REVERSE ENINEERING

Root-me.org challenges

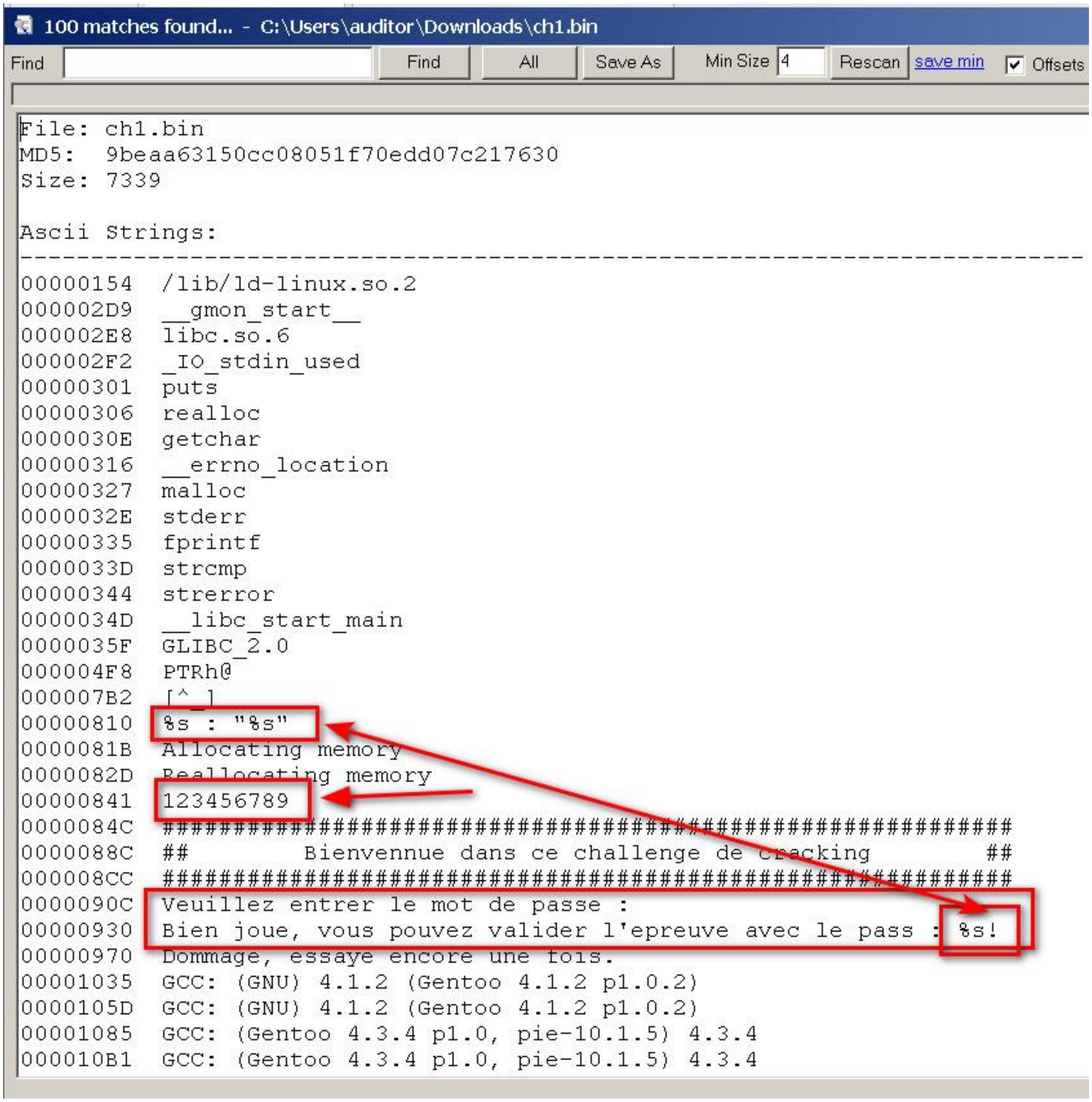
Inaam Kabbara

Table of Contents

1- ELF x86- 0 protection

First challenge is the ELF x86 from the root-me.org platform.

After downloading the file and by just trying to open the file in “strings” we can find something suspicious



