Firefox (In)Security Update Dynamics Exposed

Stefan Frei
stefan.frei@
tik.ee.ethz.ch

Thomas Duebendorfer
thomas.duebendorfer@
google.com

Bernhard Plattner
plattner@
tik.ee.ethz.ch

Swiss Federal Institute of Technology (ETH) / Google Switzerland GmbH
Zurich, Switzerland
http://www.techzoom.net/risk

ABSTRACT

Although there is an increasing trend for attacks against popular Web browsers, only little is known about the actual patch level of daily used Web browsers on a global scale. We conjecture that users in large part do not actually patch their Web browsers based on recommendations, perceived threats, or any security warnings. Based on HTTP user-agent header information stored in anonymized logs from Google's web servers, we measured the patch dynamics of about 75% of the world's Internet users for over a year. Our focus was on the Web browsers Firefox and Opera. We found that the patch level achieved is mainly determined by the ergonomics and default settings of built-in auto-update mechanisms. Firefox' auto-update is very effective: most users installed a new version within three days. However, the maximum share of the latest, most secure version never exceeded 80% for Firefox users and 46% for Opera users at any day in 2007. This makes about 50 million Firefox users with outdated browsers an easy target for attacks. Our study is the result of the first global scale measurement of the patch dynamics of a popular browser.

Categories and Subject Descriptors

C.4 [**Measurement techniques**]: Miscellaneous ; C.2.3 [**Network monitoring**]: Security

General Terms

Security, Measurement

Keywords

Web browser, Update

1. INTRODUCTION

Web browsers are the most frequently used client application in the Internet. There is an increasing threat from attacks that take advantage of vulnerabilities in popular browsers such as Internet Explorer or Firefox. A visit to a single malicious Web site with an unpatched browser is enough to get compromised. Such drive-by download attacks potentially result in millions of infected computers. In 2007, Google has found more than three million malicious Web sites that initiate drive-by downloads [14].

How can we measure the patch level of a large population of Internet browsers without access to millions of end-user computers? In this paper we present a new methodology to measure the patch level of Firefox without the need to access the computer of the end-user or to install any kind of monitoring software on it. Our key idea is to exploit the information in the HTTP *user-agent* string in order to measure the distribution and evolution of the patch level of the Firefox population. This information is readily available in the log files of most Web servers. For this research we analyzed anonymized log files of Google's search and application servers, covering 75% [10] of the world's Internet users for more than a year. We measured the distribution and dynamics of major and minor version updates in the Firefox population. We focused on Firefox as this is the most popular browser with detailed version information available in the HTTP *user-agent* string. Internet Explorer only reveals the major version of the browser, which does not reflect the patch level. The main contributions of this paper are the presentation of a new measurement methodology and an analysis of the update process of a large population of Firefox users. Firefox' auto-update mechanism is found to be very effective when enabled: most users update to a new version within three days. However, the maximum share of the latest most secure version of Firefox never exceeded 80% at any day throughout 2007. The maximum share of the *latest minor versions* is limited by the much slower migration speed *between major versions* of the same Web browser. Compared with Opera, another free browser with a different update strategy, we found that the design and ergonomics of the update mechanism plays a key role for good patch performance. We found that the severity of a vulnerability has no measurable influence on the patch adoption speed. Further more, the end-of-life (EOL) of a Web browser version has a much smaller effect on the migration to the next version than bundling the software with a new operating system version. In large part users are not immediately reacting if new security vulnerabilities are disclosed.

2. RELATED WORK

To measure patch adoption rates, Qualys analyzed the results from over 30 million scans of their vulnerability scanning service. Qualys distinguished between external (Internet facing) and internal machines and calculated how long it takes on average to patch against a known vulnerability. In their paper *"Law of vulnerabilities"* [16] they find the half-life of vulnerabilities to be 19 days on external and 62 days on internal systems. Secunia reports patch level statistics of Web browsers as a by-product of their Personal Software Inspector (PSI) [17], a vulnerability scanning tool. Based on the analysis of 200,000+ installations of PSI, Secunia found that 5.19% of Firefox, 11.96% of Opera, 9.61% of Internet

