제19회 한국정보처리학회 춘계학술발표대회 논문집 제10권 제1호 (2003. 5)

리눅스시스템에서 서비스자원소비율을 이용한 분산서비스거부공격 탐지 기법

고광선*, 강용혁*, 엄영익*
*성균관대 정보통신공학부
e-mail:{rilla91, yhkang1, yieom}@ece.skku.ac.kr

DDoS Attack Detection Scheme based on the System Resource Consumption Rate in Linux Systems

Kwang-sun Ko*, Yong-hyeog Kang*, Young Ik Eom*
*School of Information and Communication Engineering, Sungkyunkwan University

요 약

네트워크에서 발생하는 다양한 침입 중에서 서비스거부공격(DoS Attack: Denial-of-Service Attack)이란 공격자가 침입대상 시스템의 시스템 자원과 네트워크 자원을 악의적인 목적으로 소모시키기 위하여 대량의 패킷을 보냄으로써 정상 사용자로 하여금 시스템이 제공하는 서비스를 이용하지 못하도록 하는 공격을 의미한다. 기존 연구에서는 시스템과 네트워크가 수신한 패킷을 분석한 후 네트워크 세션정보를 생성하여 DoS 공격을 탐지하였다. 그러나 이 기법은 공격자가 분산서비스거부공격(DDoS Attack: Distributed DoS Attack)을 하게 되면 분산된 세션정보가 생성되기 때문에 침입을 실시간으로 탐지하기에는 부적절하다. 본 논문에서는 시스템이 가지고 있는 자원 중에서 DDoS 공격을 받을 때 가장 민감하게 반응하는 시스템 자원을 모니터링함으로써 DDoS 공격을 실시간으로 탐지할 수 있는 모델을 제안한다. 제안 모델은 시스템이 네트워크에서 수신한 패킷을 처리하는 과정에서 소모되는 커널 메모리 소비량을 감사자료로 이용한 네트워크기반 비정상행위탐지(networked-based anomaly detection)모델이다.

1. 서론

침입(intrusion)이란 침입대상 시스템이 정상적인 서비스를 할 수 없도록 고의적으로 방해하는 행위를 의미한다. 이런 침입을 탐지하기 위하여 호스트기반 탐지(host-based detection)기법에서는 단일 시스템에서 발생하는 사건, 사용자 행동, 그리고 시스템 로그자료 등을 감사자료로 사용하고, 네트워크기반 탐지(network-based detection)기법에서는 수신한 패킷수, 패킷 통계정보, 그리고 패킷 정보 등을 감사자료로 사용한다[1,2].

네트워크에서 발생하는 다양한 침입 중에서 DoS 공격이란 공격자가 침입대상 시스템의 시스템 자원과 네트워크 자원을 악의적인 목적으로 소모시키기 위하여 대량의 패킷을 보냄으로써 정상 사용자로 하여금 시스템이 제공하는 서비스를 이용하지 못하도록 하는 공격을 의미하며, DoS 공격을 탐지하기 위해 시스템과 네트워크가 수신한 패킷을 분석하여 생성된 네트워크 세션정보를 감사자료로 이용한 네트워크기반 탐지기법이 이용되었다[3-6]. 그러나 공격자가 DDoS 공격을 하게되면 네트워크 세션정보가 분산되기 때문에 침입을 실시간으로 탐지하기 어렵다.

본 논문에서는 DDoS 공격이 발생하였을 때 실시간으로 침입을 탐지할 수 있도록 시스템 자원을 이용한 네트워크기반 비정상행위 탐지모델을 제안하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구를 소개하고, 3장에서는 DDoS 공격을 Markov chain을 이용하여 설명하고, 제안하는 모델의 알고리즘을 설명한다. 4장에서는 리눅스시스템에서 구현하는 방법을 설명한다. 마지막 5장에서는 결론 및 향후 연구과제에 대해서 기술한다.

2. 관련연구

본 장에서는 네트워크 세션정보를 이용한 DoS 공격 탐지기법과 DDoS 공격 탐지기법에 대해 소개한다.

