**Penetration Testing Report**

**Contributors Name: Dharanidharan. V & Karthikaa. R  
Program: HCPT**

**Date: 10.03.23**

**Introduction**

This report document hereby describes the proceedings and results of a Black Box security assessment conducted against the **Week 3 Labs**. The report hereby lists the findings and corresponding best practice mitigation actions and recommendations.

**1. Objective**

The objective of the assessment was to uncover vulnerabilities in the **Week 3 Labs** and provide a final security assessment report comprising vulnerabilities, remediation strategy and recommendation guidelines to help mitigate the identified vulnerabilities and risks during the activity.

**2. Scope**

This section defines the scope and boundaries of the project.

|  |  |
| --- | --- |
| **Application Name** | **Cross-Site Request Forgery, Server-Side Request Forgery** |

**3. Summary**

Outlined is a Black Box Application Security assessment for the **Week 3 Labs**.

**Total number of Sub-labs: {count} Sub-labs**

|  |  |  |
| --- | --- | --- |
| **High** | **Medium** | **Low** |
| **4** | **5** | **6** |

**High - Number of Sub-labs with hard difficulty level**

**Medium - Number of Sub-labs with Medium difficulty level**

**Low - Number of Sub-labs with Easy difficulty level**

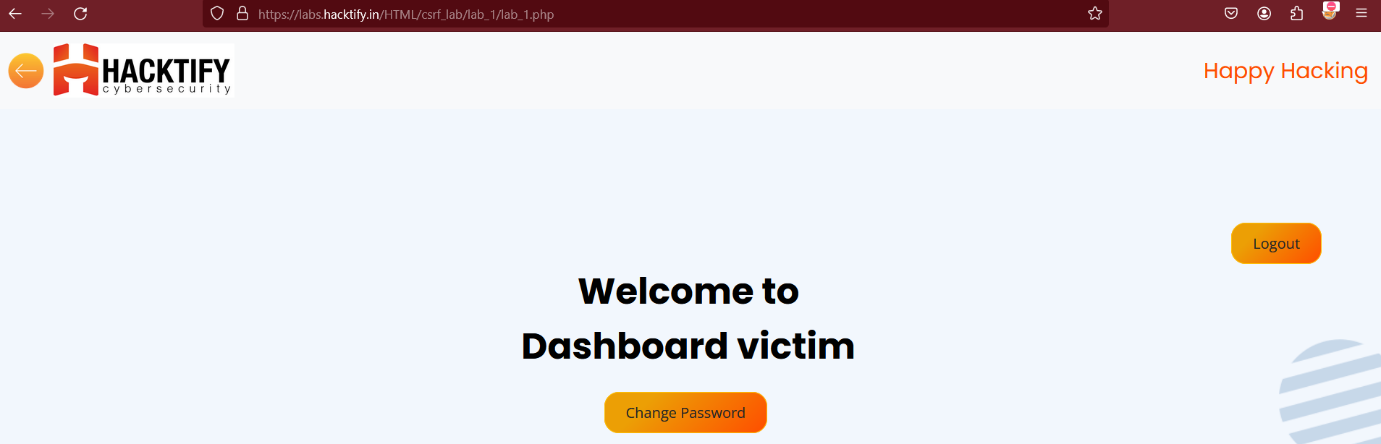
# 1. Cross-Site Request Forgery

# 1.1. Eassyy CSRF

|  |  |
| --- | --- |
| **Reference** | **Risk Rating** |
| Eassyy CSRF | **Low** |
| **Tools Used** | |
| Burp Suite and Hacktify Poc Generator. | |
| **Vulnerability Description** | |
| In the user account management functionality of a website, a CSRF (Cross-Site Request Forgery) vulnerability exists, allowing attackers to perform unauthorized actions on behalf of authenticated users. | |
| **How It Was Discovered** | |
| Manual Testing and analysis | |
| **Vulnerable URLs** | |
| https://labs.hacktify.in/HTML/csrf\_lab/lab\_1/index.php | |
| **Consequences of not Fixing the Issue** | |
| Unauthorized actions performed on behalf of authenticated users, such as changing passwords, modifying account settings, or initiating financial transactions.  Compromise of user accounts, leading to potential data loss, identity theft, or unauthorized access to sensitive information. | |
| **Suggested Countermeasures** | |
| Include anti-CSRF tokens or synchronizer tokens in forms and requests to validate the origin of requests and prevent CSRF attacks.  Implement strict validation of requests to ensure that sensitive actions, such as password changes, require explicit user consent and authentication. | |
| **References** | |
| <https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html> | |

