TEAM ZEN

Penetration Testing Report

Snipe-IT Application

# Table of Contents

[Table of Contents 1](#_Toc128937902)

[1. Testing Details 2](#_Toc128937903)

[1.1. Findings Summary 2](#_Toc128937904)

[1.2. Clear text submission of password 2](#_Toc128937905)

[Severity 3](#_Toc128937906)

[Description 3](#_Toc128937907)

[Remediation 3](#_Toc128937908)

[1.3. Cookie without HttpOnly Flag Set 2](#_Toc128937909)

[Severity 3](#_Toc128937910)

[Description 3](#_Toc128937911)

[If the HttpOnly attribute is set on a cookie, the cookie’s value cannot be read or set by the client-side JavaScript. 3](#_Toc128937912)

[Remediation 3](#_Toc128937913)

[1.4. Source code disclosure 2](#_Toc128937914)

[Severity 3](#_Toc128937915)

[Description 3](#_Toc128937916)

[Remediation 3](#_Toc128937917)

[1.5. Content type incorrectly stated. 2](#_Toc128937918)

[Severity 3](#_Toc128937919)

[Description 3](#_Toc128937920)

[Remediation 3](#_Toc128937921)

[1.6. Unencrypted Communications 2](#_Toc128937922)

[Severity 3](#_Toc128937923)

[Description 3](#_Toc128937924)

[Remediation 3](#_Toc128937925)

[2. Severity Ratings 2](#_Toc128937926)

## Testing Details

The Security Assessment team performed a security assessment of the Snipe-IT system.

## Findings Summary

During the assessment of Snipe-IT 5 findings were discovered as detailed in Table 1 - Findings Summary. Details for each finding as well as a recommendation for each finding are contained in the sections below.

|  |  |  |
| --- | --- | --- |
|  | Name | Severity |
| 1 | Clear text submission of password | High |
| 2 | Cookie without HttpOnly Flag Set | Low |
| 3 | Source code disclosure | Low |
| 4 | Content type incorrectly stated | Low |
| 5 | Unencrypted Communications | Low |

Table 1 - Findings Summary

## Clear text submission of password

### Severity

High

### Description

The login page contains a form with the following action URL, which is submitted over clear-text HTTP. If the attacker can eavesdrop on the traffic between the application and the user’s machine, they will be able to obtain the user’s credentials.

http://snipeitam.c3s.zenstrategics.com/login

Graphical user interface, text, application

Description automatically generated

Figure 1. Request header with the credentials in clear text

### Remediation

Sensitive communications should be protected with a valid transport-level encryption mechanism, have their own error handling and logging requirements Any functions where sensitive data can be accessed or privileged actions can be performed should employ their own session handling mechanism, and the session tokens should never be transmitted over unencrypted communications (as is the case with this application). If HTTP cookies are used for transmitting session tokens (again, as is the case here), then the secure flag should be set to prevent transmission over clear-text HTTP

## Cookie without HttpOnly Flag Set

### Severity

Low

### Description

The security assessment team observed the following cookie that appeared to have contained a session token, and did not have the HttpOnly flag set, exposing the data in the cookie to client-side cross-site scripting attacks.



Figure 2. Response header with cookie without HttpOnly Flag Set

If the HttpOnly attribute is set on a cookie, the cookie’s value cannot be read or set by the client-side JavaScript.

### Remediation

Setting HttpOnly attribute on the cookie that contains session tokens would make client-side attacks such as cross-site scripting, more difficult.

## Source code disclosure

### Severity

Low

### Description

The application appears to disclose some of the source code to the users. This type of exposure is classified as CWE-200, exposure of sensitive information to an unauthorized actor.

<https://cwe.mitre.org/data/definitions/200.html>

Source code may contain sensitive information and allow the attacker to gain visibility into, and leverage, potential weaknesses in the application.

Graphical user interface, text, application

Description automatically generated

Figure 3. Response header with code disclosure

### Remediation

Review the cause of the source code disclosure for potential permissions or typographical script errors.

## Content type incorrectly stated.

### Severity

Low

### Description

The response request states that the content type if font/woff2. However, it appears to contain unrecognized content. If the URL can be manipulated to end with “.html”, certain browsers may interpret the response as HTML, exposing the end user to a client side cross-scripting attack.

Text

Description automatically generated

Figure 4. Response header with mismatched content type

### Remediation

For every response containing a message body, the application should include a single Content-type header that correctly states the MIME type of the content in the response body. The response header “X-content-type-options: nosniff” should be returned in all responses to reduce the likelihood that browsers will interpret content in a way that disregards the Content-type header.

## Unencrypted Communications

### Severity

Low

### Description

Transport-level security is not implemented to protect the communications between the end client and the Snipe-IT web application.

### Remediation

Implement Transport Layer Security (TLS) to protect communications. Utilize the header ‘Strict-Transport-Security’ to force end clients to use https.

## Severity Ratings

The severity levels used for the findings are based off of the CVSS v3 rating scale. Table 2 - Severity Levels provides a description for each severity level.

|  |  |
| --- | --- |
| Critical | Exploitation of the vulnerability likely results in root-level compromise of servers or infrastructure devices.  Exploitation is usually straightforward, in the sense that the attacker does not need any special authentication credentials or knowledge about individual victims, and does not need to persuade a target user, for example via social engineering, into performing any special functions.  For critical vulnerabilities, is advised that you patch or upgrade as soon as possible, unless you have other mitigating measures in place. For example, a mitigating factor could be if your installation is not accessible from the Internet. |
| High | The vulnerability is difficult to exploit.  Exploitation could result in elevated privileges.  Exploitation could result in a significant data loss or downtime. |
| Medium | Vulnerabilities that require the attacker to manipulate individual victims via social engineering tactics.  Denial of service vulnerabilities that are difficult to set up.  Exploits that require an attacker to reside on the same local network as the victim.  Vulnerabilities where exploitation provides only very limited access.  Vulnerabilities that require user privileges for successful exploitation. |
| Low | Vulnerabilities in the low range typically have very little impact on an organization's business. Exploitation of such vulnerabilities usually requires local or physical system access. |