리눅스 방화벽 IPTABLES 활용 가이드

작성자 : 기술지원부 홍 종 우 (shairin@nextline.net)
작성일 : 2009년 07월 01일

(1) IPTABLES 란

일반적으로 외부와 연결된 네트워크는 안전하지 않으며 언제든지 악의적인 목적을 가지고 있는 사용자로부터 공격을 받을 수 있습니다. 이러한 공격으로부터 리눅스 서버를 보호하기 위하여 패킷 필터링을 통해 기본적인 방화벽을 구성 할 수 있으며 커널의 패킷 필터링 테이블에 필터링 규칙을 삽입 하거나 삭제하는 도구가 iptables 입니다.

1.1시리즈의 리눅스 커널부터 패킷 필터링을 포함하기 시작했으며 1세대 BSD의 ipfw 부터 ipfwadm, ipchanins를 거쳐 4세대의 패킷 필터링 설정도구인 iptables가 사용되기 시작 하였습니다.
Iptables를 사용하기 위해서는 netfilter를 가지고 있는 커널이 필요하며 netfilter는 2.3.15이상의 리눅스커널에 포함되어 있고 커널 설정에서 CONFIG_NETFILTER 에 'Y' 로 지정하고 컴파일한 커널이어야 사용할 수 있습니다.

(2) IPTABLES 설치하기

rpm으로 Iptables가 설치되어 있는지 확인 합니다.

```
[root@localhost5 ~]# rpm -qa | grep iptables
```

```
[root@localhost6 ~]# rpm -qa |grep iptables
iptables-ipv6-1.3.5-4.el5
iptables-1.3.5-4.1
```

설치가 되어 있지 않을 시 yum을 이용하거나 리눅스 패키지에 맞는 rpm을 다운받아 설치 하거나 소스 파일을 직접 다운받아 설치 합니다.

[1] rpm으로 설치하는 방법

http://mirror.secuidc.com/centos/5.3/os/i386/CentOS/

국내 centos 미러 사이트는 상기와 같으며 centos 5.3의 경우 위의 경로에 있는 **iptables-1.3.5-4.el5.i386.rpm** 파일을 다운받아 설치 합니다.

Wget 명령어를 이용해 **rpm** 파일을 다운 받습니다.

 # wget http://mirror.secuidc.com/centos/5.3/os/i386/CentOS/iptables-1.3.5-4.el5.i386.rpm

```
[root@localhost6 ~]# wget http://mirror.secuidc.com/centos/5.3/os/i386/CentOS/iptables-1.3.5-4.el5.i386.rpm
--06:28:23--  http://mirror.secuidc.com/centos/5.3/os/i386/CentOS/iptables-1.3.5-4.el5.i386.rpm
Resolving mirror.secuidc.com... 218.150.79.120
Connecting to mirror.secuidc.com|218.150.79.120|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 240736 (235K) [audio/x-pn-realaudio-plugin]
Saving to: `iptables-1.3.5-4.el5.i386.rpm'

100%[======================================>] 240,736     --.-K/s   in 0.05s
```

②다운받은 rpm을 설치 합니다.

```
# rpm -Uvh iptables-1.3.5-4.el5.i386.rpm
```

```
[root@localhost6 ~]# rpm -Uvh iptables-1.3.5-4.el5.i386.rpm
준비 중...                ########################################### [100%]
   1:iptables             ########################################### [100%]
```

[2] Yum을 이용해 설치하는 방법

Yum 명령어를 이용해 서버에서 바로 설치를 합니다.
#yum install -y iptables

```
[root@localhost6 ~]# yum install -y iptables
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: ftp.iij.ad.jp
 * updates: ftp.iij.ad.jp
 * addons: ftp.iij.ad.jp
 * extras: ftp.iij.ad.jp
base                                                    | 1.1 kB     00:00
updates                                                 |  951 B     00:00
primary.xml.gz                                          | 193 kB     00:00
updates                                                       301/301
addons                                                  |  951 B     00:00
extras                                                  | 1.1 kB     00:00
Setting up Install Process
Parsing package install arguments
Resolving Dependencies
--> Running transaction check
---> Package iptables.i386 0:1.3.5-4.el5 set to be updated
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package            Arch           Version                 Repository      Size
================================================================================
Installing:
 iptables           i386           1.3.5-4.el5             base           235 k

Transaction Summary
================================================================================
Install       1 Package(s)
Update        0 Package(s)
Remove        0 Package(s)

Total size: 235 k
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Finished Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing     : iptables                                              [1/1]

Installed: iptables.i386 0:1.3.5-4.el5
Complete!
```

