ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

Prevention of SQL Injection Attack on Web Applications

Shakti Kumar[1], Subhendu Dey[2], R.Karthikeyan[3], K.G.S. Venkatesan[4]

Dept. of C.S.E., Bharath University, Chennai, Tamil Nadu, India[1].

Dept. of C.S.E., Bharath University, Chennai, Tamil Nadu, India[2].

Associate Professor, Dept. of C.S.E., Bharath University, Chennai, India[3].

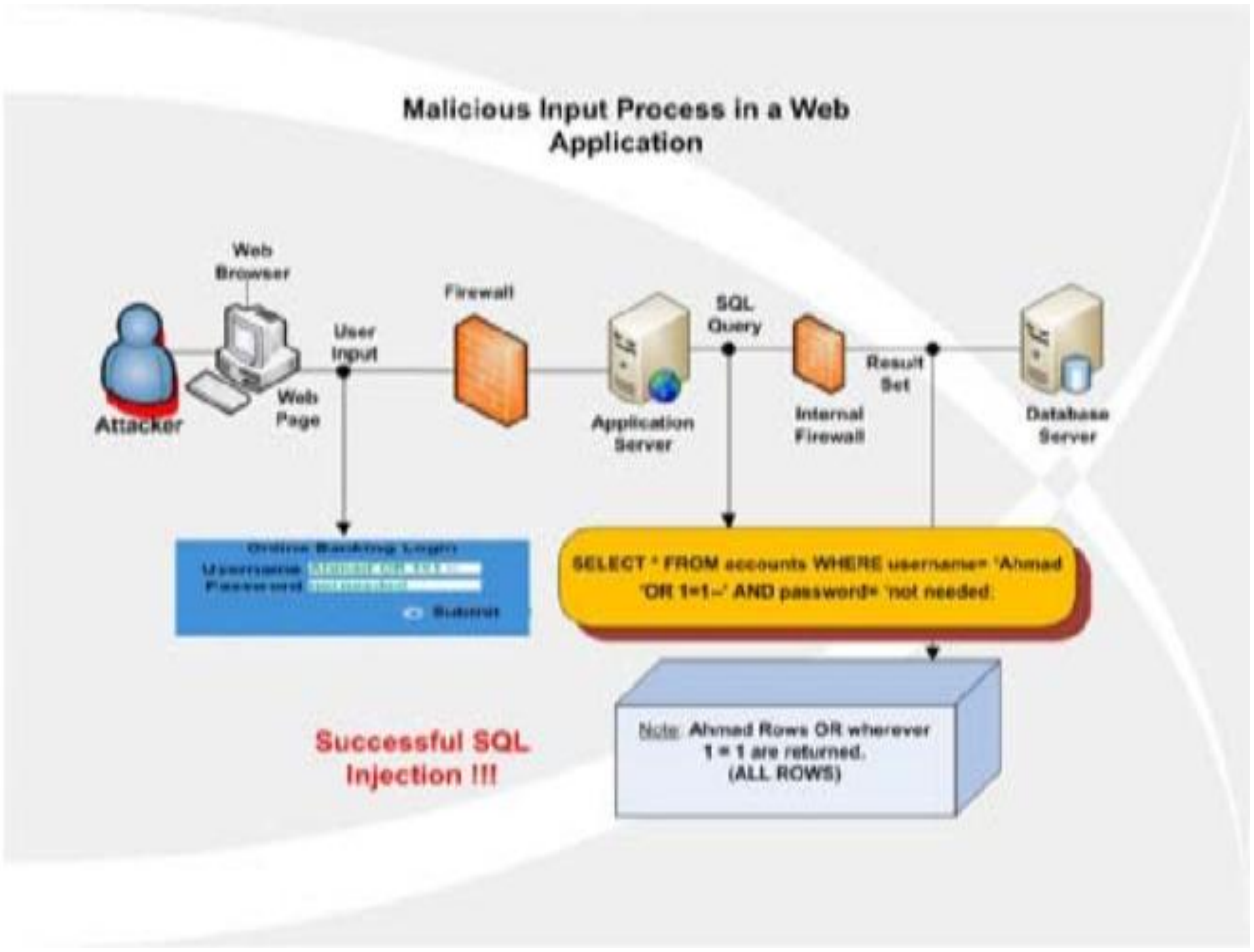
Associate Professor, Dept. of C.S.E., Bharath University, Chennai, Tamil Nadu, India[4].

