액티브 네트워크 성능향상을 위한 액티브 노드 구성 방안

최병선* 이성현* 이원구* 이재광*

*한남대학교, 컴퓨터공학과

Study on Method of Active Node for Performance Improvement on Active Network

Byoung-Sun Choi* Seoung-Hyeon Lee* Won-Goo Lee* Jae-Kwang Lee*

*Department of Computer Engineering, Hannam Univ.

요 약

본 논문에서는 액티브 네트워크 상에서 강력한 자원 관리와 액티브 응용의 제어를 위해 접근제어 메커니즘을 적용한 안전한 리눅스 커널을 분석 설계하였다. 설계된 접근제어 모델은 직무기반 접근제어를 이용하여 권한을 효과적으로 통제하고, 신분 및 규칙기반 접근제어를 이용하여 정보 및 시스템의 비밀성, 무결성, 가용성의 보장 및 시스템의 불법적인 접근을 방지할 수 있다. 리눅스 마이크로 커널 기반 접근제어 모델을 직무, 보안등급, 무결성 등급 및 소유권의 다단계 보안 정책을 기반으로 시스템의 불법적인 접근, 직무기반, 소유권 등의 다단계 보안 정책을 기반으로 하여 시스템의 불법적인 접근을 통제 할 수 있다.

I. 서론

현재의 액티브 네트워크에서는 네트워크 구조를 프로그래밍이 가능하도록 하고, 사용자 요구 기능을 수행할 수 있는 프로그램 코드를 전송 및 실행함으로써 통신망에 새로운 서비스를 보다 신속하고 경제적으로 도입하여 망 자원을 보다 적절하게 활용할 수 있도록 하고 있다. 이러한 모든 실행의 기반은 액티브 노드 OS 상에서 이루어지게 된다. 하지만, 현재의 노드 OS에서는 이러한 처리를 대부분 보안이 되지 않은 상태에서 수행하고 있다. 즉, 원천적인 노드 OS의 취약성을 드러낼 수 있게 된다. 따라서 기존의 OS 기술을 현재의 노드 OS에 적용하여 보다 안전한 OS 보안을 추구할 수 있다. 액티브 노드 시스템들은 중요한 체제를 연구하여 안전하고 신뢰성 있는 시스템 개발에 많은 투자를 하고 있다. 이러한 노드 OS 보안을 위해서 보안 커널(Secure Kernel)로 구현하고 있으며, 마이크로 커널 기술을 적용하여 보안 커널을 설계 구현하고 있다[1]. 컴퓨터 네트워크를 통한 해킹 수법이 갈수록 지능, 고도화, 국제화되고 있음에도 불구하고 국가적으로 중요한 비밀 정보를 보안 대책 없이 컴퓨터 및 네트워크 시스템을 통해 외부로 유출된다면 위험한 일이 아닐 수 없다. 따라서 국내에서도 정보보호 확립 차원에서 보안 커널을 국산화하고, 안전한 OS개발은 필수적이라 할 것이다. 본 논문에서는 DAC, MAC, RBAC등의 보안 정책과 보안모델 및 리눅스 OS의 보안 요구 사항이 될 수 있는 마이크로 커널과 TCSEC, CC에 관한 내용을 소개하고, RBAC 메커니즘을 적용한 액티브 네트워크 상에서의 안전한 노드 OS 구현에 대해 기술하고자 한다.

※ 본 연구는 대학 IT 연구센터 육성/지원사업의 연구결과로 수행되었음

II. 관련연구

1. 보안 정책(Security Policy)

1) DAC(Discretionary Access Control) 정책

DAC 정책은 사용자 계정(ID)와 각 파일에 대해 시스템에서 허용된 읽기, 쓰기, 실행 등의 접근

모드를 나타내는 인증을 기반으로 하는 정책을 채택하고 있다. 이 정책에서는 사용자가 어떤 객체에 대한 접근을 요청할 경우 명시되어진 인증을 검사하여 그 사용자가 그 객체를 특정 모드로 접근할 수 있는 경우에는 접근을 허용하며 그 외에는 거부된다. 이러한 정책은 많은 상업적인 운영체제 시스템들과 응용 프로그램들에 융통성 있는 적용이 가능하다는 장점을 가지고 있다. 반면에 하나의 데이터에 접근이 가능한 사용자가 자격이 없는 사용자에게 그 데이터를 전달할 수 있게 되므로 시스템 안전에서 정보의 흐름에 대한 실제적인 보안이 보증되지 못하는 단점을 갖는다[2].

