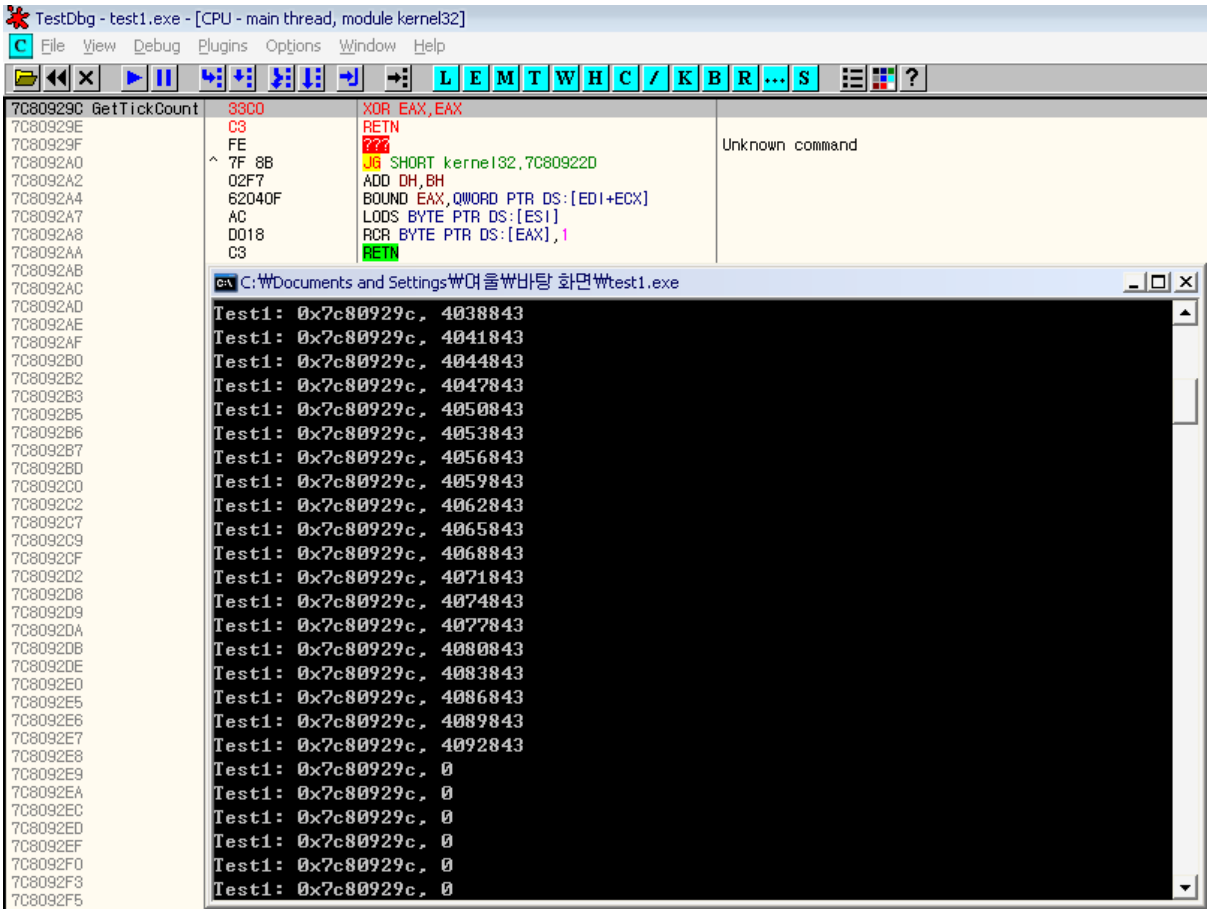
DLL과 COW(Copy-On-Write)

Written by k0nni3

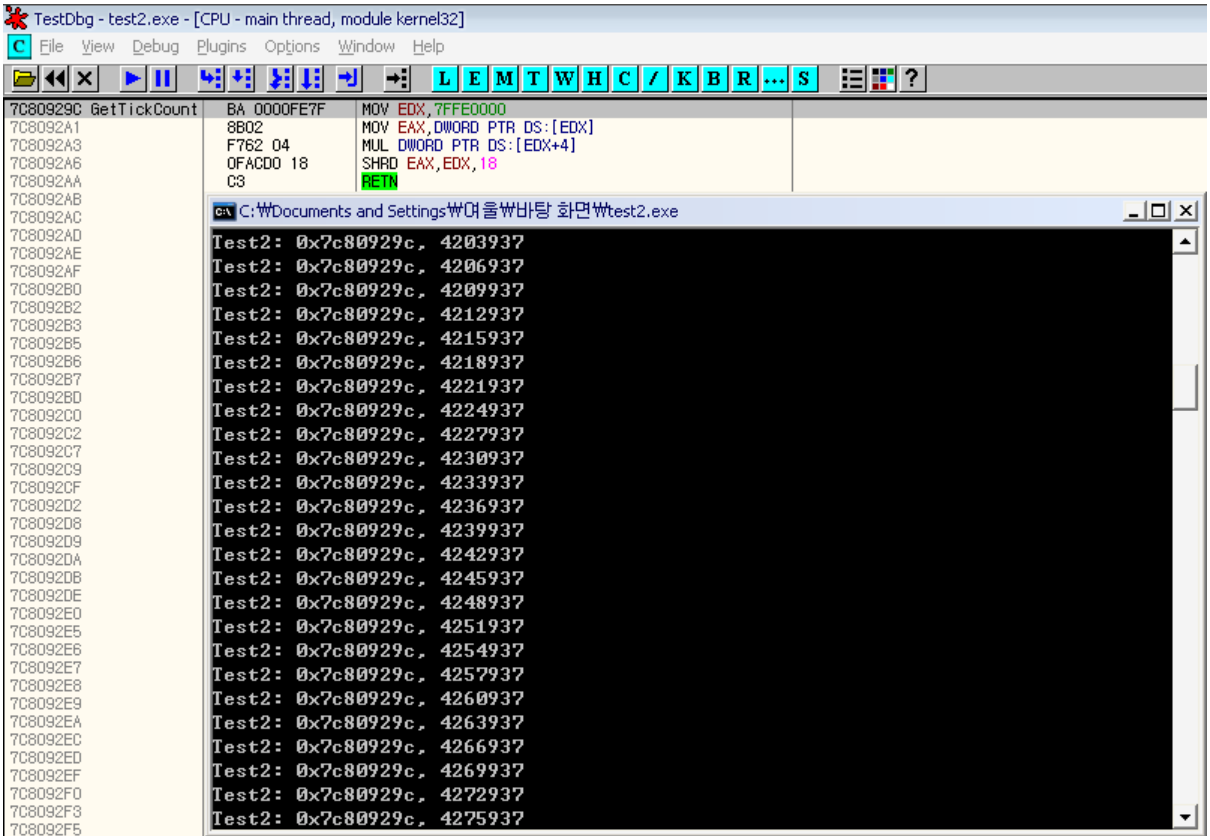
필자가 Windows Hooking에 관하여 공부하면서 고민했던 것으로 Hooking에 관해 공부해본 사람들이라면 한 번쯤 고민했을 법한 이야기를 하고자 한다. 이러한 고민은 여기서부터 출발하였다.

DLL은 공유메모리에 올라가고 그 DLL을 사용하는 모든 프로세스들은 그 DLL을 공유한다. 그렇다면 A라는 프로세스가 참조하고 있는 DLL함수를 수정하면 같은 DLL함수를 참조하는 B라는 프로세스에게도 영향을 주지 않을까?

하지만 필자의 생각과는 달리 B라는 프로세스는 전혀 영향을 받지 않았다. 간단히 예제로 살펴보자.



<그림 1> test1 – GetTickCount 함수 수정

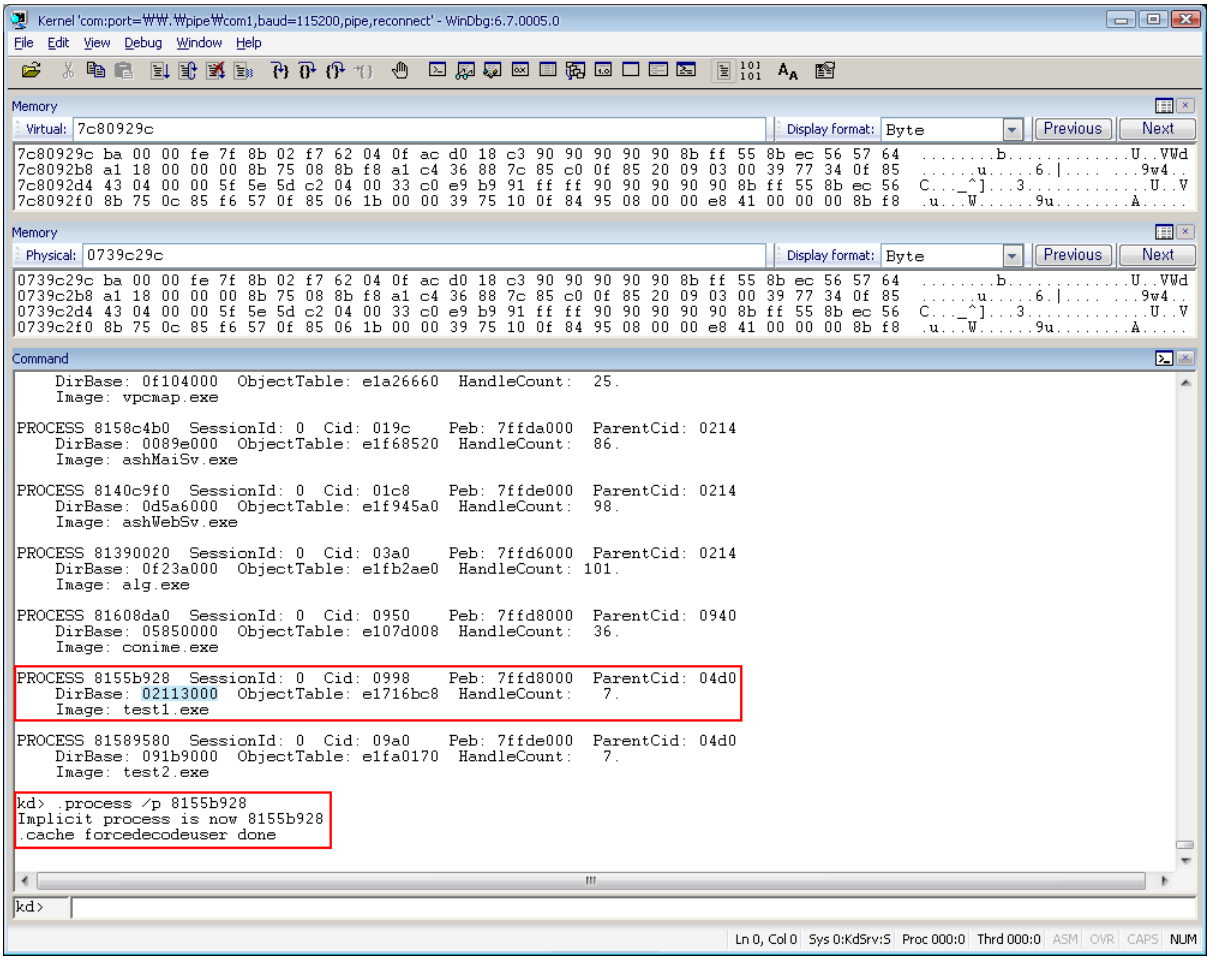


<그림 2> test2 – 원본 GetTickCount 함수

위 <그림 1>과 <그림 2>에서 보는 바와 같이 test1과 test2는 kernel32.dll의 GetTickCount 라는 함수를 사용해서 GetTickCount 함수의 주소와 처리결과를 3초를 주기로 하여 계속해서 출력하는 프로그램이다. 하지만 재미있는 것은 test1이 참조하는 GetTickCount 함수를 수정하였음에도 불구하고 test2는 어떠한 영향도 받지 않는다는 것이다.

그렇다면 Windows에서 DLL은 공유된다는 것은 잘못된 것일까? 만약 그렇다면 DLL을 사용해서 얻는 이익은 무엇인가?

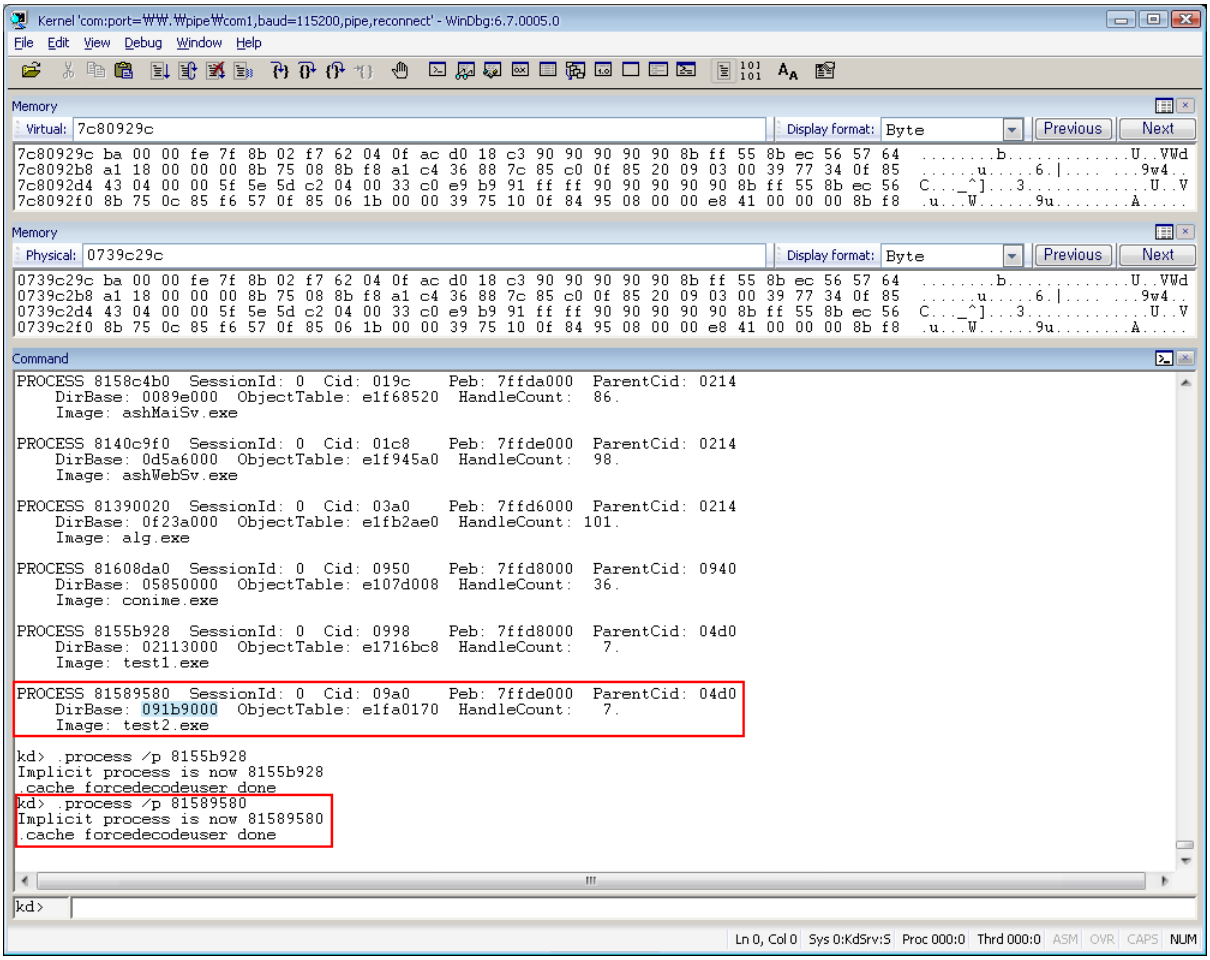
하지만 우리의 염려와는 달리 Windows에서 DLL은 공유된다. 그 사실을 확인하기 위해서는 test1과 test2가 참조하는 GetTickCount 함수에 대한 실제 물리주소를 살펴보는 것이다(이미 알고 있겠지만 위 그림에서 보여지는 0x7c80929c 는 가상주소지 물리주소가 아니다). 만약 두 프로세스가 참조하는 물리주소가 같다면 DLL은 공유된다고 볼 수 있다. 예제를 통해 살펴보자.



<그림 3> test1 – GetTickCount 함수의 물리주소 확인

.process /p 8155b928 은 windbg에 현재 보여지는 가상메모리를 test1프로세스의 가상메모리로 바꾸는 명령어이다. 즉, 현재 메모리 컨텍스트를 해당 프로세스의 메모리 컨텍스트로 바꾸어준다는 말이다(여기서 8155b928 은 test1을 나타냄).

<그림 3>에서 보는 바와 같이 test1에서 GetTickCount 함수의 가상주소는 0x7c80929c, 물리주소는 0x0739c29c 이다. 그리고 실제로 두 주소의 메모리의 값이 같은 것을 보면 동일한 함수임을 알 수 있다(<그림 2>와도 비교해 보라).



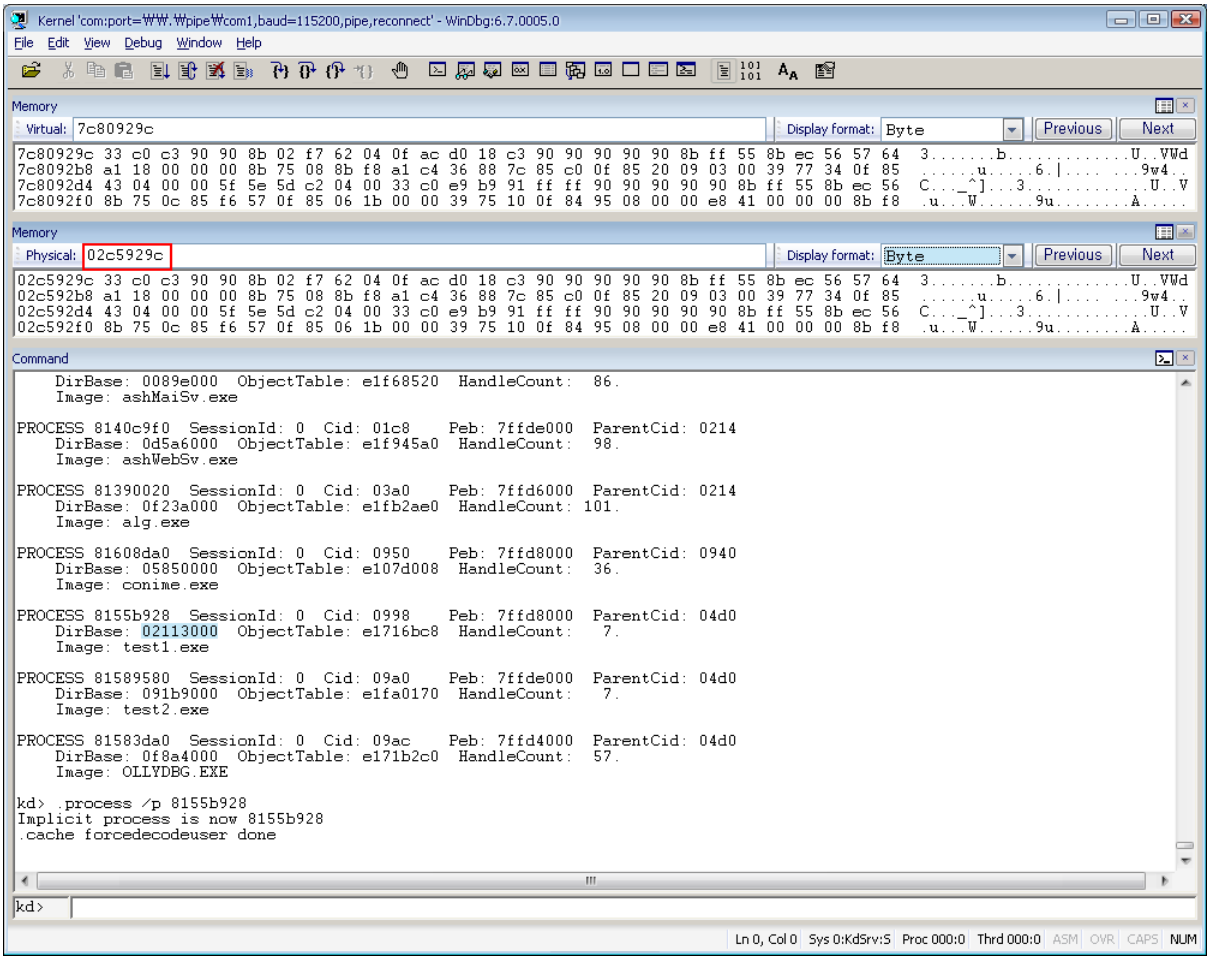
<그림 4> test2 – GetTickCount 함수의 물리주소 확인

<그림 4>의 test2도 역시 <그림 3>과 같음을 볼 수 있다. 그러므로 DLL은 공유된다고 볼 수 있다.

어떻게 물리주소를 구하는지에 관한 설명은 [Ref]을 참조하기 바란다.

그렇다면 어떻게 test1이 참조하는 함수를 수정해도 test2는 영향을 받지 않는 것일까? 