IDS의 성능 향상을 위한 패킷 폐기 방안

문 종 욱[†]·김 종 수[†]·정 기 현[††]·임 강 빈[†††]·주 민 규[††††]·최 경 희[†††††]

요 약

침입탐지시스템에 대해 많은 연구가 이루어지고 있지만 이들 연구는 침입탐지시스템내의 탐지 소프트웨어의 알고리즘에만 국한되어 있다. 하지만, 침입탐지시스템의 탐지 알고리즘이 우수하더라도 침입에 해당하는 단서인 패킷을 손실하게 되면 해당 침입을 탐지해내지 못 하게 된다. 본 논문에서는 침입 탐지 시스템의 하드웨어적인 한계와 탐지 소프트웨어의 거대화에 따른 시스템 부하로 인해서 자연히 발생하게 되는 패킷 손실을 줄이기 위해서 탐지 시스템에 불필요한 패킷으로 분류될 수 있는 패킷을 미리 폐기함으로써 얻을 수 있는 탐지 시스템의 성능 향상을 다룬다. 실험 결과에 따르면 제안한 방법에 의해서 패킷 손실이 줄어들어 실제 공격에 대한 탐지율이 개선되었다.

Policy of packet dropping for enhancing IDS performance

Jongwook Moon[†] · Jongsu Kim[†] · Gihyun Jung[††] · Kangbin Yim[†††]
Minkyu Joo[††††] · Kyunghee Choi[†††††]

ABSTRACT

Although many researches on IDS (Intrusion Detection System) have been performed, the most of them are limited to the algorithm of detection software. However, even an IDS with superior algorithm can not detect intrusion, if it loses packets which may have a clue of intrusions. In this paper, we suggest an efficient way to improve the performance of IDS by reducing packet losses occurred due to hardware limitation and abundant processing overhead introduced by massive detection software itself. The reduction in packet losses is achieved by dropping hacking-free packets. The result shows that this decrease of packet losses leads an IDS to improve the detection rate of real attack.

키워드 : 침입탐지시스템(IDS), 패킷 폐기(Packet Dropping), 패킷 분류(Packet Classification), 패킷 손실(Packet Loss)

1. 서 론

한국정보보호진흥원은 신고된 국내 해킹 피해건수가 1998년 158건, 1999년 572건, 2000년 1,943건 2001년 5333건으로 해마다 3배 가까이 늘고 있다고 밝혔다[1]. 이처럼 인터넷이라는 정보 매체는 우리에게 큰 편의를 제공하지만 해킹과 같은 기업이나 개인에게 큰 피해를 주는 사례가 사회적인 문제로 떠오르고 있다. 이러한 인터넷 보안에 대한 관심의 증가로 침입차단시스템과 침입탐지시스템이 크게 각광을 받게 되었다. 침입차단시스템은 실제 침입에 대응해 차단 등의 행동을 취하여 보호를 하는 현실에서의 경비원과 역할로 비유를 한다면 침입탐지시스템은 침입에 대한 감사 기록을 남겨 놓을 수 있는 현실에서의 보안카메라의 역할을 한다고 볼 수 있다. 본 논문에서는 이처럼 인터넷 보안 분야에서 중요하게 대두되고 있는 침입탐지시스템의 패킷 처리능력 향상 방안에 대해 연구하고 실험한 내용을 제시한다.

1980년대 Anderson에 의해 침입탐지시스템의 이론이 나온 이후로 침입탐지시스템의 성능 향상을 위한 연구가 활발히 진행 되었다. Heady와 Luger의 분류 시스템 기술을 이용한 기법에 기반으로 한 침입 탐지 기술 등을 예로 들 수 있다[2, 3]. 이러한 일련의 침입 탐지 시스템의 성능 향상을 위한 연구들은 모두 침입 탐지 알고리즘 자체에 편중되어 있다. 본 논문에서는 이러한 기존 침입탐지 시스템의 성능 향상을 위한 연구와는 다른 관점에서 접근을 시도한다. 침입탐지시스템의 대분이 범용 하드웨어와 범용 운영체제를 기반으로 하기 때문에 본질적으로 패킷 수집의 성능이 떨어질 뿐 아니라, 침입을 탐지하기 위한 부담도 커서 과도한 패킷들이 유입될 때 제대로 처리를 못 하게 된다. 이러한 침입 탐지 시스템의 하드웨어적인 한계와 탐지 소프트웨어의 거대화에 따른 시스템 부하로 인해서 자연히 발생하게 되는 비정상적인 패킷 손실을 줄이기 위해서 탐

※ 이 논문은 과기부 국가지정연구실사업의 지원으로 연구되었음.
† 준 회 원 : 아주대학교 대학원 전자공학과
†† 정 회 원 : 아주대학교 전자공학부 교수
††† 정 회 원 : 아주대학교 정보통신전문대학원 전임연구원
†††† 준 회 원 : (주)뉴어텍 연구원
††††† 정 회 원 : 아주대학교 정보 및 컴퓨터 공학부 교수
논문접수 : 2001년 12월 20일, 심사완료 : 2002년 7월 2일

지 시스템에 불필요한 패킷으로 분류될 수 있는 패킷을 미리 폐기하는 기법을 제안하고 이를 통해서 얻을 수 있는 탐지 시스템의 성능 개선 결과에 대해서 논한다.

이처럼 본 논문에서의 핵심적인 연구 항목은 침입탐지시스템에서의 패킷 폐기이다. 패킷 폐기에 관련된 연구로써 혼잡 제어에 사용되는 RED(Random Early Detection)가 활발히 진행되고 있다[4]. 이러한 RED의 적용은 이것의 속성상 라우터와 같이 출력부가 있는 모델에서나 가능하므로 네트워크 구성에서 종단시스템일 뿐 아니라 네트워크에 흐르는 패킷을 빠짐없이 수집하여 이에 대한 분석을 하는 입력부만 존재하는 모델에서는 불가능하다고 볼 수 있다. 하지만 RED를 다루는 논문들의 다양한 성능평가 지표들 중에서 패킷 손실량은 네트워크에 지나다니는 모든 패킷을 빠짐없이 탐지를 해야 하는 침입탐지시스템에서도 적용이 가능하다[5, 6].

