제1회 한국사이버테러정보전학회 춘계학술발표대회 (2004.5)

리눅스 서버의 효율적인 포트제어를 위한 페이크포트 설계 및 구현

김완경*, 소우영*, 김환국**, 서동일**
*한남대학교 컴퓨터공학과
**한국전자통신연구원

요 약

급속하게 발전한 정보통신 인프라로 인해 인터넷보급이 확산되었으나, 대부분의 사용자들의 전문 보안 지식 부재로 인하여 그 피해가 늘고 있다. 특히 리눅스를 PC로 이용하는 사용자들이 늘어감에 따라 정보유출 등 다양한 위협에 노출되어 있으며, 이는 악의적인 목적을 가진 공격자의 좋은 목표물이 되고 있다. 본 논문에서는 리눅스 서버를 이용하는 사용자들이 손쉽게 공격자로부터의 공격을 방어할 수 있도록 효율적인 포트제어를 할 수 있는 페이크포트(FakePort)를 설계 및 개발하였다.

Design and Implementation of FakePort for Efficient Port Control Based on Linux Server

Wan-Kyung Kim, Woo-Young Soh, Hwan-Kuk Kim, Dong-Il Seo

ABSTRACT

Internet is spread by the rapid development of information telecommunication infrastructure. But there are swinging damages becuase of most of the user has a shortage of knowledge. Especially, in proportion to increase of Linux user as PC, that is exposed to various threatener and is target of malicious attacker. In this paper, design and implement Fakeport based on Linux to control port efficiently and easily for protection against attack

1. 서 론

대부분의 리눅스 패키지들은 기본적으로 TCP/IP를 이용한 서비스(HTTP, TELNET, FTP, SAMBA...etc)를 제공하기 위하여 각 서비스마다 특정한 포트(port)를 열고 사용하게 된다. 리눅스를 처음 사용하는 사용자들이나 보안에 대하여 관심이 없는 사람들은 자신이 사용하지 않는 서비스용 포트가 열려 있는 것을 모른체 사용하는 경우가 많다. 대부분의 리눅스 시스템들은 이러한 해킹소지가 있는 포트들에 대하여 방어책이 존재하지 않는 경우가 많다. 기업이나 회사의 리눅스 시스템들은 방화벽이나 IDS, 백신서버 등 방어책의 일환으로 여러 가지 보안장비들을 사용하지만, 그것만으로 완벽한 보안이 보장되지 않는다. 또한 개인이나 소규모기업이 운영하는 인터넷 전자상거래 등 소호(SOHO)족이 나날이 증가하는 추세에 반해 보안에 대한

중요성의 인식은 여전히 미약하다고 할 수 있다. 리눅스가 개인 PC로도 널리 보급이 되어있는 상태에서 이러한 보안대책이 존재하지 않는 개인 사용자들은 언제든지 개인정보와 자료가 유출될 수 있다.

본 논문에서는 해킹으로 인한 위험성으로부터 해커를 속일 수 있고 또한 침입을 시도하는 대상에 대한 경고 메시지 및 역추적 할 수 있는 기본사항을 담고 있는 페이크포트를 설계 및 개발하였다. 또한 침입자가 악용하려고 하는 포트를 자동으로 막을 수 있는 기능을 추가함으로써 침입에 대한 보다 빠른 대처가 가능한 페이크포트는 개인 PC뿐만 아니라 대규모의 네트워크망이 형성되어 있는 구조에 위치에 제약을 받지 않고 단독 혹은 다른 보안솔루션과의 연동이 쉽다는 장점을 가지고 있다.

본 논문의 구성은 다음과 같다. 2 장에서는 서비스포트 번호와 포트 스캐닝에 대해 알아보고 3 장에서는 본 논문에서 제시하는 페이크포트에 대한 설계와 구현에 대하여 서술한다. 마지막으로 4장에서는 결론을 맺는다.

2. 서비스포트 번호와 포트 스캐닝

2.1 서비스 포트 번호(Service port number)[1][2]

컴퓨터 통신을 위한 구조인 OSI 7 Layer의 4번째인 트랜스포트 계층은 IP 계층의 도움을 받아 어떤 네트워크 엑세스 계층을 사용하는지에 무관하게 종점간(end-to-end) 호스트 사이의 데이터 송수신을 처리하는 역할을 한다. 트랜스포트 계층으로 인해 상위 계층(즉, 이 트랜스포트 이용자)에 해당하는 프로세스들은 이제 마치 하나의 컴퓨터 내의 프로세스 사이의 통신처럼, 거리감 없이 데이터를 송수신할 수 있게 되는 것이다.

