

VULNERABILITY REPORT

APIsec Security Assessment

Application: 20663 - DST - SPIR 3359
Host URL: <https://blue.www.olb-qa10.dev.bmo.com>
Scan ID: 019b1457-62e6-7df1-9270-00cfcc563020
Status: Complete
Generated: December 17, 2025 at 05:24 PM

Table of Contents

Executive Summary	2
Scan Statistics	2
Vulnerability Summary	2
Endpoints Scanned	2
Vulnerabilities	3
Informational Findings	8

Executive Summary

This scan identified 3 vulnerabilities and 40 informational findings.



Scan Statistics

Endpoints Scanned: 21 / 21

Total Tests: 2079

Tests Passed: 881

Tests Failed: 7

Total Findings: 43

Endpoints Affected: 20

Vulnerability Summary

Severity	Detection ID	Test Name	Endpoint	CVSS
MEDIUM	12e23285	incremental	post /launch-openaccount/services/party-data-management/party-profile	5.0
MEDIUM	1d43c389	secure	get /launch-openaccount/services/product-sales-agreement/everyday-banking-journey/init	5.0
MEDIUM	94651643	secure	get /launch-openaccount/services/session-dialogue/onboarding-journey/status	5.0

Endpoints Scanned

Method	Endpoint	Vulns	Info	Auth
post	/launch-openaccount/services/party-data-management/party-profile	1	6	Yes
get	/launch-openaccount/services/product-sales-agreement/everyday-banking-journey/init	1	5	Yes
get	/launch-openaccount/services/session-dialogue/onboarding-journey/status	1	2	Yes
post	/launch-openaccount/services/party-data-management/party-profile/address	0	3	Yes
post	/launch-openaccount/services/processing-order/id-verification-and-credit-risk	0	2	Yes
post	/launch-openaccount/services/processing-order/kyc-evaluation-and-credit-risk	0	2	Yes
post	/launch-openaccount/services/product-sales-agreement/everyday-banking-journey/init	0	2	Yes
post	/launch-openaccount/services/product-sales-agreement/everyday-banking-journey/status	0	2	Yes
patch	/launch-openaccount/services/product-sales-agreement/everyday-banking-journey/status	0	2	Yes
post	/launch-openaccount/services/session-dialogue/onboarding-journey/delivery	0	2	Yes
post	/launch-openaccount/services/session-dialogue/onboarding-journey/register	0	2	Yes
get	/launch-openaccount/services/session-dialogue/onboarding-journey/status	0	2	Yes
post	/launch-openaccount/services/session-dialogue/prospect-product-selection/batch	0	2	Yes
get	/session-dialogue/onboarding-journey/end	0	2	Yes
post	/session-dialogue/onboarding-journey/status	0	2	Yes
post	/launch-openaccount/services/contact-handler/prospect-product-selection/batch	0	1	Yes
get	/launch-openaccount/services/session-dialogue/onboarding-journey/register	0	1	Yes

Vulnerabilities

3 vulnerability findings detected

/launch-openaccount/services/party-data-management/party-profile (1 finding)

incremental

MEDIUM (CVSS: 5.0)

Category: idor
Method: post /launch-openaccount/services/party-data-management/party-profile
Last Detected: December 12, 2025 at 08:53 PM

Vulnerable Parameter:

uuid (Body)

2c706b0c-2c6d-4943-a16f-2372080f366e

uuid (Response)

2c706b0c-2c6d-4943-a16f-2372080f366e

requestId (Response)

13

Auth: Authorization

OWASP:

API1:2023

Impact:

May allow users to access other users' data by manipulating object identifiers

Remediation:

Solution

1. Enforce object-level authorization on every request (never rely on client-side checks).
2. Validate the requested object belongs to the authenticated user/tenant before returning data.
3. Use unguessable identifiers (UUIDs) only as defense-in-depth (still require authorization).
4. Add tests for cross-user access and alert on repeated authorization failures.

