INPT
المعهد الوطني للبريد والمواصلات
ⵜⴰⵙⴷⴰⵡⵉⵜ ⵏⵜⵉⵎⵓⵔⴰⵡⵉⵏ ⴷ ⵜⵉⵎⵓⵙⵙⵉⵍⵜ
Institut National des Postes et Télécommunications

2023

RAPPORT DE SCAN DES VULNÉRABILITÉS SELON OWASP TOP 10 2017

PRÉPARÉ PAR :

- SEKKAT Ayyoub
- SADIK Aymane

ENCADRÉ PAR :

- M. El Mostapha BELMEKKI

Acunetix by Invicti

Compliance Report

OWASP TOP 10 2017

Description

The primary aim of the OWASP Top 10 is to educate developers, designers, architects, managers, and organizations about the consequences of the most important web application security weaknesses. The Top 10 provides basic techniques to protect against these high risk problem areas - and also provides guidance on where to go from here.

Disclaimer

This document or any of its content cannot account for, or be included in any form of legal advice. The outcome of a vulnerability scan (or security evaluation) should be utilized to ensure that diligent measures are taken to lower the risk of potential exploits carried out to compromise data.

Legal advice must be supplied according to its legal context. All laws and the environments in which they are applied, are constantly changed and revised. Therefore no information provided in this document may ever be used as an alternative to a qualified legal body or representative.

A portion of this report is taken from OWASP's Top Ten 2017 Project document, that can be found at http://www.owasp.org.

Scan Detail

Target	http://localhost:5000/
Scan Type	Full Scan
Start Time	May 26, 2023, 7:35:46 PM GMT+1
Scan Duration	48 seconds
Requests	7253
Average Response Time	1ms
Maximum Response Time	1670ms

Compliance at a Glance

CATEGORY	
2	A1 Injection
2	A2 Broken Authentication
5	A3 Sensitive Data Exposure
0	A4 XML External Entity (XXE)
1	A5 Broken Access Control
4	A6 Security Misconfiguration
1	A7 Cross Site Scripting (XSS)
0	A8 Insecure Deserialization
3	A9 Using Components with Known Vulnerabilities
0	A10 Insufficient Logging and Monitoring

Detailed Compliance Report by Category

This section is a detailed report that explains each vulnerability found according to individual compliance categories.

A1 Injection

Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent an interpreter as part of a command or query. The attackerâ s hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.

SQL injection

SQL injection (SQLi) refers to an injection attack wherein an attacker can execute malicious SQL statements that control a web application's database server.

CWE

CWE-89

CVSS2

AV:N/AC:M/Au:N/C:P/I:P/A:P

Access Vector	Network
Access Complexity	Medium
Authentication	None
Confidentiality	Partial
Integrity Impact	Partial
Availability Impact	Partial

CVSS3

CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:N

Base Score	10
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	None
Scope	Changed
Confidentiality	High
Integrity Impact	High
Availability Impact	None

Impact

An attacker can use SQL injection to bypass a web application's authentication and authorization mechanisms and retrieve the contents of an entire database. SQLi can also be used to add, modify and delete records in a database, affecting data integrity. Under the right circumstances, SQLi can also be used by an attacker to execute OS commands, which may then be used to escalate an attack even further.

http://localhost:5000/login

URL encoded POST input **password** was set to **-1' OR 3*2*1=6 AND 000123=000123 --**

Tests performed:

- -1' OR 2+123-123-1=0+0+0+1 -- => **TRUE**
- -1' OR 3+123-123-1=0+0+0+1 -- => **FALSE**
- -1' OR 3*2<(0+5+123-123) -- => **FALSE**
- -1' OR 3*2>(0+5+123-123) -- => **FALSE**
- -1' OR 2+1-1+1=1 AND 000123=000123 -- => **FALSE**
- -1' OR 3*2=5 AND 000123=000123 -- => **FALSE**
- -1' OR 3*2=6 AND 000123=000123 -- => **TRUE**
- -1' OR 3*2*0=6 AND 000123=000123 -- => **FALSE**
- -1' OR 3*2*1=6 AND 000123=000123 -- => **TRUE**

Original value: **1**

Request

```
POST /login HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
X-Requested-With: XMLHttpRequest
Referer: http://localhost:5000/
Content-Length: 68
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
Host: localhost:5000
Connection: Keep-alive

password=-1'%20OR%203*2*1=6%20AND%20000123=000123%20--%20&username=1
```

http://localhost:5000/login

URL encoded POST input **username** was set to **-1' OR 3*2*1=6 AND 000337=000337 --**

Tests performed:

- -1' OR 2+337-337-1=0+0+0+1 -- => **TRUE**
- -1' OR 3+337-337-1=0+0+0+1 -- => **FALSE**
- -1' OR 3*2<(0+5+337-337) -- => **FALSE**
- -1' OR 3*2>(0+5+337-337) -- => **FALSE**
- -1' OR 2+1-1+1=1 AND 000337=000337 -- => **FALSE**

- -1' OR 3*2=5 AND 000337=000337 -- => **FALSE**
- -1' OR 3*2=6 AND 000337=000337 -- => **TRUE**
- -1' OR 3*2*0=6 AND 000337=000337 -- => **FALSE**
- -1' OR 3*2*1=6 AND 000337=000337 -- => **TRUE**

Original value: **1**

Request

```
POST /login HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
X-Requested-With: XMLHttpRequest
Referer: http://localhost:5000/
Content-Length: 68
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
Host: localhost:5000
Connection: Keep-alive

password=1&username=-1'%20OR%203*2*1=6%20AND%20000337=000337%20--%20
```

Recommendation

Use parameterized queries when dealing with SQL queries that contain user input. Parameterized queries allow the database to understand which parts of the SQL query should be considered as user input, therefore solving SQL injection.

