리눅스 Netfilter 프레임웍을 이용한 동적 침입 방지 기법

조은경, 고광선, 강용혁, 임현근, 엄영익
성균관대학교 정보통신공학부
e-mail : {iuno, yhkang, hklim, rilla91}@dslab.skku.ac.kr
yieom@ece.skku.ac.kr

Dynamic Intrusion Prevention Scheme based on Linux Netfilter Frameworks

Eun-kyung Cho, Kwang-sun Ko, Yong-hyeog Kang, Hyun-kun Lim, Young Ik Eom
School of Information and Communication Engineering, Sungkyunkwan University

요 약

현재 사용되고 있는 침입 탐지 시스템들은 새로운 공격 패턴을 가진 침입을 탐지하기 위하여 규칙 데이터베이스를 지속적으로 갱신하고 있다. 그러나 이러한 침입 탐지 능력은 현재 알려진 기술이나 기법들에 한정된다. 따라서, 기존 침입 탐지 시스템들은 공격을 받은 후, 현재까지 존재하지 않았던 새로운 기법이 적용된 공격이나 다소 변형된 공격을 받게 되면 침입을 탐지하지 못하거나 능동적 대처를 하지 못하였으며, tcpdump 또는 libpcap과 같은 응용 레벨의 프로그램을 사용하였기 때문에 실시간으로 패킷들을 감시할 수 없었다. 이러한 추가적인 공격들을 탐지하기 위해서 보다 강력하고 능동적인 네트워크 기반의 대응 기술이 요구되는데, 이에 본 논문에서는 이러한 요구 사항을 해결하기 위하여 패킷 감시의 정확성과 침입에 대한 실시간 대응성을 높이는 커널 레벨에서 동작하는 침입 탐지 모듈과, 악의적인 목적을 가진 패킷들을 차단하는 필터링 모듈 개발 기법을 제시하고자 한다.

1. 서론

현재 네트워크상의 침입을 방지하기 위한 시스템은 크게 두 가지로 구분할 수 있는데, 침입 징후를 탐지하기 위한 침입 탐지 시스템과 탐지된 비정상적인 침입 트래픽을 차단하기 위한 침입 대응 시스템이 그것이다[1].

최근 개발되는 보안 시스템들은 이 두 가지 시스템들이 상호 결합된 형태가 주를 이루고 있지만, 보안 관리 대상이 특정 시스템으로 한정되어 있기 때문에 악의적인 사용자가 외부에서 새로운 기술이나 방법을 적용하여 제2, 제3의 공격을 할 경우 침입을 탐지 하기 어렵다는 보안의 취약성을 지니고 있다.

따라서 기존의 침입탐지 시스템들은 갈수록 다양해지는 침입에 대해 능동적으로 대처하는데 어려움이 많으며, 대규모 네트워크 환경에서의 효율적인 탐지에 적합하지 않은 구조를 지니고 있다[2].

그러므로 이를 고려한 새로운 형태의 침입 탐지 시스템 구조를 필요로 하게 되는데, 이 필요성으로부터 본 논문에서는 분산 환경 안에서 동작하는 침입 방지 관리 모델을 기반으로, 패킷 감시의 정확성과 침입에 대한 실시간 대응성을 높이기 위한 커널 레벨의 침입 징후 포착 기술을 제시하고자 한다.

아울러 포착된 침입 징후에 능동적으로 대응하기 위한 필터링 기반의 접근 제어 기술과 침입 탐지 정보 관리 기술에 관한 연구도 병행하여 수행하며 더 나아가 라우팅 기능을 이용하여 다른 보안 구성 요소와의 연계를 통합으로써 침입 대응 기술을 네트워크 기반으로 발전 시키는 기법을 제시한다.

