Functional Privacy
or Why Cookies are Better with Milk

Robert J. Walls, Shane S. Clark, Brian Neil Levine
Dept. of Computer Science, Univ. of Massachusetts, Amherst
{*rjwalls,ssclark,brian*}*@cs.umass.edu*

Abstract

The price of Internet services is user information, and many pay it without hesitation. While myriad privacy tools exist that thwart the detailed compilation of information about user habits, these tools often assume that reduced functionality is always justified by increased privacy. In contrast, we propose the adoption of *functional privacy* as a guiding principle in the development of new privacy tools. Functional privacy has the overarching goal of maintaining all functionality while improving privacy as much as practically possible — rather than forcing users to make decisions about tradeoffs that they may not fully understand. As a concrete example of a functional privacy approach, we implemented *Milk*, a Google Chrome extension that automatically rewrites HTTP cookies to strictly bind them to the first-party domains from which they were set. We also identify existing privacy-preserving tools that we believe embody the principle of functional privacy and discuss the limitations of others.

1 Introduction

Internet users are largely desensitized to the pervasive advertisements that they encounter on most webpages, accepting them as a fact of life or understanding that the alternative may be to pay for services that are currently free. Though advertisements have been a fixture on the Internet for years, their relationship with users has changed over time. Embedded ads are no longer analogous to physical billboards that are placed in high traffic areas seeking exposure. Instead, they are highly targeted to individual users through detailed tracking carried out by far-reaching analytics and advertising firms. Privacy advocates have created tools that seek to prevent user tracking and targeted ads, but these tools often rob users of functionality without providing a tangible benefit. For example, all popular browsers allow users to disable cookies as a simple privacy control; however many sites require cookies, making web browsing effectively impossible without them.

Given the dilemma of *privacy with reduced functionality* versus *full functionality without privacy*, many choose the latter. Few possess the tenacity that privacy tools require. When something must get done, privacy proponents can find themselves with temporarily disabled privacy tools, rather than with disabled functionality and an unfulfilled task.

We propose the adoption of a *functional privacy* approach, which is the maximal reduction or elimination of user tracking possible without a reduction in service. Forcing users to select between extremes ignores a practical truth: there are users willing to relinquish information to third parties from which they receive services; but they do not wish to reveal more than is required to receive the service, and do not wish to give up any functionality from that service. For example, millions are willing to share their personal details on Facebook, Twitter, and Google; but allowing these companies to track actions across other websites adds nothing from the user's point of view.

A functional privacy tool might not provide the level of privacy attainable by a tool that disables functionality, however, users will never turn off a functional privacy tool because it never interferes with service. Our position is aimed towards the majority of web users, who are not technically savvy. We do not expect them to understand the details of web transactions, but we expect that they do know when a service does not work as expected and can blame a recently installed browser extension.

Consider the following scenario. A user disables third-party cookies in the Chrome browser and then logs into his account at `youtube.com`. On that site, videos can be paused by clicking the video itself. An ad plays after the video and he clicks the ad to pause it. Clicking ads on YouTube does not just pause the ad — it also redirects the user to the website of the advertiser. At this point, the advertiser sets a first-party cookie. Now, anytime the user visits a site with ads from that advertiser, the advertiser will automatically receive the cookie they previously set.

Published in Proceedings of USENIX Workshop on Hot Topics in Securit, 2012

The failing in this case is that third-party cookie blocking in Chrome only blocks storage and not retrieval, but advertisers can employ many methods to evade third-party cookie protections.

One solution to this specific problem is to disable cookies entirely, however many popular services require cookies. Another option is to enact a blacklist of sites that should not be trusted as first parties. This requires tedious updating of the blacklist, and without careful manual updates, the blacklist can unintentionally disable services.

We argue for a functional privacy approach to this problem. We have implemented an open-source extension to Google Chrome called *Milk* that limits such cross-site tracking without reducing functionality for the user. Using Milk, any cookie offered by a site is stored — including third-party cookies — but each cookie is bound to a specific first-party domain. In effect, Milk presents each domain on the Internet with its own cookie store. For example, if the user accepts a cookie *A* from a remote party when it visits site *a*, then the same remote party will not have access to that cookie when the user visits site *b*. Instead the remote party must set a new cookie *B*. If the cookie *A* is needed to restore preferences when returning to site *a*, for example, then it is still available to site *a*. However, the remote party cannot use the cookies to identify a Milkuser across the sites.

