네트워크기반 침입탐지 룰 자동생성을 위한 기계학습알고리즘의 비교분석

김현정*, 원일용*, 황숙희*, 이창훈*
건국대학교 컴퓨터공학과
e-mail : (khj486, mamipapa, clcc, chlee)@kkucc.konkuk.ac.kr

Analysis of Machine Learning Algorithm for Automatic Rule Generation on Network base Intrusion Detection

Hyun-Jung Kim*, Il-Young Won*, Sook-Hee Hwang*, Chang-Hoon Lee*
*Dept. of Computer Engineering, Konkuk University

요 약

현재의 침입탐지 시스템은 전문가의 수작업을 통해 공격에 대한 룰을 만들어 왔다. 최근 급속도로 증가하고 있는 새로운 공격 패턴에 대한 즉각적인 대처를 위해 침입탐지 시스템에서의 자동 룰 생성은 이미 중요한 관심사로 부각되고 있다. 본 논문에서는 자동 룰 생성을 위하여 적용될 수 있는 알고리즘의 효율성을 비교하기 위하여, 몇 가지 알고리즘을 대상으로 비교 실험을 하였다. 본 실험 결과는 앞으로 자동 룰 생성을 위한 알고리즘 선택에 지침서 역할을 할 수 있을 것이다.

1. 서론

최근 공격 패턴이 다양화 되고 , 급속도로 변화되고 있다. 이러한 다양화되고 변형되는 공격 패턴에 대한 즉각적인 대처를 위해 침입탐지 시스템에서의 자동 룰 생성은 중요하다.

현재 침입탐지 시스템의 룰 생성은 전문가의 수작업을 통하여 이루어 지고 있고, 이렇게 만들어진 룰은 침입에 대한 탐지를 위해 사용되고 있다. 이러한 방법은 전문가를 반드시 거쳐야 하는 문제점과 만들어진 룰을 유지하고 또 다른 새로운 공격에 대한 추가적인 룰 생성에 있어 많은 비용과 시간이 소요되는 문제점을 가지게 된다.

침입탐지 시스템의 자동 룰 생성을 하기 위하여 사용될 수 있는 알고리즘에는 여러 가지가 있다. 이러한 많은 알고리즘을 침입탐지 시스템에 적용 하려면 알고리즘의 효율성을 고려하지 않을 수 없다.

본 논문에서는 네트워크 기반에서의 기계학습 알고리즘을 중심으로, 룰 자동 생성을 하는 알고리즘의 효율성을 비교하기 위해 몇 가지 대표적인 침입유형을 가지고 직접 실험을 했다. 실험 후 각 알고리즘별 자동 룰 생성의 성능을 분석하고 평가 하였다.

2. 관련 연구

본 장에서는 침입탐지 시스템의 기반 지식을 정리하고, 실험할 네트워크 기반에서의 기계학습 알고리즘을 소개하였다.

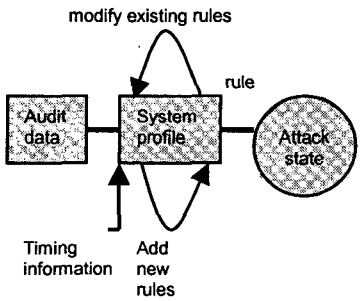
2.1 침입탐지 시스템

인터넷이 확산되고 정보에 대한 의존도가 상승되면서, 정보의 가치가 높아짐에 따라 네트워크를 통한 외부 침입의 가능성이 높아지고 있다. 또, 내부 사용자에 의한 정보 유출 및 파괴활동이 증가하면서 컴퓨터 시스템 및 네트워크를 보호하려는 노력에 관심을 기울이게 되었다. 이러한 내부 또는 외부의 침입으로부터 시스템 및 네트워크를 보호하기 위하여 시스템을 조작 하게 되는데 이것이 침입탐지 시스템이다. 침입을 다루는데 있어서 침입을 즉각적으로 탐지하며, 대처하는 기술이 필요하다. 침입 탐지 시스템은 이러한 신기술을 채용, 시스템이나 네트웍에서 일어난 각종 침입 행위들을 자동으로 탐지, 보고, 대응하는 보안 시스템이다.

