
Firewall – konfiguracja *iptables*

Jeżeli udało się nam już skonfigurować kartę sieciową, i kiedy już mamy połączenie z siecią lokalną
lub z Internetem czas najwyższy, aby się trochę ukryć przed innymi lub jak kto woli zabezpieczyć.
Potrzebny do tego nam będzie program **iptables** następca popularnego *ipchains.*
Jest on na standardowym wyposażeniu Linuksów z jądrem 2.4 i wyższych.
W jądrze przechodzimy do gałęzi:

Device Drivers --->
Networking support --->
Networking options --->
Network packet filtering (replaces ipchains) --->
IP: Netfilter Configuration --->

i zaznaczamy interesujące nas opcje.
Instrukcję kompilacji jądra znajdziecie w rozdziale 4.
Niestety PLD w wersji *1.0 ra* nie posiada pakietów iptables.
Musimy zainstalować je ze źródeł.
[Przysporzyło nam to nie lada kłopotu zanim znaleźliśmy dobre rozwiązanie]

Ściągamy źródełka i przenosimy je do katalogu /usr/src

```
# cp iptables-1.2.11.tar.bz2 /usr/src
```

Rozpakowujemy:

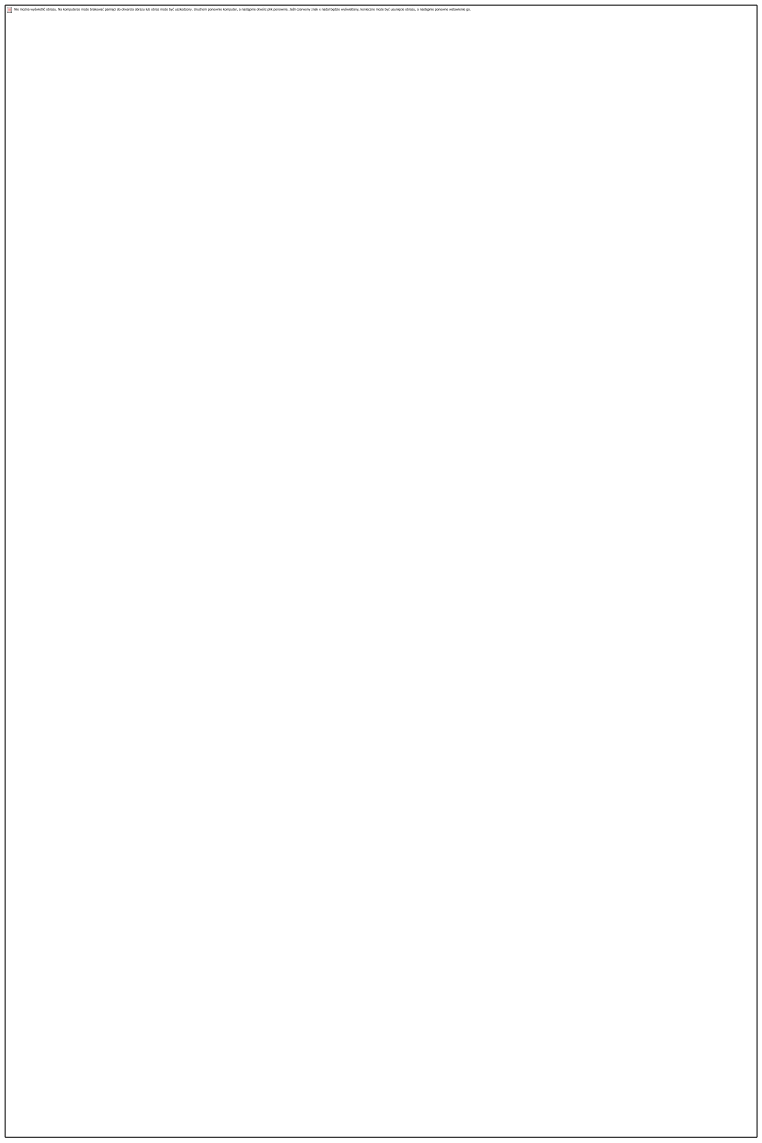
```
# tar –xjvf iptables-1.2.11.tar.bz2
```

Czas na kompilacje i instalacje:

```
# cd iptables-1.2.11
# make KERNEL_DIR=/usr/src/linux-2.4.26
# make install KERNEL_DIR=/usr/src/linux-2.4.26
```

Teraz mamy nowego iptables który znajduje się w /usr/local/sbin/iptables
Jeżeli działa nasz upragniony program możemy zabrać się do pracy.
[Więcej szczegółów o instalacji iptables dział 6]

Jak rozchodzą się pakiety ??



