**SCENARIO**

The web application under scrutiny has implemented an OAuth service, letting users log in via their social media accounts. However, the validation mechanism within the client application seems to be flawed. Due to this imperfection, there's potential for an unauthorized party to access another user's account without needing their password. For testing purposes, our target is to access Carlos's account using the known email: carlos@carlos-montoya.net. We also have credentials to a standard account: wiener:peter.

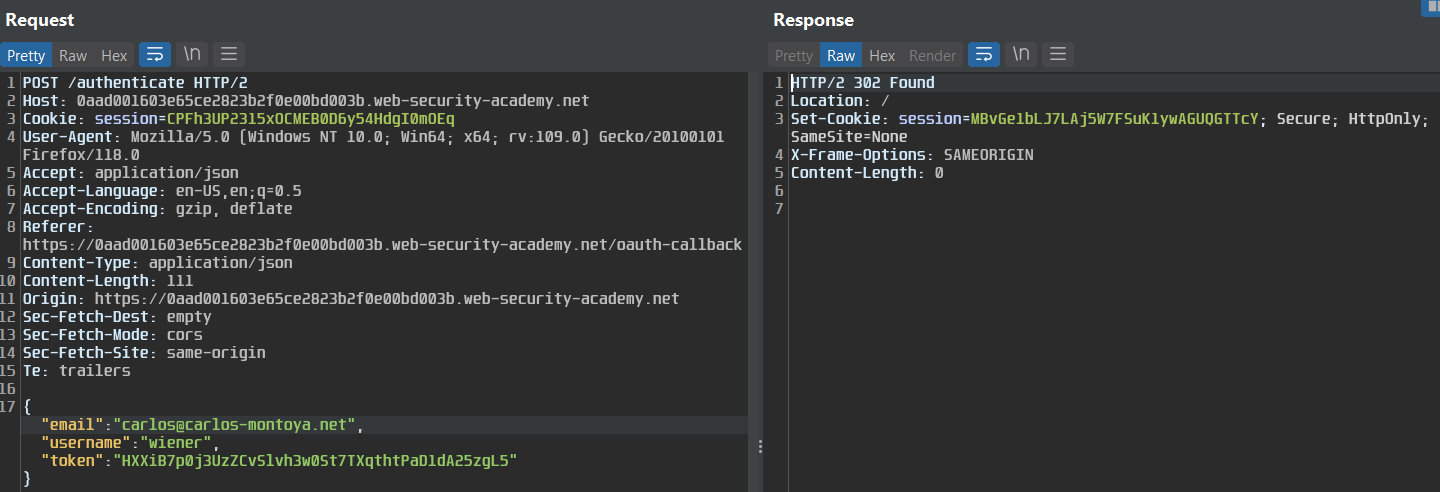
**PROCEDURE**

1. Start by navigating to the "My account" option on the web application, ensuring that all our traffic is proxied through Burp Suite.
2. As we proceed with the OAuth login, we'll be redirected back to the original blog website after authentication.
3. Transition to Burp's "Proxy" > "HTTP history" section. Here, analyze the sequence of requests and responses constituting the OAuth process. The flow commences from the authorization request - seen as GET /auth?client\_id=[...].
4. From our observations, it becomes clear that the client application (which is the blog website in this context) fetches rudimentary user data from the OAuth service. Using this data, the client application attempts a login by dispatching a POST request encompassing this information to its /authenticate endpoint. This also includes the acquired access token.
5. Now, take this POST /authenticate request and forward it to Burp's Repeater tool.
6. Within Repeater, modify the email parameter to represent Carlos's email, which is carlos@carlos-montoya.net, then resend the request. Notably, there is no error upon making this change.
7. Subsequently, perform a right-click action on the modified POST request and pick the "Request in browser" > "In original session" options. This process provides a URL.
8. Using our browser, we then navigate to the given URL.

**PAYLOAD**

Modified POST request within Burp Repeater to change the email parameter to: carlos@carlos-montoya.net

**PROOF OF CONCEPT**

****

**REMEDIATION**

1. Ensure rigorous validation on both the client and server sides, especially when processing data received from third-party services.
2. Regularly audit and test OAuth implementations to confirm they adhere to the best security practices and standards.
3. Use strict OAuth scopes to limit access to the absolute necessary user data.
4. Ensure all tokens, especially the access tokens, are short-lived and employ proper mechanisms for revocation and refreshment.
5. Monitor for suspicious login activities and implement anomaly detection mechanisms to quickly identify and handle unauthorized access attempts.