International Journal of Scientific Engineering and Research (IJSER)
www.ijser.in
ISSN (Online): 2347-3878, Impact Factor (2014): 3.05

Ranking and Impact of Web Applications' Vulnerabilities

Daljit Kaur[1], Dr. Parminder Kaur[2]

[1]Assistant Professor, Lyallpur Khalsa College, Jalandhar, Punjab, India
[2]Assistant Professor, Guru Nanak Dev University, Amritsar, Punjab, India

Abstract: ***Web applications are important, popular and common distributed systems. With the growth of internet and web, competition has increased to get web applications online. In this competition era, web applications are developed in short time frames and with number of vulnerabilities. Web developers often struggle with development timelines rather than security certifications. With many known and unknown vulnerabilities, minimizing application's vulnerabilities is a daunting challenge. These vulnerabilities represent a security risk which can lead to financial damage or loss of reputation. This paper ranks common vulnerabilities, attacks and also calculates risk using CVSS (common vulnerability scoring system) vulnerability metrics which are publicly available and used for ranking. It provides platform for understanding of vulnerabilities, ranking them, calculating vulnerability risk and using them to decide and improve the security of web applications.***

Keywords: Web vulnerabilities, Vulnerability Impact, Risk and Web security

1. Introduction

In this modern era, with the growth of internet and World Wide Web, web applications have a significant impact on social and professional life of human beings. Many legacy, private information and database systems are migrated to the Internet and the Web environments. On-line applications continue to grow more complex. While benefits of web applications are rich, risks to organization, brand and data are more costly. Everyday there are new reports on cyber attacks on leading web sites [1]. Application layer continues to be a soft target with increasing cyber attacks. 96% of web applications have one or more serious vulnerabilities [1, 2]. Web applications continue to be plagued by weaknesses deemed critical by various industry standards [1,3,4,5]. Application developers spend most of their time adding features rather than rooting out all applications vulnerabilities. They often struggle with development timelines rather than security implementation. Security breaches continue to occur in web applications because we have not addressed a core problem with those applications: insecure web development and security testing. Web-based systems have been kept running through a continual stream of patches [6]. With so many vulnerabilities to choose from, hackers can easily breach the increasingly valuable data that application has. Poorly developed Web-based applications that are mushrooming now have a high probability of failures. Technical attacks seek security weaknesses in an application and use exploits which are tailored to these vulnerabilities, for purposes like controlling the application's server, modifying the application's data or harming the application's users [7]. Risk to society due to exploitation of vulnerabilities is massive as online banking, stock market trading, transportation, even military and governmental exchanges depend on Internet based computing and communications. Yet, people are willing to take risk since Internet has made markets and transactions more efficient.

To develop secure system, security must be built from the ground up, from the beginning to the end of a product's life cycle[3]. In general, early detection is more efficient and much vulnerability are relatively easy for security teams to detect, block and fix during every phase of applications development life cycle [1]. Prevention would be the most efficient way to reduce vulnerabilities, if vulnerabilities and their risks, are known in advance to both developer and client.

This paper finds the web vulnerabilities from national vulnerability databases and others like OWASP, SANS, etc and rank them according to higher frequency in different databases. Also it calculates risk and impact of each vulnerability to understand it in an easy and better way. Section II describes web application and its security issues, Section III briefs about Risk and Security metrics, Section IV ranks vulnerabilities and calculates their impact. Section V concludes that how we can use this analysis to improve security of web applications.

2. Web Applications and Their Security Issues

A. Web Application Architecture

Web applications are computer programs that allow visitors to submit and retrieve data to/from database over Internet using preferred web browser [8]. With the popularity of web applications, they have turned out to be an omnipresent phenomenon. But due to highly technical and complex nature, web applications are extensively unknown and grossly misunderstood fixture in our daily cyber-life. Technically, web is highly programmable environment, which allows the immediate deployment of large and diverse range of applications. A contemporary website has two important components: flexible web browser and web application; both available to all at no expense. Web browsers are the software applications that allow users to retrieve data and interact with content location on web pages within a website. They interpret and run all scripts while displaying requested pages & content. Modern web pages allow personalised dynamic content to be pulled down by users according to individual preferences and settings. Furthermore, web pages may also run client side scripts that change the browser to an

interface for such applications as Yahoo Mail and Google maps. Most importantly, today's web applications allow the capture, processing, storage and transmission of sensitive customer data such as personal details, credit card number and social security information for immediate and recurrent use.