Explorer (IE) 6, and 5.4% of IE7 installations miss recent security updates. However, this data reflects the state of users that deliberately installed a security tool, a minor and biased part of the global population of browser users. In our approach we exploit the information a browser transmits in the HTTP *user-agent* [2, 4] string with every page request. The *user-agent* string usually identifies the type of browser, the operating system used and the patch-level. Using our methodology and Secunia PSI data we analyzed the global vulnerable Web browser problem in [5] and found the share of insecure Web browsers used to daily surf the Internet to be 45%. To date the *user-agent* information is mainly used to determine market shares of different browsers or operating systems. The sample size of these measurements and the numbers published vary considerably between the sources [7, 9, 13, 18]. Meanwhile attackers are known to use the *user-agent* string not for measurement but for precise exploit targeting [1, 12] and code injection [11]. Our method allows us to measure the patch adoption rate of Firefox users on a global scale, based entirely on Web server log files. Further, our measurement is not biased as no specific user interaction is needed. To the best of our knowledge, this is the first paper to exploit the information of the *user-agent* string to measure the Web browser patch adoption rate of end-users on a truly global scale.

3. METHODOLOGY

A Web browser sends the *user-agent* string in the HTTP protocol header with every request for a Web page. This string contains the *type* and *version* of the browser and the type of *operating system* the browser runs on [2, 4, 8]. To measure the number of unique browser installations active on a given day we needed a way to reliably remove duplicates resulting from multiple visits to a Web site. Relying on the client's IP address is not sufficient as a large user base surfing behind proxies is seen through a single IP address only. For our study we relied on Google's PREF cookie to eliminate duplicate visits by the same browser. We ignored the small fraction of browsers that disabled cookies due to restrictive user settings. We also ignored the small possibility of cookie id collisions and the effect of users deleting cookies manually. While Google offers its Web search also anonymously without requiring cookies, by default users allow cookies to remember their preferences. To protect user's privacy, Google was the first search engine to publicly commit back in 2007 to anonymize cookies and IP addresses in logs after 18 months and to shorten the cookie expiration time to two years. In September 2008, Google announced to anonymize logs already after 9 months. Some browsers and browser extensions allow the user to modify the information in the *user-agent* string and there are Web spider tools that impersonate a widespread browser for compatibility. There are also some proxies that change the *user-agent* string. Based on the observed dynamics of this research we expect this effect to be small. We continuously analyzed Google's Web server log data from January 2007 to April 2008 with typically three samples per week, namely Monday, Wednesday and Saturday (Pacific time zone GMT-8). Starting Oct 10th, 2007 we used daily statistics.

The parser was implemented as a MapReduce [3] that ran on hundreds of machines and processed more than a year of anonymized Google Web server logs. It identified and counted known user-agent strings for Firefox, Internet Explorer, Safari, and Opera. In sum, all other user-agent strings accounted for less than 1% of the share and were mostly attributed to non mainstream browsers, mobile devices, automated tools, and proxies.

4. MAJOR VERSION DYNAMICS

To assess the dynamics of Firefox (FF) major and minor updates we first measure the transition between the most recent major versions within our observation period. The migration to the next major version of a browser usually requires a manual installation. Minor version updates are highly automated, depending on the type of browser. For comparison, we also measure major version migrations of Microsoft's *Internet Explorer* (IE), Apple's *Safari* (SF), and *Opera* (OP). Mozilla released FF2 in October 2006. There were 13 updates of minor versions for FF2 and three updates for FF1.5 released in our observation period from January 2007 to April 2008. In the same period, Apple released the new major version 3.x of its Safari browser. However, Microsoft (IE7) and Opera (OP9) released their most recent major versions before our observation started and Google's Chrome browser and Mozilla's Firefox 3.0 were released thereafter.

