

1Click DVD to Divx Avi V. 1.21

<http://www.dvdtoavi.net/>

1. Target analysis

After install it use PeiD to know more about this target (if this is packed or about some crypto signature):

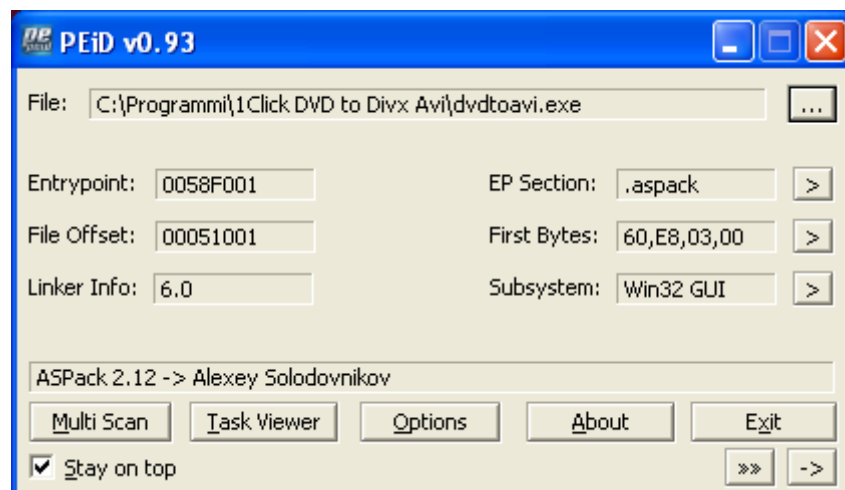


Fig. 1

Well try to unpack it with Stripper V2.07f, it succed, now scan again the target in order to know the compiler and check about some signature:

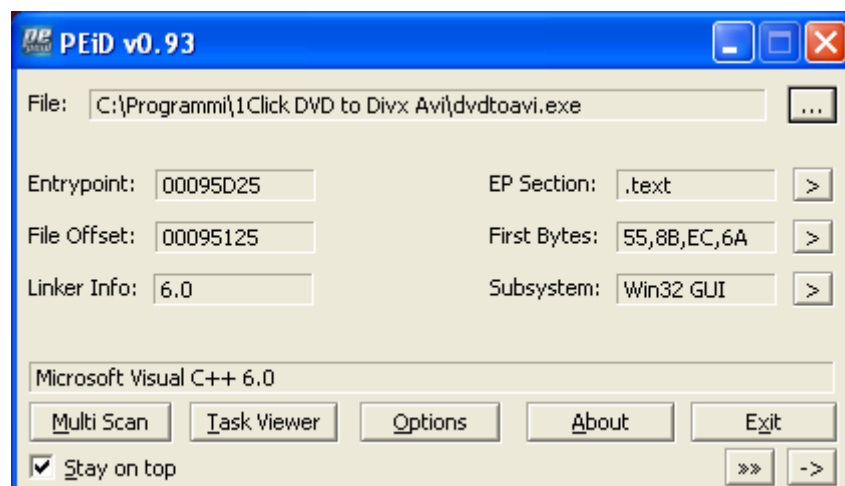


Fig. 2

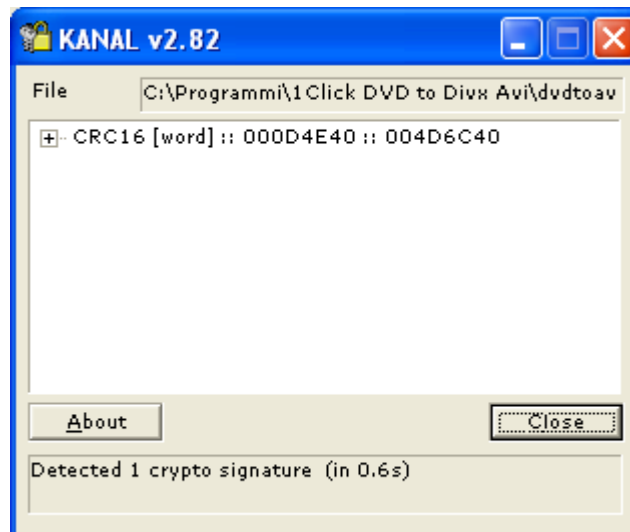


Fig. 3

Launch the unpacked target, it work well then no CRC about unpacking protection was added. Now is time to step into the code and patch it to keep full registration (plase note also that tut is pointed to show how do the serial code reversing then no more info will be write on this, simple, manual unpacking).

2. Cracking stage

Load target in OllyDbg then right click and select Search for -> All referenced text strings:

