**Week 5**

**Penetration Testing Report**

**Introduction**

This report document hereby describes the proceedings and results of a Black Box security assessment conducted against the **Week 5 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 5 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-Origin Resource Sharing** |
| --- | --- |

**3. Summary**

Outlined is a Black Box Application Security assessment for the **Week 5 Labs**.

**Total number of Sub-labs: 7 Sub-labs**

| **High** | **Medium** | **Low** |
| --- | --- | --- |
| **3** | **2** | **2** |

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

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

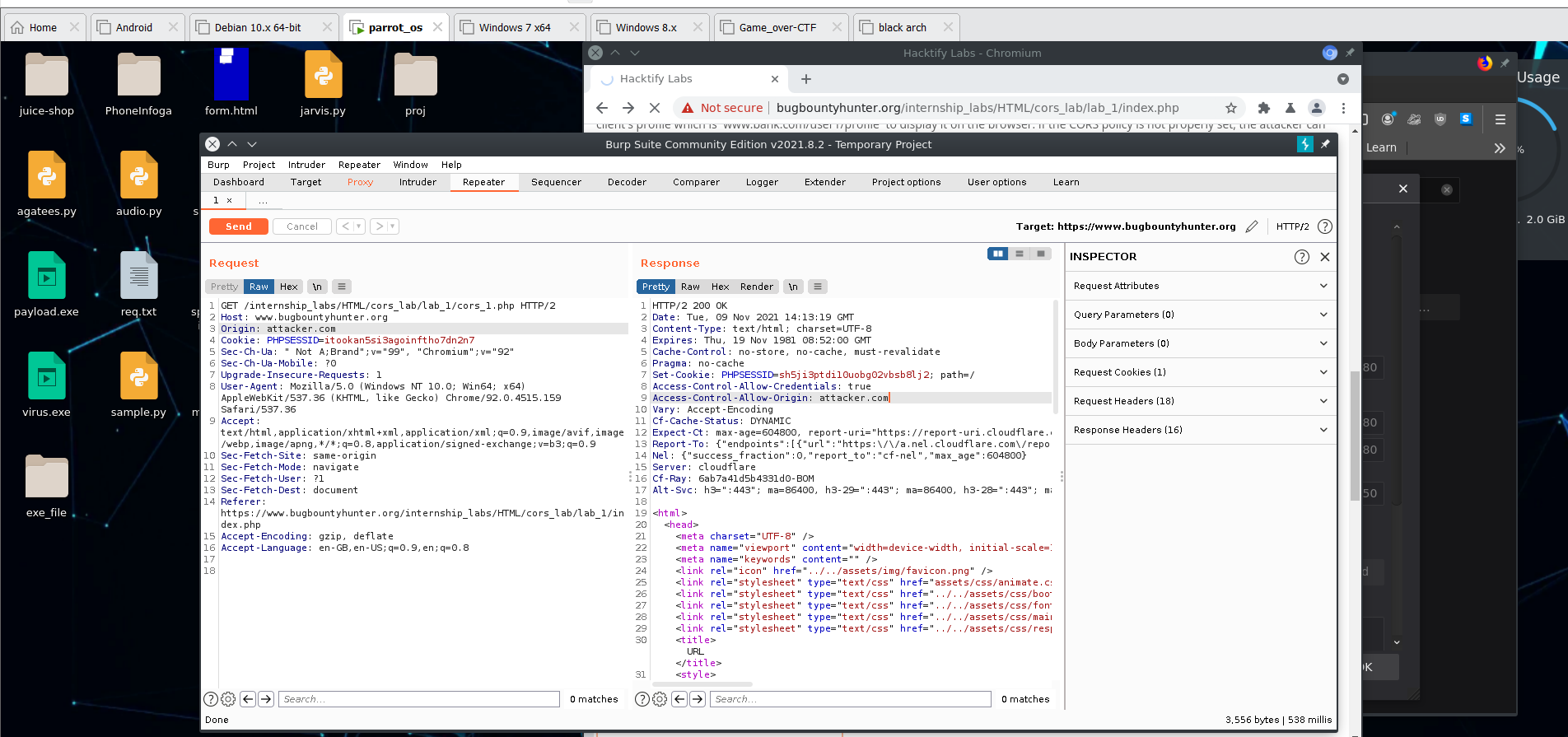
**Low - Number of Sub-labs with Easy difficulty level**

