Imperial Journal of Interdisciplinary Research (IJIR)
Vol-3, Issue-6, 2017
ISSN: 2454-1362, http://www.onlinejournal.in

SQL Injection and Web Application Security

Akshay Borase[1], Prashant Choudhari[2] & Jaypal Bhagat[3]
[1]Pune University, Information technology

***Abstract**: In the past, many popular websites have been hacked. Hackers are now active and always try to hack websites and leak data. This is why security testing of web applications is very important. Now a day I observe the one thing is that web developers only design the website they do not interested in to make their website secure. And if they want to secure their site they need extra technical staff who called penetration tester and developer need to invest more money to secure their site, due to this reason we are going to design web application security scanner.*

1. Introduction

SQL injection is the most dangerous vulnerability in the web application which can help to the attacker to extract the whole database illegally. If SQL query in not properly validated at user input then it may cause SQL injection vulnerability. The most common web application vulnerability is the failure to properly validate input coming from the client or from the environment before using it. Due to this improper validation leads to almost all of the major vulnerabilities in web applications, such as cross-site scripting, SQL injection, interpreter injection, locale/Unicode attacks, file system attacks, and buffer overflows. Sometimes data from the external entity is not trusted because it may contain evil data so that validation is the most important concept to secure the web application.

2.What is SQL Injection Attack?

A SQL injection attack consists of insertion or "injection" of a SQL query via the input data from the client to the application. A successful SQL injection exploit can read sensitive data from the database, modify database data Insert/Update/Delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system and in some cases issue commands to the operating system. SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to effect the execution of predefined SQL commands.

2. Different techniques to exploit the SQL injection vulnerable target.

2.1 Authentication bypass

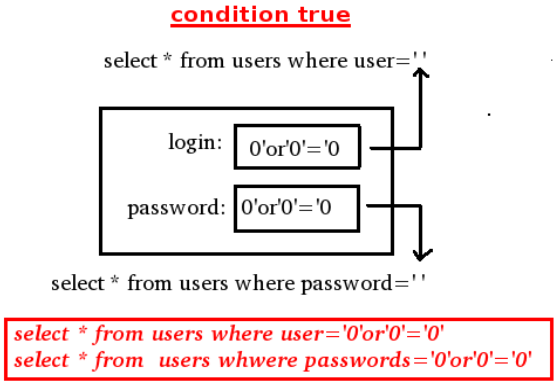


Fig1.SQL injection authentication bypass

In fig.1 show SQL injection authentication bypass attack. SQL injection can be used to bypass authentication by fooling a login page into evaluating an expression that is always true instead of checking that a login name and password is valid. So, for example, the authentication mechanism might involve an expression like: (authorize a user) WHERE Password='$password'

Using a Web interface, when prompted for his password, a malicious user might enter: 0' or '0' = '0

resulting in the query: (authorize a user) WHERE Password='0' OR '0' = '0'

The hacker has effectively injected a whole OR condition into the authentication process. Worse, the condition '0' = '0' is always true, so this SQL query will always result in the authentication process being bypassed.

2.2 Error base SQL injection attack

In the authentication bypass attack, we use 0'or'0'='0 as a payload to exploit the vulnerable target, But Error base SQL injection attack is totally different from authentication bypass because a single quote is the most popular payload to identify the target is vulnerable to error base SQL injection attack or not.

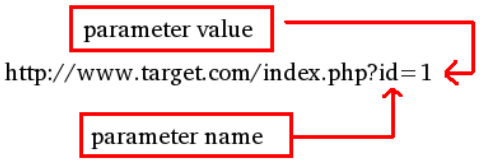


Fig.2 parameter of the URL

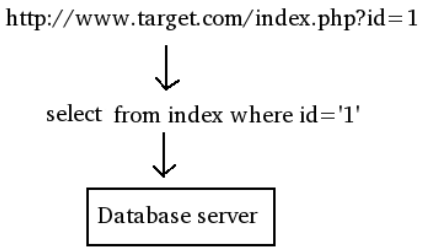


Fig.3 How its work

If we pass the single quote to the website parameter value then it reflects the database related error or little bit changes occurs in a webpage that means the target is vulnerable to SQL injection attack.

