Omijanie zapór sieciowych

Oliver Karow

Artykuł opublikowany w numerze 1/2006 magazynu *hakin9*. Zapraszamy do lektury całego magazynu.

Magazyn *hakin9*, Software-Wydawnictwo, ul. Piaskowa 3, 01-067 Warszawa, *pl@hakin9.org*

Technika

Omijanie zapór sieciowych

Oliver Karow

stopień trudności

Firewalle są często uznawane za niezawodny sposób ochrony przed nieuprawnionym dostępem. Jednak zapory sieciowe również mają swoje słabości – można je omijać, wykorzystując błędy w konfiguracji albo luki w oprogramowaniu. Intruz może zdobyć dostęp do systemu na wiele różnych sposobów.

Ochrona sieci przed atakami i niechcianym dostępem z niezaufanych sieci, takich jak Internet, to jeden z największych problemów współczesnych systemów informatycznych. W pokonaniu tych trudności pomagają firewalle. Ich podstawowym zadaniem jest oddzielanie sieci i podejmowanie decyzji, czy dany pakiet może zostać przesłany z jednego segmentu do drugiego. Zapory sieciowe można podzielić na kilka rodzajów, w zależności od sposobu realizowania przez nie tych zadań. Dwa najpopularniejsze typy to filtry pakietów wykorzystujące mechanizm rutingu oraz firewalle warstwy aplikacji, korzystające z rozwiązań typu proxy (patrz Ramka *Firewalle – podstawowe infrormacje*).

Niezależnie od rodzaju, firewall potrzebuje pewnych przesłanek, by móc podjąć decyzję czy pakiet będzie przekazany do miejsca przeznaczenia. Jest to tak zwana polityka firewalla w postaci list dostępu lub reguł filtrowania. Przyjrzyjmy się, jak można omijać takie polityki, wykorzystując złe reguły filtrowania, słabości popularnych protokołów oraz ograniczenia różnych typów firewalli.

Wykrywanie firewalli

Zanim system znajdujący się za zaporą zostanie zaatakowany, intruz musi sprawdzić, czy firewall w ogóle istnieje. Nie jest to zawsze tak oczywiste, jak się wydaje – administratorzy firewalli często stosują różne sztuczki, by zapobiec wykryciu zapory. Jednak ponieważ firewall może ingerować w rezultaty ataku, dobrze jest wiedzieć o jego istnieniu. Zajmijmy się najpierw technikami wykorzystywanymi do wykrywania takich rozwiązań.

Z artykułu dowiesz się...

- jak działają firewalle,
- jak wykrywać zapory sieciowe,
- w jaki sposób można omijać firewalle, wykorzystując nieprawidłową konfigurację lub luki w programach.

Co powinieneś wiedzieć...

- powinieneś znać protokoły TCP/IPv4,
- powinieneś znać model referencyjny ISO/OSI.

Firewalle – podstawowe informacje

Ogólnie rzecz ujmując, firewall to system połączony z różnymi sieciami, mający wiele interfejsów i mechanizm filtrowania, który umożliwia przepuszczanie lub blokowanie ruchu między sieciami. Firewalle można podzielić na kategorie według warstw protokołu TCP/IP wykorzystywanych do analizy i przesyłania pakietów.

Filtry pakietów

Filtry pakietów analizują pakiety w warstwie sieci (3) i warstwie transportu (4) modelu ISO/OSI. Oznacza to, że podczas procesu podejmowania decyzji zapory tego typu kierują się następującymi kryteriami:

- protokół (ICMP, OSPF, AH, ESP itp.),
- źródłowy adres IP,
- docelowy adres IP,
- port źródłowy,
- port docelowy,
- flagi TCP (SYN, ACK, RST, FIN itp.).

Stanowe/dynamiczne filtry pakietów

Stanowy filtr pakietów ma więcej możliwości – śledzi każde połączenie i zapisuje te informacje w wewnętrznych tablicach stanów. Kiedy pakiet wychodzący przechodzi przez filtr pakietów (nawiązuje połączenie), porty i adresy IP potrzebne do odebrania pakietów z odpowiedzią są otwierane na czas połączenia, a potem zamykane.

Co więcej, niektóre stanowe filtry pakietów mogą dynamicznie otwierać porty, które zostaną wynegocjowane podczas dozwolonego połączenia klienta i serwera. Korzystają z tego niektóre rozwiązania, takie jak Oracle czy Portmapper.

Firewalle warstwy aplikacji

Firewalle poziomu aplikacji są w stanie analizować pakiety aż do warstwy aplikacji modelu ISO/OSI. Poza tym, że mają funkcjonalność filtrów stanowych/dynamicznych, mogą też badać ładunek pakietu. O ile filtr pakietów podejmuje decyzje wyłącznie na podstawie nagłówków, o tyle firewall poziomu aplikacji może badać informacje związane z określonymi aplikacjami. Umożliwia to na przykład przepuszczanie całego ruchu HTTP na porcie 80 TCP z wyjątkiem żądań typu `CONNECT` czy `DELETE`.

Firewalle poziomu aplikacji wymagają uruchomionej specjalnej usługi proxy dla każdego protokołu monitorowanego przez zaporę. Usługi proxy nie zawsze są dostępne, więc większość producentów firewalli dodatkowo implementuje możliwości filtra pakietów i podstawową funkcjonalność proxy bez zdolności analizy protokołu.

Firewalle hybrydowe i warstwy 2

Wielu producentów stosuje technikę hybrydową, by połączyć ze sobą najlepsze cechy wszystkich firewalli, czyli łączy stanowe filtry pakietów z zaporami warstwy aplikacji. Na rynku dostępne są także firewalle warstwy 2. Nie są tak popularne jak filtry pakietów i zapory warstwy aplikacji, są zwykle stosowane na poziomie interfejsu, zależnie od producenta.

Listing 1. *Traceroute zablokowane przez firewall*

```
# traceroute www.dummycompany.de
traceroute to www.dummycompany.de (10.10.10.10), 30 hops max, 40 byte packets
 1  10.255.255.254                    0.373 ms  0.203 ms  0.215 ms
(...)
10  router.company1.de (10.1.1.254)  88.080 ms  88.319 ms  87.921 ms
11  router.company2.de (10.2.2.254)  87.881 ms  89.541 ms  88.081 ms
12  router.company3.de (10.3.3.254)  86.749 ms  86.919 ms  86.734 ms
13  router.company4.de (10.4.4.254)  87.216 ms  87.312 ms  87.307 ms
14  * * *
```

Traceroute

Śledzenie tras (*traceroutingu*) to mechanizm stosowany do wykrywania ruterów przekazujących pakiety na drodze do ich celu. Jeśli po drodze znajduje się firewall, może odpowiedzieć na pakiet traceroute.