In Figure 1 we plot the evolution of the shares of the major versions of these browsers relative to the total share of a given type of browser (all versions). Table 1 lists the release dates of major versions of these browsers. For all browsers we observe a high frequency component on top of the much slower migration rate between major versions. This component is most pronounced in IE. We call this high frequency component the *weekend-effect*, as its periodicity follows exactly the weekly workday/weekend pattern throughout the year. It can be explained by different preferences of end-users for major browser versions at work during the week and at home on the weekend. For example, IE7 consistently has a higher share at the weekends than during working days while the opposite is true for IE6. This relation is distorted over Christmas at the end of the year, supporting our interpretation. Since the end of December 2007 we observe a greater amplitude in the weekly pattern of IE6 and IE7. This could be explained with a sizable part of the IE population migrating over Christmas to new computers having Windows Vista and IE7 pre-installed. We find the weekend-effect for all major versions within the Firefox, Internet Explorer, Safari, and Opera population. Interestingly, we also found a *cross vendor weekend-effect* between Firefox and Internet Explorer, with Firefox being preferred over the weekend at the cost of Internet Explorer share [6].

On May 30th, 2007, FF1.5 reached the end-of-life (EOL) and Mozilla delivered the last security patch for FF1.5. Support should have ended in April but Mozilla extended the lifetime of FF1.5 in order to put in place a mechanism to allow FF1.5 users to upgrade to the latest FF2 release as soon as the upgrade package is available. This delivery mechanism was included in version 1.5.0.12, the last update for FF15 on May 30th. Alternatively, users could manually upgrade to FF2 any time using Mozilla's download Web site.

The impact of the EOL on the shares of FF2 and FF1.5 on May 30th is visible but smallish. Relatively few users chose to manually upgrade to FF2 as a result of the EOL of FF1.5. The release of the package to automatically upgrade FF1.5 to FF2 on June 29th, 2007, has a much bigger impact as seen in Figure 1. FF2 surpassed the combined share of FF1

