Testing Techniques and Analysis of SQL Injection Attacks

Arianit Maraj
Telecom of Kosovo
Prishtina, 10000, Kosovo

Ermir Rogova
FSHMN, University of Prishtina
Prishtina, 10000, Kosovo
Correspondence e-mail: ermir.rogova@uni.pr.edu

Genc Jakupi, Xheladin Grajqevci
Agency for Information Society
Prishtina, 10000, Kosovo

Abstract—**It is a well-known fact that nowadays access to sensitive information is being performed through the use of a three-tier-architecture. Web applications have become a handy interface between users and data. As database-driven web applications are being used more and more every day, web applications are being seen as a good target for attackers with the aim of accessing sensitive data. If an organization fails to deploy effective data protection systems, they might be open to various attacks. Governmental organizations, in particular, should think beyond traditional security policies in order to achieve proper data protection. It is, therefore, imperative to perform security testing and make sure that there are no holes in the system, before an attack happens.**

One of the most commonly used web application attacks is by insertion of an SQL query from the client side of the application. This attack is called SQL Injection. Since an SQL Injection vulnerability could possibly affect any website or web application that makes use of an SQL-based database, the vulnerability is one of the oldest, most prevalent and most dangerous of web application vulnerabilities. To overcome the SQL injection problems, there is a need to use different security systems. In this paper, we will use 3 different scenarios for testing security systems. Using Penetration testing technique, we will try to find out which is the best solution for protecting sensitive data within the government network of Kosovo.

Keywords-SQL injection; attack; web applications; security systems

I. INTRODUCTION

SQL (Structured Query Language) is used for relational databases by MSSQL, MySQL, Oracle, etc. These DBMSs today mostly are used on the back end of web applications. The appetites of attackers have increased for attacking the DBs that drive web applications. A successful attack on these DBs can give to attacker's access to broad range of sensitive data, such as account credentials or important business data.

SQL Injection is a technique that sends malicious messages to the DB trying to find unauthorized channels for gaining access. It is an application security weakness that enables different attackers to control a DB driven application by letting them access or delete sensitive data. SQL Injection occurs when application uses untrusted data into web application fields, as DB query. In cases when the application fails to "cleance" these data before adding to SQL query, attackers can use their own SQL commands which will be executed by DB [1-3].

The method used for implementation of this attack is "Gray Box" because the attacker has some preliminary information for the Web server (IP address, network topology, firewalls etc.), but does not have information regarding the web server's configuration. Following best security practices, security level of web applications should be very high. Knowing that most of web applications developers do not have enough knowledge for implementing security policies, definitely there is a need to use security equipment which protects web applications from SQL injection attacks. Firewalls and IPS (Intrusion Prevention System) are devices that are most frequently used to provide solid protection from such attacks [4].

In this paper, we will test security systems which are implemented in government network of Kosovo (Firewalls, IPS). In order to compare the functioning of these devices, we will test a web application attack without these protective devices in place (locally). Then, we will compare the results obtained with the results performed when the same application is protected with these security systems and the attack is performed from Internet (outside the local network).

II. RELATED WORK

So far, a lot of work has been done regarding penetration testing and SQL Injection attacks.

In [1], authors have presented the background for SQL Injection and prevention methods. They introduced different existing methods and some issues related to each method as well as identified some programmer mistakes. They evaluated each method based on inherited deployment requirement and evaluated method based on type of attack.

In [2], existing detection and prevention techniques against SQL injection attacks is presented and analyzed. For each technique, the authors discussed the strength and weaknesses of the SQLIA. They proposed a preventive

978-1-5386-2150-9/17/$31.00 ©2017 IEEE

method of the SQL queries in the web-based environment. This proposed strategy requires the alterations in the design of existing schema DB and a new guideline before writing any new database.

In [3] the authors presented one Technique for Detection and Prevention of SQL Injection Attack using ASCII Based String Matching. This technique will try to prevent the unauthorized access to DB and also will try to prevent data from being altered or deleted by users without the appropriate permissions. In this paper there are used four types of filtration techniques for detecting and preventing SQLIA: Malicious Text Detector, Constraint Validation, Query length validation and Text based Key Generator.

Our approach is different. We have used 3 different scenarios for testing security systems. Using Penetration testing technique, we try to find out which is the best solution for protecting sensitive data within the government network of Kosovo. Specifically, penetration testing has been conducted using SQL Injection attacks. As a target for the conducting such attacks, a test web server was set up in the government network.

