InfoWorld DeepDive

Web browser security

Staying secure on the Internet

Deep Dive

An expert guide to Web browser security

From basic tips to secure connection details to the security features of the six most popular browsers, this guide walks you through everything you need to know.

BY ROGER GRIMES

As an employer or employee, you no longer have a choice about whether or not to use the Web. **That doesn't mean that you don't have a choice about how you use it.**

Enterprise data is moving inexorably to the cloud and enterprise applications are fast abandoning desktop clients in favor of Web browser-based front ends. Sure, browsers still run on top of operating systems like Windows, OS X, or Linux. But, looked at from the perspective of workers, Web browsers pretty much are the new operating system.

With so much online activity, it's no surprise that Web-based attacks are among the most common threats out there. A 2015 report from the European Union Agency for Network and Information Security (ENISA), based on private and public sector data, ranked Web-based attacks such as malicious URLs, browser exploits, and compromised websites as the second biggest threat facing Internet users, right after malicious code. Web application attacks such as SQL injection and XSS (cross-site scripting) were the third most serious threat, the ENISA report concluded.

As an employer or employee, you no longer have a choice about whether or not to use the Web. That doesn't mean that you don't have a choice about how you use it. Making Web browsing sessions as secure as possible is one sure way to limit your exposure to online attacks. Your choice of a Web browser is the single most important factor in making those sessions secure.

Which browser should you use? And why? Let this Deep Dive be your guide.

A WORD ABOUT BROWSERS

Web browsers are complex pieces of software that interpret and interact with even more complex application code, much of it unfriendly. Today almost all Internet users rely on one of a small cadre of Web browsers, whether they're surfing via a laptop, tablet, or mobile phone. Microsoft's Internet Explorer still accounts for most activity, but Google's Chrome browser, the Mozilla Foundation's Firefox, and Apple's Safari have all seen their share of the pie grow in recent years. As of May 2015, Chrome use topped 25 percent for the first time—growth that came at the expense of both IE and Firefox. Recently, Microsoft added a fifth browser to the mix: Microsoft Edge, but adoption has been slow enough that it doesn't yet register.

Although we like to think that adoption skews towards the safer and more secure platform, Google lured users to Chrome by offering a better, faster, lighter-weight browser that also happened to be more secure. The truth is no browser is "secure." Like any complex application, Web browsers contain vulnerabilities in their underlying code—some of them exploitable. This bare fact gets revealed again and again at security conferences where researchers present novel attacks that can compromise browsers. One annual contest, Pwn2Own (part of the CanSecWest conference), offers rich cash prizes to see who can be the first to defeat the security features in major browser platforms. At the most recent Pwn2Own, none of the four major browsers could stand up to the onslaught, with each falling to remote code execution hacks.

SECURITY: EASIER (AND HARDER) THAN YOU THINK

The fact that all browsers can be hacked doesn't mean that securing your Web browsing is an impossible goal. Like the automobile, Web browsers have quickly evolved from software with few security features, through an intermediary period where security features differentiated one browser from another to where we are now: most browsers have a wealth of similar security features that users assume will be there and mostly take for granted.

As an example, a few years back, I spent several months running the five most popular browsers (Internet Explorer, Firefox, Chrome, Safari, and Opera) through a battery of security tests. I was surprised by how many security features the browsers shared (antiphishing, cookie control, anti-XSS handling, pop-up blocking, file download detection, digital certificate handling, and so on). None of the browsers allowed malware to silently install on my test systems or permitted exploitation beyond simple DoS attacks.

My conclusion: If you are running a fully patched browser on top of a fully patched operating system, the chances of getting hacked—presumably using an exploit for a novel "zero-day" vulnerability—are vanishingly small. An attacker's best chance of success is to fool you into unwittingly granting a malicious application permission to run on your system. This is why most malicious websites offer some kind of executable to install such as bogus anti-malware software, a warning message or some sort of feature "extension" offered as a browser plug-in. This kind of social engineering takes the form of phishing e-mail messages, fake browser plug-ins, Web pop-ups, and so on and is the best way around otherwise solid browser security features. Beware.

