한국산업정보학회논문지 제6권 제1호 (2001. 3)

네트워크 침입 탐지 시스템에서의 시차 문제 해결 방안

(A Solution for Timing Gap Problems on Network Intrusion Detection Systems)

차 현 철*
(Hyun-Chul Cha)

요 약 본 논문에서는 네트워크 침입 탐지 시스템의 취약점 중 하나인, 패킷 검출 시간 차이에 기인한 공격의 발생 가능성을 줄일 수 있는 방법을 제시하였다. 이런 종류의 공격 가능성을 줄이기 위하여 네트워크 기반 침입 탐지 시스템과 목적지 시스템간에 왕복 지연을 측정한 후 지연을 고려하여 패킷의 목적지 시스템 도착 예정 시간을 추정하는 방법을 제시하였다. 제시된 방법의 성능을 측정하기 위해 LAN 상에서 소켓 프로그래밍을 이용하여 성능을 평가하였으며, 성능 평가 결과 시스템과 네트워크에 부담이 되지 않을 정도의 오버헤더만 있으면 패킷 검출 시차를 상당 부분 줄일 수 있음을 보여 준다. 특히, 가장 최근에 계산한 지연을 이용하는 최근 추정 방법이 가장 좋은 성능을 가짐을 알 수 있다.

Abstract This paper proposes a method to reduce the timing gap which causes a type of attack to a network-based intrusion detection system. For this, an intrusion detection system is used in order to measure the round trip time to the destination system, and the measured round trip time is used for estimation of packet's real arrival time to destination. Socket programings are used on the LAN environments to evaluate the performance of the proposed method. The test results show that an estimation method which uses the latest round trip time has the best performance, and the timing gap can be significantly reduced with small overheads.

1. 서 론

침입(intrusion)은 정보 접근, 정보 조작, 시스템 무력화 등 대상 시스템에 대한 고의적이며 불법적인 행위로 정의할 수 있으며, 침입 탐지 시스템(IDS; Intrusion Detection System)은 이러한 침입을 목적으로 특정 시스템에 불법적으로 접속하여 시스템을 사용, 오용, 남용하는 것을 감지하고 문제점을 처리하는 시스템이라 정의할 수 있다[1,2]. 현재 침입 탐지를 수행하는 다양한 종류의 침입 탐지 시스템들이 존재하며, 침입 탐지 시스템들은 엔진의 타입, 실시간 분석 여부 및 검사하는 데이터의 종류 등에 따라 몇 가지 방법으로 분류 될 수 있다. IDS의 엔진은 그 타입에 따라 통계적 비정상 탐지(statistical anomaly detection)와 패턴 매칭 탐지(pattern matching detection) 등으로 분류 할 수 있다. 실시간 실행 여부에 따른 분류에서, 보통 취약점 스캐너(vulnerability scanner)는 주기적으로 실행되는 반면 비정상 탐지기와 패턴 매칭기는 일반적으로 실시간으로 실행된다. 마지막으로 IDS가 검사하는 정보의 종류에 따라 네트워크 데이터와 시스템 데이터 등으로 나누어 볼 수 있다[3,4].

많은 침입 탐지 시스템들은, 단일 컴퓨터 시스템 상에서의 활동(activity)들의 의심스러운 패턴을 감시하기 위해, 운영체제가 제공하는 감사 로그(audit log)를 사용한다. 이런 형태의 IDS들을 호스트 기반(host-based) 침입 탐지 시스템이라 부르며, 로컬 사용자들에 의해 시작되는 단일 시스템의 오 사용(misuses)에 관련된 공격들을 인식하기에 적합하다. 그러나 이 시스템들은 단지 고 수준의 로그 정보만을 이용함으로 저 수준에서 발생하는 네트워크 이벤트

* 동양대학교 컴퓨터공학부 조교수

들에 대해서는 제한된 정보만을 갖게 되는 중요한 단점을 가진다.