III. Testing of Security Systems in Government Network Using Penetration Testing Technique

There is a number of testing and examination techniques that exist which can be used to evaluate the security behavior of systems. One of the methods is penetration testing, which is also known as security audit. Through this method, one can simulate an attack in order to verify the weaknesses of a system. This is an opportunity to examine the behavior of a system during an attack and identify potential vulnerabilities for this particular system. The penetration testing process includes gathering detailed information regarding the target system, identifying entry points (or back doors) and reporting the findings regarding the weaknesses of the system [5]. Pen-testers use the same tools and techniques as hackers (attackers) use for accessing sensitive data. The main difference between a pen-tester and a malicious attacker is authorization. Pen-tester is obliged to obtain written consent from the owner of the network in order to start testing. After the testing process, pen-tester should write a detailed report about vulnerabilities found in the system and also is obliged to give clear instructions to the owner of the system for prevention of attacks in the future. Tests can be performed from external and internal viewpoints. External testing is performed from outside the organization's security perimeter, from the Internet. The main goal of external testing is to reveal vulnerabilities that could be used by an external attacker [6-9]. Through external testing, testers try to find important information about the system, e.g. IP addresses, operating systems, and other important details that may help identify vulnerabilities. Internal testing is performed from the inside of organization's network and is used to demonstrate the potential damage that such an attack can cause if made from the internal network [10-11].

In this paper, we will use 3 different testing scenarios in a real network:

- Web application attack with SQL Injection – without firewalls
- Web application attack with SQL Injection –the system is protected with 2 firewalls and IPS
- Web application attack with SQL Injection –the system is protected with next generation firewalls

SQL Injection can be performed manually, but this is time consuming. Therefore we will perform such a test automatically, by the use of some software tools. We will use them in tandem by feeding results obtained from one tool to the other in order to reach the ultimate goal; getting access to the web application. Some of the tools chosen to perform tests are (these tools can only be used for testing purposes):

- Acunetix Web Vulnerability Scanner- enables web application scanning for identifying potential vulnerabilities
- Burp Suite- enables capturing of packets from the client to the web application and extracts captured data
- SQLMAP – enables penetration of the DB after identification of vulnerabilities

IV. Testing of SQL Injection Attack, from Internal Security Perimeter

To perform this test we will use one web application which has security vulnerabilities.

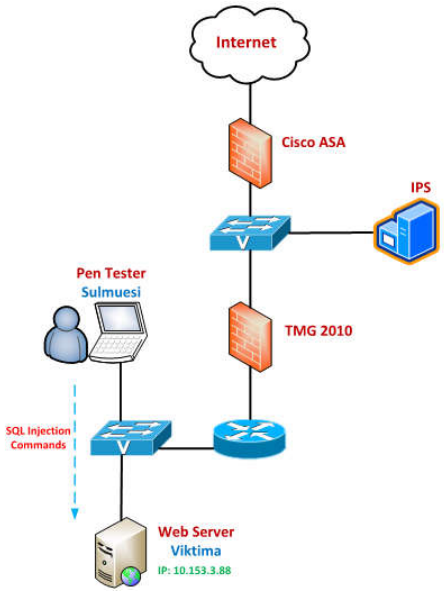

Figure 1. Network topology during SQL injection attack.

In attacker's PC we have installed Acunetix application and one virtual machine. This PC runs on Kali Linux operating system, and we also installed Burp Suite and SQLPMAP applications. Fig. 1, shows the network topology and communication flow from attacker to the victim (web server). To start the attack on the web server, initially the web application must be scanned for potential security weaknesses. These weaknesses can be used for gaining access to the system. This web application scan is performed using Acunetix application.

The scanning revealed that there are 9 high level weaknesses (most dangerous), where each vulnerability is

placed in the relevant category. Besides the number of vulnerabilities found by scanning, there are some other details identified, such as: operating system, web server type, technology used for developing web application, etc. Also here is shown the time taken to perform this scan. Through this scan we identified the links inside web application which can be used for performing SQL Injection attack.

