강제적 접근제어를 통한 프로세스 메모리 보호

심종익* · 박태규* · 김진태**

Protecting Memory of Process Using Mandatory Access Control

Jong-Ik Shim* · Tae-Kyou Park* · Jin-Tae Kim**

요 약

널리 사용되고 있는 윈도우 운영체제의 보안 취약성으로 인한 다양한 형태의 공격에 의한 침입, 자료 유출 및 무단 변조 등의 문제점이 발생되고 있다. 본 논문에서는 윈도우 운영체제의 커널 수준에서 허가되지 않은 사용과 침입을 차단하고, 프로세스 메모리에 대한 접근을 강제적으로 제어할 수 있는 다중등급보안(Multi Level Security) 시스템을 구현하였다. 신뢰성 컴퓨터 시스템 평가기준(TCSEC)의 B1 등급에서 요구하는 대부분의 기능을 윈도우 운영체제 커널에 구현하였으며, 이를 확장하여 주요 프로세스의 메모리에 대한 변조를 차단하기 위한 기능을 보안 커널에 추가하였다.

ABSTRACT

There are various attacks such as tampering, bypassing and spoofing which are caused with system-wide vulnerabilities of Windows operating system. The underlying operating system is responsible for protecting application-space mechanisms against such attacks. This paper provides the implementation of mandatory access control known as multi-level security (MLS) rating with TCSEC-B1 level on th kernel of Windows™. By adding especially the protection feature against tampering memory of processes to the security kernel, this implementation meets the responsibility against system-wide vulnerabilities.

키워드

프로세스 메모리, 접근 제어, 다중등급보안, 시스템 평가기준, 운영체제 커널

Key word

Process memory, Access control, Multi-level security, TCSEC, Kernel of Windows™

* 정회원 : 한서대학교 컴퓨터정보공학과
** 정회원 : 한서대학교 컴퓨터정보공학과 (교신저자, jtkim@hanseo.ac.kr)

접수일자 : 2011. 04. 27
심사완료일자 : 2011. 06. 01

Ⅰ. 서 론

인터넷의 확산과 더불어 정보보호에 대한 관심이 증가하고 있고, 인터넷을 이용하여 기업 활동을 하거나 상거래를 하고자 하는 수요가 폭발적으로 늘어나면서 디지털 보안은 선택이 아닌 필수 요건이 되었다. 최근 들어 악성코드와 관련된 보안 사고는 대부분 윈도우 운영체제에서 발생하고 있으며, 해마다 증가하는 추세를 보이고 있다. 갈수록 지능화되고 있는 악성코드들은 윈도우 커널 내에서 자신의 정보를 은폐하고 공격코드를 추가하는 방식으로 동작하고 있기 때문에 기존의 대응책으로는 해결할 수 없는 것이 현실이다.

국가의 중요 정보통신 기반구조를 보호하기 위한 기술은 정보통신 기반구조 방어기술 및 공격기술로 분류할 수 있으며, 정보시스템을 보호하기 위한 방어기술의 핵심은 보안운영체제 기술로 인식되고 있다[1]. 국가 정보통신 기반구조는 정보 통신망 및 정보 시스템으로 구성되어 있으므로 정보통신 기반구조의 구성 요소를 보호하기 위해서는 근본적으로 보안 커널 기술을 개발해야 한다.

예전에 이루어진 컴퓨터 시스템의 보안기술 연구로는 컴퓨터 시스템에 암호화, 접근제어, 감사 추적 등의 하드웨어, 소프트웨어를 추가하는 애드온(Add-on) 방식이 진행되어 왔다. 그러나 이러한 방식으로는 기존에 알려진 우회, 수정 등의 문제점 및 새로운 보안상의 문제점인 트로이 목마, covert channel 등의 신종 컴퓨터 범죄를 해결할 수 없고, 이러한 문제점이 발생할 때마다 새로운 보안 업데이트를 계속해서 추가해야하는 문제가 있다.

또한, 최근 들어 정보화 분야가 비약적으로 확대됨에 따라 컴퓨터에 저장된 비밀 수준의 정보들에 대한 접근 권한이 없는 내부 및 외부 사용자, 또는 프로세스 주체가 객체인 비밀 파일 또는 장치에 임의적으로 접근하여 불법적으로 객체를 읽고 쓰거나 수행하는 등의 보안 문제가 발생하고 있다. 특히, 웹을 통한 콘텐츠 서비스에 있어서는, 해커 등에 의한 시스템 루트 권한의 도용과 접근 권한이 없는 사용자 등에 의한 홈 페이지 및 관련 콘텐츠의 불법 변조 문제가 사회적 물의를 일으키고 있다.

