호스트의 연결요청과 서버의 트래픽 변화율간 관계를 이용한 DoS 공격 분석

김가을° 고광선 엄영익
성균관대학교 정보통신공학부
z-fall@hanmail.net°, {rilla91, yieom}@ece.skku.ac.kr

Analysis of DoS Attacks using Relationships between the Connection Requests of Hosts and the Traffic Transition Rate of Servers

Ka-eul Kim°, Kwang-sun Ko, Young Ik Eom
School of Information and Communication Engineering, Sungkyunkwan University

요 약

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. 현재 이러한 대부분의 DoS 공격은 인터넷 프로토콜 중 TCP 프로토콜을 주로 이용하고 있다. 이에 호스트의 연결요청이 발생하였을 경우 TCP 프로토콜을 기준으로 서버의 트래픽 변화율을 확인함으로써 DoS 공격을 분석하는 것이다. 서버의 트래픽 변화율은 호스트가 요청한 연결의 발생빈도에 따라 변화하는 서버의 연결요청 처리 시간 변화율을 확인하는 방법으로 확인할 수 있다. 이와 같은 방법으로 확인된 서버의 트래픽 변화율은 일정 시간동안의 변화율 증감을 모니터링함으로써 Dos 공격에 참여하는 호스트가 요청하는 연결의 발생빈도를 간접적으로 확인하는데 이용할 수 있다. 따라서 본 논문에서는 호스트의 연결요청과 서버의 트래픽 변화율간 관계를 이용하여 DoS 공격의 특성을 분석하고자 한다.

1. 서 론

DoS 공격은 시스템이나 네트워크 자원을 고갈시켜 해당 시스템과 네트워크가 정상적인 서비스를 제공할 수 없게 만드는 모든 공격방법을 지칭한다[1]. 통상적으로 수백 수천대의 호스트가 서버에 일정 시간동안 지속적으로 무의미하고 불필요한 트래픽을 보냄으로써 그 서버를 다운시키거나 네트워크 대역폭을 소멸시켜 서버가 서비스를 하지 못하게 한다. 현재 이러한 서비스거부공격은 대부분 TCP 프로토콜을 이용하여 서버를 공격하고 있다[2].

본 논문은 DoS 공격이 발생하였을 경우 서버의 트래픽 변화율을 확인하여 DoS 공격에 참여하는 호스트가 요청하는 연결의 발생빈도를 간접적으로 확인하고자 한다. 서버의 트래픽 변화율은 호스트 연결요청 발생빈도를 변화시킴으로써 호스트가 요청한 연결의 생성에서 소멸까지의 소요되는 시간으로 확인할 수 있다. 본 논문은 이 트래픽 변화율을 측정하기 위해서 NS-2의 NSWeb 모듈을 이용하여 가장 일반적인 단일 웹 서버 환경에서 시abf레이션을 실시한다[3].

본 논문의 구성은 다음과 같다. 2장에서는 관련연구를 소개하고, 3장에서는 NSWeb 시뮬레이션 환경을 소개하고, 4장에서는 NSWeb을 이용한 트래픽 변화결과를 나타내며 웹 서버의 트래픽 변화를 분석한다. 마지막 5장에서는 결론 및 향후 연구과제에 대해서 기술한다.

2. 관련연구

본 연구는 대학 IT연구센터 육성·지원사업의 연구결과로 수행되었음.

본 장에서는 트래픽의 정의와 웹 서버의 트래픽 변화율을 시뮬레이션하는데 사용되는 NS-2의 NSWeb 모듈을 소개한다.

2.1 트래픽의 정의

일반적으로 트래픽이란 어떤 통신장치나 시스템에 걸리는 부하를 말하며 인터넷에서는 트래픽을 패킷들의 변화로도 정의 내릴 수 있다[4].

본 논문에서는 트래픽을 웹 서버에서 처리하는 연결요청의 양으로 정의 내리고, 트래픽의 변화를 각 연결요청의 생성부터 소멸까지의 시간 변화로 정의 내린다.