본 연구는 대학 IT연구센터 육성·지원사업의 연구결과로 수행되었음.

2.1 DoS 공격 탐지기법

DoS 공격은 공격대상에 따라 두 가지로 구분된다. 하나는 대량의 악의적인 패킷들을 보냄으로써 침입대상 네트워크의 대역폭을 정상적으로 이용할 수 없도록 하는 네트워크 대상 침입이고, 다른 하나는 대량의 악의적인 연결요청으로 시스템 자원을 소모시켜 시스템으로 하여금 정상적인 서비스가 불가능하도록 하는 시스템 대상 침입이다[3].

두 가지 침입 모두 네트워크 세션정보에 의한 비정상행위 탐지기법을 이용하여 탐지할 수 있다. 세션정보는 그림 1과 같이 서비스 타입, 출발지 IP 주소, 도착지 IP 주소, 그리고 플래그 등의 패킷정보와 패킷처리시간을 이용하여 그림 2와 같이 생성된다[4].

timestamp	duration	service	src_host	dst_host	src_bytes	dst_bytes	flag	...
1.1	0	http	spoofed_1	victim	0	0	S0	...
1.1	0	http	spoofed_2	victim	0	0	S0	...
1.1	0	http	spoofed_3	victim	0	0	S0	...
1.1	0	http	spoofed_4	victim	0	0	S0	...
1.1	0	http	spoofed_5	victim	0	0	S0	...
1.1	0	http	spoofed_6	victim	0	0	S0	...
1.1	0	http	spoofed_7	victim	0	0	S0	...
...	...	...	...	...	...	...	...	...
10.1	2	ftp	A	B	200	300	SF	...
12.3	1	smtp	B	D	250	300	SF	...
13.4	60	telnet	A	D	200	12100	SF	...
13.7	1	smtp	B	C	200	300	SF	...
15.2	1	http	D	A	200	0	REJ	...
...	...	...	...	...	...	...	...	...

(그림 1) 네트워크 세션정보를 생성하기 위한 패킷정보

세션정보	(service=http,flag=S0,dst_host=victim), (service=http,flag=S0,dst_host=victim) → (service=http,flag=S0,dst_host=victim)[0.93,0.03,2]
의미	2초동안 http 서비스를 요청한 연결이 93%를 차지하고, victim을 대상으로 S0플래그가 세팅된 연결이 3번 연속해서 발생한 경우가 3% 차지함

(그림 2) 패킷정보를 분석하여 생성된 세션정보

시스템 보안관리자는 그림 2의 세션정보를 사전에 설정한 임계치 범위와 비교하여 DoS 공격을 탐지한다.

이 방법은 소수의 공격자에 의한 DoS 공격의 경우 실시간으로 네트워크 세션정보를 생성하여 침입을 탐지할 수 있지만, 다수의 좀비데몬을 이용한 DDoS 공격의 경우 짧은 시간동안 수신한 엄청난 양의 패킷정보를 분석한 후 생성된 세션정보를 이용하여 실시간으로 침입을 탐지하는 방법으로는 부적절하다.

2.2 DDoS 공격 탐지기법

DDoS 공격은 공격자가 인터넷에 연결되어 있는 보안에 취약한 다수의 시스템에 좀비데몬을 설치하고, 각각의 좀비데몬으로 하여금 침입대상 시스템에 DoS 공격하도록 하는 것을 의미한다. 이러한 DDoS 공격을 방어하기 위해서 많은 보안전문가들이 제시하는 가장 확실한 해결책은 인터넷에 연결되어 있는 모든 시스템과 네트워크의 보안능력을 향상시키는 것이다. 보안능력을 향상시키는 방법으로는 운영체제 보안패치 적용, 라우터에 ingress와 egress 필터링 적용, 그리고 각각의 네트워크에 방화벽을 도입하는 방법 등을 제시하고 있다[5].