이때 트랜스포트 계층이 네트워크로부터 수신한 메시지를 전달할 응용 프로세스(프로그램)를 구분하기 위하여 고유 번호를 배정하여 사용하는데 이 번호를 포트(port) 번호라고 한다.

TCP와 UDP는 각각 16 비트의 port 번호를 사용하므로 한 트랜스포트 계층(즉, 한 호스트)은 약 2 x 65000 개의 응용 프로세스에게 통신 서비스를 제공할 수 있다. IP 주소는 호스트까지의 패킷 전달에 사용되는 주소이고 그 호스트 내에서 이 패킷의 내용을 최종적으로 전달할 응용 프로세스를 구분하기 위하여 포트 번호가 사용된다. IETF에서 배정한 Well-known ports number는 영구적인 포트번호들로 1023번까지 지정되어 있다. TCP/IP를 이용하는 서비스 이름과 각 서비스에 배정된 포트 번호의 관계를 정리한 파일인 서비스 파일은 리눅스를 포함한 유닉스계열에서는 /etc/services에 이 내용이 들어 있다.

Network service	Port number	Internet style
echo	7/tcp(udp)	
netstat	15/tcp	
chargen	19/tcp(udp)	ttytst source
ftp-data	20/tcp	
ftp	21/tcp	
telnet	23/tcp	
smtp	25/tcp	mail
time	37/tcp(udp)	timeserver
name	42/udp	nameserver
whois	43/tcp	nickname
domain	53/tcp(udp)	
sunrpc	111/tcp(udp)	

(표 1) 서비스 파일의 예

(표 1)은 서비스 파일의 예를 나타내었다. 문제는 이 다양한 서비스 포트 중에서 일부는 리눅스 시스템 부팅 시 자동으로 열리게 된다는 점에 있다. 서비스를 하지 않는 불필요한, 혹은 보안상 허점이 많은 서비스 포트까지 자동으로 열려 해킹을 당할 수 있게 된다. 더불어 리눅스를 많이 다루어보지 못한 초보자들은 이 불필요

하게 열린 서비스포트를 쉽게 닫지 못한다는데 더 큰 문제가 있다.

2.2 포트 스캐닝(Scanning)[3]

포트 스캐닝과 해킹은 서로 밀접한 관련이 있다. 포트 스캐닝은 해킹을 하기 위한 첫 번째 시도라고 볼 수 있다. 물론 포트 스캐닝이 불법적인 해킹을 위해서만 쓰이는 것은 아니다. 역으로 생각하면 특정 호스트에 대해서 포트스캐닝을 시도하여 보안상 취약점이 있는 서비스를 제공하는지의 여부를 관리자가 점검할 수 있다.

포트 스캐닝은 특정 호스트나 서버로부터 어떤 서비스가 운영되고 있는지, 운영체제는 무엇인지 등의 여러 가지 정보를 얻을 수 있게 해준다. 열린 포트가 알려지는 것은 어떤 응용프로그램이 실행 중인지를 알게 해 주기 때문에 상당히 치명적이다. 최근 들어 많은 종류의 자동화된 스캐너들이 개발되고 있고, 결과적으로 더 많은 공격이 시도되고 있다. 또한 스캔의 근원지를 숨겨 어디서 스캐닝공격을 하는지 알 수 없게 하는 등, 공격을 탐지하기 어렵도록 많은 방법과 기술이 동원되고 있는 실정이다. 물론 스캐닝 공격을 탐지하는 스캔 디텍터들도 많지만 공개되지 않은 새로운 패턴을 이용한 스캔 공격은 사실상 잡기가 어렵다.

이러한 포트 스캐닝기법 중 Ping Sweep등의 기법은 시스템이 동작중인지 아닌지 혹은 네트워크에 연결되었는지 아닌지를 확인할 수 있게 해준다. 다음단계는 TCP와 UDP포트에 접속을 시도해 보아 시스템이 현재 어떤 서비스를 하고 있는지 또는 리스닝(Listening)상태인지 확인하는 것이다. 이 기법을 포트 스캐닝이라고 부른다. 하지만 요즘은 위의 기본적인 스캐닝방법 외에도 무수히 많은 기법을 사용하여 공격탐지를 어렵게 만든다. (표 2)는 가장 많이 사용하는 포트 스캐닝원리들을 나타내었다.