McRae의 논문에 따르면 고속 네트워크에서 침입탐지시스템의 성능 문제를 해결하기 위해 침입탐지시스템에 필요한 패킷 만을 선별하여 보내는 packet tap이라는 장치를 사용한다고 한다[7]. 하지만 이 논문에서 언급하는 것처럼 이 방식의 문제점은 '침입탐지시스템'이 관심이 있는 패킷의 수가 상당히 많다는 것이다. 그리고 이러한 방식은 packet tap 자체는 적은 오버헤드를 가져야 하고 그것은 침입탐지시스템이 관심이 있는 패킷만을 잡아낼 수 있어야 한다. 본 논문은 이와 반대로 침입탐지시스템이 관심이 없는 패킷에 대한 분류를 하여 그 패킷들을 폐기하여 탐지시스템에 전달하지 않도록 한다. 본 논문에서 제안하는 방법은 Packet Tap을 사용하는 방법에 비해 비교의 회수가 작기 때문에 고속의 처리가 가능하다. 또한 새로운 해킹 기법이 등장하게 되면 Packet Tap을 사용하는 경우는 항상 룰의 변경이 이루어져야 하는 반면 제안하는 방법은 상대적으로 변경의 여지가 작다.

본 연구에서의 실험은 커널에서 패킷 필터링이 가능한 BSD 계열의 FreeBSD 운영체제에서 일반에게 공개된 가벼운 규칙 기반의 침입탐지시스템인 snort를 패킷 손실측정이 가능하도록 수정하여서 본 논문에서 제안한 패킷 폐기를 적용한 snort가 그렇지 않은 snort보다 성능이 향상됨을 보인다. 이러한 성능이 향상됨을 보이기 위해서 전술한 성능 측정 지표로써의 패킷 손실량을 본 실험에 도입을 하였다. 본 실험에서는 의도적인 패킷 폐기로 인한 성능 향상을 확인 할 뿐 아니라 침입탐지시스템의 성능이 규칙의 적용에 snort에서 룰을 적용할 때와 그렇지 않을 때의 성능 차이를 비교하고 snort에서 감사자료를 남길 때와 아닐 때의 성능 차이를 비교한다.

본 논문의 구성은 서론에 이어서, 제2장에서는 연구의 동기가 된 침입 탐지 시스템의 성능 제한에 대한 고찰을 다루며, 제3장에서는 침입 탐지 시스템에 있어서 불법적 행동의 여지가 없는 폐기 가능한 패킷들의 분류를 한다. 제4장에서는 제안된 패킷 폐기를 통한 성능 이득을 실험하기 위한 환경과 실험 방법에 대해서 논하고, 제5장에서는 실험의 결과에 대한 분석을 한다. 마지막으로 제6장에서는 본 논문에서 도출된 결론과 향후 해결과제에 대해서 논한다.

2. 제안 동기

본 장에서는 침입 탐지 시스템의 성능 문제를 침입 탐지 시스템의 구조적인 관점과 자체적인 관점에서 기술하고 제안된 패킷 폐기를 통해서 성능 향상을 얻을 수 있음을 살펴본다. 침입 탐지 시스템의 성능은 실제 탐지 소프트웨어의 알고리즘에 따라 많이 좌우되지만 또한 하드웨어로부터 상위 소프트웨어로 이어지는 구조의 효율적 디자인에 의해서도 많은 차이가 난다. 이러한 구조적인 문제로부터 발생하는 침입 탐지 시스템의 성능 저하에 대한 두 가지 관점을 논한다. 하나는 범용 하드웨어 시스템의 한계이고 또 다른 하나는 범용 운영체제를 기반으로 한 소프트웨어의 불필요한 처리에 대해서이다.

AT&T Labs에서 발표한 연구 조사서를 통해 알 수 있듯 네트워크 처리 성능 향상을 위해서는 전체적인 시스템 구조의 변경이 불가피하다[8]. 하지만 대부분의 침입 탐지 시스템은 범용 하드웨어 시스템을 기반으로 하기 때문에 이미 정형화되어 만들어져 있는 네트워크 인터페이스 카드를 사용하고 높은 지연이 유발되는 I/O 버스를 사용하게 될 수 밖에 없다. 이것은 침입 탐지 시스템에서 중요한 기능 중 하나인 패킷 수집의 성능이 떨어질 수 있음을 내포한다.

대부분의 침입 탐지 시스템에서 운영 체제 또한 NT 또는 LINUX와 같은 범용 운영 체제를 사용한다. 커널이 침입 탐지를 위해 특화 되지 않았다는 것은 침입 탐지와 무관한 코드들이 많다는 뜻이다. 이는 많은 네트워크 부하가 걸릴 때 불필요한 코드의 처리 시간으로 인해 손해를 감소해야 됨을 의미한다. 그리고 대부분의 침입 탐지 시스템에서의 패킷 수집은 Berkeley 대학교에서 커널 내에 구현한 BPF(BSD Packet Filter)디바이스의 인터페이스를 위한 libpcap 라이브러리를 그대로 또는 약간의 변경을 통해서 이루어지고 있다. 비록 BPF 알고리즘이 다른 CSPF나 NIT의 것 보다 성능이 우수하다고는 하나 RISC를 기반으로 창안된 것이어서 CISC에서의 성능은 다시 고려해 봐야 할 것이고 사용자 측면에서는 중간의 libpcap을 또 다시 거쳐야 하므로 오버헤드가 있다[9].

침입 탐지 시스템은 네트워크에 흐르는 모든 패킷을 수집하고 이를 분석하여 불법적인 행동이나 유해 정보를 찾아내서 감사자료를 남겨 놓아야 하므로 침입 탐지 시스템이 처리 해야 할 부하가 크다. 침입 탐지 시스템의 부하에 관련해서 충분히 예상이 되는 수행해야 할 일들을 간단히 살펴보고 이러한 부하로 인해 침입 탐지에 있어서 야기되는 문제점들을 살펴보면 다음과 같다.

첫째, 침입 탐지 시스템은 침입을 탐지 하기 위해 내부에 가지고 있는 침입 패턴에 대한 데이터베이스를 검색하는 일이 필요하다. 데이터베이스 검색에 대한 소프트웨어적 알고리즘이 아무리 우수하더라도 데이터베이스의 양이 엄청나게 많을 때는 수행 시간이 길어지는 것이 당연하다. 인터넷 보급 속도의 증가와 함께 인터넷에서의 침입을 위한 방법이 빠른 속도로 새롭게 생겨나고있고 더욱이 그들 많은 방법은 침입 탐지 시스템을 속이기 위해 위장을 한다. 이로 인해 단순한 규칙에 의존하는 데이터베이스에서 복잡하고 연산량이 많아지는 쪽으로 변경되어야만 한다.

