

Test con SQL Injection

Consegna:

Dopo aver seguito le videolezioni sulle tecniche di SQL injection, verificare su una propria applicazione (**niente prove su applicazioni e siti web altrui**) che richiede inserimento dati quanto questa sia eventualmente vulnerabile e, di seguito, riportare in un documento i risultati ottenuti durante le fasi di test (oscurare informazioni relative ad account e dati personali) e le istruzioni aggiunte in seguito per proteggere l'applicazione da tali attacchi.

In questo test andrò ad utilizzare un mio form di login.

Attacco SQL Injection

La query utilizzata per il login è la seguente:

```
$query = "SELECT * FROM users WHERE email = '$username' AND password = '$password'";
```

Per poter eseguire un attacco SQL Injection sarà necessario andare ad inserire nei campi di nome utente e password questa stringa:

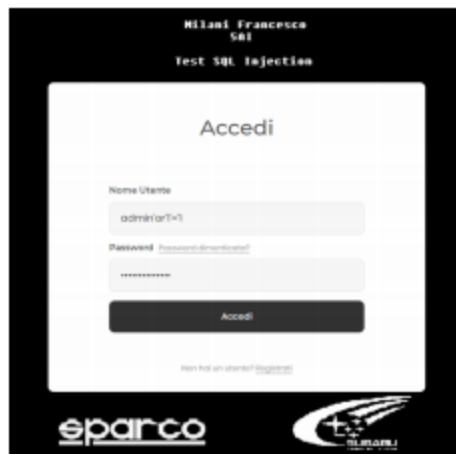
```
'OR '1'='1
```

In questo modo, la query risultante sarà:

```
SELECT * FROM users WHERE email = '' OR '1'='1' AND passowr = '' OR '1'='1';
```

Come si può notare, attraverso il primo apice andremo a chiudere il testo dove dovrebbe esserci scritto l'username, andando poi ad inserire un'operazione OR che ci restituirà TRUE, data l'uguaglianza 1=1 sempre vera. La stessa cosa avverrà anche nel campo della password.

In questo modo il database restituirà tutte le righe della tabella, quindi dando una risposta positiva allo script di login che ci lascerà perciò accedere al sito.



Questo determina la non sicurezza del nostro login, data la facilità di accesso tramite SQL Injection.

Metodi di difesa

Funzione mysql_real_escape_string

Per poterci difendere possiamo inserire una funzione mysql_real_escape, scrivendola nel nostro codice in questo modo:

```
$username = $_POST["username"];  
$username = mysql_real_escape_string($username);  
$password = $_POST["password"];  
$password = mysql_real_escape_string($password);
```

In questo modo andremo a rimuovere i caratteri speciali, quindi dalla stringa

```
' OR '1'='1'
```

verranno rimossi gli apici, quindi sarà quasi impossibile un attacco attraverso apici o caratteri speciali.

Nome utente o password errata

Nome Utente

Password

[Registrali](#)

Non hai un utente? [Registrali ora](#)

Nel caso la query fosse di questo tipo:

```
$query = "SELECT * FROM users WHERE email = $username AND password = $password"
```

si può notare come non ci sia il carattere di terminazione ' , quindi la query rimane aperta, non viene chiusa.

Per poter eseguire l'SQL Injection si può procedere all'inserimento di `1 OR 1=1` nei campi di username e password.

In questo caso il risultato sarà quindi positivo, dato che la query risultante sarà

```
$query = "SELECT * FROM users WHERE 1 OR 1=1 AND password = 1 OR 1=1"
```

Accedi

Nome Utente

Password Password dimenticata?

[Accedi](#)

Non hai un utente? [Registrali](#)

Sei loggato !

Quindi il nostro login non è ancora sicuro, dato l'avvenuto accesso attraverso SQL Injection.

Funzione is numeric

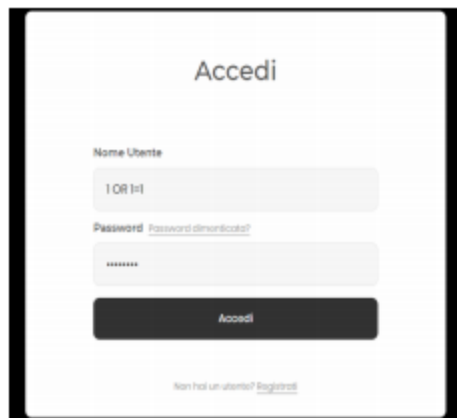
Attraverso la funzione PHP

```
is_numeric()
```

sarà eseguita una verifica sull'input, e se questo corrisponderà ad un tipo numerico non verrà eseguita la query.

La sintassi del controllo sarà quindi

```
if(is_numeric($username))  
{  
    //Errore  
}
```



The screenshot shows a login form with the title "Accedi". It contains two input fields: "Nome Utente" with the placeholder text "I OR I!" and "Password" with the placeholder text "Password dimenticata?". Below the fields is a dark button labeled "Accedi". At the bottom, there is a link that says "Non hai un utente? Registrati".



The screenshot shows the same login form as the previous one, but with an error message "Nome utente o password errata" displayed at the top. The input fields are empty, and the dark button is now labeled "Registrali". The link at the bottom remains "Non hai un utente? Registrati ora".

Questo rende impossibile l'accesso attraverso SQL Injection, in quanto non sarà possibile né inserire caratteri speciali attraverso la funzione vista in precedenza, né inserire un input numerico.