No Web Site Left Behind: Are We Making Web Security Only for the Elite?

Terri Oda, Anil Somayaji
School of Computer Science
Carleton University
Ottawa, Canada
Email: {terri,soma}@ccsl.carleton.ca

***Abstract*—The web is riddled with flaws that make it unsafe. Protection methods exist, but current web security solutions are often designed to be deployed by programmers and security experts. Unfortunately, programmers and web security experts are not always available: many sites are created by graphic designers with more artistic backgrounds, and others involve web applications installed by non-programmers who want a website to fit a targeted need. These non-expert page creators may find web security solutions confusing and difficult to implement because they assume significant technical expertise. While solutions designed for experts are valuable, solutions for non-experts are needed to make the web safer.**

***Keywords*-computer security; web security; web development; usable security**

I. Introduction

The web is becoming a dominant threat within computer security. In the second half of 2009, 82% of reported commercial vulnerabilities were related to web technologies (up from 78% in the first half of 2009) [1]. A report of cases investigated by 7safe in the UK found that 86% of studied attacks exploited a vulnerability in the web interface, while only 14% targetted other parts of the infrastructure [2]. Attackers know that valuable data and credentials pass through the web, and the web interface is usually more accessible to outsiders, thus making the web a logical point of attack.

Those wishing to protect their websites are not without resources. There are a variety of secure coding guides available as well as commercial software to aid in finding bugs and preventing attacks. Research is being done into new security policies and other ways to protect web pages. But while good solutions exist, they do not seem to have had the widespread impact on web security that one might hope. Vulnerabilities remain pervasive: A recent survey from WhiteHat Security claims 83% of sites have had at least one serious vulnerability, with 64% of sites showing a serious vulnerability at the time of the report [3].

It is easy to blame web developers for the current state of affairs. After all, they are responsible for producing secure sites and protecting their users. Presumably, any sufficiently experienced programmer should be able to learn how to write secure code. But while we may call them web developers, are web developers really programmers?

Although many sites are created by trained professionals, at least some are created and maintained by random individuals: pet owners, parents, sports teams, garage bands, etc. While some of these people may have programming experience, many of them simply want a digital soapbox to stand on or place to chat with others. Even web professionals are not all programmers: Many come not from a computer science background, but an artistic one.

However, current web security solutions rely heavily upon programming expertise. Much of the jargon used requires understanding of concepts related to computer programming, and fixing problems may require extensive reimplementation of sites using more secure programming practices. As a result, explanations of web security issues may be incomprehensible to many web page creators, both professionals and non-professionals.

Because existing web security literature and the available solutions are targeted at more experienced web programmers, we are indirectly helping perpetuate the myth that people with smaller sites need not worry about security. In addition, by requiring significant background knowledge we may have made the cost of security knowledge too high relative to the benefits for small sites.

This is a hard problem to solve, but as it stands we run the risk of ghettoizing smaller sites. While securing large sites is clearly valuable, can we really be said to be providing security for the web if so many sites are left behind?

II. Page Creators Are Not All Programmers

When computers first became popular, many magazines provided articles based upon the idea that everyone would write their own custom programs. But while hobbyists then may have typed in code from magazines, the majority of modern computer users would rather use programs created by others. We sometimes assume that the web is much the same: many users, fewer programmers, and those programmers are often professionals who write code for a living.

But the web is less like computer programming and more like desktop publishing. The desktop publishing revolution

gave individuals easy ways to produce newsletters, posters, books, and other written materials for wide distribution. Telephone poles are still covered with flyers from local bands, small businesses, and others. But many people want to get their message further than the local street corner, so freely available web applications and HTML editors hare replaced the printers and desktop publishing software. Anyone can blog about their cat, and organizations who were printing out flyers may now run small web sites where members can communicate using a bulletin board system or keep their contact information up to date. Anyone who wants to share can make use of the web.

The web has also attracted people who formerly would have pursued print design. You can see this in the titles used for web professionals. While "Web developer" ranked as the most common title in a 2008 survey, many of the titles show ties to graphic design: web designer, designer, creative director, art director among them [4]. You can also see the artistic roots of web design by observing the other services offered by web design companies: logo creation, brochures, business cards and other services traditionally offered by graphic designers imply a very artistic staff. There is always a market for making your message more beautiful. The end result is that while many sites on the web are created by programmers, others are created by graphic designers and by individuals from very diverse backgrounds.

