제 18 회 한국정보처리학회 추계학술발표대회 논문집 제 9 권 제 2 호 (2002.11)

네트워크 기반 침입탐지시스템의 많은 이벤트 중에서 실제 위험 공격탐지를 위한 연구

이은영*, 김병학*,박찬일*,정상갑*,임채호*,이광형*
*한국과학기술원 전산학과
e-mail : eylee@if.kaist.ac.kr

A Study of Finding Real attack from large amount of NIDS events

Eun Young Lee*,Byunghak Kim*,Chanil Park*,Sang-gab Jeong*,Chaeho Lim*
,Kwang Hyung Lee*
*Department of Computer Science, KAIST

요 약

네트워크 기반의 IDS(Intrusion Detection System)가 개발된 이후 네트워크 패킷 정보를 분석하여 침입을 탐지하는 방법들이 연구되고 있다. 그러나 네트워크의 규모가 커지면서 NIDS 에서 발생하는 이벤트의 양이 증가하고 거짓 이벤트의 양도 따라 증가함으로써 이를 분석하는데 어려움이 있다. 본 논문은 많은 이벤트로부터 보다 위험성 있는 공격을 탐지하는 방법을 제시하고, 이를 현재 사용되고 있는 NIDS 인 snort 에 확장시켜 구현 하였다. 본 시스템은 침입자의 의도파악을 위하여 스캔과 같은 기본적인 이벤트를 관리한다. 또한 새로운 취약점에 대한 공격에 우선순위를 두어 오래된 공격방법보다 최근의 공격방법에 더 높은 우선순위를 부여한다. 전체 request 에서 공격이라 판단되는 request 의 비율로써 사용자가 공격의도가 있는지를 파악한다.

1. 서론

NIDS 는 실시간에 침입을 탐지하기 위해서 사용되는 네트워크 감시도구이다. 많은 회사들이 NIDS 를 이용하여 침입을 감시하고 있다. 하지만 네트워크 속도의 증가와 NIDS 가 가진 한계로 인하여 실제로 침입을 감지하지 못하고 그에 대한 적절한 대응이 이루어지지 않고 있다. 본 논문은 기존의 접근과 다르게 false positive (공격이라 잘못 판단한 경고들)를 포함하는 경고들이 발생된 상태에서 필요 없는 경고를 분석하고 가장 필요하고, 중요한 경고들을 찾기 위한 방법으로 접근하였다.

본 논문에서는 다음과 같은 NIDS 의 문제점을 해결할 수 있는 방안을 제시한다.

1. NIDS 는 false positive 정보를 많이 발생해서 실제의 공격이 무엇인지 구분 할 수 없게 한다.
2. 전문적인 해커들은 10 회 이내의 시도로 공격에 성공하기 때문에 적은 정보로 정확한 판단이 이루어 지지 않으면 안 된다.
3. 기존의 NIDS 는 침입감지 능력 향상에 초점을 두고 있다. 감지후의 대응 부분에 대해서는 강조가 이루어 지지 않고 있다.
4. NIDS 의 경고는 몇 단계의 정적인 중요도를 기반으로 침입의 등급을 결정한다. 하지만 각 이벤트는 시간에 따라 그 중요도가 변해야 하며 공격의 target 이 되는 시스템의 상태에 따라 중요도가 변할 수 있다.
5. NIDS 는 가장 기본적으로 발생되는 이벤트들에 대해서는 무시하는 경향이 있다. 예를 들어 스캔 같은 공격은 공격자들이 대상 시스템의 상태를 파악하기 위해 거의 모든 경우에 실행하는 공격이지만 중요시 다루어지지 않고 있다. 따라서 스캔 공격을 이전에 한 사용자는 더 민감하게 관리되어야 한다.
6. 만약 어떤 사용자가 공격의도를 가지고 있다면, 공격자의 모든 연결시도는 공격과 연관되어 있을 것이다. 하지만. NIDS 는 이런 특징을 무시하고 있다.

snort 는 현재 널리 쓰이고 있는 NIDS 로 소스가 공

개된 무료 소프트웨어 이다. 우리는 제시한 방법을 snort 상에 구현하였다.

