Can We Build a Privacy-Preserving Web Browser We All Deserve?

The web is the biggest legacy application ever developed or supported by software engineers, but it's also blurring the line between the consumption of data and the leaking of personal details. Browser makers may be the only line of defense.

By Christoph Kerschbaumer, Luke Crouch, Tom Ritter, and Tanvi Vyas

DOI: 10.1145/3220567

Browser vendors constantly evaluate the capabilities of web features to find the right balance between enhanced experiences for the web while also preserving user privacy, weighing the legitimate desires of well-intentioned web innovators with the expected abuse of that same functionality by the ill intentioned. On one hand, web features allow websites to provide a rich user experience; on the other hand they allow adversaries to abuse their capabilities in an attempt to gather as much information about end users as possible. Hence we ask: Can we build a privacy-preserving web browser we all deserve?

The internet is a global computer network that consists of public, private, academic, business, and government networks. To date, it provides the largest collection of information, resources, and services on the planet. User agents, primarily web browsers, allow users to retrieve and display information, and experience all the different kinds of applications the World Wide Web has to offer. At the same time, current web architecture can trick web browsers into leaking confidential user information.

To mitigate this abuse, all major browser vendors (Chrome, Edge, Firefox, Opera, and Safari) are working on various mitigation strategies. Even though not all of those features are ready to be enabled by default within a browser, vendors allow end users to enable a number of privacy protections. This article highlights the benefits for each privacy-enhancing feature, but also discusses drawbacks and further explains why each feature can cause web pages to not operate on a fully functional basis.

BACKGROUND ON CURRENT WEB ARCHITECTURE

As a first line of defense to preserve user privacy, all major web browsers adhere to the guidelines of the same-origin policy, which limits a website's access to information. This policy allows web pages from the same origin to access each other's data, but prevents data access from web pages of different origins. With the same origin policy, malicious JavaScript on one origin cannot obtain access to sensitive data on another origin.

Image by Myvector / Shutterstock.com

Figure 1. Without first party isolation, the same cookie is sent no matter the first party domain.

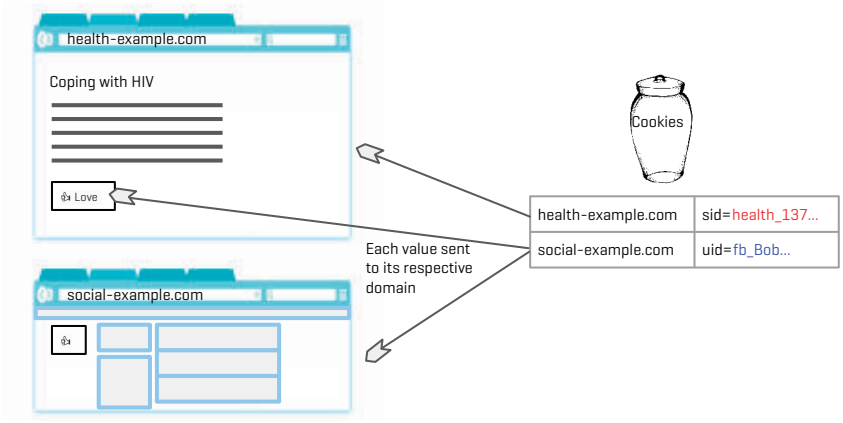
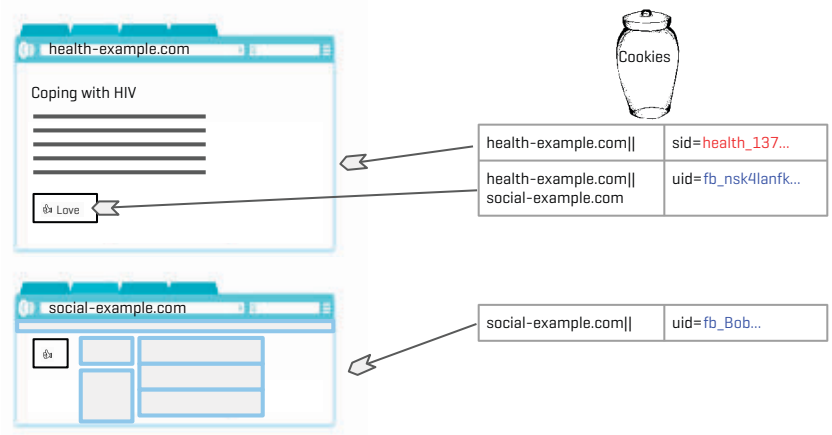


