Critical Vulnerability in Browser Security Metrics

Mustafa Acer
Carnegie Mellon Silicon Valley
mustafa.acer@sv.cmu.edu

Collin Jackson
Carnegie Mellon Silicon Valley
collin.jackson@sv.cmu.edu

Abstract

Every time a browser vendor releases a patch for a critical vulnerability, the popular news media publishes a slew of negative press article detailing the security holes that have been announced in the product. Users who read these articles often decide to switch to a "safer" browser. The negative press associated with security patch releases has a number of unhealthy effects on the industry. We challenge the conventional wisdom of the current browser security evaluation paradigm: that browsers that receive infrequent security patches are safer than browsers that receive frequent patches, that browsers with a lower bug count are safer, and that reducing browser vulnerabilities is the only path that a browser vendor can follow to improve security. We argue that patch deployment matters vastly more than patch frequency, that bug count fails to take into account differences in severity and vendor reporting methodologies, and that the security features that matter most are ignored by negative news articles. We propose methods for evaluating browser security that take into account new industry best practices such as silent patch deployment and sandboxing.

1 Introduction

We are constantly barraged by news about the latest embarrassing browser exploit, and the problem seems to be growing larger every day. Browser vendors are evaluated by their ability to stay out of the news, and security vendors like Cenzic, IBM, and Symantec regularly publish reports [6, 15, 16] that damage the reputation of browser vendors with the most reported bugs. Unfortunately, these reports discourage secure browser vendor behavior by punishing proactive patching and ignoring many factors that are important for end-user security.

We argue that the most prevalent metric, the number of publicly disclosed vulnerabilities in a browser for specific interval of time, does not represent a useful measure of vulnerability for any browser. We discuss four major weaknesses of this metric and propose a new metric that more accurately represents the actual risk to users.

2 Flaws of Current Metrics

Consider the example of Cenzic, a leading security vendor, who recently released a report [5] describing the breakdown of browser vulnerabilities for the first half of 2009. The report compared the number of publicly reported vulnerabilities in browsers for that six month period and indi-

cated that Firefox was the riskiest browser with 44% of vulnerabilities. Their methodology has several major limitations:

- **Ignores Patch Deployment.** Patches that are quickly deployed have a minimal effect on user security. Silent update patch deployment technologies can have a significant effect on user security [10, 9].
- **Discourages Disclosure.** To improve perceived security, vendors often combine unrelated bugs into a single disclosure or avoid reporting them entirely, even after security patches are widely deployed [13].
- **Ignores Severity.** Improved browser security architectures such as the sandboxed renderer in Google Chrome [4] can reduce many security bugs from critical severity (arbitrary code execution) to high severity (accessing confidential data belonging to other web sites) [2], but simple bug counting does not account for severity.
- **Ignores Plug-ins.** According to Adobe, Flash Player is installed by 99% of web users [7], and a recent report [17] suggests that 80% of Flash Player users have not yet installed the latest critical security updates. Firefox includes an update check service that help the user keep Flash Player up to date, but existing metrics do not measure its effect on end user security.

None of these problems are browser-specific; they affect security evaluations of all software. However, they are particularly severe for browsers, which constantly interact with untrusted code (HTML, JavaScript, CSS, and so on). Attackers can easily run exploits on millions of browsers by buying ad impressions or compromising a popular web site [12]. It is critical that browser vendors reduce actual exploitability, rather than waging a public relations battle over metrics that have no connection to reality.

3 Our Proposal

We propose that browsers be evaluated on the percentage of users who have at least one unpatched critical-severity vulnerability (or at least one unpatched high-severity vulnerability) on an average day during the specified interval. Unless sandboxing is used to restrict vulnerabilities in plug-ins, we propose that vulnerabilities in plug-ins also count for this calculation. This metric addresses the problems described above:

- **Takes Account of Patch Deployment.** Browsers that use faster update techniques will benefit, because users will have vulnerabilities for a shorter period of time.
- **Encourages Disclosure.** Browser vendors who disclose vulnerabilities but patch users quickly will not be penalized. Combining multiple bugs into a single disclosure will not improve a vendor's score. This reflects the reality that an attacker only needs one vulnerability to exploit a user's browser.
- **Takes Account of Severity.** Separate scores for critical and high-severity vulnerabilities ensure that browsers that use sandboxing technologies to reduce the severity of vulnerabilities will receive better scores.
- **Includes Plug-ins.** Browser vendors that do an effective job of sandboxing or updating a user's plugins will receive a better score.

4 Measurement

We have begun measuring proposed browser risk metric using techniques we developed in our previous work [12, 3, 1]. Our server collects browser and plug-in version data from the web by running JavaScript advertisements on ad networks. We compare these observations against a vulnerability database compiled from browser and plug-in vendors [8, 14, 11]. We then assign a risk score to browser and plug-in combinations. We define risk score as the percentage of browsers that have at least one known critical or high vulnerability. For any given day, this score is calculated based on the vulnerabilities reported by vendors before that day.

