

ASSIGNMENT-I

SECURE SOFTWARE ENGINEERING

VULNERABILITIES

- 1.Cross-Site Scripting XSS (CWE-79) CyberChef
- 2.Cross-site Scripting XSS (CWE-79) LimeSurvey
- 3.Cross-site Request Forgery (CWE-352) LimeSurvey
- 4.Cross-site Request Forgery (CWE-352) Jenkins

Submitted by,
Kishaiyan Vellaichamy Thangaraj
A1819309

Part-1

Vulnerability-I

Name: Kishaiyan Vellaichamy Thangaraj

Student-Id: a1819309

Link to the file-

<https://github.com/gchq/CyberChef/blob/master/src/core/operations/SeriesChart.mjs>

Link to the commit

<https://github.com/gchq/CyberChef/commit/d2174725a95feaaa7f49140eaa64c94e6e3a3e09>

Name of the file- SeriesChart.mjs

The programming language used is JavaScript.

Name of the repository- CyberChef

Number of repository stars-22,200

Number of contributors in the repositories-118

Type of Vulnerability CWE-79

The Vulnerability exploited in the above-mentioned code is Reflected Cross-Site Scripting. The Code for color in the SeriesChart.mjs is not properly escaped and therefore the color of the Chart can be changed with the user-introduced script into the code. Escaping HTML is a technique used to prevent websites from cross-site scripting. It changes the user-introduced special characters into their corresponding HTML entities, preventing the user to inject scripting code as input. Escaping HTML is an essential Security practice for developers to implement when displaying user-generated content on a webpage. It's one layer of defense against XSS attacks and helps ensure the integrity and safety of web applications. The fix uses escapeHTML to address the Reflected XSS. EscapeHTML is a Standard mitigation technique for XSS (Cross-Site Scripting).

Vulnerability-II

Name: Kishaiyan Vellaichamy Thangaraj

Student-Id: a1819309

Link to the file-

<https://github.com/LimeSurvey/LimeSurvey/blob/master/application/controllers/UserManagementController.php>

Link to the commit-

<https://github.com/LimeSurvey/LimeSurvey/commit/553f3c4c46dedbcaa452e159aa59b0a68bdc7180>

Name of the file- UserManagementController.php

The programming language used is PHP.

Name of the repository- LimeSurvey.

Number of repository stars-2300

Number of contributors in the repositories-216

Type of Vulnerability CWE-79

The Vulnerability exploited in the above-mentioned code is Reflected Cross-Site Scripting. The code in `UserManagementController.php` does not properly validate and sanitize the input from the user. This vulnerability has the potential of disclosing sensitive information when the user injects scripting code. When the input from the user is not validated as in when the user gives an HTML code snippet or JavaScript code snippet as input when he is required to enter a number and the input is not validated or sanitized then it can be used as manipulated link to get sensitive information of other users like username, password, and card details. The fix for this vulnerability is to add validation and sanitization in the code after the user has entered their input or can use `escapeHTML` to change the special characters to their corresponding HTML entities so that the browser won't run the malicious code injected by the attacker. The commit that fixes the vulnerability follows the standard mitigation technique for XSS (Cross-Site Scripting).

Vulnerability-III

Name: Kishaiyan Vellaichamy Thangaraj

Student-Id: a1819309

Link to the file-

<https://github.com/LimeSurvey/LimeSurvey/blob/master/application/controllers/HomeSettingsController.php>

Link to the commit

<https://github.com/LimeSurvey/LimeSurvey/commit/bc2bbb9ef3bedd821a0f9ada3ccf6f79e3523bcb>

Name of the file- `HomeSettingsController.php`

The programming language used is PHP.

Name of the repository- LimeSurvey.

