

IDŹ DO

PRZYKŁADOWY ROZDZIAŁ



SPIS TREŚCI

KATALOG KSIĄŻEK

KATALOG ONLINE

ZAMÓW DRUKOWANY KATALOG

TWÓJ KOSZYK

DODAJ DO KOSZYKA

CENNIK I INFORMACJE

ZAMÓW INFORMACJE
O NOWOŚCIACH

ZAMÓW CENNIK

CZYTELNIA

FRAGMENTY KSIĄŻEK ONLINE

Sieci komputerowe. Kompedium. Wydanie II

Autor: Karol Krysiak
ISBN: 83-7361-995-X
Format: A5, stron: 568



Kompletne omówienie zagadnień sieci komputerowych

- Topologie i nośniki
- Sieci bezprzewodowe
- Usługi sieciowe i protokoły
- Administrowanie siecią
- Bezpieczeństwo w sieciach

Sieci komputerowe to temat niezwykle rozległy i omawiany w dziesiątkach publikacji. Jedne traktują o protokołach, inne o usługach, jeszcze inne o administrowaniu siecią, przy czym wiele z nich zawiera ogromną ilość informacji teoretycznych często po prostu zbędnych w codziennej pracy. Typowy administrator sieci powinien posiadać kompleksową wiedzę praktyczną z każdej z tych dziedzin, aby sprawnie rozwiązywać pojawiające się problemy. Dla osoby zajmującej się sieciami komputerowymi najlepszym podręcznikiem jest taki, w którym zgromadzone są wszystkie niezbędne wiadomości praktyczne.

Książka „Sieci komputerowe. Kompedium. Wydanie II” to taki właśnie podręcznik. Drugie już wydanie tego bestsellera zawiera wszystkie informacje pozwalające na szybkie przygotowanie się do pełnienia obowiązków administratora sieci. Jest tu dawka wiadomości teoretycznych niezbędnych do zrozumienia zasad funkcjonowania sieci komputerowych, ale główną część książki stanowi opis mechanizmów sieciowych, sposobów ich konfigurowania i reagowania na występujące awarie. Książka ta może również pełnić rolę podręcznego źródła wiedzy na temat parametrów sieci.

- Topologie sieci komputerowych
- Model ISO/OSI i podział na warstwy
- Standard Ethernet
- Zasada działania i projektowanie sieci bezprzewodowych
- Protokół IP
- Usługi DNS, poczta elektroniczna, FTP i HTTP
- Szyfrowanie danych
- Administrowanie siecią LAN
- Zabezpieczanie sieci przed wirusami i włamaniami
- Wykrywanie ataków na sieć
- Stosowanie firewalli

**Jeśli zajmujesz się administrowaniem siecią,
ta książka powinna znaleźć się w Twojej bibliotece**



Spis treści

Wstęp	13
Rozdział 1. Sieci komputerowe	25
1.1. Podział sieci komputerowych w zależności od rozmiaru	25
1.2. Topologie sieci komputerowych	26
Topologia sieci	26
Topologia fizyczna	26
Topologia logiczna	27
1.3. Model ISO/OSI	28
1.4. Model protokołu TCP/IP	30
Rozdział 2. Rodzaje nośników	35
2.1. Najważniejsze technologie	35
2.2. Przewód koncentryczny	38
Zastosowania sieci 10Base-2	40
2.3. Skrętka UTP	41
Wymagania dla instalacji spełniającej założenia CAT-5	47
Cat-5e — parametry	48
2.4. Światłowód	52
Budowa światłowodu	53
Zasada działania światłowodu	53
Światłowód wielomodowy	53
Światłowód jednomodowy	54
Złącza światłowodowe	55
Standardy transmisji światłowodowych	57
2.5. Okablowanie strukturalne	59
Definicje	59
Normy	60
Projekt i definicje	61
Zalecenia	65
Telefonia	66
Serwerownia	68
Rozdział 3. Warstwa dostępu do sieci — standard Ethernet	73
3.1. Historia	73
3.2. Działanie protokołu	74
Metody transmisji	74
Norma IEEE 802.3	74
Wydajność sieci Ethernet 10 Mb/s	77
3.3. Budowa ramki Ethernet	78
Protokół LLC	80

3.4. Zasady konstruowania sieci Ethernet	82
Reguły dla Ethernetu (10 Mb/s)	82
Reguły dla Fast Ethernetu (100 Mb/s)	86
Reguły dla Gigabit Ethernetu (1 000 Mb/s)	87
3.5. Technologie	88
Full-duplex	88
MAC Control	89
Automatyczne negocjowanie parametrów łącza	89
1000Base-T	90
VLAN — IEEE 802.1q	91
QoS — 802.1p	94
STP (Spanning Tree Protocol)	95
Power over Ethernet	101
Sygnały i kodowanie	102
3.6. Protokół ARP — protokół określania adresów	104
Proxy-ARP	105
Reverse-ARP	105
Zapobieganie zdublowaniu adresów IP	106
Pakiet protokołu ARP	106
Polecenia do manipulacji tablicą ARP	108
3.7. Urządzenia sieciowe działające w warstwie dostępu do sieci	110
Karta sieciowa	110
Modem	110
Transceiver	111
Konwerter nośników	111
Regenerator (repeater)	112
Koncentrator (hub)	112
Most (bridge)	117
Przełącznik (switch)	117

Rozdział 4. Sieci bezprzewodowe	123
802.11	124
802.11b	125
Kanały (channel)	125
802.11a	126
802.11g	126
802.11n	127
Standaryzacja	127
4.1. Struktura sieci WLAN	128
4.2. Kontrola dostępu do medium	130
Mechanizmy transmisji	131
4.3. Ramka 802.11	133
4.4. Działanie protokołu 802.11	138
Skanowanie	138
Przyłączanie	139
Uwierzytelnianie	139
Kojarzenie (powiązanie)	140
Roaming — proces kojarzenia ponownego	141
Oszczędzanie energii	142
Transmisja radiowa	142
4.5. WEP i bezpieczeństwo	143
Problemy z WEP	144
Nowe protokoły bezpieczeństwa WLAN	146

4.6. Pozostałe projekty WLAN	148
4.7. Projektowanie sieci radiowych	149
Rodzaje anten	151
Obliczenia	151
Zalecenia	156
4.8. Konfigurowanie urządzeń	157
Konfiguracja sieciowej karty bezprzewodowej	157
Punkt dostępowy (Access Point)	161
4.9. Kismet	165
4.10. WiMax — 802.16	166
Rozdział 5. Warstwa Internetu	169
5.1. Protokół IP	169
Zadania spełniane przez protokół IP	170
Cechy protokołu IP	170
Budowa datagramu IP	170
5.2. Adresowanie IP	173
Klasy adresów w TCP/IP	177
Bezklasowe routowanie międzydomenowe (CIDR)	178
Adresy specjalne i klasy nieroutowalne	183
Nadawanie adresów IP interfejsowi sieciowemu	184
5.3. Routowanie datagramów IP	188
Tablica routingu	191
Polecenia służące do manipulacji tablicą routingu	193
Routing źródłowy	195
5.4. Protokół ICMP	196
Zadania protokołu ICMP	197
Format nagłówka ICMP	198
Pola Typ i Kod komunikatu ICMP	199
Polecenia wykorzystujące protokół ICMP	202
5.5. IPv6 — wersja szоста protokołu IP	205
Nagłówek IPv6	206
Adres IPv6	207
5.6. Protokoły routingu dynamicznego	208
Protokoły wektora odległości (Distance Vector)	208
Protokoły stanu łącza (Link State)	209
Jak zbudowany jest Internet?	209
5.7. Urządzenia pracujące w warstwie Internetu	214
Router	214
Rozdział 6. Warstwa transportowa	217
6.1. Port, gniazdo	217
6.2. Protokół UDP	219
6.3. Protokół TCP	220
tcpdump	223
netstat	228
Rozdział 7. Usługi warstwy aplikacji	233
7.1. DNS	233
Rejestrowanie własnej domeny	236
Ogólne informacje o serwerach DNS	238
Jak to w rzeczywistości działa?	239
Konfiguracja hosta	240
Rekordy zasobów	244

Serwery DNS	246
Konfiguracja serwera BIND	247
Sterowanie demonem named	255
Kwestie bezpieczeństwa	258
Format komunikatu DNS	262
Programy użytkowe — diagnostyka	264
7.2. SMTP	272
Serwery SMTP	272
Sprawdzanie działania serwera	275
Protokół MIME	282
Bezpieczeństwo	283
7.3. POP	284
Sprawdzanie działania serwera	285
Serwery POP	286
7.4. IMAP	287
Sprawdzanie działania serwera	287
Serwery IMAP	289
7.5. FTP	290
Tryby pracy FTP	290
Komunikacja z serwerem	292
Obsługa programu ftp	293
Serwery	295
Bezpieczeństwo	296
7.6. HTTP	297
Protokół HTTP	297
Sprawdzanie działania serwera HTTP	298
Serwery	299
Bezpieczeństwo	301
7.7. SSL	302
Certyfikaty	303
Uproszczona zasada działania SSL	303
Długość klucza	304
Pakiet stunnel	304
7.8. Telnet	305
7.9. SSH	306
7.10. Finger	307
7.11. Auth	308
7.12. NNTP	308
7.13. SNMP	309
Różnice pomiędzy wersjami SNMP	313
RMON	314
Bezpieczeństwo	316
Testowanie	316
7.14. IRC	318
7.15. Whois	318
7.16. NTP	321
7.17. Syslog	322
7.18. Bootps, DHCP	324
Nagłówki DHCP	325
Pola i ich opisy	325
Proces uzyskiwania konfiguracji	327
Konfiguracja klientów DHCP	328
Serwery DHCP	330

7.19. NetBIOS	334
Wyszukiwanie nazw NetBIOS	337
Optymalizacja	338
Bezpieczeństwo	339
7.20. VoIP	339
SIP	339
H.323	343
7.21. Urządzenia sieciowe pracujące w warstwie aplikacji	344
Komputer	344
Serwer	345

Rozdział 8. Inne protokoły 347

8.1. Token Ring	347
8.2. FDDI	348
8.3. IPX/SPX	351
Budowa pakietu IPX	351
Adresy IPX	353
Protokoły używane w IPX	353
8.4. ISDN	354
8.5. PPP	357
Ramka PPP	358
Dodatkowe możliwości PPP	359
Konfiguracja PPP w Linuksie	359
8.6. xDSL	365
ADSL	366
RADSL	367
SDSL	368
HDSL	368
VDSL	368
IDSL	368
8.7. Frame Relay	368
Opis technologii Frame Relay	369
Zasada działania FR	370
Format ramki Frame Relay	371
Mechanizmy sieci FR	372
Parametry transmisji FR	374
8.8. ATM	375
Właściwości standardu ATM	376
Interfejsy ATM	376
Rodzaje połączeń w sieciach ATM	376
Komórka ATM	377
Usługi ATM	379
Model ATM	380
Klasy ruchu	381
Trasowanie ATM	383
Dodatkowe możliwości sieci ATM	383
8.9. Sieci w gniazdku zasilającym — PLC	384
Topologia sieci PLC	384
Standardy PLC	385
Wady PLC	386
8.10. Sieci telewizji kablowych	387
Standard MCSN/DOCSIS	388

Rozdział 9. Administracja siecią LAN	389
9.1. Projektowanie sieci LAN	390
Struktura fizyczna sieci	390
Struktura logiczna sieci	393
9.2. Rozwiązywanie problemów	396
Poważna awaria	396
Użytkownik	396
Rady	397
Problemy	398
9.3. Narzędzia administratora sieci	401
Sniffery	402
Analizatory sieci	409
Testowanie dostępności usług	418
Skanery bezpieczeństwa	422
Inne narzędzia	429
9.4. Wykorzystanie protokołu SNMP	434
Konfiguracja agenta snmpd	435
Konfiguracja menedżera MRTG	437
9.5. Zarządzalne urządzenia aktywne	441
Rozdział 10. Bezpieczeństwo	443
10.1. Polityka bezpieczeństwa	445
10.2. Najważniejsze pojęcia	449
Firewall	449
NAT	451
Kryptografia	454
VPN	459
IDS	461
Wirusy	466
10.3. Konstrukcja sieci	467
10.4. Rozpoznanie terenu	474
Zbieranie danych	475
Skanowanie	477
Metody ukrywania skanowania	480
Identyfikacja systemu operacyjnego	484
10.5. Metody włamań	489
Uzyskanie dostępu	490
Destabilizacja pracy	497
10.6. Zagrożenia wewnętrzne	502
Wykrywanie snifferów	502
Sposoby omijania przełączników	504
Zasoby	507
10.7. Podsumowanie	508
Zachowanie podczas włamania	508
Rozdział 11. Firewall	511
11.1. Rodzaje firewalli	511
Tradycyjne proxy (Traditional proxies)	511
Przezroczyste proxy (Transparent proxies)	512
Tłumaczenie adresów IP (NAT)	512
Filtrowanie pakietów	512
11.2. Obsługa filtrowania pakietów w Linuksie	513
Rozwiązania komercyjne	515
Ipcchains — Linux 2.2	517

Składnia polecenia ipchains	518
Iptables — Linux 2.4	520
11.3. Tworzymy firewall	522
Podstawy	523
Konfiguracja	525
Logi systemowe	531
Problemy z działaniem firewala	532
Wyłączanie firewala	532
11.4. Dodatkowe funkcje	533
Ograniczenia na ICMP	533
Jak przepuścić nową usługę	534
Ustawianie priorytetów pakietów	535
Wykrywanie skanowania za pomocą firewala	537
Wykrywanie NAT	538

Rozdział 11.

Firewall

W tym rozdziale zajmę się sposobami kompleksowego zabezpieczenia sieci komputerowej za pomocą techniki **filtrowania datagramów** (*firewall*) oraz **tłumaczenia adresów sieciowych** NAT (*Network Address Translation*). Nie przedstawię gotowego rozwiązania, ale postaram się, abyś zrozumiał, w jaki sposób możesz stworzyć coś takiego dla swojej sieci. Uważam, że absurdem jest pobranie z sieci gotowego skryptu i uruchomienie go. Musisz rozumieć, co robi Twój firewall i jak funkcjonuje — lepiej, aby był prosty, ale żebyś poznał dokładnie jego działanie.

Każdy firewall jest inny, tak jak każda sieć jest inaczej zbudowana i różne są charaktery i poziomy wiedzy ich administratorów. Firewall rozwija się wraz z Tobą; wraz ze zwiększaniem się Twojej wiedzy o zagrożeniach i metodach przeciwdziałania im. Zmienia się też wraz z wykrywaniem coraz to nowych zagrożeń. To żywy organizm stanowiący pierwszą linię obrony dla Twojej sieci, poświęć mu więc trochę czasu i zaangażowania.

11.1. Rodzaje firewalli

Poniżej przedstawię bardziej dokładny podział firewalli niż wcześniej. Opiszę również ich cechy, na podstawie których jako administrator sieci będziesz wybierał odpowiedni rodzaj do potrzebnego Ci zastosowania. Oczywiście dobór konkretnego firewalla spośród dostępnych na rynku pozostawiam już Tobie, zwłaszcza że tryb wydawania książki jest długi, a zmiany możliwości produktów obecnych na rynku następują bardzo szybko.

Tradycyjne proxy (Traditional proxies)

Jest to rodzaj firewalla pośredniczącego; pakiety z sieci prywatnej nigdy nie wychodzą do Internetu i *vice versa*. Adresy IP w sieci prywatnej powinny być z klas nieroutowalnych. Jedynym sposobem połączenia się z Internetem jest wywołanie firewalla, ponieważ jest on jedyną maszyną mogącą łączyć się równocześnie z obiema sieciami

Uruchamiany jest na nim program zwany *proxy*, który tego dokonuje. Dla każdej usługi, która ma być dostępna z Internetu, na firewallu musi być uruchomiony osobny program pośredniczący w jej świadczeniu.

Komputery w sieci wewnętrznej muszą być specjalnie skonfigurowane w celu uzyskania dostępu do wybranych usług. Przykładowo aby ściągnąć stronę WWW, muszą połączyć się z firewallem na port 8080, zalogować się i zażądać potrzebnej strony. W tym momencie uruchamia się odpowiedni program pośredniczący, pobiera potrzebne dane i przekazuje do odpowiedniego komputera w sieci lokalnej.

Przezroczyste proxy (Transparent proxies)

Jest to drugi rodzaj firewalla pośredniczącego; pakiety z sieci prywatnej również nigdy nie wychodzą do Internetu i *vice versa*. Adresy IP w sieci prywatnej powinny być z klas nieroutowalnych. Jediną drogą połączenia się z Internetem jest wywołanie firewalla, ponieważ jest on jedyną maszyną, mogącą łączyć się równocześnie z obiema sieciami. Uruchamiany na nim program zwany *transparent proxy*, który tego dokonuje. System operacyjny zamiast dokonać routingu pakietów do Internetu, kieruje je do tego programu. Dla każdej usługi, która ma być dostępna z Internetu, na firewallu musi być uruchomiony osobny program pośredniczący w świadczeniu takiej usługi.

Przezroczyste proxy jest bardziej wygodne dla użytkowników i dla administratora. Klient nie musi wiedzieć o użyciu oprogramowania typu proxy i nie musi być specjalnie konfigurowany. Na przykład firewall jest skonfigurowany do przekierowywania (np. za pomocą polecenia *ipchains*) wszystkich połączeń do portu 80 na port 8080, na którym pracuje *transparent proxy*. Przy próbie pobrania dowolnej strony WWW transmisja przekierowywana jest na port 8080, a następnie wszystko odbywa się tak, jak w poprzednim przykładzie.

Tłumaczenie adresów IP (NAT)

Pakiety z sieci prywatnej nigdy nie wychodzą do Internetu bez specjalnej obróbki i *vice versa*. Adresy IP w sieci prywatnej powinny być z klas nieroutowalnych. W tym przypadku używamy specjalnych funkcji zmieniających niektóre pola transportowanych pakietów (adres źródłowy, port źródłowy). Zostało to już opisane.

Filtrowanie pakietów

W tym przypadku nasza sieć jest częścią Internetu, pakiety mogą poruszać się bez zmian poprzez obie sieci. Firewall na podstawie nagłówek protokołów podejmuje decyzję, czy przepuścić dany pakiet, czy nie. Filtrowanie pakietów zastosowano po to, aby ograniczyć dostęp z Internetu tylko do naszych wewnętrznych serwerów i unieemożliwić dostęp do komputerów użytkowników. Jeżeli firewall ma możliwość śledzenia sesji protokołów warstwy transportowej, nazywamy go **firewallem z inspekcją stanu** (*statefull inspection*); czasami spotyka się w literaturze również pojęcie firewalla obwodów.

Ten typ firewalla ma najmniejsze możliwości kontroli i autoryzacji dostępu, ale jest równocześnie najbardziej elastyczny i wprowadza najmniej ograniczeń dla użytkowników z sieci wewnętrznej. Nie ma potrzeby uruchamiania specjalnych programów pośredniczących.

11.2. Obsługa filtrowania pakietów w Linuksie

Linux jest typowym systemem sieciowym; możliwości filtrowania pakietów pojawiły się w nim bardzo wcześnie, bo już w wersji 1.1. Alan Cox po prostu w 1994 roku przeniósł z BSD kod używanego tam ipfw. Jos Vos rozbudował ten kod i w jądrach wersji 2.0 pojawiło się narzędzie ipfwadm do kontroli reguł filtrowania. W 1998 roku dla wersji 2.2 Paul Russell i Michael Neuling radykalnie zmienili kod jądra i wprowadzili nowe narzędzie ipchains. W 1999 roku dla jąder w wersji 2.4 i dalszych kod podsystemu sieciowego został przepisany od nowa, wprowadzono wiele przydatnych możliwości: zaawansowany routing (również w oparciu o adres źródła pakietu), zarządzanie pasmem oraz całkiem nową „ścianę ogniową”. Do zarządzania firewallem zostało przeznaczone nowe polecenie iptables. Aby go używać, musisz mieć włączony (wkompilowany) w jądrze moduł *netfilter* oraz zainstalowany pakiet *iptables*.

Pamiętaj, że pakiet *ipchains* nie jest już rozwijany, a do *iptables* ciągle pojawiają się nowe dodatki. Jest to główny powód, poza o wiele większą funkcjonalnością, dla którego skupię się na tym pakiecie (porównanie obu pakietów w tabeli 11.1). Jeśli brak jest funkcji, których potrzebujesz, spróbuj zajrzeć do zbioru dodatków o nazwie **Patch-o-Matic**; z listą najnowszych można zapoznać się pod adresem <http://www.netfilter.org/patch-o-matic/pom-extra.html>. Najciekawsze dodatki pozwalają na ograniczanie liczby połączeń z jednego IP, selekcjonowanie pakietów w zależności od opcji IP, określanie rozmiaru pakietu. Główną zaletą firewalla opartego na systemie Linux jest jego olbrzymia wręcz elastyczność, widoczna już w trakcie procesu projektowania. Pakiety komercyjne, nastawione najczęściej na schematyczne rozwiązania konstrukcji sieci, mają problemy w implementacji niektórych zadań, nie będących żadnym ograniczeniem dla iptables. Gdy dochodzi do wyboru wymaganej funkcjonalności, iptables zadziwia różnorodnością. Poniżej zamieszczam listę najciekawszych funkcji zawartych w wersji 1.2.11.

Niektóre wybrane moduły (-m state) w iptables:

- ♦ *addrtype* — weryfikacja typów adresów: *UNSPEC*, *UNICAST*, *LOCAL*, *BROADCAST*, *ANYCAST*, *MULTICAST*, *UNREACHABLE*,
- ♦ *connmark* — „oznaczanie” połączenia do późniejszego wykorzystania przez np. routing,
- ♦ *connrate* — określanie aktualnego transferu dla połączenia,
- ♦ *conntrack* — śledzenie stanu połączenia, do którego należy pakiet: *INVALID*, *NEW*, *ESTABLISHED*, *RELATED*, *SNAT*, *DNAT*,

Tabela 11.1. Porównanie funkcjonalności *ipchains* i *iptables*

	Ipchains	Iptables
Mozliwe kryteria selekcji pakietów	Adres źródłowy IP	
	Adres docelowy IP	
	Protokół (TCP, UDP) i port lub zakres portów	
	Protokół ICMP i komunikat	
	Zaistnienie fragmentacji	
	Wartości bitów TOS	
	Pakiety z flagą SYN	
	Możliwość określania reguł symetrycznych (odwrotnych) -y	
		Adres sprzętowy interfejsu MAC
		Dowolna kombinacja flag TCP
Funkcjonalność		Kryterium częstotliwości nadchodzących pakietów (zabezpieczenie przeciwko DoS)
		Właściciel pakietu w systemie
		Inspekcja stanu, śledzenie sesji dla protokołów TCP, UDP, ICMP
	Uproszczona translacja adresów, tzw. <i>Masquerading</i>	Pełna funkcjonalność NAT
	Możliwość logowania pakietów	Możliwość konfigurowania wyglądu logów
	Obsługa IPv6	
	Modułowość – olbrzymia ilość różnorodnych modułów poszerzających funkcjonalność pakietu	

- ◆ *dscp* — określenie wartości 6-bitowego pola DSCP w polu TOS (typ serwisu) w nagłówku IP,
- ◆ *dstlimit* — określenie maksymalnej liczby pakietów na sekundę dla dowolnego adresu IP lub portu; każde IP lub port posiada własny limit (licznik),
- ◆ *fuzzy* — określenie limitu pakietów za pomocą algorytmów logiki rozmytej,
- ◆ *limit* — określenie limitu liczby pakietów danego typu na sekundę,
- ◆ *mac* — określenie adresu MAC,
- ◆ *owner* — określanie właściciela (user) pakietu w systemie Linux,
- ◆ *physdev* — stosowane przy „przezroczystym” firewallu,
- ◆ *realm* — używane przy protokołach routingu dynamicznego,
- ◆ *state* — określanie stanu połączenia, do którego należy pakiet: *NVALID*, *NEW*, *ESTABLISHED*, *RELATED*, *SNAT*, *DNAT*.
- ◆ *time* — tworzenie reguł działających w wybranym czasie np. między 8 a 16,

Niektóre wybrane operacje (-j target) możliwe do wykonania w iptables:

- ♦ *BALANCE* — możliwość rozdzielania ruchu (*load balancing*) np. na dwa serwery www,
- ♦ *CLASSIFY* — klasyfikacja ruchu do zarządzania pasmem (*quality of service*)
- ♦ *CLUSTERIP* — wsparcie do tworzenia klastrów komputerowych,
- ♦ *MARK, CONNMARK* — „znakowanie” pakietów,
- ♦ *DNAT* — Destination NAT — przekierowywanie ruchu do innego celu,
- ♦ *LOG* — zalogowanie pakietu z odpowiednim komentarzem,
- ♦ *REDIRECT* — przekierowywanie pakietów „do siebie samego”,
- ♦ *REJECT* — usunięcie pakietu i powiadomienie źródła odpowiednim komunikatem ICMP,
- ♦ *ROUTE* — bezpośrednia zmiana routingu dla pakietu,
- ♦ *SNAT* — nat „źródłowy” — adres źródłowy pakietu zostanie zmodyfikowany,
- ♦ *TCPMSS* — zmiana wartości pola MSS (*maximum segment size*) w nagłówku TCP,
- ♦ *TTL* — zmiana wartości pola TTL.

Oczywiście nie wymieniałem tutaj wszystkich możliwości, w dodatku wiele nowych zamieszczonych jest w repozytoriach patch-o-matic, np. ciekawym modulem jest quota, umożliwiająca określenie maksymalnej ilości danych możliwych do ściągnięcia przez dany adres IP. Dokładną listę możliwości możesz sprawdzić na stronie domowej pakietu <http://www.netfilter.org/>. Ponadto jest wiele modułów dostępnych w Internecie, nie zamieszczonych w patch-o-matic, przykładowo moduły do wykrywania i np. ograniczania ruchu wymiany plików p2p.

Rozwiązania komercyjne

Oczywiście iptables ma również minusy, widoczne zwłaszcza podczas konkurencji z produktami dużych firm komercyjnych. Głównym, bardzo ogólnym zarzutem jest niewielka integracja firewalle Netfilter z innymi mechanizmami sieci. Chodzi mi chociażby o możliwość weryfikacji uprawnień (reguł) na podstawie loginów użytkowników, np. w usługach katalogowych (np. MS ActiveDirectory), serwerach RADIUS. Brak integracji z systemami IDS i antywirusowymi. Brak integracji z gotowymi Proxy aplikacyjnymi. I w tej dziedzinie rozwiązania komercyjne wygrywają, zwłaszcza że najczęściej wyposażone są w wygodne interfejsy do konfiguracji i zarządzania, co przyspiesza wdrożenie i zmniejsza koszty obsługi (krótszy czas pracy administratora).

Przykładowe firewalle komercyjne:

- ♦ Symantec Gateway Security — seria 5400 — <http://www.symantec.com/region/pl/product/GatewaySecurity5400.html>
- ♦ Check Point FireWall-1 — <http://www.checkpoint.com/products/firewall-1/index.html>

- ◆ Cisco PIX-535 — http://www.cisco.com/en/US/products/hw/vpndevc/ps2030/products_data_sheet09186a008007d05d.html
- ◆ Internet Security and Acceleration Server 2004 — <http://www.microsoft.com/isaserver/default.aspx>

Rady przydatne podczas wyboru komercyjnego firewalla:

- ◆ przeczytaj dokładnie dokumentację możliwości rozwiązania,
- ◆ nie wierz we wszystko, co jest podane na stronach producentów — pomiń „marketingowy bełkot”,
- ◆ postaraj się zainstalować wybrany produkt i zasymulować na nim jak najdokładniej funkcje, których będziesz potrzebował,
- ◆ nawiąż kontakt z przedstawicielami handlowymi — czasem istnieje możliwość wypożyczenia urządzenia do testów,
- ◆ zamęczaj przedstawicieli firmy pytaniami, zanim nabędziesz ich produkt; po zakupie już nie jesteś dla nich taki ważny jak przed,
- ◆ pamiętaj, że decydując się na — najczęściej drogie — rozwiązanie wybranej firmy, wybierasz późniejsze dokupywanie kolejnych komponentów systemu bezpieczeństwa tejże firmy,
- ◆ dobrze zastanów się nad wyborem wersji oferowanego rozwiązania i sposobem licencjonowania.

Wnioski:

- ◆ istnieje wiele rozwiązań — należy wybierać dokonując analizy potrzeb, kosztów oraz zasobów administracyjnych (wiedzy administratora),
- ◆ poważne firmy (większość dużych) oferują zintegrowane rozwiązania chroniące całą sieć, przy czym różnią się ich filozofie, np.: Cisco — The Self-Defending Network,
- ◆ koszty takich rozwiązań na polskim rynku są ogromne,
- ◆ zawsze należy uwzględniać koszt szkoleń i utrzymania rozwiązania w działaniu — koszt „roboczogodziny” administratora.

Przyszłość systemów bezpieczeństwa:

- ◆ Intrusion Protection System/Intrusion Prevention System — reakcja w czasie rzeczywistym,
- ◆ zintegrowane rozwiązania dotyczące zarządzania wszelkimi zagadnieniami bezpieczeństwa od firewalli, IDS/IPS, VPN, poprzez ochronę stacji roboczych i analizę stanu ich zabezpieczeń,
- ◆ „zaszywanie” mechanizmów bezpieczeństwa w urządzenia sieciowe — przełączniki, Access Pointy,
- ◆ wykorzystywanie algorytmów sztucznej inteligencji w celu reakcji na nowe ataki (ataki dnia zerowego).

Ipchains — Linux 2.2

W tym podrozdziale zamieszczam skrócony opis obiegu datagramów IP w jądrach wersji 2.2. Następny podrozdział jest poświęcony projektowi netfilter i dokładniej opisuje działanie iptables używanego w jądrach wersji 2.4.

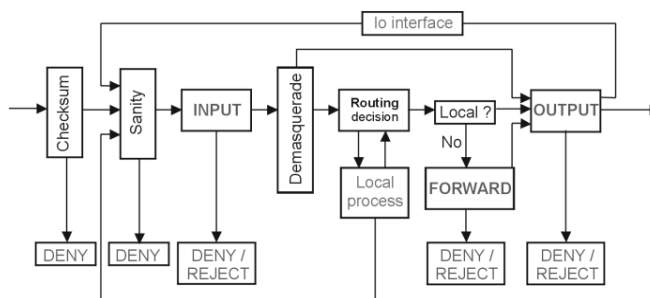
Ipchains jest linuksowym poleceniem używanym do konfiguracji reguł firewalla i tłumaczenia adresów IP (NAT). Zastąpiło ono starsze ipfwadm. Aby móc wykorzystać komputer z systemem operacyjnym Linuks do filtrowania pakietów, należy skompilować jądro systemu z opcjami (dla jądra z serii 2.2):

```
CONFIG_FIREWALL = y
CONFIG_IP_FIREWALL = y
```

Aby stwierdzić, czy jądro ma prawidłowo wkompiłowaną obsługę filtrowania datagramów, należy sprawdzić, czy w katalogu `/proc/net/` istnieje plik `ip_fwchains`, zawierający konfigurację reguł firewalla.

Przy konfigurowaniu firewalla za pomocą polecenia ipchains podstawowym pojęciem jest **łańcuch** (*chain*). Łańcuch jest to zbiór reguł filtrujących, na podstawie których podejmowana jest decyzja, czy pakiet zostanie przepuszczony, czy też usunięty. Istnieją trzy standardowe łańcuchy: *input*, *forward* i *output*, ponadto użytkownik może stworzyć własne łańcuchy.

Rysunek 11.1.
Obieg pakietów
w Linuksie 2.2



Droga datagramów IP poprzez firewall jest bardzo skomplikowana (rysunek 11.1). Na samym początku drogi pakietu sprawdzana jest suma kontrolna datagramów (*checksum*), następnie testowane są one pod kątem deformacji (*sanity*). Później pakiety przechodzą przez łańcuch wejściowy (*input chain*) i jeśli trzeba, podlegają odwrotnemu procesowi NAT (*demasquerade*), czyli adres routera usuwany jest z pola „adres docelowy” i zastępowany adresem IP docelowego komputera w sieci prywatnej. Dalej na podstawie tablicy routingu (*routing decision*) lub protokołu routującego podejmowana jest decyzja o dalszym losie pakietu. Procesy lokalne (*local process*) mogą odbierać pakiety po etapie routowania i wysyłać je poprzez etap routowania i bezpośrednio łańcuch wyjściowy (*output chain*). Jeśli pakiet nie został utworzony przez proces lokalny, jest dodatkowo sprawdzany w łańcuchu przejściowym (*forward chain*). Jeśli proces lokalny będzie się komunikował z innym procesem lokalnym, kieruje pakiet przez łańcuch wyjściowy (*output chains*) do interfejsu *lo* (*lo interface* — *loopback*). Każdy z pakietów opuszczających komputer musi zostać sprawdzony przez reguły zawarte w łańcuchu wyjściowym (*output chain*).

Składnia polecenia ipchains

Pokrótkie przedstawię podstawową składnię polecenia ipchains. Dokładny opis pozostałych opcji znajduje się w publikacji <http://www.tldp.org/HOWTO/IPCHAINS-HOWTO.html>.

Po poleceniu ipchains następuje opcja określająca jego działanie:

- F (*Flush*) — usuwa wpisy z wymienionego później łańcucha,
- A (*Append*) — dodaje nową regułę do łańcucha,
- I (*Insert*) — wstawia nową regułę do łańcucha na podane miejsce,
- D (*Delete*) — usuwa regułę z łańcucha.

Następnie występują wpisy określające źródło i cel transmisji:

- s (*source*) — źródło,
- d (*destination*) — cel.

Adresy mogą być podawane jako nr IP lub nazwa oraz jako zakresy, np.

- s 192.168.23.24 — adres źródłowy równy 192.168.23.24,
- d 192.168.23.0/24 — adres źródłowy pochodzący z sieci 192.168.23.0 o 24-bitowej masce,
- d 192.168.23.0/255.255.255.0 — zapis równoważny poprzedniemu,
- s 0.0.0.0/0 — dowolny adres.

Większość opcji umożliwia zastosowanie negacji logicznej, na przykład -s ! localhost oznacza wszystkie komputery poza lokalnym. Protokół może być podawany jako numer pobrany z pliku */etc/protocols* lub nazwa (*tcp*, *udp*, *icmp*), nie jest ważne, czy użyjesz małych, czy też wielkich liter:

- p udp — protokół UDP,

Można również podawać numer portu, dla którego układamy regułę:

- p TCP -s 0.0.0.0/0 23 — ruch protokołem TCP z dowolnego adresu z portu 23 (telnet).

Możemy definiować również zakresy portów:

- p TCP -s 192.168.1.6 20:23 — ruch protokołem TCP z adresu 192.168.1.6 z portów o numerach 20, 21, 22 i 23,
- p UDP -d 192.168.1.0/24 1024: — ruch protokołem UDP do komputerów z sieci 192.168.1.0/24 na wysokie porty (porty o numerach 1024 i więcej).

Równie dobrze możemy podać nazwę portu: -p TCP -s 0.0.0.0/0 www.

Określamy interfejs sieciowy, którego dotyczy reguła:

- i eth0 — karta sieciowa eth0,
- i eth+ — oznacza wszystkie interfejsy zaczynające się na *eth*.

Aby włączyć zapisywanie pakietów do logów systemowych, musimy dodać do reguły flagę *-l*.

Flaga *-j* specyfikuje, co należy wykonać z pakietem pasującym do reguły. Jeśli nie ma tej flagi, to reguła jest używana do prostego zliczania pakietów ją spełniających. Działaniami, które możemy zlecić, są:

- j ACCEPT — podejmuje decyzję o przepuszczeniu pakietu pasującego do reguły,
- j DENY — natychmiast likwiduje pakiety pasujące do reguły,
- j REJECT — likwiduje pakiety, wysyłając do nadawcy komunikat ICMP o nieosiągalności celu,
- j MASQ — stosuje uproszczoną odmianę NAT, nazywaną czasem „maskaradą” (jądro musi być skompilowane z *IP Masquerading enabled*); opcja prawidłowa jedynie dla łańcucha *forward*,
- j REDIRECT numer_portu — przekierowuje pakiet na lokalny port; opcja musi być użyta w łańcuchu INPUT i można jej używać jedynie dla protokołów TCP i UDP,
- j RETURN — powoduje natychmiastowe osiągnięcie końca łańcucha.

Ustalamy politykę dla łańcucha. Polityka określa, co należy zrobić z pakietem nie pasującym do żadnej z zawartych w nim reguł:

```
ipchains -P input DENY
```

Zaprezentuję poniżej przykładową konfigurację; należy ją traktować jedynie jako przykład podany w celu przedstawienia sposobu konstruowania reguł firewalla:

```
# Wyczyszczenie łańcuchów w celu pozbycia się potencjalnych pozostałości
/sbin/ipchains -F input
/sbin/ipchains -F output
/sbin/ipchains -F forward
# Przekierowanie ruchu na port używany przez usługę przezroczystego Proxy
/sbin/ipchains -A input -p tcp -s 192.1.2.0/24 -d 0.0.0.0/0 80 -j REDIRECT 8080

# Utworzenie własnego łańcucha o nazwie my-chain
/sbin/ipchains -N my-chain
# Dodawanie reguł do łańcucha my-chain
# Zezwolenie na transmisje z serwerów SMTP
/sbin/ipchains -A my-chain -s 0.0.0.0/0 smtp -d 192.1.2.10 1024: -j ACCEPT
# Zezwolenie na transmisje do serwerów SMTP
/sbin/ipchains -A my-chain -s 192.1.2.10 -d 0.0.0.0/0 smtp -j ACCEPT
# Zezwolenie na transmisje z serwerów WWW
/sbin/ipchains -A my-chain -s 0.0.0.0/0 www -d 192.1.2.11 1024: -j ACCEPT
# Zezwolenie na transmisje do serwerów WWW
/sbin/ipchains -A my-chain -s 192.1.2.0/24 1024: -d 0.0.0.0/0 www -j ACCEPT
# Zezwolenie na transmisje z serwerów DNS
```

```

/sbin/ipchains -A my-chain -p UDP -s 0.0.0.0/0 dns -d 192.1.2.0/24 -j ACCEPT
# Przekierowanie pakietów z sieci lokalnej do łańcucha my-chain
/sbin/ipchains -A input -s 192.1.2.0/24 -d 0.0.0.0/0 -j my-chain
# Konfiguracja NAT
# nie dokonuj NAT na ruchu z sieci lokalnej do sieci lokalnej
/sbin/ipchains -A forward -s 192.1.2.0/24 -d 192.1.2.0/24 -j ACCEPT
# nie dokonuj NAT dla ruchu z zewnątrz
/sbin/ipchains -A forward -s 24.94.1.0/24 -d 0.0.0.0/0 -j ACCEPT
# dokonuj NAT dla ruchu z sieci wewnętrznej na zewnątrz
/sbin/ipchains -A forward -s 192.1.2.0/24 -d 0.0.0.0/0 -j MASQ
# Nie pozwól na jakikolwiek inny ruch
/sbin/ipchains -P my-chain input DENY

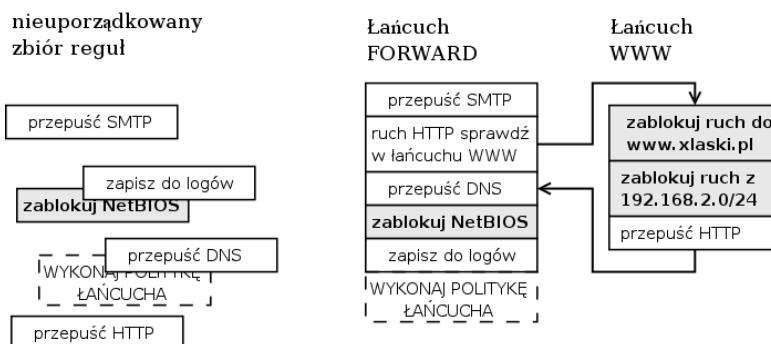
```

Iptables — Linux 2.4

Polskie tłumaczenie dokumentu *HOWTO do iptables* znajdziesz na stronie <http://lukasz.bromirski.net/docs/tlumaczenia.html>. Powinieneś się z nim zapoznać, zanim zaczniesz czytać dalszy ciąg tego działu. Przeczytaj również pozostałe tłumaczenia na stronie Łukasza Bromirskiego <http://mr0vka.eu.org> — naprawdę warto. Zajrzyj również na stronę domową pakietu netfilter <http://www.netfilter.org>. Taaaak. Wiedziałem, że nie będzie Ci się chciało czytać takiej ilości tekstu. Postaram się zatem omówić pokrótce, jak wędrują pakiety w jądrze 2.4.

Podstawowym pojęciem jest **łańcuch** (*chain*). Łańcuch to zbiór reguł filtrujących, wykonywanych w kolejności ich wystąpienia (rysunek 11.2). Istnieją trzy standardowe łańcuchy: *input*, *forward* i *output* oraz łańcuchy dodatkowe, *prerouting* i *forward*. Ponadto użytkownik może tworzyć własne łańcuchy. Łańcuch jest listą reguł, do których po kolei jest porównywany pakiet. Jeśli pakiet pasuje do którejś z reguł, wykonywana jest akcja zdefiniowana wewnątrz niej. Reguły definiujemy właśnie za pomocą polecenia iptables. Jeśli pakiet nie pasuje do żadnej reguły, wykonywana jest akcja zdefiniowana w **polityce** danego łańcucha. Tylko łańcuchy podstawowe i dodatkowe mają własną politykę, łańcuchy utworzone przez użytkownika nie mają jej, a pakiet po przejściu przez taki łańcuch wraca do miejsca (do łańcucha), z którego został wywołany.

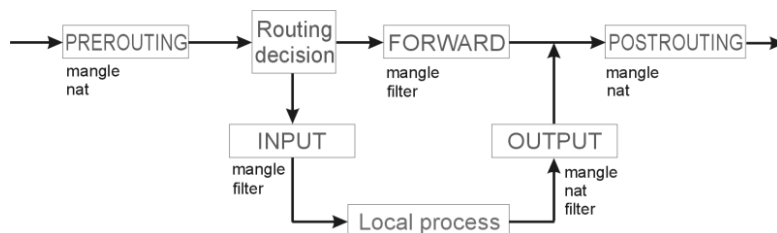
Rysunek 11.2.
Łańcuchy



W implementacji iptables pakiet sprawdzany jest najpierw w łańcuchu **PREROUTING**. Następnie podejmowana jest decyzja *Routing decision*, czy pakiet skierowany jest do komputera lokalnego. Przechodzi wtedy przez łańcuch **INPUT** i dociera do procesów

lokalnych *Local process*. Drugą możliwością jest następująca: pakiet jest skierowany do jednej z sieci, do których dany komputer jest routerem. W takim przypadku podejmowana jest decyzja o routingu danego pakietu (w uproszczeniu — definiowana karta sieciowa, przez którą należy dany pakiet dalej przesłać) i pakiet trafia do łańcucha **FORWARD**. Pakiety wygenerowane przez procesy lokalne (np. przeglądarkę WWW wysyłającą zapytanie o stronę) przechodzą przez łańcuch **OUTPUT**. Następnie wszystkie pakiety przechodzą przez łańcuch **POSTROUTING** i są wysyłane na karty sieciowe. Jeśli porównasz ten schemat (rysunek 11.3) z algorytmem dla *ipchains*, zauważysz, jak bardzo został on uproszczony i logicznie przemyślany.

Rysunek 11.3.
Obieg pakietu
w jądrach 2.4 i 2.6



Przedstawione powyżej łańcuchy są dodatkowo rozdzielone pomiędzy tablice, zawierające niektóre z nich. Pierwszą jest tabela *mangle*, służąca do zmiany zawartości pakietów przechodzących przez nasz firewall. Następnie pakiety przechodzą przez tabelę *nat*, w której definiujemy różne rodzaje translacji adresów NAT. Ostatnią jest najczęściej używana tabela *filter*. Definiujemy w niej reguły filtrowania pakietów, czyli podstawowy kod naszego firewalla. Nie wszystkie łańcuchy istnieją w poszczególnych tabelach.

Do tabeli *mangle* (polecenie: `iptables -t mangle ...`) należą wszystkie łańcuchy:

- ♦ PREROUTING,
- ♦ INPUT,
- ♦ OUTPUT,
- ♦ FORWARD,
- ♦ POSTROUTING.

Do tabeli *nat* (polecenie: `iptables -t nat ...`) należą łańcuchy:

- ♦ PREROUTING — DNAT (*destination nat*) podmieniany jest adres docelowy pakietów,
- ♦ POSTROUTING — SNAT (*source nat*) podmieniany jest adres źródłowy pakietów,
- ♦ OUTPUT — DNAT dla pakietów generowanych przez procesy lokalne.

Domyślną tabelą jest *filter* (polecenie: `iptables ...` — nie trzeba używać przełącznika `-t`), należą do niej łańcuchy:

- ♦ INPUT,
- ♦ OUTPUT,
- ♦ FORWARD.

Nie będę omawiał składni iptables; jest ona bardzo rozbudowana i zajęłaby zbyt dużo miejsca. Ponadto zostało to bardzo dobrze zrobione w wymienionym dokumencie HOWTO (z polskim tłumaczeniem). Zamieszczę tylko szablon podstawowych funkcji iptables, który może być dosyć przydatny podczas pisania firewala.

```
iptables [-t table] -[AD] chain rule-specification [options]
iptables [-t table] -I chain [rulenum] rule-specification [options]
iptables [-t table] -R chain rulenum rule-specification [options]
iptables [-t table] -D chain rulenum [options]
iptables [-t table] -[LFZ] [chain] [options]
iptables [-t table] -N chain
iptables [-t table] -X [chain]
iptables [-t table] -P chain target [options]
iptables [-t table] -E old-chain-name new-chain-name
```

Najważniejsze przełączniki w iptables:

- A (*append*) — dodaj na końcu łańcucha
- D (*delete*) — usuń regułę z łańcucha
- I (*insert*) — wstaw regułę do łańcucha na konkretne miejsce
- R (*replace*) — zamień treść wybranej reguły w łańcuchu
- L (*list*) — wyświetl zawartość łańcucha
- F (*flush*) — wyczyść (skasuj) zawartość łańcucha
- Z (*zero*) — wyzeruj liczniki pakietów i bajtów
- N (*new-chain*) — utwórz nowy łańcuch
- X (*delete-chain*) — usuń łańcuch użytkownika
- P (*policy*) — ustaw politykę dla łańcucha

Podczas tworzenia firewala postaram się omówić używane przeze mnie konstrukcje. Jeśli coś wyda Ci się niejasne, zajrzyj do HOWTO lub na stronę manuala (man iptables).