VILNIUS GEDIMINAS TECHNICAL UNIVERSITY

FACULTY OF FUNDAMENTAL SCIENCES

DEPARTMENT OF INFORMATION SYSTEMS

TWO FACTOR AUTHENTICATION

Information Technology Security Methods

Prepared by: Simonas Riška, [redacted]

Checked by: lect. [redacted]

VILNIUS 2024

CONTENT

TWO-FACTOR AUTHENTICATION (2FA) 3

ADVANTAGES AND DISADVANTAGES OF TOTP 3

COMMON TWO-FACTOR AUTHENTICATION VULNERABILITIES 4

TECH STACK FOR FRONT-END AND BACK-END 6

INSTRUCTION 7

TESTING 12

- Functional 2FA testing 12
- Security Testing for Login and 2FA 16
- Functional backup code testing 21

TWO-FACTOR AUTHENTICATION (2FA)

Two-factor authentication (in short – 2FA), also sometimes called as two-step verification or dual-factor authentication, is a security process where users when trying to log in provide two different authentication factors to verify their identity.

When 2FA is implemented, it can give better protection not only to user's credentials but also it can protect resources the user can access so only user who is authorized to access the given resource could have access. This type of authentication is usually used as a part of the effort to prevent data breaches and the potential loss of personal data.

2FA, compared to authentication methods that depend on single-factor authentication, excels in level of security. Using single-factor authentication methods user typically provides a password or passcode – only one authenticating factor, however 2FA methods rely on password as the first factor and a second factor that is different from the first – usually a security token or a biometric factor such as a fingerprint or facial scan. In this work a form of security token known as time-based one-time password (TOTP) was selected as a second factor.

ADVANTAGES AND DISADVANTAGES OF TOTP

Why Use TOTP?

- Stronger Security: TOTP creates unique codes that refresh every 30 seconds, making it tough for hackers to reuse them.
- Works Offline: Unlike SMS or email codes, TOTP doesn't need an internet connection, which adds convenience and security.
- Better Protection Against Phishing and SIM Swapping: TOTP provides better protection than the SMS code method (which can be easier if someone takes your phone number or tricks you into sharing your code with a fake website).
- Easy To Implement and Use: TOTP is widely supported by many applications, so it's easy to use with multiple services without special setup.
- Low-Cost Solution: There's no extra hardware needed, which makes it affordable and simple to deploy.

Why Not Use TOTP?

- User Experience: Typing in codes within a short time frame can be frustrating, especially if you're in a hurry or distracted.
- Phishing Risk: It's still possible to be tricked by placing code on a fake site, where attackers can immediately exploit it.
- Device Dependency: If you lose your phone, recovering access can be a hassle, especially if you don't have backup codes.
- No Context Awareness: TOTP can't adapt based on location or device integrity, which would add an extra security layer.
- Compliance Limitations: In high-security fields, TOTP may not meet certain phishing-resistant standards.

In short, TOTP is a solid choice for many situations but may not be the best fit if top-notch security or a seamless user experience is a must.

COMMON TWO-FACTOR AUTHENTICATION VULNERABILITIES

There are a few common two-factor authentication vulnerabilities:

- **Phishing attacks –** these attacks are designed in such a way that users are tricked into revealing their 2FA codes to malicious actors. Phishing attack can be orchestrated via spoofed emails, fake websites, voice calls to impersonate legitimate entities and persuade users to click on malicious links which ask for the user 2FA codes. To prevent it, the user should always check the sender, URL and content of communication especially when it asks 2FA codes and use security measures such as trusted browser, antivirus software, avoid opening attachments, downloading files from unknown sources.
- **SIM Swapping –** this attack works as an identity theft when user's phone number is transferred to a new SIM card controlled by attacker who can intercept 2FA codes sent via SMS messages or phone calls to the user's phone number and use obtained code to access the user's resources. This attack can be done by exploiting the weaknesses of mobile network operators via social engineering, poor authentication or insider threats. SMS swapping can be prevented by not using SMS or phone call as 2FA method and use more secure alternatives, like TOTP.

- **Man-in-the-Middle Attacks –** these attacks execute by exploiting the vulnerabilities of the network, device or app that the user is using to access the resources and by capturing the 2FA code to use it. MITM attack can be prevented using a secure and encrypted connection when accessing online services, such as HTTPS, VPN or SSL and also by verifying the identity and the certificate of the server, also by avoiding using public or untrusted networks or devices.
- **Malware Infections –** it can bypass 2FA by capturing the user's keystrokes, taking screenshots, retrieving clipboard data and sending them to a remote server controlled by an attacker who can obtain 2FA codes and use them to access their accounts. Malware can be delivered to user's device through various channels, like phishing emails, infected websites, removable media. Malware infections can be prevented by using reputable and updated antivirus software, by scanning devices regularly, by avoiding clicking on suspicious links, opening unknown attachments or inserting untrusted media into devices.
- **Social Engineering –** this technique can bypass 2FA by persuading users to share their 2FA codes with an attacker, who pretends to be a trusted person in a fake scenario, like emergency, reward or threat that would pressure users to act quickly and irrationally. Social engineering prevention could be verifying the identity and intention of a contact, especially when sensitive data, such as 2FA codes are involved.

TECH STACK FOR FRONT-END AND BACK-END

Front-End

1. **Framework Used:** Angular 18.2
2. **UI Components:** login, register, setup2fa, login2fa, dashboard
3. **Authentication/Authorization:** JWT (JSON Web Token) for handling authentication and authorization, session management by storing JWT tokens in secure storage (cookies with HttpOnly and Secure flags) for authenticated sessions.
4. **2FA Integration:** Using time-based one-time passwords (TOTP) via Microsoft Authenticator entering to UI to use API.
5. **Build Tools:** Angular CLI for development and building.

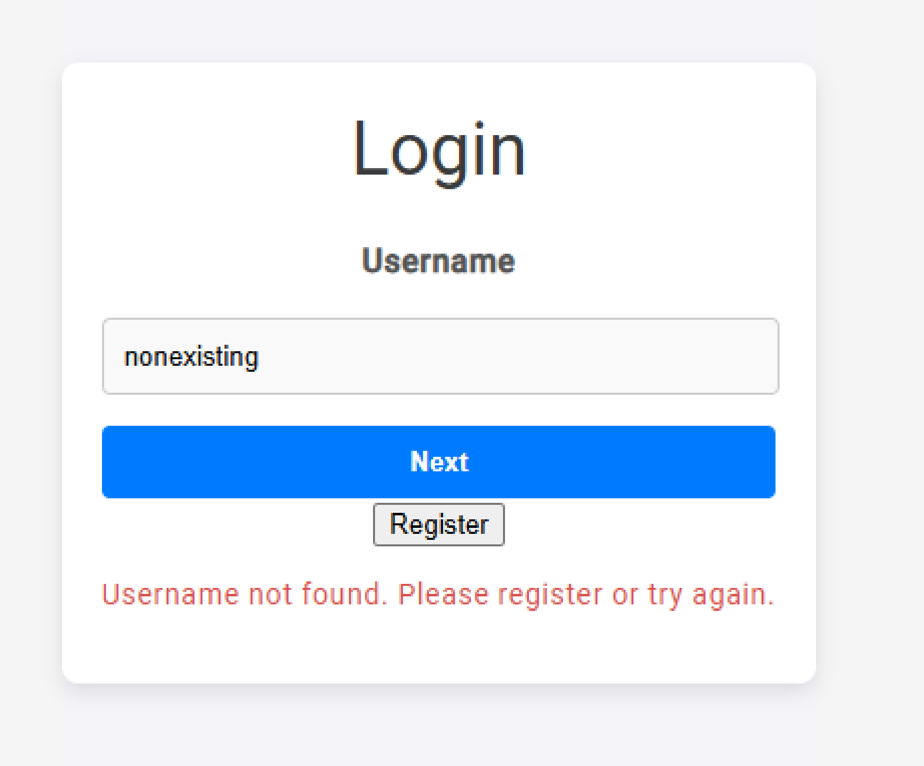
Back-End

1. **Programming Language:** C# (ASP.NET Core framework)
2. **Database:** MSSQL
3. **Authentication & Authorization:** JWT Bearer with Claims
4. **2FA:** ASP.NET Core Identity (Microsoft Authenticator app)
5. **APIs:** ASP.NET Core Web Api + Swagger
6. **Security:** Weather controller secured: accessible only with after successful 2fa login

INSTRUCTION

First Login and 2FA Setup Instructions

1. User Registration: Ensure you are registered in the system.
2. Login Process:
   - Enter Username: Enter your username and click "Next."
     - If the username doesn't exist, you cannot continue. Only valid usernames will be able to access the password input screen.

