# Understanding the Web browser threat: Examination of vulnerable online Web browser populations and the "insecurity iceberg"

Stefan Frei[1], Thomas Duebendorfer[2], Gunter Ollmann[3], Martin May[1]
[1] Communication Systems Group, ETH Zurich, Switzerland
[2] Google Switzerland GmbH
[3] IBM Internet Security Systems, USA

Contact: insecurity-iceberg@ee.ethz.ch
http://www.techzoom.net/insecurity-iceberg

ETH Zurich Tech Report Nr. 288

## 1. INTRODUCTION

In recent years the Web browser has increasingly become targeted as an infection vector for vulnerable hosts. Classic service-centric vulnerability exploitation required attackers to scan for and remotely connect to vulnerable hosts (typically servers) in order to exploit them. Unlike these, Web browser vulnerabilities are commonly exploited when the user of the vulnerable host visits a malicious Web site.

Attacks against Web browsers depend upon malicious content being rendered by the appropriate built-in interpreter (e.g., HTML, JavaScript, CSS, etc.) or vulnerable plug-in technology (e.g., Flash, QuickTime, Java, etc.) [1, 2]. Vulnerabilities lying within these rendering technologies are then exposed to any exploit techniques or malicious code developed by the attacker. Vulnerability trend reports have indicated that remotely exploitable vulnerabilities have been increasing since the year 2000 and reached 89.4% of vulnerabilities reported in 2007 [3]. A growing percentage of these remotely exploitable vulnerabilities are associated with Web browsers.

Profit motivated cyber-criminals have rapidly adopted Web browser exploitation as a key vector for malware installation. Due to the methodology of exploiting Web browser vulnerabilities and the unpredictable browsing patterns of typical users, for widespread infection of vulnerable hosts the criminals must seed a mix of popular and high-traffic websites, or incentivize users through email spam, with URLs directing potential victims to Web servers hosting their malicious content. The former method is commonly known as drive-by download, where "drive-by" refers to the fact that Web browsers must initially navigate to a malicious page and "download" refers to the covertly downloaded and executed malware - typically trojans.

As popularity of this attack vector has blossomed, there have been frequent reports of hundreds of thousands of Web sites succumbing to mass-defacement [1, 4, 5, 6, 7, 8] - where the "defacement" often consists of an embedded iframe. These iframes typically inlcude content from servers hosting malicious JavaScript code designed to exploit vulnerabilities accessible through the user's Web browser and subsequently to initiate a drive-by malware download. These mass-defacements
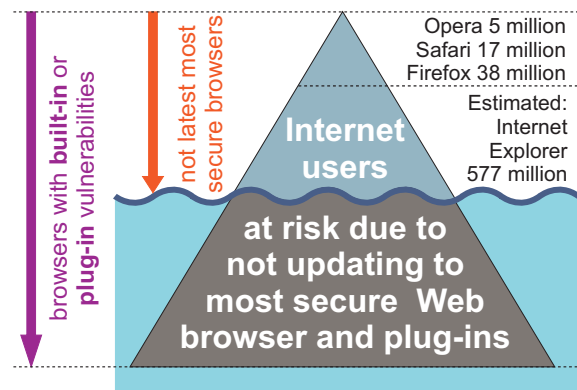


**Figure 1: The Web browser Insecurity Iceberg represents the number of Internet users at risk because they don't use the latest most secure Web browsers and plug-ins to surf the Web. This paper has quantified the visible portion of the Insecurity Iceberg (above the waterline) using passive evaluation techniques - which amounted to more than 600 million users at risk not running the latest most secure Web browser version in June 2008.**

cause once-benign sites to turn against their visitors. Even pages owned by institutions like the United Nations (un.org), the UK government (.gov.uk) and many others have succumbed to such attacks. In 2007, Google uncovered more than three million malicious Web addresses (URLs) that initiate drive-by downloads [9].

While several studies and reports have focused upon the scale of the mass-defacements and malicious content being served by compromised servers, none have provided quantitative analysis of the most critical component in drive-by download attacks - the number of users likely to become victims of the attack due to the use of insecure Web browser technologies.

The analysis presented in this paper is based on the large global user base of Google's Web search and application sites. By measuring the lower bounds of insecure Web browsers

used to daily surf the Internet, we provide new insights into the global vulnerable Web browser problem. To capture the extent of this security problem, we introduce the notion of the "Insecurity Iceberg" (see Figure 1) and estimate the number of users worldwide relying on a Web browser version different from the latest most secure version or vulnerable plug-ins, which could result in a host compromise.

Following this detailed analysis, we identify and discuss a number of current and future protection technologies that can help mitigate the escalating threat to vulnerable Web browsers.

## 2. DATA SOURCES AND METHODOLOGY

As with all studies of Internet threats and trends, the analysis and conclusions reached are dependent upon the breadth and scope of the data. While statistics concerning Web browser market shares can be found in many locations, we believe the data sources used for this detailed analysis of vulnerable Web browsers are unique in both scope, detail and quality.

The data used to measure the worldwide vulnerable Web browser population within each browser type was provided by Google, and is a subset of non-personally identifiable data accumulated by Google's search and Web application server logs from around the globe; processed daily between January 2007 and June 2008. With Google's search queries coming from more than 75% [10, 11] of Internet Web search users, our measurements of Web browser proliferation are of a truly global scale.

Our purpose of analysis was to establish the global scale of Web browser-based insecurity. A critical difference between our analysis and seemingly related "browser market share" studies [12, 13, 14, 15] is the use of **both major and minor version** information and the correlation with known security patch release dates for each Web browser type. Any discrepancies between browser version share numbers can likely be attributed to sample sizes and degree of global coverage. Key points in our data sampling and analysis methodology are as follows:

- With each page request a Web browser typically imparts information such as its *type*, *version*, and *operating system* in the HTTP USER-AGENT header field [16] which is recorded in most common Web server logs. Using this large data set, we calculated the major version usage share within Firefox, Opera, Safari, and Internet Explorer users for each day.

