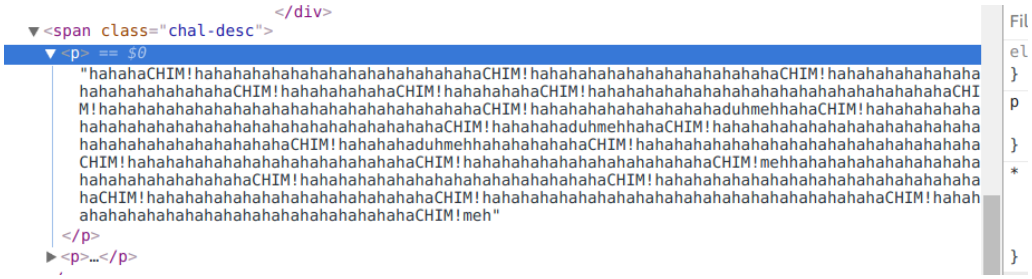
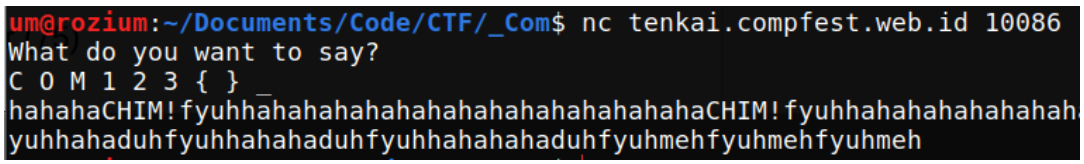
Team 7 - Write ups CTF Compfest 9

Cryptography - Cough Generator (25)

Diberikan sebuah string yang panjaaaaang yang kira-kira adalah flag yang di-enkrip. Inspect element terlebih dahulu untuk mendapatkan full string yang telah di-enkrip tadi.



Pertama-tama kita coba dulu nc `tenkai.compfest.web.id 10086`



Ternyata sebuah generator. Didapatkan kesimpulan

- huruf `C` menghasilkan `hahahaCHIM!` (bisa diliat, jumlah ha == huruf yang dimaksud),
- angka `1` menghasilkan `hahaduh` (bisa diliat, jumlah ha-1 == angka yang dimaksud),
- `spasi` menghasilkan `fyuh`,
- dan simbol menghasilkan `meh`.

Dari situ, kita bisa menggenerate string yang diberikan dan mendapatkan flag.

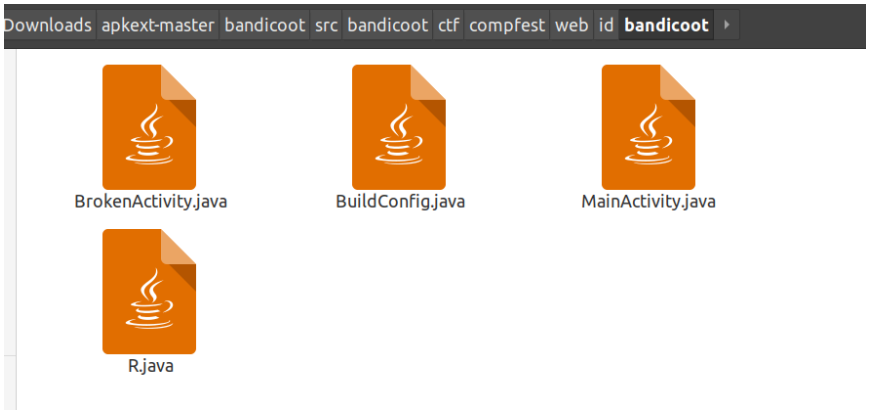
flag : **COMPFEST9{BY3_BY3_FROM_SPROUT}**

Reversing - Bandicoot (50)

Why is it always crashed? :(

BANDICOOT.APK

Diberikan sebuah file bandicoot.apk. Pertama-tama kita extract dulu file nya pakai apkext (cara make nya bisa diliat disini https://github.com/blukat29/apkext). Kemudian didapatkan file.

