JANT 한국항행학회논문지
Journal of Advanced Navigation Technology

응용 및 융합 기술

J. Adv. Navig. Technol. 19(3): 257-263, Jun. 2015

DoS 공격에 대비한 Iptables의 정책에 관한 연구

A Study on the Iptables Ruleset Against DoS Attacks

정 성 재[1] · 성 경[2*]
[1](주)스컴씨엔에스 기업부설연구소
[2]목원대학교 융합컴퓨터미디어학부

Sung-Jae Jung[1] · Kyung Sung[2*]
[1]ScomCNS Co., Ltd., Seoul 138-953, Korea
[2]Division of Convergence Computer & Media, Mokwon University, Daejeon 302-729, Korea

[요 약]

DoS 공격 및 DDoS 공격에 대한 다양한 대비 방안이 제시되고 있지만, 네트워크 및 프로토콜이 가지는 취약점으로 인해 여전히 악용되고 있다. 특히, 언제 어디서나 인터넷을 사용할 수 있는 환경이 구축되었고, 사물인터넷 시대의 진입을 앞둔 현 시점에서는 기존의 컴퓨터뿐만 아니라 가전기기 등도 DoS 공격 대상 또는 공격자 역할을 수행할 가능성이 높아지고 있다. 본 논문에서는 DoS 공격의 유형 및 특징에 대해 알아보고, 공개용 운영체제인 리눅스에 기본 탑재된 패킷필터링 도구이자 방화벽 프로그램인 iptables를 이용하여 빈번하게 발생되는 DoS 공격에 대비하는 정책 설정에 대해 기술하였다.

[Abstract]

Although a variety of preparation methods for DoS attacks and DDoS attacks are presented, it is still being exploited, the vulnerability with networks and protocols. In particular, When was built environment that can be used anywhere in the Internet, Internet of Things is entering era. Thus, the conventional computer, as well as household appliances, etc. DoS attack targets or are likely to do the attacker role is increasing. In this paper, we first find out about the type and characteristics of DoS attacks. Open source operating system, Linux has iptables that packet filtering tool and firewall programs. Using iptables to set the policy ruleset against DoS attacks.

Key word : Linux, Firewall, Packet filtering, Iptables, DoS attack.

http://dx.doi.org/10.12673/jant.2015.19.3.257

Received 6 April 2015; **Revised** 14 April 2015
Accepted (Publication) 15 May 2015 (30 June 2015)

***Corresponding Author; Kyung Sung**

Tel: +82-10-5377-6307
Email: posein@naver.com

Ⅰ. 서 론

컴퓨터 시스템에서 멀티태스킹(multi-tasking)을 지원하면서 다수의 프로세스들은 한정된 자원을 공유해서 사용하기 시작하였다. 초기의 시스템에서는 프로세스에 따른 CPU, 메모리, 디스크 등의 자원 사용에 대한 제약이 없기 때문에 하나의 프로세스가 임의적으로 자원을 점유할 수 있었다. 초기의 DoS(denial of service, 서비스 거부) 공격은 시스템 내부에서 CPU, 메모리, 디스크 등의 자원들을 고갈 시켜 특정 시스템의 다운시키는 유형이 대부분이었다. 1980년대에 개인용 컴퓨터가 보급되고 1990년대에 인터넷 사용이 보편화되면서 네트워크 기반의 다양한 DoS 공격 유형들이 등장하기 시작하였다. 1999년 DoS 공격의 확장판 형태인 DDoS (distributed denial of service) 공격이 등장하고, 2000년 초반에 야후(yahoo), 이베이(ebay), CNN, 아마존닷컴(amazon.com), 지디넷(zdnet) 등과 같은 유명 사이트들이 시스템 장애를 겪게 되면서 크게 주목 받게 되었다[1]. 2015년 현재는 DoS 및 DDoS 공격에 대한 다양한 대비 방안이 제시되고 있지만, 네트워크 및 프로토콜이 가지고 있는 취약점으로 인해 여전히 악용되고 있다. 특히, 언제 어디서나 인터넷을 사용할 수 있는 환경이 구축되었고 사물인터넷(IoT; internet of things) 시대의 진입을 앞둔 시점에서는 기존의 컴퓨터뿐만 아니라 가전기기 등도 DoS 공격 대상 또는 공격자 역할을 수행할 가능성이 높아지고 있다.