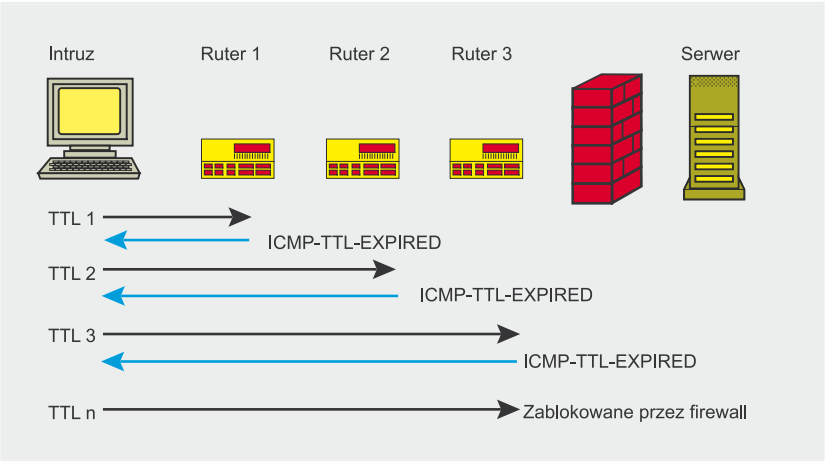
Ponieważ tracerouting to bardzo stara technika, większość firewalli ją blokuje. Wciąż jednak istnieje kilka nieporozumień związanych z funkcjonowaniem tego rozwiązania, co pozwala intruzom przedostać się przez zabezpieczenia.

Listing 1 przedstawia efekt działania polecenia *traceroute* po zablokowaniu go przez firewall. Jak widać, program ten działa, dopóki nie dotrze do systemu o adresie 10.4.4.254. Na tym hoście znajduje się coś, co blokuje traceroute.

Spróbujmy zrozumieć jak działa śledzenie tras (patrz też Rysunek 1). U celu określenia trasy pakietu IP pole TTL nagłówka IP jest wykorzystywane w taki sposób, że zostaje zmniejszone o 1 za każdym razem, gdy dociera do rutera. Jeżeli ruter otrzyma pakiet z TTL o wartości 2, zmniejszy tę wartość o 1 i jeśli otrzymany TTL jest równy lub większy od 1, pakiet zostaje przekazany do następnego rutera – zgodnie z danymi o rutingu. Natomiast jeżeli ruter otrzyma pakiet z TTL o wartości 1, zmniejszy ją i w efekcie – ponieważ otrzymana wartość będzie równa 0 – nie przekaże pakietu do następnego rutera. Zamiast tego wyśle nadawcy powiadomienie, że pakiet został odrzucony w drodze do celu.

Program traceroute rozpoczyna działanie od wysłania pierwszego pakietu z TTL równym 1, więc otrzymuje komunikat ICMP o wygaśnięciu TTL (*TTL-expired*). Następnie zwiększa TTL do 2, by przejść przez pierwszy ruter i otrzymuje taki sam komunikat od drugiego rutera na trasie. Kontynuuje ten proces do momentu dotarcia do celu. Ponieważ każdy ruter wysyła taki komunikat (o ile nie jest skonfigurowany inaczej), traceroute jest w stanie stworzyć listę ruterów.

Warto również wiedzieć, że istnieją dwie różne implementacje pro-



Rysunek 1. *Zasada działania traceroutingu*

Listing 2. *Śledzenie tras za pomocą pakietów TCP przy użyciu hping2*

```
# hping2 -T -t 1 -S -p 80 www.dummycompany.de
HPING www.dummycompany.de (eth0 10.10.10.10 ): S set, ↩
  40 headers + 0 data bytes
hop=1 TTL 0 during transit from ip=10.255.255.254 name=UNKNOWN
hop=1 hoprtt=12.4 ms
(...)
hop=10 TTL 0 during transit from ip=10.1.1.254 name=router.company1.de
hop=11 TTL 0 during transit from ip=10.2.2.254 name=router.company2.de
hop=12 TTL 0 during transit from ip=10.3.3.254 name=router.company3.de
hop=13 TTL 0 during transit from ip=10.4.4.254 name=router.company4.de
hop=14 TTL 0 during transit from ip=10.5.5.254 name=UNKNOWN
len=46 ip=10.10.10.10 flags=SA DF seq=15 ttl=107 id=12852 win=29200 rtt=95.6 ms
len=46 ip=10.10.10.10 flags=R DF seq=15 ttl=107 id=12856 win=0 rtt=194.6 ms
```

Listing 3. *Wysyłanie pakietu na zamknięty port*

```
# hping2 -S -p 99 -c 1 www.dontexist.com
HPING www.dontexist.com (eth0 192.168.10.10): S set, ↩
  40 headers + 0 data bytes
ICMP Packet filtered from ip=192.168.9.254
```

Listing 4. *Obserwacja ruchu sieciowego*

```
# tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
12:59:18.778417 IP 172.16.1.1.1866 > 192.168.10.10.99: ↩
  S 1958445360:1958445360(0) win 512
12:59:18.786914 IP 192.168.9.254 > 172.16.1.1 icmp 36: ↩
  host 192.168.10.10 unreachable - admin prohibited filter
```

gramu traceroute. Pierwsza używa pakietów ICMP *echo request* (na przykład *tracert* w systemach Windows), zaś druga – pakietów UDP (większość implementacji uniksowych). Oba warianty wykorzystują technikę opartą na polach TTL. Administrator firewalla musi więc pamiętać, by odfiltrować obie implementacje traceroute.

TCP traceroute

Ponieważ wiemy, że pole TTL jest częścią nagłówka IP i że popularne filtry traceroute blokują jedynie pakiety UDP i ICMP, możemy spróbować ominąć te filtry, wykorzystując pakiety TCP. Prześledźmy jeszcze raz trasę do docelowego rutera. Tym razem skorzystamy z narzędzia *hping2*, które umożliwia wysyłanie spreparowanych pakietów (patrz Listing 2). Jak widać, rozpoznaliśmy jeszcze jeden odcinek trasy (*hop*). O ile polecenie *traceroute* zostało zablokowane przy 13. ruterze, o tyle *hping2* dało nam dodatkowe wyniki.

Analiza pakietów zwrotnych

Aby zbadać, czy firewall istnieje, można porównać pakiety zwrotne z otwartych portów z tymi pochodzącymi z portów zamkniętych. Przeanalizujmy kilka sztuczek, które mogą ułatwić ten proces.

Użyjmy najpierw *hping2* do wysłania pakietu do naszego celu, na port, który możemy uznać za zamknięty (patrz Listing 3). Jednocześnie spróbujmy obserwować ruch sieciowy za pomocą narzędzia *tcpdump* (Listing 4). Zobaczymy komunikat ICMP *destination unreachable* w postaci wiadomości filtra *admin prohibited* z adresu 192.168.9.254. Wiadomość świadczy o tym, że dostęp do portu 99 TCP systemu docelowego jest filtrowany za pomocą listy dostępu rutera. Ponieważ jest to oczywisty dowód na istnienie firewalla, zajmijmy się inną techniką, opartą na analizie wartości TTL.

Różnice TTL

Za każdym razem, gdy pakiet IP przechodzi przez urządzenie rutujące, jego TTL zostaje zmniejszony o 1. Jeśli więc mamy serwer chroniony przez zaporę sieciową zainstalowaną na wydzielonym systemie, pakiety pochodzące z serwera mogą mieć inny TTL niż pakiety pochodzące z tego firewalla.

Teraz musimy spróbować otrzymać pakiet odpowiadający zarówno z serwera, jak i potencjalnego systemu z firewallem, a następnie porównać wartości TTL obu pakietów. Jeśli wartości te będą się różnić, będzie to prawdopodobnie świadczyło o istnieniu firewalla.

