

# Hydrogen

Se trata de parchear el binario para que ejecute el salto que nos de la pista que necesitamos para la contraseña. Se puede hacer con el debugger o directamente sobre las direcciones físicas. La flag está codificada de forma que no se pueda sacar con strings, hay una contraseña falsa para despistar.

## Solución

Ejecutamos con el debugger de radare2 (r2 -Ad hydrogen) y vemos la función main:

```
[0x7f9d8b2df100]> pdf @ main
; DATA XREF from entry0 @ 0x558955591121
327: int main (int argc, char **argv, char **envp);
; var int64_t var_59h @ rbp-0x59
; var int64_t var_58h @ rbp-0x58
; var int64_t var_54h @ rbp-0x54
; var int64_t var_50h @ rbp-0x50
; var int64_t var_40h @ rbp-0x40
; var int64_t var_38h @ rbp-0x38
; var int64_t var_30h @ rbp-0x30
; var int64_t var_28h @ rbp-0x28
; var int64_t var_20h @ rbp-0x20
; var int64_t var_18h @ rbp-0x18
; var int64_t var_8h @ rbp-0x8
0x558955591465      f30f1efa      endbr64
0x558955591469      55           push rbp
0x55895559146a      4889e5       mov rbp, rsp
0x55895559146d      4883ec60     sub rsp, 0x60
0x558955591471      64488b042528. mov rax, qword fs:[0x28]
0x55895559147a      488945f8     mov qword [var_8h], rax
0x55895559147e      31c0        xor eax, eax
0x558955591480      c745a8000000. mov dword [var_58h], 0
0x558955591487      837da800     cmp dword [var_58h], 0
0x55895559148b      741d        je 0x5589555914aa
0x55895559148d      488d3d800b00. lea rdi, str.Good__here_s_your_hint: ; 0x55895559148d
0x558955591494      b800000000   mov eax, 0
0x558955591499      e832fcffff   call sym.imp.printf ; int printf(const char *, ...)
0x55895559149e      b800000000   mov eax, 0
0x5589555914a3      e807feffff   call sym.show_hint
0x5589555914a8      eb0c        jmp 0x5589555914b6
0x5589555914aa      488d3d7f0b00. lea rdi, str.you_don_t_want_a_hint__Uhm..._OK
0x5589555914b1      e8eafbffff   call sym.imp.puts ; int puts(const char *, ...)
```

Se puede ver que en la posición 0x558955591480 se escribe el valor 0 en la variable var\_58h y justo debajo de comprueba si su valor es 0, si lo es se salta las líneas siguientes, entre las que vemos que se realiza una llamada a la función “show\_hint()”, por lo tanto tenemos que parchear para que no efectúe ese salto.

Hacemos seek sobre esa dirección y escribimos 0x9090 (son dos *nop*) para sustituir esos dos bytes (74d1) y que no haga nada:

```
[0x7f9d8b2df100]> s 0x55895559148b
[0x55895559148b]> wx 9090
```

Comprobamos que se ha reemplazado:

```
[0x55895559148b]> pdf @ main
; DATA XREF from entry0 @ 0x558955591121
327: int main (int argc, char **argv, char **envp);
; var int64_t var_59h @ rbp-0x59
; var int64_t var_58h @ rbp-0x58
; var int64_t var_54h @ rbp-0x54
; var int64_t var_50h @ rbp-0x50
; var int64_t var_40h @ rbp-0x40
; var int64_t var_38h @ rbp-0x38
; var int64_t var_30h @ rbp-0x30
; var int64_t var_28h @ rbp-0x28
; var int64_t var_20h @ rbp-0x20
; var int64_t var_18h @ rbp-0x18
; var int64_t var_8h @ rbp-0x8
0x558955591465 f30f1efa endbr64
0x558955591469 55 push rbp
0x55895559146a 4889e5 mov rbp, rsp
0x55895559146d 4883ec60 sub rsp, 0x60
0x558955591471 64488b042528. mov rax, qword fs:[0x28]
0x55895559147a 488945f8 mov qword [var_8h], rax
0x55895559147e 31c0 xor eax, eax
0x558955591480 c745a8000000. mov dword [var_58h], 0
0x558955591487 837da800 cmp dword [var_58h], 0
0x55895559148b 90 nop
..
0x55895559148d 488d3d800b00. lea rdi, str.Good__here_s_your_hint:
```

Ejecutamos y ya nos mostrará la pista:

```
[0x55895559148b]> dc
Good, here's your hint: uggcf://gurunpxrearjf.pbz/2021/03/fbynejvaqf-oyizr-vagrea-sbe-jrnz.ugzy
What's the password? █
```

Se puede ver facilmente que es una url codificada con rotación (es rotación 13):

The screenshot shows the CyberChef web application interface. The URL bar displays `https://gchq.github.io/CyberChef/#recipe=ROT13(true,true,...`. The main interface has a left sidebar with a list of operations: `rot`, `ROT13`, `ROT47`, `Rotate left`, `Rotate Image`, `Rotate right`, `Parse ObjectID timestamp`, and `Avro to JSON`. The `ROT13` operation is selected and expanded, showing settings for `Rotate lower case chars` (checked), `Rotate upper case chars` (checked), and `Rotate numbers` (unchecked) with a `Amount` of `13`. The `Input` field contains the URL `uggcf://gurunpxrearjf.pbz/2021/03/fbynejvaqf-oyizr-vagrea-sbe-jrnz.ugzy|`. The `Output` field shows the result: `https://thehackernews.com/2021/03/solarwinds-blame-intern-for-weak.html`. The interface also shows a download button for CyberChef, the last build date (13 days ago), and options for settings and support.

Entramos al enlace:



The screenshot shows a web browser window with the address bar displaying `https://thehackernews.com/2021/03/solarwinds-blame-intern-for-solarwinds123-password-lapse`. The article title is "SolarWinds Blames Intern for 'solarwinds123' Password Lapse", dated March 01, 2021, by Ravie Lakshmanan. The main image features the SolarWinds logo over a world map. A sidebar on the right contains an AT&T Business advertisement for "5G AND THE JOURNEY TO THE EDGE". The article text begins with: "As cybersecurity researchers continue to piece together the sprawling SolarWinds supply chain attack, top executives of the Texas-based software services firm blamed an intern for a critical password lapse that went unnoticed for several years."

Y se puede ver bastante claro qué contraseña va a ser, probamos “solarwinds123”:

```
[0x55895559148b]> dc
Good, here's your hint: uggcf://gurunpxrearjf.pbz/2021/03/fbynejvaqf-oynzr-vagrea-sbe-jrnrx.ugzy
What's the password?: solarwinds123
Oh, well... Here's your flag: moonCTF{0661de560d0fbe73ecf5d894e24f0982}
```

Y ya nos devuelve el flag.