침입탐지 시스템의 분류는 데이터의 소스를 기반으로 한 분류와 침입모델을 기반으로 하는 분류가 있다

먼저 데이터의 소스 중심으로 분류를 하면 단일 호스트기반(Host based) 침입탐지 시스템, 다중 호스트기반(Multi-Host based) 침입탐지 시스템, 네트워크기반(Network based) 침입탐지 시스템 등 세가지 형태로 분류할 수 있다. 단일 호스트기반 침입탐지 시스템은 단일 호스트로부터 생성되고 모아진 원시 데이터를 침입탐지에 사용하는 방법이다. 다중 호스트기반 침입탐지 시스템은 여러 호스트들로부터 생성되어 모아진 원시 데이터를 침입탐지에 사용하는 방법이며, 네트워크기반 침입탐지 시스템은 각 호스트들로부터 원시 데이터뿐만 아니라 네트워크 트래픽 데이터를 수집해서 침입탐지에 사용하는 방법이다.

다음으로 침입모델을 중심으로 분류를 하면 비정상(Anomaly Detection) 침입탐지 시스템과 오용탐지(Misuse Detection) 침입탐지 시스템으로 구분할 수 있다[2]. 비정상 침입탐지 시스템은 컴퓨터에서의 자원의 비정상적인 행위나 사용에 근거한 침입탐지 시스템이다. 정상적인 시스템 사용에 관한 파일로 시스템 상태를 유지하는데, 이때 정상 파일을 벗어나는 행위를 탐지하는 시스템이다. 오용탐지 침입탐지 시스템은 시스템이나 응용 소프트웨어에서 알려진 취약점들을 통하여 시스템에 침입할 수 있는 공격 행위들을 사전에 공격에 대한 정보를 가지고 침입에 대해 탐지하는 방법이다.



<그림 1> 오용탐지 시스템의 구조

2.2 기계학습 알고리즘

기계학습 알고리즘에는 무수히 많은 알고리즘들이 발표되고 사용되고 있다. 그 중에서 특히 침입탐지 시스템에서 많이 시도되고 있는 알고리즘인 ID3, CN2, Apriori, Back Propagation 알고리즘을 대상으로 실험을 하도록 하였다. 이에 각각의 알고리즘을 간단하게 정리 해 보고자 한다.

ID3 알고리즘은 Induction of Decision Tree 의 약자로 학습 데이터 셋으로부터 결정트리(Decision tree)를 만들어내는 시스템이다. 즉, 가장 좋은 트리란 테스트 데이터 셋에 있는 모든 객체를 정확하게 분류하고 가능한 단순하게 해야 한다. ID3 알고리즘은 우리가 찾고자 하는 룰이 찾아질 때까지 트리를 형성하는 알고리즘이다.

CN2 알고리즘은 휴리스틱한 검색으로서 엔트로피를 사용하고 있는 예에서 분류 룰의 정리된 리스트를 포함한다. 잡음이 될지도 모르는 도메인 안에서 " if... then…" 룰을 포함하여 설계되는 알고리즘이다. CN2 알고리즘은 룰이 생성될 때 ID3 알고리즘과 달리 결론에 가까운 값의 선에서 결과를 보여준다는 것이다.

Apriori 알고리즘은 연관 규칙 탐사 방법 중의 하나로 지지도를 이용하여 동시에 자주 나타나는 항목(빈발 항목 집합)들을 정제하고 빈발 항목 집합에서 생성된 규칙들은 신뢰도를 이용하여 정제하는 방식의 알고리즘이다. 이 알고리즘은 이해하기 쉽고 구현이 간단하여 많이 사용되고 있다.

Back Propagation 알고리즘은 Least mean square 알고리즘의 비선형 확장이다. 기본원리는 입력층의 각 유니트에 입력패턴을 주면, 이 신호는 각 유니트에서 변환되어 중간층에 전달되고 최후에 출력층에서 신호를 출력하게 된다. 이 출력값과 기대값을 비교하여 차이를 줄여나가는 방향으로 연결 강도를 조절하고, 상위층에서 역전파하여 하위층에서는 이를 근거로 다시 자기층의 연결 강도를 조절해나간다. 다층의 구조를 갖는 복잡한 신경망 학습 알고리즘으로 최급 하강법을 기본으로 한 매우 유용한 패턴인식 해법이다.

