情報保護學會論文誌
第16卷 第3號, 2006. 6

침입탐지시스템에서 포트 스캔 탐지 개선 및 공격 탐지와 연계한 알고리즘 설계 및 구현

박 성 철,[1†] 고 한 석[2‡]

[1]서울호서전문학교 인터넷정보보안과, [2]고려대학교 전자컴퓨터공학과

Design and implementation of port scan detection improvement and algorithm connected with attack detection in IDS

Seong-Chul Park[1†], Han-Seok Ko[2‡]

[1]Department of Internet Information Security, Seoul Hoseo College

[2]Department of Electronics and Computer Engineering, Korea University

요 약

본 논문에서는 침입탐지시스템[1]의 탐지 기법인 포트스캔 탐지에 대한 개선 및 포트스캔 탐지 결과와 연계하여 실질적인 공격 탐지 부분인 오남용(Misuse) 탐지 방법의 네트워크 기반 침입탐지시스템에 대한 탐지 능력을 극대화하는 방법에 대해 연구하였다. 또한 침입탐지시스템에서 개선된 포트스캔 탐지를 위해 전처리기인 포트스캔 탐지에 대한 일반적인 정책설정의 문제점과 오남용 탐지 엔진의 false-positive[2]를 최소화하고 포트스캔 탐지와 오남용 탐지의 수행 성능을 높이기 위한 알고리즘을 연구하였다.

ABSTRACT

This paper deals with an effective algorithm aimed at improving the port scan detection in an intrusion detection system (IDS). In particular, a detection correlation algorithm is proposed to maximize the detection capability in the network-based IDS whereby the "misuse" is flagged for analysis to establish intrusion profile in relation to the overall port scan detection process. In addition, we establish an appropriate system maintenance policy for port scan detection as preprocessor for improved port scan in IDS, thereby achieving minimum false positive in the misuse detection engine while enhancing the system performance.

Keywords : *Port Scan, False Positive, Association*

Ⅰ. 서 론

1.1 배경과 목적

현대사회는 인터넷의 발달로 정보의 공유가 활발해지면서 사회적 부가가치를 증대 시키고 있다. 이러한 이면 뒤에는 정보의 노출 및 각종 침해 행위들이 몇 년 사이에 배로 늘어나 사회적 문제로 대두되고 있다[3,4]. 지식 정보사회의 기반으로 자리 잡은 네트워크 환경으로서 인터넷은 기본 프로토콜인 TCP/IP 프로토콜[5]을 이용하여 시스템과 시스템 사이에 데이터를 주고받는다. 그런데 이 프로토콜의 설계상 취약점으로 인해 네트워크를 이용하는 것만으로도 정보시

접수일: 2006년 1월 25일; 채택일: 2006년 6월 3일
† 주저자, scpark@hoseo.or.kr
‡ 교신저자, hsko@korea.ac.kr

스템이 외부에 노출되어 침입자의 공격에 놓이게 되었다. 따라서 인터넷에 연결된 각종 정보 시스템을 보호하기 위해 시스템이 갖고 있는 자체 취약성을 분석하거나 각종 기관에서 네트워크의 접근제어를 위해 침입차단시스템 등의 보안시스템을 구축한다.

침입차단시스템[6]은 외부 침입을 차단하기 위해 기업 내부 네트워크와 외부 네트워크 사이에 위치하며 오고 가는 패킷의 상태를 파악하여 차단하는 기술이다. 그러나 컴퓨터 기술의 발전은 해킹 기술의 발전에 영향을 주어 크래커 및 해커의 침입을 차단하는 데는 한계가 있다. 또한 공격의 행태를 분석할 때 외부 침입보다 내부 사용자에 의한 공격이 더욱 더 심각하며 이러한 내부 사용자에 의한 공격은 침입차단시스템에 있어서 아무런 대책도 마련할 수 없다.

그러므로 내부 및 외부 사용자에 관계없이 공격을 효과적으로 막아낼 수 있는 침입탐지시스템이 필요하다. 그런데 침입탐지시스템은 공격 탐지에 있어서 Signature를 이용한 패턴 매칭으로 인해 오탐율이 심각하여 보안 관리자의 업무를 효율적으로 돕기보다는 혼란을 가중시키고 있다.

따라서 본 논문에서는 침입탐지시스템의 탐지 능력의 정확성을 향상시키기 위해 침입탐지시스템의 기능에 해당하는 포트스캔 공격 탐지를 그 자체로써의 정보로서 활용하는 데 그치는 것이 아니라 포트스캔 탐지 결과와 연계하여 침입탐지의 false-positive에 대한 감소를 위한 Correlation을 연구하였다. 그리고 Signature에 의한 침입탐지와 포트스캔 공격 탐지 결과를 서로 효과적으로 연계하기 위해서는 포트스캔 탐지에 대한 개선점을 연구하였다.

1.2 연구 내용 및 구성

일반적으로 침입탐지시스템은 공격 패킷을 찾기 위해 단순히 패턴 매칭에 의존한 탐지 엔진이 설계되었다.[7] 그 이유는 첫째, 실시간으로 유입되는 패킷이 대단히 많다는 것을 들 수 있고, 둘째, 유입되는 전체 패킷에서 공격에 관련된 패킷만을 따로 분리하여 탐지 엔진에 적용한다고 하여도 그 양은 대단히 많으며 계속적으로 그것을 관리한다는 것은 쉽지 않은 일이다. 그리고 대단히 많은 양에 대해 탐지 엔진이 효율적으로 성능을 유지[7]하며 Correlation을 찾는다는 것은 엄청난 부하를 가질 것이다. 그래서 포트스캔 탐지에 대해서 효과적으로 관리하고, 탐지 엔진의 결과를 개선하기 위해 적용할 수 있는 '개선된 포트스캔 탐지 및 탐지 Correlation 알고리즘'을 제안한다.

본 논문에서 제시한 개선된 포트스캔 탐지 및 탐지 Correlation 알고리즘의 주요 목적은 첫째, 특정시간 내에, 어느 정도의 스캔이 탐지되지 않으면 삭제하는 일반적인 포트스캔 탐지 방법과 Slow (Stealth) Scan[8]은 너무 시간이 지연된다는 이유로 활용되지 못하며, 일반 패킷도 스캔으로 사용되지만 탐지에 이용되지 못하는 이러한 점들을 개선할 것이다. 둘째, 포트스캔 공격을 효율적으로 관리할 방안으로 해시 테이블과 이에 삽입되는 정보는 최대한 간소화하여 구성한다. 셋째, 해시 테이블에 삽입된 포트스캔 정보를 탐지 엔진의 경고와 연계한 Correlation을 적용하는 것이다. 이러한 점들이 개선되어 탐지에 적용된다면 훨씬 향상된 침입탐지시스템이 될 것이다.

본 논문의 Ⅱ에서는 개선된 탐지 알고리즘을 설계하기 위해 포트스캔 탐지 취약점 분석 및 개선 설계를 하고, Ⅲ에서는 개선된 포트스캔 탐지를 공격탐지 엔진과 연계를 통한 Correlation의 설계에 대해 기술하였다. Ⅳ에서는 구현된 프로그램을 위한 테스트 환경을 구성하여 실험 및 평가를 제시하였다. 마지막 결론에서는 본 연구의 결과를 평가하고 향후 연구 방향에 대해 논의하였다.

