INFORMATION SECURITY

Insider Edition

AUGUST 2013

SOFTWARE SECURITY FOR WEB DEVELOPERS

Harden your organization's Web infrastructure by fostering internal secure Web application development.

USING WEB APPLICATION FIREWALLS FOR DEFENSE-IN-DEPTH

AVOIDING COMMON WEB APPLICATION VULNERABILITIES

TechTarget

Fostering Secure Web Application Development

Learn about the recent threats against Web applications and how you can help application developers acquire the necessary security tools and training to keep your organization's Web presence safe.

S**TUDIES SHOW** that three-quarters of Web applications have a security vulnerability. Often in a rush to get apps live, software developers push these apps into production with flawed code that attackers can easily exploit, often by using simple, automated vulnerability-scanning tools. As a result, it has become increasingly important to go beyond just hardening an organization's Web infrastructure and foster internal secure Web application development to prevent attacks.

This *Insider Edition* will help security professionals understand the evolving threats against Web applications, how they can prepare and inform developers with security tools and training, and how to deploy technologies to keep your organization's Web presence safe.

In our cover story, "Software Security for Web Development: Providing a Helping Hand," Dan Cornell offers advice for information security managers on how to evangelize the need for secure Web development. Cornell discusses how security pros can help Web developers create secure applications, addresses the proper role of tools and training and explains the need for ongoing collaboration between Web developers and security experts.

Next, Mike Chapple explains how to improve your defense-in-depth security strategy with Web application firewalls and a strong software development lifecycle process. Mike Cobb wraps up this *Insider Edition* with a look at today's most common Web application vulnerabilities and the countermeasures security professionals need to implement to protect against them. ■

Kara Gattine, *senior managing editor*

SOFTWARE SECURITY FOR WEB DEVELOPMENT: PROVIDING A HELPING HAND

Learn how to help foster secure software development within your organization.

By Dan Cornell

THE IMPORTANCE OF information security has permeated many enterprises to varying degrees, yet software security—and in particular Web-based software development—remains a vexing challenge for chief information security officers.

It's a great understatement to say that Web development teams have not traditionally been focused on security. Their incentives, and consequently their priorities, have been tied to implementing new features and meeting deadlines. Many development teams aren't even aware that what they do affects security; instead they view security as a job handled not by developers, but by products such as firewalls and antimalware systems.

However, the changing threat landscape and increasing frequency of Web application attacks has forced security-focused organizations to address Web application security through secure software development. The simple truth is that security measures that are "bolted on" after the software development process is complete have proven to be powerless in countering Web application attacks. For instance, firewalls only control the traffic allowed to pass over networks, but Web applications *must* be exposed *outside the firewall* in order to be accessed by legitimate users. Antimalware is similarly ineffective; it only looks for malicious code, not problems with legitimate applications.

Attackers have come to learn that nearly all Web applications can be exploited via the mistakes developers made when building them. Using any one of a long list of common Web application vulnerabilities, an attacker can make the software misbehave in any number of ways, including granting access to unauthorized data.

Antimalware is similarly ineffective; it only looks for malicious code, not problems with legitimate applications.

For CISOs, countering Web application attacks through secure software development is often a daunting proposition. While the scope of many CISOs is growing and often includes C-level access and influence, in few cases does that reach extend into the software development organization. Affecting a fairly significant set of changes to the way in which applications are developed requires not only evangelism to make the case for secure development, but also pragmatism in providing effective tools and training. Security managers must seize the opportunity to position themselves as trusted resources to foster secure software development within their organizations.

EVANGELIZING THE NEED FOR SECURE WEB DEVELOPMENT

In order to take on this role, security managers must be able to work hand-in-hand with development teams. However, developers can be wary of outsiders attempting to impose standards and behavior on them.

To enable security teams to have a positive impact on software security in their organizations, it is critical that the security professionals build and maintain credibility with developers.

The problem is that most information security professionals do not come from a strong development background—especially backgrounds in modern Web application development environments. This makes it challenging for them to effectively communicate with the development team. For that reason, security professionals shouldn't be afraid to say "I don't know" when discussing highly technical issues, yet they can't let developers brush them off or obfuscate important issues with technical mumbo-jumbo. Evangelism is, therefore, a balancing act.

