|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Contents** | **Page No.** |
| **1** | **Annexure I– Micro Project Proposal** | **1-2** |
| 1.Aims/Benefits of the Micro-Project | 1 |
| 2. Course Outcome Addressed | 1 |
| 3.Proposed Methodology | 1 |
| 4. Action Plan | 2 |
| 5. Name of Team Members with Roll No.’s | 2 |
| **2** | **Annexure II – Micro Project Report** | **3-18** |
| 1.Rationale | 3 |
| 2.Aims/Benefits of the Micro-Project | 3 |
| 3.Course Outcome Achieved | 3 |
| 4. Literature Review | 3 |
| 5.Actual Methodology Followed | 4-12 |
| 5.1 | 13 |
| 5.2 | 14 |
| 6.Actual Resources Used | 15 |
| 7.Outputs of Micro-Projects | 15 |
| 8. Skill developed / Learning out of this Micro-Project | 16 |
| 9. Applications of this Micro-Project | 16 |

**Annexure I**

**Micro Project Proposal**

**“SQL Injection”**

**1. Aims/Benefits of the Microproject:**

**I. Aims: -**

* **Understanding the Concept**: Understand how SQL injection vulnerabilities occur and how to exploit them.
* **Enhance understanding of SQL syntax and database interactions**: Gain insights into the mechanics of SQL queries and their impact on database systems.
* **Demonstrate the importance of SQL injection prevention**: Highlight the critical need for organizations to secure their databases against these attacks.

**II. Benefits: -**

* **Increased security awareness**: Raises awareness of the risks associated with SQL injection vulnerabilities and the importance of taking proactive measures.
* **Personal Growth**: Through participation in the project, individuals may experience personal growth, increased confidence, and a sense of achievement as they contribute to building understanding and solutions in this emerging field.
* **Enhanced problem-solving abilities**: Develops logical thinking and analytical skills through the process of identifying and exploiting SQL injection vulnerabilities.
* **Practical application of theoretical concepts**: Applies theoretical knowledge about SQL injection to real-world scenarios, fostering a deeper understanding of the subject matter.
* **Career advancement**: Demonstrates proficiency in a highly sought-after security skill, opening up potential career opportunities in cybersecurity and related fields.

**2. Course Outcome Addressed:**

a. Identify your entrepreneurial traits.

b. Identify the business opportunities that suits you.

c. Use the support systems to zero down to your business idea.

d. Develop comprehensive business plans. e. Prepare plans to manage the enterprise effectively.

**3. Proposed Methodology:**

Proposed Methodology for SQL Injection Microproject: -   
1. **Understanding SQL Injection**: Begin by researching and understanding what SQL injection is and how it works. This involves learning about the basics of SQL (Structured Query Language) and how databases interact with web applications. Understand the potential risks associated with SQL injection, such as unauthorized access to data, data manipulation, and even database corruption.  
  
2. **Identifying Vulnerable Points**: Next, identify potential vulnerable points within your web application where SQL injection could occur. These vulnerable points often exist in user input fields such as login forms, search bars, or any other input field that interacts with a database. Analyze the code to see how user inputs are handled and whether they are properly sanitized or validated before being used in SQL queries.  
  
3. **Creating Test Cases**: Develop test cases to simulate SQL injection attacks against the identified vulnerable points. Craft malicious inputs that exploit SQL injection vulnerabilities to retrieve sensitive data or manipulate database queries. These test cases will help you understand the extent of the vulnerability and its potential impact on your application.  
  
4. **Conducting Penetration Testing**: Use penetration testing tools and techniques to systematically test your web application for SQL injection vulnerabilities. Automated tools like SQLMap can be used to scan for vulnerabilities and attempt to exploit them. Additionally, manual testing by experienced security professionals can uncover more nuanced vulnerabilities that automated tools might miss.

5. **Analyzing Results**: Analyze the results of your tests to determine the severity of the SQL injection vulnerabilities. Identify any sensitive data that could be accessed or manipulated through the vulnerabilities. Consider the potential impact on the confidentiality, integrity, and availability of your application and its data.  
  
