세션 패턴을 이용한 네트워크기반의 비정상 탐지 모델

박 수 진† · 최 용 락††

요 약

현재는 인터넷 이용자들이 급격하게 증가하고 있으며, 초보수준의 일반 네트워크 사용자들도 인터넷상의 공개된 해킹 도구들을 사용하여 고도의 기술을 요하는 침입이 가능해졌기 때문에 해킹 문제가 더욱 심각해지고 있다. 해커들이 침입하기 위하여 취약점을 알아내려고 의도하는 다양한 형태의 침입시도들을 탐지하여 침입이 일어나는 것을 사전에 방어할 수 있는 침입시도탐지가 적극적인 예방 차원에서 더욱 필요하다. 기존의 포트 스캔이나 네트워크 취약점 검색 공격에 대응하기 위한 네트워크 기반의 비정상 침입시도 탐지 알고리즘들은 침입시도탐지에 있어 몇 가지 한계점을 갖고 있다. 기존 알고리즘들의 취약성은 Slow Scan과 Coordinated Scan을 할 경우 탐지할 수 없다. 따라서, 침입시도 유형에 제한을 받지 않고 침입시도에 관한 다양한 형태의 비정상 접속을 효과적으로 탐지할 수 있는 새로운 개념의 알고리즘이 요구된다. 본 논문에서는 평상시 정상적인 서비스 패턴을 가지고 그 패턴과 다른 비정상 서비스 패턴이 보이면 이를 침입시도로 탐지하는 개념의 SPAD(Session Pattern Anomaly Detector) 기법을 제안한다.

Anomaly Detection Model based on Network using the Session Patterns

Soo-Jin Park† · Yong-Rak Choi††

ABSTRACT

Recently, since the number of internet users is increasing rapidly and, by using the public hacking tools, general network users can intrude computer systems easily, the hacking problem is getting more serious. In order to prevent the intrusion, it is needed to detect the sign in advance of intrusion in a positive prevention by detecting the various forms of hackers' intrusion trials to know the vulnerability of systems. The existing network-based anomaly detection algorithms that cope with port-scanning and the network vulnerability scans have some weakness in intrusion detection. they can not detect slow scans and coordinated scans. therefore, the new concept of algorithm is needed to detect effectively the various forms of abnormal accesses for intrusion regardless of the intrusion methods. In this paper, SPAD(Session Pattern Anomaly Detector) is presented, which detects the abnormal service patterns by comparing them with the ordinary normal service patterns.

키워드 : 침입시도탐지(Probe Detection), SPAD(Session Pattern Anomaly Detector), 세션 패턴(Session Pattern)

1. 서 론

포트 스캔이나 네트워크 취약점 검색 공격에 대응하기 위한 네트워크 기반의 비정상 침입시도 탐지 모델로는 Phrack Magazine의 "Designing and Attacking Port Scan Detection Tools"에서 발표된 Scanlogd와 Silicon Defence사에서 Snort의 preprocessor 플러그-인으로 만든 SPADE(Statistical Packet Anomaly Detection Engine) 등이 있다[1-3]. 그러나 이러한 형태의 공개된 프로그램들은 침입시도탐지에 있어 몇 가지 한계점을 갖고 있다. Scanlogd는 일정시간 동안 정의된 임계값 이상의 연결 요청이 있을 경우 이를 침입시도로 탐지하기 때문에 일정시간보다 느리게 연결을 요청했을 때는 Slow Scan을 탐지할 수 없다. 또한, 한 호스트가 아니라 여러 호스트에서 Coordinated Scan을 할 경우 탐지할 수 없다.

SPADE 알고리즘는 자주 접근되었던 포트를 대상으로 하는 스캔은 탐지할 수 없는 취약성을 갖고 있다. 즉, 평상시의 호스트별 포트들의 접근 빈도를 저장하여 두고 어떤 접근이 있을 경우 자주 접근하지 않던 곳이면 침입시도로 탐지를 하기 때문에 자주 접근되었던 포트를 대상으로 하는 스캔은 탐지할 수 없다.

현재 국내에 널리 알려진 Scanlogd와 SPADE 알고리즘은 탐지 가능한 침입시도 유형이 극히 제한적이기 때문에, 침입시도 유형에 제한을 받지 않고 침입시도에 관한 다양한 형태의 비정상 접속을 효과적으로 탐지할 수 있는 새로운 개념의 알고리즘이 요구된다.

※ 본 논문은 산업자원부의 출연금 등으로 수행한 지역전략산업 석박사 연구인력 양성사업의 연구결과입니다.
† 정 회 원 : 대전대학교 컴퓨터공학과 강의전담교수
†† 종신회원 : 대전대학교 컴퓨터공학과 교수
논문접수 : 2004년 2월 13일, 심사완료 : 2004년 9월 9일

본 논문에서는 평상시 정상적인 서비스 패턴을 가지고 그 패턴과 다른 비정상 서비스 패턴이 보이면 이를 침입시도로 탐지하는 새로운 개념의 SPAD(Session Pattern Anomaly Detector) 기법을 제안한다.

