A Comparative Analysis of Third-Party Script Behaviour in Consent-Based and Implicit Web Tracking

Chongwen Ma
University of Nottingham
Ningbo, China
scxcm1@nottingham.edu.cn

Yuan Cheng
University of Nottingham
Ningbo, China
yuan.cheng@nottingham.edu.cn

Abstract

Modern websites rely heavily on third-party scripts for advertising and analytics, and they are required to obtain user consent before deploying non-essential tracking. However, it remains unclear whether websites that provide explicit consent interfaces actually behave differently from those that track users by default. This paper presents a comprehensive client-side measurement of third-party script behaviour under two tracking paradigms: consent-based tracking (CBT), which requires explicit user choice for executing non-essential scripts, and implicit tracking, where no such choice is given. We developed an automated measurement pipeline using Chromium to crawl a set of websites sourced from Tranco, classify tracking paradigms based on observable interfaces, and record third-party script activity during fixed pre- and post-interaction periods. Our analysis examines the scale and timing of script loading, dependency structure, and runtime execution behaviour. We specifically analyse the "no-consent" state, comparing implicit tracking with CBT sites after a scripted "Reject" action. Our results find that CBT sites load more third-party scripts and activate them earlier during the initial page load than implicit-tracking sites, often before users can even interact with a consent banner. While the overall number of third-party scripts is similar after rejection, a substantial subset of CBT sites exhibits a larger third-party footprint, more intensive dynamic code execution, and higher runtime error rates than their implicit-tracking counterparts. These findings reveal a gap between interface-level consent flows and script-level behaviour, offering practical implications for developers and regulators.

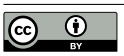
CCS Concepts

• **Security and privacy → Web application security**; **Privacy protections**; *Browser security*.

Keywords

Web Tracking, Consent Management, Web Privacy

ACM Reference Format:
Chongwen Ma and Yuan Cheng. 2026. A Comparative Analysis of Third-Party Script Behaviour in Consent-Based and Implicit Web Tracking. In *Proceedings of the Sixteenth ACM Conference on Data and Application Security and Privacy (CODASPY '26), June 23–25, 2026, Frankfurt am Main, Germany.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3800506.3803512



CODASPY '26, Frankfurt am Main, Germany

ACM ISBN 979-8-4007-2562-3/2026/06
https://doi.org/10.1145/3800506.3803512

1 Introduction

Modern web services rely heavily on third-party scripts to support advertising, analytics, and personalisation [8]. In response to growing privacy concerns, regulations such as the General Data Protection Regulation (GDPR) and the ePrivacy Directive require websites to obtain user consent before deploying non-essential tracking [9, 29].

In practice, however, the presence of a consent interface does not necessarily result in meaningful changes to the tracking logic executed in users' browsers. Consent banners and consent management platforms (CMPs) may give users an "illusion of control" [22] when the underlying tracking behaviour remains largely unchanged. Prior studies have documented such inconsistencies, including cases where tracking continues despite users' rejection, and scenarios where user interaction triggers the dynamic loading of additional third-party scripts without any explicit consent decision [4, 19].

These observations highlight a clear disconnect between what a website's compliance interface communicates and what its client-side tracking code actually does. Much of the existing regulatory and academic literature focuses on the interface layer, evaluating legal compliance based on features such as banner presence, wording, and layout. Meanwhile, security research has examined the risks introduced by third-party scripts, including vulnerable JavaScript dependencies and increased exposure to supply-chain attacks [14, 17]. Yet, there remains a lack of systematic, comparative analysis regarding how third-party scripts technically behave under different tracking paradigms.

This gap leads us to our core research question: Do websites that deploy explicit consent interfaces differ fundamentally from those relying on implicit tracking in terms of the technical behaviour and structure of their third-party scripts?

To address this question, we conduct a comparative measurement study that distinguishes between two tracking paradigms based on observable interface behaviour:

- **Consent-based tracking (CBT)**: Non-essential third-party scripts are loaded only after an explicit consent decision is made, typically through a banner or similar interface.
- **Implicit tracking**: Non-essential third-party scripts are loaded without any explicit consent decision.

We specifically analyse the *no-consent* state, comparing sites where users are not presented with a consent interface at all (implicit tracking) against sites where users explicitly reject non-essential tracking (CBT-Reject).

We compare these paradigms by examining how third-party scripts behave during controlled, automated visits that follow the same interaction sequences across sites. In particular, we contrast traces collected during a short window (the *pre-interaction* window)

before any consent choice is made with traces collected during a later window (the *post-interaction* window) after a scripted consent action such as "Accept" or "Reject". This approach allows us to assess whether the scale and timing of third-party script loading align with what the consent interface promises. Cases in which tracking increases despite a nominal opt-out are treated as inconsistencies between the consent interface and the underlying technical behaviour.

Our analysis focuses on three fundamental aspects of third-party script behaviour:

- **Dynamic loading**: The scale and timing of third-party script loading relative to the consent event.
- **Dependency**: How third-party scripts load one another at runtime, measured by the depth, out-degree, and cross-origin edge ratio of the script dependency graph.
- **Execution behaviour**: How third-party scripts behave once loaded, including dynamic code execution, deferred execution, and runtime error rates.

Building on these dimensions, we investigate the following research questions:

- **RQ1:** How do the scale and timing of third-party script loading differ between consent-based tracking sites and implicit tracking sites when subjected to the same scripted browsing flow?
- **RQ2:** How do the dependency structures and execution behaviours of third-party scripts differ between consent-based tracking sites and implicit tracking sites?

In this paper, we present a reproducible, browser-instrumented comparative study of third-party script behaviour on consent-based tracking and implicit tracking sites. Specifically, we make the following contributions:

(1) We define two observable tracking paradigms, CBT and implicit tracking, grounded solely in interface behaviour, and demonstrate that this classification can be applied reliably through automated banner detection with targeted human validation.
(2) We implement an instrumented browser-based measurement pipeline that applies the same scripted visit to each site and records third-party script activity in fixed pre- and post-interaction windows.
(3) We introduce a unified measurement framework that quantifies third-party script scale, timing, dependency, and execution behaviour, and apply it to conduct a comparative analysis of CBT and implicit tracking sites, with a focus on the *no-consent* state.
(4) We show that interface-observable consent choices (in particular, "Reject") do not reliably reduce third-party script activity and discuss the implications for evaluating tracking control beyond interface-level signals.
(5) We release our measurement framework and dataset to support reproducibility.[1]

[1]Available at: https://github.com/Yellowpeach0228/consent-vs-implicit-tracking

The remainder of this paper is organised as follows. Section 2 reviews prior work on web tracking, consent interfaces, and client-side risks of third-party scripts. Section 3 describes our measurement design, including sampling, tracking-paradigm labelling, interaction scenarios, browser instrumentation, and metric definitions. Section 4 presents the empirical results for our two research questions. Section 5 discusses the implications and limitations of our findings. Section 6 concludes the paper.

2 Related Work

This section reviews prior work on web tracking, consent paradigms, and client-side risks of third-party scripts.

2.1 Web Tracking and Consent Interfaces

Prior work has established that web tracking can be studied through concrete, observable client-side signals rather than interface claims or policy text. Bujlow et al. [5] systematised tracking techniques across server-side identifiers, client-side storage, cache-based tracking, and hybrid approaches. Roesner et al. [24] classified third-party trackers by their observable interactions with pages and users, while Englehardt and Narayanan [8] demonstrated the feasibility of reproducible audits using OpenWPM to measure third-party requests, fingerprinting scripts, and cookie-synchronisation patterns. More recent work further enriched these signals with script-level features and longitudinal traces, using HTTP(S) traffic, script inclusions, and storage operations to characterise tracker behaviour [1, 16, 21]. Together, these studies indicate that tracking intensity and structure can be measured through network and execution observables. However, they typically do not differentiate measurements by consent interactions or compare script ecosystems across distinct consent paradigms under the same scripted browsing flow.