네트워크 기반 침입 탐지 시스템(N-IDS; Network-based IDS)들은 네트워크 상에 전송되는 실제 패킷들의 내용을 검사하여 작동한다. 이들 시스템들은 패킷을 분석하고, 네트워크에서 사용되는 프로토콜들을 분석하여 그들로부터 관련되는 정보를 추출해낸다. N-IDS들의 장점을 살펴보면, 먼저, N-IDS는 간단히 구현될 수 있으며 단일 네트워크 상에 존재하는 여러 시스템에 관련된 공격을 식별하는 것 또한 용이하다. 두 번째, N-IDS는 네트워크의 가장 낮은 레벨에서 단순히 청취만 하므로 침입자가 이의 존재를 알기 어렵다. 그러므로 침입자에 의해 시스템이 폐쇄되거나 데이터가 손상되기 어렵다. 세 번째, 스니퍼(sniffer)의 설치로 인한 네트워크의 방해나 네트워크의 성능에 저하가 발생하지 않으므로, 네트워크 상의 다른 시스템들은 스니퍼의 존재에 대해 신경을 쓰지 않아도 된다. 마지막으로, N-IDS가 감시하는 TCP, UDP 등의 네트워크 프로토콜들은 각 시스템에서 사용하는 서로 다른 감사 로그와 달리 표준화되어 있으므로 다른 시스템들과 같이 사용되기 쉽다. 한편, N-IDS는 네트워크 상의 모든 패킷을 볼 수는 없으며, 네트워크 트래픽이 암호화 되어있을 경우 아무 것도 알 수 없다. 마지막으로, N-IDS가 최종 목적지 노드가 아니라는 단점을 갖는다[3,5].

결국, 호스트 기반 IDS는 네트워크 기반 IDS가 탐지할 수 없는 침입들을 검출 할 수 있으며 반대로 네트워크 기반 IDS는 호스트 기반 IDS가 탐지할 수 없는 침입들을 검출 할 수 도 있다[3]. 그러나 향후 확장일로에 있는 인터넷 등의 발전에 따라 네트워크 기반 공격은 더욱 늘어날 것이며 보다 일반적이고 복잡해 질 것이다. 이러한 이유로 인하여, 침입 탐지 시스템은 침입 활동에 대한 효율적인 탐지 도구를 제공하기 위해, 현재의 호스트 및 운영체제 중심에서 네트워크 자체나 네트워크 IDS로 그 중심을 옮겨 갈 것이다. 이런 종류의 시스템들로는 NSM, DIDS, EMERALD와 NetSTAT 등이 있다[5-7].

현재까지 N-IDS에 대한 많은 연구와 구현이 있어왔다. 현재의 모든 N-IDS는 기본적으로 약점이 많은 수동적 데이터 수집과 프로토콜 분석에 의존하고 있다. T. H. Ptacek과 T. N. Newsham은 그들의 논문 [8]에서 수동적 데이터 수집은 실제 컴퓨터 시스템 상에서 어떤 일들이 일어나고 있는지에 대한 정확한 결론을 얻기에는 통신매체 상에서 얻을 수 있는 정보가 불충분하다는 점을 지적하고 있다. 또한, 네트워크 IDS의 이러한 기본적인 취약점을 대상으로 하는 삽입(insertion) 공격, 배제(evasion) 공격 및 서비스 거부(denial of service) 공격을 정의하고, 현재 가장 많이 사용되고 있는 몇 가지 상용 N-IDS들에 대해 테스트한 결과 이들 모두 이러한 공격에 취약점을 가지고 있음을 보여준다.

네트워크 침입 탐지 시스템은 매우 중요함에도 불구하고 아직도 해결해야할 많은 문제점들을 가지고 있다. 본 논문에서는 N-IDS가 해결해야할 문제점들 중 특히, N-IDS와 그가 감시하는 시스템들이 패킷을 보게되는 시차에 기인하여 발생될 수 있는 문제를 살펴보고 이를 해결할 수 있는 방법을 제시하고자 한다.