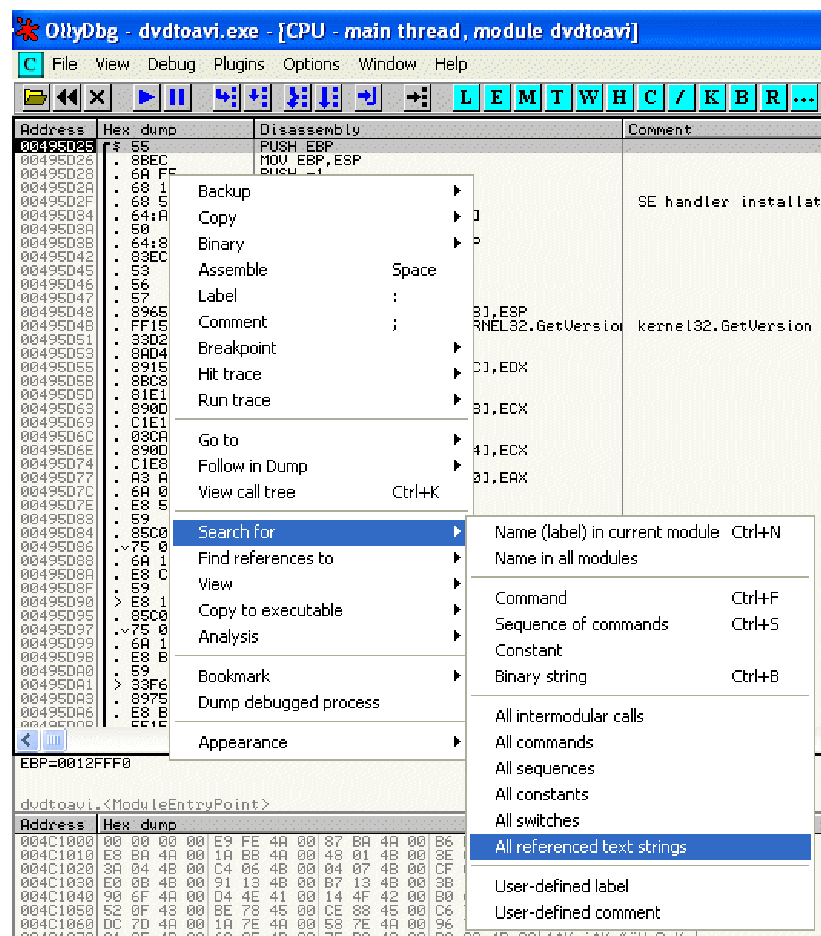


Fig. 4

Now scroll to the top and then right click to perform a text search, we have to know if there are some string referred to registration, typical string to search can be Thank, you, serial or similar stuff, well start with Thank string.