Despite regulatory changes intended to enforce explicit consent, a persistent gap remains between the promises made by consent interfaces and their client-side implementation. The GDPR and the ePrivacy Directive formalised a shift from implicit to explicit consent for non-essential processing [9, 29]. Empirical studies of consent notices, however, indicate that tracking-by-default practices remain widespread. Utz et al. [31] documented banners that rely on vague wording, passive acknowledgement, or mere presence to legitimise tracking, while Degeling et al. [7] reported substantial regional and implementation differences in the prevalence, wording, and default settings of such banners.

Recent literature treats consent as an observable event and directly compares tracking behaviour before and after a user decision. Matte et al. [19] showed that non-essential cookies and tracking requests often persist even when users select restrictive options. Tang et al. [28] reported similar patterns across jurisdictions using phase-based designs that separate pre-consent, post-consent, and post-rejection states. Tóth et al. [30] demonstrated that CMPs commonly use dark patterns and framing strategies that steer both users and site operators toward permissive configurations. Bollinger et al. [3] developed automated methods to detect GDPR consent violations at scale using machine learning classifiers, while Zimmeck et al. [35] examined how opt-out mechanisms interact with tracking infrastructure, finding that many sites fail to honour user preferences consistently.

These studies collectively show that explicit consent interfaces and tracking-by-default practices continue to coexist on the web. However, most prior work models consent primarily as a per-visit interaction state (e.g., pre-consent, post-consent, post-reject) and reports behavioural changes within a site across phases. In contrast, our study uses interface-observable behaviour as a reproducible site-level grouping variable to enable controlled cross-site comparisons under an identical scripted visit, and examines how third-party script ecosystems differ between consent-based and implicit tracking sites.

2.2 Third-Party Scripts and Client-Side Security Risks

A complementary line of work connects third-party scripting to client-side security risk. Klein et al. [12] showed that selecting "Accept all" in cookie banners increases the number of dynamically loaded third-party scripts, and reported correlations with weaker deployment of Content Security Policy (CSP) and Subresource Integrity (SRI), connecting permissive consent flows to an expanded client-side attack surface. Beyond individual consent events, Lauinger et al. [14] found that production sites often rely on outdated JavaScript libraries for extended periods, highlighting how third-party dependencies introduce structural supply-chain risks. Sprecher et al. [27] systematised these concerns by arguing that third-party script loading follows an "all-or-nothing" trust model: once a provider is allowed to execute code, deep dependency chains and dense cross-origin relationships can extend trust far beyond the first-party origin, undermining the principle of least privilege. These findings motivate our analysis, which examines not only the quantity of third-party scripts but also how they are organised into dependency structures and how they behave at runtime.

Complementary work has also developed automated frameworks for detecting consent interfaces and auditing tracking behaviour. Researchers such as Matte et al. [19], Santos et al. [11], and Bouhoula et al. [4] combined consent-interface recognition with template- or learning-based methods to detect and categorise cookie banners, and in some cases triggered banner controls to relate interface designs to observable tracking outcomes. Koch et al. [13] applied similar techniques in mobile environments via automated instrumentation. These systems demonstrate that consent interfaces and client-side tracking signals can be detected and analysed automatically, enabling reproducible audits of consent implementations. In our study, we leverage these detection techniques to classify sites into two paradigms (consent-based tracking vs. implicit tracking) and then compare their third-party script execution under a standardised scripted visit.

Finally, while major browsers have introduced platform-level tracking protections (e.g., restrictions on third-party storage or tracker blocking), these mechanisms are largely independent of consent interfaces. Accordingly, our analysis focuses on consent mechanisms and their association with third-party script behaviour rather than browser-enforced protections.

3 Methodology

This section describes our measurement design for comparing third-party script behaviour under consent-based and implicit tracking.

3.1 Overview

Our measurement pipeline consists of six stages, as summarised in Figure 1:

(1) **Sampling**: Retrieving a ranked list of popular domains from the Tranco dataset,
(2) **Filtering**: Filtering domains to obtain sites that are reachable and suitable for measurement,
(3) **Labelling**: Assigning sites to tracking paradigms based on observable consent-interface behaviour,
(4) **Scripted Visits**: Visiting each site using an automated browser script under three interaction scenarios,
(5) **Data Collection**: Recording script-related network, dependency, and execution signals within fixed pre- and post-interaction windows,
(6) **Analysis**: Computing site-level metrics and performing statistical analyses for RQ1 (script loading scale and timing) and RQ2 (dependency structure and execution behaviour).

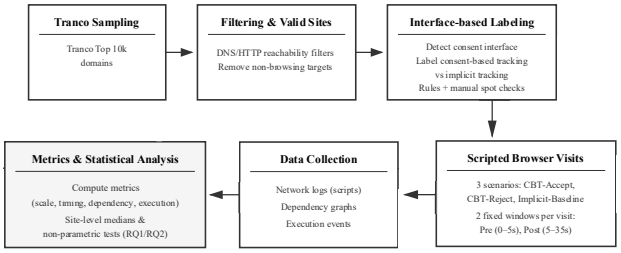


Figure 1: Overview of the measurement pipeline: sampling, filtering, labelling, scripted visits, data collection, and metric analysis.

3.2 Site Sampling and Tracking Paradigm Classification

3.2.1 Website Sampling. We begin with the top 10,000 domains obtained from a Tranco snapshot (2025-09-24 to 2025-10-24) [15]. Tranco aggregates multiple top-site lists to improve stability and reduce the potential for single-source manipulation that can occur with lists such as Alexa [25].

Before running behavioural measurements, we apply a reachability filter. For each Tranco domain, we first perform DNS resolution. If the apex domain has no A/AAAA record, we additionally try a `www`-prefixed form as a fallback. We then probe HTTP reachability using a small fallback chain over HTTPS/HTTP with HEAD or lightweight GET requests and a limited redirect budget. Domains that return any HTTP(S) response are retained (i.e., we do not require 2xx/3xx status codes).

The resulting set of reachable domains forms our study population. To ensure reliable behavioural measurements, we apply additional automated filtering stages during data collection. Specifically, we retain only domains for which our crawler successfully detects a consent interface (Section 3.2.2), completes the scripted interaction, and records valid network and script-execution traces. Domains that fail any of these stages are excluded.

Table 1: Dataset filtering pipeline and stage-level yield.

Stage	Filter	Domains Retained
0	Tranco Top-10k domains (initial pool)	10,000
1	DNS resolution	8,107
2	HTTP(S) reachability	6,372
3	Consent interface detection and paradigm classification	3,546
4	Successful scripted interaction and valid traces	3,104

After applying these filtering steps, the final dataset contains approximately 3,100 domains with valid measurement traces, as summarised in Table 1. To enable balanced comparison, we sample equal-sized subsets from each tracking paradigm for data collection.

3.2.2 Consent-Interface-Based Tracking Paradigm Classification. We classify sites as *CBT* or *implicit tracking* based on their consent interfaces. An automated Chromium client loads each landing page and monitors the rendered page for consent-related elements for up to 10 seconds. A site is labelled *CBT* if the interface (i) explicitly mentions cookies, tracking, or privacy, and (ii) offers at least one actionable option to accept non-essential tracking and one to reject or limit it to "necessary only" purposes (e.g., "Accept all" vs. "Reject all" or "Only necessary"). Conversely, we label a site as *implicit tracking* if no such interface appears within the monitoring window, or if only an informational notice is displayed without any actionable choices to accept or reject. These labels are derived solely from observable browser behaviour under our measurement conditions and are not intended as claims about legal compliance.

Our detection logic uses simple keyword- and structure-based rules inspired by prior studies on cookie-banner recognition and automated interaction [4, 12, 19]. To improve coverage across regions, the banner detection and interaction logic incorporates multilingual keyword matching for five languages that are widely used on the Internet and frequently encountered in our crawl: English, Chinese, German, French, and Spanish. This choice was intended to improve practical coverage rather than to provide exhaustive language support.