2) MAC(Mandatory Access Control) 정책

MAC 정책은 DAC 정책에 비해 보다 견고한 접근제어 방법을 제공한다. 이 정책은 각 사용자와 객체에 보안 수준을 할당하여 시스템내의 주체(subject)와 객체(object)를 계층화(Classification)하는 것을 접근제어의 기본 방법으로 삼고 있다. MAC 정책은 군대나 정부 등 보다 강력한 접근제어가 필요한 환경에서 사용되어 왔으며 TS(Top Secret), S(Secret), C(Confidential), U(Unclassified) 등의 보안 수준을 계층적으로 정의하고 있다. 앞서 설명한 것과 같이 MAC는 DAC에 비해 보다 강력한 접근제어를 할 수 있으나 그에 반해 대부분의 상업적인 사업들의 요건을 만족시킬 수 있는 유동성이 부족하여 널리 사용되지 않는다[3].

3) RBAC(Role-Based Access Control) 정책

RBAC은 시스템 안에서 사용자가 현재 실행하고 있는 행동을 기반으로 정보에 대한 사용자의 접근을 제어한다. RBAC에서는 특정 행동에 부여되는 행위들과 책임(responsibility)들의 집합으로써 역할을 정의하고 있으며 사용자들은 자신이 속해있는 역할에 의해서 정보에 대한 접근이 제어된다. RBAC 정책의 특징은 사용자와 정보를 직접적으로 연결하지 않고 사용자들 역할에 연결하고, 역할과 정보객체 사이에 접근 권한을 연결하여 사용자 권한을 논리적으로 나누게 된다. 이러한 논리적 분할은 보안의 관리문제를 간단하게 해결할 수 있다. RBAC에서는 접근제어가 적용되는 환경에 따라 많은 역할들이 정의된다. 역할의 분할은 적용되는 환경의 계층구조 또는 실행 환경에 의해 구분되어 진다[4].

2. 보안 모델(Secure OS Model)

1) 래티스(Lattice) 메커니즘

접근제어 매트릭스와 같이 주체와 객체에 직접 접근 권한을 부여하는 대신에, 주체와 객체가 지닌 속성(예:보안등급(Security Level), 범주(Category))를 접근제어에 사용한다. 래티스는 보안 클래스들을 보안의 중요도에 따라 비교우위를 가려서, 이를 선형으로 배열시킨 구조이다. 래티스는 수학적으로 정의하면, 부분 순위 연산자(partial ordering)인 <=을 지닌 집합 L로서, 집합 L의 임의의 두 원소인 a,b 가 존재할 때, 최소상한선과 최대 하한선이 집합 L 에 포함되어야 한다. 이때, 최소상한선과 최대 하한선은 각각 유일한 값이어야만 하며, 이러한 유일한 값을 이용하여 접근 권한을 결정하는 메커니즘이다.

2) BLP(Bell-Lapadula) 메커니즘

BLP 모델은 데이터의 접근제어를 통해서 시스템의 비밀성(confidentiality)을 보호하기 위한 모델이다. BLP 모델을 접근제어 매트릭스와 보안수준을 통해서 주체가 객체에 접근하는 것을 통제함으로써 시스템의 비밀성을 보호한다[5].

3. 보안 커널(Secure Kernel)

1) 통합 커널(Monolithic Kernel)

현재 일반적인 리눅스 시스템은 대부분의 시스템 기능들이 단일 주소공간의 커널에 밀집되어 있는 통합(monolithic)커널 형태이다. 보안 통합커널은 통합 커널 안에 보안 정책 및 접근통제 메커니즘을 하나의 커널에 구현한 상태이다. 속도가 빠른 장점이 있는 반면, 이러한 형태는 정책 및 메커니즘의 변경이 있을 때 복잡한 구조로 인하여 번거로운 작업이 된다.

