Vulnerability Report

Introduction:

This document outlines the identification and details of a Stored Cross-Site Script (XSS) vulnerability discovered in Real Estate Management Systems in PHP

Project Name: Real Estate Management System in PHP

Version: version 1.0

Vendor: codeastro.com

Project Link: [Real Estate Management System] (https://codeastro.com/real-estate-management-system-in-php-with-source-code/)

Vulnerability Details:

Vulnerability Type: Stored Cross-Site Script (XSS)

Impact: An attacker can able to login into the application as a normal user and inject malicious payloads inside the feedback form and might be able to perform the following: -

- Data Theft: An attacker can exploit the XSS vulnerability to steal sensitive information such as user credentials, session tokens, and personal data.
- Account Takeover: If an attacker manages to steal session tokens or other authentication data, they can potentially take over admin accounts and perform actions on behalf of the legitimate user.

Affected Component:

Parameter: content

Affected URL: http://localhost/RealEstate-PHP/profile.php

Severity: High

Description:

We've discovered that the Real Estate Management System fails to validate user input in the feedback form, making it vulnerable to malicious JavaScript injection.

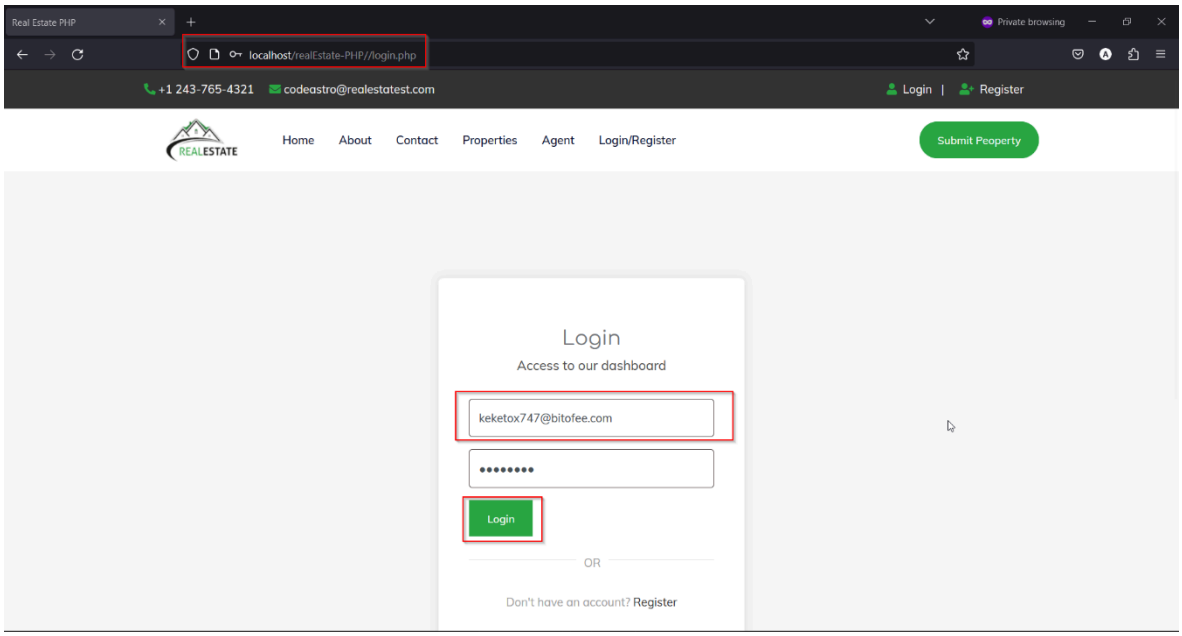
This means an attacker could insert malicious code into the feedback form, which would be stored directly in the application's database. When the admin user accesses the feedback section, this code will be executed and pop up the admin cookie.

The potential consequences are severe. The attacker could steal sensitive data, hijack admin accounts, vandalize the system, or even spread malware.

Reproduction Steps:

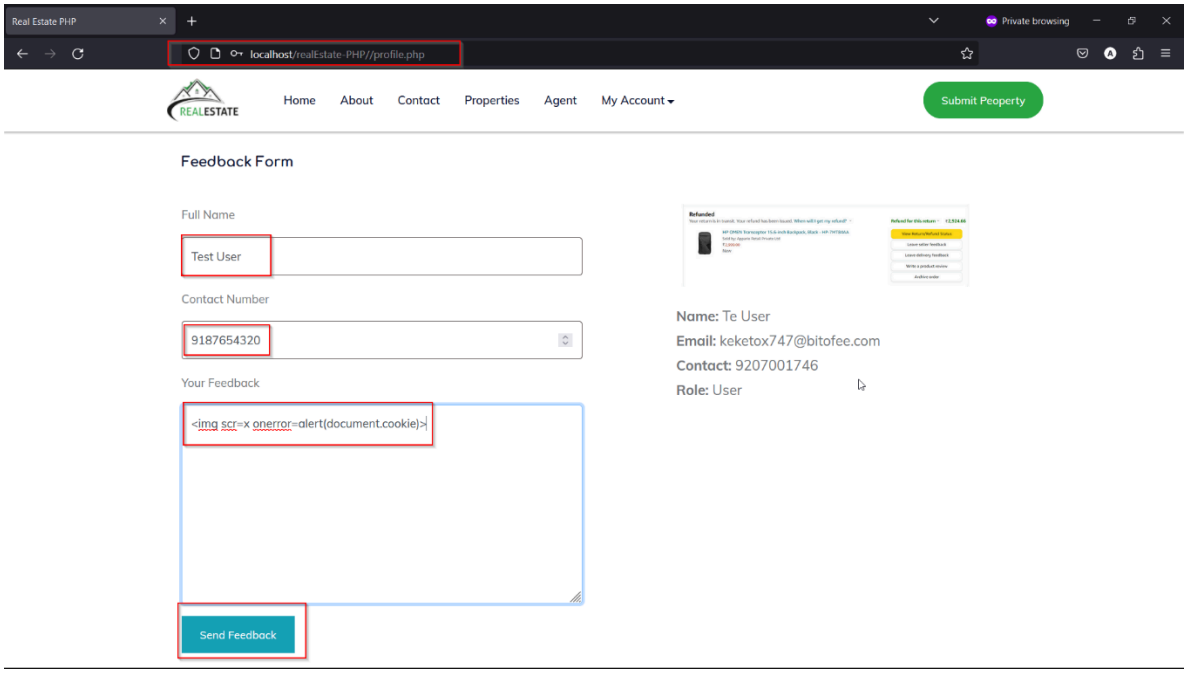
Step 1: Login into the "**Real Estate Management**" System application as a normal user

Login page URL: http://localhost/RealEstate-PHP/login.php

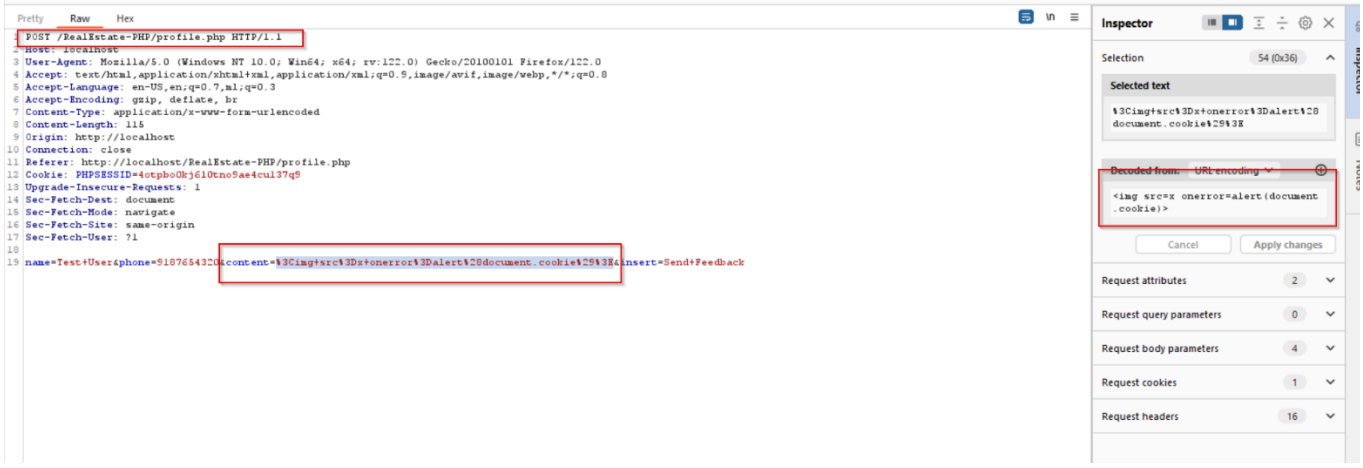


Step 2: Access the “profile.php” page and fill out the feedback form. Inside the “Your Feedback” textbox insert Malicious JavaScript Payload, same as shown Below. Click the “Send Feedback” button to submit feedback.

