**Week 3**

**Penetration Testing Report**

**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 Scripting** |
| --- | --- |

**3. Summary**

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

**Total number of Sub-labs: 11 Sub-labs**

| **High** | **Medium** | **Low** |
| --- | --- | --- |
| **3** | **3** | **5** |

**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 Scripting Labs

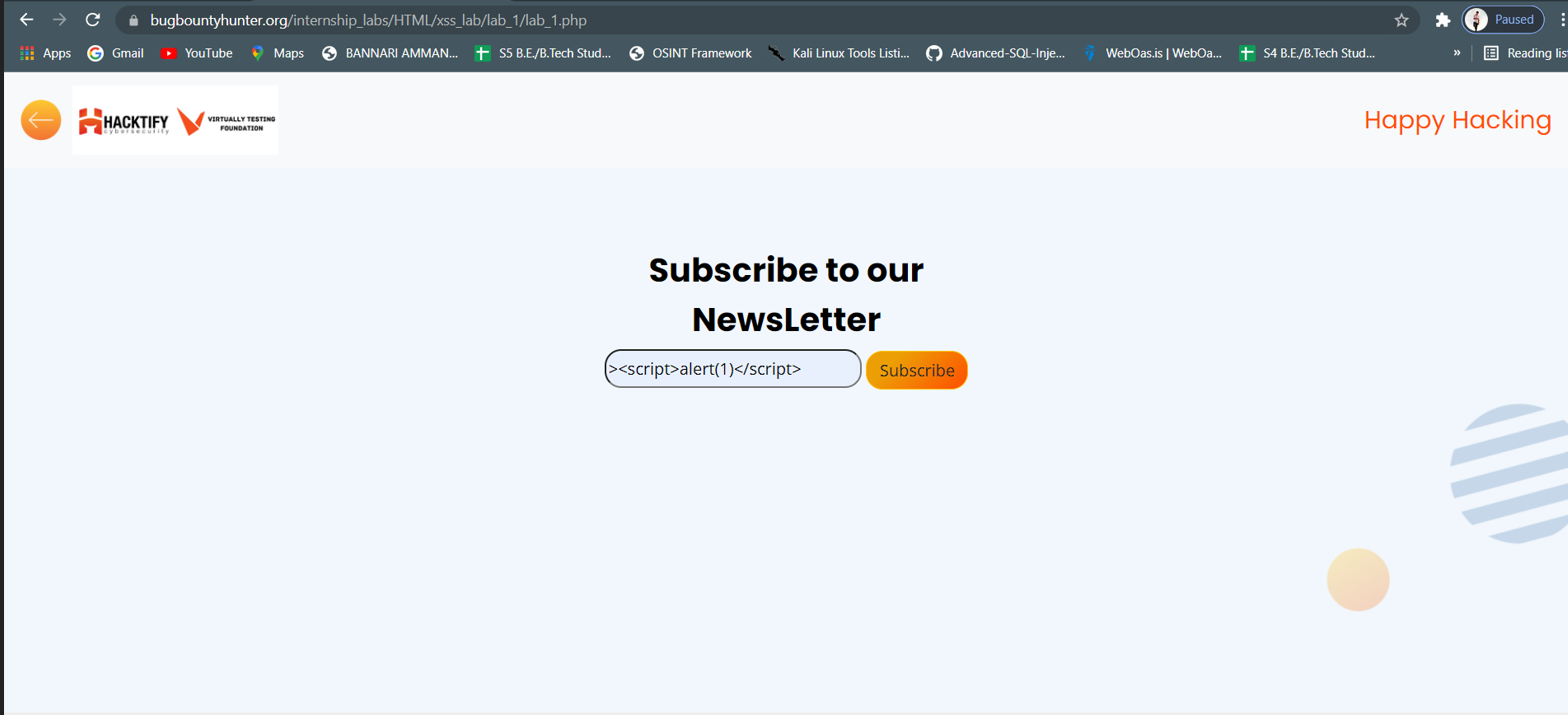
# 1.1. Let's Do IT!

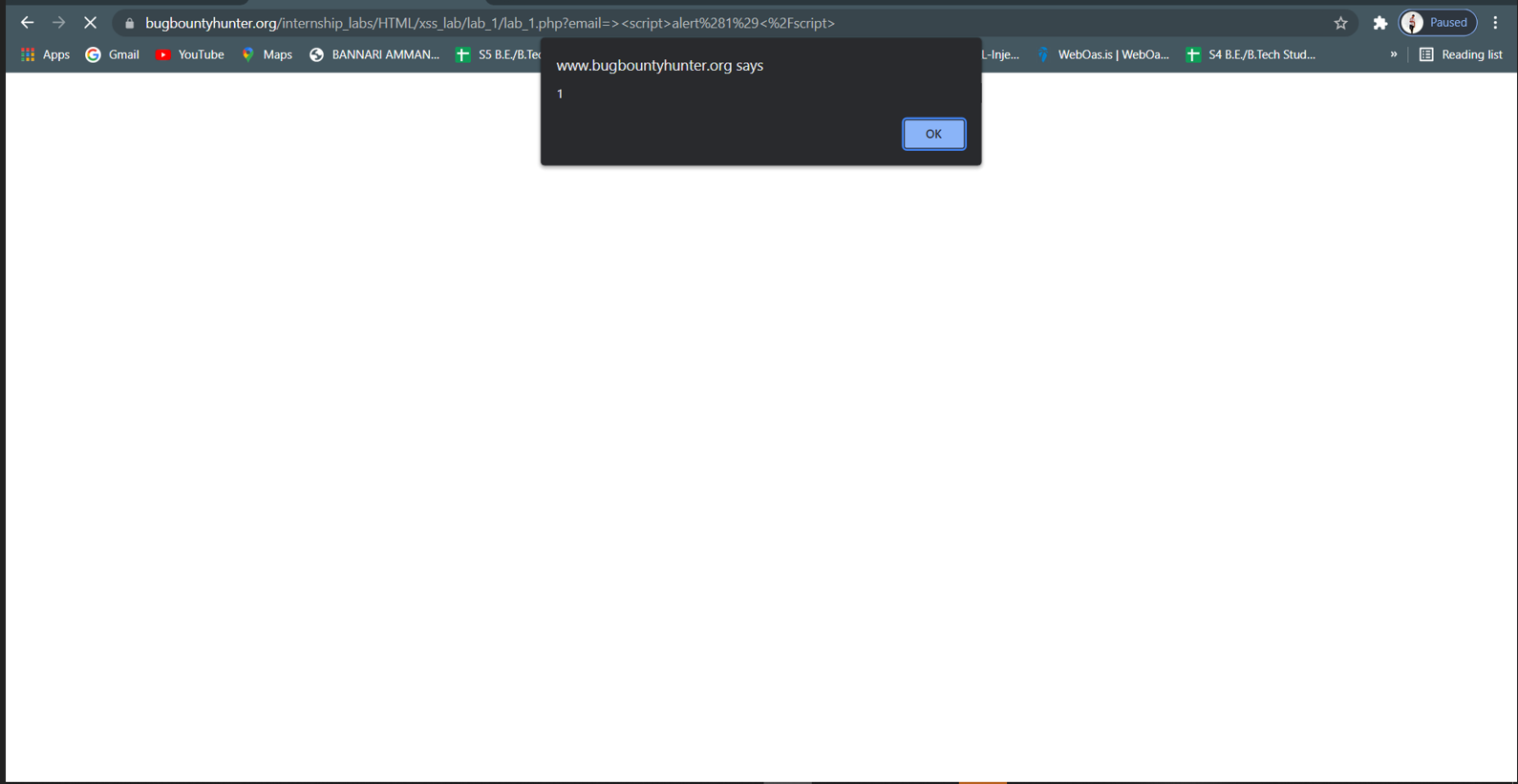
| **Reference** | **Risk Rating** |  |
| --- | --- | --- |
| Let's Do IT! | **Medium** |  |
| **Tools Used** | |  |
| By manual checking | |  |
| **Vulnerability Description** | |  |
| Reflected XSS attacks, also known as non-persistent attacks, occur when a malicious script is reflected off of a web application to the victim's browser.the payload is passed through the url. | |  |
| **How It Was Discovered** | |  |
| Manual Analysis | |  |
| **Vulnerable URLs** | |  |
| <https://www.bugbountyhunter.org/internship_labs/HTML/xss_lab/lab_1/lab_1.php> | |  |
| **Consequences of not Fixing the Issue** | |  |
| Perform any action within the application that the user can perform. View any information that the user is able to view. Modify any information that the user is able to modify. | |  |
| **Suggested Countermeasures** | |  |
| filter the user input and don’t allow javascript as user entry field and don’t allow javascript in url also, don’t allow special character as input. | |  |
| **References** | |  |
| <https://portswigger.net/web-security/cross-site-scripting/reflected> | |  |

