

Introducción a la Ciberseguridad

Trabajo Final

Tarea

Como trabajo final de Introducción a la Ciberseguridad, se debe participar en 2 CTFs internacionales de <https://ctftime.org>. En caso de no tener cuenta en CTftime, usar el mismo nick que en el CTFd de la cátedra.

En cada CTF se debe:

- Resolver al menos 3 retos
- Sobre los retos resueltos:
 - 2 retos deben ser de las categorías **REVERSING** y/o **PWN**.
 - 1 reto de cualquiera de las otras categorías.

Nota: Tener en cuenta que la selección no puede incluir retos de WARM UP, como por ejemplo, completar un mensaje en Discord o similar

Entrega

Por cada reto resuelto se debe entregar un writeup que incluya:

- Información sobre el desafío: nombre, categoria, descripcion, archivos adjuntos dados con el desafío (binarios, Dockerfile, etc)
- Resolución: Explicación de forma de resolverlo o guía paso a paso de la solución obtenida con capturas de pantalla que evidencien la resolución.
- Archivos adicionales: En caso que haya scripts diseñados para resolver el ejercicio u otro tipo de recurso, el mismo debe ser incluido.

Para evidenciar la participación en los CTFs se solicita incluir diversas capturas de pantalla:

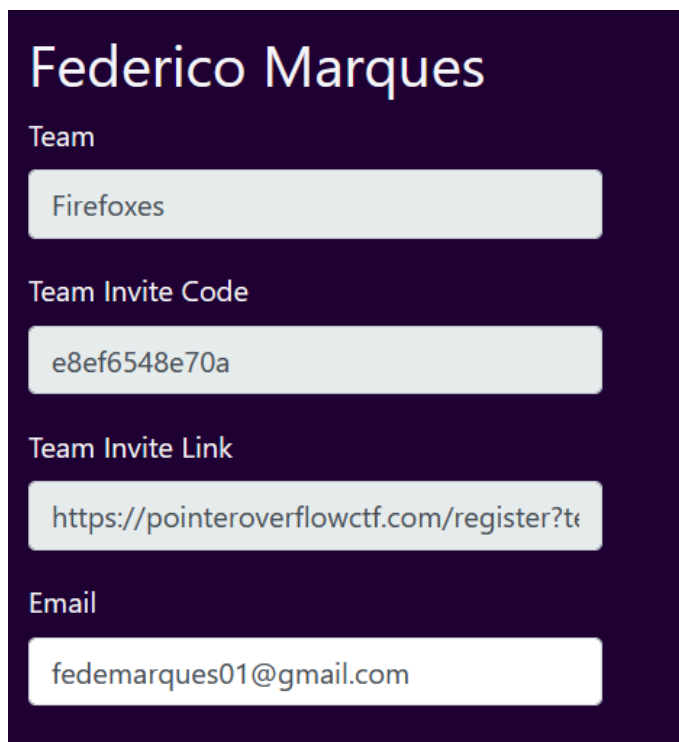
1. Registro en la plataforma de juego
2. Scoreboard cada vez que se resuelve un desafío en el que se vea:
 - al momento de hacer el submit de la flag
 - el puntaje asociado al jugador luego del submit
3. Reflexiones finales luego de la finalización del CTF con comentarios sobre lo que se aprendió y/o dificultades encontradas.

Primer CTF:

Enlace a la pagina del CTF: <https://pointeroverflowctf.com/>

Enlace en CTftime: <https://ctftime.org/event/2121>

Perfil del CTF



The image shows a CTF profile page for Federico Marques. The page has a dark purple background. The name 'Federico Marques' is at the top in white. Below it, the 'Team' section shows 'Firefoxes' in a light grey box. The 'Team Invite Code' section shows 'e8ef6548e70a' in a light grey box. The 'Team Invite Link' section shows 'https://pointeroverflowctf.com/register?te' in a light grey box. The 'Email' section shows 'fedemarques01@gmail.com' in a white box.

Reto 1 - Reverse 100 - End of the Line

Categoría: Reversing

Descripción: I got into CTFs a long time ago. Despite that, I remember how tough it was. It seemed impossible to solve all of the challenge. Heck, it seemed impossible to solve one. I remember how intimidating it was, but I also remember how amazing it felt to solve my first challenge and submit my first flag.

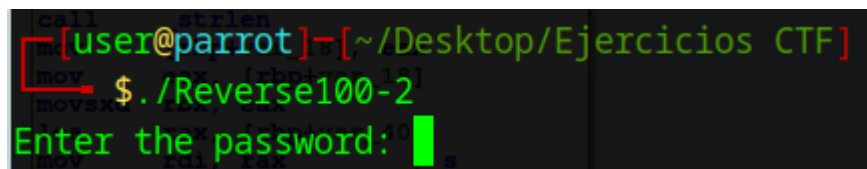
Looking back, it wasn't a tough challenge, but it was a start. Well, here's your chance. This is a recreation of the very first CTF challenge I solved. If you manage to solve this one, just remember that this is where I started, too.

Right Click, Save As... [Reverse100-2](#)