100 matches found... - C:\Users\auditor\Downloads\ch1.bin

Min Size 4

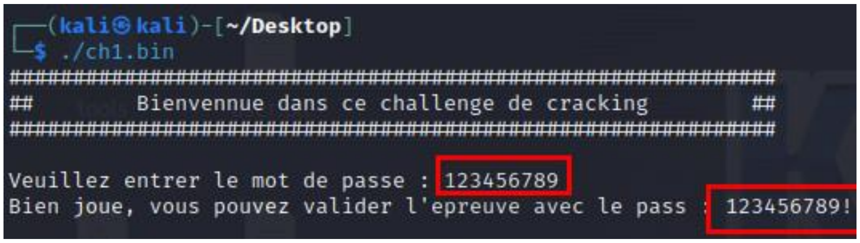
☑ Offsets

```
File: ch1.bin
MD5:  9beaa63150cc08051f70edd07c217630
Size: 7339

Ascii Strings:
---------------------------------------------------------------------------
00000154  /lib/ld-linux.so.2
000002D9  __gmon_start__
000002E8  libc.so.6
000002F2  _IO_stdin_used
00000301  puts
00000306  realloc
0000030E  getchar
00000316  __errno_location
00000327  malloc
0000032E  stderr
00000335  fprintf
0000033D  strcmp
00000344  strerror
0000034D  __libc_start_main
0000035F  GLIBC_2.0
000004F8  PTRh@
000007B2  [^_]
00000810  %s : "%s"
0000081B  Allocating memory
0000082D  Reallocating memory
00000841  123456789
0000084C  ############################################################
0000088C  ##        Bienvennue dans ce challenge de cracking        ##
000008CC  ############################################################
0000090C  Veuillez entrer le mot de passe :
00000930  Bien joue, vous pouvez valider l'epreuve avec le pass : %s!
00000970  Dommage, essaye encore une fois.
00001035  GCC: (GNU) 4.1.2 (Gentoo 4.1.2 p1.0.2)
0000105D  GCC: (GNU) 4.1.2 (Gentoo 4.1.2 p1.0.2)
00001085  GCC: (Gentoo 4.3.4 p1.0, pie-10.1.5) 4.3.4
000010B1  GCC: (Gentoo 4.3.4 p1.0, pie-10.1.5) 4.3.4
```

By reading the “strings” output we can find the printout of the file is asking for a password called initiated by %s.

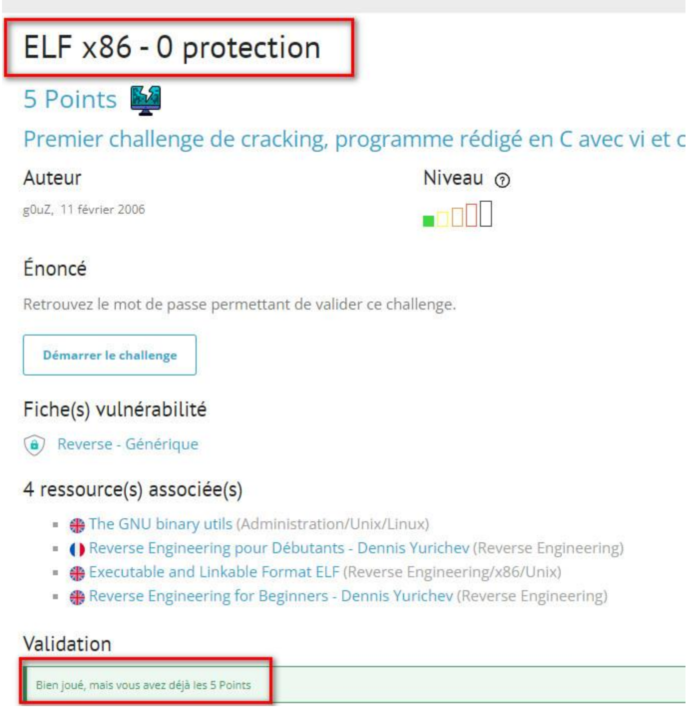
By reading all the lines we find a line containing “123456789”

By trying to run the file in kali and trying this string we can find that its correct:



```
┌──(kali㉿kali)-[~/Desktop]
└─$ ./ch1.bin
#############################################################
##        Bienvennue dans ce challenge de cracking        ##
#############################################################

Veuillez entrer le mot de passe : 123456789
Bien joue, vous pouvez valider l'epreuve avec le pass : 123456789!
```

And then by entering the password on the website and it is approved .



ELF x86 - 0 protection

5 Points

Premier challenge de cracking, programme rédigé en C avec vi et c

Auteur

g0uZ, 11 février 2006

Niveau

Énoncé

Retrouvez le mot de passe permettant de valider ce challenge.

Démarrer le challenge

Fiche(s) vulnérabilité

Reverse - Générique

4 ressource(s) associée(s)

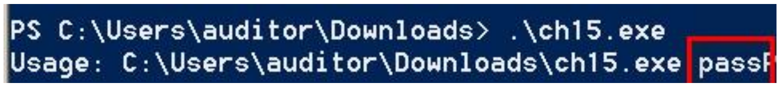
- The GNU binary utils (Administration/Unix/Linux)
- Reverse Engineering pour Débutants - Dennis Yurichev (Reverse Engineering)
- Executable and Linkable Format ELF (Reverse Engineering/x86/Unix)
- Reverse Engineering for Beginners - Dennis Yurichev (Reverse Engineering)

Validation

Bien joué, mais vous avez déjà les 5 Points

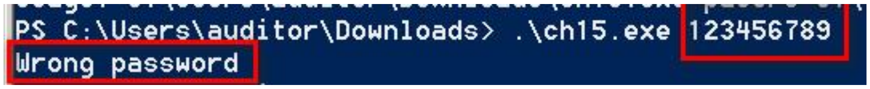
2- PE x86- 0 protection

By checking the file its an executable file so we tried to execute it in PowerShell:



