Title: OWASP Web Security Certification Criteria

Author: Mark Curphey

Version: 1.0

Status: Draft

Date: Monday, July 30, 2007

Contents

Introduction

Web site owners need a widely published and consensus driven set of criteria to design, develop, deploy and maintain secure web sites. This criteria and claims of compliance with it need to be able to be provided to a wide range of stakeholders including customers, regulators and business partners.

This document is a discussion document created by Mark Curphey . It was sponsored and produced as part of the OWASP Spring of Code, 2007. It proposes an evaluation and certification scheme for the security of web sites including recommendations for how the evaluation and certification process itself could work. This work is intended to be openly published for a reasonable period of time for public discussion, debate and feedback. After this period the OWASP Board will work with interested parties to determine any appropriate next steps. These may include adoption or integration into existing standards or the creation of something new.

The evaluation and certification scheme proposed here takes into account the motivations and needs of a variety of stakeholders. Many people including the author have been highly critical of the Payment Card Industry Data Security Standard (PCI DSS). The OWASP Web Security Certification Criteria is not a proposal to replace the PCI DSS and is not officially related in anyway shape or form. PCI DSS has been taken into account however we have intentionally chosen not to build upon or build around key PCI issues that we consider ill-conceived. In short we have decided to build on solid foundations from the ground up.

It is very important to understand that in itself this document and the project that supports it is not an evaluation scheme or criteria, but a proposal for what an effective one may look like. An in-person workshop / panel discussion is expected to take place at the next OWASP Conference in San Jose in November of 2007.

This document comprises of two main parts;

Part 1 – Implementation Considerations. This section describes key processes that should be considered in order for any evaluation and certification scheme to be effective.

Part 2 – Evaluation Criteria. This section describes the actual criteria being proposed. It adopts the recommendations from Part 1.

You can send your feedback directly to mark@curphey.com or at the OWASP mailing list dedicated to this project (https://lists.owasp.org/mailman/listinfo/owasp-webcert).

We hope this document provides value and provokes thought to all those identified in the stakeholders section.

Kind regards,

Mark Curphey and the entire OWASP Project Team.

Part 1 – Implementation Considerations

Section Summary

This section describes recommendations for implementing an evaluation and certification scheme using the criteria set out in Part 2 of this document. It is intentionally presented first as its recommendations are implied in the criteria laid out in Part 2.

In order for any scheme to be effective it must support an eco-system that develops a sustainable and scalable business model based for all stakeholders and be built around an open, free and widely supported set of criteria. The approach of this proposal is to describe a common set of criteria and a framework from which various stakeholders could then derive and administer their own individual schemes based on their specific needs. It provides the commonality that can minimize the impact to business as a whole by providing generally agreed upon

practices and criteria. Individual schemes or organizations can then focus on specific requirements. Ironically this is the same approach as object orientated programming.

If the mantra is "don't reinventing the wheel" the criteria presented here could be considered an attempt to embody the wheel so others don't need to constantly re-invent it. By adopting this approach system owners can choose to adopt the base criteria at the core of their application security program knowing that they can reduce the cost and improve the efficiency of doing business by participating in such an open and free scheme and maximizing their efforts.

Part 1 describes key principles considered when proposing this certification and evaluation scheme, many of which are considered failings in current schemes. At the core of the proposed scheme is the concept of assurance; a way to acknowledge that in the real world we need to find a balance between different approaches to evaluating security with the different needs and commercial pressure of business. Parties can require or offer various levels of assurance based on their needs or the requirements imposed on them. It also proposes Basic and Extend Criteria, a way for system owners to demonstrate security beyond the lowest common denominator and for scheme administrators to have the flexibility to tune the criteria based on industry specific criteria.

The proposed criteria itself covers people, Process and Technology, three key factors that have both an immediate and long term affect on the security posture of web sites. For each section and individual criteria defined in Part 2 unambiguous requirements are defined. A scoring system and process is presented which allows for repeatable and consistent evaluations and an auditable certification process.

Comment [MSOffice1]: Ma ke sure this gets finished.

Finally Par t 1 makes recommendations about how any evaluation and certification scheme itself should be administered including specific recommendations about how auditors themselves should be certified to evaluate web sites and how the evaluations should be recorded and distributed.

Understanding the Stakeholders

There are a number of stakeholders in the web security certification game. Only by considering the motivations, goals and intentions of all stakeholders will we build an evaluation and certification process that will work in the real world. While it is clear that many of the stakeholders have competing interests, it is also clear that there are a great deal of common interests and goals. We believe that a balance can be found and with careful thought many of the goals and interests can be accommodated without compromising or reducing the effectiveness of the goals and interests of others. Some motivations, goals and intentions are of course simply bi-polar and this must be accepted. An example maybe some security vendors who would like to make as much money as is possible while at the same time most companies want to spend the least amount of money as is possible to achieve the desired level of risk. Where this is the case any security criteria must clearly side with the "greater good" and not commercialism.

The following list of stakeholder interests has been considered;

- **Business Managers** – who are responsible for the overall web site (or portions of it) want to ensure that they are taking the appropriate steps to optimize the business performance and manage their corporate risk. This usually involves balancing security measures to provide assurance to their customers, business partners and regulators and to protect their own business, against considerations such as performance, usability, time and money.
- **System Architects** – who are responsible for specifying and designing the underlying technology that powers web sites want to ensure that their designs will meet the requirements defined by the business owners including resource constraints such as cost, time and effort. They want to ensure their designs will meet internal

and external standards and regulations and can be effectively and efficiently implemented by system developers.

- **Security Architects** – who are responsible for specifying and designing the underlying security that power web sites want to ensure that their designs meet the requirements defined by the business and work in conjunction with designs developed by the system architects. They want to ensure their designs meet internal and external standards and regulations and that their designs can be effectively and efficiently implemented by system developers.
- **System Developers** – who are responsible for implementing design specifications want to take clear direction on what to build and how to build it from the system and security architects and be able to produce software that meets the requirements within their constraints (usually time and effort).
- **Corporate Information Security Officers –** who are responsible for defining information strategy and policy and managing information security risk want to ensure that web sites are developed in line with their corporate risk management strategy and will meet the requirements of internal and external standards and regulations. They want to ensure the appropriate level of resource is spent to address the level of risk.
- **Internal Auditors –** who are responsible for ensuring that corporate standards are being met, evaluate information security and technology teams and projects usually by requesting and inspecting proof that internal and external standards and regulations have been met. They are generally concerned with people doing the right things.
- **Operational Managers –** are responsible for deploying and managing technology and want to ensure that performance and service levels are met as well as any obligations resulting from external standards sand regulations.
- **Security Consultants –** are interested in making money by developing services that they can sell and perform for companies. This includes managed services providers.

- **Security Product Vendors –** are interested in making money by developing products that they can build, sell (and in some cases run) for companies.
- **Regulators –** are interested in overseeing the greater good of a particular industry by ensuring the companies are taking appropriate steps with their security to protect users the industry itself and or society.
- **Web Site Users –** are interested in understanding if a web site is considered safe to use and or do business with.
- **Business Partners –** are interested in specifying and maintaining information security as part of a formal business relationship.