# 

# Proof of Concept



# 

# 1.2. Always Validate Tokens

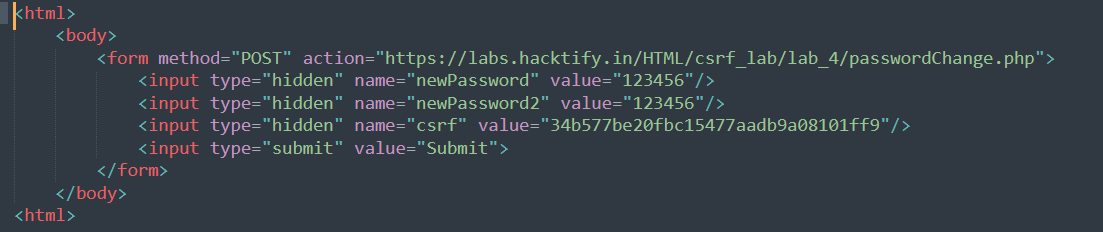
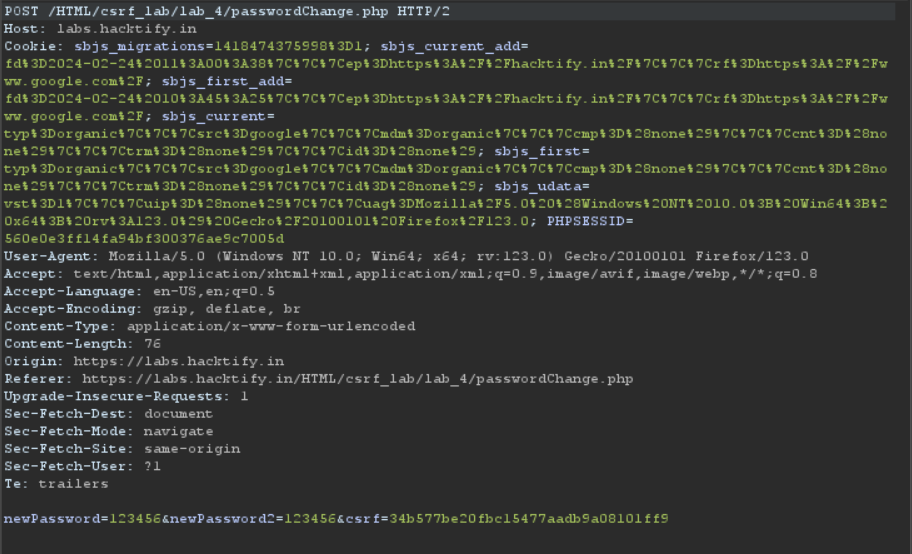
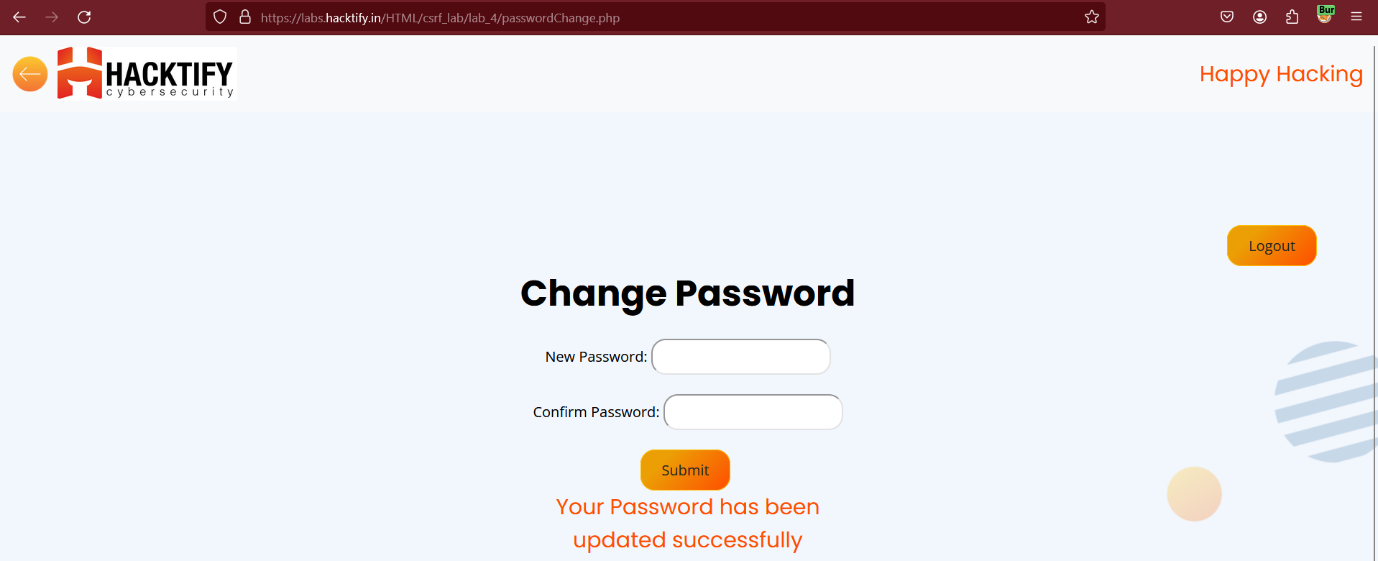
|  |  |
| --- | --- |
| **Reference** | **Risk Rating** |
| Always Validate Tokens | **Medium** |
| **Tools Used** | |
| Burp Suite and Hacktify Poc Generator. | |
| **Vulnerability Description** | |
| In the user account management functionality of a website, a CSRF (Cross-Site Request Forgery) vulnerability exists, allowing attackers to perform unauthorized actions on behalf of authenticated users. | |
| **How It Was Discovered** | |
| Manual Testing and analysis | |
| **Vulnerable URLs** | |
| https://labs.hacktify.in/HTML/csrf\_lab/lab\_2/index.php | |
| **Consequences of not Fixing the Issue** | |
| Unauthorized actions performed on behalf of authenticated users, such as changing passwords, modifying account settings, or initiating financial transactions.  Compromise of user accounts, leading to potential data loss, identity theft, or unauthorized access to sensitive information. | |
| **Suggested Countermeasures** | |
| Include anti-CSRF tokens or synchronizer tokens in forms and requests to validate the origin of requests and prevent CSRF attacks.  Implement strict validation of requests to ensure that sensitive actions, such as password changes, require explicit user consent and authentication. | |
| **References** | |
| <https://owasp.org/www-project-web-security-testing-guide/latest/> | |

