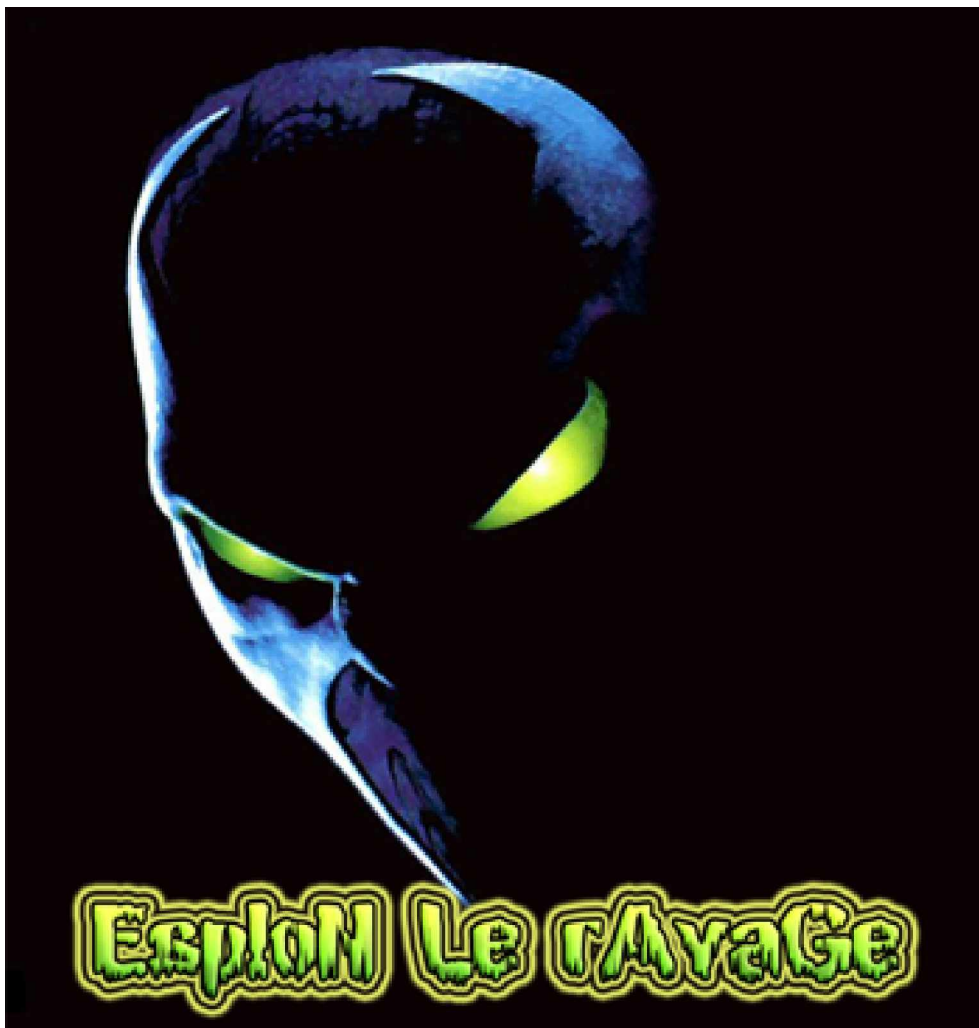


Cracker:  
"Xilisoft 3GP Video Converter 2.1.53 build-901b"

<http://www.xilisoft.com/>

---

Par:  
Esp!oN Le rAvaGe /CiM  
28/11/2005 - 16 :54



## Objectif:

- ⊕ Cracker le logiciel afin qu'il accepte tous les codes.

## Matos:

- ⊕ Windasm 8.93
- ⊕ Editeur hexadécimal.