All measurements were conducted from a UK-based network using a Chromium client configured with an `en-US` locale and the `Europe/Amsterdam` timezone. Because measurements were performed from a single geographic location, banner presentation and tracking behaviour may vary across regions; we discuss this limitation further in Section 5.4.

Additionally, we manually inspect a small random sample of both CBT and implicit sites to confirm classification accuracy and make minor adjustments where necessary.

3.3 Interaction Scenarios and Observation Windows

We run three interaction scenarios with a fixed visit timeline: CBT-Accept, CBT-Reject, and Implicit-Baseline.

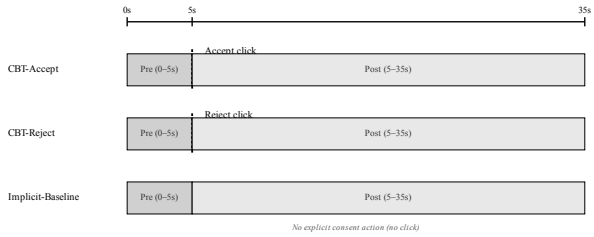


Figure 2: Interaction scenarios and observation windows.

As shown in Figure 2, 0 s marks the start of loading the landing page. Each visit is divided into a fixed pre-interaction window (0–5 s) and a 30-second post-interaction window. For CBT sites, the consent click is attempted immediately after the pre-window, and the post-interaction window is anchored to the completion timestamp of that click. For implicit tracking sites, no click is performed, and the post window starts at 5 s. This design keeps window durations comparable across scenarios while aligning the CBT post window with the scripted consent action.

3.3.1 Scripted Browsing Flow. All sites follow the same scripted visit procedure. The browser loads the landing page, remains idle for a fixed 5-second pre-window, performs a consent interaction when applicable, and then remains idle for a 30-second post-window. No additional browsing actions (e.g., scrolling or navigation) are performed, ensuring comparable measurements across sites.

3.3.2 CBT Sites: Accept and Reject Branches. For sites with a consent interface (referred to as CBT), we implement two distinct interaction branches. In *CBT-Accept*, the browser tries to click an "Accept all" (or equivalent) control following the pre-window. Conversely, in the *CBT-Reject* scenario, it attempts to click a "Reject all", "Only necessary", or a similar control instead. If the expected control is not detected at the scheduled interaction point or if the click operation cannot be completed, we log this as an interaction failure. Despite this failure, the process continues into the post-window without any further actions. Visits categorised as failed interactions are excluded from branch-level comparisons. Comparisons between CBT-Accept and CBT-Reject are made using a paired subset of sites where the corresponding click actions were successful in both branches.

3.3.3 Implicit Tracking Sites: Baseline Scenario. For sites classified as using implicit tracking, we establish a single *Implicit-Baseline* scenario. In this case, the browser uses the same client configuration but does not engage in any consent-related interactions. The post-window starts immediately after the 5-second pre-window elapses, aligning with the nominal interaction point used for the CBT branches. This baseline will later be compared against both CBT-Reject and CBT-Accept to assess differences between explicit rejection, implicit-by-default tracking, and explicit opt-in behaviour.

3.3.4 Pre- and Post-Interaction Windows. To ensure consistent comparisons, we analyse each visit using a fixed 0–5 s pre-window followed by a 30-second post-window on the 0 s visit timeline in Figure 2. The pre-window captures early activity immediately after

page loading begins. The post-window is anchored to the interaction routine: for CBT branches, it begins when the consent-click attempt completes (regardless of success) and runs for 30 seconds; for the implicit baseline, it starts at the 5 s mark and runs for 30 seconds. These values follow common web measurement practice [8, 34] and balance the need to cover delayed activity against the associated crawling costs. We denote the shared pre-window on CBT sites as **CBT-Pre**, and the post-windows as **CBT-Accept-Post**, **CBT-Reject-Post**, and **Implicit-Post** for clarity.

3.4 Browser Instrumentation and Data Collection

This section describes how we collect raw signals on third-party script loading and runtime behaviour during each automated site visit. Our goal is to capture comparable client-side activity across sites while keeping the crawl lightweight.

All measurements use an automated Chromium-based browser controlled via Puppeteer [23], following established practice in browser-based web measurement [2]. We chose Puppeteer because it enables deterministic scripted visits under a fixed browser version, provides isolation via fresh browsing contexts, and supports early instrumentation before page scripts execute.

Each visit runs in a fresh incognito context with cleared cookies, cache, and origin storage, using a fixed desktop User-Agent and viewport. We enforce a 15-second navigation timeout with up to three retries, limit per-host concurrency to reduce load on target sites, and set a 120-second per-site task timeout to accommodate network variability.

For each visit, we record three layers of client-side signals:

- **Network layer.** We log a timestamped trace of all HTTP(S) requests and responses for resources whose browser-reported type is script. For each request, we record the URL and the browser-reported initiator (when available); for each response, we record the response status code and use the `Content-Length` header (when present) as a size proxy. We classify each URL by eTLD+1 using the Public Suffix List and mark a request as third-party if its eTLD+1 differs from the page eTLD+1. We additionally tag a subset of third-party script requests as *tracker-category* using eTLD+1 matching against a predefined substring pattern list (40 substrings) [24].
- **Dependency layer.** We approximate script-to-script loading relationships by constructing a directed inclusion graph whose nodes are observed script URLs. An edge $A \rightarrow B$ indicates that script B was loaded as a consequence of A: when the browser provides an initiator for a request, we link the initiator URL to the requested script. To also cover cases where initiator attribution is missing or unreliable, we additionally record client-side evidence of dynamic script insertion (i.e., external script elements added to the DOM at runtime or script tags emitted via document-writing patterns) and add corresponding edges to the graph when a source script can be associated. The resulting graph is a best-effort approximation of inclusion structure rather than a complete causal dependency graph.
- **Execution layer.** We record timestamped runtime events by installing lightweight instrumentation before page scripts execute. The instrumentation captures three classes of signals: (i) dynamic code generation/execution, where scripts create or evaluate code at runtime; (ii) deferred scheduling, where scripts register callbacks intended to run later in the visit rather than immediately; and (iii) runtime failures observed at the page level, including uncaught exceptions and unhandled promise rejections. These event traces allow us to characterise execution behaviour at the visit level without attempting to recover full program semantics.

All events are timestamped and tagged with the site's tracking paradigm (CBT or implicit), the interaction scenario (CBT-Accept, CBT-Reject, Implicit-Baseline), and the observation window (pre- or post-interaction).

3.5 Metrics

We compute site-level metrics from the raw network, dependency, and execution logs collected in Section 3.4. Third-party script inclusion is measured based on network requests and script execution events observed during page load.

We group our metrics into four families: S (volume of third-party script activity), T (timing and pre/post change), D (dependency structure), and E (execution behaviour). Metrics are computed per site and per interaction scenario. Metrics that summarise activity within a window (S, D, and E) are computed separately for the pre- and post-interaction windows. Timing metrics are measured from the start of page loading within each scripted visit. Table 2 summarises all metrics and their relationships to RQ1 and RQ2.

3.5.1 Script-Loading Scale and Timing (RQ1). To address RQ1, we use the S and T metrics to quantify both the amount of third-party code a site loads and the timing of its appearance during the visit. Unless stated otherwise, we treat `script` request events as unique by the tuple (HTTP method, URL, frame identifier). Metrics S1–S3 capture the request volume (S1), domain diversity (S2), and resource footprint (S3) of third-party script activity in the pre- and post-interaction windows. Metrics T1 and T2 measure how quickly third-party scripts appear relative to the 0 s visit timeline in Figure 2. T1 is the elapsed time from 0 s until the first third-party `script` request is observed; T2 is defined analogously but restricted to the tracker-category subset (Section 3.4). T3 quantifies the change in third-party script request volume between the two windows on the same site (i.e., $\Delta S1 = S1_{\text{post}} - S1_{\text{pre}}$). For CBT sites, this difference reflects the immediate effect of an explicit Accept or Reject choice; for implicit tracking sites, it characterises how default activity evolves between the early and later phases of a visit.