둘째, 동시 다발적으로 일어나는 모든 불법 행동을 관리하기 위해서 침입 탐지 시스템은 이상적으로는 무한대의 세션을 관리를 해야 한다. 실질적으로는 메모리의 제한과 하드웨어 또는 운영체제의 한계로 인해서 세션의 양이 제한된다. 이러한 세션 관리는 침입 탐지 시스템의 큰 부하의 요소로 작용한다. 예를 들어서 침입 탐지 시스템으로 유입되는 패킷들에 의해 우연히 또는 악용적인 이유로 많은 양의 새로운 연결이 이루어 진다면 아무리 세련된 세션 관리 기법을 도용을 하더라도 시스템은 엄청난 부하에 직면한다.

셋째, 침입 탐지 시스템은 이른바 해킹이라 불리는 불법 침입에 대한 처리뿐 아니라 패킷 자체 내용의 유해 정보의 패턴 검색이 필요하다.

이처럼 기존의 침입탐지 시스템은 범용 하드웨어 시스템의 한계로 인해 망의 네트워크 부하가 클 때, 특히 작은 크기의 패킷이 burst하게 들어 올 때 어쩔 수 없는 패킷 손실과 커널 내에서 다른 처리의 우선 순위에 밀려서 일어나는 libpcap의 자체적인 손실은 침입 탐지 시스템에 있어서는 가장 치명적인 침입의 패턴의 일부를 잃어버리는 결과를 낳는다. 하지만 본 논문에서 제안하는 방법인 침입탐지시스템에게 필요성이 떨어지는 패킷을 미연에 폐기하는 방식을 적용하면 버려진 양에 비례해서 시스템 부하가 줄어들므로 보다 중요한 패킷의 수집 확률이 높아지게 되는 것이다.

그리고 침입탐지 소프트웨어의 자체적인 무거움으로 인해서 발생하는 패킷 손실은 탐지 소프트웨어가 인지는 하지만 처리 성능의 한계로 버려야 하는 경우이다. 이 경우에도 본 논문에서 제안하는 패킷 폐기 정책을 적용하면 현저한 성능 향상을 기대할 수 있다. 왜냐하면 침입과 무관한 패킷이 폐기되지 않고 탐지 소프트웨어까지 올라왔을 때 침입의 유형과 유해 정보에 관련된 데이터베이스를 뒤지는 등의 일련의 처리 과정을 모두 거친 후에야 패킷을 폐기하므로 미리 버려주게 되면 그러한 불필요한 처리를 보다 중요한 패킷에게 할애를 해 줄 수 있기 때문이다.

본 논문에서는 앞으로 침입 탐지 시스템의 성능 또는 기능에 문제가 없이 폐기가 가능한 항목에 대해 고찰을 한 다음 그 중 성능 측면에서 향상의 가능성이 가장 기대되는 항목인 HTML 패킷 폐기로 인한 성능 향상이 이루어짐을 실험을 통해서 측정을 하고 그 결과를 분석한다.

3. 패킷 폐기 방안

웹의 폭발적인 성장에는 멀티미디어 서비스가 큰 역할을 하였다고 볼 수 있다. 즉 네트워크의 사용 중 많은 부분을 웹이 차지하고 그 중에서도 멀티미디어 서비스가 차지하는 비중은 상당하다고 볼 수 있다. 내부 사용자가 외부 멀티미디어 서버로부터 동영상을 보거나 음악을 듣는 경우 연결 설정을 위한 일련의 패킷을 제외한 실제 영상 또는 음악을 담고 있는 패킷은 침입과 무관하므로 IDS가 해킹 여부를 검토하지 않아도 상관없다. 하지만 멀티미디어 서비스를 위한 프로토콜 타입이 패킷 헤더 내에 정의되어 있지 않으므로 단일 패킷의 헤더만으로는 실제 데이터 패킷과 연결 설정을 위한 패킷을 구분할 방법이 없다. 하지만 대부분의 사용자가 웹을 통해서 멀티미디어 서비스를 받고 있는데 이 경우는 HTTP의 MIME 정보를 감시한 후 real-player와 같은 잘 알려진 미디어 플레이어로 연결 될 때 해당 요청자의 포트 번호를 추적하면 구분이 가능하다. 그런데 이러한 기법은 멀티 미디어 서버로 접근하는 사용자별의 세션을 따로 관리하는 등 처리 비용이 많이 든다. 최근에 그 보급 속도가 급속히 확산되고 있는 VOIP(Voice Over IP)나 FOIP (Fax Over IP)등의 서비스에서 세션 정보를 제외한 대부분의 데이터 정보는 해킹의 가능성이 거의 없으므로 폐기 가능한 정보에 속한다.

위에서 살펴본 멀티미디어 패킷과 VOIP 패킷은 표준화된 헤더에 그 정보가 없기 때문에 단독 패킷 만으로는 판단을 할 수가 없다. 이는 별도의 추적 알고리즘을 구현하여야 하므로 구현에 어려움이 있다. 하지만 아래 자세하게 설명할 내부 웹 서버의 검증 받은 내용을 담고 있는 패킷(VHP : Verified HTTP Packet)을 폐기하는 방법은 패킷의 헤더의 패턴 비교 만으로 구분이 가능하다. (그림 1)은 이러한 헤더의 패턴 비교를 통해 VHP를 폐기하는 알고리즘을 의사코드로 표현한 것이다. 그래서 쉽게 구현을 할 수가 있고 또한 인터넷의 전체 트래픽 중에서 웹 트래픽이 차지하는 비율이 우월하므로[11, 12] 더욱 많은 패킷을 버려줄 수 있어서 IDS의 효율이 높아진다. 웹의 실제 내용은 IDS에게 있