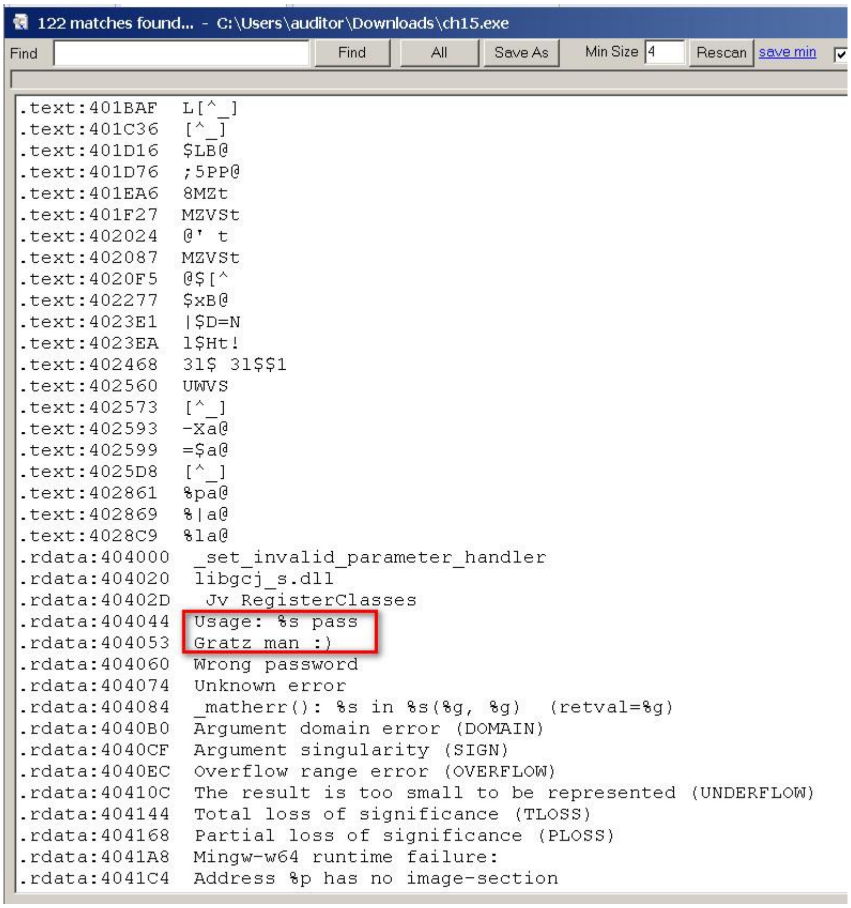
```
PS C:\Users\auditor\Downloads> .\ch15.exe
Usage: C:\Users\auditor\Downloads\ch15.exe pass
```

After executing it, it shows that it needs a password.



```
PS C:\Users\auditor\Downloads> .\ch15.exe 123456789
Wrong password
```

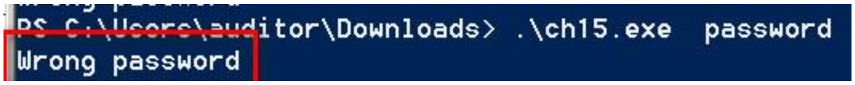
By trying the “123456789” password as an example, it shows “wrong password”.



```
.text:401BAF  L[^_]
.text:401C36  [^_]
.text:401D16  $LB@
.text:401D76  ;5PP@
.text:401EA6  8MZt
.text:401F27  MZVSt
.text:402024  @' t
.text:402087  MZVSt
.text:4020F5  @$[^
.text:402277  $xB@
.text:4023E1  |$D=N
.text:4023EA  l$Ht!
.text:402468  31$ 31$$1
.text:402560  UWVS
.text:402573  [^_]
.text:402593  -Xa@
.text:402599  =$a@
.text:4025D8  [^_]
.text:402861  %pa@
.text:402869  %|a@
.text:4028C9  %la@
.rdata:404000  _set_invalid_parameter_handler
.rdata:404020  libgcj_s.dll
.rdata:40402D   Jv RegisterClasses
.rdata:404044  Usage: %s pass
.rdata:404053  Gratz man :)
.rdata:404060  Wrong password
.rdata:404074  Unknown error
.rdata:404084  _matherr(): %s in %s(%g, %g)  (retval=%g)
.rdata:4040B0  Argument domain error (DOMAIN)
.rdata:4040CF  Argument singularity (SIGN)
.rdata:4040EC  Overflow range error (OVERFLOW)
.rdata:40410C  The result is too small to be represented (UNDERFLOW)
.rdata:404144  Total loss of significance (TLOSS)
.rdata:404168  Partial loss of significance (PLOSS)
.rdata:4041A8  Mingw-w64 runtime failure:
.rdata:4041C4  Address %p has no image-section
```

By reading the output from the Strings. It shows that the password will be located in %s.