Jądro rozpoczyna pracę z trzema predefiniowanymi listami reguł w tabeli filtrującej.
Są to łańcuchy:

- **INPUT (wejściowy),**
- **OUTPUT (wyjściowy)**
- **FORWARD (przekazujący).**

Każdy pakiet docierający do hosta jest sprawdzany pod kątem miejsca przeznaczenia.
Na tej podstawie kernel decyduje, czy ma zostać przekazany do sieci położonej gdzieś dalej czy skierowany do niego samego.
Pakiet skierowany do tego komputera pozostaje "przetrawiony" przez reguły łańcucha **INPUT**.
Jeżeli jego obecność zostanie tu zaakceptowana, pakiet będzie dopuszczony do procesu, do którego został skierowany.
W przeciwnym wypadku zostanie odrzucony.
Jeżeli posiadasz włączone przekazywanie pakietów i pakiet jest przeznaczony dla innego interfejsu sieciowego,
pakiet przechodzi przez zestaw reguł łańcucha **FORWARD**. Reguły łańcucha zadecydują, czy może zostać przesłany dalej czy zostać odrzucony.

Procesu uruchamiane na naszym hoście także mogą być źródłem pakietów wydostających się do Internetu.
Takie pakiety przechodzą przez łańcuch **OUTPUT**. Po akceptacji docierają do interfejsu sieciowego.

Podstawowe operacje na łańcuchach.

-P - zmiana zasady dla wbudowanego łańcucha
-L - listowanie reguł w łańcuchu
-F - wyczyszczenie reguł z łańcucha
-A - dodanie nowej reguły do łańcucha
-I - wstawienie reguły do łańcucha na określoną pozycję
-R - wymiana reguły na określonej pozycji
-D - skasowanie reguły
-X - skasowanie pustego łańcucha
-Z - zerowanie liczników w łańcuchu

Opcje filtrowania.

-p -użycie reguły dla konkretnego protokołu
-s -określenie adresu źródłowego pakietu
-d -określenie adresu docelowego pakietu
-i -określenie interfejsu sieciowego
--sport -określenie portu źródłowego
--dport -określenie portu docelowego

-j - określa co należy wykonać z pakietem pasującym do reguły

Możliwe stany dla **–j** to:

DENY
ACCEPT
DROP
RETURN

[Uwaga: Wielkość liter ma znaczenie!!!]

Oczywiście to nie są wszystkie reguły – patrz man iptables ☺
Iptables, w porównaniu ze Ipchains posiada rozbudowane możliwości kontroli bitów pola KOD z nagłówka TCP.
O ile Ipchains wykorzystywał tylko kontrolę bitu SYN, to Iptables może kontrolować zawartość wszystkich sześciu bitów tj.:

URG, SYN, PSH, FIN, ACK, RST.

Numery portów z jakich korzystają poszczególne protokoły możemy odnaleźć w pliku **/etc/services**
Najpopularniejsze z nich to:

port	protokół	aplikacja

7	tcp/udp	echo
20	tcp/udp	ftp-data
21	tcp/udp	ftp
22	tcp/udp	ssh
23	tcp/udp	telnet
25	tcp	smtp
53	tcp/udp	DNS
80	tcp/udp	WWW
110	tcp/udp	pop3
113	tcp	auth.
119	tcp	NNTP
443	tcp/udp	SSL

W **iptables** mamy jeszcze bardzo ważną opcję:

–m state

Umożliwia ona analizę śledzenie połączeń **ip_conntrack** i **ip_conntrack_ftp.**
Stany, które można sprawdzać to:

NEW (NOWY) - pakiet tworzący nowe połączenie;
ESTABLISHED (NAWIĄZUJĄCY) - pakiet należący do istniejącego połączenia;
RELATED (ZWIĄZANY) - pakiet związany z połączeniem już ustanowionym, ale nie będący jego częścią;
INVALID (BŁĘDNY) - pakiet błędny lub nie do zidentyfikowania

Aby to wyjaśnić posłużymy się konkretnym przykładem. Chcemy, aby nasz *firewall* udostępniał innym serwer ftp.
Najprościej zrobić to tak:

iptables –A INPUT –p tcp --dport 20:21 –j ACCEPT

Jednak w ten sposób dostęp mają użytkownicy pracujący w trybie *passive off* .
Korzystając z flagi –m możemy udostępnić ftp dla użytkowników pasywnych.

iptables –A INPUT -m state --state ESTABLISHED, RELATED –j ACCEPT

Nie sposób wymienić wszystkich opcji **iptables** i możliwości ustawień.
Każdy administrator musi dopasować firewalla do swoich potrzeb.