It is also true that many other effective browser security features give you fine-grained control over your browsing experience and that can reduce (or increase) your exposure to online risk, depending on your appetite for it. Today, each browser also presents a mix of strengths and weaknesses that will appeal to different users. I will explore these security features in detail.

MAKING A SECURE BROWSER

Many security pundits recommend any browser but Internet Explorer as the best security defense. Although there is some safety in using less frequently attacked software, a better question is which is the safest choice among the most popular browsers? What are the most important security features to look for in a browser, and what are the weaknesses to avoid?

Each new browser typically promises a more secure browsing experience, only to prove that making a truly secure Web browser is difficult. Each of the most popular browsers has dozens of patched vulnerabilities. Even the newest, Microsoft's Edge browser, which gets high marks on security, was the subject of a critical patch in September, less than three months after its public release.

Perhaps the strongest testament to how hard it is to make a secure Internet browser is the fact that even the text-only Lynx browser, which is as simple as a browser can be (it can't even display pictures or video without external programs), has had five vulnerabilities. If attackers can cause buffer overflows in a text-based browser, any browser more complex will have issues.

Given the popularity of Web-based attacks, administrators must consider surfing the Web, in general, as high-risk behavior and respond accordingly. In very high- security environments, Web browsing shouldn't be allowed on high-value IT assets, or should be severely restricted to a set of white-listed websites. Assuming that most of the IT assets in your enterprise don't meet that standard, and that your employees need to browse the Web and access Internet data and applications to do their job, you need a Web browser with an acceptable level of security. If that's the case, keep reading.

HOW TO MEASURE THE SECURITY OF A BROWSER

Vulnerability counts and the frequency of announced exploits account for much of the overall risk to a Web browser, but they are not the only relevant factors. I considered these general categories when reviewing each Internet browser:

Among the factors you should consider when choosing a browser is what other software comes bundled with that browser.

Security model

Each browser is coded on the underlying strength of the browser vendor's chosen security model. This model is what keeps the untrusted network side separated from the more trusted security zones.

For example, you want to know what defenses the vendor includes in the browser's underlying design to prevent malicious use. How is malicious redirection (such as cross-domain cross-site scripting and frame theft) prevented? Is memory secured and cleared against malicious reuse? Does the browser give end-users multiple security domains or zones in which to place different websites according to their level of associated trust? What end-user protections have been built into the browser? Does the browser have the capability of updating itself?

You also want to understand how a browser does or doesn't use the security features of the environment in which it runs. For example, do Web browsers running on recent versions of the Windows operating system take full advantage of the security features of that operating system, including DEP (Data Execution Prevention) or Enhanced Protected Mode?

Feature set and complexity

As in many other areas, complexity is the enemy of security (and safety). Additional features mean more code. More code means more opportunities to exploit that code with unexpected interactions. Conversely, a browser that is too lean in its support of media and content types may not be able to render popular websites, which forces the user to try a more feature-rich browser or install potentially insecure add-ons. Recent events tell us that these plug-ins and extensions are popular tools in the hands of cyber criminals, who use fly-by-night download websites to push malicious browser extensions onto victims' computers.

Among the factors you should consider when choosing a browser is what other software comes bundled with that browser. Google Chrome, for example, generally gets high marks on security, but comes bundled with plug-ins for viewing Flash and PDF content.

Vulnerability announcements and attacks

A key question is how many and how often critical and remotely exploitable vulnerabilities have been found and publicly announced for the browser. Vulnerabilities are a fact of life with any software application. Worth noting is whether the vulnerability counts go up or down as the vendor patches its browser. Pay close attention to news of critical or remotely exploitable holes that might be discovered by independent researchers (or worse, black-hat hackers).

How severe have the vulnerabilities been? Do they allow full system compromise or denial of service? How many vulnerabilities are currently unpatched? What is the history of zero-day attacks against the vendor? How often is the vendor's browser targeted versus a competitor's product?

Browser security tests

How does the browser fare in third-party browser security tests? In this review, all of the products passed the most well-known tests, such as the open source Browserscope.org. Still, important differences could increase (or decrease) your risk of attack and compromise.