Description

In order for an SQL injection attack to take place, the vulnerable website needs to directly include user input within an SQL statement. An attacker can then insert a payload that will be included as part of the SQL query and run against the database server.

The following server-side **pseudo-code** is used to authenticate users to the web application.

```
# Define POST variables
uname = request.POST['username']
passwd = request.POST['password']

# SQL query vulnerable to SQLi
sql = "SELECT id FROM users WHERE username='" + uname + "' AND password='" + passwd
+ "'"

# Execute the SQL statement
database.execute(sql)
```

The above script is a simple example of authenticating a user with a username and a password against a database with a table named users, and a username and password column.

The above script is vulnerable to SQL injection because an attacker could submit malicious input in such a way that would alter the SQL statement being executed by the database server.

A simple example of an SQL injection payload could be something as simple as setting the password field to `password' OR 1=1`.

This would result in the following SQL query being run against the database server.

```
SELECT id FROM users WHERE username='username' AND password='password' OR 1=1'
```

An attacker can also comment out the rest of the SQL statement to control the execution of the SQL query further.

```
-- MySQL, MSSQL, Oracle, PostgreSQL, SQLite
' OR '1'='1' --
' OR '1'='1' /*
-- MySQL
' OR '1'='1' #
-- Access (using null characters)
' OR '1'='1' %00
' OR '1'='1' %16
```

Once the query executes, the result is returned to the application to be processed, resulting in an authentication bypass. In the event of authentication bypass being possible, the application will most likely log the attacker in with the first account from the query result — the first account in a database is usually of an administrative user.

What's the worst an attacker can do with SQL?

SQL is a programming language designed for managing data stored in an RDBMS, therefore SQL can be used to access, modify and delete data. Furthermore, in specific cases, an RDBMS could also run commands on the operating system from an SQL statement.

Keeping the above in mind, when considering the following, it's easier to understand how lucrative a successful SQL injection attack can be for an attacker.

An attacker can use SQL injection to bypass authentication or even impersonate specific users.
One of SQL's primary functions is to select data based on a query and output the result of that query. An SQL injection vulnerability could allow the complete disclosure of data residing on a database server.
Since web applications use SQL to alter data within a database, an attacker could use SQL injection to alter data stored in a database. Altering data affects data integrity and could cause repudiation issues, for instance, issues such as voiding transactions, altering balances and other records.
SQL is used to delete records from a database. An attacker could use an SQL injection vulnerability to delete data from a database. Even if an appropriate backup strategy is employed, deletion of data could affect an application's availability until the database is restored.

Some database servers are configured (intentional or otherwise) to allow arbitrary execution of operating system commands on the database server. Given the right conditions, an attacker could use SQL injection as the initial vector in an attack of an internal network that sits behind a firewall.

Preventing SQL injection using parameterized queries

SQL injection is one of the most widely spread and most damaging web application vulnerabilities. Fortunately, both the programming languages, as well as the RDBMSs themselves have evolved to provide web application developers with a way to safely query the database — parameterized SQL queries.

Parameterized queries are simple to write and understand while forcing a developer to define the entire SQL statement before hand, using placeholders for the actual variables within that statement. A developer would then pass in each parameter to the query after the SQL statement is defined, allowing the database to be able to distinguish between the SQL command and data inputted by a user. If SQL commands are inputted by an attacker, the parameterized query would treat the input as a string as opposed to an SQL command.

Application developers should avoid sanitizing their input by means of escaping or removing special characters (several encoding tricks an attacker could leverage to bypass such protections) and stick to using parameterized queries in order to avoid SQL injection vulnerabilities.

References

SQL Injection (SQLi) - Acunetix
https://www.acunetix.com/websitesecurity/sql-injection/

Types of SQL Injection (SQLi) - Acunetix
https://www.acunetix.com/websitesecurity/sql-injection2/

Prevent SQL injection vulnerabilities in PHP applications and fix them - Acunetix
https://www.acunetix.com/blog/articles/prevent-sql-injection-vulnerabilities-in-php-applications/

SQL Injection - OWASP
https://www.owasp.org/index.php/SQL_Injection

Bobby Tables: A guide to preventing SQL injection
https://bobby-tables.com/

SQL Injection Cheet Sheets - Pentestmonkey
http://pentestmonkey.net/category/cheat-sheet/sql-injection

A2 Broken Authentication

Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities.

Unencrypted connection

This scan target was connected to over an unencrypted connection. A potential attacker can intercept and modify data sent and received from this site.

CWE

CWE-319

CVSS2

AV:N/AC:M/Au:N/C:P/I:P/A:N

Access Vector	Network
Access Complexity	Medium
Authentication	None
Confidentiality	Partial
Integrity Impact	Partial
Availability Impact	None

CVSS3

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Base Score	5.4
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Unchanged
Confidentiality	Low
Integrity Impact	Low
Availability Impact	None

Impact

Possible information disclosure.