Above we described how unintentionally clicking a YouTube ad can result in a stored cookie even if third-party cookies are disabled. With Milk, this cookie will not be returned to the advertiser when the user visits another page, for example, `nytimes.com`. This is because the cookie was not set while the user was visiting the `nytimes.com` domain.

Milk has a second important feature: when a user logs into a site, the cookies associated with that action are available across all domains on the Internet. This policy allows users to use third-party authenticators. Logging into YouTube, for example, will authenticate the user to Gmail (since both sites rely on a `google.com` cookie).

Below we present our argument for functional privacy and the operational details of Milk in greater detail.

2 Background

Companies can employ numerous techniques for tracking users across domains, including HTTP cookies, User Agent strings, and IP addresses. Most commonly, companies have adopted the use of cookies. We focus on cookies for clarity, and we expect that our general approach is adaptable to other tracking mechanisms.

Cookies are name-value pairs used by websites as persistent storage across inherently stateless web connections. While they are often used to store innocuous information such user preferences or authentication tokens, companies can also use them to store tracking information.

Cookies are set using either HTTP headers or Javascript. By design, cookies are automatically sent to the domain that set them whenever there is a web request to that domain. As a result, advertising services embedded across domains, such as those offered by DoubleClick, can set a cookie when a user visits one domain and read it when they visit another. DoubleClick can then track the user across domains.

Cookies are explicitly set and retrieved by specific domains. For example, cookies set by Facebook.com are only retrievable by Facebook.com. The context in which a cookie is set determines the domain. Cookies set through iframe web requests, are bound to the third-party domain. Conversely, cookies set through embedded JavaScript, not inside of an iframe, are bound to the first-party domain. Therefore, services like Google analytics, which require the site owner to embed a script, set first-party cookies for the current site.

2.1 Disabling Third Party Cookies

All popular browsers give the user the option to disable third-party cookies. This approach is not completely effective, as we have noted, and it also risks breaking important functionality.

As reported by Roesner et al. [11], third-party cookie blocking is implemented differently in different browsers, with most preventing the setting of third-party cookies but not the sending. Tracking cookies may also still be set when third-party cookies are disabled. Pop-up ads, page redirects, or user ad clicks can trigger such behavior. In Chrome, we have observed that tracking cookies still are set even when third-party cookies are disabled. For example, when a user visits `weather.com` and clicks a link a tracking cookie is set, even when that link does not appear to be an advertisement.

3 Attacker Model for Functional Privacy

Before detailing our approach, we provide a brief attacker model to clarify our assumptions. We manage three actors: *users* visit Internet web sites that provide services available from *providers*. Both the providers and *monitors* (third parties that have an agreement in place with the provider) are able to monitor the user's actions via cookies and related web mechanisms.

We make an important distinction between personal information that users choose to *relinquish* (for service) versus information unknowingly *procured* from them. When a user visits Facebook, she takes conscious action to upload photos, communicate with friends, and otherwise post personal information. However, when that

Published in Proceedings of USENIX Workshop on Hot Topics in Securit, 2012

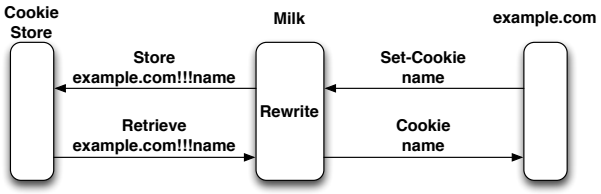

Figure 1: Milk intercepts all HTTP headers containing cookies and rewrites the keys to record which domain set the cookie. When the browser retrieves cookies to send to a server, Milk will remove any cookies that were not set by that domain. This approach effectively isolates cookies set by different domains, preventing information leakage.

same user visits different domains, say `apple.com` and `nytimes.com`, she is unknowingly tracked by advertising companies. Her information is thus procured.

