

DNS EXFILTRATION

DNS EXFILTRATION

ALEJANDRO TORAL ROMERO



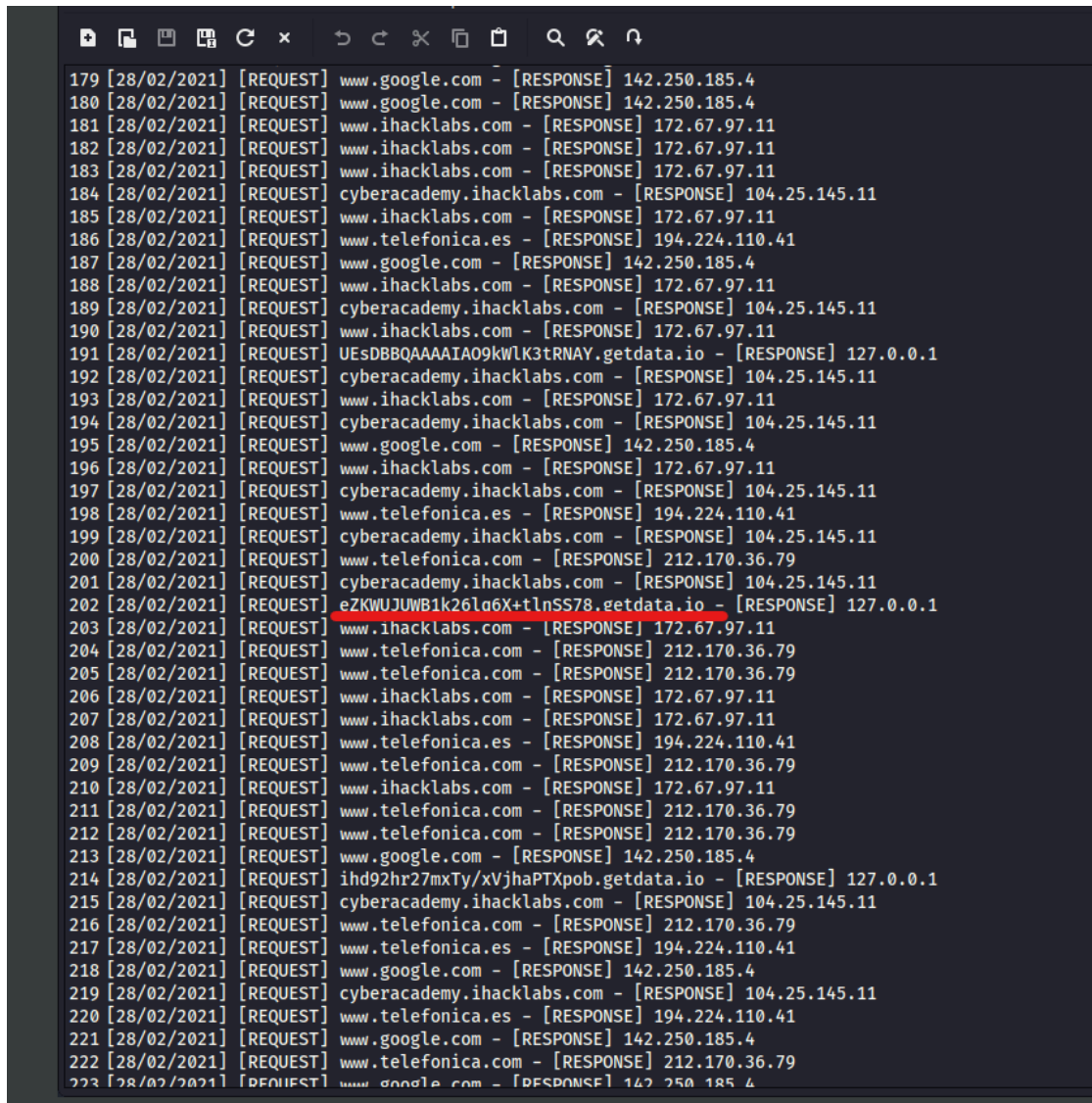
Enunciado

Los datos del equipo del director han vuelto a ser robados, el método ha sido por DNS. El atacante ha intentado que estos datos no sean descubiertos y ha ofuscado todo el proceso.