※ 본 논문은 대학 IT연구센터 육성□지원사업의 연구결과로 수행되었음

본 논문은 2장에서 기존의 침입 탐지 및 침입 방지 시스템에 대한 관련 연구를 제시하고, 3장에서 본 논문에서 제시할 시스템의 구성 및 알고리즘을 설명하며, 4장에서 실제 구현된 침입 방지 시스템의 프로토타입을 이용하여 패킷에 대한 침입 탐지 및 능동적인 대처가 가능한지에 대한 실험 및 결과를 분석한다. 마지막으로 5장에서 제시한 기법들에 관한 결론 및 향후 연구로 마무리 짓고자 한다.

2. 관련연구

본 절에서는 네트워크 기반에서 침입을 탐지하고 방지하는 시스템을 개발하는데 필요한 기존 연구들을 살펴보도록 한다.

2.1 침입 탐지 시스템의 분류

침입 탐지 시스템은 크게 네트워크 기반 침입 탐지와 호스트 기반 침입 탐지 그리고 이 두 기법이 모두 적용된 혼합형 침입 탐지로 나누어 진다. 네트워크 기반 침입 탐지는 네트워크를 통해 전송되는 트래픽을 검사하여 침입을 탐지하는 반면 호스트 기반 침입 탐지는 로컬 호스트에서 사용자의 행위나 프로세스들을 검사하여 침입을 탐지한다.

2.2 침입 탐지 기법의 분류

침입 탐지 기법에는 두 개의 상보적인 흐름이 존재하는데, 이는 각각 오용 탐지 기법(Misuse Detection Model)과 비정상 행위 탐지 기법(Anomaly Detection Model)으로 알려져 있다[3].

오용 탐지 기법은 지식 기반 침입 탐지 기법으로, 알려진 침입 행위에 관한 축적된 지식을 이용하여 정해진 모델과 일치하는 경우를 침입으로 간주한다. 이를 구현하기 위하여 전문가 시스템(Expert System), 시그너처 분석(Signature Analysis), 페트리넷(Petri-net), 상태전이 분석(State Transition Analysis), 신경망(Neural Network), 유전 알고리즘(Genetic Algorithm)과 같은 방법들을 적용한 기법이 존재하며 이러한 방식들은 알려진 침입 기법에 대해서는 비교적 높은 정확성으로 침입을 탐지하지만 새로운 패턴의 공격들은 탐지하기 힘들다는 단점이 존재한다.

이에 반하여 비정상 행위 탐지 기법은 기존의 정상적인 행위에 대한 참조 모델을 생성한 후 그 행위에서 벗어나는 경우가 있을 경우 이를 침입으로 간주함으로써 새로운 공격에 대한 탐지는 가능하게 되나 공격과 정상행위를 구별하기 위한 특정한 임계값을 설정하기 힘들다는 어려움이 존재한다. 이를 구현하기 위하여 통계적(Statistical) 방법, 전문가 시스템(Expert System), 신경망(Neural Network), 컴퓨터 면역학(Computer Immunology), 데이터 마이닝(Data Mining), HMM(Hidden Markov Model), 기계학습(Machine Learning)등을 적용한 기법들이 있으며, 침입 탐지를 위한 분석을 수행하는 방식에 따라 다시 정적 및 동적 침입 탐지로 나눌 수 있다. 정적 침입 탐지 시스템은 시스템의 스냅샷을 잡아서 분석한 후 취약한 소프트웨어나 구성 오류 등을 찾으며, 동적 침입 탐지 시스템은 시스템에 영향을 미치는 이벤트를 발생하는 즉시 획득함으로써 실시간 분석을 수행한다는 특징이 있다.

본 논문에서는 시그너처 분석 방법을 적용시킨 오용 탐지 기법과 통계적 방법을 적용시킨 비정상 행위 탐지 기법을 상황에 따라 혼합하여 구현함으로써 동적 침입 탐지의 정확성을 높이고자 한다[4,5,6].