  - Unlike ubiquitous Web browser "market share" statistics, we were not interested in the number of page hits or visits. Instead, we measured the number of unique Web browser installations active on a given day.

- Where applicable, minor version information was obtained from the USER-AGENT data to help enumerate the specific patch-level of the Web browser.

  - This method of analysis allowed for unbiased measurements of detailed browser patch level at a global scale, **without requiring interaction with the user** and is the **first global scale measurement of popular browser patch dynamics**.

  - The USER-AGENT header fields for Firefox, Safari, and Opera contain both major and minor version information, whereas Internet Explorer only contains the major version. Therefore, it was not possible to enumerate the patch level of Microsoft Internet Explorer using this method beyond its major release numbers.

- Each Web browser was counted only once per host, per day. The default Google cookie system served as a uniqueness identifier for the first visit of each actively used browser exactly once per day - and helped reduce overcounting due to multiple visits.

- For reasons of conciseness, we have taken week-day statistics as the data-points for graphing and comparison purposes.

- We measure the dynamics of major and minor software version updates and compare results with the mechanisms available to carry out the updates. We correlate our results with measurements of Secunia's Personal Software Inspector [17] to estimate the global population of Internet users not using the most recent version of their browser. Our measurement does not include the additional risk exposure of unpatched browser plug-ins or 0-day exploits.

- We acknowledge that there are multiple opinions concerning business usage and application compatibility reasons for not upgrading to a current version of a Web browser technology (in particular applications that have embedded Internet Explorer objects), but do not believe that those opinions have a bearing on the methodology used to establish the insecurity of the global Web browser problem. While not using the latest version represents a risk, it may be reduced or mitigated through the use of complementary security measures.

At no time during this study did the authors of this paper have any access to personally identifiable information. The data sets analyzed used a unique cookie value (when available) to merely identify a unique browser visit, and the provided data could not be used to identify an individual or their browsing patterns either directly or through correlation.

Google takes privacy of its users very seriously and automatically expires cookies after a limited period of time. For personalized services like Gmail or Google calendar where state information is required, cookies are used to keep the log-in status, but are optional when utilizing Google's Web search.

## 3. THE INSECURITY ICEBERG OF INTERNET USERS AT RISK

### 3.1 Measurement of Browser Versions in Daily Use

In the face of a more hostile environment, most commercial vendors of Web browser technologies have made progress over recent years in making their products more resilient to common security threats - dropping insecure features and strengthening others. Their development life-cycles have matured and

typically encompass multiple levels of secure design and vulnerability testing, as well as new processes for promptly handling externally discovered flaws. As such, most updates and patches for existing Web browser technologies (both the core browsing engine and third-party plug-ins) increasingly incorporate new and vital security fixes - a trend that is expected to continue in to the future.

For years the software industry has promoted one security best practice over all others: **always use the most recent version of the installed software and instantly apply the latest patches**. With today's hostile Internet and drive-by download attack vectors, failure to apply patches promptly or missing them entirely is a recipe for disaster; exposing the host to infection and possibly subsequent data disclosure or loss.

Analysis of Web browser USER-AGENT information from the Google data set, combined with a catalogue of known vulnerabilities and subsequent security patches associated with a particular update, enabled us to estimate the lower bound of the number of Web browsers in use repeatedly failing to apply patches, many of which fixed built-in browser vulnerabilities.

In mid June 2008, the most commonly encountered browser technologies used to navigate the Internet were Microsoft's Internet Explorer (IE) 78%, Mozilla's Firefox (FF) 16%, Apple's Safari (SF) 3%, and Opera (OP) 1% according to TheCounter.com [15]. The combined usage share of these four browsers was 98.6%, dominated by Internet Explorer and Firefox as can be seen in Table 1.

| Browser Type | IE | FF | SF | OP | Total |
|---|---|---|---|---|---|
| Share of browsers in daily use in percent | 78.3 % | 16.1% | 3.4% | 0.8% | 98.6% |
| Browsers in daily use in million (on the Internet worldwide) | 1103 | 227 | 48 | 11 | 1389 |

**Table 1: Percentage of Web browsers by type according to TheCounter.com averaged over Feb 1st to June 18th, 2008. The absolute worldwide user counts were derived from the global Internet user count of 1,408 billion users.**

## 3.2 Most secure browser

In this section, the **most secure browser** designates the latest official public release of a vendor's Web browser at a given date. Beta versions are not considered an official public release.

We used the most recent major versions of Internet Explorer 7 (IE7), Firefox 2 (FF2), Safari 3 (SF3) and Opera 9 (OP9) as the benchmark version for our most secure Web browser measurements. Microsoft's Internet Explorer version 6, independent of its patch level, is not considered the most secure version of Internet Explorer by Windows expert Brian Livingston [18] and even Microsoft calls IE7 *"an extremely important update from a security perspective"* over IE6 [19] and states *"There are dangers that simply didn't exist back in 2001, when Internet Explorer 6 was released to the world. Internet Explorer 7 makes surfing the web fundamentally safer by offering greater protection against viruses, spyware, and other online risks."*.