Podrías ayudarnos a conseguir los datos exfiltrados?

Solución

Si abrimos el fichero de log, veremos que dentro de los logs del DNS en los subdominios del dominio getdata.io hay caracteres que parecen estar en base64.



```

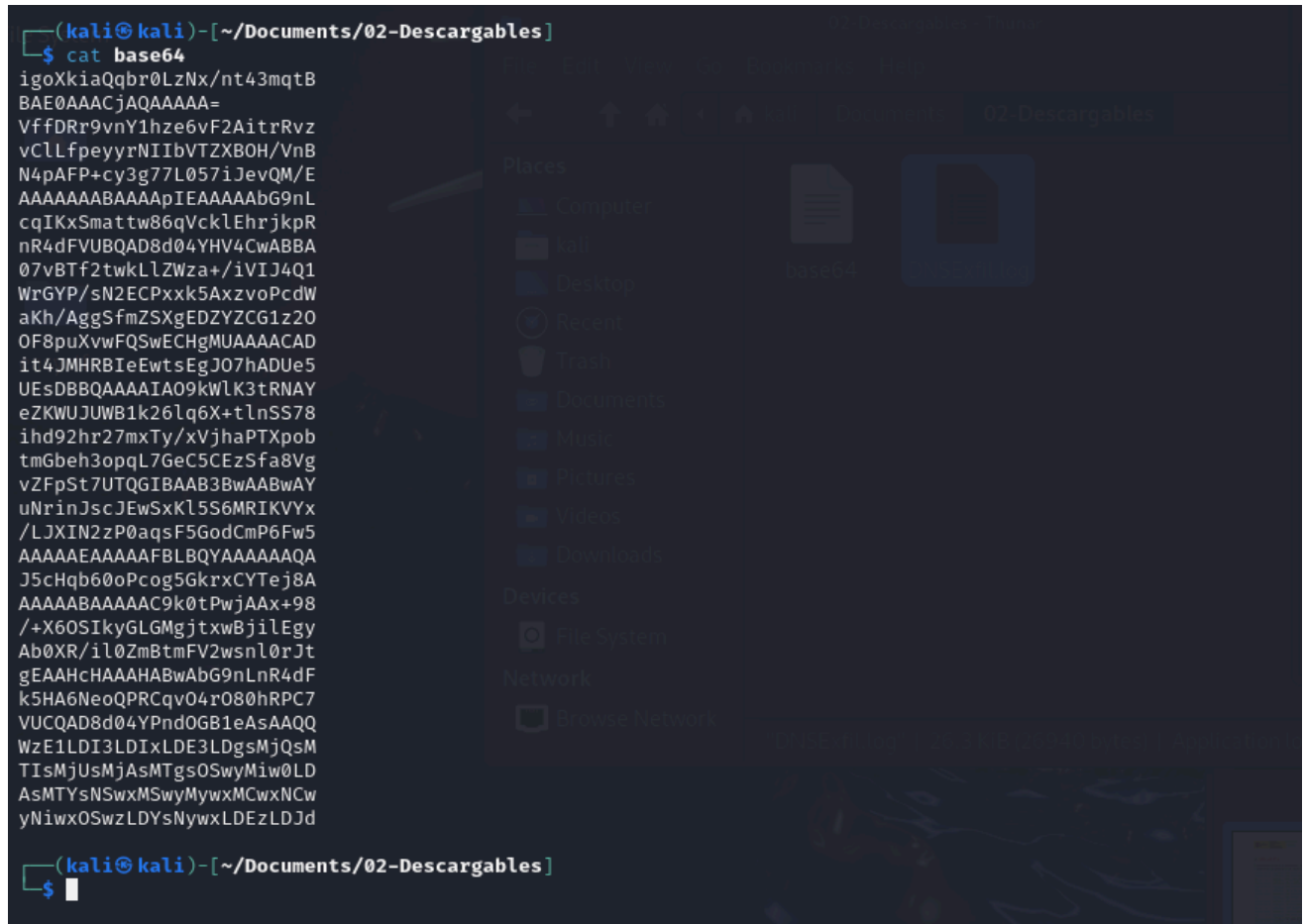
179 [28/02/2021] [REQUEST] www.google.com - [RESPONSE] 142.250.185.4
180 [28/02/2021] [REQUEST] www.google.com - [RESPONSE] 142.250.185.4
181 [28/02/2021] [REQUEST] www.ihacklabs.com - [RESPONSE] 172.67.97.11
182 [28/02/2021] [REQUEST] www.ihacklabs.com - [RESPONSE] 172.67.97.11
183 [28/02/2021] [REQUEST] www.ihacklabs.com - [RESPONSE] 172.67.97.11
184 [28/02/2021] [REQUEST] cyberacademy.ihacklabs.com - [RESPONSE] 104.25.145.11
185 [28/02/2021] [REQUEST] www.ihacklabs.com - [RESPONSE] 172.67.97.11
186 [28/02/2021] [REQUEST] www.telefonica.es - [RESPONSE] 194.224.110.41
187 [28/02/2021] [REQUEST] www.google.com - [RESPONSE] 142.250.185.4
188 [28/02/2021] [REQUEST] www.ihacklabs.com - [RESPONSE] 172.67.97.11
189 [28/02/2021] [REQUEST] cyberacademy.ihacklabs.com - [RESPONSE] 104.25.145.11
190 [28/02/2021] [REQUEST] www.ihacklabs.com - [RESPONSE] 172.67.97.11
191 [28/02/2021] [REQUEST] UEsDBBQAAAAIAO9kWLK3trNAY.getdata.io - [RESPONSE] 127.0.0.1
192 [28/02/2021] [REQUEST] cyberacademy.ihacklabs.com - [RESPONSE] 104.25.145.11
193 [28/02/2021] [REQUEST] www.ihacklabs.com - [RESPONSE] 172.67.97.11
194 [28/02/2021] [REQUEST] cyberacademy.ihacklabs.com - [RESPONSE] 104.25.145.11
195 [28/02/2021] [REQUEST] www.google.com - [RESPONSE] 142.250.185.4
196 [28/02/2021] [REQUEST] www.ihacklabs.com - [RESPONSE] 172.67.97.11
197 [28/02/2021] [REQUEST] cyberacademy.ihacklabs.com - [RESPONSE] 104.25.145.11
198 [28/02/2021] [REQUEST] www.telefonica.es - [RESPONSE] 194.224.110.41
199 [28/02/2021] [REQUEST] cyberacademy.ihacklabs.com - [RESPONSE] 104.25.145.11