III. Web Security is for Programmers

When we look at educational materials regarding web security, many assume a high level of technical proficiency/comprehension in the reader. For example, one can tell from the name alone that the 2010 CWE/SANS Top 25 Most Dangerous Programming Errors [5] is intended for those interested in programming, a fact reinforced by the list of intended users. Even the OWASP Top 10 [6], a similar list for web application security which could in theory be accessible to a more general audience, describes itself as "a great start to your secure coding security program."

Overviews like these lists serve as a starting point to understanding web security, yet they are unsuitable for many web professionals and amateurs who run websites. If these people do not understand the issues well enough to analyze their risks, they cannot make informed security decisions.

Although we might wish it otherwise, there are no magic bullets in web security. Many solutions require reimplementation of parts of a website to add in better input and output sanitization [7], complex HTML modification [8], or other checks. Though we might speed the discovery of bugs using a web application vulnerability scanner or other tools, the end result is that someone will eventually have to change the code. In theory, such changes should be done by a programmer.

Unfortunately, by making it seem like security is the job of the programmer, we give the impression that only programmers need to worry about security. While we may want solutions to be provided by programmers right now, it does not follow that only programmers need to understand security. If that were true, why do we attempt to teach users about how to create and use passwords? Users are exposed to risk, and thus are expected to take on some of the responsibility for securing their data. Similarly, web site maintainers suffer the most if their sites are compromised, even if they have just downloaded and installed some software, not written any themselves. If all web site creators and maintainers were more knowledgeable about security, they could migitate their risks by being more vigilant about updates or make better choices in the software they use.

IV. Non-Programmers Need Security

It is tempting to assume that sites created and maintained by non-programmers are not a problem. After all, the large sites seem most likely to garner attack since they are more valuable: they are more likely to handle large sums of money, they have more visitors so they can provide greater exposure, and they often have access to a large number of user credentials.

It is hard to gather numbers for how many sites are created by non-programmers. It may be that even many large sites are developed by web firms whose staff is largely artistic. Some businesses may allocate so little money or time to their web sites that security is impossible even for experts, let alone designers who would need to learn the necessary skills within their own busy schedules. But it is not like such sites advertise their potential insecurity.

What we can examine to get a picture of the scale of the problem are amateur websites. Websites for community groups, parents, sports teams, etc. are fairly likely to use free software packages to provide things like blogging software or web forums. It is fairly easy for a small group or individual to rent a virtual machine for hosting, plus many people have basic hosting provided with their Internet service. In addition, such custom solutions allow individuals and groups more control over their data and the applications they use. This can make such a setup more attractive than other managed solutions. If we can get a rough idea of the number of installs of such software, we can get a rough idea of how many sites may be maintained by non-programmers.

There are many popular free web applications for webmail, content management, and other services. Two of interest are Wordpress [9], a blogging package, and phpBB [10], a web forum application. To get a sense of the number of such sites available, we searched Google for "powered by Wordpress" which returned 78,300,000 hits and "powered by phpBB" which returned 363,000,000 hits. While not every Google hit on these phrases correspond to working install, they do indicate a large number of installs and a user community which discusses them.

Do these sites represent a significant risk to the world? With headlines like, "Half-Million Sites Mostly Running phpBB Forum Software Hacked In Latest Attack" [11] we can make an educated guess that these do indeed represent an attack vector worth noting.

Many smaller sites assume that they are not at risk because of their size. Still others assume they are safe because they do not handle money. But the who and why of attacks are not what one might assume. Social/Web 2.0 sites are currently the most likely to be attacked, with retail sites and financial sites lagging behind [12]. And the goals of hacking are also not quite what users assume: the number one goal for attackers seems to be defacement and planting malware, with monetary loss fourth on the list of goals [12]. In a mid-2008 report, IBM estimated that 75% of web sites with malicious code were legitimate sites that had been compromised [13].

One might think that if the goal of an attack was to spread malware, then larger sites would be ideal. But larger sites are better protected, so the cost of attacking them is relatively high. Attackers have automated tools to exploit web sites [14], and these tools often target popular software such as WordPress and phpBB [15], [11]. Automated compromise tools make it worthwhile for attackers to attempt to compromise smaller sites. They may have fewer users to compromise, but they still have enough to justify the effort.

