

Web Application Authentication & Vulnerability Testing on Instagram

Contents

Executive Summary.....	2
Introduction.....	2
Task Details and Working	3
Learning Outcomes.....	12
Conclusion	13

Executive Summary

This report presents a consolidated overview of authentication and vulnerability testing performed on Instagram using industry-grade tools: OWASP ZAP and Burp Suite.

The objective was to:

- Analyze how Instagram manages session authentication and resists session hijacking.
- Identify underlying security misconfigurations and header-related vulnerabilities.

Despite multiple attempts to hijack a session by replicating captured cookie values and headers, Instagram's layered security mechanisms successfully prevented unauthorized access.

Meanwhile, automated scanning through ZAP uncovered several medium to low-level vulnerabilities, demonstrating that even well-established platforms can benefit from improved content security policies and hardened cookie settings.

Introduction

The goal of this exercise was to gain hands-on experience with web application testing tools and techniques. The focus was on understanding how session cookies work, how login sessions are maintained, and how secure platforms defend against session hijacking and unauthorized access attempts. The security posture of web applications depends not only on robust backend logic but also on how they manage user sessions and enforce browser-level protections. Instagram, as a widely used platform, served as a practical case study to assess real-world defense mechanisms against common threats such as session hijacking and insecure cookie usage.

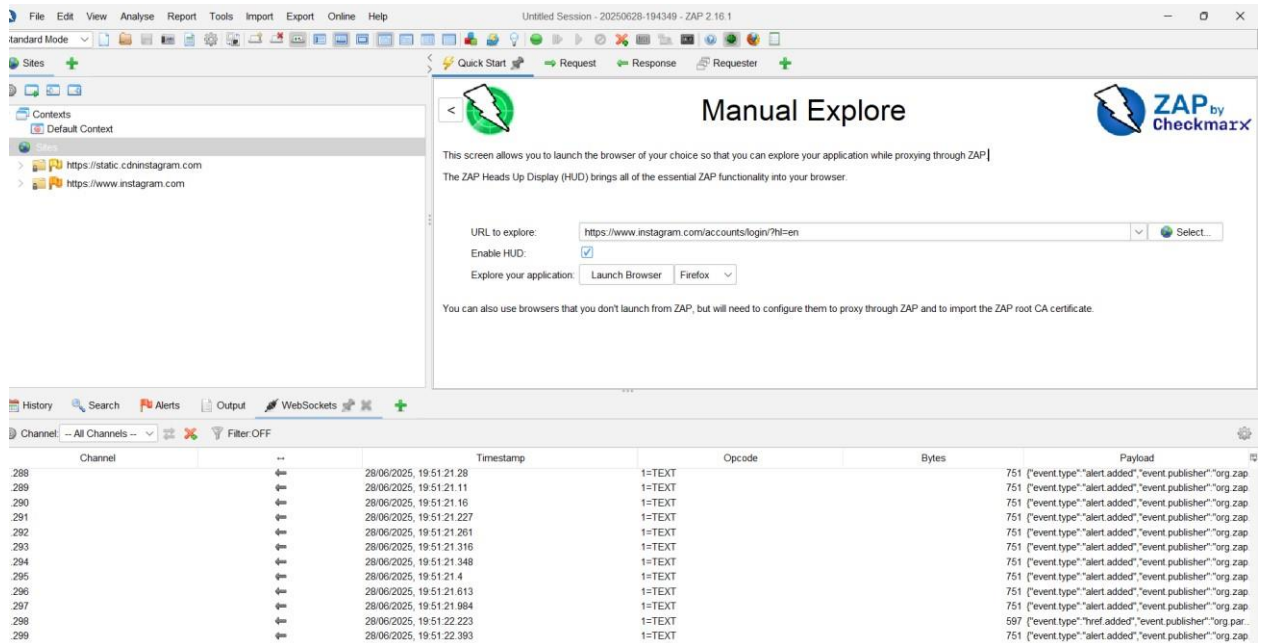
This task was divided into two major components:

- Manual **authentication testing** and **session hijack attempt** via Burp Suite.
- Automated **vulnerability scanning** of the login interface via OWASP ZAP.