200 [28/02/2021] [REQUEST] www.telefonica.com - [RESPONSE] 212.170.36.79
201 [28/02/2021] [REQUEST] cyberacademy.ihacklabs.com - [RESPONSE] 104.25.145.11
202 [28/02/2021] [REQUEST] eZKWUJUWB1k26ld6X+tlNss78.getdata.io - [RESPONSE] 127.0.0.1
203 [28/02/2021] [REQUEST] www.ihacklabs.com - [RESPONSE] 172.67.97.11
204 [28/02/2021] [REQUEST] www.telefonica.com - [RESPONSE] 212.170.36.79
205 [28/02/2021] [REQUEST] www.telefonica.com - [RESPONSE] 212.170.36.79
206 [28/02/2021] [REQUEST] www.ihacklabs.com - [RESPONSE] 172.67.97.11
207 [28/02/2021] [REQUEST] www.ihacklabs.com - [RESPONSE] 172.67.97.11
208 [28/02/2021] [REQUEST] www.telefonica.es - [RESPONSE] 194.224.110.41
209 [28/02/2021] [REQUEST] www.telefonica.com - [RESPONSE] 212.170.36.79
210 [28/02/2021] [REQUEST] www.ihacklabs.com - [RESPONSE] 172.67.97.11
211 [28/02/2021] [REQUEST] www.telefonica.com - [RESPONSE] 212.170.36.79
212 [28/02/2021] [REQUEST] www.telefonica.com - [RESPONSE] 212.170.36.79
213 [28/02/2021] [REQUEST] www.google.com - [RESPONSE] 142.250.185.4
214 [28/02/2021] [REQUEST] ihd92hr27mxTy/xVjhaPTXpob.getdata.io - [RESPONSE] 127.0.0.1
215 [28/02/2021] [REQUEST] cyberacademy.ihacklabs.com - [RESPONSE] 104.25.145.11
216 [28/02/2021] [REQUEST] www.telefonica.com - [RESPONSE] 212.170.36.79
217 [28/02/2021] [REQUEST] www.telefonica.es - [RESPONSE] 194.224.110.41
218 [28/02/2021] [REQUEST] www.google.com - [RESPONSE] 142.250.185.4
219 [28/02/2021] [REQUEST] cyberacademy.ihacklabs.com - [RESPONSE] 104.25.145.11
220 [28/02/2021] [REQUEST] www.telefonica.es - [RESPONSE] 194.224.110.41
221 [28/02/2021] [REQUEST] www.google.com - [RESPONSE] 142.250.185.4
222 [28/02/2021] [REQUEST] www.telefonica.com - [RESPONSE] 212.170.36.79
223 [28/02/2021] [REQUEST] www.google.com - [RESPONSE] 142.250.185.4