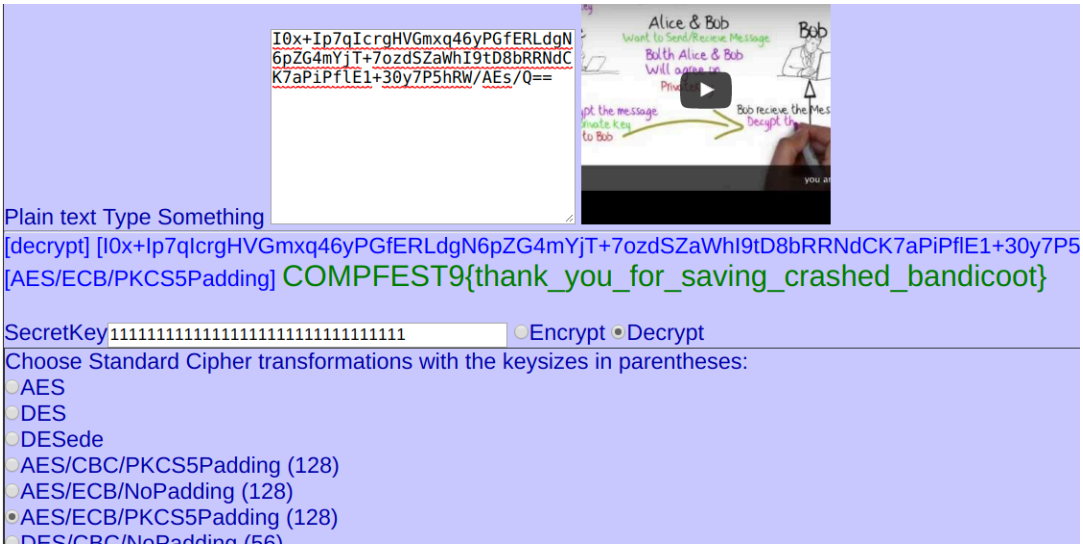


Jika diliat pada file `BrokenActivity.java`, terdapat sebuah perintah untuk menulis flag dan juga fungsi untuk dekrip flag

```
@Override
public void onLowMemory() {
    String string = "1";
    for (int i = 0; i <= 4; ++i) {
        string = new StringBuilder(string).reverse().toString() + string;
    }
    Toast.makeText((Context)this, (CharSequence)(
        "The flag is " + this.decryptFlag(string.getBytes(), "I0x+Ip7qIcrgHVGmxq46yPGfERLdgN6pZG4mYjT+7ozdSZaWhI9tD8bRRNdCK7aPiPflE1+30y7P5hRW/AEs/Q=="));
}
```

```
protected String decryptFlag(final byte[] array, final String s) {
    try {
        final SecretKeySpec secretKeySpec = new SecretKeySpec(array, "AES");
        final Cipher instance = Cipher.getInstance("AES/ECB/PKCS5PADDING");
        instance.init(2, secretKeySpec);
        return new String(instance.doFinal(Base64.decode(s, 0)));
    }
    catch (Exception ex) {
        Log.e("bandicoot", "[Error Decrypting] " + ex.getMessage());
        throw new RuntimeException("Bandicoot crashed!");
    }
}
```

Setelah membaca dan memahami maksud dari program itu, kemudian didapatkan bahwa flag di-enkripsi menggunakan algoritma AES/ECB/PKCS5PADDING dengan key “1111...” sebanyak 32 kali. Kemudian kita bisa menggunakan sebuah website online (http://8gwifi.org/CipherFunctions.jsp) untuk mendekrip flag.



flag : **COMPFEST9{thank_you_for_saving_crashed_bandicoot}**

Reversing - Not So Classic String Validator (75)