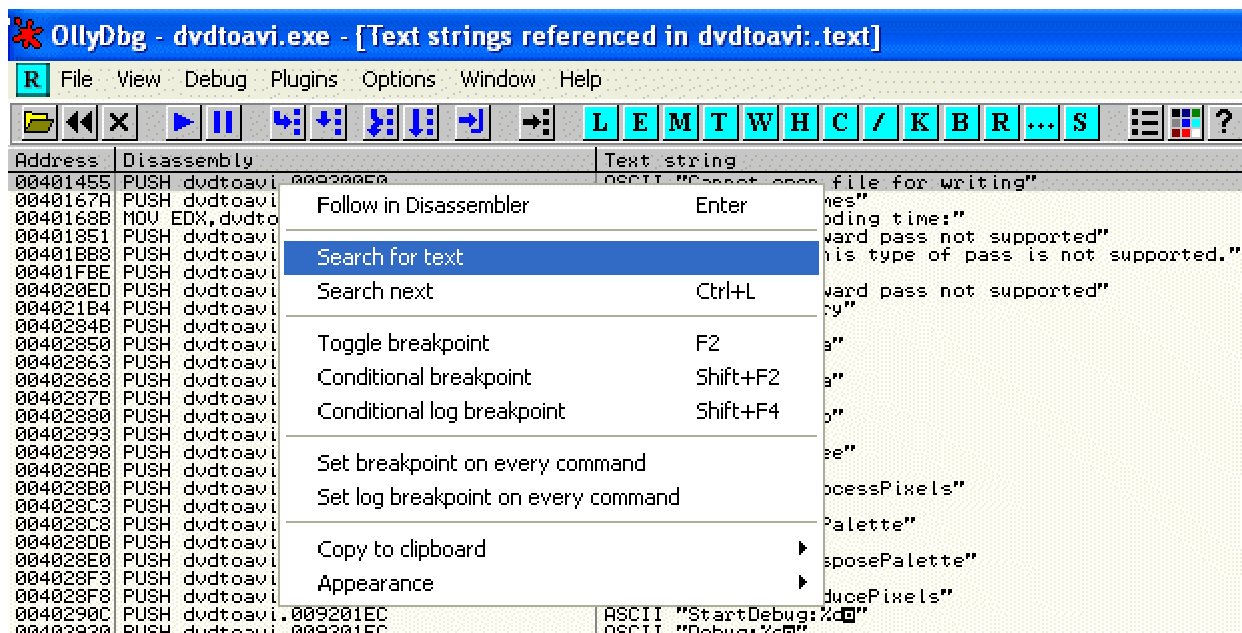


Fig. 5

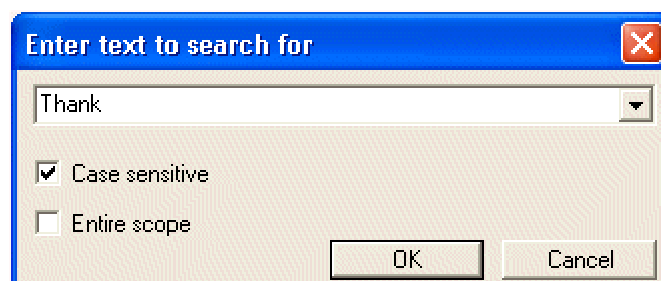


Fig. 6

Address	Disassembly	Text string
00411131	MOV EDX,dvdtoavi.00922AFC	ASCII ".avi"
004111D0	PUSH dvdtoavi.00923AE8	ASCII "Software\dvdtoavi\Output_Setting\RipNow"
00411376	PUSH dvdtoavi.00922C6C	ASCII "Software\dvdtoavi\Output_Setting\SplitMode"
00411523	PUSH dvdtoavi.00923E18	ASCII "wrong serialnumber,program terminate!"
00411558	PUSH dvdtoavi.00923E40	ASCII "Thank for registration,please restart the program."
0041157B	PUSH dvdtoavi.00923E18	ASCII "wrong serialnumber,program terminate!"
00411852	PUSH dvdtoavi.00923E18	ASCII "wrong serialnumber,program terminate!"
00411879	PUSH dvdtoavi.00923E18	ASCII "wrong serialnumber,program terminate!"
004118A0	PUSH dvdtoavi.00923E18	ASCII "wrong serialnumber,program terminate!"
004118C7	PUSH dvdtoavi.00923E18	ASCII "wrong serialnumber,program terminate!"
004118EC	PUSH dvdtoavi.00923C64	ASCII "Software\dvdtoavi\Output_Setting"
00411914	PUSH dvdtoavi.00923E00	ASCII "FileName"

Fig. 7

Place a breakpoint (click and then F2) on each entry (Thank and wrong message) and also scroll to the init of this routine on 00411504 and place also a breakpoint (F2), then launch the target (press F9), now look the behaviour, there is a first nag which tell about some limit of unregistered version.

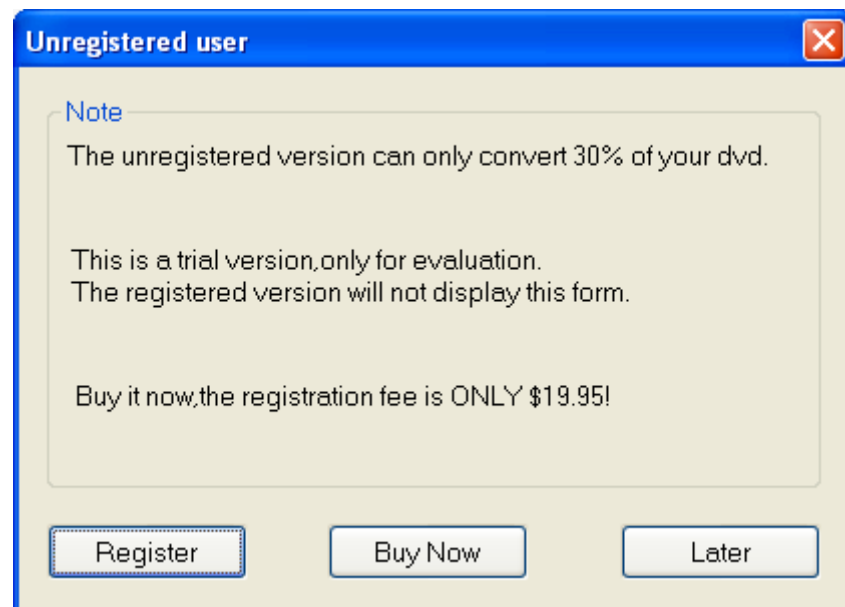


Fig. 8

Push the *Register* button and fill the field with your name and fake serial but care to keep serial length about $0x1D = 29$ character (this can be deduced from `CMP EDX,1D` after serial length calculation which is done by `CALL 004A9F6E` on `0041150D`) this will be more clear later.

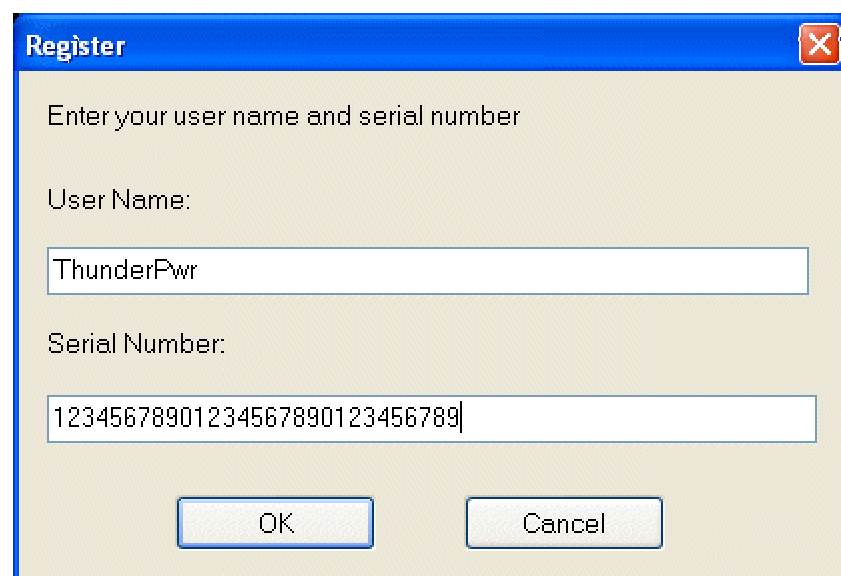


Fig. 9

When you press OK OllyDbg break on 00411504, now step int the code using F8:

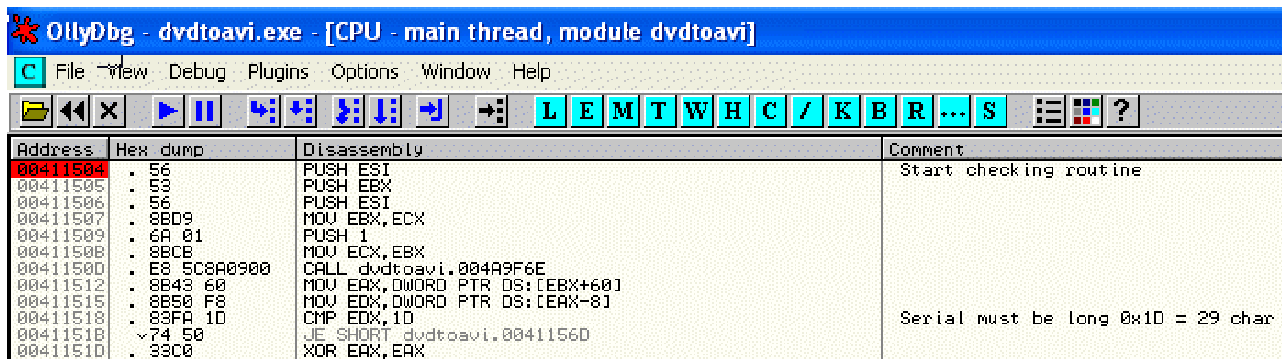


Fig. 10

Step until you reach the 00411518, if you re right compare will be and jump is taken, then there is another check about the 5 char on your serial, this must be equal to + char in order to avoid the bad guy message, then from EAX register value go into the memory dump window and change the char to + (0x1D ASCII code) then:

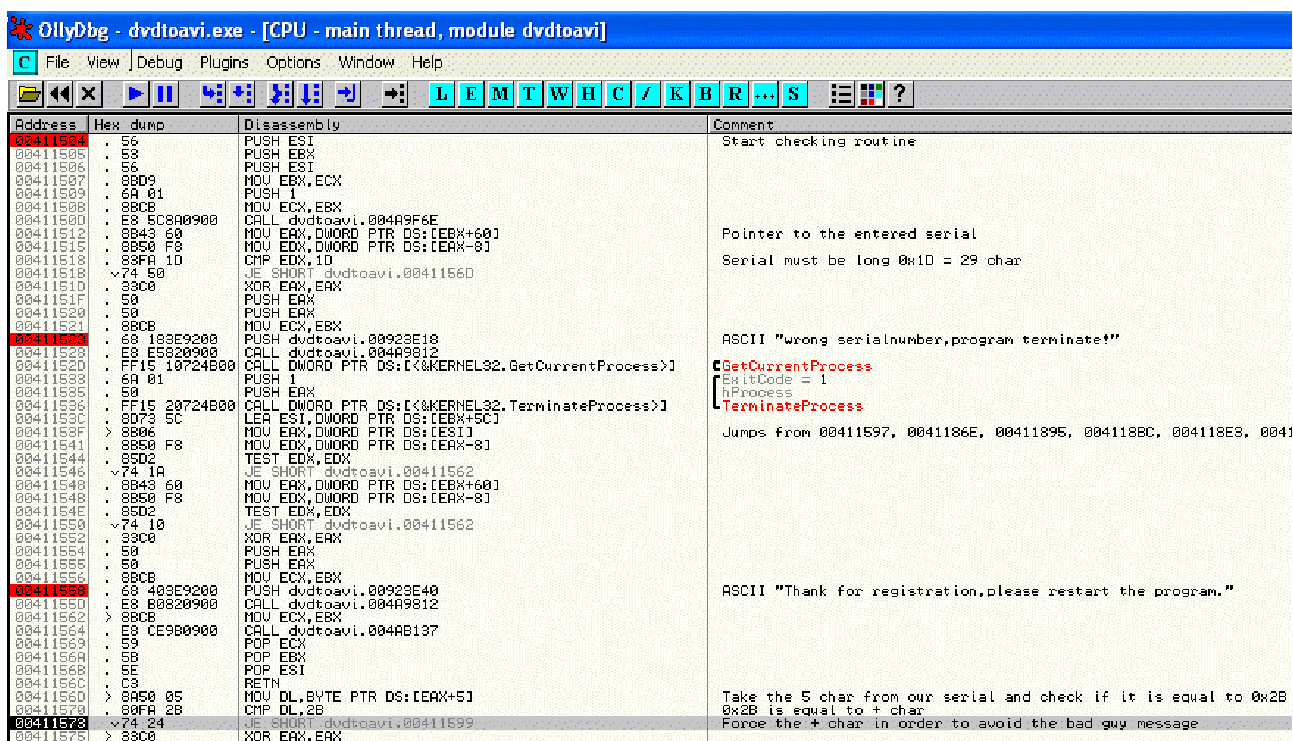


Fig. 11

Address	Hex dump	ASCII
00E93D20	31 32 33 34 35 2B 37 38 39 30 31 32 33 34 35 36	12345+7890123456
00E93D30	37 38 39 30 31 32 33 34 35 36 37 38 39 00 AD BA	7890123456789.+.

Fig. 12

And now you ve EAX 00E93D20 ASCII "12345+78901234567890123456789".

