Volume 5, Issue 12, December 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Securing Web Applications from Code Injection Attacks by Implementing Provisions of Information Security Standards

Ashish Ukidve
Principal, Vidyalankar Polytechnic, Mumbai, India

Ds S S Mantha
Ex-Chairman, AICTE
Professor, VJTI, Mumbai India

Milind Tadvalkar
Director, Vidyalankar Dnyanapeeth Trust, Mumbai, India

Abstract- The International security standardsare implemented to reduce the security failures and to mitigate their consequences. In recent years, security failures due to web application attacks are constantly on increase and posing new risks for organizations. Web application security has therefore become further crucial. Several types of attacks have emerged recentlythat are launched against web applications. The aim of this research paper is to study and analyze the various International Standards like COBIT, ISMS, OWASP, PCI/DSS and depict the scope of coverage of mitigating measureswhich focus on security of web applications from attacks such as Code Injection.

Keywords - Application; Code Injection; Error handling; Input Validation; SQL injection

I. SECURITY OF WEB APPLICATION

Application security is a set of security mechanisms built to protect confidentiality, integrity and availability of an application. Web applications have become increasing standard because they offer an easy deployment process. To achieve the purpose of modularity in design and platform independence, web-based E-commerce applications are usually developed based on three tier model. By means of the three tier model, we can separate the business logic of the web application from the front end and back end. This gives us more flexible and scalable system. The three tier model component can be used in order to understand the flow of logic of vulnerabilities in the web application. The first tier is a web browser executed on the end user's system termed as web client. The second tier is the application on server side that runs on a web server or a dedicated application server. Generally these applications implement business logic of the web system. The Third tier provides data storage and retrieval services from the various Database Management Systems as per the requirements generated from the second tier. Each of these three tiers
have their own vulnerabilities.From an attacks viewpoint, a web application is therefore an easy target as theattackers just need to exploit one of the vulnerabilities to compromise huge parts of the whole application.
Open Web Application Security Project (OWASP) has categorizedthese vulnerabilities as Cross Site scripting (XSS)being the most critical, followed by Injection Flaws. Injection Flaws[1] like ASP injections, XPath injection[2], LDAP injection[3],PHP injections and SQL injection[4]show an alarming rise in the vulnerabilities arising in the web applications. By performing these attacks, the attacker has understanding in the schema of the database, execute commands and manage to get user credential including their passwords[5]. The attack involves a SQL query which can be used to query any database for information about the database, columns and schema.

II. CODE INJECTION ATTACKS AND MITIGATIONMETHODS

Code Injection is a technique in which the attacker exploits weaknesses in input validation to execute arbitrary commands in the database. Code Injection attacks allow attackers to tamper with existing data, cause repudiation issues such as changing transactions or balances. It allows attacker to spoof identity and can subsequently allow the complete disclosure of data on the system. It can also allow attackerto destroy the data or makes it otherwise unavailable[6].
It can typically occur when the application uses input to construct dynamic query statements to access the database. Code Injection attacks occur when the code uses stored procedures that pass strings that contains unfiltered user input. The issue becomes more critical if the application uses an over privileged account to make a connection to the database. In this case it is also possible to use the database server to run operating system commands and compromise other server, in addition to being able to access, manipulate and destroy data.
One of the major causes of this attack is when an error occurs, an error message from the software system is returned as feedback giving detailed cause of error, which a malicious client can use in order to guess sensitive information about the location and nature and type of the data in addition to valuable connection details [7].
The Injection Attack can be broadly classified into two types-

(1) Blind: In this type of attack the input is still vulnerable to Injection, but error handling is performed to prevent ODBC errors from being displayed on the browser.
(2) Verbose: In this type, there is lack of error handling which provides a detailed feedback to the browser. It is highly prone to attack

Proper input validation is an effective counter measures to act as a defense against input attacks. Although input

validation is an easily understood concept,many application's security vulnerabilities result from generic input validation problems.. Since there is no pattern of malicious injection, there is no generic relation between input and risk of exploit. In many cases, individual fields require specific validation.Therefore there is no single validation rule to sanitize all input.

Code Injection vulnerability is one of the major causes of concern, because once the unauthorized intrusion is made through the vulnerable part of the security armor, the overall security is compromised [8].