```

DNS EXFILTRATION

Vamos a filtrar por todos los que pongan getdata y vamos a quedarnos solo con el texto en base 64.

```
cat DNSExfil.log | grep getdata | cut -d' ' -f3 | cut -d'.' -f1 | grep -E '^[A-Za-z0-9+/=]+$' > base64
```



DNS EXFILTRATION

Ahora vamos a decodificar línea por línea utilizando el siguiente script. Esto al ejecutarlo nos va a devolver el resultado en el fichero temp.txt. [DESCARGAR](#)

```
#!/bin/bash
num=1
cat base64 | while read line
do
    echo $line > ./temp.txt
    echo $num
    base64 -d ./temp.txt
    num=$((num + 1))
done
```

```
(kali㉿kali)-[~/Documents/02-Descargables]
$ chmod +x script.sh
```

```
(kali㉿kali)-[~/Documents/02-Descargables]
$ ./script.sh
1
```

Si nos fijamos en el resultado en la línea 29 vemos que empieza un vector de números.

```
File Actions Edit View Help
@P`base64: invalid input
22
'*****Mbase64: invalid input
23
Mbase64: invalid input
24
9"$b+cQ base64: invalid input
25
Gbbau
'Jbase64: invalid input
26
@""base64: invalid input
27
----@B-----base64: invalid input
28
U@?lWbase64: invalid input
29
[15,27,21,17,8,24,base64: invalid input
30
L
K
N
K
base64: invalid input
31
bR##2Bbase64: invalid input
32
jbbIbase64: invalid input
(kali㉿kali)-[~/Documents/02-Descargables]
$
```