Let’s try the password: “ password”

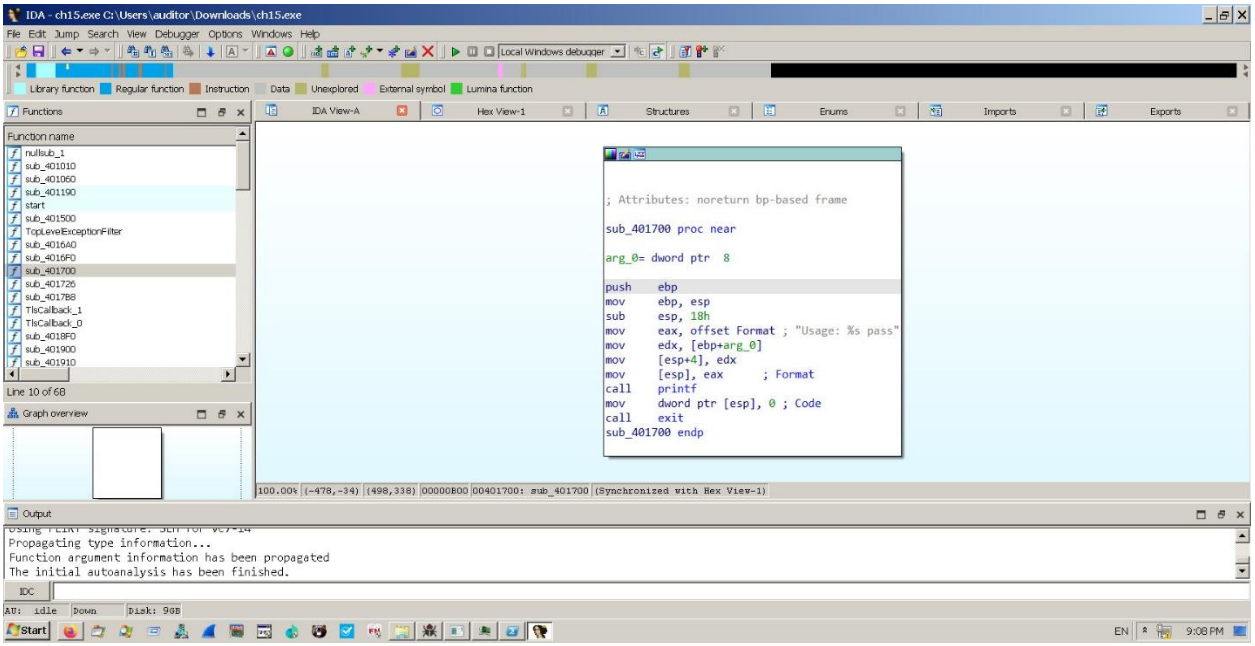


```
PS C:\Users\auditor\Downloads> .\ch15.exe  password
Wrong password
```

It also shows “wrong password”.

Let’s try to read the file in another program like IDA

Going through all the files to find where it configures the password.



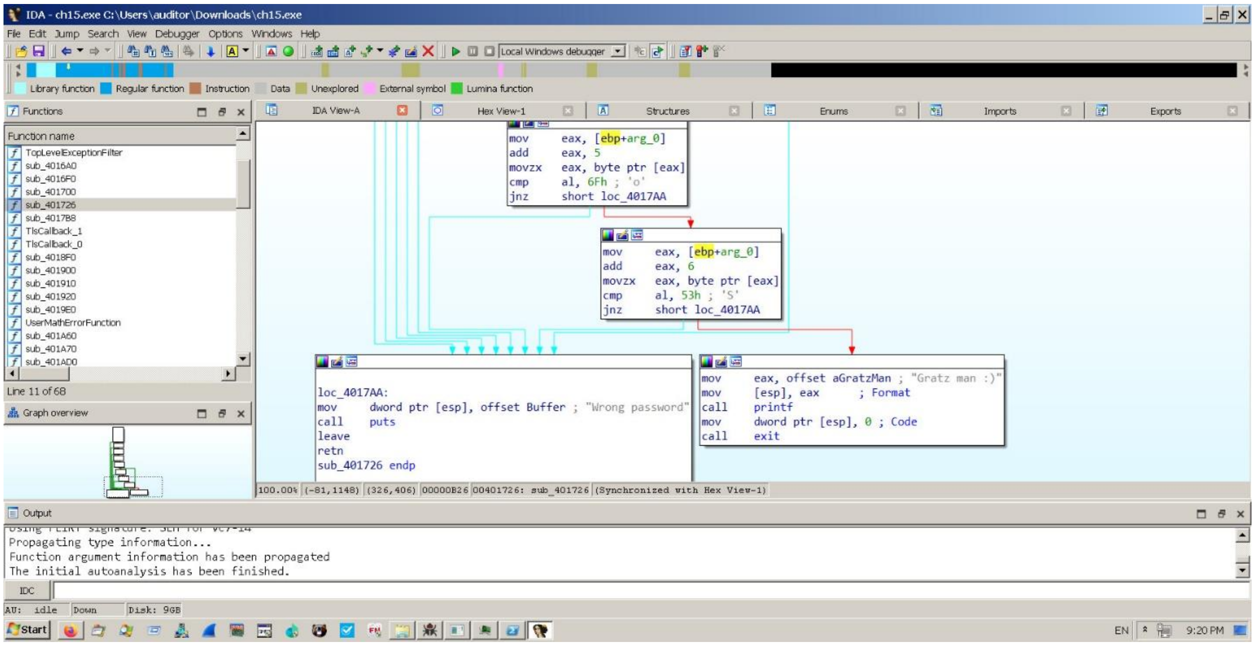
```
; Attributes: noreturn bp-based frame

sub_401700 proc near

arg_0= dword ptr  8

push    ebp
mov     ebp, esp
sub     esp, 18h
mov     eax, offset Format ; "Usage: %s pass"
mov     edx, [ebp+arg_0]
mov     [esp+4], edx
mov     [esp], eax      ; Format
call    printf
mov     dword ptr [esp], 0 ; Code
call    exit
sub_401700 endp
```

```
100.00% (-478,-34) (498,338) 00000B00 00401700: sub_401700 (Synchronized with Hex View-1)
```

```
Propagating type information...
Function argument information has been propagated
The initial autoanalysis has been finished.
```

```
AU:  idle   Down     Disk: 9GB
```

