blocklists. We already shared those domains with the major blocklist providers such as EasyList and EasyPrivacy. We also published the list of those domains[6] online.

Additionally, as part of our investigation, we reached out to both the websites where we detected email and password exfiltration and the domains receiving this information. Our outreach effort resulted in the tracker domains being removed from certain first-party websites after it was discovered that trackers were exfiltrating email addresses or passwords prior to form submission.

We developed a project website[7] where we provided all details about our study, including information about GDPR requests which we could not cover in the paper due to page limitations. We also publicly shared the source code of the crawler[8] that we used to collect data. Lastly, we published the dataset[9] of this study.

In response to the built-in protections implemented by browser vendors to prevent third-party cookies, advertisers and trackers started investigating more covert, stateless tracking techniques. One such method is browser fingerprinting, introduced next.

[6] `https://gist.github.com/asumansenol/51375ec3bfe49888c78a76100eb919c4`
[7] `https://homes.esat.kuleuven.be/~asenol/leaky-forms/`
[8] `https://github.com/leaky-forms/leaky-forms-crawler`
[9] `https://repository.ubn.ru.nl/handle/2066/283948`

2.7. Browser Fingerprinting

Browser fingerprinting is a stateless tracking method that uses unique characteristics of web browsers and devices that the browser exposes via HTTP headers and JavaScript APIs. Unlike traditional tracking methods, where data is stored on the user's device, browser fingerprinting does not require the tracker to store any client-side information.

Browser fingerprinting-based tracking mechanisms are considered more intrusive than cookie-based tracking for two primary reasons. Firstly, they are opaque to users. While cookies are visible in the browser and can be managed by users, browser fingerprinting operates by combining various features of the device's operating system and browser, making it difficult for users to identify and control. Secondly, browser fingerprinting is challenging to control because it cannot be easily removed or disabled by users. Since it does not rely on storing data on the user's device, traditional methods of managing tracking mechanisms, such as clearing cookies or adjusting browser settings, are ineffective against browser fingerprinting.

In 2009, Mayer first [71] introduced the concept of browser fingerprinting. With an experiment he carried out, he demonstrated how variations in browsing environments could be used by a remote server to identify users by gathering the content of the navigator, screen, navigator.plugins, and navigator.mimeTypes objects of browsers that connected to his experiment website. In this study, the author found that 1,278 (96.23%) of the 1,328 browsers could be uniquely identified.

A year later, Peter Eckersley conducted a large-scale measurement study, called Panopticlick, [36], by using a website where participants visit. In contrast to Mayer's study, Eckersley's study collected a significantly larger volume of fingerprints (470,161) by using a website where participants visited. Based on the data he collected, 83.6% of browser fingerprints were distinct using information from HTTP headers, JavaScript, and plugins like Flash or Java. If Flash or Java was enabled, this number rose to 94.2% as these plugins provided additional device information.

In privacy engineering, entropy serves as a crucial metric for quantifying the level of identifying information contained within a fingerprint. A higher entropy value indicates a more unique and identifiable fingerprint. Let X represent a discrete random variable with potential values $x_1, ..., x_n$ and $P(X)$ as its probability mass function, where H denotes the entropy. The entropy can be calculated using the following formula:

$$H(X) = -\sum_{i} P(x_i) log_b P(x_i)$$

In fingerprinting studies, Shannon's entropy is expressed in bits where $b = 2$. One bit of entropy reduces by half the probability of an event occurring.

While calculating the entropy, all browser features or specific subsets such as screen resolution or installed plugins can be used. Eckersley showed that plugins, system fonts, and the User-Agent rank as the three most distinguishing features with 15.4, 13.9, and 10.0 bits of entropy, respectively. This implies that, on average, only one in, respectively 32,768 (2^{15}), 16,384 (2^{14}), and 1,024 (2^{10}) other browsers is expected to share the same plugins, system fonts, or UA string as a particular browser.

In 2014, Laperdrix et al. launched the AmIUnique.org website to collect browser fingerprints and conducted an analysis of their uniqueness [61]. In 2016, they completed an analysis on 118,934 fingerprints, confirming Eckersley's earlier findings that 89.4% of collected fingerprints were unique [61]. Over the six years between both studies, they observed an evolution in fingerprint attributes. For instance, plugins were one of the most distinguishing browser features, but they were deprecated due to security threats [85, 84]. The study found that canvas fingerprinting [78] is one of the most distinguishing attributes. Canvas fingerprinting involves creating an invisible image on a canvas and then retrieving the rendered image data. The precise rendering of this image depends on various factors, including the operating system, installed fonts, font library, graphics card, graphics driver, and browser type. Furthermore, their study demonstrated that browser fingerprinting, which is primarily reliant on HTTP headers and HTML5 canvas fingerprinting, is just as effective on mobile devices as it is on desktop, with 81% of fingerprints on mobile devices being unique.

Data used in Eckersley's and Laperdrix's studies was collected from participants visiting the websites of AmIUnique and Panopticlick webpages, respectively. This approach could lead to bias since a large proportion of individuals with technical knowledge of browser fingerprinting might participate in the data collection process. As opposed to these studies Boix et al. collected 2,067,942 browser fingerprints from one of the top 15 French websites after installing browser fingerprinting code [50]. It was discovered that just 33.6% of the devices in the sample had unique fingerprints. An overview of these three studies is summarized in the Table 2.1 and Table 2.2.

Table 2.1: Summary of large-scale studies conducted on browser fingerprinting over the years.

	Panopticlick [36] (2010)	**AmIUnique [61] (2016)**		**Hiding in the Crowd [50] (2018)**	
	Desktop	**Desktop**	**Mobile**	**Desktop**	**Mobile**
Number of fingerprints	470,161	105,829	13,105	1,816,776	251,166
% of Unique fingerprints	94.2%	89.4%	81%	35.7%	18.5%

Table 2.2: Shannon's entropy for all attributes from Panopticlick, AmIUnique, and Hiding in the Crowd studies. H_M is the worst-case scenario where the entropy is maximum and all values of an attribute are unique.

	Panopticlick (2010)		**AmIUnique (2016)**		**Hiding in the Crowd (2018)**	
Attribute	**Entropy**	**Norm.**	**Entropy**	**Norm.**	**Entropy**	**Norm.**
Platform	-	-	2.310	0.137	1.200	0.057
Do Not Track	-	-	0.944	0.056	1.919	0.091
Timezone	3.040	0.161	3.338	0.198	0.164	0.008
List of plugins	15.400	0.817	11.060	0.656	9.485	0.452
Use of local/session storage	-	-	0.405	0.024	0.043	0.002
Use of an ad blocker	-	-	0.995	0.059	0.045	0.002
WebGL Vendor	-	-	2.141	0.127	2.282	0.109
WebGL Renderer	-	-	3.406	0.202	5.541	0.264
Available fonts	13.900	0.738	8.379	0.497	6.904	0.329
Canvas	-	-	8.278	0.491	8.546	0.407
Header Accept	-	-	1.383	0.082	0.729	0.035
Content encoding	-	-	1.534	0.091	0.382	0.018
Content language	-	-	5.918	0.351	2.716	0.129
User-agent	10.000	0.531	9.779	0.580	7.150	0.341
Screen resolution	4.830	0.256	4.889	0.290	4.847	0.231
List of HTTP headers	-	-	4.198	0.249	1.783	0.085
Cookies enabled	0.353	0.019	0.253	0.015	0.000	0.000
H_M **(worst scenario)**	18.843		16.860		20.980	
Number of FPs	470,161		118,934		2,067,942	

2.7.1 Active and Passive Browser Fingerprinting

Browser fingerprinting can be classified as passive or active depending on how the data is collected. Active fingerprinting involves a fingerprinting application actively running code to collect data for constructing a fingerprint. Modern web browsers provide Web APIs that enable website functionalities similar to native applications, enabling developers to access the details about users' devices. For instance, the `Navigator` [73] interface is often misused for collecting information regarding the properties of the user agent executing the script. Similarly, the `Screen` [74] interface provides details about the screen where the current window is displayed. Another example involves the use of HTML5 Canvas API for

fingerprinting, used for canvas fingerprinting–a technique first introduced by Mowery and Shacham in 2012 [78]. Canvas fingerprinting functions by drawing an invisible image on the canvas and subsequently analyzing the rendered image data. Even slight variations in the rendered image may help distinguish devices. Acar et al.'s research in 2014 showed that it was present on 5% of the websites by crawling the homepages of the top 100,000 Alexa-ranked sites [2].

In contrast to active fingerprinting, passive browser fingerprinting does not necessitate executing any client-side code; instead, it relies only on monitoring incoming network traffic to gather the necessary data [53]. The information accessible to a passive fingerprinter encompasses the user-agent string, IP address, Accept headers, clock skew [57], and protocol quirks [121]. Through passive fingerprinting techniques, a third-party tracker can potentially correlate a user's web visits without running any client-side code, making the detection of passive fingerprinting challenging for users or web privacy measurement studies.

2.7.2 Our Contributions

The user-agent string sent by browsers comprises information about a user's device, browser, and platform. Previous research on browser fingerprinting highlighted the risk of covert fingerprinting based on user-agent strings [61, 36]. The user-agent string ranks as the third most distinctive browser attribute, as discussed in Section 2.7. Browsers, including Chrome, have reduced the amount of identifying information in user-agent strings in an effort to enhance user privacy. Chrome has also added new JavaScript API and high-entropy user-agent client hints (UA-CH) to enable access to the redacted browser information. Using an instrumented crawler, we measured access to high-entropy browser features via the JavaScript API and UA-CH HTTP headers in order to assess the impact of these modifications on the top 100,000 websites in the the Chrome User Experience Report (CrUX) list [24].

We measured access delegation to third parties and examined if tracking, advertising, and browser fingerprinting scripts were already using the new client hints. The findings of our study indicate that 59.2% of the successfully visited websites have one or more scripts accessing high-entropy UA-CHs. Among these script calls, 93.8% were related to tracking and advertising and were mostly owned by Google. We showed that scripts from approximately 9K distinct registrable (eTLD+1) third-party domains exploit their unrestricted access to obtain the high-entropy UA-CHs.

We found that at least one tracker script exfiltrates the high-entropy hints to a remote server on 91.6% of the sites where the high-entropy client hints are

accessed via the JavaScript API. We found that high-entropy UA-CHs usage through HTTP headers which require opt-in or delegation is very limited. Just 1.3% of websites receive high-entropy UA-CHs via the Accept-CH header, and even fewer (0.4%) websites give high-entropy hints to external domains. All things considered, our research suggests that although efforts to minimize the passive collection of identifying browser features through user-agent reduction were successful, third-party tracking and advertising scripts have unrestricted access to high-entropy UA-CHs.

We set up a project website[10] and published the source of the crawler[11] used in the study. Lastly, we shared the data[12] publicly.

2.7.3 Other Uses of Browser Fingerprinting

Browser fingerprinting has drawn attention lately due to its impact on user privacy [13]. Some browsers have attempted to address this by limiting access to certain APIs. For instance, Tor Browser blocks Canvas and WebRTC, commonly used by fingerprinting scripts [112]. Firefox also has tried to prevent fingerprinting by blocking network requests made to download fingerprinting scripts [79]. However, it is important to note that browser fingerprinting is also used for legitimate purposes such as providing an additional security layer against service abuses such as unauthorized login attempts.

Anti-fraud protection: When using browser fingerprinting for anti-fraud services, the initial step involves assigning a risk level to a browser identified by its fingerprint. This assessment is determined by evaluating and contrasting the typical context used by the user. Subsequently, this risk level is relayed to the service's customer (e.g., shopping website), enabling informed decision-making by the website operator (for instance, requesting further verification in the case of a high-risk browser).

Anti-robots protection: The 2023 Imperva Bad Bot Report revealed that nearly half (47.4%) of internet traffic came from bots. According to the report, bad bots are responsible for 30% of automated traffic, engaging in malicious activities such as unauthorized data scraping [45]. Good bots, including search engines such as Google and Bing, form 17.4% of the automated traffic, used for indexing web content. Additionally, the fingerprints of robots operating in headless browsers allowed for their identification as observed in Vastel et al.'s study, where 31.96% of websites employed fingerprinting techniques to detect web crawlers [117].

[10] `https://homes.esat.kuleuven.be/~asenol/ua-reduction`
[11] `https://github.com/ua-reduction/ua-client-hints-crawler`
[12] `https://repository.ubn.ru.nl/handle/2066/299544`

Authentication: Browser fingerprints and biometric authentication factors share the similarities of extracting unique identifiers from entities (individuals or browsers) for identification or authentication purposes. Some studies focused on designing authentication mechanisms using browser fingerprints [116, 60, 54], Andriamilanto et al. [10] explored the potential usage of browser fingerprints for web authentication. They assessed various characteristics in a dataset of 4,145,408 fingerprints from 1,989,365 browsers, considering aspects similar to biometric authentication factors. Their findings demonstrated that over six months, the browser fingerprints maintained a uniqueness rate above 81.3%, with 94.7% of fingerprints being shared by a maximum of eight browsers. The study suggests that browser fingerprints, composed of diverse browser attributes have the potential to strengthen web authentication systems [9].

Restrictions such as blocking the usage of certain APIs and scripts can break websites that rely on these APIs for legitimate purposes. Certain browsers have been cautious about implementing broad API restrictions and blocking requests due to concerns about website malfunction. For example, Safari [101] and Brave [16] return simplified system configurations to trackers, making more devices appear identical and thus harder to differentiate. However, blocking fingerprinting scripts or limiting the information exposed by some fingerprinting surfaces such as APIs or IP addresses might negatively affect website functionality.

2.7.4 Our Contributions

To assess the use of browser fingerprinting for improving online security, particularly as an extra defense against service abuses such as credit card fraud or during user authentication, we conducted a large-scale measurement study. The study aims to determine how frequently fingerprinting scripts appear on sign-up and login pages in comparison to other website pages. In addition to web tracking, the other primary use of browser fingerprinting is authentication as explained in Section 2.7.3, which is why we selected authentication pages for our comparative study. Nevertheless, there is a lack of an accurate detection technique to find these pages. The identification and analysis of authentication pages have been studied before [95, 27, 115, 32, 123, 46, 56, 70, 7, 34] and most of the methods either used heuristics based on regex patterns [27, 115, 32, 123, 46, 56], or manual inspection [95, 34]. In particular, translations to other languages and variations on the terms “login” and “sign-up” are frequently included in these regex patterns. Nevertheless, complicated authentication flows, like multi-step login flows which is a very common design chosen by companies such as Twitter and Google, are frequently missed by these heuristics.

To recognize authentication pages automatically, we created *a novel ML model* rather than depending on regex patterns and heuristics. Our supervised model performs well on a large, manually-labeled dataset of 10,000 popular sites, achieving 96-98% precision and recall. Additionally, we examined previous approaches for identifying login signup pages and compared our ML model with other approaches from earlier studies. Using the same dataset, we demonstrated that our ML model performs noticeably better than all of them. Our findings demonstrate that while fingerprinting rates are similar, home pages and authentication pages differ in the third-party scripts they include and how frequently these scripts are classified as tracking. Specifically, fingerprinting scripts are slightly more common on login and sign-up pages (9.2%) than they are on other pages (8.9%). While we did find fingerprints on 9.2% of authentication sites, the frequency varies significantly according to how popular the website is. In the top 1K sites, about 15% of login pages use fingerprinting; however, when the rank is between 1K and 5K, this percentage drops to 8%.

Our ML model[13], a browser extension[14] for identifying and displaying login and sign-up pages, along with a web crawler[15] for detecting such pages in given URLs are publicly accessible.

[13] `https://github.com/asumansenol/double_edged_sword_data`
[14] `https://github.com/asumansenol/login_signup_detection_chrome_extension`
[15] `https://github.com/asumansenol/double_edged_sword_crawler`

Chapter 3

Countermeasures Against Online Tracking

3.1. Defenses Developed by Major Browser Vendors

3.1.1 Google

Google's Chrome browser holds a dominant market share, estimated at over 65.8% as of 2024 [49]. Concurrently, Google is a major player in online advertising and tracking, collecting extensive user data through services such as Google Ads [28]. This dual role raises concerns about conflicting interests. While it aims to provide a seamless browsing experience through Chrome, its advertising business relies on tracking user behavior. To balance effective advertising with consumer privacy, Google initiated the Privacy Sandbox Initiative [91]. Their stated objective is not to block user tracking completely but rather to substitute current technologies with alternatives likely to be perceived as less intrusive in terms of user privacy. Thus far, Chrome has put forward over 30 proposals, addressing various use cases and requirements within the Privacy Sandbox framework. These key proposals are categorized based on their aims as follows:

- Fight spam and fraud on the web
- Show relevant content and ads
- Measure digital ads

- Strengthen cross-site privacy boundaries
- Limit covert tracking

Within those categories, numerous projects were initiated. This thesis only reviews a subset of them.

Third-party cookie deprecation: Technology experts have been discussing this widely for several years to prevent cross-site tracking. Chrome is gradually phasing out support for third-party cookies. The phased approach started in Q1 2024, initially disabling third-party cookies for 1% of users to facilitate testing [18].

User Agent Reduction and Client Hints[1]: As part of the Privacy Sandbox initiative, Chrome has redacted identifiable data available within the User-Agent string–an HTTP header that could potentially be used for passive fingerprinting [114]. Chrome has recently implemented high-entropy user-agent client hints (UA-CH) and a new JavaScript API enabling access to specific browser details. To evaluate the effects of these changes on online tracking practices, we developed an instrumented crawler that can measure the collection of newly introduced high-entropy features via UA-CH HTTP headers and the JavaScript API on the top 100,000 websites [102]. We also explored whether tracking, advertising, and browser fingerprinting scripts have started using these new client hints and the JavaScript API.

Privacy Budget [90]: This proposal outlines a restriction on the amount of personal data made available to websites, aiming to make it insufficient for tracking and identification of individuals. Analogous to a monetary budget, the privacy budget operates as a limit on the amount of data available. However, unlike dollars, the currency within a privacy budget is expressed in bits. For instance, when a Privacy Budget is implemented, it restricts the volume of information accessible to a website or technology based on the assigned bit value of each data piece. For example, if a browser is identified to use the Italian language, this language setting would be considered a distinctive attribute and assigned a specific bit value. Once a predefined threshold of bits or distinguishing browser characteristics is reached, the privacy budget is exhausted.

This proposal outlines a restriction on the amount of personal data made available to websites, aiming to make it insufficient for tracking and identification of individuals. Analogous to a monetary budget, the privacy budget operates as a limit on the amount of data available. However, unlike dollars, the currency within a privacy budget is expressed in bits. For instance, when a Privacy Budget is implemented, it restricts the volume of information accessible to a website or

[1]This section reuses text from our publication [102].

technology based on the assigned bit value of each data piece. For example, if a browser is identified to use the Italian language, this language setting would be considered a distinctive attribute and assigned a specific bit value. Once a predefined threshold of bits or distinguishing browser characteristics is reached, the privacy budget is exhausted.

Mozilla expressed their concerns about the Privacy Budget proposal in a document [97]. They outlined their criticisms, including the difficulty of estimating the amount of information on a single surface, and cautioned that any design where queries made after exceeding the budget fail could lead to unpredictable breakage and compatibility issues with websites, making site development significantly more challenging. As of January 2024, this proposal is no longer active [118].

Topics API: Topics API uses interest-based advertising (IBA) without relying on tracking the user's browsing history. IBA is a central concept within the Topics API, that enables personalized advertising based on user's interests, inferred from their recent browsing activities. This diverges from contextual advertising, which targets ads based on the content of the current page. Basically with Topics API, the browser tracks and logs topics that seem relevant to them based on what the user browses. First, this information is stored in the users' device and then the user's topics of interest can be accessed by API callers (such as ad tech platforms) using the Topics API. All processes occur without disclosing further details about the user's browsing history.

On the other hand, the IAB Tech Lab's recent report [26] outlines concerns regarding substantial development costs, infrastructure changes, and operational challenges for both advertisers and media companies. According to the report, the key issue was the abandonment of event-based metrics, significant changes to programmatic advertising ecosystems, and legal and business implications.

3.1.2 Apple

In 2019, Safari introduced Intelligent Tracking Prevention [101]. This feature identifies domains that can track users online using a machine learning classifier.

When Intelligent Tracking Prevention detects a domain with cross-site tracking capabilities, it automatically blocks the cookies that are created by that domain.

In addition, Safari not only blocks third-party cookies but also blocks browser fingerprinting. In order to block browser fingerprinting, Safari returns a simplified system configuration to trackers, ensuring that more devices look identical, thereby making it challenging to single one out. This is accomplished

by spoofing the return of certain APIs such as Canvas. However, Safari's fingerprinting measures have weak spots. Doug Leith found that Safari's welcome page could leak information to third-party applications, which might then load user identifiers into the browser cache, as detailed in his study [68]. Moreover, there's a potential for Apple's iCloud processes to interact with identifiable user data, possibly for debugging purposes. Depending on how the data is distributed, these data points could become part of the browser fingerprint.

Moreover, Safari includes Private Click Measurement, an inventive approach to measuring ad clicks [101]. This method effectively prevents cross-site tracking while enabling advertisers to evaluate their web campaigns' efficacy. Since it is integrated into the browser and operates on the device, it is hidden from merchants, advertisers, and Apple, making it impossible for them to see which ads are clicked or what purchases are made. With this solution, consumers can not be tracked as they click on ads and make purchases in Safari without having to put their trust in any of the parties involved. Additionally, Apple has submitted Private Click Measurement as a new web standard to the World Wide Web Consortium [87].

3.1.3 Mozilla

With Enhanced Tracking Protection [79], Firefox protects users' privacy by blocking trackers that follow users around online to collect information about their browsing habits and interests. It also includes protections against harmful scripts, such as malware that drains the battery. Firefox uses a list of known trackers provided by Disconnect [31]. Firefox by default prohibits the following types of scripts and trackers: social media trackers, cryptocurrency miners, fingerprinters, cross-site tracking cookies, tracking content in ads, videos, and other in-page content.

In addition, Mozilla introduced Total Cookie Protection [81] to prevent cross-site tracking cookies. For blocking fingerprinting, again Firefox uses a strategy of blocking third-party requests to domains flagged by Disconnect as fingerprinters [80]. While this blocklist-based protection offers advantages, such as providing less website breakage by analyzing scripts before labeling them as trackers, it also presents disadvantages. Maintaining these lists requires manual and constant effort, and their effectiveness depends on whether the domain is labeled correctly [83, 106]. For instance, if Disconnect's fingerprinting detection method does not cover all potential fingerprinting techniques, the list may not be comprehensive. Further details on list-based blockage are discussed in Section 3.2.1.

3.2. Browser Add-ons

Since the late 1990s,[2] web browsers have supported extensions which are user-developed software components that function within a web browser. Privacy Badger [88], Disconnect [30], Ghostery[3] [47], uBlock Origin[3] [113] , AdBlock [3] and AdBlock Plus [4] are the most used browser extensions to protect users' privacy according to a past study [72]. These ad blockers use different techniques, and each has pros and cons.

3.2.1 Using a Blocklist to Block Known Trackers

This method aims to block all requests to domains that are labeled as trackers. There are numerous blocklists available, and extensions have the option to add to them or make their own. Disconnect, AdBlock Plus, and uBlock Origin use publicly available blocklists, whereas Ghostery is not open source but claims to have the largest tracker database of all the privacy tools offered.

There are certain disadvantages of blocking requests to tracker domains by using blocklists. These lists may include false positives or false negatives, where entities labeled as trackers may not be, and those labeled as non-trackers may actually be trackers. Also, some trackers are looking for ways to evade blocklists. Dimova et al. showed that certain trackers use a CNAME record on a subdomain, making it the same site as the hosting website [29]. Moreover, since custom subdomains are often used, they are less likely to be included in block lists. This makes it challenging to effectively block all tracking scripts based on only blocklists.

Blocklists have the additional drawback of potentially blocking domains that serve purposes beyond tracking. In this case, users would have to decide whether to keep the functionality and allow the tracking or to lose the functionality and prevent tracking.

Additionally, keeping blocklists up to date poses a significant challenge. With new tracker domains emerging regularly, maintaining and updating these lists by crawling domains and collecting information becomes a challenging task.

[2]Internet Explorer: 1999, Firefox: 2004, Chrome: 2010
[3]Inactive due to deprecated Web Request API

3.2.2 Using Algorithms to Identify Trackers

In this approach, any domain identified as a tracker by the algorithm gets blocked. The key advantage here is that if the algorithm functions accurately, it automatically blocks trackers without relying on an updated blocklist. This eliminates the necessity of constantly maintaining and updating a blocklist.

The algorithmic approach to blocking trackers shares similar limitations to using blocklists. Users rely on the algorithm's accuracy not only in recognizing trackers correctly but also in distinguishing non-trackers accurately. If the algorithm ends up blocking functional content, users can create exceptions. However, excessive reliance on exceptions can significantly diminish usability as users might need to add numerous exceptions to ensure websites function properly. Furthermore, there's a risk that an algorithm might fail to identify trackers as such, inadvertently permitting tracking activity, which is an undesirable outcome.

Privacy Badger uses a heuristic for deciding whether a third party is tracking users [89]. Alternatively, other browser extensions adopt a hybrid approach, combining both blocklists and heuristic algorithms. For instance, Ghostery uses two primary methods - blocklists and heuristic algorithms - to block tracker domains within network traffic [48].

3.2.3 Our Contributions

We developed Leak Inspector which is a *a proof-of-concept browser extension* that warns and protects users against personal data exfiltration from online forms. This tool aims to help both publishers and end users evaluate third-party entities that collect personal information from web forms without users' explicit consent or knowledge. The extension offers three key functionalities:

Firstly, it can block requests containing extracted personal data from online forms and highlight related form fields by displaying an add-on icon as shown in Figure 3.1.

Secondly, it logs technical details of identified sniff and leak attempts to the console, facilitating technical audits. This logged information includes the value and XPath of the sniffed input element, the source of the sniffing script, and details of the leaky request, such as the URL and POST data.

Lastly, it provides a user interface listing recent sniff and leak attempts, along with details about the tracker domain, company, and tracker category. The user

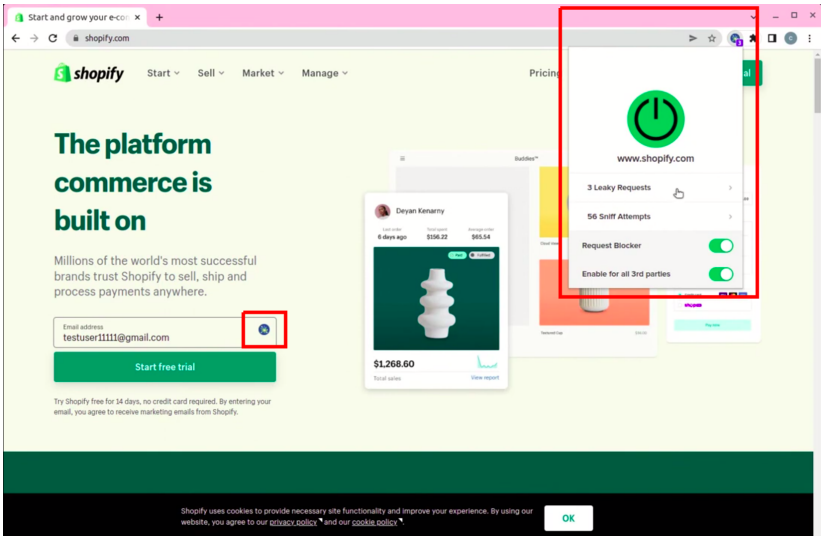


Figure 3.1: The LeakInspector user interface (UI) functions as follows: When a tracker script scans ("sniffs") an input field containing personal information, LeakInspector identifies and highlights the input field. It then proceeds to block any leaky requests containing this personal data. Within the UI, users can view recent sniff and leak attempts, along with details such as the tracker domain, company, and tracker category.

interface is built on the code adapted from DuckDuckGo's Privacy Essentials extension [33].

The source code of this browser add-on is publicly available on the project's GitHub repository[4] and it has more than 120 stars.

[4] `https://github.com/leaky-forms/leak-inspector`

Chapter 4

Conclusion and Future Research Directions

4.1. Conclusion

This dissertation investigated the complex and pervasive nature of web tracking by revealing how users' personal data are exfiltrated to tracker domains on thousands of websites. We started our work by investigating online forms including email and password fields. By developing a web crawler, we detected those fields and filled them on a large scale. Our analysis showed that users' emails and passwords are exfiltrated to tracking related domains before form submission and without users' consent and knowledge.

Furthermore, our research highlights how important it is to conduct empirical measurements of online tracking. To reach more people, the availability of measurement tools should extend beyond academic circles. We have contributed to this accessibility by publicly sharing the source code of our crawlers, other developed tools, and datasets. Furthermore, our work has already had a significant influence on a large number of internet users through articles in various media outlets including WIRED [67], Le Monde [65], Fortune [64] and Nos.nl [66]. Also, we believe that our disclosure emails led to the removal of third parties leaking personal information from websites.

With this dissertation, we presented an overview of HTTP cookies, explaining their various functions in improving user experiences while also outlining possible privacy risks, especially with regard to third-party cookies.

We then discussed the industry-wide initiatives to phase out third-party cookies and emphasized the countermeasures introduced by popular web browsers such as Apple Safari, Mozilla Firefox, and Google Chrome.

Further, we examined the recent browser behavior changes related to user-agent reduction and Client Hints. With this change, Chrome introduced a new browser API to expose the reduced information. Our research showed that creates a cat-and-mouse game; we found that at least one tracker script exfiltrates the high-entropy hints to a remote server on 91.6% of the sites where the high-entropy client hints are accessed via the JavaScript API.

Furthermore, we noticed alarming patterns by focusing on tracking and advertising practices on children's websites. With large-scale web crawls, we found that over 90% of child-directed websites had one or more trackers embedded, and about 27% had targeted ads. Upon analyzing those ads, we discovered that they promote dating, weight loss, and mental health services in addition to sex toys and flirting chat services. We shared the findings with companies and data protection organizations to solve these concerns.

Lastly, we developed a machine-learning model to identify login and signup pages. By quantifying the prevalence of fingerprinting scripts on these authentication pages, we observed fingerprinting on 9.2% of authentication pages overall, with significant variation based on website popularity. Approximately 15% of login pages among the top 1,000 sites engage in fingerprinting, compared to 8% of login pages for sites ranked between 1,000 and 5,000. Presenting the usage of browser fingerprinting for improving security on a large scale reveals the complex duality: while it aims to protect user security, it also violates user privacy. This dual nature of fingerprinting necessitates a comprehensive approach to mitigate privacy risks. For instance, simple prevention methods such as blocking fingerprinting scripts can inadvertently block legitimate uses such as authentication factors, necessitating alternative solutions that may involve applying exceptions for those pages.

With our research, we contribute to highlighting the necessity of comprehensive regulations and powerful privacy-enhanced technologies to protect user data. A more open, safe, and ethical online environment depends on the continuous effort to strike a balance between user privacy and personalization in the digital realm.

4.2. Future Research Directions

We identify a number of possible directions for future work.

Detecting personal data leakage from online forms using JavaScript information flow tracking: To identify leaks, we employed a technique from a prior study that involves searching for multi-layered encodings and hashes (e.g., Base64, SHA-256) of search strings within various components such as HTTP request URLs, headers, and POST data [41]. However, this approach has limitations as it is nearly impossible to cover all encodings and hash functions, and it may not handle custom encryption or encoding methods. Furthermore, our study illustrated an instance where a third party hashed email addresses using a fixed salt hardcoded within their script. Information flow tracking involves monitoring the movement of data across different variables, functions, and processes within the JavaScript code, observing the paths of entered personal data or sensitive information as it undergoes processing. By tracking this flow, the system identifies cases where personal data from online forms is shared or transmitted to external entities, such as third-party servers or domains, without user consent or knowledge.

Minimizing tracking in the browser: Currently web browsers only impose a few restrictions on web trackers and advertisers. However, browsers should protect users against these. Similar to our development of the LeakInspector extension, browsers can limit the information obtained from web forms on sites. For instance, highly sensitive user information such as passwords or credit card information could be restricted, permitting sharing only with the first-party entity.

Examination of fingerprinting scripts to understand real intent: We studied the attempts made by fingerprinting scripts on authentication pages, such as login and signup pages. We assessed the frequency of fingerprint collection on these pages in contrast to home pages. Our observations highlighted differences in the inclusion of third-party scripts between home pages and authentication pages. Investigating other types of fingerprinting scripts designed for non-tracking purposes could be a promising avenue for future research.

Bibliography

[1] Prepare for third-party cookie restrictions. https://developers.google.com/privacy-sandbox/3pcd. [Online; accessed 15. Mar. 2024].

[2] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. The web never forgets: Persistent tracking mechanisms in the wild. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 674–689, 2014.

[3] AdBlock – best ad blocker. https://chromewebstore.google.com/detail/adblock-%E2%80%94-best-ad-blocker/gighmmpiobklfepjocnamgkkbiglidom, 2009. [Online; accessed 11. May. 2023].

[4] AdBlock Plus – free ad blocker. https://chromewebstore.google.com/detail/adblock-plus-free-ad-bloc/cfhdojbkjhnklbpkdaibdccddilifddb, 2006. [Online; accessed 11. May. 2023].

[5] "AdChoices" Makes it Easier to Tell Your Brand's Privacy Story – Through the Power of One Icon. https://digitaladvertisingalliance.org/blog/%E2%80%98adchoices%E2%80%99-makes-it-easier-tell-your-brand%E2%80%99s-privacy-story-%E2%80%93-through-power-one-icon, 2023. [Online; accessed 01. Mar. 2024].

[6] Advanced Matching. https://developers.facebook.com/docs/meta-pixel/advanced/advanced-matching/. [Online; accessed 25. Feb. 2024].

[7] Suood Al Roomi and Frank Li. A Large–Scale Measurement of Website Login Policies. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 2061–2078, 2023.

[8] Athanasios Andreou, Giridhari Venkatadri, Oana Goga, Krishna P Gummadi, Patrick Loiseau, and Alan Mislove. Investigating ad transparency mechanisms in social media: A case study of Facebook's

explanations. In *NDSS–Network and Distributed System Security Symposium*, pages 1–15, 2018.

[9] Nampoina Andriamilanto, Tristan Allard, Gaëtan Le Guelvouit, and Alexandre Garel. A large-scale empirical analysis of browser fingerprints properties for web authentication. *ACM Transactions on the Web (TWEB)*, 16(1):1–62, 2021.

[10] Tompoariniaina Nampoina Andriamilanto. *Leveraging browser fingerprinting for web authentication*. PhD thesis, Université Rennes 1, 2020.

[11] Mattu Surya Angwin Julia and Parris Terry. Facebook Doesn't Tell Users Everything It Really Knows About Them. `https://www.propublica.org/article/facebook-doesnt-tell-users-everything-it-really-knows-about-them`, 2016. [Online; accessed 21. Feb. 2024].

[12] ARTICLE 29 Data Protection Working Party Adopted on 13 July 2010. `https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2010/wp174_en.pdf`. [Online; accessed 12. Mar. 2023].

[13] ARTICLE 29 Data Protection Working Party Adopted on 10 April 2014. `https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf`, 2023. [Online; accessed 15. Mar. 2024].

[14] Pouneh Nikkhah Bahrami, Umar Iqbal, and Zubair Shafiq. FP-Radar: Longitudinal Measurement and Early Detection of Browser Fingerprinting. *arXiv preprint arXiv:2112.01662*, 2021.

[15] Howard Beales. The value of behavioral targeting. *Network Advertising Initiative*, 1(2010), 2010.

[16] Brave browser simplifies its fingerprinting protections. `https://brave.com/privacy-updates/28-sunsetting-strict-fingerprinting-mode/`. [Online; accessed 25. Feb. 2024].

[17] Cookies Having Independent Partitioned State. `https://developers.google.com/privacy-sandbox/3pcd/chips`. [Online; accessed 25. Feb. 2024].

[18] Prepare for phasing out third-party cookies – Chrome Developers. `https://developer.chrome.com/en/docs/privacy-sandbox/third-party-cookie-phase-out`, 2023. [Online; accessed 28. Jul. 2023].

[19] Inria and the CNIL award the 2022 Privacy Protection Prize. https://www.cnil.fr/en/inria-and-cnil-award-2022-privacy-protection-prize. [Online; accessed 25. Feb. 2024].

[20] June/July 2022 crawl archive now available – Common Crawl. https://commoncrawl.org/2022/07/june-july-2022-crawl-archive-now-available, 2023. [Online; accessed 28 Feb. 2023].

[21] Cookie Matching. https://developers.google.com/authorized-buyers/rtb/cookie-guide, 2023. [Online; accessed 28 Feb. 2023].

[22] Lorrie Faith Cranor. Can users control online behavioral advertising effectively? *IEEE Security & Privacy*, 10(2):93–96, 2012.

[23] Lorrie Faith Cranor, Hana Habib, Yixin Zou, Alessandro Acquisti, Joel Reidenberg, Norman Sadeh, and Florian Schaub. Design and evaluation of a usable icon and tagline to signal an opt-out of the sale of personal information as required by ccpa. *Retrieved September 13th*, 2020.

[24] Adding Rank Magnitude to the CrUX Report in BigQuery. https://developers.google.com/web/updates/2021/03/crux-rank-magnitude, 2023. [Online; accessed 22. Feb. 2024].

[25] Mattioli Dana. On Orbitz, Mac Users Steered to Pricier Hotels. https://www.wsj.com/articles/SB10001424052702304458604577488822667325882, 2012. [Online; accessed 22. Feb. 2024].

[26] Kim Davis. The Impact of Big Data Decisions on Business Valuations. https://martech.org/iab-slams-google-privacy-sandbox/, 2024. [Online; accessed 18. Mar. 2024].

[27] Joe DeBlasio, Stefan Savage, Geoffrey M. Voelker, and Alex C. Snoeren. Tripwire: Inferring internet site compromise. In *Proceedings of the 2017 Internet Measurement Conference*, pages 341–354, 2017.

[28] Digital ad revenue share by company 2023 | Statista. https://www.statista.com/statistics/290629/digital-ad-revenue-share-of-major-ad-selling-companies-worldwide, 2023. [Online; accessed 3 Aug. 2023].

[29] Yana Dimova, Gunes Acar, Lukasz Olejnik, Wouter Joosen, and Tom Van Goethem. The CNAME of the Game: Large-scale Analysis of DNS-based Tracking Evasion, 2021-07-12.

[30] Disconnect. https://chromewebstore.google.com/detail/disconnect/jeoacafpbcihiomhlakheieifhpjdfeo, 2014. [Online; accessed 11. May. 2023].

[31] Disconnect. Disconnect Tracking Protection. https://github.com/disconnectme/disconnect-tracking-protection, 2023. [Online; accessed 11. Jun. 2023].

[32] Kostas Drakonakis, Sotiris Ioannidis, and Jason Polakis. The Cookie Hunter: Automated Black-Box Auditing for Web Authentication and Authorization Flaws. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, pages 1953–1970, New York, NY, USA, 2020. Association for Computing Machinery.

[33] DuckDuckGo Privacy Essentials. https://chromewebstore.google.com/detail/duckduckgo-privacy-essent/bkdgflcldnnnapblkhphbgpggdiikppg, 2021. [Online; accessed 29. Nov. 2023].

[34] Antonin Durey, Pierre Laperdrix, Walter Rudametkin, and Romain Rouvoy. FP-Redemption: Studying browser fingerprinting adoption for the sake of web security. In *Detection of Intrusions and Malware, and Vulnerability Assessment: 18th International Conference, DIMVA 2021, Virtual Event, July 14–16, 2021, Proceedings 18*, pages 237–257. Springer, 2021.

[35] Do Dynamic Web Pages Increase the Risk of Cyber Threats? https://www.packetlabs.net/posts/what-to-know-about-dynamic-web-content/. [Online; accessed 29. Nov. 2023].

[36] Peter Eckersley. How unique is your web browser? In *Proceedings of the 10th International Conference on Privacy Enhancing Technologies (PETS)*, pages 1–18, 2010.

[37] Brendan Eich. C is for Cookie. https://brendaneich.com/2013/05/c-is-for-cookie/, 2013. [Online; accessed 29. Nov. 2023].

[38] Brendan Eich. Latest Firefox Rolls Out Enhanced Tracking Protection. https://blog.mozilla.org/en/products/firefox/latest-firefox-rolls-out-enhanced-tracking-protection/, 2013. [Online; accessed 29. Nov. 2023].

[39] Brendan Eich. The Cookie Clearinghouse. https://brendaneich.com/2013/06/the-cookie-clearinghouse/, 2013. [Online; accessed 29. Nov. 2023].

[40] Email Prospecting. https://www.acxiom.com/email-prospecting/. [Online; accessed 25. Feb. 2024].

[41] Steven Englehardt, Jeffrey Han, and Arvind Narayanan. I never signed up for this! Privacy implications of email tracking. *Proc. Priv. Enhancing Technol.*, 2018(1):109–126, 2018.

[42] Firefox rolls out Total Cookie Protection by default to more users worldwide. https://blog.mozilla.org/en/products/firefox/firefox-rolls-out-total-cookie-protection-by-default-to-all-users-worldwide/, 2013. [Online; accessed 29. Nov. 2023].

[43] FTC Settlement Puts an End to 'History Sniffing' by Online Advertising Network Charged With Deceptively Gathering Data on Consumers. https://www.ftc.gov/news-events/news/press-releases/2012/12/ftc-settlement-puts-end-history-sniffing-online-advertising-network-charged-deceptively-gathering. [Online; accessed 12. Mar. 2024].

[44] Full Third-Party Cookie Blocking and More. https://webkit.org/blog/10218/full-third-party-cookie-blocking-and-more/. [Online; accessed 25. Feb. 2024].

[45] Anita George. 2023 Imperva Bad Bot Report. https://www.imperva.com/resources/resource-library/reports/2023-imperva-bad-bot-report/?utm_medium=PR&utm_source=PR&category=AppSec&destination=imperva.com&Lead-Source=Referral-Organic, 2023. [Online; accessed 08. Dec. 2023].

[46] Mohammad Ghasemisharif, Amrutha Ramesh, Stephen Checkoway, Chris Kanich, and Jason Polakis. O Single Sign-Off, Where Art Thou? An Empirical Analysis of Single Sign-On Account Hijacking and Session Management on the Web. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 1475–1492, 2018.

[47] Ghostery - Privacy Ad Blocker. https://chromewebstore.google.com/detail/ghostery-%E2%80%93-privacy-ad-blo/mlomiejdfkolichcflejclcbmpeaniij, 2009. [Online; accessed 11. May. 2023].

[48] How we protect users from web tracking. https://www.ghostery.com/blog/how-we-at-ghostery-protect-users-from-web-tracking, 2023. [Online; accessed 11. May. 2023].

[49] Desktop Browser Market Share Worldwide. https://gs.statcounter.com/browser-market-share/desktop/worldwide, 2023. [Online; accessed 26. Feb. 2024].

[50] Alejandro Gómez-Boix, Pierre Laperdrix, and Benoit Baudry. Hiding in the Crowd: an Analysis of the Effectiveness of Browser Fingerprinting at Large Scale. In *WWW2018 - TheWebConf 2018 : The Web Conference, 27th International World Wide Web Conference*, 2018.

[51] Aniko Hannak, Gary Soeller, David Lazer, Alan Mislove, and Christo Wilson. Measuring price discrimination and steering on e-commerce web sites. In *Proceedings of the 2014 conference on internet measurement conference*, pages 305–318, 2014.

[52] Internet Advertising Revenue Report. `https://www.iab.com/wp-content/uploads/2023/04/IAB_PwC_Internet_Advertising_Revenue_Report_2022.pdf`, 2023. [Online; accessed 26. Feb. 2024].

[53] Umar Iqbal, Steven Englehardt, and Zubair Shafiq. Fingerprinting the fingerprinters: Learning to detect browser fingerprinting behaviors. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1143–1161. IEEE, 2021.

[54] Umar Iqbal, Peter Snyder, Shitong Zhu, Benjamin Livshits, Zhiyun Qian, and Zubair Shafiq. Adgraph: A graph-based approach to ad and tracker blocking. In *IEEE Symposium on Security and Privacy (SP)*, pages 763–776, 2020.

[55] Shan Jiang, John Martin, and Christo Wilson. Who's the Guinea Pig? Investigating Online A/B/n Tests in-the-Wild. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 201–210, 2019.

[56] Hugo Jonker, Jelmer Kalkman, Benjamin Krumnow, Marc Sleegers, and Alan Verresen. Shepherd: Enabling Automatic and Large-Scale Login Security Studies. *arXiv preprint arXiv:1808.00840*, 2018.

[57] Tadayoshi Kohno, Andre Broido, and Kimberly C. Claffy. Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, 2(2):93–108, 2005.

[58] Balachander Krishnamurthy and Craig Wills. Privacy diffusion on the web: a longitudinal perspective. In *Proceedings of the 18th international conference on World wide web*, pages 541–550, 2009.

[59] Dale Kunkel, Brian L Wilcox, Joanne Cantor, Edward Palmer, Susan Linn, and Peter Dowrick. Report of the APA task force on advertising and children. *Washington, DC: American Psychological Association*, 30:60, 2004.

[60] Pierre Laperdrix, Gildas Avoine, Benoit Baudry, and Nick Nikiforakis. Morellian analysis for browsers: Making web authentication stronger with canvas fingerprinting. In *Detection of Intrusions and Malware, and Vulnerability Assessment: 16th International Conference, DIMVA 2019, Gothenburg, Sweden, June 19–20, 2019, Proceedings 16*, pages 43–66. Springer, 2019.

[61] Pierre Laperdrix, Walter Rudametkin, and Benoit Baudry. Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 878–894. IEEE, 2016.

[62] Margaret-Anne Lawlor and Andrea Prothero. Pester power–a battle of wills between children and their parents. *Journal of Marketing Management*, 27(5-6):561–581, 2011.

[63] Victor Le Pochat, Laura Edelson, Tom Van Goethem, Wouter Joosen, Damon McCoy, and Tobias Lauinger. An audit of Facebook's political ad policy enforcement. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 607–624, 2022.

[64] Meta, TikTok and thousands of major websites are found swiping data you enter on forms–even if you don't hit submit. `https://fortune.com/2022/05/12/meta-tiktok-usa-today-fox-news-websites-swiping-data-before-you-click-submit-study/`. [Online; accessed 22. Jun. 2023].

[65] Des milliers de sites Internet enregistrent les données des formulaires, avant même que le bouton envoyer ait été utilisé. `https://www.lemonde.fr/pixels/article/2022/05/12/des-milliers-de-sites-internet-enregistrent-les-donnees-des-formulaires-avant-meme-que-le-bouton-envoyer-ait-ete-utilise_6125863_4408996.html`. [Online; accessed 22. Jun. 2023].

[66] Adverteerders kijken mee als je online je e-mailadres invult. `https://nos.nl/artikel/2428358-adverteerders-kijken-mee-als-je-online-je-e-mailadres-invult`. [Online; accessed 22. Jun. 2023].

[67] Thousands of Popular Websites See What You Type–Before You Hit Submit. `https://www.wired.com/story/leaky-forms-keyloggers-meta-tiktok-pixel-study/?utm_source=twitter&utm_social-type=owned&utm_brand=wired&utm_medium=social&mbid=social_twitter`. [Online; accessed 22. Jun. 2023].

[68] Douglas J. Leith. Web browser privacy: What do browsers say when they phone home? *IEEE Access*, 9:41615–41627, 2021.

[69] Pedro Giovanni Leon, Justin Cranshaw, Lorrie Faith Cranor, Jim Graves, Manoj Hastak, Blase Ur, and Guzi Xu. What do online behavioral advertising privacy disclosures communicate to users? In Ting Yu and Nikita Borisov, editors, *Proceedings of the 11th annual ACM Workshop on Privacy in the Electronic Society, WPES 2012, Raleigh, NC, USA, October 15, 2012*, pages 19–30. ACM, 2012.

[70] Luka Lodrant. Designing a generic web forms crawler to enable legal compliance analysis of authentication sections. Master's thesis, ETH Zurich, 2022.

[71] Jonathan R Mayer. Internet anonymity in the age of web 2.0. *A Senior Thesis presented to the Faculty of the Woodrow Wilson School of Public and International A airs in partial fulfillment of the requirements for the degree of Bachelor of Arts*, page 103, 2009.

[72] Johan Mazel, Richard Garnier, and Kensuke Fukuda. A comparison of web privacy protection techniques. *Computer Communications*, 144:162–174, 2019.

[73] Navigator. `https://developer.mozilla.org/en-US/docs/Web/API/Navigator`, 2023. [Online; accessed 15. Mar. 2024].

[74] Screen. `https://developer.mozilla.org/en-US/docs/Web/API/Screen`, 2023. [Online; accessed 15. Mar. 2024].

[75] William Melicher, Mahmood Sharif, Joshua Tan, Lujo Bauer, Mihai Christodorescu, and Pedro Giovanni Leon. Preferences for web tracking. *Proceedings on Privacy Enhancing Technologies*, 2:1–20, 2016.

[76] Jakub Mikians, László Gyarmati, Vijay Erramilli, and Nikolaos Laoutaris. Crowd-assisted search for price discrimination in e-commerce: First results. In *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*, pages 1–6, 2013.

[77] Z. Moti, A. Senol, H. Bostani, F. Zuiderveen Borgesius, V. Moonsamy, A. Mathur, and G. Acar. Targeted and Troublesome: Tracking and Advertising on Children's Websites. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 117–117, Los Alamitos, CA, USA, may 2024. IEEE Computer Society.

[78] Keaton Mowery and Hovav Shacham. Pixel Perfect: Fingerprinting Canvas in HTML5. *Proceedings of W2SP*, 2012, 2012.

[79] Mozilla. Enhanced Tracking Protection in Firefox for desktop. `https://support.mozilla.org/en-US/kb/enhanced-tracking-protection-firefox-desktop`. [Online; accessed 08. Sep. 2021].

