Simplifying Application Security and Compliance with the OWASP Top 10

AN EXECUTIVE PERSPECTIVE

SecurityInnovation® 187 Ballardvale Street, Wilmington, MA 01887 978.694.1008 www.securityinnovation.com

Introduction

From a management perspective, application security is a difficult topic. Multiple parties within an organization are involved, as well as a varying collection of technologies intended to provide better security. As new threats and regulations create moving targets, it has become increasingly difficult to connect proposed remedies with specific results.

However, many leading enterprises have found an approach that cuts through much of this complexity. They are using the OWASP Top 10 list of critical security risks to focus their application security and compliance initiatives.

In this management briefing we will answer the following questions:

- Why is application security important?
- What is the OWASP Top 10?
- How can the OWASP Top 10 be used to transform application security?
- How can the OWASP Top 10 help with compliance?
- Is this approach cost-effective?
- What Tools are available to ensure best practices around the OWASP Top 10?

The concept:

build processes to prevent the ten most serious web-based attacks, and those processes will help you reduce many types of security risks, and at the same time cut development costs.

Why is Application Security Important?

Everyone acknowledges that IT security is important. Certainly the costs of failure are high: a recent survey found an average cost of $7.2 million per data breach event (or $214 per compromised customer record). The same survey found that 88% of the organizations surveyed had at least one major data breach in 2010.[1]

The problem is that, although most enterprises have invested in network and PC security, many have neglected to build adequate safeguards into their software applications.

Application security is rapidly being recognized as a top priority. Gartner has stated that: "Over 70% of security vulnerabilities exist at the application layer, not the network layer," and other researchers have estimated this figure at 90%.[2] State laws requiring the prompt disclosure of data breach problems are causing companies to look more closely at applications that process customer information. And industry standards bodies and government agencies are increasingly emphasizing application security, including the Payment Card Industry Security Standards Council and the U.S. National Institute of Standards and Technology (NIST).[3]

Organizations incorporating the OWASP Top 10 into security programs:

A.G. Edwards
British Telecom
Citibank
HP
IBM Global Services
Michigan State University
Price Waterhouse Coopers
REI
Samsung SDS (Korea)
Sprint
Symantec
The Hartford

What is the OWASP Top 10?

But what is the best way to address an issue that that affects every software developer and virtually every piece of software within an organization? That is where the OWASP Top 10 list has been helpful.

Since 2003, the Open Web Application Security Project (OWASP) has published a list of the ten most critical web application security risks.[4]

This list represents a consensus among many of the world's leading information security experts about the greatest risks, based on both the frequency of the attacks and the magnitude of their impact on businesses.

The objective of the OWASP Top 10 project is not only to raise awareness about ten specific risks, but also to educate business managers and technical personnel on how to assess and protect against a wide range of application vulnerabilities.

This use of the OWASP Top 10 has been embraced by many of the world's leading IT organizations, including those listed on this page.

The OWASP Top 10 has also become a key reference list for many standards bodies, including the PCI Security Standards Council, NIST and the FTC.

THE BOTTOM LINE:

Organizations that put in place the people, tools and processes to protect against the OWASP Top 10 risks will develop first-class application security programs capable of handling a wide range of web-based threats.

UNDERSTANDING THE SECURITY RISKS

The OWASP Top 10 risks are listed in the Appendix. Here we will give a quick overview of two of them.

The first risk on the list is "Injection." This means tricking an application into including unintended commands in the data sent to a database or another "interpreter." For example, a web form might ask for an account number. An attacker, instead of entering a legitimate account number, might enter something like this:
' OR 1=1 --

If the application sends these characters to a database, the database will collect a group of account numbers and send those back to the attacker. The consequences can be extremely serious: the attacker can get full access to hundreds of customer accounts.

Similar consequences can result from the eighth entry on the list, "Failure to Restrict URL Access." An attacker on an online shopping web site might notice that part of the address of his account page is */user/getAccounts*, and from that guess that there is another web page */manager/getAccounts* used by administrators to manage user accounts. Unless the */manager/getAccounts* page is properly protected, the attacker can use it to steal confidential customer data.

How Can the OWASP Top 10 Be Used to Transform Application Security?