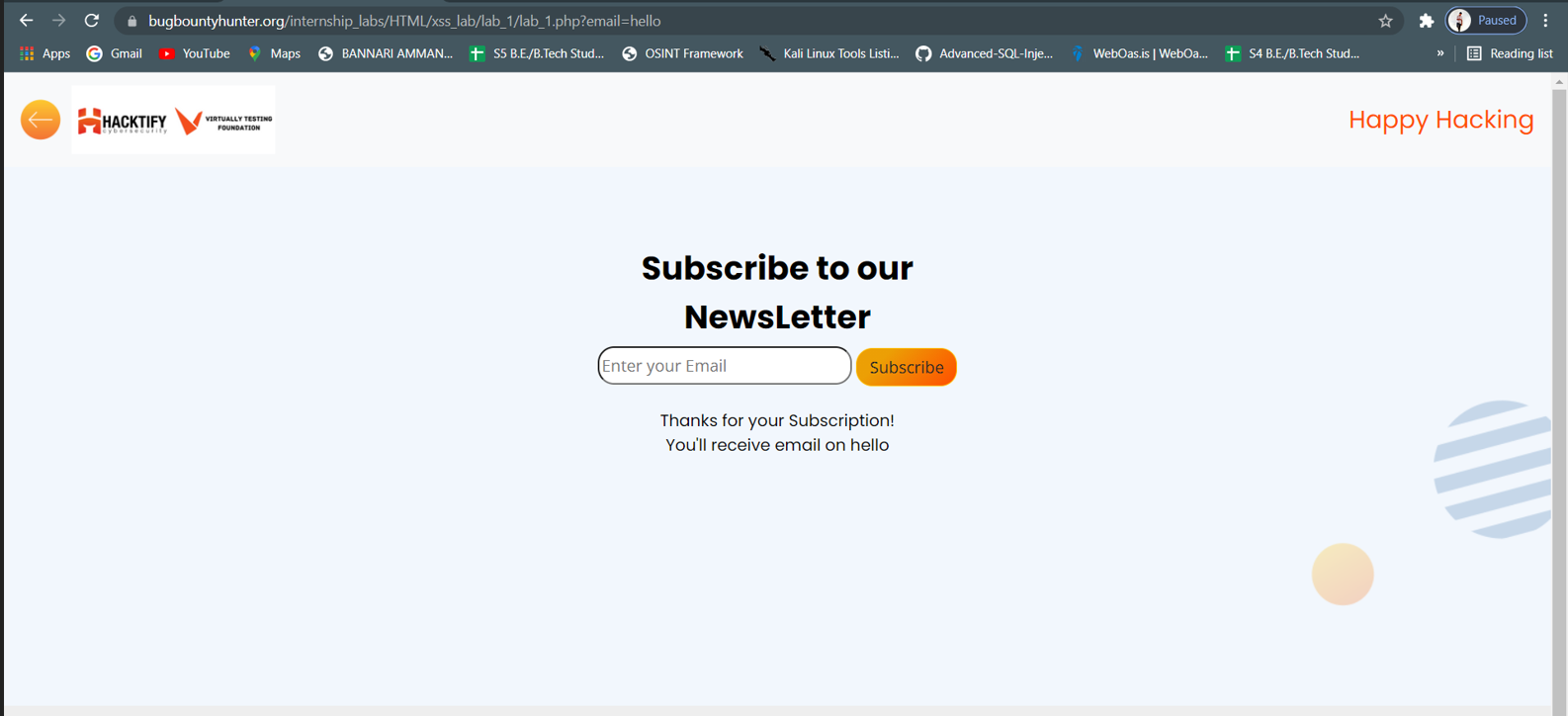
# 

# Proof of Concept

payload:><script>alert(1)</script>







# 1.2. Balancing Is Important In Life!

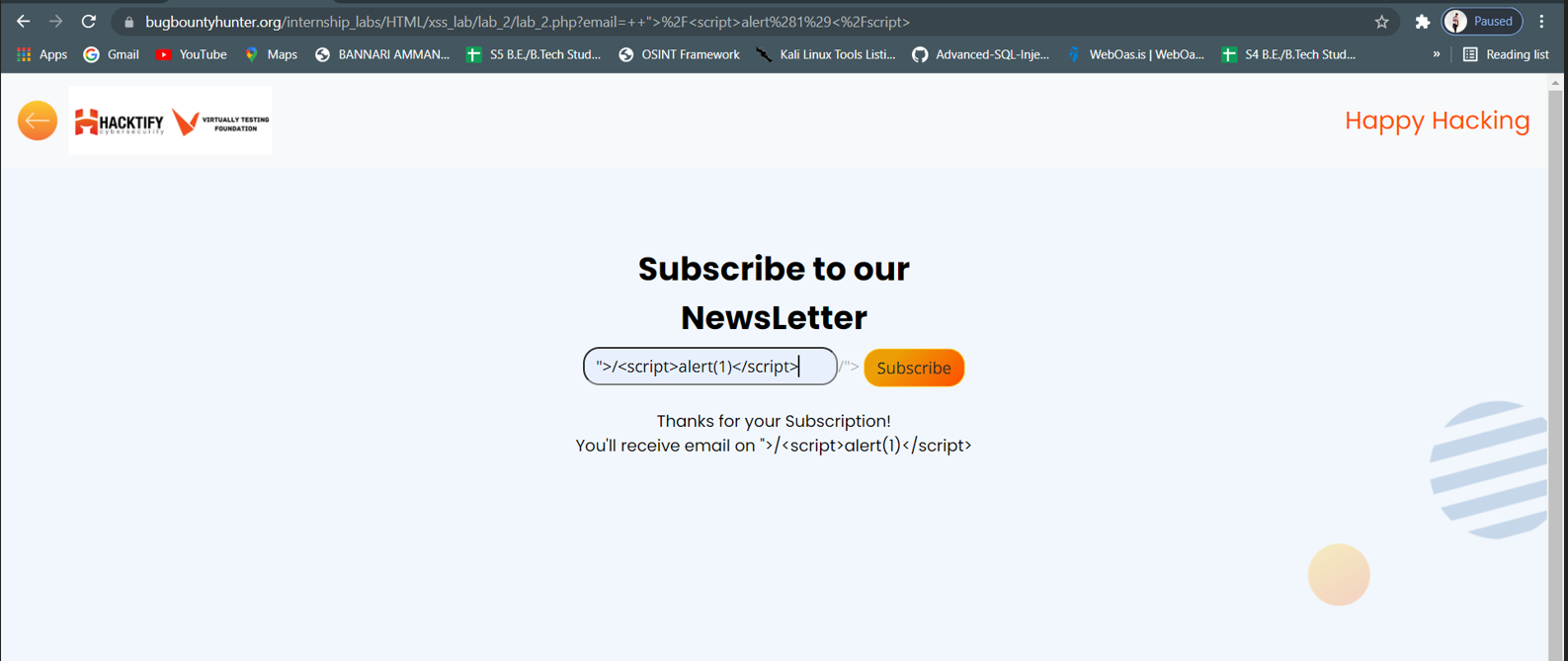
| **Reference** | **Risk Rating** |  |
| --- | --- | --- |
| Balancing Is Important In Life! | **Medium** |  |
| **Tools Used** | |  |
| By manual checking | |  |
| **Vulnerability Description** | |  |
| Reflected XSS attacks, also known as non-persistent attacks, occur when a malicious script is reflected off of a web application to the victim's browser.the payload is passed through the url.  In this vulnerability on the attacker side,the attacker can able to balancing the payload and he can able to send the arbitrary code in the url field | |  |
| **How It Was Discovered** | |  |
| Manual Analysis | |  |
| **Vulnerable URLs** | |  |
| <https://www.bugbountyhunter.org/internship_labs/HTML/xss_lab/lab_2/lab_2.php> | |  |
| **Consequences of not Fixing the Issue** | |  |
| Perform any action within the application that the user can perform. View any information that the user is able to view. Modify any information that the user is able to modify.  the attacker can able to steal cookies. | |  |
| **Suggested Countermeasures** | |  |
| Use appropriate response headers  Use Content Security Policy | |  |
| **References** | |  |
| <https://portswigger.net/web-security/cross-site-scripting> | |  |