[80] Firefox blocks fingerprinting. `https://www.mozilla.org/en-US/firefox/features/block-fingerprinting/`, 2023-04-11. [Online; accessed 22. Feb. 2024].

[81] Firefox rolls out Total Cookie Protection by default to more users worldwide. https://blog.mozilla.org/en/mozilla/firefox-rolls-out-total-cookie-protection-by-default-to-all-users-worldwide, 2023-04-11. [Online; accessed 22. Feb. 2024].

[82] Nick Nikiforakis, Alexandros Kapravelos, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *2013 IEEE Symposium on Security and Privacy*, pages 541–555. IEEE, 2013.

[83] Alexandra Nisenoff, Arthur Borem, Madison Pickering, Grant Nakanishi, Maya Thumpasery, and Blase Ur. Defining "broken": User experiences and remediation tactics when Ad-Blocking or Tracking-Protection tools break a Website's user experience. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 3619–3636, Anaheim, CA, August 2023. USENIX Association.

[84] Saying Goodbye to Our Old Friend NPAPI. https://blog.chromium.org/2013/09/saying-goodbye-to-our-old-friend-npapi.html, 2013. [Online; accessed 30. Nov. 2023].

[85] NPAPI Plugins in Firefox. https://blog.mozilla.org/futurereleases/2015/10/08/npapi-plugins-in-firefox/, 2015. [Online; accessed 30. Nov. 2023].

[86] Łukasz Olejnik, Gunes Acar, Claude Castelluccia, and Claudia Diaz. The leaking battery: A privacy analysis of the HTML5 Battery Status API. In *Data Privacy Management, and Security Assurance: 10th International Workshop, DPM 2015, and 4th International Workshop, QASA 2015, Vienna, Austria, September 21–22, 2015. Revised Selected Papers 10*, pages 254–263. Springer, 2016.

[87] Introducing Private Click Measurement, PCM. https://webkit.org/blog/11529/introducing-private-click-measurement-pcm/, 2021-02-01. [Online; accessed 18. Mar. 2024].

[88] Privacy Badger. https://chromewebstore.google.com/detail/privacy-badger/pkehgijcmpdhfbdbbnkijodmdjhbjlgp, 2014. [Online; accessed 11. May. 2023].

[89] PRIVACY BADGER DESIGN AND ROADMAP. https://github.com/EFForg/privacybadger/blob/master/doc/DESIGN-AND-ROADMAP.md, 2023. [Online; accessed 11. May. 2023].

[90] Privacy Budget. https://developers.google.com/privacy-sandbox/protections/privacy-budget#implementation_status, 2022. [Online; accessed 30. Aug. 2023].

[91] Protecting Your Privacy Online. https://privacysandbox.com/, 2023. [Online; accessed 13. Jul. 2023].

[92] Website operators are in the dark about privacy violations by third-party scripts. https://freedom-to-tinker.com/2018/01/12/website-operators-are-in-the-dark-about-privacy-violations-by-third-party-scripts/, 2023. [Online; accessed 22. Feb. 2024].

[93] Process the Request. https://developers.google.com/authorized-buyers/rtb/request-guide. [Online; accessed 26. Feb. 2024].

[94] Protos & Reference Data. https://developers.google.com/authorized-buyers/rtb/data. [Online; accessed 26. Feb. 2024].

[95] Jannis Rautenstrauch, Giancarlo Pellegrino, and Ben Stock. The Leaky Web: Automated Discovery of Cross-Site Information Leaks in Browsers and the Web. In *2023 IEEE Symposium on Security and Privacy (SP)*, 2023.

[96] Real-time Bidding. https://developers.google.com/authorized-buyers/rtb/start. [Online; accessed 26. Feb. 2024].

[97] Eric Rescorla. Technical Comments on Privacy Budget. https://mozilla.github.io/ppa-docs/privacy-budget.pdf, 2021. [Online; accessed 24. Mar. 2024].

[98] Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach. CaroleCadwalladrandEmmaGraham-Harrison, 2018. [Online; accessed 08. Mar. 2024].

[99] Franziska Roesner, Tadayoshi Kohno, and David Wetherall. Detecting and defending against third-party tracking on the web. In *Presented as part of the 9th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 12)*, pages 155–168, 2012.

[100] Esther Rozendaal, Moniek Buijzen, and Patti Valkenburg. Comparing children's and adults' cognitive advertising competences in the Netherlands. *Journal of Children and Media*, 4(1):77–89, 2010.

[101] Full Third-Party Cookie Blocking and More. https://www.apple.com/safari/docs/Safari_White_Paper_Nov_2019.pdf, 2019. [Online; accessed 30. Aug. 2023].

[102] Asuman Senol and Gunes Acar. Unveiling the Impact of User-Agent Reduction and Client Hints: A Measurement Study. In *Proceedings of the 22nd Workshop on Privacy in the Electronic Society*, pages 91–106, 2023.

[103] Asuman Senol, Gunes Acar, Mathias Humbert, and Frederik Zuiderveen Borgesius. Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1813–1830, 2022.

[104] Asuman Senol, Alisha Ukani, Dylan Cutler, and Igor Bilogrevic. The Double Edged Sword: Identifying Authentication Pages and their Fingerprinting Behavior. In *The Web Conference (WWW), 2024*, 2024.

[105] Set-Cookie. `https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie`. [Online; accessed 18. Mar. 202].

[106] Alexander Sjösten, Peter Snyder, Antonio Pastor, Panagiotis Papadopoulos, and Benjamin Livshits. Filter list generation for underserved regions. In *Proceedings of The Web Conference 2020*, pages 1682–1692, 2020.

[107] Ashkan Soltani, Shannon Canty, Quentin Mayo, Lauren Thomas, and Chris Jay Hoofnagle. Flash cookies and privacy. In *2010 AAAI Spring Symposium Series*, 2010.

[108] Ashkan Soltani, Andrea Peterson, and Barton Gellman. NSA uses Google cookies to pinpoint targets for hacking. *The Washington Post*, 10, 2013.

[109] Grant Storey, Dillon Reisman, Jonathan Mayer, and Arvind Narayanan. The future of ad blocking: An analytical framework and new techniques. *arXiv preprint arXiv:1705.08568*, 2017.

[110] Adobe Communications Team. Adobe Study: Brands That Rely Too Heavily on Third-Party Cookies are Leaving Money on The Table Today, Risking Long-Term Business Harm. `https://business.adobe.com/uk/blog/perspectives/brands-rely-too-heavily-on-third-party-cookies-survey-uk`, 2023. [Online; accessed 26. Feb. 2024].

[111] 6 Steps for Avoiding Online Form Abandonment. `https://themanifest.com/web-design/blog/6-steps-avoid-online-form-abandonment`, 2022. [Online; accessed 29. Nov. 2023].

[112] Browse Privately. Explore Freely. `https://www.torproject.org/`. [Online; accessed 18. Sept. 2023].

[113] uBlock Origin. `https://chromewebstore.google.com/detail/ublock-origin/cjpalhdlnbpafiamejdnhcphjbkeiagm`, 2014. [Online; accessed 11. May. 2023].

[114] What is User-Agent reduction? `https://developers.google.com/privacy-sandbox/protections/user-agent`. [Online; accessed 22. Jun. 2023].

[115] Steven Van Acker, Daniel Hausknecht, and Andrei Sabelfeld. Measuring login webpage security. In *Proceedings of the Symposium on Applied Computing*, pages 1753–1760, 2017.

[116] Tom Van Goethem, Wout Scheepers, Davy Preuveneers, and Wouter Joosen. Accelerometer-based device fingerprinting for multi-factor mobile authentication. In *Engineering Secure Software and Systems: 8th International Symposium, ESSoS 2016, London, UK, April 6–8, 2016. Proceedings 8*, pages 106–121. Springer, 2016.

[117] Antoine Vastel, Walter Rudametkin, Romain Rouvoy, and Xavier Blanc. FP-Crawlers: studying the resilience of browser fingerprinting to block crawlers. In *MADWeb'20-NDSS Workshop on Measurements, Attacks, and Defenses for the Web*, 2020.

[118] Mike West. An update on Privacy Budget. `https://github.com/mikewest/privacy-budget/commit/8377d68aa7a0b07bd9de04f42738bdf08623f06b`, 202. [Online; accessed 24. Mar. 2024].

[119] John Wilander. Intelligent Tracking Prevention. `https://webkit.org/blog/7675/intelligent-tracking-prevention`, 2017. [Online; accessed 29. Nov. 2023].

[120] Shuai Yuan, Jun Wang, and Xiaoxue Zhao. Real-time bidding for online advertising: measurement and analysis. In *Proceedings of the Seventh International Workshop on Data Mining for Online Advertising*, ADKDD '13, New York, NY, USA, 2013. Association for Computing Machinery.

[121] Michal Zalewski. p0f Unofficial Git-Repo. `https://github.com/p0f/p0f`, 2019-02-21. [Online; accessed 01. Jul. 2023].

[122] Qiuhan Zhao, Wenchuan Yang, and Rui Hua. Design and research of composite web page classification network based on deep learning. In *IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1531–1535, 2019.

[123] Yuchen Zhou and David Evans. SSOScan: Automated Testing of Web Applications for Single Sign-on Vulnerabilities. In *Proceedings of the 23rd USENIX Conference on Security Symposium*, SEC'14, pages 495–510, USA, 2014. USENIX Association.

Part II

Publications

List of Publications

International Conferences and Workshops with Proceedings

1. Senol, A., Acar, G., Humbert, M., and Borgesius, F. Z. Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission. In *31st USENIX Security Symposium (USENIX Security 22)* (2022), pp. 1813–1830

   • See p. 59

2. Senol, A., and Acar, G. Unveiling the Impact of User-Agent Reduction and Client Hints: A Measurement Study. In *Proceedings of the 22nd Workshop on Privacy in the Electronic Society* (2023), pp. 91–106

   • See p. 99

3. Moti, Z., Senol, A., Bostani, H., Borgesius, F. Z., Moonsamy, V., Mathur, A., and Acar, G. Targeted and Troublesome: Tracking and Advertising on Children's Websites. In *2024 IEEE Symposium on Security and Privacy (SP)* (Los Alamitos, CA, USA, may 2024), IEEE Computer Society, pp. 117–117

   • See p. 147

4. Senol, A., Ukani, A., Cutler, D., and Bilogrevic, I. The Double Edged Sword: Identifying Authentication Pages and their Fingerprinting Behavior. In *The Web Conference (WWW), 2024* (2024)

   • See p. 199

Publication

Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission

Publication Data

Senol, A., Acar, G., Humbert, M., and Borgesius, F. Z. Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission. In *31st USENIX Security Symposium (USENIX Security 22)* (2022), pp. 1813–1830

Contributions

- Principal author.

Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission

Asuman Senol[1], Gunes Acar[2], Mathias Humbert[3], and Frederik Zuiderveen Borgesius[2]

[1] imec-COSIC KU Leuven
[2] Radboud University
[3] University of Lausanne

Abstract. Web users enter their email addresses into online forms for a variety of reasons, including signing in or signing up for a service or subscribing to a newsletter. While enabling such functionality, email addresses typed into forms can also be collected by third-party scripts even when users change their minds and leave the site without submitting the form. Email addresses—or identifiers derived from them—are known to be used by data brokers and advertisers for cross-site, cross-platform, and persistent identification of potentially unsuspecting individuals. In order to find out whether access to online forms is misused by online trackers, we present a measurement of email and password collection that occurs before the form submission on the top 100,000 websites. We evaluate the effect of user location, browser configuration, and interaction with consent dialogs by comparing results across two vantage points (EU/US), two browser configurations (desktop/mobile), and three consent modes. Our crawler finds and fills email and password fields, monitors the network traffic for leaks, and intercepts script access to filled input fields. Our analyses show that users' email addresses are exfiltrated to tracking, marketing and analytics domains before form submission and without giving consent on 1,844 websites in the EU crawl and 2,950 websites in the US crawl. While the majority of email addresses are sent to known tracking domains, we further identify 41 tracker domains that are not listed by any of the popular blocklists. Furthermore, we find incidental password collection on 52 websites by third-party session replay scripts.

Keywords – online tracking, email exfiltration, web privacy

1. Introduction

Websites commonly use third-party advertising and marketing services to monetize their content. Those services heavily depend on monitoring users' online activities, at times without their knowledge and consent. Stateful tracking mechanisms such as cookies are isolated by origins, and limited to the web platform. As users' online activities are spread over a number of connected devices, tracking users only on websites does not suffice to get a complete view of their profile. The demand for an alternative mechanism to track users across websites and devices has also increased since major browser vendors such as Safari and Firefox have started blocking or partitioning third-party cookies and trackers.

Email addresses are ideal identifiers to fill this gap, since they are unique, persistent, and can even be available in the offline realm—e.g., when a user signs up for a loyalty card. Compared to other personal information such as name or postal address, email addresses are more effective for tracking users across platforms, since they are long-term, unique, and available on many websites and applications to facilitate account login, registration, and newsletter subscriptions. Data brokers and advertisers already use email hashes to identify users, track them across devices, and match their online and offline activities [63, 18, 38].

The demand for a more global and persistent identifier, along with the ongoing phase-out of third-party cookies, makes email addresses typed into online forms an attractive target for collection by trackers. However, prior work on the collection of credentials typed into online forms is limited. Besides, the collection of information *before* form submission has been even less studied. Only a 2017 news article by Surya Mattu and Kashmir Hill reported that a third party called Navistone was collecting personal information from mortgage calculator forms before the user submitted the form [54]. This is despite the high dropout rates among web users (e.g., in signup forms [12, 73]), which shows that many users indeed leave websites without submitting the form they started filling out. For instance, a survey by The Manifest found that 81% of the 502 respondents have abandoned forms at least once, and 59% abandoned a form in the last month [79].

In this paper, we investigate to what extent third-party trackers collect email addresses, and (incidentally) passwords, even if the user does not submit any form. Unlike prior work, we focus on leaks that occur before form submission, and we analyze the effect of location, of user consent to personal data processing, and of mobile vs. desktop browsing.

In addition, we evaluate the effect of users' location, of user consent to personal

data processing, and of mobile vs. desktop browsing. In particular, we run crawls from two vantage points (EU & US), with desktop and mobile-emulated browsers. In addition, we use three different cookie consent settings to investigate the effect of user consent: accept all, reject all, and no action. Our contributions include the following:

- We develop an interactive, instrumented crawler based on DuckDuckGo's Tracker Radar Collector [25] to measure email and password exfiltration on Tranco top 100K sites. We fit the crawler with a pre-trained machine-learning (ML) classifier that can robustly detect email fields. Our crawler is further able to fill the email and password fields and to intercept script access to filled input fields (Section 3.1).
- Based on a crawl of 2.8 million pages from the top 100K sites, we find that trackers collect email addresses before form submission on thousands of websites in both EU (1,844 websites) and US (2,950 websites) crawls—60% more exfiltrations when the same sites are visited from the US. We uncover 41 previously unknown tracker domains that exfiltrate email addresses. We develop a proof-of-concept browser add-on that detects sniff and exfiltration attempts on online forms.
- We discuss whether email exfiltrations by trackers are compliant with the GDPR or not (Section 5). Further, we send GDPR requests to a sample of websites and third parties, asking the purpose of their email collection, retention period and further sharing policies (Section 6).
- Finally, we uncover incidental password collection by session replay providers on 52 websites (Section 4.2). Two third-party trackers with a combined presence of five million websites released fixes to address the issue, thanks to our disclosures.

2. Background and Related Work

2.1 Background

Web tracking is the process of collecting information about users' online activities across websites. The personal information that can be collected or inferred by the trackers may include personal and sensitive information such as sexual orientation, political and religious beliefs. Tracking may be performed for various purposes including analytics, personalization, and building a behavioral profile for marketing and targeted advertisements.

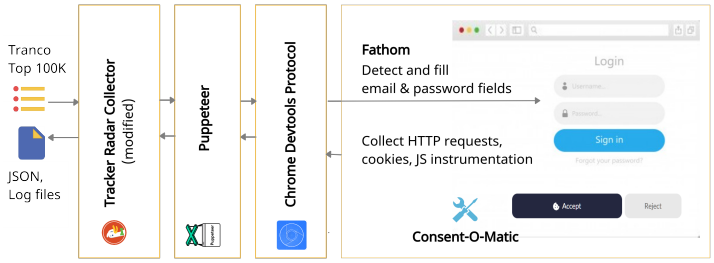


Figure 1: Components of our crawler. We integrate Firefox Relay's Fathom-based email field classifier[4] and Consent-O-Matic[5] to Tracker Radar Collector (TRC)—a web privacy measurement crawler developed by DuckDuckGo[6]. TRC is based on Puppeteer, which uses Chrome Devtools Protocol to interact with the underlying browser. We modify TRC to efficiently discover inner pages, and fill email and password fields.

The most traditional way to track users across websites is to store a unique identifier in users' cookies. However, in the last decade, more intrusive and persistent tracking mechanisms have emerged. Browser fingerprinting [26], evercookies [34] and cookie syncing [62] are such mechanisms that are harder to control and detect than the traditional cookies. As a reaction to these emergent tracking mechanisms, tracking protection countermeasures such as browser extensions and built-in browser defenses were developed. For instance, Safari's Intelligent Tracking Prevention, and Firefox's Enhanced Tracking Protection can prevent third-party tracking by identifying trackers and blocking cookies that are used for cross-site tracking [85, 57]. The countermeasures against traditional tracking mechanisms made alternatives such as tracking based on personal identifiers or "people-based marketing" [65] even more necessary.

2.2 Related Work

Online tracking Several studies investigated stateful [50, 68] and stateless [48, 43, 32] tracking techniques and their evolution over time. Taking an offensive approach, other studies proposed new tracking techniques that are difficult to detect such as canvas and GPU fingerprinting [56, 47]. Analyzing IAB Europe's Transparency and Consent Framework (TCF) cookie banners,

[4] https://github.com/cavi-au/Consent-O-Matic
[5] https://github.com/duckduckgo/tracker-radar-collector
[6] https://github.com/mozilla/fx-private-relay/blob/v1.2.2/extension/js/email_detector.js

Matte et al. found a widespread violation of the GDPR and the ePrivacy Directive; for instance by registering positive consent when the user has not made a choice [53]. Similar to our discussion on GDPR compliance of email exfiltration practices (Section 5), Mayer and Mitchell presented an overview of regulation that applies to online tracking–but their analysis predates modern privacy laws such as the GDPR [55].

Personal information leaks Lin et al. presented the first comprehensive study of privacy threats emanating from browsers' auto-fill functionality [51]. While relevant, auto-fill-related abuse is orthogonal to the types of exfiltration we investigate. Acar et al. studied personal data exfiltration by third parties, uncovering inadvertent password leaks by session replay scripts, and third parties that harvest (hashed) email addresses by injecting invisible login forms that trigger browsers' login managers [2].

Englehardt et al. built a corpus of emails by signing up to mailing lists, and they found that 30% of emails they received leaked the recipient's email address to one or more third-party servers when viewed in an email client program or web application [31]. Similar to our study, Englehardt et al. also searched and filled email fields, but their method aimed to identify leaks that occur when reading emails—not when typing email addresses on the page.

Starov et al. studied PII leakage on contact pages of the 100,000 most popular sites on the web [75]. They populated contact forms with a name, surname, email address and a sample contact message. Their results showed that, after removing accidental leakage, 6.1% (1,035) of all contact forms leaked PIIs to third parties after form submission. They also found that PIIs were leaked to third parties before submitting the contact form on 13 websites. Unlike Starov et al.'s work, we ran comparative crawls (mobile/desktop, US/EU, consent modes); and our leak detection method did not require three visits. While not directly comparable, we identified substantially more personal information leaks.

Chatzimpyrros et al. [11] and Dao et al. [17] investigated PII leaks on top 200K websites, and on 307 popular shopping websites respectively. Chandramouli et al. measured the prevalence of email header injection vulnerabilities in web forms, which can be used for phishing, spoofing, and other attacks [10]. Other prior work investigated PII leaks on mobile devices [67, 66], or compared tracking on mobile and desktop devices [89].

Our study differs from these works by focusing on email and password exfiltration during the filling of the forms. We run crawls from multiple vantage points, with different consent modes to evaluate their effect on data exfiltration. We compare email and password collection on mobile and desktop crawls. In addition, we use

GDPR requests to reach out to first and third parties to ask for the purposes of email address collection.

Web privacy measurement studies Many researchers developed their own tools to study web tracking techniques in the wild. In 2012, Mayer and Mitchell implemented FourthParty, a Firefox extension that instrumented browser APIs, HTTP traffic and cookies [55]. Using FourthParty, they examined web tracking techniques on more than 500 websites. FPDetective is based on a modified PhantomJS and Chromium and was used to measure browser fingerprinting on the top million pages [3]. Englehardt and Narayanan developed OpenWPM, which consists of an instrumentation extension and automation code that drives a full-fledged Firefox browser [32]. Jueckstock and Kapravelos contributed VisibleV8, a modified V8 JavaScript engine that logs all native JS function calls and property accesses, without the need to add specific instrumentation [46]. Akhavani et al. inspected 33 Google Chrome, 31 Mozilla Firefox, and 33 Opera browser versions released from 2016 to 2020 by using VisibleV8, and showed that different browser versions have identifiable fingerprints [4]. Recently, DuckDuckGo developed Tracker Radar Collector [25], an instrumented Puppeteer-based crawler that is used to detect trackers through large-scale crawls. We chose to build our crawler by extending Tracker Radar Collector for its simplicity and scalability. We explain the details of this process in the following section.

Login security Jonker et al. presented a framework called Shepherd, which detects login pages using a combination method of searching for login-based URLs, clickable elements and search engine APIs [45]. Shepherd also interacts with the login forms, and analyzes authentication cookies to determine whether the website is vulnerable to session hijacking. Analyzing the use of web authentication mechanisms on 100,000 domains, Van Acker et al. showed that login pages of certain open-source web frameworks and content management systems are vulnerable to several attacks under various adversary models [81]. They evaluated 51,307 login pages from 100K websites against man-in-the-middle attacks showing that 62.8% of login pages are vulnerable to adversaries with moderate resources. Van Acker et al.'s study also showed that password leaks to third parties are possible on many websites. Unlike these two studies measuring login page vulnerabilities, we measure the *actual misuse* by trackers on real-world websites.

3. Methods

3.1 Extending Tracker Radar Collector

Tracker Radar Collector (TRC) is a modular, multi-threaded crawler that is tailored for large-scale web measurements. Using Puppeteer under the hood, TRC takes advantage of all the capabilities of the Chrome DevTools Protocol. TRC uses *collectors*—modules in charge of capturing tracking-related behavior—that captures browser API accesses, cookies and requests. Unlike OpenWPM's inline instrumentation [46] that wraps functions and objects with getters, TRC uses Chrome DevTools Protocol to set conditional breakpoints that are evaluated when a certain function is called or a property is accessed. When the debugger hits a breakpoint set by TRC, the condition script collects the JavaScript stack trace and other metadata about the property access or function invocation.

In order to detect email and password exfiltration, we extended TRC by adding a *collector* that finds and fills email and password fields. Besides, we extended TRC's network instrumentation to capture WebSocket traffic and HTTP POST payloads—in addition to GET requests which are already being intercepted. We also added instrumentation to intercept JavaScript access to input fields, capturing the access time, input value, and attributes of the accessed input element. A high-level overview of our crawler is shown in Figure 1.

3.2 Discovering Inner Pages

Our crawler starts to search email and password fields on the landing pages. If no field can be found, it tries to follow links to discover fields in the inner pages. To find links that are more likely to yield email and password fields, we use a combined regular expression pattern that we extract from Firefox's Password Manager module [37]. The pattern contains several translations of words related to "sign in", "sign up" and "register". We search for this pattern in the following attributes of *a*, *button*, *div*, *span* elements: `innerText`, `title`, `href`, `placeholder`, `id`, `name` and `className`. We limit ourselves to these four elements since they can be used to create links on the page. We prioritize elements that exactly match the regular expression pattern over elements that partially match the pattern. As a final fallback, we search for links (this time only considering *a*, *button* elements) according to their page coordinates (i.e., distance from the top left corner). Based on a pilot crawl of 100K websites, we calculated the median X and Y position of the links that led to pages with email or password fields: 1113px and 64.5px, respectively. Note that, since we used a 1440px-wide viewport in the desktop crawls, this point is

very close to the viewport's top right corner, where sign-in/sign-up links are commonly found. This coordinate-based link detection method increased the number of detected email fields by around 10%. Within each link category (exact match, loose match, coordinate-based match), we prioritize 1) *a* and *button* links, 2) links that are in the viewport, 3) links that are on top of other elements (computed via `Document.elementFromPoint()`). We arrived at these prioritization steps by comparing email and password yields using different methods in pilot crawls.

While clicking the links, we keep a record of the URLs we have visited and we skip links to already visited pages. We continue to click these sorted links until we find and fill an email field, or until we clicked ten links. We choose ten as the maximum number of links to click, since pilot crawls showed diminishing returns after ten links.

3.3 Identifying Email and Password Fields

After clicking each link, we search for email and password fields on the new page and on all of its iframes. We search for iframes since a pilot crawl of top 1K Tranco sites showed that 3% of email fields are found in iframes. For detecting password fields, we search for input fields with type `password` (i.e. `input[type='password']`). However, email input fields do not need to have the `email` type (i.e. `input[type='email']`). In fact, through pilot crawls we found that many websites, including popular ones such as facebook.com, use text input elements to accommodate login with phone numbers or other username formats. To address this challenge, we integrated into our crawler a pre-trained email field classifier based on Mozilla Fathom [27]. Fathom is a supervised learning framework specialized to detect webpage parts such as popups [36]. We used the Fathom-based email field detector model used in Firefox Relay add-on [27]. Firefox Relay is a privacy-focused service from Mozilla that offers free email aliases.[7] Using the Fathom-based detector allowed us to identify 76% more email fields than we would detect by simply searching for input fields with type `email`. This substantial increase may indicate that earlier studies that relied on `email` input type could have missed a significant number of email fields.

3.4 Filling Email and Password Fields

We use a unique email address on each page by adding the site domain to the email address after a plus (+) character. This allowed us to uniquely attribute

[7]Coincidentally, Firefox Relay and similar email alias services can be used as countermeasures against email exfiltration we study in this paper.

received emails to the websites they are collected on. To address potential bot detection measures, we simulate user typing behavior by using randomized intervals for each key press and dwell times, as well as the delay times between each press. After typing into each field, we simulate pressing the 'Tab' key to switch to the next form field, while triggering the *blur* event on the previously filled element.

Englehardt et al. found that the "Show password" feature, which changes the `type` of the password field from `password` to `text`, caused certain session replay scripts to collect the passwords incidentally [29]. To measure such leaks at large, the crawler changes the password fields' type from `password` to `text` before filling the field. This allows us to simulate the effect of browser extensions such as ShowPassword [72], which displays passwords in cleartext. We then run a follow-up crawl without changing the password input type on websites where we identified password leaks. Overall, our password exfiltration measurements aim to identify the incidental collection, rather than malicious password theft.

3.5 Interaction with Consent Management Dialogs

After the introduction of the GDPR in 2018, more websites started to show dialogs to get users' consent for personal data processing. The acceptance or refusal to give consent may have an effect on how the website and the third parties may collect, process and share users' personal data. While one expects less tracking and data collection when refusing to give consent, prior research showed that in certain cases the opposite may be true: a recent study by Papadogiannakis et al. found that websites are more likely to use sophisticated tracking techniques such as ID syncing and fingerprinting when users reject cookies [64]. Regardless, web privacy studies such as ours should take consent dialog interaction into account since it may affect how websites and third parties behave.

In order to investigate the effect of users' consent preferences, we integrate Consent-O-Matic [8] into our crawler. Developed by Nouwens et al. to study dark patterns in consent dialogs, Consent-O-Matic is a browser extension that can recognize and interact (e.g., accept or reject cookies) with various Consent Management Provider (CMP) pop-ups [61]. We configure Consent-O-Matic to log detected CMPs, and perform the following interactions with the CMPs:

accept-all: Allow processing for all purposes. **reject-all**: Disallow processing for all purposes. **no-action**: Continue without interacting with the CMP dialog, if any.

3.6 Measurement Configuration

We measure email and password exfiltration on the top 100,000 Tranco websites [49].[8] Initially, we used the Tranco domains without any changes, but we encountered DNS errors even on most popular websites such as windowsupdate.com—the eighth most popular site in Tranco. To address this problem, we matched Tranco domains to URLs listed in the Chrome User Experience Report [16], which contains actual URLs visited by Chrome users. When matching domains to URLs, we pick the URL with the lower rank (more popular) if there are multiple alternatives. This minor change increased the successfully visited websites from 94,427 (EU pilot crawl) to 99,380 (EU final crawl). We used the March 2021 versions of both Tranco and Chrome UX Report lists.

To compare results based on user location, we run two simultaneous crawls from the EU (Frankfurt) and the US (New York City)—both using cloud-based servers hosted on Digital Ocean. For each crawl, we use one server with 16 cores and 32GB RAM.

We limit the maximum crawl duration on a site to 180 seconds and maximum page load time to 90 seconds. After detecting a CMP on a website, we wait 6 seconds for the CMP interaction (accept or reject) to complete. We determined these timeouts and other crawl parameters based on data from 1K pilot crawls. For instance, we measured how long the CMP operations take and set the extra wait time to the 99th percentile of the distribution (6 seconds).

In addition, we run crawls for mobile websites to measure the email and password exfiltration on the mobile web. We emulated a mobile browser by adjusting the viewport dimensions, spoofing touch support, and using a mobile user-agent string. The mobile-specific parameters we used are available in the TRC source code [25]. For mobile crawls, we fill a different email address to distinguish emails we received due to mobile and desktop crawls. We omit experiments with different consent modes for mobile crawls due to limited time and space.

3.7 Email and Password Leak Detection

Identifying encoded, hashed or obfuscated leaks is a challenge that we need to address to avoid underestimating leaks. This challenge was tackled in different ways in prior work in web privacy measurement studies. Starov et al. compare data from three different crawls to identify PII in HTTP traffic [75]. Since Starov et al.'s method requires more crawls and manual analysis, we prefer

[8] Available at `https://tranco-list.eu/list/6WGX/100000`

Table 1: Desktop crawl statistics based on servers located in the EU and the US. no-action, accept-all, reject-all indicate consent modes. Crawled pages also include inner pages that we visited.

	EU				US			
Crawl Option	**no-action**	**accept-all**	**reject-all**	**mobile**	**no-action**	**accept-all**	**reject-all**	**mobile**
Crawled URLs	100K	7,720	7,720	100K	100K	7,720	7,720	100K
Successfully loaded websites	99,380	7,716	7,716	99,363	99,437	7,714	7,716	99,409
Crawled pages	625,143	44,752	40,385	597,791	690,394	51,735	49,260	668,848
Websites where we filled email	52,055	5,076	5,115	47,825	53,038	5,071	5,077	49,615
Websites where we filled password	31,002	2,306	2,342	29,422	31,324	2,263	2,283	30,356

Englehardt et al.'s method [31], which involves searching for different encodings and hashes of search terms, including Base64 encoding, and hash functions such as SHA-256. Starting with the email and password we filled, we compute a *precomputed pool* that contains all possible sets of tokens by iteratively applying the hashes and encodings. We then search for the leaks in the referrer header, cookies, URL and POST bodies of the requests, by splitting the contents by potential separator characters, such as '='. We apply all possible decodings and we check whether the decoded result is in the precomputed pool. We repeat this process until we reach a level of three layers of encodings or decodings. We list the hash and encoding algorithms we used in Appendix 10.

We improve upon the original method by Englehardt et al. in several ways. First, in addition to splitting content by separators and decoding the resulting strings, we search for different encodings of the search terms (e.g., email and password values). This enabled us to detect leaks that do not conform to the standard `key=value` structure. Similar to the precomputed pool mentioned above, we iteratively apply the encodings. Further, we identify two new encodings and one hash method that were not covered by Englehardt et al.'s original detector. The newly discovered encoding methods include a simple substitution cipher that replaces each letter with another based on a fixed mapping. We extract this mapping from a third-party script's source code and incorporate it into the leak detector. We identified such missed leaks by using the received emails as proof of email collection. We manually analyzed scripts from parties that send emails but were not found to collect leaked emails. Using this method, we also found a third party that compresses payloads using `lzstring`, and another third party that hashes email addresses with a fixed salt, which was hard-coded in their script. Note that using (salted) email hashes may prevent this third party from matching identities with external entities such as data brokers—unless the data broker also uses the same salt for hashing emails.

3.8 Determining Tracker-Related Leaks

There may be legitimate reasons why email addresses and—to some extent—passwords are collected before form submission: For instance, checking whether an email/username picked by a user is available before form submission. To avoid counting such cases, we exclude from our analysis all requests that are sent to first-party domains, or third-party domains that are not flagged as trackers. When determining *third partyness* we make use of Tracker Radar's entity list [33], which contains a list of domains owned by a company. Using entity-to-domains mapping allows us to better determine the third parties, and prevent overcounting the leaks. In addition, we exclude cases where we filled the email on a page or on an iframe that has a different domain than the crawled website. Note that throughout the study by *domain*, we mean registrable domain name or the effective top-level domain plus one (eTLD+1).

Lastly, we only consider requests that are sent to endpoints flagged as a tracker by one of Disconnect [20], Whotracks.me [84], DuckDuckGo [24] blocklists and uBlock Origin [80]. For the Disconnect list, we also consider domains in the "Content" category, which is only blocked if Firefox is in Private Browsing mode. For uBlock Origin, we use the blocklists enabled by default in the add-on. These include EasyList, EasyPrivacy and Peter Lowe's Ad and tracking server list, among others.

Manual tracker labeling Additionally, we label the leaky request domains that are not flagged as trackers by any of the Disconnect, Whotracks.me, DuckDuckGo and uBlock Origin. For each such domain, we follow a decision algorithm explained in Appendix 10 to determine the tracker status. Thanks to this manual analysis, we uncover 41 tracker domains that are not listed in any of the popular blocklists. Manually labeled domains accounted for an increase of 13.4% and 4.2% in the number of websites with email leaks, in the EU and US crawls, respectively (for no-action, desktop crawls). We plan to share these domains with blocklists providers.

3.9 Dataset

Our main dataset consists of eight crawls, all of which were run in May and June of 2021. A total of six desktop crawls were run from the EU and the US using three consent modes: *no-action*, *accept-all*, *reject-all*. In addition, two mobile crawls were run using the *no-action* mode from the two locations. In the four, no-action crawls (100K websites), we flag the websites where we detected (but not interacted) the presence of a CMP using Consent-O-Matic. We then use these CMP-detected websites in the accept-all and reject-all crawls. For

comparability we use the same 7,720 CMP-detected websites in the accept-all and reject-all crawls on both locations—the 7,720 websites were detected in the EU crawl. While we limit our crawls to the top 100K websites, our dataset contains approximately 2.8M page visits across all crawls considering the inner pages visited when searching for email and password fields. In addition to the HTTP request and response details, our dataset also contains HTML sources, JavaScript instrumentation logs, and screenshots that can be used to debug the crawler. Each 100K website crawl took five days to run. The ethics considerations we took into account during the study can be found in Section 9.

4. Measurement Results

Table 2: Top tracker domains and associated entities that emails or passwords are exfiltrated to in desktop crawls using the no-action mode which was conducted in May'21. *: Third-party domain is not among the request initiators; that means the leak could have been triggered by another party. Prominence (Prom.) values have been multiplied by 1,000 for readability.

	EU						US					
Leak Type	**Entity Name**	**Tracker Domain**	**Key by key**	**Num. sites**	**Prom.**	**Min. Rank**	**Entity Name**	**Tracker Domain**	**Key by key**	**Num. sites**	**Prom.**	**Min. Rank**
email	Taboola	taboola.com	No	327	302.9	154	LiveRamp	rlcdn.com	No	524	553.8	217
	Adobe	bizible.com	Yes	160	173.0	242	Taboola	taboola.com	No	383	499.0	95
	FullStory	fullstory.com	Yes	182	75.6	1,311	Bounce Exchange	bouncex.net	No	189	224.7	191
	Awin Inc.	zenaps.com*	No	113	48.7	2,043	Adobe	bizible.com	Yes	191	212.0	242
		awin1.com*	No	112	48.5	2,043	Awin	zenaps.com*	No	119	111.2	196
	Yandex	yandex.com	Yes	121	41.9	1,688		awin1.com*	No	118	110.9	196
	AdRoll	adroll.com	No	117	39.6	3,753	FullStory	fullstory.com	Yes	230	105.6	1,311
	Glassbox	glassboxdigital.io*	Yes	6	31.9	328	Listrak	listrakbi.com	Yes	226	66.0	1,403
	Listrak	listrakbi.com	Yes	91	24.9	2,219	LiveRamp	pippio.com	No	138	65.1	567
	Oracle	bronto.com	Yes	90	24.6	2,332	SmarterHQ	smarterhq.io*	Yes	32	63.8	556
	LiveRamp	rlcdn.com	No	11	20.0	567	Verizon Media	yahoo.com*	Yes	255	62.3	4,281
	SaleCycle	salecycle.com	Yes	35	17.5	2,577	AdRoll	adroll.com	No	122	48.6	2,343
	Automattic	gravatar.com*	Yes	38	16.7	2,048	Yandex	yandex.ru	Yes	141	48.1	1,648
	Facebook	facebook.com	Yes	21	14.8	1,153	Criteo SA	criteo.com*	No	134	46.0	1,403
	Salesforce	pardot.com*	Yes	36	30.8	2,675	Neustar	agkn.com*	No	133	45.9	1,403
	Oktopost	okt.to*	Yes	31	11.4	6,589	Oracle	addthis.com	No	133	45.9	1,403
pswd	Yandex	yandex.com	Yes	37	12.12	4,699	Yandex	yandex.ru	Yes	45	17.23	1,688
		yandex.ru		7	2.41	12,989						
	Mixpanel	mixpanel.com	Yes	1	0.12	84,547	Mixpanel	mixpanel.com	Yes	1	0.12	84,547
	LogRocket	lr-ingest.io	Yes	1	0.12	82,766	LogRocket	lr-ingest.io	Yes	1	0.12	82,766

Results in this section are based on desktop crawls and no-action mode (no interaction with the cookie dialog) unless otherwise specified.

4.1 Email Leaks

Prevalence of leaks Table 3 shows that email addresses (or their hashes) are sent to a third-party tracker on 1,844 (EU) vs. 2,950 (US) distinct websites.

Table 3: The number of distinct websites where email and passwords are sent to first-party domains vs. third-party domains in desktop crawls using the no-action mode.

	EU			US		
	All	**Third party**	**Tracking related**	**All**	**Third party**	**Tracking related**
Email	4,395	2,633	**1,844**	5,518	3,790	**2,950**
Password	89	87	**48**	92	87	**49**

This shows that, on more than a thousand websites, trackers only collect emails when the website is visited from the US.

Table 2 gives a more detailed overview of the most common trackers that emails are leaked to. Prom. stands for prominence, a metric developed by Englehardt and Narayanan [32] which captures both the quantity and popularity of websites a third party is embedded on. We use prominence to sort third parties in Table 2 because it better represents the scale of a given third party's reach.

Table 4: Top ten websites where the filled email was collected by a tracker before form submission in desktop crawls using the no-action mode. *: Not reproducible anymore as of February 2022.

EU				US			
Rank	**Website**	**Third-party**	**Hash/encoding/compression**	**Rank**	**Website**	**Third-party**	**Hash/encoding/compression**
154	**usatoday.com***	taboola.com	Hash (SHA-256)	95	**issuu.com**	taboola.com	Hash (SHA-256)
242	**trello.com***	bizible.com	Encoded (URL)	128	**businessinsider.com**	taboola.com	Hash (SHA-256)
243	**independent.co.uk***	taboola.com	Hash (SHA-256)	154	**usatoday.com**	taboola.com	Hash (SHA-256)
300	**shopify.com**	bizible.com	Encoded (URL)	191	**time.com**	bouncex.net	Compression (LZW)
328	**marriott.com**	glassboxdigital.io	Encoded (BASE-64)			awin1.com	Hash (SHA-256 with salt)
567	**newsweek.com***	rlcdn.com	Hash (MD5, SHA-1, SHA-256)	196	**udemy.com**	zenaps.com	Hash (SHA-256 with salt)
705	**prezi.com***	taboola.com	Hash (SHA-256)	217	**healthline.com**	rlcdn.com	Hash (MD5, SHA-1, SHA-256)
754	**branch.io***	bizible.com	Encoded (URL)	234	**foxnews.com**	rlcdn.com	Hash (MD5, SHA-1, SHA-256)
1,153	**prothomalo.com**	facebook.com	Hash (SHA-256)	242	**trello.com***	bizible.com	Encoded (URL)
1,311	**codecademy.com**	fullstory.com	Unencoded	278	**theverge.com**	rlcdn.com	Hash (MD5, SHA-1, SHA-256)
1,543	**azcentral.com***	taboola.com	Hash (SHA-256)	288	**webmd.com**	rlcdn.com	Hash (MD5, SHA-1, SHA-256)

In the US crawl, rlcdn.com (LiveRamp, formerly Acxiom) is the most prominent tracker domain that collects hashed email addresses. On WebMD and Fox News websites, LiveRamp collected the MD5, SHA-1 and SHA-256 hashes of the email address typed into the login form. The EU list, on the other hand, is dominated by Taboola—an advertising company that was found to promote clickbait and other problematic content and ads [69, 90]. According to their help pages, Taboola accepts hashed emails to create target audiences [78] based on over 1.4 billion unique visitors they reach every month [77].

Cross-domain email sharing for identity matching On 565 of the 1,844 distinct websites (EU) where we identified email leaks to tracker domains,

no script from the request domain was among the request's initiators. This means that these requests are initiated by other parties. Analyzing HTTP request initiators, and JavaScript stack traces of access to input fields we found that email leaks to yahoo.com, criteo.com and dotomi.com are always initiated by other parties. The email hashes to yahoo.com, for example, are sent by a script from adthrive.com (CafeMedia)—a digital publishing and ad monetization network that Yahoo has a partnership with [71]. The Yahoo endpoint (`ups.analytics.yahoo.com`) that email hashes are sent to, is described in Yahoo's ConnectID API documentation [83]. The documentation mentions that the API can be used for ID matching and is built on Verizon Media's *ID Graph*, "delivering a higher find rate of audiences on publishers' sites [sic] user targeting". Clickagy(.com), on the other hand, sends email hashes to up to seven other tracker domains including agkn.com (Neustar) and pippio.com (LiveRamp), both of which accepts hashed emails for various services according to their public documentation and privacy policies [39, 59].

Our findings showed that email addresses or their hashes are sent to facebook.com on 21 distinct websites in the EU. On 17 of these, Facebook Pixel's Automatic Advanced Matching feature [35] was responsible for sending the SHA-256 of the email address in a `SubscribedButtonClick` event, despite not clicking any submit button. According to its documentation, Automatic Advanced Matching captures hashed customer data including email addresses, phone numbers, first and last names; from checkout, sign-in and registration forms. We believe the leaks are due to Facebook's script interpreting clicks on irrelevant buttons as "submit button clicked" events.

Website categories In order to compare email exfiltration across website categories, we query McAfee's categorization service [74]. Note that a website may have multiple categories. As shown in Table 5, Fashion/Beauty and Online Shopping are the two categories where we detect the most email exfiltrations—considering only the categories with more than 1,000 websites in our 100K sample. On the other hand, websites categorized as Public Information, Government/Military, and Games leaked less than 1% of the filled email address. A somehow surprising result was the following: despite filling email fields on hundreds of websites categorized as Pornography, we have not a single email leak. While surprising, this is in line with limited prior research on tracking on the adult websites: a limited 2016 study by Altaweel et al. found that adult websites have relatively fewer third-party trackers compared to non-adult websites with comparable popularity [5].

Effect of website popularity The number of websites with email leaks follows a close to a uniform distribution in the US crawl. On the other hand, in the EU crawl, there are substantially fewer sites with email leaks on the Tranco top 5K: only 1.28% sites on the top 5K has email leaks, compared to the

average of 1.87% in websites with rank >5000 (cf. US top 5K: 2.96%, 5K-100K: 2.95%). Popular websites and trackers may be using questionable data collection methods sparingly in the EU to avoid GDPR fines or investigations.

Top websites with leaks Table 4 shows the top ten websites with email leaks for each vantage point. We list the third-party tracker found to collect emails on these sites, along with the hashing/encoding method used when exfiltrating the email. News websites such as usatoday.com, foxnews.com and independent.co.uk, appear high on the lists. This is in line with prior work which found that news websites contain the highest number of third parties compared to other website categories [32]. Medical news and information websites webmd.com and healthline.com are other notable entries for their sensitive content.

Emails sent key by key As shown in Table 2, certain third parties send email addresses character-by-character, as the user types in their address. This behavior appears to be due to session replay scripts that collect users' interactions with the page including key presses and mouse movements [2].

HTTP and WebSocket usage Finally, we observed that the leaked emails are almost always sent over encrypted (HTTPS) connections. We only found 15 and 14 websites where emails are leaked over HTTP in the EU and the US, respectively. In addition, on 67 websites in the EU and on 132 websites in the US, the leaks were sent over the WebSocket protocol—to hotjar.com, freshrelevance.com, noibu.com and decibelinsight.net.

4.2 Password Leaks

Recall that we change the type of password elements to *text* before filling them. To better understand why passwords are collected, we manually analyzed a sample of websites, including leaks to non-tracker third parties. We found that, in some cases, passwords were sent to third parties to check the password strength. However, we have not found such a use case in leaks to trackers. We found most cases we analyzed to be due to incidental collection by session recording scripts, most prominently by Yandex Metrica.

Password collection without input type swapping Since our primary findings are based on changing the type of the password field, they only apply to a limited number of users or websites. In order to better characterize password leaks at large, we ran follow up crawls of websites where we detected a password leak; but this time we did not change the input type from *password* to *text*. We ran two such crawls, one from the EU, and one from the US; both desktop crawls. Unless otherwise specified, password leaks presented throughout this

Table 5: Per-category number of websites we crawled, filled an email field, and observed an email leak to a tracker domain (based on desktop crawls using the no-action mode). The percentage under the Leaky sites column is based on total websites where we could fill an email field (i.e. 100 * Num. of leaky sites / Num. of filled sites).

	EU/US	**EU**		**US**	
Categories	**Sites**	**Filled sites**	**Leaky sites**	**Filled sites**	**Leaky sites**
Fashion/Beauty	1,669	1,176	131 (11.1%)	1,179	224 (19.0%)
Online Shopping	5,395	3,658	345 (9.4%)	3,744	567 (15.1%)
General News	7,390	3,579	235 (6.6%)	3,848	392 (10.2%)
Software/Hardware	4,933	2,834	138 (4.9%)	2,855	162 (5.7%)
Business	13,462	7,805	377 (4.8%)	7,924	484 (6.1%)
...	...	...	...	...	...
Games	2,173	925	9 (1.0%)	896	11 (1.2%)
Public Information	2,346	1,049	8 (0.8%)	1,084	27 (2.5%)
Gov't/Military	3,754	939	5 (0.5%)	974	7 (0.7%)
Uncategorized	1,616	636	3 (0.5%)	646	2 (0.3%)
Pornography	1,388	528	**0 (0.0%)**	645	**0 (0.0%)**

paper are based on these latter crawls, without input type swapping. We found that passwords are collected by trackers on 52 distinct websites even for users who do not use ShowPassword or similar extensions. An overwhelming majority (50/52) of these leaks were due to Yandex Metrica's session recording feature. However, a manual analysis of Yandex Metrica's code showed that it has filters to exclude password fields from the collection. Comparing websites where Yandex collects passwords to websites where it does not, we found that almost all *leaky* websites were built using the React framework. Note that 7 of the 52 affected websites are in the Tranco top 20K, and some of them are major banks and other highly visible websites such as toyota.ru. We have already reported this problem to Yandex, and reached out to the affected first parties as explained in Section 6.

4.3 Vantage Points: EU vs. US

In this section, we compare the results from our two crawl vantage points: the EU (Germany) and the US (NYC). The differences in privacy regulations are the main motivation behind this comparison. In the US crawl, the number of websites with email leaks is 60% higher than that of the EU: 1,844 versus 2,950.

Comparing the websites where we detected an email leak, we find that 2,950 websites identified in the US crawl are roughly a superset of the (1,844) websites identified in the EU crawl: 94.4% of the 1,844 websites detected in the EU crawl also appears in the list of websites in the US crawl.

Tracker domains such as addthis.com, yahoo.com, doubleclick.net and criteo.com only seem to receive email addresses in the US crawls, perhaps due to stricter data protection regulations in the EU. In addition, the most prominent email collecting tracker across both crawls (rlcdn.com, LiveRamp), is not even among the top ten trackers in the EU in Table 2.[9] In certain cases, the same tracking script is served with different content based on the vantage point. For instance, securedvisit.com, the tracker that uses a substitution cipher to encrypt its payload (Section 3.7), serves a slightly different script to EU visitors that disables email collection.

Overall, our results appear to indicate that certain third parties avoid collecting EU visitors' email addresses. In Section 5, we provide a legal analysis of whether the practice of collecting emails before form submission complies with the GDPR.

4.4 The Effect of Consent

Recall that, we found consent popups only on 7,720 (7.7%) sites in the EU and 5,391 (5.4%) sites in the US (of 100K sites). Crawling these websites with three consent modes, we obtain the results in Table 6, which shows the number of websites where we detect CMPs and email leaks to trackers. When we reject all data processing, the number of sites with leaks to trackers decreases by 13% in the US, 0.05% in the EU. The reduction in leaks in both cases is limited confirming Papadogiannakis et al.'s conclusion that cookie consent choices are not effective in preventing tracking [64]. Almost no reduction in the EU leaks, however, may be counter-intuitive. This is likely due to the limited number of websites where we could detect CMPs and observe leaks.