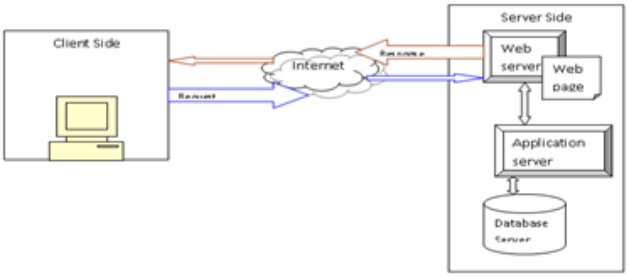

Figure 1: Three layered web application model

The Figure1, details the three layered web application model. The First layer is web browser or the user interface, where user interacts with application such as Microsoft Internet Explorer, Mozilla and Opera. The second layer which consists a web server such as Apache, iPlanet, Microsoft Internet Information Service (IIS) and Netscape Enterprise operates at server side. It receives request from the client side and processes them using some dynamic content generation technologies such as Java Server Pages(JSP) or Active Server Pages(ASP). The server returns responses containing data components such as static or dynamic pages generated by it or by data processing components. The third layer is database containing valuable data content such as customer data like user name, passwords, social security number, credit card no etc. A database or RDBMS (Relational Database Management System) inserts, retrieves and manipulates data in database through SQL queries, as well as controls access to database through access control mechanisms.

B. Security Issues

Web applications raise number of security concerns and are currently subject to attacks such as Cross Site Scripting (XSS), Cross Site Request Forgery (CSRF), URL Redirection. Web attacks are increasing day by day and application layer continuous to be a target for attackers [1]. There are large numbers of attacks that are done at application level. Security is one of the key features of any web application or software system in which unauthorized access and modification of information and resources that leads to loss of confidentiality (C), Integrity (I) and availability (A) are prevented [9]. Security has three universally accepted attributes C, I and A. Confidentiality refers to maintain prevention of unauthorized disclosure of information. Integrity refers to assurance of accuracy, completeness of information and then prevention of unauthorized modification of information. Availability refers to protection of system as well as data to ensure available when required.

Generally, three approaches are used to develop secure software. First approach is 'Penetrate and Patch', in which patches are applied to fix vulnerable applications. It is common approach to security applications but studies shows that relative cost of fixing defects during production or after product has been released is 30 to 100 times expensive [10]. The second approach focuses on securing operational environment by external devices such as firewalls, intrusion detection system (IDS). Though these mechanisms can provide some security, but may not make applications stand to attacks. Also, they can be used when development process is complete and operational. Third approach is secure software engineering and basic idea behind this is to implement well structured processes from requirement analysis through development and implementation with security in mind. Usually, a combination of these approaches can be used for securing software. Nonetheless, the future probable expenditure can be reduced by addressing security during development. Web applications can have number of security issues such as insufficient authentication, protecting sensitive data, concurrency, preventing session hijacking, providing secure configuration, input validation, exceptional handling, user authorization, protecting sensitive data etc. shown in figure 2.

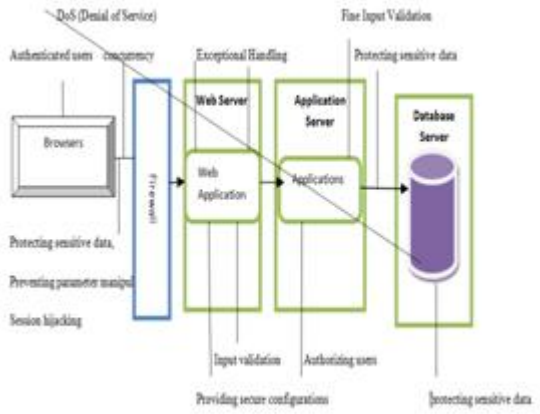

Figure 2: Web Application Security Issues

3. Risk and CVSS Metrics

Security vulnerabilities that have been discovered but remain unpatched for a while represent considerable risk for an organization. A vulnerability is a software defect or weakness in the security system which might be exploited by malicious user causing loss or harm [11].

A complete secure system is not possible but to ensure that overall security risk stays within acceptable limits, organization needs to measure risks. As Lord Calvin said "If you can't measure it, you can't improve it". We need to consider quantitative methods to make decision; subjective perception is not only enough. Risk, in general is the possibility of a harm to occur [11], technically Risk is defined as weighted measure depending on the consequence. It is stated as [12]

Risk = Likelihood of an Adverse Event × Impact of an Adverse Event..... (1)

Equation (1) evaluates risk due to single specific cause. In general, a system has multiple weaknesses. The risk of exploitation in each weakness w is given by Equation (1).

Likelihood of an adverse event is the estimation of successful attack from a group of possible attackers and it

is also called LoE (Likelihood of Exploitation). It can be determined by various Threat Agent and Vulnerability Factors [13]. Each factor has a rating scale 0-9 for calculating the overall likelihood of the vulnerability.

Threat Agent Factors for determining LoE are:

Skill Level Required- High security and penetration skill: 1, Programming and Networking skill: 3, High computer skill: 4, Some Technical Skills: 6, No Technical Skill: 9

Motivation to find and exploit Vulnerability- Low/None: 1, Medium: 4, High: 9

Opportunity and Resources Required- Full Access/expensive resources: 0, Special Access/Resources: 4, Some Access/Resources: 7, No Access/Resources: 9

Size of Threat Agents- System Administrators/ Developers: 2, Intranet users/Partners: 5, Authenticated Users: 6, Internet Users: 9

Vulnerability Factors for determining the severity of the problem are:

Discoverability- Practically Impossible: 1, Difficult: 3, Easy: 5, Automated Tools Available: 9 Exploitability Ease- Theoretical: 1, Difficult: 3, Easy: 5, Automated Tools Available: 9

Awareness- Unknown: 1, Hidden: 4, Obvious: 6, Public Knowledge: 9

Intrusion Detection- Active Detection in Application: 1, Logged and Reviewed: 3, Not Logged: 9

All the Threat Agent and Vulnerability factors for determining LoE are given a value according to the vulnerability and overall LoE of the vulnerability is calculated using medium.

Impact of the vulnerability can consider in two ways. First is the Technical Impact on application, its data and functions. The other is Business impact, which is the business consequence after a successful attack, and it is more important than technical impact. It varies with organization and only business representative can make decision about the business risk. Business Impact Factors may include Financial Damage, Reputation Damage, Non-Compliance and Privacy Violation. Business Risk is what justifies cost in fixing security requirements.

Factors for determining the Technical Impact are:

Confidentiality Loss: Minimal non-sensitive data disclosed: 2, Minimal critical data disclosed: 6, Extensive non-sensitive data disclosed: 6, Extensive critical data disclosed: 7, All Data disclosed: 9

Integrity Loss: Minimal non-sensitive data corrupted: 1, Minimal critical data corrupted: 3, Extensive non-sensitive data corrupted: 5, Extensive critical data disclosed: 7, All Data Corrupted: 9

Availability Loss: Minimal Secondary Services Interrupted: 1, Minimal Primary Services Interrupted: 5, Extensive Primary Services Interrupted: 7, All Services Lost: 9

Similar to LoE, Impact of the vulnerability is calculated using medium of all the factors affecting Impact. When statistically independent multiple causes are considered, the individual risks need to be added to obtain the overall risk. Assuming that potential exploitation of a weakness is statistically independent of others, the system risk is given by summation of individual risk values:

$$\text{System Risk} = \sum_w L_w \times I_w \quad \text{.....(2)}$$

Where L_w is the likelihood of exploitation of weakness w and I_w is the corresponding impact.

For assessing security vulnerabilities, Common Vulnerability Scoring system (CVSS) [=10] has become an industrial standard although some other alternatives are used. It is an attempt to evaluate the degree of risks posed by vulnerabilities, so mitigation efforts can be prioritized. It defines number of metrics that can be used to characterize vulnerability. The scores are calculated using metrics of vulnerability attributes based on opinions of experts in the field. The CVSS scores for known vulnerabilities are readily available on majority of public vulnerability databases on web.

We have calculated total technical impact (TI_w) as the combined impact of vulnerability w, on confidentiality, Integrity and Availability of the system. It is calculated with the formula in 3a and Average Technical Impact is calculated by dividing by 3 as in 3(b).

$$TI_w = \sum C_w, I_w, A_w \quad \text{.....(3a)}$$
$$\text{Avg. } TI_w = \sum(C_w, I_w, A_w)/3 \quad \text{.....(3b)}$$

TI_w is the total impact of vulnerability and C_w, I_w, A_w are the individual impacts on confidentiality, Integrity and Availability respectively.