Table 2 shows the usage share of the latest major browser

version within each type of Web browser (e.g. the share of IE7 within the IE population). There were 1,408 million Internet users worldwide end of March 2008 [20]. Globally only 59.1% (832 million users), make use of the latest major version of their preferred Web browser to navigate the Internet. This is an estimate for the upper bound for the global share of the most secure browsers in use. However, 576 million users surfed the Internet without using the latest major browser version of their preferred browser.

| Latest Major Version | IE7 | FF2 | SF3 | OP9 | Total |
|---|---|---|---|---|---|
| Release date of latest major version | 2006-10-18 | 2006-10-24 | 2007-10-26 | 2006-06-20 | |
| Share of latest major version within browser type | 52.5% | 92.2% | 70.2% | 90.1% | 59.1% |
| Number of latest major version in million (worldwide) | 579 | 209 | 34 | 10 | **832** |

**Table 2: Share of the latest major version within a given type of browser as seen on Google's search and application Web sites in first week of June 2008. Absolute counts were calculated using Table 1.**

Analysis of the distribution of patches within the latest major version is used to measure the share of the most secure version for each browser type.

For Firefox, Safari and Opera we used the HTTP user-agent information in the Google's Web log data sets to determine the minor version. For Internet Explorer we relied upon the results of Secunia's PSI statistics [21] to estimate the share of the most secure version.

As shown in Figure 3, we discovered that at most 83.3% of Firefox users, 65.3% of Safari users, 56.1% of Opera users, and 47.6% of Internet Explorer users were using the latest most secure browser version on any day between January 2007 to June 2008. For the latest version analysis of Safari, we only considered the date range Dec 2007 to June 2008, when Safari version 3 became widespread.

Despite the single-click integrated auto-update functionality of Firefox, rather surprisingly, 16.7% Firefox users (one out of six) continue to surf the Web with an outdated version of the Web browser. Meanwhile, 43.9% of all Opera users surf the Web with an outdated browser version. In the case of Internet Explorer, 52.4% of that user population continues to rely upon superseded versions of the Web browser.

While Table 2 represents the early June 2008 snapshot of the usage share of the latest major version within each browser type, Figure 2 depicts how these usage shares have changed over time as users migrate to the latest major version of their favorite Web browser between January 2007 to June 2008.

It is noteworthy that it has taken 19 months since the initial general availability of IE7 (public release October 2006) to reach 52.5% proliferation amongst users that navigate the Internet with Microsoft's Web browser. Meanwhile, 92.2% of Firefox users have migrated to FF2. The migration between major versions was found to be generally a slow process, ex-
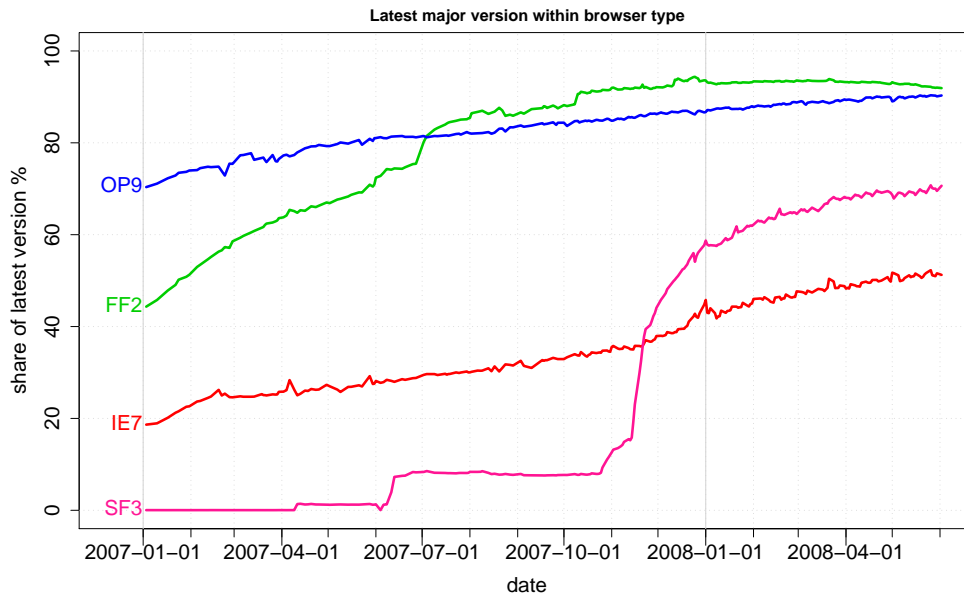
**Figure 2: Upgrade dynamics of major versions of Internet Explorer (IE7), Firefox (FF2), Opera (OP9), and Safari (SF3) from Jan 2007 to Jun 2008. The plot shows the usage share of the latest major version within each browser type.**
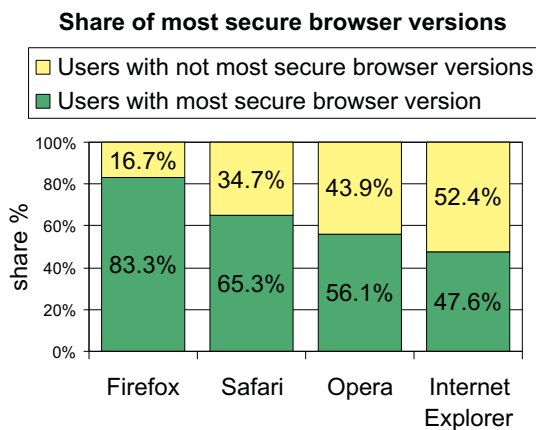


**Figure 3: Maximum share of users surfing the Web with the most secure versions of Firefox, Safari, Opera and Internet Explorer in June 2008 as seen on Google websites.**

cept for Apple's Safari SF3 which surpassed 60% share within 3 months of its release - likely influenced by Apple's controversial inclusion of the new Web browser in the auto-updates of other popular Apple software products [22].