ABSTRACT: SQL injection is a technique where the attacker injects an input in the query in order to change the structure of the query intended by the programmer and gaining the access of the database which results modification of the user's data. In the SQL injection it exploits a security vulnerability of data occurring in database layer of an application. SQL injection attack is the most common attack in websites in these days. Some malicious codes get injected to the database by unauthorized users and get the access of the database due to lack of input validation. Input validation is the most critical part of software security that is not properly covered in the design phase of software development life-cycle resulting in many security vulnerabilities. This paper presents the techniques for detection and prevention of SQL injection attack. There are no full proof defences available against such type of attacks. In this paper some predefined method of detection and modern techniques are discussed. This paper also describes countermeasures of SQL injection..

KEYWORDS: Web Application, SQLIA, detection, prevention, vulnerabilities, and web architecture.

I. INTRODUCTION

Now a days, Web application is widely used in various applications it is the reliable and efficient solution to the challenges of communicating and conducting the various organisation, business or commerce over the internet [2]. Now each and every important assignment is done by using the web application which is connected through the internet. For example electricity bill, online shopping, gaming, banking, messaging, shopping, conferences, etc. So the increase of web application involving the various security issues in the web world. The SQLIA (structured query language injection attack) is a code injection attack technique commonly used for attacking websites in which an attacker injects some SQL codes in place of the original codes to get access the database. The open web application security paper (OWASP) ranks SQLI as the most widespread website security risk in 2011. The National Institute of Standards and Technology's National vulnerability Database reported 289 SQL vulnerabilities in websites including those of IBM, HP, and MICROSOFT [1]. In December 2011, SANS Institute security experts reported a major SQL injection attack that affects approximately 160000 websites using Microsoft's Internet Information Services (IIS), ASP.NET, and SQL Server Frameworks. There are variety of techniques are available to detect SQLIA. The most preferred are Web Framework, Static Analysis, Dynamic Analysis, combined Static and Dynamic Analysis and Machine Learning Technique [3]. Web Framework provides filters to filter special characters but other attacks are not detected. Static Analysis checks the input parameter type, but it fails to detect attacks with correct input type. Dynamic Analysis technique is capable of scanning vulnerabilities of web application but is not able to detect all types of SQLIA. Combined Static and Dynamic Analysis includes the benefit of both, but this method is very complex in order to proceed. Machine Learning method can detect all types of attacks but results in number of false positives and negatives [5].



SELECT * FROM accounts WHERE username= 'Ahmad 'OR 1=1--' AND password= 'not needed'

Note: Ahmad Rows OR wherever 1 = 1 are returned. (ALL ROWS)

Fig.1. malicious input process in a web application to generate SQL injection

II. CLASSIFICATION OF SQLIA

TAUTOLOGY **:**

In the tautology attack the attacker tries to use a conditional query statement to be evaluated always true. Attacker uses WHERE clause to inject and turn the condition into a tautology which is always true. The simplest form of tautology Example SELECT *FROM Accounts WHERE user=''or1=1— 'AND pass=''AND eid= The result would be all the data in accounts table because the condition of the WHERE clause is always true [9].

A. Illegal/Logical Incorrect queries **:**

When a query is rejected an error message is returned from the database including useful debugging information. This information helps attackers to make move further and find vulnerable parameters in the application and consequently database of the application. SELECT * FROM Accounts WHERE user=' ' AND pass=' 'AND aid =convert(nit,(SELECT TOP 1name FROM objects WHERE x type='u')) [11]. In the example the attacker attempts to convert the name of the first user defined table in the metadata table of the database to 'int'. This type conversion is not legal therefore the result is an error which reveals some information that should not be shown [12].

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

B. Union queries :

In this type of queries unauthorized query is attached with the authorized query by using UNION clause. Example SELECT * FROM Accounts WHERE user='' UNION SELECT *FROM Students—'AND pass=''AND eid= The result of the first query in the example given above is null and the second one returns all the data in students table so the union of these two queries is the student table [15].

C. Piggy-Backed query :

In the query attack attacker tries to add an additional queries in to the original query string .In this injection the intruders exploit database by the query delimiter, such as ";", to append extra query to the original query Example SELECT*FROM Accounts WHERE user='';drop table Accounts—'AND pass=' ' AND eid= The result of the example is losing the credential information of the accounts table because it would be dropped branch from database [17]. In this type of attack, intruders change the behaviour of a database of application. These are the well known types of inference.

