International Journal of
Emerging Research in Management &Technology
ISSN: 2278-9359 (Volume-3, Issue-3)

Research Article | **March 2014**

Analysis of Security Issues in Web Applications through Penetration Testing

Khushal Singh
Assistant Professor
SCSE,Galgotias University, India

Vikas
Assistant Manager(IT)
Oriental Insurance Company Pvt Ltd., India

Abstract-

Penetration testing is method to find out the vulnerabilities and security threats in web application of the website. With the rapid growth WWW (World Wide Web), the Internet become the major source for exchange of information across the world and there is a prime need to secure our online data and the information from the malicious users. Now a days most attacks are done on the application layer, new exploits are emerging rapidly and to overcome these attacks we must think out of the box. The best way to secure our web application of the website is by conducting the penetration testing. In this research paper, penetration analysis of web security issues of the website is presented, using backtrack5R2 tool.

Keyword- vulnerability, attack, penetration testing, backtrack 5 R2, web application security, authentication

I. INTRODUCTION & BACKGROUND

As Internet age is raising day by day security has become a major facet to the Internet world. Security of the website in todays world is very important because now all the activities like communication, sharing the resources, e-governing, online banking, e-commerce, social networking, payment of utilities bills etc. done on the Internet. We can say web security is one of the crucial topic in both technology and everyday life. In the year 2011 there are so many attack on the website with the top most companies such as Sony, Citigroup, ADP(Automatic Data Processing) and others suffering from major breaches and the threats of malicious users are more in the year 2012 and it is increasing day by day. In web application there are number of vulnerabilities such as SQL injection attack, XSS(Cross Site Scripting) attack, cross site request forgery attack , input validation URL(Uniform Resource Allocator)etc. and these vulnerabilities can be find out by using the penetration testing.

Penetration testing is a technique through which we can secure our website. In order to prevent from threats we need to countermeasure the attacks exploited by the attacker and enhance the security by using the penetration testing methodology. penetration testing methodology defines a guideline with practical ideas which should be handled with great care in order to assess the system security correctly[2]. A penetration test, is a method of evaluating the security of a website with or without prior knowledge of a website under examination. The output of penetration testing commonly include a report which is divided into several sections which addresses the weaknesses found in the current state of a system with their countermeasures and recommendations[2]. Penetration testing is beneficial for several reasons as (a) discovering the more high-pitched risk vulnerabilities. (b) Providing evidence to affirm secure website. (c) discovering the impacts of successful attacks. The rest of the paper is organized as follows. The discussion of how penetration testing applied on web application of the website is given in section 2. In section 3 we present the web application security issues followed by the penetration analysis of web application security issues using Backtrack 5 R2 in section 4. Finally, conclusion will be given in section 5.

II. WEB APPLICATION SECURITY ISSUES

In this section we define the various web application security issues for vulnerability assessment. These points are very essential for web application penetration testing.

2.1 Detect Cookies with sensitive information by parameter manipulation checks in a web application: Cookies that are exchanged between the client and the server may contain sensitive information like Username, passwords and other user credentials like e-mail address, contact info, credit card no, etc. If this is the case then manipulating the cookie may allow the user to access somebody else's information. Prerequisite for this to work is that the cookie information does not go encrypted to the server. [3].

Prevention from the attack: Disallow storing of sensitive information in the cookie.

Attack Analysis: Using Achilles Web Proxy (Achilles works as a HTTP/HTTPS proxy that allows a user to intercept, log, and modify web traffic in the communication.), we can capture the cookies that are generated at the server side. Analyzing the cookies would reveal whether any sensitive information like usernames, passwords, Session ID, credit card number, e-mail ID or contact info is present in it or not.

2.2 Retrieve information from persistent cookies by parameter manipulation checks in a web application: Some websites store information in persistent cookies. These persistent cookies are stored on the user's local machine so that next time the user logs in, the information is automatically sent to the server of the website.

Prevention from the attack: Avoid storing important information in persistent cookies[3].

Attack Analysis: Start the Achilles Web Proxy , now log in and log out from the application once. After this again access the application, if some additional information (login ID, password, preferences, location etc.) which was not input by the user in the current request, is being sent to the server then these details are being sent from the persistent cookie.