2. 관련 연구

2.1 Scanlogd

Scanlogd는 Phrack Magazine Volume 8에 있는 "Designing and Attacking Port Scan Detection Tools"에서 설명한 간단하고 신뢰할 수 있는 포트 스캔 탐지 모델이다. 이 프로그램은 Raw TCP 소켓을 사용하여 Stealth를 포함하는 TCP 포트 스캔을 탐지하는 방식으로 포트 스캔 탐지 임계값은 다음과 같다.

포트 스캔으로 판단할 조건으로서, 주어진 시간(DELAY) 동안 같은 소스 주소로부터의 포트 접근 횟수(COUNT)를 설정한다. 두 가지 설정값들은 모두 임의로 설정이 가능하다.

```
#define SCAN_COUNT_THRESHOLD 10
#define SCAN_DELAY_THRESHOLD (CLK_TCK * 5)
```

Scanlogd는 포트 스캔에 대한 로그 데이터를 저장하고, 소스 주소 데이터를 찾기 위하여 해쉬 테이블을 사용한다. 이 방식은 일반적으로 해쉬 테이블 크기가 충분히 크다면 제대로 잘 동작한다. 평균적으로 데이터를 찾는 시간은 이진 탐색(binary search)보다는 빠르다. 얼마나 많은 데이터를 유지하는가에 따라서 Scanlogd 프로그램이 제시간에 새로운 패킷 데이터를 가져올 수 있는지 없는지를 알 수 있게 된다. 이 문제는 해쉬 충돌 횟수를 제한함으로써 해결하였고, 같은 해쉬 테이블이 제약 사항에 도달하였을 때 같은 해쉬 값에 대한 데이터는 오래된 데이터를 버리는 방식을 택하였다[1].

2.2 SPADE(Statistical Packet Anomaly Detection Engine)

SPADE는 Silicon Defence사에서 Snort의 preprocessor 플러그-인으로 만든 것으로서 네트워크를 모니터링하고 비정상 이벤트를 Snort report 메커니즘을 통해 경보를 보낸다[3]. SPADE는 Snort에 의해 수집된 패킷들 중에서 홈 넷(home-net)으로 들어오는 TCP SYN만 찾아서 anomaly score를 가지고 비정상으로 결정된 패킷을 보고한다. anomaly score는 네트워크에서 미리 관찰된 히스토리에 기반해서 결정된다. 과거에 발생이 적었던 패킷은 높은 anomaly score를 갖는다. anomaly score는 확률로부터 직접 계산된다. 즉, 주어진 IP와 Port 조합 x에 대해 이 곳을 목적지로 하는 평상시의 일반 트래픽의 패킷이 일어날 확률을 P(x)라고 한다. 식 (1)과 같이 P(x)에 $-\log_2$를 취하여 그 패킷의 anomaly score A(x)를 구한다. A(x)가 설정된 임계값보다 큰 x로의 패킷들은 침입시도로 탐지한다[4, 5].

$$A(x) = -\log_2(P(x)) \quad (1)$$

3. SPAD

3.1 SPAD의 기본 원리

침입의도에 상관없이 사용자의 모든 행위는 패킷으로 나타나며 패킷이 모여 트래픽을 이룬다. 한 서비스의 세션이란 서버와 클라이언트간의 주고 받는 패킷들을 뜻하며, 패킷의 소스가 클라이언트인지 서버인지에 따라 클라이언트 세션과 서버 세션으로 구분된다. 서비스를 수행하는 클라이언트와 서버는 해당 서비스의 프로토콜을 따르므로, 동일한 서비스의 세션들에는 일정한 규칙 즉, 패턴이 있다. 이 패턴을 따르는 세션은 정상적인 서비스를 받는 것이므로, 정상적인 패턴을 따르지 않을 경우에만 침입으로 간주할 수 있다.

서비스 프로토콜의 패턴으로 침입여부를 판단할 수가 있으므로 각 서비스마다의 프로토콜 패턴을 알아 내야하며 학습을 통해 그 프로토콜을 간접적으로 알아내는 방법을 사용한다. SPAD가 비정상을 탐지하기 위해 사용하는 패킷 데이터는 패킷의 데이터 분석을 통해 여러 패킷 데이터들 중에서 선정된 dIP와 dPort 그리고 세션에서의 패턴이다.

세션 패턴을 사용하여 침입시도를 탐지하는 원리는 정상적인 서비스를 받기 위해 클라이언트와 서버간에 주고 받는 세션을, dIP의 dPort별로 통계화하여 두었다가 이 통계를 벗어나는 클라이언트의 접근이 발생한 경우 이를 침입시도로 간주하는 방법이다.