DNS EXFILTRATION

Vamos a coger desde la línea 29 hasta la última.

```
/LJXIN2zP0aqsF5GodCmP6Fw5
AAAAAEAAAAFBLBQYAAAAAQA
J5cHqb60oPcog5GkrxCYTej8A
AAAAABAAAAAC9k0tPwjAAx+98
/+X60SIkyGLGMgjt看wBjilEgy
Ab0XR/il0ZmBtmFV2wsnl0rJt
gEAAHcHAAAHABwAbG9nLnR4dF
k5HA6NeoQPRCqv04r080hRPC7
VUCQAD8d04YPnd0GB1eAsAAQ
WzE1LDI3LDIxLDE3LDgsMjQsM
TIsMjUsMjAsMTgsOSwyMiw0LD
AsMTYsNSwxMSwyMywxMCwxNCw
yNiwXOSwzLDYsNywxLDEzLDJd
```

Si lo decodificamos nos saldrá este vector .

```
echo
"WzE1LDI3LDIxLDE3LDgsMjQsMTIsMjUsMjAsMTgsOSwyMiw0LDAsMTYsNSwx
MSwyMywxMCwxNCwyNiwXOSwzLDYsNywxLDEzLDJd" | base64 --decode
```

```
(kali@kali)-[~/Documents/02-Descargables]
$ echo "WzE1LDI3LDIxLDE3LDgsMjQsMTIsMjUsMjAsMTgsOSwyMiw0LDAsMTYsNSwxMSwyMywxMCwxNCwyNiwXOSwzLDYsNywxLDEzLDJd" | base64 --decode
[15,27,21,17,8,24,12,25,20,18,9,22,4,0,16,5,11,23,10,14,26,19,3,6,7,1,13,2]

(kali@kali)-[~/Documents/02-Descargables]
$
```

DNS EXFILTRATION

De esta forma podemos ver que el número máximo corresponde al número de líneas que hay en el archivo codificado, lo que podría resultar en el orden correcto para decodificarlo. Para esto haremos un código en python para ordenarlo. [DESCARGAR](#)

```
orden =
[15,27,21,17,8,24,12,25,20,18,9,22,4,0,16,5,11,23,10,14,26,19
,3,6,7,1,13,2]

b64list = open("./base64", "r").readlines()
base64_final = ""

for x in range(len(orden)):
    index = orden.index(x)
    base64_final += b64list[index].replace("\n", "")
print(base64_final)
```

```
(kali㉿kali)-[~/Documents/02-Descargables]
$ python ordenar.py
UESDBBQAAAAIAO9kWlK3tRNAYgEAAHcHAAAABwAbG9nLnR4dFVUCQAD8d04YPndOGB1eAsAAQAAAAABAAAAAC9k0tPwjAAx+98it4JMHRBIEEwtsEg
JO7hADUe5ihd92hr27mxTy/xVjhaPTXpob/+X6OSIkyGLGMgjt看wBjilEgyAb0XR/il0ZmBtmFV2wsnl0rJtN4pAFP+cy3g77L057iJevQM/EaKh/Agg
SfmZSXgEDZYZCG1z2OuNrInJscJEwSxKl5S6MRIKVYxtmGbeh3opqL7GeC5CEzSfa8VgcqIKxSmattw86qVcklEhrjKpRk5HA6NeoQPRCqv04r080hRP
C7/LJXIN2zP0aqsF5GodCmP6Fw5igoXkiaQqbr0LzNx/nt43mqtBeZKWUJUWB1k26lq6X+tlN5S78vCLLfpeyyrNIibVTZXB0H/VnBWrGYP/sN2ECPxx
k5AxzvoPcdWJ5cHqb60oPcog5GkrxCYTej8A07vBTf2twkLlZWza+/iVIJ4Q1VffDRr9vnY1hze6vF2AitrRvzOF8puXvwFQSwECHgMUAACADvZFpS
t7UTQGIBAAB3BwAABwAYAAAAABAAAApIEAAAAAbG9nLnR4dFVUBQAD8d04YHV4CwABBAAAAAAEAAAAAFBLBQYAAAAAQABAE0AAACjAQAAAAA=
```

