PLYNT

Plynt Certification Criteria

Version 1.0

Effective Date: February 1, 2006

The Plynt Certificate establishes that a web application has adequate measures to guard against remote adversaries and protect against a wide range of threats. This Plynt Certification Standard document defines the criteria used to evaluate an application for the Certificate. An application must demonstrate through testing that these security criteria are met before it is awarded the Plynt Certificate. The application is tested remotely to verify that it meets the Plynt criteria.

This document is organized in two parts:

I. Part I defines the certification criteria, and

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

Part I: Plynt Certification Criteria

The certification standard is composed of 25 criteria. These are organized in two sections:

1. Section 1, "*Security Protection Criteria*" identifies the defenses an application must demonstrate to get the Plynt Certificate.

2. Section 2, "*Security Requirements Criteria*", specifies the features and behavior an application must have to get the Plynt Certificate.

Section 1: Security Protection Criteria

1 **Safe against popular attacks:** The application must demonstrate through testing that it is not vulnerable to popular attacks.

Note: "Popular attacks" include but are not limited to exploits documented in web sites like www.owasp.org, www.webappsec.org, www.osstmm.org, etc.

2 **Defend against threat profile:** The application must demonstrate through testing that it defends against the threats specified in its threat profile.

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 with 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 to an application. These threats include violating the business rules and authorization rules of the application.

3 **Protect sensitive data in transmission:** The application must take adequate measures to protect sensitive data from being stolen over the network.

4 **Safeguard passwords:** The application must demonstrate through testing that a remote adversary cannot steal user passwords from the application.

Note I: The criterion requires that even after a user logs out, it must not be possible to steal the user's password.

Note II: The criterion recognizes that there are social engineering methods that could be used to steal passwords outside the application. They are not within the scope of this criterion.

5 **Protect against automated password guessing:** If the application uses passwords for authentication, it must protect against brute force password guessing attacks.

6 **Protect against manual password guessing:** If the password used by the application has less than 10,000 possible values, the application must protect against manual password guessing attacks.

Note: A 4-digit numeric PIN is an example of a password with less than 10,000 possible values.

7 **Protect secret questions from guessing attacks:** If the application provides a password recovery or 'forgot password' feature with secret question(s), it must protect against an adversary guessing the answer to the secret question(s).

8 **Protect configuration files and directory lists:** The application must not reveal configuration files and directory listings to remote users.

Note I: Configuration files, as used in this criterion, include OS, web server and application configuration files.

Note II: Directory listings, as used in this criterion, refer to a listing of all files and directories in a folder, regardless of whether there are links to those objects from the application or not. While an adversary can compile a list of links in the application by studying the html pages, such compilations are not considered directory listings.

Section 2: Security Requirements Criteria

9 **Sensitive data not stored on client:** 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 and persistent cookies on the client machine.

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

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

11 **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 launch a successful attack against the application.

12 **Known, strong cryptographic algorithms:** Any cryptographic algorithm used on the client to protect sensitive data must be a publicly known algorithm that is secure for the intended use.

13 **Code obfuscation for secrets:** If Javascripts, Applets or ActiveX controls contain secrets, they must use strong code obfuscation techniques to protect the secrets.

Note: The secrets the above criterion refers to include cryptographic keys, passwords, and algorithms considered a trade secret.

14 **Session timed out after period of inactivity:** The user session must be timed out after an appropriately defined period of inactivity.

Note I: The criterion requires that the application define and enforce a value for the minimum period of inactivity. It does not specify the minimum value.

Note ii: The criterion requires that the inactivity period chosen to timeout a user be adequate to address the threat of an inactive session being hijacked by an adversary. The criterion recognizes that the minimum period could be different for different applications, based on their pattern of usage.

15 **Re-authentication required after log out:** Once the user has logged out, the application must require authentication before granting access to private pages.

Note: The criterion requires the application to invalidate the authenticated session at the server when the user logs out. If the application does not invalidate a user session at the server immediately on log out, it must take adequate protection against the session being reused.

16 **Warning required for "Remember Me":** If the application provides a "Remember Me" feature, it must warn the user against enabling while using shared computers to access the application.

17 **Password not stored in plain text for "Remember Me":** If the application provides a "Remember Me" feature, it must not store user passwords on the client machine in plain text or in a form that can be decrypted by an adversary.

18 **Old password required before changing password:** The application must re-authenticate the user before allowing the user to change the password.

Note: This criterion does not apply for the special case where the user has forgotten the password and resets the password using the application's password recovery or "Forgot Password" feature.

19 **Random session token:** Session token(s) must be random and difficult to predict.

Note I: "Session token(s)", as used in this criterion, is the set of tokens that the application uses to track the state of the user session, regardless of whether the token is implemented as a HTTP Cookie, a URL query-string variable, a Hidden form field value, or a combination of any of these.

20 **New authentication token on log in:** The application must assign new, random authentication token(s) to a user when the user logs in.

Note I: "Authentication token(s)", for the purpose of this criterion, is the set of tokens the application uses to track the authentication status of a session and the identity of the user of that session.

Note II: If the application uses session token(s) to play the role of the authentication token, this criterion requires that the session token take a new, random value when a user logs in.

21 **No sensitive data in requests to external sites:** If the application links to external sites, it must not disclose to the external site any more sensitive data than is required by the external site.

Note: Sensitive data, as used in this criterion, includes business sensitive information, as well as session token(s) and authentication token(s).

22 **Services patched:** The services exposed by the server must not be vulnerable to publicly known, remotely exploitable bugs.

