KIPS Tr. Comp. and Comm. Sys.
Vol.6, No.5 pp.247~254 pISSN: 2287-5891

https://doi.org/10.3745/KTCCS.2017.6.5.247

A Method of Detecting Real-Time Elevation of Privilege Security Module Using User Credentials

Sim Chul Jun[†] · Kim Won Il[††] · Kim Hyun Jung[†††] · Lee Chang Hoon[††††]

ABSTRACT

In a Linux system, a user with malicious intent can acquire administrator privileges through attack types that execute shells, and can leak important user information and install backdoor program. In order to solve this problem, the existing method is to analyze the causes of the elevation of privilege, fix the problems, and then patch the system. Recently, a method of detecting an illegal elevated tasks in which information inconsistency occurs through user credentials in real time has been studied. However, since this credential method uses uid and gid, illegal elevated tasks having the root credentials may not be detected. In this paper, we propose a security module that stores shell commands and paths executed with regular privileges in a table and compares them with every file accesses (open, close, read, write) that are executed to solve the case which cannot detect illegal elevated tasks have same credential.

Keywords : System Security, Elevation of Privilege Attack, Credentials

자격증명을 이용한 실시간 권한 상승 탐지 보안 모듈

심 철 준[†] · 김 원 일[††] · 김 현 정[†††] · 이 창 훈[††††]

요 약

리눅스 시스템에서 악의적인 목적을 가진 사용자는 쉘을 수행하는 공격 유형들을 통해 관리자 권한을 획득하여 백도어 설치 및 사용자들의 중요한 정보 등을 유출 시킬 수 있다. 이러한 문제점을 해결하기 위한 기존 방법은 권한상승원인을 분석하여 문제점을 수정한 후에 패치하는 방식을 사용하였다. 최근에는 사용자 자격증명을 이용하여 실시간으로 사용자 자격증명과 태스크 실행시 발생시키는 특징을 통해 정보의 불일치가 발생하는 태스크를 탐지하는 방법이 연구되고 있다. 그러나 이러한 자격증명 방법은 단순히 uid, gid를 이용하기 때문에 자격증명의 동일한 값을 가지는 공격유형은 탐지를 못하는 경우가 발생한다. 본 논문에서는 자격증명을 이용한 탐지 방법에서 탐지를 못하는 경우를 해결하기 위해 비정상적인 권한획득 정보보다 적은 정상적인 권한 획득 정보(자격증명) 및 권한을 획득할 수 있는 shell 명령어와 path등을 Table에 저장하여 관리하고 파일(open, close, read, write)접근마다 실시간으로 Table에 있는 정상적인 권한 정보와 비교하여 탐지하는 보안 모듈을 제안한다.

키워드 : 시스템 보안, 권한 상승 공격, 자격 증명

1. 서 론

현대 사회는 하드웨어와 네트워크의 급속한 발달로 무수히 많은 정보를 쉽고 편리하게 제공받을 수 있는 환경이 되었다. 이러한 환경은 각종 소프트웨어와 관련 기술들의 발전을 가속화시켜 휴대성, 편리성 등을 우선적으로 추구하게 만들었다. 또한 상대적으로 더딘 운영체제의 발전 속도를 대체하기 위한 별도의 프레임워크들의 등장을 가져왔다. 운영체제의 발전은 보안보다 기능을 지원하기 위한 형태로 발전하게 되었다. 결과적으로 다양한 요구와 기능을 만족하는 remix OS, polaris OS와 같은 형태의 운영체제들이 양산되는 한편, 보안성에서는 상대적으로 많은 취약점을 가지게 되었다[1, 2]. 또한 빠른 네트워크를 이용한 광대한 정보의 접근이 쉬워짐에 따라 보안 위험성도 증가하게 되었다[2, 3].

악의를 가진 공격자들은 시스템 정보나 개인 정보 등을 획득하기 위해 다양한 공격 방법을 이용하는데 크게 시스템 공격과 네트워크 공격으로 분류할 수 있다. 대표적인 네트워크 공격은 DoS 등을 들 수 있다. 이들 공격의 특징은 특

† 준 회 원 : 건국대학교 컴퓨터·정보통신공학과 박사
†† 비 회 원 : 마크애니 DRM 사업부 과장
††† 준 회 원 : 건국대학교 상허교양대학 소속 초빙교수
†††† 종신회원 : 건국대학교 컴퓨터·정보통신공학과 교수
Manuscript Received : January 23, 2017
First Revision : March 15, 2017
Accepted : March 28, 2017
* Corresponding Author : Sim Chul Jun(spsj1004@gmail.com)

정 시간에 폭발적인 패킷을 집중시켜 대상 네트워크가 허용하는 대역폭의 한계를 넘어 네트워크를 마비시키는 것이다. 반면 시스템 공격은 대상 시스템이 갖는 취약점에 대한 공격을 통해 특정 권한을 획득하여 시스템을 장악하고, 내부 정보를 유출시키는 것이다[4].

시스템 공격은 시스템을 장악하고, 내부정보를 유출한다는 점에서 공격의 위험성이 더 높다고 할 수 있다. 이에 본 논문은 시스템 공격에 대한 새로운 대응 방법을 제안하고자 한다.

