Task 1: Conduct a Penetration Test

# OBJECTIVE

# Task: Perform a basic penetration test on a web application.

# Details:

# Use tools like Burp Suite or Metasploit to find security vulnerabilities.

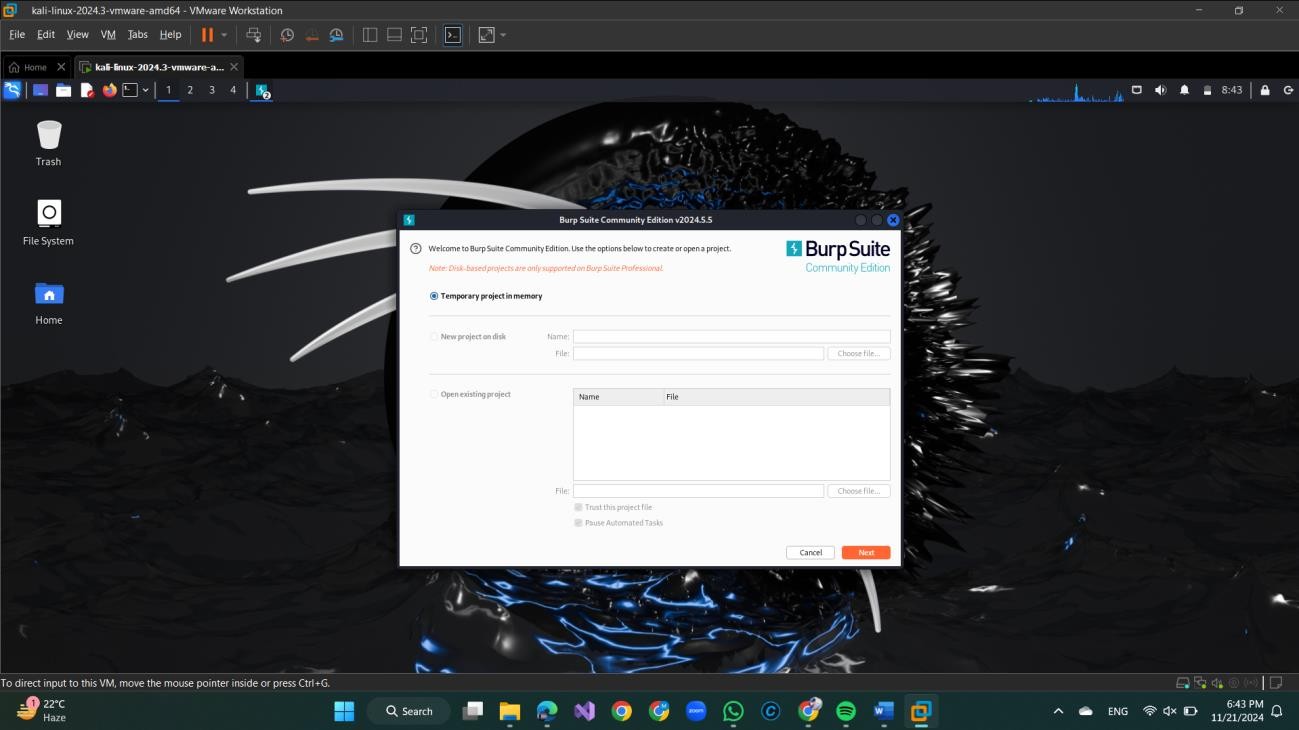
# Document findings and suggest remediation steps.

