Several Pressing Problems in Hypertext Privacy

Peter Eckersley*

November 5, 2010

Current technical discussions of Internet privacy are often discussions about the hypertext protocol stack and things that people are doing with it: HTTP, HTTPS, JavaScript, HTML5, Flash/Silverlight/JVM, social networking and (partially) geolocation. Although there are other subjects that probably deserve just as much attention — such as wireless protocols and mobile devices — this position paper will survey some of the serious unsolved privacy problems in the hypertext stack. In some cases, practical technical solutions may be available.

This position paper will not categorise all of the problems and defensive responses mentioned by the types of privacy adversaries they may apply to (which include Web service providers, advertising and tracking firms, governments of various sorts, individual employees of the aforementioned organisation types, employers, family members, and nearby and distant "black hat" hackers). The reader is encouraged, as always, to keep these considerations in mind.

1 HTTP

Embarrassingly, we still lack any systematic way to distinguish which HTTP cookies are necessary for stateful user interface features, and which exist for the sole purpose of tracking people's reading habits. Default policies for cookie acceptance and retention from 3rd party hosts remain unacceptable. Browser user interfaces for cookie control remain impractical and in some instances "malicious" [4] even for experts.

The HTTP Referrer field has emerged as one of several mechanisms that are being used to link users' identities to their reading habits across the Web [11]. At the very least, the referrer should no longer be sent for any cross-domain HTTPS request.

With a handful of exceptions, the privacy warnings in RFCs 1945 and 2616 about server logs[1] have been forgotten or ignored. Almost all webservers retain persistent comprehensive logs of user requests, and most embed 3rd party content that exists for the sole purpose of copying those logs to tracking and advertising firms' systems.

The User Agent field has turned out to be a problematic source of partially identifying entropy in HTTP clients [6, 8] — more than 10 bits of identifying information on average. Despite the small cost in retrospective debuggability, clients need to migrate towards having low-entropy User Agent strings by default.

*Electronic Frontier Foundation, pde@eff.org

[1] http://www.w3.org/Protocols/rfc2616/rfc2616-sec15.html#sec15.1.1

2 HTTPS

HTTPS is an essential component in all of our hypertext privacy and security efforts. The most pressing privacy problems in HTTPS relate to server-side deployment and operational issues, but there are also some serious issues in the protocol and widespread client implementations.

The largest current problem with HTTPS is the widespread failure to understand the requirements for secure HTTPS deployment. It is only a slight oversimplification to state that unless all of the following conditions are met, an HTTPS deployment will fail to meet some or all of its privacy objectives:

- HTTPS must cover the *entire* DNS domain on which it is deployed;
- All cookies must have the "secure" flag set;
- All content included from the domain to be secured must be delivered; over HTTPS and must come from trustworthy domains;
- An auxiliary mechanism like HSTS [10] or HTTPS Everywhere [7] is used to ensure that the browser never attempts to make HTTP connections to the domain to be secured.

We need better tools to inform webmasters as clearly as possible when they have failed to meet these hurdles.

Even when HTTPS is deployed correctly, we have some growing reasons to be concerned about the infrastructure the protocol relies to verify the identity of the servers [9, 18]. The space of possible responses to this problem is complicated, but they certainly include multi-path certificate comparison [19] and the use of DNSSEC as cross-check for CA certification [17].

3 JavaScript

Hypertext needed a standard programming language, but the fact that that role was filled by JavaScript presents serious privacy problems. JavaScript's security model is deeply inadequate for the task it has been asked to perform. It is the delivery mechanism for a very large number of privacy attacks, including many supercookies [12], identity leaks [11], most high-entropy client fingerprint components [8], CSS inspection problems [1, 15], not to mention CSRF, XSS and "clickjacking" or "UI redressing" attacks.

Many security- or privacy- conscious "power users" employ browser extensions like NoScript [3] but still remain nervous about the threats posed by JavaScript of varying degrees of malice. More effort is required to offer equivalent (or better) kinds of protection for non-expert users.

4 Flash, Silverlight, Java and other virtual machine plugins

None of the widely-deployed virtual machine plugins appear to have been designed with much careful thought for privacy. They tend to create supercookies that are outside of the browser's

normal cookie UI controls; they make HTTP connections that do not respect the user's proxy settings; they expose high-entropy system settings variables such as font lists.

In general, the privacy consequences of these systems are so bad that current advice has to be 'if you want privacy, disable plugins' [14].

