AnonymouS [ARTEAM]

Some Insights into SecuRom 7.30.0014

Version 1.0
Last Rev.: August 2007

Into this Tutorial

Forewords

It's been a while since I did any reversing. It hasn't been much reversing since the release of the X-Prot v2 unpacker. First of all because I'm lazy, but real life also had a lot going on. Anyway, I have been following the SecuRom thread (http://forums.accessroot.com/index.php?showtopic=4361&st=0) on ARTeam's forum for a while, decided to look deeper into SecuRom 7.

Initially I wanted to code an unpacker and started coding unpackers for SecuRom 7.10 through 7.12. As I began coding on an unpacker for SecuRom V7.18 I found out that the task was quite demanding so I abandoned 7.18 and moved on to 7.30.0014.

This small tutorial/essay is not about completely reversing SecuRom 7.30.0014. It's just a help for people on how to reach OEP and on the way defeating the anti-debugger trick that apparently stops a lot of people. I will also show I bypass the anti-dump trick used by SecuRom.

The tutorial/essay is not very explaining as I do think that people reading this will be somehow more than just a newbie reverser. SecuRom is a tough protection and good reversing skills are needed in order to fully reverse this protection.

AnonymouS

Editor: Shub-Nigurrath

Disclaimers

All code included with this tutorial is free to use and modify; we only ask that you mention where you found it. This tutorial is also free to distribute in its current unaltered form, with all the included supplements.

All the commercial programs used within this document have been used only for the purpose of demonstrating the theories and methods described. No distribution of patched applications has been done under any media or host. The applications used were most of the times already been patched, and cracked versions were available since a lot of time. ARTeam or the authors of the paper cannot be considered responsible for damages to the companies holding rights on those programs. The scope of this tutorial as well as any other ARTeam tutorial is of sharing knowledge and teaching how to patch applications, how to bypass protections and generally speaking how to improve the RCE art and generally the comprehension of what happens under the hood. We are not releasing any cracked application. We are what we know..

Verification

ARTeam.esfv can be opened in the ARTeamESFVChecker to verify all files have been released by ARTeam and are unaltered. The ARTeamESFVChecker can be obtained in the release section of the ARTeam site: http://releases.accessroot.com

Table of Contents

Disclaimer of ARTeam: We received this tutorial from an AnonymouS author (an author who doesn't want to see his name published) who contacted us in order to see this paper published.
It's not the first time we publish tutorials from outsiders or even from anonymous guys, so we did it again. The content is interesting and technically well done, so we decided to give it out on our tutorials pages.
If you want to be the next "anonymous" please consider that we want to deliver the same quality of content and information, you are used to from ARTeam tutorials. Any tutorial submitted to us has to pass the inspection and approval of the entire team before publication.
We applied our usual tutorials formatting to the text we received, and verified correctness of information on the technical side. Any other thing is responsibility of the author, even if the work is not original, do not complain us.

