



Web Application Penetration Test

LocalGov Drupal (LGD)

Prepared for: Invuse Limited

Prepared by: Aditya Raj Singh

Date: 2023-02-06

Disclaimer

This document and all written advice and materials provided by The SecOps Group UK Limited (The SecOps Group) to Invuse Limited (Invuse) are the intellectual property of The SecOps Group. Invuse may use these materials freely for its own business purposes, but may not distribute or reproduce this document, in whole or in part, or otherwise supply it for use by any third party, without the prior written consent of The SecOps Group.

The SecOps Group owns all intellectual property rights, including copyright, trade secrets, know-how and methodologies, in everything developed by The SecOps Group for Invuse, in whatever form and regardless of when such rights came into existence.



The SecOps Group appreciates your co-operation in protecting its intellectual property.

Index

Disclaimer	1
General Information	4
Executive Summary	5
Assessment Summary	8
Technical Details	9
1. Malicious File Upload	9
2. Missing Anti-Scripting Controls.....	14
3. Weak Password Policy	19
4. Username Enumeration	23
5. Missing Security Related Headers.....	27
6. Verbose Error Messages.....	28
7. Insufficient Session Timeout	31
8. Weak Account Lockout Mechanism.....	33
9. Verbose HTTP Response Headers	36
Appendix A: Review Methodology.....	38
Appendix B: Severity Analysis.....	38

General Information

Testing Duration

The testing activities were performed by The SecOps Group between 2023-01-25 and 2023-02-01.

Scope

Invuse required The SecOps Group to perform security assessment on the following Web Application:

- <https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site>

Rules of Engagement and Assumptions

- No Denial of Service (DoS) attacks to be performed.
- The assessment was carried out on the development environment.
- The assessment was carried out during the standard business hours.
- The following activities were out of scope:
 - API security testing.
 - Mobile application security testing.
 - Network security testing.
 - Attack surface mapping.

User Accounts

Invuse provided the following user accounts for the test:

Application	User Accounts	Role
LocalGov Drupal (LGD)	<ul style="list-style-type: none"> • Editor@invuse.com • Editor2@invuse.com 	Editor
	<ul style="list-style-type: none"> • Authoriseduser@invuse.com • Authoriseduser2@invuse.com 	Standard User
	<ul style="list-style-type: none"> • Newseditor@invuse.com • Newseditor2@invuse.com 	News Editor

Note – The user accounts created/provided for testing purposes should now be removed as the testing is complete.

Executive Summary

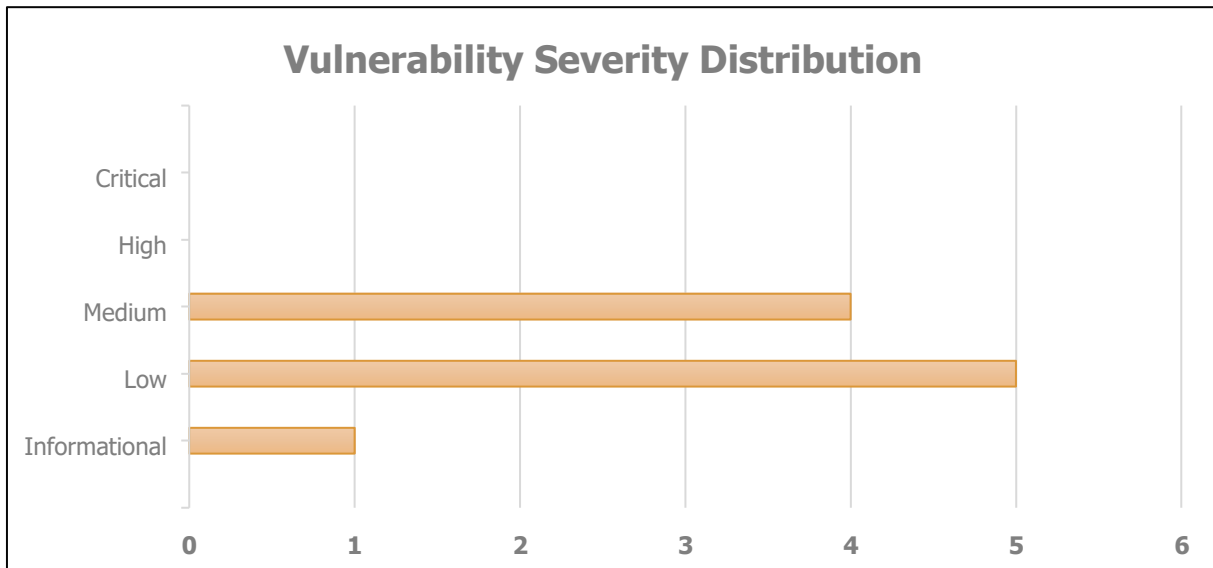
The SecOps Group conducted a comprehensive security assessment for Invuse Limited on their LocalGov Drupal (LGD) web application, to provide them an estimate of their application's existing security posture and its susceptibility to exploitation and/or data breaches. This was a grey box or authenticated type of assessment and was performed in accordance with The SecOps Group's Appendix A: Review Methodology.

During the assessment, it was observed that the application lacked input validation and processed a malicious input, which allowed the assessment team to perform Server-Side Request Forgery (SSRF), where it was possible to scan ports. This vulnerability if not mitigated can also lead to an applicationlevel denial of service attack. Further, a file upload vulnerability was discovered, which allowed the assessment team to upload malicious files on the application server. The assessment team also identified that the application lacked anti-scripting controls, which allowed several redundant requests to be sent to the server, which could negatively impact the application's performance. The assessment team also found that a strong password policy was not enforced, which allowed users to set easily- guessable passwords for their accounts. Additionally, some low-risk findings were identified, details of which are mentioned in the Technical Details section of this report.

The SecOps Group coordinated with the Invuse team to ensure safe, orderly, and complete testing of the web application in scope, within the approved scope and timelines. It was also ensured that the security issues/concerns stated by the Invuse team during the project meetings regarding the LGD web application, were addressed and reviewed.

Based on the assessment, The SecOps Group categorized the findings into **Critical / High / Medium / Low / Informational** severity risk issues, with the overall rating of the LocalGov Drupal (LGD) web application in scope to be of **Medium** risk.

Graphical Representation of the Vulnerabilities as per Risk



Positive Observations

- The application implemented access controls that prevented the assessment team from exploiting Insecure Direct Object References (IDOR).
- The assessment team did not find SQL Injection, Operating System Code Injection, or other related vulnerabilities.
- The applications were available only on encrypted channels such as TLS and no cleartext protocols were in use.
- The application implemented input validation and output encoding, which prevented the assessment team from identifying and exploiting the Cross-Site Scripting (XSS) attacks.

Findings Discovered

Key findings have been mentioned below:

- The SecOps Group identified that the application was vulnerable to SSRF attacks, which allowed port scanning, and which was leveraged to perform Cross-Site Port Attack (XSPA).
- The SecOps Group discovered that the application allowed the upload of malicious files on the server.
- The SecOps Group identified that the application lacked anti-scripting controls, which allowed several redundant requests to be sent to the server.
- The SecOps Group found that a strong password policy was not enforced on the server-side, which allowed users to set simple passwords for their accounts.

Recommendations

Recommendations for the key findings have been mentioned below:

- Implement a strong input validation on the server side against all user input and implement a whitelist, and any requests containing invalid resources should be rejected.
- Validate the files uploaded to the application to ensure that the uploaded content matches only types allowed by the application.
- Implement anti-scripting controls such as a CAPTCHA to stop automated bots from attacking the application.
- Implement a strong password policy and ensure that server-side validation of the policy is in place.

Assessment Summary

Overall Rating

Overall rating has been identified as **Medium**.

Vulnerability	Severity	Affected Resources
This finding has been reported directly to Drupal.org security team for review, following their defined processes.	Request Information	Update to be provided to LGD community
Malicious File Upload	Medium	<ul style="list-style-type: none"> https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/ <ul style="list-style-type: none"> "Add file" functionality
Missing Anti-Scripting Controls	Medium	<ul style="list-style-type: none"> https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/ <ul style="list-style-type: none"> All the create functionalities
Weak Password Policy	Medium	<ul style="list-style-type: none"> https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/user/*/edit
Username Enumeration	Low	<ul style="list-style-type: none"> https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/user/*/edit
Missing Security Related Headers	Low	<ul style="list-style-type: none"> https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site
Verbose Error Messages	Low	<ul style="list-style-type: none"> https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/sites/default/files/styles/large_3_2_2x/public/202301/xss.gif?itok=FodxpFaz https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site <ul style="list-style-type: none"> File upload functionality
Insufficient Session Timeout	Low	<ul style="list-style-type: none"> https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site
Weak Account Lockout Mechanism	Low	<ul style="list-style-type: none"> https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/user/login
Verbose HTTP Response Headers	Informational	<ul style="list-style-type: none"> https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site <ul style="list-style-type: none"> X-Generator HTTP Response Header

Technical Details

1. Finding reported to Drupal.org Security

Severity

Request Information

Description

This finding has been reported directly to [Drupal.org](https://www.drupal.org) security team for review, following their defined processes.

Affected Resources

Update to be provided to LGD community following resolution.

Observation

N/A

2. Malicious File Upload

Severity

Medium

Description

Malicious file upload can allow attackers to upload executable or malicious code. If a malicious actor can upload malware, the malicious actor could run that malicious code on the server itself or use it to perform client-side attacks against other web application users or Administrators that might access the file.

Affected Resources

- <https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site> ○
"Add file" functionality.

Observation

The assessment team discovered that the application did not validate the contents of the uploaded file and stored it on the application server without validating it. This can be misused by an adversary to upload malicious files such as malware that could affect the application server and all its users.