# 1. Cross-Origin Resource Sharing

# 1.1. CORS With Arbitrary Origin

| **Reference** | **Risk Rating** |  |
| --- | --- | --- |
| CORS With Arbitrary Origin | **Medium** |  |
| **Tools Used** | |  |
| Burp Suite | |  |
| **Vulnerability Description** | |  |
| Cross-origin resource sharing (CORS) is a browser mechanism which enables controlled access to resources located outside of a given domain. It  extends and adds flexibility to the same-origin policy. However, it also provides potential for cross-domain based attacks, if a website’s CORS  policy is poorly configured and implemented. | |  |
| **How It Was Discovered** | |  |
| Manual Analysis | |  |
| **Vulnerable URLs** | |  |
| <https://www.bugbountyhunter.org/internship_labs/HTML/cors_lab/lab_1/cors_1.php> | |  |
| **Consequences of not Fixing the Issue** | |  |
| Attacker would treat many victims to visit attacker's website, if victim is logged in, then his personal information is recorded in attacker's server.  Attacker can perform any action in the user's account, bypassing CSRF tokes. | |  |
| **Suggested Countermeasures** | |  |
| **Only allow trusted sites :** Dynamically reflecting origins from cross-domain requests without validation is readily exploitable and should be  avoided. | |  |
| **References** | |  |
| <https://portswigger.net/web-security/cors> | |  |