## 3.3 Browser Insecurity Iceberg

Just as a floating iceberg only exhibits part of its mass above the waterline, we believe that our measurement of potentially insecure Web browsers based upon major and minor version information is smaller than the global number of users at risk. Insecure Web browsers (i.e., they have "built-in" vulnerabilities and security weaknesses) are of course a critical security problem, but vulnerable plug-ins that are accessible (and ex-

ploitable) through the Web browser extend the insecurity iceberg and form the part hidden below the water surface.

### 3.3.1 Browsers with built-in vulnerabilities

We have estimated the global number of users with browsers having "built-in" vulnerabilities based upon our measurements and Secunia's PSI study.

Secunia [21] identified (for the month of May 2008) that 4.4% of IE7, 8.1% of Firefox, 14.3% of Safari (Windows only), and 15.2% of Opera users have not applied the most recent security patches available to them from the software vendor. In comparison, we discovered that 16.7% of Firefox, 34.7% of Safari (all OS), and 43.9% of Opera Web browser installations (using our Web server log-based measurements) had not applied the most recent security patches. We found that our Firefox, Safari, and Opera results were higher than those of Secunia's, differing by a factor of 2.1 (Firefox), 2.4 (Safari), and 2.9 (Opera), and attribute this difference to a probable bias for more security aware users to take advantage of Secunia's security scanner PSI than the average global community.

To derive the global population of users with browsers vulnerable to built-in vulnerabilities we used the results of our measurements for Firefox, Safari, and Opera. We chose to estimate the value for IE7 based upon the findings of Secunia as shown in Table 3:

- Estimate A
  Firefox, Safari, and Opera shares are from our Google Web log measurements. The IE7 share of 4.4% is from Secunia's measurement [21]. This is a minimum estimate as Secunia's measurement is likely biased towards more security aware users. IE6 is not considered a most secure Web browser version (independent of patch level) as per Microsoft's upgrade recommendation in [19].

- Estimate B

| Browser Type | IE w/o IE7 | IE7 | FF | SF | OP | Total |
|---|---|---|---|---|---|---|
| Share of browsers in daily use in percent (cf. Table 1) | 37.2% | 41.1% | 16.1% | 3.4% | 0.8% | 98.6% |
| Browsers in daily use worldwide in million | 523.8 | 578.7 | 226.7 | 47.9 | 11.3 | 1388.3 |
| **Estimate A** | | | | | | |
| Share of not most secure browser versions in percent | 100.0% | 4.4% | 16.7% | 34.7% | 43.9% | 43.3% |
| Not most secure browser versions in million | 524 | 25 | 38 | 17 | 5 | 609 |
| **Estimate B - correcting the bias of PSI (IE7 x 2.1)** | | | | | | |
| Share of not most secure browser versions in percent | 100.0% | 9.2% | 16.7% | 34.7% | 43.9% | 45.2% |
| Not most secure browser versions in million | 524 | 53 | 38 | 17 | 5 | **637** |

**Table 3: Estimation of the number of users not using the most secure version of their browser.**

We apply the factor 2.1 to the IE7 share (2.1 x 4.4% = 9.2%) to correct for the bias of Secunia's measurement within a security aware user population. The factor was found when comparing Firefox, Safari, and Opera data from Google log files with Secunia's data.

Our estimate B shows that at least 45.2%, or 637 million users, were not using the most secure Web browser version on any working day from January 2007 to June 2008. These browsers are an easy target for drive-by download attacks as they are potentially vulnerable to known exploits. This represents the tip of the Browser Insecurity Iceberg in Figure 1.

**Browsers with plug-in vulnerabilities**

Because our data sources were limited to the data logged by Google's Web servers and the USER-AGENT fields for major and minor Web browser version information, and as plug-in version information is not typically stored in this data field, we were not able to directly measure the number of users having out of date and vulnerable Web browser plug-ins. However, there is public evidence that this number adds to the number of users with browsers having "built-in" vulnerabilities:

- **Fully patched browsers at risk** Through vulnerable plug-ins even those hosts are at risk, which are running the latest most secure browser version.

- **Cross-browser and cross-platform plug-ins** Most plug-ins are compatible with multiple popular Web browser technologies and operating systems. Therefore a larger population of users are exposed if a common plug-in is found to be vulnerable.

- **Multiple popular plug-ins per browser** A typical Web browser has more than one plug-in application installed. Media players and other plug-ins are ubiquitous, with individual usage shares frequently exceeding 80% [2]. Table 3 lists the adopted use of some of the most popular plug-in applications - all of which are accessible through a Web browser.

- **Plug-in patching discipline** Considering our analysis of insecure Web browser usage, we deem it unlikely that the same users achieve higher patch levels for multiple plug-ins installed; with each plug-in relying on different patching and updating mechanisms. For example, Secunia's numbers that state 18.7% of all WinAMP 5 installations miss important security updates, and 21.7% of all Quicktime 7 installations are out of date [17].

Users with browsers having plug-in vulnerabilities and those found to have browsers with built-in vulnerabilities are not mutually exclusive. While we can estimate the tip of the Browser Insecurity Iceberg based on global measurements, additional users at risk are hidden below the water line as shown in Figure 1.

| Plug-In | Vendor | Share | Support |
|---|---|---|---|
| Flash Player | Adobe | 98.8% | all |
| Java | Sun | 84.0% | all |
| Media Player | Microsoft | 82.2% | IE only |
| QuickTime Player | Apple | 66.8% | all |
| Shockwave Player | Adobe | 55.6% | all |
| RealOne Player | Real Networks | 47.1% | all |
| Acrobat PDF Reader | Adobe | >80% | all |

**Table 4: Usage shares of some widely used plug-ins.**

# 4. TOWARDS A SAFER BROWSING EXPERIENCE

## 4.1 Existing Technology Solutions

There exist today a number of technologies that have already been proven to offer some degree of protection against threats that target insecure and vulnerable Web browsers. While none of these technologies are currently capable of providing full protection against the threats, wherever possible, we recommend that enterprises and vendors deploy or implement them as part of their defence in depth strategy to help reduce the surface area of potential attacks.