시스템 공격에 대한 대응은 중요하다. 시스템 공격은 먼저 대상 시스템에 권한이 없는 일반 사용자 권한의 획득으로부터 시작된다. 일반 사용자 권한을 이용하여 시스템의 기초 정보를 수집하고 이에 대한 분석을 기초로 취약점을 도출한다. 도출된 정보를 이용하여 일반 사용자가 가질 수 없는 관리자 권한이나 그에 준하는 권한을 임의적으로 획득하는 것을 권한 상승 공격[5]이라고 한다. 권한상승 공격의 대표적인 방법으로 버퍼 오버플로우 공격[6, 7]이 있다. 권한 상승공격은 특정 연산의 경쟁 상태를 이용하거나 실행 불가능한 스택을 우회하는 등의 매우 다양한 방법이 존재하기 때문에 이와 관련된 내용을 모두 방어하는 것은 매우 어렵다. 이러한 공격은 주로 운영체제에서 일어나는 것이기에 권한 상승 공격방지에 대한 연구는 안드로이드, 리눅스 등 여러 종류의 운영체제에서 많이 연구가 되고 있다. 따라서 본 논문에서는 이러한 연구를 바탕으로 권한 상승 공격이 발생시키는 특징과 이에 대한 분석을 통해 권한 상승 공격에 대한 자격증명 및 권한을 획득할 수 있는 쉘 명령어와 명령어의 경로를 정리하고, 정리된 정보를 이용하여 권한상승공격을 탐지하고자 한다.

본 논문은 다음과 같이 구성했다. 2장에서는 권한 상승 공격에 대한 관련연구를 정리했고, 3장은 권한 탐지 기법 설계 및 구현을 하고, 4장은 실험 결과 및 시스템 부하 평가를 했다. 마지막으로 5장은 결론을 제시했다.

2. 관련 연구

2.1 권한 상승 공격

최근 유효한 권한 상승 공격은 권한기반 보안의 취약점을 이용한 공격으로 RTL(Return-into-libc)[3]과 이를 이용한 ROP (Return Oriented Programing)[8] 공격유형 등이 존재한다.

RTL공격은 사용자스택(non-executable stack)보호기법이나 IDS에서 네트워크를 통해 쉘 코드가 유입되는 것을 차단하는 보호 기법을 우회하기 위한 공격방법이다. 즉 이러한 공격은 버퍼를 오버플로우 시켜 버퍼위에 있는 return address를 lib 영역으로 조작하여 원하는 libc함수를 실행 하게 하는 공격이다. Fig. 1은 RTL공격 구조이다[3, 9].

ROP공격은 함수의 반환 명령어와 복귀 주소에 대한 처리를 이용한 공격이다. 이러한 공격은 프로그램의 코드에서 사용 가능한 가젯(Gadget)들을 찾고 조합하여 공격자가 원하는 기능을 수행한다. 가젯은 메모리나 레지스터의 값들을 조작하여 필요한 연산을 수행할 수 있는 작은 코드 덩어리들이다. 가젯의 끝은 함수 반환 명령(return)이나 간접 함수 호출(indirect call), 간접 점프 구문(indirect jump)과 같은 간접 분기 문으로 구성되며, 간접 분기문의 목적 주소를 조작해서 다음 가젯으로 실행 흐름을 변경 하여 관리자 권한의 쉘을 획득하는 방법이다[8, 10]. 즉 일반적인 함수 호출 구조와는 달리 분산된 공격 코드 위치로 반환이 이루어지도록 스택을 조작하고, 다음 공격 코드가 위치한 부분으로 임의 복귀를 수행하도록 구성하여 공격을 위한 전체 흐름을 만들어낸다. Fig. 2는 ROP공격 구조이다.

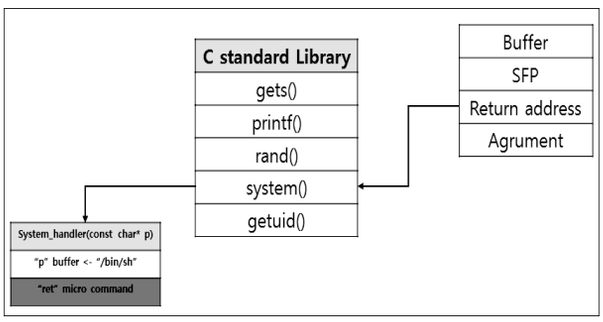


Fig. 1. RTL(return-into-libc) Attack

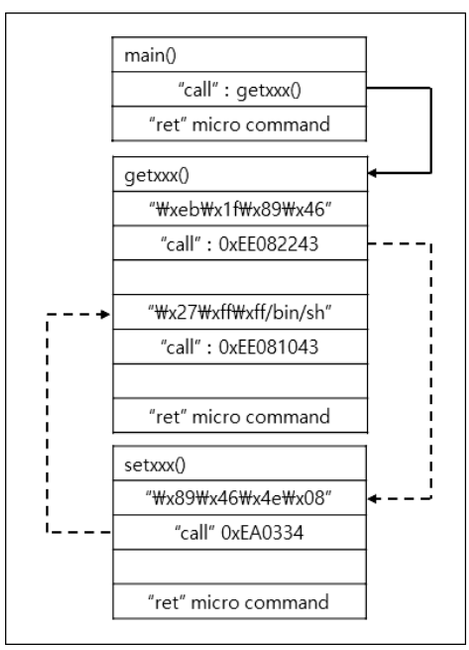


Fig. 2. ROP(Return Oriented Programing) Attack

2.2 권한 상승 공격 방지 연구

권한 상승 공격 방지 연구는 취약한 표준 라이브러리 함수 배제 및 컴파일러 확장을 통한 방어가 있으며[11], 최근에 연구된 사용자 자격 증명을 이용한 방법이 있다[1].

취약한 표준 라이브러리 함수 배제 방법은 개발 중 문제가 발생할 수 있는 코드를 사용하지 않으므로 공격의 발생 소지를 애초에 방지하는 방법이다[11]. 취약한 표준 라이브

러리 함수들은 CWE(Common Weakness Enumeration) 공식 사이트에 정리되어 있다[12]. 컴파일 확장을 통한 방법은 정적으로 사용자 스택이 변조되는 것을 방지하기 위한 형태로 구성되어 있다. 대표적으로 Stack Guard[4], Stack Shields[13], Po-Police[14], CRED(C Range Error Detection)[7]가 있다. 그리고 최근 연구된 사용자 자격 증명을 이용한 방법은 자격증명값 중 하나인 groups 정보가 불일치하는 태스크를 탐지한다[1].