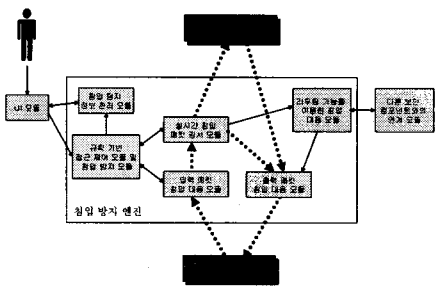
2.3 커널 기반 침입 탐지 기법의 구현

본 논문에서 제시하는 침입 방지 및 관리 기술은 커널 모듈을 기반으로 구현된다. 커널 모듈은 리눅스상에서 다양한 하드웨어 장치를 효율적으로 지원하기 위한 기술로서, 시스템이 동작하고 있는 상태에서도 디바이스 드라이버나 파일 시스템 등을 커널의 수정 없이 필요할 때마다 동적으로 추가할 수 있다는 장점이 있다[7].

따라서 침입 방지 및 관리 시스템을 모듈로 구현하여 커널에 삽입하면 시스템의 전체적인 가용성을 증대시킬 수 있을 뿐 아니라 프로그램 실행 속도를 보다 향상시킬 수 있다.

3. 리눅스 커널 기반의 동적 침입 관리 기법

본 논문에서 제시하고자 하는 분산 환경하의 침입 방지 및 관리 시스템은 그림 1과 같이 구성되어 있다.



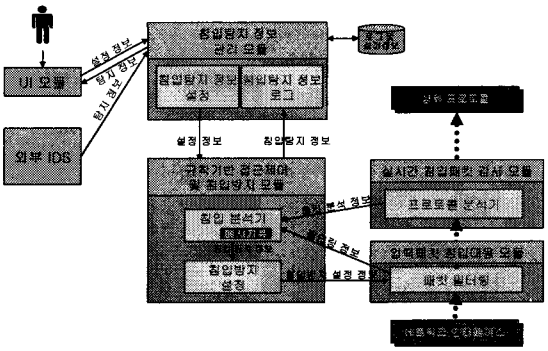
(그림 1) 분산 환경의 침입 방지 및 침입 관리 모델

침입 방지 엔진으로부터 실시간 패킷 감시를 통하여 포착된 침입 패킷들은 동적 침입 대응 모듈을 통하여 네트워크 인터페이스로의 침입이 차단되며 이 패킷들에 관한 정보는 다시 해당 네트워크의 라우팅 정책에 반영됨으로써 네트워크 레벨의 침입 관리가 이루어지게 한다.

최근의 침입 탐지 시스템들은 글로벌 네트워크에서의 상호 협력을 중시하는 경향이 있으며, 이를 통해서 탐지할 수 있는 침입의 유형 및 운영상의 장점 등이 활발히 논의되고 있는 추세이다. 따라서 그림 1 의 침입 방지 및 관리 모델에는 다른 보안 요소들과의 연계를 위한 동작이 반영되었다.

3.1 침입 방지 엔진의 기본 동작 및 알고리즘

그림 1에서 제시한 침입 방지 및 침입 관리 모델 중 침입 방지 엔진을 구성하고 있는 세부 모듈들을 살펴보면 그림 2와 같다.



(그림 2) 침입 방지 엔진의 각 대응 모듈

침입 방지 엔진에 속한 각 대응 모듈들의 동작 원리 및 알고리즘은 다음과 같이 제시된다.

3.1.1 입력 패킷 침입 대응 모듈

리눅스 내에서 자체적으로 제공하는 패킷 필터링 설정 응용 프로그램인 iptables를 직접 커널 내에서 함수로 호출하는 방식을 이용하여 침입 가능성이 있는 패킷들을 차단하는 ruleset을 설정한다.

설정된 ruleset을 이용하여 비정상적인 패킷의 진입은 1차적으로 차단되며, 필터링 된 패킷들에 대한 정보는 메시지 형식으로 작성되어 침입 분석기의 메시지 큐로 전송된 후 패킷 필터링에 대한 로그 정보를 작성하는 데 사용된다.

protocol	filtering type	dest port num	src IP addr
Start time	end time	counter	total of len

(그림 3) 필터링 정보에 대한 메시지 형식

표 1은 패킷 필터링 모듈에서 침입 분석기 모듈의 메시지 큐로 전송되는 패킷 필터링 정보에 대한 메시지 형식을 보여주고 있다. 현재 필터링 된 패킷이 사용하는 프로토콜 및 근원지 주소, 목적지 포트 번호등과 같은 기본적인 헤더 정보 이외에 필터링 된 시간과 몇 개의 패킷들이 필터링 되고 있는지(counter)와 같은 로그 정보와 관련된 필드들이 포함된다.

