通信情報保護學會論文誌
第 9 卷 第 1 號, 1999. 3

혼합형 침입차단시스템을 위한 통합 접근제어 규칙기술 언어 및 그래픽 사용자 인터페이스 구현

박 찬정*

Implementation of an Integrated Access Control Rule Script Language and Graphical User Interface for Hybrid Firewalls

Chan-Jung Park*

요 약

기존의 혼합형 침입차단시스템은 네트워크 계층에서의 패킷 필터링 기능과 응용 계층에서의 응용 게이트웨이 기능을 함께 수행하기 때문에, 모든 데이터가 응용 프로그램 계층에서 처리되는 게이트웨이 방식 침입차단시스템에 비해 성능이 뛰어나고 게이트웨이 방식 침입차단시스템처럼 다양한 접근제어가 가능하면서도 사용자에게 투명성을 제공할 수 있다. 하지만, 네트워크 계층과 응용 계층에 대응되는 보안정책을 각각 설정하여 접근을 제어하기 때문에 관리자의 침입차단시스템 관리를 용이하게 하기 위해서 일관성있는 사용자 인터페이스 개발이 요구된다. 본 논문에서는 혼합형 침입차단시스템을 위한 그래픽 사용자 인터페이스를 구현하여 접근제어 및 로그 분석, 실시간 네트워크 트래픽 감시, 날짜별 트래픽에 대한 통계처리와 같은 관리기능을 제공한다. 이 때, 접근제어를 위해서 새로운 규칙 기술 언어를 함께 제안하고, 이를 이용하여 사용자가 요구하는 다양한 형식의 접근제어 규칙을 생성하도록 한다.

Abstract

Since a hybrid firewall filters packets at a network layer along with providing gateway functionalities at an application layer, it has a better performance than an IP filtering firewall. In addition, it provides both the various kinds of access control mechanisms and transparent services to users. However, the security policies of a network layer are different from those of an application layer. Thus, the user interfaces for managing a hybrid firewalls in a consistent manner are needed. In this paper, we implement a graphical user interface to provide access control mechanisms and *management facilities* for a hybrid firewall such as log analysis, a real-time monitor for network

* 한국통신 멀티미디어연구소 인터넷연구실

traffics, and the statisics on traffics. And we also propose a new rule script language for specifying access control rules. By using the script language, users can generate the various forms of access control rules which are adapted by the existing firewalls.

Key word : Access control, Rule script language, Hybrid firewall, Internet security

Ⅰ. 서 론

침입차단시스템(firewall)의 목적은 정당한 권리를 획득하지 않은 사용자가 내부 네트워크의 자원에 접근하려는 시도를 차단하는 것과 기밀성을 띤 정보가 허가를 받지 않고 네트워크를 통해 외부로 유출되는 것을 방지하는데 있다[1]. 일반적으로, 침입차단시스템을 구축하는 방식은 크게 두 가지로 나눌 수 있다. 첫째, 전체 네트워크를 보호하여야 할 내부 네트워크와 외부 네트워크로 물리적 구분을 한 후, 라우터(router), 브리지(bridge), 또는 두 개의 네트워크 카드를 가진 개인용 컴퓨터에 패킷 필터링(filtering) 기능을 추가하여 침입차단시스템을 설치하는 방식이다. 둘째, 네트워크를 외부 네트워크와 내부 네트워크로 물리적으로 구분하는 것이 아니라, 소프트웨어만으로 응용 프로그램 게이트웨이를 만들고 보호받아야 할 내부 네트워크 안에 놓인 모든 호스트들의 응용 프로그램들은 응용 프로그램 게이트웨이 안의 프락시(proxy)만을 통해서 외부 네트워크와 접속하고, 외부 네트워크에서 내부 네트워크로 접속할 경우에도 직접 내부 네트워크에 접속할 수 없고 게이트웨이의 프락시를 통해 접속하는 방식이다.[2]

최근, 침입차단시스템을 판매하는 회사들은 위와 같은 두 가지 형태의 침입차단시스템을 통합하여 만든 혼합형(hybrid) 침입차단시스템을 개발하고 있다. 혼합형 침입차단시스템은 네트워크 인터페이스 카드간에 패킷을 전달하고 패킷에서 필요한 정보를 얻어내는 역할을 하는 필터(filter) 모듈과 필터 모듈에서 전달해 준 정보와 네트워크 관리자가 만든 규칙을 이용해서 회선의 접속을 허락하거나 감시하는 것을 결정하는 침입차단(firewall) 모듈로 구성되어 있다. 필터 모듈은 라우터와 같은 기능을 수행하기 때문에 혼합형 침입차단시스템은 게이트웨이 방식의 침입차단시스템과는 달리 사용자에게 투명성을 보장해 줄 수 있다. 또한, 침입차단 모듈이 네트워크 서비스별로 자세한 접근제어를 할 수 있기 때문에 게이트웨이 방식 침입차단시스템처럼 접근제어와 로그(log) 기록을 수행할 수 있다[2].

혼합형 침입차단시스템에서는 게이트웨이 방식 침입차단시스템과 마찬가지로 접속을 허가 여부에 대한 결정이 응용계층에서 이루어지고, 실제로 접속이 이루어진 후에는 대부분의 데이터는 응용 프로그램 계층에서 처리되지 않고, 네트워크 계층에서 직접 전송된다. 혼합형 침입차단시스템의 장점은 대부분의 데이터가 네트워크 계층에서 처리되기 때문에 모든 데이터가 응용 프로그램 계층에서 처리되는 게이트웨이 방식 침입차단시스템에 비해 성능이 뛰어나고 게이트웨이 방식 침입차단시스템처럼 다양한 접근제어가 가능하면서도 사용자에게 투명성을 제공해 줄 수 있다는 것이다.