# STEPS

# Step 1: Setting Up Burp Suite

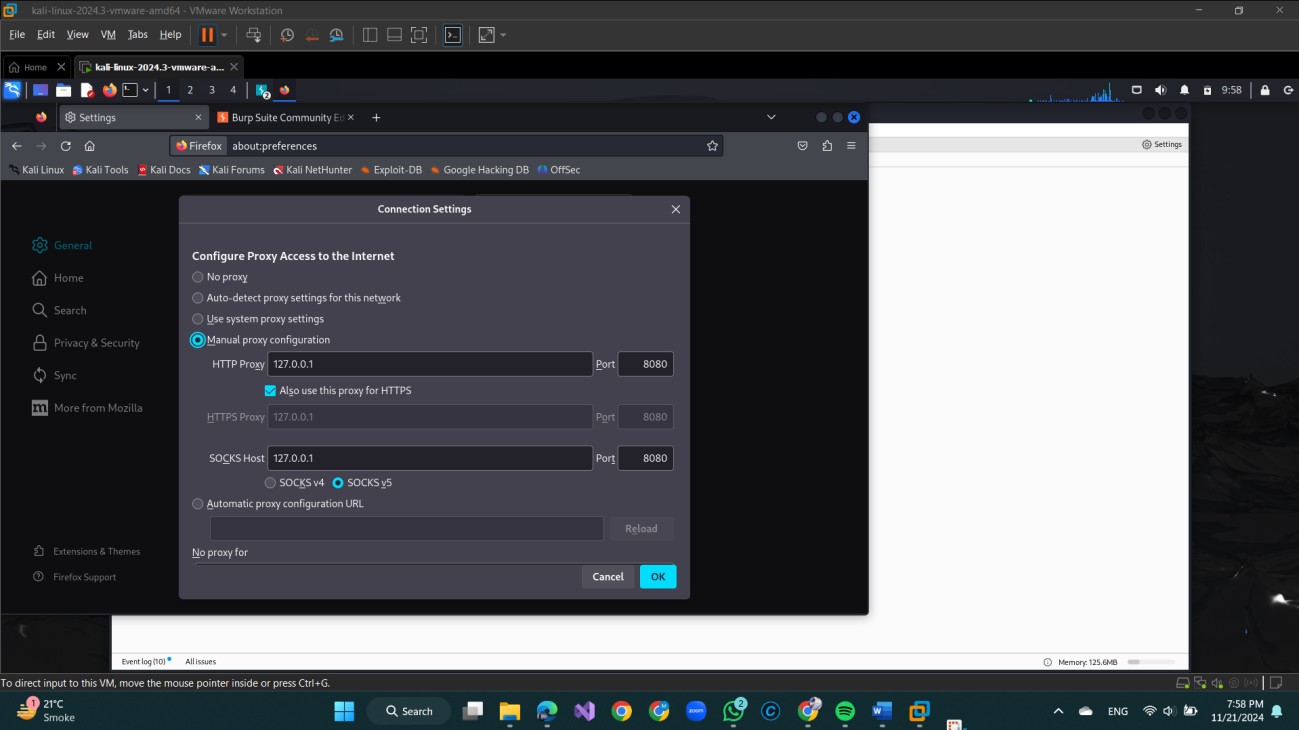
1. **Launch**:

o Launch the community free edition of Burp Suite.



