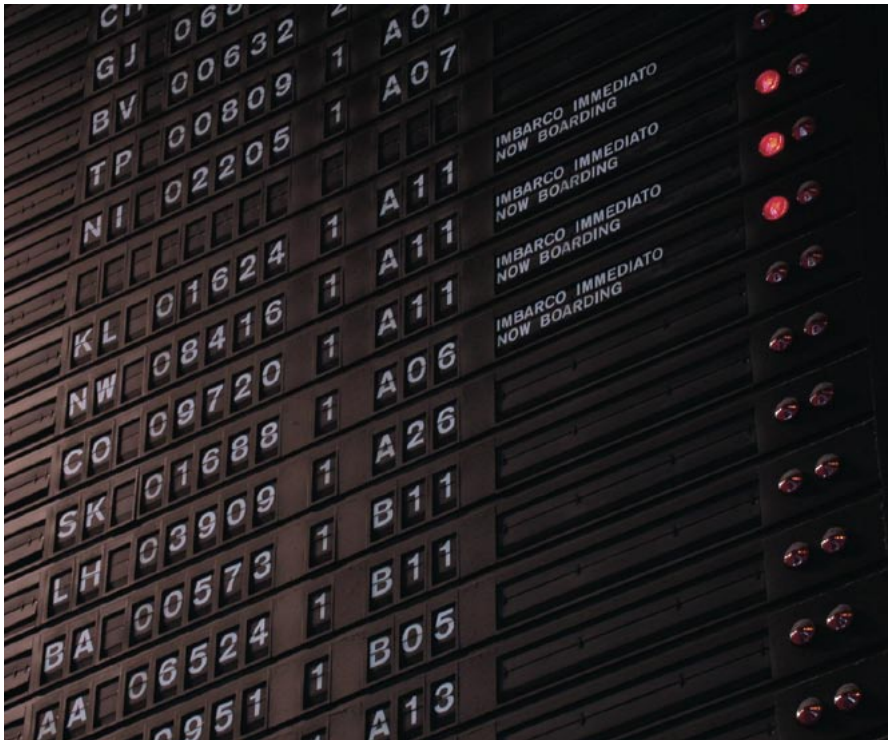


Configurando o Netfilter/IPtables com o Shorewall

Cumprindo tabela



Andrea Jaccarino: www.sxc.hu

Quando os usuários pensam em suas estações de trabalho ou seus computadores domésticos, geralmente se esquecem da segurança. Mas o perigo está à espreita, esperando para esfaquear os incautos usuários. **POR JAMES MOHR**

A única maneira de tornar seu computador completamente imune a ataques é desligá-lo da Internet. Sempre que você abre a porta de sua casa para sair à rua, algum gatuno pode esgueirar-se e entrar. Da mesma forma, os potenciais crackers estão à espera de portas abertas para invadir seu sistema.

Alguns usuários tem a errônea impressão de que os intrusos atacam apenas máquinas caras de empresas conhecidas e não estariam interessados em um computador doméstico. Ledo engano. A verdade é que cada computador conectado na Internet – mesmo os mais insignificantes – é uma vítima potencial de ataques e pode, inclusive, ser usado para originar ataques a uma terceira pessoa. Nem os usuários de conexões discadas estão a salvo. Na maioria dos casos, esses ataques são perpetrados por pessoas com nível técnico não tão bom assim, que

fazem tentativas seguindo uma longa lista de falhas de segurança já publicamente conhecidas.

Para frustrar esses pivetes digitais, pode-se simplesmente fechar todas as portas de saída. Entretanto, se você precisa oferecer serviços à Internet – por exemplo, montar um servidor web – é preciso usar outra forma de proteção.

Mesmo as microempresas precisam da proteção oferecida por um firewall. Muitos usuários não têm poder aquisitivo para bancar um produto comercial. Como usuários de Software Livre, há soluções abertas que nos dão a proteção necessária. Uma delas é o Shorewall.