When either a provider or monitor is using information relinquished by the user, then in our model the user's privacy is not violated. If the information has been procured, then the user's privacy has been violated.

We define the requirements of functional privacy as (1) users relinquish only the minimal information required for the services that they desire; (2) when faced with a choice, functional privacy tools relinquish more information rather than losing access to a service.

The utility of our model depends largely on the needs and constraints of the average user. For the purposes of this paper's argument, we consider an average user to be an adult Internet user who is primarily concerned about pervasive consumer tracking. Users living under stronger adversaries may desire privacy guarantees that are unachievable using functional privacy approaches.

Some privacy tools do seek to preserve functionality, and the best of those tools already meet our goal (to some extent) of not reducing service while increasing privacy. Roesner et al's ShareMeNot [11] is a good example: the tool blocks all cookie requests that support "social widgets" (e.g., Facebook's *like* button) until the user has specifically clicked the associated button. The tool is in a sweet spot where all privacy and functionality are maintained. In particular, the power of Roesner's approach is that privacy is reduced only in response to explicit user actions. Similarly, HTTPS Everywhere [5] enables secure web retrieval whenever it is supported by a web site, otherwise suffering the user to connect via unencrypted connections. The goal of our paper is: 1) to demonstrate that this paradigm extends beyond these two examples; 2) to argue that it is better for tools to have a designed default of always decreasing privacy rather than functionality. The advantage is that the user will never disable privacy controls.

4 Cookies are Better with Milk

We created a Chrome extension, called *Milk*[1], to illustrate the concept of functional privacy. Milk is designed to limit cross-site tracking without reducing functionality for the user. The general idea is to restrict, or *bind*, cookies to the first-party site from which they were created rather than disabling cookies entirely.

Milk implements cookie binding by intercepting cookies both before they are stored and before they are sent. Figure 1 illustrates a simplified example using a single first-party site. When the user visits `example.com`, the server responds with an HTTP `Set-Cookie` header instructing the browser to store a cookie. Milk will rewrite the cookie before it is stored, appending the key "`example.com!!!`" to the cookie's name. For subsequent web requests to `example.com`, Milk will rewrite the HTTP Cookie header to both remove the domain-specific keys and ensure that only cookies with the appropriate key are sent back to the server.

Milk also intercepts and rewrites cookies set by JavaScript's `document.cookie`. This process is similar to how Milk handles HTTP headers, and thus, we do not provide an additional figure.

4.1 Stifling Third-Party Monitors

Figure 2 demonstrates a more complicated scenario involving a third-party monitor. In this scenario, `example.com` has an agreement with third-party monitor `doubleclick.net`. When the user visits `example.com`, `doubleclick.net` will set a tracking cookie. As we described above, Milk will rewrite the `doubleclick.net` cookie and append a key specific to `example.com`.

If the user then visits `example1.com`, which also embeds `doubleclick.net`, Milk will prevent the `doubleclick.net` cookie from being sent. As a result, `doubleclick.net` will set a second cookie that Milk will then bind to `example1.com`. In short, Milk forces third-party monitors to set different cookies for each first-party domain.

4.2 Third-Party Authentication

Many websites allow users to authenticate with third-party services. For example, StackOverflow allows users to sign on using their Facebook accounts. Milk enables this functionality using a special *root cookie store* with cross-domain privileges.

Figure 3 illustrates a scenario involving the root store. In this example, Milk recognizes that the user signed into Facebook and promotes the Facebook cookies to the root

[1]The tool and its source are available at `http://forensics.umass.edu/milk.php`.