그러나 인터넷에 연결되어 있는 모든 시스템과 네트워크의 보안능력을 전체적으로 향상시키는 것은 현실적으로 불가능하기 때문에 차선책으로 제시되고 있는 방법은 DDoS 공격을 신속히 탐지하여 방어하는 방법이다[6]. DDoS 공격을 신속히 탐지하기 위하여 Bayesian 공식을 이용한 기법과 임계치를 이용한 비정상행위 탐지기법이 있는데 이 중에서 임계치를 이용한 비정상행위 탐지기법이 주로 이용된다[7].

본 논문에서 제안한 모델도 침입대상 시스템의 자원 소비량을 모니터링하여 DDoS 공격을 실시간으로 탐지하는 비정상행위 탐지모델이다.

3. DDoS 공격 탐지모델 및 기법

본 장에서는 DDoS 공격을 실시간으로 탐지하기 위하여 사용하고자 하는 서비스자원소비율에 대해 설명하고, DDoS 공격이 발생할 경우 시스템 상태변화에 대해 Markov chain을 이용하여 설명한다.

3.1 서비스자원소비율 모니터링

시스템이 가지고 있는 메모리는 소비되는 목적에 따라 시스템메모리와 서비스메모리로 구분할 수 있다. 시스템메모리란 시스템이 가지고 있는 메모리 중에서 시스템이 운영되는데 기본적으로 소비되는 메모리를 말하며, 서비스메모리란 시스템메모리를 제외한 순수 서비스를 목적으로 소비되는 메모리를 말한다. 이 중에서 DDoS 공격을 탐지하기 위해서 순수 서비스만을 목적으로한 서비스메모리를 감사자료로 사용한다.

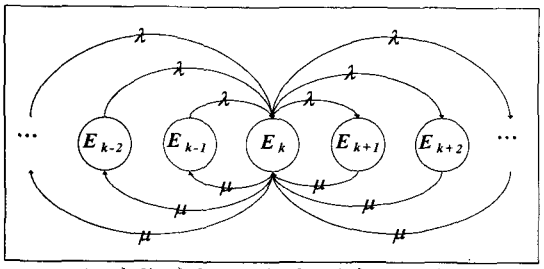
서비스메모리 중에서 시스템이 수신한 네트워크 패킷을 처리하는 과정에서 소모되는 커널 메모리를 이용하고자 하는데, 이 커널 메모리는 처리되는 패킷의 수에 비례하여 즉각적으로 소비되어 DDoS 공격에 실시간으로 반응하는 특징을 가지고 있기 때문이다.

본 논문에서는 커널 메모리를 서비스자원으로, 커널 메모리가 소비되는 량을 서비스자원 소비량으로 정의하며, 단위시간동안 서비스자원 소비량의 변화하는 정도를 서비스자원소비율(SRCR: Service Resource Consumption Rate)로 정의한다.

3.2 DDoS 공격 분석

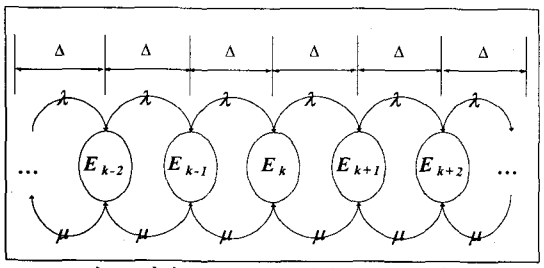
본 논문에서 상태란 정상서비스 중인 시스템이 소비하는 서비스자원 소비량으로 정의하고, 두 상태간 전이를 상태전이쌍 T로 표현한다. 예를 들어 시스템이 임의의 상태 E_k에서 E_{k+1} 혹은 E_{k+2} 상태로 전이하였을 때 상태전이쌍은 $T_{+1}(E_k, E_{k+1})$, $T_{+2}(E_k, E_{k+2})$로 정의하고, 시스템의 서비스자원 소비량이 각각 $|E_{k+1}-E_k|$, $|E_{k+2}-E_k|$만큼 증가하였다는 것을 의미한다. 또한 E_k 상태에서 E_{k-1} 혹은 E_{k-2} 상태로 전이하였을 때 상태전이쌍 $T_{-1}(E_k, E_{k-1})$, $T_{-2}(E_k, E_{k-2})$으로 정의하며, 시스템의 서비스자원 소비량이 각각 $|E_{k-1}-E_k|$, $|E_{k-2}-E_k|$만큼 감소하였다는 것을 의미한다. 또한, 서비스자원 소비량을 증가시키는 비율을 시스템의 서비스요청율(λ)이라 정의하고, 서비스자원 소비량을 감소시키는 비율을 시스템의 서비스처리율(μ)이라 정의한다.