한편, 침입차단시스템 상품들은 시스템 관리자들이 시스템 오류 및 로그 분석, 실시간 네트워크 상태 감시, 트래픽 통계 분석 등을 용이하게 수행할 수 있도록 하기 위해서 그래픽 사용자 인터페이스를 제공하고 있다[3][4][5]. 특히, 혼합형 침입차단시스템의 경우, 네트워크 계층과 응용 계층에 대응되는 보안정책을

각각 설정하여 수행하기 때문에, 관리자의 침입차단시스템 관리를 용이하게 하기 위해서 더욱 일관성 있는 사용자 인터페이스를 요구한다.

본 논문에서는 응용 계층의 침입차단시스템인 Socks[6]와 네트워크 계층에서 패킷 필터링 기능을 수행하는 Ip_filter[7]를 혼합한 혼합형 침입차단시스템을 위한 그래픽 사용자 인터페이스를 구현한다. 주요 기능으로는 접근제어 및 로그 관리, 실시간 트래픽 감시, 일별 및 주별, 달별 트래픽 통계 처리가 있다. 이 중에서, 접근제어 규칙을 기술한 파일의 수정 및 관리는 혼합형 침입차단시스템의 경우에 관리자의 부담을 가중시킨다. 즉, 혼합형 침입차단시스템은 보안정책을 기술한 규칙기술 언어 혹은 명령어들을 이용하여 네트워크를 통해 외부로부터 수신되거나 내부로부터 송신되는 메시지들의 흐름을 제어하여야 한다. 하지만, 각 네트워크 계층을 위한 접근 규칙들을 기술하는 방식은 서로 상이하기 때문에 관리자는 접근제어 규칙을 기술하기 위하여 다양한 형태의 규칙기술 문법들을 다루어야만 한다. 따라서, 본 논문에서는 접근제어 정책을 기술하기 위한 새로운 통합 규칙기술 언어를 제안하고, 제안한 언어로 표현된 규칙을 특정 침입차단시스템에 대응되는 규칙으로 변환시키기 위한 컴파일러를 구현한다. 통합 규칙기술 언어는 침입차단시스템 관리자들에게 접근제어 규칙의 설정시 용이성을 제공할 뿐 만 아니라 확장성을 제공한다.

본 논문의 구성은 다음과 같다. 2 장에서는 기존의 침입차단시스템의 그래픽 사용자 인터페이스의 특성을 비교 분석하고, 3 장에서는 통합된 접근제어 규칙기술 언어에 대해 기술한 후, 사용자의 언어 사용을 용이하게 하기 위해서 Tcl/Tk[8][9]을 이용한 사용자 인터페이스를 구현한다. 4장에서는 침입차단시스템의 관리를 위해 구현된 기능들에 대해 기술하고, 5 장에서 결론을 맺는다.

Ⅱ. 침입차단시스템의 그래픽 사용자 인터페이스

이 장에서는 기존의 침입차단시스템에서 제공하는 사용자 인터페이스를 위주로 기능과 특징들을 살펴본 후, 문제점을 비교, 분석한다.

우선, Checkpoint사의 Firewall-1 침입차단시스템[3]은 세계적으로 가장 널리 사용되고 있는 제품으로서 혼합형 침입차단시스템이다. Firewall-1에서는 대부분의 보안정책이 그래픽 사용자 인터페이스를 통해서 정의되며, 제공되는 기능들은 다음과 같다.

첫째, Firewall-1은 객체-지향적인 규칙베이스(rulebase) 에디터를 제공한다. 규칙베이스는 서로 다른 호스트간의 모든 네트워크 연결에 대한 접근 허용 및 거절, 로그 등과 같은 행위를 기술한 데이터베이스로, 에디터를 통해 쉽게 규칙들을 갱신할 수 있다. 또한, 관리자는 에디터를 통해 네트워크, 사용자, 서버, 그리고 그들간의 관계를 쉽게 기술할 수 있다.

둘째, Firewall-1은 게이트웨이를 통해 전송되는 모든 연결에 대한 정보를 추적하고 감시하기 위해서 그래픽 로그 감시기능(viewer)을 제공한다. 온라인 감시 기능은 통신 행위와 경고 등의 실시간 감시를 가능하게 하며 이와 동시에 보안정책 설치 혹은 시스템 종료와 같은 중요한 네트워크 사건 등을 출력한다.

셋째, Firewall-1은 실시간 상태, 감사, 경고와 같은 시스템 상태 감시기능를 제공한다. 그리고 승인되거나 거절, 혹은 로그된 패킷의 수와 같은 패킷의 통계치를 제공한다. 예를 들어, Firewall-1은 수상한 행동에 대해 시스템 관리자에게 실시간적으로 경고 메시지를 출력한다.

넷째, 속임-방지(anti-spoof) 기능을 제공한다. 즉, 침입차단시스템 게이트웨이를 통해서

진입하는 패킷의 원천 IP 주소를 검사하므로써 잘못된 패킷을 검출해낸다. 만일, 수상한 패킷이 검출되면, 즉시 경고 메시지를 관리자에게 보낼 수 있고 로그에도 기록할 수 있다. 이 장에서는 사용자 인터페이스 부분만을 기술하였고 자세한 사항은 Firewall-1의 사용자 지침서[3]에 기술되어 있다.

패킷 필터링 기능을 수행하는 대부분의 침입차단시스템의 경우, 접근제어를 위해 기술한 규칙들은 기술된 순서에 의존적이어서 원하지 않는 연결을 허용할 수 있기 때문에, 관리자들은 규칙들을 보안정책을 보장하는 규칙 순서대로 정확히 기술하여야 한다. 이와 같은 문제점을 제거한 것이 Eagle 침입차단시스템[4]이다. 기본적으로 Eagle은 명시되지 않은 모든 접근은 거절하고, 기존의 패킷 필터 침입차단시스템과는 다르게 접근제어 규칙 적용 시, 최초-적합(first-fit) 접근 방식이 아니고 기술된 순서에 상관 없이 최적-적합(best-fit) 접근 방식을 채택하고 있다. Eagle의 그래픽 사용자 인터페이스 프로그램인 Hawk는 하나로 일관성 있게 설계된 윈도우로 이루어져 있고, 구성요소에 대한 형상(configuration) 관리 기능과 게이트웨이 감시 혹은 로그 파일의 감시와 같은 기능을 수행한다. 즉, 게이트웨이의 특징을 정의하고, 접근제어 규칙을 기록하며 게이트웨이 사용자 관리, 네트워크 개체 정의, 특수 서비스 정의, 프로토콜 정의, 접근제어를 정의하는 시스템 파일 교정과 같은 기능을 제공한다.