3.1.2 실시간 침입 패킷 감시 모듈

프로토콜 분석기는 상위 프로토콜로 전송되는 모든 패킷의 헤더 정보를 읽어서 침입을 탐지하는데 필요한 메시지 형식으로 작성한 후 침입 분석기의 메시지 큐로 전송하며, 메시지 큐에 전송된 정보들은 침입 분석기가 주기적으로 읽어가 침입을 탐지한다.

protocol	src port num	dest port num	sequence num
ack num	TCP flag	src IP addr	dest IP addr
IP option	TTL	TOS	arrival time

(그림 4) 메시지 큐로 전송되는 패킷 정보의 형식 (TCP 패킷의 경우)

이러한 작업들은 리눅스 네트워크 디바이스를 통하여 수신 받은 패킷들을 처리하는 기본 구조인 netfilter 프레임웍을 통하여 이루어 지는데, 이는 패킷을 라우팅 전과 후(NF_IP_PRE_ROUTING, NF_IP_POST_ROUTING), 포워딩된 경우(NF_IP_FORWARD), 네트워크 디바이스에 들어온 경우와 나가는 경우(NF_IP_LOCAL_IN, NF_IP_LOCAL_OUT)등으로 나누어 임의의 특정 함수를 등록한 후 추가적인 패킷 처리 작업을 수행할 수 있도록 지원한다.

```
process_filtered_packet(all packet in network device)
{
   nf_register_hook(&process_packet_local_in);
   nf_register_hook(&process_packet_pre_routing);
   nf_register_hook(&process_packet_forward);
}
```

(그림 5) 네트워크 디바이스에 수신된 패킷들을 처리하기 위한 함수 등록

또한 netfilter 프레임웍이 패킷들을 처리하는 과정에는 공개된 침입 유형에 해당하는 패킷들을 탐지하고 방어하기 위한 오용 탐지 기법이 적용된다. 이는 침입으로 간주되는 시그너처를 추가함으로써 동일한 패턴을 가지는 해당 패킷이 차단되도록 한다.

```
case TCP_PACKET:
{
   /* Land Attack! */
   if(packet->source == packet->dest)
        filter off bad packet ;
}
```

(그림 6) 공개된 침입 유형의 패킷 차단을 위한 오용 탐지 기법의 적용 (Land Attack의 경우)

tcpdump 또는 libpcap과 같은 프로그램을 이용하여 패킷을 분석할 경우 네트워크 상에 존재하는 패킷들이나 시스템에 수신된 패킷들의 복사본을 이용하여 침입을 분석하기 때문에 침입이 발생하였을 경우 즉각적인 대응이 어렵다.

그러나, netfilter 프레임웍은 시스템이 패킷을 처리하는 과정에 직접 참여하여 침입 발생 여부를 확인할 수 있기 때문에 침입에 따른 즉각적인 대응이 가능하다는 장점이 있다.

3.1.3 규칙 기반 접근 제어 및 침입 방지 모듈

침입 분석기는 메시지 큐로부터 주기적으로 메시지를 읽어온 후, 침입 유형에 맞게 침입 상태 정보를 갱신하며, 침입을 탐지했을 경우 침입 방지 설정모듈로 탐지한 침입 정보를 넘겨준다.

type	level	data

(그림 7) 침입 정보 메시지 형식

침입 정보의 type에는 침입 방지 설정을 하기 위하여 어떤 종류의 data를 제한할 것인가에 대한 정보가 포함된다. 즉 침입 호스트를 차단할 것인지, 침입 패킷이 들어오는 해당 서비스의 포트를 차단할 것인지와 같은 침입 방지 정책에 따라 type에는 차단할 data의 형식을 구분하기 위한 값(예: PORT, SRC_ADDR)이 오며 data에는 해당 형식의 구체적인 값(예: 111.111.111.111)이 전달된다. level에는 탐지된 침입을 즉시 차단해야 하는지, 경고 메시지 형식으로 시스템에 기록해야 할 것인지와 같이 침입 대응 정도를 결정하기 위한 레벨 정보가 포함된다.