Exploitation

- An attacker changes an object identifier (path/body/query) to access another user's records.
- Predictable/incremental IDs make enumeration and scraping significantly easier.

References

- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>
- OWASP Cheat Sheet: Authorization: https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html

Description:

This test looks for object identifiers that are incremental IDs. An incremental identifier is an identifier that increments in value in a predictable way. Currently, the test does this by checking if the identifier is numerical.

The test analyzes response payloads and searches them for fields that look like IDs. The heuristic used for this analysis looks for fields with the following patterns:

- 1) The field is `id` (case-insensitive)
- 2) The field ends with `id` (case-insensitive)

The second heuristic is prone to yield false positives in some cases, so please review carefully the detections raised by this test and flag any false positives. An incremental ID is a target for data scraping or unauthorized access attempts because they are easily guessed.

Failing Test Logs

REQUEST

post /launch-openaccount/services/party-data-management/party-profile

```
post https://blue.www.olb-qal0.dev.bmo.com/launch-openaccount/services/party-data-management/party-profile
x-request-id: 13
Authorization: Bearer eyJhbGciOiJSU0EtT0FFUC0yNTYiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2Iiwi21kIjoiYmVjTzd5Z
1B5RkFvcldiekhUSVBmbzBcd3NraGpDeU42T2VpemRrNldvQSIsImN0eSi6IkpxVCJ9.W191YUnt0WVC6y1fTPqT2ulRwC2MItW
GFsnqV_tx8ZXaCE3wE-YGRqCC9w0Tb0L6hTInJrBLV1VLiAYd8sGkl_a8ZWielwCY_5oA9LfjO_m8YGW2MJ7oQKLG9d_UrZqLnU
9YcTzrpi9Kfd3HD1DimgjXQlhfa3thSWoA749g4o-CJZWst670dFeltzq4litSTGK5EkCVP1NQzDTFV3yhIBIpJInUP0kKOqL2
fNN9gy4ox7HskU4XkzYGD6zextNoNx2KsYZJleK_JLcnOYEAc11mXhZz20dHP0n9WFd5oPompK90My9e40Vzkt3_nJ2f71ggJU
WIEPvSSaSyukNw_5C35SkGatd0XwgvEf3D10A.DpiNCEcleb2ah7mDfj8mhWs4QHsh6eKqp8wsozQ9USM1YW97N01WNAnEwWx
cC_R9UUthDNHZVupN19J0FBjD_f2mpEgKrKnhDTYiFmHWMqq-F2sbeadPQ2qn3Rmr_jbb300ah2LqW0p8V2uGWCuiRar-zgVt6s
SQK9QCbYrvht_jzMpFnDjXvQXY76oj7NYR1WM_KUTGm6EeTd0yWQ2eyJtYFeeq_9qX25KCOCOTrpuJKpuYW2sr7WD90AO0RV1QSt
eyDkRxyEQ-eHvvi1XUKABJWGY4x93LnAn-3d6HXC2vV7zyDk4WPZzNg-wT1yn_TUETpq22zIHv72r803nU4hFDW08JsHq8j5o
UJ83bxtnvfaHtxwZOB-F6dosgTIXBtgVn9B6t5hVlzw1RAigFd-aPoD2UqrUo3RpPc8cz-Os2fo7-2rQ1Sz7cWXo5FRO_CAu3Hp
Xw04nsGK9z6s55geFLkCz4uURFTVI6157GMxMidhTio8APIqfjQPfaH2X5rzJEuLoqwe3xUtK477wf4GvULHVItyJA1NcIh1NEv4
OonAYV7xVvsnu7HJ5RnExWbjhHiETFleTiiR1A_t5vqdd6d-JE4M5uNbje6nHr1RZz8eVdf62CetxVOMy8pgAJJKePsB
eAVasjc7kgf-hX3fyFqrifS-GzjC8Tl1G2hzEqV-Teii-SBybKuT7wmASostQoRgFpt8IA1lyvTxAFiZoBleuhk_BDOrTydFID