마지막으로, Borderware 침입차단시스템[5]은 게이트웨이 시스템과 침입차단 보안 시스템을 하나로 만든 혼합형 침입차단시스템으로써 듀얼-홈드(dual-homed) 침입차단시스템과 같이 네트워크 카드를 여러 개 두어 패킷 필터와 응용 단계의 게이트웨이, 회선 단계의 게이트웨이를 구현한다. 표준의 네트워크 서비스 프로그램들이 수정되지 않고 동작할 수 있게 하는 프락시들을 두어, 외부 네트워크로부터의 투명한 접근을 가능하게 한다.

Borderware 침입차단시스템은 2 개의 사용자 인터페이스로 설계되었고, 원래의 메뉴-기반 콘솔 인터페이스는 시스템 패치(patches)와 원격 계정 관리를 위한 설치 등과 같은 최소한의 설치 절차를 보유하고 있다. 관리적인 기능의 대부분은 그래픽 사용자 인터페이스를 통해 진행되는데 사용자 인터페이스의 설계는 운영체제 플랫폼과는 무관하며 자바-기반의 웹 브라우저를 통해 원격 콘솔 접근 기능을 제공한다.

앞에서 기술한 3 개의 침입차단시스템에서 제공하는 사용자 인터페이스들이 제공하는 기능들을 살펴보면, 크게 접근제어, 로그 분석 및 관리, 네트워크 상태 감시, 시스템 자원 관리, 사용자 관리 등으로 구분할 수 있다. 이와 더불어 제품들은 각각 고유한 사용자 인터페이스를 제공하고 있다. 하지만, 비용의 측면을 고려해 볼 때, 고가의 제품이라는 문제를 가진다. 반면, 본 논문에서 구현하는 침입차단시스템의 사용자 인터페이스는 다른 침입차단시스템에서 제공하는 기능들을 제공하면서 공개용 침입차단시스템 프로그램들에서 사용하고 있는 서로 다른 접근제어 규칙 기술 방식을 하나의 방식으로 통합한 그래픽 사용자 인터페이스를 제공한다. 즉, 통합된 인터페이스를 제공하므로써 관리자는 하나의 인터페이스로부터 자신들이 원하는 침입차단시스템 프로그램의 접근제어 규칙을 생성하게 되어, 관리를 용이하게 할 수 있고 확장성을 향상시킬 수 있다.

Ⅲ. 혼합형 침입차단시스템의 접근제어 규칙을 기술하기 위한 사용자 인터페이스

이 장에서는 새로운 규칙기술 언어를 제안

하고, 이를 바탕으로 단일한 형태의 그래픽 사용자 인터페이스를 구현한다. 관리자는 관리자 윈도우를 통해 메뉴버튼을 클릭(click)하거나 텍스트를 입력하여 규칙을 기술한다. 규칙에 대한 기술이 완료된 후, 사용자가 원하는 특정 규칙기술 언어을 선택하면, 해당되는 규칙이 생성된다. 본 논문에서 제공하는 침입차단시스템의 사용자 인터페이스는 Reed[10]가 제안한 필터-언어를 기반으로 규칙기술 언어를 확장한다.

1. 접근제어 규칙기술 언어 비교 및 분석

네트워크 상을 이동하는 메시지들의 접근을 제어하기 위해 Socks[6], Ip_filter[7], Ipfw[11], Ipfwadm[12], Ipfirewall[13], Cisco[14] 등과 같은 여러 가지 침입차단 프로그램들이 개발되었다. Ip_filter와 Ipfw, Ipfwadm, Ipfirewall, Cisco 등은 IP 계층을 위한 침입차단시스템이며, 그림 1은 각 프로그램에서 자신의 고유한 언어로 기술한 접근제어 규칙들을 기술하고 있다.

즉, Ip_filter는 BSD 계열 혹은 SUN 운용체제를 위한 침입차단시스템으로 규칙의 기술은 그림 1의 (가)와 같다. Ipfw는 Linux 혹은 FreeBSD 2.1.0을 위한 침입차단시스템으로서 '허용(accept)' 혹은 '거절(deny)'로 규칙을 정의한다 (그림 1의 (나) 참조). Ipfwadm은 Linux 커널의 접근제어 규칙을 관리하기 위한 프로그램으로 Ipfw보다 완벽하고 사용이 용이하다. 자세한 규칙은 Ipfwadm 논문[12]에 기술되어 있고, 예제는 그림 1의 (다)와 같다. Ipfwadm은 주어진 호스트가 하나 이상의 IP 주소를 가질 때, 필요로 하는 추가적인 규칙을 자동적으로 삽입시킨다. Ipfirewall 프로그램 역시 BSD 계열의 호스트의 IP 패킷 필터링 도구로서 대부분의 상업용 라우터에 의해 제공되는 기능과 유사한 기능을 제공한다. 규칙은 그림 1의 (라)와 같다. Cisco도 IP상에서의 보안을 위한 규칙들을 정의하여 사용하고 있다 (그림 1의 (마) 참조).

```
block in on le0 quick proto tcp from 10.1.1.1 to any flags S/SA
block in on le0 quick log body proto tcp from 10.1.1.1 to 10.1.1.254
block in on le0 quick log proto tcp from any to any port = 2049
pass in on le0 quick proto tcp from any to any
block in all
pass out all
```

(가) Ip__filter의 접근제어 규칙

```
ipfw addblocking deny iface 10.1.1.1 from 10.1.1.1 to 0/0
ipfw addblocking deny iface 10.1.1.1 from 10.1.1.1 to 10.1.1.254
ipfw addblocking deny iface 10.1.1.1 from 0/0 to 0/0 2049
ipfw addblocking accept iface 10.1.1.1 from 0/0 to 0/0
ipfw addblocking deny from 0/0 to 0/0
ipfw addforwarding accept from 0/0 to 0/0
```

(나) Ipfw의 접근제어 규칙

```
ipfwadm -p deny -B -I 10.1.1.1 -P tcp -S 10.1.1.1 -D 0.0.0.0 -y
ipfwadm -p deny -B -I 10.1.1.1 -k -P tcp -S 10.1.1.1 -D 10.1.1.254
ipfwadm -p deny -B -I 10.1.1.1 -k -P tcp -S 0.0.0.0 -D 0.0.0.0 2049
ipfwadm -p accept -B -I 10.1.1.1 -P tcp -S 0.0.0.0 -D 0.0.0.0
ipfwadm -B -p deny -S 0.0.0.0 -D 0.0.0.0
ipfwadm -F -p accept -S 0.0.0.0 -D 0.0.0.0
```

(다) Ipfwadm의 접근제어 규칙

```
ipfirewall addblocking reject le0 from 10.1.1.1/32 to 0/0
ipfirewall addblocking reject le0 from 10.1.1.1/32 10.1.1.254/32
ipfirewall addblocking reject le0 from 0/0 to 0/0 2049
ipfirewall addblocking accept le0 from 0/0 to 0/0
ipfirewall addblocking reject from 0/0 to 0/0
ipfirewall addforwarding accept from 0/0 to 0/0
```

(라) Ipfirewall의 접근제어 규칙

```
access-list 100 deny tcp 10.1.1.1 0.0.0.0 0.0.0.0 255.255.255.255
access-list 100 deny tcp 10.1.1.1 0.0.0.0 10.1.1.254 0.0.0.0
access-list 100 deny tcp 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255 eq 2049
access-list 100 accept tcp 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
access-list 100 deny ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
interface ethernet 0
ip access-group 101 in
ip access-group 151 out
access-list 150 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
ip access-group 100 in
ip access-group 150 out
```

(마) Cisco의 접근제어 규칙

```
permit 192.168.2.0 255.255.255.0 0.0.0.0 0.0.0.0
permit 147.6.39.54 255.255.255.255 192.168.2.0 255.255.255.0
deny 0.0.0.0 0.0.0.0
```

(바) Socks의 접근제어 규칙

그림 1 침입차단시스템들에서 사용하는 접근제어 규칙 (Access Control Rules Defined by Firewalls)

Socks4.3은 한 개의 포트(port)(1080)로 모든 tcp 프로토콜을 사용하는 응용 프로그램들에게 회선-단계(circuit-level)의 프락시 기능을 제공한다. Socks는 Socks 서버와 Socks 클라이언트 라이브러리로 구성되어 있다[6]. 아래 그림 2와 같이 Socks 서버는 응용 계층에 위치하는 반면, Socks 클라이언트 라이브러리는 클라이언트의 응용 계층과 전송(transport) 계층 사이에 놓인다. 이와 같은 Socks의 접근제어를 위한 규칙은 그림 1의 (바)와 같다.

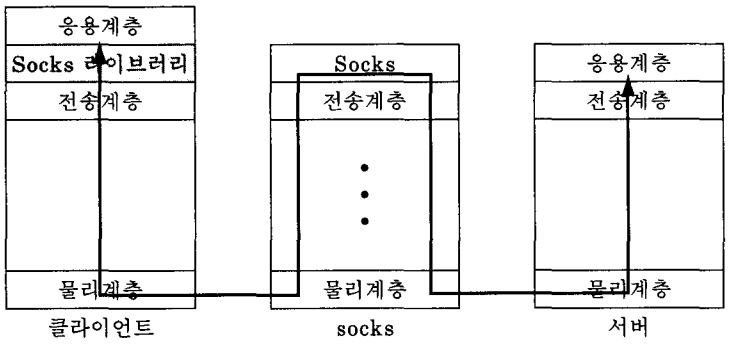


그림 2 네트워크 계층에서 SOCKS (SOCKS for a Network Layer)

위와 같이 침입차단시스템 프로그램들은 제각기 접근제어 규칙을 기술하는 방법들을 정의하여 사용하고 있기 때문에 네트워크 관리자들은 여러 개의 침입차단시스템 프로그램을 다룰 때 각각의 방법들에 대해서 잘 알고 있어야 하는 어려움을 갖는다.

2. 접근제어를 위한 통합된 규칙기술 언어

이전 절의 마지막 부분에서 언급했던 문제를 해결하기 위해서, 이 절에서는 통합된 규칙기술 언어를 제안한다. 우선, Reed[10]가 정의한 필터-언어(filter language)를 살펴본다.

Reed[10]가 제안하는 필터-언어는 패킷을 필터링하는 다양한 침입차단시스템들의 접근제어 규칙들을 생성한다. 필터-언어 컴파일러는 C 언어에서 사용하는 마크로 문장을 그대로 이용할 수 있고, 주석문을 표현할 때도 C 언어와 마찬가지로 "/* */"를 사용하거나 UNIX 쉘(shell) 스크립트처럼 "#"을 사용하게 한다. 필터-언어 컴파일러인 FLC(filter language compiler)가 처리할 수 있는 침입차단시스템 프로그램은 Ip_filter, Ipfw, Ipfwadm, Ipfirewall, Cisco, Screend이다.