4. Vulnerabilities Ranking and Impact

A. Vulnerabilities Ranking

In the literature, in web applications, various vulnerabilities are reported by different software agencies. For our work we are considering the vulnerabilities of web applications reported by OWASP, SANS, CWE, Cenzic and other website security statistics and top threats of recent years, which are most famous and reliable databases of security vulnerabilities. OWASP is an open community dedicated to enabling organizations to develop, purchase and maintain application that can be trusted. It provides a list of top ten vulnerabilities on yearly basis, research findings on current issues related to web application's security and other security related standards and practices. WASC (Web Application Security Consortium) is a non-profit organization made up of an international group of experts, industry practitioners, and organizational representatives who produce open source and widely

agreed security practices for web. The SANS/CWE top 25 is a list of most widespread and critical programming errors that can lead to serious software vulnerabilities. CWE (Common Weakness Enumeration) is a community developed formal list of common software weaknesses. MITRE maintains the CWE web site, with the support of US Department of Homeland security's National cyber security division. Cenzic Inc., is now part of Trustwave that provides a platform for identifying security weaknesses which significantly help to reduce risk and threat to system.

This section discusses the most prevalent security vulnerabilities related to web applications and their properties, and also rank them according to their report in various databases by finding their mode. Table 1 shows the list of vulnerabilities, their mode value, CWE-ID of the vulnerability and its rank. CWE- ID is the unique ID provided by CWE to each weakness/vulnerability in order to give a common baseline standard for weakness identification, mitigation, and prevention efforts [14].

Table I: Vulnerabilities Ranking

S. No.	*Vulnerability*	*CWE-ID*	*Frequency*	*Rank*
1	Session management	CWE-384	4	3
2	XSS	CWE-79	6	1
3	Authentication related	CWE-287	3	4
4	Authorization	CWE-285	4	3
5	Server Vulnerabilities and configuration	CWE-16	2	5
6	CSRF	CWE-352	6	1
7	SQLi	CWE-89	5	2
8	Unauthorized Directory Access	CWE-22	2	5
9	Remote Code Injection	CWE-94	2	5
10	Information Leakage	CWE-200	5	2
11	Content Spoofing	CWE-345	2	5
12	Insufficient Transport layer protection	CWE-818	2	5
13	Abuse of Functionality	CWE-840	1	6
14	HTTP Response Splitting	CWE-113	1	6
15	Predictable Resource Location	CWE-425	1	6
16	Brute Force	CWE-287	3	4
17	URL Redirection	CWE-601	4	3
18	Directory Indexing	CWE-548	1	6
19	Insecure Direct Object reference	CWE-932	1	6
20	Missing function level access control	CWE-935	1	6
21	Buffer overflow	CWE-120	2	5
22	Missing encryption of sensitive data	CWE-311	1	6
23	Untrusted input insecurity decision	CWE-807	1	6
24	Unrestricted upload of file with dangerous type	CWE-434	1	6
25	Execution with unnecessary privileges	CWE-250	2	5
26	path traversal	CWE-36	2	5
27	Non Http Session Cookie	CWE-614	1	6
28	By pass something	CWE-592	1	6
29	Memory corruption	CWE-119	1	6
30	File Inclusion	CWE-98	1	6

The vulnerabilities with highest mode have rank 1, and so on. The top ranked vulnerabilities related to web applications and reported up to 2014 are discussed here in brief with their major properties.

A. Cross site Scripting (XSS)

Cross site scripting is an attack technique in which an attacker echoes its malicious code into a user's browser instance. It occurs when a web application sends a page containing user supplied data to the browser without validation, filtering, or escaping [15]. Web technologies like ASP.NET, PHP, JSP may suffer with this vulnerability. Reflected XSS, stored XSS and DOM-based XSS are three variants of this attack.

