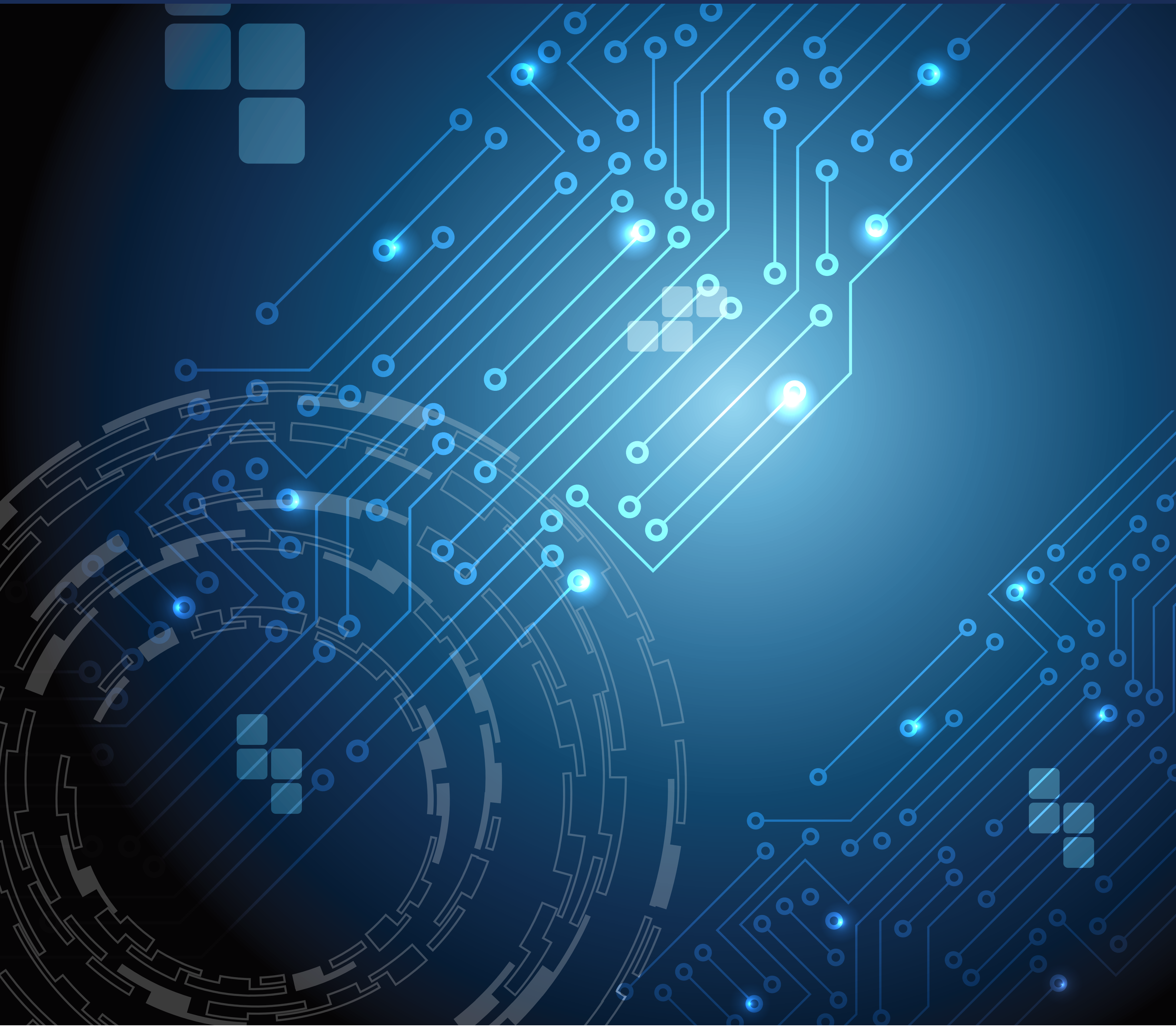
PLYNT

Plynt Certification



PALADION

BETTER SECURITY OUTCOMES

The awarding of the Plynt Certificate establishes that
a web application has adequate measures to guard against remote adver-
saries and protect against a wide range of threats, including the threats and vulnerabilities documented by organizations like OWASP, WASC and SANS.

This document defines the Plynt Certification Standard. It details the criteria that an application must meet in order to be awarded the Certificate.