# Proof of Concept

# 1.3. I Hate When Someone Uses My Tokens!

|  |  |
| --- | --- |
| **Reference** | **Risk Rating** |
| I Hate When Someone Uses My Tokens! | **Medium** |
| **Tools Used** | |
| Burp Suite and Hacktify Poc Generator. | |
| **Vulnerability Description** | |
| In the user account management functionality of a website, a CSRF (Cross-Site Request Forgery) vulnerability exists, allowing attackers to perform unauthorized actions on behalf of authenticated users. | |
| **How It Was Discovered** | |
| Manual Testing and analysis | |
| **Vulnerable URLs** | |
| https://labs.hacktify.in/HTML/csrf\_lab/lab\_4/index.php | |
| **Consequences of not Fixing the Issue** | |
| Unauthorized actions performed on behalf of authenticated users, such as changing passwords, modifying account settings, or initiating financial transactions.  Compromise of user accounts, leading to potential data loss, identity theft, or unauthorized access to sensitive information. | |
| **Suggested Countermeasures** | |
| Include anti-CSRF tokens or synchronizer tokens in forms and requests to validate the origin of requests and prevent CSRF attacks.  Implement strict validation of requests to ensure that sensitive actions, such as password changes, require explicit user consent and authentication. | |
| **References** | |
| <https://owasp.org/www-project-web-security-testing-guide/latest/> | |

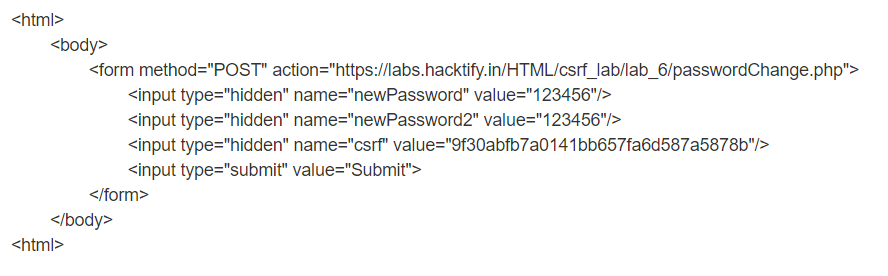
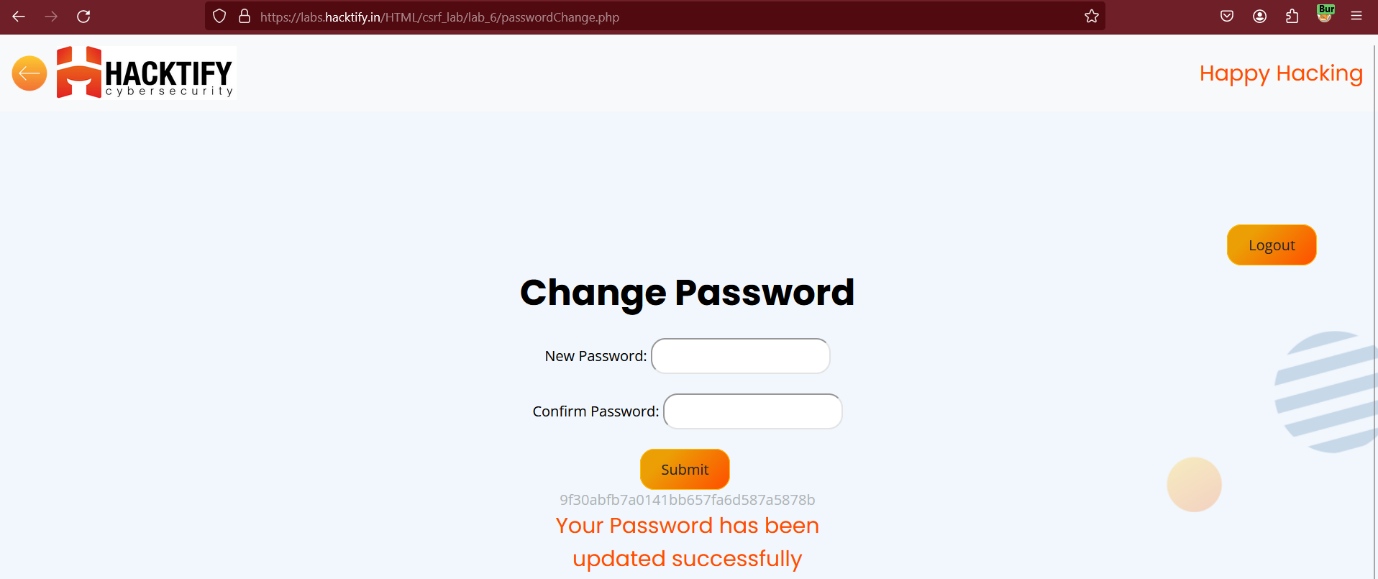
# Proof of Concept



# 1.4. GET Me or POST Me

|  |  |
| --- | --- |
| **Reference** | **Risk Rating** |
| GET Me or POST Me | **Easy** |
| **Tools Used** | |
| Burp Suite and Hacktify Poc Generator. | |
| **Vulnerability Description** | |
| In the user account management functionality of a website, a CSRF (Cross-Site Request Forgery) vulnerability exists, allowing attackers to perform unauthorized actions on behalf of authenticated users. | |
| **How It Was Discovered** | |
| Manual Testing and analysis | |
| **Vulnerable URLs** | |
| https://labs.hacktify.in/HTML/csrf\_lab/lab\_6/index.php | |
| **Consequences of not Fixing the Issue** | |
| Unauthorized actions performed on behalf of authenticated users, such as changing passwords, modifying account settings, or initiating financial transactions.  Compromise of user accounts, leading to potential data loss, identity theft, or unauthorized access to sensitive information. | |
| **Suggested Countermeasures** | |
| Include anti-CSRF tokens or synchronizer tokens in forms and requests to validate the origin of requests and prevent CSRF attacks.  Implement strict validation of requests to ensure that sensitive actions, such as password changes, require explicit user consent and authentication. | |
| **References** | |
| <https://portswigger.net/web-security/csrf/cheat-sheet> | |