TABLE I. WEB APPLICATION SCANNING FROM INSIDE NETWORK

Alerts found	Scan Time	Operating System	Technology for web development	Web server
High level - 9 Medium – 14 Low level – 8	5 min, 19 sec	Unix	PhP	Apache/2.4.16 (fedora) Open SSL/1.0.2d

Now, through Burp Suite software, we intercepted the traffic from web browser toward web application. This traffic was saved in one file (named as intercept.txt), which latter was used when we simulated the SQL Injection attack. Simulation of active attack was done using SQLMAP tool. First we needed to identify the DBMS (Database Management System) and other backend software in use which was done by, using the command below:

sqlmap –r /root/intercept.txt –level=5 –risk=3

Using this command, we have identified that the back end DBMS is MySQL >=5.0, web application technology is PHP 5.6.14, Apache 2.4.16 and web server operating system is Linux Fedora 23. After this, there we needed to identify the name of DB also (see below command):

sqlmap –r /root/intercept.txt –level=5 –risk=3 –dbms=mysql –current –db

After executing this command, we found the DB name "Seattle". Also, it is important to identify the tables inside DB:

sqlmap -r /root/intercept.txt --level=5 --risk=3 --dbms=mysql -D seattle–tables

Using the below command, we will see that in this DB there are 3 tables: "tblBlogs", "tblMembers" and "tblProducts". The table in which we are interested is "tblMembers", because this table stores personal data of users which have access to this web application. Thus, using below command we will try to identify columns inside this table:

sqlmap -r /root/intercept.txt --level=5 --risk=3 --dbms=mysql -D seattle -T tblMembers–columns

After executing this command, we can see that inside this table there are a lot of sensitive data, which can guarantee access to the web application, such as: Admin, Blog, Id, name, password and username (table attributes).To identify the username and password of root user, we will execute the following command on the table "tblMembers":

sqlmap -r /root/intercept.txt --level=5 --risk=3 --dbms=mysql -D seattle -T tblMembers

TABLE II. DATA IDENTIFICATION FOR WEB APPLICATION ADMINISTRATOR

Id	Name	Blog	Admin	Username	password
1	Admin	1	a	admin@seattlesounds .net	Assasin1

After this command is executed, we can identify these important data and we can have access in the web application. In Fig.2, we can see that we gained access in web application as Admin.

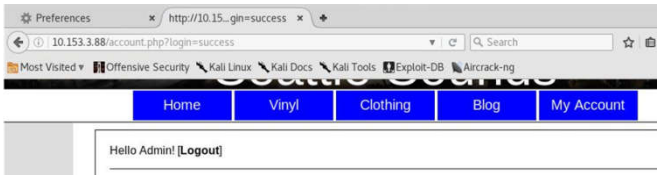

Figure 2. Access in web application.

V. TESTING OF SQL INJECTION ATTACK FROM INTERNET- ASA/TMG/IPS

In this scenario we will simulate SQL Injection attack from outside security perimeter of government network (Internet). We will test if security systems implemented in the government network can prevent such an attack.

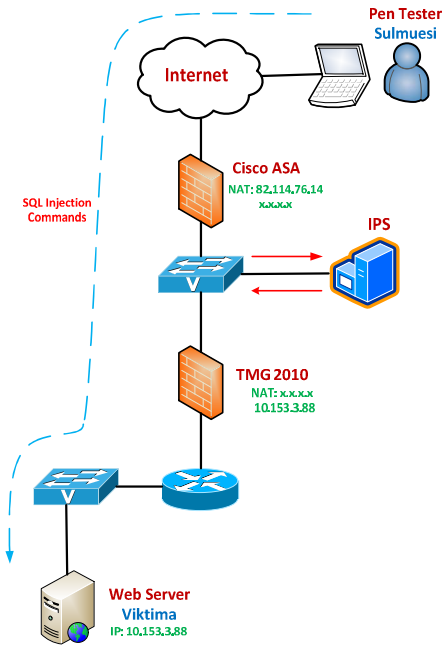

Figure 3. Network topology during SQL injection attack from Internet.

As we can see from network topology (Fig. 3), the attacker in order to get access in web application should pass 3 barriers, or security systems devices: ASA, IPS and TMG2010. ASA firewall enables traffic filtering only to the transport layer, while SQL Injection attack is done on the application level. Thus, we can conclude that ASA firewall cannot stop such an attack. IPS is security device which inspects traffic on the application layer and prevents attacks based on signatures, since malicious traffic is identified preliminary with one identification signature. How these devices will react during an SQL Injection attack, will be shown next. Like in first scenario, we will scan for weakness of web application using Acunetix. The number of high level

weaknesses is shown to be 8 in total. Also, the time for performing this scan is approximately 40 minutes, which is much higher compared to 1st scenario. One or more high-severity type vulnerabilities have been discovered by the scanner. Besides the number of vulnerabilities found, there are some other details identified, such is operating system, web server type, etc. (Table 3).