Nos bastidores

O Shorewall é o nome mais conhecido do produto chamado Shoreline Firewall. Do ponto de vista do usuário, o Shorewall é um conjunto de arquivos fáceis de configurar e, com eles, construir um firewall baseado no Netfilter [1]. Este,

por sua vez, é um recurso presente no kernel das séries 2.4.x e 2.6.x que permite que diversos módulos do próprio kernel acessem o protocolo de rede em vários lugares. Por serem partes do núcleo do sistema, os módulos em questão possuem poderes quase esotéricos e podem fazer praticamente qualquer coisa com um pacote, desde simplesmente bloquear sua passagem até manipular seu conteúdo ou cabeçalho.

O Netfilter também trabalha com o velho ipchains, o filtro de pacotes do kernel 2.2, mas precisa ser explicitamente colocado em “ipchains compatibility mode” (modo de compatibilidade com o ipchains) para que funcione.

Você pode baixar a versão mais atual no site oficial do programa em [2]. Estão disponíveis versões em pacotes RPM ou arquivos tar com o código fonte. Os pacotes RPM não foram testados em todas as distribuições, apenas com as “majors”: SuSE, Redhat e Mandrake.

Verifique o site para mais detalhes. Usuários do Debian podem usar o sistema APT, pois o Shorewall está disponível nos repositórios oficiais.

Para que o Shorewall funcione, é preciso ter instalados os pacotes *iptables* e *iproute/iproute2*. Esses pacotes são instalados por padrão na maioria das distribuições, portanto não devem ser empecilhos. A razão da necessidade do *iptables* é simples: o Shorewall não é, por si só, um firewall. Trocando em miúdos, o Shorewall não é responsável pela verificação, filtragem e manipulação de pacotes de rede. Pelo contrário: o Shorewall simplesmente lê seus arquivos de configuração e usa o comando *iptables* para carregá-las no kernel.

Como o *iptables* assume a tarefa de manipular as tabelas de filtragem do kernel, o Shorewall não é necessário mais do que uma vez a cada boot. É possível, inclusive, ver o que realmente o programa faz olhando dentro dele. Não se assuste, não é preciso um editor hexadecimal nem “cavocar” no código fonte para isso: o programa *shorewall* (normalmente em */sbin/shorewall*) não é nada mais do que um shell script.

Para que o *iptables* saiba o que fazer, é preciso dizer ao kernel quais são as restrições que se quer impor ao tráfego. Os chamados *rulesets* (conjuntos de regras) são definidas no interior do *iptables* e consistem de uma conexão e um certo número de “classificadores”. Cada *ruleset* determina se uma conexão em particular deve ser permitida ou bloqueada, se e como deve ser manipulada, regras para seu redirecionamento e assim por diante.

Esse conceito é basicamente o mesmo em qualquer firewall, seja comercial ou gratuito, e com qualquer número de máquinas, desde algumas poucas até centenas delas – embora nesse caso é provável que o administrador prefira quebrar a rede em segmentos menores. Cada segmento poderia ser gerenciado por seu próprio firewall – e o Shorewall é perfeito para isso.

Um detalhe a ser observado é que não é necessário ter um computador dedicado para ser o firewall. Embora seja a prática comum (e, geralmente, uma boa idéia), usuários domésticos prova-

Listagem 1: Um exemplo de arquivo *zones*

```
#ZONE DISPLAY COMMENT
net Net          a zona da Internet
loc Local        a rede local
fw FW            o firewall
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
#A linha acima não deve ser apagada nem alterada.
```

velmente não têm espaço ou recursos financeiros para montar computadores extras só para agir como firewalls. Se sua estação de trabalho está conectada diretamente à Internet, por que não colocar o firewall diretamente nela?

Em casa, minha rede consiste de um computador com uma conexão ADSL à Internet, uma segunda máquina com Windows XP e outra com Linux. Cada uma delas serve a um propósito diferente e eu permito apenas as conexões apropriadas a cada uma delas, tanto saindo como entrando.

Como minha rede é bastante simples, preciso editar uns poucos arquivos de configuração. Por ser uma situação bem simples e comum, minha rede doméstica é um ótimo local para começar.

