**25Q2 CSOC Helpdesk-dev.zalaris.com Manual**

**Penetration Test June 11**

PREPARED BY:

Mohammed Sharoz

REVIEWED BY:

Diwakar Thanikachalam

SIGNED-OFF BY:

Name Middle Lastname

### Introduction:

The applications that are owned or used within Zalaris are actively tested for vulnerability using automated scan followed by manual Pen-test of the report. The scan is done once a month

# 

Table of Contents

[Introduction: 3](#_Toc6033)

[2025 Q2 CSOC Manual Penetration test of Alumni Portal 5](#_Toc32300)

[Executive Summary 5](#_Toc9844)

[Client SIde JSON Injection (DOM Based): 5](#_Toc322)

[Cookie Manipulation (DOM Based): 7](#_Toc32281)

[Cross site scripting (Stored): 9](#_Toc1852)

[Cross site scripting (Reflected): 11](#_Toc13967)

[External Service Interaction(DNS): 13](#_Toc11091)

[User Agent Dependent Response: 14](#_Toc23554)

[Input Returned in Response(Stored) 16](#_Toc6003)

[Input Returned in Response (Reflected): 18](#_Toc19802)

[Frame-able Response (Potential Click jacking): 20](#_Toc16611)

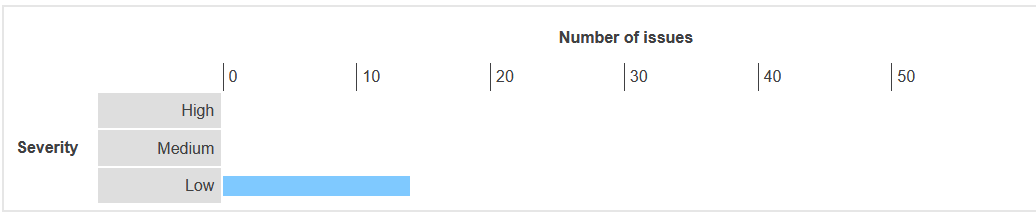
[HTML 5 Storage Manipulation(DOM Based): 23](#_Toc31930)

[TLS Certificate: 26](#_Toc17233)

### 2025 Q2 CSOC Manual Penetration test of Alumni Portal

### Executive Summary

The chart below shows the aggregated numbers of issues identified in each category. Solid colored bars represent issues with a confidence level, and the bars fade as the confidence level falls.



### 

### Client SIde JSON Injection (DOM Based):

|  |  |
| --- | --- |
| Type | Further Test Needed added to backlog |
| Status | In Progress |

**Description**

DOM-based vulnerabilities arise when a client-side script reads data from a controllable part of the DOM (for example, the URL) and processes this data in an unsafe way.

DOM-based cookie manipulation arises when a script writes controllable data into the value of a cookie. An attacker may be able to use the vulnerability to construct a URL that, if visited by another application user, will set an arbitrary value in the user's cookie.

The potential impact of the vulnerability depends on the role that the cookie plays within the application. If the cookie is used to control the behavior that results from certain user actions (for example, a 'production' versus 'demo' mode setting), then the attacker may be able to cause the user to perform unintended actions by manipulating the cookie's value. If the cookie is used to track the user's session, then the attacker may be able to perform a session fixation attack, in which they set the cookie's value to a valid token that they have obtained from the application, and then hijack the session during the victim user's subsequent interaction with the application.

Burp Suite automatically identifies this issue using dynamic and static code analysis. Static analysis can lead to false positives that are not actually exploitable. If Burp Scanner has not provided any evidence resulting from dynamic analysis, you should review the relevant code and execution paths to determine whether this vulnerability is indeed present, or whether mitigations are in place that would prevent exploitation.

**Links Tested**

|  |
| --- |
| * [/](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "1.1) * [/510](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "1.2) * [/510/tickets/new/additional-tax-deduction](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "1.3) * [/510/tickets/new/draft-template-testing](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "1.4) * [/510/tickets/new/einstellungsbogen](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "1.5) * [/510/tickets/new/employee-transfer](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "1.6) * [/510/tickets/new/ordering-an-employment-certificate-requsition-form](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "1.7) * [/510/tickets/new/schedule-an-expert-inquiry](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "1.8) * [/777](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "1.9) * [/777/tickets/new/additional-tax-deduction](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "1.10) * [/777/tickets/new/employee-transfer](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "1.11) |

Steps:

* Login into Helpdesk-dev.zalaris.com
* The source and sink values are not proper and must be tested further

**WORKLOG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| When | What | Who | Recommended Action | Action plan |
| 16 June 2025 | Need more information to test the alert received | CSOC | NA | Work Item has been created |

### Cookie Manipulation (DOM Based):

|  |  |
| --- | --- |
| Type | False Positive |
| Status | False positive no action needed |

**Description**

DOM-based vulnerabilities arise when a client-side script reads data from a controllable part of the DOM (for example, the URL) and processes this data in an unsafe way.

DOM-based cookie manipulation arises when a script writes controllable data into the value of a cookie. An attacker may be able to use the vulnerability to construct a URL that, if visited by another application user, will set an arbitrary value in the user's cookie.

The potential impact of the vulnerability depends on the role that the cookie plays within the application. If the cookie is used to control the behavior that results from certain user actions (for example, a 'production' versus 'demo' mode setting), then the attacker may be able to cause the user to perform unintended actions by manipulating the cookie's value. If the cookie is used to track the user's session, then the attacker may be able to perform a session fixation attack, in which they set the cookie's value to a valid token that they have obtained from the application, and then hijack the session during the victim user's subsequent interaction with the application.

Burp Suite automatically identifies this issue using dynamic and static code analysis. Static analysis can lead to false positives that are not actually exploitable. If Burp Scanner has not provided any evidence resulting from dynamic analysis, you should review the relevant code and execution paths to determine whether this vulnerability is indeed present, or whether mitigations are in place that would prevent exploitation.

**Links Tested**

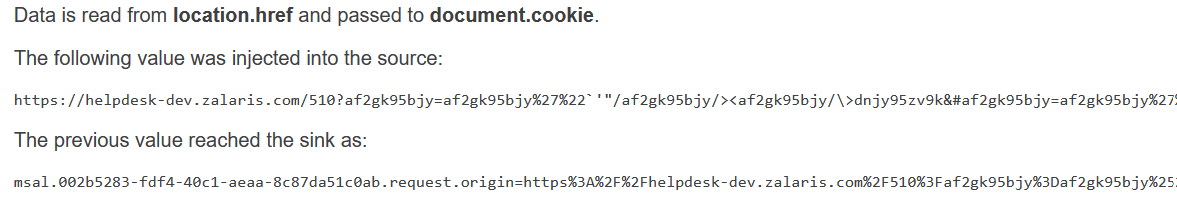
|  |
| --- |
| * [/510](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "2.1) * [/510/tickets/new/einstellungsbogen](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "2.2) * [/777/tickets/new/employee-transfer](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "2.3) |

Steps:

* Login into helpdesk-dev.zalaris.com

Access the path informed with DOM based cookie manipulation and look for any values passed on to the sink msal.002b5283-fdf4-40c1-aeaa-8c87da51c0ab.request.origin=https://helpdesk-dev.zalaris.com

* The above value is an unique identifier for authentication request followed by the origin URL which the authentication was requested. So can be used by the application to find whether the authentication origin was requested
* The screenshot for the same is given below



* The value is not user controlled it is generated by the web application itself and the method request.origin from msal is used for tracking the origin of an authentication request alone
* Therefore this alert is considered as false positive

**WORKLOG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| When | What | Who | Recommended Action | Action plan |
| 16 June 2025 | Manually Tested DOM based Cookie Manipulation | CSOC | The alert is false positive no action needed | No action needed |

### Cross site scripting (Stored):

|  |  |
| --- | --- |
| Type | False Positive |
| Status | No action Needed the alert is false positive |

**Description**

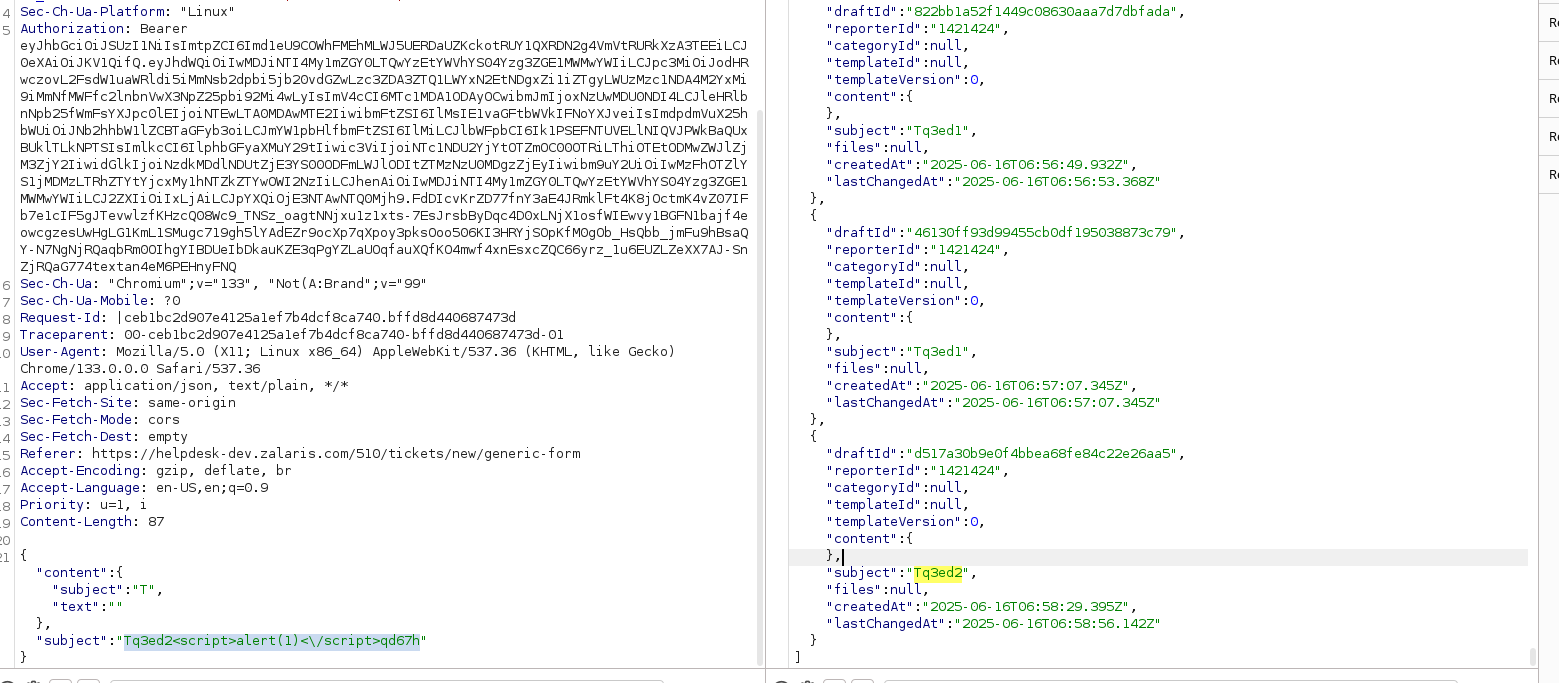
The value of the **subject** JSON parameter submitted to the URL /api/v1/drafts is copied into the HTML document as plain text between tags at the URL /api/v1/drafts. The payload **q3ed1<script>alert(1)</script>qd67h** was submitted in the subject JSON parameter. This input was returned unmodified in a subsequent request for the URL /api/v1/drafts.  
  
This proof-of-concept attack demonstrates that it is possible to inject arbitrary JavaScript into the application's response.  
  
The response does not state that the content type is HTML. The issue is only directly exploitable if a browser can be made to interpret the response as HTML. No modern browser will interpret the response as HTML. However, the issue might be indirectly exploitable if a client-side script processes the response and embeds it into an HTML context.

**Links Tested**

|  |
| --- |
| * /api/v1/drafts |

Steps:

* Login into helpdesk-dev.zalaris.com/510
* Access the path that is reported with stored cross site scripting issue
* Injected the payload , forward the request and analyze the response
* The response shows that the injected payloads are reflected but are sanitized.The screenshot for the same is attached below



* The injected script is sanitized and not reflected back to the user and not executed as a Javascript

**WORKLOG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| When | What | Who | Recommended Action | Action plan |
| 16 June 2025 | Manually Tested - Cross Site Scripting (Stored) alert | CSOC | The injected scrip is sanitized and is not reflected back to the user. No action needed |  |

### Cross site scripting (Reflected):

**Category:** Low

|  |  |
| --- | --- |
| Type | False Positive |
| Status | Have been added to back log for further testing |

**Description**

The value of the **size** request parameter is copied into the HTML document as plain text between tags. The payload **xz5ds<script>alert(1)</script>a9mey** was submitted in the size parameter. This input was echoed unmodified in the application's response.  
  
This proof-of-concept attack demonstrates that it is possible to inject arbitrary JavaScript into the application's response.  
  
The response does not state that the content type is HTML. The issue is only directly exploitable if a browser can be made to interpret the response as HTML. No modern browser will interpret the response as HTML. However, the issue might be indirectly exploitable if a client-side script processes the response and embeds it into an HTML context.