2.3 By-pass form-field restrictions of the website on client: The web application might enforce client side restrictions on the form fields of the website, where user input is accepted. These restrictions may be in the form of maximum length of input, type of input (characters, numbers). All these restrictions are generally enforced using client side scripts, which can be easily bypassed by disabling/deleting them using Achilles web proxy while the form is being loaded from the server. In this we find out whether the same validations that are done on the client side are also done on the server side[4].

Prevention from the attack: Re-checking of client-side restrictions on the server of the website.

Attack Analysis: The following steps are to be carried out:

(a)Start the Achilles proxy. (b) Login to the application. (c) Locate any form field restrictions like maximum/minimum length or character set to be used on the page of the website that are enforced through client side validations. (d) Remove the client side validation scripts or modify the length restrictions set by the server of the website using the Achilles proxy. (e) Repeat the above steps for all pages of the website.

2.4 Manipulate information in hidden form fields of the website's page: Hidden Form Fields of the website's page represent a convenient way for developers to store data in the browser and are one of the most common ways of carrying data between pages. Hidden form fields are extensively used in a variety of ways and are found to be significantly vulnerable [4].

Prevention from the attack: Encrypt the hidden form fields of the website's pages . The cryptography and validation routines must be carried out on the server-side of the website. If possible ensure that no critical data is being stored in hidden form fields of the website's page.

Attack Analysis: (a) using the achilles proxy and locate any hidden form fields used in the page of the website. (b) Remove/Modify these restrictions using Achilles proxy and send the request to the server. (c) Analyze the response from the server to see any anomalies from the normal responses.

2.5 URL (Uniform Resource Locator) manipulation of the website to detect information in query string : Traditional web applications transfer data between client and server using the HTTP or HTTPS protocols. There are essentially two methods in which a server receives input from a client: (a) Data can be passed in the HTTP headers (submitted through the cookie field, or the POST request) (b) Data can be included in the query portion of the requested URL (this happens during a GET request). The data sent with a GET request is visible on the browser's URL bar. Changes can be made to the parameters present in URL. Moreover intermediate proxy servers or firewalls may record the URL in their logs. So anybody having access to the logs would be able to view the requests [5].

Prevention from the attack: (a) When client input is required from web-based forms, avoid using the GET method to submit data to the server, as the method causes the form data to be appended to the URL and can be easily manipulated. Instead, use the POST method whenever possible. (b) All sensitive data sent over the query string must be strongly encrypted.

Attack Analysis: Start the Achilles proxy and analyze the GET requests for sensitive information in query string like Username, password, credit card number, e-mail address, session token or other personal user details.

2.6 URL manipulation of the website to manipulating Query string: we try to manipulate parameters being sent through the query string. Further more if the pages are retrieved using GET requests, then the link to these pages are stored in the history of the browser and can be viewed by clicking on the link [5].

Prevention from the attack: (a) All sensitive data sent in the query string must be strongly encrypted.(b) All data should be re-validated and sanitized at the server to ensure that data is correct and has not been tampered with. (c) If it is not required then the web page of the website should not be cached on the browser (thus will not be visible in the history).

Attack Analysis: (a) Start the Achilles proxy and analyze the GET requests if any, for sensitive information in query string like Username, password, credit card number, e-mail address, session token or other personal user details. (b) Remove/Modify these parameters using Achilles proxy and send the request to the server.(c) Analyze the response from the server to see any anomalies from the normal responses or any errors that are generated revealing backend information.

2.7 Session ID violation check of a web application to strengthen the session ID: Many "stateful" web applications use Session ID's to associate a group of online actions with a specific user. This has security implications as in most of the cases the session IDs also serve as the authentication and authorization mechanisms. Any user's web session is vulnerable to hijacking and replay attack if the session ID of the user can be guessed or regenerated by an attacker. The following may be the main reasons for the Session ID being very predictable and prone to attack:

(a) Sequential allocation of Session IDs – Each visitor to the site is allocated a session ID in sequential order. Thus, by observing our own session ID information, the simple practice of replacing it with another value a few iterations up or down will allow the attacker to impersonate another user.