2) 마이크로 커널(Micro Kernel)

마이크로 커널은 커널의 핵심적인 부분 이외의 부분은 사용자 영역에서 수행되는 서버가 시스템 기능성을 담당하도록 구현된다. 마이크로 커널에 포함된 기능은 태스크와 쓰레드 관리의 하위 부분, IPC(Interprocess Communication)와 동기화, 메모리 관리의 하위 부분, 최소 디바이스 관리, 시스템 운영 중 발생하는 각종 인터럽트 처리 등이다. 파일 시스템, 네트웍 프로토콜, 표준 유닉스 인터페이스 등과 같은 운영체제 서비스의 상위 부분은 사용자 수준의 서버 프로그램으로 구현된다. 각 서버는 IPC를 통해서 상호 동작함으로써 기존의 운영체제 상에서 서비스를 실현한다. 여기서 마이크로 커널은 애플리케이션 또는 시스템 서버와 하드웨어 사이에 메시지를 검증하고 전달하는 역할을 수행한다.[6.

III. RBAC 메커니즘 기반 보안 커널 설계

1. 마이크로 커널 기반 보안 운영체제

보안 운영체제의 개발 사례는 미국 등 각 선진국의 정부 및 민간 업체에서 많이 이루어지고 있으나 보안 운영체제의 특성상 공개되는 것을 꺼려하므로 미국 정부 주도하의 몇 가지 개발 사례를 제외하고는 그 기술적인 내용을 분석하기가 힘들었다. 현재 미국에서 개발하고 있는 Trusted OS 개발은 대부분 마이크로 커널 접근방식으로 진행중에 있으며, 아래의 그림 1과 같은 구조로 되어 있다. 마이크로 커널 접근 방식으로 개발하는 이유는 보안 운영체제의 설계 원칙인 유연성, 투명성, 모듈성, 계층적 보안성 등 설계 원칙을 잘 만족시킬 수 있기 때문이다. 본 장에서는 이러한 마이크로 커널 기반의 보안 운영체제인 L4와 Flask의 커널 구조를 살펴보고, 끝으로, 이러한 기존의 마이크로 커널 기반 운영체제를 기반으로 하여, 본 논문에서 제안한 RBAC 정책을 적용한 마이크로 커널 기반의 리눅스 운영체제를 설계하였다.

1) L4

L4는 일반적으로 마이크로 커널 추상화의 최소 집합(쓰레드 및 주소공간)에 기반하며, 단지 7개의 시스템 호출로 구현되었고, 12KB의 코드를 갖는다. L4 마이크로 커널은 쓰레드와 주소공간 내부에서 활성화되어 실행되는 것이다.서버는 커널과 사용자 프로세스 사이의 데이터 교환을 위해서 물리적인 Copy-in과 Copy-out을 사용하며, 대부분의 응용프로그램을(X-windows 제외) 사용할 수 있다.

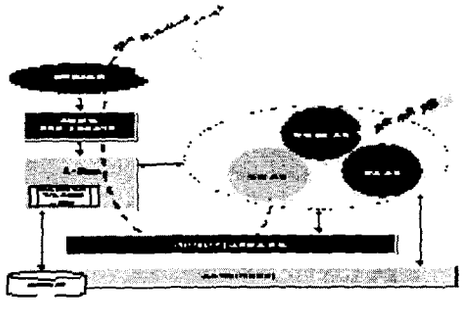

그림 1: L4Linux MLS 시스템 구성도

시스템 환경은 위의 그림 1에서 보는 바와 같이 L4/x86 마이크로 커널과 완전 호환성을 갖는 Fiasco v0.6 마이크로 커널 상에서 OS personality로 L4Linux(Linux Kernel 2.0.21)이 존재하며, 기존의 어플리케이션들은 L4Linux를 통해서 아무런 변경 없이 사용할 수가 있다. 또한 보안 정책 서버로서 인증 서버, 접근 제어 서버, 감사/추적 서버가 독립된 personality로 존재하게 되는데 이러한 보안 정책 서버는 보안 관리자만이 변경 및 설정을 할 수가 있다.[7]