평상시의 세션의 패턴을 통계화하여 두었다가 이 통계를 벗어나는 세션을 침입시도로 판단하는 SPAD는 세션의 패턴을 위해 두 가지의 특징들을 사용한다. 첫째, 클라이언트와 서버가 주고 받는 세션들의 길이, 즉, 세션의 패킷수의 최소값을 하나의 특징으로서 사용한다. 한 세션의 접속 시작부터 끝까지의 주고 받은 패킷수가 평상의 정상적인 세션들의 최소값보다 작을 경우 비정상적인 세션으로 판단하여 침입시도로 탐지한다. 둘째, 세션들을 이루는 패킷들의 시간순의 공통된 데이터 크기의 열을 또 하나의 특징으로서 사용한다. 이 열은 세션의 처음과 마지막 부분에서 관찰되며 세션의 중간 부분에도 반복적으로 관찰되기도 한다. 본 논문은 이 중에서 처음 부분의 것을 특징으로 삼으며 이와 다른 데이터 크기의 열을 갖는 세션은 비정상적인 세션으로 간주되어 침입시도로 탐지하게 된다.

3.2 SPAD 모델

SPAD는 크게 세션 분류기와 패턴 추출기 그리고 패턴 비교기로 이루어져 있다.

세션 분류기는 트래픽의 패킷들을 읽어서 출발지와 목적지가 같은 세션으로 분류하는 역할을 한다. 세션의 종류마다 패킷이 저장될 버퍼가 마련되어 있어서, 트래픽에서 다음 패킷이 입력되면 해당 버퍼로 저장되고, 패킷들이 모두 모이면 그 버퍼의 모든 패킷이 하나의 세션으로 출력된다. 완성된 세션은 실행 모드에 따라 패턴 추출기나 패턴 비교기의 입력이 된다. 실행 모드는 학습 모드(Learing Mode)와 탐지 모드(Detection Mode)가 있다. 세션 분류기에서 출력된 세션은 학습 모드인 경우 패턴 추출기로 들어가고 탐지 모드인 경우 패턴 비교기로 들어간다.

패턴 추출기는 같은 목적지를 갖는 세션들을 모아서 그 세션들에 공통된 패턴을 알아내어 출력한다. 즉, 패턴은 목적지별로 추출된다. 패턴은 두 가지의 특징(feature)들로 이루어진다. 첫째 특징은, 세션을 이루는 패킷들의 데이터 크기를 시간 순으로 늘어놨을 때 같은 목적지를 갖는 세션들에게서 공통으로 나타나는 처음 부분이다. 둘째 특징은 같은 목적지를 갖는 세션들의 최소 길이 이다. 여기서 세션의 길이는 세션을 이루는 패킷들의 개수이다. 이 두 개의 특징들이 한 짝이 되어 패턴 추출기의 출력이 된다.

패턴 비교기는 미리 만들어 놓은 패턴과 침입 시도 여부의 판단의 대상이 되는 세션을 비교하여 이 세션이 패턴과 다르다면 비정상 세션으로 간주하여 경보를 출력한다. 따라서, 패턴 비교기는 세션과 패턴의 두 개의 입력을 받는다. 입력된 세션은 패턴 추출 때와 비슷하게 첫째 특징과 둘째 특징, 즉, 세션을 이루는 패킷들의 시간 순의 데이터 크기 열과 세션 길이가 추출된다. 이들과 패턴의 두 특징들을 각각 비교하여 세션의 두 특징 중 하나의 특징이라도 패턴과 다르다면 비정상 세션으로 경보를 출력한다.

다음은 SPAD모델을 수학적으로 표현했다.

네트워크의 모든 트래픽은 정상 패킷과 공격 패킷으로 이루어져 있다. 네트워크 상의 모든 패킷은 시간, 소스 IP, 소스 Port, 목적지 IP, 목적지 Port, 플래그, 데이터, 크기 등으로 구성되어 있다. 각 패킷을 x라고 했을 때 이 패킷들이 모여 네트워크 트래픽 T를 이루며, 식 (2), 식 (3)과 같이 표현할 수 있다.

$$x = (Time, sIP, sPort, dIP, dPort, Flag, Data, Size) \quad (2)$$

$$T = \{x_1, x_2, \cdots, x_n;\ Time(x_i) < Time(x_{i+1})\} \quad (3)$$

SPAD는 세션 S를 네트워크 트래픽 중에서 한 쌍의 클라이언트와 서버가 주고 받은 패킷들의 집합으로 표현하고 클라이언트 세션과 서버 세션으로 나누면 식 (4), 식 (5)와 같다.

$$S = (Sc \cup Ss) \quad (4)$$

Sc : Client Session, Ss : Server Session

$$S = \{x_1, x_2, \cdots, x_n;\ x_i \in T,\ Time(x_1(Sc)) < Time(x_1(Ss))$$
$$Src(x_i(Sc)) = Dst(x_i(Ss)),\ Dst(x_i(Sc)) = Src(x_i(Ss))\} \quad (5)$$