## Niveau de difficulté:

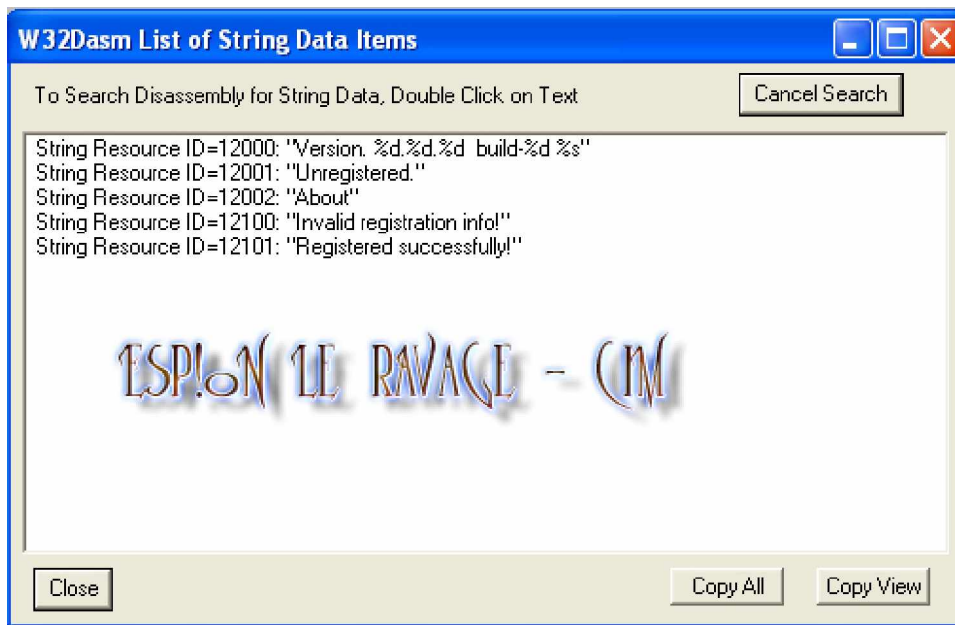
- ⊕ Moyen+

---

## Do it just for fun!

On s'attaque maintenant à un type différent (plus ou moins) de protection, car le fichier à cracker n'est pas l'exécutable lui-même, mais un fichier .dll qui va avec.

Le fichier qui nous intéresse est : " UI Lib71.dll ", faites une copie de ce fichier et désassemblez la, jetez un coup d'œil sur le listing :



Pour un listing on ne peut pas trouver mieux : 5 ligne avec 5/5 de trucs intéressants !!

Faites un double clique sur "Invalid registration info!", vous vous trouvez maintenant ici :

```

:1001858B E8D0F4FFFF          call 10017A60
* Reference To: UILib71.?IsValidRegInfo@ImRegDlg@SAHXZ
|
:10018590 E8BBF8FFFF          call 10017E50
:10018595 85C0                  test eax, eax
:10018597 7549                  jne 100185E2
:10018599 8B0DD4530310         mov ecx, dword ptr [100353D4]

* Possible Reference to String Resource ID=12100: "Invalid registration info!"
|
:1001859F 68442F0000         push 00002F04
:100185A4 8D442408         lea ecx, dword ptr [esp+8]
:100185A8 50                  push eax

```

Bonne nouvelle : Call/Test/Saut (ils sont vraiment gentils les éditeurs !!) ; c'est probable à 99% que le **10018597 7549 jne 100185E2** saute vers le bon message ; suivez le pour en avoir l'esprit clair :

```

* Referenced by a (U)nconditional or (C)onditional Jump at Address:
|:10018597(C)
|
:100185E2 8D8EF8030000         lea ecx, dword ptr [esi+000003F8]

* Reference To: MFC71.Ordinal:0F5E, Ord:0F5Eh
|
:100185E8 FF15D8260210         test dword ptr [0010308], al
:100185EE 84C0                  test al, al
:100185F0 7541                  jne 10018633

* Possible Reference to String Resource ID=12101: "Registered successfully!"

```

C'est exact, maintenant revenez d'où vous venez et entrez dans le call à l'adresse « 10018590 » :

```

* Referenced by a CALL at Addresses:
|:100044D4 , :10018590
|

Exported fn(): ?IsValidRegInfo@ImRegDlg@SAHXZ - Ord:015Ch
:10017E50 6AFF                push FFFFFFFF
:10017E52 68CF150210         push 100215CF
:10017E57 64A100000000       mov eax, dword ptr fs:[00000000]
:10017E5D 50                  push eax
:10017E5E 64892500000000     mov dword ptr fs:[00000000], esp
:10017E65 8B00000000         mov esi, 00000000
:10017E6B 8B00150210         mov ecx, dword ptr [10035160]
:10017E70 53                  push ebx
:10017E71 56                  push esi
:10017E72 8D4C2410         lea ecx, dword ptr [esp+10]
:10017E76 898424A8000000     mov dword ptr [esp+000000A8], eax

```

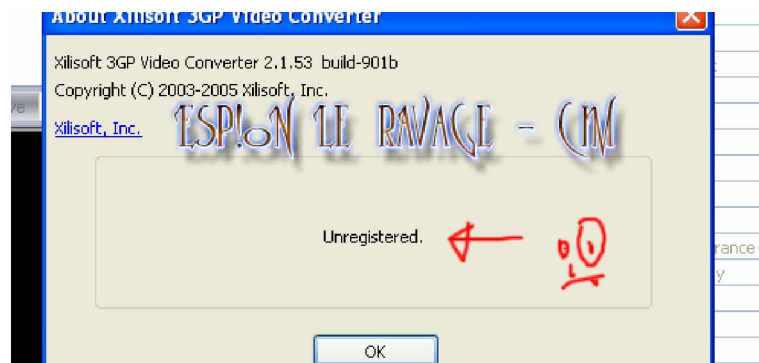
Remarquez (et rappelez vous aussi) d'abord que cette routine est appelée par DEUX call, mais ce n'est pas important à ce stade. Ce qu'il faut faire maintenant c'est de forcer ce call à ce terminer avant de faire la routine, le seul moyen et de mettre un 'ret' à la place du Push, donc notez l'offset de ce push (17E50) puis chargez votre fichier "UI Lib71.dll" (pas la copie mais le fichier d'origine, car la copie est déjà utilisée par windasm) ensuite changez 6A par C3, enregistrez puis testez :