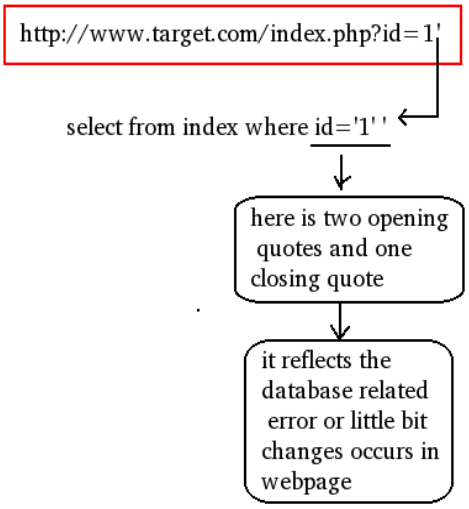


Fig.4 Why we pass single quote as a payload

3. Different another payloads to find SQL injection vulnerability.

There are a number of payloads to find the SQL injection vulnerability in target website and every payload reflects different result. Following table represents the number payloads with the result.

www.target.com.bd/buttons.php?ptid=19 '	→	It should cause error or no output on target site
www.target.com.bd/buttons.php?ptid=19 “	→	Should cause error or no output on target site
www.target.com.bd/buttons.php?ptid=19 **or 1=1**	→	Output should come but may be different output on target site
www.target.com.bd/buttons.php?ptid=19 **and 1=1**	→	Output should come but may be different output on target site
www.target.com.bd/buttons.php?ptid=19 **and false**	→	No output on target site
www.target.com.bd/buttons.php?ptid=19) **and true**	→	Same Output on target site
www.target.com.bd/buttons.php?ptid=19**' or '1'='1**	→	Any Output should come but may be different output on target site
www.target.com.bd/buttons.php?ptid=19**' and false--+**	→	No Output on target site
www.target.com.bd/buttons.php?ptid=19**' and true--+**	→	Same Output on target site

4. Vulnerability scanning with GOOGLE DORK

The concept of "Google Hacking" dates back to 2002, when Johnny Long began to collect interesting Google search queries that uncovered vulnerable systems and/or sensitive information disclosures - labeling them google Dorks. Some people call it google hacking.

Google: If you still do not know what is google, then you need to take a crash course in "how to use the internet"

Dork: Someone who has odd interests, and is often silly at times. A dork is also someone who can be themselves and not care what anyone thinks

In my opinion, A Google dork is an employee who unknowingly exposes sensitive corporate information on the Internet.

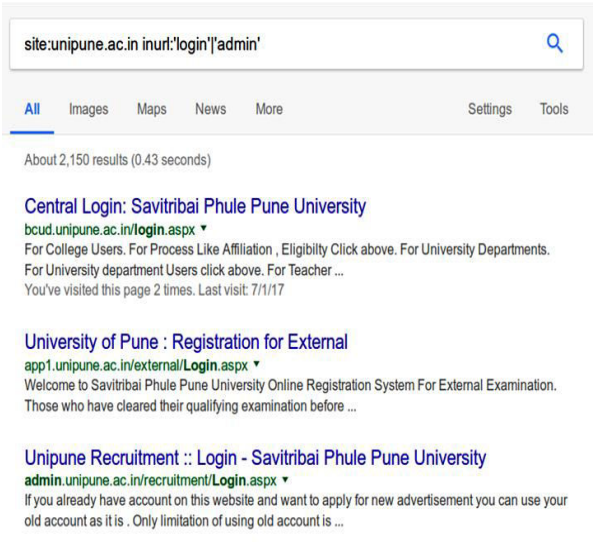
As a passive attack method, Google dorking can return usernames and passwords, email lists,

sensitive documents, personally identifiable financial information (PIFI) and website vulnerabilities.