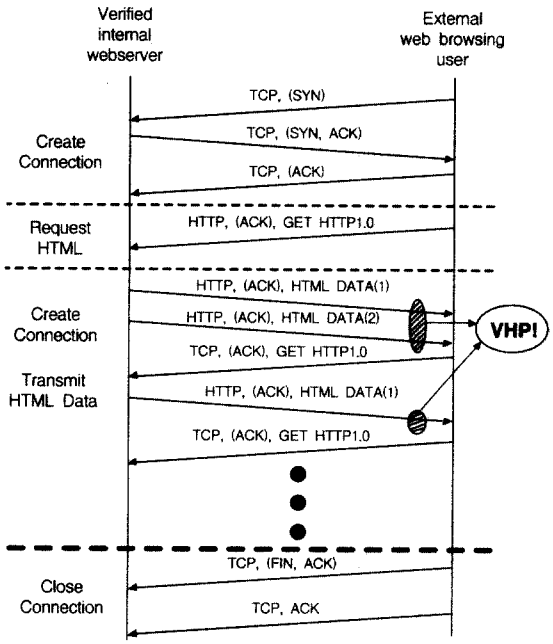
```
When packet is coming to PDE( packet dropper engine )
  drop_flag = FALSE
  if type field in MAC header is IP
    if protocol field in IP header is UDP or TCP
          if source port # in TCP or UDP header is HTTP
           if ((PDEs IP XOR dest. IP) AND NETMASK )
             if does any flag except ACK exist in TCP flags
               // This packet is real WEB Content
               drop_flag = TRUE

  if drop_flag is FALSE
       Deliver this packet to Intrusion Detection System
  else
    Nothing to do( dropping packet )
```

(그림 1) 제안한 알고리즘의 의사코드

어서 유해 정보 탐색의 대상이 되지만, 내부 웹 서버내의 내용이 유해한 정보가 없다고 가정을 하면 VHP는 IDS에게는 쓸모없는 정보가 된다.

(그림 2)는 전형적인 HTTP 연결 설정 과정을 보여준다. 본 논문에서 설정한 해킹과 무관한 패킷인 VHP는 그림에 나와 있듯이 전체 연결과정 중에 내부 웹 서버로부터 나가는 실제 웹 내용 만을 담고 있는 패킷을 말한다. 하지만 VHP를 폐기 할 때 주의할 점은 침입탐지시스템의 기능상의 문제가 없는 범위 내에서 이루어져야 한다는 것이다. 즉 침입의 단서가 될 수 있는 패킷은 버리지 않아야 한다.



(그림 2) 제안한 알고리즘에 의해서 폐기 될 VHP

(그림 2)에서 여러 패킷들 중에서 TCP 연결 설정 패킷은 꼭 침입탐지시스템에 전달되어야 한다. 왜냐하면 TCP 패킷들에 대해서 침입탐지시스템은 source IP와 destination IP에 따른 세션을 관리하고 이 세션을 기반으로 해서 침입을 추적하고 있기 때문이다. 그러므로 이러한 패킷을 버리게 되면 IDS는 해당 세션을 생성하지 못하므로 침입을 탐지 못하게 된다. 그뿐 아니라 이러한 TCP 연결 설정 때의 패킷들은 SYN Flooding이나 ACK Storm과 같은 DDOS에서 사용되는 해킹의 주된 요소로써 웹 서버를 공격하는 용도로 많이 사용되므로 IDS에게 꼭 넘겨 줘야 한다. 본 논문에서 제안한 해킹과 무관한 패킷인 VHP는 이러한 TCP 연결 설정 패킷을 담고 있지 않음을 (그림 2)에서 확인할 수 있다.

웹 서버 보안 취약점은 웹 서버 구현상의 취약점, CGI 관련 취약점, 그리고 웹 서버 구성상의 취약점으로 구분할 수 있다. 웹 서버 보안 취약점의 예로서 MS IIS 3.0이전 버전의 경우 스크립트의 내용을 볼 수 있는 원격 읽기 취약점을 들 수 있고 Apache 웹 서버의 경우 쿠키를 이용해서 프로그램 스택을 덮어씀으로써 임의의 명령을 수행할 수 있는 취약점을 들 수 있다. 그리고 CGI 관련 취약점의 예로는 외부의 사용자에게 호스트의 정보를 보여주는 취약점과 사용자의 입력을 받는 CGI의 경우 원격지의 사용자가 임의의 명령을 실행할 수 있는 취약점을 들 수 있다. 마지막으로 웹 서버 구현상의 취약점으로는 파일의 접근 권한과 심볼릭 링크로 인해서 야기되는 것들을 들 수 있다. 이처럼 HTTP 프로토콜을 이용해서도 많은 공격이 이루어진다. 그런데, 본 논문에서 제안한 VHP는 바로 HTTP 프로토콜을 기반으로 한 패킷이므로 해킹의 단서를 담은 패킷일 수 있다. 하지만 공개되어있는 IDS인 snort[13]에서 이들을 탐지하기 위한 439개의 웹 관련 규칙을 살펴본 결과 3가지를 제외하고는 모두 외부 망에서 내부 웹 서버로 들어오는 패킷으로 판별하고 있다. 내부 웹 서버로부터 외부 망으로 나가는 패킷을 가지고 판별하는 경우는 Invalid URL, 403 Forbidden과 같은 실제 해킹이라고는 보기 힘든 의심적인 행동에 대한 보고라고 볼 수 있고 전체 웹 관련 규칙에서 0.68%에 지나지 않으므로 무시할 수 있다.

4. 실험 방법

본 실험에서는 침입의 가능성이 없는 검증 받은 내부의 웹 서버로부터 나가는 웹의 실제 내용을 담은 패킷을 미연에 폐기를 함으로써 보다 침입탐지시스템에 유용한 패킷의 수신 율을 높임으로써 네트워크 부하가 큰 곳에 위치된 침입탐지시스템의 탐지 능력이 향상되는 것을 확인하고자 하는 것이 목표이다.

침입탐지시스템에게 있어서 패킷은 침입을 탐지해내기 위한 단서라고 볼 수 있다. 탐지 소프트웨어가 많은 침입 패턴을 검색할 수 있다고 하더라도 침입에 해당하는 단서를 손실하게 되면 해당 침입을 탐지해내지 못 한다. 이러한 관점에서 본 논문에서는 침입탐지시스템에서의 성능 향상과 패킷 손실 량이 밀접한 관계가 있다고 가정한다. 그래서 먼저 패킷 손실량 측정하여서 기존의 snort와 제안하는 알고리즘을 적용한 수정된 snort의 HTTP 패킷의 부하 정도에 따른 탐지시스템의 패킷 처리 능력을 측정한다. 이러한 패킷 처리 능력 향상이 실제 탐지 능력의 향상으로 이어지는지를 확인하기 위해 본 실험에서는 실제 해킹 패킷을 보냄으로써 해당 해킹에 대한 탐지율을 측정한다.