Published in Proceedings of USENIX Workshop on Hot Topics in Securit, 2012

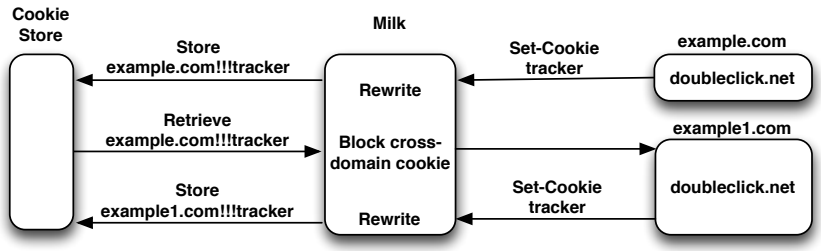

Figure 2: When a third-party monitor sets a tracking cookie, Milk prevents the browser from sending that cookie to other sites using the same tracking service. In the above figure, `doubleclick.net` sets a tracking cookie via `example.com`, which the browser would normally send back when the user visits `example1.com`, allowing the monitor to reconstruct user browsing history. Milk blocks the outgoing cookie header, preventing the monitor from reconstructing history.

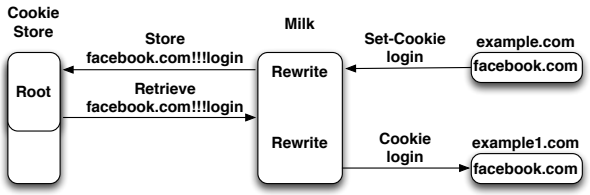

Figure 3: Milk recognizes login pages and automatically promotes their associated cookies to a "root store" with cross-domain privileges. In the above figure, both `example.com` and `example1.com` are using Facebook credentials for authentication.

store, thereby allowing the appropriate cookies to be used for authentication.

A similar example is the use of social widgets such as the Facebook *like* button. Since cookies in the root store are available across domains, Facebook could still use the *like* button to track the user across domains. There are two viewpoints on this scenario.

The strict functional privacy viewpoint is that the Facebook service *includes* the *like* button. It is a very popular feature. For example, over 56 million users have *liked* Eminem, and many other brands and icons have millions of likes. The cost of that service is user tracking wherever the button appears, but if Milk prevented the like button from working, some users would turn the tool off.

A more nuanced viewpoint that is compatible with functional privacy is that Facebook's like button can be supported without relinquishing so much information to Facebook. The ShareMeNot extension described earlier provides exactly this function. Our tests show that Milk is compatible with ShareMeNot. We could integrate the functionality natively in Milk, but in the meantime, tech-savvy users aware of the issue can install both extensions.

4.3 Limitations

Milk is designed to reduce cross-domain tracking through HTTP cookies, and as such, it will not protect against other privacy vulnerabilities such as IP address tracking, Flash objects, or intra-site tracking. Our focus on cookies is an example of our approach. We expect that complementary solutions are possible within these other spaces in future work.

Overwriting cookies is not the most elegant implementation of Milk; however, the Chrome extension API only provides limited functionality for working with cookies. We believe that the best solution would be to integrate Milk's behavior directly into the browser's cookie store.

5 Other Examples of Functional Privacy

5.1 ShareMeNot

Roesner's ShareMeNot extension for Mozilla Firefox and Google Chrome prevents third-party tracking by services that offer embeddable social sharing buttons, including Facebook, Google+, and Twitter [11]. These services normally track users across domains because they receive any login cookies from the user's browser each time the user visits a site that embeds one of the buttons. Logging out of these services is not an effective defense against this type of tracking because the service can still consider a user to be logged out without actually deleting the corresponding session cookie.

ShareMeNot stops this type of tracking by intercepting cookie headers destined for social sharing sites and blocking them, much like Milk. ShareMeNot is an example of functional privacy because it automatically re-enables social sharing buttons and allows them to send cookies if a user actually clicks on one. This feature makes the process transparent to the user while increasing privacy.

While ShareMeNot is an excellent tool, and one that provides more conservative protection than Milk for so-

Published in Proceedings of USENIX Workshop on Hot Topics in Securit, 2012

cial sharing sites, it does have limitations. Chief among these is the fact that ShareMeNot uses a blacklisting approach. While it currently supports cookie blocking for eight popular services, adding new services is a manual process that depends on how the monitor implements its embeddable button.