Enterprise manageability features

InfoWorld caters to administrators and technicians who need to accomplish tasks across an entire enterprise. It is generally easy to secure a favorite individual browser for personal use, but doing so for an entire business requires special tools. If the browser is intended for enterprise use, you need to consider how easy is it to install, configure, and manage across your user base and in geographically distributed IT environments.

BROWSER SECURITY TIPS

Instead of accusing one browser of being weaker than another, real-world testing has revealed that users should pick a browser that has the security features and functionality they desire, and then enact the following suggestions:

- Don't log on as admin or root when running an Internet browser (or use User Account Control on Windows 10, SU on Linux, etc.).

- Make sure your browser, operating system, and all add-ons and plug-ins are fully patched.
- Beware of websites or phishing scams that try to trick you into running malicious code. If you are unexpectedly prompted to install third-party software while browsing a site, open another tab and download the requested software directly from that vendor's website.
- Limit browser extensions and plug-ins to what you absolutely need. Many of these software add-ons are insecure; some are actually malware in disguise.

Unfortunately, a lot of bad things can be done in the user's context, especially **when the majority of Windows users are running as administrators.**

THE TAMING OF THE SCRIPT

Nearly all real-life exploits as well as non-malicious annoyances (pop-up ads) use JavaScript. That's why it is almost an article of faith among security folk to disable JavaScript when browsing the Web. All the browsers we review here make it easy to do so—that's a big improvement from even five years ago.

However, you'll soon notice that simply disabling JavaScript degrades the experience of using some of the most popular websites and applications out there, from Netflix to Twitter to Google Docs. Savvy users have found ways to live happily without JavaScript, but rank-and-file Web surfers may consider any security feature that breaks Netflix and YouTube the online equivalent of throwing the baby out with the bathwater.

The answer, for some, is browser plug-ins and extensions such as NoScript which lets you selectively enable and block JavaScript on a site-by-site basis. A similar tool, LibreJS by the Free Software Foundation, selectively blocks any proprietary (commercial, closed source) JavaScript, but allows open source JavaScript to run. Although these tools add overhead to your browsing sessions, they can also eliminate intrusive or malicious code from running on your system.

BROWSERS VS. XSS ATTACKS

The Web is full of XSS attack warnings. Essentially, XSS refers to the injection of malicious JavaScript into a legitimate Web page, where it can then be executed in the browsers of innocent visitors. A website is susceptible to XSS if it allows users to upload content to be shared with others and does not thoroughly inspect that content to remove potentially malicious scripts.

Take, for example, a website that hosts a user-contributable blog. Perhaps all the Web developer wanted was for users to be able to upload plain text, and never considered that this would also allow scripting. Because of this oversight, the developer never thinks to filter the content. Bad mistake.

A common test script to determine whether a website is vulnerable to XSS is:

```
<SCRIPT>alert(“XSS is possible”);</SCRIPT>
```

If you can upload that content to the website and you can see the alert when the page is viewed, then the website is XSS exploitable. An excellent tutorial on XSS issues is available on the Open Web Application Security Project website.

What's the big deal with executing a few unexpected JavaScript commands? After all, the JavaScript execution can only do what the user can do in the user's security context, right? Unfortunately, a lot of bad things can be done in the user's context, especially when the majority of Windows users are running as administrators. XSS attacks have been highly successful at reaching outside of the browser to steal confidential information the user would otherwise not want to share. In some cases they can cause buffer overflows and even complete system compromise.

Once during my professional penetration testing days, my team was hired to break into a cable company through its own cable set-top devices. The set-top boxes were completely HTML-enabled and allowed tremendously fine-grained control over the user's experience. They even included a host-based firewall.

A quick check using the test script string noted above showed that the firewall's log file was XSS exploitable. I created a script and "injected it" simply by attempting to attack the device remotely; although my attack was unsuccessful, the set-top device duly recorded the script in its log file. I then called tech support and complained that my set-top box was being attacked, and asked kindly if tech support could confirm it by reviewing the log files.