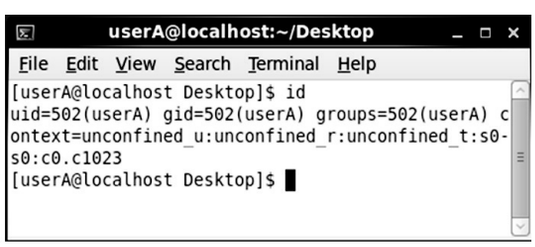
2.3 자격증명을 이용한 비정상 권한 상승 탐지

자격증명(Credentials)은 사용자가 시스템 자원에 접근하기 위한 모든 권한을 포함한 권한 집합체이며, 커널의 cred 구조체에 정의되어있다. 자격증명은 사용자가 인증된 후 한 번만 할당되고, 커널이 사용자의 요청을 처리하기 위한 특별한 경우만을 제외하고는 커널을 임의적으로 변경할 수 없다. 권한이 변경되는 특별한 경우는 시스템 호출로, 일반 사용자 권한으로는 시스템 자원을 접근할 수 없기 때문에 커널에서 일시적으로 권한을 상승시키고, 처리가 완료되면 원래의 권한으로 복원시킨다. 따라서 자격증명은 항상 동일한 값을 유지하도록 설계되어 있다[16].

일반적으로 사용자가 권한정보를 확인하는 방법은 리눅스에서 기본적으로 제공하는 “id”명령어를 이용하는 것이다. “id”명령어는 사용자의 자격증명을 지정 형식에 맞추어 출력해 준다. 출력되는 정보는 사용자의 권한을 구분하기 위해 uid, 소속된 그룹인 gid 그리고 여러 그룹에 소속된 경우 소속된 모든 그룹정보인 groups를 숫자 형태로 출력해 준다.

Fig. 3은 Cent OS에 일반 사용자 및 관리자가 시스템에 접속되었을 때 “id”명령어 실행 결과이다. 우선 Fig. 3(A)는 일반 사용자인 UserA의 권한을, Fig. 3(B)는 관리자인 root의 권한을 “id”명령어로 확인한 결과이며, 각 권한 값은 모두 동일한 값을 가지고 있음을 확인할 수 있다. Fig. 4는 일반사용자로 시스템에 접속된 후 권한 상승 공격을 통해 관리자 권한을 획득한 상태에서 실행한 “id” 명령어 결과이며,

(A)



(B)

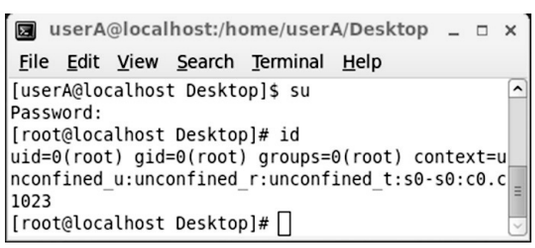


Fig. 3. Privilege Results Using “d” Command Normal State-(A) User, (B) Root

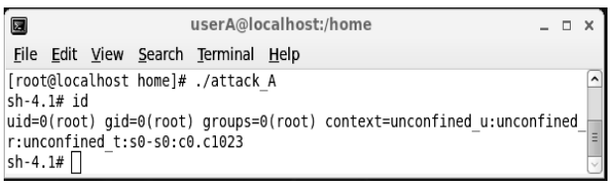


Fig. 4. Anomaly State-changed Root Authority by Attacker(attack_A)

이때 사용한 권한 상승 공격은 CVE-2013-2994[15](이하 권한 공격 A형)를 이용하였다. groups에 관리자 그룹인 0(root)과 공격을 수행한 사용자의 그룹인 502(UserA)가 같이 나타나는 것을 확인할 수 있다. 기존 연구는 이러한 권한 상승 공격에 의해 변경이 발생한 자격증명의 불일치를 이용하여 권한상승이 발생되었음을 탐지했다[1].

Fig. 5에서 보는 것과 같이 CVE-2013-2094[15](이하 권한 공격 B형)의 공격을 이용하여 권한 상승을 한 이후, “id” 명령어를 수행한 결과와 Fig. 3(B)의 결과가 완벽하게 일치하는 것을 볼 수 있다. 따라서 기존 연구의 자격증명 불일치를 이용한 탐지 방법으로는 권한 공격 B형과 같은 유형의 권한 상승 공격을 탐지할 수 없다는 문제점이 있다.

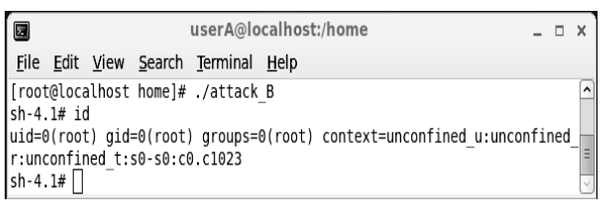


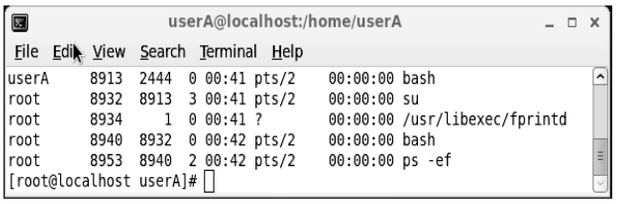
Fig. 5. Anomaly State-changed Root Authority by Attacker(attack_B)