MD5 checksum 36C41D981AFBF5965FAFBF9ABE4757DC

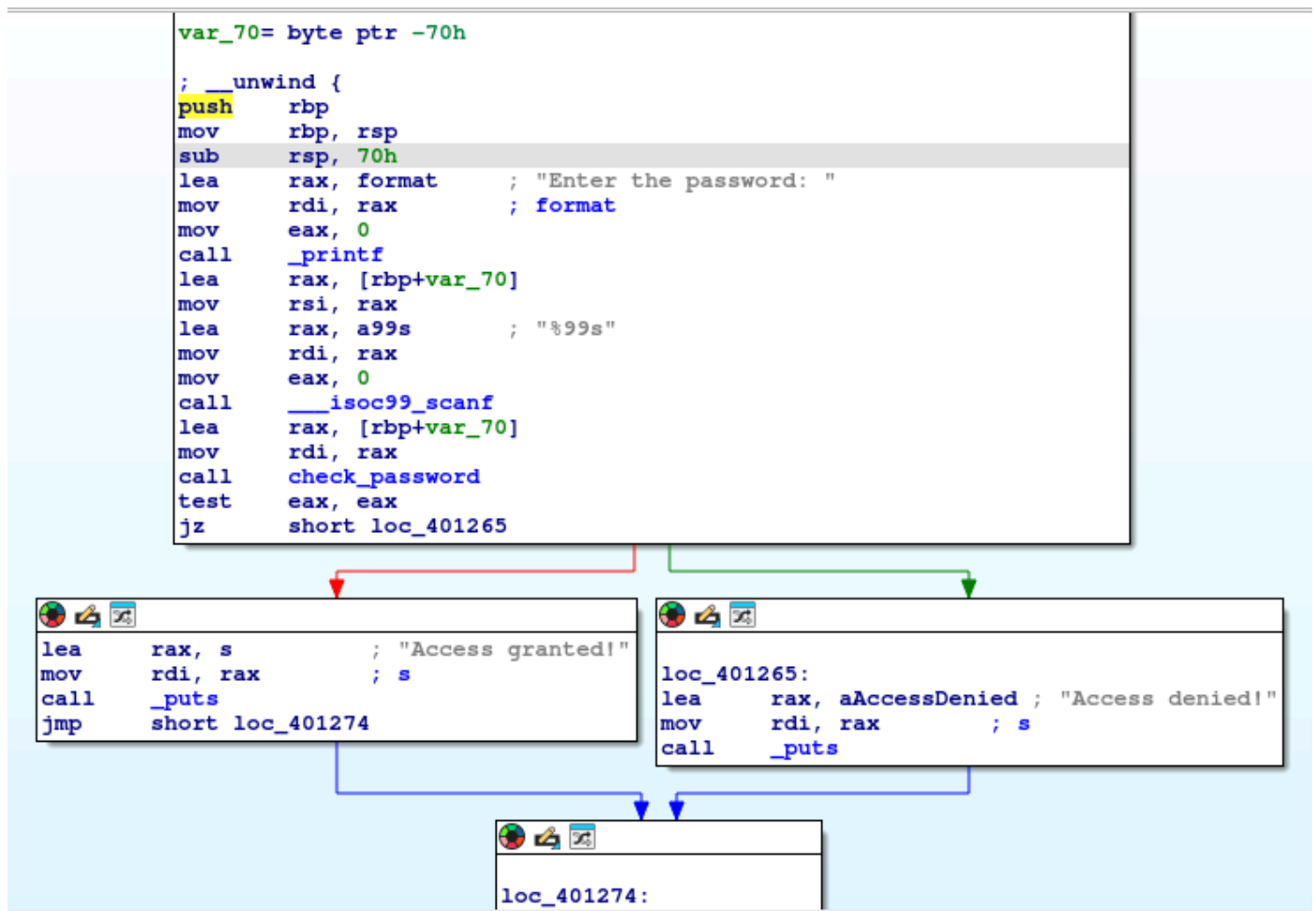
Resolución Reto 1

Para la resolución del ejercicio, lo primero que hice fue descargar el archivo adjunto, y ejecutarlo



Como se puede ver, pide una contraseña, por lo que como siguiente paso desasemble el archivo utilizando IDA

Observando el código assembly, vemos que llama a una rutina check_password para determinar si es correcta



Analizando la rutina check_password vemos que la contraseña parece estar dividida en segmentos en ASCII

```

public check_password
check_password proc near

s= qword ptr -48h
var_40= byte ptr -40h
var_38= qword ptr -38h
var_2F= qword ptr -2Fh
var_18= dword ptr -18h
var_14= dword ptr -14h
var_8= qword ptr -8

; __unwind {
push    rbp
mov     rbp, rsp
push    rbx
sub     rsp, 48h
mov     [rbp+s], rdi
mov     rax, 77757B6674636F70h
mov     rdx, 72305F30645F7073h
mov     qword ptr [rbp+var_40], rax
mov     [rbp+var_38], rdx
mov     rax, 5F72305F30645F70h
mov     rdx, 7D37306E5F3064h
mov     [rbp+var_38+1], rax
mov     [rbp+var_2F], rdx
mov     rax, [rbp+s]
mov     rdi, rax          ; s
call    _strlen
mov     [rbp+var_18], eax
mov     eax, [rbp+var_18]
movsxd  rbx, eax
lea     rax, [rbp+var_40]
mov     rdi, rax          ; s
call    _strlen
cmp     rbx, rax
jz      short loc_4011C8

```

Traduciendo los segmentos, obtenemos la siguiente posible contraseña (separada por bloques)

wu{ftcop + r0_0d_ps + _r0_0d_p + }70n_0d

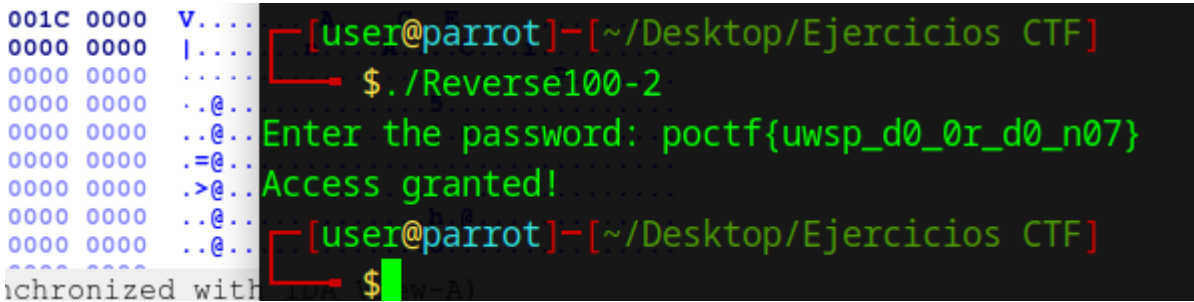
Me di cuenta que la flag esta invertida por lo que acomode cada bloque y quedó de la siguiente forma:

poctf{uwusp_d0_0rp_d0_0r_d0_n07}

Sin embargo, al escribir el tercer bloque en la pila sobrescribe una parte del segundo bloque, por lo que al final la posible contraseña es la siguiente:

poctf{uwusp_d0_0r_d0_n07}

Probando en el binario, confirmamos que es correcta:



Puntos antes de subir la flag:

2237	b_q	0
2238	Firefoxes	0
2239	PwnLA	0

Reverse 100 - End of the Line

100 Points

I got into CTFs a long time ago. Despite that, I remember how tough it was. It seemed impossible to solve all of the challenge. Heck, it seemed impossible to solve one. I remember how intimidating it was, but I also remember how amazing it felt to solve my first challenge and submit my first flag.