---

http://localhost:5000/ Verified

Request

```
GET /stock HTTP/1.1
Host: localhost:5000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
```

```
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
cookie:
session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxLzE2FiirVAgB9Hxjp.ZHD8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
```

Recommendation

The site should send and receive data over a secure (HTTPS) connection.

User credentials are sent in clear text

User credentials are transmitted over an unencrypted channel. This information should always be transferred via an encrypted channel (HTTPS) to avoid being intercepted by malicious users.

CWE

CWE-523

CVSS2

AV:N/AC:L/Au:N/C:P/I:N/A:N

Access Vector	Network
Access Complexity	Low
Authentication	None
Confidentiality	Partial
Integrity Impact	None
Availability Impact	None

CVSS3

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:N/A:N

Base Score	4.3
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Unchanged
Confidentiality	Low
Integrity Impact	None
Availability Impact	None

Impact

A third party may be able to read the user credentials by intercepting an unencrypted HTTP connection.

---

http://localhost:5000/

Forms with credentials sent in clear text:

- http://localhost:5000/login

```
Form name: <empty>
Form action: /login
Form method: POST
Password input: password
```

Request

```
GET /login HTTP/1.1
Host: localhost:5000
Pragma: no-cache
Cache-Control: no-cache
accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
accept-language: en-US
upgrade-insecure-requests: 1
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
```

Recommendation

Because user credentials are considered sensitive information, should always be transferred to the server over an encrypted connection (HTTPS).

A3 Sensitive Data Exposure

Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires pecial precautions when exchanged with the browser.

Cookies with missing, inconsistent or contradictory properties

At least one of the following cookies properties causes the cookie to be invalid or incompatible with either a different property of the same cookie, of with the environment the cookie is being used in. Although this is not a vulnerability in itself, it will likely lead to unexpected behavior by the application, which in turn may cause secondary security issues.

CWE

CWE-284

CVSS2

AV:N/AC:L/Au:N/C:N/I:N/A:N

Access Vector	Network
Access Complexity	Low
Authentication	None
Confidentiality	None
Integrity Impact	None
Availability Impact	None

CVSS3

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:N/I:N/A:N

Base Score	0.0
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Unchanged
Confidentiality	None
Integrity Impact	None
Availability Impact	None

Impact

Cookies will not be stored, or submitted, by web browsers.

---

http://localhost:5000/ Verified

List of cookies with missing, inconsistent or contradictory properties:

- http://localhost:5000/login

  Cookie was set with:

  ```
  Set-Cookie:
  session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxLzE2FiirVAgB9Hxjp.ZHD8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw; HttpOnly; Path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and sometimes unexpected defaults. It is therefore recommended to add a SameSite attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

Request

```
POST /login HTTP/1.1
Host: localhost:5000
Content-Length: 31
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Origin: http://localhost:5000
Content-Type: application/x-www-form-urlencoded
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36

username=user&password=useruser
```

Recommendation

Ensure that the cookies configuration complies with the applicable standards.

References

MDN | Set-Cookie
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie

Securing cookies with cookie prefixes
https://www.sjoerdlangkemper.nl/2017/02/09/cookie-prefixes/

Cookies: HTTP State Management Mechanism
https://tools.ietf.org/html/draft-ietf-httpbis-rfc6265bis-05

SameSite Updates - The Chromium Projects
https://www.chromium.org/updates/same-site

draft-west-first-party-cookies-07: Same-site Cookies
https://tools.ietf.org/html/draft-west-first-party-cookies-07

Content Security Policy (CSP) not implemented

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks.

Content Security Policy (CSP) can be implemented by adding a **Content-Security-Policy** header. The value of this header is a string containing the policy directives describing your Content Security Policy. To implement CSP, you should define lists of allowed origins for the all of the types of resources that your site utilizes. For example, if you have a simple site that needs to load scripts, stylesheets, and images hosted locally, as well as from the jQuery library from their CDN, the CSP header could look like the following:

```
Content-Security-Policy:
default-src 'self';
script-src 'self' https://code.jquery.com;
```

It was detected that your web application doesn't implement Content Security Policy (CSP) as the CSP header is missing from the response. It's recommended to implement Content Security Policy (CSP) into your web application.

CWE

CWE-1021

CVSS2

AV:N/AC:L/Au:N/C:N/I:N/A:N

Access Vector	Network
Access Complexity	Low
Authentication	None
Confidentiality	None
Integrity Impact	None
Availability Impact	None

CVSS3

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:N/I:N/A:N

Base Score	0.0
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Changed
Confidentiality	None
Integrity Impact	None
Availability Impact	None

Impact

CSP can be used to prevent and/or mitigate attacks that involve content/code injection, such as cross-site scripting/XSS attacks, attacks that require embedding a malicious resource, attacks that involve malicious use of iframes, such as clickjacking attacks, and others.

---

http://localhost:5000/

Paths without CSP header:

- http://localhost:5000/stock
- http://localhost:5000/login

- http://localhost:5000/
- http://localhost:5000/supprimer/
- http://localhost:5000/modifier/

Request