본 논문에서는 DoS 공격의 유형 및 특징에 대해 알아보고, 공개용 운영체제인 리눅스에 기본 탑재된 패킷 필터링(packet filtering) 도구이자 방화벽 프로그램인 iptables를 이용하여 DoS 공격에 대비한 정책 설정 방법에 알아본다.

Ⅱ. DoS 공격의 유형 및 특징

2-1 DoS 공격의 개요

DoS(denial of service, 서비스 거부) 공격이란 시스템이나 네트워크의 구조적인 취약점을 공격하여 정상적인 서비스를 할 수 없도록 방해하는 것으로 보통 과도한 부하를 발생시켜 데이터나 자원을 잠식하는 행위를 말한다[2]. DoS 공격은 크게 파괴 공격, 시스템 자원 고갈 공격, 네트워크 자원 공격으로 분류할 수 있다[3].

표 1. DoS 공격의 유형
Table 1. Types of DoS attacks.

유형	내용
파괴 공격	디스크, 데이터, 시스템 파괴
시스템 자원 고갈 공격	CPU, 메모리, 디스크의 부하 가중으로 인한 고갈
네트워크 자원 고갈 공격	불필요한 패킷 유발을 통한 네트워크 대역폭 고갈

2-2 주요 DoS 공격

1) Ping of death

Ping을 이용하여 ICMP (internet control message protocol) 패킷을 정상적인 크기보다 아주 크게 만들어 보내는 공격 방법으로 ICMP flooding의 일종이다. 크게 만들어진 패킷(65,535 bytes)은 네트워크를 통해 라우팅 되어 공격 대상이 되는 네트워크에 도달하는 동안에 아주 작은 조각(fragment)으로 쪼개어지고, 공격 대상이 되는 시스템은 작게 조각화된 패킷을 모두 처리해야 하기 때문에 정상적인 ping 보다 훨씬 많은 부하가 걸리게 되어 정상적인 서비스를 할 수 없게 만든다.

2) UDP flooding

UDP(user datagram protocol) 패킷을 대량 발생시켜 특정 시스템의 서비스를 방해하는 공격이다. 소스(source) 주소가 스푸핑(spoofing)된 시스템에서 UDP 패킷을 공격 대상이 되는 시스템에 대량 전송하여 네트워크 대역폭(bandwidth)을 소모하는 형태로 공격이 이루어진다.

3) TCP SYN flooding

네트워크 서비스들이 동시에 접속하는 사용자 수의 제한이 있다는 점을 악용한 공격법으로 특히 TCP (transmission control protocol)의 three-way handshaking과 밀접한 관계가 있다[4]. 공격자가 특정 시스템의 서비스를 방해를 목적으로 짧은 시간에 대량의 SYN 패킷을 보내어 접속 가능한 공간을 소진함으로서 다른 사용자의 접속을 막는 기법이다. TCP 기반으로 운영되는 서버 시스템은 클라이언트로부터 SYN 패킷을 받으면, 클라이언트에게 접속 가능하다는 의미로 SYN+ACK 패킷을 전송을 클라이언트로부터 ACK 패킷이 도착하기 까지 일정시간 대기하게 된다. SYN flooding 공격은 서버에 설정된 대기 시간 안에 서버가 수용할 수 있는 동시 접속자 수의 이상의 SYN 패킷을 보내고, ACK 패킷을 보내지 않는 형태로 공격이 이루어진다. SYN flooding 공격의 확인은 서버에서 netstat 명령으로 확인할 수 있는데, 결과의 state 항목에 SYN_RECV가 과도하게 발생했다면 이 공격을 의심할 수 있다.

4) Teardrop attack

데이터를 전송하기 위해서는 패킷을 분할하고 시퀀스 넘버를 생성하는 데, 이러한 시퀀스 넘버를 조작하거나 중첩시켜서 패킷화된 데이터를 재조합할 때 혼란을 일으켜 내부에 과부하를 발생시키는 공격방법이다. 유사한 공격으로 Boink, Bonk 등이 있다.

