Linux 커널 수정을 통한 Buffer Overflow Attack 방지에 관한 연구

서정택°, 최홍민, 손태식, 김동규
아주대학교 컴퓨터공학과
{ sjtgood, partout, tsshon, dkkim }@madang.ajou.ac.kr

A Study on the Protection against Buffer Overflow Attack using Modifed Linux Kernel

Jung-Taek Seo°, Hong-Min Choi, Tae-Shik Shon, Dong-Kyoo Kim
Department of Computer Engineering, Ajou University

요 약

Linux 는 다양한 하드웨어 플랫폼을 지원하며, 강력한 네트워크 지원 기능, 다양한 형식의 파일시스템 지원 기능 등 높은 성능을 자랑한다. 그러나, 소스코드의 공개로 인하여 많은 보안상의 취약성을 내포하고 있으며, 최근 이를 이용한 해킹사고가 많이 발생하고 있다. 본 논문에서는 Linux 상에 상존하는 보안 취약성을 조사하고, 보안 요구사항을 도출하며, 최근 해킹의 상당부분을 차지하고있는 Buffer Overflow 공격 방지를 위한 방안으로 커널 수정을 통해 Secure Linux 를 개발하고자 한다.

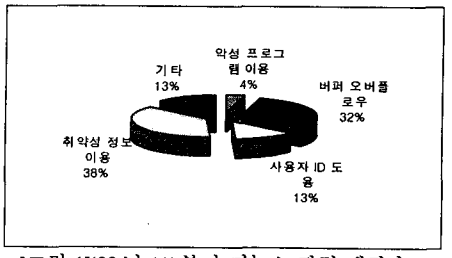
1. 서 론

인터넷을 통해 전 세계의 컴퓨터들이 네트워크로 연결되면서 해킹이나 바이러스와 같은 침해 사고들이 빈번하게 발생하여 많은 피해를 입히고 있다. 최근의 침해사례를 살펴보면 컴퓨터 시스템에 대한 해킹수법이 지능화되고 다양해지면서 특히 운영체제 등의 시스템 취약점을 이용하는 수법이 해킹의 상당 부분을 차지하고 있다. 또한 리눅스(Linux)라는 운영체제가 무료로 배포되면서 리눅스의 취약성을 이용한 해킹사고가 많이 발생되고 있다. 리눅스는 소스코드의 공개와 관련 문서의 풍부한 제공으로 쉽게 수정이 가능하며, 많은 사람의 검증을 거친 운영체제로 안정성이 뛰어나며, PC 기반에서 운영이 가능하고 설치비용도 적게 들어 최근 급속도로 보급이 확산되고 있다. 따라서 리눅스 운영체제에 보안기능을 첨가하는 것이 시급한 일이라 할 수 있다. 또한 최근의 운영체제 개발동향이 커널의 최소화 방향으로 나아가고 있으므로 마이크로 커널 기반의 안전한 리눅스 개발이 필요하다.

본 논문에서는 리눅스 운영체제에 상존하는 취약성을 고찰하고, 이러한 취약성을 제거하여 Secure Linux 시스템을 만들기 위한 보안 요구사항을 도출하며, 여러 가지 해킹 기법 중에서 최근 가장 많이 발생하고 있는 Buffer Overflow 공격에 대한 고찰과, 공격에 대한 대응방안으로 리눅스 커널 수정을 통해 Secure Linux 를 개발하고자 한다.

2. 리눅스 보안의 취약성

현존하는 대부분의 운영체제가 그러하듯이 리눅스도 많은 보안 취약성을 가지고 있다. 리눅스는 기존의 유닉스가 지닌 보안 문제점들을 고스란히 물려받고 있는데 이러한 것들은 윈도 계열의 OS 를 사용하던 사용자들에게는 매우 생소한 부분이고 따라서 매우 소홀하게 취급되고 있는 것이 지금의 현실이다. 게다가 커널의 소스가 공개되어 있고 리눅스를 돌리는 주 플랫폼이 일반인들에게 아주 익숙한 인텔 x86 CPU 라는 점 때문에 최신 보안 문제가 나왔을 때 리눅스용 해킹 스크립트가 제일 먼저 만들어지고 있다.[3]



