Manulife

# PENETRATION TEST REPORT

## GDT – manulife.com – 2025 New Test

| | |
|---|---|
| **Date** | 11/14/2025 |
| **Prepared By** | Global Cybersecurity |
| **Application** | Manulife.com |
| **Delivered To** | Sakib Sadat |

Manulife

## Table of Contents

Manulife

# 1. DOCUMENT REVISION HISTORY

This section of the report contains information about document versioning. Versioning information is displayed in Table 1 below. There will typically be re-tests performed after initial testing to confirm if findings have been remediated. Future reports will contain an updated Document Revision History table.

*Table 1: Document Revision History*

| Version | Modification | Date | Author |
|---|---|---|---|
| 1.0 | New Report | 14-Nov-25 | Gemirald Emil De Guia Macaraig |

# 2. EXECUTIVE SUMMARY

The Global Cybersecurity team was requested by the Manulife.com web application team to perform a manual penetration test against their web application.

The objective of the engagement was to find any vulnerabilities present in the web application and offer remediation steps to the Manulife.com web application team.

The findings of this test can be found in Section 6. Findings Summary.

## 3. PENETRATION TEST METHODOLOGY

There are three different types of testing that may be carried out by the Global Cybersecurity Team. Typically for web application penetration tests it will be Gray-Box testing, where some level of access is provided to the web application, but the details of a web application are not necessarily disclosed. The three types of testing are illustrated below.

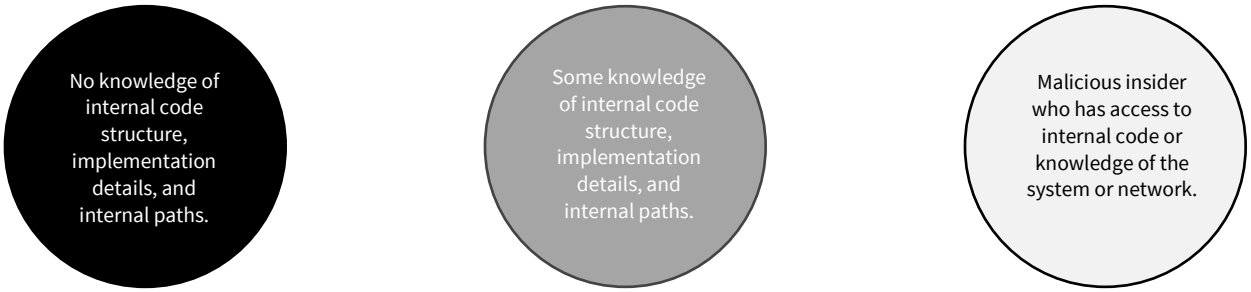


*Figure 1 An illustration which describes Black-Box, Grey-Box, and White-Box testing. The test methodologies are least-privilege to most-privilege from left to right.*

The Global Cybersecurity Team's goal is to use a variety of scripts, tools, and methodologies to carry out a series tests to find security vulnerabilities in the web application.

Depending on the type of test, the team may or may not be able to perform intrusive testing (such as active scanning) against the site. Some tools used in the assessment involve web proxies (Burpsuite Pro), scanners, fuzzers, and other enumeration techniques. Many of these tools can be found packaged in security testing distributions such as Kali Linux or ParrotOS.

The attacks typically come in the form of modified web requests, modified URL strings or form payloads - brute-forcing login forms or carrying out one of the other vulnerabilities listed in the OWASP Top 10. More detail on those vulnerabilities can be found here: https://owasp.org/www-project-top-ten/. Denial of service attacks are not typically carried out in penetration testing.

Information about the penetration testing policies can be found in Table 2 below.

*Table 2: Manulife Policy for the Application Security Testing Standard.*

| Resource | URL |
|---|---|
| Application Security Testing Standard (STA-023) | https://mfc.sharepoint.com/sites/NewGPD/Policy%20Documents/Application%20Security%20Standard%20-%20MFC-STA-023-EN.pdf |

# 4. RISK ASSESSMENT METHODOLOGY

## 4.1. SEVERITY

The **severity** assigned to each vulnerability was calculated using the NIST 800-30 Revision 1 standard. This standard acts as a guideline for evaluating the risk posed to an application. This calculation is based on the **likelihood** an attacker exploits the vulnerability, the **impact** that it would have on the business, any mitigating controls to reduce the likelihood of exploitation, and accessibility to attackers.

## 4.2. LIKELIHOOD

The Likelihood is a measurement of the capacity to carry out an attack. The factors will be the difficulty or skill required to carry out the attack, the level of access an attacker has within the web application, and the amount of access required to reach the vulnerability. The likelihood can fall into one of the following risk strata:

*Table 3: Likelihood Classification Matrix*

| Risk Category | Description |
|---|---|
| Critical | An attacker is near-certain to carry out the threat event. |
| High | An untrained user could exploit the vulnerability. The vulnerability is obvious or easily accessed. |
| Medium | The vulnerability requires some hacking knowledge to carry out the attack, or access to the vulnerability requires a greater level of access than users normally have. |
| Low | The vulnerability requires significant time, skill, access, and other resources to carry out an attack. |
| Informational | Attackers are highly unlikely or unable to leverage the vulnerability in an attack. |

## 4.3. IMPACT

The Impact is a measurement of the adverse effects carrying out an attack would have on the organization. The factors in evaluating impact are primarily on the level of service interruption (Recovery Point Objective and Recovery Time Objective), the potential of data loss, and the potential of sensitive data exposure.