2) Flask

기본적인 Flask 구조는 그림 2에서와 같이, 파일 관리자, 프로그램 관리자 등과 같은 객체 관리자(object manager)와 보안 서버(Security Server)로 구성된다. 객체 관리자는 시스템의 제어 동작을 제공하고 보안 서버는 특정 보안 정책에 대한 보안 결정을 담당한다. 객체 관리자는 그러한 보안 결정에 대한 수행을 담당한다. 특정 보안 정책이 요구되어 보안 정책에 대한 변경 사항이 필요하다 하더라도 객체 관리자는 보안 정책에 대해 독립적으로 운영되면 보안 서버만이 특정 보안 정책을 반영하기 위해 변경될 수 있다.

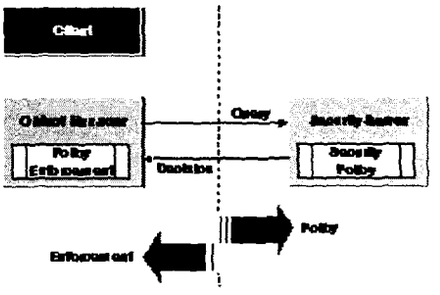

그림 2: 기본적인 Flask 구조

Flask(Fluke Security Kernel)는 고수준의 신뢰성 있는 시스템을 지원하는 Fluke(Flux μ-kernel Environment)의 한 버전이며 Flask 시스템의 목표는 다양한 보안 정책을 수용할 수 있는 유연성(flexibility)을 지원하는데 있다. Flask의 보안 구조는 DTOS(Distributed Trusted Operating System)에서 유래되며 정책 유연성(flexibility) 뿐만이 아니고 appl-ication transparency, defense-in-depth, ease of assurance과 관련 정책 변화에 따른 최소한의 코드 변환 등을 그 목적으로 한다. 이와 같은 목적을 제공하기 위하여 보안 구조는 다음 두 가지 요구 사항을 만족해야 한다. 첫째, 객체들(objects)에 대한 모든 접근을 중재할 수 있는 주체와 객체간의 분리 기능이 있어야 한다. 둘째, 객체에 대한 접근을 시도하는 주체에 대한 안전한 신분확인 기능을 제공하여야 한다.

2. 제안된 보안 운영체제 설계

다양한 보안 정책을 수용하기 위해서 다중 보안

정책을 지원한다는 것만으로 해결되지 않는다. 다양한 보안 정책을 수용하기 위해서는 첫째, 보안 정책에 의해 통제되는 high-level functions이 사용하는 low-level objects에 대한 적절한 접근 통제가 이루어져야 한다. 둘째, 접근 권한은 보안 정책과 정확히 일치되어야만 한다. 셋째, 정책은 정적(static)이지 않아야 한다. 정책의 변화 또는 동적 정책(dynamic policy)을 위하여 기존에 허용된 접근 권한에 대한 폐지(revocation)가 이루어져야 한다.

1) 정책 모듈 설계

① BLP 모듈 설계

아래의 그림 3에서 알 수 있듯이, 본 논문에서 제안한 BLP 메커니즘을 적용하여 보다 안전한 시스템에서 정보 흐름의 허용 가능한 경로를 보여주고 있다.

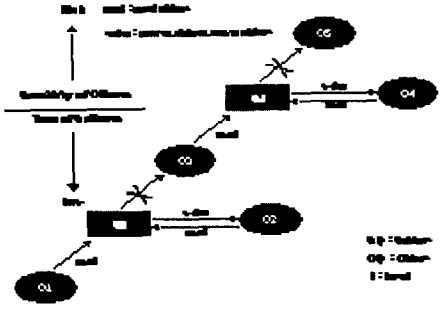

그림 3: BLP 접근 모델

즉 이러한 수정된 메커니즘은 서로 다른 보안 등급을 가지고 있는 데이터를 다루는 시스템에서 불법적인 정보 유출을 막기 위해 필요한 보안 요구 조건에 대해 정의하는데 기본적인 성질은 다음과 같다.