이 름	원 리
TCP connect scan	• 대상 시스템의 대상 포트에 TCP연결을 시도하여 구현하는 가장 기본적인 스캐닝 기법 • 구현이 가장 쉬운 반면, 공격 대상 시스템의 로그 파일에는 즉시 많은 양의 연결 메시지와 에러 메시지가 저장되기 때문에 잘 사용하지 않음
TCP SYN scan	• 완전한 TCP연결을 하지 않음 • SYN패킷을 보내고 응답을 기다림 • SYN/ACK패킷을 받는다면 그것은 그 포트가 listening상태 • RST/ACK 패킷을 받는다면 그 포트가 닫혀있다는 것을 의미 • SYN/ACK신호를 받으면 RST패킷을 만들어 즉각 연결을 끊어 버림
UDP scan	• 목적 포트에 UDP 패킷을 전송, 목적 포트가 "ICMP port unreachable" 메시지로 응답하면 포트는 닫힌 것, 반대이면 포트는 열린 것으로 추측
Stealth scan	• 고의적으로 3-way 핸드쉐이킹을 위반하는 것이 핵심 • 방화벽이나 라우터 등의 필터링규칙을 통과하거나 대상 시스템에서 현재 구동중인 로깅(logging) 메커니즘을 우회하기 위해서 사용

(표 2) 포트 스캐닝의 종류와 원리

3. 설계 및 구현

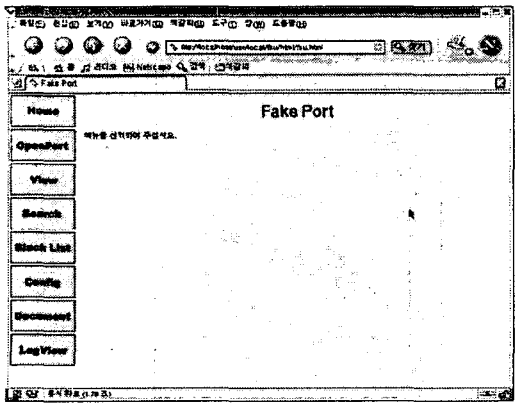
3.1 개발 환경

본 논문에서 구현한 페이크포트는 Redhat 9.0을 기본 운영체제로 사용하였으며, 사용자의 편의를 위하여 X-WINDOW 환경과 Netscape 웹 브라우저를 사용하여 GUI 환경으로 나타낼 수 있도록 하였다. 또한 특정 포트의 차단과 허용을 위해 1.2.7a를 사용하였다.

3.2 기능

페이크포트는 불필요하게 열린 포트번호를 찾아주며 손쉽게 닫아주는 역할을 할 뿐만 아니라 고속의 패킷필터링을 통한 실시간 포트스캔 탐

지를 가능하게 해준다. 또한 스캔 공격자료를 보관, 관리할 수 있으며 스캔 공격자에 대해서 시스템의 불법적인 접근을 차단할 수 있거나 자세한 로그를 생성할 수 있다. 그리고 핵심 모듈중 하나로, 위장포트를 사용하여 실제로 열려있지 않은 포트를 열려있는 것처럼 하여 거짓정보를 되돌려 주는 기능도 제공한다. 이렇게 함으로써, 공격자는 대상 시스템에 정확히 어떤 포트가 열려있는지 구별하기 어렵게 되는 것이다.

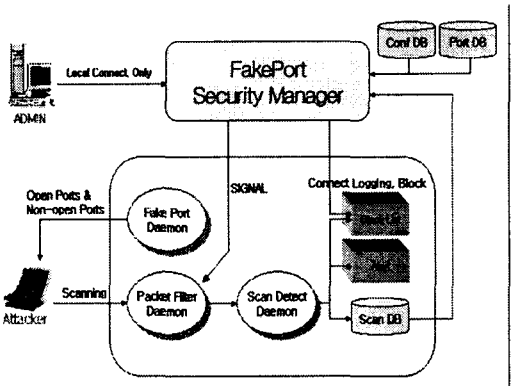


[그림 1] 페이크포트 설정화면

모듈명	기 능
SM (Scan-detect Manager)	각종 포트관리 및 제어, 스캔 감사자료 관리 블랙리스트 관리, 환경설정관리
Fake Port	위장 포트를 생성하여 거짓정보를 되돌려 줌
Packet Filter	시스템으로 들어오는 모든 패킷을 필터링 함
Scan Detect	패킷을 분석해서 스캔인지 아닌지 판별함
start / stop	페이크포트 프로그램을 구동 및 정지
Alarm	스캔공격 판단 시 관리자에게 경고 메시지 전달
PortDBMake	각종 포트 정보를 DB화하여 목록을 참조할 수 있고, 관리자가 임의로 내용을 수정하여 포트 정보를 알아보기 쉽게 함.
BlackList Logging	블랙리스트로 지정된 IP의 호스트에서 텔넷으로 시스템 접근시 차단하지 않고 모든 행동을 로깅함

