

Software Design Level Security Vulnerabilities

S. Rehman ⁽¹⁾ and K. Mustafa ⁽²⁾

(1) Department of Compute Science, Jamia Millia Islamia, New Delhi, 110025 (India)
E-mail: Shabana.infosec@gmail.com

(2) Department of Computer Science, Jamia Millia Islamia, New Delhi, 110025 (India)
E-mail: kmustafa@jmi.ac.in

ABSTRACT

Several thousand software design vulnerabilities have been reported through established databases. But they need to be structured and classified to be optimally usable in the pursuit of minimal and effective mitigation mechanism. In order we developed a criterion set for a communicative description of the same to serve the purpose as a taxonomic description of security vulnerabilities, arising in the design phase of Software development lifecycle. This description is a part of an effort to identify appropriate strategies for mitigating security flaws at the design level. In addition, an analysis is also carried out on the basis of frequency and severity of vulnerabilities identified under each class and the same has been intrinsically presented.

Keywords: Security design, Software Security, Secure Design Taxonomy, Design Vulnerabilities, Software Security Vulnerabilities

1- INTRODUCTION

Software design is one of the most important and complex part of the SDLC (software Development Life Cycle). Integrating software defence mechanisms at the time of design adds more complexity to this process. One way of making this process less tedious is the organization of knowledge of already discovered vulnerabilities and using it as a tool for mitigating vulnerabilities in next generation of software. The first step in understanding vulnerabilities is to classify them into a taxonomy based on their characteristics [1]. "An organizing framework can be used to generalize and communicate findings within research community, taxonomies, or theoretical study of classification organizes the body of knowledge that constitutes a field" [2]. It is now a well known that most of the vulnerabilities discovered in new software are same as those already discovered, and whose mitigation mechanism are available. Most of the security assessment processes involve probing the system to detect the presence of known vulnerabilities because most attacks typically exploit known vulnerabilities that have not been patched [3].

In a survey conducted by Lura Falk at el [4], it was found that 76% of the sites surveyed suffered from at least one design flaw and mitigation mechanisms of all design flaws reported in their paper are available. One of the reasons for not adopting known mitigation mechanisms by the designers at the time of

design is a lack of organized information and limited time for developing software. Software designer lacks scope to investigate and analyse all possible vulnerabilities published in different vulnerability databases. The information provided in various databases is so ambiguous and scattered that retrieving useful information from these databases is a tedious task. Current state of the art raises the need for a systematic way to organize security related information.

In different vulnerability databases, definitions of each of the vulnerabilities are unique but there are common keywords through which their causes can be traced and they can be classified according to their common factors. Here we are classifying vulnerabilities on the basis of software features that can be of two types as reported in [5], one 'Security features' and the other as 'Functional features'. However, in this study we considered only security features, in order to delimit the vast study around vulnerabilities. The main objective of the proposed taxonomy is to classify software design level vulnerabilities by identifying vulnerabilities or the classes of vulnerabilities based on the "reason of cause". At most care is taken to follow all guidelines which need to be followed to develop an efficient and successful taxonomy. The classification will help security analyst to understand the reasons of weakness brought to the system and how they are linked to each other. Severity of vulnerability classes help in analysing the effects of vulnerabilities on the system and thus facilitate the risk assessment and efficiency of countermeasures.

In this paper identification of taxonomic features is conducted and a complete taxonomic description is evolved. To the best of our knowledge there is no taxonomy reported in the literature which classifies software design level vulnerabilities on the basis of the security feature of the software. Software developers can adapt it as a tool to identify a mitigation mechanism to mitigate whole class of design level vulnerabilities in the security features of the software. The remainder of this paper is organized as follows. In section 2, the vulnerability taxonomy related work is described as foundation and the theoretical framework. Development and Identification of relevant taxonomic features is accomplished in section 3, followed by the NVD case study in section 4. Section 5 includes a situational discussion on the proposed description. Conclusion and future work is reported briefly in section 6.

2- SOFTWARE VULNERABILITY TAXONOMIES

Taxonomy refers to a classification as well as the how-to of classification, and its rationale [6]. Classifications that are created non-empirically are called a priori classifications and classifications that are created empirically by looking at the data are called a posteriori classifications [7]. While beginning the scientific study of a new field, a good taxonomy is considered an important and a necessary prerequisite for a systematic study [8,9]. Many researchers have worked in the direction of evolving taxonomies of software security vulnerability. One of the most popular examples of software vulnerability taxonomy is that of Ivan Krusal [10]. He proposed vulnerability taxonomy based on the assumptions that the programmer makes regarding the

environment in which his application execute. Anil bazaz [11] proposed a taxonomy which is based on the Process/Object Model of Computation. It establishes a relationship between software vulnerabilities, an executing process, and computer system resources such as memory, input/output, or cryptographic resources. Zhongqiang Chen et al [12] proposed a vulnerability categorization framework, which classifies vulnerabilities based on three aspects: (i) Vulnerabilities Causes, (ii) Vulnerabilities impacts and (iii) Exploitation Locations. They further classify Vulnerability Cause Aspect into 11 classes as follows:

- Access Validation Error
- Automaticity Error
- Boundary Condition Error
- Configuration Error
- Design Error
- Environment Error
- Failure on Exception
- Input Validation Error
- Origin Validation Error
- Race Condition Error
- Serialization Error

But, a critical analysis reveals several ambiguities in these classes such as Input Validation Error and Access Validation Class are defined as separate classes in spite of the fact that many a time 'Access Validation Errors' arise due to Input Validation Errors. Therefore input validation problems cause access validation problems. In order to solve such a problem of taxonomies in software vulnerability, a method of vulnerability classification based on text clustering in NVD (National Vulnerability Database) was proposed by [13], they used Cluster Overlap Index to evaluate Simple k-mean, Bisecting K-Means and Batch Som clustering algorithms. Forty-five main vulnerability clusters were selected from approximately 40,000 vulnerability records, according to Descriptor Dominance Index. An exhaustive list of various vulnerability taxonomies can be found in [1]. Their survey covers work done in security related taxonomies from 1974 until 2006. On the basis of their exhaustive surveys of vulnerability taxonomies, they proposed following basic properties of taxonomy:

- Taxonomy should be application or system specific.
- Taxonomy must be layered or hierarchical.
- First Level of Classification should be attack impact.
- Second level of Classification should be specific attack types.
- Third Level of Classification should be components-attack targets.
- Fourth Level of Classification should be system features-sources of vulnerability.
- Taxonomy classes need not be mutually exclusive.

Though above properties may or may not be the criteria of measuring the efficiency of particular taxonomy but in the absence of any metrics for measuring efficiency and effectiveness of particular taxonomy, these properties may be used to aid the process of making more efficient taxonomy. According to Ivan krusal [10], the taxonomic characteristics must satisfy following properties:

Objectivity: The feature must be identified from the object known and not form the subject knowing. The attribute being measured should be clearly observable.

Determinism: There must be clear procedure that can be followed to extract a feature.

Repeatability: Several people independently extracting the same feature for the object must agree on the value observed.

Specificity: The value of the feature must be unique and unambiguous.

Apropos to the above descriptions, the above-mentioned properties have been followed during the development of the desired taxonomic description. To satisfy the first feature we consider only those features of the distributed system which are well known and defined in literature. A case study of NVD (National Vulnerability Database, USA) [14] is one of the most popular vulnerabilities databases, is also conducted to measure the effectiveness of the taxonomy. In order to satisfy the second property i.e. determinism, we have created a decision tree for classifying design level vulnerabilities of NVD. To satisfy repeatability property, we have also included some references from other classification schemes that match our taxonomic description. The last property is not fully satisfied, as there is lots of interdependence between the features of the vulnerabilities and therefore there is chance, that one vulnerability exhibit characteristic of two classes. But utmost care is taken for limiting ambiguity to 5% only.

3- DEVELOPMENT OF TAXONOMY

The bases for the development of the successful classification are the taxonomic characteristics [2] [7]. These are well defined and largely genuine properties of the object that will be classified [10]. Therefore for classifying design level security vulnerabilities, design level security properties are identified. For identifying design level security properties, first we take the properties of the software design and relate it correspondingly to security requirements. Distributed system undertaken in consideration because security is one of the major concerns in the distributed system [20]. The definition of software architecture design for distributed systems like web systems is as follows [20]:

- The placement of the components across a network of computers, seeking to define useful patterns for the distribution of data and workload.

- The interrelationships between the components, their functional roles and the patterns of communication between them;

The initial simplification is achieved by classifying processes as server process, client processes and peer processes [20]. Therefore the first level of security of the web application is at the process level itself and second level is at interrelationship and the communication between the processes of the web application. Therefore security features are classified at two levels in terms of 'process' and 'channel or communication'. Therefore our first level of classification will depend on the two broad categories i.e., process and communication.

When it comes to security requirement for the software application, many developers and researchers have purposed a well defined set of security requirements, among those requirement, access control is identified as a superset of all security requirements. According to Matt Bishop [15], a well known security expert, 'access control matrix is the primary abstraction mechanism in computer security'. In its purest form, it can express any expressible security policy. C. Maniraman [16] has also reported access control as first issue while addressing security at design level of the software. Access Control can further be classified into three sub categories i.e. Authentication, Authorization, and Audit & Logging. Herald Terkelsen [17] in his thesis identified access control as a primary level of classification of design flaws and further categorized it into two sub-categories, authentication and authorization. WASC [18] also identifies authentication and authorization as two main attributes in software threat classification. 'Auditing and logging' is considered as third sub-category, as it aids in identifying access breaches. Therefore considering the second property of the taxonomy i.e. repeatability, we consider access control as first level of security features at the design level and therefore first level security feature can be defined as access control at process level, and access control at communication level. Access control can further be classified into three more classes i.e. authentication, authorization and 'audit and logging', as specified in the literature discussed above. The communication access control part of the design is a bit different from that of the process design as it also includes session establishment and flow of data from one computer system to another. Therefore communication level can be classified into two more classes, i.e. Session Management and Information Flow. Consequently, the first level of taxonomic description is established by us as described in Figure 1 as follows.