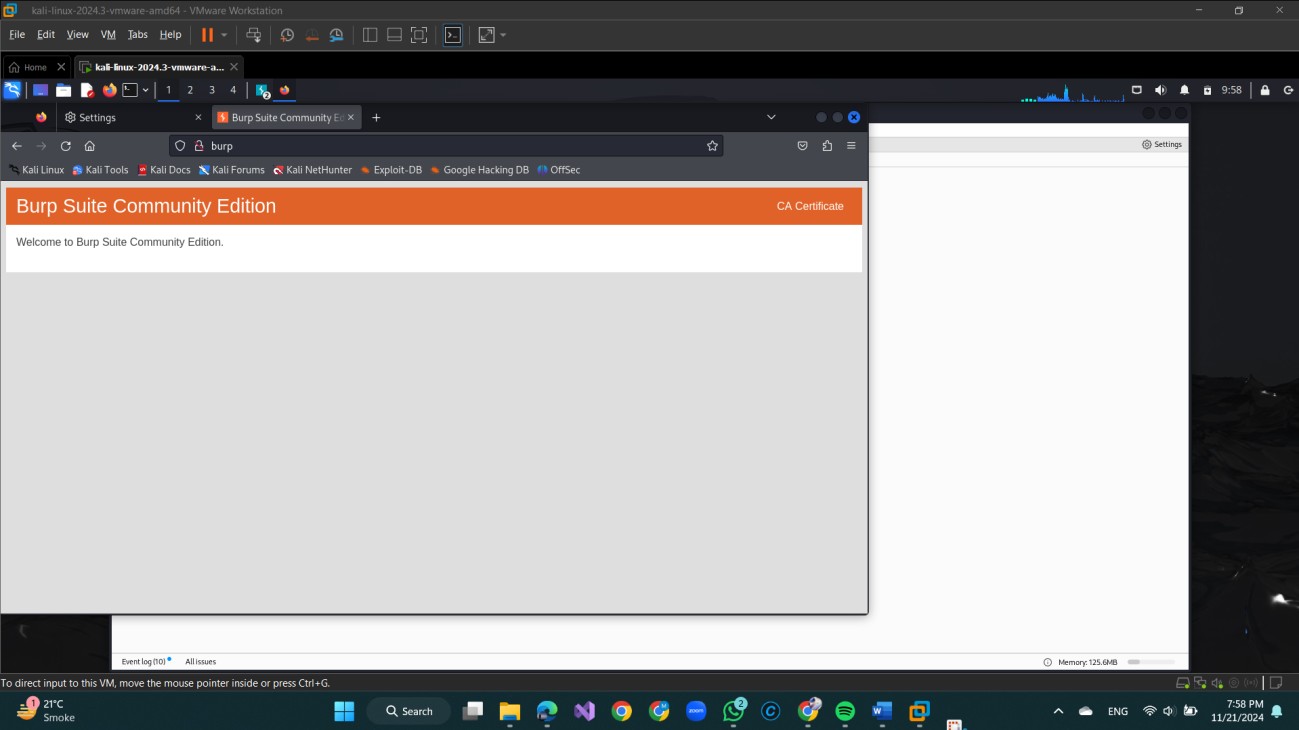
# Configure Your Browser:

* In Burp Suite, go to **Proxy > Intercept** and ensure "Intercept is off."
* **Set your browser to use Burp's proxy** (default: 127.0.0.1:8080).
  + In your browser's **Network Settings**, set manual proxy configuration to 127.0.0.1 for HTTP/HTTPS with port 8080.