*Table 4: Impact Classification Matrix*

| Risk Category | Description |
|---|---|
| Critical | An attack would cause catastrophic or severe effect(s) on operations, assets, or other organizations. |
| High | An attack would severely degrade mission capability. The attack may result in damage to assets (data exposure or loss). The organization may not be able to perform primary functions. |
| Medium | An attack would degrade mission capability. The attack may result in a damage to assets (data exposure or loss). An attack would allow for primary functions of the application to resume, but at reduced effectiveness. |
| Low | An attack would degrade mission capability in a limited capacity. The attack may result in marginal damage to assets (data exposure or loss). An attack would allow primary functions of the application to resume, but at reduced effectiveness. |
| Informational | An attack would have negligible adverse effect on the organization, or no effect. |

Manulife

## 5. REMEDIATION TIMELINES

The following remediation timelines can be found within the Application Penetration Testing Standard (**Table 2**). For convenience, it has been included in this report on the following page.

*Table 5: A summary of the Remediation Timeline Table found within the Application Penetration Testing Standard. URL provided in Table 2 of this report.*

<table>
<tr><th rowspan="2">Finding Severity Rating</th><th rowspan="2">Non-Production</th><th colspan="2">Production</th></tr>
<tr><th>External</th><th>Internal</th></tr>
<tr><td>Emergency</td><td>Not Applicable</td><td colspan="2">If any critical finding is declared as emergency by the CIRO based on his / her discretion then in that “Emergency” situation, a Security Incident-1 must be declared by Incident Management Team within one (1) hour of the declaration and the incident must remain open until the closure of the finding.<br><br>If the finding that was declared as “Emergency” cannot be remediated within four (4) hours of the declaration, then a blocking Web Application Firewall (WAF) rule must be implemented within 12 hours to mitigate the vulnerability.<br><br>o If the established timeline (12 hours) cannot be met due to technical limitations, a fully documented action plan must be provided.<br><br>The remediation deadline can be extended up to 30 days from date of the “Emergency” declaration or the next release if sooner when a blocking Web Application Firewall (WAF) rule is implemented to mitigate the vulnerability</td></tr>
<tr><td>Critical</td><td>Remediate before production launch</td><td>Remediate within 10 days of finding confirmation.<br><br>The remediation deadline can be extended up to 30 days from date of vulnerability identification or the next release if sooner when a blocking Web Application Firewall (WAF) rule is</td><td>Mitigate or remediate within 21 days upon vulnerability identification.</td></tr>
</table>

| | | | |
|---|---|---|---|
| | | implemented to mitigate the vulnerability. | |
| **High** | Remediate before production launch | Remediate **within 40 days** of vulnerability identification.<br><br>The remediation deadline can be extended **up to 90 days** from date of vulnerability identification or the next release if sooner when a blocking Web Application Firewall (WAF) rule is implemented to mitigate the vulnerability. | |
| **Medium** | Remediate before production launch | Remediate **within 90 days** of vulnerability identification.<br><br>The remediation deadline can be extended **up to 180 days** from date of vulnerability identification when a blocking Web Application Firewall (WAF) rule is implemented to mitigate the vulnerability. | |
| **Low** | Remediate within 365 days of vulnerability identification for penetration test. | | |
| **Insignificant / Informational** | No Action Required | | |

# 6. MANULIFE.COM PENTEST

## 6.1. TARGETS AND TEST DEPENDENCIES

The target for the testing was provided by the application owner, Sakib Sadat, or by one of the team members of the Manulife.com web application. The web application is hosted at the following URL(s):

*Table 6: The URL(s) used by the Manulife.com web application.*

| Application Name | URL(s) in Scope |
|---|---|
| Manulife.com | stage.manulife.com |

Accounts were provisioned for the following roles:

*Table 7: Role-Based Access Levels provisioned during the penetration test (if applicable)*

| User Role | Username |
|---|---|
| Unauthenticated User | N/A |

*Table 8: API Routes tested (if applicable)*

| API Routes |
|---|
| N/A |

## 6.2. FINDINGS SUMMARY

This section contains a summary of the findings, their rating, and their status. These findings are detailed in the next section.

If you are looking for a cheat sheet for web security, this resource covers many of the recommended settings when building your projects:
https://cheatsheetseries.owasp.org/index.html

*Table 9: Summary of findings for Manulife.com.*

| Findings | Likelihood | Impact | Rating | ID | Status |
|---|---|---|---|---|---|
| Remote Update Manager Exposed | Low | Low | Low | 66 | Open (New) |
| Verbose Error Messages | Low | Low | Low | 67 | Open (New) |
| Insecure Methods Allowed | Low | L8ow | Low | 68 | Open (New) |
| Missing/Misconfigured Security Header: Permissions-Policy | Low | Low | Low | 69 | Open (New) |
| Missing/Misconfigured Security Header: X-Permitted- | Low | Low | Low | 70 | Open (New) |

| | | | | | |
|---|---|---|---|---|---|
| Cross-Domain-Policies | | | | | |
| Missing/Misconfigured Security Header: Access-Control-Allow-Origin | Low | Low | Low | 71 | Open (New) |
| Missing/Misconfigured Content-Security-Policy Header | Low | Low | Low | 63 | Open (New) |
| Missing/Misconfigured Referrer-Policy Header | Low | Low | Low | 72 | Open (New) |

# 7. LOW FINDINGS

## 7.1. REMOTE UPDATE MANAGER EXPOSED

**Status:** Open (New)
**Severity:** Low
**Location:**

- https://stage.manulife.com/.rum/100

**Description:**
The vulnerability arises from the exposure of the /.rum/ endpoint of the Adobe Remote Update Manager (RUM). This endpoint is accessible over the network and consistently responds with a "201 Created" status, indicating that operations are being accepted without proper authentication or authorization checks. This exposure could potentially allow unauthorized users to initiate update processes or manipulate update settings, potentially leading to unauthorized software updates or installations.

**Recommended Remediation:**
Configure network firewalls and access control lists to restrict access to the /.rum/ endpoint to only authorized internal IP addresses.