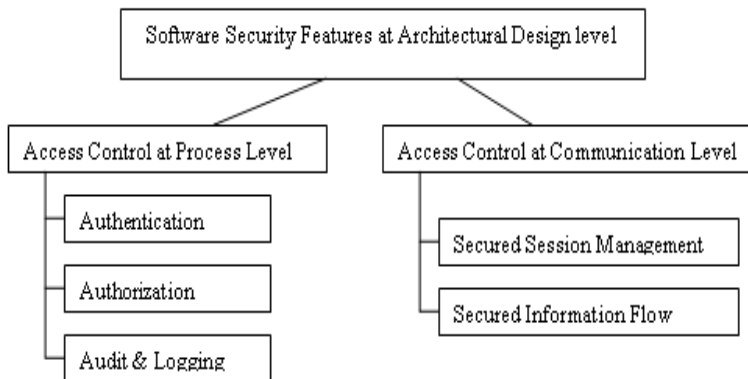


Figure 3 First level hierarchy of security features at architectural design level

CVE (Computer Vulnerabilities and Exposures) [19] is now the industry standard for ‘vulnerability and exposure’ names. Word ‘Exposure’ is included in this database to address to those vulnerabilities, which are not vulnerabilities in themselves because of indirectly facilitating attacks. Examples of such cases include error message of display system file names that help attacker to conduct attack. Use of predictable algorithms and file names also facilitate attacker to guess possible patterns. Another category that also comes under exposures is ‘not alerting’ the user about possible attacks’. While developing complex software, designers deliberately have to compromise security at several places; one way to mitigate the effect of these kinds of vulnerabilities is to alert the user about possible attacks and consequences. Therefore in this taxonomy, ‘Exposures’ is included as a first level class of vulnerabilities. Subclasses of ‘Exposures’ includes ‘Exposures in Error Messages’, ‘Predictable Algorithms or File Names’, and ‘User Alertness Leading to Access Violation’.

If we consider web application architecture diagram, process level access controls can be applied at the browser, web server, application server levels, and communication controls can be applied when there is transmission of information from one place to another. The exposures of information mainly occur at the application level. Figure 2 shows the various component of the web application with the specified location of the security controls.

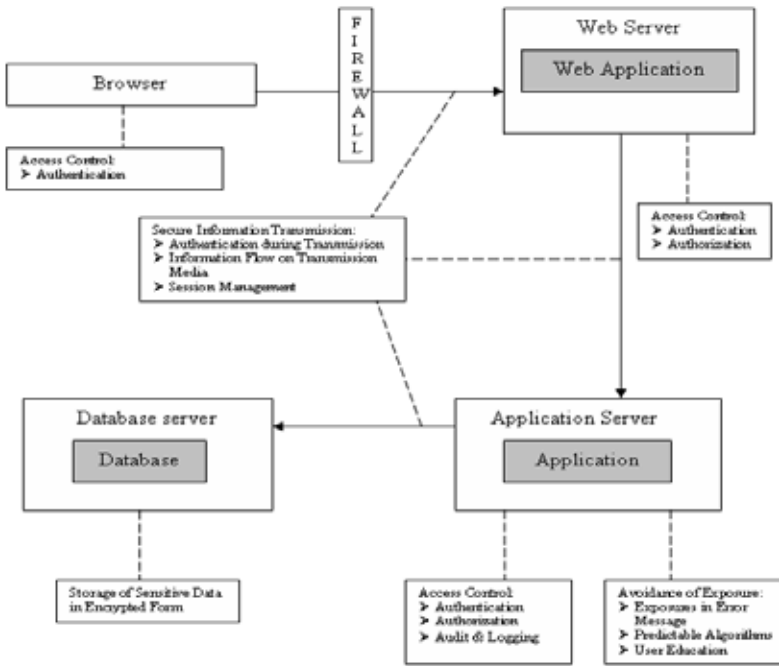


Figure 2 Web application architecture with security features

After the review of various vulnerability databases, it was found that reason of almost all the vulnerabilities were either some security feature was missing or insufficient or else wrong/incorrect. Harald Terkelsen in his thesis “Data Collection on Security Flaws Caused by Design Errors” [17], also used missing, insufficient and incorrect keywords in classifying classes of design flaw. Therefore we choose keyword to further categorize the classes of security features, as missing, insufficient and wrong. Table 1 gives the complete proposed taxonomy of design level vulnerabilities.