```
GET /stock HTTP/1.1
Host: localhost:5000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
cookie:
session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxLzE2FiirVAgB9Hxjp.ZH
D8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
```

Recommendation

It's recommended to implement Content Security Policy (CSP) into your web application. Configuring Content Security Policy involves adding the **Content-Security-Policy** HTTP header to a web page and giving it values to control resources the user agent is allowed to load for that page.

References

Content Security Policy (CSP)
https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

Implementing Content Security Policy
https://hacks.mozilla.org/2016/02/implementing-content-security-policy/

Permissions-Policy header not implemented

The Permissions-Policy header allows developers to selectively enable and disable use of various browser features and APIs.

CWE
CWE-1021

CVSS2
AV:N/AC:L/Au:N/C:N/I:N/A:N

Access Vector	Network
Access Complexity	Low
Authentication	None
Confidentiality	None
Integrity Impact	None
Availability Impact	None

CVSS3
CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:N/I:N/A:N

Base Score	0.0
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Changed
Confidentiality	None
Integrity Impact	None
Availability Impact	None

Impact

---

http://localhost:5000/

Locations without Permissions-Policy header:

- http://localhost:5000/stock
- http://localhost:5000/login
- http://localhost:5000/
- http://localhost:5000/supprimer/
- http://localhost:5000/modifier/

Request

```
GET /stock HTTP/1.1
Host: localhost:5000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
```

```
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
cookie:
session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxLzE2FiirVAgB9Hxjp.ZH
D8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
```

References

Permissions-Policy / Feature-Policy (MDN)
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Feature-Policy

Permissions Policy (W3C)
https://www.w3.org/TR/permissions-policy-1/

Clickjacking: X-Frame-Options header

Clickjacking (User Interface redress attack, UI redress attack, UI redressing) is a malicious technique of tricking a Web user into clicking on something different from what the user perceives they are clicking on, thus potentially revealing confidential information or taking control of their computer while clicking on seemingly innocuous web pages.

The server did not return an **X-Frame-Options** header with the value DENY or SAMEORIGIN, which means that this website could be at risk of a clickjacking attack. The X-Frame-Options HTTP response header can be used to indicate whether or not a browser should be allowed to render a page inside a frame or iframe. Sites can use this to avoid clickjacking attacks, by ensuring that their content is not embedded into untrusted sites.

CWE
CWE-1021

CVSS2
AV:N/AC:M/Au:N/C:N/I:P/A:N

Access Vector	Network
Access Complexity	Medium
Authentication	None
Confidentiality	None
Integrity Impact	Partial
Availability Impact	None

CVSS3
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:N/I:L/A:N

Base Score	5.8
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	None
Scope	Changed

Confidentiality	None
Integrity Impact	Low
Availability Impact	None

Impact

The impact depends on the affected web application.

---

http://localhost:5000/

Paths without secure XFO header:

- http://localhost:5000/stock
- http://localhost:5000/login
- http://localhost:5000/
- http://localhost:5000/supprimer/
- http://localhost:5000/modifier/

Request

```
GET /stock HTTP/1.1
Host: localhost:5000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
cookie:
session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxLzE2FiirVAgB9Hxjp.ZH
D8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
```

Recommendation

Configure your web server to include an X-Frame-Options header and a CSP header with frame-ancestors directive. Consult Web references for more information about the possible values for this header.

References

The X-Frame-Options response header
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options

Clickjacking
https://en.wikipedia.org/wiki/Clickjacking

OWASP Clickjacking
https://cheatsheetseries.owasp.org/cheatsheets/Clickjacking_Defense_Cheat_Sheet.html

Frame Buster Buster
https://stackoverflow.com/questions/958997/frame-buster-buster-buster-code-needed

Unencrypted connection

This scan target was connected to over an unencrypted connection. A potential attacker can intercept and modify data sent and received from this site.

CWE

CWE-319

CVSS2

AV:N/AC:M/Au:N/C:P/I:P/A:N

Access Vector	Network
Access Complexity	Medium
Authentication	None
Confidentiality	Partial
Integrity Impact	Partial
Availability Impact	None

CVSS3

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Base Score	5.4
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Unchanged
Confidentiality	Low
Integrity Impact	Low
Availability Impact	None

Impact

Possible information disclosure.

http://localhost:5000/ Verified

Request

```
GET /stock HTTP/1.1
Host: localhost:5000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
cookie:
session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxLzE2FiirVAgB9Hxjp.ZH
D8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
```

Recommendation

The site should send and receive data over a secure (HTTPS) connection.

A4 XML External Entity (XXE)

Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.

No alerts in this category

A5 Broken Access Control

Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.

Clickjacking: X-Frame-Options header

Clickjacking (User Interface redress attack, UI redress attack, UI redressing) is a malicious technique of tricking a Web user into clicking on something different from what the user perceives they are clicking on, thus potentially revealing confidential information or taking control of their computer while clicking on seemingly innocuous web pages.

The server did not return an **X-Frame-Options** header with the value DENY or SAMEORIGIN, which means that this website could be at risk of a clickjacking attack. The X-Frame-Options HTTP response header can be used to indicate whether or not a browser should be allowed to render a page inside a frame or iframe. Sites can use this to avoid clickjacking attacks, by ensuring that their content is not embedded into untrusted sites.

CWE

CWE-1021

CVSS2

AV:N/AC:M/Au:N/C:N/I:P/A:N

Access Vector	Network
Access Complexity	Medium
Authentication	None
Confidentiality	None
Integrity Impact	Partial
Availability Impact	None

CVSS3

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:N/I:L/A:N

Base Score	5.8
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	None
Scope	Changed
Confidentiality	None
Integrity Impact	Low
Availability Impact	None

Impact

The impact depends on the affected web application.

---

http://localhost:5000/

Paths without secure XFO header:

- http://localhost:5000/stock
- http://localhost:5000/login

- http://localhost:5000/
- http://localhost:5000/supprimer/
- http://localhost:5000/modifier/

Request

