securitum

Security report

SUBJECT

Kobalt Languages client portal (build pentest-retest-1, commit 46a00dc)

DATE

14.07.2026 – 22.07.2026

RETEST DATE

11.08.2026

LOCATION

Poznań (Poland)

AUTHOR

Karol Królak

VERSION

1.1

securitum

Executive summary

This document is a summary of work conducted by Securitum. The subject of the test was the Kobalt Languages pentest-retest-1 commit 46a00dc web application hosted at https://[REDACTED].kobalt.es/.

Tests were conducted using the following roles:

- [REDACTED_EMAIL1] - Tenant A (Northwind Retail)
- [REDACTED_EMAIL2]- Tenant A (Northwind Retail)
- [REDACTED_EMAIL3]- Tenant B (Calverton Pharma)
- [REDACTED_EMAIL4]- Tenant B (Calverton Pharma)

The most severe vulnerability identified during the assessment was:

- SECURITUM-2427553-001: No session invalidation after logout (JWT token reuse).

During the tests, particular emphasis was placed on vulnerabilities that might in a negative way affect confidentiality, integrity or availability of processed data.

The security tests were carried out according to generally accepted methodologies, including: OWASP TOP10, (in a selected range) OWASP ASVS 5.0 level 2 as well as internal good practices of conducting security tests developed by Securitum.

An approach based on manual tests (using the above-mentioned methodologies), supported by several automatic tools (i.a. Burp Suite Professional, feroxbuster, testssl, nikto), was used during the assessment.

The vulnerabilities are described in detail in further parts of the report.

Status after retests (11.08.2026)

On 11 August 2026, a retest of the previously identified vulnerabilities was performed.

The table below summarizes the current remediation status of each finding:

ID	Vulnerability name	Status
SECURITUM-2427553-001	No session invalidation after logout (JWT token reuse)	FIXED
SECURITUM-2427553-002	User enumeration via response time difference	FIXED
SECURITUM-2427553-003	Redundant information disclosure about the application environment in HTTP response headers	FIXED
SECURITUM-2427553-004	Support for outdated TLS ciphers	FIXED
SECURITUM-2427553-005	X-XSS-Protection header enabled	IMPLEMENTED
SECURITUM-2427553-006	Numeric resource identifiers	NOT IMPLEMENTED
SECURITUM-2427553-007	General lack of input validation	NOT IMPLEMENTED
SECURITUM-2427553-008	Lack of rate limiting mechanisms	IMPLEMENTED
SECURITUM-2427553-009	Inconsistency between invoice list and invoice detail endpoints	IMPLEMENTED
SECURITUM-2427553-010	Lack of Cache Control headers	IMPLEMENTED
SECURITUM-2427553-011	Lack of filename length validation in file upload	IMPLEMENTED

Detailed information is provided in further parts of the document.

Risk classification

Vulnerabilities are classified on a five-point scale, that reflects both the probability of exploitation of the vulnerability and the business risk of its exploitation. Below, there is a short description of the meaning of each of the severity levels:

- CRITICAL – exploitation of the vulnerability makes it possible to compromise the server or network device, or makes it possible to access (in read and/or write mode) data with a high degree of confidentiality and significance. The exploitation is usually straightforward, i.e. an attacker does not need to gain access to the systems that are difficult to reach and does not need to perform social engineering. Vulnerabilities marked as 'CRITICAL' must be fixed without delay, mainly if they occur in the production environment.

- HIGH – exploitation of the vulnerability makes it possible to access sensitive data (similar to the 'CRITICAL' level), however the prerequisites for the attack (e.g. possession of a user account in an internal system) make it slightly less likely. Alternatively, the vulnerability is easy to exploit, but the effects are somehow limited.

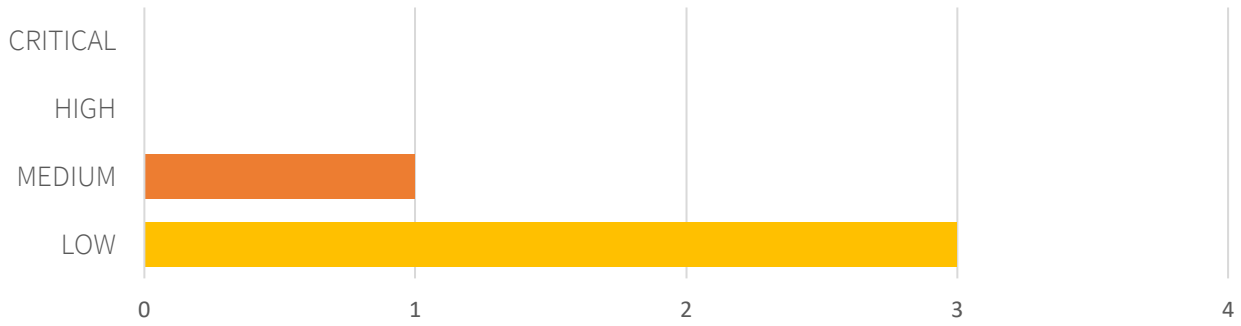
- MEDIUM – exploitation of the vulnerability might depend on external factors (e.g. convincing the user to click on a hyperlink) or other conditions that are difficult to achieve. Furthermore, exploitation of the vulnerability usually allows access only to a limited set of data or to data of a lesser degree of significance.

- LOW – exploitation of the vulnerability results in minor direct impact on the security of the test subject or depends on conditions that are very difficult to achieve in practical manner (e.g. physical access to the server).

- INFO – issues marked as 'INFO' are not security vulnerabilities per se. They aim to point out good practices, the implementation of which will lead to the overall increase of the system security level. Alternatively, the issues point out some solutions in the system (e.g. from an architectural perspective) that might limit the negative effects of other vulnerabilities.

Statistical overview

Below, a statistical summary of vulnerabilities is shown:



Additionally, seven INFO issues were reported.

Statistical overview after retest (11.08.2026)

Two INFO issues remain reported.

