JNI를 이용한 실시간 네트워크 보안 시스템

김건우°, 장희진, 박보석, 김상욱
경북대학교 컴퓨터과학과 컴퓨터언어연구실

Realtime Network Security System using JNI

Gunwoo Kim°, Heejin Jang, Boseok Park, Sangwook Kim
Department of Computer Science, Kyungpook National University

요 약

분산 환경의 발달과 네트워크를 이용한 원거리 컴퓨터 사용이 증가되면서 발생하게 된 전산망 침해 사고를 항상 감시하고 분석하여 자동적으로 대처하기 위한 전산망 보안 시스템이 연구되고 있다. 본 논문에서는 자바 네이티브 인터페이스를 이용한 실시간 네트워크 보안 시스템을 제안하다. 자바의 시스템 자원에 대한 엑세스의 한계를 극복하기 위해서 JNI를 이용한 C/Java의 연동을 통하여 보다 효율적이고 용이한 정보 수집 및 취약점 분석을 가능하게 한다. 또한 점검하고자 하는 호스트에 보안 정보 수집 모듈을 멀티 쓰레드로 상주시킴으로써 실시간으로 원하는 정보를 얻어 선택된 점검 대상에 가해지는 모든 활동들을 분석한다..

1. 개 요

전산망 내에서 취약점을 발견하여 사고를 방지해야 하고 침해 사고가 일어났을 경우에는 사고의 빠른 발견과 그에 따르는 신속한 조치가 요구된다. 이를 위해 점검하고자 하는 슬레이브 시스템의 각 보안 정보 수집 컴포넌트는 마스터에서 설정한 정책에 따른 시스템 자원에 대한 정보와 취약점을 수집하여 이를 마스터로 전송함으로써 실시간으로 보다 효율적이고 빠르게 취약점 분석 및 침해 사고를 방지할 수 있다.

전산망 보안 시스템에 대한 연구가 활발히 진행되고 있지만 슬레이브에 대한 감시 기능을 제공하는 중앙 집중식 보안 관리 시스템을 제공해주는 시스템은 별로 없다. COPS와 같은 대부분의 네트워크 보안 도구들은 마스터 시스템을 이용한 중앙 집중식이 아니라 각 시스템을 독립적으로 관리하는 기능을 제공하고 있다. SATAN의 경우 로컬 시스템뿐만 아니라 리모트 시스템에 대한 보안 분석이 가능하고 각 호스트뿐만 아니라 점검 대상으로 영역을 설정하고 점검 정도 설정이 가능하지만 웹 클라이언트를 사용하므로 웹에 한정되는 단점을 가지고 있다. OmniGuard 또한 중앙 집중식 보안 관리 시스템을 제공하고는 있지만 시스템에 종속적이다[1].

본 논문에서는 JNI를 이용한 실시간 네트워크 보안 시스템을 제시함으써 점검하고자 하는 시스템 정보와 취약점을 보다 효율적으로 수집하고 점검할 수 있으며 마스터에서는 실시간으로 이를 체크하여 해당 호스트에 적절한 조치를 취할 수 있다.

본 논문의 구성은 다음과 같다. 제 2절에서는 네트워크 보안 시스템의 구조에 대해 기술한다. 제 3절에서는 슬레이브 모듈에 대해 설명하고 이에 연관된 요소들의 관계를 정의하며 구현 예를 보인다. 제 4절에서는 앞으로의 연구 방향을 제시하고 결론을 맺는다.

2. 네트워크 보안 시스템의 구조

네트워크 보안 시스템은 기본적으로 모니터 모듈, 마스터 모듈, 슬레이브 모듈로 구성된다(그림 1). 모니터 모듈은 보안 관리 시스템에서 수집한 호스트들의 보안 정보들을 가공하여 망 관리자에게 보고한다. 마스터 모듈은 모니터 모듈로부터 설정된 영역과 정책을 저장하고 이에 따라서 슬레이브 모듈로부터 정보를 수집해서 이를 저장, 통계, 분석하고 해당 시스템에 적절한 대응을 한다. 또한 슬레이브 모듈은 점검하고자 하는 호스트에서 발생하는 보안 정보를 수집하고 취약점을 분석해서 이를 마스터 시스템에 보고하는 형태이다.

