Write-up: HackTheBox - Reversing - TearOrdear

Sebastián Sepúlveda @piblack
18 de mayo de 2019

Información: Categoria Reversing, puntuacion 20.

Descripción: Find the username and password and put them in the flag in the format: HTBusername:password.
Warning: It can produce false positives.

WRITEUP
Lo primero que realizamos al partir es un análisis estático 1 del programa `TearOrDear.exe` que está comprimido en el archivo .zip. Vemos que está escrito en .NET y es de 32 bits, por lo que ocupamos dnSpy_32bits para revisar el archivo binario.

```
root@retro:~/Descargas/teaordear# file TearORDear.exe
TearORDear.exe: PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
```

Figura 1. análisis estático con **file**

Lo primero que podemos hacer es ver cómo funciona el programa, así que lo corremos con dnSpy **(Ver Figura 2)**

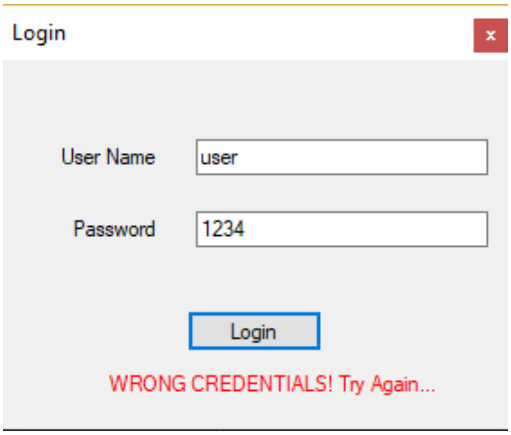

Figura 2. Login fallida en TearOrDear.exe

Al abrir dnSpy, comenzamos revisando todos los archivos de TearOrDear, sin mucho exito en la mayoria, pues no hay información relevante, hasta que en la pestaña `button1_Click` (**Ver Figura 3**), nos encontramos con **if(this.username == this.o && this.check1(s))**. Por cómo está escrito el código inferimos que es el chequeo que hace el programa para revisar si los parámetros del password son correctos o incorrectos **(Ver Figura 4)**

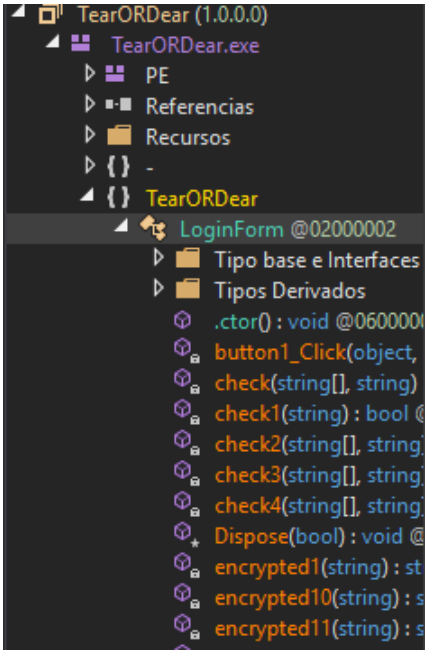

Figura 3. Ubicación de textttbutton1_Click

```
288
289         // Token: 0x0600000B RID: 11 RVA: 0x0000288C File Offset: 0x00000A8C
290         private void button1_Click(object sender, EventArgs e)
291         {
292             this.label_Result.Text = "";
293             this.kapa(sender, e);
294             this.pep = 0;
295             this.aa = this.Multiply(this.encrypted1(this.textBox_user.Text).Substring(0, 5), -1);
296             this.aa = this.aa.Remove(this.aa.Length - 1);
297             string s = this.Multiply(this.oura, -9);
298             if (this.username == this.o && this.check1(s))
299             {
300                 MessageBox.Show("Correct!");
301                 return;
302             }
303             this.label_Result.Text = "WRONG CREDENTIALS! Try Again...";
304         }
305
```

Figura 4. Contenido función textttbutton1_Click

Para que la condición sea verdadera, el *username*, en mi caso 1234, debe tener el mismo valor que la variable **o** y además la función **check1** debe devolver verdadero.