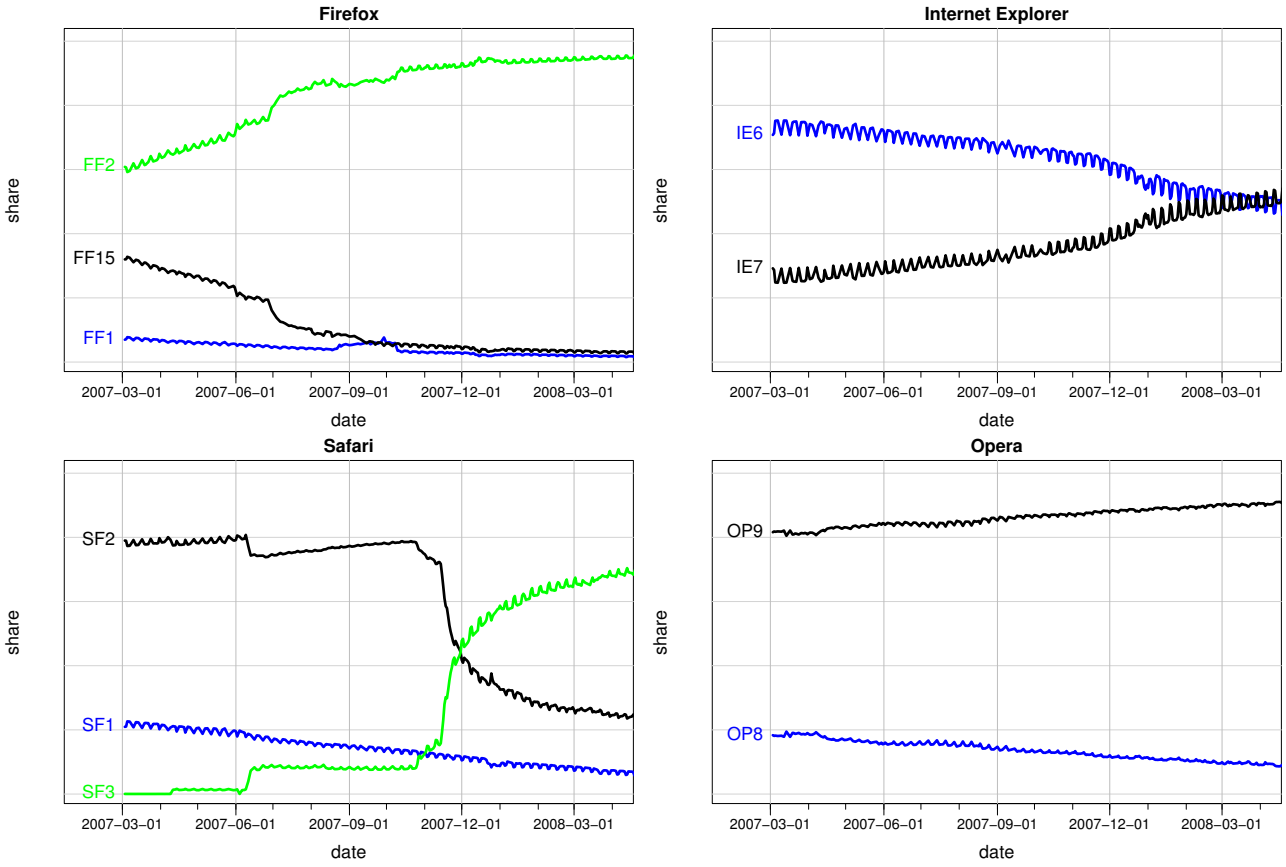

Figure 1: Percentage of major versions of the Firefox (FF), Internet Explorer (IE), Apples Safari (SF), and Opera (OP) browser over one year. 100% is the total of all versions of the respective browser. Major versions below 1% are omitted for clarity.

and FF1.5 at the end of January 2007, about 15 month after its initial release (FF1.5 is considered as a major release).

Throughout our observation period, Microsoft supported both IE6 and IE7. We found no major discontinuities in their adoption rate. In March 2007 IE7 surpassed the share of IE6, 18 month after its initial release.

Apple released the first public beta of their Safari browser SF3 for Mac and Windows users on June 11th, 2007, which correlates with the first large increase in the SF3 share at the cost of SF2. Apple seems to have an enthusiastic beta tester community that readily adopted SF3 beta (gaining almost 10% in 3 days). Bundled together with Apple's new operating system Mac OS X Leopard (10.5), SF3 increases its market share on October 26th, 2007. However, the fastest rise in SF3 market share starts on November 16th, 2007, when SF3 is bundled with an update of the then prevalent Mac OS X Tiger (10.4).

SF3 surpassed the share of SF2 in the last days of November 2007, which is very fast compared with 18 and 15 months for IE and FF respectively. However, in contrast to IE and FF, the share of SF1 was still above 10% at that time. The Opera community is only slowly migrating to the next major version. More than 18 months after the release of OP9 we still found more than 10% share of OP8, which is no longer supported.

Browser	Version	Released
Internet Explorer	6.0	2001-08-27
Internet Explorer	7.0	2007-10-18
Firefox	1.0	2004-11-09
Firefox	1.5	2005-11-29
Firefox	2.0	2006-10-24
Safari	1.0	2003-06-23
Safari	2.0	2005-04-19
Safari	3.0	2007-10-26
Opera	8.0	2005-04-19
Opera	9.0	2006-06-20

Table 1: Release of major browser versions.

We find that bundling browser updates with existing automated update mechanisms has a major impact on migration speed. Upgrading to the next major version of a browser software is often intentionally delayed due to compatibility issues with critical e.g. intranet applications.

5. MINOR VERSION DYNAMICS

In Figure 2 we plot the detailed update dynamics of minor versions of the free browsers Firefox and Opera released in 2007. We compare Firefox and Opera because both browsers are available for free, both are capable to run on multiple operating systems, and both are independent of any operating system vendor. Both browsers provide minor version infor-