Entrez un nom et n'importe quel serial, validez et ça marche :



Théoriquement on a terminé, mais c'est pas vraiment le cas : regardez le coté 'About' :



Merde alors !! Tout ce qu'on a fait c'est détourner le message d'erreur... mais il nous reste une petite astuce (hihihi... !) : vous vous rappelez du deuxième call ?

```

* Referenced by a CALL at Addresses:
|:100044D4 , :10018590 ←
|

Exported fn(): ?IsValidRegInfo@ImRegDlg@3SAHX2 - Ord:015Ch
:10017E50 6AFF          push FFFFFFFF
:10017E52 68CF150210    push 100215CF
:10017E57 64A100000000  mov eax, dword ptr fs:[00000000]
:10017E5D 50           push eax
:10017E5E 64892500000000  mov dword ptr fs:[00000000], esp
:10017E65 41000000     sub esp, 00000000
:10017E6B 8500100000   mov ecx, dword ptr [10035160]
:10017E70 53           push ebx
:10017E71 56           push esi
:10017E72 8D4C2410     lea ecx, dword ptr [esp+10]
:10017E76 898424A8000000  mov dword ptr [esp+000000A8], eax

```

Allez y jetez un coups d'œil (adresse: 100044D4) :

```
* Reference To: UILib71.?IsValidRegInfo@ImRegDlg@@SAHXZ
|
:100044D4 E877390100 call 10017E50
:100044D9 85C0 test eax, eax
:100044DB 7562 jne 1000453F

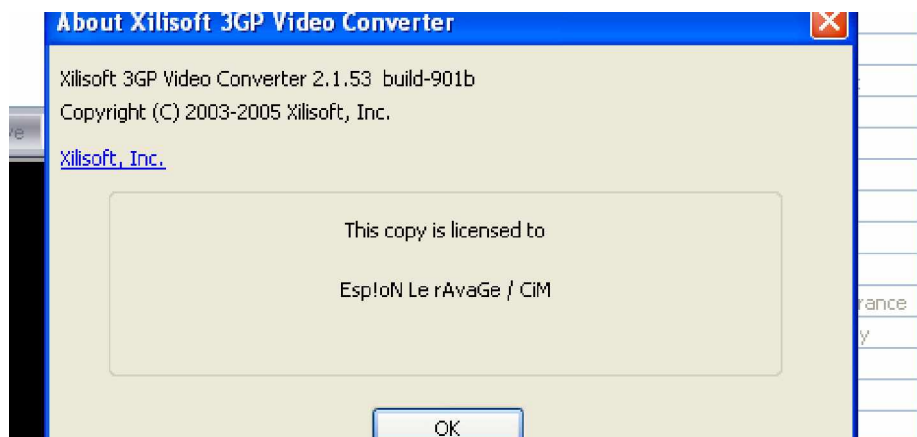
* Possible Reference to String Resource ID=12001: "Unregistered." ← 😊
|
:100044DD 68E12E0000 push 00002EE1
:100044E2 8D4C2418 mov ecx, dword ptr [esp+18]
:100044E6 51 push ecx
:100044E7 8B0DD4530310 mov ecx, dword ptr [100353D4]
* Reference To: UILib71.?GetString@ImLanguage@@QBE?AV?#CStringT@DV?#StrTraitMFC_DLL@I
```

Et surprise : Le 'Unregistered' apparaît !! Donc le ret qu'on a mis à la place du push pour détourner le message d'erreur influence aussi l'apparition de cet 'Unregistered' seule solution : empêcher ce call d'avoir lieu ! Pour faire, notez son offset (44D4) et changez à l'aide de votre éditeur hexadécimal : E877390100 par 9090909090 (90 = nop = pas d'opérations) ; enregistrez et testez... pas grand-chose : dans About on voit toujours 'Unregistered'.

Le seul suspect qui reste c'est le jne :

```
:100044DB 7562 jne 1000453F
```

Si ce saut ne se fait pas, le programme continu sa route et nous affiche « Unregistered », que faut-il faire à votre avis ? IL FAUT A TOUT PRIS QU'IL AIE LIEU CE SAUT !!! Comme cela le programme sautera le 'Unregistered' ; notez son offset (44DB) changez 75 par EB (EB= JMP = saute toujours !) ; enregistrez et testez, voilà le résultat :



Maintenant on peu dire que le programme est cracké.

## Recapitulatif:

Changements effectués:

✚ 6A	par C3	(offset 17E50)
✚ E877390100	par 9090909090	(offset 44D4)
✚ 75	par EB	(offset 44DB)

---

انتهى والحمد لله

EsploN Le rAvaGe/CiM Le 28/11/2005 -- 18 :25