[그림 1]'99 년 1/4 분기 리눅스 관련 해킹사고

리눅스의 취약성 중 최근에 발견된 공격에 가장 많이 사용되는 취약점은 다음과 같다.[1][2]

2.1 IMAP 취약점

IMAP 서버는 리눅스 운영체제를 설치할 때 기본적으로 설치되는 것으로 사용자의 메일을 서버에서 관리하도록 해주는 프로그램이다. 보통 일반 사용자들은 POP 서버를 사용하며 IMAP 서버는 관리자가 인지하지 못하는 사이에 데몬으로 동작하고 있는 경우가 많다. IMAP 서버가 사용자 확인을 위하여 ID 와 패스워드를 입력 받을 때 그 길이에 대한 한계 값 검사를 충분히 하지 않아 조작된 값을 입력함으로써 공격자는 원격에서 루트 권한으로 임의의 명령을 수행시

킬 수 있다.

2.2 POP 서버 취약점

POP서버는 PC 등 클라이언트에서 메일서버에 접속하여 메일을 송·수신하도록 하는 서비스를 제공한다. 사용자가 리눅스 시스템의 POP서버에 접속할 때 사용자 ID 와 패스워드를 입력하게 되고 이러한 인증 절차를 거쳐서 사용자는 리눅스 시스템에 저장된 자신의 메일을 확인할 수 있다. 하지만 POP 서버가 사용자 확인을 위하여 ID 와 패스워드를 입력 받을 때 그 길이에 대한 한계 값을 검사하지 않아 조작된 값을 입력함으로써 원격에서 루트 권한으로 임의의 명령을 수행시킬 수 있다.

2.3 bind 취약점

BIND 4.9.7 이전 버전과 BIND 8.1.2 이전 버전에서 inverse query 요청에 대한 응답 시 적절한 한계 값 검사를 하지 않아 버퍼 오버플로우 취약점이 존재한다. 공격자는 교묘히 조작한 패킷을 전송하여 루트 권한으로 임의의 명령을 실행시킬 수 있다.

2.4 mountd 취약점

NFS(Network File System)는 네트워크를 통하여 컴퓨터간에 파일 시스템을 공유하기 위한 클라이언트/서버 프로그램이다. NFS 클라이언트가 NFS 서버의 파일에 접근하기 위해서는 먼저 파일 시스템을 마운트 한다는 요청을 하게 되는데, 이 NFS 마운트 요청을 처리하는 소프트웨어(mountd 프로그램)에 버퍼 오버플로우 취약점이 존재한다. 특히 리눅스 시스템에서는 기본적으로 NFS 서버가 mountd 를 구동하기 때문에 매우 위험하며, 공격자는 시스템 관리자 접근 권한을 획득할 수 있다.

3. Secure Linux 를 위한 보안 요구사항[2]

현재의 보안제품 연구개발은 컴퓨터 시스템 상에 보안 서비스를 추가하는 방식으로 진행되어왔다. 하지만 이러한 방식으로는 신종 해킹을 해결할 수 없으며, 문제 발생시마다 새로운 보안제품을 계속적으로 추가하는데 드는 비용과 번거로움이 뒤따른다. 따라서 컴퓨터 시스템의 운영체제 내부 커널에 보안기능을 추가하는 연구 개발이 필요하며, 안전한 리눅스 시스템을 만들기 위한 보안 요구사항은 다음과 같다.

3.1 식별 및 인증(Identification & Authentication)

시스템 내부의 자원에 대한 접근통제의 근간이 되는 것으로 시스템을 사용하려는 개인의 식별 및 인증이다.

3.2 접근통제(Access Control)

시스템 내부의 자원에 대한 접근을 통제하는 메커니즘으로 안전한 운영체제의 핵심 기능이다.

3.3 신뢰 경로(Trusted Path)

패스워드를 변경하거나 접근 권한을 바꾸는 것과 같은 중요한 보안관리 기능을 수행할 때에는 시스템의 보안을 시행하는 부분 즉, TCB 와의 신뢰성 있는 경로가 설정되어야 한다.