5) Land attack

공격자가 임의로 자신의 IP 주소 및 포트를 대상 서버의 IP 주소 및 포트와 동일하게 하여 서버를 공격한다. 이러한 패킷을 공격 시스템에 보내면 해당 시스템은 SYN 패킷의 출발지 주소를 참조하여 응답 패킷의 목적지 주소를 SYN 패킷의 출발지

주소로 설정해서 보내는데, 이 때 패킷은 외부로 나가지 않고 자신에게 되돌아온다. 이 공격법은 SYN flooding처럼 동시 사용자 수를 증가시키므로 CPU 부하까지 발생시킨다.

6) Smurf attack

공격자는 IP 주소를 공격 서버의 IP 주소로 위장하고, ICMP request 패킷을 브로드캐스트를 통해 다수의 시스템에 전송한다. 이 때 브로드캐스트를 수신한 다수의 시스템은 ICMP echo reply 패킷을 공격자가 아닌 공격 대상의 서버로 전송하게 되면서 부하를 발생시킨다.

7) Mail bomb

Mail Bomb은 보통 폭탄 메일이라고 부르는데, 스팸 메일 형태의 공격이다. 다량의 메일을 발송하여 메일 서버의 디스크 공간을 가득 채움으로서 더 이상의 메일 수신이 불가능하게 만드는 공격이다.

8) 시스템 자원 고갈 공격

디스크, 메모리, 프로세스에 대한 자원 고갈 공격으로 주로 시스템 내부에서 이루어진다. 시스템 상에서 C 컴파일러만 사용 가능하면 간단한 작업으로 손쉽게 공격을 시도할 수 있다.

root@www:~

파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)

```
[root@www ~]# cat disk.c
#include <unistd.h>
#include <sys/file.h>
main () {
 int fd;
 char buf[1000];
 fd=creat("/root/disk",0777);
 while(1)
 write(fd,buf,sizeof(buf));
}
[root@www ~]#
```

그림 1. 디스크 고갈 공격의 예
Fig. 1. Examples of disk exhaustion attack.

root@www:~

파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)

```
[root@www ~]# cat process.c
#include <unistd.h>
main () {
   while(1)
     fork();
   return(0);
}
[root@www ~]#
```

그림 2. 프로세스 고갈 공격의 예
Fig. 2. Examples of process exhaustion attack.

root@www:~

파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)

```
[root@www ~]# cat memory.c
#include <stdio.h>
main () {
   char *m;
   while(1)
     m=malloc(1000);
}
[root@www ~]#
```

그림 3. 메모리 고갈 공격의 예
Fig. 3. Examples of memory exhaustion attack.

2-3 DDoS 공격

DDoS 공격은 DoS 공격의 확장판으로 여러 대의 공격자를 분산 배치하여 동시에 DoS 공격을 함으로서 공격 대상이 되는 시스템이 정상적인 서비스를 할 수 없도록 방해하는 공격이다.

DDoS 공격은 DoS 공격을 확장하여 사용하는 것으로 공격 범위가 매우 방대하며 최적의 효과를 보기 위해서는 공격을 증폭 시키는 중간자의 역할이 매우 중요하다. DDoS 공격은 이러한 특성화된 자동화된 도구를 이용하여 공격이 이루어진다.

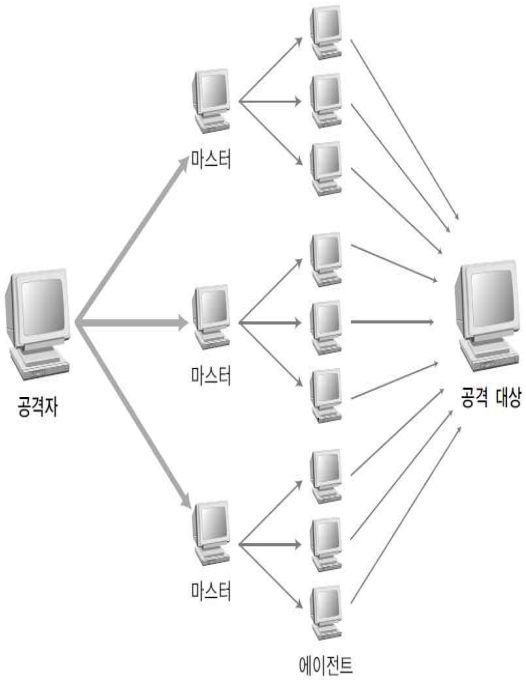

그림 4. DDoS 공격의 예
Fig. 4. Examples of DDoS attacks.