### 4.1.1 Auto-update

Although Web browser users wish perfect software that will never have any exploitable software vulnerabilities, the nearest they can realistically hope for is that any vulnerabilities are promptly fixed by the software vendors and instantly applied to their browser. Critical to this instantaneous patching process is the mechanism of "auto-update". Our measurement confirmed that Web browsers which implement an internal auto-update patching mechanism do much better in terms of faster update adoption rates than those without.

Our comparison of the update dynamics between Firefox and Opera identified that auto-update mechanisms are crucial for timely patching. Firefox's auto-update was found to be way more effective than Opera's manual update download reminder strategy.

In our measurement period from January 2007 to June 2008, most users updated to a new version of Firefox within three days of a new public release, resulting in up to 83% of users
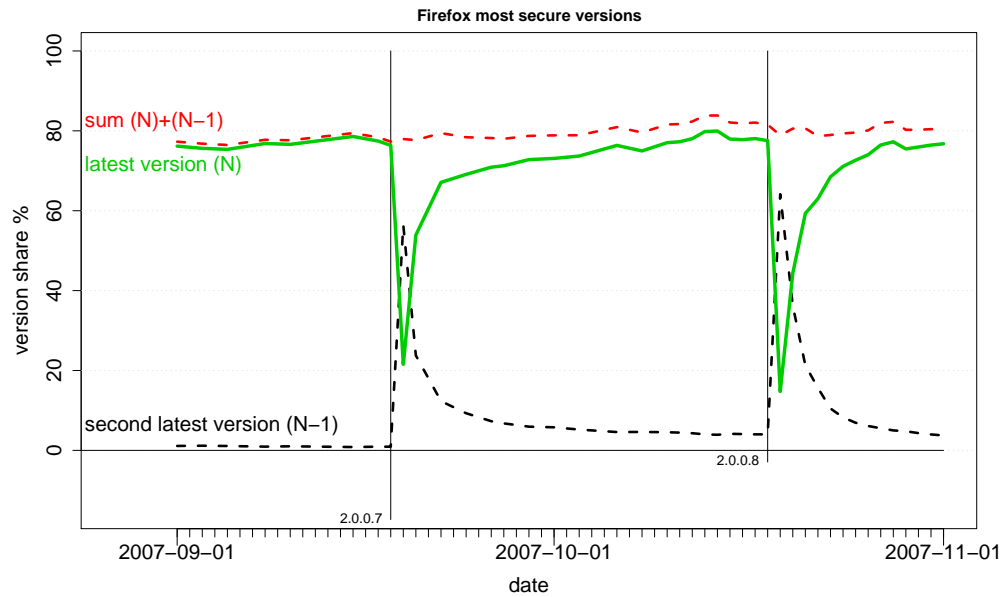
**Figure 4: Share of Firefox browser users in percent that use the latest or second latest browser version. Most users update within three days, which proves Firefox' auto-update mechanism to be very effective.**

having the most current and secure Firefox version installed. It took users of the Opera Web browser an average of 11 days before reaching an update saturation at a level of up to 56% of the users running the most current and secure Opera version. While Firefox and Opera check for updates when the browser is used, Safari relies on an external Apple-updater that appears to only poll for new updates at scheduled regular intervals while Internet Explorer gets updated as part of the monthly distributed Windows patches.

Regarding speed for upgrading to the next major browser version, Firefox, Safari and Opera users clearly outperformed Internet Explorer users (see Figure 2). Considering that Microsoft offers Internet Explorer 7 as an auto-upgrade from Internet Explorer 6 as part of the monthly Windows updates and that it requires a manual patch to prevent upgrading to version 7, it is rather surprising to see how slow the migration to the most secure version has been.

We believe the auto-update mechanism as implemented within Firefox to be the most efficient patching mechanism of the Web browsers studied. Firefox's mechanism regularly polls an online authority to verify whether a new version of the Web browser is available and typically prompts the user to update if a new version exists. With a single click (assuming that the user has administrative rights on the host), the update is downloaded and installed. Just as importantly, Firefox also checks for many of the currently installed Firefox plug-ins if they are similarly up to date, and, if not, will prompt the user to update them. Opera's update mechanism is essentially the same procedure as a manual download and re-installation of the browser.

While Microsoft's operating system auto-update functionality encompasses the Internet Explorer update mechanism even if the browser is not in use, the fact that patch updates (for both Internet Explorer 6 and 7) are typically only made available on a monthly basis means that updates are released less

frequently (when compared to Firefox), which can result in a lower short term patching effectiveness.

Based upon our findings, we strongly recommend that software vendors embrace auto-update mechanisms within their products that are capable of identifying the availability of new patches and installing security updates as quickly and efficiently as possible - ideally enabled by default and causing minimal disruption to the user. We also recommend that these same auto-update mechanisms are capable of alerting the user of any plug-ins currently exposed through the Web browser that have newer and more secure versions available.

### 4.1.2 Perimeter URL Filtering

In light of mass-defacements and the organised "seeding" of Web sites for the purpose of drive-by download attacks, current URL filtering technologies can help to mitigate a fraction of the threat. When URL filtering technologies are deployed at the perimeter of a businesses network, and all corporate users proxy their Internet page requests through it, vulnerable hosts can usually be protected from Web sites known to be hosting malicious content designed to compromise vulnerable Web browsers.