Now there is again another check, the 0xB char must be equal to + char then force it in a similar way:

```

00411599 > 8A50 0B MOV DL, BYTE PTR DS:[EAX+0]
0041159C . 80FA 2B CMP DL, 2B
0041159F . ^75 D4 JNZ SHORT dvdtoavi.00411575
004115A1 . 80FA 2B MOV DL, BYTE PTR DS:[EAX+1]

```

Now there are a new check about 0xB char, must be equal to 0x2B

Fig. 13

Address	Hex dump	ASCII
00E93D20	31 32 33 34 35 2B 37 38 39 30 31 2B 33 34 35 36	12345+78901+3456
00E93D30	37 38 39 30 31 32 33 34 35 36 37 38 39 00 AD BA	7890123456789. +

Fig. 14

then you have now EAX 00E93D20 ASCII "12345+78901+34567890123456789" next three check is similar to this two then when you reach the 04115AF address you shuld have this serial:

Address	Hex dump	ASCII
00E93D20	31 32 33 34 35 2B 37 38 39 30 31 2B 33 34 35 36	12345+78901+3456
00E93D30	37 2B 39 30 31 32 33 2B 35 36 37 38 39 00 AD BA	7+90123+56789. +

Fig. 15

EAX 00E93D20 ASCII "12345+78901+34567+90123+56789"

Well we have understand a little bit about serial structure then serial must be of this form:

+ ##### + ##### + ##### +

now we can skip all other test and just save our fake serial and keep the good boy message by using this patch (just jump to the end of this long chain text):

```

00411597 . ^EB A6 JMP SHORT dvdtoavi.0041159F
00411599 > 8A50 0B MOV DL, BYTE PTR DS:[EAX+0]
0041159C . 80FA 2B CMP DL, 2B
0041159F . ^75 D4 JNZ SHORT dvdtoavi.00411575
004115A1 . 8A50 11 MOV DL, BYTE PTR DS:[EAX+1]
004115A4 . 80FA 2B CMP DL, 2B
004115A7 . ^75 CC JNZ SHORT dvdtoavi.00411575
004115AB . 8A50 17 MOV DL, BYTE PTR DS:[EAX+17]
004115AC . 80FA 2B CMP DL, 2B
004115AF . ^ES 34530000 JMP dvdtoavi.004118E8
004115B4 . 80FA 41 CMP EDI, 41

```

Now there are a new check about 0xB char, must be equal to 0x2B

Now there are a new check about 0x11 char, must be equal to 0x2B

Now there are a new check about 0x17 char, must be equal to 0x2B

Fig. 16

Then make some step until you re able to see the registration message box on 0041155D.

Now before press again F9 go into the OllyDbg code window press CTRL+N and search for RegQueryValueA and place a breakpoint on each entry (this will be used during registration check which is performed on the next target startup).

Press OK and then quit from the target, now reload it on OllyDbg, be sure all breakpoint is fired and press F9, after some time you shuld be able to reach this nice code place.

OllyDbg - dvdtoavi.exe - [CPU - main thread, module dvdtoavi]

File View Debug Plugins Options Window Help

Address Hex dump Disassembly Comment

00409827 . 50 PUSH EAX
 00409828 . 50 PUSH EDI
 00409829 . 68 40049200 PUSH dvdtoavi.00920440
 0040982E . 68 01000000 PUSH 00000001
 00409833 . FF15 00704B00 CALL DWORD PTR DS:[&ADUAPI32.RegQueryValueA]
 00409839 . 85C0 TEST EAX, EAX
 0040983B . 74 00 JNZ dvdtoavi.00409A73
 00409841 . 8D7C24 0C LEA EDI, DWORD PTR SS:[ESP+C]
 00409845 . 33C0 XOR EAX, EAX
 00409847 . 8B37 MOV DH, BYTE PTR DS:[EDI]
 00409849 . 8BCF MOV ECX, EDI
 0040984B . 84F6 TEST DH, DH
 0040984D . 74 0C JE SHORT dvdtoavi.00409858
 0040984F . 83C1 01 ADD ECX, 1
 00409852 . 83C0 01 ADD EAX, 1
 00409855 . 8A11 MOV DL, BYTE PTR DS:[ECX]
 00409857 . 84D2 TEST DL, DL
 00409859 . 75 F4 JNZ SHORT dvdtoavi.0040984F
 0040985B . 85C0 TEST EAX, EAX
 0040985D . 74 00 JBE dvdtoavi.00409A73
 00409863 . C74424 08 0000 MOV DWORD PTR SS:[ESP+8], 200
 0040986B . 8D5424 0C LEA EDI, DWORD PTR SS:[ESP+C]
 0040986F . 8D4424 08 LEA EAX, DWORD PTR SS:[ESP+8]
 00409873 . 50 PUSH EAX
 00409874 . 52 PUSH EDI
 00409875 . 68 64239200 PUSH dvdtoavi.00922364
 00409879 . 68 01000000 PUSH 00000001
 0040987D . FF15 00704B00 CALL DWORD PTR DS:[&ADUAPI32.RegQueryValueA]
 00409885 . 85C0 TEST EAX, EAX
 00409887 . 74 00 JNZ dvdtoavi.00409A73
 0040988D . 8D7C24 0C LEA EDI, DWORD PTR SS:[ESP+C]
 00409891 . 33C0 XOR EAX, EAX
 00409893 . 8B37 MOV DH, BYTE PTR DS:[EDI]
 00409895 . 8BCF MOV ECX, EDI
 00409897 . 84F6 TEST DH, DH
 00409899 . 74 0C JE SHORT dvdtoavi.004098A7
 0040989B . 83C1 01 ADD ECX, 1
 0040989E . 83C0 01 ADD EAX, 1
 004098A1 . 8A11 MOV DL, BYTE PTR DS:[ECX]
 004098A3 . 84D2 TEST DL, DL
 004098A5 . 75 F4 JNZ SHORT dvdtoavi.0040989B
 004098A7 . 33F8 1D CMP EAX, 10
 004098A9 . 74 00 JNZ dvdtoavi.00409A73
 004098AB . 8D4424 11 MOV AL, BYTE PTR SS:[ESP+11]
 004098B4 . 3C 2B CMP AL, 2B
 004098B6 . 74 0A JC SHORT dvdtoavi.004098C2
 004098B8 . 33C0 XOR EAX, EAX
 004098BA . 81C4 0C110000 ADD ESP, 110C
 004098C0 . 5F POP EDI
 004098C1 . C3 RETN

Fig. 17 Starting registration check.

