Building Web Applications

Security Recommendations for Developers

Web applications on Internet-facing systems allow access to everyone and operate 24 hours a day and 7 days a week, which leaves them open to attack. An attacker's goals include obtaining personally identifiable information (PII), financial information, sensitive documents, or even using the site for launching further attacks. Security breaches cost commercial and government entities millions of dollars a year. A recent study by Ponemon Institute revealed that the average cost of a cyber crime attack is around $8.9 million[1]. Of the mostly costly attacks described in the study web-based attacks were in the top 3 listed. This document provides recommendations for web developers on how to build and deploy secure web applications in order to counter this threat.

Integrating and prioritizing security best practices into all stages of web application development requires support from both developers and managers, but it minimizes the potential for hidden costs and missed deadlines. The following tips, tools and techniques will help an organization secure web applications from development through deployment of a new web application into a production network.

Understand the Vulnerabilities

- The Open Web Application Security Project (OWASP) has documented the ten most critical web application vulnerabilities. The OWASP Top 10[2] list vulnerabilities which exist in web applications and how to mitigate or eliminate the vulnerabilities.
- CWE/SANS Top 25 Most Dangerous Software Errors for 2011[3] contains common software errors which may compromise an application, such as SQL Injection (SQLi), cross-site scripting (XSS) or buffer overflows. CWE/SANS also documents how to mitigate or eliminate the vulnerabilities.

[1] http://www.ponemon.org/local/upload/file/2012_US_Cost_of_Cyber_Crime_Study_FINAL6%20.pdf
[2] https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
[3] http://cwe.mitre.org/top25/index.html

Use Secure Building Blocks

- When possible, employ a secure web application framework as they have been developed with security in mind and help minimize the occurrence of common web application vulnerabilities. Example frameworks are Django, GuardRails, and Apache Shiro.
- Ensure that a secure communication channel is established to pass data between the user and the web application. Transport Layer Security (TLS) should be employed by the web application anytime data (e.g. username, password or other sensitive information) is required to traverse the Internet or network.
- A web application's database may contain information, such as credit cards, usernames, or passwords, which is considered valuable to attackers. Web application databases should be securely configured and, when possible, reside on a separate system which logs all accesses.

Audit Web Application during Development and Post-Deployment

- Performing code audits during development enables vulnerabilities to be addressed prior to deployment, when it is most cost-effective. Both commercial (e.g. HP Fortify) and open-source (e.g. Sonar) tools are available to scan source code for errors. Audits should also be performed on a regular basis to minimize the possibility of bugs being introduced into the application later.
- Vulnerability scanners help identify potential issues with web applications and operating systems. Many vulnerability scanners are able to scan various operating systems for missing patches along with performing automated penetration tests against a web application for common security issues (e.g. XSS, SQLi, directory traversal attacks and others). Scanning regularly ensures the system stays updated and common security issues do not exist.

Scan User Input for Attacks

- Unchecked user input is the main vulnerability web applications face. A web application should verify every input field to ensure only valid input requests are accepted and parsed by the system.
- OWASP has developed security application programming interfaces (APIs) for various programming languages commonly used in web applications. The OWASP Enterprise Security APIs (ESAPIs)[4] are libraries that implement methods for processing user input safely.
- Microsoft developed an ASP.NET library called AntiXSS which implements methods for safely processing user input for ASP.NET. Microsoft also developed UrlScan, a security tool for restricting different types of HTTP requests bound for Internet Information Services (IIS). UrlScan can identify and block potentially dangerous HTTP requests, such as those used in SQLi attacks.
- Deploying a Web Application Firewall (WAF) will help to prevent against invalid input attacks and other attacks destined for the web application. WAFs are intrusion detection/prevention devices that inspect each web request made to and from the web application to determine if a request is malicious. Some WAFs install on the host system while others are dedicated devices that sit in front of the web application. Commonly used WAFs include both open source (e.g. ModSecurity) and commercial (e.g. Barracuda WebApplication, Trustwave WebDefender) offerings.

[4] https://www.owasp.org/index.php/Category:OWASP_Enterprise_Security_API

Harden the Environment

- A web application depends on the security of the environment in which it resides. The host system and any other devices which support the web application should be configured in a secure manner. Various security best practices and guidelines, such as NSA Security Configuration Guides[5] and DISA STIGs[6], exist for many commonly used operating systems, applications and network devices.

Monitor for System Integrity

- System integrity should be maintained throughout a WAF's lifecycle. The WAF and the WAF's environment must be monitored regularly in order to determine if an intrusion was attempted or might have occurred. Installing anti-virus software and enabling logging will help in maintaining a secure system state. Also, remote access should be disabled unless absolutely necessary as it provides an attack vector against the application.
- Host systems, firewalls and intrusion detection device logs should be reviewed daily in order to identify attacks, suspicious behavior or possible anomalies occurring within the network which may indicate a possible intrusion. In the event an attacker was successful, these logs will provide details into the attack and will be crucial in determining the extent of the attack's damage.
- Monitor important system and web application files to detect unauthorized modifications and to ensure the system remains in a secure state. Tripwire (Open Source), OSSEC (Open Source) and Cimtrak (Commercial) are commonly used tools for performing file integrity monitoring.

[5] http://www.nsa.gov/ia/mitigation_guidance/security_configuration_guides/index.shtml
[6] http://iase.disa.mil/stigs/a-z.htm