Principles for a Good Evaluation Scheme

This section describes some key principles that we believe should underpin a security certification criteria. Each of these principles has been adopted in this proposal and provides clear benefits to the overall scheme.

These principles are;

- Risk Based Security
- Assurance
- Auditable and Unambiguous
- Repeatable

Risk Based Security

Risk based information security may not always be a trendy term, yet established thinking rarely goes out of fashion. Different types of web sites face different levels of risk.

The American National Institute of Standards and Technology (NIST) provides good definition of Risk and associated terms below;

- *Risk* is a function of the likelihood of a given *threat-source*'s exercising a particular potential *vulnerability*, and the resulting impact of that adverse event on the organization. To determine the likelihood of a future adverse event, threats to an IT system must be analyzed in conjunction with the potential vulnerabilities and the controls in place for the IT system.
- Threat: The potential for a threat-source to exercise (accidentally trigger or intentionally exploit) a specific vulnerability.
- Threat-source: Either (1) intent and method targeted at the intentional exploitation of a vulnerability or (2) a situation and method that may accidentally trigger a vulnerability.

- Threat Analysis: The examination of threat-sources against system vulnerabilities to determine the threats for a particular system in a particular operational environment.
- Vulnerability: A flaw or weakness in system security procedures, design, implementation, or internal controls that could be exercised (accidentally triggered or intentionally exploited) and result in a security breach or a violation of the system's security policy.[1]

It naturally follows that different classes of web sites have different levels of inherent risk.

Threats - Some web sites such as those run by financial services companies may have more threat agents with better capabilities and who are more motivated to attack them than perhaps a local gardening club site.

Vulnerabilities - In general the classes of vulnerabilities in different types of web sites are similar but are of course dictated by the functionality in the system. For instance some types of systems using certain types of technology such as AJAX may have to consider certain categories of vulnerabilities where a web site that doesn't use a technology such as web services may not have a consider another specific category.

Assurance

To have assurance about something implies that you have confidence in it. Having confidence that a statement or set of claims about the security of a web site is accurate is clearly very desirable yet assurance has implications beyond just accuracy. For instance a higher degree of assurance may mean that you can place more trust in a set of claims knowing that they are more likely to be repeatable and consistent.

Different assessment techniques can provide different levels of assurance (confidence) on claims about the security of a web site.

[1] NIST 800-30 - http://csrc.nist.gov/publications/nistpubs/

Specifically some tools are better than others at determining specific classes of vulnerabilities and some techniques are better at finding certain types of problems than others.

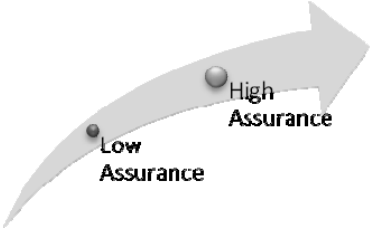
It also follows that a company who embeds security in their entire software development lifecycle (SDLC) will be more likely to consistently produce the assessed results and therefore a 3rd party can have a higher level of assurance on any claims.

We must accept that in general assurance comes at a cost. It is more expensive to produce facts on which higher assurance claims can be made. For example it is more expensive to test something in a way in which you can make a higher assurance claim. Using technical testing as the example, a higher degree of manual effort (higher cost) is generally required over lower assurance automated approaches (lower cost).

If we define the level of risk inherent in specific classes of web site, we are then able to pragmatically apply the principles of assurance. For instance it makes perfect sense that for a high transaction value financial services application we would want to have a high level of assurance in the security claims whereas a very low transaction value site would not.

Auditable and Unambiguous

Any criteria for security must be auditable; that is to say it must be something that can be inspected and validated. If something cannot be inspected and validated then it should be defined as a set of principles and not as a set of criteria. This document sets out principles in this section and criteria in the main body of the document.



Some security standards make statements that are ambiguous. One example is the PCI DSS that says “only necessary ports should be

open". The default effect of this ambiguous statement is for all sites to legitimately claim that all open ports are necessary and everyone passes. "The Remote Desktop Protocol is necessary to remotely manage the web site host" for example. In general ambiguity usually occurs when standards writers attempt to write "catch all" statements without precision. In a world where development technology is intentionally designed to create unique web sites that themselves have unique functionality, it is understandable that very few sites will share an exact set of functionality. Therefore in many cases specifying a criteria that "fits all" leads to ambiguous statements which themselves become meaningless as people can easily navigate around them with context.

A better approach to avoid ambiguity is to write concise statements and provide a flexible process to request exceptions. For example a better approach to the PCI DSS open ports issue above would be to say "Only ports 80 and 443 allowing inbound and outbound HTTP and HTTPS traffic should be enabled". All other ports requests could then be documented by an accredited auditor and submitted to an approval process along with a set of documented guidelines for approval. This approach allows for a standard to specify security that should apply to the majority and provides a fair and effective process to avoid penalizing the minority.

Repeatable

No two people are the same. While intelligent interpretation is one of the beauties of the human race it is important that evaluation claims are repeatable. Two individuals or two teams should arrive at the same conclusion and be able to justify their results. In a commercially driven world where the lowest cost or the fastest results often wins business, we must ensure that specific security claims are not compromised by commercial pressure or equally importantly that we understand the trade-offs that may have been made due to commercial pressure. The notion of assurance levels of course allows these influencing factors to be taken into account when producing security claims.

The Scope of Good Security (People, Process and Technology)

Information security is complex topic that is neither art nor science. Many still view it as a technology problem but as the well known phrase goes "if you think technology is the solution to information security then you don't understand the problem".

It is therefore important that we address the full scope of information security when making a statement about security posture. Making a statement solely about the posture of the technology implementation itself may provide valuable insight but additional statements about the people who built the system and the process they used can provide a deeper understanding and in many cases a higher degree of assurance about other statements.

The total proposed criteria includes People, Process and Technology. End users or scheme administrators would be free to choose if they wish to evaluate against any section of the scheme or all of it. This approach provides an additional level of flexibility and allows for varying degrees of assurance. While all companies would be advised to evaluate and certify their entire People, Process and Technology this may not be deemed necessary or maybe considered as a phased approach as part of a security improvement program.

The Concept of Basic and Extended Criteria

How much security a web site needs is obviously not simple question to answer and experience would tell us that it would depend on the risk profile of the site. This poses a challenge to any organization that wishes to lay down a set of criteria that an individual or a specific set of sites must meet. How can you possibly categorically require a uniform set of security controls when you don't understand the individual risk profile? While this may indeed be true, in the practical world assumptions can be made and stereo-types can be and are drawn. We typically say that a group of sites share similar risk profiles (maybe financial services or those that process credit cards) and should all implement a common set of security criteria. Usually these criteria are selected based on security, functionality and cost (to implement, maintain and audit).

