International Journal of Information & Computation Technology.
ISSN 0974-2239 Volume 4, Number 7 (2014), pp. 699-704

http://www. irphouse.com

A Review of Web Application Security for Preventing Cyber Crimes

[1]Ayush Chugh and [2]Ayushi Gupta

[1,2]Department of CSE & IT,
ITM University, Gurgaon-122017, Haryana, India

ABSTRACT

In today's era, web applications are rapidly used by the people around the world. People are contingent on vivid web applications for daily needs such as online banking, shopping, gaming, booking tickets, using webmail and the list is endless. These applications are developed using various languages like PHP, C#, ASP.NET, Python, PERL using scripts like HTML and JavaScript. Some of these applications are vulnerable to attacks. Attackers develop tools and techniques to exploit user data due to flaws in security. All this defines the need of efficient web application security. The most common attacks on web applications are by SQL Injection and XSS which are commonly used techniques by attackers for cyber-crimes. This paper describes Web Application security and common attacks for data theft and its prevention. Case studies related to network security in banking system have been described and some innovative solutions to resolve these issues have been proposed.

Keywords: Attacker, Authentication, SQL, Vulnerability, XSS.

1. INTRODUCTION

Web application security deals with the protection of web services, web applications, websites and all other benefits of web. The web applications use two types of scripts: Server side script and Client side script. The former scripts (ASP, PHP etc.) include hard stuff i.e., storing and drawing the information while the latter ones, HTML, JavaScript etc., deal with the display of information.

During the past year, attacks on web applications have grown rapidly. The two significant techniques of attack are Structured Query Language (SQL) Injection and Cross Site Scripting (XSS), apart from PHP injection, Java injection, Memory corruption, Path disclosure and Denial of Service.

Nowadays, a significant part of the population use online services to file their income tax, fill passport forms, banking services, etc. using one of these applications. This information would always be on stake if there isn't high security. They invade these applications and get access of confidential data, cause severe loss to the client and can also damage the server. The web application security is needed to prevent raid by attackers.

2. WEB APPLICATION SECURITY

Web Application Security falls under the category of Information security. The design interfaces, which the user uses to interact with backend databases, must be secure to perform the tasks over web applications. In some severe cases attacker can attain system level permissions and direct access to database which can harm the server and execute the commands according to their choice. This accounts for the need for web application security. Generally, it can be divided into two parts: *Declarative security and Program security*.

2.1 Declarative Security

It comprises of mechanisms used in an application, and shown in declarative syntax in a deployment descriptor (DD). A DD is an .xml file that shows application security structure including access controls and authorization requirements. The security information is placed into annotations by declarative security syntax. The metadata includes:

- Description of the assembly i.e., identity, name and type.
- Details of types.
- Attributes.

2.2 Program Security

It is done using programs hence user has full access and authentication control. Consequently, it makes all the components portable as there is involvement of server related components. Also, there is no need to create configuration files. The programs can be written according to the need. For better authentication and user friendly experience, a customized program can be made. The programs give a flexible experience in the development of web applications. Though this communication is slow, it is a safe process. All these aspects help in building dynamic web pages whose content varies according to the arguments passed.

3. SQL INJECTION

SQL stands for Structured Query Language. SQL Injection attack (SQLIA)is one of the top 10 web application vulnerabilities. It refers to the insertion of SQL query through the input data from client to application and is executed by backend database. It can assist an attacker to access all the high privileged data which was hidden earlier. The attacker's input is an SQL query which becomes a SQL code he can intrude the web application and can update/delete/modify the database and even perform

administrative tasks.

A successful SQL Injection attack is due to defective coding of web applications. All the features such as forums, login page, search pages, user credentials can be exploited via SQL Injection.

3.1 Blind SQL Injection

It is used when a web application is vulnerable to SQLIA but the results are not apparent to attacker. The resultant page would depend on the statement entered with the corresponding SQL query. This may demand new query for every new bit. This can be encountered by the use of a tool which will automate the whole attack.

3.2Mitigation of SQL Injection

1. *Use Parameterized Query:* You should not supply the values directly; instead the values can be passed as parameters.
2. *Automatic and Dynamic Access Control List outlining:* This should be preferred so as to continuously monitor the application behavior and adapt to the application changes.
3. *Use quote blocking function:* The function will not allow attackers to intervene inside the database. A check is maintained on the values given by user.
4. *Avoid detailed error message:* The error message can provide the attacker hint regarding the parameters. Though not descriptive, a detailed error message is sufficient for a skilled attacker. So turn off the error reporting or just type something in error message which is of no use to attacker.
5. *Impart limited permissions to users:* All the irrelevant privileges should be removed and granted later if the need be.

4. CROSS SITE SCRIPTING (Xss/css)

It is a very usual application layer hacking techniques used since 1990s. It chiefly targets the social networking websites such as Facebook, Twitter, and Orkut. It is a four stage process:

1. Attacker inserts client side script into vulnerable web server or web pages.
2. A user (victim) visits this server.
3. The code enters the victim's browser.
4. The code executes with web server privileges.