VA1E1VIotqm-hT36uOscKuvB5_P21ZyxfMZDcufzkWpcVmyps4pb5gHXazO37rYWFZRCT1tg297Jbo62gXF6DC0x-GArEUuWGoSx
OnBiM2Hu1hjTLV5-sleY4nK0YndFDTwzZuhBAg3mCoKHJTGj8G_4opArvoCeZskBOEWU08iGp_x_PS8xyHNH1Mu8RVvnCKejZ8M
AukNMOKd2CEKa2MCSAJ2XRA4zdbx0syccb7xDaj77fBy5ifAF9v27pvgoBDGNB25PbHyyKS-4A00Hjh1_wUV3uNO-pqy6xMNEY
ZNHMDqFcOrmhjtU98x8CftGZ0ND4ZJHsVs29cobM_UiEhnDPxxsQDMPqguWDVoJ2UJMvkaIs-4evk7wlwrKcFY402xM78i1IEgXp
jy03qGb6WH3bnsN-Nxt6Puhrs9M07JA2gbXewIjDyBX7ff2Uqs_fDWXH0g2zGJ4jC8G4KjAa19iHL9J5B0PEUQa0_9qdzMRG
Zfa_pfuhpYJUd3go0wjJ4i2ZKVUMPsSrJCsrlj4Uhg5Mk6hRcx8Mf60cnIMae_i92KZqdubKn8ZoIKGxRJLLejYskly6Udn12
gMgAmAgN1b39oj-HthB_3YCNGap4GNY1f1d5_AaPF8Mua9oFd0TGNF8iTRy3Q0zhKHNgWDcSw7Suddlo5gieCcrUjx3721YrRF3
0F6UOQFAVa9A-N0aUfLqw6Bz2e17TV7JdaJn4Pxj7KRnjX1aNKD25W6yvp_C-Z5kkMIw3JTA3Czcra9q3SDZr1c0CQRD10_p0lf
hJ8S-2b7zNrYK5uafH1nMgBsyV_6Fh07pjqMwkru9_Eu7E9Q3DoxXK4E5u9D9mBN1EUzZMqBr7wYj0ac196LL9WVFJTir9f
t24vlC4RxxwlUwlpj_DvND2z19KExfp7sDOpMICADGMGcjtNbqD7Itghy0MTmKQeWu920ECoHpa96TswMb90cCtpk0HaqxnijhN
NVWLTeMpy8htrA6CrLCeY348TofyXTuy-ZieuGJgcsfYZNTs_jVq7bDSWMAu1J3M5njjmU2QOCR2j5_w8Q413M7SU3-tPqTrBl_k
ZIGJtMoQJ6ir2eNjIePi1X168QKB8VWIMzBHSYMi16P8EparsHoUhRvizi_UalaiFAYKRL-HKZVnI05arsZWqThLmZd7EY1Eqc
EbKhTCNhJkiAiBu_4giqOeVsVhDP5cnom0kWtNoJgwvfq3qcmhEnUIdC3udNMqRKjwGBTk9Hz1VquyfSez99aq-Wv3-Y9RsX8GP
N5ExGZwd6Jvt_IP8IbiqoEPtPm0p4qrck-XtbgaiV6VIS6yap9Z3G8zSJXuz20g66NZWApa-nzfbIjwEoBOSczWINxUYNWxIj50
xxOA_iwLRMu7dvlkxwBWLQX1j03DdoYfmjwmZ6paIjFwVszaCpCYhd8x8bWVCC93EtX0uSgeG3yCIVFA1PHkvsPRZ3NfLgSVwR
jyQzIkxWWXESkfKXC0C1Af0tRqwmIQ75t0ri664wCn9MfaFpeox7wnjJ_TJfqyEK1-ar7sia1skc7IAaEPndusw6WR2mW5Ax0
vpFFHceLpQYEDHonaizTTeJ11hmromU1heiUlKeAc9C2JUysGZJtB-SHuz8k_gNaw0yTt1ExDxf6hnB91a3FepqjM0ty6Dc37Wd
dM_8LxjNF8FMzJWS0901IdA0iRbd9snQ_sg6kvyydT10_JdrhDAkWx11ekvNQjkAqqoZ7ucgJondeJw-BwLtrKv0sb0Eik7YYES
6b4pV6uu5XaCH0HrevPlhs0X1D1CQIE6USg4cWcdKFBaew7z8WIkV1JxGk8Z28tWEsQbbf548yOmZWPFRDYL1VeuMwYbOn98W0
F08-Pgsy8MQ3pE0mG8RLH5WUSBASqqB1DQNX0j5SKrSE1YVf08LcA5MxJN2eUwxJNU3JSP0Kjk1h4UAHMQYt1gm__9_dhCm2Yw
ZnfIvyjgZ-3VDjpc-QNpYe5x88iQNNzib88pKp6bstMtVfrAP78iPnIL-hB-T54Aar3jpXDeepCQ0R1LXywgHibnvVs-xUtsP
w_GxbF4ghRLWiGj3pYo9EN8wXD1IhiovwULga9ONY25iAEQCFVV8pD4-W2g95Ehy2joJyid0CTtpTyzAp1Udx49E9e1VZ3cd7UN
0JffFwvzrLenUM3z7VGknaywCl_m2EEBWhRCqq-_HBGBb7wtz17WFofg.gdv71NNdbHz_r2PE0uOlSw
x-fapi-interaction-id: 001
x-client-id: 001
x-api-key: 81f64d142eec79f3c669faa52023ef87-pr1
content-type: application/json
x-apigw-api-id: md8iavaxf536
```

```
{"journeyEvent": {"eventId": "E2"}, "originatorData": {"channel": "MOB"}, "personalInfo": {"birthDate": "1999-01-01", "firstName": "TestMockKYCPassww", "lastName": "TestSurnameeds", "uuid": "2c706b0c-2c6d-4943-a16f-2372080f366e"}}
```