This document proposes a Basic and Extended criteria where the Extended criteria represents incremental additional controls. A system owner may choose to adopt the Extended criteria to demonstrate additional confidence to their customers or partners or a regulatory authority may seek to impose the Extended criteria on a site that differs from the normal risk profile of others with the group or for a specific control where a specific threat is considered to be higher.

The Extended criteria gives system owners the ability to demonstrate they have gone beyond the minimum required and demonstrate security as a competitive differentiator. A system owner may choose to offer Extended Criteria as an option to specific users or groups of users.

The Extended criteria gives scheme administrators the flexibility to define minimum criteria for the majority and specific criteria where needed.

Ratings – Scorecards and Certificates

Some people such as end users simple want a binary answer about security; "Is the site considered secure or not?". Others need a more granular view such as "Are we confident this site has no SQL injection issues?". Others still want a view on a specific part of the security posture such as "How do they rate for process?" while another may only care about a single point in time implementation.

This proposal recommends a system of scorecards and certificates. A standard score card (template) is completed by the auditor to build a map of the security posture. This scorecard may be an interim report representing remediation work required or a record that can be jointly submitted to a certification process for a formal certificate and approval. A sample scorecard and sample certificate can be found in the Appendix.

Each scorecard follows the criteria sections for People, Process and Technology. Each individual section would require an individual score, the auditors identity and meta-data such as time, date and notes to be completed.

Results would calculated with one of two possible outcomes for each individual section;

1. Pass - Meets the Criteria Defined
2. Fail – Does Not Meet the Criteria Defined

Each group also has a simple "Pass" or "Fail". Any group that has one or more individual "Fail" will receive a group "Fail".

Extended criteria (see above) are scored in the same way as individual sections however scores are not taken into account when determining the group score. If a site has passed all the extended criteria for the group section then this is noted on the scorecard. Sites that pass all Basic and Extended sections for all sections are able to apply for a Gold Certificate Status.

Dealing with Exceptions

It is clear that in a world where development technology is designed to facilitate building unique solutions that one size solution will not fit all. While the approach of specifying common solutions may make sense in many circumstances, many evaluation and certification schemes are criticized for forcing specific implementations while some perfectly valid solutions may be outlawed unnecessarily. It would clearly be a futile approach to try to list and maintain all possible secure ways of implementing a specific control.

It is therefore proposed that an exception process is operated by all schemes. The process would work by allowing the certified auditor and system owner to jointly request an exception to a specific issue. The exception request would be reviewed by a regional panel with the ability to approve or reject it based on common sense. The regional panels members are suggested below. An applicant should have the right of appeal in which an issue would be referred to the governing body. Exceptions could be requested on a temporary or permanent basis. All judgments should be published openly taking care not to disclose sensitive information about system designs.

This system allows maximum flexibility for system owners but places the onus on the system owner and specifically the auditor to demonstrate secure alternatives to the documented criteria.

Managing the Scheme

This sub-section proposes how an evaluation and certification scheme may be operated. It certainly does not attempt to provide an operational plan but highlights key issues that should be considered.

Roles and Organizational Structure

It is clear that any information security evaluation and certification criteria will have implications on various entities and ensuring that undue bias does not influence the criteria is critical. A transparent, open and fair process would need to be established.

OWASP Evaluation and Certification Committee

We propose that OWASP establishes a formal OWASP Evaluation and Certification Committee that is responsible for the development and maintenance of the certification criteria itself. This committee would be made up of a Chair and member participants. The committee should be made up of at least one representative from each of the stakeholders identified in this document. As with other committees key roles such as representatives could be voted for by OWASP members on an annual basis. Any organizations implementing an evaluation and certification scheme would be invited to and encouraged to join the committee. The committee would be expected to meet in person at least once a quarter.

Scheme Governing Body

Each organization implementing an evaluation and certification scheme based on the OWASP Evaluation and Certification Criteria should establish a governing body. It would be the bodies responsibility to implement and administer all aspects of the scheme. It is important that member ship of the governing body is in no way perceived to be achieved via subscription. A representative from the governing body should reside on the OWASP Evaluation and Certification committee.

Regional Panels

Each scheme should implement regional panels. Regional panels should consist of invited certified auditors and a regional chair. The regional panel would preside over regional matters such as local legislation and make recommendations to the scheme governing body. It would also be responsible for passing judgment on all exception requests.

Scheme Advisory Panel

All schemes should have an official advisory board made up of experts on specific topics. These would likely include privacy, the law and specific technology domain experts.

Certifying Inspectors

In order to have repeatable and accurate statements about the security posture of a web site it is important to have a solid certification process for the auditors themselves. The following are considerations for an auditor certification process.

The Reliance on Inspectors – This proposal places a heavy reliance on qualified auditors. It is incredibly hard to describe exactly how a specific control should be designed or implemented. Reliance must therefore placed on the skill and judgment of qualified inspectors who are able to make informed judgments on behalf of the scheme implementers. Any scheme that heavily relies on inspectors must ensure appropriate monitoring and discipline processes support the scheme.

Certify individuals – It is imperative that only individuals themselves are certified. Companies must not be able to obtain blanket certification to perform evaluation and certification work.

Certification Types - It is imperative that individuals are granted certification for specific topics and not simply a blanket certification. Many individuals have different specializations and skill levels. These should be embraced allowing process specialists to flourish. Of course individuals could obtain and maintain all certification categories if capable. The following certification categories should be considered.

- People – A specialist able to determine the skill level and competency of a developer or team of developers to define, design, develop, deploy and maintain secure software.
- Process – A specialist able to determine a developer or development teams ability to understand the requirements and implement a secure development process.
- Technology (Architect) – A specialist able to evaluate a security of a web sites architecture.
- Technology (Code Review) – A specialist able to evaluate the security criteria by examination of the source code. Code review certifications should further be granted specific to languages and frameworks such as C#, ASP.NET or Ruby and Rails.
- Penetration Testing – A specialist able to evaluate the security criteria by penetration testing.

Certification should be linked to a certification license, itself tied to an individual. All evaluations should be submitted against the individuals license, that is to say statements made by a person against his / her license. The scheme owners would maintain the right to temporary or permanently revoke a license based on performance or other criteria. For instance if a web site that was previously certified was hacked, the regional panel may revoke the license preventing that certified auditor from submitted any further evaluations until the matter was investigated and it was determined that the issue was outside of the scope of the previous evaluation.

Random Spot Checks and Recertification – It is clear that many people can study to pass exams. Any scheme should administer random spot checks on certified auditors. All auditors should be required to recertify at least once a year.

Evaluation Periods and Certificate Registers

Brand and Reputation Management

In business where trust is everything maintaining a brand and reputation is critical. Some certification schemes have seen vendors form consortiums and alliances around a standard which through marketing can sometimes appear to the uninitiated to be formally connected or representing the governing body itself. The consortiums can make exaggerated claims and undermine the credibility of the governing body and the entire scheme. Care must be taken to ensure that an eco-system is encouraged but one which balances the needs of all stakeholders and protects the credibility of the scheme and is in keeping with the open source ethic and mantra.