- 주체 S(i)는 객체 O(i)를 오로지 C(S) >= C(O)일 경우에만 읽을 수 있다.
- 주체 S는 객체 O를 오로지 C(S) <= C(O)일 경우에만 쓸 수 있다.

원래의 BLP 모델은 ②에서와 같이 C(S) <= C(O)일 경우에도 Write연산을 허용하였다. 그러나, 그림 3에서 보는바와 같이 Write 연산이란 생성(Create), 붙여쓰기(append), 삭제(Delete) 등의 포괄적인 연산이다. 그러나 낮은 등급의 주체가 더 높은 등급의 객체를 붙여쓰기, 삭제가 가능하다는 것은 현실적으로 보안상 문제가 발생되므로, 쓰기 연산은 제한되어야 한다[7].

② RBAC 모듈 설계

RBAC 시스템은 리눅스 운영체제가 탑재된 PC를 기반으로 RBAC 기법을 인트라넷상에서 이용할 수 있도록 설계한 보안 시스템으로 관리능력을 가지고 있어 권한이 부여된 데이터 관리를 폭넓게 수행할 수 있다. 이는 RBAC 시스템에 관리도구가 있어 관계정보를 일관성 있게 유지하여 주기 때문이다. 이를 위해서는 많은 집합과 함수들이 필요하며, 이러한 집합과 함수를 이용하여 보안 운영체제에 적용할 수 있는 RBAC 모듈은 RBAC96 모델[8]에 기초하여 설계하였다. 보안 운영체제의 여러 구성요소 중에서 관리자가 RBAC을 관리하는데 있어서 가장 중요한 구성요소가 관리도구이며, 위에서 보여진 역할 간의 관계에 있어 사용자-역할, 역할-역할 관계를 데이터베이스에 저장하고 관리한다. 따라서 이들 관계에 대한 데이터베이스 정보가 일관성 있게 유지되어 있다.

③ 감사 모듈 설계

기존의 리눅스 OS에서의 감사 추적은 정적으로 기록된 사건(event) 로그 파일을 이용하여 별도의 감사 추적 프로그램을 통해서 가능하였다. 이 경우에는 보안 관리자 또는 감사인(auditor)에 의한 실시간 감사 추적이 불가능할 뿐만 아니라 다양한 사건 정보의 출력이 어려웠다. 따라서 그림 4에서 볼 수 있듯이, 감사 모듈에서는 동적인 데이터베이스를 이용하여 실시간으로 감사 추적을 할 수 있도록 시스템을 설계하였으며, 로그 정보를 데이터베이스에 실시간으로 저장하도록 구성하였다. 이 감사 모듈의 로그 정보는 본 시스템에 적합하게 정의된 다양한 속성(attribute) 정보들을 저장한다. 보안 관리자는 SQL 질의어를 통해서 저장된 정보를 실시간으로 원하는 정보만을 볼 수 있다. 이러한 로그 데이터베이스 파일은 보안 등급이 설정되어 있어 시스템 관리자와 root, 그리고 일반 사용자는 접근이 통제되며, 읽고/쓰기가 불가능하다. 오직 보안 관리자만이 인가된 보안 등급으로 이러한 작업을 할 수 있다.

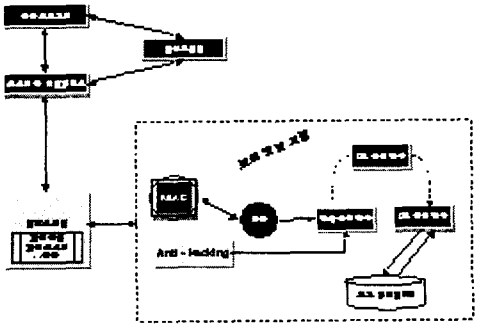

그림 4: 감사 모듈 구조

2) 마이크로 커널기반 보안 서버 설계

본 논문에서 제안한 보안서버는 그림 5의 구조도 에서 간접적으로 알 수 있듯이, MLS(Multi-level Security), Type Enforc-ement, 신분 기반 접근 통제(Identity-based Access Control), 동적 역할 기반 접근 통제(Dynamic Role-based Access Control)와 같은 세 가지 세부 정책의 조합으로 보안 정책을 수행한다. 보안서버에 의해 제공되는 접근 결정은 이 세 가지의 세부 정책을 만족하며, MLS의 구현과 동적인 역할 기반 접근 통제(Dynamic Role-based Access Control)의 지원, 신분 기반 접근 통제(Identity-based Access Control)의 보강된 지원한다는 측면에서 기존의 보안 서버와는 다르다.

