Ochrona sieci lokalnej za pomocą zapory sieciowej

Zbigniew SUSKI [1], Piotr KOŁODZIEJCZYK [2]

STRESZCZENIE: W dobie wzrastającego zagrożenia systemów sieciowych coraz większe zainteresowanie budzą różne mechanizmy mające na celu zwiększenie stopnia bezpieczeństwa systemów. Jednym z takich mechanizmów jest zapora sieciowa (ang. firewall). W artykule omówiono ogólne zasady budowy zapory sieciowej i podstawowe schematy organizacyjne. Przedstawiono mechanizmy dostępne w systemie operacyjnym Linux, które umożliwiają zbudowanie zapory w oparciu o ten system. Zamieszczono przykłady konfiguracji zapory wykorzystującej system Linux. W końcowej części opracowania przedstawiono opisy kilku oferowanych na rynku rozwiązań komercyjnych.

1. Co to jest zapora sieciowa?

Zapora sieciowa (ang. *firewall*) to konstrukcja zapewniająca kontrolowane połączenie pomiędzy siecią prywatną i Internetem (siecią publiczną). Dostarcza ona mechanizmu kontroli ilości i rodzaju ruchu sieciowego między obydwoma sieciami. Zapory sieciowe to narzędzia o dużych możliwościach, ale nie powinno się ich używać *zamiast* innych środków bezpieczeństwa, lecz *obok* nich.

Termin *firewall* zaczerpnięto z budownictwa (chociaż spotyka się też inne opinie). Bloki mieszkaniowe i budynki biurowe są często wyposażane w specjalnie skonstruowane ściany, które się opierają ogniowi. Jeśli budynek zacznie się palić, to właśnie specjalna zapora zatrzyma ogień lub przynajmniej spowolni jego rozprzestrzenianie się do czasu przybycia pomocy.

Podobna filozofia stosowana jest do ochrony sieci lokalnej przed napastnikami z zewnątrz. Stosowanie odpowiednio skonfigurowanej zapory sieciowej może minimalizować ilość strat powstałych w wyniku ataku z zewnątrz. Należy jednak pamiętać,

[1] Zakład Teleinformatyki, Instytut Automatyki i Robotyki WAT, ul. Kaliskiego 2, 00-908 Warszawa.
Wydział Nauk Komputerowych, Prywatna Wyższa Szkoła Biznesu i Administracji, ul. Bobrowiecka 9, 00-728 Warszawa.
[2] Wydział Cybernetyki WAT, ul. Kaliskiego 2, 00-908 Warszawa.

że nigdy nie będziemy mieć 100% gwarancji bezpieczeństwa, a poziom tego bezpieczeństwa jest zawsze skutkiem kompromisu pomiędzy potrzebnymi środkami, a nakładami finansowymi przeznaczonymi na ten cel.

Podstawowe funkcje, które powinna spełniać zapora sieciowa to:

a) zapewnienie „bezpiecznego" dostępu do Internetu użytkownikom sieci prywatnej,
b) zapewnienie ochrony zasobów sieci prywatnej przed atakami z zewnątrz.

Oprócz tych dwóch podstawowych funkcji można jeszcze wyszczególnić kilka dodatkowych, które z powodzeniem może realizować zapora sieciowa:

a) blokowanie dostępu do określonych miejsc w Internecie, blokowanie (całkowite lub częściowe) dostępu do Internetu określonym użytkownikom,
b) monitorowanie komunikacji pomiędzy siecią prywatną a Internetem,
c) rejestrowanie całości lub określonej części ruchu międzysieciowego,
d) tworzenie *prywatnych sieci wirtualnych* (VPN) pomiędzy oddziałami organizacji.

2. Schemat organizacyjny zapory sieciowej

Na konstrukcję zapory sieciowej zwykle składają się filtry pakietów oraz serwery proxy.

Podstawowym zadaniem zapory jest ograniczenie przepływu danych między sieciami. Przed postawieniem zapory trzeba określić, jakie rodzaje danych mają być przez nią przepuszczane, a jakie nie. Czyli trzeba zdefiniować ***politykę zapory***. Następnie należy skonstruować ***mechanizmy***, które umożliwią wprowadzenie tej polityki w życie.

Filtry pakietów to urządzenia przechwytujące każdy transmitowany pakiet danych i dopuszczające lub blokujące przesłanie tego pakietu do adresata. Decyzja o przesłaniu jest podejmowana na podstawie atrybutów rozpatrywanego pakietu. Są to m.in. adres źródłowy, adres docelowy, typ protokołu, port źródłowy, port docelowy, zawartość.

W praktyce funkcjonują dwie podstawowe strategie konfiguracji filtrów pakietów: domyślne przepuszczanie oraz domyślne powstrzymywanie. Pierwsza polega na blokowaniu tylko niektórych portów, protokołów czy adresów. Stosowana jest więc zasada: *wszystko, co nie jest zabronione jest dozwolone*. Druga strategia polega na odblokowaniu tylko niektórych portów, protokołów czy adresów. Obowiązuje więc zasada: *wszystko, co nie jest dozwolone jest zabronione.* Administrator systemu, dążący w konfiguracji zapory

sieciowej do osiągnięcia maksymalnego bezpieczeństwa, powinien oczywiście wybrać strategię domyślnego powstrzymywania.

Serwery *proxy* to pakiety programowe służące do pośredniczenia w ruchu sieciowym pomiędzy siecią prywatną a Internetem. Użytkownik sieci prywatnej, który chciałby skorzystać z usługi udostępnianej na serwerze w Internecie, rejestruje się najpierw w aplikacji serwera *proxy*. Zadaniem tego serwera jest uwierzytelnienie użytkownika i po stwierdzeniu, że ma on odpowiednie prawa, zezwolenie na skorzystanie z usługi w Internecie. Przy połączeniach z sieci zewnętrznej postępowanie jest podobne. Ponieważ serwer *proxy* działa na poziomie aplikacji, więc każdy typ aplikacji wymaga oddzielnego serwera. Taki zastaw serwerów *proxy* nazywamy bramą aplikacyjną.

Przez połączenie filtrów pakietów i serwerów *proxy*, oraz ich odpowiednie osadzenie na platformach sprzętowych, można uzyskać różne konfiguracje zapór sieciowych. Najbardziej popularne są w tej chwili cztery konfiguracje:

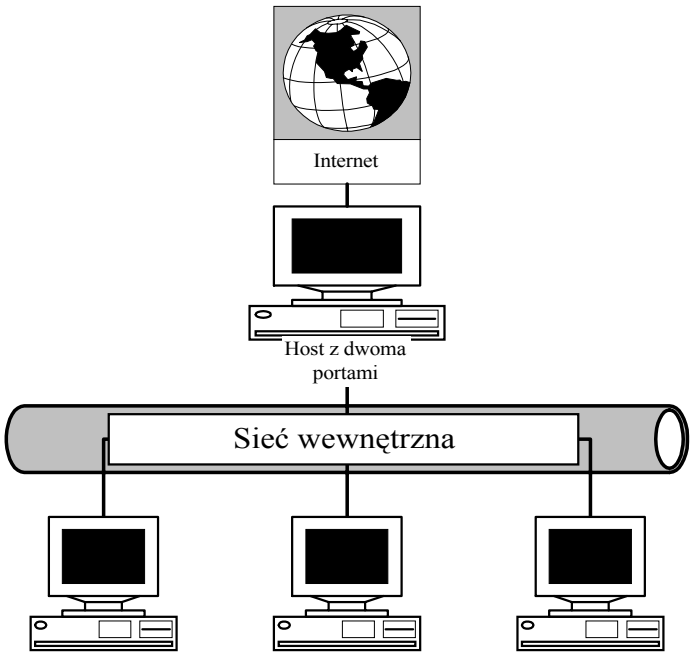
a. host z dwoma portami,
b. filtr pakietów,
c. zapora z jednym filtrem pakietów i bramą aplikacyjną,
d. zapora z dwoma filtrami pakietów i bramą aplikacyjną.

2.1. Host z dwoma portami

Jest to jedno z najstarszych rozwiązań. Polega na osadzeniu zapory na komputerze wyposażonym w dwa interfejsy sieciowe, pracującym zwykle pod kontrolą systemu operacyjnego z rodziny UNIX. Komputer w zaporze działa jednocześnie jako dławik i brama. Usługi są zwykle oferowane użytkownikom na dwa sposoby:

- użytkownik loguje się do komputera z dwoma portami,
- na hoście z dwoma portami mogą działać serwery *proxy* poszczególnych, przepuszczanych przez zaporę usług.

W systemie operacyjnym, a dokładniej mówiąc w jego jądrze musi być włączona opcja *ip_forwarding*.



Rys.1. Firewall – host z dwoma portami

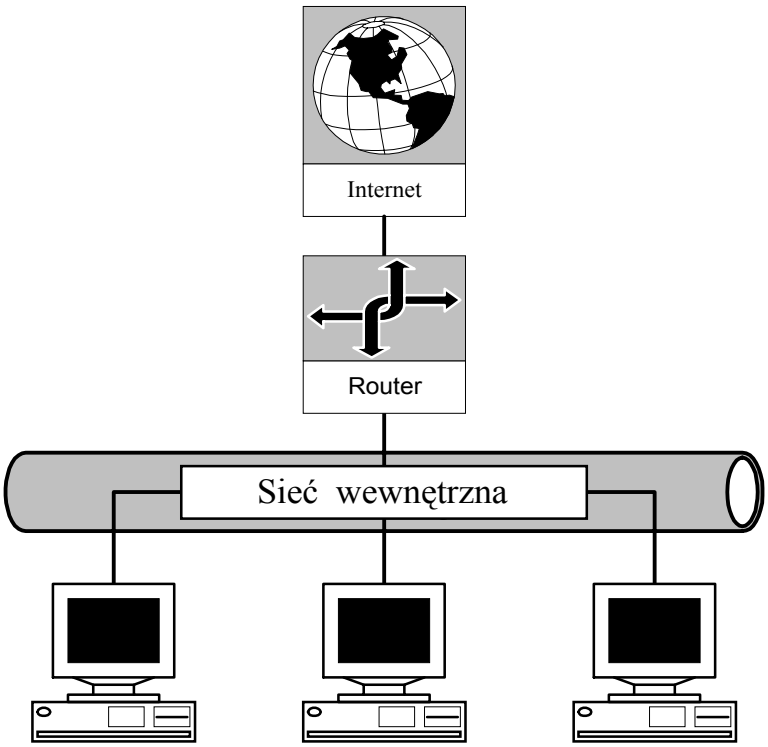
2.2. Filtr pakietów

Ten typ zapory buduje się na bazie jednego filtru pakietów. Może nim być np. *router*, w którym dostępna jest funkcja filtrowania pakietów. Jest to konfiguracja prosta i dość popularna. Programowanie filtru polega na:

- zablokowaniu pakietów wszystkich nieużywanych usług,
- zablokowaniu pakietów z ustawioną opcją *routingu źródłowego*,
- zezwoleniu na połączenia przychodzące tylko z określonych serwerów sieciowych i blokowaniu pozostałych,
- zezwoleniu komputerom z sieci wewnętrznej na połączenia z dowolnym komputerem w sieci zewnętrznej.

Do zalet takiej konfiguracji należy zaliczyć prostotę, taniość i elastyczność wyrażającą się łatwością blokowania dostępu z wybranej sieci zewnętrznej. Do wad należą:

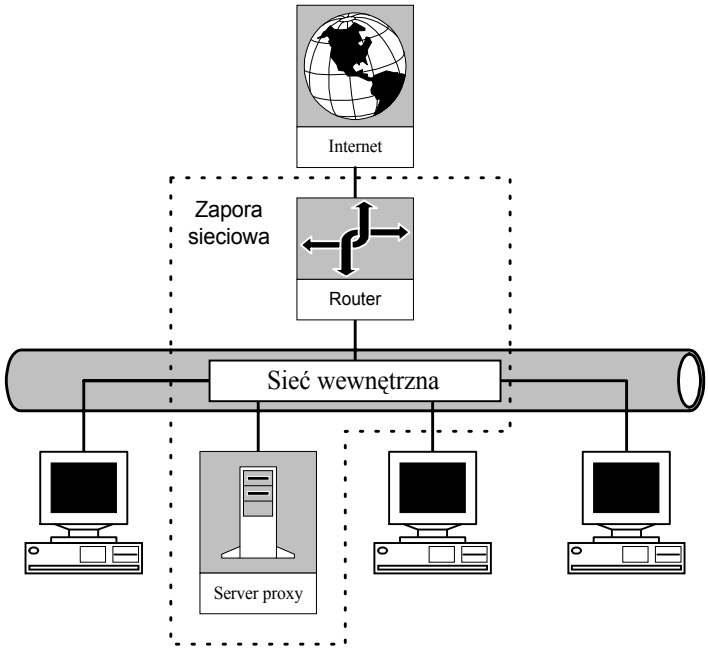
- brak lub słabo rozbudowany system rejestracji ruchu przechodzącego przez zaporę, prób włamań, udzielania użytkownikom różnego rodzaju dostępu, zwłaszcza w przypadku starszych urządzeń,
  - złożoność reguł filtrowania może być znaczna, co powoduje znaczną trudność ich weryfikowania,
  - testowanie filtru polega na eksperymentowaniu, które może być czasami dość problematyczne,
  - po złamaniu zabezpieczeń *routera*, komputery w sieci wewnętrznej będą całkowicie podatne na ataki,
  - brak zabezpieczeń przed zawartością pewnych pakietów (np. SMTP czy FTP).



Rys. 2. Firewall – filtr pakietów

2.3. Zapora z jednym filtrem pakietów i bramą aplikacyjną

Bardziej bezpieczną zaporę sieciową można zbudować stosując jednocześnie filtr pakietów i bramę aplikacyjną. Filtrem pakietów może być *router*, a bramą aplikacyjną wybrany komputer w sieci wewnętrznej. W bramie działają serwery *proxy* umożliwiające użytkownikom sieci wewnętrznej korzystanie z usług sieci zewnętrznej.



