Checkpoint FW1 e Firewall Builder

Regras para todos

A tecnologia de firewall presente no kernel do Linux já há algum tempo atende às exigências profissionais. Agora, os configuradores gráficos chegaram para diminuir a distância entre o Linux e as soluções comerciais. POR CHRISTIAN NEY

Empresas que usam linhas privadas para se conectar à Internet precisam de muita proteção contra ataques à sua rede interna. Em muitos casos, os firewalls são a primeira linha defensiva, permitindo que os administradores definam o tipo de tráfego autorizado a entrar e sair de suas redes corporativas. De quebra, um firewall também pode restringir o acesso a serviços da Internet para os funcionários e usuários do sistema.

O mercado continua a crescer, principalmente devido à penetração da própria Internet. O leque de produtos inclui desde opções livres e de código aberto (normalmente gratuitas) a soluções comerciais esplendidamente caras e com "satisfação garantida ou seu dinheiro de volta".

Praticamente todas as soluções comerciais oferecem uma forma fácil de se configurar e administrar o firewall – normalmente um programa gráfico. Em oposição a isso, a grande maioria dos firewalls de código aberto impingem ao operador uma linha de comando que não é exatamente fácil de usar. Essa é a razão pela qual muitos administradores evitam usar qualquer sistema envolvendo Software Livre.

Obviamente, uma boa interface não pode substituir o conhecimento e a experiência. Nenhuma interface gráfica vai formar especialistas de firewall do dia para a noite, mas a maioria das pessoas prefere uma abordagem visual do problema em vez de uma interminável lista de longos comandos. É mais fácil implementar um projeto de segurança se houver ferramentas gráficas disponíveis para auxiliar na tarefa.

Nossa rede de exemplo

Basearemos nossos exemplos na rede mostrada na figura 1. Mostraremos como implementar políticas de segurança usando o CheckPoint Firewall 1 NG, comercial, e o Firewall Builder, um programa de código aberto. Nossa rede usa um roteador para se conectar à Internet. A empresa fictícia que possui essa rede decidiu que hospedará seu site num servidor próprio. O servidor ficará localizado numa *DMZ* (*Zona Desmilitarizada*, uma rede à parte, com menos restrições de acesso) em vez de diretamente na rede interna. Os empregados da companhia devem poder acessar serviços de FTP, HTTP e HTTPS na Internet protegidos por um proxy interno. Um servidor de email na LAN usará o protocolo POP3 para baixar as mensagens de email de um provedor externo. Isso permite que as mensagens sejam verificadas quanto a presença de vírus em um único local.

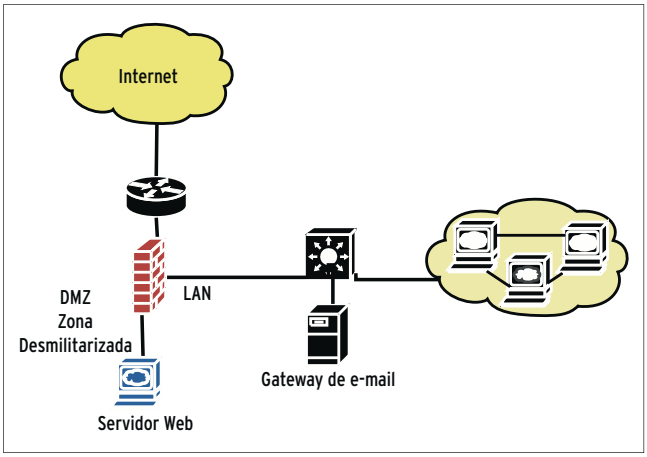

Figura 1: Um exemplo típico de configuração de uma rede pequena, mostrando a DMZ separada da rede principal.

O servidor de email é o único computador com permissão para enviar mensagens para fora usando o protocolo SMTP. O firewall terá proteção adicional, pois só poderá ser acessado por um único computador dentro da rede interna, e apenas por meio de SSH.

Um genuíno faz-tudo

O Firewall 1 NG (Next Generation), da empresa CheckPoint [1] é provavelmente o mais famoso produto comercial dessa categoria, sendo considerado um genuíno faz-tudo. O CheckPoint não só oferece um firewall com uma reputação ilibada de ser extremamente seguro como pode, a preços não tão módicos assim, ser estendido para oferecer uma solução de VPN que permite até mesmo a conexão com soluções de código aberto – como o FreeS/WAN, por exemplo. Infelizmente, até o presente momento o mundo do Software Livre foi incapaz de apresentar uma solução tão completa quando essa. O CheckPoint possui uma interessante interface de configuração chamada de *SmartDashboard* (algo como "Painel Inteligente").

