SECURE WEB APPLICATIONS DEVELOPMENT GUIDELINES

LEAN SECURITY

www.leansecurity.com.au

LEAN SECURITY
Web and Mobile Application Security Assurance

CONTENTS

LEAN SECURITY
Web and Mobile Application Security Assurance

INTRODUCTION

Web application pertains to an application program, which is kept on a remote server and transferred over the Internet by means of browser interface. Security is an important portion of the web application; therefore, through understanding and executing accurate security procedures, you can protect all your resources together with providing a secure environment wherein the users can be comfortable in working with your apps. This secure web applications development guideline is composed of technical and procedural control statements. The control statements show security requirements, which must be adhered in order to develop a web-based application directed for administration. The control statements contained in this guideline have been categorized into appropriate sections like Web Access Authentication and Control, Auditing and Logging, Error Handling, and Session Management.

Thus, before utilizing this security guideline, it is crucial to learn some of the vital security aspects that this technical guideline addresses:

- Security is derived from the analysis of threat in order to guarantee effective and accurate-for-purpose controls. Security is far different from simple categorical scenario for all the applications since this can directed to wrong controls or insufficiency in controls being established.
- Chain can be strong like its weakest link. The web applications are not separate as it interacts with other multiple back-end systems.
- Web Applications are portals wherein the critical mission data and applications can be taken from anywhere.
- Applications Development is naturally concentrated on transporting functionality features. Once the security is considered not part of the developmental process, this can turn costly, time consuming, and ineffective to have it on afterwards.

CHAPTER 1 : ARCHITECTURE OF WEB APPLICATIONS

ARCHITECTURE CONTROLS

- **Resource access**

  Applications should be constructed in a manner that the user-submitted information are not used directly by an application for the access specification to resources like the filenames, environment variables, network addresses, database records, and Operating System commands.

- **Fail safe components**

  The components in the system should be created for "fail safe" in order for maintain the confidentiality and integrity of the data during failures.

- **Database access**

  Database access originating from application code should utilize parameterized statements. Considering that, SQL directly constructed from the user input should not be utilized since it may indicate possible breach of security.

- **Web Site Partition**

  Internet facing website design should differentiate clearly between the publicly available areas and the restricted areas, which asked for authenticated access. Utilize distinct subdirectories under the virtual root directory of the application to keep the restricted pages, like the checkout functionality accessible in the standard e-commerce website that needs authenticated access and conveys sensitive data. The distinct subdirectories tolerate incorporating additional security without suffering SSL performance over the whole site.

- **Access on Administration pages**

  ALL the administration pages should be access controlled through strong password or by a two-factor strong solution authentication. The SSL encryption should be used always for the administration pages.

- **Sensitive data storage**

  Sensitive data for account codes, user names, balances, and transactional data, must be stored only in a determined method of protected backend tier. Once it is essential to store temporarily a duplicate of sensitive data in the application server tier; therefore, the additional controls should be taken in maintaining the integrity and confidentiality of the data.

  On the other hand, sensitive data like authorization, personal, and financial data should be kept on the server-side. The reason for this is that the client-side storage like the DOM storage, cookies, and flash storage are not suitable for such data. Additionally, never trust the input parameters once they are utilized by the server for making invoice or security decisions.

  Backend Databases Servers utilized for the storage of user or business information should not be allowed to create an outward internet connection. Specific Firewall rules should be applied for filtering out the outgoing attempts performed by databases-servers and for activating suitable security alerts if attempts are throw down. The Backend database servers should not be unveil to the internet by a public IP.

- **Deprecated APIs**

  The unsupported or deprecated APIs should never be utilized.

- **HTTP redirections**

  The HTTP redirections should be secured to certain domains and certain parameter values. This is because when the arbitrary destinations are permitted, this enables attacks like phishing. This can be resolved:

- By means of white-list of permitted destinations;
- Through checking integrity mechanism like the combination of target redirection domain value by MAC, which can be authenticated by the application; or
- Through mapping destination parameters via server-side lookup with the target URL.

- **External networks presentation**

  Functionality of application administrative should not be presented to the external networks like the Internet. Once the access towards administrative interface on the Internet is needed, this should be constrained by the IP address and the default admin URL must be altered.

- **Client-side technology**

  The client-side technology should never be trusted upon the enforcement of security before sending to the server, or for the protect secrets like keys, credentials, or delicate business information. The basis of this is that an attacker may inspect or change the client code to dodge security checks or take the secrets. Moreover, it is tolerable to take the client-side validation or the user interface constraints on the client-side for usability and performance purposes. Thus, these cannot be taken as security features since these features can be taken on the server-side so that it will be effective.