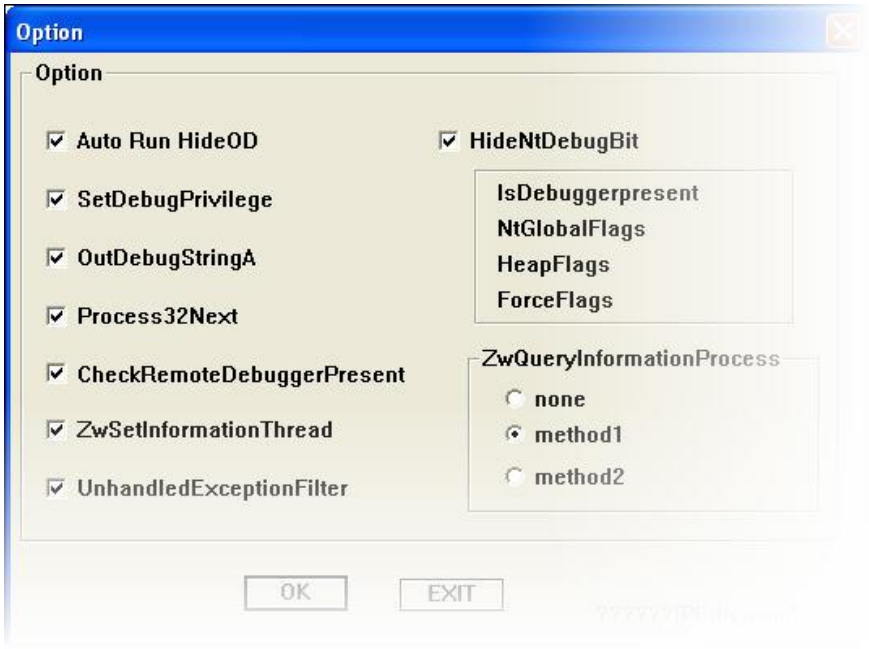
1. Settings and Target

1.1. Target

Resident Evil 4 from Capcom

1.2. Tool Used

OllyDbg V1.10
OllyDbg plugin (HideOD) with the following settings:



TaskMngr by drizz (http://drizz.t35.com/main.php)

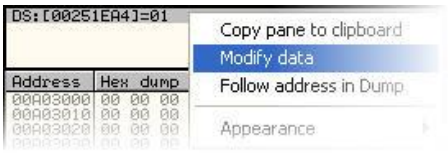
2. Defeating the mysterious debug-detection

Okay, let's go to work... Run Olly, select executable and let Olly loose. We hit to exceptions before get this error message:

In the earlier versions I used to get this whenever the ZwQueryObject trick was launched, but when I patch this I still get the error message, so I tend to think it's the mysterious debugger detection people are talking about on the ARTeam thread.

How does it detect us ?? We're using HideOD and ZwQueryObject is not the reason, so this must be a new debugger detection. Let's start tracing… I will spare you for the agonizing of tracing through huge amount for checksums with SecuRom and let you straight to the answer.

Notice the CMP BYTE PTR DS:[EAX+4],0 ??? It's really a part of a much large procedure… Anyway, modify this by setting TRUE back to FALSE (1 to 0).



To be honest, I'm not quite sure what exactly happen, but take a look at this clean code, stripped from obfuscation, trap-flag protection and checksums:

```
0577F556      FFD5          CALL EBP                          <-- call RtlAcquirePebLock
056D89E3      8B45 08       MOV EAX,DWORD PTR SS:[EBP+8]      <-- [EBP+8] == 30h
056D89E6      64:8B00       MOV EAX,DWORD PTR FS:[EAX]        <-- Get PEB address
0577FAA2      8B40 0C       MOV EAX,DWORD PTR DS:[EAX+C]      <-- PPEB_LDR_DATA
0577FDC9      8078 04 00    CMP BYTE PTR DS:[EAX+4],0         <-- TRUE if debugged
0577FDD3      75 12         JNZ SHORT game.0577FDE7           <-- jump bad boy
```

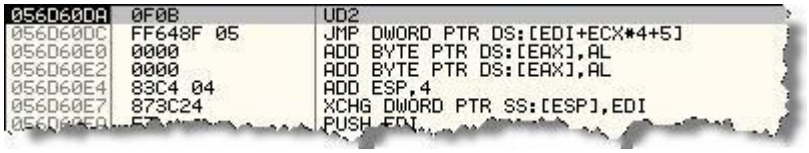
From Microsoft's library I came across this:

```
typedef struct _PEB_LDR_DATA {
  BYTE Reserved1[8];
  PVOID Reserved2[3];
  LIST_ENTRY InMemoryOrderModuleList;
} PEB_LDR_DATA,
 *PPEB_LDR_DATA;
```

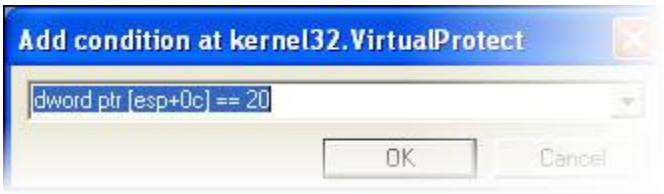
From what I can see the flag at [EAX+4] is somehow switched on when debugged. I tried coding a little test myself but I always came up with a TRUE result even if I was not running a debugger !?!

3. Reaching OEP

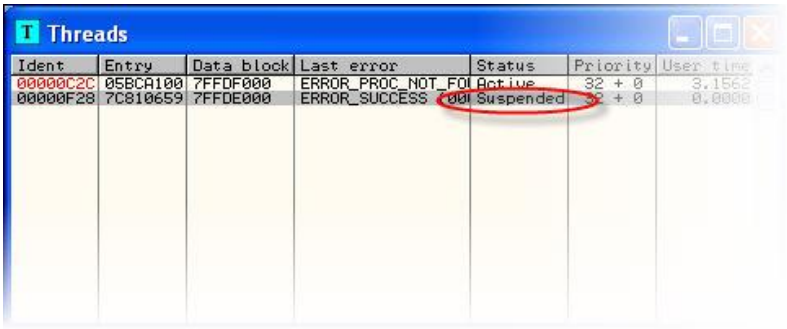
Fire up Olly and run through all exceptions, including fixing the anti-debug trick, until you reach:



Then set a conditional BPX on VirtualProtect and run it.

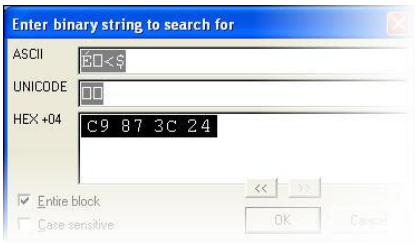


Once Olly breaks clear BPX and suspend all threads except the main one.

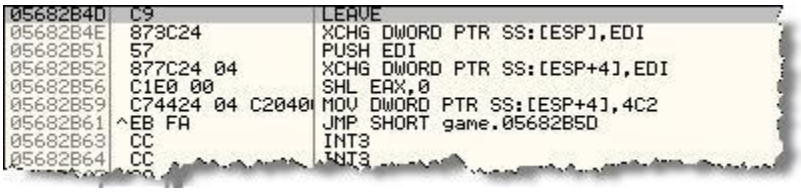


Now set a breakpoint on ReleaseMutex and let Olly run until it breaks again.
CTRL+F9 will lead you to the RETN in ReleaseMutex. Trace back into user-code and you will end up here:

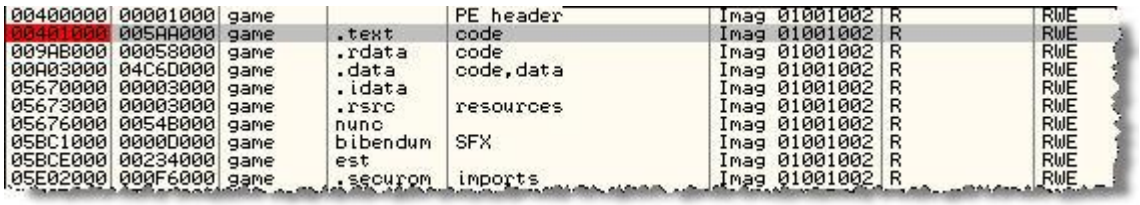
Now search for the pattern: **C9 87 3C 24**



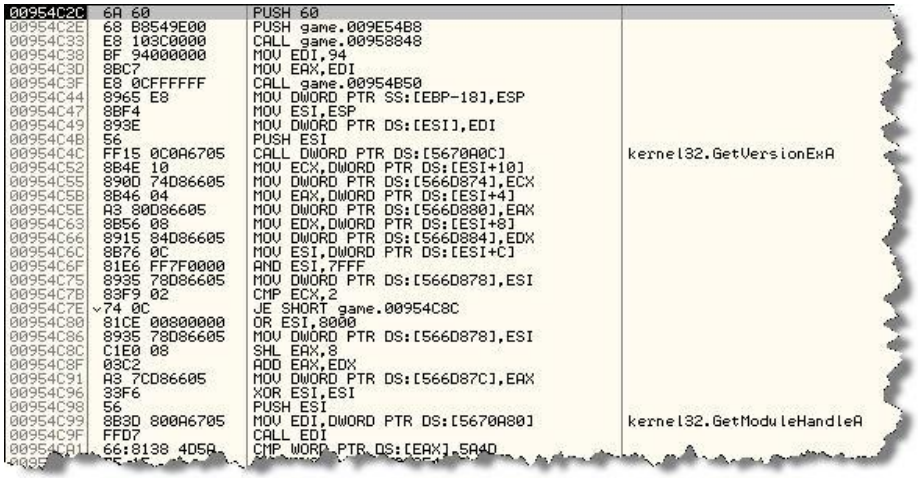
Be aware there are more offsets that match this pattern but the first one found should be the "good" one. If this is not the case, you better start tracing ;) Anyway should look something like this:



Place a Hardware breakpoint on the instruction after the LEAVE opcode and run Olly. When Olly breaks go to Memory Map and place a breakpoint code section using F2 and run Olly.



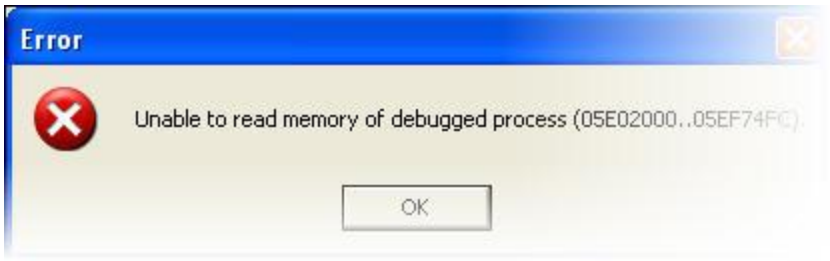
Next time Olly breaks we're at OEP.



Now we can dump…. Or can we ???

4. Defeating the anti-dumping trick.

As one can see, reaching the OEP of SecuRom 7.30.0014 is fairly easy. However, the authors did another attempt to slow us down. When we fire up our PE-dumper we get this message:



Hmm… What happens here ?? Our dumper won't dump ?!? Don't worry… This is easily defeated … Luckily for you I did all the tracing through SecuRom's checksum hell. Let's rewind time a little bit ;)

As I encountered this I decided to find out where exactly made this anti-dump trick possible. I let the executable run until the first exception and tried to dump. Here I also got the error message… So, this means that the anti-dump trick is setup before the first exception. Now I simple started the slow process of tracing through tons of checksums. First I encountered this:

```
0023F95C  0023F97C  ASCII "Wrapper - setting guard page at 05E06000h"
0023F960  0567BF57  game.0567BF57
0023F964  0023F97C  ASCII "Wrapper - setting guard page at 05E06000h"
0023F968  0023F97C  ASCII "Wrapper - setting guard page at 05E06000h"
0023F96C  05BDBD50  ASCII "Wrapper - setting guard page at %08Xh"
```

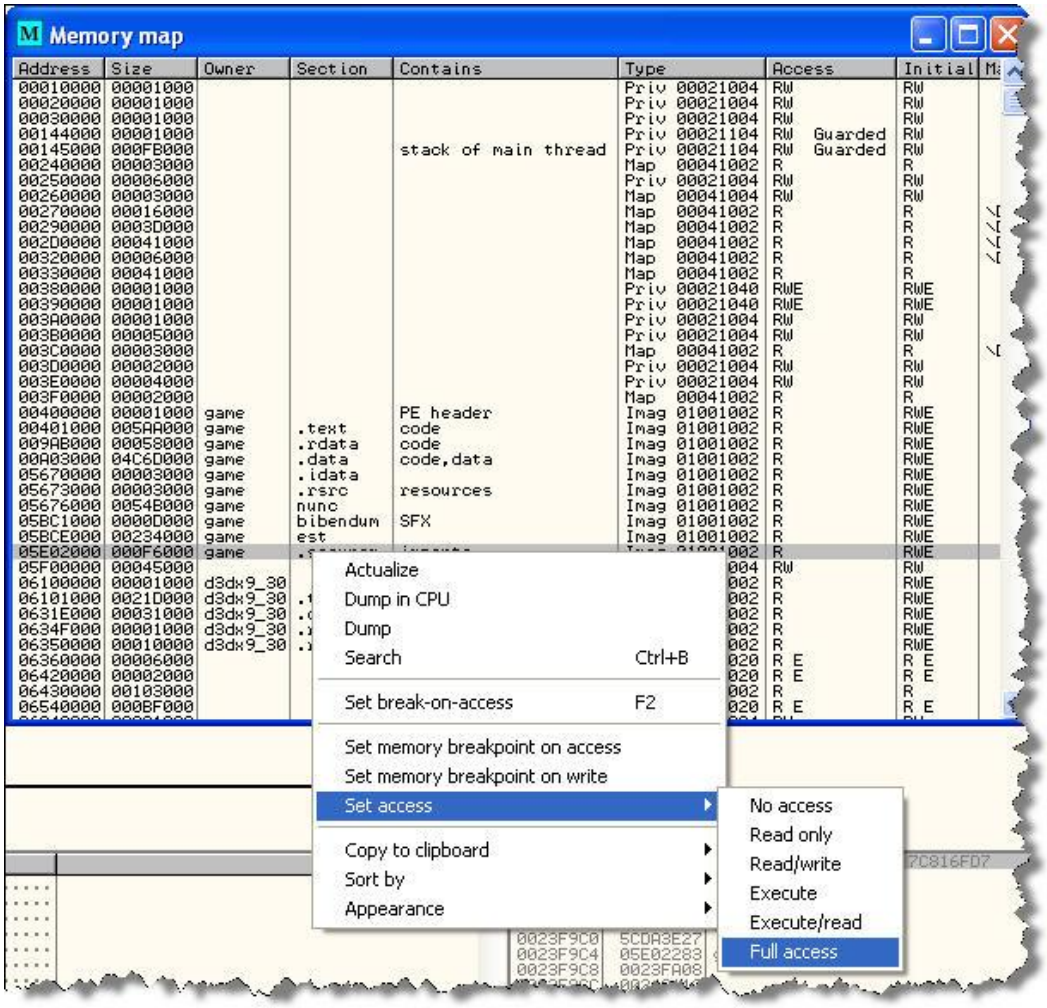
This let me to the thought that SecuRom for some strange reason GUARD PAGE the .securom section. So I set at breakpoint on VirtualProtect and after several hits I ended here:

```
0023F960  0567C150  CALL to VirtualProtect from game.0567C14A
0023F964  05E06000  Address = game.05E06000
0023F968  00001000  Size = 1000 (4096.)
0023F96C  00000001  NewProtect = PAGE_NOACCESS
0023F970  0023F9FC  pOldProtect = 0023F9FC
```

I then retraced my way back to the user code (CTRL-F9) and ended here:

```
0567C13F  9D              POPFD
0567C140  870C24          XCHG DWORD PTR SS:[ESP],ECX
0567C143  8D6424 FC       LEA ESP,DWORD PTR SS:[ESP-4]
0567C147  891C24          MOV DWORD PTR SS:[ESP],EBX
0567C14A  FF15 CB8CE005   CALL DWORD PTR DS:[<&KERNEL32.VirtualPr  kernel32.VirtualProtect
0567C150  83EC 04         SUB ESP,4
0567C153  90              NOP
```

Now we know what to do when reach the OEP, so let's fast forward to the OEP and open up the Memory Map and set the access back to FULL ACCESS on the .securom section:



Now we can finally dump the executable without any problems.

5. Conclusion

As one can see it is fairly easy to reach the OEP of SecuRom 7.30.0014. It's also possible to dump the executable once we set back the page access to FULL. As for the mysterious debugger detection, we found it and is able to fix the problem. However, reaching OEP and being able to dump the executable is not enough to defeating SecuRom. The dump is filled with VM code, code-splicings etc. etc. I only showed the way to the promised land, it's now up to you to work your way through it yourself ;)

6. Final words and greetings

As stated in the forewords this is not a complete tutorial on how to reverse SecuRom, but merely a quick step-by-step on how to reach the OEP of version 7.30.0014. I must admit that I haven't looked into the VM and code-splicing of this version. I initially kind of promised that I would code another unpacker but I doubt that I will ever code any again. Basically I haven't got the time anymore and the credits you get from your many hours of work are minimal. However, if my good friend, Nacho DJ, talk me into it I might do some more tutorials ;)

Last but not least I would like to send out my greetings to:

- My wife, Kristine and my son Frederik
- All ARTeam member (especially Nacho DJ)
- Drizz
- All I forgot

Sincerely,
AnonymouS / ARTeam