3. 권한 탐지 기법 설계 및 구현

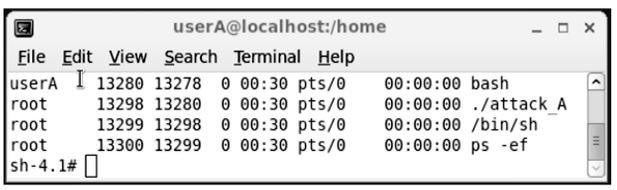
3.1 리눅스의 자격증명 과 태스크를 이용한 탐지 기법

2절에서 확인한 것과 같이 기존 연구에서는 공격을 통해 획득된 자격증명이 불일치하는 특징이 발생되지 않으면 공격을 탐지할 수 없다는 한계를 가진다. 본 연구에서는 이러한 기존 연구의 한계를 극복하기 위해 자격증명이 갖는 또 다른 특징 과 연계된 태스크의 실행 구조를 이용한 권한 상승 공격 탐지 방법을 제안한다.

리눅스의 모든 태스크 실행은 부모 태스크의 복사를 통해 이루어지고, 동일 프로세스의 생성이 아닌 새로운 태스크를 실행하기 위해서는 execXX() 계열의 시스템 호출을 이용한다. 새로운 태스크가 생성되면 task_struct 구조체를 통해 태스크의 모든 정보를 저장하고 관리한다[16]. task_struct에는 관련된 모든 태스크에 접근할 수 있는 리스트를 제공하며, 이를 통해 부모와 형제 및 자식 태스크까지 모두 접근이 가능하다. Fig. 6(A)는 “su”를 통해 실행된 태스크 리스트 구조를, Fig. 6(B)는 권한 공격 A형을 이용하여 실행된 태스크 리스트 구조를 나타낸다. 이를 통해 권한을 변경한



(A)



(B)

Fig. 6. (A) "su" Task Structure List, (B) "attack_A" Task Structure List

"su"와 공격 태스크인 "attack_A"이 모두 태스크 리스트에 나타나는 것을 확인할 수 있다. 즉, 대부분의 권한상승 공격이 관리자 권한의 쉘을 획득하는 것을 목표로 하는 특징[1, 17]이 태스크 리스트에 그대로 나타나는 것을 확인할 수 있다. 이를 바탕으로 현재 확인하려는 태스크를 비롯한 모든 부모 태스크의 권한을 확인할 수 있다.

기존 연구는 자격증명이 항상 동일한 값을 유지하는 특징을 이용 하였다면, 본 연구에서는 자격증명의 변경이 발생하는 조건과 전술한 태스크 구조를 이용한 추적을 통해 권한 상승 탐지를 수행한다. 먼저 리눅스에서 로그인한 사용자가 권한을 변경할 수 있는 유일한 방법은 "su"명령어를 이용하는 것이다[17]. "su"명령어는 시스템에 존재하는 여러 사용자로 현재 사용자를 일시적으로 변경해주며 이를 위해 변경하려는 사용자의 비밀번호 입력을 요구한다. 정상적으로 비밀번호가 입력되었다면 새로운 자격증명을 생성하고, 해당 사용자 권한을 할당하며, 사용자의 기본 쉘을 수행한다.

"su"명령어를 통해 관리자로 자격증명을 변경하는 과정은 Fig. 7과 같다. 일반 사용자인 userA의 기본 쉘인 "/bin/bash"은 해당 사용자의 자격증명인 "502"를 갖는다. 이때, "su"명령을 통해 정상적으로 관리자 권한으로 변경되면, "/bin/su"프로그램은 Fig. 6(A)와 같이 여전히 태스크 리스트에 나타난다. 이때 "su" 태스크의 자격증명은 관리자의 자격증명인 "0"으로 설정된다. 마지막으로 정상 실행된 "su"에 의해 새로운 쉘이 실행된다. 새로 실행되는 태스크는 부모 태스트인 "su"의 자격증명을 그대로 복사하기 때문에 관리자 권한 변경에만 해당하는 것이 아니라 "su"명령어를 통해 자격증명이 변경되는 모든 경우에 대해 동일한 과정을 거친다. 즉, 정상적인 자격증명의 변경 조건은 "su"태스크의 실행으로 한정지을 수 있으며, 권한이 변경된 태스크의 모든 부모 태스크 중에 반드시 "su"태스크가 실행 중이어야만 한다. 또한 발견된 "su" 태스크의 자격증명과 바로 아래 실행 중인 자식 태스크의 자격증명이 반드시 일치해야 한다. 본 연구에서는 살펴 본 바와 같이 특정 태스크의 모든 부모 태스크 중에 "su"가 존재하지 않는 상태에서 자격증명이 변경이 발생하였다는 것을 권한 상승 공격이 발생한 것으로 정의하며, 이러한 태스크의 탐지를 통해 권한 상승 공격의 발생 여부를 확인한다.

