윈도우 기반 응용프로그램 제공 서비스를 위한 Win32 API 메시지 인가 시스템의 개발

김 영 호[†] · 정 민 아[††] · 원 용 관[†††]

요 약

컴퓨터 자원의 대용량화 및 네트워크 속도의 증가로 인하여 사용자가 네트워크을 통해 원격지의 서버에 접속하여 컴퓨터를 사용하는 요구가 증가되었다. 이에 따라 중앙집중형 컴퓨팅을 통한 응용프로그램 제공 서비스도 활성화 되었다. 중앙집중형 컴퓨팅 시스템은 중앙의 대용량 컴퓨터 시스템에 설치된 응용프로그램을 공유 프로토콜을 통하여 원격 사용자에게 제공하는 응용프로그램 공유 서비스(ASP : Application Service Provision) 시스템 모델이다. 중앙집중형 컴퓨팅 시스템을 통한 응용프로그램 공유 서비스는 기밀성, 가용성, 무결성등의 보안 사항이 반드시 유지되어야 한다. 기존 원격 컴퓨팅인 Telnet, FTP 접속은 단순히 파일 및 데이터의 접근 권한을 제어함으로 보안이 유지된다. 그러나 윈도우 기반 시스템의 경우 다수의 사용자가 동일한 권한을 통해 동일한 응용프로그램을 제공 받기 때문에 사용자들 사이에 기밀성을 및 무결성을 저해 할 수 있다. 또한 다수의 사용자가 하나의 응용프로그램에 파일열기, 복사, 서식 수정 등의 여러 기능 명령어를 전송하기 때문에 파일 및 데이터 접근 제어만을 통해서는 시스템의 기밀성 유지할 수 없다. 또한 기밀성의 문제는 곧 가용성 및 무결성의 문제로 이어질 수 있다. 본 논문에서는 윈도우 기반 중앙집중형 컴퓨팅 시스템의 응용프로그램 공유 서비스를 지원함에 있어 사용자가 실행하는 Win32 API 메시지 명령어 접근제어 시스템을 제안한다. 제안하는 시스템은 GUI(Graphical User Interface) 기반의 서버에서 사용자가 서버에 접속하여 발생하는 모든 메시지(마우스, 키보드, I/O, etc…)들을 감시한다. 감시된 메시지 기반의 명령어는 미리 설정된 사용자별 보안 정책에 기반하여 해당 응용프로그램에게 전달 여부가 결정된다. 이러한 메시지 기반 상세 보안을 통해 기밀성 침해의 우려가 있는 메시지 명령어를 차단하고, 기능 명령어 차단에 의한 자원의 기밀성을 해결하였다.

Development of Win32 API Message Authorization System for Windows based Application Provision Service

Youngho Kim[†] · Mina Jung[††] · Yonggwan Won[†††]

ABSTRACT

The growth of computer resource and network speed has increased requests for the use of remotely located computer systems by connecting through computer networks. This phenomenon has boosted research activities for application service provision that uses server-based remote computing paradigm. The server-based remote computing paradigm has been developed as the ASP (Application Service Provision) model, which provides remote users through application sharing protocol to application programs. Security requirement such as confidentiality, availability, integrity should be satisfied to provide ASP service using centralized computing system. Existing Telnet or FTP service for a remote computing systems have satisfied security requirement by a simple access control to files and/or data. But windows-based centralized computing system is vulnerable to confidentiality, availability, integrity where many users use the same application program installed in the same computer. In other words, the computing system needs detailed security level for each user different from others, such that only authorized user or group of users can run some specific functional commands for the program. In this paper, we propose Windows based centralized computing system that sets security policies for each user for the use of instructions of the application programs, and performs access control to the instructions based on the security policies. The system monitors all user messages which are executed through graphical user interface by the users connecting to the system. All instructions, i.e. messages, for the application program are now passed to authorization process that decides if an instruction is delivered to the application program based on the pre-defined security polices. This system can be used as security clearance for each user for the shared computing resource as well as shared application programs.

키워드 : 접근제어 시스템(ACL : Access Control System), 응용프로그램 제공 서비스(ASP : Application Service Provider), 중앙집중형 시스템(Centralized Computing System)

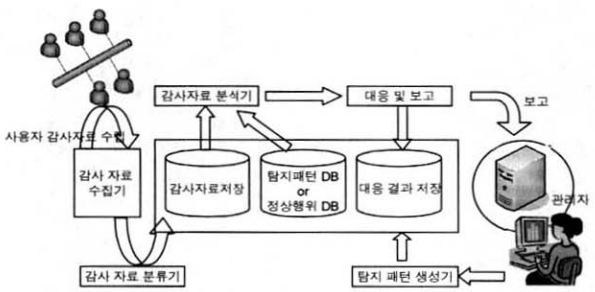
1. 서 론

윈도우 기반의 원격 응용프로그램 제공 서비스를 위한 솔루션은 통신 기술이 발달하고 컴퓨터의 계산 능력이 향상됨에 따라 널리 사용되고 있다. 응용프로그램 공유 서비스는 원거리 응용프로그램을 제공하는 서비스이다. 응용프로그램 공유 서비스는 제공 방식에 따라 서버 기반 컴퓨팅 방식과 클라이언트 기반 컴퓨팅 방식으로 나누어진다. 이 두 가지 제공 방식 중에서 서버 기반 컴퓨팅 방식이 관리 비용의

† 준 회 원 : 전남대학교 대학원 컴퓨터공학과 박사과정
†† 정 회 원 : 전남대학교 전자통신기술연구소 Post-Doc.
††† 종신회원 : 전남대학교 컴퓨터공학과 교수
논문접수 : 2003년 8월 19일, 심사완료 : 2004년 1월 10일

절감과 서버의 응용프로그램을 유지 관리하는데 많은 장점들을 가지고 있기 때문에 응용프로그램 공유 서비스 제공자들은 서버기반 컴퓨팅 방식을 많이 사용한다[1-4].