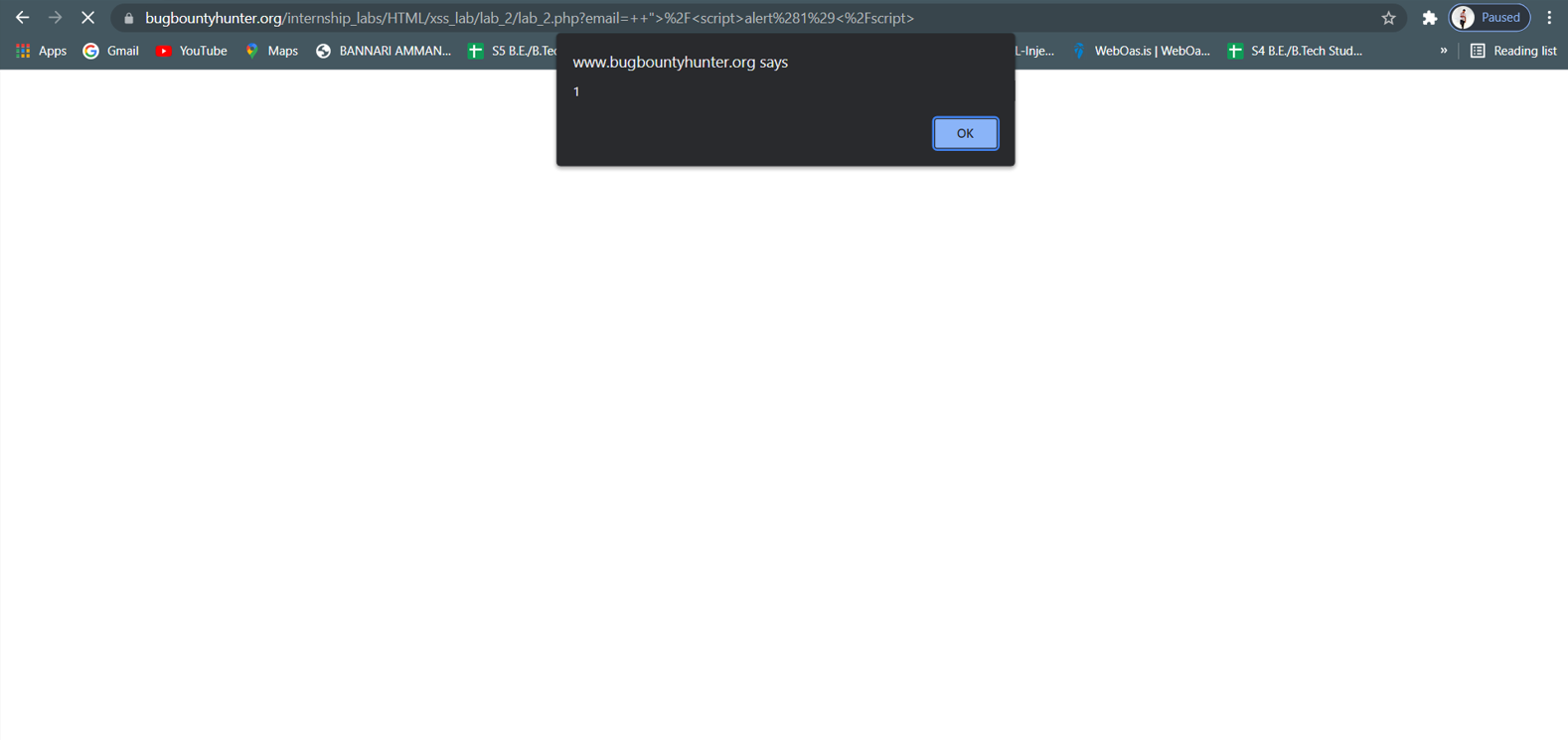
# 

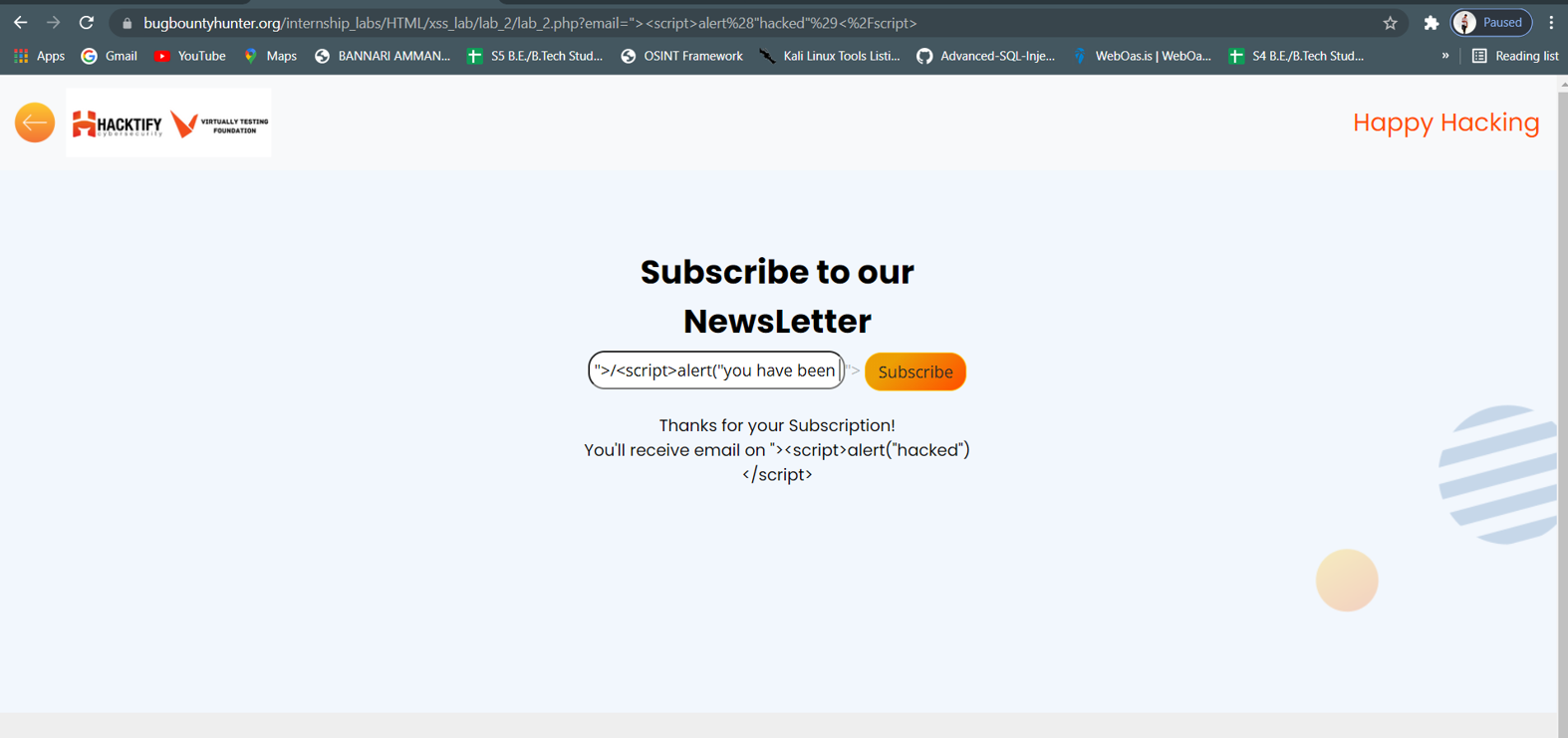
# Proof of Concept

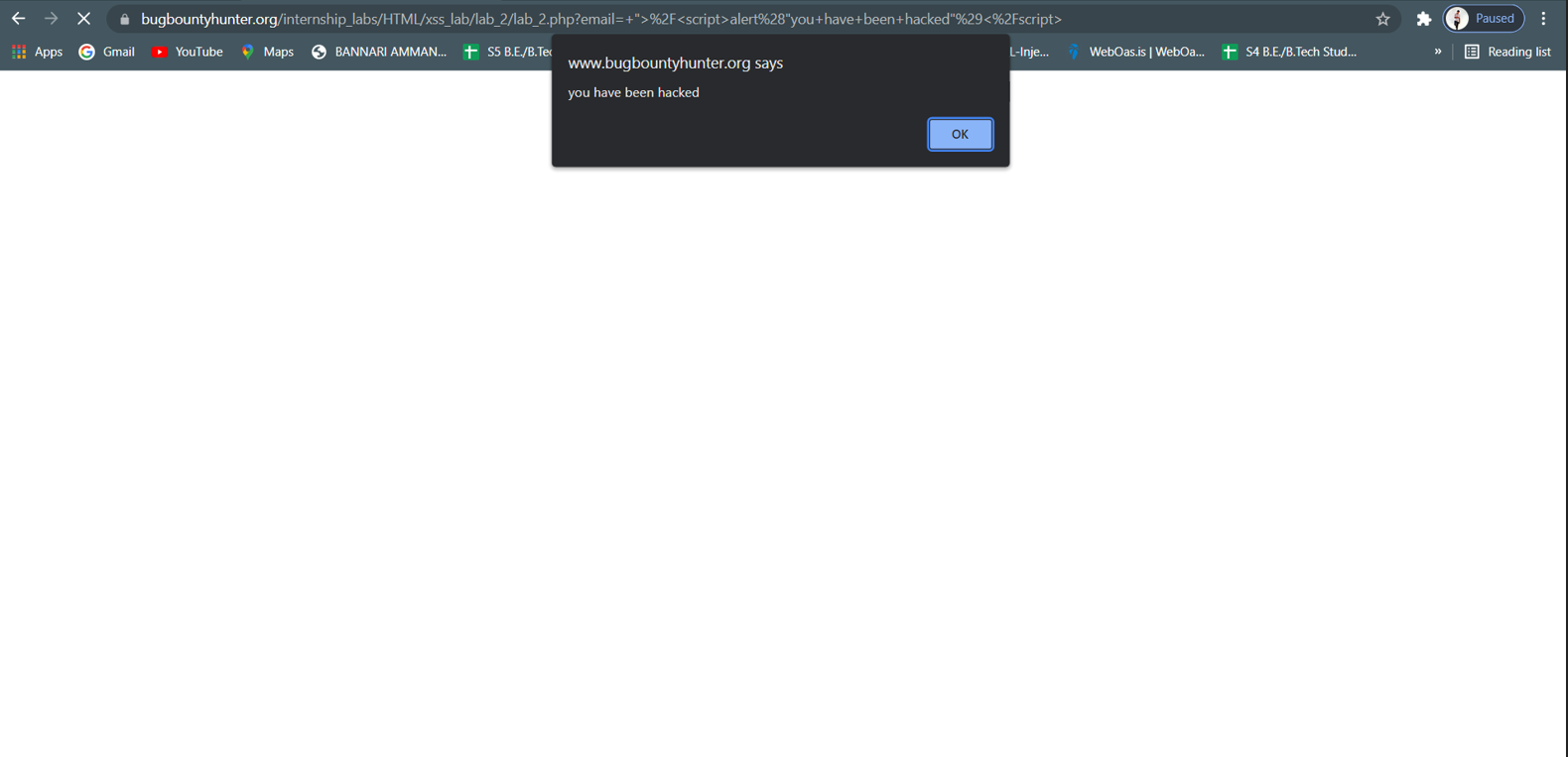
payload:">/<script>alert(1)</script>

pyload:">/<script>alert("you have been hacked")









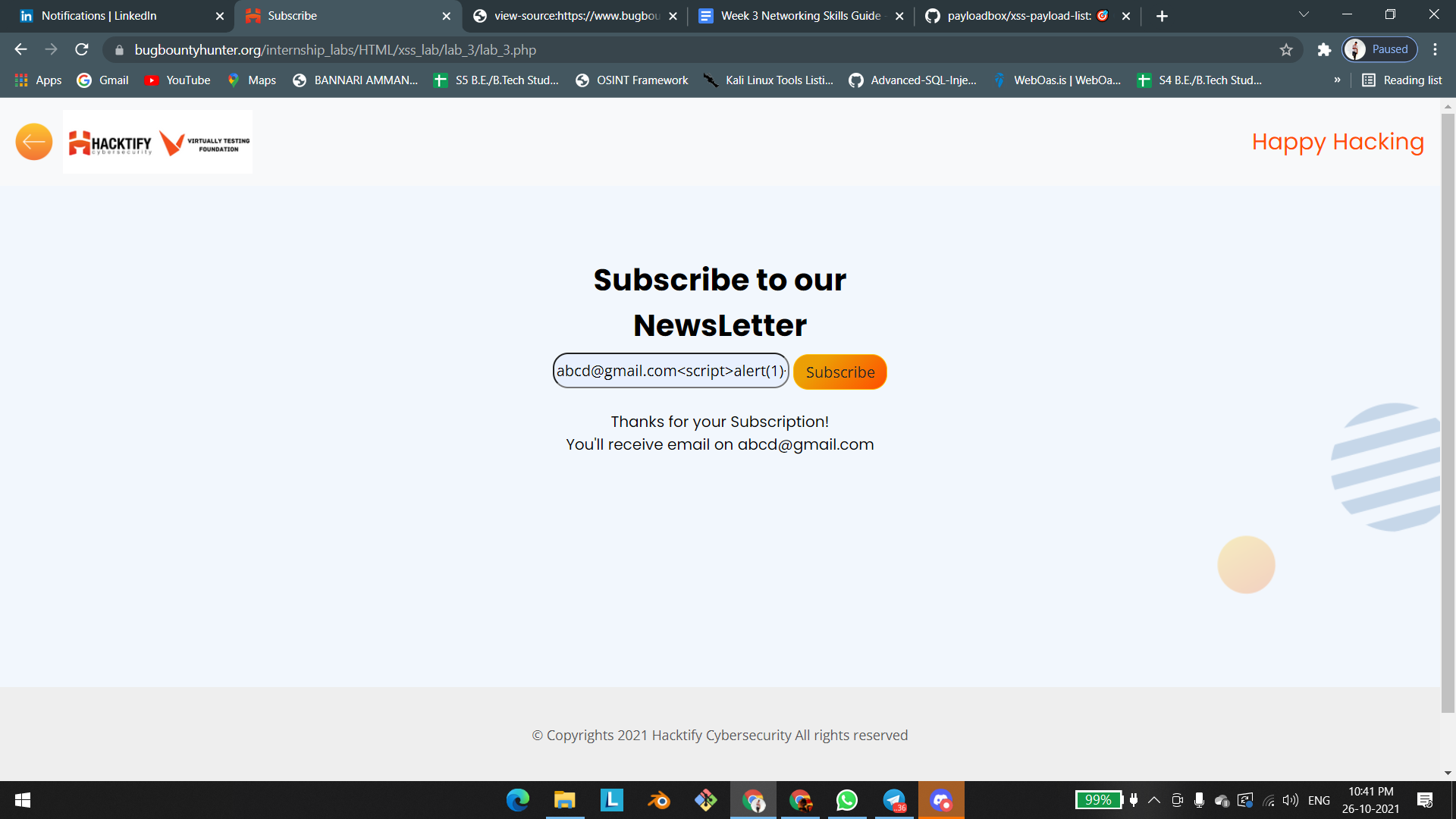
# 1.3. XSS Is Everywhere!

