멀티 플랫폼 기반의 네트워크 패킷 스캐너 설계 및 구현

Design and Implementation of a Network Packet Scanner based on Multi-Platform

이우인*, 양해술**
서울벤처정보대학원대학교 컴퓨터응용기술학과*, 호서대학교 벤처전문대학원**

Woo-In Lee(wins154@hanmail.net)*, Hae-Sool Yang(hsyang@hoseo.edu)**

요약

최근의 해킹 추세는 기업의 이윤과 관계되는 모든 IT 인프라를 대상으로 하고 있다. 이는 기존의 공격이 비서비스 포트를 통한 서비스 인프라로의 접근을 시도했다면, 현재는 기업이윤 창출의 원천인 서비스 자체를 공격하고 있다. 즉, 서비스에 직접적인 영향을 끼치고 있음에도 불구하고 기존의 보안 솔루션이나 체계로는 방어하기가 어렵고 동시에 이용자 보호, 지속 가능한 경영 자체를 위협하며 금전을 요구하는 협박의 형태로 점점 더 많은 기업에 위해를 가하고 있다. 본 논문에서는 트래픽량을 기준으로 정상, 비정상을 판단하는 예외처리기반 네트워크 침입탐지 시스템을 대상으로 멀티 플랫폼 기반의 네트워크 패킷 스캐너를 설계하고 구현하였다. Linux나 unix 환경에는 ngrep , snort, TCPdump와 같은 여러 가지 네트워크 침입탐지와 패킷 관리 도구들이 있지만 대부분 문자 방식 사용자 인터페이스(CUI : Character based User Interface)를 기반으로 구현되어 익숙하지 않은 사용자들에게는 불편함이 따른다. 제안된 시스템은 이러한 불편함을 개선하여 사용자에게 직관적이고 사용이 쉬운 인터페이스를 제공하기 위하여 그래픽 사용자 인터페이스(GUI : Graphical User Interface)기반으로 구현하였고, 모든 운영체제에서 구동될 수 있도록 멀티 플랫폼을 지원하는 Qt(C++)언어를 사용하여 설계 및 구현하였다.

■ 중심어 : | 패킷 스캐너 | 패킷관리 툴 | 정보시스템 관리 |

Abstract

The recent trend of the hacking deals with all the IT infrastructure related to the profit of the companies. Presently, they attack the service itself, the source of the profit, while they tried to access to the service infrastructure through the non-service port in the past. Although they affect the service directly, it is difficult to block them with the old security solution or the old system and they threaten more and more companies with the demand of money menacing the protection of customers and the sustainable management. This paper aims to design and implement multi-platform network packet scanner targeting the exception handling network intrusion detection system which determines normal, abnormal by traffic.

Linux and unix have the various network intrusion detection and packet management tools like ngrep, snort, TCPdump, but most of them are based on CUI (Character based User Interface) giving users discomfort who are not used to it. The proposed system is implemented based on GUI(Graphical User Interface) to support the intuitive and easy-to-use interface to users, and using Qt(c++) language that supports multi-platform to run on any operating system.

■ keyword : | Packet Scanner | Packet Management Tool | Information System Management |

접수번호 : #100126-004
접수일자 : 2010년 01월 26일
심사완료일 : 2010년 03월 10일
교신저자 : 이우인, e-mail : wins154@hanmail.net

I. 서 론

21세기 정보화 사회에서 우리나라는 인터넷 보급률과 이용률에서 세계 최고 수준의 정보통신 인프라를 갖추고 있다. 그러나 인프라 수준에 비하여 사용자 개개인의 정보보호에 대한 인식은 미비한 실정이고, 기업에서조차 물리적인 보안의식에 치중하여 각종 정보보호 침해해사고가 발생하고 있다.

국가나 지역에 관계없이 정보를 인터넷을 통해 쉽게 수집할 수 있고, 관리되는 정보의 대부분이 서버에서 집중화되어 있어 웹(web)이나 시스템관련 각종 해킹기법이 발달하게 되었다. 소프트웨어 및 하드웨어 자원이 발달 하면서 보안기술과 해킹기법의 발전이 동시에 이루어지기 때문에 컴퓨터의 전문성 측면에서 이에 대한 응용력이 부족한 일반 사용자들은 각종 해킹으로부터 피해를 입는 주요 대상층이 되고 있다[7].

최근의 해킹 추세는 기업의 이윤과 관계되는 모든 IT 인프라를 대상으로 하고 있다. 이는 기존의 공격이 비서비스 포트를 통한 서비스 인프라로의 접근을 시도했다면, 현재는 기업이윤 창출의 원천인 서비스 자체를 공격하고 있다. 즉, 서비스에 직접적인 영향을 끼치고 있음에도 불구하고 기존의 보안 솔루션이나 체계로는 방어하기가 어렵고 동시에 이용자 보호, 지속 가능한 경영 자체를 위협하며 금전을 요구하는 협박의 형태로 점점 더 많은 기업에 위해를 가하고 있다.

대표적인 네트워크 공격으로는 DDos, SYN-Flooding, Smurf, Land Attack , ARP spoofing, 정찰스캔(reconnaissance) 등이 있으며, 이들의 공격 특징으로는 모두 과다한 트래픽을 발생시킨다는 것이다[1][7]. 본 논문에서는 트래픽량을 기준으로 정상, 비정상을 판단하는 예외처리기반 네트워크 침입탐지 시스템을 대상으로 멀티 플랫폼 기반의 네트워크 패킷 스캐너(MNPS : Multi-platform Network Packet Scanner)를 설계하고 구현하는 것을 목적으로 한다.