In CTF world, string validation is a classic challenge in Reverse Engineering. I hope this easy problem is not so classic.

validator

Diberikan sebuah file validator. Pertama-tama kita generate pseudocode nya dengan IDA

```
13  printf("Input Password: ", argv, envp);
14  __isoc99_scanf("%s", s);
15  printf("Input integer constant: ");
16  __isoc99_scanf("%d", &v6);
17  v8 = strlen(s);
18  if ( v8 == 46 )
19  {
20    for ( i = 0; i < v8; ++i )
21    {
22      v4 = exp((double)s[i] * 13.0 / (double)v6);
23      if ( (signed int)floor(v4) != dxdiag[i] )
24      {
25        result = 0;
26        goto LABEL_9;
27      }
28    }
29    printf("Congrats!\nYour flag is COMPFEST9{%s}\n", s);
30    result = 0;
31  }
```

Dari pseudocode, bisa diliat program meminta input 2x yang pertama password kemudian sebuah integer constant yang nanti akan digunakan untuk validasi string yang di-input. Jika dilihat pada baris ke 23, terdapat pengecekan input kita yang sudah di operasikan sedemikian rupa dengan sebuah array ‘dxdiag’. Bisa dilihat juga pada baris ke 18, panjang input kita harus

46 karakter. Jika kita klik 2x pada array, data-data pada array akan bermunculan, dan menariknya, ada 46 data yang tersedia. Kemudian kita bikin skript nya untuk mendapatkan flag. Nilai const didapatkan dengan melakukan bruteforce dan kemudian mencocokkan apakah ada hasil yang sesuai dengan nilai pada array dxdiag, kemudian didapatkan const = 97

```
flag = "COMPFEST9{"

const = 97

charset = string.printable

data = [0x268C60, 0x6C02D, 0x12EA4, 0x628006, 4318205, 0x427, 0x1D7C16, 0x529D7, 0x876D, 0x628006,
        0x21B69B, 0x1B1E, 0x3A1, 0x41E3FD, 0x529D7, 0x13B936, 0x4B5723, 0x529D7, 0x56253A, 0x113FE6,
        0x3A1, 0x529D7, 0x7B806, 0x0B89F8, 0x108AE, 0x56253A, 0x529D7, 0x4B5723, 0x56253A, 0x0E77C,
        0x13B936, 0x876D, 0x0F1605, 0x529D7, 0x8D36B, 0x113FE6, 0x3A1, 0x8D36B, 0x19C958, 0x2889,
        0x41E3FD, 0x529D7, 0x0B89F8, 0x70A06A, 0x2889, 0x41E3FD]
i = 0
for x in range(46):
    for c in charset:
        um = math.exp(ord(c) * 13.0 / const)
        if (math.floor(um) == data[i]):
            flag += c
            i += 1
        if i == 46:
            break
    if i == 46:
        break

flag += '}'
print flag
```

```
um@rozium:~/Documents/Code/CTF/_Com$ python validator.py
COMPFEST9{naTur4l_NumB3r_is_th3_beSt_stRiNg_ch3ckEr_evEr}
```

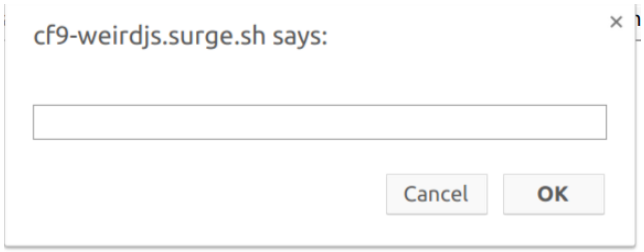
flag : **COMPFEST9{naTur4l_NumB3r_is_th3_beSt_stRiNg_ch3ckEr_evEr}**

Web - Weird.js (25)

HTTPS://CF9-WEIRDJS.SURGE.SH

Key SUBMIT

Pertama-tama akses url yang diberikan, kemudian muncul sebuah pop-up untuk memasukkan password. Kita coba masukkan ‘COMPFEST9{‘ dan kemudian muncul balasan ‘Wrong!’