[3]netfilter.org사이트에서 직접 소스를 다운 받아 설치하기

①wget 명령어를 이용해 iptables를 다운 받습니다.

wget http://ftp.netfilter.org/pub/iptables/iptables-1.4.4.tar.bz2

```
[root@localhost6 ~]# wget http://ftp.netfilter.org/pub/iptables/iptables-1.4.4.
tar.bz2
--06:40:24--  http://ftp.netfilter.org/pub/iptables/iptables-1.4.4.tar.bz2
Resolving ftp.netfilter.org... 213.95.27.115, 2001:780:45:1d:20d:93ff:fe9b:e443
Connecting to ftp.netfilter.org|213.95.27.115|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 452656 (442K) [application/x-tar]
Saving to: `iptables-1.4.4.tar.bz2'

100%[======================================>] 452,656      170K/s    in 2.6s

06:40:28 (170 KB/s) - `iptables-1.4.4.tar.bz2' saved [452656/452656]
```

②다운받은 소스압축파일의 압축을 해제 합니다.

tar xjvf iptables-1.4.4.tar.bz2

```
[root@localhost6 ~]# tar xjvf iptables-1.4.4.tar.bz2
iptables-1.4.4/
iptables-1.4.4/ip6tables.8.in
iptables-1.4.4/ip6tables-restore.c
iptables-1.4.4/xtables.c
iptables-1.4.4/iptables-multi.c
iptables-1.4.4/ip6tables-restore.8
iptables-1.4.4/ip6tables-save.c
iptables-1.4.4/ip6tables-multi.h
iptables-1.4.4/iptables-apply.8
```

③해제한 디렉토리로 이동후 ./configure 합니다.

#cd iptables-1.4.4
./configure

```
[root@localhost6 ~]# cd iptables-1.4.4
[root@localhost6 iptables-1.4.4]# ./configure
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking for gcc... gcc
checking for C compiler default output file name... a.out
checking whether the C compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables...
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
```

④설치 합니다.

```
# make && make install
```

```
[root@localhost6 iptables-1.4.4]# make && make install
make  all-recursive
make[1]: Entering directory `/root/iptables-1.4.4'
Making all in extensions
make[2]: Entering directory `/root/iptables-1.4.4/extensions'
  GEN      initext4.c
  CC       initext4.o
  AR       libext4.a
  GEN      initext6.c
  CC       initext6.o
  AR       libext6.a
  GEN      matches4.man
        + ./libipt_addrtype.man
        + ./libipt_ah.man
        + ./libxt_cluster.man
```

(3) iptables를 이용하여 기본적인 포트 필터링 하기

커널은 필터 테이블에 삭제가 불가능한 INPUT,OUTPUT,FORWARD 세개의 기본 체인을 가지고 있으며 이 외에 체인을 추가/삭제가 가능 합니다.

체인은 규칙의 점검표로 각 규칙은 "패킷의 헤더가 이렇게 되어 있으면 이곳에서 무엇을 하라"는 형태로 되어 있으며 규칙이 그 패킷에 맞지 않으면 다음 규칙을 참고 합니다.

마지막으로 더 이상 고려할 규칙이 없으면 커널은 무엇을 할 것인가를 결정하기 위하여 그체인의 정책을 확인합니다.

[1]Iptables의 기본적인 옵션
체인 관련 옵션

# iptables -N	새로운 체인 만들기 (-N)
# iptables -X	비어있는 체인을 제거하기 (-X)
# iptables -P	미리 만들어진 체인의 정책을 바꾸기 (-P)
# iptables -L	어떤 체인의 규칙들을 나열하기 (-L)
# iptables -F	체인으로부터 규칙들을 지우기 (-F)
# iptables -Z	체인내의 모든 규칙들의 패킷과 바이트의 카운드를 0으로 만들고 (-Z)

