랜덤성을 이용한 알려지지 않은 신종 웜에 대한 탐지 기법

박현도 ○ 이희조
고려대학교 컴퓨터학과 컴퓨터보안 연구실
{hyundo95○, heejo}@korea.ac.kr

Anomaly Detection System of Worm Using Randomness Check

Hyundo Park○ Heejo Lee
Dept. of Computer Science and Engineering, KOREA University

요 약

인터넷에서 일어나는 침해사고 중에서 웜에 의한 피해가 가장 심각하다. 2001 년 Code Red 웜의 출현과 2003 년 SQL Slammer 웜의 출현 이후로 웜에 감염된 이후에 행동 양상을 탐지하여 대응하는 것은 웜의 피해를 최소화 하기에는 역부족이다. 웜에 의해서 감염이 되기 이전에 웜을 탐지하여 조기에 대처하는 것이 무엇보다 중요하다. 또한 이미 알려져 있는 웜에 대한 행동양상을 이용한 웜의 탐지는 신종 웜의 출현 주기가 급격히 짧아지는 현실에 능동적으로 대처할 수 없다. 현재까지 발생한 인터넷 웜은 감염시킬 대상을 선택함에 있어서 랜덤 생성기를 사용하였으며, 향후 나타날 웜도 빠른 확산과 자신의 위치를 드러내지 않기 위해 랜덤 스케닝 방식을 사용할 것이다. 본 연구는 네트워크의 연결들을 행렬로 표현하고, 이 행렬의 랭크(rank)값을 구하여 랜덤성 체크를 하는 방식으로, 웜으로 인한 트래픽에서 발생하는 랜덤성을 탐지할 수 있도록 하였다. 이 방법은 네트워크에서 알려지지 않은 신종 웜을 탐지하도록 하므로, 웜에의한 확산을 조기 탐지할 수 있게 하고, 더불어 웜의 피해를 최소화 하는 것을 목적으로 한다.

1. 서 론

1988 년 Morris 웜이 처음으로 출현한 이래로 인터넷에서의 웜에 의한 피해사고는 끊임없이 증가하고 있다. Code Red 와 Nimda 웜은 인터넷 상의 수십만 대의 컴퓨터를 감염시켰으며, 공공분야에서부터 개인의 시스템까지 수백만 달러의 피해를 가져다 주었다[1, 2, 3, 4]. 이러한 인터넷 웜은 자기 자신을 복제하여 컴퓨터 시스템을 감염시키며, 자기 스스로 전파하여, 다른 컴퓨터 시스템을 감염시키는 독립된 프로그램이다. 웜은 인터넷 상에 있는 수많은 컴퓨터 시스템들을 감염시키기 위하여, 네트워크 조사, 취약점 검사, 자가 복제 등을 자동으로 수행한다. 전염 속도로는 현재까지 보고된 웜 중 가장 빠른 웜으로 기록되어있는 SQL_Overflow 웜은 10 여분 만에 전 세계의 취약한 서버의 90%를 감염시켰으며, 감염된 서버는 매 8.5 초마다 2 배의 수치로 증가하였다. 이 감염 속도는 이전의 Code Red 웜에 감염된 서버가 매 37 분마다 2 배로 늘어났던 것에 비해 엄청난 속도로 감염되었다는 것을 쉽게 알 수 있다[5]. 이렇게 빠른 전염 속도로 파급되는 인터넷 웜에 대응하기 위해서는 웜이 네트워크의 컴퓨터 시스템들을 감염시키는 초기에 탐지하여 예방하는 것이 제일 중요하다. 또한 신종 웜의 출현 주기가 지속적으로 짧아지고 종류도 다양해짐에 따라서 기존의 웜의 행동 양상을 이용한 웜의 탐지는 네트워크를 보호하는 대에 있어서 더 이상 충분하지 않다.