**Proof of Concept:**

1. Browse in the application.
2. Notice that when browsing, the client sends a request to /.rum/ with a 201 response.

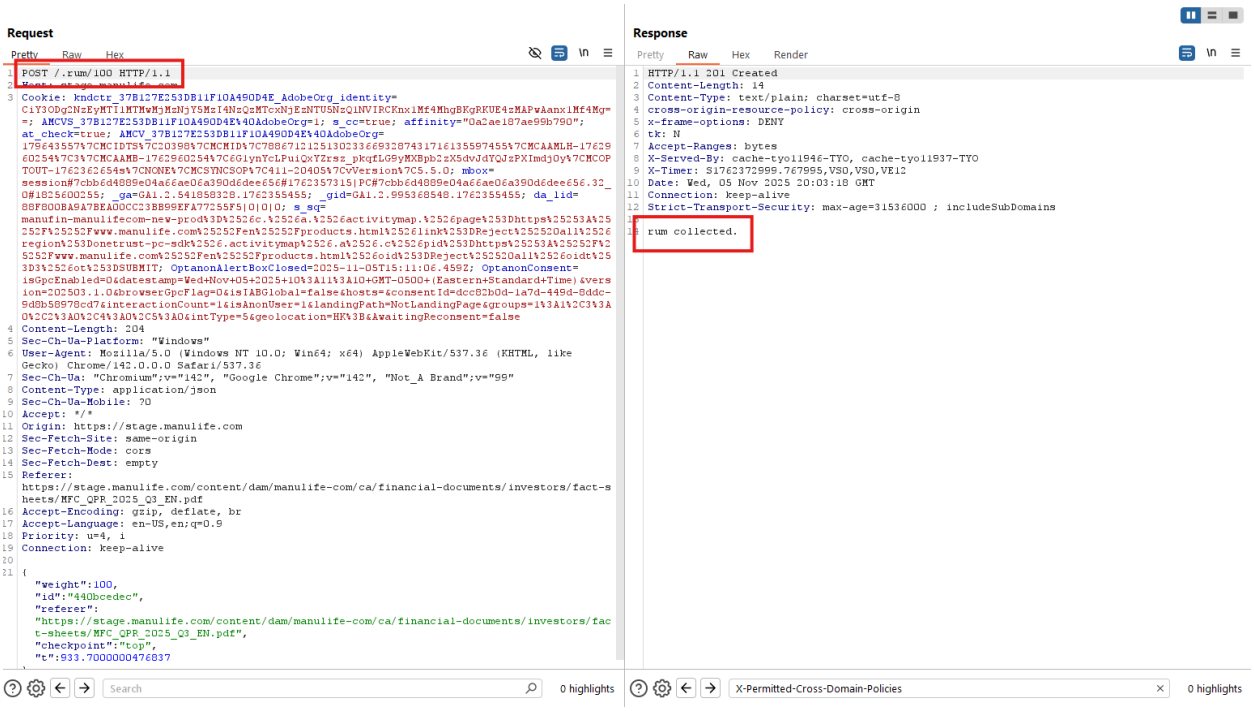


*Figure 2. /.rum/100 directory in the request.*

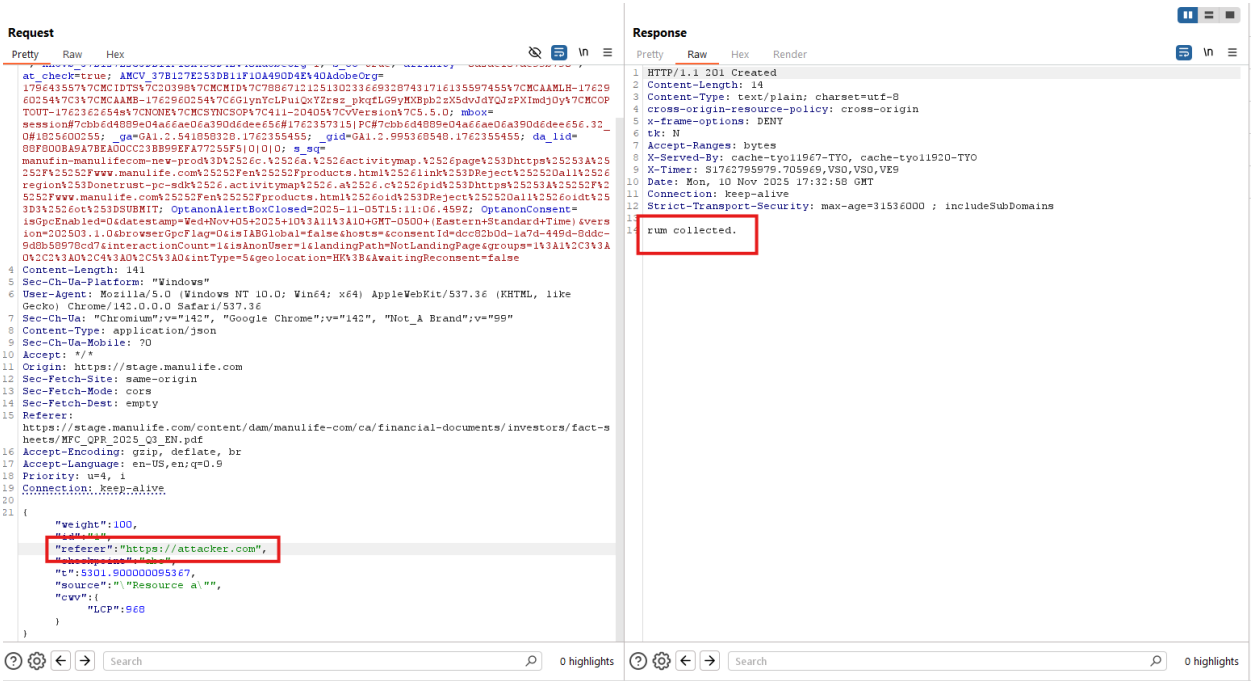


*Figure 3. Arbitrary referrer value in request.*

**Note:**
In the request body, any string can be used as an input but there is a generic response with “rum collected.”

In general, it is advisable to avoid exposing such endpoints directly to the public unless it is necessary and ensure that any exposed services are secured according to best practices.