Ⅱ. 개선된 포트스캔 탐지 알고리즘 설계

2.1 포트스캔 탐지 취약점 분석

여기서 포트스캔 탐지의 취약점을 분석하기 위해 다른 각도로 3가지를 언급한다.

첫째, 특정시간 내에, 어느 정도의 포트스캔이 탐지되지 않고 정책 설정에 따른 시간이 지나면 삭제하는 일반적인 포트스캔 탐지 방법.

공개되었거나 상용되고 있거나 할 것 없이 대다수의 침입탐지시스템은 포트스캔에 대한 탐지를 다룬다. 그런데 특정시간 내에, 어느 정도의 스캔 threshold를 넘어 탐지되지 않으면 짧은 시간 내에 삭제하는 것이 일반적인 포트스캔 탐지 방법이다. 그러나 포트스캔을 수행한 정보를 짧은 시간 내에 참고하여 경고만 보내고 삭제된다면 침입탐지시스템은 중요한 정보를 잃게 될 것이다. 왜냐하면 포트스캔이란 우연히 생성되는 패킷이 아니고 공격자가 시스템을 공격할 때 포트에 어떠한 취약점이 있는 지 또는 네트워

크 내에서 어떠한 시스템이 취약한 지를 알아보거나 아니면 공격자가 원하는 시스템이 있는 지를 파악하기 위해서 행해지는 사전 탐색의 의미를 갖는다. 그래서 포트스캔을 행한 공격자에 대한 정보를 누적하여 활용할 수 있다면 침입탐지시스템은 공격 대처에 좀 더 적극적이고 능동적인 시스템이 될 것이다.

둘째, 침입탐지 정책 설정에 따라 일정시간 안에 적은 수의 포트스캔과 함께 긴 시간 동안 포트스캔이 이루어져 무시되는 Slow(Stealth) 포트스캔 탐지 방법.

포트스캔을 짧은 시간 내에 행하여 시스템의 정보를 파악할 수 있음에도 불구하고 영리한 공격자는 자신을 노출시키지 않을 목적으로 1~2주, 또는 한 달 정도의 긴 시간을 계획하여, 계획된 시간 내에 아주 조금씩 시간 편차를 두어 스캔을 보낼 가능성은 존재한다. 특히 특정시간 내에, 포트스캔 threshold에 침입탐지시스템이 민감하다는 것을 잘 알고 있는 공격자라면 더욱이 Slow(Stealth) 포트스캔을 애용할 것이다. 침입탐지시스템은 포트스캔을 수행한 정보를 짧은 시간 내에 삭제하지만 아주 느리게 오는 Slow 포트스캔은 아예 무시해 버리는 경향이 있다. 그러나 첫 번째의 포트스캔 정보보다 두 번째 다루고 있는 포트스캔은 더욱 더 네트워크에 있는 시스템에 치명적이고 침입탐지시스템의 탐지에 있어서 아주 활용 가치가 클 것이다.

셋째, 정상적인 패킷으로 가장하여 포트스캔을 행을 시 탐지하지 못하는 포트스캔 탐지 방법.

일반적으로 포트스캔이란 어떠한 정형적인 방법이 존재한다고 생각하지만 실질적으로는 대부분 비정상적인 패킷을 생성하여 보안 시스템을 우회하기 위해 만들어졌다. 그런데 패킷의 한 단위로만 분석한다면 아무런 이상이 없어 보이는 패킷이 단지 포트스캔을 목적으로 사용될 수 있다. 어떻게 보면 두 번째 언급한 내용과 비슷한 점이 있지만 첫 번째 언급한 내용과 비교하자면 일반적인 포트스캔 탐지란 TCP NULL Scan, TCP SYN/FIN Scan, TCP FIN Scan, TCP Xmas Scan 같은 비정상적인 패킷을 탐지하는데 한정된다. 그러나 일반적인 정상 패킷도 포트스캔 탐지에 포함될 수 있다면 아주 강도 높은 침입탐지시스템이 되는데 기여할 것이다.

위에서 언급한 3가지의 가장 근본적인 문제점은 포트스캔의 특성에 의해 엄청난 양의 로그를 발생시킨다는 것이다. 그래서 엄청난 양의 포트스캔탐지 정보를 저장하는 공개 IDS인 Snort는 특정 형태의 포트스캔 패킷만을 저장하며 일정 기간 내에 어느 정도의 Threshold가 넘지 않으면 무시하며, 일정 시간 내에 포트스캔이 다시 오지 않으면 포트스캔 탐지 정보를 삭제한다. [그림 1]은 공개 IDS인 Snort의 복잡한 포트스캔탐지 정보의 자료 구조를 분석하기 위한 것으로, 각 박스는 포트스캔 탐지 정보와 관련된 것이다. 제일 상단에 존재하는 ScanList 박스는 포트스캔을 시도하고 있는 공격자 정보들의 시작과

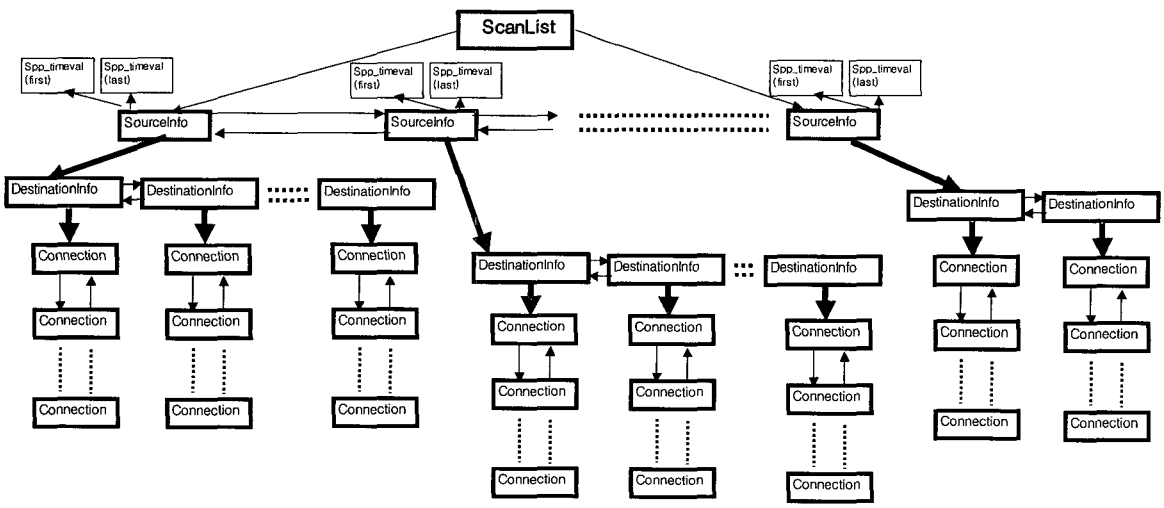

그림 1. 공개 IDS인 Snort의 포트스캔탐지 정보의 단면 자료 구조도

끝을 나타내는 포인터이며 12bytes의 용량을 차지한다. 그리고 SourceInfo 박스는 공격자들의 소스 IP로부터 시작해 78bytes, Destination 박스는 공격자가 포트스캔을 목표하는 호스트의 IP를 시작으로 20bytes, 마지막으로 Connection은 공격자가 사용하는 포트 번호와 목표 대상이 되는 포트 번호를 시작으로 최소 31bytes부터 최대 1491bytes까지 메모리를 차지할 수 있다. 만약 공격자가 한 호스트를 대상으로 1번에서 65535번까지 포트스캔을 한다면 Snort는 최소 약 2M부터 최대 97M까지 메모리 용량을 차지한다. 이러한 포트스캔을 Snort가 관리하는 하나의 로컬 네트워크 내에서 100대의 호스트를 대상으로 한다면 최소 200M에서 최대 9.7G까지 메모리 용량을 차지하게 된다.