Through copyrights and licensing, content and the ability to use it should be licensed to all stakeholders. Varying licenses should be drawn up to meet the exact needs of the various stakeholders but protect the schemes interests.

Part 2 – The Evaluation Criteria

Overview

- People
- Education
- Competency

People

Education Program

Qualified Developers

Qualified testers

Process

Define

Security requirements should be included

Security Testing time should ne included in project plans

Design

Threat modeling

Develop

Code Review

Deploy

Infrastructure

Maintain

Change Management

Technology

Infrastructure Security

It is clear that application security cannot be achieved without the orchestration of physical, network, host and database security (and often several other important supporting technologies like application servers). Many assumptions are made in design, development and implementation and many underlying security mechanisms are relied upon to provide specific services.

This clearly poses a challenge for an application security evaluation and certification criteria. Ignoring the issues is not an option yet defining and maintaining an evaluation and certification criteria for this wide range of topics is a colossal task. Many other standards have attempted to solve this challenge by defining a small subset of issues considered to be the most important.

This document proposes to take a different approach and acknowledge the importance and complexity of these supporting technologies and also acknowledge that there is currently a lack of good reusable evaluation criteria from which we can currently reference. We propose that one of the next steps from this project would be to identify potential groups and work with them to develop and maintain suitable criteria that could be referenced by anyone implementing a scheme based on this document. Topics covered would be expected to include;

- Physical Security
- Network Security
- Operating System Security
- Database Security

User Management

This section describes requirements for the managing users on the system. Separate sections cover authentication and authorization.

Number	UM-001
Name	*Username Format (Basic)*
Description	A username is an identification token used to identify a user or process
Requirement	Usernames should be an identification token unique to the system. <Scheme implementer to insert details here> <Suggestions: No SSN, no CC Number etc>
Criteria Type	Basic
High Assurance	A manual examination of the source code should be made to determine that application enforces this control.
Medium Assurance	A manual inspection of the web site should be made to determine that the application enforce this control.
Low Assurance	
Very Low Assurance	
Evaluation Notes	
Score	

Comment [MSOffice2]: Do es this even matter? I don't think it does but I know others are passionate about it.

Number	UM-00X
Name	*Unique usernames*
Description	A username is an identification token used to identify a user or process and must be unique.
Requirement	All usernames should be unique. If email addresses are used as unique usernames, application should inform user if email address is already registered (and provide a link to password recovery) only via email. If email address is not already registered – application should email a unique URL to continue the registration process. <Suggestions: Self-selected usernames, email addresses, etc>
Criteria Type	Basic
High Assurance	Manual code review
Medium	Manual penetration test

Assurance	
Low Assurance	Registration of same username is successful but cannot enumerate password Registration of same username errors
Very Low Assurance	Registration of same username is successful and errors on same password Registration using same username with different passwords succeeds Registration using hex or other character encoded usernames succeeds
Evaluation Notes	
Score	

Comment [MSOffice3]: Do es this even matter? I don't think it does but I know others are passionate about it.

Number	UM-00Y
Name	*Non-predictable usernames*
Description	A username is an identification token used to identify a user or process and must not be predictable.
Requirement	All usernames should be unpredictable. Self-registration of usernames is optional. If self-registration is unavailable to new users, then the application doesn't need to disclose whether or not a selected username already exists. <Suggestions: 16-character randomized UUID's, etc – Do not allow cust1, cust2, cust3 enumeration>
Criteria Type	Extended
High Assurance	Manual code review
Medium Assurance	Manual penetration test
Low Assurance	Automated code review or automated penetration test
Very Low Assurance	Design inspection
Evaluation Notes	
Score	

Number	UM-002

Name	*Username Format (Extended)*
Description	A username is an identification token used to identify a user or process
Requirement	Email addresses should NOT be allowed as usernames if email is one of the password reset options.
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-003
Name	*Account Management Activity over SSL*
Description	All user account management activity should be protected in transit.
Requirement	All user account management activity including account maintenance, and password resets should take place using 256 bit SSL (with valid certificates).
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-004
Name	*Password Strength*

Description	A username is an identification token used to identify a user or process
Requirement	A strong password should be enforced. <Scheme implementer to insert details here> <Suggestions: No blank passwords, not set to the same as the username, and no dictionary words or names. Passwords should be at least 7 characters in length, or at least 8 characters long if using Microsoft NTLM.>
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-00Z
Name	***Password Strength (Extended)***
Description	A username is an identification token used to identify a user or process
Requirement	A strong password should be enforced. <Scheme implementer to insert details here> <Suggestions: Password should be a minimum of 10 characters with at least one uppercase letter, one lowercase letter, one number, and one special character. Additional requirement: password should be 15 characters long if using Microsoft NTLM.>
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>

Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-005
Name	*Password Expiry*
Description	Passwords should only be valid for a limited lifetime.
Requirement	Password aging should be enforced. <Scheme implementer to insert details here> <Suggestions: change every 30 days>
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-006
Name	*Password Lockout*
Description	User accounts should be locked out if the password entered is incorrect after a fixed number of attempts.
Requirement	Password lockout should be enforced. Account holder should not be notified that any specific account has been locked out. Account holder should be advised to phone customer support and/or answer a series of security questions for successful login. If a locked out account attempts to login, it should be rejected immediately – without checking credentials. <Suggestions: After 5 bad attempts>
Criteria Type	Basic
High Assurance	
Medium Assurance	

Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-0AA
Name	***Password Lockout (Extended)***
Description	User accounts should be suspended for a certain time period if the password entered is incorrect after each credential failure starting after the second.
Requirement	Password suspension should be enforced and accounts should not be locked out (preventing attackers the ability to deny access to legitimate users). Account holder should not be notified that any specific account has been suspended. All failed login attempts (including invalid username) should respond with the same generic message, "Accounts are suspended if multiple login failures occur. If you believe that your account has been suspended, please try again later". If any suspended account attempts to login, it should be rejected immediately – without checking credentials. <Suggestions: Exponential time suspension after second bad attempt. For example, after two password tries – 5 minutes, three password tries - 10 minutes, etc>
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-007
Name	*Password History*
Description	Users should be required to choose fresh passwords when changing them to avoid cycling though easily guessable or common passwords.
Requirement	Password history should be enforced. <Scheme implementer to insert details here> <Suggestions: Rotation of previous 5>
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-008
Name	*Password Storage*
Description	Passwords should be stored in an encrypted form.
Requirement	Encrypted passwords be enforced. <Scheme implementer to insert details here> <Suggestions: Specify algorithm, key length, key management specs>
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-009