This document is organized into two parts:

1. Part I lists and defines the certification criteria

2. Part II explains the logic and rationale of the criteria

Part 1:

Plynt Certification Criteria

The certification standard is composed of 14 criteria. These are organized into two sections:

- Section 1, "Security Protection Criteria" identifies the defenses an application and the hosting environment must demonstrate to meet the certification standard
- Section 2, "Security Requirements Criteria" specifies the features, characteristics and behavior an application must maintain to enhance its security and meet the certification standard

Section 1: Security Protection Criteria

1 Safe against popular attacks:

The application must demonstrate that it is not vulnerable to popular attacks.

*Note: A list of the popular attacks we test for is present under the '**Guide to the Plynt Certification Criteria**' section.*

2 Defend against a Threat Profile:

The application must demonstrate that it defends against the threats specified in a threat profile that has been developed specifically for the application.

Note I: A threat describes the goal of an adversary. According to the National Information Systems Security Glossary, a threat is any circumstance or event that has the potential to harm an information system through unauthorized access, destruction, disclosure, modification of data, and/or denial of service.

Note II: The threat profile is a list of all possible threats an application faces, but in this criterion we focus on threats related to the business and authorization rule violations, because other threats are covered by the rest of the criteria.

3 Protect sensitive data in transmission:

The application must take adequate measures to protect sensitive data from being stolen over the network.

4 Protect passwords from theft and guessing attacks:

The application must demonstrate that a remote adversary cannot steal or guess user passwords and passphrases. At a minimum, the application must enforce a minimum length of 8-character alphanumeric passwords.

Note I: The criterion requires that even after a user logs out, the user's password or pass phrase must be protected from theft.

Note II: The criterion recognizes that there are social engineering methods that could be used to steal the password or passphrase without access to the application. These are not within the scope of this criterion.

Note III: The risk of password guessing varies depending on the predictability of the usernames used by the application.

5 Secure password recovery implementation:

If the application provides a password recovery feature like 'forgot password', then it must protect against an adversary from recovering or resetting the password of legitimate users. In addition, the application should ensure that issuance of a new password is initiated after an alternate authentication mechanism. The application should implement a secure mechanism for issuing a new password that should be short-lived and of one-time use only.

Note I: An alternate authentication mechanism may include implementations like security questions, or one-time passwords sent to a user-owned device like mobile phone or other token devices.

Note II: A secure mechanism for password issuance entails the use of tokenized links, temporary passwords, and SSL communication while resetting the password.

Note III: By password we mean any secret that is used during an authentication process, for example, passphrase, PIN, passcode and passkey, etc.

6 Secure the hosting server directly accessible by the users:

All services that a user can reach must be securely configured and patched to prevent leakage of sensitive information, unauthorized access, buffer overflow, remote code execution, and other publicly known exploits.

Note I: This criterion includes securing all directories, backup, log and configuration files. Configuration files include OS, web server and application configuration files.

Note II: Along with the HTTP(s) service on which the application is accessible, this criterion mandates protection of all the non-HTTP(s) services like SSH, RDP, FTP, etc., against attacks that would lead to server compromise, which would eventually affect the web application.

7 Secure application's interaction with web client components:

The application should ensure authentication and integrity checks for interactions with web client components.

Note I: The security of interactions specified in this criterion includes cross-domain origin checks and session management checks for web client components like Javascript, Applets, Flash, HTML5, and Web sockets.

Section 2: Security Requirements Criteria

8 Sensitive data not stored on the client machine:

The application must not store sensitive data on the client machine in easily accessible locations.

Note I: Easily accessible locations include the browser cache, the browser history, web storage, other browser menus and persistent cookies on the client machine.

Note II: The browser memory is not considered an easily accessible location for this criterion.

9 Sensitive data not hidden in pages:

The application must not hide sensitive data in html comments or hidden form fields embedded in the pages.

10 No sensitive data in error messages:

The application must not reveal sensitive information in error messages.

Note: Sensitive information includes not only business-sensitive information but also details regarding application architecture that could aid an adversary who is attempting to launch an attack against the application.

11 Code obfuscation and cryptography for secrets:

If web client technologies like Javascripts, Applets or ActiveX controls contain secrets, they must use strong code obfuscation and cryptography techniques to protect the secrets.

Note: The secrets in the above criterion refer to cryptographic keys, passwords and algorithms that are considered trade secrets.