논문의 구성은 다음과 같다. 1장의 서론에 이어 2장에서는 네트워크 침입 탐지 시스템의 취약점에 대해 분석한 후 시차 문제를 보완하는 방법을 제안하였다. 3장에서는 제시한 방법의 성능을 평가하기 위한 실험 환경을 기술하고 실험 결과를 분석하였으며 마지막으로 4장에서 결론을 다루었다

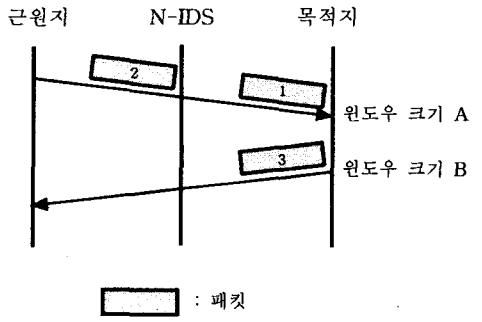
2. N-IDS에 대한 보안

2.1. N-IDS의 취약점

N-IDS는 네트워크에 접속된 컴퓨터들이 교환하는 패킷들을 스니퍼링(sniffering)하여 그들의 행동을 예측한다. 이런 수동적 네트워크 모니터링 방식은 감시하려는 컴퓨터 상에서 어떠한 일들이 발생하고 있는지 정확하게 예측할 수 없다는 단점을 가진다. 일반적으로 N-IDS는 감시하려는 컴퓨터와 완전히 다른 컴퓨터에서 수행되는 경우가 대부분이며 때로는, 완전히 다른 네트워크 상에 존재하기도 한다. 이러한 상이점으로 인해 N-IDS와 그가 감시하려는 시스템 사이에 불일치가 발생할 수 있다. 이러한 불일치는 기본적인 물리적 차이로 인해 발생하기도 하며, 운영체제나 네트워크 드라이브의 구현이 다르기 때문에 기인하기도 한다. 예를 들어, 단일 네트워크의 서로 다른 부분에 위치한 하나의 N-IDS와 하나의 종단 시스템을 고려해 볼 때, 이 두 시스템은 서로 다른 위치에서 서로 다른 시간에 주어진 임의의 패킷을 받게 된다. 이러한 시간적 차이는 중요한 의미를 갖는다. 이 시차 동안에 종단 시스템에서는 어떠한 일이 발생할 수 있으며 이로 인해 패킷의 수신이 되지 않을 수도 있다. 스니퍼 기반 IDS에 대해 특징 인식 시스템이 올바른 정보를 얻지 못하도록 프로토콜 분석을 방해하여 잘못된 결론에 이르게 하는 두 가지 공격이 존재한다. 삽입 공격에서 공격자는 N-IDS가 감시하는 종단 시스템은 받아들이지 않지만 N-IDS는 받아들이는 패킷들을 만들어 전송한다. 이 공격을 받는 N-IDS는 종단 시스템이 패킷들을 받아들이고 처리하였다고 잘못 판단하지만 실제로 종단 시스템에서는 그러한 일들이 일어나지 않는다. 공격자는 이렇게 함으로써, N-IDS에 잘못된 데이터를 삽입할 수 있게 된다. 배제 공격은 삽입 공격과 반대로 종단

시스템은 패킷을 받아들이지만 N-IDS는 패킷의 수신을 거부하는 경우에 발생한다. 배제 공격은 삽입 공격과 마찬가지로 N-IDS와 종단 시스템이 보게되는 문자열이 서로 상이하게 되므로 N-IDS의 패턴 매칭 과정의 실패를 초래할 수 있다[8].