V. Towards Web Security for Everyone

Users are said to reject security advice for rational reasons, since the cost of following the advice is higher than the benefits they see [16]. So while it may be worthwhile for attackers to compromise smaller sites, can we make it worthwhile for smaller sites to improve their security?

It may be an uphill battle. Many people have a negative impression of security as unusable, frustrating or pointless [17]. In addition, perceived risks may be much lower than actual risks. Risks for smaller sites are very difficult to classify. Since smaller sites handle less or even no money, risks cannot necessarily be turned into dollar losses. Instead, risks for site operators may be more intangible: embarrassment, angry users, private data exposed. For many these risks may not seem so severe. If someone installs malicious code on a your soccer team's website, you could just apologize and fix it; people will probably stay on the team even if the web site is broken. It may be easier to just do backups in case the site is defaced than to do than security updates. If the cost of a breach is sufficiently low, it is very difficult to create a solution that will seem worth the trouble.

This is not to say that smaller sites should not be concerned about security. Compromises on small sites can still hurt site operators and users. Many users do care about downtime and privacy. Users reuse passwords, so compromised passwords can hurt users elsewhere. Sites can be used to attack other sites or users. For small businesses, a compromise can mean real loss of revenue. For everyone, time spent on recovering a hacked website is time not spent on other things. However, while these costs are real, they may be apparent only when it it too late.

So how can we provide security for people who, effectively, do not care about security?

One approach is to assume that web developers are going to make security mistakes and we should provide environments and tools that minimize the harm they can create [18]. The web developer perceives no cost when security is provided as part of the environment. Some of the infrastructure of the web already has security features built in: JavaScript's sandbox and same origin policy both significantly limit the scope of potential attacks. Another route would be to target tools that appeal most to those uninterested in programming. For example, HTML editors such as Adobe Dreamweaver are popular among those who want to create websites without learning to program. Modifying such tools so that they produce more secure code might be one way to integrate secure coding practices as part of the workflow of a web page creator without diverting their attention from the task at hand.

We can also attempt to get people to care about security. Although security education is not always effective for disinterested users [16], without education we are unlikely to have interested users. Better educational materials can allow users to become more aware of risks and stay abreast of modern attacks: it needs to provide advice which is reasonable relative to the risks, relevant to the users, and up-to-date. Perhaps security messages need to be more like marketing messages: short, for a specific target audience, and perhaps even entertaining in order to better engage the audience with the message.

Next, it may be useful to provide some security solutions that require minimal intervention. Some research has been done on security policy mechanisms such as the Origin: header [19], Content Security Policy (CSP) [20] and SOMA [21]. The risk for programmers is that in trying to make such solutions sufficiently flexible they become totally unsuitable for less sophisticated web developers. For example, the CSP specification includes mathematical set definitions and pseudocode that may be impenetrable for many page creators. While heavyweight security policies have a place, they have a higher cost of use, and can be more expensive to implement than the costs of a breach, especially for smaller sites.

Another way to handle the problem is to push the security onto someone else along the line. Site designers are not the only ones who can provide security enhancements. Security policies can also be configured by systems administrators. (For smaller organizations, the system administrator and the web developer may be the same person.) Similarly, web application firewalls can provide another layer of security, but they may require complex tuning and monitoring, or they may become obsolete quickly as heuristics no longer

detect all attacks. Such programs have largely been created by commercial interests, and the space might benefit considerably from more research and less hype.

Web security can also be pushed on to the user level, with more complex web security suites and improved browsers. These in theory could protect users even if they visit sites with vulnerabilities and exploits. Currently, many popular security extensions require significant work on the part of the user. The popular browser security extension, NoScript, made Computerworld's list of "Top 10 Firefox Extensions to Avoid" [22] due to its poor usability. A more usable solution would probably be akin to a virus scanner (and likely bundled with one) in that it would be something the user could run in the background. The web is in some ways the worst possible scenario for antivirus because every page is a custom application, and many change regularly. Current packages focus heavily upon drive-by-downloads and may rely upon whitelisting for popular domains. This leaves users very exposed. But while client-side solutions may seem impossible at a glance, software such as NoScript shows that end-users may be more interested in security than one might expect. It is our hope that researchers will be able to find more usable solutions on the client side.