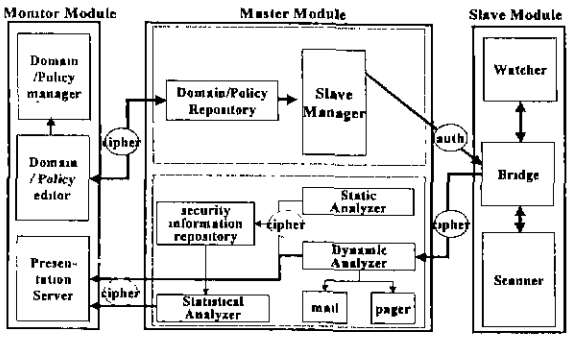

그림 1 네트워크 보안 시스템

2.1 모니터 모듈

분석/정리된 통신망 트래픽 정보, 시스템 및 네트워크 보안 문제 정보를 멀티미디어를 이용하여 도식화해서 관리자에게 디스플레이한다. 관리해야할 호스트를 선택하는 영역(Domain) 편집 기능은 네트워크 보안 시스템에 유동성을 부여하여 관리 대상을 확대/축소 가능하도록 한다. 또한 각 영역 및 호스트별로 설정에 맞는 영역을 설정하고 이를 관리할 수 있다.

2.2 마스터 모듈

모니터 모듈로부터 설정된 영역과 정책을 저장하고 슬레이브 호스트에 멀티캐스팅해서 이러한 영역과 정책에 해당하는 시스템 정보 및 취약점을 분석할 수 있게 한다. 이 과정에서 보안 관리 시스템과 보안 정보 수집 시스템간의 인증을 엄격히 행하는 프로토콜과 암호 알고리즘을 사용한다[2]. 또한 슬레이브 호스트로부터 전송되는 지속적인 시스템 보안 정보나 취약점을 저장, 분석하고 통계적인 정보로 가공한다 이러한 정보를 모니터 모듈을 통해서 망 관리자에게 디스플레이 할 뿐만 아니라 시스템에 어느 수준 이상의 위험한 상황이 발생하게 되면 그 시스템 관리자에게 메일이나 호출을 통해서 상황을 통보할 수 있다.

2.3 슬레이브 모듈

점검하고자 하는 호스트의 보안 정보나 취약점을 수집하고 이를 마스터로 전송함으로써 적절한 조치를 취할 수 있도록 한다. Scanner는 마스터로부터 요청이 왔을 경우 그에 해당하는 자료를 수집, 전송하지만 Watcher의 경우에는 계속 시스템에 상주해서 중요한 보안 정보나 위험한 상황이 발생했을 경우 이를 마스터로 전송한다. 자바는 시스템의 낮은 레벨까지는 엑세스할 수 없기 때문에 C와 연동해서 사용하고 이를 가능하게 하기 위해서 JNI를 사용한다.

3. 실시간 네트워크 보안 시스템

실시간 네트워크 보안 시스템은 보안 관리 마스터의 정책 설정에 대하여 해당 슬레이브 호스트의 보안 정보를 수집, 분석하여 실시간으로 보안 관리 마스터 시스템에 보안 정보를 전송한다.

3.1 실시간 네트워크 보안 시스템의 구성 요소

슬레이브 모듈의 구성 요소에는 수집/분석되는 정보의 범주에 따라 3가지가 포함되고, 주기적으로 이러한 정보 수집을 수행시키고 수집된 정보를 관리자에게 전달해주는 전송 부분과 호스트에서 발생하는 중요한 이벤트를 관리자에게 통보해주는 이벤트 통보 부분으로 구성된다.

Watcher(실시간 점검 컴포넌트)는 점검하고자 하는 시스템에 멀티 쓰레드의 형태로 상주하면서 중요한 시스템 정보나 위험 상황이 발생했을 경우 이를 마스터로 통보하는 역할을 한다. 관리자가 요청하는 수집 정보는 물론 미리 정의된 이벤트의 발생 시에 이를 관리자에게 통보한다. Scanner는 망 관리자가 설정한 정책에 따라 시스템 검사, 계정 검사, 파일 무결성 검사를 통해 각종 보안 정보를 마스터로 전송한다. Bridge는 JNI 모듈을 포함하고 있어 점검 대상 호스트 자원에 대한 낮은 레벨의 접근을 가능하게 하고, 또한 RMI를 통해서 마스터 시스템과의 직접적인 양방향 통신 채널을 제공한다.