D. Blind Injection **:**

This is little difficult type of attack for attacker. During the development process sometime the developer hides some error details which help the attacker to compromise with database. In this situation the attacker face the generic page provided by developer in place of an error message Example SELECT * FROM Accounts WHERE user='user1'AND1=1 - - 'AND pass=' 'AND eid= During injection it is always evaluated as true if there are no any error message, and the attacker realizes that the attack has passed user parameter is vulnerable to injection [20].

E. Timing attack :

In the Timing attack the attacker gathers information about the response time of the database. This technique is used by executing the if-then statement which results the long running query or time delay statement depending upon the logic injected in database and if the injection is true then the "WAITFOR" keyword which is along with the branches delays the database response for a specific time [22]. Example SELECT * FROM Accounts WHERE user='user1' AND ASCII (SUBSTRING ((SELECT TOP 1 name FROM sysobjects),1,1))>X WAITFOR DELAY '000:00:09'- -'AND PASS=' ' AND eid= In the example the attacker trying to find the first character of the first table by comparing its ASCII value with X . if there is a 9 second delay he realize that the answer to his question is yes. So by continuing the process the name of the first table would be discovered [25].

F. Alternate encoding :

In this type of attack the regular strings and characters are converted into hexadecimal, ASCII and Unicode. Because of this the input query is escaped from filter which scans the query for some bad character which results SQLIA and the converted SQLIA is considered as normal query. Example SELECT * FROM Accounts WHERE user='user1'; exec(char(0x8774675u8769e)) - -' AND pass=' ' AND eid= The example char () function and ASCII hexadecimal encoding are used [29]. The functions will get integer number as a parameter and return as a sample of that character. In the example it will return "SHUTDOWN", so whenever the query is interpreted the SHUTDOWN command is executed [30].

G. Stored procedure:

Stored procedure is the built in extra abstraction layer on the database defined by the programmer. By using the stored procedure the user can store its own function according to the need. It is extending the functionality of database and interacting with the system operating system. Then the attacker tries to identify the underlying database in order to exploit the database information

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

III. DETECTION SQLIA

Several ways to detect the SQLIA vulnerabilities are:

A. Code Based Detection Techniques :

This approach generally occupies for developing test suit based on codes for detecting the SQLI vulnerabilities .But the suit does not find vulnerable program points explicitly. SQL Unit Gen is a prototype tool that uses static analysis tool to generate the user input to database access point and generate unit test report contacting SQLIA patterns for these points. MUSIC (mutation based SQL injection vulnerability checking) it uses nine mutation operators to replace original queries with mutated queries. This tool automatically detects the mutated queries and runs the test tool after it generated the test results after the detection [24].

B. Concrete Attack Generations :

This type of approach uses state of art symbolic execution techniques to automatically generate test inputs that expose SQLI vulnerability in Web program. The symbolic execution based approaches use constraint solvers that can only handle numeric operation. Because inputs of Web applications are string by default .If a constraint solver can solve myriad string operations applied to inputs, developers could use symbolic execution to both detect the vulnerability of SQL statements that use inputs and generate concrete inputs that attack them [29].

C. Taint-based vulnerability detection :

SQLIA can be avoided by using static and dynamic technique to prevent tainted data from affecting untainted data, such as programmer –defined SQL query structures. Several of researchers have applied prominent static analysis techniques such as flow sensitive analysis, context sensitive analysis, alias analysis and interprocedural dependency analysis, to identify input sources and database access points and check whether every flow from a source to a sink is subject to an input validation and /or input sanitization routine, but these approaches have some limitations. They do not consider input validation using prediction, fail to specify vulnerability patterns. Gary Wassermann and Zedong Su used context free grammar to model the effects of input validation and sanitization routines .Their techniques checks whether SQL queries syntactically confine the string values returned from those routines and, if so, automatically concludes that the routines used are correctly implemented [30].

IV. PREVENTION SQLIA

A. Defensive coding :

Developers have approached a range of code based development practices to counter SQLIA. These techniques are generally based on proper input filtering, potentially harmful character and rigorous type checking of inputs [31].