5.2 HTTPS Everywhere

HTTPS Everywhere is a browser extension jointly developed by the Electronic Frontier Foundation and The Tor Project for both Mozilla Firefox and Google Chrome. The extension transparently rewrites HTTP requests to use HTTPS when possible [5]. By using a whitelisting approach via per-domain rulesets, the extension avoids breaking sites that do not support HTTPS but improves user resistance to eavesdropping when using a variety of services including Facebook, Twitter, and a variety of Google services. It is also valuable to any user on an unencrypted WiFi network.

HTTPS Everywhere's whitelisting approach is fundamental to its usability. Attempting to rewrite all HTTP headers would incur an inevitable usability cost. Even if the extension failed open when it was unable rewrite requests, it would introduce delays likely noticeable to the user. The downside of the whitelist is the fact that users miss out on opportunities to use HTTPS when navigating to sites for which no ruleset exists.

5.3 User-Agent Entropy

Some tracking services forgo tracking cookies and instead gather user-identifying information by inspecting other information sent by browsers. The EFF created a system called Panopticlick that quantifies the uniqueness of the average user's browser. Panopticlock inspects browser information, including User-Agent strings, fonts, and plug-ins, to estimate how many bits of uniqueness the combination contains. The EFF estimates that these fields represent an average of 18.1 bits of information, meaning that only one in about $300,000$ other browsers is identical [4]. Much of the information used to fingerprint browsers cannot be changed arbitrarily without sacrificing functionality.

If the browser misreports Java or Flash plug-in support, for example, it will break many websites. One of the pieces of information that can be changed without impacting functionality is the User-Agent string—as long as there are no drastic changes such as reporting IE instead of Webkit. There are browser extensions available that automatically add entropy to User Agent strings to prevent the sending of identical information to multiple sites [12]. While it may be evident to monitors that the User-Agent string has been altered [4], they cannot conclusively identify a single user across multiple visits to the same site.

6 Existing Solutions are Insufficient

Privacy advocates have proposed and implemented myriad solutions that attempt to tackle the problem of user tracking on the Internet. The tracking problem has also garnered increased attention from both governments [3,9] and the media recently. Despite these efforts, we argue that no existing technical solution directly addresses the problem without breaking popular and important Internet services and that the government focus on informed consent or opt-out mechanisms such as the "Do Not Track" HTTP header [2] is unlikely to make a major impact.

6.1 Informed Consent

Informed consent policies suffer from what Nissenbaum terms the *transparency paradox* [10], which highlights the tension between detailed, meaningful privacy policies and user ability to read and comprehend the policy. Simple, easily understood privacy policies cannot capture the complexity of how many companies handle user data and under what circumstances that data may be shared. Complex privacy policies may capture the full range of implications for user data, but users are unlikely to read them.

Researchers have estimated that reading all of the web privacy policies that an average user agrees to in a single year would require 200–250 hours, or more than one month of 40-hour work weeks [8, 13]. Complicating the application of informed consent further, privacy policies may change without any notice under current law in many jurisdictions including the United States. Finally, creating separate regulations in different legal jurisdictions gives little hope of consistent expectations for users or for companies making a good faith effort to conform to consent rules.

6.2 Do Not Track Headers

Some large organizations including Mozilla, Yahoo!, and the U.S. government officially support the use of "Do Not Track" HTTP headers as a solution to Internet tracking. Mozilla, for example, has already implemented the header in some versions of the Firefox browser but how websites should respond to receipt of the header is an open question. One representative of a major trade group, the Digital Advertising Alliance, told the New York Times that the group sees the Do Not Track header as, "forbidding the serving of targeted ads to individuals but not prohibiting the collection of data". Yahoo! has taken the same stance on the issue [14].

Published in Proceedings of USENIX Workshop on Hot Topics in Securit, 2012

Recently, Microsoft announced that the Do Not Track header would be enabled by default in Internet Explorer 10 [6]. The Do Not Track working group quickly responded by drafting a modified specification requiring the header to be strictly opt-in — leaving Internet Explorer 10's compliance in question.

If governments were to mandate that the Do Not Track header must prevent websites from setting cookies that track users across domains, the header would be functionally equivalent to Milk. Unfortunately, the process of standardization is far from complete and it is unclear whether governments are willing to legislate on the issue.