In the next screenshot we can see the file that reads the entered password and gives the output if its wrong or correct one.



```
mov     eax, [ebp+arg_0]
add     eax, 5
movzx   eax, byte ptr [eax]
cmp     al, 6Fh ; 'o'
jnz     short loc_4017AA
```

```
mov     eax, [ebp+arg_0]
add     eax, 6
movzx   eax, byte ptr [eax]
cmp     al, 53h ; 'S'
jnz     short loc_4017AA
```

```
loc_4017AA:
mov     dword ptr [esp], offset Buffer ; "Wrong password"
call    puts
leave
retn
sub_401726 endp
```

```
mov     eax, offset aGratzMan ; "Gratz man :)"
mov     [esp], eax      ; Format
call    printf
mov     dword ptr [esp], 0 ; Code
call    exit
```

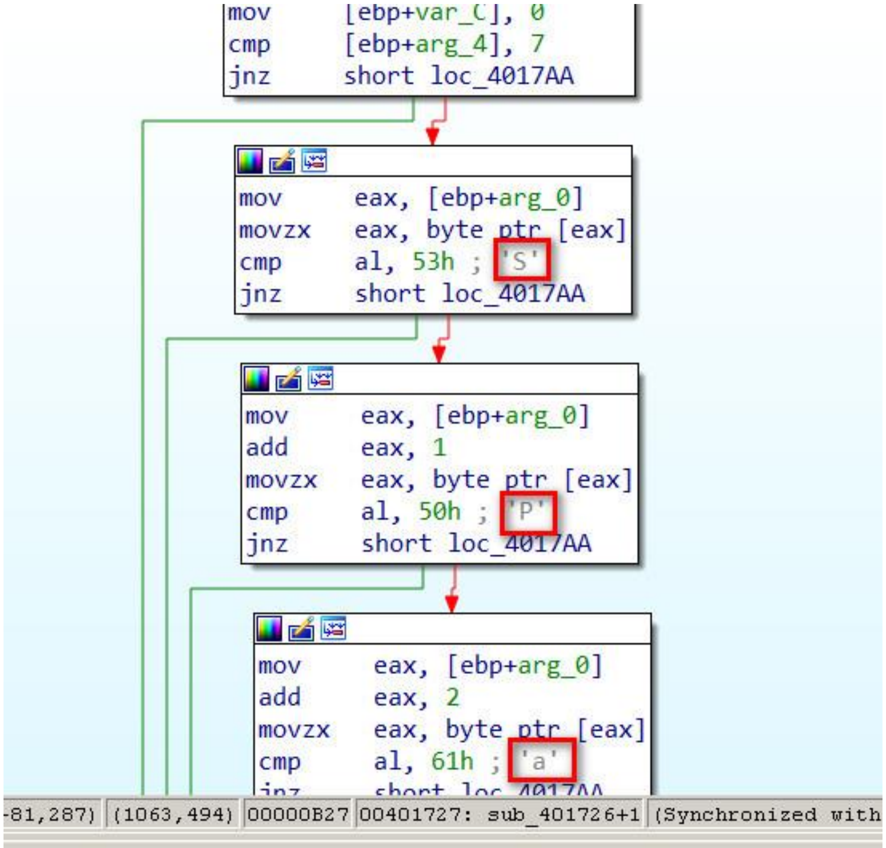
```
100.00% (-81,1148) (326,406) 00000B26 00401726: sub_401726 (Synchronized with Hex View-1)
```

```
Propagating type information...
Function argument information has been propagated
The initial autoanalysis has been finished.
```

```
AU:  idle   Down     Disk: 9GB
```

Reading this file deeply to understand how it works

It shows in each iteration there is a comment.



```
mov     [ebp+var_C], 0
cmp     [ebp+arg_4], 7
jnz     short loc_4017AA
```