Aby zmusić oba systemy do odpowiedzi, możemy wysłać jeden pakiet na port otwarty, a drugi na zamknięty port docelowego systemu – w naszym przypadku odpowiednio 80 TCP i 99 TCP (patrz Listing 5). Jak widać, wartości TTL różnią się o 1. Oznacza to, że znaleźliśmy firewall chroniący docelowy serwer.

Listing 5. *Porównanie wartości TTL*

```
# hping2 -S -p 80 -c 1 www.randomname.com
HPING www.randomname.com (eth0 192.100.100.10): ←
  S set, 40 headers + 0 data bytes
len=46 ip=192.100.100.10 flags=SA DF seq=0 ttl=55 id=0 win=5840 rtt=7.6 ms
# hping2 -S -p 99 -c 1 www.randomname.com
HPING www.randomname.com (eth0 192.100.100.10): ←
  S set, 40 headers + 0 data bytes
len=46 ip=192.100.100.10 flags=RA DF seq=0 ttl=56 id=0 win=0 rtt=7.6 ms
```

Określanie typu firewalla

Powyższe techniki pozwalają zdobyć dowód na istnienie firewalla. Jeśli uda nam się określić adres IP tego urządzenia, za pomocą sztuczek będziemy mogli zdobyć dodatkowe informacje, takie jak typ zapory i użyty system operacyjny.

TCP fingerprinting

Wykorzystajmy fakt, że każdy stos IP w systemie posługuje się określonymi wzorcami, które można wykorzystać do określenia typu i wersji systemu operacyjnego. Ponieważ większość programowych firewalli wpływa na zachowanie stosu IP, często można także określić typ i wersję zainstalowanej zapory. Użyjemy rzecz jasna narzędzia *nmap*, mającego wbudowaną funkcję wykrywania systemu operacyjnego (patrz Listing 6). Wystarczy przeskanować trzy porty, by z dużym prawdopodobieństwem określić, że nasza zapora to Checkpoint Firewall-1 NG uruchomiona w systemie Solaris.

Listing 6. *Wykrywanie systemu operacyjnego i firewalla za pomocą narzędzia nmap*

```
# nmap -sS -F -n -O  -p 80,99,443 192.168.190.1
Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) ←
  at 2005-10-09 17:23 CEST
Interesting ports on 192.168.190.1:
PORT    STATE SERVICE
80/tcp  open  http
99/tcp closed metagram
443/tcp open  https
Device type: firewall|broadband router|general purpose
Running: Checkpoint Solaris 8, Belkin embedded, Sun Solaris 8
OS details: Checkpoint Firewall-1 NG on Sun Solaris 8, ←
  Belkin DSL/Cable Router, Sun Solaris 8, Sun Trusted Solaris 8
```

Zajmijmy się innym firewallem (Listing 7) – Symantec Enterprise Firewall. Jak widać, *nmap* nie był w stanie określić systemu operacyjnego i typu zapory, ale mnogość otwartych portów może wskazywać na to, że jest to firewall oparty na proxy. Producentów takich rozwiązań jest zaś niewielu.

Listing 7. *Fingerprinting zapory Symantec Enterprise Firewall*

```
Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) ←
  at 2005-10-10 13:43 CEST
Interesting ports on 192.168.99.1:
(The 1193 ports scanned but not shown below are in state: closed)
PORT     STATE    SERVICE
21/tcp   open     ftp
22/tcp   open     ssh
23/tcp   open     telnet
25/tcp   open     smtp
53/tcp   open     domain
80/tcp   open     http
119/tcp  open     nntp
139/tcp  open     netbios-ssn
443/tcp  open     https
481/tcp  open     dvs
512/tcp  open     exec
513/tcp  open     login
514/tcp  open     shell
554/tcp  open     rtsp
1720/tcp open     H.323/Q.931
2456/tcp open     unknown
5631/tcp open     pcanywheredata
7070/tcp open     realserver
No exact OS matches for host (If you know what OS is running ←
  on it, see http://www.insecure.org/cgi-bin/nmap-submit.cgi).
```

W tej sytuacji, poza fingerprintingiem za pomocą narzędzia *nmap*, warto przyjrzeć się portom typowym dla różnych producentów zapór. Na przykład wspomniany Symantec Enterprise Firewall (SEF) używa dwóch charakterystycznych portów, 2456 TCP do administracji przez WWW i 888 TCP do uwierzytelniania *Out of Band*. Porównanie wyników skanowania z Tabelą 1 przybliży nas nieco do określenia typu firewalla (przy okazji – dobrze skonfigurowany firewall warstwy aplikacji nie będzie miał tylu otwartych zewnętrznych portów). Checkpoint Firewall-1 także korzysta z charakterystycznych portów, między innymi administracyjnych 256–264 TCP oraz 18180–18265 TCP.

Listing 8. *Sprawdzanie banerów*

```
# netcat www.raptorfirewall.nix 80
> HEAD / HTTP/1.0
< HTTP/1.1 503 Service Unavailable
< MIME-Version: 1.0
< Server: Simple, Secure Web Server 1.1
< Date: Fri, 17 Sep 2004 19:08:35 GMT
< Connection: close
< Content-Type: text/html
< <HTML>
< <HEAD><TITLE>Firewall Error: Service Unavailable</TITLE></HEAD>
```

Listing 9. *Normalne skanowanie i skanowanie z portów źródłowych*

```
# nmap -sS -p 1-65535 192.168.0.1
Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) ←
  at 2005-10-09 17:01 CEST
Interesting ports on 192.168.0.1:
(The 1658 ports scanned but not shown below are in state: closed)
PORT   STATE SERVICE
80/tcp open  http
Nmap run completed -- 1 IP address (1 host up) scanned in 6.607 seconds

# nmap -sS -g 80 -p 1024-65535 192.168.0.1
Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) ←
  at 2005-10-09 17:01 CEST
Interesting ports on 192.168.0.1:
(The 1657 ports scanned but not shown below are in state: closed)
PORT   STATE SERVICE
80/tcp open  http
6000/tcp open X11
Nmap run completed -- 1 IP address (1 host up) scanned in 6.607 seconds
```

Tabela 1. *Otwarte porty, które mogą pomóc określić typ firewalla*

Firewall	Numer portu	Przeznaczenie
Symantec Enterprise Firewall	888/TCP	Demon OOB
Symantec Enterprise Firewall	2456/TCP	Administracja przez WWW
Checkpoint FW1-NG	256/TCP	Zarządzanie
Checkpoint FW1-NG	257/TCP	FW1_log
Checkpoint FW1-NG	18181/TCP	OPSEC, protokół *Content Vectoring Protocol*
Checkpoint FW1-NG	18190/TCP	Interfejs administracyjny

Warto także wiedzieć, że *nmap* nie jest jedynym narzędziem, które można wykorzystać do skanowania firewalli – pomocne są także narzędzia takie jak *xprobe* czy *p0f*. Więcej przydatnych informacji o fingerprintingu opublikowaliśmy w Artykule *OS fingerprinting – jak nie dać się rozpoznać*, *hakin9* 4/2004.