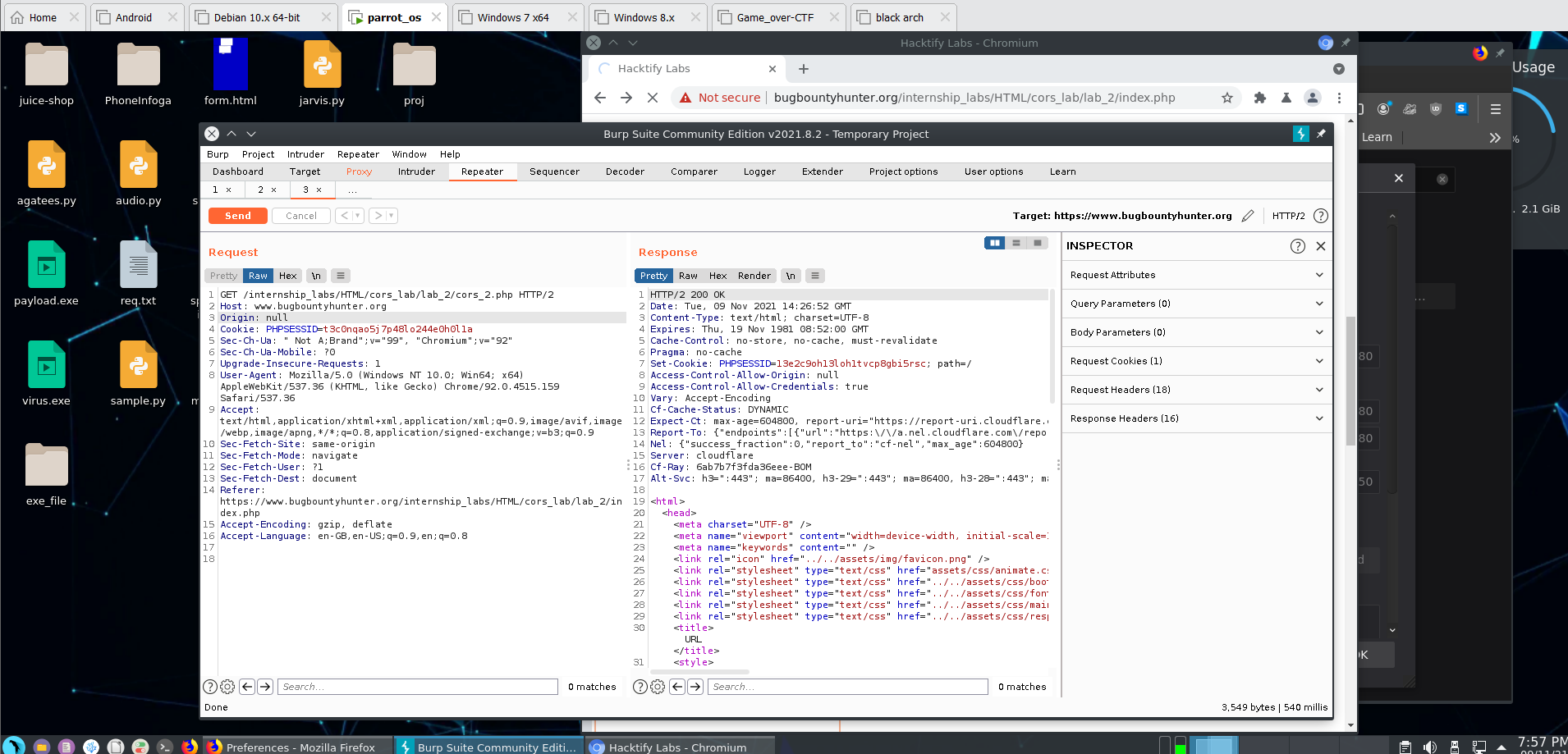
# Proof of Concept



# 1.2.CORS With Null Origin

| **Reference** | **Risk Rating** |
| --- | --- |
| CORS With Null Origin | **Low / Medium / High** |
| **Tools Used** | |
| Burp Suite | |
| **Vulnerability Description** | |
| Cross-origin resource sharing (CORS) is a browser mechanism which enables controlled access to resources located outside of a given domain. It  extends and adds flexibility to the same-origin policy. However, it also provides potential for cross-domain based attacks, if a website’s CORS  policy is poorly configured and implemented. | |
| **How It Was Discovered** | |
| Manual Analysis | |
| **Vulnerable URLs** | |
| <https://www.bugbountyhunter.org/internship_labs/HTML/cors_lab/lab_2/cors_2.php> | |
| **Consequences of not Fixing the Issue** | |
| Attacker would treat many victims to visit attacker's website, if victim is logged in, then his personal information is recorded in attacker's server.  Attacker can perform any action in the user's account, bypassing CSRF tokes. | |
| **Suggested Countermeasures** | |
| **Avoid whitelisting null :** Avoid using the header Access-Control-Allow-Origin: null . Cross-domain resource calls from internal documents  and sandboxed requests can specify the null origin. CORS headers should be properly defined in respect of trusted origins for private and  public servers. | |
| **References** | |
| <https://owasp.org/www-community/attacks/CORS_OriginHeaderScrutiny> | |