Within a few seconds I had the tech support's password and shadow files (i.e. their Linux-based password storage files) in hand. The script I had injected instructed the tech support's PC to send their password files to my email account. My coworker and I quickly sifted through the resulting password files, found everything we needed, and quickly took over the cable company's entire network. We had been hired to break into the company's set-top box to see what mischief we could cause, and within a few hours we owned the corporation's global network. That's the power of XSS attacks and why you must take them seriously.

All of the reviewed browsers included robust, built-in XSS mitigations to prevent malicious exploitation to varying degrees. That's the good news. Keep in mind that it may be a good idea to disable JavaScript (if your browser allows it) on Internet sites when a new XSS or JavaScript-enabled vulnerability is announced. And if you come across a website that appears vulnerable to XSS attacks, notify the content owner. Often they are unaware that their user-friendly site could be the host of unintended maliciousness.

BROWSER FINDINGS

As expected, each Web browser had its fair share of security advantages and disadvantages. All of the browsers reviewed here have had years to mature in response to malicious attacks.

Google Chrome

Google's Web browser has come a long way since it was first released back in the waning months of 2008. Lightweight and speedy, with a stripped-down interface, the browser has steadily pulled users from other platforms to become the second-most widely used Web browser in the world after Microsoft's Internet Explorer. Chrome was revolutionary in its browser security model: it pioneered the use of sandboxing to isolate content on each browser tab and prevent Web-based compromises of the entire browsing session. Phishing and malware alerting are built into the browser as is scanning of downloads

Google's Chrome team has been quick to embrace security improvements, such as Strict Transport Security, which allows high-security websites to force the use of a secure (S-HTTP) connection only. That has cut way down on script kiddie tricks like man-in-the-middle attacks. This is true even when the security features are developed elsewhere, such as X-Frame-Options, which were pioneered by Microsoft and are designed to prevent click-jacking. It provided some of the best and earliest protections for reflective CSRF (cross-site request forgery) attacks.

Desktop and mobile versions of the Chrome browser get high marks in evaluations by BrowserScope and others, which measure performance on basic tests like XSS, CSRF, transport security, HTTP cookie handling, and others.

Under the hood, recent versions of Chrome have been less prone to divulging exploitable vulnerabilities. Chrome went unbroken at the annual Pwn2Own hacking contest for three years running. When it finally did fall in the 2012 event, it took a chain of six separate vulnerabilities to create an exploit of the browser's security model. Although Chrome did fall in the most recent Pwn2Own in March 2015, the record will show that the same contest uncovered just one previously unknown and exploitable flaw in Chrome compared to four in IE 11, three in Mozilla Firefox, and two in Apple Safari.

More recently, the Chrome team has focused on providing users with more assurance about websites and Web content. Recent updates have pushed websites that use plain HTTP to switch to encrypted S-HTTP by visually marking those sites as "insecure." The company has also targeted secure sites that rely on insecure cryptographic algorithms (such as SHA-1 to sign their cryptographic certificates) by visually marking such sites.

Still, the Chrome platform isn't without issues, specifically when it comes to the burgeoning population of Chrome extensions: add-on programs designed to extend the browser's functionality. A survey of Chrome extensions published in 2014 found 130 that were malicious and more than 4,700 that exhibited suspicious behavior, such as requesting permissions from the browser far in extent of what was required by the extension's functionality, dropping tracking beacons on Web pages a person has visited, and so on.

Google's Chrome team has taken steps in recent years to make it harder for malicious or suspicious extensions to work, for example: barring extension "side loading" from sites other than the official Google Chrome App Store. The company has also implemented a variety of security features to prevent common attacks like XSS by way of browser extensions.

Firefox has struggled to fend off vulnerability researchers in contests like Pwn2Own. In the 2014 contest four zero-day vulnerabilities were discovered in Firefox, **leading one article to declare it the "least secure major Web browser."**

Mozilla Firefox

Given the importance of the Web browser today, it's easy to forget that a little more than a decade ago the space was a desert. After getting as close as you can get to a monopoly by bundling Internet Explorer with its Windows operating system in the 1990s, Microsoft promptly put the software on a shelf, neglecting feature development and allowing the application to become bloated and slow.