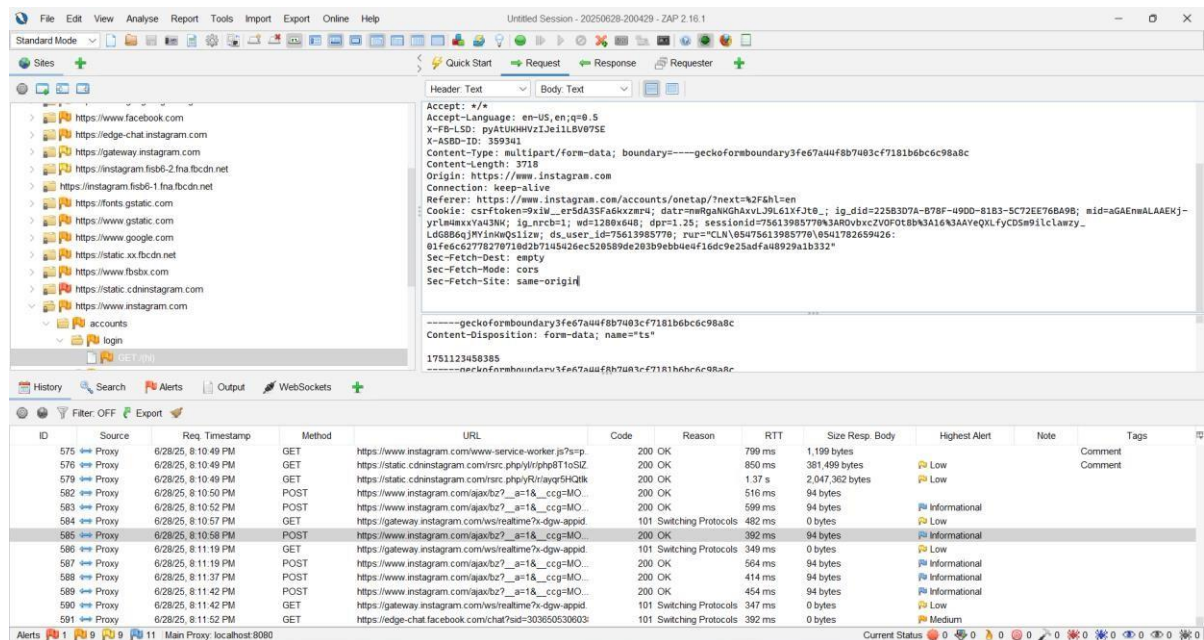
Task Details and Working

Login and Session Monitoring

- Logged into an Instagram account using OWASP ZAP's browser.



Select the post request (logged in to insta)



- Captured HTTP requests and responses, especially focusing on cookies and headers