If a Web site or particular URL is known to be malicious, it is a trivial process to prevent Web browser users navigating to the site and accessing the malicious content. However, a limitation of this protection is the extent of the URL database. If a malicious URL is not listed within the filtering database, no filtering protection is typically applied.

Major vendors that offer URL filtering solutions tend to have extensive coverage of well known and previously identified malicious Web sites, and are quick to incorporate new URL filters once additional malicious Web sites are reported. As such, URL filtering technologies have proven to perform well against most current-generation mass-defacement iframe infestations, largely due to the fact that the iframes injected into

vulnerable Web sites during the mass-defacements tend to point to only a limited set of URL's that host the malicious infection code. However, there is an expectation that the criminals behind these attacks will soon adopt new techniques designed to bypass less agile URL filtering technologies.

We believe that URL filtering technologies are a valuable protection for reducing vulnerable Web browser exploitation. Given the success of this class of protection technology in helping to mitigate the drive-by download threat, we encourage its use beyond just protecting corporate environments. Some Internet Service Providers (ISPs) already offer URL filtering services to their customers and several popular search engine providers have also begun to issue visible alerts to users for URLs known to be malicious or fraudulent. We encourage vendors to collaborate and share information on newly identified malicious URLs (in the same vein as the malware research community already shares malware and analysis results) so that the most current and exhaustive filtering lists are available for the protection of all.

## 4.2    Proposed Technology Solutions

While the previously discussed protection technologies provide a level of defense against current Web browser exploitation threats and will likely improve as they mature, we believe that two **new strategies** could be developed in the near future to increase both host protection and user awareness.

### 4.2.1    "Best before" date

As quantified earlier, a significant fraction of the Web browsers used to navigate the Internet on a daily basis have been identified as being not up-to-date in terms of having the latest security patches applied. As such, they put the users that rely upon them at risk and infections by malware from the Web can expose personal data stored on their hosts to attackers. We believe that, in the majority of cases, the absence of critical or important updates to the Web browsers can be attributed to three important factors; technological (**can't do**), motivational (**don't care**), or informational (**don't know**).

When designing a solution for the vulnerable Web browser problem, it is important to consider the three parties involved: the end user, the Web browser vendor, and the Web service provider. Each of these may intervene and help remedy the problem in different ways.

We believe that a critical path to increasing the security of Web browsers (indeed any and all inter-networked applications including online game clients) involves making the user aware of the risk they are exposing themselves and their host to, but without introducing additional complexity.

Almost all users are familiar with the concept of "sell by", "expires on", or "best before" date stamps on perishable goods. Consumers tend to rely on this date information in order to decide whether to purchase the goods, when to use the goods and when to dispose of the goods. Once a particular perishable good has exceeded its "best before" date, the consumer is forced to evaluate their personal risk to using it or disposing of it. The greater the lapse between the "best before" date and the current date, the more risk the consumer assumes by not disposing of it.

Given the state of the software industry and the growing threat of exploitable vulnerabilities within all applications (not just Web browsers), we believe that the establishment of a "best before" date for all new software releases could prove

an invaluable means to educating the user to patch or "refresh" their software applications. The same "best before" date information could also be leveraged by Internet businesses to help evaluate or mitigate the risk of customers who are using out of date software and are consequently at a higher risk of having been compromised.

In Table 5, we compare the mechanisms of the software industry with the practices of the food industry.

In general, the food industries implementation of a "best before" date has been accepted as a valuable contribution in enabling consumers to evaluate the integrity of their purchase and the likelihood of spoilage. As illustrated above, the ecosystem developed by the food industry provides a good comparison with inadequacies commonly identified with the software industry. By developing a "best before" system for software applications, we believe that both users and businesses could be better informed - particularly in the realm of Web browsers and plug-in applications.

A public mindset change is required to counter evolving Internet threats, and a "best before" dating system would make visible the risks of using out-dated and insecure software. Instead of assuming software to be secure, a "best before" dating system would enable the notification of upcoming expiration and risk associated with *out of date* or *unpatched* software so that the user is aware of the need to keep installed software "fresh". While the "best before" date for software is not actually known at the time of a software release, it will be defined upon detection and availability of a security patch for an already released version. Therefore, it has to be queried frequently by software in use.

In order to achieve a viable "best before" dating system, software vendors need to follow stricter practices in the allocation of version number information and make those version numbers more accessible. For example Firefox, Safari, and Opera send detailed version information in the USER-AGENT header field, while Internet Explorer only provides major version information (excluding patch information). This detailed level of information enables service providers to remotely establish the patching level of the Web browser and perhaps implement their own "best before" look-up services. For example, an online banking service may use the version information supplied by the user's Web browser to establish when the software was last updated and to assess the level of risk the host has been compromised with malware. Armed with that information, the banking application may decide to implement additional safeguards and inspection on subsequent transactions by the user.

While some may argue that more detailed version information within the Web browser USER-AGENT field makes it easier for an attacker to target specific versions of a Web browser, this is irrelevant given current attack methodologies that simply iterate through ten's or hundred's of exploits hoping that one will work. Access to such version information by the attacker would not increase the probability of exploitation, but merely reduce the volume of data sent to the browser by the attacker's malicious server.