Security professionals can improve their credibility with developers by providing information on real-world threats and security-related business demands that is customized for the organizations and development team needs. Companies fall under a variety of regulatory and compliance requirements, ranging from the Payment Card Industry Data Security Standard for those accepting credit card transactions, Health Insurance Portability and Accountability Act for those dealing with personal medical information and the various customer data breach notification laws. Providing the developers with data that demonstrates the negative impact to organizations that have suffered a breach or compliance failure provides valuable context that enables the developers to understand the importance of incorporating security into their development process.

Default rule sets and reporting are geared toward penetration testers and security professionals.

There are a number of sources available that provide this kind of detail. For example, vendors WhiteHat Security and Veracode report on their application testing efforts and the prevalence and longevity of different classes of software vulnerabilities. In addition, the annual Verizon Data Breach Investigations Report describes the types of attacks being used in actual breaches. Security teams that curate and distribute this information internally can help the development team focus on the most relevant and critical security concerns.

PROVIDING SECURITY TOOLS AND TRAINING

Security pros must understand the tools used by the development team in order to better integrate security into their work flow. Security tools are a valuable component of any software assurance program, and security professionals can support development teams by understanding how developers use their tools while looking for ways to augment and tune those toolsets.

In addition, when putting security tools such as automated security code analysis tools or dynamic Web security scanners into developers' hands, security teams must remember that these tools need to be configured and tweaked, and that the Web developers will use these types of tools differently than the security team would. These tools can be a great help in identifying common—and dangerous—Web application flaws like cross-site scripting and SQL injection, but can overwhelm development teams with information that lacks relevance and can be hard to understand. For example, default rule sets and reporting are geared toward penetration testers and other security professionals.

However, before exposing the developers to these tools, security teams need to ensure the rules being used focus on high-impact vulnerabilities with signatures that have a low likelihood for false positives. Failing to properly tune the security testing tools like this can result in output that the developers find overwhelming, irrelevant and even detrimental, forcing them to waste time chasing after false positives.

Also, each tool's reporting capabilities need to be modified to include information that helps the development team locate vulnerabilities in software code, as well as include specific remediation recommendations that make those reports more actionable. Some Web developers will be more comfortable with source code security scanners because they show the specific lines of code responsible for the bad application behavior. Other Web developers might be more comfortable with dynamic Web application scanners because offer working example payloads showing how a vulnerability might be exploited. Security professionals need to get a feel for how their Web development teams work and tailor tool recommendations accordingly.

The objective of putting new tools in developers' hands should be to transition vulnerabilities into software defects. Vulnerability data might be of some interest to developers, but typically what they really care about are software features and defects they need to address to alleviate security concerns. This may seem like a subtle difference, but it is an important one because it reflects the way that developers plan and manage their workloads and can highlight the lack of understanding that many security teams have about the internal operations of development teams.

THE NEED FOR ONGOING COLLABORATION

Tools should also help facilitate communication between security pros and developers. When a security analyst emails a lengthy PDF report detailing found vulnerabilities to the development team, it can be perceived as a hostile or standoffish way to communicate.

A more effective approach is to use the systems developers are already using, namely defect tracker tools, to demonstrate that they understand the software development process and want to work *with* development teams. Rather than simply emailing out impersonal reports, security professionals should use the results from security testing activities as an opportunity to sit down with development team leads to discuss the findings and begin the process of using vulnerability data from the report to identify Web application software defects.

This collaborative approach to resolving vulnerabilities provides better opportunities for security teams to work hand-in-hand with Web development teams to address flaws in production Web applications, and mutually improve the security of future Web application deployments.

Security teams can also identify security champions within the development ranks to maintain lines of communication. These champions are developers who gain special recognition for their commitment to security by taking on additional security duties in the way of training, communication or evangelism. There are a variety of reasons a developer might want to be recognized as a security champion: some may see it as an opportunity for career differentiation and professional development, while others might have a personal interest in software security. Regardless, this is a great way to scale the software security efforts beyond what the security team can manage alone.

Seeding each development team with a programmer who serves as a go-to security resource facilitates the information flow from security teams to development teams. These security champions help developers implement and improve secure development processes, and can escalate complicated or important issues back to security teams. These ongoing personal relationships help both the security and development team members take a long-term view that is less focused on the results of a particular assessment or the status of a specific vulnerability, and more focused on the ongoing process of securing the organization's software.