위에서 언급한 정의를 이용하여 임의의 상태 E_k에 있는 시스템 상태전이를 보이면 그림 3과 같다.



(그림 3) 서비스 중인 시스템의 상태 전이

그림 3에서 임의의 상태 E_k에 있는 시스템이 $T_{+i}(E_k, E_{k+i})(|i| \leq 1$, 단 i는 정수)의 상태전이만 가능하도록 time interval(Δ)을 설정하면, Δ동안 시스템이 가질 수 있는 상태전이는 그림 4와 같다.

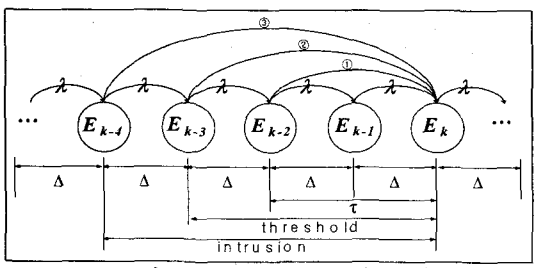


(그림 4) 서비스 중인 시스템의 Δ동안 상태전이

즉, 정상서비스 중인 시스템은 $T_{+i}(E_k, E_{k+i})$(단 i는 정수)의 상태전이가 가능하지만, Δ동안은 $T_{+i}(E_k, E_{k+i})(|i| \leq 1$, 단 i는 정수) 상태전이만 가능하다. 본 논문에서는 서비스 중인 시스템이 Δ동안 바로 인접한 상태로만 전이한다는 내용을 이용하여 DDoS 공격을 설명한다[8,9].

그림 5에서 Δ는 시스템자원 소비량을 모니터링하는 시간단위로, sliding window(τ)는 $\tau = 2\Delta$을 만족하며 서비스자원 소비량의 변화율을 확인하기 위한 시간단위로 정의하면 그림 5의 ①,②,③을 다음과 같이 설명할 수 있다.

①: τ동안 정상서비스 중인 시스템의 상태전이
②: τ동안 정상서비스 중인 시스템이 허용하는 상태전이
③: τ동안 분산서비스공격을 받았다고 판단되는 상태전이



(그림 5) τ, Δ동안 시스템 상태전이

시스템이 DDoS 공격을 받게 되면 그림 5의 ③과 같은 상태전이를 나타내게 되는데, 짧은 시간동안 다량의 패킷을 수신한 시스템은 τ동안 정상서비스 중이라고 판단할 수 있는 상태전이 임계치 범위를 벗어난 상태전이쌍 $T_{+4}(E_{k-4}, E_k)$을 보인다. 즉, 정상서비스 중인 시스템은 그림 5의 ①과 같이 τ동안 $T_{-i}(E_k, E_{k+i})(|i| \leq 2$, 단 i는 정수)의 일반적인 상태전이를 보여주고, ②와 같이 τ동안 $T_{-i}(E_k, E_{k+i})(|i| \leq 3$, 단 i는 정수)의 임계치 범위내 상태전이를 보인다.

이처럼 DDoS 공격을 받게 된 시스템은 정상서비스 중인 시스템이 허용할 수 있는 임계치 범위를 벗어난 상태전이를 보여준다.