4.5 Mobile

We detected leaks on 1,745 and 2,744 distinct mobile websites in the EU and US crawls, respectively (Table 7). Although the number of sites with leaks is lower compared to desktop crawls, the ratio of the sites with leaks to the sites where we could fill email is nearly the same in both vantage points.

[9] In fact, LiveRamp sent a 451 HTTP error code (*Unavailable For Legal Reasons*) in responses to requests made in the EU crawl.

Table 6: The number of distinct websites where emails were leaked and a CMP was detected in desktop crawls using the no-action mode.

Consent modes	EU	US
accept-all	239	242
reject-all	201	199
no-action	202	228

The mobile and desktop websites where emails are leaked to tracker domains overlap substantially but not completely. The Jaccard similarity of (leaky) desktop and mobile websites is equal to 66% in the EU and 64% in the US. The difference between the desktop and mobile results could be due to website dynamism and the time difference between the mobile and desktop crawls (more than a month).

We also found 18 tracker domains that only received email leaks on mobile crawls such as yieldify.com, td3x.com and getdrip.com. However, checking the websites associated with these domains did not suggest that they are only targeting mobile web visitors. Further, we found 24 domains that only appear in desktop crawls, further indicating that the difference could be due to factors such as time difference and website dynamism.

Table 7: The number of sites leaking emails or passwords to trackers, compared to the number of sites where we could fill an email address in desktop and mobile crawls using the no-action mode.

	Leaky/ Filled Sites EU	Leaky/ Filled Sites US
Desktop	1,844 / 60,008 (3.0%)	2,950/ 60,999 (4.8%)
Mobile	1,745 / 55,738 (3.1%)	2,744 / 57,715 (4.8%)

4.6 Emails Received on the Filled Addresses

Since our crawler fills a distinct email address for each website, we are able to attribute the received emails to distinct websites.[10] In the six-week period following the crawls, we received 290 emails from 88 distinct sites on the email addresses used in the desktop crawls, despite not submitting any form. Most

[10]A caveat to our method is the following: we did not use separate email addresses for the EU and the US crawls, thus we cannot attribute the received emails to visits from specific locations.

emails offer a discount, or just invite us back to their site. The sender websites seem to vary by topic and theme. Most notable examples include diabetes.org.uk, mypillow.com, and walmart.com.mx. On the mobile crawl email address, we received 187 emails from 71 distinct websites following the four-week period after the crawls—mobile crawls were run two weeks after the desktop crawls.

5. Does Email Exfiltration Comply With the GDPR?

In this section, we discuss how email exfiltration can breach at least three core rules of the General Data Protection Regulation (GDPR) [14]. Roughly speaking, the GDPR could be seen as a Europe-wide data privacy law. Because of length constraints, we focus on three main principles of the GDPR, omitting greater detail.

We discuss email exfiltration in general. We do not discuss to what extent specific companies comply with the GDPR. For such a company-specific analysis, each example of email exfiltration would have to be assessed separately, considering all the circumstances of that case.

Does the GDPR apply? The GDPR applies when 'personal data' are processed. Personal data are defined broadly in the GDPR. Essentially, any information that relates to an identifiable person is personal data (Article 4.1). For instance, an email address, an IP address, a tracking cookie, an identification number, and an 'online identifier' are almost always personal data. But even hashed or encrypted email addresses are generally personal data, as far as they contain a unique identifier that can be linked to a person [88]. Moreover, hashed email addresses can often be reversed [1]. 'Processing' is defined broadly too in the GDPR: virtually everything that can be done with personal data is a type of processing (Article 4(2)). Hence, if website owners or third parties exfiltrate an email address, they process personal data and the GDPR applies.

An organization that processes personal data is a 'controller' in GDPR parlance. The 'controller' is responsible for complying with the GDPR, and can be fined for non-compliance. In the case of email exfiltration, the website owner and the third party are typically both responsible (as 'joint controllers') [15, 52].

Is the GDPR relevant for companies outside Europe? The territorial scope of the GDPR is complicated, but can be summarized as follows (Article 3 GDPR). If the controller is based in the EU, the GDPR applies. But the GDPR can also apply to controllers based outside the EU. For instance, offering goods or services to Europeans can trigger the GDPR. If a website owner sells something and allows payment in Euros, and processes the personal data of

website visitors, the owner must comply with the GDPR. The GDPR also applies to controllers based outside the EU, if they 'monitor' the behavior of people in the EU. Tracking people online is an example of such monitoring [42]. Hence, if a company uses email exfiltration for tracking web users in the EU, it must comply with the GDPR.

Transparency principle The GDPR has six overarching principles relating to the processing of personal data. The first principle says that personal data must be processed 'fairly and in a transparent manner' (Article 5). The controller must provide comprehensive information about what it does with personal data, in an 'intelligible and easily accessible form, using clear and plain language' (Article 11). Moreover, the GDPR requires detailed information about, for instance, the processing 'purposes', and the 'recipients of the personal data' (Article 13 and 14). Controllers can provide such information in a privacy notice.

Does email exfiltration comply with the transparency principle? If the website does not clearly disclose that it or a third party exfiltrates email addresses, the exfiltration breaches the transparency principle. A phrase such as 'we share your personal data with selected marketing partners' does not provide sufficient transparency.

Purpose limitation principle Does email exfiltration comply with the GDPR's purpose limitation principle? Roughly summarized, the purpose limitation says that controllers can only collect personal data if they specify a clear purpose in advance. And the controller is not allowed to use the data for 'incompatible' new other purposes (Article 6(1)(b)). Suppose that the first purpose is enabling website visitors to manage their website account. The first purpose will be something like 'remembering the website visitors' login credentials so that they can open and maintain an account'. Say that the third party uses the exfiltrated email address for behavioral advertising, email marketing or tracking people around the web. Those purposes are incompatible with the original purpose, and thus prohibited.

The requirement for a legal basis such as consent Another important GDPR requirement is that the controller always needs a 'legal basis' to process personal data (Article 6). There are six possible legal bases, including consent. The requirements for valid consent are strict. For instance, a consent request that is hidden in the small print of a contract or privacy notice cannot lead to valid consent. Further, a controller cannot assume consent if people fail to opt-out (Article 4(11)). The GDPR does not always require the person's consent. However, for online tracking and behavioral advertising, the GDPR does require prior consent [87, 82].

To obtain valid consent to collect website visitors' email addresses before they click submit, the consent request would have to be specific; such as: 'Do you agree with us collecting your email address and sharing it with company, A, B, and C for email marketing before you click submit?'. Only if the website visitor clearly agrees to such a request, the visitor gives valid consent to email exfiltration. If the request was vague, or if the visitor did not clearly express their choice, the consent is invalid.

In certain situations, email exfiltration might be allowed under the GDPR without the website visitor's consent. Suppose that a security firm (third party) exfiltrates a website visitor's email address for an extra security check. Assuming that the security firm complies with all the other GDPR norms, the firm could be allowed to exfiltrate the email address without consent (based on Article 6(1)(f)).

Conclusion Email exfiltration by third parties can breach at least three GDPR requirements. First, if such exfiltration happens surreptitiously, it violates the transparency principle. Second, if such exfiltration is used for purposes such as behavioral advertising, marketing and online tracking, it also breaches the purpose limitation principle. Third, if the email exfiltration is used for behavioral advertising or online tracking, the GDPR typically requires the website visitor's prior consent. For breaching any of these three rules, controllers can be fined up to 20,000,000 Euro or up to 4% of their total worldwide annual turnover (Article 83(5)).

6. Security Disclosures, GDPR Requests, and Leak Notifications

Our methods allow us to detect email and password leaks from clients to trackers, but what happens after the leaks reach third party's servers is unknown to us. In order to better understand the server-side processing of collected emails, and to disclose cases of password collection, we have reached out to more than a hundred first and third parties. We used the real identity and university email account of one of the authors when reporting the issues or sending the GDPR requests. Moreover, we made it clear that our inquiries are sent within the context of an academic research.

Password collection disclosures Once again we note that we believe all password leaks to third parties mentioned below are incidental. We reached out to all third parties listed in Table 2. Yandex, the most prominent tracker that collects users' passwords, has quickly responded to our disclosure and rolled out a fix to prevent password collection. We have also notified more than 50

websites where passwords were collected. Since the majority of the websites embedding Yandex were in Russian, we have enclosed a Russian translation of our message in the notification email, along with our message in English. Mixpanel released an update only two days after we disclosed the issue. With this change, even the users with outdated SDKs were protected from collecting passwords involuntarily. LogRocket, who collected passwords on publicize.co's login page, have never replied to our repeated contact attempts[11]; and the password leak remained on Publicize's website for more than ten weeks, before it was fixed.

GDPR requests on email exfiltration We reached out to 58 first and 28 third parties with GDPR requests. We avoided sending blanket data access requests to minimize the overhead for the entities who were obliged to respond to our GDPR requests. Instead, we asked specific questions about how the collected emails are processed, retained and shared. In addition, we notified the top 33 websites[12] where we detected email exfiltration in the US crawl. We sent a friendly notification to these websites about the email exfiltration, rather than a formal GDPR request. We did not get any response from these 33 websites.

When selecting the first parties to send GDPR requests to, we included the most popular websites from the EU crawl, for which we could reproduce the email leaks. We asked the first parties if they were aware of the email collection on their websites, how they used the collected email addresses, and how long they retained them.

Responses from first parties: Almost half of the first parties (30/58) responded to our requests.

- fivethirtyeight.com (via Walt Disney's DPO), trello.com (Atlassian), lever.co, branch.io and cision.com were among the websites that said they had not been aware of the email collection prior to form submission on their websites and removed the behavior.
- Marriott said that the information collected by Glassbox is used for purposes including customer care, technical support, and fraud prevention.
- Tapad, a cross-device tracking company on whose website we found an email leak, said that they are not offering their services to UK & EEA users since August, 2021; and they have deleted all data that they held from these regions.

[11]We have also enrolled the help of a contact at the Electronic Frontier Foundation, who tried calling LogRocket's phone number, emailed their privacy contact address, and their cofounder—all to no avail. Our attempts to disclose the issue via LogRocket's chatbot have also failed. We have also contacted Publicize, and have not heard back.

[12]33 out of the top 50 websites for which we could reproduce the exfiltration.

- stellamccartney.com explained that the emails on their websites were collected before the submission due to a technical issue, which was fixed upon our disclosure. According to their response, the SaleCycle script that collected email addresses had not been visible to their cookie management tool from OneTrust.

Responses from third parties: Roughly half (15/28) of the third parties responded. Eight third parties, including Adobe, FullStory and Yandex said they are data processors, and asked us to send our GDPR request to the corresponding first parties.

- Taboola said in certain cases they collect users' email hashes before form submission for ad and content personalization; they keep email hashes for at most 13 months; and they do not share them with other third parties. Taboola also said they only collect email hashes after getting user consent; however, our findings and subsequent manual verification showed that was not always the case.
- Zoominfo said their "FormComplete" product appends contact details of users to forms, when the user exists in ZoomInfo's sales and marketing database. They said the ability to capture form data prior to submission can be enabled or disabled by their clients.
- ActiveProspect said their TrustedForm product is used to certify consumer's *consent to be contacted* for compliance with regulations such as the Telephone Consumer Protection Act in the US. They said data captured from abandoned forms are marked for deletion within 72 hours, is not shared with anyone including the site owner.

We picked the above responses to reflect the diversity of reasons for which email addresses are collected prior to form submission. While some collection reportedly occurs due to technical glitches, or (surprisingly) for compliance purposes; other responses point to collection for marketing, analytics and identity matching purposes. In certain cases, companies suggested that the email data are not shared with any third parties, while others have not made the same promise. The limited number of responses we received, along with potential response bias, prevent us from making generalizations. Regardless, we note the benefit of reaching out to the respective parties, despite the substantial logistics overhead. Due to limited space, we could only include a selection of the responses. We plan to publish an overview of the responses as part of our dataset.

7. Countermeasures

In recent years, all major browsers except Google Chrome implemented different forms of protection against online tracking. In 2017, Apple introduced Safari Intelligent Tracking Prevention (ITP), which combines machine learning with a rule-based system that prevents cross-site tracking [85]. Since March 2020, Safari blocks all third-party cookies [86]. Mozilla introduced tracking protection in 2018 by stripping cookies from requests to tracker domains, based on a tracker list compiled by Disconnect [60, 20].

In order to find out whether major browsers with anti-tracking features (namely, Safari and Firefox) block the exfiltrations we uncovered, we manually analyzed ten websites, each containing a distinct tracker that we found to exfiltrate email addresses. We manually filled the email fields on these websites and checked whether the exfiltration occurs by inspecting the HTTP request payloads in the devtools interface. We found that neither Safari nor Firefox blocked email exfiltrations to tracking endpoints in our small sample. This result may be expected since both browsers try to strike a balance between minimizing breakage and curtailing cross-site tracking. To this end, they allow requests to tracker domains, but they strip cookies, partition network state [30], or block access to storage that may facilitate cross-site tracking.

Browser vendors may take further steps to protect against scripts that harvest email addresses for tracking purposes. Browsers may block requests to these trackers, prevent their scripts from accessing form fields, or provide them with fake data—e.g., an empty string similar to how a zero-filled IDFA is returned on iOS devices unless the user has given their consent [7]. Similar solutions are already used by different vendors: Firefox already blocks requests to third parties that use browser fingerprinting for advertisement, analytics and social network tracking [13]. DuckDuckGo's browser extension uses JavaScript stack traces to block certain tracker cookies [21]. We believe the scale of unconsented data collection uncovered in our study justifies a similar countermeasure for scripts that harvest email addresses.

Browser extensions such as uBlock Origin [80], and browsers such as Brave [9] block requests to tracker domains, which better protects against email exfiltration than countermeasures built-in to Firefox and Safari. On mobile, users may opt for browsers that support extensions (e.g., Firefox, Safari), or use a privacy-focused mobile browser that blocks trackers such as Brave [9] and DuckDuckGo [22].

Recently, Mozilla [58], Apple [6], and DuckDuckGo [28] started to offer private email relay services that give users the ability to generate and use pseudonymous

(alias) email addresses. These privacy-focused services automatically forward emails received at the alias addresses, and allow users to keep their real email address hidden from untrusted online services.

In their study on data exfiltration from contact forms, Starov et al. developed FormLock, an extension that detects and highlights forms that may leak PII. Further, to prevent PII leakage, FormLock temporarily blocks third-party requests and prevents stashing of PII into various storage mechanisms such as cookies, localStorage and indexedDB [76].

LeakInspector Since none of the available countermeasures allow inspection of sniff and exfiltration attempts, we developed LeakInspector, a proof-of-concept browser add-on that warns users against sniff attempts and blocks requests containing personal information.

While LeakInspector has similarities to FormLock, it also supports detecting form sniff attempts and more precisely detects and prevents leak attempts to trackers. Further, LeakInspector does not require user intervention, and logs technical details of the detected sniff and leak attempts to console to enable technical audits. The logged information includes the value and XPath of the sniffed element, the origin of the sniffer script, and details of the leaky request such as URL and POST data. LeakInspector has two main features that users may enable:

Sniffer Detector When this feature is enabled, LeakInspector detects and optionally prevents sniffing of input fields where users may enter personal information such as name, email and credit card details. We use code extracted from Firefox's autofill field detection heuristics [41] to detect such input fields.

We overwrite the getter method of the HTMLInputElement prototype to intercept input field sniff attempts. We add an event listener for `input` event to all auto-fill fields to keep track of their current values. These input field values are then used to detect leaks in outgoing requests. When a script attempts to read a monitored field's value, LeakInspector processes the JavaScript stack trace and extract the script addresses. It then highlights the sniffed input field if there is a third-party script in the stack trace categorized as a tracker by DuckDuckGo's blocklist [24]–which we also use in Section 3.8. When determining third party scripts, LeakInspector takes into account domain-entity relationships [33].

Leak Detector LeakInspector intercepts HTTP requests and runs the leak detector algorithm presented in Section 3.7. It detects encoded, hashed, compressed or cleartext leaks from the monitored fields. While LeakInspector currently only uses DuckDuckGo's blocklist [24], it is possible to extend it to use other blocklists.

LeakInspector also features a user interface where recent sniff and leak attempts are listed, along with the tracker domain, company and tracker category. The user interface module is based on DuckDuckGo's Privacy Essentials add-on [23]. We believe LeakInspector may help publishers and end-users to inspect third parties that harvest personal information from online forms without their knowledge and consent.

8. Limitations

Through an iterative design process, pilot crawls and extensive sanity checking, we built our crawler and analysis processes to be robust and scalable. Where possible we set the parameters of the crawler such as timeout duration, based on data from pilot crawls. However, certain limitations apply to our data collection and analysis methods.

Leak detection While we search for an extensive set of encodings and hashes, and we substantially improved the leak detector module we inherited from the prior work, our leak detection method may still miss leaks that are custom encoded, encrypted, or compressed. Future work may improve leak detection by applying methods such as multi-stage filtering [44], and JavaScript information flow tracking [40].

Shadow DOM and crawl depth During our pilot crawls we found that we cannot detect email and password fields if they are in the Shadow DOM [70] of other elements. Since we only found two such cases in a pilot crawl of 1K websites, we believe this is an acceptable limitation. Further, our crawler is limited to crawls of one-click depth for simplicity. Input fields that can only be discovered through multiple subsequent clicks may be missed by our crawler. These limitations make our results likely lower bounds.

Blocklists We use a combination of blocklists from different providers to flag domains as trackers. These lists vary by quality and compilation method (e.g., crowdsourced vs. maintained by a company such as Disconnect). Further, we flag domains as trackers if they are present in only one of these lists. As such, our results may have both false positives and false negatives due to imperfections in those blocklists.

Domain aliases Although we only consider leaks to third-party tracker domains, we also analyzed a sample of exfiltration to first-party domains. The use cases we identified included email address verification and self-hosted analytics services. Future work could investigate exfiltrations to CNAME-based trackers that appear as first parties [19].

Bypassing cookie consent banners During the manual labeling process, we encountered modal GDPR consent dialogs that disallow proceeding without giving/rejecting to give consent. A real user would have to accept or reject data processing to interact with the page; but our web crawler could have bypassed the consent dialog, depending on how it is implemented. On a random sample of 1,000 websites, we detected 168 modal consent dialogs.

Anti-bot measures Finally, our crawler might have been served CAPTCHA pages, or treated differently due to crawling from cloud IP addresses. During a 1K website pilot crawl, we identified only three CloudFlare CAPTCHA pages that blocked our crawler.

9. Ethics Considerations

Data collection: When crawling, we took adequate measures to avoid overloading the websites. For instance, we avoided making concurrent visits to the same website.

Disclosures: We reported password leaks to both trackers and to the websites where we detected a password leak. In our emails, we provided technical details and reproduction instructions so that it is easier for the parties to reproduce and address the issue we reported. To the third parties, we sent the list of websites where they caused a password leak. To avoid any misunderstanding, we made it clear to all parties that we did not collect any visitors' email or password during our study. We did not send GDPR requests to trackers that incidentally collected passwords.

10. Conclusion

We presented a large-scale study of email and password exfiltration by online trackers before form submission. In order to address the challenges of finding and filling input fields, we integrated into our crawler a pre-trained ML classifier that detects email fields. Our results—likely lower bounds—show that on thousands of sites email addresses are collected from login, registration and newsletter subscription forms; and sent to trackers before users submit any form or give their consent. Further, we found tens of sites where passwords are incidentally collected by third parties providing session replay services. Comparing results from the EU and the US vantage points, we found that 60% more websites leaked users' emails to trackers, when visited from the US. Measuring the effect of consent choices on the exfiltration, we found their effect to be minimal. Based on our findings, users should assume that the personal

information they enter into web forms may be collected by trackers—even if the form is never submitted. Considering its scale, intrusiveness and unintended side-effects, the privacy problem we investigate deserves more attention from browser vendors, privacy tool developers, and data protection agencies.

Code and Data

The source code and the dataset from our study are publicly available at `https://github.com/leaky-forms`.

Acknowledgments

We thank Alexei Miagkov, Arvind Narayanan, Bart Jacobs, Claudia Diaz, David Roefs, Dorine Gebbink, Galina Bulbul, Gwendal Le Grand, Konrad Dzwinel, Pete Snyder, Sergey Galich, Steve Englehardt, Vincent Toubiana, our shepherd Alexandros Kapravelos, SecWeb and USENIX Security reviewers for their valuable comments and contributions. The idea for measuring email exfiltration before form submission is initially developed with Steve Englehardt and Arvind Narayanan during an earlier study [2]. Asuman Senol was funded by the Cyber-Defence (CYD) Campus of armasuisse Science and Technology. Gunes Acar was initially supported by a postdoctoral fellowship from the Research Foundation Flanders (FWO). The study was supported by CyberSecurity Research Flanders with reference number VR20192203.

References

[1] Gunes Acar. Four cents to deanonymize: Companies reverse hashed email addresses. `https://freedom-to-tinker.com/2018/04/09/four-cents-to-deanonymize-companies-reverse-hashed-email-addresses/`, 2018. [Online; accessed 11. May. 2021].

[2] Gunes Acar, Steven Englehardt, and Arvind Narayanan. No boundaries: data exfiltration by third parties embedded on web pages. *Proceedings on Privacy Enhancing Technologies*, 2020(4):220–238, 2020.

[3] Gunes Acar, Marc Juarez, Nick Nikiforakis, Claudia Diaz, Seda Gürses, Frank Piessens, and Bart Preneel. FPDetective: Dusting the Web for Fingerprinters. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*, pages 1129–1140, 2013.

[4] Seyed Ali Akhavani, Jordan Jueckstock, Junhua Su, Alexandros Kapravelos, Engin Kirda, and Long Lu. Browserprint: An Analysis of the Impact of Browser Features on Fingerprintability and Web Privacy. In *International Conference on Information Security*, pages 161–176. Springer, 2021.

[5] Ibrahim Altaweel, Maximillian Hils, and Chris Jay Hoofnagle. Privacy on adult websites. In *Altaweel et al., Privacy on Adult Websites, Workshop on Technology and Consumer Protection (ConPro)*, 2017.

[6] Hide My Email for Sign in with Apple. `https://support.apple.com/en-us/HT210425`. [Online; accessed 11. May. 2023].

[7] advertisingIdentifier | Apple Developer Documentation. `https://developer.apple.com/documentation/adsupport/asidentifiermanager/1614151-advertisingidentifier`. [Online; accessed 11. May. 2023].

[8] Rolf Bagge, Célestin Matte, éric Daspet, Kaspar Emanuel, Sam Macbeth, and Steven Roeland. Consent-O-Matic. `https://github.com/cavi-au/Consent-O-Matic/`, 2019. [Online; accessed 11. Jun. 2023].

[9] Secure, Fast & Private Web Browser with Adblocker | Brave Browser. `https://brave.com/`. [Online; accessed 11. May. 2023].

[10] Sai Prashanth Chandramouli, Pierre-Marie Bajan, Christopher Kruegel, Giovanni Vigna, Ziming Zhao, Adam Doupé, and Gail-Joon Ahn. Measuring E-Mail Header Injections on the World Wide Web. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, pages 1647–1656, 2018.

[11] Manolis Chatzimpyrros, Konstantinos Solomos, and Sotiris Ioannidis. You Shall Not Register! Detecting Privacy Leaks Across Registration Forms. In *Computer Security*, pages 91–104. Springer, 2019.

[12] What is a "good" conversion rate for your signup flow? `https://heap.io/blog/good-conversion-rate-signup-flow`, 2017. [Online; accessed 08. Dec. 2021].

[13] Cookie Status: Current Status Of Browser Tracking Prevention | cookiestatus.com. `https://www.cookiestatus.com`. [Online; accessed 22. Jun. 2023].

[14] Council of European Union. EU General Data Protection Regulation (GDPR). `https://eur-lex.europa.eu/eli/reg/2016/679/oj`, 2016. [Online; accessed 11. Jan. 2022].

[15] Court of Justice of the European Union, Case C-40/17,Fashion ID GmbH & Co. KG v Verbraucherzentrale NRW e.V., judgment of 29 July 2019 (ECLI:EU:C:2019:629). https://curia.europa.eu/juris/liste.jsf?num=C-40/17, 2019. [Online; accessed 08. Sep. 2022].

[16] Adding Rank Magnitude to the CrUX Report in BigQuery. https://developers.google.com/web/updates/2021/03/crux-rank-magnitude, 2023. [Online; accessed 22. Feb. 2024].

[17] Ha Dao and Kensuke Fukuda. Alternative to third-party cookies: investigating persistent PII leakage-based web tracking. In *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies*, pages 223–229, 2021.

[18] Data Services API: Endpoints. https://developer.myacxiom.com/code/api/endpoints/hashed-entity. [Online; accessed 22. Feb. 2023].

[19] Yana Dimova, Gunes Acar, Lukasz Olejnik, Wouter Joosen, and Tom Van Goethem. The CNAME of the Game: Large-scale Analysis of DNS-based Tracking Evasion, 2021-07-12.

[20] Disconnect. Disconnect Tracking Protection. https://github.com/disconnectme/disconnect-tracking-protection, 2023. [Online; accessed 11. Jun. 2023].

[21] DuckDuckGo. DuckDuckGo Privacy Essentials browser extension. https://github.com/duckduckgo/duckduckgo-privacy-extension/blob/bfbd47a/shared/js/content-scope/tracking-cookies-1p-protection.js#L30, 2021. [Online; accessed 29. Nov. 2023].

[22] DuckDuckGo Privacy Browser - Apps on Google Play. https://play.google.com/store/apps/details?id=com.duckduckgo.mobile.android, 2022.

[23] DuckDuckGo Browser Extensions. https://github.com/duckduckgo/duckduckgo-privacy-extension, 2023. [Online; accessed 11. Jan. 2023].

[24] DuckDuckGo Tracker Blocklist. https://staticcdn.duckduckgo.com/trackerblocking/v2.1/tds.json, 2023. [Online; accessed 11. May. 2023].

[25] Tracker Radar Collector. https://github.com/duckduckgo/tracker-radar-collector, 2020. [Online; accessed 08. Sep. 2021].

[26] Peter Eckersley. How unique is your web browser? In *Proceedings of the 10th International Conference on Privacy Enhancing Technologies (PETS)*, pages 1–18, 2010.

[27] email_detector.js – Private Relay. https://github.com/mozilla/fx-private-relay/blob/v1.2.2/extension/js/email_detector.js, 2023. [Online; accessed 08. Sep. 2022].

[28] Introducing Email Protection: The easy way to block email trackers and hide your address. https://spreadprivacy.com/introducing-email-protection-beta/. [Online; accessed 22. Feb. 2021].

[29] Steve Englehardt, Gunes Acar, and Arvind Narayanan. No boundaries for credentials: New password leaks to Mixpanel and Session Replay Companies. https://freedom-to-tinker.com/2018/02/26/, 2018. [Online; accessed 11. May. 2023].

[30] Steven Englehardt and Arthur Edelstein. Firefox 85 Cracks Down on Supercookies – Mozilla Security Blog. https://blog.mozilla.org/security/2021/01/26/supercookie-protections, 2021. [Online; accessed 11. May. 2023].

[31] Steven Englehardt, Jeffrey Han, and Arvind Narayanan. I never signed up for this! Privacy implications of email tracking. *Proceedings on Privacy Enhancing Technologies (PETS)*, 2018(1):109–126, 2018.

[32] Steven Englehardt and Arvind Narayanan. Online Tracking: A 1-million-site Measurement and Analysis. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1388–1401, 2016.

[33] entity_map.json - DuckDuckGo Tracker Radar. https://github.com/duckduckgo/tracker-radar/blob/main/build-data/generated/entity_map.json, 2023. [Online; accessed 11. Jun. 2023].

[34] Evercookie - Virtually irrevocable persistent cookies. https://samy.pl/evercookie. [Online; accessed 08. Sep. 2021].

[35] Optimise: Automatic advanced matching. https://www.facebook.com/business/m/signalshealth/optimize/automatic-advanced-matching, 2021. [Online; accessed 11. May. 2023].

[36] Fathom documentation. https://mozilla.github.io/fathom/, 2021. [Online; accessed 11. May. 2023].

[37] Firefox Password Manager Module. https://searchfox.org/mozilla-central/source/toolkit/components/passwordmgr/NewPasswordModel.jsm. [Online; accessed 11. Jun. 2023].

[38] About the customer matching process - Google Ads Help. https://support.google.com/google-ads/answer/7474263?hl=en, 2021. [Online; accessed 29. Nov. 2023].

[39] Hashing Identifiers. https://docs.liveramp.com/connect/en/hashing-identifiers.html, 2021. [Online; accessed 08. Dec. 2021].

[40] Daniel Hedin, Arnar Birgisson, Luciano Bello, and Andrei Sabelfeld. JSFlow: Tracking information flow in JavaScript and its APIs. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, pages 1663–1671, 2014.

[41] heuristicsRegexp.js - Mozilla Autofill. https://searchfox.org/mozilla-central/source/toolkit/components/formautofill/content/heuristicsRegexp.js, 2023. [Online; accessed 11. Jun. 2023].

[42] Chris Jay Hoofnagle, Bart van der Sloot, and Frederik Zuiderveen Borgesius. The European Union general data protection regulation: what it is and what it means. *Information & Communications Technology Law*, 28(1):65–98, 2019.

[43] Umar Iqbal, Steven Englehardt, and Zubair Shafiq. Fingerprinting the fingerprinters: Learning to detect browser fingerprinting behaviors. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1143–1161. IEEE, 2021.

[44] Sakshi Jain, Mobin Javed, and Vern Paxson. Towards Mining Latent Client Identifiers from Network Traffic. *Proc. Priv. Enhancing Technol.*, 2016(2):100–114, 2016.

[45] Hugo Jonker, Stefan Karsch, Benjamin Krumnow, and Marc Sleegers. Shepherd: A generic approach to automating website login. In *Workshop on Measurements, Attacks, and Defenses for the Web (MADWeb)*, 2020.

[46] Jordan Jueckstock and Alexandros Kapravelos. VisibleV8: In-browser Monitoring of JavaScript in the Wild. In *Proceedings of the Internet Measurement Conference*, pages 393–405, 2019.

[47] Tomer Laor, Naif Mehanna, Antonin Durey, Vitaly Dyadyuk, Pierre Laperdrix, Clémentine Maurice, Yossi Oren, Romain Rouvoy, Walter Rudametkin, and Yuval Yarom. DRAWNAPART: A Device Identification Technique based on Remote GPU Fingerprinting. In *Network and Distributed System Security Symposium (NDSS)*, 2022.

[48] Pierre Laperdrix, Nataliia Bielova, Benoit Baudry, and Gildas Avoine. Browser fingerprinting: A survey. *ACM Transactions on the Web (TWEB)*, 14(2):1–33, 2020.

[49] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS)*, 2019.

[50] Adam Lerner, Anna Kornfeld Simpson, Tadayoshi Kohno, and Franziska Roesner. Internet jones and the raiders of the lost trackers: An archaeological study of web tracking from 1996 to 2016. In *25th USENIX Security Symposium*, 2016.

[51] Xu Lin, Panagiotis Ilia, and Jason Polakis. Fill in the Blanks: Empirical Analysis of the Privacy Threats of Browser Form Autofill. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 507–519, 2020.

[52] René Mahieu and Joris Van Hoboken. Fashion-ID: Introducing a phase-oriented approach to data protection? *European Law Blog*, 2019.

[53] Célestin Matte, Nataliia Bielova, and Cristiana Santos. Do Cookie Banners Respect My Choice?: Measuring Legal Compliance of Banners from IAB Europe's Transparency and Consent Framework. In *IEEE Symposium on Security and Privacy (SP)*, pages 791–809, 2020.

[54] Surya Mattu and Kashmir Hill. Before You Hit 'Submit,' This Company Has Already Logged Your Personal Data. *Gizmodo*, 2017. `https://gizmodo.com/before-you-hit-submit-this-company-has-already-logge-1795906081`.

[55] Jonathan R Mayer and John C Mitchell. Third-Party Web Tracking: Policy and Technology. In *2012 IEEE Symposium on Security and Privacy*, pages 413–427. IEEE, 2012.

[56] Keaton Mowery and Hovav Shacham. Pixel Perfect: Fingerprinting Canvas in HTML5. *Proceedings of W2SP*, 2012, 2012.

[57] Mozilla. Enhanced Tracking Protection in Firefox for desktop. `https://support.mozilla.org/en-US/kb/enhanced-tracking-protection-firefox-desktop`. [Online; accessed 08. Sep. 2021].

[58] Mozilla Relay | Protect your real email address to help control your inbox. `https://relay.firefox.com/`. [Online; accessed 11. May. 2023].

[59] Privacy | Neustar. `https://www.home.neustar/privacy`, 2021. [Online; accessed 29. Nov. 2023].

[60] Nick Nguyen. Changing Our Approach to Anti-tracking – Future Releases. https://blog.mozilla.org/futurereleases/2018/08/30/changing-our-approach-to-anti-tracking, 2018. [Online; accessed 08. Sep. 2021].

[61] Midas Nouwens, Ilaria Liccardi, Michael Veale, David Karger, and Lalana Kagal. Dark Patterns after the GDPR: Scraping Consent Pop-Ups and Demonstrating Their Influence. In *CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

[62] Lukasz Olejnik, Tran Minh-Dung, and Claude Castelluccia. Selling Off Privacy at Auction. *In Network and Distributed System Security Symposium (NDSS)*, 2014.

[63] Sending oHashes to Oracle Data Cloud platform. https://docs.oracle.com/en/cloud/saas/data-cloud/data-cloud-help-center/IntegratingBlueKaiPlatform/IDManagement/sending_ohashes.html. [Online; accessed 29. Nov. 2023].

[64] Emmanouil Papadogiannakis, Panagiotis Papadopoulos, Nicolas Kourtellis, and Evangelos P. Markatos. User Tracking in the Post-cookie Era: How Websites Bypass GDPR Consent to Track Users. In *Proceedings of the Web Conference 2021*, pages 2130–2141, 2021.

[65] People-Based Marketing In The Cookiepocalypse. https://dataq.ai/blog/the-rise-of-people-based-marketing/. [Online; accessed 08. Sep. 2021].

[66] Abbas Razaghpanah, Rishab Nithyanand, Narseo Vallina-Rodriguez, Srikanth Sundaresan, Mark Allman, Christian Kreibich, Phillipa Gill, et al. Apps, trackers, privacy, and regulators: A global study of the mobile tracking ecosystem. In *The 25th Annual Network and Distributed System Security Symposium*, 2018.

[67] Jingjing Ren, Ashwin Rao, Martina Lindorfer, Arnaud Legout, and David Choffnes. ReCon: Revealing and Controlling PII Leaks in Mobile Network Traffic. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, pages 361–374, 2016.

[68] Franziska Roesner, Tadayoshi Kohno, and David Wetherall. Detecting and Defending Against Third-Party Tracking on the Web. In *9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*, pages 155–168, 2012.

[69] Md Main Uddin Rony, Naeemul Hassan, and Mohammad Yousuf. Diving Deep into Clickbaits: Who Use Them to What Extents in Which Topics with What Effects? In *Proceedings of the 2017 IEEE/ACM International*

Conference on Advances in Social Networks Analysis and Mining, pages 232–239, 2017.

[70] Using shadow DOM. https://developer.mozilla.org/en-US/docs/Web/Web_Components/Using_shadow_DOM. [Online; accessed 08. Sep. 2022].

[71] Shobha Doshi. CafeMedia integrates with Verizon Media ConnectID. https://cafemedia.com/integrating-with-verizon-media-connectid/, 2021. [Online; accessed 29. Nov. 2023].

[72] ShowPassword - Chrome Web Store. https://chrome.google.com/webstore/detail/showpassword/bbiclfnbhommljbjcoelobnnnibemabl. [Online; accessed 08. Sep. 2021].

[73] Signup Abandonment Emails Case Study: How Drip Increased Trial Signups by 15%. https://www.saasemailmarketing.net/articles/signup-abandonment-emails-increase-trial-signups/. [Online; accessed 08. Dec. 2023].

[74] Customer URL Ticketing System. https://www.trustedsource.org. [Online; accessed 22. Feb. 2024].

[75] Oleksii Starov, Phillipa Gill, and Nick Nikiforakis. Are You Sure You Want to Contact Us? Quantifying the Leakage of PII via Website Contact Forms. *Proc. Priv. Enhancing Technol.*, 2016(1):20–33, 2016.

[76] Oleksii Starov, Phillipa Gill, and Nick Nikiforakis. FormLock. https://github.com/ostarov/Formlock, 2021. [Online; accessed 11. May. 2023].

[77] Why Taboola? https://pubhelp.taboola.com/hc/en-us/articles/360003157074-Why-Taboola-. [Online; accessed 11. May. 2023].

[78] Uploading and Targeting a Customer File. https://help.taboola.com/hc/en-us/articles/360021908874-Uploading-and-Targeting-a-Customer-File, 2021. [Online; accessed 29. Nov. 2023].

[79] 6 Steps for Avoiding Online Form Abandonment. https://themanifest.com/web-design/blog/6-steps-avoid-online-form-abandonment, 2022. [Online; accessed 29. Nov. 2023].

[80] gorhill/uBlock: uBlock Origin – An efficient blocker for Chromium and Firefox. Fast and lean. https://github.com/gorhill/uBlock, 2021. [Online; accessed 11. Sep. 2023].

[81] Steven Van Acker, Daniel Hausknecht, and Andrei Sabelfeld. Measuring Login Webpage Security. In *Proceedings of the Symposium on Applied Computing*, pages 1753–1760, 2017.

[82] Michael Veale and Frederik Zuiderveen Borgesius. Adtech and Real-Time Bidding under European Data Protection Law. *German Law Journal*, 2021.

[83] yahoo-connectid/sync.spec.js. `https://github.com/yahoo/yahoo-connectid/blob/d0b56d47a7/src/sync.spec.js#L33-L34`, 2021. [Online; accessed 29. Nov. 2023].

[84] whotracks.me | Data from the largest and longest measurement of online tracking. `https://github.com/ghostery/whotracks.me`, 2017. [Online; accessed 11. May. 2023].

[85] John Wilander. Intelligent Tracking Prevention. `https://webkit.org/blog/7675/intelligent-tracking-prevention`, 2017. [Online; accessed 29. Nov. 2023].

[86] John Wilander. Full Third-Party Cookie Blocking and More. `https://webkit.org/blog/10218/full-third-party-cookie-blocking-and-more`, 2020. [Online; accessed 29. Nov. 2023].

[87] Article 29 Working Party, 'Opinion 03/2013 on purpose limitation' (WP 203), 2 April 2013. `https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2013/wp203_en.pdf`. [Online; accessed 08. Sep. 2022].

[88] Article 29 Working Party, 'Opinion 05/2014 on Anonymisation Techniques' (WP 216) 10 April 2014. `https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf`. [Online; accessed 08. Sep. 2022].

[89] Zhiju Yang and Chuan Yue. A comparative measurement study of web tracking on mobile and desktop environments. *Proceedings on Privacy Enhancing Technologies*, 2020(2), 2020.

[90] Eric Zeng, Tadayoshi Kohno, and Franziska Roesner. Bad News: Clickbait and Deceptive Ads on News and Misinformation Websites. In *Workshop on Technology and Consumer Protection (ConPro)*, 2020.

Appendix

Supported Hash and Encoding Methods for Leak Detection

Hashes and Checksums: MD2, MD4, MD5, SHA1, SHA256, SHA224, SHA384, SHA512, SHA3 (224, 256, 384, 512-bit), MurmurHash3 (32, 64, 128-bit), RIPEMD-160, Whirlpool, Salted SHA1 (salt=`QX4QkKEU`) **Encodings**: Base16, Base32, Base58, Base64, Urlencode, Entity, Deflate, Zlib, Gzip, LZstring, Custom Map (`kibp8A4EWRMKHa7gvyz1dOPt6UI5xYD3nqhVwZBXfCcFe...` `0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghi...`)

Labeling email-collecting 3rd-party domains that are not blocked by blocklists

For each domain:

1. Is the 3rd-party domain is owned by the same entity as the first party?
   a. Yes: not tracking-related (first-party exception)
2. Did we receive any email from websites where this domain collected email addresses?
   a. Yes: tracking-related
3. Identify the company website—use the initiator script (URL, source code, copyright preamble, comments) if necessary.
   a. Is the 3rd party used for email validation (check on an example first-party site taking into account UI messages (e.g. “Invalid email”) and HTTP response content (e.g., “bogus email” when we enter test@gmail.com)?
      i. Yes: not tracking-related (validation exception)
   b. Identify the business category using BuiltWith and the company website (esp. check for solutions, products, and other marketing materials). Does the business category include one of marketing, advertising, analytics?
      i. Yes: tracking-related
      ii. No: not tracking-related

Publication

Unveiling the Impact of User-Agent Reduction and Client Hints: A Measurement Study

Publication Data

Senol, A., and Acar, G. Unveiling the Impact of User-Agent Reduction and Client Hints: A Measurement Study. In *Proceedings of the 22nd Workshop on Privacy in the Electronic Society* (2023), pp. 91–106

Contributions

- Principal author.

Unveiling the Impact of User-Agent Reduction and Client Hints: A Measurement Study

Asuman Senol[1] and Gunes Acar[2]

[1] COSIC, KU Leuven
[2] Radboud University

Abstract. The user-agent string contains the details of a user's device, browser and platform. Prior work on browser fingerprinting showed that the user-agent string can facilitate covert fingerprinting and tracking of users. In order to address these privacy concerns, browsers including Chrome recently *reduced* the user-agent string to make it less identifying. Simultaneously, Chrome introduced several highly identifying (or high-entropy) user-agent *client hints* (UA-CH) to allow access to browser properties that are redacted from the user-agent string. In this empirical study, we attempt to characterize the effects of these major changes through a large-scale web measurement on the top 100K websites. Using an instrumented crawler, we quantify access to high-entropy browser features through UA-CH HTTP headers and the JavaScript API. We measure access delegation to third parties and investigate whether the new client hints are already used by tracking, advertising and browser fingerprinting scripts. Our results show that high-entropy UA-CHs are accessed by one or more scripts on 59.2% of the successfully visited sites and 93.8% of these calls were made by tracking and advertising-related scripts—primarily by those owned by Google. Overall, we find that scripts from ∼9K distinct registrable (eTLD+1) third-party domains take advantage of their unfettered access and retrieve the high-entropy UA-CHs. We find that on 91.6% of the sites where high-entropy client hints are accessed via the JavaScript API, the high-entropy hints are exfiltrated by a tracker script to a remote server. Turning to high-entropy UA-CHs sent in the HTTP headers—which require *opt-in* or delegation—we found very limited use. Only 1.3% of the websites use the `Accept-CH` header to receive high-entropy UA-CHs; and an even smaller fraction of websites (0.4%) delegate high-entropy hints to third-party domains. Overall, our findings indicate that user-agent reduction efforts were effective in minimizing the passive collection of identifying browser features, but third-party

tracking and advertising scripts continue to enjoy their unfettered access.

Keywords – user-agent string; client hints; web privacy; fingerprinting; online tracking

1. Introduction



Figure 1: User-agent reduction in desktop and mobile Chrome browsers.

The user-agent (UA) HTTP header was introduced in 1992 for "statistical purposes and the tracing of protocol violations" [64, 36]. The standard defined the UA header as a string that contains one or more tokens of software name and versions such as `LII-Cello/1.0 libwww/2.5`. The modern UA string typically contains far more extensive information about a user's browser, device and platform, and it is used for various purposes including analytics, debugging, content adaptation, or detecting incompatible, outdated or vulnerable browsers [84]. In addition to these legitimate use cases, the UA string may also enable stealthy cross-site tracking through browser fingerprinting, if combined with other device and browser information such as screen dimensions, installed fonts, or graphics capabilities [23]. A flavor of browser fingerprinting called *passive fingerprinting* only makes use of features that can be extracted from the received network packets, without running any client-side code [19, 52]. The information available to a passive fingerprinter include the user-agent string, IP address, `Accept` headers, clock skew [47] and protocol quirks [102]. Using passive fingerprinting, a third-party tracker may potentially link a user's web visits without running any client-side code, thwarting any detection efforts by the user or by web privacy measurement studies.

Empirical research on detecting passive fingerprinting on the web is almost non-existent, likely due to the elusive character of passive fingerprinting. Marketing materials of advertising companies, however, hint at the active use of user-agent data for tracking and ad targeting. For instance, TikTok recommends that websites send both the IP address and UA string of visitors through their server-to-server TikTok Events API for better ad targeting [86]. Mobile analytics and marketing company AppsFlyer's bulletin on Chrome's UA reduction mentions that user-agent data is used for "attribution via probabilistic modeling" [5].[3] AppsFlyer recommends [5] their partner ad networks to collect and share with

[3]Probabilistic attribution refers to attributing ad clicks, impressions or app installs without relying on deterministic identifiers such as AdID (Android), IDFA (iOS), or cookies.

them OS (platform) version and device model user-agent client hints—two client hints that we found to be accessed most in our measurements (Section 4.2). Further, with the announced third-party cookie phase-out by Chrome [11] and anti-tracking measures turned on by default in Safari and Firefox, fingerprinting will likely be more crucial for cross-site tracking of users [44].

```
Mozilla/5.0 (Linux; U; Android 7.0; es-us; SM-A510M Build/NRD90M)
     AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/61.0.3163.128
     Mobile Safari/537.36 XiaoMi/Mint Browser/3.7.2
```

Listing 1: An example of a user-agent string (source: https://user-agents.net/s/rx5HpTAQIN).

Beyond its uses as a passive fingerprinting vector, the information contained in the user-agent string itself can be used for ad targeting and suppression. For instance, Google's documentation for real-time bidding (RTB) mentions that the user-agent header "has historically been included in bid requests to provide useful targeting data such as the initiating device's browser and platform" [33]. Using the UA string, advertisers may choose to show luxury product ads to high-end smartphone/laptop owners, or vice-versa: they could exclude users with outdated and budget devices from their ad campaigns [56]. The user agent string may also contain the primary language or locale of the device, which may reveal a user's country-of-origin and their immigration background [30, 93]. Consider the user-agent string given in Listing 1, which belongs to a budget smartphone brand (Xiaomi) with a (now) outdated Android version, having `es-US` locale (Spanish/US). The UA string may give advertisers enough information to correctly or incorrectly add this user to an imaginary "Latinx on a limited budget" marketing segment [15], or exclude them from certain job ads by their race, as Facebook (now Meta) allowed advertisers to do in the past [45]. Furthermore, knowing a user's country-of-origin may allow targeting ads appealing to (or exploiting) their cultural sensibilities [1, 25, 37].

In January 2020, Google announced plans to reduce the information conveyed in UA strings as part of their Privacy Sandbox project [99, 69, 98]. The stated objective of the effort was to minimize the identifying information in the UA string to prevent passive fingerprinting [99]. The plans also included the reduction of information in the associated `navigator` properties: `userAgent`, `appVersion`, and `platform` [98, 31]. The implementation of these changes occurred incrementally, starting with UA reduction for desktop browsers in May 2021, and ending with the reduction of Android Mobile (and Tablet) UA string and related JavaScript APIs in May 2023 [31]. Simultaneously, Google has introduced UA *client hints* (UA-CH) in Chrome, to expose the information reduced from the UA string in a structured manner. These new client hints can be accessed by first- or third-party scripts without any restrictions using a new JavaScript interface called `navigator.userAgentData`.

In particular, `navigator.userAgentData.getHighEntropyValues` method returns high-entropy browser characteristics, including the unreduced UA string, device model and platform version. Adding these new client hints made it possible for web applications to easily access detailed UA information without parsing the complex UA string. For instance, Kline et al. report that the internal ruleset used by comScore to parse UA strings is "several thousand lines long" [46]. Thanks to the structured UA client hints web applications that rely on detailed browser information could retain their functionality. Making it easier to access high-entropy browser features may appear contradictory to the spirit of the UA reduction effort, but the stated objective of the UA reduction effort is to make *covert tracking* (via passive fingerprinting) more difficult, and active fingerprinting more transparent—*not more difficult* [73, 98]. Chrome achieves these dual objectives by reducing the information shared by default in the UA HTTP header, and requiring explicit JavaScript calls or HTTP/HTML-based *opt-in* to access high-entropy browser features. Making access to high-entropy browser properties auditable and actionable is motivated by Google's Privacy Budget proposal, which aimed to prevent active browser fingerprinting by limiting the entropy available to scripts [68, 98, 59].[4]

Table 1: User-agent client hint headers along with their descriptions, sample values, and entropy levels.

Client Hint Header	Description	Example Value	Entropy
`Sec-CH-UA`	Browser name and major version	"Chromium";v="113", "Not-A.Brand";v="24"	Low
`Sec-CH-UA-Mobile`	Boolean value indicating a mobile device	?0"	Low
`Sec-CH-UA-Platform`	Operating system name	"Linux"	Low
`Sec-CH-UA-Full-Version` (Deprecated)	Unredacted UA version	"113.0.5672.63"	High
`Sec-CH-UA-Full-Version-List`	List of unredacted UA versions	"Chromium";v="113.0.5672.63", "Not-A.Brand";v="24.0.0.0"	High
`Sec-CH-UA-Platform-Version`	Operating system version	"NT 6.0", "5.15.0", or "17G"	High
`Sec-CH-UA-Arch`	Platform architecture	"ARM", or "x86"	High
`Sec-CH-UA-Model`	Device model	"Pixel 2 XL"	High
`Sec-CH-UA-Bitness`	CPU architecture bitness	"32" or "64"	High
`Sec-CH-UA-WoW64`	Whether the UA is a 32-bit binary running on 64-bit OS	?0 or ?1	High

In this empirical study, we investigate the effects of Google Chrome's UA reduction efforts on the exposure of potentially identifying browser features. The key contributions of our research are as follows:[5]

- We present a web measurement study of user-agent client hints (UA-CHs) on the home pages of the top 100,000 sites. Using an instrumented crawler, we capture and analyze JavaScript calls, HTTP headers, and

[4]We note that the public GitHub repository of Privacy Budget proposal is last updated in late 2020 [68].