침입 분석기에서는 프로토콜 분석기와는 다르게 통계적인 방식을 이용한 비정상 행위 탐지 기법을 적용하여 미리 실험을 통하여 결정된 특정 임계값의 범위를 초과하는 경우를 침입으로 간주한다.

```
for(each packet in message queue)
{
  if ( current_pkt->ipcnt >= PKT_FLOOD_ATTCK_THRESHOLD )
       intrusion_reactor( addr, realtime, current_pkt->src_addr);
}
```

(그림 8) 비정상 행위 탐지 기법을 적용한 침입 탐지 (Packet Flooding Attack의 경우)

침입 방지 설정모듈은 침입분석기와 외부 IDS에서 오는 침입 탐지 정보를 이용하여 침입 방지를 위한 ruleset을 작성하며, 작성된 ruleset을 능동적 침입 차단을 위한 패킷 필터링 모듈로 전달한다.

```
intrusion_reactor(type, level, data);
{
   src_addr = data;
   char  *cmd[] = {"/usr/local/sbin/iptables", "-A", "INPUT", "-s",
   src_addr,, "-j", "DROP", NULL};
   set_fs(KERNEL_DS);
   execve(cmd[0], cmd, NULL);
}
```

(그림 9) 침입 차단을 위한 패킷 필터링 모듈의 호출 (침입으로 간주된 패킷의 근원지 호스트를 막는 경우)

또한 패킷 필터링 모듈에 의해 차단된 호스트가 이 후 일정 기간 동안 침입으로 간주되는 패킷을 보내지 않을 경우 다시 패킷을 보내는 것을 허용함으로써 침입 탐지 및 차단의 오판 가능성에 능동적으로 대비할 수 있도록 하며 이는 침입 호스트를 차단하기 위해 해당 서비스를 막아 놓는 경우에도 동일하게 적용된다.

위의 과정을 통하여 실시간 침입 방지 모델로 전달된 침입 패킷의 정보는 시스템이 속해 있는 네트워크의 라우터로 전송됨으로써 다른 시스템의 보안 구성요소들과 연

계하여 네트워크 수준의 침입방지 및 관리가 이루어지게 한다. 침입으로 간주된 비정상 패킷의 라우팅 우선순위를 낮추는 것과 같은 네트워크 서비스 제한을 통해 제 2, 제 3의 침입을 사전에 차단할 수 있도록 한다.