```
GET /stock HTTP/1.1
Host: localhost:5000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
cookie:
session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxLzE2FiirVAgB9Hxjp.ZH
D8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
```

Recommendation

Configure your web server to include an X-Frame-Options header and a CSP header with frame-ancestors directive. Consult Web references for more information about the possible values for this header.

References

The X-Frame-Options response header
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options

Clickjacking
https://en.wikipedia.org/wiki/Clickjacking

OWASP Clickjacking
https://cheatsheetseries.owasp.org/cheatsheets/Clickjacking_Defense_Cheat_Sheet.html

Frame Buster Buster
https://stackoverflow.com/questions/958997/frame-buster-buster-buster-code-needed

A6 Security Misconfiguration

Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securly configured, but they must be patched and upgraded in a timely fashion.

Cookies with missing, inconsistent or contradictory properties

At least one of the following cookies properties causes the cookie to be invalid or incompatible with either a different property of the same cookie, of with the environment the cookie is being used in. Although this is not a vulnerability in itself, it will likely lead to unexpected behavior by the application, which in turn may cause secondary security issues.

CWE
CWE-284

CVSS2
AV:N/AC:L/Au:N/C:N/I:N/A:N

Access Vector	Network
Access Complexity	Low
Authentication	None
Confidentiality	None
Integrity Impact	None
Availability Impact	None

CVSS3
CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:N/I:N/A:N

Base Score	0.0
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Unchanged
Confidentiality	None
Integrity Impact	None
Availability Impact	None

Impact

Cookies will not be stored, or submitted, by web browsers.

---

http://localhost:5000/ Verified

List of cookies with missing, inconsistent or contradictory properties:

- http://localhost:5000/login

  Cookie was set with:

  ```
  Set-Cookie:
  session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxL
  zE2FiirVAgB9Hxjp.ZHD8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw; HttpOnly; Path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

Request

```
POST /login HTTP/1.1
Host: localhost:5000
Content-Length: 31
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Origin: http://localhost:5000
Content-Type: application/x-www-form-urlencoded
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36

username=user&password=useruser
```

Recommendation

Ensure that the cookies configuration complies with the applicable standards.

References

MDN | Set-Cookie

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie
Securing cookies with cookie prefixes
https://www.sjoerdlangkemper.nl/2017/02/09/cookie-prefixes/

Cookies: HTTP State Management Mechanism
https://tools.ietf.org/html/draft-ietf-httpbis-rfc6265bis-05

SameSite Updates - The Chromium Projects
https://www.chromium.org/updates/same-site

draft-west-first-party-cookies-07: Same-site Cookies
https://tools.ietf.org/html/draft-west-first-party-cookies-07

Content Security Policy (CSP) not implemented

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks.

Content Security Policy (CSP) can be implemented by adding a **Content-Security-Policy** header. The value of this header is a string containing the policy directives describing your Content Security Policy. To implement CSP, you should define lists of allowed origins for the all of the types of resources that your site utilizes. For example, if you have a simple site that needs to load scripts, stylesheets, and images hosted locally, as well as from the jQuery library from their CDN, the CSP header could look like the following:

```
Content-Security-Policy:
default-src 'self';
script-src 'self' https://code.jquery.com;
```

It was detected that your web application doesn't implement Content Security Policy (CSP) as the CSP header is missing from the response. It's recommended to implement Content Security Policy (CSP) into your web application.

CWE
CWE-1021

CVSS2
AV:N/AC:L/Au:N/C:N/I:N/A:N

Access Vector	Network
Access Complexity	Low
Authentication	None
Confidentiality	None
Integrity Impact	None
Availability Impact	None

CVSS3
CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:N/I:N/A:N

Base Score	0.0
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Changed

Confidentiality	None
Integrity Impact	None
Availability Impact	None

Impact

CSP can be used to prevent and/or mitigate attacks that involve content/code injection, such as cross-site scripting/XSS attacks, attacks that require embedding a malicious resource, attacks that involve malicious use of iframes, such as clickjacking attacks, and others.

---

http://localhost:5000/

Paths without CSP header:

- http://localhost:5000/stock
- http://localhost:5000/login
- http://localhost:5000/
- http://localhost:5000/supprimer/
- http://localhost:5000/modifier/

Request

```
GET /stock HTTP/1.1
Host: localhost:5000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
cookie:
session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxLzE2FiirVAgB9Hxjp.ZH
D8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
```

```
Chrome/108.0.0.0 Safari/537.36
```

Recommendation

It's recommended to implement Content Security Policy (CSP) into your web application. Configuring Content Security Policy involves adding the **Content-Security-Policy** HTTP header to a web page and giving it values to control resources the user agent is allowed to load for that page.

References

Content Security Policy (CSP)
https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

Implementing Content Security Policy
https://hacks.mozilla.org/2016/02/implementing-content-security-policy/

Permissions-Policy header not implemented

The Permissions-Policy header allows developers to selectively enable and disable use of various browser features and APIs.