3.4 감사(Audit)

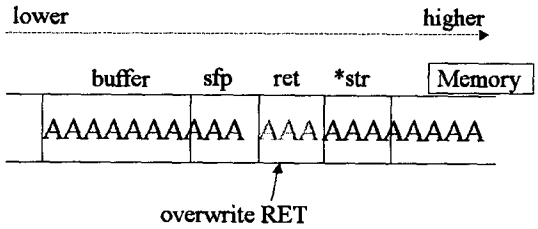
일반적으로 발생한 보안 관련 사건의 로그를 유지하여야 한다. 이러한 감사로그는 외부자로부터 보호되어야만 하며 모든 보안 관련 사건은 기록되어야 한다.

3.5 완전한 중재(Complete Mediation)

접근통제를 효과적으로 수행하기 위해서는 시스템 내부 모든 자원에 대한 모든 접근을 중재하여야 한다.

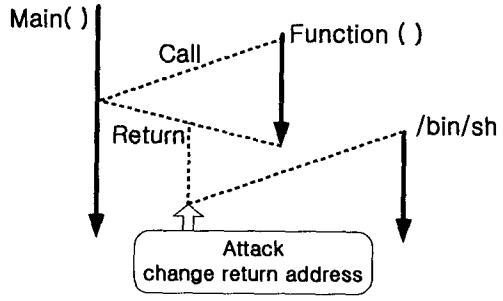
4. Buffer Overflow Attack

버퍼 오버플로우는 1988 년 전세계를 떠들썩하게 만든 Morris 가 만든 웜에서 finger 데몬을 이용한 공격이 그 시초라고 알려지고 있다. 오랜 역사를 가졌지만, 최근에 많이 발견된 공격법으로 지정된 버퍼의 크기보다 더 많은 데이터를 입력해서 프로그램이 비정상적으로 동작하도록 만드는 것이다. 프로그램 실행시의 메모리 구조를 이용해서 프로세스의 흐름을 조정(스택에서 Return Address 수정)한다. 이때 쉘을 실행시킴으로써 임의의 작업을 수행시킬 수 있는 발판을 마련하는 것이 일반적이다. 따라서 Root 권한 획득이 용이해진다.[1]



[그림 2] 버퍼 오버플로우 시 메모리 구조

대처방안으로는 소스코드의 계속적인 패치가 있을 수 있고, 운영체제 커널을 수정하여 리턴하기 전에 스택의 무결성을 체크 하거나, 스택 내에서는 프로그램의 실행을 금지시키거나, 루트 권한으로 실행되는 프로그램을 최소화 하는 등의 방법이 있다. 본 논문에서는 운영체제 커널의 수정을 통한 근본적인 해결책으로 접근하고자 한다.



[그림 3] 복귀주소의 변경

5. Buffer Overflow Attack 탐지를 위한 구현[4]

버퍼 오버플로우를 방지하기 위해 실제로 커널내에서 수정한 부분은 Setuid 프로그램의 이용여부를 체크하고 프로그램의 인자들과 환경변수들을 메모리에 적재 시키는 곳으로 버퍼의 길이가 비정상적으로 길고 NOP 연산이 처음 부분부터 연속하여 있을 경우를 검사하여 프로그램의 수행을 중단시킨다. 버퍼 오버플로우 공격을 방지하기 위해 추가된 헤더파일과 소스코드는 다음과 같다.

• HEADER FILE(/usr/src/linux/include/asm/errno.h)

```
#define EBUFATTACK 200
```

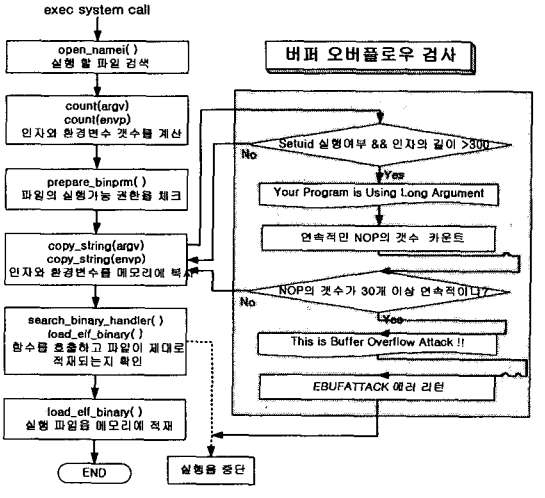
• SOURCE FILE(/usr/src/linux/fs/exec.c)

```
printk("idchage_flag=%d  len=%d \n",
idchange_flag, len);
printk("%s \n", str);
if ((idchange_flag == 1) && (len > 300)) {

  printk("Your program is using long
argument\n");

  for (i=0; i<30; i++) {
    if (str[i] != 0x90) break;
  }
  if (i==30) {
    printk("This is Buffer Overflow
Attack !!\n");
    return -EBUFATTACK;
  }
}
```