멀티 플랫폼 기반의 네트워크 패킷 스캐너는 기존의 네트워크 침입탐지 기능에 부가하여 로컬 세그먼트 안에 있는 모든 호스트와 특정 포트에 대한 스캔 기능을 지원한다. 만일 중요한 서비스를 지원하고 있는 서버나 서버의 포트가 내,외부적인 요인으로 정상적인 서비스가 중지되면 서비스의 중요도에 따라 심각한 피해가 발생할 수 있기 때문에 제안시스템에 관리 대상으로 분류하여 스캔 기능을 갖추도록 설계하였으며, 또한 linux나 unix 환경에는 ngrep , snort, TCPdump와 같은 여러 가지 네트워크 침입탐지와 패킷 관리 도구들이 있지만 대부분 문자 방식 사용자 인터페이스(CUI : Character based User Interface)를 기반으로 구현되어 익숙하지 않은 사용자들에게는 불편함이 따른다. 제안시스템은 이러한 불편함을 개선하여 사용자에게 직관적이고 사용이 쉬운 인터페이스를 제공하기 위하여 그래픽 사용자 인터페이스(GUI : Graphical User Interface)기반으로 구현하고, 모든 운영체제에서 구동될 수 있도록 멀티 플랫폼을 지원하는 Qt(C++)언어를 사용하여 설계 및 구현하였다.

II. 네트워크 공격의 유형

네트워크 기반에서 시도되는 최근 공격 유형들은 시스템과 네트워크의 정상적인 동작을 방해하는 간접적 형태의 공격들이다. 이러한 네트워크 공격의 유형들은 단일 기술을 바탕으로 설계되기도 하지만 대부분은 공격의 목적을 달성하기 위하여 유형들 간에 공격의 필요성에 따라 상호 결합하여 설계된다.

네트워크 공격을 위해 사용하는 각종 해킹사고의 유형은 [그림 1]과 같다[7].

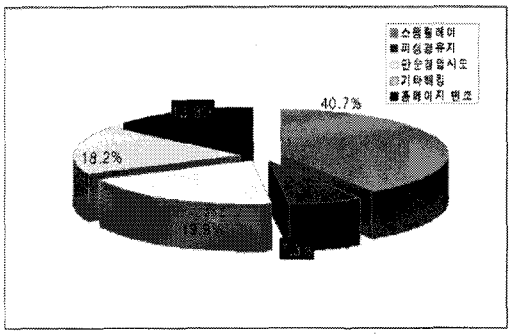


그림 1. 해킹사고 접수처리 유형별 분류

스팸릴레이는 타 시스템을 스팸 메일발송에 악용한 공격이며, 피싱경유지는 보안이 취약한 국내 시스템이 해외 위장사이트로 악용된 경우이며, 단순침입시도는 인터넷상에 연결된 시스템의 취약점을 찾기 위하여 네트워크 서비스를 파악해보는 공격으로 주로 자동화된 해킹도구나 웜바이러스에 의한 감염 시도 트래픽이 포함된 공격유형이다. 또한 기타해킹은 원격터미널접속, 웹 서비스 대상 해킹시도, 의도적인 스캔공격과 같은 공격유형이고 홈페이지변조는 특정목적을 위하여 홈페이지를 변조하여 이용자에게 서비스 이용의 불편을 주는 공격이다. 대표적인 네트워크 공격의 유형별 특징은 [표 1]과 같다.

표 1. 네트워크 공격의 유형별 특징

네트워크 공격유형	사용 Protocol	특 징
DDoS	UDP, ICMP	대역폭(bandwidth) 고갈
	TCP	연결(connection) 고갈
	HTTP	연결(connection) 고갈
SYN-Flooding	TCP	false-alarm 발생, 패킷 손실유발
Smurf	ICMP	대역폭(bandwidth) 고갈
Land attack	ICMP	시스템 과부하(overload)
ARP Spoofing	ARP	악성코드 배포
정찰스캔 (reconnaissance)	TCP/UDP ICMP	취약점 정보수집

III. 설계 및 구현관련 연구

1. 개발언어(Qt)

Qt는 크로스 플랫폼 애플리케이션으로 UI 개발을 위한 프레임워크로 사용된다. 여기에는 크로스 플랫폼 클래스 라이브러리, 코딩-디버그-컴파일-배포 등 프로그램 개발에 관련된 모든 작업을 하나의 프로그램 안에서 처리하는 환경을 제공하는 통합 개발 환경(IDE : Integrated Development Environment)이 제공되어 제안 시스템을 구현하는데 사용한다. Qt를 사용하면 애플리케이션을 작성한 후에 소스 코드를 다시 작성하지 않으면서 많은 데스크탑과 임베디드 운영 체제에 이를 배포할 수 있는 장점이 있다.

[그림 2]는 Qt의 클래스 라이브러리(library)와 개발도구, 제공 되어지는 플랫폼의 구성도이다[13].

1.1 Qt modular class library

Qt는 데스크 탑 및 임베디드 플랫폼 상의 GUI 애플리케이션 개발에 필요한 기능을 제공하기 때문에 각 플랫폼 고유의 그래픽 API를 사용함으로써 제안 시스템의 GUI 설계시 프레임웍으로 사용한다.

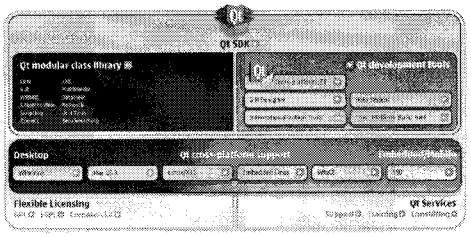


그림 2. Qt의 구성도

1.2 Qt development tools

모든 구현이 C++ 클래스로 캡슐화되어 있어 기존의 C에 비해 사용이 쉬우며, Qt의 API들은 MS Windows의 MFC와 유사하고, 상속을 사용한 기능의 확장이 편리해서 생산성이 높다. 다양한 ISO 변환 규격과 지역화를 위하여 16비트인 유니코드가 지원되어 어떤 언어로 번역된 메세지 파일만 존재하면 해당 언어로 작성된 애플리케이션을 쉽게 개발할 수 있다. 또한 기존의 모티프에서 사용하는 콜백(callback) 방식이 아닌 시그널/슬롯 방식으로 사용자의 이벤트를 처리하기 때문에 한 객체의 시그널과 다른 객체의 슬롯을 서로 연결함으로써 객체의 동작을 정의하여 제안 시스템이 유연성을 갖추도록 구현한다[3][13].