**Visualizing a "best before" date**

We believe that the "best before" dating concept could be built into most existing software applications, and thereby provide a convenient and persistent validation of the likely integrity of the software. For example, popular Web browsers

| | Food Industry | Software Industry |
|---|---|---|
| User mindset (today) | Food is a perishable good | Some software is used for years without update (to prevent compatibility issues) |
| Information and awareness | The "best before" date is easily accessible and clearly visible on all products.<br><br>Well accepted and enforced standard/practice. | If at all, the patch level of the browser is only visible upon request.<br><br>If at all, the state of auto-update (enabled/disabled/last checked) is only visible upon request.<br><br>No standard or requirement for software vendors to make security information available in their product to the end user (e.g., *display patch level or auto-update disabled*). |
| User | The consumer checks the expiration date before using food<br><br>A consumer is free to eat food after the "best before" date expired. However, he does so at his own risk and he is aware of it. | Software version information is available, but not all software packages are capable of automatically checking for updates.<br><br>No restriction for a user to work with insecure, old and unsupported software. Users are typically not aware of the risk. |
| Service provider (reseller or web service provider) | Retailers are typically not legally allowed to sell products after the "best before" date.<br><br>Vendor is liable for damages. | Web service providers do not check for insecure versions of Web browsers or browser plug-ins.<br><br>No liability for software vendors. |
| Manufacturer or producer | Producer may be liable if food sold before expiration turns out to be bad. | No liability for software vendors.<br><br>A vendor is not obliged to provide software updates |

**Table 5: Comparison of the users' mindset with regard to the food industry and the software industry**
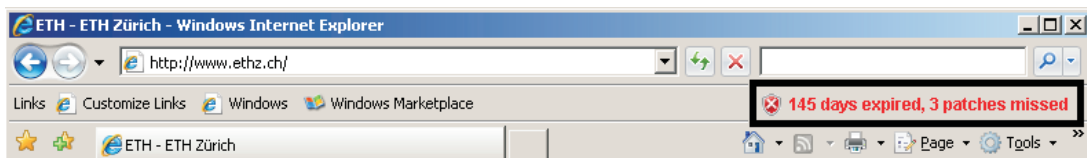


**Figure 5: Example "best before" implementation on Web browser**

could display a visual warning of expiry and how many patches are currently missing as illustrated in Figure 5.

Armed with more concise USER-AGENT version information, popular websites could also visually alert users (see Figure 6) to the fact that their Web browser is operating beyond its "best before" date and any missing updates (including providing shortcuts to the location of appropriate updates).

### 4.2.2 Authentic sources of most recent plug-in versions

As previously discussed, auto-update mechanisms can be a valuable device for keeping Web browsers up to date with the latest security patches and fixes. However, auto-update mechanisms typically do not encompass plug-ins not produced by the vendor or plug-ins not provided as part of a default in-

stallation. The Firefox Web browser auto-update mechanism is alerting users to compatible updates for plug-ins installed through, and registered by, the browser - but typically only encompasses a handful of plug-in applications commonly accessible through Web browser technologies. We believe that there are two key reasons for this - no convenient method of checking and cataloguing of installed plug-ins on the host, and no authoritative source of plug-in current version information (assisted by an update location).

Any examination of the last few years of vulnerability disclosures will reveal a plethora of critical, remotely exploitable, vulnerabilities in practically all plug-in technologies (e.g., Microsoft ActiveX, Adobe Flash, Apple QuickTime, etc.). These browser plug-ins must be similarly patched and updated, just like the Web browser itself. In order to achieve this, it is vi-
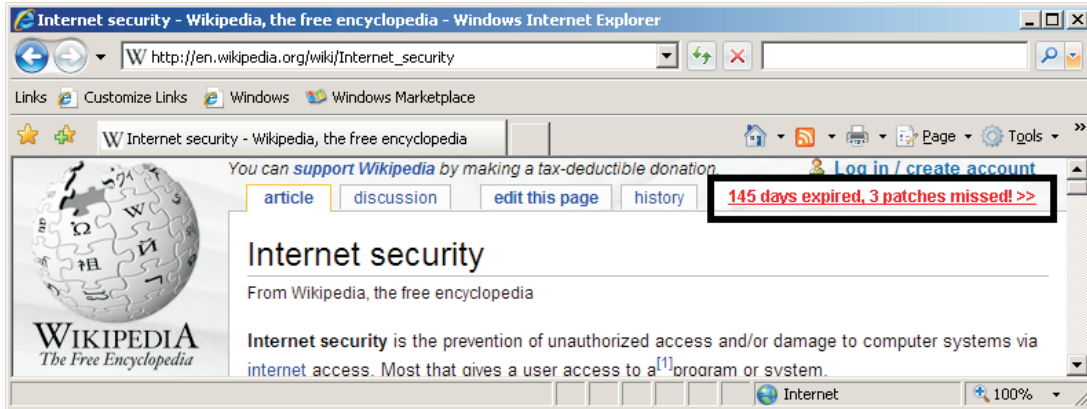
**Figure 6: Example "best before" implementation on a popular Web service, e.g., Wikipedia**

tal that the version information for an installed plug-in can be quickly compared with an authoritative source to discover whether the host has the most current and secure version.

However, it is inefficient for the different engineering teams of Web browsers and plug-ins to each develop independent solutions for the same problem. We propose that information of the most recent secure version of browsers and popular plug-ins should be systematically collected by trusted organizations and made accessible using a standardized querying process that also ensures a degree of confidence and authenticity in the version information. Using this process, Web browsers could easily check the status of the installed plug-ins and inform the user accordingly, as well as provide the information necessary to download and install the latest update. The same protocol could also be used by Web service operators to check the "best before" date information of client browsers and plug-ins requesting their services, as suggested in Section 4.2.1.

Conveniently, services and processes to collect and disseminate relevant security information to the public do already exist: CERTs and several private security information providers [23, 24, 25, 26, 27] already process vulnerability and patch information of various vendors and software packages. Providing secure version information on a standardized protocol would be in-line with their existing activities.

For plug-in technologies not registered or recorded by a searchable central authority (such as plug-ins no longer supported by the vendor that created them), the use of a "best before" dating system, reinforced by Web browser actions such as disabling all access and calls to the component, would likely aid in protecting vulnerable hosts.