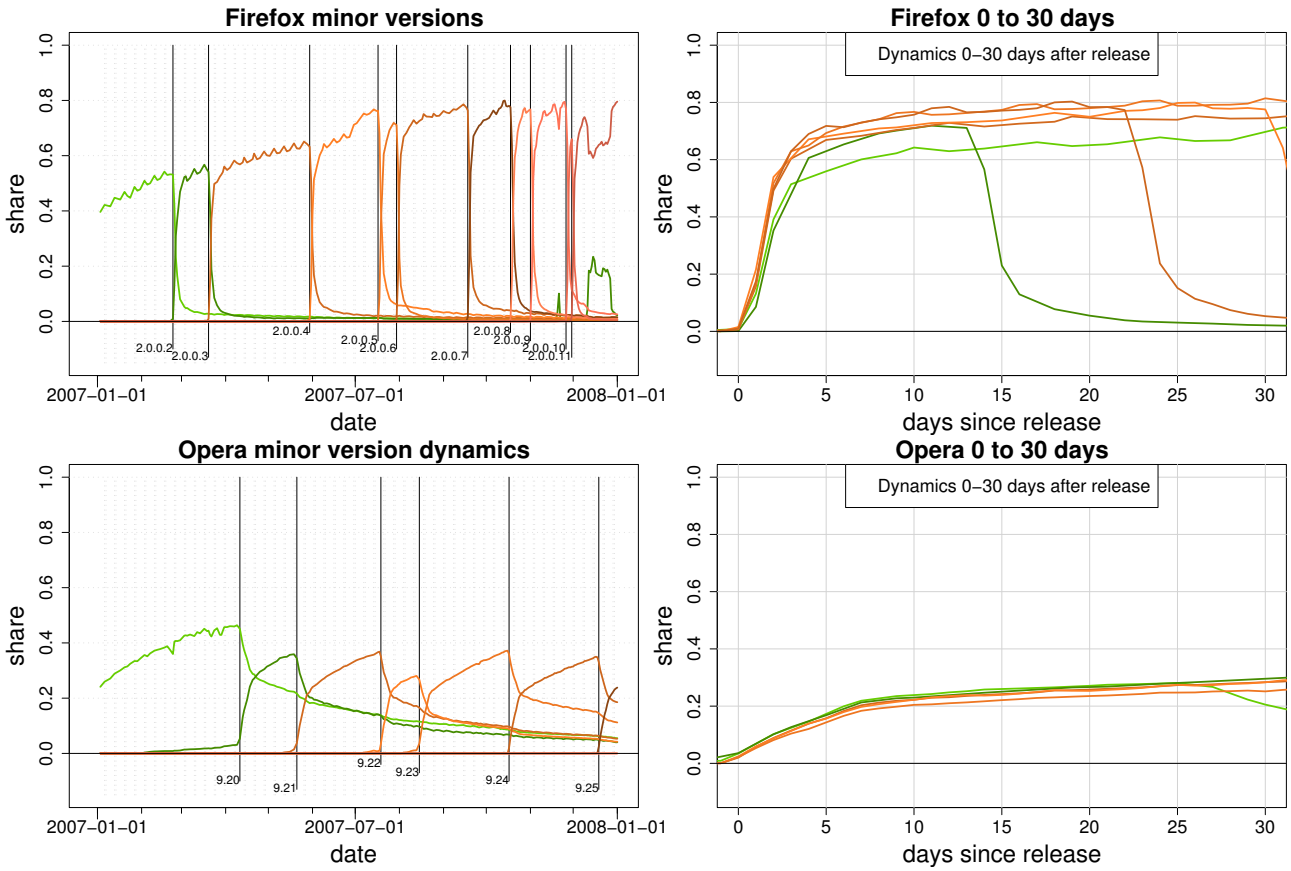

Figure 2: ***Left:*** **Percentage of minor version updates for Firefox and Opera in 2007, vertical lines depict the release of new minor versions.** ***Right:*** **Percentage of the share normalized to the release date $t = 0$ of the new version for the first 30 days (right). 100% is the total of all versions of the respective browser. Some versions got replaced before 30 days.**

mation in the user-agent string, unlike Internet Explorer. Essentially, we measure the delay between the availability of a new update and the time it is installed by users. We found two distinct regimes of the adoption rate for both browsers: a very fast initial rise followed by a much slower continuing adoption thereafter. Firefox users adopt fast, the majority of users installed a new patch within 3 days after its release while the last minor version loses share equally fast. Thereafter the adoption rate is limited by the much slower migration of users *between major versions* of Firefox from FF 1.x to FF 2.x, as shown in Figures 1 and 2. Minor versions (N) and $(N-1)$ (with (N) being the most recent version at any time) clearly dominate the dynamics of Firefox updates. The weekend is also visible as a high frequency oscillation in the update dynamics of minor versions. There is a striking difference to the update dynamics within the Opera population. As with Firefox, we observe two distinct phases in the adoption rate. However, the initial adoption is much slower for Opera compared to Firefox. On average the first fast initial rise phase is about 11 days for Opera users and the share of the most recent version saturates at about 40%; well below the level achieved by the Firefox population.