6. **Mitigation Strategies**: Finally, develop and implement mitigation strategies to address the identified SQL injection vulnerabilities. This may involve a combination of measures such as input validation, parameterized queries, escaping user inputs, and using prepared statements or ORM frameworks. Ensure that all developers working on the project are trained in secure coding practices to prevent SQL injection vulnerabilities in the future.

**4. Action Plan**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.  No. | Details of Activity | Planned  Start date | Planned  Finish date | Name of Responsible Team Members |
| 1 | Search the topic | 29/08/2024  1:00pm-3:00pm | 05/09/2024  1:00pm-3:00pm |  |
| 2 | Search the information | 12/09/2022  1:00pm-3:00pm | 19/09/2022  1:00pm-3:00pm |  |
| 3 | Material Gathered | 26/09/2022  1:00pm-3:00pm | 03/10/2022  1:00pm-3:00pm |  |
| 4 | Logic developing | 10/05/2022  1:00pm-3:00pm | 15/10/2022  1:00pm-3:00pm | Harsh Moreshwar Kale |
| 5 | Circuit making | 31/10/2022  1:00pm-3:00pm | 07/11/2022  1:00pm-3:00pm |  |
| 6 | Implementing the Circuit | 14/11/2022  1:00pm-3:00pm | 21/11/2022  1:00pm-3:00pm |  |
| 7 | Testing and Correction | 28/11/2022  1:00pm-3:00pm | 05/12/2022  1:00pm-3:00pm |  |
| 8 | Finalizing Project with its report | 12/12/2022  1:00pm-3:00pm | 19/12/2022  1:00pm-3:00pm |  |

**5.Names of Team Members with Roll No.’s:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr.**  **No.** | **Enrollment No.** | **Name of Team Member** | **Roll No.** |
| 2 | 2110950051 | Mr. Harsh Moreshwar Kale | 03 |

**Ms. Kachare S. M**

**Name and Signature of the Teacher**

**Annexure – II**

**Micro-Project Report**

**“SQL Injection”**

1. **Rationale:**

SQL injection is a web security vulnerability that allows attackers to execute arbitrary SQL queries on a database server. This can be used to access sensitive data, modify data, or even delete data. SQL injection exists because many web applications do not properly validate user input, which allows attackers to inject malicious SQL code into the application. This can have a number of adverse effects, including data breaches, financial loss, and reputational damage.

**2. Aims/Benefits of the Micro-Project:**

**I. Aims: -**

* **Understanding the Concept**: Understand how SQL injection vulnerabilities occur and how to exploit them.
* **Enhance understanding of SQL syntax and database interactions**: Gain insights into the mechanics of SQL queries and their impact on database systems.
* **Demonstrate the importance of SQL injection prevention**: Highlight the critical need for organizations to secure their databases against these attacks.

**II. Benefits: -**

* **Increased security awareness**: Raises awareness of the risks associated with SQL injection vulnerabilities and the importance of taking proactive measures.
* **Personal Growth**: Through participation in the project, individuals may experience personal growth, increased confidence, and a sense of achievement as they contribute to building understanding and solutions in this emerging field.
* **Enhanced problem-solving abilities**: Develops logical thinking and analytical skills through the process of identifying and exploiting SQL injection vulnerabilities.
* **Practical application of theoretical concepts**: Applies theoretical knowledge about SQL injection to real-world scenarios, fostering a deeper understanding of the subject matter.
* **Career advancement**: Demonstrates proficiency in a highly sought-after security skill, opening up potential career opportunities in cybersecurity and related fields.

**3. Course Outcomes Achieved:**

a) Identifying the Vulnerability of a website against the SQL Injection

b) Use the tools for perform SQL Injection on a website ethically

**4. Literature Review:**

SQL Injection: A Comprehensive Literature Review  
  
**Introduction**  
SQL injection is a prevalent web application security vulnerability that exploits vulnerabilities in SQL database queries. It allows attackers to bypass authentication, modify data, or gain unauthorized access to sensitive information. This literature review provides an overview of SQL injection, its different types, detection techniques, and prevention mechanisms.  
  