Looking back, it wasn't a tough challenge, but it was a start. Well, here's your chance. This is a recreation of the very first CTF challenge I solved. If you manage to solve this one, just remember that this is where I started, too.

Right Click, Save As... [Reverse100-2](#)
MD5 checksum 36C41D981AFBF5965FAFBF9ABE4757DC

Answer

poctf{uwusp_d0_0rp_d0_n07}

Submit

Congratulations, 100 points awarded!

Challenges

Crack

Crypto

Exploit

Forensics

Misc

OSINT

Reversing

Stego

Web

100 points

188
Reverse 100 - End of the Line
Solved
Reversing

373
Reverse 100 - Understanding Nonsense
Reversing

110
Reverse 200 - Planned Obsequence
Reversing

Puntos despues de subir la flag:

1170	Firefoxes	100
1171	Test Dummies	1

Reto 2 - Reverse 200 - Planned Obsequence

Categoría: Reversing

Descripción: In the lower-level RE challenges, I typically leave the flag or an encoded version in the code for participants to extract. This time, I've redacted the flag in the original code. If you look, you'll see it there. But don't worry, the obfuscated flag is there, buried in a variable somewhere. If you want to get it back, find the encoded flag and reverse the encoding process. It's a two-step process before you convert it back to ASCII so you will know if you're on the right track after just a few characters.

Right Click, Save As... [Reverse200-1](#)


MD5 checksum 2A51B3362A4813567E757B57AAC6BE4E

Resolución Reto 2

Similar al reto anterior, descargue el adjunto y lo ejecuté a ver que mostraba

```
[federico@parrot]-[~/Desktop/Ejercicios CTF]
$ ./Reverse200-1
Obfuscated Flag (Hex): 73 0b 22 21 1e 1c 11 22 21 73
[federico@parrot]-[~/Desktop/Ejercicios CTF]
```

Devuelve una flag ofuscada así que opte por probar decompilar el binario utilizando <https://dogbolt.org> (un decompilador online que utiliza varios decompiladores y recupera el posible código en c) para buscar la flag y ver cómo está ofuscada.

 Decompiler Explorer

Upload File
Your file must be **less than 2MB** in size. Uploaded binaries [are retained](#).

☒ angr ☒ BinaryNinja ☐ Boomerang ☐ dewolf ☐ Ghidra ☐ Hex-Rays

☐ RecStudio ☐ Reko ☐ Relyze ☐ RetDec

angr 9.2.135
87 }
88 return;
89 }
90
91 long long deobfuscate(unsigned long a0)
92 {
93 unsigned long long v0; // [sp-0x50]
94 char v1; // [bp-0x48]
95 char v2; // [bp-0x1d]
96
97 v0 = a0;
98 strncpy(&v1, "2d383c313f2432302c2d08710870356e376f086d3f083b6c", 48);
99 strncpy(&v2, "83b6c713270262a", 15);
100 return 3688790469046842168;
101 }
102
103 void print_obfuscated_flag()
104 {
105 unsigned long long v0; // [bp-0x27]
106 unsigned int v1; // [sp-0x20]
107 unsigned int v2; // [sp-0x1c]
108
109 v0 = 8086c010347c08e5c7c;
110 }

Samples
Or check out one of these samples we've provided:

BinaryNinja 4.2.6455 (2c8da1e)
117
118 int64_t print_obfuscated_flag()
119 {
120 int64_t var_27;
121 builtin_strncpy(&var_27, "REDACTED*");
122 obfuscate(&var_27);
123 printf("Obfuscated Flag (Hex): ");
124 int32_t var_1c = 0;
125
126 while (var_1c < strlen(&var_27))
127 {
128 printf("%02x ", *(&var_27 + var_1c));
129 var_1c += 1;
130 }
131
132 return putchar(0xa);
133 }
134
135 int32_t main(int32_t argc, char** argv, char** envp)
136 {
137 print_obfuscated_flag();
138 return 0;
139 }

Note que hay una función llamada deobfuscate donde hay un string codificado que puede ser la posible flag.

Siguiendo la posible ejecución del código, encontré la función obfuscate que lo que hace es realizar un xor del string y luego sumarle 3 a cada carácter.

```
uint64_t obfuscate(char* arg1)
{
    int32_t var_1c = 0;
    uint64_t result;

    while (true)
    {
        result = strlen(arg1);

        if (var_1c >= result)
            break;

        arg1[var_1c] ^= 0x5a;
        arg1[var_1c] += 3;
        var_1c += 1;
    }

    return result;
}
```

Por lo que como siguiente paso fui a Cyberchef y probe aplicar el ofuscado en reversa (restarle 3 a cada carácter de la cadena y aplicar un xor).

Operations

Search...

Favourites

Data format

Encryption / Encoding

Public Key

Arithmetic / Logic

Set Union

Set Intersection

Set Difference

Symmetric Difference

Cartesian Product

Power Set

XOR

XOR Brute Force

Recipe

From Hex

Delimiter
Auto

SUB

Key
3

Subtract

Delimiter
Space

XOR

Key
0x5a

HEX

Scheme
Standard

☐ Null preserving

Input

2d 38 3c 31 3f 24 32 30 2c 2d 08 71 08 70 35 6e 37 6f 08 6d 3f 08 3b 6c 71 32 70 26 2a

Output

poc{uwsp_4_7h1n6_of_b34u7y}

Realizando esto logré revelar la flag oculta (poc{uwsp_4_7h1n6_of_b34u7y})

Puntos antes de subir la flag

1170	Firefoxes	100
1171	Test Dummies	1

Reverse 200 - Planned Obsequence

200 Points

In the lower-level RE challenges, I typically leave the flag or an encoded version in the code for participants to extract. This time, I've redacted the flag in the original code. If you look, you'll see it there. But don't worry, the obfuscated flag is there, buried in a variable somewhere. If you want to get it back, find the encoded flag and reverse the encoding process. It's a two-step process before you convert it back to ASCII so you will know if you're on the right track after just a few characters.