Further, older versions $(N-1, N-2, ..)$ persist remarkably long after the release of version (N). After an initial fast decay, further loss of share of these older versions is very slow within the Opera population. This means that a considerable part of the Opera users stick to older, insecure versions of their browser.

In 2007, the Mozilla Foundation published 39 security advisories for Firefox and released ten new minor versions *2.0.0.2* to *2.0.0.11* of the browser FF2. Eight of these ten new versions address security vulnerabilities. Opera released six new versions in 2007, *9.20* to *9.25*, all of which fix security issues. Our measurement revealed that the adoption rate of minor versions is independent of the security risk being fixed, both for Firefox and Opera. E.g. there is no difference in the patch adoption rate between high and low risk security updates. We conclude that the initial adoption rate is governed by the design and ergonomics of the auto-update functionality of the browser. Firefox can be updated with a single click if run with administrative rights. An update of Opera is essentially the same procedure as a complete manual download and install of the browser, typically requiring many user decisions and more than ten clicks.

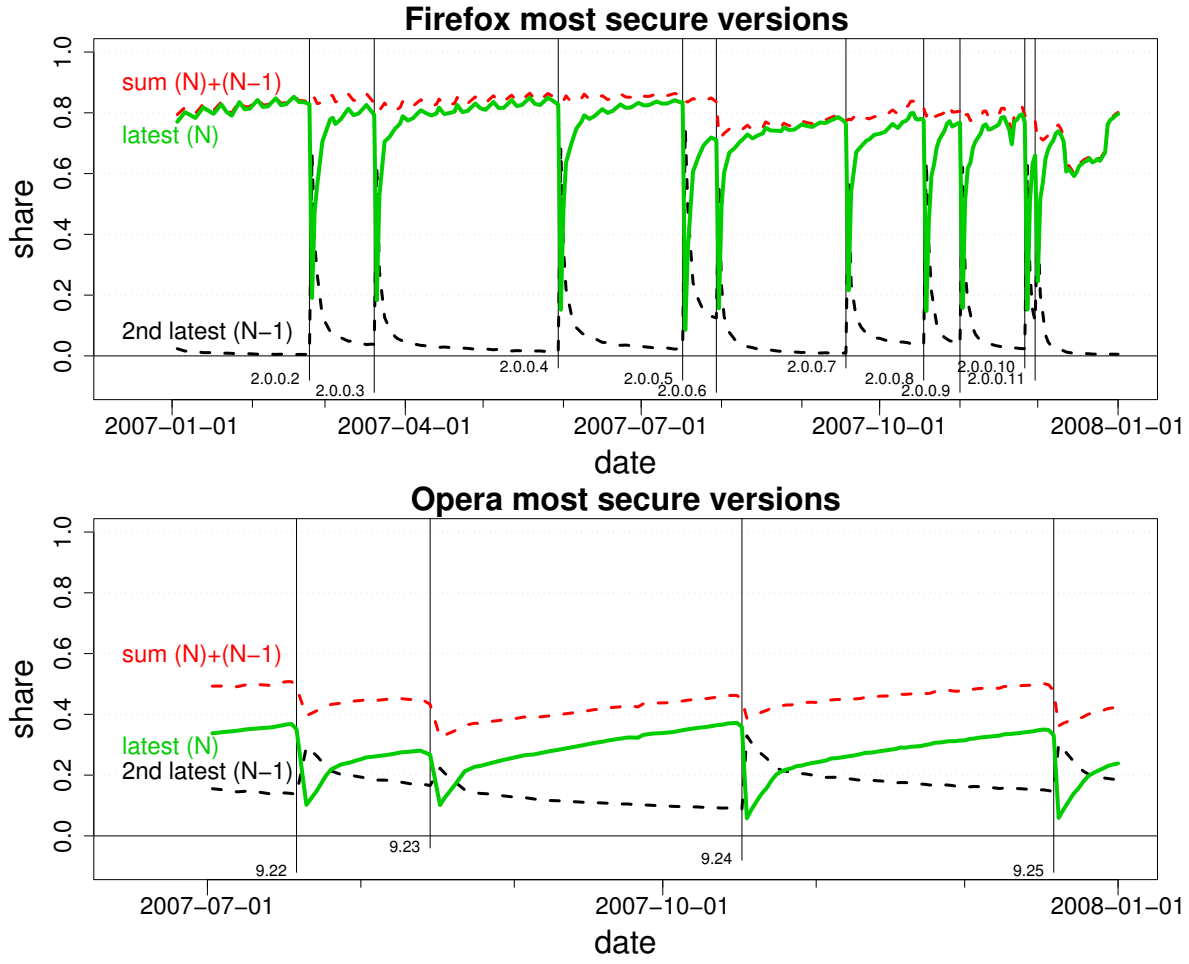

Figure 3: Evolution of the share of the most recent versions (N) of Firefox and Opera. The dotted line depicts the previous version ($N-1$) of the browser. 100% is the total of all versions of the respective browser share.

6. INSECURITY DYNAMICS

End users are exposed to security risks when surfing with an outdated version of their browser. To assess the extent and evolution of this risk exposure we measured the daily share of the most secure (= most recent) browser version (N) for Firefox and Opera throughout the year 2007. For Firefox we plot the whole year and for Opera a close-up of the second half of 2007 in Figure 3. These plots show the periods of increased risk of either browser in relation with the update cycle. We found that the *maximum share of the most secure web browser version* in active use never exceeded 80% for Firefox and 46% for Opera on any day in 2007. Even worse, after the release of Opera 9.20 on April 11th 2007, the most secure browser share for Opera remained below 35%. These numbers indicate an about four times larger vulnerable populatition than what Secunia found with their Personal Software Inspector (PSI) [17] as mentioned in Section 2. This difference can be attributed to the fact that our measurement does not need any cooperation of the user. Secunia's measurement relies on a software agent that needs to be installed first, which introduces a bias in the population measured.