Then came Firefox, a product of the Mozilla Foundation that built on top of the once-dominant Netscape Navigator browser. Originally named Phoenix—a reference to the mythical bird that could rise from its own ashes—the browser that has come to be known as Firefox began snatching market share from IE almost immediately with a slimmed-down, attractive browser that pioneered now-standard but once unheard-of features like tabbed browsing and browser plug-ins.

Firefox also established a reputation for solid security at a time when hardly a month went by without new disclosures of serious and exploitable security holes in Internet Explorer. Notably, the browser introduced browser history "cleansing," support for "same origin policy" to prevent XSS and CSRF, as well as a phishing detection feature.

Over time, the open source browser added robust add-on management to support a growing list of extensions and enterprise features. Out of the box, Firefox has features to prevent side loading of extensions and block suspicious, malicious, and phishing websites. It has robust features for managing website tracking cookies, clearing stored passwords, and erasing browsing history (even if those features are not enabled by default). Using the about:preferences option in the URL bar allows the user to configure dozens of features and security settings.

In other areas, Firefox lags considerably. Most notably: Mozilla has been slower than the competition to implement a multiprocess architecture that allows process isolation (sandboxing). That has made the platform more susceptible to compromises that allow vulnerable content on one Web page to take over the entire browser session.

The result: Firefox has struggled to fend off vulnerability researchers in contests like Pwn2Own. In the 2014 contest four zero-day vulnerabilities were discovered in Firefox, leading one article to declare it the "least secure major Web browser." Ahead of the 2016 contest, organizers said no awards would be offered for Firefox, citing a lack of "serious security improvements" in the platform since the 2015 event.

Similarly, Firefox does not give users an easy way to disable JavaScript support, relying on third-party extensions like NoScript and Ghostery to disable or selectively enable JavaScript support. Competing browsers, notably Chrome, allow users to easily disable JavaScript with a single option.

Finally, although Firefox gets high marks for being an open source application, Mozilla Foundation was criticized last year for integrating the Pocket bookmarking technology with its Firefox browser, folding proprietary and potentially vulnerable code into the otherwise open source Firefox code.

Microsoft Internet Explorer and Microsoft Edge

Internet Explorer is the most popular and the most frequently attacked browser in the world. Its popularity and complexity give it an elevated risk compared to the rest of the competition—a reality Microsoft has taken pains to address in recent updates to its browser and the Windows OS it most often runs on.

Among the most significant improvements in recent releases are the introduction of per-user and per-site control of ActiveX programs and other add-ons in IE 8.

With Edge just hitting the market, and IE still shipped with and supported in Windows 10 (though Edge is the default browser), **expect adoption of the new Edge browser to lag.**

Microsoft also added memory-protection features designed to prevent attack code from running if a memory-related vulnerability is exploited in the browser or an add-on component. Features like DEP / No eXecute and Address Space Layout Randomization are enabled by default in IE 8 and IE 9 and prevent execution of data injected into memory by marking legitimate code in memory.

User-definable security zones (also known as security domains) are also an important feature. Ultimately, less functionality translates to better security. Security zones provide a way to classify various websites as more trustworthy and, hence, suited for greater functionality. You should be able to trust your company's websites more than a site offering pirated software or a small Web page served up by someone you don't know. Security zones allow you to set various security settings and functionalities based upon the website's location, domain, or IP address.

Security domains are used in every computer security product (firewalls, IPSes, and so on) to establish security boundaries and areas of default trust. Having a security zone in a browser extends that model. IE has the most mature implementation of security domains among major browsers.

Browsers without security zones encourage you to treat all websites with the same level of trust—or reconfigure the browser or use another one for less trustworthy sites before each visit.

Other key features Microsoft added support for include the SmartScreen Filter and SmartScreen Application Reputation, which detect socially engineered malware and phishing attacks, as well as malicious downloads. Following Chrome's lead, IE has also improved privacy protection features, starting with IE 9 including Tracking Protection functionality.