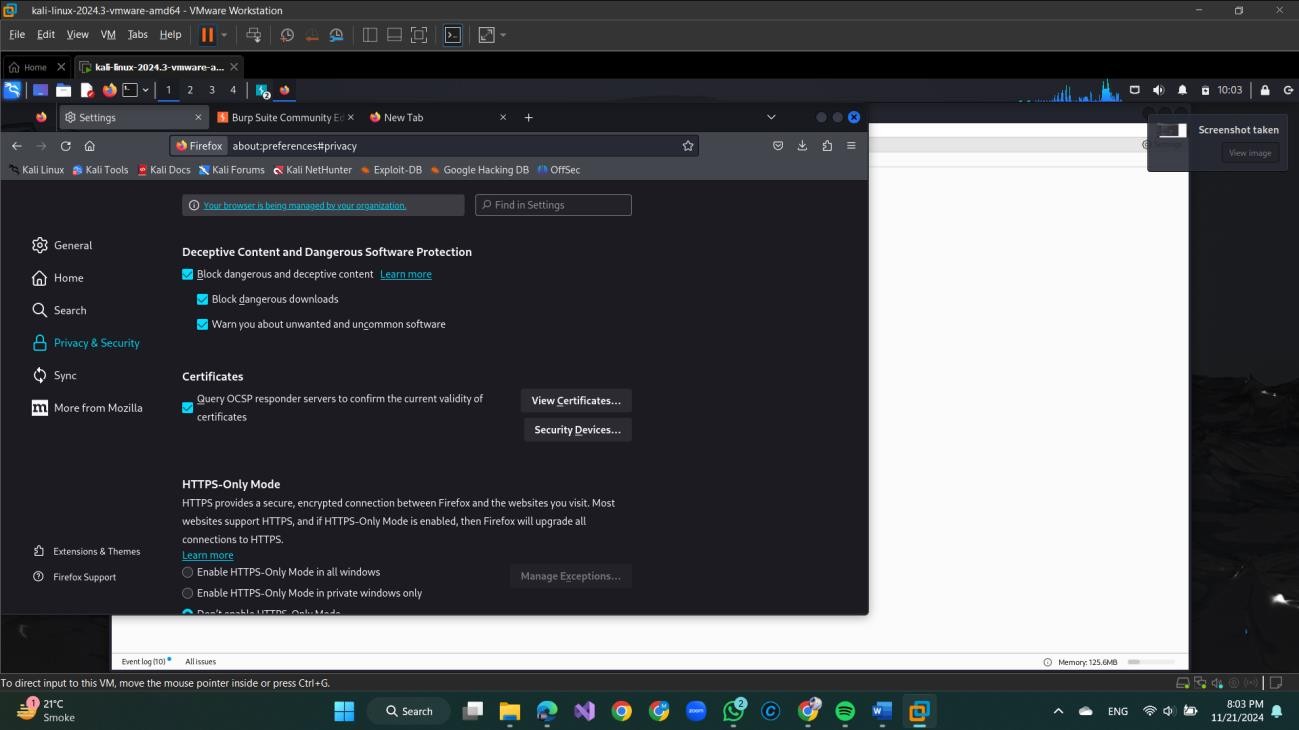
## Install Burp's CA Certificate:

* Open your browser and visit [http://burp](http://burp/) while Burp Suite is running.
* Download and install the CA certificate for analyzing HTTPS traffic.