The OWASP organization suggests that the OWASP Top 10 list can be used to "establish a strong foundation of training, standards and tools that makes secure coding possible."[5]

Enterprises who have implemented a successful application security program integrate the OWASP Top 10 into each stage of their software development lifecycle (SDLC) to design, develop and test new software applications. The diagram below demonstrates how this can be done.

Phase	OWASP Top 10 Use
Requirements and Analysis	**Threat Modeling:** use Top 10 as guide to potential attacks. Determine countermeasures.
Architecture and Design	**Security Design Guidelines:** Adopt design guidelines that will harden applications against Top 10.
Development	Adopt **coding standards** to counter Top 10. Search for Top 10 **code reviews**.
Testing	Develop **test plans** for Top 10. Test for Top 10 with **static analysis tools**. Scan for Top 10 with **web scanning tools**.
Deployment	Check for **configuration and physical deployment errors** related to Top 10.
Maintenance	Conduct **ongoing scanning** for Top 10.

1. REQUIREMENTS AND ANALYSIS

In the **Requirements and Analysis** phase, analysts consider the requirements and goals of the application, as well as possible problems and constraints. Part of this process involves threat modeling, which identifies threats and vulnerabilities relevant to the application.

The OWASP Top 10 can be used as guides to potential attacks. A thorough examination of which of those 10 risks could affect the software will suggest ways the application design can be shaped to achieve security objectives, and where resources could be applied to develop countermeasures.

2. ARCHITECTURE AND DESIGN

In the **Architecture and Design** phase, specific design guidelines can be adopted that are proven solutions to the Top 10 risks. For example, if the application is potentially susceptible to injection attacks specific guidelines can be adopted, such as always requiring centralized input validation that differentiates data (account numbers) from code (commands to the database).

3. DEVELOPMENT

In the **Development** phase, specific coding standards that have been proven to defend against the Top 10 risks can be adopted. To use our injection risk example again, developers could be required to have their software encode user-supplied input; that is, to tell the database "these characters come from a user screen, so they are definitely data and should never be executed as commands."

To address some of the "Failure to Restrict URL Access" issues, coding standards might require that every web page be protected by role-based permissions. For example, special logon screens for managers could be added to prevent attackers (and non-management employees) from accessing management screens.

Code reviews are another activity that typically occurs during the Development phase. Most developers review code only to make sure that it has the features and functions described in the specification. But developers trained to look also for vulnerabilities in the code related to the OWASP Top 10 will find many types of security issues.

4. TESTING

When the quality assurance group builds the test plan, it can ensure that specific tests are run to simulate attacks related to the Top 10 risks.

Static analysis tools which read through software code, can be programmed to look for clues in the code that the application may be vulnerable to Top 10 risks. Web scanning tools can be programmed to simulate attacks based on Top 10 vulnerabilities. For example, they could be set up to attempt injection attacks on all customer input screens.

5. DEPLOYMENT

Computer systems and software that are not configured with security in mind can open up systems to attacks. That is why the OWASP Top 10 can be very helpful in the **Deployment** phase of the software life cycle. For example, many problems can be prevented by ensuring that unnecessary utility software is shut off on servers, and that auditing and logging services are always turned on.

6. MAINTENANCE

Finally, in the **Maintenance** phase of the life cycle, a focus on the OWASP Top 10 will ensure that organizations conduct ongoing reviews and code scanning, to find out if changes to the application over time might have created any new vulnerabilities.

In short, integrating the OWASP Top 10 into every phase of the software development life cycle forces development organizations to adopt security best practices and learn how to use software testing tools. And these best practices and testing tools will help eliminate mitigate the risks, not just of the OWASP Top 10, but for many types of security risks.