표 2. DDoS 공격 도구
Table 2. DDoS attack tool.

도구	설 명
Trinoo	다수의 호스트로부터 통합된 UDP Flooding 공격을 시행한다.
TFN	Trinoo와 거의 유사한 공격 도구로 UDP Flooding 뿐만 아니라, TCP SYN Flooding, ICMP 브로드캐스트 공격도 가능하다. 그러나, 공격자 시스템과 마스터 시스템 간 연결이 암호문이 아닌 평문으로 되어 있어서 공격자가 노출된 가능성이 높다.
TFN 2K	TFN의 발전된 형태로 통신에 특정 포트를 사용하지 않고, 암호화를 사용한다. 프로그램에 의해 UDP, TCP, ICMP가 복합적으로 사용되고, 포트도 임의로 결정한다. UDP Flooding, TCP SYN Flooding, ICMP Flooding, Smurf 공격을 사용한다.
Stacheldraht	독일어로 “철조망”이라는 뜻으로 Trinoo, TFN을 참고하여 만들어졌다. TFN 2K처럼 통신할 때 암호화 기능이 추가되었고, 접속한 시도가 사용자가 올바른 공격자인지 확인하기 위해 패스워드 입력을 요구한다. 특히, 마스터와 에이전트들이 자동으로 갱신되는 특징이 있다. UDP Flooding, TCP SYN Flooding, ICMP Flooding, Smurf 공격을 사용한다.

Ⅲ. iptables의 개요

3-1 iptables의 개요

리눅스는 등장했을 무렵 초기부터 접속 제한을 위해 UNIX에서 사용하던 TCP wrapper를 비롯하여, 리눅스 커널 1.0 버전에서는 BSD의 ipfw, 커널 2.0 버전에서는 ipfwadm, 커널 2.2 버전에서는 ipchains, 커널 2.4 버전 이후로는 iptables를 사용하고 있다. iptables는 패킷 필터링(packet filtering) 도구로서 방화벽 구성이나 NAT(network translation address)에 사용된다[5]. 사용자가 iptables라는 명령으로 정책을 설정하면 해당 정책에 의거하여 동작하고, OSI 참조 모델의 2,3,4 계층(data Link, network, transport)에서 정책을 수행한다. iptables는 패킷 필터링을 직접적으로 수행하지 않고, 커널에 있는 넷필터(netfilter)라는 모듈이 필터링을 수행한다[6]. 넷필터는 리눅스가 제공하는 모든 종류의 패킷 필터링과 맹글링(mangling) 도구로 네트워크 스택으로 함수를 후킹(hooking)하는데 사용할 수 있는 커널 내부의 프레임워크이다. iptables는 패킷에 대한 필터링을 수행하게 설계된 함수를 네트워크 스택으로 후킹하기 위해 넷필터 프레임워크를 사용한다. 즉 넷필터는 iptables가 방화벽 기능을 구현할 수 있게 프레임워크를 제공한다고 보면 된다.

3-2 iptables의 구조

iptables는 커널 2.2에서 사용되던 ipchains의 사슬(chain) 구조를 그대로 승계했지만, 4계층 구조의 테이블(table)을 만들어 사용하여 이름을 iptables라 명명하였다. ipchains와 거의 유사하게 사슬에 정책을 설정하여 사용하지만, 기능과 역할을 강화하기 위해 테이블로 확장하여 테이블별로 각각의 사슬을 지정하고 해당 사슬에 정책을 설정하도록 되어 있다. iptables는 filter, nat, mangle, raw와 같은 4개의 테이블이 있다. filter는 iptables의 기본 테이블로 패킷 필터링을 담당한다. ipchains 이전 프로그램들은 filter 테이블만 가지고 사용했다고 보면 된다. nat 테이블은 network address translation의 약자처럼 IP의 주소를 변환시키는 역할을 수행한다. ipchains의 FORWARD 사슬 역할을 확장한 것으로 한 개의 공인 IP주소를 가지고 여러 대의 컴퓨터를 사용하거나, 하나의 공인 IP 주소를 가지고 여러 대의 서버를 운영하고자 할 때 주로 사용한다. mangle은 패킷 데이터를 변경하는 특수 규칙을 적용하는 테이블로 성능 향상을 위한 TOS (type of service)를 설정하고, raw는 넷필터의 연결추적 하위시스템과 독립적으로 동작해야 하는 규칙을 설정하는 테이블이다. 각 테이블은 자신만의 고유한 사슬 집합을 가지고 있지만, 사용자가 INPUT_ESTABLISHED나 DMZ_ NETWORK와 같은 공통 태그와 관련된 규칙집합을 만들기 위해 사용자 정의 사슬을 생성할 수 있다.