Since the characteristics of the above mentioned Code Injections attacks are similar, the following methods to avoid Code Injection vulnerabilities specially SQL Injections have been in practice -

1. Use a low privileged account to run the database. Executing an application that connects to the database using the database's administrator account has the potential for an attacker to perform almost any command with the database.
2. Validate text input and URL query string for improper characters (eg apostrophe, dash). Limit the amount of characters in web from input fields and URL query string to a proper amount. It is necessary to ensure acceptance of all valid data and rejection or sanitization of potentially dangerous data is either rejected or sanitized. This can be difficult when valid characters or sequences of characters also have special meaning and may involve validation of the dataagainst a specified grammar
3. Use of parameterized queries or stored procedures to access a database as against using direct passing of string. This will avoid commands inserted from user input from being directly executed as the logic of a query. If one query is mistakably bypassed, that could be enough to leave the application susceptible.
4. Do not display errors to the user that contain critical information about the database or actual source code. Error messages are useful to an attacker because they give added information about the database that might not otherwise be available. It is often thought of as being useful for the user if an error message is returned to the user if something goes wrong. It is advisable to display a generic error message that stating that an error has occurred along with unique error-id. This error-id will be used the error diagnostics on the server for rectification purpose.
5. Validation for all inputs is a must both at server side and client side. In perspective of security, Server side validation is mandatory. Client side validation can be easily evaded by turning off vbscript andjavascript. But for several other benefits, client side validation should be done in addition to the server side
6. Delete system stored procedures. In case the default installation of data base server is equivalent to AdministratorLevel in Windows then an attacker could use stored procedures to perform remote execution

Therefore, stored procedures like masterXp_cmdshell, ,xp_sendmailetc should be deleted.

III. INFORMATION SECURITY STANDARDS (ISS) AND APPLICATION SECURITY

The increasing complexity of information intensive process has led to the development ofinformation security standards addressing the organizational aspect of IT security. These standards are collation of industry best practices that are used to drive the design and implementation of process. The compliance to a variety of information security standards is mandated by regulations and organizations need to act and implement a proactive information security enforcement strategy.

The main focus of any ISS is to prevent the security failure and to mitigate their consequences. It has been widely recognized that information system security concerns the safeguarding of information system processes and data in any form. The technical control and measures, implements to achieve information security should be managerially monitored, reviewed and enhanced on a regular basis. With existing and emerging regulations in place, there are increasing uses of ISS by the industry to ensure that IT process and controls are adhering to the industrial best practices. However, it is noteworthy fact that in spite of these effortsseveral trends have emerged in the attacks launched against web applications. As a result of this situation, organizations have started the transition from being network security centric organizations to a more application security centric organizations. More companies have started to educate themselves on web application security.

Application security is among others covered in most of the security standards like Control objective for Information and Related Techniques (COBIT), Open Web Application Security Projects (OWASP), ISMS (ISO 27001&ISO27002),NIST, Payment Card Industry(PCIDSS). However, the stated lists of standards are not comprehensive. Since it is apparent from the study of Code Injection attack that the injection attacks are caused due to invalidated inputs, messages generated through errors, hence an attempt has been made to identify these standards and to see whether these standards have guidelines for validation of inputs and managing error messages. The decision to abide with an appropriate security standard can be really difficult, since these standards are process standards and are developed with common objective.

1. Control Objective for Information and Related Techniques (COBIT)

COBIT is a global open standard for control over IT. This information criteria were developed from other well-known security models and integrate the concepts of effectiveness, reliability, efficiency and compliance as well as the traditional security concepts of confidentiality, integrity and availability [9]. COBIT includes 34 IT processes, grouped into four domain includes planning and organization (P&O), acquisition and implementation (A&I), delivery and support (D&S) and monitoring (M). The processes are further classified into 318 control objectives and related audit guidelines.

IT resources include data, application system, technology, facilities and people. COBIT application systems are understood to be the sum of manual and programmed procedures. Data are defined broadly including text, sound, video and graphics.. These two IT resources application systems and a data process business events into usable information. Although application security is integrated into all four COBIT domains, however the *Section DS11 titled "Manage Data"* contains controls that are similar to the counter measures for prevention of Code Injection vulnerabilities.

Managing Data satisfies the business requirement of ensuring that data remain unaltered, accurate and valid during input, update and storage. It is enabled by an effective combination of application and general controls over the IT operation. The control objective of *DS11.7* is accuracy, completeness and authorization checks where the key areas are data input validation. The control objective of *DS11.8* is Data input error handling where the key area iscorrection and resubmission of erroneous data. The control objective of *DS11.9* and *DS11.*10 is data processing integrity, validation and editing, where the key areas are routine verification, update controls. The control objective of *DS11.11* is Data processing error handling where the key area is identification of erroneous transactions .