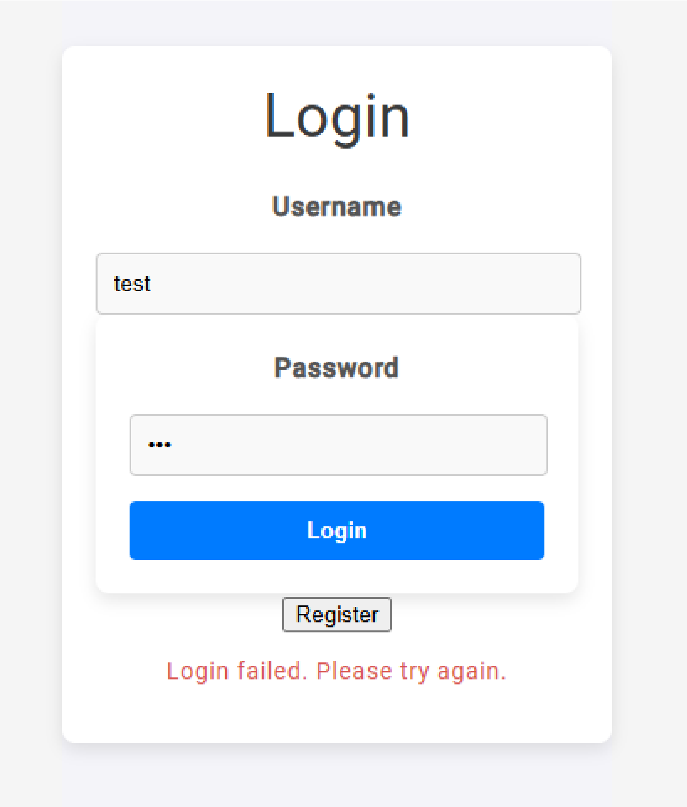


nonexisting

Username not found. Please register or try again.

1. Pic. Username not found

   - Password Attempt:
     - If the username is right but the password is wrong, the account will be locked after 5 failed login tries.



Login

Username

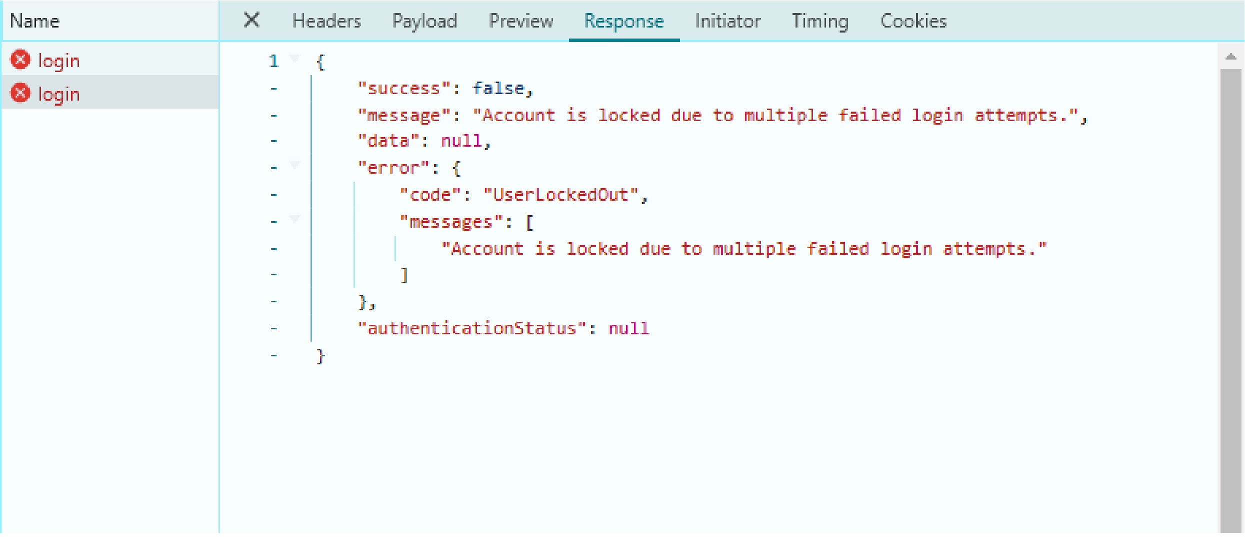
test

Password

•••

Login failed. Please try again.

2.Pic. Response after user enters incorrect password



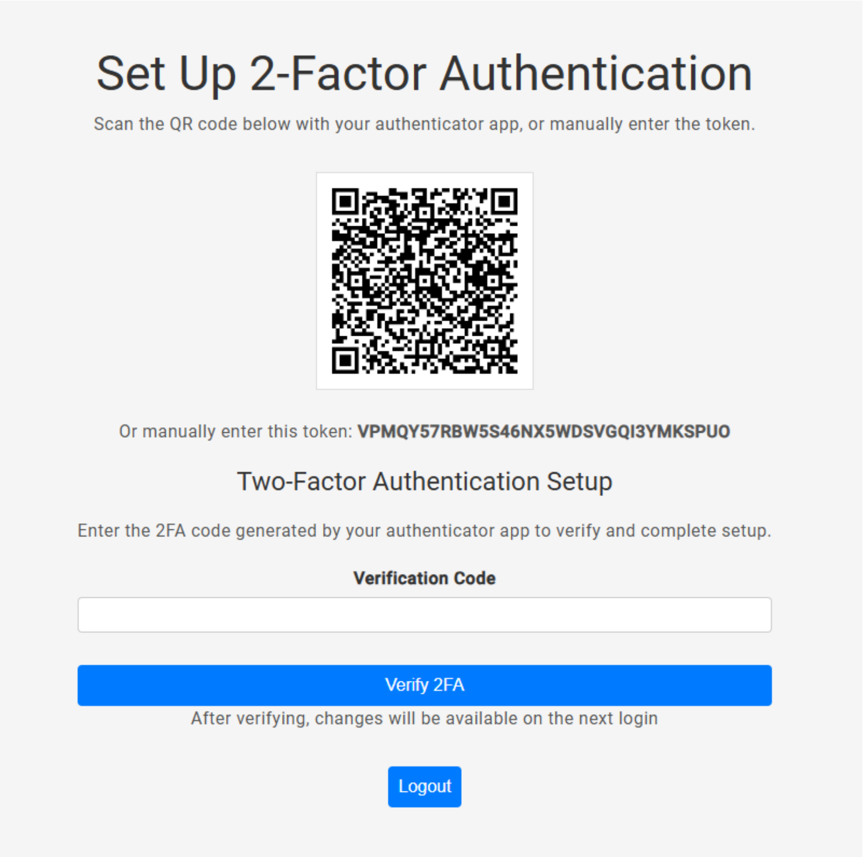
```
{
    "success": false,
    "message": "Account is locked due to multiple failed login attempts.",
    "data": null,
    "error": {
        "code": "UserLockedOut",
        "messages": [
            "Account is locked due to multiple failed login attempts."
        ]
    },
    "authenticationStatus": null
}
```

3. Pic. Response after user enters incorrect password 5 or more times

3. Two-Factor Authentication (2FA) Setup (for newly registered users):

   - After logging in with valid credentials, you must set up 2FA.

- You have two options to set up 2FA using the Microsoft Authenticator app:
  - Scan QR Code: Use the QR Code scanner in the Microsoft Authenticator app to scan the given QR code
  - Enter Token Manually: If you prefer, manually enter the token shown in the system.



Set Up 2-Factor Authentication

Scan the QR code below with your authenticator app, or manually enter the token.

Or manually enter this token: **VPMQY57RBW5S46NX5WDSVGQI3YMKSPUO**

Two-Factor Authentication Setup

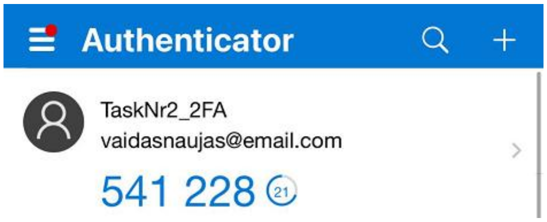
Enter the 2FA code generated by your authenticator app to verify and complete setup.

Verification Code

After verifying, changes will be available on the next login

4. Pic. Set Up 2-Factor Authentication

- After being added to the authenticator app, a temporary code will show up that only lasts for 30 seconds.