Properties: Confidentiality: High, Integrity: High, Availability: Medium, Time of Introduction (TOI) in SDLC Phase: Design (D), Implementation (I); Likelihood to Exploit (LoE): High

B. Cross site Request Forgery (CSRF)

Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. CSRF attacks specifically target state-changing requests, not theft of data, since the attacker has no way to see the response to the forged request [15,19]. With a little help of social engineering (such as sending a link via email or chat), an attacker may trick the users of a web application into executing actions of the attacker's choosing. If the victim is a normal user, a successful CSRF attack can force the user to perform state changing requests like transferring funds,

changing their email address, and so forth. If the victim is an administrative account, CSRF can compromise the entire web application.

Properties: Confidentiality: High, Integrity: High, Availability: Low, Time of Introduction (TOI) in SDLC Phase: Design (D); Likelihood to Exploit (LoE): High

C. SQL Injection (SQL i)

A SQL injection attack consists of insertion or "injection" of a SQL query via the input data from the client to the application. A successful SQL injection exploit can read sensitive data from the database, modify database data (Insert/Update/Delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system and in some cases issue commands to the operating system [15,20].

Properties: Confidentiality: High, Integrity: High, Availability: Low, Time of Introduction (TOI) in SDLC Phase: Design (D), Implementation (I), Operation (O); Likelihood to Exploit (LoE): Very High

D. Information Leakage

Information Leakage is an application weakness where an application reveals sensitive data, such as technical details of the web application, environment, or user-specific data [15,17]. Sensitive data may be used by an attacker to exploit the target web application, its hosting network, or its users. Therefore, leakage of sensitive data should be limited or prevented whenever possible. Information Leakage, in its most common form, is the result of one or more of the following conditions: A failure to scrub out HTML/Script comments containing sensitive information, improper application or server configurations, or differences in page responses for valid versus invalid data.

Properties: Confidentiality: High, Integrity: Low, Availability: Low, Time of Introduction (TOI) in SDLC Phase: Design (D), Implementation (I); Likelihood to Exploit (LoE): High

E. Session Management

Session management attack includes session fixation, insufficient session expiration and broken session attacks. This type of attack occurs when a web application permits an attacker to reuse old session credentials or session IDs for authorization without first invalidating the existing session [16]. The lack of proper session expiration may increase the likelihood of success of certain attacks such as to steal or reuse users session identifiers that further can be used to view other users account or perform a fraudulent transaction.

Properties: Confidentiality: High, Integrity: High, Availability: Low, Time of Introduction (TOI) in SDLC Phase: Design (D), Implementation (I); Likelihood to Exploit (LoE): High

F. Authorization

Insufficient authorization vulnerability occurs when the web application's users are allowed to perform a function or access data without going through proper authorization checks. Attacker can gain access to protected data by exploiting authorization security policy. Weak cryptographic key's generation, weak algorithm usage and insecure storage are the common practices leading to insufficient authorization [15].

Properties: Confidentiality: High, Integrity: High, Availability: High, Time of Introduction (TOI) in SDLC Phase: Design (D), Implementation (I), Operation (O); Likelihood to Exploit (LoE): High

G. URL Redirection

In URL Redirection attack, an http parameter may contain a URL value and could cause the web application to redirect the request to the specified URL. By modifying the URL value to a malicious site, an attacker may successfully launch a phishing scam and steal user credentials. As the server name in the modified link is identical to the original site, phishing attempts have a more trustworthy appearance [16].

Properties: Confidentiality: High, Integrity: Low, Availability: Low, Time of Introduction (TOI) in SDLC Phase: Design (D); Likelihood to Exploit (LoE): Medium

H. Authentication

Authentication related attacks occur when a web site permits an attacker to access sensitive content or functionality without having proper authentication [17]. Hence, depending upon the web site resources, the web application should not be directly accessible to users without verifying the authentication.

Properties: Confidentiality: High, Integrity: Medium, Availability: Low, Time of Introduction (TOI) in SDLC Phase: Design (D), Implementation (I); Likelihood to Exploit (LoE): High

I. Brute Force

Brute Force attack is a method in which an attacker attempts to determine an unknown secret value to gain access to a protected asset by using an automated process usually trial and-error [18]. Entropy of the value is smaller than perceived is the main fact about this attack. Log-in credentials brute forcing in web application is possible, as users usually select easy to memorize words or phrases as passwords. Such an attack attempting to log-in to a system using a large list of words and phrases as potential passwords is often called a "word list attack" or a "dictionary attack".