O CheckPoint é composto por dois tipos de módulos:

- O *Management Module* (módulo de administração) é usado para "compilar" as regras criadas com o uso da interface. Trocando em miúdos: converter as regras em linguagem humana para um formato que o firewall possa entender. O módulo transfere essas regras para um ou mais firewalls e administra os módulos de registro (log), os objetos usados na configuração e os bancos de dados que autenti-

cam os usuários com direito de uso do firewall. O Certificate Authority (uma espécie de "cartório digital") é outro módulo importante. O CA gerencia os certificados emitidos para qualquer sistema autorizado a usar um dado recurso e roda no computador que administra o sistema. Essa arquitetura tem a vantagem de possibilitar a centralização do gerenciamento de múltiplos firewalls. Quaisquer computadores envolvidos em uma transação de rede devem confiar no servidor central.

- Módulos individuais de controle de tráfego, chamados "Enforcement Modules", são usados para montar dispositivos que agem como filtros de pacotes baseados em regras. Em outras palavras, eles são o que comumente se chama de firewall. Os módulos ajustam automaticamente o sistema operacional nos quais rodam para que se tornem menos vulneráveis a ataques – em inglês, chamamos isso de "*hardening*". Isso deixa apenas uns poucos itens para os administradores ajustarem manualmente, como por exemplo desabilitar os muitos serviços desnecessários sempre presentes no /etc/inetd.conf.

O conjunto de regras de filtragem é criado por um aplicativo especial, normalmente instalado em uma máquina cliente. O administrador pode usar sua interface gráfica para criar as regras, mas a informação é gerenciada de forma centralizada pelo Management Module.

Separando a interface gráfica dos módulos de administração

Obviamente, todos os módulos e até mesmo a interface gráfica podem rodar em uma única máquina. Essa configuração, entretanto, não é recomendada, já que interferiria no desempenho global do sistema. O registro de eventos (logging) impõe uma carga por demais pesada na máquina. Por razões de segurança, o Management Module deve rodar em um servidor dedicado. Esse arranjo também proporciona uma segurança adicional, uma vez que todas as máquinas envolvidas podem ficar em uma rede separada.

Em ambiente de produção, muitas empresas podem optar por rodar ambos os módulos em uma única máquina, embora a interface de administração tipicamente esteja instalada na estação de trabalho do administrador. Infelizmente, essa máquina deve, obrigatoriamente, rodar Windows ou Solaris, pois a interface de administração não possui uma versão para Linux.

A única distribuição de Linux que o módulo de firewall suporta, oficialmente, é o Red Hat. Com **muito** trabalho é possível fazer o mesmo com um Debian. A instalação, entretanto, é um desafio e tanto, pois os procedimentos são um tanto centrados no Red Hat. Em listas de discussão que tratam do *CheckPoint* [11] descobrimos que é possível instalar o FW-1 no SuSE, no Mandrake e no Slackware. Kernels especiais são necessários em alguns casos. Como de praxe, essas variações são levadas a termo por conta e risco dos especialistas envolvidos, pois o fabricante não prestará suporte sob nenhum pretexto.

A interface gráfica do CheckPoint está dividida em quatro painéis por padrão (ver figura 2). Você pode ajustar, é claro, o modo de visão de forma a satisfazer suas necessidades e preferências. O painel à esquerda mostra

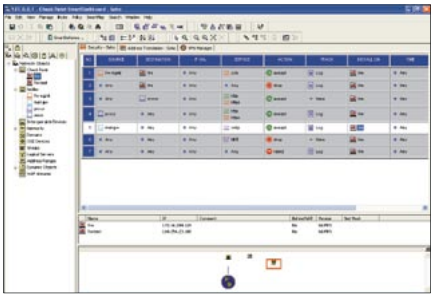

Figura 2: Exemplo de um conjunto de regras de filtragem criada com o CheckPoint SmartDashboard.

uma lista de objetos – firewalls, computadores, serviços e o que quer que um administrador atarefado precise para criar regras de filtragem. O *CheckPoint* oferece, felizmente, uma vasta seleção de definições. Até serviços praticamente extintos, como o Gopher, estão presentes, o que retira das costas do administrador um peso bastante incômodo. Com o tempo livre, o responsável pela segurança da rede pode criar novos objetos, pensar melhor nas regras ou ter tempo para um café.

Policy Editor

O *Policy Editor* (editor de regras, canto superior direito) ocupa uma grande parte da tela. O editor possui abas para as regras de filtragem, para as definições de tradução de endereços (NAT) e, dependendo da sua licença (e do quanto você pagou), outros objetos como o VPN Manager. Na primeira vez que você abre o programa, nenhuma regra de filtragem estará definida. A primeira tarefa é criar um gateway (concentrador de conexões para a Internet) que, futuramente, será também o firewall. Para isso, pode-se tanto usar um *wizard* (assistente) quanto fazer tudo manualmente. É importante adicionar o módulo ao SIC (Secure Internal Communication).