Vamos a guardar el codificado en el orden correcto en el archivo base64_completo.txt

```
echo
UESDBBQAAAAIAO9kWlK3tRNAYgEAAHcHAAAABwAbG9nLnR4dFVUCQAD8d04Y
PndOGB1eAsAAQAAAAABAAAAAC9k0tPwjAAx+98it4JMHRBIEEwtsEgJO7hA
DUe5ihd92hr27mxTy/xVjhaPTXpob/+X6OSIkyGLGMgjt看wBjilEgyAb0XR/
il0ZmBtmFV2wsnl0rJtN4pAFP+cy3g77L057iJevQM/EaKh/AggSfmZSXgEDZ
YZCG1z2OuNrInJscJEwSxKl5S6MRIKVYxtmGbeh3opqL7GeC5CEzSfa8VgcqI
KxSmattw86qVcklEhrjKpRk5HA6NeoQPRCqv04r080hRPC7/LJXIN2zP0aqsF
5GodCmP6Fw5igoXkiaQqbr0LzNx/nt43mqtBeZKWUJUWB1k26lq6X+tlN5S78
vCLLfpeyyrNIibVTZXB0H/VnBWrGYP/sN2ECPxxk5AxzvoPcdWJ5cHqb60oPc
og5GkrxCYTej8A07vBTf2twkLlZWza+/iVIJ4Q1VffDRr9vnY1hze6vF2Aitr
RvzOF8puXvwFQSwECHgMUAACADvZFpSt7UTQGIBAAB3BwAABwAYAAAAAB
AAAApIEAAAAAbG9nLnR4dFVUBQAD8d04YHV4CwABBAAAAAAEAAAAAFBLBQYAA
AAAAQABAE0AAACjAQAAAAA= > base64_completo.txt
```

DNS EXFILTRATION

Después vamos a decodificar el archivo que acabamos de crear.

```
base64 -d base64_completo.txt > datos_decodificados
```

Si le hacemos un file nos dice que es un archivo comprimido zip.

```
(kali㉿kali)-[~/Documents/02-Descargables]
$ file datos_decodificados
datos_decodificados: Zip archive data, at least v2.0 to extract, compression method=deflate
```

Lo descomprimimos .

```
(kali㉿kali)-[~/Documents/02-Descargables]
$ unzip datos_decodificados
Archive:  datos_decodificados
  inflating: log.txt
```

Dentro de este zip hay un log.txt, que si lo abrimos nos encontramos unos logs de acceso. Como se ve está utilizando la encriptación RC4.

```
[DEBUG] Password encrypted with RC4.
/login.php USER: user - PASSWORD: Ck09lJ8= - ACCESS SUCCESSFUL.
[DEBUG] Password encrypted with RC4.
/login.php USER: administrator - PASSWORD: IVQ4jPT93w== - ACCESS SUCCESSFUL.
[DEBUG] Password encrypted with RC4.
/login.php USER: oracle - PASSWORD: CUQhh/zxoWI= - ACCESS SUCCESSFUL.
[DEBUG] Password encrypted with RC4.
/login.php USER: ftp - PASSWORD: CkYxk+HxpmI= - ACCESS SUCCESSFUL.
[DEBUG] Password encrypted with RC4.
/login.php USER: pi - PASSWORD: HlQ1jZ8= - ACCESS SUCCESSFUL.
[DEBUG] Password encrypted with RC4.
/login.php USER: puppet - PASSWORD: HEgg6g= - ACCESS SUCCESSFUL.
[DEBUG] Password encrypted with RC4.
/login.php USER: ansible - PASSWORD: CU01h+7UmzsFXA+LAScdtQRrcxssJhs= - ACCESS SUCCESSFUL.
[DEBUG] Password encrypted with RC4.
/login.php USER: ec2-user - PASSWORD: AkAg1Pzj3w== - ACCESS SUCCESSFUL.
[DEBUG] Password encrypted with RC4.
/login.php USER: vagrant - PASSWORD: PEQw6g= - ACCESS SUCCESSFUL.
[DEBUG] Password encrypted with RC4.
/login.php USER: azureuser - PASSWORD: HVQgkuD93w== - ACCESS SUCCESSFUL.
[DEBUG] Password encrypted with RC4.
/login.php USER: - PASSWORD: - ACCESS SUCCESSFUL.
[DEBUG] Password encrypted with RC4.

(kali㉿kali)-[~/Documents/02-Descargables]
$
```

Entre las claves que nos encontramos vamos a elegir la de mayor longitud.