TABLE III. WEB APPLICATION SCANNING FROM INTERNET

Alerts found	Scan Time	Operating System	Technology	Web server
High level - 8 Medium – 23 Low level – 7	40 min 46 sec	Unix	PhP	Apache/2.4.16 (fedora) Open SSL/1.0.2d

Also, this scan identified the links inside web applications which can be used for performing SQL Injection attack. But the main difference is in the verification of SQL Injection weaknesses. From Internal network there are 2 verified weaknesses, whereas from Internet there are no weaknesses verified. To see which device is affected, there is a need to analyze traffic traces through security devices. After these analysis, there can be seen some traces on IPS which has blocked packets from attacker toward web application (Fig. 4). From the figure, we can see the attacker IP address, IP address of web server as well as the type of attack, which is Generic SQL Injection.

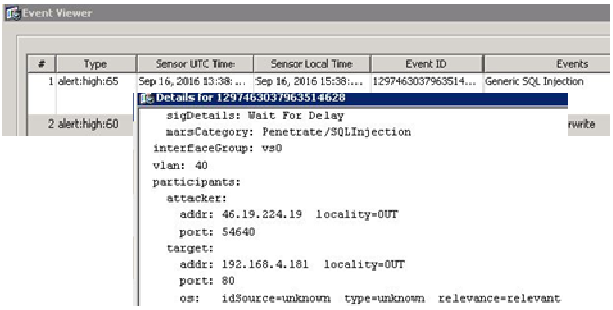

Figure 4. Traces for SQL injection attack.

In order to thoroughly test the activity of security devices, we will perform an active attack onto the web page through SQLMAP. Like in the previous scenario, we will intercept traffic from a web browser using Burp Suite. In this case, there is one critical message generated informing that this target system is protected by some kind of WAF/IPS/IDS. Therefore, this kind of attack is impossible for this scenario.

After performing this test, it became clear that TMG 2010 is not able to stop this kind of attack. Thus, in cases where IPS for any reason does not stop this attack, there is a need to use more advanced Firewalls, like Next Generation firewalls.

VI. TESTING OF SQL INJECTION ATTACK FROM INTERNET- SOPHOS UTM FIREWALL

To test the functioning and the role of Next Generation firewalls, when a web server is attacked with SQL Injection, we have used Sophos UTM Firewall. Network topology for this scenario is shown in Fig. 5. In this scenario, we will scan the web application in order to identify the weakness point. The scan revealed no weaknesses (high level) that could be used for SQL Injection attack (Fig. 5).

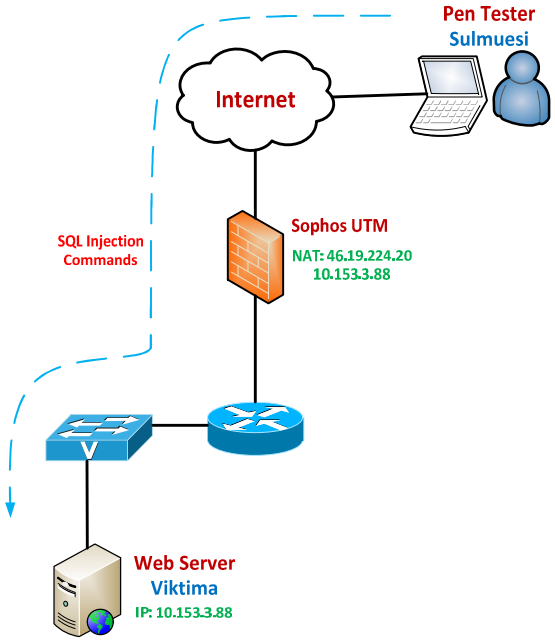

Figure 5. Network topology during SQL Injection attack – Sophos UTM.

There were identified only 5 low level alerts, which are not dangerous for SQL injection attack. The time for performing this test is only 2 minutes and 50 seconds, compared to 40 minutes which has been identified in scenario 2. Also, the scan did not identify the version of Operating system, or technology used for the web application.

TABLE IV. SCANNING RESULTS DURING ATTACK FROM INTERNET –UTM

Alerts found	Scan Time	Operating System	Technology for web development	Web server
Low level - 5	2 min, 50 sec	Unknown	Unknown	Apache

To verify if this firewall was able to protect the system from SQL injection attacks, we performed active attacks using Burp Suite and SQLMAP.

From the results, it can be seen that there is a difference with previous scenarios, especially to the usage of HASH function during client-server communication. Also, it can be noticed one critical message that appears to be not injectable. Attempts made by SQPLMAP have failed to identify the database. So, SQL failed to inject malicious code into web application through SQL Injection attack. To see if there are any traces on the firewall after this attempt, we have analyzed its logs and we have identified attacker IP address. The only change that has been observed during the attack is increasing demands to the web application (Fig.6).