Name	*Password Storage (Extended)*
Description	Passwords should be stored in a non-reversible encrypted form.
Requirement	Non-reversible encrypted passwords be enforced. <Scheme implementer to insert details here> <Suggestions: Specify algorithm such as SHA-1 >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-010
Name	*Password Reset: Secret Question and Answer*
Description	
Requirement	A secret question, secret answer password reset system should be enforced. Automated password resets should only be initiated upon successful completion of a secret question and secret answer. Care should be taken when using secret questions because accounts can be brute-forced very easily when using this method. If used, a lockout or account suspension should also be utilized after a certain amount of failed responses to the secret answer function. Application should email the user a link to change password and never display an actual workable password (or re-authenticate the user into an active session with the application). <Suggestions: Asking the name of the high school attended is better than asking the name of a first pet. >
Criteria Type	Extended
High Assurance	

Medium Assurance	
Low Assurance	
Very Low Assurance	User-provided challenges
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-011
Name	*Password Reset: Secondary Channel*
Description	Passwords reset requests should be dealt with using a secondary channel such as email.
Requirement	All password reset requests should be processes via a secondary channel. New passwords should never be issued to the requestors web browser without validation over the secondary channel first. The most common secondary channel for password reset functionality is through email. Clear-text, workable password should never be sent to the user. Instead, a unique, time-limited, single-use URL that relies on SSL should be sent to the account holder's email address. Visiting this URL will allow the user to change the account password, after which another email is sent indicating that the password has changed. <Suggestions: Suggest email >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-012

Name	*Password Reset: Out of Band Channel*
Description	Password reset requests should be dealt with in an out of band channel such as SMS.
Requirement	All password reset requests should be processes via an out of band channel. New passwords should never be issued to the requestors web browser without validation over the out of band channel first. <Scheme implementer to insert details here> <Suggestions: SMS, Phone calls (automated or service-person), mail-order. >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	UM-0AB
Name	*Non-predictable initial passwords*
Description	A password is a verification token used to identify a user or process and must not be predictable.
Requirement	All passwords should be unpredictable. If initial passwords are chosen for the user before they are able to pick their own password then the password must be unpredictable. <Suggestions: 16-character random password, etc>
Criteria Type	Basic
High Assurance	Manual code review
Medium Assurance	Manual penetration test
Low Assurance	Automated code review or automated manual penetration test
Very Low Assurance	Design review

Evaluation Notes	
Score	

Comment [MSOffice4]: Do es this even matter? I don't think it does but I know others are passionate about it.

Number	UM-013
Name	*User Logout*
Description	All users should be able to easily logout of the application.
Requirement	A user logout link should be clearly visible on all authenticated pages which when clicked would log the user out and invalidated all session information. <Scheme implementer to insert details here> <Suggestions: None>
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Authentication

Number	AUTHN-0AA
Name	*Login Form via POST*
Description	Information sent over HTML forms should be sent over POST instead of the GET method, even if submitted over SSL.
Requirement	The login form should only be displayed on a form that is part of a page which is served up over an HTTP POST request. <Scheme implementer to insert details here> <Suggestions: POST is the only method to be used for login forms.>
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	AUTHN-001
Name	*Login Form via SSL*
Description	Users should be able to easily determine that they are entering their credentials at a legitimate site. This protects against phishing and other "man in the middle" attacks.
Requirement	The login form should only be displayed on a form that is part of a page which is served up over bit SSL with a valid certificate. <Scheme implementer to insert details here> <Suggestions: Add SSL version and cipher suite (key lengths) >
Criteria Type	Basic

High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	AUTHN-002
Name	*Login Data via SSL*
Description	All sensitive data should only be sent via SSL.
Requirement	The login form should only be displayed on a form that is part of a page which is served up over bit SSL with a valid certificate. <Scheme implementer to insert details here> <Suggestions: Add SSL version and cipher suite (key lengths) >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	AUTHN-0CC
Name	*Fail-Closed Login Method*
Description	The login methodology should fail-closed instead of fail-open.
Requirement	The application should not be subject to a logic flaw where an unhandled exception would cause the application to login a user (or default user) or grant any session tokens.

	<Scheme implementer to insert details here> <Suggestions: TBD>
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	AUTHN-003
Name	***Auto-Complete Off on Login Form***
Description	The auto-complete HTML tag can automatically fill in form field data.
Requirement	Auto-complete should be explicitly set to off for all login forms. <Scheme implementer to insert details here> <Suggestions: >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	AUTHN-0DD
Name	***Caching Off on Login Form***
Description	The caching of HTML or scripts should all be turned off on the login form.
Requirement	Caching should be explicitly set to off for all login forms. The “no-cache” directive applies to server

	responses, while the "no-store" applies to both client requests and server responses, which is ideal for privacy situations such as these. Note that this shouldn't be relied upon, as malicious or compromised caches may not obey or recognize these directives. Some browsers also allow auto-completion which saves form data – so this feature should also be turned off. <Scheme implementer to insert details here> <Suggestions: HTTP responses with *Cache-Control: no-store* and *Expires: 0* or possibly *Pragma: no-cache* if Cache-Control header directive is not supported. For auto-completion – use *autocomplete=off* >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	AUTHN-004
Name	***Re-Authentication***
Description	Re-authenticating users prior to significant events reduces the risk if their session was hijacked.
Requirement	Users should be required to re-authenticate before significant transactions <Scheme implementer to insert details here> <Suggestions: Define transactions >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	

Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	AUTHN-005
Name	***Two Factor Authentication***
Description	Using two factors for authentication such as something you own and something you possess improves the quality of authentication.
Requirement	Systems should implement two factor authentication for users or groups of users. <Scheme implementer to insert details here> <Suggestions: Tokens or others >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Session Management

Number	SESS-001
Name	*Session Expiry*
Description	HTTP is stateless meaning that it is left to the application to implement state control.
Requirement	All user session tokens should expire after a period of inactivity. The application should make this actionable in the same way as if the user had logged out of the application.sc <Scheme implementer to insert details here> <Suggestions: 20 mins >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	SESS-002
Name	*Remember Me*
Description	HTTP is stateless meaning that it is left to the application to implement state control.
Requirement	Remember Me functionality that creates a long-term session should only be used to present data that would normally be made available to a user without authenticating. Where Remember Me functionality is used a clear statement about the security implications and a clear list of the data being made available should be shown. <Scheme implementer to insert details here> <Suggestions: Don't do it!>
Criteria Type	Basic

High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	SESS-003
Name	*Session Token Transport*
Description	If an attacker intercepts the users session token then he could assume the identity of the user.
Requirement	All session tokens issued after authentication must only be sent over SSL. <Scheme implementer to insert details here> <Suggestions: Add SSL version and cipher suite (key lengths) >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	SESS-004
Name	*Session Token Strength*
Description	If an attacker generate or predict a users session token then he could assume the identity of the user.
Requirement	All session tokens should be created so they could not be computed or recreated by an attacker. Session token should be able to withstand cryptanalysis (using current cryptanalysis estimations

	here) for a period of 100 times greater than the longest time the session could possibly be kept alive at the time of the inspection. <Scheme implementer to insert details here> <Suggestions: Insert bit length, token generation libraries or token types >
Criteria Type	Basic
High Assurance	Utilize NIST FIPS-140-2 to check for statistical anomalies with code review
Medium Assurance	Manual or automated code review
Low Assurance	Using tools such as Stompy or Burp Sequencer to automate analysis during automated or manual penetration tests
Very Low Assurance	Use WebScarab to analyze a large number of session ID's from various parts of the application from different usernames, IP addresses, and time-periods. Automate penetration testing of session ID's across the application
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	SESS-005
Name	***Session Reuse***
Description	If an attacker generate or predict a users session token then he could assume the identity of the user.
Requirement	All session tokens should be created so they could not be reused on a host other than the one it was issued to. <Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	Concurrent logins work

Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Authorization

Number	ATHZ-001
Name	*Session Token Refresh on Login*
Description	The session token should change when a user logs into an application.
Requirement	In order to prevent earlier tokens from controlling state, a newly refreshed token should be used to identify an assumed logged in user's session. <Scheme implementer to insert details here> <Suggestions: TBD >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATHZ-002
Name	*Per-page Tokens*
Description	Per-page tokens can be used for finer-grained access control.
Requirement	Each time a user makes a request for a new page, each page token is checked against the last page's token attributes and values. This is after verifying the primary session token. This can be used to watch the path a user takes through an application, and potentially validate every action a user performs that would affect access control (authorization for access to objects, files, etc). Some out-of-sequence attacks that affect access control or logic can be optionally prevented with this information. <Scheme implementer to insert details here>

	<Suggestions: TBD >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATHZ-003
Name	*Check Privileges*
Description	Every user has a session token which can be tied to access controls based on privileges.
Requirement	Using a table of privileges, every credential can be checked for each URL, file, or database privilege by mapping user types, roles, or individual permissions to each username. <Scheme implementer to insert details here> <Suggestions: Use a privilege table with programmatic, DAC, RBAC, or declarative controls. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Comment [S5]: Does this dictate architecture? For instance what happens with a SAML implementation where you pass token to something?

Number	ATHZ-004
Name	*Check Permissions*
Description	Check permissions to databases, local files, and

	system accounts.
Requirement	Users with logged-in credentials will often have access to databases, local files (download or upload), and may execute calls to an operating system as a non-privileged account. These permissions should be verified before operations are performed. <Scheme implementer to insert details here> <Suggestions: TBD >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Data Validation

Number	DVAL-001
Name	*Validate Input*
Description	Malicious data can lead to an array of security vulnerabilities.
Requirement	All data being accepted from users, or other systems should be validated to ensure that it is of the correct type, of an expected length, of the correct syntax (including having no illegal characters (see below)), and if numerical of the correct range. Only data with expected characteristics should be processed. <Scheme implementer to insert details here> <Suggestions: None>
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	DVAL-002
Name	*Canonicalize / Encode Input*
Description	Data can be represented in different ways called encodings. Encodings can be used by attackers to create attacks.
Requirement	All input should be encoded into a normalized form before being processed. <Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Extended (Basic?)

High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	DVAL-003
Name	*Validate Output*
Description	Malicious data can lead to an array of security vulnerabilities.
Requirement	All data being processed should be validated to ensure that it is of the correct type, of an expected length, of the correct syntax (including having no illegal characters (see below)), and if numerical of the correct range. Only data with expected characteristics should be processed. <Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	DVAL-004
Name	*Canonicalize / Encode Output*
Description	Data can be represented in different ways called encodings. Encodings can be used by attackers to

	create attacks.
Requirement	All output should be encoded into a normalized form before being processed <Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	DVAL-005
Name	***Server-Side Validation***
Description	All data validation must take place inside a trusted boundary.
Requirement	All data validation on which security related decisions are made, must take place on the server. <Scheme implementer to insert details here> <Suggestions: None>
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	DVAL-006
Name	***Server-Side Encoding***
Description	All data encoding must take place inside a trusted

	boundary.
Requirement	All data encoding on which security related decisions are made, must take place on the server. <Scheme implementer to insert details here> <Suggestions: LDAP, Web services, XML, HTML, Javascript, anything except SQL. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Preventing Specific Attacks

Number	ATTACKS-001
Name	*Cross Site Scripting (XSS)*
Description	Cross-site scripting (XSS) is a type of vulnerability where malicious code is injected into the web pages viewed by other users. There are typically two types of XSS, reflected where the malicious input is immediately reflected back to a user and Stored where the malicious input is stored to be processed by the system at a later date.
Requirement	All data input to the application should be checked to ensure that it will not lead to cross site scripting vulnerabilities. <Scheme implementer to insert details here> <Suggestions: Implement an anti-XSS library such as the MSFT ACE library>
Criteria Type	Basic
High Assurance	
Medium Assurance	

Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-001
Name	*Cross Frame Scripting*
Description	Cross-frame scripting is a type of vulnerability where malicious code is injected into the frames viewed by other users.
Requirement	Frames can use unique names tied to the session token. As time goes on, this vulnerability class will get smaller due to the fact that modern browsers are not susceptible to cross-frame scripting. <Scheme implementer to insert details here> <Suggestions: Use named frames with session token information. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-002
Name	*SQL Injection*
Description	SQL Injection is a type of vulnerability where malicious data is injected into a web form which can execute database calls, often bypassing other security controls.
Requirement	All data input to the application should be checked to ensure that it will not lead to SQL injection

	vulnerabilities. <Scheme implementer to insert details here> <Suggestions: Use parameterized queries. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-001
Name	*URL Redirection and HTTP Header Injection*
Description	URL redirection is a type of vulnerability where user-controlled data is allowed to provide a redirection. An adversary can then construct a URL which forces a user to go to an unauthentic website. In a similar way, cookies, HTTP splitting/smuggling (on responses in this manner) and other HTTP headers can also be injected through user-controllable input.
Requirement	Redirects and HTTP responses should never allow user-controllable input (especially from the DOM) to change the desired behavior. If user-controllable input is allowed - all data input to the application should be checked to ensure that it will not lead to URL redirection vulnerabilities or HTTP header injections. <Scheme implementer to insert details here> <Suggestions: Do not allow user-controllable input to affect HTTP headers. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low	

Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-003
Name	*Cross Site Request Forgeries (CSRF)*
Description	Cross-site request forgery, also known as one click attack or session riding and abbreviated as CSRF (Sea-Surf) or XSRF, is a kind of malicious exploit of websites. Although this type of attack has similarities to cross-site scripting (XSS), cross-site scripting requires the attacker to inject unauthorized code into a website, while cross-site request forgery merely transmits unauthorized commands from a user the website trusts.
Requirement	Cross-site request forgeries should be prevented by using some sort of state in the browser. While stored (second-order) XSS vulnerabilities that are also present in the application will invalidate modern-day CSRF protections, it is not widely known that reflected (first-order) XSS flaws cannot steal session tokens that are provided in standard CSRF protections. If a CSRF protection measure is used, it should not be turned off on any page where it is required. <Scheme implementer to insert details here> <Suggestions: Use EnableViewStateMac="true" or other anti-CSRF measure, which are normally hidden form fields with a unique session token. >
Criteria Type	Basic Extended: See *Per-Page Tokens*
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-004
Name	*Parameter Tampering*
Description	Parameter tampering is an attack where a malicious user modifies any parameters sent between the browser and the server to modify the behavior of the security of the application.
Requirement	There should be knowledge that user input cannot be trusted, including the values or existence of any parameters. <Scheme implementer to insert details here> <Suggestions: Think about how unexpected user behavior will be handled by the application. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*JSON Hijacking*
Description	JSON, or Javascript hijacking, allows an attacker to bypass the same-origin policy when a web application uses Javascript to communicate confidential information.
Requirement	Requests for JSON data/code should include a session token (or a per-object token) in a similar way as CSRF protections. The application should also only accept requests to JSON data as POST operations. Additionally, when on-site requests to JSON objects are made, invalid Javascript can be inserted to force the client to remove any hijacked

	script tags. <Scheme implementer to insert details here> <Suggestions: anti-CSRF measures, requests as POST only, and by using while(1); or other invalid Javascript. >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*External Entities and Schema Poisoning*
Description	An external entity attacks an application that parses XML input from un-trusted sources. Schema poisoning manipulates the XML schema to alter processing information.
Requirement	XML external entity (XXE) bugs can access local files, remote web services, or even remote websites that a user accesses. Verify that access to local files is inaccessible from XML by setting DOCTYPE, ELEMENT, and/or ENTITY and referencing the entity in the XML body. <Scheme implementer to insert details here> <Suggestions: Utilize a Message Inspector pattern to enforce message-level security. Use trusted documents/schemas with well-known or certified URI's and/or disallow DTD or external entity references, if possible. >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	

Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*SSI Injection*
Description	Server-side include injection can be used to read in arbitrary files where programmatic SSI's are used to read in scripts/files at runtime.
Requirement	Validate all input where server-side includes are performed. <Scheme implementer to insert details here> <Suggestions: Validate input >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*XML Injection*
Description	XML can be injected as user-supplied data in similar ways as HTML or Javasrcipt.
Requirement	XML data injected may also be capable of sending duplicate elements or attributes, which may override the previous values. <Scheme implementer to insert details here> <Suggestions: Validate input. >
Criteria Type	Basic
High Assurance	
Medium	

Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*XML Entity Expansion*
Description	Un-trusted external entities can contain large file references or recursion that can cause an XML parser to eat up resources causing an XDoS (XML Denial-of-Service).
Requirement	DOM parsers will typically load the entire XML document into memory. <Scheme implementer to insert details here> <Suggestions: Use trusted documents/schemas with well-known or certified URI's and/or disallow DTD or external entity references, if possible. Use a SAX based parser or a release that provides performance enhancements. >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*SOAP Injection*
Description	SOAP messages contain XML-formatted data that can be injected in a similar way as XML or XSS.
Requirement	SOAP injection can be handled in similar ways as

	XML or XSS injections. <Scheme implementer to insert details here> <Suggestions: Validate input. Encode XML characters. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	***SOAP Array Abuse***
Description	SOAP data can contain large integers in arrays that can cause XDoS.
Requirement	SOAP array sizes can be specified manually, and do not have boundaries. Setting large array sizes can cause an XDoS (XML Denial-of-Service). <Scheme implementer to insert details here> <Suggestions: Use WS-Security, but also look for attacks against this service. >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	***XPath / XQuery Injection***

Description	Data stored in XML documents are often queried by web applications using the XPath or XQuery languages in a similar way that SQL queries databases.
Requirement	User-supplied input in XPath or XQuery can be problematic as even a single change in one operator can modify the query far from the original intention. <Scheme implementer to insert details here> <Suggestions: Validate input. Blacklist query language operators. Whitelist should be alphanumeric characters only with no whitespace. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

c

Number	ATTACKS-00X
Name	*Mail Command Injection*
Description	User-supplied data (especially email address fields in forms) may eventually be used for execution of email commands, including SMTP messages.
Requirement	Validate user-supplied information before executing any mail commands. <Scheme implementer to insert details here> <Suggestions: Check for CR/LF's. Match email addresses to specific regex. Data containing a period or other SMTP command character should be stripped or disallowed. >
Criteria Type	Basic
High Assurance	
Medium Assurance	

Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Comment [S6]: Is this the same as the general case of command command injection?

Number	ATTACKS-00X
Name	*Buffer Overflows*
Description	Buffer overflows allow arbitrary execution of code by overwriting memory from a buffer that was incapable of handling all of the user data sent to it.
Requirement	Buffer overflows may attack the stack or the heap, and may be a simple (but hard to detect) "off-by-one" error. Any section of memory can be used for code execution, so exploitation countermeasures such as NOEXEC on stack, or ASLR, may not be enough to prevent this attack. <Scheme implementer to insert details here> <Suggestions: Statically check for these errors and code around them. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*Integer Vulnerabilities*
Description	Integer vulnerabilities are typically a compiler problem that is affected by an arithmetic calculation.
Requirement	Integer overflows and signedness errors are the

	primary issues that typically result in a integer vulnerability. <Scheme implementer to insert details here> <Suggestions: Statically check for these errors. >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*Format String Vulnerabilities*
Description	Format strings are parameters in use by the C printf() functions that can be modified by user-controllable input.
Requirement	When an adversary controls a format string, memory can be overwritten in order to execute arbitrary code. <Scheme implementer to insert details here> <Suggestions: Statically check for these vulnerabilities. >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X

Name	*Forced Browsing*
Description	Users can browse to a predictable resource location in order to access a file, a directory of files, or other object that is supposed to be otherwise protected.
Requirement	Predictable resource locations (PRL's) are usually accessibly by anyone who can reach the application, so there must be a protection in place to prevent brute-force discovery of a PRL. <Scheme implementer to insert details here> <Suggestions: Implement access controls. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Comment [S7]: This part of authz, well move it inside the general auth

Number	ATTACKS-00X
Name	*Path Traversal*
Description	Path traversal allows use of the application to reach files or directories locally accessible on the web server that hosts the application, usually outside of the web-root directory.
Requirement	User-submitted data can affect what files and directories are reachable from the application. If required –proper decoding, canonicalization, allowed file types, and/or web-root file system virtualization/location should be implemented in order to prevent path traversal attacks. <Scheme implementer to insert details here> <Suggestions: Validate input whenever touching the file system API. >
Criteria Type	Basic
High Assurance	

Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*HTTP Splitting*
Description	HTTP splitting forces the originator of HTTP messages to send two (or more) valid messages instead of only one.
Requirement	<Scheme implementer to insert details here> <Suggestions: Sanitize CR's and LF's. Use SSL. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*HTTP Smuggling*
Description	HTTP smuggling forces the originator of HTTP messages to send a stream of data that can be interpreted in more than one way.
Requirement	<Scheme implementer to insert details here> <Suggestions: Sanitize CR's and LF's. Use SSL. >
Criteria Type	Basic
High Assurance	
Medium Assurance	

Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*Command Injection*
Description	Scripts or applications allow injection of commands to the web server hosting the application.
Requirement	If applications do not perform shell execution of commands then usually arbitrary users can also not perform these sorts of OS commands. If user-submitted input must be taken into the userland underlying OS through a web application, every conceivable metacharacter and whitespace should be blacklisted, with specific sets of alphanumeric characters as whitelisted input. <Scheme implementer to insert details here> <Suggestions: Validate input. Java API can use Runtime.exe, while ASP.NET can use Process.Start >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*Local File Inclusion*
Description	Include files can be loaded based on user-controllable data from local files on the web server hosting the application.