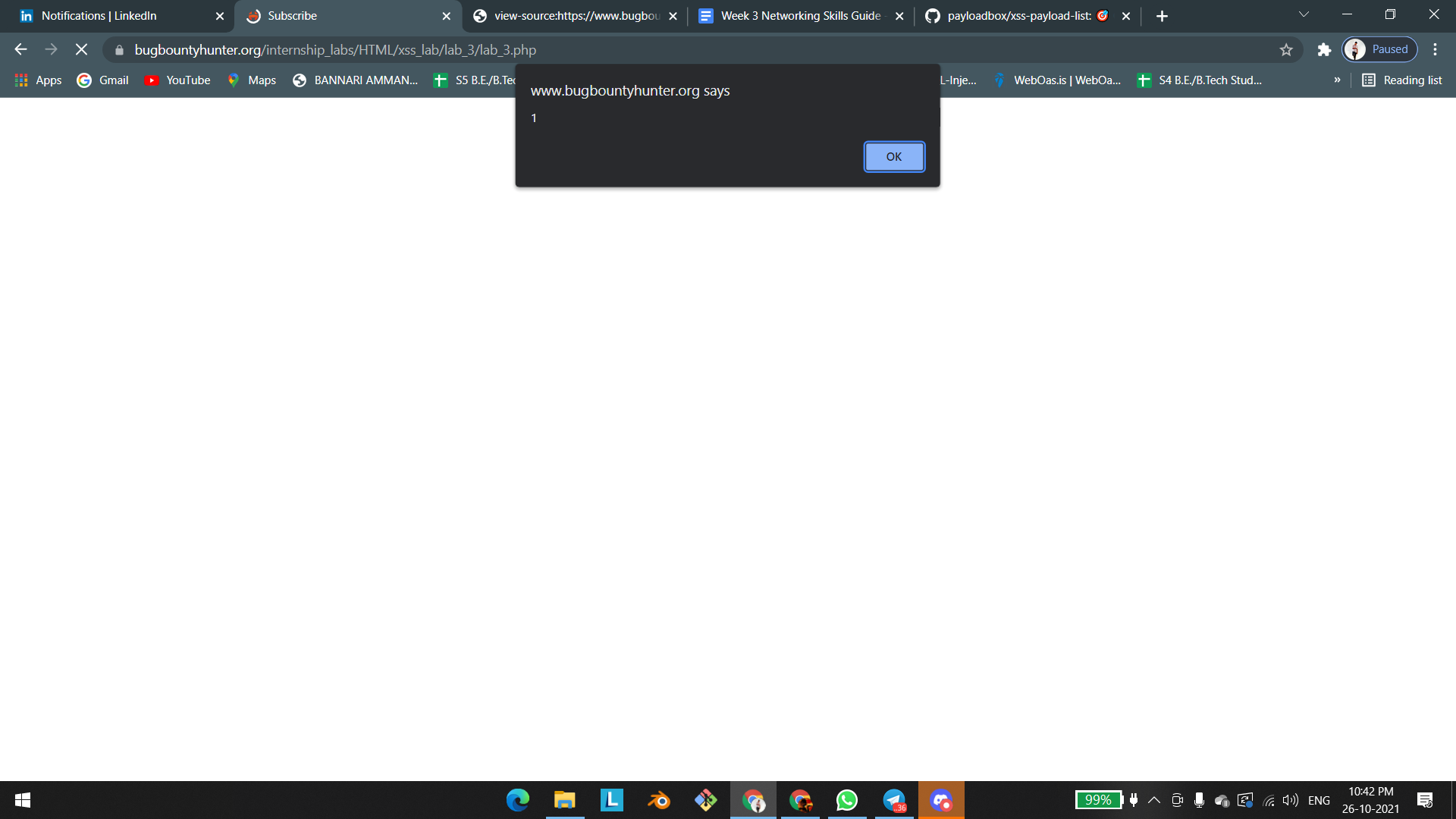
| **Reference** | **Risk Rating** |  |
| --- | --- | --- |
| XSS Is Everywhere! | **Low** |  |
| **Tools Used** | |  |
| by Manual analysis | |  |
| **Vulnerability Description** | |  |
| This reflected XSS would execute after making a POST request with an XSS payload in the path of the request. As a result, the server would directly insert the payload into the response, allowing the XSS to trigger on the page | |  |
| **How It Was Discovered** | |  |
| Manual Analysis | |  |
| **Vulnerable URLs** | |  |
| <https://www.bugbountyhunter.org/internship_labs/HTML/xss_lab/lab_3/lab_3.php> | |  |
| **Consequences of not Fixing the Issue** | |  |
| on the victim side each visit malicious script is activated.  To distribute the malicious link, a perpetrator typically embeds it into an email or third party website. The link is embedded inside an anchor text that provokes the user to click on it, which initiates the XSS request to an exploited website, reflecting the attack back to the user.  this may lead to account takeover .  I can execute JS code on the websites's users | |  |
| **Suggested Countermeasures** | |  |
| don’t allow to execute the javascript coming from the HTTP request as a dynamic content. | |  |
| **References** | |  |
| <https://portswigger.net/blog/exploiting-xss-in-post-requests> | |  |

