Title: Vulnerability Assessment Report on VulnWeb XSS

Date: [07/08/2023]

Prepared by: [Rahul Kumar Shrivastav]

Project: VulnWeb XSS Vulnerability Assessment

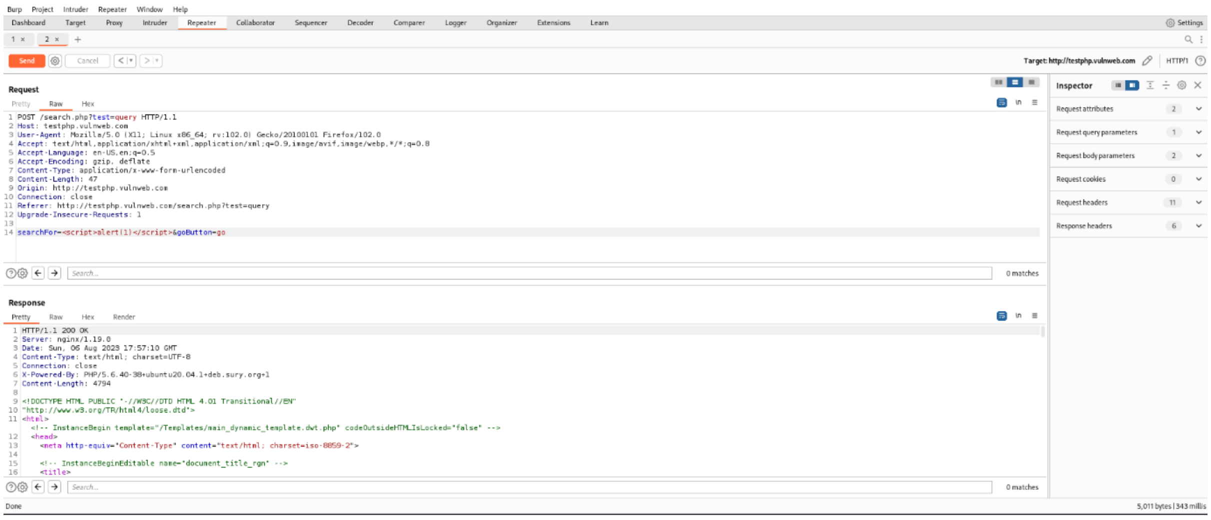
1. Executive Summary:

The following report presents the findings of a vulnerability assessment conducted on VulnWeb with a focus on Cross-Site Scripting (XSS) vulnerabilities. The assessment aimed to identify and evaluate potential security weaknesses related to XSS attacks within the web application. The assessment involved comprehensive testing methodologies and tools to ensure a thorough analysis.

Issue detail The name of an arbitrarily supplied URL parameter is copied into the HTML document as plain text between tags. The payload <script>alert(1) </script>as submitted in the name of an arbitrarily supplied URL parameter. This input was echoed unmodified in the application's response. This behavior demonstrates that it is possible to inject new HTML tags into the returned document. An attempt was made to identify a full proof-of-concept attack for injecting arbitrary JavaScript but this was not successful. You should manually examine the application's behavior and attempt to identify any unusual input validation or other obstacles that may be in place.

The attacker-supplied code can perform a wide variety of actions, such as stealing the victim's session token or login credentials, performing arbitrary actions on the victim's behalf, and logging their keystrokes.

Request:-



```
POST /search.php?test=query HTTP/1.1
Host: testphp.vulnweb.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 47
Origin: http://testphp.vulnweb.com
Connection: close
Referer: http://testphp.vulnweb.com/search.php?test=query
Upgrade-Insecure-Requests: 1

searchFor=<script>alert(1)</script>&goButton=go
```