서버기반 컴퓨팅 방식은 서버가 응용프로그램 및 모든 계산과정을 처리하기 때문에 중앙집중형 시스템(Centralized Computing System)이라고 한다. 응용프로그램 공유 서비스를 위한 서버기반 컴퓨팅 시스템에서 클라이언트는 암호화된 네트워크를 통해 원격 서버의 응용프로그램을 실행할 수 있다. 사용자가 클라이언트를 통해 접속 하면, 서버는 사용자의 로그인 정보와 응용프로그램 실행정보를 감사 자료로 저장한다.



(그림 1) 침입탐지 시스템의 구조

감사 자료를 이용해 사용자와 서버 사이에 응용프로그램 실행 명령어들을 관리자에게 보고하고, 침입패턴에 대응하는 (그림 1)과 같은 침입탐지 시스템(IDS : Intrusion Detection System)들이 개발되었다. 그러나 기존의 침입탐지 시스템은 침입이라고 판단되는 데이터를 수집하여 관리자에게 알리는 기능에 치중해 왔으며, 이에 대한 대처는 아직까지 관리자의 몫으로 남아 있다.

더욱이 마이크로소프트사의 윈도우 기반 응용프로그램 공유 서비스의 경우 다수의 사용자가 실행하는 하나의 응용프로그램은 다양한 Win32 API 메시지를 가지고 있다. 다양한 Win32 API 메시지의 여과 없는 처리는 다양한 침입패턴을 유발하며, 비정상행위의 경우의 수가 증가한다. 예를 들어, 윈도우 시스템의 손님그룹(Guest Group)에는 웹을 사용할 수 있는 인터넷 사용자(사용자명 : IUSR_Computer 이름)가 존재한다. 이때 웹에서 특정 실행파일을 사용하면 인터넷 사용자가 관리자 권한을 획득하였다. 이는 실행파일에 관리자 권한을 사용하는 메시지가 존재하였기 때문이다.

즉, 시스템 관리자가 사용자의 파일과 디렉터리에 접근권한을 설정하여도 Win32 API 메시지를 제어 하지 않는다면, Win32 API 메시지를 생성 또는 변조를 이용해 허용되지 않은 명령을 실행할 수 있다. 따라서 Win32 API 메시지를 통해 데이터 접근 및 수정, 응용프로그램을 실행 등의 권한을 획득 할 수 있다. 즉, 침입자는 Win32 API 메시지를 통해 서비스 거부공격(DOS : Denial-of-Service), 백도어(Backdoor), 트로이안(Trojans), 루트킷(RootKits)등의 서버를 저해하는 공격을 할 수 있다.

본 논문은 클라이언트에서 전송하는 Win32 API 메시지 실행 정보를 메시지 가로채기를 이용해 실시간으로 감시한다. 또한 사용자별 상이한 보안 정책을 설정함으로써 사용자에게 주어진 권한만을 실행하고, 침입자에 의해 생성되거나 수정되어진 Win32 API 메시지를 실행할 수 없는 명령 인가 시스템을 제안한다. 제안된 명령 인가 시스템은 응용프로그램 공유서비스에서 제공되는 응용프로그램에 대한 Win32 API 메시지를 각 사용자별 접근제어 목록에 기반을 두어 제어함으로써 사용자에게 허가된 권한만을 실행할 수 있도록 한다.

본 논문은 2장에서 응용프로그램 공유 서비스에 대해 설명하고, 3장에서 기존 Win32 API 메시지에 대한 처리 과정 및 문제점과 메시지 가로채기를 이용한 Win32 API 메시지 인가 시스템에 대해 설명한다. 4장에서는 사용자 접근권한 설정을 위한 Win32 API 메시지 접근권한 목록 및 Win32 API 메시지 인가 시스템의 구조를 설명한다. 5장에서는 Win32 API 메시지 인가 시스템과 운영체제와의 상호 관련성을 분석하기 위한 성능 평가 결과를 기술한다. 마지막으로 6장은 제안한 시스템을 이용한 결론을 내린다.

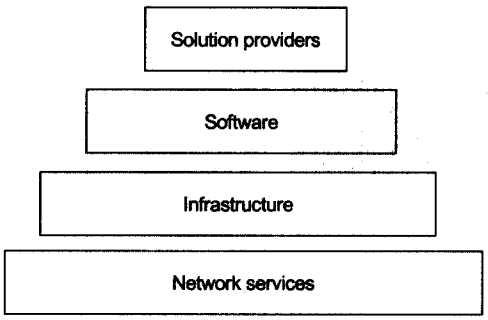
2. 응용프로그램 공유 서비스(ASP : Application Service Provision)

본 논문에서 언급하는 응용프로그램 공유 서비스(Application Service Provision)는 그래픽 사용자 인터페이스(GUI : Graphical User Interface)를 이용해 윈도우에서 사용하는 응용프로그램을 클라이언트에 완벽히 재구성 해주기 때문에 기존의 COM과 CORBA와 같은 데이터 중심의 데이터 공유 서비스와는 다르다. 이 장에서는 응용프로그램 공유 서비스에 대해 설명하고, 관리자 측면과 사용자 측면에서의 장점을 통해 응용프로그램 공유 서비스의 사용 목적 및 구성에 대해 알아본다.