Sprawdzanie banerów

Aby zdobyć więcej danych i upewnić się, z jakim firewallem mamy do czynienia, możemy wykorzystać technikę sprawdzania banerów. Na przykład łańcuch tekstu *Server: Simple, Secure Web Server 1.1* pozwoli nam z łatwością zidentyfikować demon HTTP jako część zapory Symantec Personal Firewall (Listing 8).

Trzeba pamiętać, że takie dane nie są zbyt wiarygodne – w przypadku większości demonów można je łatwo zmienić. Jednak w połączeniu z informacjami uzyskanymi podczas fingerprintingu TCP i numerami otwartych portów stają się cenną przesłanką ułatwiającą identyfikację typu zapory.

Omijanie firewalli

Kiedy intruz pozna już typ zastosowanego firewalla, ma kilka możliwości, by go ominąć. Przyjrzyjmy się sposobom zdobywania nieuprawnionego dostępu do systemów chronionych zaporą, takim jak wykorzystywanie źle skonfigurowanych list dostępu, słabości protokołów czy błędów w oprogramowaniu.

Ataki portu źródłowego

Zacznijmy od prostych filtrów pakietów. Podejmują one decyzje, analizując nagłówki IP lub TCP/UDP każde-

Tabela 2. *Minimalna lista dostępu dla ruchu HTTP*

L.p.	Źródłowy adres IP	Docelowy adres IP	Port źródłowy	Port docelowy	Działanie	Opis
1	Wewnętrzny	Zewnętrzny	>1024 TCP	80 TCP	Zezwól	Zezwolenie na żądania HTTP pochodzące od klienta
2	Zewnętrzny	Wewnętrzny	80 TCP	>1024 TCP	Zezwól	Zezwolenie na odpowiedź serwera na żądanie HTTP
3	Dowolny	Dowolny	Dowolny	Dowolny	Odrzuć	Reguła czyszcząca

Tryby FTP: aktywny i pasywny

Protokół FTP wykorzystuje do komunikacji dwa kanały pomiędzy klientem a serwerem. Kanał komend jest używany do wysyłania poleceń serwerowi i odpowiedzi do klienta. Jeżeli przesyłane są dane (wysyłanie lub pobieranie pliku, pobieranie listy zawartości katalogu), zostaje ustanowiony dodatkowy kanał danych. Protokół FTP stosuje dwa tryby nawiązywania kanału danych – aktywny i pasywny. Różnica między nimi polega na tym, która strona inicjuje ten typ połączenia.

W przypadku trybu aktywnego serwer FTP łączy się z klientem. Następnie klient FTP informuje serwer za pomocą komendy `PORT`, pod jakim adresem IP i na jakim porcie będzie nasłuchiwać połączeń z serwera. Natomiast w trybie pasywnym to właśnie klient FTP łączy się z serwerem, a potem serwer musi poinformować klienta, jaki adres i port umożliwia nawiązanie kanału danych.

W celu wejścia w tryb pasywny klient FTP musi wysłać komendę `PASV`. W odpowiedzi serwer nadeśle klientowi informację o gniazdach w formacie `IP,IP,IP,IP,Hbyte,Lbyte`, gdzie *Hbyte* i *Lbyte* to porty, z którymi należy się połączyć. Natomiast oktety adresu IP są oddzielone przecinkami, nie kropkami (patrz też Rysunki 2 i 3).

go pakietu. Zwykle sprawdzają źródłowy i docelowy adres IP oraz docelowy i źródłowy port w celu podjęcia decyzji o przepuszczeniu lub zablokowaniu pakietów.

Aby stworzyć prostą regułę dostępu pozwalającą użytkownikom wewnętrznej sieci surfować po Internecie (czyli sieci zewnętrznej), potrzebujemy dwóch reguł – jednej dla pakietów wychodzących (żądań HTTP) i drugiej dla przychodzących (odpowiedzi z serwera). Stworzenie odpowiedniej reguły wymaga jedynie wiedzy o tym, że serwer WWW nasłuchuje domyślnie na porcie 80 TCP, zaś numeru portu źródłowego wybieranego przez klienta HTTP (przeglądarkę) nie da się przewidzieć, choć zazwyczaj jest on wyższy niż 1024. Tabela 2 prezentuje minimalną listę dostępu dla takiej sytuacji.

Na pierwszy rzut oka ten zestaw reguł nie wygląda niebezpiecznie. Reguła 1 zezwala na wychodzące żądania HTTP, zaś reguła 2 zezwala na pakiety odpowiadające. Trzecia reguła służy do blokowania pozostałego ruchu, więc została nazwana regułą czyszczącą. Jednak jeśli przyjrzymy się bliżej regule 2, zobaczymy, że pakiet pochodzący z Internetu, skierowany do sieci wewnętrznej, nadchodzący z portu źródłowego 80 i przychodzący na port wyższy niż 1024, przedostanie się przez filtr pakietów.

Taka technika nosi nazwę *ataku wysokiego portu* (*high port attack*) lub *ataku portu źródłowego* (*source port attack*). Wykorzystuje ona bowiem fakt, że atakujący musi tylko zmodyfikować swojego klienta do wykorzystywania znanego portu, na przykład 80 TCP, jako portu źródłowego. Dzięki temu będzie mógł zaatakować usługi za firewallem nasłuchujące na wysokich portach. Niektóre z takich usług to serwer X-Window (6000-6063/TCP), Windows Terminal Server (3389 TCP) oraz popularne porty aplikacji webowych, takich jak Jakarta Tomcat (8080 TCP) czy BEA WebLogic (7001 TCP).

Do sprawdzenia, czy nasz firewall jest podatny na ten atak możemy użyć narzędzia *nmap* z opcją `-g`, co pozwoli nam zdefiniować port źródłowy. Listing 9 prezentuje różnicę między skanowaniem normalnym a wykorzystującym port źródłowy.

Jak widać, za pomocą tej prostej techniki odkryliśmy jeszcze jeden otwarty port (6000 TCP). Jednak intruz nie będzie w stanie połączyć się z tym portem, chyba że port źródłowy klienta będzie ustawiony na 80 TCP.

Najprostszą metodą nawiązania połączenia za pomocą portu źródłowego jest użycie programu Fpipe firmy Foundstone. Jest to narzędzie dla Windows, lecz działa także w Linuksie (pod Wine). Uruchomienie go z następującymi opcjami:

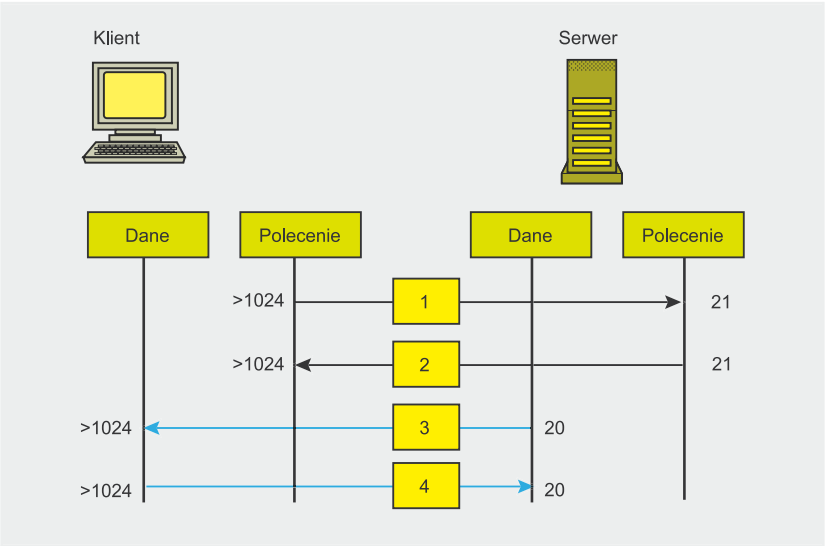
```
> FPipe -l 100 -s 80 ←↩
  -r 6000 192.168.0.1
```

otworzy demona nasłuchującego na lokalnym porcie 100. Wszystkie pakiety wysłane na ten port będą mieć w nagłówku port źródłowy 80 i zostaną przekierowane na adres 192.168.0.1:6000.