Al buscar el valor de **o** observamos que vale **roiw**!@# **(Ver Figura 4)**

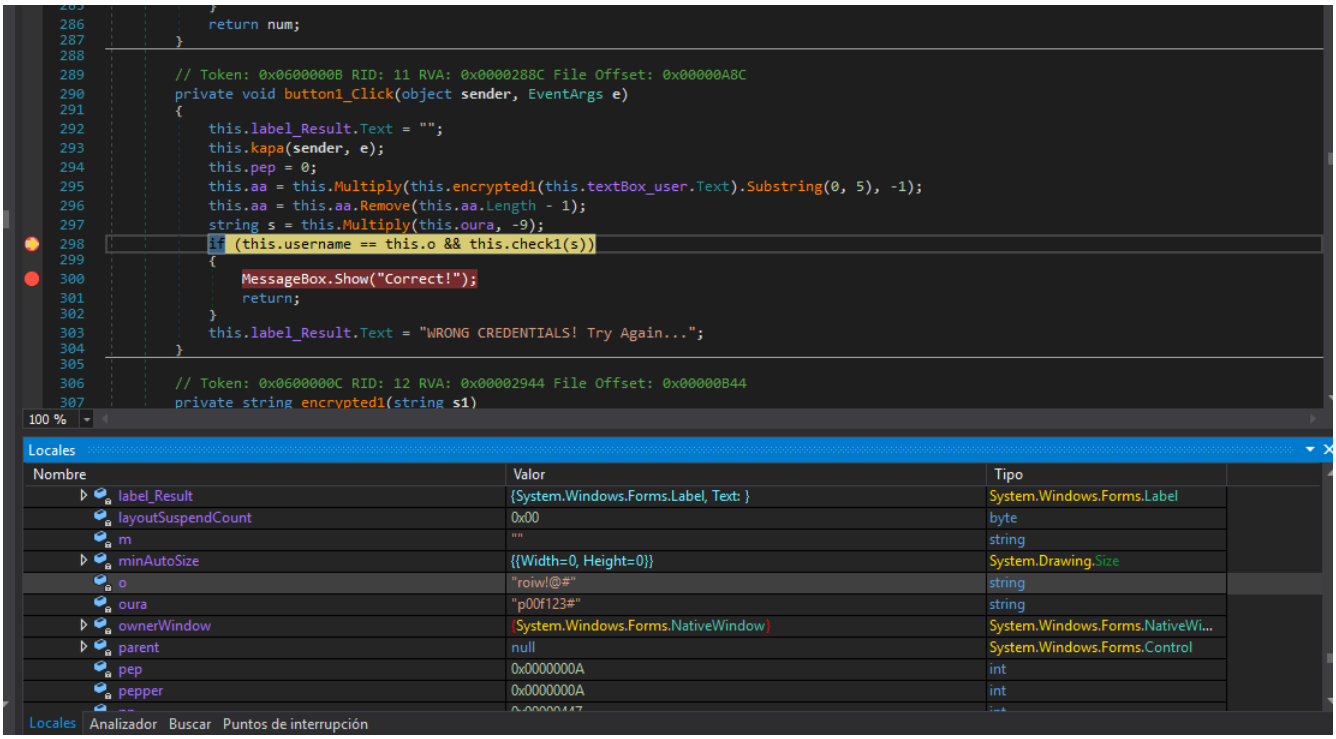

Figura 5. Resultado al generar una ruptura en la linea 298 del código, password

Luego de realizar lo anterior, revisamos la función **check** que devuelve verdadero si lo que hemos introducido en **textBox_user**, en nuestro caso *user* es igual a la variable **aa** y también si el **array[0]** es igual a **array[22]** que siempre es verdad pues ambos son "q".