**Types of SQL Injection**  
 1) Union-based injection: Injects malicious code into a SQL query using the UNION operator to combine two or more queries.  
 2) Blind injection: Uses error messages or time-based responses to infer information about the underlying database structure without direct feedback.  
 3) Piggybacking injection: Inserts malicious code into a legitimate SQL query issued by another user.  
 4) Out-of-band injection: Exfiltrates data or executes commands through channels other than the database server, such as DNS lookups or HTTP requests.  
  
**Detection Techniques**  
  
 Static code analysis: Examines source code for potential vulnerabilities, such as unvalidated input or hardcoded SQL queries.  
 Dynamic analysis: Instruments application code to detect SQL injection attempts during runtime, such as by monitoring query parameters or statement execution.  
 Network traffic analysis: Inspects network traffic to identify anomalous database queries or suspicious patterns.  
  
**Prevention Mechanisms**  
  
 Input validation and filtering: Enforce strict input validation rules to reject malicious characters or SQL keywords.  
 Prepared statements: Use parameterized SQL queries to prevent SQL injection by separating data from code.  
 Stored procedures: Encapsulate complex SQL queries within stored procedures to control access and prevent injection vulnerabilities.  
 Web application firewalls (WAFs): Filter and block malicious traffic targeting SQL databases.  
  
**Recent Developments**  
  
 Machine learning (ML) for detection: ML algorithms are being explored for automating SQL injection detection and improving accuracy.  
 Code obfuscation: Obfuscating SQL code can make it more difficult for attackers to identify injection points.  
 Data encryption: Encrypting sensitive data stored in the database can mitigate the impact of SQL injection attacks.  
  
SQL injection remains a significant threat to web applications. Understanding different types of injections, detection techniques, and prevention mechanisms is crucial for protecting against this vulnerability. Ongoing research and advancements continue to improve the detection and mitigation of SQL injection attacks.

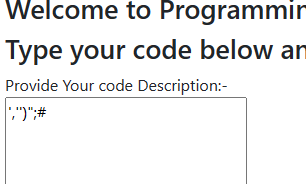
**5. Actual Methodology Followed:**

5.1 Introduction to SQL Injection  
  
In simple terms, SQL injection is like a sneaky trick that hackers use to get into a website's secret vault of information. Imagine a website is like a big library with lots of books (which are actually stored in a database). Normally, when you ask the librarian (the website) for a specific book (information), they'll give it to you safely. But with SQL injection, it's like someone slips a secret code into the librarian's instructions, so instead of giving you the book you asked for, they might accidentally give you access to all the books in the library! This happens because the hacker tricks the website into running their own commands on the library's database, giving them unauthorized access to sensitive information like usernames, passwords, or even the ability to delete or change things in the database. So, it's like a sneaky way for bad guys to break into a website and mess things up.  
  
5.2 What is SQL Injection?   
  
SQL Injection is a security vulnerability where hackers can sneak harmful commands into a database by tricking the application into running unauthorized SQL queries. This happens when the application doesn't properly check or clean the data it receives from users before using it in SQL commands. As a result, attackers can exploit this weakness to access, modify, or delete sensitive data in the database.

5.3 Is the SQL Injection is Ethical or not?   
  
SQL injection is unethical and often illegal because it involves exploiting vulnerabilities in a system's security to gain unauthorized access to data or manipulate it. This compromises the integrity and privacy of the data stored in databases. Additionally, unauthorized access to sensitive information can lead to various legal consequences, including fines and penalties, making SQL injection both unethical and potentially punishable by law.  
  
5.4 History of SQL Injection  
  
SQL injection, as a method of attacking web applications, was first brought to public attention by Chris Anley in 1998. Anley, a computer security researcher, discovered and reported this vulnerability as a means of exploiting websites that interacted with SQL databases. Before Anley's disclosure, many developers were unaware of the potential dangers of insecurely handling user input in SQL queries. However, his findings shed light on a significant security flaw that existed in countless web applications at the time. Following Anley's report, SQL injection quickly gained notoriety within the cybersecurity community. Hackers and malicious actors began to exploit this vulnerability to gain unauthorized access to sensitive data stored within databases. By injecting malicious SQL code into input fields, attackers could manipulate database queries to execute arbitrary commands or extract confidential information.