(b) Session ID values are too short – The full range of valid session IDs could be covered during an automated attack before there is time for the session to expire. (c) Common hashing techniques – While many commercial web services

have built in functions for calculating hashed information, these mechanisms are well known and available for reproduction. A hashing function will indeed create a session ID value that appears to be unique and great care should be taken to ensure that predicable information is not used in the generation of the hash. For example, there have been cases where the "unique" hash was based upon the local system time, and the IP address of the connecting host. Using the same hashing function, the attacker would be able to pre-calculate a large number of time dependant hashes and use them to brute force any existing session from that service. (d) Session Obfuscation – The use of a custom method of obscuring data and using it for session management. It is never a sound idea to include client or other confidential information within a session ID. For example, some organizations have even tried encoding the user's name and password within the session ID using a shifted Unicode and hexadecimal representation of the information [5,6].

Prevention from the attack: The two critical characteristics of a good Session ID generating algorithm are the randomness and length of the Session ID's that it generates. The session ID generating algorithm should generate Session ID's that are highly random and have long length.

Attack Analysis: (a) Log in and Log out from the application about 25 times. (b) Capture the session ID's generated during each login. (c) Analyze these Session ID's to find any pattern that may be evident from them.

2.8 Account control security of a web application using strengthen the password: Passwords are used for authentication and authorization on a web site. All the information related to a user that is stored in a web site has no privacy if the password is not strong enough to prevent guessing. Hence it is important for the web application to enforce strong password policies for the users[5,6].

Prevention from the attack: (a) Password should be long, at least eight characters. Where possible, allow users to use pass phrases instead of passwords. Pass phrases tend to be easier to remember and harder to steal. Make sure that each password or pass phrase has multiple types of characters. Ideally every password should have lower case letters, upper case letters, numbers, and punctuation.

(b) The user should be asked for the old password while he chooses his old password.

Attack Analysis: (a) Create a user with a very simple password, if it is allowed without any warning or error then the password policy of the site can be said to be weak. (b) Try to change the password of an existing user to a very simple one, if it goes through then it means that there is a weak password policy.

2.9 Directory traversal check of a web application: Implementations of HTTP origin servers should be careful to restrict the documents returned by HTTP requests to be only those that were intended by the server administrators. If an HTTP server translates HTTP URLs directly into file system calls, the server must take special care not to serve files that are not to be delivered to HTTP clients. For example, Unix, Microsoft Windows, and other operating systems use ".." as a path component to indicate a directory level above the current one. On such a system, an HTTP server must disallow any such construct in the Request-URL (F.g. http://traget address/../../../../winnt/repair/sam._) if it would otherwise allow access to a resource outside those intended to be accessible via the HTTP server. Similarly, files intended for reference only internally to the server (such as access control files, configuration files, and script code) must be protected from inappropriate retrieval, since they might contain sensitive information. At times, the directory structure of a web application is used to implicitly set access control rules; for instance, access to a particular module is restricted by placing that module's code in a different folder with different access rights. The web application grants access to that folder after verifying the credentials of the user. In "directory traversal attacks", intruders try to gain access to these restricted folders bypassing the credential checking mechanisms of the web application. If the application does not properly check and handle meta-characters used to describe paths then it is possible that the application is vulnerable to a "Path Traversal" attack. The attacker can construct a malicious request to return data about physical file locations such as. Traversing back to system directories, which contain binaries makes it possible to execute system command outside designated paths [7].

Prevention from the attack: (a) Strong validation of user input. If the input validation strategy is such that it only accepts the expected input then the problem is significantly reduced.(b) Where possible make use of path normalization functions, which are provided by the development language.(c) Remove offending path strings such as "../" as well as their Unicode variants from user input.(d) Preventing path traversal and path disclosure is a challenging task especially for large distributed web systems consisting of several applications.

Attack Analysis: One of the principal security functions of a web server is to restrict user requests so they can only access files within the web folders. Most servers are vulnerable to double dot "../" directory traversal exploitation if extended Unicode character representations are used in substitution for "/" and "\". This vulnerability provides a way for a malicious user to provide a special URL to the web site that will access any files whose name and location he knows, and which is located on the same logical drive as the web folders. This would potentially enable a malicious user who visited the web site to gain additional privileges on the machine specifically, it could be used to gain privileges commensurate with those of a locally logged-on user. Gaining these permissions would enable the malicious user to add, change or delete data, run code already on the server, or upload new code to the server and run it.