Payload used: <img src=x onerror=alert(document.cookie)>

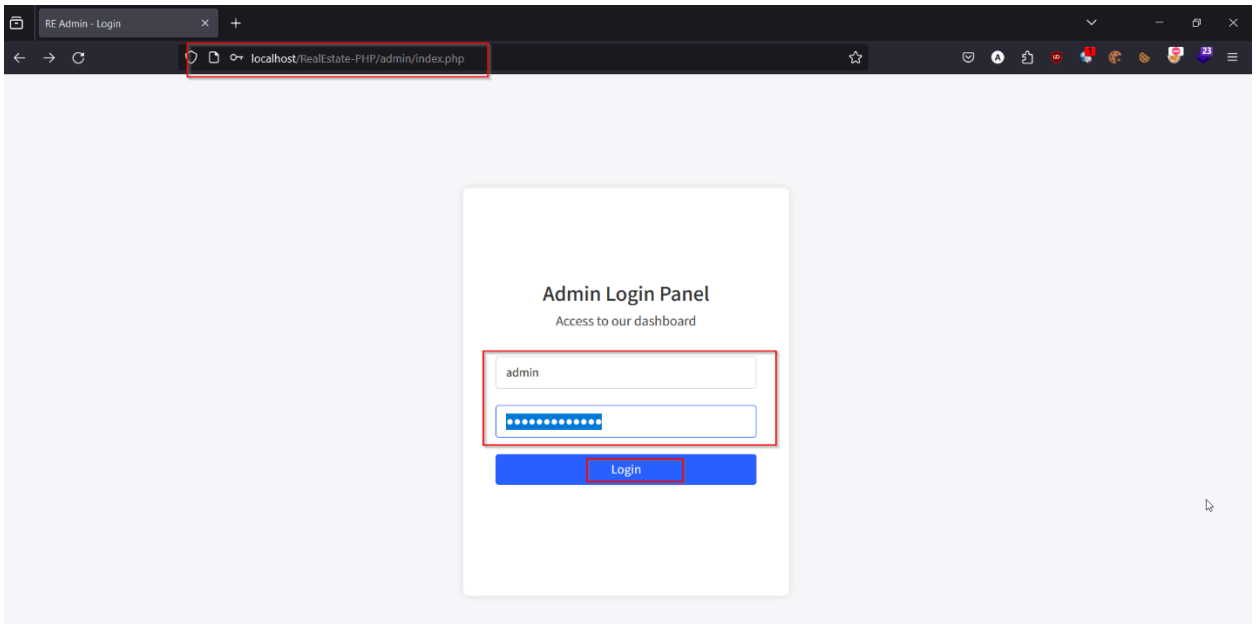


Step 3: The below screenshot shows the request captured on Burp Suite for the Feedback

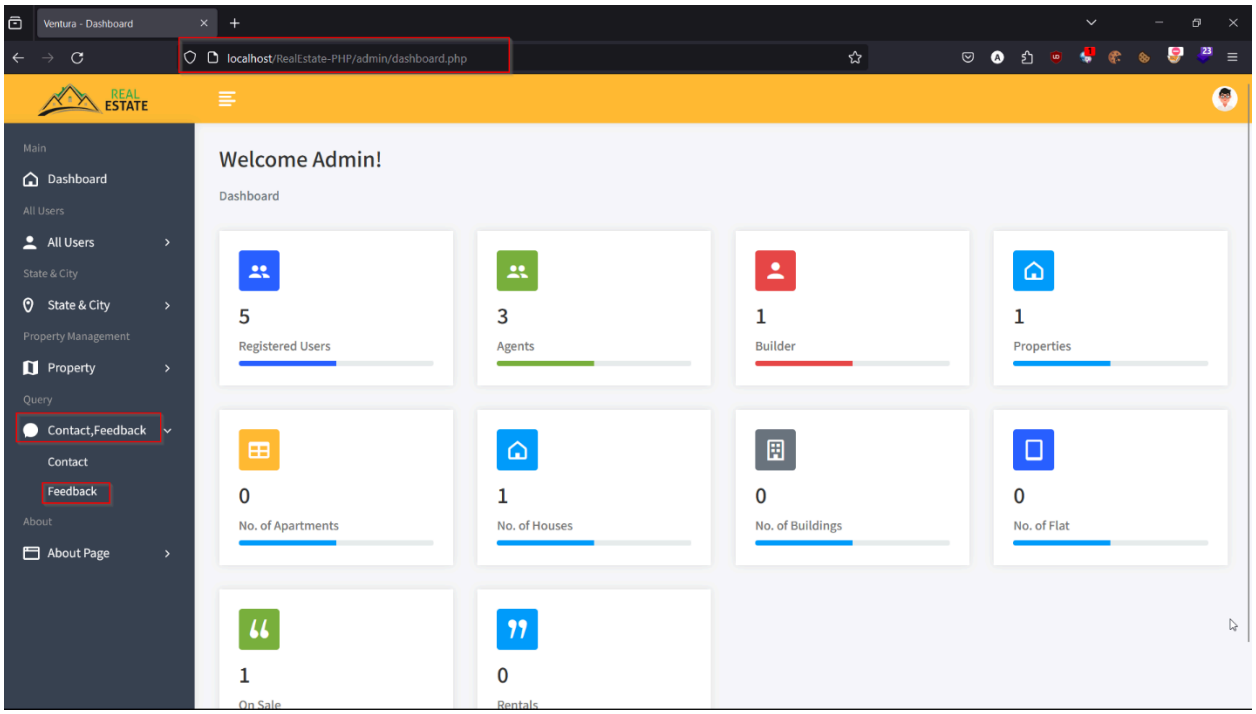


Step 4: Login as an admin User.