본 논문에서, 우리는 네트워크의 다른 컴퓨터 시스템을 감염시키는 웜을 탐지하는 기법으로 ADUR(Anomaly Detection system of worm Using Randomness check) 모델을 제안한다. 이 모델은 웜이 생성하는 IP 주소의 랜덤성을 체크 함으로서 네트워크에서 웜의 공격을 판별해 내는 기법이다.

1.1 관련연구

웜을 탐지하는 기법들 중에는 크게 두 가지로 구분할 수 있다. 트래픽의 임계치를 이용하는 방법과 웜의 행동 양상을 이용하여 탐지하는 방법들이 있다. 트래픽의 임계치에 기반을 둔 방법들은 정상적인 트래픽과 비 정상적인 트래픽의 구분이 모호하기 때문에 잘못된 탐지를 하는 비율이 높다. 트래픽 임계치에 기반을 둔 방법들은 간단하지만 비 정상적인 트래픽을 탐지하는 대에 있어서 에러률이 높다. 트래픽 임계치에 기반을 둔 연구로는 Stealth-Watch[6], Mazu Enforcer[7]등의 시스템들이 있다. 이에 반해 웜의 행동 양상에 기반을 둔 관련 연구로는 DEWF[8], Early Bird System(EBS)[9], Honey-pots[10], NetBait[11] 등이 있으며 활발히 연구중이다.

본 논문의 2 장에서는 웜이 랜덤하게 생성하는 IP 주소의 랜덤성을 체크하는 방법에 대하여 기술한다. 3 장에서는 제안하는 ADUR 모델의 작동 원리에 대해서 기술한다. 4 장에서는 ADUR 모델대한 시뮬레이션 결과를 보이고, 5 장에서는 간략한 요약과 함께 향후 과제에 대하여 기술한다.

2. 랜덤성

웜은 자기 스스로 자신의 복사본을 생성하고, 스스로 다른 컴퓨터를 감염시키는 행위를 수행한다. 웜이 감염시키기 위한 대상을 결정하기 위하여 가장 많이 사용하는 방법으로는 랜덤하게 IP 주소를 생성하여 감염시키기 위한 목표 주소로 이용한다.

2.1 웜과 랜덤성의 관계

웜은 자동적으로 네트워크의 다른 시스템을 감염시키는 독립 프로그램이다. 다른 시스템을 감염시키기 위하여 웜은 랜덤 생성기를 이용하여 공격을 하기 위한 목표 시스템의 IP 주소를 생성한다. 예를 들어, Nimda 웜은 공격 대상 IP 주소를 다음과 같이 생성한다.[3]

◦생성되는 IP 주소의 50%가 처음 두 개의 Octet 이 같다.
◦생성되는 IP 주소의 25%가 처음 한 개의 Octet 이 같다.

◦생성되는 IP 주소의 25%가 모든 Octet 이 랜덤하게 생성된다.

ADUR 모델은 웜이 네트워크에 감염시킬 컴퓨터 시스템을 선택하는 대에 있어서 랜덤 IP 주소 생성기를 사용하는 것에 기반을 둔다. 생성된 랜덤 IP 주소의 나열은 랜덤성을 가진다. 우리는 IP 주소를 행렬에 표현한 후, 그 행렬의 랭크값의 계산을 통하여 랜덤성을 체크함으로서 알려지지 않은 새로운 웜에 대한 탐지를 할 수 있다.

2.2 랜덤한 이진 행렬의 랭크값

랜덤 $m \times n$ 이진 행렬의랭크[12]값은 다음의 확률을 가지고 $r = 1, 2, ..., \min(m, n)$ 의 값을 갖는다.[13]

$$2^{r<n+m-r>-nm} \prod_{i=0}^{r-1} \frac{(1-2^{i-n})(1-2^{i-m})}{(1-2^{i-r})} \quad (1)$$

(1)의 결과에 의해서 랜덤 64×64 이진행렬은 99.995%의 확률로 랭크 값이 60 이상의 값을 가진다. 이 값은 64×64 이진행렬의 모든 행이 랜덤에 가까울 때 랭크 값이 60 이상의 값을 갖는 것을 나타낸다.