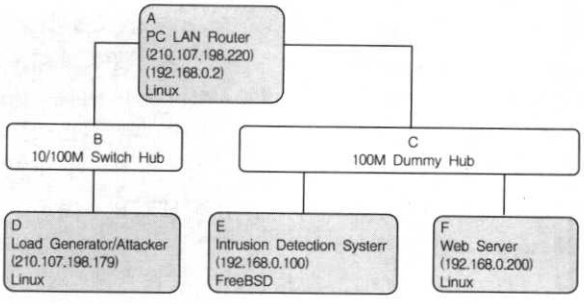
침입탐지시스템의 패킷 처리 능력을 측정하기 위한 실험에서 사용하는 패킷은 탐지시스템에 쓸모 있는 패킷과 그렇지 않은 패킷을 구분한다. 즉 침입의 가능성 유무를 구분한다. 이 실험에서는 두 가지 프로토콜의 패킷 만을 사용하는데 하나는 웹 페이지 요청으로 인한 HTTP 패킷(TCP 패킷)이고 나머지 하나는 UDP 패킷이다. 사실 TCP 패킷 전체가 본 논문에서 제안한 해킹과 무관한 패킷인 VHP인 것은 아니지만 실험의 편의를 위해서 TCP 전체를 해킹과 상관없는 패킷으로 규정을 한다. 반대로 UDP 패킷은 침입탐

지시스템에게 있어서 중요한 즉, 침입의 가능성이 많은 패킷으로 간주한다. 침입의 가능성이 있는 패킷으로서 UDP 패킷을 선정한 이유는 TCP 프로토콜 기반의 패킷들은 네트워크가 혼잡할 때 라우터와 같은 네트워크 장비에서 UDP 보다 많은 패킷 손실이 발생하기 때문에 실험 시에 결과에 오차가 많이 생기기 때문이다.

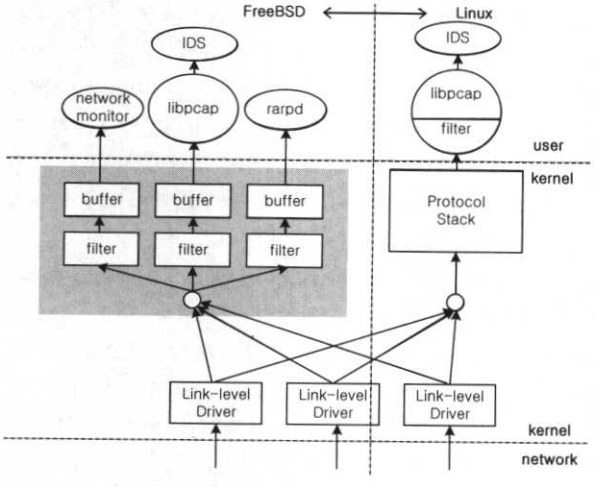
실험에 사용한 측정 도구에 대해서 간단히 살펴보면 다음과 같다. 실험에 사용된 snort는 실시간 트래픽 분석과 IP 네트워크 상에서 감사 및 기록이 가능한 가벼운 네트워크 침입 탐지 시스템이다.[13] Snort는 패킷 수집 라이브러리인 libpcap에 기반하여 쉽게 정의할 수 있는 침입 탐지 규칙 들에 일치되는 네트워크 트래픽을 감시하고, 기록할 수 있는 도구이다. 그리고 http_load는 원래 웹 서버의 성능을 측정하기 위해서 만든 도구로써 사용자가 원하는 웹 페이지를 필요한 수만큼 요청할 수 있다.[14] 본 논문에서는 이 도구를 이용해서 침입탐지시스템에 HTTP 트래픽의 부하를 조절한다. Mgen은 UDP 트래픽을 만들어서 IP 네트워크의 성능을 테스트하기 위해 만든 도구이며, 패킷의 크기와 전송률 등을 사용자가 임의로 변경할 수 있다.[15] 실제 공격을 위해서 사용한 툴은 targa라는 여러 해킹 기법을 패키지화 한 프로그램을 사용하였다. Teardrop과 같은 IP 패킷조각 재조합의 함수의 구현 오류를 이용한 공격은 침입탐지시스템에서 매 패킷 마다 탐지를 할 수 있는 반면 SYN Flooding과 같은 서비스 거부형 공격의 경우 대부분의 침입탐지시스템은 일정 시간 동안에 들어온 패킷의 량을 가지고 판별을 한다. 그래서 본 논문에서는 공격 패킷 개수와 탐지된 회수가 동일해서 탐지율을 측정하기 편리한 Teardrop 공격을 이용해서 실험을 하였다.

이들 도구들을 이용한 실험을 위한 네트워크 환경은 (그림 3)에서 보는 것과 같다. 네트워크 세그먼트 구분을 위해서 PC 기반의 라우터를 두었다. 본 실험이 제대로 이루어지기 위해서는 외부에서(A 라우터 좌측의 B 허브를 통해 공유하는 망) 내부로(A 라우터의 우측의 C 허브를 통해 공유하는 망) 또는 그 반대로의 트래픽의 대역폭에 큰 제한이 없어야 한다. 그래서 A 라우터는 고성능의 1Ghz Dual Zeon CPU를 장착한 서버급 PC를 사용하였다. 또한 같은 이유로 A 라우터의 인터페이스는 보통의 비대칭형 라우터와는 달리 인터페이스의 대역을 대칭적으로 구성하였다.

C 허브를 100M 더미 허브로 사용한 이유는 E 시스템에 침입탐지 시스템이 위치하므로 같은 세그먼트 F로 가는 패킷도 모두 감시해야 하기 때문이다. 그림에서 D는 배경 잡음 패킷과 HTTP 패킷을 생성하고 또한 실제 공격 패킷도 생성한다. 본 실험에서 D에서 UDP와 HTTP가 동시에 생성되어 E 침입탐지시스템과 F 웹 서버로 전달된다. 그리고 F 웹 서버의 웹 내용은 검증이 되었다고 가정을 한다. 즉 이 서버로부터 실제 외부 사용자에게로 나가는 패킷 중 연결 설정 정보를 지니지않은 패킷은 불법 침입 정보가 아니므로 E 침입탐지시스템에서 해당 패킷을 폐기가 가능하다.