이에 따라, 최근에는 컴퓨터 시스템의 운영체제의 내부 커널에 보안 기능을 포함시키는 보안 커널의 연구 개발이 진행되고 있다. 이런 보안 커널의 개발에 따라 최근에는 사용자 또는 프로세스인 주체가 객체인 비밀 파일 또는 장치에 접근시에 보안성이 강화되었다.

이미 미국을 비롯한 선진국은 신뢰성 있는 운영체제를 연구 개발하여 운영체제 커널 수준에 보안을 탑재한 컴퓨터 시스템을 널리 사용하고 있다. 국내에서도 그 동안 각 분야에서 운영체제 시스템을 개발하면서 축적된 커널 분석 및 설계 기술과 학계, 연구소, 산업체 등에서 보유하고 있는 컴퓨터 보안 기술을 접목시켜서, 국내 자체의 기술진에 의하여 안전한 운영체제를 커널 수준에서 연구하여, 독자적인 정보보호 원천기술을 확보하기 위한 연구가 진행되어 왔으며 그 결과 가시적인 성과가 나타나고 있는 상황이다. TCSEC(Trusted Computer System Evaluation Criteria)의 B1급 이상 컴퓨터 시스템에서는 대부분 보안 커널을 구현하고 있다[2].

본 논문에서는 윈도우 운영체제에 대한 다양한 공격 유형과 기법을 정형화하고, 이를 기반으로 강제적 접근 제어 기법과 프로세스 메모리 변조 차단 기법으로 구분하여 효과적인 대응 방안과 메커니즘을 제안한다. 알려지지 않은 윈도우 커널 정보 및 관련 메커니즘의 수집과 분석을 통하여 윈도우 운영제제 공격에 대한 대응 기술 구현에 적극 활용하였으며, 시스템 활용도와 안정성을 극대화할 수 있도록 구현하였다.

2장에서 현재 윈도우 운영체제에서 문제점으로 대두되고 있는 보안 현황과 이를 해결할 수 있는 원천 기술인 보안 커널에 대해 설명하고, 3장에서는 윈도우 커널에서의 다중등급보안 모델과 보안레이블 그리고 강제적 접근제어를 중심으로 기술하고, 4장에서는 구현된 강제적 접근제어 시스템을 확장하여 구현된 프로세스의 메모리에 대한 접근제어 개발 방법론에 대하여 기술한다. 그리고 5장에서 결론을 맺는다.

Ⅱ. 윈도우 운영체제의 보안 현황 및 보안 커널

2.1 윈도우 운영체제의 보안 현황

윈도우 운영체제는 여러 가지 보안상의 문제점이 내재하고 있지만 그 중에서도 가장 큰 문제점으로 대두되는 것은 시스템 관리자인 Administrator에게 권한이 집중

되어있다는 것이다. 윈도우 운영체제에서 Superuser의 권한을 획득하면 프로세스뿐만 아니라 모든 자료에 대한 모든 권한을 취득할 수 있다는 것을 의미한다.

윈도우 운영체제에서 침입 방법으로 이용되는 몇 가지 예를 보면 Buffer Overflow, Worm, IP Spoofing, Sniffing, Trojan Horse, Virus, Administrator 권한 탈취, Denial of Service, 프로세스 메모리 변조, 사전 공격 등이 있다. 이러한 문제점의 대응책으로 여러 가지가 있지만 가장 안전한 방안은 보안 커널이라 할 수 있다. 현 윈도우 운영체제의 커널 수준에서의 보안 연구는 보안의 취약성에 비하여 상당히 부족한 상황이다.

2.2 보안 커널

지금까지의 정보보호 기술은 거의 대부분이 네트워크 또는 응용프로그램 수준에서 연구되어 왔으나, 이러한 방식으로는 기존에 알려진 우회, 수정 등의 문제점 및 새로운 보안상의 문제점인 신종 컴퓨터 범죄를 해결할 수 없다. 더불어 문제점 발생 시 마다 보안제품을 계속적으로 추가하는데 드는 비용과 번거로움이 발생되어 왔다. 이러한 문제점을 원천적으로 해결하고자 컴퓨터 시스템의 운영체제 내부 커널에 보안기능을 포함시키는 보안 커널의 연구 개발을 추진하여 제품을 생산하고 있는 추세이다. 즉, 철저한 정보보호체제를 지원하기 위해서는 운영체제 내부에 보안 커널 기능이 구현되어야만 응용계층에서 시도되는 원문 변조, 우회경로에 의한 불법접근, 프로세스 메모리 변조 등에 의한 공격을 원천적으로 봉쇄할 수 있다[3-6]. 즉 엄격한 보안이 요구되는 국가기관, 기업 등에서 내부자에 의해 이루어지는 정보 유출을 근본적으로 차단하고 보다 세분화된 보안 체제를 구축하기 위해서는 커널 기반의 강제적 접근통제 시스템이 필수적으로 요구된다.