RESPONSE

200

```
HTTP 200
x-request-id: 13
x-global-transaction-id: d1f1e4cd693c82210261f070
date: Fri, 12 Dec 2025 20:59:14 GMT
server: Server
x-amz-apigw-id: Vf1FUHGU4osF4yQ=
x-amzn-requestid: 44f092de-3101-4fda-b32e-bea0ba06a9ed
vary: Accept-Encoding
x-ratelimit-limit: name=default,800;
x-ratelimit-remaining: name=default,746;
x-amzn-trace-id: Root=1-693c8222-755deaf098dbdda1924dffcf
connection: keep-alive
content-type: application/json; charset=utf-8
akamai-grn: 0.74643017.1765573153.c27af73
```

```
{"status": "Success", "uuid": "2c706b0c-2c6d-4943-a16f-2372080f366e", "message": "No record is found", "requestId": "13"}
```

secure

MEDIUM (CVSS: 5.0)

Category: cookies
Method: get /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/party-product-selection
Last Detected: December 12, 2025 at 08:53 PM
OWASP: API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your cookies are correctly configured. We examine the `Set-Cookie` header in the API responses, if present, and check if it's correctly configured.

It's generally not recommended to use cookies with APIs for credentials and sensitive data, as there are better alternatives to handle authentication, such as JSON Web Tokens.

Incorrectly configured cookies make your API vulnerable to credentials leakage via cross-site scripting (XSS) and other form of attacks.

Secure cookie configuration requires an expiration date for the cookie, after which the browser deletes its associated data. We set the cookie's expiration date using one of the following attributes:

- **Secure**: it forces the browser to send cookies over HTTPS. Missing this attribute means exposing credentials over non-encrypted traffic. Hence, the absence of this attribute causes the test to raise a detection with CVSS 7.
- **HttpOnly**: this attribute prevents cookies from being accessed or modified by JavaScript code. Missing this attribute means malicious could seize the cookie and use it for malicious purposes. Hence, missing this attribute raises a detection with CVSS 7.
- **Expires**: the expiration date of the cookie.
- **Max-Age**: the time, in seconds, for which the browser must keep the cookie data stores.