2.1 응용프로그램 공유 서비스의 계층구조

응용프로그램 공유 서비스는 크게 네트워크 서비스(Network Service), 기반구조(Infrastructure), 소프트웨어(Software), 솔루션 제공(Solution Providers)의 네 계층으로 나누어지며 (그림 2)와 같다. 네트워크 서비스 계층은 최하위 계층으로 네트워크의 트래픽(Traffic)을 관리하고, 통신 보안 및 통신 방식을 정의한다. 정의된 통신방식을 이용해 응용프로그램 공유 서비스를 제공한다. 기반 구조 계층은 새로운 사용자에게 서버의 하드웨어 컴퓨팅 자원을 할당하고, 할당된 자원을 네트워크 서비스 계층에 전달한다. 소프트웨

어 계층은 응용프로그램 공유 서비스를 위해 여러 응용프로그램을 사용자에게 서비스하고, 사용자 관리 및 제공 서비스를 관리하는 단계이다. 솔루션 제공 계층은 응용프로그램을 통하여 제공되는 데이터와 정보들을 말한다.



(그림 2) ASP 채널의 단계

2.2 서버기반 응용프로그램 공유 서비스 이점

서버기반 응용프로그램 공유 서비스는 서비스 제공자 측면에서 다음과 같은 장점을 가지고 있다. 첫째, 사용자의 수에 따라 소프트웨어를 구입할 필요가 없기 때문에 소프트웨어 구입에 소요되는 분산 비용의 절감을 가져올 수 있다. 둘째, 응용프로그램이 서버에 설치되기 때문에 관리자가 사용자에게 직접 방문해 설치해야하는 시간과 비용을 절감할 수 있다. 셋째, 사용자의 수에 따라 서버의 응용프로그램 라이센스를 받기 때문에 불법복제의 대한 위험을 감소할 수 있다. 넷째, 서버관리자에 의한 빠른 업그레이드는 항상 최신의 버전을 공유할 수 있는 장점이 있다. 다섯째, 일관된 응용프로그램은 서로 다른 버전에 의해 발생되는 복잡성을 감소할 수 있다.여섯째, 유지 관리를 하면서 발생되는 관리상의 부대비용을 감소 할 수 있다.

또한 서버기반의 중앙집중형 시스템은 사용자에게 다음과 같은 장점을 가지고 있다. 첫째, 구형 컴퓨터라도 네트워크를 통해 응용프로그램이 제공되기 때문에 대량의 컴퓨팅 자원 없이도 원격의 슈퍼 컴퓨팅 환경을 사용할 수 있다. 둘째, 사용자가 응용프로그램을 직접 설치할 필요가 없기 때문에 설치 부담이 감소한다. 셋째, 서버 관리자가 이미 설치한 응용프로그램을 사용하기 때문에 응용프로그램의 버전에 대한 호환성 및 설치 공간에 대하여 고려할 필요가 없다. 넷째, 네트워크만 연결이 가능하다면 시스템 부팅 및 시스템 다운에 대한 시간을 절약할 수 있다[5].

2.3 응용프로그램 공유 서비스를 위한 중앙집중형 시스템의 구성

응용프로그램 공유 서비스를 위한 중앙집중형 시스템은 Thin-Client와 다중 인터랙티브 사용자로 구성된다. Thin-Client란 대용량의 자원을 필요하지 않는 클라이언트를 말하며, 다중 인터랙티브 사용자란 서버에 다수의 사용자가 사용하지만, 서로 다른 사용자의 작업 환경을 침해하지 않고, 독립적인 작업 환경을 제공하는 것을 말한다.

2.3.1 Thin Client

Thin Client는 네트워크를 통해 그래픽 및 텍스트를 포함한 모든 서버의 자원을 클라이언트의 원격 데스크탑 화면에 재구성하여 출력하고, 클라이언트의 키보드, 마우스 등의 주변기기로부터 입출력 정보를 입력받아 서버에 전송한다. Thin Client는 과거의 더미 터미널(Dummy Terminal)과 같은 원격 컴퓨팅 능력과 저장 장치를 갖게 되고, Thin-Client는 그래픽 사용자 인터페이스를 완벽하게 클라이언트에 구성해 준다. Thin Client는 그래픽 사용자 인터페이스의 구성을 위해 T128 프로토콜이라는 응용프로그램 공유 프로토콜을 통하여 한정된 네트워크 대역폭에서 클라이언트와 서버 사이에 교환되는 정보를 최소화하고 효율을 최대화 한다[6].

예를 들어, Win32 기반의 Thin-Client는 메모리 용량 130 KB, 작업 설정을 위해 300KB, 디스플레이를 위해 100KB 정도만을 사용하고, T128 프로토콜을 사용해 28Kbps의 모뎀에서도 동작이 가능하다. 따라서 사용자는 Thin Client를 통해 원거리에서 네트워크를 통하여 슈퍼컴퓨터나 분산 컴퓨팅 환경을 사용하는 똑같은 효과를 얻을 수 있다.

2.3.2 다중 인터랙티브 사용자

다중 인터랙티브 사용자란 여러 명의 사용자가 홈 디렉터리, 프로파일, 데스크탑, 윈도우 스테이션 등의 서버 자원을 사용자의 공유권한에 맞게 작업할 수 있음을 의미한다. 공유권한이란 사용자에 따라 사용가능한 응용프로그램, 디스크 자원, 이외 하드웨어 자원 등을 말한다. 또한, 서로 다른 사용자가 같은 시스템을 사용하더라도, 사용자는 서버 시스템을 자신만이 사용하는 것과 같이 느끼게 되며, 사용자 별로 독립된 작업공간을 가질 수 있다. 즉 사용자 측면에서 볼 때 인터랙티브 사용자는 한명이지만, 시스템 측면에서 볼 때 여러 명의 인터랙티브 사용자를 고려해 시스템은 동작한다.

기존 대부분의 원격 컴퓨팅 환경은 만약 한명의 사용자가 자원을 실행 중일 경우, 다른 사용자는 이미 다른 사용자가 접근하고 있는 자원에는 접근을 할 수 없다. 그러나 다중 인터랙티브 사용자 지원은 사용자 컴퓨터의 물리적인 메모리를 서버의 가상메모리와 매핑 시키고, 각 사용자에게 하나의 독립된 세션을 생성하여 이미 사용하고 있는 자원을 다른 사용자가 사용할 수 있게 된다[7].