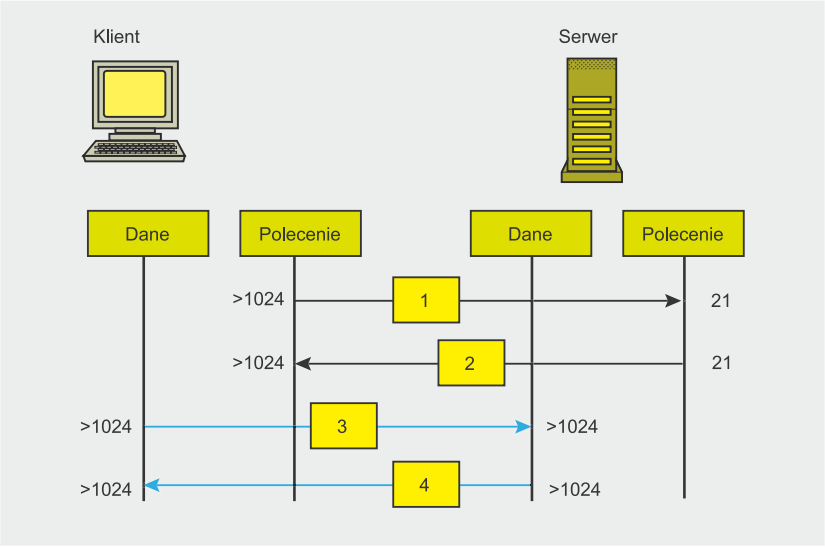
Jeśli testujemy podatność firewalla na ataki z portów źródłowych, powinniśmy także przeprowadzić próby przy użyciu portów 53 (DNS), 21 (FTP) i 88 (Kerberos) – przeznaczenie tych protokołów sprawia, że związanie z nimi reguły części zapór są bardzo słabe. Na przykład Check-

Tabela 3. *Zestaw reguł firewalla stanowego dotyczących ruchu HTTP*

L.p.	Źródłowy adres IP	Docelowy adres IP	Port źródłowy	Port docelowy	Flaga TCP	Działanie	Opis
1	Wewnętrzny	Zewnętrzny	>1024 TCP	80 TCP	SYN	Zezwól	Zezwolenie na wychodzący od klienta ruch HTTP
2	Zewnętrzny	Wewnętrzny	80 TCP	>1024 TCP	!SYN	Zezwól	Zezwolenie serwerowi na odpowiedzi na żądania HTTP
3	Dowolny	Dowolny	Dowolny	Dowolny	Dowolna	Odrzuć	Reguła czyszcząca



Rysunek 2. *Działanie aktywnego trybu FTP*



Rysunek 3. *Działanie pasywnego trybu FTP*

point FW1 do wersji 4.1 używał sugerowanych reguł (*Implied rules*), zezwalających na cały ruch DNS w dowolnym kierunku.

Implementacja filtra IPSec firmy Microsoft, którą można skonfigurować jako lokalny firewall, ma podobną lukę. Wbudowana, niewidoczna reguła zezwala na cały ruch pochodzący ze źródłowego portu 88 (Kerberos). Zapobieganie temu atakowi wymaga zmian w rejestrze systemowym.

Firewalle stanowe

Aby uniemożliwić atakującemu nawiązywanie połączeń z wewnętrznymi systemami przez symulowanie odpowiedzi na wcześniejsze żądania, firewall musi potrafić rozróżniać pakiety odpowiadające od pakietów nawiązujących nowe połączenie. Zapora może w tym celu sprawdzać różne flagi wewnątrz nagłówków TCP. Ponieważ każda nowa sesja TCP/IP rozpoczyna się od ustawionej flagi SYN, a wszystkie kolejne pakiety mają ustawioną przynajmniej flagę ACK, jest to znaczne ułatwienie dla firewalla. W dodatku wewnętrzna tablica stanów pomaga śledzić sesje, szczególnie w przypadku komunikacji UDP.

Jak możemy zobaczyć w Tabeli 3, odpowiedź serwera HTTP zostanie przepuszczona tylko wtedy, jeśli nagłówek TCP nie ma ustawionej flagi SYN (negacja wyrażona za pomocą znaku wykrzyknika). W takim wypadku atak z portu źródłowego już nie zadziała (chyba że ktoś używający iptables zapomni ustawić pozycji !SYN w regule), a intruz będzie musiał poszukać innej techniki.

Wykorzystywanie aktywnego trybu FTP

Jednym z najczęściej używanych w Internecie protokołów jest FTP (*File Transfer Protocol*). Proto-

Tabela 4. *Zestaw reguł zezwalający na aktywne połączenia FTP*

L.p.	Źródłowy adres IP	Docelowy adres IP	Port źródłowy	Port docelowy	Flaga TCP	Działanie	Opis
1	Wewnętrzny	Zewnętrzny	>1024 TCP	21 TCP	SYN	Zezwól	Kanał komend
2	Zewnętrzny	Wewnętrzny	21 TCP	>1024 TCP	!SYN	Zezwól	Kanał komend
3	Zewnętrzny	Wewnętrzny	20 TCP	>1024 TCP	SYN	Zezwól	Kanał danych
4	Wewnętrzny	Zewnętrzny	>1024 TCP	20 TCP	!SYN	Zezwól	Kanał danych
5	Dowolny	Dowolny	Dowolny	Dowolny	Dowolna	Odrzuć	Reguła czyszcząca

kół ten może działać na dwa różne sposoby – w trybie aktywnym lub pasywnym (patrz Ramka *Tryby FTP: aktywny i pasywny*). Najważniejszą różnicą między nimi jest sposób nawiązywania połączeń. W trybie aktywnym klient FTP tworzy kanał komend, zaś serwer tworzy kanał danych. W trybie pasywnym oba kanały są ustalane przez klienta FTP.

Atak na aktywny tryb FTP jest bardzo podobny do ataku na porty źródłowe. Jednak w tym wypadku tryb aktywny wymusza na firewallu akceptację pakietów przychodzących z ustawioną flagą SYN dla kanału danych (w Tabeli 4 znajduje się przykładowy zestaw reguł). Oznacza to, że nawet jeśli zapora sprawdza flagi SYN, nie uchroni to przed intruzem próbującym nawiązać połączenie na porty wyższe niż 1024 ze źródłowego portu 20.

Aby sprawdzić, czy firewall jest podatny na ten typ ataku, możemy użyć programu *nmap* – tym razem z opcją `-g 20`, zamiast `-g 80` jak w poprzednim przypadku. Do zmodyfikowania portu źródłowego w celu nawiązania połączenia z wewnętrzną usługą na wysokim porcie można użyć programu Fpipe.