`Expires` and `Max-Age` control how long the cookies will be stored in the browser. Without these attributes, the cookies remain in the browser until the browser session closes completely. If your `Set-Cookies` header doesn't include one of these attributes, the test raises an informational detection with CVSS 7.

It is also best practice to restrict which domains can receive the cookie. You can constrain the domains using one of the following fields:

- **Domain**: specifies the domain and associated subdomains that can receive the cookie. The browser will not send cookies to other domains.
- **SameSite**: this attribute controls whether and how the browser forwards cookies to other sites. It has three options: 1) `Strict` only includes cookies in requests originating from your own website; 2) `Lax` includes cookies in redirects to other websites; 3) `None` includes cookies in all requests to other websites. By default, SameSite's value is Lax (SameSite=Lax), which means cookies are only forwarded on top-level redirects, e.g. when we click on a link to go to another site. A threat actor could exploit this feature if they find a way to insert a malicious link within your website. If this attribute is missing or has a permissive configuration, the test raises an information detection with CVSS 5.

To learn more about securely configuring cookies, check out the following resource: ["Using HTTP cookies"](<https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies>).

Failing Test Logs

REQUEST

get /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/p...

```
get https://blue.www.olb-qal0.dev.bmo.com/launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/party-product-selection
x-request-id: 13
x-fapi-interaction-id: 001
x-client-id: 001
x-api-key: 81f64d142eec79f3c669faa52023ef87-pr1
product_id: 1
content-type: application/json
offer_id: 1
x-apigw-api-id: srgbjeqrz2

null
```

RESPONSE

400

```
HTTP 400
x-global-transaction-id: d1f1e4cd693c8349026224c0
date: Fri, 12 Dec 2025 21:04:09 GMT
content-length: 134
connection: keep-alive
content-type: application/json
akamai-grn: 0.56643717.1765573449.la6f97c3

{"httpCode": "400", "httpMessage": "Bad Request", "moreInformation": "One or more required API parameters are missing in the API request."}
```

/launch-openaccount/services/session-dialogue/onboarding-journey/init (1 finding)

secure

MEDIUM (CVSS: 5.0)

Category: cookies

Method: get /launch-openaccount/services/session-dialogue/onboarding-journey/init

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your cookies are correctly configured. We examine the `Set-Cookie` header in the API responses, if present, and check if it's correctly configured.

It's generally not recommended to use cookies with APIs for credentials and sensitive data, as there are better alternatives to handle

authentication, such as JSON Web Tokens.

Incorrectly configured cookies make your API vulnerable to credentials leakage via cross-site scripting (XSS) and other form of attacks.

Secure cookie configuration requires an expiration date for the cookie, after which the browser deletes its associated data. We set the cookie's expiration date using one of the following attributes:

- **Secure**: it forces the browser to send cookies over HTTPS. Missing this attribute means exposing credentials over non-encrypted traffic. Hence, the absence of this attribute causes the test to raise a detection with CVSS 7.
- **HttpOnly**: this attribute prevents cookies from being accessed or modified by JavaScript code. Missing this attribute means malicious could seize the cookie and use it for malicious purposes. Hence, missing this attribute raises a detection with CVSS 7.
- **Expires**: the expiration date of the cookie.
- **Max-Age**: the time, in seconds, for which the browser must keep the cookie data stores.

`Expires` and `Max-Age` control how long the cookies will be stored in the browser. Without these attributes, the cookies remain in the browser until the browser session closes completely. If your `Set-Cookies` header doesn't include one of these attributes, the test raises an informational detection with CVSS 7.