Our analysis reveals the significant risk that end users are exposed to while surfing. Given the number and popularity of attacks against web browsers [14, 15] this is a considerable security threat. Without further layers of protection (security proxies, intrusion prevention systems, anti virus) these users are an easy target for any attacker. Suppressing or obfuscating the browsers version information in the user-agent header does not mitigate the threat: attackers are known to try exploiting vulnerabilities blindly, regardless of reported software version. Not included in our analysis are vulnerabilities due to insecure browser plug-ins which must be patched separately. Thus, the latest version of a web browser can still be vulnerable through insecure plug-ins. As a side note, we noticed a drop of the FF secure version share from mid to end of December 2007, where Firefox 2.0.0.2 significantly eats into the share of the latest version 2.0.0.11. A possible reason might be the bundling of the old Firefox version with a popular software package.

7. CONCLUSIONS

We presented an method to measure the patch level of a large browser population based on Web server logs only.

No cooperation of the user or any installation of monitoring software is needed for our measurements. Using anonymized logs from Google's worldwide search and application Web sites, our presented analysis is of truly global scale covering approximately 75% of the Internet users for more than a year. When observing Firefox update dynamics, we could differentiate two phases: a fast initial adoption of a new version, and then a slower continuing adoption limited by the major browser version migration. The Firefox update dynamics measurements revealed that despite the single click integrated auto-update functionality, rather surprisingly, one out of five Firefox users surfs the Web with an outdated browser version. This results in millions of vulnerable Web surfers that can easily be attacked by drive-by downloads. For Opera, more than half of all users stay with an outdated version. This is likely due to the higher complexity of the update mechanism of Opera, requiring extensive interaction and many decisions from the user. Firefox' auto-update mechanism is far superior in terms of speed and maximum new version adoption with 80% for Firefox compared to 46% for Opera. We highly recommend that software security updates for end-users are deployed in an automatic fashion with minimal user involvement in order to minimize the vulnerable software population. Given that browser plug-ins can also be vulnerable, our measurement provides a lower bound for the actual vulnerable population.