Our preliminary results are shown Figure 1. We found that the percentage of vulnerable of browsers increases significantly if the plug-ins vulnerabilities are included in the calculation. For example, we found that only about 4% of Google Chrome users have critical or high vulnerabilities. However, 30% are vulnerable if Flash Player vulnerabilities included. Fortunately, the latest beta version of Google Chrome now includes Flash Player updates in its silent update process [18]. We expect Google Chrome's plugin-adjusted risk score to drop significantly once this beta version is rolled out to all users.

One limitation of our experimental methodology is that we are unable to evaluate the patch level of Internet Explorer. Unlike other browsers, Internet Explorer does not advertise which security patches have been applied when querying the user agent string. It might be possible to detect the browser's patch status by triggering an exploit, but this would potentially expose users to risk or interfere with their browsing session. An open research problem is how to evaluate Internet Explorer's patch level in an ethical experiment.

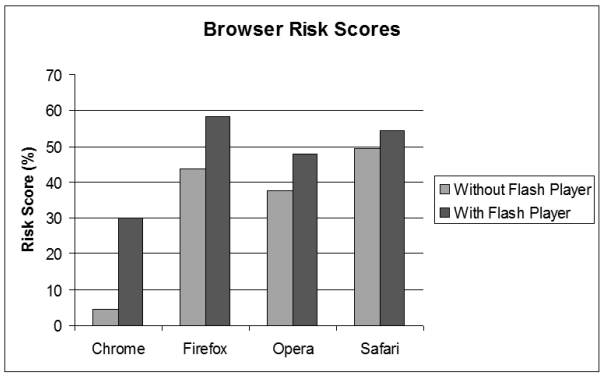

Figure 1: Average browser risk scores over a forty-day span.

5 Conclusion

We expect that adoption of more constructive metrics for browser security will increase openness and disclosure without harming user security. By evaluating the security functionality that matters most to users and web application developers, we will encourage browser vendors to innovate and compete on security features that can have the most positive impact.

References

[1] Gaurav Aggarwal, Elie Burzstein, Collin Jackson, and Dan Boneh, *An analysis of private browsing modes in modern browsers*, To appear in USENIX Security 2010.

[2] The Chromium Authors, *Severity guidelines for security issues*, `http://www.chromium.org/developers/severity-guidelines`.

[3] Adam Barth, Collin Jackson, and John C. Mitchell, *Robust defenses for cross-site request forgery*, Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS), 2008.

[4] Adam Barth, Collin Jackson, Charles Reis, and the Google Chrome Team, *The security architecture of the Chromium browser*, September 2008, Technical Report.

[5] Cenzic, *Web application security trends report*, November 2009, http://www.cenzic.com/downloads/Cenzic_AppSecTrends_Q1-Q2-2009.pdf.

[6] Inc. Cenzic, *Web application security trends report*, 2009, http://www.cenzic.com/downloads/Cenzic_AppSecTrends_Q1-Q2-2009.pdf.

[7] Adobe Corporation, *Flash usage statistics*, September 2009, http://www.adobe.com/products/player_census/flashplayer.

[8] Mozilla Corporation, *Known vulnerabilities in mozilla products*, September 2009, http://www.mozilla.org/security/known-vulnerabilities.

[9] Thomas Duebendorfer and Stefan Frei, *Why silent updates boost security*, CRITIS 2009 Critical Infrastructures Security Workshop, May 2009.

[10] Stefan Frei, Thomas Duebendorfer, and Bernhard Plattner, *Firefox (in)security update dynamics exposed*, ACM SIGCOMM Computer Communication Review **39** (2009), no. 1, 16–22.

[11] Google Inc, *Google Chrome releases*, November 2009, http://googlechromereleases.blogspot.com.

[12] Collin Jackson, Adam Barth, Andrew Bortz, Weidong Shao, and Dan Boneh, *Protecting browsers from DNS rebinding attacks*, Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS), 2007.

[13] Window Snyder, *Critical vulnerability in Microsoft metrics*, November 2007, http://blog.mozilla.com/security/2007/11/30/critical-vulnerability-in-microsoft-metrics/.

[14] Opera Software, *Opera security advisory*, September 2009, http://www.opera.com/support/kb.

[15] IBM Security Solutions, *X-Force 2009 trend and risk report*, http://www-935.ibm.com/services/us/iss/xforce/trendreports/.

[16] Symantec, *Internet security threat report*, 2010, http://www.symantec.com/business/theme.jsp?themeid=threatreport.

[17] Trusteer, *Flash security hole advisory*, August 2009, http://www.trusteer.com/files/Flash_Security_Hole_Advisory.pdf.

[18] Linus Upson, *Bringing improved support for adobe flash player to google chrome*, March 2010, http://blog.chromium.org/2010/03/bringing-improved-support-for-adobe.html.