## 5. CONCLUSION

Access to Google's global Web server logs enabled the authors to provide the first in-depth global perspective on the state of insecurity for Web browser technologies. Understanding the nature of the threats against Web browser and their plug-in technologies is important for continued Internet usage. As more users and organizations depend upon these browser technologies to access ever more complex and distributed business applications, any threats to the underlying platform equate to a direct risk to business continuity and integrity.

By measuring the patching processes of Web browser user populations, we have been able to identify the potential global scale of Web-based malicious exploitation of browser technologies and prove how existing mechanisms such as Firefox's auto-update can outperform more complex and less timely solutions.

Based on direct measurements of the adoption of new Web browser updates based upon available USER-AGENT major and minor browser software version numbers, and by combining that data with Secunia's latest PSI local-host scanning results for plug-in patch adoption (even though sample sizes are radically different), we quantified the lower bounds of the Web browser population vulnerable to attacks through security weaknesses.

Unfortunately, just like a floating iceberg, we were only able to measure and accurately estimate the tip above the water. The tip of the Web browser insecurity iceberg was measured to be 637 million (or 45.2%) Internet users at risk worldwide due to not running the latest most secure browser version. Meanwhile, hidden below the surface, the iceberg extends further encompassing users that rely on outdated vulnerable browser plug-ins. Due to an inability to passively enumerate the versions of any plug-ins hosts have installed (due to this information not typically being imparted in HTTP requests logged by Web servers), this was out of scope for our passive measurement based study.

To help combat existing and rapidly evolving threats such as malicious drive-by downloads, we have proposed the concept of a "best before" date for software and related mechanisms to tackle user awareness and provide a vehicle for online businesses to better assess the risk level of their customers' hosts.

While none of the mechanisms proposed within this paper can guarantee to fully protect against exploitation, we are confident that widespread adoption and improvements of these technologies would dramatically reduce the dimensions of the *insecurity iceberg* and shrink the attack surface.

## 6. REFERENCES

[1] Dan Goodin, TheRegister, "Attack code in the wild targets new (sort of) Adobe Flash vuln," http://www.theregister.co.uk/2008/05/27/new_adobe_flash_vuln, May 2008.

[2] Adobe Inc., "Browser plug-in Market Shares," http://www.adobe.com/products/player_census/flashplayer/tech_breakdown.html.

[3] IBM Internet Security Systems - X-Force, "IBM Internet Security Systems X-Force 2007 Trend Statistics," http://www.ibm.com/services/us/iss/pdf/etr_xforce-2007-annual-report.pdf, December 2007.

[4] Heise security, "Hundreds of thousands of web pages infected with malicious JavaScript," http://www.heise-online.co.uk/news/, April 2008.

[5] Websense, "Mass Attack JavaScript injection - UN and UK Government websites compromised," http://securitylabs.websense.com/content/Alerts/3070.aspx, April 2008.

[6] Dan Goodin, TheRegister, "Trend Micro gets slashed in attack of the killer iframes," http://www.theregister.co.uk/2008/03/13/trend_micro_website_infected, March 2008.

[7] F-Secure Inc., "Mass SQL Injection," http://www.f-secure.com/weblog/archives/00001427.html, April 2008.

[8] SANS, "Mass File Injection Attack," http://isc.incidents.org/diary.html?storyid=4405, May 2008.

[9] 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.

[10] NetApplications.com, "Search Engine Market Share," http://marketshare.hitslink.com/articles.aspx, March 2008.

[11] NetApplications.com, "Search Engine Worldwide Market Share," http://marketshare.hitslink.com/report.aspx?qprid=4, April 2008.

[12] Janco Associates Inc., "Browser and OS Market Share White Paper," http://www.itproductivity.org/browser.php, June 2008.

[13] NetApplications.com, "Browser Market Share," http://marketshare.hitslink.com/report.aspx?qprid=3.

[14] OneStat.com, "OneStat WebAnalytics," http://www.onestat.com/html/aboutus_pressbox53-firefox-mozilla-browser-market-share.html.

[15] Jupitermedia Corporation, "TheCounter.com. Web Analytics," http://www.thecounter.com/stats/2008/June/browser.php.

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

[17] Secunia, "Secunia PSI study: 28% of all detected applications are insecure; current last month statistics," http://secunia.com/blog/11/.

[18] B. Livingston, "Is IE 7 Really More Secure Than IE 6?" http://itmanagement.earthweb.com/columns/executive_tech/article.php/3639566, October 2006.

[19] S. Hardmeier, "Microsoft Better Browsing - Internet Explorer 7 offers improved security and productivity," http://windowshelp.microsoft.com/Windows/en-us/help/a426bb85-708c-4b75-87e2-874f9be3b4aa1033.mspx, October 2006.

[20] Internet World Statistics, "World Internet Users and Population Statsitics," http://www.internetworldstats.com/stats.htm, March 2008.

[21] Secunia Inc., "Secunia Software Inspector Statistics," http://secunia.com/software_inspector_statistics, May 2008.

[22] CNET, "Mozilla CEO says Apple's Safari auto-update 'wrong'," http://news.cnet.com/8301-10784_3-9901006-7.html, May 2008.

[23] US-CERT, "United States Computer Emergency Readiness Team (US-CERT)," http://www.us-cert.gov.

[24] Secunia Inc., "Secunia," http://www.secunia.com.

[25] IBM Internet Security Systems - X-Force, "IBM ISS X-Force," http://xforce.iss.net.

[26] SecurityFocus/Symantec, "SecurityFocus," http://www.securityfocus.com.

[27] FrSIRT, "French Security Incident Response Team (FrSIRT)," http://www.frsirt.com.