세션들의 집합을 세션 클래스라고 하며, 클라이언트 세션 클래스와 서버 세션 클래스로 나눌 수 있으며 식 (6), 식 (7), 식 (8)과 같이 표현된다.

$$C = (Cc \cup Cs) \quad (6)$$

Cc : Client Session Class, Cs : Server Session Class

$$Cc = \{S_{c1}, S_{c2}, \cdots, S_{cn};\ Dst(S_{ci}) = Dst(S_{ci+1})\} \quad (7)$$

$$Cs = \{S_{s1}, S_{s2}, \cdots, S_{sn};\ Src(S_{si}) = Src(S_{si+1})\} \quad (8)$$

SPAD는 평상시의 정상적인 세션 클래스 패턴을 저장해 놓고, 어떤 세션이 입력으로 들어오면 저장된 정상 세션 클래스 패턴과 비교하여 침입시도를 탐지한다. 여기에서, 세션 클래스의 패턴을 PTN으로 표현하여, SPAD는 두가지 특징을 가지고 PTN을 만든다. 식 (9), 식 (10), 식 (11)과 같다.

$$PTN = (F_1, F_2) \quad (9)$$

PTN : Session Class Pattern

$$F_1 = f_1(C) \quad (10)$$

$$F_2 = f_2(C) \quad (11)$$

F_1, F_2 : Pattern Feature

패턴의 특징은 특징 평가 함수 $f_1(C)$, $f_2(C)$로 표현되며, 이 두 함수 집합이 SPAD의 특징 함수 집합 E_{SPAD}가 되며 식 (12)와 같다. E_{SPAD}를 변경하는 것에 의해 확장 및 향상이 가능하다. 패턴 특징 F_1은 세션을 구성하는 패킷의 데이터 크기 열의 공통 패턴을 뜻하며 식 (13)과 같다. 패턴특징 F_2는 세션의 최소 크기 즉, 세션을 구성하는 패킷의 최소 횟수를 뜻하며 식 (14)와 같다.

$$E_{SPAD} = (f_1, f_2) \quad (12)$$

E_{SPAD} : Feature Function Set

$$f_1(C) = \{n_1, n_2, \cdots, n_m;$$
$$Size(x_i(S_j)) = Size(x_i(S_{j+1})) = n_i, x_i \in S_i, S_j \in C\} \quad (13)$$

$$f_2(C) = \min\{n(S_i)\},\ S_i \in C \quad (14)$$

$f_1(C)$, $f_2(C)$: Feature Evaluation Function

평상시 정상적인 세션 클래스들의 패턴과 입력 세션을 비교하여 다르면 비정상 세션으로 탐지하며 식 (15)와 같다.

$$P = \{S \mid PTN(\{S\}) \neq PTN^n\} \quad (15)$$

PTN^n = Normal Session Pattern

4. 구현 결과 분석

SPAD 모델 평가의 신빙성을 위해서 DARPA 프로젝트로 MIT Lincoln 연구소에서 만든 침입탐지평가 데이터 중에 1999년 데이터 집합을 이용하여 시뮬레이션 하였다[8-11]. IDS 평가 데이터에는 5주분량의 패킷들이 수집되어있는데, 이중 1, 2, 3째주는 평상시 데이터이고, 4, 5째주는 공격이 들어있는 데이터이다. SPAD는 세션 패턴 추출로 1, 3째주 데이터를 사용하였고, 4째주 데이터를 테스트 데이터로 사용하여 탐지율과 오경보율을 얻었다. 4, 5째주 데이터에는 <표 1>과 같이 5개의 큰 분류의 침입 공격들이 들어있는데, 이중 침입시도 공격인 Probe만을 탐지하는 율로 평가했다.

〈표 1〉 공격 종류

분 류	종 류
Denial of Service Attacks	Apache2, arppoison, Back, Crashiis, dosnuke, Land, Mail-bomb, SYN Flood, Ping of Death, Process Table, selfping, Smurf, sshprocesstable, Syslogd, tcpreset, Teardrop, U-dpstorm
User to Root Attacks	anypw, casesen, Eject, Ffbconfig, Fdformat, Loadmodule, ntfsdos, Perl, Ps, sechole, Xterm, yaga
Remote to Local Attacks	Dictionary, Ftpwrite, Guest, Httptunnel, Imap, Named, ncftp, netbus, netcat, Phf, ppmacro, Sendmail, sshtrojan, Xlock, Xsnoop
Probes	**insidesniffer, Ipsweep, ls_domain, Mscan, NTinfoscan, Nmap, queso, resetscan, Saint, Satan**
Data	Secret