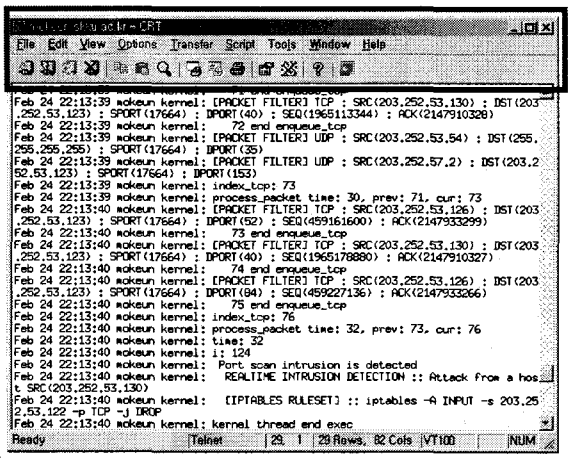
3.1.4 침입 탐지 정보 관리 모듈

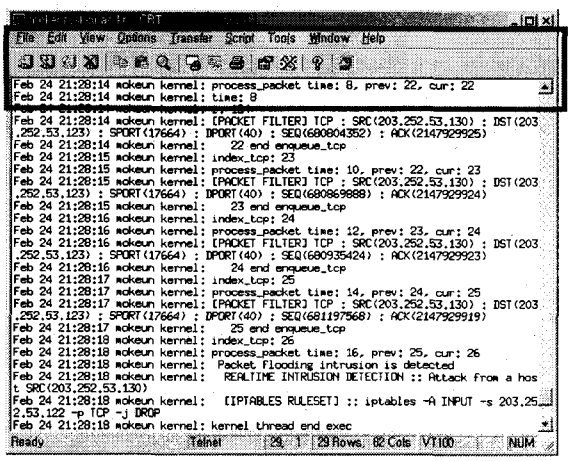
설정 모듈은 사용자의 요청 또는 시스템의 초기화가 발생 될 때, 규칙기반 접근제어 모듈과 침입방지 모듈에게 침입설정 정보를 전송하고, 로그 모듈은 침입탐지 정보를 규칙기반 접근제어 모듈과 침입방지 모듈로부터 받아서 로그 정보로 유지하도록 한다.

4. 결과 분석

본 절에서는 실제 본 논문에 제시된 기법이 적용된 커널 모듈 기반의 침입 탐지 및 방지 프로그램을 구현하여 침입 패킷에 대한 탐지와 능동적 대처가 가능한지 여부를 확인한다.

이를 위하여 인위적인 침입 샘플 데이터를 발생시킨 후 시스템에 나타나는 결과를 관찰하며 세부 모듈간 통신이 정상적으로 이루어지고 있는가를 확인한다. 침입 실험에 반응하는 침입탐지 및 대응 모듈의 동작 결과는 운영되는 커널 정보와 함께 화면에 출력된다.





5. 결론 및 향후 연구 과제

본 논문에서는 커널 모듈 프로그래밍 기법을 기반으로 netfilter 프레임웍을 이용한 실시간 침입 탐지 및 방지 기법을 제시하였다. 리눅스에서 제공하는 iptables 응용 프로프로그램을 직접 모듈상에서 함수로 호출하는 기법을 이용하여 침입에 능동적으로 대응하고자 하였으며 이를 라우팅 정책으로 반영시켜 네트워크 레벨의 침입 탐지 시스템으로 발전시키는 방향을 모색하였다.

이는 기존 응용 프로그램 레벨에서 동작하는 침입 탐지 기술과 비교하여 안정성 및 실행 속도 측면에서 성능 향상을 가져올 수 있도록 하며, 모듈간 통신 데이터 형식을 표준화 시키는 작업을 통하여 다른 IDS들과 연계할 수도 있다. 따라서 향후 리눅스 시스템의 보안 기능을 강화하고 패킷 처리 능력을 향상시키는데 보다 나은 서비스를 제공할 수 있는 기반이 될 것으로 기대한다.

참고문헌

[1] Stephen Northcutt and Judy Novak, Network Intrusion An Analyst's Handbook, 2nd Edition New Riders, 2001.

[2] Qi Zhang and Ramaprabhu Janakiraman, "A Distribute Approach to Network Intrusion Detection and Prevention," Washington University Technical Report # WUCS-01-30, 2001.

[3] Aurobind Sundaram, "An Introduction to Intrusion Detection," ACM Crossroads, 1996.

[4] David Marchette, "A Statistical Method for Profiling Network Traffic," Proceedings of The Workshop on Intrusion Detection and Network Monitoring, 1999.

[5] Wenke Lee, Salvatore J. Stolfo and Kui W. Mok, "A Data Mining Framework for Building Intrusion Detection Models," Proceedings of the 7th USENIX Security Symposium, 1998.

[6] Fengmin Gong, "Next Generation Intrusion Detection Systems," IntruVert Networks INC., 2002.

[7] Alessardo Rubini and Jonathan Corbet, Linux Device Driver, 2nd Edition O'REILLY, 2002.