안티바이러스 자가보호 무력화에 대한 연구

구사무엘 , 김슬기

A Study on The Security Vulnerabilities in Self Protectionof Anti Viruses

Samuel Koo , Seul-Gi Kim

1. 서론

2009년 7.7 DDoS 공격에 이어 지난 26일 SK Communications가 해킹으로 인해 3500 만명의 네이트 회원 개인정보가 정보가 유출 되었으며, 분석 결과 피해자의 컴퓨터에서 안티 바이러스에 감지되지 않은 악성코드가 심겨져 있었으며, 이 악성코드의 유포 경로로써 이스트소프트의 업데이트 서버를 경유하였다는 점에서 큰 충격을 주고 있다. 앞의 공격들 이후, virus의 탐지 특히 signatured virus가 아닌, targeted virus에 대한 대응의 일환으로 heuristic 탐지 기법이 강화되고 있으며, CC-RA 에 안티 바이러스의 자가보호 기능에 대한 요구사항이 들어가는 등 많은 대응책을 마련하고 있다.

하지만 공격자가 안티바이러스를 무력화 시키는 것이 가능하다면, 안티 바이러스가 성공적으로 virus의 탐지를 하는 것이 가능한지 의문이 제기되고 있다. 이에 최신 안티바이러스들의 자가보호 기능에 대하여 살펴보고, 향후 발생 가능한 공격 유형을 제시함으로써 그에 대한 대응방안을 개발하고 적용하여 대응능력을 향상시키고자 한다.

2. 국내 안티 바이러스 점유율

안티 바이러스의 취약점에 대하여 논하기 전에, 국내의 일반 사용자들에 백신 점유율을 알아보면 2월을 기준으로 E사 A 45.24%, A사 V 26.12%, D사 DC 17.75%, 그리고 N사 NV 8.58% 으로 통계 되었다. 이 결과를 볼 때, 국내 시장은 무료 백신이 큰 점유율을 가지고 있으며, 앞서 제기된 4가지 백신에 대한 바이러스를 만들 경우, 국내 일반 사용자들은 공격의 범위에 포함될 수 있다는 것을 의미한다. 그럼으로, 본 연구는 점유율이 가장 높은 4가지 안티 바이러스를 기준으로 분석을 하여보고자 한다.

3. 자가보호

자가보호란 무결성이 유지되어야 하는 안티바이러스가 자신이 사용하는 파일, 레지스트리, 프로세스, 서비스, 커널객체(이벤트, 뮤텍스, 네임드 파이프, 메일 슬롯, 공유 메모리) 등을 외부에서 접근하는 것을 관리 또는 차단하는 것을 의미한다.

3.1. 자가보호 방법

3.1.1. 유저 후킹

유저 후킹의 방법으로는 API(Application Programming Interface)에 대한 IAT (Import Address Table) 후킹, EAT (Export Address Table) 후킹, Detour (Inline Function Hooking) 등이 많이 사용되고 있다.

Windows 시스템에서 프로세스간의 메모리 영역이 분리되어 있음으로, 전역 후킹을 위해서는 시스템 전체의 프로세스에 DLL을 삽입하거나 코드 조각을 삽입하여야 한다.

3.1.2. 커널 후킹

커널 후킹의 방법으로는 NTAPI(Native Application Programming Interface)에 대한 SSDT (System Service Descriptor Table) 후킹, IDT(Interrupt Descriptor Table) 후킹, SYSENTER, 필터드라이버, IRP(I/O Request Packet) 후킹 등이 많이 사용되고 있다. Windows 시스템에서 커널 모듈은 하나의 메모리 공간을 공유한다. 이 영역에서의 후킹은 한번의 설치로 시스템 전체에 영향을 미칠 수 있는 장점으로, 악성코드 제작자와 안티바이러스 제작자 모두 선호하는 위치이다.

3.2. 자가보호 대상

3.2.1. 파일 보호

파일 보호의 목적은 안티바이러스 구성요소의 무결성 보장에 있다. 파일의 구성요소는 일반적으로 실행파일 (EXE, DLL, SYS)와 데이터 파일 (DAT, LOG, CONF)가 있다.

3.2.2. 레지스트리 보호

레지스트리 보호의 목적은 안티바이러스 설정의 무결성 보장에 있다. 레지스트리는 안티바이러스의 라이선스 정보, 검사 예외 설정, 그리고 최종 검사 날짜 등의 민감한 정보가 존재할 가능성이 높다.