# Proof of Concept

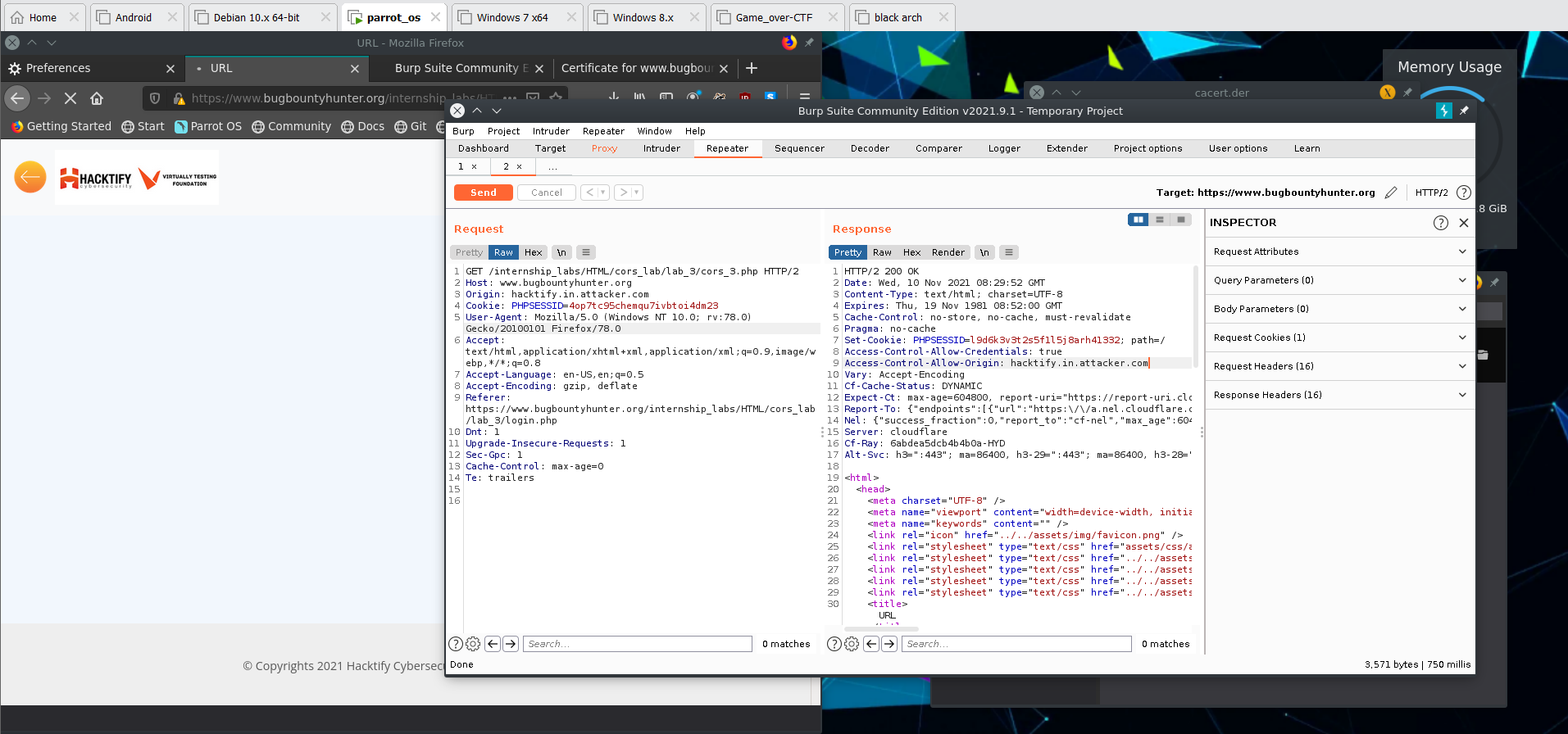


# 1.3. CORS With Prefix Match

| **Reference** | **Risk Rating** |
| --- | --- |
| CORS With Prefix Match | **Low / Medium / High** |
| **Tools Used** | |
| Burp Suite | |
| **Vulnerability Description** | |
| Cross-origin resource sharing (CORS) is a browser mechanism which enables controlled access to resources located outside of a given domain. It  extends and adds flexibility to the same-origin policy. However, it also provides potential for cross-domain based attacks, if a website’s CORS  policy is poorly configured and implemented. | |
| **How It Was Discovered** | |
| Manual Analysis | |
| **Vulnerable URLs** | |
| <https://www.bugbountyhunter.org/internship_labs/HTML/cors_lab/lab_3/cors_3.php> | |
| **Consequences of not Fixing the Issue** | |
| Attacker would treat many victims to visit attacker's website, if victim is logged in, then his personal information is recorded in attacker's server.  Attacker can perform any action in the user's account, bypassing CSRF tokes. | |
| **Suggested Countermeasures** | |
| **Proper configuration of cross-domain requests :** If a web resource contains sensitive information, the origin should be properly specified  in the Access-Control-Allow-Origin header. | |
| **References** | |
| <https://owasp.org/www-community/attacks/CORS_OriginHeaderScrutiny> | |