[5]The code and data from our study are available at `https://github.com/ua-reduction`.

HTML `<meta>` elements, which can be used to access, opt-in or delegate high- and low-entropy user-agent client hints.

- Monitoring the newly introduced JavaScript interface, we find that one or more third-party scripts access high-entropy client hints on 52,392 (58.4% of the successfully visited) websites. An overwhelming majority (93.9%) of these scripts are classified as advertising related. From those sites where we observed third-party access, we observe third-parties exfiltrate high-entropy values to their endpoints on 47,691 (91.0%) sites.
- Using JavaScript instrumentation, we detect fingerprinting attempts by third-party scripts and find that only a minority of the fingerprinting scripts already take advantage of newly introduced high-entropy client hints.
- We examine the usage of high-entropy CH headers sent in HTTP requests and find that their use is scarce. This indicates that the UA reduction efforts succeeded in minimizing the exposure of high-entropy browser features to HTTP endpoints that may use them for passive fingerprinting.

2. Background and Related Work

In the more than 30 years since its introduction, the user-agent string has grown into a long, detailed and somewhat confusing string that contains many ossified parts. Perhaps only a minority of tech professionals know what `KHTML` is, what is special about `AppleWebKit` version 537.36 [82], and why Chrome and Safari mention `Mozilla/5.0` in their UA string [91, 64]—despite having seen these strings potentially thousands of times. Parallel to this ossification, many new properties of the browser, operating system (OS) and physical device were added to the UA string, so web servers can send compatible and localized web content. In addition to legitimate use cases such as delivering optimized versions of images or videos for devices with different form factors [80], the UA string can also be used to obtain a potentially unique *browser fingerprint* by combining it with other information [40, 9, 57, 23, 48].

The UA string influences the uniqueness of a user's fingerprint due to the discriminating information it contains. In a seminal study by Eckersley [24], the UA string ranked as the third-most distinguishing browser feature after fonts and plugins. Specifically, by collecting the fingerprints of 470K volunteered participants, Eckersley found that the UA string contains 10.0 bits of Shannon entropy—which means only one in 1024 (2^{10}) other browsers is expected to share the same UA string for a given browser. Following a similar approach,

a 2016 study by Laperdrix et al. [48] showed that UA string is the most identifying feature for mobile browsers and third for identifying features for desktop browsers. Surprisingly, one of every four smartphones in their dataset had a unique UA string—primarily due to the inclusion of details such as the device model, phone carrier, and firmware version. We note that while we focus on browsers as the primary software to access the web, mobile apps, smart devices and software libraries can also be classified as user agents, and have user-agent strings.

2.1 What Changed and How?

In January 2020 Chrome developers announced their "[i]ntent to deprecate and freeze the UA string" [98]. While the initially announced timeline was not followed due to COVID, Chrome reduced or froze part of the UA string in several phases [31]. The milestones included replacing the minor version number with zeros and using a fixed Android version. By February 2023, the UA string for Android mobile and tablet devices was fully reduced [31]. Google's UA Reduction effort had three main components:

1. Reducing the granularity of the UA string and freezing parts of it (Figure 6). The minor version number is replaced with zeros in Chrome 101 (June 2022). As an example, Chrome/101.3.2.1 would become Chrome/101.0.0.0 [2] (Figure 1). Further, CPU and platform-related details are simplified for the desktop browsers in Chrome 107 (February 2023), and the Android version number is replaced with the fixed string "10" (May 2023) [31].

2. The second component involves the implementation of user-agent client hint (UA-CH) HTTP headers—a structured alternative to the UA string. UA-CH headers are enabled by default in Chrome 89, released in March 2021 [73].

3. The third component introduces a new JavaScript interface called `NavigatorUAData`, which contains properties and methods for scripted access to UA-CHs (introduced in Chrome 90, released in April 2021) [54].

2.2 User-Agent Client Hints

User-agent client hints (UA-CHs) offer a structured alternative to retrieve the information redacted from the UA string. This client hints to complement the UA reduction efforts to maintain compatibility. A client hint is introduced for all UA string components, a complete list of which is given in Table 1. UA-CHs

are categorized as low- and high-entropy, based on how identifying they are. For instance, *full-version* hint, which contains a detailed build version is categorized as high entropy; while *mobileness* hint indicating whether a device is mobile or not is categorized as low entropy. The entropy category of a UA-CH determines whether the corresponding header is automatically appended to every request or not: low-entropy hints are sent in every request, while high-entropy hints require explicit opt-in or delegation. To the best of our knowledge, the division between the high- and low-entropy client hints is not based on publicly available research. We observe that rare values for low-entropy hints such as *platform=Linux* or *Fuchsia*, can be identified if combined with other browser features.

The list of user-agent client hints is not fixed. For instance, Chrome recently added a client hint representing the form factor of the user's device and the corresponding `Sec-CH-UA-Form-Factor` header [55]. This information was historically conveyed as a `<deviceCompat>` token in the user-agent string (e.g., "Tablet", "VR", etc.) and was added during our study. Other proposed user-agent client hints include a hint that indicates whether the browser needs an update [28] and whether the browser is a self-declared bot [29]. Note that UA-CHs are just one type of client hints; others include *Device Client Hints*, *Network Client Hints* and *User Preference Media Features Client Hints* [53].

During our investigation, we spotted a few mismatches between the latest UA-CH proposal[6] and its implementation in Chrome. For instance, since August 2021, the UA-CH standard requires an empty string to be returned in the *platform version* for Linux devices [90]. However, as of July 2023, Chrome still returns the Linux kernel version (e.g., 5.15.0) in the platform version client hint on Linux [83].

All the information redacted from the user-agent string can be obtained through high-entropy client hints. These hints can be accessed via HTTP headers, or by using the JavaScript API. Notably, the JavaScript API does not require any permission or opt-in from the website—even for the scripts running in third-party iframes. On the other hand, to receive high entropy UA-CHs in the HTTP headers even a first-party needs to opt-in. Detailed explanations of both methods are provided in the subsequent sections.

Access to User-Agent Client Hints via HTTP

By default, Chrome appends three low-entropy client hints, i.e, *platform name*, *major browser version*, and *mobileness* in each HTTP request, if the connection is secure (i.e., HTTPS). On the other hand, high-entropy client hints require

[6] User-Agent Client Hints proposal is a Draft Community Group Report and it is still in the pre-standardization stage [95].

[7] `https://developer.chrome.com/docs/privacy-sandbox/user-agent/`

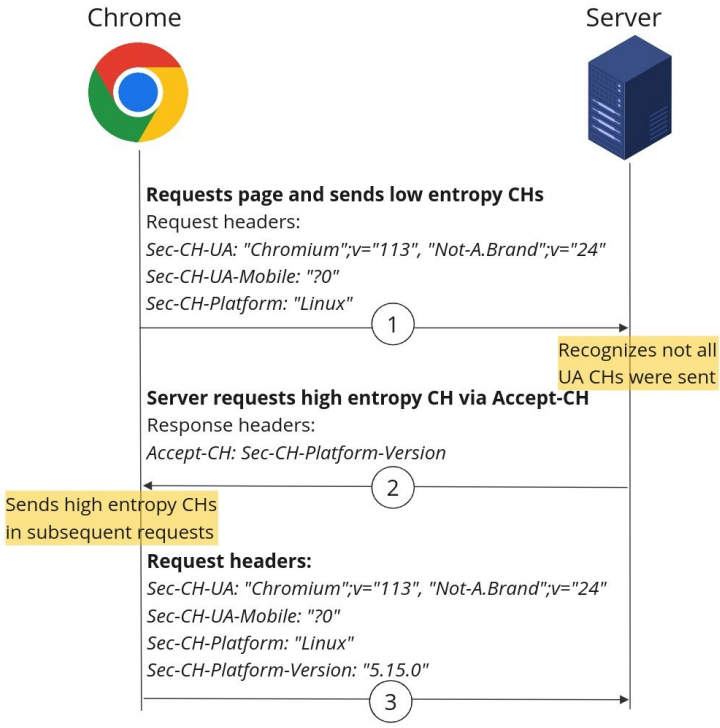


Figure 2: The `Accept-CH` response header mechanism used by servers to request high-entropy user-agent client hints from clients. After receiving an `Accept-CH` header during a secure top-level navigation request, the browser appends the requested client hint headers in subsequent requests. The image is adapted from Google's documentation.[7]

explicit opt-in or delegation to be sent to first, or third parties, respectively. Top-level origins may request high-entropy UA-CH headers by sending an `Accept-CH` response header as shown in Step 2 in Figure 2. `Accept-CH` can contain a list of CH header names such as `Sec-CH-UA-Platform-Version`. The browser then sends these additional CH headers in all subsequent requests to the first-party domains (Step 3 in Figure 2). Additionally, client hints can be requested at the connection time using HTTP/2 or HTTP/3 `Accept_CH` frames, or using the TLS Application-Layer Protocol Settings (ALPS) extension mechanism [39, 38, 3]. Requesting the client hints at the connection layer instructs the browser to append the UA-CHs to the first HTTP request, which saves a round trip at the application layer.

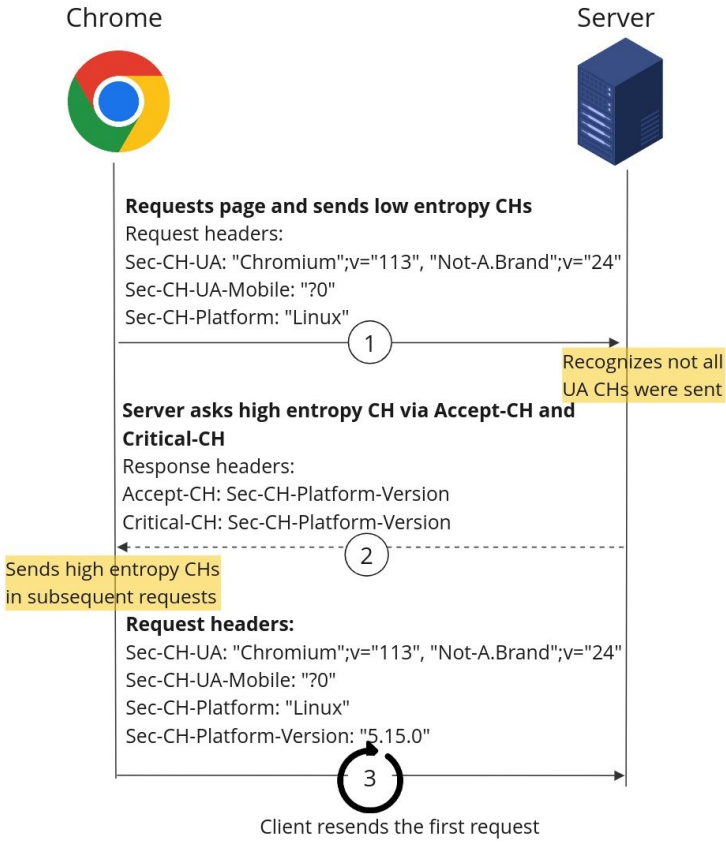


Figure 3: Requesting high-entropy CHs initially using `Critical-CH` header: When the server asks for a `Critical-CH`, the client will retry the initial webpage request with it. The image is adapted from Google's documentation.[8]

Servers can also use the `Critical-CH` response header when certain high-entropy UA hints are required on the first navigational request. Clients that receive a response with a `Critical-CH` header must check if high-entropy hints are sent in the original request. If not, the client will resend the original request with the requested hints (Step 3 in Figure 3).

Access to User-Agent Client Hints via the JavaScript API In the past, the UA information was exposed to scripts through the `userAgent`, `appVersion`, and `platform` properties of the `navigator` object. However, as part of UA reduc-

[8] `https://developer.chrome.com/docs/privacy-sandbox/user-agent/`

tion efforts, the information accessible through those properties was also reduced. In order to provide an alternative mechanism for scripts to access high-entropy browser features, Chrome introduced a JavaScript API for user-agent client hints [88]. This API includes an interface named `NavigatorUAData` (Appendix 6), which provides low-entropy UA values, and an asynchronous method called `navigator.userAgentData.getHighEntropyValues`, offering high-entropy UA values [55]. We have provided an example of a `navigator.userAgentData` call and its corresponding output value in Appendix 6. The `getHighEntropyValues` method allows client-side scripts to selectively retrieve high-entropy UA client hints such as `architecture`, `bitness`, `platformVersion`, `uaFullVersion` and `fullVersionList`.[9] This method offers a more explicit and controlled approach to retrieving browser features, allowing for better monitoring and transparency. Like UA-CH headers, `getHighEntropyValues` is restricted to HTTPS pages. An example call and output in Google Chrome are shown in the Listing 2.

```
navigator
  .userAgentData.getHighEntropyValues(
      ["brands", "mobile", "bitness", "platform",
      "platformVersion", "architecture",  "model",
      "uaFullVersion", "fullVersionList", "wow64"])
  .then((ua) => { console.log(ua) });
// OUTPUT:
{
    "architecture": "x86"
    "bitness": "64"
    "brands":[
        {
            "brand":" Not A;Brand",
            "version":"24"
        },
        {
            "brand":"Chromium",
            "version":"113"
        },
        {
            "brand":"Google Chrome",
            "version":"113"
        }
    ],
    "fullVersionList":[
        {
            "brand":" Not A;Brand",
            "version":"24.0.0.0"
        },
        {
            "brand":"Chromium",
            "version":"113.0.5672.63"
        },
        {
            "brand":"Google Chrome",
            "version":"113.0.5672.63"
        }
    ],
    "mobile": false,
```

[9]Note that the camelCase UA client hint names in JavaScript correspond to UA-CH headers listed in Table 1; e.g., `platformVersion` => `Sec-CH-UA-Platform-Version`.

```
    "model": "",
    "platform": "Linux",
    "platformVersion": "5.15.0",
    "uaFullVersion": "113.0.5672.63",
    "wow64": false
}
```

Listing 2: Sample output of the `getHighEntropyValues` in Google Chrome. returning requested user-agent client hints including the high-entropy ones.

Remarkably, any first- or third-party script can call the `getHighEntropyValues` method to access high-entropy UA-CHs without permission, opt-in, or delegation from the website. This is in stark contrast to how sending high-entropy UA-CH HTTP headers requires active opt-in even for the website (or the first-party domain) itself.

GREASE for User-Agent Brands To prevent reliance on specific UA strings in specific positions, the Generate Random Extensions and Sustain Extensibility (GREASE) technique [55] is adopted from TLS [6] to UA client hints. In the UA-CH context, GREASE involves introducing intentionally incorrect or random entries in the UA brands list to increase compatibility and prevent sites from blocking unknown browsers. Specifically, browsers are required to include multiple values in the brands list, with one being an arbitrary value. The order of values must change over time to avoid dependence on specific positions, and a random set of separators should be used [89]. Unfortunately, the recent bug reports by Vivaldi —a Chromium-based browser— show that GREASE does not ensure equal treatment [101, 8]. According to reports, websites already started parsing client hints with misplaced assumptions, and deny access to browsers with small market shares due to unexpected brand CH values [101].

2.3 Delegating Hints to Third-Parties

By default, UA-CHs are not sent for third-party subresource requests, and third parties cannot request high-entropy UA-CHs themselves. Instead, the first-party server must send a `Permissions Policy` header that lists the origins that may receive the UA-CHs.[10] Examples of implementing this policy can be found in Figure 4.

Let's take an example where example.com embeds a resource from sample-cdn.com which needs the full version of the browser. The `Sec-CH-UA-Full-Version` hint can be requested by https://example.com, but it must be explicitly delegated to https://www.sample-cdn.com using the `Permissions-Policy`

[10]The `Permissions Policy` (previously, `Feature Policy`) header allows enabling and disabling browser features such as microphone and camera for all or certain origins.

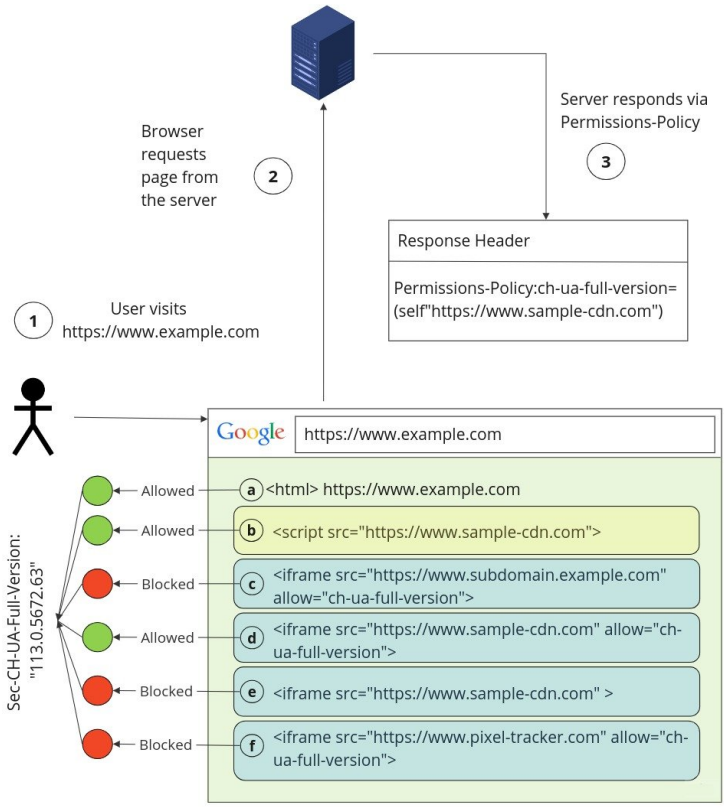


Figure 4: Delegating client hints to Third-Party Origins by `Permissions-Policy`. The image is based on a figure in Google's documentation.[11]

response header as shown in Step 3 in Figure 4. Then the request to subresources on sample-cdn.com (script source ⓑ in Figure 4) includes the delegated hint which is `Sec-CH-UA-Full-Version`: "113.0.5672.63".

Additionally, multiple hints can be defined for multiple origins. For instance, example.com delegates multiple hints to multiple origins with this response header as shown in Listing 3.

```
Accept-CH: Sec-CH-UA-Full-Version
Permissions-Policy: ch-ua-full-version=(self "https://www.sample-cdn.com"),
    ch-ua-model=(self "https://www.sample-cdn.com" "https://analytics.com")
```

Listing 3: Client hint delegation to multiple domains.

[11] `https://developer.chrome.com/docs/privacy-sandbox/user-agent/`

To enable the delegation of CHs to cross-origin iframes, it is essential to specify the CHs in the iframe's `allow` attribute, along with the `Permissions Policy` response header. Figure 4, presented illustrates this process in ⓓ. When an iframe with the source https://sample-cdn.com includes the `allow` attribute, it can receive the `Sec-CH-UA-Full-Version` CH header. However, as shown in ⓔ of Figure 4, the absence of the `allow` attribute results in the exclusion of this high-entropy CH.

Sample ⓒ in Figure 4 illustrates a same-site, cross-origin frame (subdomain.example.com). Despite having an iframe `allow` attribute for UA-CHs, the high-entropy hint will not be sent when loading this iframe, because it is not included in the origin list.

2.4 User-Agent Client Hint Opt-in and Delegation via HTML

Publishers who cannot modify their website's `Permissions Policy` HTTP header can use HTML to opt-in to high-entropy client hints or delegate them to third parties. The HTML `<meta>` element can be used to opt-in to high-entropy CHs for first-party requests. In this case, the `http-equiv="Accept-CH"` attribute, along with the `content` attribute can be used. The `content` attribute holds the list of hints to be delegated, along with the desired recipient origins. Similarly, the `http-equiv` attribute of the `<meta>` element enables explicit delegation of high-entropy CHs to third-party domains. To achieve this, the `http-equiv="delegate-ch"` attribute is used, along with the `content` attribute, which designates the hints to be delegated and the corresponding recipient origins.

2.5 Statuses and Positions of Other Browser Vendors

In this section, we summarize the status and viewpoints of Apple and Mozilla with respect to UA reduction and UA client hints. Due to space limitations we only provide a very high-level summary of each vendor's position and we do not claim that the following is an exhaustive description:

Mozilla Independent of Google's efforts, Mozilla had already frozen the rendering engine version (`rv`) in Firefox 110 to avoid being detected as IE11 [76]. Mozilla also reduced the information exposed in the UA string over time [63, 60]. With respect to introducing user-agent client hints, Mozilla initially held a negative stance. Mozilla initially labeled UA-CHs as *harmful* in their web standard positions [77], citing concerns such as additional fingerprinting risks, but they subsequently updated their position to *neutral* [61]. While there are

bugs to track the implementation of UA-CHs [58, 62], no work has been done toward that implementation as of July 2023.

In their critique of UA-CHs, Mozilla highlights the bandwidth overhead of sending multiple low-entropy `Sec-CH` headers in each request, along with additional fields requested by sites using the `Accept-CH` mechanism [85]. Note that while HTTP/2 and HTTP/3 support header compression, even the compressed fields may consume a non-trivial amount of bandwidth. We leave the bandwidth overhead measurement of UA-CHs to future work.

Apple As early as 2017 Apple attempted to freeze the UA string to reduce compatibility issues and defend against browser fingerprinting [96, 74, 17]. Seemingly due to compatibility concerns, they later unfroze the major OS version [75]. Apple initially held a negative stance against UA-CHs and filed several issues on the public GitHub repository, which are addressed except for one as of July 2023 [81, 97]. The only remaining issue filed by Apple concerns the `getHighEntropyValues` method, which returns all requested high-entropy hints, including those that have not been explicitly opted in via the `Accept-CH` header [78]. This means, regardless of its origin and security context, a script can access all high-entropy UA client hints, even if the first-party domain did not send an `Accept-CH` header to opt-in to high-entropy client hints. Citing the W3C specification, Apple engineers suggested that prior opt-in must be required for accessing high-entropy hints from scripts [78]. Currently, the first party can *opt-out* from exposing high-entropy client hints to scripts from any origin by sending a restrictive `Permissions-Policy` header.

Note that Apple requires all browsers running on iOS to use its own WebKit rendering engine [14]. Thus, Apple's position towards client hints also determines the availability of UA-CHs on iOS versions of Chrome, Firefox and other browsers.

2.6 Related Work

The research community has expressed considerable interest in the UA string and its implications for web privacy and security. In this section, we discuss the most pertinent research in this domain.

In his pioneering study, Eckersley [23] demonstrated that it is feasible for websites to uniquely identify users at scale through their browser fingerprints. Collecting browser fingerprints of over 470K volunteers, Eckersley showed that over 94% of the users having Flash or Java have a unique fingerprint. Following Eckersley's lead, several studies introduced novel ways to fingerprint browsers such as Canvas, Canvas Font and Audio Context API fingerprinting, as well as

WebRTC local IP discovery [57, 27]. In our study, we use heuristics proposed by Englehardt and Narayanan [27] to detect these four types of fingerprinting attempts. Vastel et al. [94] devised a technique to link evolving browser fingerprints. Analyzing 98,598 browser fingerprints from 1,905 distinct browser instances collected over a two-year period, they found that fingerprints regularly change due to software updates, but their algorithm can link evolving browser fingerprints with reasonable accuracy for months.

A type of fingerprinting called *passive fingerprinting* involves extracting unique device fingerprints without running any client-side JavaScript code. Passive fingerprinting relies on information readily available to a web server such as IP address and HTTP headers, including `User-Agent`, `Accept-Language`, and `Accept` headers [52, 19]. These fingerprints can then be used to track and identify devices across different browsing sessions. The main privacy concern with passive fingerprinting is that it enables covert tracking since it is undetectable to any client-side analysis, including web measurement studies that use heavily-instrumented browsers [52]. An influential study by Kohno et al. presented a method to remotely identify a specific physical device using its *clock skews* [47]. Using timestamps sent in network packets, their method could identify devices that were geographically distant, connected via different access technologies, or behind firewalls/NATs. Analyzing web server logs of Hotmail and Bing services, Yen et al. [100] found that 60-70% of the distinct user-agent strings in their dataset were unique. Kline et al. analyzed a corpus of over one billion user-agent strings collected by comScore, a media traffic measurement and analytics company [46]. They found that less popular (*low-volume*) UA strings generate the overwhelming majority of traffic [46].

Variations in how different devices and software implement specific network protocols can lead to distinguishable patterns at the network layer. Zalewski implemented `p0f`, which passively identifies operating systems and software using differences in their protocol implementations [102]. `p0f` uses a variety of network and application-level features extracted from TCP, IP and HTTP headers including the layout and ordering of TCP options; syntax and order of HTTP headers; TCP timestamp increments; and various other implementation quirks. A remarkable feature of `p0f` is how little data it needs for fingerprinting: a single standard SYN packet could be enough to distinguish many systems. While `p0f` mainly uses protocol quirks for fingerprinting, it uses the user-agent string to detect mismatches between the declared and actually used software and OS, which may indicate *dishonest software* or manipulation by on-path proxies. Following a similar approach, several tools and libraries were developed for TLS fingerprinting [87, 71, 50]. Using differences in TLS versions, ciphersuites, supported extensions and compression options, these projects aimed to identify the TLS software or libraries for objectives such as detecting unpatched browsers,

or malicious software that impersonates real browsers. Anderson and McGrew's research showed that precise OS version detection is possible by merging TLS features with TCP/IP and HTTP protocol data [4].

Another line of research and applications used UA strings to detect malware and compromised hosts. Grill et al. detected infected hosts by using the discrepancy between a user's actual UA string, and the UA string used by potential malware [34]. UA strings can also be used for detecting malicious and benign bots. The npm package "isbot", for instance, claims to distinguish between good bots such as web crawlers and bad bots that launch DDoS attacks and spam campaigns [41].

We contribute to the literature by performing a large-scale web measurement study focusing on the implications of user-agent reduction and the simultaneous introduction of potentially identifying user-agent client hints. Our study provides empirical data that can be used to evaluate the effectiveness of these major changes and their impact on web privacy.

3. Methodology

In this section, we describe the methods and tools developed to measure the usage of UA-CH HTTP headers and the JavaScript API.

3.1 Extending Tracker Radar Collector

In order to develop our crawler, we extended Tracker Radar Collector (TRC), a Puppeteer-based web crawler that records certain JavaScript API accesses, HTTP requests & responses, cookies, and other data related to web measurements. Since TRC only saves a pre-defined list of HTTP headers, we extended its allow-list by adding the ten UA-CH HTTP headers listed in Table 1, and other HTTP headers used to opt-in and delegate high-entropy CHs (e.g., `Accept-CH` and `Critical-CH`). In addition, we parsed the *meta* and `iframe` elements' attributes, as opt-in and delegation can also be accomplished through these means (Section 2.4). In order to monitor access to high-entropy UA values by scripts, we intercept calls to `navigator.userAgentData.getHighEntropyValues` and save the arguments and the call stack. We use the function arguments to identify the requested high-entropy client hints, and capturing the JavaScript call stack allows us to identify the caller script's URL. Additionally, we instrument fingerprinting-related method calls and property accesses using a separate collector (Section 3.2).

For both `getHighEntropyValues` and fingerprinting detection, we override the relevant object's *getters* to intercept the function calls. While TRC has functionality for intercepting JavaScript API calls, we added a separate collector, because the existing collector misses the initial function calls due to a known TRC issue [20]. Moreover, we extended TRC's network instrumentation to save all WebSocket traffic and HTTP POST payloads to better examine whether any high-entropy values are being exfiltrated. This effort complemented the existing network instrumentation of TRC, which already intercepted GET requests.

Only visiting the webpage for a few seconds may not accurately represent an actual user's visit. To better simulate a real user, we scrolled to the bottom of the page and then scrolled up, waiting for 5 seconds before gathering the necessary data. During the initial phases of the study, Chrome in headless mode did not support client hints. While we initially planned to run our crawler in headful mode within a virtual display (e.g. via `Xvfb`), client hint support was added to headless Chrome in February 2023 [12].

We override the crawler's UA string and client hints using Puppeteer's *page.setUserAgent* [70] method. This method allows passing a `userAgent` string and structured UA metadata (`userAgentMetadata`) which we use to specify all possible UA-CHs. The UA values we use to override were taken from a real Chrome browser running on Linux, and are given in Appendix 6.

3.2 Detecting Fingerprinting Attempts

To determine whether existing fingerprinting scripts are leveraging the newly introduced high-entropy client hints, our first step is to identify these fingerprinting scripts. However, this can be a challenging task, as discerning the true intent of a script may be difficult. For instance, a script might use the AudioContext API solely for processing and synthesizing audio in web applications, without using the data for fingerprint creation. Previous research has employed either machine learning-based techniques [40] or heuristics [27] to detect such scripts. Due to the scope of our study, we adopted Englehardt and Narayanan's heuristics (see Appendix 6) to detect fingerprinting scripts, focusing exclusively on four sub-types: Canvas, WebRTC, Canvas Font, or AudioContext fingerprinting [27]. Briefly explaining each fingerprinting method, Canvas fingerprinting involves using the HTML5 Canvas API to draw images with slight variations for user tracking [57]. WebRTC local IP discovery exploits the WebRTC API to obtain the local IP of the browser [27]. Canvas Font fingerprinting utilizes the Canvas API to render text in various fonts and styles, enabling the analysis of rendering differences to create distinct fingerprints [27]. Similarly, AudioContext API fingerprinting use signal processing features to

extract a fingerprinting of the audio stack of the device [27]. To intercept function calls, return values and function arguments, we overrode the getter and setter methods of those Web APIs. The script responsible for overriding was injected into every page including their subframes immediately after the page was created, before the page's scripts run.

3.3 Interaction with Consent Dialogs

Since the GDPR came into effect in 2018, numerous websites started to display consent dialogs when accessed from the European Union (EU)—or even from the US, to some limited extent [72]. To fully measure the access to the high-entropy UA-CHs, we opted to accept all personal processing (or cookies). In order to handle consent dialogs accurately and automatically, we used code extracted from Priv-Accept [51, 42], a specialized crawler designed to accept consent dialogs on websites. Since Priv-Accept was only available in Python, we ported its consent interaction code to JavaScript and integrated it into our crawler. Essentially, our Priv-Accept (NodeJS) script searched for specific HTML elements such as `a`, `button`, `div`, `span`, `form`, `p`, containing keywords such as "Accept". Upon detecting such an *accept element*, Priv-Accept automatically clicked it to accept all personal data processing and cookies.

3.4 Detection of High-Entropy Value Exfiltrations

Do the scripts that access high-entropy UA values send it to their servers or share it with other third parties? The high-entropy UA hints retrieved by the scripts may potentially be used for feature detection or other client-side optimization, which could alleviate privacy concerns since the high-entropy values may not leave the browser. In other cases, scripts that retrieve the high-entropy UA values via JavaScript calls may send them to several third-party domains, potentially for tracking and advertising purposes. Detecting these exfiltrations relies on identifying the presence of high-entropy UA values in the HTTP request payloads (excluding the `Sec-CH-*` headers). A potential challenge for accurately detecting such exfiltrations is to identify the sent high-entropy hints in encoded, hashed, or obfuscated payloads. Previous studies in web privacy measurement have taken various approaches to tackle this challenge. In order to detect Personally Identifiable Information (PII) exfiltrations from contact forms, Starov et al. [79] conducted a comparative analysis of HTTP parameters observed in multiple crawls. However, their method requires manual analysis and multiple crawls, where they provide different identifiers. Instead, we follow Englehardt et al.'s approach [26], which involves searching for multi-

layered encodings and hashes (e.g., Base64, SHA-256) of search strings in the HTTP request URL, headers and POST data.

Naively searching for all high-entropy UA values in the HTTP traffic may lead to false positives. Consider the high-entropy hint `Sec-CH-UA-Bitness`, which may take the values of 32 and 64. These two-digit values may appear in the HTTP traffic due to unrelated reasons, for instance as part of random strings. In order to prevent such false positives, we only searched for two high-entropy client hints: the platform version (`5.15.0`) and the full UA version (`113.0.5672.63`). We begin by generating a precomputed pool containing all possible sets of tokens through iterative encodings and hashes of the platform version and full version. We then search for exfiltrations in the referrer header, URL, and POST bodies of the requests, using potential separator characters (e.g., '=') to split the content. We exhaustively apply all possible decodings and check if the decoded result matches any entry in the precomputed pool. This process is repeated until we reach a maximum of three layers of encodings or decodings. A detailed list of the encoding and hash algorithms used can be found in Appendix 6.

3.5 Identifying Tracking-Related Requests

To identify tracking-related requests, we employed the uBlock Origin Core npm package [35], which replicates the blocking behavior of the widely used uBlock Origin tracking protection extension [92]. By using the default filter lists employed by uBlock Origin, including EasyList [21] and EasyPrivacy [22], we determined the blocked status of each request. To achieve this, we provided uBlock Origin Core with the resource type of the request (e.g., image or script), as well as the page and request URL. When detecting whether a script is tracking-related or not, we use the script URL extracted from the JavaScript call stacks (§3.1).

3.6 The Crawl

In June 2023, we crawled the homepages of the top 100K sites in the April 2023 Chrome User Experience Report list [43], excluding sites with identical fully qualified domain names. 91% (89,763 sites) of websites could be visited successfully while the remaining visits failed due to errors including DNS and timeout errors. Priv-Accept accepted personal data processing on around 21% of the visited sites, as revealed by the crawl logs. The crawl was performed using a cloud-based (DigitalOcean) server located in the United States (US). The crawl took four days, and it was run on a server equipped with 8 vCPU cores and 16GB of RAM. We chose a server located in the US, mainly to minimize

the consent dialogs our crawler needs to deal with. While we use Priv-accept to detect and interact with consent dialogs (Section 3.3), certain consent dialogs may be missed by our crawler, leading to a bias in the measurement. We leave it to future work to perform a comparative study based on multiple vantage points.

4. Measurement Results

Recall that user-agent client hints can be gathered using JavaScript calls and user-agent client hints HTTP headers. Below, we present separate measurements for each access method to conduct a comprehensive analysis.

4.1 `getHighEntropyValues` Calls and Exfiltrations

We observed a call to `getHighEntropyValues` method by one or more third-party scripts on a total of 52,392 unique sites, which amounts to 58.4% of the successfully visited sites. Table 2 gives an overview of the presence of these JavaScript API calls across the analyzed websites. We found that almost all calls (98.6%) are due to third-party and tracking-related scripts. According to Chrome usage metrics, `getHighEntropyValues` is invoked during 53.3% of the page loads [65]. We believe that our measurement (58.4%) is slightly higher because our study focused on the top 100K sites. Recall that we determine the caller script using the JavaScript stack traces observed during a call to `getHighEntropyValues`. In order to detect tracking-related scripts we use the labels returned by the uBlock Origin for the caller script's URL as explained in Section3.5.

Table 2: The number of distinct websites where high-entropy UA-CHs were retrieved by a script and exfiltrated. On over 91% of the websites, scripts exfiltrate high-entropy UA-CHs that they retrieve via the `getHighEntropyValues` method.

	All	**Third party**	**Tracking related**
`getHighEntropyValues` calls	53,148	52,392	51,630
Hi-ent. UA-CH exfiltrations	48,355	47,691	47,285

We analyzed the categories of third-party scripts (obtained via inspecting JavaScript stack traces) that retrieve high-entropy client hints to understand

the purposes of these API calls. For domain categorization, we relied on the Tracker Radar dataset [16], which classifies known (mostly tracker) domains into one or more categories. The results shown in Table 3 reveal that the most common categories are associated with tracking, advertising, marketing, and analytics. Notably, the most prevalent category among them is Ad Motivated Tracking.

Table 3: Most common categories of third-party scripts calling `getHighEntropyValues` method. The right-hand column shows the number of distinct websites where we detected at least one call to `getHighEntropyValues` from the respective category. The categorization is based on DuckDuckGo's Tracker Radar dataset,[12] where a script is assigned one or more categories based on its domain.

Script Category	**Num. Sites**
Ad Motivated Tracking	44,084
Advertising	43,976
Audience Measurement	40,901
Third-Party Analytics Marketing	40,491
Analytics	40,347
Action Pixels	13,224
Embedded Content	4,523
CDN	4,342
Social - Share	2,338
Ad Fraud	1,339

Next, we turn to specific third-party tracker scripts that call `getHighEntropyValues` on most websites. We found that five scripts that retrieve high-entropy hints on most sites come from Google, and most of them are advertising-related. Specifically, googletagmanager.com, googlesyndication.com, and doubleclick.net emerged as the prominent tracker domains among the identified third-party domains as shown in Table 4.

We conducted an analysis to determine which client hints are more frequently retrieved via the `getHighEntropyValues` method by third-party scripts. For this analysis, we used the arguments of `getHighEntropyValues` captured through our JavaScript instrumentation (Section 3.2). Analyzing function call arguments, we found that the most frequently requested UA client hints via the JavaScript API are `model` and `platformVersion` as shown in Table 5. Notably, a third-party mistakenly calls `getHighEntropyValues` with an mistyped argument, `uaFulVersion`, on seven distinct sites. Additionally, on 174

[12] https://github.com/duckduckgo/tracker-radar

Table 4: Top tracker domains that call getHighEntropyValues (left) and exfiltrate (right) high-entropy values.

High Entropy API calls		**High Entropy API exfiltrations**	
Tracker domain	**Num. Sites**	**Tracker domain**	**Num. Sites**
googletagmanager.com	28,929	google-analytics.com	22,517
googlesyndication.com	6,843	google.com	9,325
doubleclick.net	3,633	doubleclick.net	8,853
googletagservices.com	1,414	googlesyndication.com	2,018
googleadservices.com	673	crwdcntrl.net	985
quantserve.com	437	sharethis.com	531
taboola.com	330	gemius.pl	356
clarity.ms	192	taboola.com	315
statcounter.com	161	id5-sync.com	253
wpadmngr.com	152	tynt.com	202

sites the method is called with the argument *None* by scripts served from the ampproject.org domain. In this case, Chrome only returns low-entropy hints.

Table 5: `getHighEntropyValues` arguments sorted by the number of distinct sites they were observed on. The rightmost column indicates the distinct script domains that used the particular hint.

UA Client Hint	**Num. Sites**	**Num. of Script Domains**
model	52,270	970
platformVersion	52,214	1,052
platform	51,529	746
fullVersionList	51,321	792
architecture	51,229	618
bitness	50,874	554
uaFullVersion (deprecated)	50,743	359
wow64	50,132	80
mobile	7,615	283
brands	5,208	462
None	174	1
uaFulVersion	7	1

Upon analyzing the HTTP URL, POST data, and referrer fields (Section 3.4), we found that the `fullVersion` or `platformVersion`, along with their corresponding encodings, were being exfiltrated to a third-party domain across 47,691 distinct websites.[13] Notably, in the majority of cases (88.8%), these values were transmitted without any encoding or obfuscation. However, in certain instances, they were encoded using URL encoding (6.8%) or Base64 encoding (4.1%). The righthand column of Table 4 shows the most common domains that high-entropy UA-CHs are exfiltrated to. Google domains including Google Analytics is followed by Lotame (crwdcntrl.net), which offers its audience data to advertisers, marketers and publishers [49]. Our analysis further revealed that on 91.6% of the sites where the *getHighEntropyValues* was called by a tracker script, the high-entropy values were exfiltrated to tracker domains, as shown in Table 2. Furthermore, despite intercepting WebSocket requests, no exfiltration was observed through this channel.

Our analysis of whether `getHighEntropyValues` is more likely to be called on popular websites yielded a negative result. As Table 10 in Appendix 6 shows, the Tranco rank of a website does not seem to correlate with more or less access to high-entropy UA-CHs via the JavaScript API.

To gain insights into the motivations behind tracker domains exfiltrating high-entropy values, we conducted a comparative analysis of high-entropy value exfiltration across various categories of third-party domains that *receive* these values. In Appendix 6, Table 11 shows that Advertising and Audience Measurement are the top two categories significantly associated with the exfiltration of high-entropy values. Interestingly, these categories align with the categories of the scripts calling the `getHighEntropyValues` method, as shown in Table 3. In fact, the top five categories listed in both tables are exactly the same.

In addition to analyzing scripts involved in fingerprinting, we investigated those accessing high entropy values. Among the 2,190 distinct third-party fingerprinting scripts we identified, a surprisingly low 12.5% make calls to the `getHighEntropyValues` method (Table 12 in Appendix 6).

4.2 The Collection of User-Agent Client Hint HTTP Headers

Recall that UA-CHs are categorized as low and high-entropy depending on how identifying they are (Table 1). Low-entropy CHs are automatically included in every request, while high-entropy CHs require explicit opt-in for first parties or

[13] Note that we excluded all UA-CH headers from this analysis.

delegation for third parties. In the following tables, low and high entropy hints are presented separately due to this distinction.

Table 6: The number of distinct sites where CH headers were observed on requests made to all parties, third parties, and tracker domains.

Ent.	UA-CH Header	All	Third Party	Tracking Related
high	Sec-CH-UA-Platform-Version	886	331	134
	Sec-CH-UA-Model	886	329	132
	Sec-CH-UA-Full-Version-List	696	261	67
	Sec-CH-UA-Arch	667	257	63
	Sec-CH-UA-Full-Version	581	217	25
	Sec-CH-UA-Bitness	491	217	25
	Sec-CH-UA-Wow64	401	210	21
low	Sec-CH-UA	89,141	78,476	67,560
	Sec-CH-UA-Mobile	89,141	78,476	67,560
	Sec-CH-UA-Platform	89,141	78,478	67,560

Table 6 displays the count of distinct websites where non-empty client hints (CHs) were detected in requests made to both first-party and third-party domains. Notably, `Sec-CH-UA-Platform-Version` and `Sec-CH-Model` were the most frequently observed CHs, mirroring the most received CH via the `getHighEntropyValues` method.

Specifically, `Sec-CH-UA-Platform-Version` was sent on 886 (1%) sites, and among those, it was transmitted to a third-party domain on 331 sites. Our findings were in line with Chrome usage metrics, where `Sec-CH-UA-Platform-Version` was sent on 1.67% of the sites [66]. Further, we found that on 622 sites, the low entropy UA-CHs were not sent in any request. Additional analysis of these cases revealed two main reasons. Firstly, certain sites utilized insecure HTTP connections, leading to the browser not sending UA-CHs. Secondly, a limited number of sites only triggered the initial request, with no subsequent requests made.

As illustrated in Table 13 in Appendix 6, the Advertising category stands out as the most common category of third-party domains where high-entropy UA-CH headers were sent to. Similar to the dominant categories associated with `getHighEntropyValues` calls, Advertising and Ad-Motivated tracking categories appear high in this table.

The top two client hints (`Sec-CH-UA-Model` and `Sec-CH-UA-Platform-Version`) were opted-in by first-party domains using the `Accept-CH` header, consistent with the client hints retrieved via the JS method. As depicted in

Table 7, our analysis found that `Sec-CH-UA-Model` and `Sec-CH-UA-Platform-Version` were requested in the `Accept-CH` header on 1,046 and 870 distinct sites, respectively. During our analysis, we discovered that certain `Accept-CH` CH header values include typos. Specifically, some of these values lacked the 'Sec-' prefix, such as `ua-platform-version`. For example, `Accept-CH` headers sent by bing.com contained values such as `UA-Full-Version`, `UA-Platform`, and `UA-Arch` (all three incorrect), `Sec-CH-UA-Full-Version-List` (correct). Similarly, issuu.com sent CHs in the `Accept-CH` header without the `Sec` prefix, such as `UA-Arch` and `UA-Full-Version`. In those particular cases, Chrome sent the correctly typed `Sec-CH-UA-Full-Version-List` to bing.com, but did not send any high-entropy hints to issuu.com.

Table 7: The number of distinct sites where UA-CHs were opted-in via the `Accept-CH` header.

Ent.	UA-CH Header	Num. Sites
high	Sec-CH-UA-Model	1,046
	Sec-CH-UA-Platform-Version	870
	Sec-CH-UA-Full-Version-List	824
	Sec-CH-UA-Arch	799
	Sec-CH-UA-Full-Version	538
	Sec-CH-UA-Bitness	443
	Sec-CH-UA-Wow64	354
low	Sec-CH-UA-Platform	818
	Sec-CH-UA	434
	Sec-CH-UA-Mobile	403

Table 8 shows that high-entropy CH headers are delegated to third-party domains on very few sites (0.4%). Primarily, sites delegate high-entropy UA-CHs to all domains using the '*' character, granting access to the delegated UA CH headers to all nested browsing contexts within the document, regardless of their origin.. fandom.com and google.com are the two domains that were most commonly delegated via the `Permissions Policy` header. Only on a few websites high-entropy UA-CHs delegated to specific origins. For example, on chase.com, `Sec-CH-UA-Full-Version-List` is delegated to the subdomains of Chase by specifying "https://*.chase.com". Similarly, certain sites such as idiva.com, indiatime.com, and mensxp.com permit sending high-entropy CHs to a specific origin (ase.clmbtech.com).

Table 8: The number of distinct sites delegating high-entropy UA-CHs via the `Permissions Policy` header. Most websites use '*' instead of specific origins to delegate to all possible origins. These cases are indicated in the 'Allow All' column.

Ent.	UA-CH Header	Allow All(*)	Total
high	ch-ua-platform-version	312	338
	ch-ua-model	310	337
	ch-ua-full-version-list	259	266
	ch-ua-arch	261	266
	ch-ua-bitness	221	225
	ch-ua-full-version	225	225
	ch-ua-wow64	218	222
low	ch-ua-platform	218	225
	ch-ua	4	6
	ch-ua-mobile	4	6

4.3 User-Agent Client Hint Opt-in and Delegation via HTML

Besides `Permissions Policy` and `Accept-CH` headers, opt-in and delegation of high-entropy UA-CHs can also be accomplished through the use of HTML `<meta>` element attributes and iframes' allow attributes. However, we observed that the usage of these methods for delegation was relatively low. Specifically, `http-equiv='accept-ch' <meta>` tag was employed for opt-in by first parties on 117 sites; the iframes' allow attribute was used for delegation to third parties on 32 sites, and the `http-equiv='delegate-ch' <meta>` tag was used for delegation to third parties on just 11 sites, as shown in Table 9. A potential reason for the limited use of the HTML-based method is that it introduces a delay in sending the high-entropy client hints since the browser needs to parse the HTML content to discover the opt-in. This may cause the browser to not send the requested high-entropy UA-CHs in the several requests that immediately follow the response—a phenomenon which we observed in our data, but not quantified due to limited scope.

4.4 Reduction in High-Entropy User-Agent Exposure

Finally, we compare the domains that could access high-entropy UA-CHs via the `getHighEntropyValues` method and those that actually use that privilege. Figure 5 shows that 25,052 (61%) of the 41,229 distinct third-party domains

Table 9: The number of distinct sites where delegation is accomplished through HTML.

Delegation	Num. Sites
http-equiv='accept-ch'	117
iframe-allow	32
http-equiv='delegate-ch'	11

encountered in our crawl serve active content with resource type reported by Puppeteer as "script" or "document" — the latter of which can contain inline scripts. Scripts from these domains *can* start collecting the high-entropy UA-CHs overnight, by adding a call to the `getHighEntropyValues` method. This could triple the number of distinct domains that access such high-entropy UA-CHs, which currently amount to 9,095 (22%) distinct third-party domains. While high-entropy UA-CHs can be accessed by third parties via HTTP headers, our results suggest that third-party delegation is rarely observed (Table 8).

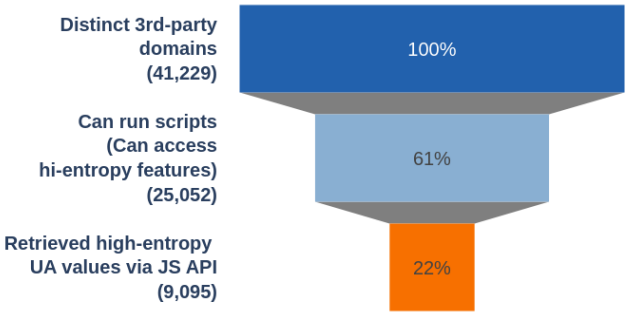


Figure 5: Comparing distinct 3rd-party domains that *can* run scripts and thus *can* access high-entropy UA-CHs and those that *accessed* high-entropy UA-CHs via the JavaScript API.

5. Discussion

The UA reduction efforts seem to have achieved their goal of limiting the potentially identifying information readily available in the UA HTTP header. Thanks to the *off-by-default* design of high-entropy UA-CH HTTP headers, we found that less than 1% of third parties receive high-entropy client hints over HTTP. On the other hand, the same high-entropy client hints are made accessible to scripts without any check for their origin or security context. This unfettered access enabled the collection and exfiltration of high-entropy UA-CHs by tracker scripts on close to 60% of the websites we studied. Particularly notable is that Google, primarily through its ads and analytics-related scripts, stands out as the most prevalent user of the `getHighEntropyValues` method and owns the most common domain (google-analytics.com) to which high-entropy UA-CHs are exfiltrated to by scripts (Table 4). We also note that while UA reduction efforts aim to limit covert tracking, Google's recently introduced Server-Side Tagging (SST) mechanism has almost the opposite effect [32]. SST shifts data sharing with third parties from client to server-side, potentially hiding the third parties that collect web users' detailed online activities.

Somehow surprisingly, even if Chrome engineers decide to further reduce identifying information available through the UA-CHs, they may not be allowed to do so—at least in the short term. Responding to the UK Competition and Markets Authority's investigation into the Privacy Sandbox project, Google has committed to ensuring that all the information previously present in the user-agent string will remain accessible through the UA CHs until the third-party cookie deprecation [13]. This was part of the commitments made to limit the potential anti-competition effects of the Privacy Sandbox project.