앞으로의 구성은 다음과 같다. 2 장에서는 관련연구를 설명하고, 3 장에서는 제안된 방법을 설명하며, 4 장에서는 구현과 실행결과를 보여주고, 5 장에서 결론을 맺는다.

2. 관련 연구

NIDS 는 네트워크 상에 센서를 설치하고 그곳을 지나가는 패킷을 수집, 분석하여 침입을 탐지한다. 여기서는 NIDS 에서의 공격 위험도 측정과 snort 상에서의 위험도 측정에 관하여 설명하겠다.

대부분의 NIDS 는 자체적으로 시스템 취약성의 위험도에 따라 공격의 위험도를 측정한다. 각 시스템 환경에 따라 그 기준은 조금씩 다르기도 하지만, 다음과 같이 "상","중","하"로 구분 하여 표시해준다

위협도 "상"

1. 원격 공격자가 시스템 보안을 위배할 수 있는 취약점 (예, 사용자 또는 관리자 권한 획득)
2. 시스템의 완전한 통제 권한을 획득 가능한 로컬 공격
3. 일반 필드에서 많이 발생하는 사고와 관련된 취약성 (CERT/CC 와 연관)

위협도 "중"

: "상" 이나 "하"에 해당하지 않는 취약성

위협도 "하"

1. 시스템의 중요한 정보나 통제의 손실을 야기하지는 않지만, 공격자에게 또 다른 취약성을 발견하거나 다른 공격에 사용할 수 있는 정보를 제공하는 취약성
2. 대부분의 기관에 중요하지 않다고 판단되는 취약성

시스템의 취약성에 대한 데이터베이스로는 NIST (National Institute of Standards and Technology)에서 제공하는 ICAT (a searchable index of information on computer vulnerabilities) 데이터베이스가 있다. 이는 현재 사용되는 대부분의 시스템의 취약성들을 모아 놓은 것으로, 사용자가 자신의 시스템이 가진 취약성을 검색함으로써 자신의 시스템을 평가하고 취약성을 보안하는데 도움을 준다.

snort 에서 위험도 평가를 위해 사용하는 방법으로는 signature 를 위험도별로 그룹화하는 방법과, 각각의 signature 의 option 으로 priority 를 부여 하는 방법이 있다. snort 에서는 위의 분류와 같이 위험도를 세 그룹으로 나누어 high, medium, low 로 구분을 한다.

Table 1: High Priority Classifications - Priority 1

Classtype	Description
attempted-admin	Attempted Administrator Privilege Gain
attempted-user	Attempted User Privilege Gain
shellcode-detect	Executable code was detected
successful-admin	Successful Administrator Privilege Gain
...	...

Table 2: Medium Priority Classifications - Priority 2

Classtype	Description
attempted-dos	Attempted Denial of Service
attempted-recon	Attempted Information Leak
bad-unknown	Potentially Bad Traffic
...	...

Table 3: Low Priority Classifications - Priority 3

Classtype	Description
icmp-event	Generic ICMP event
misc-activity	Misc activity
network-scan	Detection of a Network Scan
...	...

위와 같은 분류와 함께 snort 는 rule option 에 priority 라는 구성성분을 주어 분류된 그룹과 함께 공격의 위험도를 판단하는데 이용한다.

이러한 Priority 의 설정은 취약성 자체에 초점을 두어 위험도를 설정하기 때문에 새로운 취약점에 대한 공격이나 전문가에 의한 공격을 지나칠 수 있다는 문제점을 가지고 있다.

3. 제안된 방법

본 장에서는 서론에서 제시한 NIDS 의 문제점을 보안하여 위험성 높은 공격 호스트를 찾기 위해 세 가지에 우선순위를 두어 이를 이용하는 방법을 제시한다.

3.1 스캔 이벤트 관리

공격자들은 시스템을 공격하기 전에 그 시스템의 정보를 수집한다. 스캔 같은 도구를 이용하여 시스템의 운영체제, 제공하는 서비스, 버전 등의 정보를 알아내어 취약점을 파악한 후 이를 이용하여 시스템에 대한 공격을 시도한다. 즉, 어떤 호스트로부터 스캔 이벤트가 검출되었다는 것은 앞으로 이 호스트가 시스템을 공격할 의도를 지니고 있음을 나타낸다. 따라서 NIDS 에서 탐지한 스캔과 같은 이벤트를 공격 호스트 별로 관리함으로써 공격자의 의도를 파악할 수 있다. 또한 scan 이벤트를 분석하여 false positive 를 줄일 수 있다.