Contents

Security report *1*

Executive summary *2*

Status after retests (11.08.2026) 3

Risk classification 4

Statistical overview 4

Statistical overview after retest (11.08.2026) 5

Change history *7*

Vulnerabilities in the web application *8*

[FIXED][MEDIUM] SECURITUM-2427553-001: No session invalidation after logout (JWT token reuse) 9

[FIXED][LOW] SECURITUM-2427553-002: User enumeration via response time difference 12

[FIXED][LOW] SECURITUM-2427553-003: Redundant information disclosure about the application environment in HTTP response headers 15

[FIXED][LOW] SECURITUM-2427553-004: Support for outdated TLS ciphers 17

Informational issues *18*

[IMPLEMENTED][INFO] SECURITUM-2427553-005: X-XSS-Protection header enabled 19

[NOT IMPLEMENTED][INFO] SECURITUM-2427553-006: Numeric resource identifiers 20

[NOT IMPLEMENTED][INFO] SECURITUM-2427553-007: General lack of input validation 21

[IMPLEMENTED][INFO] SECURITUM-2427553-008: Lack of rate limiting mechanisms 23

[IMPLEMENTED][INFO] SECURITUM-2427553-009: Inconsistency between invoice list and invoice detail endpoints 25

[IMPLEMENTED][INFO] SECURITUM-2427553-010: Lack of Cache Control headers 28

[IMPLEMENTED][INFO] SECURITUM-2427553-011: Lack of filename length validation in file upload 31

Change history

Document date	Version	Change description
11.08.2026	1.1	Retest of the previously identified vulnerabilities. The report was updated to include the post-retest status in the summary of work performed and for each vulnerability. In addition, a statistical summary of the retest results was added.
22.07.2026	1.0	Creation of the security report.

Vulnerabilities in the web application

[FIXED][MEDIUM] SECURITUM-2427553-001: No session invalidation after logout (JWT token reuse)

STATUS AFTER RETESTS (11.08.2026)

The vulnerability has been remediated. The application now correctly invalidates JWT tokens upon logout — a subsequent request to GET `/api/auth/me` using the previously issued token returns:

```
HTTP/2 401 Unauthorized
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: https://[REDACTED].kobalt.es
Alt-Svc: h3=":443"; ma=2592000
Cache-Control: no-store, no-cache, must-revalidate, max-age=0
Content-Security-Policy: frame-ancestors 'none';form-action 'self';default-src 'self';base-uri
'self';font-src 'self' https: data:;img-src 'self' data:;object-src 'none';script-src
'self';script-src-attr 'none';style-src 'self' https: 'unsafe-inline';upgrade-insecure-requests
Content-Type: application/json; charset=utf-8
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: same-origin
Date: Mon, 10 Aug 2026 08:56:48 GMT
[…]

{"error":"Session expired"}
```

Confirming that the token is no longer accepted by the server after the logout operation. Server-side token invalidation is now properly enforced.

SUMMARY

The application implements a user session mechanism using stateless JWT tokens. Based on the conducted tests, it was determined that the logout operation performed by the POST `/api/auth/logout` endpoint does not invalidate the token on the server side. The token issued before logout remains accepted by the API for its entire lifetime (TTL = 4 hours).

More information:

- https://github.com/OWASP/ASVS/raw/v5.0.0/5.0/OWASP_Application_Security_Verification_Standard_5.0.0_en.pdf - (7.4.1)

PREREQUISITES FOR THE ATTACK

Possession of a valid user session token, obtainable for example by intercepting network traffic, accessing the victim's device or browser storage, capturing the token from proxy or server logs, or through a cross-site scripting attack.

TECHNICAL DETAILS (PROOF OF CONCEPT)

In the first step, the validity of the session token was verified, and the authenticated user's information was retrieved:

```
GET /api/auth/me HTTP/2
Host: [REDACTED].kobalt.es
Cookie: session=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.[REDACTED]
```

```
[…]
```

Server response:

```
HTTP/2 200 OK
[…]
Date: Fri, 17 Jul 2026 09:19:14 GMT
[…]

{
  "user": {
    "id": 4139,
    "name": "John Snow",
    "email": "[REDACTED_EMAIL1]",
    "customerId": 604,
    "customerName": "Northwind Retail"
  }
}
```

Next, the logout operation was performed:

```
POST /api/auth/logout HTTP/2
Host: [REDACTED].kobalt.es
Cookie: session=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.[REDACTED]
[…]
```

Server response:

```
HTTP/2 200 OK
[…]
Set-Cookie: session=; Path=/; Expires=Thu, 01 Jan 1970 00:00:00 GMT
[…]

{"ok":true}
```

After logging out, another request was sent using the same session token:

```
GET /api/auth/me HTTP/2
Host: [REDACTED].kobalt.es
Cookie: session=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.[REDACTED]
[…]
```

Server response confirming that the session token is still valid:

```
HTTP/2 200 OK
[…]
Date: Fri, 17 Jul 2026 09:25:02 GMT
[…]

{
  "user": {
    "id": 4139,
    "name": "John Snow",
    "email": "[REDACTED_EMAIL1]",
    "customerId": 604,
    "customerName": "Northwind Retail"
  }
}
```

LOCATION

- https://[REDACTED].kobalt.es — POST /api/auth/logout
- https://[REDACTED].kobalt.es — GET /api/auth/me

RECOMMENDATION

It is recommended to implement a server-side token blocklist mechanism based on the token's unique identifier (`jti` claim). Upon each logout operation, the `jti` value should be stored in a key-value store (e.g., Redis) with a TTL matching the token's remaining validity period. Every token validation should include a check to determine whether the `jti` is present in the blocklist before allowing the request to proceed.

More information:

- https://cheatsheetseries.owasp.org/cheatsheets/JSON_Web_Token_Cheat_Sheet.html
- https://www.descope.com/blog/post/jwt-logout-risks-mitigations
- https://supertokens.com/blog/revoking-access-with-a-jwt-blacklist

[FIXED][LOW] SECURITUM-2427553-002: User enumeration via response time difference

STATUS AFTER RETESTS (11.08.2026)

The vulnerability has been remediated. The POST `/api/auth/magic-link` endpoint now returns consistent response times regardless of whether the submitted email address exists in the system, approximately 51 ms for a registered address and 53 ms for a non-existent one.

SUMMARY

The application provides a magic-link authentication mechanism in which the user submits their email address via the POST `/api/auth/magic-link` endpoint and receives a sign-in link to their inbox. The endpoint always returns an identical response body (`{"ok":true}`) regardless of whether the submitted address exists in the system.

During testing, a difference in response time was observed depending on whether the submitted email address corresponds to an existing account - approximately 1,688 ms for valid addresses versus approximately 45 ms for non-existent ones. Despite the uniform response body, this discrepancy constitutes a measurable side-channel that allows an attacker to determine whether a given email address is registered in the application.

More information:

- https://cwe.mitre.org/data/definitions/208.html
- https://github.com/OWASP/ASVS/raw/v5.0.0/5.0/OWASP_Application_Security_Verification_Standard_5.0.0_en.pdf - (6.3.8)

PREREQUISITES FOR THE ATTACK

No authentication required. The endpoint is publicly accessible.

TECHNICAL DETAILS (PROOF OF CONCEPT)

Two requests were sent to the magic-link endpoint. One with a valid registered email address and one with a non-existent address. Response times were measured using Burp Suite Repeater.