TaskNr2_2FA
vaidasnaujas@email.com
541 228

5. Pic. Generated code

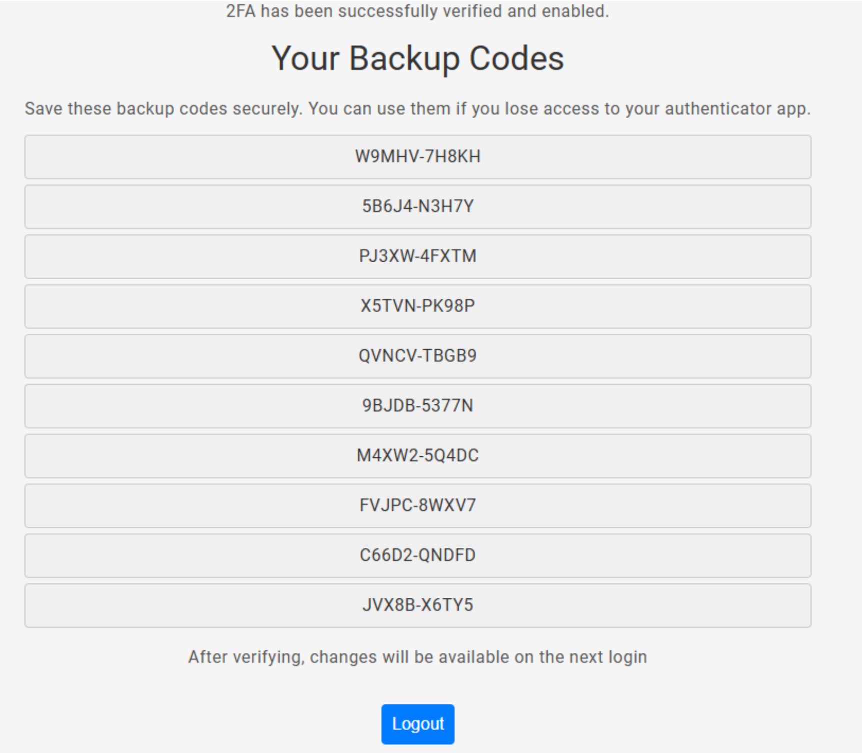
6. Enter Temporary Code:

    - Enter the current code generated by the authenticator app and then click on the "Verify 2FA button" to continue.

7. Backup Codes:

    - After entering a valid code, backup codes will be generated for emergency access.

      These are available for use in case you are unable to access the authenticator app.



2FA has been successfully verified and enabled.

Your Backup Codes

Save these backup codes securely. You can use them if you lose access to your authenticator app.

W9MHV-7H8KH

5B6J4-N3H7Y

PJ3XW-4FXTM

X5TVN-PK98P

QVNCV-TBGB9

9BJDB-5377N

M4XW2-5Q4DC

FVJPC-8WXV7

C66D2-QNDFD

JVX8B-X6TY5

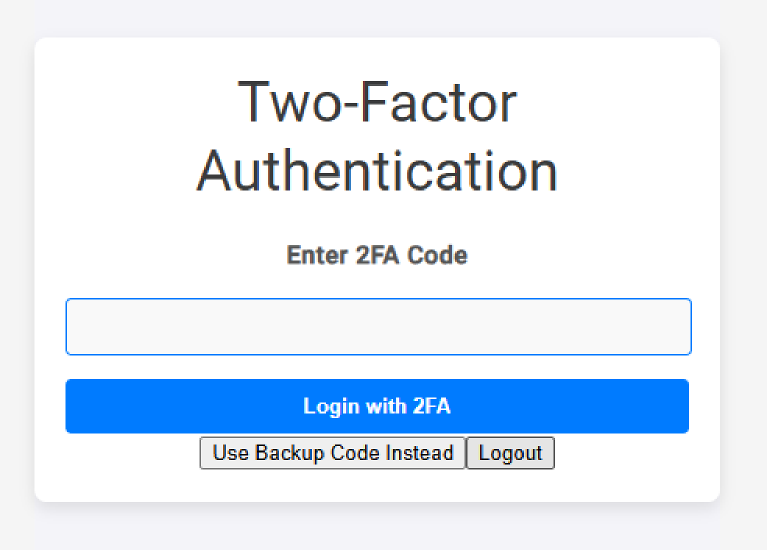
After verifying, changes will be available on the next login

6. Pic. Backup codes

Login after 2FA is set

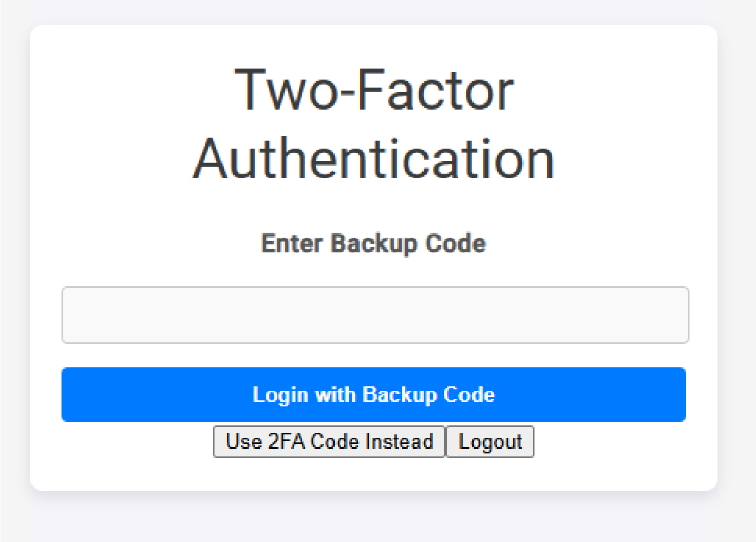
1. Enter Username and Password: Log in by entering your valid username and password.
2. Enter 2FA Code or Backup Code:

    - Enter the code that is generated and is visible in the Microsoft Authenticator app.



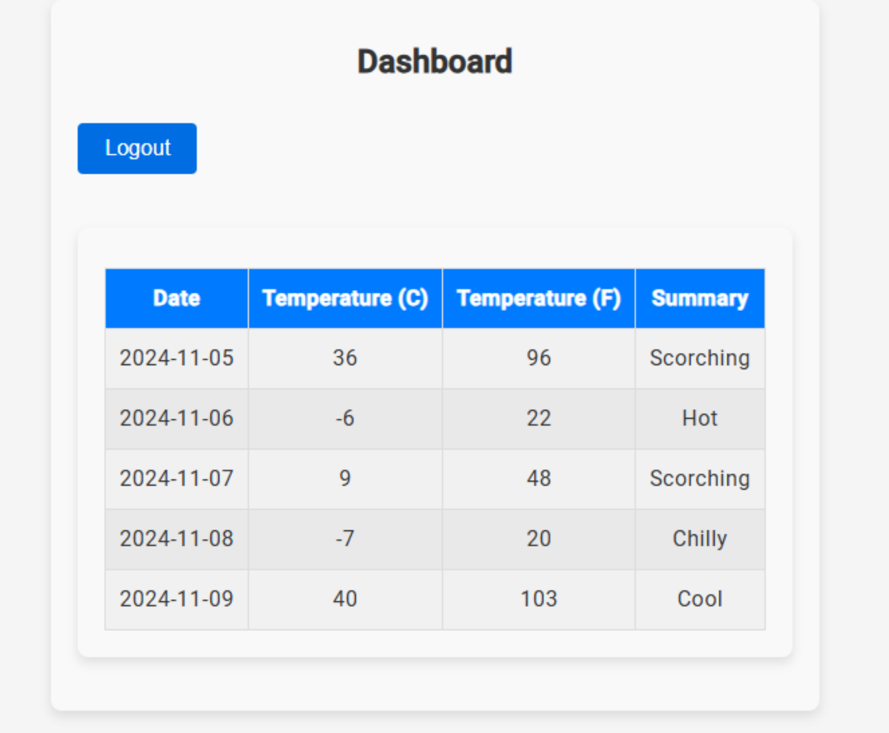
7. Pic. 2FA input

- If user doesn’t have access to the app, backup code can be used instead. Be aware that each backup code can only be used once.