현재 인터넷 등에서 가장 많이 사용되고 있는 TCP는 여러 가지 삽입 공격과 배제 공격에 대한 취약점을 가지고 있다. 이 중에서 N-IDS가 극복해야할 가장 큰 문제는 종단 시스템의 윈도우에 관련된 문제이다[8]. 한 연결의 윈도우는 다른 쪽이 너무 많은 데이터를 보내 버퍼링 될 수 없는 것을 방지하기 위해 받을 데이터 바이트의 수를 알리는 역할을 한다. 윈도우를 초과하여 보내진 데이터는 폐기된다[9]. 더구나 N-IDS가 윈도우 크기의 변경을 검출하는 시간은 종단 시스템이 변경을 검출하여 거기에 반응하는 시간과 다르게 된다. N-IDS와 종단 시스템의 검출 시간 차이에 도착한 패킷들은 문제가 된다. 이런 문제를 고려하지 않는 N-IDS는 삽입 공격에 취약점을 갖는다. <그림 1>에서는 N-IDS에서 이런 문제가 발생할 수 있는 예를 보여준다.



<그림 1>. 시차에 따른 삽입 공격 발생 예

이 그림에서, 근원지에서 보낸 패킷 1이 목적지에 도착하였을 때의 목적지 윈도우 크기가 A이었고 패킷 1을 받은 후 어떤 이유에서 윈도우 크기를 B로 축소 변경하였다고 가정한다. N-IDS는 되돌아가는 패킷 3을 볼 때까지는 윈도우 변경 사실을 알 수 없다. 그러므로 만약 N-IDS가 패킷 2를 패킷 3보다 먼저 보게 된다면 N-IDS는 패킷 2를 받아들이게 되나 목적지는 윈도우 크기 축소 변경에 따라 패킷 2를 폐기할 수 있다. 또한, 네트워크 침입의 경우에는 통신 프로토콜의 전제 조건인 상호 협력이 만족되지 않으므로 이외에도 매우 다양한 예외 조건이 발생할 수 있다. 물론, N-IDS가 확인(acknowledgement)이나 재전송 패킷 등을 보고 TCP 스트림을 적절히 재구성할 수 도 있으나 개개의 패킷에 대한 확인 패킷이 만들어지는 것은 아니며, 더구나 N-IDS는 TCP 연결 상에 수동적 참여자이므로 재전송 등을 요구할 수 없으므로 데이터를 잃어버리기 쉽다.

2.2. N-IDS의 취약점 보완 방법

N-IDS와 그가 감시하는 시스템들이 패킷을 보게되는 시차에 기인하여 많은 문제들이 발생한다. 그러므로, 본 논문에서는 N-IDS와 종단 시스템간의 패킷 검출 시간 차이를 해결할 수 있는 방법을 제시하고자 한다. N-IDS와 종단 시스템은 서로 상이한 시스템에 존재하는 경우가 대부분이므로 검출 시간 차이를 완전히 해결할 수는 없을 것이다. 하지만 N-IDS에서 검출 시간의 차이를 줄일 수 있는 방법을 고려해 볼 수 있다. 즉, N-IDS에서 종단 시스템까지 패킷의 전송에 소요되는 시간을 예측할 수 있다면 N-IDS는 자신을 통과한 패킷이 언제쯤 종단 시스템에 도착할 것인가를 예측할 수 있게 된다.

이를 위해 N-IDS가 종단 시스템에 주기적으로 패킷을 생성하여 전송하고 종단 시스템은 이에 대해 응답하게 할 수 있다. 이들 패킷간의 출발, 도착 시간 차이는 N-IDS와 종단 시스템간의 왕복 지연(round-trip time)이 되며 N-IDS는 이 왕복 지연을 실제 TCP 패킷의 종단 시스템 도착 예정 시간을 구하기 위해 사용할 수 있다. 일반적으로 N-IDS는 망의 초입에 위치하게 되므로 N-IDS가 종단 시스템보다 패킷을 먼저 보게 된다고 가정할 수 있다. 임의의 패킷 i가 근원지에서 출발하여, N-IDS에 도착하는 시간을 t_{I_i}라하고 종단 시스템에 도착하는 시간을 t_{D_i}라 하면, N-IDS와 종단 시스템간의 도착 시간 차이 t_{G_i}는 식(1)과 같이 나타낼 수 있다.