Request with a valid email address:

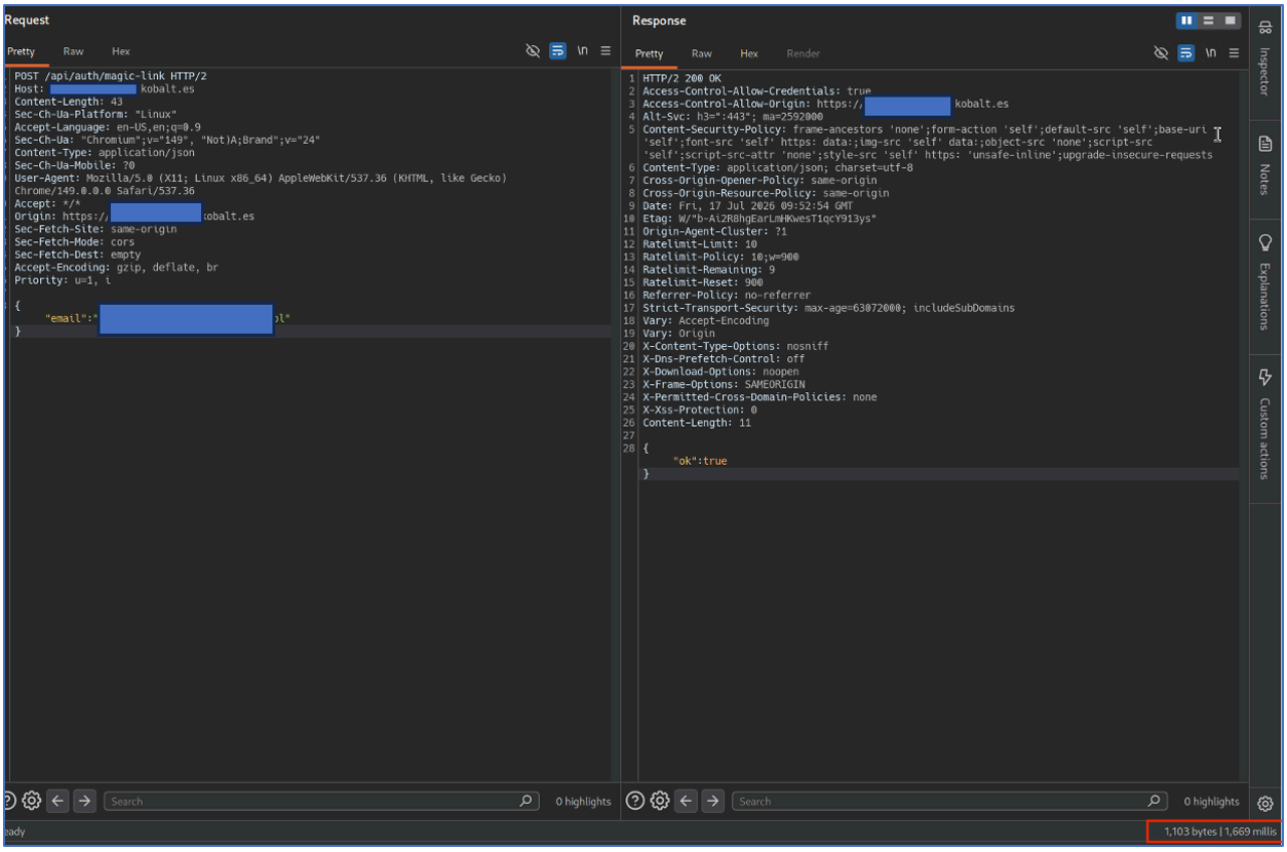
```
POST /api/auth/magic-link HTTP/2
Host: [REDACTED].kobalt.es
[…]

{"email":"[REDACTED_EMAIL1]"}
```

Server response time 1,669 ms:

```
HTTP/2 200 OK
[…]

{"ok":true}
```



Request with a non-existent email address:

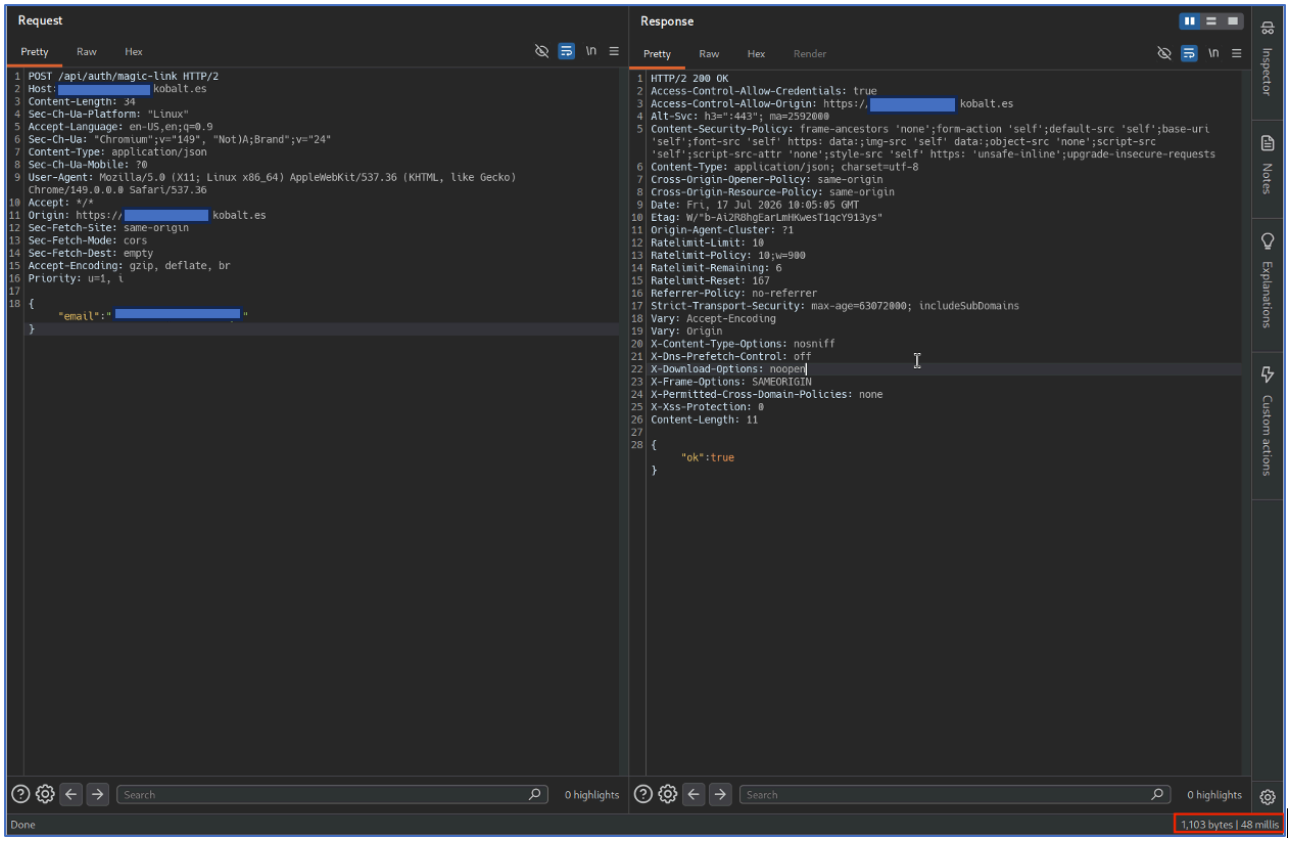
```
POST /api/auth/magic-link HTTP/2
Host: [REDACTED].kobalt.es
[…]

{"email":"[…]"}
```