Configuração básica

O arquivo de configuração principal fica em */etc/shorewall/shorewall.conf*. O arquivo permite que se configure qualquer coisa, desde a ativação até a desativação do sistema. Embora seja possível configurar um grande número de valores nesse arquivo, ainda não encontrei razões suficientes para fazê-lo em minha rede doméstica.

Você já deve estar familiarizado com a palavra *segmento* para referir-se a porções específicas em uma rede. O Shorewall usa, para isso, o termo *zona*. Em casa, possuo quatro zonas: *fw* (o próprio firewall, ou seja, o meu computador), *net* (a Internet) e *loc* (a rede local). A quarta zona chama-se *games*, mas falaremos dela mais adiante.

Os nomes das zonas devem ser curtos (cinco caracteres ou menos) e podem conter letras e números. Observe que não é possível usar a zona especial *all*. A variável *FW*, no arquivo *shorewall.conf*, define a zona específica do seu firewall. O valor padrão para ela é *fw*. Lembre-se que não é possível usar esse nome em outra zona!

Mesmo que não esteja oferecendo nenhum serviço a computadores fora de sua rede local (ou seja, na Internet) ainda assim a zona da Internet será necessária. Lembre-se que as regras do *IPtables* definem conexões ponto-a-ponto, ou seja, precisam de um ponto de origem e outro de destino. Um desses pontos seria, talvez, a sua estação de trabalho, mas o outro certamente é a Internet. Portanto, é necessário defini-la como uma zona específica.

Esses nomes são apenas convenções. Embora sejam os valores *default* do Shorewall, pode-se usar o nome que bem entender, desde que haja consistência em todos os arquivos de configuração.

Por padrão, as permissões no diretório */etc/shorewall* são definidas com o valor 700, o que significa que apenas o proprietário (ou seja, o root) tem acesso aos arquivos. Mesmo acesso apenas de leitura para outros usuários seria perigoso, pois alguém *poderia* descobrir um furo de segurança e explorá-lo.

As zonas são definidas no arquivo */etc/shorewall/zones*. Cada linha possui três valores: o nome da zona (usado para referenciar essa zona nos outros arquivos), nome legível (que aparece

Listagem 2: Um exemplo de arquivo *interfaces*

```
#ZONE INTERFACE BROADCAST OPTIONS
net ppp0 - routefilter,norfc1918
loc eth0 detect -
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
#A linha acima não deve ser apagada nem alterada.
```

Listagem 3: um exemplo de arquivo *policy*

```
#SOURCE    DEST      POLICY    LOG LEVEL
fw         net       ACCEPT    info
fw         loc       ACCEPT    info
loc        net       REJECT    info
loc        fw        ACCEPT    info
net        all       DROP      info
all        all       REJECT    info
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
#A linha acima não deve ser apagada nem alterada.
```

quando o Shorewall está carregando as regras) e um comentário. A listagem 1 mostra um arquivo *zones* básico.

Definindo as regras de comunicação

Embora tenhamos definido as zonas, o firewall ainda não sabe como conectá-las. Não há associação entre o nome da zona e a rede que representa. Para isso, precisamos mexer no arquivo */etc/shorewall/interfaces*, que contém uma tabela com quatro colunas: zona, interface, broadcast e opções.

Em meu sistema, o arquivo *interfaces* parece-se com o da listagem 2.

A zona é o nome definido no arquivo *zones*. Já a interface é o nome (no Linux) da interface de rede conectada a essa zona. Por exemplo, minha conexão ADSL usa o protocolo Point-to-Point Protocol (PPP), portanto o nome da interface é *ppp0*. O nome da interface para a minha placa de rede Ethernet é *eth0*. Para descobrir os nomes de suas interfaces de rede, use o comando */sbin/ifconfig*. A coluna *broadcast* é o endereço de difusão da rede conectada àquela placa. Como era de se esperar, a coluna *options* especifica quaisquer opções que se queira usar.

Em meu sistema há duas linhas:

```
net ppp0 routefilter,norfc1918
loc eth0 detect -
```

A zona da Internet está conectada à interface *ppp0*. Já a zona local está associada à interface *eth0*. Como há três zonas por padrão, você deve estar se perguntando onde foi parar a linha referente ao firewall. Bem, de forma simples, a zona *firewall* conecta-se às outras por uma das interfaces já especificadas. Portanto, pense no arquivo

interface como sendo um modo de definir quais interfaces a zona *firewall* usa para se comunicar com as outras zonas.