2.10 Error handling checks of a web application that generate errors: Passing on the ODBC error messages to the browser reveals a lot of information about the application and database to the user and hence needs to be avoided by handling errors at the application level itself. Web applications frequently generate error conditions during normal operation. Out of memory, null pointer exceptions, system call failure, database unavailable, network timeout, and hundreds of other common conditions can cause errors to be generated. These errors must be handled according to a well

thought out scheme that will provide a meaningful error message to the user, diagnostic information to the site maintainers, and no useful information to an attacker. Even when error messages don't provide a lot of detail, inconsistencies in such messages can still reveal important clues on how a site works, and what information is present under the covers. For example, when a user tries to access a file that does not exist, the error message typically indicates, "file not found". When accessing a file that the user is not authorized for, it indicates, "access denied". The user is not supposed to know the file even exists, but such inconsistencies will readily reveal the presence or absence of inaccessible files or the sites' directory structure[6,7].

Prevention from the attack: Proper handling of errors at the application level should be done so that no information revealed through errors is sent back to the browser.

Attack Analysis: Errors can be generated by giving such input to the application, which is not normally expected. This can be accomplished by filling up the input fields. For example: (a) Giving numbers in input fields where it expects text and vice-versa. (b) Changing the hidden parameters being sent to the server (c) Meddling with query string being passed on the URL (d) Sending overly long strings in input fields (e) Sending wrong cookie values.

The risk of web application checkpoint can be low, medium, and high depending upon how deep we manipulate the parameter of the web application. The risk and impact of the Web Application Security checkpoint is shown in table1.

Table I : Web Application Security checkpoint Impact & Risk

S. No.	Web Application Security checkpoint	Impact	Risk
1.	Detect Cookies with sensitive information.	Access to somebody else's information is possible.	High
2.	Retrieve information from persistent cookies.	Revealing of User credentials like username, password, e-mail id, credit card number, etc.	High
3.	By-pass Form-field restrictions of the website on client.	Enumeration of information, unauthenticated access.	High
4.	Manipulate information in Hidden Form fields of the website's page.	Cross account access, bypassing restrictions, generating errors from the server which may help in enumeration of the database.	High
5.	URL Manipulation of the website to detect information in query string.	Launch of parameter manipulation attacks, cross account access.	Medium
6.	URL Manipulation of the website to manipulating Query string.	Unauthorized access to information, generation of errors that can help in enumeration of the database.	High
7.	Session ID Violation Check of a web application to strengthen the session ID.	Hijacking an active session, Unauthorized access to user accounts.	Medium
8.	Account Control Security of a web application using strengthen the password.	Guessing of passwords leading to access to private information.	Medium
9.	Directory traversal check of a web application.	Access to restricted modules and administrative consoles. Access to sample programs, some of which may be insecure and gaining additional privileges on the machine.	Medium
10.	Error Handling Checks of a web application that Generate errors.	Enumeration of backend information, enumeration of application components.	Medium

III. PENETRATION ANALYSIS OF WEB APPLICATION SECURITY ISSUES USING BACKTRACK

BackTrack is an operating system based on the Ubuntu GNU/Linux distribution aimed at digital forensics and penetration testing use [8]. BackTrack is a versatile operating system that comes with number of security assessment and penetration testing tools [8].

The illustration for the BackTrack testing process is given below [9].