12 Re-authentication required for sensitive activities:

The application must re-authenticate the user before allowing the user to perform an operation involving sensitive data. A few examples of operations involving sensitive data are “Change Password”, “Transfer Funds” and “Transaction Approval”.

Note: This criterion does not apply to the special case where the user has forgotten the password and resets the password using the application’s password recovery or “Forgot Password” feature.

13 No sensitive data in requests to external entities:

If the application maintains links to external entities i.e. sites, web services and web sockets, it must not disclose any sensitive data other than that required to conduct the operation with the external entity.

Note: Sensitive data, as used in this criterion, include personal identity, personal health and payment card information, as well as session token(s) and authentication token(s).

14 No sample or test applications:

The server must not make available any sample or test application to remote users.

Part 2:

Guide to the Plynt Certification Criteria

What's the logic behind the Plynt Certification Criteria?

The certification criteria specify the standards that an application must meet to establish that it has adequate security measures. An application must prove that it resists attacks as well as demonstrate the implementation of security features that enhance its security. Accordingly, the certification criteria reflect these two aspects. The "Security Protection Criteria", detailed above in Section I, defines the threats that the application must show itself to be protected against. Section II, "Security Requirements Criteria", specifies the features and characteristics the application must maintain to enhance its security.

Plynt standard concerns itself only with High and Medium risk vulnerabilities of the application that violate the Plynt criteria. Even though low risk vulnerabilities are not considered while evaluating an application's security against Plynt certification standard, they are identified during the test and reported.

My application faces threats that you don't specify in the criteria. How will you address those?

It's quite likely that your application faces threats that are specific to its business context. For instance, a banking application needs to protect against the threat of funds being siphoned off and a gaming application needs to defend against the rules of the game being violated. The certification criteria require the application to protect against these threats in Criterion2, above, "Defend against Threat Profile". The threat profile is a customized listing of the threats relevant to that specific application. The threat profile is developed by the Plynt team with inputs from the application owner. The security engineers then develop test cases specific to the threat profile to verify that the application is safe against those threats.

What do you mean by popular attacks and why don't the criteria talk about SQL Injection, Cross-Site Scripting,etc.?

When writing the criteria, the authors first considered listing each individual attack that would be tested. However, since the criteria are revised with the discovery of every new attack, and there was no agreed standard taxonomy of attacks, it is reasonable to categorize all of the popular attacks under Criterion 1. Criterion 1, "Safe against popular attacks", catches the attacks like those listed in this question. Some of the popular attacks the application must defend against are listed here:

Attack Type	Coverage		
	OWASP Top 10	WASC Threat Classification	CWE/SANS Top 25
Injection Attacks	• A1-Injection	• WASC-29 LDAP Injection • WASC-30 Mail Command Injection • WASC-28 Null Byte Injection • WASC-31 OS Commanding • WASC-36 SSI Injection • WASC-19 SQL Injection • WASC-39 XPath Injection • WASC-23 XML Injection • WASC-46 XQuery Injection • WASC-25 HTTP Response Splitting • WASC-05 Remote File Inclusion (RFI) • WASC-43 XML External Entities • WASC-07 Buffer Overflow • WASC-06 Format String • WASC-03 Integer Overflows • WASC-12 Content Spoofing • WASC-20 Improper Input Handling	• CWE-89 Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') • CWE-676 Use of Potentially Dangerous Function • CWE-78 Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection') • CWE-829 Inclusion of Functionality from Untrusted Control Sphere • CWE-131 Incorrect Calculation of Buffer Size • CWE-120 Buffer Copy without Checking Size of Input ('Classic Buffer Overflow') • CWE-134 Uncontrolled Format String • CWE-190 Integer Overflow or Wraparound
Cross-Site Scripting Attacks	• A3-Cross-Site Scripting (XSS)	• WASC-8 Cross-Site Scripting • WASC-22 Improper Output Handling • WASC-12 Content Spoofing	• CWE-79 Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
Cross-Site Request Forgery Attack (CSRF)	• A8-Cross-Site Request Forgery (CSRF)	• WASC-9 Cross-Site Request Forgery	• CWE-352 Cross-Site Request Forgery (CSRF)
Denial-of-Service attacks	NA	• WASC-10 Denial of Service • WASC-21 Insufficient Anti-automation • WASC-41 XML Attribute Blow up • WASC-44 XML Entity Expansion • WASC-35 SOAP Array Abuse	NA
Path traversal attacks	• A7-Missing Function Level Access Control	• WASC-33 Path Traversal	• CWE-22 Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
Phishing attacks	• A10-Unvalidated Redirects and Forwards	• WASC-38 URL Redirector Abuse	• CWE-601 URL Redirection to Untrusted Site ('Open Redirect')
Request/ Response smuggling attacks	NA	• WASC-26 HTTP Request Smuggling • WASC-27 HTTP Response Smuggling	NA
Attacks targeting weak authentication mechanism	• A2-Broken Authentication and Session Management	• WASC-01 Insufficient Authentication	• CWE-306 Missing Authentication for Critical Function • CWE-807 Reliance on Untrusted Inputs in a Security Decision
Attacks targeting insecure session management	• A2-Broken Authentication and Session Management	• WASC-18 Credential/Session Prediction • WASC-37 Session Fixation • WASC-47 Insufficient Session Expiration	• CWE-862 Missing Authorization • CWE-863 Incorrect Authorization
WS MITM attack	NA	• WASC-32 Routing Detour	NA