그림 3은 필터언어 문법에 맞게 기술한 접근제어 규칙이며, FLC를 이용하여 그림 3의 접근제어 규칙을 컴파일할 때, 옵션으로 사용자가 원하는 침입차단시스템 (Ip_filter, Ipfw, Ipfwadm, IPfirewall, Cisco, Screend 중에 하나)을 설정하면, 사용자가 선택한 침입차단시스템의 접근제어 규칙 (그림 1의 (가)~(마) 중에서 하나)을 생성한다. 즉, 통합된 하나의 접근제어 규칙기술 언어로 다양한 침입차단시스템의 접근제어 규칙들을 기술할 수 있다.

하지만, 필터-언어를 처리하는 컴파일러에서는 침입차단시스템 프로그램들 중에서 Ip_filter의 경우, 구버전만을 지원하고 있고, IP 계층만을 고려하고 있다. 따라서, 본문에서 제안한 혼합형 침입차단시스템을 위한 접근제어 규칙을 기술하는데 한계점을 갖는다. 따라서, 본 논문에서 구현하고자 하는 혼합형 침입차단시스템의 접근제어 규칙을 기술하기 위해서는 필터-언어의 문법을 확장하여 최신의 Ip_filter 버전을 지원할 수 있어야 하며 응용계층의 대표적인 침입차단시스템들 중에 하나인 Socks를 위한 접근제어 규칙도 처리할 수 있도록 하여야 한다.

```
1    #define BAR foo
2    #define foo 10.1.1.1
3    #define bar 10.1.1.254
4    #if defined(____cisco____)
5    interface ethernet0
6    access-list 101
7    #endif
8    #if defined(____ipfilter____) || defined(____ipfirewall____) || defined(____ipfw21r____)
9    interface le0
10   #endif
11   #if defined(____ipfw____) || defined(____ipfwadm____)
12   interface 10.1.1.1
13   #endif
14   policy block in all
15   if ( in ) then {
16       set protocol tcp
17       if ( from host BAR and opening ) then {
```

```
18          block .
19          }
20          if ( from foo and to host bar ) then {
21          log body and block .
22          }
23          if ( to port 2049 ) then {
24          log and block .
25          }
26          pass .
27      }
28      end-policy
29      policy pass out all
30      end-policy
```

그림 3 필터-언어의 예제(An Example Script Defined by Filter Language)

본 논문에서 제안하는 접근제어 규칙기술 언어는 Reed[10]가 제안한 언어를 기반으로 확장되었고, 문법 체계를 BNF 형식으로 나타내면 다음 그림 4와 같다. 예를 들면, 규칙은 "policy" 혹은 "end-policy" "set" 문장, "if" 문장 등으로 시작될 수 있다. 만일 "policy"로 시작한 경우에는 그림 4의 2번 줄에 기술된 문장으로 확장된다. 즉, 허용(pass)이나 거절(deny)을 나타내는 키워드가 오게 되고, "in" "out"과 같은 방향에 대한 정보가 뒤따르게 되며, 마지막으로는 호스트에 대한 정보가 놓이게 된다.

제안하는 규칙기술 언어의 문법과 그림 1의 (가)~(마)에서 표현한 다른 침입차단시스템의 규칙기술 문법과 비교해 보면, 판독성(readability)은 높고 언어의 복잡도가 높지 않음을 알 수 있다. 즉, 접근제어 규칙을 기술하기 위해서 구조적 프로그래밍 기법에서 사용하는 선택 (if-then)과 순차 기법 ({, })을 사용한다. 또한, 다른 문법에서 표현되는 필수적인 기능들은 모두 포함하면서, 사용자들이 쉽게 규칙을 기술할 수 있고 판독성을 높이는 색인어(keyword)들을 사용한다.

```
1   rule ::= bpolicy | epolicy | set | interface | if | shell | rule
2   bpolicy ::= "policy" action inout pfromto
3   epolicy ::= "endpolicy"
4   set ::= "set" setopt
5   setopt ::= "netmask" nid | protocol | "identd" id | "userlist" ulist
6   interface ::= "interface" id
7   if ::= "if" "(" expr ")" "then" "{" statements "}"
8   shell ::= "shell" slist
9   action ::= log | "block" | "pass"
10  inout ::= "in" | "out"
11  pfromto ::= "all" | "from" host "to" host
12  log ::= "log" | "log body"
13  host ::= hid "/" nid [doport]
14  expr ::= subexpr | subexpr "and" expr
15  subexpr ::= protocol | fromto dohost | fromto donet | fromto doport | "on" id | inout | tcpbits | with
16  statements ::= doact "." | statement | statement statements
17  statement ::= set | interface | if
```

```
18  protocol ::= "protocol" id
19  fromto ::= "from" | "to"
20  dohost ::= hid | "host" hid
21  donet ::= nid | "net" nid "mask" nid | "net" nid " | "netmask" nid
22  doport ::= "port" port
23  tcpbits ::= "established" | "opening" | "flags" id
24  with ::= "with" woption
25  doact ::= action | action and doact
26  port ::= portcmp id | id portout id | id portin id
27  woption ::= "frag" | "short" | "ipopts" | extra
28  portout ::= "><"
29  portin ::= "<>"
30  extra ::= "opt" eopt
31  portcmp ::= "eq" | "ne" | "ge" | "gt" | "le" | "lt" | "=" | "!=" | ">=" | ">" | "<=" | "<"
32  hid ::= nid | id "." id "." id "." id
33  nid ::= digits | digits "." digits "." digits "." Digits
34  id ::= [digits | letters] +
35  ulist ::= id | id,ulist
36  slist ::= string | string " " slist
37  digits ::= [0-9]+
38  letters ::= [a-zA-Z]+
39  string ::= [digits | letters | schar]+
40  schar ::= "!" | "$" | "@" | "#" | "%" | "^" | "&" | "*" | "?"
```

그림 4 통합 규칙기술 언어의 BNF (BNF of the Integrated Rule-Script Language)

3. 통합된 규칙기술 언어 컴파일러

이 절에서는 제안한 규칙기술 언어를 처리하기 위한 컴파일러를 기술한다. 컴파일러는 다음 그림 5와 같은 구조를 갖는다. 언어 처리 절차는 다음과 같다.