3.5.2 Dependency and Execution Behaviours (RQ2). To address RQ2, we characterise how third-party scripts are organised and how they behave at runtime. Metrics D1–D3 summarise the structure of each site's third-party dependency graph, capturing the maximum depth of nested inclusions, the typical fan-out of third-party nodes, and the extent to which third-party dependency edges cross eTLD+1 boundaries. Together, these metrics describe how third-party inclusions propagate through nested and cross-site loading relationships.

Metrics E1–E3 translate the execution traces into aggregate behavioural signals. E1 counts dynamic code execution events observed during the visit, while E2 counts deferred execution events

Table 2: Summary of metrics. S/T metrics address RQ1 (scale and timing of third-party script loading); D/E metrics address RQ2 (dependency structure and execution behaviour).

ID	Type	Description
S1	Scale	Number of third-party script request events
S2	Scale	Number of distinct third-party domains (eTLD+1)
S3	Scale	Total size (bytes) of third-party script resources
T1	Timing	Time to first third-party script request
T2	Timing	Time to first tracker-category script request
T3	Timing	Change in third-party script request count from pre to post window
D1	Dependency	Maximum depth from page to any third-party node
D2	Dependency	Average out-degree of third-party nodes in the dependency graph
D3	Dependency	Fraction of third-party dependency edges crossing eTLD+1 boundaries
E1	Execution	Count of dynamic code execution events (window-level)
E2	Execution	Count of deferred execution events (delay > 1 s or idle callbacks)
E3	Execution	Window-level error rate normalised by S1

scheduled beyond 1 s or into idle time. For E3, we quantify a window-level error rate as the number of page-level runtime failure signals observed within window w (JavaScript runtime errors and unhandled promise rejections) normalised by the third-party script request count in the same window. Formally, $E3_w = |errors_w|/S1_w$, where $errors_w$ denotes the set of captured failure signals whose timestamps fall inside window w, and $S1_w$ is the corresponding third-party script request count in that window. We interpret E3 as an indicator of how error-prone the page's runtime environment is in that window, without attributing errors to specific scripts.

3.5.3 Interface–Behaviour Inconsistencies After Rejection. We use the RQ1 scale metrics to quantify how often observed third-party script activity conflicts with the intended effect of a reject choice on CBT sites. This analysis is restricted to CBT-Reject visits where the interface exposes an explicit reject or necessary-only option *and* the scripted reject action succeeds. A site is flagged as exhibiting an *interface–behaviour inconsistency* if, after rejection, at least one of the following holds: (i) $\Delta S1 > 0$, indicating an increase in third-party script request count; (ii) $\Delta S2 > 0$, indicating an increase in the number of distinct third-party domains (eTLD+1); or (iii) a tracker-category domain from our predefined pattern list appears in the post-interaction window but not in the pre-interaction window. These indicators are computed solely from behavioural traces and do not feed back into the interface-based labels; instead, they are treated as outcome variables reported in Section 4.

3.6 Statistical Analysis

Our unit of analysis is the site. For each metric and scenario–window combination, we first aggregate repeated crawls of the same domain into a single site-level value using the median, and exclude visits that failed or did not complete as described in Section 3.4. Medians reduce the influence of extreme values. The resulting distributions of S1–S3, T1–T3, D1–D3 and E1–E3 are skewed and heavy-tailed, so we use non-parametric statistical tests.

For comparisons between tracking paradigms (e.g., CBT-Pre vs. implicit pre windows, or CBT-Reject vs. Implicit-Baseline), we use two-sided Mann–Whitney U tests [18] on the corresponding site-level samples. For within-site comparisons on the same CBT domains (e.g., CBT-Pre vs. CBT-Reject), we use two-sided Wilcoxon signed-rank tests [33] on the per-site differences. Unless stated otherwise, we use a significance threshold of $\alpha = 0.05$ and report p-values together with standard effect sizes: rank-biserial correlation r for the non-parametric tests and Cohen's d for paired differences, where larger absolute values indicate stronger effects [6]. We do not apply formal multiple-comparison corrections; instead, we focus on the direction and magnitude of effects and on whether related metrics show consistent patterns. Plots in Section 4 present site-level distributions using standard boxplots, with means and standard deviations reported only as additional context.

To address the common concern that non-significant results do not imply equivalence, we additionally apply the two one-sided tests (TOST) procedure for mean equivalence when comparing CBT-Reject against implicit tracking. We use equivalence bounds of $\Delta = \pm 1$ third-party script, which represents the smallest non-zero difference for this discrete metric and a practically meaningful tolerance in a privacy setting. We report 90% confidence intervals (CIs) under a pooled-variance t formulation; equivalence would require the 90% CIs to fall entirely within $[-\Delta, +\Delta]$.

We interpret rank-biserial correlations following established guidelines: $|r| < 0.1$ as negligible, $0.1 \le |r| < 0.3$ as small, $0.3 \le |r| < 0.5$ as medium, and $|r| \ge 0.5$ as large. For key comparisons, we additionally report the proportion of variance explained (r^2) to provide a concrete sense of practical significance. We note that small effect sizes in large-scale web measurement studies can still reflect meaningful and reliable phenomena, given the substantial heterogeneity of real-world websites.

4 Empirical Results

We report results at the site level. Unless stated otherwise, all comparisons use the non-parametric tests described in Section 3.6, and we discuss both statistical significance and effect sizes.

4.1 Dataset Coverage and Data Quality

Table 3 summarises the measurement dataset. After applying the data-quality filters in Section 3.4, we obtained 1,124 valid CBT-Pre traces, 1,131 CBT-Reject traces, and 849 implicit tracking traces. These counts are window-specific: a CBT site may yield a valid trace in one window but not the other. Consequently, paired analyses use the intersection of sites with valid traces in both windows (e.g., n = 1,116 for the CBT-Pre vs. CBT-Reject pairing in Figure 4(b)).

Table 3: Dataset coverage and observation windows.

Scenario	Sites	Valid traces	Time window
CBT-Pre	1 124	1 124	0–5 s (initial load)
CBT-Reject	1 131	1 131	5–35 s (post-reject)
Implicit-Pre	849	849	0–5 s (baseline)
Implicit-Baseline	849	849	5–35 s (post-interaction)
Total	3 104	3 104	—

We excluded visits in which the browser fails to complete the scripted flow or does not produce client-side activity in the observation windows, for example due to navigation timeouts, missing click targets on consent interfaces, or redirects that reduce the dwell time. These failures constitute only a minority of visits and are excluded from all subsequent analyses. The remaining traces show non-zero network and script activity across both tracking paradigms and interaction scenarios.

4.2 RQ1: Script Loading Scale and Timing

4.2.1 Initial Page Load Before Any Consent Decision. We first examined third-party script loading during the initial page load, before any consent decision can reasonably be made. Both CBT-Pre and implicit tracking sites are visited with a clean browser profile and without additional interaction. Differences in this window therefore reflect default loading strategies.

Figure 3(a) shows that CBT-Pre sites load more third-party script requests in the first 5 s than implicit tracking sites (median 7 vs. 4 script request events; $p < 0.0001$, $r \approx 0.164$). The effect size is small ($r \approx 0.164$, $r^2 = 0.027$), indicating that the tracking paradigm accounts for only a modest fraction of the variance. In practice, this suggests that other factors (e.g., site category, geographic targeting, and monetisation strategy) play a substantial role in determining third-party script activity. Figure 3(b) shows that CBT-Pre sites also reach their first third-party script earlier (median 229 ms vs. 523 ms; $p < 0.0001$, $r \approx 0.23$). In other words, sites that later display a consent banner already expose users to a larger and earlier third-party footprint before any choice is available.