You missed criterion xyz!

Thank you for pointing out. We are constantly looking at ways to improve the criteria – so your suggestions will help.

What do you mean by sensitive data in the criteria?

Sensitive data are any data that if compromised could lead to the following:

- Loss to the business's finance or reputation
- Invasion of the privacy of individuals
- Adversaries gaining an advantage to launch attacks

Sensitive data include business-sensitive information, authentication credentials, session token(s), authentication token(s) and any details regarding application architecture that could aid an adversary in launching a successful attack against the application.

The criteria says that, "The criterion requires that even after a user logs out, the user's password, pass phrase and PIN must be protected from theft". Please explain this.

In general, the application must safeguard passwords, passphrases and PINs. The note mentioned in Criteria 4 in Section 1, highlights that the password, passphrase or PIN should be protected even when the user has logged out. Vulnerabilities in some authentication schemes enable the password, passphrase or PIN to be stolen if a user who has logged out leaves the browser window open. This is described in the Paladion paper "Stealing Passwords via Browser Refresh".

An application must re-authenticate the user after log out – Isn't this what all applications do?

Experience shows that not all applications enforce re-authentication after logout. To make matters worse, some platforms like .Net do not invalidate a user session immediately when their signout method is called. To learn more, please read the paper "ASP.Net Forms Authentication" from Foundstone. As the note points out, if an application does not invalidate a session immediately, it should protect against the session being reused. Also, this Microsoft support document (http://support.microsoft.com/default.aspx?scid=kb;en-us;900111) has examples of such protection.

How does Plynt criteria map to OWASP Top 10, WASC threat classification and SANS Top 25?

Plynt criteria covers almost all the threats and vulnerabilities documented in OWASP Top 10, WASC and SANS 25, except some of the categories marked with asterisk (*), because certain tests covered in these categories cannot be performed during a Grey box test. By Grey box test we mean "dynamic tests that are performed using login credentials shared by the customer".