Comment [S8]: Merge these two into one

Requirement	PHP, ASP, and other languages that allow dynamic execution can allow include files to be loaded locally. <Scheme implementer to insert details here> <Suggestions: Validate input. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*Remote File Inclusion*
Description	Include files can be loaded based on user-controllable data from remote locations.
Requirement	PHP and other possibly other languages that allow dynamic execution can allow include files to be loaded from a remote location. <Scheme implementer to insert details here> <Suggestions: Validate input. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	ATTACKS-00X
Name	*Denial-of-Service*

Description	A denial of service attack prevents users from accessing the application.
Requirement	<Scheme implementer to insert details here> <Suggestions: A rate-based IPS (RBIPS) such as Radware APSolute or Cisco Guard can be used to block TCP/IP, HTTP, and application resource attacks >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Comment [S9]: Avoid specifying a architecture or product

Number	ATTACKS-0BB
Name	*Timing Attacks*
Description	One standard, single component should be used for all failed login response.
Requirement	A standard message should be applied to all users regardless of reason for the failure of the login. If one component is responsible for the response, whether or not credentials are checked – there should be a standard time delay and regulated response from the server/application. <Suggestions: TBD>
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Data Protection (Transit and Storage)

Number	DATAPROT-00X
Name	*Valid SSL*
Description	
Requirement	<Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	DATAPROT-00X
Name	*Extended Server Certificates*
Description	
Requirement	<Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	DATAPROT-00X

Name	*Mutual Authentication Certificates*
Description	
Requirement	<Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	DATAPROT-00X
Name	*Mask Password Fields*
Description	Password fields can be masked so that other users cannot shoulder-surf a user's password while he/she logs into the application.
Requirement	Note that passwords may end up in the source or HTTP request headers. <Scheme implementer to insert details here> <Suggestions: Use asterisks or similar cover for passwords as input. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Comment [S10]: Move to user management?

Number	SECMON-00X
Name	*Logging*
Description	Security related events can be recorded for providing an audit trail.
Requirement	Most web servers can be configured to store HTTP Referer data in the headers, as well as full POST operations. System and database logs may also include information useful for auditing web applications. <Scheme implementer to insert details here> <Suggestions: Use Mod-Security with audit_log in web server module or reverse proxy mode. Application should also provide custom logging. >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	SECMON-00X
Name	*Operator Event Notification*
Description	Certain security-related events should notify operators
Requirement	<Scheme implementer to insert details here> <Suggestions: Notify administrators on critical security events. >
Criteria Type	Extended
High Assurance	

Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	SECMON-00X
Name	*Operator Exception Notification*
Description	Some application unhandled exceptions should notify operators.
Requirement	<Scheme implementer to insert details here> <Suggestions: Notify operators on critical unhandled exception errors. >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	SECMON-00X
Name	*Log Event / Exception on Threshold*
Description	Incrementing events and exceptions should be logged with detail.
Requirement	<Scheme implementer to insert details here> <Suggestions: Crash dumps and exception information should be as detailed as possible and other network and system information should also be gathered if possible. >

Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	SECMON-00X
Name	*Logout User on Threshold*
Description	Large amounts of events/exceptions caused by one user in a certain time period should force the user to logout and re-authenticate.
Requirement	<Scheme implementer to insert details here> <Suggestions: Increment counters for security events or application exceptions and logout a user who causes them after a threshold is met. >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	SECMON-00X
Name	*Disable User on Threshold*
Description	Large amounts of events/exceptions caused by one user in a certain time period should disable the user or suspend their account.
Requirement	<Scheme implementer to insert details here> <Suggestions: Increment counters for security events

	or exceptions and disable or suspend a user who causes them after a threshold is met. >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Information Disclosure

Number	INFODISC-00X
Name	*Remove Debugger Flags*
Description	Compiled applications should have debugger flags removed when live.
Requirement	<Scheme implementer to insert details here> <Suggestions: Remove debugger flags or strip binaries. >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	INFODISC-00X
Name	*Generic Script Errors*
Description	
Requirement	<Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	INFODISC-00X
Name	*Generic Application Exceptions*
Description	
Requirement	<Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	INFODISC-00X
Name	*Generic Server and Database Messages*
Description	
Requirement	<Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Comment [S11]: Same as below

Comment [S12]: Remove as its security obscurity

Number	INFODISC-00X
Name	*Strip Banner and Header Information*
Description	
Requirement	<Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >

Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	INFODISC-00X
Name	*Prevent Fingerprinting*
Description	
Requirement	<Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	INFODISC-00X
Name	*Reference Objects Indirectly from Client*
Description	
Requirement	<Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low	

Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Comment [S13]: Consider remove, doesn't buy you anything

Number	INFODISC-00X
Name	*Remove HTML and Javascript Comments*
Description	
Requirement	<Scheme implementer to insert details here> <Suggestions: MAC address or some other ID form >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Miscellaneous

Number	MISC-00X
Name	*Security Statement*
Description	Security statements provide clear guidance to users about their security expectations and how to report potential security issues.
Requirement	All pages should contain a link to a security statement that clearly defines the security measures used, users expectations of security and clear instructions about how to report any security issues. <Scheme implementer to insert details here> <Suggestions: In conjunction with the OWASP Legal project we should develop a sample statement >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	MISC-00X
Name	*Security Statement (Extended)*
Description	Security statements provide clear guidance to users about their security expectations and how to report potential security issues.
Requirement	The security statement should provide a clear service level agreement detailing when a user reporting a security issue will expect to receive acknowledgement and detailing the longest and average times the site has taken to address security issues in the past. <Scheme implementer to insert details here>

	<Suggestions: In conjunction with the OWASP Legal project we should develop a sample statement >
Criteria Type	Extended
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Number	MISC-00X
Name	***Privacy Statement***
Description	Privacy statements provide clear guidance to users about their privacy expectations and how to report potential privacy issues.
Requirement	The privacy statement should provide a clear service level agreement detailing exactly what data the site will hold, who it will disclose it to and under what circumstances. <Scheme implementer to insert details here> <Suggestions: In conjunction with the OWASP Legal project we should develop a sample statement >
Criteria Type	Basic
High Assurance	
Medium Assurance	
Low Assurance	
Very Low Assurance	
Evaluation Notes	<To Be Completed By Inspector>
Score	<Pass or Fail : To Be Completed By Inspector>

Appendix

Reference Implementation – The OWASP Evaluation Criteria

Sample Completed Scorecard

Sample Certificate