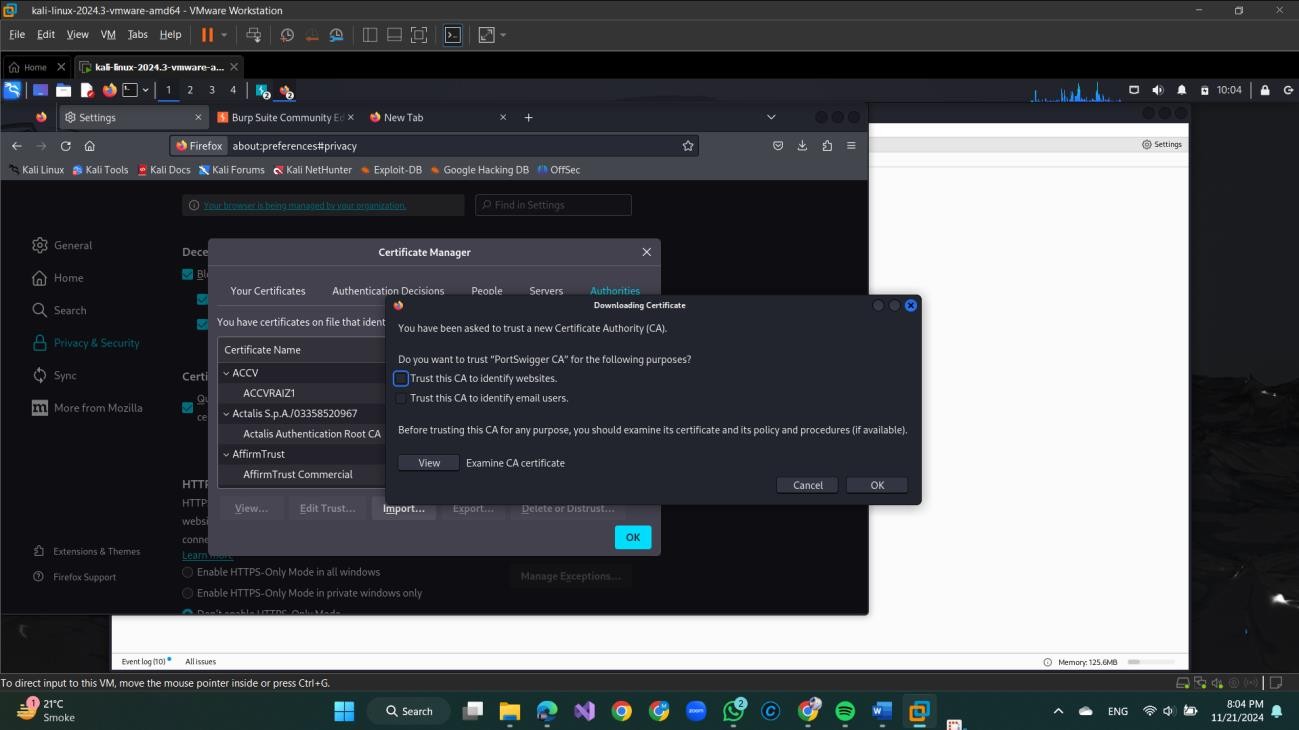


Go to Privacy & Settings

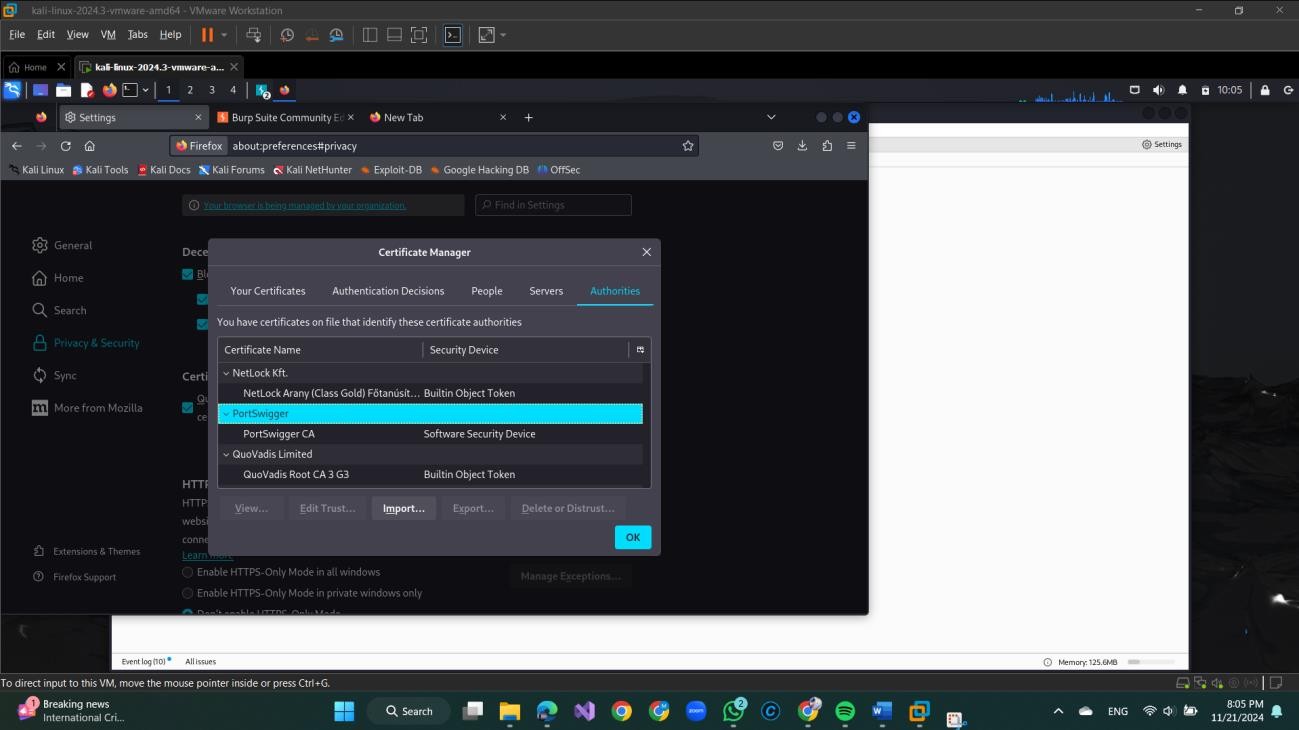
Click on view certificates then click on import.