**Links Tested**

|  |
| --- |
| * /api/v1/tickets |

Steps:

* Login into Helpdesk-dev.zalaris.com/510
* Access the following page reported in the alert /api/v1/tickets?contactId=1421414&size=3xz5ds%3cscript%3ealert(1)%3c%2fscript%3ea9mey&sort=waitingForCustomer%2Cdesc&sort=lastChangedAt%2Cdesc
* Inject a script into the applications size parameter and forward the request look for the response. The response shows that the script is sanitized.
* The screenshot for the same is attached below



* The payload is reflected as a normal text and not executed as a Java Script

**WORKLOG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| When | What | Who | Recommended Action | Action plan |
| 11 June 2025 | Manually Tested the reported Reflected XSS issue | CSOC | The Injected scripts are reflected in the response but, not executed as Java Scripts and are sanitized. Considering this alert as informational | Need further testing to break out of the sanitized area |

### External Service Interaction(DNS):

**Category:** Low

|  |  |
| --- | --- |
| Type | False Positive |
| Status | False positive no action needed |

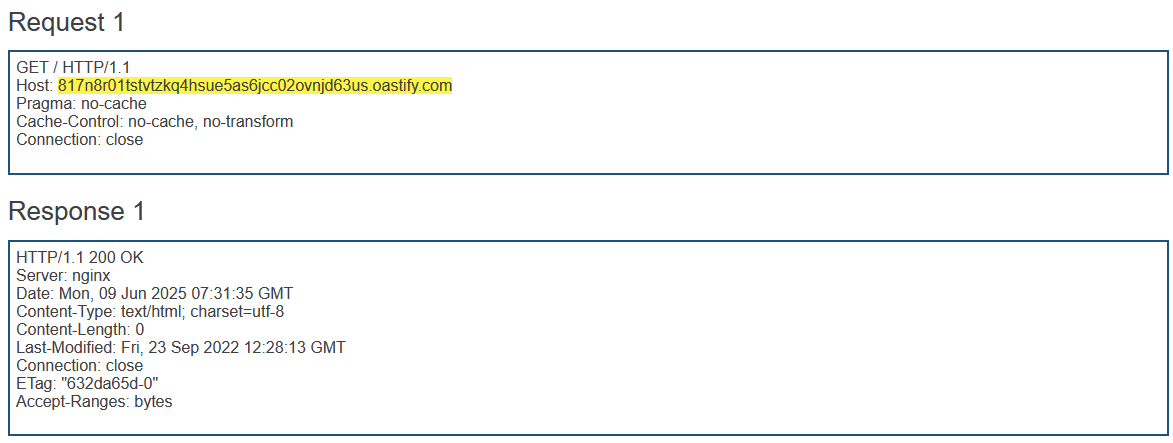
**Description**

It is possible to induce the application to perform server-side DNS lookups of arbitrary domain names.  
  
The payload **817n8r01tstvtzkq4hsue5as6jcc02ovnjd63us.oastify.com** was submitted in the SSL SNI value and the HTTP Host header.  
  
The application performed a DNS lookup of the specified domain.

**Links Tested**

|  |
| --- |
| * /country-flags/de.svg |

**Note:** The alert provided is false positive since the request was directly made to the burp collaborator. Screenshot for the same is attached below



But the path that is mentioned in the alert is /country-flags/de.svg which is not releveant to external DNS Interaction

**WORKLOG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| When | What | Who | Recommended Action | Action plan |
| 12 June 2025 | External DNS Interaction Manually Tested | CSOC | No Action Needed. The Alert is False Positive | No Action Needed |

### User Agent Dependent Response:

**Category:** Low

|  |  |
| --- | --- |
| Type | Informational Alert |
| Status | The alert is informational still work item has been created to test for User Agent Dependent Response |

**Description**

Application responses may depend systematically on the value of the User-Agent header in requests. This behavior does not itself constitute a security vulnerability, but may point towards additional attack surface within the application, which may contain vulnerabilities.

This behavior often arises because applications provide different user interfaces for desktop and mobile users. Mobile interfaces have often been less thoroughly tested for vulnerabilities such as cross-site scripting, and often have simpler authentication and session handling mechanisms that may contain problems that are not present in the full interface.

To review the interface provided by the alternate User-Agent header, you can configure a match/replace rule in Burp Proxy to modify the User-Agent header in all requests, and then browse the application in the normal way using your normal browser.

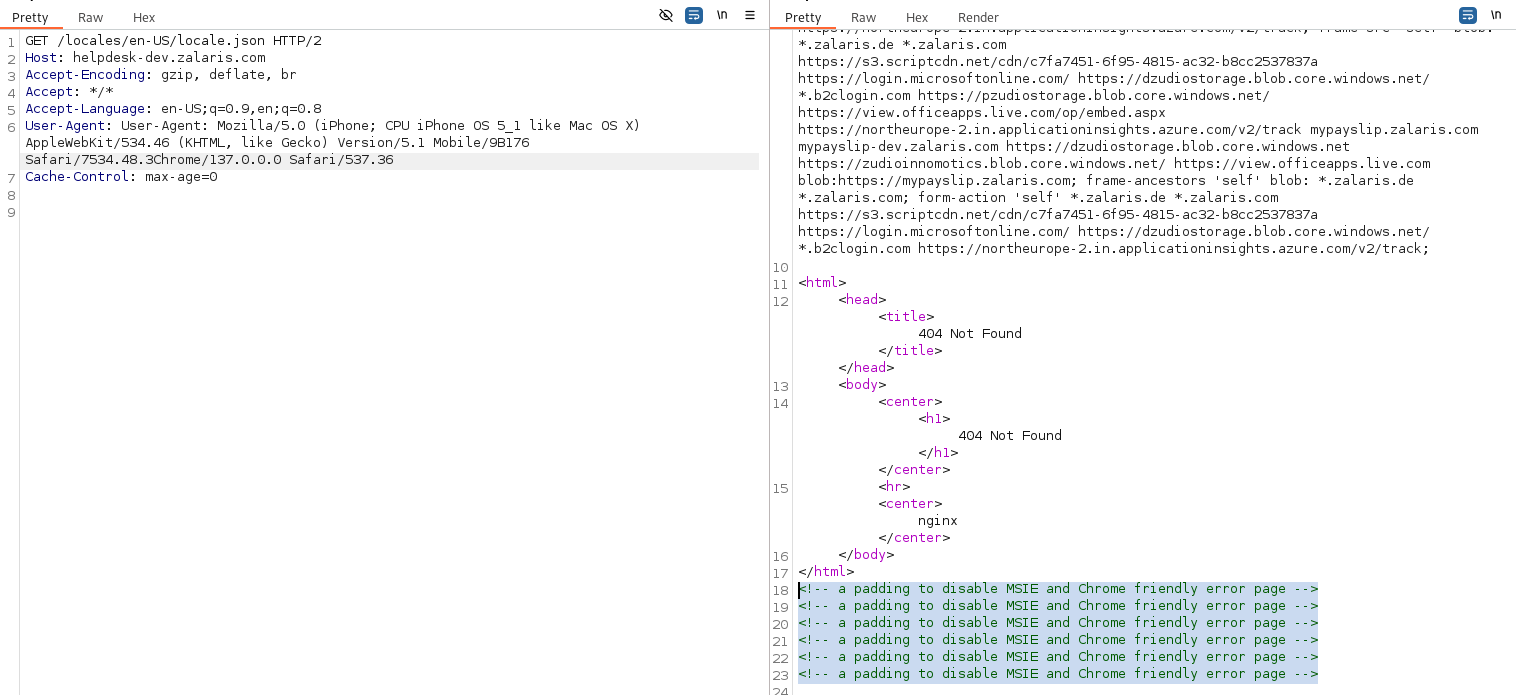
**Links Tested**

|  |
| --- |
| * **/locales/en-US/locale.json** |

Steps:

* Access the path reported with User Agent Dependent response
* The Response Varies according to the User Agent. The screenshot for the same is attached below





* This behavior is itself is not an issue since the User Agent Header is used to display or render the content to the user based on the device used . Still there needs to be User-Agent Based test that is needed to execute scripts such as Shell-Shock

**WORKLOG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| When | What | Who | Recommended Action | Action plan |
| 11 June 2025 | Manually Tested the User Agent based attack reported in the mandatory Automated Audit | CSOC | The alert is just informational | Perform Injection attacks against the User agent header targeting the server or the application. . Work Item 332 Backlog Test : User Agent Based Attack Vector |

### Input Returned in Response(Stored)

**Category:** Low

|  |  |
| --- | --- |
| Type | Informational Alert |
| Status | No Action needed . The Alerts are informational. |

**Description**

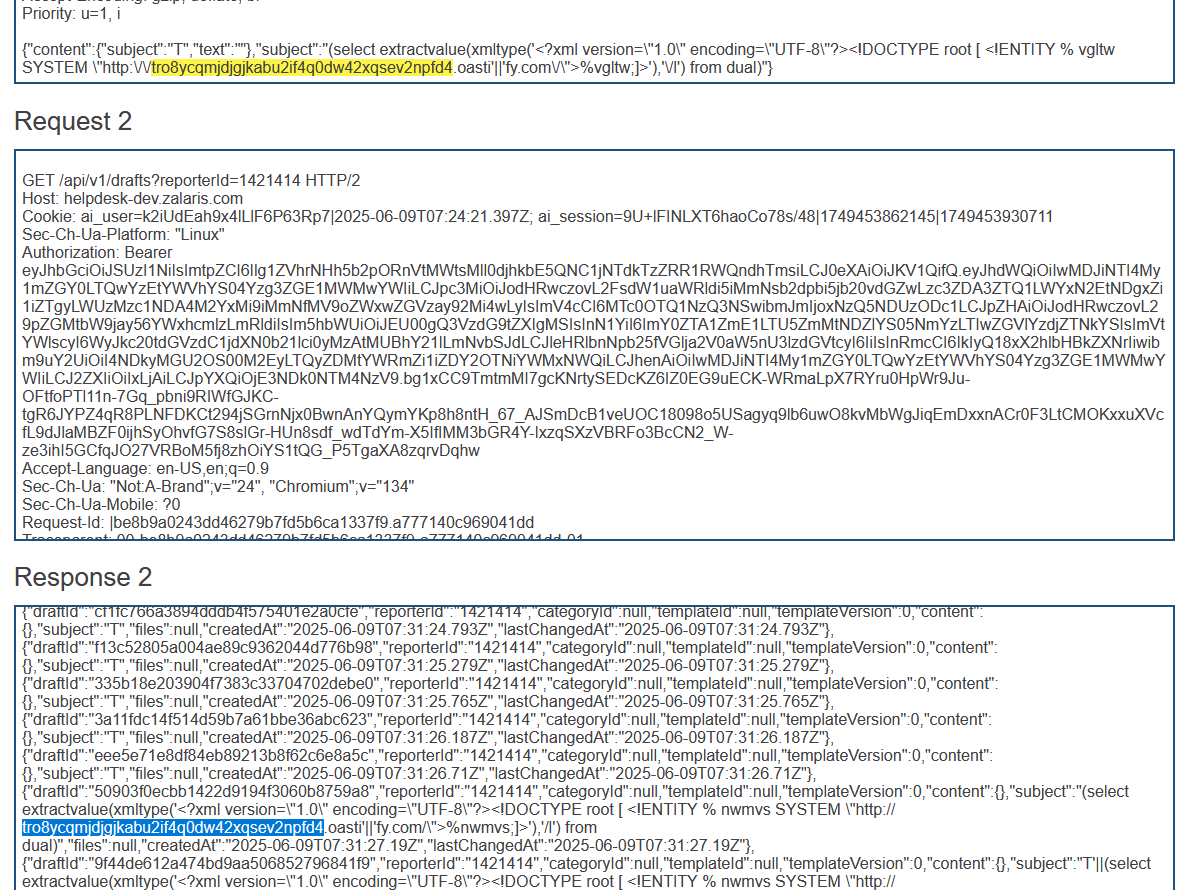
The value of the **subject** JSON parameter submitted to the URL /api/v1/drafts is copied into the response for the URL /api/v1/drafts.  
  
Burp has captured the first observed location where this stored input is returned. There might be other locations within the application where the same input is returned. To identify all such locations, perform a full crawl of the application and then do a global search for the highlighted value.