Watcher, Scanner, Bridge의 기능을 보면 시스템 검사, 계정 검사, 파일 무결성 검사, 수집 정보 전송, 실시간 이벤트 통보의 5가지 요소로 분류할 수 있고 이를 이용하여 실시간 보안 에이전트의 전체 개념을 나타내보면 그림 3.1과 같다.

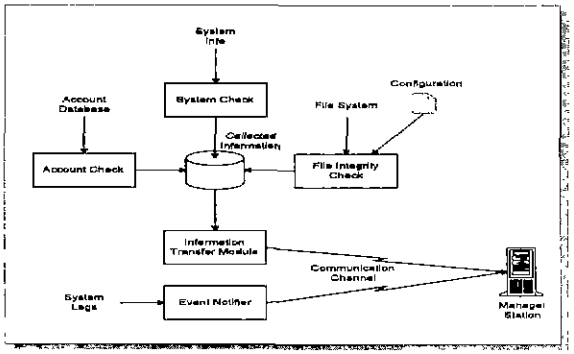

그림 3.1 슬레이브 시스템의 구성 요소

3.2 마스터/슬레이브 간의 실시간 네트워크

보안 정보 수집 컴포넌트는 시스템 커널 정보와 데이터베이스 자료구조, 그리고 시스템 함수를 사용한다. 따라서 자바 가상 기계에서 구현된 마스터 시스템과 슬레이브 시스템과의 연결 설정을 위해서는 두 번의 단계를 거쳐야한다.

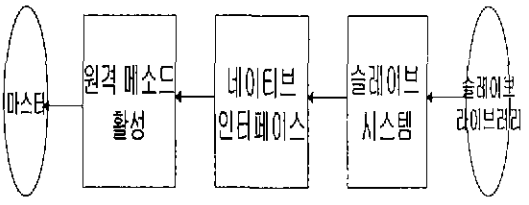

그림 3.2 슬레이브 라이브러리 접근 과정

3.2.1 RMI를 통한 마스터와의 통신

RMI 인터페이스는 AgentServerInterface.java에서 정의되어 있다. 시스템 확장시에 인터페이스를 확장하고, AgentServer에 추가된 가상 메소드를 구현하여 객체 등록소(Object Registry)에 등록하면 된다. 또한 원격 객체는 AgentServerInterface에 선언된 메소드들 중에서 원격 호출을 지원하기 위한 메소드의 기능을 구현한다. getSystemInfo메소드는 원격 호출을 예외상황 처리를 위해서 RemoteException 예외상황 응용 프로그램에 던진다.

등록기의 생성 및 객체의 등록은 원격 객체를 저장하고 요청을 기다리는(listening) 역할을 하는 객체 저장소를 생성하며 createRegistry가 동적으로 객체 저장소를 생성한다. 인터페이스에 따라 구현된 원격 객체는 객체 등록기에 동적으로 바인딩되어야 클라이언트에서 호출 가능하다. 구현된 AgentServer객체를 생성하고 객체 등록기에 생성된 객체를 저장한다. 저장되는 객체는 URL포맷으로 패핑되어 등록기에 바인딩된다.

rmic 컴파일러에서 자동으로 Stub와 Skeleton을 생성한다. 본 시스템에서는 AgentServer_Stub.class와 AgentServer.Skeleton.class가 생성된다 클라이언트에서 Stub를 참조하여 원격객체를 참조하므로 클라이언트에 Stub객체를 복사한다. 그리고 서버의 위치(URL포맷)와 객체의 이름을 인자로 원격 객체를 리턴 받는다 클라이언트에서는 AgentServer가 로컬에 있는 것처럼 엑세스할 수 있다. 즉 AgentServer의 getSystemInfo를 활성화한다

3.2.2 낮은 레벨의 시스템 자원 접근

네이티브 메소드란 자바 클래스에서 선언되었지만 자바 언어가 아닌 언어로 구현된 메소드를 의미한다. 네이티브 메소드는 함수로 구현된다. 네이티브 메소드를 연결하기 위해서, 자바 가상 기계 쪽에서는 자바 클래스에 네이티브 메소드를 구현하는 함수를 제공해야 한다. 여기서 주의해야 하는 것은 자바 클래스에서 선언된 네이티브 메소드의 시그너쳐와 네이티브 언어로 작성된 시그너쳐는 서로 일치해야 한다.