That information can be used for any number of illegal activities, including cyber terrorism, industrial espionage, identity theft and cyber stalking

Example of google dork:-
How to find all login pages of the target website using google dork
(Target:-www.unipune.ac.in)

Site:unipune.ac.in inurl"login"|"admin"



5. Problems in existing vulnerability scanners.

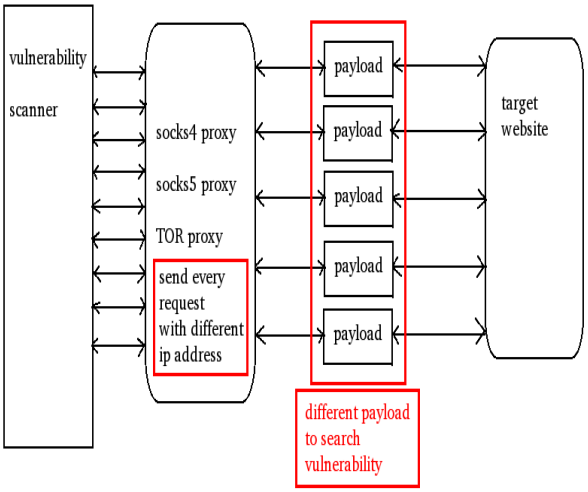
There are so many automated web application scanners (Nessus, Nikto, acunetix) are already available but the problem in that web application scanners is they can't scan subdomains automatically.

And another problem is that

Numbers of payloads are pass to the every webpage of target website from same IP address of computer system on which scanner is running due to this reason target website predict that it is DOS attack and block the IP address of scanner.

6. Propose system

After observing the different type of vulnerability scanner I decide to make open source vulnerability scanner which can use number of IP addresses to scan the target website through anonymous proxy (eg. TOR, socks4, socks5). every request send with different IP address.



Google dork is also the best way to find the vulnerability in the website. google dork is a command for google which helps to filter the data search and find the critical directory in the website as well as vulnerabilities. So that we are going to implement google dork engine also to find the vulnerabilities.

Brute force attack and google dork is the best way to find the subdomains in target website.

7. Acknowledgements

The satisfaction that accompanies the successful completion of any task would be incomplete without mentioning the people who make it possible. We are grateful to a number of individuals whose professional guidance along with encouragement has made it very pleasant endeavor to undertake this project report. We have a great pleasure in presenting the project report "SQL injection and web application security" under the guidance of Prof. S.V.Todkari We are truly grateful to project guide Prof. S.V.Todkari for her valuable guidance and encouragement. Her encouraging words went a long way in providing the patience and perseverance, which were needed to complete this project report successfully. I would like to express my gratitude to Dr.M.G.Jadhav, Principal of Jayawantrao Sawant College of Engineering and Prof.S.V.Todkari, Head of Department of Information Technology for their support and guidance. Finally we express my sincere thanks to our parents and all those who helped us directly or in directly in many ways in completion of this project report.

8. References

[1]National Vulnerability Database.
http://nvd.nist.gov/

[2] OWASP Top 10 Project.
https://www.owasp.org/index.php/Category:
OWASP Top Ten Project

[3]SQL injection
https://www.owasp.org/index.php/SQL_Injection
http://www.acunetix.com/websitesecurity/sql-injection/
https://www.exploit-db.com/papers/14340/

[4]Google Dork
http://whatis.techtarget.com/definition/Google-dork

[5] Google dork database
https://www.exploit-db.com/google-hacking-database/

[6] Tor project
https://www.torproject.org/

[7] Details about sql injection
http://www.securityidiots.com/

[8] Vulnerabilities in OSVDB Disclosed by Type by Quarter. http: //osvdb.org

[9] Jignesh Doshi, Bhushan Trivedi, "Assessment of SQL Injection Solution Approaches", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 10, October 2014.

[10] Atefeh Tajpour, Suhaimi Ibrahim, Mazdak Zamani, Mahdi Sharifi, "Effective Measures for Evaluation of SQL Injection Detection and Prevention Tools", Journal of Convergence Information Technology(JCIT) Volume8, Number14, September 2013.