Figure 2. With first party isolation, separate cookies are sent depending on the first party domain.



The same-origin policy also defines which cookies a site can access. A cookie is a small piece of data that a server sends to the user's browser. The browser then stores such cookies locally on the user's device and sends that piece of data back with subsequent requests made to the same origin. Hence, cookies are a mechanism to remember stateful information in the otherwise stateless HTTP protocol. While cookies find legitimate usage on almost every website, tracking sites also rely on cookies to trace users across multiple independent sites on the internet.

TRACKING FUNDAMENTALS

For end users it is obvious to understand that a "first party," which is the website the end user visits and interacts with, is able to log and track movements of the end user interacting with the page. It is less obvious however that "third parties" can also track the movements of end users. Third parties are commonly hidden trackers, such as ad networks embedded on almost every web page, which leverage the architecture of the internet to track users across the web. To illustrate the problem, consider the following example:

Step 1. A user visits the website https://social-example.com and social-example requests the browser to store a cookie.

Step 2. The user's browser will return the cookie back to social-example.com whenever the user visits the site in the future. If the user visits https://health-example.com, the cookies from social-example.com will not be sent because health-example is a different origin.

Step 3. However, if https://social-example.com is embedded within https://health-example.com, social-example's cookies will be sent with the subrequest for social-example.com.

Step 4. Now social-example.com has gained sensitive information and knows the user, as identified by their cookies, has visited health-example.com.

To make things worse, third parties quite often rely on a combination of cookies and other tracking technologies to uniquely identify an end user in an attempt to gather as much information about end users as possible. Trackers then use that data for a variety of purposes, mostly selling data for profitable re-targeting purposes.

PRIVACY PRESERVING FEATURES IN A WEB BROWSER

Browsers have deployed multiple protections to prevent against information leakage. Unfortunately, along with privacy protections, most of these features are also coupled with inadvertently breaking other web experiences. For example, third-party login portals or payment gateways can cause a confusing and disjointed user experience.

In the next sections, we describe various features that most browsers have built-in, but are mostly disabled by default because of said breakage. However, thoughtful users can opt into using one or more of these protections in order to mitigate privacy threats to their data.

Private browsing mode. When browsing the web, a browser remembers lots of information, mostly for convenience. A browser can remember a user's browsing history and cookies, which allows quick access to the pages a user has already visited. However, there are times when this behavior is not necessarily desired, on a shared work computer for example.

Using private browsing within a web browser allows an end user to browse the internet without saving

any information about the sites and pages visited. Therefore, all cookies, and other persistent identifiers, set in private browsing sessions are destroyed and deleted within the browser when the private browsing session ends. This deletion of the browsing state prevents trackers from being able to track end users across multiple sites on the internet.

Stripping third-party cookies. Third-party cookies are cookies that are set by a website other than the one that appears in the address bar. Imagine a webpage that includes a gadget from a social media site. That gadget, which is considered third-party content, can set a cookie that can then be read by the social media site.

Most web browsers have an option to block such third-party cookies. The downside of blocking third-party cookies is that browsers are also dropping cookies from third-parties that are actually cooperating with the main page to make a website work; for example, logging in with your email provider or making payments with your preferred payment site.

The advantage of removing cookies from all third-party requests, however, is it prevents cookie-based tracking from all third parties. In other words, blocking third-party cookies hinders trackers in creating a profile of an end users' browsing habits and therefore also prevents targeted advertisement.