기존에 개발된 보안 커널 또는 운영체제의 경우에는 객체를 크게 디스크에 저장된 정규 파일, 디렉터리, 특수 파일인 장치파일 등으로 구성하였다. 따라서, 사용자 또는 프로세스의 주체가 파일, 디렉터리 또는 장치파일에 접근할 경우에는 강화된 접근제어 방식으로 보안성을 유지할 수 있었다. 그러나 프로세스가 소유하고 있는 프로세스 메모리에 대한 접근 제어 및 보안 기술은 그 연구 및 개발이 부족한 점이 많았다.

따라서, 프로세스 메모리에 대한 해킹 및 덤프로 프로세스 메모리상에 적재된 정보들이 외부로 유출될 수 있는 보안상의 문제점이 발생할 수 있다. 기존의 보안 커널 및 운영체제로는 이러한 문제를 원천적으로 해결하기 힘든 문제가 있다.

본 논문에서는 상기와 같은 문제점을 해결하기 위해, 컴퓨터 시스템에서 일정한 보안 등급을 보유하는 주체와 객체 사이의 등급과 카테고리 관계에 수정된 BLP(Bell & Lapadula) 보안모델을 적용하였다. 여기서, 주체는 프로세스가 비밀 수준의 보안 등급이고, 객체는 프로세스 메모리를 읽거나(read), 쓰거나(write) 하는 것을 말한다. 보안 모델을 통해 접근을 제어함으로써 보안 등급을 가지는 프로세스가 타 프로세스에 의해 프로세스 메모리의 정보가 유출 또는 변형되지 않도록 보호될 수 있는 강제적 프로세스 메모리 접근 제어 방법을 제공한다.

나아가, 컴퓨터 시스템에 응용프로그램이 설치될 때에 보안 관리자로 하여금 보안 등급 및 카테고리 정보를 배정하고 이를 설정하도록 하여, 응용프로그램이 실행되어 생성되는 프로세스와, 프로세스에 의해서 생성되는 자식 프로세스에 보안 관리자가 설정한 보안 등급 및 카테고리 정보가 계속 상속될 수 있도록 하는 방법을 제공한다.

Ⅲ. 다중등급보안 시스템의 설계

3.1 다중등급보안 모델

다중등급보안(MLS : Multi-Level Security)이란 주체에 보안등급(Clearance)과 보호범주(Category)가 부여되어 보안등급과 보호범주가 부여된 임의의 객체에 접근하는 것을 통제하는 방식으로서 직책과 직무에 따라 보안권한이 엄격히 요구되는 조직에서 필수적으로 요구되는 보안 기능이며, MLS 정책은 다중등급 주체와 객체 간의 상호작용을 지원하는 BLP(Bell & LaPadula) 모델의 한 형태이다[7]. 즉, 주체와 객체에 보안레이블을 부여하고 강제적 접근제어 정책을 사용하여 다중등급보안을 시행하게 된다.

3.2 다중등급보안에서의 개발 요구사항

다중등급보안 시스템은 보안 관리자가 설정한 보안 등급 및 보호범주가 일치하는 사용자에게만 자료에 대한 접근을 허용하는 방식으로서, 직책과 직무에 따라 보

안권한이 엄격히 요구되는 조직에서 필수적으로 요구되는 보안 기능이다. 본 연구에서는 커널 기반의 다중등급보안 시스템을 개발하고, 이를 응용하여 프로세스 메모리의 변조를 차단할 수 있는 방안을 제시한다.

본 논문에서 제시하는 MLS 윈도우 시스템 개발에서의 주요 요구기능은 다음과 같다.

먼저, 모든 주체와 객체에 보안등급과 보호범주라고 하는 보안레이블을 부여한다. 기존의 연구에서 대상으로 하는 파일 및 디렉터리는 물론 추가적으로 특정 프로세스가 소유하고 있는 메모리 영역도 하나의 객체로 간주하여 보호한다.

