SCENARIO

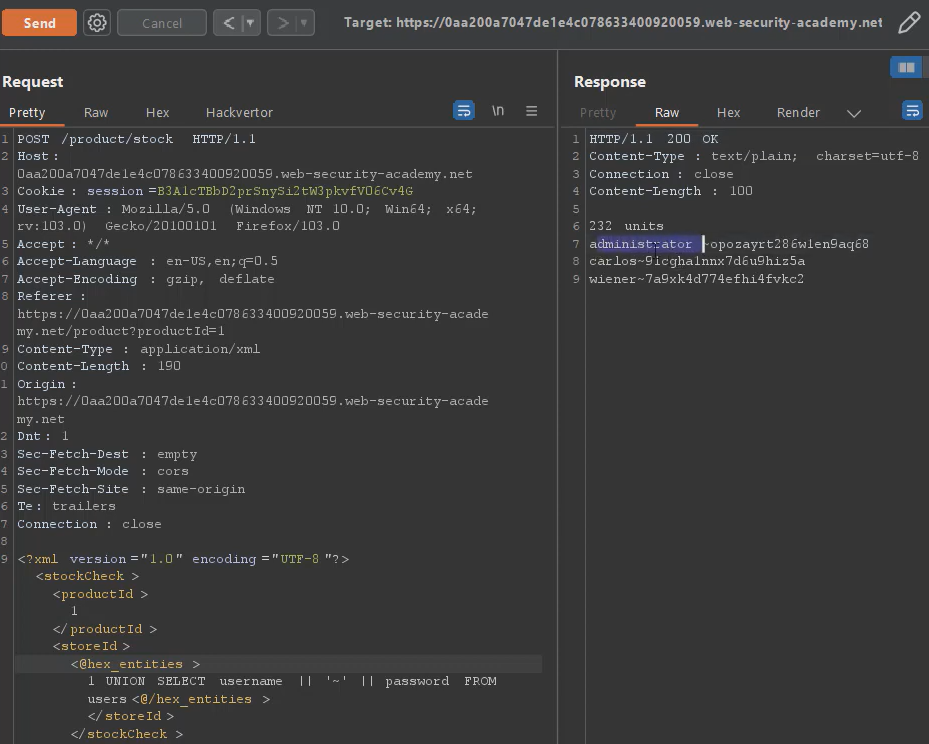
The online store's product search function seems to have a typical SQL injection vulnerability. When testing with specific payloads, the application reveals data from the database.

**PROCEDURE**

1. Head to the product search page and input a generic term, like "shirt".
2. Intercept the search query request with Burp Suite.
3. Inject a SQL payload into the search parameter: shirt' UNION SELECT username, password FROM users--.
4. If the application displays user credentials in the search results, the injection was successful.
5. Refine the query to target the administrator account.
6. Log in using the administrator's password.

**PAYLOAD**

shirt' UNION SELECT username, password FROM users--

**PROOF OF CONCEPT**

**REMEDIATION**

1. Use parameterized SQL queries to avoid the risks associated with SQL injections.
2. Implement strong validation for all input parameters.
3. Limit the capabilities of the SQL account used by the application.
4. Implement a web application firewall (WAF) to monitor for and block potential attacks.
5. Periodically review and update systems to guard against known vulnerabilities.