No caso da interface *ppp0*, a coluna *broadcast* possui um hífen (-). Como uma conexão PPP não possui broadcast, eu poderia ter deixado a coluna em branco. Entretanto, eu desejava especificar opções adicionais, portanto precisava do “-” para “guardar o lugar” da coluna *broadcast* e, assim, chegar à coluna *options*. Se não houvessem opções, essa coluna poderia ser deixada em branco. Como “mania” pessoal, entretanto, sempre coloco os hífens para me lembrar de que alguma coisa existe naquela posição.

A opção *routefilter* diz ao kernel para rejeitar, naquela interface, qualquer pacote que deveria ter sido roteado por outra interface. Em nosso caso, se a *ppp0* recebesse, de dentro para fora, um pacote que deveria sair pela *eth0*, este seria bloqueado imediatamente. Esse recurso é chamado de *anti-spoofing*.

A segunda opção, *norfc1918*, ordena ao kernel que não roteie endereços especificados como “privados” pela RFC 1918. A RFC (Request For Comments) de número 1918 é um documento do IETF (Internet Engineering Task Force) que lista as faixas de endereço IP que podem ser usadas por qualquer um sem precisar de permissão especial e que, *sob hipótese alguma*, devem ser roteadas para a Internet. Essa opção assegura para a Internet. Essa opção assegurada que, realmente, não serão. Mais detalhes sobre a RFC 1918 podem ser obtidos em [3].

Nesse ponto deparei com um dilema. Quero que meus computadores possam acessar a Internet; mas, como possuem endereços RFC 1918, eles estão bloqueados. Como fazer para acessar a Internet, então? Já voltaremos a esse problema.

É possível configurar o Shorewall para comportar-se de qualquer maneira que desejemos, desde que manipulemos suas zonas. Cada zona pode possuir uma *política de funcionamento (policy)*, configurada – como já deve ter ficado óbvio neste ponto da matéria – pelo arquivo *policy*. Nele, os campos são: *source* (a zona de origem), *dest* (a zona de destino), *policy* (a ação a ser tomada por padrão) e o *log level* (quanta informação registrar nos logs).

A Listagem 3 mostra um exemplo do arquivo *policy*. Observe atentamente as duas últimas linhas. A primeira afirma que qualquer tráfego desconhecido vindo da Internet é sumariamente bloqueado (ou seja, ignorado).

Há quatro opções possíveis para a política de conexões:

ACCEPT – Aceita os pedidos de conexão.

DROP – Ignora (“derruba” ou “bloqueia”) o pedido de conexão.

REJECT – Além de ignorar, retorna uma mensagem de erro ao computador que requisitou a conexão.

CONTINUE – Permite colocar estações em mais de uma zona e aplicar políticas a todas elas.

O nível de registro determina quanta informação é enviada ao subsistema de registro de eventos do kernel, o *syslog*. Lembre-se sempre de que tudo isso é feito pelo *iptables*, que é, em última análise, outro subsistema do kernel. Tudo o que o Shorewall faz é especificar a prioridade das mensagens a serem registradas. Para mais informações sobre o registro de eventos do sistema, consulte a página de manual do *syslogd (man syslogd)*.

As duas últimas linhas do meu arquivo *policy* são:

```
net all DROP info
all all REJECT info
```

A primeira linha nos diz que qualquer tráfego não coberto por nenhuma outra regra (veremos essas regras mais adiante) e que venha da Internet será sumariamente bloqueado (ação DROP). Entretanto, qualquer outro tipo de tráfego será rejeitado (ação REJECT), o que quer dizer que o emissor das men-

sagens bloqueadas receberá um aviso relatando o motivo do bloqueio. Pode parecer meio estranho, mas quando raciocinamos sobre o comportamento de ambas as ações vemos que faz todo o sentido do mundo. Quando um pacote chega ao firewall vindo da Internet e destina-se a uma porta ou serviço que bloqueamos voluntariamente, queremos simplesmente impedir sua entrada. *De forma alguma* queremos informar o que aconteceu ao remetente. Se este for um indivíduo mal-intencionado (um cracker, por exemplo) esse tipo de aviso pode ajudá-lo a contornar nossas barreiras de segurança. Entretanto, pacotes originados em qualquer outra rede que não a Internet (em nosso caso, minha máquina e minha rede interna) são rejeitados, permitindo que os programas-cliente possam ter alguma idéia do que aconteceu.