보안운영체제는 주체인 프로세스가 보안등급이 부여된 모든 정보 객체에 접근할 때마다 권한을 체크하여 접근을 강제적으로 통제한다. 보안 권한과 속성을 프로세스가 임의로 변경할 수 없으며, 비밀로 분류된 자료는 오직 보안운영체제의 허가를 받아야만 접근할 수 있다. 강제적 접근제어는 TCSEC의 B1급 이상의 컴퓨터에서 반드시 요구되는 중요한 기술로서 파일, 디렉터리뿐만 아니라 보안등급이 부여된 프로세스의 메모리 영역에도 동일하게 적용되도록 구현하였다.

3.3 다중등급보안에서의 접근제어

본 논문에서 사용하는 다중등급보안 방식에 기초한 강제적 프로세스메모리 접근 제어 방법은, (a) 시스템 보안 관리자로 하여금 시스템에 설치되는 응용프로그램에 보안 등급과 카테고리 정보를 배정하여 설정하도록 하는 단계, (b) 시스템 내에 설치된 응용프로그램이 실행되어 프로세스가 생성될 때 프로세스 구조체에 필드를 할당하고 해당 응용프로그램에 설정된 보안 등급과 카테고리 정보를 수록하는 단계, 및 (c) 실행중인 프로세스가 주체가 되고 다른 실행중인 프로세스를 객체로 하여 주체가 객체의 프로세스 메모리를 액세스할 때, 상기 객체의 프로세스 구조체에 수록된 보안 등급과 카테고리 정보를 서로 비교하여 수정된 BLP 모델에 따라 객체의 프로세스 메모리에 대한 주체의 접근 권한을 커널모드에서 결정하는 단계 등을 포함한다.

강제적 접근제어는 그림 1과 같이 주체의 보안레이블과 주체가 접근하고자 하는 객체의 보안레이블을 비교하여 보안정책에 따라 접근을 제어하는 방법이다.

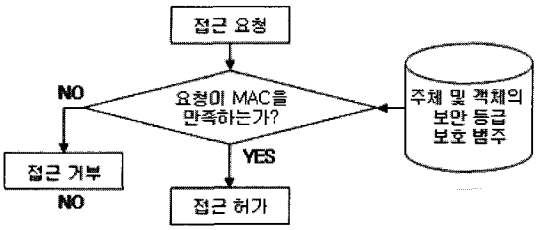

그림 1. 강제적 접근제어
Fig. 1. Mandatory access control

강제적 접근제어의 판단 기준이 되는 보안레이블은 접근제어의 대상이 되는 주체 및 객체의 중요도를 나타내는 정보이다. 즉, 객체에 포함된 정보의 비밀성과 이러한 비밀 정보에 대하여 주체가 갖는 정형화된 권한에 근거하여 객체에 대한 접근을 제한하는 방법으로 한 주체와 한 객체간의 접근제어 관계를 정의한다. 데이터에 대한 접근은 주체와 객체가 갖는 보안등급의 정의를 통한 강제적인 정책에 의하여 결정된다. 객체가 갖는 보안레이블 중 보안등급은 정보에 대한 등급을, 보호범주는 객체 정보가 언급하는 응용분야를 의미한다. 즉, 보호범주는 조직의 시스템 영역 또는 부서를 반영하는데, 회사 조직에서는 기획부, 영업부, 개발부 등의 영역을 뜻한다. 주체에서의 보안등급은 그 주체에 할당될 수 있는 신뢰의 정도를 나타낸다.

PCB
Process Lock
Create Time
Exit Time
Rundown Protect
Unique Process ID
Active Process Links
• • •
Priority Class
Vad Root
보안등급 필드(clearance)
카테고리 정보 필드(category)

그림 2. 프로세스 구조체의 구성
Fig. 2. Construction of process structures

그림 2는 본 논문에서 사용하는 다중 등급 보안 방식에 기초한 강제적 프로세스 메모리 접근 제어 방법에서 구현한 프로세스 구조체의 구성을 나타낸 도면이다. 그림 2를 보면, 프로세스 구조체의 구성은 기존의 윈도우즈 OS에서 정의된 프로세스 구조체에 다중 등급 보안 정보의 수록을 위한 2개의 필드(clearance 및 category)를 추가로 정의하여 사용한다.

그림 3은 다중 등급 보안 방식에 기초한 강제적 프로세스 메모리 접근 제어 방법의 전체적인 흐름을 나타낸 순서도이다.