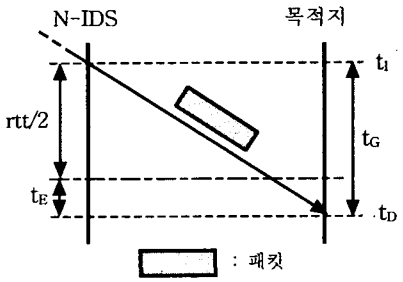
$$t_{G_i} = t_{D_i} - t_{I_i} \geq 0 \qquad (1)$$

본 논문에서는 t_{G_i}를 감소시키기 위해 부가적인 정보를 사용하기로 한다. 임의의 한 패킷이 전송되는데 소요되는 전송 지연은 전송되는 데이터의 양과 회선의 전송률에 따라 가변적이다. TCP 패킷은 가변 길이를 가지며 LAN 상의 전송률 역시 트래픽의 양에 따라 달라지므로 전송 중인 패킷의 전송 지연을 정확히 알 수는 없다. 본 논문에서는 이를 추정하는 방법을 사용한다. 추정을 위해 ICMP 메시지 등이 사용될 수 있다.

N-IDS와 종단 시스템간에 패킷의 왕복 지연(round trip time)을 rtt로 표기할 때, N-IDS는 주기적으로 종단 시스템에 요청 메시지들을 전송하여 되돌아오는 응답 메시지들을 보고 N-IDS와 종단 시스템간의 각각의 왕복 지연인 rtt_i를 계산한다. N-IDS는 종단 시스템으로 향하는 TCP

패킷을 스니퍼링 하였을 때 즉시 TCB(TCP Control Block)나 윈도우 크기 변경 등의 각종 변경을 수행하지 않고, 이전에 계산된 rtt_i 의 절반 값 이후에 변경을 수행하면 결국 t_{G_i} 값을 t_{E_i} 값으로 줄일 수 있게 된다. 이를 수식으로 표현하면 식 (2)와 같으며, N-IDS와 목적지 시스템간의 시간 관계를 <그림 2>에 보였다.

$$t_{E_i} = t_{D_i} - (t_{I_i} + rtt_i/2) \qquad (2)$$



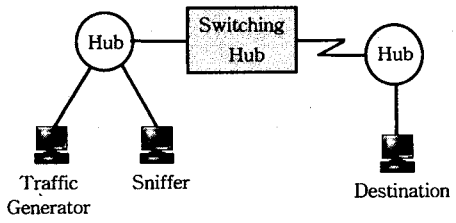
<그림 2>. N-IDS와 목적지간의 시간관계

결국, N-IDS는 자신이 보호하려는 종단 시스템과의 왕복 지연을 주기적으로 측정하고 이를 고려하여 윈도우 크기 등의 관련 정보를 수정하게 된다. 그러므로, 즉시 관련 정보를 수정하는 것에 비해 패킷 검출 시차를 줄일 수 있게 되며 궁극적으로는 삽입 공격의 가능성을 줄일 수 있게 된다.

3. 실험 및 고찰

3.1. 실험 환경

본 논문에서 제안한 방법의 효과를 분석하기 위해 <그림 3>과 같은 환경에서 실험을 실시하였다. 100Base-T 방식의 Fast Ethernet 근거리 통신망 환경에서, 목적지 시스템과 스니퍼 시스템은 둘 다 SUN Ultra 1 워크스테이션이며 운영체제는 Solaris 2.6이 사용되었다. 트래픽 생성기(traffic generator)는 SUN Sparc 10 워크스테이션 상에 Solaris 2.5.1 운영체제를 사용하고 있다.