Server response time 48 ms:

```
HTTP/2 200 OK
[…]

{"ok":true}
```



The response body is identical in both cases, however the difference in response time allows reliable distinction between existing and non-existent accounts. An attacker can leverage this difference to enumerate registered email addresses without triggering any visible error response.

LOCATION

- https://[REDACTED].kobalt.es — POST /api/auth/magic-link

RECOMMENDATION

It is recommended to ensure that the response time for the POST `/api/auth/magic-link` endpoint is consistent regardless of whether the submitted email address exists in the system. This can be achieved by always performing the backend lookup even for non-existent addresses or by introducing a fixed minimum response delay (e.g. using setTimeout) that normalizes response times across all cases.

More information:

- https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html
- https://passkwall.medium.com/timing-based-username-enumeration-whats-a-fix-versus-mitigation-8a0f8c7191f2

[FIXED][LOW] SECURITUM-2427553-003: Redundant information disclosure about the application environment in HTTP response headers

STATUS AFTER RETESTS (11.08.2026)

The vulnerability has been remediated. The `Server` response header no longer discloses the nginx version.

Request sent to the application:

```
POST /api/auth/magic-link HTTP/2
Host: test
[…]

{"email":"nonexists@test.pl"}
```

Response:

```
HTTP/2 405 Method Not Allowed
Alt-Svc: h3=":443"; ma=2592000
Content-Type: text/html
Date: Mon, 10 Aug 2026 09:30:36 GMT
Content-Length: 150

<html>
<head><title>405 Not Allowed</title></head>
<body>
<center><h1>405 Not Allowed</h1></center>
<hr><center>nginx</center>
</body>
</html>
```

SUMMARY

During the audit, it was observed that the tested application returns redundant information in the HTTP response headers about the technologies in use. This behaviour can help an attacker to better profile the application environment, which then can be used to carry out further attacks.

More information:

- https://wiki.owasp.org/index.php/Testing_for_Web_Application_Fingerprint_(OWASP-IG-004)
- https://github.com/OWASP/OWASP-Testing-Guide/blob/master/4-Web-Application-Security-Testing/4.2.2%20Fingerprint%20Web%20Server%20(OTG-INFO-002)
- https://www.cybersecurity-help.cz/vdb/soft/f5_networks/nginx/1.29.5/
- https://cwe.mitre.org/data/definitions/200.html
- https://github.com/OWASP/ASVS/raw/v5.0.0/5.0/OWASP_Application_Security_Verification_Standard_5.0.0_en.pdf - (13.4.6)

PREREQUISITES FOR THE ATTACK

No authentication required. The server version is disclosed in HTTP responses to unauthenticated requests.

TECHNICAL DETAILS (PROOF OF CONCEPT)

Example of the HTTP request sent to the application:

```
POST /api/auth/magic-link HTTP/2
Host: test
[…]

{"email":"[…]"}
```

The request was intentionally sent with an invalid `Host` header value in order to trigger a server-level error response, which reveals the underlying web server version in the `Server` header.

In response, the application returns:

```
HTTP/2 405 Method Not Allowed
Alt-Svc: h3=":443"; ma=2592000
Content-Type: text/html
Date: Fri, 17 Jul 2026 10:53:11 GMT
Server: nginx/1.29.5
Content-Length: 559

<html>
<head><title>405 Not Allowed</title></head>
<body>
<center><h1>405 Not Allowed</h1></center>
<hr><center>nginx/1.29.5</center>
</body>
</html>
<!-- a padding to disable MSIE and Chrome friendly error page -->
<!-- a padding to disable MSIE and Chrome friendly error page -->
<!-- a padding to disable MSIE and Chrome friendly error page -->
<!-- a padding to disable MSIE and Chrome friendly error page -->
<!-- a padding to disable MSIE and Chrome friendly error page -->
<!-- a padding to disable MSIE and Chrome friendly error page -->
```

LOCATION

- Generic — applies to all HTTP responses returned by the tested application.

RECOMMENDATION

It is recommended to remove all unnecessary headers from the HTTP responses that reveal information about the technologies used.

More information:

- https://nginx.org/en/docs/http/ngx_http_core_module.html#server_tokens

[FIXED][LOW] SECURITUM-2427553-004: Support for outdated TLS ciphers

STATUS AFTER RETESTS (11.08.2026)

The vulnerability has been remediated. Both weak CBC cipher suites have been disabled.

SUMMARY

The tested application supports weak TLS ciphers, which are used to set up a secure communication channel. This could pose a risk of compromising or modifying sensitive user data if an attacker eavesdrops network traffic (Man in the Middle attack, MitM).

More information:

- https://cwe.mitre.org/data/definitions/757.html
- https://cwe.mitre.org/data/definitions/326.html
- https://github.com/OWASP/ASVS/raw/v5.0.0/5.0/OWASP_Application_Security_Verification_Standard_5.0.0_en.pdf - (12.1.2)

PREREQUISITES FOR THE ATTACK

Performing a Man in the Middle attack.

TECHNICAL DETAILS (PROOF OF CONCEPT)

The scan was performed using testssl.sh (https://github.com/testssl/testssl.sh).

The server on which the tested application is running supports the following TLS ciphers:

```
TLS 1.2:
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
```

LOCATION

- https://[REDACTED].kobalt.es/

RECOMMENDATION

It is recommended to disable support for the TLS ciphers mentioned above.

The current recommended algorithm configuration can be found at:

- https://ssl-config.mozilla.org/

More information:

- https://cheatsheetseries.owasp.org/cheatsheets/Transport_Layer_Security_Cheat_Sheet.html

Informational issues

[IMPLEMENTED][INFO] SECURITUM-2427553-005: X-XSS-Protection header enabled

STATUS AFTER RETESTS (11.08.2026)

The `X-XSS-Protection` header is set to `0`. The header can be safely removed if this functionality is not required.