Finally, one might also question the relative significance of privacy risks posed by passive fingerprinting in relation to the widespread use of active and overt tracking—where Google is the most prevalent tracker entity through its ads, analytics and tag manager products [67]. Further, it is unclear to us whether overt cross-site tracking methods such as (active) browser fingerprinting or cookie syncing are more transparent or auditable from an end-user's perspective. While we recognize the improved transparency in our research, web measurement studies like ours cannot offer privacy assurances akin to a private-by-design system.

5.1 Limitations

Similar to other web measurement research, our study has several limitations regarding representativeness, reproducibility [18] and coverage. With a single form factor (desktop) and vantage point in the US, we cannot claim our results are representative of real users' experiences around the world. Further, websites may detect our crawler as an automated bot, and treat it differently [103]. We rely on TRC's anti-bot measures [7] which thwarts bot detection to a certain extent, but we acknowledge that its efficacy may be limited. With respect to coverage, our crawler only visited home pages and could not visit pages behind paywalls or login walls.

A specific limitation of our measurement was the inability to capture opt-ins at the connection time via HTTP/2 and HTTP/3 frames [39, 3] since those features are disabled in Puppeteer due to incompatibility with the Chrome DevTools Protocol [10]. We believe this limitation did not affect our prevalence measurements, but could skew the measurement of *mechanisms* used to convey the UA-CH opt-ins. In particular, our results do not capture the use of HTTP/2 or HTTP/3 frames for opting-in to high-entropy UA-CHs. In order to verify that we have not missed any HTTP/2 or HTTP/3 `Accept_CH` frames, we made sure that we can attribute all high-entropy UA-CH headers to a corresponding HTTP `Accept-CH`, `Permissions Policy` headers or `<meta>` tags, except on five websites. The most likely reason why we cannot pin down the opt-in mechanism on those five websites is `<meta>` tags that are temporarily injected and removed before we query the DOM, which is another minor limitation of our approach. Regardless, we note that these limitations do not affect our core measurement of high-entropy JavaScript API usage, which does not require any opt-in or delegation.

Finally, our measurement presents a snapshot of the current state of the web, which may change over time. For instance, while we found `getHighEntropyValues` method to be called on 59.2% of the websites, the older, now-reduced alternative `navigator.userAgent` property was accessed on 88.3% of the websites. We expect the number of websites where `getHighEntropyValues` is called to increase over time, as also shown by the upwards trend in the Chrome telemetry data for this particular JavaScript method [65].

6. Conclusion

We presented the first empirical study of the impact of user-agent string reduction and the introduction of high-entropy user-agent client hints in the Chrome browser. Through an extensive analysis of the top 100K websites using an instrumented crawler, we quantified the collection and exfiltration of high-entropy browser features via UA-CH HTTP headers and the newly introduced JavaScript API. Our findings showed that third-party scripts accessed high-entropy user agent client hints on nearly 60% of the analyzed sites. These scripts consist primarily of tracking and advertising scripts, with a notable presence of scripts owned by Google. Over 90% of the websites where high-entropy client hints were accessed via the JavaScript API, the obtained hints were exfiltrated to remote servers by tracker scripts. We find the use of high-entropy UA-CH headers to be very limited: only 1.3% of the first parties opt-in to receive high-entropy UA-CH headers, and even fewer websites (0.4%) delegate high-entropy hints to third-party domains. Overall, the efforts to reduce UA string identification appeared to be effective in minimizing the number of third parties that received the full UA string in HTTP headers, with only 3% of third-party domains receiving high-entropy UA-CH headers. Our study suggests that users and website operators may prefer JavaScript access for managing high-entropy browser features due to performance and operational considerations. In conclusion, while the UA reduction efforts appear to be effective for curbing passive browser fingerprinting, the introduction of high-entropy user-agent client hints has limited the overall privacy benefits of this major change. Given the extensive collection and exfiltration of highly identifying client hints by tracker scripts, we believe browser vendors should consider imposing stricter controls on scripted access to high-entropy user-agent client hints.

Acknowledgments

We thank Martin Thomson, Iness Ben Guirat, Zahra Moti and Luqman Zagi for their comments. Asuman Senol received funding from CYD Campus, armasuisse Science and Technology.

References

[1] Advertising Strategies for Targeting U.S. Hispanics. https://www.comscore.com/lat/Prensa-y-Eventos/Blog/Advertising-Strategies-

for-Targeting-U.S.-Hispanics, 2023. [Online; accessed 28. Jul. 2023].

[2] Victor Tan Ali Beyad. User-Agent Reduction deprecation trial. https://developer.chrome.com/en/blog/user-agent-reduction-deprecation-trial/, 2022-02-24. [Online; accessed 09. Jul. 2023].

[3] client-hints-infrastructure/reliability.md at main · WICG/client-hints-infrastructure. https://github.com/WICG/client-hints-infrastructure/blob/main/reliability.md#connection-level-settings, July 2023. [Online; accessed 28. Jul. 2023].

[4] Blake Anderson and David McGrew. Os fingerprinting: New techniques and a study of information gain and obfuscation. In *2017 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9. IEEE, 2017.

[5] Bulletin: Chrome's User-Agent Client Hints. https://support.appsflyer.com/hc/en-us/articles/12305445230737-Bulletin-Chrome-s-User-Agent-Client-Hints, 2023. [Online; accessed 28. Jul. 2023].

[6] D. Benjamin. Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility. https://datatracker.ietf.org/doc/html/rfc8701, 2020-01-01. [Online; accessed 12. Jul. 2023].

[7] Konrad Dzwinel et al. Brad Slayter, Sam Macbeth. DuckDuckGo Tracker Radar Collector. https://github.com/duckduckgo/tracker-radar-collector, 2021. [Online; accessed 01. Jan. 2023].

[8] Sec-CH-UA Brand being abused by sites, blocking clients. https://bugs.chromium.org/p/chromium/issues/detail?id=1433548, 2023-04-16. [Online; accessed 14. Jul. 2023].

[9] Yinzhi Cao, Song Li, and Erik Wijmans. (cross-) browser fingerprinting via os and hardware level features. In *Proceedings 2017 Network and Distributed System Security Symposium*. Internet Society, 2017.

[10] 1348106 – AcceptCHFrame breaks expected order of CDP events - chromium. https://crbug.com/1348106, 2023. [Online; accessed 1. Aug. 2023].

[11] Prepare for phasing out third-party cookies – Chrome Developers. https://developer.chrome.com/en/docs/privacy-sandbox/third-party-cookie-phase-out, 2023. [Online; accessed 28. Jul. 2023].

[12] Client hints not sent in headless chrome. https://bugs.chromium.org/p/chromium/issues/detail?id=1212793, 2021-05-24. [Online; accessed 01. Jul. 2023].

[13] Notice of intention to accept modified commitments offered by Google in relation to its Privacy Sandbox Proposals. https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/1036204/211126_FINAL_modification_notice.pdf, 2023. [Online; accessed 8. Aug. 2023].

[14] Contributors to Wikimedia projects. WebKit – Wikipedia. https://en.wikipedia.org/w/index.php?title=WebKit&oldid=1164531541, 2023. [Online; accessed 1. Aug. 2023].

[15] Solutions: Syndicated Audiences: CultureCode® - Claritas LLC. https://claritas.com/culture-code, 2022. [Online; accessed 28. Jul. 2023].

[16] Konrad Dzwinel et al. David Harbage, Sam Macbeth. DuckDuckGo Tracker Radar. https://github.com/duckduckgo/tracker-radar, 2020. [Online; accessed 01. Jan. 2023].

[17] Jon Davis. Safari 11.1. https://webkit.org/blog/8042/release-notes-for-safari-technology-preview-46/, 2017-12-20. [Online; accessed 09. Jul. 2023].

[18] Nurullah Demir, Matteo Große-Kampmann, Tobias Urban, Christian Wressnegger, Thorsten Holz, and Norbert Pohlmann. Reproducibility and replicability of web measurement studies. In *Proceedings of the ACM Web Conference 2022*, pages 533–544, 2022.

[19] Nick Doty. Mitigating Browser Fingerprinting in Web Specifications. https://www.w3.org/TR/fingerprinting-guidance, 2019. [Online; accessed 28. Jul. 2023].

[20] Early browser API accesses and function calls are missed. https://github.com/duckduckgo/tracker-radar-collector/issues/77, 2023. [Online; accessed 29. Jul. 2023].

[21] EasyList: General advert blocking filters. https://easylist.to/easylist/easylist.txt, 2023. [Online; accessed 11. Jun. 2023].

[22] EasyPrivacy: General advert blocking filters. https://easylist.to/easylist/easyprivacy.txt, 2023. [Online; accessed 11. Jun. 2023].

[23] Peter Eckersley. How unique is your web browser? In *Proceedings of the 10th International Conference on Privacy Enhancing Technologies (PETS)*, pages 1–18, 2010.

[24] Peter Eckersley. How unique is your web browser? *Privacy Enhancing Technologies*, pages 1–18, 2010.

[25] Emotional Branding - Hispanic Marketing & Public Relations website and podcast. https://hispanicmpr.com/resources/hmpr-products/emotional-branding, June 2008. [Online; accessed 28. Jul. 2023].

[26] Steven Englehardt, Jeffrey Han, and Arvind Narayanan. I never signed up for this! Privacy implications of email tracking. *Proc. Priv. Enhancing Technol.*, 2018(1):109–126, 2018.

[27] Steven Englehardt and Arvind Narayanan. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 1388–1401, 2016.

[28] Extend Sec-CH-UA to include a boolean field to identify if browser requires an update · Issue #76 · WICG/ua-client-hints. https://github.com/WICG/ua-client-hints/issues/76, 2023. [Online; accessed 29. Jul. 2023].

[29] Sec-CH-Bot for self declared 'good bots' that want to avoid analytics pollution · Issue #119 · WICG/ua-client-hints. https://github.com/WICG/ua-client-hints/issues/119, 2023. [Online; accessed 29. Jul. 2023].

[30] How To Target Facebook Ads To "Immigration to the United States" Audience | AdTargeting. https://adtargeting.io/facebook-ad-targeting/immigration-to-the-united-states, 2023. [Online; accessed 28. Jul. 2023].

[31] User-Agent Reduction. https://www.chromium.org/updates/ua-reduction, 2023. [Online; accessed 13. Jul. 2023].

[32] Why and when to use server-side tagging? https://developers.google.com/tag-platform/learn/sst-fundamentals/3-why-and-when-sst, 2023. [Online; accessed 8. Aug. 2023].

[33] User-Agent targeting. https://developers.google.com/authorized-buyers/rtb/useragent-targeting, 2023. [Online; accessed 28. Jul. 2023].

[34] Martin Grill and Martin Rehák. Malware detection using HTTP user-agent discrepancy identification. In *2014 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 221–226. IEEE, 2014.

[35] Raymond Hill. uBlock Origin - An efficient blocker for Chromium and Firefox. Fast and lean. https://github.com/gorhill/uBlock, 2017-12-20. [Online; accessed 19. Jul. 2023].

[36] Request Headers in the HTTP protocol. https://www.w3.org/Protocols/HTTP/HTRQ_Headers.html#user-agent, 1994-05-03. [Online; accessed 22. Jul. 2023].

[37] We Are All Human. New Study Reveals Hispanics Feel More Excluded From The American Culture And Disregarded By Big Brands. https://www.prnewswire.com/news-releases/new-study-reveals-hispanics-feel-more-excluded-from-the-american-culture-and-disregarded-by-big-brands-301857672.html, 2023. [Online; accessed 28. Jul. 2023].

[38] Client Hint Reliability. https://datatracker.ietf.org/doc/html/draft-davidben-http-client-hint-reliability#section-4, 2023. [Online; accessed 28. Jul. 2023].

[39] TLS Application-Layer Protocol Settings Extension. https://datatracker.ietf.org/doc/html/draft-vvv-tls-alps, 2023. [Online; accessed 28. Jul. 2023].

[40] Umar Iqbal, Steven Englehardt, and Zubair Shafiq. Fingerprinting the fingerprinters: Learning to detect browser fingerprinting behaviors. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1143–1161. IEEE, 2021.

[41] isbot. https://github.com/omrilotan/isbot, 2023. [Online; accessed 30. Jul. 2023].

[42] Nikhil Jha, Martino Trevisan, Luca Vassio, and Marco Mellia. The internet with privacy policies: Measuring the web upon consent. *ACM Transactions on the Web (TWEB)*, 16(3):1–24, 2022.

[43] Barry Pollard Johannes Henkel. Adding Rank Magnitude to the CrUX Report in BigQuery. https://developer.chrome.com/blog/crux-rank-magnitude/, 2021-03-09. [Online; accessed 13. Jul. 2023].

[44] Seb Joseph. 'The elephant in the room': Companies persist with fingerprinting as a workaround to Apple's new privacy rules. https://digiday.com/media/the-elephant-in-the-room-companies-persist-with-fingerprinting-as-a-workaround-to-apples-new-privacy-rules, April 2021.

[45] Terry Parris Julia Angwin, Jr. Facebook Lets Advertisers Exclude Users by Race. https://www.propublica.org/article/facebook-lets-advertisers-exclude-users-by-race, July 2023. [Online; accessed 28. Jul. 2023].

[46] Jeff Kline, Paul Barford, Aaron Cahn, and Joel Sommers. On the structure and characteristics of user agent string. In *Proceedings of the 2017 Internet Measurement Conference*, pages 184–190, 2017.

[47] Tadayoshi Kohno, Andre Broido, and Kimberly C. Claffy. Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, 2(2):93–108, 2005.

[48] Pierre Laperdrix, Walter Rudametkin, and Benoit Baudry. Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 878–894. IEEE, 2016.

[49] Lotame. `https://better.fyi/trackers/crwdcntrl.net/`, 2023. [Online; accessed 31. Jul. 2023].

[50] Marek. SSL fingerprinting for p0f — Idea of the day. `https://idea.popcount.org/2012-06-17-ssl-fingerprinting-for-p0f`, 2012. [Online; accessed 30. Jul. 2023].

[51] nikhiljha95 Martino Trevisan, Antonino Musmeci. Priv-Accept. `https://github.com/marty90/priv-accept`, 2022-04-13. [Online; accessed 13. Jul. 2023].

[52] Jonathan R Mayer and John C Mitchell. Third-party web tracking: Policy and technology. In *2012 IEEE symposium on security and privacy*, pages 413–427. IEEE, 2012.

[53] HTTP Client hints - HTTP | MDN. `https://developer.mozilla.org/en-US/docs/Web/HTTP/Client_hints`, 2023. [Online; accessed 31. Jul. 2023].

[54] NavigatorUAData - Web APIs | MDN. `https://developer.mozilla.org/en-US/docs/Web/API/NavigatorUAData`, 2023. [Online; accessed 22. Jul. 2023].

[55] Yoav Weiss Mike Taylor. User Agent Hints. `https://wicg.github.io/ua-client-hints`, 2023. [Online; accessed 14. Jul. 2023].

[56] Kyle Morehouse. Increase Ad ROI With Audience Suppression. `https://blog.adobe.com/en/publish/2017/01/24/increase-ad-roi-audience-suppression`, 2022. [Online; accessed 28. Jul. 2023].

[57] Keaton Mowery and Hovav Shacham. Pixel perfect: Fingerprinting canvas in HTML5. *Proceedings of W2SP*, 2012, 2012.

[58] 935216 - (client-hints) Implement Client-Hints HTTP header. https://bugzilla.mozilla.org/show_bug.cgi?id=935216, 2013-11-05. [Online; accessed 19. Jul. 2023].

[59] Analysis of Google's Privacy Budget Proposal | The Mozilla Blog. https://blog.mozilla.org/en/mozilla/google-privacy-budget-analysis, 2023. [Online; accessed 30. Jul. 2023].

[60] 1609304 - Reduce Gecko's User-Agent strings. https://bugzilla.mozilla.org/show_bug.cgi?id=1609304, 2020-01-14. [Online; accessed 19. Jul. 2023].

[61] Mozilla Specification Positions. https://mozilla.github.io/standards-positions/#ua-client-hints, 2023. [Online; accessed 23. Jun. 2023].

[62] Implement Navigator.userAgentData. https://bugzilla.mozilla.org/show_bug.cgi?id=1750143, 2022-01-13. [Online; accessed 14. Jul. 2023].

[63] Changing the UA String - MozillaWiki. https://wiki.mozilla.org/Changing_the_UA_String, 2023. [Online; accessed 17. Jul. 2023].

[64] Robert Nyman. User-Agent detection, history and checklist – Mozilla Hacks - the Web developer blog. https://hacks.mozilla.org/2013/09/user-agent-detection-history-and-checklist, 2013-09-12. [Online; accessed 14. Jul. 2023].

[65] Percentage of page loads over time – GetHighEntropyValues. https://chromestatus.com/metrics/feature/timeline/popularity/4520, 2023. [Online; accessed 25. Jul. 2023].

[66] Percentage of page loads over time – PlatformVersion. https://chromestatus.com/metrics/feature/timeline/popularity/3273, 2023. [Online; accessed 31. Jul. 2023].

[67] Barry Pollard. Third Parties | 2021 | The Web Almanac by HTTP Archive. https://almanac.httparchive.org/en/2021/third-parties, 2023.

[68] privacy-budget. https://github.com/mikewest/privacy-budget, 2023. [Online; accessed 30. Jul. 2023].

[69] Protecting Your Privacy Online. https://privacysandbox.com/, 2023. [Online; accessed 13. Jul. 2023].

[70] Page.setUserAgent() method. https://pptr.dev/api/puppeteer.page.setuseragent, 2023. [Online; accessed 20. Jul. 2023].

[71] Qualys SSL Labs - Projects / HTTP Client Fingerprinting Using SSL Handshake Analysis. https://www.ssllabs.com/projects/client-fingerprinting, July 2023. [Online; accessed 30. Jul. 2023].

[72] Ali Rasaii, Shivani Singh, Devashish Gosain, and Oliver Gasser. Exploring the Cookieverse: A Multi-Perspective Analysis of Web Cookies. In *International Conference on Passive and Active Network Measurement*, pages 623–651. Springer, 2023.

[73] Yoav Weiss Rowan Merewood. User-Agent Client-Hints. https://developer.chrome.com/en/articles/user-agent-client-hints/, 2021-09-10. [Online; accessed 09. Jul. 2023].

[74] Safari 11.1. https://developer.apple.com/library/archive/releasenotes/General/WhatsNewInSafari/Articles/Safari_11_1.html#//apple_ref/doc/uid/TP40014305-CH14-SW7, 2018-02-22. [Online; accessed 09. Jul. 2023].

[75] 182629 – [macOS, iOS] Expose OS marketing version in UserAgent. https://bugs.webkit.org/show_bug.cgi?id=182629#c6, 2023. [Online; accessed 28. Jul. 2023].

[76] Dennis Schubert. Freeze 'rv:' segment in the User Agent string to 'rv:109.0' to avoid erroneous IE11 detection. https://bugzilla.mozilla.org/show_bug.cgi?id=1805967, 2022-12-15. [Online; accessed 09. Jul. 2023].

[77] Henri Sivonen. Downgrade User Agent Client Hints to 'harmful' · Issue #552 · mozilla/standards-positions. https://github.com/mozilla/standards-positions/issues/552, 2021-07-12. [Online; accessed 19. Jul. 2023].

[78] Maciej Stachowiak. getHighEntropyValues should not return all values, only values that have been opted in. https://github.com/WICG/ua-client-hints/issues/151, 2021-11-02. [Online; accessed 23. Jun. 2023].

[79] Oleksii Starov, Phillipa Gill, and Nick Nikiforakis. Are You Sure You Want to Contact Us? Quantifying the Leakage of PII via Website Contact Forms. *Proc. Priv. Enhancing Technol.*, 2016(1):20–33, 2016.

[80] Yousouf Taghzouti, Antoine Zimmermann, and Maxime Lefrançois. Content negotiation on the Web: State of the art. *arXiv preprint arXiv:2204.10097*, 2022.

[81] Mike Taylor. User-Agent Client Hints & UA Reduction. https://github.com/w3ctag/design-reviews/issues/640, 2021-05-27. [Online; accessed 23. Jun. 2023].

[82] Mike Taylor. The hidden meaning of 537.36 in the Chromium User-Agent string. https://miketaylr.com/posts/2021/05/webkit-537-36-meaning.html, 2021-05-28. [Online; accessed 22. Jul. 2023].

[83] Mike Taylor. Return empty string on Linux for platform-version hint. https://bugs.chromium.org/p/chromium/issues/detail?id=1381304&q=platformVersion&can=2, 2022-11-04. [Online; accessed 02. Jul. 2023].

[84] Mike Taylor and Yoav Weiss. User Agent Hints. https://wicg.github.io/ua-client-hints, 2023. [Online; accessed 14. Jul. 2023].

[85] Martin Thomson. UA Client Hints is less than harmful. https://github.com/mozilla/standards-positions/pull/579, 2021-09-29. [Online; accessed 02. Jul. 2023].

[86] About Events API | TikTok For Business. https://ads.tiktok.com/help/article/events-api, 2023. [Online; accessed 28. Jul. 2023].

[87] tls-fingerprinting/fingerprintls at master · LeeBrotherston/tls-fingerprinting. https://github.com/LeeBrotherston/tls-fingerprinting/tree/master/fingerprintls, 2023. [Online; accessed 30. Jul. 2023].

[88] User-Agent Client Hints API. https://developer.mozilla.org/en-US/docs/Web/API/User-Agent_Client_Hints_API, 2023. [Online; accessed 28. Jul. 2023].

[89] Intent to Prototype: User-Agent Client Hints GREASE Update. https://groups.google.com/a/chromium.org/g/blink-dev/c/ueudFsZzT1M, 2021-10-14. [Online; accessed 22. Jul. 2023].

[90] Merge pull request #253 from kyraseevers/issue248 · WICG/ua-client-hints@56df868. https://github.com/WICG/ua-client-hints/commit/56df868c6c3564461b4c0a42ea603b80fa94c8d2, 2023. [Online; accessed 28. Jul. 2023].

[91] WebAIM: History of the browser user-agent string. https://webaim.org/blog/user-agent-string-history, 2008-09-03. [Online; accessed 22. Jul. 2023].

[92] uBlock Origin - An efficient blocker for Chromium and Firefox. Fast and lean. https://github.com/gorhill/uBlock, 2023. [Online; accessed 10. Jul. 2023].

[93] The UK Uses Targeted Facebook Ads To Deter Migrants. Now Meta Is Releasing the Data. https://newlinesmag.com/reportage/the-uk-uses-targeted-facebook-ads-to-deter-migrants-now-meta-is-releasing-the-data, 2023. [Online; accessed 28. Jul. 2023].

[94] Antoine Vastel, Pierre Laperdrix, Walter Rudametkin, and Romain Rouvoy. Fp-stalker: Tracking browser fingerprint evolutions. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 728–741. IEEE, 2018.

[95] Types of documents W3C publishes. https://www.w3.org/standards/types/#x2-pre-standardization-proposals-notes, 2023. [Online; accessed 31. Jul. 2023].

[96] 180365 – Limit user agent versioning to an upper bound. https://bugs.webkit.org/show_bug.cgi?id=180365, 2017. [Online; accessed 30. Jul. 2023].

[97] Contribute to WICG/ua-client-hints. https://github.com/WICG/ua-client-hints/issues?q=is%3Aissue+author%3Aothermaciej, 2021-11-02. [Online; accessed 01. Jul. 2023].

[98] Yoav Weiss. Intent to deprecate and freeze: The user-agent string. https://groups.google.com/a/chromium.org/g/blink-dev/c/-2JIRNMWJ7s/m/yHe4tQNLCgAJ, 2020-01-14. [Online; accessed 13. Jul. 2023].

[99] Alexandra White. User-Agent reduction. https://developer.chrome.com/docs/privacy-sandbox/user-agent/, 2021-11-09. [Online; accessed 8. Aug. 2023].

[100] Ting-Fang Yen, Yinglian Xie, Fang Yu, Roger Peng Yu, and Martin Abadi. Host fingerprinting and tracking on the web: Privacy and security implications. In *NDSS*, volume 62, page 66, 2012.

[101] YngveNPettersen. Brand info is being used to block clients. https://github.com/WICG/ua-client-hints/issues/293, 2022-02-16. [Online; accessed 14. Jul. 2023].

[102] Michal Zalewski. p0f Unofficial Git-Repo. https://github.com/p0f/p0f, 2019-02-21. [Online; accessed 01. Jul. 2023].

[103] David Zeber, Sarah Bird, Camila Oliveira, Walter Rudametkin, Ilana Segall, Fredrik Wollsén, and Martin Lopatka. The representativeness of automated web crawls as a surrogate for human browsing. In *Proceedings of The Web Conference 2020*, pages 167–178, 2020.

Appendix

navigator.userAgentData interface description (IDL)

```
navigator.userAgentData
// OUTPUT:
"brands":[
    {
        "brand":" Not A;Brand",
        "version":"24"
    },
    {
        "brand":"Chromium",
        "version":"113"
    },
    {
        "brand":"Google Chrome",
        "version":"113"
],
"mobile": false,
"platform": "Linux"
```

Listing 4: Sample output of navigator.userAgent interface's properties

```
dictionary NavigatorUABrandVersion {
  DOMString brand;
  DOMString version;
};

dictionary UADataValues {
  DOMString architecture;
  DOMString bitness;
  sequence<NavigatorUABrandVersion> brands;
  DOMString formFactor;
  sequence<NavigatorUABrandVersion> fullVersionList;
  DOMString model;
  boolean mobile;
  DOMString platform;
  DOMString platformVersion;
  DOMString uaFullVersion; // deprecated in favor of
    fullVersionList
  boolean wow64;
};

dictionary UALowEntropyJSON {
  sequence<NavigatorUABrandVersion> brands;
  boolean mobile;
  DOMString platform;
};

[Exposed=(Window,Worker)]
```

```
interface NavigatorUAData {
  readonly attribute FrozenArray<NavigatorUABrandVersion> brands;
  readonly attribute boolean mobile;
  readonly attribute DOMString platform;
  Promise<UADataValues> getHighEntropyValues(sequence<DOMString>
    hints);
  UALowEntropyJSON toJSON();
};

interface mixin NavigatorUA {
  [SecureContext] readonly attribute NavigatorUAData userAgentData
    ;
};

Navigator includes NavigatorUA;
WorkerNavigator includes NavigatorUA;
```

Listing 5: NavigatorUAData interface

Crawler's UA Override Values

We used the following details to override the crawler's UA string and client hints via the *page.setUserAgent* [70] method of the Chrome DevTools Protocol.

```
brands: [
    {brand: 'Chromium', version: '113'},
    {brand: 'Not-A.Brand', version: '24'}
],
fullVersionList: [
    {brand: 'Chromium', version: '113.0.5672.63'},
    {brand: 'Not-A.Brand', version: '24.0.0.0'}
],
fullVersion: "113.0.5672.63",
platform: 'Linux',
platformVersion: '5.15.0',
architecture: 'x86',
model: '',
mobile: false,
bitness: '64',
wow64: false,
```

Listing 6: Overriden UA Values

Fingerprinting Detection Heuristics

Below you can find the heuristics used for the detection of fingerprinting attempts in this study. These heuristics were first proposed by Englehardt and Narayanan [27] and then improved by Iqbal et. al [40].

Canvas Fingerprinting Identification: A script categorized as a canvas fingerprinting script based on the following criteria:

1. The script calls the `fillText` or `strokeText` methods to write text on a canvas element, and style adjustments are made using the `fillStyle` or `strokeStyle` methods of the rendering context.
2. The script uses the `toDataURL` method to extract the canvas image.
3. The script does not call the `save`, `restore`, or `addEventListener` methods on the canvas element.

WebRTC Fingerprinting Identification: A script is categorized as a WebRTC fingerprinting script according to the following conditions:

1. The script calls the `createDataChannel` or `createOffer` methods of the WebRTC peer connection.
2. The script calls the `onicecandidate` or `localDescription` methods of the WebRTC peer connection.

Canvas Font Fingerprinting Identification: A script is recognized as a canvas font fingerprinting script based on these guidelines:

1. The script modifies the `font` property of a canvas element to include more than 20 different fonts.
2. The script calls the `measureText` method of the rendering context more than 20 times.

AudioContext Fingerprinting Identification: A script is identified as an AudioContext fingerprinting script if it satisfies the following conditions:

1. The script calls any of the following methods of the audio context: `createOscillator`, `createDynamicsCompressor`, `destination`, `startRendering`, or `oncomplete`.

Supported Encodings & Hashes for Leak Detection

Encodings: Base16, Base32, Base58, Base64, Urlencode, Entity, LZstring, Hashes: MD5, SHA1, SHA256, SHA512

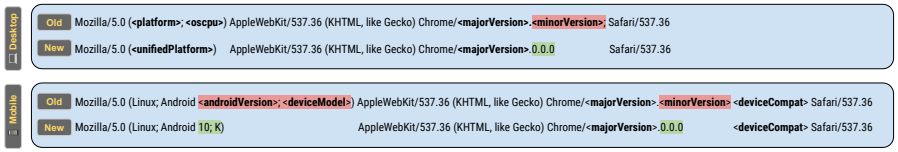


Figure 6: User-agent string format change for desktop and mobile Chrome browsers after UA reduction.

Popularity of Sites Which Call getHighEntropyValues

Table 10: The distribution of CrUX websites that call getHighEntropyValues. Results are sliced by site rank. and based on successful visits.

	Sites Calling getHighEntropyValues	
Rank Interval	**Website (count)**	**Website (%)**
1 to 1K	508	55.8%
1K to 5K	2,029	57.2%
5K to 10K	2,671	59.6%
10K to 50K	21,092	58.9%
50K to 100K	26,092	58.0%
1 to 100K	52,392	58.4%

Purposes of Third-Party Scripts That Exfiltrate High-entropy Values

Table 11: The top third-party domain categories along with the respective counts of websites where we observed at least one third-party domain exfiltrates high-entropy UA values to their endpoints. The categorization is based on Tracker Radar's domains,[14]acknowledging the possibility of a website belonging to multiple categories simultaneously.

Category	Num. Sites
Advertising	41,124
Audience Measurement	27,526
Ad Motivated Tracking	27,253
Third-Party Analytics Marketing	26,220
Analytics	25,509
CDN	15,512
Online Payment	15,084
Social - Share	2,471
Embedded Content	2,154
Action Pixels	625

Scripts Calling getHighEntropyValues and also Fingerprinting

Table 12: We found 2,190 distinct third-party fingerprinting scripts. 12.5% of them also call the `getHighEntropyValues` method.

	All	Third party	Tracking related
Fingerprinting	4,689	2,190	784
FPing and `getHighEntropyValues` calls	545	274	157

[14]https://github.com/duckduckgo/tracker-radar

Purposes of third-party scripts that receive high-entropy UA-CH Headers

Table 13: The top third-party domain categories and the corresponding number of websites where at least one high-entropy UA-CHs was sent to a third party.

Category	**Num. Sites**
Advertising	99
Ad Motivated Tracking	96
Embedded Content	90
CDN	86
Online Payment	65
Analytics	65
Audience Measurement	64
Third-Party Analytics Marketing	53
Action Pixels	22
Ad Fraud	22

Publication

Targeted and Troublesome: Tracking and Advertising on Children's Websites

Publication Data

Moti, Z., Senol, A., Bostani, H., Borgesius, F. Z., Moonsamy, V., Mathur, A., and Acar, G. Targeted and Troublesome: Tracking and Advertising on Children's Websites. In *2024 IEEE Symposium on Security and Privacy (SP)* (Los Alamitos, CA, USA, may 2024), IEEE Computer Society, pp. 117–117

Contributions

- I co-developed the web crawler to collect data and I ran the crawls on the cloud servers. I developed the method to detect targeted ads and wrote initial analysis scripts for online tracking measurements, including browser fingerprinting.

Targeted and Troublesome: Tracking and Advertising on Children's Websites

Zahra Moti[2], Asuman Senol[1], Hamid Bostani[2], Frederik Zuiderveen Borgesius[2], Veelasha Moonsamy[3], Arunesh Mathur[4], and Gunes Acar[2]

[1] COSIC, KU Leuven
[2] Radboud University
[3] Ruhr University Bochum
[4] Independent Researcher

Abstract. On the modern web, trackers and advertisers frequently construct and monetize users' detailed behavioral profiles without consent. Despite various studies on web tracking mechanisms and advertisements, there has been no rigorous study focusing on websites targeted at children. To address this gap, we present a measurement of tracking and (targeted) advertising on websites directed at children. Motivated by the lack of a comprehensive list of child-directed (i.e., targeted at children) websites, we first build a multilingual classifier based on web page titles and descriptions. Applying this classifier to over two million pages from the Common Crawl dataset, we compile a list of two thousand child-directed websites. Crawling these sites from five vantage points, we measure the prevalence of trackers, fingerprinting scripts, and advertisements. Our crawler detects ads displayed on child-directed websites and determines if ad targeting is enabled by scraping ad disclosure pages whenever available. Our results show that around 90% of child-directed websites embed one or more trackers, and about 27% contain targeted advertisements—a practice that should require verifiable parental consent. Next, we identify improper ads on child-directed websites by developing an ML pipeline that processes both images and text extracted from ads. The pipeline allows us to run semantic similarity queries for arbitrary search terms, revealing ads that promote services related to dating, weight loss, and mental health, as well as ads for sex toys and flirting chat services. Some of these ads feature repulsive, sexually-explicit and highly-inappropriate imagery. In summary, our findings indicate a trend of non-compliance with privacy regulations and troubling ad safety practices among many advertisers and child-directed websites. To ensure the protection of children and create a safer

online environment, regulators and stakeholders must adopt and enforce more stringent measures.

Keywords – online tracking, advertising, children, privacy

1. Introduction

The proliferation of online tracking for analytics, behavioral advertising, and marketing has resulted in over a decade's worth of research into this ecosystem. Prior research has shown that not only is online tracking rampant on the web [42] but that trackers use increasingly-invasive tracking mechanisms—e.g., third-party cookies, tracking pixels, evercookies, and browser fingerprinting [15, 50, 3, 42]—to relentlessly build detailed profiles of users across the web without any consent for targeted advertising.

Such privacy concerns aside, online advertising has shown to be problematic in other ways. Ads and ad networks are a vector for distributing ransomware, malicious programs, and cryptojackers—posing a serious security threat to users [135, 79, 119, 62, 2, 83, 134, 105, 120]. Ad networks also suffer from click fraud, which is estimated to reach $100 billion in 2023 [133, 46]. Finally, online ads often contain clickbait, untrustworthy, or distasteful content that peddle software downloads, listicles, and health supplements—all of which users find problematic to their online experience [138].

While online tracking and targeted advertising pose a threat to users of all ages, children especially bear an acute cost. Children may not fully understand the consequences of online tracking and revealing their personal data online [140, 73], but they yield immense "pester power" to influence their parents' purchase decisions [77]. Thus, children are an attractive target audience for advertisers and marketers alike [77, 74], as they are more vulnerable to persuasive advertising [68, 21, 22], and susceptible to harmful content [106, 9].

Despite the aforementioned evidence that suggests a differential impact on children, there is little empirical research on online tracking and advertising practices on children's websites. The lack of a comprehensive and updated list of websites directed at children poses a major challenge for studying children's websites. Previous large-scale internet measurement studies have relied on popular website lists such as Tranco [78] and Alexa [8] (before it shut down in 2021 [131]), but these lists may not specify website categories, and even when they do, the website categories may not be reliable and comprehensive [84, 128]. As a result, prior work [130, 22] has only examined online tracking on at most a hundred children's websites and has been restricted in scope and methods—lacking a comprehensive investigation of both online tracking and advertising. To overcome this limitation, we built our own repository of child-directed websites. We trained a text-based classifier that detects children's websites using HTML metadata fields such as `<title>` and `<description>`. The classifier is based on a pre-trained multilingual model that we fine-tuned for our binary classification task. Applying the classifier to the Common Crawl dataset [28], we compiled a

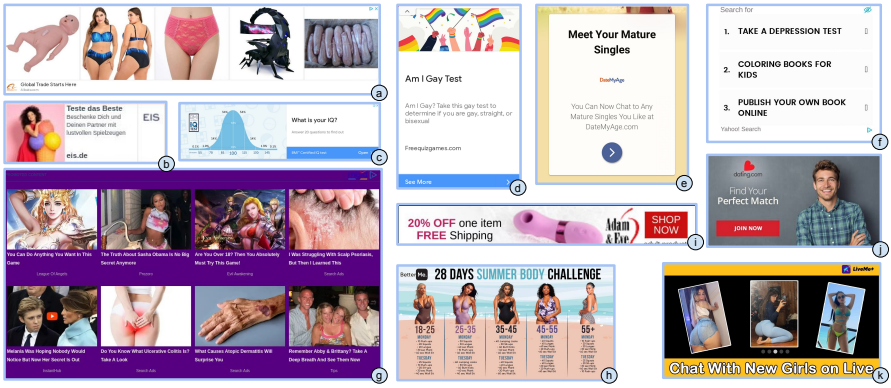


Figure 1: A sample of improper ads found on child-directed websites in our crawls.

list of 2K manually verified child-directed websites.

To study several online tracking, ad targeting, and problematic ad practices, we crawl our list of 2K child-directed websites—varying the location (five vantage points) and form factors (desktop & mobile). Starting with ad targeting, we study the extent to which ads that appear on children's websites are targeted—a practice that has come under increasing scrutiny both in the EU and the US [34, 115, 30]. We then present an exploratory analysis of ads from categories deemed problematic for children, such as dating, weight loss, mental health, and ads that contain racy content. Next, we turn to online tracking, which is a necessary ingredient of behavioral advertising. We study the ecosystem of trackers and specifically quantify the prevalence of trackers, cookies, and use of browser fingerprinting techniques such as Canvas, Canvas Font, AudioContext Fingerprinting, and WebRTC local IP discovery [42].

Our work is especially pertinent in light of impending regulatory changes. In the US, there have been calls [30] to update the Children's Online Privacy Protection Act (COPPA) [53] in order to prohibit "internet companies from collecting personal information from users who are 13 to 16 years old without their consent" and to "ban targeted advertising to children and teens." The current US President Joe Biden has called for a ban on collecting data on and serving targeted ads to children [115]; whereas in the EU, the upcoming Digital Services Act (DSA) will specifically prohibit ads targeted at children [34].

Our research seeks to offer empirical evidence on advertising and tracking practices employed on children's websites by making the following contributions:

- Using a lightweight classifier based on web page metadata fields, we build a repository of child-directed websites and crawl them to measure tracking and advertising practices using multiple vantage points and form factors (desktop & mobile).
- We measure targeted ads using two ad vendors' (Google and Criteo) ad disclosure statements, and find that targeting is enabled for 73% of the ads we could measure.
- Using text and images extracted from the ads, we detect *racy* ads, and ads about *weight loss*, *dating*, and *mental health* using semantic similarity search based on a lightweight, multilingual language model. While this content analysis is exploratory, our method enables human-in-the-loop investigations with arbitrary queries, and it paves the way for the automatic content analysis of ads.
- We also find ads linking to malicious content, improper ads of sex toys, dating services, and ads containing sexually suggestive images (Figure 1).

All the data and software from our study were made available to researchers.[5]

2. Related Work

2.1 Web Tracking Measurements

Over the past decade, several web privacy measurements have shown the scale and complexity of online tracking [104, 42, 65, 26, 32]. Research on *stateful* tracking has examined how unique tracking identifiers are stored on the client side [107] using cookies [72, 27], localStorage [15], cache (ETags) [15], or other client-side storage mechanisms. On the other hand, research on *stateless* tracking has examined the use of fingerprinting, a mechanism that exploits differences in browsers and devices to obtain a likely unique identifier [85]. Past research has shown that there are various fingerprinting vectors, including fonts, clock skew, GPUs, audio hardware, installed writing scripts and browser extensions, among others [40, 71, 91, 42, 24, 75, 76]. Research on defense against fingerprinting has contributed methods to detect fingerprinting, tracking and advertising [111, 3, 42, 65, 66, 110]. Our study borrows heuristics from prior work [42, 65] to detect fingerprinting scripts, and we use existing filter lists to identify trackers and advertisers.

[5]We share the list of identified child-directed websites and a sample of advertisement disclosures on https://github.com/targeted-and-troublesome/.

2.2 Tracking & Ads On Children's Media

Motivated by the challenges posed by ads to children, Cai and Zhao [22] manually labeled ads displayed on 117 children's websites. They found that 68% of the websites featured ads, and less than half complied with COPPA. The authors also argued that children are unlikely to distinguish many ads from the website's original content. Vlajic et al. [130] investigated online tracking on twenty websites from Alexa's "Kids and Teens" category [8] from two vantage points (EU & US). The authors manually analyzed the HTTP headers and quantified hidden images (i.e., likely tracking pixels) loaded from ads and analytics domains. Compared to this past work, we study orders of magnitude more websites, follow more rigorous tracking measurement methods, and compare results across different vantage points. Additionally, we automatically detect targeted ads using ad disclosure pages and present an exploratory analysis of the content of ads that appear on children's websites.

Focusing on mobile platforms, Reyes et al. [103] dynamically analyzed around 6,000 free children's Android apps and found that most apps violate COPPA due to their use of third-party SDKs.

2.3 Improper and Malicious Ads

A recent line of research has investigated the content of online ads. Zeng et al. [138] conducted a survey with 1,000 participants to determine the type of advertising content (e.g., chumboxes, clickbait, political, and low-quality content) that makes people dislike ads. In [137], the same authors also studied problematic ads in the news and misinformation websites, where they found problematic ads served by native ad platforms. In a study leading to the 2020 US elections, Zeng et al. [139] found that ads for misleading political polls that aim to hoover email addresses are widely used in online political advertising. Subramani et al. [119] studied the role of web push notifications in ad delivery, especially malicious ads. Through a large-scale study of malicious ads, Zarras et al. [135] showed that some ad exchanges are more prone to serving malicious ads due to inadequate detection. Akgul et al. [7] examined influencer VPN ads on YouTube and found advertisements disseminating misleading claims about online safety. Ali et al. [10] measured how the distribution of potentially harmful ads on Facebook varies across users. Venkatadri et al. [129] used Facebook's advertiser interface to study how Facebook obtains personal identifiers used in advertising. In a concurrent work, Zhao et al. [142] analyzed mobile ads aimed at children, uncovering inappropriate advertisements by certified mobile ad SDKs. Medjkoune et al. [87] showed that advertisers can target ads to

children by placing their ads in children-focused videos—bypassing YouTube's age restrictions.

2.4 Ad Transparency

In response to concerns about targeted advertising, ad networks and platforms have offered ad transparency interfaces that allow users to ascertain when and how they are being targeted. Andreou et al. [12] investigated Facebook's ad explanations and found that they are often incomplete or misleading. Researchers have also argued that ad networks should provide users with interpretable explanations and increase the visibility of disclosure mechanisms [45].

Bin Musa and Nithyanand [92] developed ATOM, a technique for determining data sharing between online trackers and advertisers. They used simulated personas to crawl websites, collect ad images, and conduct statistical analyses to identify correlations between tracker presence and advertiser behavior. Liu et al. [81] developed a framework called *AdReveal* to investigate different mechanisms used in online targeted advertising. Vallina et al. [127] used statements found in Google's ad disclosure pages in their crowdsourced measurement of online behavioral advertisements. To detect stealthy ads that aim to bypass adblockers, Storey et al. [117] developed an extension that detects the AdChoices icon using perceptual hashing. While we considered applying Storey et al.'s method, we found URL-based detection of ad disclosure links (§4.6) to be more reliable and efficient.

2.5 Website Categorization

The majority of studies on web categorization have focused on text-based classifiers because most web content and metadata are text-based [114, 141]. Various studies used machine learning models such as BERT and recurrent neural networks to learn contextual representations and features of web pages using meta tags and body content [58, 141, 114]. Other researchers proposed image-based web page classification techniques using pre-trained convolutional neural networks and using Google image search results [14, 82]. In our work, we built a lightweight classifier by fine-tuning an existing distilled language model and using text-based website metadata to detect child-directed websites.

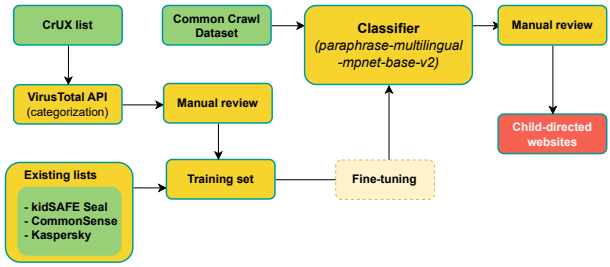


Figure 2: Pipeline for building a list of child-directed websites.

3. Building a List of Child-Directed Websites

It is estimated that there are more than one billion websites on the Internet [94], but only a small fraction are targeted at children. A central challenge, therefore, is identifying the websites that contain content directed to children. We initially searched and found three curated lists of children's websites: kidSAFE Seal Program [88], CommonSense (filtered for children below the age of 13) [18], and a list compiled by Kaspersky [122]. Unfortunately, these lists contained only a total of 355 websites, some of which were no longer online.

To expand our limited list, we experimented with web categorization services such as McAfee, WebShrinker, and SimilarWeb, but decided to use VirusTotal's (academic) API because other services were either not freely available or did not let us query in bulk. VirusTotal aggregates category labels from third-party scanners and categorization services, including BitDefender and TrendMicro [29]. We used the VirusTotal API to retrieve web category data for the top one million websites from the Chrome User Experience Report (CrUX) list from May 2022 [31]. We observed VirusTotal's rate limits (20K/day/per academic license) during the process, which took roughly four weeks. By searching for substrings "kid" and "child" in the returned category labels and removing false positives (such as "Child abuse"), we obtained 1,264 websites categorized as related to children. However, our manual verification of these websites following the criteria presented in Appendix 8 revealed that 68.6% of them were false positives, yielding only 396 child-directed websites. Note that the low accuracy and inconsistency of domain classification/categorization services align with findings from prior work [128]. Combining our initial 355 websites with our verified list of 396 websites and removing all inaccessible (5) and duplicate (164) websites, we obtained a total of 582 child-directed websites.

Motivated by the lack of accurate, up-to-date, and comprehensive sources of child-directed websites, we built a classifier to detect child-directed websites

using the list of 582 websites as labeled data. Figure 2 illustrates the training and fine-tuning process. We define "child-directed websites" as websites that are primarily intended for use by children, and contain content, activities, or other features that are likely to be appealing to children (see Appendix 8, for our criteria). Note that our labeling criteria do not fully overlap with COPPA's definition [53]; for example, we do not require that sites have the actual knowledge of collecting children's data. Thus, we do not claim to measure compliance with COPPA or other relevant laws. In our study, children refer to individuals under 13, aligning with US and EU regulations (§6.1).

3.1 Labeled Data for ML Classifier

Many web page classification methods use the entire text of the page [58] and its images [82], which can be resource-intensive and time-consuming. Alternatively, researchers have explored web page classification on metadata fields such as `<title>`, `<description>`, and `<keywords>`, which tend to be shorter and shown to have strong correlations with the topic of web pages [114]. We followed the latter approach for its computational efficiency and reasonable accuracy. Our preliminary analysis of over 500K web pages from the most popular one million websites in the Common Crawl dataset [28] showed that more than 97% of the websites have a title, 63% of the websites include a description, and 24% contain a keywords meta tag. Based on these availability statistics, we used the titles and descriptions for classification, leaving out the keywords. To extract the titles and descriptions, we used the following HTML tags: `title, description, og:[title|description], and twitter:[title|description]`. Applying this method to the WAT metadata files from the June-July 2022 Common Crawl snapshot [28], we extracted the titles and descriptions, limiting ourselves to the top million websites in the Tranco [78] or the CrUX [31] list. We further refined our data by keeping a single page with the shortest URL from each hostname, which is more likely to be the home page. This resulted in metadata from 2.28 million pages, which included pages from the subdomains of the top million Tranco and CrUX websites. We also extracted the same title and metadata information from the 582 known child-directed websites using a simple script based on Playwright [96]. In both instances, when the page had more than one description or title available, we picked the longest one.

After completing the data collection process, we constructed a training set for our classifier. For negative samples, we randomly selected 2,500 of the 2.28 million pages and manually checked to remove children's websites. Our positive samples consisted of 576 title-description pairs after filtering out websites with titles shorter than ten characters.

3.2 Building the ML Classifier

Our training data contained a limited number of labeled samples and our input consisted of text-based meta fields, potentially in multiple languages. This made designing naive classifiers such as bag-of-words and TF-IDF less suitable for our task. Instead, we employed a pre-trained and multi-lingual language model. Pre-trained models have proven to be adequate for general text classification tasks, but they need to be fine-tuned for more specific downstream tasks [58]. In particular, we decided to use the *Paraphrase-Multilingual-MPNet-base-v2* (PM-MPNet-v2) model from the *SentenceTransformers* [100, 109] library, which is a pre-trained multilingual and distilled model based on the MPNet method [113]. The distillation process [100, 101] involves training a smaller model (student) to mimic the behavior of a larger model (teacher). In particular, PM-MPNet-v2 is fine-tuned with a large number of paraphrased sentences in more than 50 languages [100]. However, PM-MPNet-v2 cannot be directly used for text classification since it only produces embeddings that are useful for semantic similarity-related tasks. Thus, we used HuggingFace's *Trainer* API [61] and *AutoModelForSequenceClassification* [60] class to fine-tune the model and add a binary classification layer on top of the PM-MPNet-v2's embedding outputs. As input to the classifier, we used the concatenation of title and descriptions since this combination gave the best accuracy compared to using title or description alone. In particular, we fine-tuned the model to detect child-directed websites using the training set explained in §3.1. We used the HuggingFace Transformers [99] and Ray Tune libraries' Population Based Training (PBT) algorithm [123, 67] to find the best-performing hyperparameters (batch size=12, epochs=2, and learning rate=4.2e-05). The fine-tuning process took roughly five minutes on a consumer-grade GPU (GeForce RTX 3080 Ti).