Proof of Concept

The assessment team navigated to the "Create Directory page" section, filled in the form with necessary details and attached an "EICAR" file.

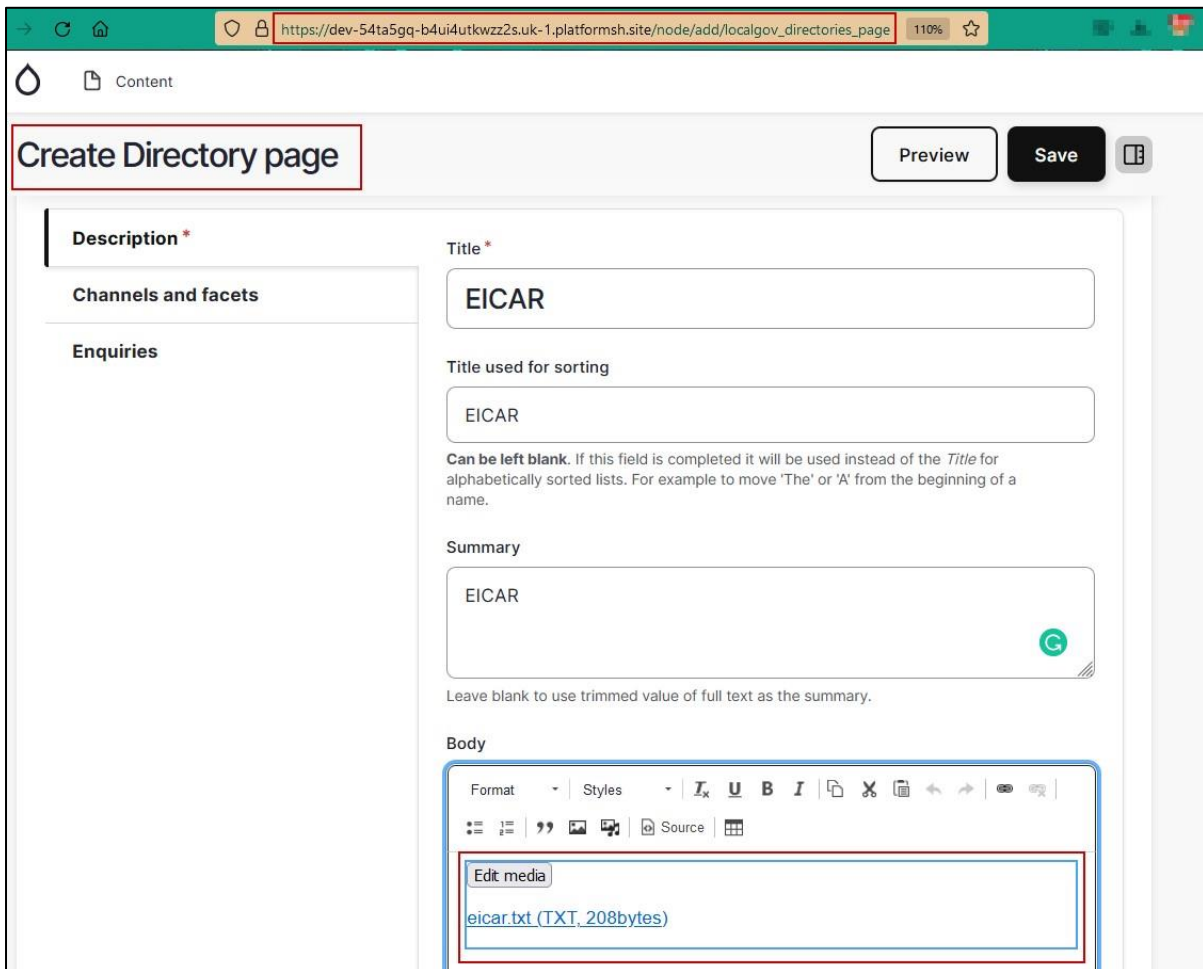


Figure 8 – Attached Malicious File

Note – EICAR is a malicious file which is used for testing purposes and is seemingly harmless. The assessment team clicked the "Save" button and found that the malicious test file was successfully uploaded to the application server.

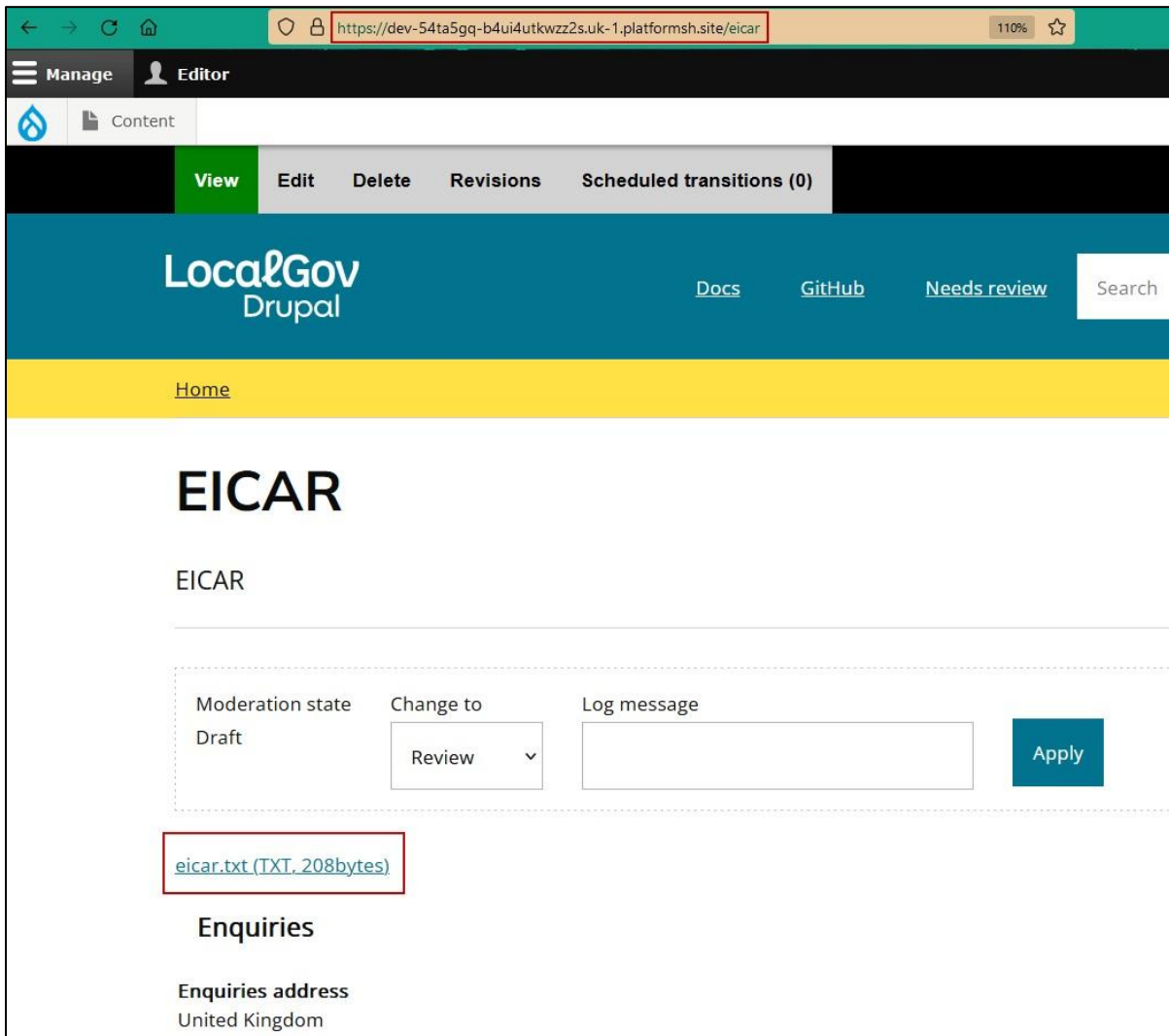


Figure 9 – Malicious Test File Uploaded Successfully

The assessment team then navigated to "https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/sites/default/files/2023-01/eicar_0.txt" and confirmed that the uploaded malicious test file was present on the application server.

Request

Pretty Raw Hex

```

1 GET /sites/default/files/2023-01/eicar_0.txt HTTP/2
2 Host: dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36

```

ⓘ ⚙ ⏪ ⏩ Search...

Response

Pretty Raw Hex Render

```

1 HTTP/2 200 OK
2 Cache-Control: max-age=300
3 Content-Type: text/plain
4 Date: Wed, 01 Feb 2023 21:36:59 GMT
5 Etag: W/"63d904c9-d0"
6 Expires: Wed, 01 Feb 2023 21:41:59 GMT
7 Last-Modified: Tue, 31 Jan 2023 12:08:41 GMT
8 Strict-Transport-Security: max-age=0
9 Vary: Accept-Encoding
10 X-Debug-Info: eyJyZXRYaWVzIjowfQ==
11 X-Platform-Cache: MISS
12 X-Platform-Cluster: b4ui4utkwzz2s-dev-54ta5gq
13 X-Platform-Processor: 5577s3j1753jfheo7a65gsj4xi
14 X-Platform-Router: iw3dd35h5s52gla51bxo62tyhe
15 X-Robots-Tag: noindex, nofollow
16 Traceresponse: 00-173fd1083eb097ebdfb7befd78d7f5d9-5b12baf413fb1ef6-00
17 Content-Length: 208
18
19 X5O!P%@AP[4\PZX54(P^)7CC)7)$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
20 X5O!P%@AP[4\PZX54(P^)7CC)7)$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
21 X5O!P%@AP[4\PZX54(P^)7CC)7)$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*

```

Figure 10 – Uploaded Malicious Test File Present on the Server

Note - The uploaded malicious file could be opened in the end user's browser with the original EICAR file content. This also indicated that the application lacked server-side anti-virus protection. The assessment team uploaded the malicious test file to "VirusTotal" and confirmed that it was malicious and was detected by multiple security vendors.