패킷 필터링 및 방화벽 구성은 기본 테이블인 filter의 3개 사슬에 정책을 설정하면 되고, 다수의 서버 관리를 할 때는 filter 및 nat 테이블을 연동해서 사용한다. 특히 nat 테이블의 PREROUTING과 POSTROUTING 사슬은 커널 내부에서 IP 라우팅 계산을 수행하기 전과 후에 패킷 헤더를 수정하기 위해 사용한다.

표 3. iptables의 테이블과 사슬
Table 3. Table & Chains of iptables.

사슬(chain)	테이블(table)			
	filter	nat	mangle	raw
INPUT	○		○	
FORWARD	○		○	
OUTPUT	○	○	○	○
PREROUTING		○	○	
POSTROUTING		○	○	○

표 4. Filter 테이블의 사슬 및 기능
Table 4. Chain and Functionality of the filter table.

사슬	기능
INPUT	패킷필터링 및 방화벽 관련 정책들을 설정하는 사슬로 실제적인 접근 통제를 담당하는 역할을 수행한다. 커널 내부에서 라우팅 계산을 마친 후 로컬 리눅스 시스템이 목적지인 패킷(즉 로컬 소켓이 목적지인 패킷)에 적용된다.
OUTPUT	다른 시스템으로의 접근을 차단할 때 사용하는 사슬로 리눅스 시스템 자체가 생성하는 패킷을 제어하는 사슬이다.
FORWARD	리눅스 시스템을 통과하는 패킷을 관리하는 사슬로 한 네트워크를 다른 네트워크와 연결하기 위해 iptables 방화벽을 사용해서 두 네트워크 간의 패킷이 방화벽을 통과하는 경우에 사용되고, NAT 기반으로 하나의 공인 IP를 여러 대의 사설 IP를 사용하는 시스템들을 공유해서 사용할 경우 사설 IP를 사용하는 시스템의 접근 제어 정책을 설정할 때 사용한다.

3-3 iptables의 사용

iptables를 이용하여 정책을 설정할 때 가장 중요한 것은 실질적 룰(rule)에 해당하는 매치(match)와 타겟(target)이다. 타겟은 iptables에서 패킷이 규칙과 일치할 때 취하는 동작을 말하고, 매치는 iptables가 규칙 타겟에 의해 명시되는 동작에 따라 패킷을 처리하기 위해서 만족해야 하는 조건들이다. 예를 들어 TCP 패킷에만 적용하려면 –p (--protocol) 매치를 사용해서 적용시키면 된다. iptables의 기본 사용법은 'iptables [-t table] acton chain match [-j target]'이다. 테이블(table)의 기본 설정은 filter이고, 다른 테이블 지정할 때에는 –t 옵션을 사용해서 사용한다. 액션(action)은 사슬을 지정, 설정, 제어할 때 사용하는데, 주로 –N, -A와 같은 대문자 옵션을 사용한다. 사슬(chain)을 설정하고 하는 사슬을 명기하는데 INPUT, OUTPUT과 같이 입력하면 되고, 대소문자를 구문한다. 마지막으로 실질적인 룰에 해당하는 매치와 타겟을 지정하면 된다. 매치는 -d, -p와 같은 소문자 옵션을 사용해서 설정하고, 타겟은 –j(--jump) 옵션을 사용하여 설정한다.

표 5. iptables의 주요 타겟
Table 5. Essential target of the iptables.