To reduce false positives, we employed the modified "Classify-Verify" technique [116], which involves setting an acceptance threshold t, and accepting a prediction only if it is above t. Following Juarez et al. [69], we choose the threshold that maximizes $F_{\beta=0.5}$, which gives more weight to precision to reduce false positives. F_β is a weighted harmonic mean of precision and recall, adjustable to emphasize either metric according to the classification task's needs [108]. A grid search of different threshold values shows that the maximum $F_{\beta=0.5}$ is achieved when $t = 0.93$, which reduces the false positive by 50%. Ultimately, our classifier achieved a precision of 86% and a recall of 70% using 10-fold cross-validation, as detailed in Table 1.

Table 1: Classification results before and after applying threshold (with 10-fold cross-validation).

	Precision	Recall	F-beta	TP	FP
Without threshold	0.79	0.81	0.79	47	12
With threshold	0.86	0.70	0.82	40	6

3.3 The List of 2K Children's Websites

Using the fine-tuned classifier, we calculated the label and probability score for 2.28M web pages from Common Crawl, excluding websites used in the training process. This process took roughly 5 hours. Our classifier identified 53,092 web pages as children's websites. Due to time constraints, we focused on manually verifying the top 2,500 websites sorted by classifier probability, starting with websites that are most likely to be child-directed. An evaluation of our classifier and the details of our manual verification process can be found in Appendix 8.

Our final list contained 2,004 websites in 48 distinct languages after eliminating false positives and deduplicating websites by their registrable domain (TLD+1). English was the most prevalent, accounting for 63% of all websites. The prevalence of other prominent languages, including Russian, Spanish, French, German, and Portuguese, ranging between 3% and 6%. The list included 582 websites from the training data and 1,422 websites identified by the classifier.

Website ranks: 1,422 of the 2,004 websites were ranked in the top 1 million Tranco list (median rank 304K). While over a quarter of the websites are in the top 200K ranks, websites from all popularity levels are captured in our list. 404 of the 582 websites that are not ranked by Tranco were ranked in the top one million by the CrUX list. Only 163 (8%) websites were not ranked either by Crux or Tranco in the top one million.

DNS0 Kids filter check: DNS0 Kids [35] is a domain name resolver that detects and filters out content that is not suitable for children such as adult, dating, and copyright-infringing content. To find out the status of the websites in our list, we compared DNS0 Kids with CloudFlare's DNS resolver. If a website can be resolved by CloudFlare, but not by DNS0, we treated it as blocked. We found that only ten (0.5%) of the 2,004 websites in our list were blocked by DNS0. Reviewing these ten websites, we found six of them to contain pirated videos, including cartoons. The remaining four websites contained activities for children but it was not obvious to us why they were blocked.

4. Web Tracking and Advertising Measurements

To assess the prevalence of trackers, fingerprinting scripts, and (targeted) advertisements on child-directed websites, we extended Tracker Radar Collector (TRC) [124]. TRC is a Puppeteer-based [97] web crawler, which consists of modules called *collectors* that record different types of data during a crawl, such as HTTP requests/responses, cookies, screenshots, and JavaScript API calls. New collector modules can be easily added to collect data necessary to perform different web measurements such as ours. Specifically, we added the following collectors to TRC:

- `FingerprintCollector` (§4.1): detects fingerprinting related function calls and property accesses
- `LinkCollector` (§4.3): extracts inner page links
- `VideoCollector` (§4.5): captures the crawl video
- `AdCollector` (§4.6): detects ads and scrapes ad disclosures

We also used the existing TRC collectors, including `RequestCollector` to capture request/response details and detect tracking-related requests (§4.2), `TargetCollector` to detect newly-opened tabs in §4.6, `CookieCollector` to analyze cookies, and finally `CMPCollector` (§4.4) to interact with the consent dialogs and consent management platforms (CMP). We used TRC's anti-bot measures [124], which thwarts bot detection to a certain extent by overwriting artifacts typically probed by anti-bot scripts (e.g., `navigator.plugins`, `Notification.permission`) [37]. For the mobile crawls, we emulated a mobile browser using TRC's built-in features to spoof viewport dimensions, touch support, and the user-agent string.

4.1 Identifying Fingerprinting Attempts

Identifying fingerprinting scripts can be challenging due to obfuscation and potential false positives. For example, scripts may use Canvas API for both drawing images or fingerprinting the user's browser [91]. We draw on well-established methods to distinguish between fingerprinting and benign use of fingerprinting vectors [65, 42]. Specifically, we focused on Canvas, WebRTC, Canvas Font, or AudioContext fingerprinting and detected them using the heuristics presented by Iqbal et al. [65]. To detect fingerprinting attempts, we modified the `getter` and `setter` methods of

the several Web APIs such as `CanvasRenderingContext2D.fillText` and `HTMLCanvasElement.toDataURL` to intercept potentially fingerprinting-related function calls and property accesses. Although TRC can intercept JavaScript API calls, we implemented a separate collector (`FingerprintCollector`) to avoid a known issue that prevented TRC from intercepting early function calls [39]. `FingerprintCollector` simply injects the instrumentation script into each page and its descendant frames as soon as they are created. We verified that our collector captures calls missed by TRC on both our custom-developed fingerprinting test pages and external demo pages such as BrowserLeaks [20].

4.2 Identifying Rracking-related Requests

To identify tracking-related requests, we utilized the uBlock Origin Core [56] npm package, which mimics the tracking protection of the widely-used uBlock Origin extension [125]. We used uBlock Origin's default filter lists, which include EasyList and EasyPrivacy, among others [59]. To accurately detect tracking-related requests, we provided uBlock Origin Core with the request's resource type (e.g., image or script) and the page and request URL derived from the HTTP request/response details recorded by the crawler. Then, we mapped the tracker domains to their owner entities (i.e., organizations/companies) using DuckDuckGo's entity map [43]. Using entities to quantify tracker prevalence reduces overcounting as multiple domains can be owned by the same business (e.g., googleanalytics.com and doubleclick.net are both owned by Google).

4.3 Discovering Inner Pages

We refrained from only focusing on homepages as prior work found that websites' inner pages tend to contain more trackers and cookies [126, 13]. Thus, we also gathered five inner links from each of the 2,004 websites by conducting four separate link-collection crawls (desktop and mobile crawls from Frankfurt and NYC). We preferred to crawl sites from two vantage points to reduce the time and effort required for the link collection process. We excluded external domain links and documents such as PDFs and images, and we preferred links near the viewport's center to avoid unrelated links from footers or less visible page areas. After gathering these inner links, we merged them with the homepage URLs to form the final set.

4.4 Interacting with Consent Dialogs

Since the GDPR came into effect, websites typically show consent dialogs when viewed from the EU and to some extent even from the US [98]. Ignoring these dialogs may lead to *undermeasurement* of the tracking and advertising practices. We decided to provide affirmative consent to all data processing request options (accept all) in our crawls to measure the full extent of advertisements and tracking a child could experience. To handle consent dialogs in an accurate and automated manner, we used DuckDuckGo's `autoconsent` library [36], which comes bundled with TRC [38]. Autoconsent incorporates rules from Consent-O-Matic [16, 93], allowing programmatic interactions with the CMPs.

4.5 Video Screen Captures

To detect ads and scrape their disclosures, our crawler performed a series of interactions with the page, including dismissing popup dialogs, interacting with CMPs, and clicking on visible ad disclosure links(§4.6). To monitor these interactions, we added a video capture functionality to the crawler (`VideoCollector`). We used videos of the crawler's interactions to troubleshoot potential issues with the crawler process as well as to label animated ads and other crawl artifacts manually.

4.6 Identifying Ads and Ad Targeting Criteria

The `AdCollector` performed three main functions: 1) detecting ads, 2) scraping ads —including its screenshots, links, iframes, scripts, videos, and background images, and 3) detecting and scraping ad disclosure pages to determine whether an ad is targeted or not.

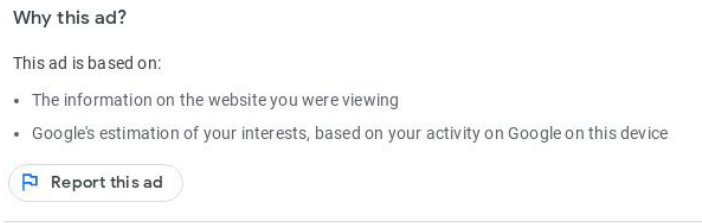
Detecting ads: To detect ads, we built on Zeng et al.'s [138] approach to use EasyList's rules [121]. EasyList rules are commonly employed by popular adblockers to block or hide ads. For each detected ad element, the crawler recorded a set of attributes, including its position on the page, its dimensions, class, ID, and name, in addition to the complete HTML source and a screenshot. If the ad element contained any child elements, which was mostly true, the crawler recursively recorded their details, including all links, images, video, script, and iframe elements. Small elements ($< 30px$ in either dimension) and elements lacking any link, image, background image, or video were excluded.

The crawler not only took screenshots of each ad but also downloaded image and video elements that were descendants of the ad element. These media

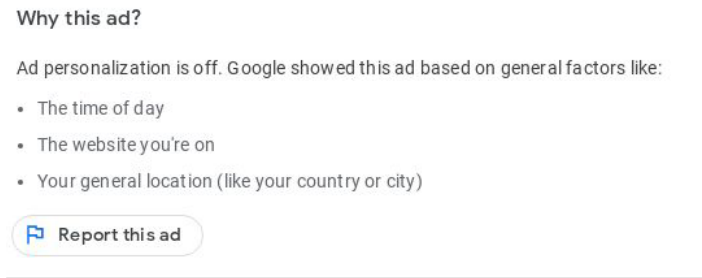
were utilized in the ML-based ad content analysis pipeline 4.6 alongside the ad screenshots. The crawler sent a single HTTP request during the page visit with the appropriate HTTP headers—such as the HTTP Referer [sic.] set to the current address-bar URL—when downloading these files. Finally, the crawler saved data-URL images found within the ad's subtree. Bin Musa and Nithyanand [92] also utilized EasyList's rules for ad identification in their study on tracker-advertiser relationships, but their implementation differs from ours. While they focus on detecting image-containing HTTP responses using the EasyList filter set, we query the DOM to detect ad elements, such as div elements, and their relevant descendant elements, such as images, iframes, links (`a`), and videos. Operating at the DOM level also allows us to detect and scrape ad disclosure pages to detect targeted ads. To verify how accurately our crawler detects ads, we performed a sanity check on a random sample of 105 ads (15 ads from each crawl). The crawler correctly detected ads in 85% of cases, misidentified non-ads in 7.5%, and captured blank or empty ads in 7.5%. Some ad screenshots also included multiple (2.8%) or only part (4.5%) of the ads. However, the overall accuracy and quality of our ads appear to be higher than prior work by Zeng et al. [137], which reported 34% unrendered (blank/unreadable) ads. We attribute this difference in data quality to two potential reasons. First, we use a more realistic crawler equipped with anti-bot measures; and second, unlike Zeng et al., we opted not to click the ads—which may trigger more stringent anti-bot, anti-fraud protections that prevent the delivery or rendering of the ads. Further, to evaluate whether our EasyList-based detector missed any ads, we manually reviewed 50 random pages where no ads were detected by the crawler. Our review did not reveal any false negatives, suggesting that our ad detection was robust. We also verified the accuracy of the ad images separately downloaded by the crawler, finding all of them to be present in the ads shown on the page.

Determining targeting criteria: To measure the prevalence of targeted advertisements at scale, we automated the process of scraping ad disclosure (e.g., "Why this ad") pages. While the content of ad disclosure pages may vary by ad platform, they generally explain in broad terms why a specific advertisement was shown to a user. The reasons may include, for instance, *Google's estimation of your interests* or *Websites you've visited*. The disclosure pages may also contain information about the website and the advertiser, and whether ad targeting is turned off for the website or a specific ad. Two example disclosure pages for a targeted and non-targeted ad are shown in Figure 3.

Ad disclosure pages are reachable by clicking the AdChoices icon and the "Why this ad" button for Google ads [4] and other ad providers. Initially, we attempted to detect the ad disclosure links using fuzzy image matching based on the AdChoices icon. However, we found that the icon's shape and



(a) Targeted ad



(b) Non-targeted ad

Figure 3: Google's ad disclosure pages indicating whether an ad is targeted or not. The top figure belongs to a targeted ad (indicated by *Google's estimation of your interests*), while the bottom one is for a non-targeted ad (indicated by *Ad personalization is turned off.*)

visibility substantially vary across different ad vendors, and sometimes the icon can be hidden, making it unclickable. We chose to identify ad disclosure links through their URLs, focusing on a fixed set of providers we could detect reliably and deterministically. By analyzing ad disclosure pages in our pilot crawls, we compiled a list of hostnames (i.e., `adssettings.google.com`, `privacy.us.criteo.com` and `privacy.eu.criteo.com`) that appear in the ad disclosure links, and explain whether an ad is targeted or not. We limited our investigation to ad disclosure pages from these two providers because other providers we encountered in our pilot crawls did not offer useful information about the targeting criteria of the ads.

Once the crawler detects and clicks on the AdChoices link, the ad disclosure page opens in a new tab. We intercepted this new tab, stored its URL, screenshot and text contents (via `document.innerText`) for analysis. The scraped text contents are then used to detect whether ad targeting is enabled or not. Specifically, we searched in the ad disclosure texts for specific disclosure

statements indicating whether and how an ad was targeted. The disclosure statements include, for instance, *Google's estimation of your interests* (targeted), *Websites you've visited* (targeted) and *Ad personalization is turned off* (non-targeted). If one or more statements indicating targeted ads occur in an ad disclosure text, we label the ad as targeted. Otherwise, we label the ad as non-targeted. Note that we count behavioral or retargeted ads also in the targeted category. We compiled a list of 18 statements (Appendix 8) incrementally, using over 40K ad disclosure texts extracted during the crawls. We ensured that all ad disclosures contain at least one of these statements, to make sure our analysis is exhaustive.

Interacting with the page and ads: Upon loading a page, our crawler waited for 2.5 seconds and dismissed any popup dialogs using heuristics from prior work [84]. We dismissed these dialogs to prevent them from blocking our crawler's interactions with the webpage. The crawler then waited for another second before scrolling through the page in 10 steps, taking strides of about 500–600 pixels each interlaced with a random delay of 500–600 milliseconds. Finally, after waiting for another second, it scrolled up to the beginning of the page using the same scrolling behavior. We engineered this up-and-down scrolling behavior to allow the webpage to load any ad slots that are lazily loaded as the user scrolls the page below the landing fold.

The crawler then identified all ads on the page. It set the border color of each ad to red to visually mark the ads for manual review. The crawler then took a screenshot of the entire page and then scraped each ad in a top-down fashion. To ensure that an advertisement is fully seen, it scrolled down to each ad before taking its screenshot. Finally, the crawler detected ad disclosure links and clicked each one individually to capture all ad disclosure texts and screenshots. We limited the number of scraped ads per page visit to ten, which limits over-representation by a few websites with many ads.

Analyzing advertisement content: We identified and measured four kinds of ads in our corpus: weight loss ads, mental health ads, dating services ads, and ads that contain clickbait racy content. While our dataset of ads can be used to perform fuller content analysis, we focused on these four categories since prior work [54, 23] and regulatory reports [25] have argued that these can be especially harmful to children. In fact, many ad networks' moderation policies [5, 102] explicitly restrict these categories of ads from appearing on children's websites. We note that the categories we focused on are not exhaustive, and our ads analysis is exploratory, serving as a preliminary investigation into this critical problem.

An overview of the ad content analysis pipeline is shown in Figure 4. To identify ads containing click-bait racy content, we employed Google Cloud Vision API's

SafeSearch Detection [63], which is a service that uses deep learning to analyze images and identify potentially unsafe content. It evaluates images against categories such as adult, violent, racy, and medical content and returns likelihood scores for each category, ranging from 'VERY_UNLIKELY' to 'VERY_LIKELY.' Upon manually evaluating the output generated by the algorithm, we focused on the 'racy' category with a likelihood of 'VERY_LIKELY'. We also tested Microsoft's Adult Content Detection [89], part of Azure Cognitive Services, to identify racy images. However, due to more false positives compared to Google Cloud Vision API, we chose the latter for our study.

We used the Google Cloud Vision API to extract text from ad images following a similar approach to Bin Musa and Nithyanand [92]. The text in each image was extracted using the Optical Character Recognition (OCR) feature of the API, specifically by employing the `fullTextAnnotation` attribute of the API response. This allowed us to extract text data at different levels, such as page, paragraph, and word. We opted to use the paragraph level since it gives the best separation in ads promoting multiple unrelated products. Despite their name, paragraphs returned by the API were relatively short and akin to sentences (21 characters, on average).

We then employed semantic similarity to identify the most similar ad texts (paragraphs) corresponding to a given search query, which in our case were "weight loss", "mental health", and "dating". This approach is versatile and can be used to retrieve ads related to any arbitrary words or phrases. To compute the embeddings of the queries and ad paragraphs, we used the "paraphrase-multilingual-mpnet-base-v2" model, the distilled multilingual model we used to classify web pages (§3.2). To find the most similar results, we calculated the cosine similarity between the embeddings of the search query and each ad paragraph and sorted them accordingly. Next, we manually reviewed the 100 most similar distinct paragraphs and their associated images, including ad screenshots or background ad images, to identify those that pertained to the three categories of interest. We also experimented with BERTopic [57] to create topic models and searched for clusters similar to our chosen categories. While this resulted in well-grouped texts, it required manual verification of numerous (several thousand) clusters. Sorting based on semantic similarity proved to be faster, more flexible, and easier to implement and evaluate, making it the preferred approach for manual reviewing.

4.7 List of Crawls

The main dataset used in our study consists of seven crawls (Table 2), all of which were run in April 2023 using cloud-based servers on Digital Ocean. The

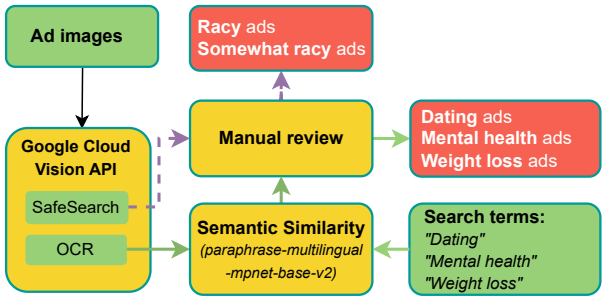


Figure 4: Overview of the ad content analysis pipeline.

crawls include five desktop and two mobile crawls from five and two vantage points, respectively. We limited the mobile browser crawls to two vantage points because we do not focus on mobile-desktop comparison, which we leave to future work. The vantage points used for crawls consisted of Frankfurt, Amsterdam, London, San Francisco, and New York City (NYC), to capture ads and tracking from different jurisdictions. Crawls were run in parallel using separate servers with moderate resources (8 vCPU cores, 16GB RAM). Each crawl took between 13 and 32 hours to complete. A clean browser profile is used for each page visit to prevent ads targeted to our browsing history. During each crawl, we visited both landing and inner pages, following the process described in §4.3. When applicable, we accepted all personal data processing on consent (cookie) dialogs. The order of visited pages was randomized within each vantage point. Note that the San Francisco crawl used inner links extracted from the NYC crawl, while the London and Amsterdam crawl used inner links from the Frankfurt crawl. While this constraint did not appear to impact the success rate of visits across these vantage points, future research could explore identifying inner pages during the crawling process.

5. Measurement Results

Table 2 summarizes the overall statistics for measurement crawls. A total of 71,567 pages were loaded successfully across all crawls. The success rate of our crawler was over 93%, according to the successful visit criteria we developed and applied (Appendix 8). For simplicity, certain comparative results presented below are based on desktop crawls from NYC and Frankfurt, representing one location each in the US and the EU.

Table 2: Crawl statistics based on different vantage points. *: Avg./Sum of all sites visited across all crawls.

Form factor	Vantage point	Successfully loaded pages	Successful crawling rate
Desk.	NYC	10,310	95%
	SF	10,301	95%
	LON	10,270	95%
	FRA	10,221	95%
	AMS	10,014	93%
Mobile	NYC	10,168	94%
	FRA	10,283	96%
Avg./Sum		71,567	95%

5.1 Ad Targeting and Content Analysis

Our crawler scraped 70,303 ads from 804 of the 2,004 distinct websites across seven crawls. An average of 36% of the pages contained one or more ads, and we detected targeted ads on 27% of the pages we crawled. The crawler scraped 10,839 and 9,447 ads on average in the crawls from the US and Europe, respectively.

Over 70% of ads with disclosures are targeted in nature

Our crawler captured a total of 40,281 ad disclosure pages, which we used to determine the advertiser's identity and whether ad targeting is enabled or not. There are fewer disclosure pages than ads due to ads without disclosure links and failures in detecting or opening those links. In fact, we only consider ad disclosures from two ad providers: Google (97.8%) and Criteo (2.2%), since ad disclosure pages of other providers did not reveal the targeted status of the ad or the advertisers' identity. Limiting our analysis to 40,281 ads with disclosure pages, we found that targeting was enabled for 73% of the ads with disclosures. Comparing across different privacy jurisdictions, we find 68% of the ads on average were targeted in the EU crawls, compared to 76% in the UK and the US crawls. Comparing the crawls from the two US cities (SF & NYC), we find that 67% of the ads were targeted in the SF desktop crawl, compared to 79% and 82% in the NYC-based desktop and mobile crawls, respectively. Although these variations might be attributed to stricter privacy regulations like CCPA and GDPR, our available data and methods do not permit us to make this attribution. Comparing the Tranco ranks of the 689 websites that contain at least one targeted ad to 59 websites that only contain non-targeted ads, we find

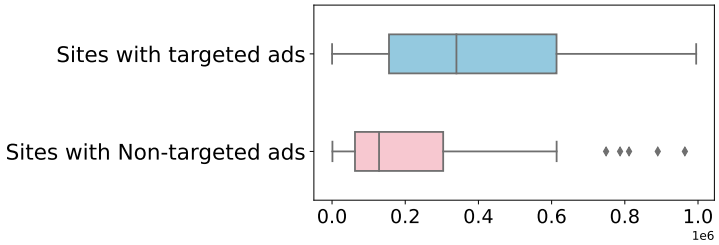


Figure 5: Tranco rank (x-axis) distribution of sites that use targeted vs. non-targeted ads. Popular websites (below) appear to be more prone to disabling ad targeting.

Table 3: Number of visits and scraped ads, along with percentages of ads/targeted ads per crawl. *: Percentage of targeted ads is only based on ads with disclosures. In the rightmost two columns, we include a site if we scraped at least one ad/targeted ad from one of its pages.

Form factor	Vantage point	# ads	% sites with ads	% sites with targeted ads	% targeted ads
Desk.	NYC	11,288	38%	30%	79%
	SF	10,950	38%	28%	67%
	LON	9,702	36%	27%	76%
	FRA	9,700	36%	26%	68%
	AMS	9,250	35%	26%	67%
Mobile	NYC	10,278	36%	29%	82%
	FRA	9,135	33%	26%	70%
Avg./Sum		70,303	36%	27%	73%

a tendency for popular websites to disable ad targeting (Figure 5). Sites with targeted ads had a median rank of $\sim 340K$, while those with only non-targeted ads had a median rank of $\sim 128K$. Note that we only include 40,281 ads for which we can determine the targeted status in this analysis.

Ads can be targeted from anywhere

The "About the advertiser" section in Google's ad disclosures shows the name and location (country) of the advertisers. This information is only available

in 70% of the ad disclosures in our dataset. Extracting these fields from the ad disclosure texts, we identified 1,685 distinct advertisers from 81 different countries. Advertisers with the most ads in our data are displayed in Table 4. We note that due to the transient, targeted and localized nature of ad campaigns, the list in Table 4 may not represent the most common advertisers on child-directed websites in general. Further, in certain cases (e.g., Gloworld LLC and Marketism), an advertising or marketing agency is listed on the ad disclosure page instead of the company offering the advertised products or services.

The top ten advertisers are located in seven different countries and three continents. We observed that many of those advertisers are located far from our crawl vantage points, thus indicating that children visiting websites in our list can be targeted with ads from anywhere in the world. Reviewing a sample of 100 ads from each advertiser, we marked in the rightmost column their predominant ad theme. Five of the ten advertisers display ads for search results about various products—such as depression tests, belly fat reduction, senior meals, and electronic payments—on lesser-known search engines such as IngoSearch [64]. Ads from Betterme [19], a "behavioral healthcare app" with more than 100M installations, featured plans for weight loss, muscle gain, and intermittent fasting (e.g., Figure 1 ⓗ).[6] Brain Metrics Initiative displays ads for IQ tests, an example for which is given in Figure 1 ⓒ. Alibaba Hong Kong, on the other hand, displays ads featuring racy and disturbing images of products sold on alibaba.com. For instance, the ad on the top left (ⓐ) in Figure 1 features recurring images in Alibaba ads: a naked baby model (leftmost), rabbit meat (rightmost), and a semi-transparent underwear ad in the middle. We investigate similar racy clickbait ads and other improper ads in the following subsection.

Improper ads on child-directed sites

In total, our crawler collected 199,935 screenshots and images from the 70,303 scraped ads. After deduplicating the images, we queried the Cloud Vision API to obtain the category and OCR texts of the resulting 98,264 distinct images. We manually reviewed 741 images classified as 'VERY_LIKELY' racy by the API. Separately, we reviewed 1,136 ad images with OCR text semantically most similar to our search terms (mental health, dating, and weight loss). Due to study limitations, we only examined the ads related to the top 100 distinct texts for each term. Since each distinct text may appear in multiple ads differently, we labeled the images separately and used videos captured by the crawler when the ad was animated or the ad screenshot was obscured. Table 5 shows the

[6]We note that Better.me's data sharing practices with third parties were investigated by Privacy International, but the company reportedly took corrective action[11].

Table 4: Top ten advertisers by the number of ads across all crawls.

Advertiser	**Location**	**# ads**	**% targeted**	**Type of ads**
Vinden.nl B.V.	Netherlands	4,707	86%	Search results
EXPLORADS	Cyprus	3,265	73%	Search results
All Response	UK	2,453	68%	Search results
Gloworld LLC	USA	2,365	55%	Online learning
Amomama M.	Cyprus	921	72%	Workout muscle gain, weight loss
Media Quest	UAE	910	79%	Search results
Brain Metrics I.	Cyprus	814	50%	IQ tests
BetterMe	Cyprus	731	85%	Weight loss
Marketism	Israel	645	49%	Search results
Alibaba.com HK	Hong Kong	541	86%	Products sold on Alibaba.com

Table 5: Number of improper ads identified for each crawl.

Form factor	**Vantage point**	**Dating**	**Mental health**	**Weight loss**	**Racy**	**Somewhat racy**	**Total**
Desk.	NYC	4	21	16	21	26	88
	SF	7	9	15	6	25	62
	LON	10	17	48	12	31	118
	FRA	1	0	48	19	25	93
	AMS	8	4	82	10	33	137
Mobile	NYC	22	25	113	98	17	275
	FRA	18	5	190	11	6	230
Total		70	81	512	177	163	1003

number of improper ads identified in each crawl, amounting to 1,003 across 311 distinct websites. A notable finding is the higher prevalence of such ads on mobile devices compared to desktops in general.

Racy images. We found 177 racy ads and 163 somewhat racy ads, considered edge cases due to their potential inappropriateness for child-directed websites. These ads were identified across 80 distinct websites mostly ranked within the top one million according to the Tranco list, with a median rank of 426K. Figure 1 ⓐ, ⓖ, ⓚ are examples of some of these ads. Notably, the majority of the racy ads were encountered in mobile crawls; especially within the NYC crawl (98/177). Out of 177 racy ads, only 38 had disclosure pages. Among these, targeting was enabled for 35 ads.

Mental health. By manually labeling 236 ad images, we identified 81 ads offering mental health services on 48 distinct websites. Examples of ads in this category contained "take a depression test" (Figure 1 ⓕ), "online psychiatrists," "how to get over depression," and a "mental health chatbot which helps people with depression." We excluded false positives that were not mental health service offerings, such as ads for "mental health counselor salaries," "online psychology courses," and "psychology books."

Dating. Manually labeling 231 ad images, we identified 70 dating platform ads on 48 distinct websites, most of which targeted mobile users. The ads promoted dating platforms such as "dating.com," and "Live Me," a live streaming app with ads featuring suggestive imagery (Figure 1, ⓙ, ⓚ). Another ad for DateMyAge.com featured a call to "[m]eet your mature singles" (ⓔ). False positives removed during the manual labeling of this category included ads for customer relationship tools, romantic holiday tours, and online appointment services.

Weight loss. We identified 512 weight loss-related ads (plans, apps, products) on 170 distinct websites by labeling 669 ad images. Notably, there was a higher number of weight loss ads on mobile devices, indicating campaigns targeting mobile users. Examples of text featured in these ads included "intermittent fasting for weight loss," "keto weight loss plan," and "eating plan to lose weight" (Figure 1 ⓗ).

In Figure 1, we provide additional examples of advertisements that are likely not suitable for children. Examples of these included an ad for a test called "Am I Gay Test" ⓓ, for a sex toy ⓘ and a sex toy shop ⓑ[7] featuring an image of ice cream that could be appealing to children, and ads featuring clickbait and sexually suggestive images. The ads were found on websites related to K-12 e-learning, kids games, coloring books and worksheets, among others. The ads

[7]Reportedly Germany's largest online adult retailer [41]

Table 6: Average number of third-party and tracker domains, and the prevalence of tracking and fingerprinting on child-directed websites from five vantage points.

Form factor	Vantage point	3rd-Party domains	Tracker domains	Tracker entities	% sites with 3rd Parties	% sites with trackers	% site with FP
Desk.	NYC	31.6	23.4	20.0	95%	90%	9%
	SF	29.3	21.3	17.8	95%	91%	9%
	LON	21.3	14.3	10.6	96%	91%	7%
	FRA	23.2	15.6	11.7	95%	90%	10%
	AMS	21.4	14.3	10.6	93%	89%	7%
Mobile	NYC	29.8	21.8	18.4	95%	91%	9%
	FRA	22.6	15.2	11.5	95%	90%	11%

in Figure 1 do not necessarily fit our four investigated categories but showcase the diversity of improper ads.

Malicious ad links. Finally, we present an exploratory analysis of whether ads on child-directed websites link to malicious pages. We submitted a sample of links extracted from the ad elements to the VirusTotal API in August 2023. Specifically, we removed links with duplicate hostnames, and for Google ads, we extracted a direct link to the ad landing page using the 'adurl' parameter [17]. While the overwhelming majority of the links were classified as benign, 149 of the nearly 3,940 scanned links were flagged as malicious or phishing by at least one scan engine. Notably, the word "taboola" was mentioned in 78 of the 149 detected links as a URL parameter that seems to indicate the ad network (`network=taboola`).

5.2 Tracking and Fingerprinting Analysis

Table 6 shows the prevalence of third-party trackers detected across different crawls. We find that around 90% of the websites have at least one tracker domain, and over 93% embed at least one third-party domain.

Third-party trackers. The average number of tracker domains per site differs significantly, e.g., 15.6 and 23.4 in Frankfurt and NYC crawls, respectively, while the median is 15 and 16, respectively. The difference in averages can be attributed to websites with the high number of trackers in the NYC crawl. This explanation is in line with the results displayed in Table 8, which shows the top five websites with the most trackers in Frankfurt and NYC crawls. Most of these websites are among the top one million, which means they likely

Table 7: Prevalence of tracker entities in terms of number of distinct websites in Frankfurt and NYC desktop crawls.

FRA		NYC	
Entity	**# Sites**	**Entity**	**# Sites**
Google	1,702	Google	1,718
Facebook	458	Microsoft	549
Index Exchange	424	Adobe	543
Xandr	416	Xandr	516
Adform	412	The Trade Desk	501
The Trade Desk	390	Index Exchange	495
OpenX	378	IPONWEB	467
Adobe	366	Facebook	456
Quantcast	361	Magnite	446
PubMatic	359	OpenX	426

receive substantial traffic. Notably, all of these sites displayed ads that were targeted. The numbers shown in the table - number of trackers, requests, and cookies - reflect averages across the web pages. In the NYC crawl, visiting `mathfunworksheets.com` triggered a total of 1,547 requests involving 161 unique third-party tracker entities (i.e., organizations/companies). Another website, `woojr.com` found to contain 148 distinct third-party tracker entities when visited from NYC. This website includes resources for children's activities and educational materials, including printable worksheets and fun activity pages. When visited from Frankfurt, `www.wowescape.com`, a website offering various games for children and teenagers, triggered requests to 95 distinct third-party tracker entities.

Most prevalent trackers. Table 7 shows the tracker entities with the most prevalence in Frankfurt and NYC desktop crawls. We found a tracking-related request to Google domains, including its analytics, advertising and tag management scripts on ∼84% of the 2,004 child-directed websites in both crawls. Facebook is the second most prevalent entity in the Frankfurt crawl mostly due to Facebook Pixel (on 427 websites), which facilitates ad retargeting and conversion measurement, among others [95]. Largely thanks to Linked Insight Tag (`px.ads.linkedin.com`, 466 websites), Microsoft is the second most prevalent entity in the NYC crawl. Linked Insight Tag serves multiple purposes, including retargeting, conversion measurement, and providing demographic insights about website visitors [80].

Regional differences. To explore the differences in tracker entities across vantage points, we compared the tracker entities from Frankfurt and NYC desktop crawls. Despite a considerable overlap among the detected tracker

Table 8: Websites with the most distinct tracker entities. The table shows the websites' distinct third-party tracker entities, the number of requests, cookies, and Tranco rank.

Loc.	Website	# Trackers	# Requests	# Cookies	Rank
	mathfunworksheets.com	161	1,547	395	669K
	woojr.com	148	2,181	391	83K
NYC	innerchildfun.com	139	1,235	336	308K
	kidzfeed.com	138	1,050	272	797K
	thecolor.com	138	1,068	260	192K
	wowescape.com	95	392	55	258K
	webgames.io	94	564	92	155K
FRA	coloriages-pour-enfants.net	90	401	66	919K
	theschoolrun.com	87	417	91	760K
	testsworld.net	86	478	138	-

entities (Jaccard index=0.85), we also identified variations. Specifically, our investigation unveiled 47 tracker entities exclusive to the Frankfurt crawl and 118 tracker entities that were only found in the NYC crawl. For instance, tracking related requests to *advanced STORE* [6] (`ad4m.at` & `ad4mat.net`, 236 websites) exclusively appear in the crawl from Frankfurt, whereas *Throtle*, a company that provides an identity graph to marketers and advertisers, only appears on 171 websites in the NYC crawl [112]. Furthermore, we find that the majority of the websites in both Frankfurt and NYC crawls (âĹij70% and âĹij72%, respectively) contain third-party trackers that set at least one cookie with the `SameSite=None` attribute and a lifespan of over three months. Primarily through `doubleclick.net` domain, Google set these cookies on over 51% of the websites. While identifying the individual purposes of these cookies is out of scope, this combination of cookie attributes (esp. setting `SameSite=None`) makes it possible to track users across websites.

Sites with and without ads. As part of our investigation, we conducted an additional analysis to compare how the number of third parties and trackers change between websites with and without ads. Figure 6 shows that websites with ads tend to have substantially more third-party and tracker domains. More specifically, the figure shows websites with ads tend to contain two to four times more third-party and tracker domains.

Browser Fingerprinting. We now discuss our findings on fingerprinting scripts on child-directed websites. Table 6 shows that we detect fingerprinting scripts on 176 (9%) and 218 (10%) websites in Frankfurt and NYC crawls, respectively. The overall prevalence of fingerprinting aligns with the recent research by Iqbal et al., which finds fingerprinting on 10.18% of the top-100K websites [65]. One of the most prevalent fingerprinters in both crawls is an

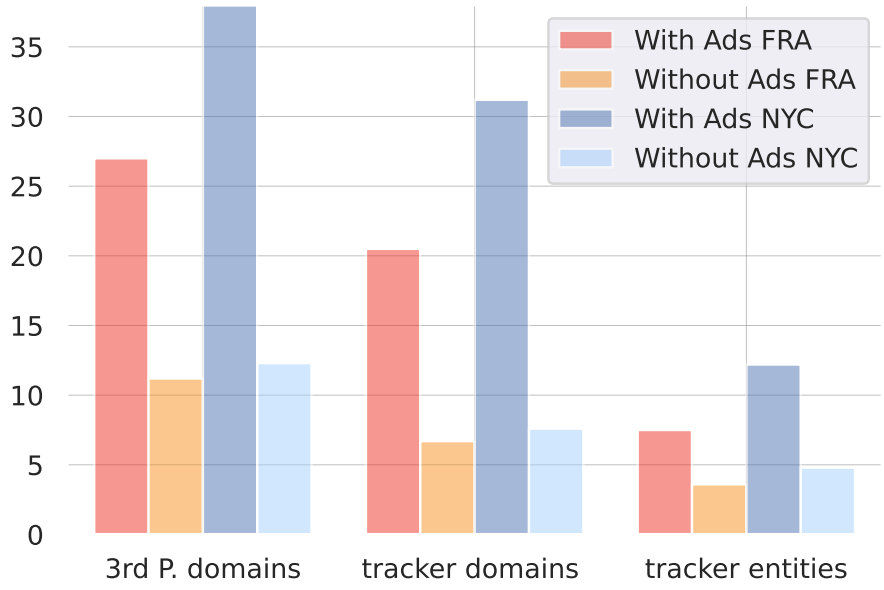


Figure 6: Comparison of the average number of third-party and tracker domains/entities on websites containing ads vs. not containing ads.

online payment company (Stripe; 66, 67 sites on Frankfurt and NYC crawls, respectively). According to their help pages [118] Stripe primarily employs fingerprinting for fraud prevention purposes. Webgains (82 sites in the Frankfurt crawl), an affiliate marketing company, also mentions fingerprinting in their Data Processing Agreement with Merchants [132], but without specifying its purpose. The most commonly used fingerprinting method is Canvas fingerprinting, present on about 208 sites in the Frankfurt crawl and about 172 sites in the NYC crawl.

We found one or more trackers to be present on more than 90% of mobile websites (Table 6), which is similar to our finding for the desktop websites. NYC and Frankfurt crawls differ slightly in the number of ads: we scraped 10,278 ads in the NYC crawl and only 9,135 in the Frankfurt crawl—the latter is the crawl with the least amount of ads. Slightly more (29 versus 26%) websites in the NYC mobile crawl have targeted ads; and NYC mobile crawl has the highest proportion of targeted ads (82%) across all crawls. We also discovered that improper ads, particularly racy and weight loss ads, were more prevalent on mobile devices compared to desktops.

6. Discussion

Our research paints a troubling picture of tracking and inappropriate advertising practices on child-directed websites. Advertisements featuring sexually suggestive imagery and ads about weight loss, dating, and mental health may pose potential risks to children's emotional and psychological welfare. We discuss the legal implications, ethical considerations and limitations of our study below.

6.1 Legal Implications

In this section, we discuss what the law says about tracking and advertising practices uncovered in our research. We focus on the EU General Data Protection Regulation (GDPR) and the US Children's Online Privacy Protection Act (COPPA)[8].

The GDPR and the ePrivacy Directive. Under the GDPR, companies are only allowed to process personal data if they have a 'legal basis' for such processing. The GDPR provides six possible legal bases (article 6 GDPR). However, generally, the data subject's consent is the only possible legal basis for online tracking and behavioral (targeted) advertising [49]. Moreover, the ePrivacy Directive [44] requires, in short, companies to ask the internet user for consent before they use tracking cookies or similar tracking technologies (Article 5(3)).

The GDPR's requirements for valid consent are strict. Consent is only valid if it is really voluntary ('freely given'), and 'specific' and 'informed'. The data controllers (the website owner and the companies involved in tracking and targeted advertising) must 'be able to demonstrate that the data subject has consented to process of his or her personal data' (Article 7(1) GDPR). The GDPR's requirements for valid consent also apply to consent (for cookies etc.) as prescribed by the ePrivacy Directive. The GDPR has specific rules for consent by children. Roughly summarized, children cannot give valid consent; the parent should give consent instead (Article 8 GDPR). EU member states have set different minimum consent ages, ranging from 13 to 16 years [90]. Hence, only parental consent can legitimize tracking on a children's website. Observe that a parent clicking a consent dialog (as done by our crawler) does not constitute

[8]We do not analyze whether specific companies breach the law. For such an analysis, each case would have to be examined separately, considering all the circumstances of that specific case. Rather, we discuss legal requirements in general terms.

parental consent under GDPR. Even in low-risk cases, verification of parental responsibility via email may be necessary [48].

The EU Digital Services Act. The rules for tracking and targeting children will become stricter in the EU. From 17 February 2024 on, the EU Digital Services Act [33] applies. Article 28 says, roughly summarized, that online platforms must not use behavioral advertising 'when they are aware with reasonable certainty that the recipient of the service is a minor' [33]. This prohibition cannot be overridden with the consent of the child or the parent. The DSA also requires "very large online platforms" [47] (with more than 45 million users in the EU) to publish the advertisements that it presented to users in an online repository, together with information about, for instance, the targeting criteria (Article 33, 39 DSA). The methods that we used in this paper could be used to check the completeness and accuracy of data published in those repositories.

COPPA. COPPA regulates companies offering a website or online service directed to children under the age of 13. Specifically, COPPA applies to companies using children's "personal information", which includes "persistent identifiers such as cookies and device fingerprints" (COPPA §312.2) [51]. The website owner is responsible for data collection by third parties through its site. Such third parties must also comply with COPPA. Companies based outside the US must also comply with COPPA if their services are directed to children in the US [51].

Our results showed that 27% of the child-directed websites use targeted advertising. Under COPPA, data collection for targeted advertising on these websites is only allowed after getting parents' Verifiable Parental Consent (VPC). VPC entails utilizing stringent verification methods, such as credit card verification, face recognition, or government ID checks [52]. This makes VPC much more complex than simply clicking an accept button on a dialog. We note that our crawler cannot simply give VPC.

6.2 Research Ethics

Our crawler visited over 166K pages and it triggered many ad impressions that could be viewed by a real visitor (likely a child). Given the huge scale of the digital ad market (projected to reach US$700bn in 2023 [55]) we believe these ad impressions are a negligible cost for raising the transparency around tracking and ads targeted to children. Furthermore, we took several measures to limit our footprint on the crawled websites. For instance, we only crawled five inner pages from each site in a crawl, and we randomly shuffled the target URLs to avoid concurrently visiting the inner pages of a website. We also took

appropriate measures to ensure that no harm was done to collaborators involved in the project, especially when dealing with explicitly graphic images.

Disclosures and outreach In July 2023, we reached out to five companies that we found to serve racy ads. One company invited us to a call and explained the likely reason for showing inappropriate ads (websites failing to label themselves as child-directed and a false negative in the ad company's automated child-directed site detection tool). They added the sites in question to the child-directed list and pledged further investigation. Another thanked us and began an internal review. The third redirected our query to the relevant department. Moreover, we disclosed 34 racy ads to Google by manually visiting the ad disclosure URLs of each racy (Google) ad; and using the *Report this ad* button. Note that, to identify the ad vendors involved in serving the ad, we used a combination of ad images, and src/href attributes of the ads' descendant iframe, image and link elements (§4.6).

In addition, we shared our preliminary results with a European data protection agency (DPA), and a consumer protection agency. Both showed interest; the DPA asked if there were any websites from their country containing improper ads. The consumer protection agency invited us to present our work. We also shared our findings with civil society and industry organizations including the 5Rights Foundation [1]. We plan to further share our study with regulators and other relevant stakeholders.

While using the VirusTotal API, we found and reported three porn websites miscategorized as kids-related to the respective third-party categorization service. Although no response was received, the categories were later rectified.

6.3 Limitations

Our study predominantly covers websites targeting younger children, as we define 'children' as under 13, aligning with both US and EU regulations. While our classifier detected child-directed sites in 48 languages, it may have a potential bias towards English content, which is overrepresented in the training data. The classifier may prefer sites with descriptive titles and content or may carry biases related to website age, design, or accessibility. Moreover, the classifier favors precision over recall to reduce the manual labeling workload.

Our research is the first attempt to build a large list of child-directed websites. Classifying websites as child-directed or not is challenging, primarily due to the existence of gray areas that complicate the labeling process. For instance, many websites offer content that appeals to both children and adults. While we tried to exclude websites that can only be used by teachers or parents, certain

websites included pages such as online exercises, videos or games that can be consumed by children. To validate our list, we performed further analysis on two subsets of websites: two senior researchers relabeled a random 100-website sample; one of the senior researchers relabeled a random subset of 100 websites with targeted (50) and inappropriate ads (50). In both cases, we identified less than 9% of websites that are related to children, but mainly catered to teachers or parents (8.6%, 8.3%, respectively). While this impurity could be avoided with more conservative labeling, our analysis of 200 websites strongly suggests these cases do not skew our primary findings.

While we found fewer targeted ads in the EU than in the US, we cannot directly attribute this to differences in privacy regulation or another specific factor. Failure to detect and interact with consent dialogs may be a confounding factor, among others.

When detecting targeted ads, we only used ad disclosure pages from two providers (Google and Criteo) due to the unavailability of useful ad disclosures from other vendors. Thus, our targeted ad detection method depends on the accuracy, completeness and precision of Google and Criteo's ad disclosures.

Websites may treat cloud-based IP addresses or automated browsers differently [136, 70, 26]. To curb such effects, we used the anti-bot detection features of TRC [124]. Reviewing the screenshots captured during the visits, we observed very few blocked visits.

We conducted four sets of inner link collection crawls: two from NYC and two from Frankfurt, encompassing both desktop and mobile crawls. This constraint does not appear to impact the success rate of visits across these vantage points; nonetheless, future research could explore the possibility of identifying inner pages during the crawling process.

Since we use a fresh profile for each page visit, we may not capture re-targeted or other personalized ads that are only shown to users with a behavioral profile. Future work could extend our method to incorporate personas and warm-up crawls to study such ads. Overall we do not claim that our findings are representative of tracking and advertising practices on child-directed websites. Our focus in this study is not on how ads are targeted, but simply on *whether the targeting is enabled or not.*

7. Conclusion

We presented an empirical study of online tracking and advertisements on over 2,000 child-directed websites. Building a lightweight and versatile ML pipeline

to analyze ad content, we identify hundreds of cases of improper ads, including weight loss and mental health ads, and ads featuring dating services, racy and sexually suggestive imagery. Our study reveals several notable trends: websites featuring advertisements tend to contain two to four times more trackers, mobile websites exhibit a greater prevalence of inappropriate ads, and popular websites are less likely to deploy targeted advertisements. Our findings provide concrete evidence of troublesome practices that are likely illegal, unethical, or simply careless. We call for more research, regulation and enforcement to limit the ongoing violation of children's privacy and well-being.

8. Acknowledgments

Asuman Senol was funded by the Cyber-Defence (CYD) Campus of armasuisse Science and Technology. Veelasha Moonsamy was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2092 CASA - 390781972 and the BMBF project DigiFit - grant number 16KIS1646K. We thank our shepherd and the reviewers for their constructive feedback.

References

[1] 5Rights. `https://5rightsfoundation.com`, November 2023. [Online; accessed 29 Nov. 2023].

[2] Lawrence Abrams. Ransomware access brokers use Google ads to breach your network. *BleepingComputer*, January 2023. `https://www.bleepingcomputer.com/news/security/ransomware-access-brokers-use-google-ads-to-breach-your-network`.

[3] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. The web never forgets: Persistent tracking mechanisms in the wild. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 674–689, 2014.

[4] Transparency and ad disclosures - Google Ads Help. `https://support.google.com/google-ads/answer/9729263?hl=en`, 2023. [Online; accessed 28 Feb. 2023].

[5] Ads & Made for Kids content. `https://support.google.com/adspolicy/answer/9683742?hl=en`, 2023. [Online; accessed 1 Mar. 2023].

[6] Advanced Store. https://www.advanced-store.com/en/. [Accessed: February 24, 2023].

[7] Omer Akgul, Richard Roberts, Moses Namara, Dave Levin, and Michelle L Mazurek. Investigating Influencer VPN Ads on YouTube. In *IEEE Symposium on Security and Privacy (SP)*, pages 876–892, 2022.

[8] Alexa.com (from the Wayback Machine Archive). https://web.archive.org/web/20210327233130/https://www.alexa.com, 2023. [Online; accessed 28 Feb. 2023].

[9] Moondore Ali, Mark Blades, Caroline Oates, and Fran Blumberg. Young children's ability to recognize advertisements in web page designs. *British Journal of Developmental Psychology*, 27(1):71–83, 2009.

[10] Muhammad Ali, Angelica Goetzen, Alan Mislove, Elissa Redmiles, and Piotr Sapiezynski. All things unequal: Measuring disparity of potentially harmful ads on Facebook. In *6th Workshop on Technology and Consumer Protection*, 2022.

[11] An unhealthy diet of targeted ads: an investigation into how the diet industry exploits our data. https://privacyinternational.org/long-read/4603/unhealthy-diet-targeted-ads-investigation-how-diet-industry-exploits-our-data, 2021. [Online; accessed 5 May 2023].

[12] Athanasios Andreou, Giridhari Venkatadri, Oana Goga, Krishna P Gummadi, Patrick Loiseau, and Alan Mislove. Investigating ad transparency mechanisms in social media: A case study of Facebook's explanations. In *NDSS–Network and Distributed System Security Symposium*, pages 1–15, 2018.

[13] Waqar Aqeel, Balakrishnan Chandrasekaran, Anja Feldmann, and Bruce M Maggs. On Landing and Internal Web Pages: The Strange Case of Jekyll and Hyde in Web Performance Measurement. In *Proceedings of the ACM Internet Measurement Conference*, pages 680–695, 2020.