4.1 세션 분류

세션 분류기는 네트워크상의 패킷들을 얻기 위하여 tcpdump의 패킷 로그 데이터를 사용한다. tcpdump에 의해 수집된 패킷 로그 파일은 오프라인으로 세션 분류기의 입력으로 사용되고, 세션 분류기는 이들을 세션별로 분류하여 세션 데이터 파일을 출력한다. (그림 1)은 세션 데이터 파일의 한 예를 보인다.

```
   <Time>          <sIP>          <sPort>     <dIP>          <dPort>  <Size>
22:22:41.767425 172.16.114.148  sgi-dgl  197.218.177.69  ftp      0 0 0 16 0 30 6 0 28 6 0 0 9 0 28 6 0 0 10 0 28 6 0 0 8
22:22:41.767605 197.218.177.69  ftp      172.16.114.148  sgi-dgl  0 96 0 68 0 48 19 30 53 24 29 30 53 24 29 30 53 24 20 30
22:26:08.635773 196.227.33.189  3312     172.16.112.100  ftp      0 0 0 16 0 30 6 0 28 6 0 0 9 0 28 6 0 0 12 0 28 6 0 0 8
22:26:08.636436 172.16.112.100  ftp      196.227.33.189  3312     0 47 72 31 28 30 53 24 29 30 53 24 29 30 53 24 20 30 77
22:32:45.263859 194.27.251.21   3572     172.16.112.100  ftp      0 0 0 16 0 31 6 0 26 0 6 0 0 9 0 26 6 0 0 18 0 26 6 0 0
22:32:45.266181 172.16.112.100  ftp      194.27.251.21   3572     0 47 72 31 28 30 53 24 29 30 53 24 29 30 53 24 20 30 73
22:33:25.713107 172.16.113.84   8287     197.218.177.69  ftp      0 0 0 16 0 30 6 0 26 6 0 0 9 0 26 6 0 0 10 0 26 6 0 0 8
22:33:25.713450 197.218.177.69  ftp      172.16.113.84   8287     0 96 68 48 19 30 53 24 29 30 53 24 29 30 53 24 20 30 71
22:33:51.706187 195.73.151.50   3635     172.16.112.100  ftp      0 0 0 16 0 31 6 0 26 6 0 0 9 0 26 6 0 0 7 0 26 6 0 0 12 0
22:33:51.706888 172.16.112.100  ftp      195.73.151.50   3635     0 47 72 31 28 30 53 24 29 30 53 24 29 30 53 24 29 30 53
```

(그림 1) 세션 데이터 파일

세션 데이터 파일은 여러 개의 동일한 형식의 행들로 구성된다. 각 행은 클라이언트 세션이거나 서버 세션이며 이 두 세션이 짝을 이룬다. 각 행의 앞부분에는 그 세션을 구성하는 패킷들의 공통적인 정보가 기록되고, 그 뒤에는 각 패킷들의 크기들이 순서대로 뒤따른다. 각 필드의 명칭과 순서는 (그림 1)의 첫 행과 같이 표현된다. <Time>은 해당 세션이 시작된 시각으로서 호스트가 세션의 첫 패킷을 최초로 보내거나 받은 시각으로 기록된다. <sIP>와 <sPort>는 각각 패킷 출발지의 IP 주소와 Port 번호이다. <dIP>와 <dPort>는 각각 패킷 목적지의 IP 주소와 Port 번호이다. <size>는 이 세션의 패킷들의 바이트 단위의 크기를 패킷의 시간 순서로 나열한 것이다.

(그림 1)의 세션 데이터 파일은 (그림 2)와 같이 클라이언트 세션과 서버 세션으로 나누어진다.

```
   <Time>          <sIP>          <sPort>     <dIP>          <dPort>  <Size>
22:22:41.767425 172.16.114.148  sgi-dgl  197.218.177.69  ftp      0 0 0 16 0 30 6 0 28 6 0 0 9 0 28 6 0 0 10 0 28 6 0 0 8 0
22:26:08.635773 196.227.33.189  3312     172.16.112.100  ftp      0 0 0 16 0 30 6 0 28 6 0 0 9 0 28 6 0 0 12 0 28 6 0 0 8 0
22:32:45.263859 194.27.251.21   3572     172.16.112.100  ftp      0 0 0 16 0 31 6 0 26 0 6 0 0 9 0 26 6 0 0 18 0 26 6 0 0 8
22:33:25.713107 172.16.113.84   8287     197.218.177.69  ftp      0 0 0 16 0 30 6 0 26 6 0 0 9 0 26 6 0 0 10 0 26 6 0 0 8 0
22:33:51.706187 195.73.151.50   3635     172.16.112.100  ftp      0 0 0 16 0 31 6 0 26 6 0 0 9 0 26 6 0 0 7 0 26 6 0 0 12 0
```