# 

# Proof of Concept

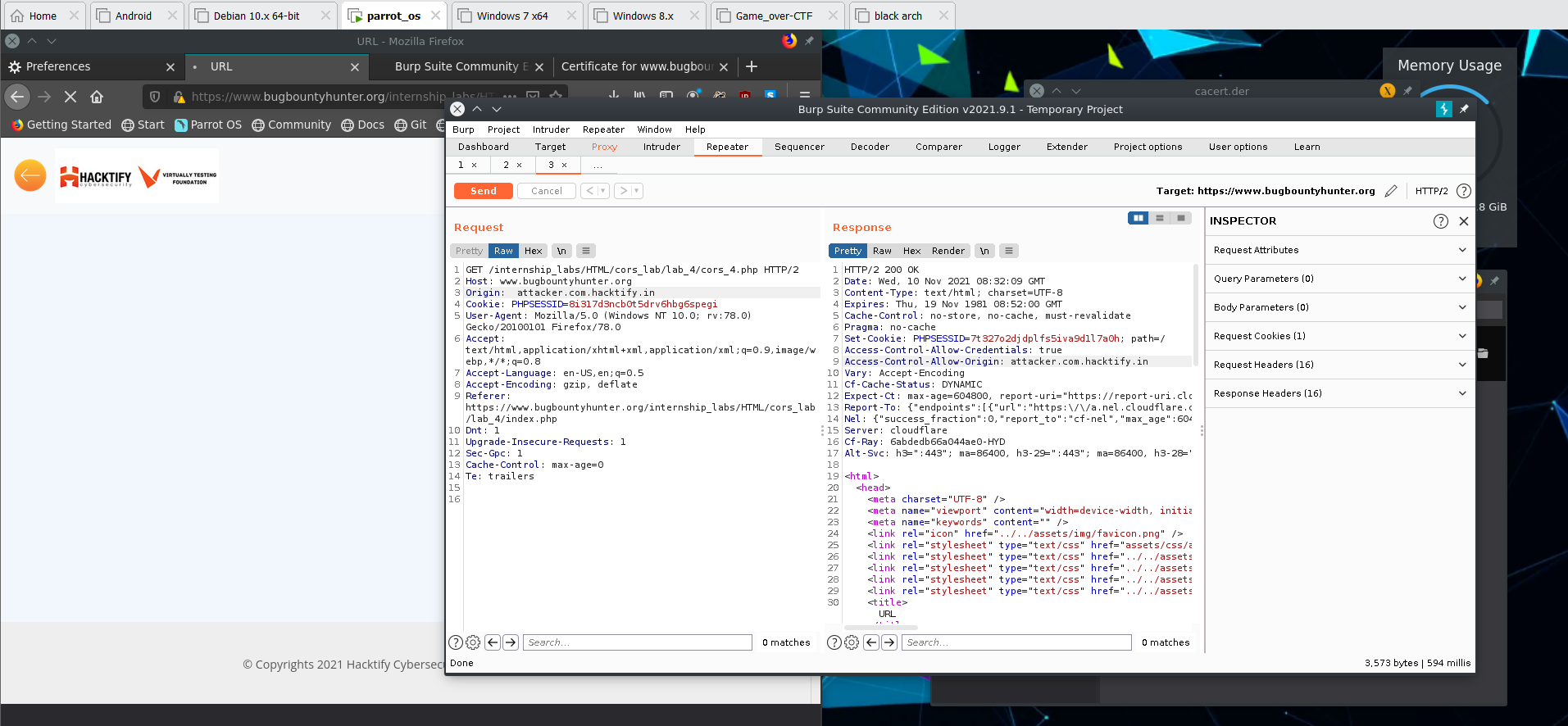


# 1.4. CORS With Suffix Match

| **Reference** | **Risk Rating** |
| --- | --- |
| CORS With Suffix Match | **Low / Medium / High** |
| **Tools Used** | |
| Burp Suite | |
| **Vulnerability Description** | |
| Cross-origin resource sharing (CORS) is a browser mechanism which enables controlled access to resources located outside of a given domain. It  extends and adds flexibility to the same-origin policy. However, it also provides potential for cross-domain based attacks, if a website’s CORS  policy is poorly configured and implemented. | |
| **How It Was Discovered** | |
| Manual Analysis | |
| **Vulnerable URLs** | |
| <https://www.bugbountyhunter.org/internship_labs/HTML/cors_lab/lab_4/cors_4.php> | |
| **Consequences of not Fixing the Issue** | |
| Attacker would treat many victims to visit attacker's website, if victim is logged in, then his personal information is recorded in attacker's server.  Attacker can perform any action in the user's account, bypassing CSRF tokes. | |
| **Suggested Countermeasures** | |
| **Only allow trusted sites :** Dynamically reflecting origins from cross-domain requests without validation is readily exploitable and should be  avoided. | |
| **References** | |
| <https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS> | |