O fato de que objeto *firewall* possua internamente recursos de *anti-spoofing* entre suas regras nos dá uma camada adicional de proteção interna. Essas regras asseguram que invasores não possam conseguir acesso usando técnicas de impostura – em outras palavras, fingindo ser alguém de dentro. O exemplo clássico é o *IP Spoofing*: o atacante forja um endereço IP que pertença à nossa rede interna. Quando você ativa as regras de filtragem do firewall, o editor emite um aviso para cada interface

Inspeção de estado versus tabelas de estado

Em listas de discussão e newsgroups, usuários novatos freqüentemente fazem perguntas a respeito da diferença entre um filtro de pacotes por estado de conexão (o chamado *Stateful Firewall*) e a técnica usada no CheckPoint, batizada de *Stateful Inspection* (inspeção de estado) – e isso não é surpresa, pois o pedante nome dado pela empresa leva a uma conclusão errônea. Todos os filtros de pacotes discutidos neste arquivo oferecem a opção de simplificar o conjunto de regras usando as chamadas tabelas de estado (*state tables*). O uso dessas tabelas melhora o desempenho do filtro de pacotes e, ao mesmo tempo, a segurança.

Inspeção de estado de conexão

O Checkpoint usa a mesma tecnologia [10], mas também analisa os dados úteis do pacote – ou seja, os dados da camada de aplicação, da mesma forma que um proxy. Para conseguir isso, faz uso de scripts especialmente criados para a tarefa. Como resultado, o monitoramento das comunicações é muito mais eficiente.

que ainda não esteja com o *anti-spoofing* ligado, como um lembrete para que você pare tudo e faça isso. As mensagens também são gravadas no registro de eventos do sistema (syslog).

Com essas etapas cumpridas, podemos começar a definir os objetos necessários. Uma regra, basicamente, é composta por:

- Objeto de origem (Source);
- Objeto de destino (Target);
- O serviço usado para a comunicação;
- Uma restrição para a regra se a conexão usa uma VPN específica;
- Definição de registro no *log* se a regra for usada;
- Em quais firewalls essa regra deve ser aplicada – isso permite definir a mesma regras para vários firewalls;
- O horário em que essa regra deve estar ativa;
- Um comentário sucinto e informativo, que esclareça a qualquer um o objetivo dessa regra.

A regra precede uma lista de objetos, que é, no fundo, uma visão mais detalhada sobre os objetos criados pelo usuário. A lista revela várias das características intrínsecas de cada um, como, por exemplo, seu número IP. Isso pode ser extremamente útil quando se tem que lidar com um grande número de objetos e precisamos vislumbrar rapidamente os IPs usados em uma subrede específica.

Visão geral da rede

O último painel contém o *Visual Policy Editor*, um mapa gráfico de todos os nós, suas interligações e a infra-estrutura da rede. Um mapa desse tipo é sempre bárbaro caso existam conexões entre um punhado de computadores usando uma VPN – além de permitir que o administrador de rede se exiba para os colegas e superiores quando o chefe pede "para ontem" um relatório com o diagrama atualizado da rede. É possível exportar o diagrama como arquivo de imagem no formato do Microsoft Visio. Isso permite manter a documentação da rede sempre em dia – coisa extremamente necessária se os firewalls e a rede são administrados por uma equipe. Infelizmente, tudo isso só está disponível se você pagar – e não é barato!

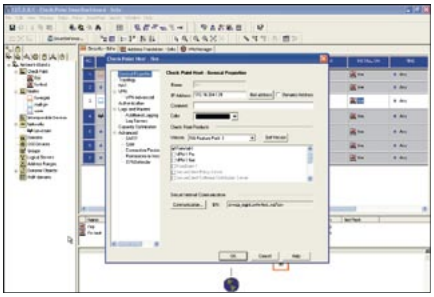

Figura 3: Os objetos do CheckPoint Firewall-1 permitem que quase qualquer tipo de operação de rede possa ser feita.

É muito simples criar regras individuais. Simplesmente arraste e solte os objetos desejados nos campos apropriados. É possível usar o mouse para deslocar regras – recurso que não estava disponível nas versões anteriores. A estrutura bem definida da interface torna a vida do administrador bastante simples, mesmo para os que não possuem muita experiência com esse tipo de tecnologia. Entretanto, a quantidade fabulosa de opções, nem todas óbvias ou aparentes, sempre deixa algum espaço para otimizações.

O recurso de permitir que o firewall faça tradução de endereços (NAT) para qualquer objeto é utilíssimo e pode poupar bastante trabalho. Obviamente, você pode configurar o NAT manualmente. Dependendo de como for feito, isso pode significar alguma configuração de roteamento adicional no firewall para refletir a estrutura real de NAT da sua rede – em versões anteriores, isso era *sempre* necessário. Por exemplo, pode ser preciso definir uma rota a partir do IP externo "válido", atrás do qual todos vão se esconder, para o IP interno.

Versões posteriores ao HotFix 3 da CheckPoint podem monitorar tráfego de rede em busca de atividade malévola, como o "Ping da Morte" ou comandos SMTP e SMNP ilegais. Esse recurso é chamado de *SmartDefense*, para combinar com os outros produtos da linha "Smart". Fique atento para as novidades nessa área.

Depois de conseguir uma configuração que funcione, podemos passar à árdua tarefa de deixar o sistema mais seguro, já que o firewall não trabalha apenas no nível dos pacotes (camada 3 do modelo OSI) mas também analisa o tráfego dos serviços como um IDS – *Intrusion Detection System* ou *Sistema de Detecção de Intrusos*.