3.3 DDoS 공격 탐지 알고리즘

정상서비스 중인 시스템의 서비스자원 소비량을 모니터링하여 서비스자원소비율의 임계치를 구하는 알고리즘은 그림 5,6,7과 같다.

```
/* trSRCR[trCNT]: 모니터링된 서비스자원소비량 */
/* trCNT: 모니터링 회수                        */
while (임계치를 구하기 위한 시스템 시험운영) {
    when (Δ시간마다) {
        trSRCR[trCNT] = 모니터링된 서비스자원 소비량;
        trCNT++;
    }
}
```

(그림 5) 시험운영을 통한 서비스자원 소비량 모니터링

정상서비스 중인 시스템에서 Δ마다 모니터링된 서비스자원 소비량과 모니터링 회수를 trSRCR[trCNT]와 trCNT에 저장한다.

```
/* level : Δ마다 전이할 수 있는 총 단계 */
/* cnt[level]: 단계별 모니터링 회수      */
while (최적의 Δ, level 값이 구해질때까지) {
    for (int i = 0; i < trCNT; i++) {
        templevel = (int)trSRCR[trCNT]/level;
        sum_SRCR[templevel] = trSRCR[trCNT];
        cnt[templevel]++;
    }
    for (int i = 0; i < level; i++) {
        mean_trSRCR[level] = sum_SRCR[level]/cnt[level];
    }
    필요시 Δ, level 조정;
}
```

(그림 6) level별 평균 서비스자원 소비량 계산

trSRCR[trCNT]에 저장된 서비스자원 소비량은 level 값에 따라 그룹화하여 단계별 평균 서비스자원 소비량을 구한 후 mean_trSRCR[level]에 저장하고, 필요시 Δ와 level 값을 조정한다.

```
/* window_size: τ/Δ */
for (int i = 0; i < level; i++) {
    for (int i = 0; i < window_size; i++)
        threshold[level][window_size]
            = 각 level 마다 τ동안의 서비스자원 변화율;
            /* τ동안의 서비스자원 변화율은          */
            /* mean_trSRCR[level]와 Δ값으로 구함 */
}
필요시 window_size 값 조정;
```

(그림 7) 서비스자원 소비량의 임계치를 구하는 알고리즘

각 단계별 τ동안 Δ마다 변화할 수 있는 변화율을 threshold[level][window_size]에 저장하고 필요시 window_size값을 조정한다.

서비스자원소비율의 임계치를 이용한 DDoS 공격 탐지 알고리즘은 그림 8과 같다.

```
/* curSRCR[curCNT]: 모니터링된 서비스자원 소비량 */
/* curCNT: 모니터링 회수                          */
/* templevel: 모니터링된 서비스자원 소비량에 가장  */
/*            인접한 level */
while (DDoS 공격 탐지) {
    when (Δ시간마다) {
        curSRCR[curCNT] = 모니터링된 서비스자원 소비량;
        curCNT++;
        mon_SRCR[window_size]
            = 모니터링시점을 기준으로 최근 τ동안 서비스자
              원 변화율 저장;
        templevel = 모니터링된 서비스자원소비량에 가장 인접
                    한 level 저장;
        for (int i = 0; i < window_size; i++)
            threshold[templevel][window_size]과
                mon_SRCR[window_size]간 비교;
        if (비교결과 모든 변화율이 임계치 범위를 초과하면)
            최근 τ동안 모든 패킷 분석;
    }
}
```

(그림 8) DDoS 공격을 탐지하는 알고리즘

감시대상 시스템의 서비스자원 소비량을 Δ마다 모니터링하여 curSRCR[curCNT]에 저장한 후 최근 τ동안의 서비스자원 변화율을 mon_SRCR[window_size]에 저장한다. 모니터링 시점의 서비스자원 소비량에 가장 인접한 level을 구하여 templevel에 저장하고, threshold[templevel][window_size]와 mon_SRCR[window_size]간 서비스자원 변화율을 비교한다. 비교결과 모니터링된 서비스자원 변화율이 임계치 범위보다 모두 클 경우 모니터링 시점을 기준으로 최근 τ동안 DDoS 공격을 받았다고 판단하고 패킷을 분석한다.