1.3 Qt cross-platform support

기존의 툴킷이 하나의 플랫폼에서만 작동하는 애플리케이션을 개발할 수 있는 것과는 다르게 Qt는 개발한 애플리케이션을 다른 플랫폼으로 포팅하는 것이 자유롭다. 즉, Qt로 작성한 기존 애플리케이션의 소스 코드를 단지 해당 플랫폼에서 다시 컴파일하는 것으로 Windows, Unix, Linux, Mac OS X 환경에서 실행시킬

수 있다. 제안 시스템이 멀티 플랫폼에서 동작되도록 구현하는 성능을 제공하는 Qt의 기능이다[14].

2. Libpcap library

패킷을 캡처하기 위한 도구로는 BPF(Berkeley Packet Filter), DLPI, NIT, SNOOP, SNIT, SOCK_PACKET, LSF(Linux Socket Filter), drain등 각 운영체제별로 다양한 도구가 있으며, libpcap은 이러한 모든 도구들을 수용하는 유연성을 갖추고 있는 API(Application Programming Interface)이다.

제안시스템은 운영체제에 관계없이 멀티 플랫폼을 지원하는 패킷 스캐너를 구현하는 것을 목적으로하기 때문에 운영체제에 상관없이 범용적으로 사용가능한 API를 제공해주고 공용프로그램 혹은 공용라이브러리의 제작이 가능한 libpcap 라이브러리를 사용한다. 현재 상용화된 네트워크기반 침입탐지시스템 제품의 상당수가 패킷분석을 위해서 libpcap을 사용하고 있다[6].

2.1 패킷 캡처(packet capture)

패킷 캡처는 네트워크상에서 돌아다니는 패킷을 들여다보는 것으로 사용하는 호스트가 포함된 네트워크를 관리하는 라우터(router)가 일반적인 라우터라면, 내부로 향하는 모든 패킷은 브로드캐스팅(broadcasting)된다. 이는 스위칭 라우터가 아닌 경우라면 모든 로컬(local) 네트워크의 패킷을 들여다 볼 수 있음을 의미한다. 운영체제는 자신에게 도착된 패킷 중에서 목적지가 자신인 패킷만을 선별처리해서 응용계층(application layer)까지 전송한다.

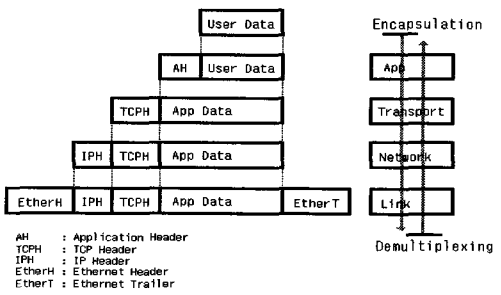


그림 3. encapuslation과 demultiplexing

[그림 3]은 libpcap이 패킷의 캡처를 하는 과정에서 송신시의 캡슐화(encapuslation) 과정과 수신시의 역다중화(demultiplexing) 과정이다[9].

2.2 libpcap의 사용 API

패킷 캡처는 여러 가지 목적으로 사용될 수 있다. 네트워크 침입탐지를 위한 프로그램이 가장 대표적인 응용이며, 네트워크 트래픽 감시(traffic monitoring), 네트워크 디버깅(network debugging)을 위한 용도로도 사용된다. 제안시스템을 구현하는데 필수적으로 사용하는 libpcap의 API는 [표 2]와 같다.

표 2. Libpcap의 사용 API

libpcapAPI	API format	programming interface
device& network	int pcap_lookupnet()	네트워크 & mask 번호
	char* pcap_lookupdev	네트워크 디바이스 포인터
	int pcap_datalink	링크 레이어 유형
packet capture initial	pcatp_t *pcap_open_live	패킷 캡처 디스크립터 선언
	pcap_t *pcap_open_offline	패킷 캡처 읽기
packet capture	u_char *pcap_next	패킷 포인터 리턴
	pcap_loop & pcap_dispatch	패킷 캡처 회수지정
packet filtering	pcap_compile	입력 패킷 필터링 지정
	pcap_setfilter	지정 필터 적용

3. Libnet

Libnet은 다양한 플랫폼을 경유하여 패킷을 만들고, 보낼 수 있는 고수준의 인터페이스를 제공하는 C 라이브러리로 다양한 보완적 기능뿐만 아니라 IP-layer와 Link-layer에서의 이동식 패킷형성 인터페이스를 갖추고 있다. libnet을 이용하여 패킷을 생성할 때 자료는 패킷 헤더에 들어갈 정보만 필요하며 실제 버퍼에 채우고 소켓(socket)에 써주는 작업은 libnet 라이브러리가 자동적으로 실행한다. 버퍼에 패킷 헤더를 생성하기 위해서는 순서에 관계없이 IP헤더와 TCP, UDP, ICMP, ARP헤더를 선택하여 원하는 패킷을 만들 수 있다. 또한 libnet 라이브러리 함수 중에는 체크섬(check_sum) 계산만을 별도로 해주는 함수가 존재하기 때문에 반복해서 체크섬 계산을 수행함으로 발생하는 프로그램이

나 실행시간의 과부하를 줄일 수 있다[8][9].

3.1 Libnet의 운용환경

Libnet은 네트워크에 패킷을 생성하고 전송할 수 있으며, 패킷 전송은 두가지 타입(IP-layer and Link-layer)을 갖는다. 패킷 캡처에 대한 기능이 없기 때문에 이 기능은 libpcap 라이브러리를 사용한다. 따라서 제안시스템은 libnet을 이용하여 네트워크에 패킷을 생성하고 전송하며 libpcap을 이용하여 패킷 캡처를 수행하도록 설계 및 구현한다. libnet의 운용환경은 OpenBSD, FreeBSD, NetBSD, BSD/OS, BSDi, LINUX, Solaris, IRIX, MacOS에 제한된다.

3.2 Libnet의 구조

네트워크에 패킷을 생성하고 전송하기 위하여 libnet이 사용하는 구조는IP 헤더 20바이트와 TCP 헤더 20바이트 그리고 Ethernet 14바이트의 고정길이를 갖는다.

3.3 Libnet의 패킷 생성 및 전송의 표준절차

Libnet이 임의의 네트워크 패킷을 생성하고 전송하기 위한 표준 절차는 [표 3]과 같다.