(a) 클라이언트 세션

```
   <Time>          <sIP>          <sPort>     <dIP>          <dPort>  <Size>
22:22:41.767605 197.218.177.69  ftp      172.16.114.148  sgi-dgl  0 96 0 68 0 48 19 30 53 24 29 30 53 24 29 30 53 24 20 30
22:26:08.636436 172.16.112.100  ftp      196.227.33.189  3312     0 47 72 31 28 30 53 24 29 30 53 24 29 30 53 24 20 30 77
22:32:45.266181 172.16.112.100  ftp      194.27.251.21   3572     0 47 72 31 28 30 53 24 29 30 53 24 29 30 53 24 20 30 73
22:33:25.713450 197.218.177.69  ftp      172.16.113.84   8287     0 96 68 48 19 30 53 24 29 30 53 24 29 30 53 24 20 30 71
22:33:51.706888 172.16.112.100  ftp      195.73.151.50   3635     0 47 72 31 28 30 53 24 29 30 53 24 29 30 53 24 29 30 53
```

(b) 서버 세션

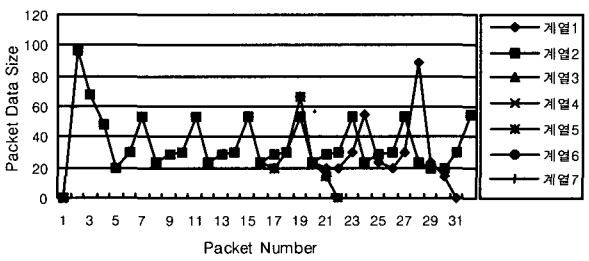
(그림 2) 분리된 세션 데이터 파일

4.2 패턴 추출

패킷 로그 데이터로부터 분류된 세션 데이터는 패턴 추출기의 입력으로 사용되어, 각 호스트들의 서비스들 마다 고유의 세션 패턴이 추출된다. 다음은 서버 세션 데이터로 패턴 추출을 한 것이다.

4.2.1 정상 세션 패턴

정상 세션 패턴이란 정상적인 클라이언트가 평상시에 정상적인 서비스를 서버로부터 받을 때 사용된 세션들의 공통된 패턴을 뜻한다. FTP, Telnet등 여러 서비스에 대한 그래프들이 있지만 그 중 대표적인 FTP에 대한 그래프를 예로 든다. (그림 3)는 FTP서비스의 한 호스트에 대한 정상 세션 패턴을 보여 주는 세션 데이터 그래프들이다. 이 그래프들로부터 패턴 추출기는 각 호스트들의 포트별로 패턴에 사용되는 두 가지의 특징, 즉 공통 경로와 최소 세션 길이를 추출하여 세션 패턴 파일로 출력한다.



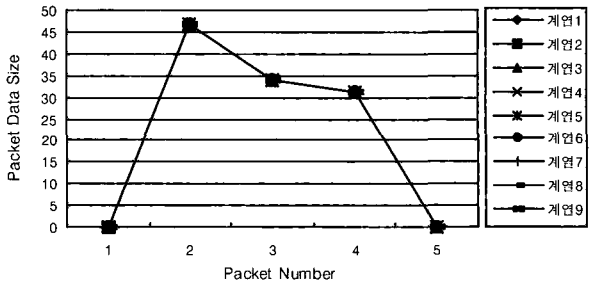
(그림 3) FTP 서비스의 정상 세션 데이터 그래프

4.2.2 침입시도 세션 패턴

공격자가 침입시도에 사용하는 세션은 정상 세션 패턴과 다른 특징을 갖는다. 이 세션들은 서비스의 프로토콜을 위반하는가 또는 준수하는가의 여부에 따라 두 가지로 분류될 수 있다.

첫째, 서비스 프로토콜을 위반하는 침입시도의 세션 패턴이다. 대부분의 침입시도들이 여기에 해당하며, 종류로는 Portsweep과 Ipsweep 그리고 reset 등이 있다. 서비스를 받지 않고 자신의 목적인 호스트나 서비스의 활성 상태 확인을 마치면, 프로토콜을 종료하는 방법으로 정상적인 프로토콜을 따르지 않는다. 따라서, 이 종류들은 패턴의 길이가 짧고, 데이터를 거의 주고 받지 않는 특징을 가지고 있기 때문에 쉽게 탐지할 수 있다.

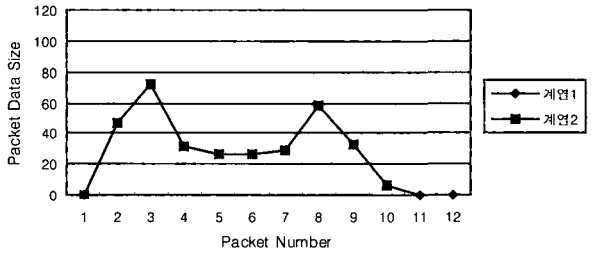
서비스 프로토콜을 위반하는 침입시도인 Portsweep의 FTP 세션 데이터 그래프 예는 (그림 4)와 같다.