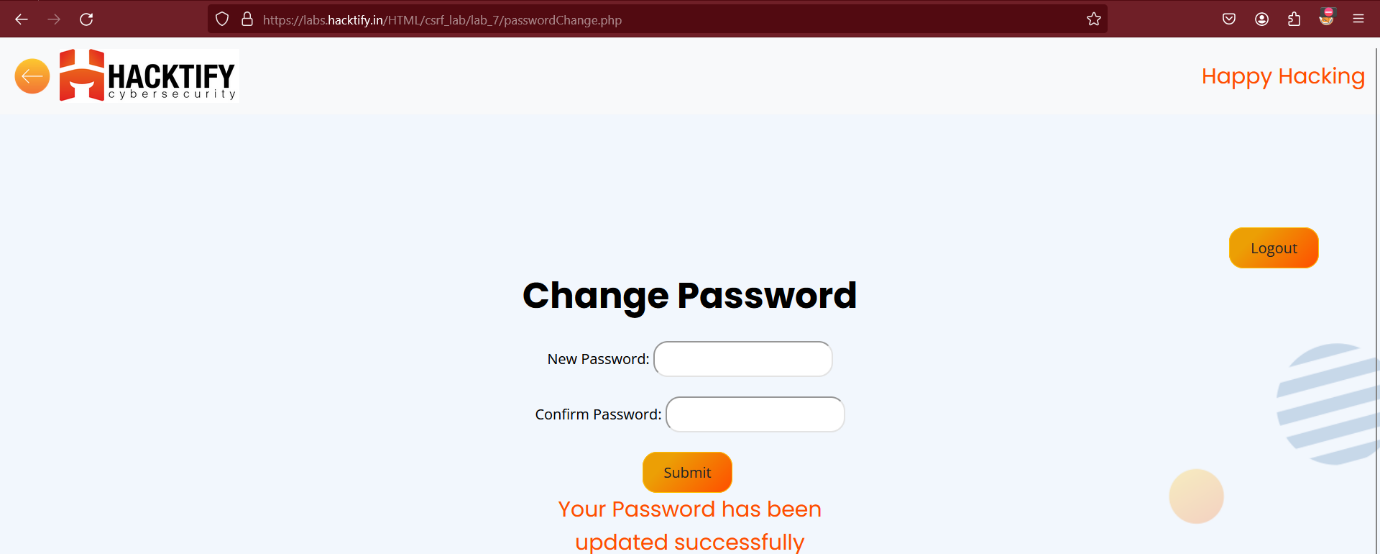
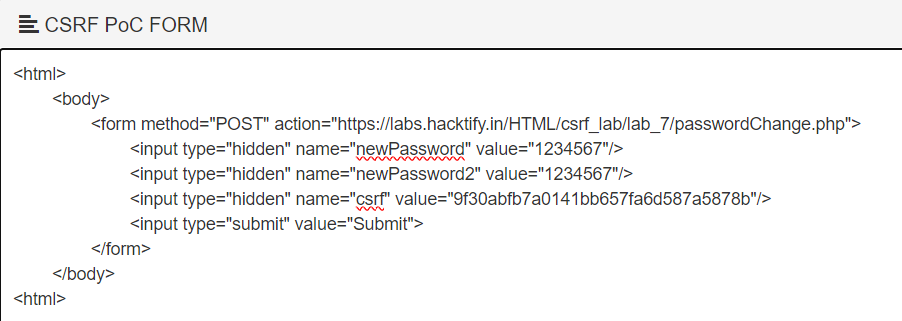
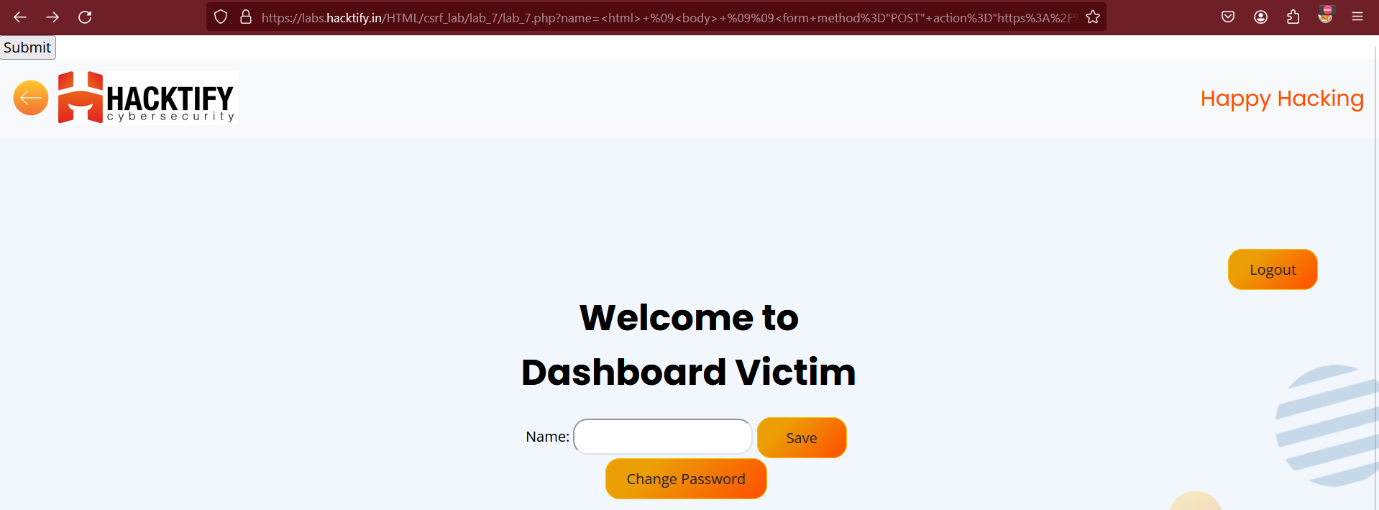
# Proof of Concept