2. OWASP (Open Web Application Security Project guidelines)

The guidelines provided by OWASP have explicitly defined the Code Injection vulnerabilities and also the preventive measures that need to be taken.

The primarily objective of OWASP is to ensure that the application is able to take care of all forms of input data, whether obtained from the user, infrastructure and database system or external entitles. Very common web application security flaw is the failure to properly confirm input from the client or environment. This leads to almost all of the major weaknesses in applications, such as interpreter injection and buffer overflows. As client has every option to tamper with data, the data received from the client should never be trusted without validations. Integrity checks such as a transaction ID must be included, wherever data transfers from a trusted boundary to a less trusted one, such as from the application to the user's browser, or to a third party payment gateway. To ensure that the data is having correct syntax, within length boundaries, strongly typed, contains only permitted characters, or if numeric is properly signed and within boundaries; validation must be performed on every tier/layer. For example, the presentation tier should validate for web related issues, directory lookups should check for LDAP injection, persistence layers should validate for persistence issue such as SQL / HQL injection and so on. Business rules are known during design and they influence implementation.

3. ISMS (ISO27001 &ISO 27002)

ISO 27001 an international standard has been prepared to provide a model for forming, implementing , operating, monitoring , reviewing, maintaining and improving the information security management system(ISMS)[10]. The design and implement of an organization ISMS is influenced in their needs and objectives. The process employed is based on type and size of the organization. This standard adopts the PDCA (Plan-Do-Check &Act) model. It provides a robust model for implementing security design based on risk assessment. Section A.12.2 of ISO 27001states that the control objectives and controls along with other necessary controls shall be implemented as part of the ISMS process for correct processing in application.

As per ISO 27002,additional controls including ethical and logical controls such as policies, procedures, processes and organizational structure need to be implemented. One of the objectives stated in this standard in Section 12.2 is to prevent errors, loss and unauthorized modification of information in application and to make sure applications process information correctly. Appropriate controls should be designed into applications including validation of input data, internal processing and output data to ensure correct processing.

In subsection 12.2.1 under input data validation, thedata input to application should be validated to ensure correctness and appropriateness of this data. Checks should be applied to the input of parameter tables ,standing data and business transactions. The following implementationguidelines should be considered:

a) Input checks, such as dual input, limiting fields to specific ranges of input data, boundary checking to detect the following errors.
   - Unauthorized or inconsistent control data
   - Out-of-range-values
   - Invalid characters in data field
   - Incomplete or missing data
   - Exceeding data volume limits (upper/lower)
   - Invalid characters in data field

b) Procedures for testing the credibility of input data
c) Procedures for responding to validationerrors

Autoexamination and validation of input data can be considered, to reduce the risk of errors and to prevent standard attacks including code injections and buffer overflow.

In subsection 12.2.2 under control of internal processing, data validation checks to be integrated into applications to detect corruption of information due to processing errors or deliberate attacks.

4. NIST Application Security Checklist

This checklist has a wider scope than other standards as it includes an examination of the application source code as well

as server software (web server, database etc), network infrastructure and operating system. Checks also cover production environment security including backups, user account management etc. Validation of control occurs through either black box testing of application functionality or examination of the source code. In the NSIT application security checklist validations and error handling are cover under "Code Based Elements" category[11].

5. The Payment Card Industry / Data Security Standard(PCI/DSS)

The Payment Card Industry (PCI) Data Security Standard (DSS) describers 12 requirements are organized in 6 logically related groups which are the "control objectives"[12]. Under "Maintain a VulnerabilityManagement Program" group it states in the requirement number6 titled "Develop and update secure systems andapplications" states that unscrupulous individuals usesecurity vulnerability to gain privileged access to system. For in-house developed applications, numerous vulnerabilities can be avoided by using standard development processes and secure coding practices. Insubsection 6.5 it is recommended to develop all web applications based on secure coding guidelines stated in Open Web Application Security Project guidelines and review custom application coding vulnerabilities. Prevention of common coding vulnerabilities in software development processes, is included *in subsection 6.5 asfollow* :

6. Unvalidated input

Injection flaws (for example SQL injection)
Improper error handling
It is stated in Subsection 6.6 to ensure that all web applications are protected against known attacks by applying either of the following methods:

1. Having all custom application code reviewed for common vulnerabilities by an organization that specializes in application security
2. Installing an application layer firewall in front of web applications.