2.2 개선된 포트스캔 탐지 정보의 구성 설계

개선된 포트스캔에 대한 탐지를 하기 위해서는, 2.1에서 언급한 포트스캔의 취약점 중 첫 번째, '특정시간 내에, 어느 정도의 포트스캔이 탐지되지 않고 정책 설정에 따른 시간이 지나면 삭제하는 일반적인 포트스캔 탐지 방법'을 보완하기 위해, 특정 시간 내에 포트스캔이 탐지되지 않고 일정한 시간이 지나면 정보들을 삭제하는 정책설정은 새로이 설계되는 전처리기에서 개선하기로 한다. 두 번째, '침입탐지 정책 설정에 따라 일정시간 안에 적은 수의 포트스캔과 함께 긴 시간 동안 포트스캔이 이루어져 무시되는 Slow(Stealth) 포트스캔 탐지 방법'은 시간이라는 요소는 배제되기 때문에 신경 쓰지 않아도 된다. 마지막으로, '정상적인 패킷으로 가장하여 포트스캔을 했을 시 탐지하지 못하는 포트스캔 탐지 방법'이라는 항목은 Captured 패킷만 보고 정상적인지 아닌지 구분하는 방법이 없다. 그러나 세션을 관리하여 Stateful Inspection을 통해 정상인지 아닌지 판별하게 되며, 비정상적일 경우 포트스캔 탐지 테이블에 저장할 수 있다.

2.2.1 포트스캔 탐지 정보 저장 구조 및 버킷에 연결된 슬롯의 구성 내용

포트스캔 탐지 정보를 장시간 보관하고 정보의 삽입, 수정 및 검색이 용이 하도록 해시 테이블을 사용해 Key에 연관되는 정보는 Linked List로 관리한다. 포트스캔 탐지 정보를 저장할 해시 테이블의 구조는 [그림 2]와 같다.

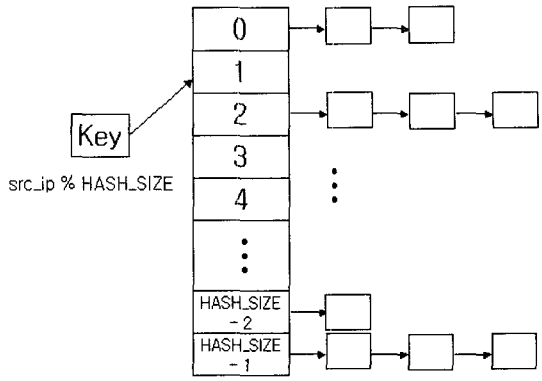

그림 2. 해시 테이블을 이용한 탐지된 포트스캔 정보의 저장

해시 테이블에서 Key는 Source IP가 되며, 포트스캔 탐지 정보구성은 [표 1]과 같이 Source IP, Protocol Type, 최초 탐지된 시각, 마지막 탐지 시각, 포트스캔이 탐지된 시간, 탐지된 포트스캔 Count, Frequency Count, 포트스캔이 탐지되지 않은 시간, Next Slot Address 등으로 구성한다. 탐지된 포트스캔의 모양은 궁극적으로 해시 테이블을 구성하며, [그림 2]처럼 추상화된 모습으로 그려질 수 있다.

표 1. 해시 테이블에서 포트스캔 탐지 정보 Slot의 구성

Field Name	Description
Source IP	포트스캔을 시도한 공격자 IP
Protocol Type	포트스캔 패킷의 Protocol Type
최초 탐지된 시각	공격자의 IP가 처음으로 포트스캔의 패킷이 인식된 시각
마지막 탐지시각	공격자의 IP가 마지막으로 포트스캔의 패킷이 인식된 시각
포트스캔이 탐지된 시간 (Active Time)	실제로 포트스캔이 탐지되는 것의 시간(초)을 누적한 결과
탐지된 Port Scan Count	포트스캔 시 보내어진 최초부터 패킷의 누적된 횟수
Frequency Count	Attacker의 Source IP가 포트스캔을 행한 후 30초 지난 뒤 탐지되면 횟수를 1만큼 증가
포트스캔이 탐지되지 않은 시간 (Idle Time)	Attacker의 Source IP가 포트스캔을 했던 후부터 다음 포트스캔이 탐지된 사이의 시간
Next Slot Address	이미 탐지된 포트스캔 정보의 Slot 주소를 저장하여 Linked List로 연결하게 될 다음 주소 저장

2.2.2 탐지된 포트스캔 정보의 구성 과정

해시 테이블에 정보가 처리되는 것을 살펴보면, 먼저 패킷이 포트스캔 처리기에서 탐지되어 Source IP와 해시 테이블의 Size에 의해 해시 값이 결정되고 IP 및 Protocol에 의해 해당하는 슬롯이 있는지 검사한다. 그런 후, 슬롯이 존재하지 않으면 포트스캔 정보 크기만큼 동적으로 메모리에 할당되어 슬롯 리스트에 추가되고, 존재하면 해당하는 슬롯의 정보 중 Idle Time, Port Scan Count, Frequency Count 및 Active Time을 수정하게 된다. 설명한 것의 흐름은 [그림 3]과 같다.

위의 [그림 3]에서 소스 IP의 존재 유무를 판별하는 루틴이 존재하는 데, 이것은 해시 Key가 충돌될 경우이다. 해시 알고리즘에서 가장 좋은 검색 성능을 발휘할 때는 Key가 골고루 각각의 버킷마다 분산되어 단일 유형의 슬롯이 존재할 경우이다. 그러나 Key가 충돌할 경우는 아직 해결되지 않았지만 문제를 완전히 해결하지 못한다고 해도 최선의 방법은 존재하는 데, 순차 연결리스트 대신 검색 성능이 좋은 알고리즘을 사용하는 것이다. 바로 이진 검색 트리 같은 알고리즘이 이에 해당한다. 이에 대해서는 검색 성능을 평가하는 4.3절에서 좀 더 다루기로 한다.

Ⅲ. 개선된 포트스캔 탐지정보를 공격 탐지정보와 Association을 통한 Correlation 설계

본 부분에서는 침입탐지시스템의 핵심 부분인 공격 탐지 정보와 탐지된 포트스캔 정보와의 Correlation을 위한 연계를 설계하였다.