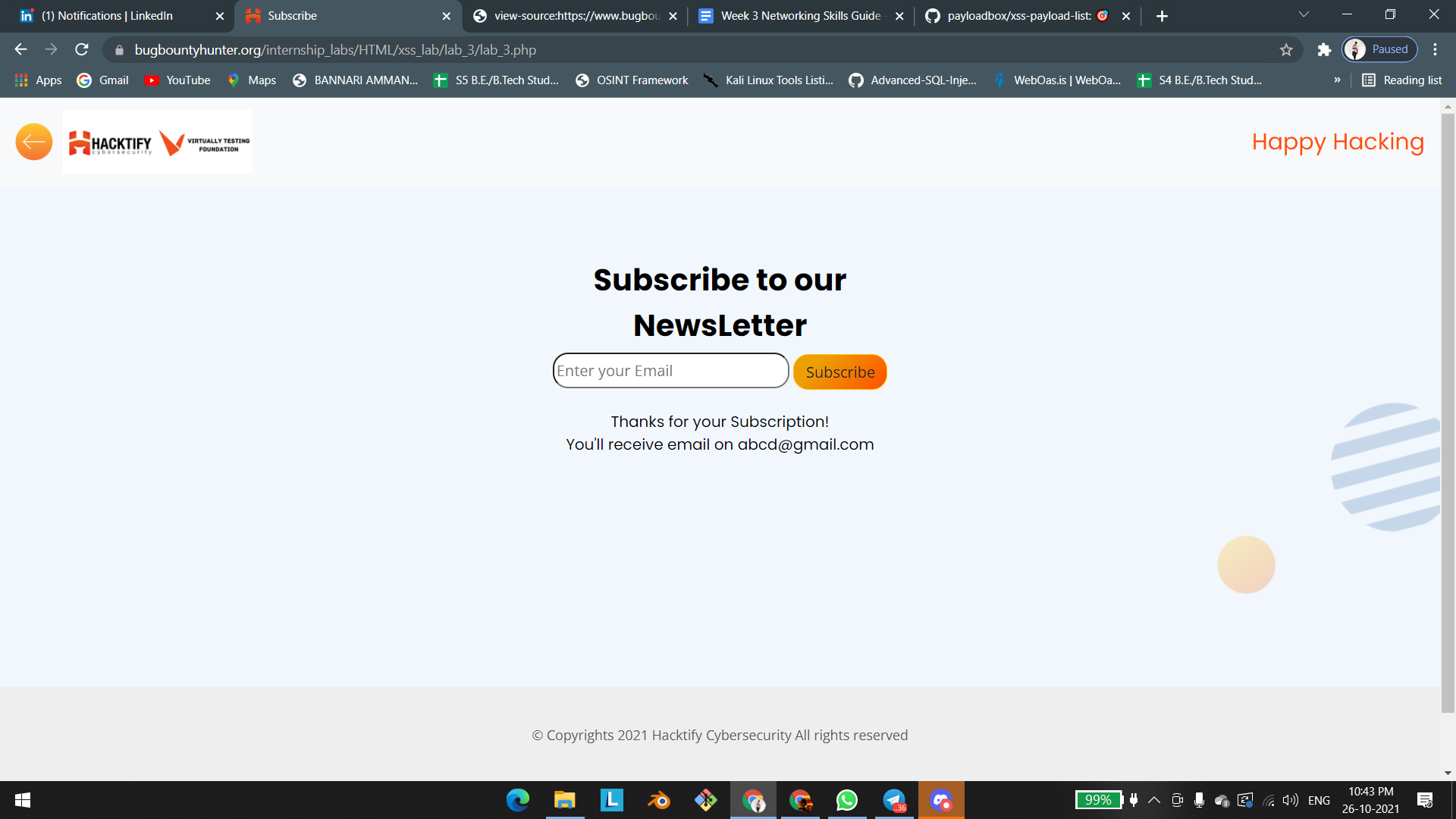
# 

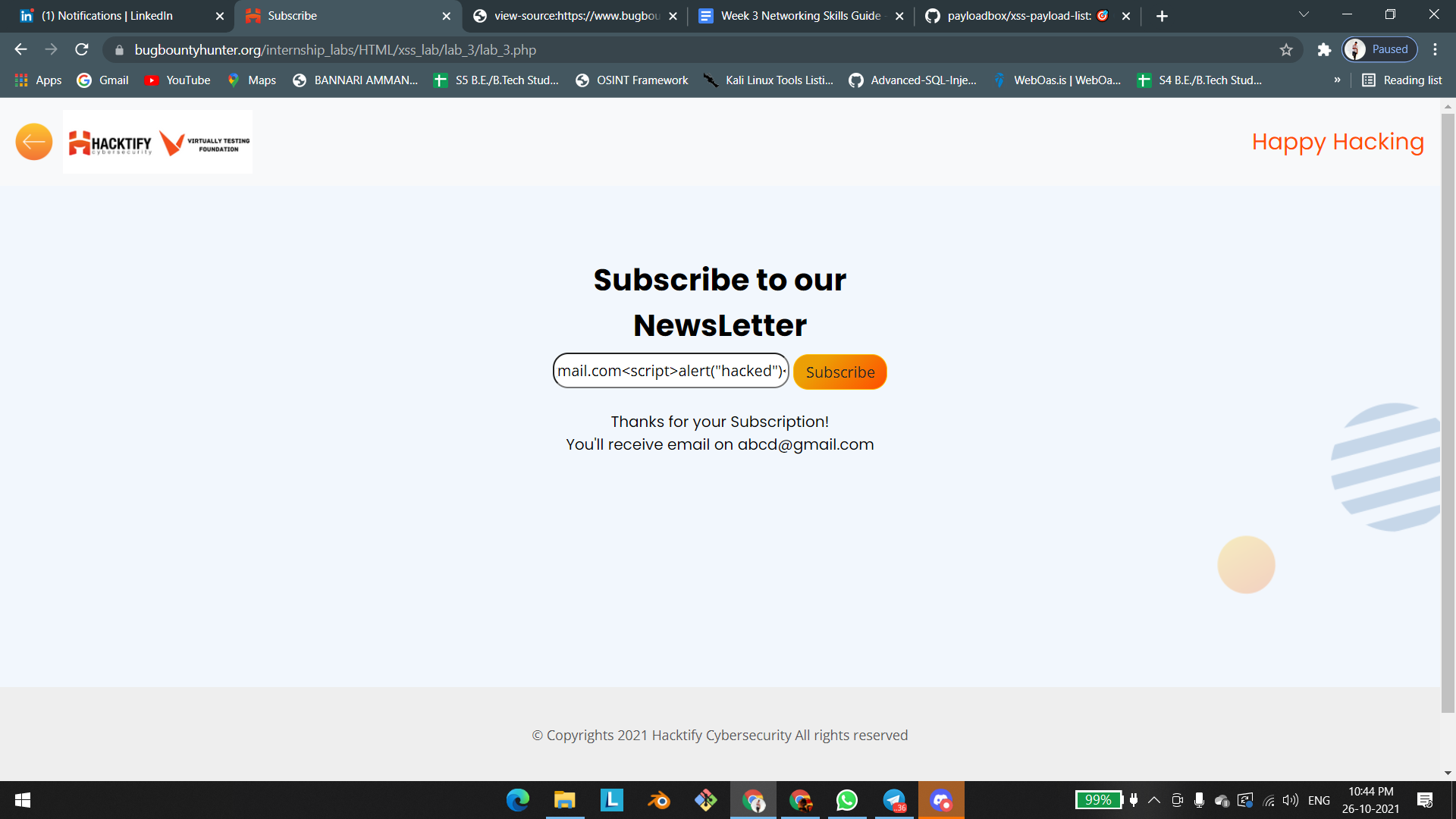
# Proof of Concept

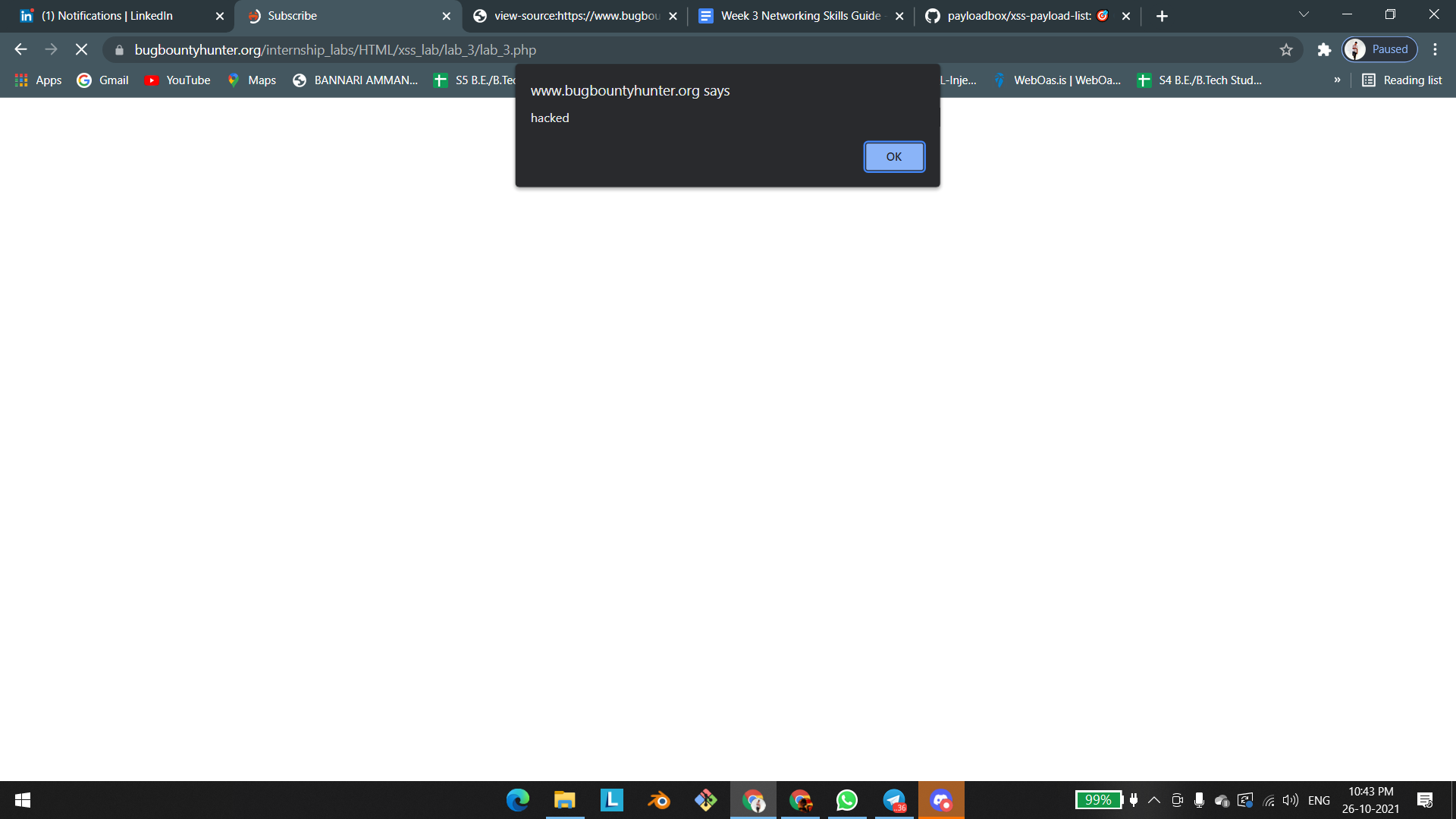
payload:abcd@gmail.com<script>alert(1)</script>











# 1.4. Alternatives Are Must!

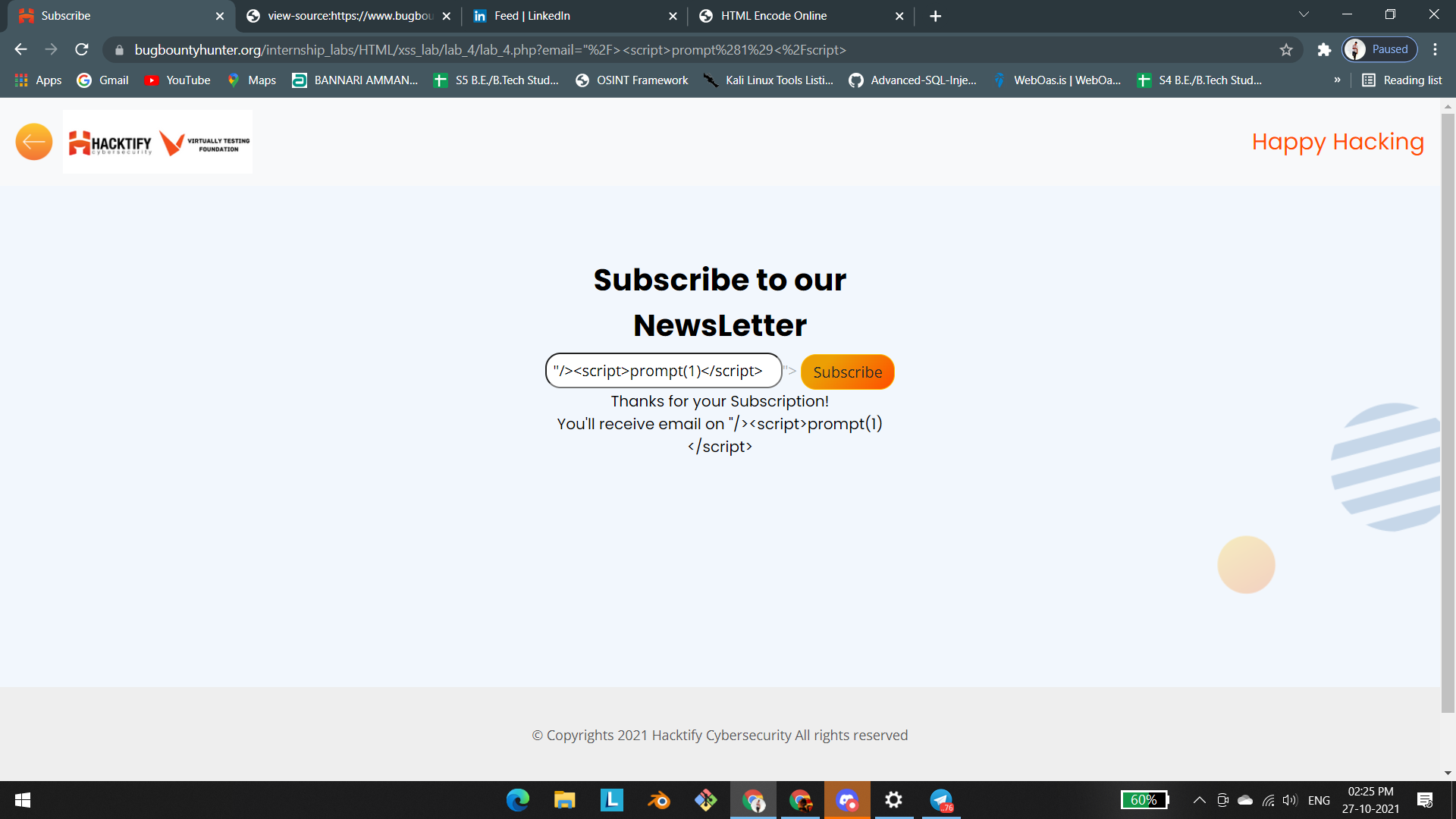
| **Reference** | **Risk Rating** |  |
| --- | --- | --- |
| Alternatives Are Must! | **Medium** |  |
| **Tools Used** | |  |
| BY manual analysis | |  |
| **Vulnerability Description** | |  |
| Reflected xss  sanitization of alert script.  Reflected XSS attacks, also known as non-persistent attacks, occur when a malicious script is reflected off of a web application to the victim's browser. | |  |
| **How It Was Discovered** | |  |
| Manual Analysis | |  |
| **Vulnerable URLs** | |  |
| <https://www.bugbountyhunter.org/internship_labs/HTML/xss_lab/lab_4/lab_4.php> | |  |
| **Consequences of not Fixing the Issue** | |  |
| the attacker can able to play a game through prompt script he/she asking the user name and password through prompt script when the victim enter his account details means it may lead to account takeover. | |  |
| **Suggested Countermeasures** | |  |
| sanitize the all type of javascript through user input. | |  |
| **References** | |  |
| <https://www.imperva.com/learn/application-security/reflected-xss-attacks/> | |  |

