Product: Suhail Sallahudin

Report: My New Report

PEN-DOC20240312210409

Shub, https//:www.shub_pentest.com

12-03-2024

Project Overview

Description

Hello, **THIS IS BOLD**, *THIS IS ITALIC*

H1 TAGS

H2 Tags

H3 Tags

> Quote anjir

- list unordered
- list unordered
- list unordered
- list unordered
-

1. Ordered List
2. Ordered List
3. Ordered List

Executive Summary

TBC

Summary of Findings Identified

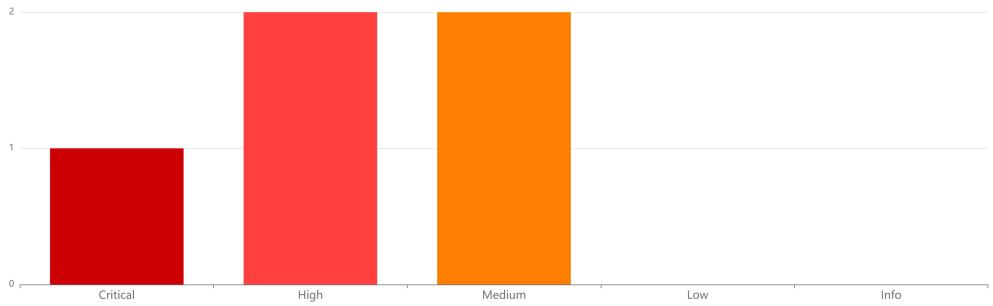

Figure 1: Executive Summary

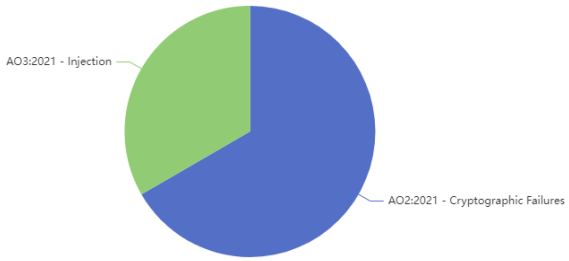

Figure 2: Breakdown by OWASP Categories

 # 1 Critical MD5 Usage

 # 2 High SQL Injection

 # 3 High Cross Site Scripting (XSS)

 # 4 Medium Missing 'HttpOnly' Cookie Attribute (HTTP)

 # 5 Medium Backup File Scanner (HTTP) - Reliable Detection Reporting

Scope

In Scope

TBC

Out of Scope

TBC

Methodology

TBC

Recommendations

TBC

Findings and Risk Analysis

MD5 Usage

Severity: Critical
CVSS Score: 9.2
CVSS Vector: CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:N/VA:N/SC:H/SI:N/SA:N

OWASP

2 - Cryptographic Failures

Description

One of the critical vulnerabilities discovered during our penetration test is related to Cryptographic Failures, listed as A02:2021 in the OWASP Top Ten 2021. This is classified as a serious flaw where the application does not correctly implement cryptographic processes. This may stem from various factors including usage of outdated cryptographic algorithms, hard-coded keys, or improper storage of cryptographic keys.

Location

www.exmple.com

Proof of Concept

TBC

Recommendation

- Always use up to date cryptographic algorithms that have been extensively vetted by the security community.
- Avoid hard coding encryption keys in the source code and use secure server-side storage for encryption keys.
- Regularly conduct automated and manual code review processes to identify any improper usage of cryptographic functions.
- Develop a patch and update management program to keep all software and systems up to date.

References

https://owasp.org/www-project-top-ten/2021/A02_2021-Cryptographic_Failures

SQL Injection

Severity: High
CVSS Score: 8.6
CVSS Vector: CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:N/VI:N/VA:N/SC:N/SI:N/SA:N

OWASP

2 - Cryptographic Failures

Description

SQL injection occurs when an attacker manipulates SQL queries by introducing malicious code through string concatenation. This typically happens when an application directly combines user input with SQL statements without proper validation or sanitization. Here's how SQL injection is introduced through string concatenation:

1. User Input: A web application takes user input, which could be provided through forms, query parameters, or other input fields. This user input is usually in the form of strings.
2. SQL Query Building: The application constructs SQL queries by directly concatenating the user input with SQL statements. For example, consider the following pseudocode:

   ```
   username = user_input  # User-provided input
   sql_query = &quot;SELECT * FROM users WHERE username = &#x27;&quot; + username +
   ↪  &quot;&#x27;&quot;
   ```