(그림 4) FTP 서비스의 비정상 세션 데이터 그래프

둘째, 서비스 프로토콜을 준수하는 침입시도의 세션 패턴이다. 이러한 종류의 침입시도들로는, FTP 서비스에 대한 NTinfoscan과 Telnet 서비스에 대한 SATAN 등이 있다. 이들은 접속해서 서비스를 즉시 끊는 패턴이 아니라, 취약점 검색을 위해 정상적인 서비스 프로토콜을 따르기 때문에 정상 서비스 세션과 단순히 구분되기 어려운 특징을 갖는다.

(그림 5)는 NTinfoscan이 FTP 서비스 포트를 침입시도하는 세션 데이터 그래프를 보인다.



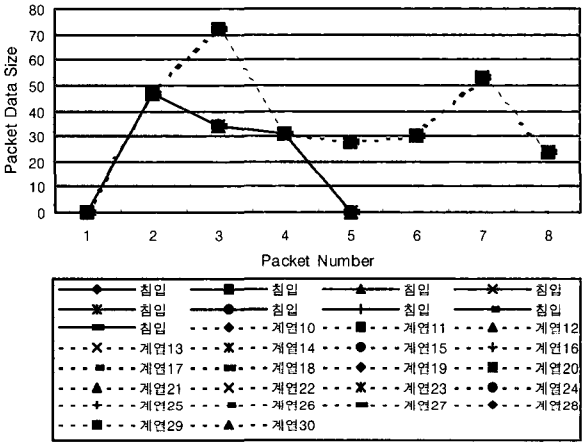
(그림 5) NTinfoscan의 FTP 서비스 침입시도 세션 데이터 그래프

4.3 패턴 비교

패턴 비교기는 미리 만들어 놓은 패턴과 침입 시도 여부의 판단의 대상이 되는 세션을 비교하여 이 세션이 패턴과 다르다면 비정상 세션으로 간주하여 경보를 출력한다. 따라서, 패턴 비교기는 세션과 패턴의 두 개의 입력을 받는다. 입력된 세션은 패턴 추출 때와 비슷하게 첫째 특징과 둘째 특징, 즉, 세션을 이루는 패킷들의 시간 순의 데이터 크기 열과 세션 길이가 추출된다. 이들과 패턴의 두 특징들을 각각 비교하여 세션의 두 특징 중 하나의 특징이라도 패턴과 다르다면 비정상 세션으로 경보를 출력한다.

올바른 판단을 얻기 위해서는 정상적인 패턴과 침입시도 패턴 사이에 뚜렷한 차이가 있어야 한다. 다음은 정상 서비스 세션과 침입시도 세션의 패턴들의 차이를, 침입시도 세션의 서비스 프로토콜 준수 여부에 따라 두 가지로 나누어 설명한다. 첫째, 정상 서비스 세션과 서비스 프로토콜을 위반하는 침입시도 세션의 패턴 비교이다.

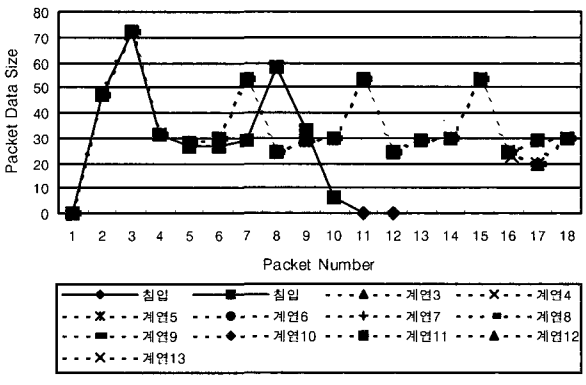
(그림 6)은 FTP 서비스에 대해 침입시도와 정상 세션의 뚜렷한 패턴 차이를 보여주며 이러한 침입시도는 탐지가 가능하다.



(그림 6) FTP 서비스 프로토콜을 위반하는 침입시도와 정상 세션의 패턴 비교

둘째, 정상 서비스 세션과 서비스 프로토콜을 준수하는 침입시도 세션의 패턴 비교이다. 위에서 설명한 첫 번째 경우보다 더 정상 세션에 가까운 패턴을 보이지만, 세션의 길이나 모양이 정상 세션과 차이를 보이므로 탐지가 가능하다.

(그림 7)은 각각 172.16.112.100 IP 주소의 FTP 서비스 포트에 대한 정상 세션과 침입시도 세션의 패턴 비교를 보여준다.



(그림 7) FTP 서비스 프로토콜을 준수하는 침입시도와 정상 세션의 패턴 비교

위에서 보인 분석 결과와 같이 침입시도가 정상 세션의 패턴과 다르면 서비스 프로토콜을 준수하는지 위반하는지의 여부에 상관없이 탐지가 가능함을 알 수 있다.

