California
DEPARTMENT OF TECHNOLOGY
Office of Information Security

(916) 445-5239
HTTPS://CDT.CA.GOV/SECURITY/

SECURE SOFTWARE
INFORMATION SHEET NO. 3
MAY 2017 (REVISED)

WEB APPLICATION VULNERABILITIES: *MORE THAN A MERE NUISANCE*

Many organizations are moving to a greater Internet presence. Unfortunately, maintaining a presence in cyber space does not come without its challenges. A number of well-known attacks, particularly those originating from the Internet, target software that is known to include interfaces, protocols, design features, or development faults that are both well understood and widely publicized to harbor inherent weaknesses; such software includes Web applications (including browser and server components), Web services, database management systems, and operating systems. According to the SysAdmin, Audit, Network, Security (SANS) Institute, "all web frameworks (PHP, .NET, J2EE, Ruby on Rails, ColdFusion, Perl, etc) and all types of web applications are at risk from web application security defects, ranging from insufficient validation through application logic errors."

Attackers seek and leverage these exploitable defects and vulnerabilities, and many compromises are accomplished through a sequencing of exploits that target a combination of vulnerabilities in one or more components. Three of the most common risks cited in SANS most recent *Top 20 Internet Security Attack Targets* list are the PHP Remote File Include, the SQL Injection, and Cross-Site Scripting (XSS). Often viewed as merely a nuisance, these can and do provide attackers with the ability to accomplish other types of security compromises such as denial of service, unauthorized access, and escalation of privilege, consequently impacting availability, integrity, and confidentiality of information systems.

Recommended Mitigation Strategies for Agencies:

- Build security into and throughout the Software Development Lifecycle including:
  - Collecting and documenting security requirements during the analysis phase.
  - Rigorous and disciplined development principles and practices.
  - Security checkpoints (peer, design, and code reviews, security testing, validate all input, etc.).
  - Project management with a security emphasis (e.g., realistic timeline) to accomplish the latter.

    - Developer tools and resources in the performance of security reviews and tests.
    - Thorough documentation in order to formulate effective security test plans.
    - Removal or isolation of “dormant” code and/or software functions.
    - Eliminating the use of static and hardcoded passwords.
- Use Web scanning tools to help identify vulnerabilities (many provide specific remediation information).
- Inspect your Web application’s framework configuration and harden appropriately.
- Implement rigorous and disciplined change management practices.
- Always test and deploy patches and new versions of PHP as it is released.

Recommended Mitigation Strategies for Developers:

- Join secure coding organizations to boost skills and learn about secure coding problems and solutions, such as Open Web Application Security Project (OWASP) at http://www.owasp.org.

- Test your applications using the OWASP testing guide.

Additional Resources to assist state agencies in mitigating this issue:

SANS, *Top 20 Internet Security Attack Targets*
http://www.sans.org/top20/?portal=24d24c5e9a711368f477ca662a5ea643

NIST, *Guideline for Securing Public Web Servers* (the long and technical version)
http://csrc.nist.gov/publications/nistpubs/800-44-ver2/SP800-44v2.pdf

NIST, *Securing Web Servers* (an abbreviated outline by NIST – somewhat dated but still applicable) https://www.nist.gov/publications/securing-web-servers

U.S. Department of Homeland Security’s Software Assurance Program, *Build Security In*
https://buildsecurityin.us-cert.gov/daisy/bsi/home.html

Department of Defense, Defense Information Systems Agency’s *WebServer* – *Security Technical Implementation Guide*
http://www.dtic.mil/dtic/tr/fulltext/u2/a575447.pdf