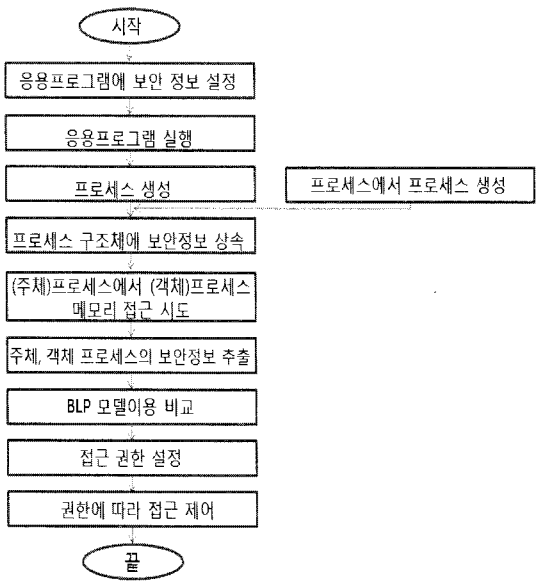

그림 3. 강제적 프로세스 메모리 접근 제어 방법의 순서도
Fig. 3. Flowchart of mandatory memory of process access control

그림 3에서 보면, 시스템 보안 관리자로 하여금 시스템 내의 응용프로그램에 대하여 보안 정보를 배정하도록 하고, 각 응용프로그램에 배정된 보안 정보를 부여하는 절차가 진행된다. 보안 정보를 배정할 때에 보안 등급 및 카테고리 정보로 구분하며 확장 파일 속성의 할당된 필드에 배정된 보안 정보를 각각 수록하여 보안 정보를 설정하는 처리를 진행한다.

이렇게 보안 정보가 설정되어 설치된 응용프로그램은 사용자 또는 실행 중인 다른 프로세스에 의해 실행된다. 응용프로그램의 실행은 응용프로그램의 실행 파일을 실행함으로써 이루어진다. 또한, 사용자(또는 관리자)에 의해 실행될 수도 있고, 실행 중인 다른 프로세스에 의해 실행될 수도 있으며, 네트워크를 통해 외부로부터 실행 명령을 전달받아 실행될 수도 있다. 응용프로그램이 실행되면, 프로세스가 생성된다. 응용프로그램의 실행에 의해 생성되는 프로세스는 하나 또는 그 이상이 될 수도 있다. 응용프로그램의 실행에 의해 프로세스가 생성되기도 하지만, 생성되어 실행중인 프로세스에 의해서도 또 다른 프로세스가 생성될 수도 있다. 이와 같이, 시스템 내에 프로세스가 생성되면, 프로세스 메모리에 프로그램이 적재되며, 이와 함께 커널 메모리에는 생성된 프로세스에 대한 정보들이 프로세스 구조체에 수록된다. 이 커널 메모리는 커널모드에서만 액세스할 수 있는 메모리 영역이고 시스템 내의 프로세스들에 대한 관리와 구분을 할 수 있도록 한다. 이처럼 프로세스 생성과 함께 프로세스 구조체에 정보를 수록할 때에 상기 응용프로그램에서 설정된 보안정보가 상속되어 수록된다.

다음으로, 실행중인 프로세스에서 프로세스 메모리에 대한 접근을 시도하는 이벤트가 발생한다. 이때에는 다른 프로세스가 소유하고 있는 프로세스 메모리에 대한 접근 시도인 경우와, 자신이 소유하고 있는 프로세스 메모리에 대한 접근 시도인 경우를 모두 포함한다. 여기서, 프로세스 메모리의 접근을 시도하는 프로세스를 주체로 정의하고, 상기 주체에 의해 접근 시도를 받은 프로세스 메모리를 소유한 프로세스를 객체로 정의한다. 이때 역시, 주체와 객체가 동일할 수도 있고 다를 수도 있다. 접근 시도가 발생하면 커널 모드에서는 주체와 객체 프로세스의 보안 정보를 추출하는 절차를 진행한다.

이때 프로세스 구조체에 할당된 2개의 필드에서 각각 보안 등급과 카테고리 정보를 조회하여 추출하게 된다. 이렇게 주체와 객체의 보안 정보가 각각 추출되면, 수정된 BLP 모델을 이용하여 각각의 보안 정보를 비교하는 절차를 진행한다. 이 절차는 BLP 모델에 기초하여 주체와 객체에 대하여 각각의 보안 등급과 카테고리 정보를 비교하게 된다. 구체적으로 주체의 보안 등급이 객체의 보안 등급과 같은가 또는 높은가를 비교하고, 주체의 카테고리가 객체의 카테고리와 같은가 또는 포함하는가를 비교하는 등의 절차가 진행된다.