8. Pic. Backup code input

3. Access Granted: After entering a valid 2FA code or backup code, you will receive access to the application.



Date	Temperature (C)	Temperature (F)	Summary
2024-11-05	36	96	Scorching
2024-11-06	-6	22	Hot
2024-11-07	9	48	Scorching
2024-11-08	-7	20	Chilly
2024-11-09	40	103	Cool

9. Pic. Dashboard after successful login

TESTING

Functional 2FA testing

Test Cases for login functionality

Test Case ID	TC-2FA-F-001
Test Title	Setup up 2FA
Precondition	User is already registered
Steps	1. Using valid credentials fill in "Username" and "Password" inputs 2. Click "Log in" button 3. In "Set up 2FA" window, scan QR code, or enter token manually in auth app 4. Fill in "Verification code" input and click "Verify 2FA"
Expected result	2FA has been successfully verified and enabled.
Actual result	2FA has been successfully verified and enabled.
Status	**Pass**

Test Case ID	TC-2FA-F-002
Test Title	Login after 2FA is set
Steps	1. Using valid credentials fill in “Username” and “Password” inputs 2. Click “Log in” button and wait for 2FA prompt 3. Enter valid 2FA code and proceed with login
Expected result	The user enters the correct 2FA code and logs in successfully.
Actual result	After user enters the correct 2FA code logs in successful.
Status	**Pass**

Test Case ID	TC-2FA-F-003
Test Title	Valid username and incorrect password
Steps	1. Fill in “Username” input with valid username 2. Fill in “Password” input with invalid password 3. Click “Log in” button
Expected result	An error about incorrect credentials appears, and the user is unable to move on to the 2FA step.
Actual result	An error about incorrect credentials appears, “Login failed. Please try again.”
Status	Pass

Test Case ID	TC-2FA-F-004
Test Title	Expired 2FA Code
Steps	1. Using valid credentials fill in “Username” and “Password” inputs 2. Click “Log in” button and wait for 2FA prompt 3. Wait for token to be expired, then enter it
Expected result	After expired token is entered and user clicks “Login with 2FA” button, message that incorrect token is entered should be displayed
Actual result	Expired 2FA code can be used, weather app is opened
Status	**Fail**

Test Case ID	TC-2FA-F-005

Test Title	Invalid 2FA Code
Steps	1. Using valid credentials fill in “Username” and “Password” inputs 2. Click “Log in” button and wait for 2FA prompt 3. Enter invalid 2FA code
Expected result	The login is denied and an error message appears.
Actual result	After user enters invalid 2FA code, message “Failed to log in with 2FA code” is returned
Status	**Pass**

Test Case ID	TC-2FA-F-006
Test Title	Used 2FA Code
Steps	1. Using valid credentials fill in “Username” and “Password” inputs 2. Click “Log in” button and wait for 2FA prompt 3. Reuse 2FA code in other session that was used and login was successful
Expected result	The login is denied and an error message appears, used 2FA token becomes invalidated once used
Actual result	2FA code can be reused for other login session
Status	**Fail**

Test Case ID	TC-2FA-F-007
Test Title	Lockout
Steps	1. Fill in “Username” input with valid username 2. Fill in “Password” input with invalid password 3. Click “Log in” button 4. Enter invalid “2FA” code 5. Repeat these steps at least 3 times
Expected result	If a user enters an incorrect 2FA code three times or more in a row, the system will temporarily lock them out. An account is locked because of several unsuccessful 2FA login attempts. The message "Please attempt again at a later time" should appear. The user must not try logging in again until the lockout period is over.

Actual result	The system permits multiple attempts with wrong 2FA codes without blocking the user, even following 3 or more unsuccessful tries. There is no lockout message shown, so the user can keep trying to log in with incorrect codes.
Status	**Fail**

Test Case ID	TC-2FA-F-008
Test Title	Login with Empty Username
Steps	1. Open the login window. 2. Leave the "Username" input empty. 3. Click on the "Next" button.
Expected result	A message should be displayed stating that the username field must not be left blank The user should remain on the same login page.
Actual result	Request to back-end is sent, and response "The username field is required" is returned, missing validation from front-end side.
Status	**Fail**

Test Case ID	TC-2FA-F-009
Test Title	Login with non-existing username
Steps	1. Open the login window. 2. Fill in the "Username" input with random value 3. Click "Next" button.
Expected result	Error message is displayed, indicating that username not exists, user remains on the same page
Actual result	Error message “Error checking username. Please try again.” Is displayed, users stay on same page
Status	**Pass**

Test Case ID	TC-2FA-F-010
Test Title	Login existing username
Steps	1. Open the login window. 2. Fill in the "Username" (registered username) input 3. Click "Next" button.
Expected result	After the user enters a valid username, the system should:

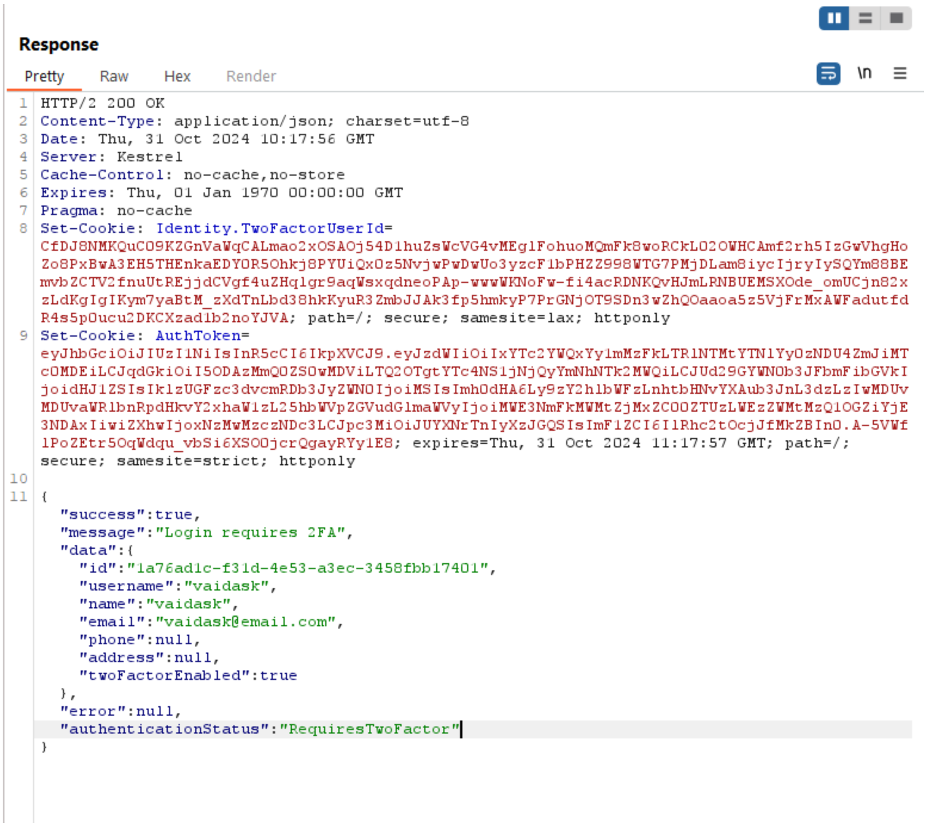
	- Return a status code of 200 OK. - Display the password input field, allowing the user to proceed with the login.
Actual result	User is allowed to proceed with login, but A response with status code 400 Bad Request is returned with the message "Username in use".
Status	**Fail**

Security Testing for Login and 2FA

Test Case ID	TC-2FA-S-001
Test Title	SQL Injection in Username Field
Steps	1. Fill in the "Username" input field with: ' OR '1'='1' --. 2. Click the "Next" button.
Expected result	Status Code: 400 Bad Request The system should not proceed to the next step. The status code should be 400 Bad Request or another suitable error code indicating that invalid input was detected.
Actual result	- Status Code: 200 OK - Response: "Username is available." - Behavior: The system did not proceed to the next step but returned an incorrect response indicating the username is available.
Status	**Fail**

Test Case ID	TC-2FA-S-002
Test Title	/Auth/login endpoint response for a user who has 2FA.
Steps	1. Using valid credentials fill in “Username” and “Password” inputs 2. Click “Log in”

Result (TC-2FA-S-002):



```
HTTP/2 200 OK
Content-Type: application/json; charset=utf-8
Date: Thu, 31 Oct 2024 10:17:56 GMT
Server: Kestrel
Cache-Control: no-cache,no-store
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Pragma: no-cache
Set-Cookie: Identity.TwoFactorUserId=
CfDJ8NMKQuCO9KZGnVaWqCALmao2xOSAOj54D1huZsWcVG4vMEglFohuoMQmFk8woRCkLO2OWHCAmf2rh5IzGwVhgHo
Zo8PxBwA3EH5THEnkaEDYOR5Ohkj8PYUiQxOz5NvjwPwDwUo3yzcF1bPHZZ998WTG7PMjDLam8iycIjryIySQYm88BE
mvbZCTV2fnuUtREjjdCVgf4uZHqlgr9aqWsxqdneoPAp-wwwWKNoFw-fi4acRDNKQvHJmLRNBUEMSXOde_omUCjn82x
zLdKgIgIKym7yaBtM_zXdTnLbd38hkKyuR3ZmbJJAk3fp5hmkyP7PrGNjOT9SDn3wZhQOaaoa5z5VjFrMxAWFadutfd
R4s5pOucu2DKCXzad1b2noYJVA; path=/; secure; samesite=lax; httponly
Set-Cookie: AuthToken=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxYTc2YWQxYy1mMzFkLTR1NTMtYTNlYyOzNDU4ZmJiMT
cOMDEiLCJqdGkiOiI5ODAzMmQOZSOwMDViLTQ2OTgtYTc4NS1jNjQyYmNhNTk2MWQiLCJUd29GYWNOb3JFbmFibGVkI
joidHJ1ZSIsIklzUGFzc3dvcmRDb3JyZWNOIjoiMSIsImhOdHA6Ly9zY2hlbWFzLnhtbHNvYXAub3JnL3dzLzIwMDUv
MDUvaWRlbnRpdHkvY2xhaW1zL25hbWVpZGVudGlmaWVyIjoiMWE3NmFkMWMtZjMxZC00ZTUzLWEzZWMtMzQ1OGZiYjE
3NDAxIiwiZXhwIjoxNzMwMzczNDc3LCJpc3MiOiJUYXNrTnIyXzJGQSIsImF1ZCI6IlRhc2tOcjJfMkZBInO.A-5VWf
1PoZEtr5OqWdqu_vbSi6XSOOjcrQgayRYy1E8; expires=Thu, 31 Oct 2024 11:17:57 GMT; path=/;
secure; samesite=strict; httponly

{
  "success":true,
  "message":"Login requires 2FA",
  "data":{
    "id":"1a76ad1c-f31d-4e53-a3ec-3458fbb17401",
    "username":"vaidask",
    "name":"vaidask",
    "email":"vaidask@email.com",
    "phone":null,
    "address":null,
    "twoFactorEnabled":true
  },
  "error":null,
  "authenticationStatus":"RequiresTwoFactor"
}
```