Serviço de tradução

Depois de completar a lista de regras e o trabalho de pré-instalação, é preciso transferir as regras para a estação de administração num formato que os firewalls possam entender. No jargão da CheckPoint, isso é chamado de "compilação". O arquivo de texto com as regras de filtragem, legível para os humanos, é convertido num formato interno denominado *Inspect script*, que por sua vez é reconvertido no chamado *Inspect code*. É esse código que será instalado nos firewalls. *Inspect* é uma linguagem de script criada pela CheckPoint especificamente para seus produtos. Isso permite que os especialistas (e os inimigos das interfaces gráficas) possam fazer as coisas manualmente.

Como as modificações manuais não alteram os objetos mostrados na interface gráfica – o que só acontece durante a compilação – é melhor deixarmos as alterações manuais para os especialistas. O perigo de seu conjunto de regras de filtragem ficar inconsistente é gigantesco,

Firewall por estado de conexão

Essa tecnologia é baseada num princípio assaz simples. Como uma conexão TCP será sempre iniciada por um pacote SYN, o filtro usa as regras para verificar se esse tipo de comunicação é permitida. Se for, a conexão com o cliente é, primeiro, estabelecida, e depois incluída em uma tabela de estados como tal. As tabelas são armazenadas na memória do kernel, permitindo acesso extremamente rápido. O filtro de pacotes pode então usar esses dados para verificar os pacotes seguintes (que não possuem o flag SYN ligado) e decidir se eles pertencem – ou não – a uma conexão autorizada. Em outras palavras, não há necessidade de escrever um punhado de regras para lidar com todos os aspectos da conexão. Por estarem relacionados àquela primeira conexão já listada na tabela, os pacotes são autorizados automaticamente.

A redução do número de regras proporciona um desempenho bem maior para o firewall, permitindo que o sistema reconheça pacotes contíguos, como os pertencentes a uma conexão FTP, que não tenham relação direta com a conexão estabelecida previamente e marcada na tabela. O Linux usa um "ajudante" (*helper*) para esse fim. Esse comportamento também aumenta a segurança porque não há a possibilidade de se abrir múltiplas portas – veja a discussão sobre o assunto em [9].

especialmente se levarmos em conta que a documentação é bastante inadequada – mesmo que tenhamos pago por ela.

O CheckPoint tem seu recurso de propagação proprietário, chamado de *Secure Internal Communication* (SIC) para transferir o conjunto de regras de filtragem aos *Enforcement Modules*. O protocolo é baseado na conhecida técnica SSL / TLS, e usa Criptografia de Chave Pública para verificar a identidade dos nós envolvidos na transação, assegurar a integridade dos dados e criptografar todo o tráfego da rede. Com isso, a confidencialidade e a segurança dos dados é garantida.

O CheckPoint Firewall 1 NG é particularmente útil para aqueles que precisam de uma solução completa, do tipo "satisfação garantida ou seu dinheiro de volta". O produto é particularmente recomendado quando se pretende usar algum tipo de VPN, pois oferece o recurso de gerenciamento centralizado. Se você possui sistemas que precisam de alta disponibilidade, não há muita escolha. Os filtros de pacotes do mundo do Software Livre têm severas restrições no tocante a recursos desse tipo. Um exemplo básico: sincronização de tabelas que funcione direito e seja estável.

Irmão caçula

Se você já trabalhou com o CheckPoint antes, o Firewall Builder [3] o fará sentir-se em casa. Em aparência, é bem semelhante ao *SmartDashboard* do FW-1. A metodologia de separar a administração em uma interface gráfica e manter os firewalls de longe também foi "adotada".

O Firewall Builder permite gerenciar múltiplos firewalls ao mesmo tempo. Há quatro tecnologias de filtragem reconhecidas:

- iptables [4] (Linux com kernel da série 2.4 ou 2.6)
- ipfilter [5] (FreeBSD e NetBSD, com versões para vários outros "sabores" de Unix)
- pf [6] (OpenBSD)
- Cisco Pix

O Cisco Pix possui sua própria interface HTML, mas ela é usada apenas para a configuração inicial. Para que o Firewall Builder possa administrar essa plataforma, é necessário adquirir um módulo adicional, não-livre. Seu preço, entretanto (500 dólares americanos [7]) é baixo o bastante para ser uma pechincha em relação a outros produtos.

A abordagem modular torna o Firewall Builder interessante quando é preciso administrar de forma centralizada firewalls de várias procedências. Também é bastante indicado se um administrador já acostumado com o CheckPoint precisa trocar de plataforma.

O Firewall Builder não só pode trabalhar com múltiplos filtros de pacotes como também rodar em muitos sabores de Unix. Até o momento da edição, havia pacotes para inúmeras distribuições Linux, FreeBSD, OpenBSD e até para o Mac OS X.