Target	설 명
ACCEPT	패킷을 허가하는 것으로 본래 라우팅대로 진행
DROP	패킷을 거부하는 것으로 더 이상 어떤 처리도 수행되지 않고 버림
LOG	패킷을 syslog에 전달하여 기록. 일반적으로 /var/log/messages에 저장
REJECT	패킷을 버리고 동시에 적당한 응답 패킷을 전달. 예를 들면 TCP인 경우 TCP 재설정(reset) 패킷, UDP인 경우 ICMP 포트 도달 불가(port unreachable) 메시지를 전송
RETURN	호출 사슬 내에서 패킷 처리를 계속 진행

표 6. iptables의 액션
Table 6. Action of the iptables.

Action	설 명
–N	새로운 사용자 정의 사슬을 만든다.(--new-chain)
–X	비어있는 사슬을 제거한다. 단 기본 사슬은 제거할 수 없다. (--delete-chain)
–P	사슬의 기본 정책을 설정한다. (--policy)
–L	현재 사슬의 규칙을 나열한다. (--list)
–F	사슬로부터 규칙을 제거한다. (--flush)
–Z	사슬내의 모든 규칙들의 패킷과 바이트의 카운트를 0으로 만든다. (--zero)
–A	사슬에 새로운 규칙을 추가한다. 해당 사슬에 맨 마지막 규칙으로 등록된다. (--append)
–I	사슬에 규칙을 맨 첫 부분에 삽입한다. 룰 넘버를 사용하여 특정 부분에 삽입할 수도 있다. (--insert)
–R	사슬의 규칙을 교환한다. (--replace)
–D	사슬의 규칙을 제거한다. (--delete)

표 7. iptables의 주요 매치
Table 7. Essential match of the iptables.

매치	설 명
–s	출발지 IP주소나 네트워크와 매칭. 도메인, IP 주소, 넷마스크값을 이용하여 표기 (--source, --src)
–d	목적지 IP주소나 네트워크와 매칭. 도메인, IP 주소, 넷마스크값을 이용하여 표기(--destination, --dst)
–p	특정 프로토콜과 매칭. TCP, UDP, ICMP와 같은 이름을 사용하고 대소문자는 구분하지 않음. 이 옵션을 사용하지 않으면 모든 프로토콜이 대상이 됨.(--protocol)
–i	입력 인터페이스와 매칭(--in-interface)
–o	출력 인터페이스와 매칭(--out-interface)
!	아닌(NOT)의 의미로 사용하는데, 특정 매칭을 제외할 때 사용
–m	좀 더 세밀하게 제어할 때 사용하는 매칭 옵션 (--match)
--state	연결 상태와 매칭. INVALID, ESTABLISHED, NEW, RELATED를 사용
--string	특정한 패턴과 매칭

Ⅳ. iptables의 정책 설정

iptables를 이용하여 DoS 공격 유형 중 빈번하게 발생하는 ICMP flooding, UDP flooding, TCP flooding 공격에 대한 정책 설정을 하도록 한다.

4-1 ICMP flooding 대비 설정

ICMP flooding 공격은 공격자가 대량의 icmp-echo-request 패킷을 발생시켜 응답을 요구하는 공격이므로 해당 패킷의 요청은 거부함으로써 대비가 가능하다. ICMP라는 새로운 사슬

생성 후에 INPUT 사슬에 해당 정책을 추가하므로 손쉽게 대비할 수 있다.

```
# iptables -N ICMP
# iptables -A INPUT -p icmp -j ICMP
# iptables -A ICMP -p icmp  --icmp-type echo-request
 -j DROP
```

4-2 UDP flooding 대비 설정

UDP flooding 공격은 ICMP flooding 공격처럼 특정 패킷을 거부하는 설정은 할 수 없다. 특히 DNS 서비스를 제공하는 서버라면 UDP 프로토콜을 완전 거부하는 정책의 수립도 불가능하다. 이런 경우에는 특정 IP 주소에서 초당 들어오는 패킷의 수를 정해서 일정 수준 이상이면 거부하는 정책으로 설정하면 된다. UDP라는 사슬 생성 후에 특정 IP 주소로부터 1초에 10개 이상의 패킷이 들어오면 거부하고, 관련 로그를 기록하도록 설정한다.

```
# iptables -N UDP
# iptables -A INPUT -p udp -j UDP
# iptables -A UDP -p udp --dport 80 -m recent --update
 --seconds 1 --hitcount 10 -j DROP
 # iptables -A UDP -j LOG --log-prefix "UDP FLOOD“
```

4-3 TCP flooding 대비 설정

서버에서 제공되는 서비스들의 대부분이 TCP 프로토콜 기반으로 제공되므로 운영 중인 서비스의 상황에 맞게 설정해야 한다. 가장 많이 사용되는 정책으로는 초 당 허용되는 접속 개수 제한, IP 주소 당 허용되는 접속 개수 제한, 특정 IP 주소에서 오는 초 당 요구 수 제한이 있다.