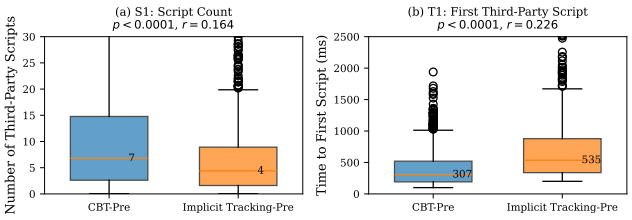


Figure 3: Initial third-party script activity before any consent decision. (a) S1: number of third-party script requests in the first 5 s (higher = more third-party fetching activity). (b) T1: time to the first third-party script request (lower = earlier exposure).

Table 4: T2: Time to First Tracker-Category Script Request

Metric	CBT-Pre	IT-Pre
Sites with tracker-category (n)	434	436
Mean (ms)	1349	1465
Median (ms)	1099	1287
SD (ms)	1233	1225
Activated within 5 s	100%	100%
Mann-Whitney U p	0.054	
Effect size (r)	0.075	

Table 4 presents the timing of tracker-category script activation (T2). Among sites that loaded at least one tracker-category script in the pre-interaction window (434 CBT-Pre sites and 436 implicit tracking sites), CBT-Pre sites activated such scripts slightly earlier than implicit tracking sites (median 1,099 ms vs. 1,287 ms), though this difference is not statistically significant ($p = 0.054$, $r = 0.075$). Notably, all observed sites activated their first tracker-category script within 5 s of page load, which is earlier than most users can realistically read and respond to a consent banner. This finding suggests that, for many CBT sites, the consent interface overlays an already active tracking stack rather than acting as a gate that delays third-party code until user approval.

4.2.2 "No-Consent" State: CBT-Reject vs. Implicit Tracking. We next compared the post-interaction window between CBT-Reject and implicit tracking sites, which together represent a logical "no-consent" state. As shown in Figure 4(a), S1 values in this window are similar: CBT-Reject traces have a median of 3 third-party script requests, while implicit tracking traces have a median of 2. This difference is not statistically significant ($p = 0.243$, $r = 0.031$). However, non-significance does not imply equivalence. A TOST equivalence test with bounds of ±1 script request also fails to establish equivalence ($p_{\text{TOST}} = 0.349$; 90% CI $[-1.40, 2.48]$), indicating that the mean difference cannot be concluded to lie within a one-request tolerance.

Figure 4(b) reports within-visit changes on CBT sites, $\Delta S1 = S1_{\text{post}} - S1_{\text{pre}}$. Here, $\Delta S1 > 0$ indicates more third-party script requests in the later window, whereas $\Delta S1 < 0$ indicates fewer requests. This comparison captures temporal progression within a visit and does not, by itself, establish that the reject click caused the observed change.

4.2.3 Post-reject Inconsistencies on CBT Sites. To understand how individual CBT sites react to a reject choice, we compared the pre- and post-interaction windows for the same domains. Measured by S1, the distribution of per-site changes is highly skewed: many sites reduce or maintain their third-party script counts after rejection, while a substantial minority increase them. A paired non-parametric test on the per-site differences detects a small but statistically significant shift ($p = 0.0001$, Cohen's $d \approx -0.07$), indicating a slight net decrease in script counts across the paired samples despite substantial site-level heterogeneity.

Figure 4(b) summarises the direction of these within-visit changes. Out of 1,116 CBT sites with valid paired measurements in both the

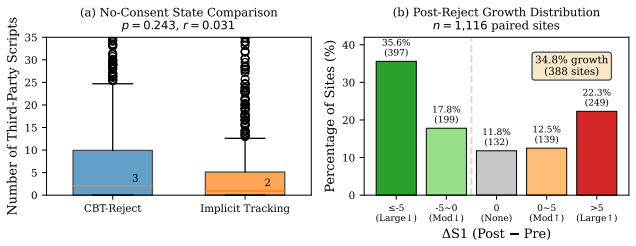


Figure 4: Third-party script activity in the logical "no-consent" state. (a) Post-interaction S1 for CBT-Reject vs. implicit tracking. (b) Within-visit change on CBT sites, $\Delta S1 = S1_{\text{post}} - S1_{\text{pre}}$.

pre-interaction (0–5 s after page load) and the 30 s post-reject window, 388 (34.8%) exhibit $\Delta S1 > 0$ (more third-party scripts observed in the later window), while 728 (65.2%) show no increase or a decrease. Notably, 249 sites (22.3%) exhibit large increases ($\Delta S1 > 5$), with extreme cases adding over 100 additional script requests in the later window.

Table 5 refines these within-visit changes using domain-level signals. A subset of CBT sites not only increases script counts but also introduces additional third-party domains and advertising or analytics providers that were absent in the early window. Overall, a reject option does not reliably constrain third-party activity to the level observed in the initial pre-interaction phase. Substantial third-party expansion can still occur later in the same visit under the reject scenario. Importantly, $\Delta S1$ captures within-visit time progression and does not, on its own, establish that the reject action causally triggers the additional loading.

Table 5: Within-visit $\Delta S1$ **under the reject scenario on CBT sites (0–5 s vs. 30 s post-reject window).**

Panel A: Script Count Changes (ΔS1)

Change Pattern	Sites	%
Large decrease (ΔS1 ≤ -5)	397	35.6%
Moderate decrease (-5 < ΔS1 < 0)	199	17.8%
No change (ΔS1 = 0)	132	11.8%
Moderate increase (0 < ΔS1 ≤ 5)	139	12.5%
Large increase (ΔS1 > 5)	249	22.3%

Panel B: Summary Indicators

Category	Sites	%
Script-count increase (ΔS1 > 0)	388	34.8%
No growth (ΔS1 ≤ 0)	728	65.2%

Table 6 highlights five sites with the largest post-reject growth in third-party script requests ($\Delta S1 > 150$). Manual inspection of network traces reveals several recurring patterns: (i) rapid advertising-request cascades in the post-interaction window, where a header-bidding component (e.g., Prebid.js) triggers many successive third-party requests to multiple bidding endpoints and fallbacks within a short time span; (ii) non-essential third-party bundles (e.g., analytics suites) that still load in full despite a "necessary only" selection; (iii) Google Tag Manager configurations that inject scripts after a fixed delay, effectively decoupling script loading from the consent choice; and (iv) secondary CMPs that introduce additional script bundles after the primary consent flow completes. These cases show that the post-reject behaviour can diverge from what the interface suggests. While additional third-party script requests appear in the post-reject window, our measurements do not establish whether these increases are caused by the reject action itself or reflect delayed initialisation, page-lifecycle events, or secondary CMP logic.

Table 6: Extreme Post-Reject Script Growth: Selected Cases

Site	Pre	Post	ΔS1	Observed Pattern
Site A	65	378	+313	Advertising-request cascade
Site B	14	271	+257	Full analytics bundle loaded
Site C	20	271	+251	Delayed Tag Manager script injection
Site D	16	263	+247	Non-essential scripts still loaded
Site E	0	177	+177	Secondary CMP script bundle

4.3 RQ2: Dependency Structure and Execution Behaviour

4.3.1 Structural Complexity of Third-Party Dependencies. Figure 5 compares third-party dependency structure in the post-interaction window. We report the percentage of sites in each category, so shifts toward the right-hand bins reflect more structurally complex dependency structures. $D1$ measures the maximum inclusion depth from the page to any third-party script node (higher values indicate longer inclusion chains). Most sites in both groups have flat or very shallow chains, but CBT-Reject sites show a slightly higher tendency toward deeper chains than implicit tracking sites. This difference is statistically significant but small ($p \approx 0.024$, $r \approx 0.06$).

For $D2$ (average out-degree) and $D3$ (fraction of third-party dependency edges crossing eTLD+1 boundaries), the distributions for CBT-Reject and implicit tracking sites are nearly identical. Over 80% of sites have $D2 = 0$, indicating that third-party script nodes have no outgoing dependency edges in the observed graph, and over 90% have $D3 = 0$, indicating that no observed third-party-to-third-party dependency edge crosses an eTLD+1 boundary. The corresponding tests do not reject the null of equal distributions ($p > 0.17$). Overall, dependency structure in the post-reject state is broadly similar across the two paradigms, with only a modest shift toward deeper inclusion chains on CBT-Reject sites.