(그림 3) 네트워크 구성도



(그림 4) FreeBSD와 LINUX에서의 패킷 필터의 위치

E 침입탐지시스템의 운영체제는 FreeBSD를 사용하였다. 침입탐지시스템이 사용하는 패킷 수집 라이브러리인 libpcap이 BSD계열s 운영체제와 여타 다른 운영체제와 내부 패킷 처리 구조가 많은 차이가 난다. 가장 큰 차이점은 FreeBSD의 경우 (그림 4)에서 보는 바와 같이 사용자 영역에서 변경 가능한 패킷 필터가 커널 내부에 존재하고, LINUX의 경우는 libpcap 라이브러리에서 처리를 한다는 것이다.

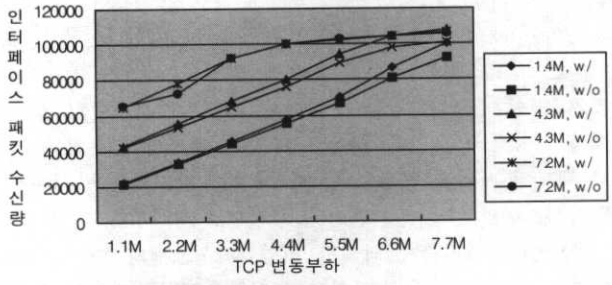
5. 실험결과 및 분석

본 논문에서는 크게 두 가지 실험을 하고 있다. 먼저 제안하는 알고리즘을 적용한 경우에 의심의 여지가 많은 패킷의 수신 율이 향상됨을 보여서 침입탐지시스템의 성능 향상 가능성을 보인다. 다음으로 높은 네트워크 부하를 걸고 내부의 서버에 해킹을 할 때 제안한 알고리즘을 적용한 탐지시스템의 침입 탐지율이 실제적으로 향상됨을 보인다.

제안한 알고리즘에 의해서 침입탐지시스템의 패킷 처리 능력이 향상되는 것을 확인하는 실험은 모두 10초간 수행이 되었으며 총 10회의 실험을 한 후 순간적인 통신망 부하의 이상변화에 의한 영향을 줄이기 위해 최대치와 최저치를 제외한 나머지의 8개의 실험치의 평균값으로 결과를 표

시하였다. 실험에서 네트워크 부하의 조절을 위해서 UDP를 고정부하로 사용하고 TCP를 변동 부하로 사용을 하였다. 실험에서 사용한 UDP 패킷은 네트워크 부하 조절의 용도일 뿐 아니라 해킹의 여지가 있는 패킷으로 간주를 한다. 그러므로 실험에서 보이는 UDP 패킷 수신 량의 차이는 침입의 증거 수집량의 차이라고 보아도 무방하다. 그리고 실험 결과에서 표현되는 모든 TCP 패킷은 외부의 유저가 내부의 웹 서버의 웹 페이지를 가져가기 위한 HTTP 프로토콜에 의해 파생된 패킷이다. 즉 이 TCP 패킷은 제안하는 알고리즘에 의해 일부가 폐기가 된다. 그러므로 TCP 패킷이 많아질수록 제안한 알고리즘을 적용한 탐지시스템은 불필요하게 처리할 패킷량이 상대적으로 줄어 든다. 결국 본 논문에서 UDP를 고정하고서 TCP를 증가시켜가며 실험하는 이유는 침입의 증거 량을 일단 일정하게 한 다음 잡음 패킷이 증가할 때 잡음 패킷 중 확실히 침입의 여지가 없는 패킷으로 판단되는 패킷을 미리 폐기하는 경우가 보다 침입의 증거를 많이 수집할 수 있음을 보이는 것이다.

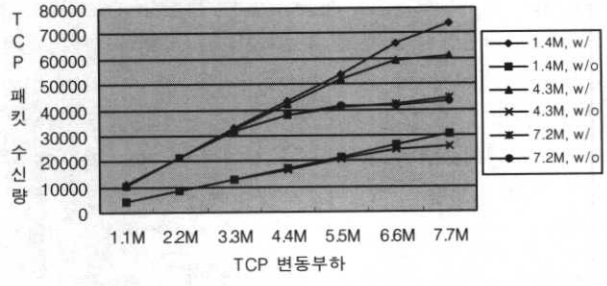
(그림 5)은 탐지시스템의 네트워크 인터페이스에서 패킷의 전체 수신 량을 보여준다. 이는 제안하는 패킷 폐기를 위해서 사용된 필터 엔진의 사용 여부가 인터페이스에서의 수신 량에 미치는 영향을 고찰하기 위함이다. 먼저 고정 부하가 가장 낮은 1.4M인 경우를 보면 제안한 알고리즘을 적용한 snort에서 측정한 "w/o HTTP" 그래프의 인터페이스에서 수신 량이 기존의 snort에서 측정한 "w/ HTTP" 그래프 보다 떨어짐을 알 수 있다. 이것은 매 패킷마다 패킷 필터링을 위한 비교 연산이 수행되기 때문에 그 비교 연산을 수행한 시간에 비례해서 인터페이스에서의 패킷 수신율이 떨어지는 것이다. 고정 부하가 4.3M인 경우도 마찬가지로 필터링을 수행하는 경우가 필터링 연산의 오버헤드로 인해서 인터페이스에서 수신량이 떨어진다. 고정 부하가 1.4M일 때는 그래프가 선형적인 반면 4.3M인 경우는 그래프가 특정 수치로 수렴하는 곡선의 모양으로 나타나는데 이유는 고정 네트워크 부하가 이미 포화상태에 이를 만큼 크기 때문이다. 같은 이유로 고정 부하가 7.3M인 경우에 수렴 치에 더 빨리 도달한다.



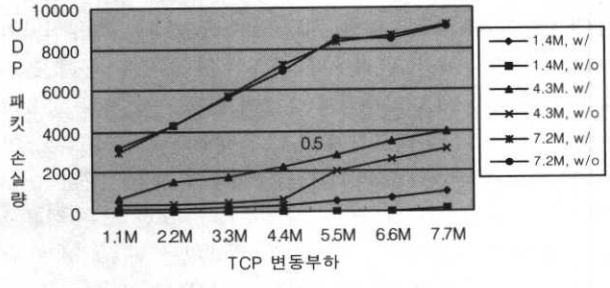
(그림 5) 인터페이스에서의 패킷 수신 량

이러한 결과는 본 논문에서 제안하는 알고리즘의 적용으로 인한 단점으로 볼 수 있다. VHP를 걸러내기 위한 패킷 필터링 작업이 커널 내에서 시스템의 자원을 소모해서 해야 하므로 당연한 결과라고 볼 수 있다. 하지만 앞으로의 결과는 인터페이스에서 상대적인 손실의 차이를 충분히 만회함을 보여준다. 그리고 본 논문에서 제안하는 알고리즘을 별도의 임베디드 시스템에 구축을 하면 이러한 인터페이스에서의 손실은 없어질 것이다.