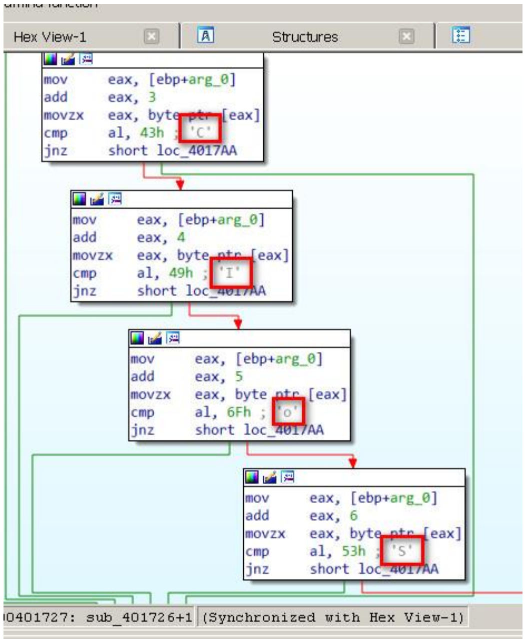
```
mov     eax, [ebp+arg_0]
movzx   eax, byte ptr [eax]
cmp     al, 53h ; 'S'
jnz     short loc_4017AA
```

```
mov     eax, [ebp+arg_0]
add     eax, 1
movzx   eax, byte ptr [eax]
cmp     al, 50h ; 'P'
jnz     short loc_4017AA
```

```
mov     eax, [ebp+arg_0]
add     eax, 2
movzx   eax, byte ptr [eax]
cmp     al, 61h ; 'a'
jnz     short loc_4017AA
```

```
-81,287) (1063,494) 00000B27 00401727: sub_401726+1 (Synchronized with
```

Let’s try to combine these comments and try it as a password



```
mov     eax, [ebp+arg_0]
add     eax, 3
movzx   eax, byte ptr [eax]
cmp     al, 43h ; 'C'
jnz     short loc_4017AA
```

```
mov     eax, [ebp+arg_0]
add     eax, 4
movzx   eax, byte ptr [eax]
cmp     al, 49h ; 'I'
jnz     short loc_4017AA
```

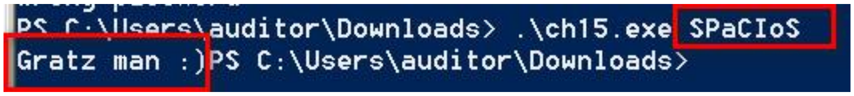
```
mov     eax, [ebp+arg_0]
add     eax, 5
movzx   eax, byte ptr [eax]
cmp     al, 6Fh ; 'o'
jnz     short loc_4017AA
```

```
mov     eax, [ebp+arg_0]
add     eax, 6
movzx   eax, byte ptr [eax]
cmp     al, 53h ; 'S'
jnz     short loc_4017AA
```

```
00401727: sub_401726+1 (Synchronized with Hex View-1)
```

So by combining these letters we will get “SPaCIoS”

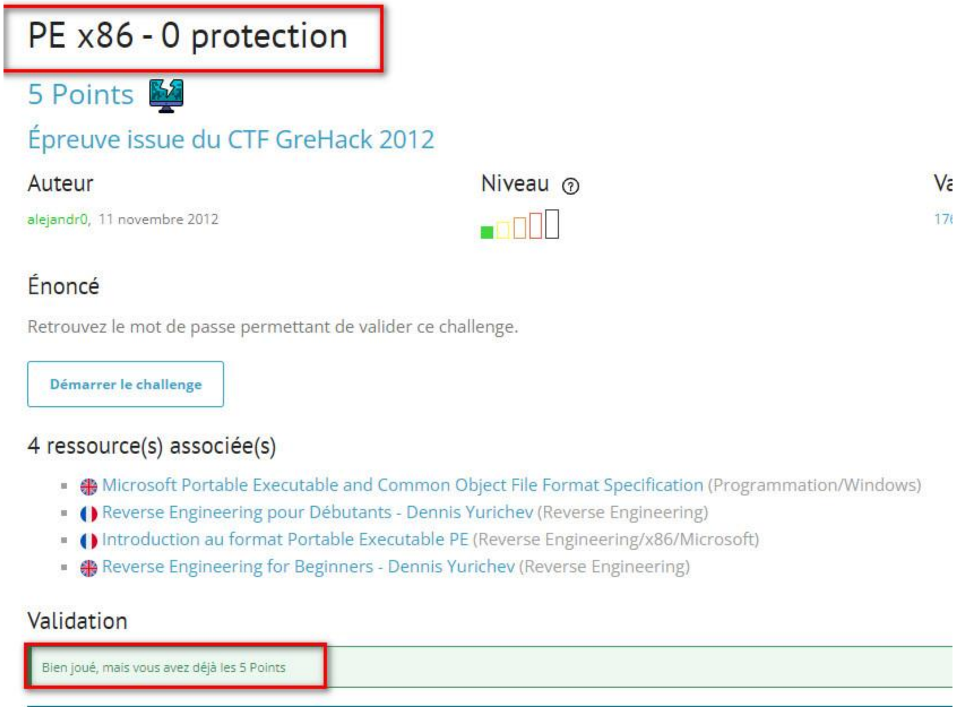
Let’s try it as a password:



```
PS C:\Users\auditor\Downloads> .\ch15.exe SPaCIoS
Gratz man :)PS C:\Users\auditor\Downloads>
```

We get it correctly.