<그림 3>. 실험 환경

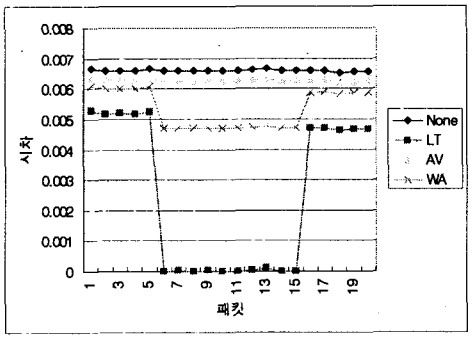
트래픽 생성기에서는 소켓 프로그래밍[10]을 사용하여 작성한 프로그램을 실행하여, 목적지 시스템과 TCP 연결을 설정한 후 1024 바이트 같은 크기의 패킷들을 생성하여 목적지 시스템으로 송신한다. 목적지 시스템은 트래픽 생성기가 보내온 패킷에 대해 단순히 확인 패킷을 만들어 응답하며 데이터 패킷을 보내지는 않는다. 스니퍼와 목적지 시스템에서는 각각 스니퍼링 소프트웨어인 tcpdump[11,12]를 사용하여 패킷의 도착 시각을 기록하였다. 또한, 스니퍼에서는 주기적으로 ICMP timestamp request 패킷을 만들어 목적지 시스템으로 전송하여 되돌아오는 ICMP timestamp reply 패킷으로부터 스니퍼와 목적지 시스템간의 왕복 지연을 측정한다. ICMP 패킷의 생성은 로우(raw) 소켓을 사용하였다. 스니퍼에서는 TCP 패킷이 자신에게 도착한 시간에 임의의 왕복 지연 값을 고려하여 해당 패킷의 목적지 시스템 도착 예정 시간을 추정하게 되며 이 때, 사용된 왕복 지연은 세 가지 종류를 고려하였다. 즉, 가장 최근에 계산된 왕복 지연인 rtt_{LT} 를 사용하는 최근 추정, 왕복 지연의 평균인 rtt_{AV} 를 사용하는 평균 추정, 마지막으로 식 (3)의 가중 평균 rtt_{WA} 를 이용하는 가중 평균 추정 방법을 사용하였다.

$$rtt_{WA} = w \times rtt_n + (1-w) \times \frac{\sum_{i=1}^{n-1} rtt_i}{n-1}, \quad w \text{ is weight and } 0 \le w \le 1 \qquad (3)$$

3.2. 실험 결과

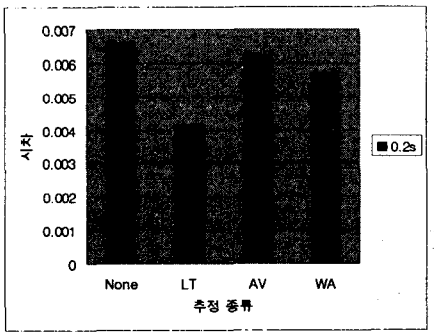
<그림 4>에서는 본 논문에서 제시한 세 가지 추정 방법의 동작을 설명하기 위해 실험 결과중 임의의 일련의 20개 패킷의 도착 시차를 보였다. 스니퍼에서는 0.2초 주기로 왕복 지연을 계산하였고 가중평균 계산을 위한 w 는 0.25를 사용하였다. 이 그림에서 None으로 표시된 선은 스니퍼와 목적지간의 도착 시차를 나타내며 거의 동일한 값을 가짐을 알 수 있다. LT, AV, WA로 표시된 선은 각각 최근 추정, 평균 추정, 가중평균 추정에 따라 추정된 시차를 나타내며 세 가지 추정법 모두 추정을 사용하지 않을 경우에 비해 더 작은 시차를 가짐을 보여준다. 즉, 실제 목적지 시

스템에의 도착시간에 보다 근접함을 알 수 있다. 이 그림에서 처음 5개, 다음 10개, 마지막 5개가 각각 같은 왕복 지연을 사용하고 있다. 특히, 중앙에 위치한 10개 패킷의 경우 ICMP 패킷의 지연과 TCP 패킷의 지연이 거의 일치하여 최근 추정의 경우 거의 시차가 발생하지 않음을 알 수 있다. 또한, 가중평균 추정의 경우가 평균 추정의 경우보다 왕복 지연의 변동에 더 민감하게 반응함도 알 수 있다.