# 

# Proof of Concept

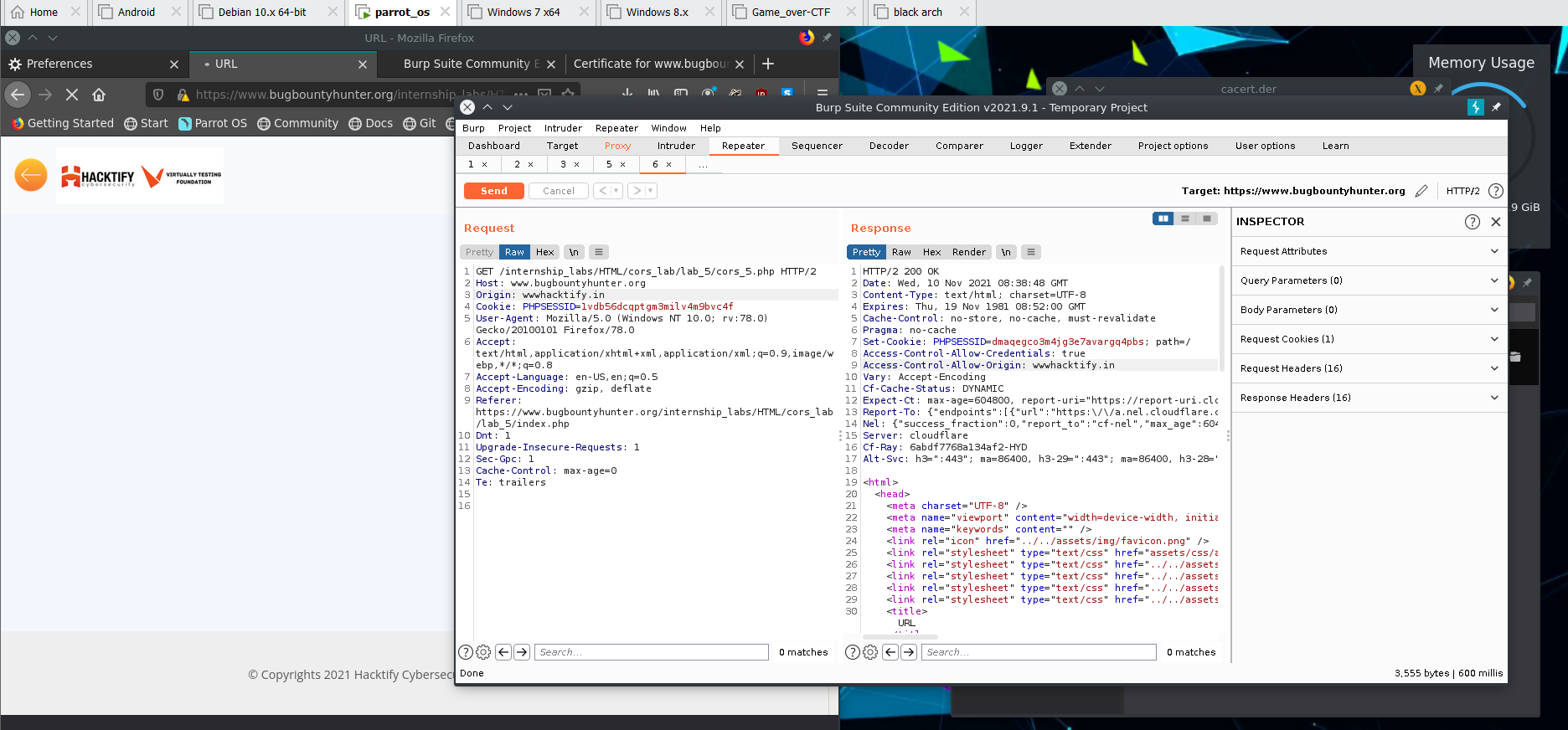


# 1.5.CORS With Escape Dot

| **Reference** | **Risk Rating** |
| --- | --- |
| CORS With Escape Dot | **Low / Medium / High** |
| **Tools Used** | |
| Burp Suite | |
| **Vulnerability Description** | |
| Cross-origin resource sharing (CORS) is a browser mechanism which enables controlled access to resources located outside of a given domain. It  extends and adds flexibility to the same-origin policy. However, it also provides potential for cross-domain based attacks, if a website’s CORS  policy is poorly configured and implemented. | |
| **How It Was Discovered** | |
| Manual Analysis | |
| **Vulnerable URLs** | |
| <https://www.bugbountyhunter.org/internship_labs/HTML/cors_lab/lab_5/cors_5.php> | |
| **Consequences of not Fixing the Issue** | |
| Attacker would treat many victims to visit attacker's website, if victim is logged in, then his personal information is recorded in attacker's server.  Attacker can perform any action in the user's account, bypassing CSRF tokes. | |
| **Suggested Countermeasures** | |
| **Only allow trusted sites :** Dynamically reflecting origins from cross-domain requests without validation is readily exploitable and should be  avoided. | |
| **References** | |
| <https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS> | |