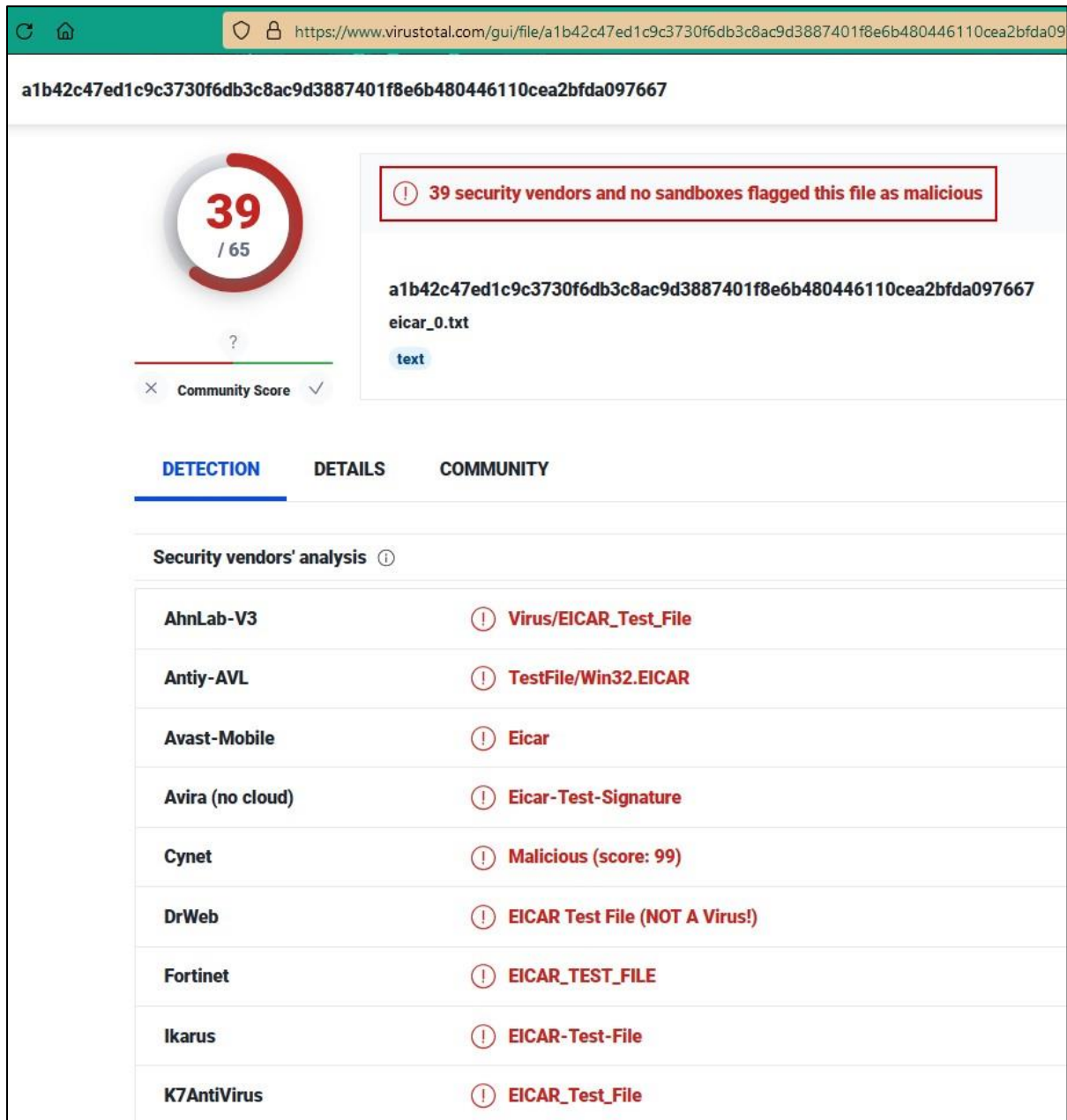


Figure 11 – Confirmation Using VirusTotal

Recommendation

- Examine the content of uploaded files.
- Check all the uploaded files for HTML/JavaScript tags and viruses.
- If web application users can download uploaded files, provide a Content-type header, and a content-disposition header which specifies that browsers should handle the file as an attachment.

References

https://cheatsheetseries.owasp.org/cheatsheets/File_Upload_Cheat_Sheet.html

http://www.eicar.org/anti_virus_test_file.htm

3. Missing Anti-Scripting Controls

Severity

Medium

Description

Web applications process numerous calls from multiple clients, but there is a limit to the number that they can handle within a certain time. As the number of concurrent calls increase, the web application may reach that limit, which could impact an organization's service uptime.

Affected Resources

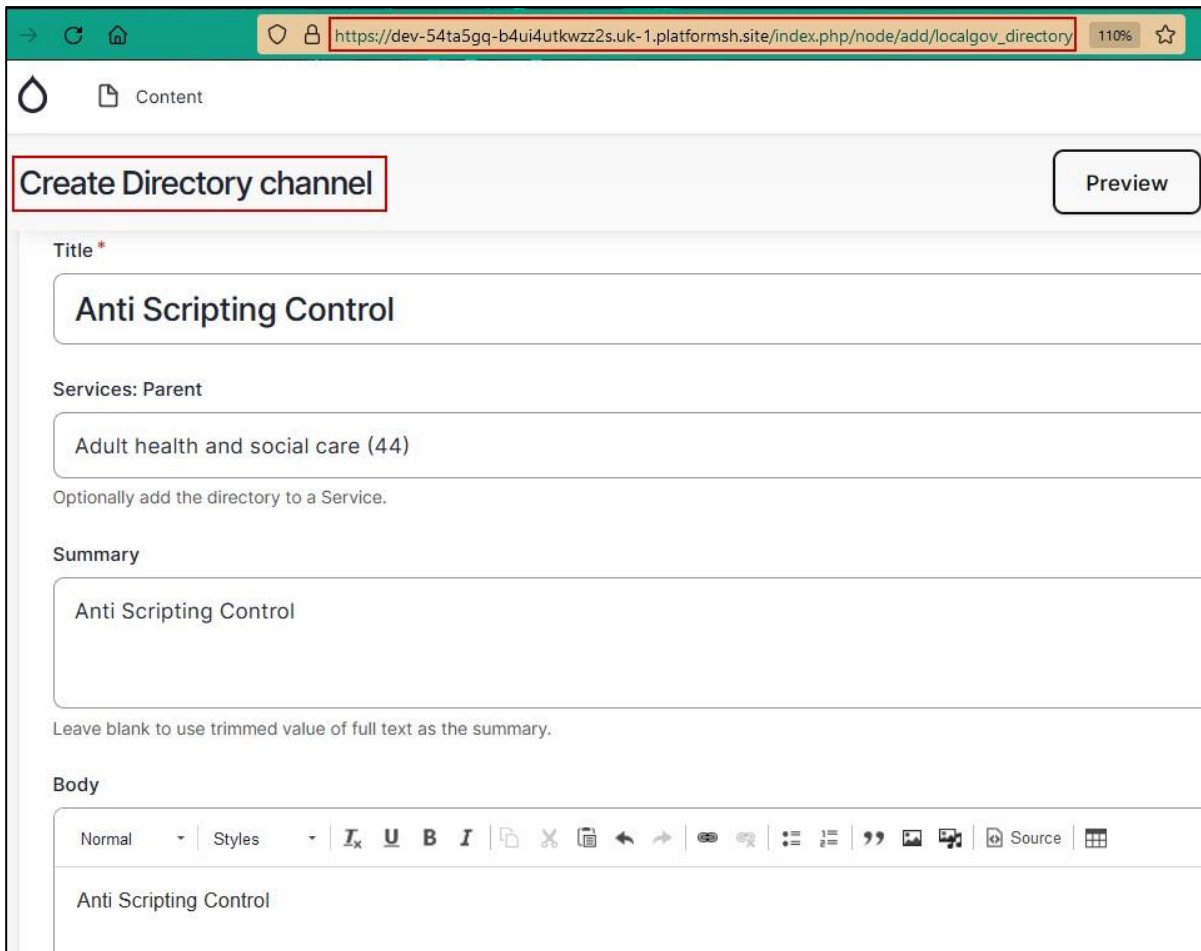
- <https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site> ○
All the "create" functionalities.

Observation

The assessment team discovered that the application lacked anti-scripting controls on "Add Content", "Media" and other similar types of functionalities throughout the application. This issue was leveraged by the assessment team for creating several posts. An adversary might also leverage this misconfiguration for uploading many files to the application server, causing the application server's resources to be depleted and resulting in denial-of-service attacks.

Proof of Concept

The assessment team logged into the application, navigated to the "Create Directory Channel" section, and filled up the form with the necessary details.



The screenshot shows a web browser window with the address bar containing the URL: `https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/index.php/node/add/localgov_directory`. The page title is "Content". The main heading is "Create Directory channel" with a "Preview" button to its right. The form fields are as follows:

- Title ***: Anti Scripting Control
- Services: Parent**: Adult health and social care (44)
- Summary**: Anti Scripting Control

Below the summary field, there is a note: "Leave blank to use trimmed value of full text as the summary." At the bottom, there is a rich text editor with the text "Anti Scripting Control".

Figure 12 – Create Directory Channel Section

The "create directory" request was intercepted using the Burp Suite proxy and was forwarded to the Intruder for further analysis.