10. Pic. Response after login only with credentials

Exposure to Sensitive Data - Response:

The Set-Cookie header for Identity.TwoFactorUserId and AuthToken exposes information, including a JWT token and possibly confidential data. We can implement token masking or minimizing unnecessary data exposure.

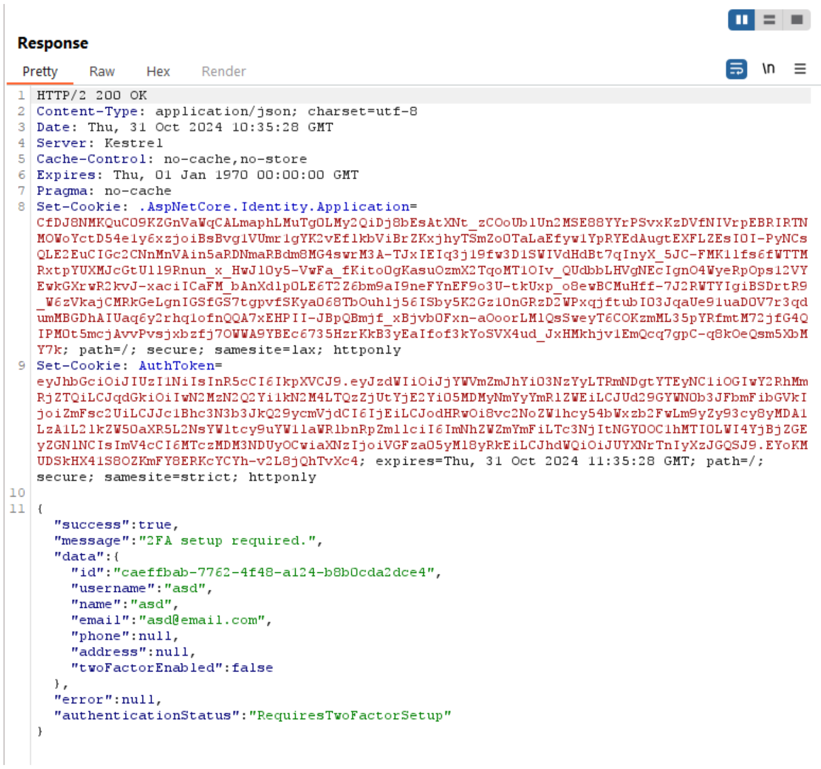
Potential Exposure of Information:

The reply indicates {"success":true,"message":"Login requires 2FA"}, revealing to the attacker that 2FA is turned on for this account. We could use a general error message that would minimize leaking of information.

Detailed user data in the response: The response body contains user-specific information such as id, username, email, etc. Revealing all this data during a login attempt could lead to increased vulnerability. Restrict the data given back, particularly sensitive or identifiable information, until authentication is completely confirmed.

Test Case ID	TC-2FA-S-03
Test Title	/Auth/login endpoint response for a user who hasn’t set up 2FA.
Steps	1. Using valid credentials fill in “Username” and “Password” inputs 2. Click “Log in”

Result (TC-2FA-S-003):



```
HTTP/2 200 OK
Content-Type: application/json; charset=utf-8
Date: Thu, 31 Oct 2024 10:35:28 GMT
Server: Kestrel
Cache-Control: no-cache,no-store
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Pragma: no-cache
Set-Cookie: .AspNetCore.Identity.Application=
CfDJ8NMKQuCO9KZGnVaWqCALmaphLMuTgOLMy2QiDj8bEsAtXNt_zCOoUblUn2MSE88YYrPSvxKzDVfNIVrpEBRIRTN
MOWoYctD54ely6xzjoiBsBvglVUmrlgYK2vEflkbViBrZKxjhyTSmZoOTaLaEfywlYpRYEdAugtEXFLZEsIOI-PyNCs
QLE2EuCIGc2CNnMnVAin5aRDNmaRBdm8MG4swrM3A-TJxIEIq3j19fw3D1SWIVdHdBt7qInyX_5JC-FMKllfs6fWTTM
RxtpYUXMJcGtUll9Rnun_x_HwJlOy5-VwFa_fKitoOgKasuOzmX2TqoMT1OIv_QUdbbLHVgNEcIgnO4WyeRpOps12VY
EwkGXrwR2kvJ-xaciICaFM_bAnXdlpOLE6T2Z6bm9aI9neFYnEF9o3U-tkUxp_o8ewBCMuHff-7J2RWTYIgiBSDrtR9
_W6zVkajCMRkGeLgnIGSfGS7tgpvfSKya068TbOuhlj56ISby5K2Gz1OnGRzD2WPxqjftubIO3JqaUe91uaDOV7r3qd
umMBGDhAIUaq6y2rhqlofnQQA7xEHPII-JBpQBmjf_xBjvbOFxn-aOoorLMlQsSweyT6COKzmML35pYRfmtM72jfG4Q
IPMOt5mcjAvvPvsjxbzfj7OWWA9YBEc6735HzrKkB3yEaIfof3kYoSVX4ud_JxHMkhjv1EmQcq7gpC-q8kOeQsm5XbM
Y7k; path=/; secure; samesite=lax; httponly
Set-Cookie: AuthToken=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJjYWVmZmJhYiO3NzYyLTRmNDgtYTEyNC1iOGIwY2RhMm
RjZTQiLCJqdGkiOiIwN2MzN2Q2Yi1kN2M4LTQzZjUtYjE2YiO5MDMyNmYyYmRlZWEiLCJUd29GYWNOb3JFbmFibGVkI
joiZmFsc2UiLCJJc1Bhc3N3b3JkQ29ycmVjdCI6IjEiLCJodHRwOi8vc2NoZW1hcy54bWxzb2FwLm9yZy93cy8yMDA1
LzA1L21kZW5OaXR5L2NsYWltcy9uYW1laWRlbnRpZmllciI6ImNhZWZmYmFiLTc3NjItNGY0OC1hMTI0LWI4YjBjZGE
yZGNlNCIsImV4cCI6MTczMDM3NDUyOCwiaXNzIjoiVGFza05yMl8yRkEiLCJhdWQiOiJUYXNrTnIyXzJGQSJ9.EYoKM
UDSkHX41S8OZKmFY8ERKcYCYh-v2L8jQhTvXc4; expires=Thu, 31 Oct 2024 11:35:28 GMT; path=/;
secure; samesite=strict; httponly

{
  "success":true,
  "message":"2FA setup required.",
  "data":{
    "id":"caeffbab-7762-4f48-a124-b8b0cda2dce4",
    "username":"asd",
    "name":"asd",
    "email":"asd@email.com",
    "phone":null,
    "address":null,
    "twoFactorEnabled":false
  },
  "error":null,
  "authenticationStatus":"RequiresTwoFactorSetup"
}
```

11. Pic. Login endpoint response for a user who hasn’t set up 2FA.

Detailed 2FA Status Disclosure:

The authenticationStatus field with "RequiresTwoFactorSetup" is an information leak. It reveals that the user has not set up 2FA, which an attacker could potentially exploit. A more generic response (like "Additional authentication required") would avoid leaking setup status.