【절차 1】 사용자는 시스템에서 제공하는 그래픽 사용자 인터페이스를 이용하여 규칙을 입력한다.

【절차 2】 사용자가 접근제어 규칙을 모두 입력한 후, 접근제어 규칙 생성기가 그림 3과 같은 통합 규칙기술 언어로 표현된 규칙을 생성한 후, 파일에 저장한다.

【절차 3】 사용자가 원하는 특정 네트워크 계층의 침입차단시스템을 지정하면, 해당 침입차단시스템에 맞는 규칙을 생성하는 생성기는 최종적으로 사용자가 요구한 접근제어 규칙을 생성하게 된다.

【절차 4】 사용자가 요구한 접근제어 규칙을 생성할 때, 규칙 오류 탐색기에 의해 기술된 규칙에 구문적인 오류가 존재하는지의 여부가 확인되고, 만일 오류가 존재하면, 사용자 인터페이스를 통해서 결과가 전달되어진다.

그림 5와 같이 규칙기술 언어 처리기의 각 요소들은 다음과 같은 기능을 수행한다. 통합 규칙기술 언어 생성기는 사용자가 그래픽 사용자 인터페이스로부터 입력한 내용을 입력으로 제안한 규칙기술 언

어 문법에 맞는 통합 접근제어 규칙을 생성한 후, 파일로 저장한다 (【절차 2】). 특정 침입차단시스템에 대응되는 규칙 생성기는 확장된 FLC로써 【절차 2】의 출력을 입력으로 받아드려 사용자가 원하는 네트워크 계층의 특정 침입차단시스템의 접근제어 규칙을 생성한 후,

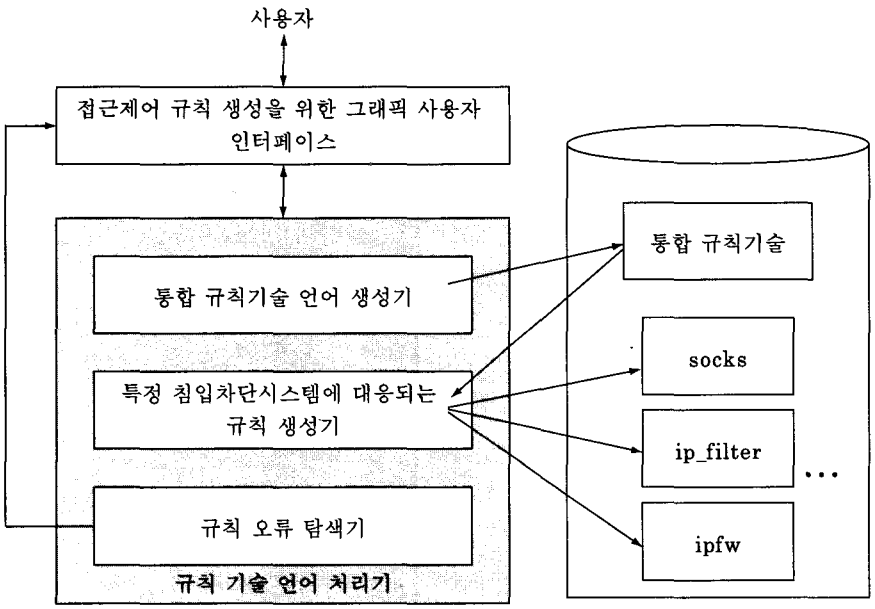


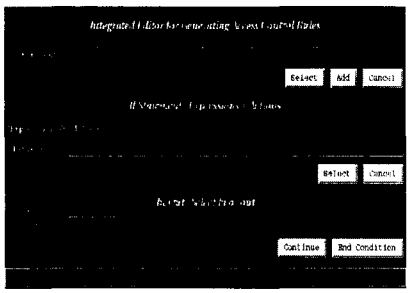
그림 5 접근제어 규칙을 생성하기 위한 처리기 (Processor for Generating Access Control Rules)

파일로 저장한다 (【절차 3】). 규칙 오류 탐색기는 특정 접근제어 규칙을 생성하는 과정에서 사용자가 규칙을 올바르게 설정하지 못한 경우, 규칙 오류를 탐지하며 만일, 오류가 있는 경우, 이를 다시 사용자에게 알려주는 기능을 수행한다 (【절차 4】).

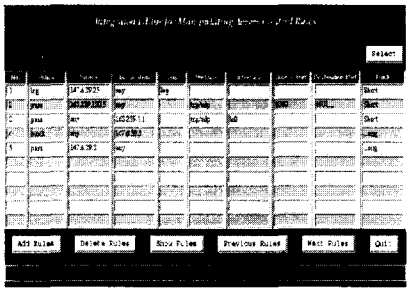
4. 통합 규칙기술 언어의 그래픽 사용자 인터페이스

이 절에서는 접근제어 규칙의 생성 및 삭제, 출력과 같은 관리 기능을 수행하는 그래픽 사용자 인터페이스를 구현한 내용을 기술한다. 하나의 규칙은 다음 그림 6의 (가)와 같이 허용 혹은 거절과 같은 보안정책과 출발지 및 도착지 호스트 명, 트래픽 로그 여부, 프로토콜 명 등과 같은 정보로 이루어져 있다. 화면의 구성은 다음과 같다. 사용자는 어떤 네트워크 계층을 위한 침입차단시스템 접근제어 규칙을 설정할 것인지를 결정한다. 네트워크 계층이 결정되면, 그에 대응되는 규칙들이 동시에 10 개씩 화면에 출력된다. 규칙이 10 개 이상인 경우는 "Next Rules" 버튼과 "Prev Rules" 버튼을 이용하여 다음 10 개의 규칙 혹은 이전 10 개의 규칙을 출력하는 화면으로 이동한다.