- **Documentation of Data Flows**

  Data flows should be documented and parts where data moves throughout trust boundaries should be highlighted.

- **Limitation of Meta-characters**

  It is crucial to limit the Meta-characters to the portions of the application demanding them. If such characters are needed, they should be neutralized by means of escaping or encoding before being functioned on in the setting where the special meaning occurs.

- **Application Capability of Switching Off**

  Applications should have the capacity to switching off or limit functionality, which is potential to be under continuous attack. The granularity feature and level of limitation should be identified for every individual application, knowing the business necessities and threats faced. Once the attackers or the fraudsters recognize faintness in the system, they commonly transfer quickly to develop maximum value prior to the closing of vulnerability. This activity can create losses to increase rapidly since the system drops under continual attack.

- **Passing of Data parameters**

  Data or parameters should not pass from side to side between the server and the client except the data is needed on the client-side for execution. Keep data in the session item on the server-side where practicable rather than depending on the hidden sections and other browser-side storing mechanisms.

CHAPTER 2 : WEB ACCESS AUTHENTICATION AND CONTROL

AUTHENTICATION CONTROLS

- **Accessibility of Restricted content**

  Restricted data or application functionality should only be accessible towards legitimate users.

- **Accessibility on Login Pages**

  Web registration and login pages should only be obtainable through HTTPS. This will guarantee that the login is executed over a safe channel, avoiding any tampering or sniffing attacks. Remember that this point exactly addresses the exercise of holding login form on HTTP page, irrespective if the form POSTs the demand to HTTPS target.

- **Transmission of Login Credential**

  Web login credentials should be transmitted only through POST request. The GET method must not be utilized for the transmission of any delicate data. The reason behind this is that, the IDS systems, load balancers, web proxies, and application servers, may contain a record of URL and GET restrictions leading to disclosure of credential.

  In addition, the data transferred through GET may be leaked over the browser history tool and when third party hosted content implanted in the website is demanded. For the authentication of application level, client login credentials should be transferred by means of HTTP POST procedure. GET method should not be utilized.

  Certainly, GET method need less understanding and skill by the malicious user or attacker, as majority of the browser applications displays the URL comprising information relevant to GET request. Moreover, information presented in the shown URL can be bookmarked and stored locally; however, effectively caching login credentials for other user of the system of the client.

The URL's commonly logged in the access log of the web server, together with the disk cache and client browsers history file. However, it should be noted that the usage of the POST procedure is NOT a complete guarantee, for the reason that some complex attacks are potential through a determined attacker.

- **Web Login Forms Auto-Completion**

  Web login forms should possess an Auto-completion turned off with all credential sectors. This must be coded explicitly into the web form. With shared workstation setting, this will guarantee that the delicate data is not kept in a determined form, which succeeding users may access. Furthermore, browser username or the password databases are normally targeted for attack via browser exploits or Trojans.

- **Accounts lockout**

  Authentication systems should lockout accounts once a configurable number unsuccessful to login attempts. This for the authentication schemes to avoid brute force of credential guessing attempts.

  Additionally, the configurable number of the unsuccessful logins should be lower than or equivalent to 10. Credential lockout systems should not be susceptible to inconsequential attacks on Denial of Service. Since the usernames are commonly guessable by means of brute force, large quantity of accounts may be locked out in a programmed method if naive lockout structure is utilized. If the lockout occurs, the application should not provide any indication that the lockout occurred. Hence, locked-out user should be informed through SMS or email message in order to notify them of the attempt of hacking into their account. Lockout mechanism should also be sustained with strong unlock mechanism, like the automatic unlock subsequent to a configurable period and/or reset password functionality.

Option for Logout

All the authenticated application website pages should offer a logout option that is easy to locate.

Login Failure For Generic Error Page

In a login, once any of the credentials provided for login are incorrect or unexpected error was trapped, then the behaviour of the application must not show invalidity or validity of either the password or username. The behaviour of the application should be reliable and similar generic error page should be indicated irrespective of which feature of the credential was improper. Thus, should the client fail to transmit the appropriate credentials and fail the conforming authentication method, no information must be sent back to the client showing why the authentication failed. Client should be offered with general message for “Authentication Failure”.

Sending informative messages like the “Incorrect Password” or “User does not exist”, assist attacker to determine user accounts and guess the passwords.