(표 3) 페이크포트 모듈별 기능

위 [그림 1]은 페이크포트의 메인 화면을 보여주고 있다. 페이크포트의 각 모듈과 기능은 크게 포트관리 및 제어, 그리고 스캔 자료를 관리하는 유저 인터페이스(UI)인 웹 서버와 패킷을 걸러내서 스캔을 탐지하는 모듈로 나눌 수 있다. 자세히 살펴보면 (표 3)과 같다.



[그림 2] 페이크포트 구조도

[그림 2]는 각 모듈의 상호 연관성을 잘 나타내주는 페이크포트 개략적인 구조도이다.

페이크포트 구동모듈로 프로그램을 실행시키면 3개의 프로세스가 형성된다. 이후부터 페이크포트는 시스템으로 유입되는 모든 패킷을 감시하기 시작한다. 모든 패킷은 Scan Detect모듈로 넘겨 스캔인지를 판단하게 한다. Packet Filter 모듈은 오직 필터링만 하고 Scan Detect모듈은 오직 스캔인지만을 판단함으로써 패킷손실을 최소화하였다. 그러다가 공격자가 포트스캔을 하면 가장먼저 페이크포트 모듈이 허위로 열어둔 포트가 모두 열려있다고 거짓정보를 되돌려주고 Scan Detect 모듈은 스캔공격사실을 DB에 저장시키고 알람모듈을 구동시킨다.

관리자는 알람모듈로부터 받은 공격사실을 통보 받거나, 수시로 페이크포트 SM 모듈을 통해 포트스캔 공격자료를 상세히 검토하게 된다. 페

이크포트 SM 모듈은 미니 웹서버로써 최소한의 동작처리만을 위해 자체 제작하였다. 모든 사용자에게 관리의 편리성을 위하여 친숙한 웹브라우저 환경을 선택하였다. 관리자는 스캔공격자료를 살펴보고 해당 호스트에 대해서 블랙리스트로 지정할 수 있으며 여러 가지 환경설정을 바꿀 수 있다.

3.2 페이크포트 실행

각각의 실행파일과 데몬의 구성을 살펴보면, fsu란 데몬은 데몬을 쉽게 실행시켜 주기 위한 데몬이다. 리눅스에서 아파치를 실행시킬 때 httpd start 또는 종료할 때 httpd stop으로 쉽게 실행과 종료를 할 수 있다. 페이크포트도 이러한 방식으로 작동시키게 하는 역할이 fsu데몬이며 그에 관한 소스는 fsu.c 이다.

```
Strcpy(how, argc[1]);   //arg[1]으로부터 시작인지 종료인지의 방식을 얻는다.

if(strcmp(how, "start") == 0)   //arg[1]이 시작하라는 의미 이면
{
            .
            .
            .
   sprintf(working_file, "%s/bin/filterd&", install_dir);
 //filterd를 백그라운드로 실행.
   system(working_file);
   sleep(1);

   sprintf(working_file,          "%s/bin/detectd&",
install_dir); //detectd를 백그라운드로 실행.
   system(working_file);
   sleep(1);

   /* fakedaemon을 백그라운드로 실행 */
   sprintf(working_file,          "%s/bin/fakedaemon&",
install_dir);
   system(working_file);
   printf("\nFSU START...\n"); // pid.dat란 곳에 각각의 프로세스 아이디를 저장
            .
            .
            .
else if(strcmp(how, "stop") == 0)
{
   sprintf(working_file, "%s/bin/pid.dat", install_dir);
   fd  = open(working_file, O_RDONLY);
   read(fd, (char *)&daemonp, sizeof(daemonp));

   kill(daemonp.filterpid, SIGINT);    //시그널을 통해 filterd 데몬을 종료.
   sleep(1);
   kill(daemonp.detectpid, SIGINT); //시그널을 통해 detectd 데몬을 종료.
   sleep(1);
   kill(daemonp.fakepid, SIGINT);    //시그널을 통해 fakedaemon 데몬을 종료.
   close(fd);

   printf("\nFSU STOP..\n");
}
else
   printf("Usage: %s start or %s stop\n",argc[0], argc[0]);
```