Most of the software vulnerability taxonomies proposed till date were targeting either whole class of software vulnerabilities or vulnerabilities in particular software like web browser, network protocols etc. There are few vulnerability taxonomies that addressed the causes of vulnerabilities at different SDLC levels. None of the vulnerability taxonomies address main causes of vulnerabilities at the root of the software i.e. requirements and design. Here we are proposing a taxonomy that can be used to identify causes of security feature vulnerabilities in the early phases of the SDLC i.e. requirement and design phases of SDLC.

Table 1 Taxonomy of Design Level Vulnerabilities

First Level	Second Level	Third level	Fourth Level
Access Control	Access Control at Process Level	Authentication	Missing Authentication procedure
			Insufficient Authentication procedure
			Wrong Authentication procedure
		Authorization	Missing Authorization procedure
			Insufficient Authorization procedure
			Wrong Authorization procedure
		Audit & logging	Missing Audit and logging
			Insufficient Logging or Audit of information
			Wrong Audit or Logging of information
	Access Control at Communication Level	Secured Session Management	Missing Secured Session management
			Insufficient Secured Session Management
			Wrong Secured Session Management
		Secured Information Flow	Missing Encryption of Sensitive Data During Transmission
			Insufficient Encryption of Sensitive Data during Transmission
			Wrong Encryption of Sensitive Data during Transmission
	Exposures leading to Access Violation	Exposures in Error Message	Missing Secured Error Message
			Insufficient Secured Error Message
			Wrong Secured Error Message
		Predictable Algorithm /sequence numbers/file names	Missing Randomness in the Random Sequence Ids
			Insufficient Randomness in the Random Sequence Ids
			Wrong Randomness in the Random Sequence Ids or Wrong Choice of File Name
User Alertness		Missing User Alerting Information	
		Insufficient User Alerting Information	
		Wrong User Alerting Information	

3-1 ACCESS CONTROLS AT PROCESS LEVEL

Process level access control vulnerabilities involve those arising in each process of the distributed system. It does not include vulnerabilities at

interconnection or communication level of the component. The process level vulnerabilities can further be classified under following classes:

- i). Authentication
 - Missing Authentication procedure
 - Insufficient Authentication procedure
 - Wrong Authentication procedure
- ii). Authorization
 - Missing Authorization procedure
 - Insufficient Authorization procedure
 - Wrong Authentication procedure
- iii). Audit and Logging
 - Missing Logging and Audit
 - Insufficient Logging or Audit of information
 - Wrong Logging or Audit of information

3-2 ACCESS CONTROL AT COMMUNICATION LEVEL

Communication level access control vulnerabilities involve those vulnerabilities that deal with access control only at the interconnection or communication level of the component. The communication level vulnerabilities can further be classified under following classes:

- i). Secured Session Management
 - Missing Secured Session Management
 - Insufficient Secured Session Management
 - Wrong Secured Session Management
- ii). Secured Information Flow
 - Missing Encryption of Sensitive Data During Transmission
 - Insufficient Encryption of Sensitive Data During Transmission
 - Wrong Encryption of Sensitive Data During Transmission

3-2 EXPOSURES LEADING TO ACCESS VIOLATION

This class of security vulnerabilities includes those vulnerabilities that do not directly facilitate attack but indirectly expose information that helps attacker to exploit some other vulnerability. The classification of various kinds of exposures is given as follows:

- i). Exposure in Error Messages
- ii). Predictable Algorithms /Sequence Numbers/File Names
- iii). User Alerting Information

In order to measure the classification capability of the taxonomy, a case study is conducted on the National Vulnerability Database (NVD, USA)[14].Details of the implication and output is describe as follows.

4- NVD DESIGN LEVEL SECURITY VULNERABILITY CLASSIFICATION: A CASE STUDY

We are in the process of using proposed taxonomy to develop a process of identifying proper mitigation techniques at the design level of the software. The taxonomy is a critical component of the process, because it provides an ordered classification of known vulnerabilities. Furthermore, it affords us a distinct advantage by simplifying the mapping of security requirement to the known vulnerabilities. We can simply check which security requirement can have which vulnerability and also check their severity that can be further used to calculate the risk at the design level. Realizing that it is important to implement the taxonomy on updated vulnerability databases, we are working on developing a process that can automatically feed data from the various vulnerability databases.

This process is important because of the dynamic nature of vulnerabilities and software security. It will ensure the longevity and usefulness of our taxonomy by classifying data of new vulnerabilities. We also recognize the importance of assessing the effectiveness of the taxonomy against real world exploits. To this end, we intend to classify design level vulnerabilities derived from NVD [14] till Feb, 2010. In addition to verifying the taxonomy, classifying vulnerabilities from NVD will help us in improving the taxonomy. NVD (National Vulnerability Database) is widely used security vulnerability database. Four Hundred and Twenty Seven vulnerabilities (427) were defined as design level security vulnerabilities in this database, as on 19 February 2010. In our taxonomy we are considering only those design issues that exist in the security feature design. On the basis of above identified taxonomic criterion, a decision tree is created to decide which vulnerability fall under which class of taxonomy.