②체인 내부의 규칙을 조작하는 옵션

# iptables -A INPUT -p tcp --dport 80 -j ACCEPT	체인에 새로운 규칙을 추가하기 (-A)
# iptables -I INPUT 1 -p tcp --dport 80 -j ACCEPT	체인의 어떤 지점에 규칙을 삽입하기 (-I)
# iptables -R INPUT 1 -p tcp -dport 80 -j ACCEPT	체인의 어떤 지점의 규칙을 교환하기 (-R)
# iptables -D INPUT 1	체인의 어떤 지점의 규칙을 제거하기 (-D)
# iptables -D INPUT -p tcp -dport 80 -j ACCEPT	체인에서 일치하는 첫번째 규칙을 제거하기 (-D)

③필터링 옵션

# iptables -A INPUT -p tcp/udp/all	Tcp 나 udp나 all 로 모두를 지정할수 있다.
# iptables -A INPUT -p tcp -s 192.168.0.3	-s 는 접근하는 클라이언트의 아이피를 지정한다.
# iptables -A INPUT -p tcp -d 192.168.0.4	-d 는 클라이언트가 접속할 서버의 아이피를 지정한다.forward 가 아닌경우는 보통 자신이 된다.
# iptables -A INPUT -i eth0 -p tcp # iptables -A OUTPUT -o eth1 -p tcp	보통 -i 는 들어오는 디바이스를 지정하고 -o 는 패킷이 나가는 디바이스를 지정한다. Forwarding 시에 유용하다.
# iptables -A INPUT -p tcp -j ACCEPT	타겟에 대한 행동을 지정한다.

[2]iptables**의 설정 및 저장방법**

Iptables의 설정은 명령어를 통해 직접 설정할 수 있으며 리부팅시 초기화 되기 때문에 스크립트 파일을 생성해 /etc/rc.local 등에 삽입하여 부팅시 설정이 되도록 하거나 다음 명령어를 사용하여 현재의 규칙을 저장 합니다.

```
#/sbin/service  iptables  save
```

규칙을 저장 시 /etc/sysconfig/iptables 파일에 기록되어 iptables가 재시작 되거나 부팅이 될 때마다 적용이 됩니다. (위 파일은 iptables 시작 스크립트에서 기본으로 불러오는 파일입니다.)

```
# iptables-save > /root/iptables.dat
```

또는 위와같이 일정한 파일에 저장할수 있도록 지정할 수가 있습니다.

저장된 룰셋을 적용하는 방법은 아래와 같다.

```
# iptables-restore < /root/iptables.dat
```

[3]Iptables **적용 예제**
Iptables를 이용하여 규칙을 수정 하실시 네트워크가 단절될 수 있으니 주의를 요하며 가급적 console 에서 작업을 하시기 바랍니다.

```
#iptables -P INPUT DROP
#iptables -P FORWARD DROP
```

입력체인인 INPUT 체인과 다른 네트워크로 포워딩 되는 FORWARD 체인의 기본 규칙을 DROP으로 수정합니다.(콘솔에서 작업하거나 스크립트를 통해 사용할 포트를 같이 오픈하지 않으면 원격상에

서는 즉시 차단이 되어 네트워크가 단절되니 주의하시기 바랍니다.)

```
#iptables -P OUTPUT ACCEPT
```

출력 체인인 OUTPUT 체인의 기본 규칙을 ACCEPT로 수정 합니다.

```
# iptables -A INPUT -i lo -j ACCEPT
```

입력체인인 INPUT 체인에 로컬 Loopback 장치 트래픽 허용 합니다.

```
#iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
#iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

INPUT 체인과 OUTPUT 체인에 존재하는 접속에 속하는 패킷(응답 패킷을 가진것)과 기존의 접속 부분은 아니지만 연관성을 가진 패킷 (ICMP 에러나 ftp데이터 접속을 형성하는 패킷)을 허용하는 규칙을 추가합니다.

```
#iptables -A INPUT -p tcp --sport 1024: --dport 22 -m state --state NEW -j ACCEPT
```

tcp 프로토콜의 1024이후 포트를 출발지로 하며 목적지가 22포트인 트래픽을 허용하는 규칙을 입력 체인인 INPUT 체인에 추가합니다.

상기의 룰을 스크립트로 만들어 적용을 시킨 후 현재 iptables 룰을 확인하는 명령어를 사용하면 다음과 같이 확인할 수 있습니다.

```
#iptables -L
```

```
[root@localhost6 ~]# iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTAB
LISHED
ACCEPT     tcp  --  anywhere             anywhere            tcp spts:1024:65535
 dpt:ssh state NEW

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTAB
LISHED
```

기본
ssh
있습

Iptables 의 룰셋을 확인하실 때 아래와 같이 하시면 보시기 더 편리합니다.