포트스캔 탐지정보와 공격탐지정보가 Correlation을 위해 포트스캔 탐지 처리기에서 탐지 정보가 저장되고, 공격탐지엔진에서 공격탐지정보를 축출하여 저장된 포트스캔 탐지정보를 참조하는 것은 7단계 속에서 계속적으로 순환되어 침입탐지시스템이 실행되게 된다.

① 네트워크 카드로부터 Promiscuous mode에 의해 패킷을 올린다.
② 시스템에 맞게 Decoding 과정을 거친다.
③ 올라온 패킷에 대해 각각 고유의 Preprocessor 과정을 거친다.
④ 포트스캔이 탐지되면 해당하는 패킷의 소스 IP에 의해 Key가 계산되어 버킷을 검색한다. 버킷에서 다음 슬롯 주소가 NULL이면 새로운 슬롯을 추가하고 다음 주소가 존재하면 슬롯에서 소스 IP와 프로토콜을 비교하면서 연결된 리스트를 이동한다.
⑤ ④ 단계에서 연결된 리스트를 이용하면서 소스 IP와 프로토콜이 같은 슬롯을 찾았다면 내용을 수정하고, 연결리스트를 끝까지 이동했으나 소스 IP와 프로토콜이 존재하는 슬롯을 찾지 못했다면 마지막 슬롯의 다음에 현 패킷의 포트스캔 탐지 정보를 추가한다.
이미 언급되었지만 이때 소스 IP는 다르지만 같은 Key 값이 존재하는 충돌이 발생하면 포트스캔 탐지 정보는 슬롯에 연속적으로 연결된다.
⑥ 탐지 엔진에서 패턴 매칭 후 공격과 관련된 패킷

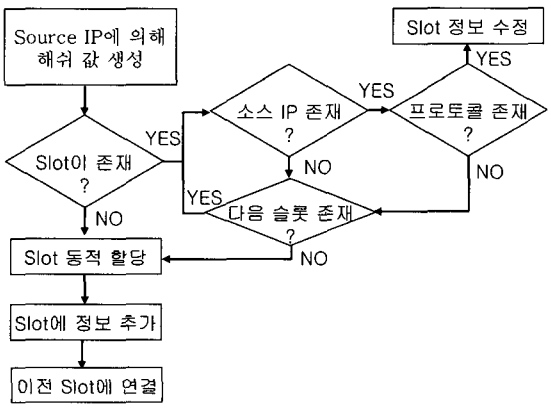

그림 3. 해시 테이블에 탐지된 포트스캔 정보 구성

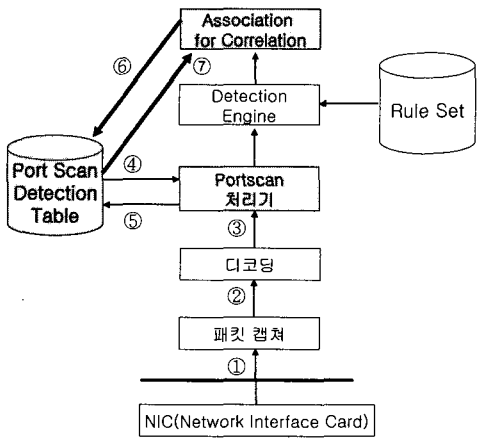

그림 4. 포트스캔 탐지 정보와 공격 탐지 정보간의 Correlation 구조도

이 존재한다면 해시 테이블에서 포트스캔 탐지 정보를 해시 Key(Source IP)와 프로토콜에 의해 검색된다.

⑦ 만약 해시 Key 값과 프로토콜에 관계된 포트스캔 탐지 테이블에 해당하는 것이 존재한다면 공격 탐지 패킷에 대해 가중치를 부가한다.

〔그림 4〕는 순환하게 되는 7단계를 구조적으로 보여준 것이다. 〔그림 4〕에서 순환되는 7단계 중 1~3단계는 일반 적인 침입탐지시스템에 사용된 방식과 똑같으며, 본 논 문 에서 연구된 4 ~ 5 단계인 포트스캔 탐지 테이블의 정보 구성은 패킷에 연결된 슬롯에 포트스캔 탐지 정보 의 삽입, 수정 및 검색은 이미 언급된 내용이며, 6~ 7 단계는 포트스캔 탐지 테이블과 공격 탐지 정보의 연계를 통한 상관관계를 찾기 위한 과정으로 이루어지는 데 다음 절인 3.1에서 언급된다.

3.1 포트스캔 탐지정보와 탐지엔진과의 Association 을 통한 Correlation 과정

위의 2.2.2절에서는 포트스캔이 탐지되어 해시 테이블에 구성되는 과정에 대해서 알아보았다. 이 번 절에서는 탐지된 포트스캔 정보를 이용하여 탐지 엔진에서 탐지된 공격과 Correlation 되는 절차에 대해 논의하기로 한다. Correlation되는 과정을 살펴보면 〔그림 5〕와 같다.

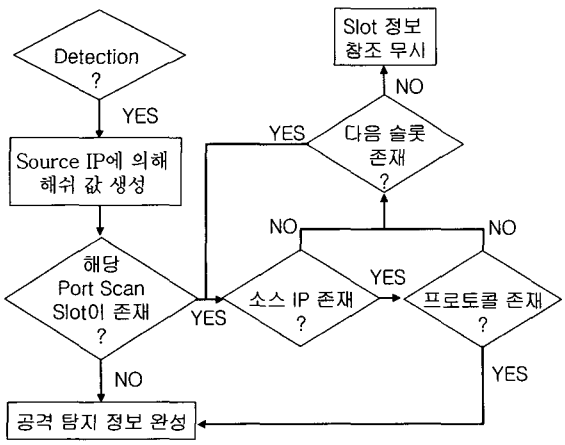

그림 5. 공격 탐지 후 포트스캔 탐지 정보와 Correlation되는 과정

침입탐지시스템은 NIC으로부터 패킷을 Capture 하면 패킷을 분석할 수 있도록 Decoding을 하며, 공격 탐지에 앞서 패킷을 포트스캔 탐지에 의해 먼저 처리하게 되며 그 후 패턴 매칭에 의해 패킷이 공격성 여부를 확인하게 된다. 이런 과정을 거친 후, 공격 패킷으로 판명되면 포트스캔 정보가 있는 해시 테이블에 공격자의 소스 IP를 Key로 해서 검색하게 되는데 Key에 관련된 포트스캔 정보가 있다면 공격 정보에 첨부되어 공격 탐지 정보가 완성된다.

Ⅳ. 실험과 평가

4.1 테스트 환경

테스트를 수행하기 위하여 네트워크상의 패킷을 수집하여 분석할 수 있는 위치 즉, 패킷들이 오고가는 스위치 허브의 미러링 포트를 이용하거나 허브에 들어온 패킷을 모두 복사하여 전달하는 더미 허브를 이용한다. 그 후 연구된 내용들은 네트워크에 시스템의 취약점을 분석하는 툴인 SuperScan[9]을 이용하여 포트스캔 및 공격 테스트 하였다. 테스트 환경으로는 〔그림 6〕과 같다.