To avoid complexity, decision tree is divided into four phases as shown in Figure 3, Figure 4, Figure 5 and Figure 6 respectively. Phase I is a summary of whole classification process. In this phase there are four decision levels, which are explained as follows:

1. Is the vulnerability defined as design level vulnerability in NVD (National Vulnerability Database)?
Explanation: If the answer is yes then tree will traverse to the next decision level otherwise it terminates, which indicates that it is not a design level vulnerability in NVD.
2. Does the vulnerability indicate that it is just exposing some confidential information?
Explanation: If the answer is yes then tree will traverse Phase IV, otherwise it will proceed to next decision level.

3. Does the vulnerability definition indicate transfer of information from one computer to another?

Explanation: If the answer is yes then tree will traverse Phase III, otherwise it will proceed to next decision level.

4. Is the vulnerability at process level?

Explanation: If the answer is yes then tree will traverse Phase II, otherwise it terminates, which indicates that it is not a design level vulnerability in NVD.

Phase II, phase III, and phase IV represent three major classes of the taxonomy i.e. i).Access Control at the Process Level ii).Access Control at the Communication level and iii).Exposure Leading to Access Violation. In Figure 3, Figure 4 and Figure 5, decision trees are shown with all the levels of decisions. We have successfully classified 121 vulnerabilities using these decision trees as reported in section 5.

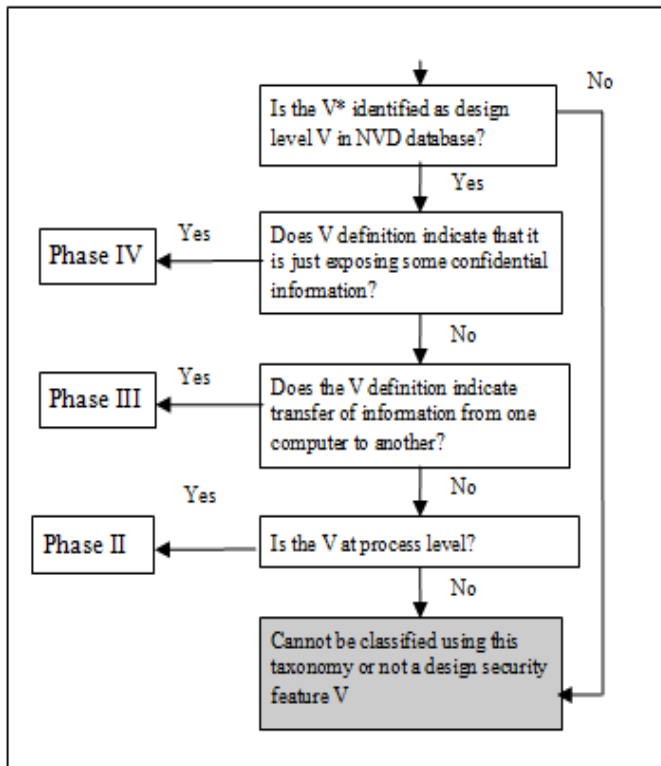


Figure 3 Phase I Decision Tree
 * (Note: V stands for vulnerability)