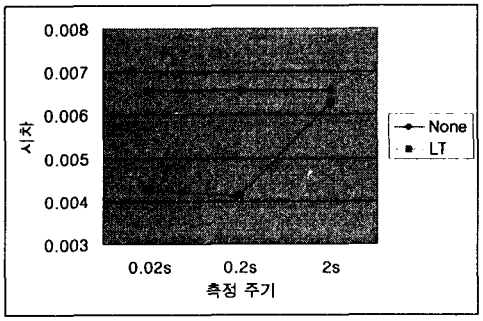
<그림 4>. 각 추정 방법의 동작 예

<그림 5>는 1K 바이트 크기의 TCP 패킷 500개가 전송되고, 0.2초 주기로 왕복지연을 측정한 경우에 세 가지 추정법의 평균 시차를 보여준다. 세 가지 추정법 모두 추정을 하지 않을 경우보다 더 좋은 성능을 가짐을 알 수 있으며 최근 추정이 가장 좋은 성능을 가진다. 최근 추정, 평균 추정, 가중평균 추정은 추정을 사용하지 않을 경우에 비해 각각 38%, 6%, 14%의 성능 향상을 가진다.

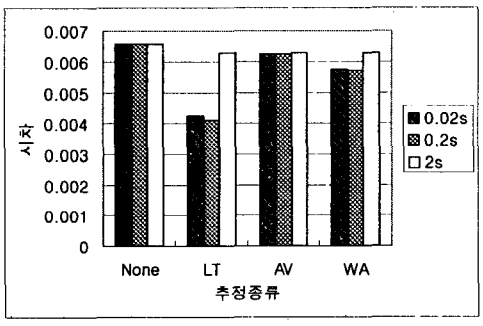


<그림 5>. 각 추정 방법의 평균 시차 분석

<그림 6>에서는 왕복 지연의 측정 주기가 미치는 영향을 분석하기 위해 1KB의 패킷 500개를 전송하였을 경우에 측정 주기의 변화에 따른 최근 추정법의 성능 변화를 보였다. 측정 주기는 0.02초에서 2초까지 3가지 경우로 하였다. 측정 주기는 0.2초인 경우가 가장 좋은 성능을 보이고 있다. 0.02초의 경우가 0.2초에 비해 성능이 조금 떨어지며 이것은 너무 잦은 왕복 지연 값의 변경이 오히려 정확한 추정에 방해가 됨을 보여 준다. 측정 주기가 2초인 경우에는 최초 한 번의 측정을 모든 추정에 사용하게 되며 성능이 급격히 나빠짐을 알 수 있다. 본 논문에서는 N-IDS와 목적지 시스템간의 패킷 검출 시차를 줄이기 위해 부가적으로 ICMP 패킷을 생성한다. 그러나 빈번한 ICMP 패킷의 생성은 스니퍼와 목적 시스템 또, 네트워크에 부담이 되므로 가능한 한 측정 주기를 늦추는 것이 유리할 것이다. 실험 결과 0.2초 정도의 측정주기로 38%의 성능 향상을 가져옴을 알 수 있으며 0.2초당 ICMP 패킷 하나를 생성하는 것은 시스템이나 네트워크에서 부담이 되지 않을 것이다.



<그림 6>. 측정 주기의 영향 비교



<그림 7>. 측정 주기별, 측정 종류별 비교

<그림 7>에서는 500KB의 데이터가 전송되었을 때 측정 주기별, 측정 종류별 성능을 나타내었다. 왕복 지연의 측정 주기에 관계없이 최근 추정이 가장 좋은 성능을 보이며, 추정 방법에 관계없이 측정 주기 0.2초인 경우가 가장 좋은 성능을 보여준다.