Last Login

The initial page that user see after positive login must advise possibly the user once they last logged in.

Salted And Hashed Credentials

Customer fixed credentials, like the passwords, should be kept as one-way hashes, instead of a reversible encoding. Salt should be supplemented to the password, before hashing for limiting the efficiency of the offline password attacks. The underlying principle here is that, keeping the passwords salted hashes improves the password difficulty for pre-computation attacks that would then permit the retrieval of the password for clear text form. The secure method of implementing this control is through Password-Based Key Derivation Function V2 (PBKDF2).

In this structure, password is combined with salt, and repeats several times for hash scheme. The result of the PBKDF2 operation is kept together with the count iteration and salt. If verification of password is needed, operate the PBKDF2 through the provided password and stored salt and iteration count. Relate the result of operation with stored outcome password verifier.

The condense approach implies that you never really store passwords. Rather, you take the password coming from the user and authenticate it through combining with the value of stored salt, recounting the hash and relating it with stored hash. Strong hashing algorithm like theSHA1 must be utilized. Where potential, salts must be kept separately with password hashes, in a separate database or flat file. Once the values of the salt are taken together with password hashes; therefore, they do not alleviate the risk made by dictionary attack or brute force. The control must be applied depending on the assessment of risk.

- **Management of User Account**

Administrators should be capacitated to prevent, “permanently” ban users, or inactivate the user accounts. This is because there are several reasons for account suspending, involving abuse and customer reporting suspicions about their compromised credentials.

- **Authenticated Changes On Sensitive Information**

The method that allows customer to alter sensitive information in their account must involve authentication challenge. The principle behind this is that, an attacker having the capacity to change sensitive data related with a customer could utilize this to take higher access towards the financial assets or identity of the customer. Additionally, sensitive information in this context involves personal data and used data within the model security. The customers also need to have an email confirming that the details were altered.

Change of Credentials

Where a user alters their existing credential by means of self-service, user should supply the current credential in connection to the process of credential change. This stops attacker to have a temporary access over a session; however, not a credential, coming from a new credential setting.

Legitimate Transactions

Where there is necessary threat of copying attacks, application should authenticate if the value transaction is genuine. Reliant on the risk at financial asset, authentication transaction will need the step-up or re-authentication to multiple authentication of transaction initiator.

CHAPTER 3 : NETWORK SECURITY

Network security is one of the important components of the entire IT security features. This guarantees that anxieties in single node are not transferred easily to, or possess significant damaging effects with other nodes.

CONTROLS ON NETWORK SECURITY

- **Least Privilege**

  The norm of least privilege orders that admission to resources should be restricted tightly as possible, through certifying that systems and users have the least traditional privileges needed to conduct their role. The least privilege should be taken to the network architectures for the reduce amount of the infrastructure of organisation wherein data may travel, lessen the quantity of points that have access towards the networks of the Organisation and which are obtainable from the provided point, and reduce the possible surface attack of all infrastructures.

  A dedicated account with the required and minimal level of privileges should be definite for operating the server-side application and retrieving the backend services and databases. In connection with the user level, the user impersonation should be performed to control the granular resource admission. Make sure that the sensitive information like the credentials of service user and connection strings are not kept under a plaintext configuration files.

GENERAL RULES ON DEFENCE IN DEPTH

A defence-in-depth method should be accepted in order to offer greater flexibility to the network and/or compromise system. This method aims to minimize the impact and risk and of security breach, by means of multiple implementations of layered controls to:

- Defend against attacks.

- Spot attacks when it occurs.
- React properly to reduce impact and recuperate from incidents.

- **Isolation and Separation**

  The design of networks should offer the systems separation through the purpose and access standards. The systems separation offers ease towards management and eliminate the possible effect of security breach through limiting the effects of extra attacks and separating the individual domain systems.

  Rationale: dividing components across separate networks ensures that each layer or segregated network will be protected in the event of a security breach or penetration.

- **Web Application Development**

  This section presents the security controls concerning about the general advances of all web applications together with the security controls relevant to the input data validation within the entire web applications.

DEVELOPMENT CONTROL

- **Due Diligence**

  If development work is performed with third parties, the continuing due diligence should be carry out to confirm the development security of environment and procedures.

- **Secrets Change**

  In the growth of live solution, all the secrets- the passwords and keys, related with an application should be altered. All the credentials and test accounts are to be eliminated and debugging selection must be deactivated.