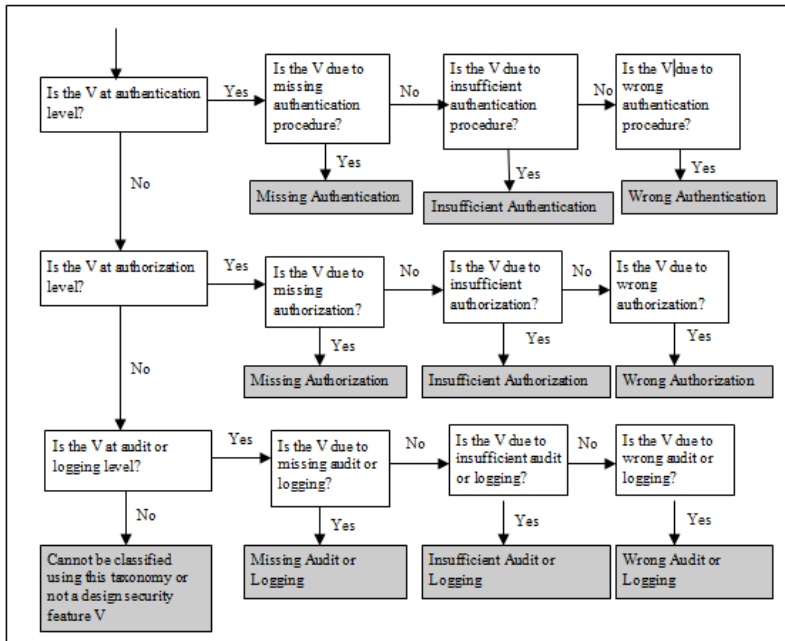


Figure 4 Phase II Decision Tree

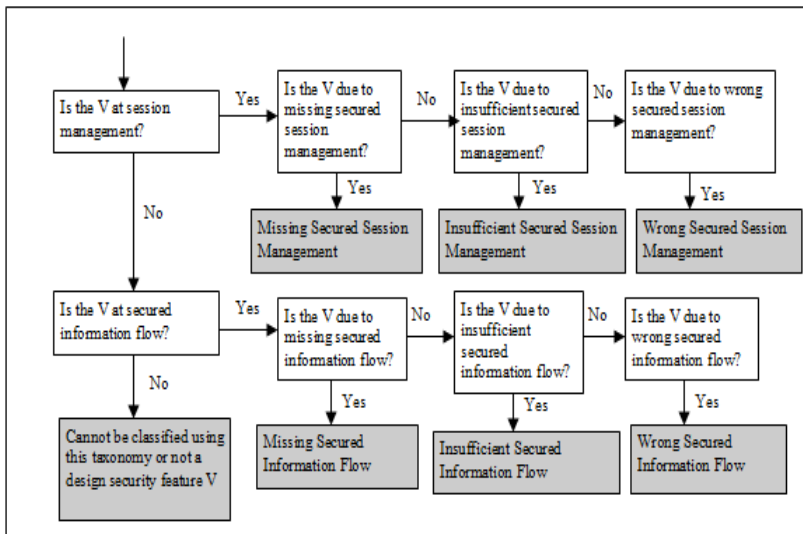


Figure 5 Phase III Decision Tree

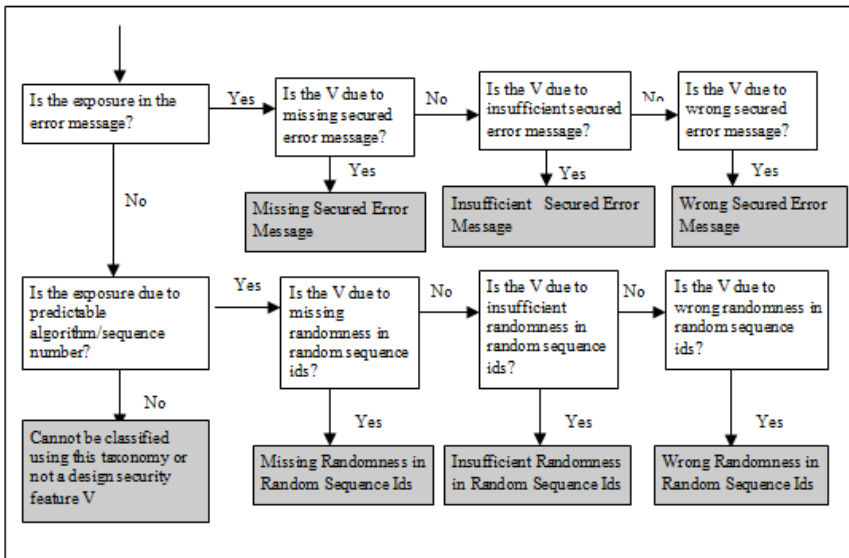


Figure 6 Phase IV Decision Tree

Table 2 describes the full taxonomy at the top level that has been used to classify 121 design level vulnerabilities of National Vulnerability Database, during the reporting period of 2003-2009. NVD uses CVSS (Common Vulnerabilities Scoring System) formulae for the estimation of severity; and the same as vulnerability score is specified. Frequency and Severity of vulnerability are the two major variables considered highly pertinent in ‘Risk Estimation Process’. Table 2 presents the taxonomy with frequency and the severity of NVD design level vulnerabilities which can provide useful input in the risk estimation process.

Table 2 Taxonomic Classification of NVD Design Level Vulnerabilities

First Level Feature	Second Level Feature	Third level Feature	Vulnerabilities Reported in NVD	Severity	Average Severity
Access Control at Process	Authentication	Absence of Authentication Procedure	2007-6480	9.4	8.9
			2007-6333	10	
	Insufficient Authentication Procedure	2008-6232	7.5	5.8	
		2008-6231	7.5		
		2002-2293	4.6		
		2009-0217	5.0		
		2008-5618	5.0		
		2007-4556	6.8		
	Wrong Authentication	2008-1160	7.5		
		2008-1079	4.3		