ⓘ Payload Positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

⊕ Target:

```

1 POST /index.php/node/add/localgov_directory HTTP/2
2 Host: dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site
3 Cookie: SSE [REDACTED]
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 1417
10 Origin: https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site
11 Referer: https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/index.php/node
12 Upgrade-Insecure-Requests: 1
13
14 op=Save&title%5B0%5D%5Bvalue%5D=Anti+Scripting+Control+SS&changed=1675169627&f
mgHTuXYjkXvgOWLZYGcO5jIbapA06fTFjApPYzct3yg&form_id=node_localgov_directory_fc
Adult+health+and+social+care+%2844%29&body%5B0%5D%5Bsummary%5D=Anti+Scripting+
body%5B0%5D%5Bformat%5D=wysiwyg&localgov_directory_facets_enable%5Bcountry%5D=
organisation_type&localgov_directory_facets_enable%5Btype_of_support%5D=type_c
localgov_directories_org&localgov_directory_channel_types%5Blocalgov_directori
localgov_directory_channel_types%5Blocalgov_directories_venue%5D=localgov_dire
localgov_directory_channel_types%5Blocalgov_directory_promo_page%5D=localgov_d
moderation_state%5B0%5D%5Bstate%5D=draft&localgov_review_date%5B0%5D%5Breview%

```

Figure 13 – Intercepted Request Using Burp Suite

The intercepted request was replayed fifty times using the Burp Intruder.

Request	Payload	Status	Error	Timeout	Length	Comment
50	50	303	<input type="checkbox"/>	<input type="checkbox"/>	1597	
49	49	303	<input type="checkbox"/>	<input type="checkbox"/>	1597	
48	48	303	<input type="checkbox"/>	<input type="checkbox"/>	1597	
47	47	303	<input type="checkbox"/>	<input type="checkbox"/>	1597	
46	46	303	<input type="checkbox"/>	<input type="checkbox"/>	1597	
45	45	303	<input type="checkbox"/>	<input type="checkbox"/>	1597	
44	44	303	<input type="checkbox"/>	<input type="checkbox"/>	1597	
43	43	303	<input type="checkbox"/>	<input type="checkbox"/>	1597	
42	42	303	<input type="checkbox"/>	<input type="checkbox"/>	1597	
41	41	303	<input type="checkbox"/>	<input type="checkbox"/>	1597	
40	40	303	<input type="checkbox"/>	<input type="checkbox"/>	1597	

Request	Response
Pretty <u>Raw</u> Hex	<pre> 1 POST /index.php/node/add/localgov_directory HTTP/2 2 Host: dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site 3 Cookie: SSE=... 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,im 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Content-Type: application/x-www-form-urlencoded 9 Content-Length: 1420 10 Origin: https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site 11 Referer: https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/index.php/n 12 Upgrade-Insecure-Requests: 1 13 Connection: close 14 15 op=Save&title%5B0%5D%5Bvalue%5D=Anti+Scripting+Control+50&changed=167516962 directory_facets_enable%5Bcountry%5D=country&localgov_directory_facets_enab ov_directory_promo_page%5D=localgov_directory_promo_page&revision_log%5B0%5 </pre>

Figure 14 – Directory Channel Creation Request Replayed Fifty Times

The assessment team then navigated to the “Content” section of the application and observed that the requests were successful, and fifty posts were created.

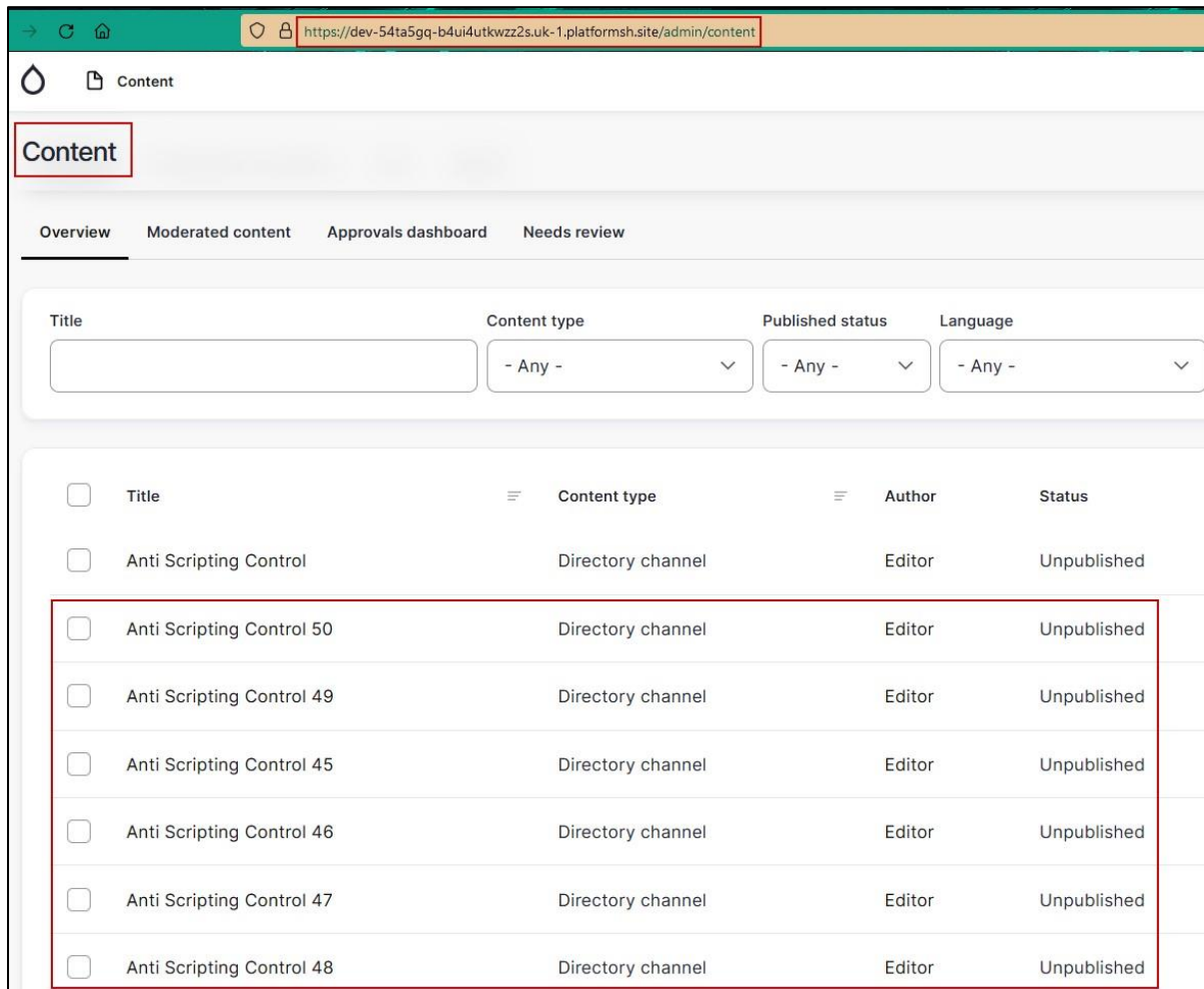


Figure 15 – Successfully Created Directory Channels

Recommendation

Limit the number of requests that can be made by authenticated and unauthenticated users. Consider implementing limits for the number of requests that authenticated users can make per second. REST API standards recommend returning an "HTTP 429" header to inform the user that too many requests were made.

References

https://cheatsheetseries.owasp.org/cheatsheets/REST_Security_Cheat_Sheet.html

https://cheatsheetseries.owasp.org/cheatsheets/Denial_of_Service_Cheat_Sheet.html

4. Weak Password Policy

Severity

Medium

Description

A simple password is also simple to guess. A malicious actor can perform password guessing and access any user account if a strong password policy is not set. A strong password policy ensures that the passwords are complex and contains a mix of letters in upper and lower cases, numbers, and special characters.

Affected Resources

- https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/user/*/edit

Observation

The assessment team observed that the application lacked a strong password policy validation on the server side and allowed the user to set a weak password such as "1", "mypassword" and "user123".

Proof of Concept

The assessment team logged into the application and navigated to the "Editor" section.

→ ↻ 🏠 🔒 <https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/user/4/edit>

🔍 📄 Content

Editor

Current password

Required if you want to change the *Email address* or *Password* below. Reset your password.

Email address*

The email address is not made public. It will only be used if you need to be contacted about your account or for opted-in notifications.

Password

To change the current user password, enter the new password in both fields.

Figure 16 – Edit Section

The assessment team then supplied the Current password and entered a sample password in the "Password" and "Confirm Password" fields.

The screenshot shows a web browser window with the address bar containing the URL: <https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/user/4/edit>. The page title is "Editor". Below the title, there are three main sections:

- Current password:** A text input field containing 12 black dots. Below it, a note states: "Required if you want to change the *Email address* or *Password* below. Reset your password."
- Email address *:** A text input field containing the email address "aditya@secops.group". Below it, a note states: "The email address is not made public. It will only be used if you need to be contacted about your account or for opted-in notifications."
- Password:** A text input field containing 8 black dots. Below it is a password strength indicator bar that is approximately 50% yellow and 50% grey. Below the bar, it says "Password strength: **Fair**".

Below the password section, there is a **Confirm password** section with a text input field containing 8 black dots. At the bottom of the form, it says "Passwords match: **yes**".

Figure 17 – Sample Password

Further, the assessment team changed the new password to a single character and forwarded the request.



Figure 18 – Single Character Password

The assessment team tried to login into the LGD application using a valid username and a single character password and observed that the login was successful.