Right Click, Save As... **Reverse200-1**
MD5 checksum 2A51B3362A4813567E757B57AAC6BE4E

Answer

Submit

Congratulations, 200 points awarded!

Challenges

Crack

Crypto

Exploit

Forensics

Misc

OSINT

Reversing

Stego

Web

338

Reverse 100 - Well Said but Poorly Heard

Reversing

190

Reverse 100 - Understanding Nonsense

Reversing

375

Reverse 100 - Understanding Nonsense

Reversing

112

Reverse 200 - Planned Obsequence

Reversing

162

Reverse 200 - We Do It Live

Reversing

124

Reverse 300 - Separating the Firmament

Reversing

35

Reverse 300 - Think Different, Be Similar

Reversing

18

Reverse 300 - Beef-Witted Mushrumps

Reversing

Puntos despues de subir la flag:

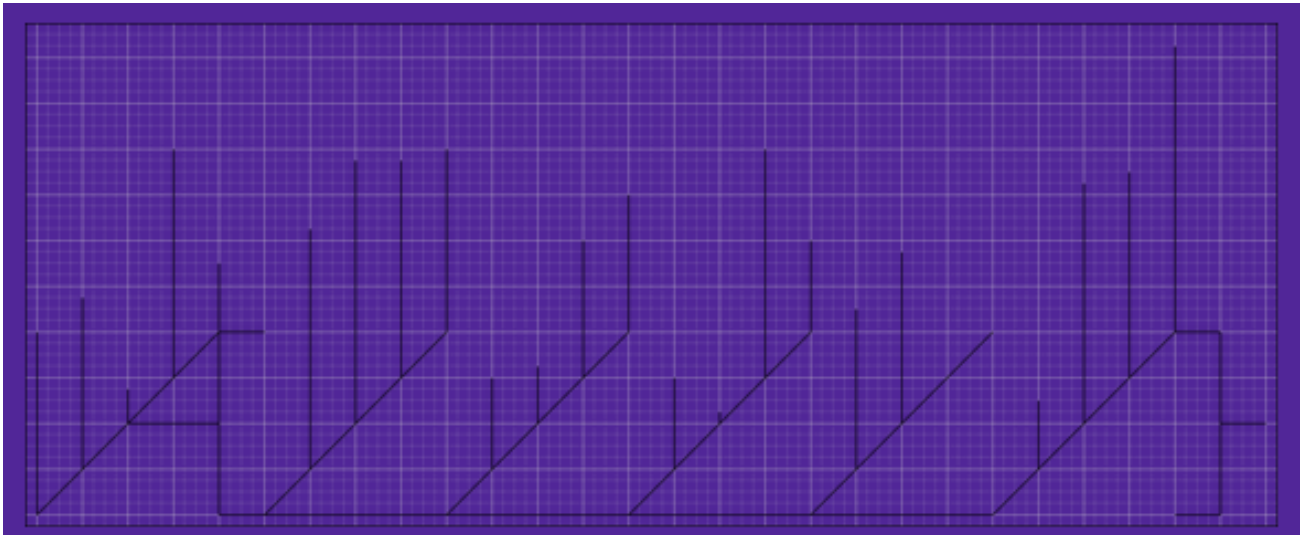
808	cool names	300
809	Firefoxes	300
810	ByteForce	200

Reto 3 - Misc 300 - Beneath the Pea Beneath the Pillow

Categoría: Misc

Descripción:As a kid in school we used to do something called "study hall" where you basically sat in the auditorium for an hour doing homework or whatever. I used to intentionally fail a course every year so I could test out of it within the first two weeks of the following semester and get an extra hour of study hall. Weird policy, I know, but I did it every year from middle school to the end of high school.

Well, that gave me plenty of time to sit there quietly reading or whatever, but mostly just staring off into space. The auditorium walls were brick, and I spent a lot of time just looking at them letting my imagination run wild. Eventually I devised ways to spell out secret messages using them. For example, see below. Don't forget to convert to leetspeak using the alphabet in the rules!



MD5 checksum 2e838ce450d3ca850b28d4b48f0f3931

Resolución Reto 3

Lo primero que note fueron las llaves de la flag que se pueden apreciar en la imagen, y teniendo en cuenta el formato de las flags del ctf (poctf{uwsp_message}) note que cada línea vertical formaba un carácter de la flag.

Si contamos la cantidad de cuadrados de longitud que tiene cada línea vertical, el resultado de la cuenta se corresponde con cada letra del abecedario

Por ejemplo: La primer linea da 16 , que corresponde a la letra p

Cada diagonal es una palabra, entonces contando la longitud de cada linea nos queda lo siguiente:

16,15,03,20,06 + 21,23,19,16 + 08,05,12,12 + 8,1,20,8 + 14,15 + 6,21,18,25

Decodificando la cadena en el cyberchef (y añadiendo los _ y las llaves correspondientes), nos queda la siguiente flag: **poctf{uwsp_hell_hath_no_fury}**

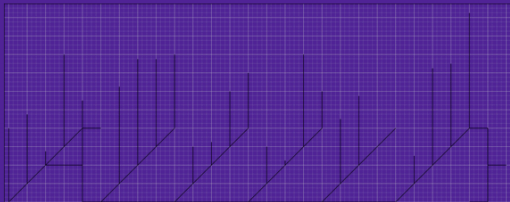
Aplicando 1337 a la flag queda de la siguiente manera: **poctf{uwsp_h3ll_h47h_n0_fury}**

Puntos antes de subir la flag:

808	cool names	300
809	Firefoxes	300
810	ByteForce	200

following semester and get an extra hour of study hall. Weird policy, I know, but I did it every year from middle school to the end of high school.

Well, that gave me plenty of time to sit there quietly reading or whatever, but mostly just staring off into space. The auditorium walls were brick, and I spent a lot of time just looking at them letting my imagination run wild. Eventually I devised ways to spell out secret messages using them. For example, see below. Don't forget to convert to leetspeak using the alphabet in the rules!