# 

# Proof of Concept

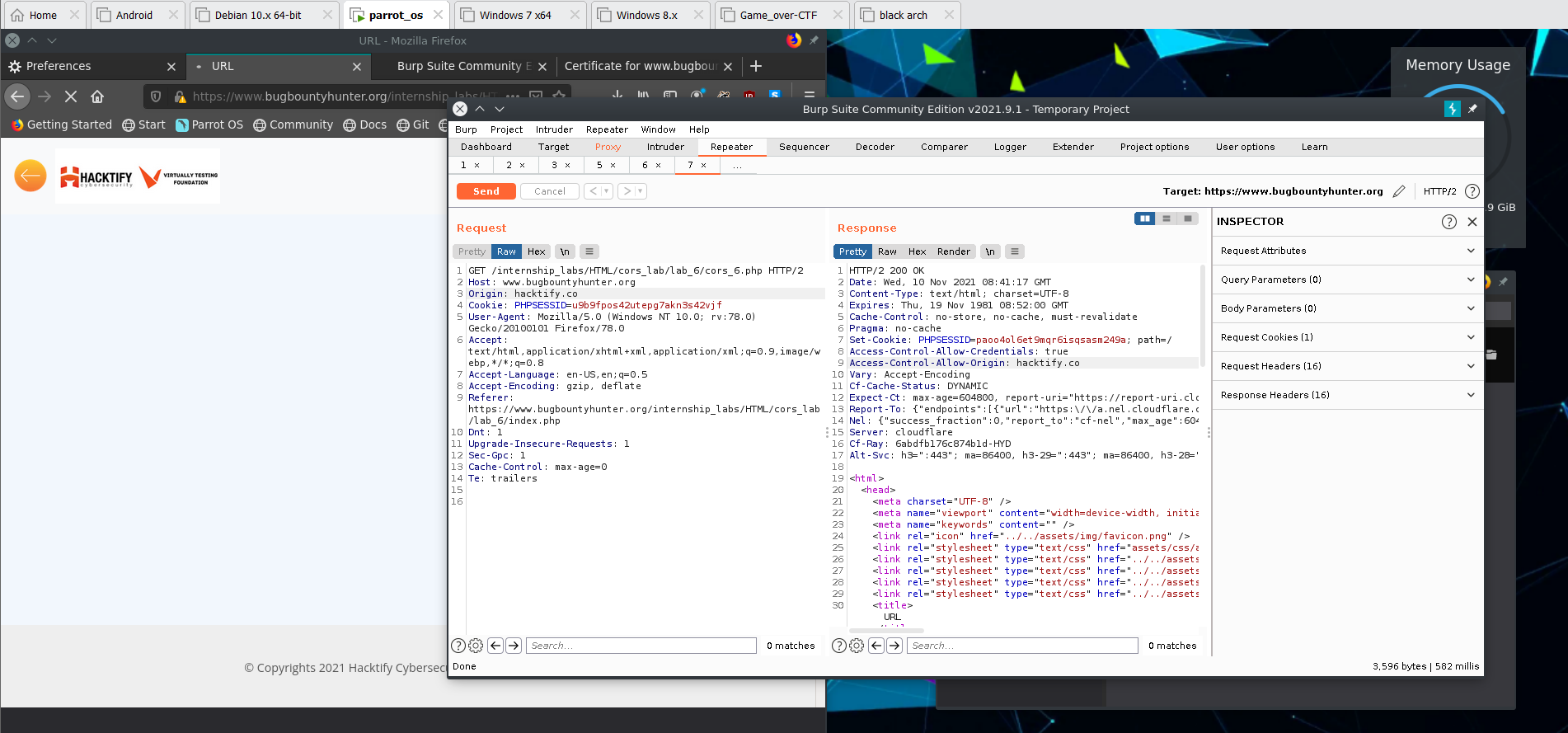


# 1.6. CORS With Substring Match

| **Reference** | **Risk Rating** |
| --- | --- |
| CORS With Substring Match | **Low / Medium / High** |
| **Tools Used** | |
| Burp Suite | |
| **Vulnerability Description** | |
| Cross-origin resource sharing (CORS) is a browser mechanism which enables controlled access to resources located outside of a given domain. It  extends and adds flexibility to the same-origin policy. However, it also provides potential for cross-domain based attacks, if a website’s CORS  policy is poorly configured and implemented. | |
| **How It Was Discovered** | |
| Manual Analysis | |
| **Vulnerable URLs** | |
| <https://www.bugbountyhunter.org/internship_labs/HTML/cors_lab/lab_6/cors_6.php> | |
| **Consequences of not Fixing the Issue** | |
| Attacker would treat many victims to visit attacker's website, if victim is logged in, then his personal information is recorded in attacker's server.  Attacker can perform any action in the user's account, bypassing CSRF tokes. | |
| **Suggested Countermeasures** | |
| **Proper configuration of cross-domain requests :** If a web resource contains sensitive information, the origin should be properly specified  in the Access-Control-Allow-Origin header. | |
| **References** | |
| <https://owasp.org/www-community/attacks/CORS_OriginHeaderScrutiny> | |