표 3. libnet의 패킷 생성 및 전송 표준 절차

절차	처리 내용	layer 구분	사 용 함 수
Memory Initialization	패킷 메모리 할당	IP	libnet_init_ packet() libnet_destroy_ packet()
		Link	
Network Initialization	네트워크 전송 인터페이스 확보	IP	libnet_open_raw_ sock()
		Link	libnet_open_link_ interface()
Packet Construction	패킷 생성	IP	libnet_build_ip(), libnet_build_tcp()
		Link	libnet_build_ ethernet()
Packet Checksums	Checksum 계산	IP	libnet_do_ checksum()
		Link	
Packet Injection	네트워크 패킷 전송	IP	libnet_write_ ip()
		Link	libnet_write_ link_layer()

Ⅳ. 패킷 스캐너 설계 및 구현

[그림 4]는 제안시스템의 전체구성도이다. 멀티 플랫폼 기반의 네트워크 패킷 스캐너는 내부 호스트들의 동작 여부를 확인하기 위해 ARP 호스트 스캔을 수행 할 수 있게 설계되었으며, 정상적인 서비스가 중단되면 정보시스템의 기능이 마비되는 중요 서비스 포트(http, ftp, telnet 등)의 모니터링이 필요한 경우 실시간 포트(port) 스캔을 통해 서비스 동작 여부를 확인할 수 있는 기능을 갖추도록 설계하였다. 세그먼트 안에 일어나는 모든 패킷의 활동들을 분석하고 저장하기 위해 포트 미러링(port mirroring) 기술을 사용하였으며, 데이터베이스에 데이터를 기반으로 사용자에게 GUI 환경의 뷰어(viewer) 기능을 제공하고, 네트워크 침입탐지에 필요한 통계자료도 데이터베이스 데이터를 가공하여 사용되도록 설계 및 구현하였다.

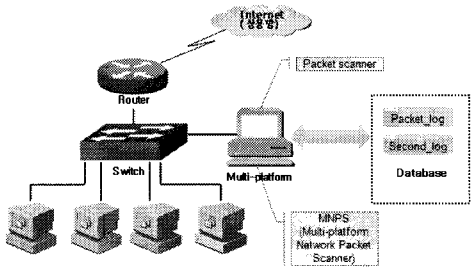


그림 4. 패킷 스캐너의 구성도

1. 데이터베이스 설계 및 구현

Mysql은 표준 데이터베이스 질의 언어인 SQL(Structured Query Language)을 사용하는 관계형 데이터베이스 관리시스템이다. 기본적으로 Unix, Linux, Windows 운영체제 등에서 사용할 수 있으며 매우 빠르고 유연하며 사용하기 쉬운 특징이 있다. 기능적으로 다중 쓰레드(multi thread)를 지원하며 C, C++, JAVA, Perl, PHP, Pyton Script 등을 위한 응용프로그램 인터페이스를 제공한다. 즉 리눅스 운영체제와 아파치 서버(apache server) 프로그램, mysql, php의 구성은 상호연동이 유연하기 때문에 웹 개발 환경에 많이 이용되고 있다. 제안 시스템은 멀티 플랫폼을 기반으로 하

기 때문에 다양한 운영체제에 이식성을 갖춘 mysql을 패킷 로그에 관련된 데이터를 관리하는 DBMS로 사용한다.

1.1 Table 설계 및 구현

PACKET_LOG 테이블은 원시 데이터를 가공하고 실시간으로 PACKET_LOG 테이블에 수정한다.SECOND_LOG 테이블은 초단위(second)로 트래픽량을 측정하고 가공하며 SECOND_LOG 테이블에 수정한다. [그림 5]는 제안시스템에서 사용하는 PACKET_LOG와 SECOND_LOG 테이블의 설계화면이다.

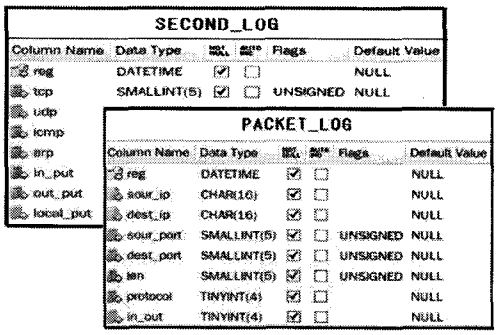


그림 5. 제안 시스템의 테이블 설계

PACKET_LOG 테이블은 해당 패킷의 캡처시간, 출발지나 목적지 포트 번호와 길이, 사용 프로토콜과 같은 엔티티로 구성하며 SECOND_LOG 테이블은 관리 프로토콜별 발생한 패킷량과 초당 내외부로 유입 또는 유출되는 패킷량을 관리하기 위한 엔티티로 구성된다.

1.2 데이터베이스 성능개선 설계

제안 시스템은 데이터의 증가로 발생하는 DBMS와 응용프로그램 사이의 오버헤드를 줄이기 위하여 압축 기억 엔진(archive storage engine)을 사용한다. 압축 기억 엔진은 대량의 데이터 발생으로 데이터베이스가 차지하는 공간을 줄이고 효율적으로 사용하기 위해 도입된 것으로 로컬 세그먼트에서 발생되는 패킷의 수가 많아 데이터를 압축 또는 가공 없이 저장한다면 데이터베이스 용량의 낭비와 데이터 질의(query)언어를 처리할 때 오버헤드로 인한 속도저하가 필연적으로 발생되는 문제를 해결하기 위한 방법이다.

2. 네트워크 침입탐지 모듈 설계 및 구현

비정상(anomaly) 탐지를 목적으로 구현한 통계기반 네트워크 침입탐지는 초당 로컬 세그먼트 상에서 발생되는 트래픽 량을 데이터베이스에 저장하고 이를 기준으로 단일표본 t검정을 통해 기존의 트래픽량의 평균과 표본 비교를 통하여 정상범주에 속하는지 판단하여 침입을 탐지한다. 주 감시 대상은 TCP, UDP, ICMP, ARP 패킷으로 네트워크상에서 가장 빈번하게 사용되는 프로토콜이다[1].