그러나 사용자의 공유 권한 및 독립된 작업공간은 Unix, Linux와 비교하자면 하나의 터미널과 같은 기능에 불과 하다. 다른 시스템에서도 마찬가지 이지만 응용프로그램 공유

서비스를 위한 시스템도 사용자가 악의적인 행동을 한다면, 응용프로그램 공유 서비스를 하는 시스템은 정상적인 서비스를 보장해 줄 수 없다. 즉 시스템 관리자 권한을 갖는 Win32 API 메시지를 생성하여 서버에 전송할 경우 기존 시스템은 이를 속수무책으로 수용할 수밖에 없다. 이에 대한 자세한 기술은 3장의 서론에서 기술한다.

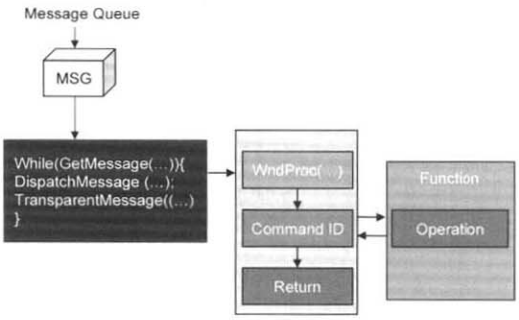
3. 응용프로그램 공유 서비스의 보안상 문제점

응용프로그램 공유 서비스는 응용프로그램의 모든 정보를 네트워크를 통해 전송하기 때문에 통신 보안기술을 이용하여 외부침입자의 공격을 차단하였다. 그러나 외부 침입자가 아닌 내부의 침입자가 생겼을 경우 이러한 통신 보안기술로는 시스템의 보안을 유지 할 수 없다. RunAS와 같은 해킹 프로그램을 예를 들어 보자. RunAS는 사전 로그인된 사용자를 통해 타인의 계정 심지어 관리자 계정까지 획득할 수 있다. 이 방법은 로그인된 사용자 계정에서 Win32 API 함수 LogonUser를 호출해 관리자로 로그인 한다. LogonUser함수는 사용자 로그인 Win32 API 메시지를 호출하고 사용자 정보를 변경시킨다. Win32 API 메시지를 통해 사용자 정보를 변경한 사용자는 실행권한이 없는 경우에도 변경된 사용자 정보를 이용해 Win32 API 메시지 생성하여 응용프로그램을 실행 시킬 수 있는 위험한 툴이다. 즉 접속한 사용자가 관리자(Administrator User)가 아니라도 RunAS를 통해서는 관리자가 될 수 있는 것이다. 따라서 내부 침입자는 Win32 API 메시지의 생성 또는 변경을 이용해 응용프로그램 공유 서비스를 제공하는 서버 시스템의 보안을 저해한다. 이와 같이 윈도우 시스템 운영을 위해 설계된 Win32 API 메시지가 아무런 제약 없이 처리될 경우 보안상 문제점을 발생시킨다[8].

본 장에서는 Win32 API 메시지의 처리과정을 살펴보고, Win32 API 메시지에 대한 인가절차의 필요성과 인가 절차 과정을 기술한다.

3.1 Win32 API 메시지의 처리

(그림 3)은 윈도우 시스템의 Win32 API 메시지 명령어 처리 과정을 나타낸 것이다. Thin Client의 Win32 API 메시지가 발생되면 통신망을 통해 응용프로그램 서비스를 제공하는 서버에 전달된다. 전달된 메시지 명령어는 운영체제가 관리하는 메시지 큐(Message Queue)에 모두 저장된다. 서버의 응용프로그램들은 GetMessage() 함수에 의해 응용프로그램의 해당하는 메시지를 호출하고, DispatchMessage()와 TransparentMessage() 함수는 응용프로그램 실행을 위한 윈도우 처리 함수인 프로시저(WndProc)를 호출한다[9]. 호출된 프로시저는 윈도우 메시지 식별자(Windows Message Command ID)에 맞는 명령어 처리를 위한 여러 함수를 호출함으로써 응용프로그램이 수행된다[9, 10].



(그림 3) Win32 Message 명령 처리과정

3.2 메시지 명령어 보안의 필요성

윈도우 기반 응용프로그램 공유 서비스는 기존의 다른 운영체제와 같이 사용자 데이터 접근제어(Data Access Control)로 데이터 보안 설정을 하고 있다. 그러나 응용프로그램 공유 서비스를 제공하는 윈도우 시스템은 시스템을 사용하는 누구나 응용프로그램 내부에 Win32 API 메시지 명령어를 송신 할 수 있는 보안상 문제점이 존재한다. 즉, 응용프로그램 공유서비스를 사용하는 사용자는 누구나 메시지 큐에 Win32 API 메시지를 저장하거나 삭제할 수 있다. 이는 다수 사용자에게 응용프로그램 서비스를 제공하는 서버의 과부하 및 보안상의 문제점을 야기할 수 있다. 이는 내부 침입자가 윈도우 시스템에 로그인 되면 Win32 API 메시지의 생성 및 변경을 통해 관리자 몰래 시스템을 제어할 수 있음을 의미한다.