```
#iptables -nL
```

또는 아래와 같이 각 룰셋의 적용순서까지 확인 가능한 방법도 있습니다.

```
#iptables -nL --line-numbers
```

iptables를 적용시에는 항시

iptables -P INPUT DROP 과 같은 명령은 주의를 해야합니다.
이 명령은 기본정책자체가 모두 차단하기 때문에 정책 설정 이후에는
반드시 iptables -A INPUT -p tcp -dport 22 -j ACCEPT 와 같이 터미널 접속은 반드시 허용해야만 합니다.

예제는 아래와 같습니다.아래의 예제와 같이 INPUT 정책을 DROP 시킬경우에는 터미널과 웹 접속은 기본적으로 열어주는 것이 좋습니다.

```
#!/bin/sh

# iptables 실행파일의 위치를 IPT 변수로 지정
IPT=/sbin/iptables

# INPUT, FORWARD 의 기본정책은 DROP, OUTPUT은 모두 오픈
$IPT -P INPUT DROP
$IPT -P FORWARD DROP
$IPT -P OUTPUT ACCEPT

# 22, 80번 포트의 tcp 접속은 모두 접속을 오픈
$IPT -A INPUT -p tcp --dport 22 -j ACCEPT
$IPT -A INPUT -p tcp --dport 80 -j ACCEPT
```

정책을 모두 비우고 삭제할시에는 아래와 같은 스크립트를 만들어서 테이블을 비우고 삭제합니다.

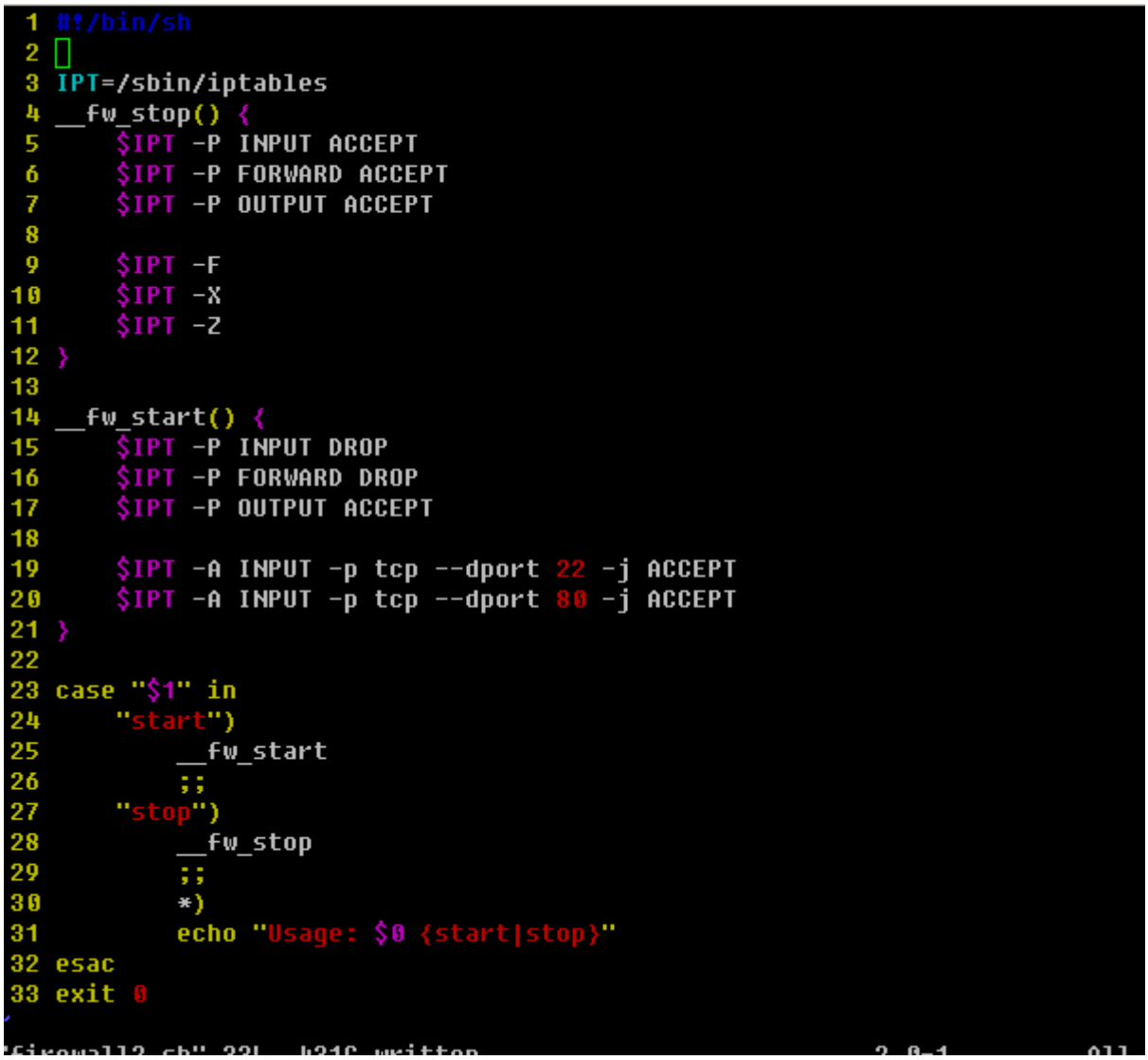
```
#!/bin/sh