2.3 공격

여러가지 네트워크 기반의 공격 유형 중에서 Land, Winnuk, Teardrop, Trinoo, Stacheldraht 공격을 대상으로 실험을 하였다. 각각의 공격유형을 정리했다.

Land 공격은 비정상적인 패킷을 생성하는 방법인 서비스 거절 공격에 속하는 공격중의 하나로, 패킷을 보낼 때 출발지 주소와 도착지 주소를 똑같게 할 경우 받는 컴퓨터가 전혀 예상치 못한 패킷들을 보내는 공격이다.

Winnuk 공격은 NetBIOS 포트(139)로 OOB(Out Of Band)데이터를 보냄으로 이루어진다. 윈도우의 경우 이 곳 139 번 포트로 OOB 데이터가 오면 이를 어떻게 처리해야 할 지를 몰라 생기는 서비스 거절 공격이다.

Teardrop 공격은 IP 패킷 조각들을 전송함으로써 공격이 이루어진다. 윈도우 NT 뿐만 아니라 리눅스 시스템들도 많이 받는 공격이다. 이 공격이 성공할 수 있는 원인은 IP 패킷 조각들을 조합하는 데에서 정확하게 조정하지 않는데 있다. 데이터를 전송할 때 프로토콜 처리부에서 전달된 데이터가 IP 패킷에 모두 수용될 수 없는 경우 데이터는 여러 개의 IP 패킷 조각들로 나뉘어 전달된다. 그리고 나서 모든 패킷 조각들이 수신된 후에 이들 패킷 조각을 하나로 조합한다. 이때 미리 할당 해놓은 버퍼가 오버플로우 되지않도록 하기위해 각 패킷 조각에 대해 길이가 너무 크지 않도록 하여 검사를 한다. 그러나 패킷 조각의 길이가 0 보다 작은지를 검사하지 않는다는 것이 문제가 된다. 그러므로 교묘하게 감춰진 패킷들을 보냄으로써 이 공격을 받을 경우에는 재부팅 혹은 시스템이 정지된다.

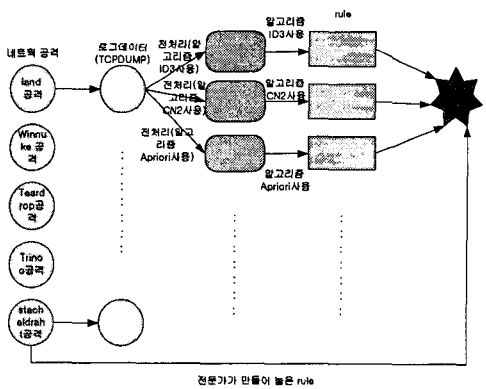
Trinoo 공격은 서비스 거절 공격의 방법으로 원하고자 하는 목적을 달성하기가 어려워지자 네트워크

환경을 활용하는 방식이 나타났다. 이를 분산 서비스 공격이라고 하는데 이러한 공격중의 하나다. 서버에게 공격을 명령하는 마스터에게 연결하는 원격제어 프로그램(클라이언트)에 의해 작동된다. 많은 소스로부터 통합된 UDP flood 서비스 거부 공격을 유발하는데 사용되는 도구이다. 공격자는 Trinoo 마스터에 접속 하거나 그 이상의 IP 주소를 대상으로 하여 서비스 거부 공격을 수행하게 한다. 그러면 하나 혹은 여러 개의 IP 주소를 공격하도록 데몬들과 통신을 한다. 즉, 이 공격은 오로지 한 곳만을 공격하는 가장 단순한 방법이다.

Stacheldraht 공격은 해커와 해커의 공격 명령을 받아 에이전트 시스템에 공격 명령을 전달하는 시스템(마스터 시스템), 실제 공격을 수행하는 시스템(에이전트 시스템)들 사이에 통신이 암호화 되어 해킹에 대한 감시로부터 보호하도록 되어있다. 주로 발견되는 시스템을 보면 솔라리스 2.X 인데, RPC 서비스의 버퍼 오버플로우 때문에 원격에서 쉽게 해당 서버를 장악할 수 있기 때문이다. 솔라리스 시스템들이 해커에 의해 에이전트 시스템이나 마스터 시스템으로 사용되었다. 이 서비스 거절 공격의 내용은 ICMP flood, SYN flood, UDP flood, Smurf 공격을 할 수 있다.

