**SCENARIO**

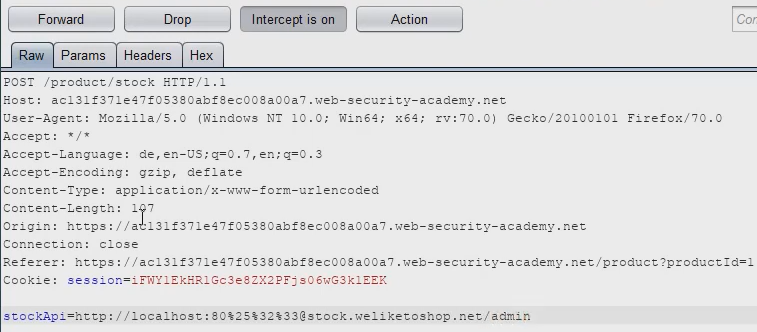
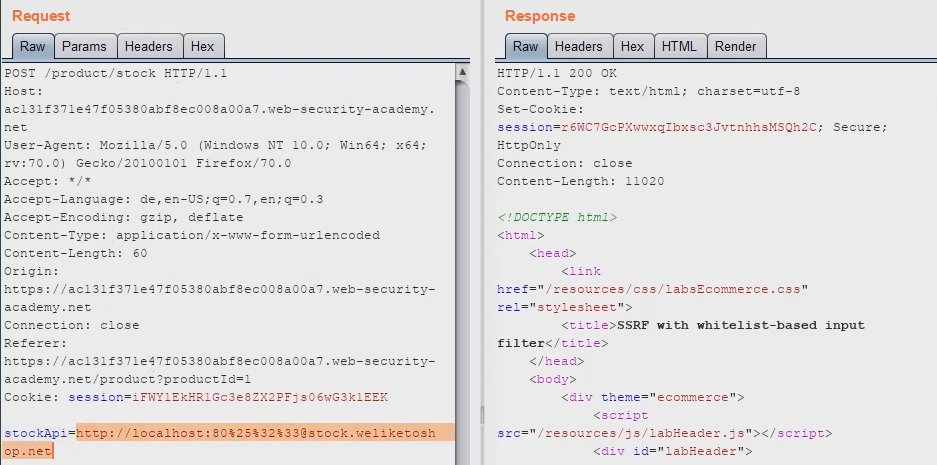
The application's stock check feature is designed to pull data from an internal system. However, upon closer inspection, there seems to be a safeguard against SSRF attacks. This safeguard appears to validate hostnames against a predetermined whitelist. Our aim is to circumvent this defense to gain unauthorized access to the internal admin panel.

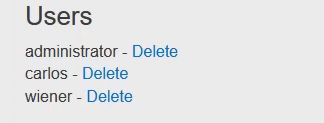
**PROCEDURE**

1. As we begin, we opt for a product and engage its "Check stock" feature. The intercepted request highlights that the stockApi parameter doesn't readily allow external host requests.
2. Further experimentation hints at a quirky behavior: the application's URL parser recognizes and accepts embedded credentials within URLs.
3. To exploit this, we introduce a special character, such as #, within the username of the embedded credentials. Observing the behavior, we notice that this character causes URL parsing anomalies.
4. Leveraging double URL encoding on the special character, we manage to deceive the URL parser and gain access to the restricted admin interface.
5. With this newfound access, we proceed to eliminate the target user from the system.

**PAYLOAD**

Crafted URL utilizing double URL encoding and embedded credentials, for example: http://localhost:80%2523@stock.weliketoshop.net/admin/delete?username=carlos.

******PROOF OF CONCEPT**

****

**REMEDIATION**

1. **Robust URL Parsing:** Deploy advanced URL parsing techniques to catch and reject URLs with embedded credentials.
2. **Block Special Characters:** Disallow certain special characters, like #, in the URLs to prevent potential exploitation.
3. **Regular Audits:** Continuously monitor and review the stock check feature's behavior, ensuring it adheres to expected functionality.
4. **Improved Whitelisting:** Refine the whitelist mechanism to ensure that only truly safe domains and IP addresses are allowed.