데이터베이스에 저장된 트래픽량의 평균을 산출하여 새로 들어온 트래픽량(X)과 비교를 통해 그 값이 정상범주에 속하는지를 판단해야 한다. 하지만 정상적인 범주의 트래픽인지를 판단하기 위하여 단순히 기존 트래픽량의 평균량만으로 판단하게 되면 개별 트래픽량에 대한 극단적인 값에 영향을 받기 때문에 값에 대한 추정과 검정을 실시하도록 하며, 제안시스템은 [표 4]의 검증된 단일표본 t검정을 이용하며, 침입탐지에 대한 흐름도는 [그림 6], 알고리즘은 [표 5]와 같다[2][6].

표 4. 단일표본 t검정

$T = \frac{\bar{X} - \mu_0}{s/\sqrt{n}}$	X : 갱신된 초당 패킷량(X) u : 데이터베이스의 초당 트래픽 량의 평균 s : u의 표준편차 n : X의 개수

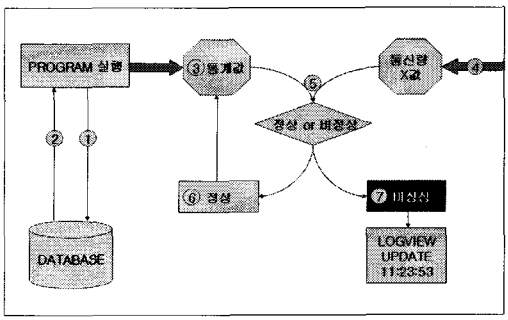


그림 6. 네트워크 침입탐지 흐름도

① 네트워크 침입탐지 프로그램에서 데이터베이스에 기존 통계량을 질의(select) 한다.
② 기존의 데이터베이스에서 자료를 축약하고 평균과 분산/표준편차를 이용한 통계 값을 계산한다.
③ 데이터베이스 질의(select)를 통해 받은 값을 프로그램에서 통계 기준 점으로 설정한다.
④ 패킷 캡처를 시작하면 실시간으로 패킷을 캡처하고, 프로토콜(TCP, UDP, ICMP, ARP)별 트래픽량을 초단위로 저장한다.
⑤ 데이터베이스에서 가져온 통계 값과 캡처된 트래픽량을 비교하여 캡처된 트래픽량이 기준치를 초과(비정상)하였는지 기준치와 비등하거나 그보다 작은지(정상)를 비교한다.
⑥ 정상인 경우 통계 값을 비교하는 ③번으로 회귀하여 위의 절차를 반복하여 실행한다.
⑦ 비정상의 경우에는 침입탐지 프로그램 LOGVIEW를 업데이트(update)한 후 사용자에게 경고(alarm) 메시지를 표현한다.

표 5. 단일표본 t검정 알고리즘

```
bool T_Test(int x,int avg,double sdt)
{
        double t;
        const double pree = 12.706;

        t = (x - avg) / ( sdt / sqrt(1));

        if(t < pree)
        {
          return true; // 귀무가설을 기각하지 않음
        }
        else
        {
          return false; // 귀무가설을 기각
        }
        return false;
}
```

3. 패킷 캡처/스캐너 설계 및 구현

3.1 패킷 캡처 설계 및 구현

커널 수준(kernel level)이 아닌 응용 수준(application level)에서 libpcap 라이브러리를 이용하여 패킷 캡처가 가능하도록 설계 및 구현하였다. 패킷 캡처의 특징으로는 모든 패킷을 대상으로 하는 것이 아닌, 사용이 가장 빈번한 프로토콜(TCP, UDP, ICMP, ARP)로 제한하고 대상이 되는 프로토콜의 통신량을 모니터링하는 것으로 [표 6]과 같으며 알고리즘은 [표 7]과 같다.

표 6. 패킷 캡처의 대상 프로토콜 및 패킷 정의

Protocol		Packet	
TCP	사용량	In	Incomming 패킷
UDP	사용량	Out	Outgoing 패킷
ICMP	사용량	Local	Local 패킷
ARP	사용량	Broad	Broadcasting 패킷

표 7. 패킷 캡처 알고리즘

```
while((res = pcap_next_ex(pcd, &header,
      (const u_char**)&packet)) >= 0)
{
    if(m_stop == true)
    {
        if(m_ms.date != t_sData->
           at(t_sData->size() - 1).date)
        {
            MinDataReset();  ·····①
            }
                break;
      }
    ...
    ep = (struct ether_header *)packet;
    packet = packet + sizeof(struct ether_header);
    m_PData.p_size = header->len;

    ether_type = ntohs(ep->ether_type);

    if(ether_type == ETHERTYPE_IP) ·····②
    {
        flag = IpPacket(packet);
    }
    else if(ether_type == ETHERTYPE_ARP)
    {
        m_PCount.arpc++;
        flag = true;
        ArpPacket(packet);
    }
    if(flag)
    {
        int re = InOutPacket(m_addr,m_PData.
            sour_ip,m_PData.dest_ip,
            m_check);·····③
          if(1 == re)
          {
                m_PData.in_out =1;
                m_inout.in++;
           }
                else if(2 == re)
                {
                        m_PData.in_out =2;
                        m_inout.out++;
                }
                else if(3 == re)
                {
                        m_PData.in_out =3;

                        m_inout.local++;
                }
                else
                {
                        m_PData.in_out =4;
                        m_inout.local++;
                }
          m_mutex.lock();
          InsertPacket(m_db1,m_ms.date,
                m_PData);···④
          m_mutex.unlock();
    }
}
```

패킷 캡처 알고리즘에서 사용자가 캡처 작업을 멈추었을 때 초단위로 저장되는 데이터베이스의 동기화를 위해 사용하며, MinDataReset() 함수를 호출하여 즉시 초단위로 가공된 구조체(m_ms)의 값을 할당하고 데이터베이스에 second_log 테이블에 저장한다(①). pcap필터를 사용하지 않고 IP패킷과 ARP패킷만을 캡처하기 위해 bool형 flag변수를 사용한다. 이 값이 논리연산의 참(true)이면 IP 또는 ARP 패킷으로 처리한다(②). InOutPacket()함수는 내부에서 외부로 나가는 패킷, 외부에서 내부로 들어오는 패킷, 로컬 네트워크에서 활동하는 패킷을 구별하기 위해 구현된 함수이다(③). 패킷이 발생 될 때 데이터베이스 packet_log 테이블에 저장한다(④).