Number of repository stars-2300

Number of contributors in the repositories-216

Type of Vulnerability CWE-352

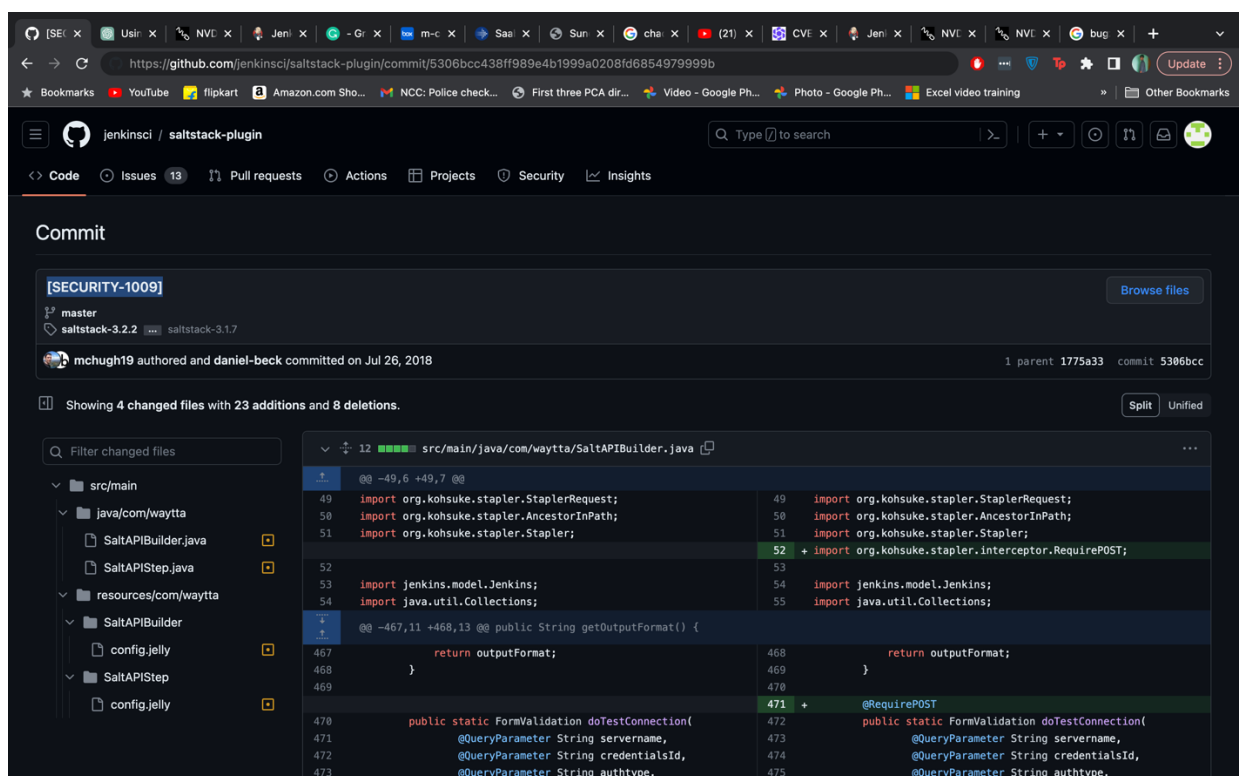
Cross-Site Request Forgery (CSRF) is a security vulnerability that occurs in websites. It exploits the trust the web application has in a user's browser. The fore-mentioned code is vulnerable to a CSRF attack where the reset all boxes button can be done with HTTP 'GET' request which indicates that the code injected can reset all boxes in the get request. This is a CSRF attack as the code couldn't differentiate a 'GET' request from a 'POST' request. The attacker could easily reset all the boxes in the GET request with malicious code injected into the HTTP request. The solution for this problem is to validate the HTTP request and allow resetting all the boxes only when it is a POST request in `HomeSettingsController.php`. Restricting certain actions to specific HTTP methods is a standard mitigation technique for CSRF attacks and many other attacks.

Part-2

Jenkins SaltStack plugin Vulnerability:

Methods implementing form validation were not subjected to permission checks by the SaltStack Plugin. Therefore, people with Overall/Read access to Jenkins had the ability to force Jenkins to send HTTP requests to attacker-specified URLs by connecting to an attacker-specified URL using attacker-specified credentials IDs obtained by a different technique and capturing credentials saved in Jenkins. The CWE-ID for this type of Cross-Site Request Forgery (CSRF) is CWE-352. When searching online with the CVE-ID got information about the vulnerability from National Vulnerability Database. The Description of the Vulnerability had the files that had the vulnerable code. Another method of locating the repository is using the fixing commit in GitHub to get to the repository. The commit has fixed the Cross-site Request Forgery for version above 3.1.6.

The provided solution for this vulnerability is to import `requirePOST` from `interceptor` to annotate a method with `'@RequirePOST'` to indicate that the method used to invoked if the incoming HTTP request is a POST request. The Instance of Jenkins Item is then checked using `checkpermission` method on the constant `Item.CONFIGURE` that refers to specific permission level defined in Jenkins and provides the ability to configure settings of Jenkin item.



```
commit 5306bcc438ff989e4b1999a0208fd685497999b
parent 1775a33
author mchugh19 <mchugh19@users.noreply.github.com>
committer daniel-beck <daniel-beck@users.noreply.github.com>
Date: Jul 26, 2018

[SECURITY-1009] saltstack-3.2.2

Showing 4 changed files with 23 additions and 8 deletions.

diff --git a/src/main/java/com/waytta/SaltAPIBuilder.java b/src/main/java/com/waytta/SaltAPIBuilder.java
index 49..51
@@ -49,6 +49,7 @@
import org.kohsuke.stapler.StaplerRequest;
import org.kohsuke.stapler.AncestorInPath;
import org.kohsuke.stapler.Stapler;
+import org.kohsuke.stapler.interceptor.RequirePOST;

import jenkins.model.Jenkins;
import java.util.Collections;

@@ -467,11 +468,13 @@ public String getOutputFormat() {
    return outputFormat;
}

+@RequirePOST
public static FormValidation doTestConnection(
    @QueryParam String servername,
    @QueryParam String credentialsId,
    @QueryParam String authtype,
```

Student name	Student ID	CVE-ID	Link	Fixing Commit	Type (CWE)	CVSS version 2 metrics	CVSS version 2 base score	Comparison with NVD	CVSS version 3 metrics	CVSS version 3 base score	Comparison with NVD
Kishaiyan Vellaichamy Thangaraj	A1819309	CVE-2018-1999027	https://github.com/jenkinsci/	[SECURITY-1009]	CWE-352	(AV:N/AC:M/Au:S/C:P/I:P/A:P)	6.0	6.8	AV:N/AC:H/PR:L	7.1	7.5

The CVSS version 2 metrics are:

Access vector-Network because the attack can happen through network.

Access Complexity-Medium because anyone with access to Jenkins can access.

Authentication- low because there is no authentication for the permission.

Confidentiality- Partial

Integrity- Partial

Availability- Partial

Base score-6.0

NVD score-6.8

The Scores are different because the Authentication is set to low whereas it is none in National Vulnerability Database.

The CVSS version 3 metrics are:

Attack Vector: Network

Attack Complexity: High

Privileges Required: Low

User Interaction: Required

Scope: Unchanged

Confidentiality: High

Integrity: High

Availability: High

Base score:7.1

NVD score:7.5

The difference in score is due to the Privileges required because the attacker would need api access to Jenkins to exploit this vulnerability.