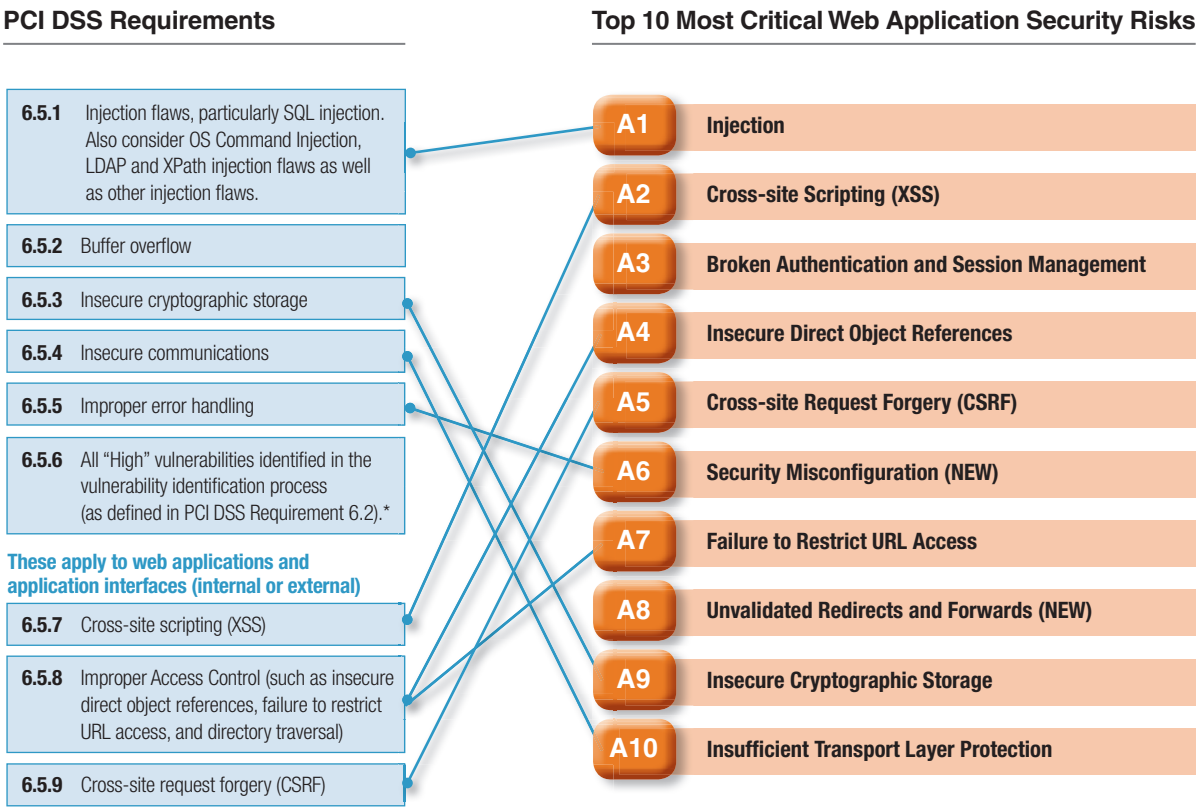
How Can the OWASP Top 10 Help with Compliance?

For some enterprises, addressing the OWASP Top 10 risks is mandatory for industry and regulatory compliance. For others it is optional, but provides an excellent way of demonstrating a high level of effort in addressing compliance issues.

PCI DSS

The PCI DSS rules specifically require addressing the Top 10. PCI DSS requirement 6.5 states: “Develop applications based on secure coding guidelines. Prevent common coding vulnerabilities in software development processes...as industry best practices for vulnerability management are updated (for example, the OWASP Guide, SANS CWE Top 25, CERT Secure Coding, etc.), the current best practices must be used for these requirements.”

In fact, the PCI DSS requirements 6.1 to 6.9 map directly to 8 of the OWASP Top 10, as shown in the diagram shown on the next page.



* This requirement is considered a best practice until June 30, 2012, after which it becomes a requirement.

OTHER STANDARDS

Most standards and regulations are not as explicit as PCI DSS in addressing the OWASP Top 10. However, several others do call for following best practices in the area of application security.

For example, the Department of Defense and Defense Information Systems Agency (DISA) recently published the 114-page Application Security and Development Security Technical Implementation Guide with detailed recommendations for creating a secure SDLC.[6] HIPAA requires that covered organizations perform risks analysis and risks assessments, and in some cases ensure that proper controls are in place for web applications. And a new ISO standard is under development: ISO 27034: Guidelines for Application Security.

Essentially, auditors are likely to view the failure to address the OWASP Top 10 as a sign that the organization is falling short of compliance with many standards, while integrating the Top 10 into the software development life cycle demonstrates that many best practices have been implemented as part of the security process.