Wykorzystanie pasywnego FTP

Większość współczesnych serwerów FTP umożliwia stosowanie trybu pasywnego, niestety nie można tego powiedzieć o wielu klientach (choćby domyślnym kliencie FTP firmy Microsoft). Jednak nawet korzystanie z pasywnego FTP może nie wystarczyć do ochrony systemu przed niechcianym dostępem do wewnętrznych komputerów. Zajmijmy się więc komunikacją FTP w trybie pasywnym – aby zwiększyć czytelność przykładów, do nawiązywania połączeń użyjemy narzędzia netcat (patrz Listing 10).

Pierwsze sześć linii to standardowa komunikacja FTP dotycząca łączenia i logowania do serwera. W siódmej linii serwer FTP jest informowany o wykorzystaniu pasywnego trybu do transferu danych. W odpowiedzi serwer (linia ósma) wskazuje klientowi, który adres IP i port będzie akceptować połączenie w kanale danych.

Listing 10. *Komunikacja w trybie pasywnym FTP*

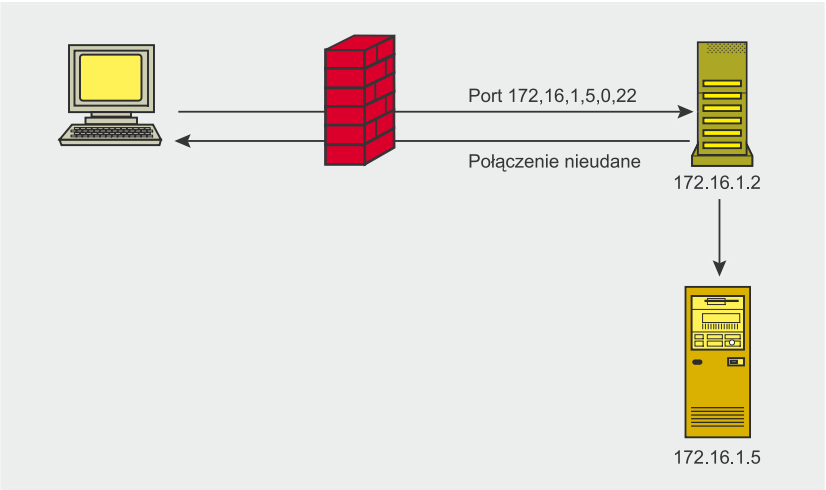
```
# nc ftp.hakin9.org 21
< 220-Welcome to hakin9.org.
> user anonymous
< 331 Please specify the password.
> pass secret
< 230 Login successful.
> pasv
< 227 Entering Passive Mode (192,168,200,23,230,242)
```

Listing 11. *Otwarcie porty za pomocą pasywnego trybu FTP*

```
# nc ftp.hakin9.org 21
220-Welcome to hakin9.org.
user anonymous
331 Please specify the password.
pass secret
230 Login successful.
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA227 ←
  Entering Passive Mode (192,168,200,23,0,2)
500 command not understood: ←
  'AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA227  ←
  Entering Passive Mode (192,168,200,23,0,22)'
```

Chroniąca serwer FTP zapora nie ma informacji wystarczających do określenia, który port zostanie wybrany na kanał danych przez serwer. Istnieją dwie możliwości modyfikacji reguł w celu zezwolenia na komunikację:

- Otwarcie wszystkich wysokich portów dla połączeń przychodzących do serwera FTP. Jest to bardzo niebezpieczne, szczególnie w przypadku istnienia wielu serwerów w jednej sieci, więc to rozwiązanie nie jest zalecane.
- Analiza komunikacji między klientem a serwerem. Jeżeli firewall zarejestruje w kanale komend polecenie w postaci `227 Entering Passive Mode (IP,IP,IP,IP,Hbyte,Lbyte)` wysłane z serwera do klienta, stworzy tymczasową regułę zezwalającą na połączenie przychodzące na adres IP i port zdefiniowane w komendzie.



Rysunek 4. *Skanowanie FTP bounce*

Przy takiej konfiguracji można oszukać zaporę, zmuszając ją do otwarcia wybranego portu. Ponieważ parametr w formacie `IP,IP,IP,IP,Hbyte,Lbyte` jest wysyłany przez kanał komend od serwera do klienta, intruz może zmusić serwer FTP do wysłania spreparowanego komunikatu. Można to zrobić przez wymuszenie komunikatu o błędzie zawierającego łańcuch dotyczący połączenia pasywnego.

Jeżeli do serwera FTP zostanie wysłana nieistniejąca komenda, w niektórych przypadkach zwróci on komunikat o błędzie zawierający wysłaną komendę, na przykład *cannot understand command AAAAAAAAAA227 Entering Passive Mode 1,2,3,4,0,22*. Jeśli więc obliczymy rozmiar komunikatu o błędzie w taki sposób, by nie mieścił się w jednym pakiecie IP, a nieistniejąca komenda znajdzie się w osobnym pakiecie (a ciąg komendy związanej z połączeniem pasywnym znajdzie się w następnym pakiecie), być może uda się otworzyć dodatkowy port na firewallu.

Gdy firewall odczyta pierwszy pakiet zawierający znaki `A`, po prostu przekaże pakiet. Ale jeżeli odczyta też łańcuch `227 Entering Passive Mode (192,168,200,23,0,22)`, stworzy tymczasową regułę zezwalającą na połączenie klienta FTP z portem 22 serwera 192.168.200.23. Podobny mechanizm tworzenia dynamicznych filtrów może być też wykorzystywany w innych protokołach, takich jak sqlnet Oracle.

Skanowanie FTP bounce

Skanowanie FTP *bounce* (patrz Rysunek 4) wykorzystuje funkcje aktywnego FTP do skanowania systemów za firewallem. Serwer tworzy kanał danych, nawiązując połączenie z otwartym portem klienta FTP. Ponieważ serwer nie jest w stanie rozpoznać portu wykorzystywanego przez klienta do transferu danych, dostarczenie tych informacji za pośrednictwem kanału komend staje się obowiązkiem wspomnianego klienta.

Operację tę wykonuje się za pomocą komendy `PORT`. Składnia jest następująca: `IP,IP,IP,IP,Hbyte,Lbyte`, na przykład `PORT 192,168,100,10,0,123`. Serwer jest wtedy w stanie nawiązać połączenie z adresem 192.168.100.10:123.

Zrozumiałe jest, że adres IP nie musi się ograniczać do adresu klienta – niektóre serwery FTP zezwalają na użycie dowolnego adresu. Po wydaniu polecenia w rodzaju `dir` serwer próbuje połączyć się z określonym adresem IP i portem. W zależności od tego, czy port jest zamknięty czy otwarty, serwer zwróci kod błędu lub kod sukcesu. Analiza kodu statusu umożliwia atakującemu sprawdzenie stanu portu. Program *nmap* umożliwia skanowanie FTP *bounce*:

```
$ nmap -b \
  anonymous@myftpserver:21 \
  targetserver
```

HTTP proxy bouncing

Programowe firewalle często filtrują ruch HTTP, działając na zasadzie proxy HTTP, transparentnego lub nie. Problem z takim proxy polega na tym, że jeśli jest źle skonfigurowane, może dawać dostęp do wewnętrznych serwerów.