2.2 NSWeb

NSWeb은 NS-2를 기반으로 한 웹 서비스 시뮬레이션 도구이며 서버와 호스트를 동시에 시뮬레이션 할 수 있다. 기본적으로 NSWeb은 NS-2와 마찬가지로 Tcl을 이용하고 서버를 시뮬레이션 하기 위해서 topology와 scenario파일을 사용한다.

NSWeb에서는 호스트의 연결 요청과 서버의 트래픽 변화율간의 관계를 확인할 수 있도록 다음과 같은 4가지 옵션들을 제공한다.

- sessions: 호스트별 연결 수
- intersession: 두 호스트 연결 간 OFF 시간
- sessionlength: 서로 다른 연결에 대한 요청 수
- interpage: 두 호스트 요청 간 OFF 시간

본 논문에서는 호스트별 연결 수의 발생빈도를 이용하도록 한다.

또한 NSWeb을 사용하면 session.log 파일이 기본적으로 생성되고, NSWeb 모듈에서 이 파일을 분석해주는 도구를 제공한다. 이 도구는 logan로 이 logan은 session.log파일을 connections, transfer, transaction,

nodes, objects라는 5개의 파일로 만들어 주는데, 본 논문에서는 connections 파일을 이용한다. connections 파일 항목 중에는 연결요청의 생성과 소멸 시간을 확인할 수 있는 Birth Time과 Death Time은 각각 존재하며, Birth~Death Time은 연결요청의 생성과 소멸 시간 간 간격을 의미한다.

본 논문은 Birth~Death Time을 트래픽의 변화로 정의하고 호스트의 연결요청 빈도 변화에 따른 Birth~Death Time 변화를 분석한다. 이는 Birth~Death Time이 각 연결요청에 대한 서비스 시간을 의미하기 때문이다. 하나의 서버가 각 연결요청에 제공하는 Birth~Death Time은 호스트 연결 요청 빈도가 증가하게 되면 줄어들게 됨을 예상할 수 있다.

3. 시뮬레이션 환경

본 장에서는 NSWeb 시뮬레이션 환경에 대해 설명한다.

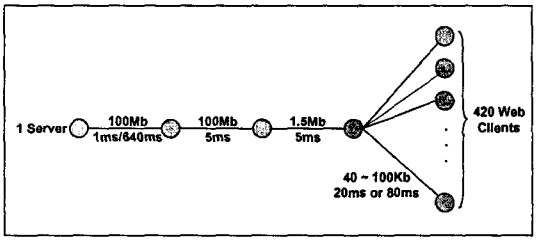
3.1 시스템 환경

- 운영 체제: 와우 리눅스 7.1 (커널 2.4.2)
- NS-2: ns-allinone-2.1b8
- NSWeb: nsweb-0.1

3.2 시뮬레이션 환경

본 논문의 시뮬레이션에서 사용하는 NS-2환경은 그림 1과 같다.

- 웹 서버: 1
- 클라이언트: 420
- 호스트와 서버 간 연결 형식
  : PIPELINED (HTTP 1.1)
- 웹 서버에서 제공하는 PAGE 수: 1000
- 웹 서버에서 제공하는 Object 형식: PAGE



(그림 1) 시뮬레이션 토폴로지

본 논문은 호스트의 연결 요청 빈도를 변화시키며 결과를 얻기 위해서 -sessions 옵션을 사용한다. 그리고 NSWeb은 -sessions 옵션을 이용하여 호스트의 연결 요청 빈도를 지수분포 (exponential distribution)에 따라 값을 변화시킨다.

3.3 시뮬레이션 순서

연결요청 발생빈도는 0.1~100까지 10배씩 증가시킨다.