## 7.2. VERBOSE ERROR MESSAGES

**Status:** Open (New)
**Severity:** Low
**Location:**

- Across the application (This vulnerability is systemic)

**Description:**
Verbose error messages have been detected on at least one web server within the tested environment. Detailed error reporting should never be provided on a production application. Error messages give very useful information to an attacker about the application. Error messages can give out the following information:

- The language it was developed in, such as PHP, Java, C# or VB.NET.
- The stack trace of the program that failed.
- The version numbers for the framework in use.
- Database connection strings and/or queries.
- Development class names and object structures including application paths.

**Recommended Remediation:**
Display generic error message for all endpoints.

**Proof of Concept:**

1. Open and browse on the application.
2. Force an error by browsing on a non-existent location.

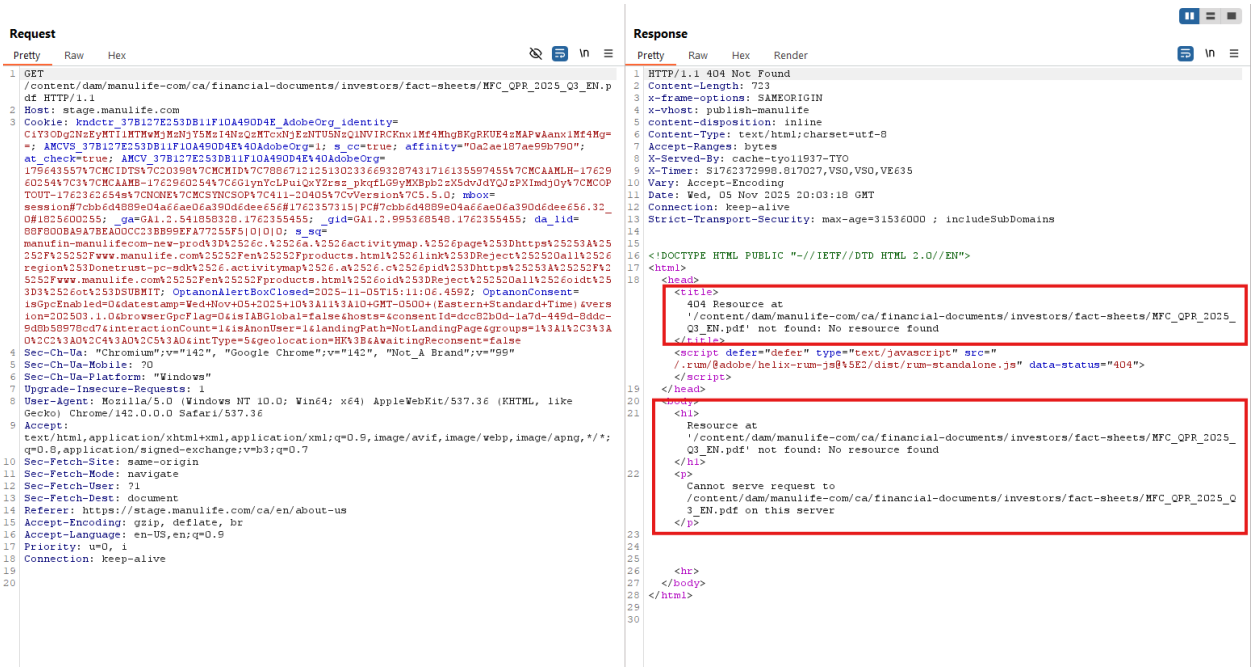


*Figure 4. Error message that displays application path.*

## 7.3. INSECURE METHODS ALLOWED

**Status:** Open (New)
**Severity:** Low
**Location:**

- Across the application (This vulnerability is systemic)

**Description:**
The website supports the OPTIONS and TRACE verbs to be sent. This will disclose which other verbs are accepted by the application and is used to gather more information about what web requests an application will accept. OPTIONS and TRACE should be disabled.

**Recommended Remediation:**
Disable the insecure verbs from being accepted in web requests to the server. This is a web server configuration issue.

**Proof of Concept:**

1. Browse in the application.
2. Change the verb/method from GET or POST to OPTIONS or TRACE in the request header.
3. Notice in the response it displays “Allow” header that has several verbs or methods that the application uses.