It is also best practice to restrict which domains can receive the cookie. You can constrain the domains using one of the following fields:

- **Domain**: specifies the domain and associated subdomains that can receive the cookie. The browser will not send cookies to other domains.
- **SameSite**: this attribute controls whether and how the browser forwards cookies to other sites. It has three options: 1) `Strict` only includes cookies in requests originating from your own website; 2) `Lax` includes cookies in redirects to other websites; 3) `None` includes cookies in all requests to other websites. By default, SameSite's value is Lax (SameSite=Lax), which means cookies are only forwarded on top-level redirects, e.g. when we click on a link to go to another site. A threat actor could exploit this feature if they find a way to insert a malicious link within your website. If this attribute is missing or has a permissive configuration, the test raises an information detection with CVSS 5.

To learn more about securely configuring cookies, check out the following resource: ["Using HTTP cookies"](<https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies>).

Failing Test Logs

REQUEST get /launch-openaccount/services/session-dialogue/onboarding-journey/init

```
get https://blue.www.olb-qa10.dev.bmo.com/launch-openaccount/services/session-dialogue/onboarding-journey/init
x-request-id: 10
x-fapi-interaction-id: 001
x-client-id: 001
x-api-key: 81f64d142eec79f3c669faa52023ef87-prl
content-type: application/json

null
```

RESPONSE 400

```
HTTP 400
x-global-transaction-id: d1f1e4cd693c818601b65873
date: Fri, 12 Dec 2025 20:56:38 GMT
content-length: 134
connection: keep-alive
content-type: application/json
akamai-grn: 0.74643017.1765572998.clea15f

{"httpCode": "400", "httpMessage": "Bad Request", "moreInformation": "One or more required API parameters are missing in the API request."}
```

Informational Findings

40 informational findings detected

/launch-openaccount/services/party-data-management/party-profile (6 info findings)

hsts

INFO

Category: headers

Method: patch /launch-openaccount/services/party-data-management/party-profile

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category: headers

Method: patch /launch-openaccount/services/party-data-management/party-profile

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

API4:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The

RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

hsts

INFO

Category: headers

Method: post /launch-openaccount/services/party-data-management/party-profile

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category: headers

Method: post /launch-openaccount/services/party-data-management/party-profile

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

API4:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

hsts

INFO

Category: headers

Method: get /launch-openaccount/services/party-data-management/party-profile

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category: headers

Method: get /launch-openaccount/services/party-data-management/party-profile

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

API4:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

/launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/pa...

hsts

INFO

Category:	headers
Method:	patch /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/party-product-selection
Last Detected:	December 12, 2025 at 08:53 PM
OWASP:	
API8:2023	

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

network

INFO

Category:	infoLeak
Method:	patch /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/party-product-selection
Last Detected:	December 12, 2025 at 08:53 PM
OWASP:	
API8:2023	

Remediation:**Solution**

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API response headers contain any network details that could potentially be revealing information about internal network topology. The test analyzes all the headers in the response looking for IPv4 and IPv6 patterns.

Revealing internal network configuration allows threat actors to gain insights about your network topology. Such information can be leveraged in SSRF attacks to target specific addresses within your network, or in a server breach to achieve lateral movement between networks.

hsts**INFO**

Category:	headers
Method:	post /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/party-product-selection
Last Detected:	December 12, 2025 at 08:53 PM
OWASP:	
API8:2023	

Remediation:**Solution**

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

hsts**INFO**

Category:	headers
Method:	get /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/party-product-selection
Last Detected:	December 12, 2025 at 08:53 PM
OWASP:	
API8:2023	

Remediation:**Solution**

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category:	headers
Method:	get /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/party-product-selection
Last Detected:	December 12, 2025 at 08:53 PM
OWASP:	
API8:2023	
API4:2023	

Remediation:**Solution**

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

hsts

INFO

Category: headers

Method: post /launch-openaccount/services/party-data-management/party-profile/address-compliance/get

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category: headers

Method: post /launch-openaccount/services/party-data-management/party-profile/address-compliance/get

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

API4:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
 - RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
 - RateLimit-Reset: specifies how much time is left in the current time window.
- You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

network

INFO

Category: infoLeak
Method: post /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/status
Last Detected: December 12, 2025 at 08:53 PM
OWASP: API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API response headers contain any network details that could potentially be revealing information about internal network topology. The test analyzes all the headers in the response looking for IPv4 and IPv6 patterns.