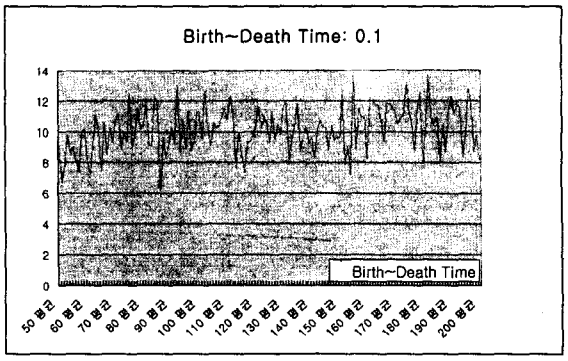
① 연결요청 발생빈도에 따라 5번 시뮬레이션을 실시한다.
② 5번의 시뮬레이션 결과의 평균을 구한다.
③ 연결 요청 발생빈도를 변화시킨다.
④ 위의 1,2번을 반복한다.
⑤ 변화에 따른 결과들의 평균을 구한다.

4. 실험 및 결과

본 장에서는 NSWeb의 시뮬레이션 결과를 그래프로 나타낸다. 결과는 호스트 연결요청의 빈도를 각각 exponential = 0.1, 1.0, 10.0, 100.0으로 변화시킨 결과를 정리하였으며, exponential = 0.1에 해당하는 결과는 다른 결과와 비교하기 위한 기본값으로 정의한다.

결과는 트래픽 변화에 주목하기 위해서 각 연결요청의 Birth시간을 1초 단위로 그룹화하고, 각 초 단위로 Birth~ Death Time의 평균을 구한다. 결과를 분석하기 위하여 분석구간을 50초에서 200초 사이로 설정하며, 해당 구간에서 Birth~Death Time값들의 평균을 구한다.

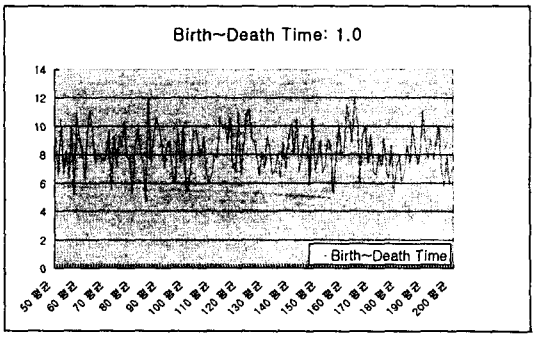
연결요청 발생빈도의 지수분포 값이 0.1일 경우 결과는 그림 2와 같다.



(그림 2) Exponential = 0.1

- Birth ~ Death Time 평균값: 10.17366 ms

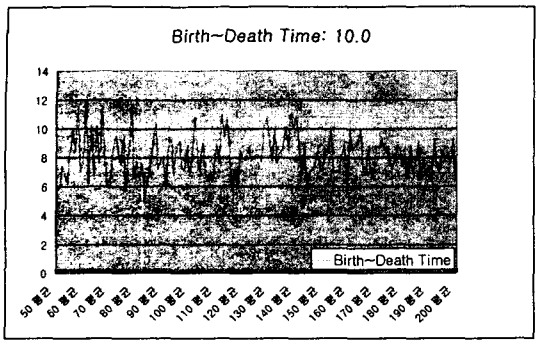
연결요청 발생빈도의 지수분포 값이 1.0일 경우 결과는 그림 3과 같다.



(그림 3) Exponential = 1.0

- Birth ~ Death Time 평균값: 8.279026 ms

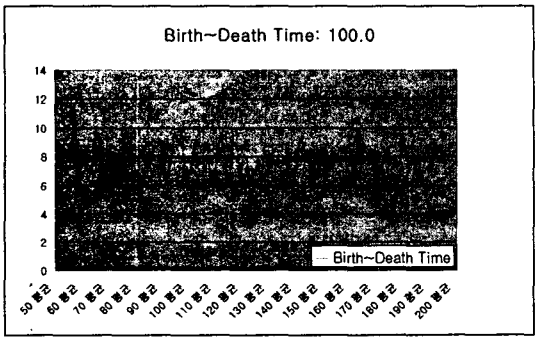
연결요청 발생빈도의 지수분포 값이 10.0일 경우 결과는 그림 4와 같다.