Properties: Confidentiality: High, Integrity: High, Availability: High, Time of Introduction (TOI) in SDLC Phase: Design (D), Implementation (I); Likelihood to Exploit (LoE): High

J. Server and Application Vulnerabilities

Server and Application Misconfiguration attacks exploit configuration weaknesses found in server and web applications. Many applications come with unnecessary and unsafe features, such as debug features, enabled by default. These features may provide a means for a hacker to bypass authentication methods and gain access to sensitive information, perhaps with elevated privileges. Likewise, default installations may include well-known usernames and passwords, hard-coded backdoor accounts, special access mechanisms, and incorrect permissions set for files accessible through web servers [17]. All of these misconfigurations may lead to unauthorized access to sensitive information.

Properties: Confidentiality: High, Integrity: High, Availability: High, Time of Introduction (TOI) in SDLC Phase: Design (D); Likelihood to Exploit (LoE): High

5. Vulnerabilities Impact on System

This section calculates the Total Technical Impact of each vulnerability, Average impact and Risk of each vulnerability on the system. For each vulnerability, impact value is assigned from 0 to 9, for its impact on Confidentiality (C), Integrity(I) and Availability (A).Total Technical Impact and average impact is calculated with equation 3(a) and 3(b) respectively. In the similar way, Likelihood of Exploitation (LoE) of each vulnerability is assigned value from range 0-9 with its highest exploitability 9, medium 7 and low 3. Next, system risk is calculated according to equation 2 and the results are shown in Table 2.

Table II: Vulnerabilities Impact and Risk

S. No.	*Vulnerability*	*C*	*I*	*A*	*ToI*	*LoE*	*Total Technical Impact*	*Average Technical Impact*	*System Risk*
1	XSS	7	7	5	D,I	7	19	6.333333333	44.333
2	CSRF	7	7	3	D	7	17	5.666666667	39.666
3	SQLi	7	7	3	D,I,O	9	17	5.666666667	51
4	Information Leakage	7	3	3	D,I	7	13	4.333333333	30.333
5	Session management	7	7	3	D,I	7	17	5.666666667	39.666
6	URL Redirection	7	3	3	D	5	13	4.333333333	21.6666
7	Authorization	7	7	7	D,I,O	7	21	7	49
8	Authentication related	7	5	3	D,I	7	15	5	35
9	Brute Force	7	7	7	D,I	7	21	7	49
10	Server Vulnerabilities and configuration	7	7	7	D	7	21	7	49
11	Content Spoofing	3	7	3	D,I	5	13	4.333333333	21.666
12	Insufficient Transport layer protection	7	7	3	D,O	3	17	5.666666667	17
13	Directory Indexing	7	3	3	D	3	13	4.333333333	13
14	path traversal	7	7	7	D,I	7	21	7	49
15	Buffer overflow	7	7	7	D,I,O	7	21	7	49

It shows that SQLi vulnerability has the highest system risk value, and the vulnerabilities authorization, brute force, server mis-configuration, path traversal, and buffer overflow have the next value of risk and highest technical impact. SQLi is on the highest technical risk as it is easy to exploit and popular. While on the other hand, the vulnerabilities Insufficient Transport Layer Protection and Directory indexing are not as easy to exploit, they need sound technical knowledge for exploitation and thus have less risk on the system. Moreover, the system risk is dependent on the technical and business impact and here we are considering technical impact only as business impact varies with organization and their data. Any organization can find out the total impact of vulnerability by summing up the technical and business impact (according to its need). This may help the organization to decide that what they want in their web sites and how they can develop more secure web sites just by keeping in mind the risk that their organization can afford as shown in figure3.

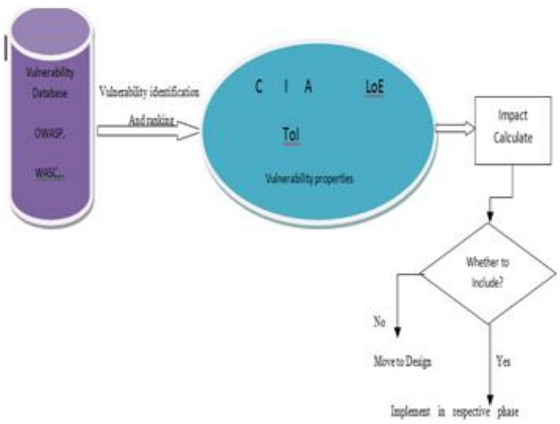

Figure 3: Vulnerability Classification and Decision making

Tolerance level of risk is decided by the organizations which may help them to decide whether to take the preventive actions for the vulnerability or to just ignore it without wasting much time on its preventive implementation.