BLP 모델에 기초한 보안정보의 비교 절차를 거친 후

에, 주체의 객체에 대한 접근 권한을 결정하게 된다. 즉, 주체가 객체에 대해 읽기(read, execute), 쓰기(write, create, append, delete)를 할 수 있는지의 여부를 확인하고 이를 권한으로 결정한다. 이때의, 읽기, 쓰기의 대상은 객체 프로세스가 소유한 프로세스 메모리이다.

운영체제 또는 커널모드에서는 접근 권한이 결정되면 그에 따라 주체의 프로세스 메모리에 대한 접근을 제어한다. 즉, 주체가 객체에 대한 접근(읽기, 쓰기) 권한이 부여된 경우라면, 주체가 객체의 프로세스 메모리에 대한 접근 시도는 허가되어 주체는 객체의 프로세스 메모리에 접근(읽기, 쓰기)하고 작업을 수행한다. 반면, 주체가 객체에 대한 접근 권한이 부여되지 않은 경우라면, 주체의 프로세스 메모리에 대한 접근 시도는 거절되어 주체의 객체에 대한 프로세스 메모리 접근은 실패하게 된다.

Ⅳ. 프로세스 메모리 보호 인터페이스

응용 프로그램을 위해 윈도우 운영체제가 제공하는 함수는 API(Application Program Interface)로 제공된다[8]. API는 응용 프로그램이 운영체제와 같은 시스템 프로그램과 통신할 때 사용되는 언어나 메시지 형식을 가지며, 프로그램 내에서 실행을 위해 특정 서브루틴에 연결을 제공하는 함수를 호출하는 것으로 구현된다. 그림 4는 윈도우 운영체제에서, 응용 프로그램에서의 API 호출에 의해 발생한 요청이 시스템 내부에서 처리되는 과정을 보이고 있다. 다양한 API들은 동적 연결 라이브러리 형태로 제공되며 대부분 Kernel32.dll, User32.dll, Gdi32.dll에 존재한다. 응용 프로그램은 실행 시 그림 5와 같이 자신의 프로세스 주소 공간으로 이들 동적 연결 라이브러리를 매핑한 후 사용한다.

프로세스의 메모리를 보호하기 위해서는 후킹이 필요하다. 후킹은 코드를 실행하는 특정 섹션을 가로채는 기술을 의미한다[9,10]. 특정 목적으로 파일의 생성이나 혹은 프로세스의 메모리 접근 등과 같은 이벤트를 추적하고자 할 때 해당 이벤트를 그림 6과 같은 영역에서 후킹할 수 있다. 후킹은 운영체제의 동작을 수정하기 위한 유용한 방법 또한 제공한다. 주어진 인자 또는 리턴 값을 수정하여 프로그램의 동작을 제어하거나 이벤트 호출을 로그에 남길 수 있다. 이런 식으로 API 후킹을 가로챈 뒤에 어떠한 작업을 해주는 함수 즉, API 후킹 프로시저를 선언 및 정의 하여 API 후킹을 시도할 수 있다.