(그림 4) Exponential = 10.0

- Birth ~ Death Time 평균값: 8.081961 ms

연결요청 발생빈도의 지수분포 값이 100.0일 경우 결과는 그림 5와 같다.



(그림 5) Exponential = 100.0

- Birth ~ Death Time 평균값: 7.6524694 ms

따라서, 시뮬레이션 결과는 표 1과 같이 정리할 수 있다.

(표 1) 연결요청 발생빈도 증가에 따른 Birth~Death Time의 변화율

호스트 연결 요청의 발생빈도 (exponential)	Birth~Death Time	
	[50~200]의 평균	변화율 (%)
0.1	10.17366 ms	0 % (default)
1.0	8.279026 ms	-18.6 %
10	8.081961 ms	-20.6 %
100	7.6524694 ms	-24.8 %

◉ 위 결과의 평균값을 비교해 보게 되면 호스트 연결요청의 빈도가 증가할수록 Birth~Death Time이 작아진다는 것을 알 수 있다. 여기서 Birth~Death Time은 서버와 호스트가 연결되어 있는 시간인데, 이것은 서버가 호스트에게 제공하는 서비스 시간으로 해석할 수 있다. 따라서 많은 수의 호스트 연결요청이 들어오게 되면 그만큼 서버는 일정 시간 내에 다수에 서비스를 해야 하기 때문에 연결의 Birth~Death Time이 짧아진다.

◉ 호스트 연결요청 빈도가 exponential 함수에 따라 증가함에도 Birth~Death Time은 exponential 함수를 따르지 않고, Birth~Death Time 감소는 선형적인 함수를 가진다.

◉ 위 결과는 호스트 연결요청 빈도의 변화를 통해 연결요청의 Birth~Death Time의 변화율을 결과로 얻었으며, 이는 간접적으로 실제 웹 서버에서는 서버 측 연결 요청의 Birth~Death Time 변화율을 이용하여 호스트의 연결요청의 발생빈도를 알 수 있다.

5. 결론 및 향후연구

현재 많은 서버들이 DoS 공격에 대상이 되고 있다. 이러한 DoS 공격은 일정시간동안 서버에 엄청난 수의 연결요청을 맺음으로써 공격을 시작하므로, 연결요청의 수 증가에 따른 서버의 트래픽 변화에 주목할 필요가 있다.

연결요청의 수가 증가하면 서버는 호스트에 서비스 시간을 줄이게 되는데, 본 논문에서는 이 서비스 시간을 호스트의 연결요청이 생성되어서 소멸될 때까지의 시간으로 확인하였다. 실제로 시뮬레이션 결과 이 시간은 연결요청의 수가 증가할수록 감소하였다. 이러한 연결요청의 생성과 소멸 간 시간을 모니터링하여 DoS 공격에 참여하는 호스트의 연결요청 발생빈도를 유추할 수 있다.

향후 연구내용으로는 -sessions 옵션 이외에 NSWeb에서 제공하는 다양한 옵션을 변화시켜, 다양하며 복합적인 서버의 트래픽 변화율을 확인하고자 한다.

참고문헌

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

[2] Rocky K. C. Chang, Defending against "Flooding-Based Distributed Denial-of-Service Attacks: ATutorial," IEEE Communications Magazine Oct. 2002.

[3] JDrg Wallerich, Design and Implementation of a WWW Workload Generator for the NS-2 Network Simulator, http://www.net.uni-sb.de/~jw/nsweb/doku/ -last updated: Nov. 2001.

[4] Kun-chan Lan, Alefiya Hussain, and Debojyoti Dutta, "Effect of Malicious Traffic on the Network," In Proceedings of Passive and Active Measurement Workshop Apr. 2003.