CONCLUSION

For a CISO to successfully facilitate Web application software security, developers cannot see security as a tax levied on them. Instead of being a roadblock to development team success, security teams instead need to make themselves a valuable resource for developers. Just as effective security professionals act as a risk management resource for managers and executives, they need to act as security enablers for development teams, helping them build secure Web applications with minimum hassle. Secure software does not happen overnight, but by taking a long-term view of the process, the security team can ensure a harmonious, team-focused effort, which ultimately will result in Web applications developed with far fewer security flaws. ■

DAN CORNELL *has more than 12 years of experience architecting, developing and securing Web-based software systems. As a principal of Denim Group, he leads the organization's technology team overseeing methodology development and project execution for Denim Group's customers. He also heads the Denim Group application security research team, investigating the application of secure coding and development techniques to the improvement of Web-based software development methodologies.*

USING WEB APPLICATION FIREWALLS FOR DEFENSE-IN-DEPTH

Improve your defense-in-depth security with Web application firewalls and a strong software development lifecycle process.

By Mike Chapple

WEB APPLICATIONS REMAIN one of the most vulnerable parts of the enterprise computing infrastructure. Organizations have taken extraordinary measures over the past decade to shore up network security as well as platform and hardware security in the data center, but Web applications have generally been left behind. The complexity of Web applications combined with a lack of security awareness among developers leads to a woeful state of vulnerability to SQL injection and cross-site scripting, even though these types of attacks have been happening for years. Estimates from application security vendor WhiteHat indicate that it would take the banking industry alone 400 days to patch 90% of the flaws in their applications.

This creates a compliance dilemma. The Payment Card Industry Data Security Standard (PCI DSS), for instance, requires regular application code scans to identify flaws that attackers could exploit. However, the time and work involved in this endeavor is, for most organizations, simply too great. One potential solution to this dilemma is the use of compensating controls to safeguard against existing and future application flaws. Web application firewalls (WAFs) fit this bill, and PCI DSS allows their use in lieu of regular application scans. These devices, like traditional network firewalls, monitor traffic coming into a network, with a specific emphasis on the requests headed to Web servers. They perform deep content inspection on these requests, looking for HTTP commands that violate an organization's security policy and/or include potentially malicious code, such as a SQL injection attack.

In this article, we'll explore not only the role Web application firewalls can play in securing a Web application infrastructure and meeting compliance demands, but also the secure software development processes that can shore up an organization's application security strategy.

MEETING PCI DSS REQUIREMENTS WITH WAFS

The most recent revisions to PCI DSS include specific language around Web application firewalls. PCI DSS requirement 6.6 mandates that organizations with public-facing Web applications address emerging threats with at least one of two approaches:

- Performing manual or automated application vulnerability assessments on an annual basis and after any changes.
- Installing a Web application firewall in front of public-facing Web applications.

This is an area where I'd recommend going above and beyond the requirement and implementing both of these security controls. While the standard may only require one, this is too important an area of risk to rely upon a single control.

That said, when it comes to certifying compliance, I'd suggest using the Web application firewall as the "official" response to section 6.6. Why? Because it's much easier for a company to demonstrate that it has a properly configured Web application firewall in place than to demonstrate that the existence of a rigorous program of Web application assessment. There's simply less subjectivity, making it easier for an assessor to check the "In place" box on your company's report on compliance.

Another important use of the Web application firewall is to provide "virtual patching" of known Web application vulnerabilities.

Another important use of the Web application firewall is to provide "virtual patching" of known Web application vulnerabilities. For example, an organization might use a Web application firewall to block requests that would exploit a known SQL injection bug in vendor-provided code. As the organization has no ability to create a patch for the vendor's application, a Web application firewall can obviate the need for the patch by blocking the suspect requests before they reach the application.

This strategy does have limitations, however. First, be sure that the WAF rules put in place are custom-tailored to each known vulnerability and are as broad as possible to block an attack without preventing critical business activity. This requires careful analysis by a security professional on a case-by-case basis, as every vulnerability is slightly different. Document the results of this analysis and provide it to PCI DSS auditors if they request it. Finally, keep in close contact with the vendor to determine when a patch is available, and apply it within the PCI DSS-mandated one-month window.

Also consider submitting WAF rules to your merchant bank as a formal compensating control for PCI DSS section 6.5 if a vendor patch is likely to be a long time in the making. This will go a long way toward satisfying auditors' stringent demands for documented compensating controls and limits liability if a compromise does occur.

NO SUBSTITUTE FOR GOOD DEVELOPMENT

It's important to keep in mind that, while Web application firewalls can help block inbound attacks to an organization before they reach the Web server, they are no substitute for good development practices. No information security control is 100% effective, and that includes Web application firewalls. Some dangerous requests will slip through the cracks and for that reason it is essential to develop robust code that is capable of standing up to an attack. Web application firewalls are simply another layer at an organization's disposal when building a defense-in-depth strategy for Web app security.