DNS EXFILTRATION

Vamos a crear un php para hacerle fuerza bruta a la clave con el diccionario rockyou.

[DESCARGAR](#)

```
<?php
error_reporting(0);
require_once("../rc4.php");
$sciphertext = "CU81h7UmzsFXA+LASEtdQRxex8sJhs=";
echo "Start date: " . date("Y M d H:i") . "\n";
$fp = fopen("/usr/share/wordlists/rockyou.txt", "r");
while (!feof($fp)) {
    $linea = str_replace("\n", "", fgets($fp));
    $data = "";
    try {
        $data = rc4($linea, base64_decode($sciphertext));
    }
    catch (DivisionByZeroError $e) {
        continue;
    }
    $valid = strstr($data, "flag");
    if ($valid) {
        echo "Clave: " . $linea . " - Datos: " . $data;
    }
}
fclose($fp);
?>
```


DNS EXFILTRATION

El fichero rc4.php tendrá lo siguiente. [DESCARGAR](#)

```
<?php
function rc4($key, $str) {
    $s = array();
    for ($i = 0; $i < 256; $i++) {
        $s[$i] = $i;
    }
    $j = 0;
    for ($i = 0; $i < 256; $i++) {
        $j = ($j + $s[$i] + ord($key[$i % strlen($key)])) % 256;
        $temp = $s[$i];
        $s[$i] = $s[$j];
        $s[$j] = $temp;
    }
    $i = 0;
    $j = 0;
    $res = '';
    for ($y = 0; $y < strlen($str); $y++) {
        $i = ($i + 1) % 256;
        $j = ($j + $s[$i]) % 256;
        $temp = $s[$i];
        $s[$i] = $s[$j];
        $s[$j] = $temp;
        $res .= $str[$y] ^ chr($s[(($s[$i] + $s[$j]) % 256)]);
    }
    return $res;
}
?>
```

Cuando ejecutemos nuestro php empezará a probar claves. Cuando una de estas sea correcta nos lo mostrará junto con el texto cifrado que es la flag.

```
(kali@kali)-[~/Documents/02-Descargables/php-rc4]
$ php descodificar_rc4.php
```

DNS EXFILTRATION

Tras esperar un rato hemos obtenido la clave correcta con su correspondiente flag.

```
Probando clave: 014789ss
Probando clave: 014789nino
Probando clave: 014789hugo014789
Probando clave: 014789fuck
Probando clave: 014789ch
Probando clave: 014789as
Probando clave: 01478999
Probando clave: 01478998
Probando clave: 01478987410
Probando clave: 0147898
Probando clave: 014789778
Probando clave: 014789772
Probando clave: 0147896yaz
Probando clave: 01478966
Probando clave: 0147896542.79520
Probando clave: 014789654
Probando clave: 0147896520
Probando clave: 01478963o
Probando clave: 01478963n
Probando clave: 01478963hk
Probando clave: 014789636542
Probando clave: 0147896352
Probando clave: 014789633
Probando clave: 0147896325yo
Probando clave: 0147896325samuel2281boynton
Probando clave: 0147896325eli
Probando clave: 0147896325abcd
Probando clave: 014789632587410014789632587410014789632587410014789632587410
Clave encontrada: 014789632587410014789632587410014789632587410014789632587410
Texto descifrado: flag{DNS_3xf2♦tr4t10n}
```

Clave: 014789632587410014789632587410014789632587410014789632587410

Flag: DNS_3xf1ltr4t10n