5 Conclusions

Retrofitting real privacy onto the hypertext standards is a daunting task, and it is doubtful that we can expect to succeed in this task when defending real-world users against real-world adversaries. Projects that take technical privacy threats seriously, such as Tor [5] or RequestPolicy [16], remain a long way from being continuously usable by most people on a daily basis.

For some categories of privacy adversaries, the best we can hope for in terms of routinely usable defaults is to improve the technical situation far enough that defensive technology could possibly "meet in the middle" with well-designed, strengthened privacy laws.

The X-Do-Not-Track proposal [2, 13] is one example of an attempt to get technical and legal/policy mechanisms to work together towards this kind of hybrid privacy objective. We should consider whether there are more possibilities of this sort.

References

[1] CSS history hack demonstration. http://www.whattheinternetknowsaboutyou.com/.

[2] Do Not Track: Universal Web Tracking Opt-out. http://donottrack.us/.

[3] NoScript. http://noscript.net/.

[4] Conti, G., and Sobiesk, E. Malicious interface design: exploiting the user. In *Proc. 19th international conference on World Wide Web* (New York, NY, USA, 2010), WWW '10, ACM, pp. 271–280.

[5] Dingledine, R., Mathewson, N., and Syverson, P. Tor: The Second-Generation Onion Router. In *In Proceedings of the 13th USENIX Security Symposium* (2004), pp. 303–320. https://svn.torproject.org/svn/projects/design-paper/tor-design.pdf.

[6] Eckersley, P. Browser Versions Carry 10.5 Bits of Identifying Information on Average. EFF Deeplinks, 2010. https://www.eff.org/deeplinks/2010/01/tracking-by-user-agent.

[7] Eckersley, P. Encrypt the Web with the HTTPS Everywhere Firefox Extension. EFF Deeplinks, 2010. https://www.eff.org/deeplinks/2010/06/encrypt-web-https-everywhere-firefox-extension (Code co-authored by Mike Perry; rulesets curated by Seth Schoen and many other contributors).

[8] Eckersley, P. How Unique is Your Web Browser? In *Proc. Privacy Enhancing Technologies Symposium* (2010), vol. 6205, LNCS. https://panopticlick.eff.org/browser-unqiueness.pdf.

[9] Eckersley, P., and Burns, J. An Observatory for the SSLiverse. Presentation at DEFCON 18, 2010. https://www.eff.org/files/DefconSSLiverse.pdf.

[10] Hodges, J., Jackson, C., and Barth, A. HTTP Strict Transport Security (HSTS). Internet-draft, IETF, July 2010. http://tools.ietf.org/html/draft-hodges-strict-transport-sec-02.

[11] Krishnamurthy, B., and Wills, C. On the Leakage of Personally Identifiable Information Via Online Social Networks. In *Proc. ACM SIGCOMM Workshop on Online Social Networks* (2009), ACM.

[12] McKinkley, K. Cleaning Up After Cookies. iSec Partners White Paper, 2008. https://www.isecpartners.com/files/iSEC_Cleaning_Up_After_Cookies.pdf.

[13] Narayanan, A. "Do Not Track" Explained. 33 bits, 2010. http://33bits.org/2010/09/20/do-not-track-explained/.

[14] Perry, M. Torbutton Design Doccumentation, 2009. https://www.torproject.org/torbutton/design.

[15] Robinson, S. Flipping Typical (demonstration of CSS font detection). http://flippingtypical.com/.

[16] Samuel, J., and Zhang, B. RequestPolicy: Increasing Web Browsing Privacy through Control of Cross-Site Requests. In *Proc. Privacy Enhancing Technologies Symposium* (2009), vol. 5672, LNCS.

[17] Schultze, S. A Major Internet Milestone: DNSSEC and SSL. Freedom to Tinker, 2010. http://www.freedom-to-tinker.com/blog/sjs/major-internet-milestone-dnssec-and-ssl.

[18] Soghoian, C., and Stamm, S. Certified Lies: Detecting and Defeating Government Interception Attacks Against SSL. In *Proc. HotPETS* (2010). `http://petsymposium.org/2010/papers/hotpets10-Soghoian.pdf`.

[19] Wendlandt, D., Andersen, D. G., and Perrig, A. Perspectives: Improving ssh-style host authentication with multi-path probing. In *Proceedings of the USENIX Annual Technical Conference (Usenix ATC)* (June 2008).