MD5 checksum 2e838ce450d3ca850b28d4b48f0f3931

Answer

Submit

Congratulations, 300 points awarded!

Challenges

Crack
Crypto
Exploit
Forensics
Misc
OSINT

Reversing
Stego
Web

148 Misc 100 - I Never Knew Glory Misc	133 Misc 100 - Could It Be the Winter Misc	167 Misc 100 - A Coward Dies Twice Misc	245 Misc 200 - My Name is Human Misc
23 Misc 200 - Embrace the Suck	70 Misc 200 - Anything Worth Doing Wrong	36 Misc 300 - The Hollow Howl of Foreign Winds	28 Misc 300 - With the Pea Br... the Pillow Solved

Puntos despues de subir la flag:

554	Scriptcrackers	600
555	Firefoxes	600
556	drinkwater	500

Conclusiones del primer CTF:

Este CTF me permitió ver cómo es un CTF por fuera de lo que se vio durante la cursada de esta materia, me resultó entretenido y emocionante ver tanta variedad de ejercicios (sobre todo en la categoría de misceláneos). A su vez, me permitió implementar técnicas vistas durante la cursada así como ver ejercicios del mismo tema pero con resoluciones distintas a lo que se hizo durante la materia (como el reto 1 donde parte de la pila se sobrescribía con la flag)

Segundo CTF:

Enlace a la pagina del CTF: <https://2025.knightctf.com/>

Enlace en CTFtime: <https://ctftime.org/event/2610/>

Perfil del CTF

Reto 1 - Knight's droid - 50 puntos

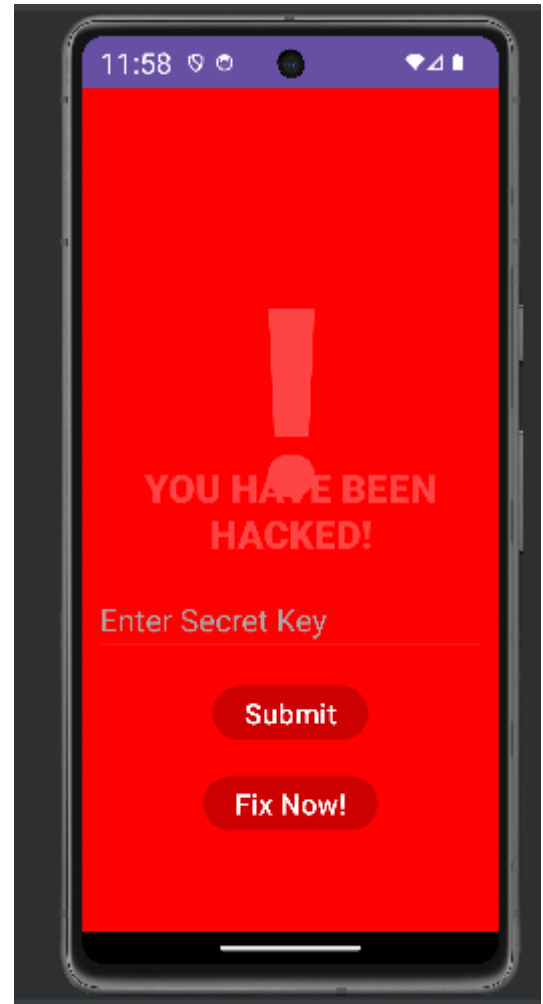
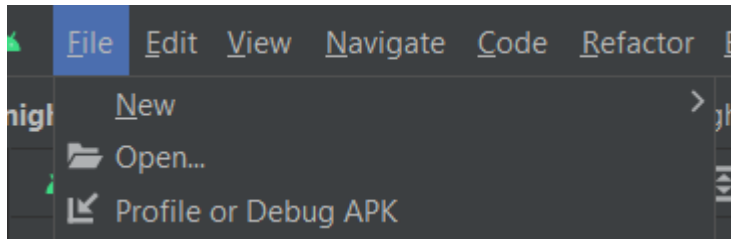
Categoría: Reversing

Descripción: For ages, a cryptic mechanical guardian has slumbered beneath the Knight's Citadel. Some say it holds powerful secrets once wielded by ancient code-wielding Knights. Many have tried to reactivate the droid and claim its hidden knowledge—yet none have returned victorious. Will you be the one to solve its riddles and awaken this legendary machine?

Adjunto: [knights_droid.zip](#)

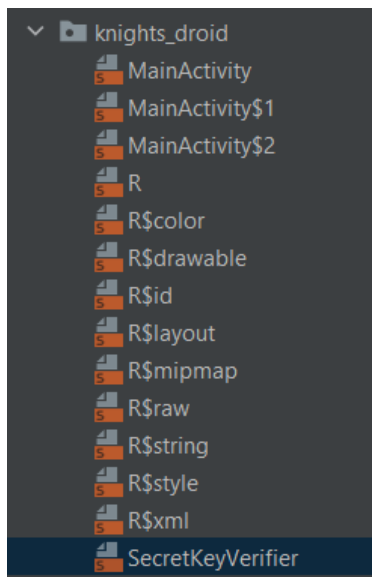
Resolución Reto 1

Para la resolución del ejercicio, lo primero que hice fue descargar el archivo adjunto, note que el archivo comprimido tenía dentro un .apk, por lo que utilice android studio para intentar ejecutarlo y entender qué hace utilizando la opción **Profile or Debug APK**