Target Scoping: Target Scoping define what has to be tested, how it should be tested and what are the conditions should be applied for the test process.[8]

Information Gathering: In this phase we collect as much information about the target as we can. [8]

Target Discovery: In this phase we find out how the devices in the target domain are interconnected . [8]

Enumerating Target: In this phase we find out the open ports by using port scanning techniques such as full-open, half-open, and stealth, scan can. [8]

Vulnerability Mapping: In this phase we find out vulnerabilities based on the disclosed ports in above phase. [8]

Social Engineering: Social engineering is a term that depicts a non-technical sort of intrusion that based on human interaction and often involves tricking other people to break normal security procedures. [8]
Target Exploitation: In this phase we exploit the target system based on the vulnerabilities fond in above phases. [8]
Privilege Escalation: In this technique the attacker access the system beyond limitation of the normal web application user. [8]
Documentation and Reporting
In this phase report is made on the basis of vulnerabilities found. [8]
If the security issues which are explained in section 3 are not treated well by the security professionals, then the website is not secure. We show with the help of backtrack tool that how the attacker can reveal the sensitive information from the website if the security issues define in section 3 are not overcome. We hide the website name in the screen shot to avoid defacing the website.
As shown in the figure 1, the input to the tool is as follows:

$$root@bt{:}/pentest/database/sqlmap\#\ ./sqlmap.py\ \ -h$$

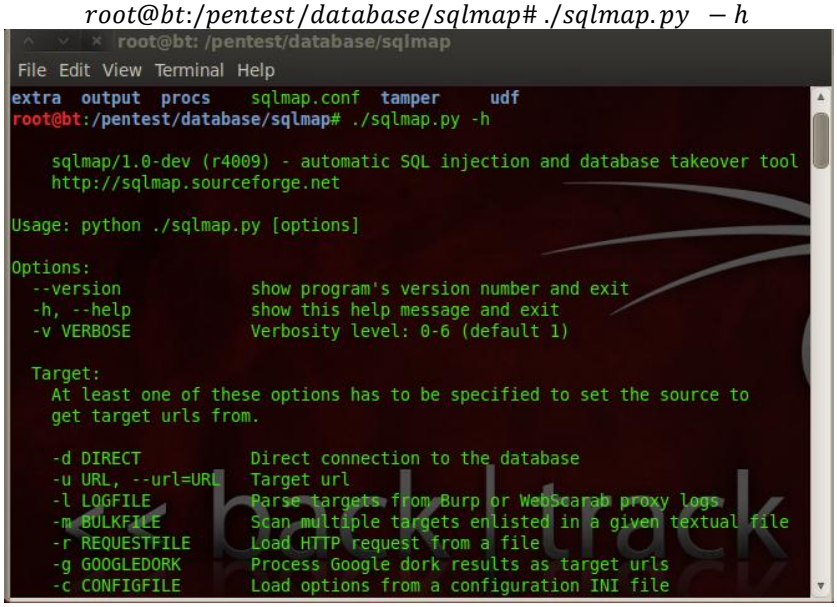

Fig.1:Options in Sqlmap Tool

Figure2 shows how the database of the targeted website is extracted using the following input:

$$root@bt{:}/pentest/database/sqlmap\#\ ./sqlmap.py\ -u\ http{:}//www.abc.com/viewfaculty.php?id = 12\ --current\ -db$$

The database name is extracted because the website is vulnerable, using database finger-printing the attacker discovers the type and version of database that a web application is using. Certain types of databases respond differently to different queries and attacks, and this information can be used to extracting the database as when "HAVING" clause is used it filter the records that "GROUP BY" [12] clause returns as shown in the figure 2.

