

# Unpacking With Tracers I +NCR/CRC! [ReVeRsEr] of ARTeam

Version 1.0 – January 2006

1.	Abstract	2
2.	Working With Tracers	3
3.	References	24
4.	Conclusions	24
5.	History	24
6.	Greetings	24

### Keywords

Tracers.



## 1. Abstract

Hi you all!!!. This is my first tutorial for a non spanish group. So forgive me if something's a little difficult to understand. Besides, this is my first tutorial for such an important and well known cracking group as **ARTeam**.

Well, sorry but I haven't introduced myself. My name's **+NCR/CRC!** [**ReVeRsEr**] and I'm a member of CracksLatinoS! 2005. Many of you surely know that cracking list, by one of its founders, the Master Ricardo Narvaja. He, besides CracksLatinoS, has also been a member of lists like [**aRC**], [**RVLCN**] and **PDAToolBoxHispano**.

Well, I hope my nerves won't let me down and that the work will be interesting enough for you.

In this tutorial we'll do some unpacking using Tracers, a new technique developed by **AkirA** (a friend of mine and author of tutorials about Xprotector and Themida, You surely know him ;) and now implemented by yours truly.

It'll be an easy unpacking, but what's important are the tracers.

Before commencing I warn you that my way of working will be the same as in CracksLatinoS. The ones that already know me are surely used to it. And for the new ones, I hope you find it interesting.

Now, to the tutorial...

As usual I will provide sample code with this tutorial, and non-commercial sample victims. All the sources have been tested with Win2XP and Visual Studio 6.0 (Visual C++). The techniques described here are general and not specific to any commercial applications. The whole document must be intended as a document on programming advanced techniques, how you will use these information will be totally up to your responsibility.





# 2. Working With Tracers

As I said before, we're going to perform an easy unpacking of a packer that completely destroys the IAT as well as the table of jumps. We'll automatize things very little 'cause our objective is to show the technique and besides, all these things are still in a testing phase since I asked **kaos\_xlro** to make some changes in the source programs. So it's still a long way to go ;)

This is my second tutorial on tracers, the first one was written for CracksLatinoS and if anyone wants to download it, you can go to Ricardo Narvaja's FTP or just drop me a line.

CrackME2 por Yllera

Let's start by introducing the program, this is my Guinea pig:

Figure 1 - My guinea pig =).

First of all, let's use the **RDG Packer Detector** to identify the packer:



Figure 2 – Hying's PE-Armor v0.46.

It tells us that it's Hying's PE Armor v0.46. Now, let's try with PEiD to see if we can find the OEP:



Figure 3 – The real OEP?.



Well, it seems that this packed program's OEP is at 401000. Let's check it out with Olly.

I open my OllyDbg, with the patch for VProtector, Anti-CAPTION and CLASS WINDOW and the HideDebugger plugin activated with all its options, and I load the program:

*	💥 - [CPU - main thread, module Hying's]									
С	File	View	Debug	Plugins	Options	Window	v H	elp		
	••	×	► II	4	}: I:	-	+	L	EM	TWH
004 004 004 004 004 004 004 004 004 004	7700 7700 7700 7700 7701 7701 7701 7701	5901590159	E8 AA90 2D79979 9999999 3D79979 2D79979 9999999 9999999 9999999 9999999 999999	0000 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	ALL Hyin D 000770 D 000000 D 0000770 D 000770 D 000770 D 000000 D 000000 D 000000 D 000000 D 000000	9's.0047 2D 00 3D 2D 2D 00 00 00 00 00	770A			

Figure 4 – EntryPoint.

There you can see the packer's EntryPoint (EP). Now, let's try to get to the OEP. But first, let's configure Olly in the following way: We go to Options – Debugging Options – Exceptions and we only check the 'Ignore Memory Access Violations in KERNEL32.dll' box:

🗄 Debugging options 🛛 🛛 🔊						
Commands   Disasm   CPU   Registers   Stack   Analysis 1   Analysis 2   Analysis 3						
Security Debug Events Exceptions Trace SFX Strings Addresses						
✓ Ignore memory access violations in KERNEL32						
Ignore (pass to program) following exceptions:						
🔲 INT3 breaks						
🔲 Single-step break						
Memory access violation						
Integer division by 0						
Invalid or privileged instruction						
All FPU exceptions						
Ignore also following custom exceptions or ranges:						
OEEDFACE Add last exception						
40000005						
80000004 (SINGLE STEP) C0000005 (ACCESS VIOLATION) Delete selection						
UK Undo Cancel						

Figure 5 – Configuring OllyDbg.



Then, I press F9 (Run) and an exception occurs:



Figure 7 – An Access Violation.

In order to pass the exception we only need to press <Shift+F7> and then F9 again. Another exception occurs and I again pass it by pressing <Shift+F7> followed by F9. If we do this several times, we can see that many more exceptions occur. So we repeat the above process until the program is running (it hasn't any anti-debug tricks). In that very moment, I press the minus sign key ('—') to go back in Olly and I see what the last exception was (befote the program run) so that I can write a script later on with OllyScript that will leave us in that last exception without working too much:

start: eob break run

break: cmp eip,581fe8 je final esti jmp start

#### final: ret

So we run the script and the program stops just in that last exception:

OllyScript	•	Run script
OllyUni	•	Abort
Breakpoint Manager	•	Pause
Anti Anti BPM	•	Resume
OllyDbg PE Dumper	+	Step
UnhandledExceptionFilter 0.22p	1	About

Figure 8 – Running a script.



Run Olly Script	ot (	? 🔀
Buscar en:	: 🎦 [ARTeam]_Unpacking With Tracers_by_+NC 🗾 🛭 🖨 📸 📰 🕇	
Documentos recientes Escritorio Mis documentos	LastException	
Mi PC		
Mis sitios de red	Nombre: LastException Abrin   Tipo: Olly Scripts Cancel	lar
	F Abrir como archivo de sólo lectura	

Figure 9 – Selecting the script.

And soon after, we get the warning telling us that the script finished its work:



Figure 10 – Script finished.

And we are at the last exception, that in my case is:

-			
005	81FE8	AD	LODS DWORD PTR DS:[ESI]
005	81FE9	CD 20	INT 20
005	81FEB	E8 07000000	CALL 00581FF7
005	81FF0	C783 83C013EB	MOV DWORD PTR DS:[EBX+EB13C083].2EB580B
005	81FFA	CD 20	INT 20
005	81FFC	8300 02	ADD EAX.2
005	81FFF	EB 01	JMP SHORT 00582002
005	82001 -	E9 50C3E8E8	JMP E940E356
665	88882	2000	ODD DUTE DTD DO SEOUR OF

*Figure 11 – My last exception.* 

Access violation when reading [00000000] - use Shift+F7/F8/F9 to pass exception to program

Figure 12 – An access violation.

I pass this last exception with <Shift+F7> and then I go to the Memory Map by pressing <Alt+M> in order to put a BPM on Access in the code section of the program, the one just below the PE-HEADER: