Open Web Application Security Project (OWASP)

Response to Draft NIST Special Publication 800-118 Guide to Enterprise Password Management

Summary

The Open Web Application Security Project (OWASP) supports new and improved standards and guidance. We are pleased to contribute to the development of this special publication.

Applications, and especially web applications and web services, are increasingly being targeted to spread malware and to access sensitive data. Many web applications are available over untrusted networks or are by accessed by untrusted clients. Passwords are almost universally used in such software. Therefore, we believe that additional emphasis is required on aspects of enterprise password management relating to software applications. The security issues can be of a different nature to other types of password-protected information systems.

We welcome the references to other documents, but believe further detail and/or reference to more sources of information would be a great benefit to readers of the final document. This may be using additional footnotes or by the inclusion of a bibliography/further guidance section. In particular we would like to recommend three of OWASP's key projects—the Development Guide, the Testing Guide and the Application Security Verification Standard (ASVS)—containing detailed information on good development, testing and verification practices respectively. These include significant guidance on the use of passwords and related authentication, authorization and session management.

For all passwords, it is vitally important that the method of generating passwords and complexity rules should be appropriate for the level of risk, and should be kept secret. Any knowledge on complexity rules (such as minimum and maximum length, character sets, etc) or technologies used, reduce the effective number of options for brute force attacks and thus make the task of cracking passwords very much less costly, especially if this is combined with non-existent or poorly implemented lockout policies.

Our recommendations are:

- increase information in the document on application-related issues
- provide additional detail and references to assist readers
- password complexity requirements must be related to risk and should be kept secret

Our detailed point-by-point response follows.

Detailed Response

2 Introduction to Passwords and Password Management

Explanation Resources that require protection are sometimes available without authentication, or the password-based authentication mechanisms may not be implemented correctly. Reference: Authentication Verification Requirements, Application Security Verification Standard (ASVS) Web Application Edition, 2008, OWASP http://www.owasp.org/index.php/ASVS

Suggested changes At the end of the third paragraph add "It is important to verify that all files, web pages, etc protected by the password mechanism actually require authentication. The authentication controls across a web application should be verified using standards such as the Application Security Verification Standard (ASVS) Web Application Edition from OWASP."

3.1.1 Password Capturing : Storage

Explanation Use of the "autocomplete" attribute is recommended for HTML form elements containing sensitive data. "Remember me" functionality on public computers, where a user can simply return to their personalized account can be dangerous. Reference 1: Authentication - Browser Remembers Passwords, A Guide to Building Secure Web Applications and Web Services, v2.0.1, OWASP http://www.owasp.org/index.php/Category:OWASP_Guide_Project Reference 2: Authentication - Remember Me, A Guide to Building Secure Web Applications and Web Services, v2.0.1, OWASP http://www.owasp.org/index.php/Category:OWASP_Guide_Project

Suggested changes Add "For a web application, the 'autocomplete' attribute should be implemented with the value 'off' in rendered HTML form fields, or whole HTML forms, where sensitive data such as passwords are entered, but this should not be relied upon. Additionally, avoid the use of 'remember me' functionality on public systems, especially where more sensitive data is accessed or the application is considered to be a higher risk. If 'remember me' is implemented, do not turn it on by default, advise users of the risks before they opt in and never use a predictable 'pre-authenticated' token."

3.1.2 Password Capturing : Transmission

Explanation Web Services must not send passwords in plain text. Replay attacks are a significant type of vulnerability found in web applications. Reference 1: Web Services, A Guide to Building Secure Web Applications and Web Services, v2.0.1, OWASP http://www.owasp.org/index.php/Category:OWASP_Guide_Project Reference 2: Session Management - Session Token Replay, A Guide to Building Secure Web Applications and Web Services, v2.0.1, OWASP http://www.owasp.org/index.php/Category:OWASP_Guide_Project

Suggested changes In the bulleted list, change the end of the 3rd item from "Examples are switching from telnet to Secure Shell (SSH) and from HTTP to HTTP Secure (HTTPS)." to "Examples are using WS-Security (Web Services Security), switching from telnet to Secure Shell (SSH) and from HTTP to HTTP Secure (HTTPS).". Add "Privacy of data in transit is paramount and web applications should guard against replay attacks by careful design of session management - the addition of entropy, secret key and salt is also a good approach for transmission of passwords over a non encrypted tunnel. Message hashing (with salt), which includes a nonce, also helps

prevent replay. There should also be provision of a robust logout mechanism (which destroys the session server side e.g. deletes the session cookie in a typical web application), inclusion of a logout link or button on every page which requires authentication, and content anti-caching measures."