CWE

CWE-1021

CVSS2

AV:N/AC:L/Au:N/C:N/I:N/A:N

Access Vector	Network
Access Complexity	Low
Authentication	None
Confidentiality	None
Integrity Impact	None
Availability Impact	None

CVSS3

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:N/I:N/A:N

Base Score	0.0
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Changed
Confidentiality	None
Integrity Impact	None
Availability Impact	None

Impact

---

http://localhost:5000/

Locations without Permissions-Policy header:

- http://localhost:5000/stock
- http://localhost:5000/login
- http://localhost:5000/
- http://localhost:5000/supprimer/
- http://localhost:5000/modifier/

Request

```
GET /stock HTTP/1.1
Host: localhost:5000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
cookie:
session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxLzE2FiirVAgB9Hxjp.ZH
D8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
```

References

Permissions-Policy / Feature-Policy (MDN)
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Feature-Policy

Permissions Policy (W3C)
https://www.w3.org/TR/permissions-policy-1/

No HTTP Redirection

It was detected that your web application uses HTTP protocol, but doesn't automatically redirect users to HTTPS.

CWE
CWE-16

CVSS2
AV:N/AC:L/Au:N/C:N/I:N/A:N

CVSS3
CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:N/I:N/A:N

Access Vector	Network
Access Complexity	Low
Authentication	None
Confidentiality	None
Integrity Impact	None
Availability Impact	None

Base Score	0.0
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Changed
Confidentiality	None
Integrity Impact	None
Availability Impact	None

Impact

In some circumstances, it could be used for a man-in-the-middle (MitM) attack

---

http://localhost:5000/

Request

```
GET / HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
Host: localhost:5000
Connection: Keep-alive
```

Recommendation

It's recommended to implement best practices of HTTP Redirection into your web application. Consult web references for more information

References

HTTP Redirections
https://infosec.mozilla.org/guidelines/web_security#http-redirections

A7 Cross Site Scripting (XSS)

XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.

Cross site scripting

Cross-site Scripting (XSS) refers to client-side code injection attack wherein an attacker can execute malicious scripts into a legitimate website or web application. XSS occurs when a web application makes use of unvalidated or unencoded user input within the output it generates.

CWE

CWE-79

CVSS2

AV:N/AC:L/Au:N/C:P/I:P/A:N

Access Vector	Network
Access Complexity	Low
Authentication	None
Confidentiality	Partial
Integrity Impact	Partial
Availability Impact	None

CVSS3

CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:N

Base Score	5.3
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	None
Scope	Unchanged
Confidentiality	None
Integrity Impact	Low
Availability Impact	None

Impact

Malicious JavaScript has access to all the same objects as the rest of the web page, including access to cookies and local storage, which are often used to store session tokens. If an attacker can obtain a user's session cookie, they can then impersonate that user.

Furthermore, JavaScript can read and make arbitrary modifications to the contents of a page being displayed to a user. Therefore, XSS in conjunction with some clever social engineering opens up a lot of possibilities for an attacker.

---

http://localhost:5000/stock Verified

URL encoded GET input **search_query** was set to **the'"()&%<zzz><ScRiPt >i0ls(9731)</ScRiPt>**

Request

```
GET /stock?search_query=the'"()%26%25<zzz><ScRiPt%20>i0ls(9731)</ScRiPt> HTTP/1.1
Referer: http://localhost:5000/
Cookie:
session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxLzE2FiirVAgB9Hxjp.ZHD8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0 Safari/537.36
Host: localhost:5000
Connection: Keep-alive
```

Recommendation

Apply context-dependent encoding and/or validation to user input rendered on a page

Description

In order for a Cross-site scripting (XSS) attack to take place, an attacker does not directly target a victim. Instead, an attacker exploits a vulnerability in a web application visited by a victim, where the web application is used to deliver the malicious JavaScript. The victim's browser is not able to distinguish between malicious and legitimate JavaScript, and therefore, executes the attacker's malicious payload.

Since cross-site scripting (XSS) is user input which is interpreted as code. In order to prevent XSS, secure input handling is necessary. The two fundamental methods of handling untrusted user input are **encoding** and **validation**.

Encoding - Escapes user input so that browsers interpret it as **data**, not as code
Validation - Filters user input so that browsers interpret it as code without malicious commands

Encoding and validation are two different techniques to preventing cross-site scripting (XSS). Deciding which should be used highly depends on the **context** within which the untrusted user input is being inserted.

The following are two examples of the most common cross-site scripting (XSS) contexts.

```
<!-- HTML element -->
<div>userInput</div>

<!-- HTML attribute -->
<input value="userInput">
```

The method for preventing cross-site (XSS) scripting in the two examples above is different. In the first example, where user input is inserted in an HTML element, HTML encoding is the correct way to prevent XSS. However, in the second example, where user input is inserted in an HTML attribute, validation (in this case, filtering out `'` and `"`)is the appropriate prevention method.

```
<!-- Application code -->
<input value="userInput">
```

```
<!-- Malicious string -->
"><script>...</script><input value="

<!-- Resulting code -->
<input value=""><script>...</script><input value="">
```

In **most** of the time, encoding should be performed whenever user input is included in a page, however, as with the above example, in some cases, encoding has to be replaced by or complemented with validation.

It's important to remember that secure input handling has to take into account which context of a page the user input is inserted into.