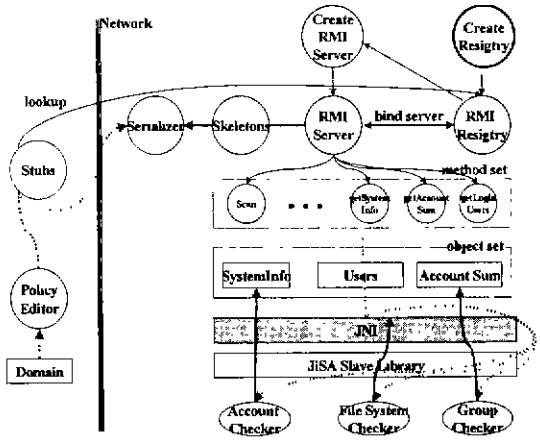

그림 3.3 JNI와 RMI를 통한 실시간 네트워크 보안 시스템

3.3 구현 예 : 실시간 네트워크 보안 시스템

이 절에서는 JNI와 RMI를 이용한 실시간 보안 에이전트 시스템을 적용한 예를 보인다 이 시스템의 가장 큰 특징은 점검하고자 하는 시스템에서 중요한 파일 변경이나 위험상황이 발생했을 경우 실시간으로 망 관리자에게 이를 통보하여 적절한 조치를 취할 수 있게 하고 이를 위해서 C와 Java를 연동 함으로써 시스템 자원을 보다 낮은 레벨까지 접근할 수 있다는 데 있다. 또한 마스터 시스템은 Java로 구현함으로써 시스템에 독립적이다.

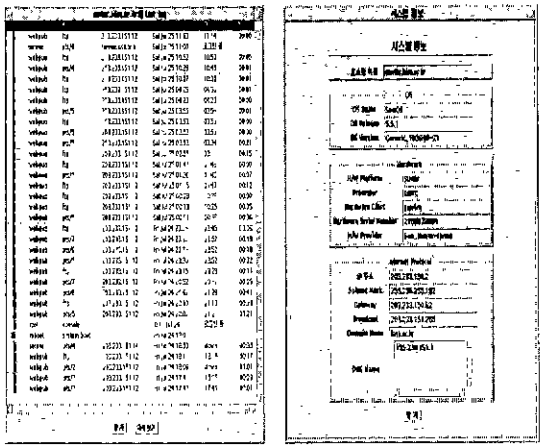

그림 3.4 실시간 네트워크 보안 시스템의 예

4 결론

지금까지 전산망 전체의 취약점을 한눈에 파악하고 수집된 정보들의 효율적인 분석과 빠른 대처를 위한 실시간 네트워크 보안 시스템을 제시하였다. 실시간 네트워크 보안 시스템은 Bridge와 Watcher, Scanner로 구성되어 있다. Bridge는 RMI를 사용해서 마스터와의 연동을 가능하게 하고 JNI를 통해서 보다 낮은 레벨의 시스템 자원에 접근할 수 있게 한다. Watcher는 시스템을 계속 점검하다가 위험 상황이 발생할 경우 마스터에 통보해서 적절한 조치를 취할 수 있도록 하고, Scanner는 실정된 정책에 따른 마스터로부터의 요청에 대한 정보 수집을 한다. 앞으로의 연구방향은 네트워크 보안 시스템을 기반으로 마스터 시스템에서 로컬 시스템이나 리모트 시스템에 대해서 시스템에 독립적이고 실시간으로 대처하는 모듈을 구현하는 것이다

5. 참고 문헌

[1] 한국정보보호센터, "공개 보안 도구," 정보보호현황, 1996,10,pp229-247

[2] Stallings,W. Network and Internetwork Security Principles and Practice. NewJersey, NY: Prentice-Hall, 1995