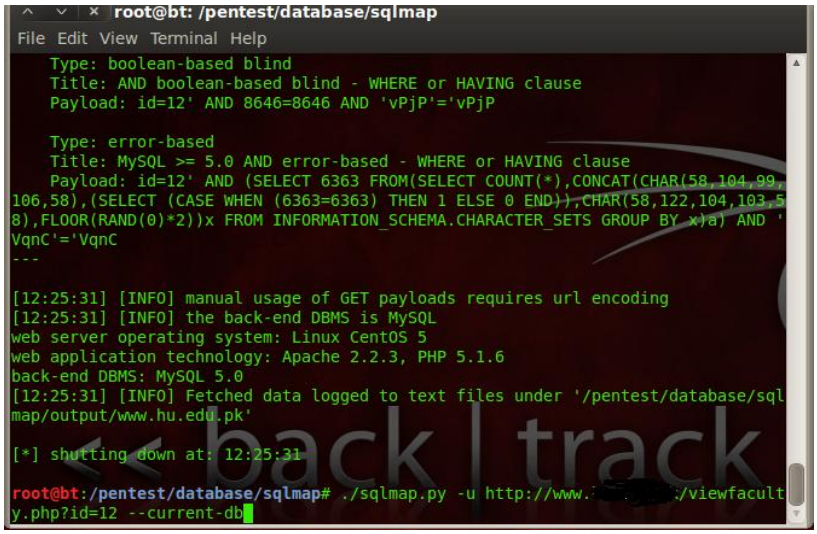

Fig.2: Input to the sqlmap tool to reveal the database of the targeted website

In figure 3, the database name 'c3results' is revealed from the targeted website. Knowing the type and version of the database used by the website allows an attacker to obtain database specific attacks. By knowing the database schema the attacker can correctly extract data from a database

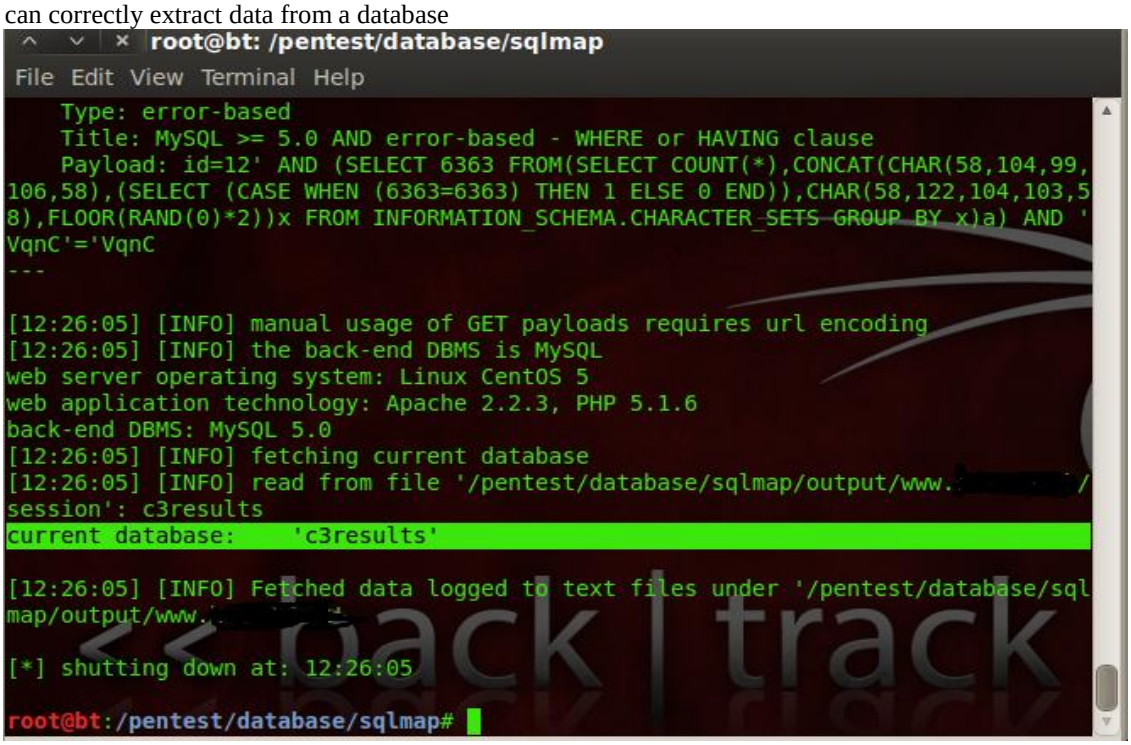

Fig.3:Revealing the Databse name of the targeted website

Now, there will be a need to extract the tables from the database,an attacker gathers important information about the type and structure of the back-end database of a Web application. The default error page returned by application servers will reveal vulnerable/injectable to an attacker. After extracting the various table names from the database of the website, we choose the 'admin' table to extract the columns of the table 'admin' Figure 4 shows the columns of the table admin. The attackers gain information about the schema of the backend database.