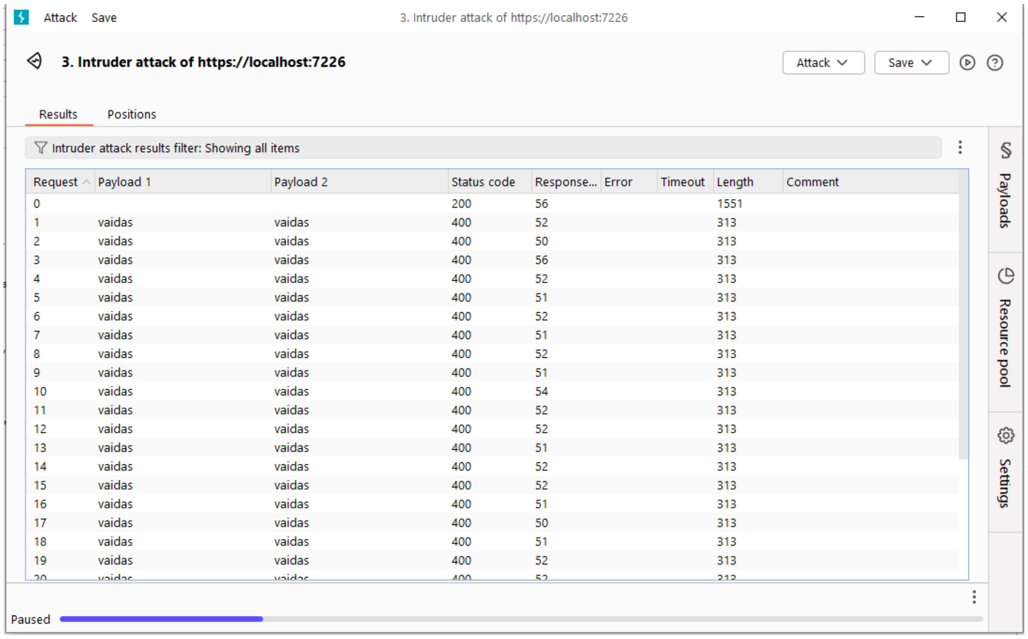
Unnecessary Cookie for Identity.TwoFactorUserId:

Since 2FA is not set up for this user, the server should avoid setting the Identity.TwoFactorUserId cookie altogether, or at least set it to a non-sensitive placeholder value. This reduces the exposure of unnecessary cookies.

Test Case ID	TC-2FA-S-004

Test Title	/Auth/login brute force attack
Steps	1. Fill out the login form with your test credentials, then click "Submit." The request will be intercepted by Burp Suite. Navigate to the "Intruder" tab with this intercepted request. 2. Select "Cluster bomb attack" 3. Define the payload positions by marking the username and password fields. 4. Load a list of usernames (for this case valid username is used) and passwords from a wordlist (Sec List is used) 5. Click "Start attack" button
Expected result	After a certain number of failed attempts, the account should lock or temporarily restrict login attempts.
Actual result:	Brute force attack can be executed
Status:	**Fail**

Result (TC-2FA-S-004):


Attack
Save
3. Intruder attack of https://localhost:7226
3. Intruder attack of https://localhost:7226
Attack
Save
Results
Positions
Payloads
Resource pool
Settings

Intruder attack results filter: Showing all items

Request	Payload 1	Payload 2	Status code	Response...	Error	Timeout	Length	Comment
0			200	56			1551	
1	vaidas	vaidas	400	52			313	
2	vaidas	vaidas	400	50			313	
3	vaidas	vaidas	400	56			313	
4	vaidas	vaidas	400	52			313	
5	vaidas	vaidas	400	51			313	
6	vaidas	vaidas	400	52			313	
7	vaidas	vaidas	400	51			313	
8	vaidas	vaidas	400	52			313	
9	vaidas	vaidas	400	51			313	
10	vaidas	vaidas	400	54			313	
11	vaidas	vaidas	400	52			313	
12	vaidas	vaidas	400	52			313	
13	vaidas	vaidas	400	51			313	
14	vaidas	vaidas	400	52			313	
15	vaidas	vaidas	400	52			313	
16	vaidas	vaidas	400	51			313	
17	vaidas	vaidas	400	50			313	
18	vaidas	vaidas	400	51			313	
19	vaidas	vaidas	400	52			313	
20	vaidas	vaidas	400	52			313	

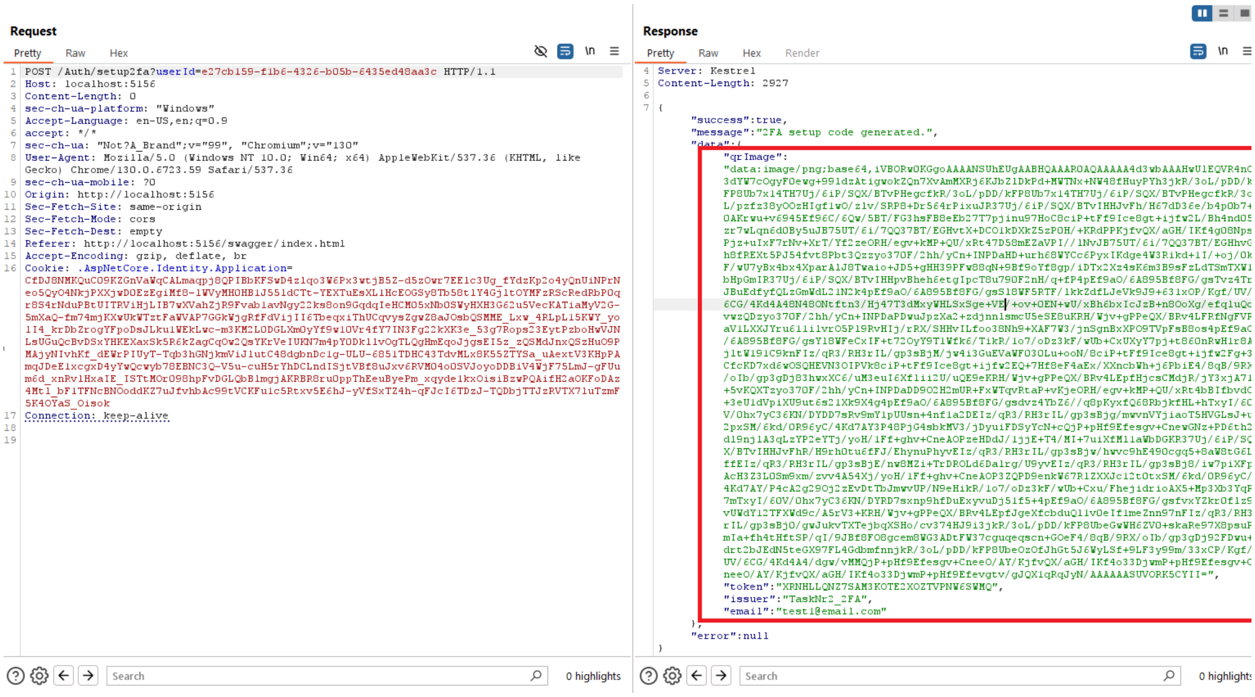
Paused

12. Pic. Login brute force attack

Test Case ID	TC-2FA-S-005
Test Title	Verify Secure Transmission of 2FA Token and QR Image - /Auth/setup2fa

Steps	1. Authenticate as a test user and initiate the 2FA setup process 2. Examine the response payload 3. Check if a secure URL is provided instead of a base64-encoded image string for the QR code.
Expected result	The response payload does not expose the token directly and provides a secure URL to access the QR image, avoiding excessive data transmission.
Actual result	1. Sensitive Data Exposure: The response exposes the token, which should not be transmitted openly. 2. Direct QR Image: QR code provided as a base64 string instead of a secure link, increasing response size and risk. 3. Excess Data: Fields like issuer and email are unnecessary, increasing potential data exposure.
Status	**Fail**

Result (TC-2FA-S-005):



```
POST /Auth/setup2fa?userId=e27cb159-f1b6-4326-b05b-6435ed48aa3c HTTP/1.1
Host: localhost:5156
Content-Length: 0
sec-ch-ua-platform: "Windows"
Accept-Language: en-US,en;q=0.9
accept: */*
sec-ch-ua: "Not?A_Brand";v="99", "Chromium";v="130"
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/130.0.6723.59 Safari/537.36
sec-ch-ua-mobile: ?0
Origin: http://localhost:5156
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:5156/swagger/index.html
Accept-Encoding: gzip, deflate, br
Cookie: .AspNetCore.Identity.Application=
CfDJ8NMKQuCO9KZGnVaWqCALmaqpj8QPIBbKFSwD4zlqo3W6Px3wtjB5Z-d5zOwr7EElc3Ug_fYdzKp2o4yQnUiNPrN
eo5QyO4NkjPXXjwDOEzEgiMf8-lWVyMHOHB1J55ldCTt-YEXTuEsXL1HcEOGSy8Tb58tlY4GjltOYMFzRScRedRbPOq
r8S4rNduPBtUITRViHjLIB7wXVahZjR9Fvab1avNgy22ks8on9GqdqIeHCMO5xNbOSWyHXH3G62u5VecKATiaMyV2G-
5mXaQ-fm74mjKXwUkWTztFaWVAP7GGkWjgRfFdVijII6TbeqxiThUCqvysZgwZ8aJOsbQSMME_Lxw_4RLpL15KWY_yo
lI4_krDbZrogYFpoDsJLku1WEkLwc-m3KM2LODGLXmOyYf9w1OVr4fY7IN3Fg22kXK3e_53g7Rops23EytPzboHwVJN
LsUGuQcBvDSxYHKEXaxSk5R6kZagCqOw2QsYKrVeIUKN7m4pYODkllvOgTLQgHmEqoJjgsEI5z_zQSMdJnxQSzHuO9P
MAjyNIvhKf_dEWrPIUyT-Tqb3hGNjkmViJlutC48dgbnDc1g-ULU-6851TDHC43TdvMLx8K55ZTYSa_uAextV3KHpPA
mqJDeE1xcgxD4yYwQcwyb78EBNC3Q-V5u-cuH5rYhDCLndISjtVBf8uJxv6RVMO4oOSVJoyoDDBiV4WjF75LmJ-gFUu
m6d_xnRvlHxaIE_ISTtMOrO98hpFvDGLQbBlmgjAKRBR8ruOppThEeuByePm_xqyde1kxOisiBzwPQAifH2aOKFoDAz
4Mtl_bF1TFNcBNOoddKZ7uJfvhbAc99tVCKFu1c5Rtxv5E6hJ-yVfSxTZ4h-qFJcI6TDzJ-TQDbjTTJzRVTX7luTzmF
5K4OYaS_Oisok
Connection: keep-alive