Rys.3. Zapora z jednym filtrem pakietów i bramą aplikacyjną

W tej konfiguracji filtr pakietów jest skonfigurowany w sposób zapewniający:

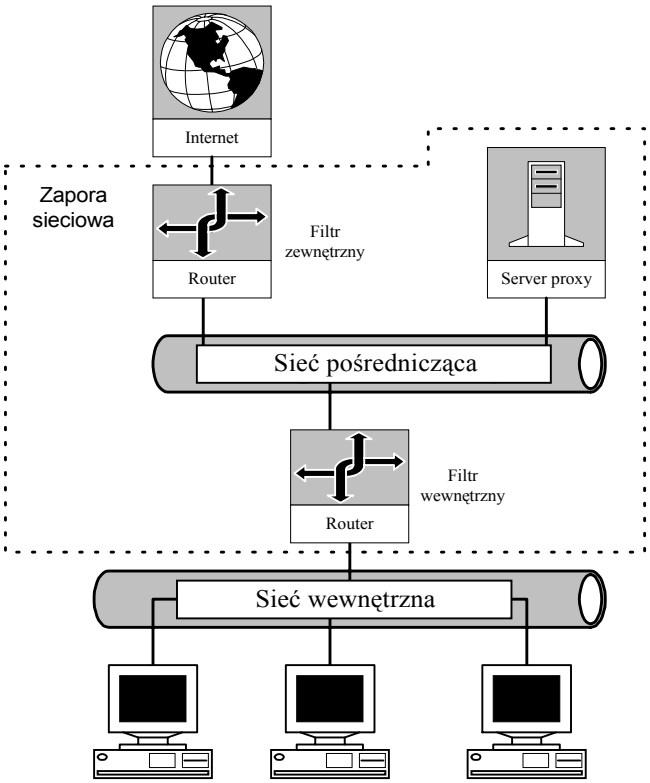
- blokowanie pakietów usług, które nie są potrzebne w sieci wewnętrznej,
- blokowanie pakietów przesyłanych w ramach *routingu źródłowego* lub mających ustawione nietypowe opcje,
- blokowanie pakietów, których miejscem przeznaczenia jest sieć wewnętrzna (poza adresem bramy),
- przepuszczanie pakietów, których adresem źródłowym lub docelowym jest adres bramy aplikacyjnej.

Jeżeli komputer w sieci wewnętrznej chce się skontaktować z siecią zewnętrzną, to pakiet komunikacyjny musi przejść przez serwer *proxy* funkcjonujący w bramie

aplikacyjnej. Użytkownicy z sieci zewnętrznej zanim dostaną się do sieci wewnętrznej, muszą się połączyć z odpowiednim serwerem *proxy*.

2.4. Zapora z dwoma filtrami pakietów i bramą aplikacyjną

W takiej konfiguracji filtr zewnętrzny i serwer *proxy* pełnią takie same funkcje jak w konfiguracji z jednym filtrem pakietów i bramą aplikacyjną. Nowym elementem jest filtr wewnętrzny, pełniący funkcję awaryjną. Jeśli intruzowi uda się włamać do serwera *proxy* i przejąć nad nim kontrolę, filtr wewnętrzny uniemożliwi mu posłużenie się serwerem *proxy* do przeprowadzenia ataków na inne komputery w sieci wewnętrznej (dzięki np.: blokowaniu pakietów usług, które nie są potrzebne w sieci wewnętrznej).



Rys.4. Zapora z dwoma filtrami pakietów i bramą aplikacyjną

We wszystkich konfiguracjach wykorzystujących bramę aplikacyjną, zamiast jednej można używać wielu bram - po jednej dla każdego protokołu. Prostsze rozwiązanie polega na zastosowaniu jednej bramy i przeznaczeniu kilku serwerów na poszczególne usługi sieci wewnętrznej.

Wszystkie komputery można pogrupować w kilka oddzielnych sieci, które będą się komunikować za pomocą specjalnych komputerów - bramek, *routerów* i zapór sieciowych. Mogą one do wewnętrznej komunikacji wykorzystywać Internet i odpowiednie systemy kryptograficzne.

Należy pamiętać, że nieuczciwi pracownicy mają o wiele dogodniejsze położenie do dokonywania zniszczeń niż włamywacze zewnętrzni. Odpowiednia konfiguracja zapór wewnętrznych może pomóc w ograniczeniu ich działań destrukcyjnych.

3. Proces budowania zapory sieciowej

Proces budowania czy *stawiania* zapory sieciowej można podzielić na kilka etapów:

1. Planowanie konfiguracji zapory sieciowej

Jest to bardzo ważny etap, ponieważ błędy popełnione przy planowaniu konfiguracji wpłyną na wszystkie dalsze etapy i w rezultacie końcowy efekt działania zapory może być całkowicie odmienny od oczekiwanego. W pierwszej kolejności należy odpowiedzieć na pytanie: *co chronić?* Jeżeli ochronie mają podlegać dwa lub trzy komputery, to prawdopodobnie zamiast budować zaporę sieciową wystarczy zastosować zabezpieczenia na poziomie pojedynczych hostów. Zapora sieciowa należy do mechanizmów *cięższego kalibru*.

Kolejne zadanie to rozpoznanie topologii sieci oraz potrzeb w zakresie aplikacji i protokołów. Polegać ono będzie na analizie topologii sieci pod kątem bezpieczeństwa, na zidentyfikowaniu systemów operacyjnych i aplikacji działających w sieci, czego efektem może być np.: konieczność skorzystania z usług ekspertów w dziedzinie bezpieczeństwa poszczególnych aplikacji.

Należy również dokonać analizy zależności służbowych. Polegać ona będzie na analizie kompetencji decyzyjnych i potrzeb dostępu do zasobów poszczególnych użytkowników czy grup użytkowników. Konieczne jest przy tym uświadomienie użytkownikom wszystkich potrzebnych zmian w konfiguracji sieci oraz wszelkich ich

wątpliwości. Od użytkowników w dużym stopniu będzie zależało bezpieczeństwo sieci i ostatnią rzeczą, jaka jest nam potrzebna to niezadowoleni użytkownicy.

Kolejna decyzja dotyczy konfiguracji zapory. Przede wszystkim należy określić czy wystarczy filtrowanie pakietów, czy też należy zastosować serwery proxy, a jeżeli tak to jakie.

Wreszcie powinniśmy rozpatrzyć czy skonstruować własną zaporę czy też kupić pakiet gotowej zapory. Samodzielnie można wykonać całkiem dobrą zaporę, lecz jeden błąd przy jej konfiguracji może spowodować katastrofę. Z drugiej strony najlepsza, źle skonfigurowana, kupiona zapora również może być przyczyną poważnych kłopotów.

Rozwiązanie alternatywne polega na wykupieniu usługi monitorowania zapory. Jeżeli nie mamy możliwości monitorowania zapory sieciowej 24 godziny na dobę przez 7 dni w tygodniu, to może lepiej taką usługę wykupić?

2. Zdefiniowanie reguł dostępu do zasobów sieciowych

W oparciu o poczynione obserwacje i wykonane analizy opracowujemy zasady korzystania z zasobów sieci. W tym etapie określamy: kto i w jaki sposób ma dostęp do sieci i jej zasobów. Reguły musimy odpowiednio dostosować do posiadanej infrastruktury. Oznacza to uwzględnienie stosowanych platform sprzętowych, czy protokołów sieciowych.

3. Znalezienie zapory odpowiedniej dla naszych potrzeb

W oparciu o zdobyte informacje, wykonane analizy i ustalone reguły dostępu do zasobów możemy właściwie wybrać potrzebną nam zaporę sieciową.

4. Właściwa instalacja i konfiguracja zapory

5. Drobiazgowe przetestowanie zapory

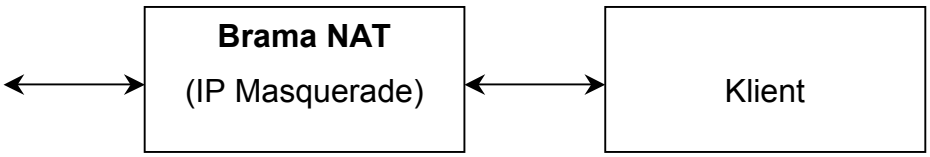
Testowanie zapory powinno odbyć się w dwóch etapach. W pierwszym należy przeprowadzić testowanie zasad korzystania z sieci prywatnej przez użytkowników zewnętrznych. W drugim testujemy wewnętrzne reguły korzystania z sieci. Oba etapy należy wykonać dokładnie, ponieważ jest to ostatnia czynność przed włączeniem zapory do sieci.

Mimo że zapora sieciowa to bardzo skuteczny środek ochrony sieci prywatnej, należy być świadomym jej wad. Jedną z nich jest fakt, że zapora, której konfigurację

zorientowano na maksymalne bezpieczeństwo sieci, będzie jednocześnie upośledzać jej działanie. Inną wadą jest zgromadzenie w jednym miejscu wszystkich składników zapory, ponieważ ich pokonanie daje intruzowi pełny dostęp do sieci prywatnej.

Istnieje kilka zagrożeń dla bezpieczeństwa sieci, które nie są eliminowane przez zastosowanie zapory:

- naruszenie bezpieczeństwa od wewnątrz, ponieważ zapora sieciowa nie chroni zasobów przed atakiem od strony użytkowników wewnętrznych,
- bezpośrednie połączenie z Internetem – jeśli użytkownik wewnętrzny połączy się z Internetem z pominięciem zapory np.: poprzez łącze telefoniczne, to stanowi to poważną lukę w bezpieczeństwie,
- często zapory sieciowe nie potrafią chronić sieci prywatnej przed wirusami,
- niektóre skanery (np.: *stealtch skaner*) potrafią skanować aktywne porty komputerów nawet za zaporą.



Rys. 5. Translacja adresów

4. Translacja adresów

W większości zapór sieciowych można uruchomić mechanizm translacji adresów. Translacja adresów (*Network Address Translation - NAT*) jest formą maskowania rzeczywistych adresów urządzeń z ochranianej sieci. Umożliwia przydzielenie komputerom z sieci wewnętrznej adresów z puli adresów nie rejestrowanych w sieci Internet (RFC 1597) oraz zapewnienie tym komputerom możliwości dwustronnego komunikowania się z komputerami sieci Internet. NAT umożliwia rozbudowę i rekonfigurację sieci TCP/IP bez obawy o wyczerpanie się oficjalnie przyznanych adresów IP. Dodatkowo umożliwia ukrycie wewnętrznej struktury sieci przed światem zewnętrznym i dostęp z zewnątrz tylko do wybranych serwerów.

W jądrze Linuxa v. 2.2.x funkcja ta dostępna jest pod nazwą ***IP Masquerade***.

Stacja kliencka powinna zdefiniować bramę NAT jako swój *gateway*. Jeżeli tak nie jest, to brama NAT powinna funkcjonować jako *server proxy arp*.

Pakiet pochodzący od klienta otrzymuje nowy numer portu źródłowego i adres źródłowy. W takiej postaci jest wysyłany. Brama zapamiętuje zrealizowane przekształcenie. Gdy pakiet wraca, to jest rozpoznawany jako przekształcony. Przywracany jest wówczas oryginalny adres klienta i pakiet trafia do klienta.

5. Mechanizmy systemu operacyjnego LINUX umożliwiające zbudowanie zapory sieciowej

W standardowej wersji dystrybucyjnej systemu LINUX zawarte są podstawowe narzędzia umożliwiające skonstruowanie zapory sieciowej. W przykładach opisana jest konfiguracja dla dystrybucji *RedHat*, która w tej chwili jest chyba najbardziej popularna wśród użytkowników. W przypadku wykorzystania innej dystrybucji mogą wystąpić w konfiguracji pewne różnice. W jądrze wersji 2.0 dostępny był mechanizm *IP Firewall*, w wersji 2.2 został on zastąpiony przez mechanizm *IPChains*. W wersji 2.4, nad którą prace jeszcze trwają udostępniony zostanie mechanizm *IP Tables*.

Pakiet *ipchains* udostępnia trzy mechanizmy przydatne przy budowaniu zapory sieciowej:

- filtrowanie pakietów,
- maskowanie adresów IP,
- przezroczysty serwer *proxy*.

Filtrowanie pakietów polega na selekcji pakietów przychodzących i wychodzących, ograniczając obustronną komunikację między siecią wewnętrzną a zewnętrzną. Zazwyczaj polega to głównie na zablokowaniu wejścia do sieci wewnętrznej, poza kilkoma wybranymi usługami. Aby zapewnić wysoki stopień bezpieczeństwa sieci wewnętrznej należy fizycznie oddzielić ją od sieci zewnętrznej. W takim wypadku dosyć naturalnym podejściem jest skonfigurowanie *firewalla* jako *routera* dla całej sieci wewnętrznej.

Maskowanie adresów IP (*masquerading*) polega na zmienianiu adresów IP w przesyłanych pakietach. *Firewall* przechwytuje wszystkie pakiety wysyłane przez

klientów z sieci wewnętrznej. Następnie jako adres źródłowy ustawia swój adres i wysyła tak zmienione pakiety do sieci zewnętrznej. Po odebraniu pakietu z odpowiedzią adres docelowy zostanie zmieniony na adres klienta i pakiet zostanie przesłany do sieci wewnętrznej. W ten sposób komputery w sieci lokalnej są zupełnie niewidzialne dla świata zewnętrznego, chociaż mogą dokonywać połączeń z komputerami zewnętrznymi. Dzięki temu można podłączyć komputery w sieci lokalnej do Internetu nawet jeśli nie mają one oficjalnie zarejestrowanych adresów IP, a używają adresów tzw. klasy publicznej 192.168.x.y

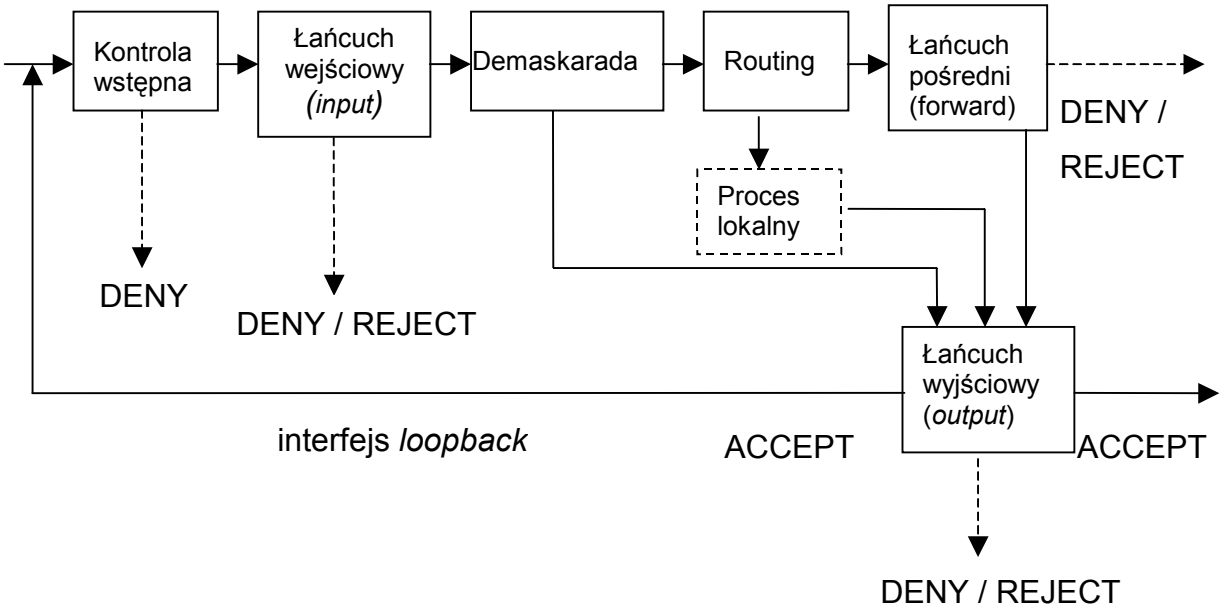
Przezroczysty serwer proxy (*transparent proxy server*) umożliwia przekierowywanie wybranych pakietów do portów lokalnych *firewalla*. W ten sposób pakiety zamiast do miejsca przeznaczenia trafiają do zapory sieciowej, w której może funkcjonować serwer określonej usługi.