# 1.5. XSS The Saviour

|  |  |
| --- | --- |
| **Reference** | **Risk Rating** |
| XSS The Saviour | **Hard** |
| **Tools Used** | |
| Burp Suite and Hacktify Poc Generator. | |
| **Vulnerability Description** | |
| In the user account management functionality of a website, a CSRF (Cross-Site Request Forgery) vulnerability exists, allowing attackers to perform unauthorized actions on behalf of authenticated users. | |
| **How It Was Discovered** | |
| Manual Testing and analysis | |
| **Vulnerable URLs** | |
| https://labs.hacktify.in/HTML/csrf\_lab/lab\_7/index.php | |
| **Consequences of not Fixing the Issue** | |
| Unauthorized actions performed on behalf of authenticated users, such as changing passwords, modifying account settings, or initiating financial transactions.  Compromise of user accounts, leading to potential data loss, identity theft, or unauthorized access to sensitive information. | |
| **Suggested Countermeasures** | |
| Include anti-CSRF tokens or synchronizer tokens in forms and requests to validate the origin of requests and prevent CSRF attacks.  Implement strict validation of requests to ensure that sensitive actions, such as password changes, require explicit user consent and authentication. | |
| **References** | |
| <https://owasp.org/www-project-web-security-testing-guide/latest/> | |

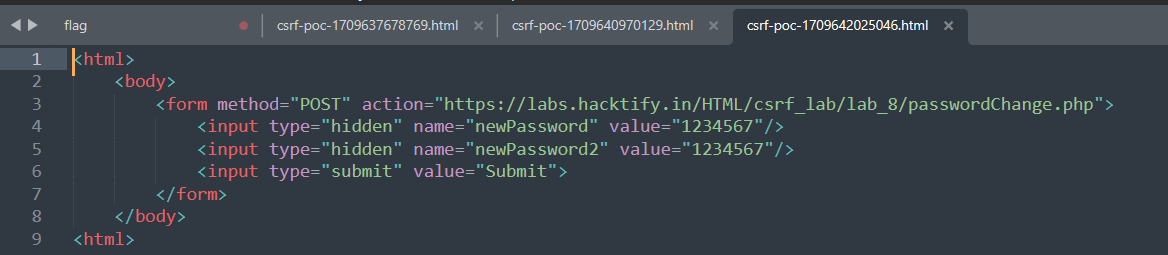
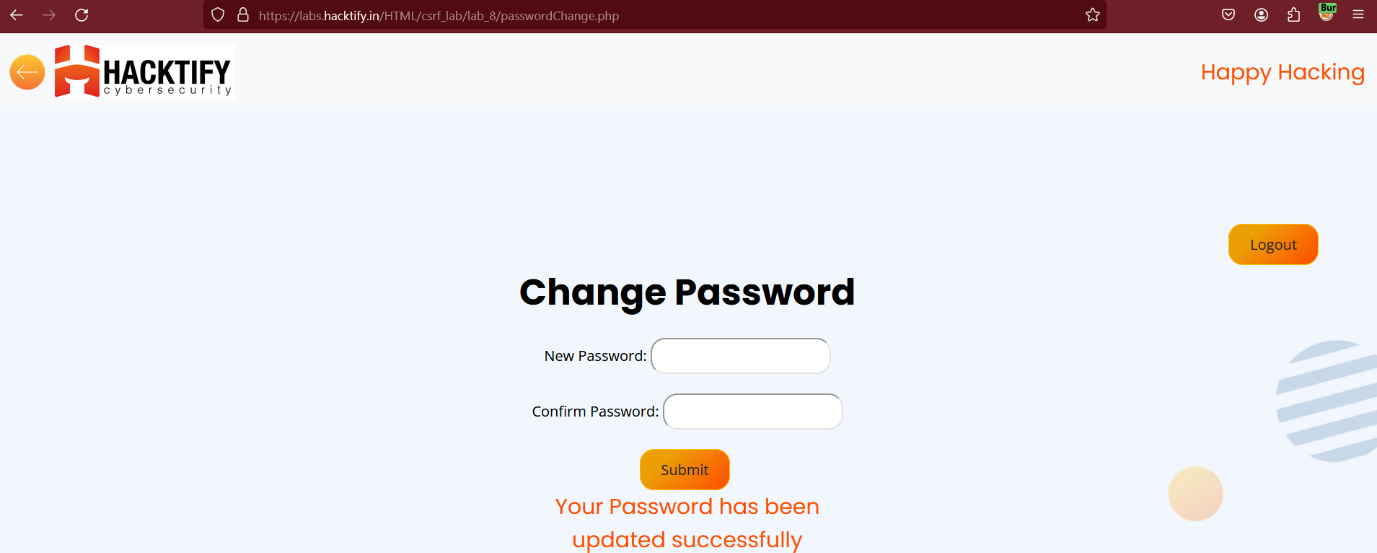
# Proof of Concept