SUMMARY

It was observed that HTTP responses contain `X-XSS-Protection` header. This header is not supported anymore by majority of the browsers (such as Chrome, Mozilla Firefox, Microsoft Edge), and in very rare and specific cases may open an application to the XS-Leak vulnerability. The role of `X-XSS-Protection` header was taken over by a Content Security Policy.

More information:

- https://markitzeroday.com/headers/content-security-policy/2018/02/10/x-xss-protection.html
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protection
- https://portswigger.net/daily-swig/google-deprecates-xss-auditor-for-chrome

More information on XS-Leak attack:

- https://cheatsheetseries.owasp.org/cheatsheets/XS_Leaks_Cheat_Sheet.html

LOCATION

- https://[REDACTED].kobalt.es/

RECOMMENDATION

It is recommended to verify if the `X-XSS-Protection` header is necessary (for example, if an application is used in very old browsers, which do not support Content Security Policy). If not, it should be deleted.

It should be noted that its occurrence does not automatically open an application to new vulnerabilities (as the exploitation of XS-Leaks may be very sophisticated and not possible in each case), and this recommendation is only a suggestion, allowing for additional hardening.

[NOT IMPLEMENTED][INFO] SECURITUM-2427553-006: Numeric resource identifiers

STATUS AFTER RETESTS (11.08.2026)

The client confirmed that numeric identifiers are generated by the external XTRF system and are outside of their control. The risk is mitigated by server-side ownership verification enforced on every resource access.

The recommendation has not been implemented as no structural changes were made to replace the predictable numeric identifiers.

The client confirmed that these identifiers originate from an external system integration (XTRF) and cannot be modified on the application level. To address the risk, the client relies on server-side access control (ownership verification) enforced on every resource request as a compensating control.

While effective authorization mitigates the risk of unauthorized access (IDOR), the application still utilizes sequential/predictable IDs, which allows for resource enumeration and insight into total object volumes. The risk has been accepted by the client.

SUMMARY

It was observed that the application uses numeric resource identifiers (e.g., quote number with identifier 4330). While this approach does not currently constitute a security vulnerability by itself (the application additionally verifies the session), in the event of an unauthorized access vulnerability, the predictability of the identifiers would significantly facilitate an attack. It is recommended to use unpredictable identifiers, such as UUIDv4.

PREREQUISITES FOR THE ATTACK

Application account.

TECHNICAL DETAILS (PROOF OF CONCEPT)

Examples of identifiers used in the application:

- https://[REDACTED].kobalt.es/quotes/4330,
- https://[REDACTED].kobalt.es/invoices/4794,
- https://[REDACTED].kobalt.es/projects/41661.

LOCATION

Listed in the "Technical details" section.

RECOMMENDATION

It is recommended to use unpredictable resource identifiers, such as UUIDv4.

[NOT IMPLEMENTED][INFO] SECURITUM-2427553-007: General lack of input validation

STATUS AFTER RETESTS (11.08.2026)

The recommendation has not been implemented. The client confirmed that server-side schema validation is in place for types, lengths, and unknown fields, and that output is escaped where it is rendered. Character filtering was intentionally omitted as project names and reference numbers legitimately contain punctuation. As noted in the original finding, this does not currently pose a direct security risk.

SUMMARY

During the tests, it was observed that most fields do not have proper server-side validation implemented; currently, only partial client-side validation is performed.

The vulnerability does not currently pose a security risk, but in the case of communication between the application and other systems, it may cause unexpected behavior in those systems.

More information:

- https://github.com/OWASP/ASVS/raw/v5.0.0/5.0/OWASP_Application_Security_Verification_Standard_5.0.0_en.pdf - (2.2.1, 2.2.2)

TECHNICAL DETAILS (PROOF OF CONCEPT)

Below is an example request with the fields highlighted in yellow that do not have any server-side validation:

```
POST /api/projects/create HTTP/2
Host: [REDACTED].kobalt.es
[…]

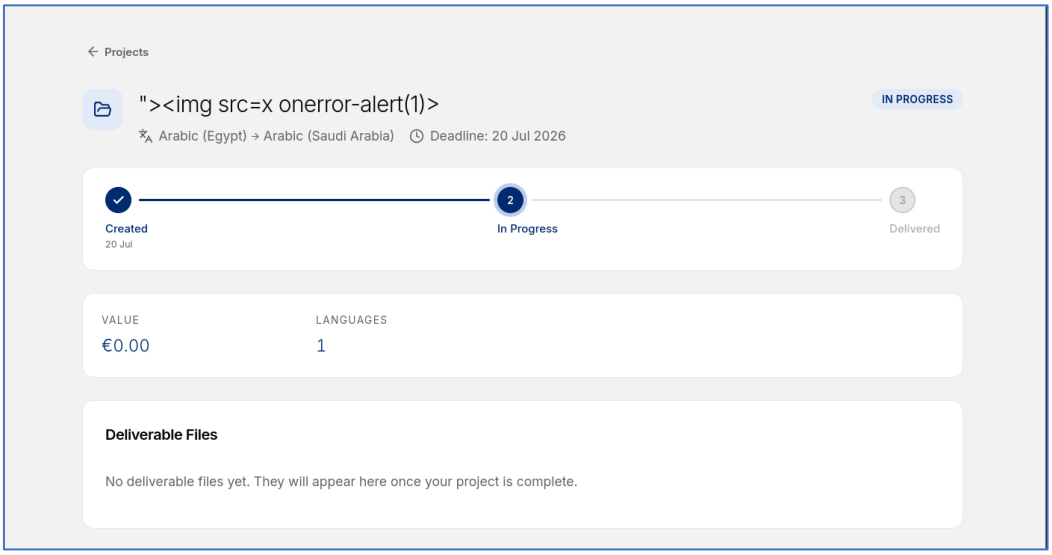
{"name":"\"><img src=x onerror-
alert(1)>","serviceId":65,"sourceLanguageId":9,"targetLanguageIds":[18],"specializationId":28,"de
liveryDate":1784502000000,"referenceNumber":"\"><img src=x onerror-alert(1)>","notes":"\"><img
src=x onerror-alert(1)>"}
```

In response, the application accepts the above request:

```
HTTP/2 200 OK
[…]

{"id":"5FUM4REJNR[…]","projectId":"5FUM4REJNRDQ[…]","isClassicProject":false,"quoteIdNumber":null
,"name":"\"><img src=x onerror-
alert(1)>","status":"OPEN","categoryIds":[],"budgetCode":null,"clientId":605,"serviceId":65,"orig
in":"API
v2","clientDeadline":null,"clientReferenceNumber":null,"clientNotes":null,"internalNotes":null,"v
olume":{"value":null,"unitId":1},"languages":{"sourceLanguageId":null,"targetLanguageIds":[],"spe
cializationId":13,"languageCombinations":[]},[…]
```

Browser preview:



LOCATION

- https://[REDACTED].kobalt.es - POST /api/projects/create
- https://[REDACTED].kobalt.es - POST /api/quotes

RECOMMENDATION

It is recommended to implement filtering that rejects values that do not match the expected pattern for a given field. Validation should be implemented on the server side and not only on the client side.