Kita coba inspect-element dan melihat something menarik

Saya pun teringat pernah mengerjakan soal PicoCTF(kalau ga salah) yang hampir mirip dan teringat bahwa ini adalah JSF*ck, kemudian googling dan dapat sebuah thread https://reverseengineering.stackexchange.com/questions/10950/how-to-deobfuscated-javascript
Tinggal ikuti langkah-langkah yang diberikan dan didapatkan flag

```
1 (function() {
2 alert(prompt() === 'COMPFEST9{js_is_getting_weirder_everyday}' ? 'Correct!' : 'Wrong!');
3 })
```

flag : **COMPFEST9{js_is_getting_weirder_everyday}**

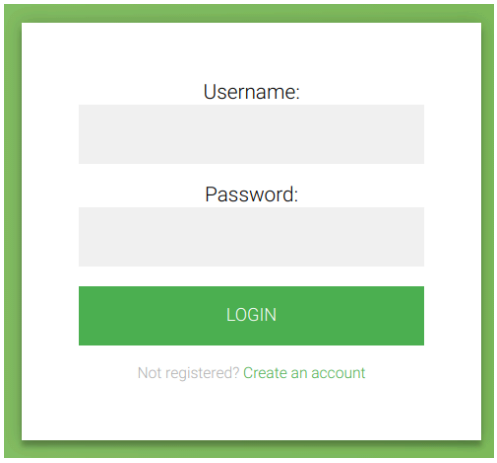
Web - Can You Get The Admin File? (50)