Revealing internal network configuration allows threat actors to gain insights about your network topology. Such information can be leveraged in SSRF attacks to target specific addresses within your network, or in a server breach to achieve lateral movement between networks.

/launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/st...

hsts

INFO

Category: headers
Method: patch /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/status
Last Detected: December 12, 2025 at 08:53 PM
OWASP: API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category:	headers
Method:	patch /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/status
Last Detected:	December 12, 2025 at 08:53 PM
OWASP:	
API8:2023	
API4:2023	

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

/session-dialogue/onboarding-journey/status (2 info findings)

hsts

INFO

Category:	headers
Method:	post /session-dialogue/onboarding-journey/status
Last Detected:	December 12, 2025 at 08:53 PM
OWASP:	
API8:2023	

Remediation:**Solution**

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category:	headers
Method:	post /session-dialogue/onboarding-journey/status
Last Detected:	December 12, 2025 at 08:53 PM

Remediation:**Solution**

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

hsts

INFO

Category: headers

Method: post /launch-openaccount/services/session-dialogue/prospect-product-selection/bmo-web/get

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

network

INFO

Category: infoLeak

Method: post /launch-openaccount/services/session-dialogue/prospect-product-selection/bmo-web/get

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API response headers contain any network details that could potentially be revealing information about internal network topology. The test analyzes all the headers in the response looking for IPv4 and IPv6 patterns.

Revealing internal network configuration allows threat actors to gain insights about your network topology. Such information can be leveraged in SSRF attacks to target specific addresses within your network, or in a server breach to achieve lateral movement between networks.

hsts

INFO

Category: headers

Method: post /launch-openaccount/services/session-dialogue/onboarding-journey/registered-device

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category: headers

Method: post /launch-openaccount/services/session-dialogue/onboarding-journey/registered-device

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

API4:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
 - RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
 - RateLimit-Reset: specifies how much time is left in the current time window.
- You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

/launch-openaccount/services/session-dialogue/onboarding-journey/delivery-target-v...

hsts

INFO

Category: headers
Method: post /launch-openaccount/services/session-dialogue/onboarding-journey/delivery-target-verification
Last Detected: December 12, 2025 at 08:53 PM
OWASP:
API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category: headers
Method: post /launch-openaccount/services/session-dialogue/onboarding-journey/delivery-target-verification
Last Detected: December 12, 2025 at 08:53 PM
OWASP:
API8:2023
API4:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

/launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/pa...

hsts

INFO

Category: headers

Method: post /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/party-product-selection/validation

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

network

INFO

Category: infoLeak

Method: post /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/party-product-selection/validation

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:**Solution**

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API response headers contain any network details that could potentially be revealing information about internal network topology. The test analyzes all the headers in the response looking for IPv4 and IPv6 patterns.

Revealing internal network configuration allows threat actors to gain insights about your network topology. Such information can be leveraged in SSRF attacks to target specific addresses within your network, or in a server breach to achieve lateral movement between networks.

/launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/do...

hsts**INFO**

Category: headers

Method: post /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/documents/get

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:**Solution**

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category:	headers
Method:	post /launch-openaccount/services/product-sales-agreement/everyday-banking-agreement/documents/get
Last Detected:	December 12, 2025 at 08:53 PM
OWASP:	
API8:2023	
API4:2023	

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

/launch-openaccount/services/processing-order/id-verification-and-credit-risk-adju...

hsts

INFO

Category:	headers
Method:	post /launch-openaccount/services/processing-order/id-verification-and-credit-risk-adjudication
Last Detected:	December 12, 2025 at 08:53 PM
OWASP:	
API8:2023	

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category:	headers
Method:	post /launch-openaccount/services/processing-order/id-verification-and-credit-risk-adjudication
Last Detected:	December 12, 2025 at 08:53 PM
OWASP:	
API8:2023	
API4:2023	

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

/launch-openaccount/services/processing-order/kyc-evaluation-and-credit-risk-adjud...

