CTF常用工具之汇总

原创

Mr白猫 于 2018-03-22 15:36:58 发布 4417 收藏 20
分类专栏： CTF学习笔记 文章标签： CTF Python 工具

本文链接：https://blog.csdn.net/qq_37185297/article/details/79655010

CTF常用工具之汇总

前言：因为最近在玩一下CTF(sql注入是真的不会玩呀？(T_T) (T_T))所以经常会用到一些像什么进制转换呀？base64 32 16编码解码呀？当然这些工具网上都有！但是每次都打开网页 总是感觉很麻烦所以就萌生了一个想法能不自己写一个脚本把这些功能集合在一起！就像kali里面的工具一样！这样的话想用的时候岂不是很方便？所以从昨天下午开始写到今天上午中午终于把这个工具脚本写完了！我们先看一下预览图

当然这里只是集成我自己常用的一些功能！如果你们需要扩容也是非常容易的，

(1) 首先说一下这个脚本需要的包
import hashlib
import base64
from urllib import quote,unquote
import argparse
此工具需要运行在python3的环境下
(2) 下面就是代码部分，里面有详细的注释，如果还是看不懂代码里面有我的QQ

-- coding:utf-8 --

```
import hashlib
import base64
from urllib import quote,unquote
import argparse

“””

    名字：CTF之常用工具汇总

    作者：白猫

    时间：2018-3-22

    QQ ：1058763824

“””

def menu():
```

```
usage = """-m MD5 encryption
   -s      SH1 encryption
   -h      Show help information
   -b64    Base64 encode
   -b32    Base32 encode
   -b16    Base16 encode
   -db64   Base64 decode
   -db32   Base32 decode
   -db16   Base16 decode
   -urlen  URL encode
   -urlde  URL decode
   -bin    Binary To Decimal
   -octal  Octal  to Decimal
   -hex    Hexadecimal to Decimal
   -dbin   Decimal To Binary
   -doctal Decimal to Octal
   -dhex   Decimal to Hexadecimal
   -ord    Letter To ASCII           Example  -ord asdfasfa      -ord='dfafs afasfa  asfasf'
   -chr    ASCII  To Letters         Example  -chr 105           -chr = '102 258 654'


"""

#在使用ord 和chr命令的时候要注意如果输入的字符和数字不包含空格则直接实用例子前面的命令如果包含空格则使用后面的命令

parser = argparse.ArgumentParser()

parser.add_argument('-m',dest='md',help='MD5 encryption')

parser.add_argument('-s', dest='sh', help='SH1 encryption')

parser.add_argument('--h',action="store_true",help='Show help information')

parser.add_argument('-b64', dest='b64', help='Base64 encode')

parser.add_argument('-b32', dest='b32', help='Base32 encode')

parser.add_argument('-b16', dest='b16', help='Base16 encode')

parser.add_argument('-db64', dest='db64', help='Base64 decode')

parser.add_argument('-db32', dest='db32', help='Base32 decode')

parser.add_argument('-db16', dest='db16', help='Base16 decode')

parser.add_argument('-urlen', dest='urlen', help='URL encode')

parser.add_argument('-urlde', dest='urlde', help='URL decode')

parser.add_argument('-bin', dest='bin', help='Binary To Decimal')

parser.add_argument('-octal', dest='octal', help='Octal  to Decimal')

parser.add_argument('-hex', dest='hex', help='Hexadecimal to Decimal')

parser.add_argument('-dbin', dest='dbin', help='Decimal To Binary ')

parser.add_argument('-doctal', dest='doctal', help='Decimal to Octal ')

parser.add_argument('-dhex', dest='dhex', help='Decimal to Hexadecimal')
```

```
parser.add_argument('-dhex', dest='dhex', help='Decimal to Hexadecimal')

parser.add_argument('-ord', dest='ord', help="Letter To ASCII              Example  -ord aaaaaa  , -ord=\"aaa a
aa\"")

parser.add_argument('-chr', dest='chr', help="ASCII  To Letter             Example  -chr 105     ,  -chr = \"10
1 101\" ")

options = parser.parse_args()

if options.md:

    s = options.md

    md5(s)

elif options.sh:

    s = options.sh

    sh1(s)

elif options.b64:

    s = options.b64

    stringToB64(s)

elif options.b32:

    s = options.b32

    stringToB32(s)

elif options.b16:

    s = options.b16

    stringToB16(s)

elif options.db64:

    s = options.db64

    b64ToString(s)

elif options.db32:

    s = options.db32

    b32ToString(s)

elif options.db16:

    s = options.db16

    b16ToString(s)

elif options.urlen:
```

```
    s = options.urlen
    urlEncode(s)
elif options.urlde:
    s = options.urlde
    urlDecode(s)
elif options.bin:
    s = options.bin
    binToDec(s)
elif options.octal:
    s = options.octal
    octToDec(s)
elif options.hex:
    s = options.hex
    hexToDec(s)
elif options.dbin:
    s = options.dbin
    decToBin(s)
elif options.doctal:
    s = options.doctal
    decToOct(s)
elif options.dhex:
    s = options.dhex
    decToHex(s)
elif options.doctal:
    s = options.doctal
    decToOct(s)
elif options.dhex:
    s = options.dhex
    decToHex(s)
elif options.ord:
```

```
    s = options.ord
    lettToASCII(s)
elif options.chr:
    s = options.chr
    asciiToLett(s)


else:
    helpInfo()
```

def helpInfo():

```
print("""
```

-m MD5 encryption
-s SH1 encryption
–h Show help information
-b64 Base64 encode
-b32 Base32 encode
-b16 Base16 encode
-db64 Base64 decode
-db32 Base32 decode
-db16 Base16 decode
-urlen URL encode
-urlde URL decode
-bin Binary To Decimal
-octal Octal Decimal to Decimal
-hex Hexadecimal to Decimal
-dbin Decimal To Binary
-doctal Decimal to Octal
-dhex Decimal to Hexadecimal
-ord Letter To ASCII attention Example -ord asdfasfa -ord=”dfafs afasfa asfasf”
-chr ASCII To Letters Example -chr 105 -chr = “102 258 654”
“”“)

进行MD5加密

def md5(s):

```
original = s

md  = hashlib.md5()

s = s.encode(encoding = 'utf-8')

md.update(s)

print('Original:'+original)

print('Md5 Encryption:'+md.hexdigest())
```

进行sh1加密

def sh1(s):

```
original = s

sh = hashlib.sha1()

s = s.encode(encoding='utf-8')

print('Original:' + original)

print('SH1 Encryption:' + sh.hexdigest())
```

将字符串转换为base64编码格式

def stringToB64(s):

```
encode = base64.b64encode(s)

print('Original:' + s)

print('Base64 encode:' + encode)
```

将base64编码格式转化为正常的字符类型

def b64ToString(s):

```
decode = base64.b64decode(s)

print('Base64:' + s)

print('Base64 decode:' + decode)
```

将字符串转为b32编码格式

def stringToB32(s):

```
encode = base64.b32encode(s)

print('Original:' + s)

print('Base32 encode:' + encode)
```

将base32编码格式转化为正常的字符类型

def b32ToString(s):

```
decode = base64.b32decode(s)

print('Base32:' + s)

print('Base32 decode:' + decode)
```

将字符串转为base16编码格式

def stringToB16(s):

```
encode = base64.b16encode(s)

print('Original:' + s)

print('Base16 encode:' + encode)
```

将base16编码格式转化为正常的字符类型

def b16ToString(s):

```
decode = base64.b16decode(s)

print('Base16:' + s)

print('Base16 decode:' + decode)
```

进行url编码

def urlEncode(s):

```
encode = quote(s)

print('Original:' + s)

print('URL encode:' + encode)
```

进行url编码

def urlDecode(s):

```
decode = unquote(s)

print('URL encode:' + s)

print('URL decode:' + decode)
```

将二进制转化为十进制

def binToDec(s):

```
result = int(s,2)

print('Binary :'+str(s))

print('Decimal :' + str(result))
```

将八进制转化为十进制

def octToDec(s):

```
result = int(s, 8)

print('Octal :' + str(s))

print('Decimal :' + str(result))
```

将十六进制转化为十进制

def hexToDec(s):

```
result = int(s, 16)

print('Hex :' + str(s))

print('Decimal :' + str(result))
```

将十进制转化为二进制

def decToBin(s):

```
s = int(s)

result = bin(s)

print('Decimal:' + str(s))

print('Binary:' + str(result))
```

将十进制转化为八进制

def decToOct(s):

```
s = int(s)

result = oct(s)

print('Decimal :' + str(s))

print('Octal :' + str(result))
```

将十进制转化为十六进制

def decToHex(s):

```
s = int(s)

result = hex(s)

print('Decimal :' + str(s))

print('Hex :' + str(result))
```

将字母转化为对应的ASCII

def lettToASCII(s):

print('Letters:'+s)

result = ''

for i in s:

```
    result = result+str(ord(i)) + ' '
```

print('ASCII :'+result)

将ASCII转化为对应的字母以及字符

def asciiToLett(s):

list=s.split(' ')

result = ''

print('ASCII :'+s)

for i in list:

```
    i = int(i)

    result =result + chr(i)
```

print('Letters :'+result)

if **name**=='**main**':

```
menu()
```

(3) 下面附上工具的调试情况图

后记这个工具的可扩展新还是非常强的！你们完全可以根据自己就实际需要来扩展。哎，要去看关于sql注入的文章了/(ㄒoㄒ)/~~