It should be noted that the server should not return the values that caused the error but only indicate that: "An invalid value was entered in the XYZ field", optionally with additional information about the allowed characters, e.g.: "Allowed characters are letters and numbers".

More information:

- https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html

[IMPLEMENTED][INFO] SECURITUM-2427553-008: Lack of rate limiting mechanisms

STATUS AFTER RETESTS (11.08.2026)

The recommendation has been implemented. Retest confirmed the presence of `Ratelimit-Limit: 600`, `Ratelimit-Policy: 600;w=900`, `Ratelimit-Remaining`, and `Ratelimit-Reset` headers in responses for all listed endpoints.

SUMMARY

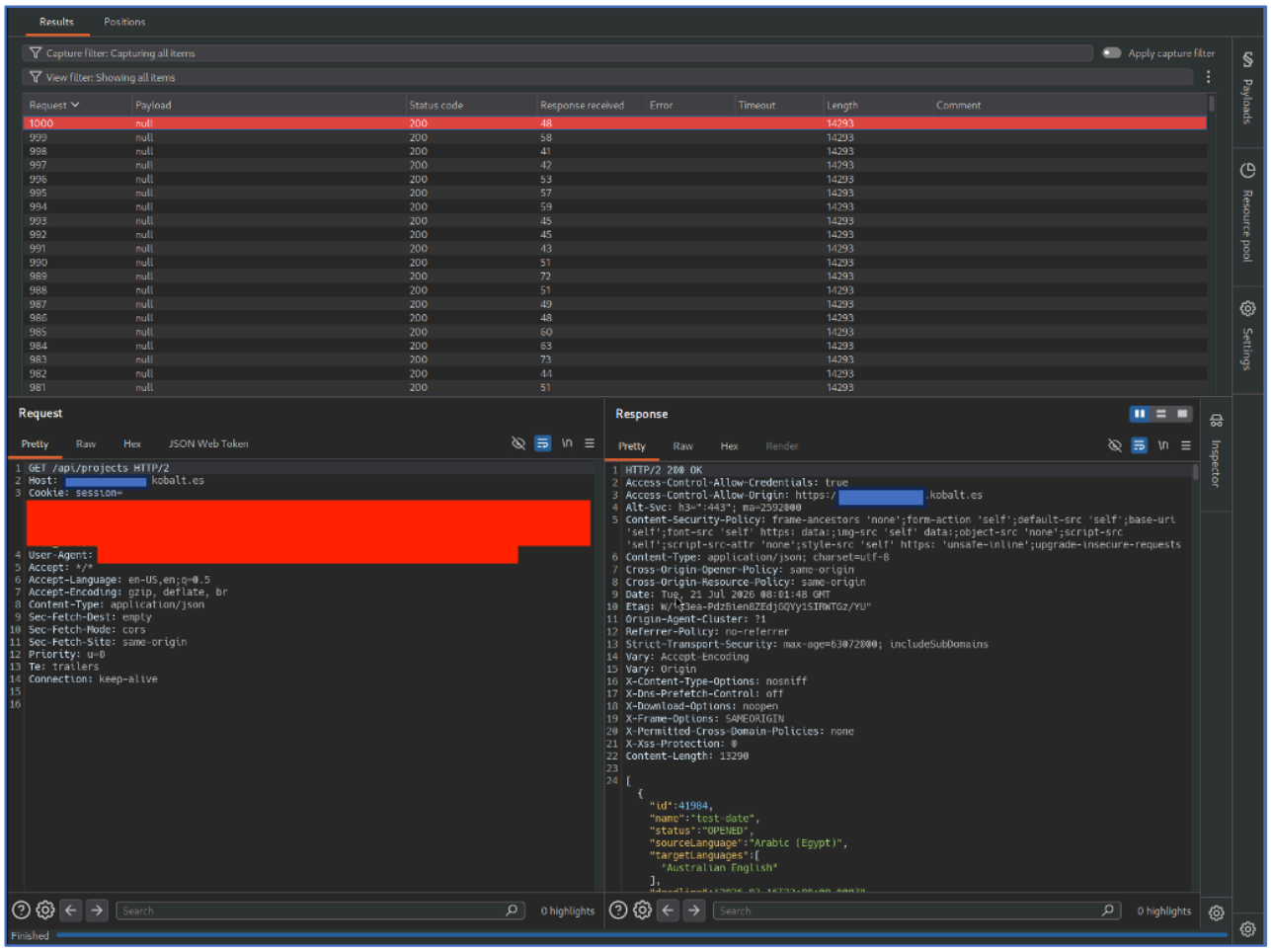
During the tests, it was determined that authenticated API endpoints returning sensitive business data do not implement rate limiting. An attacker with a valid session token can send an unlimited number of requests without triggering any throttling or blocking mechanism, potentially enabling automated data harvesting or Denial of Service attacks.

More information:

- https://owasp.org/API-Security/editions/2023/en/0xa4-unrestricted-resource-consumption/
- https://cwe.mitre.org/data/definitions/770.html
- https://owasp.org/www-community/attacks/Denial_of_Service
- https://github.com/OWASP/ASVS/raw/v5.0.0/5.0/OWASP_Application_Security_Verification_Standard_5.0.0_en.pdf - (2.4.1)

TECHNICAL DETAILS (PROOF OF CONCEPT)

The verification was performed using the Burp Suite Intruder tool by sending 1,000 requests to the GET `/api/projects` endpoint. All requests were processed with an HTTP `200` response code, without any throttling, blocking, or an HTTP `429` (Too Many Requests) response.



LOCATION

- https://[REDACTED].kobalt.es - GET /api/projects
- https://[REDACTED].kobalt.es — GET /api/quotes
- https://[REDACTED].kobalt.es — GET /api/invoices
- https://[REDACTED].kobalt.es — GET /api/analytics

RECOMMENDATION

It is recommended to implement API communication rate limiting (e.g., HTTP requests) within defined time intervals.