Is this Approach Cost-effective?

At this point in our discussion some readers might say: "Why do we need such a new set of programs? Don't software developers already know how to implement application security?" But in fact, very few have been educated on secure coding practices. And even when they have been, emerging threats require refresher courses every year or two based on how attach methodologies continue to change. So educational programs built around the OWASP Top 10 provide essential education that most developers might not seek to acquire on their own.

Other readers might ask "Wouldn't it be cheaper to buy a few software testing tools and let them detect vulnerabilities in applications?" But software testing tools are almost useless unless developers learn how to use them and know where to point them. In fact, they can be worse than useless, because if not used properly they can generate large numbers of "false positives" that cause resources to be wasted hunting down non-existent bugs.

A third common misconception is that programs designed to improve application security can be focused only on software coding. Many security and compliance requirements are missed during the requirements and design phases of the life cycle, and many vulnerabilities are created during the deployment and maintenance phases.

JUSTIFICATION

Do application security programs have a return on investment?

Part of the answer obviously relates to preventing costly security breaches, and the emergence of advanced threats. As mentioned earlier, a recent survey found an average cost of $7.2 million per data breach, or $214 per compromised customer record, to cover expenses like customer notification, regulatory fines, and cleaning up the damage to internal systems. More than ever, enterprises must take into account the potential for serious damage to reputation and to customer relationships.

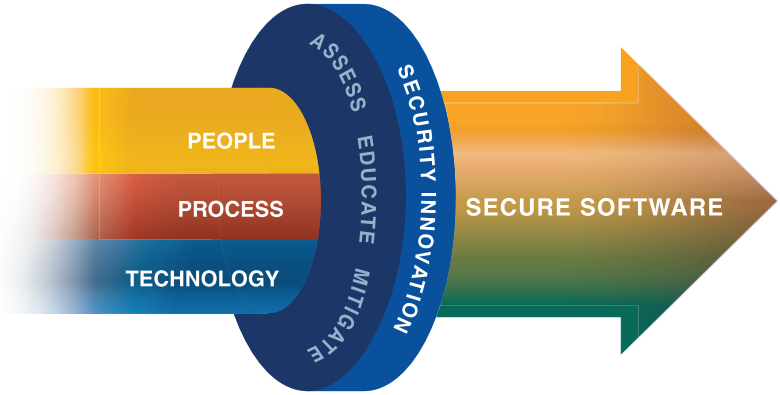
A second area is compliance. Compliance activities can be costly and time-consuming, and can take management attention away from more strategic projects. A well-documented application security program built around the OWASP Top 10 can streamline compliance processes and free up resources for more productive tasks.

Finally, a program that identifies application security issues early can save a tremendous amount of money over trying to identify and fix requirements in the later phases of the software development life cycle. Studies have calculated that preventing defects in the design phase requires one-tenth the effort of catching and fixing those defects at the system test phase. Gartner estimates that removing 50 percent of software vulnerabilities prior to applications being put into production can reduce configuration management and incident response costs by 75 percent.[7]

What Tools are available to ensure best practices around the OWASP Top 10?

As discussed in this paper, a program built around the OWASP Top 10 can provide a powerful foundation to effectively focus and organize an application security program. But implementing such a program successfully the first time requires an accumulation of knowledge and experience.

Security Innovation provides products, training and consulting services to help organizations build and deploy secure software, but also in implementing a best practices model based on the OWASP top 10.



These offerings include:

- Consulting services to assess application risk across the entire application portfolio, how to implement a secure software development life cycle, including SDLC assessment and optimization, code reviews, threat modeling and penetration testing.
- TeamProfessor eLearning, including courses like “OWASP Top Ten: Threats and Mitigations,” “How to Test for the OWASP Top Ten,” and many courses on secure coding practices for ASP.Net, Java, C++, Windows and other development environments.
- TeamMentor, the industry’s largest and only secure software development knowledgebase repository that provides intelligence at every stage of the development lifecycle, the perfect “In-Practice” reference guide for novice and advanced developers and designers, architects, project managers and security teams.