4. 결 론

침입 탐지 시스템의 중요성은 인터넷 등의 네트워크 발전과 더불어 날로 그 중요성을 더해가고 있다. 본 논문에서는 침입 탐지 시스템을 살펴보고 특히, 네트워크 침입 탐지 시스템에 초점을 맞추어 그 취약점을 고찰하였다. N-IDS는 아직 해결되지 않은 몇 가지 약점을 가지고 있으며 본 논문에서는 그 중에서 특히, N-IDS와 종단 목적지 시스템간에 패킷의 검출 시간 차이에 기인하여 발생하는 공격의 가능성을 줄일 수 있는 방법을 제시하였다. 즉, 삽입 공격과 배제 공격의 가능성을 줄이기 위하여 N-IDS와 목적지 시스템간에 지연을 측정한 후 지연을 고려하여 패킷의 목적지 시스템 도착 예정 시간을 추정하는 방법을 제시하였다. 이를 위해 N-IDS는 주기적으로 ICMP 패킷을 생성하여 자신이 보호하려는 종단 시스템에 전송한 후 되돌아오는 응답 메시지를 받아 지연을 계산한 후 이 지연 이후에 N-IDS에서 관리하는 각종 정보들을 변경하게 된다.

제시된 방법의 성능을 측정하기 위해 Fast-Ethernet LAN 상에서 소켓 프로그래밍을 이용하여 성능을 평가하였으며, 성능 평가 결과 시스템과 네트워크에 부담이 되지 않을 정도의 부담만 있으면 패킷 검출 시차를 상당 부분 줄일 수 있음을 알 수 있었다. ICMP 패킷의 생성 주기는 0.2초 정도의 주기를 가질 때 가장 좋은 성능을 보여주며 아울러 가장 최근에 계산한 지연을 이용하는 최근 추정 방법이 가장 좋은 추정 성능을 보여줌을 알 수 있었다.

본 논문의 향후 연구과제는 실제 계산된 지연을 이용하여 N-IDS가 TCP 스트림을 재구성하는 방법과 실제 네트워크 환경에서 이를 구현하는데 필요한 추가적인 연구가 수행될 것이다.

참고 문헌

[1] D. E. Denning, "An Intrusion-Detection Model," Proc. of the IEEE Symposium on Security and Privacy, pp. 118-131, 1986.

[2] 한국정보보호센터, 실시간 네트워크 침입 탐지 시스템 개발에 대한 연구, Dec., 1998.

[3] T. Escamilla, Intrusion Detection Network Security Beyond the Firewall, Addison-Wesley, 1998

[4] S. Kumar, E. Spafford, "A pattern matching model for misuse intrusion detection," Proc. of the 17th National Computer Security Conference, pp. 11-21, Oct., 1994.

[5] A. Sundaram, "An Introduction to Intrusion Detection," http://www.acm.org/crossroads/xrds2-4/intrus.html

[6] G. Vigna, R. A. Kemmerer, "NetSTAT : A Network-based Intrusion Detection Approach," Proc. of ACSAC'98, 1998.

[7] 김병구, 정태명, "침입탐지 기술의 현황과 전망," 정보과학회지, 18권 1호, pp.29-39, Jan., 2000.

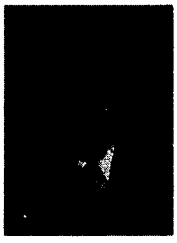
[8] T. H. Ptacek, T. N. Newsham, "Insertion, Evasion and Denial of Service : Eluding Network Intrusion Detection," http://www.nai.com/products/security/advisory/papers/ids.pdf, Jan., 1998.

[9] W. R. Stevens, TCP/IP illustrated, Vol. 1, Addison-Wesley, 1994.

[10] W. R. Stevens, UNIX Network Programming, Prentice-Hall, 1990.

[11] libcap software, ftp://ftp.ee.lbl.gov/libcap.tar.Z

[12] tcpdump software, ftp://ftp.ee.lbl.gov/tcpdump.tar.Z



차 현 철 (Cha Hyun-Chul)

1988 경북대학교 통계학과(학사)
1993 경북대학교 컴퓨터공학과(석사)
1998 경북대학교 컴퓨터공학과(박사)
2000 Arizona State Univ.(Post-Doc.)
1995-현재 동양대학교 컴퓨터공학부 조교수

관심분야: B-ISDN/ATM, Wireless ATM, 광 인터넷, 네트워크 보안, 침입 탐지