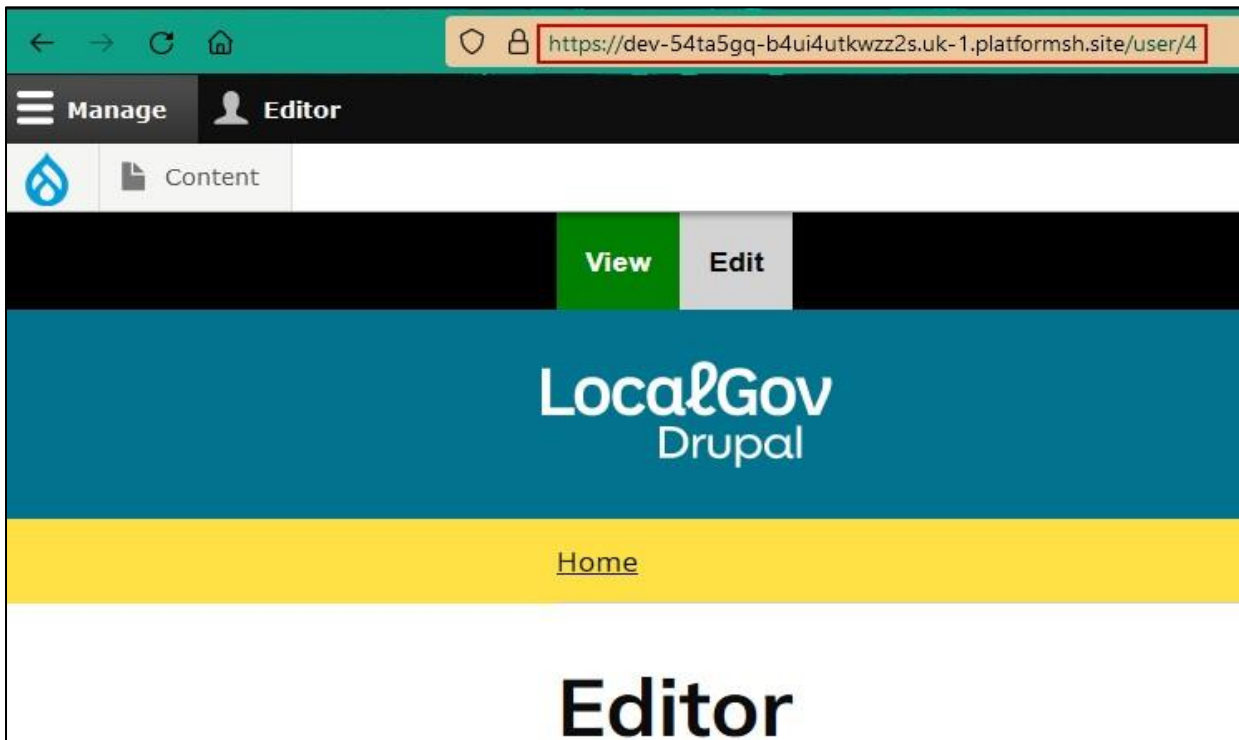


Figure 19 – Successful Login Using the Single Character Password

Recommendation

Follow recommended password protection guidance as detailed by NCSC

<https://www.ncsc.gov.uk/collection/small-business-guide/using-passwords-protect-your-data>.

Do not allow significant portions of the user's account name, company name or full name as passwords.

References

[https://owasp.org/www-project-web-security-testing-guide/latest/4-](https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/04-Authentication_Testing/07Testing_for_Weak_Password_Policy)

[Web_Application_Security_Testing/04-](https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/04-Authentication_Testing/07Testing_for_Weak_Password_Policy)

[Authentication_Testing/07Testing_for_Weak_Password_Policy](https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/04-Authentication_Testing/07Testing_for_Weak_Password_Policy)

<https://pages.nist.gov/800-63-3/sp800-63b.html>

5. Username Enumeration

Severity

Low

Description

Username Enumeration occurs when a malicious actor can determine the valid users of an application/system. This vulnerability usually exists on the login or forgot password page of an application, where an error message reveals that a username is present or absent on the system when valid or invalid credentials are entered. After enumerating valid users, a malicious actor can gain access to the system using password guessing or automated brute-force attacks. Username enumeration essentially occurs when an application gives different responses when valid and invalid data in various fields are entered.

Affected Resources

- https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/user/*/edit

Observation

The assessment team observed that the application provided different responses when a valid and then an invalid email was entered in the 'Password Reset' functionality. These different responses allowed the assessment team to determine the valid users of the application.

Proof of Concept

The assessment team logged into the application and navigated to the "Edit" section.

→ ↻ 🏠 🔒 <https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/user/6/edit>

🔍 Content

News

Current password

Required if you want to change the *Email address* or *Password* below. **Reset your password.**

Email address *

The email address is not made public. It will only be used if you need to be contacted about your account or for opted-in notifications.

Password

To change the current user password, enter the new password in both fields.

Figure 20 – Edit Section

The assessment team then supplied an existing user's email in the "Email address" field and observed that the application generated the following email, which confirmed that a valid user with the entered email already existed in the application.

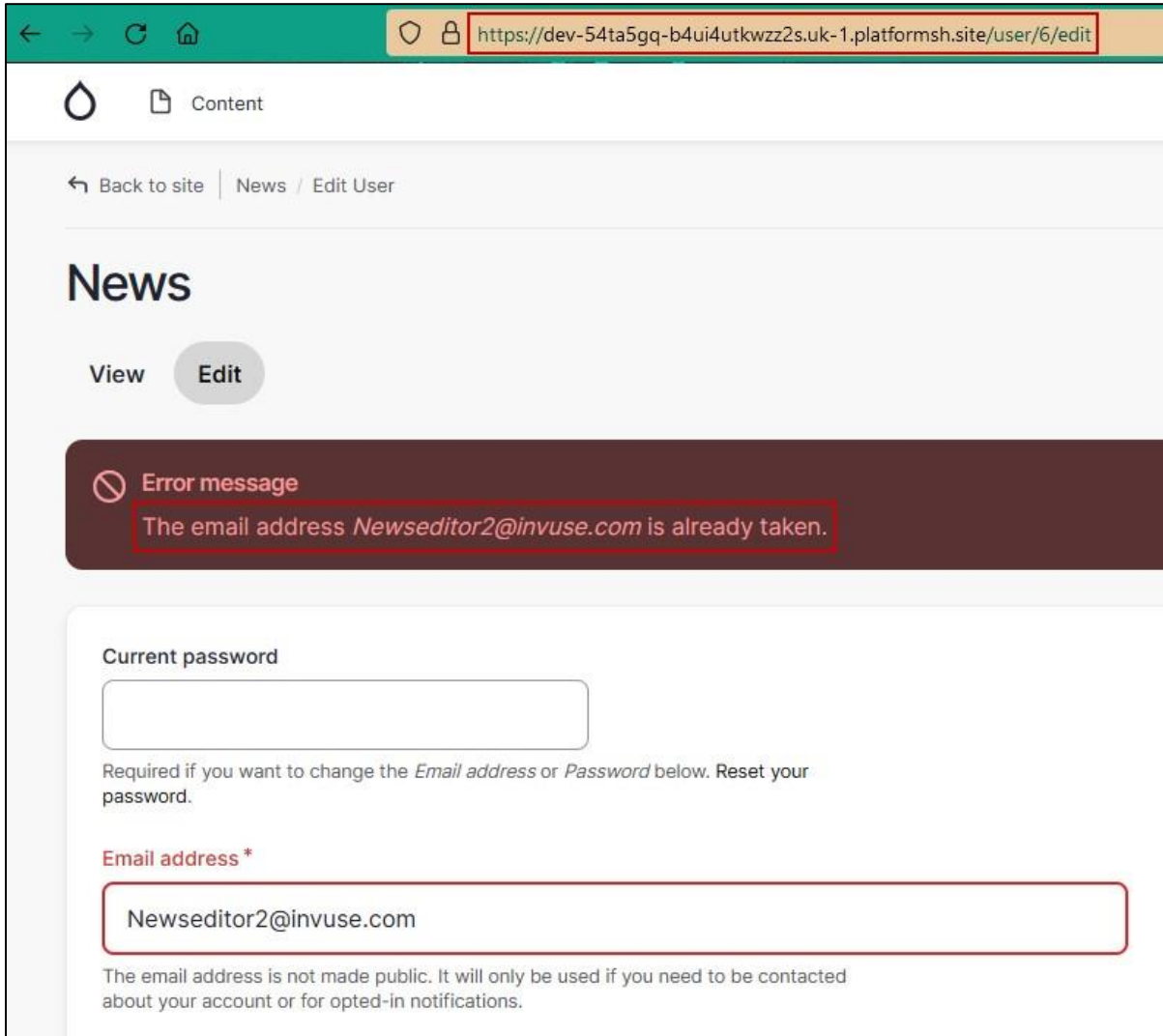


Figure 21 – Error Indicating an Existing User

The assessment team then supplied a non-existing user's email in the "Email address" field and observed that the email was updated successfully, thus confirming that the user with the entered email did not exist in the application.