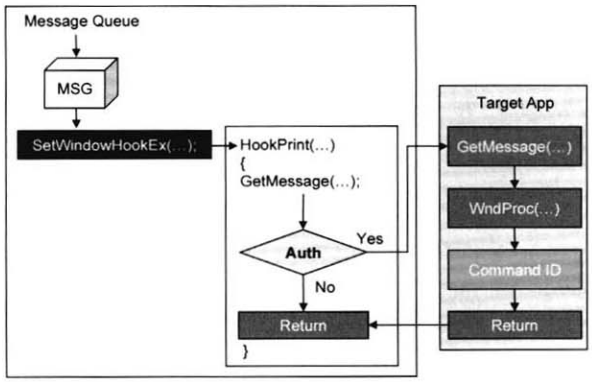
따라서 기존 데이터 접근제어권한 뿐만 아니라 Win32 API 메시지에 대한 접근제어가 필요하다. 이를 위해 본 논문에서는 Win32 API 메시지 가로채기를 이용한 메시지 인가 시스템을 제안한다.

3.3 Win32 API 메시지 가로채기

메시지 가로채기(Message Hook) 과정이란 (그림 3)에서 설명한 Win32 API 메시지 처리과정에서 메시지 큐에 저장된 Win32 API 메시지 명령어를 응용프로그램에 전달하기 전에 메시지 큐에서 사전에 정해진 임시의 공유메모리로 강제 이동시키는 것을 말한다. 즉 운영체제 커널의 메시지 큐에서 해당 응용프로그램 Win32 API 메시지를 수신하지 못하도록 임시 공유메모리에 저장해 놓는다.

(그림 4)은 메시지 가로채기 과정을 나타낸다. (그림 4)은 (그림 3)와 비교할 때 SetWindowHookEx()를 통해 메시지 가로채기를 초기화하는 부분이 추가 되어 있으며, HookProc를 통해 최종 응용프로그램에 전달하기 전에 메시지를 가로채어 임시 공유 메모리 상에 저장하고 실행을 보류시

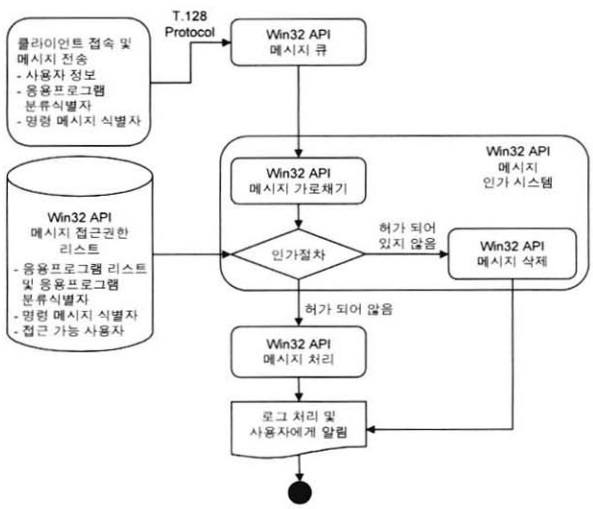
킨다. 이후, Win32 API 메시지 인가 시스템은 사용자에게 Win32 API 메시지의 사용권한이 허가되었을 경우 응용프로그램에 전달하고, 허가되어 있지 않을 경우 Win32 API 메시지를 공유 메모리에서 삭제한다.



(그림 4) 메시지 가로채기 과정

3.4 Win32 API 메시지 인가 시스템의 처리 과정

Win32 API 메시지 인가 시스템은 Thin Client가 전송한 Win32 API 메시지를 Win32 API 메시지 가로채기를 이용해 응용프로그램에서 전달하는 것을 일시 중지 시킨다. (그림 5)와 같이 메시지 큐에서 가로채기된 Win32 API 메시지는 접근권한 리스트 리스트를 근거로 사용자가 발생한 Win32 API 메시지가 허가 권한이 있는지 검사한다. 이때, Win32 API 메시지가 사용자에게 허가되어 있다면 Win32 API 메시지는 해당 응용 프로그램에 전달하여 보류된 Win32 API 메시지를 수행한다. 만약 사용자에게 Win32 API 메시지가 허용되어 있지 않다면, Win32 API 메시지는 큐에서 삭제되고, 해당 응용 프로그램은 실행 명령을 전달받지 않아 사용자의 Win32 API 메시지를 무시한다.



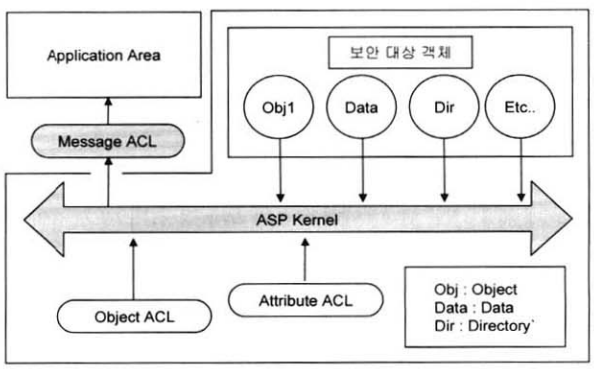
(그림 5) Win32 API 메시지 인가 처리 과정

4. Win32 API 메시지 인가 시스템의 구현

4.1 Win32 API 메시지 접근제어 목록

본 장에서는 Win32 API 메시지 인가 시스템을 위한 Win32 API 접근권한 리스트에 대해 설명한다. 기존의 데이터 접근권한 시스템은 접근제어 목록(ACL : Access Control List)을 통해 사용자에 대해 객체의 접근권한, 실행권한, 데이터 속성 설정 등을 통제해 왔다[11-13]. 3장에서는 데이터 접근제어 권한만을 이용한 보안 시스템은 윈도우 기반 Win32 API 메시지 처리과정에서 보안상의 문제점이 있다고 기술 하였다. 즉 RunAS와 같은 프로세서 자체 또는 RunAS가 전송하는 Win32 API 메시지를 처리하는 것은 응용프로그램 공유 서비스를 하는 서버에 치명적인 손상을 줄 수 있다.