HTTP://TENKAI.COMPFEST.WEB.ID:18889

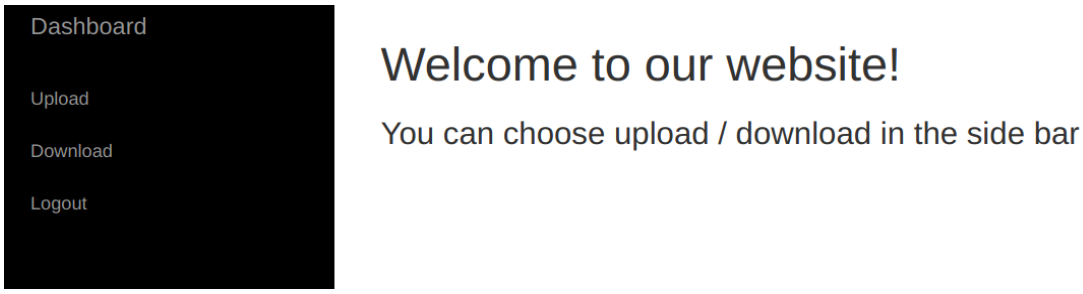
Key

SUBMIT

Tanpa pikir panjang, kita buka web nya kemudian ada login form



Ternyata kita bisa daftar juga, kita coba daftar dan kemudian login



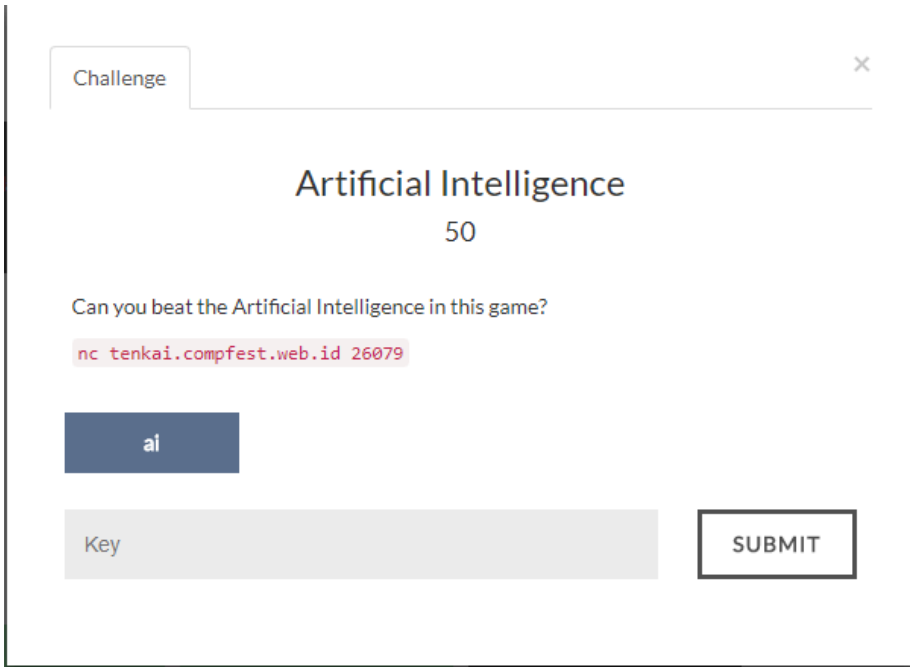
Di web tersebut kita bisa upload/download/logout, kita coba upload dan ternyata yang kita upload-an kemudian muncul di menu download

Jika kita lihat pada url, file yang di-upload tadi memiliki id 2223, kita coba ganti id dengan 1 maka muncul sebuah file yang di upload oleh admin

Namun file tersebut bukan flag, dan jika diperkirakan flag berada di sebuah id. Cara mendapatkan file nya tidak lain adalah dengan mem-bruteforce url sampai ketemu. Awal-nya dicoba bikin skrip python dengan request+session namun bermasalah pada bagian login nya, ada error gitu :(, jadi mau tak-mau demi 50 poin kita melakukan brute-force manual, tapi ga manual-manual juga, kita gunakan add-on firefox yang bernama HackBar, tinggal mencet Alt+Shift+=, id pada url akan bertambah 1 demi 1, setelah sampai pada id 898, alhamdulillah didapatkan flagnya

flag : **COMPFEST9{4lw4y5_u53_4uth_f0r_53cur1ty}**

Artificial Intelligence - 50 Points



Artificial Intelligence merupakan sebuah permainan pick the stone dimana anda harus mengalahkan AI. Terdapat 50 batu dan kita dapat mengamil 1-5 batu secara bergantian dengan AI. Yang terakhir menghabiskan batu adalah pemenangnya.

Sayangnya kita tidak mungkin mengalahkan AI dengan cara biasa. Pada langkah pertama AI selalu mengambil batu sebanyak 2 buah sehingga sisa batu adalah 48. Setiap turn kita memilih mengambil batu sebanyak n, AI akan mengambil batu sebanyak 6-n setelahnya, sehingga setiap rangkap turn, batu akan berkurang sebanyak 6. Karena 48 mod 6 adalah 0, dan turn terakhir selalu dimiliki oleh AI, maka AI akan selalu menang.
Saya mencoba disas fungsi main dan melihat-lihat isinya. Ternyata terdapat fungsi yang bernama invalid yang mengecek inputan kita. Saya lalu men disas fungsi invalid tersebut.

```
Dump of assembler code for function invalid:
   0x0000000000000930 <+0>:     push   %rbp
   0x0000000000000931 <+1>:     mov    %rsp,%rbp
   0x0000000000000934 <+4>:     mov    %edi,%eax
   0x0000000000000936 <+6>:     mov    %ax,-0x4(%rbp)
   0x000000000000093a <+10>:    cmpw   $0x0,-0x4(%rbp)
   0x000000000000093f <+15>:    jle    0x948 <invalid+24>
   0x0000000000000941 <+17>:    cmpw   $0x5,-0x4(%rbp)
   0x0000000000000946 <+22>:    jle    0x94f <invalid+31>
   0x0000000000000948 <+24>:    mov    $0x1,%eax
   0x000000000000094d <+29>:    jmp    0x954 <invalid+36>
   0x000000000000094f <+31>:    mov    $0x0,%eax
   0x0000000000000954 <+36>:    pop    %rbp
   0x0000000000000955 <+37>:    retq
End of assembler dump.
```