# 1.6. Rm -Rf Token

|  |  |
| --- | --- |
| **Reference** | **Risk Rating** |
| Rm -Rf Token | **Hard** |
| **Tools Used** | |
| Burp Suite and Hacktify Poc Generator. | |
| **Vulnerability Description** | |
| In the user account management functionality of a website, a CSRF (Cross-Site Request Forgery) vulnerability exists, allowing attackers to perform unauthorized actions on behalf of authenticated users. | |
| **How It Was Discovered** | |
| Manual Testing and analysis | |
| **Vulnerable URLs** | |
| https://labs.hacktify.in/HTML/csrf\_lab/lab\_8/index.php | |
| **Consequences of not Fixing the Issue** | |
| Unauthorized actions performed on behalf of authenticated users, such as changing passwords, modifying account settings, or initiating financial transactions.  Compromise of user accounts, leading to potential data loss, identity theft, or unauthorized access to sensitive information. | |
| **Suggested Countermeasures** | |
| Include anti-CSRF tokens or synchronizer tokens in forms and requests to validate the origin of requests and prevent CSRF attacks.  Implement strict validation of requests to ensure that sensitive actions, such as password changes, require explicit user consent and authentication. | |
| **References** | |
| <https://owasp.org/www-project-web-security-testing-guide/latest/> | |

# Proof of Concept



# 2. Server-Side Request Forgery

# 2.1. Get the 127.0.0.1

|  |  |
| --- | --- |
| **Reference** | **Risk Rating** |
| Get the 127.0.0.1 | **Low** |
| **Tools Used** | |
| No tools are used | |
| **Vulnerability Description** | |
| In the SSRF (Server-Side Request Forgery) vulnerability lab, a security flaw exists in the website's functionality, allowing attackers to initiate unauthorized requests from the server to arbitrary destinations, including internal or external resources. | |
| **How It Was Discovered** | |
| Manual Analysis | |
| **Vulnerable URLs** | |
| https://labs.hacktify.in/HTML/ssrf\_lab/lab\_1/index.php | |
| **Consequences of not Fixing the Issue** | |
| Access to sensitive internal resources or data, such as files, databases, or configuration information, leading to data leakage or unauthorized access.  Bypassing of security controls, such as firewalls or network segmentation, to exfiltrate data, pivot to other systems, or escalate privileges within the network. | |
| **Suggested Countermeasures** | |
| Validate and sanitize user-supplied input to ensure that URLs or other parameters used for making server-side requests are properly formatted and do not allow access to internal or restricted resources.  Use whitelists or allowlists to restrict the destinations that the server can access, limiting requests to trusted domains or specific IP addresses | |
| **References** | |
| <https://cheatsheetseries.owasp.org/cheatsheets/Server_Side_Request_Forgery_Prevention_Cheat_Sheet.html> | |