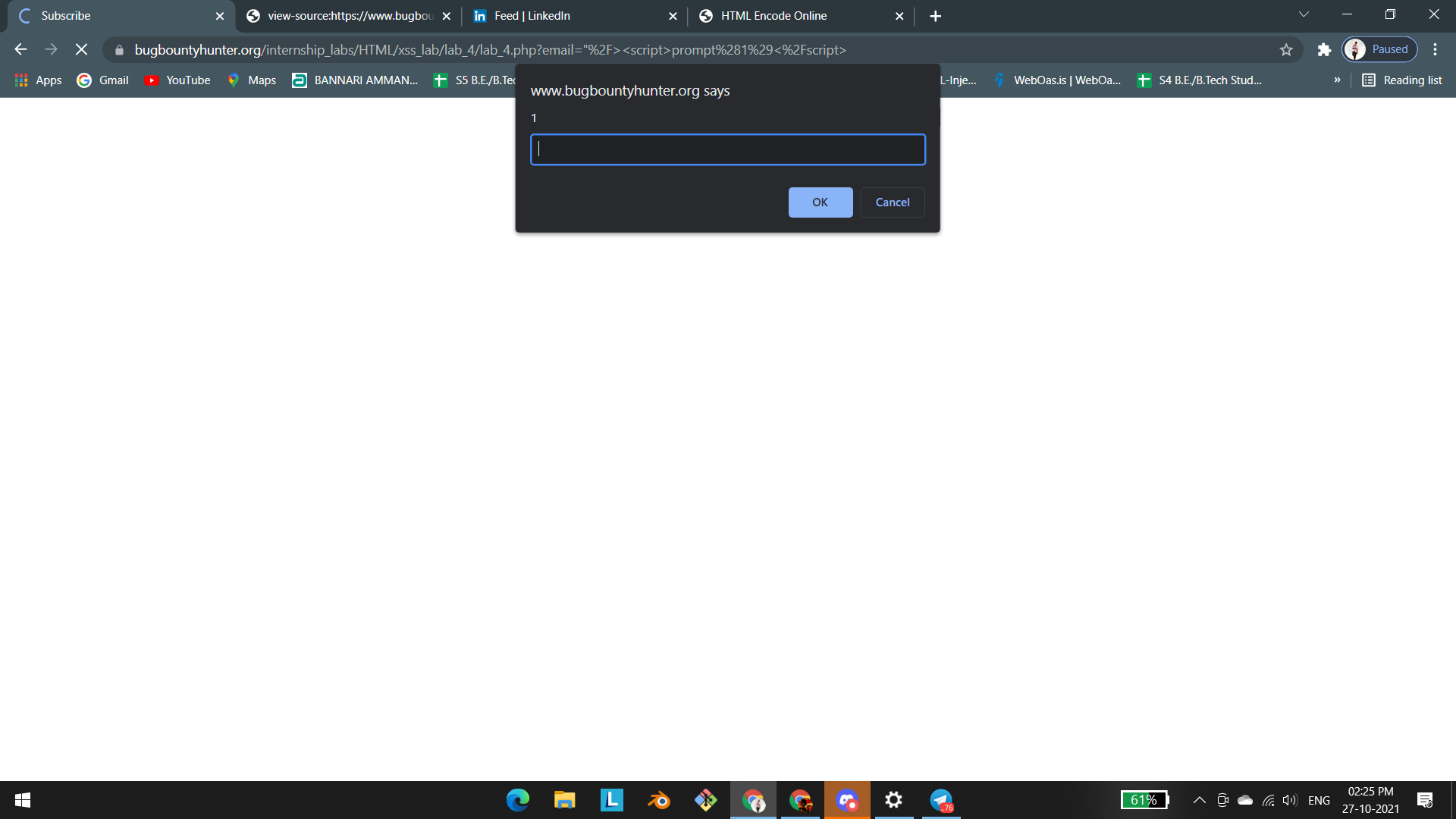
# 

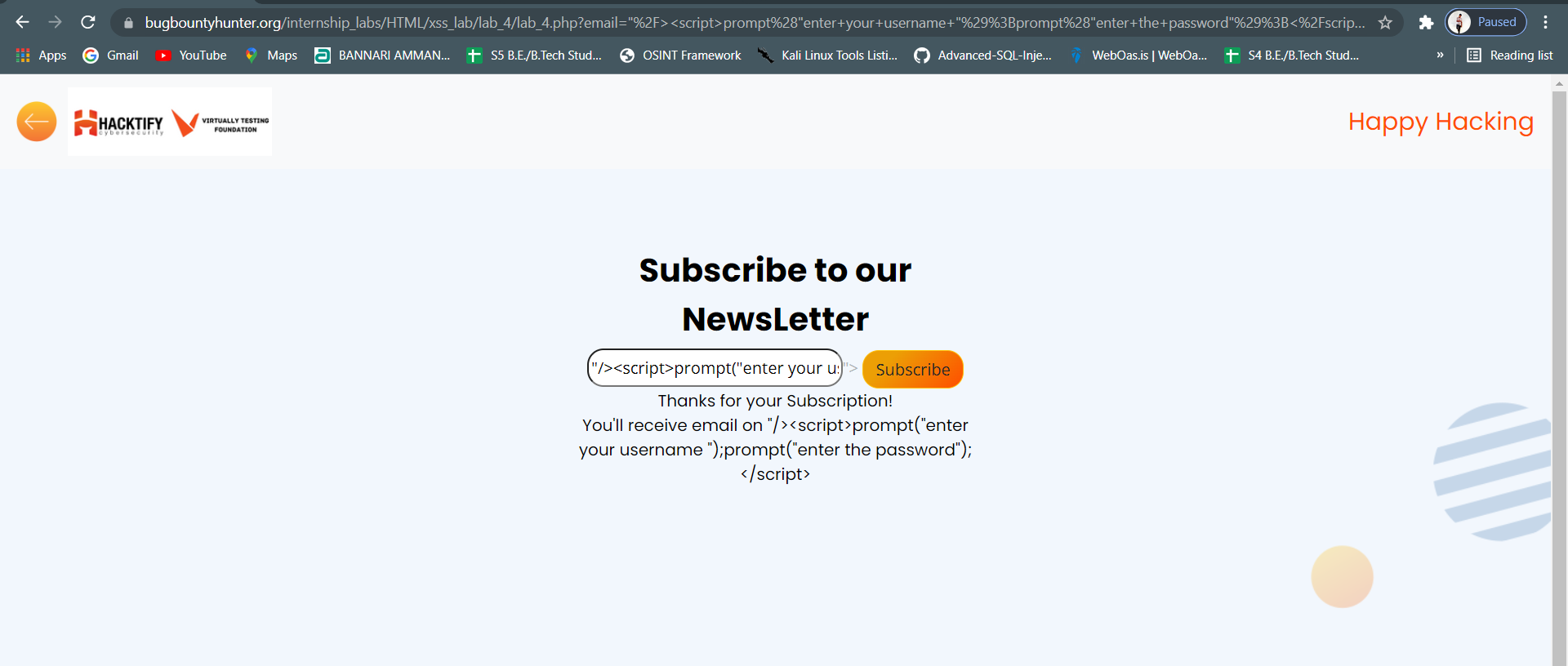
# Proof of Concept

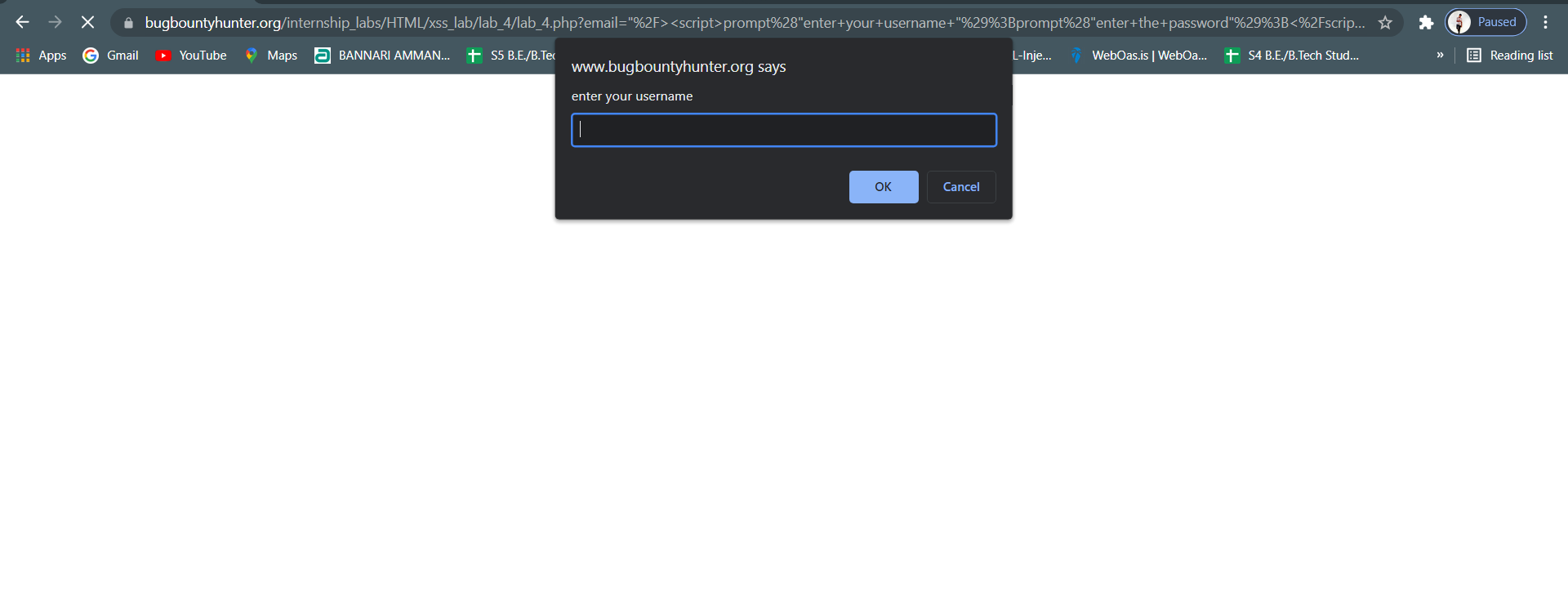
payload:"/><script>prompt(1)</script>

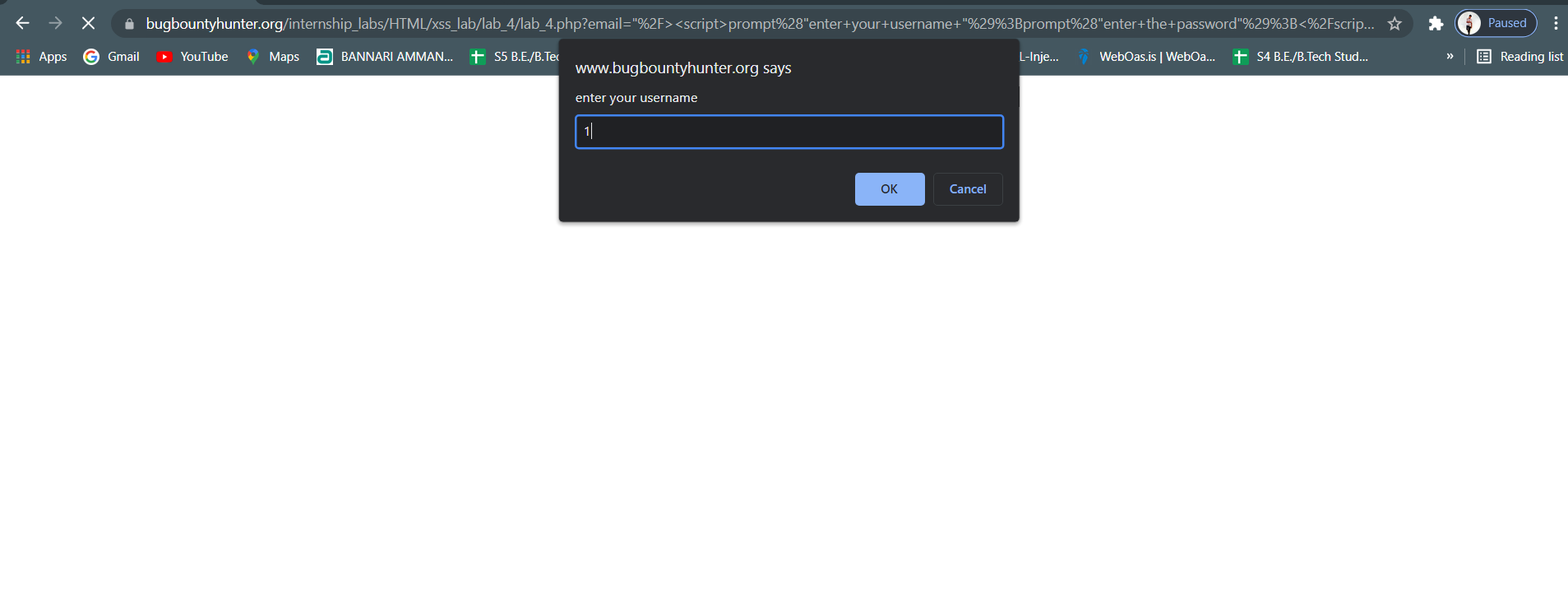
payload:"/><script>prompt("enter your username ");prompt("enter the password");</script>

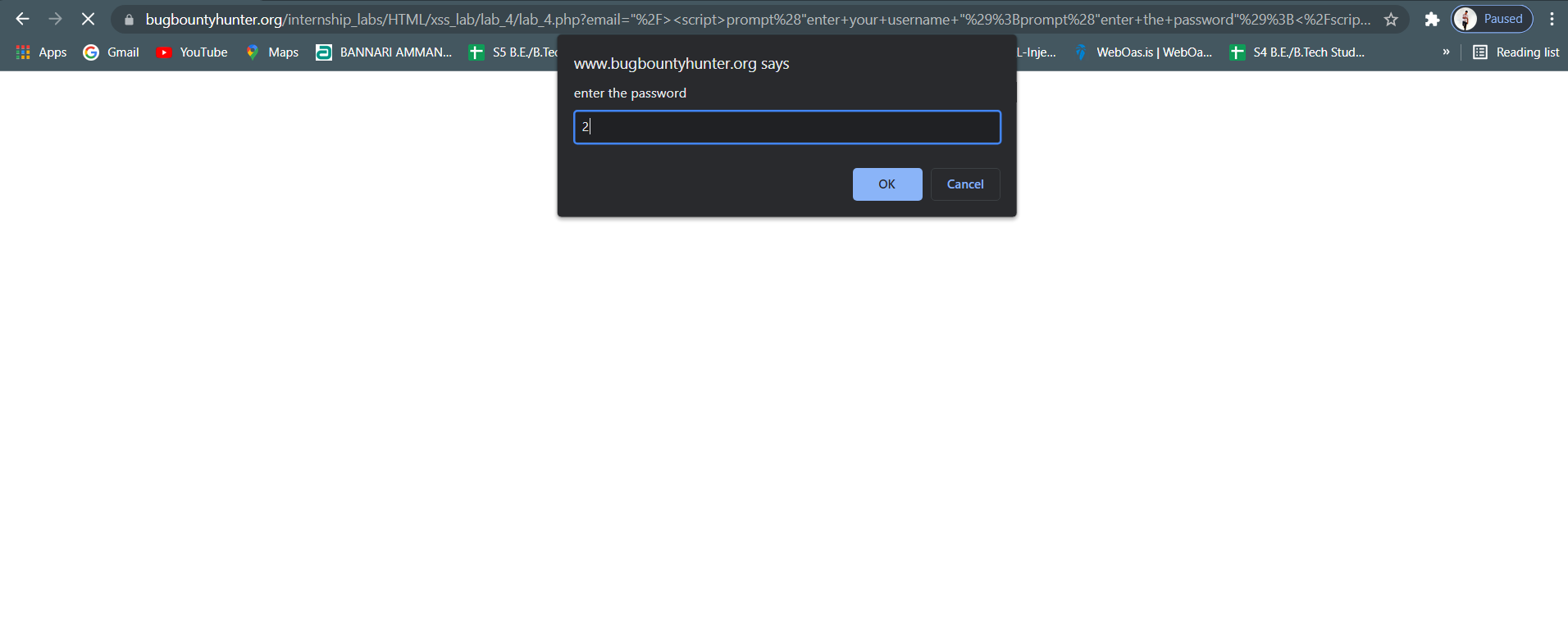












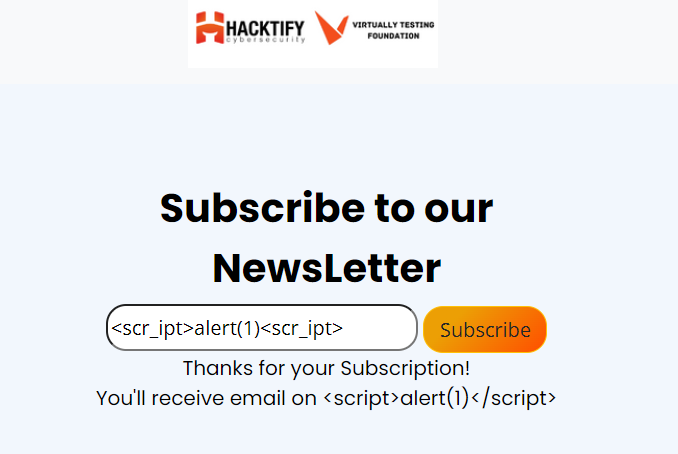
# 1.5. Developer Hates Scripts!

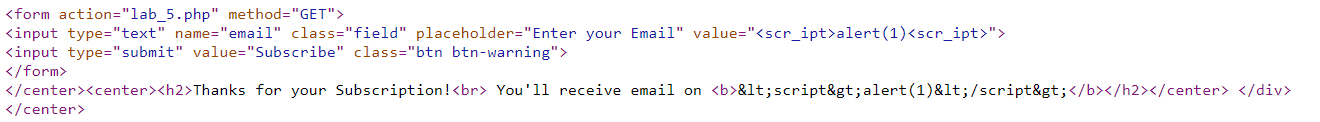
| **Reference** | **Risk Rating** |  |
| --- | --- | --- |
| Developer Hates Scripts! | **Medium** |  |
| **Tools Used** | |  |
| By manual analysis | |  |
| **Vulnerability Description** | |  |
| Reflected xss  sanitization of script tag | |  |
| **How It Was Discovered** | |  |
| Manual Analysis | |  |
| **Vulnerable URLs** | |  |
| <https://www.bugbountyhunter.org/internship_labs/HTML/xss_lab/lab_5/lab_5.php> | |  |
| **Consequences of not Fixing the Issue** | |  |
| attacker can able to steal user credential and cookie it may lead to account takeover | |  |
| **Suggested Countermeasures** | |  |
| sanitize the javascript and html tags in user input field and don’t allow to execute those type of scripts and tag | |  |
| **References** | |  |
| <https://www.imperva.com/learn/application-security/reflected-xss-attacks/>  <https://www.rapid7.com/resources/xss-vulnerabilities-explained/> | |  |