The objective of this is to help confirm that source code compromise or development and system tests will not lead for direct compromise of production systems. Agency is liable for confirming that passwords are diverse on production and staging environments. The agency must also have to ensure that all credentials are preserved and secure all the time. Logins on shared should be eliminated. If staff requires exclusive admission to the backend application, the account should be arrange and associated exclusively with certain staff member. Accounts carrying privileged access should be eliminated if the related staff member is not anymore working in the agency. Keys and passwords should not be transported in an unprotected way like an email.

- **Version-Control System**

  All codes should be kept in a private access-controlled, managed, and version-control system.

- **Administrative Access**

  No developer must have the system administrative access towards the master system version-control. The principle of this is that, there is separation of duties relevant to this. Even though the Lead Developers can have the administrative access towards code branches, no developer should possess administrative access towards the host operation of the master system version-control or the administrative control over the code repository.

- **Version Check**

  When the developers check their alteration with the version-control system, performed message should recognize them unmistakably together with the made changes.

- **Staging System Configuration**

  If a staging system is utilized for concluding functional and testing security, this system should be organized in order to be functionally similar with the production system. This involves the security controls, like the SSL certificates, password policies, and traffic encryption.

- **Security Model**

  All the applications should carry a security model that is documented.

- **Changes in Production Code**

  Changes towards production code should be linked with bug report or feature or change request. Moreover, the changes on codes should be governed by the version-control and alteration control systems prior to making the new code live. Emergency alteration might require to be conducted retrospectively by these systems.

- **Versioning of Software Dependencies**

  The software dependencies like the Virtual Machines and libraries should have similar number version in the development, production, and test. Similarly, the compiled code should be collected using similar version of compiler number in development, production, and testing. The real dependencies implementations may vary during development, sine the development systems occasionally utilize different stages to those of production and test.

- **Protection of Source Code**

  The Source code on manageable systems like the laptops should be secured with complete disk encryption. Principle governing this is that, this will confirm that codes are not in cases of theft or physical loss on the development systems.

- **Development Server's Access**

  The access towards the development servers should be constrained.

- **Design walk-through**

  Web Application development stage must involve a planned walk-through of design to guarantee that the system performs what it is made expected for, and that it should not perform anything that far from its scope. This can be assisted by peers not included in the structure under review.

- **Code reviews**

  The code reviews, automated or manual or automated, should carry out for the applications with the security function or sensitive business.

- **Test plan**

  Test plan and the associated scripts should be established, which involves functional testing, security testing, and implementation testing. Preferably, these documents must be kept under similar document management system or version control like the documentation of base development.

CHAPTER 4 : FILE UPLOAD CONTENT

The following are some of the security controls that are primarily related to the process of uploading files by web applications.

FILE UPLOAD CONTENT CONTROLS

- File Upload Transmission. It is necessary that all the files should be uploaded over the secure channel, which in the first place guarantees confidentiality.
- File Size. The size of the files must always set a certain limit on maximum size of the file. Moreover, this should always be a configurable perimeter. When the Internet websites or the Intranet primarily accept the uploaded files, check the file type, file size, and if you passed the two sets, the next one would be the Anti-Virus check.
- File Location Access.
    a. There is a need the clients must not be able to access directly to the uploaded files because the access to the uploaded files should be primarily managed by the application itself.
    b. Both the location and the name of the files must be primarily controlled by the application and would not depend on the user supplied value including the document metadata, MME part header. In addition to that, when either the internet or intranet application primarily needs to accept the files names as the input, there is a need that the file path as well as be name should be properly checked in order to ensure that the file system name along with the path re valid in the application. This would help the users to avoid successfully supplying all the paths that primarily contains “..” to transvers the outside of virtual directory hierarchy of the application.

- Filetype Specification. The application must only allow upload files of the allowed filetypes such as zip files. Moreover, the application must now allow upload of the executable filetypes such as the PDF and only the data files may be uploaded. In addition to that, gifs, jpgs pngs, txt are some of the acceptable filetypes.
- Uploaded Files Permission. There is always a need that the uploaded files should be created with the non-executable permissions ate OS level, so it would be better if the file system does not primarily low the file execution.
- Virus Scan. Upon uploading, the unencrypted files should be virus scanned and the proper antivirus solution must be primarily implemented on the server in order to protect from the viruses.
- Validation. There is always a need that all the stored data in the files that would be processed by application should be validated.
- Files Encryption. All the files that contain sensitive data spooled or stored temporarily at the application server tier or web should always be encrypted.
- View Authorization. The uploaded files should only be presented to the authorized for viewing all the files.
- File Download. Content-Disposition Header must be set for the previously uploaded files, which are downloaded subsequently. This is responsible for forcing the browser in order to offer save and open dialogue box. Moreover, the safe default filename should always be specified.
- File Size Limit Check. Check the maximum limit size against the actual uploaded data. All the uploaded files should be aborted upon reaching the value such as the applications should perform size check during the process of uploading and not after completing the upload.
- Filenames. The automatically generated filenames should not be predictable. Moreover, filenames should adhere to the pre-defined and non-executable, and extension names.
- Temporary Files. Either the transient or the temporary files must be removed from file system right after the successful processing and within 48 hours once the processing is not that successful.