Header: Text | Body: Text

Header: Text

Body: Text

Header: Table (adv) | Body: Text

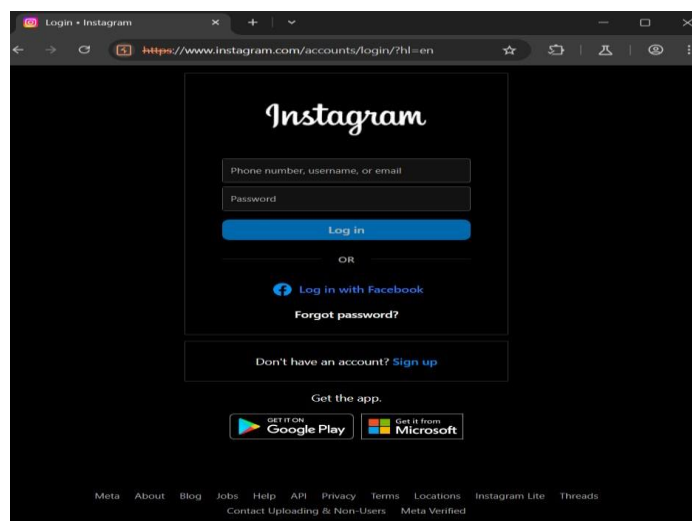
Type	Parameter Name	Value
url	jazoest	26470
url	lsd	pyAtUKHHVzJei1LBV07SE
url	ph	C3
cookie	csrftoken	9xiW__er5dA3SFa8kxzm4
cookie	datr	nwRgaNKGhAxxLJ9L61XfJt0_
cookie	dpr	1.25
cookie	ds_user_id	75613985770
cookie	ig_did	225B3D7A-B78F-49DD-81B3-5C72EE76BA9B
cookie	ig_nrcb	1
cookie	mid	aGAEnwALAAEKj-yrIm4mxY443NK
cookie	nur	"CLNl05475613985770l0541782659426:01fe6c62778270710d2b7145426ec520589de20:
cookie	sessionid	75613985770%3AR0vbxZVOFOt8b%3A16%3AAyQXlfyCDs9P1clawzy_LdG8B6qjM.
cookie	wd	1280x648

-----geckoformboundary3fe67a44f8b7403cf7181b6b6c698a8c
Content-Disposition: form-data; name="ts"
1751123458385
-----geckoformboundary3fe67a44f8b7403cf7181b6b6c698a8c

Noted down csrftoken, sessionid, ds-user-id, datr, ig_did and useragent.

Session Cookie Injection Attempt

- Opened Burp Suite's Chromium-based browser and navigated to Instagram login page.

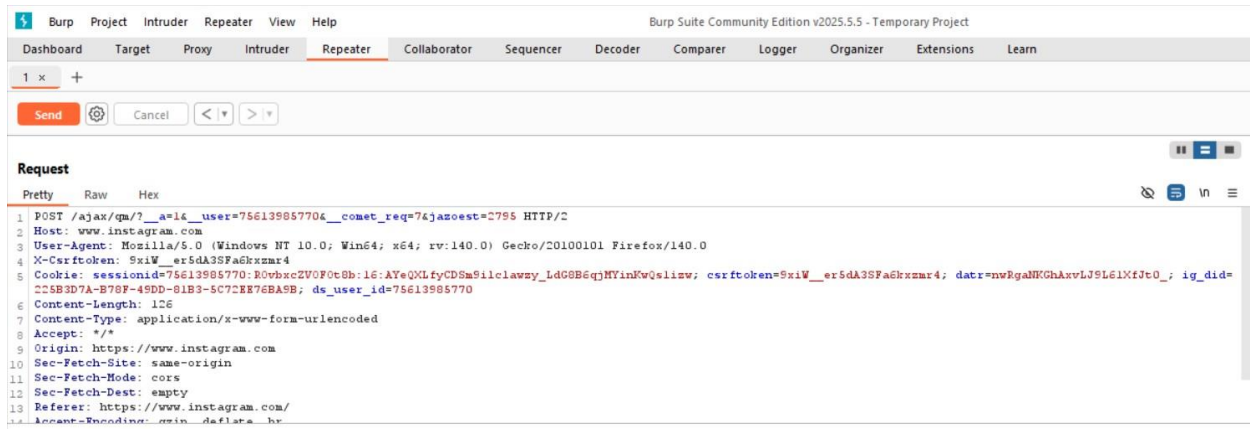


- Using the **Repeater** feature, a POST request was crafted to inject the captured cookies and headers.