# 

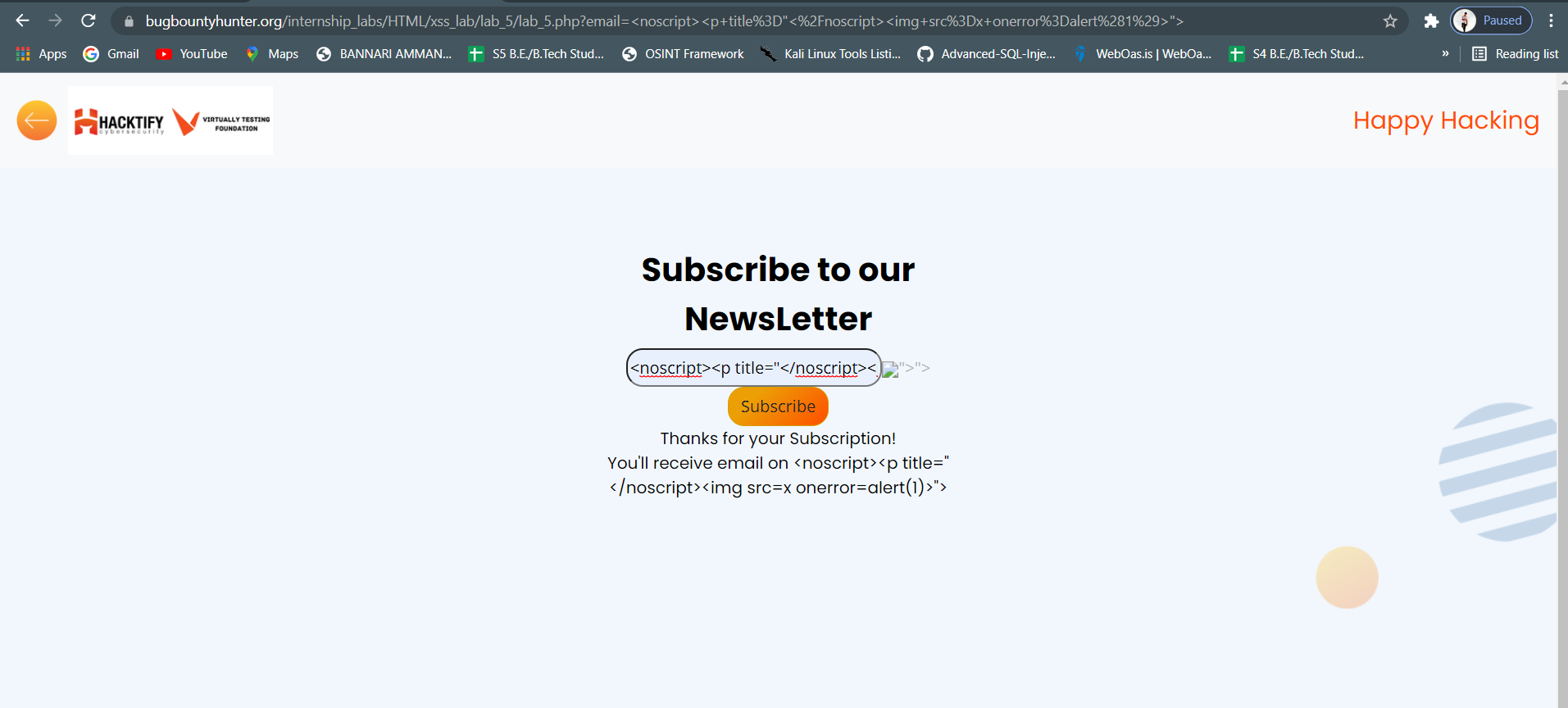
# Proof of Concept

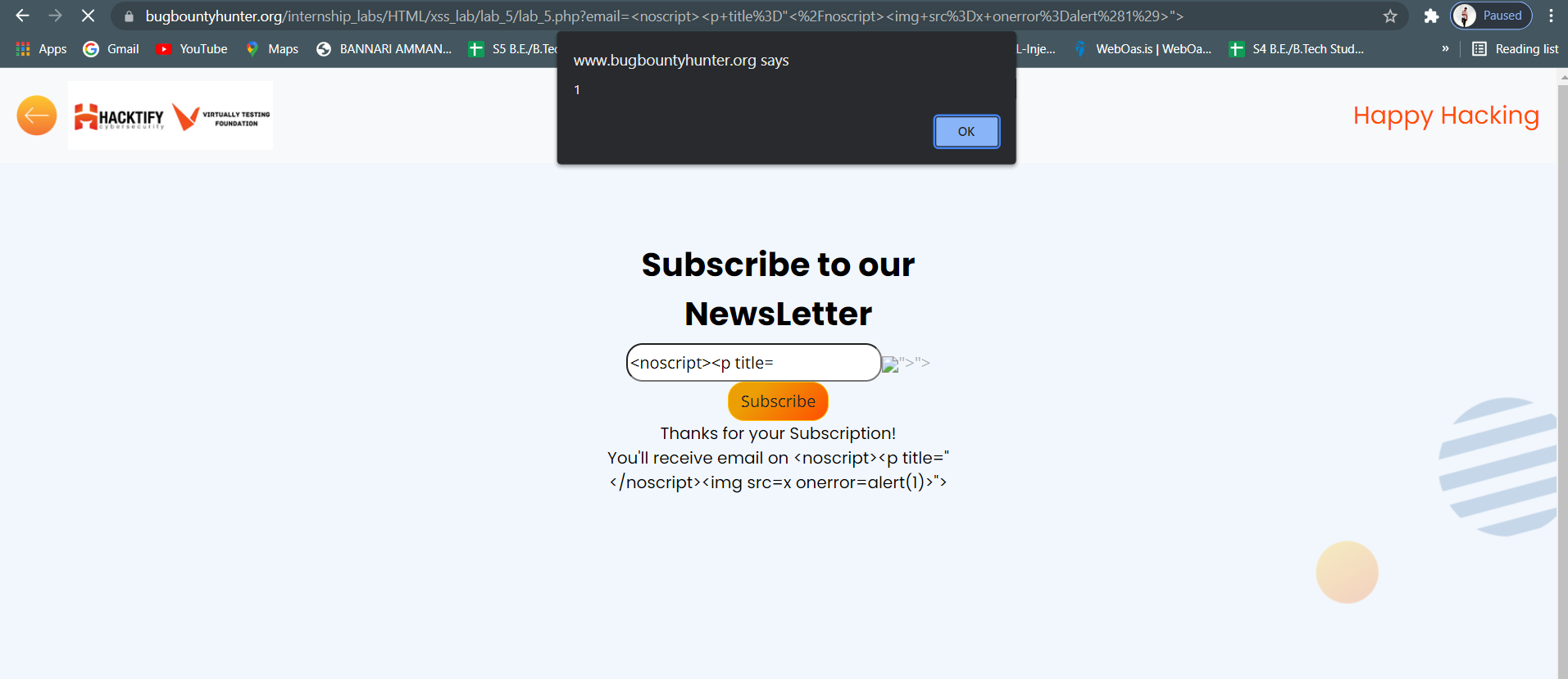
script tag gets sanitized so we are using noscript tag





payload: <noscript><p title="</noscript><img src=x onerror=alert(1)>">





# 1.6. Change The Variation!

| **Reference** | **Risk Rating** |  |
| --- | --- | --- |
| Change The Variation! | **Low / Medium / High** |  |
| **Tools Used** | |  |
| Tools that you have used to find the vulnerability. | |  |
| **Vulnerability Description** | |  |
| About the vulnerability and its working | |  |
| **How It Was Discovered** | |  |
| Automated Tools / Manual Analysis | |  |
| **Vulnerable URLs** | |  |
| URLs of the vulnerable pages in the lab | |  |
| **Consequences of not Fixing the Issue** | |  |
| What will be the consequences if the vulnerability is not patched? | |  |
| **Suggested Countermeasures** | |  |
| Give some Suggestions to stand against this vulnerability | |  |
| **References** | |  |
| URLs to the sources used to know more about this vulnerability | |  |

# 

# Proof of Concept

# 1.7. Encoding Is The Key?

| **Reference** | **Risk Rating** |  |
| --- | --- | --- |
| Encoding Is The Key? | **Low / Medium / High** |  |
| **Tools Used** | |  |
| Tools that you have used to find the vulnerability. | |  |
| **Vulnerability Description** | |  |
| About the vulnerability and its working | |  |
| **How It Was Discovered** | |  |
| Automated Tools / Manual Analysis | |  |
| **Vulnerable URLs** | |  |
| URLs of the vulnerable pages in the lab | |  |
| **Consequences of not Fixing the Issue** | |  |
| What will be the consequences if the vulnerability is not patched? | |  |
| **Suggested Countermeasures** | |  |
| Give some Suggestions to stand against this vulnerability | |  |
| **References** | |  |
| URLs to the sources used to know more about this vulnerability | |  |

# 

# Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

# 1.8.XSS With File Upload (File Name)

| **Reference** | **Risk Rating** |  |
| --- | --- | --- |
| {Sub-lab-1 Name} | **Low / Medium / High** |  |
| **Tools Used** | |  |
| Tools that you have used to find the vulnerability. | |  |
| **Vulnerability Description** | |  |
| About the vulnerability and its working | |  |
| **How It Was Discovered** | |  |
| Automated Tools / Manual Analysis | |  |
| **Vulnerable URLs** | |  |
| URLs of the vulnerable pages in the lab | |  |
| **Consequences of not Fixing the Issue** | |  |
| What will be the consequences if the vulnerability is not patched? | |  |
| **Suggested Countermeasures** | |  |
| Give some Suggestions to stand against this vulnerability | |  |
| **References** | |  |
| URLs to the sources used to know more about this vulnerability | |  |

# 

# Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

# 1.9.XSS With File Upload (File Content)

| **Reference** | **Risk Rating** |  |
| --- | --- | --- |
| {Sub-lab-1 Name} | **Low / Medium / High** |  |
| **Tools Used** | |  |
| Tools that you have used to find the vulnerability. | |  |
| **Vulnerability Description** | |  |
| About the vulnerability and its working | |  |
| **How It Was Discovered** | |  |
| Automated Tools / Manual Analysis | |  |
| **Vulnerable URLs** | |  |
| URLs of the vulnerable pages in the lab | |  |
| **Consequences of not Fixing the Issue** | |  |
| What will be the consequences if the vulnerability is not patched? | |  |
| **Suggested Countermeasures** | |  |
| Give some Suggestions to stand against this vulnerability | |  |
| **References** | |  |
| URLs to the sources used to know more about this vulnerability | |  |

# 

# Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

# 1.10.Stored Everywhere!

| **Reference** | **Risk Rating** |  |
| --- | --- | --- |
| {Sub-lab-1 Name} | **Low / Medium / High** |  |
| **Tools Used** | |  |
| Tools that you have used to find the vulnerability. | |  |
| **Vulnerability Description** | |  |
| About the vulnerability and its working | |  |
| **How It Was Discovered** | |  |
| Automated Tools / Manual Analysis | |  |
| **Vulnerable URLs** | |  |
| URLs of the vulnerable pages in the lab | |  |
| **Consequences of not Fixing the Issue** | |  |
| What will be the consequences if the vulnerability is not patched? | |  |
| **Suggested Countermeasures** | |  |
| Give some Suggestions to stand against this vulnerability | |  |
| **References** | |  |
| URLs to the sources used to know more about this vulnerability | |  |

# 

# Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

# 1.11.DOM's Are Love!

| **Reference** | **Risk Rating** |  |
| --- | --- | --- |
| {Sub-lab-1 Name} | **Low / Medium / High** |  |
| **Tools Used** | |  |
| Tools that you have used to find the vulnerability. | |  |
| **Vulnerability Description** | |  |
| About the vulnerability and its working | |  |
| **How It Was Discovered** | |  |
| Automated Tools / Manual Analysis | |  |
| **Vulnerable URLs** | |  |
| URLs of the vulnerable pages in the lab | |  |
| **Consequences of not Fixing the Issue** | |  |
| What will be the consequences if the vulnerability is not patched? | |  |
| **Suggested Countermeasures** | |  |
| Give some Suggestions to stand against this vulnerability | |  |
| **References** | |  |
| URLs to the sources used to know more about this vulnerability | |  |

# 

# Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab