CTF HACKON 2021

REVERSING – ALIENS

https://www.hackon.es/
@HackOnURJC

ENUNCIADO

Estamos en el espacio y hemos visto una nave. Ha contactado con nosotros y nos ha mandado el siguiente archivo. Quizás ahí se encuentre la contraseña secreta para que nos dejen acceder a su nave…

FLAG

HackOn{Sp4c3_1nV4d3r5}

SOLUCIÓN

En el terminal ejecutamos file y vemos que no nos devuelve nada interesante más que es texto.

```
└─$ file aliens
aliens: ASCII text, with very long lines, with no line terminators
```

Abrimos el archivo y vemos que es hexadecimal, por lo que desde consola ejecutamos un comando que nos permita pasar de hexadecimal a ASCII (o con cualquier herramienta online) y lo guardamos en un nuevo archivo.

```
└─$ perl -ne 's/([0-9a-f]{2})/print chr hex $1/gie' < aliens >> aliens.txt
```

```
└─$ cat aliens.txt
def Chri1(O00000000000000O0):
    if len(O00000000000000O0) != 14:
        print("@Alien@gena detectado!")
        return
    else:
        if fuscat1(O00000000000000O0):
            print("@Bienvenido a la nave!")
            return
        else :
            print("@Alien@gena detectado!")
            return
def fuscat1(OOOOOOOOOOOOOOOO):
    O0000000000000000 = [ord(O0000000000000000) for O0000000000000000 in OOOOOOOOOOOOOOOO]
    O0000000000000000 = [112,83,99,52,95,51,110,49,52,86,51,100,53,114]
    O0000000000000000 = True
    O0000000000000000 = 0
    O0000000000000000 = 1
    while O00000000000000O0 and O0000000000000000 < 14:
        if O0000000000000000[O0000000000000000] != O0000000000000000[O0000000000000000]:
            O00000000000000O0 = False
        if O0000000000000000[O0000000000000000] == O0000000000000000[O0000000000000000]:
            O00000000000000O0 = O00000000000000O0 + 2
            O0000000000000000 = O0000000000000000 + 2
        else:
            O00000000000000O0 = False
    return O00000000000000O0

print("Bienvenido a la nave, indique su contrase@a de acceso: ")
Niswanobi = str(input())
Chri1(Niswanobi)
```

Se puede observar que hemos recuperado un código en Python, por lo que lo guardamos como fichero *.py*. Lo ejecutamos y vemos que se nos pide una contraseña, introducimos una al azar y vemos que el programa termina.

```
└─$ python3 aliens.py
Bienvenido a la nave, indique su contraseña de acceso:
dskgf
¡Alienígena detectado!
```

Ahora tenemos que analizar el código. Aunque se puede ver desde la consola de comandos o abriéndolo con un editor de texto, es mejor hacerlo en un editor de códigos para que se resalte todo bien y se pueda analizar mejor.

Primero, podemos ver que nuestro input es lo que se pasa como parámetro a la función Chri1:

```
Niswanobi = str(input())
Chri1(Niswanobi)
```

Podemos cambiar el nombre de esta variable a miClave para saber qué es en todo momento.

Si examinamos la función Chri1 vemos que nuestro input toma un nuevo nombre, compuesto por 0's y O's. Lo primero que hace es comparar si la longitud de nuestro input es igual a 14 y, si no lo es, nos echa del programa. Por lo tanto, ya sabemos que nuestra clave tiene que tener 14 caracteres. También renombramos esta variable.

```
def Chri1(O0000000000000000):
    if len(O0000000000000000) != 14:
        print("\xa1Alien\xedgena detectado!")
        return
    else:
        if fuscat1(O0000000000000000):
            print("\xa1Bienvenido a la nave!")
            return
        else :
            print("\xa1Alien\xedgena detectado!")
            return
def Chri1(miInput):
    if len(miInput) != 14:
        print("\xa1Alien\xedgena detectado!")
        return
    else:
        if fuscat1(miInput):
            print("\xa1Bienvenido a la nave!")
            return
        else :
            print("\xa1Alien\xedgena detectado!")
            return
```

Si nuestra contraseña es de longitud 14, pasamos al else, donde llama a una función: fuscat1.

Marcamos el parámetro que se ha pasado y vemos que se usa en la función *ord*.

```
def fuscat1(O0000000000000000):
    O0000000000000000 = [ord(O0000000000000000) for O0000000000000000 in O0000000000000000]
    O0000000000000000 = [112,83,99,52,95,51,110,49,52,86,51,100,53,114]
```

Si buscamos qué hace, descubrimos que convierte el parámetro a Unicode.

Python ord()

The ord() function returns an integer representing the Unicode character.

Por lo tanto, ahí está convirtiendo nuestro input entero a Unicode y lo almacena en una variable, que renombramos también, junto con sus apariciones.

```
miClaveUnicode = [ord(c) for c in password]
```

En la siguiente línea, podemos deducir que es la clave que buscamos, por lo que renombramos esas apariciones a claveAcceso.

```
O0000000000000000 = [112,83,99,52,95,51,110,49,52,86,51,100,53,114]
def fuscat1(password):
    miClaveUnicode = [ord(O0000000000000000) for O0000000000000000 in password]
    claveAcceso = [112,83,99,52,95,51,110,49,52,86,51,100,53,114]
    O0000000000000000 = True
    O0000000000000000 = 0
    O0000000000000000 = 1
    while O0000000000000000 and O0000000000000000 < 14:
        if claveAcceso[O0000000000000000] != miClaveUnicode[O0000000000000000]:
            O0000000000000000 = False
        if claveAcceso[O0000000000000000] == miClaveUnicode[O0000000000000000]:
            O0000000000000000 = O0000000000000000 + 2
            O0000000000000000 = O0000000000000000 + 2
        else:
            O0000000000000000 = False
    return O0000000000000000
```

Podemos observar también que hay una variable booleana y dos auxiliares, que marcándolas vemos que se utilizan en los ifs como iteradores. Renombramos todas las variables para entender todo mejor.

```
def fuscat1(password):
    miClaveUnicode = [ord(O0000000000000000) for O0000000000000000 in password]
    claveAcceso = [112,83,99,52,95,51,110,49,52,86,51,100,53,114]
    boolean = True
    x = 0
    y = 1
    while boolean and x < 14:
        if claveAcceso[x] != miClaveUnicode[y]:
            boolean = False
        if claveAcceso[y] == miClaveUnicode[x]:
            x = x + 2
            y = y + 2
        else:
            boolean = False
    return boolean
```

Ahora únicamente queda entender el código:

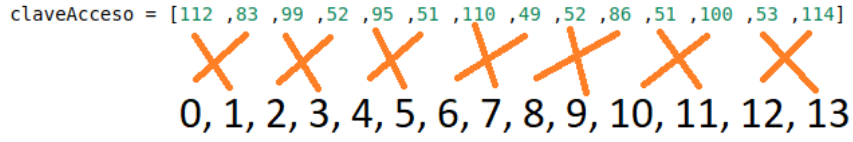
Mientras *boolean* sea *True* y *x* no valga 14, se comprueba si la clave secreta en *x* es igual a la de nuestro input en *y*. Vemos que al principio *x* vale 0 e *y* vale 1. En caso de que sea igual, se pasa al siguiente if y se comprueba si la clave secreta en *y* es igual a la de nuestro input en *x*.

Aquí ya vemos que se han intercambiado los valores de x y de y, haciendo una cruz.

Además, en caso de que sean iguales, x e y se aumentan en 2 unidades. Es decir, que en la primera iteración, *x* pasaría a valer 3 e *y* 2.

Como se mantendrían las condiciiones para pasar el while, se comprueba si la clave secreta en *3* es igual a la de nuestro input en *2*. En caso de que sea igual, se pasa al siguiente if y se comprueba si la clave secreta en *2* es igual a la de nuestro input en *3*.

Aquí ya vemos un patrón: se coge el valor de nuestro input en *x* y el valor de la clave secreta en *y*. Luego se intercambian las variables y se aumentan en 2. Va comparando en cruz:



Por ello, vemos que el primer valor de claveAcceso sería el segundo, el segundo el primero, el tercero el cuarto, etc. Reordenamos y obtenemos lo que sería la clave verdadera con la función *chr*:

```
claveAcceso = [83,112,52,99,51,95,49,110,86,52,100,51,114,53]
claveEnClaro = [chr(c) for c in claveAcceso]
print(claveEnClaro)
```

```
└─$ python3 ./a.py
['S', 'p', '4', 'c', '3', '_', '1', 'n', 'V', '4', 'd', '3', 'r', '5']
```

```
└─$ python3 ./aliens.py
Bienvenido a la nave, indique su contraseña de acceso:
Sp4c3_1nV4d3r5
¡Bienvenido a la nave!
```