4.3.2 Dynamic Execution Behaviours. Figure 6 compares execution-level behaviours in the post-interaction window for CBT-Reject and implicit tracking sites.

For $E1$ (dynamic code execution), CBT-Reject traces exhibit a heavier tail than implicit tracking traces, and the distributional difference is statistically significant ($p \approx 0.001$, $r \approx 0.09$), even though the median remains zero in both groups, indicating that the difference is driven by a subset of high-activity sites. Although the effect size is small, it is practically meaningful because dynamic code execution is a sparse, high-impact behaviour: most sites

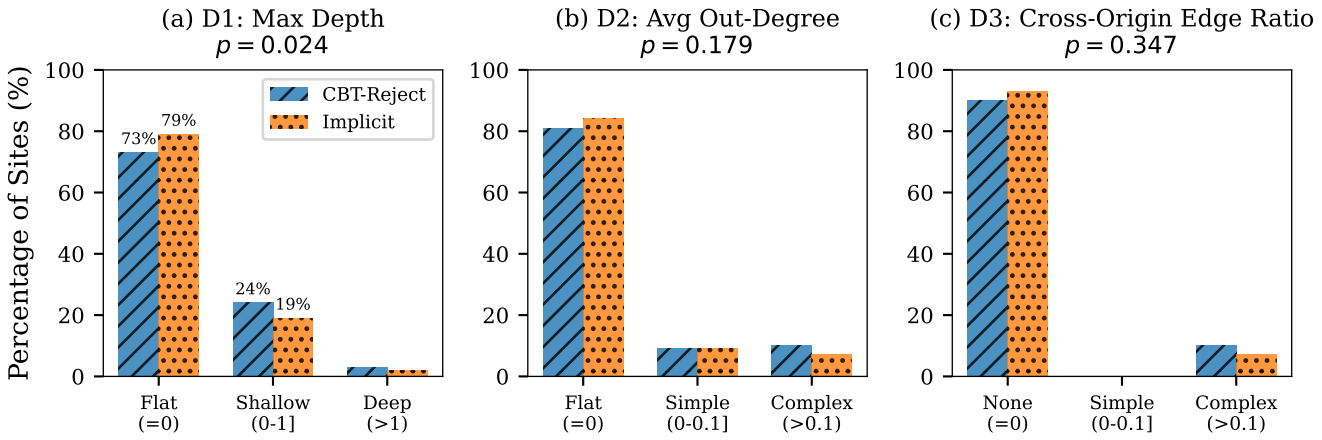


Figure 5: Dependency-structure distributions in the post-interaction window for CBT-Reject and implicit tracking sites: (a) D1 max depth, (b) D2 avg out-degree, (c) D3 cross-eTLD+1 edge ratio.

never trigger it, but the subset that does can substantially increase the complexity and unpredictability of client-side execution. The observed shift therefore indicates that CBT-Reject is more likely to coincide with these high-activity cases, even if the typical site remains unchanged.

For *E*2 (deferred execution), the two groups show similar distributions ($p \approx 0.31$), suggesting that not all runtime signals vary by tracking paradigm under our scripted flow. In contrast, *E*1 and *E*3 exhibit detectable differences, indicating that the observed divergence is selective rather than a uniform shift across all execution metrics.

For *E*3 (runtime error rate), CBT-Reject sites exhibit higher error rates than implicit tracking sites ($p = 0.031$, $r \approx 0.057$; Table 7). The mean differs substantially (22.0% vs. 5.6%), but the median is zero in both groups, indicating that the average is driven by a subset of high-error outliers. Consistent with this, a larger proportion of CBT-Reject sites fall into the high-error category (E3 > 5%: 18.6% vs. 13.1%). Although the effect size is small, it is still meaningful because runtime errors reflect fragile integrations and failure-prone execution paths under certain conditions. A plausible explanation is that the elevated dynamic code execution observed under CBT-Reject (E1) co-occurs with a noisier runtime environment: dynamically generated or injected code can be more failure-prone when consent-dependent dependencies are missing, loaded conditionally, or initialised late. We emphasise that our measurements capture site-level co-occurrence and do not attribute individual errors to specific scripts.

In summary, following a reject choice, the dependency structure of third-party scripts on CBT sites is broadly similar to that of implicit tracking sites, with only a minor shift toward deeper inclusion chains. The more visible differences appear at the execution level: CBT-Reject sites exhibit heavier-tailed dynamic execution activity and elevated error rates, suggesting a noisier runtime environment rather than a fundamentally different dependency architecture.

Table 7: E3: Runtime Error Rate Analysis

Metric	CBT-Reject	Implicit
Mean error rate	21.987%	5.625%
Median error rate	0.000%	0.000%
High error sites (>5%)	210 (18.6%)	111 (13.1%)
Relative difference	+290.9%	
Statistical Test		
Mann-Whitney U	p = 0.0307	
Effect size (r)	0.057	

5 Discussion and Implications

Our measurements show that CBT and implicit tracking are technically closer than their interfaces suggest. CBT sites load more third-party scripts and activate them earlier during the initial page load, even before users can reasonably respond to a consent interface. In the post-reject window, overall third-party script counts on CBT and implicit tracking sites appear similar. However, a substantial subset of CBT sites expands its third-party footprint after a reject choice and exhibits more dynamic code execution and higher runtime error rates. In this section, we discuss the implications of these findings for users, web practitioners, and policy makers, and outline limitations and directions for future work.

5.1 Implications for Users

From an end-user perspective, the observed behaviour is less clear-cut than consent banner text suggests. In principle, rejecting optional cookies or non-essential tracking should be the privacy-protective choice, while accepting should permit broader tracking. In practice, however, an explicit reject decision does not reliably reduce observable third-party activity within the same visit. On a sizable subset of CBT sites, the post-reject window (30 s) contains more third-party scripts and tracking-related domains than the

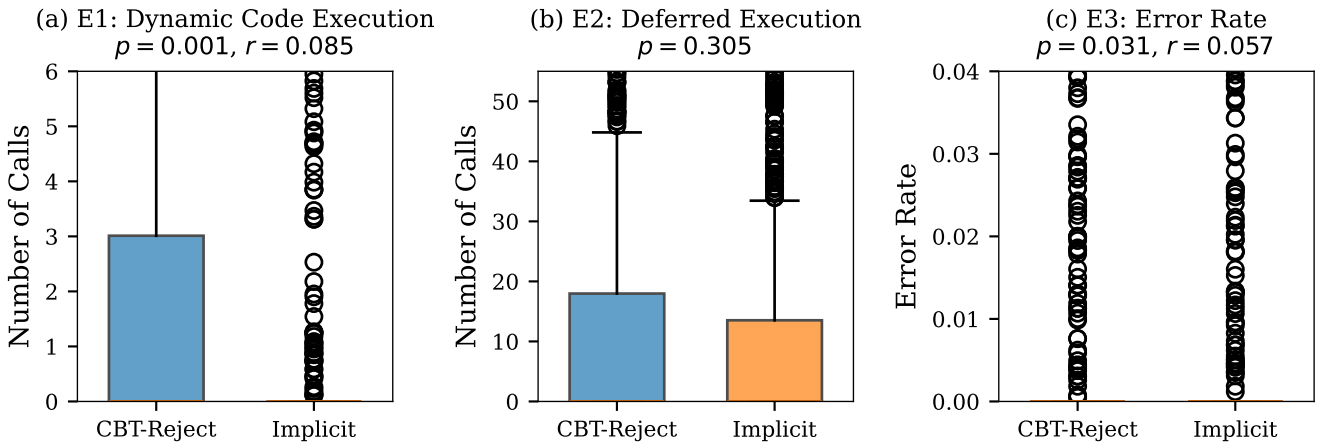


Figure 6: Execution-metric distributions in the post-reject state for CBT-Reject and implicit tracking sites: (a) E1 dynamic execution, (b) E2 deferred execution, (c) E3 error rate.

pre-interaction window (0–5 s), although our measurements cannot determine whether this is triggered by the reject action itself or by other within-visit dynamics (e.g., delayed initialisation).