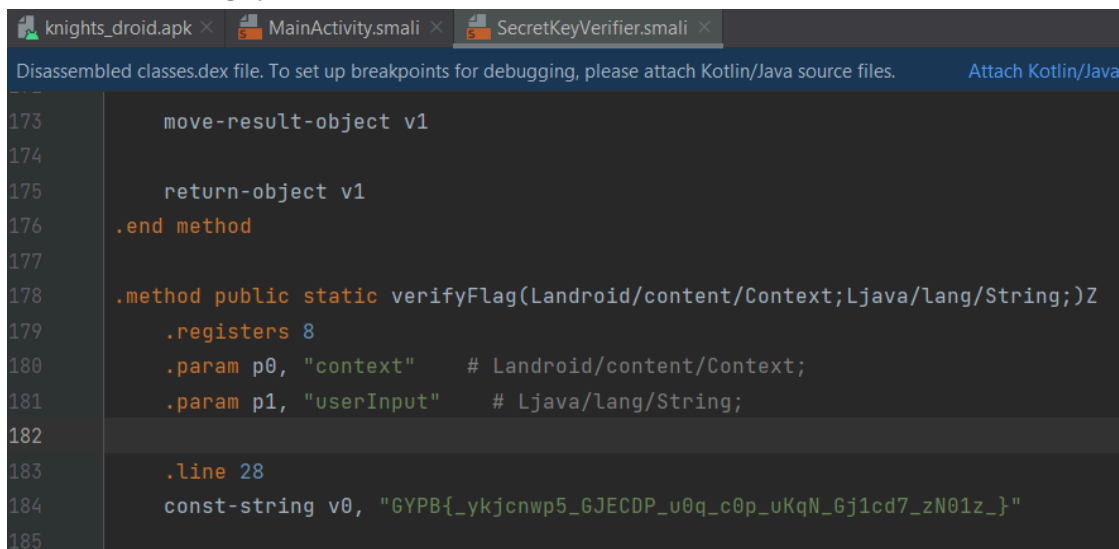


Como siguiente paso decidí revisar los archivos de la aplicación que generó Android Studio al abrir el APK (hace como un desmontado del .apk, lo cual es justo lo que necesitaba para resolver el ejercicio).

Entre los archivos, había uno llamado **SecretKeyVerifier**.



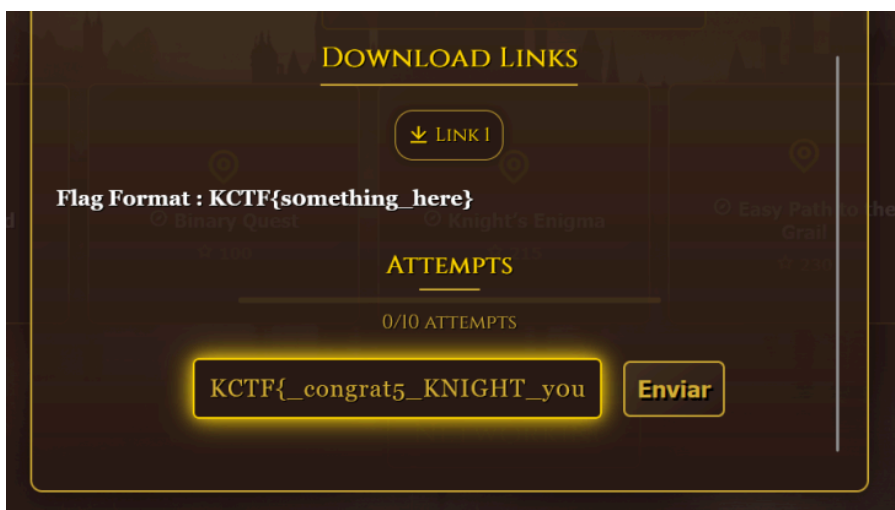
Analizando el contenido del archivo encontré lo siguiente que era sospechosamente similar al formato de la flag que estaba buscando



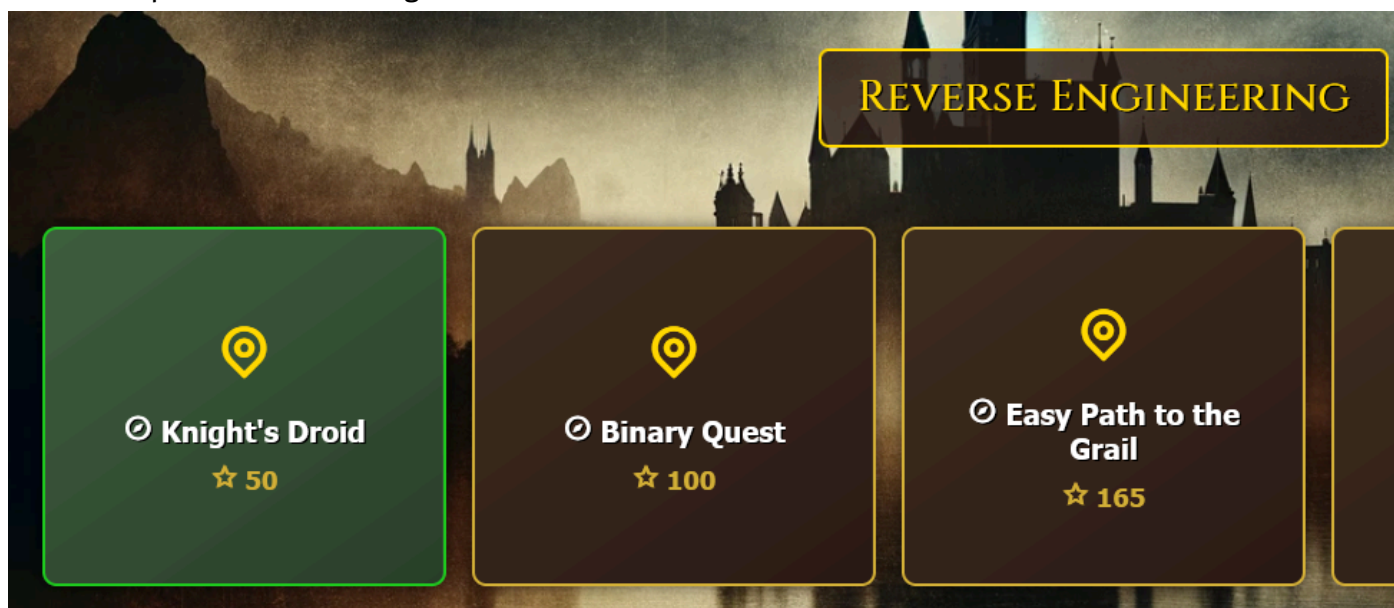
Claramente es un cifrado César de la flag buscada, así que fui al dcode a descifrar lo que decía.