3.2 signature 의 priority 관리

초보 공격자들은 이미 잘 알려진 오래 전의 취약점을 이용하는 반면, 시스템 전문가들은 최근에서야 알려지게 된 이용한 공격을 시도한다. 이미 잘 알려진 취약점들은 대부분 관리자에 의해 패치되어 그 공격의 위험이 적지만 새로이 알려진 취약점들은 그 대처가 늦어져 그만큼 공격위험이 크다. 따라서 시스템의 취약성의 효율적 관리가 필요하다. 새로운 취약점이 발견되었을 시에는 이를 탐지할 수 있는 새로운 signature 가 시스템에 추가되어야 한다. 또한 대부분의 공격들이 새롭게 발견된 취약점에 초점을 두고 행해지므로 최근에 새롭게 추가된 signature 에 더 높은 우선순위를 두어야 한다. 즉, 새롭게 추가된 signature 에 오래된 signature 보다 높은 priority 를 주어 탐지되었을 때, 위험의 정도를 파악할 수 있다. 즉 공격호스트별로 signature 별 통계를 보여줌으로써 분석이 용이하게 해준다.

시간에 따른 signature priority 는 Browne 의 exploit formula 를 적용한다.

$C = I + S\sqrt{M}$ --- Figure 1

(C 는 기록된 사건들의 합계이며, M 은 exploit cycle 이후 지난 시간, I 와 S 는 자료의 분석을 통해 정해진 임계 값이다.)

보다 간단한 실험을 위해 M 은 항상 100 보다 작고,I 는 10, S 는 1 로 가정하여 다음과 같이 사용하고자 한다.

$P = 10 - \sqrt{M}$ --- Figure 2

(여기서 P 는 signature 의 priority 를 말한다.)

3.3 alert request / total request 관리

탐지된 이벤트의 수보다는 실제 request 중에 공격이라 탐지된 이벤트의 비율이 보다 위험한 공격을 파악하는데 도움을 준다. 시스템을 공격하고자 하는 사람은 request 에 attack signature 를 많이 포함하고 있어 정상 사용자 보다 그 비율이 높을 것이다. 또한, 초보 공격자들은 시스템을 공격할 때 여러 번 시도를 하나 전문적인 공격자들은 적은 시도 만에 공격을 성공할 수 있다. 이렇듯 전체 request 에서 탐지된 이벤트들 간의 비율을 관리하여 공격 횟수가 아닌 비율에 우선순위를 주어 보다 위험한 공격자를 구분할 수 있다. 예로 아래와 같이 공격 호스트 A 는 공격 호스트 C 보다 alert 횟수는 많다. 그러나 그 비율에 있어 공격 호스트 C 가 더 위험성을 가지고 있다.

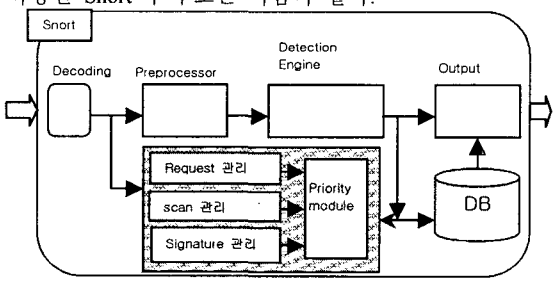
Source IP	Destination IP	Alert	Total	Alert/ Total
A	B	100	1000	10%
C	D	20	100	20%

4. 구현 및 실험

본 장에서는 3 장에서 제안된 방법을 Snort 상에 구현하고 실험을 통해 구현하였다.

4.1 시스템 구조

확장된 Snort 의 구조는 다음과 같다.



[그림 1] 확장된 Snort 구조

위 시스템은 Linux 환경에서 C 를 써서 구현하였으며 snort 의 다른 기능에 영향을 미치지 않도록 하였다.