Internet Explorer 10 and 11 continue those stability and security improvements. IE 10 added enhanced memory protection and support for the HTML5 sandbox attribute, which restricts iframe elements that contain untrusted content. Importantly, IE 10 also added Enhanced Protected Mode based on the feature introduced with Windows 8, which includes AppContainer, a security feature that keeps pages from reading or writing to the rest of the operating system. IE11 builds on the improvements in IE 10, adding verification that add-ons such as browser helper objects, toolbars, or Active X controls have an AppContainer-compatible flag.

Notably, Internet Explorer has best-in-class enterprise support, superior security granularity, and multiple security zones in which to deploy websites with different trust requirements. It is the only browser with serious enterprise management features, providing more than 1,200 customizable settings across multiple security zones. For example, the U.S. government requires what is called Federal Desktop Core Configuration on all of its software and Federal Information Processing Standards ciphers only. Tens of millions of PCs fall under these requirements. Only IE allows these policies to be enforced across all desktops. It is difficult to achieve with any other browser.

Despite those improvements, however, Microsoft is ending continued development of Internet Explorer and focusing on its recently introduced Edge browser, a ground-up remake of its Web browser with security among the top considerations. Many of the most significant security improvements in Edge come from features it does not support, namely legacy technologies such as VML, VB Script, toolbars, browser helper objects and, most especially, ActiveX. In their place, Edge adds a new rendering engine with expanded support for the rich media capabilities of HTML5.

Edge also extends application sandboxing functionality throughout the application, removing the desktop-only sandboxing found in IE 10 and 11. In contrast, Microsoft Edge allows all content processes to run in application containers regardless of context. The feature is on by default.

With Edge just hitting the market, and IE still shipped with and supported in Windows 10 (though Edge is the default browser), expect adoption of the new Edge browser to lag. However, Microsoft's commitment to ending support of IE along with earlier versions of Windows will soon create significant pull to Windows 10 and Edge among both consumers and businesses, so the time to prepare for that future is now.

Overall, Opera is a solid browser but a small fish in a very big pond. The biggest challenges facing this 20-year-old application stem from the limited resources of its parent company, Opera Software.

Opera

The Opera browser has worn the mantle of that "other" Web browser for a long time—a really long time. The first version was officially released in 1996, a spin-off of the Norwegian telecommunications firm Telenor.

Two decades later, Opera is a solid browser that has seen its market share grow slowly in recent years. It now accounts for a bit less than 2% of the worldwide browser market, with 350 million users worldwide -– just 55 million use the desktop version of the software. Still it is a popular choice in some of the fastest-growing Internet markets: developing areas such as Africa, Russia, India, and Indonesia.

Opera offers versions for Linux, Mac, and Windows. Still, much of its development has focused on the burgeoning mobile device market, where it has pioneered features to accelerate mobile content delivery. Opera Mini for mobile devices and an Android version account for the bulk of usage.

That said, Opera deserves more market share in the PC world. It has impressive security granularity, good anti-DoS handling and strict Extended Validation certificate handling. Like competing browsers, it offers standard privacy features such as browsing history cleansing and cookie management. More recent versions have boosted fraud and malware protections (enabled by default) that are in line with competing platforms like Chrome. Websites that are on lists of known suspicious sites display warnings. Potential phishing sites are marked by fraud warnings. Also like other leading browsers, Opera automatically pushes out updates and no user action is required to install them—a key feature.

As a closed source browser, Opera presents challenges for privacy- and security-conscious users who want browser application code to be publicly audited. However, in recent years, Opera has fostered closer ties with Google and tacked closer to Chrome. The company jettisoned its Presto rendering engine for Google's WebKit in 2013 and leveraged code from the Chromium project. When Google switched to the Blink engine later that year, Opera followed and has also committed to contributing back to the Blink engine. That has made significant portions of the underlying code of the Opera browser open source, somewhat alleviating concerns.

The switch to Chromium and Blink has also reduced much of the pressure on Opera to maintain and defend an aging Web rendering engine, which had become an issue for the company in recent years. Embracing Chromium has allowed the company to focus more on feature development.