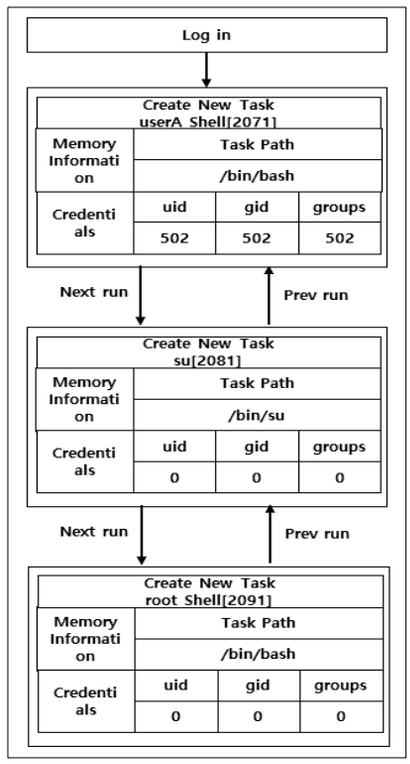


Fig. 7. "su" Command Flow

Fig. 8은 이러한 관점에서 권한 공격 A와 B의 실행에 따른 자격증명 정보를 커널에서 확인한 결과를 나타낸다. Fig. 8(A)의 userA 태스크는 "/bin/bash"의 태스크 경로를 갖고, 자격증명은 "502"값을 갖고 있다. userA 태스크에서 "attack_A" 프로그램을 이용하여 새로운 태스크를 생성하였으며, 이때 생성된 태스크의 경로는 "attack_A"이고, 자격증명값 중 groups값이 "0, 502"값을 갖게 된다. 마지막으로 "attack_A" 프로그램이 실행 된 후 비정상적인 권한 획득이 발생된 쉘을 갖게 되는데 이때 태스크의 경로는 "/bin/sh"이고, 자격증명 값 중 groups의 값이 "0, 502"가 된다. 그리고 Fig. 8(B)의 userA 태스크는 "/bin/bash"의 태스크 경로를 갖고, 자격증명은 "502"값을 갖고 있다. userA 태스크에서 "attack_B" 프로그램을 이용하여 새로운 태스크를 생성하였으며, 이때 생성된 태스크의 경로는 "attack_B"이고, 자격증명의 값은 "0"을 갖게 된다. 마지막으로 "attack_B"프로그램이 실행 된 후 비정상적인 권한 획득이 발생된 쉘을 갖게 되는데 이때 태스크의 경로는 "/bin/sh"이고, 자격증명 값은 "0"이 된다. 따라서 권한 상승 공격이 발생한 2가지 경우 모두 자격증명의 권한 값이 변경된 시점의 태스크가 "su"아닌 상태에서 변경이 발생하였다는 것을 확인할 수 있다.

공격 탐지를 위해 "su" 태스크를 구분하기 위해서는 반드시 전체 경로를 이용해야 한다. 그것은 커널에서 current 매

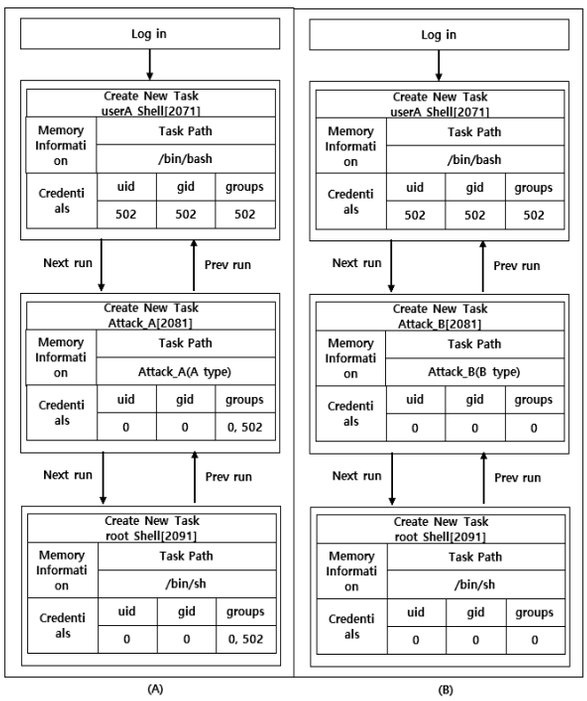


Fig 8. (A) A Type Attack Create Task, (B) B Type Attack Create Task

크로를 통해 획득할 수 있는 태스크 이름이 경로에 포함되지 않은 단순 실행 파일명만을 가지기 때문이다. 또 공격 태스크 이름이 “su”라면 공격 프로그램임에도 불구하고 정당한 태스크로 오인할 수 있기 때문이다. 따라서 해당 파일명을 포함하여 전체 경로를 획득해야만 정확한 “su”태스크의 여부를 확인할 수 있다.

3.2 제안된 권한 탐지 모듈 구성

권한 상승 공격에 의해 관리자 권한을 갖는 쉘이 실행되는 공격들은 3.1절에서 살펴본 바와 같이 자격증명의 변경이 발생하는 부모 태스크의 추적을 통해 탐지를 수행한다. 즉 일반사용자가 정상적인 권한획득을 통해 태스크 정보 중 자격증명과 실행 파일명을 포함한 태스크 경로를 Table 1과 같이 정리하여 탐지를 위한 기준으로 사용한다.

정상적인 권한상승을 통해 쉘을 획득하는 방법은 “su”명령어를 이용하는 방법이 유일하기 때문에 “su”명령어로 권한이 상승된 후 태스크의 정보 중 자격증명의 값과 태스크 경로를 Table 1과 같이 저장하여 관리를 한다. 태스크의 경로는 init, /bin/su, /bin/sh, /bin/bash등 있으며, init는 부모 태스크의 추적을 종료하기 위해 태스크의 시작임으로 최상위 부모이고, 쉘은 공격을 통해 실행할 수 있는 모든 종류의 쉘을 정리한 것이다. 마지막으로 “/bin/su”는 정상적인 자격증명 변경 태스크 탐지를 위해 포함하였으며, 권한 상승 시 사용되는 명령어이다. 자격증명 값은 리눅스 배포판마다 다르게 설정되어 있다. Cent OS, Fedora, Redhat계열은 사용자 자격증명 값이 500이상부터 사용되고, Debian, Ubuntu계열의 사용자 자격증명 값은 1000이상부터 사용된다[18]. 본 논문에서는 Cent OS를 이용하므로 각 값은 500으로 설정하였다.