Note: "Exposed", as used in this criterion, refers to services accessible remotely.

23 **Access to server filtered:** The server must be protected by a filter that allows access only to ports required for the use and administration of the server.

Note: The criterion does not specify the ports that must be blocked or allowed; the specific ports may vary with applications. The criterion however requires that only those required by remote users and administrators be allowed by the filter.

24 **No sample or test applications:** The server must not make available any sample or test application to remote users.

25 **No sensitive data in source code:** The application must not disclose sensitive data in any source code that is accessible to remote users.

Part II: Guide to the Certification Criteria

What's the logic behind the Plynt Certification Criteria?
The certification criteria specifies the standards an application must meet to establish that it has adequate security measures. An application must resist attacks on it as well as implement security features that enhance its security. Accordingly, the certification criteria includes two sets: the first, "Security Protection Criteria" in Section I defines threats the application must protect against. In Section II, "Security Requirements Criteria" specifies the features and characteristics of the application that enhance its security.

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 criteria require the application to protect against these threats in the second criterion: defend against threat profile. The threat profile is the listing of all threats relevant to the application. Before the certificate is issued, the application must demonstrate through testing that it defends against these threats. The threat profile is developed by the Plynt team with inputs from the application owner; the security engineers then develop test cases to verify that the application is safe against those threats.

Why doesn't the criteria talk about SQL Injection, Cross Site Scripting etc.?
That's a good observation! When writing the criteria, the authors first considered listing all the attacks they wanted the application tested for. They soon discovered the pitfalls of that approach: the criteria would have to be revised with the discovery of every new attack, and there was no standard taxonomy of attacks to start with, anyway. Look at criterion 1: "Safe against popular attacks", that's the criterion which catches many of those attacks. Here're some of the popular attacks the application must defend against:

Variable Manipulation	Denial of Service	Brute Force
Bypass input validation	Session Hijacking	Session Fixation
Content Spoofing	Cross-site Scripting	Cross Site Tracing
Buffer Overflow	Format String Attack	SQL Injection
OS Command Injection	LDAP Injection	XPath Injection
SSI Injection	Blind Injection attacks	Directory Indexing
Directory Traversal	Authentication Bypass	Filter evasion

You missed criterion xyz!
Thank you for pointing out. We are constantly looking at ways to improve the criteria – so your suggestions will help. While not all suggestions might make it to the standard after evaluation, they ensure that the criteria gets the best ideas for certifying applications.

What do you mean by sensitive data in the criteria?
Sensitive data is any data that if compromised could lead to:

- Loss of business in financial or reputation terms
- Loss of privacy of individuals
- Adversaries gaining an advantage to launch attacks

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

What's all that about passwords with less than 10,000 values, "password guessing" etc.?
It's simple, really: an application must first protect against automated tools that try to guess passwords – the use of CAPTCHAs is a good example. In case the password is, say a 4-digit numeric PIN, that takes limited range of values, then an adversary might try to guess the password manually, and the application should defend against that – temporarily locking the account after 3 failed attempts is a good example.

The standard says that the application must not allow passwords to be stolen even after the user logs out. Am I missing something?
In general, the application must safeguard passwords. The note highlights that the password should be protected even when the user has logged out. Vulnerabilities in some authentication schemes enable the password 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 – but then, which app doesn't do that?
Sounds obvious, rt.? Experience, however, shows that not all applications enforce re-authentication after log out. 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. This Microsoft support document has examples of such protection:

`http://support.microsoft.com/default.aspx?scid=kb;en-us;900111`

You don't mention SSL anywhere in the criteria… gotcha!
The Standard doesn't mention SSL, or any specific encryption technology: that's intentional. The criterion "Sensitive data protected 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, do not demand SSL. The application is free to choose the technology, so long as it protects sensitive data in transmission.

What's with the "No sensitive data in requests to external sites"?
This is a favorite of the authors. Links to external sites are often a source of data leakage. Along with the URL, the HTTP request also includes a referrer field that contains the URL of the page the request originates from. 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 have access to those logs and can collect the sensitive information trivially.

No sensitive data in source code available to users; ok, but how will users access any source code of a web application?
You forget client-side javascripts. Worse, sometimes web server mis-configurations lead to server-side source code being available to adversaries. The criterion requires that any source code that's available to users not contain sensitive data.

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 criteria only demand that no sensitive data be revealed in error messages as that has direct impact on the application's security.

The latest version of the Plynt Certification Criteria is available at http://plynt.com/criteria/

Visit http://plynt.com/ for more details on certification and pricing.

Plynt, Inc.

12801 Worldgate Dr., Ste 500
Herndon, VA 20170, USA

Phone: 1(866)759-6824 or 1(866)PLYNT24
Fax: 1-703-871-3936
Email: plynt@plynt.com

Global Delivery Centers

Herndon, VA

12801 Worldgate Dr., Ste 500
Herndon, VA 20170, USA
Phone: +1-703-871-3934
Fax: +1-703-871-3936

Mumbai, India

307, Devavrata, Sector 17,
Vashi, Navi Mumbai - 400703
Phone: +91-22-27892889
Fax: +91-22-55913580

Bangalore, India

N-307, Manipal Centre
Dickenson Road, Bangalore - 560042
Phone: +91-80-25092108
Fax: +91-80-25092108

Kuala Lumpur, Malaysia

F313, Phileo Damansara 1,
46350 Petaling Jaya, Malaysia
Phone: +60-3-7960-4275
Fax: +60-3-7660-4273