Regras para mais de um firewall

Assim como o Checkpoint, o Firewall Builder é dividido em duas partes: o programa de administração, que roda na estação de trabalho do administrador e possui uma interface gráfica, e o firewall propriamente dito, rodando normalmente em máquinas que servem como *gateways*. Em contraste com o Checkpoint, usa um sistema de configuração para criar os conjuntos de regras e então transfere-as para as máquinas onde rodam os firewalls. Não há um mecanismo de transferência próprio, como no Checkpoint, mas o administrador pode usar o bom e velho SSH. Um daemon independente que cuide da transferência das regras, rodando em cada firewall, está no forno para as próximas versões – e isso já se reflete na interface gráfica – mas ainda não está disponível. Como seu "irmão maior", o Firewall Builder usa conexões protegidas por Chaves Públicas para a comunicação entre as máquinas.

A versão atual possui um script chamado *fwv_install*, muito útil ao administrador do firewall. O script faz uso de autenticação SSH via Chave Pública para transferir as regras aos firewalls. Depois de definir o conjunto de regras adequado, basta selecionar a opção *Rules* > *Install* para transferi-lo.

A interface gráfica chama o script *fwb_install* em segundo plano e mostra a saída do comando em uma nova janela. Para evitar redigitação, o sistema ativa o programa *ssh-agent*. O agente armazena todas as suas informações de conexão e senha até que você o desative novamente.

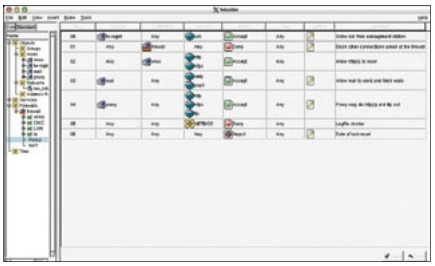

Figura 4: Um conjunto de regras de filtragem criado pelo Firewall Builder, como exemplo de uma possível solução para a rede mostrada na Figura 1.

Portátil, graças ao XML

O administrador pode definir um certo número de opções de transferência para o script. Por exemplo, o local para gravar o arquivo com o conjunto de regras no sistema de arquivos do firewall. Também é possível definir se o arquivo XML usado para "compilar" o conjunto de regras deve ser gravado junto com o arquivo "compilado" ou qual usuário se registrará ("logará") no firewall para a transferência e outras operações.

O procedimento tem a vantagem de evitar que o usuário *root* opere o firewall, propiciando um considerável aumento de segurança. Observe que o conjunto de regras não só é transferido mas também automaticamente ativado pelo script, coisa que requer privilégios de root. Se você definir um usuário com menos privilégios, certifique-se de que sua conta esteja cadastrada no sudo (*/etc/sudoers*).

Da mesma forma que com o *Checkpoint*, o Firewall Builder não trará nenhuma regra quando for executado pela primeira vez. Entretanto, já possui definições para os serviços mais comuns nos protocolos TCP e UDP e mensagens ICMP, além de já conhecer os endereços das LANs privadas. É possível usar um assistente para adicionar os computadores conectados à sua rede. Para detectar os nós de sua rede, o assistente pode – entre outras "espertezas" – ler seu arquivo */etc/hosts*, fazer uma transferência de zona DNS (*zone transfer*), usar SNMP ou simplesmente rodar um *scanner*.

Quando testamos essa função, o programa produziu um erro a cada vez que fazia uma consulta ao DNS. Por outro lado, usando SNMP é possível obter os dados de contato, a localização e até a descrição do objeto em questão. Isso

Figura 5: O Firewall Builder permite uma configuração bastante granular das várias opções de filtragem.

aumenta a base de conhecimento do administrador sobre a rede e pode (deve!) ser usado para documentá-la melhor.

Interfaces semelhantes – plágio?

O assistente incluído com o programa é uma ferramenta bastante útil para administradores inexperientes que precisam construir rapidamente um conjunto de regras inicial. Metódico, cuida das primeiras coisas primeiro: define uma DMZ e usa os dados iniciais para criar sozinho um simples mas eficaz conjunto básico de regras de filtragem. Isso cria uma fundação sólida sobre a qual é fácil construir um firewall, seja ele simples ou complexo. Obviamente, esse assistente não impede que você crie – manualmente, como bom *hacker* – os objetos de que precisar. Também não impede que você use os outros "druidas" disponíveis.

A interface gráfica lembra muito a do Checkpoint, incluindo campos muito semelhantes:

- Objeto de origem (Source);
- Objeto de destino (Target);
- O serviço usado para a comunicação;
- Uma ação a ser tomada caso a regra se aplique a um determinado pacote (liberar, rejeitar, bloquear e contabilizar);
- Definição de registro no *log* se a regra for usada;
- O horário em que essa regra deve estar ativa;
- Um comentário sucinto e informativo, que esclareça a qualquer um o objetivo dessa regra.

Além dessas regras "globais", cada interface de rede do firewall pode ter suas próprias regras, especificando a direção do fluxo de tráfego da rede (entrando, saindo ou ambos). O CheckPoint costumava ter essa opção, mas por algum motivo ela foi retirada da versão 5, a mais recente.