B. Manual defensive coding practices:

Based on the security reports such as OWSAP's SQL cheat sheet and Chris Anley's white paper provide useful manual defensive coding guidelines. Parameterized queries or stored procedures: The attacker take advantage of dynamic SQL by replacing the original queries and create some parameterized query in database [32]. These attacks force to developer for first define the SQL code structure before including parameters in query. Because parameters are bound to the defined SQL structure, thereafter it is not possible to inject additional SQL code Escaping: If dynamic queries cannot be avoided, escaping all user-supplied parameters is the best option. Then the developer should identify the all input sources to define the parameter that need escaping, follow database-specific escaping procedures, and use standard defining libraries instead of the custom escaping methods. Data type validation: After following the steps for the parameterized query and escaping the developer must properly validate the input data type. The developer must define the input data type is string or numeric or any other type and input data given by user is incorrect then it could easily reject. White list filtering: Some of the special character which is normally used during injection .so the developer should characterise such special character as the black list filtering. The filtering approach is suitable for the well structured data. Such as email address, dates, etc. and developer should keep a list of legitimate data patterns and accept only matching input data [33].

C. SQL DOM:

The manual defensive coding is the best way to avoid the SQLIA. The approach SQL DOM is introduced by Russell McClure and Ingolf Kruger. In the SQL DOM uses the encapsulation of database queries to provide a safe way to avoid the SQLIA problem by changing the query building process from one that uses string concatenation to a systematic one that uses a type checked API [39]. In the process a set of classes that enables automated data validation and escaping. Developers provide their own database schema and construct SQL statement using its API's. It is especially useful when the developer needs to be using the dynamic SQL in place of the parameterized queries for getting flexibility [34].

D. Runtime prevention:

Runtime prevention may be more complex than the defensive coding .Because some of the approaches require code instrumentation to enable runtime checking. But it is able to prevent from all SQLIA [37].

E. Randomization:

The approach is proposed by Boyd and Keromytis in which randomized SQL query language is used, pointing a particular CGI in an application, where a proxy server used in between the SQL server and Web server. It sends SQL query with a randomized value to the proxy server, which is received by the client and derandomized and sends it to the server. This technique has two main advantages is security and portability. But if the random value is predicted then it is not useful [41].

F. Learning based prevention:

This approach is based on a runtime monitoring system deployed between the application server and database server, it intercept all queries and check SQL keywords to determine whether the queries syntactic structure are legitimate before the application sends them to the database programming and also the applied mathematical tools account for the science half, the terribly talent in analysis and abstract model one. The three Basics of electronic network Simulation seven formulation typically represents the art portion. A protracted list of steps in death penalty a simulation method, as given in [8], appears to replicate this standard claim [40].

V. CONCLUSIONS

In this paper various types of SQL injection mechanism, detection type and prevention techniques are discussed .we found that there is no one complete foolproof solution to database security and have some issues hard to eliminate .Any organization that attempts to secure a database system, must consider the security of the overall environment including the communication channel, user access methods, the database, and any application which is used to access the database .As all we can say a well thought –out combination of hardware and software solutions with modern database security approach need to be implemented to make modern database system more secure.

VI. ACKNOWLEDGEMENT

The author would like to thank the Vice Chancellor, Dean-Engineering, Director, Secretary, Correspondent, HOD of Computer Science & Engineering, Dr. K.P. Kaliyamurthie, Bharath University, Chennai for their motivation and constant encouragement. The author would like to specially thank **Dr. A. Kumaravel**, **Dean , School of Computing**, for his guidance and for critical review of this manuscript and for his valuable input and fruitful discussions in completing the work and the Faculty Members of Department of Computer Science &Engineering. Also, he takes privilege in extending gratitude to his parents and family members who rendered their support throughout this Research work.