본 논문은 Win32 API 메시지 처리과정의 보안상의 문제점을 해결하기 위해 Win32 API 메시지 인가 시스템을 정의 하였다. (그림 6)은 Win32 API 메시지 명령 인가 시스템이 기존 접근제어 권한 보안 시스템에 추가적으로 응용프로그램 공유 서비스 윈도우 커널(ASP Kernel)과 응용프로그램 사이의 추가적 메시지 접근제어를 하는 의미를 가진다. (그림 6)의 보안객체 대상, Object ACL, Attribute ACL은 기존의 데이터 접근 권한 시스템이 가지는 보안 대상이다. 본 논문은 Win32 API 메시지 명령의 접근제어가 없는 종전의 문제를 차단하기 위해 Message ACL이라는 Win32 API 메시지 접근권한 리스트를 삽입하였다. 메시지 인가 시스템은 Message ACL에 근거하여 프로세서 및 Win32 API 데시지에 대한 사용자의 허용 여부를 판단한다.



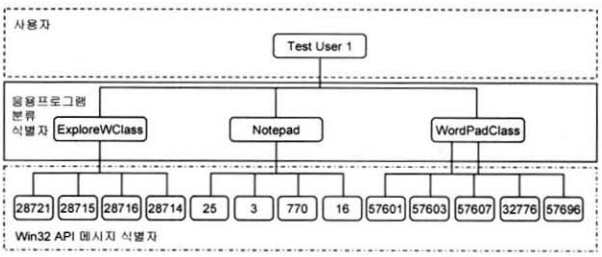
(그림 6) 기존 접근제어의 유형 및 추가적인 Win32 API 메시지 접근권한 리스트(Message Access Control List)

Win32 API 메시지 접근권한 리스트를 만들기 위해 응용프로그램과 응용프로그램 내부의 Win32 API 메시지에 대한 고려를 하였다. 윈도우 기반 응용프로그램은 고유의 응용프로그램 분류 식별자(class identification)를 가지고 있다. 또한 응용프로그램 분류 식별자와 함께 Win32 API 메시지 식별자(Windows 32Bit Message Identification)는 해

당 명령을 실행하는 역할을 한다.

예를 들어, 사용자 'Test User 1'에 대한 워드패드에 대해 응용프로그램 분류 식별자는 'WordPadClass'을 허용하고, 워드 패드의 '파일' 메뉴의 '열기' 명령은 Win32 API 메시지 식별자가 '57601'로 정의 되어 있다. 즉 'Test User 1'은 응용프로그램 분류 식별자 'WordPadClass'에 의해 워드패드의 '파일-열기' 명령을 사용 가능 하다. 이외에 Win32 API 메시지 접근제어 목록에 추가되어 있지 않는 응용프로그램 및 Win32 API 메시지는 모두 사용이 불가능 한 것으로 정의 하였다. 이는 시스템 사용자의 최소권한 법칙에 의해 정하였다.

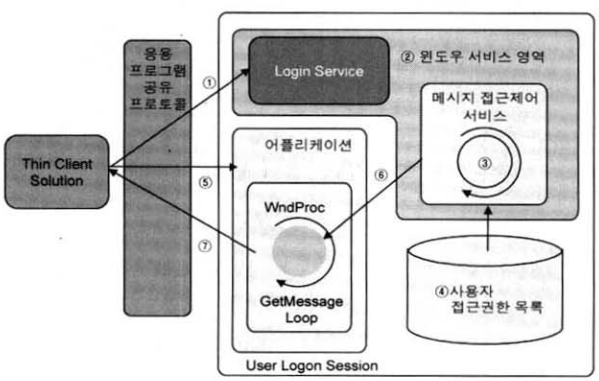
따라서 각 응용프로그램에 대해 응용프로그램 분류 식별자와 메시지 명령 식별자에 따른 사용자별 Win32 API 메시지 접근제어 목록(Message ACL)을 정의한다. Win32 API 메시지 인가 시스템은 정의되어진 접근제어 목록(Message ACL)에 등록되어진 응용프로그램 및 해당 응용프로그램의 메시지만을 실행한다. Win32 API 접근제어 목록은 (그림 7)과 같은 트리 구조를 보여준다.



(그림 7) 응용프로그램에 대한 Message ACL

4.2 시스템 구조

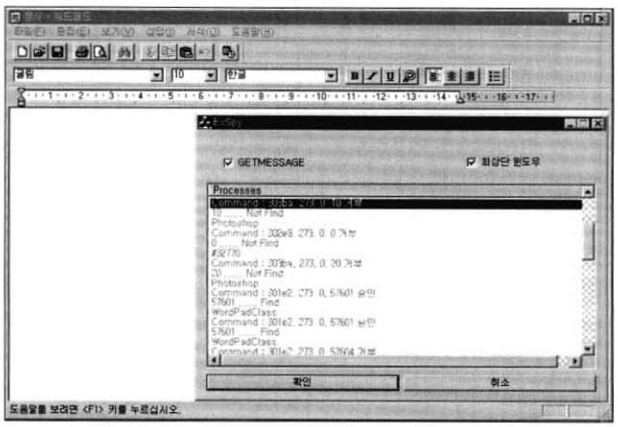
Win32 API 메시지 인가 시스템은 (그림 8)의 순서로 작동을 한다. 먼저 응용프로그램 공유 프로토콜을 통해 사용자가 로그 온 한다(①), 로그인된 사용자는 각 세션 별로 윈도우 서비스를 시작한다(②). 각 사용자별 세션 영역은 자동으로 메시지 접근제어 서비스에서 Win32 API 접근제어 시스템을 시작하고(③), 메시지 접근제어 목록을 로드한다(④). 사용자가 응용프로그램 실행하면(⑤), 응용프로그램의 Win32 API 메시지는 Win32 API 메시지 인가 시스템이 가로채기에 들어간다(⑥). 그리고 해당 응용프로그램과 Win32 API 메시지 접근제어 시스템에 의해 각 사용자에게 Win32 API 메시지 허용 여부를 사용자에게 전달한다(⑦).