No caso do Firewall Builder, a opção pode ser uma armadilha, pois se aplica a uma única interface de rede no firewall, não à rede que se conecta a essa interface. Por outro lado, a opção permite emular as regras de anti-spoofing do CheckPoint de forma bastante próxima, evitando que potenciais invasores forjem pacotes IP que pareçam vir de sua LAN interna, permitindo que ganhem acesso não autorizado pela interface externa.

O Firewall Builder também permite configurar a tradução de endereços (NAT). Ao contrário do CheckPoint, casos especiais em que um nó interno precisa ser acessado de fora precisam ser roteados manualmente. O programa não oferece esse recurso automaticamente.

Arrastando e soltando regras e objetos

Ambos os produtos permitem que se crie e mova objetos e regras usando o mouse. As mudanças aplicadas em um objeto são imediatamente propagadas para cada instância da regra no conjunto. As diferenças estão nos bastidores. Embora o Firewall Builder não ofereça a mesma gama de funções que o CheckPoint, a grande vantagem da solução livre é sua independência de um filtro de pacotes específico.

Uma solução possível para a rede mencionada no início do artigo (ver figura 1) é mostrada na figura 4. As regras #0 #1 asseguram que a máquina chamada *fw-mgmt* seja a única com permissão de usar SSH para acessar os firewalls, bem como de registrar nos *logs* esses acessos. A regra #2 permite acesso por meio dos protocolos HTTP e HTTPS ao servidor Web que está na DMZ. Como o servidor web possui seus próprios arquivos de log, não ativamos o registro de eventos para ele.

As regras 3 e 4 permitem que o proxy e o servidor de email na rede interna acessem a Internet. Todo e qualquer acesso é registrado. A regra 5 serve apenas para limpar os arquivos de log,, apagando os eventos gerados pelos *broadcasts* NetBIOS enviados à LAN pelas máquinas com Windows. Os arquivos ficariam extremamente atravancados se não fossem limpos regularmente.

Os últimos blocos de regras bloqueiam qualquer acesso não explicitamente permitido por qualquer outra regra. Em contraste com as duas outras regras que bloqueiam acesso, essa regra usa a ação REJECT em vez de DROP. Isso permite que a conexão seja finalizada como se deve, usando a flag RST do protocolo TCP. Se fosse usado DROP, não haveria resposta para pacotes entrantes, o que levaria a um incômodo *timeout* depois de um período pré-determinado de tempo.

Sem complicação!

Obviamente, essas regras podem ser refinadas para incluir recursos, como por exemplo permitir apenas pacotes ICMP Redirect do roteador conectado aos firewalls, possibilitando a detecção de ataques nesse nível. É uma questão de compromisso entre risco e eficiência.

Há uma boa razão para manter as coisas simples quando estiver projetando seu firewall e montando seus conjuntos de regras. Qualquer regra que obrigue o kernel a verificar cada pacote que chegue irá diminuir o desempenho do sistema. Além disso, as regras devem ser inteligíveis para o administrador. Um bom conjunto de regras pode ser inútil se ninguém souber como funciona.

O Firewall Builder armazena o conjunto de regras e os objetos associados em um arquivo XML bem simples. Os compiladores interpretam, então, esse arquivo para gerar as instruções para o firewall apropriado. Usar o XML como formato intermediário permite que os administradores possam olhar o "motor" e descobrir o que o sistema está fazendo enquanto o chefe toma café.

O produto final não tenta esconder o fato de que o programa usa scripts altamente automatizados para cada uma das plataformas de firewall suportadas, poupando trabalho para o administrador, que teria que fazer tudo "na unha". Com a exceção do PIX, todos são shell scripts extremamente espertos que incluem instruções para os filtros de pacotes. Os scripts também detectam endereços dinâmicos (DHCP) e alteram a configuração para que reflita as mudanças.

A configuração abrange alguns detalhes normalmente esquecidos quando escrevemos as mesmas regras manualmente, como por exemplo prevenção de spoofing para todas as interfaces de rede.

Listagem 1: Regras em uso

```
01 [...]
02
03 echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter
04 echo 1 > /proc/sys/net/ipv4/conf/all/log_martians
05 echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
06 echo 0 > /proc/sys/net/ipv4/icmp_echo_ignore_all
07 echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses
08 echo 30 > /proc/sys/net/ipv4/tcp_fin_timeout
09 echo 1800 > /proc/sys/net/ipv4/tcp_keepalive_intvl
10 echo 1 > /proc/sys/net/ipv4/tcp_syncookies
11 [...]
12
13 #  Regra 0(NAT)
14 $IPTABLES -t nat -A POSTROUTING -o eth2 -s 192.168.0.0/24 -j MASQUERADE
15 [...]
16
17 $IPTABLES -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
18 $IPTABLES -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
19 $IPTABLES -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
20 [...]
21
22 # Regra 0(eth2)
23 # Regra para ativar o anti-spoofing
24 $IPTABLES -N eth2_In_RULE_0
25 $IPTABLES -A INPUT -i eth2 -s $interface_eth2 -j eth2_In_RULE_0
26 $IPTABLES -A INPUT -i eth2 -s 192.168.1.1 -j eth2_In_RULE_0
27 $IPTABLES -A INPUT -i eth2 -s 192.168.0.1 -j eth2_In_RULE_0
28 $IPTABLES -A INPUT -i eth2 -s 192.168.0.0/24 -j eth2_In_RULE_0
29 $IPTABLES -A INPUT -i eth2 -s 192.168.1.2 -j eth2_In_RULE_0
30 $IPTABLES -A FORWARD -i eth2 -s $interface_eth2 -j eth2_In_RULE_0
31 $IPTABLES -A FORWARD -i eth2 -s 192.168.1.1 -j eth2_In_RULE_0
32 $IPTABLES -A FORWARD -i eth2 -s 192.168.0.1 -j eth2_In_RULE_0
33 $IPTABLES -A FORWARD -i eth2 -s 192.168.0.0/24 -j eth2_In_RULE_0
34 $IPTABLES -A FORWARD -i eth2 -s 192.168.1.2 -j eth2_In_RULE_0
35 $IPTABLES -A eth2_In_RULE_0 -m limit --limit 10/second -j LOG --log-level info --log-prefix "RULE 0 -- DENY "
36 $IPTABLES -A eth2_In_RULE_0 -j DROP
37 [...]
38
39 # Regra 0(global)
40 # Permite conexão SSH para o firewall vinda apenas da máquina do administrador.
41 $IPTABLES -N RULE_0
42 $IPTABLES -A INPUT -p tcp -s 192.168.0.2 -d $interface_eth2 --destination-port 22 -m state --state NEW  -j RULE_0
43 $IPTABLES -A INPUT -p tcp -s 192.168.0.2 -d 192.168.1.1 --destination-port 22 -m state --state NEW  -j RULE_0
44 $IPTABLES -A INPUT -p tcp -s 192.168.0.2 -d 192.168.0.1 --destination-port 22 -m state --state NEW  -j RULE_0
45 $IPTABLES -A RULE_0 -m limit --limit 10/second -j LOG --log-level info --log-prefix "RULE 0 -- ACCEPT "
46 $IPTABLES -A RULE_0  -j ACCEPT
```

Menos regras, melhor entendimento

O processo cria um conjunto de regras bastante abrangente mas, ao mesmo tempo, pequeno o suficiente para ser facilmente digerido por homens e máquinas. Os recursos do iptables são muito bem usados, incluindo aí as tabelas de estado de conexão e os "atalhos" disponíveis no Netfilter. Regras abrangendo múltiplas portas são apenas um exemplo. Com elas, uma mesma regra pode ser aplicada a vários serviços. Isso evita criar um grande número de regras idênticas, uma para cada porta.

A aba Firewall no objeto de mesmo nome permite que se especifique quantos desses truques podem ser aplicados. Também é possível definir o nível de registro no log para o filtro de pacotes.

A listagem 1 dá alguns exemplos. Todos esses itens estão incluídos no conjunto de regras. Mas preste muita atenção: você precisa, **obrigatoriamente,** saber o que está fazendo! Se agir sem conhecimento de causa, pode acabar escrevendo regras para funções do iptables que sequer estão implementadas.

Ainda na listagem 1, vemos muitos dos conceitos usados pelo Firewall Builder na criação das regras, baseado no que fizemos na interface gráfica. O script não só leva em conta endereços dinâmicos como também usa o sistema de arquivos */proc* para modificar opções e comportamento do kernel e, por exemplo, habilitar recursos como o redirecionamento de IP (*IP forwarding*). Observe as verificações de consistência que permitem ao kernel certificar-se de que o endereço IP foi originado na inter-

face correta (*rp_filter*) e definir temporizações seletivas para algumas conexões.

As regras para NAT ficam no início

O conjunto de regras começa, efetivamente, definindo a tradução de endereços (NAT). Isso evita problemas potenciais na interação entre o NAT e as demais regras. Essa seção é seguida pelas regras que asseguram a permissão das conexões já estabelecidas. Isso simplifica em muito o conjunto de regras. O firewall pode automaticamente reconhecer fluxos de dados relativos a uma conexão já existente, como o FTP, por exemplo, e não precisa de regras adicionais para tratar todas as situações decorrentes.

As regras exclusivas das interfaces de rede (regras de anti-spoofing) são definidas antes das regras "globais". Essa etapa pode ser deixada de fora, se desejado, e o módulo do kernel pode ser usado no lugar. Depois disso temos, enfim, as regras globais. Como o exemplo mostra, uma cadeia é definida para cada regra. Isso simplifica a contabilidade em um estágio posterior, mesmo que o método pareça mais complicado do que realmente é.

A regra #0 (global) no conjunto mostra que o software segue sem desvios o projeto do administrador para o firewall. Mesmo se o administrador quiser usar SSH no lado interno do firewall, aplicar essa regra ao objeto firewall abriria a porta 22 em todas as interfaces – incluindo aí a Internet.