The evolution of SQL injection techniques has mirrored advancements in web technology and cybersecurity. As developers implement new frameworks and methodologies, hackers continually devise innovative ways to exploit vulnerabilities and bypass security defenses.

Today, SQL injection stands as one of the most common and dangerous web vulnerabilities, posing a significant risk to the confidentiality, integrity, and availability of sensitive data stored within databases. Its enduring prevalence underscores the importance of ongoing education, vigilance, and robust security measures to mitigate the risks posed by this persistent threat.  
  
5.5 How Does SQL Injection Work?   
  
SQL injection works by exploiting a web application's input validation mechanisms. An attacker can craft a specially designed input that contains malicious SQL code, which is then passed to the database server. The database server executes the malicious code, resulting in the unauthorized execution of SQL queries.







5.6 Dark Side of SQL Injection  
  
Data Theft or Destruction: SQL injection attacks can allow attackers to steal sensitive data stored in databases. By exploiting vulnerabilities in web applications, hackers can execute malicious SQL queries to extract confidential information, such as usernames, passwords, credit card numbers, and personal details. In more severe cases, attackers may even tamper with or delete critical data, leading to significant financial or reputational damage.

Unauthorized Access to Sensitive Information: SQL injection vulnerabilities can grant unauthorized access to sensitive areas of a web application or database. Attackers can manipulate SQL queries to bypass authentication mechanisms and gain elevated privileges, enabling them to view or modify confidential data beyond their authorized scope.

Denial of Service Attacks: SQL injection vulnerabilities can also be leveraged to launch denial of service (DoS) attacks against web applications and underlying databases. Attackers can flood the application with malicious SQL queries, causing excessive resource consumption, database overload, or system crashes. These DoS attacks can disrupt the availability of the web application, rendering it inaccessible to legitimate users and causing financial losses or reputational damage to the organization.

**6. Actual Resources Used:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr.**  **No.** | **Name of resource / material** | **Specification** | **Quantity** | **Remarks** |
| 1 | Computer | Windows 11, RAM 16 GB | 1 |  |
| 2 | Operating System | Windows 11 | 1 |  |
| 3 | Browser | Microsoft Edge | 1 |  |

**7. Skill developed / Learning out of this Micro-Project:**

1. Understanding Web Security: Through researching and simulating SQL injection attacks, participants gain a deeper understanding of web security principles and vulnerabilities. They learn how attackers exploit weaknesses in web applications to compromise data integrity, confidentiality, and availability.

2. Practical Application of Cybersecurity Concepts: Participants apply theoretical knowledge of cybersecurity concepts to real-world scenarios by identifying, testing, and mitigating SQL injection vulnerabilities. This hands-on experience enhances their practical skills in detecting and addressing security threats.

3. Critical Thinking and Problem-Solving: Engaging in penetration testing and vulnerability assessment exercises requires critical thinking and problem-solving skills. Participants analyze web application code, assess potential security risks, and devise effective mitigation strategies to protect against SQL injection attacks.

4. Programming and Scripting Skills: Participants may enhance their programming and scripting skills by writing custom code to simulate SQL injection attacks or develop automated testing scripts. This experience strengthens their ability to manipulate and interact with databases programmatically.

5. Communication and Documentation: Documenting findings, test methodologies, and mitigation strategies fosters effective communication skills. Participants learn to articulate technical concepts, present their findings clearly, and collaborate with team members to address security vulnerabilities.

6. Risk Management: Participants gain insights into risk management practices within the context of cybersecurity. They assess the likelihood and potential impact of SQL injection vulnerabilities on web applications and develop strategies to mitigate these risks effectively.

**8. Applications of this Micro-Project:**

* Enhancing the security of e-commerce websites to safeguard customer payment information.
* Securing healthcare management systems to protect sensitive patient records from unauthorized access.
* Strengthening online banking platforms to prevent fraudulent activities and ensure data confidentiality.
* Safeguarding government portals to protect citizen information and maintain data integrity.
* Securing educational institution databases to prevent tampering with student records and grades.
* Protecting social media platforms to prevent unauthorized access to user accounts and personal information.
* Securing supply chain management systems to safeguard sensitive business data and trade secrets.

##########