These findings suggest that consent banners alone are an unreliable control surface for limiting third-party tracking. While interface choices remain important for expressing user intent and supporting legal compliance, the mapping from a "reject" choice to script-level behaviour is weak and inconsistent. Users seeking stronger privacy protections may therefore combine consent decisions with technical safeguards at the browser or network layer, such as tracking-protection modes [26, 32], blocker extensions [20], or restrictive settings for third-party cookies and storage. More broadly, our results highlight a transparency gap: users are asked to make privacy decisions without a reliable way to anticipate or verify how third-party code behaves after those decisions are made.

5.2 Implications for Web Developers and Operators

For web developers, site operators, and consent management platform (CMP) providers, the results underscore the need to align implementation practice with the intended semantics of consent flows. Many CBT sites load a substantial set of third-party scripts before any consent decision, and a significant subset introduces additional third-party scripts or domains even after a reject choice. This pattern undermines the intended purpose of explicit consent and increases both privacy risk and system complexity.

Developers should minimise pre-consent third-party loading by deferring non-essential scripts until after explicit opt-in and by separating strictly necessary functionality from advertising, analytics, and profiling components. Once a reject choice is offered, non-essential third-party scripts should be genuinely gated on that decision, rather than being loaded unconditionally or reintroduced through indirect dependency chains. Our dependency and execution metrics suggest concrete engineering targets: reduce dependency-chain depth, avoid unnecessary cross-origin loading cascades, and limit opaque dynamic code execution. Meeting these targets can lower supply-chain risk and reduce the likelihood of fragile, error-prone integrations. Periodic internal audits using browser-side traces similar to those collected in this study could help developers verify that their implementations align with stated consent policies.

5.3 Implications for Regulators and Policy Management

For regulators, standards bodies, and organisations responsible for privacy governance, our findings highlight a gap between interface-level compliance checks and client-side technical behaviour. Current enforcement and certification practices often focus on whether a site presents a consent banner, offers particular textual options, and records consent states correctly. Our measurements show that these properties alone are insufficient to ensure that reject choices meaningfully constrain third-party tracking.

Regulatory guidance and audit frameworks could be strengthened by incorporating simple, reproducible script-level indicators, in line with recent recommendations [10]. Examples include comparing pre- and post-decision third-party script counts and domains, detecting the appearance of new advertising or analytics providers after a reject choice, and flagging deployments where post-reject windows consistently exhibit growing third-party footprints. Integrating such metrics into regulatory inspection tools, CMP certification processes, and internal compliance programmes would make it harder for superficially compliant consent flows to mask continued tracking, and would provide organisations clearer technical benchmarks for demonstrating effective enforcement of user choices.

5.4 Limitations and Future Work

Our study has several limitations. First, we focus on desktop browsing with a single browser configuration and network environment. In addition, all measurements were conducted from a single geographic region, which may affect both banner presentation and

tracking behaviour. Different devices, browsers, connection conditions, regions, or user behaviours (e.g., scrolling or navigating within a site) may produce different tracking dynamics. Second, our measurements are limited to client-visible third-party scripts and related browser-side activities. We do not observe server-side processing, cross-device identifiers, or the full semantics of individual scripts, and we cannot directly infer underlying data flows or business logic. More generally, our pre- and post-interaction design captures observable behavioural differences but does not, by itself, establish causal relationships between user interactions and subsequent tracking behaviour. Third, our classification of tracking paradigms is based on observable interfaces within a fixed observation window and is not intended as a legal assessment of compliance with any specific regulatory framework. Some sites may implement consent logic in ways that are not fully captured by our metrics. Finally, our analysis is based on a snapshot of popular domains from a specific point in time and does not capture how consent implementations evolve in response to regulatory changes, platform policies, or market pressures. In addition, website category may influence tracking deployments and thus represents a potential confounding factor that we do not explicitly control for in our sampling design.

Future work could extend our methodology to mobile platforms and region-specific samples, incorporate richer classifications of third-party functionality and data flows, and perform longitudinal measurements to track how consent practices and third-party ecosystems change over time. Another promising direction is to combine our script-level metrics with user-centric studies that assess how different consent designs affect user understanding, expectations, and behaviour, thereby connecting consent decisions directly with technical enforcement.

6 Conclusion

This paper presented a comparative, browser-based measurement of third-party script behaviour under two observable tracking paradigms: CBT and implicit tracking. We showed that CBT sites tend to load more third-party scripts and activate them earlier during the initial page load than implicit tracking sites, and that a substantial subset of CBT sites expands its third-party footprint even after users explicitly reject non-essential tracking. In contrast, the overall dependency structure and execution patterns observed after rejection remain broadly similar to those of implicit tracking sites, although CBT deployments exhibit slightly more dynamic code execution and higher runtime error rates.

Overall, these findings suggest that explicit consent interfaces and implicit tracking practices are technically more intertwined. By combining an interface-level classification of tracking paradigms with script-level technical metrics, our study provides a reproducible measurement approach and identifies actionable intervention points for users, practitioners, and policy makers who seek to narrow the gap between consent in theory and tracking in practice.

Acknowledgments

We thank the anonymous reviewers for their valuable comments and helpful suggestions.

References

[1] Gunes Acar, Marc Juarez, Nick Nikiforakis, Claudia Diaz, Seda Gürses, Frank Piessens, and Bart Preneel. 2013. FPDetective: dusting the web for fingerprinters. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security* (Berlin, Germany) *(CCS '13)*. Association for Computing Machinery, New York, NY, USA, 1129–1140. doi:10.1145/2508859.2516674

[2] Syed Suleman Ahmad, Muhammad Daniyal Dar, Muhammad Fareed Zaffar, Narseo Vallina-Rodriguez, and Rishab Nithyanand. 2020. Apophanies or Epiphanies? How Crawlers Impact Our Understanding of the Web. In *Proceedings of The Web Conference 2020* (Taipei, Taiwan) *(WWW '20)*. Association for Computing Machinery, New York, NY, USA, 271–280. doi:10.1145/3366423.3380113

[3] Dino Bollinger, Karel Kubicek, Carlos Cotrini, and David Basin. 2022. Automating Cookie Consent and GDPR Violation Detection. In *31st USENIX Security Symposium (USENIX Security 22)*. 2893–2910. https://www.usenix.org/conference/usenixsecurity22/presentation/bollinger

[4] Ahmed Bouhoula, Karel Kubicek, Amit Zac, Carlos Cotrini, and David Basin. 2024. Automated Large-Scale Analysis of Cookie Notice Compliance. In *33rd USENIX Security Symposium (USENIX Security 24)*. 1723–1739. https://www.usenix.org/conference/usenixsecurity24/presentation/bouhoula

[5] Tomasz Bujlow, Valentín Carela-Español, Josep Solé-Pareta, and Pere Barlet-Ros. 2017. A Survey on Web Tracking: Mechanisms, Implications, and Defenses. *Proc. IEEE* 105, 8 (2017), 1476–1510. doi:10.1109/JPROC.2016.2637878

[6] Jacob Cohen. 1988. *Statistical Power Analysis for the Behavioral Sciences* (2nd ed.). Lawrence Erlbaum Associates. doi:10.4324/9780203771587

[7] Martin Degeling, Christine Utz, Christopher Lentzsch, Henry Hosseini, Florian Schaub, and Thorsten Holz. 2019. We value your privacy... now take some cookies: Measuring the GDPR's impact on web privacy. In *Proceedings 2019 Network and Distributed System Security Symposium*. 15 pages. doi:10.14722/ndss.2019.23378

[8] Steven Englehardt and Arvind Narayanan. 2016. Online Tracking: A 1-million-site Measurement and Analysis. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (Vienna, Austria) *(CCS '16)*. Association for Computing Machinery, New York, NY, USA, 1388–1401. doi:10.1145/2976749.2978313

[9] European Commission. 2016. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:32016R0679 [Online]. Accessed: 2026-04-08.