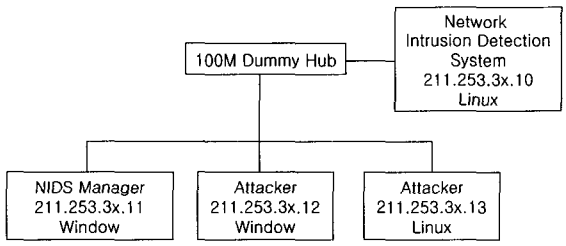

그림 6. 침입탐지시스템을 포함한 테스트 환경

본 논문에서는 개선된 포트스캔 알고리즘이 적용된 침입탐지시스템과 탐지 엔진에서 패턴 매칭 후 포트 스캔 정보와 Correlation으로 연관된 침입탐지시스템을 구현하였다. 개선된 포트스캔 알고리즘이 적용된 모니터링 화면은 〔그림 7〕과 같으며 포트스캔 정보와 Correlation을 구현한 화면은 〔그림 8〕과 같다.

4.2 개선된 포트스캔 탐지정보의 메모리 할당 용량

개선된 포트스캔 탐지의 저장이 메모리 크기에 얼마나 영향을 미치는 지 살펴보자. 버킷의 한 개 크기(4bytes)와 해시 테이블의 전체 버킷의 개수를

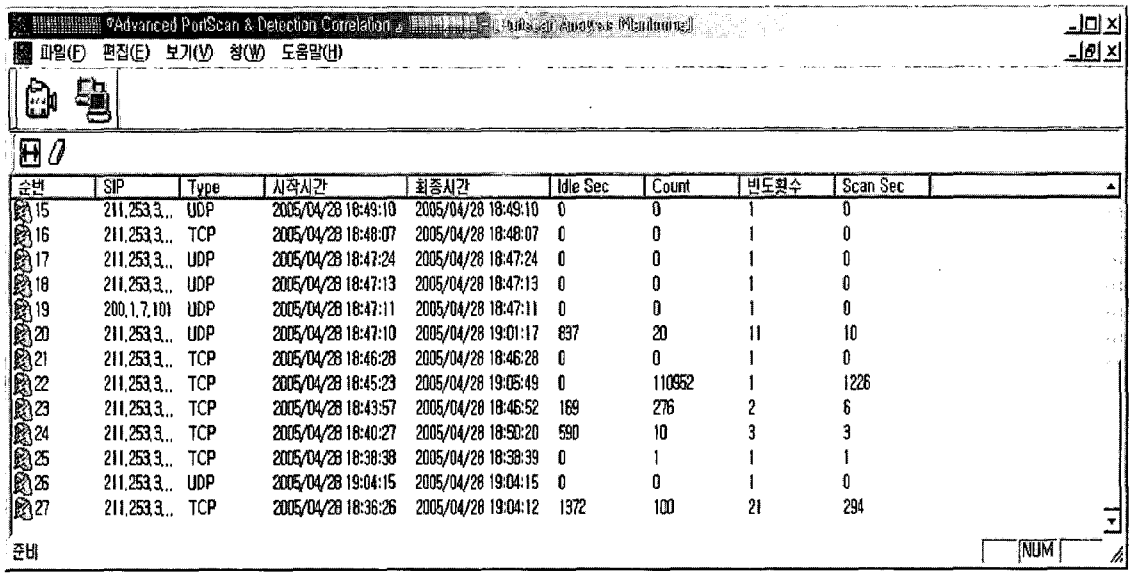

그림 7. Port Scan을 탐지하는 화면

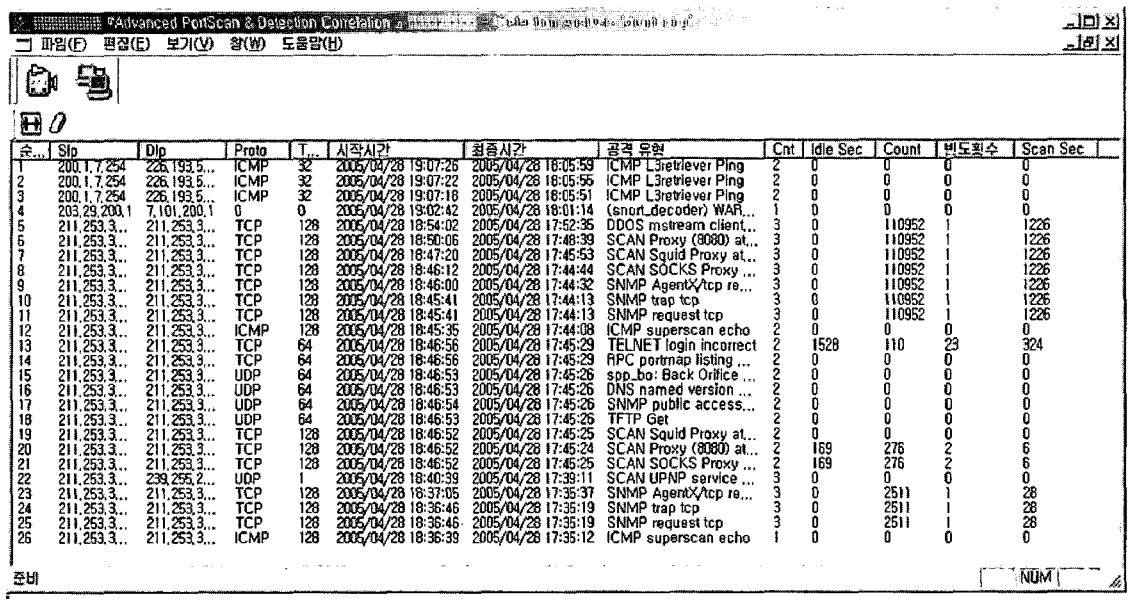

그림 8. 공격정보와 포트스캔 정보의 관계를 보여주는 화면

30,000개로 잡고, 슬롯 한 개의 크기(33bytes)와 다른 종류의 IP 또는 Protocol는 3,000개로 계산했을 때 다음과 같다.

전체 버킷 수 × 버킷 한 개 = 전체 버킷 크기
30,000개 × 4 bytes = 120K bytes

전체 슬롯 수 × 슬롯 한 개 = 전체 슬롯 크기
3,000개 × 33 bytes = 99K bytes

전체 버킷 크기 + 전체 슬롯 크기 = 메모리 차지 용량
120K + 99K bytes = 219K bytes

개선된 포트스캔 탐지는 이와 같이 포트스캔을 수행한 포트 수가 많아지더라도 메모리 용량을 크게 차지하지 않는다. 이를 Snort의 포트스캔 탐지와 비교한다면 더욱 명확하게 비교가 될 것이다. 따라서 개선된 포트스캔 탐지와 Snort 포트스캔 탐지의 메모리 용량을 〔표 2〕로 비교하면 다음과 같다.