Check the Trust this CA to identity websites then click on ok.

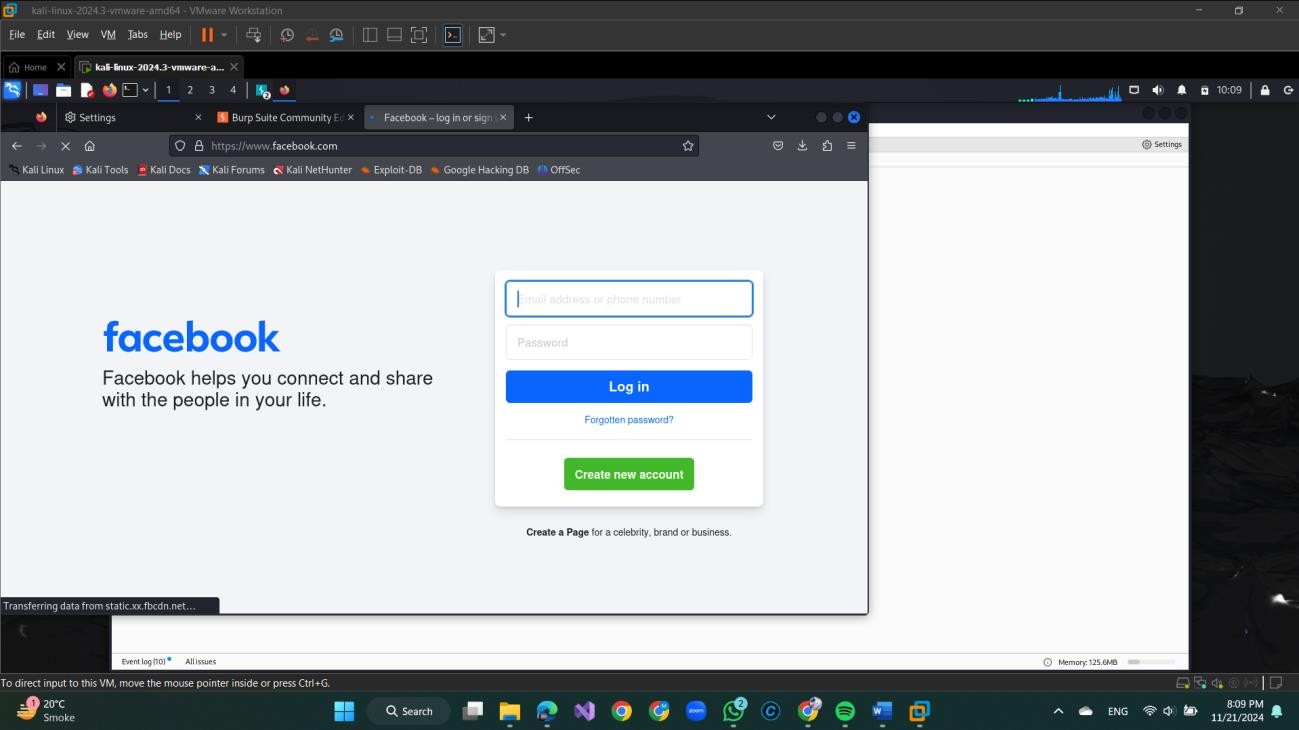


Scroll Down & Click on PortSwigger then click ok



# Step 2: Manual Testing with Proxy:

* Ensure **Intercept** is off in **Proxy**.
* Visit key pages of the web app (e.g., login forms) and enter test data.



* + View requests and responses between your browser and the web app in **Proxy > HTTP history**.