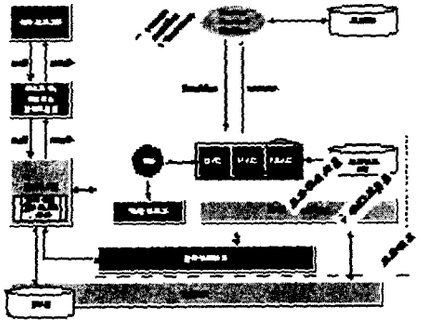

그림 5: 제안된 마이크로 커널 기반의 보안 운영체제 구조도

IV. 결론 및 향후 연구

인터넷은 컴퓨터 네트워크를 통한 해킹 수법이 갈수록 지능, 고도화, 국제화되고 있음에도 불구하고 국가적으로 중요한 비밀 정보를 보안 대책 없이 컴퓨터 및 네트워크 시스템을 통해 외부로 유출된다면 위험한 일이 아닐 수 없다. 따라서, 본 논문에서는 안전한 운영체제를 위해 리눅스 마이크로 커널에 역할기반 접근제어 메커니즘을 적용하였다. 마이크로 커널에 RBAC을 적용하여 역할에 따른 사용자의 접근 권한을 다양하게 부여할 수 있다. RBAC메커니즘을 이용한 마이크로 커널 기반의 보안 운영체제에 대한 완벽한 설계와 이 설계를 바탕으로 TCSEC B2등급의 보안 운영체제를 구현할 것이며, 이는 다양한 IDS 및 방화벽의 하부 구조에도 이용될 것이다. 마이크로 커널에 기반 한 RBAC 메커니즘을 적용한 안전한 리눅스 커널을 분석하고 설계하였다. 이러한 설계 방법은 보안 관련 서버가 마이크로 커널로 설계되어 안전성 측면에서 이점이 있으며, 기존 리눅스 커널의 수정을 최소화하면서, 다른 접근제어 모델을 사용할 때에는 관련 모듈만 교체만 하면 되기 때문에 이미 설정된 보안 정책을 손쉽게 변화 할 수 있다. 후에 통합 커널 방식의 보안 시스템과의 성능 비교가 더 진행되어야 할 것이다.

참고문헌

[1] 박태규, 임연호, "리눅스 커널 기반의 안전한 OS개발," Proc. of KOSTI 2000, 2000년 12월.

[2] 손득종, "디자인 패턴을 이용한 객체 지향적인 RBAC모델," 석사학위 논문, 아주대학교, 2000년 2월.

[3] J. G. Ko et al. "Design and Implementing for Secure OS based on Linux," Proc. of WISA2000, pp.175-182, Nov. 2000.

[4] 김대중, 김현정, 김정래, 박태규, 조인구, 임연호 "다중등급보안 리눅스 기반의 RBAC 시스템 구현," Proc. of CISC 2001, pp.39-42, 2001년 11월.

[5] 박재경 "Linux에서 BLP 보안모델의 특성구현," 석사학위 논문, 홍익대학교, 1996년 2월.

[6] Ulfar Erlingsson ,Athanasios Kyparlis "Microkernels," http://os.korea.ac.kr/~yuko/research/microkernel/cornel.htm.

[7] 홍승표, 최용호, 박태규 "L4Linux 다단계 보안을 위한 접근제어 서버 프로토타입 설계 및 구현," Proc. of CISC '99, pp.97-107, 1999년 11월.

[8] The University of Western Ontario and RAVI SANDHU and QAMAR MUNAWER George Mason University, "Conf-iguring Role-Based Access Control to Enforce Mandatory and Discretionary Access Control Policies", ACM Transactions on Information and System Security, Vol. 3, No. 2, pp.85-90, May. 2000.