Request 관리는 공격 호스트 별로 request 의 대한 정보를 가지고 request 가 있을 때마다 update 된다. 전체 request 중 alert request 의 비율은 priority 로 [0-10]값을 가진다.

scan 관리는 호스트마다 탐지된 scan 의 횟수와 내용을 관리한다. Scan 의 횟수에 대한 priority 는 [0-10]의 값을 가진다.

signature 관리는 시간에 따라 signature 의 priority 를 관리하고, 각 공격 호스트 별로 탐지된 signature 의 높은 우선순위를 보여준다. Signature 의 우선순위는 [0-10]의 값을 가진다.

Priority module 은 각 priority 의 값을 가중치에 맞게 적용하여 전체 priority 를 설정한다.

4.2 실험

본 실험은 nmap 을 이용하여 목표 시스템을 공격한 후 이의 결과를 보여준다. 실험을 위하여 시간에 따른 priority of signature 를 다음과 같이 임의로 정하였고, priority module 에서 각 priority 값 들의 비중은 같다고 가정하였다.

Table 4 : 실험에서 가정한 signature 의 priority

signature	Priority
scan null	1
scan nmap tcp	3
scan xmas	5
scan nmap fingerprint attempt	9

목표 호스트: A (143.248.139.193)

공격 호스트 B(143.248.139.181),C(143,248,139.166)

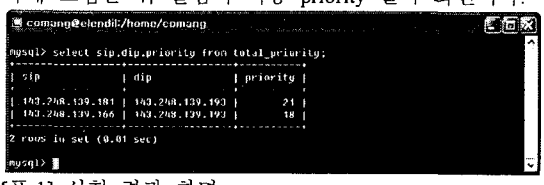
공격 호스트 B 는 A 에 scan null 과 scan xmas 공격을 시도하고 공격 호스트 C 는 A 에 scan nmap tcp 과 scan nmap finger print attemp 를 시도하였다.

Alert request / Total request
143.248.139.181->143.248.139.193 (639/772) : 8
143.248.139.166->143..248.139.193(498/931) : 5

scan priority
143.248.139.181->143.248.139.193 (637/637) : 10
143.248.139.166->143..248.139.193(495/637) : 7

priority of signature
143.248.139.181->143.248.139.193 : 3
143.248.139.166->143..248.139.193 :6

아래 그림은 위 실험의 최종 priority 결과 화면이다.



[표 1] 실험 결과 화면

5. 결론

본 논문에서는 서론에서 제시한 NIDS 의 문제점을 해결할 수 있는 방안을 제안하였다.

본 논문에서 제안하는 방법은 NIDS 의 탐지 능력의 향상보다는 탐지 후의 대응부분에 초점을 맞추고 있다. 먼저 시스템 스캔시도와 같은 기본적인 이벤트를 관리하여 공격자 의도의 파악과 함께 보다 위험한 공

격자를 따로 관리할 수 있으며, 이러한 이벤트를 분석하여 false positive 를 줄임으로써 중요한 이벤트를 찾는데 도움을 준다. 각 signature 의 시간에 따른 중요도를 변화 시킴으로써 최신의 공격기법에 초점을 두어 관리할 수 있다. 또한 모든 연결 시도를 관리하여 실제 request 중 탐지된 이벤트들의 비율에 우선순위를 두어 적은 시도로 시스템의 공격에 성공하는 전문적인 해커들에 대응할 수가 있다.

앞으로 시간에 따른 중요도 변화뿐 아니라, 시스템 자산과 특징을 이용하여 중요도를 변화시킬 수 있는 signature 의 관리에 대한 연구가 필요하다

참고문헌

[1] www.snort.org

[2]서상용 박사, 한국지질자원원구원 "Threatness Based Scan Detector"(TBSD) 네트워크 스캐닝 공격을 탐지하는 도구

[3] http://icat.nist.gov/icat.cfm : ICAT is a searchable index of information on computer vulnerabilities.

[4]http://cve.mitre.org :Common Vulnerabilities and Exposures (CVE) Vulnerability Naming Scheme

[5] http://secritymap.net

[6] http://www.whitehats.com/ ids

[7] http://securityfocus.com

[8] Martin Roesch "Snort-Lightweight Intrusion Detection for Networks" – Stanford Telecommunications, Inc

[9] William A. Arbaugh, Hillary K. Brown, William Fithen, John McHugh , "A trend analysis exploitations" CS-TR-4200 UMIACS-TR-2000-76