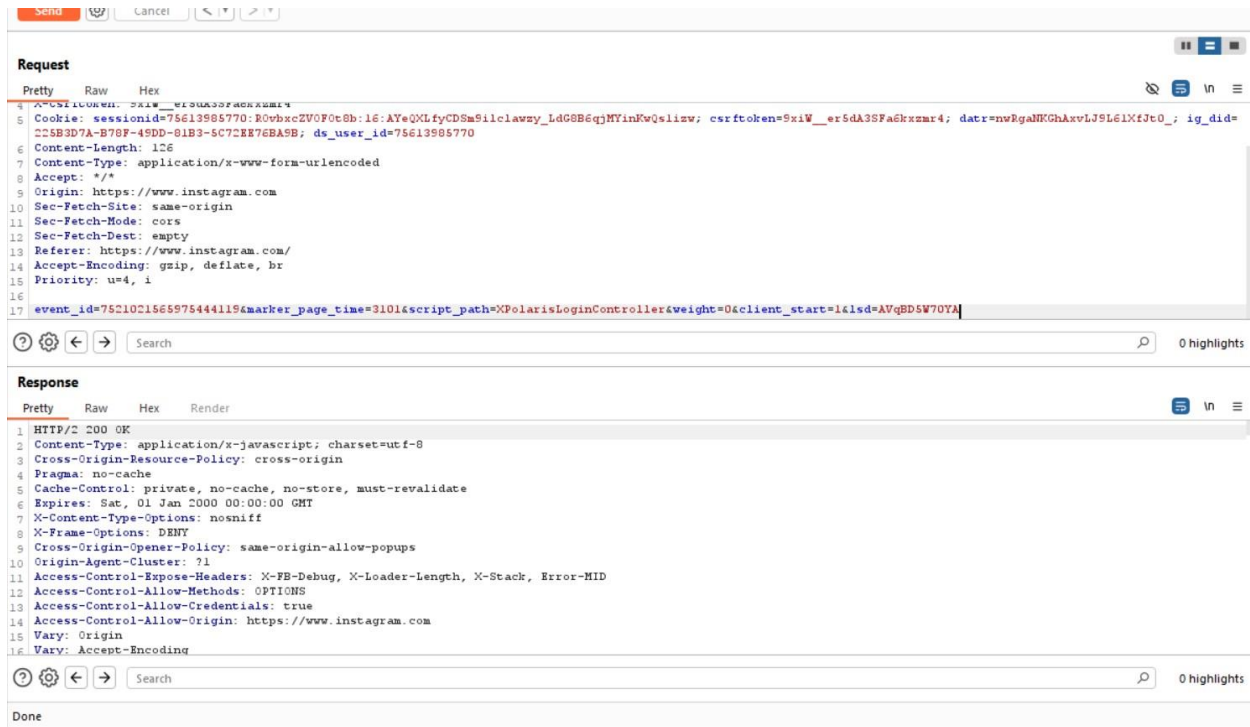
Pasted the cookies and headers into the login POST request in Burp Suite's Repeater.

- Copied cookie values
- User-Agent to match the browser used in ZAP, so it looks like the request is coming from the same device.

Response Evaluation



The response was 200ok but, No access to the authenticated session was granted.



It didn't logged me in redirected back to login page



```
24 for (;
    ;
  ) {
    {
      "ar":1,"error":1357001,"errorSummary":"Log in to continue","errorDescription":"Please log in to your account.","isNotCritical":1,"rid":"ABeQ6Up4V0kjiqfWExCpqbE"
    },
    "payload":{
      "dialog":{
        "title":{
          "html":"Not Logged In"
        },
        "body":"Please log in to continue.",
        "buttons":{"cancel",({
          "name":"login","label":"Log In","href":"https://www.instagram.com/login.php?next=https://www.instagram.com/u00253A\\u00252F\\u00252Fwww.instagram.com\\u00252F"
        })
      },
      "modal":true,
      "onloadRegisterSJSF":{
        "define":[{"cr":310,["RunVWV"],{
          "rc":["RunVWV",null]
        }}
      ]
    }
  }
}
```

Root Cause Analysis

Instagram prevented session replay by implementing:

- Device/browser fingerprinting
- TLS session binding
- Token lifecycle validation
- IP address/session flow monitoring

This showed that copying cookies alone is insufficient to bypass login security on hardened platforms.