		Procedure	2003-1507	10.0	7.9	
			2002-2402	10.0		
			2009-3843	10.0		
			2008-6228	7.5		
			2008-1256	10		
			2007-6547	6.8		
			2009-4197	4.7		
		Authoriza tion	Absence of Authorization Procedure	2007-5424	7.5	6.5
				2007-5129	5.0	
				2007-4609	6.4	
	2007-3378			6.8		
	2007-3168			7.8		
	2007-3089			4.3		
	2007-0328			9.3		
	Insufficient Authorization Procedure	2006-1620	7.2	6.1		
		2006-1283	2.1			
		2004-2759	4.8			
		2003-1428	4.6			
		2002-1502	6.8			
		2003-1544	7.8			
		2002-2393	7.1			
		2009-0637	5.1			
		1999-0668	5.8			
		2008-1729	9.3			
	Wrong Authorization Procedure	2007-6332	5.8	5.6		
		2007-4889	6.8			
		2008-5503	4.9			
		2007-6358	4.9			
		2007-5718	6.9			
		2004-2697	7.2			
		2002-2267	2.1			
		2002-2274	6.8			
	Auditing & Logging	Missing Audit or logging of Information	2007-6724	10	7.8	
2007-6722			5.0			
2007-4879			2.6			
Missing Audit or Logging of Information	2008-4315	7.5	4.9			
	2008-1203	7.5				
	2007-6098	8.5				
Insufficient Audit or Logging of Information	2008-1998	5.0	6.0			
	2006-6301	5.0				
	2006-6302	4.7				
Wrong Audit or Logging of Information	2008-0887	4.3	3.0			
	2007-5715	6.4				
	2003-1363	7.2				
Access Control at Transmis sion Level	Secured Informati on Flow	Missing Encryption of Sensitive Data during Transmission	2008-0441	2.1		
			2007-5373	2.1		
			2007-0413	4.4		
			2003-1437	2.1		
			2003-1454	5.0		
		2003-1476	2.1			

		In sufficient Encryption of Sensitive Data during Transmission	2007-1858	2.6	5.9	
			2007-6330	10		
			2007-5034	4.3		
			2005-6330	10.0		
			2007-6591	4.3		
			2007-6592	4.3		
		Wrong use of Encryption		Nil	0.0	0.0
	Secured Session Management	Missing Secured Session Management		2008-5086	7.2	6.4
				2007-4702	9.3	
				2007-4703	10.0	
				2007-4704	10.0	
				2007-5653	9.3	
				2005-3774	5.0	
				2007-5273	2.6	
				2007-5274	2.6	
				2007-5276	4.3	
				2007-5277	4.3	
		2007-5232	4.3			
		Insufficient Secured Session Management		2008-1531	4.3	4.8
2007-5281				5.0		
2007-6286				5.0		
Wrong Secured Session Management		Nil	0.0	0.0		
Avoidable Exposure	Exposure in Error Messages	Missing Secured Error Message	2008-1261	5.0	5.0	
			Insufficient Secured Error Message	2003-1399	1.9	3.5
				2008-4232	5.0	
		Wrong secured Error Message		2007-4872	5.0	
		2007-6271	5.0	5.0		
		Predictable Algorithm/sequence numbers/file names	Missing Randomness in the Random Sequence Ids		2007-3871	9.3
	2008-3630				6.4	
	Insufficient Randomness in the random Sequence Ids			2009-2165	7.5	7.0
				2008-6564	7.6	
2007-6546				6.4		
2007-4733		9.3				
2008-2019		7.5				
2008-2020	6.8					
2008-0299	4.3					
2007-2930	4.3					
2008-1390	9.3					
Wrong		2008-1146	6.8	6.8		
		2008-1147	6.8			

	Implementation of Randomness or using Wrong Choice of File names	2008-1148	6.8	6.1	
		2008-1796	4.9		
		2007-6061	5.0		
		2002-2392	6.4		
	User Alerting Information	Missing User Alerting Information	2006-0374	7.5	5.0
			2002-2307	5.0	
			2007-3754	4.3	
			2007-4590	3.3	
		Insufficient User Alerting Information	2008-0594	5.0	5.0
			2008-1902	5.0	
		Wrong User Alerting Information	Nil	0.0	0.0

5- DISCUSSION AND IMPLICATIONS

NVD has defined 427 vulnerabilities as design level vulnerabilities till now. After classifying these vulnerabilities using the proposed taxonomy, it was found that only 121 are actually design level security specific vulnerabilities and several do not have proper explanation. As we are considering only application design level vulnerabilities, we have also excluded operating system design level vulnerabilities. The severity of each vulnerability is also defined in NVD, which can be helpful in identifying the criticality of vulnerability, and hence aid the process of risk analysis. The classification of vulnerabilities in the security features of the software shows that most of the vulnerabilities occur in the access control security feature of the software and authorization sub-class have maximum number of occurrence of vulnerabilities. 'Access control at Process level' super class alone is responsible for 46% vulnerabilities in the security features. Therefore access control at Process level is the most vulnerable part in the software and need special attention by the software designers.