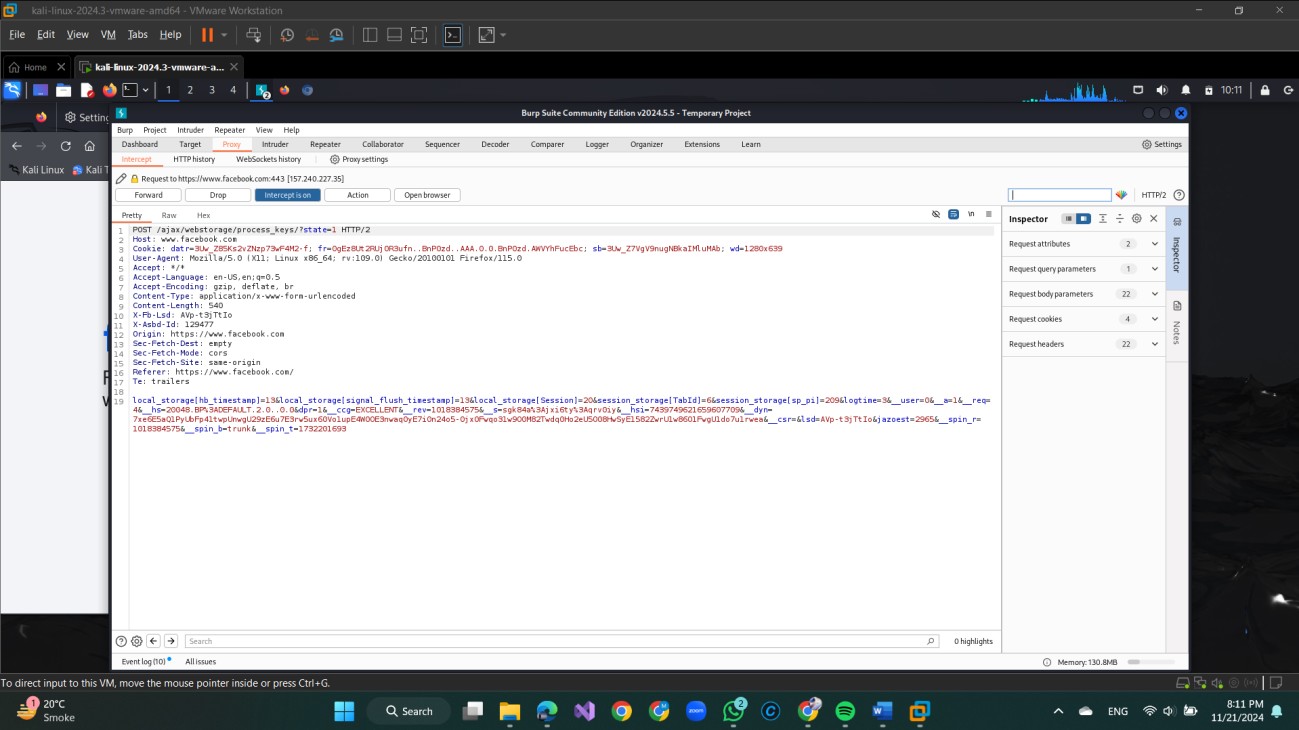
## Automated Scanning:

* + In **Target > Site Map**, right-click the target domain and select **Scan**.
  + Choose specific sections (like login pages or forms) for active scanning, checking for vulnerabilities such as SǪL Injection and Cross-Site Scripting (XSS).