그림 6의 (가)에서 "Add Rules"라는 메뉴 버튼을 선택하면, 그림 6의 (나)와 같이 사용자가 규칙을 기술하기 위해서 선택할 수 있는 문장들이 출력되고, 추가하고 싶은 문장을 선택하면 된다. "Delete Rules"는 삭제하기를 원하는 규칙의 번호를 입력하여 규칙을 삭제할 수 있다. "Show Rules"은 그래픽 사용자 인터페이스를 통해 생성한 규칙이 실제로 어떻게 표현되었는지 제안한 규칙기술 언어의 형태로 출력한다. 이 때, 텍스트로 표현된 규칙들을 직접 수정할 수 있고, 수정된 내용은 다시 그림 6의 (가) 화면에 반영이 되어진다.



(가) 접근제어 규칙들을 출력한 화면



(나) 새로운 규칙을 삽입하는 화면

그림 6 접근제어 규칙을 제어하는 화면들의 예
(Example Screens for Controling Access Control Rules)

위와 같은 사용자 인터페이스를 통해 사용자의 입력을 받아들이면, 규칙 생성기는 다음 그림 7과 같은 형태의 통합 규칙기술 언어로 되어 있는 파일을 생성하게 되고, 마지막으로 사용자가 지정한 네트워크 계층을 위한 특정 규칙 파일이 갱신되어 다시 화면에 출력이 된다.

Ⅳ. 혼합형 침입차단시스템 관리를 위한 그래픽 사용자 인터페이스

본 논문에서 구현한 혼합형 침입차단시스템을 위한 그래픽 사용자 인터페이스는 접근제어 규칙 관리 및 가상 사설망(VPN) 기능, 네트워크 주소 변환(NAT)[11] 기능, 실시간 트래픽 감시 기능, 로그 관리 및 통계 처리 기능을 제공하고 있다. 모든 기능들을 위한 사용자 인터페이스는 Tcl/Tk[8][9]로 구현되었다. 이 장에서는 구현한 기능들을 차례대로 기술한다.

```
1   #!/bin/csh -f
2   if ( in and protocol tcp and interface le0 and from any and to net 147.6.39.4
3       mask 32 ) then {
4       log .
5   }
6
7   if ( out and from any and to net 147.6.39.4 mask 32 ) then {
8       pass .
9   }
10
11  policy block in all
12  end-policy
```

그림 7 통합 규칙기술 언어로 된 스크립트 예제
(An Example Script written in the Integrated Rule Script Language)

1. 파일 구조

본 논문에서 제안하는 그래픽 사용자 인터페이스를 통해 관리되는 파일들은 접근제어 규칙 파일과 로그 파일, 네트워크 주소변환 파일, 라우팅(routing)에 관한 정보를 저장하고 있는 파일들이다. 일반적으로, 파일의 내용은 직접 판독하기에는 이해의 어려움이 따르기 때문에 관리자의 부담을 가중시킨다. 따라서, 파일의

내용을 이해하기 쉽도록 그림 8과 같은 형식으 로 파일을 요약하여 화면에 출력한다.

보안 정책	출발지 주소	도착지 주소	포로토콜	인터페이스	포트(port)정보

(가) 접근제어 규칙 파일

보안 정책	서버명	도착지포트	사용자 목록	쉘 명령어

(나) 가상 사설망 규칙 파일

월	일	시간	데몬의 이름	로그에 대한 설명

(다) 로그 파일

보안 정책	인터페이스	출발지 주소	도착지 주소	프로토콜	출발지 포트	도착지 포트

(라) 네트워크 주소 변환 파일

네트워크 인터페이스	도착지 주소	네트워크 마스크(mask)

(마) 라이팅 파일

그림 8 파일의 내용 (Content of Files)

2. 네트워크 감시

감시(monitoring) 기능은 네트워크를 감시하기 위한 기능으로써 실시간 트래픽 감시와 패킷 정보 출력 기능, 네트워크 게이트웨이 정보를 제공한다. 그림 9와 같이 실시간 트래픽 감시 기능은 실시간적으로 침입차단시스템으로 들어오는 트래픽과 나가는 트래픽 정보를 Swatch[15]라는 프로세스를 이용하여 검출해서 포트번호와 입력 및 출력 방향 등과 함께 출력한다. Swatch는 지정된 로그 파일에서 검출을 원하는 문자열이 발견되면, 사용자에게 경고 메시지를 실시간적으로 전송한다. 그 밖에 호출기 호출과 같은 기능도 수행할 수 있다.

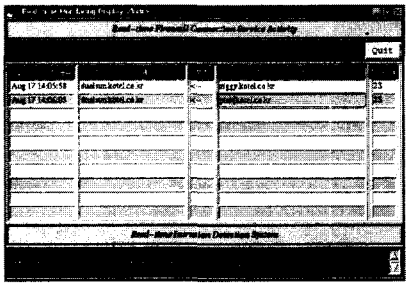
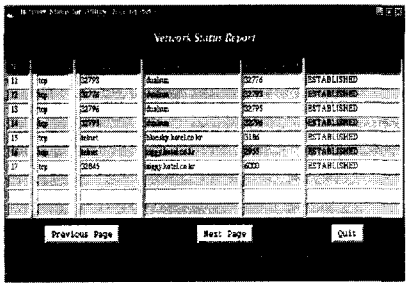
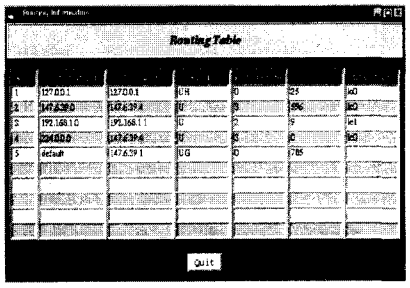


그림 9 Swatch를 이용한 실시간 트래픽 감시 화면의 예 (An Example Screen for Monitoring Real-Time Traffics with Swatch)

한편, 네트워크 상태 감시 기능은 현재 시스템에 연결되어 있는 네트워크 개체들의 목록을 그림 10의 (가)와 같이 출력하고, 라우터 감시 기능 그림 10의 (나)와 같이 라우터 테이블 정보를 출력한다.



