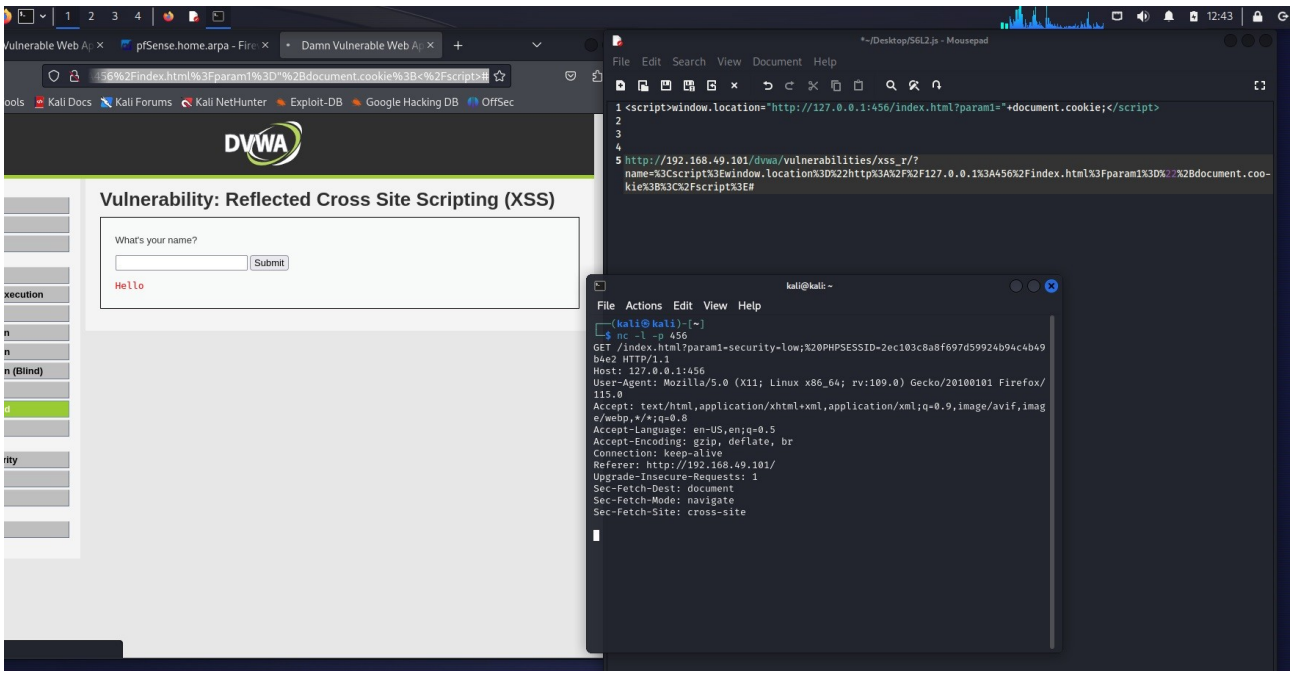
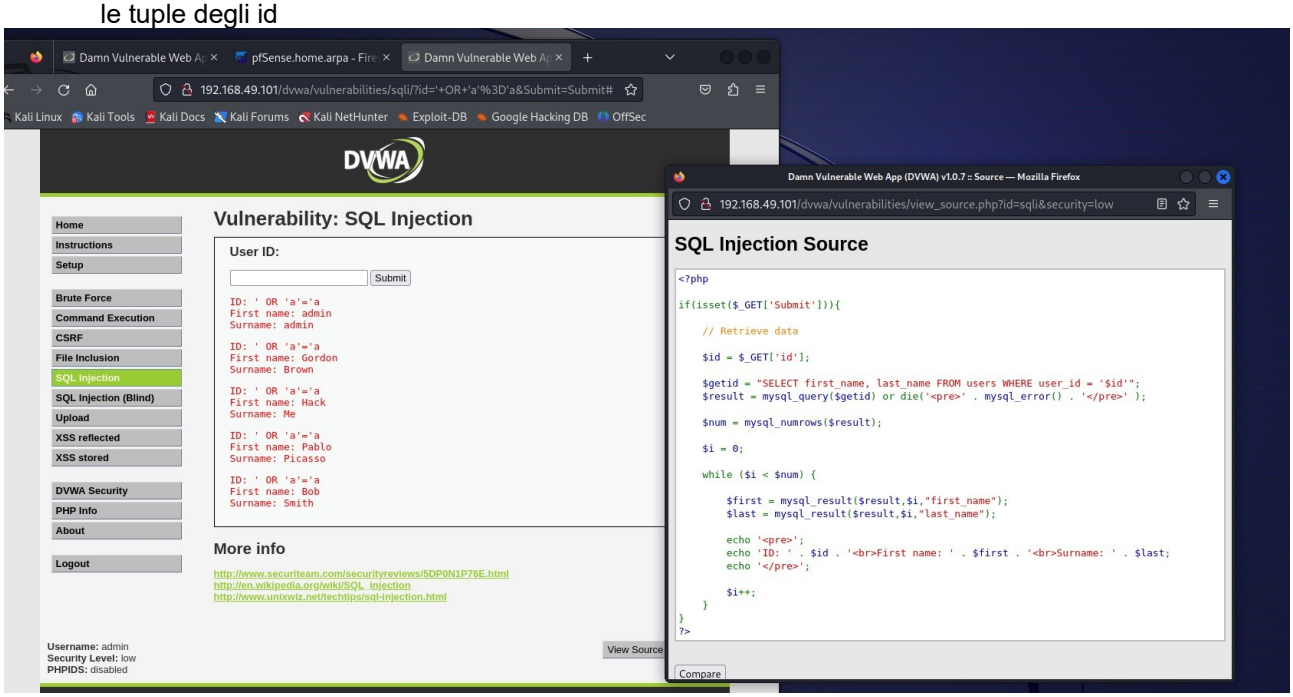


