FreeBSD jako trasownik i firewall - sprawozdanie

Mikołaj Radzewicz

9 maja 2005

Spis treści

1 Wstęp

Zadanie miało na celu sprawdzić funkcjonowanie i przydatność komputera pracujące pod systemem FreeBSD w roli trasownika i firewall'a sieciowego. Rozwiązanie ma funkcjonować w małej sieci domowej, gdy dostawca udostępnia tylko jeden numer IP dostępu do Internetu. Rozwiązanie oparłem na IPFilter i natd.

2 Sieć

Komputer, pełniący funkcję bramy, pracuje pod systemem FreeBSD w wersji 5.3-BETA7, natomiast na stacji roboczej podłączona do sieci LAN zainstalowany był Windows 98. Komputer brama posiadał dwa interfejsy sieciowe (*dc0* jako interfejs zewnętrzny i *vr0* wewnętrzny dla lokalnej sieci) z przydzielonym przez serwer DHCP adresem IP 62.179.47.54 dla interfejsu *dc0* i statycznie przydzielonym adresem IP 10.0.0.1 i masce 24 bitowej dla interfejsu *vr0*.

3 Konfiguracja trasownika i firewall'a

Konfiguracje komputera jako trasownik można było przeprowadzić na dwa sposoby:

- dokonując odpowiednie wpisy w pliku */etc/rc.conf*, co wymuszałoby odpowiednie zmiany przy podnoszeniu systemu,
- wprowadzać odpowiednie komendy z pod konsoli.

Diskless w laboratorium komputerowym spowodował, że wybrałem drugie rozwiązanie - skrypt konfigurujący w odpowiedni sposób maszynę. Poniżej zamieszczam zawartość skryptu (*nat*).

```
#!/bin/sh
#nazwa pliku z regulami firewall'a
regfirewall="ipf.rules"
#nazwa pliku z regulami nat'a
regnat="ipnat.rules"
#sciezka polozenia plikow z regulami
#setenv gdzie $1
export gdzie=$1
ifzew="dc0"
ipzew="62.179.47.54"
#dany host dziala jako router-wlaczenie przekazywania pakietow
sysctl net.inet.ip.forwarding=1
if [ -w $gdzie  ]
```

```
then
touch $gdzie/$regfirewall


(...)

touch $gdzie/$regnat
echo -e "map $ifzew 10.0.0.0/24 -> $ipzew/32" >$gdzie/$regnat
else
echo "Nie masz prawa do pisania w katalogu $gdzie" 1>&2
fi
#uruchomienie kolejno firewall'a i demona odpowiedzialnego za translacje adresow
/etc/rc.d/ipfilter forcestart
/etc/rc.d/natd forcestart
#usuwa wszystkie aktualne reguly i tabele mapowan NAT'a i zaczytuje odpowiednie
reguly z pliku
ipnat -CF -f $gdzie/$regnat
#umozliwia logowanie pakietow z /dev/ipl - patrzy na syslogd
ipmon -s &
```

Skrypt przeładowujący ustawienia zamieszczam poniżej (*reset*):

```
#!/bin/sh
regfirewall="ipf.rules"
#Wyczyszczenie aktywnej list regul i ustawienie jako nowej z pliku
ipf -AF -f $1/$regfirewall
/etc/rc.d/ipfilter forcerestart
```

3.1 Reguły filtrujące i ich krótkie omówienie.

1. block in on $ifzew
2. block out on $ifzew
3. block in quick on $ifzew from 192.168.0.0/16 to any
4. block in quick on $ifzew from 172.16.0.0/12 to any
5. block in quick on $ifzew from 10.0.0.0/8 to any
6. block in quick on $ifzew from 127.0.0.0/8 to any
7. block in quick on $ifzew from 0.0.0.0/8 to any
8. block in quick on $ifzew from 169.254.0.0/16 to any
9. block in quick on $ifzew from 192.0.2.0/24 to any

10. block in quick on $ifzew from 204.152.64.0/23 to any
11. block in quick on $ifzew from 224.0.0.0/3 to any
12. #block in log quick on $ifzew proto icmp from any to $ipzew
13. block in log quick on $ifzew fastroute proto udp from any to any port 33434 >< 33465
14. pass in quick on $ifzew proto icmp all icmp-type 0
15. pass out quick on $ifzew proto icmp all icmp-type 0
16. pass in quick on $ifzew proto icmp all icmp-type 3
17. pass out quick on $ifzew proto icmp all icmp-type 3
18. pass in log quick on $ifzew proto icmp all icmp-type 8
19. pass out quick on $ifzew proto icmp all icmp-type 8
20. pass in quick on $ifzew proto icmp all icmp-type 11
21. pass out quick on $ifzew proto icmp all icmp-type 11
22. pass in quick on $ifzew proto tcp from 194.29.146.3 to any port = 22 flags S keep state
23. pass in quick on $ifzew proto tcp from any port > 1023 to $ipzew port = 80 flags S keep state
24. pass out quick on $ifzew proto tcp from $ifzew port > 1023 to any port = 80
25. pass in quick on $ifzew proto tcp from any port = 80 to $ipzew port > 1023 flags A/A
26. block return-rst in log first on $ifzew proto tcp from any to any port = 22
27. #block return-icmp-as-dest(port-unr) in log on $ifzew proto udp from any to any port = 22
28. pass out quick on $ifzew proto tcp/udp all keep state keep frags

Reguły: 3-5 to pula adresów prywatnych IP, 6-to klasa adresów używana przez loopback przez program na lokalnej maszynie dlatego, powinna też być blokowama, 7-traktowana jest przez stos IP na różne sposoby, dlatego powinna być blokowana, 8-używana jest do auto-konfiguracji systemy, gdy

jeszcze nie otrzymał on adresu z DHCP, 11-wycięta klasa D i E sieci, używana głównie do ruchu rozgłoszeniowego, 13-powoduje, że nasz komputer nie jest widoczny jako trasownik dla traceroute'a, 13-trzeba przepuszczać icmp (przekroczenie terminu), aby działał traceroute, 14-21 zapewnia działanie pinga i traceroute'a w obie strony, 22-pozwala na zdalne łączenie się z moim komputerem przez ssh, jeżeli pakiet nawiązujący połączenie pochodzi z volta, 23-25 umożliwiają funkcjonowanie serwerowi Apache, 26-zwraca komunikat reset do komputera, który powoduje połączyć się zdalnie z moim systemem, jeżeli nie "jest" to "volt".

4 Testy

Testy przeprowadzałem na bieżąco - wprowadzając kolejne reguły i sprawdzając ich skuteczność. Niezwykle przydatne okazały się takie narzędzia i aplikacje jak *ping, traceroute, tracert, telnet* czy *ssh* do testowania skuteczności reguł. Firewall'a testowałem zarówno z wewnętrznej sieci LAN, jak i Internetu.

5 Podsumowanie

Przy konfiguracji komputera mającego pełnić funkcję bramy i firewall'a korzystałem z Handbook'a FreeBSD (rozdziały 24 i 25) (*www.freebsd.org*), pomocy systemu FreeBSD, a także dokumentacji samego autora IPFilter'a na stronie *http://coombs.anu.edu.au/~avalon/ip-filter.html*. Niezwykle przydatna okazała się również książka "Internet Firewalls. Tworzenie zapór ogniowych" autorstwa Zwicky'ego i Simon'a. Należy pamiętać, że logowane komunikaty są standardowo zapisywane w pliku */var/log/security*, jednak plik ten może być zmieniony po wprowadzeniu stosownej modyfikacji w pliku */etc/syslog.conf*. FreeBSD dobrze sobie radzi w roli trasownika i firewall'a dla małej sieci LAN (tylko na takiej mogłem sprawdzić), chociaż napotkałem pewne problemy przy wpisaniu jednej z reguł. System zawieszał się kilkakrotnie przy użyciu opcji *fastroute* dla wszystkich przychodzących pakietów TCP.
Podana konfiguracja funkcjonuje do dnia dzisiejszego u mnie w domu nie sprawiając żadnych problemów. Różnorodność opcji sprawia, iż każdy użytkownik po zapoznaniu się z dokumentacja może bez przeszkód uzyskać odpowiednie, pasujące mu rozwiązanie u siebie na komputerze. Proces konfiguracji nie nastręcza większych problemów jednak jest czasochłonny (przeprowadzanie testów) i wymaga od użytkownika pewnej wiedzy z zakresu sieci komputerowych oraz uwagi przy wprowadzaniu reguł (kolejność ich jest bardzo istotna! - czasami chcąc zablokować nieumiejętnie jedne pakiety jakąś regułą, możemy je kolejną regułą przepuścić).