3.2.3. 프로세스 보호

프로세스 보호의 목적은 안티바이러스의 실행 보장에 있다. 프로세스는 안티바이러스의 실행 상태의 대표적 형태이며, 프로세스 보호는 실행의 단위인 쓰레드에 대한 보호를 포함한다. 프로세스의 무결성이 보장 받지 못할 경우, 안티 바이러스는 정상적 동작을 보장 받지 못한다.

3.2.4. 서비스 보호

서비스 보호는 프로세스 보호의 특수한 형태이다. 서비스란 Windows 시스템에서 특별한 권한을 가지는 프로세스를 말하며, 많은 안티바이러스의 실시간 감지가 서비스에 존재한다.

3.2.5. 드라이버 보호

드라이버에는 안티바이러스에 핵심적인 부분과 자가보호가 구현되어 있을 가능성이 크다. 자가보호에 대해 주 관리하는 부분임으로, 드라이버에 대한 무결성은 상당히 중요하다.

3.2.6. 커널객체 보호

커널객체는 안티바이러스가 구동됨에 있어서 필요한 동기화 객체 및 드라이버 또는 서비스와 I인터 프로세스 커뮤니케이션 함에 있어서 필수적인 요소들이며, 해당 요소에 대한 무결성을 보장 받지 못할 경우, 이 후 안티바이러스의 동작을 예측할 수 없게 된다.

4. 자가보호 지원 여부

8월 14일 현재를 기준으로 안티 바이러스들의 버전과 자가보호 기능 지원 여부는 다음과 같다.

	안티바이러스 이름	최신 버전	지원여부
1	E사 A	1.55 , 2.0 Beta	O
2	A사 VL	2011.08.14.00	O
3	I사 A2	3.0	O
4	H사 V2	2011	O
5	D 포털 DC	1.5.0.84	X
6	N 포털 NV	1.0.78	X

A, V, A2, V2 는 자가보호를 지원하고 있으며, D포털 DC와 N포털 NV은 지원하고 있지 않음을 알 수 있다. 전 세계 시장에서의 점유율을 기준으로 최상위에 존재하는 AVG, AVAST, NORTON, MCAFEE, ESET NOD32, AVIRA, SYMANTEC 등이 자가보호를 전부 제공하고 있다는 점에서 다음 클리너와 네이버 백신의 자가보호 미 지원은 의문이 드는 부분이다.

5. 관련연구 및 제안하는 방법

과거에는 자가보호 기능이 존재하지 않아, 안티 바이러스 공격을 대상으로 하는 악성코드들이 어렵지 않게 백신들을 무력화 시키는 것이 가능 하였다. 그러나 최근에는 대부분의 안티 바이러스들이 자가보호 기능을 탑재하여, 과거 보다는 비교적 백신에 대한 공격이 어려워지게 되었다. 그럼으로, 악성코드 제작자들은 안티 바이러스의 자가보호에 대한 연구를 많이 진행하고 있을 것으로 예상된다.

5.1. TerminateProcess 공격

자가보호 기능이 존재하지 않았던 과거에 많이 사용되었던 방법으로 안티 바이러스의 프로세스를 Windows가 지원하는 API인 TerminateProcess를 이용하여 강제 시키는 공격으로써, 안티 바이러스 프로세스를 전체 프로세스 목록에서 검색하여 찾은 후, 해당 프로세스에 대하여 OpenProcess API를 PROCESS_TERMINATE 권한으로 호출하여, 핸들을 획득하고, TerminateProcess API를 이용하여 안티 바이러스를 강제 종료시키는 공격이다.

5.2. WinStationTerminateProcess 공격

TerminateProcess 공격과 유사하나, Windows의 구성요소인 svchost.exe는 안티 바이러스에게 신뢰되는 프로세스일 가능성이 높다는 점에 악용하여, svchost.exe 에게 특정 프로세스의 종료를 요청하는 API인 WinStationTerminateProcess API를 안티 바이러스 프로세스를 대상으로 하여 호출함으로써, 안티 바이러스를 강제 종료 시키는 공격이다.

5.3. DebugActiveProcess 공격

본 공격은 프로세스를 직접 종료 시키는 것이 아닌, 공격자가 만든 악성코드가 안티 바이러스의 디버거 프로세스로 설정함으로써, 안티 바이러스 프로세스에 통제권을 얻는 것이 목적인 공격이다. 실행중인 프로세스의 디버거를 지정하는 API인 DebugActiveProcess를 안티 바이러스 프로세스를 대상으로 하여 호출함으로써, 안티 바이러스를 악성코드의 통제 범위에 넣는 공격이다.