(그림 6)은 TCP 패킷을 점차적으로 증가시켜 가면서 의도된 HTTP 패킷 폐기가 이루어짐을 확인하기 위해 TCP 패킷 수신량을 측정하였다. 고정부하가 4.3M인 경우에 변동 부하가 5.5M일 때를 보면 제안한 알고리즘을 적용한 snort에서 측정한 경우가 기존의 snort에서 측정한 경우보다 3만개 정도 패킷을 적게 수신한 것을 알 수 있다. 즉 본 논문에서 제안한 알고리즘에 의해서 폐기된 량이 3만개인 것이다. (그림 7)은 의도된 패킷의 폐기로 인해서 보다 침입의 여지가 있는 패킷으로 가정한 UDP 패킷의 손실량이 줄어드는 것을 확인하기 위해 측정한 것이다. (그림 7)을 볼 때 한 가지 주의할 사실은 다른 그래프는 수신 량을 측정하고 있는데 반해 이 그래프는 손실량을 측정하고 있다는 사실이다.



(그림 6) TCP 패킷 분석량



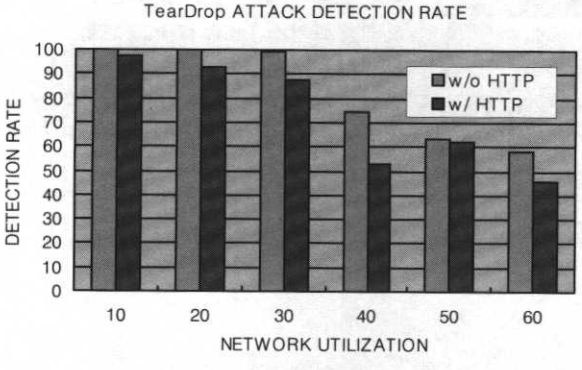
(그림 7) UDP 패킷 손실량

(그림 6)에서는 제안한 알고리즘에 의한 패킷 폐기를 적용한 snort(w/o HTTP)에서 분석한 TCP의 개수가 훨씬 줄어들었음을 알 수 있다. 이는 분석할 TCP의 개수가 줄어든 만큼의 시스템의 여유로 보다 해킹의 여지가 많은 패킷에 시스템의 자원의 할애가 가능하다는 것을 의미한다. 보다 침입의 가능성이 많은 패킷 분석에 침입탐지시스템의 자원이 할애 되었다는 사실은 (그림 7)를 통해서 알 수 있다. 본 논문에서는 해킹의 여지가 있는 패킷으로 UDP 패

킷으로 설정을 하였는데 (그림 7)에서 고정부하가 1.4M 일 때 필터링을 적용한 쪽이 UDP의 손실이 거의 일어나지 않는 것에 비해서 필터링을 하지 않은 쪽은 UDP의 손실이 상당히 많음을 알 수 있다.

(그림 5), (그림 6), (그림 7)에서 고정 네트워크 부하가 가장 큰 7.3M인 경우는 이미 시스템의 포화상태에 도달한 상태여서 필터링을 적용한 경우와 아닌 경우의 차이가 나지 않음을 볼 수 있다. 네트워크의 부하가 크다는 것을 패킷의 간격의 관점에서 살펴보면 패킷의 간격이 조밀하다는 뜻과 동일하다. snort와 같이 매 패킷마다 처리를 하는 경우 자신이 패킷을 하나 처리하는 시간이 패킷 간의 간격보다 길 때는 패킷의 손실이 일어난다. 네트워크의 부하가 큰 만큼 패킷 간의 간격이 좁아지게 되고 그만큼의 패킷의 손실이 생긴다. 이러한 부하로 인한 손실의 무한 반복 속에서 본 논문이 제안하는 의도적 패킷 폐기로 인한 이득이 상실되어 버린다. 즉 이미 과부하가 걸린 시스템의 경우는 본 논문에서 제안하는 의도적 패킷 폐기가 큰 효율을 발휘할 수 없다. 이것은 탐지 소프트웨어가 동작하고 있는 시스템의 사양(CPU 속도, 캐시의 크기, 메모리 크기)에 의해 제한되는 부분이라 볼 수 있다.

(그림 8)은 배경 잡음 트래픽과 웹 트래픽을 1 : 1로 유지하면서 측정된 결과이다. 네트워크 이용률을 증가시키며 매번 1000개의 teardrop 공격을 하여서 제안한 패킷 폐기 알고리즘을 적용한 snort와 기존의 snort에서 측정한 침입 탐지 율을 측정하여 도식화하였다. 그림에서 알 수 있듯이 제안한 알고리즘을 적용한 snort에서 측정된 침입 탐지 율이 더욱 높다. 즉 앞선 실험을 통해서 보여준 탐지시스템의 패킷 처리 능력의 향상으로 침입의 가능성이 많은 패킷에 시스템의 자원이 많이 할애되어서 나온 결과로 해석이 된다. 또한 네트워크의 이용률이 증가함에 따라 제안한 알고리즘을 적용한 snort에서 측정한 탐지 율과 기존의 snort에서 측정한 탐지율의 차이가 커짐을 알 수 있다. 이것은 HTTP패킷의 량이 많아짐에 따라 제안한 알고리즘이 적용된 snort의경우에 보다 많은 패킷이 미연에 폐기될 수 있어서 그만큼의 여분의 처리 시간을 벌게 되어서 차이가 커지는 것이다.



(그림 8) TearDrop 공격 탐지율

6. 결 론

본 논문에서는 침입탐지시스템의 성능에 대한 문제점 개선을 위해서 네트워크 부하의 감소가 보장되고 불법 침입도 유해 정보도 아닌 내부 서버의 실제 웹 내용을 담은 패킷의 폐기를 제안하였다. 실험 결과를 기반으로 해서 다음과 같이 결론을 지을 수 있다. 필터링 엔진 사용 여부에 따른 침입탐지시스템의 성능에 대한 결과는 네트워크의 부하가 시스템의 성능의 한계를 넘어서지 않는 범위 내에서는 예측대로 제안한 의도적인 패킷 폐기(필터링 엔진을 사용하는 경우)를 한 시스템이 그로 인한 시스템의 성능 이득으로 보다 침입의 여지가 많은 패킷의 처리에 할애가 가능해져서 불법의 소지가 많은 패킷의 손실이 작았다. 하지만 시스템의 성능의 한계를 넘어서는 범위 내에서는 의도적인 패킷 폐기를 한 시스템과 그렇지 않은 시스템의 성능의 차이가 나지 않았다. 이러한 패킷 처리 능력의 확대가 실제 공격에 대한 침입 탐지율을 향상시키고 있음을 teardrop 공격을 시도하여서 확인하였다.