   In this example, the user's input is directly included in the SQL query without any validation or sanitation.
3. Malicious Input: An attacker identifies the input field as vulnerable and provides malicious input that contains SQL code. This input is crafted to manipulate the SQL query's structure and behavior.
4. Injection: The attacker's input is inserted into the SQL query. For instance, if the attacker inputs `username&#x27; OR &#x27;1&#x27;=&#x27;1`, the resulting query becomes:

   ```
   SELECT * FROM users WHERE username = &#x27;username&#x27; OR &#x27;1&#x27;=&#x27;1&#x27;
   ```

   The injected code `&#x27;1&#x27;=&#x27;1&#x27;` is always true, effectively bypassing any username check and returning all records in the "users" table.
5. Query Execution: The modified SQL query is executed by the application and sent to the database server for processing. The database server interprets the query, including the injected code, leading to unintended behavior.

6. Results: The database server returns the results, which may include unauthorized access to data, data manipulation, or other actions as directed by the attacker's injected SQL code.

SQL injection through string concatenation is a common and severe security risk because it allows attackers to execute arbitrary SQL code within the application's database. To prevent this vulnerability, it's essential to use prepared statements or parameterized queries to ensure that user input is treated as data and not as executable SQL code. Additionally, input validation and strict data type checking can help protect against SQL injection by rejecting input that does not conform to expected formats.

Location

TBC

Proof of Concept

TBC

Recommendation

To mitigate this vulnerability, the following actions are recommended:

1. Implement prepared statements or parameterized queries in the application code to separate user input from SQL statements.
2. Perform input validation and filtering to ensure that user inputs conform to expected formats and data types.
3. Review and update the codebase to escape special characters in user input before incorporating them into SQL queries if prepared statements cannot be used.
4. Conduct a thorough security review of the application to identify and fix any existing SQL Injection vulnerabilities.

References

- http://www.securiteam.com/securityreviews/5DP0N1P76E.html
- http://en.wikipedia.org/wiki/SQL_injection
- https://www.owasp.org/index.php/SQL_Injection
- http://projects.webappsec.org/w/page/13246963/SQL%20Injection
- http://www.w3schools.com/sql/sql_injection.asp
- http://unixwiz.net/techtips/sql-injection.html

Cross Site Scripting (XSS)

Severity: High
CVSS Score: 7.8
CVSS Vector: CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:N/VI:N/VA:N/SC:L/SI:H/SA:N

OWASP

3 - Injection

Description

XSS DESC

Location

TBC

Proof of Concept

TBC

Recommendation

XSS DESC

References

XSS DESC XSS DESC XSS DESC XSS DESC XSS DESC XSS DESC

Missing 'HttpOnly' Cookie Attribute (HTTP)

Severity: Medium
CVSS Score: 5.0
CVSS Vector:

Description

The remote HTTP web server / application is missing to set the 'HttpOnly' cookie attribute for one or more sent HTTP cookie.

Location

10.10.124.125

Proof of Concept

The cookie(s):

Set-Cookie: PHPSESSID=***replaced***; path=/ Set-Cookie: PHPSESSID=***replaced***; path=/ Set-Cookie: security=low

is/are missing the "HttpOnly" cookie attribute.

Recommendation

- Set the 'HttpOnly' cookie attribute for any session cookie
  - Evaluate / do an own assessment of the security impact on the web server / application and create an override for this result if there is none (this can't be checked automatically by this VT)

References

- https://www.rfc-editor.org/rfc/rfc6265#section-5.2.6
- https://owasp.org/www-community/HttpOnly
- https://wiki.owasp.org/index.php/Testing_for_cookies_attributes_(OTG-SESS-002)

Backup File Scanner (HTTP) - Reliable Detection Reporting

Severity: Medium
CVSS Score: 5.0
CVSS Vector:

Description

The script reports backup files left on the web server.

Location

10.10.124.125

Proof of Concept

The following backup files were identified (<URL>:<Matching pattern>):

http://10.10.124.125/config/config.inc.php.bak:^<?(php|=)

Recommendation

Delete the backup files.

References

- http://www.openwall.com/lists/oss-security/2017/10/31/1

NMap Scan Data

Host: 10.10.243.102

- **Port 22** (closed): ssh unknown
  - No scripts found.
- **Port 80** (open): http Apache httpd
  - **http-title**: Site doesn't have a title (text/html).
  - **http-server-header**: Apache
- **Port 443** (open): http Apache httpd
  - **http-server-header**: Apache
  - **http-title**: Site doesn't have a title (text/html).
  - **ssl-cert**: Subject: commonName=www.example.com Not valid before: 2015-09-16T10:45:03 Not valid after: 2025-09-13T10:45:03