Dapat terlihat input kita dibandingkan dengan konstanta 0 dan 5 (range 1-5), dengan cmpw, Saya mencoba untuk melakukan overflow untuk dapat mengalahkan AI. Karena w pada cmpw berarti word, atau 32 bit, maka batas dari nilai yang dibandingkan adalah 2^(32-1) = 2147483648. Oleh karena itu saya mencoba memasukkan input batu sebanyak 2147483648 + 1 yaitu 2147483649, dan ternyata overflow berhasil dan muncullah flag yang kita cari.

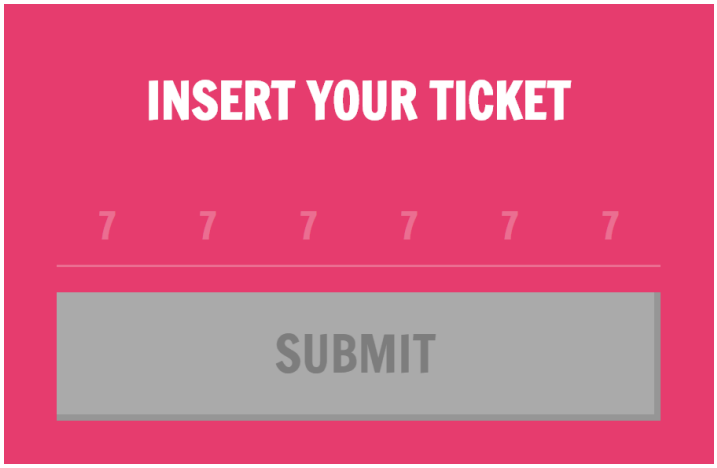
```
-- Your Turn --
Number of stones: 48
How many stones you want to pick? 2147483649
You pick -2147483647 stone

You Won!
COMPFEST9{int3g3r_155u35_101}dapid@UbuntuAsus:~/Downloads$
```