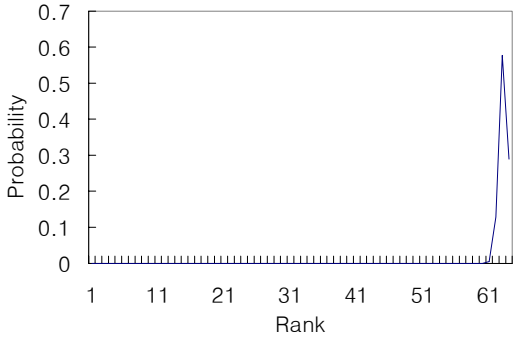

그림 1. 랜덤 행렬의 랭크값의 확률 분포

그림 1 에서 볼 수 있듯이 64×64 이진행렬이 랜덤한 행렬이라면, 그 행렬의 랭크값은 60 이상의 값을 갖는다. 60 이하의 랭크값에 대한 확률은 0 와 같다. 그러므로, 어떠한 행렬의 랭크값을 60 이상의 값을 갖는다면, 그 행렬은 랜덤한 이진 수의 나열로 이루어져 있는 행렬이라 판단할 수 있다.

3. ADUR 모델의 작동 원리

ADUR 모델은 네트워크 트래픽이 표현된 행렬의 랜덤성을 체크함으로서 현재 네트워크가 웜에 의해서 공격받고 있는지 또는 현재 네트워크가 웜에 감염이 되어있는지를 판별하는 모델이다.

3.1 ADUR 모델

```
While (until end of monitoring) Do
        //네트워크의 트래픽을 행렬에 표현한다.
        Packet Capture();
        Add in Matrix();
        //1 초가 되면 트래픽을 표현한 행렬의 랭크값을 조사한다.
        If (time tick == 1) Do
                //정상 트래픽을 제거하기 위하여 바로 전에 구성한 행렬과 XOR 한다.
                M't = Mt ⊕ Mt-1
                //랭크값이 경고기준을 넘기면 경고한다.
                If ( R(M't) >= level of warning) Do
                        Warning();
                END If
        END If
END While
```

3.2 네트워크 트래픽을 행렬에 표현

로컬 네트워크로 유입되는 트래픽을 조사하여 트래픽을 행렬에 표현하는 방법은 다음과 같다. 네트워크 트래픽을 1 초 동안 행렬에 다음과 같은 방법으로 덮어쓰기를 반복한다. 이렇게 1 초간 구성된 행렬을 이용하여 랜덤성을 체크한다.

표현하고자 하는 트래픽의 4 개의 Octet 을 (2)와 같이 구분한다.

$$IP_1.IP_2.IP_3.IP_4 \quad (2)$$

트래픽을 행렬에 표현할 때에는 64×64 의 전체행렬 M 에서 4×4 의 부분행렬 m 로 나누어 부분행렬에 표현한다. 부분행렬의 (1, 1)의 위치는 전체행렬 M 에서의 (i, j) 의 위치에 들어가게 된다. (i, j) 를 결정하는 방법은 (3)에 따른다.

$$i = (IP_4 / 16) \times 4$$
$$j = (IP_4 \bmod (16)) \times 4 \quad (3)$$

또한, 부분행렬 m 의 표현방법은 (4)를 따른다. m_i 는 부분행렬의 i 번째 행을 말한다.

$$m_1 = first\ 4bit\ of\ IP_3$$
$$m_2 = last\ 4bit\ of\ IP_3$$
$$m_3 = first\ 4bit\ of\ IP_4 \quad (4)$$
$$m_4 = last\ 4bit\ of\ IP_4$$