(가) 네트워크 상태 보고 화면



(나) 라우팅 정보 테이블

그림 10 감시를 위한 화면들 (Screens for Monitoring)

3. 통계 처리

통계처리 기능에는 표 1과 같이 일별 및 주간별, 월별로 외부로부터 얼마만큼의 접속이 있었는지 접속회수와 얼마만큼의 데이터들이 전송되었는지를 나타내는 데이터 전송량을 시간별 및 날짜별로 구분하여 출력한다. 즉, 사용자가 출력해 보고자 하는 날짜를 지정하면, 다음 그림 11과 같이 그래프 형식으로 출력이 된다. 그림 11은 7월 27일부터 한 주간 접속된 통계치를 시간별로 나타내고 있다. 즉, 그래프에서 의미하는 내용은 주로 오전 10시와 오후 2시, 7시, 9시에 연결이 많았음을 알 수 있다.

날짜 / 통계치	일별	주간별	월별
접속회수	① 시간대별	① 시간대별 ② 날 짜 별	① 시간대별 ② 날 짜 별
데이터량	① 시간대별	① 시간대별 ② 날 짜 별	① 시간대별 ② 날 짜 별

표 1 통계자료 기준

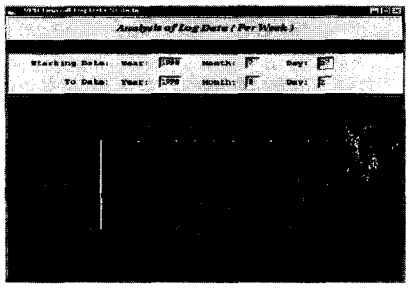


그림 11 통계 자료 화면의 예
(An Example Screen for Statistical Information)

4. 도움말

관리자의 사용상 편이함을 제공하기 위하여 각 화면의 아래부분에는 간단한 도움말을 출력하며 전체적인 도움말을 가장 상위 메뉴에서 제공하고 있다.

V. 결 론

침입차단시스템 구축 시에 접근제어 규칙을 정확히 기술하는 것은 매우 중요한 문제이다. 하지만, 어떤 침입차단시스템을 사용하느냐에 따라서, 상이한 규칙기술 언어들을 제공하고 있기 있기 때문에 관리자에게 어려움을 주고 있다. 만일, 하나의 언어로 다양한 침입차단시스템의 접근제어 규칙들을 생성한다면, 관리의 용이성을 제공할 수 있다. 본 논문에서는 여러 가지 침입차단시스템에서 제공하는 접근제어 규칙들을 생성하기 위해서 단일한 형태의 접근제어 규칙기술 언어를 제안하였고, 관리자의 편의성을 도모하기 위해서, Tcl/Tk를 이용하여 그래픽 사용자 인터페이스를 구현하였다. 또한, 네트워크 관리를 위해서, 로그 관리 및 실시간 트래픽 감시, 날짜별 통계처리 기능을 제공하였다.

현재, 보다 진보된 보안 서비스를 제공하기 위해서 사용자 인증 및 관리를 위한 기능을 개발중에 있으며, 향 후 동적 패킷 필터링과 같은 향상된 침입차단시스템 관리 기능을 추가할 예정이다.

참고문헌

[1] M. J. Ranum, "Thinking about Firewalls," Proceedings of the International Conference on Systems and Network Security and Management, 1993.

[2] W. R. Cheswick and S. M. Bellovin, Firewalls and Internet Security, Addison-Wesley, 1994.

[3] CheckPoint Software Technologies Ltd., FireWall-1 User Guide, Version2.1, June 1996.

[4] Raptor Systems, The Eagle Firewall Technical White Paper, http://www.raptor.com/whitepaper/ title.html, 1997.

[5] Secure Computing Corporation, Borderware Firewall Server 4.0 White Paper, http://www.sctc. com/borderware/white.html, November 1996.

[6] Y. Lee, Socks: A Protocol for TCP Proxy across Firewalls, http://www.socks.nec.com, 1996

[7] D. Reed, IP Filter, http://cheops.anu.edu.au/~avalon/ip-filter.html, 1995.

[8] B. B. Welch, Practical Programming in Tcl and Tk, 2nd Ed., Prentice Hall, 1997.

[9] J. K. Ousterhout, Tcl and the Tk Toolkit, Addison-Wesley, 1994.

[10] D. Reed, Filter Language Compiler, http://coombs.anu.edu.au/~avalon/flc.html, 1995.

[11] The FreeBSD Documentation Project, FreeBSD Handbook, htttp://x.physics.usyd.edu/FreeBSD/handbook/handbook.html, 1997.

[12] J. Vos and W. Konijnenberg, IPFWADM:Linux Firewall Facilities for Kernel-Level Packet Screening, http://www.ase.ee/moshkow/koi/SECURITY/ipfwadm/paper.txt, 1996.

[13] Boulet Fermat Associates, IPFIREWALL: An Internet Firewall Security Tool, http://www. ipfire-wall.com, 1995.

[14] Cisco, Increasing Security on IP Networks, http://www.cisco.com/univercd/cc/td/doc/cisintwk/ics/icssec ur. htm, 1998.

[15] S. E. Hansen and E. T. Atkins, "Automated System Monitoring and Notification with Swatch," Proceedings of LISA, 1993.

□ 著者紹介

박 찬 정

1988년 2월 서강대학교 전자계산학과 졸업(공학사)
1990년 2월 한국과학기술원 전산학과 졸업(공학석사)
1998년 2월 서강대학교 전자계산학과 졸업(공학박사)
1990년 ~ 현재 한국통신 멀티미디어연구소 근무중

※ 주관심분야 : 다단계보안데이타베이스, 침입차단 및 침입탐지 시스템, 전자상거래