Table 1. Credentials Info and Task Path Table

Type	Credentials Info(Threshold)			Task Path
	uid	gid	groups	
Value	fixed	fixed	fixed	init
	<500	<500	<500	/bin/sh
				/bin/bash
				/bin/csh
				/bin/ksh
				/bin/zsh
				/bin/tcsh
	all	all	all	/bin/su

이렇게 생성된 Table 1은 비정상적으로 권한이 상승되었는지 판단하는 기준으로 사용된다.

태스크의 실행은 대부분 execve()를 이용하고, 자원에 대한 접근은 파일로 추상화되어 처리된다. 결국 자원에 대한 접근은 파일처리를 위한 시스템호출을 이용하게 된다. 따라서 태스크를 실행하는 execve()와 파일처리를 위한 open(), read(), write(), close() 등 시스템호출을 후킹하는데 사용된다. Table 2는 래핑을 수행하는 시스템 호출과 목적을 나타낸다.

Table 2. System Call

System Call	Detail
execve()	- Run Task. - Executing tasks through shell command and code
open()	- Requests for access to resources.
read()	- Loads resources into information.
write()	- Write for information.
setgroups()	- System calls for module evasion.

Fig. 9는 제안된 권한 탐지 모듈의 동작 원리를 크게 시스템 호출 후킹 단계와 검증 단계로 구성하였다. 시스템 후킹 단계는 시스템 Table 2에 정의된 시스템 콜들을 후킹하여 시스템 콜에 발생 여부를 확인하는 단계이다. 만일 시스템 콜이 발생되었으면 검증 단계에서 비정상 권한상승이 발생되었는지 확인한다. 검증 방법은 1차, 2차 검증 방법을 이용하였다. 1차 검증 단계에서는 신규로 생성된 태스크의 정보인 자격증명 정보와 Table 1에 정의된 자격증명 값을 확인 하여 불일치일 경우 권한 상승으로 보고, 불일치하지 않으면 2차 검증을 한다. 2차 검증에서는 태스크 경로를 비교한다. 부모의 태스크 경로와 Table 1의 태스크 경로를 비교하여 불일치 시 비정상 권한 상승으로 판단한다. 이러한 검증을 이용한 탐지 방법으로 본 연구는 이전 연구가 탐지하지 못했던 정상적인 자격증명과 동일한 권한 상승을 탐지하게 되었다.

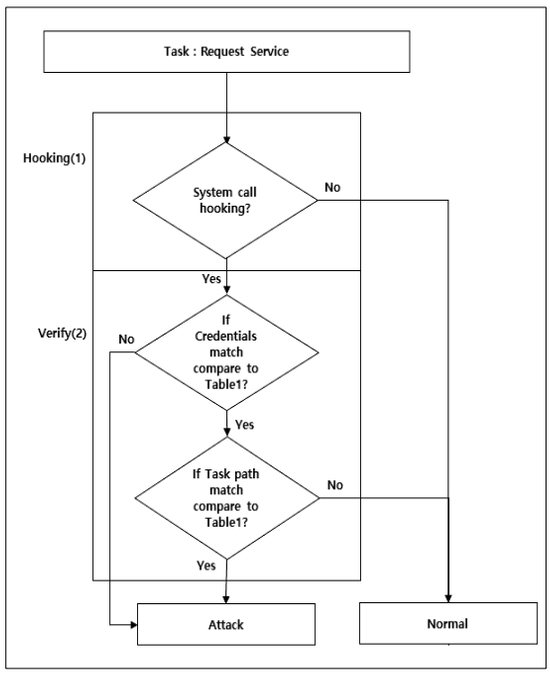


Fig. 9. Module Process Flow

4. 실험 결과

4.1 탐지 실험 결과

권한 상승 탐지가 정상적으로 수행되는지 여부를 확인하기 위해 권한 상승 공격 A와 권한 상승 공격 B를 이용하여 실험을 수행하였다. 실험에는 인증을 정상적으로 수행하는 일반 사용자 쉘과 관리자 쉘을 동시에 실행하여 공격에 의한 쉘의 동작과 다중 프로세스환경에서 탐지여부를 확인하였다. 비정상 권한 획득 탐지를 위한 실험 환경은 다음과 같다.

CPU : Intel Core i5
OS : Cent OS 6.3
Kernel : 2.6.32-279

일반 사용자는 user A(uid:502, gid:502: groups:502), user B(uid:508, gid:508, groups:508)로 하였다. A형 공격은 자격증명의 groups가 불일치하는 공격 유형을, B형 공격은 자격증명의 관리자 권한과 동일한 유형을 사용하였다. 탐지하기 위한 테스트 Case는 다음과 같다. 이때 공격에 의한 권한상승은 “->>”으로 표기하였고, 정상적인 권한상승은 “->”으로 표기하였다.

Case 1 userA ->> root
Case 2 userA -> userB ->> root
Case 3 userB ->> root -> root

Table 3는 A형 공격 프로그램이고, Table 4은 B형 공격 프로그램을 이용한 실험 결과이다.

Table 3. A Type Attack Program TEST Result

Test Case		Attack Program	Result Values				Detect Result
			uid	gid	groups	task	
Case1	userA ->>root	attack_A	0	0	0, 502	attack_A	success
Case2	userA ->userB ->>root	attack_A	0	0	0, 502	attack_A	success
Case3	user ->>root ->root	attack_A	0	0	0, 502	attack_A	success

Table 4. B Type Attack Program TEST Result

Test Case		Attack Program	Result Values				Detect Result
			uid	gid	groups	task	
Case1	userA ->root	attack_B	0	0	0	attack_B	success
Case2	userA ->userB ->>root	attack_B	0	0	0	attack_B	success
Case3	userA ->>root ->root	attack_B	0	0	0	attack_B	success

Table 3과 Table 4에 나타난 결과 중 Table 4는 자격증명이 모두 동일한 값을 가지고 있어도 탐지되었음을 보여주고 있다. 이 실험을 통해 본 논문에서 자격증명과 태스크 경로를 Table 1에 정의된 내용과 비교하여 탐지함으로써 기존 연구에서 탐지가 불가능한 공격유형도 탐지하였다.