hsts

INFO

Category:	headers
Method:	post /launch-openaccount/services/processing-order/kyc-evaluation-and-credit-risk-adjudication
Last Detected:	December 12, 2025 at 08:53 PM
OWASP:	
API8:2023	

Remediation:**Solution**

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category:	headers
Method:	post /launch-openaccount/services/processing-order/kyc-evaluation-and-credit-risk-adjudication
Last Detected:	December 12, 2025 at 08:53 PM
OWASP:	
API8:2023	
API4:2023	

Remediation:**Solution**

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

hsts

INFO

Category: headers
Method: get /session-dialogue/onboarding-journey/end
Last Detected: December 12, 2025 at 08:53 PM
OWASP: API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category: headers
Method: get /session-dialogue/onboarding-journey/end
Last Detected: December 12, 2025 at 08:53 PM
OWASP:
API8:2023
API4:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

/launch-openaccount/services/session-dialogue/onboarding-journey/status (2 info fi...)

hsts

INFO

Category: headers
Method: get /launch-openaccount/services/session-dialogue/onboarding-journey/status
Last Detected: December 12, 2025 at 08:53 PM
OWASP:
API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category: headers
Method: get /launch-openaccount/services/session-dialogue/onboarding-journey/status
Last Detected: December 12, 2025 at 08:53 PM
OWASP:
API8:2023
API4:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

/launch-openaccount/services/session-dialogue/onboarding-journey/init (2 info find...)

hsts

INFO

Category: headers

Method: get /launch-openaccount/services/session-dialogue/onboarding-journey/init

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

rateLimit

INFO

Category: headers

Method: get /launch-openaccount/services/session-dialogue/onboarding-journey/init

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

API4:2023

Remediation:**Solution**

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether the API includes the RateLimit header in the responses. If responses don't contain a rate limiting header, the test raises an informational detection.

It's best practice to implement rate-limiting policies for APIs, and to document those policies using the RateLimit header. The RateLimit header makes it clear for API consumers how often they can call the API, hence avoiding accidental misuse of the API. Rate limiting should be implemented on a per-endpoint or per-user-flow basis, with more sensitive endpoints having stricter rate-limit policies.

You can document your rate-limiting policy with the following headers:

- RateLimit-Limit: specifies how many requests are allowed for a given window of time.
- RateLimit-Remaining: specifies how many requests are left to the user in the current time window.
- RateLimit-Reset: specifies how much time is left in the current time window.

You can learn more about the RateLimit header with the 'RateLimit Header Fields for HTTP' document, by R. Polli and A. Martinez (<https://www.ietf.org/archive/id/draft-polli-ratelimit-headers-02.html>).

/launch-openaccount/services/contact-handler/prospect-product-selection/bmo-web (1...)

hsts**INFO**

Category: headers

Method: post /launch-openaccount/services/contact-handler/prospect-product-selection/bmo-web

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:**Solution**

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.

hsts

INFO

Category: headers

Method: get /launch-openaccount/services/session-dialogue/onboarding-journey/registered-device/2c706b0c-2c6d-4943-a16f-2372080f3

Last Detected: December 12, 2025 at 08:53 PM

OWASP:

API8:2023

Remediation:

Solution

1. Set cookies with Secure, HttpOnly, and SameSite attributes appropriately.
2. Avoid cookies for API credentials where possible; prefer token-based auth.
3. Shorten session lifetime and rotate tokens/identifiers regularly.

Exploitation

- Missing cookie attributes can enable session theft via XSS, downgrade, or cross-site request scenarios.

References

- MDN: Set-Cookie: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie>
- OWASP API Security Top 10: <https://owasp.org/www-project-api-security/>

Description:

This test checks whether your responses contain a header named HTTP 'Strict-Transport-Security' (HSTS) and whether it is correctly configured. Securely configured HSTS protect your users by forcing browsers to use HTTPS, hence mitigating the risk of data leaks in man-in-the-middle and other attacks.