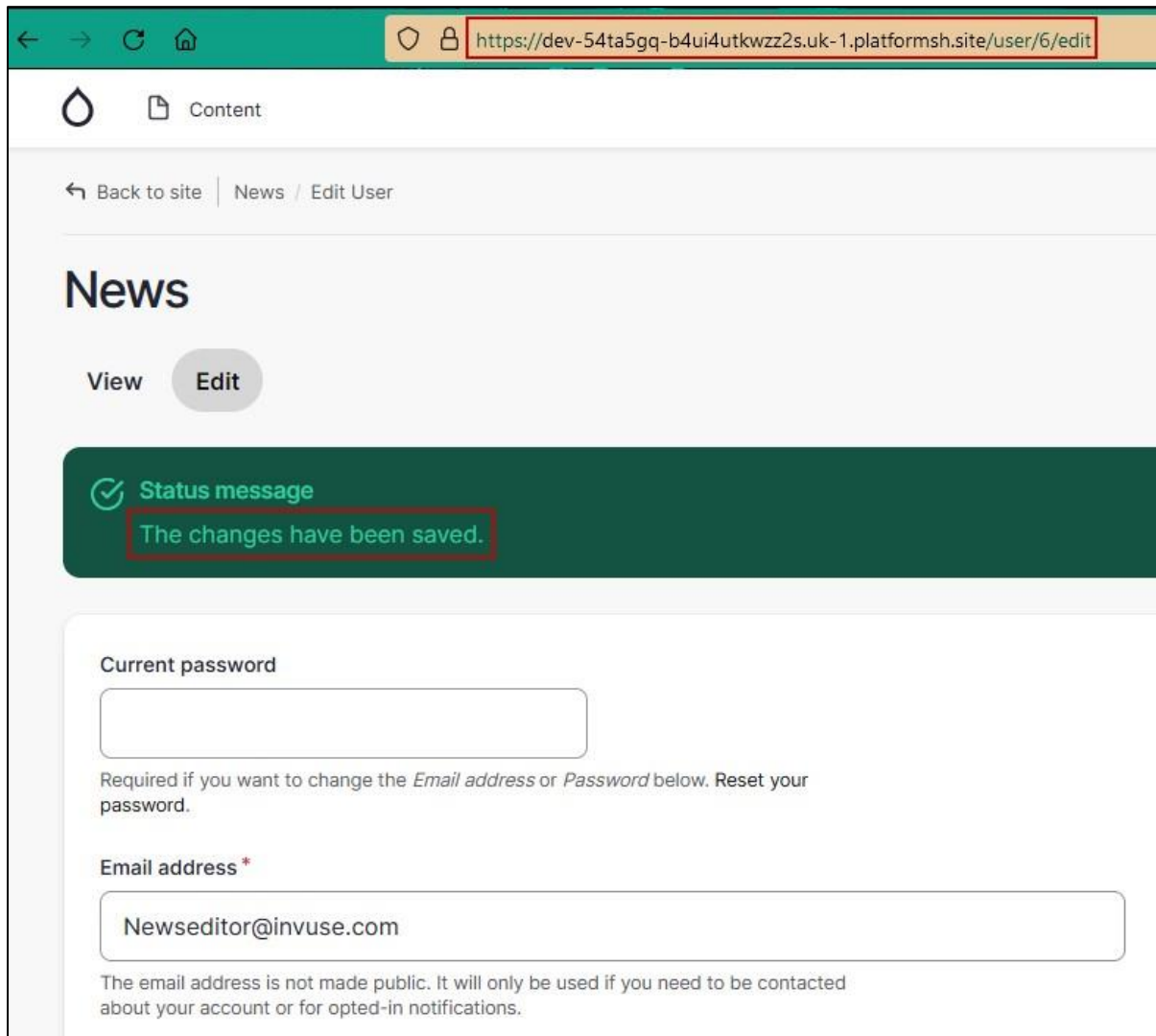


Figure 22 – Message Indicating a Non-Existing User

Recommendation

- Configure web applications so that error messages do not indicate whether a user account had been correct or not.
- For login forms, use a generic error message such as 'Invalid User ID or Password' for all failed logins.
- For password reset forms, report that instructions have been sent to the email address on file, regardless of whether the submitted username was correct or not.

References

<https://cwe.mitre.org/data/definitions/204.html>

https://www.owasp.org/index.php/Brute_force_attack

https://www.owasp.org/index.php/Forgot_Password_Cheat_Sheet

[https://www.owasp.org/index.php/Testing_for_User_Enumeration_and_Guessable_User_Account_\(OWASP-AT-002\)](https://www.owasp.org/index.php/Testing_for_User_Enumeration_and_Guessable_User_Account_(OWASP-AT-002))

6. Missing Security Related Headers

Severity

Low

Description

The application did not implement certain HTTP security headers, which help in protecting the application against attacks including Cross-site Scripting (XSS) and Clickjacking.

Affected Resources

- <https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site>

Observation

The assessment team found that the following security headers were missing in the web application response:

- Content-Security-Policy
- Referrer-Policy
- Permissions-Policy

Proof of Concept

The assessment team analyzed the security headers of the application using "Shcheck.py" and observed that the application lacked three security-related headers.

```

└─$ shcheck.py https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site
=====
> shcheck.py - santoru .....
-----
Simple tool to check security headers on a webserver
=====

[*] Analyzing headers of https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site
[*] Effective URL: https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site
[*] Header X-Frame-Options is present! (Value: SAMEORIGIN)
[*] Header X-Content-Type-Options is present! (Value: nosniff)
[*] Header Strict-Transport-Security is present! (Value: max-age=0)
[!] Missing security header: Content-Security-Policy
[!] Missing security header: Referrer-Policy
[!] Missing security header: Permissions-Policy
[!] Missing security header: Cross-Origin-Opener-Policy
[!] Missing security header: Cross-Origin-Resource-Credentials
[!] Missing security header: Expect-CT
-----
[!] Headers analyzed for https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site
[+] There are 3 security headers

```

Figure 23 – Missing Security Related HTTP Headers

Recommendation

Implement the security-related HTTP headers to improve the overall security posture of the application.

References

<https://tools.ietf.org/html/rfc7234>

<https://cheatsheetseries.owasp.org/cheatsheets/HTTP-Headers-Cheat-Sheet.html>

7. Verbose Error Messages

Severity

Low

Description

Verbose error message is when the application throws sensitive error messages such as stack traces, database queries or dumps and error codes. These error messages can be the first line of attack point where an attacker is able to get the information about the application’s underlying technology like the software or framework name and versions. An attacker can accordingly search for vulnerabilities and exploits to harm the application or system, users, and technology.

Affected Resources

- https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/sites/default/files/styles/large_3_2_2x/public/202301/xss.gif?itok=FodxpFaZ
- <https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site> ○ File upload functionality

Observation

The assessment team observed that the application lacked a robust error handling mechanism and produced a verbose error message, containing the application's stack trace and revealing internal paths and other relevant details, which can be used by an adversary in crafting further attacks.

Proof of Concept

The assessment team navigated to the "Create Directory channel" section and attached a broken GIF image.

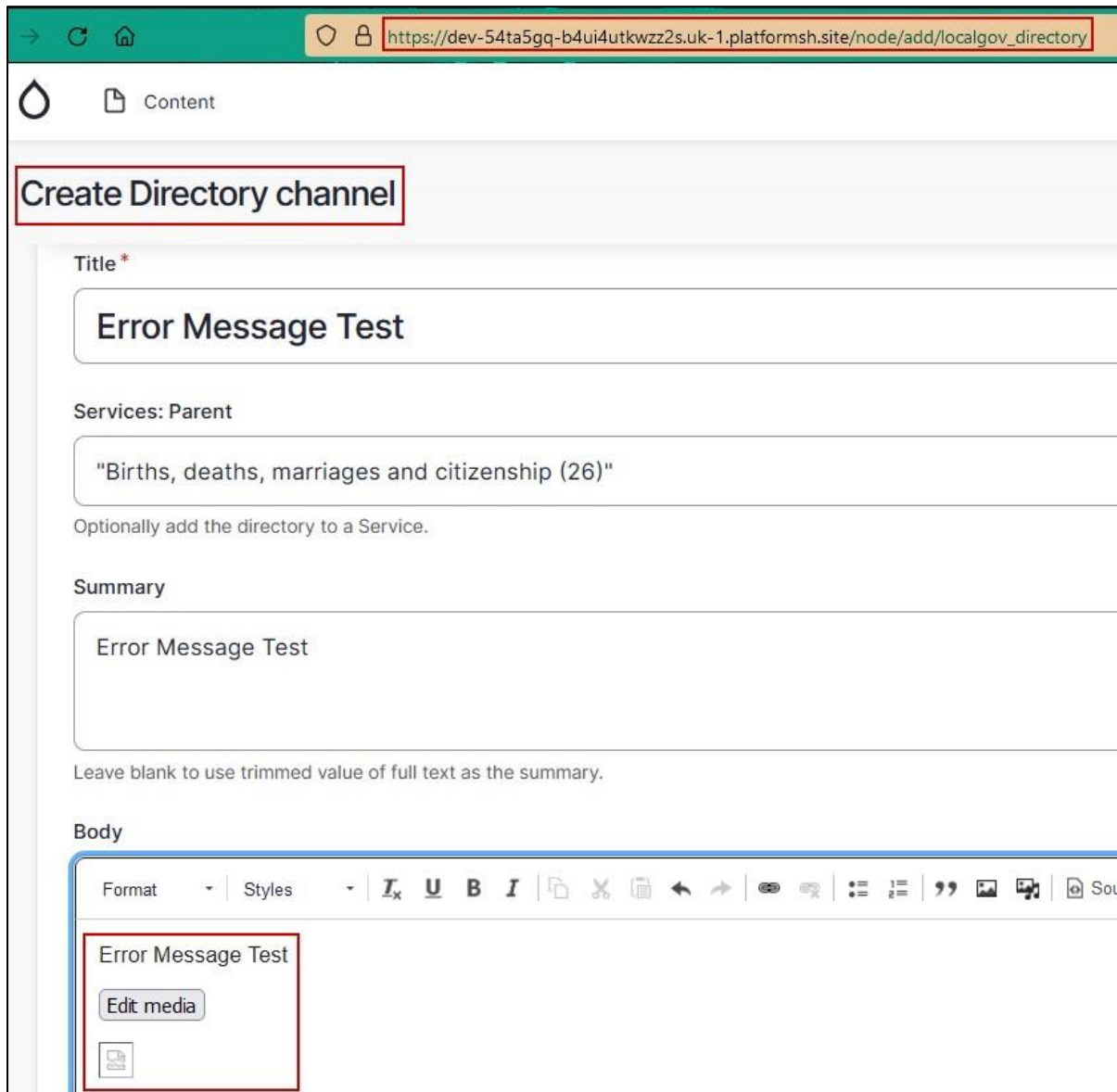


Figure 24 – Attached Broken GIF Image

The assessment team then clicked on the "Preview" button and observed that the application generated a Stack Trace error message.