6.3 Strict Technical Solutions

On the other end of the defensive spectrum are the many technical solutions that make major changes to user experience in the name of privacy. Some solutions from this category include disabling cookies completely, routing traffic through Tor, or running a variety of content-blocking browser extensions such as NoScript, or AdBlockPlus [1, 7]. While a technically sophisticated user may understand how to effectively use these techniques, they have the potential to break important functionality. NoScript, for example, which is not intended primarily as a privacy protection tool but is often used as such, breaks many popular or important services — including Facebook, Pandora and Verified by Visa. Users must identify broken pages and manually whitelist domains to re-enable functionality.

While this approach may be feasible for a small number of advanced users, it requires a level of understanding and interaction that cannot reasonably be expected of average users. There is a long history of research from the usable security community highlighting how unlikely users are to make conservative security or privacy decisions.

7 Conclusions

We propose the adoption of functional privacy as a guiding principle in the development of new privacy tools. Functional privacy has the overarching goal of maintaining functionality while improving privacy as much as practically possible — rather than forcing users to make decisions about tradeoffs that they may not fully understand.

As a concrete example of a functional privacy approach, we implemented Milk, a Google Chrome extension that automatically rewrites HTTP cookies to bind them to the domains from which they were set. By binding cookies to specific domains, Milk stops a cookie set by one domain from being retrieved by another, preventing third-parties from surreptitiously tracking users across sites. Milk maintains a root store of cookies that are allowed to be used across domains. Cookies from any site where the user completes a password-protected log in are added to the root store. This simple mechanism allows the tool to determine for which sites the user prefers full functionality.

Acknowledgments

This work was supported in part by a National Science Foundation Graduate Research Fellowship and in part by a grant from the UMass President's Office. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

References

[1] Adblock plus — for annoyance-free web surfing. `http://adblockplus.org/en`, Apr. 2012.

[2] Do not track - universal web tracking opt out. `http://donottrack.us/`, 2012.

[3] British Broadcasting Corporation. 'Action needed' to meet UK's cookie tracking deadline. `http://www.bbc.com/news/technology-17745938`, Apr. 2012.

[4] P. Eckersley. How unique is your web browser? In *Proc. Intl Conf on privacy enhancing technologies (PET)*, pages 1–18. Springer-Verlag, 2010.

[5] Electronic Frontier Foundation. HTTPS everywhere. `https://www.eff.org/https-everywhere`, 2012.

[6] B. Lynch. To track or not to track? not just a question, a choice for consumers and industry. `http://blogs.technet.com/b/microsoft_on_the_issues/archive/2012/06/08/to-track-or-not-to-track-not-just-a\-question-a-choice-for-consumers-and-industry.aspx`, June 2012.

[7] G. Maone. Noscript - javascript/java/flash blocker for a safer firefox experience! - what is it? - informaction. `http://noscript.net/`, Apr. 2012.

[8] A. McDonald and L. Cranor. The cost of reading privacy policies. *I/S: A Journal of Law and Policy for the Information Society*, 2008 Privacy Year in Review issue, 2008.

[9] J. Melvin. U.S. regulators push for online "do not track" system. `http://www.reuters.com/article/2012/03/26/net-us-internet-privacy-ftc-idUSBRE82P0PF20120326`, Mar. 2012.

[10] H. Nissenbaum. A contextual approach to privacy online. *Daedalus, the Journal of the American Academy of Arts & Sciences*, 2011.

[11] F. Roesner, T. Kohno, and D. Wetherall. Detecting and Defending Against Third-Party Tracking on the Web. In *Proc. USENIX NSDI*, 2012.

[12] J. Tantalo. tantalor/user-agent-entropy. `https://github.com/tantalor/user-agent-entropy`, Loaded May 7, 2012.

[13] S. Vedantam. To read all those web privacy policies, just take a month off work. `http://www.npr.org/blogs/alltechconsidered/2012/04/19/150905465/`, Apr. 2012.

[14] E. Wyatt and T. Vega. Conflict Over How Open 'Do Not Track' Talks Will Be. `http://www.nytimes.com/2012/03/30/technology/debating-the-path-to-do-not-track.html`, Mar. 2012.