[10] European Data Protection Board. 2020. Guidelines 05/2020 on consent under Regulation 2016/679. https://edpb.europa.eu/our-work-tools/our-documents/guidelines/guidelines-052020-consent-under-regulation-2016679_en [Online]. Accessed: 2026-04-08.

[11] Colin M. Gray, Cristiana Santos, Nataliia Bielova, Michael Toth, and Damian Clifford. 2021. Dark Patterns and the Legal Requirements of Consent Banners: An Interaction Criticism Perspective. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 172, 18 pages. doi:10.1145/3411764.3445779

[12] David Klein, Marius Musch, Thomas Barber, Moritz Kopmann, and Martin Johns. 2022. Accept All Exploits: Exploring the Security Impact of Cookie Banners. In *Proceedings of the 38th Annual Computer Security Applications Conference* (Austin, TX, USA) *(ACSAC '22)*. Association for Computing Machinery, New York, NY, USA, 911–922. doi:10.1145/3564625.3564647

[13] Simon Koch, Benjamin Altpeter, and Martin Johns. 2023. The OK Is Not Enough: A Large Scale Study of Consent Dialogs in Smartphone Applications. In *32nd USENIX Security Symposium (USENIX Security 23)*. 5467–5484. https://www.usenix.org/conference/usenixsecurity23/presentation/koch

[14] Tobias Lauinger, Abdelberi Chaabane, Sajjad Arshad, William Robertson, Christo Wilson, and Engin Kirda. 2017. Thou Shalt Not Depend on Me: Analysing the Use of Outdated JavaScript Libraries on the Web. In *Proceedings of the 2017 Network and Distributed System Security Symposium (NDSS)*. Internet Society, 15 pages. doi:10.14722/ndss.2017.23414

[15] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. 2019. Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In *Proceedings 2019 Network and Distributed System Security Symposium*. Internet Society, 15 pages. doi:10.14722/ndss.2019.23386

[16] Ada Lerner, Anna Kornfeld Simpson, Tadayoshi Kohno, and Franziska Roesner. 2016. Internet Jones and the Raiders of the Lost Trackers: An Archaeological Study of Web Tracking from 1996 to 2016. In *25th USENIX Security Symposium (USENIX Security 16)*. USENIX Association. https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/lerner

[17] Kyungchan Lim, Yonghwi Kwon, and Doowon Kim. 2023. A Longitudinal Study of Vulnerable Client-side Resources and Web Developers' Updating Behaviors. In *Proceedings of the 2023 ACM on Internet Measurement Conference* (Montreal QC, Canada) *(IMC '23)*. Association for Computing Machinery, New York, NY, USA, 162–180. doi:10.1145/3618257.3624804

[18] Henry B. Mann and Donald R. Whitney. 1947. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics* 18, 1 (1947), 50–60. doi:10.1214/aoms/1177730491

[19] Célestin Matte, Nataliia Bielova, and Cristiana Santos. 2020. Do Cookie Banners Respect my Choice? Measuring Legal Compliance of Banners from IAB Europe's Transparency and Consent Framework. In *2020 IEEE Symposium on Security and Privacy (SP)*. 791–809. doi:10.1109/SP40000.2020.00076

[20] Georg Merzdovnik, Markus Huber, Damian Buhov, Nick Nikiforakis, Sebastian Neuner, Martin Schmiedecker, and Edgar Weippl. 2017. Block Me If You Can: A Large-Scale Study of Tracker-Blocking Tools. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*. 319–333. doi:10.1109/EuroSP.2017.26

[21] Shaoor Munir, Sandra Siby, Umar Iqbal, Steven Englehardt, Zubair Shafiq, and Carmela Troncoso. 2023. CookieGraph: Understanding and Detecting First-Party Tracking Cookies. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security* (Copenhagen, Denmark) *(CCS '23)*. Association for Computing Machinery, New York, NY, USA, 3490–3504. doi:10.1145/3576915.3616586

[22] Mischa Nouwens, Ilaria Liccardi, Michael Veale, David Karger, and Lalana Kagal. 2020. Dark Patterns after the GDPR: Scraping Consent Pop-ups and Demonstrating Their Influence. In *Proceedings of the 2020 CHI conference on human factors in computing systems*. 1–13. doi:10.1145/3313831.3376321

[23] Puppeteer Documentation. 2025. Controlling Chrome DevTools Protocol. https://pptr.dev/guides/evaluate-javascript [Online]. Accessed: 2026-04-08.

[24] Franziska Roesner, Tadayoshi Kohno, and David Wetherall. 2012. Detecting and Defending Against Third-Party Tracking on the Web. In *9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*. USENIX Association, 155–168. https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/roesner

[25] Quirin Scheitle, Oliver Hohlfeld, Julien Gamba, Jonas Jelten, Torsten Zimmermann, Stephen D. Strowes, and Narseo Vallina-Rodriguez. 2018. A Long Way to the Top: Significance, Structure, and Stability of Internet Top Lists. In *Proceedings of the Internet Measurement Conference 2018* (Boston, MA, USA) *(IMC '18)*. Association for Computing Machinery, New York, NY, USA, 478–493. doi:10.1145/3278532.3278574

[26] Peter Snyder, Soroush Karami, Benjamin Livshits, and Narseo Vallina-Rodriguez. 2023. Pool Party: Exploiting Browser Resource Limits for Web Tracking. In *32nd USENIX Security Symposium*. 7091–7105. https://www.usenix.org/conference/usenixsecurity23/presentation/snyder

[27] Steven Sprecher, Christoph Kerschbaumer, and Engin Kirda. 2022. SoK: All or Nothing - A Postmortem of Solutions to the Third-Party Script Inclusion Permission Model and a Path Forward. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS &P)*. 206–222. doi:10.1109/EuroSP53844.2022.00021

[28] Brian Tang, Duc Bui, and Kang G Shin. 2025. Navigating cookie consent violations across the globe. In *Proceedings of the 34th USENIX Conference on Security Symposium*. 5817–5836. https://www.usenix.org/conference/usenixsecurity25/presentation/tang

[29] The European Parliament and the Council of the European Union. 2002. Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32002L0058 [Online]. Accessed: 2026-04-08.

[30] Michal Tóth, Nataliia Bielova, and Vincent Roca. 2022. On dark patterns and manipulation of website publishers by CMPs. *Proc. Priv. Enhancing Technol.* 2022 (2022), 478–497. https://api.semanticscholar.org/CorpusID:249951415

[31] Christine Utz, Martin Degeling, Sascha Fahl, Florian Schaub, and Thorsten Holz. 2019. (Un)informed Consent: Studying GDPR Consent Notices in the Field. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security* (London, United Kingdom) *(CCS '19)*. Association for Computing Machinery, New York, NY, USA, 973–990. doi:10.1145/3319535.3354212

[32] WebKit Team. 2023. Intelligent Tracking Prevention. https://webkit.org/tracking-prevention/ [Online]. Accessed: 2026-04-08.

[33] Frank Wilcoxon. 1945. Individual Comparisons by Ranking Methods. *Biometrics Bulletin* 1, 6 (1945), 80–83. doi:10.2307/3001968

[34] Zhiju Yang and Chuan Yue. 2020. A comparative measurement study of web tracking on mobile and desktop environments. *Proceedings on Privacy Enhancing Technologies* 2020, 2 (2020), 24–44. doi:10.2478/popets-2020-0016

[35] Sebastian Zimmeck, Oliver Wang, Kuba Alicki, Jocelyn Wang, and Sophie Eng. 2023. Usability and enforceability of global privacy control. *Proceedings on Privacy Enhancing Technologies* 2023, 2 (2023), 265–281. doi:10.56553/popets-2023-0052