- Files Presentation to the Users. The file upload should not be directly presented to the users within the browser and the uploaded data should always be presented as either clickable links o process data validated as safe.
- Archives. If the archives are permitted, there is always a need that the file upload control should be applied to the content that is extracted. Moreover, there should always be a configurable limit right to the number of the nested zip files, which the archives could primarily contain.
- Integrity Check. There is a need that the applications should perform integrity check to the file stored prior to the process of processing to ensure that no files are altered in storage.

CHAPTER 5 : EXTERNAL DEPENDENCIES

Here are the security controls primarily relating to links between the web applications and the remote systems or the third-party codes.

EXTERNAL DEPENDENCIES CONTROLS

- Code Documentation of Third party. The third-party code usage as the part of the application should always be documented. Moreover. This includes the detailed purpose, support model, version, as well, as how all the software was integrity checked or sourced.
- Code Verification of Third party. This should be verified as secure and rusted as the part of the application. For commercial components, it is the vendor's duty to provide accurate evidences that they already perform source code view.
- Script hosting of Third party. This should be directly hosted by web application and must not be hosted by any third party. Moreover, this should not also sourced dynamically upon the session of the users. If the scripts are primarily tampered with source or by spoofing, the security of the application could be undermined.
- External Content. Contents including the banner ads should not be sourced coming from the domains that are untrusted.
- External dependencies Control. There is a need that the code must not be loaded from the remote systems. Either the IT Security or Group IS must approve the third party libraries primarily loaded by JavaScript.

CHAPTER 6 : CONFIGURATION MANAGEMENT CONTROLS

Here are the security codes relating to web applications configuration.

DNS MANAGEMENT

The Organization Domain Names should be managed in accordance to Organization Domain Name Standard.

MANAGED VS. UNMANAGED HOSTING

Both the E-commerce and the data collection assets should be hosted primarily in the Organization environments.

GENERAL CONFIGURATION CONTROLS

- Minimal Privileges. All the user accounts related application must contain minimal privileges needed for the operation within the host OS as well in other systems. In addition to that, the application must also contain minimum acces to host system mainly required to the functionality and most of the applications available would function even without the Windows Registry Access, Sending Raw TCP/UDP, access to other applications in the wed running on same host, and file system that is access outside application virtual directories.
- Configuration Files Accessibility. Both the static content and the configuration files must be read-only for either user accounts of the web application or the web server.
- Accepted HTTP Methods. Servers should only accept HEAD HTTP, POST, and GET by default while other HTTP methods could only be utilized if required by the application.

- File Upload Mechanism. File uploads mechanism including WebDAV support and MS FrontPage extensions should be disabled if not required by the application.
- Custom Error Pages Creation. This must be created for the use of application and web servers that primarily generate the responses of the customers and there is a need to remove the default error pages.
- Accessible File Types. Configuring the web server allows access to the required file types including the jpg, jsp, and css. The file type request should return customized error page.
- Required Components Deployment. There is a need ha the listing of the directory must be disabled on both the application and web servers.
- Environment Passwords Storage. The environment passwords should not be hardwired into the application instead they should be stored in the configuration file or database that are primarily located outside address space of re servers. Secrets including the cryptographic keys or passwords should never be hardcoded to the application instead they should be placed at the right configuration container.
- Unused Functionality Removal. Deployed components primarily offer the unused functionality that must be disabled or removed.
- Compliance with the Security Standards. The systems hosting web applications should comply with the important standard of the host security.
- Debug Code Promotion. All the processes should be in place in order to ensure that the debug code is not promoted to the production except during the diagnostic efforts that are undertaken by normal change management channels.
- Configuration Data Storage. This must be stored external to the web or application server content areas. Storing the data under the webrobot and other similar locations would make it accessible to the attackers.
- Compilers Deployment. No compliers should be deployed on the systems' production.