5.4. DebugBreakProcess 공격

특정 프로세스에서 중단점 예외를 발생 시키는 API인 DebugBreakProcess를 안티 바이러스를 대상으로 호출함으로써, 안티 바이러스 프로세스를 Suspend 상태로 만들어, 무력화 시키는 공격이다.

5.5. Terminating Thread 공격

안티 바이러스의 자가보호가 프로세스에 대해서만 보호 하도록 잘못 구성되어 있을 경우, 가능한 공격으로써, 프로세스의 모든 쓰레드가 파괴될 경우 프로세스가 종료된다는 원리에 기반하고 있는 공격이다.

5.6. Suspending Thread 공격
안티 바이러스의 자가보호 기능에 의해 쓰레드 종료가 막혀 있을 경우, 쓰레드의 상태를 Suspend로 만들 수 있는 API인 SuspendThread를 사용하여 안티 바이러스의 모든 쓰레드를 Sus아무 동작도 할 수 없는 suspend 상태로 만들어, 무력화 시키는 공격이다.

5.7. NULL Context Overwrite 공격
쓰레드의 context를 변경시킬 수 있는 API인 SetThreadContext를 이용하여 안티 바이러스 프로세스의 context의 모든 값을 0으로 만듬 으로써, 안티 바이러스 프로세스를 비정상 종료 시키는 공격이다.

5.8. Create Abnormal Thread 공격
특정 프로세스에 새로운 쓰레드를 생성할 수 있는 API인 CreateRemoteThread를 이용하여 안티 바이러스 프로세스에 시작 지점을 잘못된 주소 또는 호출 프로세스 자신을 종료 시키는 API인 ExitProcess를 가리키게 함으로써 안티바이러스 프로세스의 자가 종료를 유도하는 공격이다.

5.9. Memory NoAccess 공격
메모리의 접근권한을 변경할 수 있는 API인 VirtualProtectEx를 이용하여 안티 바이러스 프로세스에 모든 메모리 권한을 NoAccess로 변경시킴으로써, 안티 바이러스 프로세스의 비정상 종료를 유도 시키는 공격이다.

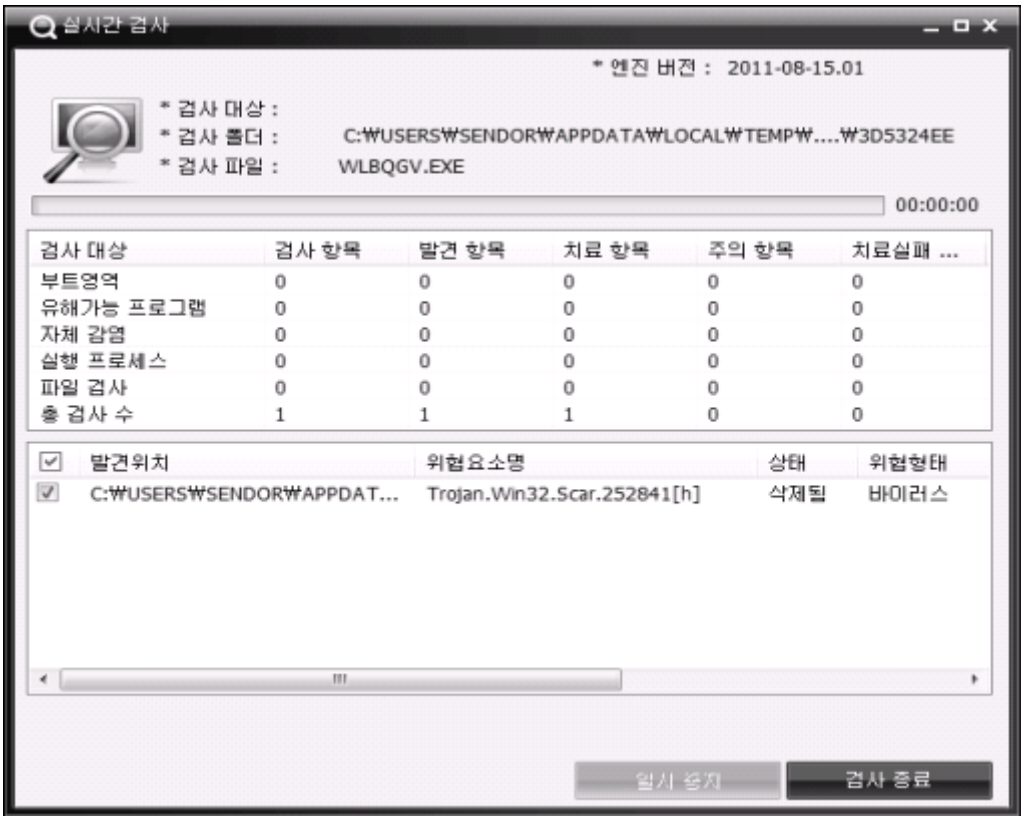
5.10. NULL Memory Overwrite 공격
특정 프로세스에 메모리를 기록하여 주는 API인 WriteProcessMemory를 이용하여 안티 바이러스 프로세스의 모든 메모리 공간에 NOP(No Operation) 코드를 기록함으로써, 안티 바이러스 프로세스의 비정상 종료를 유도 시키는 공격이다.