# Step 3: Intercepting and Modifying Requests

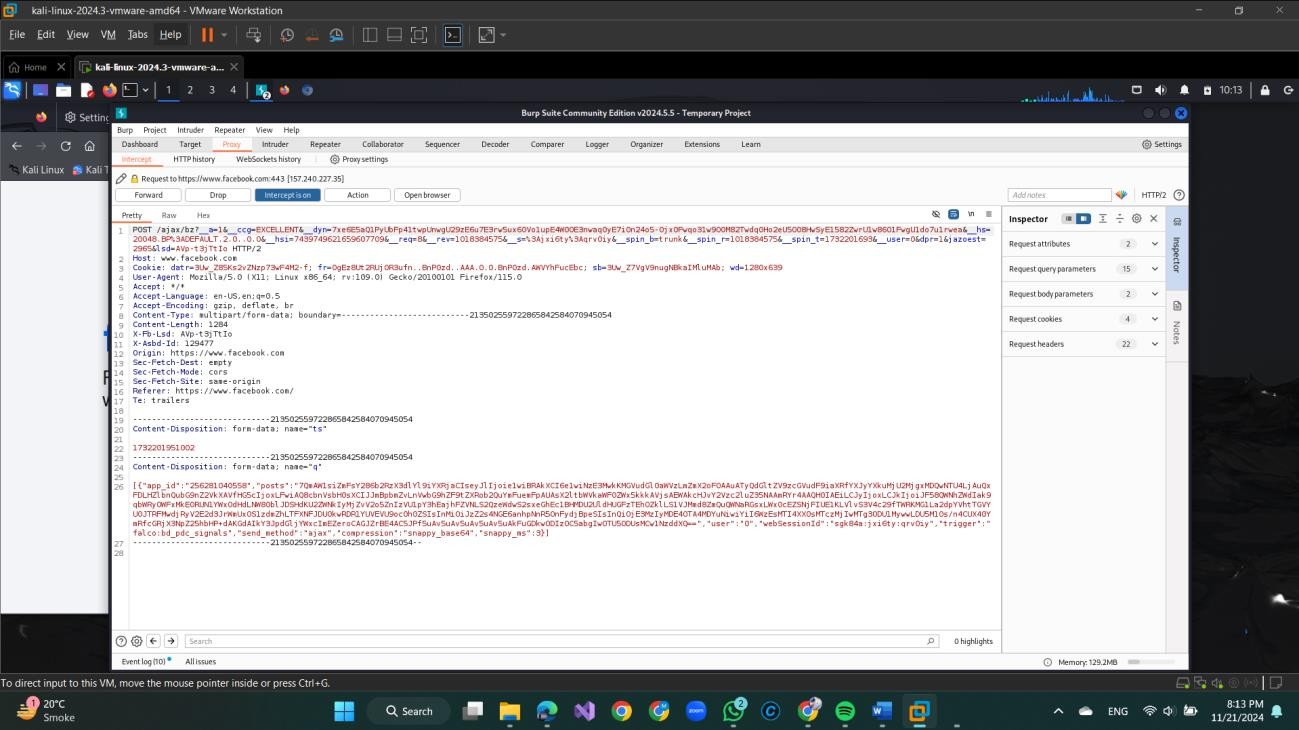
## Intercepting:

* + Turn **Intercept on** in **Proxy > Intercept**.
  + Perform an Action act as the web app (e.g., submit a form). The HTTP request will appear in Burp's **Intercept** tab.



# Modifying Requests:

* You can edit the parameters in the request before forwarding it to the server.
* Example: Modify the parameter to ' OR 1=1-- to test for SǪL Injection and forward the request.
* Observe the response from the server to identify potential vulnerabilities.



# Step 4: Testing for Vulnerabilities

## SǪL Injection:

* + Use **Proxy > Intercept** or **Repeater** (explained later) to modify input fields or URL parameters (e.g., entering ' OR 1=1-- in a login form) and test for SǪL Injection.

## Cross-Site Scripting (XSS):

* + Inject script tags (e.g., <script>alert('XSS')</script>) into input fields.
  + If the script runs in the browser, the application is vulnerable to XSS.

## Cross-Site Request Forgery (CSRF):

* + Use **Repeater** to craft and send requests without proper CSRF tokens to test if the app lacks protection against unauthorized actions

# Conclusion

No vulnerabilities were found after testing the Facebook login page using Burp Suite. This outcome suggests that Facebook has implemented strong security measures to protect against common web vulnerabilities, such as SǪL Injection and cross-site Scripting (XSS).