4.2 시스템 부하 평가

제안 모듈은 실시간으로 권한상승을 탐지하기 때문에 시스템적으로 부하가 발생되는지 시스템 부하 평가를 하였다.

모듈의 시스템 부하 평가는 리눅스 시스템 전체 디렉터리에 존재하는 전체 파일에 접근하기 위해 open(), close() 그리고 read()의 시스템호출을 사용하였고, 이에 따른 경과 시간을 평가 기준으로 삼았다. Fig. 10은 user 권한에서 모듈이 적재된 상태와 적재되지 않은 상태의 시스템 부하 테스트 결과이고, Fig. 11은 root 권한에서 모듈이 적재된 상태와 적재되지 않은 상태의 시스템 부하 테스트 결과이다. 마지막으로 Fig. 12는 Attack을 통해 root권한을 획득한 상태에서 모듈이 적재된 상태와 적재되지 않은 상태의 시스템 부하 테스트 결과이다. 모듈은 적재하지 않은 경우와 적재한 경우를 각각 100회씩 반복 수행하여 집계하였다.

Table 5와 같이 user권한에서 모듈이 적재되지 않은 상태에서는 1415msec이고, 모듈이 적재된 상태에서는 1477msec가 걸려 62msec의 차이가 발생하였다. 그리고 root권한에서 모듈이 적재되지 않은 상태에서는 1654msec이고, 모듈이 적재된 상태에서는 1736msec가 걸려 87msec의 차이가 발생하였다. 마지막으로 Attack을 통해 root권한을 획득한 상태에서 모듈이 적재되지 않은 상태에서는 1637msec이고, 모듈이 적

재된 상태에서는 1726msec가 걸려 89msec의 차이가 발생하였다. 시스템 콜을 후킹함으로써 모듈이 적재된 상태에서 평균적으로 79msec의 차이가 발생하였다. 기존 연구는 Open, Close만을 사용하였고, 평균 40msec가 걸렸다[1]. 본 논문은 Open, Read, Close를 사용하였기 때문에 기존 논문의 파일 I/O가 본 논문의 파일 I/O 보다 사용 빈도가 낮다. 따라서 기존 논문과 비교하였을 때 시스템호출 후킹을 통한 정보수집 및 탐지가 시스템에 미치는 영향이 미미함을 알 수 있다.

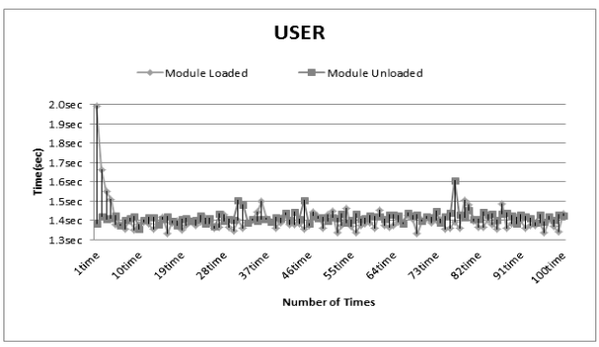


Fig. 10. User Module Unloaded/Loaded

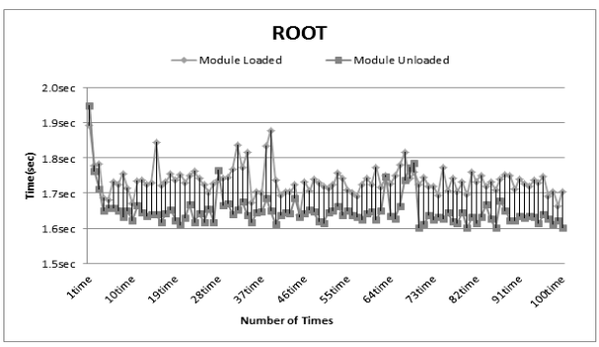


Fig. 11. Root Module Unloaded/Loaded

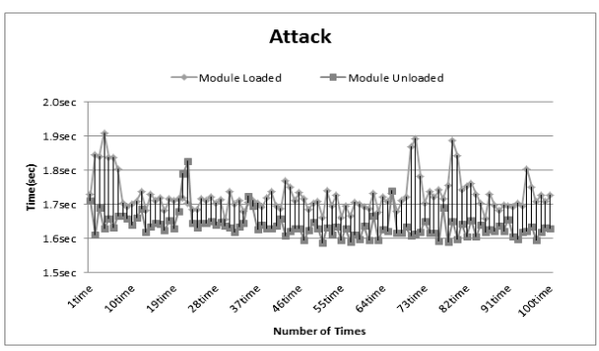


Fig. 12. Attack Module Unloaded/Loaded

Table 5. Compare Loaded and Unloaded Module

Type	User (msec)	Root (msec)	Attack (msec)
Module Unloaded	1415	1649	1637
Module Laded	1477	1736	1726
Difference	62	87	89