DDoS 공격으로 판단된 τ동안의 패킷에 대한 분석 알고리즘은 본 논문에서 제외한다.

4. 구현방법

리눅스시스템에는 시스템상태정보를 제공하는 '/proc' 디렉토리 안에는 커널의 메모리 사용정보를 확인할 수 있는 slab 파일이 존재한다. slab 파일 내 여러 항목 중에는 수신한 패킷에 의해 세션이 성립되었을 경우 성립된 세션현황을 보여주는 ip_conntrack 항목이 있는데, ip_conntrack 항목은 1개당 약 350 바이트의 스왑(swap)되지 않는 커널 메모리가 사용되고 있다는 것을 의미한다.

즉, ip_conntrack 항목은 수신한 패킷 수에 비례하여 커널 메모리 소비량을 간접적으로 나타내므로 리눅스시스템이 제공할 수 있는 ip_conntrack 항목의 최대값과 정상서비스 중인 리눅스시스템이 보여주는 ip_conntrack 항목의 Δ마다 소비량 및 τ동안 변화율을 이용하여 비정상행위탐지기법에 적용하면 실시간으로 DDoS 공격을 탐지할 수 있다.

5. 결론 및 향후연구

DDoS 공격을 방어하기 위해서 많은 보안전문가들이 제시하는 가장 현실적인 해결책은 신속히 침입을 탐지하는 것이다. 기존 연구에서는 DoS 공격을 탐지하기 위하여 네트워크 세션정보를 이용한 탐지기법이 사용되었는데, DDoS 공격과 같이 대량의 패킷이 발생하였을 경우에는 분산된 세션정보가 생성되기 때문에 실시간으로 DDoS 공격을 탐지하는 기법으로는 부적절하다.

따라서 본 논문에서는 리눅스시스템에서 분산서비스거부 공격을 실시간으로 탐지하기 위하여 네트워크에서 수신한 패킷을 처리하는 과정에서 소모되는 커널 메모리 소비량을 간접적으로 확인할 수 있는 ip_conntrack 항목을 이용하여 비정상행위 탐지모델을 제안하였다.

향후 연구 내용으로는 본 논문에서 제시한 모델을 시뮬레이션을 통하여 검증하도록 하고, DDoS가 발생하였을 때 실시간으로 반응하는 서비스자원을 추가로 제시하도록 하겠다.

참고문헌

[1] Joseph S. Sherif, and Tommy G. Dearmond, "Intrusion Detection: Systems and Models," Proceedings of Eleventh IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2002.

[2] Stefan Axelsson, "Research in Intrusion-Detection Systems: A Survey," http://www.ce.chalmers.se/staff/sax/survey.ps.

[3] L.Stein, The world wide web security faq, version 2.0.1, http://www.w3.org/Security/Faq/ -visited 04.10.2000.

[4] Wenke Lee, Salvatore J. Stolfo, Kui W. Mok, "A Data Mining Framework for Building Intrusion Detection Models," Proceedings of the 1999 IEEE Symposium on Security and Privacy, 1999.

[5] Frank Kargl , Joern Maier , Michael Weber, "Protecting Web Servers from Distributed Denial of Service Attacks," Proceedings of the tenth international conference on World Wide Web April 2001.

[6] Rocky K. C. Chang, "Defending against Flooding-Based Distributed Denial-of-Service Attacks: A Tutorial," IEEE Communications Magazine, Volume:40 Issue: 10, Oct 2002.

[7] Ed Skoudis, Counter Hack: A Step-by-Step Guide to Computer Attacks and Effective Defenses, Prentice Hall PTR, 2003.

[8] Leonard Kleinrock, QUEUEING SYSTEMS Volume 2: Computer applications, Wiley Interscience, 1976.

[9] S. Jha, K. Tan, and R. Maxion, "Markov chains, Classifiers, and Intrusion Detection," Computer Security Foundations Workshop (CSFW), June 2001.