Like other browsers, Opera leverages native Windows security features such as DEP and Address Space Layout Randomization, giving it parity with competing browsers, though not distinguishing it in any way.

Overall, Opera is a solid browser but a small fish in a very big pond. The biggest challenges facing this 20-year-old application stem from the limited resources of its parent company, Opera Software, especially compared with those of its chief competitors Google, Microsoft, Apple, and the Mozilla Foundation. The announcement in February 2016, that a group of Chinese investors has offered to acquire Opera Software for a reported $1.2 billion could add a new chapter to the Opera story: providing much needed cash to continue product development while also giving Opera a foothold in the massive Chinese market. But at what cost in security and privacy protections? Stay tuned.

Apple Safari

Apple's Safari browser is a late entry to the competition, given that Apple only launched the browser in 2003 and that its distribution was limited to OS X until 2008, when the first stable Windows version of the browser was released. Safari never fulfilled the promise of being a faster, cooler, IE-killer (Chrome appears to have beaten Apple to the punch on that). But its market share is growing due in large part to the popularity of Apple's iOS, which runs on iPads and iPhones and bundles Safari as the default browser.

On the security front, Safari has often been late to the party. It was among the last of the major browsers to add now-standard features like anti-phishing and antimalware protections and support for Extended Validation certificates to verify websites. Still, Apple's browser has caught up and now offers those features as well as a solid complement of standard browsing security offerings. On the front end, Safari offers good pop-up blocking, good local password protection, and a surprisingly accurate anti-phishing filter, features for managing website cookies and data, a private browsing option, browser history cleansing, and a feature to disable JavaScript on Web pages. Safari always automatically prompts the user before downloading files, and it prevents some high-risk files from being executed before downloading. Safari also has good default cookie control.

Under the covers, Safari sports process separation and a robust sandboxing model that separates key browser functions like Web page rendering, networking, and plug-in handling with isolated sandboxes. Safari isn't hacker-proof (researchers demonstrated a method of breaking out of its sandbox using a previously undiscovered "use after free" vulnerability at last year's Pwn2Own competition), but it has generally stood up well in comparison to its competitors in such contests.

Unfortunately, Safari falls short in other areas. The browser is alone among major platforms in continuing to support SSL Version 3, which is widely considered insecure, and in not supporting recent improvements that can streamline site verification and session integrity, such as OCSP (online certificate status protocol) stapling and session tickets.

Any browser you use should support TLS and offer it by default to S-HTTP-protected websites. SSL 1.0 and 2.0 are considered insecure and are no longer supported.

THE IMPORTANCE OF SECURE BROWSER CONNECTIONS

Although most users don't know it, their Web browser plays a key part in determining the strength of the ciphers used between their client and an S-HTTP-protected website. Encryption ciphers used in the SSL/TLS negotiations can range from very strong to weak, and involve asymmetric ciphers, symmetric ciphers, key exchange algorithms, and hash functions.

Long an arcane and overlooked part of the browser security discussion, connection security has been elevated to a high-priority issue in recent years. That change has been driven by a number of events, among them the revelations of former NSA contactor Edward Snowden about the lengths to which the U.S. intelligence community and other governments will go with online surveillance—including the use of man-in-the-middle attacks.

Just as important have been serial revelations of serious security vulnerabilities in common online encryption protocols and toolsets. The Heartbleed vulnerability affecting OpenSSL was revealed in April 2014 and prompted frenzied updates to that software suite and a wholesale review of the OpenSSL code base.

That was followed by the discovery of the Poodle vulnerability in October 2014, which affected CBC-mode (cipher block chaining) ciphers used in SSL Version 3 and could allow a network attacker to extract plaintext content such as cookie information from an SSL-encrypted Web connection.

Finally, in early 2015, researchers discovered the FREAK vulnerability in OpenSSL and Apple SSL/TLS clients that permit man-in-the-middle attacks that could downgrade strong encryption used by a slew of websites to a weaker (and breakable) alternative encryption type.

The long and short of these serial flaws has been to focus public attention and the attention of the security and development communities on tightening and improving browsers' use of encryption and eradicating the use of insecure and vulnerable ciphers.