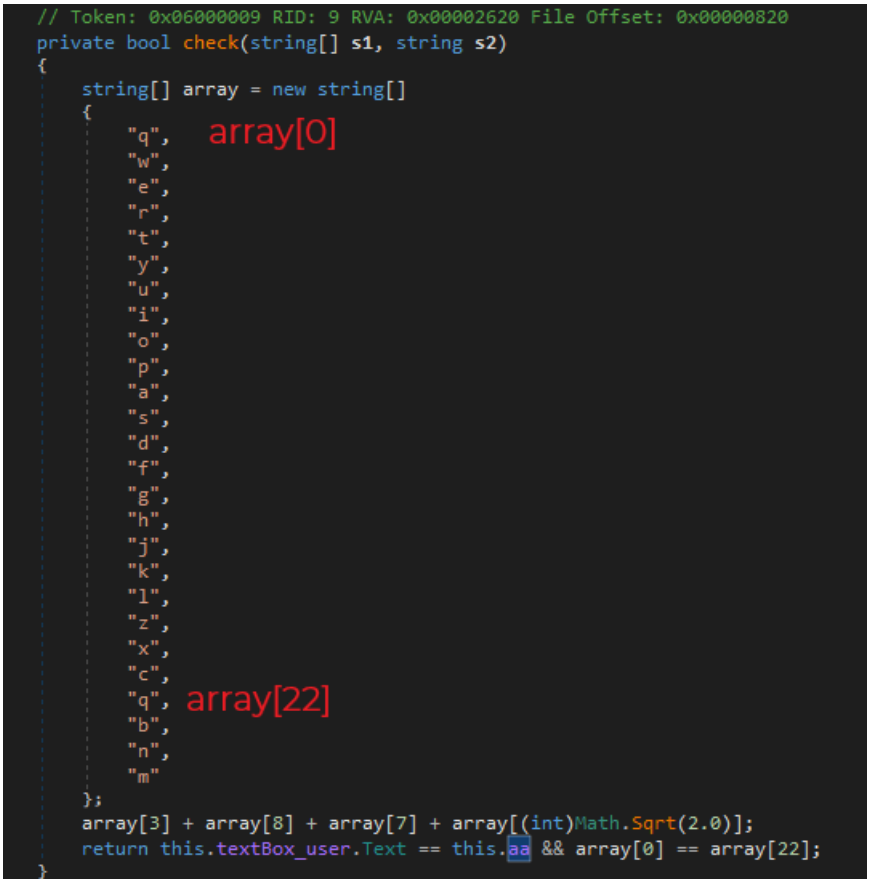

Figura 6. Función check con return

Al revisar el valor de **aa** (Ver Figura 7) vemos que el usuario debería ser igual a `piph`

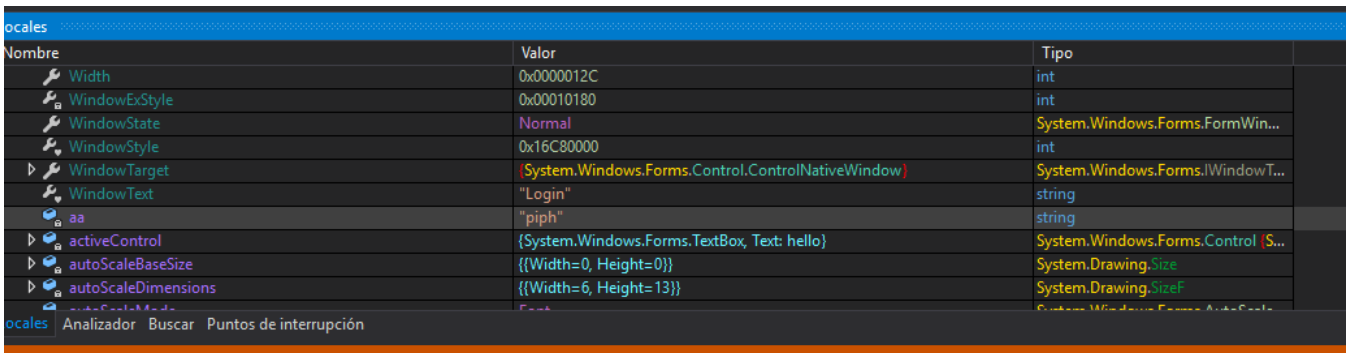

Figura 7. Resultado al generar una ruptura en la linea 298 del código, user

Comprobamos que estamos en lo cierto (Ver Figura 8)

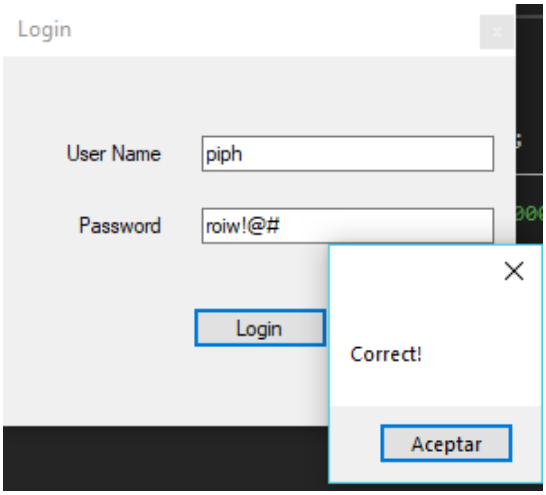

Figura 8. User y Password correctos

Como el flag es HTBusername:password, obtenemos el resultado.

Flag: HTB{piph:roiw!@#}