이 프로그램은 각각의 실행을 백그라운드로 실행후 pid.dat란 곳에 구조체로 각각의 프로세스 아이디를 저장한다. 프로세스 아이디를 저장할 구조체는 다음과 같다.

```
typedef struct _daemonpid
{
   int filterpid;   //filterd의 프로세스 아이디
   int detectpid; //detectd의 프로세스 아이디
   int fakepid;   //fakedaemon의 프로세스 아이디
   int start;      //시작되었는지를 체크
} daemonpid;
```

위의 구조체를 통하여 각각의 프로세스 아이디를 저장한 다음 콘솔에서 fsu stop을 실행하면 각각의 프로세스 아이디에 시그널을 주어서 각각의 데몬을 종료시키게 된다.

daemonpid 구조체의 int start의 기능은 한번 시작한 데몬을 다시 실행 할 수 없도록 하게 하는 기능을 가지고 있다. 페이크포트 start를 실행하면 daemonpid.start가 1로 세팅되고, 페이크포트 start가 다시 실행 할 수 없도록 하게 한다.

페이크포트의 핵심인 필터링(filtering) 모듈과 스캔 디텍트(scan detect) 모듈에 대해서 알아볼 것이다. 페이크포트는 이외에도 페이크포트와 더

불어 세 가지의 프로세스가 동작해서 작업을 처리한다.

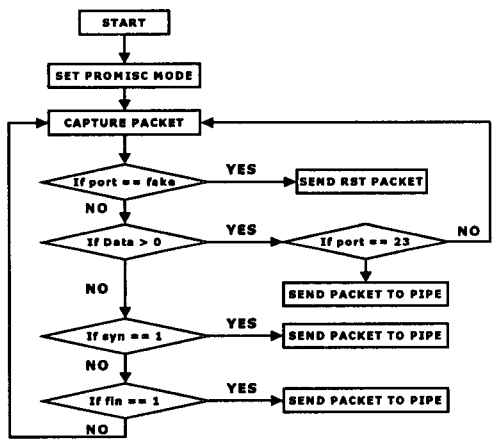
필터링 데몬은 패킷 필터링만을 위해서 동작하며, 디텍트 데몬은 필터링한 패킷을 분석하여 스캔인지 탐지를 한다. 마지막으로 페이크포트 데몬은 허위 포트를 열어 공격자에게 거짓정보를 돌려주는 역할을 한다. 소스는 filter.c, detect.c, fakedaemon.c로 구성되어 있다.

filter.c의 기능은 패킷을 받아서 파이프에 저장하며 허위 포트를 열어준 것에 대해 세션을 종료하는 기능과 실제 열린 포트에 대해 관리하는 기능을 가지고 있다.

detect.c는 파이프에서 패킷을 읽어서 스캔임을 탐지하는 기능과 탐지 후 이메일과 DB 그리고, log 파일에 탐지 내용을 저장하는 기능을 가지고 있으며 스캔한 IP에 대해 블랙리스트를 관리하며 블랙리스트에 대해 로그를 남기는 기능을 가지고 있다.

fakedamon.c는 단순히 fsu.conf란 파일에 저장된 포트번호에 대해 포트를 여는 기능을 한다.

fsu.c는 이러한 데몬을 쉽게 실행 종료하기 위한 소스이다.



[그림 3] 페이크포트 필터링

filter.c를 컴파일하여 만든 데몬으로서 패킷 필터링을 주요 기능으로 하고 있다. 이 데몬은 패킷을 받아서 파이프에 계속적으로 보낸다. 파이프를 사용하여 필터링 데몬과 탐지 데몬을 나눈 이유는 기존의 스캔 탐지 시스템의 단점인 패킷을 놓치지 않기 위해 프로세스를 여러 개로 두어 사용하는 것이다. [그림 3]은 패킷 필터링하는 과정을 흐름도로 나타내주고 있다.

일반적으로 스캐너는 3-handshake를 이용한 syn 패킷을 날림으로써 스캔을 하지만 최근의 nmap등의 스캐너는 fin 플래그도 사용한다. 위에서 걸러낸 패킷으로 스텔스 스캔 또한 탐지해낼 수 있다.

filterd는 위의 패킷 필터링 기능을 우선으로 하며 다른 기능으로는 허위 포트에 대한 세션 블로킹과 실제로 열린 포트를 저장하는 기능을 갖추고 있다.