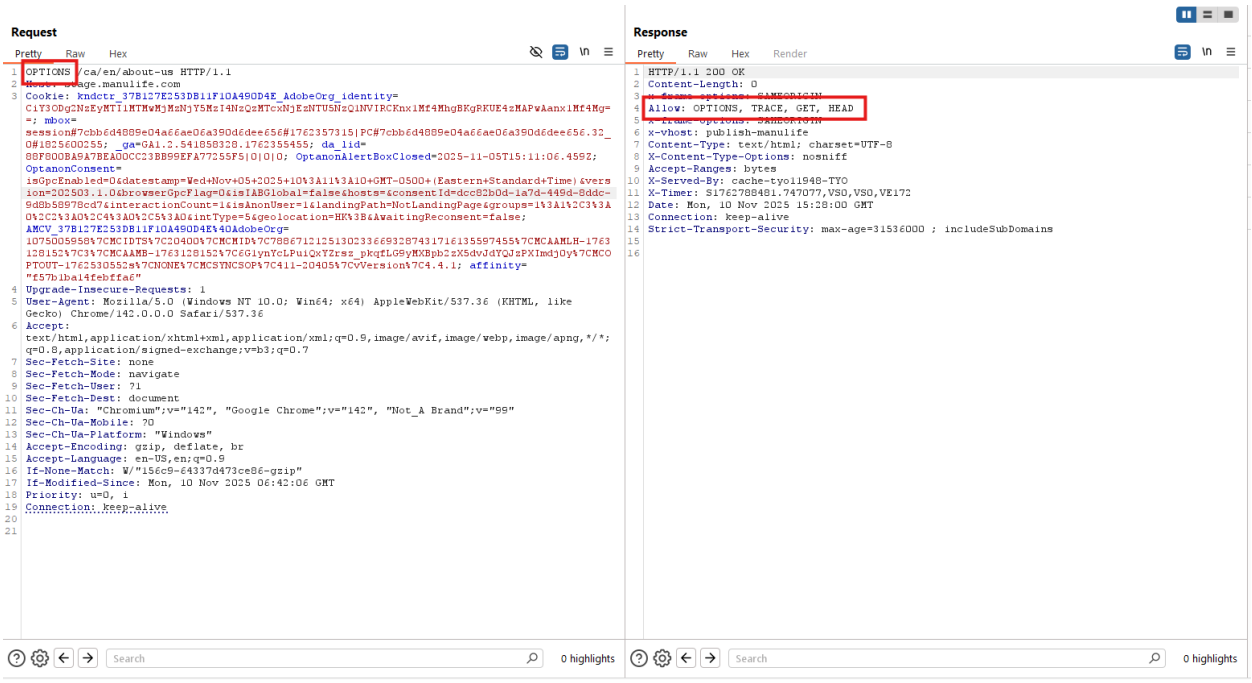


*Figure 5. OPTIONS method allowed.*

**Note:**
The TRACE Method is being blocked by the firewall, but it is still listed as allowed header in the “Allow” header.

## 7.4. MISSING/MISCONFIGURED SECURITY HEADER: PERMISSIONS-POLICY

**Status:** Open (New)
**Severity:** Low
**Location:**

- Across the application (This vulnerability is systemic)

**Description:**
Permissions Policy provides mechanisms for web developers to explicitly declare what functionality can and cannot be used on a website. You define a set of "policies" that restrict what APIs the site's code can access or modify the browser's default behavior for certain features. This allows you to enforce best practices, even as the codebase evolves — as well as more safely compose third-party content.

Permissions Policy is similar to Content Security Policy but controls features instead of security behavior.

The web provides functionality and APIs that may have privacy or security risks if abused. In such cases, you may wish to strictly limit how functionality is used on a website. In each case, there should be an intuitive or non-breaking way for web developers to detect and handle cases where a feature is disabled.

**Recommended Remediation:**

| Header Name | Recommended Header Value |
|---|---|
| Permissions-Policy | accelerometer=(),ambient-light-sensor=(),autoplay=(),battery=(),camera=(),display-capture=(),document-domain=(),encrypted-media=(),fullscreen=(),gamepad=(),geolocation=(),gyroscope=(),layout-animations=(self),legacy-image-formats=(self),magnetometer=(),microphone=(),midi=(),oversized-images=(self),payment=(),picture-in-picture=(),publickey-credentials-get=(),speaker-selection=(),sync-xhr=(self),unoptimized-images=(self),unsized-media=(self),usb=(),screen-wake-lock=(),web-share=(),xr-spatial-tracking=() |