5. 결 론

본 논문에서는 공격자가 침입을 하기 전에 미리 시도해보는 포트스캔이나 네트워크 취약점 검색 공격을 탐지하기 위하여, 일반적인 인터넷 서비스의 정상적인 서비스 패턴으로부터 비정상적인 침입시도의 세션 패턴을 탐지하는 SPAD-(Session Pattern Anomaly Detector)를 제안하였다.

SPAD는 침입시도 판단의 단서로 사용자가 서비스를 실제로 이용하고 있는지 아닌지 여부에 초점을 맞추고 있다. 평상시 정상적인 서비스 패턴을 추출하여 기억시키고, 그 패턴과 다른 비정상 서비스 패턴이 보이면 이를 침입시도로 탐지한다. 추출하여 기억된 패킷 데이터 특징으로는 dIP, dPort 조합에서 서버와 클라이언트 세션의 주고 받는 데이터 크기의 패턴과, 그 서비스를 정상적으로 받을 때 패킷을 주고 받는 최소 횟수를 적용하였다.

이 SPAD 알고리즘의 성능 분석을 위해 MIT에서 만든 "IDS Evaluation Data Set"을 이용하여 시뮬레이션을 수행하였고, TP율 81.3%와 FP율 9.5%의 성능을 얻었다. 탐지 성공률이 81.3%인 원인은 다양한 정상 패턴을 추출하는데 필요한 데이터가 제한되었기 때문이다. MIT에서 만든 정상 데이터가 2주 분량에 불과하여 그 이외의 정상 데이터 패턴들의 추출이 불가능했기 때문에 탐지 성공률이 낮아졌다고 사료된다.

기존의 탐지 알고리즘인 Scanlogd는 일정시간 동안 정의된 임계값 이상의 연결 요청이 있을 경우 이를 침입시도로 탐지하기 때문에 일정시간보다 느리게 연결을 요청했을 때는 Slow Scan을 탐지할 수 없다. 또한 한 호스트가 아니라 여러 호스트에서 Coordinated Scan을 할 경우 탐지할 수 없다. SPADE 알고리즘는 평상시의 호스트별 포트들의 접근 빈도를 저장하여 두고 어떤 접근이 있을 경우 자주 접근하지 않던 곳이면 침입시도로 탐지를 하기 때문에 자주 접근되었던 포트를 대상으로 하는 스캔은 탐지할 수 없다.

제안한 SPAD는 기존의 탐지 알고리즘인 Scanlogd나 SPADE에서 탐지할 수 없었던 Slow Scan과 Coordinated Scan, 자주 접근되던 곳으로의 스캔을 탐지할 수 있다.

참 고 문 헌

[1] "Designing and Attacking Port Scan Detection Tools," Phrack Magazine, Vol.8, Issue 53, July, 1998.

[2] "실시간 네트워크 불법 Scan 자동탐지 도구(RTSD) 공개", http://www.certcc.or.kr/.

[3] http://www.silicondefense.com/software/spice/index.htm.

[4] Stuart Staniford, James A. Hoagland and Joseph M. Mcalerney, "Practical Automated Detection of Stealthy Portscans," http://www.silicondefense.com/software/spice/index.htm.

[5] James A. Hoagland and Stuart Staniford, "Viewing IDS alerts : Lessons from SnortSnarf," IEEE, 2001.

[6] "The Art of Port Scanning," Phrack Magazine, Vol.7, Issue 51, September, 1997.

[7] "IP Network Scanning & Reconnaissance," http://www.trustmatta.com.

[8] John McHugh, "Testing Intrusion Detection Systems : A Cririque of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory," ACM Transactions on Infromation and System Security, Vol.3, No.4, pp.262-294, November, 2000.

[9] http://www.ll.mit.edu/IST/ideval/index.html.

[10] attack database, http://www.ll.mit.edu/IST/ideval/docs/docs_index.html.

[11] Off-Line Simulation Network, http://www.ll.mit.edu/IST/ideval/docs/docs_index.html.

박 수 진

e-mail : kokiliko@hotmail.com

1997년 한밭대학교 제어계측공학과 학사

1999년 아주대학교 컴퓨터공학과 석사

2003년 대전대학교 컴퓨터공학과 박사

2003년~현재 대전대학교 컴퓨터공학과 강의전담교수

관심분야 : 침입탐지, 컴퓨터통신 보안, 컴퓨터 포렌식스

최 용 락

e-mail : yrchoi@dju.ac.kr

1976년 중앙대학교 전자계산학과 학사

1982년 중앙대학교 전자계산학과 석사

1989년 중앙대학교 전자계산학과 박사

2001년~2003년 대전대학교 공과대학 학장

1986년~현재 대전대학교 컴퓨터공학과 정교수

관심분야 : 컴퓨터통신 보안, 컴퓨터 포렌식스, DRM