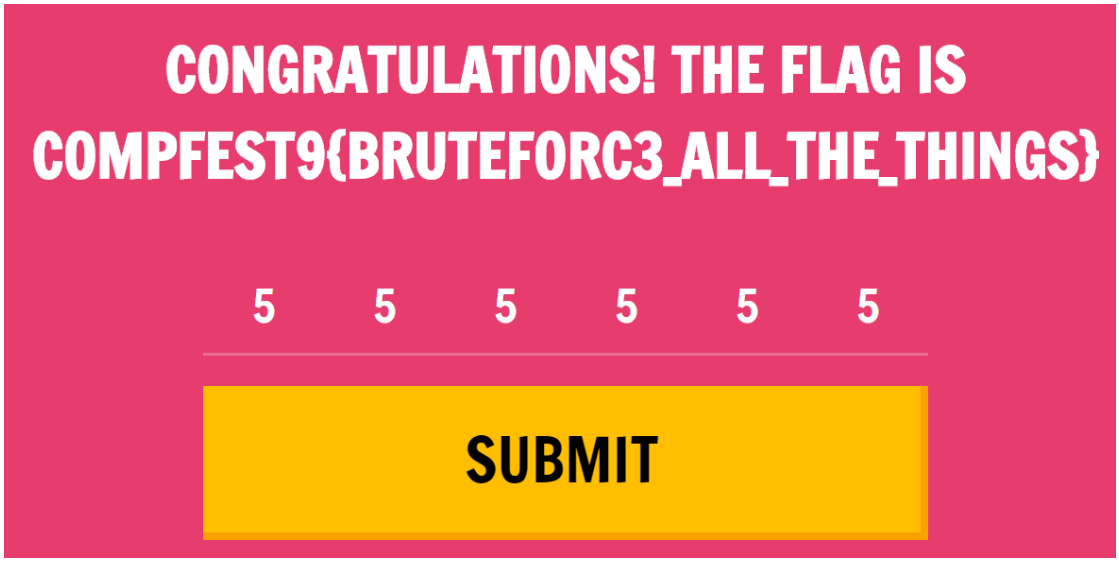
Flag: COMPFEST9{int3g3r_155u35_101}

Jackpot! - 150 Points

Pada challenge Jackpot! Terdapat link website yang jika dibuka akan muncul tampilan seperti ini:



Sepertinya kita harus memasukkan ticket yang benar untuk mendapatkan flag. Untuk mengetahui mekanismenya lebih lanjut, saya mencoba memasukkan angka 555555. Dan ternyata memang jackpot, saya langsung mendapatkan flag yang dicari:



Flag: COMPFEST9{BRUTEFORC3_ALL_THE_THINGS}

COMPFEST9 CTF 2017 – Can You Get The Plain Text

Category: cryptografi
Points: 50
Description : -
Hint: use python2
Attached file : *encrypt.py cipher_text.zip*

Write-up

User di beri sebuah code encripsi dalam bahasa python. Setelah dibaca kode tersebut melakukan enkripsi xor dan memecah nya menjadi beberapa file. Dari kode enkripsi dapat diketahui bahwa file plain text akan dipecah setiap panjang flag (dalam kasus ini 10 byte), dan apabila kurang dari 10 byte, maka akan ditambah dengan byte flag nya sampai pas 10 byte.

```
plain_text_file = open('ciphertext_112' , 'r')
plain_text = plain_text_file.read()
panjang = len(plain_text)
for n in range(0,110):
    k = 0
    key_file = open('flag_'+ str(n+1), 'r')
    key = key_file.read()
    cipher_text_file = open('plain_text_'+str(n+1), 'w')
    for i in range (12):
        for j in range(10):
            cipher_text_file.write(chr(ord(plain_text[k]) ^ ord(key[j])))
            k += 1
    key_file.close()
    cipher_text_file.close()
plain_text_file.close()
```

Dari informasi tersebut kita tinggal mengenerate semua kemungkinan flag nya. Flag ada kemungkinan benar apabila jika di xor kan pada 10 byte file akan menghasilkan hasil COMPFEST{. Jangan lupa menggabungkan ke12 part ciphertext menjadi satu kesatuan. (gabungan semua ciphertext saya beri nama file ciphertext_112)

```
plain_text_file = open('ciphertext_112', 'r')
plain_text = plain_text_file.read()
key = "COMPFEST9{"
x = 0
for i in plain_text:
    if (x < 110):
        plain_text_fix = plain_text[x+0]+plain_text[x+1]+plain_text[x+2]+plain_text[x+3]+plain_text[x+4]+plain_text[x+5]+plain_text[x+6]+plain_text[x+7]+plain_text[x+8]+plain_text[x+9]
        cipher_text_file = open('flag_' + str(x + 1), 'w')
        l = ""
        for j in range(0,10):
            m = (chr(ord(plain_text_fix[j]) ^ ord(key[j])))
            cipher_text_file.write(l[d])
        for z in range (x % 10):
            c = l[9]
            k = list(l)
            for y in range (9,0,-1):
                k[y] = k[y-1]
            k[0] = c
            l = "".join(k)
        for d in range (10):
            cipher_text_file.write(l[d])
        cipher_text_file.close()
    x+=1
plain_text_file.close()
key_file.close()
```

Setelah itu kita tinggal men-xor-kan semua kemungkinan flag dengan text enkripsi. Lalu buka lah satu persatu mana.

Flag : COMPFEST9{r3u51n9_th3_0n3_t1m3_p4d}