Tworzymy naszą „ścianę ognia”!

Zakładamy, że nasz **firewall** pracuje na komputerze w sieci lokalnej, który ma dostęp do Internetu.
Otwieramy edytor tekstu i tworzymy plik np. o nazwie firewall_start:

```
#!/bin/sh
##########################################################################
#######
```

```
# Nazwa i lokalizacja programu
IPTABLES=iptables

# Sciezka do pliku wykonywlanego
PATH="/usr/sbin"

# Adres serwera - naszego kompa

SERWER="192.168.0.3"

# Adres komputera z ktorego logujemy sie jako admin :)

KOMP="192.168.0.10"

# Przestrzen adresowa naszej sieci wewnetrzenej i karta ja obslugujaca

WEW_NET="192.168.0.1/28"
WEW_DEV="eth0"

# Adres wyjsciowy - zewnetrzny i karta obslugujaca

ZEW_NET="0/0"   # 0/0 wskazuje jakikolwiek adres IP
ZEW_DEV="eth1"

# Uslugi TCP,ktore chcemy przepuszczac - "" puste oznacza wszystkie porty
# kolejne oddzielamy przecinkami

TCP_IN="www,ssh,ftp-data,ftp"           # 80,22,20,21
TCP_OUT="www,ftp,ftp-data,irc,ssh"      # 80,21,20, ,22


# Uslugi UDP,ktore przepuszczamy - "" puste oznacza wszytskie porty
# kolejne oddzielone przecinkami

UDP_IN="443,465,995"
UDP_OUT=""

# Uslugi ICMP, ktore chcemy przepuszczac - "" puste oznacza wszystkie typy
# kolejne oddzielone przecinkami

ICMP_IN=""
ICMP_OUT=""

##########################################################################
##########

# Usuwamy poprzednie reguly

$IPTABLES -F INPUT
```

```
$IPTABLES -F FORWARD
$IPTABLES -F OUTPUT

# Ustawienie domyslnej polityki

$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT ACCEPT  # akceptujemy wszystko co od nas wychodzi
$IPTABLES -P FORWARD DROP


# Zapisujemy caly nasz ruch w logach

$IPTABLES -A INPUT -j LOG -m limit --limit 15/hor            # 15 logów na godzine
$IPTABLES -A OUTPUT -j LOG -m limit --limit 15/hour
$IPTABLES -A FORWARD -j LOG -m limit --limit 15/hour

# Ladujemy mozliwosc sledzenia polaczen

modprobe ip_conntarck
modprobe ip_conntarck_ftp

# Wylaczamy odpowiedzi na pingi

   #echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_all

# Ochrona przed atakami typu Smurf

echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

# Wlaczamy ochrone przed komunikacja ICMP error

echo "1" > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses

# Wlacza logowanie dziwnych pakietow (spoofed. source routed. redirects)

echo "1" > /proc/sys/net/ipv4/conf/all/log_martians

# Nie akceptujemy datagramu IP z opcja "source route"

echo "0" > /proc/sys/net/ipv4/conf/all/accept_source_route

# Nie przyjmujemy pakietow ICMP redict, ktore moga zmienic nasza tablice routingu

echo "0" /proc/sys/net/ipv4/conf/all/accept_redirects

# Wszystkie karty nie beda przyjmowaly pakietow z sieici innych niz te
# z tablicy routingu

echo "1" /proc/sys/net/ipv4/conf/all/rp_filter
```

```
# Pozwalamy pakietom biegac po naszym komputerze
# czyli odblokowujemy petle zwrotna LOOPBACK

$IPTABLES -A INPUT -i lo -j ACCEPT