To Learn More:

For more information, please visit Security Innovation’s web site at **http://www.securityinnovation.com.**

To evaluate the company’s eLearning products, please contact us at **+ 1.877.694.1008 x1 or getsecure@securityinnovation.com.**

Appendix

OWASP Top 10 Application Security Risks —2010

A1	Injection	Injection flaws, such as SQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing unauthorized data.
A2	Cross-site Scripting (XSS)	XSS flaws occur whenever an application takes untrusted data and sends it to a web browser without proper validation and escaping. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.
A3	Broken Authentication and Session Management	Application functions related to authentication and session management are often not implemented correctly, allowing attackers to compromise passwords, keys, session tokens, or exploit other implementation flaws to assume other users' identities.
A4	Insecure Direct Object References	A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, or database key. Without an access control check or other protection, attackers can manipulate these references to access unauthorized data.
A5	Cross-site Request Forgery (CSRF)	A CSRF attack forces a logged-on victim's browser to send a forged HTTP request, including the victim's session cookie and any other automatically included authentication information, to a vulnerable web application. This allows the attacker to force the victim's browser to generate requests the vulnerable application thinks are legitimate requests from the victim.
A6	Security Misconfiguration	Good security requires having a secure configuration defined and deployed for the application, frameworks, application server, web server, database server, and platform. All these settings should be defined, implemented, and maintained as many are not shipped with secure defaults. This includes keeping all software up to date, including all code libraries used by the application.
A7	Insecure Cryptographic Storage	Many web applications do not properly protect sensitive data, such as credit cards, SSNs, and authentication credentials, with appropriate encryption or hashing. Attackers may steal or modify such weakly protected data to conduct identity theft, credit card fraud, or other crimes.
A8	Failure to Restrict URL Access	Many web applications check URL access rights before rendering protected links and buttons. However, applications need to perform similar access control checks each time these pages are accessed, or attackers will be able to forge URLs to access these hidden pages anyway.
A9	Insufficient Transport Layer Protection	Applications frequently fail to authenticate, encrypt, and protect the confidentiality and integrity of sensitive network traffic. When they do, they sometimes support weak algorithms, use expired or invalid certificates, or do not use them correctly
A10	Unvalidated Redirects and Forwards	Web applications frequently redirect and forward users to other pages and websites, and use untrusted data to determine the destination pages. Without proper validation, attackers can redirect victims to phishing or malware sites, or use forwards to access unauthorized pages.

Notes

[1] Ponemon Institute, *2010 Annual Study: U.S. Cost of a Data Breach*, March 2011. Press release: http://www.ponemon.org/blog/post/cost-of-a-data-breach-climbs-higher. Full study: http://bit.ly/hlgCne.

[2] Estimate from Gartner, quoted in Computerworld, February 25, 2005: http://www.computerworld.com/printthis/2005/0,4814,99981,00.html.

[3] See https://www.pcisecuritystandards.org/documents/pci_dss_v2.pdf; https://www.pcisecuritystandards.org/pdfs/summary_of_changes_highlights.pdf; NIST Special Publication 800-53A Rev 1:Guide for Assessing the Security Controls in Federal Information Systems and Organizations: http://csrc.nist.gov/publications/nistpubs/800-53A-rev1/sp800-53A-rev1-final.pdf.

[4] OWASP Top 10 project home: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project. OWASP Top 10 for 2010 release document: http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202010.pdf. OWASP Top 10 for 2010 slide presentation: http://owasptop10.googlecode.com/files/OWASP_Top_10_-_2010%20Presentation.pptx.

[5] OWASP Top 10 for 2010 release document: http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202010.pdf.

[6] Department of Defense/DISA Application Security and Development Security Technical Implementation Guide, Vol. 3, Release 2: http://iase.disa.mil/stigs/downloads/zip/u_application_security_and_development_stig_v3r2_20101029.zip.

[7] The cost of finding defects at different stages of the development life cycle were estimated by B. W. Boehm and P. N.Papaccio in: Understanding and Controlling Software Costs, IEEE Transactions on Software Engineering, Vol. 14, No. 10, p.1462-1477, October 1988, and by IDC and IBM Systems Sciences Institute, quoted in Microsoft Security Development Lifecycle: http://www.cert.uy/historico/pdf/MicrosoftSDL.pdf. The Gartner estimate is from: http://www.gartner.com/press_releases/asset_106327_11.html.