By also trying it in the root-me website we can find that it’s the correct password:



PE x86 - 0 protection

5 Points

Épreuve issue du CTF GreHack 2012

Auteur

alejandr0, 11 novembre 2012

Niveau

Va

17

Énoncé

Retrouvez le mot de passe permettant de valider ce challenge.

4 ressource(s) associée(s)

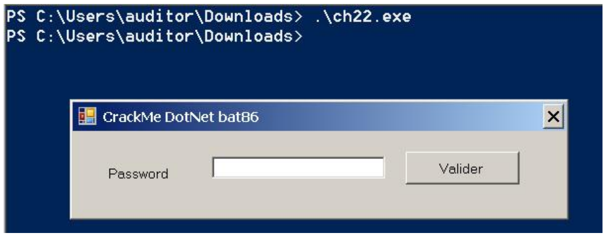
- Microsoft Portable Executable and Common Object File Format Specification (Programmation/Windows)
- Reverse Engineering pour Débutants - Dennis Yurichev (Reverse Engineering)
- Introduction au format Portable Executable PE (Reverse Engineering/x86/Microsoft)
- Reverse Engineering for Beginners - Dennis Yurichev (Reverse Engineering)

Validation

Bien joué, mais vous avez déjà les 5 Points

3- PE DotNet- 0 protection

This challenge has an executable file so let's try to execute it in PowerShell first to check what it will give:

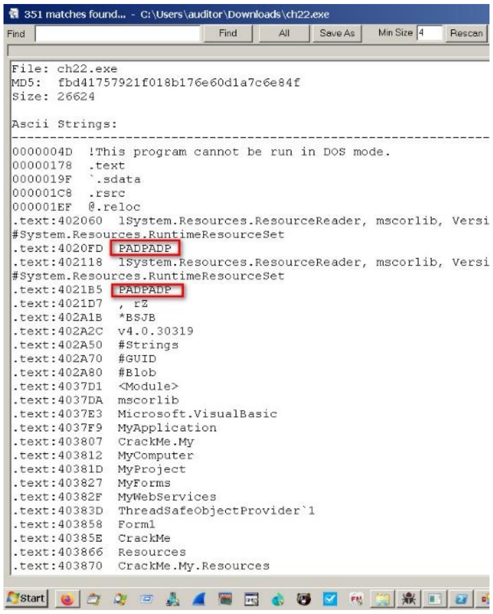


```
PS C:\Users\auditor\Downloads> .\ch22.exe
PS C:\Users\auditor\Downloads>
```

So this file is executed to ask for a password first.

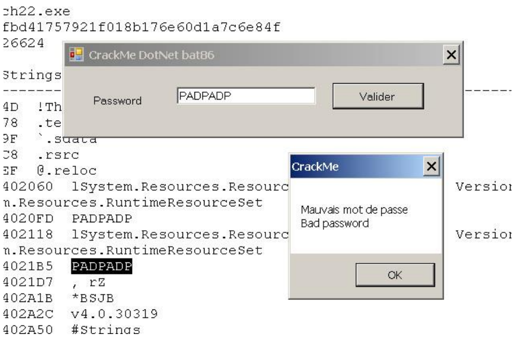
Let's try to find the password.

First by reading the strings output for this file.



```
File: ch22.exe
MD5:  fbd41757921f018b176e60d1a7c6e84f
Size: 26624

Ascii Strings:
---------------------------------------------------------------------------
0000004D  !This program cannot be run in DOS mode.
00000178  .text
0000019F  `.sdata
000001C8  .rsrc
000001EF  @.reloc
.text:402060  lSystem.Resources.ResourceReader, mscorlib, Versi
#System.Resources.RuntimeResourceSet
.text:4020FD  PADPADP
.text:402118  lSystem.Resources.ResourceReader, mscorlib, Versi
#System.Resources.RuntimeResourceSet
.text:4021B5  PADPADP
.text:4021D7  , rZ
.text:402A1B  *BSJB
.text:402A2C  v4.0.30319
.text:402A50  #Strings
.text:402A70  #GUID
.text:402A80  #Blob
.text:4037D1  <Module>
.text:4037DA  mscorlib
.text:4037E3  Microsoft.VisualBasic
.text:4037F9  MyApplication
.text:403807  CrackMe.My
.text:403812  MyComputer
.text:40381D  MyProject
.text:403827  MyForms
.text:40382F  MyWebServices
.text:40383D  ThreadSafeObjectProvider`1
.text:403858  Form1
.text:40385E  CrackMe
.text:403866  Resources
.text:403870  CrackMe.My.Resources
```