```

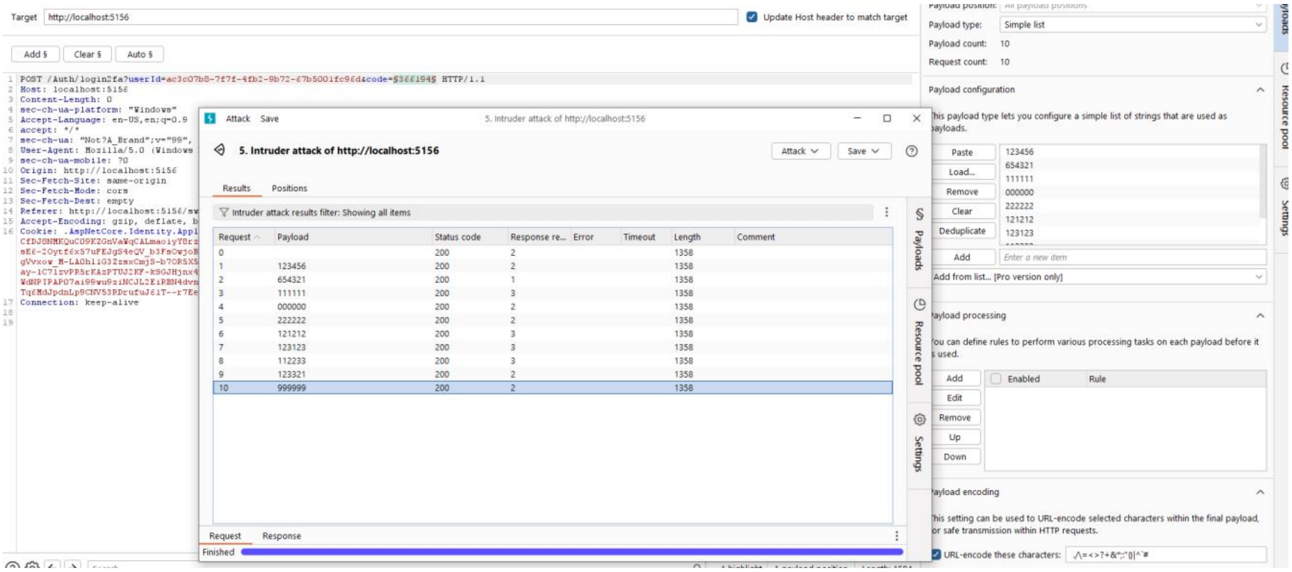
```
Server: Kestrel
Content-Length: 2927

{
    "success":true,
    "message":"2FA setup code generated.",
    "data":{
        "qrImage":
        "data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAABHQAAAR0AQAAAAA4d3wbAAAHwUlEQVR4nC
        3dYW7cOgyF0ewg+991dzAtigwokZQn7XvAmMXRj6KJbZlDkPd+MWTNx+NW48fHuyPYh3jkR/3oL/pDD/k
        FP8Ub7x14TH7Uj/6iP/SQX/BTvPHegcfkR/3oL/pDD/kFP8Ub7x14TH7Uj/6iP/SQX/BTvPHegcfkR/3c
        L/pzfz38yOOzHIgf1wO/zlv/SRP8+Dr564rPixuJR37Uj/6iP/SQX/BTvIHHJvFh/H67dD36e/b4pOb7+
        OAKrwu+v6945Ef96C/6Qw/5BT/FG3hsFB8eEb27T7pjinu97HoC8ciP+tFf9Ice8gt+ijfw2L/Bh4ndO5
        zr7wLqn6dOBy5uJB75UT/6i/7QQ37BT/EGHvtX+DCO1kDXkZ5zPOH/+KRdPPKjfvQX/aGH/IKf4gO8Nps
        Pjz+uIxF7rNv+XrT/Yf2zeORH/egv+kMP+QU/xRt47D58mEZaVPI//1NvJB75UT/6i/7QQ37BT/EGHhvG
        h8fREXt5PJ54fvt8Pbt3Qzzyo37OF/2hh/yCn+INPDaHD+urh68WYCc6PyxIKdge4W3Rikd+1I/+oj/Ok
        F/wU7yBx4bx4XparAlJ8Twaio+JD5+gHH39PFw88qN+9Bf9oYf8gp/iDTx2Xz4sK6m3B9sFzLdTSmTXW1
        bHpGmIR37Uj/6iP/SQX/BTvIHHpvBheh6etgIpcT8u79OF2nH/q+fP4pEf9aO/6A895Bf8FG/gsTvz4Tr
        JBuEdfyfQLzGmWdL21N2k4pEf9aO/6A895Bf8FG/gsS18WF5RTF/1kkZdfLJeVk9J9+631xOP/Kgf/UV/
        6CG/4Kd4A48N48ONtftn3/Hj47T3dMxyWHLSxSge+VE/+ov+OEN+wU/xBh6bxIcJzB+n8OoXg/efqluQc
        vwzQDzyo37OF/2hh/yCn+INPDaPDwuJpzXa2+zdjnnlsmcU5eSE8uKRH/Wjv+gPPeQX/BRv4LFRfNgFVF
        aV1LXXJYru611ilvrO5Pl9RvHIj/rRX/SHHvILfoo38Nh9+XAF7W3/jnSgnBxXPO9TVpFsB8os4pEf9aC
        /6A895Bf8FG/gsYl8WFeCxIF+t72OyY9T1Wfk6/TikR/1o7/oDz3kF/wUb+CxUXyY7pj+t86OnRwH1r8A
        j1tW191C9knFIz/qR3/RH3rIL/gp3sBjM/jw4i3GuEVaWFO3OLu+ooN/8ciP+tFf9Ice8gt+ijfw2Fg+3
        CfcKD7xd6wOSQHEVN3OIPVk8ciP+tFf9Ice8gt+ijfw2EQ+7Hf8eF4aEx/XXncbWh+j6PbiE4/8qB/9RX
        /oIb/gp3gDj83hwxXC6/uM3euI6Xflii2U/uQE9eKRH/Wjv+gPPeQX/BRv4LEpfHjcsCMdjR/jY3xjA71
        +5vKQXTzyo37OF/2hh/yCn+INPDaDD9O2H2mUR+FxWTqvRtaP+vKjeORH/egv+kMP+QU/xRt4bBIfbvdC
        +3eU1dVpiXU9ut6s21Xk9X4g4pEf9aO/6A895Bf8FG/gsdvz4YbZ6//q8pKyxfQ68RbjkfHL+hTxyI/6C
        V/Ohx7yC36KN/DYDD7sRv9mY1pUUsn+4nf1a2DEIz/qR3/RH3rIL/gp3sBjg/mwvnVYjiaoT5HVGLsJ+u
        2pxSM/6kd/OR96yC/4Kd7AY3P48PjG4sbkMV3/jDyuiFDSyYcN+cQjP+pHf9Efesgv+CnewGNz+PD6th2
        dl9njlA3qLzYP2eYTj/yoH/1Ff+ghv+CneAOPzeHDdJ/1jjE+T4/MI+7uiXfMl1aWbDGKR37Uj/6iP/SQ
        X/BTvIHHJvFhR/H9rh0tu6fFJ/EhynuPhyvEIz/qR3/RH3rIL/gp3sBjw/hwvc9hE49Ocgq5+8aW8tG6L
        ffEIz/qR3/RH3rIL/gp3sBjE/nw8MZi+TrDROLd6Dalrg/U9yvEIz/qR3/RH3rIL/gp3sBj8/iw7piXFp
        AcH3Z3LOSm9xm/zvv4A54Xj/yoH/1Ff+ghv+CneAOP3ZQPD9enkW67RlZXXJc12t0txSM/6kd/OR96yC/
        4Kd7AY/P4cA2g29Oj2zEvDtTbJmwvUP/N9eHikR/1o7/oDz3kF/wUb+Cxu/FhejidrioAX5+Mp3Xb3YqP
        7mTxyI/6OV/Ohx7yC36KN/DYRD7sxnp9hfDuExyvuDj51f5+4pEf9aO/6A895Bf8FG/gsfvxYZkr0flz9
        vUWdY12TFXWd9c/A5rV3+KRH/Wjv+gPPeQX/BRv4LEpfJgeXfcbduQl1vOeIf1meZnn97nFIz/qR3/RH3
        rIL/gp3sBjO/gwJukvTXTejbqXSHo/cv374HJ9i3jkR/3oL/pDD/kFP8UbeGwWH6ZVO+skaRe97X8psuP
        mIa+fh4tHftSP/qI/9JBf8FO8gcem8WG3ADtFW37cguqeqscn+GOeF4/8qB/9RX/oIb/gp3gDj92FDwu+
        drt2bJEdN5teGX97FL4GdbmfnnjkR/3oL/pDD/kFP8UbeOzOfJhGt5J6WyLSf+9LF3y99m/33xCP/Kgf/
        UV/6CG/4Kd4A4/dgw/vMMQjP+pHf9Efesgv+CneeO/AY/KjfvQX/aGH/IKf4o33DjwmP+pHf9Efesgv+C
        neeO/AY/KjfvQX/aGH/IKf4o33DjwmP+pHf9Efevgtv/gJQX1qRqJyN/AAAAAASUVORK5CYII=",
        "token":"XRNHLLQNZ7SAM3KOTE2XOZTVPNW6SWMQ",
        "issuer":"TaskNr2_2FA",
        "email":"test1@email.com"
    },
    "error":null
}
```