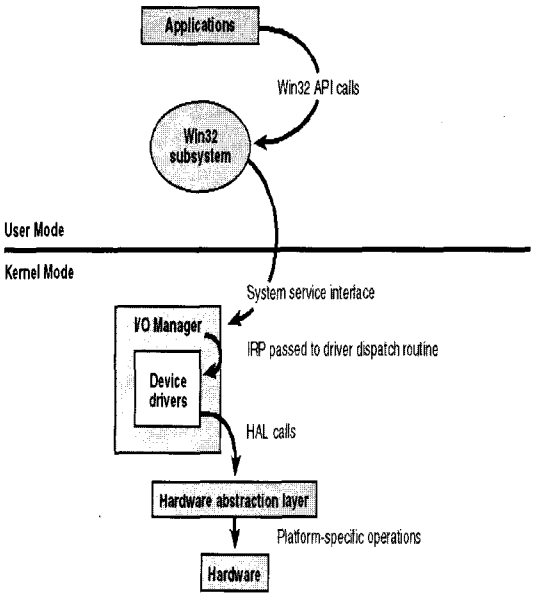

그림 4. API 요청 흐름도
Fig. 4. Flowchart of API request

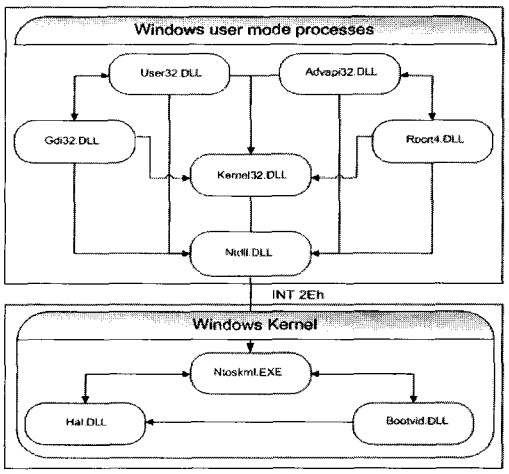

그림 5. DLL 구성도
Fig. 5. Construction of DLL

이 접근의 개념은 원본 API 함수의 주소를 파악하고 이 함수의 첫 번째 몇 바이트를 사용자 정의 API 함수로 연결시키는 JMP 명령으로 변경하는 것이다. 이 기술의 개념은 PE(Portable Executable) 윈도우 파일 포맷의 구조에 의존한다. 그림 7은 PE 포맷의 실행 파일 구조를 개략적으로 보이고 있다.

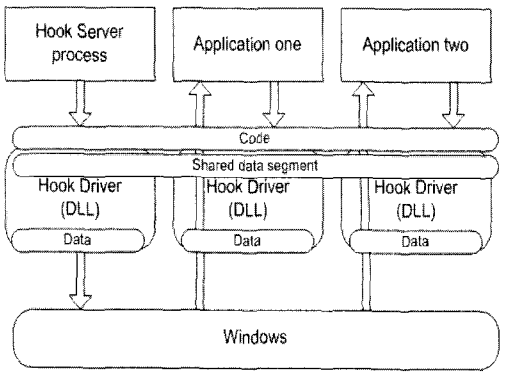

그림 6. 후킹 영역
Fig. 6. Hooking area

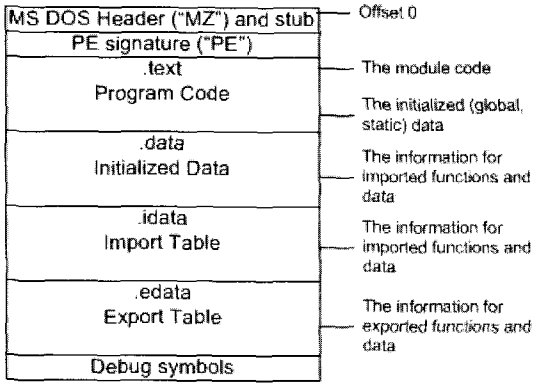

그림 7. PE 포맷
Fig. 7. PE Format

일반적으로 PE 이진 파일이 생성되면 실행 시의 가상 메모리 구조를 따르는 형태의 코드와 데이터 섹션을 갖게 된다. PE 파일 포맷은 몇 가지 논리적인 섹션으로 구성된다.

".idata" 섹션은 IAT(Import Address Table)에 관한 정보를 담고 있다. PE 구조의 이 부분은 IAT 변경을 기반으로 하는 프로그램을 작성하는데 매우 중요하다. 프로그램 로더는 응용 프로그램을 로드하면서 그에 링크된 동적 연결 라이브러리를 함께 메모리에 로드한다. 각각의 동적 연결 라이브러리가 로드되는 주소는 미리 알 수 없기 때문에 로더는 임포트된 각각의 함수들의 실제 주소를 알지 못한다. 로더는 프로그램이 임포트된 함수를 성공적으로 호출할 수 있도록 별도의 작업을 수행하여야만 한다. 임포트된 함수에 대한 각각의 호출이 메모리 상에서 함수 코드가 위치하는 동일한 주소로 전달되어야만 한다. 임포트된 함수에 대한 각각의 호출은 사실상 IAT를 거쳐 간접 JMP 명령을 통하는 간접적인 호출이다. 즉, 단지 IAT 내부의 모든 임포트 주소를 수정하기만 하는 것이다. IAT를 수정하는 후킹 시스템은 임포트된 함수의 주소를 가지고 있는 위치를 찾아 사용자 정의 함수의 주소로 덮어써서 바꿔주는 것이다. 이 과정에서 새로 제공하는 함수는 기존의 함수와 동일한 형태이어야 한다. 즉, 함수의 호출 결과값이 저장되는 변수나 함수의 인자가 동일해야 한다.

즉, IAT 내부의 임포트된 함수의 주소를 변경함으로서 후킹된 함수에 대한 호출은 새로운 함수로 연결되게 된다. 함수 후킹 프로시저를 HookMemoryAccess 함수로 가정하고, 실제 함수를 이용하여 프로세스의 메모리에 대한 변조를 시도하는 프로세스가 해당 프로세스를 접근할 수 있는 권한이 없을 경우에는 실제 함수를 호출하지 않고 연산 결과로 FALSE를 리턴하게 된다. 그 결과 그림 5에서 보이는 Kernel32.dll 호출 이후에 Ntdll.dll로 진입하지 못하고 리턴되어 프로세스의 메모리 변조 시도가 실패한다. 권한 체크는 보안 커널에서 주체와 객체의 보안레이블을 참조하여 비교한 다음 강제적 접근제어에 의해 결정된다.