(그림 8) 시스템 구성도

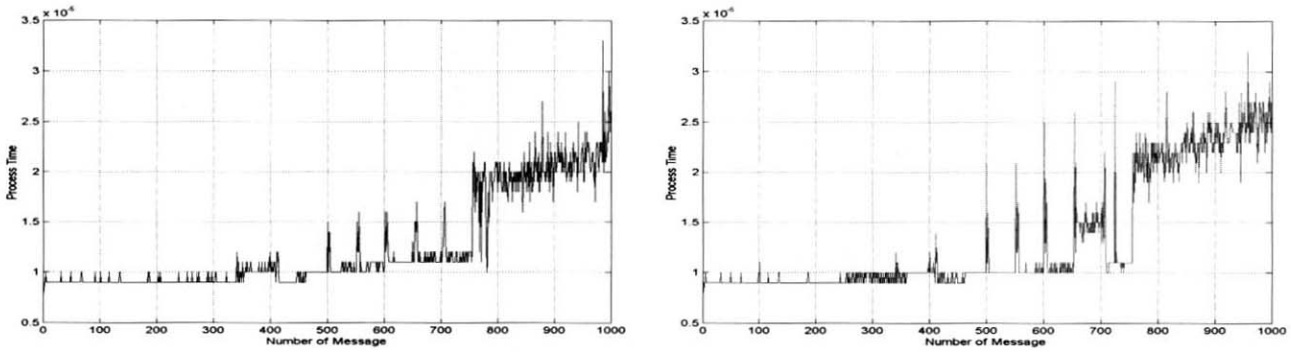
5. 실험 및 결과

이 장에서는 제안한 Win32 API 메시지 인가 시스템이 사용자가 정해진 사용자의 권한에 맞는 Win32 API 메시지만을 수행하는지 여부를 검사를 수행하였다. 이를 위해 응용프로그램에 대한 Win32 API 메시지 접근제어 목록은 워드패드에 대해서 메시지 식별자 '57601' 기능에 대해서만 허가해 놓았으며, Photoshop 프로그램에 대해서는 허가한 메시지가 없다. 만약 사용자가 포토샵과 워드패드 두 프로그램 동시에 사용할 경우, 사용자는 포토샵과 워드패드에 대해 (그림 9)와 같은 출력 창을 볼 수 있을 것이다. 출력창의 Photoshop과 WordPadClass는 응용프로그램 식별자이며, 이때 포토샵에서 실행하는 모든 메시지는 실행을 거부하는 것을 볼 수 있다. 그리고 WordPadClass에 대해서는 '57601' 메시지 식별자에 대해서만 허가되는 경우를 보인다.



(그림 9) 사용자 명령 거부/승인 화면

또한 운영체제의 커널과 응용프로그램 사이에 존재하는 인가처리 과정이 응용프로그램 공유 서비스를 제공하는 서버 시스템에 어떠한 영향을 주는지 파악하기 위해 Win32 API 메시지인가 시스템의 유무에 따른 시스템의 메시지 처리 시간을 측정하였다. 실험 환경은 CPU : Pentium 3 Xeon 700Mhz Dual, 메모리 : 512M Byte에서 실험하였으며, 최적 실험(Bare bone test)을 위해 응용프로그램 공유 서비스 및 Win32 API 메시지인가 시스템 이외의 필수 서비스를 제외하고 대부분의 윈도우 서비스를 중지하였다. 또한 메시지 가로채기의 영향만을 알아보기 위해 응용프로그램은 메시



(그림 10) 성능테스트그래프(메시지 가로채기 무 : 좌, 메시지 가로채기 유 : 우)

지에 대한 어떠한 처리도 수행하지 않도록 구성하였다.

단일 클라이언트에서 하나의 Win32 API 메시지를 1~1000개까지 순차적으로 전송하였으며, 메시지 큐 시스템의 특성상 이전 32비트로 구성된 메시지가 실행되어야 다음 메시지가 실행됨으로 1000명의 사용자가 동시에 1개씩의 메시지를 전송하였다고 가정하여도 무방하다. 여기서 고려할 점은 본 논문의 1000개의 메시지는 메시지가 응용프로그램에 전송되어 어떠한 후 처리도 수행하지 않는 루프를 구성한 것이다. 만약 응용프로그램이 영상처리, 음성처리, 명령의 수행 과정 등과 같은 데이터 처리가 추가될 경우 처리 시간은 더욱 많이 소요될 것이다.

(그림 10)에서 Win32 API 메시지의 수가 500개를 초과하면 메시지 가로채기가 있는 경우가 메시지 가로채기가 없는 경우보다 시간지연 그래프의 오실레이션이 많은 것을 볼 수 있다. 그러나 <표 1>에서 메시지 수에 따른 평균 처리 시간을 보면, Win32 API 메시지의 개수가 500개 미만일 때는 0.2 μ/sec로 시간차가 없다고 볼 수 있다. 또한 메시지 수가 500개를 넘으면 메시지 가로채기의 유/무와 관계없이 동반 상승한다. 이는 Win32 API 메시지 처리 과정에서 메시지 큐에서 메시지를 호출하여 CPU에서 메시지를 처리하는데 있어 운영체제가 한계를 느끼기 때문이다.

〈표 1〉 메시지 인가절차 처리 소요 시간

(단위 : 1×10^{-6} sec)

인가절차 유무 / 메시지 개수	인가처리 가로채기 없는 경우	인가처리 가로채기 있는 경우
1~100	9.050	9.050
101~200	9.165	9.165
201~300	9.190	9.370
301~500	9.974	9.985
501~751	11.288	11.864
751~1000	20.120	22.868

위 실험 결과를 통해 Win32 API 메시지인가 시스템의 방식인 메시지 가로채기 방식은 시스템의 Win32 API 메시지 처리 시간에 큰 영향을 주지 않는 것을 알 수 있다.