3.3 연속되는 두 행렬의 XOR 연산

(3)과 (4)에 의하여 구성된 64×64 의 전체행렬 M 은 1 초간의 네트워크 트래픽을 표현한 행렬이다. ADUR 는 이렇게 구성된 행렬 M 의 랜덤성을 이용하여 네트워크 내의 웜을 탐지하는 모델이다. ADUR 모델에 사용되는 행렬은 1 초의 구간 동안 구성된 행렬의 랜덤성을 조사하기 전에 바로 이전 1 초간 구성된 행렬과 XOR 을 수행한 결과이다. ADUR 는 행렬의 랜덤 성을 이용하기 때문에 일반적인 트래픽이 랜덤 값을 계산하는 것에 영향을 미치지 말아야 한다. 그러므로 바로 전에 구성된 행렬과 XOR 을 함으로서 랭크값에 영향을 주지 말아야 할 정상적인 트래픽은 대부분 소거가 된다. 소거되지 않는 정상 트래픽은 시간 t 에 새로이 생성되어진 연결이나 $t-1$ 에 끝난 연결이 되며, 이들은 웜에 의한 랜덤 스캐닝의 숫자에 비해서는 상대적으로 미약한 개수가 되어 랭크값의 변화에 큰 영향을 끼치지 못하게 된다.

$$R(M_t') = R(M_t \oplus M_{t-1}) \quad (5)$$

$R(M_t)$ 은 t 시간에서의 행렬 M 의 랭크값을 계산하는 함수이다. $R(M_t')$ 는 정상적인 트래픽을 제거한 행렬 M' 의 t 시간에서의 랭크값을 계산하는 함수이다.

정상적인 트래픽 중에 대다수가 호스트간의 1 초 이상의 연결을 유지한다. 그러한 트래픽을 XOR 을 통하여 제거할 수 있다.

3.4 랜덤 연결과 랭크값의 관계

웜은 감염시키고자 하는 대상을 결정하기 위해서 랜덤 생성기를 이용하여 랜덤하게 대상을 결정한다. 이렇게 랜덤 생성기에 의해서

생성된 랜덤 수는 네트워크에서 호스트들간의 랜덤한 연결로 표현된다. 결국 이러한 랜덤 연결은 현재 네트워크의 트래픽을 표현한 행렬에 표현되게 되며, 또한 이 랜덤 트래픽은 XOR 에서 제거되지 않는다. 그 이유는 이 연결은 랜덤하게 선택된 연결이며, 다음 시간 구간에서는 다른 랜덤한 연결을 시도하기 때문이다. 그러므로, 이 연결은 랭크값에 영향을 주게 된다.

4. ADUR 모델의 시뮬레이션

본 ADUR 모델의 시뮬레이션에 사용된 일반적인 트래픽은 2004 년 7 월 29 일에 A 대학교 트래픽을 캡쳐하여 이용하였다.

4.1 XOR 효과

ADUR 모델은 정상적인 일반 트래픽이 랭크값에 영향을 주지 않도록 하기 위하여 바로 전 시간에 구성했던 행렬과 XOR 을 통하여 정상적인 일반 트래픽을 제거한다.

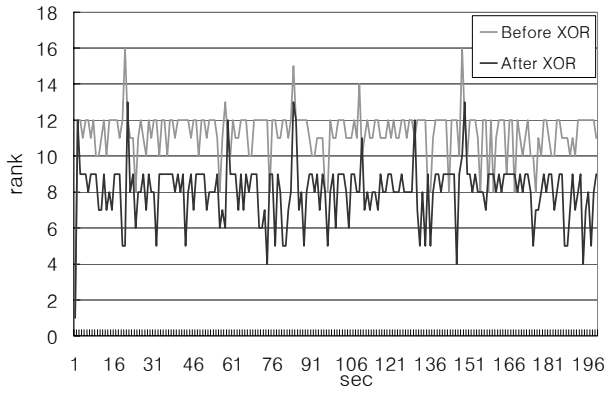

그림 2. XOR 과 랭크값과의 관계

<그림 2>은 정상적인 일반 트래픽이 제거되기 전과 제거된 후의 비교 그래프이다. 현재 네트워크 트래픽을 표현한 행렬과 바로 전 시간에 네트워크 트래픽을 표현한 행렬의 XOR 후에 남는 결과는 새로운 연결을 시도하는 트래픽만 남게 된다.