# 

# Proof of Concept



# 1.7.CORS With Arbitrary Subdomain

| **Reference** | **Risk Rating** |
| --- | --- |
| CORS With Arbitrary Subdomain | **Low / Medium / High** |
| **Tools Used** | |
| Burp Suite | |
| **Vulnerability Description** | |
| Cross-origin resource sharing (CORS) is a browser mechanism which enables controlled access to resources located outside of a given domain. It  extends and adds flexibility to the same-origin policy. However, it also provides potential for cross-domain based attacks, if a website’s CORS  policy is poorly configured and implemented. | |
| **How It Was Discovered** | |
| Manual Analysis | |
| **Vulnerable URLs** | |
| <https://www.bugbountyhunter.org/internship_labs/HTML/cors_lab/lab_7/cors_7.php> | |
| **Consequences of not Fixing the Issue** | |
| Attacker would treat many victims to visit attacker's website, if victim is logged in, then his personal information is recorded in attacker's server.  Attacker can perform any action in the user's account, bypassing CSRF tokes. | |
| **Suggested Countermeasures** | |
| **Only allow trusted sites :** Dynamically reflecting origins from cross-domain requests without validation is readily exploitable and should be  avoided. | |
| **References** | |
| <https://portswigger.net/web-security/cors> | |

# 

# Proof of Concept