References

Cross-site Scripting (XSS) Attack - Acunetix
https://www.acunetix.com/websitesecurity/cross-site-scripting/

Types of XSS - Acunetix
https://www.acunetix.com/websitesecurity/xss/

XSS Filter Evasion Cheat Sheet
https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet

Excess XSS, a comprehensive tutorial on cross-site scripting
https://excess-xss.com/

Cross site scripting
https://en.wikipedia.org/wiki/Cross-site_scripting

A8 Insecure Deserialization

Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.

No alerts in this category

A9 Using Components with Known Vulnerabilities

Components, such as libraries, frameworks, and other software modules, almost always run with full privileges. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server

takeover. Applications using components with known vulnerabilities may undermine application defenses and enable a range of possible attacks and impacts.

Cookies with missing, inconsistent or contradictory properties

At least one of the following cookies properties causes the cookie to be invalid or incompatible with either a different property of the same cookie, of with the environment the cookie is being used in. Although this is not a vulnerability in itself, it will likely lead to unexpected behavior by the application, which in turn may cause secondary security issues.

CWE

CWE-284

CVSS2

AV:N/AC:L/Au:N/C:N/I:N/A:N

Access Vector	Network
Access Complexity	Low
Authentication	None
Confidentiality	None
Integrity Impact	None
Availability Impact	None

CVSS3

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:N/I:N/A:N

Base Score	0.0
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Unchanged
Confidentiality	None
Integrity Impact	None
Availability Impact	None

Impact

Cookies will not be stored, or submitted, by web browsers.

---

http://localhost:5000/ Verified

List of cookies with missing, inconsistent or contradictory properties:

- http://localhost:5000/login

  Cookie was set with:

  ```
  Set-Cookie:
  session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxL
  zE2FiirVAgB9Hxjp.ZHD8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw; HttpOnly; Path=/
  ```

This cookie has the following issues:

```
- Cookie without SameSite attribute.
When cookies lack the SameSite attribute, Web browsers may apply different and
sometimes unexpected defaults. It is therefore recommended to add a SameSite
attribute with an appropriate value of either "Strict", "Lax", or "None".
```

Request

```
POST /login HTTP/1.1
Host: localhost:5000
Content-Length: 31
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Origin: http://localhost:5000
Content-Type: application/x-www-form-urlencoded
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36

username=user&password=useruser
```

Recommendation

Ensure that the cookies configuration complies with the applicable standards.

References

MDN | Set-Cookie
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie

Securing cookies with cookie prefixes
https://www.sjoerdlangkemper.nl/2017/02/09/cookie-prefixes/

Cookies: HTTP State Management Mechanism
https://tools.ietf.org/html/draft-ietf-httpbis-rfc6265bis-05

SameSite Updates - The Chromium Projects

https://www.chromium.org/updates/same-site
draft-west-first-party-cookies-07: Same-site Cookies
https://tools.ietf.org/html/draft-west-first-party-cookies-07

Content Security Policy (CSP) not implemented

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks.

Content Security Policy (CSP) can be implemented by adding a **Content-Security-Policy** header. The value of this header is a string containing the policy directives describing your Content Security Policy. To implement CSP, you should define lists of allowed origins for the all of the types of resources that your site utilizes. For example, if you have a simple site that needs to load scripts, stylesheets, and images hosted locally, as well as from the jQuery library from their CDN, the CSP header could look like the following:

```
Content-Security-Policy:
default-src 'self';
script-src 'self' https://code.jquery.com;
```

It was detected that your web application doesn't implement Content Security Policy (CSP) as the CSP header is missing from the response. It's recommended to implement Content Security Policy (CSP) into your web application.

CWE
CWE-1021

CVSS2
AV:N/AC:L/Au:N/C:N/I:N/A:N

Access Vector	Network
Access Complexity	Low
Authentication	None
Confidentiality	None
Integrity Impact	None
Availability Impact	None

CVSS3
CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:N/I:N/A:N

Base Score	0.0
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Changed
Confidentiality	None
Integrity Impact	None
Availability Impact	None

Impact

CSP can be used to prevent and/or mitigate attacks that involve content/code injection, such as cross-site scripting/XSS attacks, attacks that require embedding a malicious resource, attacks that involve malicious use of iframes, such as clickjacking attacks, and others.

---

http://localhost:5000/

Paths without CSP header:

- http://localhost:5000/stock
- http://localhost:5000/login
- http://localhost:5000/
- http://localhost:5000/supprimer/
- http://localhost:5000/modifier/

Request

```
GET /stock HTTP/1.1
Host: localhost:5000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
cookie:
session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxLzE2FiirVAgB9Hxjp.ZH
D8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
```

Recommendation

It's recommended to implement Content Security Policy (CSP) into your web application. Configuring Content Security Policy involves adding the **Content-Security-Policy** HTTP header to a web page and giving it values to control resources the user agent is allowed to load for that page.

References

Content Security Policy (CSP)
https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

Implementing Content Security Policy
https://hacks.mozilla.org/2016/02/implementing-content-security-policy/

Permissions-Policy header not implemented

The Permissions-Policy header allows developers to selectively enable and disable use of various browser features and APIs.