REFERENCES

1. W. G. Halfond and A. Orso, "AMNESIA: Analysis and Monitoring for NEutralizing SQL-Injection Attacks", in Proceedings of the IEEE and ACM International Conference on Automated Software Engineering (ASE 2005), Long Beach, CA, USA, Nov 2005.
2. Justin Clarke, SQL Injection Attacks and Defense, Second Edition ,Syngress Publication, July 2, 2012,ISBN-13: 9781597494243
3. Inyong Lee , Soonki Jeong Sangsoo Yeoc, Jongsub Moond, "A novel method for SQL injection attack detection based on removing SQL query attribute", Journal Of mathematical and computer modeling, Elsevier 2011.
4. W.G.J. Halfond, J. Viegas, and A. Orso, "A Classification of SQL Injection Attacks and Countermeasures," Proc. Int'l Symp. Secure Software Eng.(ISSSE 06),IEEECS, 2006; www.cc.gatech.edu/fac/Alex.Orso/papers/halfond.viegas.orso.ISSS E06.pdf.
5. S. W. Boyd and A. D. Keromytis,"SQLrand: Preventing SQL Injection Attacks", in Proceedings of the 2nd Applied Cryptography and Network Security Conference, pages 292–302, Jun. 2004.
6. K.G.S. Venkatesan. Dr. V. Khanna, Dr. A. Chandrasekar, "Autonomous System(AS) for mesh network by using packet transmission & failure detection", Inter. Journal of Innovative Research in computer & comm. Engineering, Vol. 2, Issue 12, PP. 7289 – 7296, December - 2014.
7. K.G.S. Venkatesan and M. Elamurugaselvam, "Design based object oriented Metrics to measure coupling & cohesion", International journal of Advanced & Innovative Research, Vol. 2, Issue 5, PP. 778 – 785, 2013.
8. Teerawat Issariyakul • Ekram Hoss, "Introduction to Network Simulator NS2".
9. S. Sathish Raja and K.G.S. Venkatesan, "Email spam zombies scrutinizer in email sending network Infrastructures", International journal of Scientific & Engineering Research, Vol. 4, Issue 4, PP. 366 – 373, April 2013.
10. G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," IEEE J. Sel. Areas Communication., Vol. 18, No. 3, PP. 535–547, Mar. 2000.
11. K.G.S. Venkatesan, "Comparison of CDMA & GSM Mobile Technology", Middle-East Journal of Scientific Research, 13 (12), PP. 1590 – 1594, 2013.
12. P. Indira Priya, K.G.S.Venkatesan, "Finding the K-Edge connectivity in MANET using DLTRT, International Journal of Applied Engineering Research, Vol. 9, Issue 22, PP. 5898 – 5904, 2014.
13. K.G.S. Venkatesan and M. Elamurugaselvam, "Using the conceptual cohesion of classes for fault prediction in object-oriented system", International journal of Advanced & Innovative Research, Vol. 2, Issue 4, PP. 75 – 80, April 2013.
14. Ms. J.Praveena, K.G.S. Venkatesan, "Advanced Auto Adaptive edge-detection algorithm for flame monitoring & fire image processing", International Journal of Applied Engineering Research, Vol. 9, Issue 22, PP. 5797 – 5802, 2014.
15. K.G.S. Venkatesan. Dr. V. Khanna, "Inclusion of flow management for Automatic & dynamic route discovery system by ARS", International Journal of Advanced Research in computer science & software Engg., Vol.2, Issue 12, PP. 1 – 9, December – 2012.
16. Needhu. C, K.G.S. Venkatesan, "A System for Retrieving Information directly from online social network user Link ", International Journal of Applied Engineering Research, Vol. 9, Issue 22, PP. 6023 – 6028, 2014.
17. K.G.S. Venkatesan, R. Resmi, R. Remya, "Anonymizimg Geographic routing for preserving location privacy using unlinkability and unobservability", International Journal of Advanced Research in computer science & software Engg., Vol. 4, Issue 3, PP. 523 – 528, March – 2014.
18. Selvakumari. P, K.G.S. Venkatesan, "Vehicular communication using Fvmr Technique", International Journal of Applied Engineering Research, Vol. 9, Issue 22, PP. 6133 – 6139, 2014.
19. K.G.S. Venkatesan, G. Julin Leeya, G. Dayalin Leena, "Efficient colour image watermarking using factor Entrenching method", International Journal of Advanced Research in computer science & software Engg., Vol. 4, Issue 3, PP. 529 – 538, March – 2014.
20. K.G.S. Venkatesan. Kausik Mondal, Abhishek Kumar, "Enhancement of social network security by Third party application", International Journal of Advanced Research in computer science & software Engg., Vol. 3, Issue 3, PP. 230 – 237, March – 2013.
21. Annapurna Vemparala, Venkatesan.K.G., "Routing Misbehavior detection in MANET'S using an ACK based scheme", International Journal of Advanced & Innovative Research, Vol. 2, Issue 5, PP. 261 – 268, 2013.
22. K.G.S. Venkatesan. Kishore, Mukthar Hussain, "SAT : A Security Architecture in wireless mesh networks", International Journal of Advanced Research in computer science & software Engineering, Vol. 3, Issue 3, PP. 325 – 331, April – 2013.
23. Annapurna Vemparala, Venkatesan.K.G., "A Reputation based scheme for routing misbehavior detection in MANET"S ", International Journal of computer science & Management Research, Vol. 2, Issue 6, June - 2013.
24. K.G.S. Venkatesan, "Planning in FARS by dynamic multipath reconfiguration system failure recovery in wireless mesh network", International Journal of Innovative Research in computer & comm. Engineering, Vol. 2, Issue 8, August - 2014.
25. K.G.S. Venkatesan, AR. Arunachalam, S. Vijayalakshmi, V. Vinotha, "Implementation of optimized cost, Load & service monitoring for grid computing", International Journal of Innovative Research in computer & comm. Engineering, Vol. 3, Issue 2, PP. 864 – 870, February - 2015.
26. R. Karthikeyan, K.G.S. Venkatesan, M.L. Ambikha, S. Asha, "Assist Autism spectrum, Data Acquisition method using Spatio-temporal Model", International Journal of Innovative Research in computer & communication Engineering, Vol. 3, Issue 2, PP. 871 – 877, February - 2015.
27. K.G.S. Venkatesan, B. Sundar Raj, V. Keerthiga, M. Aishwarya, "Transmission of data between sensors by devolved Recognition", International Journal of Innovative Research in computer & comm. Engineering, Vol. 3, Issue 2, PP. 878 – 886, February - 2015.
28. K.G.S. Venkatesan, N.G. Vijitha, R. Karthikeyan, "Secure data transaction in Multi cloud using Two-phase validation", International Journal of Innovative Research in computer & comm. Engineering, Vol. 3, Issue 2, PP. 845 – 853, February - 2015.
29. K.G.S. Venkatesan, "Automatic Detection and control of Malware spread in decentralized peer to peer network", International Journal of Innovative Research in computer & comm. Engineering, Vol. 1, Issue 7, PP. 15157 – 15159, September - 2013.
30. Satthish Raja, S K.G.S. Venkatesan, "Electronic Mail spam zombies purify in email connection", International Journal of Advanced Research in Computer Science Engineering & Information Technology, Vol. 1, Issue 1, PP. 26 – 36, June – 2013.
31. K.G.S. Venkatesan. Dr. V. Khanna, S.B. Amarnath Reddy, "Providing Security for social Networks from Inference Attack", International