Najłatwiejszym sposobem przetestowania zapory pod kątem podatności na atak *proxy bouncing* jest ustawienie zewnętrznego interfejsu firewalla jako proxy HTTP i próba przeglądania stron na lokalnych serwerach WWW.

Ustawienie proxy dla przeglądarki Lynx wygląda następująco:

```
# http_proxy='mojfirewall.pl:8080'
# no_proxy='localhost'
# export http_proxy no_proxy
```

Przeglądanie lokalnych stron WWW jest już proste:

```
# lynx 192.168.100.20
```

Ciekawą cechą tej techniki jest fakt, że za jej pomocą można dostać się z zewnątrz nawet do prywatnych adresów IP – atakujący łączy się tylko z oficjalnym adresem firewalla i wysyła do demona HTTP żądanie połączenia z celem. Ponieważ demon zna również wewnętrzne, prywatne adresy IP, może się z nimi połączyć.

Warto również spróbować zdobyć dostęp do różnych portów wewnętrznych serwerów:

```
# lynx 192.168.100.20:25
```

Jednak niektóre przeglądarki, jak Mozilla Firefox, domyślnie blokują

Fragmentacja

Każdy system operacyjny próbuje ustawić maksymalny rozmiar pakietu IP tak, by był on równy maksymalnemu rozmiarowi ramki w warstwie 2. W przypadku Ethernetu to maksimum jest równe 1518 bajtom i nosi nazwę *Maximum Transfer Unit* (MTU). Ponieważ ramka Ethernet potrzebuje 18 bajtów na dane nagłówkowe, przestrzeń dostępna dla pakietu IP to 1500 bajtów.

Podczas wędrówki w sieci pakiet IP może napotkać ruter, który w związku z ograniczeniami w stosowanej technologii warstwy 2 nie będzie w stanie przyjmować tak dużych pakietów. Aby dane mogły przejść przez urządzenie o MTU mniejszym niż 1500, muszą zostać podzielone na wiele mniejszych pakietów. To zjawisko nazywa się fragmentacją.

Z kolei serwer docelowy musi zebrać wszystkie fragmenty IP i poukładać je w odpowiedniej kolejności – ten proces nosi nazwę *reasemblacji*. Wymaga on pewnej ilości danych, by poskładać pakiety w poprawnym porządku i nie wymieszać fragmentów z różnych połączeń przychodzących do tego samego serwera.

Nagłówek IPv4 zawiera dwa pola niezbędne do reasemblacji – *Fragment offset* i *Identification* (ID). Każdy fragment tego samego datagramu ma takie samo pole ID. Umożliwia to stosowi IP rozpoznanie wszystkich pakietów należących do datagramu. Do układania pakietów w odpowiedniej kolejności używane jest pole *Fragment Offset*. Pierwszy fragment pakietu ma zerowy offset, zaś każdy następny zwiększa się o długość części fragmentu z danymi. Bit *More fragments* (MF) nagłówka IP określa, czy aktualny fragment jest ostatni.

takie żądania po stronie klienta. Lepiej więc wykonywać testy za pomocą narzędzi netcat czy telnet.

HTTP Connect

Polecenie HTTP CONNECT zwykle jest używane do tunelowania ruchu SSL przez serwer proxy. Serwer ten po prostu otwiera sesję TCP pomiędzy proxy a docelowym serwerem i przekazuje dane klienta. Niestety, niektóre firewalle nie sprawdzają poprawności docelowych adresów IP oraz portów, tym samym otwierając intruzom możliwości ataku.

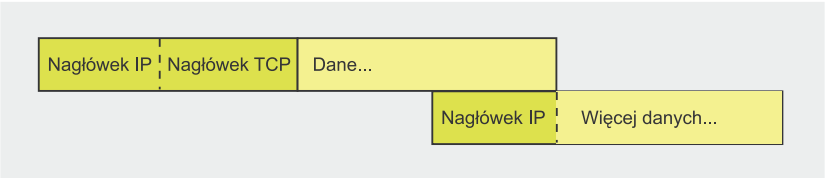
Zapory powinny być konfigurowane tak, by porty służące do administracji były dostępne wyłącznie z wewnętrznych interfejsów sieciowych. Uniemożliwia to atakującym wykorzystanie eksploitów przeciwko demonowi logowania użytkowników lub zgadywanie haseł firewalla. Podatność związana z poleceniem CONNECT umożliwia intruzom nawiązywanie połączenia z interfejsem administratora z zewnętrznych sieci:

```
# nc firewall 8080
CONNECT 127.0.0.1:22 HTTP/1.0
SSH-1.99-OpenSSH_3.8p1
```

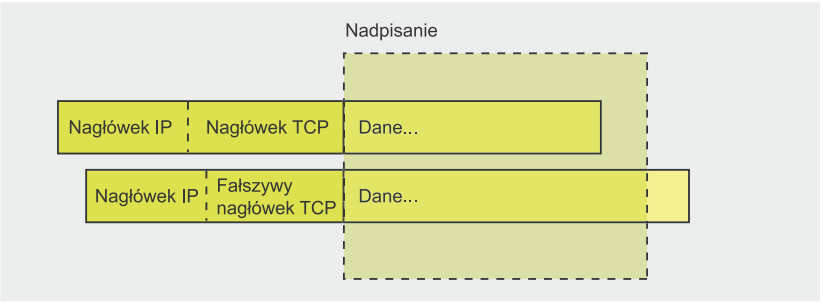
Atakujący mogą także wykorzystać CONNECT do nawiązywania połączeń z wewnętrznymi systemami. Polecenie to – tak samo jak atak *proxy bounce* – umożliwia łączenie się z wewnętrznymi maszynami za pomocą prywatnych adresów IP:

```
# nc firewall 8080
CONNECT 10.1.1.100:25 HTTP/1.0
220 mailserver ESMTP
```

Jak widać, sprawdzanie firewalli pod kątem podatności CONNECT jest bardzo łatwe. Ta sama technika może być używana do zdobywania informacji o wewnętrznych zakresach adresów IP i do skanowania przypominającego atak FTP *bounce*. Co ciekawe, czołowe firewalle, takie jak Checkpoint FW1 lub Astaro Secure Linux, były w starszych wersjach podatne na ataki HTTP Connect.



Rysunek 5. *Zwykła reasemblacja pakietów TCP*



Rysunek 6. *Atak nakładających się fragmentów – nadpisanie nagłówka*

Atak nakładających się fragmentów

Celem ataku nakładających się fragmentów (*overlapping fragment*) jest nadpisanie danych z nagłówka UDP lub TCP już po podjęciu przez firewall decyzji w oparciu o pierwszy fragment. Jeżeli podczas komunikacji TCP lub UDP wystąpi fragmentacja, tylko pierwszy fragment zawiera dane z nagłówka TCP/UDP (takie jak port docelowy). Atak ten można przeprowadzić na przykład wtedy, gdy reguła firewalla zezwala na połączenia z portem 80 TCP serwera WWW, ale jednocześnie zabrania połączeń z demonem SSH na tym samym serwerze (port 22).