After this first simple checking about serial validity you re in 004098C2, this is a hell of checking about serial code, then scroll down a bit until you re able to reach the routine end and look:

```

00409A71 . 74 0A JE SHORT dvdtoavi.00409A7D
00409A73 . 33C0 XOR EAX, EAX
00409A75 . 81C4 0C110000 ADD ESP, 110C
00409A7B . 5F POP EDI
00409A7C . C3 RETN
00409A7D . B8 01000000 MOV EAX, 1
00409A82 . 81C4 0C110000 ADD ESP, 110C
00409A88 . 5F POP EDI
00409A89 . C3 RETN

```

Fig. 18

Now look also into the routine code, there are many RETN instruction and this sequence of command is repeated every test well, now we have to exit from code with success then skip all the bad exit in order to keep the right way then place this patch:

OllyDbg - dvdtoavi.exe - [Patches]

File View Debug Plugins Options Window Help

Address	Size	State	Old	New
0040991F	2	Active	JE SHORT dvdtoavi.0040992B	JMP SHORT dvdtoavi.0040992B
00409975	2	Active	JE SHORT dvdtoavi.00409981	JMP SHORT dvdtoavi.00409981
004099C9	2	Active	JE SHORT dvdtoavi.004099D5	JMP SHORT dvdtoavi.004099D5
00409A1D	2	Active	JE SHORT dvdtoavi.00409A29	JMP SHORT dvdtoavi.00409A29
00409A71	2	Active	JE SHORT dvdtoavi.00409A7D	JMP SHORT dvdtoavi.00409A7D

Fig. 19

On the last jump located on 00409A71 we have:

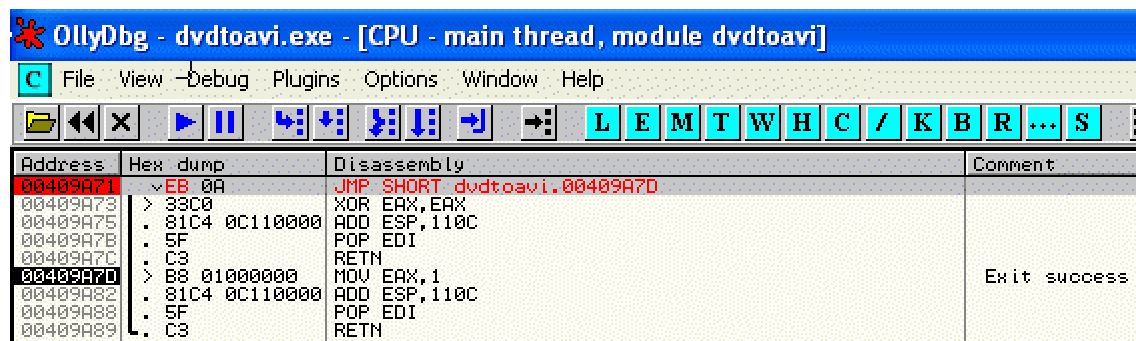


Fig. 20

Now save all the patches and then press F9, main window appear, no nag at this time then go straight to the About menu and choose the About option:

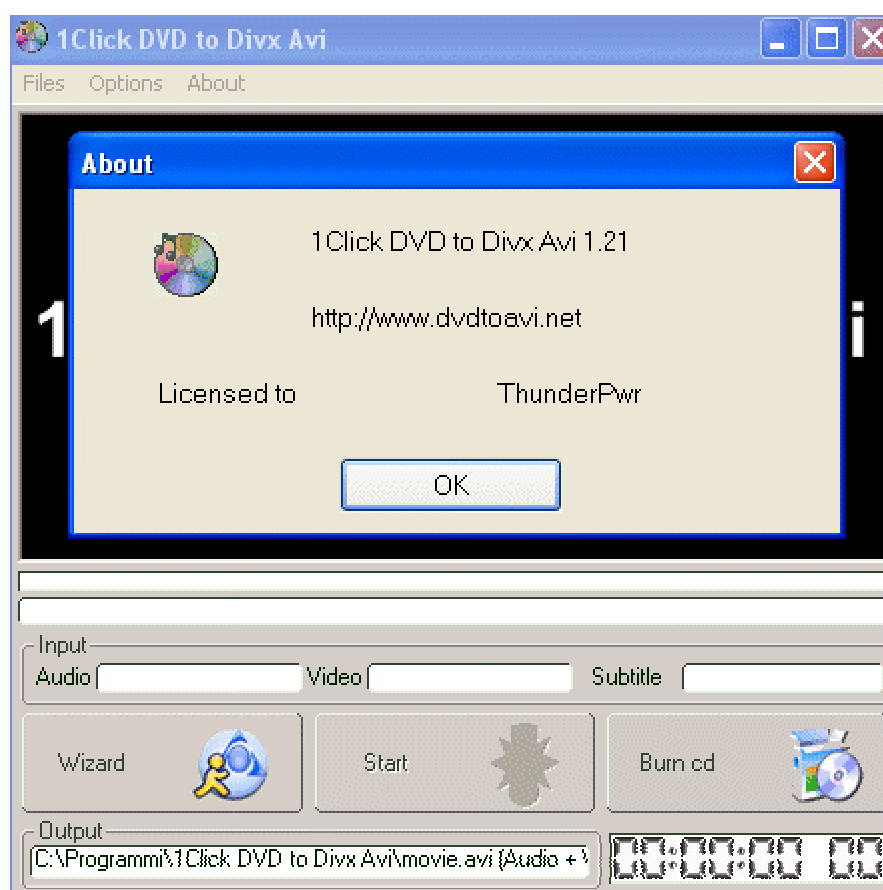


Fig. 21

Well mate work done program is fully registered but remind the fake serial must have some fixed char e.g.:

11111+11111+11111+11111+11111

can be used after program patching.

3. Serial reversing and keygenning stuff

Well, first step about registering is done, but best way to cracking is without patching or keep patching as small as possible, obviously the best way to crack is register the target without need of patching it. This task can be achieved by serial fishing if serial code can be retrieved from direct inspection through the code flow during debug stage (e.g. before serial checking right serial is pushed on the stack in a readable form then just write down it and use later to register application or in some case this is hardcoded then build directly into the executable code or some related module, like as DLL). This is really a simple situation, slightly complication arise when serial is not shown in clear form but it is checked from some calculation, hardest situation is about obfuscation of verification code or packed code, hashing technique and more complicated stuff.

About this program isn't possible keep serial by direct inspection (serial fishing) because serial is checked after some calculation (look on the code after you've press OK button into the registration window or just after application start-up). For this target serial checking is no more hard and registration name is not related to the right serial composition (this is easily shown during registration step, only the entered registration code is subject to check).

Well, now is time to reverse the serial checking routine in order to found a valid serial and then, if you like, build a keygen for this target.

I've choose to made analysis from the startup code (code is more compact than final check) then we have to work from 004098DA to 00409A89.

After first checking about char delimiter '+', main checking start from 004098DA, following is the first section of a total of five section, all section is equal regard register used to make calculation then we have to reverse the first section, other is the same.

00409800	891C24	MOV DWORD PTR SS:[ESP],EBX	
0040980D	897424 04	MOV DWORD PTR SS:[ESP+4],ESI	
004098E1	33FF	XOR EDI,EDI	
004098E3	33C0	XOR EAX,EAX	
004098E5	8BD8	MOV EBX,EAX	
004098E7	> 0FBE441C 0C	MOVSX EAX,BYTE PTR SS:[ESP+EBX+C]	keep the EBX character into the serial code string
004098EC	8D7424 0C	LEA ESI,DWORD PTR SS:[ESP+C]	
004098F0	50	PUSH EAX	
004098F1	03F3	ADD ESI,EBX	
004098F3	E8 1C7B0000	CALL dvdtoavi.00411414	
004098F8	59	POP ECX	
004098F9	0FBE16	MOVSX EDX,BYTE PTR DS:[ESI]	
004098FC	0FB6C0	MOVZX EAX,AL	
004098FF	35C0	TEST EAX,EAX	
00409901	< 74 06	JE SHORT dvdtoavi.00409909	
00409903	8D7C17 C9	LEA EDI,DWORD PTR DS:[EDI+EDX-37]	
00409907	EB 04	JMP SHORT dvdtoavi.0040990D	
00409909	> 8D7C17 D0	LEA EDI,DWORD PTR DS:[EDI+EDX-30]	
0040990D	> 83C3 01	ADD EBX,1	Increment the counter
00409910	83FB 05	CMP EBX,5	Check if we have examined all the five character
00409913	< 7C D2	JL SHORT dvdtoavi.004099E7	Jump if we have to examine other char
00409915	8B1C24	MOV EBX,DWORD PTR SS:[ESP]	
00409918	8B7424 04	MOV ESI,DWORD PTR SS:[ESP+4]	
0040991C	83FF 28	CMP EDI,28	
0040991F	< 74 0A	JE SHORT dvdtoavi.0040992B	Jump if the examined serial section is correct
00409921	33C0	XOR EAX,EAX	
00409923	81C4 0C110000	ADD ESP,110C	
00409929	5F	POP EDI	
0040992A	C3	RETN	
0040992B	> 891C24	MOV DWORD PTR SS:[ESP],EBX	

Fig. 22

This is a very simple algorithm, for each section (composed of five char) each char is examined and if this ASCII code is between 0x41 and 0x46 EAX is keep to 0 otherwise EAX is set to be equal to 1, this comparison is made into the CALL inside the loop.

00411414	\$ 8A4424 04	MOV AL,BYTE PTR SS:[ESP+4]	AL is equal to the current ASCII value for the readed char
00411418	3C 41	CMP AL,41	
0041141A	< 7C 0A	JL SHORT dvdtoavi.00411426	
0041141C	3C 46	CMP AL,46	
0041141E	< 7F 06	JG SHORT dvdtoavi.00411426	
00411420	B8 01000000	MOV EAX,1	EAX=1 if char is between A and F
00411425	C3	RETN	
00411426	> 33C0	XOR EAX,EAX	EAX=0 otherwise
00411428	C3	RETN	

Fig. 23

In order to keep the serial composition simple suppose to have only number, then EAX=0 anyway.