[14] Fahri Aydos, A Murat Özbayoğlu, Yahya Şirin, and M Fatih Demirci. Web page classification with Google Image Search results. *arXiv preprint arXiv:2006.00226*, 2020.

[15] Mika D Ayenson, Dietrich James Wambach, Ashkan Soltani, Nathan Good, and Chris Jay Hoofnagle. Flash cookies and privacy II: Now with HTML5 and ETag respawning. *Available at SSRN 1898390*, 2011.

[16] Rolf Bagge, Célestin Matte, éric Daspet, Kaspar Emanuel, Sam Macbeth, and Steven Roeland. Consent-O-Matic. https://github.com/cavi-au/Consent-O-Matic/, 2019. [Online; accessed 11. Jun. 2023].

[17] Sean Bell. How URL Tracking Systems are Abused for Phishing. *Security Boulevard*, 2020. https://securityboulevard.com/2020/10/how-url-tracking-systems-are-abused-for-phishing.

[18] Best Websites for Kids. https://www.commonsensemedia.org/website-lists, 2023. [Online; accessed 28 Feb. 2023].

[19] BetterMe. https://betterme.world, 2023. [Online; accessed 4 May 2023].

[20] Browserleaks - Check your browser for privacy leaks. https://browserleaks.com, 2023. [Online; accessed 3 Aug. 2023].

[21] Moniek Buijzen. Reducing children's susceptibility to commercials: Mechanisms of factual and evaluative advertising interventions. *Media Psychology*, 9(2):411–430, 2007.

[22] Xiaomei Cai and Xiaoquan Zhao. Online advertising on popular children's websites: Structural features and privacy issues. *Computers in Human Behavior*, 29(4):1510–1518, 2013.

[23] Angela J Campbell. Rethinking children's advertising policies for the digital age. *Loy. Consumer L. Rev.*, 29:1, 2016.

[24] Yinzhi Cao, Song Li, and Erik Wijmans. (cross-) browser fingerprinting via os and hardware level features. In *Proceedings 2017 Network and Distributed System Security Symposium*. Internet Society, 2017.

[25] Age-restricted ads online. https://www.asa.org.uk/static/44dc1935-0766-4378-91171e6954ae560a/Age-restricted-ads-online-targeting-guidance.pdf, 2023. [Online; accessed 28 Feb. 2023].

[26] Darion Cassel, Su-Chin Lin, Alessio Buraggina, William Wang, Andrew Zhang, Lujo Bauer, Hsu-Chun Hsiao, Limin Jia, and Timothy Libert. OmniCrawl: Comprehensive Measurement of Web Tracking With Real Desktop and Mobile Browsers. *Proc. Priv. Enhancing Technol.*, 2022(1):227–252, 2022.

[27] Quan Chen, Panagiotis Ilia, Michalis Polychronakis, and Alexandros Kapravelos. Cookie swap party: Abusing first-party cookies for web tracking. In *Proceedings of the Web Conference*, pages 2117–2129, 2021.

[28] June/July 2022 crawl archive now available – Common Crawl. https://commoncrawl.org/2022/07/june-july-2022-crawl-archive-now-available, 2023. [Online; accessed 28 Feb. 2023].

[29] Contributors. https://support.virustotal.com/hc/en-us/articles/115002146809-Contributors, 2023. [Online; accessed 28 Feb. 2023].

[30] Senators Markey and Cassidy Reintroduce COPPA 2.0, Bipartisan Legislation to Protect Online Privacy of Children and Teens. https://www.markey.senate.gov/news/press-releases/senators-markey-and-cassidy-reintroduce-coppa-20-bipartisan-legislation-to-protect-online-privacy-of-children-and-teens, 2023. [Online; accessed 28 Feb. 2023].

[31] Adding Rank Magnitude to the CrUX Report in BigQuery. https://developers.google.com/web/updates/2021/03/crux-rank-magnitude, 2023. [Online; accessed 22. Feb. 2024].

[32] Savino Dambra, Iskander Sanchez-Rola, Leyla Bilge, and Davide Balzarotti. When Sally Met Trackers: Web Tracking From the Users' Perspective. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 2189–2206, 2022.

[33] Regulation (EU) 2022/2065 of the European Parliament and of the Council of 19 October 2022 on a Single Market for Digital Services and amending Directive 2000/31/EC (Digital Services Act). http://data.europa.eu/eli/reg/2022/2065/oj, 2022. [Online; accessed 28 Feb. 2023].

[34] Digital Services Package: Commission welcomes the adoption by the European Parliament of the EU's new rulebook for digital services. https://ec.europa.eu/commission/presscorner/detail/en/IP_22_4313, 2022. [Online; accessed 28 Feb. 2023].

[35] DNS0 Kids - A childproof version of the Internet. https://www.dns0.eu/kids, 2023. [Online; accessed 4 May 2023].

[36] DuckDuckGo. autoconsent. https://github.com/duckduckgo/autoconsent, 2023. [Online; accessed 28 Feb. 2023].

[37] DuckDuckGo. notABot.js. https://github.com/duckduckgo/tracker-radar-collector/blob/main/helpers/notABot.js, 2023. [Online; accessed 3 Aug. 2023].

[38] Web Tracking Protections | DuckDuckGo Help Pages. https://help.duckduckgo.com/duckduckgo-help-pages/privacy/web-tracking-protections/#cookie-pop-up-management, 2023. [Online; accessed 3 May 2023].

[39] Early browser API accesses and function calls are missed. https://github.com/duckduckgo/tracker-radar-collector/issues/77, 2023. [Online; accessed 2 Aug. 2023].

[40] Peter Eckersley. How unique is your web browser? In *Proceedings of the 10th International Conference on Privacy Enhancing Technologies (PETS)*, pages 1–18, 2010.

[41] Eis.de Is Now Germany's Biggest Adult Retailer. https://www.venus-adult-news.com/en/web-tech/e-commerce/eis-de-is-now-germanys-biggest-adult-retailer, 2023. [Online; accessed 3 Aug. 2023].

[42] Steven Englehardt and Arvind Narayanan. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 1388–1401, 2016.

[43] entity_map.json - DuckDuckGo Tracker Radar. https://github.com/duckduckgo/tracker-radar/blob/main/build-data/generated/entity_map.json, 2023. [Online; accessed 11. Jun. 2023].

[44] ePrivacy Directive. Directive 2002/58/EC of the European Parliament and of the Council. http://data.europa.eu/eli/dir/2002/58/2009-12-19, 2009. [Online; accessed 28 Feb. 2023].

[45] Motahhare Eslami, Sneha R Krishna Kumaran, Christian Sandvig, and Karrie Karahalios. Communicating algorithmic process in online behavioral advertising. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2018.

[46] Estimated cost of digital ad fraud worldwide from 2018 to 2023. https://www.statista.com/statistics/677466/digital-ad-fraud-cost/, 2023. [Online; accessed 28 Feb. 2023].

[47] European Commission. Digital Services Act: Commission designates first set of Very Large Online Platforms and Search Engines. https://ec.europa.eu/commission/presscorner/detail/en/IP_23_2413, 2023.

[48] European Data Protection Board. Guidelines 05/2020 on consent under Regulation 2016/679. https://edpb.europa.eu/sites/default/files/files/file1/edpb_guidelines_202005_consent_en.pdf, 2020. [Online; accessed 28 Feb. 2023].

[49] European Data Protection Board. Binding Decision 4/2022 on the dispute submitted by the Irish SA on Meta Platforms Ireland Limited and its

Instagram service (Art. 65 GDPR). https://edpb.europa.eu/our-work-tools/our-documents/binding-decision-board-art-65/binding-decision-42022-dispute-submitted_en, 2022. [Online; accessed 28 Feb. 2023].

[50] Imane Fouad, Nataliia Bielova, Arnaud Legout, and Natasa Sarafijanovic-Djukic. Missed by filter lists: Detecting unknown third-party trackers with invisible pixels. *Proceedings on Privacy Enhancing Technologies*, 2020(2):499–518, 2020.

[51] FTC. Complying with COPPA: Frequently Asked Questions. https://www.ftc.gov/node/60394, 2020. [Online; accessed 28 Feb. 2023].

[52] FTC. Fortnite Video Game Maker Epic Games to Pay More Than Half a Billion Dollars over FTC Allegations of Privacy Violations and Unwanted Charges. https://www.ftc.gov/node/80135, 2022. [Online; accessed 28 Feb. 2023].

[53] FTC. Children's Online Privacy Protection Rule ("COPPA"). https://www.ftc.gov/node/60175, 2023. [Online; accessed 28 Feb. 2023].

[54] Liza Gak, Seyi Olojo, and Niloufar Salehi. The Distressing Ads That Persist: Uncovering The Harms of Targeted Weight-Loss Ads Among Users with Histories of Disordered Eating. *Proc. ACM Hum.-Comput. Interact.*, 6(CSCW2), Nov 2022.

[55] Digital Advertising - Worldwide | Statista Market Forecast. https://www.statista.com/outlook/dmo/digital-advertising/worldwide, 2023. [Online; accessed 28 Feb. 2023].

[56] @gorhill/ubo-core. https://www.npmjs.com/package/@gorhill/ubo-core, 2023. [Online; accessed 28 Feb. 2023].

[57] Maarten Grootendorst. BERTopic: Neural topic modeling with a class-based TF-IDF procedure. *arXiv preprint arXiv:2203.05794*, 2022.

[58] Amit Gupta and Rajesh Bhatia. Ensemble approach for web page classification. *Multimedia Tools and Applications*, 80(16):25219–25240, 2021.

[59] Raymond Hill and Nik Rolls. uBlock. https://github.com/gorhill/uBlock/blob/491bc87e94a503a17fd11cdee35c1f1b6fea24be/platform/mv3/make-rulesets.js#L1285-L1296, 2023. [Online; accessed 28 Feb. 2023].

[60] Hugging Face. AutoModels. https://huggingface.co/transformers/v3.0.2/model_doc/auto.html, 2023. [Online; accessed 28 Feb. 2023].

[61] Hugging Face. trainer · Hugging Face. https://huggingface.co/docs/transformers/main_classes/trainer, 2023. [Online; accessed 28 Feb. 2023].

[62] Ionut Ilascu. Hackers push malware via Google search ads for VLC, 7-Zip, CCleaner. *BleepingComputer*, January 2023. https://www.bleepingcomputer.com/news/security/hackers-push-malware-via-google-search-ads-for-vlc-7-zip-ccleaner.

[63] Google Inc. Google Cloud Vision API's SafeSearch Detection. https://cloud.google.com/vision/docs/detecting-safe-search, 2021. [Online; accessed 4 May 2023].

[64] IngoSearch. https://ingosearch.com, 2023. [Online; accessed 4 May 2023].

[65] Umar Iqbal, Steven Englehardt, and Zubair Shafiq. Fingerprinting the fingerprinters: Learning to detect browser fingerprinting behaviors. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1143–1161. IEEE, 2021.

[66] Umar Iqbal, Peter Snyder, Shitong Zhu, Benjamin Livshits, Zhiyun Qian, and Zubair Shafiq. Adgraph: A graph-based approach to ad and tracker blocking. In *IEEE Symposium on Security and Privacy (SP)*, pages 763–776, 2020.

[67] Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, et al. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*, 2017.

[68] Deborah Roedder John. Consumer socialization of children: A retrospective look at twenty-five years of research. *Journal of consumer research*, 26(3):183–213, 1999.

[69] Marc Juarez, Sadia Afroz, Gunes Acar, Claudia Diaz, and Rachel Greenstadt. A critical evaluation of website fingerprinting attacks. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pages 263–274, 2014.

[70] Jordan Jueckstock, Shaown Sarker, Peter Snyder, Aidan Beggs, Panagiotis Papadopoulos, Matteo Varvello, Benjamin Livshits, and Alexandros Kapravelos. Towards realistic and reproducible web crawl measurements. In *Proceedings of the Web Conference 2021*, pages 80–91, 2021.

[71] Tadayoshi Kohno, Andre Broido, and Kimberly C. Claffy. Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, 2(2):93–108, 2005.

[72] David Kristol and Lou Montulli. Rfc2965: Http state management mechanism, 2000.

[73] Priya Kumar, Shalmali Milind Naik, Utkarsha Ramesh Devkar, Marshini Chetty, Tamara L. Clegg, and Jessica Vitak. 'No Telling Passcodes Out Because They're Private': Understanding Children's Mental Models of Privacy and Security Online. *Proc. ACM Hum.-Comput. Interact.*, 1(CSCW), dec 2017.

[74] Dale Kunkel, Brian L Wilcox, Joanne Cantor, Edward Palmer, Susan Linn, and Peter Dowrick. Report of the APA task force on advertising and children. *Washington, DC: American Psychological Association*, 30:60, 2004.

[75] Tomer Laor, Naif Mehanna, Antonin Durey, Vitaly Dyadyuk, Pierre Laperdrix, Clémentine Maurice, Yossi Oren, Romain Rouvoy, Walter Rudametkin, and Yuval Yarom. DRAWNAPART: A Device Identification Technique based on Remote GPU Fingerprinting. In *Network and Distributed System Security Symposium (NDSS)*, 2022.

[76] Pierre Laperdrix, Oleksii Starov, Quan Chen, Alexandros Kapravelos, and Nick Nikiforakis. Fingerprinting in Style: Detecting Browser Extensions via Injected Style Sheets. In *USENIX Security Symposium*, pages 2507–2524, 2021.

[77] Margaret-Anne Lawlor and Andrea Prothero. Pester power–a battle of wills between children and their parents. *Journal of Marketing Management*, 27(5-6):561–581, 2011.

[78] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium*, 2019.

[79] Zhou Li, Kehuan Zhang, Yinglian Xie, Fang Yu, and XiaoFeng Wang. Knowing your enemy: understanding and detecting malicious web advertising. In *Proceedings of the ACM conference on Computer and communications security*, pages 674–686, 2012.

[80] LinkedIn Insight Tag | LinkedIn Marketing Solutions. `https://business.linkedin.com/marketing-solutions/insight-tag`, August 2023. [Online; accessed 3 Aug. 2023].

[81] Bin Liu, Anmol Sheth, Udi Weinsberg, Jaideep Chandrashekar, and Ramesh Govindan. AdReveal: Improving transparency into online targeted advertising. In *Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks*, pages 1–7, 2013.

[82] Daniel López-Sánchez, Angélica González Arrieta, and Juan M Corchado. Visual content-based web page categorization with deep transfer learning and metric learning. *Neurocomputing*, 338:418–431, 2019.

[83] MalVirt | .NET Virtualization Thrives in Malvertising Attacks. `https://www.sentinelone.com/labs/malvirt-net-virtualization-thrives-in-malvertising-attacks`, 2023. [Online; accessed 28 Feb. 2023].

[84] Arunesh Mathur, Gunes Acar, Michael J Friedman, Eli Lucherini, Jonathan Mayer, Marshini Chetty, and Arvind Narayanan. Dark patterns at scale: Findings from a crawl of 11k shopping websites. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–32, 2019.

[85] Jonathan R Mayer and John C Mitchell. Third-party web tracking: Policy and technology. In *2012 IEEE symposium on security and privacy*, pages 413–427. IEEE, 2012.

[86] Mary L McHugh. Interrater reliability: the kappa statistic. *Biochemia Medica*, 22(3):276–282, 2012.

[87] Tinhinane Medjkoune, Oana Goga, and Juliette Senechal. Marketing to children through online targeted advertising: Targeting mechanisms and legal aspects. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, CCS '23, pages 180–194, New York, NY, USA, 2023. Association for Computing Machinery.

[88] Member list. `https://www.kidsafeseal.com/certifiedproducts.html`, 2023. [Online; accessed 28 Feb. 2023].

[89] Microsoft. Content Moderator - Image Moderation. `https://docs.microsoft.com/en-us/azure/cognitive-services/content-moderator/image-moderation-api`, 2021.

[90] Ingrida Milkaite and Eva Lievens. Status quo regarding the child's article 8 GDPR age of consent for data processing across the EU. *BIK PORTAL*, 20(20/12/2019), 2019.

[91] Keaton Mowery and Hovav Shacham. Pixel perfect: Fingerprinting canvas in HTML5. *Proceedings of W2SP*, 2012, 2012.

[92] Maaz Bin Musa and Rishab Nithyanand. Atom: ad-network tomography. *Proceedings on Privacy Enhancing Technologies*, 4:295–313, 2022.

[93] Midas Nouwens, Ilaria Liccardi, Michael Veale, David Karger, and Lalana Kagal. Dark Patterns after the GDPR: Scraping Consent Pop-Ups and Demonstrating Their Influence. In *CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

[94] Infographic: How Many Websites Are There? `https://www.statista.com/chart/19058/number-of-websites-online`, 2023. [Online; accessed 28 Feb. 2023].

[95] Pixel for Marketing API - Meta Pixel - Documentation - Meta for Developers. `https://developers.facebook.com/docs/meta-pixel/implementation/marketing-api`, August 2023. [Online; accessed 2 Aug. 2023].

[96] Fast and reliable end-to-end testing for modern web apps | Playwright. `https://playwright.dev`, 2023. [Online; accessed 28 Feb. 2023].

[97] puppeteer. Puppeteer. `https://github.com/puppeteer/puppeteer`, 2023. [Online; accessed 28 Feb. 2023].

[98] Ali Rasaii, Shivani Singh, Devashish Gosain, and Oliver Gasser. Exploring the Cookieverse: A Multi-Perspective Analysis of Web Cookies. In *International Conference on Passive and Active Network Measurement*, pages 623–651. Springer, 2023.

[99] Ray Project (Anyscale). Hyperparameter Search with Transformers and Ray Tune. `https://huggingface.co/blog/ray-tune`, 2023. [Online; accessed 28 Feb. 2023].

[100] Nils Reimers and Iryna Gurevych. Sentence Embeddings using Siamese BERT-Networks. *arXiv preprint arXiv:1908.10084*, 2019.

[101] Nils Reimers and Iryna Gurevych. Making monolingual sentence embeddings multilingual using knowledge distillation. *arXiv preprint arXiv:2004.09813*, 2020.

[102] Restricted Content, Products, and Services. `https://help.taboola.com/hc/en-us/articles/115000220793-Restricted-Content-Products-and-Services`, 2023. [Online; accessed 1 Mar. 2023].

[103] Irwin Reyes, Primal Wijesekera, Joel Reardon, Amit Elazari Bar On, Abbas Razaghpanah, Narseo Vallina-Rodriguez, Serge Egelman, et al. "Won't somebody think of the children?" examining COPPA compliance at scale. In *The 18th Privacy Enhancing Technologies Symposium*, 2018.

[104] Franziska Roesner, Tadayoshi Kohno, and David Wetherall. Detecting and defending against third-party tracking on the web. In *Presented as part of the 9th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 12)*, pages 155–168, 2012.

[105] Rogue GIMP Google Ad Pushed Info-Stealer Malware Through Website Replica. `https://www.bitdefender.com/blog/hotforsecurity/rogue-gimp-google-ad-pushed-info-stealer-malware-through-website-replica/`, 2022. [Online; accessed 28 Feb. 2023].

[106] Esther Rozendaal, Moniek Buijzen, and Patti Valkenburg. Comparing children's and adults' cognitive advertising competences in the Netherlands. *Journal of Children and Media*, 4(1):77–89, 2010.

[107] Iskander Sanchez-Rola, Xabier Ugarte-Pedrero, Igor Santos, and Pablo G Bringas. The web is watching you: A comprehensive review of web-tracking techniques and countermeasures. *Logic Journal of the IGPL*, 25(1):18–29, 2017.

[108] scikit-learn developers. Scikit-learn: Machine Learning in Python. `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.fbeta_score.html`, 2011–. Version 1.0.2 [Online; accessed 28 Feb. 2023].

[109] sentence-transformers/all-mpnet-base-v2 Hugging Face. `https://huggingface.co/sentence-transformers/all-mpnet-base-v2`, 2023. [Online; accessed 28 Feb. 2023].

[110] Sandra Siby, Umar Iqbal, Steven Englehardt, Zubair Shafiq, and Carmela Troncoso. WebGraph: Capturing advertising and tracking information flows for robust blocking. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 2875–2892, 2022.

[111] Alexander Sjösten, Daniel Hedin, and Andrei Sabelfeld. EssentialFP: Exposing the Essence of Browser Fingerprinting. In *IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 32–48, 2021.

[112] Solutions. `https://throtle.io/solutions`, 2021. [Online; accessed 31 Jul. 2023].

[113] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. Mpnet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, 33:16857–16867, 2020.

[114] Xiang Song, Yi Zhu, Xuemei Zeng, and Xingshu Chen. Hierarchical contaminated web page classification based on meta tag denoising disposal. *Security and Communication Networks*, 2021, 2021.

[115] State of the Union Address. https://www.whitehouse.gov/state-of-the-union-2023/, 2023. [Online; accessed 28 Feb. 2023].

[116] Ariel Stolerman, Rebekah Overdorf, Sadia Afroz, and Rachel Greenstadt. Classify, but verify: Breaking the closed-world assumption in stylometric authorship attribution. In *IFIP Working Group*, volume 11, page 64, 2013.

[117] Grant Storey, Dillon Reisman, Jonathan Mayer, and Arvind Narayanan. The future of ad blocking: An analytical framework and new techniques. *arXiv preprint arXiv:1705.08568*, 2017.

[118] Stripe | Payment Processing Platform for the Internet. https://stripe.com/en-nl, 2023. [Online; accessed 28 Feb. 2023].

[119] Karthika Subramani, Xingzi Yuan, Omid Setayeshfar, Phani Vadrevu, Kyu Hyung Lee, and Roberto Perdisci. When push comes to ads: Measuring the rise of (malicious) push advertising. In *Proceedings of the ACM Internet Measurement Conference*, pages 724–737, 2020.

[120] Ege Tekiner, Abbas Acar, A Selcuk Uluagac, Engin Kirda, and Ali Aydin Selcuk. SoK: cryptojacking malware. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 120–139, 2021.

[121] The EasyList authors. Easylist. https://easylist.to/easylist/easylist.txt, 2023. [Online; accessed 28 Feb. 2023].

[122] The list of kids' websites. https://kids.kaspersky.com/kids-website-list, 2023. [Online; accessed 28 Feb. 2023].

[123] The Ray Team. Tune: Scalable Hyperparameter Tuning — Ray 2.2.0. https://docs.ray.io/en/latest/tune/, 2023. [Online; accessed 28 Feb. 2023].

[124] Tracker Radar Collector. https://github.com/duckduckgo/tracker-radar-collector, 2023. [Online; accessed 28 Feb. 2023].

[125] uBlock Origin - An efficient blocker for Chromium and Firefox. Fast and lean. https://github.com/gorhill/uBlock, 2023. [Online; accessed 10. Jul. 2023].

[126] Tobias Urban, Martin Degeling, Thorsten Holz, and Norbert Pohlmann. Beyond the front page: Measuring third-party dynamics in the field. In *Proceedings of The Web Conference 2020*, pages 1275–1286, 2020.

[127] Pelayo Vallina et al. *Advanced Methods to Audit Online Web Services*. PhD thesis, Universidad Carlos III de Madrid, Spain, 2023.

[128] Pelayo Vallina, Victor Le Pochat, ÃĄlvaro Feal, Marius Paraschiv, Julien Gamba, Tim Burke, Oliver Hohlfeld, Juan Tapiador, and Narseo Vallina-Rodriguez. Mis-shapes, Mistakes, Misfits: An Analysis of Domain Classification Services. In *Internet Measurement Conference*, IMC 2020, pages 598–618, 2020.

[129] Giridhari Venkatadri, Elena Lucherini, Piotr Sapiezynski, and Alan Mislove. Investigating sources of PII used in Facebook's targeted advertising. *Proc. Priv. Enhancing Technol.*, 2019(1):227–244, 2019.

[130] Natalija Vlajic, Marmara El Masri, Gianluigi M Riva, Marguerite Barry, and Derek Doran. Online tracking of kids and teens by means of invisible images: COPPA vs. GDPR. In *Proceedings of the 2nd International Workshop on Multimedia Privacy and Security*, pages 96–103, 2018.

[131] We retired Alexa.com on May 1, 2022. `https://web.archive.org/web/20221020130106/https://support.alexa.com/hc/en-us/articles/4410503838999`, 2023. [Online; accessed 28 Feb. 2023].

[132] Join the Smart Affiliate Marketing Network | WEBGAINS. `https://www.webgains.com/public/en`, 2023. [Online; accessed 28 Feb. 2023].

[133] Kenneth C Wilbur and Yi Zhu. Click fraud. *Marketing Science*, 28(2):293–308, 2009.

[134] Now even YouTube serves ads with CPU-draining cryptocurrency miners. `https://arstechnica.com/information-technology/2018/01/now-even-youtube-serves-ads-with-cpu-draining-cryptocurrency-miners/`, 2018. [Online; accessed 28 Feb. 2023].

[135] Apostolis Zarras, Alexandros Kapravelos, Gianluca Stringhini, Thorsten Holz, Christopher Kruegel, and Giovanni Vigna. The dark alleys of madison avenue: Understanding malicious advertisements. In *Proceedings of the Conference on Internet Measurement Conference*, pages 373–380, 2014.

[136] David Zeber, Sarah Bird, Camila Oliveira, Walter Rudametkin, Ilana Segall, Fredrik Wollsén, and Martin Lopatka. The representativeness of automated web crawls as a surrogate for human browsing. In *Proceedings of The Web Conference 2020*, pages 167–178, 2020.

[137] Eric Zeng, Tadayoshi Kohno, and Franziska Roesner. Bad News: Clickbait and Deceptive Ads on News and Misinformation Websites. In *Workshop on Technology and Consumer Protection (ConPro)*, 2020.

[138] Eric Zeng, Tadayoshi Kohno, and Franziska Roesner. What makes a "bad" ad? User perceptions of problematic online advertising. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–24, 2021.

[139] Eric Zeng, Miranda Wei, Theo Gregersen, Tadayoshi Kohno, and Franziska Roesner. Polls, clickbait, and commemorative $2 bills: problematic political advertising on news and media websites around the 2020 US elections. In *Proceedings of the 21st ACM Internet Measurement Conference*, pages 507–525, 2021.

[140] Jun Zhao, Ge Wang, Carys Dally, Petr Slovak, Julian Edbrooke-Childs, Max Van Kleek, and Nigel Shadbolt. 'I Make up a Silly Name': Understanding Children's Perception of Privacy Risks Online. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–13, New York, NY, USA, 2019.

[141] Qiuhan Zhao, Wenchuan Yang, and Rui Hua. Design and research of composite web page classification network based on deep learning. In *IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1531–1535, 2019.

[142] Yanjie Zhao, Tianming Liu, Haoyu Wang, Yepang Liu, John Grundy, and Li Li. Are mobile advertisements in compliance with app's age group? In *Proceedings of the ACM Web Conference 2023*, pages 3132–3141, 2023.

Appendix

Criteria for labeling child-directed websites

To identify a child-directed website, we manually visit and review its design, content, and policies, including necessary translations to English. A site is labeled as child-directed if any of the following conditions are met:

- Does the website include content, activities, or games that can be used by children?
- Does the website promote products (e.g., apps, sites, books, videos, workshops, animations, etc.) designed for and usable by children online?
- Does the website include content or promote products whose end users are children, but children's parents must first subscribe or register?

A site is not child-directed if one of the following is true:

- The website redirects to another page that is not child-directed.
- The website features children-related products intended for adult use, such as parents or teachers.
- The website is generally appealing to adults (e.g., news or academic websites).

Manual Verification of the Classifier Output

Two researchers manually labeled a total of 2,500 websites detected as child-directed by the classifier. Initially, both researchers jointly labeled a 50-website sample, reaching agreement on 45 of these decisions (Cohen's Kappa=0.79 [86]). The remaining websites were divided into two equal batches and labeled individually by the two researchers. This process took approximately one person-week to complete. We followed the criteria for identifying child-directed websites (Appendix 8) and considered four potential labels for each website: *child-directed*, *child-related*, *non-children* and *unknown*. The majority of the websites (64%) were labeled as child-directed, , while 23% were identified as child-related, indicating they are relevant to children but primarily intended for parents or teachers. About 4.5% of the websites were inaccurately classified as non-children's sites, such as entertainment bands, academic forums, and movie streaming services, mostly due to vague or brief titles and descriptions. In some cases, discerning whether websites were targeted at children, parents, or teachers was challenging, leading to 5.5% being labeled as 'unknown' due to uncertainty about their target audience. Of the misclassified websites, we found that four were adult entertainment websites (0.16%) that had very short metadata fields mentioning words such as "teens", "cartoon", "animations" which likely caused the misclassification.

Ad Transparency Statements

The following ad transparency statements are used to classify advertisements as targeted or non-targeted. Note that targeted categories also include retargeting and behavioral ads. The statements are compiled from Google's and Criteo's ad disclosure interfaces, reached via the AdChoices icon. When searching for the statements, we use exact, case-insensitive search.

Targeted:

- Google's estimation of your interests
- Websites you've visited
- Your similarity to groups of people the advertiser is trying to reach
- Your activity on Google on this device
- According to your activity on this device
- You have enabled ad personalization
- Information collected by the publisher. The publisher partners with Google to show ads
- Google's estimation of the languages you know, based on your activity on this device
- Your visit to the advertiser's website or app
- The advertiser's interest in reaching new customers who haven't bought something from them before

Non-Targeted:

- Ad personalization is turned off
- You have turned off ad personalization
- Ad personalization is off
- The time of day or your general location
- Google shows ads based on general factors like the time of day and the info on a page, our policies, and your ad personalization settings
- The information on the website you were viewing
- General factors about the placement of the ad

Detecting failed or errored visits

We classified a visit as failed under the following conditions: if the first request elicited a 4XX or 5XX error; if the size of the first non-3XX response (root document) was less than 512 bytes, following Le Pochat et al. [78]; or if there was no successful (200 OK) response.

Meta-Review

The following meta-review was prepared by the program committee for the 2024 IEEE Symposium on Security and Privacy (S&P) as part of the review process as detailed in the call for papers.

Summary

This paper examines the presence of both tracking and inappropriate ads on websites directed at children. The authors built a pipeline to identify children's websites and collect their advertisements, which prior work achieved manually. After collecting these advertisements, the paper also demonstrates that in addition to a high prevalence of trackers which should require informed parental consent, many children's sites also display inappropriate ads.

Scientific Contributions

- Independent Confirmation of Important Results with Limited Prior Research
- Provides a New Data Set For Public Use
- Creates a New Tool to Enable Future Science
- Provides a Valuable Step Forward in an Established Field
- Other

Reasons for Acceptance

1. This paper makes novel progress on studying tracking and targeted advertising as it relates to children, particularly in its dramatic increase in scope compared to prior work.
2. The authors provide a new dataset of child-directed websites that is much larger than prior work, along with the other data and code used for analysis in regards to better measuring tracking and targeted ads.

Noteworthy Concerns

1. "Improper" as an ad criterion is not clearly defined. Although the authors reference prior work on this topic, further discussion of how and why they picked the specific four criteria used in the paper would be useful for contextualizing their results.

2. One reviewer noted that the recall of child-directed websites was rather low at 70%, leading to potential skew in the analysis.

Response to the Meta-Review

1. Our analysis of improper ads is exploratory, and it serves as an initial step toward a more extensive investigation into the problematic advertising practices. We focus on four types of improper ads (weight loss, mental health, dating services, and racy content), because prior research [54, 23] and regulatory guidelines [25] have argued that these can be especially harmful to children. Moreover, these categories are generally restricted by ad networks' own moderation policies [5, 102].

   While our study focuses on these four categories, our versatile ML improper ad detection pipeline can be easily adapted to detect different kinds of improper ads, for instance, by simply changing the search queries (Figure 4).

2. The lower recall rate is the result of employing the "Classify-Verify" method in our classifier. This approach enhances precision by applying a threshold, which leads to lower recall but effectively reduces false positives and the manual labeling workload (§3.2). In particular, following Juarez et al. [69], we choose the threshold that maximizes $F_{\beta=0.5}$, which gives more weight to precision to reduce false positives. Researchers who adopt our methods can easily tweak the threshold to strike a different balance based on their classification needs and manual verification budgets.

   We acknowledge that selective (abstaining) classification may have a disparate effect on websites that are classified with low confidence. Other factors including the website language, title and descriptions may also skew the results, as we note in the Limitations. We acknowledge these limitations and do not claim that our list of child-directed websites is representative of the child-directed websites at large.

Publication

The Double Edged Sword: Identifying Authentication Pages and their Fingerprinting Behavior

Publication Data

Senol, A., Ukani, A., Cutler, D., and Bilogrevic, I. The Double Edged Sword: Identifying Authentication Pages and their Fingerprinting Behavior. In *The Web Conference (WWW), 2024* (2024)

Contributions

- Primary author (shared authorship with Alisha Ukani)

The Double Edged Sword: Identifying Authentication Pages and their Fingerprinting Behavior

Asuman Senol*[1], Alisha Ukani*[2], Dylan Cutler[3], and Igor Bilogrevic[3]

[1] COSIC, KU Leuven
[2] UC San Diego
[3] Google

Abstract. Browser fingerprinting is often associated with cross-site user tracking, a practice that many browsers (e.g., Safari, Brave, Edge, Firefox, and Chrome) want to block. However, less is publicly known about its uses to enhance online safety, where it can provide an additional security layer against service abuses (e.g., in combination with CAPTCHAs) or during user authentication. To the best of our knowledge, no fingerprinting defenses deployed thus far consider this important distinction when blocking fingerprinting attempts, so they might negatively affect website functionality and security.

To address this issue we make three main contributions. First, we introduce a novel machine learning-based method to automatically identify authentication pages (i.e. Login and sign-up pages). Our supervised algorithm achieves 96-98% precision and recall on a manually-labelled dataset of almost 1,000 popular sites. Second, we compare our algorithm with methods from prior works on the same dataset, showing that it significantly outperforms all of them. Third, we quantify the prevalence of fingerprinting scripts across login and sign-up pages (10.2%) versus those executed on other pages (9.2%); while the rates of fingerprinting are similar, home pages and authentication pages differ in the third-party scripts they include and how often these scripts are labeled as tracking. We also highlight the substantial differences in fingerprinting on login and sign-up pages. Our work sheds light on the complicated reality that fingerprinting is used to both protect user security

*Both authors contributed equally to this research, which was partially conducted while they were with Google.

and invade user privacy; this dual nature must be considered by fingerprinting mitigations.

Keywords – browser fingerprinting; online authentication; privacy, machine learning

1. Introduction

Browser fingerprinting has garnered attention for its widespread use for tracking individuals' online activities. Doing so involves collecting a set of attributes from a user's web browser and device in order to derive a *quasi*-unique identifier that persists across different websites. Some attributes that could be used for creating a browser fingerprint include browser features and configurations (e.g. the User-Agent string, the canvas API, installed plugins), OS features (e.g. emoji sets), and even hardware features (e.g. battery level [15]). A 2021 study found that approximately 10% of sites perform fingerprinting [23]. This rate may increase over time, since the deprecation of third-party cookies [36, 54, 45] might nudge online trackers to switch to cookieless alternatives, such as fingerprinting.

What makes fingerprinting attractive for tracking – the ability to uniquely identify a device – also gives it potential for enhancing security. Consider the case of an unauthorized user logging into a victim's account with the correct username and password, but a different fingerprint. The site could send a multi-factor authentication (MFA) prompt to the victim. This approach not only enhances security but also minimizes user inconvenience by forgoing additional hardware and complex authentication procedures. Beyond preventing account compromise, fingerprinting can also help web services detect bots–and thus prevent click fraud–and prevent cookie hijacking [13]. In the absence of privacy-friendly alternatives, it is important to take into account the context in which fingerprinting occurs when making an enforcement decision against it.

Durey et al. studied the usage of browser fingerprinting as a means to enhance web security [13]. They manually analyzed four-page categories – login, sign-up, payment, and shopping cart – across 1,485 pages from 446 domains. While their work provided an important initial set of results, it suffered from two main limitations. First, it relied solely on a manual analysis of websites, which is not scalable nor generalizable. Second, it did not consider the impact of any existing anti-fingerprinting tool on such websites.

To address these limitations, we developed a novel machine learning (ML) model to identify login and sign-up pages, and we present a large-scale empirical study on the usage of browser fingerprinting on these pages. Our model, which achieves at least 96% precision and recall for each page category, can run on-device for on-the-fly inference, as demonstrated by our publicly available Chrome extension. Many studies analyze login and sign-up pages [53, 41, 9], predominantly relying on heuristic methods for detection in a centralized, server-based setting. However, given the dynamic nature of the web, rule-based detection techniques quickly become outdated (e.g. login and sign-up pages with multi-step designs or lacking password fields) and thus more dynamic

approaches, such as our ML-based solution, are necessary.

Our study makes the following key contributions:

- A web measurement study evaluating browser fingerprinting for security on the top 100,000 websites' login and sign-up pages. Using an instrumented crawler, we detect fingerprinting attempts and potential login/sign-up indicators.
- An ML model[4] identifying login and sign-up pages with high precision and recall (96-98%), which is publicly available.
- A browser extension[5] for identifying and displaying login and sign-up pages, along with a web crawler[6] for listing such pages in given URLs, both of which are publicly available.

Our results show that 9.2% of login and sign-up pages perform fingerprinting, compared to 8.9% across all pages. While the rates are similar, fingerprinting scripts on home pages are more likely to be classified as trackers and far more likely to perform canvas fingerprinting than scripts on login and sign-up pages. For sites that fingerprint on at least one authentication (including both login and sign-up pages) page, 50% of them fingerprint only on the login page. When sites fingerprint on both login and sign-up pages, they use scripts from the same set of third-parties in 98% of cases. These new findings show the multifaceted intent behind fingerprinting scripts on the web.

2. Background and Related Work

In this section we provide background on fingerprinting, including its use for web authentication, and describe prior work on identifying login and sign-up pages.

2.1 Browser Fingerprinting & Mitigations

Browser fingerprinting is a method used to generate a unique identifier that can link the same browser across different domains or visits. It is derived by joining multiple, measured attributes of the user's browser, typically via HTTP

[4] https://github.com/asumansenol/double_edged_sword_data
[5] https://github.com/asumansenol/login_signup_detection_chrome_extension
[6] https://github.com/asumansenol/double_edged_sword_crawler

headers and JavaScript APIs. Fingerprinting is effective for tracking because it is stateless, less visible to the users, and – unlike tracking cookies – difficult to disable. For example, a script may learn about the user's timezone and the list of fonts they have installed on their system. As the script collects more information, it can potentially uniquely distinguish a user among millions of visitors to a given site.

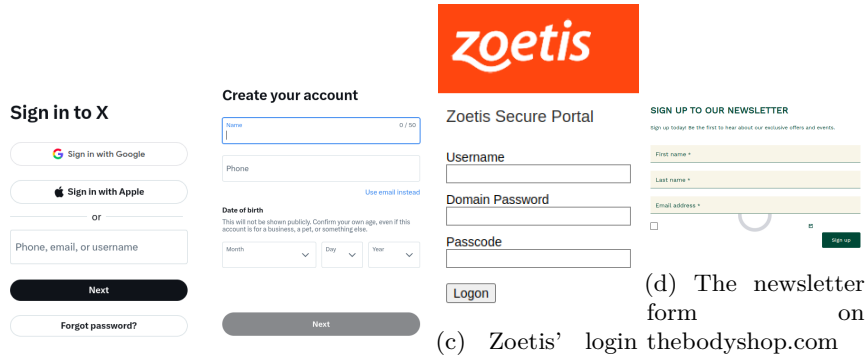
Mitigations for fingerprinting broadly rely on four approaches: randomization, normalization, heuristics, and machine learning. Randomization methods, like Privaricator [37] and FPRandom [28], add noise to APIs like canvas so that the same user presents different fingerprints during different sessions. However, adding noise may affect API functionality and is reversible, potentially serving as a fingerprint itself. Another approach, normalization, aims to standardize fingerprints for multiple users and is implemented by the Tor and Brave browsers [49, 7].

While randomization and normalization attempt to disrupt fingerprinting scripts, the next two approaches try to identify fingerprinting to block it entirely. Heuristic-based identification methods like Privacy Badger [40], JShelter [27], and Disconnect [10] rely on predefined rules, which can miss some fingerprinting scripts and require continuous updates. Learning-based methods such as FP-Inspector [23] are more effective but usually have worse precision than heuristic-based methods.

2.2 Fingerprinting for Web Authentication

Many studies have proposed using browser fingerprinting as an additional authentication technique [4, 3, 20, 51, 39, 42]. Doing so can protect users whose credentials are stolen; if an attacker gets a user's credentials, the website can detect that the attacker has a different fingerprint than the victim. Then the site could show a multi-factor authentication (MFA) prompt to the attacker to protect the user. Prior work has also proposed using fingerprinting to protect against cookie hijacking by detecting when cookies are used on a device with a different fingerprint [13], and to quickly identify bots [13, 55]. Our work focuses more on how sites may use fingerprinting to prevent account compromise than on bot detection and click fraud prevention.

Our work builds on the analysis from Durey et al. [13], which manually analyzed 1,485 pages from 446 domains to detect browser fingerprinting on a variety of page categories, including login and sign-up pages. Login and sign-up pages each compose 12-13% of the pages they analyze, and their classifier flags 23.4% and 31.1% of login and sign-up pages as performing fingerprinting, respectively. Finally, they analyze 14 scripts developed for security and find four that are used



(a) Twitter's login form without password field.

(b) Twitter's sign-up form without password field.

(c) Zoetis' login form without email and with two password fields.

(d) The newsletter form on thebodyshop.com closely resembles the account registration form.

Figure 1: Example web forms: ⓐ is a login form without a password field; ⓑ is a sign-up form without a password field; ⓒ is a login form without an email field; and ⓓ is a newsletter sign-up form that resembles an account creation form.

exclusively on login and sign-up pages for payment platforms, fraud prevention, and bot detection. We build on this paper by analyzing a larger set of sites and performing automated analysis; we also analyze the most popular sites to understand the prevalence of fingerprinting in a more general context.

We also build on the work of Lin et al. [30], which presents and evaluates an attack that uses fingerprints to bypass MFA. They detect login pages using the approach from Drakonis et al. [12], which we evaluate in §3.1. They detect login pages for 11,527 of the Alexa top 20K websites and find that the majority of these sites perform some fingerprinting. We add to these results by using a more sophisticated login page detection technique and analyzing a larger set of sites (100K versus 20K). We compare findings in detail in §4.3. Lin et al. were some of the first to present concrete security vulnerabilities that stem from using fingerprinting for authentication; we provide a more comprehensive overview of how the web ecosystem uses fingerprinting for authentication.

2.3 Login/Sign-Up Page Detection

Several studies have tried to identify and analyze authentication pages [41, 8, 52, 12, 57, 19, 26, 31, 2, 13]. Most of the approaches rely either on manual inspection [41, 13] or on heuristics based on regex patterns [8, 52, 12, 57, 19, 26].

Such regex patterns often include variations of the terms "login" and "sign-up" and include translations to other languages. Some of these studies used additional heuristics in addition to the regex strings, such as checking the visibility of elements and the types of inputs in forms (e.g. number of password fields) [12, 26]. Other studies queried search engines to discover authentication pages for a given domain [52, 19, 26].

However, these heuristics often fail to detect complex authentication flows, such as multi-step login (as illustrated in Figure 1ⓐ). In this example, the login form only contains a username field and requires the user to click before showing a password field. There are similar multi-step flows for sign-up forms (see Figure 1ⓑ). In addition, heuristics and regex patterns can lead to misclassification. For example, some heuristics from prior work classify forms with multiple password elements as sign-up forms [12], or forms with at least three visible input fields as sign-up forms [26]. The form in Figure 1ⓒ is a login form that would be missed by these heuristics. Regex patterns can also misclassify newsletter forms as sign-up forms, as shown in Figure 1ⓓ.

Instead of relying on heuristics and regex patterns, two studies trained machine learning models [31, 2]. The feature sets used by both studies tried to capture three key components: 1) the presence of login/sign-up keywords, 2) the number of password input fields, and 3) the number of all form input fields. For instance, Al Roomi and Li [2] achieved 94.5%/96.3% precision/recall for login forms, and 77.1%/99.5% precision/recall for sign-up forms, while Lodrant [31] achieved 71% accuracy. Both works found detecting multi-step authentication forms challenging.

3. Methodology

In this section we describe the methodology we used to collect the website data via an instrumented crawler, how we extracted features to train the ML model to detect login and sign-up pages, and the methods we used to detect fingerprinting on such pages.

3.1 Existing Login/Sign-Up Page Detection Methods

In this section, we describe three existing methods used to identify login and sign-up pages.

Prior Work Heuristics

Automated discovery of login and sign-up pages is an area of study in prior works. We implement heuristics from one study by Drakonakis, Ioannidis, and Polakis into our crawler, as their code is publicly available [12].[7] Their methodology identifies login and sign-up pages using a combination of regex string searching (for English phrases such as "register," "login," and "my profile") and heuristics based on DOM elements. For example, the latter includes the number of password elements, the presence of inputs for phone numbers or dates, and the visibility of those input elements. This methodology was also used by Lin et al. in their preliminary analysis of the use of fingerprinting on login and sign-up pages [30].

Autofill Heuristics

Autofill is a Chrome feature that automatically generates new passwords when the user visits a sign-up form. Users can also opt to save their credentials with Chrome so that Autofill can automatically fill form fields on their behalf. While it has been studied for its security risks [29, 1, 38] and its impact on developers [35], to the best of our knowledge, it has not been studied as a tool for automated login/sign-up form detection.

```
<form>
  <input type="text" name="username" pm_parser_annotation="
    username_element">
  <input type="password" name="password" pm_parser_annotation="
    new_password_element">
</form>
```

Listing 1: Example of Autofill annotations for a web form

We collect Autofill output by enabling the `show-autofillsignatures` Chrome flag, which adds HTML attributes (called `pm_parser_annotation`) to each form input element. An example is shown in Listing 1. The Autofill annotations are only available to Puppeteer crawlers in the "new" headless mode, which includes the code in the `//chrome` path (which was previously included in headful crawlers but not "old" headless crawlers) where Autofill is implemented [34]. These annotations only include Autofill's client-side heuristics; Autofill also has a server-side component [5] but it is inaccessible outside of Google. So, the client-side heuristics alone may not perfectly reflect users' actual experience with the Autofill feature.

Once the annotations are collected, classification is simple: if we see annotations for creating a new password (`new_password_element`) or confirming a password (`confirmation_password_element`), then we classify the form as a sign-up form.

[7] The heuristics from prior work [12] are available at `https://gitlab.com/kostasdrk/xdriver3-open/-/blob/master/js/scripts.js`

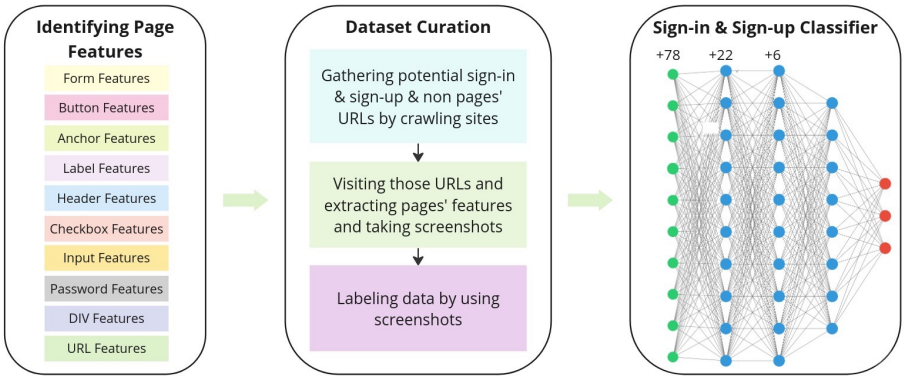


Figure 2: ML pipeline for classifying login/sign-up pages.

Otherwise, if there are annotations for a username field (`username_element`) and a password field (`password_element`), we classify the form as a login form.

Fathom-based Login & Sign-up Classifier

Fathom is a supervised-learning framework developed by Mozilla for identifying various components of web pages [16]. Their repository of rule sets showcases the integration of multiple ML models designed to detect various types of web page elements, such as pop-ups and even specific HTML components like price tags [17]. Fathom includes classifiers for login and sign-up pages [32, 47], which we integrated into our web crawler without any modifications.

3.2 Our Login/Sign-up Page Classifier

Hereafter we describe how we generate the test, validation, and training datasets, as well as the login and sign-up page classifier depicted in Figure 2.

Page Type Identification & Feature Extraction

We used the Chrome User Experience Report (CrUX) [25] to compile a list of the top 100 sites and manually browsed them to find the login and sign-up pages, if present. We manually reviewed the source code of these pages and extracted 88 features related to distinct aspects of the design and interaction modalities. For instance, we created a regex pattern that includes many variations of terms like "login" and "sign-up" as well as their translations into several other languages; we checked form attributes, button text content and attributes, header attributes, and other HTML elements for regex matches. Finally, we checked the presence of a checkbox element with a "Remember Me" pattern for login pages. We

Table 1: Classifier performance on test dataset.

Page Type	Accuracy	Precision	Recall	F1-score
Login	0.98	0.99	0.98	0.98
Sign-up	0.95	0.96	0.96	0.96
Neither	0.98	0.99	0.99	0.99

created variations of these features that would check if they are in a form, are in an iFrame, and are visible. The full feature set is detailed in Appendix 6.

Dataset Curation

We created a training dataset by crawling the CrUX top 10K homepages and visiting 47K other pages linked from the homepages that match the aforementioned regex string created to find login and sign-up pages (we describe this regex string and our inner page collection process in more detail in §3.2). We manually labeled a random sample of 1,500 login pages, 1,000 sign-up pages, and 2,500 non-authentication pages, totaling 5,000 labeled pages. However, we were unable to collect all 88 features for our model for 93 (1.9%) pages due to bot detection mechanisms; more specifically, we collected an average of 2.19 features on non-authentication pages. We filtered our dataset to only retain pages where we could collect a minimum of three features, which resulted in a dataset of 1,299 login pages, 973 sign-up pages, and 2,453 non-authentication pages for a total of 4,725 labeled pages. We split this into 67% for training and 33% for testing.