**Proof of Concept:**

1. Browse in the indicated location.
2. Observe that the header is missing in the response.

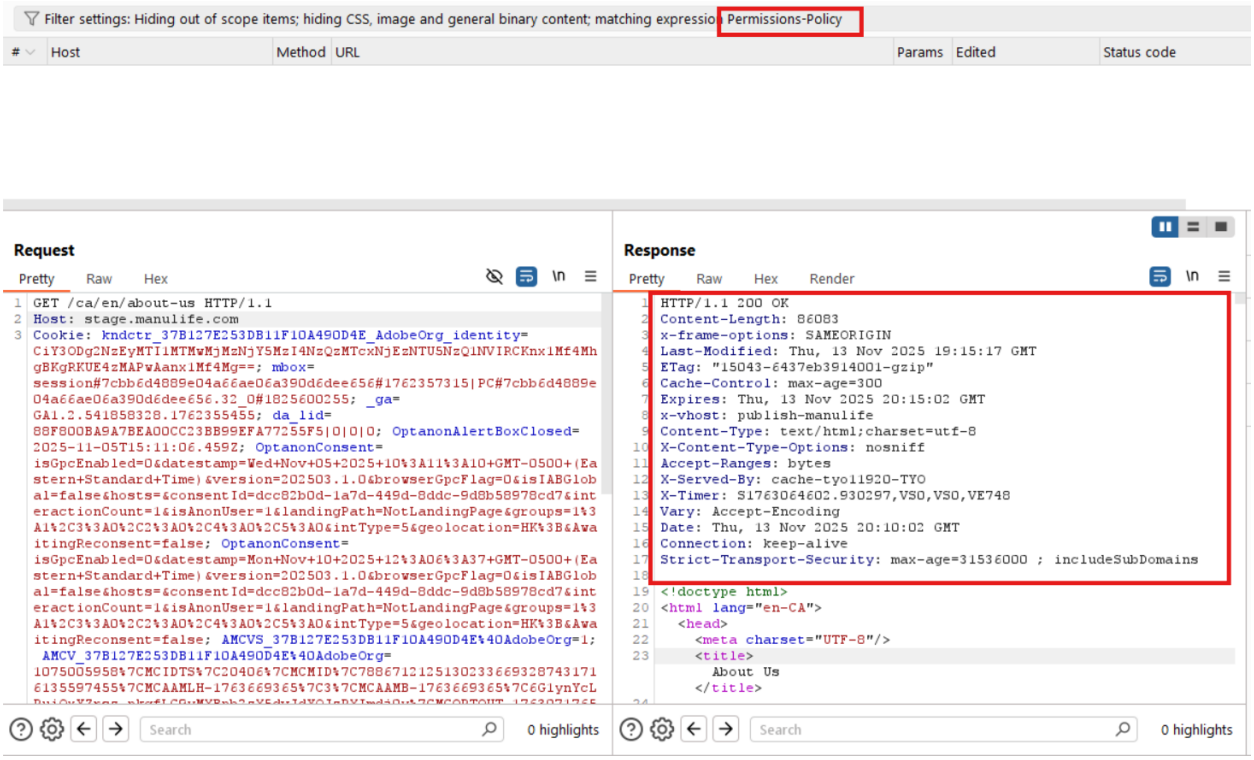


*Figure 6. Missing header in Staging environment.*

## 7.5. MISSING/MISCONFIGURED SECURITY HEADER: X-PERMITTED-CROSS-DOMAIN-POLICIES

**Status:** Open (New)
**Severity:** Low
**Location:**

- Across the application (This vulnerability is systemic)

**Description:**
This header is used to limit which data external resources, such as Adobe Flash and PDF documents, can access on the domain. Failure to set the X-Permitted- Cross-Domain-Policies header to 'none' value allows other domains to embed the application's data in their content.

If there is no requirement to load application data within web clients such as Adobe Flash Player or Adobe Acrobat (not limited to these), then the header should be configured according to the Remediation guideline below.

**Recommended Remediation:**

| Header Name | Recommended Header Value |
|---|---|
| X-Permitted-Cross-Domain-Policies | none |

**Proof of Concept:**

1. Browse in the indicated location.
2. Observe that the header is missing in the response.

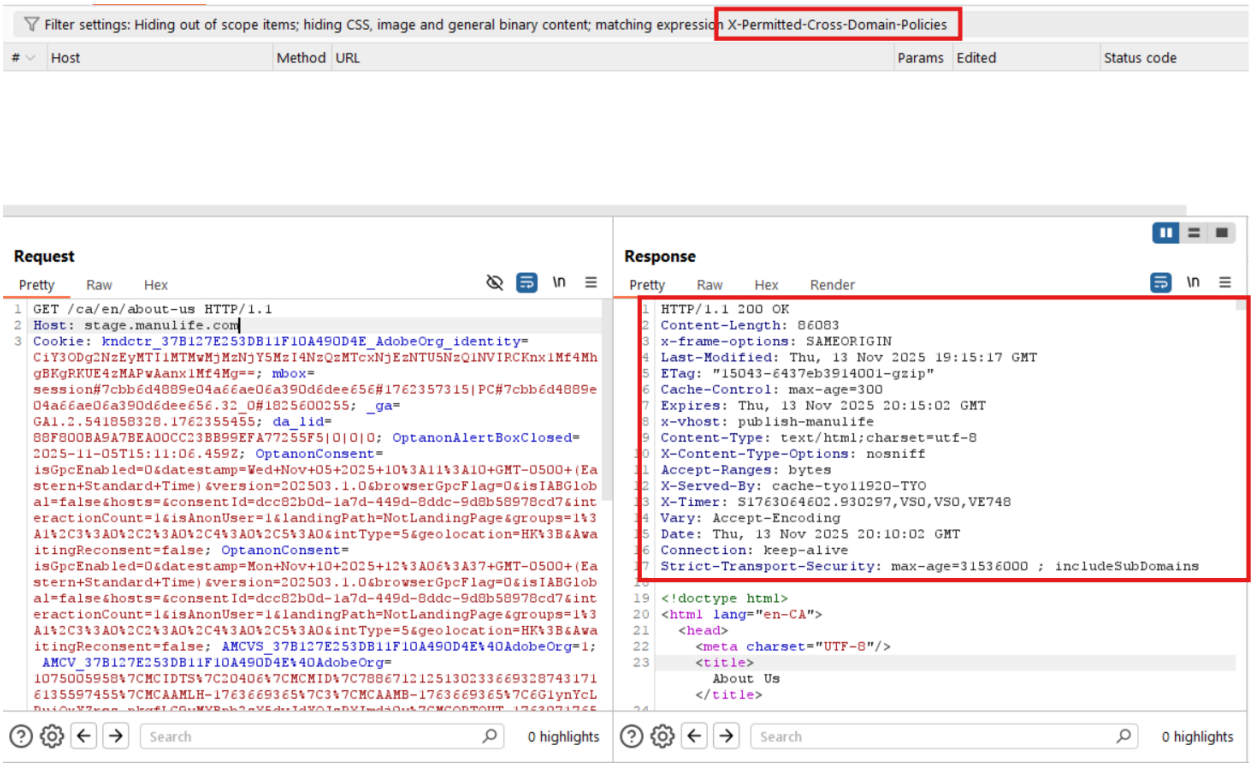


*Figure 7. Missing header in Staging environment.*

## 7.6. MISSING/MISCONFIGURED SECURITY HEADER: ACCESS-CONTROL-ALLOW-ORIGIN

**Status:** Open (New)
**Severity:** Low
**Location:**

- Across the application (This vulnerability is systemic)

**Description:**
Cross-Origin Resource Sharing (CORS) policy controls whether and how content running on other domains can perform two-way interaction with the domain that publishes the policy.

If another domain is allowed by the policy, then that domain can potentially attack users of the application. If a user is logged in to the application, and visits a domain allowed by the policy, then any malicious content running on that domain can potentially retrieve content from the application and sometimes carry out actions within the security context of the logged in user.

**Recommended Remediation:**

- Configure the Access-Control-Allow-Origin header to allow requests only from the domain that you trust or from same domain
- Validate the origin header
- Rather than using a wildcard or programmatically verifying supplied origins, use a whitelist of trusted domains.