3. 실험

3.1 실험 방법



<그림 2> 실험 방법

각의 네트워크 공격을 TCPDUMP 를 이용하여 스캔한후 이 내용을 각각의 기계학습 알고리즘에 맞게 전처리 과정을 거쳐 가공한다. 각의 알고리즘에 적합하도록 전처리를 통해 만들어진 데이터를 가지고 이를 각의 알고리즘에 적용시켜서 룰을 생성한다. 이때 생성된 룰을 기존에 전문가에 의해 만들어진 룰을 통해 비교 실험을 함으로써 같거나 비슷한 룰이 생성되는지를 볼 수가 있다.

이를 통해 각각의 네트웍 공격에 대해 기계 학습 알고리즘을 적용한 룰들을 기존 전문가에 의해 만들어진 룰과의 대조를 통해 어떤 알고리즘이 룰을 찾아내는 데 훨씬 더 우수한지를 찾아볼 수가 있다.

3.2 실험 환경

실행 환경으로는 유닉스와 리눅스, 윈도우 환경에서 실험을 하였다. 네트워크 공격으로는 앞의 관련 연구에서 설명한 공격들을 사용하였다. 전처리 과정은 각각의 알고리즘에 알맞게 변형시켰다. 그리하여 다음과 같은 결과로 전처리 과정이 된 것을 볼 있다.

<표 1> 데이터를 전처리 한 결과

기존의 Data file		전처리 과정을 거친 후
Source address	203.252.134.61 ~ 80	a
	203.252.134.146 ~148	b
	others	c
Destination address	203.252.134.61 ~ 80	d
	203.252.134.146 ~148	e
	others	f
Src port	http	80
	ftp	23
	smtp	25
	others	others
TCP flag	syn	s
	ack	ack
	fin	f
	reset	r
	push	p
	urgent	urg
	.ack	dack
Window size	1000 ~ 19999.	small.
	20000 ~ 29999	mid
	30000 ~ 39999	max
	범위에 없거나 범위에 포함되지 않을 경우	others
protocol	arp	arp
	who-has	who-has
	reply	reply
	icmp	icmp
	is -at	Is-at
broadcast	255.255.255.255	c
	255.255.0.255	d
	others	e
time	hh:mm:ss.frac 부분에서 시간의 뒷부분의 차이	frac = finish-start ex) 12:21:12.**12345**, 12:22:13.**12445** 12445-12345 = **100**
IP fragement	frag id:size@offset+	h
	frag id:size@offset	i
udp	udp < 60	d
	Udp >= 60	e
ack size	0 ~ 99	a
	100 ~ 199	b
	200 ~ 299	c
	300 이상 혹은 범위에 없는 경우	others
icmp message types	router-advertisement	j
	Router-solicitation	k
	echo replay	m
	Echo request	n
	others	l

3.3 실험 내용

실험한 결과를 알고리즘별로 각 네트워크 공격을 TCPDUMP 를 사용하여 스캔한 것으로 전처리를 하여 만들어진 데이터 파일들을 가지고 룰이 생성된 것을 볼 수 있다. 그러나 여기서 각각의 알고리즘에 대한 실험한 결과를 보여주는데 실험한 전부를 내용에 다 루려면 내용이 너무 많아서 실험으로 하고자 했던 공격들 중 Land 공격에 대한 실험 결과만을 보여주기로 하겠다. 나머지 공격들도 같은 방식으로 하였다.