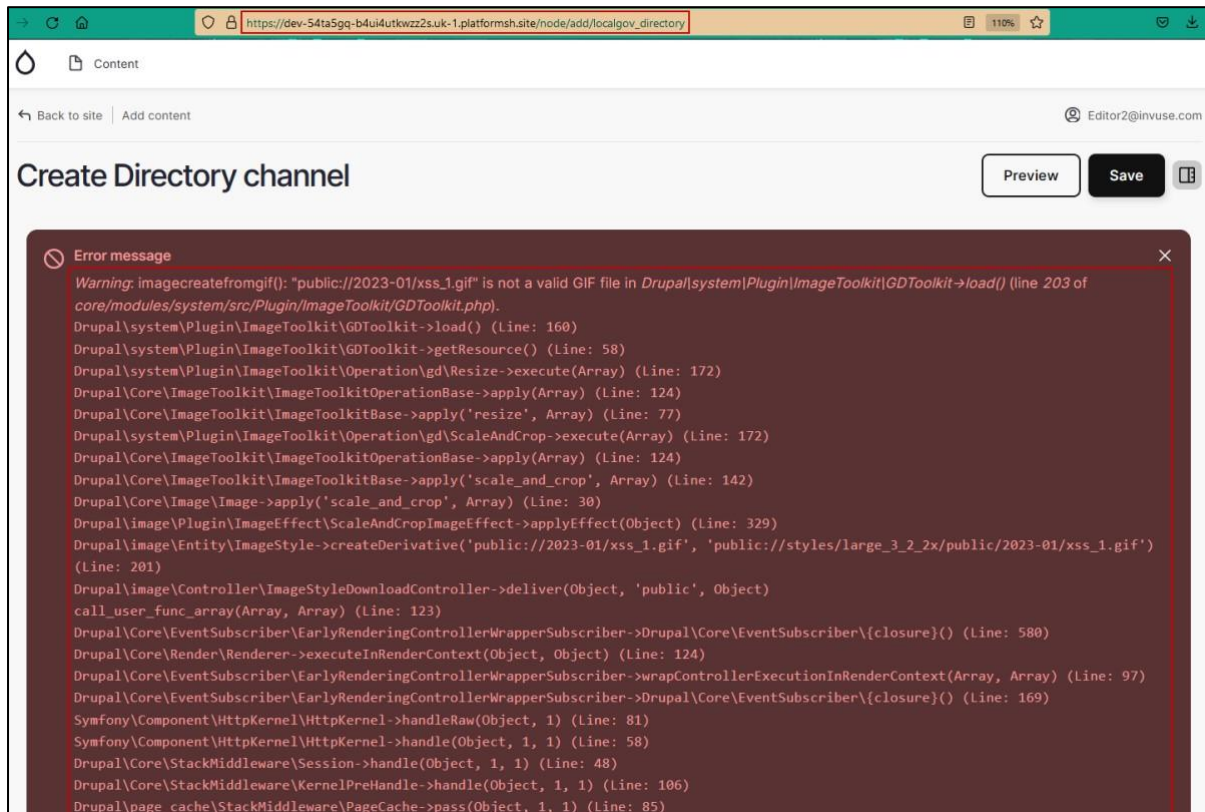


Figure 25 – Verbose Error Message

Recommendation

- Error handling should be properly implemented by the developers in the application's code to avoid revealing unnecessary details or sensitive information. Custom error pages can be created.

References

https://owasp.org/www-community/Improper_Error_Handling

https://projects.webappsec.org/f/WASC-TC-v1_0.txt

https://www.owasp.org/index.php/OWASP_Periodic_Table_of_Vulnerabilities__Information_Leakage

8. Insufficient Session Timeout

Severity

Low

Description

Session timeout occurs when a user does not perform any action on the website in the given time frame or logs out of the application. This time is set at the web server. Application not having a timeout or having an insufficient session timeout can lead to the misuse of the session ID where a malicious actor can steal or reuse any user’s session identifiers. A session must be invalidated on the server side once a user logs out or leaves the session idle.

Affected Resources

- <https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site>

Observation

The assessment team observed that the application had an insufficient session timeout mechanism and allowed a session of 23 days.

Proof of Concept

The assessment team logged into the LGD application and observed that the application had Insufficient session timeout and allowed a session for 23 days.

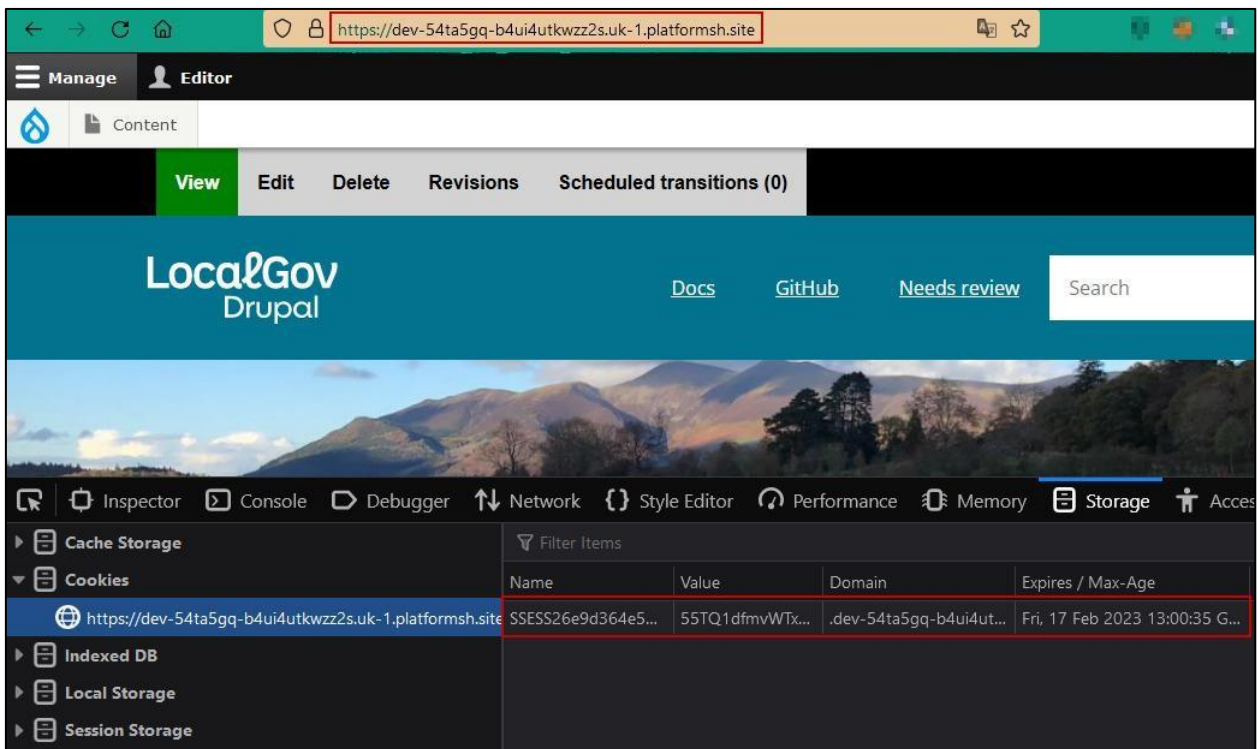


Figure 26 – Insufficient Session Timeout

Recommendation

- It is recommended to set the session timeout value to 2-5 minutes if the application contains high-risk sensitive data. Implement the logout functionality in the application to destroy the session identifiers. Invalidate the session ID after the use by the users to avoid reusing by an attacker.

References

https://owasp.org/www-community/Session_Timeout

9. Weak Account Lockout Mechanism

Severity

Low

Description

With an insufficient account lockout policy, malicious actors could perform automated dictionary or brute-force attacks against the user and administrative accounts. In a brute-force attack, a malicious actor will guess many passwords rapidly, looking for one password that matches the account password. These attacks often use dictionaries of the most commonly-used passwords, such as "password", "12345", or the season and the year.

Affected Resources

- <https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site/user/login>

Observation

The assessment team observed that the application had a weak account lockout policy. The application tried to prevent brute force attacks and blacklisted the tester's IP address; however, this restriction was easily circumvented by rotating the IP address and the assessment team was able to login into the application using the valid password. This indicated that an adversary could perform password-guessing attacks by simply implementing an IP rotation mechanism after 3 failed attempts.

Proof of Concept

The assessment team used the Burp Intruder to brute force the user accounts and discovered that the application restricted the IP address after four failed password-guessing attempts.

Request ^	Payload	Status	Error	Timeout	Length	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	30936	
1	root	200	<input type="checkbox"/>	<input type="checkbox"/>	30936	
2	password	200	<input type="checkbox"/>	<input type="checkbox"/>	30936	
3	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	30936	
4	test@123	200	<input type="checkbox"/>	<input type="checkbox"/>	30936	
5	admin123@	403	<input type="checkbox"/>	<input type="checkbox"/>	4498	
6	Invuse@123	403	<input type="checkbox"/>	<input type="checkbox"/>	4498	

Request	Response
Pretty Raw Hex <u>Render</u>	<div style="background-color: #fff9c4; padding: 10px;"> <p>ITHC</p> <p>Login failed</p> <p>There have been more than 5 failed login attempts for this account. It is temporarily blocked. Try again later or request a new password.</p> </div>

Figure 27 – Blocked IP Address

The assessment team rotated the IP address and tried to log into the application using the correct password, confirming that the application permitted login attempts upon IP rotation.