**Proof of Concept:**

1. Browse in the indicated location.
2. Observe that the header has a wildcard value.

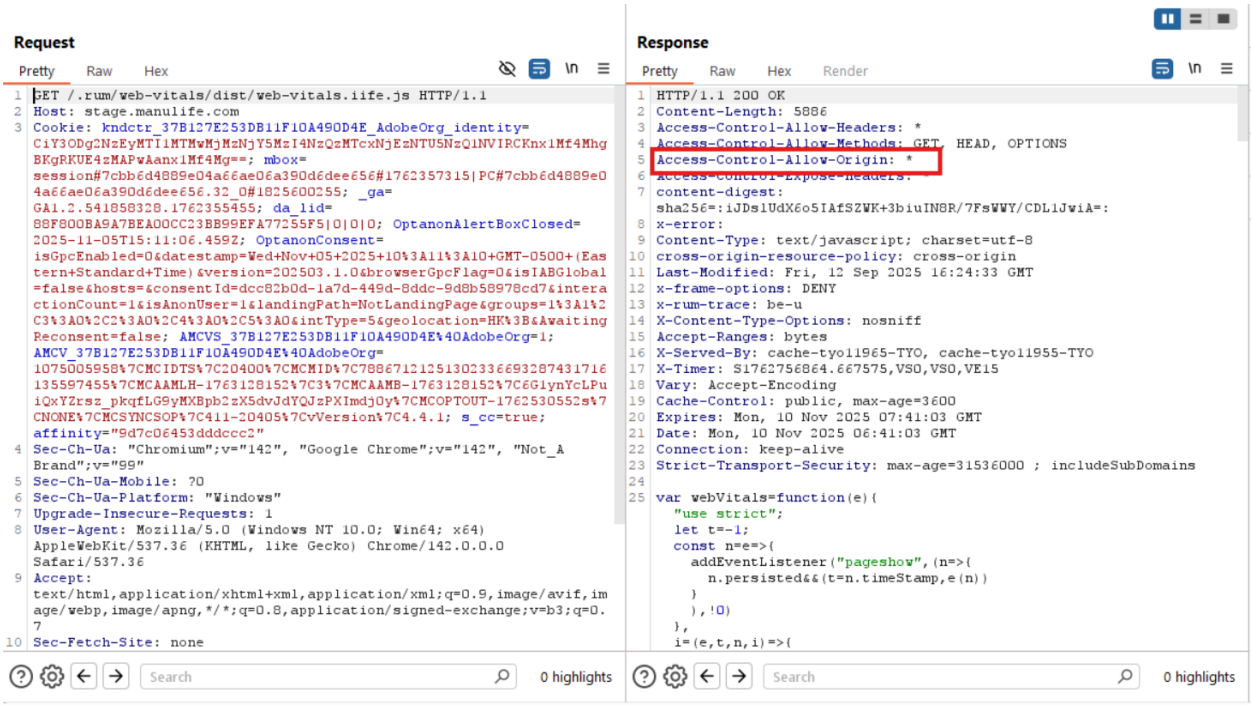


*Figure 8. ACAO with * in response.*

## 7.7. MISSING/MISCONFIGURED SECURITY HEADER: CONTENT-SECURITY-POLICY

**Status:** Open (New)
**Severity:** Low
**Location:**

- Across the application (This vulnerability is systemic)

**Description:**
CSP is a browser security mechanism that aims to mitigate XSS and some other attacks. It works by restricting the resources (such as scripts and images) that a page can load and restricting whether a page can be framed by other pages.

To enable CSP, a response needs to include an HTTP response header called Content-Security-Policy with a value containing the policy. The policy itself consists of one or more directives, separated by semicolons.

**Recommended Remediation:**

| Header Name | Recommended Header Value |
|---|---|
| Content-Security-Policy | Any value getting all green in CSP Evaluator |