Note bem: estamos configurando o *comportamento padrão* das zonas. Mas o que isso significa? Simples: que se você não definir nenhuma regra específica para cada um dos tipos de conexão, o comportamento padrão definido no arquivo *policy* será aplicado.

No arquivo *policy* da rede que tenho em casa possuo apenas essa linha padrão para a zona *net*. Teoricamente, qualquer pacote vindo da Internet deveria ser bloqueado. Entretanto, possuo um servidor web em minha máquina com um bocado de informações de referência que gostaria de acessar do trabalho. Uma vez que essa conexão se dá através da Internet, a linha correspondente no arquivo *policy* impede que eu acesse meus bookmarks e anotações quando estiver longe de casa.

A resposta está no arquivo *rules*, que é o coração do Shorewall. Aqui definimos como o firewall deve se comportar para cada tipo específico de conexão. Podemos criar regras usando qualquer abstração criada pelo Shorewall: zonas, serviços de rede (portas), segmentos de rede, máquinas individuais e praticamente qualquer combinação desses elementos. Quando uma requisição de conexão chega vinda da Internet, o sistema primeiro verifica se alguma regra definida no arquivo *rules* pode decidir o que fazer com ela. Se nenhuma regra apropriada existir, usa-se a regra padrão definida no arquivo *policy*.

As colunas no arquivo *rules* são: ação a ser tomada, origem da requisição, destino da conexão, protocolo usado, porta destino, porta de origem e o destino original. Destino original? Calma, já veremos do que se trata.

Além das ações já vistas no arquivo *zone*, o arquivo *rules* guarda algumas surpresas. A ação DNAT permite que se faça o chamado NAT – *Network Address Translation* (tradução de endereços de rede). O DNAT – *Destination Network Address Translation* – é uma especialização do NAT e permite que requisições de conexão possam ser redirecionadas para computadores dentro da rede, com endereços diferentes do original e mesmo em portas diferentes.

A ação REDIRECT redirecionará conexões para portas específicas na mesma máquina – útil para redirecionar pedidos de conexão HTTP para o proxy local (por exemplo o squid [4]). Para redes pequenas, que não precisam ter acesso irrestrito à Internet, descobri que a ação REDIRECT é extremamente útil. Pode-se, por exemplo, permitir apenas tráfego HTTP para os clientes internos, bloqueando qualquer outra coisa. No caso da minha rede, faço algo parecido com a seguinte linha no arquivos *rules*:

```
REDIRECT loc 3128 tcp www - !10.2.38.0/24
```

Resumidamente, a ação acima diz que todas as conexões que usem o protocolo de transporte TCP e que requisitem um serviço de web (protocolo de aplicação HTTP, normalmente disponível na porta 80) sejam redirecionadas para a porta 3128. Essa é a porta em que o servidor proxy Squid “escuta”.

Aqui temos a figura do destino original. Especificamos uma classe C inteira usando a notação CIDR (*Classless Inter-Domain Routing*): 10.2.38.0/24. O sinal de exclamação na frente do endereço indica que estamos *negando* a regra, ou seja, pacotes vindos de qualquer endereço são redirecionados ao Squid *exceto* os endereços pertencentes à rede 10.2.38.0/24. A razão é óbvia: quero que todos na minha rede local acessem o servidor web sem passar pelo proxy, reservado apenas ao tráfego externo.

A ação LOG primeiro registrará o evento no *syslog*. Ao contrário das

outras ações, entretanto, não interromperá o processamento das regras abaixo dela. Em vez disso, passará o comando para a próxima regra assim que registrar o evento. Dessa forma, é possível registrar um evento e, depois, aplicar outra ação a um mesmo pacote.