5.11. Duplicate Handle from Maximum Allowed 공격
안티 바이러스의 프로세스에 대한 자가보호가 후킹으로 되어 있을 경우, 가능한한 최대의 권한으로 핸들을 얻어주는 Flag인 MAXIMUM_ALLOWED를 OpenProcess 또는 OpenThread API와 함께 사용하여 핸들을 얻어 낸 후, 핸들을 복사하여 주는 API인 DuplicateHandle을 이용하여 더 높은 권한으로 복사 시켜, 4.1 ~ 4.10 공격 중, 권한의 부족으로 이루어지지 못했던 것을 우회하여 공격하는 것이 가능하다.

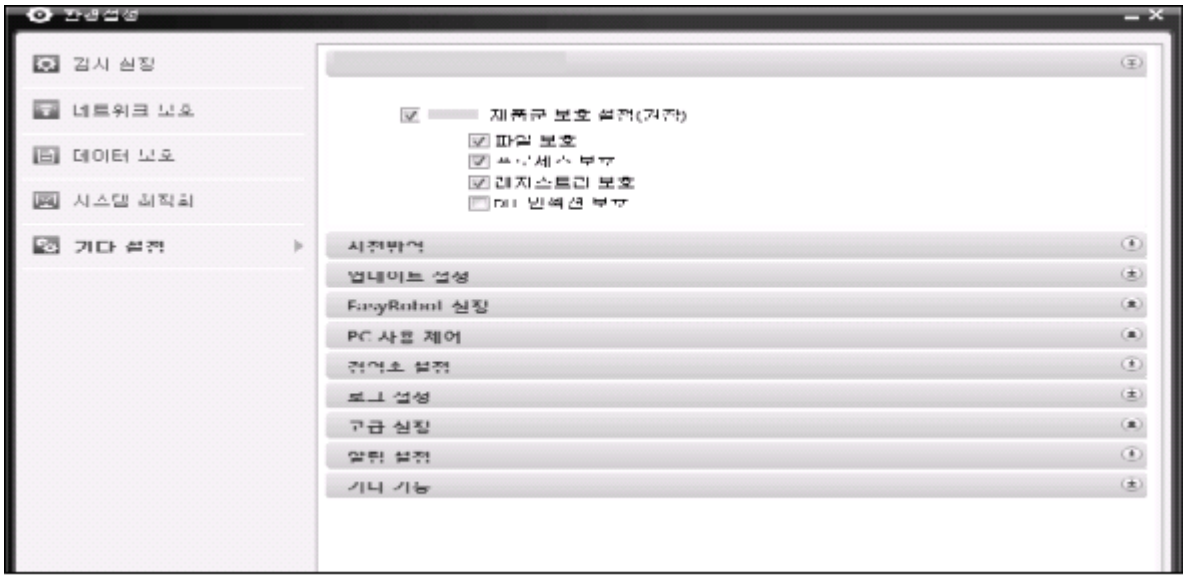
5.12. Duplicate Handle from CSRSS 공격
Windows 시스템 상에 존재하는 모든 프로세스의 핸들은 csrss.exe 프로세스 안에 존재하며, 이 핸들을 DuplicateHandle API를 이용하여 복사함으로써 후킹을 이용하여 구성된 안티 바이러스의 자가 보호를 우회하여 권한을 얻어 낸 후, 4.1 ~ 4.10 공격 중, 권한의 부족으로 이루어지지 못했던 것을 우회하여 공격하는 것이 가능하다.