**Links Tested**

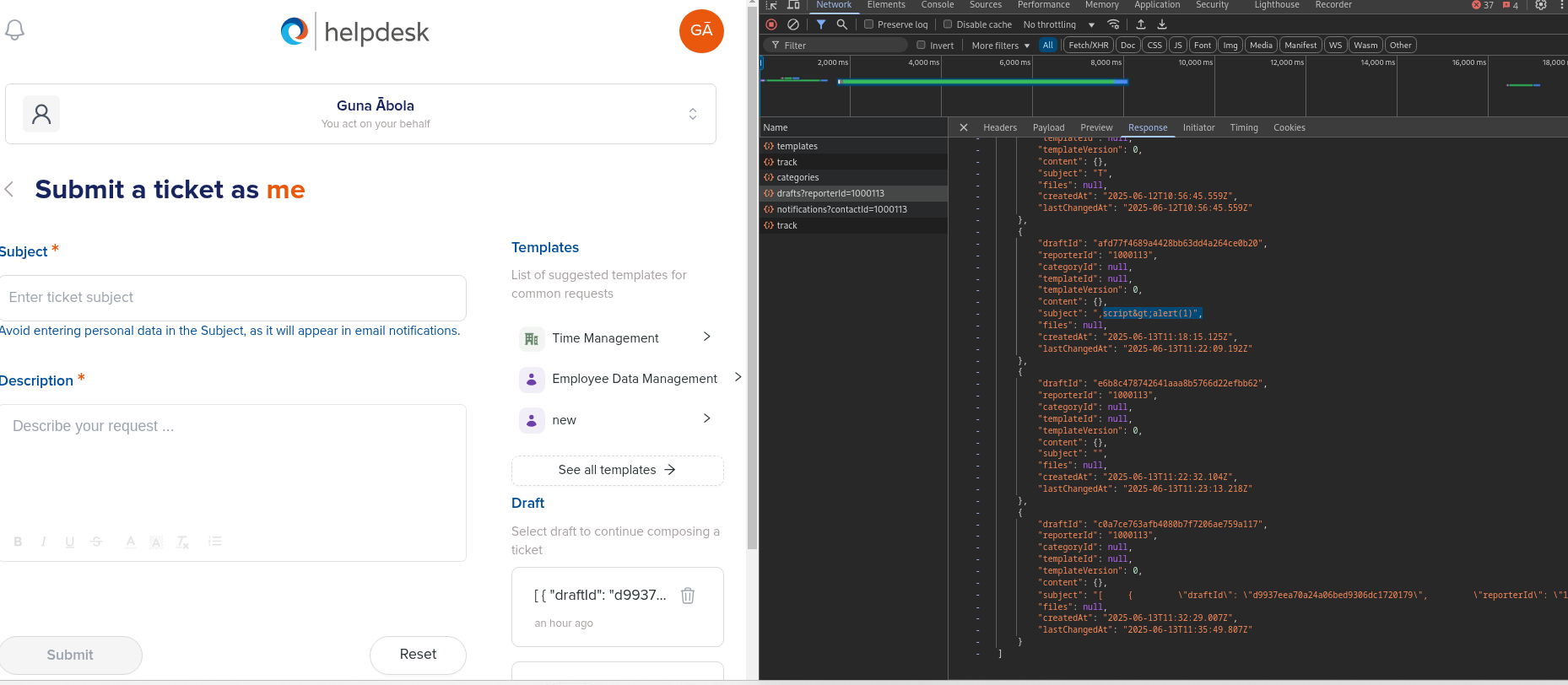
|  |
| --- |
| * **/api/v1/drafts** |

Steps:

* Login into helpdesk-dev.zalaris.com/510
* Access the path reported to have the input reflected - this the draft page for ticket . The screenshot showing the input given and the same value available in the response is given below



* Inject payloads in the subject field and analyze response of the same
* The given inputs are reflected in the response of the draft page’s but are sanitized. The screenshot for the same is attached below
* The inputs are sanitized and not reflected to the user . No issue found



**WORKLOG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| When | What | Who | Recommended Action | Action plan |
| 13 June 2025 | Manually Tested input returned in response (Stored) | CSOC | The inputs are sanitized and not reflected to the user . No issue found | No action needed |

### Input Returned in Response (Reflected):

**Category:** Low

|  |  |
| --- | --- |
| Type | Informational Alert |
| Status |  |

**Description**

Reflection of input arises when data is copied from a request and echoed into the application's immediate response.

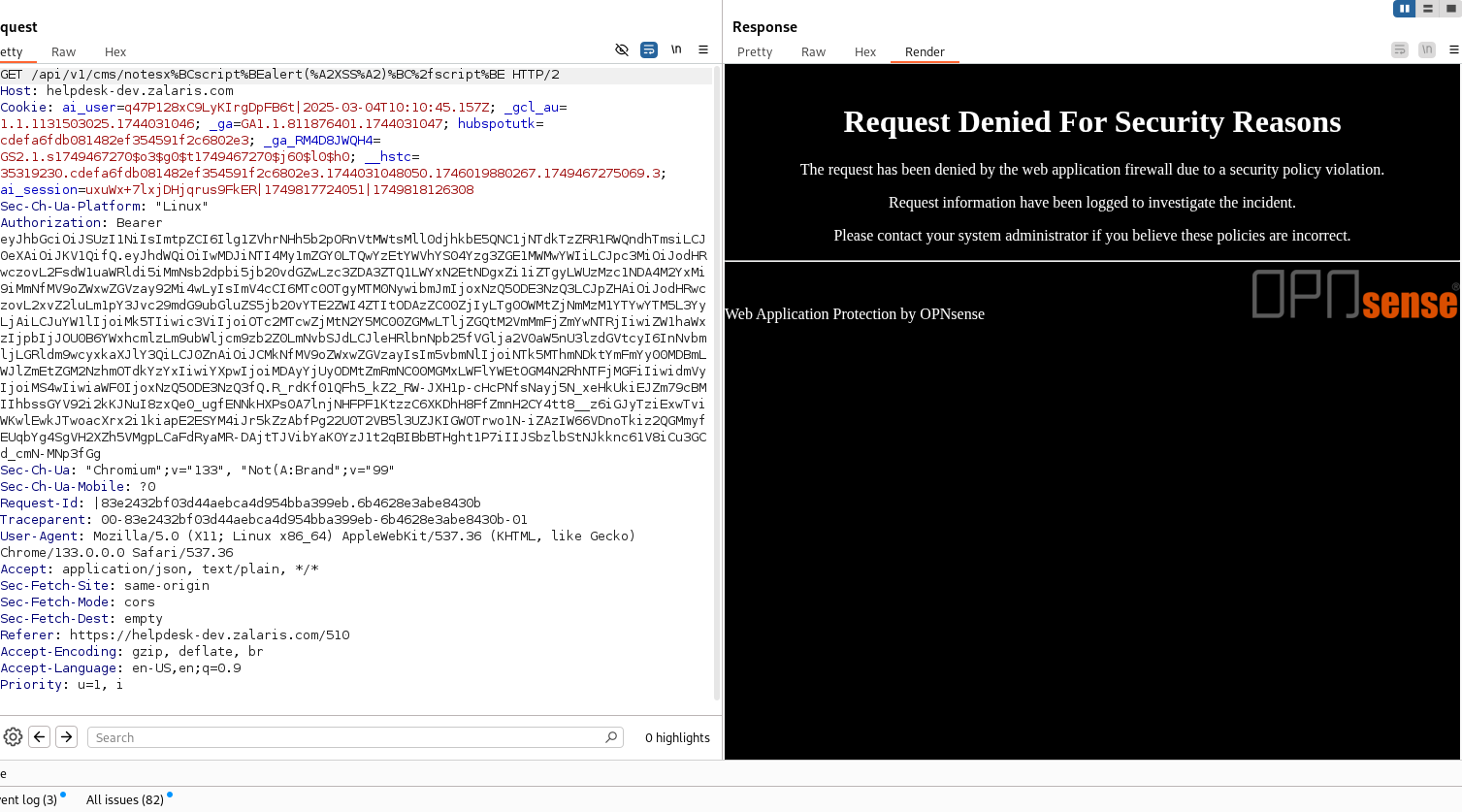
Input being returned in application responses is not a vulnerability in its own right. However, it is a prerequisite for many client-side vulnerabilities, including cross-site scripting, open redirection, content spoofing, and response header injection. Additionally, some server-side vulnerabilities such as SQL injection are often easier to identify and exploit when input is returned in responses. In applications where input retrieval is rare and the environment is resistant to automated testing (for example, due to a web application firewall), it might be worth subjecting instances of it to focused manual testing.

**Links Tested**

|  |
| --- |
| * [/api/v1/cms/notes [URL path filename]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.1) * [/api/v1/cms/notes [URL path folder 2]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.2) * [/api/v1/cms/notes [URL path folder 3]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.3) * [/api/v1/cms/updates [URL path filename]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.4) * [/api/v1/cms/updates [URL path folder 3]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.5) * [/api/v1/contacts/authenticatedUser [URL path filename]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.6) * [/api/v1/contacts/authenticatedUser [URL path folder 2]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.7) * [/api/v1/contacts/authenticatedUser [URL path folder 3]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.8) * [/api/v1/drafts [URL path filename]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.9) * [/api/v1/drafts [URL path folder 2]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.10) * [/api/v1/drafts/41d006eb49be48a2bc706b3251fe3c43 [URL path filename]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.11) * [/api/v1/drafts/41d006eb49be48a2bc706b3251fe3c43 [URL path folder 2]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.12) * [/api/v1/drafts/41d006eb49be48a2bc706b3251fe3c43 [URL path folder 3]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.13) * [/api/v1/health-check [URL path filename]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.14) * [/api/v1/health-check [URL path folder 2]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.15) * [/api/v1/notifications [URL path filename]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.16) * [/api/v1/templates/categories [URL path filename]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.17) * [/api/v1/templates/categories [URL path folder 3]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.18) * [/api/v1/templates/slug/additional-tax-deduction [URL path folder 3]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.19) * [/api/v1/templates/slug/additional-tax-deduction [URL path folder 4]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.20) * [/api/v1/templates/slug/employee-transfer [URL path folder 3]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.21) * [/api/v1/templates/slug/employee-transfer [URL path folder 4]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.22) * [/api/v1/tickets [size parameter]](C:/Users/Mohammed Sharoz/Downloads/2025 Q2 CSOC Helpdesk Automated Audit June 9.html" \l "8.23) |

Steps:

* Login into helpdesk-dev.zalaris.com/510
* Crawl the application and access the link reported with the input reflected
* Replace the reflected value with XSS scripts and analyze the response
* The response shows that the firewall has detected a potential payload or response for the injected payload and has prevented the application from displaying the result t o the user. The Screenshot for the same is attached below



**WORKLOG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| When | What | Who | Recommended Action | Action plan |
| 16 June 2025 | Manually Tested input returned in response (reflected) | CSOC | Work item created for input returned in response - work item 335 Created for further testing to bypass the XSS prevention by firewall | Work Item 335 Created . Test must be done and confirmed as whether vulnerable or not |

### Frame-able Response (Potential Click jacking):

**Category:** Low

|  |  |
| --- | --- |
| Type | False Positive |
| Status | Tested and confirmed as False positive no action needed |

**Description**

If a page fails to set an appropriate X-Frame-Options or Content-Security-Policy HTTP header, it might be possible for a page controlled by an attacker to load it within an iframe. This may enable a clickjacking attack, in which the attacker's page overlays the target application's interface with a different interface provided by the attacker. By inducing victim users to perform actions such as mouse clicks and keystrokes, the attacker can cause them to unwittingly carry out actions within the application that is being targeted. This technique allows the attacker to circumvent defenses against cross-site request forgery, and may result in unauthorized actions.

Note that some applications attempt to prevent these attacks from within the HTML page itself, using "framebusting" code. However, this type of defense is normally ineffective and can usually be circumvented by a skilled attacker.

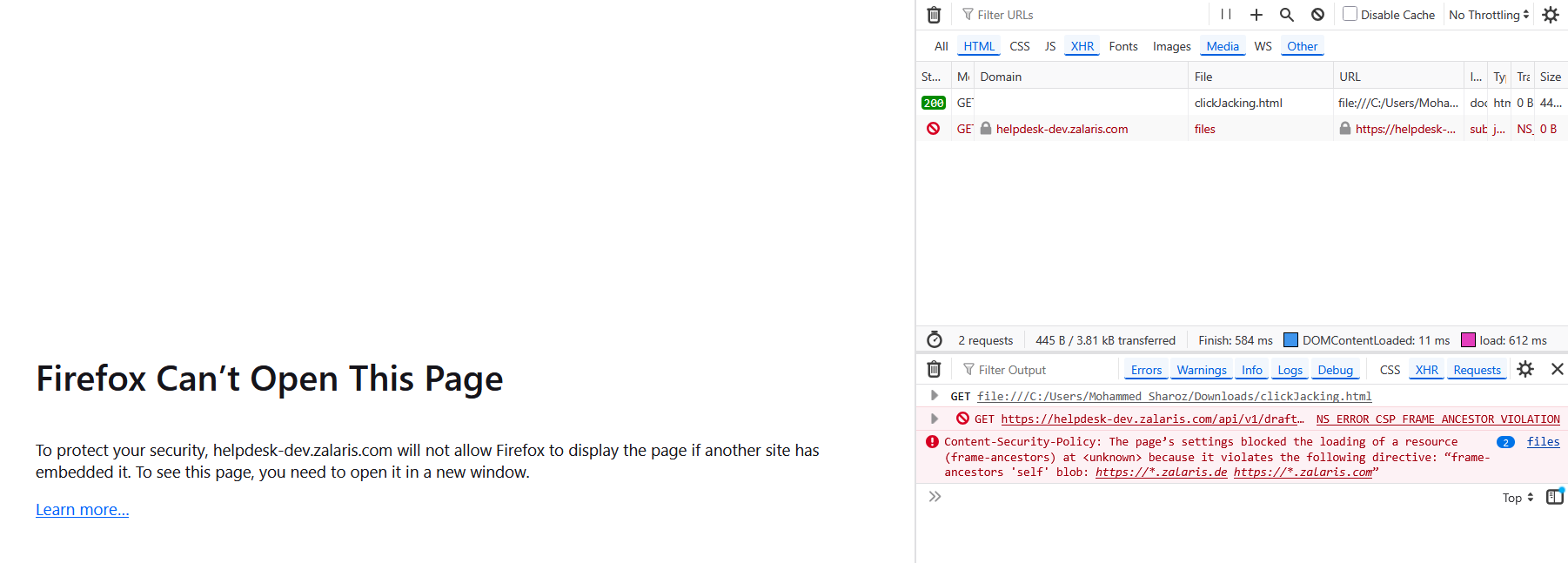
You should determine whether any functions accessible within frameable pages can be used by application users to perform any sensitive actions within the application.