V. 결론

본 논문에서는 원래의 BLP 모델[7]을 실제 적용하기에 부적절한 면이 있어, 이를 개선한 수정된 BLP 모델을 적용하여 구현하였다. 수정된 BLP 모델을 적용한 강제적 접근제어 시스템에서는 주체의 차별적인 보안등급을 통해 정보를 보호할 수 있다. 즉, 보안등급이 없는 프로세스 또는 보안등급이 맞지 않는 프로세스가 임의적으로 보안등급을 갖는 파일, 디렉터리에 대한 읽기, 쓰기, 실행은 물론 프로세스가 소유하고 있는 메모리에 대

한 접근을 수정된 BLP 보안 모델에 기초하여 커널 모드에서 원천적으로 차단할 수 있다. 시스템 관리자라 하더라도 다른 보안 정보 파일을 임의적으로 접근하는 행위를 차단할 수 있다. 즉, 부정한 목적의 악성 프로그램 등을 이용하여 Administrator의 권한을 획득하더라도 보안 등급을 획득할 수 없기 때문에 프로세스의 메모리를 무단으로 접근 및 변조하는 것을 방지할 수 있다.

본 논문에서 구현한 시스템은 커널을 기반으로 하고 있기 때문에 원천적으로 시스템 보안을 가능하게 하며, 이를 이용하여 상위 수준(level) 즉, 응용 프로그램에서 보안 기능을 추가로 이용하기 쉽다는 장점을 제공한다. 결론적으로 본 논문에서 구현한 보안 커널에 의해 실질적으로 시스템의 보안이 강화되었음을 확인하였으나, API 후킹에 의하여 부가적으로 수행하는 보안 기능에 의하여 시스템의 성능 저하는 불가피하게 발생될 수밖에 없다. 따라서 향후 마이크로 벤치마킹에 의하여 API 호출 시 발생하는 각각의 오버헤드를 측정, 분석하여, 이를 최소화하는 연구와 노력이 필요할 것이다.

참고문헌

[1] 박태규, 임연호, "커널 기반의 보안 리눅스 운영체제 구현," 제11권, 제4호, pp. 33-43, 정보보호학회논문지, 2001.

[2] DoD, "Trusted Computer System Evaluation Criteria," DoD 5200.28.STD, 1985.

[3] Dabak, Phadke, and Borate, Undocumented Windows NT, M&T Books, 1999.

[4] R. Nagar, Windows NT File System Internals O'Reilly, 1997.

[5] P. Orwick and G. Smith, Developing Drivers with the Windows Driver Foundation, Microsoft Press, 2007.

[6] C. Cant, Writing Windows WDM Device Drivers, CMP; Pap/Cdr Edition, 1999.

[7] Bell and Lapadula, "Secure Computer System : Mathematical Foundations and Model," MITRE Report MTR 2547, 1973.

[8] 정창성, 강민규, "안전한 컴퓨팅을 위한 보안 인터페이스 구축," 정보처리학회, 2006.

[9] 이병오, 드라이버 개발자를 위한 윈도우 파일 시스템, 사이버출판사, 2006.

[10] http://www.osronline.com

저자소개

심종익(Jong-Ik Shim)

1985년: 인하대학교 전자계산학사
1987년: 인하대학교 전자계산학석사
1998년: 인하대학교 전자계산공학박사

1986년 ~ 1993년: LG연구소 선임연구원
1994년 ~ 현재: 한서대학교 컴퓨터정보공학과 교수
※관심분야: 데이터베이스보안, 실시간 시스템, 데이터 베이스 등

박태규(Tae-Kyou Park)

1980년: 경북대학교 전자공학사
1989년: 충남대학교 전산학석사
1996년: 성균관대학교 정보공학박사

1992년 ~ 현재: 한서대학교 컴퓨터정보공학과 교수
※관심분야: 정보보호, 실시간 운영체제 등

김진태(Jin-Tae Kim)

1987년: 중앙대학교 전자공학사
1989년: 중앙대학교 전자공학석사
1993년: 중앙대학교 전자공학박사
1995년 ~ 현재: 한서대학교 컴퓨터정보공학과 교수

※관심분야: 영상처리, 디지털 포렌직 등