More information:

- https://cheatsheetseries.owasp.org/cheatsheets/Denial_of_Service_Cheat_Sheet.html
- https://www.npmjs.com/package/express-rate-limit

[IMPLEMENTED][INFO] SECURITUM-2427553-009: Inconsistency between invoice list and invoice detail endpoints

STATUS AFTER RETESTS (11.08.2026)

The recommendation has been implemented. The GET `/api/invoices/{id}`, GET `/api/invoices/{id}/pdf` and GET `/api/invoices` endpoints now return HTTP 200 with the correct invoice data for the authenticated owner. The client confirmed that the root cause was a field-name mismatch in the ownership check, which has been corrected for both endpoints.

SUMMARY

An inconsistency was identified between the behavior of the invoice list endpoint and the invoice detail endpoint. The GET `/api/invoices` endpoint correctly returns invoices belonging to the authenticated user's tenant. However, attempting to access the same invoice directly via GET `/api/invoices/{id}` results in a `403 Forbidden` response, even when the requesting user is the legitimate owner of the resource.

This behavior was confirmed across two separate user accounts belonging to the same tenant (Calverton Pharma). This is not a security vulnerability - tenant isolation is correctly enforced and no unauthorized data access was observed. The issue is reported as an informational finding to highlight a functional inconsistency in the API implementation.

TECHNICAL DETAILS (PROOF OF CONCEPT)

Request to the invoice list endpoint (returns invoice data correctly):

```
GET /api/invoices HTTP/2
Host: [REDACTED].kobalt.es
Cookie: session=<TOKEN_12.1>
[…]
```

Server response:

```
HTTP/2 200 OK
[…]
Date: Mon, 20 Jul 2026 09:43:21 GMT
[…]

[
  {
    "id": […],
    "number": "2026/[…]",
    "status": "SENT",
    "totalGross": 1,
    "totalNet": 1,
    "currency": "EUR",
    "issueDate": "2026-07-14T22:00:00.000Z",
    "dueDate": "2026-08-13T22:00:00.000Z",
    "paidDate": null
  }
]
```

Request to the invoice detail endpoint using the same invoice ID:

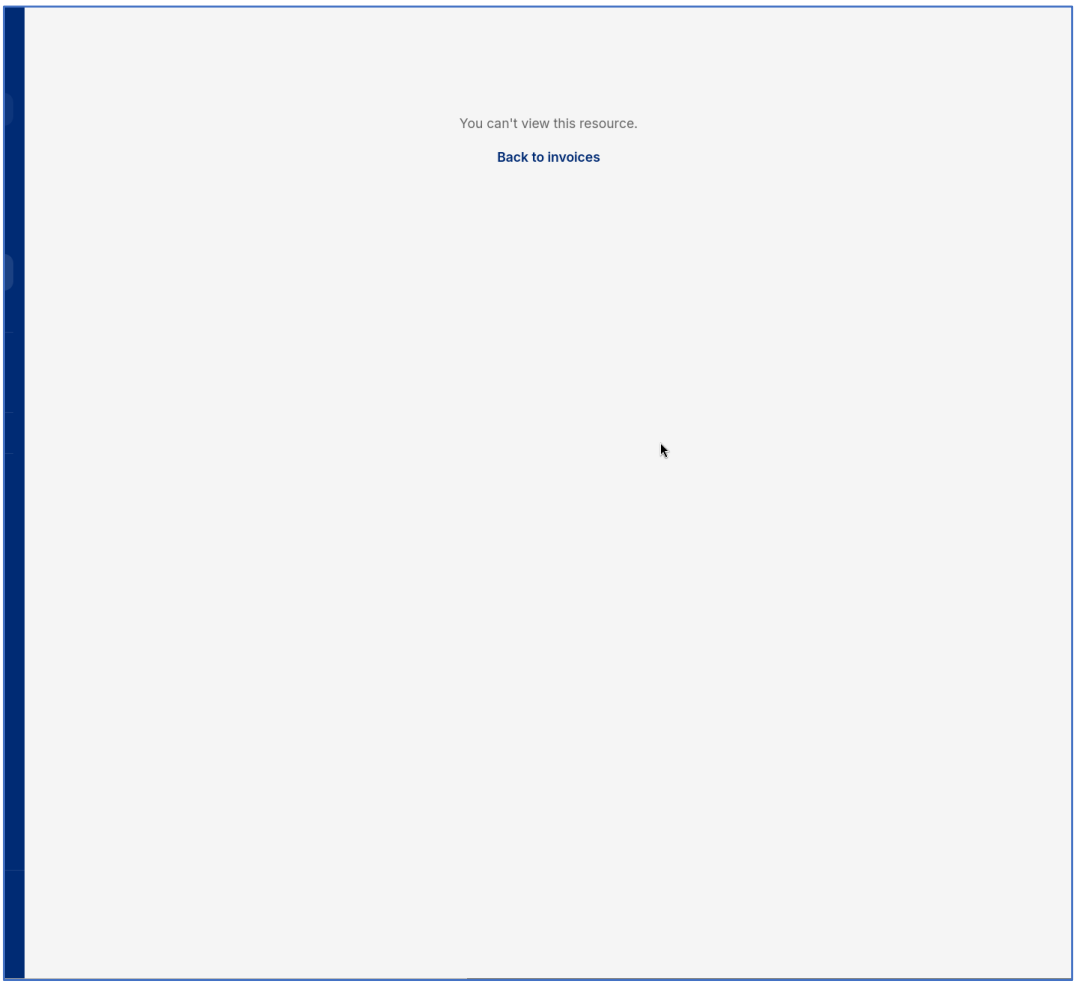
```
GET /api/invoices/4[…] HTTP/2
Host: [REDACTED].kobalt.es
Cookie: session=<TOKEN_12.1>
[…]
```

Server response:

```
HTTP/2 403 Forbidden
[…]
Date: Mon, 20 Jul 2026 09:45:16 GMT
[…]

{"error":"Forbidden"}
```

The same result was observed for a second user account within the same tenant (12.2 — Laura Dern), confirming that the issue is not user-specific.



The screenshot above confirms that the invoice detail view is inaccessible in the UI, even when accessed by the invoice owner.

LOCATION

- https://[REDACTED].kobalt.es — GET /api/invoices/{id}

RECOMMENDATION

It is recommended to verify the authorization logic of the GET `/api/invoices/{id}` endpoint and align it with the logic applied in GET `/api/invoices`. If it is intended to be functional, the authorization check should be corrected to allow legitimate owners to access their own resources by ID.

[IMPLEMENTED][INFO] SECURITUM-2427553-010: Lack of Cache Control headers

STATUS AFTER RETESTS (11.08.2026)

The recommendation has been implemented.