Developers can protect against vulnerabilities in their applications by integrating information security controls into each phase of their software development lifecycle (SDLC). The National Institute of Standards and Technology provides some great guidance to help developers with this. Here are some highlights worth mentioning specifically:

- **Initiation stage:** Developers should engage security experts and ensure that they have detailed the confidentiality, integrity and availability concerns alongside of the other business requirements for the application. This phase should also include a preliminary risk assessment.

- **Development stage:** Risk assessments should continue, and developers should select the security controls they will incorporate into their code and validate those controls during unit testing.

- **Implementation phase:** Security control integration and acceptance testing should take place.

- **Operational phase:** Throughout the time an application is in use, enterprises should incorporate change management, auditing, incident handling and intrusion detection procedures.

- **Sunset phase:** Organizations must consider the proper disposal of sensitive information.

Developers who embrace this approach and incorporate good development practices using a lifecycle approach are much more likely to build robust applications that stand up to common Web threats.

THE BOTTOM LINE

The bottom line is Web application firewalls play an important role in safeguarding today's Web applications against current and emerging threats. Combined with a strong SDLC process that incorporates risk assessment and testing, WAFs can serve as the cornerstone of a solid defense-in-depth approach to Web application security in the enterprise. ■

MIKE CHAPPLE, *Ph. D., CISA, CISSP, is an IT security manager with the University of Notre Dame. He previously served as an information security researcher with the National Security Agency and the U.S. Air Force. Chapple is a frequent contributor to SearchSecurity.com, and serves as its resident expert on enterprise compliance, frameworks and standards for its Ask the Experts panel. He is a technical editor for* Information Security *magazine and the author of several information security titles, including the* CISSP Study Guide *and* Information Security Illuminated.

AVOIDING COMMON WEB APPLICATION VULNERABILITIES

Learn about the most common Web application vulnerabilities and the countermeasures you need to implement to protect against them.

By Michael Cobb

THE OPEN WEB APPLICATION SECURITY PROJECT (OWASP) recently released this year's list of the Top 10 critical Web application security flaws. Sadly, the list is little changed from previous years, showing that those responsible for application design and development are still failing to address known and well-documented errors. Many of the most common Web app vulnerabilities are so widespread that crimeware kits feature search-and-exploit tools targeting them, making it trivial for even novice attackers to exploit these flaws.

In this article, we'll look at the top five common Web application vulnerabilities and provide guidance on how enterprises can fix the original problems and combat attacks that try to exploit them.

INJECTION VULNERABILITIES AND CROSS-SITE SCRIPTING

These are two of the most common and serious flaws that occur in Web applications. There are various forms of injection attacks, including SQL, operating system, email and LDAP injection, and they all work by sending malicious data to an application as part of a command or query. Carefully crafted data can trick an application into executing unintended commands or accessing unauthorized data. SQL injection occurs when attackers take advantage of sites that generate SQL queries using user-supplied data without first checking to make sure it is valid. This allows an attacker to submit malicious SQL queries and pass commands directly to a database. As an example of this kind of attack, it's thought that SQL injection was used to access Sony's PlayStation database and plant unauthorized code as part of the 2011 attacks on the PlayStation Network.

Validation functions need to clean any input of characters or strings that could possibly be used maliciously before passing it on to scripts and databases.

Cross-site scripting (XSS) attacks target an application's users by injecting code, usually a client-side script such as JavaScript, into a Web application's output. Whenever the compromised output or page is viewed, the browser executes the code, allowing an attacker to hijack user sessions, redirect the user to a malicious site or simply deface the page. XSS attacks are possible within the contents of a dynamically generated page whenever an application incorporates user-supplied data without properly validating or escaping it.

To prevent both injection and XSS attacks, an application should be configured to assume that all data, whether it's from a form, URL, cookie or even the application's database, has come from an untrusted source. Review every point where user-supplied data is handled and processed, and check to make sure it is validated. Validation functions need to clean any input of characters or strings that could possibly be used maliciously before passing it on to scripts and databases. Input must be checked for type, length, format and range. Developers should make use of existing security control libraries, such as OWASP's Enterprise Security API or Microsoft's Anti-Cross Site Scripting Library, instead of writing their own validation checks. Also, ensure that any values accepted from the client are checked, filtered and encoded before being passed back to the user.