**Links Tested**

|  |
| --- |
| * /510/tickets/new/sick-leave-certifikcate1 |

Steps:

* Access the path helpdesk-dev.zalaris.com/510
* Capture the request using burp proxy and analyze the response headers
* The application has X-Frame-Options header set with the value “DENY”
* Also the CSP has security policy set so that application does not load in a any frame whether the ancestors [current frame] that requests for he resource to be loaded in the iframe is not ‘zalaris.com’
* The screenshot for the same is attached below



* The content Security Policy for Help-desk is given below and the frame-ancestor value for the same is highlighted

|  |
| --- |
| Content-Security-Policy: default-src 'unsafe-inline' 'self' blob: \*.zalaris.de \*.zalaris.com https://s3.scriptcdn.net/cdn/c7fa7451-6f95-4815-ac32-b8cc2537837a https://login.microsoftonline.com/ https://dzudiostorage.blob.core.windows.net/ \*.b2clogin.com https://pzudiostorage.blob.core.windows.net/ https://northeurope-2.in.applicationinsights.azure.com/v2/track https://\*.in.applicationinsights.azure.com https://helpdesk-dev.zalaris.com https://alhvbcpn2.accounts.cloud.sap https://ar2hdcts7.accounts.cloud.sap/ mypayslip-dev.zalaris.com mypayslip.zalaris.com; script-src 'self' \*.zalaris.de \*.zalaris.com https://s3.scriptcdn.net/cdn/c7fa7451-6f95-4815-ac32-b8cc2537837a https://login.microsoftonline.com/ https://dzudiostorage.blob.core.windows.net/ \*.b2clogin.com https://cdn.form.io/ace/1.4.12/mode-html.js https://cdn.form.io/ace/1.4.12/mode-json.js https://cdn.form.io/flatpickr/flatpickr.min.js https://cdn.form.io/ace/1.4.10/ace.js https://cdn.form.io/ace/1.4.10/theme-xcode.js https://cdn.form.io/ace/1.4.10/mode-html.js https://cdn.form.io/ace/1.4.10/mode-json.js https://cdn.form.io/ace/1.4.10/worker-html.js https://cdn.form.io/ace/1.4.10/worker-json.js https://pzudiostorage.blob.core.windows.net/ https://northeurope-2.in.applicationinsights.azure.com/v2/track https://cdn.form.io/flatpickr-formio/4.6.13-formio.3/l10n/; img-src 'self' \*.zalaris.de \*.zalaris.com data: dzudiostorage.blob.core.windows.net https://pzudiostorage.blob.core.windows.net/; style-src 'unsafe-inline' 'self' \*.zalaris.de \*.zalaris.com https://s3.scriptcdn.net/cdn/c7fa7451-6f95-4815-ac32-b8cc2537837a https://dzudiostorage.blob.core.windows.net/ \*.b2clogin.com https://cdn.form.io/flatpickr/flatpickr.min.css https://pzudiostorage.blob.core.windows.net/ https://helpdesk-dev.zalaris.com/; media-src 'self' \*.zalaris.de \*.zalaris.com https://s3.scriptcdn.net/cdn/c7fa7451-6f95-4815-ac32-b8cc2537837a https://dzudiostorage.blob.core.windows.net/ \*.b2clogin.com https://pzudiostorage.blob.core.windows.net/ https://northeurope-2.in.applicationinsights.azure.com/v2/track; font-src 'self' \*.zalaris.de \*.zalaris.com https://s3.scriptcdn.net/cdn/c7fa7451-6f95-4815-ac32-b8cc2537837a https://alumnidev.b2clogin.com/alumnidev.onmicrosoft.com/ https://login.microsoftonline.com/ https://dzudiostorage.blob.core.windows.net/ data: https://pzudiostorage.blob.core.windows.net/ https://northeurope-2.in.applicationinsights.azure.com/v2/track; frame-src 'self' blob: \*.zalaris.de \*.zalaris.com https://s3.scriptcdn.net/cdn/c7fa7451-6f95-4815-ac32-b8cc2537837a https://login.microsoftonline.com/ https://dzudiostorage.blob.core.windows.net/ \*.b2clogin.com https://pzudiostorage.blob.core.windows.net/ https://view.officeapps.live.com/op/embed.aspx https://northeurope-2.in.applicationinsights.azure.com/v2/track mypayslip.zalaris.com mypayslip-dev.zalaris.com https://dzudiostorage.blob.core.windows.net https://zudioinnomotics.blob.core.windows.net/ https://view.officeapps.live.com blob:https://mypayslip.zalaris.com; frame-ancestors 'self' blob: \*.zalaris.de \*.zalaris.com; form-action 'self' \*.zalaris.de \*.zalaris.com https://s3.scriptcdn.net/cdn/c7fa7451-6f95-4815-ac32-b8cc2537837a https://login.microsoftonline.com/ https://dzudiostorage.blob.core.windows.net/ \*.b2clogin.com https://northeurope-2.in.applicationinsights.azure.com/v2/track; |

**WORKLOG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| When | What | Who | Recommended Action | Action plan |
| 11 June 2025 | Frame-able Response (Potential Click-jacking) is manually tested | CSOC | No action needed. The following security Measures are set :  Frame ancestors set to load helpdesk only if the ancestor / current page where the frame is loaded is belongs to zalaris  The X-Frame-Options header are set to the value “DENY” in most pages  The configuration Frame ancestors together with the X-Frame-Options header prevents the site from loading inside an iframe | No Action Needed |

### HTML 5 Storage Manipulation(DOM Based):

**Category:** Low

|  |  |
| --- | --- |
| Type | False Positive |
| Status | No action needed |

**Description**

DOM-based vulnerabilities arise when a client-side script reads data from a controllable part of the DOM (for example, the URL) and processes this data in an unsafe way.

HTML5 storage manipulation arises when a script stores controllable data in the HTML5 storage of the web browser (either localStorage or sessionStorage). An attacker may be able to use this behavior to construct a URL that, if visited by another application user, will cause the user's browser to store attacker-controllable data.

This behavior does not in itself constitute a security vulnerability. However, if the application later reads the data back from storage and processes it in an unsafe way, then an attacker may be able to leverage the storage mechanism to deliver other DOM-based attacks, such as cross-site scripting and JavaScript injection.

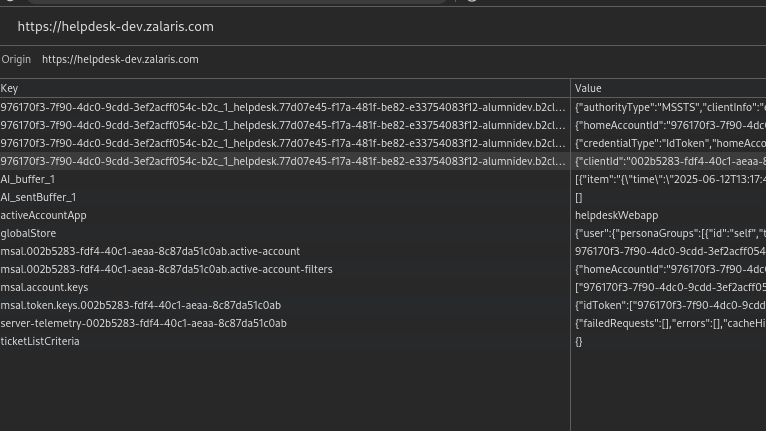
Burp Suite automatically identifies this issue using dynamic and static code analysis. Static analysis can lead to false positives that are not actually exploitable. If Burp Scanner has not provided any evidence resulting from dynamic analysis, you should review the relevant code and execution paths to determine whether this vulnerability is indeed present, or whether mitigations are in place that would prevent exploitation.

**Links Tested**

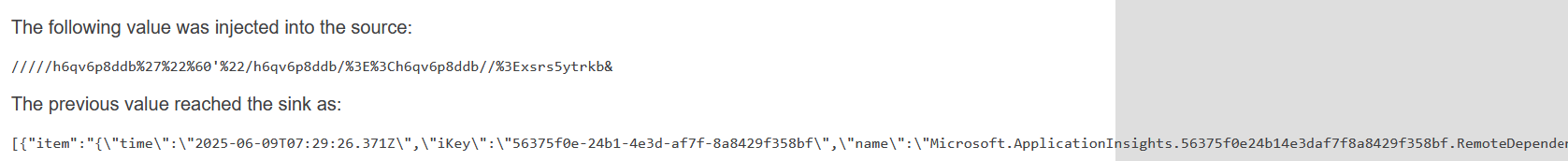
|  |
| --- |
| * / * /510 * /510/tickets/new/additional-tax-deduction * /510/tickets/new/draft-template-testing * /510/tickets/new/einstellungsbogen * /510/tickets/new/employee-transfer * /510/tickets/new/ordering-an-employment-certificate-requsition-form * /510/tickets/new/schedule-an-expert-inquiry * /777 * /777/tickets/new/additional-tax-deduction * /777/tickets/new/employee-transfer |

Steps:

* Login into helpdesk-dev.zalaris.com
* Access the browser’s local storage and check whether the application has items stored in local storage and session storage
* The screenshot below shows that the application has local storage and session storage enabled



* The value that is sent from the source to the sink is targeted to be sent to the azure insights and is not processed back by the helpdesk application . The screenshot from the automated audit is attached below [value from location.pathname passed onto sessionStorage.setItem ]



* The key value of azure insights present in the session storage is given below

|  |
| --- |
| **AI\_buffer\_1** : [{"item":"{\"time\":\"2025-06-12T14:28:10.455Z\",\"iKey\":\"56375f0e-24b1-4e3d-af7f-8a8429f358bf\",\"name\":\"Microsoft.ApplicationInsights.56375f0e24b14e3daf7f8a8429f358bf.RemoteDependency\",\"tags\":{\"ai.user.id\":\"q47P128xC9LyKIrgDpFB6t\",\"ai.session.id\":\"+tGUt703hc424qXKMLcM/y\",\"ai.device.id\":\"browser\",\"ai.device.type\":\"Browser\",\"ai.operation.name\":\"/510/tickets\",\"ai.operation.id\":\"a7f8feb9a6f34197a3076e9730ce5f7e\",\"ai.cloud.role\":\"helpdesk-webapp\",\"ai.internal.sdkVersion\":\"javascript:3.3.7\"},\"data\":{\"baseType\":\"RemoteDependencyData\",\"baseData\":{\"id\":\"|a7f8feb9a6f34197a3076e9730ce5f7e.2668600240024ee9.\",\"ver\":2,\"name\":\"GET https://helpdesk-dev.zalaris.com/api/v1/notifications?contactId=1000113\",\"resultCode\":\"200\",\"duration\":\"00:00:00.345\",\"success\":true,\"data\":\"GET https://helpdesk-dev.zalaris.com/api/v1/notifications?contactId=1000113\",\"target\":\"helpdesk-dev.zalaris.com\",\"type\":\"Ajax\",\"properties\":{\"HttpMethod\":\"GET\"},\"measurements\":{}}}}","cnt":0}] |

* The value AI\_Buffer\_1 is the key and the subsequent value after “:” is where the data from the sink goes to
* This data is not processed by the help-desk application again and has no means to alter the applications GUI or run scripts
* Therefore the reported alert can be considered as False Positive

**WORKLOG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| When | What | Who | Recommended Action | Action plan |
| 13 June 2025 | Manually tested the HTML5 Storage Manipulation (DOM based Attack) | CSOC | False Positive . No action needed | False Positive . No action needed |

### TLS Certificate:

**Category:** Low

|  |  |
| --- | --- |
| Type | Informational alert |
| Status | TLS Information are manually verified. No issues are found |

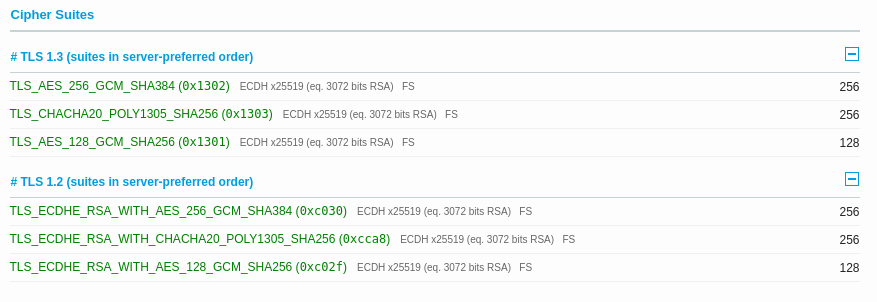
**Description**

The server presented a valid, trusted TLS certificate. This issue is purely informational.

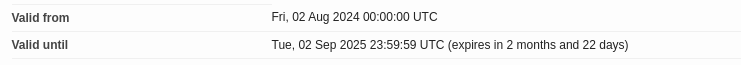
**Steps:**

* Access Qualys ssl labs : <https://www.ssllabs.com/ssltest/analyze.html?d=helpdesk-dev.zalaris.com>
* The following must be checked from the result obtained
  1. Cipher Suite Strength
  2. Certificate Validity
  3. Chain Issues

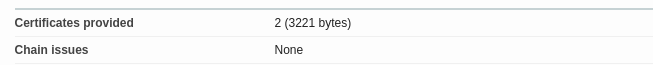
The cipher suite strength has no issues screenshot for the same is attached below



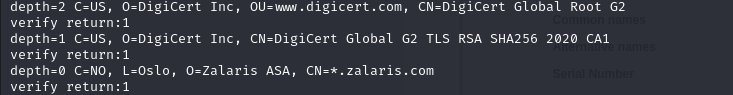
The Certificate has validity set till September 2nd 2025



No Chain issues are found .



Certificate chain verified via open\_ssl command no issue found. Screenshot for the same is attached below



**WORKLOG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| When | What | Who | Recommended Action | Action plan |
| 10 June 2025 | Manually analyzed the TLS configuration for issues | CSOC | No Issues Found | No action needed |