Plynt Criteria	Mapping		
	OWASP Top 10	WASC Threat Classification	CWE/SANS Top 25
Safe against popular attacks	• A1-Injection • A3-Cross-Site Scripting (XSS) • A8-Cross-Site Request Forgery (CSRF) • A7-Missing Function Level Access Control • A10-Unvalidated Redirects and Forwards • A2-Broken Authentication and Session Management	• WASC-29 LDAP Injection • WASC-30 Mail Command Injection • WASC-28 Null Byte Injection • WASC-31 OS Commanding • WASC-36 SSI Injection • WASC-19 SQL Injection • WASC-39 XPath Injection • WASC-23 XML Injection • WASC-46 XQuery Injection • WASC-25 HTTP Response Splitting • WASC-05 Remote File Inclusion (RFI) • WASC-43 XML External Entities • WASC-07 Buffer Overflow • WASC-06 Format String • WASC-03 Integer Overflows • WASC-12 Content Spoofing • WASC-20 Improper Input Handling • WASC-8 Cross-Site Scripting • WASC-22 Improper Output Handling • WASC-12 Content Spoofing • WASC-9 Cross-Site Request Forgery • WASC-10 Denial of Service • WASC-21 Insufficient Anti-automation • WASC-41 XML Attribute Blowup • WASC-44 XML Entity Expansion • WASC-35 SOAP Array Abuse • WASC-33 Path Traversal • WASC-38 URL Redirector Abuse • WASC-26 HTTP Request Smuggling • WASC-27 HTTP Response Smuggling • WASC-01 Insufficient Authentication • WASC-18 Credential/Session Prediction • WASC-37 Session Fixation • WASC-47 Insufficient Session Expiration • WASC-32 Routing Detour	• CWE-89 Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') • CWE-676 Use of Potentially Dangerous Function • CWE-78 Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection') • CWE-829 Inclusion of Functionality from Untrusted Control Sphere • CWE-131 Incorrect Calculation of Buffer Size • CWE-120 Buffer Copy without Checking Size of Input ('Classic Buffer Overflow') • CWE-134 Uncontrolled Format String • CWE-190 Integer Overflow or Wraparound • CWE-79 Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') • CWE-352 Cross-Site Request Forgery (CSRF) • CWE-22 Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal') • CWE-601 URL Redirection to Untrusted Site ('Open Redirect') • CWE-306 Missing Authentication for Critical Function • CWE-807 Reliance on Untrusted Inputs in a Security Decision
Defend against Threat Profile	• A7-Missing Function Level Access Control • A4-Insecure Direct Object References	• WASC-40Insufficient Process Validation • WASC-42 Abuse of Functionality • WASC-02 Insufficient Authorization	• CWE-862 Missing Authorization • CWE-434 Unrestricted Upload of File with Dangerous Type • CWE-863 Incorrect Authorization • CWE-306 Missing Authentication for Critical Function • CWE-250 Execution with Unnecessary Privileges*
Protect sensitive data in transmission	• A6-Sensitive Data Exposure	• WASC-04 Insufficient Transport Layer Protection	• CWE-311 Missing Encryption of Sensitive Data* • CWE-327 Use of a Broken or Risky Cryptographic Algorithm* • CWE-759 Use of a One-Way Hash without a Salt*
Protect passwords from theft and guessing attacks	• A6-Sensitive Data Exposure	• WASC-11 Brute Force	• CWE-307 Improper Restriction of Excessive Authentication Attempts
Secure Password Recovery Implementation	• A2-Broken Authentication and Session Management	• WASC-42 Abuse of Functionality • WASC-49 Insufficient Password Recovery	• CWE-306 Missing Authentication for Critical Function

Plynt Criteria	Mapping		
	OWASP Top 10	WASC Threat Classification	CWE/SANS Top 25
Secure the hosting server directly accessible by the users	• A5-Security Misconfiguration • A9-Using Components with Known Vulnerabilities	• WASC-34 Predictable Resource Location • WASC-16 Directory Indexing • WASC-14 Server Misconfiguration • WASC-17 Improper File system Permissions • WASC -15 Application Misconfiguration • WASC-13 Information Leakage	• CWE-732 Incorrect Permission Assignment for Critical Resource*
Secure application's interaction with web client components	• A2-Broken Authentication and Session Management • A5-Security Misconfiguration	• WASC-15 Insecure Indexing • WASC-14 Server Misconfiguration	• CWE-829 Inclusion of Functionality from Untrusted Control Sphere • CWE-863 Incorrect Authorization
Sensitive data not stored on client	• A6-Sensitive Data Exposure	• WASC-13 Information Leakage	• CWE-311 Missing Encryption of Sensitive Data* • CWE-327 Use of a Broken or Risky Cryptographic Algorithm* • CWE-759 Use of a One-Way Hash without a Salt*
Sensitive data not hidden in pages	• A6-Sensitive Data Exposure	• WASC-13 Information Leakage	NA
No sensitive data in error messages	• A6-Sensitive Data Exposure	• WASC-13 Information Leakage	NA
Code obfuscation and cryptography for secrets	• A6-Sensitive Data Exposure	• WASC-13 Information Leakage	• CWE-798 Use of Hard-coded Credentials*
Re-authentication required for sensitive activities	• A7-Missing Function Level Access Control	• WASC-40Insufficient Process Validation	• CWE-306 Missing Authentication for Critical Function
No sensitive data in requests to external entities	• A6-Sensitive Data Exposure	NA	NA
No sample or test applications	• A5-Security Misconfiguration	• WASC-34 Predictable Resource Location • WASC-14 Server Misconfiguration	NA

Note I: A single threat or vulnerability classification of OWASP, WASC and SANS has been mapped to multiple Plynt Criteria wherever needed.