표 2. 공격자가 호스트마다 전체 포트를 포트스캔 했을 경우 차지하는 메모리 용량 비교

호스트 수	Snort Port Scan 탐지	개선된 Port Scan 탐지
1 hosts	2M ~ 97M	120,033bytes
100 hosts	200M ~ 9.7G	123.3K
1,000hosts	2G ~ 97G	153K
10,000hosts	20G ~ 970G	450K
100,000hosts	200G ~ 97T	3.42M

* K = 1000 Bytes, M = Mega Bytes, G = Giga Bytes, T = Tera Bytes

〔표 2〕는 호스트가 늘어남에 따라 Snort와 개선된 포트스캔이 차지하는 메모리 용량을 비교해 보았다. Snort에서 포트스캔하려는 호스트 수가 늘어남에 따라 메모리를 차지하는 용량은 급격하게 늘어남을 볼 수 있다. 그래서 Snort는 포트스캔된 탐지 정보를 계속적으로 메모리에 둘 수가 없어 일정 시간마다 주기적으로 정리를 하였다. 그러나 개선된 포트스캔에서는 Snort 방식처럼 세세한 정보 저장을 탈피하여 포트스캔 탐지 정보를 요약(어떤 공격자가 위협적인지를 모니터링하고 또한 이 정보를 이용하여 탐지 엔진과 상관관계를 따져 false positive를 제거하기위한 용도)하여 메모리 할당을 줄이며, 최초 생성 시 해시 테이블에 관계된 Key를 관리하기 위한 메모리 영역만 할당한다. 그런 후, 포트스캔 탐지가 이루어지는 즉시 버킷의 Key를 계산하여 메모리를 동적으로 할당하거나 또는 이미 존재한다면 내용을 수정하기 때문에 아무리 포트 수가 늘어도 아주 작게 메모리를 소비하므로 계속적으로 메모리에 요약 데이터를 두어도 크게 문제가 없을 것이다.

위의 데이터는 단지 메모리를 차지하는 용량을 비교하기 위해 정량적으로 계산한 데이터일 뿐이다. 실제로 네트워크에 흘러 다니는 패킷 중 아주 극소수만 포트스캔을 수행할 것이기 때문에 훨씬 더 메모리를 적게 차지 할 것이다. 〔그림 9〕는 위의 〔표 2〕에서 Snort 포트스캔 탐지의 내용을 평균한 것과 개선된 포트스캔 탐지에 대하여 호스트가 늘어남에 따라 메모리 용량의 변화를 그래프로 나타낸 것이다.

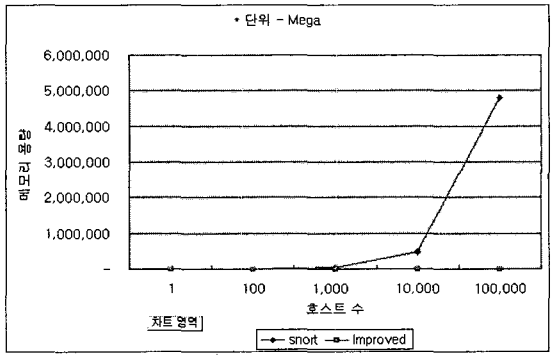

그림 9. Snort 포트스캔 탐지와 개선된 포트스캔 탐지의 메모리 할당량 비교 그래프

4.3 개선된 포트스캔 탐지정보의 검색 성능

개선된 포트스캔 탐지의 경우 슬롯에 데이터를 새로이 입력하거나 기존 정보를 수정할 시 또는 탐지 엔진과 Correlation을 위해 접근할 시 어느 정도의 검색 성능을 보이는 지 살펴보자.

Snort는 소스 IP를 찾기 위해 순차 검색을 하는데 검색 성능은 O(N)이고, 또한 소스 IP를 찾은 다음 목적지 IP를 찾기 위해 순차 검색을 한다. 그 다음으로는 포트를 검색하기 위해 또 한 번의 순차 검색을 거친다. 그래서 총 세 번의 순차 검색을 거치게 되는 데 검색 성능은 각각 O(N)이다. 세 개의 검색 성능의 합은 3 x O(N) = O(N)이다.

그러나 개선된 포트스캔에 대한 탐지 정보를 검색하는 데에는 해시 알고리즘을 사용해서 O(1)이므로 이는 아무리 데이터가 많다 해도 검색하는 데 안정적인 성능을 보인다. 그런데 해시 알고리즘을 사용하는 개선된 포트스캔 탐지는 소스 IP를 Key로 사용하므로 Key의 크기에 따라 차이는 있지만, 충돌(collision) 가능성은 존재한다. 만약 Key 값에 충돌이 발생하면 탐지된 포트스캔 정보를 계속해서 뒤에 연결되도록 하는 연결리스트 방식을 사용한다.

해시 알고리즘에서 O(1)는 Key 값에 대한 충돌이 적다는 전제하에 계산된 검색 성능이므로 아주 많은 충돌이 발생하면 검색 성능은 떨어질 수 있다. 이를 해결하기 위해서는 해시 테이블의 크기를 아주 크

게 늘리면 되는데, 보통 네트워크의 규모에 따라 유입되는 IP의 개수가 차이가 나기 때문에 이 유입된 IP 중에 포트스캔할 수 있는 가능성은 훨씬 더 작으므로 정책적으로 네트워크 규모에 따른 해시 Key 크기를 조절해야 한다. 예를 들면 포트스캔 탐지 정보의 메모리 할당량에서 Key 수를 30,000개로 했을 때, 메모리는 120K가 할당되고, 3,000,000개로 했을 때, 12M 할당된다. 보통 Key 수를 3,000,000개로 잡았을 때 Key에 대한 충돌은 아주 미소(微少)할 것이다. 그러나 대처법으로 네트워크 규모에 따라 정책적으로 적절히 Key 크기를 결정하고, 만약 Key에 대한 충돌이 발생한다 하더라도 순차적인 연결리스트를 사용할 것이 아니라 이진 검색 트리라든지 다른 알고리즘을 사용해 검색 성능을 안정적으로 가져갈 수 있다. 순차적인 연결리스트의 검색 성능은 최악의 경우 O(N)이지만, 이진 검색 트리는 $O(\log_2 N)$이므로 Key 1000번 정도의 충돌이 일어났다 해도 순차 연결리스트는 최악의 경우 1000번 검색을 해야 하지만 이진 검색 트리는 $\log_2 1000 \div 10$번 정도로 숫자가 늘어나도 순차 연결리스트에 비해 아주 안정적인 성능을 보인다[그림 10].

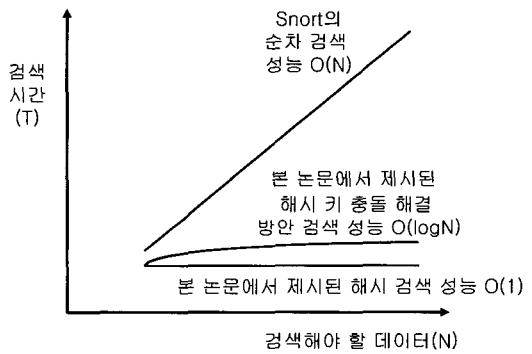

그림 10. 순차 및 해시 검색 알고리즘 성능 비교

4.4 False Positive를 감소시키는 방법으로서의 포트스캔 활용

4.4.1 개선된 포트스캔 탐지의 정보 분석

포트스캔 처리기에서 해시 테이블에 저장한 탐지된 포트스캔 정보를 분석해 보면, 실질적인 정보 필드는 아래 [표 3]과 같다.

[표 3]에 나타나 있는 정보 중 Active Time, Port Scan Count, Frequency Count, Idle Time들이 가장 중요한 정보인데, 이들 간의 관계와 공격에 대한 정보와의 관계를 다음 절에서 논의하기로 한다.