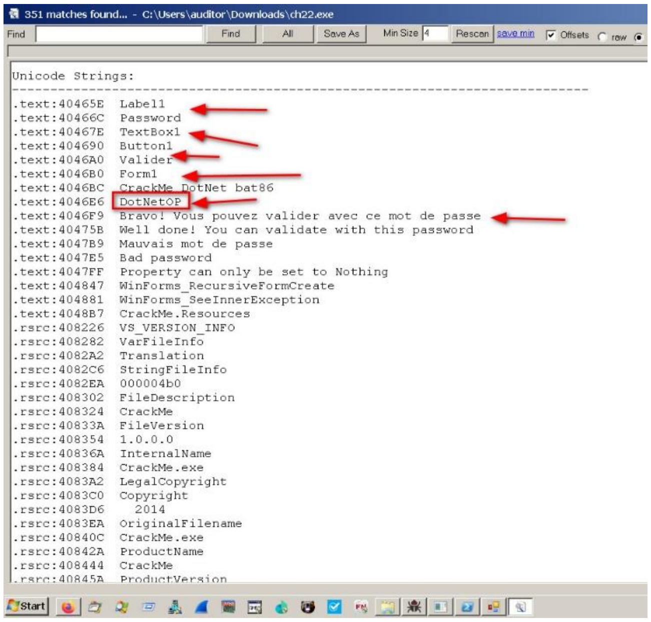
By observing the output we can see that there is a string repeated twice. Let's try it to see if it's the password.



Password PADPADP

Mauvais mot de passe
Bad password

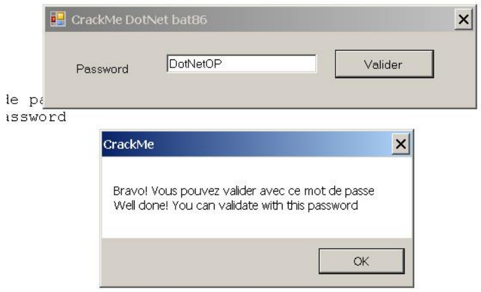
So it gives wrong password. By reading the whole output we find the Unicode strings for this file. By deeply reading the data from it we can find some important information.



```
Unicode Strings:
---------------------------------------------------------------------------
.text:40465E  Label1
.text:40466C  Password
.text:40467E  TextBox1
.text:404690  Button1
.text:4046A0  Valider
.text:4046B0  Form1
.text:4046BC  CrackMe DotNet bat86
.text:4046E6  DotNetOP
.text:4046F9  Bravo! Vous pouvez valider avec ce mot de passe
.text:40475B  Well done! You can validate with this password
.text:4047B9  Mauvais mot de passe
.text:4047E5  Bad password
.text:4047FF  Property can only be set to Nothing
.text:404847  WinForms_RecursiveFormCreate
.text:404881  WinForms_SeeInnerException
.text:4048B7  CrackMe.Resources
.rsrc:408226  VS_VERSION_INFO
.rsrc:408282  VarFileInfo
.rsrc:4082A2  Translation
.rsrc:4082C6  StringFileInfo
.rsrc:4082EA  000004b0
.rsrc:408302  FileDescription
.rsrc:408324  CrackMe
.rsrc:40833A  FileVersion
.rsrc:408354  1.0.0.0
.rsrc:40836A  InternalName
.rsrc:408384  CrackMe.exe
.rsrc:4083A2  LegalCopyright
.rsrc:4083C0  Copyright
.rsrc:4083D6    2014
.rsrc:4083EA  OriginalFilename
.rsrc:40840C  CrackMe.exe
.rsrc:40842A  ProductName
.rsrc:408444  CrackMe
.rsrc:40845A  ProductVersion
```

By reading these informations we can know that it displays how to build the windows that we saw before.

With small knowledge about programing we can know that they create a label called Label1 and they wrote "Password" in it. Then the TextBox1 is where the user will write the password. And then they put a button called "valider". Finally the whole form is Form1 with the title "CrackMe DotNet bat86". The suspicious string is "DotNetOP". Then we have the popup that gives the information either its correct password or wrong. So let’s try this string to check if it’s the password.

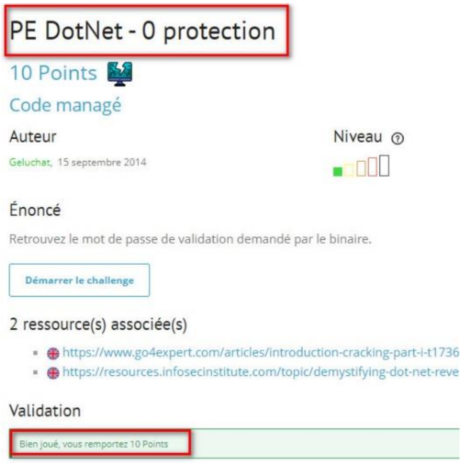


DotNetOP

Bravo! Vous pouvez valider avec ce mot de passe
Well done! You can validate with this password

Harray is the correct password.

And by also trying it on the website we can find its also the correct one.



PE DotNet - 0 protection

10 Points

Code managé

Auteur
Geluchat, 15 septembre 2014

Niveau

Énoncé

Retrouvez le mot de passe de validation demandé par le binaire.

2 ressource(s) associée(s)

- https://www.go4expert.com/articles/introduction-cracking-part-i-t1736
- https://resources.infosecinstitute.com/topic/demystifying-dot-net-reve

Validation

Bien joué, vous remportez 10 Points