You don’t mention SSL anywhere in the criteria!

The Criteria doesn’t mention SSL or any specific encryption technology. The criterion “Protect sensitive data in transmission” requires that the application take adequate protection to safeguard sensitive data in transit. SSL is a good example of such protection. The criteria, however, does not demand SSL. The application is free to choose the technology, as long as it demonstrates that it protects sensitive data during transmission.

Please explain, “No sensitive data in requests to external entities”.

Links to external entities i.e. sites, web services and web sockets are often a source of data leakage. For example, in a request to the external site along with the URL, the HTTP request also includes a referrer field that contains the URL of the page from which the request originated. If that URL (and the referrer field in turn) contains sensitive data, then that data will get logged in the web server logs of the external site. Administrators of the external site, or anyone who might gain administrator access, have access to those logs and can collect sensitive information.

“Why only these criteria? Does this cover everything?

The Plynt criteria focus on the high-impact threats popular today. The certificate assures that an application is safe against those threats. The authors chose to omit criteria that are not essential or do not have a material impact on security. For example, the certificate does not require that the application catch all error messages. While that’s a good practice to follow, it would have been unduly stringent for the certificate. The standard only demands that no sensitive data be revealed in error messages, because it has direct impact on the application’s security.

Why a new version of the criteria?

New technologies, new threats and new attacks are continually being created. The Plynt team periodically reviews the Plynt certification criteria to ensure the criteria address all known current threats. Details of the changes made to the Plynt Certification Criteria version 4.0 are listed below.

The following Criteria have been merged:

With an objective of making the Plynt criteria more precise yet effective, certain criteria have been merged as listed below:

Version 3.0 Criterion 4 – "Safeguard passwords" & Criterion 5 – "Protect against password guessing" have been merged into Version 4.0 Criterion 4 –**“Protect passwords from theft and guessing attacks”,** because the end objective of both the criteria was to ensure that application passwords are secured.

Version 3.0 Criterion 7 – "Insecure Configuration Settings on servers accessible directly by users" & 14 – "Web server protected against known vulnerabilities" have been merged into Version 4.0 Criterion 6 – **“Secure the hosting server directly accessible by the users”,** because the end objective of both the criteria was to ensure that the server that hosts the web application and is directly accessible by the users is hardened, i.e., the latest patches are installed and it is securely configured.

While it’s imperative that every remotely accessible service is securely implemented, the Plynt certification concerns itself with web-server-level vulnerabilities and just the vulnerabilities of other services (for example, SSH, RDP & SMTP, etc.) that can lead to server compromise and impact the hosted web application.

Version 3.0 Criterion 11 – "Code obfuscation for secrets" & Criterion 16 – "No sensitive data in source code" have been merged into Version 4.0 Criterion 11 – **“Code obfuscation and cryptography for secrets”,** because the end objective of both the criteria was to ensure that any source code that is accessible by a user does not disclose any sensitive data.

The following Criteria have been modified

Version 3.0 Criterion 1 – "Safe against popular attacks" – a list of popular attacks has been updated and a reference to this list has been provided.

Version 3.0 Criterion 2 – "Defend against Threat Profile" – notes have been modified to provide better clarity of what is covered under this criterion and what is the focus area.

Version 3.0 Criterion 6 –"Secure Forgot Password Implementation" – has been modified to Version 4.0 Criterion 5 – "Secure password recovery implementation". There could be multiple ways in which a user can recover the password, which may not necessarily be termed as the "Forgot Password" feature. Also, a description of the criteria has been updated to provide more clarity on what are covered in these criteria.

Version 3.0 Criterion 11 – "Code obfuscation for secrets" – has been modified to Version 4.0 Criterion 11 –"Code obfuscation and cryptography for secrets", because the secrets in client-side code can be protected via code obfuscation and cryptography techniques.

Version 3.0 Criterion 11 – "No sensitive data in requests to external sites" – has been modified to Version 4.0 Criterion 13 – "No sensitive data in requests to external entities". In today's service-oriented and distributed application environment, applications do not interact only with external sites; instead, they could also interact with external web services or web sockets.

The following Criteria has been added

Version 4.0 Criterion 7 – "Secure application's interaction with web client components" – for covering the threats and attack vectors that exploit the cross-origin interaction settings of the application.