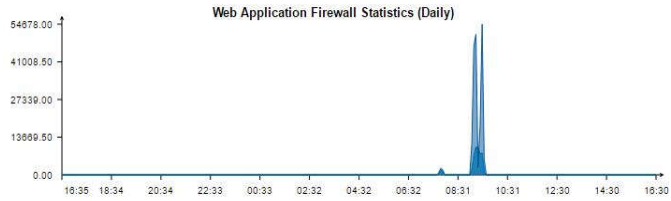

Figure 6. Demands to the web application - "Sophos UTM".

VII. CONCLUSIONS

The tests performed on government network, have made it clear that Penetration testing is of paramount importance. The advantage of using this technique is that it uses the same tools as hackers do, but in a legal way. Another advantage of this technique is the final report that is produced, which provides recommendations on avoiding security holes which can be identified on a system.

From the results, we can see that the next generation firewall offered very good protection from SQL Injection attacks. It can also be seen that the risk is much greater when web applications are attacked from the internal network or when the attack is performed from the Internet and the IPS is out of function. To increase the level of security for web applications which are hosted in Government network and to protect the network from SQL Injection attacks, there is a need to constantly monitor logs of security devices in order to identify attackers' IP addresses. Also, web applications should be configured properly, in order to verify data which go from the web application towards DB. Password fields should use high-level encryption. IPS should be updated with latest signatures. Since TMG 2010 is not able to stop an SQL Injection attack, there is a need to urgently replace them with Next Generation firewalls. All web applications hosted in government network should be placed behind next generation firewall, since government network is extended to a lot of Institutions and it is impossible to monitor and control all devices connected in this network.

VIII. FUTURE WORK

As future work, we will try to demonstrate that even when protecting systems are configured correctly, if the user is not aware of the dangers or the sources of applications, unconsciously they can allow access to the attacker. In order to realize such an attack, we will use Reverse TCP attack in combination with Social engineering Technique. Through this exercise we will see if implemented security systems (Antivirus, Firewall and IPS) can stop such attacks.

REFERENCES

[1] M. S. Aliero, A.A. Ardo, I. Ghani, M. Atiku, "Classification of SQL Injection Detection And Prevention Measure ",IOSR Journal of Engineering (IOSRJEN), ISSN (e): 2250-3021, ISSN (p): 2278-8719

[2] T. Singh Kalsi, N. Kaur et al., "Methods for preventing SQL injection attacks: a review", International Journal of Advanced Engineering Technology E-ISSN 0976-394

[3] I. Balasundarama, E. Ramaraj, "An Efficient Technique for Detection and Prevention of SQL Injection Attack using ASCII Based String Matching" Volume 30, 2012, Pages 183–190, International Conference on Communication Technology and System Design 2011

[4] A. Maraj, G.Jakupi, E.Rogova, Xh.Grajqevci, "Testing of network security systems through DoS attacks" , "Mediterranean Conference on Embedded Computing (MECO 2017), IEEE/Scopus conference, Montenegro, June 2017

[5] T. Hayajneh, BJ Mohd, A. Itradat, ANQuttoum "Performance and Information Security Evaluation with Firewalls," International Journal of Security and Its Applications, SERSC, Vol. 7, No. 6, pp 355-372, 2013. (DOI: 10.14257/ijsia.2013.7.6.36)

[6] M. Denis, C. Zena, T. Hayajneh,"Penetration Testing: Concepts, Attack Methods, and Defense Strategies",ISBN: 978-1-4673-8490-2, DOI: 10.1109/LISAT.2016.7494156

[7] C.Anley, J.Heasman, F. Lindner, and G. Richarte, "The Shellcoder's Handbook: Discovering and Exploiting Security Holes. 2007. Wiley

[8] P. David. "Your First Penetration Test". WatchGuardLiveSecurity. URL: http://www.corecom.com/external/livesecurity/pentest.html (retrieved 11January, 2017)

[9] Penetration Testing: Assessing Your Overall Security Before Attackers Do SANS Institute InfoSec, June 2006

[10] Midian, P. How to ensure effective penetration test. Information Security Technical Report, 8(4), 65- 77. [5] Basta, A., &Halton, W. (2008). Computer Security and Penetration Testing. USA: Thomson Course Technology

[11] W. Georgia, P. Van Eeckhoutte,"Penetration Testing"NoStarch Press Inc., 2014. Print.