5. 결 론

리눅스 OS에서 권한상승 공격에 대한 대응 방안은 권한상승 공격에 대한 분석을 통해 OS을 패치하는 방법만 존재하였으나, 최근 연구에 따르면 자격증명을 이용하여 실시간으로 권한상승을 탐지하는 방법이 제안되었다. 이러한 방법은 기존 방법보다 실시간으로 탐지가 가능하여 권한상승에 대한 대처를 빠르게 할 수 있다. 그러나 공격을 통해 정상권한과 동일한 형태의 권한이 획득된 경우 탐지할 수 없다는 단점을 가지고 있다. 이에 본 논문이 제안한 권한상승 실시간 탐지 모듈은 권한 획득과 관련된 자격증명 정보와 실행되는 태스크 경로를 Table 1 형식으로 저장하고, 시스템 콜 후킹을 통해 자격증명 값과 태스크 경로를 비교하여 권한 상승 여부를 탐지하였다. 실험 결과, 제안한 권한상승 실시간 탐지 모듈을 시스템에 적재 하였을 때, 파일 Open, Read, Close의 평균 수행 속도는 1646msec이고 적재되는 않은 상태에서는 평균 1567msec로 79msec의 차이가 발생하기 때문에 시스템에 큰 부하가 없었다. 또한 기존 연구가 탐지할 수 없는 정상 자격증명을 획득하도록 구성되어 있는 공격유형도 탐지할 수 있었다. 이러한 자격증명을 이용한 권한 상승 탐지 연구는 자격증명의 정보를 활용하는 리눅스뿐만 아니라, 더 나아가 리눅스를 기반으로 하는 다양한 운영체제에 적용할 수 있도록 더 많은 연구가 향후 진행되어야 할 것이다.

References

[1] Kim Won-il, Yoo Sang-Hyun, Kwak Ju-Hyun, and Lee Chang_hoon, “A Study for Task Decetion Acquiring Abnormal Permission in Linux,” *KIPS*, Vol.3, No.11, pp.427-432, 2014.

[2] A. Johri and G. L. Luckenbaugh, “Trusted path mechanism for an operating system,” U.S. Patent, No.4,918,653, 1990.

[3] M. Tran et al., “On the expressiveness of return-into-libc attacks,” in *International Workshop on Recent Advances in Intrusion Detection*, Springer Berlin Heidelberg, pp.121-141, 2011.

[4] C. Cowan et al., “StackGuard : Automatic adaptive detection and prevention of buffer-overflow attacks,” *Proceedings of the 7th USENIX Security Symposium*, Vol.81, 1998.

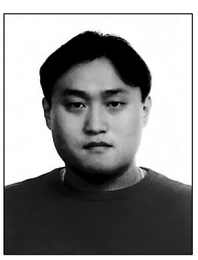
[5] One Aleph, “Smashing the stack for fun and profit,” *Phrack Magazine,* Vol.7, No.49, pp.14-16, 1996.

[6] Jeong Min Lee, Hyun Wook Kim, and Woo Hyun Ahn, “BinaryReviser: A Study of Detecting Buffer Overflow Vulnerabilities Using Binary Code Patching,” *Korea computer congress,* Vol.38, No.1, pp.122-125, 2011.

[7] Ruwase Olatunji and Monica S. Lam, “A Practical Dynamic Buffer Overflow Detector,” *NDDS,* 2004.

[8] Kim Ju-Hyuk and Oh Soo-Hyun, “Detection in Stack region,” *Journal of the Korea Academia-Industrial Cooperation Society*, Vol.15, No.5, pp.3132-3131, 2014.

[9] Pax Team, “address space layout randomization(ASLR)” [Internet], htte://pax.grsecurity.net/docs/aslr.txt.
[10] Hilmi Ozdoganoglu, et al., “SmashGuard: A hardware solution to prevent security attacks on the function return address,” *IEEE Transactions on Computers*, Vol.55, No.10, pp.1271-1285, 2006.
[11] Mark G. Graff and Kenneth R. van Wyk, “Secure coding: principles and practices,” O’Reilly Media. Inc., 2003.
[12] Common Weakness Enumeration [Internet], http://cwe.mitre.org/ (2014. 10. 29.).
[13] Vendicator. Stack Shield, “A stack smashing technique protection tool for Linux,” [Internet], http://www.angelfire.com/sk/stackshield/info.html (2000).
[14] Etoh. Hiroaki, “GCC extension for protecting applications from stack-smashing attacks (propolice)” [Internet], http://www.trl.ibm.com/projects/security/ssp (2003).
[15] Common Vulnerabilities and Exposures [Internet], http://cve.mitre.org/ (2014. 10. 29).
[16] CREDENTIALS IN LINUX [Internet], http://www.kernel.org/doc/Documentation/security/credentials.txt.
[17] The su Command [Internet], http://www.linfo.org/su.html.
[18] The Linux Information Project, “User ID Definition” [Internet], http://www.linfo.org/uid.html (2014. 10. 1).



심 철 준
e-mail : spsj1004@gmail.com
2011년～현 재 건국대학교 컴퓨터·정보통신공학과(박사)
2003년～현 재 아이지코 연구소 소장
관심분야 : 시스템보안, 영상처리, 증강현실

김 원 일
e-mail : wikim@markany.com
2014년 건국대학교 컴퓨터·정보통신공학과(박사)
2015년～현 재 마크애니 DRM 사업부 과장
관심분야 : 보안, 시스템 프로그래밍, 운영체제

김 현 정
e-mail : nygirl@konkuk.ac.kr
2015년 건국대학교 컴퓨터·정보통신공학과(박사)
2016년～현 재 건국대학교 상허교양대학 소속 초빙교수
관심분야 : 보안, 인공지능, 영상인식

이 창 훈
e-mail : chlee@konkuk.ac.kr
1996년～2002년 건국대학교 정보통신원장
2001년～2002년 건국대학교 정보통신대학과 학장
1980년～현 재 건국대학교 컴퓨터·정보통신공학과 교수
관심분야 : 인공지능, 운영체제, 정보보안