$IPTABLES -A OUTPUT -o lo -j ACCEPT

# Pozwalamy na korzstanie z protokolow w trybie passive on

$IPTABLES -A INPUT -m state --state ESTABLISHED, RELATED -j ACCEPT

# Pozwalamy na pingowanie nas i innych (opcje te zadzialaja jezli nie wylaczymy
# pingowania) patrz wyzej

   #$IPTABLES -A INPUT -p icmp -s 0/0 -d 0/0 -j ACCEPT
   #$IPTABLES -A OUTPUT -p icmp -s 0/0 -d 0/0 -j ACCEPT

$IPTABLES -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT # tylko my
mozemy pingowac

# Odblokowujemy uslugi na serwerze dla innych przychodzacych z zewnatrz

  #$IPTABLES -A INPUT -p tcp -s 0/0 --dport 20:22 -j ACCEPT # czyli ftp-
data,ftp,ssh,
  #$IPTABLES -A INPUT -p tcp  -s 0/0 --dport 80 -j ACCEPT # czyli www
  #$IPTABLES -A INPUT -p udp -s 0/0 --dport 20:22 -j ACCEPT # czyli ftp-
data,ftp,ssh,

$IPTABLES -A INPUT -p tcp -s 0/0 --sport 20:22 -j ACCEPT # czyli ftp-data,ftp,ssh,
$IPTABLES -A INPUT -p tcp  -s 0/0 --sport 80 -j ACCEPT # czyli www
$IPTABLES -A INPUT -p udp -s 0/0 --sport 20:22 -j ACCEPT # czyli ftp-data,ftp,ssh,

$IPTABLES -A INPUT -p tcp -s 0/0 --dport 113 -j ACCEPT # identyfikacja

$IPTABLES -A INPUT -p udp -s 0/0 --dport 1025:1100 -j ACCEPT # dla hotha

# Odblokowujemy uslugi na sewerze dla danego adresu IP - KOMP patrz wyzej
definicje TCP_IN, UDP_IN

   #$IPTABLES -A INPUT -p tcp -s $KOMP -m multiport --dport $TCP_IN  -j
ACCEPT # protokol tcp
   #$IPTABLES -A INPUT -p udp -s $KOMP -m multiport --dport $UDP_IN -j
ACCEPT # protokol udp

   #$IPTABLES -A INPUT -p udp -s $KOMP --dport 137:139 -j ACCEPT # protokol
udp

# Zezwalamy na wszystko z danego adresu IP – administracja z tego adresu ☺
```

```
$IPTABLES -A INPUT -s $KOMP -j ACCEPT  # dlatego powyższe reguly sa
wylaczone


# dostep do DNS

$IPTABLES -A INPUT -p tcp -s 0/0 --sport 53 -d $SERWER  -j ACCEPT
$IPTABLES -A INPUT -p udp -s 0/0 --sport 53 -d $SERWER -j ACCEPT

   #$IPTABLES -A OUTPUT -p tcp -s $SERWER -d 0/0 --dport 53 -j ACCEPT
   #$IPTABLES -A OUTPUT -p udp -s $SERWER -d 0/0 --dport 53 -j ACCEPT

   #mozna i tak

$IPTABLES –A INPUT –s 213.173.209.70 –j ACCEPT  # DNSy tele2 ☺
$IPTABLES –A INPUT –s 213.173.209.71–j ACCEPT

# Teraz troche się pozamykamy ☺
# Pakiety z adresem zrodlowym ustawionym na nasz

$IPTABLES -A INPUT -i $WEW_DEV -s $SERWER -j DROP            # atak Land

# Pakiety z adresow nierutowlanych, multicast i zarezerwowanych

$IPTABLES -A INPUT -i $WEW_DEV -s 10.0.0.0/8 -j DROP          #class A
$IPTABLES -A INPUT -i $WEW_DEV -s 172.16.0.0/12 -j DROP       #class B
  #$IPTABLES -A INPUT -i $WEW_DEV -s 192.168.0.0/16 -j DROP      #class C - z
tego korzystamy
$IPTABLES -A INPUT -i $WEW_DEV -s 224.0.0.0/4 -j DROP         #multicast
$IPTABLES -A INPUT -i $WEW_DEV -d 224.0.0.0/4 -j DROP         #multicast
$IPTABLES -A INPUT -i $WEW_DEV -s 240.0.0.0/5 -j DROP         #reserved
$IPTABLES -A INPUT -i $WEW_DEV -s 127.0.0.0/5 -j DROP         #lo
```