13. Pic. Setup2fa endpoint response for a user who hasn’t set up 2FA.

Test Case ID	TC-2FA-S-006
Test Title	/Auth/login2fa brute force attack

Steps	1. Fill out the login form with your test credentials, approve 2FA." The request will be intercepted by Burp Suite. Navigate to the "Intruder" tab with this intercepted request. 2. Select "Cluster bomb attack" 3. Define the payload positions by marking code field. 4. Leave same userId 5. Click "Start attack" button
Expected result	After a certain number of failed attempts, the account should lock or temporarily restrict login attempts.
Actual result:	Brute force attack can be executed
Status:	**Fail**

Result (TC-2FA-S-006):



Target: http://localhost:5156

```
POST /Auth/login2fa?userId=ac3c07b8-7f7f-4fb2-9b72-67b5001fc96d&code=§366194§ HTTP/1.1
Host: localhost:5156
Content-Length: 0
sec-ch-ua-platform: "Windows"
Accept-Language: en-US,en;q=0.9
accept: */*
sec-ch-ua: "Not?A_Brand";v="99",
User-Agent: Mozilla/5.0 (Windows
sec-ch-ua-mobile: ?0
Origin: http://localhost:5156
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:5156/sw
Accept-Encoding: gzip, deflate, b
Cookie: .AspNetCore.Identity.Appl
CfDJ8NMKQuC09KZGnVaWqCALmaoiyY8r
sE6-2Oytf6x57uFEJgS4eQV_b3FsOwjoB
gVvxow_M-LA0h11G3ZzsxCmjS-b7OR5X5
ay-1C7lzvPR5rKAzPTUJ2KF-kSGJHjnx4
WdNPIPAP07ai99wu9ziNCJL2EiRBN4dvn
Tq6MdJpdnLp9CNV53RDrufuJ61T--r7Ee
Connection: keep-alive
```

5. Intruder attack of http://localhost:5156

Intruder attack results filter: Showing all items

Request	Payload	Status code	Response re...	Error	Timeout	Length	Comment
0		200	2			1358	
1	123456	200	2			1358	
2	654321	200	1			1358	
3	111111	200	3			1358	
4	000000	200	2			1358	
5	222222	200	2			1358	
6	121212	200	3			1358	
7	123123	200	3			1358	
8	112233	200	3			1358	
9	123321	200	2			1358	
10	999999	200	2			1358	

Finished

Payload position: All payload positions
Payload type: Simple list
Payload count: 10
Request count: 10

Payload configuration

This payload type lets you configure a simple list of strings that are used as payloads.

123456
654321
111111
000000
222222
121212
123123

Payload processing

You can define rules to perform various processing tasks on each payload before it is used.

Payload encoding

This setting can be used to URL-encode selected characters within the final payload, for safe transmission within HTTP requests.

☑ URL-encode these characters: ./\=<>?+&*;:"{}|^`#

14. Pic. Login brute force attack

Functional backup code testing

Test Cases for Backup Authentication Solution (One-Time Backup Codes)

Test Case ID	TC-2FA-F-011
Test Title	Backup Code Generation from App Dashboard
Precondition	User is already registered
Steps	1. Using valid credentials fill in "Username" and "Password" inputs 2. Click "Log in" button

	3. In "Set up 2FA" window, scan QR code, or enter token manually in auth app 4. Fill in "Verification code" input and click "Verify 2FA"
Expected result	The application needs to create a series of backup codes that can only be used once for logging in if the user is unable to use their Authenticator app. The codes need to be shown once and have a secure download feature available.
Actual result	Backup codes are created, but there is no secure download feature **available**
Status	**Pass**

Test Case ID	TC-2FA-F-012
Test Title	Backup Code Validity
Steps	1. Use valid backup code to sign 2. Try to re-use identical code once more.
Expected result	Backup codes can only be used once and shouldn't be reusable after use.
Actual result	Backup codes can be used only once
Status	**Pass**

Test Case ID	TC-2FA-F-013
Test Title	Use backup code as 2FA code
Steps	1. Enter backup code to 2FA Code input
Expected result	Error message is returned, that code is invalid
Actual result	Error message "Failed to log in with 2FA code." Is returned
Status	**Pass**

Functional forgot password testing

Test Cases for "Forgot Password" with 2FA (Microsoft Authenticator)

Test Case ID	TC-2FA-F-013
Test Title	Forgot Password - Basic Flow with 2FA
Steps	1. Click the "Forgot Password" button on the login page. 2. Input the email or username connected to the account.

	3. Click on the email link to reset the password. 4. Finish the 2FA verification using Microsoft Authenticator once you have changed your password.
Expected result	The reset password link has been sent to the email address. The user creates a new password and is asked to confirm with Microsoft Authenticator for 2FA. Accessing the account requires successful verification through 2-factor authentication.
Actual result	-
Status	Not implemented

Test Case ID	TC-2FA-F-014
Test Title	Email Verification and Expiration of Reset Link
Precondition	Password reset initiation is done, email with reset link is sent
Steps	1. Try using the reset link after its expiration time.
Expected result	The reset link needs to become invalid after a set amount of time (e.g., 15-30 minutes). After trying to access expired link, error message should be displayed related to that, example “Password reset link is expired”
Actual result	-
Status	Not implemented

Test Case ID	TC-2FA-F-015
Test Title	Forgot Password with Microsoft Authenticator Unavailable
Precondition	Password reset initiated
Steps	1. Create a new password. 2. If asked for 2FA, select "I don't have my Authenticator" or a similar choice.
Expected result	The system must provide different verification choices (backup codes, SMS, or email-based 2FA).

	Access will be allowed once alternative 2FA verification has been completed.
Actual result	-
Status	Not implemented

Test Case ID	TC-2FA-F-016
Test Title	Attempting to Use an Old Password After Reset
Precondition	Reset the password and log in with the new password.
Steps	1. Attempt to log in again using the old password..
Expected result	The old password should be invalid The system must reject login tries with the previous password, guaranteeing that only the new password will be accepted.
Actual result	-
Status	Not implemented

Test Case ID	TC-2FA-F-017
Test Title	Multiple "Forgot Password" Requests
Steps	1. Initiate multiple "Forgot Password" requests in a short time frame (5 min)
Expected result	The system must deactivate old reset links after a new one is created. Only the most recently generated reset link should be functional, making older links unusable.
Actual result	-
Status	Not implemented

Test Case ID	TC-2FA-F-018
Test Title	Rate-Limiting on "Forgot Password" Requests
Steps	1. Initiate multiple "Forgot Password" requests in a short time frame (5-10 request per minute)
Expected result	The system should limit the number of password reset attempts to prevent abuse. After a certain number of attempts

	(approximately 3-5), any additional requests should be temporarily restricted, with the user being notified of rate limits.
Actual result	-
Status	Not implemented

Testing Results	
Total test cases:	23
Pass:	**8**
Fail:	**9**
Not started:	6