Finally, we observed that up to 5% of the Web surfers tend to use newer browser versions (e.g. IE7 vs. IE6) at home on the week-end compared to at work during the week. We call this the week-end effect. Our measurement of Web browsers' migration dynamics shows that automatic mechanisms to deliver upgrades (e.g. bundling a new major version with an operating system upgrade) outperform mechanisms requiring user interaction. We found that the severity of a vulnerability has no influence on the patch adoption speed and the end-of-life (EOL) of a Web browser version only has a small effect.

In large part users are not immediately reacting to recommendations of Web browser producers or when new security vulnerabilities are disclosed. A more user friendly update process and a reduced frequency of needed software updates has the potential to dramatically reduce the number of poorly patched systems worldwide.

8. ACKNOWLEDGMENTS

We would like to thank *Tom Dillon*, *Niels Provos* (both from Google), *Gunter Ollmann* (IBM Internet Security Systems), *Martin May*, and *Daniela Brauckhoff* (both ETH Zurich), and the anonymous reviewers for their valuable feedback for this paper.

9. REFERENCES

[1] Baumgartner, K. Storm 2007 - Malware 2.0 has arrived. `http://www.virusbtn.com/pdf/conference_slides/2007/BaumgartnerVB2007.pdf`.

[2] T. Berners-Lee, R. Fielding, and H. Nielsen. Hypertext transfer protocol – HTTP/1.0. RFC 1945, Internet Engineering Task Force, May 1996.

[3] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. In *Communications of the ACM, vol. 51, no. 1 (2008), pp. 107-113*. ACM, 2008.

[4] R. Fielding, J. Gettys, J. Mogul, H. Nielsen, and T. Berners-Lee. Hypertext transfer protocol – HTTP/1.1. RFC 2068, Internet Engineering Task Force, Jan. 1997.

[5] S. Frei, T. Dübendorfer, G. Ollmann, and M. May. Understanding the web browser threat. Technical Report 288, ETH Zurich, June 2008.

[6] Frei, S., Dübendorfer, T., Plattner B. Repository of high-resolution browser update dynamics plots. `http://www.techzoom.net/risk`.

[7] Janco. Browser and OS Market Share White Paper. `http://www.e-janco.com/Samples/BrowserSample.pdf`, Apr. 2008.

[8] Mozilla Foundation. User-Agent Definition. `http://www.mozilla.org/build/revised-user-agent-strings.html`.

[9] Net Applications. Browser Market Share. `http://marketshare.hitslink.com/report.aspx?qprid=3`, Jan. 2008.

[10] Net Applications. Search Engine Market Share. `marketshare.hitslink.com/report.aspx?qprid=4`, Apr. 2008.

[11] Ollmann, G. User Agent Attacks. `http://www.technicalinfo.net/blog/security/20080121_UserAgentAttacks.html`.

[12] Ollmann, G. X-Morphic Exploitation. `http://www.iss.net/documents/whitepapers/IBM_ISS_x-morphic_exploitation.pdf`.

[13] OneStat. Web Analytics. `http://www.onestat.com/html/aboutus_pressbox53-firefox-mozilla-browser-market-share.html`.

[14] N. Provos, P. Mavrommatis, M. A. Rajab, and F. Monrose. Google Technical Report: All Your iFRAMEs Point to Us. `http://research.google.com/archive/provos-2008a.pdf`, 2008.

[15] N. Provos, D. McNamee, P. Mavrommatis, K. Wang, and N. Modadugu. The Ghost In The Browser - Analysis of Web-based Malware. In *Proceedings of HotBots 2007, Usenix*, April 2007.

[16] Qualys Research Report. Laws of Vulnerabilities. `http://www.qualys.com/docs/Laws-Report.pdf`.

[17] Secunia. Personal Software Inspector (PSI). `http://secunia.com/blog/17/`.

[18] TheCounter.com. Web Analytics. `http://www.thecounter.com/stats`.