(1) Land 공격에 대한 TCPDUMP

```
00:39:42.139853 arp who-has 203.252.134.71 tell 203.252.134.146
00:39:42.140027 arp reply 203.252.134.71 is-at 0:4f:4e:5:53:89
00:39:42.140043 203.252.134.71 > 203.252.134.71: ip-proto-0 20
00:39:48.890985 203.252.134.71 > 203.252.134.71: ip-proto-0 20
00:39:54.259065 203.252.134.71 > 203.252.134.71: ip-proto-0 20
```

(2) Land 공격에 대한 ID3 알고리즘의 결과

```
Decision tree:

sourceaddress in b,c: others
sourceaddress = a:
:...destinationaddress = d: land
    destinationaddress in e,f: others
Extracted rules:
Rule 1: sourceaddress = a
        destinationaddress = d
          -> class land
Rule 2: sourceaddress in b, c
          -> class others
Rule 3: destinationaddress in e, f
          -> class others
```

(3) Land 공격에 대한 CN2 의 알고리즘의 결과

```
*UNORDERED-RULE-LIST*
 IF     sourceaddress = a
   AND destinationaddres = d
 THEN   class = land
 IF     sourceaddress = c
 THEN   class = others
 IF     sourceaddress = b
 THEN   class = others
 IF     destinationaddres = f
 THEN   class = others
```

(4) Land 공격에 대한 Apriori 의 알고리즘의 결과

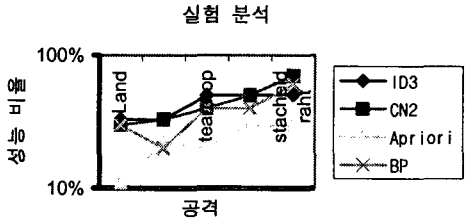
```
d <- land   (39.2%, 100.0%)
d <- b   (19.6%, 100.0%)
d <- c   (15.7%, 87.5%)
a <- land   (39.2%, 100.0%)
a <- e   (25.5%, 92.3%)
others <- e   (25.5%, 100.0%)
others <- b   (19.6%, 100.0%)
others <- c   (15.7%, 100.0%)
land <- d a   (39.2%, 100.0%)
a <- d land   (39.2%, 100.0%)
d <- a land   (39.2%, 100.0%)
others <- d b   (19.6%, 100.0%)
d <- others b   (19.6%, 100.0%)
others <- d c   (13.7%, 100.0%)
d <- others c   (15.7%, 87.5%)
e <- a others   (25.5%, 92.3%)
others <- a e   (23.5%, 100.0%)
a <- others e   (25.5%, 92.3%)
```

3.4 실험 결과

실험후 생성된 룰로 침입탐지에 대한 전체적인 성능을 각 알고리즘별, 공격별 분석 한 결과는 다음과 같다.

<표 2> 공격과 알고리즘에 대한 성능 분석 결과

	ID3	CN2	Apriori	BackPropagation
Land	33	30	11	30
Winnuke	33	33	22	20
Teardrop	50	40	20	40
Trinoo	50	50	30	40
Stacheldraht	50	70	30	60



<그림 3> 네트워크 공격에 따른 실험 분석결과

4. 결론

지금까지의 실험을 통해 얻은 결과로 보면 CN2 알고리즘을 사용하는 것이 자동 룰을 생성하는데 적합하다는 결론이 나왔다. 비교실험을 통하여 본 결과는 기존에 사람들이 경험에 의해 만들어 놓은 룰들과 비교하여 보았을 때 룰을 비교함에 있어서 똑같지는 않아도 비슷한 경우의 룰이 생성되는 것을 볼 수 있었다. CN2 알고리즘이 가장 패턴 자동 생성을 하기위한 출발점으로 사용하는 알고리즘으로 적합하다고 생각한다.

향후과제로 CN2 알고리즘을 통하여 오용탐지 시스템에서가 아닌 비정상적인 시스템에서도 자동 룰 생성을 통해 정상의 데이터와 공격을 당한 데이터를 판단할 수 있다고 본다.

참고문헌

[1] Stephan Northcutt, Judy Novak, DonamcLachlan "Network Intrusion Detection & Analyst's Handbook", second Ed, New Ride September, 2000

[2] R. Agrawal, T.Imielinski, and A. Swap Mining association rules between sets of items large database. 1993

[3] Giovanni vigna, Richard A, Kemmel " NetSTAT: A Network-based Intrusion Detection Approach" , 1999

[4] Joel Scambray, Stuart McClure, George Kurtz , " Hacking Exposed : Network Security Secrets & Solutions, Second Edition ", Osborne, 2001

[5]Ill-Young Weon "A Framework for Constructing Features and Model for Network based anomaly IDS using Baysian Network", 2001

[6] D. Wolpert. Stacked generalization. Neural Networks,1992