우선 헤더파일 내에 버퍼 오버플로우 공격의 발생 여부를 표시하기 위한 에러코드를 리턴하기위해서 EBUFATTACK라는 상수를 선언했다. 소스 내에서는 Setuid가 설정된 프로그램의 실행 여부와 인자의 길이가 300을 넘는지를 비교하며, 버퍼의 앞부분에 NOP가 계속해서 들어오는 것을 체크하여 버퍼 오버플로우 공격을 판정하고 공격으로 판정되면 EBUFATTACK 코드를 리턴한다. 이와 같은 방법으로 버퍼 오버플로우 공격이 판별되면 프로그램 실행을 종료 시킴으로써 공격자의 루트권한 획득을 저지시킬 수 있다.



[그림 4] 커널 내부의 버퍼 오버플로우 방지 흐름도

리눅스 커널 수정전의 버퍼 오버플로우 공격 실험 결과는 다음과 같다. 공격자가 *myexploit* 를 실행 시킴으로써 쉽게 루트 권한을 얻게 되는 것을 확인할 수 있다.

```
[seojt@airborne seojt]$ whoami
seojt
```

```
[seojt@airborne seojt]$./myexploit
bash# whoami
root
```

< 커널 수정 전의 버퍼 오버플로우 공격 예 >

리눅스 커널 내에 버퍼 오버플로우를 검하하는 소스코드를 삽입하고 같은 조건 하에서 공격을 시도하였다. 결과적으로 버퍼 오버플로우가 탐지되고 실행을 중단시키는 것을 확인할 수 있었다.

```
[seojt@airborne seojt]$ whoami
seojt
[seojt@airborne seojt]$./myexploit
Your program is using long argement.
This is Buffer Overflow Attack !!
Program END
[seojt@airborne seojt]$ whoami
seojt
```

6. 결론 및 향후 연구방향

최근 공격 기법으로 많이 사용되고 있는 버퍼 오버플로우 공격에 대한 그 동안의 대응책으로는 공격 발생시마다 프로그램을 패치하는 방법이 주였다. 본 논문의 구현에 있어서도 최근 리눅스 버전들은 프로그램상에서 패치가 되었음을 알 수 있었지만 패치에 의존하는 방식은 새로운 환경과 새로운 공격에 대한 취약성 발견 시마다 패치를 해줘야 하는 번거로움과 패치하는 과정 역시 쉽지 않아 옳은 방법이라 할 수 없다. 따라서 본 논문에서는 버퍼 오버플로우 공격의 3 가지 특성을 이용하여 버퍼 오버플로우 공격을 탐지하는 모듈을 커널에 삽입하였다. 커널 수정을 통한 접근은 프로그램의 패치 보다 근본적인 해결책이라 할 수 있으며, 프로그램의 인자로 넣는 공격 기법과 프로그램을 실행시킬 때 환경변수를 이용하는 공격 모두를 차단할 수 있으며 운영체제 자체의 취약성을 근본적으로 제거할 수 있는 접근법이다. 향후 연구방향으로는 운영체제의 취약성을 근본적으로 해결할 수 있도록 Secure Linux 개발이 필요하다.

7. 참고문헌

[1] 한국정보보호센터, 루트권한 획득 방지를 위한 Secure Linux 개발 보고서, 1999.9

[2] 이철원, 김홍근, 박태규,"리눅스 보안 연구개발 동향", 한국정보처리학회지, 한국정보처리학회, 1999. 11.

[3] 포항공과대학교 유닉스 보안 연구회, Security + for UNIX, 1999.

[4] http://kernelkorea.org