SUMMARY

During the testing, it was observed that authenticated API endpoints returning sensitive user data do not include appropriate Cache-Control headers in their responses. The absence of these headers means the browser applies default caching behavior, which may result in sensitive data being stored in the local cache. If another user gains access to the same device or browser, they may be able to retrieve previously cached responses containing account information.

It should be noted that the session token is stored in an `httpOnly` cookie and is not subject to browser caching. The risk relates primarily to the caching of API response bodies containing business data such as project lists, quotes, invoices and analytics.

Appropriate Cache Control headers responsible for preventing the saving of sensitive application data in the temporary memory of the browser should include:

- Cache-Control,
- Pragma,
- Expires.

`Cache-Control` is a header containing specific directives for the caching engine that are supposed to instruct the browser: which elements, and whether they should be stored in the temporary storage. Although some directives may also be used in a HTTP request, it is recommended to focus mainly on HTTP responses.

`Pragma` in case of using the “no-cache” directive means the same as “Cache-Control: no-cache” – that is, forces the browser to send a query to the server and download the current version before downloading the version that is saved in the browsers cache.

`Expires` contains the date, time or value that determines when the server response is no longer valid. Frequently used values in the form of a past date or “0” mean that the resource is no longer valid.

For more information on implementing caching headers, please visit:

- https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/04-Authentication_Testing/06-Testing_for_Browser_Cache_Weaknesses
- https://www.rfc-editor.org/rfc/rfc9111#name-field-definitions
- https://cwe.mitre.org/data/definitions/524.html
- https://cwe.mitre.org/data/definitions/525.html

More information:

- https://github.com/OWASP/ASVS/raw/v5.0.0/5.0/OWASP_Application_Security_Verification_Standard_5.0.0_en.pdf - (14.3.2)

TECHNICAL DETAILS (PROOF OF CONCEPT)

Sent request:

```
GET /api/auth/me HTTP/2
Host: [REDACTED].kobalt.es
Cookie: session=[REDACTED]
[…]
```

Response:

```
HTTP/2 200 OK
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: https://[REDACTED].kobalt.es
Alt-Svc: h3=":443"; ma=2592000
Content-Security-Policy: frame-ancestors 'none';form-action 'self';default-src 'self';base-uri
'self';font-src 'self' https: data:;img-src 'self' data:;object-src 'none';script-src
'self';script-src-attr 'none';style-src 'self' https: 'unsafe-inline';upgrade-insecure-requests
Content-Type: application/json; charset=utf-8
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: same-origin
Date: Tue, 21 Jul 2026 08:47:02 GMT
Etag: W/"84-ajhGZZF8uvtPgV7MMXyv93rPNeg"
Origin-Agent-Cluster: ?1
Referrer-Policy: no-referrer
Strict-Transport-Security: max-age=63072000; includeSubDomains
Vary: Accept-Encoding
Vary: Origin
X-Content-Type-Options: nosniff
X-Dns-Prefetch-Control: off
X-Download-Options: noopen
X-Frame-Options: SAMEORIGIN
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 0
Content-Length: 132

{"user":{"id":4139,"name":"John
Snow","email":"[REDACTED_EMAIL1]","customerId":604,"customerName":"Northwind Retail"}}
```

The response does not contain a Cache-Control header with a no-store directive. The same behavior was observed across all authenticated endpoints returning sensitive data, including `/api/projects`, `/api/quotes`, `/api/invoices` and `/api/analytics`.

LOCATION

- https://[REDACTED].kobalt.es - GET /api/auth/me
- https://[REDACTED].kobalt.es - GET /api/projects
- https://[REDACTED].kobalt.es - GET /api/quotes
- https://[REDACTED].kobalt.es - GET /api/invoices
- https://[REDACTED].kobalt.es - GET /api/analytics

RECOMMENDATION

It is recommended to implement Cache Control headers in the server response, according to ASVS standard:

```
Cache-Control: no-store, no-cache, must-revalidate, max-age=0
```

```
Pragma: no-cache
Expires: Tue, 07 Jul 2001 07:00:00 GMT
```

More information:

- https://web.dev/articles/http-cache
- https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/04-Authentication_Testing/06-Testing_for_Browser_Cache_Weaknesses

[IMPLEMENTED][INFO] SECURITUM-2427553-011: Lack of filename length validation in file upload

STATUS AFTER RETESTS (11.08.2026)

The recommendation has been implemented. Files with filenames up to 255 characters are accepted, while filenames exceeding this limit are rejected with `HTTP 400 Bad Request`. Additionally, oversized requests that exceed the parser limit now return `{"error":"Malformed upload request."}` instead of the previous `HTTP 500 Internal Server Error`.

SUMMARY

The file upload functionality does not implement server-side validation of the uploaded filename length. Submitting a file with an excessively long filename causes the application to return an `HTTP 500 Internal Server Error` instead of a proper `HTTP 400 Bad Request` response with an appropriate validation message.

The filename parameter is passed to the backend without length validation, resulting in a server-side exception. Although no sensitive information is disclosed and the error response does not expose internal application details, repeated requests containing oversized filenames may cause unnecessary exception handling and increased logging activity.

More information:

- https://cheatsheetseries.owasp.org/cheatsheets/File_Upload_Cheat_Sheet.html
- https://cwe.mitre.org/data/definitions/20.html
- https://github.com/OWASP/ASVS/raw/v5.0.0/5.0/OWASP_Application_Security_Verification_Standard_5.0.0_en.pdf - (2.2.1)

TECHNICAL DETAILS (PROOF OF CONCEPT)

```
python3 -c "print('A' * 20000 + '.txt')"
```

The following request was sent to the application containing a filename exceeding 20,000 characters in length:

```
POST /api/quotes/GGYEX7MU4NAUV[…]/files HTTP/2
Host: [REDACTED].kobalt.es
Cookie: session=[REDACTED]
User-Agent: [REDACTED]
[...]

-----------------------------406391838132624635994192915461
Content-Disposition: form-data; name="files"; filename="AAAA[…]AAAAAAAAAA.txt"
Content-Type: text/plain

test

-----------------------------406391838132624635994192915461
Content-Disposition: form-data; name="category"

SOURCE_DOCUMENT
-----------------------------406391838132624635994192915461–
```

Server response:

```
HTTP/2 500 Internal Server Error
[…]
Date: Mon, 20 Jul 2026 13:51:45 GMT
[…]

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Error</title>
</head>
<body>
<pre>Internal Server Error</pre>
</body>
</html>
```

LOCATION

- https://[REDACTED].kobalt.es — POST /api/quotes/{id}/files

RECOMMENDATION

It is recommended to implement server-side validation of the filename length before passing it to the backend system. Filenames should be limited to a reasonable maximum length (e.g., 255 characters) and the application should return a descriptive `HTTP 400 Bad Request` response when the limit is exceeded.