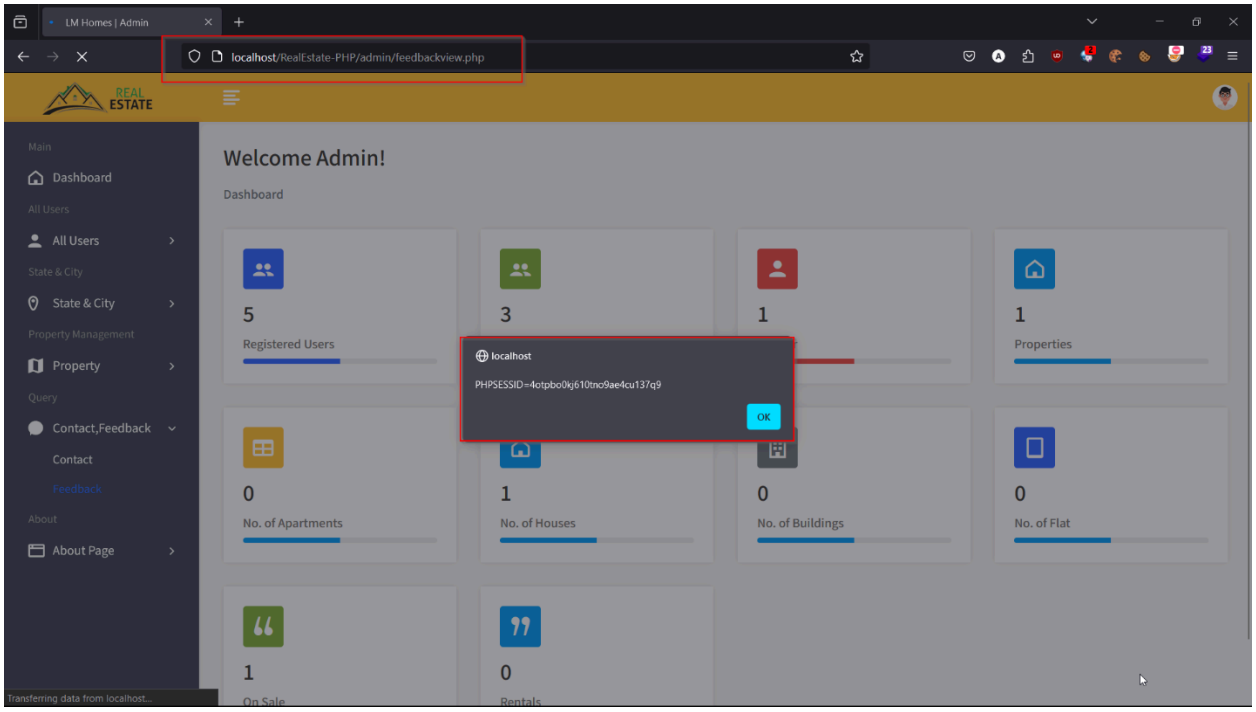
URL for admin login: http://localhost/RealEstate-PHP/admin/



Step 5: From the Admin dashboard access the "**Contact, Feedback >> Feedback**" for visiting user input feedback.



Step 6: The below screenshot shows that the user inserted a malicious payload executed successfully and pop-up the admin cookie.



Mitigations:

We recommend the following for mitigating the Stored Cross-Site Scripting (XSS):

- **Content Security Policy (CSP):** Implement CSP headers to specify which sources of content are allowed to be executed on a web page. CSP can mitigate XSS attacks by restricting the execution of scripts and other resources to trusted sources.
- **Output Encoding:** Encode user-generated content properly before displaying it in web pages. Use appropriate encoding mechanisms like HTML entity encoding, JavaScript escaping, and URL encoding to prevent browsers from interpreting input as executable code.
- **Context-Specific Output Escaping:** Understand the context in which user input will be used (e.g., HTML body, attribute, JavaScript, CSS) and apply the appropriate escaping mechanism for that context. Different contexts require different escaping techniques to ensure security.
- **Security Headers:** Implement security headers like X-XSS-Protection, X-Content-Type-Options, and X-Frame-Options to provide additional layers of defense against various types of attacks, including XSS.

- Implement **‘HttpOnly’** and **‘Secure”** flags for cookies**.**