표 3. 미(未)가공된 포트스캔 탐지 정보 레코드

헤더	포트스캔 탐지 정보	설명
Source IP	2049244627	공격자 포트스캔을 시도한 호스트 IP 주소, 해시 Key가 됨
Protocol Type	6	공격자가 포트스캔한 프로토콜 종류 (6 - TCP)
Start Time	1114384181	공격자가 포트스캔을 시작한 시간
Last Time	1114385212	공격자가 포트스캔을 끝낸 마지막 시간
Active Time	512	공격자의 포트스캔 활동 시간의 누적
Port Scan Count	87800	누적된 포트스캔 총 횟수
Frequency Count	100	시작 후, 30초 이상 쉬었을 경우의 횟수
Idle Time	3000	포트스캔 정지 후 30초 이상에서 쉬는 시간
Next List	0x80562350	버킷에서 연결된 다음 슬롯 주소

4.4.2 개선된 포트스캔 탐지 정보를 침입 탐지에 적용

탐지된 공격 정보 레코드의 필드는 공격이 탐지된 후 포트스캔 정보와의 관계는 [표 4]와 같다.

일반적으로 침입탐지시스템에 제공되는 탐지 기능은 단순히 Signature를 이용했기 때문에 매우 많은 False Positive를 발생시켰고, 결국은 실제 해킹 의도와 일반적인 트래픽을 구분하지 못하는 결과를 초래하여 제 기능을 발휘하지 못했다. 그래서 포트스캔 탐지 정보를 다른 공격 탐지 정보처럼 독자적으로 취급하는 것이 아니라 포트스캔 공격이외의 공격 정보와 연계한 Correlation을 발생시켜 좀 더 섬세한 공격 탐지를 연구한 것이다. 결론적으로 [표 4]와 같이 포트스캔 핵심 탐지 정보(Active Time, Port Scan Count, Frequency Count, Idle Time)가 뒷받침된다면 공격에 대한 가중치로 이용하여 공격 탐지에 대한 False Positive를 줄일 수 있는 판단 기준치가 된다.

표 4. 포트스캔 탐지 정보와 결합된 가공되기 전의 탐지된 공격 정보 레코드

헤더	공격 탐지 정보	설명
Source IP	1109720531	공격자 IP 주소이고, 포 트 스캔 탐지 정 보를 검 색하게 되는 Key가 됨
Source Port	37135	공격자가 이용한 포트 번호
Destination IP	2049244627	공격자가 공격하고자하는 목적지 IP 주소
Destination Port	41216	공격자가 공격하는 목적지 포트 번호
Protocol Type	17	공격자가 이용한 패킷의 프로토콜 타입(17 - UDP)
TTL	128	공격이 시도된 패킷의 Time to live
Attack Info	"SNMP public access udp"	탐지된 공격 설명
Attack Count	1	공격한 횟수
Start Time	1114384181	침입탐지시스템에서 공격을 감지한 시간
Last Time	1114385212	공격한 마지막 시간
Active Time	512	공격자의 포트스캔 활동 시간의 누적
Port Scan Count	87800	누적된 포트스캔 총 횟수
Frequency Count	100	시작 후, 30초 이상 쉬었을 경우의 횟수
Idle Time	3000	포트스캔 정지 후 30초 이상에서 쉬는 시간

4가지 포트스캔 탐지 정보의 관계를 살펴보면, Port Scan Count 많다면 Active Time이 길게 되고 Active Time이 길면 Port Scan Count 또한 많게 된다(그림 11). Frequency Count는 수치가 크게 될 때 Idle Time이 길 게 될 수도 있고 그 반대는 성립하지 않는다.

그러나 Port Scan Count 정보는 공격에 대해 False Positive인지 아닌지를 구분할 수 있는 판단 자료로 가장 잘 활용될 수 있다. 또한 나머지 3가지 정보(Active Time, Frequency Count, Idle Time) 역시 누계된 Port Scan Count가 일반적인 포트스캔인지 또는 Slow 포트스캔인지를 구분할 수 있는 정보를 제공한다. 예를 들면 〔표 5〕와 같다.

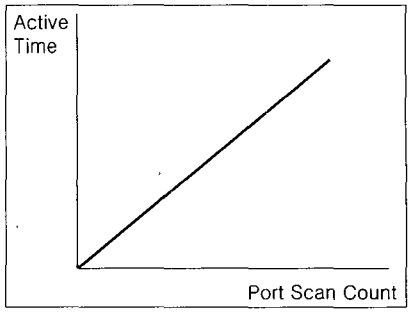

그림 11. Active Time과 Port Scan Count의 관계

위의 예처럼 Port Scan Count가 87800 정도가 침입탐지시스템에 감지가 되었다면 이것을 실시한 IP는 공격을 하기위해 네트워크 및 시스템의 정보를 알아내었다고 생각할 수 있다. 그러나 공격자가 단순히 포트스캔을 한 것인지 아니면 교묘히 포트스캔을 한 것인지는 이 수치만으로 판단하기가 힘들 것이다. 포트스캔은 일반적인 포트스캔과 Slow 포트스캔으로 구분할 수 있는 데 일반적 포트스캔은 침입탐지시스템의 감지 여부에 관계없이 실시한 것이며 Slow 포트스캔은 아주 작은 수의 포트스캔에 대해 30초 이상의 시간 간격을 두며 빈도는 여러 번 보내게 될 것이다. 일반적인 포트스캔의 예는 〔표 5〕이며 Slow 포트스캔에 대한 예는 〔표 6〕과 같다.

표 5. Normal 포트스캔 예

Active Time	Port Scan Count	Frequency Count	Idle Time
500	87800	1	0

표 6. Slow 포트스캔의 예

Active Time	Port Scan Count	Frequency Count	Idle Time
500	87800	10000	1538000

현존하는 침입탐지시스템에서 일반적인 포트스캔은 쉽게 감지될 수 있다. 그러나 정책에 따라 포트스캔이 오지 않을 시 30초 또는 몇 분이 지나면 감지된 포트스캔 정보는 그 데이터양이 엄청나게 되어 삭제된다. 그러나 Slow 포트스캔은 긴 시간대에 아주 작은 양만이 실시되므로 탐지조차 되지 않는다.

그러면 〔표 5〕와 〔표 6〕의 실질적인 예를 가지고 Normal 포트스캔과 Slow 포트스캔에 대해 분석해보자. 〔표 5〕는 Normal 포트스캔의 예인데 Idle Time '0'을 보면 포트스캔을 시작한 후 한 번도 쉬지

않고 계속해서 포트스캔을 Active Time인 512초 동안 진행되었으며, Port Scan Count는 87800 정도로 감지되었다. 당연히 한 번도 쉬지 않았기 때문에 Frequency Count는 1이다. 이것은 일반적으로 어떠한 침입탐지시스템이던지 쉽게 탐지해 낼 수 있으나 지속적으로 데이터를 저장하여 관리하지 않기 때문에 앞으로 네트워크에 얼마나 영향을 미치는 공격자인지에 대한 것은 알 수가 없을 것이다. [표 5]는 Slow 포트스캔의 예인데 Idle Time '1,538, 000'을 보면 약 17일간 포트스캔이 진행되었으나 실질적으로 스캔이 이루어진 시간 즉, Active Time은 500초 정도 밖에 되지 않는다. Active Time과 Idle Time을 비교해보면 그 차이가 아주 많이 나며 Port Scan Count는 87800번 정도로 이루어졌으나 Frequency Count는 10000 정도로 아주 여러 번 진행되었다. 이 Slow 포트스캔에 대해 요약하면 시스템에 탐지 되지 않기 위해 아주 긴 시간 동안 아주 조금씩 여러 번 포트스캔을 실시했다는 것을 알 수 있다.