3.1.3 Password Capturing : User Knowledge and Behavior

Explanation Sharing of passwords and use of passwords revealed through social engineering are a particular threat to web applications. Reference 1: Phishing, A Guide to Building Secure Web Applications and Web Services, v2.0.1, OWASP http://www.owasp.org/index.php/Category:OWASP_Guide_Project Reference 2: Session Management - Session Token Replay, A Guide to Building Secure Web Applications and Web Services, v2.0.1, OWASP http://www.owasp.org/index.php/Category:OWASP_Guide_Project

Suggested change Add "For web applications, use session fixation controls to strongly tie a single browser to a single session and prevent multiple sessions by the same user. Users should be provided with information on previous successful and unsuccessful login attempts and activities undertaken to help them identify potential mis-use of their own account. If an event occurs, the account owner should be informed of the event via out-of-band means (see also 3.3.1)."

3.2.1 Password Guessing and Cracking : Guessing

Explanation Web applications can be particularly vulnerable to password guessing, where the attacker could have continual access to the target system. Reference 1: Authentication - Change Password, A Guide to Building Secure Web Applications and Web Services, v2.0.1, OWASP http://www.owasp.org/index.php/Category:OWASP_Guide_Project Reference 2: Authentication - Brute Force, A Guide to Building Secure Web Applications and Web Services, v2.0.1, OWASP http://www.owasp.org/index.php/Category:OWASP_Guide_Project Additionally, the provision or leakage of any knowledge on the password generation technology, method or complexity significantly reduces the effective number of options for brute force attacks.

Suggested changes In the first bullet point, add "On public systems such as web applications, a lower threshold of 5 or 10 failed attempts would be more common." Add "For web applications, the system also needs to guard against distributed guessing attacks, where different user accounts may be targeted and/or different sessions/hosts are used to launch the attack simultaneously. One example of this is reverse brute force where the same password is tried repeatedly but the attacker cycles through user account identities.'" Add "Password change mechanisms should require entry of the old password and enforce protection against guessing attacks in the same was as the login." Add "Password complexity requirements should be kept secret. Complexity requirements should be based on risk so that not all roles may not be subject to the same policies and, especially for higher-risk accounts such as those with administrative privileges, the policies should be revisited as a matter of regular re-evaluation. If there is any evidence of disclosure of such information, the processes for generating passwords should be completely changed."

3.3.1 Password Replacing : Forgotten Password Recovery and Resets

Explanation Password reset mechanisms vary in complexity, and are often

implemented less securely than login mechanisms. Reference: Authentication - Automated Password Resets, A Guide to Building Secure Web Applications and Web Services, v2.0.1, OWASP
http://www.owasp.org/index.php/Category:OWASP_Guide_Project

Suggested change Add "For publicly accessible systems such as web applications, password recovery should not be implemented. Instead password reset tied with some out-of-band communication should be used e.g. for a web application, out-of-band might be mobile phone SMS, or email, or conventional post."

3.4 Using Compromised Passwords

Explanation If we accept that some passwords will be compromised, it is important that accounts have the minimum privileges necessary. Thus, a read-only user shouldn't be able to create, update and delete data for example, or undertake administrative functions. Reference: Secure Coding Principles - Security Principles - Principle of Least Privilege, A Guide to Building Secure Web Applications and Web Services, v2.0.1, OWASP
http://www.owasp.org/index.php/Category:OWASP_Guide_Project

Suggested change Add "The account accessed, using any password, should have the least privilege to undertake the required functionality."

About OWASP

This response is submitted on behalf of the Open Web Application Security Project (OWASP) by the OWASP Global Industry Committee. OWASP is a worldwide free and open community focused on improving the security of application software. Our mission is to make application security "visible," so that people and organizations can make informed decisions about application security risks. Everyone is free to participate in OWASP and all of our materials are available under a free and open software license. The OWASP Foundation is a U.S. recognized 501(c)(3) not-for-profit charitable organization, that ensures the ongoing availability and support for our work at OWASP. Further information:

- OWASP Foundation
  http://www.owasp.org/index.php/OWASP_Foundation
- About The Open Web Application Security Project
  http://www.owasp.org/index.php/About_OWASP
- The Open Web Application Security Project
  http://www.owasp.org/
- OWASP Global Industry Committee
  http://www.owasp.org/index.php/Global_Industry_Committee
- OWASP Guide Project
  http://www.owasp.org/index.php/Category:OWASP_Guide_Project
- OWASP Testing Project
  http://www.owasp.org/index.php/Category:OWASP_Testing_Project
- OWASP Application Security Verification Standard Project
  http://www.owasp.org/index.php/ASVS