6. 실험 및 결과

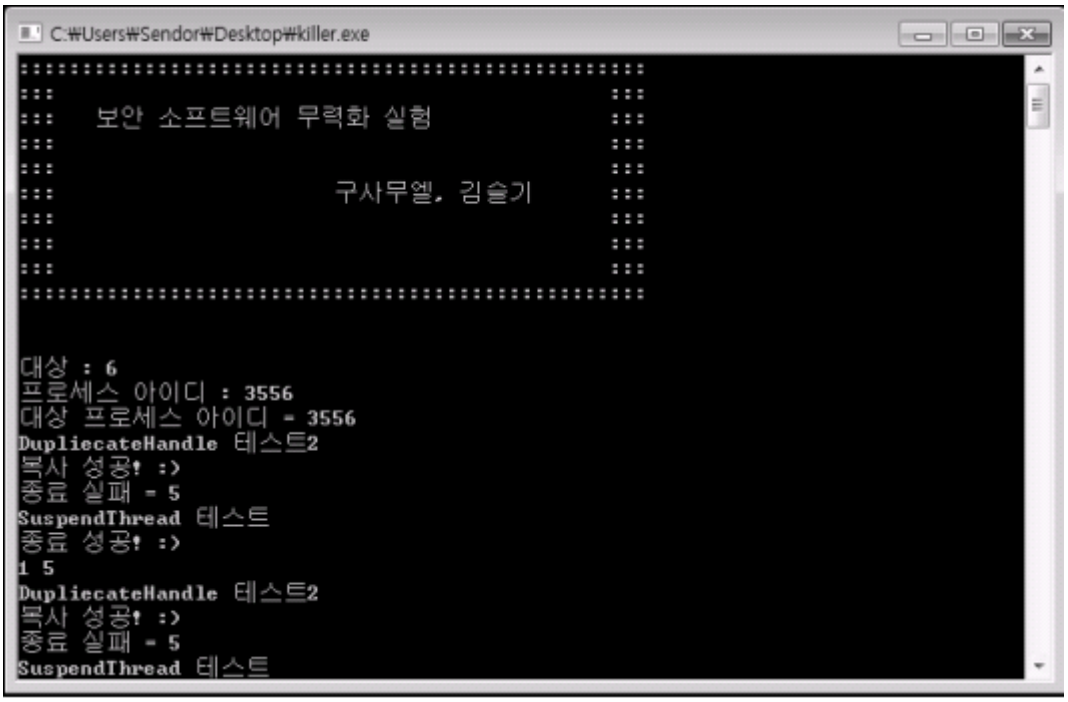
6.1. 실험 환경 및 데이터
제안한 방법 및 알고리즘을 C++ 개발환경인 Visual Studio를 이용하여 구현하였다. 또한 실험을 위해서 Trojan.Win32.Scar.252841 를 수집 하였으며, 수집한 악성코드는 윈도우 상에서 실행될 수 있는 DDoS 에이전트(Agent)이며 악성코드의 진단명은 H사 V 안티바이러스의 진단명을 따른다.수집한 악성코드를 안티바이러스를 무력화 전 과 후 실행시키어 보고, 변화를 관찰 하였다.



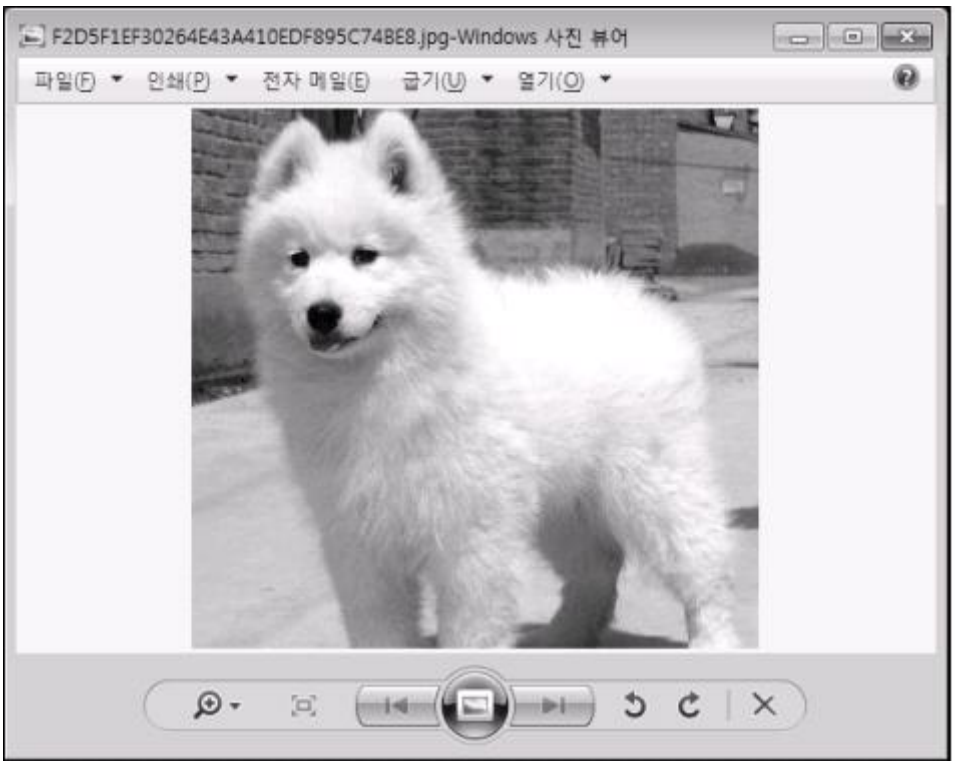
안티바이러스를 무력화하기 전, 악성코드는 안티바이러스 소프트웨어에 의해 감지 및 삭제되는 것을 볼 수 있었다. 안티바이러스 무력화에 대한 테스트를 진행하기 전, 정확한 실험을 위하여 안티바이러스의 설정에서 자가보호 기능의 동작 여부를 다시 확인 하였다.