Stripping referrers. When you click a link on a web page to navigate to a second page, the browser sends the exact address of the first page to the second through the so called "referrer value." Most sites log this data for operational, statistical, and even legal purposes. For example, advertisers may require a list of referrer values so they can determine if their ads are displayed alongside certain content. Unfortunately, many sites also log this data to collect as much information about users as possible.

Browsers also send a referrer value when requesting embedded resources, like ads or other social media snippets, integrated in a modern website. In other words, the referrer value allows embedded content to know exactly what pages a user visits. In some cases, the referrer also includes sensitive information (such as the username) that is also leaked to the embedded resource. To mitigate the risk of websites tracking users throughout the internet with the referrer, one can modify the referrer settings within one's browser.

Until browser vendors can ship more of the presented privacy enhancing features by default, we encourage end users to take precocious actions.

Tracking protection. The tracking protection feature relies on a pre-generated list of known tracking sites that browsers can use to identify and block tracking requests. Since requests to tracking sites on the list are entirely blocked, tracking sites do not have access to any data, including a user's cookies and referrer values. Please note that this type of protection is only as good as the blocklist; different browser vendors rely on different blocklists.

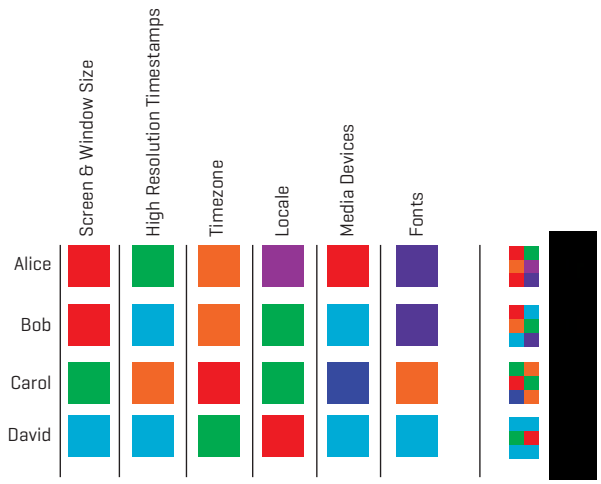
Generally speaking, a basic protection list blocks commonly known analytics trackers, social-media sharing trackers, and advertising trackers. It is worth noting that all blocklist generators constantly maintain and update their list of known trackers to provide the best tracking protection possible for end users.

First-party isolation. First-party isolation segments all browsing data by the top-level domain (the URL displayed in the address bar) the user visits. Instead of using the third-party resource's origin as a single key to set and retrieve cookies within the browser, the first-party isolation approach relies on the origin of the third-party resource in combination with the first party domain. Together those two domains form a double-key, which the browser then uses to look up stored cookies.

Let's consider the following example: If a user visits and logs into https://social-example.com, without first-party isolation, they will receive a cookie that will identify them to social-example. If they then visit https://health-example.com, which contains an embedded resource from https://social-example.com, then social-example will receive the cookie identifying the user (see Figure 1).

However, if they have first party isolation enabled, https://social-example.com will not receive the cookie associated with the prior visit

Figure 3. Websites create a fingerprint of an end user based on the various settings exposed to the web.



to social-example, it will instead receive no cookie or a random cookie not associated with any prior use of social-example (see Figure 2). The first-party isolation model still allows third-party content to provide additional functionality to sites, but prevents third parties from being able to track users across sites using traditional tracking mechanisms.

Containers. Containers are a mechanism to separate the end user's online identity, providing an alternative approach to mitigate tracking without causing the types of breakage we have described with other privacy-preserving features. With containers, website data is segregated into different sections, like "personal", "work", "shopping", and "banking." Data from the browsing activity in one container is inaccessible from websites in a different container.

Containers do not aim to completely prevent tracking, instead, containers limit the amount of tracking to a specific container. In more detail, within a shopping container an end user might still see targeted ads, but the targeting only occurs in that one container and those ads will not follow the user to the work container. Please note containers are not a full-proof tracking prevention tool, since sites can still see a user's IP address, fingerprint the browser, and track a user within a single container.

However, containers can reduce the naive, traditional tracking that happens on the web. Containers further allow isolation of social sites that a user logs in to, and also prevents search providers from tracking an end user across the web, since one can isolate these sites into their own container.

Fingerprinting resistance. Exposing more and more capabilities to websites also exposes more and more details about a user's computer to websites. For example, a website can detect if the user has a webcam attached, what fonts a user has installed, or what the user's time zone is. In fact, a website knows exactly how large a user's browser window is, and even knows if a user has a gamepad attached. Naively, one might think lots of users have a webcam attached and also have the same fonts installed. However, when one combines the full set of features, the number of users with a particular set of attached devices, fonts, time zones, etc. can quite accurately identify a user (see Figure 3).

To make things worse, third parties quite often rely on a combination of cookies and other tracking technologies to uniquely identify an end user.

Once end users started using privacy enhancing features, advertisers started to query web-exposed user information to determine a fingerprint in order to maintain their revenue streams. Using such a fingerprint allows trackers to follow a user around the web, even when they do not send a direct identifier like a cookie or referrer. To combat this, web browsers started to incorporate modes that resist such fingerprinting techniques by simply faking browser responses or normalizing the aspects of a user's browser. For example, making your browser reply with "english" for your language setting already makes a tracker's job harder when trying to identify what language you speak and potentially what country you are in.

Please note, it's extremely difficult to pefect a fingerprinting resistance mode, and it's almost impossible to have all users look identical. Instead of having a unique fingerprint, a user will generally share a fingerprint that is common among hundreds, thousands, or tens of thousands of other users. And when hidden in a crowd that large, the advertiser's unique signal becomes useless noise.

CONCLUSION

It is hard to programmatically determine what content is needed for site functionality and what content abuses current web architecture to extract information from end users.

The most privacy conscious users may enable third-party cookie blocking and resist fingerprinting. In contrast, users who just want advertisers to stop following them may use private browsing or containers. Users who fall in between these two classes may select a combination of these techniques, such as tracking protection, first-party isolation, and referrer stripping.

In summary, it is possible to build a privacy-preserving web browser and ultimately we would like to see browsers ship more of those features by default. However, we also acknowledge it is a hard task to find the right balance between usability and privacy, often devolving to trial-and-error.

Until browser vendors can ship more of the presented privacy enhancing features by default, we encourage end users to take precocious actions by enabling experimental privacy-enhancing features based on the privacy level they are seeking, and providing browsers feedback on what detrimental impacts they experience.

Biographies

Christoph Kerschbaumer is the content security tech lead at Mozilla with more than 10 years of experience in secure systems development. His work ranges from designing systems with fail-safe defaults to fighting cross-site scripting as well as preventing man-in-the-middle attacks. Kerschbaumer received his Ph.D. in computer science from the University of California, Irvine where he based his research on information flow tracking techniques within web browsers.

Luke Crouch is a privacy and security engineer at Mozilla. He builds privacy and security protections and measures their impact. He also has 10 years of experience building full-stack web applications and services.

Tom Ritter is a distinguished security engineer and recovering consultant now at Mozilla, working on anti-exploitation, Tor, and other new and evolving security features. Previously, he did all manner of security consulting and practice management at NCC Group and iSEC Partners. While consulting, Ritter participated in numerous public audit reports including TrueCrypt and Tor Browser; and presented his research on NPR, CNN, and other media outlets.

Tanvi Vyas is a tech lead at Mozilla, specializing in security and privacy user experience. In this role, she leads the development of usable security and privacy features in Firefox, working on implementation, product requirements, user research, and design. Before joining Mozilla, Vyas was a paranoid at Yahoo!, improving the security of Yahoo websites and infrastructure.

1528-4972/18/06 $15.00