BROKEN AUTHENTICATION AND SESSION MANAGEMENT

Web applications have to handle user authentication and establish sessions to keep track of each user's requests as HTTP does not provide this capability. Unless all authentication credentials and session identifiers are protected with encryption at all times and protected against disclosure from other flaws such as XSS, an attacker can hijack an active session and assume the identity of a user. In case an attacker discovers a session where the original user has failed to log out (walk-by attacks), all account management functions and transactions should require re-authentication, even if the user has a valid session ID. Two-factor authentication should also be considered for high-value transactions.

Areas that often require more attention include how session identifiers are handled and the methods used for changing a user's credentials.

To uncover authentication and session-management problems, enterprises can perform both code review and penetration tests. Developers can use automated code and vulnerability scanners to uncover potential security issues. Areas that often require more attention include how session identifiers are handled and the methods used for changing a user's credentials. If budgets don't stretch to cover commercial versions, there are plenty of open source and lite versions that highlight places where a closer inspection of code or processes is required.

This is another flaw that stems from poor application design based on the false assumption that users will always follow the application rules. For example, if a user's account ID is shown in the page URL or in a hidden field, a malicious user may be able to guess another user's ID and resubmit the request to access their data, particularly if the ID is a predictable value. The best ways to prevent this vulnerability are to use random, unpredictable IDs and file and object names, and to never expose the actual names of objects. Common places where this data is incorrectly exposed are URLs and links, hidden form fields, the unprotected view state in ASP.NET, drop-down list boxes, JavaScript code and client-side objects like Java applets. Each time sensitive files or content are accessed, verify the user is authorized to access them.

SECURITY MISCONFIGURATION

The infrastructure that supports a Web application comprises a complex variety of devices and software, including servers, firewalls, databases and OS and application software. All these elements need to be securely configured and maintained, with the application running with the least privileges necessary, yet many systems are never fully hardened. A primary cause of poor system administration is that those responsible for managing Web applications and the infrastructure that supports them have never undergone the necessary training.

Providing adequate training and resources for those tasked with day-to-day network and application management is as essential as making security and privacy a priority throughout all phases of the development process. Finally, schedule a penetration test for Web applications that handle sensitive data of any kind. This is a proactive method of assessing its ability to withstand an attack and finding vulnerabilities before a hacker does.

These five common Web application vulnerabilities have been a thorn in the side of IT security for years. They are not new and neither are the fixes for them, but until the security of Web apps is prioritized, attackers seeking to commit theft, fraud and cyberespionage will all continue to take advantage of these flaws. ■

MICHAEL COBB, *CISSP-ISSAP, is a renowned security author with more than 15 years of experience in the IT industry and another 16 years of experience in finance. He is founder and managing director of Cobweb Applications Ltd., a consultancy that helps companies secure their networks and websites, and also helps them achieve ISO 27001 certification. He co-authored the book* IIS Security *and has written numerous technical articles for leading IT publications. Michael is also a Microsoft Certified Database Administrator and a Microsoft Certified Professional.*

TechTarget

TechTarget Security Media Group

EDITORIAL DIRECTOR **Robert Richardson**

FEATURES EDITOR **Kathleen Richards**

SENIOR SITE EDITOR **Eric Parizo**

SENIOR MANAGING EDITOR **Kara Gattine**

ASSOCIATE MANAGING EDITOR **Rachel Shuster**

ASSOCIATE EDITOR **Brandan Blevins**

FEATURES WRITER **Sally Johnson**

DIRECTOR OF ONLINE DESIGN **Linda Koury**

COLUMNISTS **Marcus Ranum, Gary McGraw, Doug Jacobson, Julie A. Rursch, Matthew Todd**

CONTRIBUTING EDITORS **Michael Cobb, Scott Crawford, Peter Giannoulis, Ernest N. Hayden, Jennifer Jabbusch Minella, David Jacobs, Nick Lewis, Kevin McDonald, Sandra Kay Miller, Ed Moyle,Lisa Phifer, Ben Rothke, Anand Sastry, Dave Shackleford, Joel Snyder, Lenny Zeltser**

VICE PRESIDENT/GROUP PUBLISHER **Doug Olender**
dolender@techtarget.com

TechTarget
275 Grove Street,
Newton, MA 02466
www.techtarget.com

COVER IMAGE AND PAGE 3: LASSEDESIGNEN/FOTOLIA