Right now, the last resort is that if a user is convinced that security is necessary, they can hire expert security help. But how many people would hire a security guard to watch their bicycle? How many would become a police officer to ensure that their home was safe? The current web security models assume you either are willing to hire an expert or become one, both of which are excessive for many web sites. It seems like there should be a large market for the web equivalent of a bike lock, if only such a thing existed. But for now, expert help is perhaps the safest security suggestion.

VI. Conclusion

While it is incredibly valuable to provide high quality security resources and solutions for those best able to understand and implement them, this should not be done exclusively. Even web professionals may find the existing literature challenging due to the assumptions made within. As a result, web security is mostly accessible to the elite of the web world. But if we want to provide a safer web experience for all users, we cannot assume that only the elite matter. Users will continue to visit smaller sites for a variety of reasons. Dealing with this problem is difficult, as it is hard to motivate smaller sites to care about security, and even harder to teach them to be secure. If we can view this as a challenge rather than a reason to avoid trying, we stand to make great gains in overall security. As of now, we still have a long way to go.

Acknowledgements

This work was supported by NSERC through the ISSNet strategic network and the Discovery grant program. We also would like to thank the members of the Carleton Computer Security Laboratory who have provided ongoing feedback for this research.

References

[1] "Web application security trends report – q3-q4, 2009," *Cenzic Inc.*, 2009.

[2] "UK security breach investigations report: An analysis of data compromise cases security breach investigations report: An analysis of data compromise cases," *7safe*, 2010.

[3] "Fall 09 website security statistics report," WhiteHat Security, Tech. Rep., 2009.

[4] A List Apart, "Findings from the a list apart survey for people who make websites, 2008," 2008. [Online]. Available: http://aneventapart.com/alasurvey2008/

[5] "2010 CWE/SANS top 25 most dangerous programming errors," The MITRE Corporation, Tech. Rep., Feb 25 2010. [Online]. Available: http://cwe.mitre.org/top25/

[6] "OWASP top 10," OWASP, Tech. Rep., 2007. [Online]. Available: http://www.owasp.org/index.php/Top_10_2007

[7] "XSS (cross site scripting) prevention cheat sheet," *OWASP*, Jan 16 2010, available from http://www.owasp.org/.

[8] C. Jackson and H. J. Wang, "Subspace: Secure cross-domain communication for web mashups," in *Proc. of the 16th International World Wide Web Conference (WWW2007)*, Banff, Alberta, May 8-12 2007.

[9] "Wordpress." [Online]. Available: http://wordpress.com

[10] "phpbb." [Online]. Available: http://www.phpbb.com

[11] "Half-million sites mostly running phpbb forum software hacked in latest attack," *CyberInsecure.com*, May 12 2008.

[12] "The web hacking incidents database 2009: Bi-annual report," *Breach Security*, Aug 2009.

[13] "IBM Internet Security Systems X-Force® 2008 mid-year trend statistics," IBM Global Technology Services, Tech. Rep., Jul 2008.

[14] "X-force® 2009 trend and risk report: Annual review of 2009," IBM Security Solutions, Tech. Rep., 2009.

[15] F. Howard, "Wordpress injection attack and "affiliate ping-pong"," *SophosLabs blog*, 2010.

[16] C. Herley, "So long, and no thanks for the externalities: The rational rejection of security advice by users," *Proc. of The 2009 New Security Paradigms Workshop (NSPW'09)*, pp. 133—144, Sep 8-11 2009.

[17] A. Adams and M. A. Sasse, "Users are not the enemy," *Communications of the ACM*, vol. 42, no. 12, pp. 41–46, 1999.

[18] G. Wurster and P. C. van Oorschot, "The developer is the enemy," *New Security Paradigms Workshop (NSPW'08)*, Sep 2008.

[19] A. Barth, C. Jackson, and J. C. Mitchell, "Robust defenses for cross-site request forgery," in *Proc. of ACM Computer and Communications Security (CCS'08)*, 2008.

[20] B. Sterne, "Security/csp/spec," Mozilla Corporation, Tech. Rep., 2009. [Online]. Available: https://wiki.mozilla.org/Security/CSP

[21] T. Oda, G. Wurster, P. van Oorschot, and A. Somayaji, "SOMA: Mutual approval for included content in web pages," in *Proc. of ACM Computer and Communications Security (CCS'08)*, Oct 27-31 2008, pp. 89–98.

[22] P. Smith, "Top 10 firefox extensions to avoid," *Computerworld*, Apr 2007.