3.2 호스트 스캐너 설계 및 구현

[그림 7]은 로컬 네트워크에 위치하는 호스트의 범위를 입력받아 내부 네트워크를 체크하는 그래픽 사용자 인터페이스(GUI)이며, 호스트 스캔 스레드(thread) 알고리즘은 [표 8]과 같다.

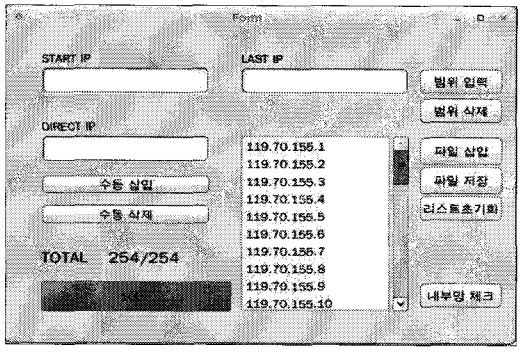


그림 7. 호스트 스캐너 GUI

m_mutex.lock는 개발 사용 도구인 QT전용의 QMutex이며 동기화 시켜준다(①). 스레드를 생성하여 arppacket으로 ip값을 넘겨 처리하고(②), 동기화를 해제한다(③). join함수를 실행시키지 않으면 도중에 메인 스레드가 종료되어 프로세스가 종료된다(④).

표 8. 호스트 스캔 스레드 생성 알고리즘

```
void   HostThread::run()
{
 ...
   for(int   i=0;i(m_iplist->size();i++)
   {
       strcpy(name,m_iplist->at(i).toAscii());

       m_mutex.lock();                    ..... ①
       if(pthread_create(&thread_ip[i],NULL,
                arppacket,(void*)name)(0) .....②
       {
            return;
       }
       usleep(10000);
       m_mutex.unlock();                  ..... ③
   }
   for(i=0;i(m_iplist->size();i++)

   {
        pthread_join(thread_ip[i],&t_return); ... ④
   }
}
```

3.3 포트 스캐너 설계 및 구현

[그림 8]은 호스트 스캔이 완료된 호스트에 대하여 포트 스캔을 실행하는 그래픽 사용자 인터페이스이며 3가지의 옵션(option)을 지원하고, 포트 스캔의 수행 결과를 출력하는 화면이다. 포트 스캔의 수행 결과는 트리구조로 표현한다. 1 ~ 65535 사이의 모든 포트 번호를 대상으로 스캔 작업을 수행하는 'ALL', 자주 사용되는 포트 번호만 목록화 하여 스캔 작업을 수행하는 'OFTEN PORT', 사용자가 수동으로 입력한 포트 번호만 스캔 작업을 수행하는 'SELECT PORT'로 구현하였다.

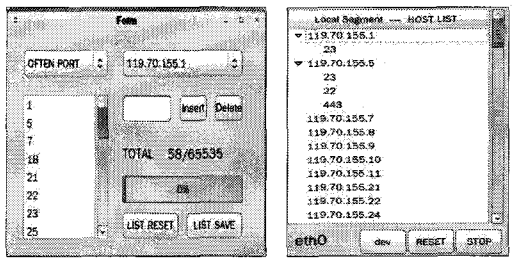


그림 8. 포트 스캐너 GUI

[표 9]는 포트 스캔을 위한 스레드를 생성하는 알고리즘으로 사용자가 지정한 포트의 개수만큼 Loop문을 지정하고(①), 리소스의 효율적인 사용을 위해 스레드는 항상 1000개를 유지한다(②). 프로그램의 실행시 매번 틀린 개수의 스레드를 동적으로 사용하기 위하여 메모리를 할당하고(③), Scan_ip 함수에 원하는 포트번호를 인자 값으로 넘겨서 스레드를 생성한다(④). 동적으

로 할당한 스레드의 메모리를 해제하고(⑤), 작업의 끝을 QEvent를 통해 MainWindow에 알린다(⑥).

표 9. 포트 스캔 스레드 생성 알고리즘

```
void   PortThread::run()
{
   ...
   for(int   i=0;i(pList->size();i++)·····①
   {
      while(1)
      {
        if(count  〈 1000)    ····· ②
        {
          pthread_t* thread_id =
          (pthread_t*)malloc
           (sizeof(pthread_t)); ····· ③
            pthread_create(thread_id,NULL,scan_ip,
              (void*)&pList->at(i)); ····· ④
          count++;
          free(thread_id); ····· ⑤
          usleep(1500);
          break;
          }
          else
          {
              usleep(15);
              continue;
          }
      }
   }
   for(int   i=0;i(suList.size();i++)
   {
        safeList->push_back(suList.at(i));
   }
   QEvent *event =
         new QEvent(QEvent::Type(5004)); ··· ⑥
   QApplication::postEvent(receiver,event);
}
```

3.4 Data Viewr 설계 및 구현

패킷 캡처의 대상이 되는 프로토콜(TCP, UDP, ICMP, ARP) 통신량을 모니터링하고 로컬(local) 네트워크의 패킷을 모니터링하여 로컬 세그먼트에서 이루어지는 모든 패킷의 활동을 사용자가 관리할 수 있도록 하기위하여 수집된 데이터를 [그림 9]와 같이 Data Viewr를 통해 그래프로 출력하도록 구현 하였다.

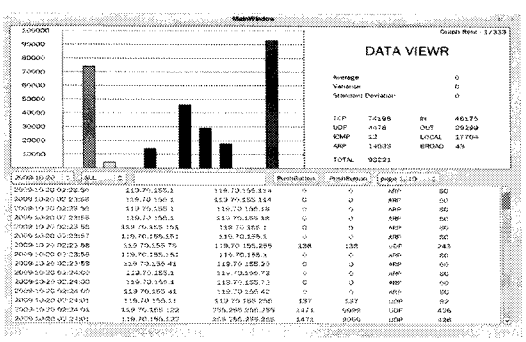


그림 9. Data Viewr

데이터베이스에서 트래픽의 통계와 패킷의 세부사항을 검색하기 위한 데이터베이스 검색 스레드 알고리즘은 [표 10]과 같다.