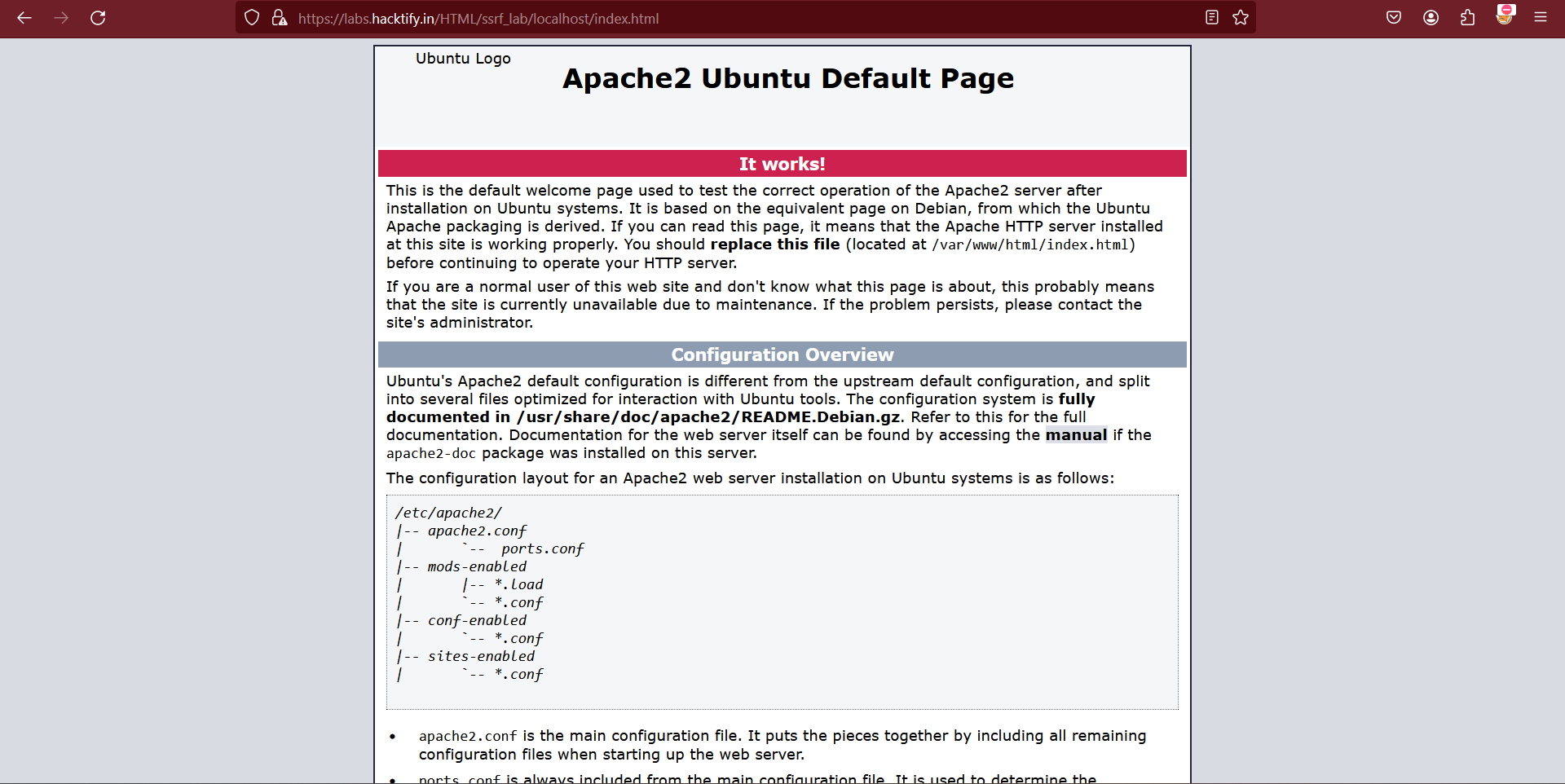
# Proof of Concept

# 

# 2.2. Http(s)? Nevermind!!

|  |  |
| --- | --- |
| **Reference** | **Risk Rating** |
| Http(s)? Nevermind!! | **Low** |
| **Tools Used** | |
| No tools are used | |
| **Vulnerability Description** | |
| In the SSRF (Server-Side Request Forgery) vulnerability lab, a security flaw exists in the website's functionality, allowing attackers to initiate unauthorized requests from the server to arbitrary destinations, including internal or external resources. | |
| **How It Was Discovered** | |
| Manual Analysis | |
| **Vulnerable URLs** | |
| https://labs.hacktify.in/HTML/ssrf\_lab/lab\_2/index.php | |
| **Consequences of not Fixing the Issue** | |
| Access to sensitive internal resources or data, such as files, databases, or configuration information, leading to data leakage or unauthorized access.  Bypassing of security controls, such as firewalls or network segmentation, to exfiltrate data, pivot to other systems, or escalate privileges within the network. | |
| **Suggested Countermeasures** | |
| Validate and sanitize user-supplied input to ensure that URLs or other parameters used for making server-side requests are properly formatted and do not allow access to internal or restricted resources.  Use whitelists or allowlists to restrict the destinations that the server can access, limiting requests to trusted domains or specific IP addresses | |
| **References** | |
| <https://portswigger.net/web-security/ssrf/cheat-sheet> | |