Atakujący fragmentuje datagram IP (patrz Ramka *Fragmentacja*) i ustawia w nagłówku TCP port docelowy na 80. Fragment dociera do firewalla i spełnia wymogi reguły *Allow*. Ponieważ wszystkie fragmenty IP datagramu mają ten sam adres IP i pole ID, firewall przepuszcza wszystkie następne fragmenty z tym samym IP i tymi samymi adresami IP (źródłowym i docelowym), które miał pierwszy fragment.

Offset pierwszego datagramu jest zerowy, zaś jego koniec znajduje się na przykład w bajcie 128. Offset drugiego fragmentu powinien mieć wartość rozpoczynającą się zaraz po bajcie 128. Jeśli offset jest mniejszy niż 128, część pierwszego fragmentu zostanie nadpisana (tak zwany offset ujemny). Jeżeli atakujący obliczy offset drugiego fragmentu w taki sposób, by nadpisał on docelowy port nagłówka TCP, będzie można zmienić wartość portu z 80 na 22 (patrz Rysunki 5 i 6).

Po zakończeniu procesu reasemblacji, na firewallu lub docelowym hoście, zostaje nawiązane połączenie na port 22 TCP (zamiast 80). Ominięcie firewalla się powiodło.

Istnieje kilka innych implementacji ataków wykorzystujących fragmentację – w Ramce *W Sieci* można znaleźć przykład ciekawej techniki agresji na IPFilter w systemach z rodziny BSD.

Ataki wykorzystujące tunelowanie

Intruz może mieć potrzebę komunikowania się przez firewall, na przykład z koniem trojańskim lub tylną furtką zainstalowaną w wewnętrznym systemie. Atakujący wysyła, dajmy na to, polecenia do trojana i chce otrzymać rezultaty tych komend z powrotem.

Jeśli reguły filtrowania zapory umożliwiają tylko ruch wychodzący w typowych protokołach, takich jak HTTP, FTP czy DNS, intruz musi użyć do komunikacji jednego z nich. Na nieszczęście dla atakują-

Tabela 5. *Przykłady podatności firewalli*

Produkt	Podatność
Checkpoint Secure Platform	Możliwość ominięcia reguł firewalla
Checkpoint VPN-1	Przepełnienie bufora ASN.1
Checkpoint VPN-1	Przepełnienie bufora ISAKMP
Cisco IOS Firewall	Przepełnienie bufora proxy uwierzytelniającego
Cisco Catalyst 6500/6700	Możliwość ominięcia modułu ACL usług firewalla

cego niektóre współczesne firewalle sprawdzają składnię ruchu warstwy aplikacji pod kątem zgodności z dokumentami RFC. Jeżeli zawartość połączenia nie jest zgodna z RFC, zostanie ono zablokowane.

Intruzi o tym wiedzą i wykorzystują w atakach tunelowanie, korzystając z narzędzi, które nie naruszają zasad określonych w RFC – ukrywają dane w poprawnych poleceniach protokołów. Jeśli dodatkowo dane te są zakodowane lub zaszyfrowane za pomocą 7-bitowych znaków ASCII, wykrycie ich przez firewall jest prawie niemożliwe. Dobrym przykładem są tunele oparte na protokołach DNS i HTTP. Mimo że narzędzia do tunelowania HTTP zgodne z RFC – takie jak rwwwshell (patrz Ramka *W Sieci*) – są względnie łatwe w implementacji. tunele DNS są nieco bardziej skomplikowane.

Ciekawym tunelem DNS wykorzystującym między innymi technikę zwaną *namedropping* jest ten wykorzystujący protokół *Name Server Transport Protocol* (NSTX), wymaga on jednak klienta i serwera zgodnego z NSTX. W dodatku serwer musi być autorytatywny dla domeny (patrz Ramka *W Sieci*). Wyobraźmy sobie, że atakujący jest autorytatywny dla domeny *zladomena.com* i że skutecznie zaatakował już serwer wewnątrz sieci chronionej przez firewall. Intruz chce mieć możliwość zdalnej kontroli nad systemem z zewnątrz – wysyłać polecenia i otrzymywać odpowiedzi.

Jeżeli klient chce przesłać dane do serwera, wysyła żądanie otrzymania spreparowanej nazwy hosta, na przykład *b2xpdmVyIGthcm93.zladomena.com*, gdzie *b2xpdmVyIGthcm93* to zakodowane dane. Ponieważ wewnętrzny serwer nazw nie obsługuje tej domeny, przekaże żądanie do serwera NSTX atakującego. Serwer DNS intruza może wydobyć i odkodować nazwę hosta z żądania.

Aby móc odesłać dane z powrotem do klienta, serwer nazw atakującego umieszcza dane w rekordzie TXT. Jest to wolny tekstowy rekord, który może być używany do innych celów – na przykład do publikowania kluczy PGP. Firewall będzie miał więc kłopot z rozróżnieniem między poprawnym rekordem TXT a ukrytą informacją od trojana.

Więcej informacji o atakach wykorzystujących tunelowanie można znaleźć w dokumencie *Firewall Piercing* (patrz Ramka *W Sieci*).

Podatności firewalli

Bezpieczeństwo sieci zależy od zabezpieczeń firewalla. Jeśli ten ostatni jest podatny na ataki przepełnienia bufora, można go ominąć bez najmniejszego problemu – atakujący może go skonfigurować według własnych potrzeb. W przypadku podatności dających intruzom zdalny dostęp do powłoki, wszystkie ataki na wewnętrzne systemy pochodzą z adresu IP zapory. Brak wielopoziomowego firewalla oznacza brak dalszych zabezpieczeń sieci.

Podatności związane ze zdalnym wykonaniem kodu są niestety odnajdowane w czołowych firewallach całkiem często. Wystarczy zajrzeć na stronę *http://www.securityfocus.com/* (patrz Tabela 5).

Wnioski

Jest wiele metod ominięcia firewalli. niektóre z nich wynikają z małych możliwości produktów, inne zaś ze złej konfiguracji bądź luk w urządzeniach. Jednak wdrożenie wielopoziomowej technologii pełnostanowych firewalli i regularne kontrole środowiska zapory mogą zapewnić dobrą ochronę wewnętrznych sieci. ●

O autorze

Oliver Karow pracuje jako główny konsultant ds. bezpieczeństwa u jednego z producentów rozwiązań w zakresie bezpieczeństwa. Obecnie koncentruje się na firewallach, technologiach IDS/IPS, audytach bezpieczeństwa i testach penetracyjnych. Oliver studiuje też informatykę na jednej z niemieckich uczelni korespondencyjnych. Pracuje w branży IT od 1994 r. a od roku 1999 zajmuje się bezpieczeństwem informatycznym.

W Sieci

- *http://cert.uni-stuttgart.de/archive/bugtraq/2001/04/msg00121.html* – Thomas Lopatic, *A fragmentation attack against IP Filter*,
- *http://www.ccc.de/congress/2004/fahrplan/files/221-firewallpiercing_21c3.pdf* – Maik Hensche i Frank Becker – *Firewall Piercing – Creative exploitation of valid Internet protocols*,
- http://www.thc.org/download.php?t=r&f=rwwwshell-2.0.pl.gz – rwwwshell, implementacja tunelu HTTP,
- *http://www.csnc.ch/static/services/research/dnstunnel.html* – implementacja tunelu DNS.