표 10. 데이터베이스 검색 스레드 알고리즘

```
int type = 0;
switch(m_check)
{
case 1:
              SelectAll();
             type = 6000;
             break;
default:
              break;
}
QEvent *event =
             new QEvent(QEvent::Type(type));
QApplication::postEvent(receiver,event);
```

V. 성능실험 및 결과

본 논문에서 제안한 멀티 플랫폼 기반의 네트워크 패킷 스캐너는 linux나 unix 환경에는 snort과 같은 여러 가지 네트워크 침입탐지와 패킷 관리 도구들이 있지만 대부분 문자 방식 사용자 인터페이스(CUI)를 기반으로 구현되어 익숙하지 않은 사용자들에게는 불편함이 따른다. 제안 시스템은 이러한 불편함을 개선하여 사용자에게 직관적이고 사용이 쉬운 인터페이스를 제공하기 위하여 그래픽 사용자 인터페이스(GUI)기반으로 구현하고, 모든 운영체제에서 구동될 수 있도록 멀티 플랫폼을 지원하는 Qt(C++)언어를 사용하여 설계 및 구현하였다. 성능실험을 위한 플랫폼 환경은 Inter(R) Pentium(R)4 CPU 3.00GHz, 1G RAM에 linux 커널에 기반한 완전한 오픈소스 운영체제 ubuntu 8.10, ubuntu 9.04를 대상으로 하였다. 성능실험을 위해서 시스템이나 네트워크를 운영할 때 보안상의 목적이나 장애처리를 위해 [표 11]과 같이 unix 계열 운영체제에서 빈번하게 사용하는 대표적인 툴(ngrep , snort, TCPdump)을 선정하였으며, 각 툴들이 가진 기능과 제안시스템이 가진 기능을 비교하여 결과를 도출하였다.

표 11. 성능실험 대상별 지원환경 결과

성능실험 대상	사용자 지원	운영체제	비 고
ngrep	CUI	Unix 계열	기능별 제품
snort	CUI	Unix 계열	기능별 제품
TCPdump	CUI	Unix 계열	Windows(Windump)
MNPS	GUI	Unix 계열 (ubuntu)	Multi-platform

제안시스템(MNPS)은 사용자 편의를 위한 그래픽 사용자 인터페이스(GUI : Graphical User Interface)를 지원하여 네트워크나 보안관리를 위한 사용자의 시간과 노력을 경감시킬 수 있다.

표 12. 성능실험 대상별 기능비교 결과

성능 실험 대상	기능비교 결과					
	패킷 캡처	호스트 스캔	포트 스캔	헤더 분석	데이터 분석	실시간 경고
ngrep	○	△	△	△	-	-
TCPdump	○	△	△	△	-	-
snort	○	○	○	○	○	-
MNPS	○	○	○	○	-	○

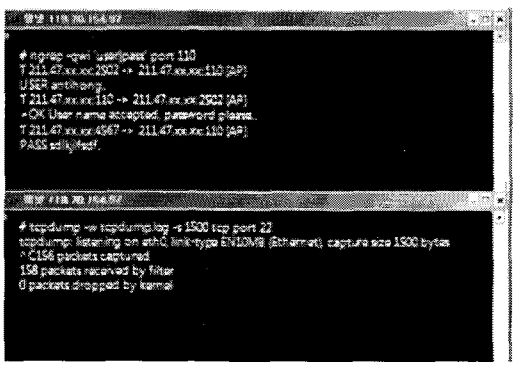


그림 10. 개별 명령어에 의한 부분적인 분석

[표 12]의 성능실험 대상별 기능비교와 같이 ngrep이나 TCPdump가 [그림 10]과 같이 개별 명령어에 의한 부분적인(△) 호스트 스캔, 포트 스캔, 헤더 분석 기능을 제공하지만 제안시스템은 snort와 같이 스캔을 통한 다양한 네트워크 공격에 대하여 종합적인(○) 기능을 제공한다.

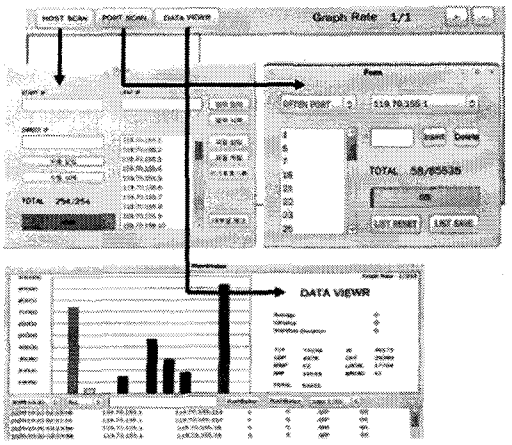


그림 11. 제안시스템의 기능실험

또한 snort은 지정 프로토콜에 대한 모니터링 기능은 제공하지만 실시간 경고 기능이 없어 관리자의 즉각적인 대응에 한계를 가지고 있지만, 제안시스템은 [그림 7]의 호스트 스캔, [그림 8]의 포트 스캔 기능에 부가하여 [그림 11]과 같이 "Data Viewr" 기능으로 지정 프로토콜에 대한 모니터링 기능을 실시간으로 제공한다.

VI. 결론

본 논문에서는 트래픽량을 기준으로 정상, 비정상을 판단하는 예외처리기반 네트워크 침입탐지 시스템을 대상으로 멀티 플랫폼 기반의 네트워크 패킷 스캐너를 설계하고 구현하는 것을 목표로 제안되었다. 이를 위해 멀티 플랫폼 기반의 네트워크 패킷 스캐너는 기존의 네트워크 침입탐지 기능에 부가하여 로컬 세그먼트 안에 있는 모든 호스트와 특정 포트에 대한 스캔 기능을 지원하도록 설계 및 구현하였다. 중요한 서비스를 지원하고 있는 서버나 서버의 포트가 내,외부적인 요인으로 정상적인 서비스가 중지되면 서비스의 중요도에 따라 심각한 피해가 발생할 수 있기 때문에 제안시스템에 관리 대상으로 분류하여 스캔 기능을 갖추도록 설계하였다.