CWE

CWE-1021

CVSS2

AV:N/AC:L/Au:N/C:N/I:N/A:N

Access Vector	Network
Access Complexity	Low
Authentication	None
Confidentiality	None
Integrity Impact	None
Availability Impact	None

CVSS3

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:N/I:N/A:N

Base Score	0.0
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Changed
Confidentiality	None
Integrity Impact	None
Availability Impact	None

Impact

---

http://localhost:5000/

Locations without Permissions-Policy header:

- http://localhost:5000/stock
- http://localhost:5000/login
- http://localhost:5000/
- http://localhost:5000/supprimer/
- http://localhost:5000/modifier/

Request

```
GET /stock HTTP/1.1
Host: localhost:5000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
Accept-Language: en-US
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
lication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
cookie:
session=.eJyrVsosjk9Myc3MU7Iy0FHKyU9PT02JB_FKikpTdZQKEouLy_OLUpSslEqLU4tAWEkHzMxLzE2FiirVAgB9Hxjp.ZH
D8Fw.BVqmmbTyk3P2UYcBjd_v_rrUDqw
Referer: http://localhost:5000/login
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.0.0 Safari/537.36
```

References

Permissions-Policy / Feature-Policy (MDN)
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Feature-Policy

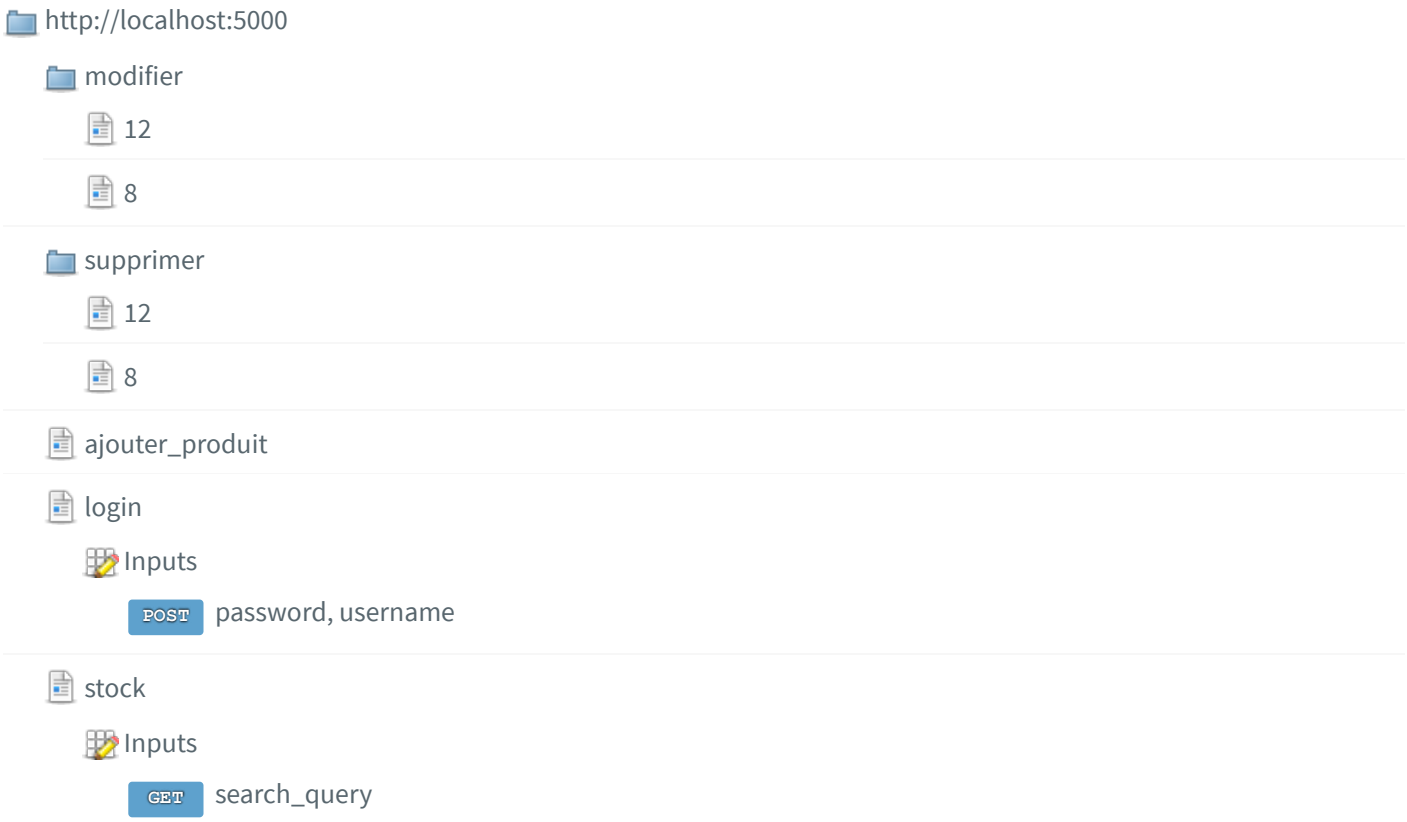
Permissions Policy (W3C)
https://www.w3.org/TR/permissions-policy-1/

A10 Insufficient Logging and Monitoring

Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systesm, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rathre than inernal processes or monitoring.

No alerts in this category

Coverage


http://localhost:5000
modifier
12
8
supprimer
12
8
ajouter_produit
login
Inputs
POST password, username
stock
Inputs
GET search_query