그런데 일반적으로 Port Scan Count에 대한 수치만 중요시하는 경우가 많다. 그러나 [표 5]와 [표 6]을 비교하면 Active Time과 Port Scan Count는 거의 같다. Frequency Count와 Idle Time은 아주 큰 차이를 보이고 있다. 그러므로 일반적으로 중요시 여기는 Port Scan Count만 중요하게 여기는 것은 침입탐지시스템이 침입 탐지에 대해 비효율을 유발시킨다고 할 수 있겠다.

일반적인 포트스캔과 Slow 포트스캔을 두고 볼 때, Slow 포트스캔이 훨씬 더 위험한 공격자이다. 왜냐하면 공격자가 침입탐지시스템을 우회할 것을 염두에 두고 아주 천천히 조금씩 여러 번 나누어 포트를 스캔 했다는 것은 침입탐지시스템에 대해 아주 잘 이해를 하고 있으며 무엇인가 아주 위험한 계획을 하고 있다고 생각할 수 있다. 일반적으로 개발되었던 침입탐지시스템들은 포트스캔에 대해 어떻게 정보를 관리하고 그것을 이용하는 지에 대해 그렇게 많이 신경을 쓰지 않았고 또한 포트스캔을 실시한 공격자도 마찬가지였으며 아주 깊이 이해하는 공격자도 드물다. 결론적으로 일반적인 포트스캔을 실시한 공격자는 평범한 공격자이며, Slow 포트스캔 공격자를 지능적인 공격자라고 생각할 수 있다.

V. 결론 및 향후 연구 과제

본 논문에서는 첫 번째, 전처리기에 해당하는 포트스캔 공격 탐지의 문제점을 분석하고 해결방안을 제시하였다.

1) 포트스캔은 그 자체의 특성으로 인하여 엄청난 양의 로그를 발생시키며 그 로그 양을 처리하기 위하여 침입탐지시스템은 많은 부하를 감당해야 하는 문제점이 있었다. 이러한 포트스캔 공격탐지 로그를 효율적인 데이터 구조와 알고리즘을 이용해서 저장하여 로그의 양을 줄이고 또한 그 양과 복잡한 데이터 구조로 인해 저장에 소요되는 시스템의 부하를 줄일 수 있었다.
2) 앞에서 언급한 엄청난 양의 로그로 인하여 아주 세밀한 부분과 오랜 시간 포트스캔 공격탐지를 저장하지 못하여 Slow 포트스캔에 대한 탐지에 무방비 상태였으나 첫 번째 문제에 대한 해결과 정책 설정(시간과 공격 횟수에 대한 Threshold)을 없앰으로써 해결되었다.
3) 특정한 형태의 패킷 뿐 만 아니라 정상적인 형태의 패킷에 있어서도 포트에 대한 스캔이 가능한데도 불구하고 일반적인 침입탐지시스템에서는 이를 간과하였다. 정상적인 형태의 패킷에 대해서 포트스캔 공격 탐지가 이루어지려면 세션에 대한 관리가 필요하며 이러한 정보가 포트스캔 탐지에 포함될 수 있음을 제시했다.

두 번째, 포트스캔 탐지 정보를 그 자체로서만 활용하는 것이 아니라 공격 탐지 정보와 연계해 Correlation함으로써 False Positive를 감소시키는 방법으로서 사용될 수 있음을 보였다.

두 번째 항목에서의 아쉬운 점은 포트스캔 탐지에 있어서 문제점의 해결을 제시한 것과 포트스캔 탐지 요약 정보들(Active Time, Port Scan Count, Frequency Count, Idle Time)을 이용하여 공격탐지 정보의 False Positive를 줄일 수 있는 방법을 보였지만 4개의 요약 정보들에 있어서 단일한 가중치를 얻어내는 수학적 모델을 제시하지 못했다. 이는 차후 좀 더 깊이 패턴 인식[10]과 함께 연구할 계획이다.

참 고 문 헌

[1] Jay Beale, James C. Foster, Jeffrey Posluns, Brian Caswell, Snort Intrusion Detection 2.0, SYNGRESS, pp.2-9, Feb, 2003

[2] Stephen Northcutt, Judy Novak, Network intrusion detection an analyst's handbook, 2nd Edition, New Riders, pp.217-224, Aug, 2002
[3] 이현우, "네트워크 공격 기법의 패러다임 변화와 대응방안", SecurityMap Network, Oct, 2001
[4] IA Team, Anual Report 2001.5~2002.5 Macro security Report, Macrotechnology, pp.30-36, May, 2002
[5] W. Richard Stevens, TCP/IP Illustrated, Volume1, ADDISON-WESLEY, pp.69-82, Oct, 2000
[6] Elizabeth D. Zwicky, Simon Cooper, & D. Brent Chapman, BUILDING INTERNET FIREWA- LLS, 2nd Edition , O'REILLY, pp.51-59, Aug, 2000
[7] 백순용, "침입 탐지 시스템에서의 Key와 패턴 우선 순위 변경 알고리즘 설계", 숭실대학교, 석사논문, Jun, 2004
[8] PLUS, Security Plus for UNIX, Yong Jin.com, pp.520-522, Jan, 2001
[9] CGI Vulnerability Scan, http://www.wangproduc ts.co.uk
[10] Richard O. Duda, Peter E. Hart, David G. Stork, Pattern Classification, Second Edition, WILEY INTERSCIENCE, 2001

〈著者紹介〉

박 성 철 (Seong Chul, Park) 정회원
1998년 동국대학교 정보관리학과 학사
1997년 ~ 2000년 (주)유풍 전산실
2000년 ~ 2001년 윈스테크넷 부설 보안연구소 주임연구원
2001년 ~ 2003년 넷시큐어테크놀러지 보안연구소 선임연구원
2005년 고려대학교 전자컴퓨터공학과 석사
2003년 ~ 현재 서울호서전문학교 인터넷정보보안과 교수
〈주관심 분야: 침입 탐지/방지/차단 시스템, 네트워크 및 시스템 보안, 운영체제〉

고 한 석 (Han Seok, Ko)
1982년 Canegie-Mellon Univ. 전기 공학과 학사
1986년 Maryland College Park, 시스템 공학과 석사
1988년 존스 홉킨스 대학교 전기 공학과 석사
1992년 Catholic Univ. of America, 전기공학과 박사
1983년 ~ 1995년 White Oak 연구소, 책임 연구원.
1992년 ~ 1995년 Univ. of Maryland Baltimore Country, 조교수.
1995년 ~ 현재 고려대학교 전자컴퓨터공학과 교수.
〈주관심 분야: 전자 공학, 통신 공학, Computer Security, Navigation and Tracking〉