Notably SSL has been replaced by TLS 1.0 as the S-HTTP standard. It is possible in many browsers to select which SSL and TLS versions are enabled. Any browser you use should support TLS and offer it by default to S-HTTP-protected websites. SSL 1.0 and 2.0 are considered insecure and are no longer supported. Following the revelation of the Poodle vulnerability, SSL 3.0 was also deprecated (in June 2015) and most browsers dropped support of it as well, though Safari still supports its use.

CIPHERS

Browsers support a range of common SSL/TLS symmetric ciphers. Commonly used ciphers include AES-GCM (Advanced Encryption Standard-Galois/Counter Mode) or AES-CBC with good integrity checks via SHA-1 or SHA-256 and forward secrecy.

Among SSL/TLS asymmetric ciphers, ECC (Elliptical Curve Cryptography) is now considered the gold standard. ECDSA (Elliptic Curve Digital Signature Algorithm) and ECDH (Elliptic Curve Diffie-Helman) are included in the U.S. government's Federal Information Processing Standards as part of what is called Suite B. Browsers must support Suite B to be considered for use by the U.S. government. Browsers should offer ECC as the first asymmetric cipher, followed by RSA or Diffie-Helman.

HASH FUNCTIONS

Common cryptographic hash functions include, in order of strongest to weakest: SHA-512, SHA-384, SHA-256, SHA-1, and MD5. MD5 has been demonstrated to have greater cryptographic weaknesses. SHA-1 and SHA-256 are now the most popular hashes. Suite B recommends the use of SHA-2, specifically SHA-256 or SHA-384.

KEY SIZES

SSL/TLS symmetric key sizes may range from 40-bit (the old SSL standard) to 512-bit (very strong). Symmetric key sizes of 128-bit to 256-bit are considered secure for most normal security operations. 256-bit keys are standard, although 128-bit keys are still popular.

In general, longer key sizes are stronger within a particular cipher. For example, a 256-bit AES key is stronger than a 128-bit AES key. However, you can't always use key size as a strength measurement between cipher families. For example, 384-bit ECC is considered stronger than 1024-bit Diffie-Hellman. Plus, you can have a really horrible cipher with a really long key size and still have poor protection. As a matter of fact, users should be wary of newly announced ciphers from questionable sources that claim ultralong key sizes (e.g. 1 million bits, etc.). A good cipher doesn't need an ultralong key size. If the cipher algorithm is good, smaller key sizes can be used and the cipher will remain strong.

BROWSER CIPHER ORDER

When a browser first connects to a SSL/TLS protected website, the first packet in the handshake includes the browser's preferred cipher order, including all the ciphers the browser currently supports. Both the client and the website must agree on which ciphers to use before they continue. With any luck, the website will pick the strongest cipher the client supports.

By offering the strongest cipher first, the browser increases the likelihood that a Web server will pick it, if it supports it. Using stronger cipher orders shows a browser vendor's commitment to cipher strength.

THE BROWSERS COMPARED

Browsers in this review largely support similar—if not identical—lists of strong cipher suites. Most major browsers rank strong, elliptic curve algorithms among the top five or 10 cipher suites they offer. Chrome (Version 48) has ECDSA and ECDH with TLS, AES and SHA 256-bit key as the first cipher showing. That's also true of Firefox (Version 43) and Opera (Version 35) browsers. Safari differs, offering the SHA 384-bit key as its top choice.

Still, Safari is alone among major browsers in continuing to support a number of weak cipher suites, including ciphers using RC4, which has documented weaknesses that make it vulnerable to compromise.

I encourage you to check out the individual reviews to see the security strengths and weaknesses of each browser. (Note that my tests no longer reflect the most current versions of the browsers and that some security features may have changed.) Most of all, however, don't forget the main lesson: A fully patched system prevented all silent attacks regardless of the browser. If you keep your browser, its add-ons, and the underlying operating system patched and up to date—and you're careful about which "helper" applications you download and run—the Internet will be a much safer place.

An InfoWorld security columnist since 2005, Roger Grimes holds more than 40 computer certifications and has authored eight books on computer security.