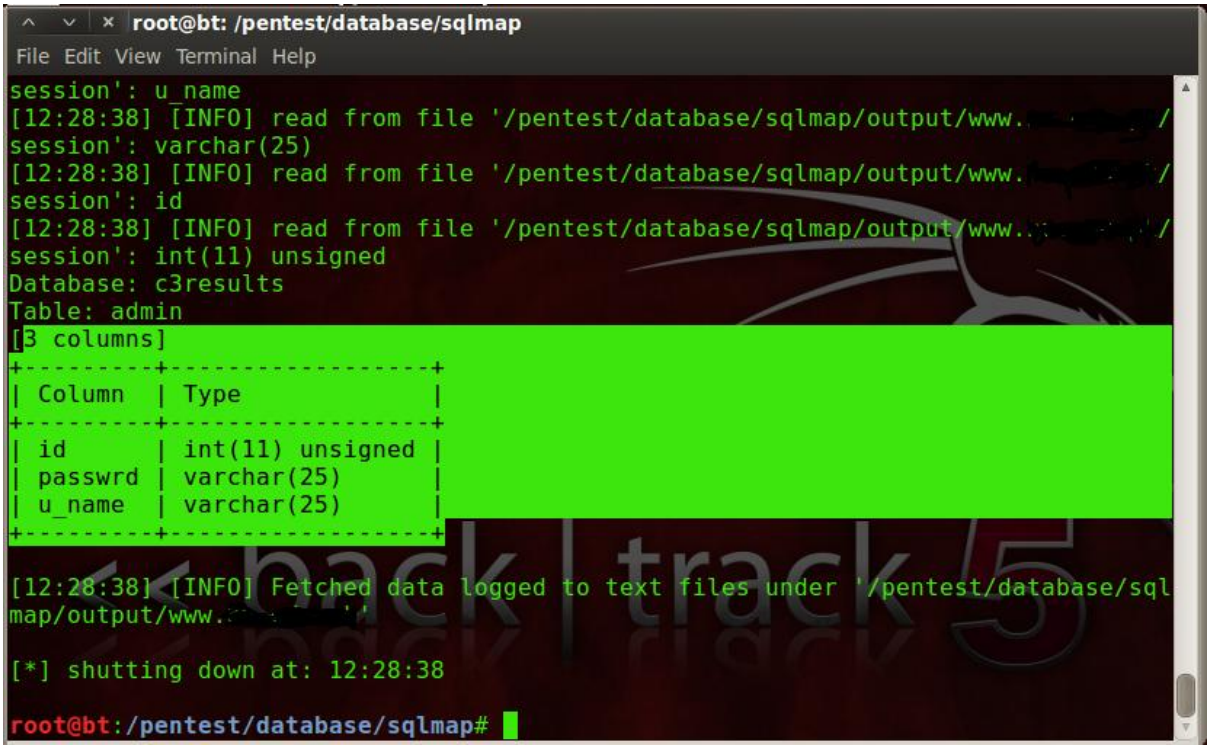

Fig.4: Column from the table Admin

In figure 5 the values of the columns of the admin table are shown. These are the desired results which we want. As we can see the website does not follow the strong password policy as defined in section 2.8. The Password should be combination of uppercase and lowercase letters.