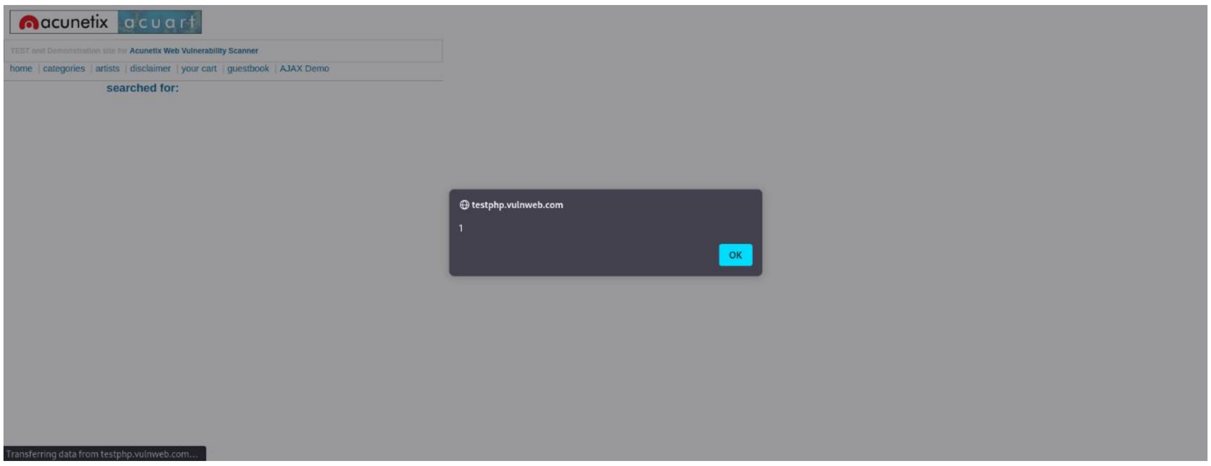
```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Sun, 06 Aug 2023 17:57:10 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
Content-Length: 4794

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
  <!-- InstanceBegin template="/Templates/main_dynamic_template.dwt.php" codeOutsideHTMLIsLocked="false" -->
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-2">

    <!-- InstanceBeginEditable name="document_title_rgn" -->
    <title>
```

Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Send
Cancel
Target: http://testphp.vulnweb.com
Request
Response
Pretty Raw Hex Render
Inspector
Request attributes 2
Request query parameters 1
Request body parameters 2
Request cookies 0
Request headers 11
Response headers 6
0 matches
Done
5,011 bytes | 343 millis

Response:-


acunetix acuart
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo
searched for:
testphp.vulnweb.com
1
OK
Transferring data from testphp.vulnweb.com...

Remediation Recommendations: Provided detailed recommendations for mitigating the identified vulnerabilities.

In most situations where user-controllable data is copied into application responses, cross-site scripting attacks can be prevented using two layers of defenses:

- Input should be validated as strictly as possible on arrival, given the kind of content that it is expected to contain. For example, personal names should consist of alphabetical and a small range of typographical characters, and be relatively short; a year of birth should consist of exactly four numerals; email addresses should match a well-defined regular expression. Input which fails the validation should be rejected, not sanitized.

● User input should be HTML-encoded at any point where it is copied into application responses. All HTML metacharacters, including < > " ' and =, should be replaced with the corresponding HTML entities (< > etc). In cases where the application's functionality allows users to author content using a restricted subset of HTML tags and attributes (for example, blog comments which allow limited formatting and linking), it is necessary to parse the supplied HTML to validate that it does not use any dangerous syntax; this is a non-trivial task

Critical: one critical XSS vulnerabilities were identified. These vulnerabilities pose a significant risk to the application's security, potentially allowing an attacker to execute malicious scripts within a user's browser.

High: one high-severity XSS vulnerabilities were detected. These vulnerabilities could lead to unauthorized data exposure or manipulation.

Medium: one medium-severity XSS vulnerabilities were found. While these vulnerabilities may not have an immediate critical impact, they could still be exploited to compromise user data or perform other malicious actions.

Low: one low-severity XSS vulnerabilities were discovered. These vulnerabilities have a lower potential for harm but should still be addressed to ensure a more robust security posture.

5. Recommendations:

Based on the findings, the following recommendations are suggested to mitigate the identified XSS vulnerabilities:

Input Validation and Sanitization: Implement strict input validation and sanitization mechanisms to prevent untrusted data from being executed as scripts.

Context-Aware Output Encoding: Encode output data based on the context it will be displayed in (HTML, JavaScript, etc.) to prevent script execution.

Content Security Policy (CSP): Implement a robust CSP to control the sources from which content can be loaded, reducing the risk of XSS attacks.

Regular Security Audits: Conduct regular security audits and vulnerability assessments to identify and address emerging threats and vulnerabilities.

Security Training: Provide training for developers to raise awareness of XSS vulnerabilities and best practices for secure coding.

Patching and Updates: Keep all software components and libraries up to date with the latest security patches.

6. Conclusion:

The vulnerability assessment on VulnWeb revealed the presence of XSS vulnerabilities of varying severity. Addressing these vulnerabilities is crucial to safeguarding the application and its users from potential attacks. By implementing the recommended remediation strategies, the application's security can be significantly improved, reducing the risk of exploitation.