5.1.1. Przepływ pakietów w systemie Linux

Istotną rzeczą dla zrozumienia działania mechanizmu *firewalla* w systemie Linux jest poznanie drogi przepływu pakietów. Ilustruje to rys. 6.

Łańcuch (*chain*) to zbiór reguł, którym powinien odpowiadać pakiet. Jeżeli nagłówek pakietu spełnia warunek zdefiniowany w regule, to wykonywana jest akcja określona w tej regule. Akcja może określać przekazanie pakietu do kolejnego łańcucha lub działanie specjalne. Do tych działań specjalnych należą:

- ACCEPT oznacza przepuszczenie pakietu.
- DENY oznacza odrzucenie pakietu.
- REJECT oznacza odrzucenie pakietu i powiadomienie poprzez ICMP o tym fakcie nadawcy.
- MASQ dotyczy łańcucha pośredniego i łańcuchów użytkownika, oznacza maskowanie pakietu poprzez zastąpienie adresu nadawcy adresem lokalnym. Przychodzące pakiety zwrotne, stanowiące odpowiedzi na pakiety maskowane, będą automatycznie rozpoznawane i *demaskowane.*
- REDIRECT dotyczy tylko łańcucha wejściowego i łańcuchów użytkownika, oznacza przekierowanie pakietu do gniazda lokalnego, czyli do lokalnego serwera *proxy*.



Rys.6. Przepływ pakietów w systemie Linux

Jeżeli reguła nie dotyczy danego pakietu, to sprawdzana jest następna reguła. Jeżeli nagłówek pakietu nie spełnia warunku w żadnej regule, to jądro uwzględnia zdefiniowaną ogólną strategię danego łańcucha.

Kontrola wstępna obejmuje sprawdzenie m.in. sum kontrolnych oraz poprawności konstrukcji pakietów docierających z sieci wewnętrznych i zewnętrznych. Po kontroli sprawdzane są reguły zdefiniowane jako łańcuch wejściowy.

Demaskarada – jeżeli pakiet zawiera odpowiedź na pakiet, który przy wysłaniu podlegał maskaradzie, to teraz ma miejsce proces odwrotny i przekazanie pakietu bezpośrednio do łańcucha wyjściowego. Pakiety, które nie podlegają demaskaradzie, są przekazywane do modułu routingu.

Routing - badane jest pole odbiorcy dla określenia, czy pakiet powinien zostać przekazany procesowi lokalnemu, czy też skierowany do innego komputera. Po przejściu przez proces lokalny i łańcuch wyjściowy pakiet może być skierowany do interfejsu *loopback* lub poprzez łańcuch pośredni i łańcuch wyjściowy skierowany na zewnątrz.

Definiowanie reguł filtrowania realizuje się poprzez polecenie *ipchains*. W starszych wersjach jądra (2.0.x) wykorzystywany był *ipfwadm*.

5.1.2. Konfiguracja jądra systemu

Aby system operacyjny mógł wykonywać funkcje zapory sieciowej, jądro systemu musi zostać skompilowane z odpowiednimi opcjami. Wybieranie opcji można zrealizować różnymi metodami. Jedna z nich, dość wygodna w użyciu polega na wykorzystaniu interfejsu *menuconfig.* Uruchamia się go poprzez polecenie *make menuconfig.* Na ekranie zostanie wyświetlona hierarchiczna lista opcji jądra, w której należy dokonać odpowiednich zaznaczeń. Interesujące nas opcje znajdują się w sekcji *Networking options*.

IP: Drop source routed frames (CONFIG_IP_NOSR)

Zwykle w transmitowanym pakiecie umieszczone są adresy IP źródła i przeznaczenia. *Routingiem* (czyli wyznaczaniem trasy przesyłania pakietu) zajmują się komputery zaangażowane w przesyłanie zwane *routerami*. One decydują , którą drogą dalej przesłać pakiet. Jednakże w protokole IP zawarta jest możliwość wyspecyfikowania pełnej drogi dla danego pakietu już przy jego wysyłaniu. Pakiety, w których w pełni określono drogę przesyłania określane są jako "trasowane według nadawcy", albo inaczej jako pakiety z ustawioną opcją *routingu źródłowego*. Powstaje pytanie, czy przy nadejściu takiego pakietu należy brać pod uwagę wymagania dotyczące trasy przesyłania, czy też pakiet taki należy odrzucić. Honorowanie trasy może wprowadzić kłopoty związane z bezpieczeństwem, wobec czego zaleca się dla tej opcji ustawić Y (yes).

Network firewalls (CONFIG_FIREWALL)

IP: firewalling (CONFIG_IP_FIREWALL)

Ustawienie tej opcji jest wymagane jeżeli wykorzystujemy protokół IP. Opcja ta wymagana jest również, gdy chcemy włączyć przezroczyste *proxy*.

IP: forwarding/gatewaying (CONFIG_IP_FORWARD)

Opcja ta umożliwia wykorzystywanie naszego komputera jako *routera* dla sieci lokalnej. W takim przypadku w komputerze są zainstalowane przynajmniej dwie karty sieciowe. Jądro nie jest w stanie wykryć więcej niż jednej karty sieciowej przy starcie komputera i należy je skonfigurować ręcznie. Jeśli komputer jest podłączony do dwóch sieci, wówczas należy wybrać N.

Jeżeli topologia sieci jest bardziej skomplikowana, na przykład rozpatrywany komputer jest podłączony do trzech sieci oraz chcemy, aby funkcjonował jako zapora

sieciowa pomiędzy dwoma z nich i jako *router* dla pozostałych, wówczas należy wybrać Y (yes) i włączyć opcję *IP firewalling*.

Jeśli zamierzamy używać mechanizmu *IP masquerading*, wówczas należy bezwzględnie wybrać Y. Podobnie postępujemy w przypadku, gdy chcemy skonfigurować komputer jako serwer SLIP lub serwer PPP, poprzez który uzyskiwać będziemy dostęp do Internetu. Odpowiedź Y musimy wybrać również w przypadku, gdy chcemy uruchomić proces *mrouted* realizujący *multicast routing*.

IP: masquerading (CONFIG_IP_MASQUERADE)

Jeśli chcemy realizować maskowanie adresów IP, to należy tę opcję ustawić. Aby używać maskowania, należy również włączyć opcje: *Network Firewalls, IP forwarding/gatewaying, IP firewalling.* Korzystne, chociaż nie konieczne, jest włączenie opcji *IP always defragment.*

IP: transparent proxying (CONFIG_IP_TRANSPARENT_PROXY)

Opcja ta umożliwia w sposób przezroczysty dla klientów przekierowywanie określonych pakietów do lokalnego serwera, określanego jako *transparent proxy server*. Dzięki temu komputery są przekonane, że są połączone z właściwym komputerem, podczas gdy w rzeczywistości połączone są z lokalnym serwerem *proxy*. Przekierowanie jest uaktywniane poprzez zdefiniowanie, przy użyciu narzędzia *ipchains* specjalnych reguł wejściowych dla zapory sieciowej.

IP: accounting (CONFIG_IP_ACCT)

Włączenie tej opcji uaktywnia rejestrowanie ruchu w sieci IP i umożliwia generowanie różnych statystyk w tym zakresie. Zarejestrowane dane dostępne są w pliku /*proc/net/ip_acct*. Dokładny zakres rejestrowanych informacji można zdefiniować przy pomocy narzędzia *ipchains*.

IP: ICMP masquerading (CONFIG_IP_MASQUERADE_ICMP)

Maskowanie włączane przez opcję *CONFIG_IP_MASQUERADE* obsługuje tylko pakiety TCP i UDP (oraz błędy ICMP dla istniejących połączeń). Omawiana opcja włącza dodatkową obsługę maskowania pakietów ICMP.

IP: ipautofw masquerading (CONFIG_IP_MASQUERADE_IPAUTOFW)

Napisany przez Richarda Lynch program *ipautofw* pozwala na maskowanie protokołów, które do tej pory nie były w pełni obsługiwane.

Kernel/User network link driver (CONFIG_IP_FIREWALL_NETLINK)

Włączany przez tę opcję sterownik pozwala na dwustronną komunikację pomiędzy pewnymi częściami jądra lub modułami i procesami użytkowymi. Procesy użytkowe uzyskują możliwość czytania i zapisywania danych do specjalnych plików znakowych o numerze głównym 36 obecnych w katalogu */dev*. Opcję należy włączyć, jeśli chcemy używać serwisu *arpd*, który pomaga utrzymać możliwie małą wewnętrzną pamięć ARP (odwzorowanie pomiędzy adresami IP i adresami sprzętowymi w sieci lokalnej).

5.1.3. Dodatkowa konfiguracja systemu

Aby możliwe było „przekazywanie" pakietów, co jest niezbędne dla wykonania maskowania adresów IP, należy uaktywnić ten mechanizm zmieniając w pliku */etc/sysconfig/network* linię z opcją FORWARD_IPV4 na : FORWARD_IPV4=yes.

Aby zapora sieciowa zbudowana na Linuxie dobrze realizowała swoje funkcje, należy spełnić jeszcze kilka dodatkowych wymagań, które ze względu na swą złożoność i niejednokrotnie potrzebę dokładnych i obszernych opisów nie zostały w niniejszym opracowaniu uwzględnione. Oto krótka specyfikacja tych dodatkowych wymagań:

- Zainstalować odpowiednie serwery *proxy,* np. dla usług http i ftp. Można je znaleźć np.: w pakiecie Trusted Information System Firewall Toolkit (TIS Toolkit). Po zainstalowaniu należy zdefiniować reguły ich wykorzystywania.
- Usunąć zbędne i niepewne programy usługowe, które zostały standardowo zainstalowane w systemie, np.: *NFS, rexec, rlogin, rsh, telnet, ftp*.
- Połączenia z komputerem pełniącym funkcje zapory sieciowej powinny odbywać się tylko szyfrowanymi kanałami wymiany informacji, przy użyciu takich programów jak *ssh*.
- Główny demon sieciowy *inetd* należy zupełnie zrekonfigurować: niektóre ze standardowych funkcji (np.: echo) należy wyłączyć, ponieważ mogą one posłużyć do wykonania ataków typu *Denial-of-Service*.

- Jeżeli istnieje potrzeba instalacji serwera poczty elektronicznej, należy poważnie zastanowić się, czy nie zrezygnować z programu *Sendmail*, znanego z licznych luk, na rzecz innego uważanego za bardziej bezpieczny, np.: *smail* czy *qmail*.
- Należy zabezpieczyć system przed niepożądanymi zmianami w plikach konfiguracyjnych czy binarnych. Można to wykonać przy pomocy programu *Tripwire*, który pozwala wykryć zmiany w plikach i katalogach.

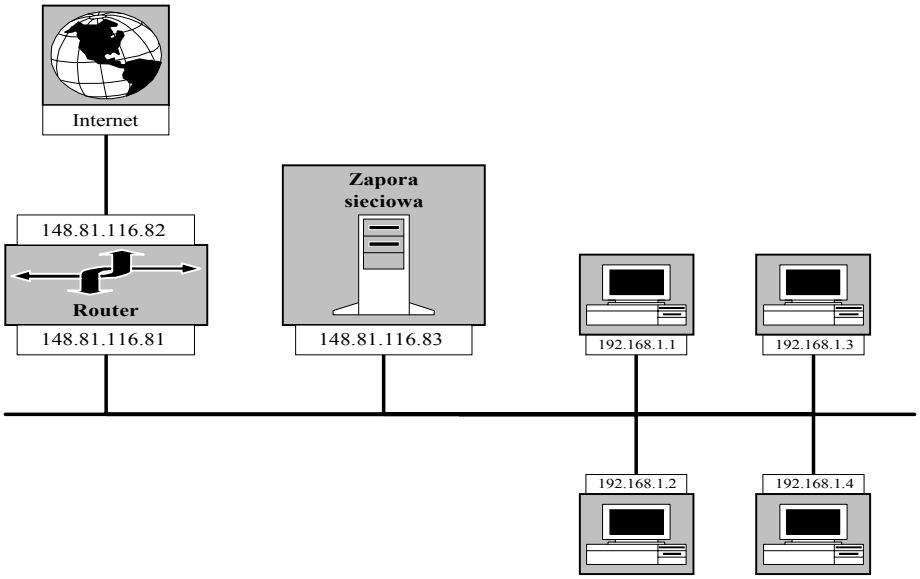
5.1.4. Przykład 1

Przykładowa sieć ma topologię przedstawioną na rys. 7.
Charakterystyka sieci:

- komputer pełniący funkcje bramy w zaporze sieciowej wyposażony jest w jeden interfejs sieciowy o numerze IP z puli światowej
- *router* jest skonfigurowany w sposób zapewniający przepuszczanie pakietów tylko do i od komputera-bramy w zaporze sieciowej,
- komputery klienckie w sieci wewnętrznej mają przydzielone adresy IP z puli prywatnej, wobec czego ich pakiety nie są przepuszczane przez *router*,
- zapora sieciowa pełni funkcję bramki dla komputerów klienckich, wykonuje dla nich maskowanie adresów IP oraz proste filtrowanie (tzn.: z domyślnym przepuszczaniem pakietów) polegające na blokowaniu niektórych usług i adresów.

Konfiguracja *routera* nie zostanie tutaj przedstawiona, gdyż jest ona bardzo mocno zależna od stosowanego sprzętu. Konfiguracja interfejsu sieciowego w komputerze-bramie może być wykonana podczas instalacji systemu lub też przez modyfikację pliku /etc/sysconfig/network-scripts/ifcfg-eth0. W pliku tym zapisane są podstawowe dane konfiguracyjne konkretnego interfejsu. Zawartość tego pliku w naszym przypadku powinna być następująca:

```
DEVICE=eth0
IPADDR=148.81.116.83
NETMASK=255.255.255.0
NETWORK=148.81.116.0
BROADCAST=148.81.116.255
ONBOOT=yes
```



Rys. 7. Topologia sieci z przykładu 1

Oprócz tego należy jeszcze zdefiniować „alias” dla interfejsu eth0, o numerze IP 192.168.1.254, aby możliwa była komunikacja z komputerami sieci wewnętrznej. Wykonujemy to poleceniem:

```
ifconfig eth0:0 192.168.1.254
```

Polecenie to powinno zostać oczywiście wpisane do skryptów startowych systemu, np.: do /etc/rc.d/rc.sysinit

Tablica *routingu* w komputerze-bramie powinna mieć postać:

```
Destination   Gateway        Genmask         Flags    Metric  Ref  Use  Iface
192.168.1.0   *              255.255.255.0     U        0      0    0   eth0
148.81.116.0  *              255.255.255.0     U        0      0    0   eth0
127.0.0.0     *              255.0.0.0         U        0      0    0   lo
Default       148.81.116.81  0.0.0.0           UG       0      0    0   eth0
```

Tylko dodanie trasy domyślnej wymaga dodatkowego omówienia, ponieważ pozostałe linie tablicy *routingu* są tworzone automatycznie przy starcie systemu, jeśli interfejs sieciowy jest prawidłowo skonfigurowany. Tę trasę domyślną określamy poleceniem:

```
route add default gw 148.81.116.81
```

lub modyfikujemy odpowiednią linię pliku /etc/sysconfig/network:

```
GATEWAY=148.81.116.81
```

Spowoduje to automatyczną konfigurację tablicy *routingu* przy starcie systemu. Maskowanie adresów IP uruchamiamy następującym poleceniem:

```
ipchains –A forward -j MASQ -s 192.168.1.0/24 -d ! 192.168.1.0/24
```

co oznacza: wykonaj maskowanie dla wszystkich połączeń o adresie źródłowym z zakresu 192.168.1.1 ÷ 254 i adresie docelowym spoza tego zakresu.

Kolejnym etapem konfiguracji zapory sieciowej jest ustawienie filtrowania. Zakładamy, że chcielibyśmy zablokować użytkownikom w sieci wewnętrznej możliwość korzystania z protokołu *http* i *telnet* oraz możliwość jakiejkolwiek łączności z komputerem o numerze IP 148.81.116.98; znajdującym się poza zaporą. Wykonamy to następującym zestawem poleceń:

```
ipchains -A input -j DENY  -s 192.168.1.0/24  -p tcp  --dport http
ipchains -A input -j DENY  -s 192.168.1.0/24  -p tcp  --dport telnet
ipchains -A input -j DENY  -s 192.168.1.0/24  -d 148.81.116.98
```

Po wydaniu tych poleceń reguły zapory wylistowane przy pomocy polecenia *ipchains -L* powinny wyglądać następująco:

```
Chain input (policy ACCEPT):
target     prot  opt   source          destination     ports
DENY       tcp   -----192.168.1.0/24  anywhere        any -> www
DENY       tcp   -----192.168.1.0/24  anywhere        any -> telnet
DENY       all   -----192.168.1.0/24  148.81.116.98   n/a
Chain forward (policy ACCEPT):
target     prot  opt   source          destination     ports
MASQ       all   -----192.168.1.0/24  !192.168.1.0/24 n/a
Chain output (policy ACCEPT):
```

Aby system był w ten sposób skonfigurowany po starcie systemu, przedstawione polecenia należy wpisać do skryptu startowego systemu, np.: /etc/rc.d/rc.sysinit.

5.1.5. Przykład 2

W kolejnym przykładzie przyjęliśmy topologię sieci przedstawioną na rys. 8.

Charakterystyka sieci:

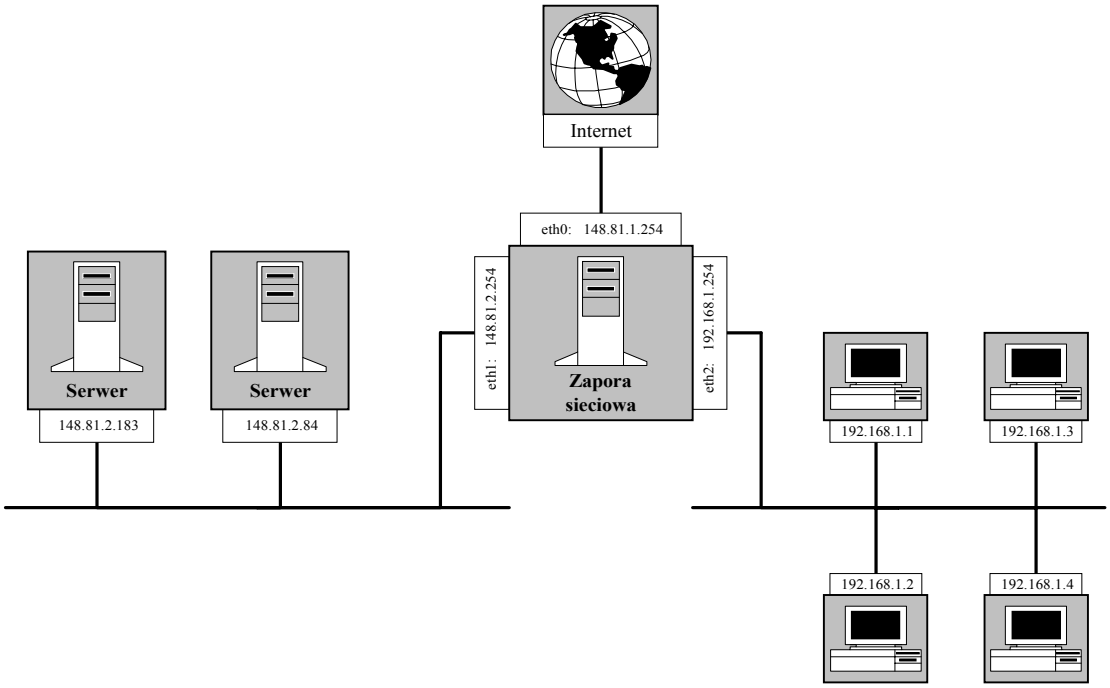
- *komputer w zaporze wyposażony jest w trzy interfejsy sieciowe: dla połączenia z Internetem i połączenia z sieciami lokalnymi; pełni więc też rolę routera,*
- *jedna z podsieci zawiera serwery z adresami IP z puli światowej,*
- *druga podsieć zawiera tylko komputery klienckie z adresami IP z puli prywatnej,*
- *zapora sieciowa wykonuje maskowanie adresów IP dla komputerów klienckich,*
- *zapora sieciowa wykonuje filtrowanie, z domyślnym blokowaniem pakietów, polegające na przepuszczaniu pakietów tylko głównych usług sieciowych.*

Aby było możliwe wykorzystanie kilku interfejsów sieciowych, Linux musi je obsługiwać, tzn. musi posiadać moduły obsługujące konkretne karty sieciowe. Sprawdzenia, czy system rozpoznał posiadane przez nas karty sieciowe można dokonać poprzez przegląd komunikatów jądra zapisywanych podczas startu systemu do pliku /var/log/messages. W czasie testowania niniejszego przykładu wykorzystywane były karty sieciowe PCI zgodne ze standardem NE2000, które system rozpoznał bez problemu.

Następnie dla każdego interfejsu sieciowego należy utworzyć plik /etc/sysconfig/network-scripts/ifcfg-eth*x* (gdzie „*x*” to numer interfejsu). W pliku tym zapisywane są podstawowe dane konfiguracyjne konkretnego interfejsu. W omawianym przypadku zawartości tych plików są następujące:

Plik /etc/sysconfig/network-scripts/ifcfg-eth0:

```
DEVICE=eth0
IPADDR=148.81.1.254
NETMASK=255.255.255.0
NETWORK=148.81.1.0
BROADCAST=148.81.1.255
ONBOOT=yes
```



Rys. 8. Topologia sieci z przykładu 2

Plik /etc/sysconfig/network-scripts/ifcfg-eth1:

```
DEVICE=eth1
IPADDR=148.81.2.254
NETMASK=255.255.255.0
NETWORK=148.81.2.0
BROADCAST=148.81.2.255
ONBOOT=yes
```

Plik /etc/sysconfig/network-scripts/ifcfg-eth2:

```
DEVICE=eth2
IPADDR=192.168.1.254
NETMASK=255.255.255.0
NETWORK=192.168.1.0
BROADCAST=192.168.1.255
ONBOOT=yes
```

Tablica *routingu* komputera w zaporze wyświetlona poleceniem *route* musi wyglądać następująco:

```
Destination   Gateway       Genmask         Flags   Metric  Ref  Use  Iface
148.81.1.0    *             255.255.255.0     U       0      0    0   eth0
148.81.2.0    *             255.255.255.0     U       0      0    0   eth1
192.168.1.0   *             255.255.255.0     U       0      0    0   eth2
127.0.0.0     *             255.0.0.0         U       0      0    0   lo
default       148.81.1.1    0.0.0.0           UG      0      0    0   eth0
```

Sposób dodania trasy domyślnej opisano w przykładzie 1. Pozostałe linie tablicy *routingu* są tworzone automatycznie przy starcie systemu. Domyślną trasę pakietów określamy zakładając, że bramka (*gateway*) ma numer IP 148.81.1.1.

Przyjmujemy następujące założenia odnośnie filtrowania pakietów w zaporze:

- komputer na którym została uruchomiona zapora sieciowa udostępnia następujące usługi:
  - ssh – do zdalnej pracy na zaporze (np.: w celu wykonania zmian w konfiguracji) z każdego miejsca w Internecie,
  - DNS – tylko dla komputerów z obu podsieci wewnętrznych,
- serwery udostępniają następujące usługi:
  - http
  - ssh
  - smtp
  - DNS
  - telnet
  - POP3
- użytkownicy pracujący na komputerach klienckich mogą korzystać z następujących usług:
  - http
  - ssh
  - smtp
  - DNS
  - telnet – tylko do serwerów w drugiej podsieci

> POP3 – tylko do serwerów w drugiej podsieci

> dostarczanych przez protokół ICMP, np.: echo

Maskowanie adresów IP dla komputerów klienckich uruchamiamy poleceniem:

```
ipchains –A forward -j MASQ -s 192.168.1.0/24 -d ! 192.168.1.0/24
```

co oznacza: wykonaj maskowanie dla wszystkich połączeń o adresie źródłowym z zakresu 192.168.1.1 ÷ 254 i adresie docelowym spoza tego zakresu.

Dla wbudowanego łańcucha „*input*" określającego reguły dostępu do zapory sieciowej przyjmujemy jako politykę domyślną - odrzucanie pakietów (DENY). Dla wbudowanego łańcucha „*forward*" określającego reguły maskowania oraz filtrowania ruchu pomiędzy wewnętrzną siecią serwerów, wewnętrzną siecią klientów oraz Internetem, przyjmujemy jako politykę domyślną - odrzucanie pakietów (DENY). Wbudowany łańcuch „*output*" nie będzie odfiltrowywał żadnych pakietów. Jako politykę domyślną przyjmujemy przepuszczanie pakietów (ACCEPT). Realizujemy to poleceniami:

```
ipchains –P input DENY
ipchains –P forward DENY
ipchains –P output ACCEPT
```

Ustawienie reguł filtrowania dla pakietów przychodzących i wychodzących przez interfejs *loopback* jest konieczne, ponieważ niektóre programy mogą korzystać z tego interfejsu. Nie stanowi to luki w systemie bezpieczeństwa ponieważ interfejs *loopback* nie jest dostępny z zewnątrz. Ustawienie to realizujemy poleceniem:

```
ipchains –A input –j ACCEPT –i lo
```

Ustawienie reguł filtrowania dla łańcucha *input*:

- pozwolenie na korzystanie z ssh i DNS

```
ipchains -A input -j ACCEPT –d 192.168.1.254 -p tcp --dport ssh
ipchains -A input -j ACCEPT –d 148.81.1.254  -p tcp --dport ssh
ipchains -A input -j ACCEPT –d 148.81.2.254  -p tcp --dport ssh
ipchains -A input -j ACCEPT –d 148.81.2.254  -p tcp --dport domain
ipchains -A input -j ACCEPT –d 148.81.2.254  -p udp --dport domain
ipchains -A input -j ACCEPT –d 192.168.1.254 -p tcp --dport domain
ipchains -A input -j ACCEPT –d 192.168.1.254 -p udp --dport domain
```

przepuszczenie pakietów będących odpowiedziami na połączenia tcp:

```
ipchains -A input -j ACCEPT -p tcp ! -y
```

- przepuszczenie odpowiedzi dla protokołu udp:

```
ipchains -A input -j ACCEPT –d 148.81.2.254  -p udp --sport domain
ipchains -A input -j ACCEPT –d 192.168.1.254 -p udp --sport domain
```

- przepuszczenie pozostałych pakietów, nie kierowanych bezpośrednio do zapory sieciowej; należy tu użyć dodatkowego łańcucha *inputnet*, ponieważ w poleceniu może wystąpić tyko jedna opcja -d:

```
ipchains –N inputnet
ipchains -A inputnet -j ACCEPT -d ! 192.168.1.254
ipchains -A input -j inputnet -s 192.168.1.0/24
ipchains -A input -j inputnet -d 192.168.1.0/24
ipchains -A input -j inputnet –s 148.81.1.0/24
ipchains -A input -j inputnet –d 148.81.1.0/24
ipchains -A input -j inputnet –s 148.81.1.0/24
ipchains -A input -j inputnet –d 148.81.1.0/24
```

Ustawienie reguł filtrowania dla łańcucha *forward*:

- blokowanie pakietów pochodzących z lub kierowanych do sieci lokalnych (należy pamiętać o umieszczeniu reguły maskowania IP na początku łańcucha forward):

```
ipchains -A forward -j DENY –s 127.0.0.0/8
ipchains -A forward -j DENY –d 127.0.0.0/8
ipchains -A forward -j DENY –s 192.168.0.0/16
ipchains -A forward -j DENY –d 192.168.0.0/16
```

- blokowanie pakietów rozgłoszeniowych:

```
ipchains -A forward -j DENY –d 192.168.1.255
ipchains -A forward -j DENY –d 148.81.1.255
ipchains -A forward -j DENY –d 148.81.2.255
ipchains -A forward -j DENY –d 255.255.255.255
```

- przepuszczenie pakietów dotyczących korzystania z wybranych usług:

```
ipchains -A forward -j ACCEPT -p tcp ! -y
ipchains -A forward -j ACCEPT -p icmp
```

```
ipchains -A forward -j ACCEPT –s 192.168.1.0/24 -p tcp --dport http
ipchains -A forward -j ACCEPT –s 192.168.1.0/24 -p tcp --dport ssh
ipchains -A forward -j ACCEPT –s 192.168.1.0/24 -p tcp --dport smtp
ipchains -A forward -j ACCEPT –s 192.168.1.0/24 -p tcp --dport domain
ipchains -A forward -j ACCEPT –s 192.168.1.0/24 -p udp –b --dport domain
ipchains -A forward -j ACCEPT –d 148.81.2.0/24 -p tcp --dport http
ipchains -A forward -j ACCEPT –d 148.81.2.0/24 -p tcp --dport ssh
ipchains -A forward -j ACCEPT –d 148.81.2.0/24 -p tcp --dport smtp
ipchains -A forward -j ACCEPT –d 148.81.2.0/24 -p tcp --dport domain
ipchains -A forward -j ACCEPT –d 148.81.2.0/24 -p udp –b --dport domain
ipchains -A forward -j ACCEPT –d 148.81.2.0/24 –s 192.168.1.0/24
                  -p tcp --dport telnet
ipchains -A forward -j ACCEPT –d 148.81.2.0/24 –s 192.168.1.0/24
                  -p tcp --dport pop-3
```

Po wydaniu przedstawionych poleceń reguły wyświetlone poleceniem *ipchains -L* powinny wyglądać następująco:

```
Chain input (policy DENY):
Target     rot   opt   source     destination     ports
ACCEPT     all   ----- anywhere   anywhere        n/a
ACCEPT     tcp   !y--- anywhere   anywhere        any -> any
ACCEPT     tcp   ----- anywhere   192.168.1.254   any -> ssh
ACCEPT     tcp   ----- anywhere   148.81.1.254    any -> ssh
ACCEPT     tcp   ----- anywhere   148.81.2.254    any -> ssh
ACCEPT     tcp   ----- anywhere   148.81.2.254    any -> domain
ACCEPT     udp   ----- anywhere   148.81.2.254    any -> domain
ACCEPT     udp   ----- 148.81.2.254    anywhere   domain -> any
ACCEPT     tcp   ----- anywhere   192.168.1.254   any -> domain
ACCEPT     udp   ----- anywhere   192.168.1.254   any -> domain
ACCEPT     udp   ----- 192.168.1.254   anywhere   domain -> any
inputnet   all   ----- 192.168.1.0/24  anywhere   n/a
inputnet   all   ----- anywhere   192.168.1.0/24  n/a
inputnet   all   ----- 148.81.1.0/24   anywhere        n/a
inputnet   all   ----- anywhere        148.81.1.0/24   n/a
inputnet   all   ----- 148.81.1.0/24   anywhere        n/a
inputnet   all   ----- anywhere   148.81.1.0/24   n/a
ACCEPT     tcp   ----- anywhere   192.168.1.254   any -> telnet
```

```
Chain forward (policy DENY):
target     prot  opt   source     destination     ports
MASQ       all   -----192.168.1.0/24  !192.168.1.0/24 n/a
DENY       all   -----127.0.0.0/8     anywhere   n/a
DENY       all   -----anywhere   127.0.0.0/8     n/a
DENY       all   -----192.168.0.0/16  anywhere   n/a
DENY       all   -----anywhere   192.168.0.0/16  n/a
DENY       all   -----anywhere   192.168.1.255   n/a
DENY       all   -----anywhere   148.81.1.255    n/a
DENY       all   -----anywhere   148.81.2.255    n/a
DENY       all   -----anywhere        255.255.255.255 n/a
fornet     all   -----192.168.1.0/24  anywhere   n/a
fornet     all   -----anywhere        148.81.2.0/24   n/a
ACCEPT     tcp   -----192.168.1.0/24  148.81.2.0/24   any -> telnet
ACCEPT     tcp   -----192.168.1.0/24  148.81.2.0/24   any -> pop-3

Chain output (policy ACCEPT):
Chain inputnet (6 references):
target     prot  opt   source     destination     ports
ACCEPT     all   -----anywhere   !192.168.1.254  n/a

Chain fornet (2 references):
target     prot  opt   source     destination     ports
ACCEPT     tcp   !y---anywhere   anywhere   any    -> any
ACCEPT     icmp  -----anywhere        anywhere        any    -> any
ACCEPT     tcp   -----anywhere        anywhere        any    -> www
ACCEPT     tcp   -----anywhere        anywhere        any    -> ssh
ACCEPT     tcp   -----anywhere        anywhere        any    -> smtp
ACCEPT     tcp   -----anywhere        anywhere        any    -> domain
ACCEPT     udp   -----anywhere        anywhere        any    -> domain
ACCEPT     udp   -----anywhere        anywhere        domain -> any
```

Aby taka konfiguracja była realizowana automatycznie przy starcie systemu, należy utworzyć odpowiedni skrypt i jego wywołanie umieścić w skrypcie startowym systemu, np.: /etc/rc.d/rc.sysinit. Należy tu dodać, że warto byłoby tak zmienić skrypty startowe systemu aby w pierwszej kolejności ustawić blokowanie wszystkich pakietów. W następnej

kolejności uruchomić interfejsy sieciowe, a potem wywołać skrypt realizujący właściwą konfigurację filtrowania. Treść takiego skryptu konfiguracyjnego byłaby następująca:

```
ipchains –P input DENY
ipchains –P forward DENY
ipchains –P output ACCEPT
ipchains –A input –j ACCEPT –i lo
ipchains -A input -j ACCEPT -p tcp ! -y
ipchains -A input -j ACCEPT –d 192.168.1.254 -p tcp --dport ssh
ipchains -A input -j ACCEPT –d 148.81.1.254 -p tcp --dport ssh
ipchains -A input -j ACCEPT –d 148.81.2.254 -p tcp --dport ssh
ipchains -A input -j ACCEPT –d 148.81.2.254 -p tcp --dport domain
ipchains -A input -j ACCEPT –d 148.81.2.254 -p udp –b --dport domain
ipchains -A input -j ACCEPT –d 192.168.1.254 -p tcp --dport domain
ipchains -A input -j ACCEPT –d 192.168.1.254 -p udp –b --dport domain
ipchains –N inputnet
ipchains -A inputnet -j ACCEPT -d ! 192.168.1.254
ipchains -A input -j inputnet -s 192.168.1.0/24
ipchains -A input -j inputnet -d 192.168.1.0/24
ipchains -A input -j inputnet –s 148.81.1.0/24
ipchains -A input -j inputnet –d 148.81.1.0/24
ipchains -A input -j inputnet –s 148.81.1.0/24
ipchains -A input -j inputnet –d 148.81.1.0/24
ipchains –A forward -j MASQ -s 192.168.1.0/24 -d ! 192.168.1.0/24
ipchains -A forward -j DENY –s 127.0.0.0/8
ipchains -A forward -j DENY –d 127.0.0.0/8
ipchains -A forward -j DENY –s 192.168.0.0/16
ipchains -A forward -j DENY –d 192.168.0.0/16
ipchains -A forward -j DENY –d 192.168.1.255
ipchains -A forward -j DENY –d 148.81.1.255
ipchains -A forward -j DENY –d 148.81.2.255
ipchains -A forward -j DENY –d 255.255.255.255
ipchains –N fornet
ipchains -A fornet -j ACCEPT -p tcp ! -y
ipchains -A fornet -j ACCEPT -p icmp
ipchains -A fornet -j ACCEPT -p tcp --dport http
ipchains -A fornet -j ACCEPT -p tcp --dport ssh
```

```
ipchains -A fornet -j ACCEPT -p tcp --dport smtp
ipchains -A fornet -j ACCEPT -p tcp --dport domain
ipchains -A fornet -j ACCEPT -p udp -b --dport domain
ipchains -A forward -j fornet -s 192.168.1.0/24
ipchains -A forward -j fornet -d 148.81.2.0/24
ipchains -A forward -j ACCEPT -d 148.81.2.0/24 -s 192.168.1.0/24
                -p tcp --dport telnet
ipchains -A forward -j ACCEPT -d 148.81.2.0/24 -s 192.168.1.0/24
                -p tcp --dport pop-3
```

5.1.6. Mechanizm *IPTables*

Jak wspomniano na początku pkt. 5, w jądrze wersji 2.4 dostępny będzie nowy mechanizm filtrowania pakietów i translacji adresów znany pod nazwą *IPTables.* Można się z nim zapoznać w dostępnych aktualnie wersjach rozwojowych jądra 2.3.x. Wersja stabilna jest jeszcze w fazie testowania (wersja 2.4.0)[3].

Mechanizm *IPTables* wydaje się być prostszy i bardziej przejrzysty od poprzednich a przez to łatwiejszy do zrozumienia. Umożliwi to konstruowanie łatwiejszych w konserwacji zapór sieciowych, uniknąć wielu pomyłek przy ich budowie i tym samym zwiększyć bezpieczeństwo sieci.

Całość została podzielona na logiczne klasy-tablice (*tables*). Każda z nich zawiera predefiniowane zbiory reguł. Obecnie zaimplementowano następujące tablice:

- ***filter*** – jej zadaniem jest filtrowanie pakietów,
- ***nat*** – jej zadaniem jest translacja adresów źródłowych i docelowych,
- ***mangle*** – jej zadaniem jest znakowanie pakietów.

Tablica ***filter*** jest głównie przeznaczona do budowy zapór filtrujących, kontroli i zliczania ruchu w sieci, diagnostyki sieci. W tablicy tej umieszczono następujące predefiniowane zbiory reguł:

- INPUT – zbiór reguł dla pakietów przeznaczonych dla lokalnego hosta,

[3] Jądra systemu Linux mają identyfikatory postaci: A.B.C. A jest numerem wersji, B- numerem podwersji, C- numerem łaty (*patch*). Wersja jest stabilna, jeżeli B jest liczbą parzystą. Pozostałe to wersje rozwojowe (beta), przeznaczone głównie dla tworzących jądro programistów, a także osób chcących sprawdzić jego nowe możliwości.

- OUTPUT - zbiór reguł dla pakietów pochodzących z lokalnego hosta,
- FORWARD - zbiór reguł dla pakietów przechodzących przez lokalny *router*.

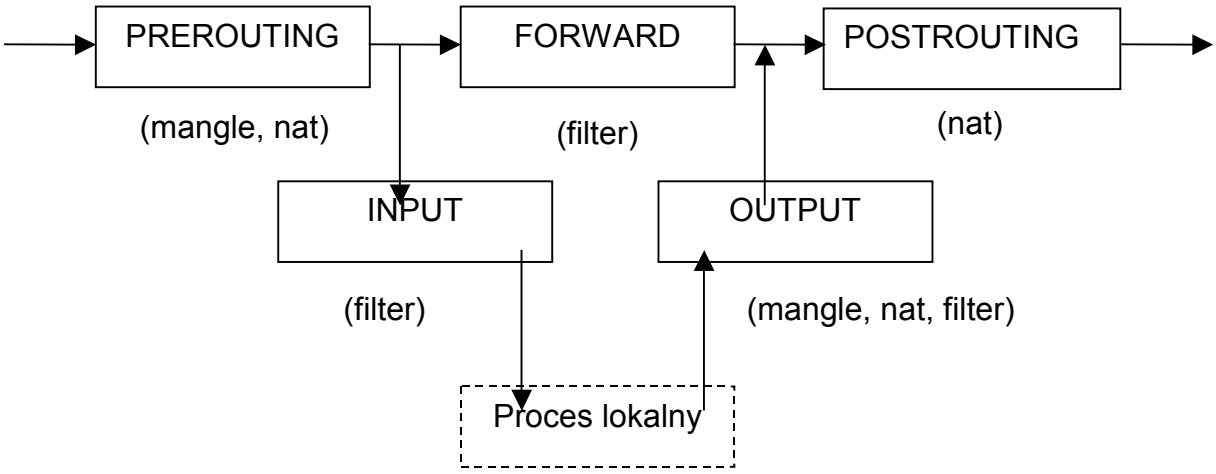
Tablica ***nat*** jest głównie przeznaczona do maskowania adresów IP (*IP masquerading*), przekierowania portów (*port forwarding*), budowy przezroczystych *proxy* (*transparent proxying*). W tablicy tej umieszczono następujące predefiniowane zbiory reguł:

- PREROUTING – zbiór reguł dla pakietów przychodzących z zewnątrz hosta,
- OUTPUT - zbiór reguł dla pakietów pochodzących z lokalnego hosta,
- POSTROUTING - zbiór reguł dla pakietów wychodzących na zewnątrz hosta.

Tablica ***mangle*** służy głównie do kontroli przepływu danych (ograniczanie pasma, routing rozszerzony). W tablicy tej umieszczono następujące predefiniowane zbiory reguł:

- PREROUTING – jak dla tablicy ***nat***,
- OUTPUT – jak dla tablicy ***nat***.

Oprócz predefiniowanych, można również wykorzystywać własne zbiory reguł. Kolejność przetwarzania poszczególnych zbiorów reguł przedstawiono na rys. 9.



Rys. 9. Droga pakietów przez tablice I*PTables*

W regule definiowany jest wzorzec i akcja. Wzorzec to kryterium jakie musi spełnić pakiet, aby wykonana została na nim określona akcja. W IPTables można definiować następujące wzorce:

- docelowy i źródłowy adres IP,
- protokół (TCP, UDP, ICMP),

- interfejs sieciowy,
- flagi pakietów (SYN, ACK, FIN, URG, itd.),
- typy pakietów (*echo-reply, echo-reguest, destination-unreachable*),
- docelowy i źródłowy port pakietów TCP i UDP,
- adres MAC interfejsów ethernetowych,
- częstotliwość napływu pakietów,
- staus pakietu (NEW, ESTABLISHED, RELATED, INVALID),
- właściciel pakietu (UID, GID, PID, SID).

Lista akcji przedstawia się następująco:

- DROP – zniszczenie pakietu,
- ACCEPT – przepuszczenie pakietu,
- RETURN – zakończenie przetwarzania bieżącego zbioru reguł,
- QUEUE – umieszczenie pakietu w kolejce do procesu użytkownika,
- MARK – oznakowanie pakietu,
- REJECT – zniszczenie pakietu z powiadomieniem nadawcy (przez ICMP),
- TOS – ustawienie flag TOS (*Type Of Service*) pakietu,
- DNAT – translacja adresu docelowego,
- SNAT – translacja adresu źródłowego,
- REDIRECT – przekierowanie pakietu na inny port,
- MASQUERADE – maskowanie adresu IP.

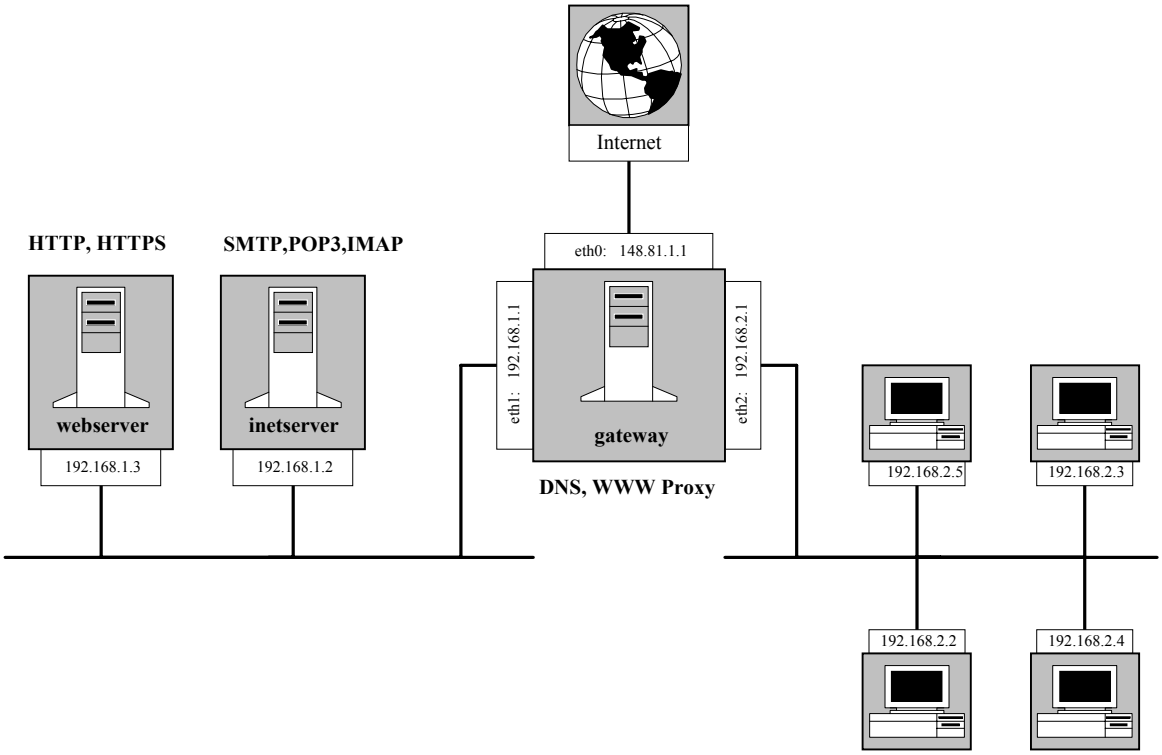
5.1.7. Przykład 3

W kolejnym przykładzie przyjęliśmy topologię sieci przedstawioną na rys. 10.

Charakterystyka sieci:

- komputer w zaporze wyposażony jest w trzy interfejsy sieciowe: dla połączenia z Internetem i połączenia z sieciami lokalnymi; pełni więc też rolę *routera,*
- jedna z podsieci zawiera serwery z prywatnymi adresami IP,
- druga podsieć zawiera tylko komputery klienckie z adresami IP również z puli prywatnej,
- zapora sieciowa wykonuje filtrowanie, z domyślnym blokowaniem pakietów, polegające na przepuszczaniu pakietów tylko głównych usług sieciowych.

Postawiony cel można osiągnąć konfigurując komputer w zaporze przy pomocy przedstawionego poniżej ciągu poleceń. Polecenia te powinny zostać umieszczone w jednym ze skryptów wywoływanych podczas startu systemu.



Rys. 10. Topologia sieci z przykładu 3

Reguły wstępne:

```
iptables -N log_and_drop
iptables -A log_and_drop -j LOG
iptables -A log_and_drop -j DROP
iptables -N log_and_reject
iptables -A log_and_reject -j LOG
iptables -A log_and_reject -j REJECT
```

Konfiguracja NAT

Pakiety przychodzące z Internetu skierowane do hosta *gateway* na port 80 (WWW) oraz port 443 (HTTPS), zostaną skierowane do hosta *webserver.*

```
iptables -t nat -A PREROUTING -i eth0 -p TCP -d 148.81.1.1
          --dport www -j DNAT --to-destination 192.168.1.3:www
iptables -t nat -A PREROUTING -i eth0 -p TCP -d 148.81.1.1
          --dport https -j DNAT --to-destination 192.168.1.3:https
```

Pakiety kierowane do hosta *gateway* na port 25 (SMTP), zostaną skierowane do hosta *inetserver:*

```
iptables -t nat -A PREROUTING -i eth0 -p TCP -d 148.81.1.1
          --dport smtp -j DNAT --to-destination 192.168.1.2:smtp
```

Ruch HTTP i HTTPS przychodzący z sieci LAN i kierowany na zewnątrz, przepuszczamy przez przezroczyste *proxy*. Rolę serwera *proxy* pełnić będzie program SQUID nasłuchujący w porcie 8081 (*tproxy*):

```
iptables -t nat -A PREROUTING -i eth2 -p TCP -d ! 148.81.1.1
         --dport www -j REDIRECT --to-port tproxy
iptables -t nat -A PREROUTING -i eth2 -p TCP -d ! 148.81.1.1
         --dport https -j REDIRECT --to-port tproxy
```

Dla wszelkiego ruchu wychodzącego do Internetu ustawiamy maskowanie adresów:

```
iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to-source 148.81.1.1
```

Konfiguracja filtra

- **Zbiór reguł INPUT:**

Tworzymy własny zbiór reguł pod nazwą *ext_in*:

```
iptables -N ext_in
```

Kierujemy do tego zbioru wszystkie pakiety przychodzące z interfejsu eth0:

```
iptables -A INPUT -i eth0 -j ext_in
```

Odrzucamy i rejestrujemy pakiety nie skierowane na zewnętrzny adres IP:

```
iptables -A ext_in -d ! 148.81.1.1 -j log_and_drop
```

Dopuszczamy ruch DNS oraz SSH:

```
iptables -A ext_in -p TCP -d 148.81.1.1 --dport domain -j ACCEPT
iptables -A ext_in -p UDP -d 148.81.1.1 --dport domain -j ACCEPT
iptables -A ext_in -p TCP -d 148.81.1.1 --dport ssh -j ACCEPT
```

Dopuszczamy ruch ICMP oraz TCP i UDP na portach wolnodostępnych:

```
iptables -A ext_in -p TCP --dport 1024-65535 -j ACCEPT
iptables -A ext_in -p UDP --dport 1024-65535 -j ACCEPT
iptables -A ext_in -p ICMP -j ACCEPT
```

Całą resztę ruchu z zewnątrz odrzucamy i rejestrujemy:

```
iptables -A ext_in -j log_and_drop
```

Nie ograniczamy ruchu z wnętrza obydwu sieci wewnętrznych do hosta *gateway*:

```
iptables -A INPUT -i eth2 -s 192.168.2.0/24 -d 192.168.2.1 -j ACCEPT
iptables -A INPUT -i eth1 -s 192.168.1.0/24 -d 192.168.1.1 -j ACCEPT
```

Całą resztę ruchu z sieci wewnętrznych odrzucamy i rejestrujemy:

```
iptables -A INPUT -j log_and_drop
```

- **Zbiór reguł OUTPUT**

Blokujemy nawiązywanie połączeń z podsiecią komputerów klienckich, a resztę dopuszczamy:

```
iptables -A OUTPUT -o eth2 --match state --state NEW
iptables -P OUTPUT -j ACCEPT
```

- **Zbiór reguł FORWARD**

Definiujemy 6 podzbiorów reguł:

```
iptables -N fw_lan_isn
iptables -N fw_lan_inet
iptables -N fw_lan_lan
iptables -N fw_isn_inet
iptables -N fw_inet_lan
iptables -N fw_inet_isn
```

Kierujemy pakiety do poszczególnych podzbiorów reguł:

```
iptables -A FORWARD -i eth2 -s 192.168.2.0/24 -o eth1 -j fw_lan_isn
iptables -A FORWARD -i eth2 -s 192.168.2.0/24 -o eth0 -j fw_lan_inet
iptables -A FORWARD -i eth1 -s 192.168.1.0/24 -o eth2 -j fw_isn_lan
iptables -A FORWARD -i eth1 -s 192.168.1.0/24 -o eth0 -j fw_isn_inet
iptables -A FORWARD -i eth0 -o eth2 -j fw_inet_lan
iptables -A FORWARD -i eth0 -o eth1 -j fw_inet_isn
```

Dopuszczamy ruch ICMP, resztę odrzucamy i rejestrujemy:

```
iptables -A FORWARD -p ICMP -j ACCEPT
iptables -A FORWARD -j log_and_drop
```

Ustawiamy reguły w każdym ze zdefiniowanych podzbiorów. Prześledzenie ich pozostawiamy czytelnikowi:

```
iptables -A fw_lan_isn -p TCP -d webserver --dport ssh -j ACCEPT
iptables -A fw_lan_isn -p TCP -d webserver --dport www -j ACCEPT
iptables -A fw_lan_isn -p TCP -d webserver --dport https -j ACCEPT
iptables -A fw_lan_isn -p TCP -d inetserver --dport ssh -j ACCEPT
iptables -A fw_lan_isn -p TCP -d inetserver --dport smtp -j ACCEPT
iptables -A fw_lan_isn -p TCP -d inetserver --dport pop-3 -j ACCEPT
iptables -A fw_lan_isn -p TCP -d inetserver --dport imap -j ACCEPT
iptables -A fw_lan_isn -p TCP --dport 1024-65535 -j ACCEPT
iptables -A fw_lan_isn -p UDP --dport 1024-65535 -j ACCEPT
iptables -A fw_lan_inet -p TCP --dport www -j log_and_drop
iptables -A fw_lan_inet -p TCP --dport https -j ACCEPT
iptables -A fw_lan_inet -p TCP --dport ftp -j ACCEPT
iptables -A fw_lan_inet -p TCP --dport ftp-data -j ACCEPT
iptables -A fw_lan_inet -p TCP --dport ssh -j ACCEPT
iptables -A fw_lan_inet -p UDP --dport domain -j ACCEPT
iptables -A fw_lan_inet -p TCP --1024-65535 -j ACCEPT
iptables -A fw_lan_inet -p UDP --1024-65535 -j ACCEPT
iptables -A fw_isn_lan -j ACCEPT
iptables -A fw_isn_inet -j ACCEPT
iptables -A fw_inet_isn -p TCP -d webserver --dport www -j ACCEPT
iptables -A fw_inet_isn -p TCP -d webserver --dport https -j ACCEPT
iptables -A fw_inet_isn -p TCP -d inetserver --dport smtp -j ACCEPT
iptables -A fw_inet_isn -p TCP --dport 0-1023 -j log_and_drop
```

```
iptables -A fw_inet_isn -p TCP --dport 1024-65535 -j ACCEPT
iptables -A fw_inet_isn -p UDP --dport 1024-65535 -j ACCEPT
iptables -A fw_inet_lan -p TCP --dport 0-1023 -j log_and_drop
iptables -A fw_inet_lan -p UDP --dport 0-1023 -j log_and_drop
iptables -A fw_inet_lan --match state --state NEW -j log_and_drop
iptables -A fw_inet_lan -j ACCEPT
```

6. Komercyjne zapory sieciowe

Na rynku dostępnych jest wiele komercyjnych zapór sieciowych. Należą do nich m.in.:

- AltaVista Firewall 98
- BorderWare Firewall Server
- Cisco PIX Firewall
- Check Point Firewall-1
- Gauntlet Firewall
- Raptor Firewall
- ISA Server 2000

AltaVista Firewall 98

5 maja 1998 firma Digital Equipment Corporation, wprowadziła na rynek AltaVista Firewall 98, najnowszą wówczas wersję wielokrotnie nagradzanego oprogramowania, działającego w środowisku systemów Windows NT i UNIX. Oprogramowanie AltaVista Firewall 98, certyfikowane przez stowarzyszenie NCSA, jest zaporą, która chroni sieć aktywnie, automatycznie włączając mechanizmy przeciwdziałające i podejmujące zaawansowane akcje, gdy atak staje się groźny. W połączeniu z AltaVista Tunnel 98 oprogramowanie AltaVista Firewall 98 zapewnia administratorom sieci możliwość tworzenia wirtualnych sieci prywatnych (*virtual private network VPN*) na bazie Internetu.

Konstruktorzy AltaVista Firewall 98 przewidzieli obsługę izolowanych sieci LAN lub tzw. stref zdemilitaryzowanych (DMZ), w których przedsiębiorstwa mogą umieszczać serwery wspomagające klientów lub umożliwiające prowadzenie handlu. Strefy DMZ zawierające serwery WWW, serwery pocztowe lub niewidoczne serwery FTP są chronione

przez mechanizmy AltaVista Firewall 98, ale są nadal dostępne poprzez Internet. Graficzny interfejs użytkownika umożliwia administratorom systemu ustalanie w prosty sposób zasad bezpieczeństwa oddzielnych dla stref DMZ i sieci intranetowych.

Użytkownicy AltaVista Firewall 98 mogą integrować wyroby innych producentów zapewniając bezpieczeństwo całego środowiska poprzez stosowanie protokołu CVP (*Content Vectoring Protocol*). Część zwykłych aplikacji obsługujących protokół CVP zawiera oprogramowanie Finjan Java Screening dla apletów w języku Java oraz Norton Antivirus firmy Symantec dla ochrony przed wirusami.

Administratorzy mogą tworzyć i wdrażać specyficzne zasady dla poszczególnych grup użytkowników i serwerów. Poprzez specyfikację praw dostępu dla poszczególnych serwerów, grup lub oddziałów, można uzyskać szerokie i elastyczne możliwości definiowania zasad bezpieczeństwa w obrębie całej firmy. Na przykład, tylko pracownicy działu osobowego mogą mieć dostęp do serwera obsługującego ten dział, natomiast wszyscy zatrudnieni będą mogli skorzystać z technicznych opisów produktów, które mieszczą się na serwerze działu marketingu.

Jako znaczące rozszerzenie możliwości aktywnego reagowania AltaVista Firewall 98 na próby włamań wprowadzono opcję ustawiania progu liczby zdarzeń, które powodują włączenie alarmu. Gdy takie zdarzenie występuje w systemie, zapora notuje ile razy ono wystąpiło, włączając alarm tylko w przypadku przekroczenia ustalonej liczby zdarzeń. Taka możliwość redukuje liczbę fałszywych alarmów powodowanych na przykład wprowadzeniem przez użytkownika nieprawidłowego hasła jeden lub dwa razy.

BorderWare Firewall Server

BorderWare Firewall Server jest kompletnym serwerem Internetu oraz systemem bezpieczeństwa. Uniemożliwia niepowołanym użytkownikom dostęp do poufnych informacji w sieciach wewnętrznych, zapewniając jednocześnie autoryzowanym użytkownikom korzyści płynące z pełnego dostępu do Internetu.

BorderWare Firewall Server łączy w sobie internetowe serwery poziomu aplikacji takie jak World Wide Web, Mail, News i Name Service oraz "*proxy*" dla aplikacji takich jak Mosaic, Telnet i FTP, z przezroczystym firewallem IP. Wszystkie standardowe aplikacje sieciowe, takie jak Telnet, FTP czy Mosaic, mogą pracować bez żadnych modyfikacji ponieważ serwery proxy są dla nich przeźroczyste Firewall umożliwia dostęp z Internetu

do sieci wewnętrznej jedynie legalnym użytkownikom. Schematy autoryzacji użytkowników wykorzystują do generowania jednokrotnych haseł algorytmy oparte na DES. Aby wygenerować jednorazowe hasło służące do zweryfikowania tożsamości, użytkownik musi posiadać specjalną kartę bezpieczeństwa CryptoCard ("*token*") i osobisty numer identyfikacyjny (PIN). BorderWare Firewall Server potrafi wykorzystywać serwer autentykacji SafeWord™ w celu potwierdzania tożsamości użytkowników sieci. Ale już sam BorderWare Firewall Server może korzystać z ponad 20 typów kart autentykacyjnych, włącznie z SafeWord DES Gold.

BorderWare Firewall Server umożliwia badanie, kontrolowanie, audytowanie i weryfikowanie całego ruchu sieciowego przychodzącego i wychodzącego z sieci zaufanej, zarówno na poziomie pakietów, jak i na poziomie połączenia.

Jest on konfigurowany jest za pomocą dowolnej przeglądarki WWW obsługującej aplety Javy. Poprzez interfejs graficzny napisany w Javie można zmienić tryb pracy serwera, prawa dostępu czy też zdefiniować nowy poziom bezpieczeństwa. Odpowiednio przygotowane przez producenta skrypty sprzężone z interfejsem graficznym znakomicie ułatwiają konfigurowanie pakietu. Można określić z których serwerów i w jakich godzinach można korzystać, a z których nie. Np. od 8.00 do 15.00 zabraniamy korzystać z WWW i FTP. Dla każdego dnia tygodnia można przygotować inną konfigurację. Dzięki zastosowaniu zaawansowanego systemu autoryzacji użytkowników wykorzystującego szyfrowanie transmisji z użyciem mechanizmu SSL (*Secure Sockets Layer*) praktycznie wykluczono możliwość zmieniania ustawień serwera przez osobę nieuprawnioną. Dzięki temu z powodzeniem można bezpiecznie zarządzać *firewallem* z dowolnego miejsca w Internecie.

W celu podwyższenia poziomu zabezpieczeń sieciowych BorderWare Firewall Server oferuje możliwość zwielokrotnionej translacji adresów (*Multiple Address Translation MAT*). W wyniku uzyskujemy połączenie ochrony przed atakiem z zewnątrz z obsługą wielu serwerów internetowych jednej klasy, np. serwerów WWW przypisanych do różnych domen. Serwery umieszczone w opcjonalnej, bezpiecznej sieci serwerów (*Secure Server Network - SSN*) mogą być widziane z zewnątrz poprzez odpowiednie nazwy symboliczne lub adresy IP. SSN jest opcjonalnym rozwiązaniem dającym możliwość zintegrowania z firewallem serwerów pochodzących od innych producentów bez naruszania integralności samego firewalla, przy zapewnieniu ochrony "obcym" serwerom. Komputery, na których pracują dodatkowe serwery są umieszczane w sieci podłączonej do trzeciego interfejsu

sieciowego firewalla. Ten trzeci segment sieci jest odseparowany od pozostałych segmentów, co powoduje, że w przypadku włamania do jednego z hostów w SSN nie zostanie naruszone bezpieczeństwo chronionej sieci wewnętrznej. Dzięki SSN nie trzeba dodatkowych serwerów umieszczać przed firewallem narażając je tym samym na bezpośrednie ataki, ani w sieci wewnętrznej, co z kolei obniżyłoby jej zabezpieczenia.

Dzięki MAT możliwe jest również ukrycie za jednym BorderWare Firewall Serverem kilku dublujących się serwerów, rozkładających pomiędzy siebie obciążenie ruchem sieciowym. Uzyskujemy wtedy duże lepsze parametry pracy systemu, zwłaszcza w przypadku intensywnie odwiedzanych serwerów WWW. W celu filtrowania dostępu do dokumentów identyfikowanych przez URL (np. stron WWW) zintegrowano z firewallem system SmartFilter™. Zapobiega to wykorzystywaniu sieci w celach inniuch niż zawodowe, zatykaniu przepustowości łącza internetowego i stracie czasu pracowników.

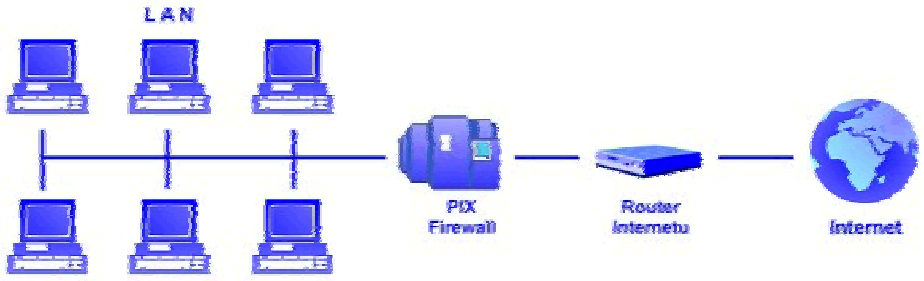
W BorderWare Firewall Server wbudowany jest całkowicie automatyczny system zarządzania kluczami szyfrowania, który pozwala na łatwą i bezpieczną wymianę kluczy. Elementem tego systemu jest mocny algorytm szyfrowania, co przy zautomatyzowaniu procesu generowania nowych kluczy redukuje prawdopodobieństwo uzyskania przez niepowołane osoby dostępu do kluczy szyfrowania. Wpływa to również na efektywność przebiegu procedury wymiany kluczy.

Rozszerzając zakres zabezpieczeń poza firewall wprowadzono system Wirtualnej Sieci Prywatnej (VPN) zapewniający bezpieczną i poufną komunikację przez Internet. BorderWare Firewall Server ma możliwość łączenia się z innymi produktami zgodnymi ze standardem IPSec tworząc zaszyfrowany kanał komunikacyjny. Przy wykorzystaniu Internetu jak szkieletu takiej sieci uzyskujemy za niewielką cenę możliwość efektywnej i poufnej komunikacji klasy WAN z różnymi miejscami na całym świecie.

Cisco PIX Firewall

Cisco PIX Firewall (Private Internet Exchange) do ochrony sieci przed niepowołanym dostępem z zewnątrz, wykorzystuje mechanizm translacji adresów NAT (*Network Address Translation*). Technologia NAT jest dostępna w systemie operacyjnym routerów Cisco (*Cisco IOS*) od połowy 1996 roku.

PIX jest urządzeniem wyposażonym w dwa porty Ethernet. W klasycznej konfiguracji jeden z portów dołączony jest do sieci lokalnej, drugi natomiast do wyodrębnionego segmentu, w którym znajduje się tylko *router* internetowy. Ilustruje to rys. 11.



Rys. 11. Topologia sieci wykorzystującej Cisco PIX Firewall

PIX operuje na bezpiecznym jądrze czasu rzeczywistego, które zapewnia dodatkowy poziom zabezpieczeń. Urządzeniem tym praktycznie nie trzeba administrować. Jest też bezpieczniejsze niż firewall oparty na systemie UNIX. Wszelkie operacje dostępu są kontrolowane i rejestrowane za pomocą standardowego mechanizmu rejestrowania wydarzeń (*syslog TCP/IP Berkeley UNIX*). PIX gromadzi szereg informacji istotnych z punktu widzenia bezpieczeństwa oraz administrowania siecią.

Oprogramowanie PIX jest skalowalne i łatwe do konfigurowania. Typowa konfiguracja zajmuje około 5 minut. Oprogramowanie to oferuje również wysoką wydajność (maksymalnie ponad 16000 równoczesnych połączeń, translację adresów z prędkością do 45 Mb/s).

Zastosowanie karty Cisco PIX Private Link umożliwia bezpieczną komunikację między wieloma systemami PIX przez Internet z użyciem standardu algorytmu szyfrowania DES (*Data Encryption Standard*). Para takich urządzeń pozwala korzystać z Internetu jako bezpiecznego medium transmisji dla poufnych informacji. Ilustruje to rys. 12. Technicznie jest to realizowane w taki sposób, że pakiet IP po dotarciu do PIX Private Link jest szyfrowany, umieszczony w pakiecie UDP i przesłany przez Internet do bliźniaczego urządzenia. Takie rozwiązanie sprawia, że są szyfrowane także adresy IP komputerów biorących udział w komunikacji, natomiast zastosowanie protokołu UDP nie powoduje nadmiernego wzrostu ilości transmitowanych danych.

Rys. 12. VPN utworzona przy pomocy PIX Private Link

PIX został również wyposażony w mechanizm, zapobiegający przechwytywaniu sesji TCP na podstawie przewidywania numerów sekwencyjnych TCP. Podczas gdy większość dostępnych na rynku pakietów programowych wykorzystujących do transmisji protokół TCP/IP posługuje się zwykłym, liniowym zwiększaniem numerów sekwencyjnych, co upraszcza przewidywanie numerów następnych a tym samym przejmowanie sesji, PIX posługuje się losową generacją tych numerów.

Check Point Firewall-1

Check Point Firewall-1 jest oprogramowaniem umożliwiającym kreowanie własnej polityki bezpieczeństwa dla całych przedsiębiorstw i dowolnie dużych sieci po zainstalowaniu na jednej stacji roboczej. Bazuje na technologii wielowarstwowej inspekcji (*Stateful Multi-Layer Inspection Technology*), charakteryzującej się wysokim stopniem bezpieczeństwa i dużą wydajnością. Oferuje kombinację zabezpieczeń na poziomie sieci i aplikacji użytkowych, gwarantując szczelność zabezpieczeń dla dowolnie dużych organizacji, z jednoczesnym przezroczystym dostępem do zasobów Internetu. Wszystkie przychodzące i wychodzące pakiety danych są sprawdzane pod kątem zgodności z założoną polityką bezpieczeństwa i natychmiast odrzucane w przypadku jej naruszenia. Jak w większości tego typu produktów dostępna jest również translacja adresów IP (*NAT*).

Dzięki dynamicznym mechanizmom autoryzacyjnym na poziomie aplikacji, Check Point Firewall-1 umożliwia realizowanie bezpiecznych połączeń dla ponad 100 wbudowanych serwisów takich jak World Wide Web, FTP, RCP, Mbone i cała rodzina UDP łącznie z *RealAudio* (dźwięk) i VDO (obraz).

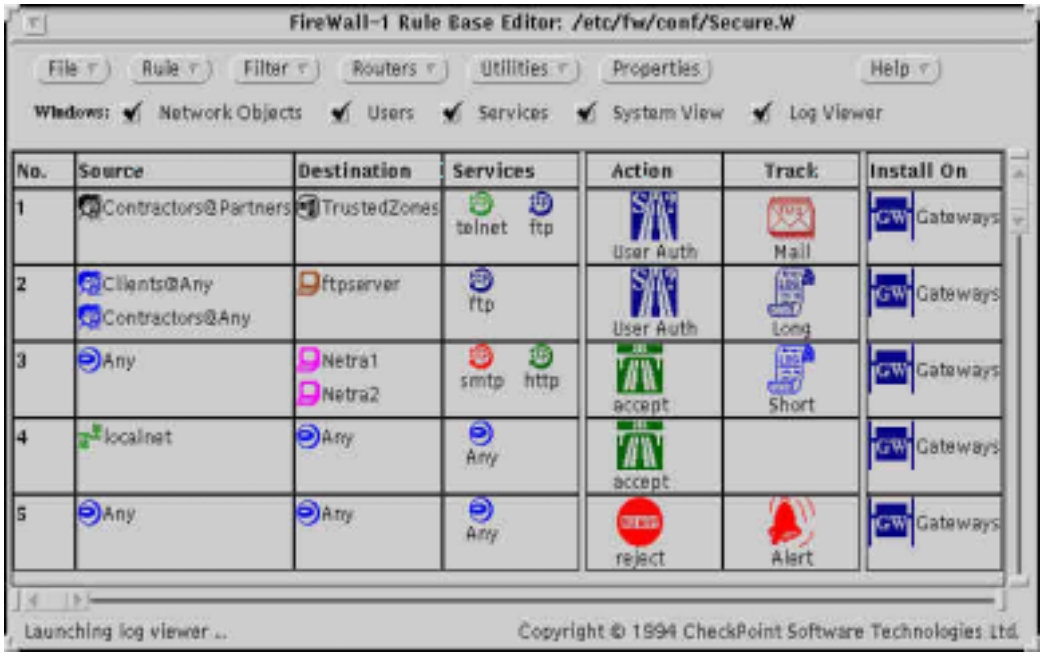
Moduł szyfrujący VPN (*Virtual Private Network*) umożliwia tworzenie wirtualnych, prywatnych i komercyjnych sieci na bazie publicznych sieci rozległych takich jak Internet. Zastosowana metoda Diffie-Hellmana pozwala wybrać kombinację parametrów gwarantujących odpowiednią do potrzeb szybkość działania, efektywność i bezpieczeństwo określonej konfiguracji. Dostępny jest także moduł szyfrujący DES oraz moduł wykorzystujący do przesyłania danych protokół IPSec. Jeszcze jedną metoda szyfrowaniu ruchu TCP/IP umożliwiająca tworzenie wirtualnych sieci prywatnych jest SKIP (*Simple Key Management for IP)*.

Dzięki modułowi *SecuRemote Client* użytkownicy przenośnych stacji Microsoft Windows mają możliwość połączenia bezpośrednio lub poprzez dostawcę usług Internet z siecią korporacyjną w sposób analogiczny do połączenia wewnątrz własnej sieci. Zostało to zrealizowane poprzez rozszerzenie idei VPN na odległe stacje robocze. Poszczególni użytkownicy mogą uzyskać zagwarantowany szyfrowany dostęp do chronionych danych firmy bez potrzeby instalacji oprogramowania Firewall-1 w miejscu swojej pracy. Istnieje także możliwość oferowania dostępu do szyfrowanych danych serwera za opłatą. Istotną cechą klienta PC używającego *SecuRemote* jest brak konieczności zmiany lokalnego środowiska pracy (karta sieciowa i transport TCP/IP pozostają te same) oraz obsługa dynamicznych adresów IP używanych w typowych połączeniach modemowych. Administrator zarządza dostępem takich klientów za pomocą edytora zasad bezpieczeństwa (*Rules Editor*).

Moduł *Client Level Security* obsługuje autoryzację klientów za pomocą specjalnych urządzeń (np. *SecurID*) lub kluczy programowych takich jak hasło systemu UNIX lub wewnętrzne hasło FireWall-*1*. Moduł *User Level Security* gwarantuje chroniony dostęp dla wybranych użytkowników przy użyciu metod analogicznych jak zastosowane w *Client Level Security*. Autoryzacja użytkowników dotyczy indywidualnych kont użytkowych, natomiast autoryzacja klientów dotyczy wszystkich użytkowników na maszynie z wybraną aplikacją.

Check Point FireWall-1 jest dostępny dla platform Solaris Intel, Solaris SPARC, Windows NT i HP-UX. Gwarantuje to zapewnienie pełnej współpracy modułów zainstalowanych na różnych platformach sprzętowo-programowych.

Graficzny Moduł Zarządzający (GUI) zaprojektowano do pracy wg modelu klient-serwer. Dzięki temu modułowi, administrator może zarządzać bazą danych Check Point FireWall z dowolnego węzła sieci - komputera Solaris Intel, SPARC, HP-UX, Windows 95 lub Windows NT. Przykład okna modułu przedstawiono na rys. 13.



Rys. 13. Przykład okna dialogowego Graficzny Moduł Zarządzający *Check Point FireWall-1*

FireWall-1 HTTP Authentication Proxy pozwala na kontrolowane świadczenie usług WWW. Ochrona dostępu dotyczyć może dowolnej liczby serwerów webowych. Dostępna jest również funkcja nadzorowania serwisu WWW. Administrator może określić ogólne zasady korzystania z usługi HTML przez poszczególnych użytkowników. Może np. zezwolić na przeglądanie pewnych serwerów tylko w określonym czasie lub zabronić odwiedzania wybranych lokalizacji.

Dostępny w pakiecie serwer SMTP zastępuje oryginalny serwer UNIXa oferując rozszerzoną kontrolę nad systemem poczty elektronicznej. Administrator ma szereg dodatkowych możliwości takich jak: ukrywanie rzeczywistych adresów pocztowych poszczególnych użytkowników sieci lokalnej, przekierowywanie nadchodzącej poczty, selekcję przychodzących listów, blokowanie załączników do przesyłek pocztowych, odrzucanie listów przekraczających wyznaczony rozmiar.

W ramach diagnostyki antywirusowej Firewall-1 bada pliki kopiowane do sieci lokalnej za pośrednictwem protokołu FTP pod kątem przenoszenia wirusów. Bada również poziom bezpieczeństwa apletów Javy transmitowanych do sieci lokalnej.

Check Point FireWall-1 ma możliwość równoważenia obciążenia serwerów sieciowych. Napływające z zewnątrz zadania są przechwytywane i kierowane do odpowiednich serwerów zgodnie z przyjętym algorytmem. Jest to szczególnie przydatne do regulacji obciążenia lustrzanych serwerów HTTP. Obsługiwane są następujące metody regulacji:

- *Server Load* - zadanie zostaje skierowane do najmniej obciążonego serwera,
- *Round Robin* - wybór serwera odbywa się w sposób cykliczny,
- Domain - zadania są kierowane do serwera, którego nazwa DNS jest najbliższa nazwie komputera inicjującego to zadanie,
- *Load Measuring* - zadania są kierowane do najmniej obciążonego serwera zgodnie ze wskazaniami procedury napisanej przez administratora,
- *Round Trip* - kierowanie zadania są kierowane do komputera o najkrótszym czasie odpowiedzi,
- Random – losowy wybór serwera.

W pakiecie zaimplementowano synchroniczną pracę modułów. Umożliwia to powielanie pracy komputerów zaporowych i wzajemne przejmowanie funkcji w przypadku awarii jednego z nich. Dostępne są również mechanizmy antyspoofingowe.

Gauntlet Firewall

Gauntlet Firewall łączy metody zabezpieczeń typu *firewall-application gateway* z tak ważnymi cechami jak zintegrowane zarządzanie, czy zabezpieczenie transmitowanych informacji. W wersji dla systemu UNIX dostępna jest także usługa wirtualnych sieci prywatnych VPN. Budowa Gauntlet Firewall pozwala firmom realizować politykę bezpieczeństwa umożliwiając dostęp jedynie do tych usług które zostały skonfigurowane przez administratora i które mogą być bezpiecznie użytkowane. Oto lista najważniejszych z nich:

HTTP Proxy	Finger Proxy	RealVideo	Packet Filter
Gopher	SMTP Proxy	Proxy	NAT
SHTTP	POP Proxy	Xing Proxy	SecureID
SSL	Lotus Notes	Netshow Proxy	Integrated VPN
JavaGuard	Proxy	VDOLive Proxy	DES56
ActiveXGuard	SNMP Proxy	Sybase Proxy	PCX Client
KeywordGuard	NNTP Proxy	Oracle Proxy	Authentication
URL screening	LPR Proxy	Logging System	Server
RSH Proxy	Whois Proxy	reports	DNS Hiding
Telnet Proxy	Circuit Proxy	alerts	Content Vector
Rlogin Proxy	RealAudi	scripting	Protocol
Rlogin Proxy	Proxy	SNMP agent	
X11 Proxy			

Zarządzanie pakietem Gauntlet Firewall realizuje się z poziomu dowolnej przeglądarki zdolnej do obsługi appletów języka JAVA. System dostępu do danych czasu rzeczywistego umożliwia współpracę z najpopularniejszymi serwisami multimedialnymi, takimi jak Microsoft Net Show czy VDOlive.

Gauntlet Firewall współpracuje z powszechnie uznanymi programami antywirusowymi. Użytkownik może wybrać dowolne oprogramowanie antywirusowe dostosowane do jego potrzeb i z łatwością skonfigurować Gauntlet Firewall do współpracy z tym oprogramowaniem. Ma to na celu kontrolę antywirusową przychodzących z Internetu plików, wiadomości lub całego generowanego ruchu.

Jako iż technologie takie jak Java lub ActiveX stanowią istotne niebezpieczeństwo dla systemów komputerowych, Gauntlet Firewall może całkowicie zablokować dostęp tego typu appletów do zabezpieczanej sieci. Dodatkowo posiada rozbudowane możliwości filtrowania adresów URL.

W wersji dla systemu UNIX, pakiet zawiera możliwości pracy w standardzie GVPN (*Global Virtual Private Networks*. Dla stworzenia bezpiecznego połączenia wykorzystuje dla szyfrowania przesyłanych informacji algorytm DES z kluczem 56 bitowym.

Gauntlet Firewall może być zarządzany i konfigurowany za pomocą takich systemów zarządzania jak HP OpenView lub CA Unicenter. Dodatkowo oferuje

w chwili obecnej możliwość zarządzania i dostępu do urządzeń SNMP w sieci lokalnej, Intranecie lub innej sieci rozległej bez wystawiania systemu na niebezpieczeństwo.

Omawiany pakiet współpracuje z najważniejszymi systemami potwierdzania autentyczności takimi jak *SecurID* firmy Security Dynamics, *AssureNET Pathways, Radius, S/Key* firmy Bellcore, *CRYPTOCard RB-1*, *Digital Pathways SecurNet Key*, *Digipass*, *Enigma Logics*, *NSA Fortezza*, *Vasco Data Security Access Key II*, *V-ONE SmartCat*. Administrator może udostępnić usługi tylko dla użytkowników którzy potwierdzą swą tożsamość.

Gauntlet Firewall pracuje na platformach: Windows NT, UNIX BSDI/OS (Intel), Solaris, Hewlett Packard HP-UX, Silicon Graphics IRIX.

Raptor Firewall 6.0 dla Windows NT

Raptor Firewall dla Windows NT to system zabezpieczeń łączący cechy filtra pakietowego IP z systemem filtrów aplikacyjnych "*proxy*". Oprogramowanie zawiera szereg mechanizmów bezpieczeństwa, takich jak: *anty-spoofing*, bezpieczne tunelowanie przez Internet dzięki obsłudze protokołów IPSec oraz ISAKMP/Oakley, mechanizmy autoryzacji oraz mechanizmy blokowania analizujące treść przesyłanych informacji: WebNOT i NewsNOT.

Raptor wprowadzony do sprzedaży w 1996 roku był pierwszym systemem firewall dla Windows NT. Raptor oferuje ścisłą integrację z serwisami i mechanizmami autoryzacji Windows NT Wykorzystuje mechanizmy wielowątkowości i wieloprocesorowości systemu operacyjnego. Ukrywa wszystkie systemy i adresy sieci wewnętrznej oraz zakazuje wszystkich połączeń z zewnątrz prócz jawnie dozwolonych. Raptor chroni także przed atakami na poziomie aplikacji, które zazwyczaj nie są wykrywane na poziomie filtrów sieciowych i transportowych. Raptor stosuje bezpieczne filtry aplikacyjne typu *proxy*, analizujące następujące protokoły:

FTP	NTP	RealAudio	Gopher
SMTP (e-mail	SQL*Net	Java	H.323
telnet	HTTP i HTTPS	NNTP (News)	

CIFS/SMB*Raptor* stosuje unikalny algorytm "*best fit*" realizujący dopasowanie reguł dostępu do połączeń sieciowych, co zmniejsza ryzyko popełnienia błędu przez administratora konfigurującego firewall. Czynności administracyjne są realizowane poprzez graficzny interfejs użytkownika (GUI), co znacznie je upraszcza.

Do zalet tego pakietu należy zaliczyć szczegółowy zapis połączeń przechodzących przez firewall. Zapis ten pozwala administratorom szybko analizować zachowanie sieci i reagować na ewentualne nieprawidłowości. Zebrane informacje mogą być też eksportowane do relacyjnej bazy danych w celu przeprowadzenia szczegółowej analizy lub billingu.

ISA Server 2000[4]

ISA Server 2000 (*Internet Security and Acceleration*) jest następcą MS Proxy Servera 2.0. Oprócz realizowanej dotychczas przez ten pakiet funkcji bufora transmisji internetowych, nowy pakiet może pełnić funkcję *firewalla* i serwera VPN. Możliwe jest również budowanie konfiguracji klastrowych wykorzystujących usługi *Network Load Balancing z Windows 2000 Advanced Server.* Łączenie kilku serwerów buforujących w jeden system przyśpiesza dostęp do stron internetowych.

Zapora sieciowa zbudowana w oparciu o ISA Server daje możliwość filtrowania nie tylko na poziomie transmisji pakietów, lecz również poprzez analizę *stateful inspection* oraz technologię filtrów aplikacyjnych. Ostatnia z wymienionych technologii umożliwia analizę kontekstu sesji komunikacyjnej, czyli zawartości pakietów a nie tylko ich nagłówków.

Połączenie VPN może być realizowane w dwóch trybach. W pierwszym, serwer ISA pośredniczy w komunikacji pomiędzy siecią wewnętrzną i zewnętrzną funkcjonując w sposób anonimowy. W drugim możliwe jest bezpośrednie nawiązanie połączeń pomiędzy klientem w sieci wewnętrznej a węzłem w sieci publicznej. Translacja adresów jest realizowana przez moduł *Secure NAT.*

Podstawową zaletą pakietu jest integracja z Windows 2000 i *Active Directory*. Administrator może tworzyć tzw. polisy bezpieczeństwa, określające zasady na jakich

[4] ISA Server 2000 w wersji beta jest dostępny pod adresem http:\\www.microsoft.com\isaserver.

może się odbywać komunikacja poprzez *firewall.* Użytkownik próbujący uzyskać dostęp do Internetu komunikuje się z ISA Serwerem. Ten pyta *Active Directory*, czy dany użytkownik ma prawo korzystać z określonej usługi. Serwer usługi katalogowej odpowiada serwerowi ISA i jeżeli połączenie jest dopuszczalne, to jest ono zestawiane.

W czasie pracy tworzona jest szczegółowa kronika zdarzeń. Administrator może być informowany o zdarzeniach za pośrednictwem poczty elektronicznej. Odpowiednie akcje mogą być również podejmowane w sposób automatyczny. Polegać to będzie zwykle na uruchamianiu odpowiednich skryptów. W pakiecie dostępna również będzie funkcja generowania różnego rodzaju raportów statystycznych, np. o wykorzystaniu stron WWW.

Literatura

[1] V. Ahuja, *Network & Internet Security.* Academic Press 1996 (tłum. MIKOM 1997).

[2] E. Amoroso, *Intrusion Detection : Introduction to Internet Surveillance, Correlation, Traps, Trace Back, and Response*, AT&T Inc., 1999 (tłum. RM 1999).

[3] D. Atkins. *Internet Security: Professional Reference.* New Riders Publishing 1997 (tłum. LT&P 1997).

[4] B. Ball, *Using Linux,* Prentice Hall 1997 (tłum. MIKOM 1999).

[5] B. Chapman, E. Zwicky, *Building Internet Firewalls*, O'Reilly Press, 1996.

[6] D. E. Comer, *Internetworking with TCP/IP, Vol I,* Prentice Hall 1992, (tłum. WNT 1998).

[7] S.Garfinkel, G.Spafford. *Practical Unix and Internet Security.* O'Reilly & Associates 1996 (tłum. RM 1997).

[8] L.Klander. *Hacker Proof*. Jamsa Press, 1997 (tłum. MIKOM 1998).

[9] T.J. Watson, *Address Allocation for Private Internets*, RFC 1597.

[10] K. Egevang, P. Francis, *The IP Network Address Translator (NAT)*, RFC 1631.

[11] M. Grennan, *Firewall and Proxy Server HOWTO*, Sep. 1999.

[12] D. Ranch, *Linux IP Masquerade HOWTO*, Jan. 2000.

[13] P. Russell, *Linux IPCHAINS-HOWTO,* Mar. 1999.

[14] R. Russell, *Linux 2.4 NAT HOWTO,* May 2000.

[15] R. Russell, *Linux 2.4 Packet Filtering HOWTO*, May 2000.

Recenzent: prof. dr hab. inż. Stanisław Paszkowski
Praca wpłynęła do redakcji: 30.06.2000