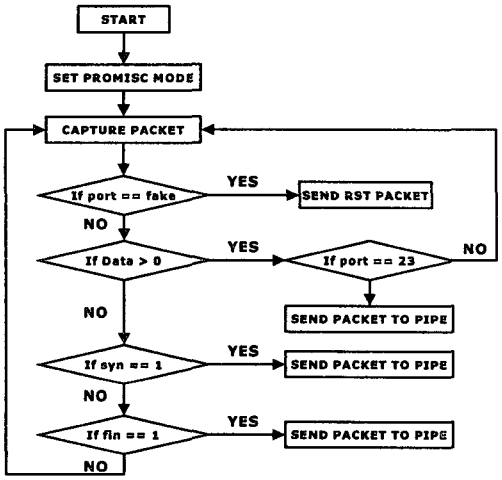
허위 포트에 대한 세션 블러킹은 다음과 같다.

```
for( i = 0; i<fakeopen; i++)
{
    if(portopen[i] == ntohs(tcp->dest))
        search = 1;
}
if(search == 1)
{
    sprintf(saddr, "%s", inet_ntoa(ip->saddr));
    sprintf(daddr, "%s", inet_ntoa(ip->daddr));
    if(tcp->ack == 1)
    {
        tcp_send(daddr, saddr, ntohs(tcp->dest),
htohs(tcp->source),
ntohl(tcp->ack_seq), htohl(tcp->seq), (4*(tcp->doff)
- sizeof(ep.tcp)));
    }
    search = 0;
}
```

위에서 먼저 허위 포트임을 찾으며 허위 포트일 경우 tcp_send란 함수를 통하여 rst가 1인 패킷을 날려서 TCP 세션을 블로킹한다. rst는 reset을 뜻하는 연결 재 설정을 요청하는 플래그이다. 이 플래그를 사용하여 sequence 번호와 acknowledge 번호를 이용하여 보내면 TCP의 세션을 블로킹 할 수 있다. 이러한 TCP 세션을 핸들링하는 것의 대표적인 공격 중에 Hijacking

이란 공격을 들 수 있다.

탐지 모듈인 detect.c 소스에는 스캔을 탐지하는 기능과 탐지된 아이피에 대해 블랙리스트를 저장하는 기능 및 탐지 한 결과를 이메일과 DB 그리고, log 파일에 저장하며 블랙리스트에 대해 로깅 하는 기능을 가지고 있다. 아래 [그림 4]는 탐지모듈의 흐름도이다.



[그림 4] Scan Detect

4. 결 론

인터넷 사용자의 급증과 맞물려 보안의 중요성이 나날이 증대되고 있음에도 불구하고 보안의식은 여전히 부족한 실정이다. 또한 보안전문지식을 습득할 수 있는 기회와 조건이 아직까지는 부족한 현실이다. 본 논문에서는 보안전문지식이 부족한 일반사용자들이 침입에 대응하여 손쉽게 제어, 통제 할 수 있는 페이크포트를 설계 및 구현하였다.

향후 과제로서는 개인 사용자뿐만 아니라 방화벽 또는 IDS등의 보안솔루션을 사용하는 대규모의 네트워크에 적용하기 위해 API가 개발되어야 할 것이다. 이 경우 페이크포트는 방화벽과 연동되어 우회공격에 대응하여 페이크포트에서 한번 더 차단할 수 있는 기능과 정책설정에 의하여 스캔에 이은 불법적인 접속을 시도하는 경우 그에 대한 패턴을 저장하고 있는 IDS와의 연동에 의해 해당 ip를 블랙리스트에 추가하고 포트를 차단할 수 있는 역할을 수행 할 수 있을 것이다.

참고문헌

[1] Marina del Rey, "TRANSMISSION CONTROL PROTOCOL, DARPA INTERNET PROGRAM PROTOCOL SPECIFICATION", Defense Advanced Research Projects Agency Information Processing Techniques Office, 1981, 9

[2] J. Postel, " User Datagram Protocol ", ISI, 1980,8

[3] Rich Jankowski, "Scanning and Defending Networks with Nmap", The Linux Community's Center for Security, 6,2002

김 완 경

2003년 ~ 현재 한남대학교 컴퓨터공학과 석사과정

소 우 영

1992년 ~ 현재 한남대학교 컴퓨터공학과 교수

김 환 국

2002년 5월 ~ 현재 한국전자통신연구원 네트워크보안구조팀

서 동 일

1994년 3월 ~ 현재 한국전자통신연구원 네트워크보안구조팀 연구팀장