Model Training

We used the TensorFlow [48] framework to train a multi-class classifier. For each visited page, we generate an 88-dimensional feature vector and a label, which we fed to a neural network with two dense hidden layers containing 16 and 8 units, respectively. The output softmax layer is mapped to the three classes, i.e., login, sign-up, and neither. We trained the model for 200 epochs using the cross-entropy loss function.

Model Performance

The classifier achieves high performance on the test dataset (see Table 1). Specifically, it scores a recall of 0.98 and 0.96 and a precision of 0.99 and 0.96 on login and sign-up pages, respectively. The model has an error rate of 1%-4%, and is able to correctly label the vast majority of the login and sign-up pages in the dataset. It is also able to identify SSO login pages.

Fingerprinting Detection

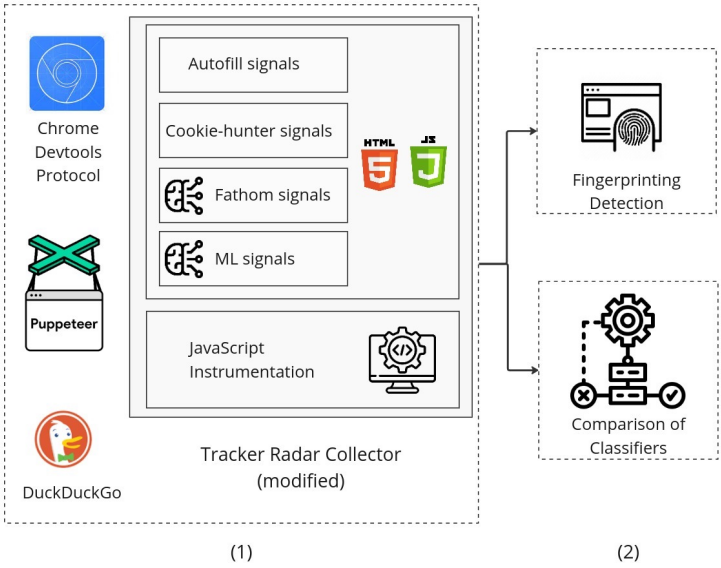


Figure 3: (1) Our crawler extends the Tracker Radar Collector. We collect login and sign-up signals, fingerprinting signals, and JavaScript execution traces. (2) We then compare all authentication page detection methods and detect fingerprinting scripts.

Browser fingerprinting has a rich history, marked by the ongoing evolution of fingerprinting and detection thereof. Prior work has used both ML-based methods [23] and heuristic approaches [15] to identify fingerprinting scripts. In this work, we relied on the heuristics established by Englehardt and Narayanan (described in Appendix 6) that monitor the Canvas, WebRTC, Canvas Font, and AudioContext APIs to detect fingerprinting scripts [15]. We logged function calls (including arguments and return values) by overriding getter and setter functions on all pages, including subframes, immediately after the document was created.

Crawler Implementation

Our web crawler is a fork of Tracker Radar Collector (TRC)[8], a crawler created by DuckDuckGo. TRC is built on Puppeteer and is designed to capture specific interactions with JavaScript APIs, HTTP requests and responses, cookies, and other relevant data for web measurements. We use the "new" headless mode [34] to collect Autofill signals (see §3.1).

[8] `https://github.com/duckduckgo/tracker-radar-collector`

Furthermore, we implemented a distinct collector to instrument method calls and property accesses related to fingerprinting (as explained in §3.2). In our approach, we modify objects' *getters* to intercept function calls.[9]

To more closely imitate a genuine user, our crawler scrolls to the bottom of the page and back to the top, and pauses for 5 seconds before collecting data. We also experimented with filling in some form elements to activate a wider range of fingerprinting scripts. We conducted two preliminary crawls on 1,000 domains: one where we filled in input fields, and one where we did not. These crawls revealed that 102 scripts attempted fingerprinting on 136 sites, regardless of whether input fields were filled or not. Since there was no difference in fingerprinting behavior, we did not fill in any form fields in subsequent crawls.

We executed both the homepage and inner page crawls using cloud-based servers provided by DigitalOcean, located in the United States. Each individual crawl was completed within a three-day timeframe, using a server with 8 vCPU cores and 16GB of RAM. We used a US-based server to minimize encounters with cookie consent modals.

Interaction with Consent Dialogs

After GDPR was enacted in 2018 [18], sites began to show cookie consent banners to give users transparency and control over the use of personal information. To increase the likelihood of triggering fingerprinting scripts, we chose to consent for all forms of personal data processing, including accepting cookies. To automatically interact with cookie consent banners, we integrated code derived from Priv-Accept [33, 24], a specialized crawler designed for this purpose. Priv-Accept finds HTML elements such as `<a>`, `<button>`, and `<div>`, then checks for keywords such as "Accept," and triggers a click action. We ported the Priv-Accept code from Python to JavaScript without changing any logic.

Collection of Potential Inner Authentication Pages

Login and sign-up flows might not be displayed on the homepages but rather on dedicated inner pages. Hence, our crawler must visit both types of pages. Our crawler follows a two-pass approach. On the first pass, it visits homepages and runs the four login/sign-up detection techniques (described in §3.1), collects fingerprinting signals, and identifies internal links that may be candidates for login/sign-up pages. On the second pass, the crawler visits these candidate pages and runs the four detection techniques and collects fingerprinting signals. The detection techniques are used to confirm if the candidate pages are actually authentication pages. We now explain this process in more depth.

[9] Although TRC already has the capability to intercept JavaScript API calls, we introduced a separate collector due to a known TRC bug that causes it to miss the initial function calls, documented in a public GitHub issue [14].

When crawling homepages, we collected links on the homepage that are potential candidates for login and sign-up forms. To increase the likelihood of discovering such pages, we employed a combined regular expression pattern used in our feature extraction (as described in §3.2). This pattern includes various word translations related to "login," "sign up," and "register." We employed this pattern search across multiple attributes of `<a>` elements, including `innerText, title, href, ariaLabel, placeholder, id, name, and className`. During this process, we filtered out URLs directing to non-HTML files, such as PDFs or images.

We validated our approach by crawling 300 of the CrUX top 1K most popular websites and checked how many login and sign-up pages our regex strings could identify. Our crawler initially navigated to the homepage of each website. If it found both login and sign-up pages on the homepage, it terminated; otherwise, it visited up to 15 (5 login + 5 sign-up + 5 neither) inner pages linked on the homepage that our regex pattern matched. Of the 300 websites in our sample, 4.7% had errors while loading the homepage. We successfully detected login or sign-up pages on 49% of sites, while 31% of sites did not contain such pages. In 7.7% of the remaining pages, the login and sign-up pages were only visible after interacting with an HTML element. In 4.7% cases, the login and sign-up pages were present but could not be detected by our ML model. In only 1.7% cases, the correct links were not identified using our regex pattern.

We also collected inner pages that were not login or sign-up pages. For these pages, we prioritized links closer to the viewport center to prevent the collection of unrelated links located in less visible areas, such as footers. To enhance crawl efficiency, we restricted the number of inner links to five for each category.

Table 2: Assessment of login and sign-up detection methods through the analysis of a randomly sampled set of 998 websites.

	Prior Work Heuristics		**Autofill**		**Fathom**		**Our ML-based solution**	
	Precision	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**
Login	0.83	0.51	0.51	0.75	0.77	0.79	0.97	0.89
Sign-up	0.58	0.54	0.47	0.66	0.36	0.95	0.83	0.92
Neither	0.71	0.97	0.80	0.69	0.88	0.78	0.92	0.95

Table 3: For each detection technique (explained in §3), we list the number of distinct login and sign-up pages it identifies as well as the number of domains (i.e. number of CrUX list entries) it can identify a login and sign-up page for.

	Prior Work Heuristics	**Autofill**	**Fathom**	**Our ML-based solution**
Total login pages	42,375	91,220	52,307	52,805
Total sign-up pages	17,517	31,103	138,639	21,988
Domains with at least one login page	22,369 (23.68%)	31,840 (33.70%)	24,963 (26.42%)	27,059 (28.64%)
Domains with at least one sign-up page	12,199 (12.91%)	15,620 (16.53%)	42,672 (45.16%)	15,998 (16.93%)

4. Measurement Results

Following recent best practices [44, 43], we crawled the top 100K domains from the CrUX report [25] (as of April 2023) in August 2023. We only used this dataset in our subsequent analysis. We excluded 1,155 URLs on the list with identical fully qualified domain names but different schemes; as a result, we attempted to crawl 98,845 homepages and successfully visited 94,482 homepages (95.8%). After collecting crawler results (including login and sign-up signals, inner links, screenshots, and fingerprinting attempts) for the homepages, we extracted the inner links for the second round of crawling. We attempted to crawl 474,436 inner pages and successfully visited 446,688 inner pages (94.4%). Priv-Accept facilitated the acceptance of personal data processing on 26.9% of pages crawled (including homepages and inner pages).

4.1 Comparison of Login/Sign-Up Detection Techniques

To compare login/sign-up detection techniques, we manually labeled a random sample of 998 pages (including both homepages and inner pages, based on the top 100K-crawl). These 998 pages included 261 login pages, 160 sign-up pages, 23 pages that had both login and sign-up functionality, and 532 non-authentication pages (plus 22 pages that had errors loading). We computed precision and recall scores for each detection technique, which we show in Table 2, and the number of pages classified as authentication pages by each technique in Table 3.

We found that our ML approach had the best precision/recall across page categories, demonstrating F1 scores of 0.93, 0.87, and 0.94 for login, sign-up, and neither categories, respectively. This is, in part, due to its ability to account for multi-step designs, to validate the visibility of login and sign-up elements, and to consider their presence in iFrames. Since this approach is more effective than the others, we use it to classify login and sign-up pages for all subsequent analysis. Other techniques were not as effective; for example, Fathom has 304 false positives for sign-up pages, which results in a low precision score of 0.36. We found these false positives were due to Fathom misclassifying newsletter and contact forms as sign-up forms. Another source of false positives was Fathom classifying 209 pages as both login and sign-up pages. In contrast, our Autofill-based approach could only classify pages as either login, sign-up or neither. We manually analyzed 23 pages that we knew were both login and sign-up pages to see how Autofill classified them. Of these 23 pages, Autofill classified 10 of them as only login, 12 as only sign-up, and one as neither. So, its inability to classify forms as both lowered its accuracy.

Table 4: The list of primary fingerprinting domains and related entities where at least one fingerprinting attempt was detected during a crawl conducted in August 2023. *Some entities may have multiple associated scripts.

All pages				Login and sign-up pages			
Entity	**Domain/Script**	**Category**	**Num. sites**	**Entity**	**Domain/Script**	**Category**	**Num. sites**
Adscore Tech.	adsco.re	Ad Motivated Tracking Ad Fraud	1,907	Signifyd Inc.	signifyd.com	Fraud Prevention	239
-	wpadmngr.com	Advertising	1,418	Alibaba Group	aeis.alicdn.com/AWSC/ WebUMID/1.93.0/um.js *	Marketing Analytics	201
Signifyd Inc.	signifyd.com	Fraud Prevention	1,414	Amazon Tech.	ssl-images-amazon.com	Marketing Advertising	171
Bounce Exchange	bounceexchange.com	Ad Motivated Tracking Advertising	1,330	Bounce Exchange	bounceexchange.com	Ad Motivated Tracking Advertising	159
InsurAds	insurads.com	Analytics	1,229	Sift Science, Inc.	sift.com	Fraud Prevention	148
Alibaba Group	aeis.alicdn.com/AWSC /WebUMID/1.93.0/um.js *	Marketing Analytics	959	FingerprintJS	cdnjs.cloudflare.com/ajax/libs/ fingerprintjs2/2.1.2/fingerprint2.min.js	Fraud Prevention Analytics	144
Rambler Holding	top100.ru	Audience Measurement	913	Amazon Tech.	d38xvr37kwwhcm.cloudfront.net/ js/grin-sdk.js	Marketing Advertising	139
Benhauer	salesmanago.pl	Customer Engagement	112	CHEQ AI Tech.	clickcease.com	Fraud Prevention	118
CHEQ AI Tech.	clickcease.com	Fraud Prevention	719	Rambler Holding	top100.ru	Audience Measurement	113
-	franecki.net	Marketing Analytics	589	Benhauer	salesmanago.pl	Customer Engagement	112

Table 5: The overall count of unique web pages where fingerprinting attempts were observed. Inner pages include login and sign-up pages.

	Homepages	**Inner pages**	**Login**	**Sign-up**
All	8,067 (8.5%)	40,828 (9.2%)	4,872 (9.2%)	2,737 (12.5%)
3rd party	4,639 (4.9%)	24,701 (5.6%)	2,294 (4.3%)	1,539 (6.8%)

4.2 Fingerprinting by Page Type

Rates of Fingerprinting

Table 5 shows that 9.2% of the pages our crawler visited were flagged as fingerprinting. This figure is slightly lower than the 10% rate from a 2021 study [23]. But, when we consider only login and sign-up pages, the percentage rises slightly to 10.2%, with the majority of scripts being attributed to third-party sources.

Third Party Fingerprinting Scripts

We find that websites often treat their login pages differently than their sign-up pages. Of the 1,902 domains that include third-party fingerprinting scripts on a login or sign-up page, 914 (48.05%) fingerprint on only the login page. The remaining sites are evenly split between fingerprinting only on the sign-up page (473 domains, 24.87%) and fingerprinting on both pages (515 domains, 27.08%). This pattern holds across popularity rankings; for every site rank bucket, 48-56% of the websites that fingerprint on an authentication page only do so on the

login page. While it is difficult to determine the intent behind fingerprinting, this suggests that sites are fingerprinting to enhance user security.

When sites fingerprint on both their login and sign-up pages, they almost exclusively use the same fingerprinting scripts on both. Of these 515 domains, 505 (98.06%) used scripts from the same set of third parties (based on the domains that provide the scripts). Of the 10 domains that had a different set of third parties, six include additional third parties on one of the pages, and one appears to use content served from the same entity that uses distinct domains (`bmcdn5.com` and `bmcdn6.com`). Only two domains appeared to use scripts from different entities. For example, `sunglasshut.com` includes a potentially fingerprinting script from `smct.io` on its sign-up page, but not on its login page. This script is from the company intent.ly (per the Disconnect entity list [11]), which advertises services for measuring and increasing customer conversions [22].

Tracking versus Non-Tracking Scripts

We labeled fingerprinting scripts as tracking or non-tracking using classifications from uBlock Origin Core [50], which rely on blocklists such as EasyList and EasyPrivacy. We found that home pages have the highest rate of tracking at 61.46%, compared to login pages at 50.50% and sign-up pages at 55.65%. The rate of tracking on authentication pages is surprising; sites are more likely to fingerprint on their login pages but less likely to use tracking scripts on login pages compared to sign-up pages. We checked how often each fingerprinting attribute (canvas, canvas fonts, WebRTC, and AudioContext) is called by tracking and non-tracking scripts. We found the prevalence of each attribute is about the same except for canvas font fingerprinting, which is used by non-tracking scripts more often (2.81% versus 0.73%).

Fingerprinting APIs

Lin et al. found that sites performed canvas fingerprinting an order of magnitude more than canvas fonts, WebRTC, or AudioContext fingerprinting [30]. We similarly investigate the rates of each type of fingerprinting for login, sign-up, and homepages in Figure 4. We confirm the finding from Lin et al. that canvas fingerprinting is the most popular; in fact, nearly every homepage that performs fingerprinting engages in canvas fingerprinting (93.10%). Interestingly, we find similar frequencies of the types of fingerprinting performed by login and sign-up pages. This supports our earlier finding that when sites fingerprint on both login and sign-up pages, they use the same fingerprinting scripts.

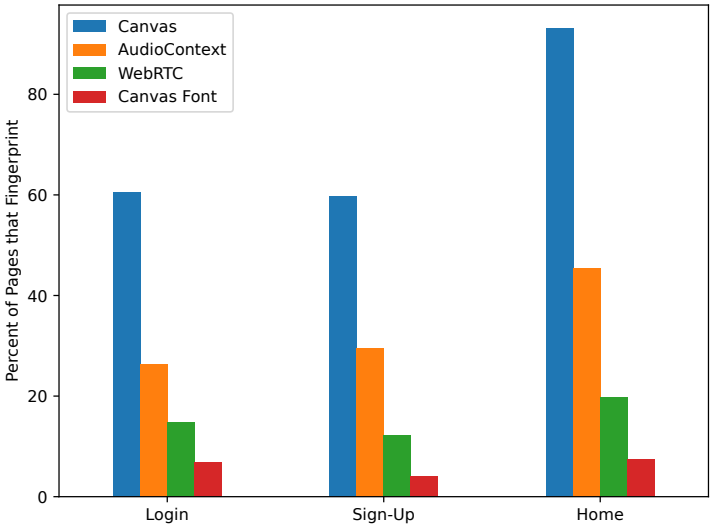


Figure 4: For login, sign-up, and homepages that perform fingerprinting, we plot the percent of pages that call each fingerprinting API.

4.3 Comparison to Prior Results

Compared to a previous study by Durey et al. [13] that examined the presence of fingerprinting scripts on payment, shopping cart, and authentication pages, we found fewer fingerprinting scripts on authentication pages. In the previous study, they reported the detection of fingerprinting scripts on sign-up pages (31.1%), login pages (23.4%), and homepages (23.0%). Our results, on the other hand, show lower percentages: 9.2% for login pages, 8.5% for homepages, and 12.5% for sign-up pages. Nevertheless, the trend of observing more fingerprinting scripts on sign-up pages compared to login pages, and more scripts on login pages than homepages, remains consistent.

Several distinctions exist between our study and the prior one that explain the different fingerprinting rates. First, there is a disparity in the number of websites examined: we analyzed 100,000 sites while they only assessed login and sign-up pages from 446 domains. Furthermore, Durey et al. chose those 446 domains because they are more likely to collect sensitive personal information (e.g. sites for job searching or dating) or financial information (e.g. gambling, e-commerce); these sites are more likely to take care to prevent fraud. Our papers also differ in fingerprinting detection. We consider four primary browser APIs (Canvas, WebRTC, Canvas Font, and AudioContext fingerprinting), while

Table 6: Percent of websites that perform fingerprinting on various page categories, grouped by popularity according to the CrUX dataset.

	At Least 1 Page	Homepage	Login	Sign-up
Top 1K	25.75%	22.24%	14.73%	8.82%
1K-5K	19.01%	17.10%	8.36%	5.70%
5K-10K	16.30%	15.02%	6.63%	4.14%
10K-50K	12.61%	11.61%	5.42%	3.08%
50K-100K	10.60%	9.85%	4.57%	2.59%

Durey et al. include activity like accessing the `navigator` and `screen` objects.

We searched for the security organizations mentioned in the previous study in our own dataset. Notably, we observed that one of the most prominent domains, `sift.com`, also ranked among the top fingerprinting domains in our list. While other security companies (Nudata Security - `nudatasecurity.com`, Simility - `simility.com`) also appeared in our results, they were observed on only a limited number of pages.

Lin et al. measured how many of the Alexa top 20K sites perform fingerprinting on login pages [30]. They found 11.5K login pages for the Alexa top 20K (5,736 for the top 10K and 5,791 for rank 10K-20K), and found that the majority check basic device attributes such as the navigator and window objects. Using the criteria for fingerprinting from FP-Inspector [23] (i.e. the presence of canvas, canvas font, WebRTC, and AudioContext fingerprinting), Lin et al. found that 18% of login pages perform fingerprinting.[10]

There are a few potential explanations for why we find a lower rate of fingerprinting than Lin et al. [30]. First, we study a larger pool of sites (100K) than Lin et al (20K). Prior work has found that more popular websites are more likely to perform fingerprinting [23], so our fingerprinting rate will naturally decline as we include less popular websites. We present the rate of fingerprinting for each site-rank bucket in Table 6. As this table shows, we are able to confirm the finding that more popular sites are more likely to fingerprint. Interestingly, we find that this trend holds for each page category.

[10]Lin et al. find that the majority of the login pages they identified perform some form of fingerprinting such as accessing the navigator or window objects [30]. We use a more conservative definition of fingerprinting as explained in §3.2. Fortunately, Lin et al. describe the percentage of the login pages and homepages that perform each type of fingerprinting that we consider, and so we compare these rates. Lin et al. find that sites perform canvas fingerprinting on 2,133 of the 11,527 login pages they identified, yielding a fingerprinting rate of 18.50%.

4.4 Potential Usage for Anti-Fraud

It is difficult to infer the intent behind the use of fingerprinting. However, there are some clues that suggest sites may be using fingerprinting for anti-fraud purposes.

As indicated in Table 4, the most common fingerprinting domain on authentication pages, `signifyd.com`, is associated with a fraud prevention company. Additionally, we observed another script from a well-known fraud prevention company, served on `sift.com`, appearing among the top-10 fingerprinting domains on authentication pages. This suggests that browser fingerprinting is employed for purposes beyond tracking on these pages.

Through a small-scale manual analysis, we found that disabling fingerprinting scripts broke login functionality. Due to the scalability challenges, we conducted a manual inspection of 30 websites where fingerprinting attempts were detected on login pages. We visited these websites using the JShelter browser extension [27], which blocks fingerprinting scripts. Our findings revealed that the login functionality on `deezer.com` and `hepsiburada.com` (both of which have 12M+ active users) experienced disruptions when fingerprinting scripts were blocked, as depicted in Appendix 6. Specifically, on Deezer's login page, the login button became non-responsive, preventing users from signing in. Similarly, on Hepsiburada, attempting to log in from the homepage resulted in the login page failing to load. However, Hepsiburada does not use fingerprinting to detect cookie hijacking, as we were able to replay cookies and log into our account from a different browser.

It is also possible for websites to use fingerprinting for both anti-fraud and advertising (including ad-driven tracking) simultaneously. For instance, a widely used third-party script on 7% of authentication pages is from the aforementioned websites `sift.com` and `siftscience.com`; these are associated with a single fraud prevention company [46]. However, when we manually examined this script, we noticed that the users' fingerprints were sent to `hexagon-analytics.com`, which is controlled by the analytics company Hexagon Data [21].

5. Limitations

Like other research in web measurement, our study may be limited in terms of representativeness and coverage. For instance, websites may identify our crawler as an automated bot and treat it differently than genuine traffic. Although we

depend on TRC's anti-bot measures [6], which provide some mitigation against bot detection, we recognize that their effectiveness may be limited [56].

The applicability and efficacy of our fingerprinting approach may also be limited. We rely on the technique developed by Englehardt et al. [23], which intentionally focuses on only four browser APIs; this technique achieves high precision but low recall, which was appropriate as we wanted to analyze high-confidence fingerprinting attempts on authentication pages.

Finally, our link detection method exclusively considers hyperlinks represented by the `<a>` element. However, certain login and sign-up forms may only become visible when triggered by user interaction with an HTML element. Our methodology may overlook these types of form. In addition, we do not attempt to fill out forms, and so we may miss additional multi-step authentication forms.

6. Conclusion

Browser fingerprinting, which is often associated with online tracking, is also sometimes used for user security by preventing account breaches, detecting bots, and thwarting cookie hijacking. Understanding the security implications of fingerprinting is crucial as mitigations are implemented. Our study fills a gap by examining the security aspects of fingerprinting at a large scale, particularly on login and sign-up pages. We introduced a highly accurate ML model (96-98% recall and precision) that successfully identified login (52,805) and sign-up (21,988) pages among 100,000 popular websites. Fingerprinting attempts were detected on 9.2% of these pages, slightly higher than the 8.9% rate across all pages. Notably, some of the top invasive fingerprinting scripts on login and sign-up pages were associated with fraud prevention companies, indicating diverse motivations for fingerprinting. We also show that when websites fingerprint on authentication pages, they are far more likely to only fingerprint on their login page. However, when they fingerprint on both login and sign-up pages, they almost always use the same set of fingerprinting scripts. Our contributions include an empirical web measurement study, a precise machine learning model, and practical tools for detecting login and sign-up pages. These findings show the multifaceted role of browser fingerprinting beyond user tracking.

Acknowledgements

We thank our anonymous reviewers for their time and invaluable feedback. We are also grateful to Alex C. Snoeren, Gunes Acar, Audrey Randall, Katherine Izhikevich, Miro Haller, Nina Taft, Enrico Bacis, Deepak Ravichandran and Hamza Harkous for their feedback.

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-2038238. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] Ayush Agarwal, Sioli O'Connell, Jason Kim, Shaked Yehezkel, Daniel Genkin, Eyal Ronen, and Yuval Yarom. Spook. js: Attacking Chrome strict site isolation via speculative execution. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 699–715. IEEE, 2022.

[2] Suood Al Roomi and Frank Li. A Large–Scale Measurement of Website Login Policies. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 2061–2078, 2023.

[3] Furkan Alaca and Paul C. Van Oorschot. Device fingerprinting for augmenting web authentication: classification and analysis of methods. In *Proceedings of the 32nd annual conference on computer security applications*, pages 289–301, 2016.

[4] Nampoina Andriamilanto, Tristan Allard, Gaëtan Le Guelvouit, and Alexandre Garel. A large-scale empirical analysis of browser fingerprints properties for web authentication. *ACM Transactions on the Web (TWEB)*, 16(1):1–62, 2021.

[5] server proto. `https://source.chromium.org/chromium/chromium/src/+/main:components/autofill/core/browser/proto/server.proto;drc=cefcacc55347e318a439f3112d96a1c73cfba56c`. [Online; accessed 30. Aug. 2023].

[6] Konrad Dzwinel et al. Brad Slayter, Sam Macbeth. DuckDuckGo Tracker Radar Collector. `https://github.com/duckduckgo/tracker-radar-collector`, 2021. [Online; accessed 01. Jan. 2023].

[7] The best privacy online. `https://brave.com/`. [Online; accessed 18. Sept. 2023].

[8] Joe DeBlasio, Stefan Savage, Geoffrey M. Voelker, and Alex C. Snoeren. Tripwire: Inferring internet site compromise. In *Proceedings of the 2017 Internet Measurement Conference*, pages 341–354, 2017.

[9] Yana Dimova, Tom Van Goethem, and Wouter Joosen. Everybody's Looking for SSOmething: A large-scale evaluation on the privacy of OAuth authentication on the web. *Proceedings on Privacy Enhancing Technologies*, 4:452–467, 2023.

[10] Disconnect. Disconnect Tracking Protection. `https://github.com/disconnectme/disconnect-tracking-protection`, 2023. [Online; accessed 11. Jun. 2023].

[11] Freedom from tracking. `https://disconnect.me/`. [Online; accessed 18. Sept. 2023].

[12] Kostas Drakonakis, Sotiris Ioannidis, and Jason Polakis. The Cookie Hunter: Automated Black-Box Auditing for Web Authentication and Authorization Flaws. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, pages 1953–1970, New York, NY, USA, 2020. Association for Computing Machinery.

[13] Antonin Durey, Pierre Laperdrix, Walter Rudametkin, and Romain Rouvoy. FP-Redemption: Studying browser fingerprinting adoption for the sake of web security. In *Detection of Intrusions and Malware, and Vulnerability Assessment: 18th International Conference, DIMVA 2021, Virtual Event, July 14–16, 2021, Proceedings 18*, pages 237–257. Springer, 2021.

[14] Early browser API accesses and function calls are missed. `https://github.com/duckduckgo/tracker-radar-collector/issues/77`, 2023. [Online; accessed 29. Jul. 2023].

[15] Steven Englehardt and Arvind Narayanan. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 1388–1401, 2016.

[16] Daniel Hertenstein Erik Rose. Fathom. `https://github.com/mozilla/fathom`, 2017. [Online; accessed 21. Aug. 2023].

[17] Daniel Hertenstein Erik Rose. Ruleset Zoo. `https://mozilla.github.io/fathom/zoo.html`, 2017. [Online; accessed 21. Aug. 2023].

[18] General Data Protection Regulation (GDPR). [Online; accessed 05. Sept. 2023].

[19] Mohammad Ghasemisharif, Amrutha Ramesh, Stephen Checkoway, Chris Kanich, and Jason Polakis. O Single Sign-Off, Where Art Thou? An Empirical Analysis of Single Sign-On Account Hijacking and Session Management on the Web. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 1475–1492, 2018.

[20] Tom Van Goethem, Wout Scheepers, Davy Preuveneers, and Wouter Joosen. Accelerometer-based device fingerprinting for multi-factor mobile authentication. In *International Symposium on Engineering Secure Software and Systems*, pages 106–121. Springer, 2016.

[21] Optimize.Personalize.Monetize. `https://www.hexagondata.com/en/services-marketer/`, 2023. [Online; accessed 30. Aug. 2023].

[22] intent.ly. `https://intent.ly/en/`. [Online; accessed 30. Aug. 2023].

[23] Umar Iqbal, Steven Englehardt, and Zubair Shafiq. Fingerprinting the fingerprinters: Learning to detect browser fingerprinting behaviors. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1143–1161. IEEE, 2021.

[24] Nikhil Jha, Martino Trevisan, Luca Vassio, and Marco Mellia. The internet with privacy policies: Measuring the web upon consent. *ACM Transactions on the Web (TWEB)*, 16(3):1–24, 2022.

[25] Barry Pollard Johannes Henkel. Adding Rank Magnitude to the CrUX Report in BigQuery. `https://developer.chrome.com/blog/crux-rank-magnitude/`, 2021-03-09. [Online; accessed 13. Jul. 2023].

[26] Hugo Jonker, Stefan Karsch, Benjamin Krumnow, and Marc Sleegers. Shepherd: a Generic Approach to Automating Website Login. In *Workshop on Measurements, Attacks, and Defenses for the Web (MADWeb)*. Reston: Internet Society, 2020.

[27] JShelter. `https://jshelter.org/`. [Online; accessed 18. Sept. 2023].

[28] Pierre Laperdrix, Benoit Baudry, and Vikas Mishra. FPRandom: Randomizing core browser objects to break advanced device fingerprinting techniques. In *Engineering Secure Software and Systems: 9th International Symposium, ESSoS 2017, Bonn, Germany, July 3-5, 2017, Proceedings 9*, pages 97–114. Springer, 2017.

[29] Xu Lin, Panagiotis Ilia, and Jason Polakis. Fill in the Blanks: Empirical Analysis of the Privacy Threats of Browser Form Autofill. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, pages 507–519, New York, NY, USA, 2020. Association for Computing Machinery.

[30] Xu Lin, Panagiotis Ilia, Saumya Solanki, and Jason Polakis. Phish in sheep's clothing: Exploring the authentication pitfalls of browser fingerprinting. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1651–1668, Boston, MA, August 2022. USENIX Association.

[31] Luka Lodrant. Designing a generic web forms crawler to enable legal compliance analysis of authentication sections. Master's thesis, ETH Zurich, 2022.

[32] Login Forms Ruleset. `https://mozilla.github.io/fathom/zoo/login.html`, 2023. [Online; accessed 21. Aug. 2023].

[33] nikhiljha95 Martino Trevisan, Antonino Musmeci. Priv-Accept. `https://github.com/marty90/priv-accept`, 2022-04-13. [Online; accessed 13. Jul. 2023].

[34] Peter Kvitek Mathias Bynens. Chrome‘s Headless mode gets an upgrade: introducing headless=new. `https://developer.chrome.com/articles/new-headless/`, 2023. [Online; accessed 21. Aug. 2023].

[35] Ariana Mirian, Nikunj Bhagat, Caitlin Sadowski, Adrienne Porter Felt, Stefan Savage, and Geoffrey M. Voelker. Web Feature Deprecation: A Case Study for Chrome. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pages 302–311, 2019.

[36] Mozilla. Firefox rolls out Total Cookie Protection by default to all users worldwide. `https://blog.mozilla.org/en/mozilla/firefox-rolls-out-total-cookie-protection-by-default-to-all-users-worldwide/`, 2022. [Online; accessed 30. Aug. 2023].

[37] Nick Nikiforakis, Wouter Joosen, and Benjamin Livshits. Privaricator: Deceiving fingerprinters with little white lies. In *Proceedings of the 24th International Conference on World Wide Web*, pages 820–830, 2015.

[38] Sean Oesch and Scott Ruoti. That was then, this is now: A security evaluation of password generation, storage, and autofill in browser-based password managers. In *Proceedings of the 29th USENIX Conference on Security Symposium*, pages 2165–2182, 2020.

[39] Davy Preuveneers and Wouter Joosen. Smartauth: dynamic context fingerprinting for continuous user authentication. In *Proceedings of the 30th annual ACM symposium on applied computing*, pages 2185–2191, 2015.

[40] Privacy Badger is a browser extension that automatically learns to block invisible trackers. `https://privacybadger.org/`. [Online; accessed 18. Sept. 2023].

[41] Jannis Rautenstrauch, Giancarlo Pellegrino, and Ben Stock. The Leaky Web: Automated Discovery of Cross-Site Information Leaks in Browsers and the Web. In *2023 IEEE Symposium on Security and Privacy (SP)*, 2023.

[42] Walter Rudametkin. *Improving the Security and Privacy of the Web through Browser Fingerprinting.* PhD thesis, Université de Lille, 2021.

[43] Kimberly Ruth, Aurore Fass, Jonathan Azose, Mark Pearson, Emma Thomas, Caitlin Sadowski, and Zakir Durumeric. A world wide view of browsing the world wide web. In *Proceedings of the 22nd ACM Internet Measurement Conference*, pages 317–336, 2022.

[44] Kimberly Ruth, Deepak Kumar, Brandon Wang, Luke Valenta, and Zakir Durumeric. Toppling top lists: Evaluating the accuracy of popular website lists. In *Proceedings of the 22nd ACM Internet Measurement Conference*, pages 374–387, 2022.

[45] Justin Schuh. Building a more private web: A path towards making third party cookies obsolete. `https://blog.chromium.org/2020/01/building-more-private-web-path-towards.html`, 2020. [Online; accessed 30. Aug. 2023].

[46] Take control of payment fraud. `https://sift.com/`, 2023. [Online; accessed 30. Aug. 2023].

[47] SignUpFormRuleset.sys.mjs. `https://searchfox.org/mozilla-central/source/toolkit/components/passwordmgr/SignUpFormRuleset.sys.mjs`, 2023. [Online; accessed 21. Aug. 2023].

[48] TensorFlow. `https://www.tensorflow.org/`, 2015. [Online; accessed 21. Aug. 2023].

[49] Browse Privately. Explore Freely. `https://www.torproject.org/`. [Online; accessed 18. Sept. 2023].

[50] uBlock Origin - An efficient blocker for Chromium and Firefox. Fast and lean. `https://github.com/gorhill/uBlock`, 2023. [Online; accessed 10. Jul. 2023].

[51] Thomas Unger, Martin Mulazzani, Dominik Frühwirt, Markus Huber, Sebastian Schrittwieser, and Edgar Weippl. SHPF: Enhancing HTTP(S) Session Security with Browser Fingerprinting. In *2013 International Conference on Availability, Reliability and Security*, pages 255–261. IEEE, 2013.

[52] Steven Van Acker, Daniel Hausknecht, and Andrei Sabelfeld. Measuring login webpage security. In *Proceedings of the Symposium on Applied Computing*, pages 1753–1760, 2017.

[53] Maximilian Westers, Tobias Wich, Louis Jannett, Vladislav Mladenov, Christian Mainka, and Andreas Mayer. SSO-Monitor: Fully-Automatic Large-Scale Landscape, Security, and Privacy Analyses of Single Sign-On in the Wild. *arXiv preprint arXiv:2302.01024*, 2023.

[54] John Wilander. Full Third-Party Cookie Blocking and More. `https://webkit.org/blog/10218/full-third-party-cookie-blocking-and-more/`, 2020. [Online; accessed 30. Aug. 2023].

[55] Shujiang Wu, Pengfei Sun, Yao Zhao, and Yinzhi Cao. Him of many faces: Characterizing billion-scale adversarial and benign browser fingerprints on commercial websites. In *30th Annual Network and Distributed System Security Symposium, NDSS*, 2023.

[56] David Zeber, Sarah Bird, Camila Oliveira, Walter Rudametkin, Ilana Segall, Fredrik Wollsén, and Martin Lopatka. The representativeness of automated web crawls as a surrogate for human browsing. In *Proceedings of The Web Conference 2020*, pages 167–178, 2020.

[57] Yuchen Zhou and David Evans. SSOScan: Automated Testing of Web Applications for Single Sign-on Vulnerabilities. In *Proceedings of the 23rd USENIX Conference on Security Symposium*, SEC'14, pages 495–510, USA, 2014. USENIX Association.

Appendix

Login & Sign-up Page Features

1. **Form Features**
   (a) hasLoginKeywordsInFormAttributes
   (b) hasRegisterKeywordsInFormAttributes

(c) hasNewsletterKeywordsInFormAttributes
(d) hasForgotKeywordsInFormAttributes

2. **Button Features**

(a) hasLoginKeywordsInButtonAttributes
(b) hasLoginKeywordsInAttributesOnAForm
(c) hasLoginKeywordsInTextContent
(d) hasLoginKeywordsInTextContentOnAForm
(e) hasRegisterKeywordsInAttributes
(f) hasRegisterKeywordsInAttributesOnAForm
(g) hasRegisterKeywordsInTextContent
(h) hasRegisterKeywordsInTextContentOnAForm
(i) hasNewsletterKeywordsInAttributes
(j) hasNewsletterKeywordsInAttributesOnAForm
(k) hasNewsletterKeywordsInTextContent
(l) hasNewsletterKeywordsInTextContentOnAForm
(m) hasNextKeywordsInAttributes
(n) hasNextKeywordsInAttributesOnAForm
(o) hasNextKeywordsInTextContent
(p) hasNextKeywordsInTextContentOnAForm
(q) hasForgotKeywordsInAttributes
(r) hasForgotKeywordsInAttributesOnAForm
(s) hasForgotKeywordsInTextContent
(t) hasForgotKeywordsInTextContentOnAForm
(u) hasNextButtonCloseToUsername
(v) hasNextButtonCloseToUsernameOnAForm
(w) hasLoginButtonCloseToUsername
(x) hasLoginButtonCloseToUsernameOnAForm
(y) hasSignupButtonCloseToUsername
(z) hasSignupButtonCloseToUsernameOnAForm

3. **Anchor Features**

(a) hasForgotKeywordsInAttributes
(b) hasForgotKeywordsInAttributesOnAForm

(c) hasForgotKeywordsInTextContent
(d) hasForgotKeywordsInTextContentOnAForm

4. **Label Features**

(a) hasRememberMeKeywordsInAttributes
(b) hasRememberMeKeywordsInAttributesOnAForm
(c) hasRememberMeKeywordsInTextContent
(d) hasRememberMeKeywordsInTextContentOnAForm
(e) hasNewsletterKeywordsInAttributes
(f) hasNewsletterKeywordsInAttributesOnAForm
(g) hasNewsletterKeywordsInTextContent
(h) hasNewsletterKeywordsInTextContentOnAForm

5. **Header Features**

(a) hasLoginKeywordsInAttributes
(b) hasLoginKeywordsInAttributesOnAForm
(c) hasLoginKeywordsInTextContent
(d) hasLoginKeywordsInTextContentOnAForm
(e) hasRegisterKeywordsInAttributes
(f) hasRegisterKeywordsInAttributesOnAForm
(g) hasRegisterKeywordsInTextContent
(h) hasRegisterKeywordsInTextContentOnAForm
(i) hasNewsletterKeywordsInAttributes
(j) hasNewsletterKeywordsInAttributesOnAForm
(k) hasNewsletterKeywordsInTextContent
(l) hasNewsletterKeywordsInTextContentOnAForm
(m) hasForgotKeywordsInAttributes
(n) hasForgotKeywordsInAttributesOnAForm
(o) hasForgotKeywordsInTextContent
(p) hasForgotKeywordsInTextContentOnAForm

6. **Checkbox Features**

(a) hasNewsletterKeywordsInAttributes
(b) hasNewsletterKeywordsInAttributesOnAForm

(c) hasRememberMeKeywordsInAttributes
(d) hasRememberMeKeywordsInAttributesOnAForm

7. **Input Features**

(a) hasLoginKeywordsInAttributes
(b) hasLoginKeywordsInAttributesOnAForm
(c) hasRegisterKeywordsInAttributes
(d) hasRegisterKeywordsInAttributesOnAForm
(e) hasNewsletterKeywordsInAttributes
(f) hasNewsletterKeywordsInAttributesOnAForm
(g) hasAnyEmail
(h) hasAnyUsername
(i) hasAnyPEmailOnAForm
(j) hasAnyUsernameOnAForm
(k) hasMultipleInputs

8. **Password Features**

(a) hasLabelOrAriaLabelOrPlaceholderContainsConfirm:
(b) hasLabelOrAriaLabelOrPlaceholderContainsConfirmOnAForm
(c) hasLabelOrAriaLabelOrPlaceholderContainsCurrent
(d) hasLabelOrAriaLabelOrPlaceholderContainsCurrentOnAForm
(e) hasLabelOrAriaLabelOrPlaceholderContainsNew
(f) hasLabelOrAriaLabelOrPlaceholderContainsNewOnAForm
(g) hasAnyPasswordField
(h) hasMultiplePasswordFields

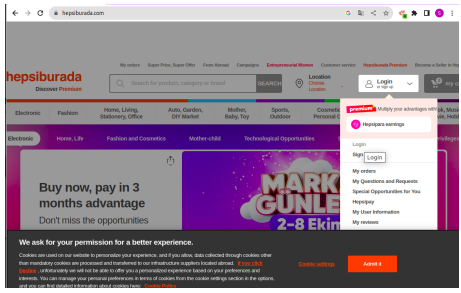
9. **Div Features**

(a) hasAlreadyHaveAnAccountKeywords
(b) hasAlreadyHaveAnAccountKeywordsOnAForm
(c) hasDontHaveAnAccountKeywords
(d) hasDontHaveAnAccountKeywordsOnAForm
(e) hasNewsletterKeywords

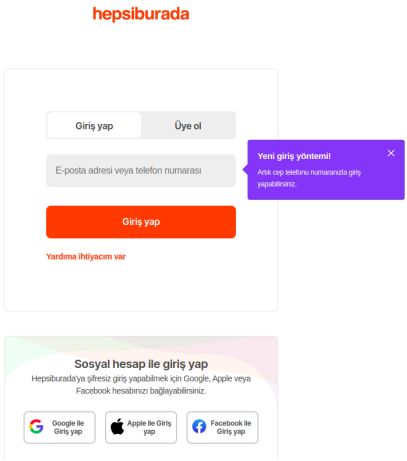
10. **URL Features**

(a) hasResetKeywordsInURL
(b) hasNewsletterKeywordsInURL

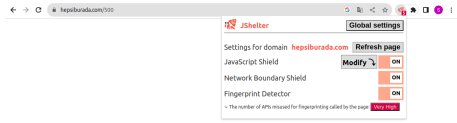
Web Page Behaviors When Fingerprinting Scripts are blocked



(a) The login form of Hepsiburada with JShelter deactivated.

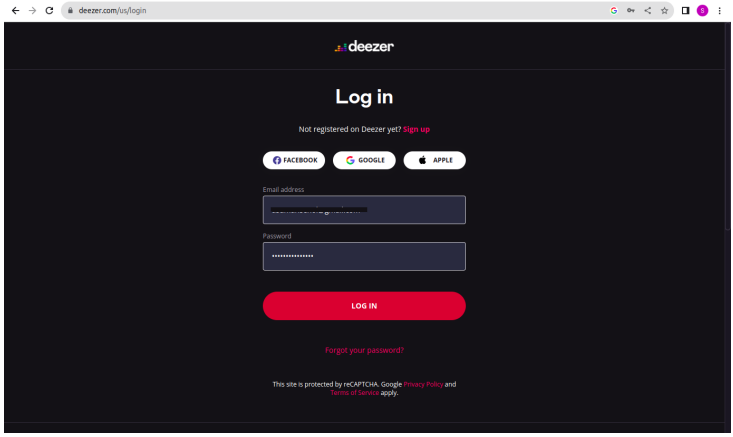


(b) Upon clicking the login button in Figure ⓐ, users are directed to the Hepsiburada login page with JShelter disabled.

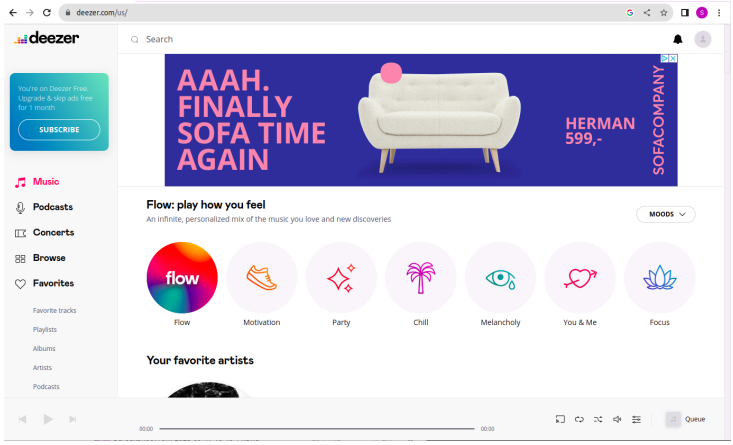


(c) The Hepsiburada login page fails to load when the extension is active, and the extension has disabled fingerprinting scripts.

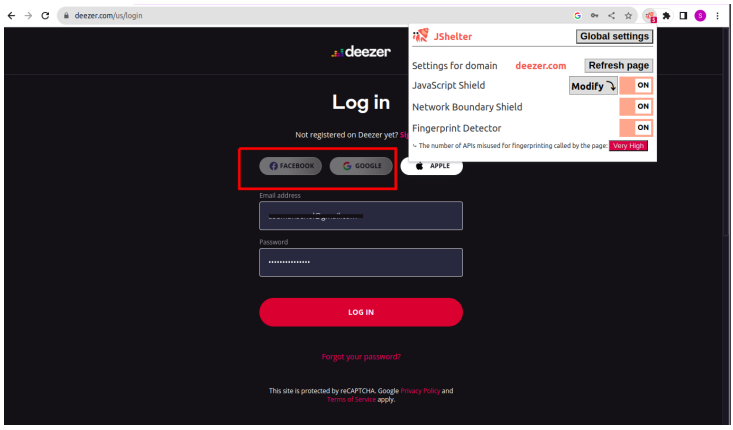
Figure 5: The Hepsiburada login page fails to load when the extension is active, and the extension has disabled fingerprinting scripts.



(a) Deezer's login form when JShelter was disabled.



(b) Deezer's homepage when a user is signed in and fingerprinting scripts are not blocked



(c) Hepsiburada's Login Page Behavior with JShelter Activation

Figure 6: Effect of fingerprinting script blocking on Deezer's login page. Deezer's login form behaves abnormally when JShelter is activated and blocks fingerprinting scripts ⓒ, causing disruptions in the functionality of login buttons.

Fingerprinting Detection Heuristics

Here are the heuristics employed in this study to identify fingerprinting attempts. Initially proposed by Englehardt and Narayanan [15], these heuristics were subsequently refined by Iqbal et al. [23].

Canvas Fingerprinting: A script is classified as a canvas fingerprinting script based on the following criteria:

1. The script uses the canvas element to write text using methods such as `fillText` or `strokeText` and applies styling using methods like `fillStyle` or `strokeStyle` within the rendering context.
2. The script invokes the `toDataURL` method to extract the canvas image.
3. The script does not make use of the `save`, `restore`, or `addEventListener` methods on the canvas element.

WebRTC Fingerprinting: A script is identified as a WebRTC fingerprinting script according to the following conditions:

1. The script invokes methods like `createDataChannel` or `createOffer` within the WebRTC peer connection.
2. The script calls methods such as `onicecandidate` or `localDescription` within the WebRTC peer connection.

Canvas Font Fingerprinting: A script qualifies as a canvas font fingerprinting script based on the following criteria:

1. The script sets the `font` property on a canvas element to utilize more than 20 different fonts.
2. The script invokes the `measureText` method of the rendering context more than 20 times.

AudioContext Fingerprinting: A script is identified as an AudioContext fingerprinting script according to the following conditions:

1. The script invokes any of the following methods within the audio context: `createOscillator`, `createDynamicsCompressor`, `destination`, `startRendering`, and `oncomplete`.

Bibliography

[1] Moti, Z., Senol, A., Bostani, H., Borgesius, F. Z., Moonsamy, V., Mathur, A., and Acar, G. Targeted and Troublesome: Tracking and Advertising on Children's Websites. In *2024 IEEE Symposium on Security and Privacy (SP)* (Los Alamitos, CA, USA, may 2024), IEEE Computer Society, pp. 117–117.

[2] Senol, A., and Acar, G. Unveiling the Impact of User-Agent Reduction and Client Hints: A Measurement Study. In *Proceedings of the 22nd Workshop on Privacy in the Electronic Society* (2023), pp. 91–106.

[3] Senol, A., Acar, G., Humbert, M., and Borgesius, F. Z. Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission. In *31st USENIX Security Symposium (USENIX Security 22)* (2022), pp. 1813–1830.

[4] Senol, A., Ukani, A., Cutler, D., and Bilogrevic, I. The Double Edged Sword: Identifying Authentication Pages and their Fingerprinting Behavior. In *The Web Conference (WWW), 2024* (2024).

Curriculum

Asuman Şenol, originally from Ankara, completed her studies in Mathematics at the Middle East Technical University and later earned her Master's degree in Cyber Security at the same university. Following five years of experience as a software engineer in the industry, she started her doctoral journey by joining COSIC in 2020.



FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF ELECTRICAL ENGINEERING
COSIC
Kasteelpark Arenberg 10, bus 2452
B-3001 Leuven
asuman.senol@esat.kuleuven.be
http://www.esat.kuleuven.be/cosic