Se você estiver interessado em como essas regras funcionam na prática, recomendamos estudar com afinco a configuração gerada por nossa rede de exemplo. Isso irá melhorar sua curva de aprendizado.

Especialidades do netfilter

Permitir a criação de serviços personalizados é mais uma dentre as muitas funções interessantes do Firewall Builder. Com esse truque, torna-se possível usar opções altamente específicas do filtro de pacotes que está sendo configurado. Em particular, uma combinação das extensões do Netfilter/iptables pode oferecer um número bastante grande de recursos avançados em comparação com o filtro de pacotes do kernel padrão.

Um exemplo é o *string patch* [8], que permite analisar o fluxo de dados à procura de uma cadeia de caracteres específica e aplica regras específicas a eles. Você pode, por exemplo, procurar pela chamada ao *cmd.exe* que vários worms na selva da Internet emitem – e que infestam os logs do Apache. O patch evita que a chamada sequer chegue a seu servidor Web, sendo bloqueada no firewall.

A opção de definir regras especiais para cada uma das interfaces de rede é outro recurso inteligente que permite aprimorar o efeito das regras no desempenho global do firewall. Muitas opções são aplicadas a um objeto em particular por padrão, sem que o administrador tenha de definir explicitamente – um exemplo clássico é o bloqueio de pacotes com roteamento na origem (source routing). Você pode comparar essa funcionalidade com as regras implícitas do CheckPoint. Dependendo da plataforma de firewall usada, o produto livre pode até mesmo oferecer mais opções de configuração que o CheckPoint.

Conclusão

O Firewall Builder é uma escolha particularmente interessante para administradores que não precisam das opções avançadas dos produtos comerciais, ou não podem usá-los devido ao seu parque heterogêneo de plataformas de firewall.

O Firewall Builder permite que os administradores com pouca experiência em configuração de filtros de pacotes criem conjuntos de regras com um mínimo de esforço. O fato de sua configuração ser aplicada de forma transparente permite que se aprenda à medida que o firewall fica mais complexo. ■

Mais front-ends: GuardDog e KNetFilter

O GuardDog é um configurador de iptables construído com a biblioteca Qt. É muito interessante para usuários com pouca ou nenhuma experiência em redes e protocolos. Permite que o usuário defina zonas em sua rede e escolha os serviços permitidos para cada zona. O GuardDog implementa a maioria dos protocolos mais comuns – particularmente em ambiente doméstico.

Embora esse tipo de firewall pessoal possa ser uma boa idéia para computadores de mesa, mostra suas limitações quando precisa ser aplicado a cenários mais complexos. Felizmente, o padrão desse programa é bloquear tudo o que não for explicitamente permitido – na medida certa para usuários inexperientes, evitando que permitam acesso ilimitado de crackers à sua máquina. Para mais informações sobre o GuardDog, veja a matéria sobre ele nesta edição, à página 20.

O Knetfilter é outro front-end para o iptables que, além disso, usa o tcpdump e o Nmap para monitoramento. Também oferece recursos de QoS – Quality of Service (qualidade de serviço). O programa espera que o administrador defina todas as regras sem qualquer ajuda. Não há serviços pré-definidos. Aliás, os serviços são distinguidos apenas por suas portas de origem e destino. O administrador obrigatoriamente tem que saber os números de cabeça ou providenciar um papel com uma "colinha".

Se isso não for problema, o KNetfilter é capaz de criar regras extremamente granulares, levando em conta, inclusive, informações vindas de mensagens ICMP ou do estado das conexões. O KNetfilter não é, definitivamente, um brinquedo para iniciantes, pois requer conhecimentos mais ou menos profundos a respeito do iptables. Administradores experientes, entretanto, vão amar o poder de configuração e o nível extremo de detalhe que a ferramenta oferece, embora seja necessário um chope ou dois para se acostumar com a interface gráfica.

INFORMAÇÕES

[1] CheckPoint: *http://www.checkpoint.com/products/protect/firewall-1.html*

[2] Lista de discussão dos "Gurus" do FireWall-1: *http://www.phoneboy.com/staticpages/index.php?page=20030517034933897*

[3] Firewall Builder: *http://www.fwbuilder.org*

[4] iptables, filtro de pacotes do Linux: *http://www.netfilter.org*

[5] ipfilter, filtro de pacotes do Free/NetBSD: *http://coombs.anu.edu.au/~avalon/ip-filter.html*

[6] pf, filtro de pacotes do OpenBSD: *http://www.benzedrine.cx/pf.html*

[7] Compilador PIX para o Firewall Builder: *http://www.netcitadel.com/pix.htm*

[8] Patch para o iptables: *http://www.netfilter.org/documentation/pomlist/pom-extra.html#string*

[9] Anatomia de um firewall por estado de conexão: *http://www.giac.org/practical/gsec/Lisa_Senner_GSEC.pdf*

[10] Tabela de estados do Firewall-1: *http://www.spitzner.net/fwtable.html*

[11] Listas de discussão da Checkpoint: *http://msgs.securepoint.com/fw1/*