Automated Vulnerability Scanning via OWASP ZAP

1. Scanning Phase

- Logged into an Instagram account using OWASP ZAP's browser.

File Edit View Analyse Report Tools Import Export Online Help

Untitled Session - 20250628-194349 - ZAP 2.16.1

Standard Mode

Sites

Contexts

Default Context

https://static.cdinstagram.com

https://www.instagram.com

Manual Explore

This screen allows you to launch the browser of your choice so that you can explore your application while proxying through ZAP |

The ZAP Heads Up Display (HUD) brings all of the essential ZAP functionality into your browser.

URL to explore: Select...

Enable HUD: ☒

Explore your application: Launch Browser Firefox

You can also use browsers that you don't launch from ZAP, but will need to configure them to proxy through ZAP and to import the ZAP root CA certificate.

History Search Alerts Output WebSockets

Channel: -- All Channels -- Filter OFF

Channel	Timestamp	Opcode	Bytes	Payload
288	28/06/2025, 19:51:21.28	1=TEXT	751	["event.type":"alert added","event.publisher":"org.zap
289	28/06/2025, 19:51:21.11	1=TEXT	751	["event.type":"alert added","event.publisher":"org.zap
290	28/06/2025, 19:51:21.16	1=TEXT	751	["event.type":"alert added","event.publisher":"org.zap
291	28/06/2025, 19:51:21.227	1=TEXT	751	["event.type":"alert added","event.publisher":"org.zap
292	28/06/2025, 19:51:21.261	1=TEXT	751	["event.type":"alert added","event.publisher":"org.zap
293	28/06/2025, 19:51:21.316	1=TEXT	751	["event.type":"alert added","event.publisher":"org.zap
294	28/06/2025, 19:51:21.348	1=TEXT	751	["event.type":"alert added","event.publisher":"org.zap
295	28/06/2025, 19:51:21.4	1=TEXT	751	["event.type":"alert added","event.publisher":"org.zap
296	28/06/2025, 19:51:21.613	1=TEXT	751	["event.type":"alert added","event.publisher":"org.zap
297	28/06/2025, 19:51:21.984	1=TEXT	751	["event.type":"alert added","event.publisher":"org.zap
298	28/06/2025, 19:51:22.223	1=TEXT	597	["event.type":"href added","event.publisher":"org.par
299	28/06/2025, 19:51:22.393	1=TEXT	751	["event.type":"alert added","event.publisher":"org.zap

Instagram

www.instagram.com/accounts/onetap/?next=%2F&hl=en

Instagram

Home

Search

Out

Off

0

0

0

1

1

Dashboard

Meta About Blog Jobs Help API Privacy Terms Locations Instagram Lite Threads Contact Uploading & Non-Users Meta Verified

Save your login info?

We can save your login info on this browser so you don't need to enter it again.

[Save info](#)

[Not now](#)

Sites

Start

Start

Off

0

3

5

7

+

- Running active scan to find vulnerabilities

The screenshot shows the ZAP 2.16.1 Manual Explore window. The 'URL to explore' field contains 'https://www.instagram.com/accounts/login/?hl=en'. The 'Enable HUD' checkbox is checked. The 'Explore your application' dropdown is set to 'Launch Browser'. Below the input fields, a message states: 'You can also use browsers that you don't launch from ZAP, but will need to configure them to proxy through ZAP and to import the ZAP root CA certificate.'

The 'Active Scan' tab is selected, showing a progress bar at 100%. The 'Sent Messages' table displays the following data:

ID	Req. Timestamp	Resp. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Header	Size Resp. Body
762	6/29/25, 4:22:28 PM	6/29/25, 4:22:30 PM	GET	https://www.instagram.com/accounts/login/?hl=en%22%7C	200	OK	2.5 s	5,068 bytes	614,618 bytes
764	6/29/25, 4:22:30 PM	6/29/25, 4:22:33 PM	GET	https://www.instagram.com/accounts/login/?hl=en%27%26	200	OK	2.95 s	5,044 bytes	614,630 bytes
766	6/29/25, 4:22:33 PM	6/29/25, 4:22:36 PM	GET	https://www.instagram.com/accounts/login/?hl=en%27%7C	200	OK	2.47 s	5,330 bytes	614,604 bytes
767	6/29/25, 4:22:36 PM	6/29/25, 4:22:38 PM	GET	https://www.instagram.com/accounts/login/?hl=en%28time	200	OK	2.5 s	5,325 bytes	614,559 bytes
768	6/29/25, 4:22:38 PM	6/29/25, 4:22:41 PM	GET	https://www.instagram.com/accounts/login/?hl=en%27%7Ctime	200	OK	2.06 s	5,045 bytes	614,723 bytes
769	6/29/25, 4:22:41 PM	6/29/25, 4:22:43 PM	GET	https://www.instagram.com/accounts/login/?hl=en%22%26	200	OK	2.08 s	5,044 bytes	614,601 bytes
770	6/29/25, 4:22:43 PM	6/29/25, 4:22:45 PM	GET	https://www.instagram.com/accounts/login/?hl=en%22%7C	200	OK	2.61 s	5,045 bytes	614,574 bytes
771	6/29/25, 4:22:45 PM	6/29/25, 4:22:49 PM	GET	https://www.instagram.com/accounts/login/?hl=en%27%26	200	OK	4.03 s	5,045 bytes	614,591 bytes
773	6/29/25, 4:22:49 PM	6/29/25, 4:22:51 PM	GET	https://www.instagram.com/accounts/login/?hl=en%27%7C	200	OK	1.54 s	5,325 bytes	614,536 bytes
774	6/29/25, 4:22:51 PM	6/29/25, 4:22:52 PM	GET	https://www.instagram.com/accounts/login/?hl=en%27%7C	200	OK	1.51 s	5,044 bytes	614,515 bytes
776	6/29/25, 4:22:53 PM	6/29/25, 4:22:56 PM	GET	https://www.instagram.com/accounts/login/?hl=en%3Bget+	200	OK	2.73 s	5,068 bytes	614,524 bytes

Alerts found are exported and analysed. Scanning targeted insecure headers, cookie attributes, and content security issues.

The screenshot shows the ZAP 2.16.1 Alerts window. The 'Alerts (29)' list on the left includes 'PII Disclosure (2)', 'CSP: Failure to Define Directive with No Fallback (10)', 'CSP: Wildcard Directive (4)', 'CSP: script-src unsafe-eval (2)', 'CSP: script-src unsafe-inline (6)', 'CSP: style-src unsafe-inline (10)', 'Content Security Policy (CSP) Header Not Set', 'Cross-Domain Misconfiguration (3)', 'Missing Anti-clickjacking Header (2)', 'Session ID in URL Rewrite (29)', and 'Cookie No HttpOnly Flag (12)'. The 'PII Disclosure (2)' alert is selected, showing the following details:

PII Disclosure
 URL: https://static.cdinstagram.com/btmanifest/1024298864/instagrammain
 Risk: High
 Confidence: High
 Parameter: Attack
 Evidence: 4499963836709
 CWE ID: 359
 WASC ID: 13
 Source: Passive (10062 - PII Disclosure)
 Input Vector: Description:
 The response contains Personally Identifiable Information, such as CC number, SSN and similar sensitive data.
 Other Info:

2. Alert Collection and Analysis

- Collected scan report in detail and analyzed findings based on impact and exploitability.