A ação QUEUE redireciona pacotes para programas que rodem no espaço do usuário, que por sua vez os manipulam e os devolvem à pilha de rede.

Para configurar seu arquivo *rules*, assumo como regra geral que se deve *bloquear* tudo e liberar apenas os serviços necessários – e olhe lá! muitas pessoas fazem o contrário, começando com uma rede completamente escancarada e desligando, um a um, os serviços desnecessários – ou pior: desligando só o que já foi invadido! Essa é uma prática perigosíssima: se esquecer de fechar alguma coisa, pode ser (ou melhor, será) alvo de um ataque em breve. No primeiro caso, esquecer de abrir alguma coisa bloqueará momentaneamente algum serviço, que pode ser aberto facilmente depois. O que é uma ligeira inconveniência comparada ao risco real de ataque?

Escovando bits

Já deve estar óbvio para o leitor como a nomenclatura dos arquivos do Shorewall funciona. Por exemplo, é mais que evidente que o arquivo */etc/shorewall/hosts* permite a definição de máquinas (hosts) específicas. Normalmente não é necessário mexer nesse arquivo em redes pequenas. Quanto menor for a rede, mais provável que a configuração em uma zona seja configurada da mesma maneira (ou, mais especificamente, que todas as máquinas conectadas a uma mesma *interface* do firewall possuam configuração semelhante).

Talvez esse não seja seu caso e você precise de regras de acesso diferentes para cada máquina de uma mesma rede ou segmento. Há alguns meses meu filho começou a jogar um RPG via Internet. Para isso, precisava de mais do que acesso HTTP via proxy.

Para liberar esse acesso, defini um conjunto específico de computadores como uma nova zona. Para tanto, criei uma nova linha para ela no meu arquivo *zone*, à qual chamei *game*. Depois, criei

uma linha no arquivo *hosts* contendo, apenas, o computador do meu filho.

```
game eth0:10.2.38.13
```

Mas não é só. Lembra-se de que minha rede local era 10.2.38.0/24? Este é um dos endereços privados definidos na RFC 1918. Mesmo que eu quisesse rotear esses endereços pelo meu ADSL, posso apostar que meu provedor de Internet os bloquearia mais adiante. O que podemos fazer, então?

A solução é o que chamamos de *IP masquerading*, ou mascaramento de IP. Como o nome indica, um endereço IP válido (ou seja, roteável) é usado para “mascarar” os endereços IP privados. No meu caso, o endereço IP da linha ADSL (que possui um endereço válido) mascara os IPs da minha rede local. Portanto, preciso criar uma linha no arquivo */etc/shorewall/masq* semelhante à seguinte:

```
ppp0 10.2.38.0/24
```

A interpretação é simples: todo o tráfego vindo da rede 10.2.38.0/24 e que saia para a Internet através da interface *ppp0* tem que ser mascarado com o IP dessa interface.

Nesse ponto, a configuração básica do mascaramento já está pronta. Agora, entretanto, é que vem a parte *penosa*. Não foi fácil conseguir informações sobre o jogo e sobre como jogá-lo através de um firewall. Para falar bem a verdade, toda a documentação que encontrei falava praticamente a mesma coisa: se quiser jogar, desabilite seus firewalls!

Entretanto, quase que por acidente consegui fazer o jogo funcionar. Para tanto, coloquei o nível de registro (log) no arquivo *policy* em modo *debug* e observei detalhadamente o que acontecia, anotando todas as tentativas de conexão do computador de meu filho com *qualquer* máquina. Depois, verifiquei via consultas DNS se aquela máquina pertencia aos desenvolvedores do jogo. Por último, adicionei a porta específica usada pelo jogo ao meu arquivo *rules*.

Se eu quisesse, poderia ter adicionado regras para acesso ao serviço WWW (porta 80). Entretanto, como já

uso Squid como proxy e meu filho precisava apenas do acesso ao jogo, continuei usando a cláusula REDIRECT para conexão a websites.

Caçando bugs

É bem possível (provável, até) que você tenha problemas ao configurar o Shorewall. Portanto, a capacidade de mostrar todos os detalhes de cada conexão é uma ferramenta valiosa.