- Web server. This should only return the web pages in just a response to the requests of HTTP that primarily contains right host header. The interpreted server-side code should be pre-complied upon the deployment.
- Secrets Encryption. Secrets should be primarily stored in configuration of the application in the encrypted format.
- Clickjacking Protection. There is a need that the webs server must be configured in order to prevent the attacks of Clickjacking. This new class of attack primarily involves the attacker loading the target site in the transparent type of frame right over the site of the attacker. In order to prevent this type of attack, it is needed that the hosting provider should configure the header of X-Frame-Options on web server and then set it to SAMEORIGIN or DENY.
- Cookie Usage Control

COOKIE USAGE CONTROL

- Sensitive Data. Both the personal and sensitive data should not be placed in the cookies in a form of plain text.
- Secure Flag. There is a need that cookies should be marked with secure flags if there is explicit business requirement for transmitting over the HTTP. Moreover, cookies that have sensitive data including session identifiers should be marked as secure ones without any exception.
- Cookie Lifetime. It is necessary to limit cookie lifetime in order to reduce time window where the attackers could utilize captured cookie necessary for gaining a spoofed access to a certain application.
- Persistent Cookies. Both the session and authentication cookies should not be persistent and the authentication and personalization cookies should be separated.

CHAPTER 7 : CRYPTOGRAPHY

Here are the security controls relating to the utilization of cryptography in the web applications.

CRYPTOGRAPHY CONTROLS

- Cryptographic Functions Implementation. Neither the developers nor cryptographic algorithms should not implement the cryptographic functions. This helps in the process of minimizing risks of the weaknesses in cryptographic implementations.
- Keys Storage. It is needed that the application keys should not be embedded in the application itself. The use of the hard code keys or the labels primarily adds the complexity to key management and at the same time increases risks when it comes to the key exposure. In addition to that, this could also have major impact on system security and so creating key changes is very difficult.
- Cryptographic Functions Usage. This should always be used correctly. If the message is very small for cryptographic function, this would be padded appropriately. The random number generators should be seeded with enough entropy as well as re-seeded the accurately.
- Key Comprise Plan. This should be in place in order to recover the security of the system when the key compromise or there is a failure in the cryptographic algorithm. The areas that need to be considered include the identification of the compromised credentials or data, remediation of the compromised credentials or data such as the re-issue of the credentials or data re-encryption. Moreover, the data rollover to the new key and the business impact of the activities should also be considered.
- Secrets and Keys Protection. The cryptographic private or secret keys should be appropriately protected. The requirement for the cryptographic techniques as control primarily necessitates protection of cryptographic keys that underpins them. Moreover, the management process for the cryptographic keys would help in ensuring cryptosystem's integrity.

- Protected messages integrity check. If you would use the encryption in order to protect the message content, there is a need that the ciphertext should undergone integrity-check. The absence of the integrity checking would allow the attackers to alter content of the encrypted massage.
- Predictable date fields reduction. Encrypting the messages with the use of Cipher Block Chaining mode requires placing the fields with the high level of entropy right at the start of every message. This helps in facilitating the traffic analysis as well as the subsequent cryptographic attacks.

CHAPTER 8 : ERROR HANDLING, AUDITING, AND LOGGING

Here are the security controls related to the process of error handling auditing, and logging within the web applications.

CONTROLS FOR ERROR HANDLING

- Notification. If the application or the system errors, the user would be notified and they should be presented with the generic system error page, which in the first place does not primarily reveal error causes or expose the details regarding the application's internals.
- Exceptions handling. The application containers should definitely handle the exceptions, which are not caught by the lower level. The exceptions should not be propagated to clients because they may become prone to further system attack by means of providing a detailed type of information on architecture of the application.
- Avoid Cascaded Failures. Do not cascade failures. When the exceptions are handled, code should be structured in a way that exception does not primarily allow the tests or the checks to be bypassed. There is a need that the tests mainly fail by the default and exceeds in which specified tests successfully completes.
- Custom Error Messages. Configuring the web server error response mechanism would provide the clients custom error messages.
- Web Server HTTP Status Revelation. HTT status revealing the error conditions on server must not be set by the web servers.
- Deep Exception Handling. The exceptions should be handled at its appropriate level of depth right within the application and the thrown exceptions should be specific. Exceptions should not be primarily propagated to clients because they could aid system attacks by means of providing information on architecture of the applicant.