Unless a web application has strong, centralized mechanism for validating all input from HTTP requests, vulnerabilities based on malicious inputs are very likely to exist. Most of these standards do not provide assurance that an application is completely secured and that the application cannot be compromised but have substantial guidelines for input validation.
Securing computer systems should be a very important part of system design, development and deployment. The difficult part of building software is the specification, design and testing of this conceptual construct, not the implementation and testing of the implementation. Syntax errors will be made, but they are fuzz compared to the conceptual errors in most systems. Software flaws are caused because complexity makes software entities hard to design and to manage their development, and increasing complexity makes errors more probable.
New software functions result in side effects that are difficult to predict, increasing the complexity rapidly as the software size grows. Software cannot be simplified by redesign because it has to conform with old programs which add to complexity. Software is subject to pressures for change all the time. Constant introduction of new features augment the pool of vulnerabilities.
The question arises when a benchmark is needed across heterogonous standards. Most of the international security standards do not explicitly state the problem of Injection attacks and the consequent vulnerabilities of the data, but rather provide a series of steps which results in securing the system and implicitly cover Injection threat. Lot of efforts is done in designing the security standards, but in spite of it, Code Injection breach is frequently overlooked and is exploited by the hacker to the maximum extent

IV. CONCLUSION

A single unprotected query statement can result in compromising the security of the application, data or database server. Developers must be disciplined enough to apply the security methods to every web accessible procedure and function. Every dynamic query must be protected. It is apparent that safeguarding of security is becoming more difficult because the possible attack technologies are becoming increasingly sophisticated. Software rooted vulnerabilities like Code Injections can be prevented, if the developers seriously incorporate the rule of validation while developing web applications. In spite the fact that the concept of validation is deep rooted and widely covered in almost all the International standards guidelines yet attacks are at a rise due to Code Injection vulnerabilities. There is an urgent need to make the developers and users aware about the security standards and to encourage them to implement the standards meticulously, so as to minimize such attacks. It is also imperative to make these standards easily available, so that, their usage percolates down even to smaller organizations.

REFERENCES

[1] http://www.owasp.org/index.php
[2] Antunes, N.; Laranjeiro, N.; Vieira, M.; Madeira, H.; "Effective Detection of SQL/XPath Injection Vulnerabilities in Web Services"; Services Computing, 2009. SCC '09. IEEE International Conference on 21-25 Sept. 2009 Page(s):260 – 267
[3] Alonso, J.M. Bordon, R. Beltran, M. Guzman, A. Informatica , Mostoles; " LDAP injection techniques"; Communication Systems, 2008. ICCS 2008. 11th IEEE Singapore International Conference ; page(s): 980 - 986; Location: Guangzhou
[4] Merlo, E.; Letarte, D.; Antoniol, G; "Insider and Ousider Threat-Sensitive SQL Injection Vulnerability Analysis

in PHP", IEEE WCRE 2006 13th Working Conference on Oct. 2006 Page(s):147 – 156

[5] Elisa Bertino, AshishKamra, James p Early,"Profiling Database Application To Detect SQL Injection Attacks", IEEE PCCC 2011.

[6] Stephen Thomas, Laurie Williams,"Using Automated fix Generation to Secure SQL Statement", Third International Workshop on Software Engineering for Secure Systems (SESS'07); ICSE Workshops 2007. Third International Workshop on 20-26 May 2009 Page(s):9 - 9

[7] Xiang Fu Xin Lu Boris PeltsvergerShijun Chen Kai QianLixin Tao "AStatic Analysis Framework For Detecting SQL Injection Vulnerability" , 31st Annual International Computer Software and Applications Conference(COMPSAC 2007Page(s):87 – 96

[8] W.Halfond and A. Orso, "AMNESIA: Analysis and Monitoring for NEutralizing SQL-Injection Attacks", In Proceedings of the 20th IEEE / ACM International conference on Automated software engine , pages 174-183.

[9] http://www.isaca.org/Template.cfm?SectionSecurity&Template

[10] www.iso27001.com

[11] http://csrc.nist.gov/publiations/pubssps.html/800-27/sp800-27.pdf

[12] www.PCI.com/ pci_dss_v1-1[1].pdf