1) 초 당 허용되는 접속 개수 제한

```
# iptables -A INPUT -p tcp --dport 80 --syn -m limit --limit
  100/s -j ACCEPT
```

2) IP 주소 당 허용되는 접속 개수 제한

```
 # iptables -A INPUT -p tcp --dport 80 --syn -m connlimit
--limit -above 30 -j DROP
```

3) 특정 IP 주소에서 오는 초 당 요구 수 제한

```
# iptables -A INPUT -p tcp --dport 80 -m recent --update
 --seconds 1 --hitcount 10 -j DROP
```

Ⅴ. 결 론

DoS 공격은 무선 네트워크 사용이 보편화되고, 클라우드 컴퓨팅 기반으로 전환되는 현 시점에서도 여전히 위협적인 공격 방법으로 볼 수 있다. 특히, IoT 시대의 앞 둔 현 시점에서는 다양한 기능을 제공하기 위해 운영체제가 탑재된 많은 기기들이 DoS 공격 대상 또는 공격자 역할을 수행할 가능성이 높아지고 있다. 최근 서버나 데스크톱뿐만 아니라 스마트폰, 자동차, 가전 기기 등 다양한 분야에서 공개용 운영체제인 리눅스가 사용되고 있다. 리눅스 커널에 기본 탑재되어 있는 iptables를 활용하면 DoS 공격에 대비한 방화벽도 손쉽게 구성할 수 있고, 시스템이나 기기 자체적으로 패킷 필터링 정책 설정을 통한 보안 강화에도 큰 도움이 되리라고 사료된다. 향후 연구 과제로는 실질적인 사례 적용에 따른 성능 분석을 통해 다양한 DoS 공격에 따른 정량적 수치를 제시할 필요가 있을 것으로 판단된다.

감사의 글

이 논문은 2104년도 목원대학교 연구년 지원에 의하여 연구되었습니다.

참고 문헌

[1] Fortinet [Internet]. Available: https://blog.fortinet.com/post/ddos-a-brief-history/.

[2] US-CERT, United States Computing Emergency Readiness Team [Internet]. Available: https://www.us-cert.gov/ncas/tips/ST04-015/.

[3] D. I. Yang, *Information Security Introduction and Practice*, 3th ed. Seoul, Korea: Hanbit, 2014.

[4] CERT, Software Engineering Institute[Internet]. Available: https://www.cert.org/historical/advisories/CA-1996-21.cfm?/

[5] Y. M. Bae, and S. J Jung, “A study on the linux firewall,” *Journal of Security Engineering*, Vol. 8, No. 5, pp. 599-610, Oct. 2011.

[6] Netfilter Project [Internet]. Available: http://www.netfilter.org/.

정 성 재 (Sung-Jae Jung)
1998년 2월 : 한남대학교 컴퓨터공학과 (공학사)
2003년 8월 : 한남대학교 컴퓨터공학과 (공학석사)
2011년 2월 : 한남대학교 컴퓨터공학과 (공학박사)
2005년 3월 ~ 2010년 2월: 한남대학교 국제IT교육센터 전임강사
2012년 9월 ~ 현재 : ㈜스컴씨엔에스 기업부설연구소 소장
※ 관심분야 : 리눅스, 정보보호, 시스템보안, 클라우드 컴퓨팅, 서버 가상화

성 경 (Kyung Sung)
1994년 3월 ~ 2004년 2월 : 동해대학교 컴퓨터공학과 교수
2004년 3월 ~ 2014년 2월 : 목원대학교 컴퓨터교육과 교수
2014년~현재 : 목원대학교 공과대학 융합컴퓨터미디어학부 교수
※ 관심분야 : 정보보호 및 정보관리,가상현실,컴퓨터네트워크, 신경회로망, 컴퓨터교육