6. 결 론

원거리의 중앙집중형 서버에서 제공되는 응용프로그램 공유 서비스(ASP : Application Service Provision)는 사용자에게 응용프로그램을 자유롭게 사용할 수 있는 작업환경을 제공한다. 이때 사용자는 Thin Client를 사용하여 ASP를 제공하는 서버에 접속하지만, 마이크로소프트의 윈도우 기반 응용프로그램 공유 서비스의 Win32 API 메시지의 여과 없는 처리는 침입자로 하여금 Win32 API 메시지의 생성 및 변경 할 수 있는 보안상의 문제점을 일으킨다.

본 논문은 Win32 API 메시지의 보안상의 문제점을 해결하기 위해 Win32 API 메시지의 접근제어 및 감사를 수행하는 Win32 API 메시지 인가 시스템을 제안하였다. 제안한 인가시스템에서는 응용프로그램에 대해 응용프로그램 분류 식별자와 Win32 API 메시지 식별자를 통해 사용자별 메시지 접근제어 목록을 정의하고, 응용프로그램이 Win32 API 메시지를 실행하기 전에 Win32 API 메시지 가로채기 기법을 이용해 사용자의 행동을 실시간으로 감사한다. 만약 응용프로그램의 분류 식별자와 Win32 API 메시지 식별자가 사용자에게 허용되지 않을 경우, 메시지 인가 시스템은 해당 메시지를 삭제하고 응용프로그램은 발생된 메시지를 무시하게 된다. 이러한 처리 과정을 통해 응용프로그램 공유 서비스를 제공하는 시스템에 대하여 Win32 API 메시지에 대해 각 사용자에게 적절한 권한 설정을 하였다.

그러나 제안한 Win32 API 메시지 명령인가 시스템은 다수의 응용프로그램에 많은 사용자가 연관되어 있기 때문에 접근제어 목록에 대한 정의가 복잡하다. 따라서 메시지 접근제어 목록에 효율적인 정의 및 관리가 이루어져야 할 것으로 보인다.

참 고 문 헌

[1] Borko Furht, "An Innovative Internet Architecture for Application Service Providers," IEEE, Proceeding of the 33rd Hawaii International Conference on System Sciences,

Vol.6, p.6051, Jan., 2000.
[2] Todd W. Mathers, "Windows NT/2000 Thin Client Solutions : Implementing Terminal Services and Citrix MetaFrame," NewReaders, Jun., 2000.
[3] Microsoft Corporation, "Windows 2000 Terminal Services Technologies : http://www.microsoft.com/windows2000/technologies/terminal/default.asp," Microsoft Windows Terminal Service.
[4] Citrix MetaFrame White Paper, "Server-Based Computing : http://www.citrix.com," Citrix Systems, While Paper, 1999.
[5] Lixin, "Shifting Paradigms with the Application Service Provider Model," October, IEEE, Vol.34, No.10, pp.32-39, Oct., 2001.
[6] ASP News Review, "Anatomy of an ASP Defining the new genus," ASPnews.com, May, 2000.
[7] Frank Kim, "Run Your Apps on a Variety of Desktop Platforms With Terminal Server," MSJ(Microsoft System Journal), pp.10-19, Jan., 1999.
[8] Ed skoudis, "Counter Hack : A Step by step Guide to Computer Attacks and Effective Defense," Prentice Hall, p.126, Jul., 2001.
[9] Microsoft Platform SDK Documentation : Message Queuing, "Messages and Message Queues : http://msdn.microsoft.com," Microsoft Corporation.
[10] Microsoft Platform SDK Documentation : Event Loging, "Reding Event Log : http://msdn.microsoft.com/library/default.asp?url=/library/en-us/debug/eventlog_7145.asp," Microsoft, Corp.
[11] Ravi S.Sandhu* and Pierangela Samarati, "Access Control : Principles and Practice," IEEE, Computer, Vol.32, No.9, pp.40-48, Sep., 1994.
[12] 최용락, 소우영, 이재광, 이임영 역저, "통신망 정보 보호", 도서출판 그린, p.277, 1996.
[13] Keith Brown, "Programming Windows Security," Addison Wesley, Chapter 6, Jul., 2000.

김 영 호

e-mail : melchi@grace.chonnam.ac.kr
2000년 조선대학교 제어계측공학과(학사)
2002년 전남대학교 컴퓨터공학과(석사)
2002년~현재 전남대학교 컴퓨터공학과 박사과정
관심 분야 : 정보보호, 네트워크, 영상처리, 패턴인식

정 민 아

e-mail : majung@grace.chonnam.ac.kr
1992년 전남대학교 전산통계학과(학사)
1994년 전남대학교 대학원 전산통계학과 (이학석사)
2002년 전남대학교 대학원 전산통계학과 (이학박사)
2002년~2003년 광주과학기술원 정보통신학과 Post-Doc.
2003년~현재 전남대학교 전자통신기술연구소 Post-Doc.
관심분야 : 데이터마이닝, 데이터베이스, 정보보호, 생물정보학

원 용 관

e-mail : ykwon@grace.chonnam.ac.kr
1986년 한양대학교 전자공학과(학사)
1991년 Univ. of Missouri, 컴퓨터공학과 (공학석사)
1995년 Univ. of Missouri, 컴퓨터공학과 (공학박사)
1991년~1995년 Univ. of Missouri, Research/Teaching Assistant
1995년~1996년 전자통신연구원
1996년~1999년 한국통신 연구개발본부
1999년~현재 전남대학교 컴퓨터공학과 부교수
관심분야 : 생물정보학, 네트워크 관리, 데이터마이닝, 정보보호, 컴퓨터 비젼