4.1Mitigation

1. To mitigate XSS attack, you must prefer escaping of string input. The script should include the location of the unwanted strings to be placed in a HTML document.
2. You must employ additional security measures when running a website functioning on cookie based authentication.
3. Use plug-ins which could block untrusted websites.
4. Blocking the scripts would be beneficial if you have knowledge about them. Many defensive technologies are available which guarantees minimum XSS.

Those technologies are Mozilla's Content Security Policy, JavaScript Sandbox and Auto-escaping templates.

5. Case Study

In today's internet banking system it is a must for all the companies to have proper security arrangements. For the confidentiality purpose, which is a must in banking sector says that data should not be interpreted by anyone in between transmission other than the sending and receiving authorities, this can be achieved by data encryption techniques. If any amendments are being made in data transmission that can be detected using cryptographic techniques like Hash or message digest functions which guarantee data integrity.

5.1Applications at Risk

Evolution of e-commerce systems has prescribed the need of new technologies like Virtual Private networks (VPN's), firewalls, extranets, proxy servers, Secure Sockets Layer (SSL) and so on.

Sections of banking applications that can use these services are:

1. Corporate Banking System
2. Home Banking Applications
3. Auxiliary services like FX Trading, Trade Finance, Portfolio Management, etc.

For example, consider an imaginary transaction where an internet user starts a transaction to pay a large foreign currency cash value to a recipient. The details of this transaction like amount and recipient account details must remain confidential. In this case, bank requires absolute assurance that the initiator of the instruction is genuinely who they intend to be. The bank will require some form of electronic authentication, similar to an identity document that might be presented under normal "paper" circumstances. Final transaction value might be affected by fluctuation in currency market after the transaction was initiated, this could work in anyone's favor but it requires both parties to be unable to reject either payment or its receipt; this condition states a need for the sender and bank to digitally sign their messages.

5.2Solutions

Data security requirements are defined in terms of the need for confidentiality, integrity and authentication.

1. **Confidentiality and integrity**

The Secure Sockets Layer (SSL) encrypts and hashes all traffic, thus providing a high degree of confidentiality and integrity to both parties in a transaction.

2. **Authentication**

SSL can provide server and client authentication through the deployment of digital certificates but there is a drawback of these certificates. These are attached to the

software not the end user; therefore there is an absence of digital signature hence it is not feasible to provide authentication to multiple users at a single browser. Thus the best way, a user can authenticate himself is by means of some form of token that he holds which contains a digital certificate and a password or PIN which he knows. For this User Smart Cards are a good means which contain a signature key and certificate, protected by PIN and accessible from a browser plug-in or Java applet.

6. CONCLUSIONS

Web application security is the demand of the hour. Online business is most affected by web application vulnerabilities. The main problem is that most of the web applications don't have testing of desktop software. Therefore there is always a growing concern for security. There must be a way to deal with the unexpected input from a browser, unhandled error messages, unchecked database call. You can use content management system like Joomla! to construct a website with dynamic content. There can also be a way to prevent injections which is to avoid characters which have a distinct meaning in SQL. Single quote (') can be replaced by two single quotes ('') to form an accurate SQL string literal. You must assign limited permissions to the database and extend these when the need arises.

You must prefer PHP over any other language as PHP is moving towards object oriented architecture and there is use of PDO classes which give privilege to make prepared statements to prevent SQL Injections.

In order to secure your web applications you must be aware about the above said attacks and should be prepared for any kind of new attacks. Use web application vulnerability evaluation tools in order to check the current security level of your web application or website. Educate developers, security professionals, testers regarding the risks and steps to mitigate the attacks. You must be sure how you pass the data and don't expose your logic. A single loop hole can lead to the destruction of your whole database. You must always have a backup and keep it safe.

Acknowledgements:

We greatly acknowledge the research guidance provided by Mr. Avdesh Bhardawaj, Asst. Professor, ITM University, Gurgaon, Haryana, India.

7. REFERENCES

.

[1] Kim-Kwang Raymond Choo (2011) The cyber threat landscape: challenges and future research directions computers & security, Volume 30, Issue 8, Pages 719–731

[2] Kindy, D.A. and Pathan, A.-S.K., "A Survey on SQL Injection: Vulnerabilities, Attacks, and Prevention Techniques" *in Proc. 15th IEEE Symposium on Consumer Electronics,*Singapore, 2011, pp. 468-471.

[3] Klaus Julisch (2013) Understanding and overcoming cyber security anti-patterns, Computer Networks, Volume 57, Issue 10, pp 2206–2211

[4] Robin Dyer, (2013) "External reactive detection v. internal proactive prevention: The holistic approach to integrate change", Journal of Financial Crime, Vol. 20 Iss: 3, pp.287-292

[5] Tajpour, A., Masrom, M., Heydari, M.Z., and Ibrahim, S., "SQL injection detection and prevention tools assessment," *in Proc. 3rd IEEE International Conference on Computer Science and Information Technology*, China 2010, pp. 518-522.