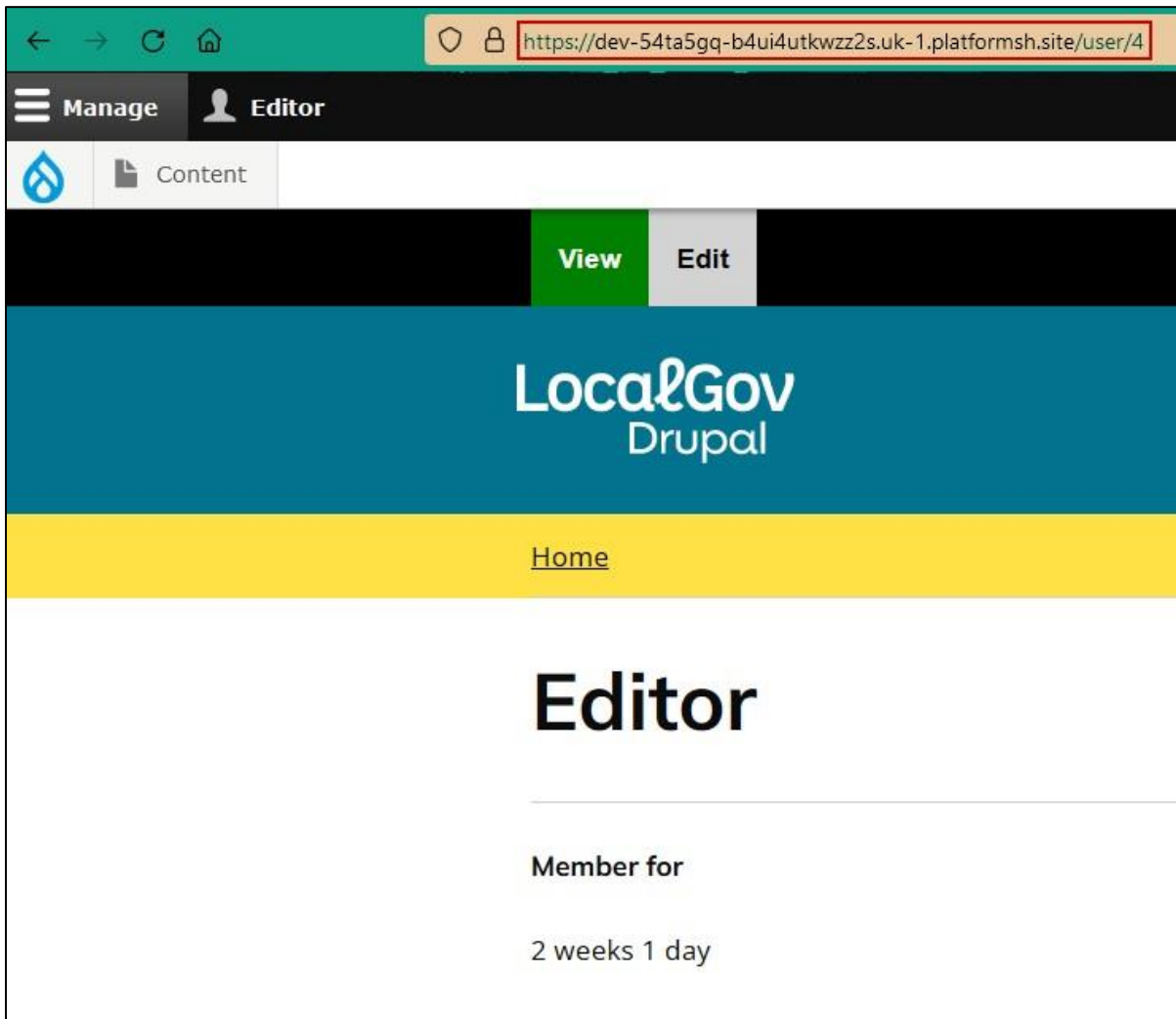


Figure 28 – Successful Login Post IP Rotation

Recommendation

- It is recommended to implement a time-based lockout. The lockout limit should be set according to the business requirement of the application. Also, implement CAPTCHA and Twofactor authentication (2FA) to further strengthen the application security.

References

https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/04-Authentication_Testing/03Testing_for_Weak_Lock_Out_Mechanism

10. Verbose HTTP Response Headers

Severity Informational

Description

In its default configuration, the application occasionally displays the server technology or CMS that it utilizes. This provides the actual version data in some cases and merely the technology name in others. In any situation, it is critical to carefully regulate the data provided in both the HTTP response header and the HTTP response body to ensure that no technical or server details are present.

Affected Resources

- <https://dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site> ○
X-Generator HTTP Response Header

Observation

The assessment team found that the application revealed the version of Drupal CMS it was using. If an exploit is released for the revealed version of Drupal CMS in the near future, this might assist an adversary in narrowing down the publicly accessible exploits for a greater probability of success.

Proof of Concept

The assessment team navigated to the LGD application and observed that the application revealed the Drupal CMS version via the X-Generator HTTP response header.

```

Request
Pretty Raw Hex
1 GET / HTTP/2
2 Host: dev-54ta5gq-b4ui4utkwzz2s.uk-1.platformsh.site
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/109.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8

Response
Pretty Raw Hex Render
1 HTTP/2 200 OK
2 Cache-Control: must-revalidate, no-cache, private
3 Content-Language: en
4 Content-Type: text/html; charset=UTF-8
5 Date: Mon, 30 Jan 2023 09:43:26 GMT
6 Expires: Sun, 19 Nov 1978 05:00:00 GMT
7 Strict-Transport-Security: max-age=0
8 X-Content-Type-Options: nosniff
9 X-Debug-Info: eyJyZXRYaWVzIjowfQ==
10 X-Drupal-Cache: HIT
11 X-Drupal-Cache-Contexts: languages:language_interface preview_link_route route theme timezon
12 X-Drupal-Cache-Max-Age: -1 (Permanent)
13 X-Drupal-Cache-Tags: block_view config:block.block.localgov_alert_banner_scarfolk config:blo
lgov_events_search_scarfolk config:block.block.localgov_guides_contents_scarfolk config:bloc
calgovdrupal_scarfolk config:block.block.localgov_service_status_message_scarfolk config:blo
k config:block.block.localgov_step_by_step_navigation_scarfolk config:block.block.localgov_s
:image.style.small_28_9 config:image.style.small_28_9_2x config:paragraphs.settings config:r
4 paragraph:181 paragraph:182 paragraph:188 paragraph:195 paragraph:94 paragraph:95 paragrap
14 X-Drupal-Dynamic-Cache: MISS
15 X-Frame-Options: SAMEORIGIN
16 X-Generator: Drupal 9 (https://www.drupal.org)
17 X-Platform-Cache: MISS
18 X-Platform-Cluster: b4ui4utkwzz2s-dev-54ta5gq
19 X-Platform-Processor: 5577s3j1753jfheo7a65gsj4xi

```

Figure 29 – Verbose HTTP Response Headers

Recommendation

- Perform output validation to filter/escape/encode technology-specific data that is being passed from the server in an HTTP response header.

References

<https://cheatsheetseries.owasp.org/cheatsheets/HTTP-Headers-Cheat-Sheet.html>

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Server>

Appendix A: Review Methodology

The assessment has been done in 4 phases:

1. **Preparation:** This phase involves, network recon, asset walkthrough, crawling, content discovery, Open-Source Intelligence (OSINT) and understanding the logic and flow of the application and other services.
2. **Automated Scanning:** Using automated scanning tools and scripts to scan the assets for vulnerabilities.
3. **Manual Assessment:** Manual testing of the assets using proxy tools, reviewing business flow, and attempting to circumvent it.
4. **Analysis and Reporting:** The identified issues were confirmed, analyzed for severity based upon network context and a detailed report with vulnerability information, proof of concepts and recommendations was prepared.

Our testing methodologies cover comprehensive security vulnerability models, i.e., Sans 25, OWASP, Top 10, OSSTMM, etc. Our Security engagements employ automated tools (commercial, open-source and in-house tools), followed by manual testing for comprehensive assessment and convergence. All our security reviews cover the following areas (as applicable):

Coverage Area	
Asset Discovery	Data Leakage and Exposure
Recon and Open-Source Intelligence	Weak or Missing Security Policies
Authentication and Authorization Testing	Excessive Service Exposure
Security Patches and Outdated Resource	Default Credentials and Configurations
Security Configuration Check	Weak Cryptographic Implementations

Appendix B: Severity Analysis

The severity analysis of the application has been primarily based upon three factors:

Impact: How would the vulnerability affect the assets?

Likelihood: What is the likelihood of a malicious actor being able to exploit the vulnerability and how easy it would be to do so.

Risk: What risk does the vulnerability pose to the application and its users.

All the factors have been evaluated by the consultant to the best of his ability, considering the application context and other available information at the time of assessment. The following is the list of ratings provided in the report:

Critical

- Vulnerabilities have immediate impact and remediation should be implemented at maximum priority.

High

- Vulnerabilities have significant impact and remediation should be implemented at priority.

Medium

- Vulnerabilities have moderate impact and remediation should be implemented after Critical and High severity vulnerabilities have been patched.

Low

- Vulnerability exploitation is not trivial and/or exposure is minimal. Remediation should be implemented after Critical, High, and Medium severity vulnerabilities have been patched.

Informational

- Vulnerabilities have no impact Vulnerability. However, as a best practice the remedy should can be applied.