Linux나 unix 환경에는 ngrep , snort, TCPdump와 같은 여러 가지 네트워크 침입탐지와 패킷 관리 도구들이 있지만 대부분 문자 방식 사용자 인터페이스(CUI :

Character based User Interface)를 기반으로 구현되어 익숙하지 않은 사용자들에게는 불편함이 따른다. 제안된 시스템은 이러한 불편함을 개선하여 사용자에게 직관적이고 사용이 쉬운 인터페이스를 제공하기 위하여 그래픽 사용자 인터페이스(GUI : Graphical User Interface)기반으로 구현하였고, 모든 운영체제에서 구동될 수 있도록 멀티 플랫폼을 지원하는 Qt(C++)언어를 사용하여 설계 및 구현하였다.

성능실험을 위해서 시스템이나 네트워크를 운영할 때 보안상의 목적이나 장애처리를 위해 일반적으로 사용하는 unix계열 운영체제에서 빈번하게 사용하는 대표적인 툴(ngrep , snort, TCPdump)을 선정하여 각 툴들이 가진 기능과 제안시스템이 가진 기능을 비교하여 결과를 도출하였다.

향후 연구 과제로는 제안 시스템이 패킷 캡처를 통한 프로토콜 분석기능에 부가하여 바이러스(virus)나 웜(worm)과 같은 다양한 네트워크 침입탐지 기능을 갖추도록 데이터 분석 기능을 확대하여 설계하는 연구가 필요하다.

참 고 문 헌

[1] 김기현, 김동욱, 박정곤, 권진현, 김도형, *탐지 오인률이 낮은 실시간 Anomaly IDS 개발*, 정보통신연구진흥원, 2008.

[2] 강석복, *통계적 추정과 가설검정*, 경문사, 2002.

[3] 자스민 블랑쉐, 마크 서머필드, *Qt4를 이용한 C++ GUI 프로그래밍*, ITC, 2009.

[4] 라용환, 천은홍, "비정상 연결시도를 탐지한 포트스캔 탐지 시스템의 설계 및 구현", 한국사이버테러정보전학회 정보·보안논문지, Vol.7, No.1, pp.63-75, 2007.

[5] 김재광, 김가을, 고광선, 강용혁, 엄영익, "비정상 트래픽 제어 프레임워크를 위한 퍼지로직 기반의 포트스캔 공격 탐지기법", 한국정보처리학회 제23회 춘계학술발표대회 논문지, pp.1185-1188, 2005.

[6] 김익수, 조혁, 김명호, "스캔 기반의 인터넷 웜 공격 탐지 및 탐지룰 생성 시스템 설계 및 구현", 정보처리학회논문지, Vol.12, No.98, pp.191-200, 2005.

[7] 한국정보보호진흥원, *"2008-정보보호 실태조사"*, 한국정보보호진흥원, 2009.

[8] H. Debar, D. Curry, and B. Feinstein, *The Intrusion Detection Message Exchange Format*, IETF Internet Draft, draft-ietf-idwg-idmef-xml-14, 2005.

[9] Y. Tang, L. Qian, B. Bou-Diab, A. Krishnamurthy, G. Damm, and Y. Wang, "High-Performance Implementation for Graph-Based Packet Classification Algorithm on Network Processor," IEEE International Conference on Communications (ICC 2004), Vol.2, pp.1268-1272, 2004.

[10] Anthony Jonesn, *Network Programming for Microsoft Windows - 2nd Edition*, 정보문화사, 2002.

[11] Frederic Cuppens, Alexander Mierge, "Alert Correlation in a Cooperative Intrusion Detection Framework," IEEE Symposium on Security and Privacy 2002.

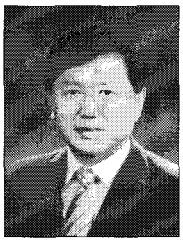
[12] IETF, *A Simple Network Management Protocol*, RFC 1157.

[13] Qt programming, "http://kylix.borlandforum.com/impboard/impboard.dll?action=read&db=kylix_tutorial&no=1"

[14] Qt technical paper, "http://qt.nokia.com/products"

저 자 소 개

이 우 인(Woo-In Lee) 정회원



- 1991년 2월 : 서울산업대학교 산업공학과(학사)
- 1995년 8월 : 숭실대학교 정보과학대학원 정보산업학과(석사)
- 2006년 ~ 2008년 : 서울벤처정보대학원대학교 컴퓨터응용기술학과 박사과정 수료
- 1981년 6월 ~ 1987년 2월 : 시스템공학연구소 연구원
- 1991년 7월 ~ 1995년 5월 : (주)선경유통 기술지원팀장
- 1997년 9월 ~ 2002년 2월 : 숭실대학교 전자계산원 강사
- 1999년 3월 ~ 2005년 2월 : 김포대학 겸임교수
- 2007년 7월 ~ 현재 : (주)유보트아이엔씨 대표이사

<관심분야> : S/W공학(특히, S/W 품질보증과 품질평가, 품질감리 및 컨설팅,SI), S/W 프로젝트관리, 품질경영

양 해 술(Hae-Sool Yang) 정회원

- 1975년 2월 홍익대학교 전기공학과 졸업(학사)
- 1878년 8월 성균관대학교 정보처리학과 졸업(석사)
- 1991년 3월 : 日本 오사카대학 정보공학과 S/W공학전공(공학박사)
- 1975년 5월 ~ 1979년 6월 : 육군중앙경리단 전자계산실 시스템분석장교
- 1980년 3월 ~ 1995년 5월 : 강원대학교 전자계산학과 교수
- 1986년 12월 ~ 1987년 12월 : 日本 오사카대학교 객원연구원
- 1995년 6월 ~ 2002년 12월 : 한국S/W품질연구소 소장
- 1999년 11월 ~ 현재 : 호서대학교 벤처전문대학원 교수

<관심분야> : S/W공학(특히, S/W 품질보증과 품질평가, 품질감리 및 컨설팅, OOA/OOD/OOP, SI), S/W 프로젝트관리, 품질경영