Well if you execute the loop, after some calculation EDI keep your result which must be equal to 0x28 = 40 dec (first section), by the way you ve also to observe:

LEA EDI,DWORD PTR DS:[EDI+EDX-30]

Is always executed due to the CALL result (EAX = 0) but also if we have only number a subtraction to 30 from between ASCII code is equal to subtract 0 in a decimal way then your check is simply the sum which follow from the first five character. From this point of view a valid serial can be:

99994 -> 9+9+9+9+4 = 40 (0x28)

in more readable form, if you prefer, we can code also a simple VB function able to perform this task (txtSerial is a text box which contain the serial registration code):

```
Dim serial As String
Dim currChar As String

Dim EAX As Integer
Dim EBX As Integer
Dim EDX As Integer

serial = txtSerial.Text

' =====
' Checking loop #1
' =====

loop1:
EAX = Asc(Mid$(serial, EBX + 1, 1)) ' MOVZX EAX,BYTE PTR SS:[ESP+EBX+C]
'-----
'CALL dvdtoavi.00411414
If ((EAX < 65) Or (EAX > 70)) Then
    EAX = 0
Else
    EAX = 1
End If
'-----
EDX = Asc(Mid$(serial, EBX + 1, 1)) ' MOVZX EDX,BYTE PTR DS:[ESI]

If EAX = 1 Then
    EDI = EDI + EDX - 55
Else
    EDI = EDI + EDX - 48 ' 48 is ASCII code for zero
End If

EBX = EBX + 1 ' ADD EBX,1
If EBX < 5 Then ' CMP EBX,5
    GoTo loop1 ' JL SHORT dvdtoavi.004098E7
End If
' =====
MsgBox "Result loop #1 = " + Hex(EDI), vbOKOnly, "Result"
```

Well, next section is similar, there is only one difference and this reside into the comparing check after the loop, we have:

section #1:	0x28	(address 0x0040991C)
section #2:	0x29	(address 0x00409972)
section #3:	0x2A	(address 0x004099C6)
section #4:	0x2B	(address 0x00409A1A)
section #5:	0x2C	(address 0x00409A6E)

Then a valid serial code is:

99994+99995+99996+99997+99998

and this is valid for every used user name.

Now is time to check it on the original packed executable, before to do it uninstall and then install again the application in order to clear previously registry information.

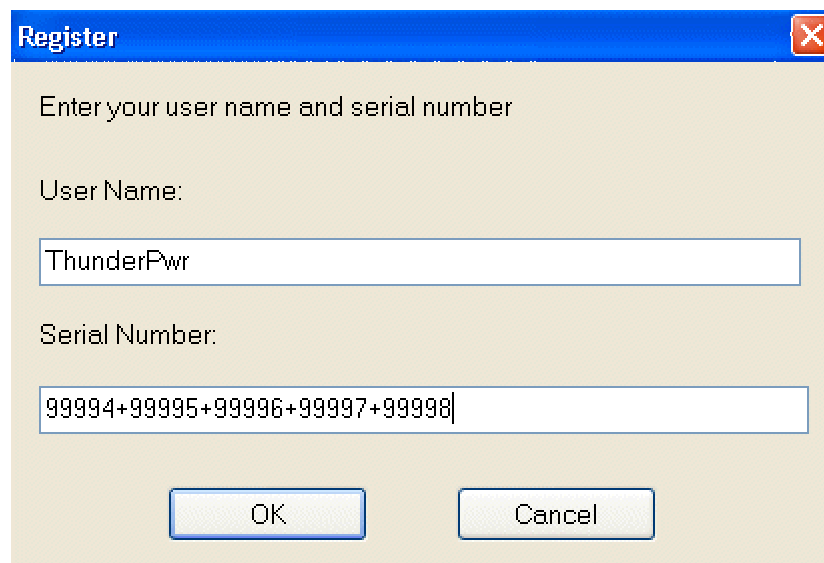


Fig. 24

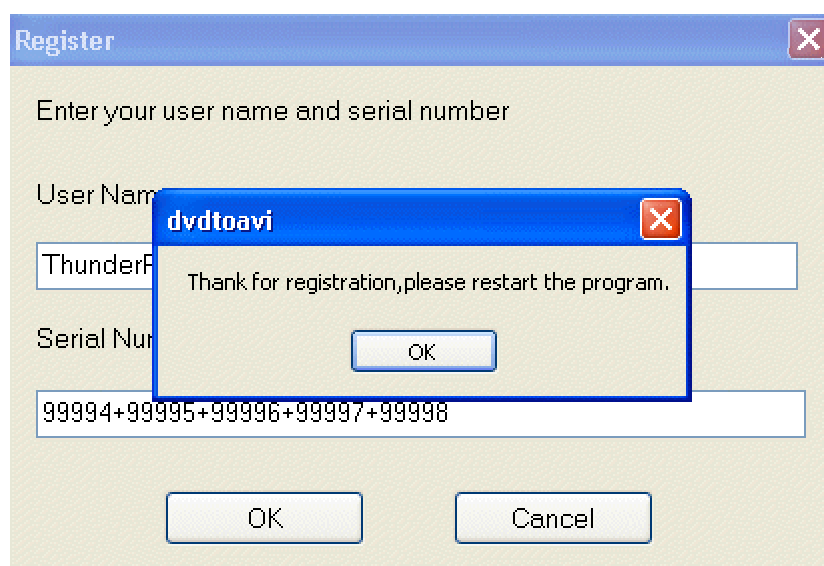


Fig. 25