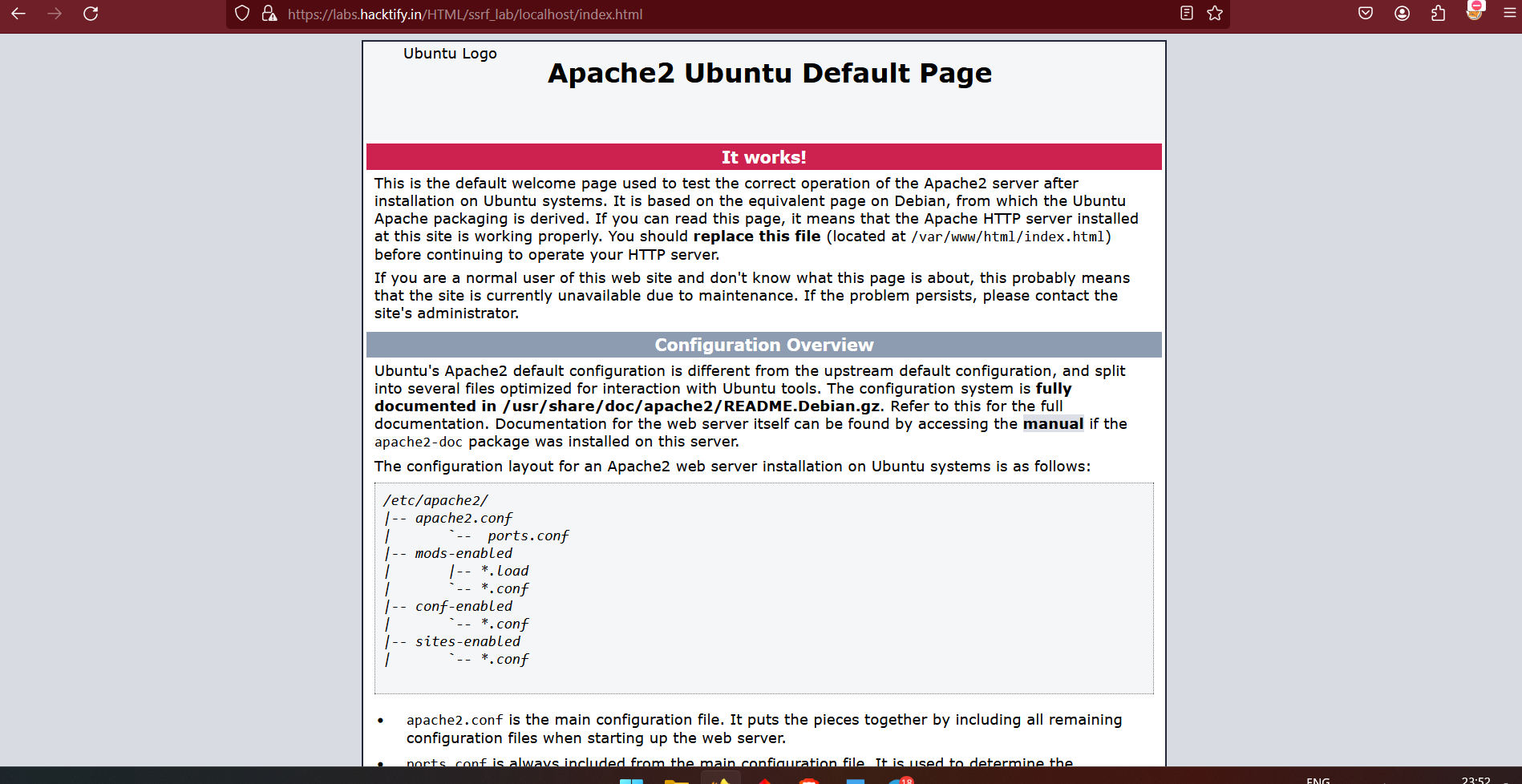
# Proof of Concept

****

# 2.3. “:” The Saviour

|  |  |
| --- | --- |
| **Reference** | **Risk Rating** |
| “:” The Saviour | **Low** |
| **Tools Used** | |
| No tools are used | |
| **Vulnerability Description** | |
| In the SSRF (Server-Side Request Forgery) vulnerability lab, a security flaw exists in the website's functionality, allowing attackers to initiate unauthorized requests from the server to arbitrary destinations, including internal or external resources. | |
| **How It Was Discovered** | |
| Manual Analysis | |
| **Vulnerable URLs** | |
| https://labs.hacktify.in/HTML/ssrf\_lab/lab\_3/index.php | |
| **Consequences of not Fixing the Issue** | |
| Access to sensitive internal resources or data, such as files, databases, or configuration information, leading to data leakage or unauthorized access.  Bypassing of security controls, such as firewalls or network segmentation, to exfiltrate data, pivot to other systems, or escalate privileges within the network. | |
| **Suggested Countermeasures** | |
| Validate and sanitize user-supplied input to ensure that URLs or other parameters used for making server-side requests are properly formatted and do not allow access to internal or restricted resources.  Use whitelists or allowlists to restrict the destinations that the server can access, limiting requests to trusted domains or specific IP addresses | |
| **References** | |
| <https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Server%20Side%20Request%20Forgery/README.md> | |

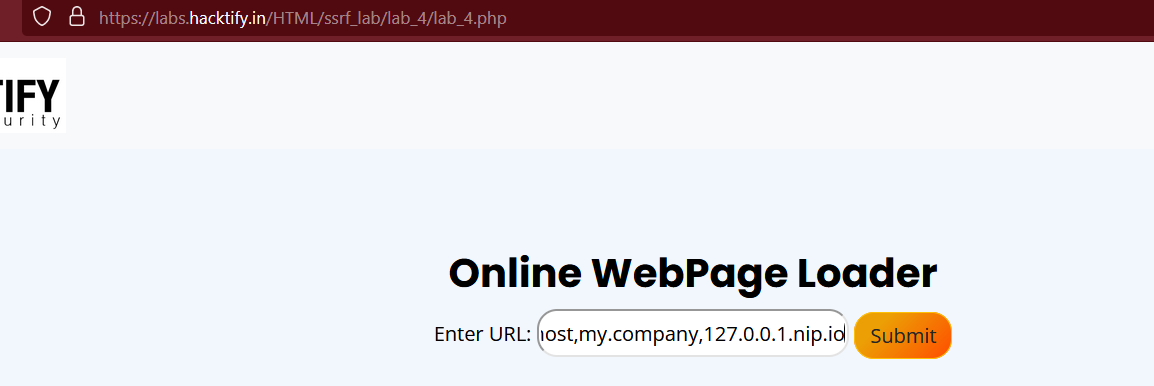
# Proof of Concept



# 2.4. Messed Up Domain!!

|  |  |
| --- | --- |
| **Reference** | **Risk Rating** |
| Http(s)? Nevermind!! | **Medium** |
| **Tools Used** | |
| No tools are used | |
| **Vulnerability Description** | |
| In the SSRF (Server-Side Request Forgery) vulnerability lab, a security flaw exists in the website's functionality, allowing attackers to initiate unauthorized requests from the server to arbitrary destinations, including internal or external resources. | |
| **How It Was Discovered** | |
| Manual Analysis | |
| **Vulnerable URLs** | |
| https://labs.hacktify.in/HTML/ssrf\_lab/lab\_4/index.php | |
| **Consequences of not Fixing the Issue** | |
| Access to sensitive internal resources or data, such as files, databases, or configuration information, leading to data leakage or unauthorized access.  Bypassing of security controls, such as firewalls or network segmentation, to exfiltrate data, pivot to other systems, or escalate privileges within the network. | |
| **Suggested Countermeasures** | |
| Validate and sanitize user-supplied input to ensure that URLs or other parameters used for making server-side requests are properly formatted and do not allow access to internal or restricted resources.  Use whitelists or allowlists to restrict the destinations that the server can access, limiting requests to trusted domains or specific IP addresses | |
| **References** | |
| <https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/CSRF%20Injection> | |

# Proof of Concept



# 2.10. Look an SSRF on Cloud!

|  |  |
| --- | --- |
| **Reference** | **Risk Rating** |
| Look an SSRF on Cloud! | **Hard** |
| **Tools Used** | |
| No tools are used | |
| **Vulnerability Description** | |
| In the SSRF (Server-Side Request Forgery) vulnerability lab, a security flaw exists in the website's functionality, allowing attackers to initiate unauthorized requests from the server to arbitrary destinations, including internal or external resources. | |
| **How It Was Discovered** | |
| Manual Analysis | |
| **Vulnerable URLs** | |
| https://labs.hacktify.in/HTML/ssrf\_lab/lab\_10/index.php | |
| **Consequences of not Fixing the Issue** | |
| Access to sensitive internal resources or data, such as files, databases, or configuration information, leading to data leakage or unauthorized access.  Bypassing of security controls, such as firewalls or network segmentation, to exfiltrate data, pivot to other systems, or escalate privileges within the network. | |
| **Suggested Countermeasures** | |
| Validate and sanitize user-supplied input to ensure that URLs or other parameters used for making server-side requests are properly formatted and do not allow access to internal or restricted resources.  Use whitelists or allowlists to restrict the destinations that the server can access, limiting requests to trusted domains or specific IP addresses | |
| **References** | |
| <https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Server%20Side%20Request%20Forgery/README.md> | |

# Proof of Concept