LOG MANAGEMENT CONTROLS

- Sensitive Information Logging. The sensitive information including session ID's and passwords should not be logged. There are instances that the details may log, but the most sensitive parts should be sanitized upon logging.
- Logged Information. Web applications should log the date, time, and user ID, transaction or action type, has of the session ID, IP address of the client, return code, error code, user agent, referred URL, accept language, and accept encoding for tracing the intrusion attempts as well as auditing.
- Attack Code. Logs that contain the supplied data of the user may have functional attack code. Mitigating the threat would primarily require data suitably encoded or mechanism for viewing the logs should be immune to attacks.
- Events Logging. Failures and success should be logged in security-related events including users' login and log out, critical transactions, failed attempts in logging in, account lockouts, and policy violations,

AUDITING CONTROLS

- Integrity Protection. Integrity protection must be utilized for logs required for purposes of auditing.

CHAPTER 11: SESSION MANAGEMENT

Here are the session management related security controls in web applications.

There is a need that the organization applications must fulfill the OWASP authentication as well as the session management verification requirements to the Level 4.

Authentication Verification Requirements:

http://code.google.com/p/owasp-asvs/wiki/Verification_V2

Session Management Verification Requirements:

http://code.google.com/p/owasp-asvs/wiki/Verification_V3

SESSION MANAGEMENT CONTROLS

- Unique Session Identifier. Session identifier should identity each user uniquely in order to prevent access to data of other users. This should be unique for each of the logon session to ensure that the attackers could not replay compromised identifier. This should be derived from the cryptographically strong random number sources to ensure that the attackers could not predict identifier. Moreover, the session identifier should be 128-bits in its length.

- Credential Transmission. Credentials should be transmitted with the use of the secure protocol and cookies are utilized as the session identifier container, marking them “secure” would definitely guarantee session identifier is successfully transmitted over the secure protocol.

- Session Identifier Expiration and Timeout. This should expire upon logout because this restricts the chance of session being reutilized right after user has already finished. This should timeout right after the inactivity’s configurable period. Both the timeout and logout should validate identifier on server and the client. Regardless of the activity, session should automatically timeout right after the configurable period. In addition to that, valid request coming from the client should mainly revoke session identifier of users.

- Session ID Content. Make sure that the session ID's are active for the limited period and most of all are dependent on the application's type as well as value of the accessible information. The session ID content is of expected type and size, and so the information quality is verified upon the processing. In addition to that, session ID content should not contain the unexpected information. There is a need to keep the personalization cookies that primarily contain the user-specific preferences as well as sensitive data separate from the authentication cookies.
- Attack. There is a need that the applications should ensure that the requests that would change internal application state have to be originated by users opposed to CSRF or Cross-Site Request Forgery attack.

CHAPTER 9 : PHYSICAL SECURITY

Here are the security controls relating to the physical security in environment in which the web applications are primarily hosted.

Physical Entry Controls

Data Canter Physical Access

This should be restricted to and most of all secured with various authentication methods.

CHAPTER 10: INFORMATION LEAKAGE MANAGEMENT

Here are the controls primarily designed in order to prevent information leakage by the web applications.

INFORMATION LEAKAGE CONTROLS

- Field Auto-Completion. This should be turned off for the sensitive form the fields including email, password, username, and the credit card number fields.
- Sensitive Information caching. This should be prevented because attackers having access to the system of the user could access the cached web pages that contain private information.
- Consumer Data Transmission. Pages that primarily accept, provide, and request a customer data should only be accessible by HTTPS.
- Mixed HTTP and HTTPS. Web pages should not have mixture components of HTTPS and HTTP because this allows attack injection into the secure HTTPS session.
- Sensitive Data. This should be concealed if the users type data in the browser. In addition to that, mother's maiden name or passwords should not be included in web page that is presented to users. The HTML comments should not include information regarding the development environment, application design, source codes, developer names, and a lot more. If the sensitive data is posted it should respond with 302 redirect in order to get response page.
- Group Policy Constraints. Both the consumer data collection and protection should be handled in accordance with Organization Group Policies.
- Legal and Regulatory Constraints. There is a need that applications should honor the legal as well as regulatory constraints about masking or displaying sensitive financial, personal, and other data that is sensitive.

CONCLUSION

We are now living in a highly modernized society, so we can always expect innovations and improvements. Web developers are becoming more and more popular and this is all because of the excellent web applications available today. If you are among the web developers who are struggling when it comes to developing new web applications, this eBook is the one that you definitely need.

This eBook would serve as your ultimate guide and most of all a steppingstone in developing secure web applications that would set you apart from other web developers out there. This is completely packed with guidelines essential for helping you in the process of developing applications that are needed by most.