Alerts

Risk=Medium, Confidence=High (4)

<https://www.instagram.com> (4)

CSP: Failure to Define Directive with No Fallback (1)

► GET <https://www.instagram.com/accounts/login/?hl=en>

CSP: Wildcard Directive (1)

► GET https://www.instagram.com/common/referer_frame.php?cb=1

CSP: script-src unsafe-inline (1)

► GET https://www.instagram.com/common/referer_frame.php?cb=1

CSP: style-src unsafe-inline (1)

► GET <https://www.instagram.com/accounts/login/?hl=en>

Risk=Medium, Confidence=Medium (1)

<https://www.instagram.com> (1)

Missing Anti-clickjacking Header (1)

► GET https://www.instagram.com/common/referer_frame.php?cb=1

Risk=Low, Confidence=Medium (4)

<https://www.instagram.com> (4)

Cookie No HttpOnly Flag (1)

► GET <https://www.instagram.com/accounts/login/?hl=en>

Cookie with SameSite Attribute None (1)

► GET <https://www.instagram.com/accounts/login/?hl=en>

Cookie without SameSite Attribute (1)

► GET https://www.instagram.com/api/v1/web/login_page/?hl=en

Cross-Domain JavaScript Source File Inclusion (1)

► GET <https://www.instagram.com/accounts/login/?hl=en>

Risk=Low, Confidence=Low (1)

Risk=Informational, Confidence=High (3)

<https://www.instagram.com> (3)

Authentication Request Identified (1)

► POST <https://www.instagram.com/api/v1/web/accounts/login/ajax/?hl=en>

Information Disclosure - Information in Browser localStorage (1)

► GET <https://www.instagram.com/accounts/login/?hl=en>

Information Disclosure - Information in Browser sessionStorage (1)

► GET <https://www.instagram.com/accounts/login/?hl=en>

Risk=Informational, Confidence=Medium (3)

<https://www.instagram.com> (3)

Information Disclosure - Sensitive Information in URL (1)

► POST https://www.instagram.com/ajax/qm/?__a=1&__user=0&__comet_req=7&jazoest=21059

Modern Web Application (1)

► GET <https://www.instagram.com/accounts/login/?hl=en>

Session Management Response Identified (1)

► GET <https://www.instagram.com/accounts/login/?hl=en>

Risk=Informational, Confidence=Low (4)

<https://www.instagram.com> (2)

Loosely Scoped Cookie (1)

► GET <https://www.instagram.com/accounts/login/?hl=en>

User Controllable HTML Element Attribute (Potential XSS) (1)

► GET <https://www.instagram.com/accounts/login/?hl=en>

Identified Vulnerabilities (ZAP Findings Summary)

Vulnerability	Severity	Description	Potential Risk
Content Security Policy (CSP) Not Strict	Medium	CSP allows unsafe inline scripts or external domains.	Can lead to Cross-Site Scripting)
Missing Clickjacking Protection	Medium	No X-Frame-Options or Content-Security-Policy: frame-ancestors.	Enables UI redress attacks
Weak Cookie Attributes	Medium/Low	Session cookies lack HttpOnly, Secure, or SameSite flags.	Susceptible to session theft
Missing X-Content-Type-Options Header	Low	Allows MIME sniffing by browsers.	Can bypass content-type restrictions
Missing X-XSS-Protection Header	Informational	No built-in browser XSS filter enabled.	Reduced clientside script filtering

These vulnerabilities, though not critical on their own, can be chained by attackers to craft sophisticated exploits if left unpatched.

Learning Outcomes

- **Understanding Session Cookies:** Learned how session cookies are used to maintain login sessions and how critical they are for user authentication.
- **Real-World Web Security:** Understood how major web applications like Instagram implement strong security features to prevent session hijacking.
- **Limitations of Manual Session Replay:** Discovered the limitations of simple cookie manipulation and the necessity of matching multiple parameters such as IP, device, and TLS fingerprints.

Conclusion

This task demonstrated how real-world platforms like Instagram implement **multi-layered security mechanisms** to safeguard against session hijacking. Despite capturing and replaying valid session cookies, Instagram's backend systems validated environment fingerprinting and session flow, resulting in **login denial**.

Additionally, the vulnerability scan revealed that while Instagram is highly secure, there are still opportunities to **enhance protection** by adopting stricter Content Security Policies, **proper cookie attributes**, and enforcing anti-clickjacking headers. These findings reinforce the importance of defense-in-depth and continual vulnerability assessments in modern web application security.