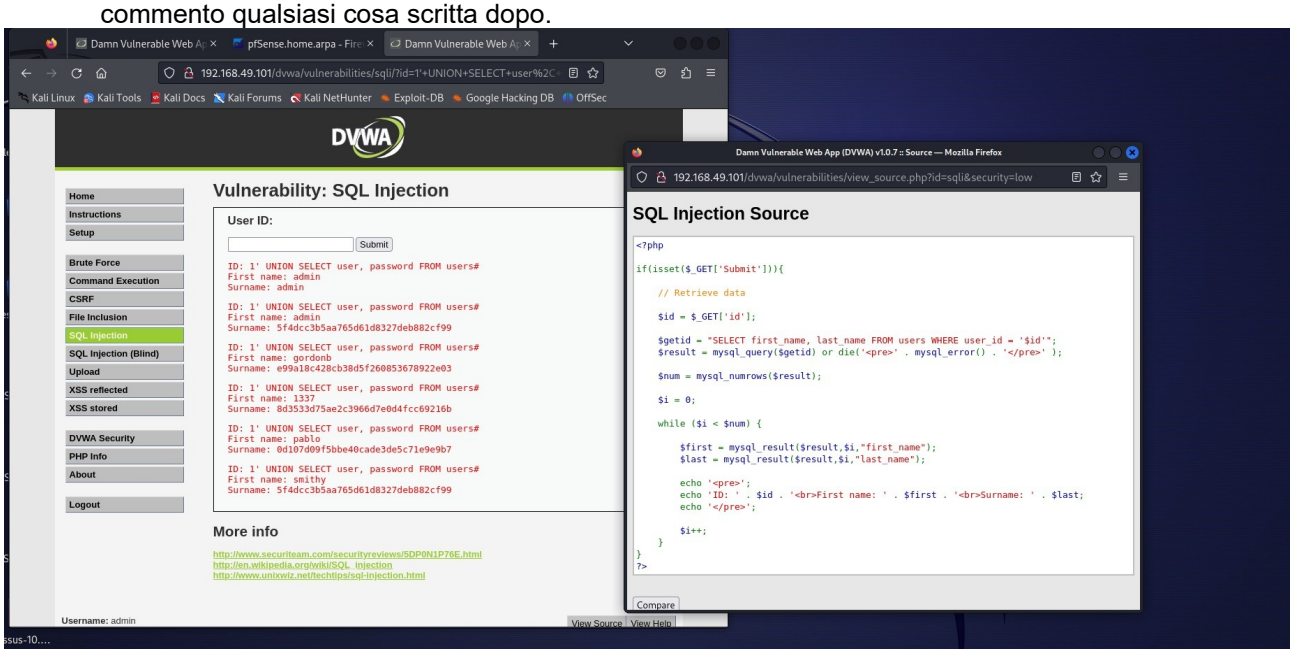
Come si vede dall'immagine, questo e' il risultato di un xss(cross site scripting) reflected; prima di tutto si comincia ad investigare sulla pagina di input per vedere se accetta degli script di tipo html, per esempio inserendo <i>prova si vede se risulta un output interpretato oppure no; in caso di esito positivo si prosegue con l'inserimento di codice javascript, per esempio inserendo <script>alert('prova')</script> vediamo se compare un popup o no; in caso di esito positivo si puo procedere alla compilazione di codice.
Sull'immagine si vede il codice che si utilizzera' scritto su wordpad, in questo caso serve a creare una finestra sull'indirizzo ip della macchina virtuale sulla porta 456 in cui devono comparire i cookie del browser di chi utilizza questa pagina attraverso l'url che andremo a modificare(nel codice, al posto dell'indirizzo ip di solito si inserisce un dominio in modo tale che sia possibile utilizzare una vpn); infatti aprendo il terminale e utilizzando netcat sulla porta in questione si puo vedere il php session id.
A questo punto si inserisce il codice scritto in un sito che lo compila per l'url(ad esempio urlencoder.org) e lo si attacca all'url originale e compare scritto come nell'immagine, evidenziato sotto il codice originale.
Adesso l'url e' pronto per essere inviato alla vittima prescelta alla quale potremo leggere i suoi cookie per poterli utilizzare a nostro piacimento in caso dovesse usarlo. Ovviamente ci sono xss che possono fare cose ben peggiori, dipende dalla complessita' del codice malevolo che viene incollato all'url sorgente. Questo attacco e' di tipo riflesso perche' non viene attuato direttamente da noi ma parte quando e' la vittima stessa che avvia l'url e si “autoattaca” quindi l'attacco viene riflesso su se stessi(cioe' chi usa l'url).
Un xss persistent invece viene incollato direttamente sulla pagina e quindi rimane sempre attivo ed e' piu pericoloso perche' puo fare molte piu vittime in quanto chiunque visita quella pagina viene attaccato.

Anche nel caso dell'SQL injection bisogna prima andare per tentativi; si comincia a scrivere diversi elementi come ad esempio comandi sql, operatori logici, commenti etc... e si vede come risponde la pagina.
In base a come risponde si costruisce la stringa che modifichi la query al database; si puo procedere in 2 modi :

- il primo e' usando l' operatore logico OR che consegna sempre un risultato TRUE in tutti i casi sia presente almeno un risultato TRUE e in questo modo l'intero parametro WHERE(presente nella parte di query non modificabile) viene annullato e di conseguenza compare l'intera tabella con tutte le tuple degli id



- il secondo e' usando il comando UNION, unendo il SELECT di first name e surname dell'id 1(il numero e' l'unica cosa scelta da noi in quanto tutto il codice scritto davanti a 1 e' fisso e non modificabile) con il SELECT di user e password di tutte le tuple prese dalla stessa tabella(questo SELECT e' completamente inventato da noi e modifica il senso di tutta la query non modificabile che c'e' davanti e veniamo a conoscenza degli attributi user e password presenti in tabella, facendo vari tentativi si e' scoperto il nome della tabella); alla fine si inserisce un cancelletto per rendere commento qualsiasi cosa scritta dopo.



Osservando il codice sorgente della pagina si puo notare come l'unica variabile modificabile nella query e' id perche' ha la $ davanti; noi abbiamo fatto le nostre aggiunte sostituendo quel singolo elemento al fine di stravolgere totalmente il risultato della query