Journal of Computer Science Engineering & Scientific Technology, March – 2015.

32. K.G.S. Venkatesan, Dr. Kathir. Viswalingam, N.G. Vijitha, " Associate Adaptable Transactions Information store in the cloud using Distributed storage and meta data manager", International Journal of Innovative Research in computer & communication Engineering, Vol. 3, Issue 3, PP. 1548 – 1555, March - 2015.
33. Wi11iamGJ.Halfond,JeremyViegas,AlessandroOrs. AClassifieatio of SQL Injeetion Attacks and Countermeasures. In Proceedings of the IEEE International Symposium on Secure Software Engineering,2006.
34. K.G.S. Venkatesan, "Comparison of CDMA & GSM Mobile Technology", Middle-East Journal of Scientific Research, 13 (12),
35. PP. 1590 – 1594, 2013.
36. David Morgan. Web Injection Attacks[J]. Network Security,2006.
37. Varian Luong "Intrusion detection and prevention system: SQL injection attacks", 2010.
38. K.G.S. Venkatesan, Dr. V . Khanna,, Jay Prakahs Thakur, Banbari Kumar, " Mining user profile Exploitation cluster from computer program Logs", International Journal of Innovative Research in computer & communication Engineering, Vol. 3, Issue 3, PP. 1556 – 1561, March - 2015.
39. Ferruh Mavituna. Deep Blind SQL Injection. Portcullis security. com2008.
40. Advanced SQL injection [Online]. Available: http://www.nextgenss.com/papers/advanced_SQL_injection.pdf.
41. Atefeh Tajpour et al. "Evaluation of SQL Injection Detection and Prevention Techniques" Second International Conference on Computational Intelligence, 2010.