La flag conseguida es entonces: **KCTF{_congrat5_KNIGHT_y0u_g0t_yOuR_Kn1gh7_dR01d_}**
Puntos antes de subir la flag:



Puntos despues de subir la flag:



Reto 2 - Worthy Knight - 215 puntos (al momento de la resolución)

Categoría: Reversing

Descripción: The gates of the Crimson Keep stand locked, sealed by cryptic runes from ages past. Many challengers have tested their might against these ancient wards—yet all were found wanting. Will you speak the correct incantation and earn the Keep's hidden treasures? Prove your valor and stand among legends... if you truly are a Worthy Knight.

Adjunto: [Worthy_Knight.zip](#)

Resolución Reto 2

Como en el reto anterior, descargue y descomprima el archivo adjunto, contenía un binario llamado **worthy.knight**, así que lo ejecute para ver que hace

```
[x]-[federico@parrot]-[~/Desktop/Ejercicios CTF]
$ ./worthy.knight

(Knight's Adventure)

      0
      <M>
      /W\
     ^   \|/
    /|\   |   *
   / \   / \  /|\
  /  \  /  \ /|\

~~~~~

Welcome, traveler. A mighty dragon blocks the gate.
Speak the secret incantation (10 runic letters) to continue.
```

Básicamente nos pide un string de 10 letras para avanzar (y probablemente obtener la flag), por lo que abrí el binario en el decompiler Explorer para entender mejor qué hace.

Observando el código decompilado encontré lo siguiente:

```
} while (pbVar8 != local_c8 + 10);
if ((byte)(local_c8[1] ^ local_c8[0]) == 0x24) {
    if (local_c8[1] == 0x6a) {
        if ((local_c8[2] ^ local_c8[3]) == 0x38) {
            if (local_c8[3] == 0x53) {
                local_10a = 0;
                pbVar8 = local_108;
                local_10c = local_c8._4_2_ << 8 | (ushort)local_c8._4_2_ >> 8;
                sVar6 = strlen((char *)&local_10c);
                MD5((uchar *)&local_10c, sVar6, pbVar8);
                pcVar5 = local_f8;
                do {
                    bVar1 = *pbVar8;
                    pcVar10 = pcVar5 + 2;
                    pbVar8 = pbVar8 + 1;
                    sprintf(pcVar5, "%02x", (ulong)bVar1);
                    pcVar5 = pcVar10;
                } while (local_d8 != pcVar10);
                local_d8[0] = '\0';
                iVar4 = strcmp(local_f8, "33a3192ba92b5a4803c9a9ed70ea5a9c");
                if (iVar4 == 0) {
                    if ((local_c8[6] ^ local_c8[7]) == 0x38) {
```

Estas son las condiciones que debe cumplir la cadena para ser correcta a partir del código encontrado:

- El char2 xor char1 = 0x24
- El char2 = 0x6a
- char 3 XOR char4 = 0x38
- char4 = 0x53
- char5 y char6 se rotan a la izquierda 8 bits y luego se realiza un encriptado md5 que da como resultado el hash **33a3192ba92b5a4803c9a9ed70ea5a9c**
- char7 XOR char8 = 0x38
- char8 = 0x61
- char10 XOR char9 = 0x20
- char10 = 0x69

Como las operaciones XOR son simétricas, podemos encontrar los valores de los caracteres que nos faltan

- char2 XOR 0x24 = char1 -> 0x6a XOR 0x24 = 0x4e
char1 = 0x4e
- 0x38 XOR char4 = char3 -> 0x38 XOR 0x53 = 0x6b
char3 = 0x6b
- 0x38 XOR char8 = char7 -> 0x38 XOR 0x61 = 0x59
char7 = 0x59
- char10 XOR 0x20 = char9 -> 0x69 XOR 0x20 = 0x49
char9 = 0x49
- Desencriptados el md5 con md5 decrypt, nos da **Tf**, los rotamos 8 bits a la derecha (el proceso inverso) y obtenemos **fT**

El resultado que tenemos hasta ahora es el siguiente: **4e 6a 6b 53 fT 59 61 49 69** que pasado todo a ASCII nos queda NjkSfTYaIi

Hora de probar!

```
Welcome, traveler. A mighty dragon blocks the gate.
Speak the secret incantation (10 runic letters) to continue.

Enter your incantation: NjkSfTYaIi

The kingdom's gates open, revealing the hidden realm...
      ( (
        \ \
      .--. ) ) .--.
    (   )/_/ (   )
    '  '      '  '

"Huzzah! Thy incantation is true. Onward, brave knight!"

The final scroll reveals your reward: KCTF{NjkSfTYaIi}
```

Puntos antes de subir la flag:

SQUAD: HUDSON KNIGHTS

TOTAL POINTS

50

NOBLE STANDING

516TH PLACE

DOWNLOAD LINKS

Binary Quest ☆ 100

Easy Path to the Grail ☆ 150

Knights Enigma ☆ 175

LINK 1

Flag Format : KCTF{something_here}

ATTEMPTS

0/10 ATTEMPTS

KCTF{NjkSfTYaIi}|

Enviar

Puntos después de subir la flag:

REVERSE ENGINEERING

Knights Droid ☆ 50

Binary Quest ☆ 100

Easy Path to the Grail ☆ 150

Knights Enigma ☆ 175

Worthy Knight ☆ 215



Reto 3 - Forward, Yet it falls back - 160 puntos

Categoría: Cryptography

Descripción: We discovered a peculiar string that appears standard but yields gibberish under normal decoding. Some analysts detect suspicious symmetry, hinting at reflection or an inverted dimension. Others suspect hidden block boundaries or a “backwards encoding,” while a few insist it's “rotated on a different axis.”

Our only clue: “Symbols may shift forward, but the key is often in reversing what we think is correct.”

Good luck peeling back the layers—sometimes you must step backward to see what’s right before you.

Adjunto: [Reverse.zip](#)

Resolución Reto 3

Comencé por descargar el archivo adjunto y ver que tenía dentro. Dentro del .zip había un txt que decía lo siguiente:

```
base32: G7G2DGQ5SY4DCK5YVUDRR0JI3U0AUUNTVR6XKDOKQ04CAAKK2MJA=====
```

```
key = 0123456789ABCDEF
```

```
iv = FEDCBA9876543210|
```

Teniendo en cuenta que la descripción mencionaba una simetría sospechosa y teniendo en cuenta que provee una key y un vector de inicialización opte por desencriptarlo mediante AES-128, ya que es un tipo de encriptado simétrico.

Entre a cyberchef y prepare la siguiente receta:

Recipe

From Base32

Alphabet

A-Z2-7=

☒ Remove non-alphabet chars

AES Decrypt

Key

3456789ABCD ...

UTF8 ▾

IV

FEDCBA98765 ...

UTF8 ▾

Mode

CBC

Input

Raw

Output

Raw

To Decimal

Delimiter

Space

☐ Support signed values

Input

G7G2DGQ5SY4DCK5YVUDRROJ13U0AUUNTVR6XKDOKQ04CAAKK2MJA===|

Output

}ED0C3D_3srev3R_3srev3R{FTCK

La llave y vector estaban en decimal, por lo que cambie el tipo de parámetro a UTF-8 en lugar de HEX (el valor por defecto) y modifique que la entrada es también de tipo Raw (ya que es la forma de output de decodificarlo de base32).

De esa forma obtuve lo que parece ser la flag invertida, por lo que la acomode y obtuve la siguiente flag:

KCTF{R3vers3_R3vers3_D3CODE}

Nota: A lo largo de lo que duraba el torneo el puntaje de los ejercicios fue variando, por lo que para el momento de resolver este ejercicio el reto 2 realizado bajo a 200 puntos, por lo que la puntuación final era al momento de resolver el reto 3 era de 250 puntos.

Puntos antes de subir la flag:

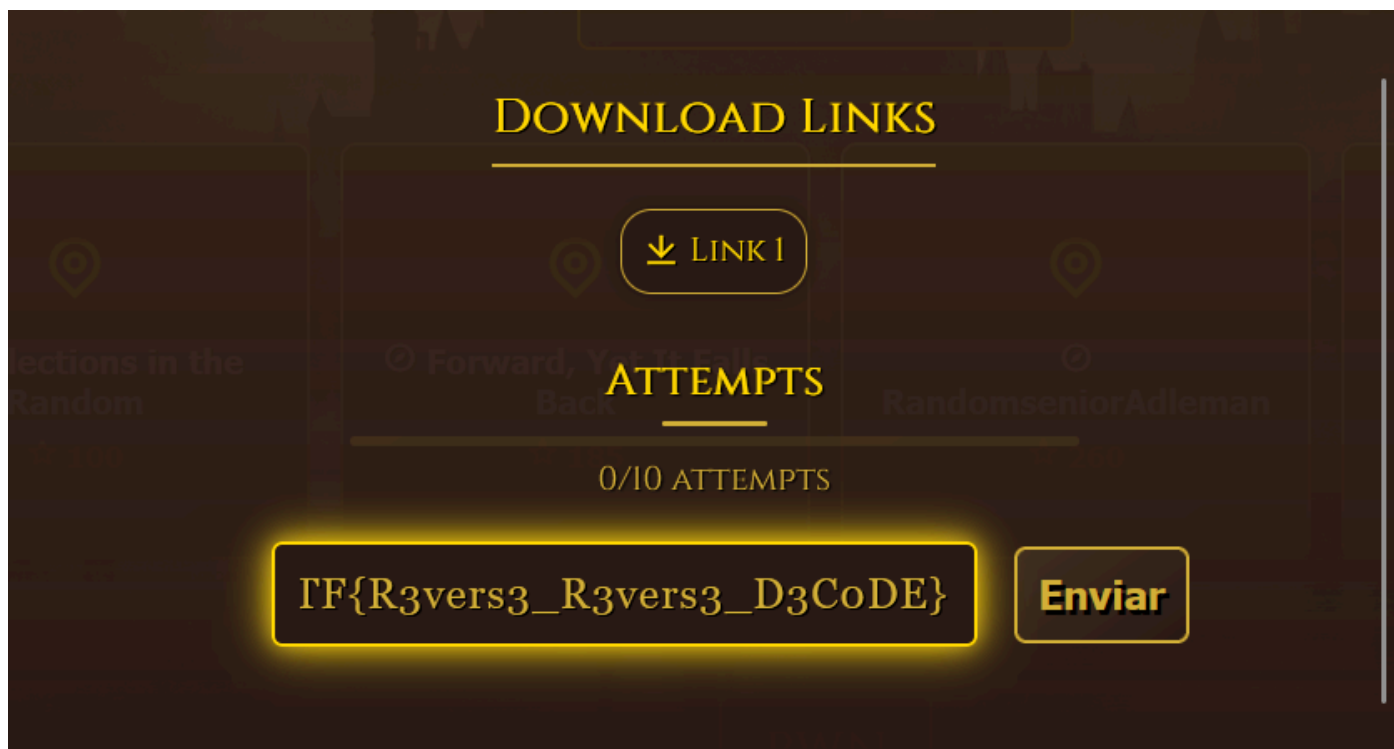
SQUAD: HUDSON KNIGHTS

TOTAL POINTS

250

NOBLE STANDING

424TH PLACE



Puntos despues de subir la flag:





Conclusiones del segundo CTF:

Disfruté más de este CTF que del primero que me anoté, ya que era muy entretenido ver ejercicios con una temática en particular (en este caso de caballeros). Todo el diseño de la página, los ejercicios y las descripciones lo hacían muy divertido.

Me pareció desafiante y al mismo tiempo emocionante los tipos de ejercicios de reversing que habían, ya que uno se basaba en desmontar un apk (algo que nunca había hecho) y el otro me hizo tomarme un tiempo largo para pensar y entender verdaderamente lo que el programa hacía.

Quedé muy satisfecho luego de haber participado de este ctf, porque como dije previamente los ejercicios divertidos y desafiantes me hicieron poner en práctica en profundidad el proceso de la ingeniería inversa y el cómo se aplica a todo tipo de archivos y no solo a binarios.

Quizás me hubiese gustado poder realizar más ejercicios de otras categorías (tanto en este como en el primer CTF), pero son temas en los que no poseo mucho conocimiento.

Me interesa profundizar por mi cuenta en las demás categorías que hay en estos campeonatos y en un futuro intentar participar con intención de quedar más arriba en la tabla de puntuaciones, o vivir la experiencia de participar de un CTF con un equipo.