그것은 바로 COW(Copy-On-Write)라는 개념 때문이다. Windows는 COW라는 개념을 이용하는데, 이 개념은 프로세스(test1)가 참조하는 DLL함수를 수정하려고 하면, Windows는 해당 DLL함수를 메모리의 다른 곳에 복사해 두고 복사되어진 메모리에 있는 DLL함수를 수정한다. 그리고 나서 프로세스의 페이지테이블 주소를 복사된 메모리 주소으로 바꾸게 된다. 그렇게 되면 다른 프로세스에서는 (test2)는 처음과 같이 0x0739c29c 주소의 함수를 참조하기 때문에 전혀 영향을 받지 않게 된다.

그럼 여기서 test1이 참조하는 GetTickCount 함수를 수정하고 나서 실제 물리주소가 바뀌었는지 확인해보도록 하자.



<그림 5> test1 – GetTickCount 함수 수정후 물리주소 확인

위 <그림 5>에서 보는 바와 같이 GetTickCount 함수를 수정 후 실제 물리주소는 0x02c5929c 로 바뀌었음을 확인할 수 있다(<그림 1>과 비교해 보면 기계어 코드가 동일함을 확인할 수 있다).

대단한 내용은 아니지만 저와 같은 고민에 빠진 사람들이 있다면 조금이나 도움이 되기를 바라는 마음에 작성하게 되었습니다.

혹시 틀린 부분이나 수정되어야 할 부분이 있다면 k0nni3@gmail.com 으로 메일 보내주시면 감사하겠습니다.

[Ref]

- Windows 구조와 원리 그리고 Codes