**Proof of Concept:**

1. Browse in the indicated location.
2. Observe that the header is missing in the response.

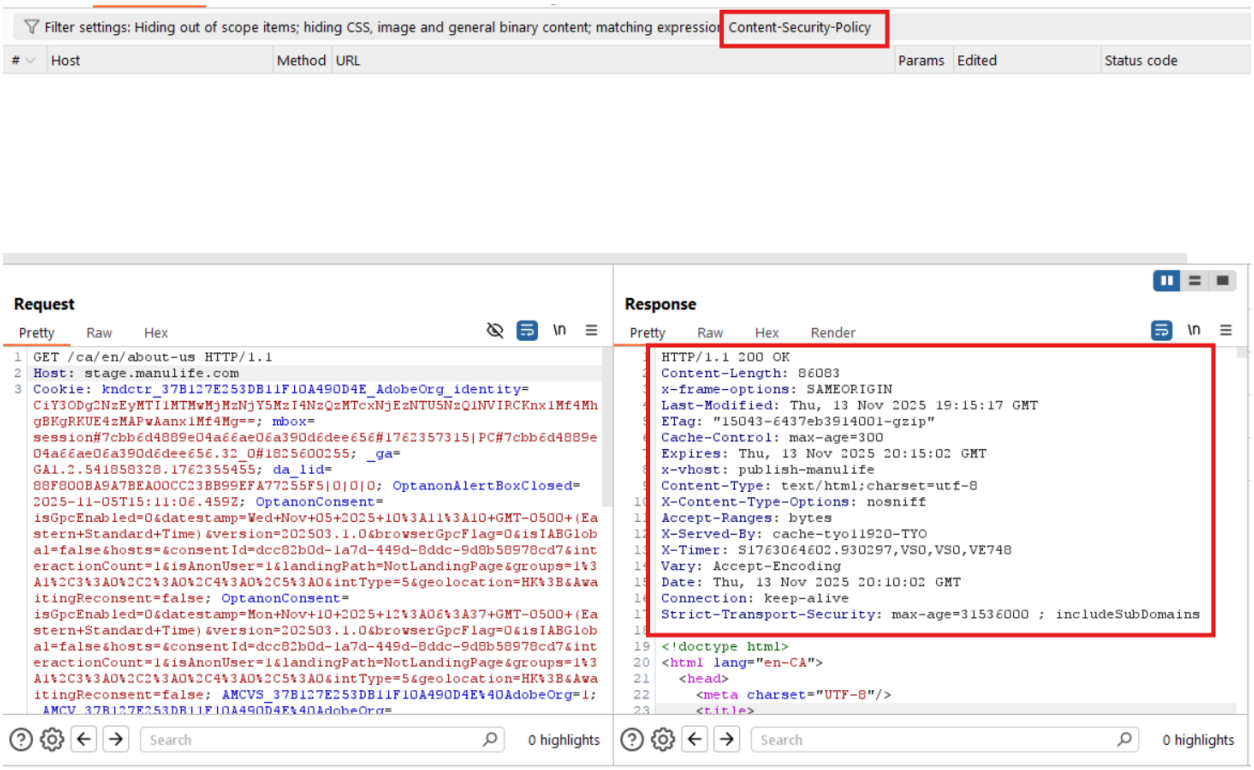


*Figure 9. Missing header in Staging environment.*

## 7.8. MISSING/MISCONFIGURED SECURITY HEADER: REFERRER-POLICY

**Status:** Open (New)
**Severity:** Low
**Location:**

- Across the application (This vulnerability is systemic)

**Description:**
This header protects against Cross-domain Referrer leakage. When a web browser makes a request for a resource, it typically adds an HTTP header, called the "Referrer" header, indicating the URL of the resource from which the request originated. This occurs in numerous situations, for example when a web page loads an image or script, or when a user clicks on a link or submits a form.

If the resource being requested resides on a different domain, then the Referrer header is still generally included in the cross-domain request. If the originating URL contains any sensitive information within its query string, such as a session token, then this information will be transmitted to the other domain. If the other domain is not fully trusted by the application, then this may lead to a security compromise.

You should review the contents of the information being transmitted to other domains, and also determine whether those domains are fully trusted by the originating application.

Today's browsers may withhold the Referrer header in some situations (for example, when loading a non-HTTPS resource from a page that was loaded over HTTPS, or when a Refresh directive is issued), but this behavior should not be relied upon to protect the originating URL from disclosure.

Note also that if users can author content within the application then an attacker may be able to inject links referring to a domain they control in order to capture data from URLs used within the application.

**Recommended Remediation:**

| Header Name | Recommended Header Value |
|---|---|
| Referrer-Policy | no-referrer |

**Proof of Concept:**

1. Browse in the indicated location.
2. Observe that the header is missing in the response.

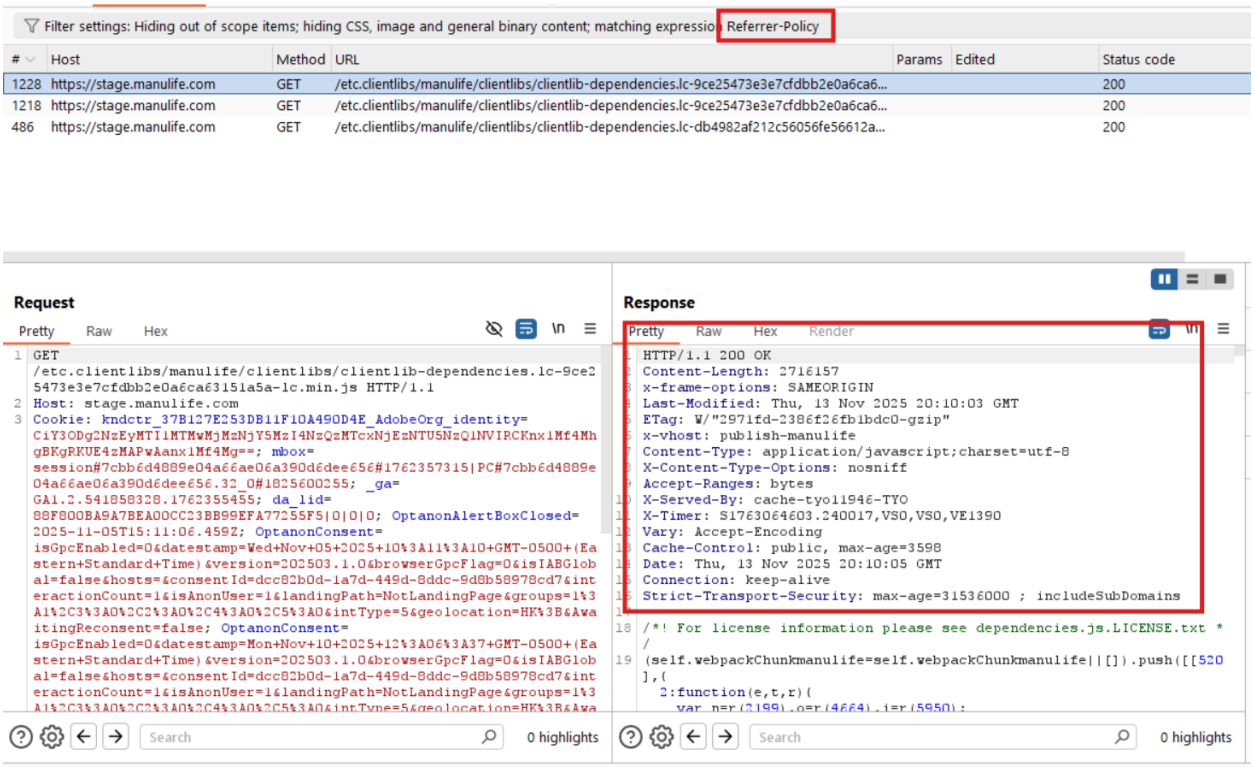


*Figure 10. Missing header in Staging environment.*