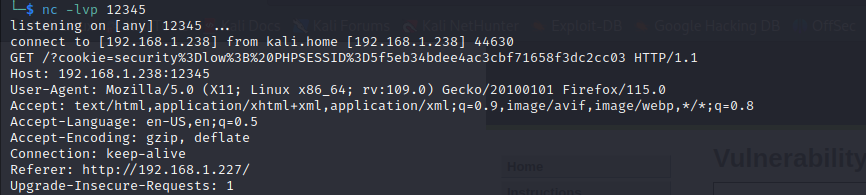
VULNERABILITA XSS STORED E INJECTION SQL BLIND (BLIND)

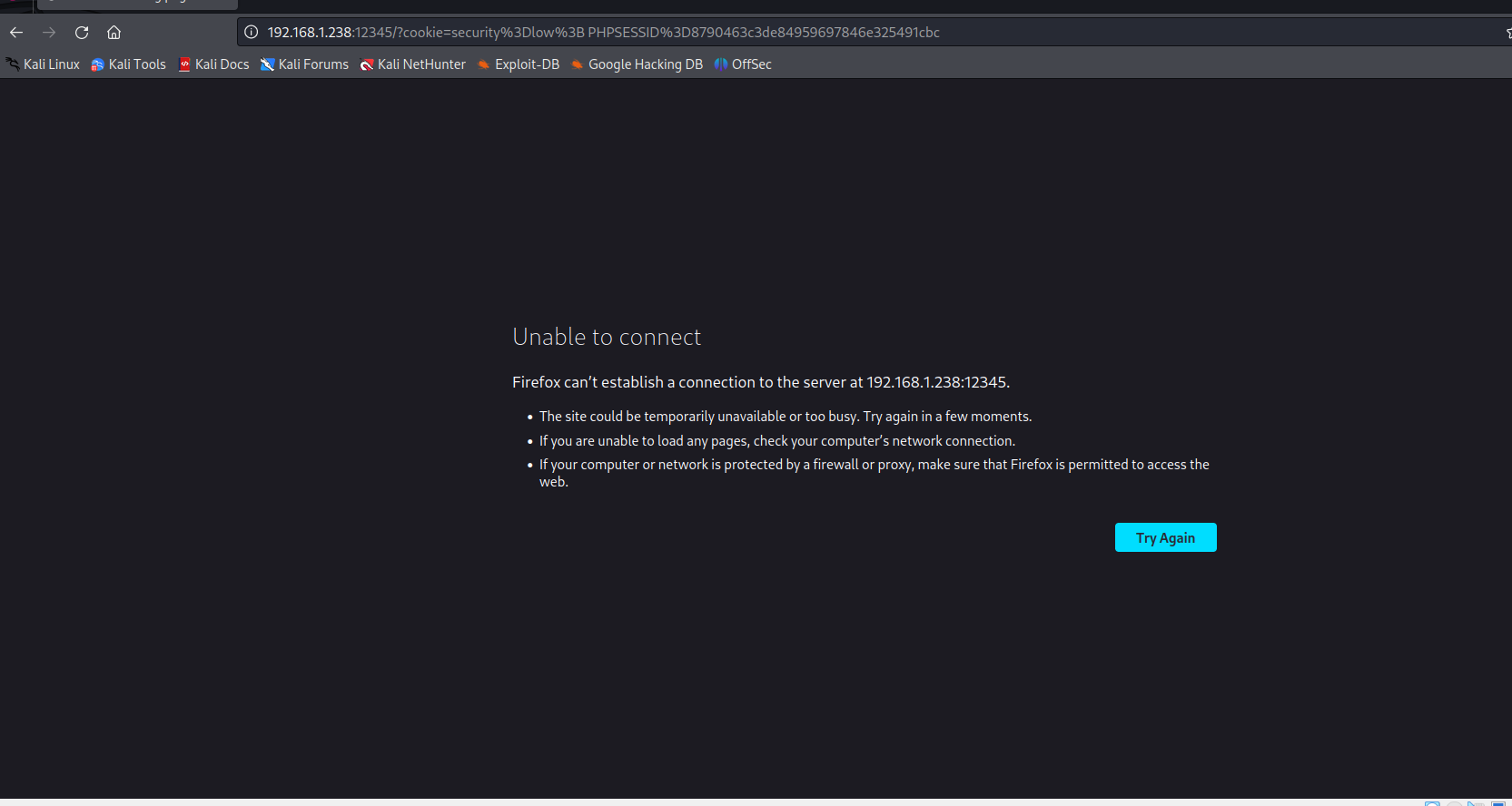
La prima vulnerabilità che abbiamo esaminato è l'XSS stored, un tipo di vulnerabilità molto pericolosa poiché consente l'inserimento di codice malevolo all'interno di una pagina web specifica su un server. Questo permette a un attaccante (in questo caso, chiamato "black hat") di intercettare le connessioni dei client senza che gli utenti ne siano consapevoli. È molto più pericolosa rispetto all'XSS reflected, poiché è più difficile da rilevare una volta che l'attaccante è riuscito a eseguirla. Inoltre, consente all'attaccante di colpire più persone contemporaneamente. Questa vulnerabilità è causata da un errore dovuto alla negligenza del programmatore della pagina, il quale non ha effettuato un corretto filtraggio dell'input fornito dagli utenti.

Per correggere questa vulnerabilità, è fondamentale implementare un adeguato processo di validazione e sanificazione dell'input degli utenti. Inoltre, è consigliabile utilizzare Content Security Policy (CSP) per mitigare il rischio di attacchi XSS. Inoltre, è importante tenere sempre aggiornate le librerie e i framework utilizzati per costruire l'applicazione web al fine di correggere eventuali falle di sicurezza note.

ESEGUZIONE DEL ATTACCO



In questo screnn si puo visualizare come l attacco sia andato a buon fine



questo e il ridirizamento alla “pagina” alla quale il client viene reindirizzato 

In questo specifico caso sono anche andato a modificare l html poche aveva un massimo di caratteri possibili di 50 nella text area ho alzato questo valore

Avendo così la possibilità di inserire lo script malevolo

Mi sono accorto della mancanza dell input non controllato analizzando il back end della pagina . 

INJECTION SQL (BLIND)

L'iniezione SQL (cieca) è un tipo di attacco di iniezione SQL in cui l'attaccante non può vedere direttamente i risultati delle proprie query. Questo perché l'applicazione vulnerabile non restituisce i risultati della query SQL nella risposta, o restituisce la stessa risposta indipendentemente dalla query.

Nonostante ciò, gli aggressori possono comunque sfruttare le vulnerabilità di iniezione SQL cieche per estrarre dati dal database o eseguire comandi SQL arbitrari. Lo fanno inviando query che restituiscono risposte diverse a seconda che la query sia vera o falsa.

Esistono due principali tipi di attacchi di iniezione SQL cieca:

* Iniezione SQL cieca basata sul tempo: in questo tipo di attacco, l'attaccante misura il tempo impiegato dall'applicazione per rispondere alle proprie query. Se la query è vera, la risposta sarà più rapida, perché il database dovrà eseguire meno lavoro.
* Iniezione SQL cieca con risposta condizionale: in questo tipo di attacco, l'attaccante invia query che attivano risposte diverse a seconda che la query sia vera o falsa. Ad esempio, l'attaccante potrebbe inviare una query che chiede se il nome utente è uguale a un valore specifico. Se il nome utente è uguale al valore, l'applicazione potrebbe restituire una pagina diversa o visualizzare un messaggio di errore diverso.

Una volta che l'attaccante ha determinato come distinguere tra risposte vere e false, può iniziare a estrarre dati dal database. Lo fa inviando query che chiedono al database di restituire valori specifici, come il nome utente, la password o il numero di carta di credito di un utente.

Per prevenire gli attacchi di iniezione SQL cieca, i sviluppatori dovrebbero utilizzare query parametrizzate. Le query parametrizzate garantiscono che l'input dell'utente non possa interferire con la struttura della query SQL