안티바이러스의 설정이 올바르게 되어 있음을 확인하였음으로, 제안에 대한 구현을 실행하여 안티바이러스 프로세스에 대하여 무력화 실험을 수행 하였다.



안티바이러스 소프트웨어가 작동을 멈추었을 것으로 예상됨으로, 무력화 전 감지 후 삭제 되었던 악성코드를 다시 다운로드 후 실행시키어 보았다.



악성코드가 안티바이러스에 의해 검출되지 않고, 정상적으로 실행된 것을 볼 수 있다.
안티바이러스 제품 별 단계에 차이는 있었으나, 모두 무력화 되었다. 무력화 후 10초 정도 시스템 딜레이가 발생하는 경우가 존재했으나, 그 후엔 발생하지 않았다. D사의 DC의 경우 해당 악성코드를 잡아내지 못하여, 실험이 불가능 하였으나, 자체보호가 존재하지 않아 쉽게 종료가 가능함과, 시작프로그램 감시 기능의 우회를 통해 가능 여부를 확인 하였다.

7. 결론 및 향후 연구

안티바이러스의 자가보호의 잘못된 신뢰범위 지정 및 보호에 대하여 공격할 수 있는 방법을 제안 하였다. 제안한 방법은 잘못된 API 신뢰범위를 가지는 안티바이러스 자가보호 시스템에서 나타날 수 있는 취약점을 정리하고, 악성코드 샘플의 검출 여부 실험을 통해 자가보호 및 실시간 탐지 우회 여부에 대하여 연구 하였다. 그러나 본 연구는 지나치게 적은 안티바이러스 샘플 수와 악성코드 샘플 수로 인해, 높은 오차를 가질 수 있으며 향후 연구에서는 다양한 클래스의 안티바이러스를 대상으로 제안한 방법을 다수의 악성코드를 가지고 적용하여 보다 개선된 무력화 방법을 연구하고자 한다.

참고 문헌

[1] 김성우, 해킹 파괴의 광학 개정판, 와이미디어, 2006.
[2] D. Barroso, “ "Botnets-The Silent Threat”", ENSIA Position Paper, no. 3, pp. 1-9, November 2007.
[3] Greg Hoglund, James Butler, Rootkits: Subverting the Windows Kernel, Pearson Education, Inc, 2006.
[4] Charles Petzold, Programming Microsoft Windows 5th Edition, Microsoft Press, 2002.
[5] 강태우, 조재익, 정만현, 문종섭, “ PI call의 단계별 복합분석을 통한 악성코드 탐지” 정보보호학회논문지, 제17권 제6호, pp. 89-98, 2007년 12월.
[6] 김완경, 소우영, “ 윈도우 XP 커널 기반 API 후킹 탐지 도구 설계 및 개발” 보안공학연구논문지, 제7권
제4호, pp. 385-397, 2010년 8월.
[7] 박남열, 김용민, 노봉남, “ 우회기법을 이용하는 악성코드 행위기반 탐지 방법” 정보보호학회논문지, 제16권 제3호, pp. 17-28, 2006년 6월.