Considering that it is not easy to beat the competition in the web development industry, reading this eBook would give you knowledge and ideas on where to start leading to success that would definitely satisfy you the most. This is now the best time to show off what you got and let this eBook bring out the best in you.

ABOUT LEAN SECURITY

SECURITY SOLUTIONS YOU CAN RELY ON

For dedicated managed security and IT solutions that are guaranteed effective and reliable, more online business owners are choosing Lean Security over any other internet security firm period. We are the only firm that works laterally with our clients every step of the way to ensure their needs are met and their web applications are secure at all times. When you need a team of experts who will listen and respond to your IT needs, trust Lean Security to show you what we can do for you today.

THE SECURITY SOLUTIONS YOU NEED

Headquartered in Sydney, Australia and serving the international business community, Lean Security was founded under the principle of offering our clients real-world solutions to all of their online business needs. We are more than an IT consultancy, we offer managed security solutions designed to keep your web applications secure and your business running smoothly. We are an Australian owned and operated company and you can be assured that your data is controlled by us, right here in Australia.

OUR PHILOSOPHY

Our team of experienced professionals strive to provide a higher level of service and support that our clients can't get anywhere else. We offer best in class products and rely on our over 10 years of practical security industry experience to provide our customers with truly world class online business solutions.

Lean Security showcases the best value for the IT and online security products and offers our clients a wide range of customizable services including:

- Secure Managed Cloud Hosting
- WAF Managed Service

- Managed Web Application Security Testing
- Managed DDoS Protection
- Managed Event Correlation /SEIM Solutions
- Managed Network Vulnerability Assessments

CHOOSE LEAN SECURITY AS YOUR APPLICATION SECURITY PROVIDER

We at LEAN SECURITY furnish organizations and associations with a simple and savvy method for dealing with the security dangers connected with corporate web and versatile applications. LEAN Security gives oversaw helplessness examining and web application infiltration testing administration. This implies establishing the data cen5teres without any need of equipment or programs to be installed, you can pay as per your need which means you can start with the little and then you may extend if you need more services, it totally up to you. Thirdly there will be so compelling reason to employ and prepare any web IT Security staff. Let our expert group handle all the specialized testing. And yes, you will be having a very simple fixed pricing per application (or per subscription) makes it easier to manage the budget.

At LEAN Security, we will be helping you through:

1. **Our Technology:** We use an assortment of business and open source apparatuses and items to convey the best security administrations to our clients. A rundown of the apparatuses utilized include:

    - **Nessus Vulnerability Scanner:** The most widely organized vulnerability assessment & management solution
    - **Qualys Vulnerability Scanner:** Qualys is a source of cloud security, compliance and related services for small and medium-sized businesses and large companies
    - **Metasploit:** Penetration Testing Software
    - **Netsparker:** Work as a False Positive Free Web Application Security Scanner
    - **SQLMap:** Automatic SQL injection and database takeover tool
    - **Burp Suite:** Burp Suite is an cohesive platform for execution security testing of web applications.

2. **Process:** For getting the services, you don't need to wait some weeks or even months until your website is verified by a security consultancy. Our SLAs are very simple and you get results much faster and provide you the reliability that you will not regret on your decision. Our SLA's provide:

    - **Basic Assessment** - branding web sites and mobile applications without the data collection features in 3 business days only.
    - **Standard Security Assessment** - corporate web sites with data collection functions and simple web applications (reservations, order handling etc.) in 5 business days only.
    - **Premium Assessment and Penetration Testing** - ecommerce applications or complex web applications with multiple roles and privileges in 10 business days only.

3. **Our Consultants:** Our customers will get enthusiastic account managers and project managers to help accomplish the project goals and results. Our team will help you to:

    - Analyze your business necessities and find the clarifications to address your challenges.
    - Generate a security assessment plan to meet timelines
    - Provide inclusive reporting on the position of the project
    - Intensify any issues that need quick and speedy resolutions
    - Track you resources and budget

Moreover, all your projects are checked by our skilled professionals with respected industry certifications like:

- **CISSP** - Certified Information Systems Security Professional
- **CISA** - Certified Information Systems Auditor
- **CISM** - Certified Information Security Manager
- **GPEN** - GIAC Penetration Tester
- **GCIH** - GIAC Certified Incident Handler
- **GWAPT** - GIAC Web Application Penetration Tester
- **GXPN** - GIAC Exploit Researcher and Advanced Penetration Tester