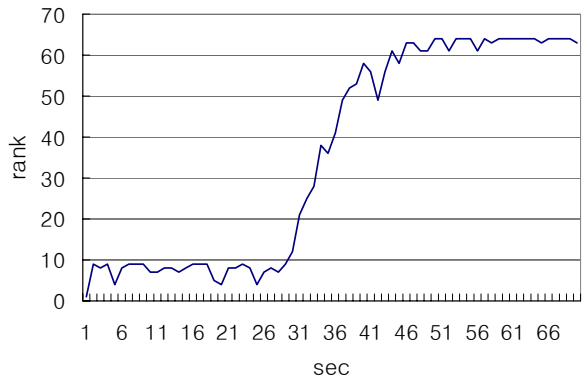

그림 3. 랜덤 연결과 랭크값의 관계

4.2 랜덤연결과 랭크값의 관계

<그림 3>은 네트워크의 트래픽을 (5)의 함수에 의해 랭크값을 조사하기 시작한지 30 초 후에서부터 1 초당 랜덤 연결을 1 씩 증가시키면서 랭크값의 변화 추이를 그래프로 표현한 것이다. 랜덤한 연결이 15 개 이상이 되면서 랭크값은 60 이상의 값이 유지되는 모습을 볼 수 있다. Code Red 웜이 초당 100 개의 쓰래드를 생성시키는 것에[14] 비교하면 ADUR 는 웜이 네트워크에 파급되는 초기에 웜을 탐지할 수 있음을 보여준다.

5. 결론

본 연구는 웜이 감염을 시키고자 하는 대상을 결정 할 때에 IP 주소값을 랜덤하게 생성하는 것을 체크하여 웜에 대한 탐지를 수행하는 모델을 제시한다. 그 방법으로 랭크값의 확률적 분포를 이용하였다. 앞으로 실제 웜의 트래픽을 이용하여 많은 테스트를 수행할 것이다.

6. 참고문헌

[1]R. Russell and A. Machie, "Code Red II Worm", Tech. Rep., Incident Analysis, Security Focus, Aug. 2001.

[2]A. Machie, J. Roculan, R. Russell, and M. V. Velsen, "Nimda Worm Anaysis", Tech. Rep., Incident Anaysis, SecurityFocus, Sept. 2001.

[3]CERT/CC, "CERT Acvisory CA-2001-26 Nimda Worm", http://www.cert.org/advisory/CA-2001-26.html, Sept. 2001.

[4]D. Song, R. Malan, and R. Stone, "A Snapshot of Global Internet Worm Activity", Tech. Rep., Arbor Network, Nov. 2001.

[5]Hyundo Park and Heejo Lee, "Evaluation of Malicious codes", Tech. Rep., IIRTIRC, 2004.

[6]Lancope Inc., 'Stealth-Watch', Intrusion Detection System, http://lancope.com

[7]Mazu Networks, 'Mazu Enforcer', Protection system against DDoS attack, http://www.mazunetworks.com

[8]Xuan Chen and John Heidemann, "Detecting Early Worm Propagation through Packet Matching", Feb. 2004.

[9]S. Singh, C. Estan, G. Varghese, and S. Savage., "The earlybird system for real-time detection of unknown worms", Tech. Rep., CS2—3-0761, University of California, San Diego, Aug. 2003.

[10]Lance Spitzner., "Honeypots, definitions and value of honeypots", May 2003.

[11]Brent N. Chun, Jason Lee, and Hakim Weatherspoon., "Netbait: a distributed worm detection service.", Feb. 2003.

[12]Howard Anton, "Elementary Linear Algebra", 7th ed. JHON WILEY & SONS, INC. 1994.

[13]George Marsaglia and Liang-Huei Tsay, "Matrices and the Strcture of Random Number Sequences", Linear Algebra and Its Applications 67, pp. 147-156, 1985.

[14]Cliff Changchun Zou, Weibo Gong and Don Towsley, "Code Red Worm Propagation Modeling and Analysis", CCS'02, November 18-22, 2002.