Jak już mamy napisane reguły to musimy zmienić prawa dostępu naszego pliku:

 # chmod 700 firewall_start

Teraz wystarczy go wywołać:

 # ./firewall_start

W tym momencie nasz firewall działa. Dla pewności możemy *pingować* nasz serwer z innego komputera.
Jeżeli nie będzie odpowiedzi na *pingi* znaczy się że wszystko jest ok.
Administrując danym serwerem, będziemy zmuszeni ciągle modyfikować nasze reguły na potrzeby sieci.
To jest przykład tylko podstawowej konfiguracji *ściany ogniowej*.

Dobrze by było napisać skrypt, który wyczyści nasze reguły i wyłączy filtrowanie.
O to przykład „zatrzymania” firewalla:

```
#!/bin/sh

#########################################################################
#

# Nazwa i lokalizacja programu

IPTABLES=iptables

#########################################################################
#

# Czyszczenie poprzednich wpisow (INPUT, OUTPUT, FORWARD)

$IPTABLES -F

# Ustawienie domyslnej polityki

$IPTABLES -A INPUT -j ACCEPT
$IPTABLES -A OUTPUT -j ACCEPT
$IPTABLES -A FORWARD -j ACCEPT

# Wlaczamy odpowiedzi na pingi

echo "0" > /proc/sys/net/ipv4/icmp_echo_ignore_all

# Ochrona przed atakami typu Smurf

echo "0" > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

# Wylaczamy ochrone przed komunikacja ICMP error

echo "0" > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses

# Wylacza logowanie dziwnych pakietow (spoofed. source routed. redirects)

echo "0" > /proc/sys/net/ipv4/conf/all/log_martians

# Akceptujemy datagramu IP z opcja "source route"

echo "1" > /proc/sys/net/ipv4/conf/all/accept_source_route

# Przyjmujemy pakiety ICMP redict, ktore moga zmienic nasza tablice routingu

echo "1" /proc/sys/net/ipv4/conf/all/accept_redirects

# Wszystkie karty beda przyjmowaly pakiety z innych sieci

echo "0" /proc/sys/net/ipv4/conf/all/rp_filter
```

Zapiszemy go pod nazwa **firewall_stop.**
Zmieniamy prawa dostępu:

```
# chmod 700 firewall_stop
```

i wyłączamy zaporę

```
# ./firewall_stop
```

Aby nasz skrypt był wykonywany przy każdym uruchomieniu systemu dodajemy wpis do pliku:

/etc/rc.d/rc.local

Wywołując polecenie:

iptables –L –n

otrzymamy listę uruchomionych reguł.

Logi z firewalla.

Po uruchomieniu skryptu firewall_start przydało by się, aby informacje o działaniu zapory były przechowywane w logach. W tym celu wpisujemy na koniec pliku **/etc/syslog.conf** następujący kod:

```
*.*      /dev/tty12
*.*       /var/log/wszystko
```

Restartujemy demona syslogd:

```
# killall  -HUP syslogd
```

Od tego momentu w pliku /var/log/wszystko (także na konsoli 12 – alt+12) będziemy mieli wszystkie logi systemowe.
Jak włączyć 12 konsolę opiszemy w kolejnym rozdziale.

Jeżeli korzystamy z demona **syslog-ng** sprawa jest jeszcze prostsza, gdyż możemy skierować logi z firewalla do
osobnego pliku. Edytujemy plik **/etc/syslog-ng/syslog-ng** i szukamy reguły *iptables:*

destination iptables   { file("/var/log/iptables"); file("/dev/tty12"); };

zmieniamy ją tak jak powyżej. Aby nasze zmiany doszły do skutku wywołujemy polecenie:

syslog-ng –f /etc/syslog-ng/syslog-ng

Od tej pory wszystkie logi z firewalla będziemy mieli w pliku /**var/log/iptables** i na 12 konsoli.

Należy uważać, aby nasze logi nam nie przepełniły partycji, czyli należy poprawnie skonfigurować
plik **/etc/logrotate.conf.**

Dla zainteresowanych podaję linki z naszymi plikami:

- firewall_start
- firewall_stop

0.SPIS TRESCI | **8. DODATKOWE KONSOLE**