# iptables 실행파일의 위치를 IPT 변수로 지정
IPT=/sbin/iptables

# 정책 해제시에는 반드시 아래와 같이 정책을 ACCEPT로 변경한다.
# INPUT, FORWARD 의 기본정책은 DROP, OUTPUT은 모두 오픈
$IPT -P INPUT ACCEPT
$IPT -P FORWARD ACCEPT
$IPT -P OUTPUT ACCEPT

# 정책들을 체인으로부터 삭제합니다.
# -F는 모든 체인테이블을 비운다.
# -X는 모든 체인의 룰셋을 삭제한다.
# -Z는 모든 체인의 바이트를 0으로 만든다.
$IPT -F
$IPT -X
$IPT -Z
```

Funtion과 case 문을 사용해서 좀더 미려한 스크립트로 만들수 있습니다.
아래와 같이 만들고 start/stop 인자를 통해서 시작/중지를 할 수가 있습니다.



```
#!/bin/sh

IPT=/sbin/iptables
__fw_stop() {
    $IPT -P INPUT ACCEPT
    $IPT -P FORWARD ACCEPT
    $IPT -P OUTPUT ACCEPT

    $IPT -F
    $IPT -X
    $IPT -Z
}

__fw_start() {
    $IPT -P INPUT DROP
    $IPT -P FORWARD DROP
    $IPT -P OUTPUT ACCEPT

    $IPT -A INPUT -p tcp --dport 22 -j ACCEPT
    $IPT -A INPUT -p tcp --dport 80 -j ACCEPT
}

case "$1" in
    "start")
        __fw_start
        ;;
    "stop")
        __fw_stop
        ;;
        *)
        echo "Usage: $0 {start|stop}"
esac
exit 0
```

다양한 스크립트를 만들수 있지만, 각 funtion들의 내용은 테스트를 어느정도 해놓고 이와 같은 스크립트를 작성하는 것이 좋습니다. 해당 스크립트가 오류를 일으키거나 방화벽으로 인해서 차단되는 경우가 발생하지 않도록 주의하기 위해서 테스트는 반드시 필요합니다.

만일 start function에 룰셋을 넣는 것이 번거롭다면 iptables-save 를 통해서 iptables 데이터 파일을 만들고 /etc/sysconfig/iptables 에 복사하여 아래와 같이 불러들이는 것도 좋습니다.

```
#!/bin/sh

IPT=/sbin/iptables
__fw_stop() {
    $IPT -P INPUT ACCEPT
    $IPT -P FORWARD ACCEPT
    $IPT -P OUTPUT ACCEPT

    $IPT -F
    $IPT -X
    $IPT -Z
}

__fw_start() {
    . /etc/sysconfig/iptables
}

case "$1" in
    "start")
        __fw_start
        ;;
    "stop")
        __fw_stop
        ;;
        *)
        echo "Usage: $0 {start|stop}"
esac
exit 0
```

여기서 iptables의 활용에 관한 내용을 마치도록 하겠습니다.

다음차에는 모듈에 대한 사용법을 올리도록 하겠습니다.

감사합니다.