'Access Control at Communication Level' constitutes 26 % of total vulnerabilities in the security features. 'Secure Session management' class is most prevailing class in transmission, as most of the attacks that occur during communication are of session hijack. 'Exposures Leading to Access Violation' class constitutes those exposures that are not vulnerabilities but they facilitate attacks. The dictionary of common vulnerabilities and exposures (CVEs), sponsored by the department of homeland security CVE (Common Vulnerabilities and Exposures) [19] includes exposures that indirectly leads to attacks. In our classification 'Exposures' constitute 28% of total security feature vulnerabilities, which shows that large number of attacks occurs due to unnecessary exposure of information that facilitates attack. In addition to being used for assessing software security at design level, the taxonomy can be used to classify vulnerabilities in other phases also. First three levels of the taxonomy can also be used to classify security specific requirements of a under development software. After analysing available current vulnerability databases, security requirement analyst can classify vulnerabilities in the

various classes and draw a decision tree. After classification of requirements, a class-wise mitigation measure can be evolved or devised to remove vulnerability in the requirement phase itself. Furthermore, the information embodied by the taxonomy can be used to improve requirements for creating secure software or to develop checklists for designing and implementing secure software and so forth.

6- CONCLUSION AND FUTURE WORK

Security specific vulnerabilities can be classified on several bases such as respective impact, severity, cause and attack vectors. Proposed taxonomic description enables an analyst to classify only those security features that can be improved and mitigated by the software designers during the design phase. Cause of vulnerabilities is taken as a prime criterion of classification. The proposed taxonomic description may be used by the software designer right from the requirement to coding phase of SDLC to identify possible vulnerable areas. Among the further extension of this work may include the identification of mitigation mechanisms that can be directly adopted by the designers to mitigate the typical class of vulnerabilities. Classification of security features of the software design phase on the basis of impact and attack vector is another prompt future work as a straight forward extension.

REFERENCES

- [1] V.M. Ijure, and R. D. Williams, "Taxonomies of Attacks and Vulnerabilities," IEEE Communications 2008, Volume 10, No. 1, pp.6-19, 2008.
- [2] R. L. Glass and I. Vessey, "Contemporary Application Domain Taxonomy," IEEE Software, Vol.12, Issue 4, pp.63-67,1995.
- [3] Arce, "More Bang for the Bug: An Account of 2003's Attack Trends," IEEE Sec. & Privacy, vol. 2, no. 1, pp.66-68, Feb 2004.
- [4] L. Falk, A. Prakash and K. Borders, "Analyzing websites for user-visible security design flaws," ACM International Conference Proceeding Series; Vol. 337, pp.117-126, 2008.
- [5] N. A. S. Abdullah, R. Abdullah, M. H. Selamat, A. Jaafar "Software Security Characteristics for Function Point Analysis", Proc. of IEEE International Conference on Industrial Engineering and Engineering Management, held on 8-11 Dec 2009, Hong Kong, pp.394-397, 2010.
- [6] C. Groves, "The What, Why and How of Primate Taxonomy," International Journal of Primatology, Vol. 25, No. 5, Australia, pp.1105-1126, October 2004.

- [7] G.G. Simpson, "The Principles of Classification and a Classification of Mammals," New York. Columbia biological series; No. 20, Columbia University Press. New York, pp 247, 1961.
- [8] J. D. Howard and T. A. Longstaff, "A Common Language for Computer Security Incidents," Sandia tech. rep. SAND98-8667, Oct. 1998.
- [9] U. Lindquist and E. Jonsson, "How to Systematically Classify Computer Security Intrusions," Proc. IEEE Symp. Sec. and Privacy, pp.154–63, May 1997.
- [10] I. V. Krsul, "Softwrae Vulnerability Analysis," Ph.D Thesis submitted at Purdue University, USA, 1998.
- [11] A. Bazaz, J. D. Arthur, "Towards a Taxonomy of Vulnerabilities," HICSS, pp.163a, 40th Annual Hawaii International Conference on System Sciences (HICSS'07), 2007.
- [12] Z. Chen, Y. Zhang and Z. Chen, "A Categorization Framework for Common Computer Vulnerabilities and Exposures," The Computer Journal, Advance Access, pp13, May 2009.
- [13] S. Tang, M. Zhang, J. Tian "Text Clustering on National Vulnerability Database," Second International Conference on Computer Engineering and Applications, pp.295-299, 2010.
- [14] National Vulnerabilities Database, USA, URL: <http://nvd.nist.gov/>, (Accessed on 19th Feburary,2010)
- [15] M. Bishop, "Computer Security: Art and Science", Pearson Education, Singapore, pp.85, 2003.
- [16] C. Muniraman, "A Practical Approach To Include Security In Software Development", University Of Houston-Victoria, Issues In Information Systems, Volume VIII, No. 2 ,pp.193-199,2007.
- [17] H. Terkelsen, "Data Collection on Security Flaws Caused by the Design Errors", Thesis, Department of Computer Science, Gjovik University Collage, pp.36, 2006.
- [18] WASC, Threat Classification 2010, url:<http://projects.webappsec.org/Threat-Classification> (Accessed on 19th February, 2010).
- [19] Common Vulnerabilities and exposures, The MITRE Corporation, <http://cve.mitre.org/> (Accessed on 19th February, 2010).
- [20] G. Coulouris, J. Dollimore, T. Kindberg, "Distributed Systems: Concepts and Design", Fourth Edition, Pearson Education, 2007.