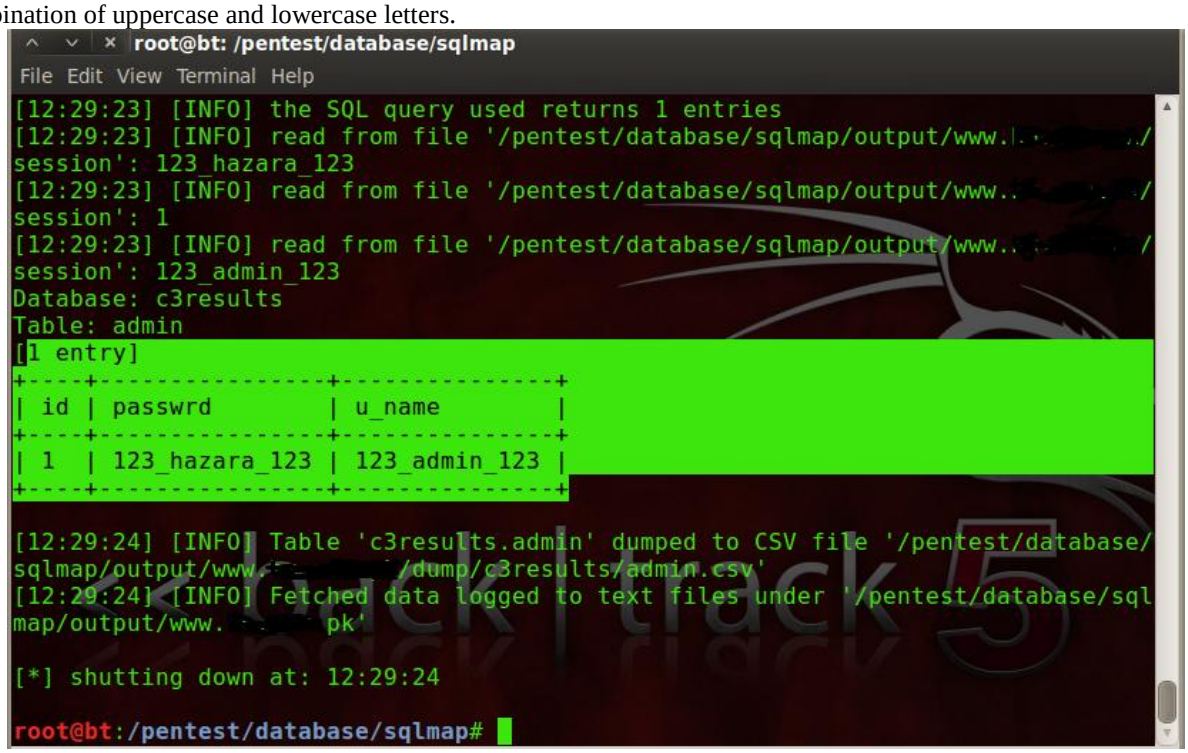

Fig.5: Revealing the Password and Username of Admin

IV. CONCLUSION

In backtrack there are so many utilities and one utility might be better to perform a particular kind of job than the other utility. As a penetration tester or ethical hacker we must know which utility is good to perform the task better. We found that we cannot discover every type of vulnerability by using the single technique, so the pentester must think out of the box and do the penetration testing that will be beneficial for the client and if the cost of the penetration testing is more than the profit of the organization than there is no mean to conduct the penetration testing in such a case the pentester must give the suitable alternate advice to the client.

REFERENCES

1. D. Geer, J. Harthorne, :Penetration testing: A Duet.: Proceedings of the 18th Annual Computer Security Applications Conference.: ACSAC (2002)
2. G.J.William, Halfond, S.R.Choudhary and a.Orso. : Penetration Testing with Improved Input Vector Identification.International Conference on Software Testing Verification and Validation(2009)
3. J. Wack, M. Tracy, and M. Souppaya. : Guideline on network security testing.: NIST SP800-42 (2003)
4. Penetration Testing. : Proceedings of the Application Security Conference (2008)
5. Igor Kotenko and Mihail Stepashkin. : Analyzing vulnerabilities and measuring security level at design and exploitation stages of computer network life cycle.Springer-Verlag Berlin Heidelberg (2005)
6. Meng, Wu, Yang and Yu.: Research of an E-mail forensic and analysis system based on visualization.: Asia-Pacific Conference on Computational intelligence and Industrial Application (2009)
7. S. Elbaum, G. Rothermel, S. Karre, and M. F. : User-Session Data to Support Web Application Testing. IEEE Transactions On Software Engineering, pp. 18--202. (2005)
8. penetration testing framework, http://www.backtrack-linux.org/Shakeel ali and Tedi Heriyanto.:Master the art of penetration testing with backtrack .52--5 (2011)
9. Ethical Hacking, http://www. .com/2012/03/dnsenum-on-backtrack-5.html#!/2012/03/dnsenum-on-backtrack-5.html
10. Ethical Hacking, http://www.ehacking.net/search?q=zenmap
11. Ethical Hacking, http://www.ehacking.net/2011/06/how-to-use-armitage-in-backtrack-5.html
12. Stephen McLaughlin, Dmitry Podkuiko, Sergei Miadzvezhanka, Adam Delozier, and Patrick McDaniel.:Multi-vendor Penetration Testing in the Advanced Metering Infrastructure, Department of Computer Science and Engineering,2006.