Uma técnica útil de depuração é colocar o nível de *logging* em modo *debug*, o que gera uma quantidade espantosa de informação. Entrar em detalhes sobre os pormenores de cada evento está fora do escopo deste artigo. São, entretanto, fáceis de inferir mesmo sem verificar todas as linhas. Por exemplo, observe o registro a seguir:

```
Nov 1 11:19:32 saturn kernel: ➤
Shorewall:net2all:DROP:IN=ppp0 ➤
OUT= MAC=
SRC=1.2.3.4 DST=10.2.38.11 ➤
LEN=48 TOS=0x00 PREC=0x00 ➤
TTL=116 ID=47048 DF
PROTO=TCP SPT=1 292 DPT=1080 ➤
WINDOW=64240 RES=0x00 SYN URGP=0
```

Este é um registro padrão de */var/log/messages*. No começo temos a data, o nome do computador e o subsistema do syslog (no caso, o kernel). Depois disso vem a mensagem real. Como se pode notar, o pacote entrou pela *ppp0* (meu modem ADSL) e o pacote foi bloqueado (DROP). Podemos ver, também, que a conexão *net2all* estava em uso, ou seja, o pacote estava vindo da Internet em direção a alguma das outras zonas.

Se voltarmos à discussão sobre o arquivo *policy*, veremos que o procedimento padrão era bloquear (DROP) qualquer pacote que viesse da Internet em direção a qualquer outra interface (*all*). Algum dos pacotes tentou chegar à porta de destino (DPT) de número 1080, a porta do serviço *socks*. Não posso nada rodando ali e certamente não me comuniquei com ninguém que precisasse dela. Como não é uma porta “padrão”, nem há um serviço associado a ela por *default*, não há razão para ninguém acessá-la pela Internet. Para mim parece óbvio: alguém estava me testando para tentar explorar uma falha conhecida do Windows.

Se o computador remoto (ou mesmo outro computador na mesma rede) continuasse a tentar acessar várias portas em minha máquina, eu poderia colocá-lo numa “lista negra”. Para isso, basta colocar os IPs (ou redes inteiras, no formato CIDR) no arquivo */etc/shorewall/blacklist*. Isso quer dizer que, em detrimento de todas as regras que porventura permitissem acesso, pacotes vindos desses IPs são sempre bloqueados.

Por padrão, o Shorewall usa o subsistema de registro (*syslog*) do Linux para enviar as mensagens automaticamente para */var/log/messages*. Mesmo quando se configura o *syslogd* para enviar mensagens a outro arquivo, acho maçante que as mensagens do meu firewall estejam misturadas a mensagens do kernel.

Para resolver a questão usamos ULOG, que deve estar habilitado no kernel. A boa notícia é que isso é padrão nas distribuições modernas. A má é que o pacote *ulogd*, necessário para que os usuários possam usar o serviço, quase nunca está disponível nas distros. Em todo caso, pode-se baixá-lo de [5].

Uma vez que o *ulogd* esteja configurado e rodando, não se usa mais os níveis de registro do *syslogd* no arquivo *policy*. Em vez disso, use ULOG (tudo em maiúsculas). A configuração do *ulogd* é independente do *syslogd*, portanto qualquer mudança em */etc/syslog.conf* não afeta o *ulogd*.

Uma última dica: o Webmin possui um módulo para configuração do Shorewall, disponível em [6]. Para mais informações sobre o Webmin, visite o site oficial do programa em [7]. ■

INFORMAÇÕES

- [1] Site oficial do Netfilter:
<http://www.netfilter.org>
- [2] Site oficial do Shorewall:
<http://www.shorewall.net>
- [3] Endereços para Redes Privadas (RFC 1918):
<http://rfc.net/rfc1918.html>
- [4] Proxy Squid:
<http://www.squid-cache.org/>
- [5] Ulog:
<http://gnumonks.org/projects/ulogd>
- [6] Módulo do Webmin para administrar o Shorewall: <http://www.webmin.com/download/modules/shorewall.wbm.gz>
- [7] Site oficial do Webmin:
<http://www.webmin.com>