본 논문에서는 패킷 필터링의 부하의 영향을 최소화하기 위해서 필터 엔진이 커널 내에 있는 BSD 계열의 운영 체제인 FreeBSD에서 실험을 하였다. FreeBSD의 경우 필터 엔진이 커널에 내부에 있지만 결국 침입 탐지시스템 내에서 필터링을 처리해야 하므로 그로 인해서 생기는 시스템 자원의 소모가 생겼다. 이것은 제안한 알고리즘을 적용해서 생기는 단점이라고 볼 수 있다. 하지만 실험을 통해 시스템 자원 소모로 인해 생긴 인터페이스에서의 손실이 만회가 되고 있음을 보였다. 결국 본 논문에서 제안한 패킷 폐기 알고리즘을 별도의 임베디드 시스템으로 구현을 하게 되면 필터링을 위한 시스템 자원 소모를 줄일 수 있을 뿐 아니라 침입탐지시스템에 전달되는 네트워크 부하도 줄일 수 있어서 보다 효율적일 것으로 예상된다.

참 고 문 헌

[1] http://www.certcc.or.kr/.

[2] R. Heady, G. Luger, A. Maccabe, and M. Servilla, "The architecture of a network level intrusion detection system," Technical Report, Department of Computer Science, University of New Mexico, August, 1990.

[3] S. Kumar, "Classification and Detection of Computer Intrusions," PhD thesis, Purdue University, West Lafayette, IN 47907, pp.38-61, 1995.

[4] Sally Floyd, Van Jacobson, "Random Early Detection Gateways for Congestion Avoidance," IEEE/ACM Transactions on Networking, 1993.

[5] M. Christiansen, K. Jeffay, D. Ott, and F. Donelson Smith, "Tuning RED for web traffic," In ACM SIGCOMM2000, pp.4-12, August, 2000.

[6] B. Braden, D. Clark, j. Crowcroft, B. Davie, S. Deering, D.

Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, j. Wroclawski, L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet," RFC2309, 1998.

[7] Andrew McRae, "A Infrastructure for Deterministic Packet Classification," AUUG national conference, September, 1999.

[8] Charles D. Cranor, R. Gopalakrishnan, Peter Z. Onufryk, "Architectural consideration for CPU and network interface integration," In IEEE Micro, pp.18-19, January-February, 2000.

[9] S. McCanne and V. Jacobson, "The BSD Packet Filter : A New Architecture for Userlevel Packet Capture," In Proceedings of the 1993 Winter USENIX Conference, San-Diego, CA, pp.1-3, January, 1993.

[10] W. R. Stevens, "TCP/IP Illustrated, volume The Protocols of Professional Computing Series," Addison-Wesley, Vol.1, 1994.

[11] "NSFNET backbone traffic distribution by service," April, 1995. This document is avail-able at ftp://nic.merit.edu/nsfnet/statistics/1995/nsf-9504-ports.gz.

[12] K. Thompson, G. J. Miller, and R. Wilder, "Wide-Area Internet Traffic Patterns and Characteristics," IEEE/ACM Transactions on Networking, pp.10-23, November, 1997.

[13] http://www.snort.org/docs/lisapaper.txt.

[14] http://www.acme.com/software/http_load/.

[15] http://manimac.itd.nrl.navy.mil/MGEN/.

문 종 욱

e-mail : lache@ajou.ac.kr

2000년 아주대학교 전자공학과(학사)

2002년 아주대학교 전자공학과(석사)

2002년~현재 아주대학교 전자공학과 박사과정

관심분야 : 네트워크 보안, RTOS, 임베디드 시스템, 병렬처리 등

김 종 수

e-mail : promise@ajou.ac.kr

2000년 아주대학교 전자공학과(학사)

2002년 아주대학교 전자공학과(석사)

2002년~현재 아주대학교 전자공학과 박사과정

관심분야 : 초고속 통신망, 임베디드 시스템, 정보보안, RTOS 등

정 기 현

e-mail : khchung@ajou.ac.kr

1984년 서강대학교 전자공학과(학사)

1988년 Univ. of Illinois, EECS(석사)

1990년 Univ. of Purdue, 전기전자공학부(박사)

1991년~1992년 현대반도체 연구소

1993년~현재 아주대학교 전자공학부 교수

관심분야 : 컴퓨터구조, VLSI 설계, 멀티미디어 및 실시간 시스템 등

임 강 빈

e-mail : hotspot@csl.ajou.ac.kr

1992년 아주대학교 전자공학과(학사)

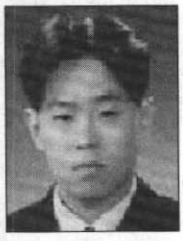
1994년 아주대학교 전자공학과(석사)

2001년 아주대학교 전자공학과(박사)

1999년~2000년 (미)아리조나 주립대 객원연구원

1997년~현재 아주대학교 정보통신전문대학원 전임연구원

관심분야 : 실시간 운영체제, 네트워크 보안, 내장형 시스템 등



주 민 규

e-mail : zzu@cesys.ajou.ac.kr

2000년 아주대학교 정보 및 컴퓨터 공학 졸업(학사)

2000년 아주대학교 정보통신 전문대학(석사)

2002년~현재 (주)뉴어텍 연구원

관심분야 : 임베디드 시스템, RTOS, 통신 프로토콜, 인터넷 QoS

최 경 희

e-mail : khchoi@ajou.ac.kr

1976년 서울대학교 사범대학 수학교육과(학사)

1979년 프랑스 그랑데꼴 ENSEIHT, 정보공학 및 응용수학(석사)

1982년 프랑스 Univ. of Paul Sabatier(박사)

1991년~1991년 프랑스 렌느 IRISA 연구소 교환교수

1982년~현재 아주대학교 정보 및 컴퓨터 공학부교수

관심분야 : 운영체제, 분산처리, 실시간 시스템 등