6. Conclusion and Future Work

In this paper security vulnerabilities related to web applications were ranked according to their frequencies in various vulnerabilities database. This paper also describes top ten vulnerabilities found and calculates the technical risk of each vulnerability, which may help the organizations to decide to afford it or not. The major need is to devise a strategy to develop web applications immune to the vulnerabilities. One technique may be development of security requirements based on the vulnerability analysis and assessment. Requirements are considered as foundation stone on which the entire software is to be built and the requirements phase is the foremost opportunity for the product team to consider, how security will be integrated into a development process. This research work may help to provide effective and efficient ways to incorporate security 'in inception itself' in the development life cycle enabling secured web applications.

References

[1] "Application Vulnerability Trends Report: 2014" Cenzic, Inc. Retrieved from http: //info.cenzic.com/rs/cenzic/images/Cenzic-Application-Vulnerability-Trends-Report-2013.pdf
[2] Mead, R.Nancy and G.McGraw, A Portal for software Security. IEEE Security and Privacy. Published by IEEE Computer society 2005.
[3] "Cisco 2014 Annual Security Report" by Cisco. Retrieved from http: //www.cisco.com/web/offers/lp/2014-annual-security-report
[4] "HP 2012 cyber Security Report" by HP. Retrieved from http: //www.hpenterprisesecurity.com/collateral/whitepaper/HP2012CyberRiskReport_0213.pdf
[5] "Understanding Web Application security challenges", Retrieved from ftp: //ftp.software.ibm.com/software/rational/web/whitepapers/r_wp_webappsecurity.pdf
[6] "The Ten Most Critical Web Application Security Vulnerabilities". Retrieved from http: //umn.dl.sourceforge.net/sorceforge/owasp/OWASPTopTen2010.pdf,2010
[7] Imperva's Web Application Attack Report, Edition 2, January 2012. Retrieved From http: //www.imperva.com/download.asp?=114
[8] Web Application, Retrieved from http: //en.wikipedia.org/wiki/Web_application
[9] R.Crook, D.Ince, L.Lin and B. Nuseibeh, "Security Requirement engineering: When Anti Requirements Hit the Fan", Requirement Engineering, IEEE.pp 203-205 in 2002
[10] C.B.Haley, R.Laney and J.D. Moffett, "Security Requirement engineering: A framework for Representation and analysis", IEEE Transactions on software Engineering, 34(1), pp.133-153 in 2008
[11] C.P. Pfleeger and S.L. Pfleeger, "Security in Computing", 3rd edition. Prentice Hall PTR, 2003.
[12] H.c.Joh and Y.K. Malaiya, "Defining and Assessing Quantitative Security risk Measures Using vulnerability Life Cycle and CVSS Metrics", International Conference Security and Management. SAM'11
[13] "The OWASP Risk Rating Methodology", Retrieved from https: //www.owasp.org/index.php/OWASP_Risk_Rating_Methodology#The_OWASP_Risk_Rating_Methodology
[14] CWE/SANS Top 25 Most Dangerous Programming Errors and Common Weakness Enumeration (CWE). Retrieved from http: //cwe.mitre.org/top25
[15] R.Kumar, "Development of security Requirements for Web Applications", Ph.D Thesis, Jamia Millia Islamia (A Central university), New Delhi, India. September 2011
[16] R.Kumar, "Revisiting Security Vulnerabilities: Web application Perspective", International Journal of Advance Research in Computer science and Software Engineering Vol.3, Issue6, June 2013.
[17] S.B. Chavan and B.B. Meshram, "Classification of Web Application Vulnerabilities", IJESIT Vol2, issue 2, March 2013.
[18] D.Endler," Brute force Exploitation of web application session Ids", Idefense 2001. Retrieved from http: //www.cgisecurity.com/lib/SessionIDs.pdf
[19] R.D.Kombade and B.B. Meshram," CSRF Vulnerabilities and Defensive Techniques" IJCNIS, 2012.
[20] A.TajPour, S. Ibrahim and M.Sharifi, "Web Application Security by SQL Injection Detection Tools", IJCSI Vol.9 Issue2, No. 3, March 2012